



# **QGIS Desktop 3.34 User Guide**

**QGIS Project**

**02.05.2024**





---

## Inhaltsverzeichnis

---

<b>1</b>	<b>Präambel</b>	<b>3</b>
1.1	Was ist neu in QGIS 3.34 . . . . .	4
<b>2</b>	<b>Vorwort</b>	<b>5</b>
<b>3</b>	<b>Gebrauch der Dokumentation</b>	<b>7</b>
3.1	GUI Schreibstile . . . . .	7
3.2	Text oder Tastatur Schreibstile . . . . .	8
3.3	Betriebssystemspezifische Anweisungen . . . . .	8
<b>4</b>	<b>Funktionalitäten</b>	<b>9</b>
4.1	Daten visualisieren . . . . .	9
4.2	Daten erkunden, abfragen und Karten layouts . . . . .	10
4.3	Daten erstellen, editieren, verwalten und exportieren . . . . .	10
4.4	Daten analysieren . . . . .	11
4.5	Karten im Internet veröffentlichen . . . . .	11
4.6	Erweiterte QGIS Funktionalität durch Erweiterungen . . . . .	11
4.6.1	Kern-Plugins . . . . .	11
4.6.2	Externe Python Plugins . . . . .	12
4.7	Python-Konsole . . . . .	12
4.8	Bekannte Probleme . . . . .	12
4.8.1	Begrenzung der Anzahl von geöffneten Dateien . . . . .	12
<b>5</b>	<b>Erste Schritte</b>	<b>13</b>
5.1	QGIS installieren . . . . .	13
5.1.1	Installation mit Binärdateien . . . . .	13
5.1.2	Kompilieren des Quellcodes . . . . .	13
5.1.3	Auf externen Medien installieren . . . . .	14
5.1.4	Beispieldaten herunterladen . . . . .	14
5.2	QGIS starten und beenden . . . . .	14
5.3	Beispielsitzung: Laden von Raster- und Vektorlayern . . . . .	15
<b>6</b>	<b>Arbeiten mit Projektdateien</b>	<b>23</b>
6.1	Vorstellung von QGIS-Projekten . . . . .	23
6.2	Behandlung von zerstörten Dateipfaden . . . . .	25
6.3	Ausgaben erzeugen . . . . .	26
<b>7</b>	<b>QGIS GUI</b>	<b>27</b>
7.1	Menüleiste . . . . .	28
7.1.1	Projekt . . . . .	28
7.1.2	Bearbeiten . . . . .	29

7.1.3	Ansicht . . . . .	33
7.1.4	Layer . . . . .	35
7.1.5	Einstellungen . . . . .	38
7.1.6	Erweiterungen . . . . .	38
7.1.7	Vektor . . . . .	38
7.1.8	Raster . . . . .	40
7.1.9	Datenbank . . . . .	42
7.1.10	Web . . . . .	42
7.1.11	Netz . . . . .	42
7.1.12	Verarbeitung . . . . .	43
7.1.13	Hilfe . . . . .	43
7.1.14	QGIS . . . . .	43
7.2	Bedienfelder und Werkzeugkästen . . . . .	44
7.2.1	Werkzeugleisten . . . . .	44
7.2.2	Bedienfelder . . . . .	45
7.3	Statusleiste . . . . .	45
7.3.1	Suchleiste . . . . .	45
7.3.2	Anzeige von Aktionen . . . . .	46
7.3.3	Steuerung der Kartenansicht . . . . .	46
7.3.4	Meldungen . . . . .	47
<b>8</b>	<b>Das Browser Bedienfeld</b> . . . . .	<b>49</b>
8.1	Ressourcen, die über den Browser geöffnet/ausgeführt werden können . . . . .	52
8.2	Die obersten Einträge des Brower-Bedienfelds . . . . .	52
8.2.1	Favoriten . . . . .	52
8.2.2	Räumliche Lesezeichen . . . . .	52
8.2.3	Projektverzeichnis . . . . .	53
8.2.4	Laufwerke und Dateisystem . . . . .	53
8.2.5	Datenbankeinträge . . . . .	54
8.2.6	Kacheln und Internetdienste . . . . .	57
8.3	Datenquellen und Dateien . . . . .	57
<b>9</b>	<b>QGIS anpassen</b> . . . . .	<b>59</b>
9.1	Optionen . . . . .	59
9.1.1	Allgemein . . . . .	60
9.1.2	System . . . . .	62
9.1.3	User Profiles Settings . . . . .	64
9.1.4	KBS und Transformation . . . . .	64
9.1.5	Datenquellen . . . . .	68
9.1.6	Darstellung . . . . .	72
9.1.7	Karte & Legende . . . . .	76
9.1.8	Kartenwerkzeuge . . . . .	78
9.1.9	Elevation settings . . . . .	82
9.1.10	3D settings . . . . .	82
9.1.11	Colors settings . . . . .	83
9.1.12	Fonts Settings . . . . .	84
9.1.13	Layouts settings . . . . .	85
9.1.14	Variables settings . . . . .	86
9.1.15	Authentication settings . . . . .	86
9.1.16	Network settings . . . . .	87
9.1.17	GPS settings . . . . .	89
9.1.18	Locator settings . . . . .	91
9.1.19	Acceleration settings . . . . .	92
9.1.20	IDE settings . . . . .	93
9.1.21	Processing settings . . . . .	96
9.1.22	Advanced settings . . . . .	97
9.2	Working with User Profiles . . . . .	98
9.2.1	The concept . . . . .	98

9.2.2	Setting user profile	99
9.3	Projekteigenschaften	100
9.3.1	General Properties	100
9.3.2	Eigenschaften Metadaten	102
9.3.3	View Settings	103
9.3.4	CRS Properties	103
9.3.5	Transformations Properties	104
9.3.6	Styles Properties	104
9.3.7	Data Sources Properties	105
9.3.8	Relations Properties	107
9.3.9	Eigenschaften Variablen	108
9.3.10	Macros Properties	108
9.3.11	Eigenschaften QGIS Server	109
9.3.12	Eigenschaften Zeitlich	110
9.3.13	Terrain Properties	110
9.3.14	Sensors Properties	112
9.4	Anpassung	113
9.5	Tastenkürzel	115
9.6	Running QGIS with advanced settings	116
9.6.1	Command line and environment variables	116
9.6.2	Deploying QGIS within an organization	121
<b>10</b>	<b>Arbeiten mit Projektionen</b>	<b>123</b>
10.1	Überblick zur Projektionsunterstützung	123
10.2	Layer Coordinate Reference Systems	123
10.3	Project Coordinate Reference Systems	125
10.4	Coordinate Reference System Selector	127
10.5	Eigenes Koordinatenbezugssystem definieren	128
10.5.1	Integrate an NTV2-transformation in QGIS	129
10.6	Datum Transformations	130
<b>11</b>	<b>Visualizing Maps</b>	<b>133</b>
11.1	2D Map View	133
11.1.1	Funktionen des Kartenfensters	134
11.1.2	Controlling map rendering	135
11.1.3	Zeitabhängige Darstellung in der Kartenansicht	136
11.1.4	Bookmarking extents on the map	138
11.1.5	Decorating the map	140
11.1.6	Beschriftungstools	148
11.1.7	Messen	154
11.1.8	Weitere Kartenansichten hinzufügen	156
11.1.9	Exportieren der Kartenansicht	157
11.2	3D-Ansicht	160
11.2.1	Scene Configuration	161
11.2.2	Navigation options	168
11.2.3	Creating an animation	168
11.2.4	3D vector layers	169
11.3	Elevation Profile View	169
11.3.1	The interface	170
11.3.2	Creating an elevation profile	171
11.3.3	Interacting with the profile Views	172
<b>12</b>	<b>Allgemeine Werkzeuge</b>	<b>175</b>
12.1	Hilfe	175
12.2	Bedienfelder	175
12.2.1	Layerfenster	175
12.2.2	Layer Styling Panel	182
12.2.3	Layer Order Panel	185
12.2.4	Overview Panel	185

12.2.5	Log Messages Panel . . . . .	186
12.2.6	Undo/Redo Panel . . . . .	186
12.2.7	Statistical Summary Panel . . . . .	186
12.2.8	Debugging/Development Tools Panel . . . . .	187
12.3	Embedding layers from external projects . . . . .	190
12.4	Interacting with features . . . . .	191
12.4.1	Selecting features . . . . .	191
12.4.2	Identifying Features . . . . .	194
12.5	Save and Share Layer Properties . . . . .	199
12.5.1	Verwalten von benutzerdefinierten Stilen . . . . .	199
12.5.2	Storing Styles in a File or a Database . . . . .	200
12.5.3	Layer definition file . . . . .	202
12.6	Documenting your data . . . . .	202
12.6.1	Metadata . . . . .	202
12.6.2	Bemerkungen zu Layern . . . . .	203
12.7	Storing values in Variables . . . . .	204
12.8	Authentifizierung . . . . .	205
12.9	Common widgets . . . . .	206
12.9.1	Farbauswahl . . . . .	206
12.9.2	Symbol Widget . . . . .	210
12.9.3	Remote or embedded file selector . . . . .	210
12.9.4	Visibility Scale Selector . . . . .	211
12.9.5	Spatial Extent Selector . . . . .	211
12.9.6	Font Selector . . . . .	212
12.9.7	Unit Selector . . . . .	213
12.9.8	Number Formatting . . . . .	214
12.9.9	Mischmodi . . . . .	215
12.9.10	Datendefinierte Übersteuerung Setup . . . . .	217
<b>13</b>	<b>Level up mit Ausdrücken</b>	<b>221</b>
13.1	Ausdrücke . . . . .	221
13.1.1	Der Ausdruckseditor . . . . .	221
13.1.2	Funktions Editor . . . . .	226
13.2	Liste der Funktionen . . . . .	228
13.2.1	Aggregatsfunktionen . . . . .	228
13.2.2	Matrixfunktionen . . . . .	238
13.2.3	Farbfunktionen . . . . .	249
13.2.4	Funktionen für Bedingungen . . . . .	256
13.2.5	Funktionen zum Umwandlung . . . . .	258
13.2.6	Benutzerdefinierte Funktionen . . . . .	263
13.2.7	Datum und Zeit Funktionen . . . . .	263
13.2.8	Felder und Werte . . . . .	274
13.2.9	Files and Paths Functions . . . . .	274
13.2.10	Form Functions . . . . .	277
13.2.11	Unschärfer Vergleich Funktion . . . . .	277
13.2.12	Allgemeine Funktionen . . . . .	279
13.2.13	Geometriefunktionen . . . . .	281
13.2.14	Layout Funktionen . . . . .	343
13.2.15	Kartenlayer . . . . .	344
13.2.16	Kartenfunktionen . . . . .	346
13.2.17	Mathematische Funktionen . . . . .	350
13.2.18	Meshes Functions . . . . .	358
13.2.19	Operatoren . . . . .	360
13.2.20	Processing Functions . . . . .	369
13.2.21	Rasterfunktionen . . . . .	369
13.2.22	Funktionen zu Datensätzen und Attributen . . . . .	371
13.2.23	Relations . . . . .	380
13.2.24	Sensors Functions . . . . .	380

13.2.25	Zeichenkettenfunktionen	380
13.2.26	User Expressions	389
13.2.27	Variables	390
13.2.28	Kürzlich verwendete Funktionen	393
<b>14</b>	<b>Die Stilbibliothek</b>	<b>395</b>
14.1	The Style Manager	395
14.1.1	The Style Manager dialog	395
14.1.2	Setting a Color Ramp	401
14.1.3	Creating a Legend Patch Shape	403
14.2	The Symbol Selector	405
14.2.1	The symbol layer tree	406
14.2.2	Configuring a symbol	407
14.3	Setting a label	424
14.3.1	Formatting the label text	426
14.3.2	Configuring interaction with labels	434
14.4	Creating 3D Symbols	443
14.4.1	Point Layers	443
14.4.2	Line layers	444
14.4.3	Polygon Layers	445
14.4.4	Shading the texture	447
14.4.5	Application example	447
<b>15</b>	<b>Verwaltung von Datenquellen</b>	<b>449</b>
15.1	Öffnen von Daten	449
15.1.1	The Browser Panel	451
15.1.2	The DB Manager	454
15.1.3	Provider-based loading tools	455
15.1.4	The Layer Metadata Search Panel	475
15.1.5	QGIS Custom formats	476
15.1.6	QLR - QGIS Layer Definition File	476
15.1.7	Connecting to web services	476
15.2	Erstellung von Layern	482
15.2.1	Neue Vektorlayer erstellen	482
15.2.2	Creating new layers from an existing layer	490
15.2.3	Creating new DXF files	494
15.2.4	Creating new layers from the clipboard	495
15.2.5	Creating virtual layers	495
15.3	Exploring Data Formats and Fields	498
15.3.1	Raster data	498
15.3.2	Vektordaten	498
15.3.3	Layers crossing 180° longitude	506
<b>16</b>	<b>Arbeiten mit Vektordaten</b>	<b>509</b>
16.1	Vektorlayereigenschaften	509
16.1.1	Eigenschaften Information	510
16.1.2	Eigenschaften Quelle	510
16.1.3	Eigenschaften Symbolisierung	513
16.1.4	Labels Properties	535
16.1.5	Diagrammeigenschaften	546
16.1.6	Eigenschaften Masken	550
16.1.7	3D-Ansicht Eigenschaften	551
16.1.8	Eigenschaften Felder	553
16.1.9	Eigenschaften Attributformular	554
16.1.10	Eigenschaften Verknüpfungen	562
16.1.11	Auxiliary Storage Properties	563
16.1.12	Eigenschaften Aktionen	571
16.1.13	Eigenschaften Anzeige	576
16.1.14	Eigenschaften Darstellung	579

16.1.15	Eigenschaften Zeitlich	581
16.1.16	Eigenschaften Variablen	582
16.1.17	Elevation Properties	582
16.1.18	Eigenschaften Metadaten	584
16.1.19	Eigenschaften Abhängigkeiten	584
16.1.20	Eigenschaften Legende	584
16.1.21	Eigenschaften QGIS Server	585
16.1.22	Eigenschaften Digitalisieren	585
16.2	Mit Attributtabelle arbeiten	588
16.2.1	Foreword: Spatial and non-spatial tables	588
16.2.2	Introducing the attribute table interface	589
16.2.3	Interacting with features in an attribute table	594
16.2.4	Using action on features	596
16.2.5	Editieren Attributwerte	598
16.2.6	Exploring features attributes through the Identify Tool	602
16.2.7	Storing and fetching an external resource	603
16.3	Editierfunktionen	604
16.3.1	Einrasttoleranz und Suchradius einstellen	605
16.3.2	Optionen für das Einrasten und Digitalisieren	605
16.3.3	Topologisches Editieren	608
16.3.4	Einen vorhandenen Layer editieren	610
16.3.5	Erweiterte Digitalisierung	620
16.3.6	Formen digitalisieren	629
16.3.7	Das Bedienfeld Erweiterte Digitalisierung	632
16.3.8	Mit Verarbeitung Objekte innerhalb eines Layers verändern	640
16.4	Connecting and Editing Data Across Layers	642
16.4.1	Joining features between two layers	642
16.4.2	Setting relations between multiple layers	644
<b>17</b>	<b>Arbeiten mit Rasterdaten</b>	<b>659</b>
17.1	Dialogfenster Rasterlayereigenschaften	659
17.1.1	Eigenschaften Information	660
17.1.2	Eigenschaften Quelle	660
17.1.3	Eigenschaften Symbolisierung	661
17.1.4	Transparency Properties	671
17.1.5	Histogram Properties	672
17.1.6	Eigenschaften Darstellung	673
17.1.7	Eigenschaften Zeitlich	674
17.1.8	Pyramids Properties	675
17.1.9	Elevation Properties	677
17.1.10	Eigenschaften Metadaten	678
17.1.11	Eigenschaften Legende	678
17.1.12	Display Properties	679
17.1.13	Eigenschaften QGIS Server	680
17.1.14	Identify raster cells	681
17.2	Rasteranalyse	682
17.2.1	Rasterrechner	682
17.3	Georeferencer	685
17.3.1	Wie benutzt man den Georeferenzierer	685
<b>18</b>	<b>Working with Mesh Data</b>	<b>691</b>
18.1	What's a mesh?	691
18.2	Supported formats	693
18.3	Mesh Dataset Properties	694
18.3.1	Information Properties	694
18.3.2	Source Properties	695
18.3.3	Symbolology Properties	696
18.3.4	3D View Properties	701

18.3.5	Rendering Properties	703
18.3.6	Temporal Properties	703
18.3.7	Elevation Properties	704
18.3.8	Metadata Properties	705
18.4	Editing a mesh layer	705
18.4.1	Overview of the mesh digitizing tools	706
18.4.2	Exploring the Z value assignment logic	706
18.4.3	Selecting mesh elements	707
18.4.4	Modifying mesh elements	708
18.4.5	Reindexing meshes	712
18.5	Mesh Calculator	712
<b>19</b>	<b>Working with Vector Tiles</b>	<b>715</b>
19.1	What are Vector Tiles?	715
19.2	Supported Formats	716
19.3	Vector Tiles Dataset Properties	717
19.3.1	Information Properties	717
19.3.2	Source Properties	717
19.3.3	Symbology and Labels Properties	718
19.3.4	Rendering Properties	719
19.3.5	Metadata Properties	720
<b>20</b>	<b>Working with Point Clouds</b>	<b>721</b>
20.1	Introduction to Point Clouds	721
20.2	Point Clouds Properties	722
20.2.1	Eigenschaften Information	722
20.2.2	Eigenschaften Quelle	723
20.2.3	Eigenschaften Symbolisierung	726
20.2.4	3D-Ansicht Eigenschaften	731
20.2.5	Eigenschaften Darstellung	733
20.2.6	Elevation Properties	733
20.2.7	Eigenschaften Metadaten	734
20.2.8	Statistics Properties	734
20.3	Virtual point cloud	735
<b>21</b>	<b>Working with 3D Tiles</b>	<b>739</b>
21.1	What are 3D Tiles?	739
21.2	Data Sources	740
21.2.1	Using data from Cesium ion	740
21.2.2	Using Google 3D data	741
21.3	3D Tiles Properties	741
21.3.1	Information Properties	741
21.3.2	Source Properties	742
21.3.3	Symbology Properties	742
21.3.4	3D View Properties	743
21.3.5	Rendering Properties	744
21.3.6	Elevation Properties	744
21.3.7	Metadata Properties	744
<b>22</b>	<b>Kartenlayout</b>	<b>745</b>
22.1	Überblick über das Drucklayout	745
22.1.1	Sample Session for beginners	745
22.1.2	Layoutverwaltung	746
22.1.3	Menüs, Werkzeuge und Bedienfelder des Drucklayouts	747
22.2	Layout-Elemente	761
22.2.1	Layout Items Common Options	761
22.2.2	Das Kartenelement	766
22.2.3	Das 3D Kartenelement	776
22.2.4	Das Beschriftungselement	777

22.2.5	Das Legendenelement	781
22.2.6	Das Maßstabselement	789
22.2.7	Das Tabellenelement	793
22.2.8	The Marker, Picture and North Arrow Items	802
22.2.9	The Elevation Profile Item	806
22.2.10	Das HTML-Rahmen Element	807
22.2.11	Das Formelement	811
22.3	Eine Ausgabe erzeugen	814
22.3.1	Exporteinstellungen	814
22.3.2	Speichern als Rasterbild	815
22.3.3	Als SVG exportieren	816
22.3.4	Als PDF exportieren	817
22.3.5	Einen Atlas erzeugen	818
22.4	Creating a Report	824
22.4.1	What is it?	824
22.4.2	Get started	824
22.4.3	Layout Report Workspace	825
22.4.4	Exporteinstellungen	840
<b>23</b>	<b>Arbeiten mit OGC / ISO-Protokollen</b>	<b>841</b>
23.1	WMS/WMTS Client	842
23.1.1	Übersicht über die WMS-Unterstützung	842
23.1.2	Überblick über die WMTS Unterstützung	842
23.1.3	WMS/WMTS Server auswählen	843
23.1.4	WMS/WMTS Layer laden	845
23.1.5	Tilesets	847
23.1.6	Das Objekte abfragen Werkzeug	848
23.1.7	Viewing Properties	848
23.1.8	Show WMS legend graphic in table of contents and layout	852
23.2	WCS Client	853
23.3	WFS und WFS-T Klient	853
<b>24</b>	<b>Arbeiten mit GPS Daten</b>	<b>857</b>
24.1	Introducing GNSS/GPS Data	857
24.1.1	Was ist GPS?	857
24.1.2	Transferring or loading GPS data	857
24.2	Live GPS tracking	860
24.2.1	GPS toolbar	860
24.2.2	GPS Information Panel	861
24.2.3	Ein Bluetooth GPS fürs Live Tracking anbinden	863
24.2.4	Examples of GPS devices connection in QGIS	864
<b>25</b>	<b>Authentifizierungssystem</b>	<b>867</b>
25.1	Authentifizierungssystem Übersicht	867
25.1.1	Authentifikationsdatenbank	868
25.1.2	Hauptpasswort	868
25.1.3	Authentifikationskonfiguration	869
25.1.4	Authentifizierungsmethoden	872
25.1.5	Hauptpasswort und Auth Konfig Hilfsmittel	876
25.1.6	Gebrauche Authentifikationskonfiguration	877
25.1.7	Python-Bindung	878
25.2	Benutzerauthentifizierung Workflows	878
25.2.1	HTTP(S) Authentifikation	878
25.2.2	Datenbank Authentifikation	879
25.2.3	PKI Authentifikation	880
25.2.4	Defekte Layer behandeln	887
25.2.5	Kennung der Authentifikationskonfiguration ändern	888
25.2.6	QGIS Server Unterstützung:	889
25.2.7	SSL Serverausnahme	889



25.3	Sicherheitsüberprüfung . . . . .	892
25.3.1	Einschränkungen . . . . .	893
<b>26</b>	<b>GRASS GIS Integration</b>	<b>895</b>
26.1	Beispieldatensatz . . . . .	895
26.2	GRASS Layer visualisieren . . . . .	895
26.3	Daten in eine GRASS LOCATION importieren via drag and drop . . . . .	896
26.4	GRASS Daten im QGIS Browser verwalten . . . . .	896
26.5	GRASS Optionen . . . . .	896
26.6	GRASS Plugin starten . . . . .	896
26.7	GRASS Mapset öffnen . . . . .	897
26.8	Information zur GRASS-Datenbank . . . . .	897
26.9	Daten in eine GRASS LOCATION importieren . . . . .	898
26.9.1	Eine neue GRASS LOCATION erstellen . . . . .	898
26.9.2	Eine neue GRASS MAPSET erstellen . . . . .	900
26.10	Das GRASS Vektormodell . . . . .	900
26.11	Einen neuen GRASS Vektorlayer erstellen . . . . .	901
26.12	Digitalisieren und Editieren eines GRASS Vektorlayers . . . . .	901
26.13	Einstellung der GRASS Region . . . . .	903
26.14	Die GRASS Werkzeugkiste . . . . .	904
26.14.1	Arbeiten mit GRASS Modulen . . . . .	904
26.14.2	GRASS Beispielanwendung . . . . .	908
26.14.3	Anpassen der Module . . . . .	912
<b>27</b>	<b>QGIS Verarbeitung Umgebung</b>	<b>913</b>
27.1	Einführung . . . . .	913
27.2	Configuring the Processing Framework . . . . .	916
27.2.1	General . . . . .	916
27.2.2	Menus . . . . .	918
27.2.3	Models and Scripts . . . . .	918
27.2.4	Providers . . . . .	918
27.3	The Toolbox . . . . .	918
27.3.1	Der Algorithmus Dialog . . . . .	920
27.3.2	Von Algorithmen erstellte Datenobjekte . . . . .	927
27.4	Das Protokoll . . . . .	928
27.4.1	Das Verarbeitung Protokoll . . . . .	928
27.4.2	Das Verarbeitung Log . . . . .	929
27.5	The model designer . . . . .	930
27.5.1	The model designer interface . . . . .	930
27.5.2	Creating a model . . . . .	933
27.5.3	Speichern und laden von Modellen . . . . .	943
27.5.4	Ein Modell editieren . . . . .	944
27.6	Die Batch Processing Schnittstelle . . . . .	945
27.6.1	Einführung . . . . .	945
27.6.2	Die Parameter-Tabelle . . . . .	946
27.6.3	Füllen der Parameter-Tabelle . . . . .	947
27.6.4	Ausführen eines Batch-Prozesses . . . . .	948
27.7	Verarbeitung Algorithmen von der Konsole aus verwenden . . . . .	948
27.7.1	Algorithmen von der Python Konsole aus aufrufen . . . . .	949
27.7.2	Scripte erstellen und diese aus der Werkzeugkiste starten. . . . .	953
27.7.3	Pre- und Post-execution Script Hooks . . . . .	956
27.8	Using processing from the command line . . . . .	956
27.9	Writing new Processing algorithms as Python scripts . . . . .	958
27.9.1	Extending QgsProcessingAlgorithm . . . . .	958
27.9.2	The @alg decorator . . . . .	963
27.9.3	Input and output types for Processing Algorithms . . . . .	964
27.9.4	Handing algorithm output . . . . .	967
27.9.5	Die Kommunikation mit dem Benutzer . . . . .	967

27.9.6	Ihre Skripte dokumentieren . . . . .	967
27.9.7	Flags . . . . .	967
27.9.8	Bewährte Verfahren für das Schreiben von Skript-Algorithmen . . . . .	968
27.10	Konfiguration externer Anwendungen . . . . .	968
27.10.1	Ein Hinweis für Windows Anwender . . . . .	968
27.10.2	Eine Bemerkung zu den Datenformaten . . . . .	968
27.10.3	Eine Bemerkung zu Vektorlayern mit ausgewählten Objekten . . . . .	969
27.10.4	Using third-party Providers . . . . .	969
<b>28</b>	<b>Datenanbieter und Algorithmen in Verarbeitung</b>	<b>981</b>
28.1	GDAL Algorithmus-Anbieter . . . . .	981
28.1.1	3D Tiles . . . . .	981
28.1.2	Cartography . . . . .	983
28.1.3	Database . . . . .	1004
28.1.4	File tools . . . . .	1012
28.1.5	GPS . . . . .	1013
28.1.6	Interpolation . . . . .	1016
28.1.7	Layer tools . . . . .	1028
28.1.8	Netz . . . . .	1031
28.1.9	Modellierungswerkzeuge . . . . .	1043
28.1.10	Network analysis . . . . .	1052
28.1.11	Plots . . . . .	1064
28.1.12	Point Cloud Conversion . . . . .	1070
28.1.13	Point Cloud Data Management . . . . .	1076
28.1.14	Point Cloud Extraction . . . . .	1089
28.1.15	Raster analysis . . . . .	1094
28.1.16	Raster Creation . . . . .	1147
28.1.17	Raster terrain analysis . . . . .	1165
28.1.18	Raster tools . . . . .	1178
28.1.19	Vektoranalyse . . . . .	1188
28.1.20	Vector creation . . . . .	1214
28.1.21	Vector general . . . . .	1239
28.1.22	Vector geometry . . . . .	1277
28.1.23	Vector overlay . . . . .	1391
28.1.24	Vector selection . . . . .	1409
28.1.25	Vector table . . . . .	1426
28.1.26	Vector Tiles . . . . .	1442
28.2	GDAL-Algorithmen Bereitstellung . . . . .	1446
28.2.1	Raster analysis . . . . .	1446
28.2.2	Raster conversion . . . . .	1476
28.2.3	Raster extraction . . . . .	1484
28.2.4	Raster miscellaneous . . . . .	1493
28.2.5	Rasterprojektionen . . . . .	1513
28.2.6	Vector conversion . . . . .	1518
28.2.7	Vector geoprocessing . . . . .	1524
28.2.8	Vector miscellaneous . . . . .	1533
28.3	OTB Anwendungen Bereitstellung . . . . .	1542
<b>29</b>	<b>Plugins</b>	<b>1543</b>
29.1	QGIS Plugins . . . . .	1543
29.1.1	Kernerweiterungen und externe Erweiterungen . . . . .	1543
29.1.2	Der Erweiterungen Dialog . . . . .	1544
29.2	Using QGIS Core Plugins . . . . .	1547
29.2.1	DB Manager Plugin . . . . .	1547
29.2.2	Geometry Checker Plugin . . . . .	1550
29.2.3	MetaSearch Catalog Client . . . . .	1554
29.2.4	Offline Editing Plugin . . . . .	1561
29.2.5	Topology Checker Plugin . . . . .	1563

29.3	QGIS Python Konsole . . . . .	1565
29.3.1	The Interactive Console . . . . .	1566
29.3.2	The Code Editor . . . . .	1568
<b>30</b>	<b>Hilfe und Support</b>	<b>1571</b>
30.1	Mailinglisten . . . . .	1571
30.1.1	QGIS Users . . . . .	1571
30.1.2	QGIS Developers . . . . .	1571
30.1.3	QGIS Community Team . . . . .	1571
30.1.4	QGIS Translations . . . . .	1572
30.1.5	QGIS Project Steering Committee (PSC) . . . . .	1572
30.1.6	QGIS-Anwendergruppen . . . . .	1572
30.2	Matrix / IRC . . . . .	1572
30.3	Kommerzieller Support . . . . .	1572
30.4	BugTracker . . . . .	1573
30.5	Blog . . . . .	1573
30.6	Plugins . . . . .	1573
30.7	Wiki . . . . .	1573
<b>31</b>	<b>Mitwirkende</b>	<b>1575</b>
31.1	Autoren . . . . .	1575
31.2	Übersetzer . . . . .	1576
31.3	Übersetzungsstatistiken . . . . .	1577
<b>32</b>	<b>Anhänge</b>	<b>1579</b>
32.1	Anhang A: GNU General Public License . . . . .	1579
32.2	Anhang B: GNU Free Documentation License . . . . .	1583
32.3	Anhang C: QGIS-Dateiformate . . . . .	1589
32.3.1	QGS/QGZ - Das QGIS-Projekt-Dateiformat . . . . .	1589
32.3.2	QLR – die QGIS-Layerdefinitionsdatei . . . . .	1591
32.3.3	QML - Das QGIS-Stil-Dateiformat . . . . .	1592
32.4	Anhang D: QGIS R Skriptsyntax . . . . .	1593
32.4.1	Eingaben . . . . .	1594
32.4.2	Ausgaben . . . . .	1594
32.4.3	Syntax Summary for QGIS R scripts . . . . .	1594
32.4.4	Beispiele . . . . .	1596
32.5	Anhang E: Netzwerkverbindungen der QGIS-Anwendung . . . . .	1598
<b>33</b>	<b>Literatur und Internetreferenzen</b>	<b>1601</b>







Dies ist das Benutzerhandbuch für das geografische Informationssystem (GIS) QGIS. QGIS unterliegt der GNU General Public License. Weitere Informationen finden Sie auf der QGIS-Homepage unter <https://www.qgis.org>.

Der Inhalt dieses Dokuments wurde nach bestem Wissen der Autoren und Redakteure verfasst und überprüft. Trotzdem sind Fehler möglich.

Daher übernehmen die Autoren, Bearbeiter und Herausgeber keine Verantwortung oder Haftung für Fehler in diesem Dokument und deren mögliche Folgen. Wir empfehlen Ihnen, mögliche Fehler zu melden.

Dieses Dokument wurde mit reStructuredText gesetzt. Es ist verfügbar als reST Quellcode auf [github](#), und online als HTML und PDF unter <https://www.qgis.org/en/docs/>. Übersetzte Versionen dieses Dokuments können auch über den Dokumentationsbereich des QGIS-Projekts durchsucht und heruntergeladen werden.

Weitere Informationen zum Mitwirken an diesem Dokument und der Übersetzung finden Sie unter <https://qgis.org/en/site/getinvolved/index.html>.

### **Verweise in diesem Dokument**

Dieses Dokument enthält interne und externe Links. Durch Klicken auf einen internen Link bewegt man sich innerhalb des Dokuments, während durch Klicken auf einen externen Link eine Internetadresse geöffnet wird.

### **Autoren und Redakteure der Dokumentation**

Die Liste der Personen, die zum Schreiben, Überprüfen und Übersetzen der folgenden Dokumentation beigetragen haben, finden Sie unter [Mitwirkende](#).

Copyright (c) 2004 - 2020 QGIS Development Team

**Internet:** <https://www.qgis.org>

### **Lizenz des Dokuments**

Es wird die Erlaubnis gewährt, dieses Dokument zu kopieren, zu verteilen und/oder zu modifizieren, unter den Bestimmungen der GNU Free Documentation License, Version 1.3 oder jeder späteren Version, veröffentlicht von der Free Software Foundation; ohne unveränderliche Abschnitte, ohne vordere Umschlagtexte und ohne hintere Umschlagtexte. Eine Kopie der Lizenz wird im Kapitel [Anhang B: GNU Free Documentation License](#) bereitgestellt.

## **1.1 Was ist neu in QGIS 3.34**

Diese Version von QGIS enthält Hunderte von Fehlerkorrekturen und viele neue Funktionen und Verbesserungen gegenüber **QGIS 3.28**. Eine Liste der neuen Funktionen finden Sie in den Änderungsprotokollen unter <https://qgis.org/de/site/forusers/visualchangelogs.html>.

Wir empfehlen die Verwendung dieser Version gegenüber früheren Versionen.



Willkommen in der wunderbaren Welt der Geographischen Informationssysteme (GIS)!

QGIS is an Open Source Geographic Information System. The project was born in May 2002 and was established as a project on SourceForge in June the same year. We have worked hard to make GIS software (which is traditionally expensive proprietary software) available to anyone with access to a personal computer.

QGIS currently runs on most Unix platforms, Windows, and macOS. QGIS is developed using the Qt toolkit (<https://www.qt.io>) and C++. This means that QGIS feels snappy and has a pleasing, easy-to-use graphical user interface (GUI). There are also independently created applications that allow you to take QGIS into the field. These applications can run on Android and iOS.

QGIS soll ein benutzerfreundliches GIS sein, das allgemeine Funktionen und Merkmale bietet. Das ursprüngliche Ziel des Projekts war die Bereitstellung eines GIS-Datenviewers. QGIS hat einen Punkt in seiner Entwicklung erreicht, an dem es für den täglichen Bedarf zur GIS-Datenbetrachtung, für die Datenerfassung, für fortgeschrittene GIS-Analysen und für Präsentationen in Form von anspruchsvollen Karten, Atlanten und Berichten verwendet wird. QGIS unterstützt eine Vielzahl von Raster- und Vektordatenformaten, wobei die Unterstützung neuer Formate durch die Plugin-Architektur leicht erweitert werden kann.

QGIS wird unter der GNU General Public License (GPL) veröffentlicht. Die Entwicklung von QGIS unter dieser Lizenz bedeutet, dass Sie den Quellcode einsehen und verändern können, und garantiert, dass Sie, unser glücklicher Anwender, immer Zugang zu einem GIS-Programm haben werden, das kostenlos ist und frei verändert werden kann. Sie sollten eine vollständige Kopie der Lizenz mit Ihrer Kopie von QGIS erhalten haben und Sie können sie auch im Anhang *Anhang A: GNU General Public License* finden.

---

### **Tipp: Aktuellste Dokumentation**

Die aktuellste Version dieses Dokuments können Sie immer im Dokumentationsbereich der QGIS Internetseite unter <https://www.qgis.org/en/docs/> finden.

---



---

## Gebrauch der Dokumentation

---

Dieser Abschnitt beschreibt die in diesem Handbuch benutzten einheitlichen Schreibstile.

### 3.1 GUI Schreibstile

Die GUI Schreibstile sollen das Erscheinungsbild der Grafischen Benutzeroberfläche nachahmen. Im Allgemeinen gibt eine Darstellung das normale Erscheinungsbild wieder, also ohne das der Mauszeiger darüber ist, so dass der Benutzer die GUI nach etwas, das wie die Darstellung im Handbuch aussieht, absuchen kann.

- Menü Optionen: *Layer ► Rasterlayer hinzufügen* oder *Einstellungen ► Werkzeugkasten ► Digitalisierung*

- Werkzeug:  Rasterlayer hinzufügen

- Schaltfläche: *Als Vorgabe speichern*

- Titel einer Dialogbox: *Layereigenschaften*

- Reiter: *Allgemein*

- Kontrollkästchen: ☒ *Darstellen*

- Radiobutton: ☐ *Postgis SRID* ☐ *EPSG ID*

- Wähle eine Zahl:

- Wähle ein Wort:

- Datei auswählen: ...

- Wähle eine Farbe:

- Schieberegler:

- Eingabetext:

Ein Schatten zeigt, dass dieses GUI Element mit der Maus anwählbar ist.

## 3.2 Text oder Tastatur Schreibstile



Dieses Handbuch enthält auch Darstellungen, die sich auf Texte, Tastaturbefehle und Codes beziehen, um verschiedene Einheiten, wie Klassen oder Methoden anzuzeigen. Diese Stile entsprechen aber nicht dem tatsächlichen Aussehen von Texten oder Codezeilen innerhalb von QGIS.

- Querverweise: <https://qgis.org>
- Tastenkombinationen: Drücken Sie `Strg+B`, was heisst dass Sie doe Strg-Taste drücken und halten sollen und dann die B-Taste drücken sollen.
- Name einer Datei: `lakes.shp`
- Name einer Klasse: **New Layer**
- Methode: `classFactory`
- Server: `myhost.de`
- User Text: `qgis --help`



Codezeilen werden durch eine Schriftart mit festgelegter Breite angezeigt:

```
PROJCS["NAD_1927_Albers",  
GEOGCS["GCS_North_American_1927",
```


## 3.3 Betriebssystemspezifische Anweisungen

GUI Sequenzen und kleine Textzeilen werden wie folgt formatiert: Klicken Sie   Datei **X** QGIS ► QGIS beenden. Dies beschreibt dass Sie unter Linux, Unix und Windows Betriebssystemen erst das Datei-Menü und dann ‚QGIS beenden‘ klicken sollten während Sie unter dem macOS-Betriebssystem erst das QGIS-Menü klicken und dann ‚Verlassen‘ klicken sollten.

Größere Texte können als Liste formatiert werden:

-  Mache dies
-  Mache das
- **X** Oder jenes

oder als Paragraph

 **X** Mache dies und dies und dies. Dann mache dies und dies und dies, und dies und dies und dies, und dies und dies und dies.

 Tun Sie das. Dann tun sie das und das und das, das und das und das, das und das und das, das und das.

Screenshots, die im Benutzerhandbuch erscheinen, wurden auf verschiedenen Plattformen erstellt.

QGIS bietet eine Fülle von GIS Funktionen, bereitgestellt durch Kernfunktionen und Erweiterungen. Die Suchleiste macht es einfach nach Funktionen, Datensätzen und mehr zu suchen.

Im Folgenden finden Sie eine kurze Zusammenfassung von sechs allgemeinen Kategorien von Funktionen und Plugins, gefolgt von ersten Einblicken in die integrierte Python-Konsole.

### 4.1 Daten visualisieren

Es ist möglich, Vektor- und Rasterdaten (in 2D und 3D) in unterschiedlichen Formaten und mit verschiedenen Projektionen zu visualisieren, ohne die Daten selbst in ein internes oder gemeinsames Format konvertieren zu müssen. Zu den unterstützten Datenformaten gehören:

- Tabellen und Views aus räumlichen Datenbanken wie PostGIS, SpatiaLite und MS SQL Spatial, Oracle Spatial, Vektorformate die durch die installierte OGR Bibliothek unterstützt werden, darunter GeoPackage, ESRI Shapedateien, MapInfo, SDTS, GML und viele mehr. Siehe Kapitel *Arbeiten mit Vektordaten*.
- Raster- und Bilddatenformate die von der installierten GDAL (Geospatial Data Abstraction Library) Bibliothek unterstützt werden wie z.B. GeoTIFF, ERDAS IMG, ArcInfo ASCII GRID, JPEG, PNG und viele mehr. Siehe Kapitel *Arbeiten mit Rasterdaten*.
- Mesh Daten (siehe *Working with Mesh Data*)
- Vektorkacheln (Vector Tiles; siehe *Arbeiten mit Vektorkacheln (Vector Tiles)*)
- GRASS Raster- und Vektordaten aus GRASS Datenbanken (Location/Mapset). Siehe Kapitel *GRASS GIS Integration*.
- Online-Geodaten, die als OGC-Webdienste bereitgestellt werden, einschließlich WMS, WMTS, WCS, WFS und WFS-T. Siehe Abschnitt *Arbeiten mit OGC / ISO-Protokollen*.

Die QGIS-Authentifizierungsinfrastruktur hilft Ihnen bei der Verwaltung von Benutzern/Passwörtern, Zertifikaten und Schlüsseln für Webdienste und andere Ressourcen.

- Tabellen (ODS, XLS(X), ...)

Temporäre Daten werden unterstützt.

## 4.2 Daten erkunden, abfragen und Karten layouten

Sie können Karten zusammenstellen und interaktiv räumliche Daten mit einer benutzerfreundlichen GUI erkunden. Die vielen in der GUI erhältlichen hilfreichen Werkzeuge beinhalten:

- QGIS Browser
- Spontanreprojektion
- 2D und 3D Kartendarstellungen
- DB Manager
- Drucklayout
- Berichte
- Kartenübersichtsfenster
- Räumliche Lesezeichen
- Beschriftungswerkzeuge
- Identifizieren/Selektieren von Objekten
- Editieren/Visualisieren/Suchen von Attributdaten
- Datendefinierte Objektbeschriftung
- Datendefinierte Vektor- und Rastersymbolisierungswerkzeuge
- Atlas Kartenzusammenstellung mit Gradnetz-Layern
- Nordpfeil, Maßstabsbalken und Urheberrechtsnachweis für Karten
- Unterstützung für das Speichern und Wiederherstellen von Projekten

## 4.3 Daten erstellen, editieren, verwalten und exportieren

Sie können Vektor- und Rasterlayer erstellen, bearbeiten und in zahlreiche Formate exportieren. QGIS bietet folgendes:

- Werkzeuge zur Vektordigitalisierung
- Erzeugen und Bearbeiten vieler verschiedener Dateiformate und GRASS Vektorlayer
- Georeferenzierungswerkzeug um Vektoren und Bilder zu georeferenzieren
- GPS-Werkzeuge zum Im- und Export von GPX-Daten, zur Konvertierung anderer GPS-Datenformate ins GPX-Format sowie das direkte Hoch-/Runterladen direkt auf/von einem GPS-Gerät (unter GNU/Linux wurde usb: zur Liste der GPS-Geräte hinzugefügt)
- Unterstützung für das Darstellen und Bearbeiten von OpenStreetMap-Daten
- Erzeugen von Tabellen in räumlichen Datenbanken aus Dateien mit der DB-Verwaltung
- Verbesserte Handhabung von räumlichen Datenbanktabellen
- Werkzeuge um Vektorattributtabelle zu verwalten
- Möglichkeit Screenshots als georeferenzierte Bilder zu speichern
- DXF - Export-Tool mit erweiterten Funktionen um Stile und Plugins zu exportieren und CAD -ähnliche Funktionen auszuführen

## 4.4 Daten analysieren

Sie können räumliche Datenanalysen mit räumlichen Datenbanken und anderen OGR-unterstützten Formaten durchführen. QGIS bietet derzeit Werkzeuge für Vektoranalyse, Rasteranalyse, Stichproben, Geoprocessing, Geometrie und Datenbankmanagement. Sie können auch die integrierten GRASS-Werkzeuge verwenden, die die gesamte GRASS-Funktionalität von mehr als 400 Modulen umfassen (siehe Abschnitt [GRASS GIS Integration](#)). Oder Sie können mit dem Processing-Plugin arbeiten, das ein leistungsfähiges Framework für die Geodatenanalyse bietet, um native Algorithmen aus QGIS und Algorithmen von Drittanbietern aufzurufen, wie z.B. GDAL, SAGA, GRASS, OTB, R und mehr (siehe Abschnitt [Einführung](#)). Alle Analysefunktionen werden im Hintergrund ausgeführt, so dass Sie Ihre Arbeit fortsetzen können, während die Verarbeitung läuft.

Mit dem grafischen Modellierer können Sie in einer intuitiven grafischen Umgebung Funktionen zu einem vollständigen Arbeitsablauf kombinieren/verketten.

## 4.5 Karten im Internet veröffentlichen

QGIS kann als WMS-, WMTS-, WMS-C-, WFS-, OAPIF- und WFS-T-Client verwendet werden (siehe Abschnitt [Arbeiten mit OGC / ISO-Protokollen](#)), und QGIS Server (siehe [QGIS-Server-manual](#)) ermöglicht es Ihnen, Ihre Daten über die WMS-, WCS-, WFS- und OAPIF-Protokolle mit Hilfe eines Webservers im Internet zu veröffentlichen.

## 4.6 Erweiterte QGIS Funktionalität durch Erweiterungen

QGIS kann an Ihre speziellen Bedürfnisse mit seiner erweiterbaren Plugin-Architektur und Bibliotheken, die zum Erstellen von Plugins benutzt werden können, angepasst werden. Sie können sogar neue Anwendungen mit C++ oder Python erstellen!

### 4.6.1 Kern-Plugins

Kern-Plugins sind:

1. DB Manager (austauschen, bearbeiten und darstellen von Layern und Tabellen in die und aus Datenbanken; Ausführen von SQL-Abfragen)
2. Geometrieprüfung (Geometrien auf Fehler prüfen)
3. GDAL-Georeferenzierung (Projektionsinformationen mit GDAL zu Rasterdatensätzen hinzufügen)
4. GPS Werkzeuge (GPS Daten laden und importieren)
5. GRASS (integriertes GRASS GIS)
6. MetaSearch Catalogue Client (mit Metadatenkatalogdiensten kommunizieren, die den Catalog Service for the Web (CSW) -Standard unterstützen)
7. Offline-Bearbeitung (Ermöglicht die Offline-Bearbeitung und Synchronisation mit Datenbanken)
8. Verarbeitung (der Verarbeitungswerkzeugkasten für räumliche Daten für QGIS)
9. Topologie-Prüfung (Topologiefehler in Vektorlayern finden)

## 4.6.2 Externe Python Plugins

QGIS bietet eine steigende Anzahl von externen Python Plugins, die von der Community bereitgestellt werden, an. Diese Plugins werden in dem offiziellen Plugins Repository vorgehalten und können auf einfache Art und Weise mit der Python Plugin Installation installiert werden. Siehe Kapitel *Der Erweiterungen Dialog*.

## 4.7 Python-Konsole

Um Skripte zu schreiben können Sie die integrierte Python-Konsole benutzen, die Sie mit dem Menü *Erweiterungen* ► *Python-Konsole* öffnen können. Die Konsole öffnet sich als nicht-modales Fenster. Für das Zusammenspiel mit der QGIS Umgebung gibt es die Variable `qgis.utils iface`, die eine Instanz von `QgisInterface` ist. Diese Schnittstelle ermöglicht den Zugang zum Kartenfenster, zu Menüs, Werkzeugleisten und anderen Teilen der QGIS-Anwendung. Sie können ein Skript schreiben und ins QGIS-Anwendungsfenster ziehen und es wird automatisch ausgeführt.

Weitere Informationen zum Arbeiten mit der Python-Konsole und das Programmieren von QGIS Erweiterungen und Anwendungen finden Sie unter PyQGIS-Developer-Cookbook.

## 4.8 Bekannte Probleme

### 4.8.1 Begrenzung der Anzahl von geöffneten Dateien

Wenn Sie ein großes QGIS-Projekt öffnen und Sie sicher sind dass alle Layer gültig sind aber einige Layer als schlecht markiert sind, stehen Sie wahrscheinlich vor diesem Problem. Linux (und auch andere Betriebssysteme) hat eine prozessbezogene Begrenzung von geöffneten Dateien. Ressourcengrenzen beziehen sich auf einen Prozess und werden vererbt. Der `ulimit` Befehl, den die Shell zur Verfügung stellt, verändert die Begrenzung nur für den aktuellen Shell-Prozess; die neue Begrenzung wird von allen untergeordneten Prozessen geerbt.

Sie können sich alle Informationen zu `ulimit` anzeigen lassen, indem Sie folgenden Befehl eingeben:

```
$ ulimit -aS
```

Sie können die aktuell zulässige Anzahl von geöffneten Dateien pro Prozess mit dem folgenden Befehl in der Konsole sehen:

```
$ ulimit -Sn
```

Um die Begrenzungen für eine **vorhandene Sitzung** zu ändern, können Sie so etwas verwenden:

```
$ ulimit -Sn #number_of_allowed_open_files
$ ulimit -Sn
$ qgis
```

Alternativ können Sie auch das neuere Utility `prlimit` verwenden. Weitere Informationen dazu: <https://manpages.ubuntu.com/manpages/latest/man1/prlimit.1.html>

#### Um es für immer zu beheben

Unter den meisten Linux-Systemen werden die Ressourcenbegrenzungen beim Login durch das `pam_limits` Modul gemäß den Einstellungen in `etc/security/limits.conf` oder `/etc/security/limits.d/*.conf` eingestellt. Sie müssten in der Lage sein die Dateien zu editieren wenn Sie Root-Rechte haben (auch über `sudo`), die Änderungen werden jedoch erst wirksam wenn Sie sich erneut anmelden.

Mehr Informationen:

<https://www.cyberciti.biz/faq/linux-increase-the-maximum-number-of-open-files/> <https://linuxaria.com/article/open-files-in-linux>



Dieses Kapitel gibt einen schnellen Überblick über die Installation von QGIS, den Download von QGIS-Beispieldaten und die Durchführung einer ersten einfachen Sitzung zur Visualisierung von Raster- und Vektordaten.

### 5.1 QGIS installieren

Das QGIS-Projekt bietet verschiedene Möglichkeiten, QGIS in Abhängigkeit von Ihrer Plattform zu installieren.

#### 5.1.1 Installation mit Binärdateien

Standardinstallationen sind für  MS Windows und  macOS verfügbar. Binärpakete (rpm und deb) oder Software-Repositorys werden für viele Varianten von GNU/Linux  bereitgestellt.

Weitere Informationen und Anweisungen für Ihr Betriebssystem finden Sie unter <https://download.qgis.org>.

#### 5.1.2 Kompilieren des Quellcodes

Wenn Sie QGIS mit Hilfe des Quellcodes erstellen wollen, lesen Sie bitte die Installationsanweisungen. Sie erhalten diese zusammen mit dem QGIS-Quellcode in einer Datei namens `INSTALL`. Sie können sie auch online unter [https://github.com/qgis/QGIS/blob/release-3\\_34/INSTALL.md](https://github.com/qgis/QGIS/blob/release-3_34/INSTALL.md) finden.

Wenn Sie eine bestimmte Version und nicht die in der Entwicklung befindliche Version erstellen möchten, sollten Sie `Master` durch den Release-Zweig (üblicherweise im Formular `release-X_Y`) im oben genannten Link ersetzen (Installationsanweisungen können abweichen).

### 5.1.3 Auf externen Medien installieren

Es ist möglich, QGIS (mit allen Plugins und Einstellungen) auf einem Flash-Laufwerk zu installieren. Dies wird durch die Definition einer Option `–profiles-path` erreicht, die den standardmäßigen Pfad *user profile* überschreibt und **QSettings** zwingt, auch dieses Verzeichnis zu verwenden. Siehe Abschnitt *System* für weitere Informationen.

### 5.1.4 Beispieldaten herunterladen

Dieses Benutzerhandbuch enthält Beispiele, die auf dem QGIS-Beispieldatensatz (auch Alaska-Datensatz genannt) basieren. Laden Sie die Beispieldaten von <https://github.com/qgis/QGIS-Sample-Data/archive/master.zip> herunter und entpacken Sie das Archiv an einem beliebigen Ort auf Ihrem System.




Der QGIS-Beispieldatensatz enthält Geodaten von Alaska und deckt sämtliche Übungen und Screenshots dieser Dokumentation ab, inklusive einer kleinen GRASS GIS Datenbank. Das Koordinatenbezugssystem ist Albers Equal Area mit der Maßeinheit 'feet' (EPSG-Code 2964).

```
PROJCS["Albers Equal Area",
GEOGCS["NAD27",
DATUM["North_American_Datum_1927",
SPHEROID["Clarke 1866",6378206.4,294.978698213898,
AUTHORITY["EPSG","7008"]],
TOWGS84[-3,142,183,0,0,0,0],
AUTHORITY["EPSG","6267"]],
PRIMEM["Greenwich",0,
AUTHORITY["EPSG","8901"]],
UNIT["degree",0.0174532925199433,
AUTHORITY["EPSG","9108"]],
AUTHORITY["EPSG","4267"]],
PROJECTION["Albers_Conic_Equal_Area"],
PARAMETER["standard_parallel_1",55],
PARAMETER["standard_parallel_2",65],
PARAMETER["latitude_of_center",50],
PARAMETER["longitude_of_center",-154],
PARAMETER["false_easting",0],
PARAMETER["false_northing",0],
UNIT["us_survey_feet",0.3048006096012192]]
```

Wenn Sie beabsichtigen, QGIS als grafisches Frontend für GRASS zu verwenden, finden Sie eine Auswahl von Beispiellorten (z.B. Spearfish oder South Dakota) auf der offiziellen GRASS GIS Website <https://grass.osgeo.org/download/data/>.


## 5.2 QGIS starten und beenden

QGIS kann wie jede andere Anwendung auf Ihrem Computer gestartet werden. Das bedeutet, dass Sie QGIS wie folgt starten können:

- über  das Anwendungsmenü,  das Startmenü oder  das Dock
- Doppelklicken Sie das QGIS Icon oder die Desktop Verknüpfung
- Doppelklicken Sie auf eine vorhandene QGIS-Projektdatei (mit der Erweiterung `.qgz` oder `.qgs`). Beachten Sie, dass dadurch auch das Projekt geöffnet wird.
- `qgis` in die Kommandozeile eintippen, vorausgesetzt, QGIS wurde ihrem PFAD hinzugefügt oder Sie befinden sich im Installationsordner

Um QGIS zu stoppen, benutzen Sie:

-   die Menüoptionen *Projekt ► QGIS beenden* oder benutzen Sie das Tastenkürzel `Strg+Q`



-  *QGIS* ► *QGIS beenden*, oder benutzen Sie das Tastenkürzel `Ctrl+Q`
- oder benutzen Sie das rote Kreuz in der rechten, oberen Ecke des Hauptfensters der Anwendung.

## 5.3 Beispielsitzung: Laden von Raster- und Vektorlayern

Nachdem Sie nun *QGIS installiert* und einen *Beispiel-Datensatz* zur Verfügung haben, werden wir eine erste Beispielsitzung demonstrieren. In diesem Beispiel werden wir einen Raster- und einen Vektorlayer laden. Wir verwenden:

- den `landcover` Rasterlayer (`qgis_sample_data/raster/landcover.img`)
- und den `lakes` Vektorlayer (`qgis_sample_data/gml/lakes.gml`)

Wobei `qgis_sample_data` der Pfad zum ausgepackten Datensatz ist.

1. Starten Sie QGIS wie in *QGIS starten und beenden* gezeigt
2. The data we will be working with are in Albers Equal Area, so let's set the project's CRS accordingly:
  1. Click the  *Select projection* button in the bottom right of QGIS interface. The project properties dialog opens with the *CRS* tab active.
  2. Type `2964` in the  *Filter* text area.
  3. Select the row with `NAD27 / Alaska Albers` CRS name.

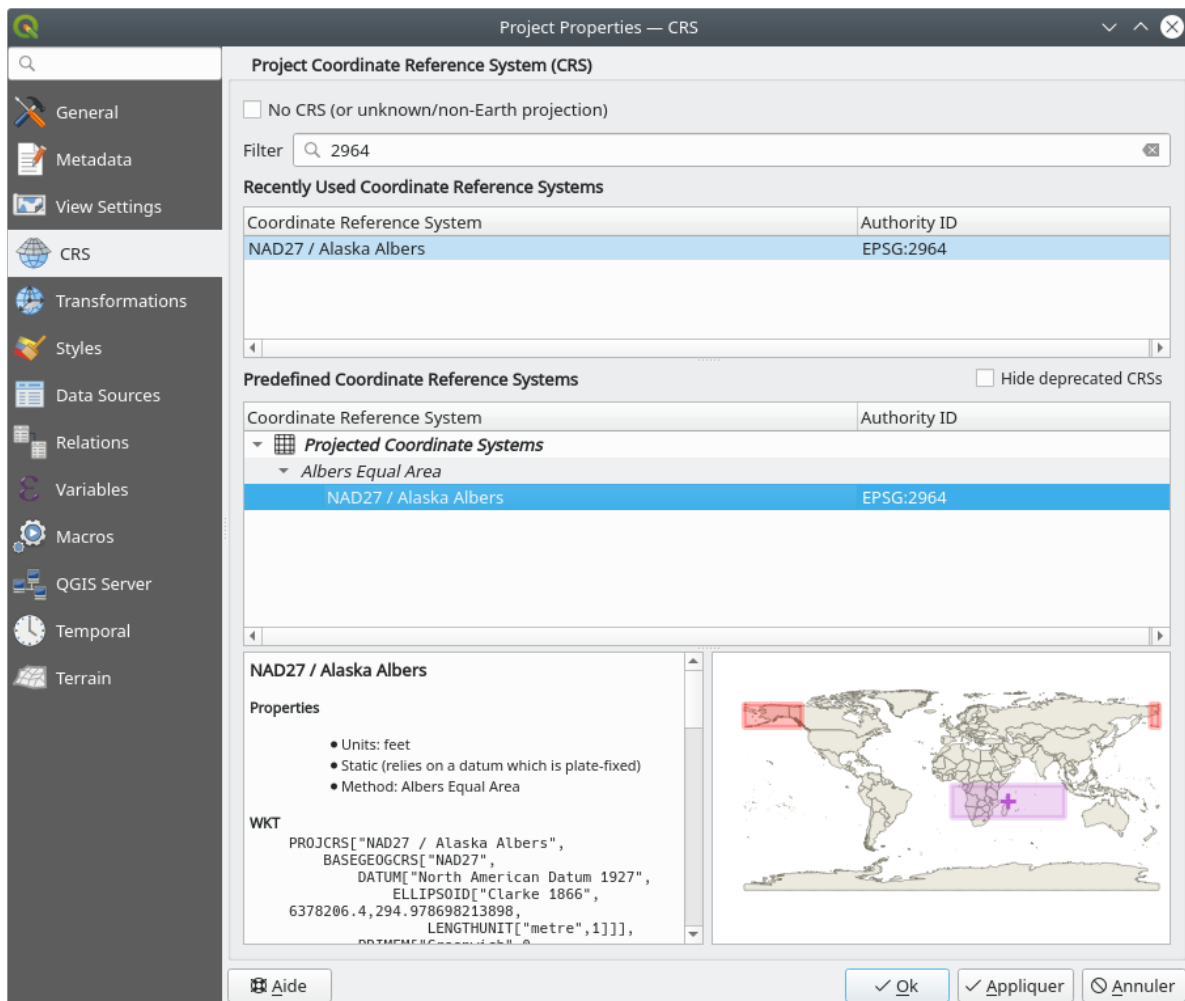




Abb. 5.1: Auswahl des Koordinatenbezugssystems für die Datei

4. Klicken Sie auf *OK*

**Bemerkung:** You can ignore/close for now the „ballpark transform“ message that could display.

3. Load the files in QGIS:

1. Klicken Sie auf die Schaltfläche  **Datenquellenverwaltung öffnen**. Die Datenquellenverwaltung sollte sich im Browsermodus öffnen.
2. Navigieren Sie zum Verzeichnis `qgis_sample_data/raster/`
3. Wählen Sie die ERDAS IMG-Datei  **landcover.png** mit einem Doppelklick aus. Der Layer wird im Hintergrund geladen, während das Fenster Data Source Manager geöffnet bleibt.

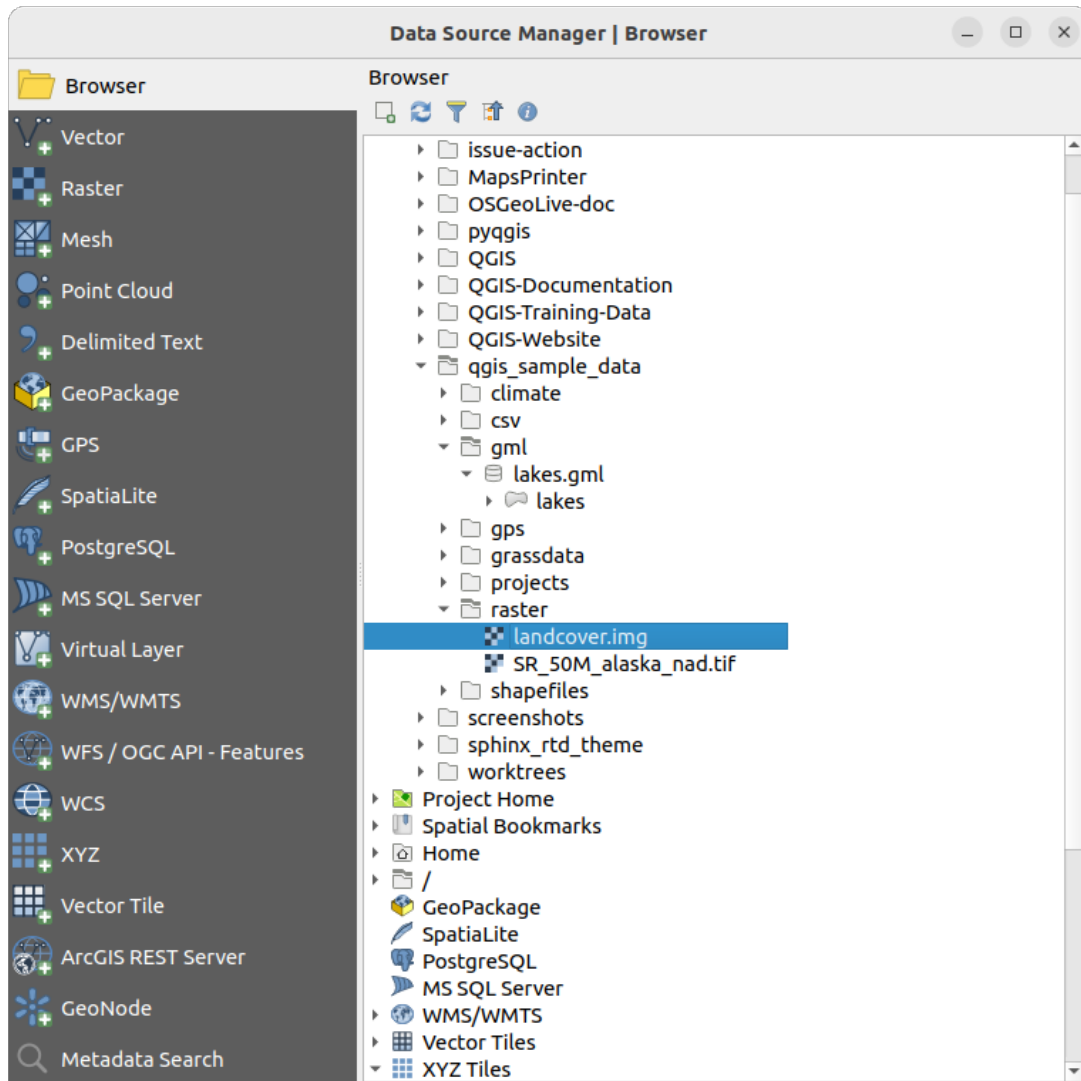


Abb. 5.2: Daten zu einem neuen Projekt in QGIS hinzufügen

4. To load the lakes data, browse to the folder `qgis_sample_data/gml/`, and drag and drop the `lakes.gml` file over QGIS main dialog. (Or just double-click as mentioned above.)
5. Das Dialogfeld *Elemente zum Hinzufügen auswählen* öffnet sich und durchsucht die Datei, da das `.gml`-Dateiformat mehr als einen Layer gleichzeitig enthalten kann.

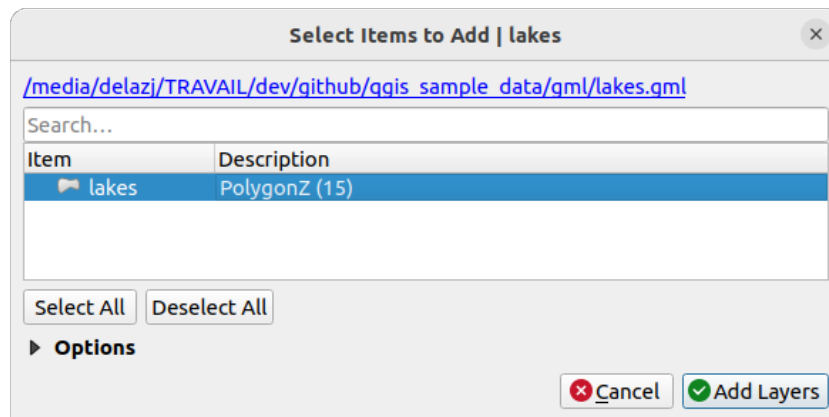





Abb. 5.3: Layer aus einer Datei auswählen

6. In unserem Fall gibt es nur einen Layer:  *lakes*. Wählen Sie diesen aus und klicken Sie auf *Layer hinzufügen*.



7. The layer is added to the *Layers* panel

4. Schließen Sie das Fenster „Datenquellenverwaltung“

In the *Layers* panel, you can notice that the *lakes* layer displays  Layer has no coordinate reference system set next to it. Let's adjust that.

1. Click the  icon. The *Coordinate Reference System Selector* dialog opens.
2. As done earlier, find and select the *NAD27 / Alaska Albers* CRS entry.
3. Klicken Sie auf *OK*

Jetzt haben Sie die beiden Layer in Ihrem Projekt, dargestellt mit zufälligen Farben. Los geht's, machen wir einige Anpassungen am Layer mit den Seen.

1. Wählen Sie das Werkzeug  *Hineinzoomen* in der Werkzeugleiste *Kartennavigation* aus
2. Zoomen Sie in ein Gebiet, in dem sich ein paar Seen befinden.
3. Doppelklicken Sie auf den Layer *lakes* in der Legende. Der Dialog *Layereigenschaften* öffnet sich
4. So ändern Sie die Farbe der Seen:
  1. Klicken Sie auf den Reiter  *Symbolisierung*
  2. Wählen Sie blau als Füllfarbe aus.

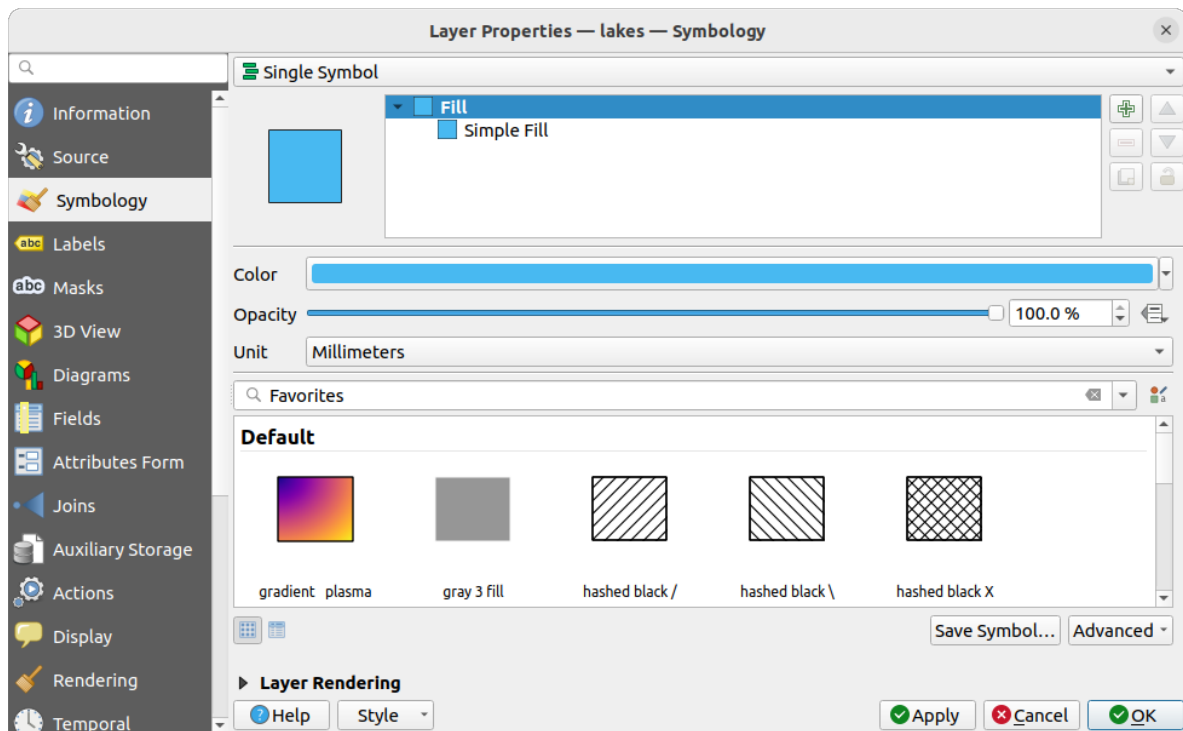


Abb. 5.4: Farbe für die Seen auswählen

3. Drücken Sie **OK**. Die Seen werden in der Kartenansicht jetzt in blau dargestellt.
5. So stellen Sie die Namen der Seen dar:
  1. Öffnen Sie den Dialog *Eigenschaften* des Layers `lakes`
  2. Klicken Sie auf den Reiter **abc** *Beschriftungen*
  3. Wählen Sie *Einzelne Beschriftungen* aus dem Aufklappmenü aus, um die Beschriftung anzuschalten.
  4. Wählen Sie aus der Liste *Beschriften mit* das Feld `NAMES`.

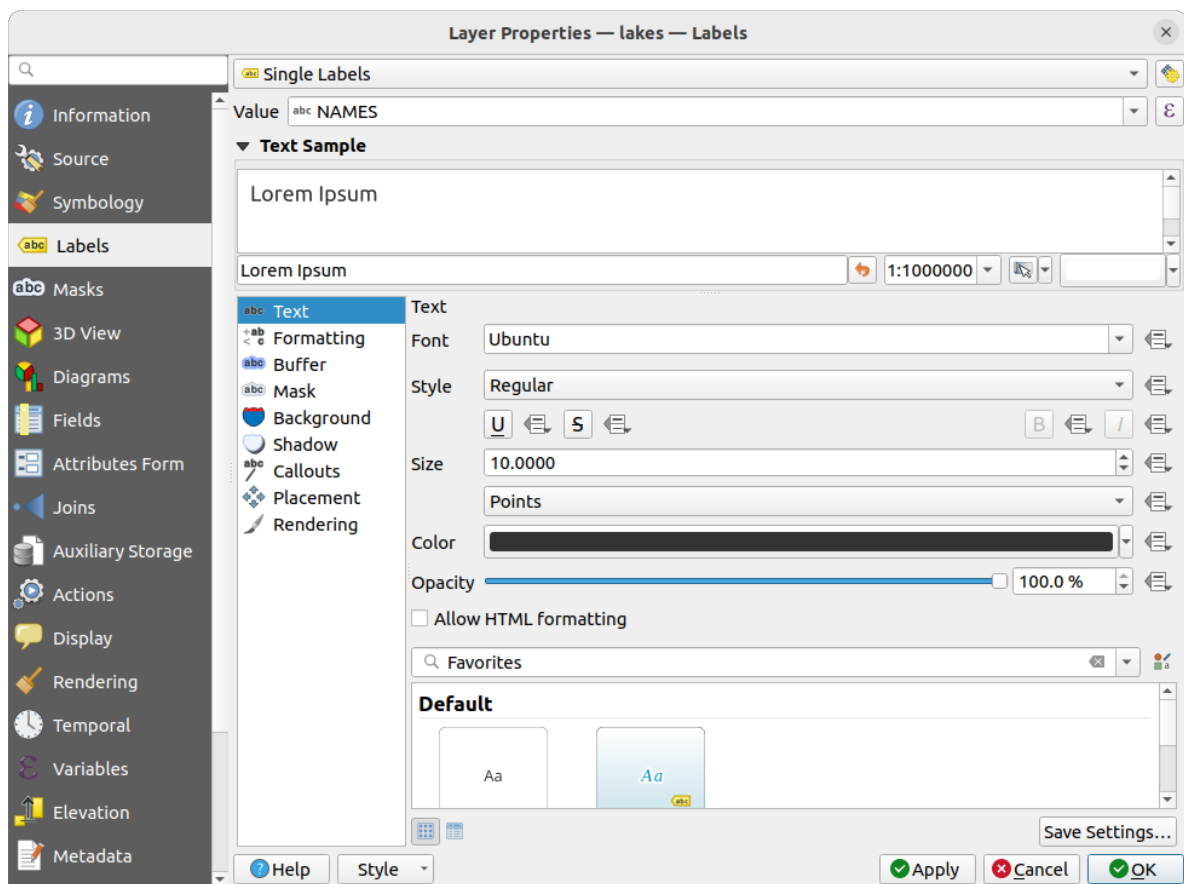


Abb. 5.5: Die Namen der Seen anzeigen

5. Drücken Sie auf *Anwenden*. Die Namen werden nun über den Grenzen dargestellt.
6. Sie können die Lesbarkeit der Beschriftungen verbessern, indem Sie einen weißen Puffer drumherum legen:
  1. Klicken Sie auf den Reiter *Puffer* in der Liste links
  2. Haken Sie ☒ *Textpuffer zeichnen* an
  3. Wählen Sie ``3`` als Größe aus
  4. Klicken Sie auf *Anwenden*
  5. Prüfen Sie nach, ob das Ergebnis gut aussieht und verändern Sie den Wert, falls nötig.
  6. Klicken Sie abschließend auf *OK*, um den *Layereigenschaften* Dialog zu schließen und die Änderungen zu übernehmen.



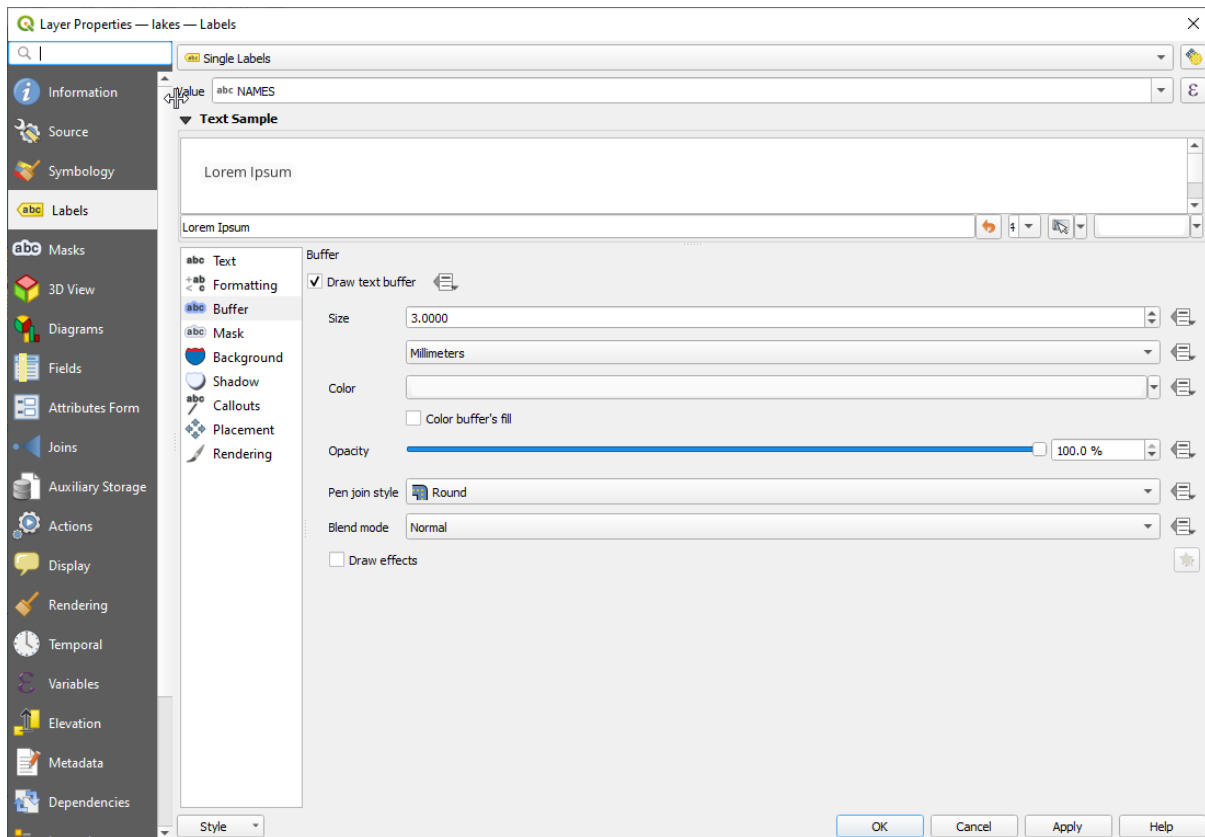


Abb. 5.6: Showing Buffers around Labels

Jetzt wollen wir einige Dekorationen hinzufügen, um die Karte in Form zu bringen und aus QGIS zu exportieren:

1. Wählen Sie im Menü *Ansicht ► Dekorationen ► Maßstab* aus
2. Im Dialog, der sich öffnet, haken Sie ☒ *Aktiviere Maßstab* an
3. Passen Sie die Einstellungen des Dialogs nach Ihren Wünschen an
4. Drücken Sie auf *Anwenden*
5. Auf die gleiche Art können Sie aus dem Menü *Dekorationen* weitere Objekte (Nordpfeil, Urheberrechtshinweis...) mit Ihren Einstellungen zur Kartenansicht hinzufügen
6. Klicken Sie auf den Menüpunkt *Projekt ► Import/Export ► Karte als Bild speichern...*
7. Drücken Sie auf *Speichern* im geöffneten Dialog
8. Wählen Sie einen Speicherpfad und ein Format aus, indem Sie nochmals auf *Speichern* drücken.
9. Drücken Sie auf *Projekt ► Speichern unter...*, um Ihre Änderungen als Projektdatei *.qgz* zu speichern.

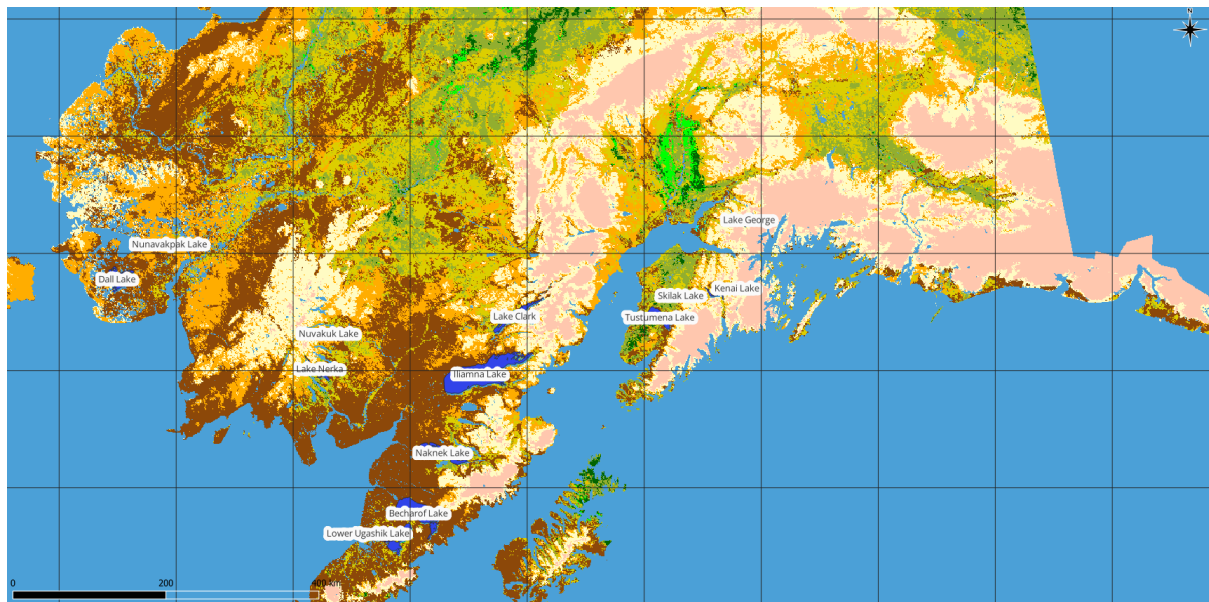


Abb. 5.7: Showing Exported Map with Decorations

Das war's. Sie sehen, wie einfach es ist, Raster- und Vektorlayer in QGIS darzustellen, ihre Eigenschaften zu ändern und Ihre Karte in einem Bildformat zu speichern, die Sie in anderer Software benutzen können. Lassen Sie uns nun weitergehen und lernen Sie mehr über die verfügbaren Funktionen und Einstellungen und wie Sie sie benutzen können.

---

**Bemerkung:** Lesen Sie das QGIS Übungshandbuch, um beim Lernen von QGIS mit schrittweisen Übungen weiterzumachen.


---


## Arbeiten mit Projektdateien

### 6.1 Vorstellung von QGIS-Projekten


Der aktuelle Zustand Ihrer QGIS-Sitzung heißt Projekt. QGIS hat stets nur ein Projekt in Arbeit. Eine Einstellung kann projektspezifisch oder eine anwendungsweite Voreinstellung für neue Projekte (siehe Abschnitt *Optionen*) sein.

QGIS kann den aktuellen Zustand in eine *QGIS-Projektdatei* speichern; benutzen Sie dafür das Menü *Projekt* ►  *Speichern* oder *Projekt* ►  *Speichern als...*

**Bemerkung:** Sobald das Projekt verändert wurde, erscheint das Zeichen \* in der Titelleiste und QGIS wird Sie standardmäßig fragen, ob Sie die Änderungen speichern wollen. Dieses Verhalten steuern Sie über die Einstellung  *Bei Bedarf nachfragen, ob geänderte Projekte und Datenquellen gespeichert werden sollen*, die Sie unter *Einstellungen* ► *Optionen* ► *Allgemein* finden.

Sie können bereits vorhandene Projekte über das Bedienfeld Browser öffnen oder über die Menüpunkte *Projekt* ►  *Öffnen...*, *Projekt* ► *Neu aus Vorlage* oder *Projekt* ► *Zuletzt verwendet* ►.

Beim Programmstart wird Ihnen eine Liste mit *Projektvorlagen* und *zuletzt geöffneten Projekten* angezeigt, die Screenshots, Namen und Dateipfade (für bis zu zehn Projekte) enthält. Die Liste *Kürzlich verwendet* ist praktisch, um an Ihre zuletzt verwendeten Projekte zu kommen. Doppelklicken Sie auf einen der Einträge, um ein Projekt oder eine Projektvorlage zu öffnen. Klicken Sie mit rechts auf einen Eintrag, um ihn *der Liste hinzu[zu]fügen*, das *Verzeichnis [zu] öffnen...* oder ihn *von der Liste [zu] entfernen*. Sie können auch einen neuen Layer hinzufügen und damit automatisch ein neues Projekt erzeugen. Die Liste verschwindet dann und die Kartenansicht wird sichtbar.

Wenn Sie Ihre Sitzung löschen und neu beginnen möchten, gehen Sie zu *Projekt* ►  *Neu*. Dabei werden Sie aufgefordert, das bestehende Projekt zu speichern, wenn seit dem Öffnen oder letzten Speichern Änderungen vorgenommen wurden.

Sobald Sie ein neues Projekt erzeugen, zeigt die Titelleiste bis zum Speichern den Titel *Unbenanntes Projekt* an.

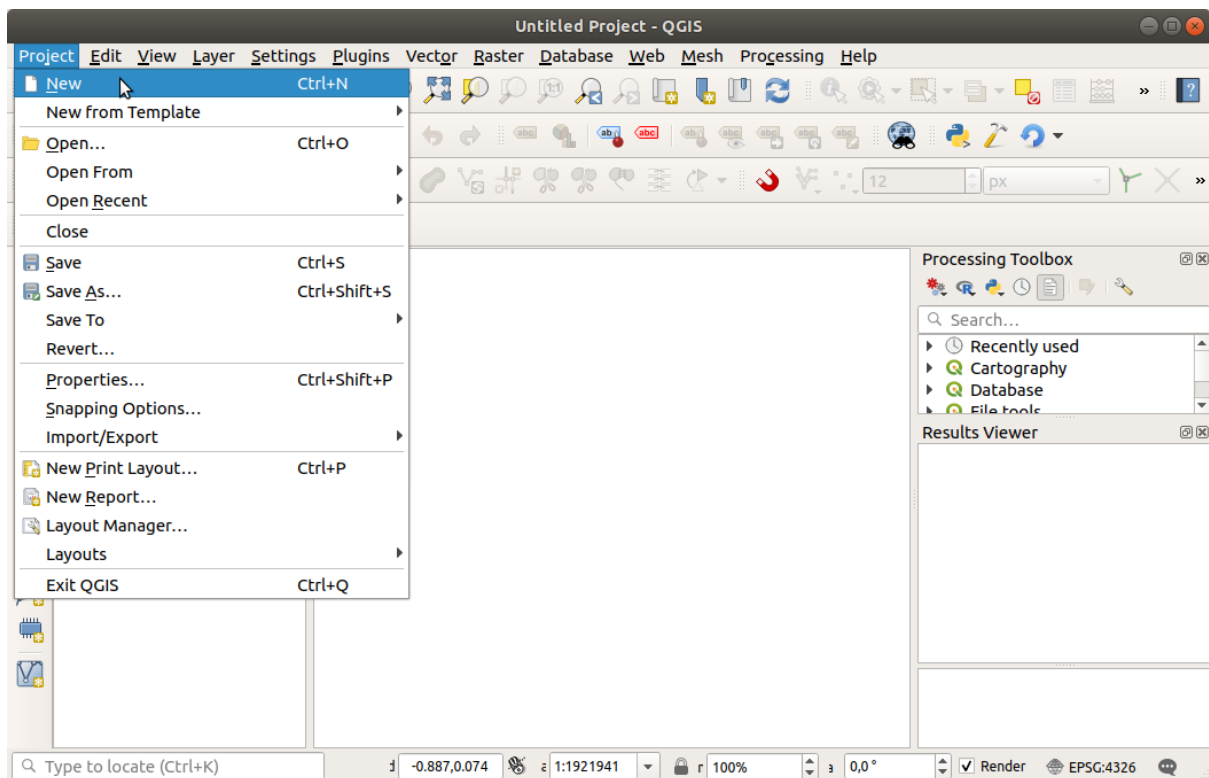


Abb. 6.1: Start eines neuen Projekts in QGIS


In einer Projektdatei sind folgende Informationen gespeichert:

- Hinzugefügte Layer
- Welche Layer können abgefragt werden
- Layereigenschaften, inklusive Symbologie und Styles
- Bemerkungen zu Layern
- 2D und 3D Kartenansichten
- Projektion für jede Kartenansicht
- Der zuletzt betrachtete Ausschnitt für jede Kartenansicht
- Drucklayouts
- Drucklayout Elemente mit Einstellungen
- Drucklayouts Atlas Einstellungen
- Digitalisierungseinstellungen
- Tabellen Beziehungen
- Projekt Makros
- Projekt Vorgabestile
- Plugin-Einstellungen
- QGIS-Servereinstellungen aus der Registerkarte QGIS-Server in den Projekteigenschaften.
- Im DB-Manager gespeicherte Abfragen

Die Projektdatei wird im XML-Format gespeichert (siehe [QGS/QGZ - Das QGIS-Projekt-Dateiformat](#)). Damit ist es möglich, die Datei außerhalb von QGIS zu bearbeiten, wobei Sie aber wissen sollten, was Sie tun. Das Dateiformat der

Projektdatei wurde bereits mehrere Male erneuert. Deshalb kann es sein, dass Projektdateien älterer QGIS-Versionen nicht mehr richtig funktionieren.

---

**Bemerkung:** Standardmäßig warnt Sie QGIS, wenn es Versionskonflikte gibt. Dieses Verhalten steuern Sie im Reiter *Allgemein* im Menü *Einstellungen ► Optionen* ( *Warnung ausgeben, wenn QGIS-Projekt einer früheren Version geöffnet wird*).

---

Immer wenn Sie eine `.qgs`-Projektdatei speichern, wird eine Sicherungsdatei im selben Verzeichnis angelegt, die die Dateierweiterung `.qgs~` hat.

The extension for QGIS projects is `.qgs` but when saving from QGIS, the default is to save using a compressed format with the `.qgz` extension. The `.qgs` file is embedded in the `.qgz` file (a zip archive), together with its associated SQLite database (`.qgd`) for *auxiliary data*. You can get to these files by unzipping the `.qgz` file.

---

**Bemerkung:** Der Mechanismus *Auxiliary Storage Properties* macht ein gepacktes Projekt besonders nützlich, weil es die Hilfsdaten einbettet.

---

Projects can also be saved/loaded to/from a PostgreSQL, GeoPackage or Oracle database using the following Project menu items:


- *Projekt ► Öffnen aus*
- *Projekt ► Speichern als*

Both menu items have a sub-menu with a list of extra project storage implementations (PostgreSQL, GeoPackage and Oracle). Clicking the action will open a dialog to pick a GeoPackage connection and project, a PostgreSQL connection, schema and project or Oracle connection, owner and project.


Projects stored in GeoPackage, PostgreSQL or Oracle can also be loaded through the QGIS browser panel, either by double-clicking them or by dragging them to the map canvas.

## 6.2 Behandlung von zerstörten Dateipfaden

Wenn Sie ein Projekt öffnen, kann QGIS daran scheitern einzelne Datenquellen zu erreichen, weil ein Dienst oder eine Datenbank nicht erreichbar ist oder weil eine Datei umbenannt oder verschoben wurde. QGIS öffnet dann den Dialog *Nicht verfügbare Layer behandeln*, der die nicht gefundenen Layer auflistet. Sie können:

- auf das Feld *Datenquelle* doppelklicken, den Pfad jedes Layers anpassen und auf *Änderungen anwenden* klicken;
- eine Zeile auswählen, auf *Durchsuchen* klicken, den korrekten Ort auswählen und danach auf *Änderungen anwenden* klicken;
- auf *Auto-Finden* klicken, die Verzeichnisse durchsuchen und versuchen alle oder die ausgewählten zerstörten Dateipfade automatisch zu reparieren. Achtung: das Durchsuchen kann eine Weile dauern. Klicken Sie danach auf *Änderungen anwenden*.
- Die Nachricht ignorieren und Ihr Projekt mit dem oder den zerstörten Pfaden öffnen, indem Sie auf *Nicht verfügbare Layer behalten* klicken. Der Layer wird dann im Bedienfeld *Layer* angezeigt, aber er enthält keine Daten, bis Sie den Pfad durch einen Klick auf  *Nicht verfügbarer Layer!* neben dem jeweiligen Layer im Bedienfeld *Layer* oder auf *Datenquelle reparieren...* im Kontextmenü des Layers reparieren.




Sobald Sie mit dem Werkzeug *Datenquelle reparieren...* erstmals einen Dateipfad repariert haben, sucht QGIS durch alle anderen zerstörten Pfade und versucht selbständig alle anderen zu reparieren, die den selben zerstörten Dateipfad haben.

-  *Nicht verfügbare Layer entfernen* entfernt alle nicht verfügbaren Layer aus dem Projekt.

Wenn Sie QGIS von der Kommandozeile aus mit der Option `--skipbadlayers` starten, können Sie den Dialog *Nicht verfügbare Layer behandeln* beim Start überspringen.

### 6.3 Ausgaben erzeugen

Es gibt mehrere Möglichkeiten, um Ausgaben aus Ihrer QGIS-Sitzung zu erzeugen. Wir haben das Speichern als Projektdatei bereits in *Vorstellung von QGIS-Projekten* besprochen. Andere Möglichkeiten, Ausgabedateien zu erzeugen, sind:

- Creating images: *Project ► Import/Export ►  Export Map to Image...* outputs the map canvas rendering to an image format (PNG, JPG, TIFF...) at custom scale, resolution and size. Including georeference information in the exported image is possible, simply enable  *Append georeference information (embedded or via world file)*. See *Exportieren der Kartenansicht* for more details.
- PDF-Dateien erzeugen: *Projekt ► Import/Export ► Karte als PDF exportieren...* gibt die Kartenansicht als PDF im gewünschten Maßstab, in der gewünschten Auflösung und mit einigen erweiterten Einstellungen (Vereinfachung, Georeferenzierung, ...) aus. Für eine genauere Beschreibung siehe *Exportieren der Kartenansicht*.
- DXF-Dateien erzeugen: *Project ► Import/Export ► Projekt als DXF speichern...* startet einen Dialog, in dem Sie den ‚Symbolisierungsmodus‘, den ‚Symbolisierungsmaßstab‘ und die Vektorlayer, die Sie nach DXF exportieren möchten, festlegen. Mithilfe des ‚Symbolisierungsmodus‘ können Sie Symbole des QGIS-Symbolsatzes mit hoher Ähnlichkeit exportieren (siehe Abschnitt *Creating new DXF files*).
- Karten herstellen: *Projekt ►  Neues Drucklayout...* öffnet einen Dialog, in dem Sie die aktuelle Kartenansicht layouten und drucken können (siehe Abschnitt *Kartenlayout*).

Die grafische Benutzeroberfläche (GUI) von QGIS ist in der unteren Abbildung dargestellt (die Zahlen 1 bis 5 in den gelben Kreisen kennzeichnen wichtige Elemente der Oberfläche und werden nachfolgend erläutert).

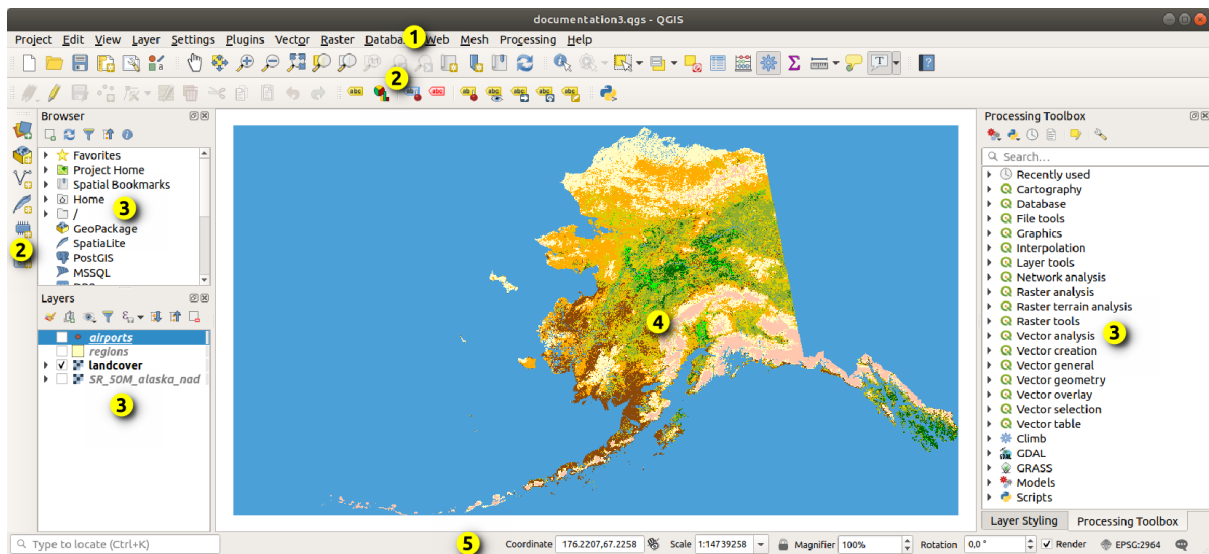


Abb. 7.1: QGIS GUI mit Alaskabeispieldatensatz

**Bemerkung:** Das Aussehen einzelner Bereiche (Titelleiste, etc.) kann in Abhängigkeit vom Betriebssystem und dem Fenstermanager abweichen.

Das QGIS-Hauptfenster (Abb. 7.1) setzt sich aus fünf Bestandteilen bzw. Arten von Bestandteilen zusammen:

1. *Menüleiste*
2. *Werkzeugleiste*
3. *Bedienfelder*
4. *Kartenansicht*

### 5. *Statusleiste*

Weiter unten in diesem Kapitel finden Sie nähere Erklärungen zu den einzelnen Punkten.

## 7.1 Menüleiste

Die Menüleiste ermöglicht den Zugriff auf QGIS-Funktionen über übliche, hierarchisch gegliederte Menüs. Die Menüs, ihre Optionen, die zugehörigen Symbole und Tastaturkürzel werden im Folgenden beschrieben. Die Tastaturkürzel können neu konfiguriert werden (*Einstellungen ► Tastaturkürzel*).

Die meisten Menü-Optionen haben ein entsprechendes Werkzeug und umgekehrt. Die Menüs sind jedoch nicht genau wie die Symbolleisten organisiert. Die Positionen der Menü-Optionen in den Symbolleisten sind in der Tabelle unten angegeben. Plugins können den Menüs neue Optionen hinzufügen. Für weitere Informationen über Werkzeuge und Werkzeugleisten siehe *Werkzeugkästen*.

---

**Bemerkung:** QGIS ist eine plattformübergreifende Anwendung. Die Werkzeuge sind im Allgemeinen auf allen Plattformen verfügbar, können aber je nach Betriebssystem in unterschiedlichen Menüs untergebracht sein. Die nachstehenden Listen zeigen die gängigsten Positionen, einschließlich bekannter Variationen.

---












### 7.1.1 Projekt

Das Menü *Projekt* bietet verschiedene Optionen für das Arbeiten mit *Projektdateien* an:

- Erstellen Sie mit *Neu* eine Projektdatei von Grund auf oder verwenden Sie eine andere Projektdatei als Vorlage (siehe *Projektdateien* für die Konfiguration von Vorlagen)
- *Open...* a project from a file, a GeoPackage, PostgreSQL or Oracle database
- *Schließen* eines Projekts oder Zurücksetzen auf den zuletzt gespeicherten Zustand
- *Save* a project in .qgs or .qgz file format, either as a file or within a GeoPackage, PostgreSQL or Oracle database
- Exportieren Sie das Kartenfenster in verschiedenen Formaten oder verwenden Sie ein *Drucklayout* für eine komplexere Ausgabe
- Legen Sie die Projekteigenschaften und die Fangoptionen für die Geometriebearbeitung fest.




Tab. 7.1: Die Einträge im Projekt-Menü

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
 <i>Neu</i>	Ctrl+N	Projekt	Vorstellung von QGIS-Projekten
<i>Neu aus Vorlage ►</i>			Vorstellung von QGIS-Projekten
 <i>Öffnen...</i>	Strg+O	Projekt	Vorstellung von QGIS-Projekten
<i>Öffnen aus ►</i>			Vorstellung von QGIS-Projekten
► <i>GeoPackage...</i>			Vorstellung von QGIS-Projekten
► <i>PostgreSQL...</i>			Vorstellung von QGIS-Projekten
► <i>Oracle...</i>			Vorstellung von QGIS-Projekten
<i>Zuletzt verwendetes ►</i>	Alt+J+R		Vorstellung von QGIS-Projekten
<i>Schließen</i>			Vorstellung von QGIS-Projekten
 <i>Speichern</i>	Strg+S	Projekt	Vorstellung von QGIS-Projekten
 <i>Speichern als...</i>	Strg+Shift+S	Projekt	Vorstellung von QGIS-Projekten
<i>Speichern als ►</i>			Vorstellung von QGIS-Projekten
► <i>Vorlagen...</i>			Vorstellung von QGIS-Projekten
► <i>GeoPackage...</i>			Vorstellung von QGIS-Projekten
► <i>PostgreSQL...</i>			Vorstellung von QGIS-Projekten
<i>Zurücknehmen...</i>			
 <i>Eigenschaften...</i>	Strg+Shift+P		Projekteigenschaften
<i>Fangoptionen...</i>			Optionen für das Einrasten und Digitalisieren
<i>Import/Export ►</i>			
►  <i>Karte als Bild speichern...</i>			Exportieren der Kartenansicht
►  <i>Karte als PDF exportieren...</i>			Exportieren der Kartenansicht
► <i>Projekt als DXF speichern...</i>			Creating new DXF files
► <i>Layer aus DWG/DXF importieren...</i>			Importing a DXF or DWG file
 <i>Neues Drucklayout...</i>	Strg+P	Projekt	Kartenlayout
 <i>Neuer Bericht...</i>			Creating a Report
 <i>Layout-Verwaltung...</i>		Projekt	Kartenlayout
<i>Layouts ►</i>			Kartenlayout
<i>Modelle ►</i>			The model designer
 <i>QGIS beenden</i>	Strg+Q		

Unter **X** macOS entspricht der Befehl :guilabel: *QGIS beenden* dem Befehl *QGIS ► QGIS beenden* (Cmd+Q).

## 7.1.2 Bearbeiten






















The *Edit* menu provides most of the native tools needed to edit layer attributes or geometry. To enable *Edit* menu options you need to switch to editing mode by clicking on  *Toggle editing* (see [Editierfunktionen](#) for details).

Tab. 7.2: Die Einträge im Bearbeiten-Menü

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
 <i>Rückgängig</i>	Strg+Z	Digitalisierung	<i>Rückgängig und Wiederholen</i>

Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
 Wiederholen	Strg+Shift+Z	Digitalisierung	Rückgängig und Wiederholen
 Ausgewählte Objekte ausschneiden	Strg+X	Digitalisierung	Objekte ausschneiden, kopieren und einfügen
 Objekte kopieren	Strg+C	Digitalisierung	Objekte ausschneiden, kopieren und einfügen
 Objekte einfügen	Strg+V	Digitalisierung	Objekte ausschneiden, kopieren und einfügen
Objekte einfügen als ►			Mit Attributtabelle arbeiten
► Neuer Vectorlayer...			Mit Attributtabelle arbeiten
► Temporärlayer...	Strg+Alt+V		Mit Attributtabelle arbeiten
 Ausgewähltes löschen		Digitalisierung	Ausgewählte Objekte löschen
Auswahl ►			Selecting features
►  Objekte über Rechteck und Einzelklick wählen		Auswahl	Selecting features
►  Objekte über Polygon wählen		Auswahl	Selecting features
►  Objekte freihändig oder über Einzelklick wählen		Auswahl	Selecting features
►  Objekte über Radius oder Einzelklick wählen		Auswahl	Selecting features
►  Objekte nach Wert wählen...	F3	Auswahl	Selecting features
►  Objekte nach Ausdruck wählen...	Strg+F3	Auswahl	Selecting features
►  Objektauswahl aller Layer aufheben	Strg+Alt+A	Auswahl	Selecting features
►  Objektauswahl des aktuell aktiven Layers aufheben	Strg+Umschalt	Auswahl	Selecting features
► Objekte wiederwählen			Selecting features
►  :guilabel: Alle Objekte wählen	Strg+A	Auswahl	Selecting features
►  Objektauswahl umkehren		Auswahl	Selecting features
 Datensatz hinzufügen	Strg+.	Digitalisierung	
 Punktojekt hinzufügen	Strg+.	Digitalisierung	Objekte digitalisieren
 Linienobjekt hinzufügen	Strg+.	Digitalisierung	Objekte digitalisieren
 Polygonobjekt hinzufügen	Strg+.	Digitalisierung	Objekte digitalisieren
 Add Circular String		Shape Digitizing	Circular string by radius
 Add Circular String by Radius		Shape Digitizing	Circular string by radius
Add Circle ►		Shape Digitizing	Kreise zeichnen






















Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
►  Add Circle from 2 Points		Shape Digitizing	Kreise zeichnen
►  Add Circle from 3 Points		Shape Digitizing	Kreise zeichnen
►  Add Circle from 3 Tangents		Shape Digitizing	Kreise zeichnen
►  Add Circle from 2 Tangents and a Point		Shape Digitizing	Kreise zeichnen
►  Add Circle by a Center Point and Another Point		Shape Digitizing	Kreise zeichnen
Add Rectangle ►		Shape Digitizing	Rechtecke zeichnen
►  Add Rectangle from Extent		Shape Digitizing	Rechtecke zeichnen
►  Add Rectangle from Center and a Point		Shape Digitizing	Rechtecke zeichnen
►  Add Rectangle from 3 Points (Distance from 2nd and 3rd point)		Shape Digitizing	Rechtecke zeichnen
►  Add Rectangle from 3 Points (Distance from projected point on segment p1 and p2)		Shape Digitizing	Rechtecke zeichnen
Add Regular Polygon ►		Shape Digitizing	Regelmäßige Polygone zeichnen
►  Add Regular Polygon from Center and a Point		Shape Digitizing	Regelmäßige Polygone zeichnen
►  Add Regular Polygon from Center and a Corner		Shape Digitizing	Regelmäßige Polygone zeichnen
►  Add Regular Polygon from 2 Points		Shape Digitizing	Regelmäßige Polygone zeichnen
Add Ellipse ►		Shape Digitizing	Ellipsen zeichnen
►  Add Ellipse from Center and 2 Points		Shape Digitizing	Ellipsen zeichnen
►  Add Ellipse from Center and a Point		Shape Digitizing	Ellipsen zeichnen
►  Add Ellipse from Extent		Shape Digitizing	Ellipsen zeichnen
►  Add Ellipse from Foci		Shape Digitizing	Ellipsen zeichnen
Beschriftung hinzufügen ►			Beschriftungstools
►  Beschriftungstext		Beschriftung	Beschriftungstools
►  Beschriftungsformular		Beschriftung	Beschriftungstools
►  HTML-Beschriftung		Beschriftung	Beschriftungstools
►  SVG -Beschriftung		Beschriftung	Beschriftungstools
Attribute bearbeiten ►			



Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
►  Attribute der gewählten Objekte ändern		Digitalisierung	Editiere Attributwerte
►  Attribute gewählter Objekte vereinen		Erweiterte Digitalisierung	Attribute gewählter Objekte vereinen
Geometrie bearbeiten ►			
►  Objekt(e) verschieben		Erweiterte Digitalisierung	Objekt(e) verschieben
►  Objekt(e) kopieren und verschieben		Erweiterte Digitalisierung	Objekt(e) verschieben
►  Objekt(e) drehen		Erweiterte Digitalisierung	Objekt(e) drehen
►  Objekt(e) skalieren		Erweiterte Digitalisierung	Objekt skalieren
►  Objekt vereinfachen		Erweiterte Digitalisierung	Objekt vereinfachen
►  Ring hinzufügen		Erweiterte Digitalisierung	Ring hinzufügen
►  Teil hinzufügen		Erweiterte Digitalisierung	Teil hinzufügen
►  Ring füllen		Erweiterte Digitalisierung	Ring füllen
►  Ring löschen		Erweiterte Digitalisierung	Ring löschen
►  Teil löschen		Erweiterte Digitalisierung	Teil löschen
►  Objekt überarbeiten		Erweiterte Digitalisierung	Objekte überarbeiten
►  Linie versetzen		Erweiterte Digitalisierung	Linie versetzen
►  Objekt teilen		Erweiterte Digitalisierung	Objekte trennen
►  Teile zerlegen		Erweiterte Digitalisierung	Teile zerlegen
►  Gewählte Objekte verschmelzen		Erweiterte Digitalisierung	Gewählte Objekte verschmelzen
►  Knotenwerkzeug (alle Layer)		Digitalisierung	Das Knotenwerkzeug
►  Knotenwerkzeug (aktiver Layer)		Digitalisierung	Das Knotenwerkzeug
►  Linie umkehren		Erweiterte Digitalisierung	Linie umkehren
►  Objekt abschneiden/ausdehnen		Erweiterte Digitalisierung	Objekt abschneiden/ausdehnen







Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
 <i>Punktsymbole drehen</i>		<i>Erweiterte Digitalisierung</i>	<i>Punktsymbole drehen</i>
 <i>Punktsymbolversatz</i>		<i>Erweiterte Digitalisierung</i>	<i>Punktsymbolversatz</i>

Werkzeuge, die vom ausgewählten Geometrietyp des Layers abhängen, d. h. Punkt, Linie oder Polygon, werden entsprechend aktiviert:

Tab. 7.3: Die verschiedenen „Objekt verschieben“ Werkzeuge

Menüeintrag	Punkt	Linie	Polygon
<i>Objekt(e) verschieben</i>			
<i>Objekt(e) kopieren und verschieben</i>			

### 7.1.3 Ansicht

Die Karte wird in Kartenansichten dargestellt. Mit den Werkzeugen aus dem *Ansicht* Menü können Sie mit diesen Ansichten interagieren. Sie können zum Beispiel:

- Erstellen eines zusätzlichen 2D- oder 3D-Kartenfensters neben der Haupt-Ansicht
- Auf einen beliebigen Ort *Zoomen* oder *Verschieben*
- Query the attributes or geometry of the displayed features
- Optimieren der Kartenansicht durch Kommentare, Dekorationen oder Voransichtsmodi
- Aufrufen aller Bedienfelder und Symbolleisten

Über das Menü können Sie auch die QGIS-Oberfläche selbst umgestalten, indem Sie Aktionen ausführen wie:
























- *Vollbildmodus umschalten*: erweitert das QGIS-Fenster auf den gesamten Bildschirm und blendet die Titelleiste aus
- *Toggle Panel Visibility*: shows or hides enabled *panels* - useful when digitizing features (for maximum canvas visibility) as well as for (projected/recorded) presentations using QGIS main canvas
- *Nur Karte umschalten*: blendet alle Bedienfelder, Symbolleisten, Menüs und die Statusleiste aus und zeigt nur die Kartenansicht an. In Kombination mit der Option „Vollbild“ wird nur die Karte auf dem Bildschirm angezeigt

Tab. 7.4: Die Einträge im Ansicht-Menü

Menüeintrag	Tastenkürzel	Werkzeuge	Referenz
 <i>Neue Kartenansicht</i>	Strg+M	<i>Kartennavigi</i>	<i>2D Map View</i>
<i>3D-Kartenansichten ►</i>			<i>3D-Ansicht</i>
 <i>Neue 3D-Kartenansicht</i>	Strg+Alt+M	<i>Kartennavigi</i>	<i>3D-Ansicht</i>
 <i>3D-Kartenansichten verwalten</i>			<i>3D-Ansicht</i>
 <i>Karte verschieben</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Karte zur Auswahl verschieben</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>











Fortsetzung auf der nächsten Seite


Tab. 7.4 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeugk	Referenz
 <i>Hineinzoomen</i>	Strg+Alt++	<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Herauszoomen</i>	Strg+Alt+-	<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Objekte abfragen</i>	Strg+Shift+I	<i>Attribute</i>	<i>Identifying Features</i>
<i>Messen ►</i>		<i>Attribute</i>	<i>Messen</i>
►  <i>Linie messen</i>	Strg+Umschalt+M	<i>Attribute</i>	<i>Messen</i>
►  <i>Fläche messen</i>	Strg+Umschalt+J	<i>Attribute</i>	<i>Messen</i>
►  <i>Measure Bearing</i>		<i>Attribute</i>	<i>Messen</i>
►  <i>Winkel messen</i>		<i>Attribute</i>	<i>Messen</i>
 <i>Statistische Zusammenfassung</i>		<i>Attribute</i>	<i>Statistical Summary Panel</i>
 <i>Elevation Profile</i>			<i>Elevation Profile View</i>
 <i>Volle Ausdehnung</i>	Strg+Shift+F	<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Zur Auswahl zoomen</i>	Strg+J	<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Auf Layer zoomen</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Auf eigene Auflösung zoomen (100%)</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Zoom zurück</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
 <i>Zoom vor</i>		<i>Kartennavigi</i>	<i>Funktionen des Kartenfensters</i>
<i>Dekorationen ►</i>	Alt+V + D		<i>Decorating the map</i>
►  <i>Gitter...</i>			<i>Gitter</i>
►  <i>Maßstab...</i>			<i>Maßstab</i>
►  <i>Bild...</i>			<i>Image Decoration</i>
►  <i>Nordpfeil...</i>			<i>Nordpfeil</i>
►  <i>Titelbeschriftung...</i>			<i>Title Label</i>
►  <i>Urheberrechtshinweis...</i>			<i>Copyright Label</i>
►  <i>Layoutausdehnung...</i>			<i>Layout Extents</i>
<i>Vorschaumodus ►</i>			
► <i>Normal</i>			
► <i>Monochrom simulieren</i>			
► <i>Farbenblindheit simulieren (Graustufen)</i>			
► <i>Farbenblindheit simulieren (kein Rot)</i>			
► <i>Farbenblindheit simulieren (kein Grün)</i>			
► <i>Farbenblindheit simulieren (kein Blau)</i>			
 <i>Kartenhinweise anzeigen</i>		<i>Attribute</i>	<i>Eigenschaften Anzeige</i>

Fortsetzung auf der nächsten Seite

Tab. 7.4 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeugk	Referenz
 Neues räumliche Lesezeichen...	Strg+B	Kartennavigi	<i>Bookmarking extents on the map</i>
 Räumliche Lesezeichen anzeigen	Strg+Shift+B	Kartennavigi	<i>Bookmarking extents on the map</i>
 Räumliche Lesezeichenverwaltung anzeigen			<i>Bookmarking extents on the map</i>
 Aktualisieren	F5	Kartennavigi	
Layersichtbarkeit ►			<i>Layerfenster</i>
►  Alle Layer anzeigen	Strg+Shift+U		<i>Layerfenster</i>
►  Alle Layer ausblenden	Strg+Shift+H		<i>Layerfenster</i>
►  Zeige gewählte Layer			<i>Layerfenster</i>
►  Ausblenden von gewählten Layern			<i>Layerfenster</i>
►  Gewählte Layer umschalten			<i>Layerfenster</i>
► Gewählte Layer einzeln umschalten			<i>Layerfenster</i>
►  Abgewählte Layer verbergen			<i>Layerfenster</i>
Bedienfelder ►			<i>Bedienfelder und Werkzeugkästen</i>
Werkzeugkästen ►			<i>Bedienfelder und Werkzeugkästen</i>
Volle Ausdehnung	F11		
Bedienfeldsichtbarkeit umschalten	Strg+Tab		
Nur Karte umschalten	Strg+Umschalt+Tab		

Unter  Linux KDE befinden sich *Bedienfelder ►*, *Werkzeugleisten ►* und *Vollbildmodus umschalten* im Menü *Einstellungen*.

## 7.1.4 Layer



















Das *Layer* Menü bietet eine große Auswahl an Werkzeugen, um neue Datenquellen zu *erstellen*, diese zu einem Proejkt *hinzuzufügen* oder zu *bearbeiten*. Darüber hinaus finden Sie in diesem Menü Werkzeuge, um:

- Einen Layer zu *duplizieren* und so eine Kopie zu erhalten, bei der sie Namen, Stil (Symbologie, Beschriftungen, ...), Verknüpfungen, ... verändern können. Das Duplikat benutzt dabei die gleiche Datenquelle wie der ursprüngliche Layer.
- Layer oder Layer-Gruppen *Kopieren* und in ein anderes Projekt *Einfügen*. Wie bei *Duplizieren* basieren die Layer weiterhin auf der gleichen Datenquelle, die Layer-Eigenschaften in dem neuen Projekt können aber unabhängig davon geändert werden.
- Oder Sie können *Eingebettete Layer und Gruppen...* aus einem anderen Projekt als Nur-Lesen-Kopien einfügen, deren Eigenschaften nur im Ursprungs-Projekt geändert werden können (see *Embedding layers from external projects*).

Das *Layer* Menü enthält auch Werkzeuge, um Layer-Eigenschaften zu kopieren und einzufügen (Symbologie, Maßstab, KBS, ...).



Tab. 7.5: Die Einträge im Layer-Menü

Menüeintrag	Tastenkürzel	Werkzeugleiste	Referenz
 <b>Datenquellenverwaltung</b> Layer erzeugen ►	Strg+L	<i>Datenquellenverwa</i>	<i>Datenquellenverwaltung</i> <i>Neue Vektorlayer erstellen</i>
►  <b>Neuer GeoPackage-Layer...</b>	Strg+Umschalt	<i>Datenquellenverwa</i>	<i>Einen neuen GeoPackage-Layer erstellen</i>
►  <b>Neuer Shapedatei-Layer...</b>		<i>Datenquellenverwa</i>	<i>Einen neuen Shapedatei-Layer erstellen</i>
►  <b>Neuer SpatiaLite-Layer...</b>		<i>Datenquellenverwa</i>	<i>Creating a new SpatiaLite layer</i>
►  <b>Neuer Temporärlayer...</b>		<i>Datenquellenverwa</i>	<i>Creating a new Temporary Scratch Layer</i>
►  <b>Neuer Netzlayer...</b>		<i>Datenquellenverwa</i>	<i>Creating a new Mesh layer</i>
►  <b>New GPX Layer...</b>		<i>Datenquellenverwa</i>	<i>Creating a new GPX layer</i>
►  <b>Neuer virtueller Layer...</b> Layer hinzufügen ►		<i>Datenquellenverwa</i>	<i>Creating virtual layers</i> <i>Öffnen von Daten</i>
►  <b>Vectorlayer hinzufügen...</b>	Strg+Shift+V	<i>Layerverwaltungsw</i>	<i>Loading a layer from a file</i>
►  <b>Rasterlayer hinzufügen...</b>	Ctrl+Shift+R	<i>Layerverwaltungsw</i>	<i>Loading a layer from a file</i>
►  <b>Netzlayer hinzufügen...</b>		<i>Layerverwaltungsw</i>	<i>Loading a mesh layer</i>
►  <b>Getrennte Textdatei als Layer hinzufügen...</b>	Strg+Umschalt	<i>Layerverwaltungsw</i>	<i>Importing a delimited text file</i>
►  <b>PostGIS-Layer hinzufügen...</b>	Strg+Umschalt	<i>Layerverwaltungsw</i>	<i>Database related tools</i>
►  <b>SpatiaLite-Layer hinzufügen...</b>	Strg+Umschalt	<i>Layerverwaltungsw</i>	<i>SpatiaLite Layers</i>
►  <b>MS-SQL-Server-Layer hinzufügen...</b>		<i>Layerverwaltungsw</i>	<i>Database related tools</i>
►  <b>Oracle Spatial-Layer hinzufügen...</b>		<i>Layerverwaltungsw</i>	<i>Database related tools</i>
►  <b>Add SAP HANA Spatial Layer...</b>		<i>Layerverwaltungsw</i>	<i>Database related tools</i>
►  <b>Virtuellen Layer hinzufügen/bearbeiten...</b>		<i>Layerverwaltungsw</i>	<i>Creating virtual layers</i>
►  <b>WMS/WMTS-Layer hinzufügen...</b>	Strg+Umschalt	<i>Layerverwaltungsw</i>	<i>WMS/WMTS Layer laden</i>
►  <b>XYZ-Layer hinzufügen...</b>			<i>Using XYZ Tile services</i>
►  <b>WCS-Layer hinzufügen...</b>		<i>Layerverwaltungsw</i>	<i>WCS Client</i>
►  <b>Add WFS / OGC API - Features Layer...</b>		<i>Layerverwaltungsw</i>	<i>WFS und WFS-T Klient</i>
►  <b>Add ArcGIS REST Server Layer...</b>		<i>Layerverwaltungsw</i>	<i>Using ArcGIS REST Servers</i>
►  <b>Vektorkachellayer hinzufügen...</b>			<i>Using Vector Tiles services</i>
►  <b>Punktwolkenlayer hinzufügen...</b>			<i>Working with Point Clouds</i>
►  <b>Neuen GPX-Layer hinzufügen...</b> Eingebettete Layer und Gruppen...			<i>Introducing GNSS/GPS Data</i> <i>Embedding layers from external projects</i>

Fortsetzung auf der nächsten Seite




Tab. 7.5 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeugleiste	Referenz
Aus Layerdefinitionsdatei hinzufügen...			<i>Layer definition file</i>
 Georeferenzierung...			<i>Georeferencer</i>
 Stil kopieren			<i>Save and Share Layer Properties</i>
 Stil einfügen			<i>Save and Share Layer Properties</i>
 Layer kopieren			
 Layer/Gruppe einfügen			
 Attributtabelle öffnen	F 6	<i>Attribute</i>	<i>Mit Attributtabelle arbeiten</i>
Attributtabelle filtern ►			<i>Mit Attributtabelle arbeiten</i>
►  Attributtabelle öffnen (gewählte Objekte)	Umschalt+F6	<i>Attribute</i>	<i>Mit Attributtabelle arbeiten</i>
►  Attributtabelle öffnen (sichtbare Objekte)	Strg+F6	<i>Attribute</i>	<i>Mit Attributtabelle arbeiten</i>
►  Attributtabelle öffnen (geänderte und neue Objekte)		<i>Attribute</i>	<i>Mit Attributtabelle arbeiten</i>
 Bearbeitungsstatus umschalten		<i>Digitalisierung</i>	<i>Einen vorhandenen Layer editieren</i>
 Layeränderungen speichern		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
 Aktuelle Änderungen ►		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Speichern gewählter Layer		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Verwerfen für gewählter Layer		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Abbruch für gewählter Layer		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Alle Layer speichern		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Alle Layeränderungen verwerfen		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
► Abbruch für alle Layer		<i>Digitalisierung</i>	<i>Änderungen speichern</i>
Speichern als...			<i>Creating new layers from an existing layer</i>
Als Layerdefinitionsdatei speichern...			<i>Layer definition file</i>
 Layer/Gruppe löschen	Strg+D		
 Layer duplizieren			
Maßstabsabhängige Sichtbarkeit des/der Layer setzen			<i>Visibility Scale Selector</i>
KBS von Layer(n) setzen	Strg+Shift+C		<i>Layer Coordinate Reference Systems</i>
Layer-KBS dem Projekt zuweisen			<i>Project Coordinate Reference Systems</i>
Layerereigenschaften...			<ul style="list-style-type: none"> <li>• <i>vector</i></li> <li>• <i>raster</i></li> <li>• <i>mesh</i></li> <li>• <i>point cloud</i></li> <li>• <i>vector tiles</i></li> </ul>
Filter...	Strg+F		<i>Abfrageeditor</i>
 Beschriftung			<i>Labels Properties</i>






Fortsetzung auf der nächsten Seite

Tab. 7.5 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeugleiste	Referenz
 <i>In der Übersicht anzeigen</i>			<i>Overview Panel</i>
 <i>Alle in der Übersicht anzeigen</i>			<i>Overview Panel</i>
 <i>Alle aus Übersicht entfernen</i>			<i>Overview Panel</i>

## 7.1.5 Einstellungen



Tab. 7.6: Die Einträge im Einstellungen-Menü

Menüeintrag	Referenz
<i>Benutzerprofile ►</i>	<i>Working with User Profiles</i>
► <i>Standard</i>	<i>Working with User Profiles</i>
► <i>Aktuelles Profilverzeichnis öffnen</i>	<i>Working with User Profiles</i>
► <i>Neues Profil...</i>	<i>Working with User Profiles</i>
 <i>Stilverwaltung...</i>	<i>The Style Manager</i>
 <i>Benutzerprojektionen...</i>	<i>Eigenes Koordinatenbezugssystem definieren</i>
 <i>Tastenkürzel...</i>	<i>Tastenkürzel</i>
 <i>Oberflächenanpassung...</i>	<i>Anpassung</i>
 <i>Optionen...</i>	<i>Optionen</i>

Unter  Linux KDE finden Sie im Menü *Einstellungen* noch weitere Werkzeuge, wie *Bedienfelder ►*, *Werkzeugleisten ►* und *Vollbildmodus umschalten*.

## 7.1.6 Erweiterungen

Tab. 7.7: Die Einträge im Erweiterungen-Menü



Menüeintrag	Tastenkürzel	Werkzeugleiste	Referenz
 <i>Erweiterungen verwalten und installieren...</i>			<i>Der Erweiterungen Dialog</i>
 „ <i>Python-Konsole</i>	Strg+Alt+P	<i>Erweiterungswerk:</i>	<i>QGIS Python Konsole</i>

Wenn Sie QGIS das erste Mal starten werden nicht alle Erweiterungen geladen.

## 7.1.7 Vektor

So sieht das Menü *Vektor* aus, wenn alle Kernerweiterungen aktiviert sind.

Tab. 7.8: Die Einträge im Vektor-Menü

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
 <i>Geometrien prüfen...</i>			<i>Geometry Checker Plugin</i>
 <i>Topology-Prüfung</i>		<i>Vektor</i>	<i>Topology Checker Plugin</i>















Fortsetzung auf der nächsten Seite

Tab. 7.8 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
<b>Geoverarbeitungswerkzeuge ▶</b>	Alt+O + G		
▶  Puffer...			<i>Buffer</i>
▶  Zuschneiden...			<i>Clip</i>
▶  Konvexe Hülle...			<i>Convex hull</i>
▶  Differenz...			<i>Difference</i>
▶  Auflösen...			<i>Dissolve</i>
▶  Verschneidung...			<i>Intersection</i>
▶  Symmetrische Differenz...			<i>Symmetrical difference</i>
▶  Vereinigungen...			<i>Union</i>
▶  Ausgewählte Polygone entfernen...			<i>Eliminate selected polygons</i>
<b>Geometrie-Werkzeuge ▶</b>	Alt+O + E		
▶  Zentroide...			<i>Centroids</i>
▶  Geometrien sammeln...			<i>Collect geometries</i>
▶  Stützpunkte extrahieren...			<i>Extract vertices</i>
▶  Mehr- zu einteilig...			<i>Multipart to singleparts</i>
▶  Polygone zu Linien...			<i>Polygons to lines</i>
▶  Vereinfachen...			<i>Simplify</i>
▶  Gültigkeit prüfen...			<i>Check validity</i>
▶  Delaunay-Triangulation...			<i>Delaunay triangulation</i>
▶  Nach Anzahl verdichten...			<i>Densify by count</i>
▶  Geometrieattribute hinzufügen...			<i>Add geometry attributes</i>
▶  Linien zu Polygonen...			<i>Lines to polygons</i>
▶  Voronoi-Polygone...			<i>Voronoi polygons</i>
<b>Analyse-Werkzeuge ▶</b>	Alt+O + A		
▶  Linienschnittpunkte...			<i>Line intersections</i>
▶  Mittlere Koordinate(n)...			<i>Mean coordinate(s)</i>
▶  Grundstatistik für Felder...			<i>Grundstatistik für Felder</i>
▶  Punkte in Polygon zählen...			<i>Count points in polygon</i>
▶  Distanzmatrix...			<i>Distance matrix</i>
▶  Eindeutige Werte auflisten...			<i>List unique values</i>
▶  Nächster Nachbaranalyse...			<i>Nearest neighbour analysis</i>
▶  Linienlängen summieren...			<i>Sum line lengths</i>
<b>Datenmanagement-Werkzeuge ▶</b>	Alt+O + D		
▶  Vektorlayer zusammenführen...			<i>Merge vector layers</i>
▶  Layer reprojizieren...			<i>Reproject layer</i>
▶  Räumlichen Index erzeugen...			<i>Create spatial index</i>

Fortsetzung auf der nächsten Seite

Tab. 7.8 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
►  <i>Attribute nach Position verknüpfen...</i>			<i>Join attributes by location</i>
►  <i>Vektorlayer teilen...</i>			<i>Split vector layer</i>
<i>Forschungswerkzeuge</i> ►	Alt+O+R		
►  <i>Gitter erzeugen...</i>			<i>Create grid</i>
►  <i>Layerausdehnung extrahieren...</i>			<i>Extract layer extent</i>
►  <i>Zufällige Punkte in Grenzen...</i>			<i>Random points in extent</i>
►  <i>Zufällige Punkte in Polygonen...</i>			<i>Random points in polygons</i>
►  <i>Zufällige Punkte aus Linien...</i>			<i>Random points on lines</i>
►  <i>Nach Position selektieren...</i>			<i>Auswahl nach der Lage</i>
►  <i>Select Within Distance...</i>			<i>Select within distance</i>
►  <i>Zufällige Punkte in Layergrenzen...</i>			<i>Random points in layer bounds</i>
►  <i>Zufällige Punkte in Polygonen...</i>			<i>Random points inside polygons</i>
►  <i>Zufällige Auswahl...</i>			<i>Random selection</i>
►  <i>Zufällige Auswahl in Untermengen...</i>			<i>Random selection within subsets</i>
►  <i>Regelmäßige Punkte...</i>			<i>Regular points</i>

Standardmäßig fügt QGIS *Verarbeitungsalgorithmen* zum *Vektor* Menü hinzu, gruppiert nach Untermenüs. Dies bietet Tastaturkürzel für viele gängige vektorbasierte GIS-Aufgaben von verschiedenen Entwicklern. Wenn nicht alle diese Untermenüs verfügbar sind, aktivieren Sie das Processing Plugin in *Erweiterungen* ► *Erweiterungen verwalten und installieren...*

Beachten Sie, dass die Liste der Algorithmen und ihr Menü mit beliebigen Verarbeitungs-Algorithmen (s. *Configuring the Processing Framework*) oder einigen externen ref:Erweiterungen <plugins>.geändert/erweitert werden kann.

## 7.1.8 Raster


















Dies sind die Punkte, die Sie im *Raster* Menü finden, wenn alle Kernerweiterungen aktiviert sind.

Tab. 7.9: Die Einträge im Raster-Menü

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
 <i>Rasterrechner...</i>			<i>Rasterrechner</i>
<i>Align Rasters...</i>			<i>Align rasters</i>
<i>Analyse</i> ►			
►  <i>Perspektive...</i>			<i>Aspect</i>
►  <i>„Leerwert“ füllen...</i>			<i>Fill NoData</i>
►  <i>Gitter (Beweglicher Mittelwert)...</i>			<i>Grid (Moving average)</i>
►  <i>Gitter (Datenkriterien)...</i>			<i>Grid (Data metrics)</i>
►  <i>Gitter (Inverse Distanz zu einer Potenz)...</i>			<i>Grid (Inverse distance to a power)</i>
►  <i>Gitter (Nächster Nachbar)...</i>			<i>Grid (IDW with nearest neighbor searching)</i>

Fortsetzung auf der nächsten Seite

Tab. 7.9 – Fortsetzung der vorherigen Seite

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
►  Fast schwarz...			<i>Near black</i>
►  Schummerung...			<i>Hillshade</i>
►  Nähe (Rasterabstand)...			<i>Proximity (raster distance)</i>
►  Rauhigkeit...			<i>Roughness</i>
►  Sieben...			<i>Sieve</i>
►  Hang...			<i>Slope</i>
►  Topographische Position (TPI)...			<i>Topographic Position Index (TPI)</i>
►  Oberflächenrauigkeit (TRI)...			<i>Terrain Ruggedness Index (TRI)</i>
Projektionen ►			
►  Projektion zuweisen...			<i>Projektion zuweisen</i>
►  Projektion extrahieren...			<i>Projektion extrahieren</i>
►  Transformieren (Reprojizieren)...			<i>Warp (reproject)</i>
Sonstiges ►			
►  Virtuelles Raster generieren...			<i>Build virtual raster</i>
►  Rasterinformation...			<i>Raster information</i>
►  Verschmelzen...			<i>Merge</i>
►  Übersichten erzeugen (Pyramiden)...			<i>Build overviews (pyramids)</i>
►  Kachelindex...			<i>Tile index</i>
Extraktion ►			
►  Raster auf Ausdehnung zuschneiden...			<i>Clip raster by extent</i>
►  Raster auf Layermaske zuschneiden...			<i>Clip raster by mask layer</i>
►  Kontur...			<i>Contour</i>
Konvertierung ►			
►  PCT nach RGB...			<i>PCT to RGB</i>
►  Vektorisieren (Raster nach Vektor)...			<i>Polygonize (raster to vector)</i>
►  Rastern (Vektor nach Raster)...			<i>Rasterize (vector to raster)</i>
►  RGB nach PCT...			<i>RGB to PCT</i>
►  Umwandeln (Format konvertieren)...			<i>Translate (convert format)</i>

Standardmäßig fügt QGIS *Verarbeitungsalgorithmen* zum *Vektor* Menü hinzu, gruppiert nach Untermenüs. Dies bietet Tastaturkürzel für viele gängige vektorbasierte GIS-Aufgaben von verschiedenen Entwicklern. Wenn nicht alle diese Untermenüs verfügbar sind, aktivieren Sie das Processing Plugin in *Erweiterungen ► Erweiterungen verwalten und installieren...*

Beachten Sie, dass die Liste der Algorithmen und ihr Menü mit beliebigen Verarbeitungs-Algorithmen (s. *Configuring the Processing Framework*) oder einigen externen ref:Erweiterungen <plugins>.geändert/erweitert werden kann.

## 7.1.9 Datenbank

Dies sind die Punkte, die Sie im *Datenbank* Menü finden, wenn alle Kernerweiterungen aktiviert sind. Wenn keine Datenbank-Erweiterung aktiviert ist, ist das *Datenbank* Menü nicht sichtbar.

Tab. 7.10: Die Einträge im Datenbank-Menü


Menüeintrag	Tastenkürzel	Werkzeugleist	Referenz
<i>Offline-Bearbeitung</i> →	Alt+D + O		<i>Offline Editing Plugin</i>
►  <i>In Offline-Projekt konvertieren...</i>		<i>Datenbank</i>	<i>Offline Editing Plugin</i>
►  <i>Synchronisieren</i>		<i>Datenbank</i>	<i>Offline Editing Plugin</i>
 <i>DB-Verwaltung...</i>		<i>Datenbank</i>	<i>DB Manager Plugin</i>

Wenn Sie QGIS das erste Mal starten werden nicht alle Erweiterungen geladen.

## 7.1.10 Web

Dies sind die Punkte, die Sie im *Web* Menü finden, wenn alle Kernerweiterungen aktiviert sind. Wenn keine Web-Erweiterung aktiviert ist, ist das *Datenbank* Menü nicht sichtbar.

Tab. 7.11: Die Einträge im Web-Menü



Menüeintrag	Tastenkürzel	Werkzeugleist	Referenz
<i>MetaSearch</i> ►	Alt+W + M		<i>MetaSearch Catalog Client</i>
►  <i>Metasearch</i>		<i>Web</i>	<i>MetaSearch Catalog Client</i>
► <i>Help</i>			<i>MetaSearch Catalog Client</i>

Wenn Sie QGIS das erste Mal starten werden nicht alle Erweiterungen geladen.

## 7.1.11 Netz


Das *Netz* Menü bietet Werkzeuge, die für die Bearbeitung von *Netz-Layern* benötigt werden. Erweiterungen können diesem Menü neue Punkte hinzufügen.

Tab. 7.12: Die Einträge im Netz-Menü

Menüeintrag	Tastenkürzel	Werkzeugleist	Referenz
 <i>Netzrechner...</i>			<i>Mesh Calculator</i>
 <i>Oberflächen und Stützpunkte reindizieren</i>			<i>Reindexing meshes</i>

## 7.1.12 Verarbeitung





Tab. 7.13: Die Einträge im Verarbeitung-Menü

Menüeintrag	Tastenkürzel	Werkzeug	Referenz
 <i>Werkzeugkiste</i>	Ctrl+Alt+T		<i>The Toolbox</i>
 <i>Model Designer...</i>	Strg+Alt+G		<i>The model designer</i>
 <i>Protokoll...</i>	Strg+Alt+H		<i>Das Protokoll</i>
 <i>Ergebnisanzeige</i>	Strg+Alt+R		<i>Konfiguration externer Anwendungen</i>
 <i>Objekte insitu ändern</i>			<i>Mit Verarbeitung Objekte innerhalb eines Layers verändern</i>

Wenn Sie QGIS das erste Mal starten werden nicht alle Erweiterungen geladen.

## 7.1.13 Hilfe

Tab. 7.14: Die Einträge im Hilfe-Menü

Menüeintrag	Tastenkürzel	Werkzeugleist	Referenz
 <i>Hilfe-Übersicht</i>	F1		<i>Direkthilfe</i>
<i>API Dokumentation</i>			
<i>Erweiterungen ►</i>			
<i>Ein Problem melden</i>			
<i>Brauchen Sie professionelle Unterstützung?</i>			
 <i>QGIS-Homepage</i>	Strg+H		
 <i>Check QGIS-Version überprüfen</i>			
 <i>Über</i>			
 <i>Dauerhafte Fördermitgliedschaften</i>		<i>QGIS-</i>	

## 7.1.14 QGIS


Dieses Menü ist nur unter **X** macOS verfügbar und enthält einige betriebssystem-spezifische Befehle.

Tab. 7.15: Die Einträge im QGIS-Menü

Menüeintrag	Tastenkürzel
<i>Preferences</i>	
<i>Über QGIS</i>	
<i>QGIS verbergen</i>	
<i>Show All</i>	
<i>Andere verbergen</i>	
<i>QGIS beenden</i>	Cmd+Q

*Einstellungen* entspricht *Einstellungen ► Optionen*, *Über QGIS* entspricht *Hilfe ► Über* und *QGIS beenden* entspricht *Projekt ► QGIS beenden* in anderen Betriebssystemen.

## 7.2 Bedienfelder und Werkzeugkästen

Über das Menü *Ansicht* (oder  *Einstellungen*) können Sie (*Bedienfelder ►*) und Werkzeugleisten (*Werkzeugkästen ►*) ein- und ausschalten. Bedienfelder und Werkzeugleisten können ebenfalls (de)aktiviert werden, indem Sie mit der rechten Maustaste auf die Menü- oder Symbolleiste klicken und das gewünschte Element auswählen. Beides läßt sich innerhalb der QGIS-Oberfläche beliebig verschieben und platzieren. Die Liste kann durch die Aktivierung von Kern- oder externen *Erweiterungen* erweitert werden.

### 7.2.1 Werkzeugleisten

Über die Werkzeugleisten haben Sie Zugriff auf die meisten Funktionen in den Menüs sowie auf zusätzliche Werkzeuge zur Interaktion mit der Karte. Zu jedem Element der Werkzeugleiste gibt es eine Pop-up-Hilfe: fahren Sie mit der Maus über das Element und es wird eine kurze Beschreibung des Werkzeugs angezeigt.


In der folgenden Tabelle sind die verfügbaren Werkzeugleisten aufgeführt:

Tab. 7.16: QGIS Werkzeugleisten

Name der Werkzeugleiste	Wichtigste Referenz für das Werkzeug
<i>Erweiterte Digitalisierung</i>	<i>Erweiterte Digitalisierung</i>
<i>Beschriftung</i>	<i>Beschriftungstools</i>
<i>Attribute</i>	<i>Mit Attributtabelle arbeiten, Allgemeine Werkzeuge</i>
<i>Datenquellenverwaltungsleiste</i>	<i>Verwaltung von Datenquellen</i>
<i>Datenbank</i>	<i>DB Manager Plugin</i>
<i>Digitalisierung</i>	<i>Einen vorhandenen Layer editieren</i>
<i>GPS</i>	<i>Live GPS tracking</i>
<i>GRASS</i>	<i>GRASS GIS Integration</i>
<i>Direkthilfe</i>	
<i>Beschriftung</i>	<i>The Label Toolbar</i>
<i>Layerverwaltungswerkzeugleiste</i>	<i>Öffnen von Daten</i>
<i>Kartennavigation</i>	<i>Funktionen des Kartenfensters</i>
<i>Netzdigitalisierung</i>	<i>Editing a mesh layer</i>
<i>Erweiterungswerkzeugleiste</i>	<i>Erweiterungen</i>
<i>Projekt</i>	<i>Arbeiten mit Projektdateien, Kartenlayout, Die Stilbibliothek</i>
<i>Processing Algorithms</i>	<i>Configuring the Processing Framework</i>
<i>Raster</i>	<i>Erweiterungen</i>
<i>Auswahl</i>	<i>Selecting features</i>
<i>Werkzeugleiste für Formen</i>	<i>Formen digitalisieren</i>
<i>Einrastwerkzeugleiste</i>	<i>Einrasttoleranz und Suchradius einstellen</i>
<i>Vektor</i>	<i>Erweiterungen</i>
<i>Web</i>	<i>Erweiterungen, MetaSearch Catalog Client</i>

**Bemerkung:** Externe Erweiterungen können die Standard-Werkzeugleisten um eigene Werkzeuge ergänzen oder auch ganz neue Werkzeugleisten zu QGIS hinzufügen.

#### Tipp: Werkzeugleiste wiederherstellen

Wenn Sie versehentlich eine Werkzeugleiste ausgeblendet haben, können Sie diese mit *Ansicht ► Werkzeugkästen ►* (oder  *Einstellungen ► Werkzeugkästen ►*) wieder einblenden. Wenn aus irgendeinem Grund eine Werkzeugleiste (oder ein anderes Element) komplett von der Oberfläche verschwunden ist, finden Sie Tipps, wie Sie sie wiederherstellen können, unter *Wiederherstellung der ursprünglichen GUI*.



## 7.2.2 Bedienfelder

QGIS bietet viele Bedienfelder. Bedienfelder sind spezielle Elemente, mit denen Sie interagieren können (Auswahl von Optionen, Aktivieren von Kästchen, Eintragen von Werten...), um komplexere Aufgaben durchzuführen.

Die folgende Liste zeigt die Standard Bedienfelder, die von QGIS zur Verfügung gestellt werden:

Tab. 7.17: Bedienfelder

Name des Bedienfelds	Tastenkürzel	Referenz
<i>Erweiterte Digitalisierung</i>	Strg+4	<i>Das Bedienfeld Erweiterte Digitalisierung</i>
<i>Browser</i>	Strg+2	<i>The Browser Panel</i>
<i>Browser (2)</i>		<i>The Browser Panel</i>
<i>Diagnose-/Entwicklungswerkzeuge</i>	F12	<i>Debugging/Development Tools Panel</i>
<i>Höhenprofil</i>		
<i>Geometrie Überprüfung</i>		<i>Eigenschaften Digitalisieren</i>
<i>GPS-Information</i>	Strg+0	<i>Live GPS tracking</i>
<i>GRASS Werkzeuge</i>		<i>GRASS GIS Integration</i>
<i>Layerreihenfolge</i>	Strg+9	<i>Layer Order Panel</i>
<i>Layergestaltung</i>	Strg+3	<i>Layer Styling Panel</i>
<i>Layer</i>	Strg+1	<i>Layerfenster</i>
<i>Protokoll</i>		<i>Log Messages Panel</i>
<i>Übersicht</i>	Strg+8	<i>Overview Panel</i>
<i>Verarbeitungswerkzeuge</i>		<i>The Toolbox</i>
<i>Ergebnisanzeige</i>		<i>The Toolbox</i>
<i>Fangen- und Digitalisierungsoptionen</i>		<i>Einrasttoleranz und Suchradius einstellen</i>
<i>Räumliche Lesezeichenverwaltung</i>	Strg+7	<i>Bookmarking extents on the map</i>
<i>Statistik</i>	Strg+6	<i>Statistical Summary Panel</i>
<i>Zeitsteuerung</i>		<i>Das Zeitsteuerung-Bedienfeld</i>
<i>Kachelmaßstab</i>		<i>Tilesets</i>
<i>Rücknahme/Wiederholung</i>	Strg+5	<i>Undo/Redo Panel</i>
<i>Stützpunktbearbeitung</i>		<i>Das Bedienfeld Stützpunktbearbeitung</i>

## 7.3 Statusleiste

In der Statusleiste finden Sie allgemeine Informationen über die Kartenansicht, über den Status während einer Verarbeitung und zu verfügbaren Aktionen. Außerdem stehen Ihnen Werkzeuge zur Steuerung der Kartenansicht zur Verfügung.

### 7.3.1 Suchleiste

Auf der linken Seite der Statusleiste befindet sich die Suchleiste, ein Werkzeug für die Schnellsuche, mit dem Sie jedes Objekt oder jede Option in QGIS finden und ausführen können:

1. Klicken Sie in das Textfeld, um die Suchleiste zu aktivieren oder drücken Sie **Strg+K**.
2. Geben Sie einen Text ein, der mit dem gesuchten Objekt verknüpft ist (Name, Tag, Schlüsselwort, ...). Standardmäßig werden Ergebnisse für alle aktivierten Suchmuster ausgegeben, aber Sie können die Suche auf einen bestimmten Bereich beschränken, indem Sie für Ihrem Text einen *Such-Filter* in Form eines voranstellen Präfix festlegen. Bei der Eingabe von `! cad` wird z.B. nur Layer zurückgeben, deren Name `cad` enthält.

Der Such-Filter kann auch mit einem Doppelklick im dem Menü ausgewählt werden, das bei einer Eingabe in das Textfeld angezeigt wird.

3. Klicken Sie auf das gewünschte Ergebnis, um eine entsprechende Aktion (je nach Art des Elements) auszuführen.

---

**Tipp: Beschränkung der Suche auf bestimmte Felder in der Attributtabelle des aktiven Layers**

Standardmäßig durchläuft eine Suche mit dem Filter „Objekte des aktiven Layers“ (f) die gesamte Attributtabelle des betreffenden Layers. Sie können die Suche auf ein bestimmtes Feld beschränken, indem Sie das Präfix @ verwenden. So gibt z.B. f @name sal oder @name sal gibt nur die Objekte zurück, deren Attribut „name“ die Zeichenfolge ‚sal‘ enthält. Die automatische Textvervollständigung ist beim Schreiben aktiv und der Vorschlag kann mit der Taste Tab übernommen werden.

Eine erweiterte Kontrolle über die abgefragten Felder ist über den Reiter *Felder* der Layereigenschaften möglich. Lesen Sie das Kap. [Eigenschaften Felder](#) für Details.

---

Die Suche wird über Threads abgewickelt, so dass die Ergebnisse angezeigt werden können, sobald ein Filter auf sie stößt, auch wenn die Suche noch anhält. Das bedeutet, dass z. B. ein Dateisuchfilter die Ergebnisse nacheinander anzeigt, während der Dateibaum noch durchsucht wird. Dadurch wird sichergestellt, dass die Benutzeroberfläche immer ansprechbar ist, auch wenn eine zeitintensive Suche durchgeführt wird (z. B. wenn ein Online-Dienst durchsucht wird).

---


**Bemerkung:** Der Nominatim-Suchfilter verhält sich möglicherweise anders (keine automatische Vervollständigungssuche, Verzögerung beim Abrufen von Ergebnissen, ...) da die [OpenStreetMap-Nominatim-Nutzungsrichtlinie](#) Anwendung finden.

---



---

**Tipp: Schneller Zugriff auf die Konfigurationen des Locators**

Klicken Sie auf das  Suchen-Symbol im Textfeld der Suchleiste, um die Liste der Filter anzuzeigen, die Sie verwenden können oder um über den *Konfigurieren* Eintrag die Registerkarte *Suche* aus dem Menü *Einstellungen* ► *Optionen...* direkt zu öffnen.



---

## 7.3.2 Anzeige von Aktionen


Im Bereich neben der Suchleiste werden ggf. Informationen zu durchgeführten Aktionen angezeigt (z. B. Auswahl von Objekten in einem Layer, Entfernen von Layern, Entfernung und Richtung beim Verschieben) oder eine lange Beschreibung des Werkzeugs, über das Sie den Mauszeiger halten (nicht für alle Werkzeuge verfügbar).

Bei langwierigen Operationen, wie z. B. dem Erstellen von Statistiken von Rasterlayern, dem Ausführen von Verarbeitungsalgorithmen oder dem Rendern mehrerer Layer in der Kartenansicht, wird ein Fortschrittsbalken in der Statusleiste angezeigt.

## 7.3.3 Steuerung der Kartenansicht


Die Funktion  *Koordinate* zeigt die aktuelle Position der Maus an und ändert sich, wenn Sie den Mauszeiger über die Kartenansicht bewegen. Sie können die Einheiten (und die Genauigkeit) dieser Anzeige im Menü *Projekt* ► *Eigenschaften...* ► *Allgemein* einstellen. Wenn Sie auf die kleine Schaltfläche links neben dem Textfeld klicken, können Sie zwischen den Optionen *Koordinate* und  *Ausdehnung* umschalten. Für die Ausdehnung werden die aktuellen Koordinaten der unteren linken und oberen rechten Ecke der Kartenansicht in den eingestellten Karteneinheiten angezeigt.



Neben der Koordinatenanzeige wird der jeweils aktuelle *Maßstab* der Kartenansicht angezeigt. Es gibt einen Maßstabsselektor, mit dem Sie zwischen *vordefinierten Maßstäben* wählen können.

Klicken Sie auf der rechten Seite der Maßstabsanzeige auf die Schaltfläche , um den Maßstab zu sperren. Wenn Sie jetzt mit Lupe oder Mausekze die Ansicht Vergrößern oder Verkleinern können Sie in die Kartenansicht hineinzoomen, ohne den Maßstab zu verändern. Das ist hilfreich, um z.B. die Positionen von Beschriftungen und Symbolen genauer

sehen und einstellen zu können. Die Vergrößerungsstufe wird neben dem Maßstab in Prozent angegeben. Wenn die *Vergrößerung* einen Wert von 100% hat, wird die aktuelle Karte nicht vergrößert, d.h. sie wird maßstabsgetreu im Verhältnis zur Auflösung des Monitors (DPI) dargestellt. Ein Standardwert für die Vergrößerung kann unter *Einstellungen ► Optionen ► Darstellung ► Zeichenverhalten* definiert werden, was bei hochauflösenden Bildschirmen sehr nützlich sein kann, um kleine Symbole zu vergrößern. Zusätzlich kann unter *Einstellungen ► Optionen ► Karte & Legende ► DPI* eingestellt werden, ob QGIS die physikalische DPI des jeweiligen Monitors berücksichtigt oder die logische DPI des Gesamtsystems verwendet.


Rechts neben dem Vergrößerungswerkzeug können Sie eine Drehung (im Uhrzeigersinn) für die Kartenansicht in Grad festlegen.

Auf der rechten Seite der Statusleiste kann das Kontrollkästchen  *Zeichnen* verwendet werden, um das Rendering der Kartenansicht vorübergehend auszusetzen (siehe Abschnitt [Controlling map rendering](#)).

Rechts neben dem  *Zeichnen* Kontrollkästchen befindet sich die Schaltfläche  *EPSG:code*, die das aktuelle Projekt-KBS anzeigt. Wenn Sie darauf klicken, öffnet sich die *Projekteigenschaften* mit dem Reiter *KBS* und Sie können die Kartenansicht neu projizieren (oder auch andere Projekteigenschaften anpassen).

---


### **Tipp: Die richtige Maßstabseinheit im Kartenfenster einstellen**

Wenn Sie QGIS starten, ist das Standard KBS WGS 84 (EPSG 4326) und die Einheiten sind Grad. Das bedeutet, dass QGIS die Koordinate in Grad anzeigt. Wenn Sie eine andere Anzeige wünschen (oder auch um ggf. eine korrekte Anzeige zu erhalten) können Sie diese Einstellung entweder manuell im Menü *Projekt ► Eigenschaften...* unter dem Reiter *Allgemein* ändern (z.B. in Meter). Oder Sie können über das Symbol  *EPSG:code* (siehe oben) das korrekte KBS einstellen. Dadurch werden die Einheiten so gesetzt, wie es die Projektion vorgibt (z.B. `+units=us-ft`).

Beachten Sie, dass das Standard-KBS für neu angelegte Projekte in *Einstellungen ► Optionen ► KBS und Transformationen ► KBS-Behandlung* eingestellt werden kann.

---

## **7.3.4 Meldungen**

Die Schaltfläche  *Meldungen* ganz rechts in der Statusleiste öffnet das *Protokoll* Bedienfeld, das Informationen über Hintergrundprozesse enthält (QGIS-Start, Laden von Plugins, Verarbeitungswerkzeuge...).



---

### Das Browser Bedienfeld

---

- *Ressourcen, die über den Browser geöffnet/ausgeführt werden können*
- *Die obersten Einträge des Brower-Bedienfelds*
  - *Favoriten*
  - *Räumliche Lesezeichen*
  - *Projektverzeichnis*
  - *Laufwerke und Dateisystem*
  - *Datenbankeinträge*
  - *Kacheln und Internetdienste*
- *Datenquellen und Dateien*

Das QGIS Browser Bedienfeld ist ein sehr gutes Werkzeug zum Suchen, Prüfen, Kopieren und Laden von QGIS-Ressourcen. In diesem Bedienfeld werden nur Ressourcen angezeigt, die von QGIS genutzt werden können.

Mit dem *Browser* Bedienfeld können Sie Daten finden, untersuchen und hinzufügen, näheres finden Sie in dem Kapitel *The Browser Panel*. Darüber hinaus unterstützt das *Browser* Bedienfeld das Ziehen und Ablegen vieler QGIS-Ressourcen, wie Projektdateien, Python-Skripte, Processing-Skripte und Processing-Modelle.

Python-Skripte, Processing-Skripte und Processing-Modelle können auch zur Bearbeitung in einem externen Editor bzw. dem grafischen Modellierer geöffnet werden.

Sie können Layer aus dem *Layer* Bedienfeld in das *Browser* Bedienfeld ziehen und ablegen, zum Beispiel in ein GeoPackage oder eine PostGIS-Datenbank.

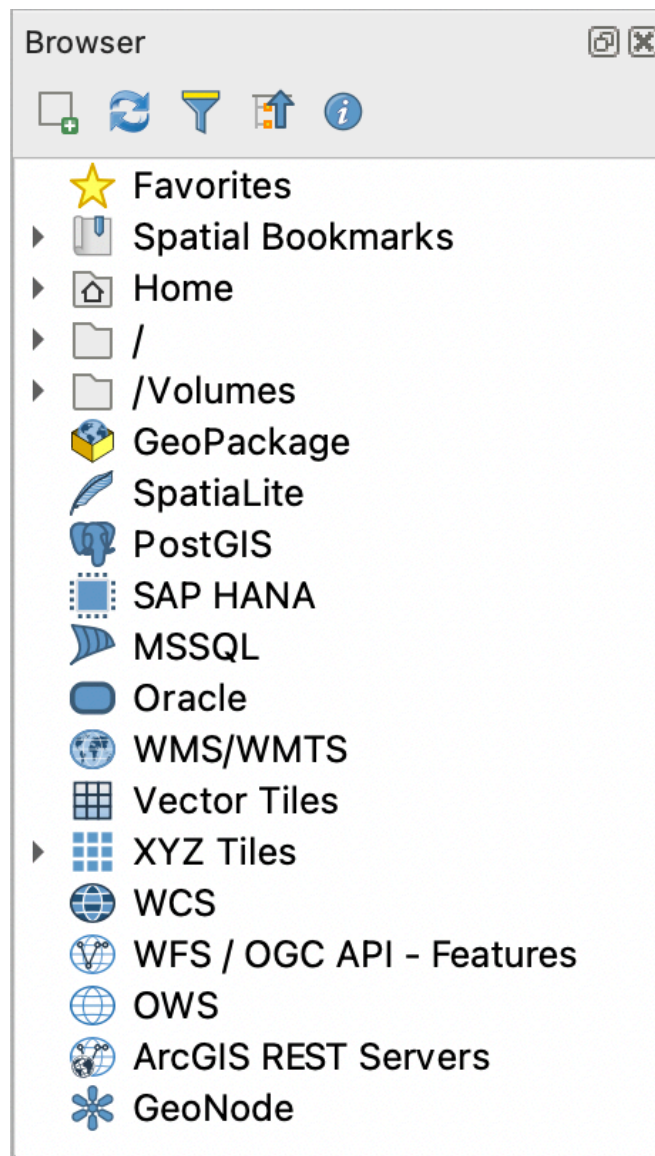





Abb. 8.1: Das Browser Bedienfeld

Das Browser-Bedienfeld (Abb. 8.1) ist als erweiterbarer, hirachischer Ordnerbaum, mit einigen festen Einträgen auf oberster Ebene, organisiert, die die vom Browser verwalteten Ressourcen ordnen. Einträge werden durch Klicken auf


links neben dem Eintragsnamen erweitert. Ein Zweig wird durch Klicken auf eingeklappt. Die Schaltfläche  Alle einklappen reduziert alle Einträge auf die oberste Ebene.

Im Menü *Einstellungen* ► *Oberflächenanpassung* ist es möglich, die Anzeige von Ressourcen zu deaktivieren. Wenn Sie z.B. keine Python-Skripte im Browser anzeigen möchten, können Sie den Eintrag *Browser* ► *py* deaktivieren, und wenn Sie Ihr Home-Verzeichnis im Browser loswerden möchten, können Sie den Eintrag *Browser* ► *special:Home* deaktivieren.

Ein Filter (  Filter Browser) kann für die Suche auf Basis von Eintragsnamen verwendet werden. Wenn ein Filter gesetzt ist, erscheint eine Schaltfläche mit  Optionen - mit dem Pulldown-Menü hier können Sie:

- *Groß-/Kleinschreibung beachten* ein- und ausschalten
- Die Syntax des Filters auf einen der folgenden Werte setzen:
  - *Normal*
  - *Platzhalter*

## – Regulärer Ausdruck

Über die Schaltfläche  Eigenschaften-Element ein-/ausschalten kann ein Werkzeug aktiviert werden, welches am unteren Rand des Bedienfelds nützliche Informationen über Einträge und Ressourcen anzeigt. Wenn es aktiviert ist (und Informationen verfügbar sind), öffnet es sich wie in Abb. 8.2 gezeigt.

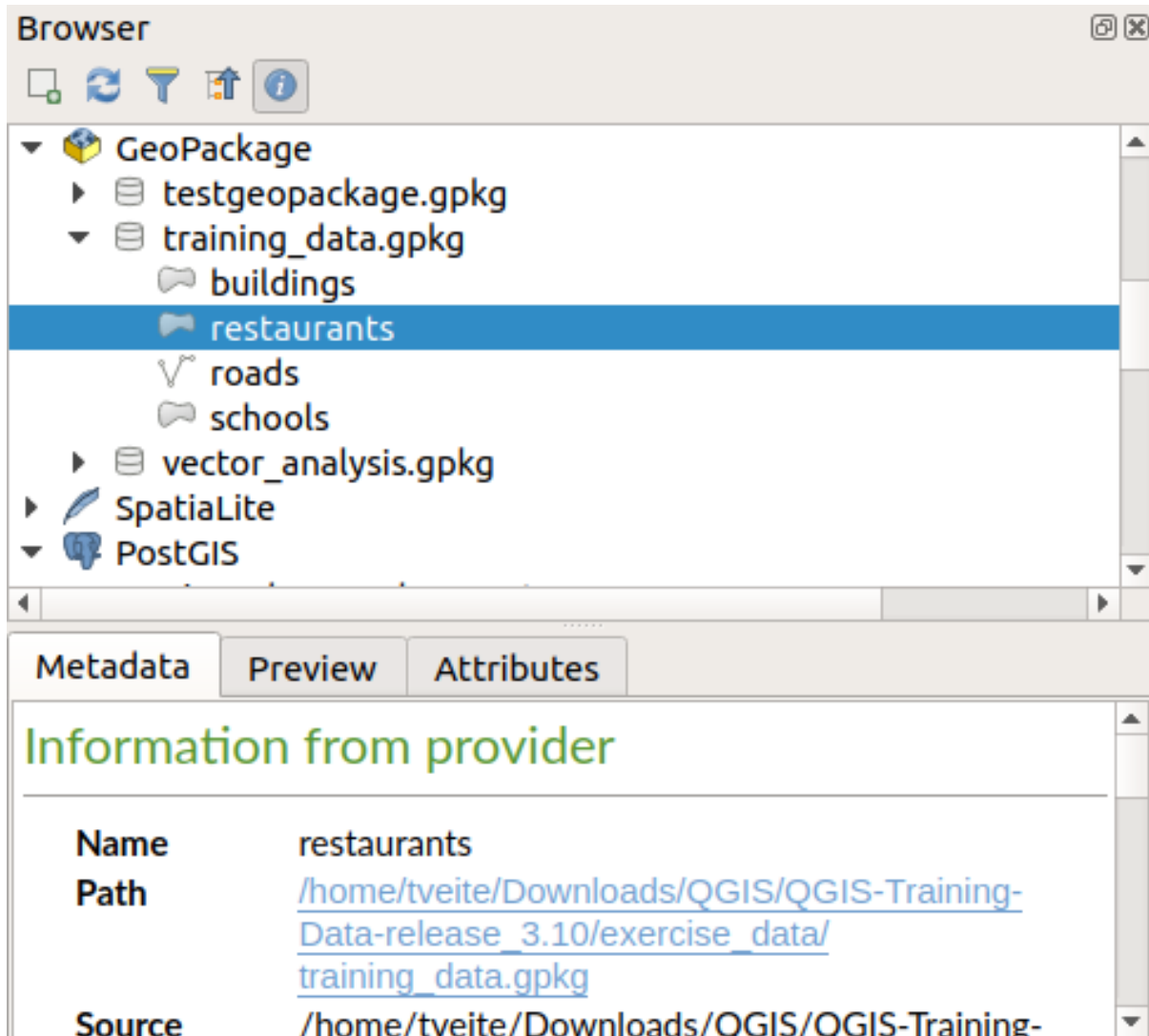



Abb. 8.2: The properties widget

Ein zweites Browser-Bedienfeld kann durch Aktivieren des *Browser (2)* Bedienfelds im Menü *Ansicht ► Bedienfelder* geöffnet werden. Ein zweites Bedienfeld kann nützlich sein, wenn Layer zwischen Ressourcen kopiert werden sollen, die sich tief unten in verschiedenen Zweigen der Browser-Hierarchie befinden.

## 8.1 Ressourcen, die über den Browser geöffnet/ausgeführt werden können

Im Bedienfeld des Browsers können viele Aufgaben bewältigt werden:

- Fügen Sie Vektor-, Raster- und Mesh-Layer zu Ihrer Kartenansicht hinzu, indem Sie darauf doppelklicken, diese auf die Kartenansicht ziehen oder, nach Auswahl des Layers, auf die Schaltfläche  Gewählte Layer hinzufügen klicken
- Ausführen von Python-Skripten (einschließlich Verarbeitungsalgorithmen) durch Doppelklick oder Ziehen auf die Kartenansicht
- Modelle durch Doppelklick oder Ziehen auf die Kartenansicht ausführen
- Aus QGIS-Projektdateien über das Kontextmenü *Symbole extrahieren...*
- Öffnen Sie Dateien mit ihren Standardanwendungen (im Kontextmenü *<file type> extern öffnen...*). Beispiele: HTML-Dateien, Tabellenkalkulationen, Bilder, PDFs, Textdateien, ...
- Einträge kopieren
- Umbenennen und Löschen (mehrerer) Layer (Kontextmenü: *Verwalten ►*)
- Mit *In Dateien anzeigen* aus dem Kontextmenü können sie ein Datei-Explorer-Fenster des entsprechenden Ordners öffnen

Darüber hinaus kann es im Kontextmenü ressourcenspezifische Aktionen geben, z.B. „Projekt öffnen“ oder „Layer exportieren“.

## 8.2 Die obersten Einträge des Brower-Bedienfelds

### 8.2.1 Favoriten

Häufig verwendete Speicherorte können über das Kontextmenü als Favoriten gekennzeichnet werden. Diese Speicherorte werden dann hier angezeigt.

Ebenfalls über das Kontextmenü kann mit *Favorit umbenennen...* ein Alias vergeben oder mit *Favorit entfernen* wieder aus den Favoriten gelöscht werden.

### 8.2.2 Räumliche Lesezeichen

Hier finden Sie ihre räumlichen Lesezeichen, untergliedert in *Benutzerlesezeichen* und *Projektlesezeichen*.

Mit einem Rechtsklick auf „räumliche Lesezeichen“ können Sie ein *Neues räumliches Lesezeichen...* erstellen, die *Räumliche Lesezeichenverwaltung anzeigen* lassen, *Räumliche Lesezeichen importieren...* oder *Räumliche Lesezeichen exportieren...*

Mit dem Kontextmenü für Lesezeichengruppen können Sie *Räumliche Lesezeichen exportieren...*, ein (*Neues räumliches Lesezeichen...*) erstellen, *Lesezeichengruppe umbenennen*, und *Lesezeichengruppe löschen*.

Für Lesezeicheneinträge haben Sie im Kontextmenü die Optionen *Zum Lesezeichen zoomen*, *Räumliche Lesezeichen bearbeiten...* und *Räumliche Lesezeichen löschen*



### 8.2.3 Projektverzeichnis

Nachdem die Projektdatei zum ersten Mal gespeichert wurde, steht im *Browser*-Bedienfeld der *Projektverzeichnis* Eintrag zur Verfügung. Dieser soll den schnellen Zugriff auf Daten und andere Dateien des Projekts ermöglichen.

Standardmäßig ist dies der Ordner, in dem das Projekt gespeichert wurde, kann aber über die Option *Projekt ► Eigenschaften... ► Allgemein ► Projekt-Stammverzeichnis* oder durch einen Rechtsklick auf den Eintrag *Projektverzeichnis* und Auswahl von *Projektverzeichnis wählen...* geändert werden. Das Anpassen dieses Ordners ist besonders nützlich, wenn QGIS-Projekte nicht im Stammordner eines organisatorischen „Projekts“ zusammen mit Datensätzen gespeichert sind.



### 8.2.4 Laufwerke und Dateisystem

Standardmäßig werden im *Browser* Bedienfeld auch verschiedene Ordner angezeigt. Welche dies sind, ist vom verwendeten Betriebssystem abhängig.

Diese Ordner sind:


- Unter Windows wird als *Home* das Benutzerverzeichnis des aktuell angemeldeten Benutzers gezeigt
- auf Unix-basierten Systemen wird auf das Root-Verzeichnis */* verwiesen
- Alle verbundenen Laufwerke, sowohl lokale als auch Netzlaufwerke, werden angezeigt. Je nach Betriebssystem werden sie direkt aufgelistet (z.B. *C : \*, *D : \*) oder über den Eintrag */Volumes*.

Über das Kontextmenü haben Sie für jeden Ordner (auch Unterordner) und jedes Laufwerk folgende Optionen:

- *Aktualisieren des Inhalts*
- Mit *Neu ►* können Sie ein *Verzeichnis*, ein *GeoPackage* oder eine *ESRI Shape-Datei* neu erstellen
- Das Verzeichnis (*Im Browser verstecken*); dies kann mittels eines Rechtsklicks auf ein anderes Verzeichnis über *Verborgene Elemente ►* wieder sichtbar gemacht werden
- Mit *Farbe setzen...*: können sie Farbe, in der das Verzeichnis angezeigt wird, anpassen. Dies kann die schnelle Navigation in komplexen Ordnerstrukturen erleichtert
- enable *Scanning*:
  -  *Änderungen verfolgen*: ermöglicht es manuell festzulegen, ob ein bestimmtes Verzeichnis auf Änderungen überwacht und automatisch aktualisiert werden soll oder nicht. Diese Einstellung gilt für das ausgewählte Verzeichnis und alle Unterverzeichnisse. Standardmäßig werden lokale Verzeichnisse automatisch überwacht, Remote- oder Netzlaufwerke nicht. Damit haben sie z.B die Möglichkeit, die Überwachung von Netzlaufwerken zu aktivieren, wenn Sie wissen, dass es keine Probleme gibt, oder die Überwachung großer Verzeichnisse zu deaktivieren, um lange Ladezeiten zu vermeiden.
  -  *Verzeichnis schnell durchsuchen*
- *Verzeichnis öffnen...* öffnet das Verzeichnis im Dateibrowser
- open the directory in a terminal window (*Open in Terminal...*)
- mit *Eigenschaften...* können sie Informationen über das Verzeichnis abrufen, die in QGIS zur Verfügung stehen; mit *Verzeichniseigenschaften...* rufen sie die Informationsseite des Betriebssystems auf

## 8.2.5 Datenbankeinträge

Abhängig von Ihrem Betriebssystem und den installierten Treibern haben Sie ggf. Zugriff auf verschiedene Datenbanktypen, die Sie in QGIS verwenden können. In der nachfolgenden Tabelle sind die verschiedenen Möglichkeiten im Kontextmenü für jede Ebene der Datenbankstruktur aufgeführt.

Ebene	Kontextmenü	Datenbank-Typ						
		 GeoPackage ( <sup>1</sup> )	 Spatialite	 PostGIS	 HANA	 SAP	 MS	 Oracle
Oberste Ebene	Eine <i>Neue Verbindung..</i> zu einer existierenden Datenbank erstellen	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>
	einen neue <i>Datenbank erzeugen...</i>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
	in einer Datei <i>Verbindungen speichern...</i>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
	aus einer Datei <i>Verbindungen laden...</i>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Verbindung/	Eine <i>Verbindung Aktualisieren</i>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
	Die <i>Einstellungen einer Verbindung bearbeiten...</i>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
	Die <i>Verbindung entfernen</i>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
	<Name_Datei <i>löschen</i>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
	<i>Datenbank komprimieren (VACUUM)</i>	<input checked="" type="checkbox"/>						
	Ein <i>Neues Schema...</i> erstellen			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
	Eine <i>Neue Tabelle...</i> erstellen	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			




























Fortsetzung auf der nächsten Seite

Tab. 8.1 – Fortsetzung der vorherigen Seite

Schema	Eine <i>SQL-Abfrage</i> ausführen...					
	Ein Schema aktualisieren					
	Schema-Aufgaben ► Schema umbenennen					
	Schema-Aufgaben ► Schema löschen...					
Tabelle/Layer	Eine <i>Neue Tabelle...</i> erstellen					
	Eine <i>SQL-Abfrage</i> ausführen...					
	Tabellen-Aufgaben ► Tabelle umbenennen.					
	Tabellen-Aufgaben ► Tabelle trunkieren...					
	Eine <i>SQL-Abfrage</i> ausführen...					
	Layer exportieren ► in Datei...					
	Verwalten ► Layer <Name_Layer> umbenennen.					
	Verwalten ► Layer <Name_Layer> löschen...					
	Verwalten ► ausgewählte Layer löschen...					





































































Fortsetzung auf der nächsten Seite

Tab. 8.1 – Fortsetzung der vorherigen Seite

	Verwalten					
	► Layer zu Projekt hinzufügen					
	Verwalten					
	► ausgewählte Layer zu Projekt hinzufügen					
	Layer Eigenschaften Dialog öffnen					
	Datei Eigenschaften Dialog öffnen					
Felder	Neues Feld hinzufügen...					
Feld	Set Alias...					
	Set Comment...					
	Feld löschen...					

<sup>1</sup> The different entries may also be available for some GDAL supported [vector file formats](#) such as ESRI File Geodatabase, FlatGeobuf, GeoParquet, NetCDF, ... when compatible.

## 8.2.6 Kacheln und Internetdienste

Ebene	Kontextmenü	Art des Dienstes							
		 WMS/WFS	 Vector Tiles	 XYZ Tiles	 WCS	 OGC API Features	WFS / ArcGIS-REST-Server	 Scene	
Oberste Ebene	Eine <i>Neue Verbindung...</i> erstellen								
	Eine <i>Neue allgemeine Verbindung...</i> erstellen								
	Eine <i>Neue ArcGIS-Vektorkacheldienst-Verbindung...</i> erstellen								
	in einer Datei <i>Verbindungen speichern...</i>								
	aus einer Datei <i>Verbindungen laden...</i>								
Verbind.	<i>Verbindung Aktualisieren</i>								
	<i>Verbindung bearbeiten...</i>								
	<i>Verbindung löschen</i>								
	Im Browser <i>Dienst-Informationen anzeigen</i>								
Tabelle	<i>Layer exportieren ► in Datei...</i>								
	<i>Layer zum Projekt hinzufügen</i>								
	<i>Layerseigenschaften... Dialog öffnen</i>								
	Im Browser <i>Dienst-Informationen anzeigen</i>								

## 8.3 Datenquellen und Dateien

- QGIS-Projektdateien - das Kontext-Menü für QGIS-Projektdateien bietet die folgenden Möglichkeiten:
  - (*Projekt öffnen*)
  - (*Symbole extrahieren...*): öffnet die Stillverwaltung, mit der Sie dann die Symbole des betreffenden Projekts in eine XML-Datei exportieren, zum Standardstil hinzufügen oder als PNG- oder SVG-Datei exportieren können
  - Die (*Dateieigenschaften...*) öffnen

Sie können die Projektdatei erweitern, um die enthaltenen Layer zu sehen. Das Kontextmenü eines Layers bietet dann dieselben Aktionen wie an anderen Stellen des Browsers.

- QGIS Layer Definition files (QLR) - das Kontext-Menü für QLR-Dateien bietet die folgenden Möglichkeiten:
  - *Layer exportieren ► in Datei...*

- *Layer zum Projekt hinzufügen*
  - Die (*Layerereigenschaften...*) anzeigen
- Processing models (.model3) - das Kontext-Menü für model3-Dateien bietet die folgenden Möglichkeiten:
  - *Model ausführen...*
  - *Model bearbeiten...*
- QGIS print composer templates (QPT) - das Kontext-Menü für QPT-Dateien bietet die folgenden Möglichkeiten:
  - (*Neues Layout aus Vorlage*)
- Python Skripte (.py) - das Kontext-Menü für Python Skripte bietet die folgenden Möglichkeiten:
  - (*Skript ausführen...*)
  - (*In externem Editor öffnen*)
- Unterstützte Rasterformate - das Kontext-Menü für Raster-Dateien bietet die folgenden Möglichkeiten:
  - (*Lösche Datei <Datei\_Name>*)
  - *Layer exportieren ► in Datei...*
  - *Layer zum Projekt hinzufügen*
  - Anzeigen von *Layerereigenschaften...*, bzw. *Dateieigenschaften...*)

Einige Dateiformate können auch extern geöffnet werden: *Öffne <Format> extern ...*

- Unterstützte Vektorformate - das Kontext-Menü für Vektor-Dateien bietet die folgenden Möglichkeiten:
  - (*Lösche Datei <Datei\_Name>*)
  - *Layer exportieren ► in Datei...*
  - *Layer zum Projekt hinzufügen*
  - Anzeigen von *Layerereigenschaften...*, bzw. *Dateieigenschaften...*)






Einige Dateiformate können auch extern geöffnet werden: *Öffne <Format> extern ...*

---



## QGIS anpassen

---

Sie können QGIS auf vielfache Art und Weise an Ihre Bedürfnisse anpassen. Im Menü *Einstellungen* finden Sie verschiedene Werkzeuge hierfür:

-  *Stilverwaltung...*: erstellen und verwalten von *Symbolen*, *Farbverläufen*, etc.
-  *Benutzerprojektionen...*: definieren Sie Ihre eigenen :ref:`Koordinatenbezugssystem<sec\_custom\_projections>`.
-  *Tastenkürzel...*: legen Sie ihre eigenen <shortcuts> fest.
-  *Oberflächenanpassung...*: passen Sie die *Oberfläche* an, indem Sie Dialoge oder Werkzeuge verstecken, die Sie für Ihre Arbeit nicht benötigen.
-  *Optionen...*: hier können Sie verschiedene :ref:`<gui\_options>` festlegen, die in unterschiedliche Bereiche gegliedert sind. Diese werden in den Einstellungen des aktuellen *Benutzerprofils* gespeichert und sind immer dann wirksam, wenn Sie mit diesem Profil arbeiten.

### 9.1 Optionen

 Einige grundlegende Einstellungen für QGIS können Sie mit dem Dialog *Optionen* vornehmen. Wählen Sie im Menü *Einstellungen* ►  *Optionen* aus. Hier können Sie die Einstellungen an Ihre Bedürfnisse anpassen. Einige Änderungen können einen Neustart von QGIS erfordern um wirksam zu werden.

Im Nachfolgenden werden die einzelnen Reiter beschrieben, in denen Sie die Optionen ihren Bedürfnissen anpassen können.

---

**Bemerkung:** Plugins können durch ihre eigenen Einstellungsmöglichkeiten die unten beschriebenen Optionen erweitern

Bitte beachten Sie, dass im Folgenden nur die Einstellungen der Kernanwendung beschrieben werden, wenn aber *Erweiterungen* installiert werden, ist es möglich, dass diese ihre eigenen Optionen in die Standarddialoge einfügen. Damit wird verhindert, dass jede Erweiterung einen eigenen Konfigurationsdialog mit eigenen Menüeinträgen mitbringt.

---

## 9.1.1 Allgemein

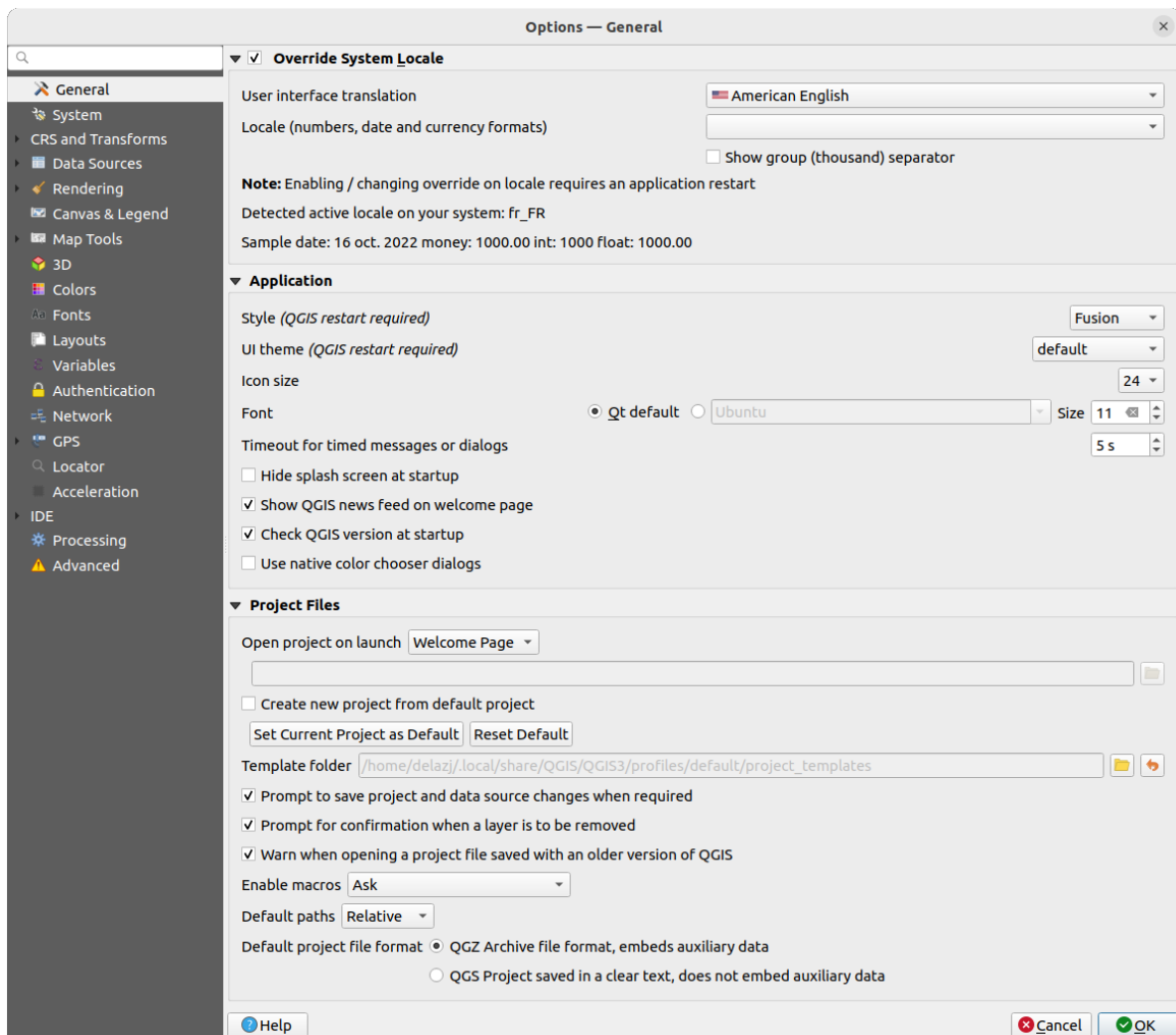


Abb. 9.1: Allgemein

### System-Locale übersteuern

Standardmäßig verwendet QGIS die Konfiguration Ihres Betriebssystems für die Sprach- und Zahleinstellungen. Wenn Sie diese Gruppe aktivieren, können Sie dieses Verhalten anpassen:




- Wählen Sie in *Benutzeroberflächenübersetzung* die Sprache aus, die für die Benutzeroberfläche verwendet werden soll
- Wählen Sie in *Locale (Zahlen-, Daten- Währungsformate)* das System, in dem Datum und numerische Werte eingegeben und dargestellt werden sollen
- ☒ *Gruppen-/Tausendertrennung anzeigen*

Am unteren Rand des Bereichs wird eine Zusammenfassung der ausgewählten Einstellungen und deren Interpretation angezeigt.




### Anwendung

- Wählen Sie den *Stil (QGIS-Neustart erforderlich)*, d.h. das Aussehen und die Anordnung der Elemente in Dialogen. Die Auswahlmöglichkeiten hängen von Ihrem Betriebssystem ab.



- Legen Sie das *Oberflächenthema* (QGIS-Neustart erforderlich) fest . Mögliche Werte sind ‚default‘, ‚Night Mapping‘, und ‚Blend of Gray‘.
- Legen Sie die *Icon-Größe* fest .
- Legen Sie die *Schriftart* und ihre *Größe* fest. Die Auswahl kann  *Qt Vorgabe* oder eine benutzerdefinierte Schrift sein.
- Ändern Sie die *Dauer von zeitweiligen Meldungen und Dialogen*
- ☐ *Splashscreen beim Start nicht anzeigen*
- ☒ *QGIS-Neuigkeiten auf Willkommenseite anzeigen*: zeigt einen QGIS-Newsfeed auf der Startseite an, der Sie über Projektneuigkeiten informiert (Datum und Zusammenfassung von Benutzer-/Entwicklertreffen, Community-Umfragen, Ankündigungen von Releases, verschiedene Tipps...)
- mit ☒ *QGIS-Version beim Start überprüfen* werden Sie informiert, wenn eine neue QGIS-Version veröffentlicht wurde
- ☐ *Native Farbauswahldialoge verwenden* (siehe [Farbauswahl](#))

## Projektdateien

- *Projekt beim Start öffnen* bietet die folgenden Möglichkeiten:
  - ‚Willkommenseite‘ (Standardeinstellung): kann den „Newsfeed“ anzeigen, die Projektvorlagen und die zuletzt verwendeten Projekte des aktuellen *Benutzerprofils* (mit Vorschau); standardmäßig wird kein Projekt geöffnet
  - ‚Neu‘: öffnet ein neues Projekt, das auf der Standardvorlage basiert
  - ‚Zuletzt verwendetes‘: öffnet das zuletzt verwendete Projekt
  - ‚Bestimmtes‘: öffnet ein festgelegtes Projekt; benutzen sie die ... Schaltfläche, um das Projekt festzulegen
- Unter ☒ *Neues Projekt aus Vorgabeprojekt erstellen* können Sie das *Aktuelle Projekt als Vorgabe speichern* oder die *Vorgabe zurücksetzen*. In der Zeile darunter können Sie ein Verzeichnis festlegen, in dem Sie ihre benutzerdefinierten Projektvorlagen speichern. Diese werden dann dem Menü *Projekt ► Neu aus Vorlage* hinzugefügt.
- Wählen Sie ☒ *Bei Bedarf nachfragen, ob geänderte Projekte und Datenquellen gespeichert werden sollen*, um zu verhindern, dass Ihre Änderungen verloren gehen.
- ☒ *Beim Löschen von Layern Bestätigung anfordern*
- ☒ *Warnung ausgeben, wenn QGIS-Projekt einer früheren Version geöffnet wird*: Sie können Projekte, die mit einer älteren QGIS-Version gespeichert wurden, jederzeit öffnen; das Öffnen einer neueren Projektdatei mit einer älteren Version kann aber scheitern, weil neuere Entwicklungen in dieser Version noch nicht verfügbar sind.
- *Makros aktivieren* : diese Option stellt eine Sicherheitseinstellung dar und legt fest, wie das Ausführen von externen Makros gehandhabt wird; Sie können zwischen ‚Nie‘, ‚Fragen‘, ‚Nur in dieser Sitzung‘ und ‚Immer (nicht empfohlen)‘ wählen.
- *Vorgabepfade*: legt fest, ob Pfade zu Dateien und Layern, die in neuen Projekten verwendet werden, als ‚Absolut‘ oder ‚Relativ‘ zur Projektdatei gespeichert werden. Diese Einstellung kann auf Projektebene überschrieben werden.
- *Vorgabeprojektdateiformat*
  -  *QGZ-Archivdateiformat, enthält Zusatzdaten* (siehe [Hilfsspeicher](#))
  -  *QGS-Projekt in Klartext speichern, unterstützt keine Zusatzdaten*: die Zusatzdaten werden in einer separaten .qgd-Datei zusammen mit der Projektdatei gespeichert

## 9.1.2 System

### SVG-Pfade

Es können *Suchpfad(e) für SVG-Symbole (Scalable Vector Graphic)* hinzugefügt oder entfernt werden. Alle SVG-Dateien, die sich in diesen Verzeichnissen befinden, stehen dann zur Verfügung, um die Objekte ihrer Karte zu symbolisieren, zu beschriften oder um Dekorationen hinzuzufügen (z.B. in der Layer-Symbolisierung oder der Stilverwaltung).

Lesen Sie auch das Kapitel *Remote or embedded file selector* für verschiedene Möglichkeiten, auf SVG-Dateien in einem QGIS-Pfad zu verweisen.

### Erweiterungspfade

Fügen Sie *Nach zusätzlichen C++-Erweiterungsbibliotheken zu durchsuchende Pfad(e)* hinzu oder entfernen Sie diese.

### Dokumentationspfade

Sie können *Pfad(e) um nach QGIS-Hilfe zu suchen* einfügen oder entfernen. Die eingetragenen Pfade werden von der QGIS-Hilfe benutzt. Als Voreinstellung finden Sie einen Link auf das offizielle QGIS-Onlinehandbuch, das der benutzen QGIS-Version entspricht. Sie können jedoch andere Links einfügen und sie in eine Reihenfolge von oben nach unten bringen: jedes mal, wenn Sie auf eine Schaltfläche *Hilfe* in einem Dialog klicken, wird der oberste Link geprüft; falls keine passende Seite gefunden wird, wird der nächste geprüft usw.

---

**Bemerkung:** Die Dokumentation wird nur für die QGIS Long Term Releases (LTR) versioniert und übersetzt. Das bedeutet, dass Sie, wenn Sie in einer regulären Version (z.B. QGIS 3.0) auf die Schaltfläche Hilfe klicken, als Voreinstellung das nächste LTR-Benutzerhandbuch angezeigt bekommen (d.h. 3.4 LTR), das aber Beschreibungen von Programmeigenschaften enthalten kann, die erst in neueren Versionen (3.2 und 3.4) enthalten sind. Falls keine LTR-Dokumentation verfügbar ist, wird die *testing*-Dokumentation angezeigt, die Programmeigenschaften von neueren und Entwicklerversionen enthält.

---

### Einstellungen

Sie können die *Benutzeroberflächeneinstellungen zurücksetzen (Neustart erforderlich)* falls Sie eigene *Anpassungen* vorgenommen haben, diese aber wieder verwerfen wollen.

### Umgebung

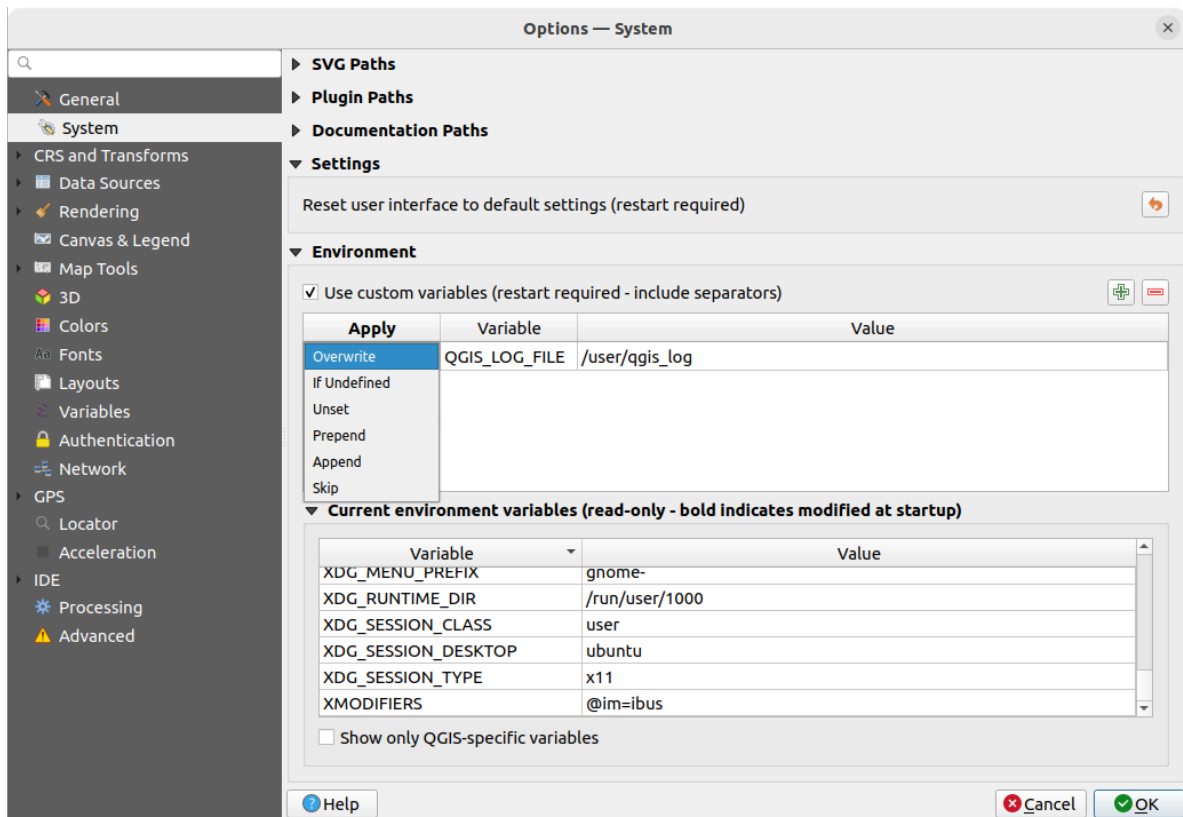




Abb. 9.2: Systemumgebungsvariablen

Die Systemumgebungsvariablen können in der Gruppe **Umgebung** eingesehen und viele davon auch konfiguriert werden. Dies ist nützlich für Plattformen wie Mac, wo eine GUI-Anwendung nicht unbedingt die Shell-Umgebung des Benutzers übernimmt. Es ist auch nützlich, um Umgebungsvariablen für die externen Werkzeuge, die von der Werkzeugkiste gesteuert werden (z. B. SAGA, GRASS), anzuzeigen und zu setzen und um die Debugging-Ausgabe für bestimmte Abschnitte des Quellcodes zu aktivieren.

Wenn Sie das Kontrollkästchen *Benutzerdefinierte Umgebungsvariablen verwenden (Neustart erforderlich - Trennzeichen einschließen)* aktiviert haben, können Sie eine  neue Variable hinzufügen oder eine  Variable entfernen. Für jedes neue Element können Sie einen *Variable*-Namen, einen *Wert* und die zu verwendende *Anwenden*-Methode konfigurieren, wobei die folgenden Methoden zur Verfügung stehen:

- *Überschreiben*: ersetzt einen bereits existierenden Wert der Variablen
- *Wenn definiert*: diesen Wert für die Variable verwenden, wenn er nicht bereits auf einer höheren Ebene definiert wurde (z. B. auf Betriebssystem- oder Anwendungsebene)
- *Nicht gesetzt*: die Variable aus der Umgebung entfernen (der Parameter *Wert* wird nicht verwendet)
- *Voranstellen*: den Wert dem bereits bestehenden Wert der Variablen voranstellen
- *Anhängen*: den Wert dem bereits bestehenden Wert der Variablen anfügen
- *Überspringen*: der Eintrag wird in der Liste für zukünftige Referenzzwecke aufbewahrt, aber nicht benutzt

Bereits definierte Umgebungsvariablen werden in *Aktuelle Umgebungsvariablen* angezeigt, wobei es möglich ist, sie durch Aktivierung der Checkbox *Nur QGIS-spezifische Variablen anzeigen* zu filtern.


## 9.1.3 User Profiles Settings

**Bemerkung:** For more information on how to manage user profiles, please read the dedicated section at [Working with User Profiles](#).

## 9.1.4 KBS und Transformation

**Bemerkung:** Für weitere Informationen darüber, wie QGIS mit der Projektion von Layer umgeht, lesen Sie bitte den Abschnitt [Arbeiten mit Projektionen](#).

### KBS-Behandlung

Auf dem  *KBS-Behandlung*-Reiter können Sie festlegen, welches Koordinatenbezugssystem für ein neues Projekt oder einen neuen Layer verwendet werden soll.

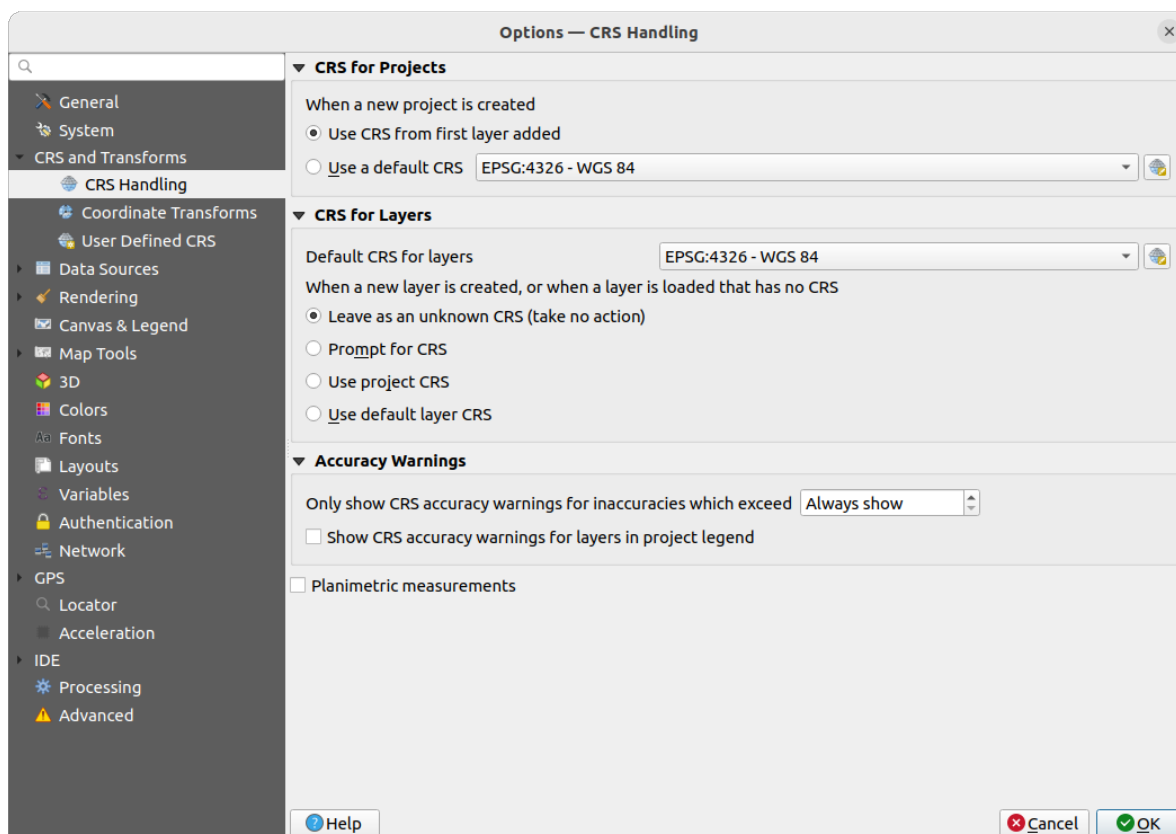




Abb. 9.3: KBS Einstellungen

### KBS für Projekte

Um einem neu erzeugten Projekt ein KBS zuzuweisen, gibt es zwei Möglichkeiten:

-  *KBS des ersten hinzugefügten Layer verwenden:* das KBS des Projekts wird auf das KBS des ersten geladenen Layers gesetzt





-  *Voreingestelltes KBS verwenden*: Ein vorausgewähltes KBS wird standardmäßig auf jedes neue Projekt angewendet und bleibt beim Hinzufügen von Layern zum Projekt unverändert.

Die Auswahl wird für die Verwendung in späteren QGIS-Sitzungen gespeichert. Das Koordinatenbezugssystem des Projekts kann weiterhin über das Menü *Projekt ► Eigenschaften... ► KBS* angepasst werden.

### KBS für Layer



*Vorgabe-KBS für Layer*: hier können Sie ein Standard-KBS auswählen, das beim Erstellen eines Layers verwendet werden soll (wenn unten ausgewählt)

Sie können die Aktion festlegen, die ausgeführt werden soll, wenn ein neuer Layer erstellt wird oder wenn ein Layer ohne KBS geladen wird:

-  *Bei unbekanntem KBS belassen (nichts tun)*
-  *KBS abfragen*
-  *Projekt KBS verwenden*
-  *Projekt KBS verwenden*

### Genauigkeitswarnungen


*Nur KBS Genauigkeitswarnungen anzeigen, wenn Ungenauigkeiten größer sind als*: hier kann eingestellt werden, wann eine Warnung angezeigt wird, wenn Sie einen Datensatz erstellen oder ändern und ein KBS auf der Grundlage eines Bezugssystems mit geringerer Genauigkeit auswählen. Die Voreinstellung ist *Immer anzeigen*, es kann aber auch ein Wert (in Metern) eingestellt werden. Benötigt eine QGIS Version mit mindestens **PROJ 8.0**.

 *KBS-Genauigkeitswarnung für Layer nur in Projektlegende anzeigen*: Wenn diese Option aktiviert ist, wird jeder Layer mit einem KBS mit Genauigkeitsproblemen (d. h. ein dynamisches KBS, für das keine Koordinaten-Epoche verfügbar ist, oder ein KBS, das auf einem Bezugssystem mit einer inhärenten Ungenauigkeit basiert, die den vom Benutzer festgelegten Grenzwert überschreitet) im *Layer* Bedienfeld mit dem Warnsymbol  gekennzeichnet, das anzeigt, dass es sich um einen Layer mit geringer Genauigkeit handelt.

Dies wurde für den Einsatz in den Bereichen Ingenieurwesen, BIM, Asset Management und anderen Bereichen entwickelt, in denen Ungenauigkeiten dieser Art potenziell sehr gefährlich oder teuer sind!

 *Planimetric measurements*: sets the default for the *planimetric measurements* property for newly created projects.

## Koordinatentransformationen

Auf dem  *Koordinatentransformationen* Reiter können Sie Koordinatentransformationen und Operationen festlegen, die beim Laden eines Layers in ein Projekt oder bei der Reprojektion eines Layers angewendet werden sollen.

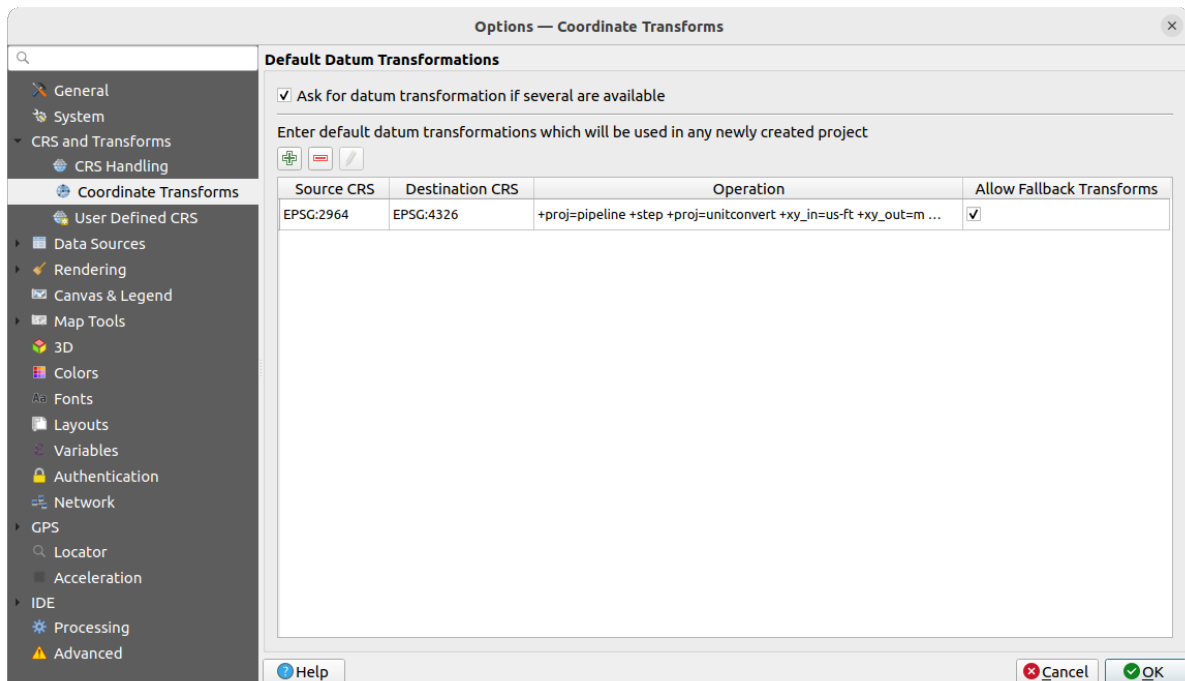



Abb. 9.4: Transformationseinstellungen


### Datumstransformationsvorgabe

Hier können Sie einstellen, wie verfahren werden soll, wenn Layer in ein anderes KBS projiziert werden müssen:

- automatisch die QGIS-Standardeinstellungen für Transformationen anwenden
- oder Sie können individuelle Optionen einstellen:
  -  *Datumstransformation erfragen, wenn mehrere verfügbar sind*
  - aus einer vordefinierte Liste von Datumstransformationen eine bestimmte auswählen, die standardmäßig angewendet werden soll; für Details dazu siehe [Datum Transformations](#)

Sie können Vorgabedatumstransformationen  hinzufügen,  entfernen oder  bearbeiten, die für alle neu erzeugten Projekte verwendet werden sollen.

### Benutzerdefinierte KBS

„. Auf dem Reiter  *Benutzerdefinierte KBS* können Sie Ihre eigenen KBS festlegen. Diese müssen dem WKT- oder Proj-Format entsprechen.

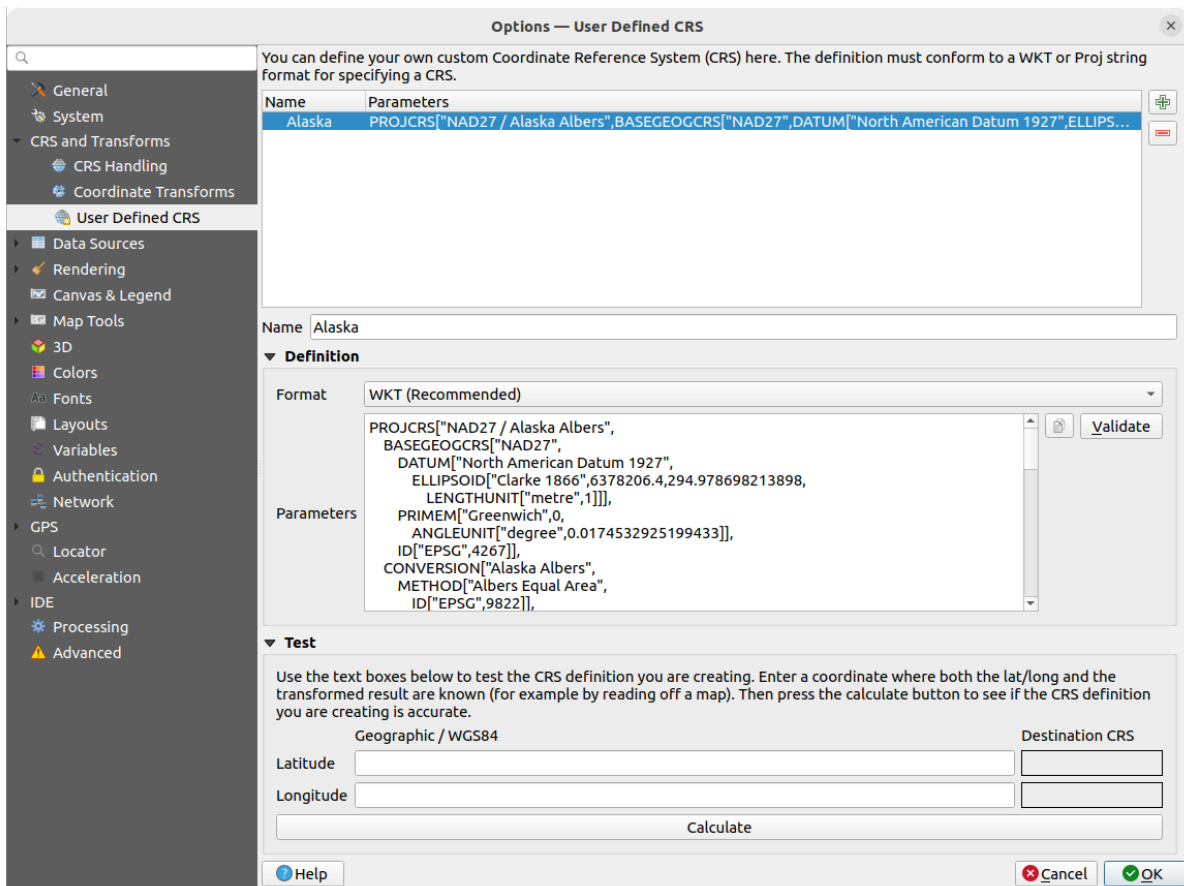




Abb. 9.5: Benutzerdefinierte KBS


Tragen Sie unter der Liste im Textfeld *Name* den gewünschten Namen ein und verwenden Sie die  Neues KBS hinzufügen Schaltfläche um ein neues KBS anzulegen. Wenn Sie ein vorhandenes KBS löschen möchten, können Sie die  KBS entfernen Schaltfläche verwenden.

## Definition

### • *Format*

- WKT (empfohlen)
- Proj-Zeichenkette (Altlast - nicht empfohlen)

### • *Parameter*

-  Sie können die Parameter eines vorhandenem KBS kopieren
- mit *Validieren* können Sie testen, ob Ihr Ausdruck korrekt ist

## Test

Hier können Sie Ihre erstellte KBS-Definition anhand einer Umrechnung von Breiten- und Längengrad testen. Geben Sie eine bekannte Koordinate, um zu prüfen, ob die Umrechnung das korrekte Ergebnis liefert.

## 9.1.5 Datenquellen

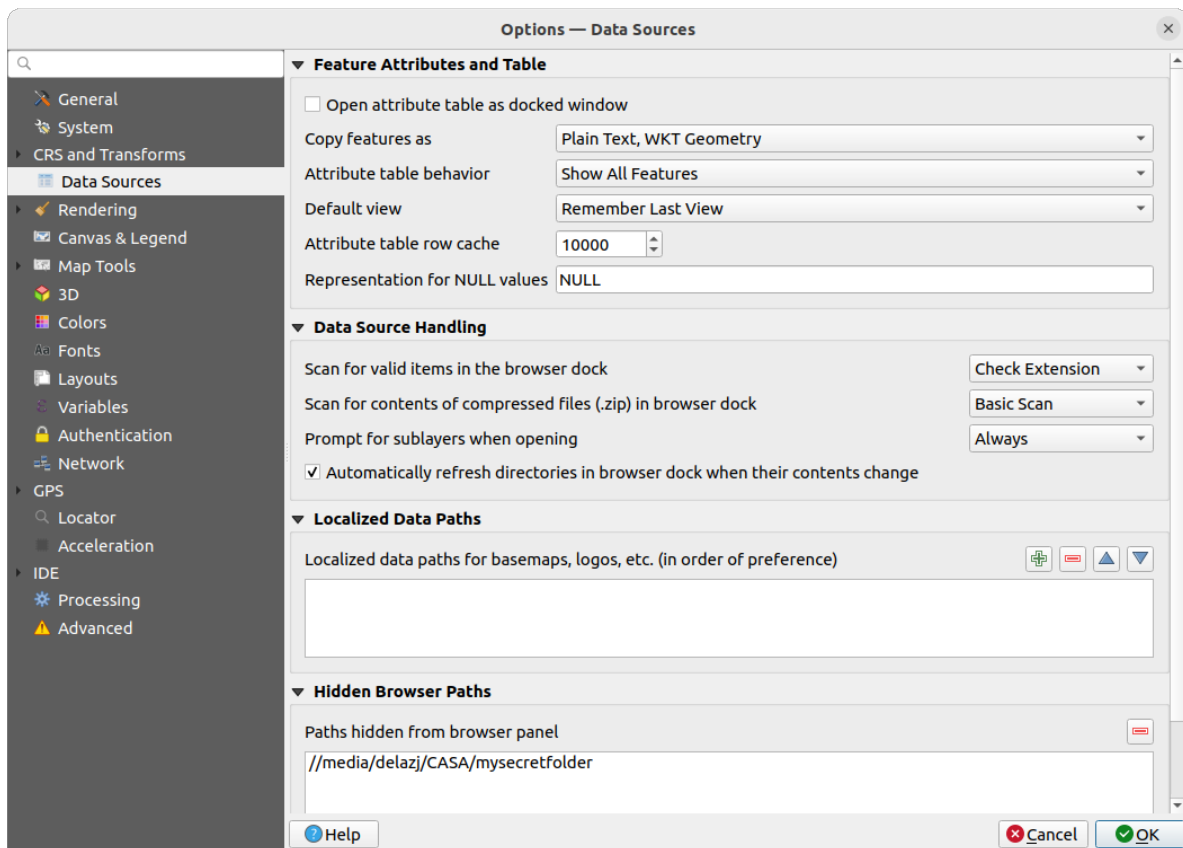


Abb. 9.6: Datenquellen

### Objektattribute und -tabelle

- *Attributtabelle im gedockten Fenster öffnen*
- *Objekte kopieren als* ‚Einfacher Text, WKT-Geometrie‘, ‚Einfacher Text, keine Geometrie‘, oder ‚GeoJSON‘ wenn Objekte in andere Anwendungen kopiert werden
- *Attributtabellenverhalten* : setzt beim Öffnen einen Filter auf die Attributtabelle; es gibt die folgenden Möglichkeiten: ‚Alle Objekte anzeigen‘, ‚Alle gewählten Objekte anzeigen‘ and ‚Sichtbare Objekte der Karte anzeigen‘.
- *Voreingestellte Ansicht*: definiert den Ansichtsmodus der Attributtabelle beim Öffnen; mögliche Einstellungen sind ‚Letzte Ansicht merken‘, ‚Tabellenansicht‘ oder ‚Formularansicht‘
- *Attributtabellenzeilencache* : hier wird die Anzahl der Zeilen der zuletzt geladenen Tabelle festgelegt, die im Cache gespeichert werden; der Cache wird gelöscht, wenn die Attributtabelle geschlossen wird
- *Repräsentation für NULL-Werte*. Hier können Sie einen Wert festlegen, der in Datenfeldern angezeigt wird, die einen NULL-Wert enthalten




### **Tipp: Schnelleres Öffnen von Attributtabellen mit sehr vielen Zeilen**

Wenn Sie mit Layern mit einer großen Anzahl von Datensätzen arbeiten, kann das Öffnen der Attributtabelle lange dauern, da für die Anzeige alle Zeilen des Layers abfragt werden. Wenn Sie das **Attributtabellenverhalten** auf **Sichtbare Objekte der Karte anzeigen** setzen, wird QGIS beim Öffnen der Tabelle nur die Features in der aktuellen Kartenansicht abfragen, was ein schnelleres Laden der Daten ermöglicht.



Beachten Sie, dass bei dieser Einstellung die Attributtabelle immer nur Objekte enthält, die in der Kartenansicht sichtbar waren, als die Tabelle geöffnet wurde. Das bedeutet, dass auch nach dem Verschieben des Kartenausschnitts mit **Alle Objekte anzeigen** in der Tabelle keine neuen Objekte anzeigen werden. Sie können jedoch die angezeigten Objekte aktualisieren, indem Sie die Option **Features auf der Karte sichtbar machen** auswählen.

### Datenquellenbehandlung

- *Nach gültigen Element im Browser suchen* . Sie können zwischen ‚Erweiterung prüfen‘ und ‚Dateiinhalte prüfen‘ wählen.
- *Inhalt komprimierter Dateien (.zip) im Browser durchsuchen*  legt fest, wie detailliert die Suche nach Informationen (für das Browser Bedienfeld) innerhalb solcher Dateien durchgeführt wird; die möglichen Einstellungen sind ‚Nein‘, ‚Grundsuche‘ and ‚Vollsuche‘.
- *Beim Öffnen nach Unterlayern fragen*. Einige Raster unterstützen Unterlayer, diese werden in GDAL auch „subdatasets“ genannt; ein Beispiel sind netCDF-Dateien. Mit dieser Option können Sie festlegen, wie mit Unterlayern verfahren werden soll, wenn eine Datei mit Unterlayern geöffnet wird. Sie haben die folgenden Auswahlmöglichkeiten:
  - ‚Immer‘: Immer fragen (ob es Unterlayer gibt)
  - ‚Wenn nötig‘: Fragen ob Layer keine Bänder aber Unterlayer hat
  - ‚Nie‘: Nie fragen, lädt dann nichts
  - ‚Alle laden‘: Nie auffordern aber alle Unterlayer laden
-  *Verzeichnisse im Browser-Dock automatisch aktualisieren, wenn sich der Inhalt ändert*: Ermöglicht es Ihnen, die Überwachung von Verzeichnissen im *Browser* Bedienfeld zu deaktivieren (z. B. um eine mögliche Verlangsamung aufgrund von Netzwerklatenz zu vermeiden).

### Lokalisierungsdatenpfade

Es ist möglich, lokalisierte Pfade für alle Arten von dateibasierten Datenquellen zu verwenden. Dabei handelt es sich um eine Liste von Pfaden, die zur Abstraktion des Speicherorts der Datenquelle verwendet werden. Wenn z. B. `C:\my_maps` in den lokalisierten Pfaden aufgeführt ist, wird ein Layer mit der Quelle `C:\my_maps\my_country\ortho.tif` im Projekt unter `localized:my_country\ortho.tif` gespeichert.

Die Pfade sind in der Reihenfolge ihrer Präferenz aufgelistet, d.h. QGIS sucht die Datei zuerst im ersten Pfad, dann im zweiten usw.

### Ausgeblendeter Browserpfad

Hier werden alle Ordner aufgelistet, die Sie aus dem *Browser* Bedienfeld über das Kontextmenü ausgeblendet haben. Wenn Sie einen Ordner aus der Liste entfernen, wird er im *Browser* Bedienfeld wieder sichtbar.

## GDAL

**GDAL** ist eine Bibliothek für den Datenaustausch von Geodaten, die eine große Anzahl von Vektor- und Rasterformaten unterstützt. Sie bietet Treiber zum Lesen und oft auch zum Schreiben von Daten in diesen Formaten. Der Reiter *GDAL* zeigt die Treiber für Raster- und Vektorformate und ihre Beschreibung an.

### GDAL Raster- und Vektortreiber

Die Reiter *Rastertreiber* und *Vektortreiber* ermöglichen es Ihnen, festzulegen, welcher GDAL-Treiber zum Lesen und/oder Schreiben von Dateien verwendet werden soll, wenn mehr als ein GDAL-Treiber verfügbar ist.

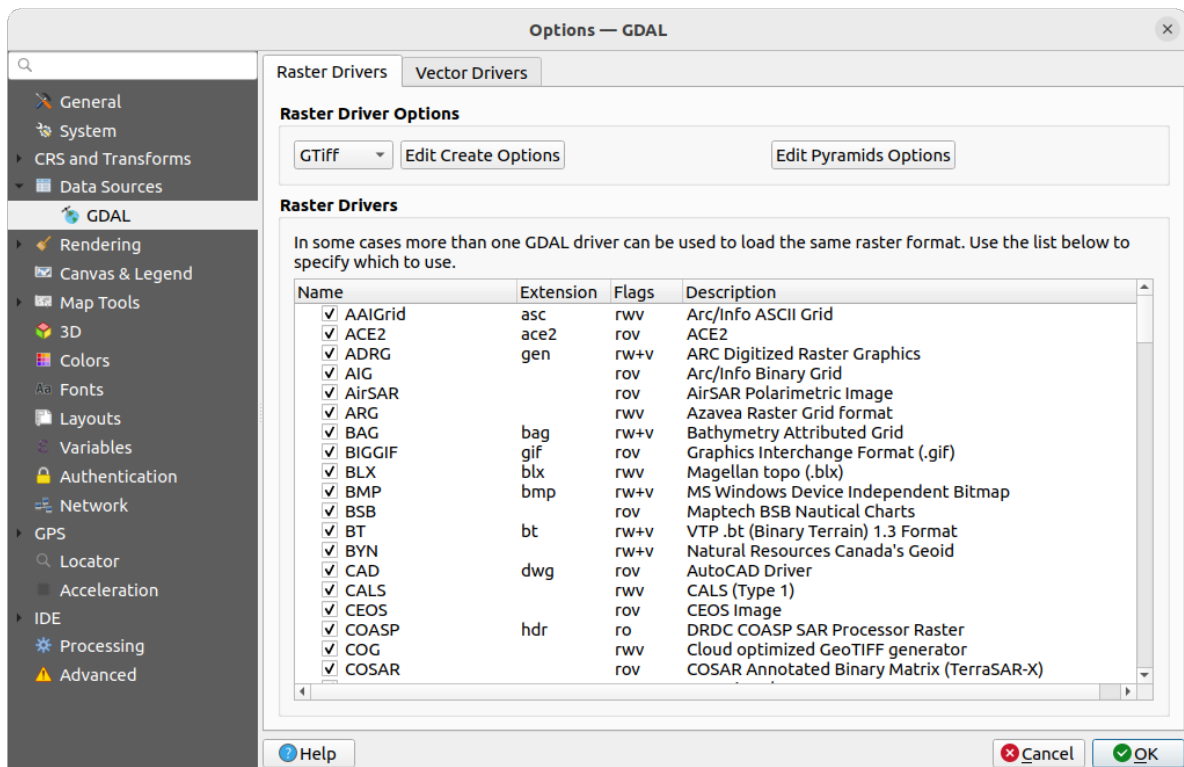


Abb. 9.7: GDAL - Rastertreiber

**Tipp:** Mit einem Doppelklick auf einen Rastertreiber für den Lese- und Schreibrechte bestehen (rw+) öffnen Sie den *Erzeugungsoptionen* Dialog zur Anpassung.

### Rastertreiberoptionen

Dieser Bereich bietet Möglichkeiten zur Anpassung des Verhaltens von Rastertreibern, auf die Lese- und Schreibzugriff besteht:

- *Erzeugungsoptionen bearbeiten:* ermöglicht das Bearbeiten oder Hinzufügen verschiedener Profile für die Dateitransformation, d.h. eine Reihe vordefinierter Kombinationen von Parametern (Art und Grad der Komprimierung, Blockgröße, Übersicht, Farbmeterik, Alpha...), die bei der Ausgabe von Rasterdateien verwendet werden. Die verfügbaren Parameter hängen vom Treiber ab.

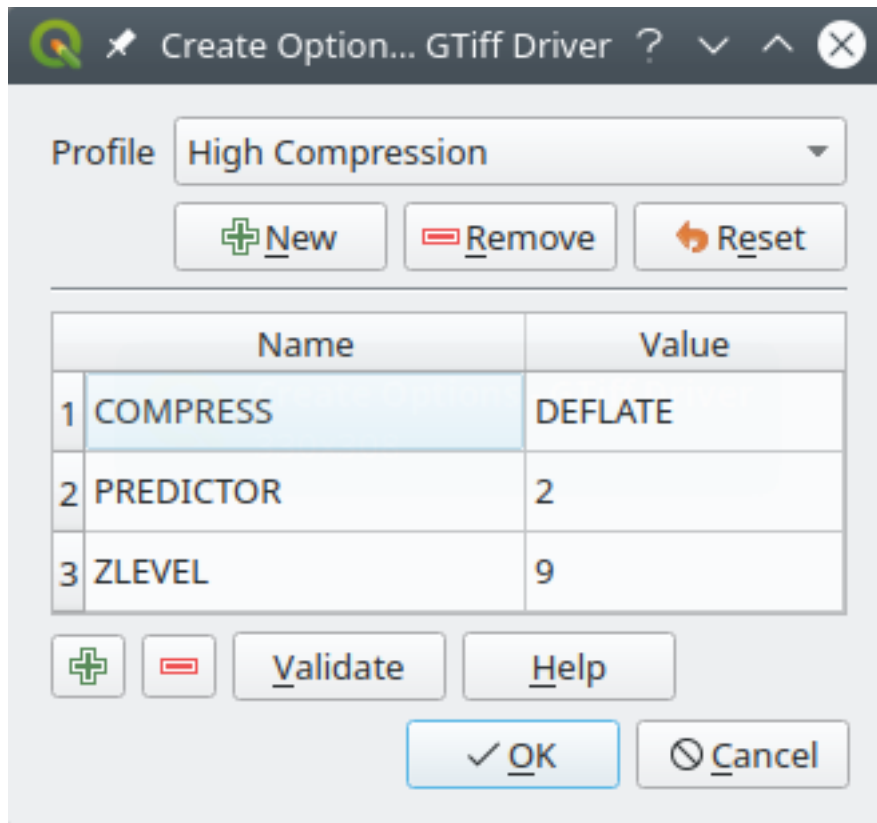




Abb. 9.8: Beispiel für die Erstellung eines Optionsprofils (für GeoTiff)

Im oberen Teil des Dialogfensters werden die aktuellen Profile aufgelistet, und Sie können neue Profile hinzufügen oder eines entfernen. Sie können das Profil auch auf seine Standardparameter zurücksetzen, wenn Sie diese geändert haben. Einige Treiber (z. B. GeoTiff) haben einige Beispielp Profile, mit denen Sie arbeiten können.

Im unteren Bereich des Dialogfensters finden Sie folgende Optionen:

- Mit der  Schaltfläche können Sie neue Zeilen einfügen, in die Sie den Parameternamen und den Wert eintragen
  - Mit der  Schaltfläche können Sie den ausgewählten Parameter löschen
  - Mit der *Prüfen* Schaltfläche können Sie prüfen, ob die für das angegebene Format eingegebenen Erstellungsoptionen gültig sind
  - Benutzen Sie die *Hilfe* Schaltfläche, um die benötigten Parameter herauszufinden oder um zur [GDAL Rastertreiber Dokumentation](#) zu gelangen.
- *Pyramiden-Optionen bearbeiten*

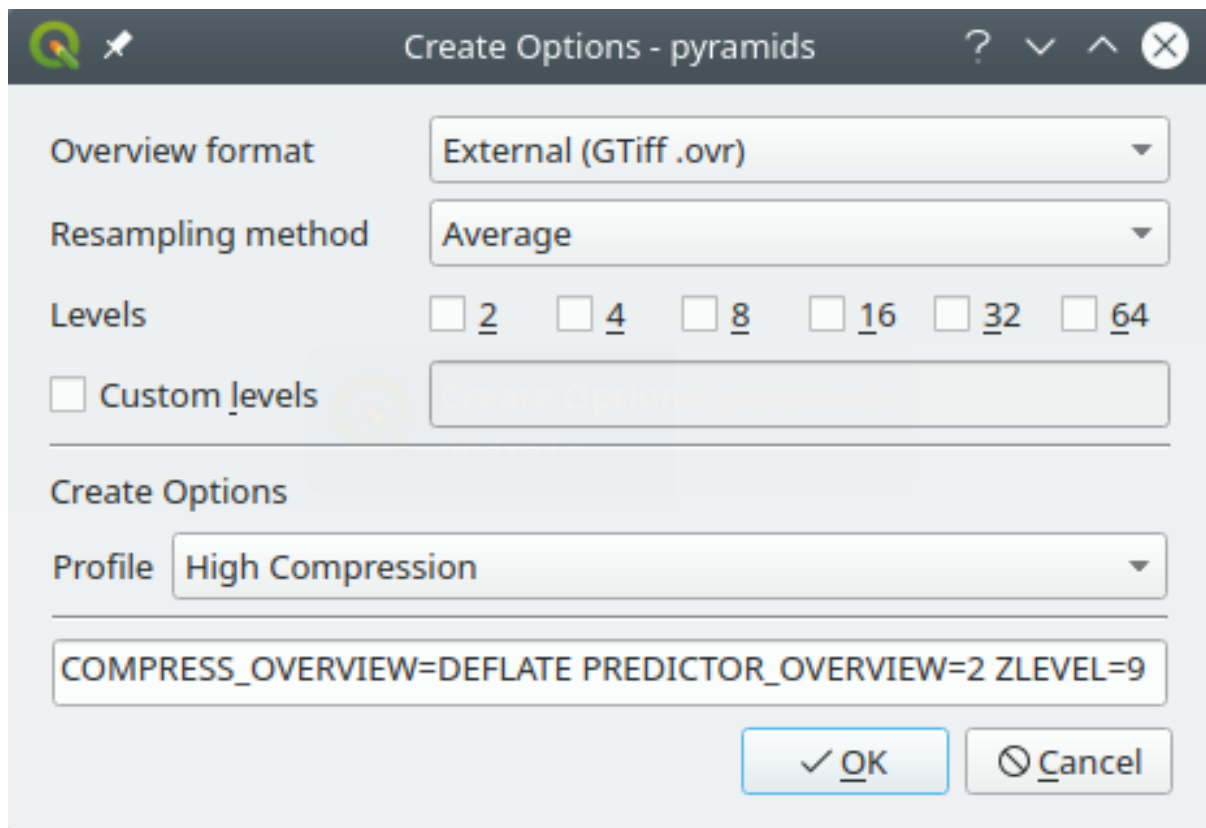


Abb. 9.9: Beispiel eines Pyramidenprofils

### 9.1.6 Darstellung

Der Reiter  *Darstellung* bietet Einstellungen zur Steuerung der Darstellung von Layern in der Kartenansicht.

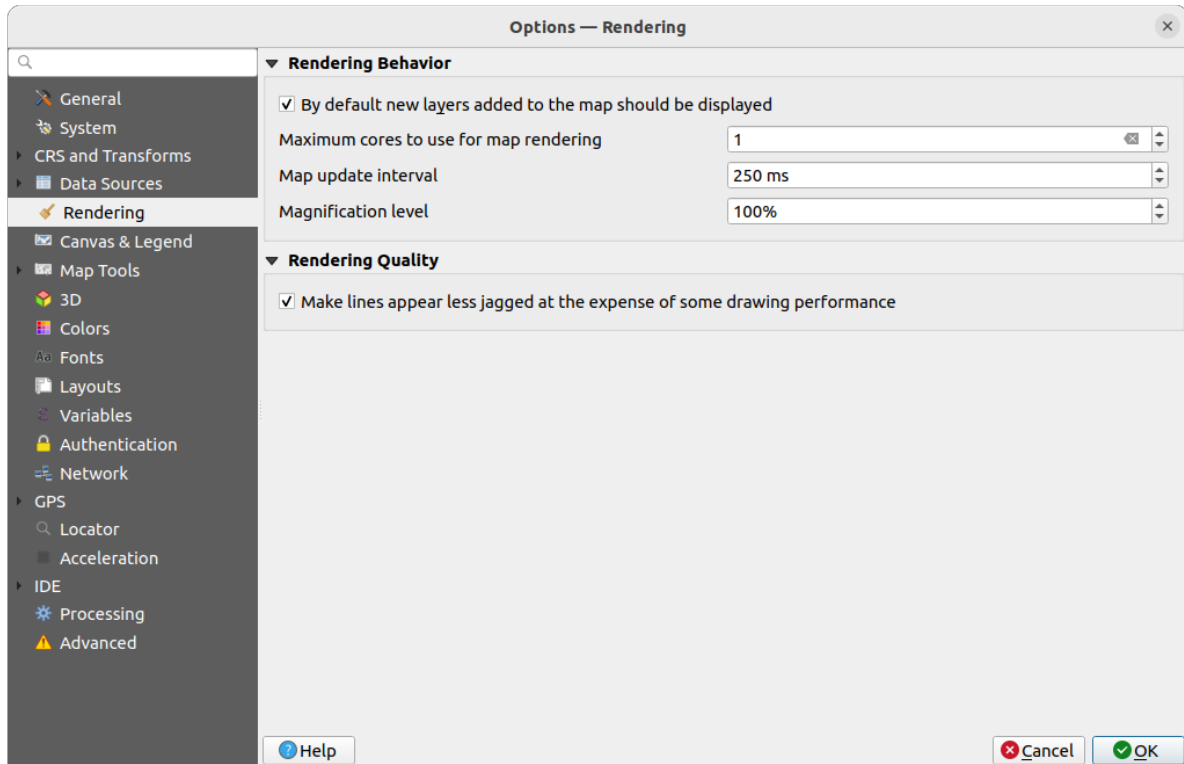


Abb. 9.10: Übersicht des Darstellungs-Reiters

### Zeichenverhalten

- ☒ *Normalerweise werden alle neuen Layer im Kartenfenster angezeigt:* Die Deaktivierung dieser Option kann beim Laden mehrerer Layer nützlich sein, um zu vermeiden, dass jeder neue Layer in der Kartenansicht gerendert wird und so das Laden verlangsamt wird
- Legen Sie die Anzahl der *Maximal zu benutzenden Kerne* fest
- Die Kartenansicht wird im Hintergrund in einem separaten Image gerendert, und bei jedem *Kartenaktualisierungsintervall* (Standardwert 250 ms) wird der Inhalt dieses (nicht auf dem Bildschirm angezeigten) Images zur Aktualisierung der sichtbaren Kartenansicht verwendet. Wenn das Rendering der Kartenansicht jedoch schneller als diese Zeitspanne abgeschlossen ist, wird es sofort angezeigt.
- *Vergrößerungsgrad* (siehe Abschnitt: <magnifier>)

### Zeichenqualität

- ☒ *Linie auf Kosten der Zeichengeschwindigkeit weniger gezackt zeichnen*

## Einstellungen zur Vektor-Darstellung

Der Reiter  *Vektor* enthält spezifische Einstellungen für die Darstellung von Vektor-Layern.

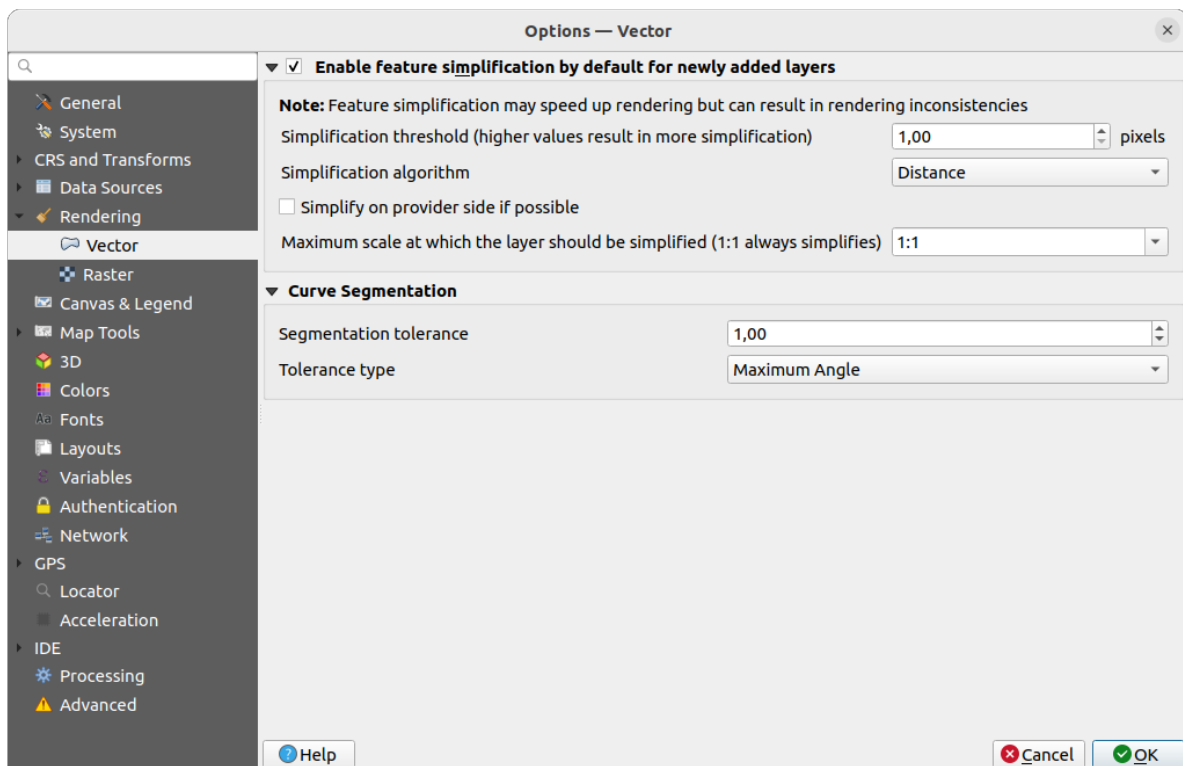



Abb. 9.11: Einstellungen zur Vektor-Darstellung

-  *Geometrievereinfachung für neue Layer voreinstellen:* Sie können die Geometrie von Objekten in Vektorlayern vereinfachen (weniger Knoten), was zu einer schnelleren Darstellung führt. Beachten Sie jedoch, dass dies zu Inkonsistenzen bei der Darstellung führen kann. Verfügbare Einstellungen sind:
  - Vereinfachungsschwelle (höhere Werte führen zu stärkerer Vereinfachung)
  - Vereinfachungsalgorithmus: Diese Option führt eine „on-the-fly“ Vereinfachung von Objekten durch und beschleunigt so das Rendering der Geometrie. Die Geometrie der verwendeten Datenquellen wird dabei nicht verändert. Dies ist wichtig, wenn Sie Berechnungen haben, die die Geometrien der Objekte verwenden (z.B. die Fläche): es stellt sicher, dass diese Berechnungen auf der ursprünglichen Geometrie und nicht auf der vereinfachten Geometrie durchgeführt werden. Es stehen drei verschiedene Algorithmen hierfür zur Verfügung: ‚Entfernung‘ (Standard), ‚SnapToGrid‘ und ‚Visvalingam‘.
  - ☐ Wenn möglich auf Anbieterseite vereinfachen: die Geometrien werden von der Datenquellenseite (PostGIS, Oracle...) vereinfacht. ACHTUNG: im Gegensatz zur lokalen Vereinfachung können die geometriebasierten Berechnungen davon betroffen sein!
  - Größter Maßstab bis zu dem der Layer vereinfacht werden soll (1:1 vereinfacht immer)

---

**Bemerkung:** Neben der globalen Einstellung kann die Vereinfachung von Objekten für jeden einzelnen Layer über *Layereigenschaften* ► *Darstellung* eingestellt werden.

---

- *Kurvensegmentierung*
  - *Segmentierungstoleranz:* Diese Einstellung steuert die Art und Weise, wie Kreisbögen gerendert werden. **Je kleiner** der maximale Winkel (zwischen den beiden aufeinanderfolgenden Scheitelpunkten und dem Kurvenzentrum in Grad) oder die maximale Differenz (Abstand zwischen dem Segment der beiden

Scheitelpunkte und der Kurvenlinie in Karteneinheiten), desto **mehr geradlinige** Segmente werden beim Rendern verwendet.

- *Toleranztyp*: kann *Maximalwinkel* oder *Maximaldifferenz* zwischen Annäherung und Kurve sein.

## Raster Einstellungen

Der Reiter  *Raster* enthält spezifische Einstellungen für die Darstellung von Raster-Layern.

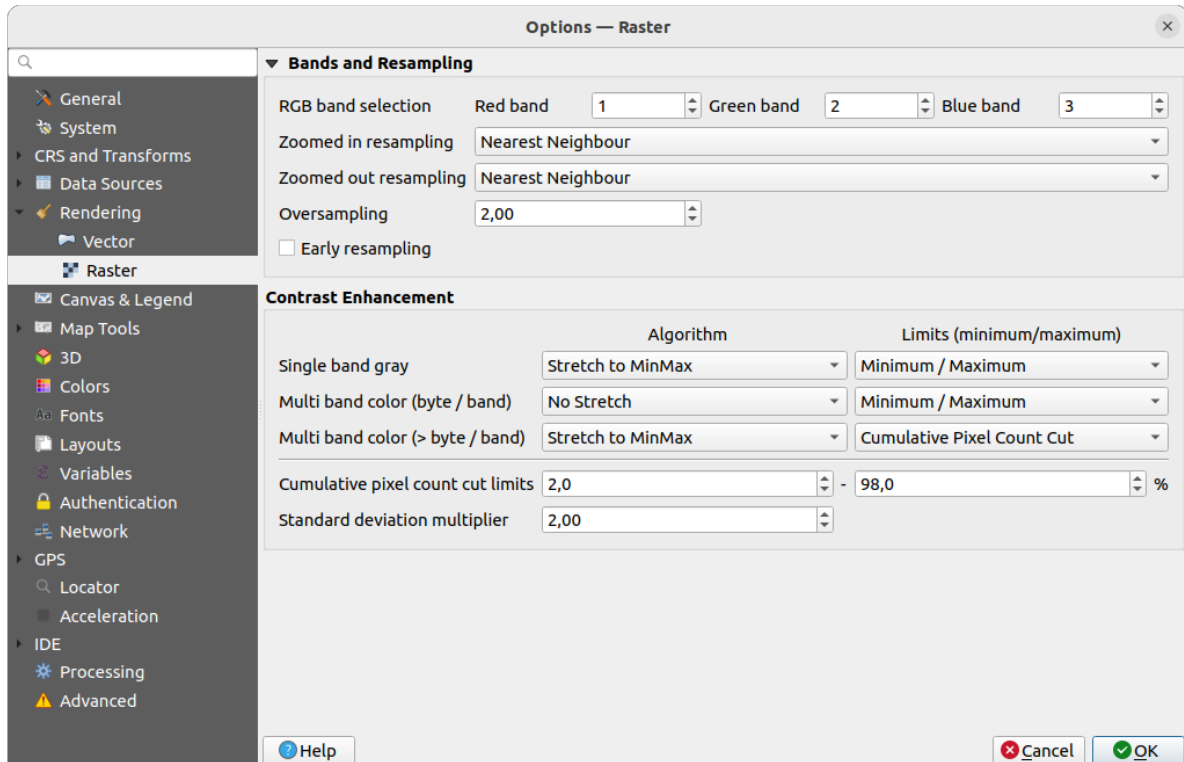



Abb. 9.12: Raster Einstellungen

Bereich *Kanäle und Abtastung*:

- Mit *RGB Kanalauswahl* können Sie die Nummer für den Roten, Grünen und Blauen Kanal festlegen.
- The *Zoomed in resampling* and the *Zoomed out resampling* methods can be defined. For *Zoomed in resampling* you can choose between three resampling methods: ‚Nearest neighbour‘, ‚Bilinear (2x2 kernel)‘ and ‚Cubic (4x4 kernel)‘. For *Zoomed out resampling* you can choose between ‚Nearest Neighbour‘ and ‚Average‘. You can also set the *Oversampling* value (between 0.0 and 99.99 - a large value means more work for QGIS - the default value is 2.0).
-  *Frühe Abtastung*: ermöglicht die Berechnung des Raster-Renderings auf der Seite des Datenanbieters (Datenquelle), wo die Auflösung der Quelle bekannt ist, und sorgt für ein besseres Zoomen beim Rendering mit QGIS benutzerdefiniertem Stil. Dies ist sehr praktisch für Vektor Tiles, die mit einer *Interpretationsmethode* geladen werden. Die Option kann auch auf Layer-Ebene gesetzt werden (*Symbolisierung* Eigenschaften).

*Kontrastverbesserung* kann für *Einkanalgraustufen*, *Multikanalfarbe (Byte/Kanal)* oder *Multikanalfarbe (> Byte/Kanal)* angewendet werden. Für jeden Punkt können Sie folgendes festlegen:

- den zu verwendenden *Algorithmus*, welcher die Werte ‚Kein Strecken‘, ‚Strecken auf MinMax‘, ‚Auf MinMax strecken und zuschneiden‘ oder ‚Zuschneiden auf MinMax‘ haben kann
- die *Grenzen (Minimum/Maximum)* festlegen, welche die Werte ‚Kumulativer Pixelanzahl-Schnitt‘, ‚Minimum/Maximum‘ oder ‚Mittlere +/- Standardabweichung‘ haben kann.

Die *Kontrastverbesserung* enthält darüber hinaus folgende Optionen:

- *Kommulative Pixelanzahl-Schnittgrenzen*
- *Standardabweichungsfaktor*

## 9.1.7 Karte & Legende

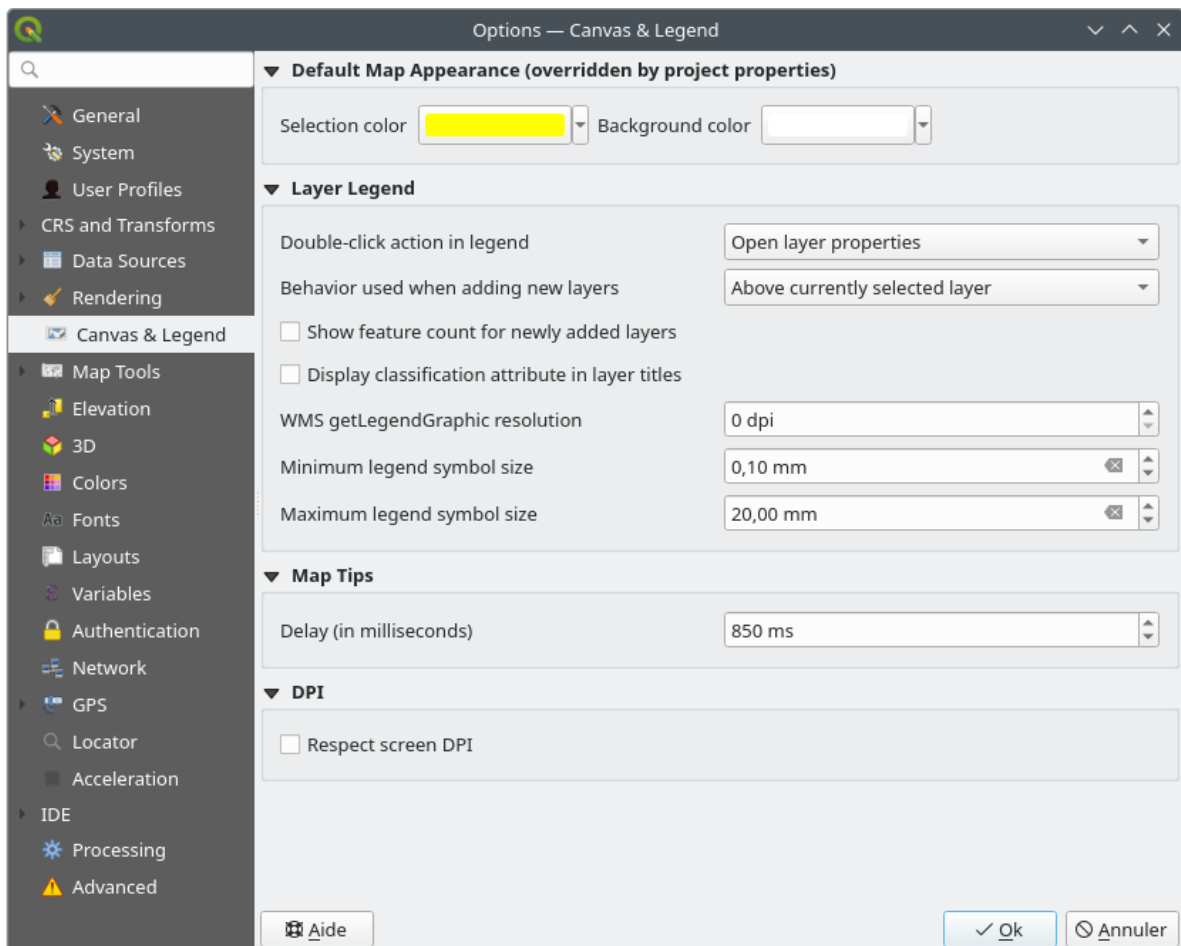





Abb. 9.13: Optionen auf dem Karte und Legende Reiter



Hier haben Sie die folgenden Einstellungsmöglichkeiten:

- unter **Voreingestelltes Kartenaussehen (Projekteigenschaften überschreiben)**: können Sie die *Auswahlfarbe* und die *Hintergrundfarbe* die verwendet werden sollen einstellen.
- Anpassungsmöglichkeiten für die **Layerlegende**:
  - *Double click action in legend*: whether a double-click on a layer should either *Open layer properties* (default), *Open attribute table* or *Open layer styling dock*.
  - *Behavior used when adding new layers*: determines where layers are placed in the *Layers* panel when loaded into QGIS. It can be:
    - \* *Above currently selected layer*
    - \* *Always on top of the layer tree*
    - \* *Optimal index within current layer tree group*: Unlike the other options that sort the new layers among them and place them as a stack at the desired location, this option extends the sorting logic to the



whole layer tree (or the active group) and inserts new layers in an „optimal“ fashion by insuring that point layers sit on top of point layers, followed by line layers on top of lines, followed by polygon layers, etc.

-  *Objektanzahl für neu hinzugefügten Vektorlayer anzeigen*: zeigt im Layer Bedienfeld die Anzahl der Objekte neben dem Layernamen an. Die Anzahl der Objekte in den Klassen, falls vorhanden, wird ebenfalls angezeigt. Sie können mit der rechten Maustaste auf einen Layer klicken, um die Anzahl der Objekte ein- oder auszuschalten.
-  *Klassifikationsattribute in Layertiteln anzeigen* im „Layer“ Bedienfeld, z. B. bei der Anwendung einer kategorisierten oder regelbasierten Darstellung (s. Abschnitt *Eigenschaften Symbolisierung* für weitere Informationen).
- die *WMS-getLegendGraphic-Auflösung*
- mit *Minimale Legendensymbolgröße* und *Maximale Legendensymbolgröße* können Sie die Größe, mit der Symbole im „guilabel:Layer“ Bedienfeld angezeigt werden, einstellen
- die *Verzögerung (in Millisekunden)* mit der *Kartenhinweise* eines Layers angezeigt werden
- wenn  *Bildschirmauflösung berücksichtigen*: aktiviert ist, wird QGIS versuchen, die Kartenansicht mit physikalisch korrektem Verhältnis auf dem Bildschirm darzustellen, abhängig von der Auslösung des Monitors. Symbologien mit angegebenen Bildschirmgröße werden ebenfalls genau dargestellt, d.h. bei einer Einstellung von 10mm wird ein Symbol auch 10mm groß auf dem Bildschirm angezeigt. Die Schriftgrößen der Beschriftungen in der Kartenansicht können jedoch von denen in der QGIS Benutzeroberfläche oder anderen Anwendungen abweichen. Wenn diese Einstellung deaktiviert ist, verwendet QGIS die logische DPI des Betriebssystems, die mit anderen Anwendungen auf dem System übereinstimmt. Allerdings können die Kartenansicht und die Symbologiegröße auf dem Bildschirm ungenau sein. Insbesondere auf Bildschirmen mit hohen DPI-Werten erscheinen Symbologien wahrscheinlich zu klein.

Es wird empfohlen, die Option  *Bildschirmauflösung berücksichtigen* zu aktivieren, insbesondere wenn Sie mehrere oder unterschiedliche Monitore verwenden und optisch hochwertige Karten erstellen wollen. Die Deaktivierung von  *Bildschirmauflösung berücksichtigen* erzeugt eine Ausgabe, die möglicherweise besser für Karten geeignet ist, die nur für die Verwendung auf dem Bildschirm bestimmt sind, insbesondere wenn die Schriftgrößen anderen Anwendungen entsprechen sollen.

---

**Bemerkung:** Die Darstellung in Drucklayouts wird von der Einstellung *Bildschirmauflösung berücksichtigen* nicht beeinflusst; hier wird immer die angegebenen Auflösung verwendet. Beachten Sie auch, dass diese Einstellung die vom Betriebssystem gemeldete physische Bildschirm-Auflösung verwendet, der möglicherweise nicht bei allen Bildschirme korrekt ist.

---

## 9.1.8 Kartenwerkzeuge

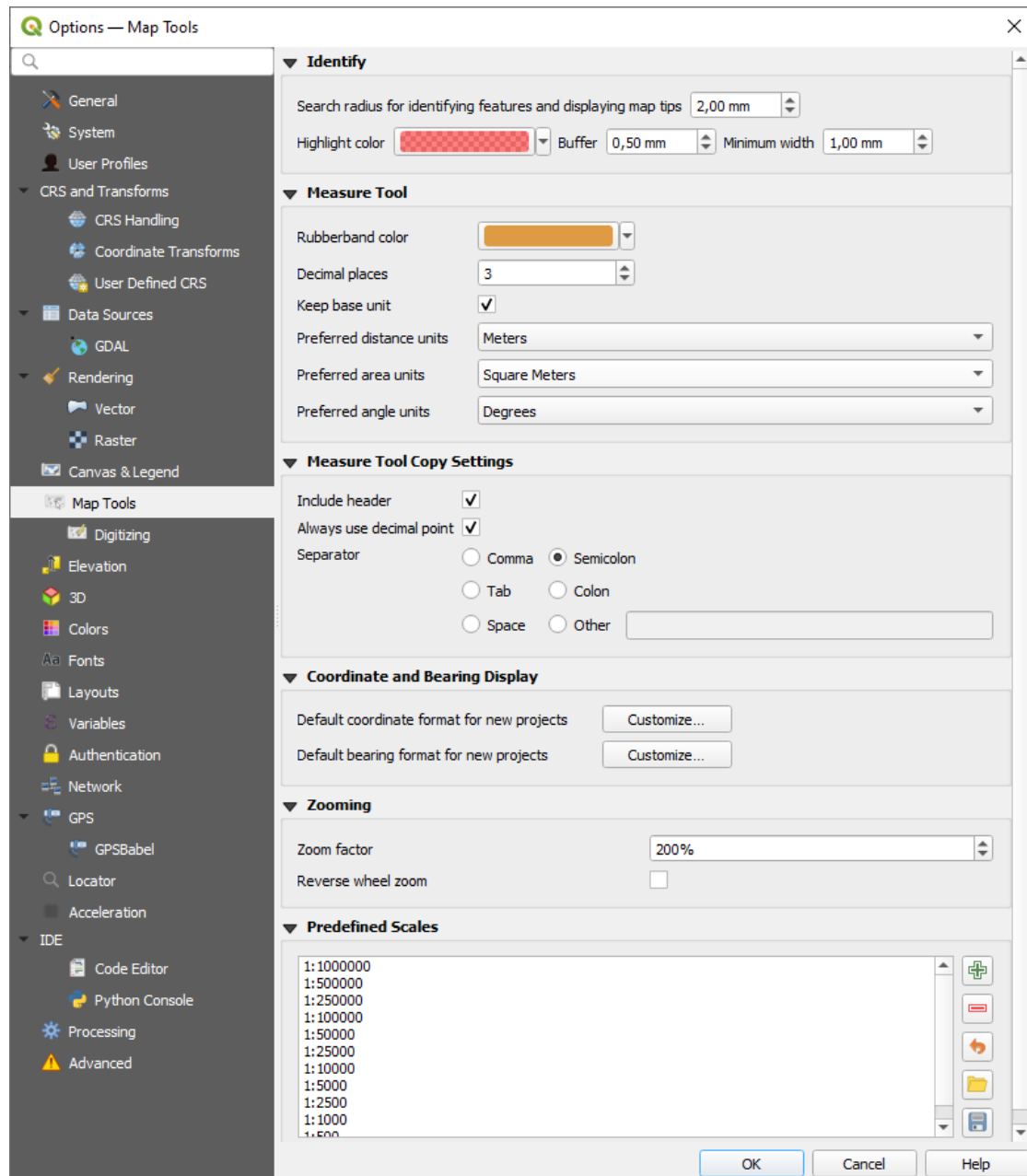



Abb. 9.14: Einstellungen Kartenwerkzeuge

Im obersten Bereich können einige Einstellungen im Bezug auf das *Objekte abfragen-Werkzeug* vorgenommen werden

- Mit *Suchradius* für die Objektidentifikation und zur Anzeige von Kartenhinweisen kann der Abstand eingestellt werden, innerhalb dessen das Identifizierungswerkzeug Ergebnisse anzeigt
- Mit der *Hervorhebungsfarbe* kann eingestellt werden, wie abgefragte Objekte angezeigt werden sollen
- *Puffer* legt die Größe des Puffers fest, mit der die Hervorhebung dargestellt wird
- *Minimalbreite* legt fest, wie dick die Umrisslinie eines hervorgehobenen Objektes sein soll




### Messwerkzeug

- Definieren Sie die *Gummibandfarbe* für das Messwerkzeug

- Definieren Sie die Anzahl der *Dezimalstellen*, die beim Messen angezeigt werden
- Wenn  *Basiseinheit beibehalten* aktiviert ist, werden große Zahlen nicht automatisch umgewandelt (z. B. Meter in Kilometer)
- *Preferred distance units*: options are ‚Meters‘, ‚Kilometers‘, ‚Feet‘, ‚Yards‘, ‚Miles‘, ‚Nautical Miles‘, ‚Centimeters‘, ‚Millimeters‘, ‚Inches‘, ‚Degrees‘ or ‚Map Units‘
- *Preferred area units*: options are ‚Square meters‘, ‚Square kilometers‘, ‚Square feet‘, ‚Square yards‘, ‚Square miles‘, ‚Hectares‘, ‚Acres‘, ‚Square nautical miles‘, ‚Square centimeters‘, ‚Square millimeters‘, ‚Square inches‘, ‚Square degrees‘ or ‚Map Units‘
- Die Optionen für das *Bevorzugte Winkelmaß* sind: ‚Grad‘, ‚Bogenmaß‘, ‚Gon/Neugrad‘, ‚Bogenminuten‘, ‚Bogensekunden‘, ‚Umdrehungen‘, ‚Milliradianen (SI Definition)‘ und ‚Mil (NATO/militärische Definition)‘ or mil (NATO/military definition)



Some of these options can be overridden at the [project level](#).

### Measure Tool Copy Settings

These settings allow you to control the behavior of the *Copy* button in the  Measure Line window. You can choose to  *Include header* to keep column names, and you can also select *Separator* of your choice. You can also choose  *Always use decimal point* to keep your coordinates copied to the clipboard with a dot as a decimal separator, even if your language settings in QGIS options are set up to use a comma. In this case, if you don't have the *Always use decimal point* option checked, you will be unable to select comma as a field separator.


### Koordinaten- und Kursanzeige

In diesem Bereich können Sie die folgenden Parameter *Anpassen ...*:



- *Vorgabekoordinatenformat für neue Projekte*, wie es im Koordinatenfeld in der QGIS-Statusleiste und im Bereich *Koordinaten* des  *Objekte abfragen* Werkzeugs angezeigt wird
- *Kursformat für neue Projekte* wie es in der Richtungsanzeige der Statusleiste und vom  *Kurs messen* Werkzeug angezeigt wird

Diese Optionen können auf [Projekt-Ebene](#) überschrieben werden.

### Zoomen

- Legen Sie einen *Zoomfaktor* für Zoom-Werkzeuge und Mausrad fest.
-  *Reverse wheel zoom* allows you to adjust mouse wheel scrolling direction.

### Vordefinierte Maßstäbe

Hier finden Sie eine Liste vordefinierter Maßstäbe, die standardmäßig in den maßstabsbezogenen Dropdown-Menüs angezeigt werden (z. B. in der *Maßstab* Statusleiste, in der Auswahl der Sichtbarkeitsmaßstäbe oder in den Einstellungen für weitere 2D-Kartenansichten). Mit den Schaltflächen  und  können Sie Ihre eigenen Maßstäbe hinzufügen oder entfernen. Sie können die Liste auch in eine XML-Datei exportieren oder daraus importieren. Auf der rechten Seite steht eine Schaltfläche zur Verfügung, um die Liste wieder auf die Standardeinträge zurückzusetzen.

Im Dialogfeld der Projekteigenschaften können Sie auch eine eigene Liste von Maßstäben festlegen, die diese globale Liste in den Dropdown-Menüs überschreibt.

## Digitalisieren

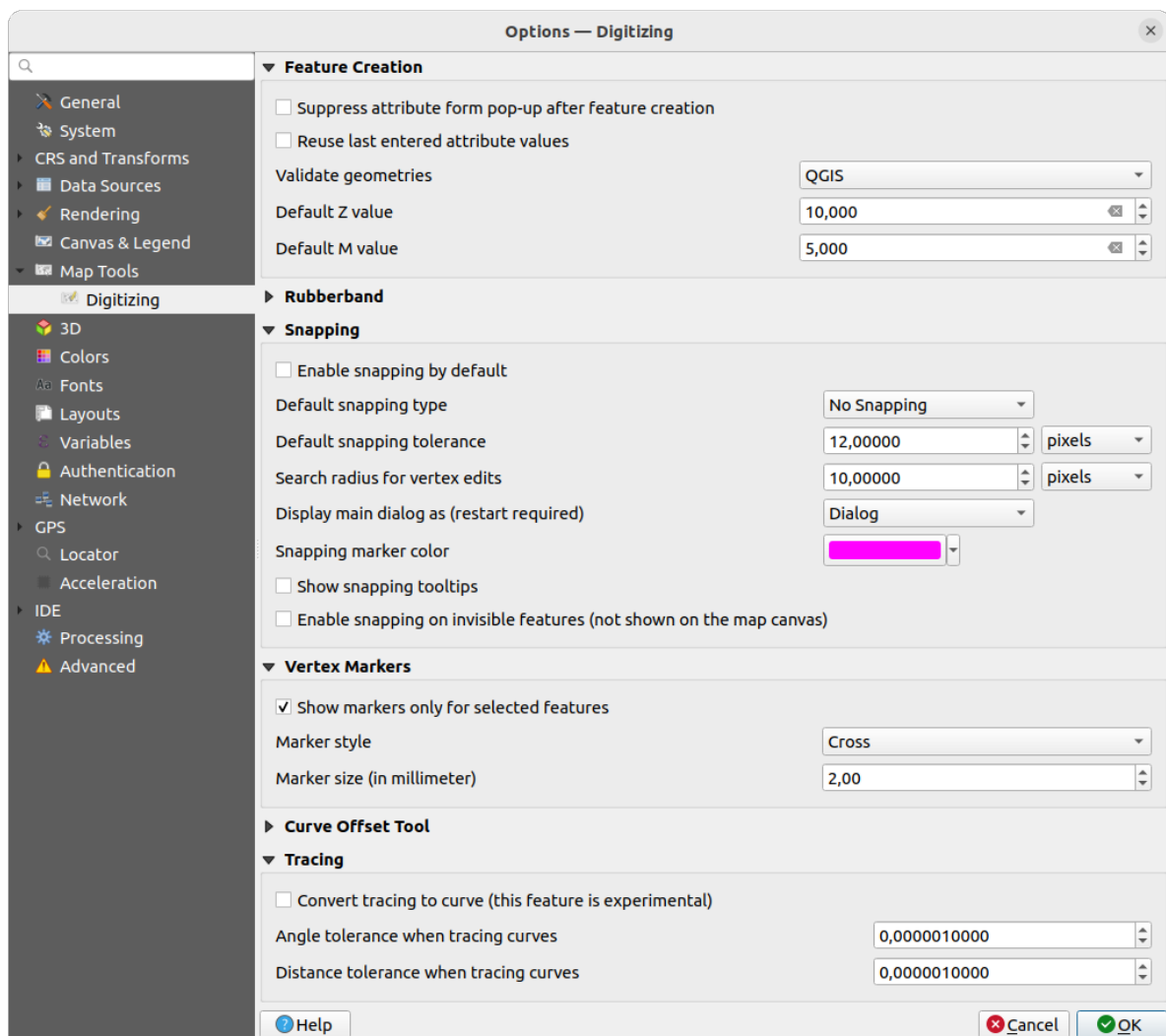


Abb. 9.15: Digitalisieren

Hier können Sie grundsätzlich Einstellungen für die *Bearbeitung eines Vektorlayers* festlegen (Geometrien und Attribute).

### Objekterzeugung

- ☒ *Attributeingabeformular bei der Objekterstellung unterdrücken*; diese Auswahl kann aber für jeden Layer in den Layer-Eigenschaften überschrieben werden.
- ☒ *Letzte Attributwerteingabe wiederverwenden*: merkt sich den zuletzt verwendeten Wert eines jeden Attributs und verwendet ihn als Standard für das nächste digitalisierte Objekt (für jeden Layer separat). Dieses Verhalten kann auch für einzelne Felder gesteuert werden (see *Einstellung der Feldfunktionsweisen*).
- *Geometrien prüfen*. Das Bearbeiten komplexer Linien/Polygone mit vielen Stützpunkten kann zu einer erheblichen Verlangsamung der Darstellung führen. Grund dafür ist das Standard-Validierungsverfahren, das viel Zeit benötigen. Um die Darstellung zu beschleunigen ist es möglich die Geometrieüberprüfung auszuschalten oder GEOS zu wählen. Die GEOS Geometrieüberprüfung ist deutlich schneller, hat aber den Nachteil, dass nur das erste Geometrieproblem gemeldet wird.





Beachten Sie, dass je nach Auswahl die Berichte über Geometriefehler unterschiedlich ausfallen können (see *Types of error messages and their meanings*)

- Der *Z-Vorgabewert* wenn neue 3D Objekte erstellt werden kann hier festgelegt werden.



### Gummiband

- Hier können Sie die *Linienstärke*, *Linienfarbe* and die *Füllfarbe* festlegen.
- *Gummiband bei der Stützpunktbearbeitung nicht aktualisieren.*


### Einrasten

- Mit  *Einrasten voreinstellen* legen Sie fest, dass die Einrastfunktion aktiviert ist, wenn ein Projekt geöffnet wird
- Define *Default snap mode*  (,Vertex‘, ,Segment‘, ,Centroid‘, ,Middle of segments‘, ,Line endpoints‘, ,Area‘)
- Definieren Sie die *Einrasttoleranzvorgabe* in Karteneinheiten oder Pixeln
- Definieren Sie den *Suchradius für Stützpunktbearbeitung* in Karteneinheiten oder Pixeln
- Mit *Hauptfenster anzeigen in (erfordert Neustart)* können Sie festlegen ob das Einrasten-Fenster als ,Dialog‘ oder als ,Dock‘ angezeigt wird
- Die *Einrastmarkierfarbe* kann ausgewählt werden
- Wenn  *Einrasthinweise anzeigen* aktiviert ist, wird z.B. der Name des Layers angezeigt, an den gefangen wird. Das ist hilfreich, wenn mehrere Layer übereinander liegen.
-  *Einrasten auf verborgene (nicht auf der Karte sichtbare) Objekte aktivieren*

### Stützpunktmarkierungen


-  *Markierungen nur für gewählte Objekte anzeigen*
- Legen Sie für die Stützpunktmarken den *Markierungsstil*  (,Kreuz‘ (standard), ,Teiltransparenter Kreis‘ oder ,Keine‘) fest
- Definieren Sie die *Markierungsgröße (in Millimeter)*

**\*\*Werkzeug zum versetzen von Linien \*\***

Die nächsten 3 Optionen beziehen sich auf das  *Linie versetzen* Werkzeug in *Erweiterte Digitalisierung*. Durch die verschiedenen Einstellungen ist es möglich die Form des Linienversatzes zu beeinflussen. Diese Optionen sind von GEOS 3.3 an möglich.

- Der *Verbindungsstil* kann ,Rund‘, ,Eckig‘ oder ,Abgefast‘ sein
- Anzahl der *Quadrantensegmente*
- *Eckengrenze*

### Spurverfolgung

Wenn  *Spurverfolgung in Kurve umwandeln* aktiviert ist, können Sie während der Digitalisierung automatisch Kurvensegmente erstellen. Bitte beachten Sie, dass es sich um eine experimentelle Funktion handelt und das der Datenanbieter, den Sie dazu verwenden, diese Funktion unterstützen muss.

## 9.1.9 Elevation settings

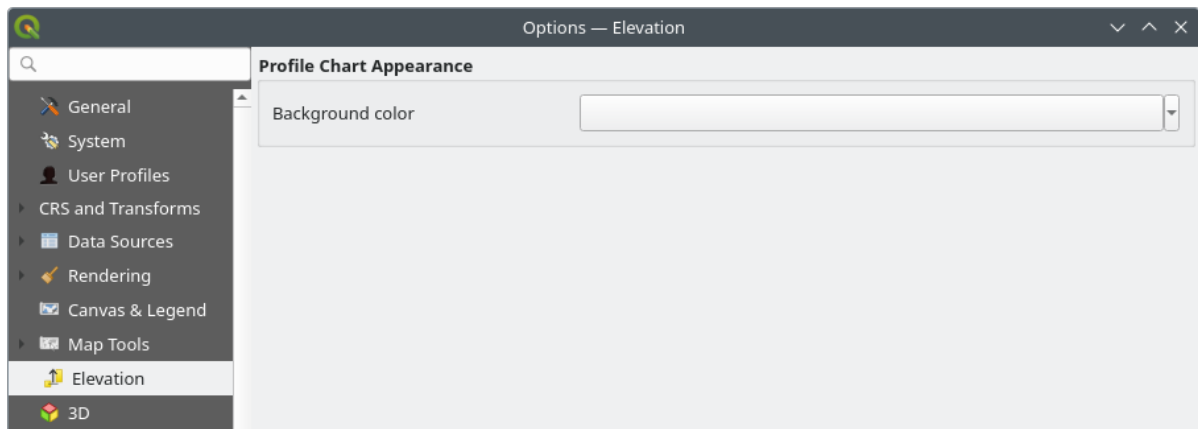



Abb. 9.16: Elevation settings

In the  **Elevation** menu, you can set a specific color to use as the *Background color* for *elevation profiles*. This can make the chart more readable for certain datasets, e.g. point clouds with RGB coloring, where the default background color is too similar to point colors to be easily discernable. If unchanged, the elevation profiles will continue to display using the standard system background color.

## 9.1.10 3D settings

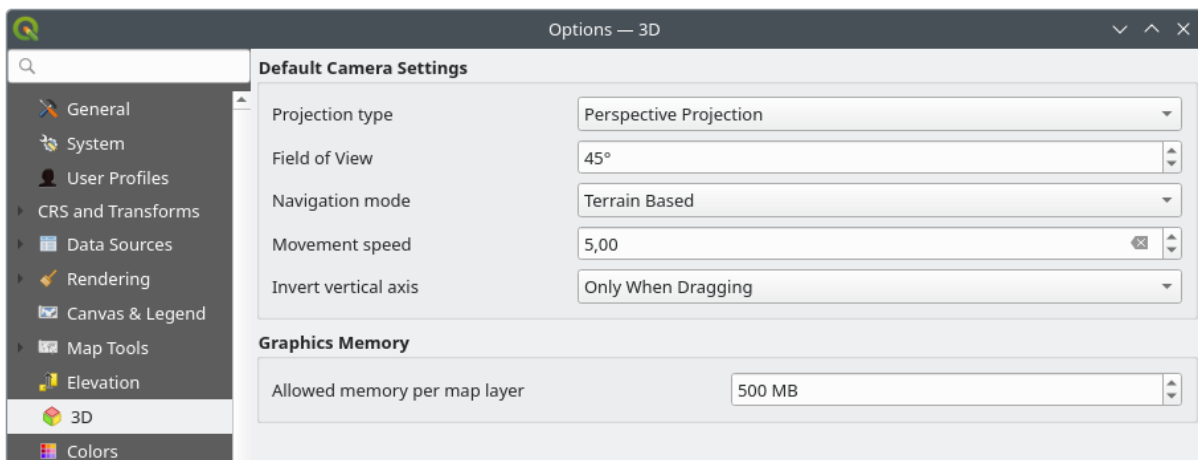



Abb. 9.17: 3D settings

The  **3D** menu helps you configure some default settings to use for any *3D Map view*. These can refer to:

- *Default Camera Settings:*
  - *Projection type:* allowing to view the 3D scene in a:
    - \* *Perspective projection* (default): Parallel lines appear to meet in the distance. Objects appear to shrink the farther they are from the camera.
    - \* or an *Orthogonal projection*: Parallel lines appear parallel. Objects appear the same size regardless of distance.

- Camera's *Field of view*: only relevant in perspective projection, specifies the current vertical field of view in degrees and determines how much of the scene is visible to the camera. Default value is 45°.
- *Navigation mode*: provides different means to interact with the 3D scene. Available modes are:
  - \* *Terrain based*: the camera follows around a fixed position on the surface of the terrain as the scene is navigated.
  - \* *Walk mode (first person)*

Depending on the selected mode, *navigation commands* differ.

- *Movement speed*
- *Invert vertical axis*: Controls whether vertical axis movements should be inverted from their normal behaviour. Only affects movement in the *Walk mode*. It can be set to:
  - \* *Never*
  - \* *Only when dragging*: causes the vertical motion to inverted only when performing a click-and-drag camera rotation
  - \* *and Always*: causes the motions to be inverted when both click-and-dragging and when the camera movement is locked to the cursor (via a ~ key press)
- Under *Graphics memory*, the *Allowed memory per layer* option lets you set the GPU memory limit configuration on each layer. This is useful for users utilizing large 3D scenes which exhaust the available GPU memory resources. When a limit is hit, a warning is also displayed, which should assist in troubleshooting large scenes.

### 9.1.11 Colors settings

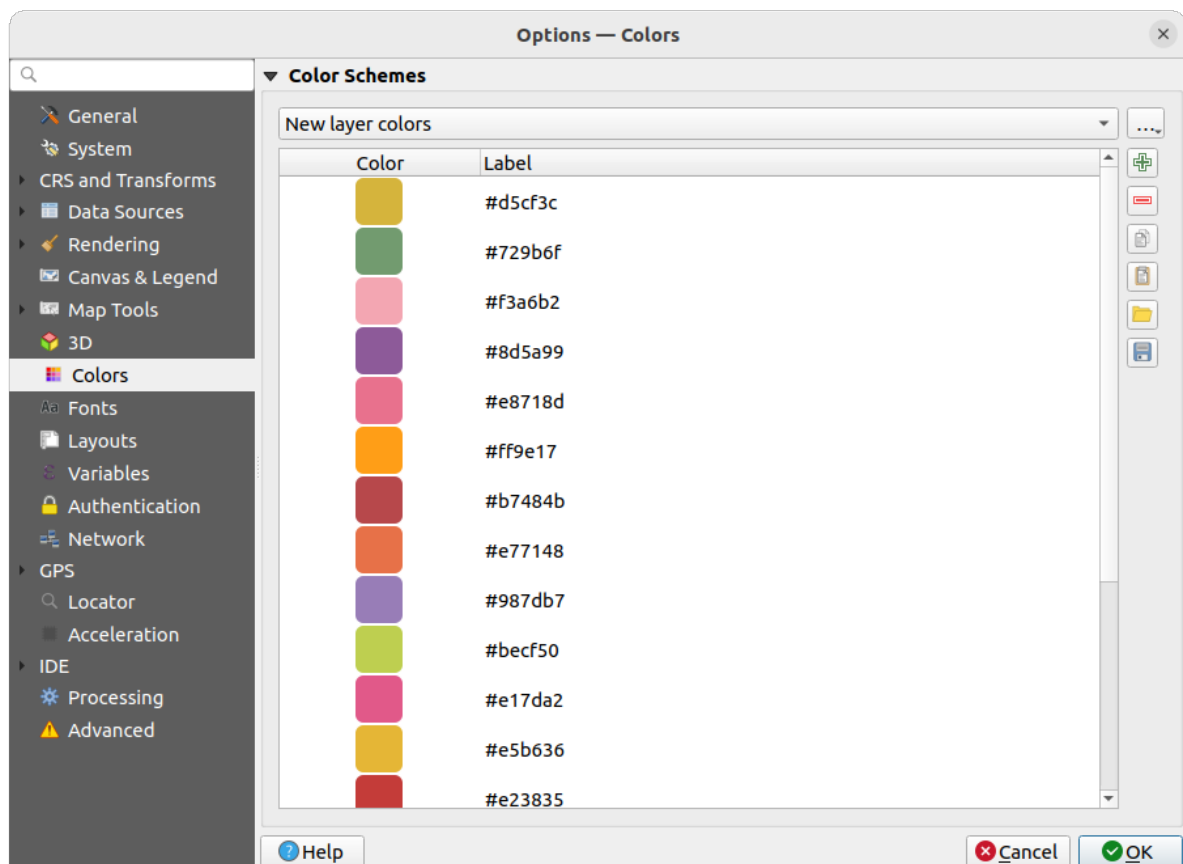








Abb. 9.18: Colors settings

This menu allows you to create or update palettes of colors used throughout the application in the *color selector widget*. You can choose from:

- *Recent colors* showing recently used colors
- *Standard colors*, the default palette of colors
- *Project colors*, a set of colors specific to the current project (see *Styles Properties* for more details)
- *New layer colors*, a set of colors to use by default when new layers are added to QGIS
- or custom palette(s) you can create or import using the ... button next to the palette combobox.

By default, *Recent colors*, *Standard colors* and *Project colors* palettes can not be removed and are set to appear in the color button drop-down. Custom palettes can also be added to this widget thanks to the *Show in Color Buttons* option.

For any of the palettes, you can manage the list of colors using the set of tools next to the frame, ie:

-  Add or  Remove color
-  Copy or  Paste color
-  Import or  Export the set of colors from/to .gpl file.

Double-click a color in the list to tweak or replace it in the *Color Selector* dialog. You can also rename it by double-clicking in the *Label* column.

### 9.1.12 Fonts Settings

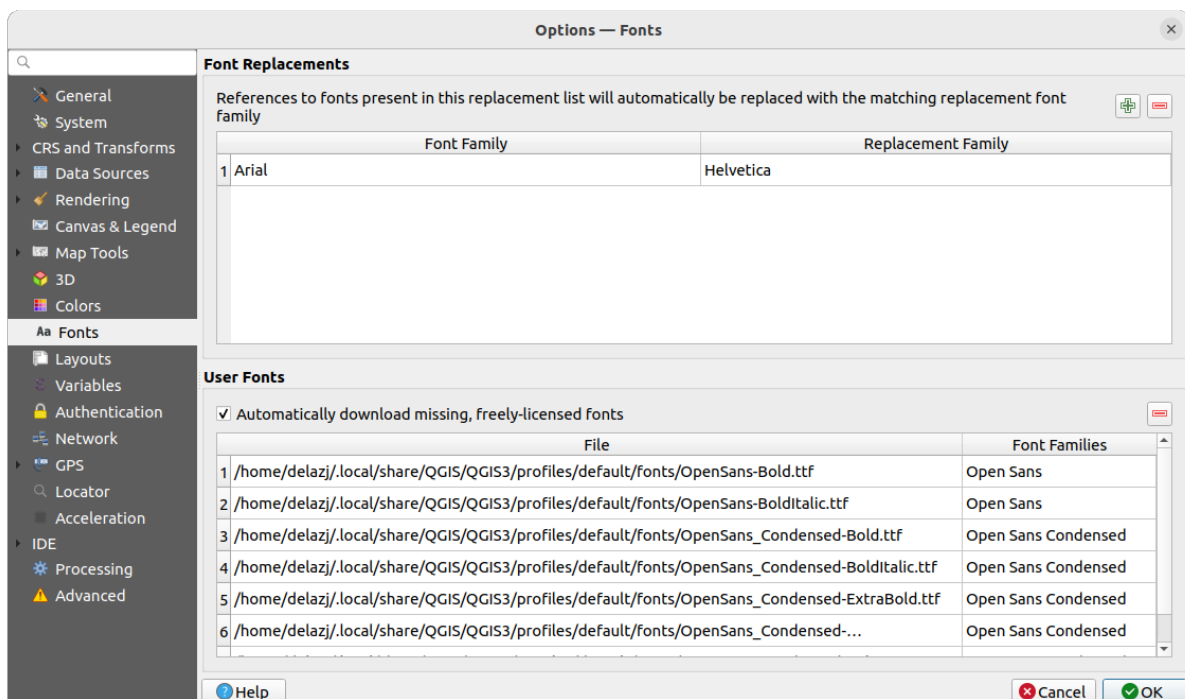



Abb. 9.19: Fonts settings

The *Fonts* tab provides support to manage fonts used across the projects:

- *Font Replacements*: gives ability to populate a list of automatic font replacements to apply when loading projects or styles, allowing better support for projects and styles to be used across different operating systems (e.g. replace „Arial“ with „Helvetica“).



- *User Fonts*: Allows you to place TTF or OTF fonts in the `fonts` sub-folder of the *user profile*. These fonts can be automatically loaded at QGIS startup time. This provides a way to use fonts without requiring them to be installed on an operating system level, which is often blocked in enterprise environments. The panel lists all installed user fonts and allows you to manage (i.e. remove) previously installed user fonts.

It is also possible to  *Automatically download missing, freely-licensed fonts*: E.g. if you open a project or style, or try to load a vector tile layer that references fonts that aren't currently available, then a hard-coded list of freely licensed fonts to download via URL is consulted to determine whether it's possible to automatically download the font to the user profile font directory (with notification of the font license).

### 9.1.13 Layouts settings

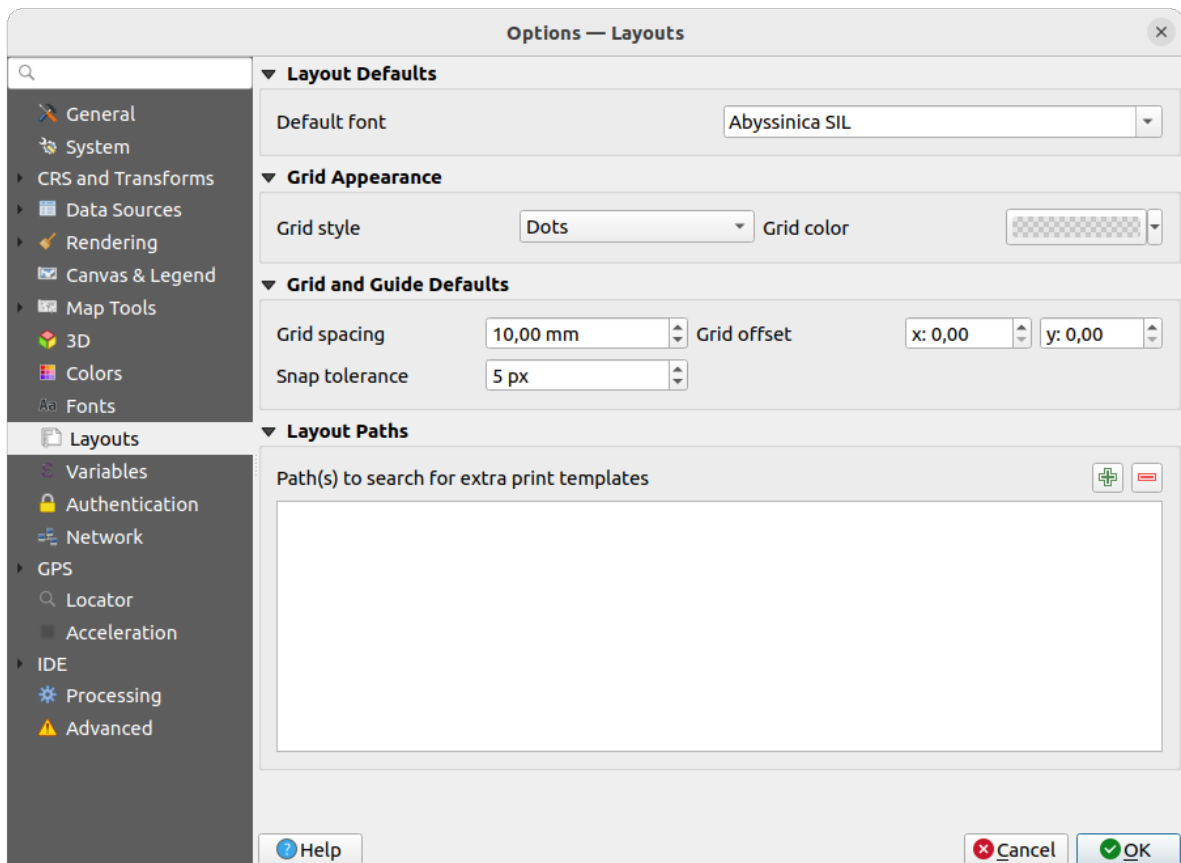


Abb. 9.20: Layouts settings

#### Zusammenstellungsvoreinstellungen

You can define the *Default font* used within the *print layout*.

#### Gitterdarstellung

- Define the *Grid style* („Solid“, „Dots“, „Crosses“)
- Definieren Sie *Gitterfarbe*

#### Gitter- und Führungsvoreinstellungen

- Define the *Grid spacing*
- Define the *Grid offset* for X and Y
- Define the *Snap tolerance*

## Layout Paths

- Define *Path(s) to search for extra print templates*: a list of folders with custom layout templates to use while creating new one.

### 9.1.14 Variables settings

The *Variables* tab lists all the variables available at the global-level.

It also allows the user to manage global-level variables. Click the  button to add a new custom global-level variable.

Likewise, select a custom global-level variable from the list and click the  button to remove it.

More information about variables in the *Storing values in Variables* section.

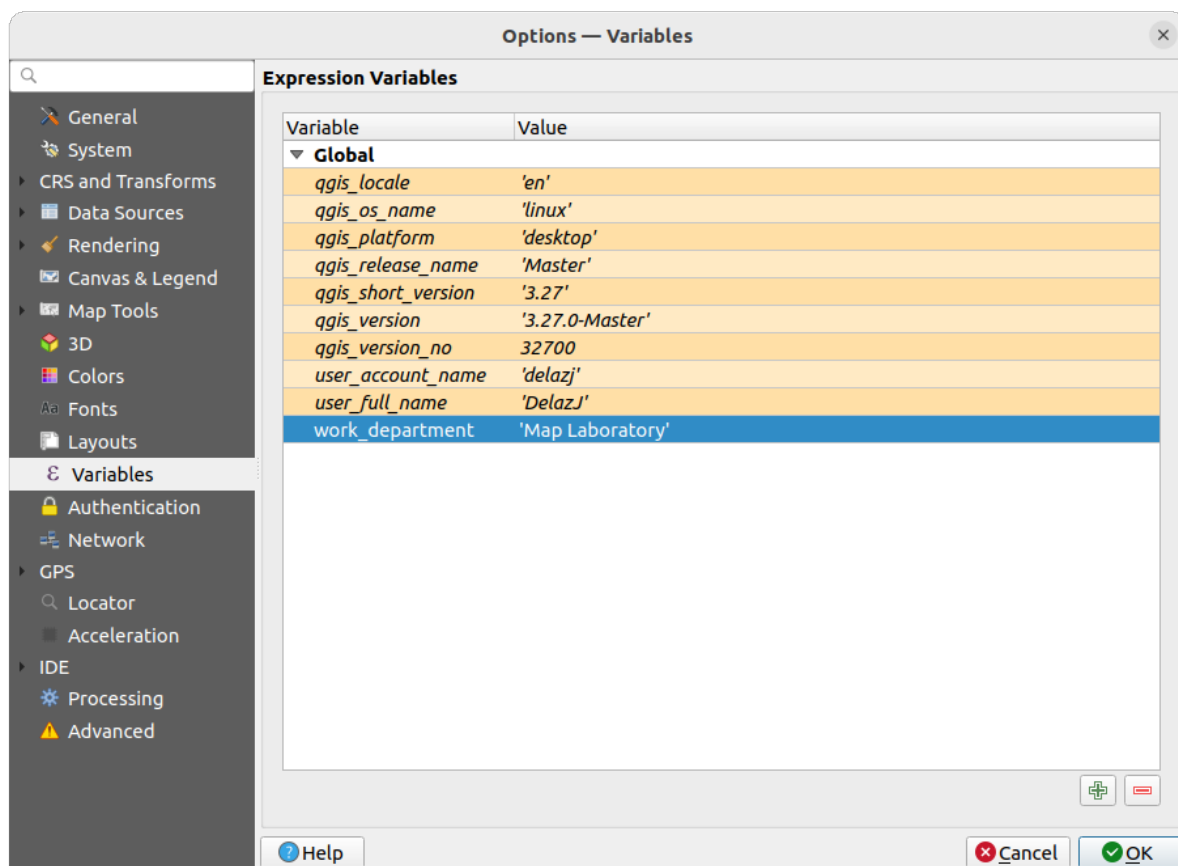





Abb. 9.21: Variables settings

### 9.1.15 Authentication settings

In dem *Authentifikation* Reiter können Sie Authentifikationskonfigurationen vornehmen und PKI Zertifikate verwalten. Für mehr Details siehe *Authentifizierungssystem*.

To manage authentications, you can use the list of tools next to the frame, ie:

-  Add new authentication configuration
-  Remove selected authentication configuration
-  Edit selected authentication configuration

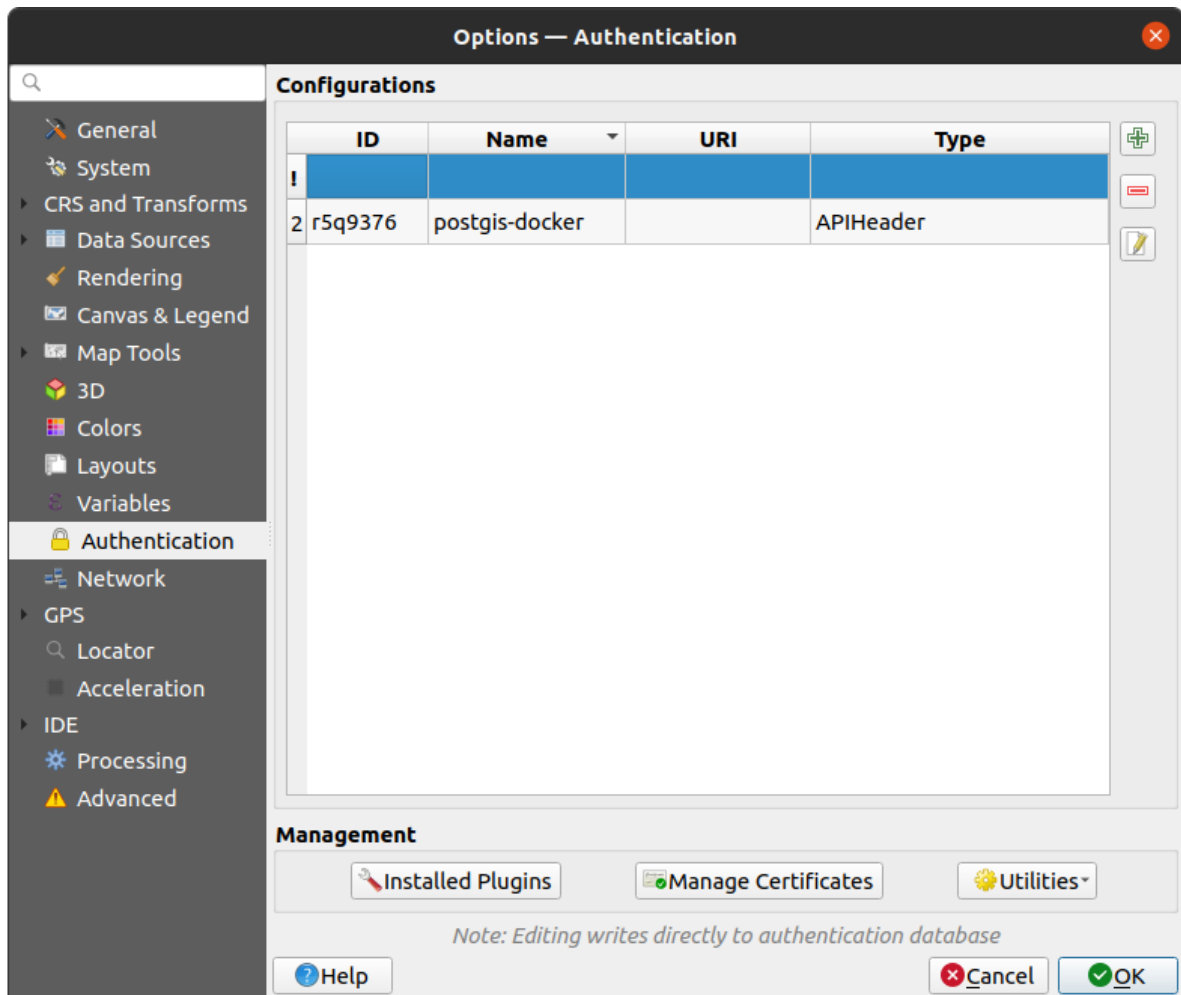


Abb. 9.22: Authentication settings

### 9.1.16 Network settings

#### Allgemein

- Definieren Sie *Zeitüberschreitung für Netzwerkanfragen (ms)* - Standard ist 60000
- Define *Default expiration period for WMS Capabilities (hours)* - default is 24
- Define *Default expiration period for WMS-C/WMTS tiles (hours)* - default is 24
- Define *Max retry in case of tile or feature request errors*
- Definieren Sie *User-Agent*

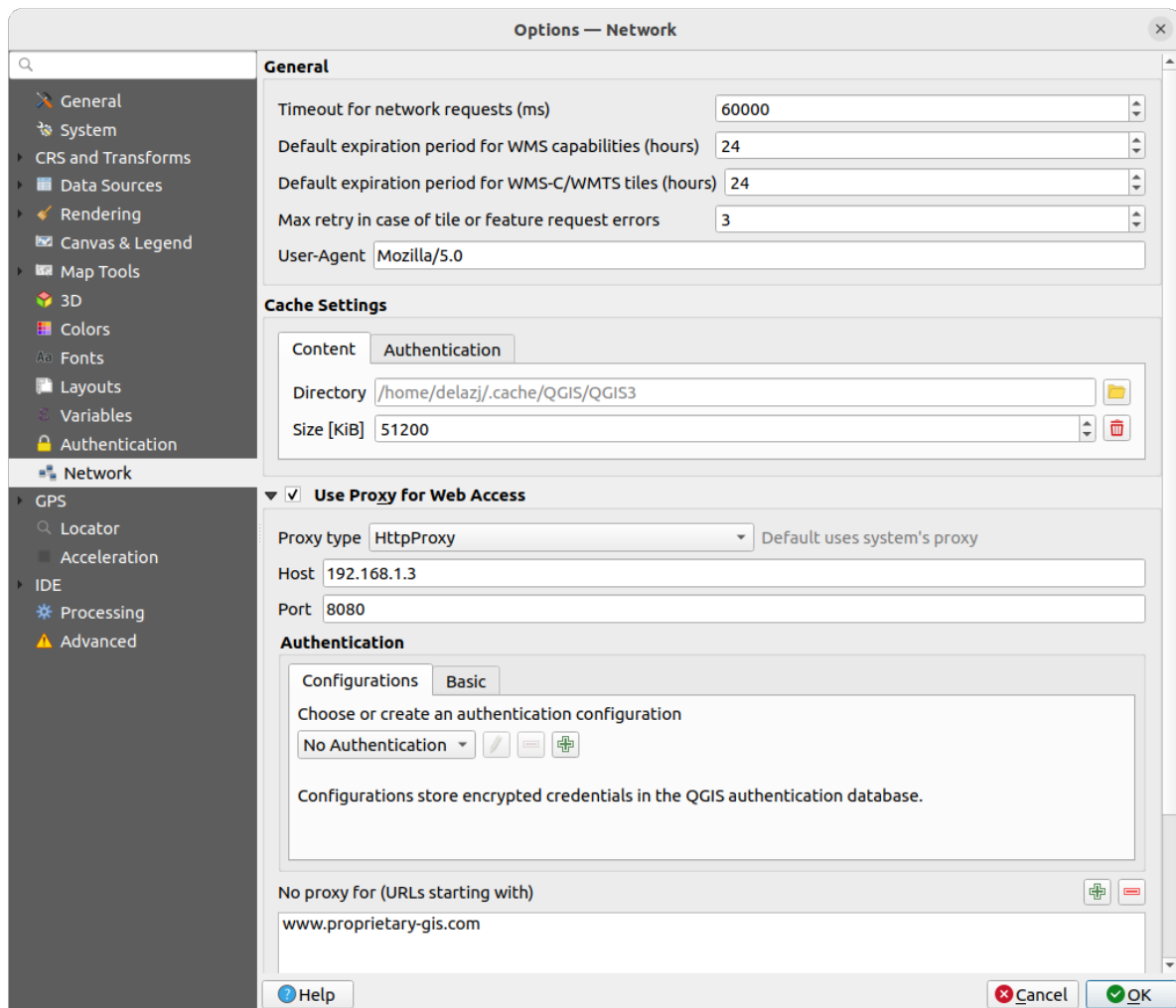


Abb. 9.23: Network and proxy settings

## Cache-Einstellungen

Defines the *Directory* and a *Size* for the cache. Also offers tools to *automatically clear the connection authentication cache on SSL errors (recommended)*.

## Proxy for web access

- Use proxy for web access
- Set the *Proxy type* according to your needs and define ‚Host‘ and ‚Port‘. Available proxy types are:
  - *Default Proxy*: Proxy is determined based on system’s proxy
  - *Socks5Proxy* ►: Proxy für jede Art von Verbindung. Unterstützt TCP, UDP, Bindung an einen Port (eingehende Verbindungen) und Authentifizierung.
  - *HttpProxy* ►: Umgesetzt mit dem „CONNECT“-Befehl, unterstützt nur ausgehende TCP-Verbindungen und Authentifizierung.
  - *HttpCachingProxy*: Umgesetzt mit normalen HTTP Befehlen ist dies nur im Zusammenhang mit HTTP Befehlen sinnvoll einzusetzen.
  - *FtpCachingProxy*: Mit einem FTP-Proxy umgesetzt ist dies nur sinnvoll im Zusammenhang mit FTP-Anforderungen anzuwenden

Credentials of proxy are set using the *authentication widget*.


Excluding some URLs can be added to the text box below the proxy settings (see Abb. 9.23). No proxy will be used if the target url starts with one of the string listed in this text box.

If you need more detailed information about the different proxy settings, please refer to the manual of the underlying QT library documentation at <https://doc.qt.io/archives/qt-5.9/qnetworkproxy.html#ProxyType-enum>

### Tipp: Proxy richtig anwenden

Using proxies can sometimes be tricky. It is useful to proceed by ‚trial and error‘ with the above proxy types, to check if they succeed in your case.

## 9.1.17 GPS settings

The  *GPS* dialog helps you configure GPS devices connections and properties in QGIS. It also provides settings for GPS tracking and data digitizing.

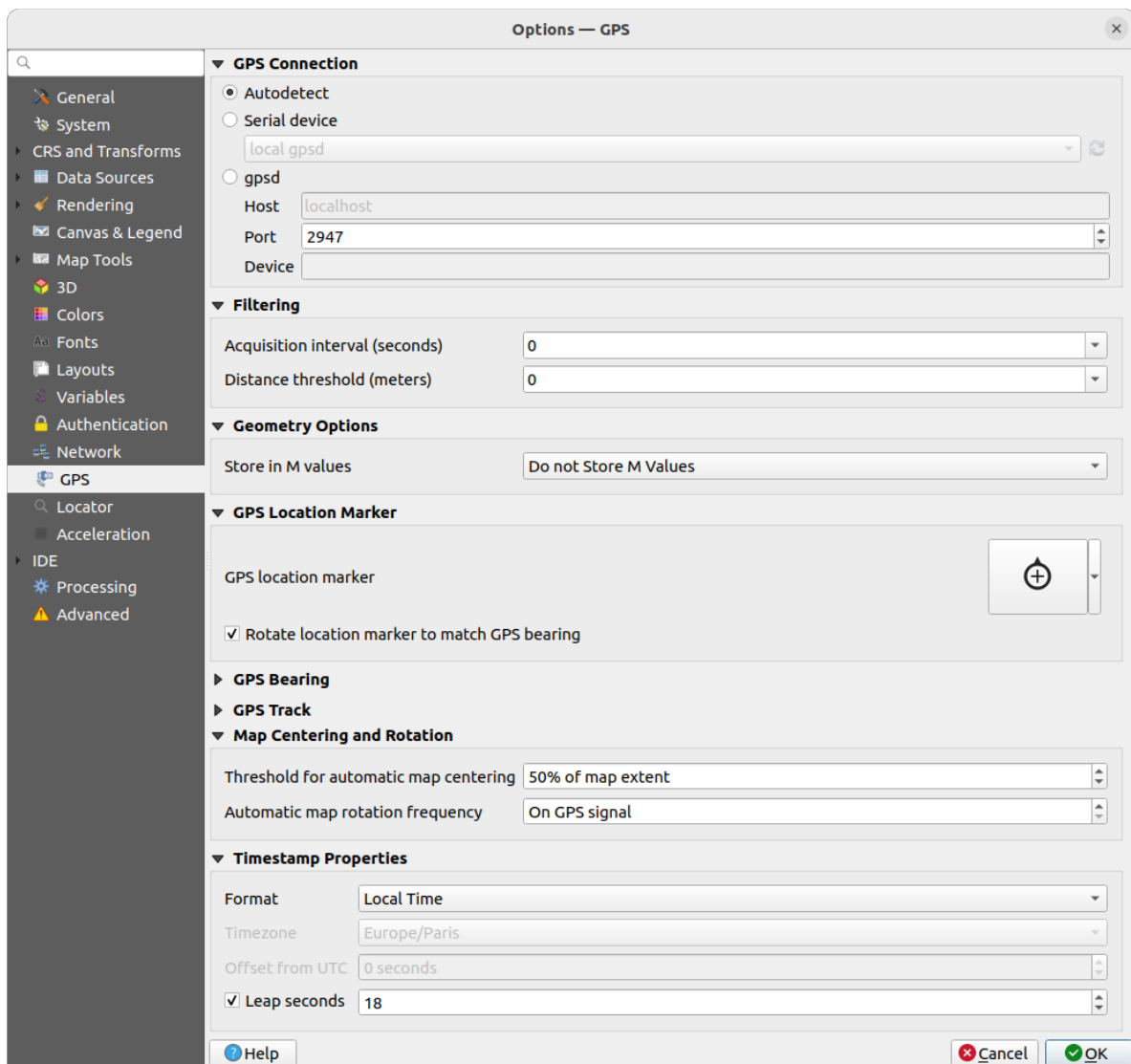







Abb. 9.24: GPS settings

You can specify:

- *GPS Connection*: provides different means to connect the device to QGIS

-  *Autodetect*
-  *Serial device* (reload required if a new GPS Device is connected)
-  *gpsd* (selecting the *Host* and *Port* your GPS is connected to, and providing a *Device* name)

In case of connection problems, you can try to switch from one to another.

- *Filtering*: You can set an *Acquisition interval (seconds)* and/or a *Distance threshold (meters)* parameters to keep the cursor still active when the receiver is in static conditions.
- Under *Geometry Options*, the *Store in M values* option allows to create geometries with M values from the inbuilt GPS logging tools. This applies to both features digitized from GPS logs and from the [Log to Geopackage/Spatialite](#) functionality... Options include storing timestamps (as ms since epoch), ground speed, altitudes, bearings, and accuracy components as m values.
- *GPS Location Marker*
  - Configure a symbol for the *GPS location marker*, indicating the current GPS position
  -  *Rotate location marker to match GPS bearing*: whether the marker symbol should be rotated to match the GPS direction
- *GPS Bearing*:
  - configure a *Bearing line style* using QGIS line symbol properties
  - set whether to  *Calculate Bearings from travel direction*: If checked, the bearing reported by the GPS device will be ignored and the bearing will instead be calculated by the angle between the previous two GPS locations.
- *GPS Track*: set symbol to use for the *Track line style*
- *Map Centering and Rotation*: defines when the map canvas is updated according to the user displacement on the field:
  - *Threshold for automatic map centering*: defines the minimal offset of the GPS position from the map canvas center to trigger an automatic *map recentering* if enabled.
  - *Automatic map rotation frequency*: defines how often the map rotation to match the GPS bearing could happen; it can be on a custom duration or *On GPS signal*.
- *Timestamp Properties* to configure how time values are displayed and stored in the data. Parameters include the *Format* which can be:
  - *Local time*
  - *UTC*
  - *UTC with offset*, to account for daylight savings offsets or other complex time zone issues
  - a specific *Time zone*

Moreover, *Leap seconds* correction can be applied, by adding the seconds to GPS timestamp.

## GPSTabel

**GPSTabel** converts waypoints, tracks, and routes between popular GPS receivers such as Garmin or Magellan and mapping programs like Google Earth or Basecamp. Literally hundreds of GPS receivers and programs are supported. QGIS relies on GPSTabel to interact with these devices and *manipulate their data*.

For details on how-to, please refer to *Loading to or from a device*.

### 9.1.18 Locator settings

The **Locator** tab lets you configure the *Locator bar*, a quick search widget available on the status bar to help you perform searches in the application. It provides some default filters (with prefix) to use:

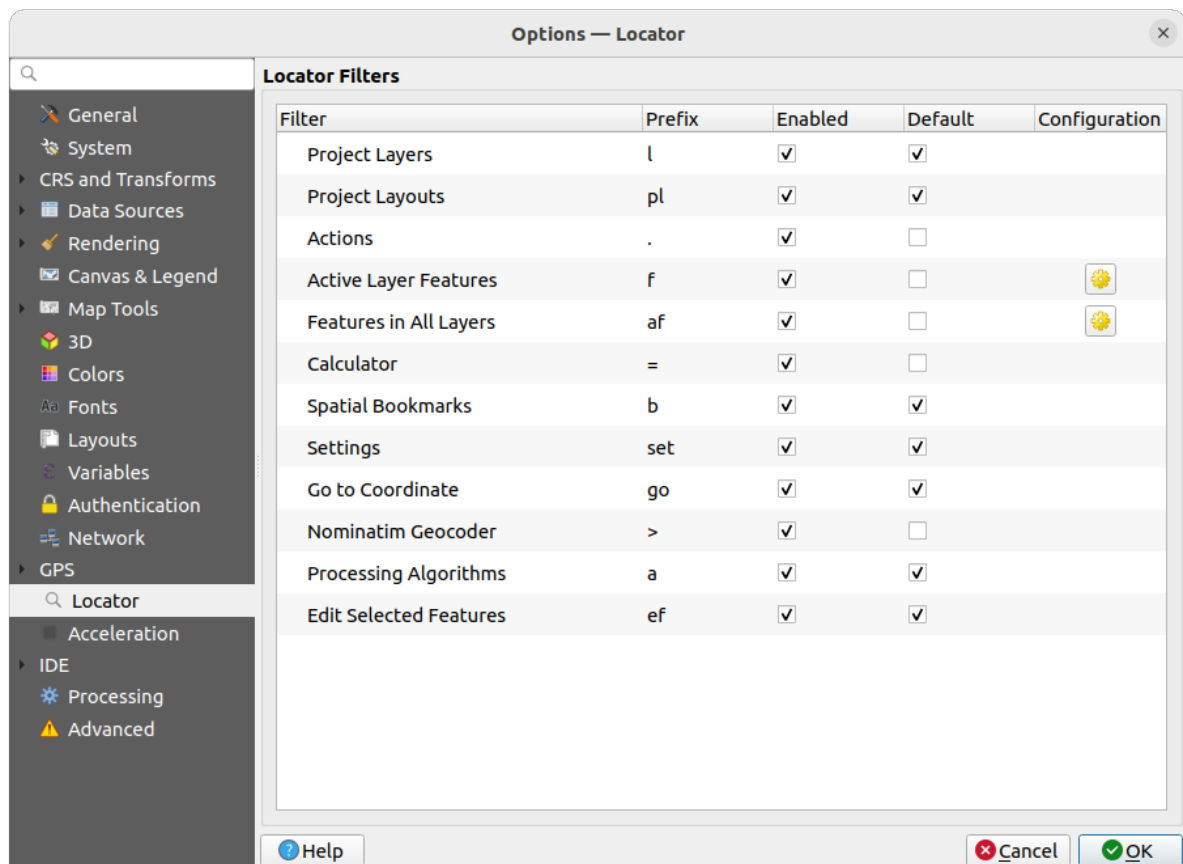


Abb. 9.25: Locator settings

- *Project Layers* (l): finds and selects a layer in the *Layers* panel.
- *Project Layouts* (pl): finds and opens a print layout.
- *Actions* (.): finds and executes a QGIS action; actions can be any tool or menu in QGIS, opening a panel...
- *Active Layer Features* (f): searches for matching attributes in any field from the current active layer and zooms to the selected feature. Press to configure the maximum number of results.
- *Features in All Layers* (af): searches for matching attributes in the *display name* of each *searchable layers* and zooms to the selected feature. Press to configure the maximum number of results and the maximum number of results per layer.
- *Calculator* (=): allows evaluation of any QGIS expression and, if valid, gives an option to copy the result to the clipboard.

- *Spatial Bookmarks* (b): finds and zooms to the bookmark extent.
- *Settings* (set): browses and opens project and application-wide properties dialogs.
- *Go to Coordinate* (go): pans the map canvas to a location defined by a comma or space separated pair of x and y coordinates or a formatted URL (e.g., OpenStreetMap, Leaflet, OpenLayer, Google Maps, ...). The coordinate is expected in WGS 84 (epsg: 4326) and/or map canvas CRS.
- *Nominatim Geocoder* (>): geocodes using the [Nominatim](#) geocoding service of the OpenStreetMap Foundation.
- *Processing Algorithms* (a): searches and opens a Processing algorithm dialog.
- *Edit Selected Features* (ef): gives quick access and runs a compatible *modify-in-place* Processing algorithm on the active layer.

In the dialog, you can:

- customize the filter *Prefix*, i.e. the keyword to use to trigger the filter
- set whether the filter is *Enabled*: the filter can be used in the searches and a shortcut is available in the locator bar menu
- set whether the filter is *Default*: a search not using a filter returns results from only the default filters categories.
- Some filters provide a way to configure the number of results in a search.

The set of default locator filters can be extended by plugins, eg for OSM nominatim searches, direct database searching, layer catalog searches, ...

### 9.1.19 Acceleration settings

OpenCL acceleration settings.

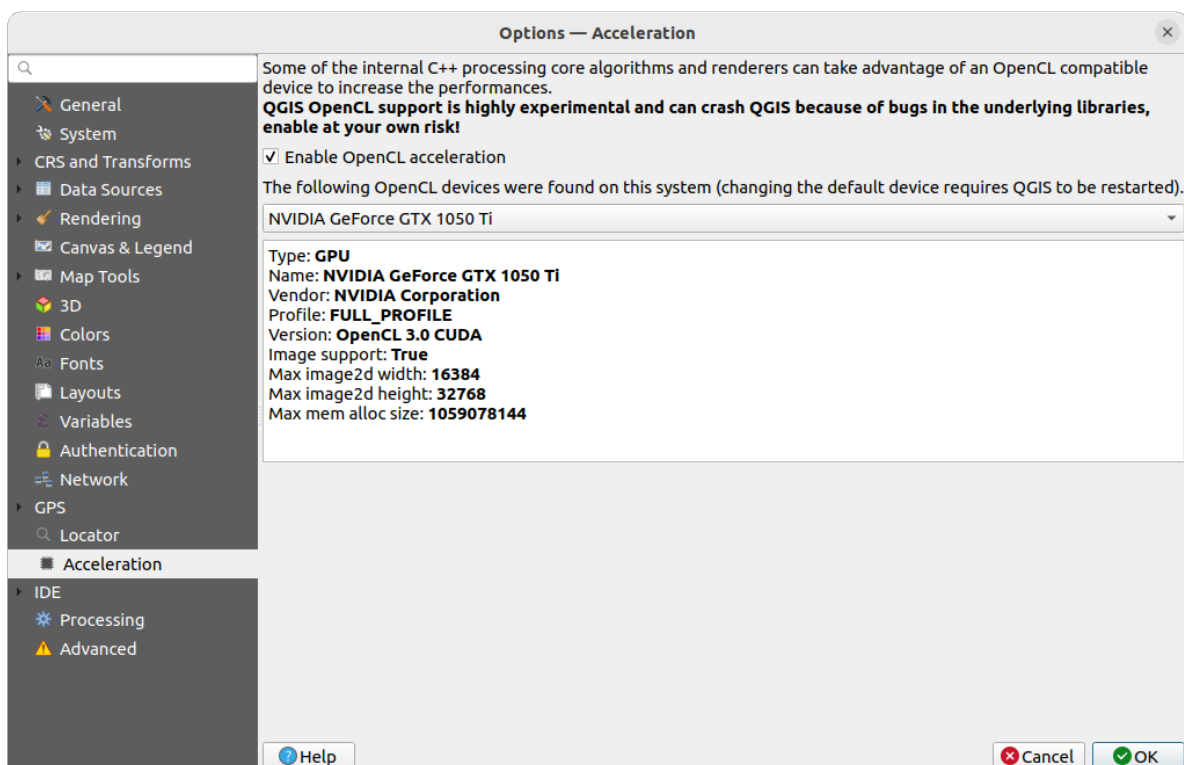



Abb. 9.26: Acceleration settings

Depending on your hardware and software, you may have to install additional libraries to enable OpenCL acceleration.



## 9.1.20 IDE settings

### Code Editor settings

In the  *Code Editor* tab, you can control the appearance and behaviour of code editor widgets (Python interactive console and editor, expression widget and function editor, ...).

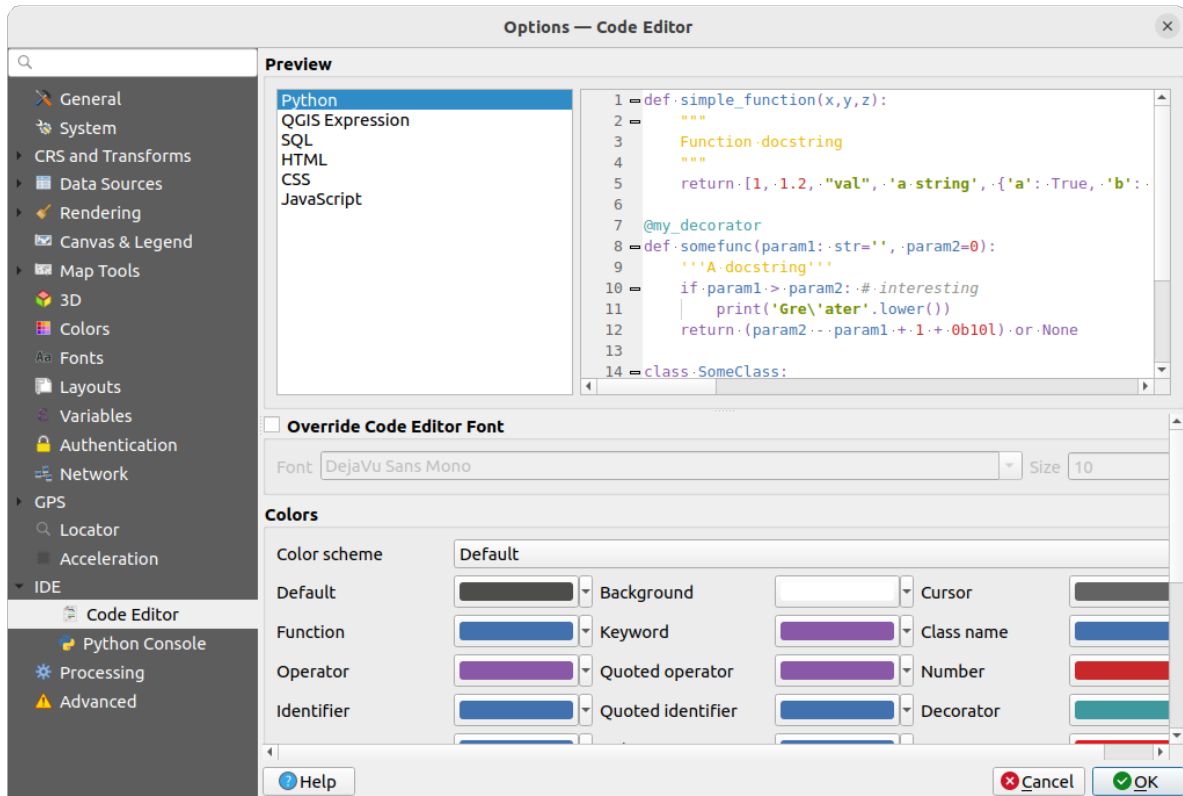




Abb. 9.27: Code Editor settings

At the top of the dialog, a widget provides a live preview of the current settings, in various coding languages (Python, QGIS expression, HTML, SQL, JavaScript). A convenient way to adjust settings.

- Check ☐ *Override Code Editor Font* to modify the default *Font* family and *Size*.
- Under the *Colors* group, you can:
  - select a *Color scheme*: predefined settings are Default, Solarized Dark and Solarized Light. A Custom scheme is triggered as soon as you modify a color and can be reset with selecting a predefined scheme.
  - change the *color* of each element in code writing, such as the colors to use for comments, quotes, functions, background, ...

## Python Console settings

The  *Python Console* settings help you manage and control the behavior of the Python editors (*interactive console*, *code editor*, *project macros*, *custom expressions*, ...). It can also be accessed using the  Options... button from:

- the *Python console* toolbar
- the contextual menu of the *Python console* widget
- and the contextual menu of the code editor.

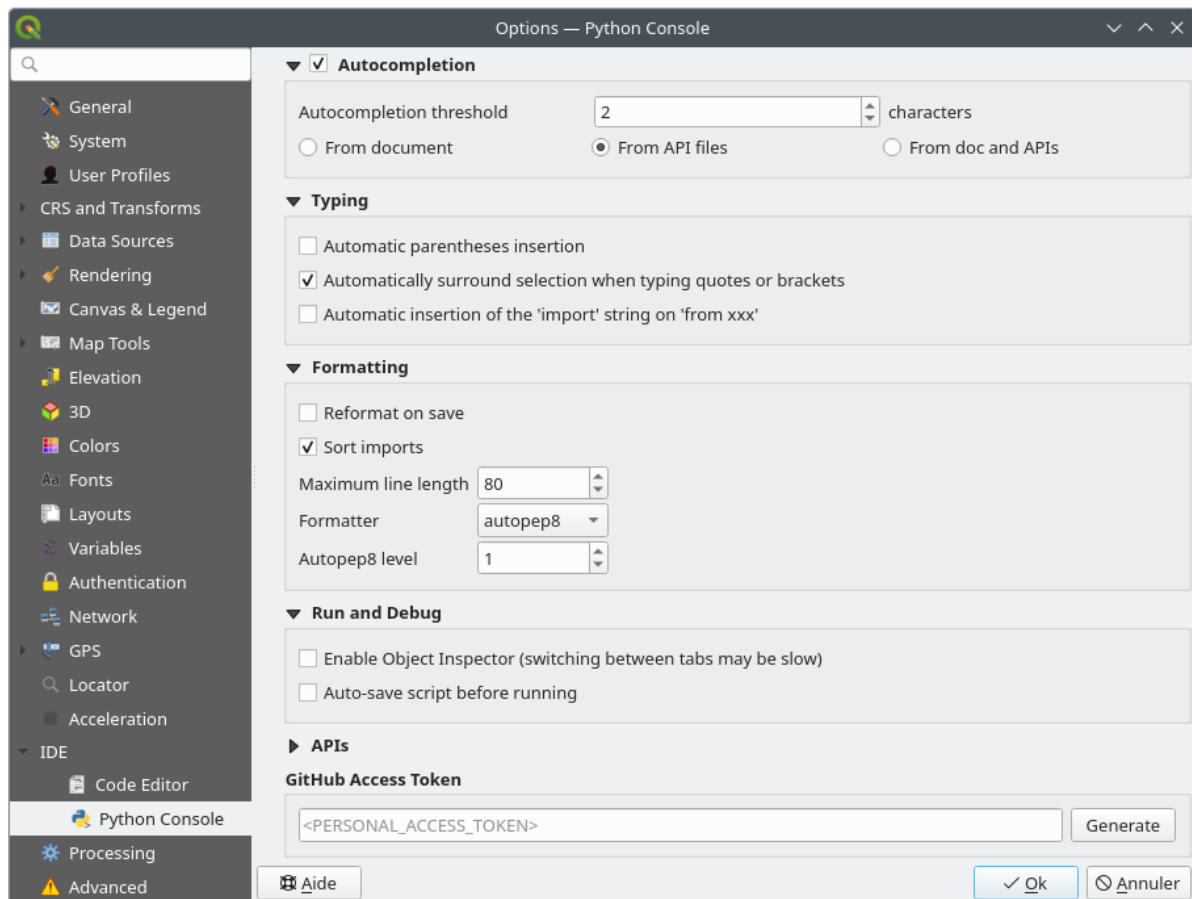





Abb. 9.28: Python Console settings

You can specify:

-  *Autocompletion*: Enables code completion. You can get autocompletion from the current document, the installed API files or both.
  - *Autocompletion threshold*: Sets the threshold for displaying the autocompletion list (in characters)
- under *Typing*
  -  *Automatic parentheses insertion*: When no text is selected, if an opening character (parentheses, quotes, brackets, ...) is entered, inserts the matching closing character just after the cursor. Note that this behavior is disabled if the current cursor is inside a string or comment.
  -  *Automatically surround selection when typing quotes or brackets*: When an opening character is entered, the selected text is enclosed in the opening/closing pair. Selection remains the same, so it is possible to quote a selected word and enclose it in parentheses just by typing " then (.

Special case for multiline selection with quotes and double quotes: selection is enclosed in triple single/double quotes.


- ☐ *Automatic insertion of the ,import' string on ,from xxx':* Enables insertion of ,import' when specifying imports
- under *Formatting*, you can add automated tools to reformat the code you are writing:
  - ☐ *Reformat on save:* formatting is applied just before saving the script
  - ☒ *Sort imports:* sorts ,import' statements using the [isort library](#)
  - *Maximum line length:* controls how the formatter will wrap the lines, and controls the editor ruler
  - *Formatter* - supported tools are *autopep8* and *black*, with dedicated option:
    - \* *Autopep8 level* - more details at [autopep8 advanced usage](#)
    - \* ☐ *Normalize quotes:* replaces all single quotes with double quotes if possible
- under *Run and Debug*
  - ☐ *Enable Object Inspector (switching between tabs may be slow)*
  - ☐ *Auto-save script before running:* Saves the script automatically when executed. This action will store a temporary file (in the temporary system directory) that will be deleted automatically after running.

For *APIs* you can specify:

- ☒ *Using preloaded APIs file:* You can choose if you would like to use the preloaded API files. If this is not checked you can add API files and you can also choose if you would like to use prepared API files (see next option).
- ☐ *Using prepared APIs file:* If checked, the chosen \* .pap file will be used for code completion. To generate a prepared API file you have to load at least one \* .api file and then compile it by clicking the *Compile APIs...* button.

Under *GitHub access token*, you can generate a personal token allowing you to share code snippets from within the Python code editor. More details on [GitHub authentication](#)

## 9.1.21 Processing settings

The  *Processing* tab provides you with general settings of tools and data providers that are used in the QGIS Processing framework. More information at [QGIS Verarbeitung Umgebung](#).

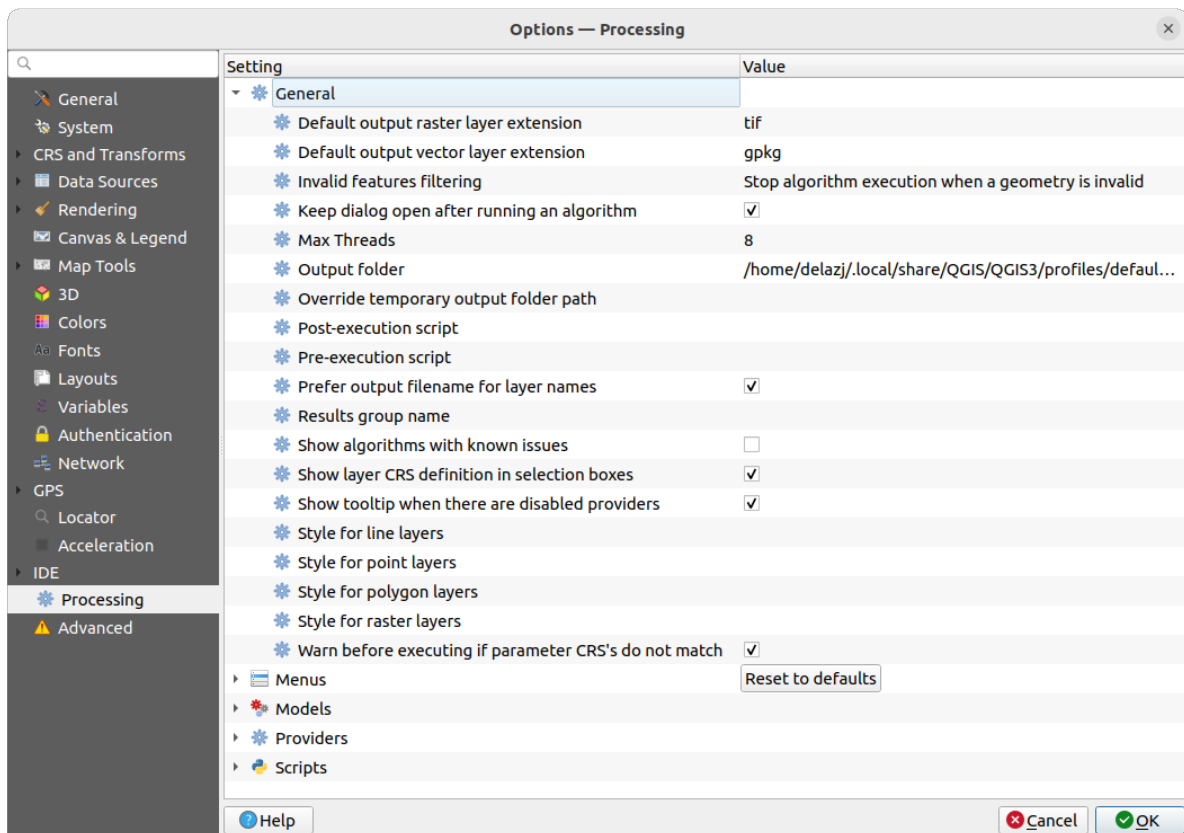


Abb. 9.29: Processing settings

## 9.1.22 Advanced settings

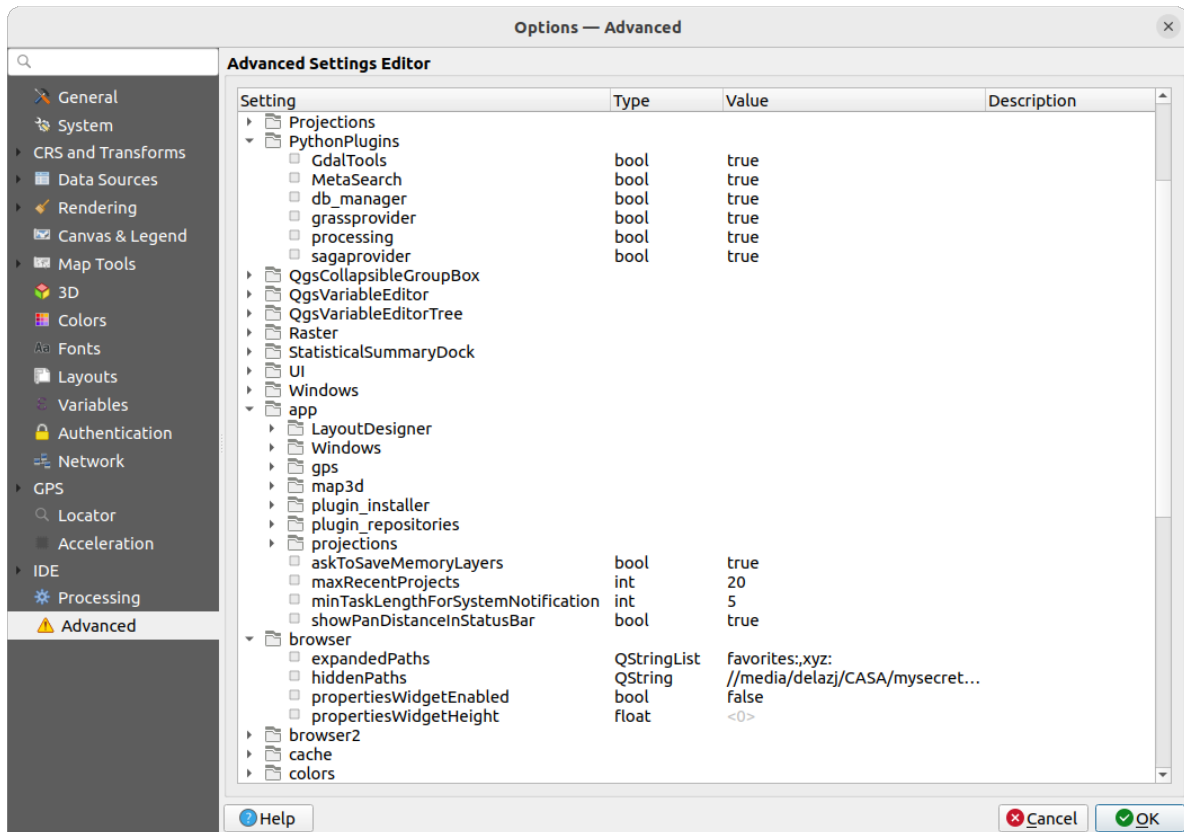


Abb. 9.30: Advanced settings

All the settings related to QGIS (UI, tools, data providers, Processing configurations, default values and paths, plugins options, expressions, geometry checks...) are saved in a `QGIS/QGIS3.ini` file under the active *user profile* directory. Configurations can be shared by copying this file to other installations.

From within QGIS, the *Advanced* tab offers a way to manage these settings through the *Advanced Settings Editor*. After you promise to be careful, the widget is populated with a tree of all the existing settings, and you can edit their value. Right-click over a setting or a group and you can delete it (to add a setting or group, you have to edit the `QGIS3.ini` file). Changes are automatically saved in the `QGIS3.ini` file.

### **Warnung: Avoid using the Advanced tab settings blindly**

Be careful while modifying items in this dialog given that changes are automatically applied. Doing changes without knowledge can break your QGIS installation in various ways.





## 9.2 Working with User Profiles

### 9.2.1 The concept

The *Settings ► User Profiles* menu provides functions to set and access user profiles. A user profile is a unified application configuration that allows to store in a single folder:

- all the *global settings*, including locale, projections, authentication settings, color palettes, shortcuts...
- GUI configurations and *customization*
- grid files and other proj helper files installed for datum transformation
- installed *plugins* and their configurations
- project templates and history of saved project with their image preview
- *processing settings*, logs, scripts, models.

By default, a QGIS installation contains a single user profile named `default`. But you can create as many user profiles as you want:

1. Click the *New profile...* entry.
2. You'll be prompted to provide a profile name, creating a folder of the same name under `~/<UserProfiles>/` where:
  - `~` represents the **HOME** directory, which on  Windows is usually something like `C:\Users\<username>`.
  - and `<UserProfiles>` represents the main profiles folder, i.e.:
    -  `.local/share/QGIS/QGIS3/profiles/`
    -  `%AppData%\Roaming\QGIS\QGIS3\profiles\`
    -  `Library/Application Support/QGIS/QGIS3/profiles/`

The user profile folder can be opened from within QGIS using the *Open Active Profile Folder*.

3. A new instance of QGIS is started, using a clean configuration. You can then set your custom configurations.

If you have more than one profile in your QGIS installation, the name of the active profile is shown in the application title bar between square brackets.

As each user profile contains isolated settings, plugins and history they can be great for different workflows, demos, users of the same machine, or testing settings, etc. And you can switch from one to the other by selecting them in the *Settings ► User Profiles* menu. You can also run QGIS with a specific user profile from the *command line*.

---

**Tipp: Run QGIS under a new user profile to check for bug persistence**

When you encounter weird behavior with some functions in QGIS, create a new user profile and run the commands again. Sometimes, bugs are related to some leftovers in the current user profile and creating a new one may fix them as it restarts QGIS with the new (clean) profile.

---

## 9.2.2 Setting user profile

By default, QGIS opens a new session with the profile of the last closed session. This, among other settings, can be customized in *Settings* ► *Options* ► *User Profiles* tab:

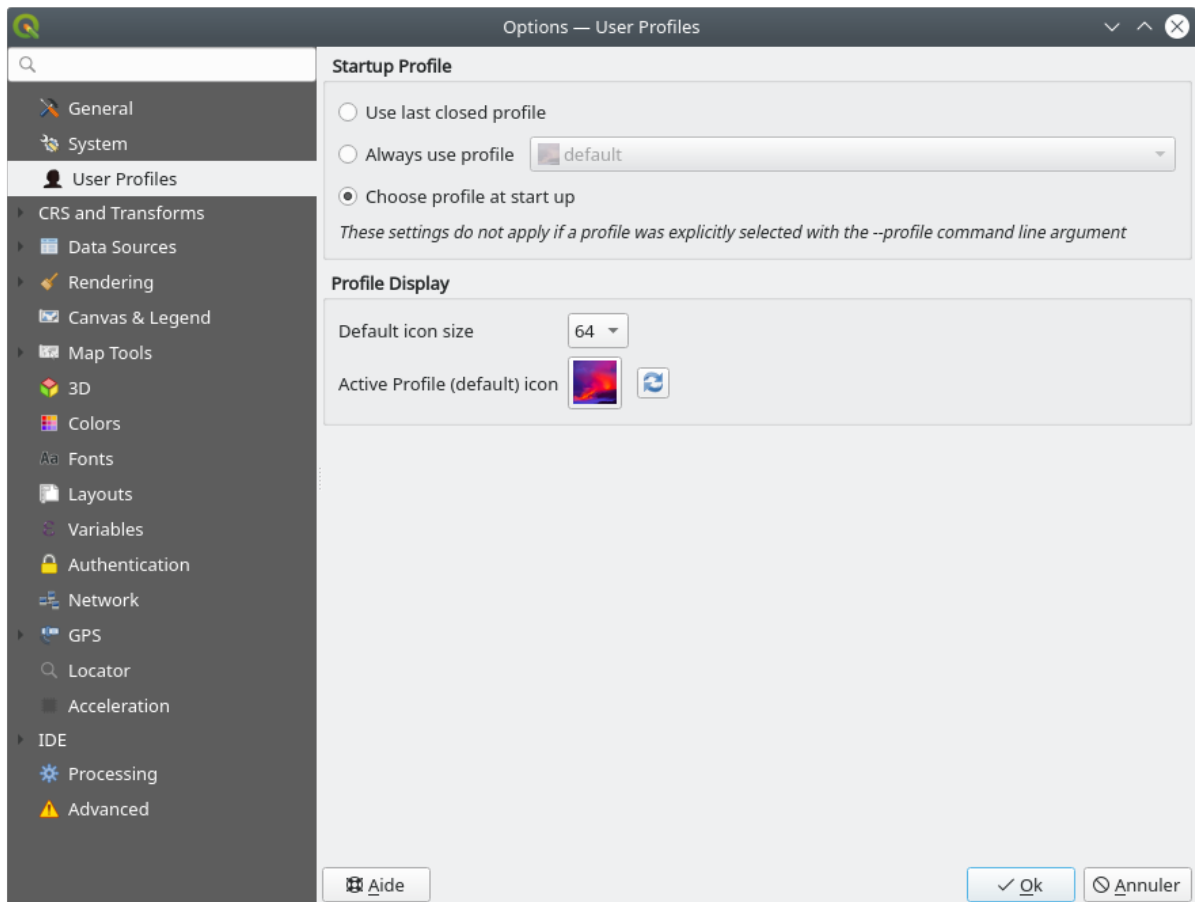





Abb. 9.31: User profiles settings

- *Startup profile*: indicates the user profile to use when starting a QGIS session. It can be:
  - *Use last closed profile*
  - a specific user profile to select from *Always use profile* drop-down menu
  - *Choose profile at start up*: Opens the *User Profile Selector* dialog listing available user profiles. Double-click an entry or select a profile and press *OK* to use that user profile for the starting session. It is also possible to  *Add new profile* to the list: a profile is created from scratch and automatically run with the opening session.
- Under *Profile display*, you can set:
  - The size of icons to use when selecting a profile from the *User Profile Selector* dialog.
  - A specific icon to display next to the current profile in the *Settings* ► *User profiles* menu or the *User Profile Selector* dialog. Press  *Reset profile icon* to remove custom modifications.

## 9.3 Projekteigenschaften



In the properties window for the project under *Project ► Project Properties*, you can set project-specific options. The project-specific options overwrite their equivalent in the *Options* dialog described above.

### 9.3.1 General Properties

In the  *General* tab, the *General settings* let you:



- see the location of the project file
- set the folder for the project home (available in the *Project home* item of the *Browser* panel). The path can be relative to the folder of the project file (type it in) or absolute. The project home can be used for storing data and other content that is useful for the project. Convenient when dataset and project files are not stored at the same place. If not filled, the *Project home* defaults to the project file folder.
- einen Titel für das Projekt hinter dem Projektpfad vergeben
- die Farbe auswählen, die markierte Objekte haben, wenn sie ausgewählt sind
- die Hintergrundfarbe wählen: die Farbe für den Kartenhintergrund
- einstellen, ob der Pfad zu den Layern des Projekts als absolute (voll) oder als relative zu der Projektdatei gespeichert werden soll. Sie bevorzugen wahrscheinlich relative Pfade, wenn Layer und Projektdatei bewegt werden oder geteilt oder wenn Sie von Computern auf verschiedenen Plattformen auf das Projekt zugreifen wollen.
- wählen Sie Artefakte bei der Darstellung von Kartenkacheln vermeiden. Beachten Sie, dass diese Option zu Leistungseinbußen führen kann.
- *Remember attribute tables windows and docks between sessions*: If checked for a project, then any opened attribute tables will be saved into the project and immediately restored when loading that project. This can improve workflows when you constructed a project with a particular set of attribute table configurations for your requirements, and re-setting up these attribute tables is a hassle.

Calculating areas and distances is a common need in GIS. However, these values are really tied to the underlying projection settings. The *Measurements* frame lets you control these parameters. You can indeed choose:

- the *Ellipsoid*, on which distance and area calculations are entirely based; it can be:
  - **None/Planimetric**: returned values are in this case cartesian measurements. This option can be set as default for new projects from the *Settings ►  Options ►  CRS Handling* menu
  - a **Custom** one: you'll need to set values of the semi-major and semi-minor axes.
  - or an existing one from a predefined list (Clarke 1866, Clarke 1880 IGN, New International 1967, WGS 84...).
- the *Units for distance measurements* for length and perimeter, and the *Units for area measurements*. These settings which default to their corresponding *global options* override them in the current project. They are used in:
  - Attributtabellen Feld mit Aktualisierungsleiste
  - Feldrechner Berechnungen
  - *Identify tool* derived length, perimeter and area values
  - *measure dialog*
  - *scale bar decoration*

The *Coordinate and Bearing display* allows you to customize the display of:



- the coordinates shown in the *Coordinates* box on QGIS status bar and in the *Derived* section of the  Identify features tool's results
- the bearing value displayed in the status bar for the map canvas panning direction and by the  Measure bearing tool.

Available parameters are:

- *Display coordinates using* either:
  - Map Units, based on the project CRS
  - Map Geographic (degrees): based on the project CRS if it is of geographic type, otherwise uses its associated geographic CRS. This is helpful e.g. for non-earth celestial bodies.
  - or Custom Projection Units: allows to rely on any CRS you desire for coordinates display
- In the *Coordinate CRS* option, you can view or define the CRS to use depending on your display mode.
- *Coordinate format*: you can configure it as Decimal Degrees, Degrees, Minutes or Degrees, Minutes, Seconds, and whether it should display:
  - ☐ Show directional suffix
  - ☐ Show leading zeros for minutes and seconds
  - ☐ Show leading zeros for degrees
  - ☐ Show trailing zeros
- *Coordinate precision*: the number of decimal places can be automatic (derived from the type of CRS) or set manually
- *Coordinate order*: you can opt to display the coordinates in the native order of the CRS (Default) or switch it to either Easting, Northing (Longitude, Latitude) or Northing, Easting (Latitude, Longitude) order
- *Bearing format* possible values are 0 to 180°, with E/W suffix, -180 to +180° or 0 to 360°. The number of *Decimal places* as well as whether to *Show trailing zeros* can be set.

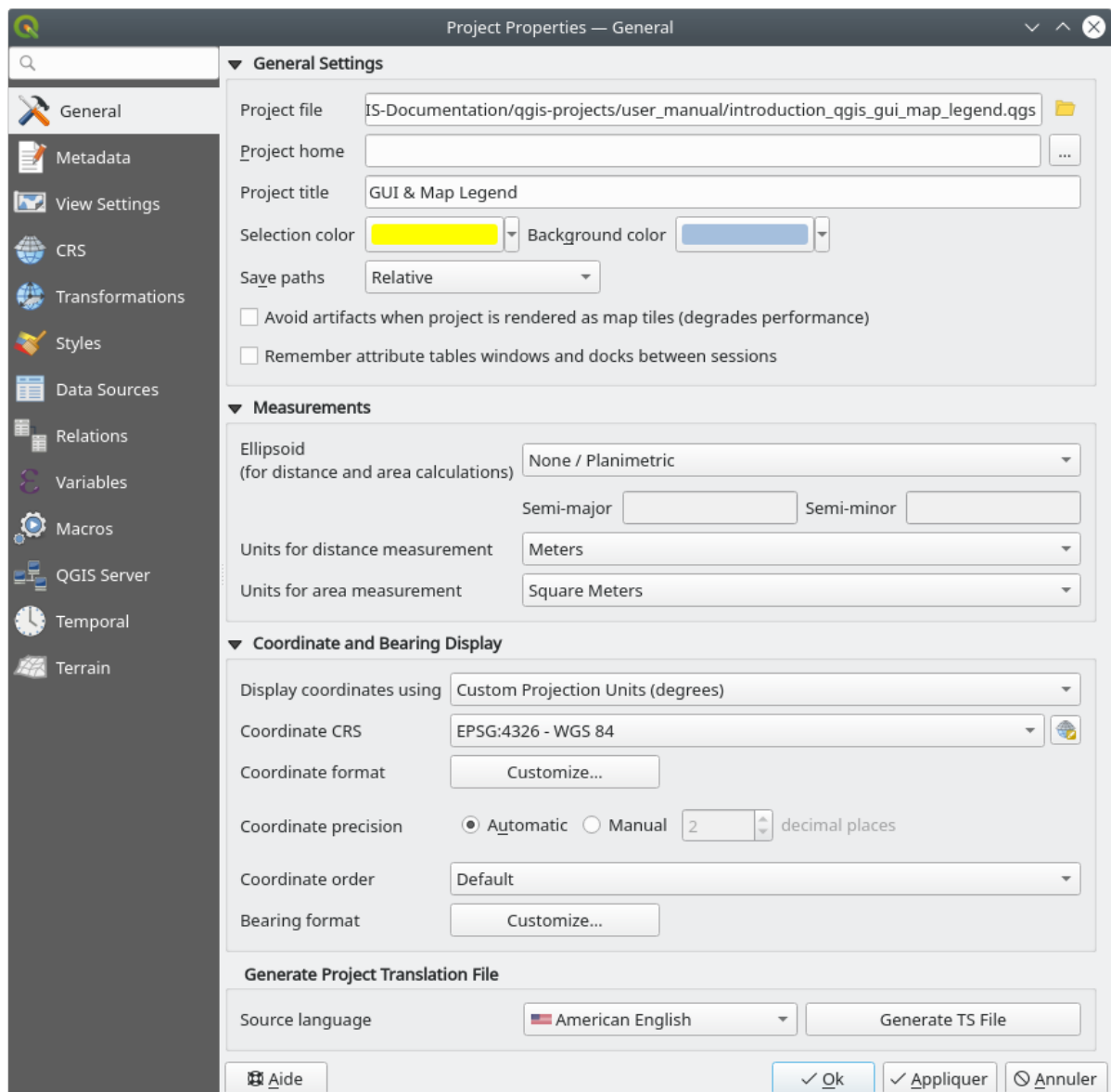



Abb. 9.32: General tab of the Project Properties dialog

## 9.3.2 Eigenschaften Metadaten

The  *Metadata* tab allows detailed metadata to be defined, including (among the others): author, creation date, language, abstracts, categories, keywords, contact details, links, history. There is also a validation functionality that checks if specific fields were filled, anyway this is not enforced. See [vector layer metadata properties](#) for some details.

### 9.3.3 View Settings

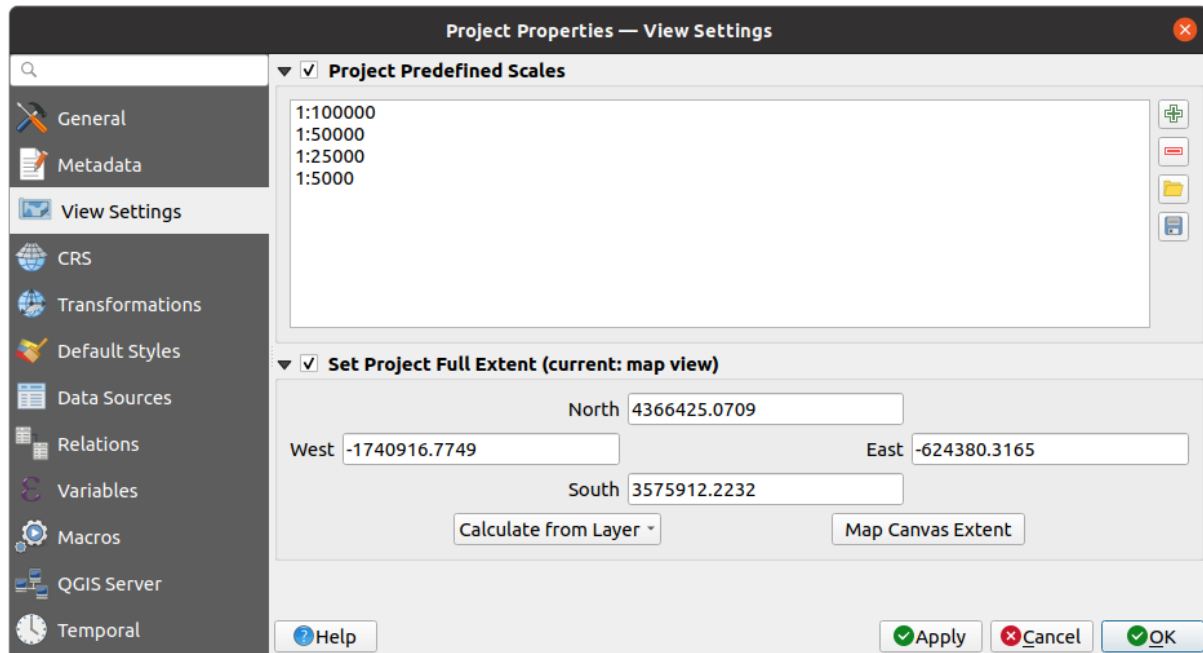





Abb. 9.33: View Settings tab of the Project Properties dialog


The  *View Settings* tab provides means to control the project map canvas. You can:

- set *Project predefined scales*: the list of scales to display in scale-related drop-down widgets, such as the status bar *Scale*, the visibility scales selector or secondary 2D map view settings,... in replacement of the global *predefined scales*.
- *Set Project full Extent*: this extent will be used instead of the extent of all layers when zooming to full map extent (). It's useful when a project contains web layers/national layers/global layers yet the actual area of interest for the project is a smaller geographic area. The project full extent coordinates can be set with the *extent selector* widget.


### 9.3.4 CRS Properties

**Bemerkung:** For more information on how QGIS handles project projection, please read the dedicated section at *Arbeiten mit Projektionen*.


The  *CRS* tab helps you set the coordinate reference system to use in this project. It can be:

-  *No CRS (or unknown/non-Earth projection)*: layers are drawn based on their raw coordinates
- or an existing coordinate reference system that can be *geographic*, *projected* or *user-defined*. Layers added to the project are translated on-the-fly to this CRS in order to overlay them regardless their original CRS.

## 9.3.5 Transformations Properties

The  *Transformations* tab helps you control the layers reprojection settings by configuring the datum transformation preferences to apply in the current project. As usual, these override any corresponding global settings. See *Datum Transformations* for more details.

## 9.3.6 Styles Properties

Under  *Styles* tab, you can configure symbols and colors inherent to the project, allowing to safely share the project among different machines.

The *Default Symbols* group lets you control how new layers will be drawn in the project when they do not have an existing `.qml` style defined. You can set *Marker*, *Line*, *Fill* to apply depending on the layer geometry type as well as default *Color Ramp* and *Text Format* (e.g. when enabling labeling). Any of these items can be reset using the *Clear* entry from the corresponding drop-down widget.

In the *Options* group, you can:

- Apply a default *Opacity* to new layers
- ☒ *Assign random colors to symbols*, modifying the symbols fill colors, hence avoiding same rendering for all layers.

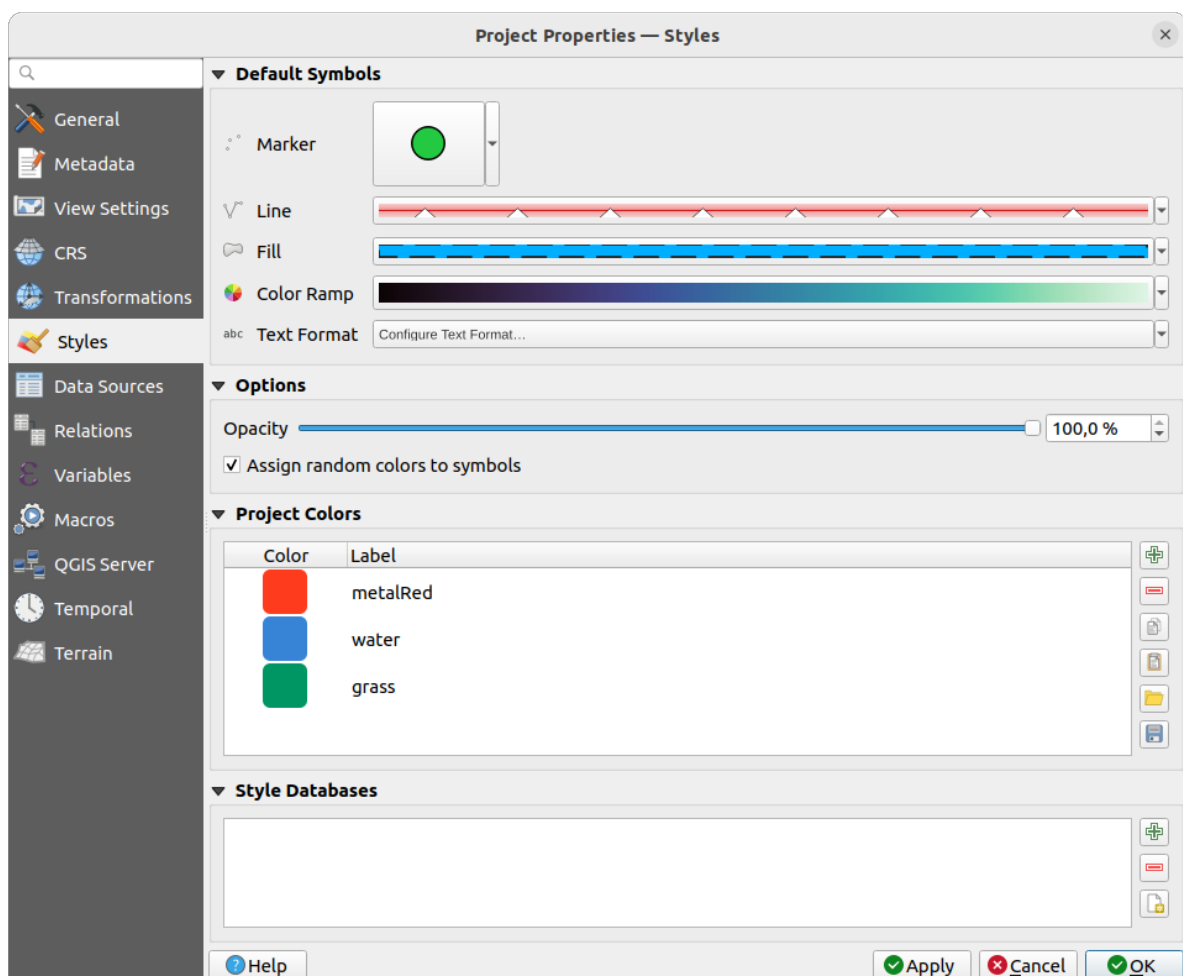








Abb. 9.34: Styles tab

There is also an additional section where you can define specific colors for the running project. Like the *global colors*, you can:

-  Add or  Remove color
-  Copy or  Paste color
-  Import or  Export the set of colors from/to .gpl file.

Double-click a color in the list to tweak or replace it in the *Color Selector* dialog. You can also rename it by double-clicking in the *Label* column.

These colors are identified as *Project colors* and listed as part of *color widgets*.

---

**Tip:** Use project colors to quickly assign and update color widgets

Project colors can be referred to using their label and the color widgets they are used in are bound to them. This means that instead of repeatedly setting the same color for many properties and, to avoid a cumbersome update you can:



1. Define the color as a project color
  2. Click the *data defined override widget* next to the color property you want to set
  3. Hover over the *Color* menu and select the project color. The property is then assigned the expression `project_color('color_label')` and the color widget reflects that color.
  4. Repeat steps 2 and 3 as much as needed
  5. Update the project color once and the change is reflected EVERYWHERE it's in use.
- 


### 9.3.7 Data Sources Properties

In the  *Data Sources* tab, you can:




- *Transaction mode*, defines how edits are sent to the data provider:
  - *Local Edit Buffer*: edits are buffered locally and sent to the provider when toggling layer editing mode or clicking *Save edits*.
  - *Automatic Transaction Groups*: on supported datasources (postgres and geopackage databases) the edit state of all tables that originate from the same database are synchronized and executed in a server side transaction. Also, instead of buffering edit changes locally, they are directly sent to a transaction in the database which gets committed when toggling layer editing mode or clicking *Save edits*.
  - *Buffered Transaction Groups*: all editable layers, disregarding from which provider, are toggled synchronously and all edits are saved in a local edit buffer. Saving changes is executed within a single transaction on all layers (per provider).

Note that you can change this option only if no layer is being edited in the project.

-  *Evaluate default values on provider side*: When adding new features in a PostgreSQL table, fields with default value constraint are evaluated and populated at the form opening, and not at the commit moment. This means that instead of an expression like `nextval('serial')`, the field in the *Add Feature* form will display expected value (e.g., 25).
-  *Remember editable layer status between sessions*: makes sure that all layers that are editable in a project will be remembered as such when saving the project, as well as making sure that those layers are immediately made editable whenever the project is restored.
- Configure the *Layers Capabilities*, i.e.:
  - Set (or disable) which layers are *identifiable*, i.e. will respond to the *identify tool*. By default, layers are set queryable.

- Set whether a layer should appear as `read-only`, meaning that it can not be edited by the user, regardless of the data provider's capabilities. Although this is a weak protection, it remains a quick and handy configuration to avoid end-users modifying data when working with file-based layers.
- Define which layers are `searchable`, i.e. could be queried using the *locator widget*. By default, layers are set searchable.
- Define which layers are defined as `required`. Checked layers in this list are protected from inadvertent removal from the project.
- Define which layers are `private`, i.e. hidden from the *Layers* panel. This is meant for accessory layers (basemap, join, lookups for value-relations, most probably aspatial layers, ...) that you still need in a project but you don't want them to pollute the legend tree and other layer selection tools. If set visible, they are still displayed in the map canvas and rendered in the print layout legend. Use the  *Filter legend* ► *Show private layers* option in the *Layers* panel top toolbar to temporarily turned them on for any interaction.

The *Layers Capabilities* table provides some convenient tools to:

- Select multiple cells and press *Toggle Selection* to have them change their checkbox state;
  -  *Show spatial layers only*, filtering out non-spatial layers from the layers list;
  -  *Filter layers...* and quickly find a particular layer to configure.
- Under the *Advanced Settings* group, you can select  *Trust project when data source has no metadata*: To speed up project loading by skipping data checks. Useful in QGIS Server context or in projects with huge database views/materialized views. The extent of layers will be read from the QGIS project file (instead of data sources) and when using the PostgreSQL provider the primary key unicity will not be checked for views and materialized views.

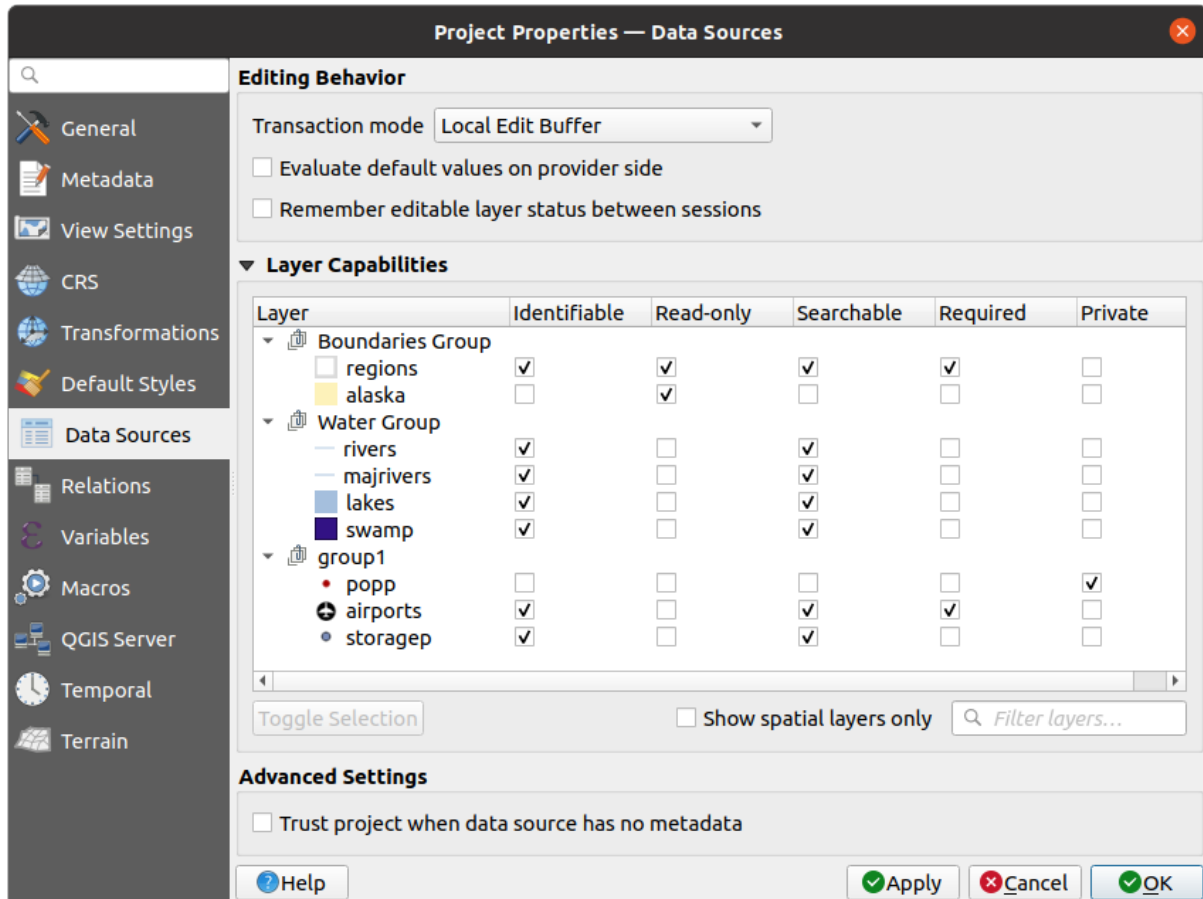



Abb. 9.35: Data Sources tab

### 9.3.8 Relations Properties

The  **Relations** tab is used to define relations between layers. The relations can be of one to one, many to many or polymorphic type. They are defined in the project properties dialog. Once relations exist for a layer, a new user interface element in the form view (e.g. when identifying a feature and opening its form) will list the related entities. This provides a powerful way to express e.g. the inspection history on a length of pipeline or road segment. You can find out more about relations support in section [Setting relations between multiple layers](#).

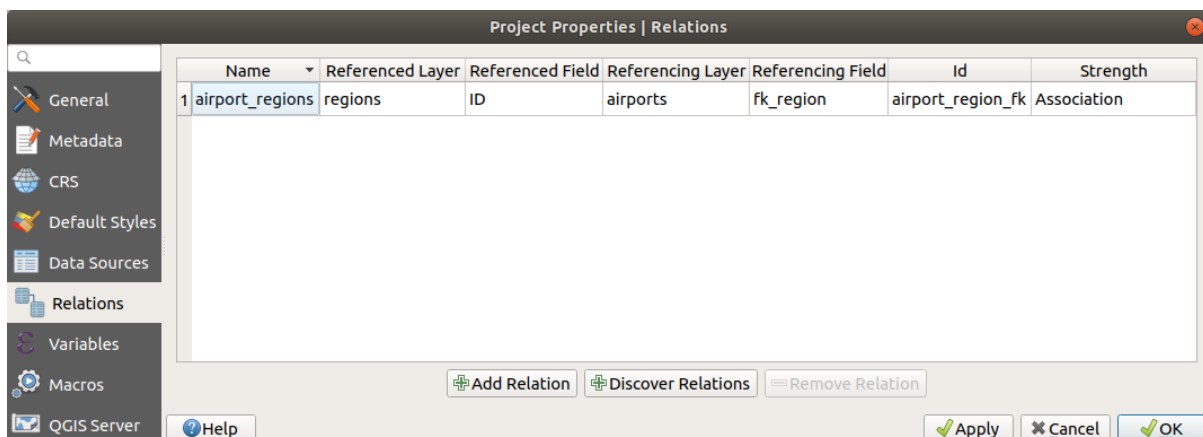






Abb. 9.36: Relations tab

### 9.3.9 Eigenschaften Variablen

The  *Variables* tab lists all the variables available at the project's level (which includes all global variables). Besides, it also allows the user to manage project-level variables. Click the  button to add a new custom project-level variable. Likewise, select a custom project-level variable from the list and click the  button to remove it. More information on variables usage in the General Tools *Storing values in Variables* section.

### 9.3.10 Macros Properties

The  *Macros* tab is used to edit Python macros for projects. Currently, only three macros are available: `openProject()`, `saveProject()` and `closeProject()`.

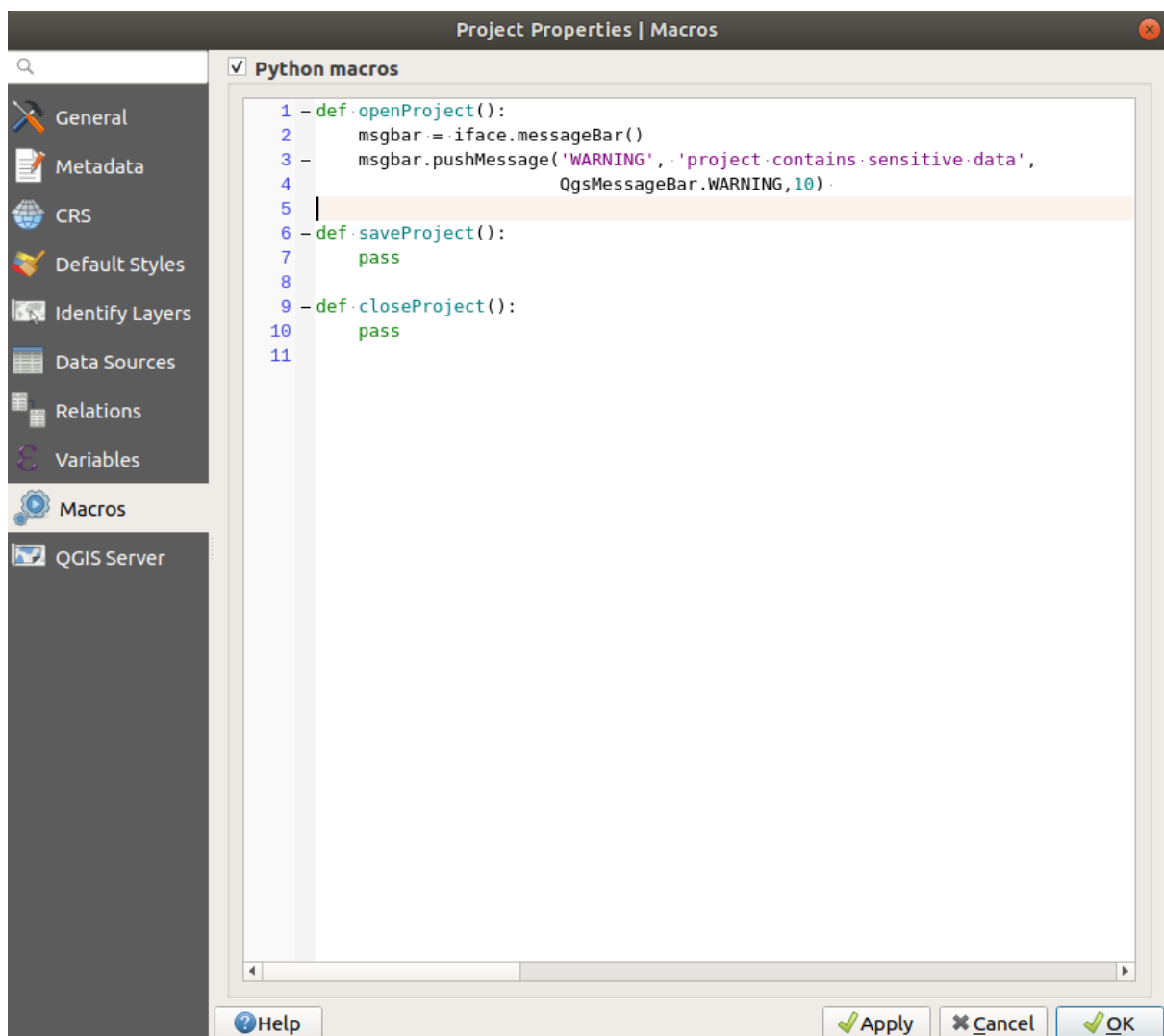

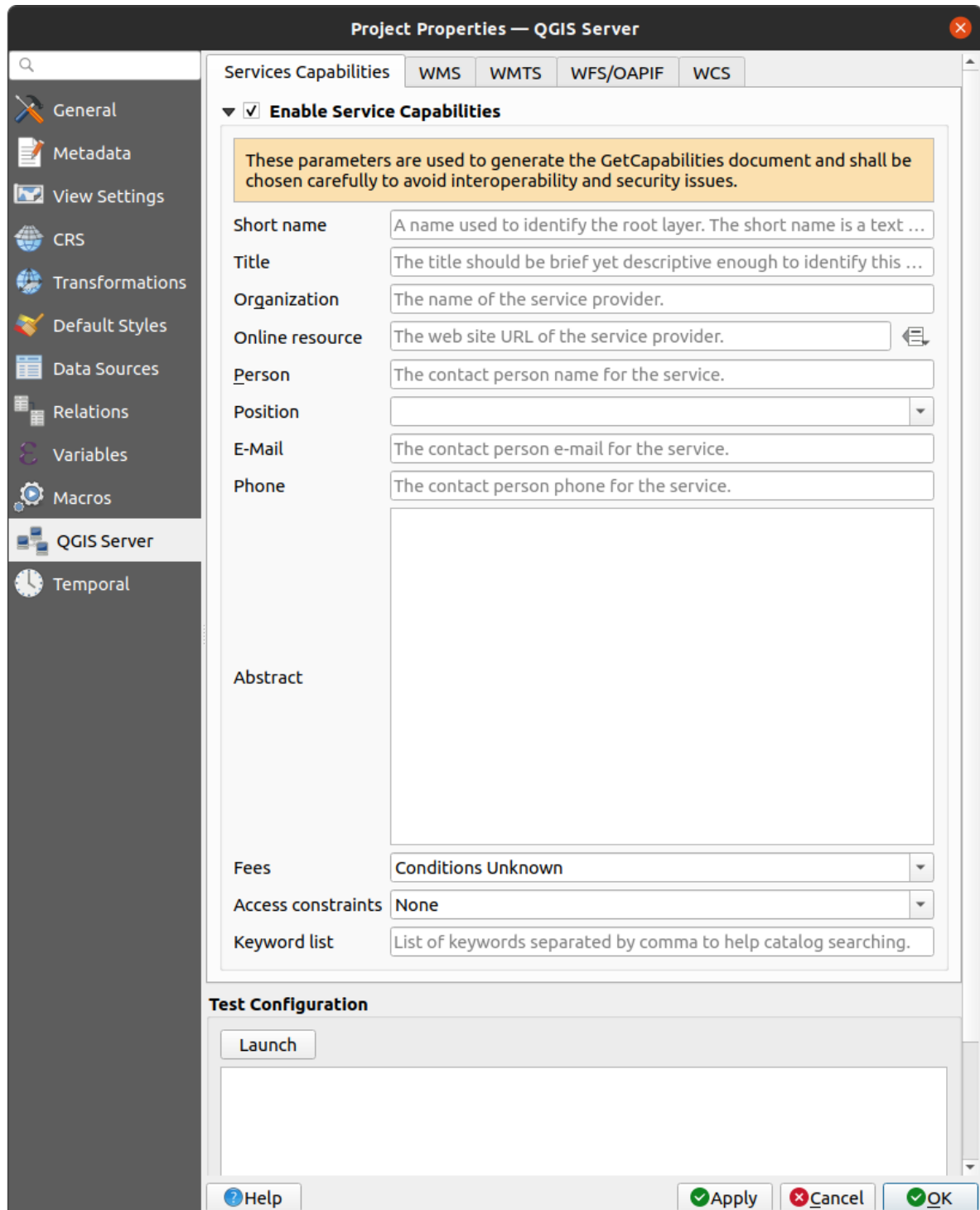


Abb. 9.37: Macro settings



### 9.3.11 Eigenschaften QGIS Server

The  *QGIS Server* tab allows you to configure your project in order to publish it online. Here you can define information about the QGIS Server WMS and WFS capabilities, extent and CRS restrictions. More information available in section [Creating wms from project](#) and subsequent.



**Project Properties — QGIS Server**

Services Capabilities | WMS | WMTS | WFS/OAPIF | WCS

☒ **Enable Service Capabilities**

These parameters are used to generate the GetCapabilities document and shall be chosen carefully to avoid interoperability and security issues.

Short name: A name used to identify the root layer. The short name is a text ...

Title: The title should be brief yet descriptive enough to identify this ...

Organization: The name of the service provider.

Online resource: The web site URL of the service provider.

Person: The contact person name for the service.

Position:

E-Mail: The contact person e-mail for the service.

Phone: The contact person phone for the service.

Abstract:

Fees: Conditions Unknown

Access constraints: None

Keyword list: List of keywords separated by comma to help catalog searching.


**Test Configuration**

Launch

Help Apply Cancel OK

Abb. 9.38: QGIS Server settings

### 9.3.12 Eigenschaften Zeitlich

The  *Temporal* tab is used to set the temporal range of your project, either by using manual *Start date* and *End date* inputs or by calculating it from the current project temporal layers. The project time range can then be used in the *Temporal controller panel* to manage the map canvas *temporal navigation*.

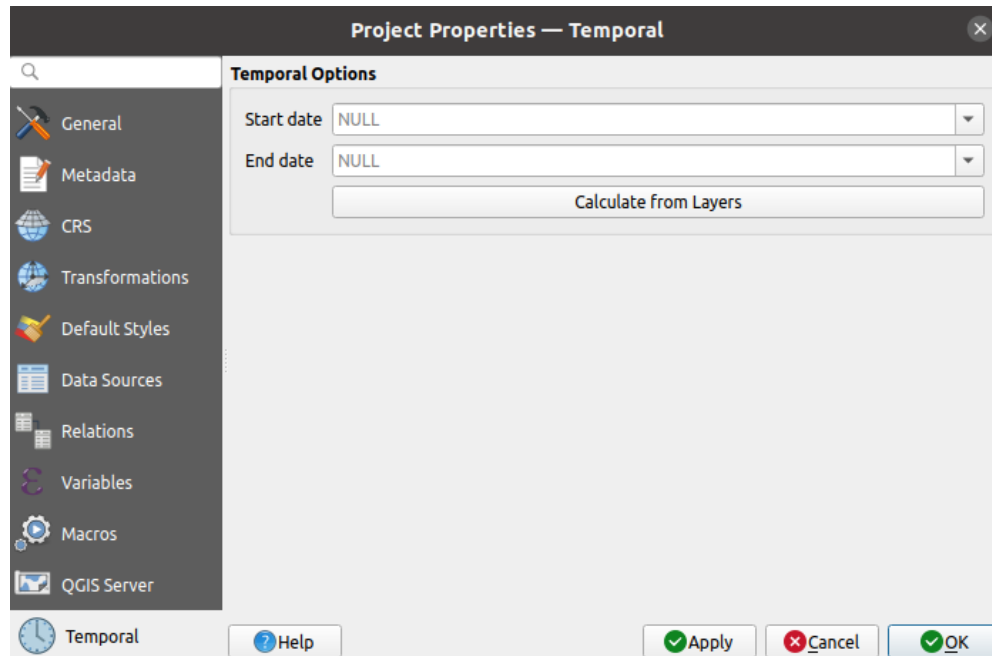



Abb. 9.39: Project Temporal tab

### 9.3.13 Terrain Properties

The  *Terrain* tab helps you configure default settings for the terrain and elevation. When any new *3d map* is created in the project, the map will default to using the same terrain settings as are defined for the project. The project elevation settings will also be respected by the *elevation profile* tool.

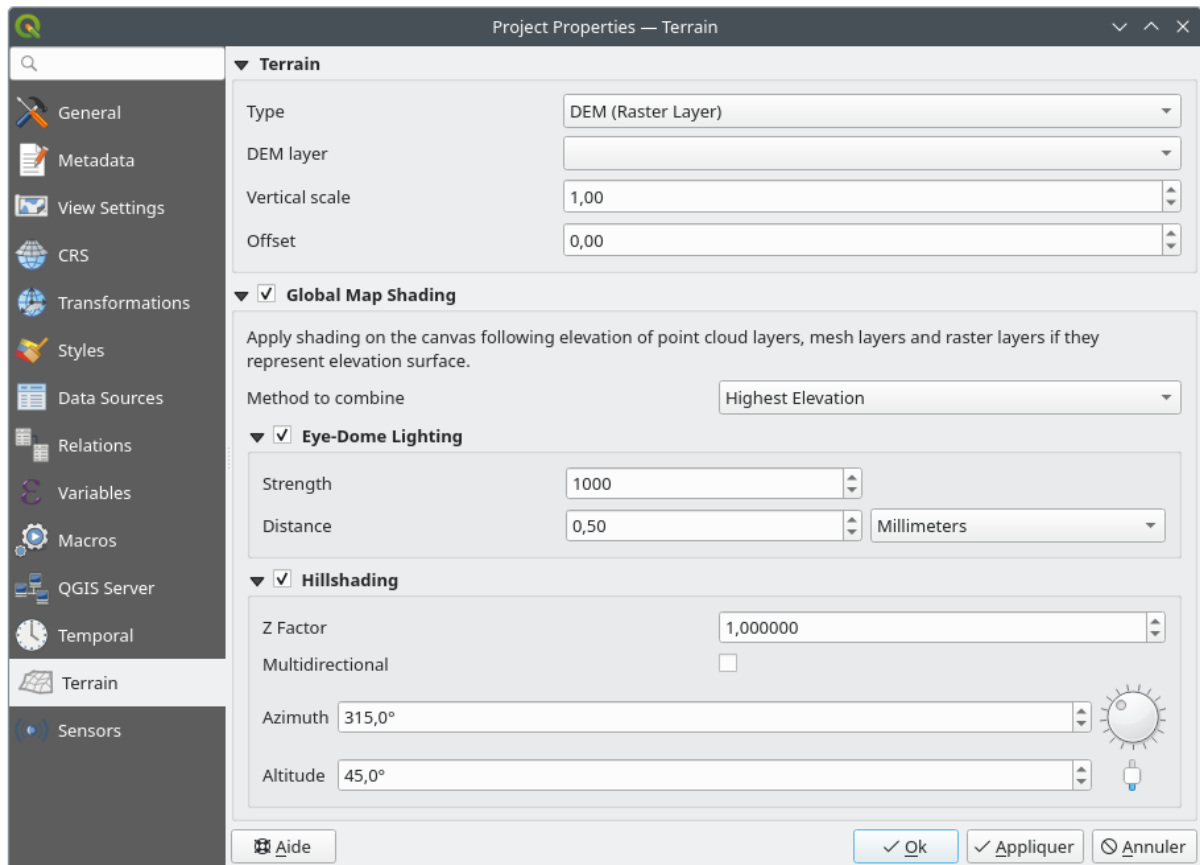


Abb. 9.40: Project Terrain tab

- *Terrain* and elevation options are available for:
  - *Flat terrain* with *Terrain height* setting
  - *DEM (Raster Layer)*: with setting for defining the *Raster layer*, a *Vertical scale* factor to apply to band values and a vertical *Offset*
  - *Mesh*: with setting for defining the *Mesh layer*, a *Vertical scale* factor to apply to vertices Z value and a vertical *Offset*

These settings can be overwritten from the 3D map [configuration dialog](#).

- With ☐ *Global map shading* settings, you apply a global shading effect to the map, based on the elevation of all the active layers that have elevation properties enabled, including:
  - *raster layers*
  - *mesh layers*
  - *point cloud layers*

*Method to combine*: the elevation value at any particular position for all the relevant layers are combined, and the considered elevation is chosen depending on one of these methods:

- *Highest elevation* value
- *Based on layer's order*: the elevation on the topmost layer in the layer tree is considered.

Supported shading options are:

- ☒ *Eye-dome lighting*: applies shading effects to the map canvas for a better depth rendering. Following parameters can be controlled:
  - \* *Strength*: increases the contrast, allowing for better depth perception


- \* *Distance*: represents the distance of the used pixels off the center pixel and has the effect of making edges thicker.
- ☐ *Hillshading*, shaping some reliefs on the map using shading (levels of gray):
  - \* *Z Factor*: Scaling factor for the elevation value
  - \* ☐ *Multidirectional*: Specify if multidirectional hillshading is to be used
  - \* *Azimuth*: The azimuth of the light source
  - \* *Altitude*: The elevation angle of the light source

---

**Bemerkung:** A shortcut to *Global map shading* properties is available through the *Layer Styling* panel.

---

### 9.3.14 Sensors Properties

The  *Sensors* tab is used to configure sensors and toggle their connection status. When active, sensors will passively collect data in the background and make available their latest data to expressions and python scripts.

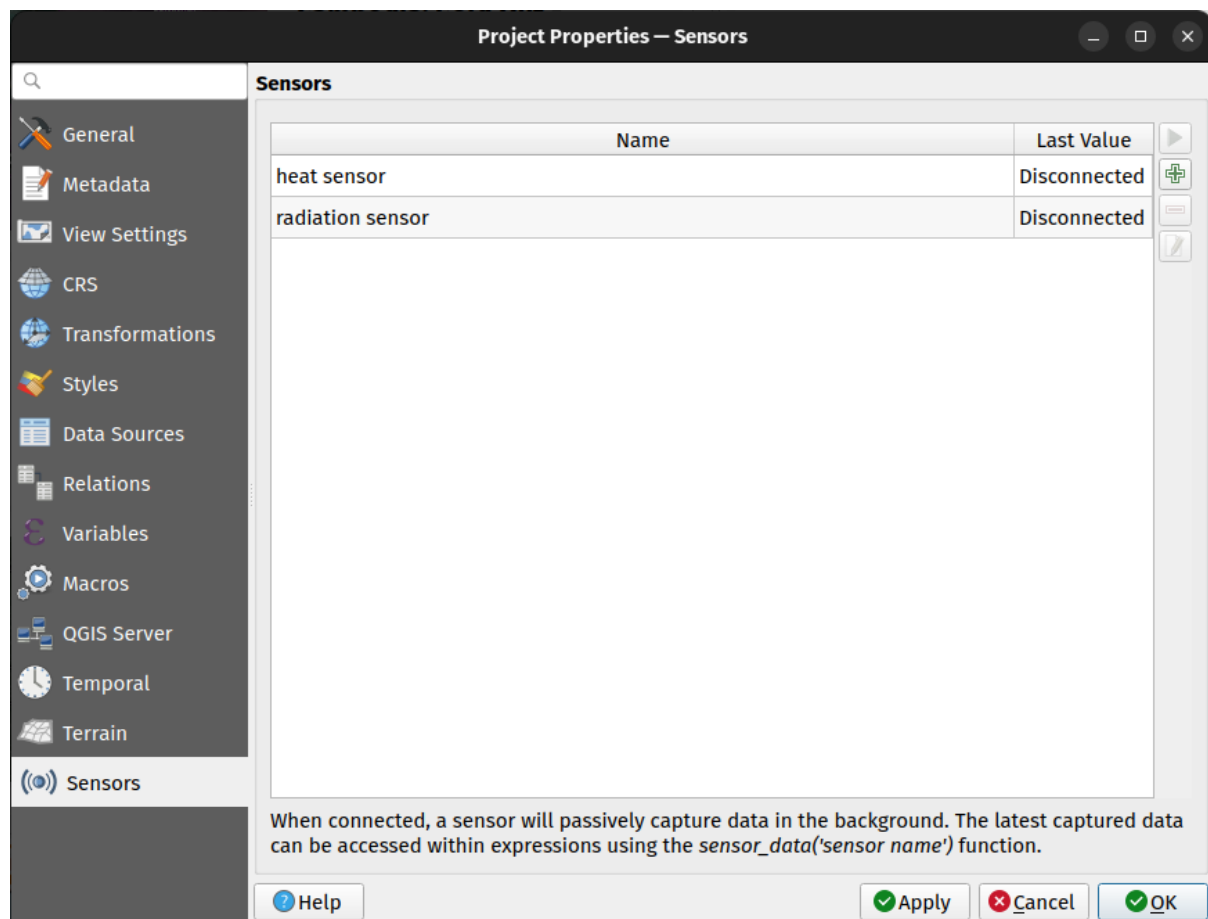



Abb. 9.41: Project Sensors tab

To add a new sensor, click on the  button. A settings sub-panel will open and allow you to configure:

- the *Sensor name*: used to retrieve sensor values in expressions and python scripts;
- the *Sensor type*: TCP, UDP, serial port, etc.; and

- additional type-specific details (e.g. host name and port)

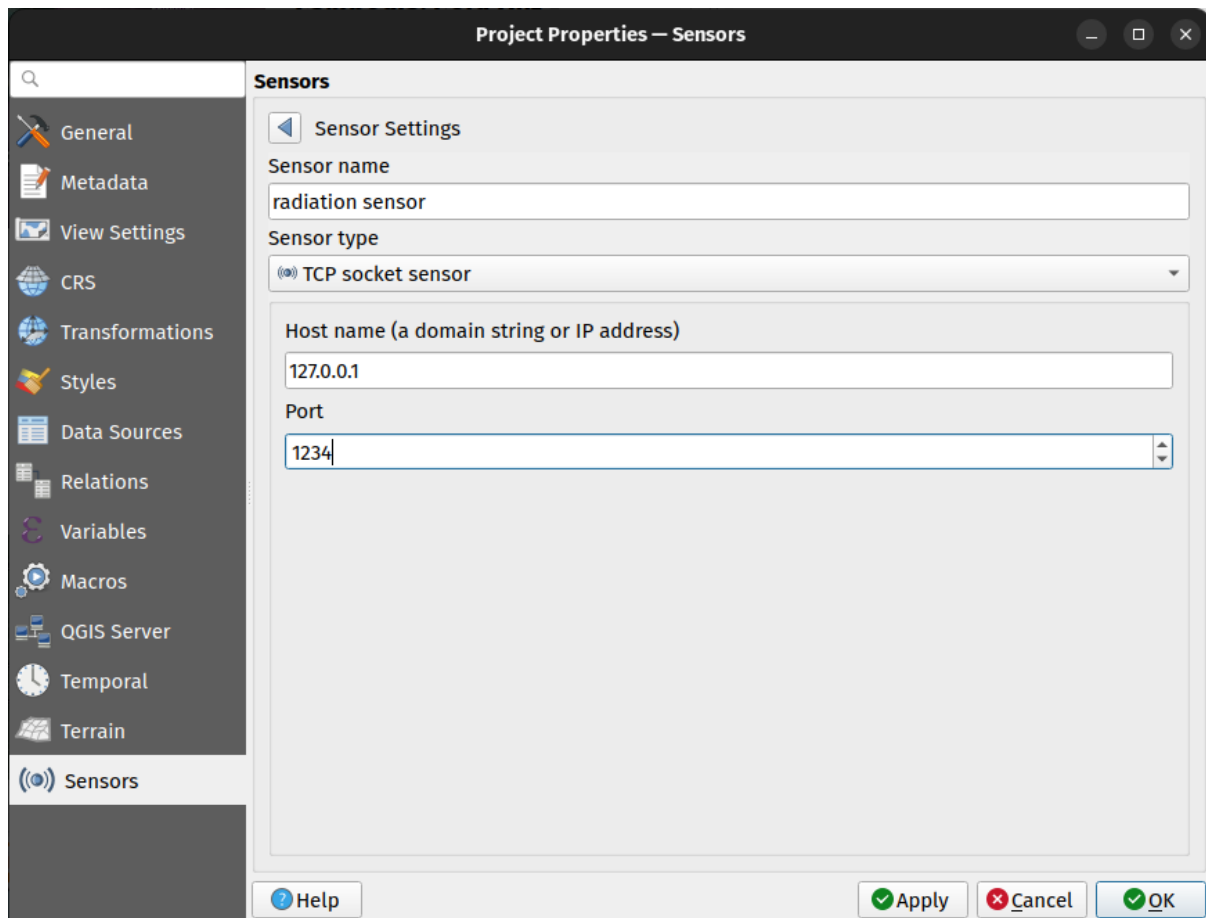



Abb. 9.42: Sensor Settings sub-panel

Once a sensor is configured, you can use the  **Start** button to connect the sensor. Once active, the last collected data will be displayed in the sensors table's *Last value* column.

## 9.4 Anpassung

The *Customization* dialog lets you (de)activate almost every element in the QGIS user interface. This can be very useful if you want to provide your end-users with a ‚light‘ version of QGIS, containing only the icons, menus or panels they need.

---

**Bemerkung:** Bevor Ihre Änderungen übernommen werden, müssen Sie QGIS neu starten.

---

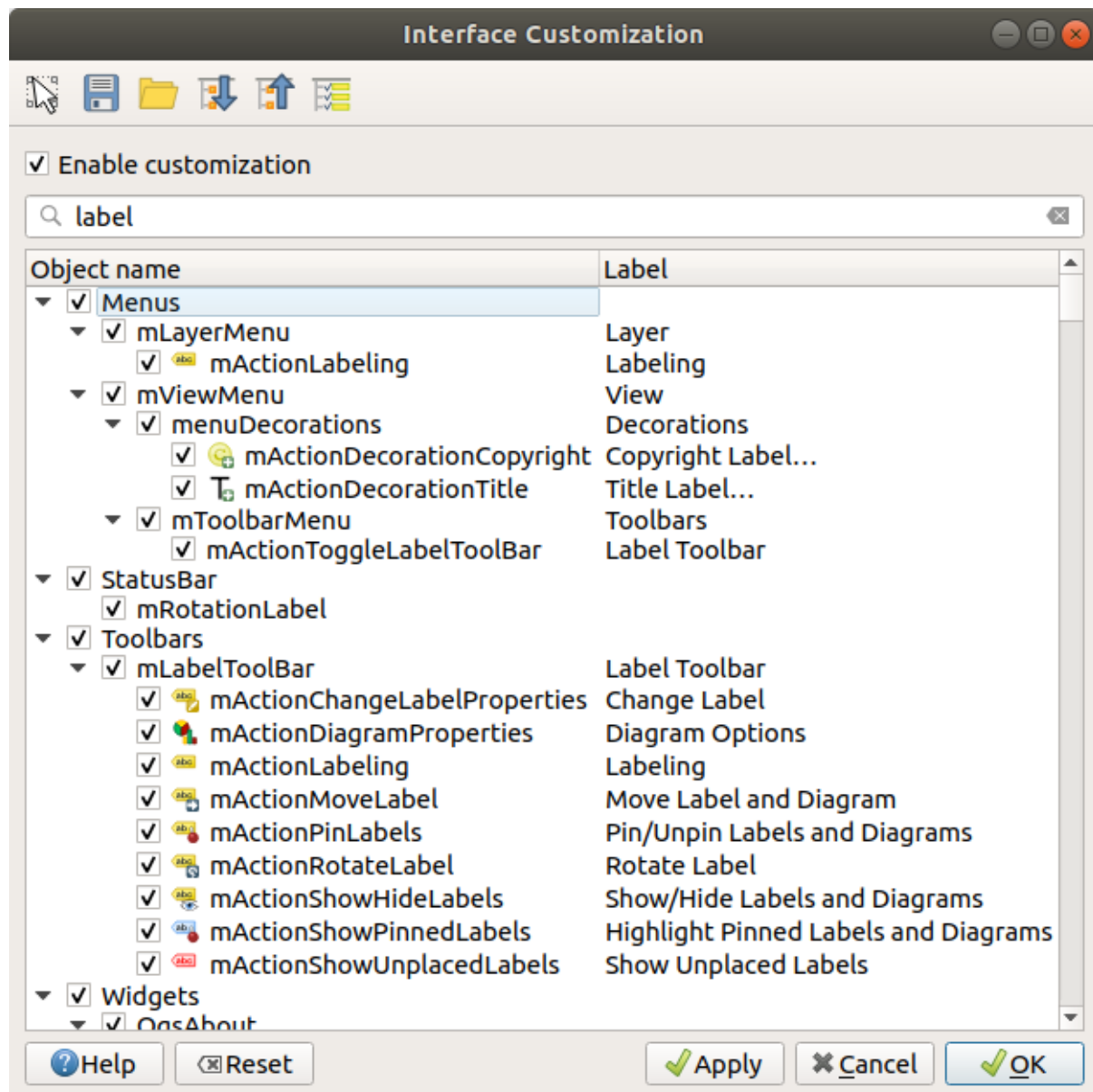




Abb. 9.43: Der Anpassung Dialog



Die  *Anpassung aktivieren* Checkbox anzuklicken ist der erste Schritt, zur QGIS Anpassung. Dies aktiviert die Werkzeugleiste und das Widget-Panel, hier können Sie einige QGIS Elemente unsichtbar schalten.

Die konfigurierbaren Elemente können sein:

- ein **Menü** oder einige ihrer Untermenüs von dem *Menüleiste*
- das komplette **Panel** (siehe *Bedienfelder und Werkzeugkästen*)
- die **Statusleiste** beschrieben in *Statusleiste* oder einige ihrer Elemente
- eine **Werkzeugleiste**: die komplette oder einige ihrer Icons
- oder irgendein **Widget** aus irgendeinem QGIS Dialog: Beschriftung, Knöpfe, Komboboxen...

With  Switch to catching widgets in main application, you can click on an item in QGIS interface that you want to be hidden and QGIS automatically unchecks the corresponding entry in the Customization dialog. You can also use the *Search* box to find items by their name or label.



Once you setup your configuration, click *Apply* or *OK* to validate your changes. This configuration becomes the one used by default by QGIS at the next startup.

Die Modifikationen können auch in einer `.ini` Datei gespeichert werden unter Verwendung des  Als Datei speichern Knopfs. Dies ist eine praktische Möglichkeit, eine gemeinsame QGIS Schnittstelle zwischen mehreren Benutzern zu teilen. Klicken Sie einfach auf  Laden von Datei von dem Zielcomputer aus, um die `.ini` Datei zu importieren. Sie können auch *command line tools* starten und verschiedene Einstellungen für verschiedene Anwendungsfälle speichern.

---

### Tipp: Voreingestelltes QGIS einfach wiederherstellen


Die erste QGIS GUI-Konfiguration kann durch eine der Methoden unten wiederhergestellt werden:

- deaktivieren Sie  *Anpassung aktivieren* im Anpassungsdialog oder klicken Sie  Alle überprüfen
- pressing the *Reset* button in the **Settings** frame under *Settings* ► *Options* menu, *System* tab
- starte QGIS an einer Eingabeaufforderung mit der folgenden Befehlszeile `qgis --customization`
- setting to `false` the value of *UI* ► *Customization* ► *Enabled* variable under *Settings* ► *Options* menu, *Advanced* tab (see the *warning*).

In den meisten Fällen müssen Sie QGIS neu starten, damit die Änderungen angewendet werden.

---

## 9.5 Tastenkürzel

QGIS provides default keyboard shortcuts for many features. You can find them in section *Menüleiste*. Additionally, the menu option *Settings* ►  *Keyboard Shortcuts...* allows you to change the default keyboard shortcuts and add new ones to QGIS features.

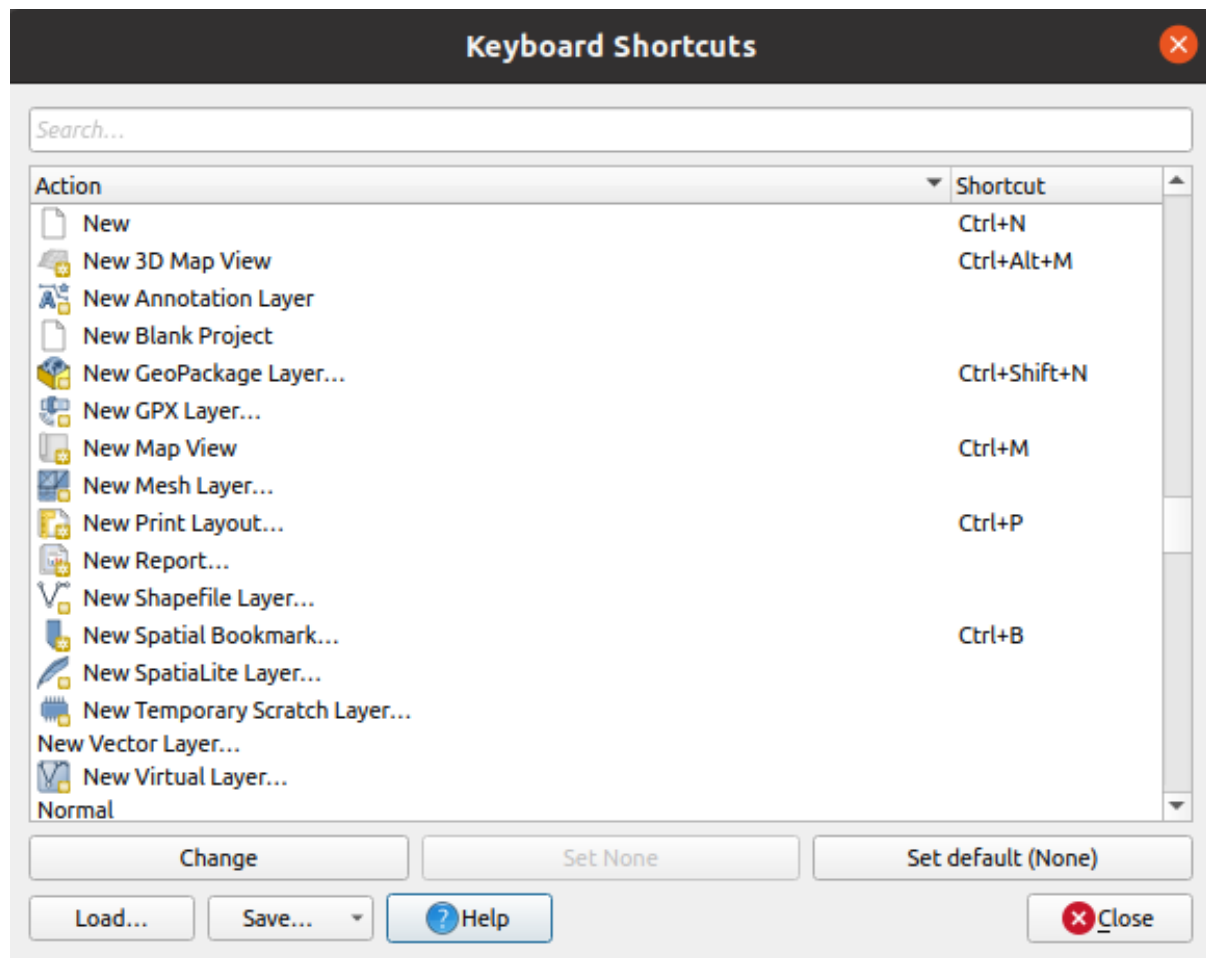


Abb. 9.44: Define shortcut options

Configuration is very simple. Use the search box at the top of the dialog to find a particular action, select it from the list and click on :

- *Change* and press the new combination you want to assign as new shortcut
- *Set None* to clear any assigned shortcut
- or *Set Default* to backup the shortcut to its original and default value.

Proceed as above for any other tools you wish to customize. Once you have finished your configuration, simply *Close* the dialog to have your changes applied. You can also *Save* the changes either as an .XML file with only the User Shortcuts or with all Shortcuts or as an .PDF file with all Shortcuts and *Load* them into another QGIS installation.

## 9.6 Running QGIS with advanced settings

### 9.6.1 Command line and environment variables

We've seen that *launching QGIS* is done as for any application on your OS. QGIS provides command line options for more advanced use cases (in some cases you can use an environment variable instead of the command line option). To get a list of the options, enter `qgis --help` on the command line, which returns:

```
QGIS is a user friendly Open Source Geographic Information System.
Usage: /usr/bin/qgis.bin [OPTION] [FILE]
OPTION:
```

(Fortsetzung auf der nächsten Seite)



(Fortsetzung der vorherigen Seite)

```

[-v, --version]      display version information and exit
[-s, --snapshot filename]  emit snapshot of loaded datasets to given file
[-w, --width width] width of snapshot to emit
[-h, --height height]    height of snapshot to emit
[-l, --lang language]    use language for interface text (changes_
↪existing override)
[-p, --project projectfile] load the given QGIS project
[-e, --extent xmin,ymin,xmax,ymax]  set initial map extent
[-n, --nologo]          hide splash screen
[-V, --noverioncheck]    don't check for new version of QGIS at startup
[-P, --noplugins]       don't restore plugins on startup
[-B, --skipbadlayers]    don't prompt for missing layers
[-C, --nocustomization]  don't apply GUI customization
[-z, --customizationfile path]      use the given ini file as GUI_
↪customization
[-g, --globalsettingsfile path]      use the given ini file as Global_
↪Settings (defaults)
[-a, --authdbdirectory path] use the given directory for authentication_
↪database
[-f, --code path]      run the given python file on load
[-d, --defaultui]      start by resetting user ui settings to default
[--hide-browser]       hide the browser widget
[--dxf-export filename.dxf]    emit dxf output of loaded datasets to_
↪given file
[--dxf-extent xmin,ymin,xmax,ymax]    set extent to export to dxf
[--dxf-symbology-mode none|symbolayer|feature] symbology mode for dxf_
↪output
[--dxf-scale-denom scale]          scale for dxf output
[--dxf-encoding encoding]          encoding to use for dxf output
[--dxf-map-theme maptheme]        map theme to use for dxf output
[--take-screenshots output_path]  take screen shots for the user_
↪documentation
[--screenshots-categories categories] specify the categories of_
↪screenshot to be used (see QgsAppScreenShots::Categories).
[--profile name]                load a named profile from the user's profiles_
↪folder.
[-S, --profiles-path path] path to store user profile folders. Will_
↪create profiles inside a {path}\profiles folder
[--version-migration] force the settings migration from older version if_
↪found
[--openclprogramfolder]          path to the folder containing the sources_
↪for OpenCL programs.
[--help]                        this text
[--]                            treat all following arguments as FILES

FILE:
Files specified on the command line can include rasters,
vectors, and QGIS project files (.qgs and .qgz):
1. Rasters - supported formats include GeoTiff, DEM
and others supported by GDAL
2. Vectors - supported formats include ESRI Shapefiles
and others supported by OGR and PostgreSQL layers using
the PostGIS extension

```

### Tipp: Example Using command line arguments

You can start QGIS by specifying one or more data files on the command line. For example, assuming you are in the `qgis_sample_data` directory, you could start QGIS with a vector layer and a raster file set to load on startup using the following command: `qgis ./raster/landcover.img ./gml/lakes.gml`

### **--version**

This option returns QGIS version information.

### **--snapshot**

This option allows you to create a snapshot in PNG format from the current view. This comes in handy when you have many projects and want to generate snapshots from your data, or when you need to create snapshots of the same project with updated data.

Currently, it generates a PNG file with 800x600 pixels. The size can be adjusted using the `--width` and `--height` arguments. The filename can be added after `--snapshot`. For example:

```
qgis --snapshot my_image.png --width 1000 --height 600 --project my_project.qgs
```

### **--width**

This option returns the width of the snapshot to be emitted (used with `--snapshot`).

### **--height**

This option returns the height of the snapshot to be emitted (used with `--snapshot`).

### **--lang**

Based on your locale, QGIS selects the correct localization. If you would like to change your language, you can specify a language code. For example, `qgis --lang it` starts QGIS in Italian localization.

### **--project**

Starting QGIS with an existing project file is also possible. Just add the command line option `--project` followed by your project name and QGIS will open with all layers in the given file loaded.

### **--extent**

To start with a specific map extent use this option. You need to add the bounding box of your extent in the following order separated by a comma:

```
--extent xmin,ymin,xmax,ymax
```

This option probably makes more sense when paired with the `--project` option to open a specific project at the desired extent.

### **--nologo**

This option hides the splash screen when you start QGIS.

### `--noverversioncheck`

Skip searching for a new version of QGIS at startup.

### `--nopugins`

If you have trouble at start-up with plugins, you can avoid loading them at start-up with this option. They will still be available from the Plugins Manager afterwards.

### `--nocustomization`

Using this option, any existing *GUI customization* will not be applied at startup. This means that any hidden buttons, menu items, toolbars, and so on, will show up on QGIS start up. This is not a permanent change. The customization will be applied again if QGIS is launched without this option.

This option is useful for temporarily allowing access to tools that have been removed by customization.

### `--skipbadlayers`

Using this option, you can avoid QGIS prompting the *Handle Unavailable Layers* dialog at startup. The project file is loaded, with missing layers kept as unavailable. More details on the topic at *Behandlung von fehlerhaften Dateipfaden*.

### `--customizationfile`




Using this option, you can define a UI customization file, that will be used at startup.

### `--globalsettingsfile`

The equivalent environment variable is `QGIS_GLOBAL_SETTINGS_FILE`.

Using this option, you can specify the path for a Global Settings file (`.ini`), also known as the Default Settings. The settings in the specified file replace the original inline default ones, but the user profiles' settings will be set on top of those.

QGIS looks for the default global settings file in the following order and only the first found file will be used:

- path specified by the commandline parameter
- path defined by the environment variable
- the AppDataLocation folder, where persistent application data can be stored; it is managed by the user or system administrator and is not touched by installer and does not require any additional setup like passing commandline parameters or settings environment variable. Depending on the OS, it is:
  -  `$HOME/.local/share/QGIS/QGIS3/`
  -  `C:\Users\<username>\%AppData%\Roaming\QGIS\QGIS3\`
  -  `$HOME/Library/Application Support/QGIS/QGIS3/`
- the installation directory, i.e. `your_QGIS_package_path/resources/qgis_global_settings.ini`.

Presently, there's no way to specify a file to write settings to; therefore, you can create a copy of an original settings file, rename, and adapt it.

Setting the `qgis_global_setting.ini` file path to a network shared folder, allows a system administrator to change global settings and defaults in several machines by only editing one file.

### `--authdbdirectory`

This option is similar to `--globalsettingsfile`, but defines the path to the directory where the authentication database will be stored and loaded.

### `--code`

This option can be used to run a given python file directly after QGIS has started.

For example, when you have a python file named `load_alaska.py` with following content:

```
from qgis.utils import iface
raster_file = "/home/gisadmin/Documents/qgis_sample_data/raster/landcover.img"
layer_name = "Alaska"
iface.addRasterLayer(raster_file, layer_name)
```

Assuming you are in the directory where the file `load_alaska.py` is located, you can start QGIS, load the raster file `landcover.img` and give the layer the name 'Alaska' using the following command:

```
qgis --code load_alaska.py
```

### `--defaultui`

On load, **permanently resets** the user interface (UI) to the default settings. This option will restore the panels and toolbars visibility, position, and size. Unless it's changed again, the default UI settings will be used in the following sessions.

Notice that this option doesn't have any effect on *GUI customization*. Items hidden by GUI customization (e.g. the status bar) will remain hidden even using the `--defaultui` option. See also the `--nocustomization` option.

### `--hide-browser`

On load, hides the *Browser* panel from the user interface. The panel can be enabled by right-clicking a space in the toolbars or using the *View ► Panels (Settings ► Panels* in  Linux KDE).

Unless it's enabled again, the Browser panel will remain hidden in the following sessions.

### `--dxf-*`

These options can be used to export a QGIS project into a DXF file. Several options are available:

- `-dxf-export`: the DXF filename into which to export the layers;
- `-dxf-extent`: the extent of the final DXF file;
- `-dxf-symbology-mode`: several values can be used here: `none` (no symbology), `symbollayer` (Symbol layer symbology), `feature` (feature symbology);
- `-dxf-scale-denom`: the scale denominator of the symbology;
- `-dxf-encoding`: the file encoding;
- `-dxf-map-theme`: choose a *map theme* from the layer tree configuration.

### **--take-screenshots**

Takes screenshots for the user documentation. Can be used together with `--screenshots-categories` to filter which categories/sections of the documentation screenshots should be created (see `QgsAppScreenShots::Categories`).

### **--profile**

Loads QGIS using a specific profile from the user's profile folder. This option takes precedence over *user profile startup setting*.

### **--profiles-path**

With this option, you can choose a path to load and save the profiles (user settings). It creates profiles inside a `{path}\profiles` folder, which includes settings, installed plugins, processing models and scripts, and so on.

This option allows you to, for instance, carry all your plugins and settings in a flash drive, or, for example, share the settings between different computers using a file sharing service.

The equivalent environment variable is `QGIS_CUSTOM_CONFIG_PATH`.

### **--version-migration**

If settings from an older version are found (e.g., the `.qgis2` folder from QGIS 2.18), this option will import them into the default QGIS profile.

### **--openclprogramfolder**

Using this option, you can specify an alternative path for your OpenCL programs. This is useful for developers while testing new versions of the programs without needing to replace the existing ones.

The equivalent environment variable is `QGIS_OPENCL_PROGRAM_FOLDER`.

## **9.6.2 Deploying QGIS within an organization**

If you need to deploy QGIS within an organization with a custom configuration file, first you need to copy/paste the content of the default settings file located in `your_QGIS_package_path/resources/qgis_global_settings.ini`. This file already contains some default sections identified by a block starting with `[]`. We recommend that you keep these defaults values and add your own sections at the bottom of the file. If a section is duplicated in the file, QGIS will take the last one from top to bottom.

You can change `allowVersionCheck=false` to disable the QGIS version check.

If you do not want to display the migration window after a fresh install, you need the following section:

```
[migration]
fileVersion=2
settings=true
```

If you want to add a custom variable in the global scope:

```
[variables]
organisation="Your organization"
```

A lot of settings can be set using the `.INI` file such as WMS/WMTS, PostGIS connections, proxy settings, maptips, default values for algorithms parameters... For example, if you want to configure organization wide default values for some algorithms it would be handy if you could configure it in the prepared ini-file for all users. If the users then open the toolbox to execute an algorithm, the default values will be there. To set the default value e.g. for `GRID_SIZE` parameter for the `native:intersection` algorithm, you could write the following into the `.INI` file:

```
[Processing]
DefaultGuiParam\native%3Aintersection\GRID_SIZE=0.01
```

Not all settings can be prepared from QGIS GUI, but to discover the possibilities of the settings `.INI` file, we suggest that you set the config you would like in QGIS Desktop and then search for it in your `.INI` file located in your profile using a text editor.

Finally, you need to set the environment variable `QGIS_GLOBAL_SETTINGS_FILE` to the path of your customized file.

In addition, you can also deploy files such as Python macros, color palettes, layout templates, project templates... either in the QGIS system directory or in the QGIS user profile.

- Layout templates must be deployed in the `composer_templates` directory.
- Project templates must be deployed in the `project_templates` directory.
- Custom Python macros must be deployed in the `python` directory.

---

## Arbeiten mit Projektionen

---

Ein Koordinatenbezugssystem oder CRS ist ein Verfahren zur Zuordnung von numerischen Koordinaten zu einer Position auf der Erdoberfläche. QGIS unterstützt ca. 7.000 Standard-CRSs mit jeweils unterschiedlichen Anwendungsfällen, Vor- und Nachteilen! Die Auswahl eines geeigneten Referenzsystems für Ihre QGIS-Projekte und -Daten kann eine komplexe Aufgabe sein, aber glücklicherweise hilft Ihnen QGIS dabei, diese Auswahl zu treffen und macht die Arbeit mit verschiedenen CRSs so transparent und genau wie möglich.

### 10.1 Überblick zur Projektionsunterstützung

QGIS has support for approximately 7,000 known CRSs. These standard CRSs are based on those defined by the European Petroleum Search Group (EPSG) and the Institut Geographique National de France (IGNF), and are made available in QGIS through the underlying „Proj“ projection library. Commonly, these standard projections are identified through use of an authority:code combination, where the authority is an organisation name such as „EPSG“ or „IGNF“, and the code is a unique number associated with a specific CRS. For instance, the common WGS 84 latitude/longitude CRS is known by the identifier EPSG:4326, and the web mapping standard CRS is EPSG:3857.

Custom, user-created CRSs are stored in a user CRS database. See section *Eigenes Koordinatenbezugssystem definieren* for information on managing your custom coordinate reference systems.

### 10.2 Layer Coordinate Reference Systems

In order to correctly project data into a specific target CRS, either your data must contain information about its coordinate reference system or you will need to manually assign the correct CRS to the layer. For PostGIS layers, QGIS uses the spatial reference identifier that was specified when that PostGIS layer was created. For data supported by GDAL, QGIS relies on the presence of a recognized means of specifying the CRS. For instance, for the Shapefile format this is a file containing an ESRI Well-Known Text (WKT) representation of the layer's CRS. This projection file has the same base name as the .shp file and a .prj extension. For example, alaska.shp would have a corresponding projection file named alaska.prj.

Whenever a layer is loaded into QGIS, QGIS attempts to automatically determine the correct CRS for that layer. In some cases this is not possible, e.g. when a layer has been provided without retaining this information. You can configure QGIS behavior whenever it cannot automatically determine the correct CRS for a layer:

1. Open *Settings* ►  *Options...* ► *CRS*

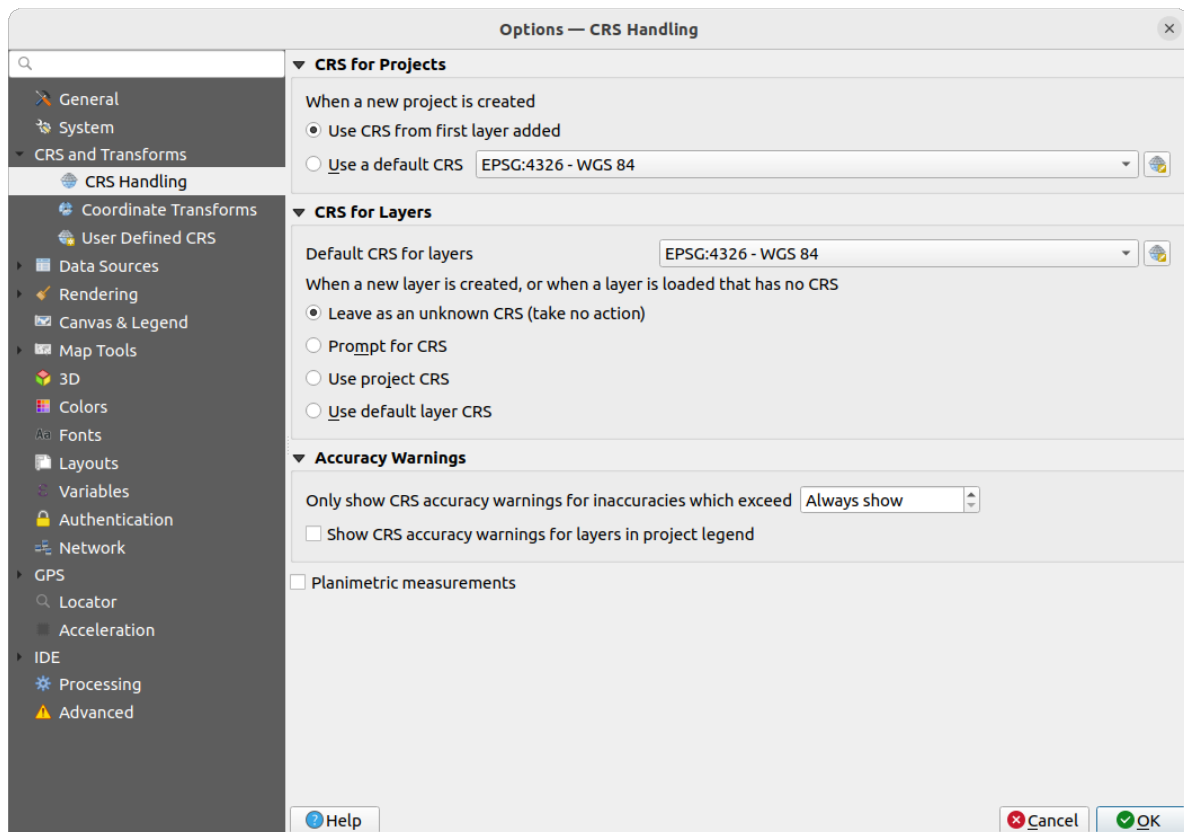







Abb. 10.1: The CRS tab in the QGIS Options Dialog

2. Under the *CRS for layers* group, set the action to do *when a new layer is created, or when a layer is loaded that has no CRS*. One of:

-  *Leave as unknown CRS (take no action)*: there will be no prompt to select a CRS when a layer without CRS is loaded, deferring CRS choice to a later time. Convenient when loading a lot of layers at once. Such layers will be identifiable in the *Layers* panel by the  icon next to them. They'll also be un-referenced, with coordinates from the layer treated as purely numerical, non-earth values, i.e. the same behavior as all layers get when *a project is set to have no CRS*.
-  *Prompt for CRS*: it will prompt you to manually select the CRS. Selecting the correct choice is crucial, as a wrong choice will place your layer in the wrong position on the Earth's surface! Sometimes, accompanying metadata will describe the correct CRS for a layer, in other cases you will need to contact the original author of the data to determine the correct CRS to use.
-  *KBS des Projektes nutzen*
-  *Use default layer CRS*, as set in the *Default CRS for layers* combobox above.

**Tip:** To assign the same CRS to multiple layers that have no crs or have a wrong one in one operation:

1. Select the layers in the *Layers* panel
2. Press **Ctrl+Shift+C**. You could also right-click over one of the selected layers or go to *Layer ► Set CRS of layer(s)*
3. Find and select the right CRS to use
4. And press *OK*. You can confirm that it has been set correctly in the *Source* tab of the layers' properties dialog.



Note that changing the CRS in this setting does not alter the underlying data source in any way, rather it just changes how QGIS interprets the raw coordinates from the layer in the current QGIS project.

## 10.3 Project Coordinate Reference Systems

Every project in QGIS also has an associated Coordinate Reference System. The project CRS determines how data is projected from its underlying raw coordinates to the flat map rendered within your QGIS map canvas.

QGIS supports „on the fly“ CRS transformation for both raster and vector data. This means that regardless of the underlying CRS of particular map layers in your project, they will always be automatically transformed into the common CRS defined for your project. Behind the scenes, QGIS transparently reprojects all layers contained within your project into the project's CRS, so that they will all be rendered in the correct position with respect to each other!

It is important to make an appropriate choice of CRS for your QGIS projects. Choosing an inappropriate CRS can cause your maps to look distorted, and poorly reflect the real-world relative sizes and positions of features. Usually, while working in smaller geographic areas, there will be a number of standard CRSs used within a particular country or administrative area. It's important to research which CRSs are appropriate or standard choices for the area you are mapping, and ensure that your QGIS project follows these standards.

By default, QGIS starts each new project using a global default projection. This default CRS is EPSG:4326 (also known as „WGS 84“), and it is a global latitude/longitude based reference system. This default CRS can be changed via the *CRS for New Projects* setting in the *CRS* tab under *Settings* ► *Options...* (see Abb. 10.1). There is an option to automatically set the project's CRS to match the CRS of the first layer loaded into a new project, or alternatively you can select a different default CRS to use for all newly created projects. This choice will be saved for use in subsequent QGIS sessions.

The project CRS can also be set through the *CRS* tab of the *Project* ► *Properties...* dialog. It will also be shown in the lower-right of the QGIS status bar.

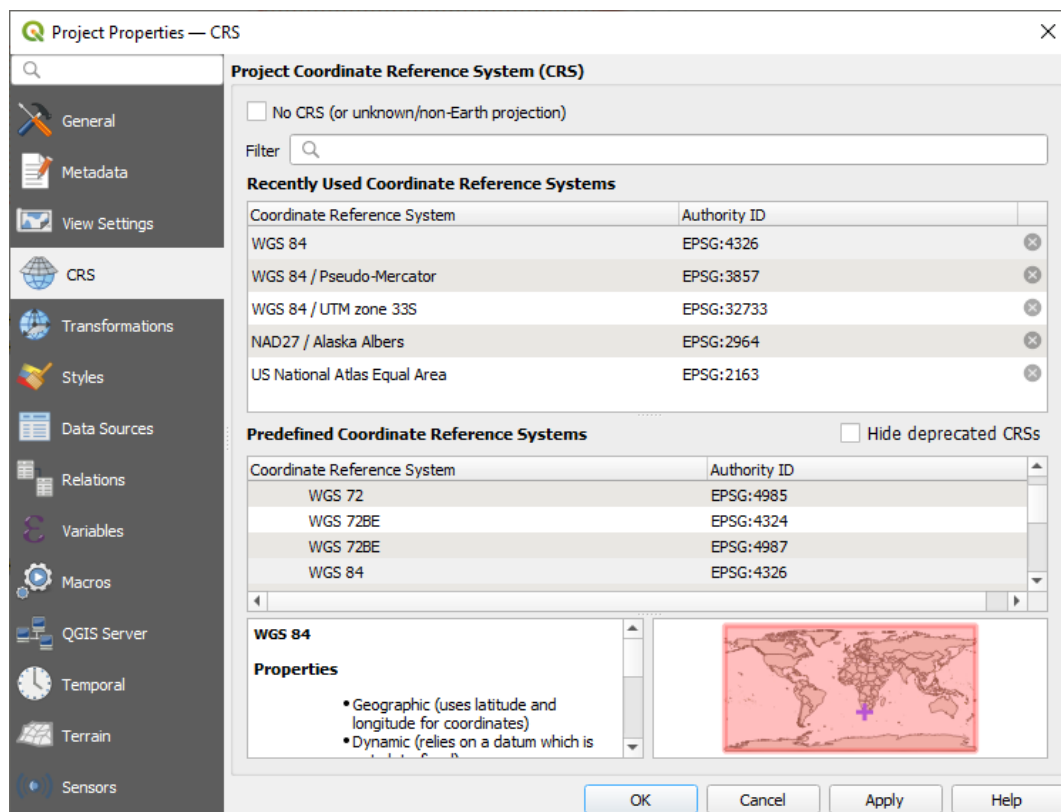



Abb. 10.2: Dialog Projekteigenschaften

Available options are:

-  *No CRS (or unknown/non-Earth projection)*: Checking this setting will disable ALL projection handling within the QGIS project, causing all layers and map coordinates to be treated as simple 2D Cartesian coordinates, with no relation to positions on the Earth's surface. It can be used to guess a layer CRS (based on its raw coordinates or when using QGIS for non earth uses like role-playing game maps, building mapping or microscopic stuff. In this case:
  - No reprojection is done while rendering the layers: features are just drawn using their raw coordinates.
  - The ellipsoid is locked out and forced to `None/Planimetric`.
  - The distance and area units, and the coordinate display are locked out and forced to „unknown units“; all measurements are done in unknown map units, and no conversion is possible.
- or an existing coordinate reference system that can be *geographic*, *projected* or *user-defined*. A preview of the CRS extent on earth is displayed to help you select the appropriate one. Layers added to the project are translated on-the-fly to this CRS in order to overlay them regardless of their original CRS. Use of units and ellipsoid setting are available and make sense and you can perform calculations accordingly.

Whenever you select a new CRS for your QGIS project, the measurement units will automatically be changed in the *General* tab of the *Project properties* dialog (*Project ► Properties...*) to match the selected CRS. For instance, some CRSs define their coordinates in feet instead of meters, so setting your QGIS project to one of these CRSs will also set your project to measure using feet by default.

---

### **Tipp: Setting the project CRS from a layer**

You can assign a CRS to the project using a layer CRS:

1. In the *Layers* panel, right-click on the layer you want to pick the CRS
2. Select *Set project CRS from Layer*.

The project's CRS is redefined using the layer's CRS. Map canvas extent, coordinates display are updated accordingly and all the layers in the project are on-the-fly translated to the new project CRS.

---

## 10.4 Coordinate Reference System Selector

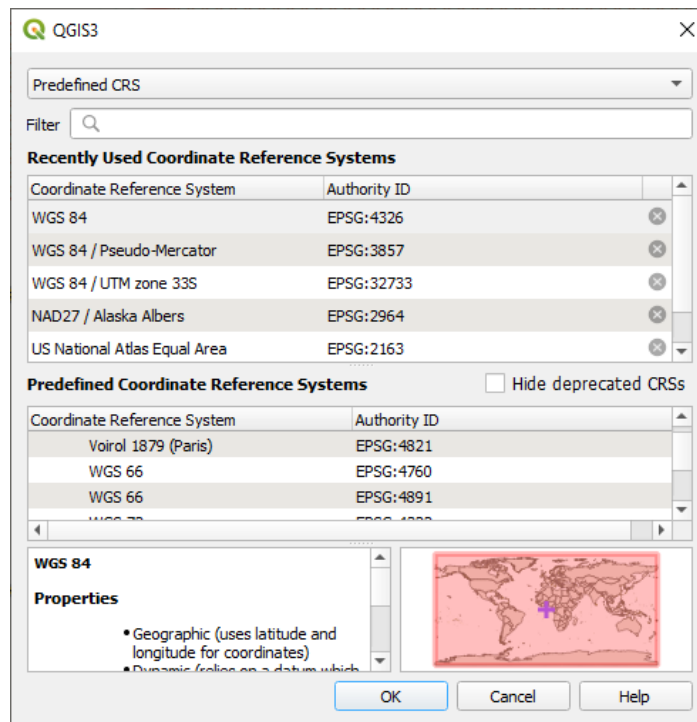





Abb. 10.3: CRS Selector

This dialog helps you assign a Coordinate Reference System to a project or a layer, provided a set of projection databases. Items in the dialog are:


- **Filter:** If you know the EPSG code, the identifier, or the name for a Coordinate Reference System, you can use the search feature to find it. Enter the EPSG code, the identifier or the name.
- **Recently used coordinate reference systems:** If you have certain CRSs that you frequently use in your everyday GIS work, these will be displayed in this list. Click on one of these items to select the associated CRS.
- **Coordinate reference systems of the world:** This is a list of all CRSs supported by QGIS, including Geographic, Projected and Custom coordinate reference systems. To define a CRS, select it from the list by expanding the appropriate node and selecting the CRS. The active CRS is preselected.
- **PROJ text:** This is the CRS string used by the PROJ projection engine. This text is read-only and provided for informational purposes.

If you want to remove a recently used CRS from the list that can be done in several ways:

- Press  Remove from recently used CRS button on the right side.
- Press `Del` after selecting CRS that you want to remove.
- Right-click on used CRS and choose  Remove selected CRS from recently used CRS to remove selected CRS or  Clear all recently used CRS to remove all used CRS.

The CRS selector also shows a rough preview of the geographic area for which a selected CRS is valid for use. Many CRSs are designed only for use in small geographic areas, and you should not use these outside of the area they were designed for. The preview map shades an approximate area of use whenever a CRS is selected from the list. In addition, this preview map also shows an indicator of the current main canvas map extent.

## 10.5 Eigenes Koordinatenbezugssystem definieren

Wenn QGIS nicht das Koordinatenbezugssystem das Sie brauchen zur Verfügung stellt können Sie ein Benutzerdefiniertes Koordinatensystem definieren. Um ein KBS zu definieren wählen Sie  *Benutzerkoordinatenbezugssystem* ... aus dem Menü *Einstellungen*. Benutzerkoordinatenbezugssysteme werden in Ihrer QGIS Benutzerdatenbank gespeichert. Zusätzlich zu Ihrem Benutzerkoordinatensystem enthält diese Datenbank auch Ihre Räumlichen Lesezeichen und andere Benutzerdaten.


Defining a custom CRS in QGIS requires a good understanding of the PROJ projection library. To begin, refer to „Cartographic Projection Procedures for the UNIX Environment - A User's Manual“ by Gerald I. Evenden, U.S. Geological Survey Open-File Report 90-284, 1990 (available at <https://pubs.usgs.gov/of/1990/of90-284/ofr90-284.pdf>).

This manual describes the use of `proj` and related command line utilities. The cartographic parameters used with `proj` are described in the user manual and are the same as those used by QGIS.

Der Dialog *Definition eines Benutzerkoordinatensystems* braucht nur zwei Einträge, um eine eigene Projektion zu definieren:

1. Ein beschreibender Name
2. The cartographic parameters in PROJ or WKT format

To create a new CRS:

1. Click the  Add new CRS button
2. Enter a descriptive name
3. Select the format: it can be *Proj String* or *WKT*
4. Add the CRS *Parameters*.

---

**Bemerkung: Prefer storing the CRS definition in WKT format**

Although both `Proj String` and `WKT` formats are supported, it's highly recommended to store projection definitions in the `WKT` format. Therefore, if the available definition is in the `proj` format, select that format, enter the parameters and then switch to `WKT` format. QGIS will convert the definition to the `WKT` format that you can later save.

---

5. Click *Validate* to test whether the CRS definition is an acceptable projection definition.

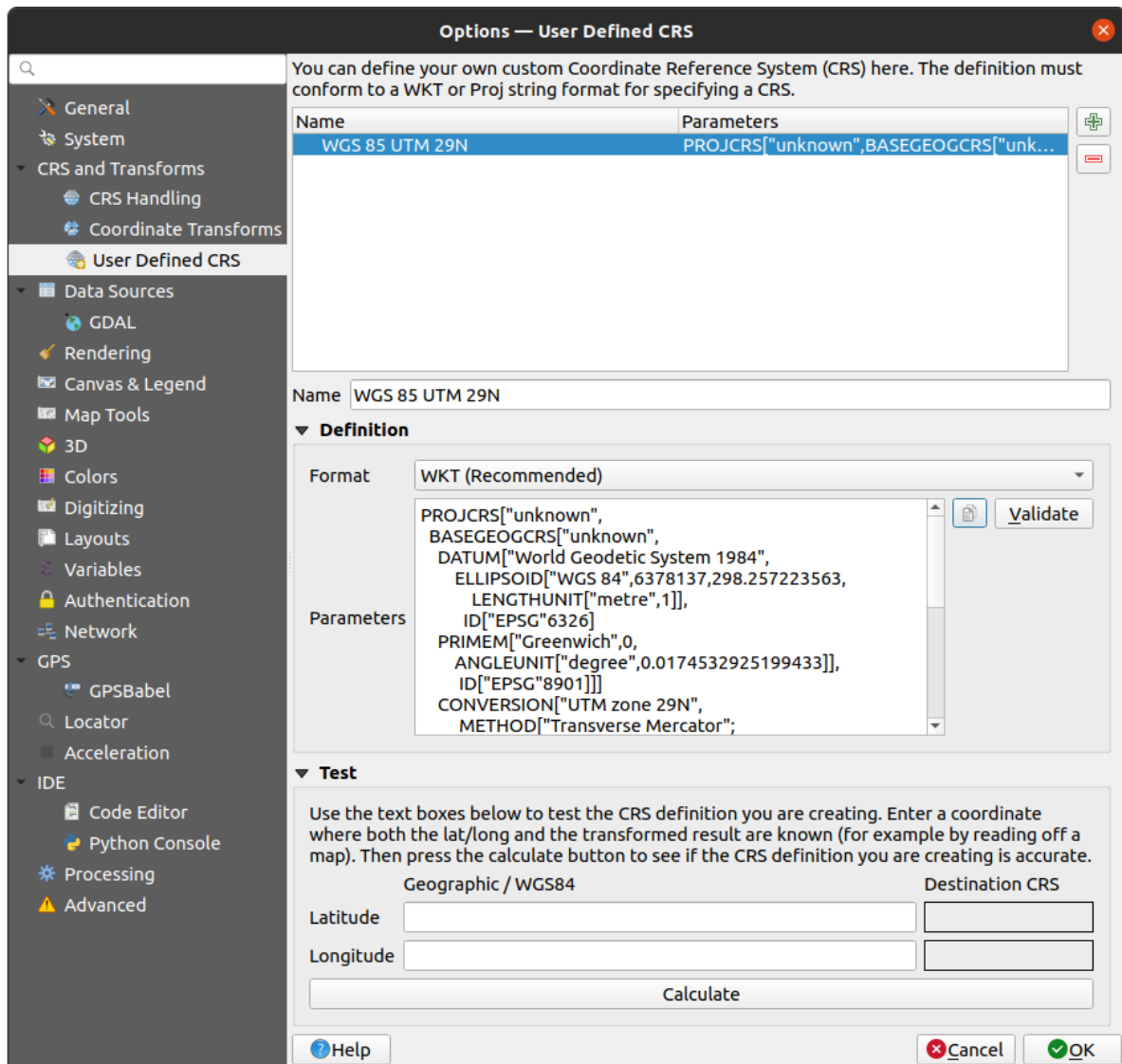


Abb. 10.4: Der Benutzerkoordinatensystem Dialog

You can test your CRS parameters to see if they give sane results. To do this, enter known WGS 84 latitude and longitude values in *North* and *East* fields, respectively. Click on *Calculate*, and compare the results with the known values in your coordinate reference system.

### 10.5.1 Integrate an NTV2-transformation in QGIS

To integrate an NTV2 transformation file in QGIS you need one more step:

1. Place the NTV2 file (.gsb) in the CRS/Proj folder that QGIS uses (e.g. C:\OSGeo4W64\share\proj for windows users)
2. Add **nadgrids** (+nadgrids=nameofthefile.gsb) to the Proj definition in the *Parameters* field of the *Custom Coordinate Reference System Definition* (Settings ► Custom Projections...).




Abb. 10.5: Setting an NTV2 transformation


## 10.6 Datum Transformations

In QGIS, ‘on-the-fly’ CRS transformation is enabled by default, meaning that whenever you use layers with different coordinate systems QGIS transparently reprojects them to the project CRS. For some CRS, there are a number of possible transforms available to reproject to the project’s CRS!


By default, QGIS will attempt to use the most accurate transformation available. However, in some cases this may not be possible, e.g. whenever additional support files are required to use a transformation. Whenever a more accurate transformation is available, but is not currently usable, QGIS will show an informative warning message advising you of the more accurate transformation and how to enable it on your system. Usually, this requires download of an external package of transformation support files, and extracting these to the `proj` folder under your QGIS *user profile* folder.

If desired, QGIS can also prompt you whenever multiple possible transformations can be made between two CRSs, and allow you to make an informed selection of which is the most appropriate transformation to use for your data.

This customization is done in the *Settings* ►  *Options* ► *Transformations* tab menu under the *Default datum transformations* group:

- using  *Ask for datum transformation if several are available*: when more than one appropriate datum transformation exist for a source/destination CRS combination, a dialog will automatically be opened prompting users to choose which of these datum transformations to use for the project. If the *Make default* checkbox is ticked when selecting a transformation from this dialog, then the choice is remembered and automatically applied to any newly created QGIS projects.
- or defining a list of appropriate datum transformations to use as defaults when loading a layer to a project or reprojecting a layer.

Use the  button to open the *Select Datum Transformations* dialog. Then:

1. Choose the *Source CRS* of the layer, using the drop-down menu or the  *Select CRS* widget.
2. Provide the *Destination CRS* in the same way.
3. A list of available transformations from source to destination will be shown in the table. Clicking a row shows details on the settings applied and the corresponding accuracy and area of use of the transformation.

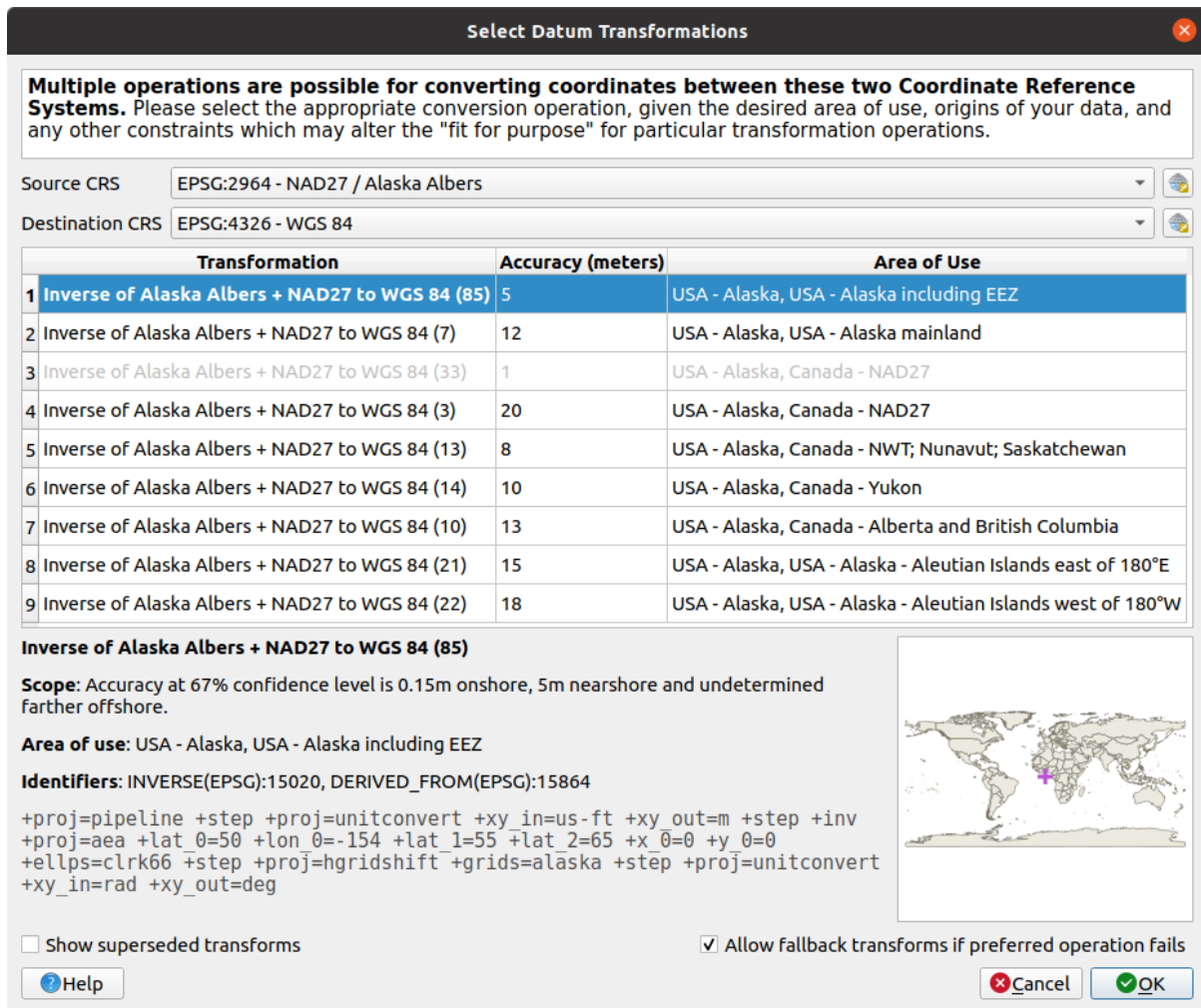




Abb. 10.6: Selecting a preferred default datum transformation

In some cases a transformation may not be available for use on your system. In this case, the transformation will still be shown (greyed) in this list but can not be picked until you install the required package of transformation support. Usually, a button is provided to download and install the corresponding grid, which is then stored under the `proj` folder in the active *user profile* directory.

- Find your preferred transformation and select it
- Set whether you ☒ *Allow fallback transforms if preferred operation fails*
- Click **OK**.

A row is added to the table under *Default Datum Transformations* with information about the *Source CRS*, the *Destination CRS*, the *Operation* applied for the transformation and whether *Allow fallback Transforms* is enabled.

From now, QGIS automatically uses the selected datum transformations for further transformation between these two CRSs until you remove it (  ) from the list or change the entry (  ) in the list.

Datum transformations set in the *Settings* ► *Options* ► *Transformations* tab will be inherited by all new QGIS projects created on the system. Additionally, a particular project may have its own specific set of transformations specified via the *CRS* tab of the *Project properties* dialog (*Project* ► *Properties...*). These settings apply to the current project only.





## Visualizing Maps

Creating maps is one of the „business ends“ of QGIS. Map views are canvases to visualize spatial layers loaded in the project, with the various rendering settings applied to them. They can be of *2D* or *3D* type or plotted along a *profile line*, show different scale or extent, or display a different set of the loaded layers thanks to *map themes*.

### 11.1 2D Map View

The 2D map view (also called **Map canvas**) is the central place where maps are displayed. QGIS opens by default with a single map view (called *main map*), showing layers in 2D, and tightly bound to the *Layers* panel. That window reflects the rendering (symbology, labeling, visibilities...) you applied to the loaded layers.

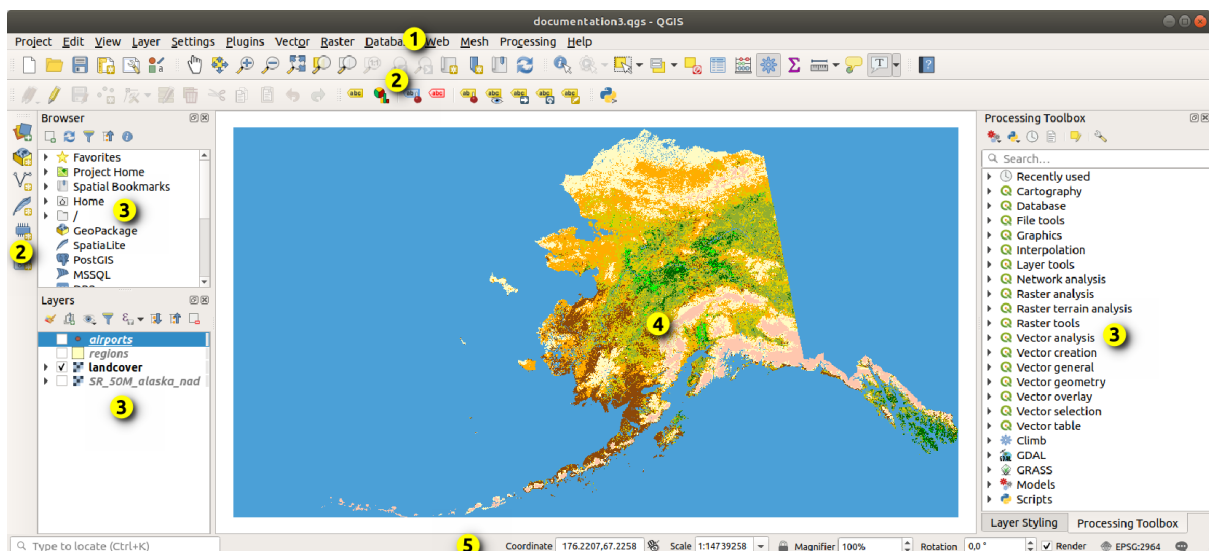














Abb. 11.1: QGIS GUI

## 11.1.1 Funktionen des Kartenfensters

Wenn Sie einen Layer hinzufügen (siehe z.B. *Öffnen von Daten*), sucht QGIS automatisch nach dem KBS des Layers. Wenn für das Projekt ein anderes KBS eingestellt ist, dann wird die Ausdehnung des Layers „on-the-fly“ in dieses KBS übersetzt (siehe *Project Coordinate Reference Systems*). Wenn Sie mit einem leeren QGIS-Projekt beginnen und wird die Kartenansicht auf die Ausdehnung des/der zuerst hinzugefügten Layer gezoomt. Wenn bereits Layer im Projekt vorhanden sind, wird keine Größenänderung der Kartenansicht durchgeführt, so dass nur Features sichtbar sind, die innerhalb der aktuellen Kartenansicht liegen.

Click on the map view and you should be able to interact with it, panning or zooming to different areas of the map. Dedicated tools are provided in the *Navigation Toolbar* and in the *View* menu, with handful shortcuts from the keyboard or the mouse buttons.


Tab. 11.1: Map canvas navigation tools

Werkzeug	Anwendung
 Pan Map	<ul style="list-style-type: none"> <li>Single left click: the map is centered on the clicked point, at the same scale</li> <li>Hold down the left mouse button and drag the map canvas.</li> </ul>
 Zoom In	<ul style="list-style-type: none"> <li>Single left click: the map is centered on the clicked point, while the scale gets doubled</li> <li>Drag a rectangle on the map canvas with the left mouse button to zoom in to an area.</li> <li>Hold the <b>Alt</b> key to switch to the  Zoom Out tool.</li> </ul>
 Zoom Out	<ul style="list-style-type: none"> <li>Single left click: the map is centered on the clicked point, while the scale gets halved</li> <li>Drag a rectangle on the map canvas with the left mouse button to zoom out from an area.</li> <li>Hold the <b>Alt</b> key to switch to the  Zoom In tool.</li> </ul>
 Pan Map to Selection	Pan the map to the selected features of all the selected layers in the <i>Layers</i> panel.
 Zoom To Selection	Zoom to the selected features of all the selected layers in the <i>Layers</i> panel. <i>Also available in the layer contextual menu</i>
 Zoom To Layer(s)	Zoom to the extent of all the selected layers in the <i>Layers</i> panel. <i>Also available in the layer contextual menu</i>
 Zoom Full	Zoom to the extent of all the layers in the project or to the <i>project full extent</i> .
 Zoom Last	Zoom the map to the previous extent in history.
 Zoom Next	Zoom the map to the next extent in history.
 Zoom to Native Resolution	Zoom the map to a level where one pixel of the active raster layer covers one screen pixel. <i>Also available in the layer contextual menu</i>
Mouse wheel	<ul style="list-style-type: none"> <li>Pan map: Hold and drag the mouse wheel.</li> <li>Zoom: Roll the mouse wheel to zoom in or zoom out. With <b>Ctrl</b> key pressed while rolling the mouse wheel results in a finer zoom.</li> <li>Press the back or forward button to browse the map canvas zoom history.</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 11.1 – Fortsetzung der vorherigen Seite

Werkzeug	Anwendung
Keyboard	<ul style="list-style-type: none"> <li>• Pan map: Hold down the <code>Space</code> key and move the mouse. Press the arrow keys to pan up, down, left and right.</li> <li>• Zoom in: Press <code>PgUp</code> or <code>Ctrl++</code></li> <li>• Zoom out: Press <code>PgDown</code> or <code>Ctrl+-</code></li> <li>• Zoom to area: When certain map tools are active (Identify, Measure...), hold down <code>Shift</code> and drag a rectangle on the map to zoom to that area. Not compatible with active selection or edit tools.</li> </ul>

Mit einem rechten Maustaste in die Karte können Sie die Koordinaten des angeklickten Punktes kopieren, entweder im KBS der Karte, in WGS84-Koordinaten oder in einem benutzerdefiniertes KBS ( *Koordinaten kopieren*). Die kopierten Informationen stehen dann in der Zwischenablage des Betriebssystems zur Verfügung und können über „Einfügen“ oder entsprechende Tastenkombianationen in einen Texteditor, einer Tabellenkalkulation, einem Skript, etc eingefügt werden.

### 11.1.2 Controlling map rendering



QGIS rendert standardmäßig alle sichtbaren Layer wenn das Kartenfenster aktualisirt werden muss. Die Abläufe, die eine ‚Erneuerung‘ der Wiedergabe verursachen sind:

- changing the visibility of a layer
- modifying symbology of a visible layer
- adding a layer
- panning or zooming
- resizing the QGIS window


QGIS ermöglicht es, den Wiedergabeprozess auf verschiedene Arten zu kontrollieren.

- at the *global level*
- per layer, using e.g. the *scale dependent rendering*
- or with dedicated tools in the GUI.

To stop the map drawing, press the `ESC` key. This will halt the refresh of the map canvas and leave the map partially drawn. It may however take a bit of time after pressing `ESC` for the map drawing to halt.

To suspend rendering, click the  *Render* checkbox in the bottom-right corner of the status bar. When  *Render* is unchecked, QGIS does not redraw the canvas in response to any of the usual triggers mentioned earlier. Examples of when you might want to suspend rendering include:


- adding many layers and symbolizing them prior to drawing
- adding one or more large layers and setting scale dependency before drawing
- adding one or more large layers and zooming to a specific view before drawing
- any combination of the above


Wenn Sie die  *Zeichnen* aktivieren, findet automatisch eine Erneuerung der Wiedergabe des Kartenfensters statt.

### 11.1.3 Zeitabhängige Darstellung in der Kartenansicht

In QGIS können Layer zeitlich gesteuert dargestellt werden, d.h. die Kartenansicht wird zeitabhängig verändert. Um dies durchführen zu können, benötigen Sie:

1. Einen oder mehrere Layer, für den zeitlich-dynamische Eigenschaften festgelegt wurden. QGIS unterstützt die zeitliche Steuerung für verschiedene Layer-Typen mit unterschiedlichen Einstellungen. Diese unterscheiden sich in erster Linie bei der Einstellung des Zeitintervalls, in dem der Layer angezeigt werden soll:
  - *Raster Layer*: es wird festgelegt, ob der Layer angezeigt werden soll oder nicht
  - *WMTS Layer*: es wird festgelegt, ob die Daten auf der Grundlage eines statischen Zeitbereichs oder nach einem dynamischen Zeitbereich dargestellt werden sollen
  - ref: *Vektor-Layer* <vectortemporalmenu>: Objekte können auch über Zeitwerte gesteuert werden, die in der Attributtabelle enthalten sind
  - *mesh layers*: zeigt dynamisch die Werte der aktiven Datensatzgruppen an

Wenn zeitlich-dynamische Eigenschaften für einen Layer aktiviert sind, ist neben dem Layer im *Layer* Bedienfeld ein Symbol  sichtbar, um anzuzeigen, dass der Layer zeitgesteuert ist. Klicken Sie auf das Symbol, um die zeitlichen Einstellungen zu ändern.

2. Aktivierung der zeitlichen Steuerung der Kartenansicht mit dem *Zeitsteuerung* Bedienfeld. Das Bedienfeld selber kann angezeigt werden über
  - das Symbol  *Zeitsteuerungsfenster* in der *Kartennavigationswerkzeugleiste*
  - oder über das Menü *Ansicht* ► *Bedienfelder* ► *Zeitsteuerung*

#### Das Zeitsteuerung-Bedienfeld

Das *Zeitsteuerung* Bedienfeld bietet die folgenden Möglichkeiten:

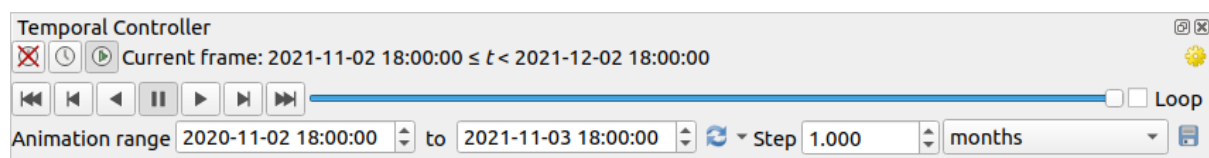













Abb. 11.2: Zeitsteuerung-Bedienfeld im Navigationsmodus

-  *Zeitliche Navigation abschalten*: alle zeitlichen Einstellungen sind deaktiviert und die aktiven Layer werden wie gewohnt angezeigt
-  *Zeit-Navigation in festem Bereich*: ein Zeitbereich wird festgelegt und nur Layer (oder Objekte), deren Zeitbereich sich mit diesem überschneidet, werden auf der Karte angezeigt
-  *Animierte Zeitnavigation*: ein Zeitbereich und eine Schrittlänge innerhalb dieses Zeitbereichs werden festgelegt und es werden nur Layer (oder Objekte) angezeigt, die sich innerhalb des jeweiligen Schrittes befinden
-  *Einstellungen* für die allgemeine Steuerung der Animation:
  - *Bildrate*: Anzahl der Bilder, die pro Sekunde angezeigt werden
  -  *Kumulativer Bereich*: alle Animationsbilder haben die gleiche Startzeit, aber unterschiedliche Enddaten und -zeiten. Dies ist nützlich, wenn Sie Daten in Ihrer zeitlichen Visualisierung akkumulieren möchten, anstatt ein „bewegliches Zeitfenster“ über die Daten laufen zu lassen.

## Animation über einen Zeitverlauf

Eine zeitliche Animation besteht aus einer Reihe von Bildern aller zeitlich aktiven Layer zu einem bestimmten Zeitpunkt innerhalb eines Zeitraums. So erstellen Sie eine zeitliche Animation:

1. Schalten Sie auf  *Animierte Zeitnavigation* um, so dass die Bedienelemente des Players angezeigt werden
  2. Geben Sie den *Animationsbereich* ein, der berücksichtigt werden soll. Mit der  Schaltfläche kann dieser wie folgt automatisch definiert werden:
    - *Auf ganzen Bereich setzen* legt den Animationsbereich so fest, dass die Zeiträume aller zeitlich aktivierten Layer berücksichtigt werden
    - *Auf Projektbereich voreinstellen* legt den Animationsbereich so fest, wie er in den *Projekteigenschaften* definiert ist
    - *Bereich eines einzelnen Layers setzen* legt den Bereich auf die Spanne der Werte eines einzelnen Layers fest
  3. Mit dem *Zeit- Schritt*, wird die Zeitspanne in einzelne Intervalle unterteilt. Es werden verschiedene Einheiten unterstützt z. B. *Sekunden* oder *Jahrhunderte*. Als Schritt ist auch die Option *Quellzeitstempel* verfügbar: Wenn sie ausgewählt ist, bewirkt sie, dass die zeitliche Navigation zwischen allen verfügbaren Zeitbereichen der Layer im Projekt wechselt. Dies ist nützlich, wenn in einem Projekt verschiedene Layer enthalten sind, welche nicht in einem zusammenhängenden Zeitraum liegen. Dies kann z. B. bei einem WMS-T-Dienst der Fall sein, wenn Bilder bereitgestellt werden, die unregelmäßige Zeitpunkten wiedergeben. Diese Option ermöglicht es Ihnen, mit einem Schritt einfach in den Zeitbereich zu wechseln, in dem das nächste Bild verfügbar ist.
  4. Klicken Sie auf die Schaltfläche , um eine Vorschau der Animation zu erhalten. QGIS erzeugt eine Serie von Bildern mit den Layern oder Objekten, die zu den eingestellten Zeiten dargestellt werden. Die Anzeige der Layer oder Objekte hängt davon ab, ob deren Zeitstempel innerhalb des jeweiligen Zeitintervalls liegen.
- The animation can also be previewed by moving the time slider. Checking the  *Loop* checkbox will repeatedly run the animation while clicking  stops a running animation. A full set of video player buttons is available.
- Durch horizontales Scrollen mit dem Mausekranz (sofern von Ihrer Maus unterstützt) und dem Cursor auf der Kartenansicht können Sie auch den Schieberegler für die zeitliche Navigation vorwärts und rückwärts verschieben ( "scrubben").
5. Klicken Sie auf die Schaltfläche  *Animation exportieren*, wenn Sie die Serie von Bildern exportieren wollen. Diese können später in einem Videoeditor zusammengefügt werden:

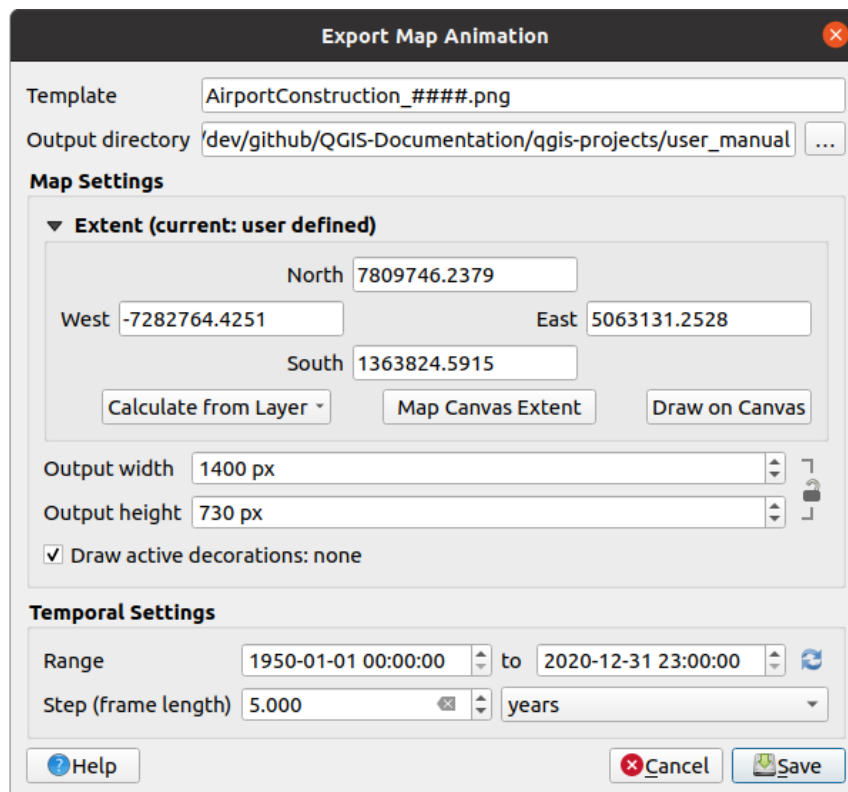


Abb. 11.3: Fenster zum Exportieren von Bildern einer Animation


- In der *Vorlage* für den Dateinamen werden die Platzhalter #### durch fortlaufende Bildnummern ersetzt
- Das *Ausgabeverzeichnis*
- Unter *Karteneinstellungen* können Sie:
  - Die räumliche *Ausdehnung* neu festlegen
  - Die Auflösung der Bilder festlegen (*Ausgabebreite* und *Ausgabehöhe*)
  - *Aktive Dekorationen anzeigen* legt fest, ob aktive *Dekorationen* in den Ausgabe-Bildern dargestellt sein sollen
- Unter *Zeiteinstellungen* können Sie folgende Parameter neu festlegen:
  - den *Zeit-Bereich* für die Animation
  - die *Schrittweite (Einzelbildlänge)* in der gewünschten Einheit

### 11.1.4 Bookmarking extents on the map

Spatial Bookmarks allow you to „bookmark“ a geographic location and return to it later. By default, bookmarks are saved in the user's profile (as *User Bookmarks*), meaning that they are available from any project the user opens. They can also be saved for a single project (named *Project Bookmarks*) and stored within the project file, which can be helpful if the project is to be shared with other users.

## Ein Lesezeichen erstellen

Um ein Lesezeichen zu erstellen:

1. Zoom and pan to the area of interest.
2. Select the menu option **View** ►  *New Spatial Bookmark...*, press **Ctrl+B** or right-click the  *Spatial Bookmarks* entry in the *Browser* panel and select *New Spatial Bookmark*. The *Bookmark Editor* dialog opens.

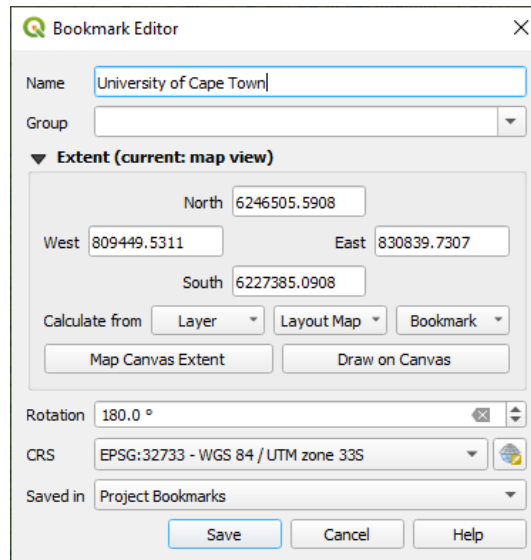



Abb. 11.4: The Bookmark Editor Dialog



3. Enter a descriptive name for the bookmark
4. Enter or select a group name in which to store related bookmarks
5. Select the extent of the area you wish to save, using the *extent selector* widget
6. Change the map *Rotation*
7. Indicate the *CRS* to use for the extent
8. Select whether the bookmark will be *Saved in User Bookmarks* or *Project Bookmarks* (by default, this drop-down list is set to *User Bookmarks*)
9. Press *Save* to add the bookmark to the list

Sie können mehrere Lesezeichen mit demselben Namen erstellen.

## Arbeiten mit Lesezeichen









To use and manage bookmarks, you can either use the *Spatial Bookmarks* panel or *Browser*.

Select **View** ►  *Show Spatial Bookmark Manager* or press **Ctrl+7** to open the *Spatial Bookmarks Manager* panel.

Select **View** ►  *Show Bookmarks* or **Ctrl+Shift+B** to show the  *Spatial Bookmarks* entry in the *Browser* panel.

You can perform the following tasks:

Tab. 11.2: Managing bookmark actions

Task	Spatial Bookmark Manager	Browser
<b>Zoom to Bookmark</b>	Double-click on it, or select the bookmark and press the  <i>Zoom to bookmark</i> button.	Double-click on it, drag and drop it to the map canvas, or right-click the bookmark and select <i>Zoom to Bookmark</i> .
<b>Delete bookmark</b>	Select the bookmark and click the  <i>Delete bookmark</i> button. Confirm your choice.	Right-click the bookmark and select <i>Delete Spatial Bookmark</i> . Confirm your choice.
<b>Export bookmarks XML</b>	Click the  <i>Import/Export Bookmarks</i> button and select  <i>Export</i> . All the bookmarks (user or project) are saved in an xml file.	Select one or more folders (user or project) or subfolders (groups), then right-click and select  <i>Export Spatial Bookmarks....</i> The selected bookmark subset is saved.
<b>Import bookmarks from XML</b>	Click the  <i>Import/Export Bookmarks</i> button and select  <i>Import</i> . All bookmarks in the XML file are imported as user bookmarks.	Right-click the <i>Spatial Bookmarks</i> entry or one of its folders (user or project) or subfolders (groups) to determine where to import the bookmarks, then select  <i>Import Spatial Bookmarks</i> . If performed on the <i>Spatial Bookmarks</i> entry, the bookmarks are added to <i>User Bookmarks</i> .
<b>Edit bookmark</b>	You can change a bookmark by changing the values in the table. You can edit the name, the group, the extent and if it is stored in the project or not.	Right-click the desired bookmark and select <i>Edit Spatial Bookmark....</i> The <i>Bookmark Editor</i> will open, allowing you to redefine every aspect of the bookmark as if you were creating it for the first time. You can also drag and drop the bookmark between folders (user and project) and subfolders (groups).

You can manage bookmark actions by right-click on the desired bookmark in the *Spatial Bookmarks Manager*. You can also zoom to bookmarks by typing the bookmark name in the *locator*.

### 11.1.5 Decorating the map

Decorations include Grid, Title Label, Copyright Label, Image, North Arrow, Scale Bar and Layout Extents. They are used to ,decorate' the map by adding cartographic elements.



## Gitter



*Grid* allows you to add a coordinate grid and coordinate annotations to the map canvas.

1. Select menu option *View ► Decorations ► Grid...* to open the dialog.

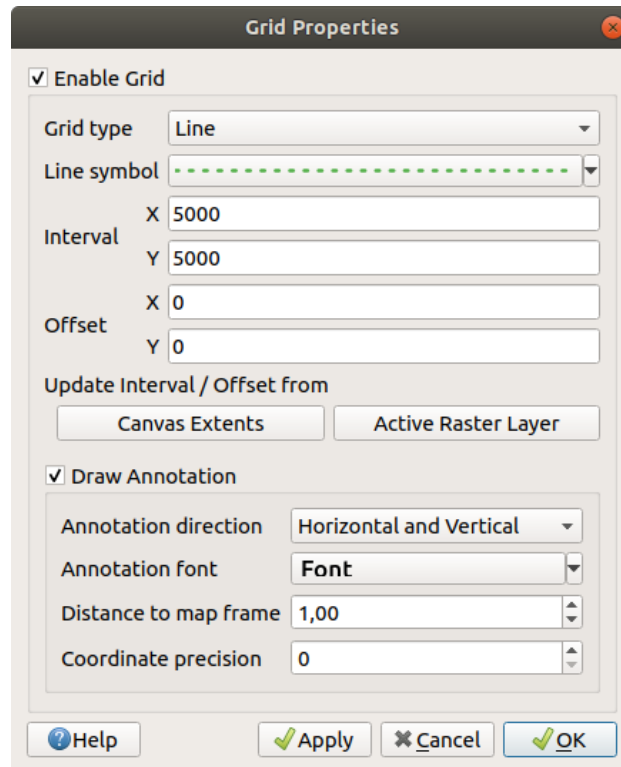


Abb. 11.5: Der Gitter Dialog

2. Tick ☒ *Enable grid* and set grid definitions according to the layers loaded in the map canvas:
  - The *Grid type*: it can be *Line* or *Marker*
  - The associated *Line symbol* or *marker symbol* used to represent the grid marks
  - The *Interval X* and *Interval Y* between the grid marks, in map units
  - An *Offset X* and *Offset Y* distance of the grid marks from the bottom left corner of the map canvas, in map units
  - The interval and offset parameters can be set based on the:
    - *Canvas Extents*: generates a grid with an interval that is approximatively 1/5 of the canvas width
    - *Active Raster Layer* resolution
3. Tick ☒ *Draw annotations* to display the coordinates of the grid marks and set:
  - The *Annotation direction*, ie how the labels would be placed relative to their grid line. It can be:
    - *Horizontal* or *Vertical* for all the labels
    - *Horizontal and Vertical*, ie each label is parallel to the grid mark it refers to
    - *Boundary direction*, ie each label follows the canvas boundary, and is perpendicular to the grid mark it refers to
  - The *Annotation font* (text formatting, buffer, shadow...) using the *font selector widget*

- The *Distance to map frame*, margin between annotations and map canvas limits. Convenient when *exporting the map canvas* eg to an image format or PDF, and avoid annotations to be on the „paper“ limits.
  - The *Coordinate precision*
4. Click *Apply* to verify that it looks as expected or *OK* if you're satisfied.

## Title Label

**T** *Title Label* allows you to decorate your map with a **Title**.

To add a Title Label decoration:

1. Select menu option *View ► Decorations ► Title Label...* to open the dialog.

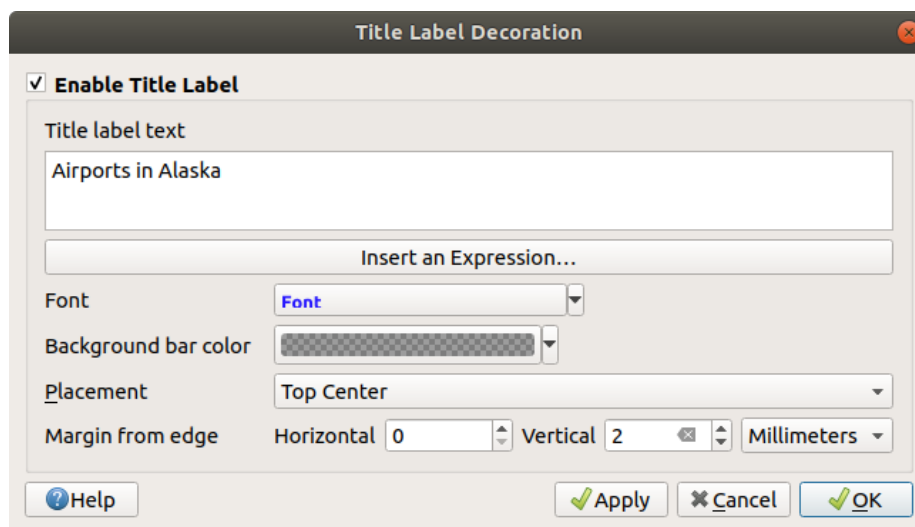


Abb. 11.6: The Title Decoration Dialog

2. Make sure ☒ *Enable Title Label* is checked
3. Enter the title text you want to place on the map. You can make it dynamic using the *Insert or Edit an Expression...* button.
4. Choose the *Font* for the label using the *font selector widget* with full access to QGIS *text formatting* options. Quickly set the font color and opacity by clicking the black arrow to the right of the font combo box.
5. Select the *color* to apply to the title's *Background bar color*.
6. Choose the *Placement* of the label in the canvas: options are *Top left*, *Top Center* (default), *Top Right*, *Bottom left*, *Bottom Center* and *Bottom Right*.
7. Refine the placement of the item by setting a horizontal and/or vertical *Margin from Edge*. These values can be in **Millimeters** or **Pixels** or set as a **Percentage** of the width or height of the map canvas.
8. Click *Apply* to verify that it looks as expected or *OK* if you're satisfied.

## Copyright Label



**Copyright Label** can be used to decorate your map with a **Copyright** label.

To add this decoration:

1. Select menu option *View ► Decorations ► Copyright Label...* to open the dialog.

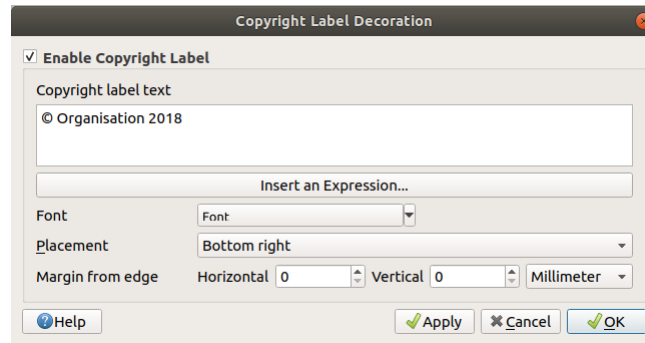


Abb. 11.7: The Copyright Decoration Dialog

2. Make sure *Enable Copyright Label* is checked
3. Enter the copyright text you want to place on the map. You can make it dynamic using the *Insert or Edit an Expression...* button.
4. Choose the *Font* for the label using the *font selector widget* with full access to QGIS *text formatting* options. Quickly set the font color and opacity by clicking the black arrow to the right of the font combo box.
5. Choose the *Placement* of the label in the canvas: options are *Top left*, *Top Center*, *Top Right*, *Bottom left*, *Bottom Center*, and *Bottom Right* (default for Copyright decoration)
6. Refine the placement of the item by setting a horizontal and/or vertical *Margin from Edge*. These values can be in **Millimeters** or **Pixels** or set as a **Percentage** of the width or height of the map canvas.
7. Click *Apply* to verify that it looks as expected or *OK* if you're satisfied.

## Image Decoration



**Image** allows you to add an image (logo, legend, ..) on the map canvas.

To add an image:

1. Select menu option *View ► Decorations ► Image...* to open the dialog.

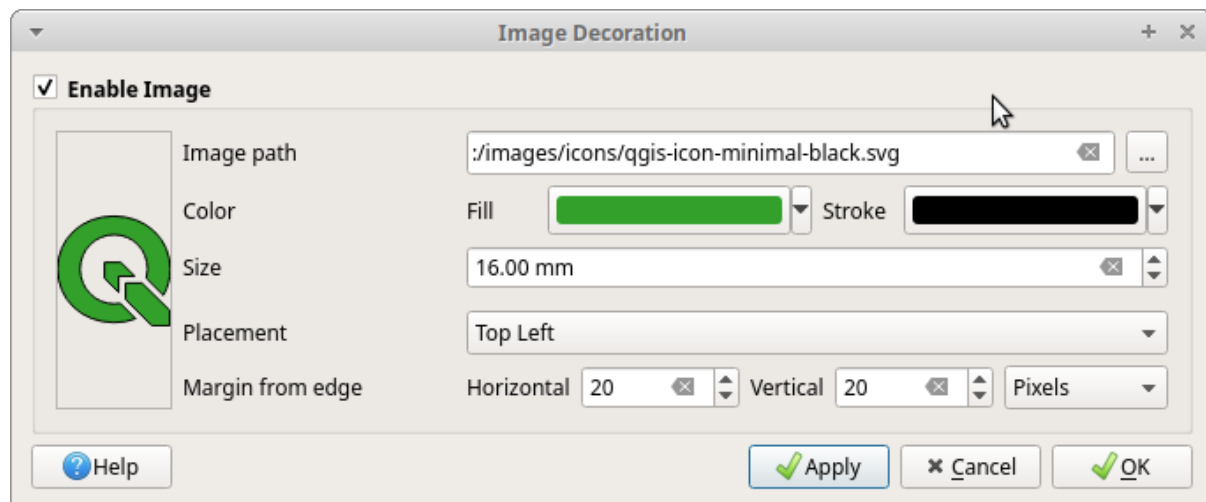



Abb. 11.8: The Image Decoration Dialog

2. Make sure ☒ *Enable Image* is checked
3. Select a bitmap (e.g. png or jpg) or SVG image using the ... <sup>Browse</sup> button
4. If you have chosen a parameter enabled SVG then you can also set a *Fill* or *Stroke* (outline) color. For bitmap images, the color settings are disabled.
5. Set a *Size* of the image in mm. The width of selected image is used to resize it to given *Size*.
6. Choose where you want to place the image on the map canvas with the *Placement* combo box. The default position is *Top Left*.
7. Set the *Horizontal* and *Vertical Margin from (Canvas) Edge*. These values can be set in **Millimeters**, **Pixels** or as a **Percentage** of the width or height of the map canvas.
8. Click *Apply* to verify that it looks as expected and *OK* if you're satisfied.

## Nordpfeil

 *North Arrow* allows you to add a north arrow on the map canvas.

To add a north arrow:

1. Select menu option *View ► Decorations ► North Arrow...* to open the dialog.

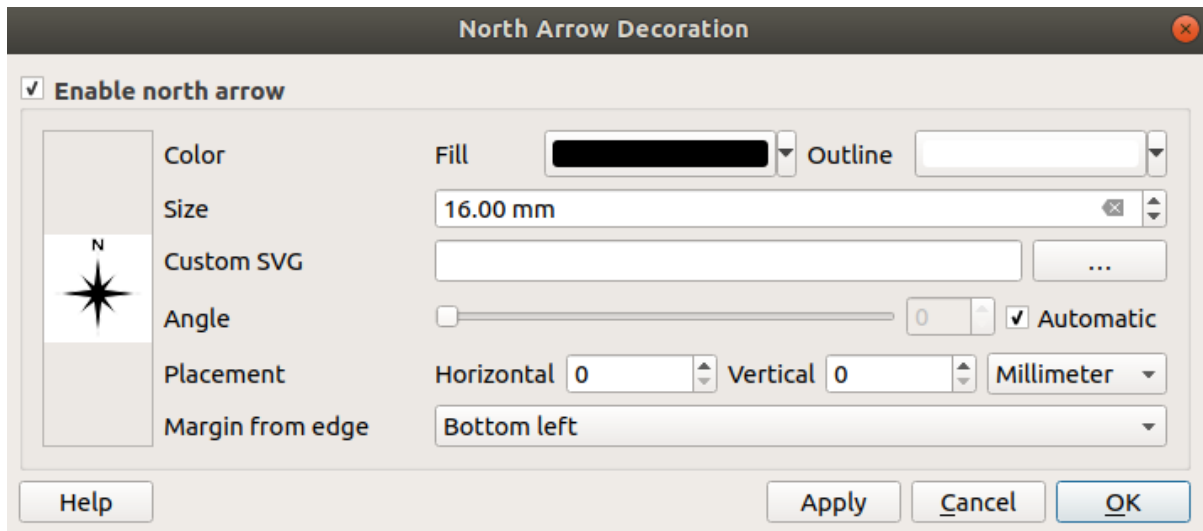




Abb. 11.9: Der Nordpfeil Dialog

2. Make sure  *Enable north arrow* is checked
3. Optionally change the color and size, or choose a custom SVG
4. Optionally change the angle or choose **Automatic** to let QGIS determine the direction
5. Optionally choose the placement from the Placement combo box
6. Optionally refine the placement of the arrow by setting a horizontal and/or vertical *Margin from (Canvas) Edge*. These values can be in **Millimeters** or **Pixels** or set as a **Percentage** of the width or height of the map canvas.
7. Click *Apply* to verify that it looks as expected and *OK* if you're satisfied.

## Maßstab

 *Scale Bar* adds a simple scale bar to the map canvas. You can control the style and placement, as well as the labelling of the bar. The scale bar respects the active *project's distance unit* as defined in *Project properties* ► *General* ► *Units for distance measurement*.

Das Hinzufügen eines Maßstabbalkens geht folgendermaßen:

1. Select menu option *View* ► *Decorations* ► *Scale Bar...* to open the dialog

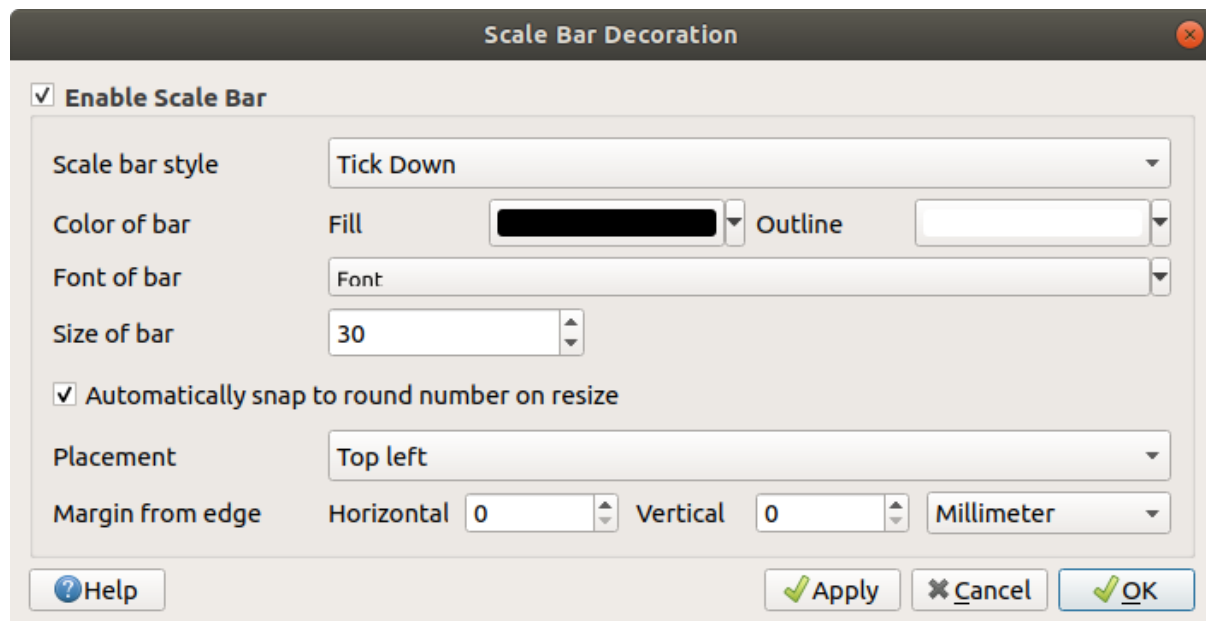


Abb. 11.10: Der Maßstab Dialog

2. Make sure ☒ *Enable scale bar* is checked
3. Choose a style from the *Scale bar style* combo box
4. Select the *Color of bar* by choosing a fill color (default: black) and an outline color (default: white). The scale bar fill and outline can be made opaque by clicking on the down arrow to the right of the color input.
5. Select the font for the scale bar from the *Font of bar* combo box
6. Set the *Size of bar* in the active unit
7. Optionally check ☒ *Automatically snap to round number on resize* to display easy-to-read values
8. Choose the placement from the *Placement* combo box
9. You can refine the placement of the item by setting a horizontal and/or vertical *Margin from (Canvas) Edge*. These values can be in **Millimeters** or **Pixels** or set as a **Percentage** of the width or height of the map canvas.
10. Click *Apply* to verify that it looks as expected or *OK* if you're satisfied.

## Layout Extents

*Layout Extents* adds the extents of *map item(s)* in print layout(s) to the canvas. When enabled, the extents of all map items within all print layouts are shown using a lightly dotted border labeled with the name of the print layout and map item. You can control the style and labeling of the displayed layout extents. This decoration is useful when you are tweaking the positioning of map elements such as labels, and need to know the actual visible region of print layouts.

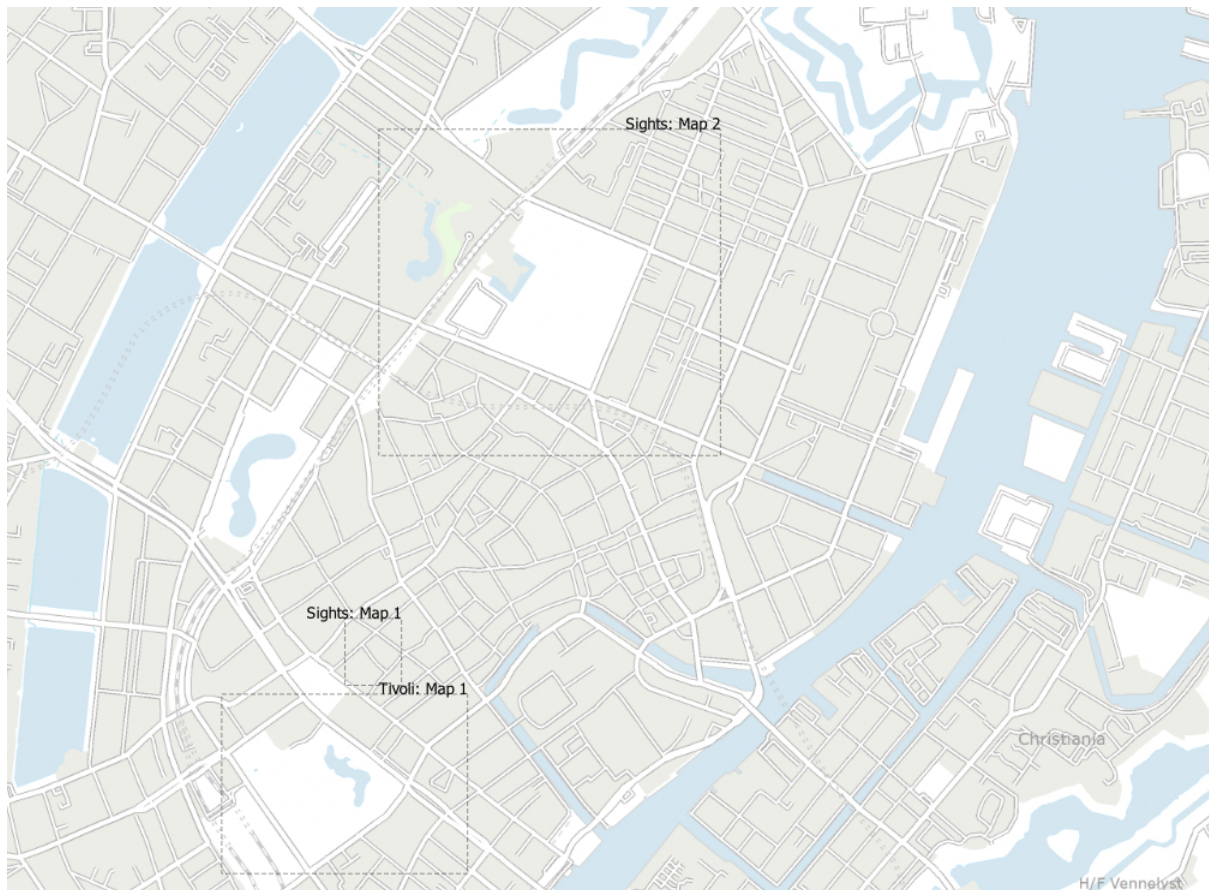


Abb. 11.11: Example of layout extents displayed in a QGIS project with two print layouts. The print layout named ‚Sights‘ contains two map items, while the other print layout contains one map item.

To add layout extent(s):

1. Select **View ► Decorations ► Layout Extents** to open the dialog

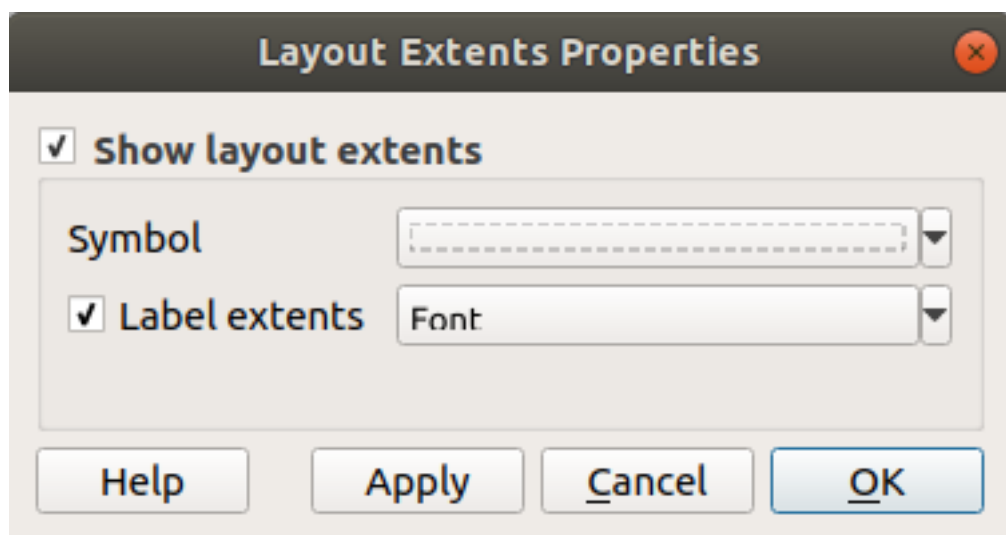



Abb. 11.12: The Layout Extents Dialog

2. Make sure  *Show layout extents* is checked.

3. Optionally change the symbol and labeling of the extents.
4. Click *Apply* to verify that it looks as expected and *OK* if you're satisfied.

---

### **Tipp: Decorations Settings**

When you save a QGIS project file, any changes you have made to Grid, North Arrow, Scale Bar, Copyright and Layout Extents will be saved in the project and restored the next time you load the project.

---

## 11.1.6 Beschriftungstools

Annotations are another type of elements added onto the map canvas to provide additional information that can not be depicted by the rendered layers. Unlike *labels* that rely on attribute values stored in vector layers, annotations are independent details, stored within the project file itself.

Two families of annotations are available in QGIS:

- **Feature annotations:** they are actual georeferenced features of text, marker, line or polygon type stored within a special layer type called „annotation layer“. They are tied to a particular geographic location, meaning that moving your map, changing the scale or changing projection won't cause your annotations to jump around the map. Rather, they'll be locked in place to the location you've drawn them.
- **Balloon annotations:** these are individuals annotations of text, form or image type placed inside a bubble. They can be associated to any layer for their visibility, are drawn on top of the map canvas. The size is dependent from the map canvas scale, and its position can be anchored.

---

### **Tipp: Layout the map with annotations**












You can print or export annotations with your map to various formats using:

- map canvas export tools available in the *Project* menu
  - *print layout*, in which case you need to check *Draw map canvas items* in the corresponding map item properties
- 

The *Annotations Toolbar* provides a set of tools to create and interact with both families of annotations.




Tab. 11.3: The Annotations Toolbar actions

Werkzeug	Anwendung	Scope
 New Annotation Layer	Create a new layer to store annotations	Feature annotations
Main Annotation Layer Properties	Control settings of the Main Annotation Layer	
 Modify Annotations	Select, move, resize and modify symbology properties of annotations	
 Create Polygon Annotation	Create an annotation as a polygon feature	
 Create Line Annotation	Create an annotation as a polyline feature	
 Create Marker Annotation	Create an annotation as a point feature	
 Create Text Annotation at Point	Create an annotation as a text label	Balloon annotations
 Create Text Annotation along Line	Create an annotation as a curved text along a linestring	
 Text Annotation	Select and create a text formatted annotation	
 HTML Annotation	Select and create annotation with an HTML file's content	
 SVG Annotation	Select and create annotation showing an SVG file	
 Form Annotation	Select and create annotation showing attributes of a vector layer in a custom form file	

## Feature Annotations

Feature annotations are stored in **annotation layers**. Unlike conventional layers, an annotation layer is available only in the current project and can contain features of different types (text, marker, line, polygon). The layer has no attributes and no symbology associated, but instead each feature can be symbolized on an item-by-item basis, through *Layer Styling* panel.



Two types of annotation layer are available in QGIS:

- A common *Annotation Layer*: you can create one using the  New Annotation Layer tool. It is listed in the *Layers* panel, allowing you to control its features' visibility, move it to show above or below particular layers in your map, like any common layer. Double-click the layer and you can access its properties.
- The *Main Annotation Layer*: By default, this is where annotations are stored when no annotation layer is available in the project or is selected at creation time. This layer is always drawn on the very top of your map and you won't see it listed in the *Layers* panel alongside the other layers in your project, meaning that its features are always visible. The *Main Annotation Layer Properties* entry on the *Annotations* toolbar helps you open its properties dialog.

## Layer-Eigenschaften

The properties dialog of an annotation layer provides the following tabs:






- *Information*: a read-only dialog representing an interesting place to quickly grab summarized information and metadata on the current layer. This may include the layer extent, count of items per annotation type and total count, CRS details, ...
- *Source*: defines general settings for the annotation layer. You can:
  - Set a *Layer name* that will be used to identify the layer in the project (in the *Layers Panel*, with expressions, ...)

- Display the layer's *Assigned Coordinate Reference System (CRS)*: you can change the layer's CRS, selecting a recently used one in the drop-down list or clicking on  **Select CRS** button (see *Coordinate Reference System Selector*). Use this process only if the CRS applied to the layer is a wrong one or if none was applied.
- *Rendering*:
  - You can set the *Maximum (inclusive)* and *Minimum (exclusive)* scale, defining a range of scale in which features will be visible. Out of this range, they are hidden. The  **Set to current canvas scale** button helps you use the current map canvas scale as boundary of the range visibility. See *Visibility Scale Selector* for more information.
  - *Opacity*: You can make the underlying layer in the map canvas visible with this tool. Use the slider to adapt the visibility of your vector layer to your needs. You can also make a precise definition of the percentage of visibility in the menu beside the slider.
  - *Blending mode* at the *Layer* level: You can achieve special rendering effects with these tools that you may previously only know from graphics programs. The pixels of your overlaying and underlying layers are mixed through the settings described in *Mischmodi*.
  - Wenden Sie *Zeicheneffekte* auf alle Layerobjekte an, mit dem *Zeicheneffekte* Knopf.

Some of these options are accessible from the feature annotation *Symbology* properties.

## Interacting with features

The Feature annotations have dedicated tools for creation depending on their type:

-  Create Polygon Annotation
-  Create Line Annotation
-  Create Marker Annotation
-  Create Text Annotation at Point
-  Create Text Annotation along Line

All the usual QGIS shortcuts for creating features apply when creating annotation items. A line or polygon annotation is drawn by left-clicking once for each vertex, with a final right mouse click to complete the shape. Snapping can be enabled while you draw, you can use the *Advanced Digitizing Tools* to precisely place vertices, and even switch the *drawing tools* to the streaming mode for completely free-form shapes.

Unlike common layers, an annotation layer does not need to be active before you select its features. Simply grab the




**Modify Annotations** tool and you will be able to interact with any feature annotation:

- **Selection**: left-click on the annotation. By default, annotations are rendered in the order of creation, meaning that recent annotations will be placed on top of older. You may need to play with the Z-index property of features in order to select ones they would sit above.
- **Moving**: Left click on a selected annotation item to start moving it. A right-click or pressing **Esc** key cancels the move, while a second left click will confirm the move. The displacement can also be controlled pressing the cursor keys:
  - **Shift+key** for big movement
  - **Alt+key** for 1 px movement
- **Geometry modification**: for line or polygon annotations, left-click on a vertex of the geometry, move and click again. Double-click a segment to add a new vertex.
- **Delete**: Pressing the **Del** or **Backspace** key while an annotation is selected will delete that annotation

- *Change feature symbology*





## Feature symbology

A selected annotation will display its *Symbology* properties in the *Layer styling* panel. You can:

- Modify the appearance using full capabilities of:
  - the *symbol properties* for polygon, polyline and marker annotations
  - the *text format properties* for text-based annotations. A text area allows you to construct the string to display using QGIS expression functions. It is also possible to set the *Alignment* for text annotation at point (left, center or right of the text point).
- For text annotation at point, also configure whether it should *Ignore map rotation* or *Rotate with map*. In both cases, a custom *Angle* can be set for the feature orientation.
- For text annotation along a line, configure an *Offset from line* in the unit of your choice
- Configure a  *Reference scale*: indicates the map scale at which symbol or text sizes which uses paper-based units (such as millimeters or points) relate to. The sizes will be scaled accordingly whenever the map is viewed at a different scale. For instance, a line feature wide of 2mm at 1:2000 *Reference scale* will be rendered using 4mm when the map is viewed at 1:1000.
- Set a *Z-index*: a feature with a higher index is placed on top of features with lower index. A convenient setting for both feature display and selection.
- Modify some of the *Layer rendering* settings

## Balloon annotations

You can add balloon annotations through *Edit ► Add Annotation ►* menu or from the *Annotations Toolbar*:

-  *Text Annotation* for custom formatted text
-  *HTML Annotation* to place the content of an `html` file
-  *SVG Annotation* to add an `SVG` symbol
-  *Form Annotation*: useful to display attributes of a vector layer in a customized `ui` file (see Abb. 11.13). This is similar to the *custom attribute forms*, but displayed in an annotation item. Also see this video <https://www.youtube.com/watch?v=0pDBuSbQ02o&feature=youtu.be&t=2m25s> from Tim Sutton for more information.

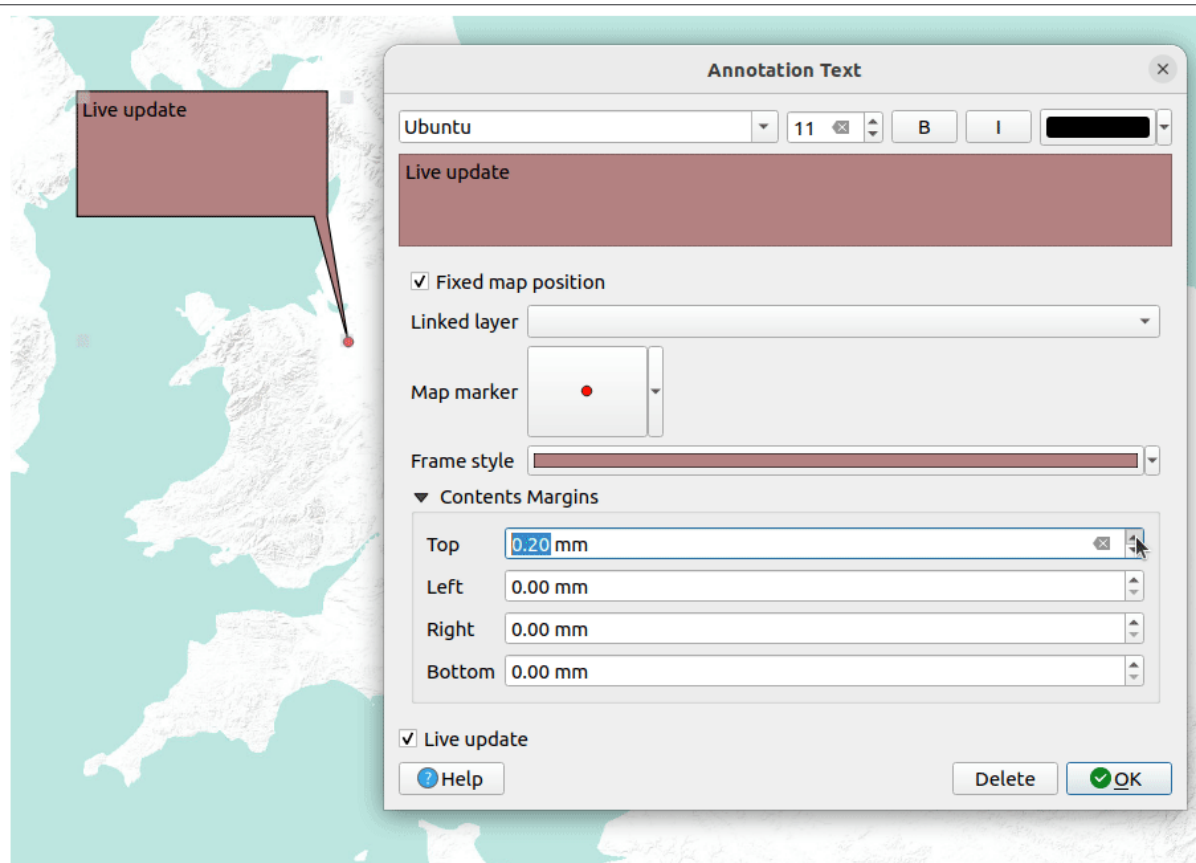


Abb. 11.13: Examples of balloon annotations

To add a balloon annotation, select the corresponding tool and click on the map canvas. An empty balloon is added. Double-click on it and a dialog opens with various options. This dialog is almost the same for all the annotation types:

- At the top, a file selector to fill with the path to an `html`, `svg` or `ui` file depending on the type of annotation. For text annotation, you can enter your message in a text box and set its rendering with the normal font tools.
- ☒ *Fixed map position*: when unchecked, the balloon placement is based on a screen position (instead of the map), meaning that it's always shown regardless the map canvas extent.
- *Linked layer*: associates the annotation with a map layer, making it visible only when that layer is visible.
- *Map marker*: using *QGIS symbols*, sets the symbol to display at the balloon anchor position (shown only when *Fixed map position* is checked).
- *Frame style*: sets the frame background color, transparency, stroke color or width of the balloon using QGIS symbols.
- *Contents margins*: sets interior margins of the annotation frame.
- ☒ *Live update* allows you to live preview your changes.

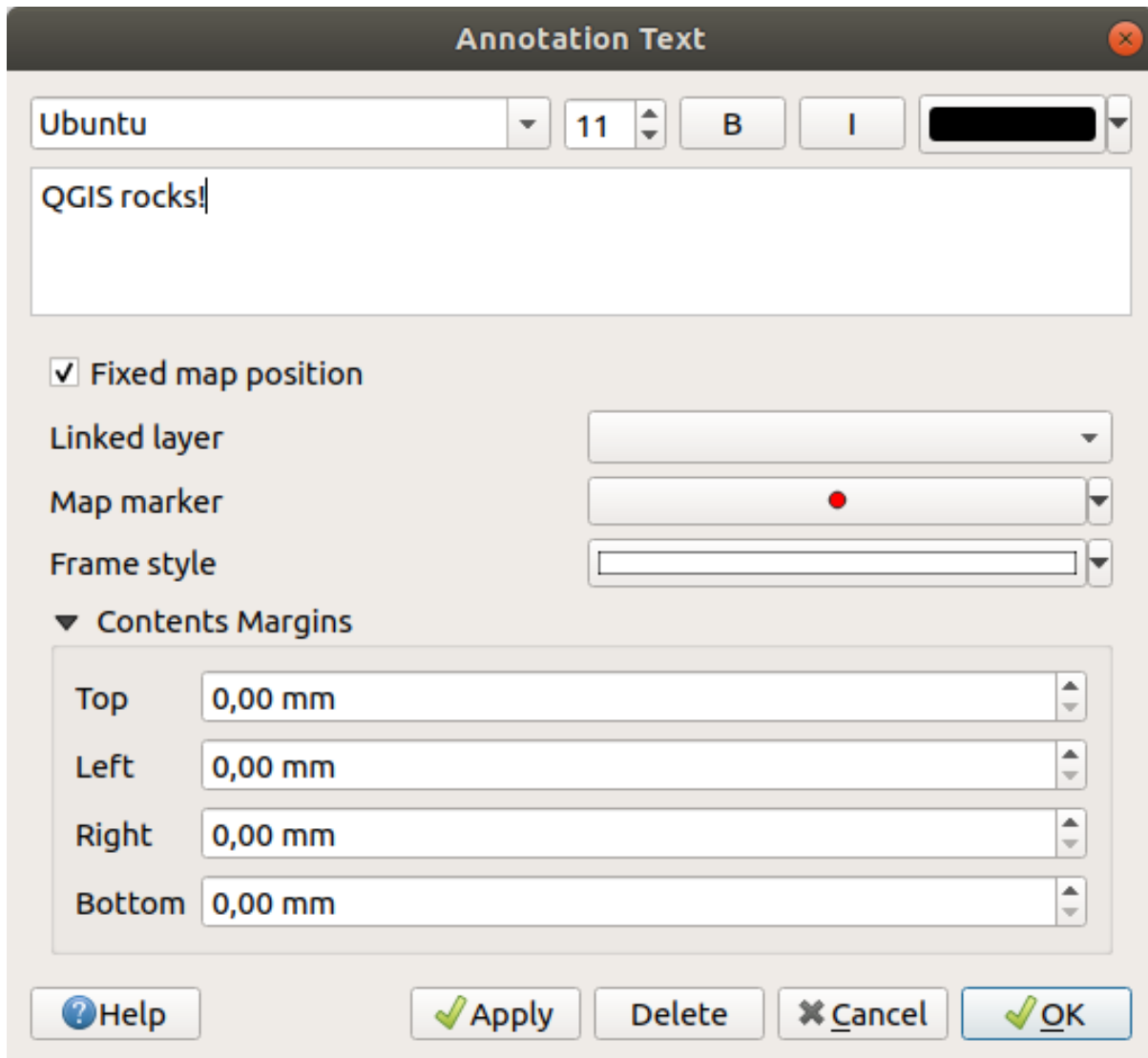





Abb. 11.14: A balloon annotation text dialog

When a balloon annotation tool is active, you can also:

- Select an annotation
- Resize an annotation
- Move an annotation by map position (by dragging the map marker) or by moving only the balloon.
- Delete an annotation: select it and either press the `Del` or `Backspace` button, or double-click it and press the *Delete* button in the properties dialog.
- Right-click and in the contextual menu:
  -  *Copy coordinate* of the annotation's map marker in various CRS
  -  *Edit* properties of the annotation. Same as double-clicking the annotation.
  -  *Delete* the annotation

## 11.1.7 Messen

### Allgemeine Informationen

QGIS bietet vier Mittel zur Geometriemessung:






- interactive measurement tools 
- measuring in the  Field Calculator
- derived measurements in the *Identifying Features* tool
- the vector analysis tool: *Vector ► Geometry Tools ► Export/Add Geometry Columns*



Das Messen funktioniert innerhalb projizierter Koordinatensystemen (z. B. UTM) und unprojizierten Daten. Die ersten drei Messwerkzeuge verhalten gleich, für globale Projekteinstellungen:

- Unlike most other GIS, the default measurement metric is ellipsoidal, using the ellipsoid defined in *Project ► Properties... ► General*. This is true both when geographic and projected coordinate systems are defined for the project.
- If you want to calculate the projected/planimetric area or distance using cartesian maths, the measurement ellipsoid has to be set to „None/Planimetric“ (*Project ► Properties... ► General*). However, with a geographic (ie unprojected) CRS defined for the data and project, area and distance measurement will be ellipsoidal.

However, neither the identify tool nor the field calculator will transform your data to the project CRS before measuring. If you want to achieve this, you have to use the vector analysis tool: *Vector ► Geometry Tools ► Add Geometry Attributes...* Here, measurement is planimetric, unless you choose the ellipsoidal measurement.

### Measure length, areas, bearings and angles interactively

Click the  icon in the Attribute toolbar to begin measurements. The down arrow near the icon switches between  length,  area,  bearing or  angle. The default unit used in the dialog is the one set in *Project ► Properties... ► General* menu.

For the *Measure Line* and the *Measure Area* the measurements can be done in  *Cartesian* or  *Ellipsoidal* measure.


---

#### Bemerkung: Configuring the measure tool

While measuring length or area, clicking the *Configuration* button at the bottom of the widget opens the *Settings ► Options ► Map Tools* menu, where you can select the rubberband color, the precision of the measurements and the unit behavior. You can also choose your preferred measurement or angle units, but keep in mind that those values are overridden in the current project by the selection made in the *Project ► Properties... ► General* menu, and by the selection made in the measurement widget.

---

All measuring modules use the snapping settings from the digitizing module (see section *Einrasttoleranz und Suchradius einstellen*). So, if you want to measure exactly along a line feature, or around a polygon feature, first set its layer snapping tolerance. Now, when using the measuring tools, each mouse click (within the tolerance setting) will snap to that layer.

The  *Measure Line* measures distances between given points. The tool then allows you to click points on the map. Each segment length, as well as the total, shows up in the measure window. In the measure window, you will see coordinates for all your points and distances. Keep in mind that the first row will contain only coordinates, as it represents your starting point. Now it is possible to copy all your line measurements at once to the clipboard using the *Copy* button. Clicking the *Configuration* button you will access to *Measure Tool Copy Settings* where you can set up *copy options*. To stop measuring, click the right mouse button.

Note that you can use the drop-down list near the total to change the *measurement units* interactively while working with the measure tool. This unit is retained for the widget until a new project is created or another project is opened.

The *Info* section in the dialog explains how calculations are made according to the CRS settings available.

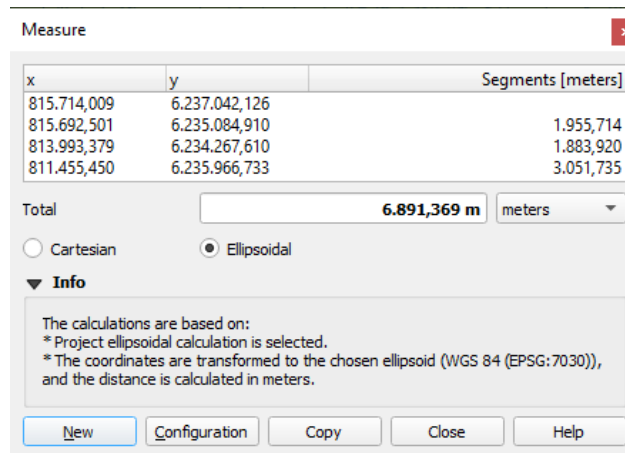


Abb. 11.15: Entfernung messen



**Measure Area:** Areas can also be measured. In the measure window, the accumulated area size appears. Right-click to stop drawing. The Info section is also available as well as the ability to switch between different *area units*.

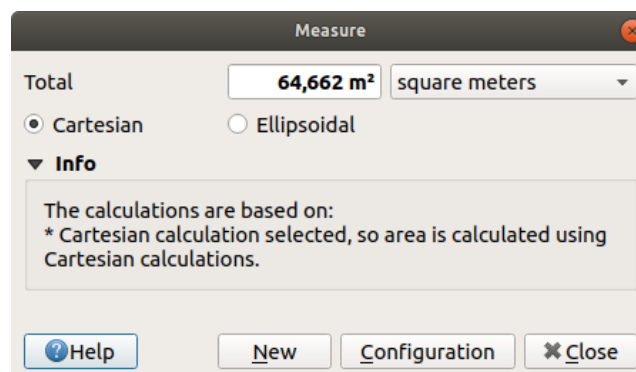


Abb. 11.16: Fläche messen



**Measure Bearing:** You can also measure bearings. The cursor becomes cross-shaped. Click to draw the first point of the bearing, then move the cursor to draw the second point. The measurement is displayed in a pop-up dialog.

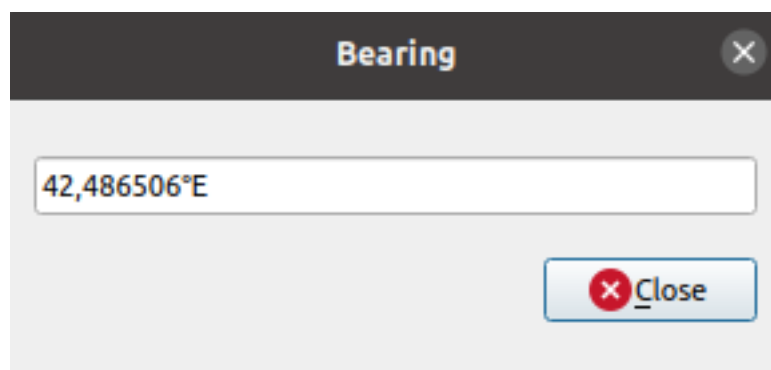



Abb. 11.17: Measure Bearing

 **Measure Angle**: You can also measure angles. The cursor becomes cross-shaped. Click to draw the first segment of the angle you wish to measure, then move the cursor to draw the desired angle. The measurement is displayed in a pop-up dialog.

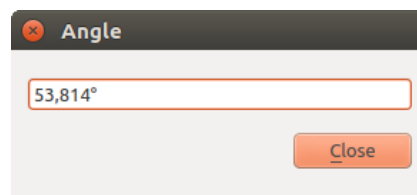



Abb. 11.18: Winkel messen

## 11.1.8 Weitere Kartenansichten hinzufügen

It is also possible to open additional map views whose content could diverge from the *Layers* panel current state. To add a new map view, go to **View** ►  **New Map View**. A new floating widget, mimicking the main map view's rendering, is added to QGIS. You can add as many map views as you need. They can be kept floating, placed side by side or stacked on top of each other.

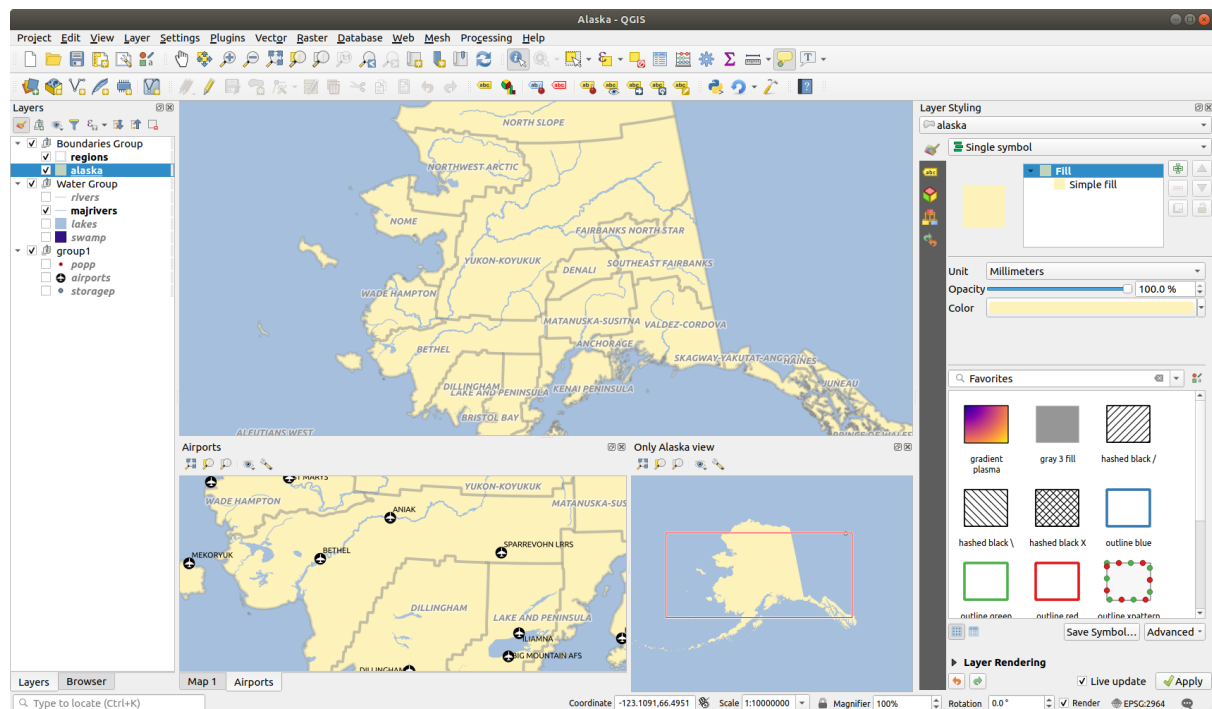














Abb. 11.19: Verschiedene Kartenansichten mit unterschiedlichen Einstellungen

Am oberen Rand jeder Kartenansicht befindet sich eine Werkzeugleiste mit den folgenden Möglichkeiten:

-  Volle Ausdehnung,  Zu ausgewählten Objekten zoomen und  Auf Layer zoomen um in der Ansicht zu navigieren
-  Kartenthema der Ansicht setzen, um das *Kartenthema* zur Anzeige in dieser Kartenansicht auszuwählen. Wenn es auf (keine) eingestellt ist, folgt die Darstellung den Änderungen im :guilabel:`Layer`-Bedienfeld.
-  Ansichtseinstellungen um die Kartenansicht zu konfigurieren:
  -  *Ansichtszentrum mit Hauptkarte synchronisieren*: synchronisiert die Mitte der Kartenansichten, ohne den Maßstab zu ändern; so können Sie eine Übersichtskarte oder eine vergrößerte Karte erstellen, die der Mitte der Kartenansicht folgt.





-  *Ansicht mit Auswahl synchronisieren*: das gleiche wie *Zu ausgewählten Objekten zoomen*
- *Maßstab*
- *Drehung*
- *Vergrößerung*
-  *Maßstäbe synchronisieren*: der Maßstab wird mit dem der Hauptkarte synchron gehalten. Ein *Maßstabsfaktor* kann dann angegeben werden, so dass Sie eine Ansicht erhalten, die immer z.B. den zweifachen Maßstab der Hauptkarte hat.
-  *Anmerkungen anzeigen*
-  *Cursor-Position anzeigen*
-  *Hauptkartenausdehnung anzeigen*
-  *Beschriftungen anzeigen*: erlaubt es die Beschriftungen auszublenden, auch wenn sie in den Einstellungen des Layers aktiviert sind
- *Karten-KBS ändern...*
- *Ansicht umbenennen...*

### 11.1.9 Exportieren der Kartenansicht

Die von Ihnen erstellten QGIS-Projekte können als komplexe *Layouts* oder *Berichte* in verschiedene Formate exportiert werden. Es ist aber auch möglich, die aktuelle Darstellung in der Kartenansicht direkt zu exportieren, ohne ein zusätzliches Layout. Dieser schnelle „Screenshot“ der Kartenansicht hat einige praktische Funktionen.

Um die aktuellste Darstellung im Kartenfenster zu exportieren, müssen sie folgendes tun:

1. Gehen sie zum Menü *Projekt ► Import/Export*
2. abhängig davon, was Sie exportieren möchten, wählen Sie entweder
  -  *Karte als Bild speichern ...*
  - oder  *Karte als PDF exportieren...*

The two tools provide you with a common set of options. In the dialog that opens:

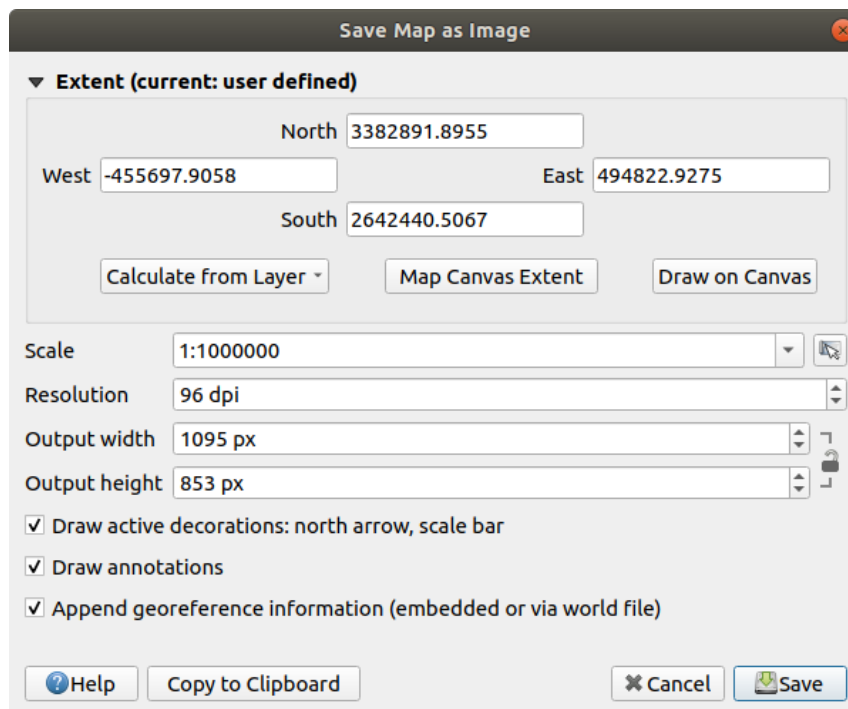


Abb. 11.20: The Save Map as Image dialog

1. Choose the *Extent* to export: it can be the current view extent (the default), the extent of a layer or a custom extent drawn over the map canvas. Coordinates of the selected area are displayed and manually editable.
2. Enter the *Scale* of the map or select it from the *predefined scales*: changing the scale will resize the extent to export (from the center).
3. Set the *Resolution* of the output
4. Control the *Output width* and *Output height* in pixels of the image: based by default on the current resolution and extent, they can be customized and will resize the map extent (from the center). The size ratio can be locked, which may be particularly convenient when drawing the extent on the canvas.
5. ☒ *Draw active decorations*: in use *decorations* (scale bar, title, grid, north arrow...) are exported with the map
6. ☒ *Draw annotations* to export any *annotation*
7. ☒ *Append georeference information (embedded or via world file)*: depending on the output format, a world file of the same name (with extension PNGW for PNG images, JPGW for JPG, ...) is saved in the same folder as your image. The PDF format embeds the information in the PDF file.
8. When exporting to PDF, more options are available in the *Save map as PDF...* dialog:

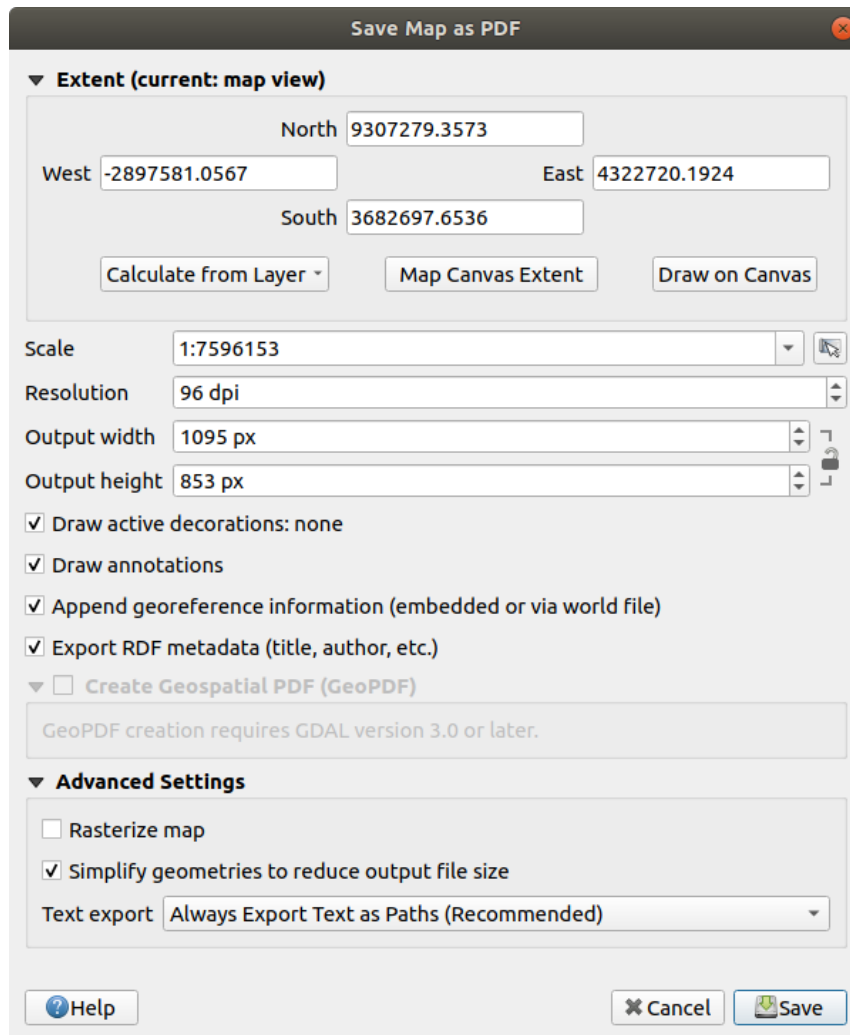


Abb. 11.21: The Save Map as PDF dialog

- ☒ *Export RDF metadata* of the document such as the title, author, date, description...
- ☐ *Create Geospatial PDF (GeoPDF)*: Generate a [georeferenced PDF file](#). You can:
  - Choose the GeoPDF *Format*
  - ☒ *Include vector feature information* in the GeoPDF file: will include all the geometry and attribute information from features visible within the map in the output GeoPDF file.

---

**Bemerkung:** A GeoPDF file can also be used as a data source. For more on GeoPDF support in QGIS, see <https://north-road.com/2019/09/03/qgis-3-10-loves-geopdf/>.

---

- *Rasterize map*
- ☒ *Simplify geometries to reduce output file size*: Geometries will be simplified while exporting the map by removing vertices that are not discernibly different at the export resolution (e.g. if the export resolution is 300 dpi, vertices that are less than 1/600 inch apart will be removed). This can reduce the size and complexity of the export file (very large files can fail to load in other applications).
- Set the *Text export*: controls whether text labels are exported as proper text objects (*Always export texts as text objects*) or as paths only (*Always export texts as paths*). If they are exported as text objects then they can be edited in external applications (e.g. Inkscape) as normal text. BUT the side effect is that the


rendering quality is decreased, AND there are issues with rendering when certain text settings like buffers are in place. That's why exporting as paths is recommended.

9. Click *Save* to select file location, name and format.

When exporting to image, it's also possible to *Copy to clipboard* the expected result of the above settings and paste the map in another application such as LibreOffice, GIMP...

## 11.2 3D-Ansicht

3D visualization support is offered through the 3D map view. You can create, manage and open 3D map views via *View ► 3D Map Views ►* menu:

1. By clicking on  *New 3D Map View* you can create a new 3D map view. A floating and dockable QGIS panel will appear (see *The 3D Map View dialog*). It has the same extent and view as the 2D main map canvas and provides a set of navigation tools to turn the view into 3D.
2. By clicking on *Manage 3D Map Views* you get in the 3D Map Views Manager. Here you get the ability to open, duplicate, remove and rename 3D map views.
3. If you created one or more 3D map views, you see them listed in *3D Map Views*. You can turn them on and off by clicking on. They will be saved by saving the project, even if they are turned off.

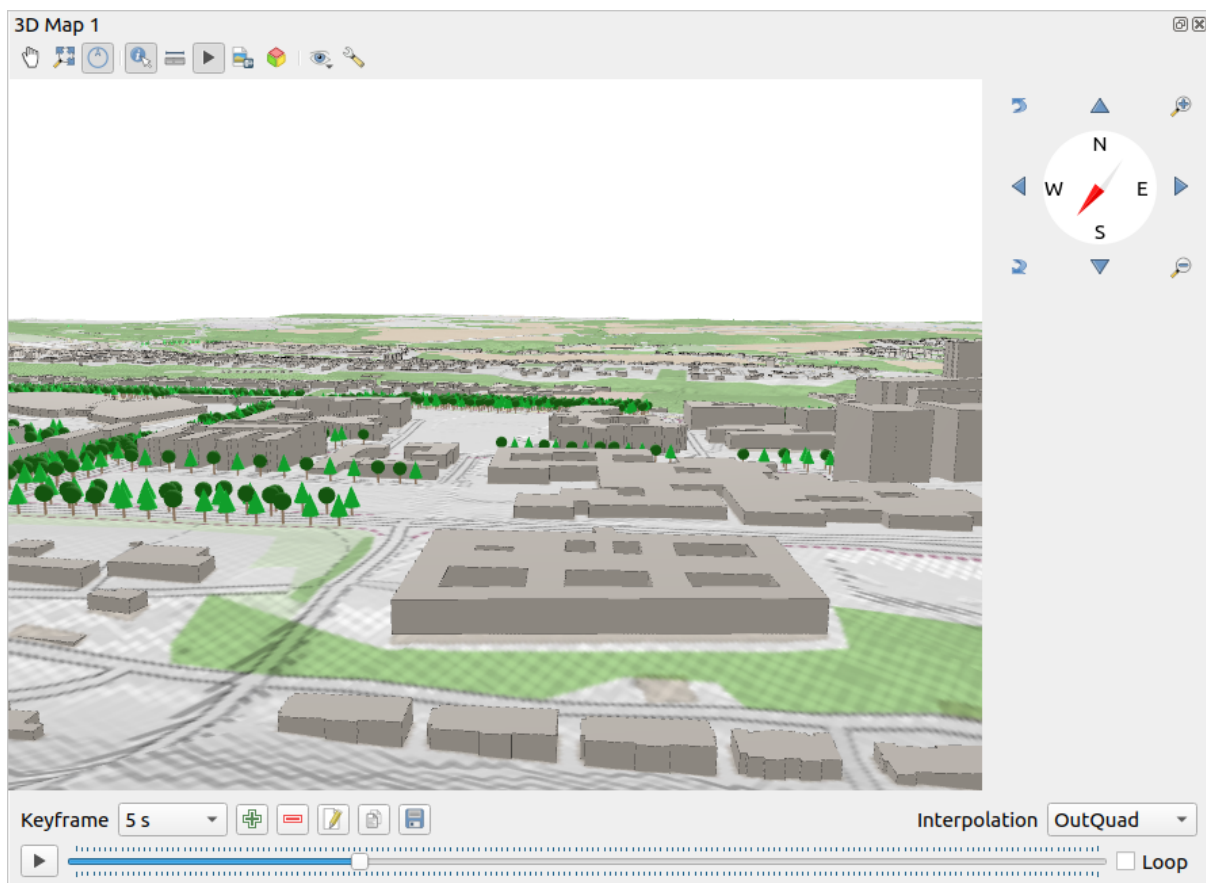



















Abb. 11.22: The 3D Map View dialog

The following tools are provided at the top of the 3D map view panel:

-  *Camera Control*: moves the view, keeping the same angle and direction of the camera

-  **Zoom Full**: resizes the view to the whole layers' extent, or the *reference extent* if set
-  **Toggle On-Screen Notification**: shows/hides the navigation widget (that is meant to ease controlling of the map view)
-  **Identify**: returns information on the clicked point of the terrain or the clicked 3D feature(s) – More details at *Identifying Features*
-  **Measurement Line**: measures the horizontal distance between points
-  **Animations**: shows/hides the *animation player* widget
-  **Save as Image...**: exports the current view to an image file format
-  **Export 3D Scene**: exports the current view as a 3D scene (.obj file), allowing post-processing in applications like Blender... The terrain and vector features are exported as 3D objects. The export settings, overriding the layers *properties* or map view *configuration*, include:
  - *Scene name* and destination *Folder*
  - *Terrain resolution*
  - *Terrain texture resolution*
  - *Model scale*
  -  *Smooth edges*
  -  *Export normals*
  -  *Export textures*
-  **Set View Theme**: Allows you to select the set of layers to display in the map view from predefined *map themes*.
- The  **Options** menu provides shortcuts to:
  - Add visual effects to the 3D rendering, such as showing *shadows*, *eye dome lighting* or *ambient occlusion*
  - Synchronize the views (*2D map view follows 3D camera* and/or *3D camera follows 2D Map view*)
  - *Show visible camera area in 2D map view*
  -  **Configure** the 3D map view *settings*.
-  **Dock 3D Map View**: switch from docked widget to top level window

### 11.2.1 Scene Configuration

The 3D map view opens with some default settings you can customize. To do so, expand the  **Options** menu at the top of the 3D canvas panel and press the  **Configure** button to open the *3D configuration* window.

In the 3D Configuration window there are various options to fine-tune the 3D scene:

## General

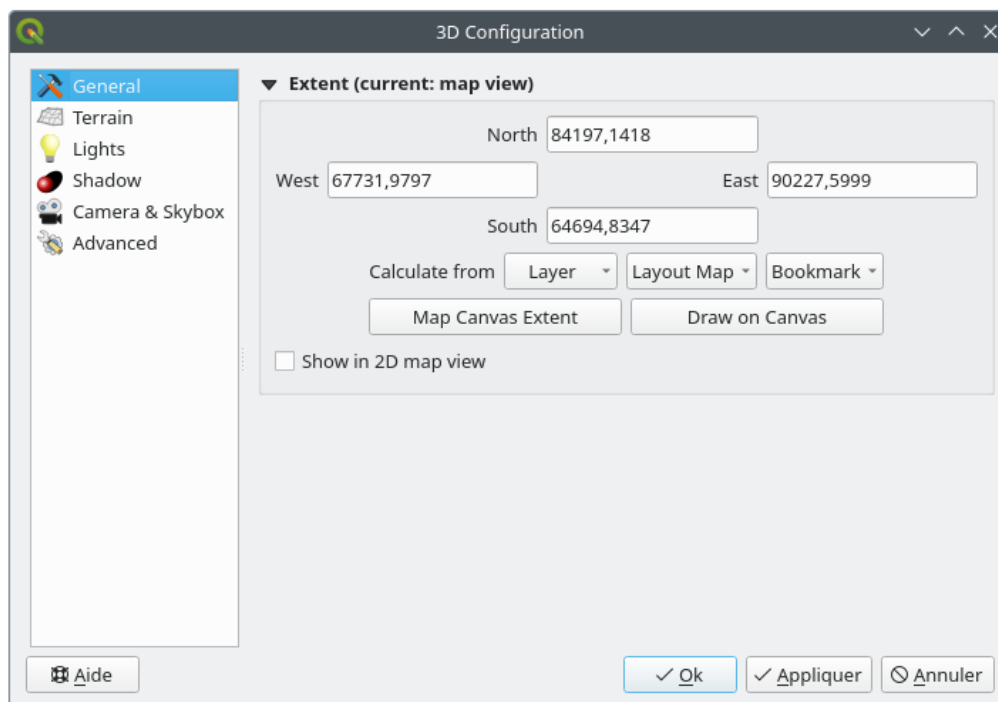




Abb. 11.23: The 3D Map General Configuration dialog

Under the  *General* tab, you can:

- Limit the 3D scene's contents to a 2D map extent, using the *spatial extent selector*: terrain and features are clipped at the specified extent and only the parts falling within the extent are loaded in the scene. Elevation range of the terrain, mesh and pointcloud layers is also taken into account so that the camera is not positioned below the scene's contents, particularly when using the terrain's vertical scale setting to exaggerate the elevation differences.

The set extent is also used as reference extent when pressing the  *Zoom full* button in 3D map view.

- Press  *Show in 2D map view* to display in the main map canvas a rubberband corresponding to the current extent of the 3D scene.

## Terrain

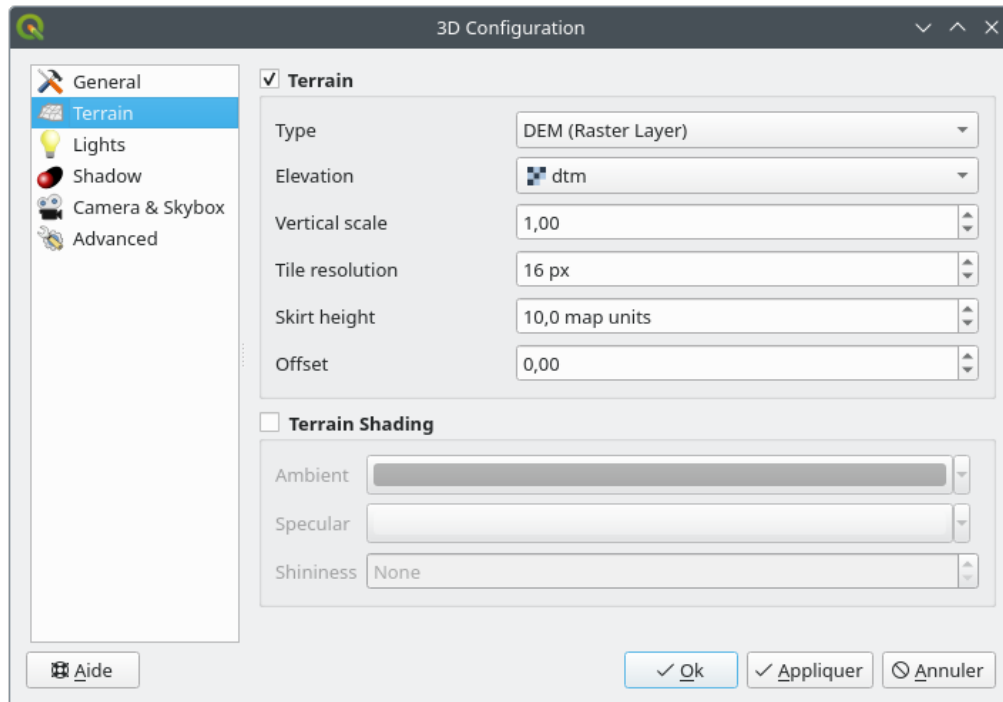


Abb. 11.24: The 3D Map Terrain Configuration dialog

- **Terrain:** Before diving into the details, it is worth noting that the terrain in a 3D view is represented by a hierarchy of terrain tiles and as the camera moves closer to the terrain, existing tiles that do not have sufficient details are replaced by smaller tiles with more details. Each tile has mesh geometry derived from the elevation raster layer and texture from 2D map layers.
  - The elevation terrain *Type* can be:
    - \* a *Flat terrain*
    - \* a loaded *DEM (Raster Layer)*
    - \* an *Online service*, loading *elevation tiles* produced by Mapzen tools – more details at <https://registry.opendata.aws/terrain-tiles/>
    - \* a loaded *Mesh* dataset
  - *Elevation:* Raster or mesh layer to be used for generation of the terrain. The raster layer must contain a band that represents elevation. For a mesh layer, the Z values of the vertices are used.
  - *Vertical scale:* Scale factor for vertical axis. Increasing the scale will exaggerate the height of the landforms.
  - *Tile resolution:* How many samples from the terrain raster layer to use for each tile. A value of 16px means that the geometry of each tile will consist of 16x16 elevation samples. Higher numbers create more detailed terrain tiles at the expense of increased rendering complexity.
  - *Skirt height:* Sometimes it is possible to see small cracks between tiles of the terrain. Raising this value will add vertical walls („skirts“) around terrain tiles to hide the cracks.
  - *Offset:* moves the terrain up or down, e.g. to adjust its elevation with respect to the ground level of other objects in the scene.

This can be useful when there is a discrepancy between the height of the terrain and the height of layers in your scene (e.g. point clouds which use a relative vertical height only). In this case adjusting the terrain

elevation manually to coincide with the elevation of objects in your scene can improve the navigation experience.

- When a mesh layer is used as terrain, you can configure the *Triangles settings* (wireframe display, smooth triangles, level of detail) and the *Rendering colors settings* (as a uniform color or *color ramp based*). More details in the *Mesh layer 3D properties* section.
- ☐ *Terrain shading*: Allows you to choose how the terrain should be rendered:
  - Shading disabled - terrain color is determined only from map texture
  - Shading enabled - terrain color is determined using Phong's shading model, taking into account map texture, the terrain normal vector, scene light(s) and the terrain material's *Ambient* and *Specular* colors and *Shininess*

## Lights

From the *Lights* tab, press the  menu to add

- up to eight *Point lights*: emits light in all directions, like a sphere of light filling an area. Objects closer to the light will be brighter, and objects further away will be darker. A point light has a set position (X, Y and Z), a *Color*, an *Intensity* and an *Attenuation*
- up to four *Directional lights*: mimics the lighting that you would get from a giant flash light very far away from your objects, always centered and that never dies off (e.g. the sun). It emits parallel light rays in a single direction but the light reaches out into infinity. A directional light can be rotated given an *Azimuth*, have an *Altitude*, a *Color* and an *Intensity*.

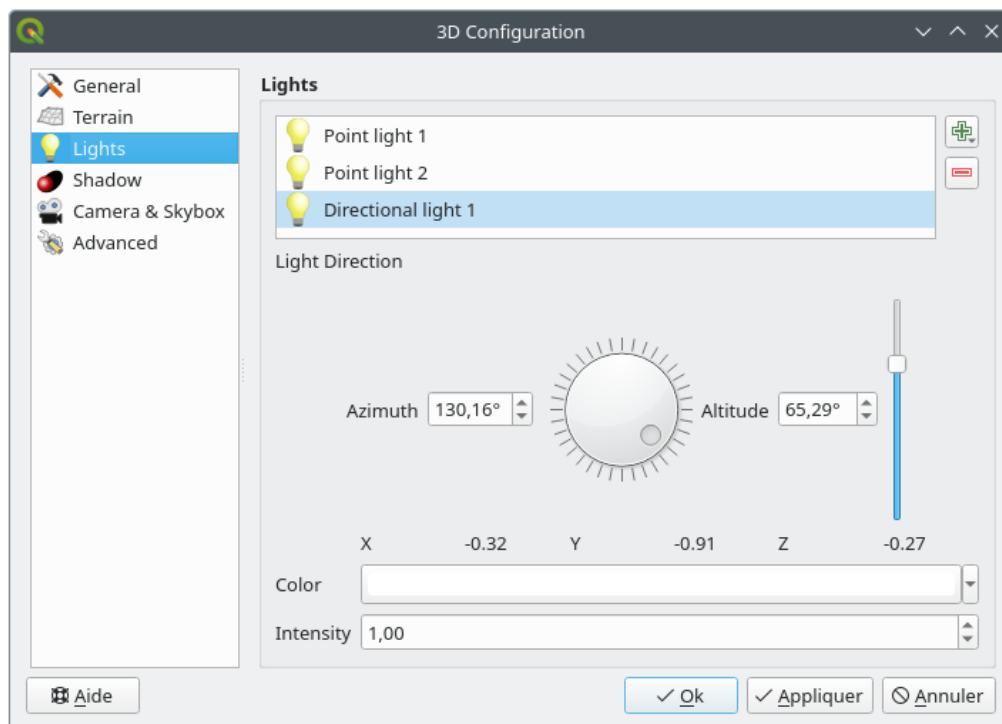


Abb. 11.25: The 3D Map Lights Configuration dialog



## Shadow

Check ☐ *Show shadows* to display shadows within your scene, given:

- a *Directional light*
- a *Shadow rendering maximum distance*: to avoid rendering shadow of too distant objects, particularly when the camera looks up along the horizon
- a *Shadow bias*: to avoid self-shadowing effects that could make some areas darker than others, due to differences between map sizes. The lower the better
- a *Shadow map resolution*: to make shadows look sharper. It may result in less performance if the resolution parameter is too high.

## Camera & Skybox

In this tab, you can control different parameters like camera, 3D axis, navigation synchronization and skybox.

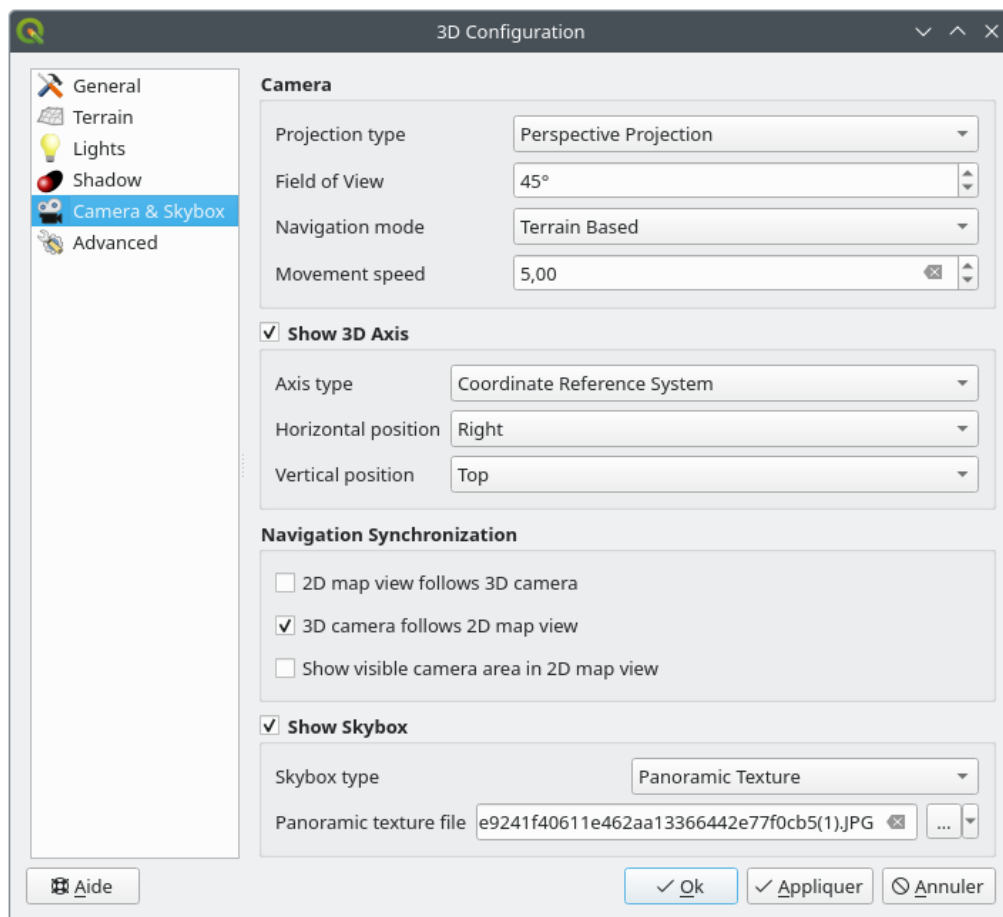


Abb. 11.26: The 3D Map Camera Configuration dialog

- The *Camera* parameter group overrides some *default camera settings* made in the *Settings ► Options ► 3D* dialog.
- Check ☐ *Show 3D Axis* to enable 3D axis tool. This parameter group allows to set the axis type and its position.
  - With the *Coordinate Reference System* type an orthogonal axis will be represented.

- With the *Cube* type, a 3D cube will be represented. The cube faces can be used to change the camera view: for example, click on the *north* face to set the camera to see from the north.

---

**Tipp:** Right-click the 3D axis to quickly set its position and type, and the camera view.

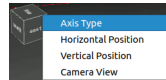


Abb. 11.27: The 3D Axis context menu

---

- The *Navigation Synchronization* parameter group adds options to synchronize 2D view with 3D camera position or 3D camera position with 2D view or bi directional synchronization. The last option displays the extent visible from the 3D camera over the 2D map view.
- Check ☐ *Show skybox* to enable skybox rendering in the scene. The skybox type can be:
  - *Panoramic texture*, with a single file providing sight on 360°
  - *Distinct faces*, with a texture file for each of the six sides of a box containing the scene

Texture image files of the skybox can be files on the disk, remote URLs or embedded in the project ([more details](#)).

### Advanced

- *Map tile resolution*: Width and height of the 2D map images used as textures for the terrain tiles. 256px means that each tile will be rendered into an image of 256x256 pixels. Higher numbers create more detailed terrain tiles at the expense of increased rendering complexity.
- *Max. screen error*: Determines the threshold for swapping terrain tiles with more detailed ones (and vice versa) - i.e. how soon the 3D view will use higher quality tiles. Lower numbers mean more details in the scene at the expense of increased rendering complexity.
- *Max. ground error*: The resolution of the terrain tiles at which dividing tiles into more detailed ones will stop (splitting them would not introduce any extra detail anyway). This value limits the depth of the hierarchy of tiles: lower values make the hierarchy deep, increasing rendering complexity.
- *Zoom levels*: Shows the number of zoom levels (depends on the map tile resolution and max. ground error).
- ☐ *Show labels*: Toggles map labels on/off
- ☐ *Show map tile info*: Include border and tile numbers for the terrain tiles (useful for troubleshooting terrain issues)
- ☐ *Show bounding boxes*: Show 3D bounding boxes of the terrain tiles (useful for troubleshooting terrain issues)
- ☐ *Show camera's view center*
- ☐ *Show camera's rotation center*
- ☐ *Show light sources*: shows a sphere at light source origins, allowing easier repositioning and placement of light sources relative to the scene contents
- ☐ *Show frames per second (FPS)*
- ☐ *Show debug overlay*: visual overlay which displays some useful debugging and profiling information. This allows in particular to quickly see the frame graph and the scene graph.

- ☐ *Show Eye Dome Lighting (EDL)*: a post processing effect which enhances depth perception. Each pixel's depth (distance off the camera) is compared to its neighboring pixels' depth and gets highlighted according to that depth difference, making the edges stand out. Affects the whole scene and can be combined with *Screen Space Ambient Occlusion*. Following parameters can be controlled:
  - *Lighting strength*: increases the contrast, allowing for better depth perception
  - *Lighting distance*: represents the distance of the used pixels off the center pixel and has the effect of making edges thicker.
- Add screen-space ☐ *Ambient Occlusion (SSAO)*: a post processing effect which also enhances depth perception by applying a darker shading to areas which are less exposed to ambient lighting. Affects the whole scene and can be combined with *Eye dome Lighting*. Following parameters can be controlled:
  - *Radius*: how far we will reach to calculate ambient occlusion
  - *Intensity*: how strong the effect should be (higher values make things darker)
  - *Occlusion threshold*: how many neighboring points need to be occluded for the effect to appear (lower values than 50% will make the output darker, but possibly providing greater range of occlusion)

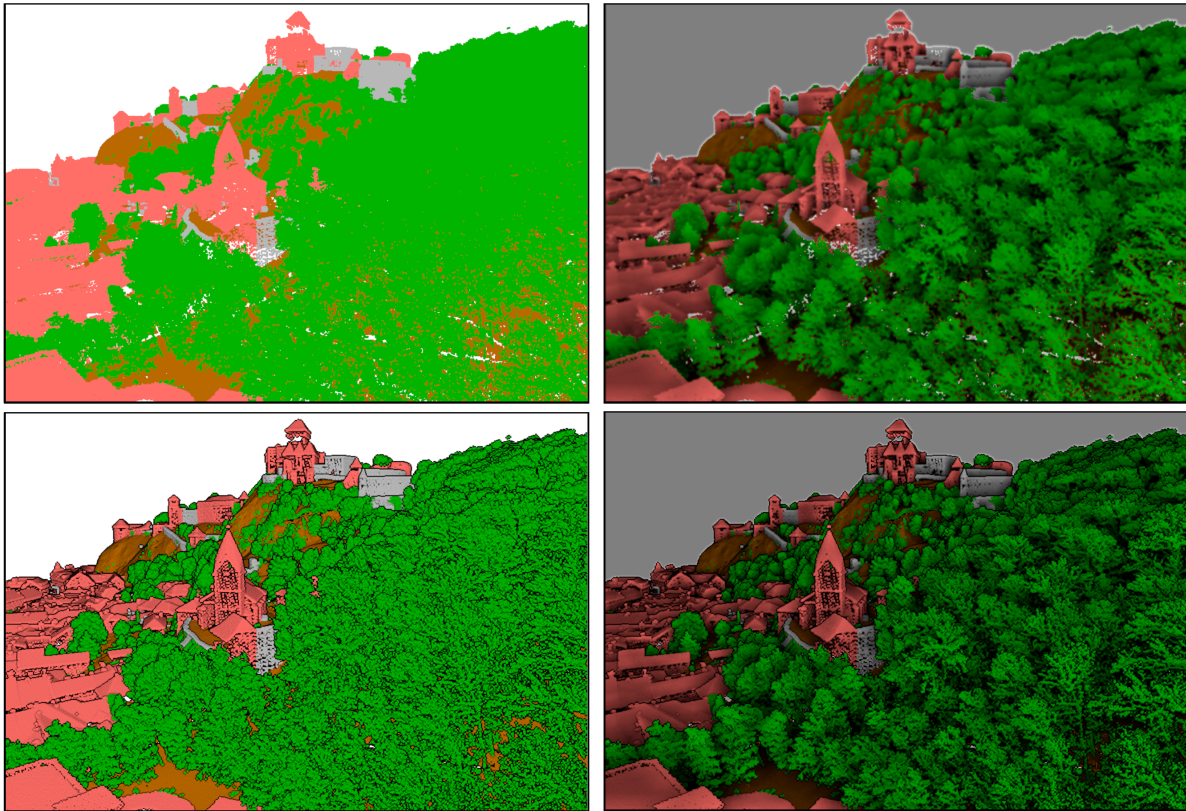







Abb. 11.28: Rendering Point clouds in 3D map using Eye Dome Lighting (EDL) and/or Screen-Space Ambient Occlusion (SSAO)


From top, left to right: No effect – SSAO only – EDL only – SSAO and EDL

- ☐ *Debug Shadow Map*: renders the scene as a red-black image from the point of view of the light used for shadows (for troubleshooting). The widget is set with a proportional *Size* to the 3D map view's, and docked in a *Corner*.
- ☐ *Debug Depth Map*: renders the scene's depth map as an image with nearer pixels being darker (for troubleshooting). The widget is set with a proportional *Size* to the 3D map view's, and docked in a *Corner*.

## 11.2.2 Navigation options




To explore the map view in 3D:


- Tilt the terrain (rotating it around a horizontal axis that goes through the center of the window)
  - Press the  Tilt up and  Tilt down tools
  - Press `Shift` and use the up/down keys
  - Ziehen Sie die Maus bei gedrückter mittlerer Maustaste vorwärts/rückwärts.
  - Drücken Sie `Shift` und ziehen Sie die Maus mit gedrückter linker Maustaste vorwärts/rückwärts.
- Rotate the terrain (around a vertical axis that goes through the center of the window)
  - Turn the compass of the navigation widget to the watching direction
  - Press `Shift` and use the left/right keys
  - Drag the mouse right/left with the middle mouse button pressed
  - Press `Shift` and drag the mouse right/left with the left mouse button pressed
- Change the camera position (and the view center), moving it around in a horizontal plan
  - Drag the mouse with the left mouse button pressed, and the  Camera control button enabled
  - Press the directional arrows of the navigation widget
  - Use the up/down/left/right keys to move the camera forward, backward, right and left, respectively
- Change the camera altitude: press the `Page Up`/`Page Down` keys
- Change the camera orientation (the camera is kept at its position but the view center point moves)
  - Press `Ctrl` and use the arrow keys to turn the camera up, down, left and right
  - Press `Ctrl` and drag the mouse with the left mouse button pressed
- Zoom in and out
  - Press the corresponding  Zoom In and  Zoom Out tools of the navigation widget
  - Scroll the mouse wheel (keep `Ctrl` pressed results in finer zooms)
  - Drag the mouse with the right mouse button pressed to zoom in (drag down) and out (drag up)


To reset the camera view, click the  Zoom Full button on the top of the 3D canvas panel.

## 11.2.3 Creating an animation

An animation is based on a set of keyframes - camera positions at particular times. To create an animation:

1. Toggle on the  Animations tool, displaying the animation player widget
2. Click the  Add keyframe button and enter a *Keyframe time* in seconds. The *Keyframe* combo box now displays the time set.
3. Using the navigation tools, move the camera to the position to associate with the current keyframe time.
4. Repeat the previous steps to add as many keyframes (with time and position) as necessary.
5. Click the  button to preview the animation. QGIS will generate scenes using the camera positions/rotations at set times, and interpolating them in between these keyframes. Various *Interpolation* modes for animations are available (eg, linear, inQuad, outQuad, inCirc... – more details at <https://doc.qt.io/qt-5/qeasingcurve.html#EasingFunction-typedef>).

The animation can also be previewed by moving the time slider. Keeping the *Loop* box checked will repeatedly run the animation while clicking  stops a running animation.


Click  Export animation frames to generate a series of images representing the scene. Other than the filename *Template* and the *Output directory*, you can set the number of *Frames per second*, the *Output width* and *Output height*.

### 11.2.4 3D vector layers

A vector layer with elevation values can be shown in the 3D map view by checking *Enable 3D Renderer* in the *3D View* section of the vector layer properties. A number of options are available for controlling the rendering of the 3D vector layer.

## 11.3 Elevation Profile View

The *Elevation Profile* panel is a plotting tool for side view, for visualizing elevation data along a line. It supports vector, raster, mesh and point cloud layers. Data can be of 2D or 3D type.

To add an elevation profile view, go to *View* ►  *Elevation Profile* menu. You can add as many profile views as you want, and they can be docked, piled on top of each other, or floating.

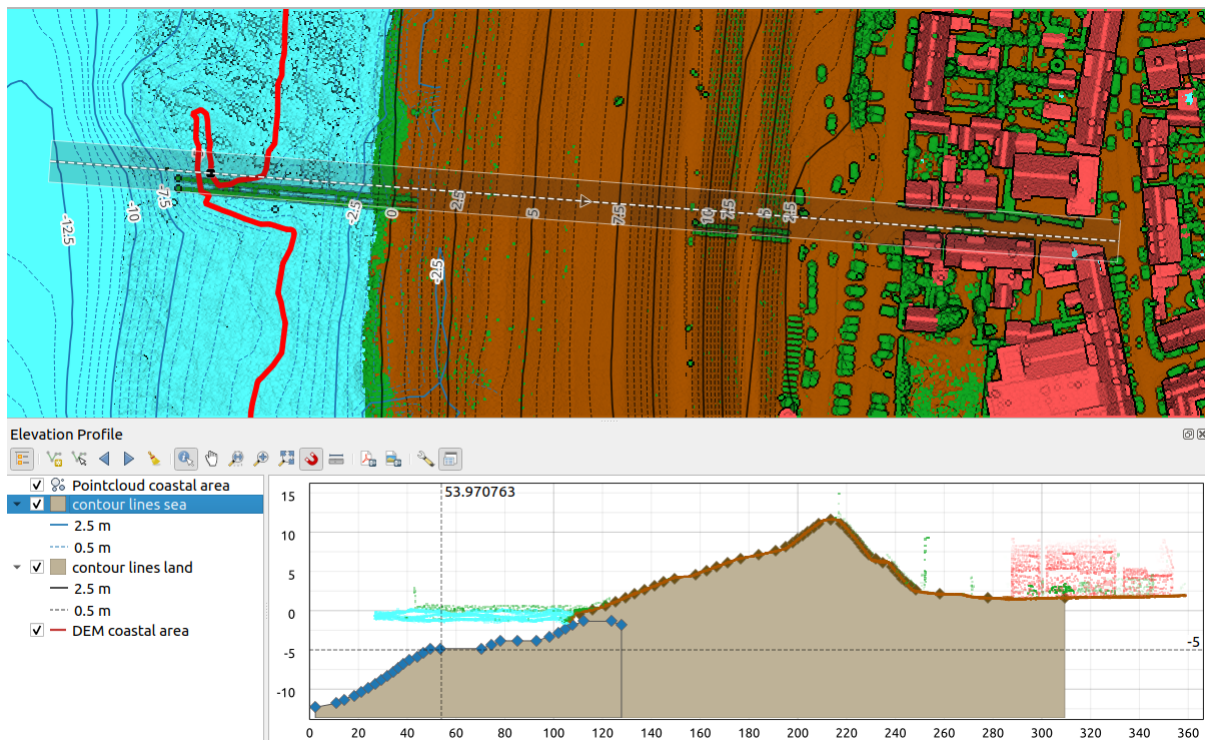


Abb. 11.29: Elevation Profile dialog embedded below main map canvas




















Figure shows a coastline with a town behind a sea-dike. A larger tolerance applied to the elevation profile line returns more point cloud features.



### 11.3.1 The interface


At the top of the *Elevation Profile* panel, a toolbar provides you with the following tools:


Tab. 11.4: Elevation Profile View toolbar

Tool	Shortcut	Description
 Add Layers	hold Ctrl while drag-and-drop	Allows you to add new layers to a plot.
 Show Layer Tree		Shows or hides a list of project layers to configure rendering in the profile view.
 Capture Curve		Draws interactively a line over the map canvas to represent the profile curve.
 Capture Curve From Feature		Picks an existing line feature on the map canvas and generate a profile curve along that line.
 Nudge Left	Ctrl+Alt	Allows you to slowly move the capture line across the map to the left (e.g. to find the optimal profile line based on the elevation).
 Nudge Right	Ctrl+Alt	Allows you to slowly move the capture line across the map to the right (e.g. to find the optimal profile line based on the elevation).
 Clear		Removes the profile line and any plot displayed in the <i>Elevation Profile</i> view
 Identify Features		Identifies features in the plot canvas via either a single click, or click-and-drag rectangle. Results are shown in the standard <i>Identify Results</i> dock.
 Pan	Space	Click and drag to pan the plot canvas. Can also be done with the middle mouse button.
 Zoom X Axis		Zooms in/out along the horizontal axis, keeping the vertical ratio
 zoom	Ctrl+Shift	Click or click-and-drag a rectangle over the plot to zoom in. Press Alt and click to instead zoom out.
 Zoom Full		Zooms the <i>Elevation Profile</i> view to the extent of the capture line
 Enable Snapping		Allows to snap to the edge or vertex of the plot features in the profile view. Convenient for accurate retrieval of coordinates or distance measurements.
 Measure Distances		Measures horizontal and vertical distances
 Export As PDF		Exports plots to PDF (as high quality vector objects)
 Export As Image		Exports plots to several image formats
 Export Results		Exports plots as 3D features, 2D Profile and Distance/Elevation Table. (The results can be saved as DXF files, CSV files or any of the standard writable vector spatial formats)
 Options		Provides access to configuration settings of the profile elevation line.
 Lock distance/elevation scales		Ensures that the horizontal and vertical scales are always kept equal (so that e.g. a 45° slope will appear as a 45° slope in the profile view).
► Distance units		Allows to render distances in the profile chart with units other than the map canvas units.
► Tolerance		Sets how far from the actual profile line a point can reside within to be included in the results. Uses the map units and is ignored by other layer or geometry types.

Fortsetzung auf der nächsten Seite

Tab. 11.4 – Fortsetzung der vorherigen Seite

Tool	Shortcut	Description
 Dock Elevation Profile View		Switch between docked and floating status of the view




In the bottom left, a copy of the *Layers* panel can be displayed pushing the  *Show Layer Tree* button. This is however an independent widget, with its own set of visible layers, in a custom stack order. It allows you to control layers rendering and behavior within the plot canvas:

- Tick the box next to the layer name to set whether it should be rendered in the plot canvas
- Drag-and-drop layers up or down to change the order of the layers
- style rendering of layers in the profile view: double-click a layer or right-click and select *Properties...* to open the layer's *Elevation* properties tab for configuration. A summary of elevation settings is displayed as tooltip when hovering over the layer.

On the right of the layer tree, the plot canvas is the main place you can preview the elevation profile of the enabled layers. It is based on a graduated grid in which the horizontal axis displays the length of the profile line and the vertical axis displays the Z elevation of the observed features. It also allows a set of interactions such as zooming, panning, measuring, identifying features, ... using the tools at the top.



### 11.3.2 Creating an elevation profile

To create a profile view, you can:

1. Go to *View* ►  *Elevation Profile* menu. The *Elevation profile* panel opens.
2. Create the profile line along which the terrain and the features will be rendered. Select a drawing tool:
  -  *Capture Curve*: click left over the main map canvas to add vertices and click right to finalize a line that will be used as profile line
  - or  *Capture Curve From Feature*: click a line feature on the map canvas to select it as the profile line. If multiple features are present at the clicked point then a popup menu will appear allowing you to select among them.

All the line digitizing capabilities such as the *snapping options*, *tracing*, *digitizing techniques* or the advanced *digitizing panel* are available for use.

The plot canvas may start rendering some features.

3. The next step is to configure the elevation properties of the layers you want to visualize.
  1. Push on the  *Show Layer Tree* button to display the list of layers. By default, not all the layers of the project are loaded and referenced in the profile tool; e.g., raster layers which are not marked as having elevation data in their properties are ignored. Clicking the  *Add layers* button will show a filtered list of possible layers which can be added to the plot, but which currently aren't in the plot. Applying the dialog with selected layers will automatically mark them as having elevation data and immediately add them to the plot.

---

**Tipp:** Holding **Ctrl** key, it is possible to drag and drop additional layers from the main layer tree to one in the elevation profile tool.


---

2. Toggle visibility of the layers you are interested in. These are the only ones rendered in the profile view and selected layers can be different from the main *Layers* panel's.

3. Double-click a layer name or right-click and select *Properties*. The *Elevation* properties tab of the layer opens. This is the place you configure how each feature or terrain should render on the profile view. Available properties depend on the layer type:

- *Raster Elevation Properties*
- *Vector Elevation Properties*
- *Point Cloud Elevation Properties*
- *Mesh Elevation Properties*

The profile view starts rendering terrain or features of active layers crossing the given profile curve, as soon as they have elevation properties configured.

4. Under  Options drop-down menu, you can set the *Tolerance* value. This value is used to create a flat buffer around the elevation profile line, visible in the main map canvas. Any visible point feature overlapping that buffer will be captured in the plot canvas.

### 11.3.3 Interacting with the profile Views







When an elevation profile line is created, the plot canvas zooms to its full extent. On the X-axis you can see the length of the profile and, on the Y-axis the height range between minimum and maximum height captured, both in map units.

When you move the mouse pointer in the elevation profile view, you will see two crossing dot lines:




- the vertical line shows the height information
- the horizontal line shows the distance from the beginning of the elevation profile Line

When you move the mouse pointer in the elevation profile canvas, you can also see a black dot move along the elevation profile line on the main map canvas. At the middle of the profile line, an arrow indicates its direction.




As for the main map canvas, QGIS provides means to navigate on the plot canvas:

-  Pan is used to move the elevation profile extent in any direction you want. Holding **Space** key while moving the mouse also shifts the plot canvas extent.
-  Zoom X Axis is used to zoom in along the horizontal axis, keeping the ratio of the vertical axis (the elevation) the same. Left click to stretch out the plot along the X axis, with the clicked point at the center of the axis, or drag a rectangle to stretch out the plot along the X axis to the rectangle width. Hold **Alt** while using  Zoom X Axis to zoom out along that axis.
-  Zoom is used to zoom in on a certain point (using left click), or to a certain extent (dragging a rectangle on the area). Hold **Alt** while using  Zoom to instead zoom out. In combination with the **Ctrl** key you can zoom in or out more smoothly.
-  Zoom Full is the default zoom level used at the beginning and shows the full extent of the profile line, with all returned features. Use it to reset the zoom level.

It is also possible to interact with the elements displayed in the plot canvas:

- Press  Enable snapping button to accurately catch points, vertices or edges of the features, for an accurate measurement or coordinates report.
-  Identify Features is used to identify features of the visible layers in the layer tree. You can drag a rectangle across several features in the profile view to query all of them. When compatible with the format (e.g. vector, point cloud), these features will be highlighted in the main map canvas.
-  Measurement Distances: click or select two points in the plot canvas to report the horizontal *Distance*, the *Elevation* and the *Total distance* between them, in map units.



-  Nudge Left and  Nudge right are used to shift the position of the elevation profile line in the map canvas to its left or right. The plot canvas will be redrawn, showing features and terrain overlapping the profile line buffer. The line is moved sideways using the *Tolerance* value in  Options menu.

**Warnung:** Closing an elevation profile view or the project currently removes the view from the project.

For more details, give a look to [QGIS elevation profile/cross section tool – a deep dive!](#), a presentation done by Nyall Dawson.



### 12.1 Hilfe

Wann immer Sie Hilfe zu einem bestimmten Thema benötigen, können Sie die entsprechende Seite im aktuellen Benutzerhandbuch über die Schaltfläche :guilabel: *Hilfe* aufrufen, die in den meisten Dialogen verfügbar ist — bitte beachten Sie, dass Plugins von Drittanbietern auf spezielle Webseiten verweisen können.

### 12.2 Bedienfelder

Standardmäßig stellt QGIS viele Panels zur Verfügung, mit denen Sie arbeiten können. Einige dieser Panels werden im Folgenden beschrieben, während andere in verschiedenen Teilen des Dokuments zu finden sind. Eine vollständige Liste der von QGIS bereitgestellten Standardpanels finden Sie über die *View ► Panels ►* Menü und unter *Bedienfelder*.

#### 12.2.1 Layerfenster

The *Layers* panel (also called the map legend) lists all the layers in the project and helps you manage their visibility and shape the map. You can show or hide the panel by pressing `Ctrl+1`.

QGIS provides a variety of ways to add layers to a project:

- using the *Add* button from the dedicated data provider tab in the *Data source manager* dialog
- from QGIS *Browser panel* or *DB Manager*: double-click, drag-and-drop files and layers onto QGIS or use the contextual menu
- drag-and-drop files from the Operating System files explorer onto QGIS

In all these scenarios, you can open one or many layers at a time. New layers are added to the *Layers* panel:

1. if dropped over the *Layers* panel, at the exact location they are released
2. in the other cases, at a location that respects the global *behavior used when adding new layers* setting
3. and in case of multiple layers, they are sorted in a way that increases the chance of their stacking being logical and features being visible as most as possible, using the following logic (top to bottom):
  - vector point layers
  - vector line layers

- vector polygon layers
- point cloud layers
- mesh layers
- raster layers

At the top of the *Layers* panel, a toolbar allows you to:









-  Open the layer styling dock (F7): toggle the *Layer Styling* panel on and off.
-  Add new group: see *Interact with groups and layers*
-  Manage Map Themes: control visibility of layers and arrange them in different *map themes*.
-  filter layers in the legend tree:
  - *Filter Legend by Map Content*: only the layers that are set visible and whose features intersect the current map canvas have their style rendered in the layers panel. Otherwise, a generic NULL symbol is applied to the layer. Based on the layer symbology, this is a convenient way to identify which kind of features from which layers cover your area of interest.
  - *Show Private Layers*: a convenient shortcut to display and interact with *private layers* in the *Layers* panel without modifying the project settings.
-  Filter Legend by Expression: apply an expression to remove styles from the selected layer tree that have no feature satisfying the condition. This can be used to highlight features that are within a given area/feature of another layer. From the drop-down list, you can edit and clear the expression currently applied.
-  Expand All or  Collapse All layers and groups in the layers panel.
-  Remove Layer/Group currently selected.




Abb. 12.1: Layer Toolbar in Layers Panel






---

**Bemerkung:** Tools to manage the layers panel are also available for map and legend items in print layouts


---

### Kartenthemen konfigurieren

Die  Manage Map Themes drop-down Schaltfläche bietet einen Zugang zu den Einstellungen zur Sichtbarkeit der Layer im *Layers* Panel:

-  *Alle Layer anzeigen*
-  *Alle Layer ausblenden*
-  *Show Selected Layers*
-  *Hide Selected Layers*
-  *Toggle Selected Layers*: changes the visibility of the first selected layer in the panel, and applies that state to the other selected layers. Also accesible through *Space* shortcut.
- *Toggle Selected Layers Independently*: changes the visibility status of each selected layer

-  *Hide Deselected Layers*

Beyond the simple control of layer visibility, the  *Manage Map Themes* menu allows you to configure **Map Themes** in the legend and switch from one map theme to another. A map theme is a **snapshot** of the current map legend that records:

- the layers set as visible in the *Layers* panel
- **and** for each visible layer:
  - the reference to the *style* applied to the layer
  - the visible classes of the style, ie the layer checked node items in the *Layers panel*. This applies to *symbolologies* other than the single symbol rendering
  - the collapsed/expanded state of the layer node(s) and the group(s) it's placed inside


To create a map theme:

1. Check a layer you want to show
2. Configure the layer properties (symbolology, diagram, labels...) as usual
3. Expand the *Style* ► menu at the bottom and click on *Add...* to store the settings as *a new style embedded in the project*

---

**Bemerkung:** A map theme does not remember the current details of the properties: only a reference to the style name is saved, so whenever you apply modifications to the layer while this style is enabled (eg change the symbology rendering), the map theme is updated with new information.

---

4. Repeat the previous steps as necessary for the other layers
5. If applicable, expand or collapse groups or visible layer nodes in the *Layers* panel
6. Click on the  *Manage Map Themes* button on top of the panel, and *Add Theme...*
7. Enter the map theme's name and click *OK*

The new theme is listed in the lower part of the  drop-down menu.

You can create as many map themes as you need: whenever the current combination in the map legend (visible layers, their active style, the map legend nodes) does not match any existing map theme contents as defined above, click on *Add Theme...* to create a new map theme, or use *Replace Theme* ► to update a map theme. You can rename the active map theme with *Rename Current Theme...* or use the *Remove Current Theme* button to delete it.

Map themes are helpful to switch quickly between different preconfigured combinations: select a map theme in the list to restore its combination. All configured themes are also accessible in the print layout, allowing you to create different map items based on specific themes and independent of the current main canvas rendering (see *Map item layers*).

## Overview of the context menu of the Layers panel

At the bottom of the toolbar, the main component of the Layers panel is the frame listing all the layers added to the project, optionally organized in groups. A layer with a checked box next to it displays its contents overlapping the map canvas extent, unless a *scale-based visibility* is set. A layer can be selected and dragged up or down in the legend to change the Z-ordering. Z-ordering means that layers listed nearer the top of the legend are drawn over layers listed lower down in the legend. Also a layer or a group of layers can be dragged across several QGIS instances.
































































































































---

**Bemerkung:** The Z-ordering behavior can be overridden by the *Layer Order* panel.

---

Depending on the item selected in the panel, a right-click shows a dedicated set of options presented below.

Tab. 12.1: Contextual menus from *Layers* panel items

Option	Group	Vektorlayer	Rasterlayer	Mesh Layer	Point Cloud Layer	3D Layer
 Zoom to Layer(s)/Group						
 Zoom to Selection						
 In der Übersicht anzeigen						
Show Feature Count						
 Show Label						
Copy Layer/Group						
Rename Layer/Group						
 Zoom to Native Resolution (100%)						
Stretch Using Current Extent						
 Update SQL Layer...						
 Edit Virtual Layer...						
 Add Group						
 Duplicate Layer						
 Remove Layer/Group...						
Move Out of Group						
Move to Top						
Move to Bottom						
Check and all its Parents						
Group Selected						
 Attributtabelle öffnen						
 Bearbeitungsstatus umschalten						
 Aktuelle Änderungen ►						
Filter...						
Change Data Source...						
Repair Data Source...						
Actions on selections ► (in edit mode)						
► Duplicate Feature						
► Duplicate Feature and Digitize						
Set Layer Scale Visibility...						
Zoom to Visible Scale						
Layer CRS ►						
► Set Project CRS from Layer						
► Set to.. (recent CRSs)						
► Set Layer CRS...						
Set Group CRS...						
Set Group WMS Data...						
 Mutually Exclusive Group						

Fortsetzung auf der nächsten Seite

Tab. 12.1 – Fortsetzung der vorherigen Seite

Option	Group	Vektorlayer	Rasterlayer	Mesh Layer	Point Cloud Layer	3D Layer
<i>Check and all its children (Ctrl-click)</i>						
<i>Uncheck and all its children (Ctrl-click)</i>						
<i>Make Permanent</i>						
<i>Export ►</i>						
► <i>Save As...</i>						
► <i>Save Features As...</i>						
► <i>Save Selected Features As...</i>						
► <i>Save As Layer Definition File...</i>						
► <i>Save As QGIS Layer Style File...</i>						
<i>Styles ►</i>						
► <i>Copy Style</i>						
► <i>Paste Style</i>						
► <i>Add...</i>						
► <i>Rename Current...</i>						
► <i>Edit symbol...</i>						
► <i>Copy Symbol</i>						
► <i>Paste Symbol</i>						
<i>Add Layer Notes...</i>						
<i>Edit Layer Notes...</i>						
<i>Remove Layer Notes</i>						
<i>Eigenschaften...</i>						

For GRASS vector layers, Toggle editing is not available. See section [Digitalisieren und Editieren eines GRASS Vektorlayers](#) for information on editing GRASS vector layers.

## Interact with groups and layers

Layers in the legend window can be organized into groups. There are different ways to do this:

1. Press the icon to add a new group. Type in a name for the group and press **Enter**. Now click on an existing layer and drag it onto the group.
2. Select more than one layer, then press the icon. The selected layers are automatically inserted into the new group.
3. Select some layers, right-click in the legend window and choose *Group Selected*. The selected layers will automatically be placed in a new group.

To move a layer out of a group, drag it out, or right-click on it and choose *Move Out of Group*: the layer is moved from the group and placed above it. Groups can also be nested inside other groups. If a layer is placed in a nested group, *Move Out of Group* will move the layer out of all nested groups.

To move a group or layer to the top of the layer panel, either drag it to the top, or choose *Move to Top*. If you use this option on a layer nested in a group, the layer is moved to the top in its current group. The *Move to Bottom* option follows the same logic to move layers and groups down.

The checkbox for a group will show or hide the checked layers in the group with one click. With **Ctrl** pressed, the checkbox will also turn on or off all the layers in the group and its sub-groups.

Ctrl-click on a checked / unchecked layer will uncheck / check the layer and all its parents.













Enabling the **Mutually Exclusive Group** option means you can make a group have only one layer visible at the same time. Whenever a layer within the group is set visible the others will be toggled not visible.

It is possible to select more than one layer or group at the same time by holding down the Ctrl key while clicking additional layers. You can then move all selected layers to a new group at the same time.

You may also delete more than one layer or group at once by selecting several items with the Ctrl key and then pressing Ctrl+D: all selected layers or groups will be removed from the layers list.

### More information on layers and groups using indicator icon



In some circumstances, formatting or icons next to the layer or group in the *Layers* panel change to give more information about the layer/group. These elements are:

-  to indicate that the layer is in edit mode and you can modify the data
-  to indicate that the layer being edited has some unsaved changes
-  to indicate *a filter* applied to the layer. Hover over the icon to see the filter expression and double-click to update the query
-  to identify layers that are *required* in the project, hence non removable
-  to identify an *embedded group or layer* and the path to their original project file
-  to identify a layer whose data source was not available at the project file opening (see *Behandlung von fehlerhaften Dateipfaden*). Click the icon to update the source path or select *Repair Data Source...* entry from the layer contextual menu.
-  to remind you that the layer is a *temporary scratch layer* and its content will be discarded when you close this project. To avoid data loss and make the layer permanent, click the icon to store the layer in any of the GDAL vector formats supported by QGIS.
-  to identify a layer used in *offline editing mode*.
-  to identify a layer that has no/unknown CRS
-  for layers with coordinates stored in a coordinate reference system which is inherently low accuracy (requires the *corresponding setting* to be enabled)
-  to identify a temporal layer controlled by canvas animation
-  to identify a layer that has *notes* associated
- A grayed name, when the map canvas current scale is outside the layer's visibility scale range (as set in its *Rendering* properties). Select the contextual menu *Zoom to Visible Scale* option to zoom the map to the layer's nearest visibility scale bound.



## Control layers rendering through grouping

Groups are a means of structuring layers within a tree in the project but they can also impact how their component layers are rendered, namely as a single flattened object during map renders.

The option for such a rendering is available within the *Layer Styling* panel whenever a group is selected. Under the  **Symbology** tab, check  *Render Layers as a Group* to enable a set of options to control the appearance of the child layers as a whole, instead of individual layers:

- *Opacity*: Features from child layers which are obscured by other child layers remain obscured, and the opacity applies to the „whole of group“ only.

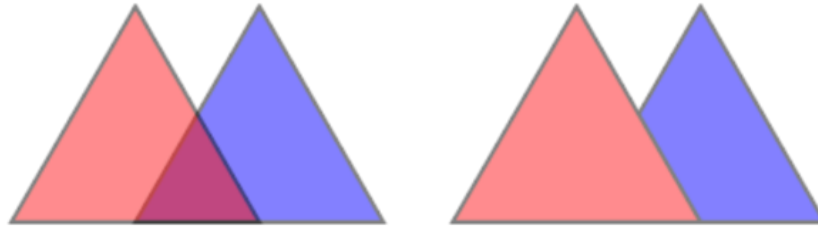


Abb. 12.2: Setting opacity on layers vs on a group

The image on the left shows two layers being rendered at 50% opacity (underlying features are visible, but semi-masked by the 50% red feature on top). The second image shows the result of setting the opacity on the group (parts of the blue underlying child layer is completely obscured by the red layer on top and then the result is rendered at 50% opacity).

- *Blend modes*: Just like opacity, setting a *blend mode* (like multiply, overlay, ...) for an entire group results first in flattening features of child layers, with upper ones obscuring lower. The rendering is then obtained by blending the flat group and the layers sitting below the group.
  - When the child layers have blend modes assigned, it is applied before flattening but the scope is restricted to only affecting other child layers from that group, and not other layers sitting below the whole group.
  - Some more *blending modes* options are available for child layers in groups, in their *Symbology* tab which perform „clipping“ style operations on other child layers during the render. You can e.g. clip the render of one layer's content by the content in a second „mask“ layer.
- *Layer effects*: applies *effects* only to the flattened render of the child layers; So e.g. a drop shadow effect applied to the group would not be visible for obscured child layers.

When a group is set to *Render layers as a group*, then only the group will be shown in the *Layer Order* panel list. Group children will not be visible in this order list, as their ordering is determined by the placement of the group layer.

## Editing layer style

From the *Layers* panel, you have shortcuts to change the layer rendering quickly and easily.

Right-click on a layer and select *Styles* ► in the list in order to:

- see the *styles* currently available for the layer. If you defined many styles for the layer, you can switch from one to another and your layer rendering will automatically be updated on the map canvas.
- copy part or all of the current style, and when applicable, paste a copied style from another layer
- *Rename current...* style
- *Add* a new style (which is actually a copy of the current one)
- or *Remove current* style (only when multiple styles are available).

---

**Tipp: Quickly share a layer style**



From the context menu, copy the style of a layer and paste it to a group or a selection of layers: the style is applied to all the layers that are of the same type (vector, raster, mesh, point cloud, ...) as the original layer and, for vector layers, have the same geometry type (point, line or polygon).

---

When using a symbology based on features classification (e.g. *categorized*, *graduated* or *rule-based* for vector layers, or *classification* for point clouds), right-clicking a class entry in the *Layers* panels makes it possible to edit the visibility of the classes (and their features) and avoid (un)checking them one by one:

-  *Toggle Items*
-  *Show All Items*
-  *Hide All Items*

With vector layer, the contextual menu of a class leaf entry also gives access to:

-  *Select features*: selects in the layer all the features matching that class
-  *Show in attribute table*: opens an attribute table filtered to only the features matching that class
- update the *symbol color* using a **Color Wheel**. For convenience, the recently used colors are also available at the bottom of the color wheel.
- *Edit Symbol...*: opens the *Symbol Selector* dialog to change feature symbol (symbol, size, color...).
- *Copy Symbol*
- *Paste Symbol*

---

**Tipp:** Double-clicking a class leaf entry also opens the *Symbol Selector* dialog.

---








## 12.2.2 Layer Styling Panel


The *Layer Styling* panel (also enabled with `Ctrl+3`) is a shortcut to some of the functionalities of the *Layer Properties* dialog. It provides a quick and easy way to define the rendering and the behavior of a layer, and to visualize its effects without having to open the layer properties dialog.

In addition to avoiding the blocking (or „modal“) layer properties dialog, the layer styling panel also avoids cluttering the screen with dialogs, and contains most style functions (color selector, effects properties, rule edit, label substitution...): e.g., clicking color buttons inside the layer style panel causes the color selector dialog to be opened inside the layer style panel itself rather than as a separate dialog.

From a drop-down list of current layers in the layer panel, select an item and:

- Depending on the active item, set:
  -  *Symbology* for groups (see *Control layers rendering through grouping*)
  -  *Symbology*,  *Transparency*, and  *Histogram* properties for raster layer. These options are the same as in the *Dialogfenster Rasterlayereigenschaften*.
  -  *Symbology*,  *Labels*,  *Mask* and  *3D View* properties for vector layer. These options are the same as in the *Vektorlayereigenschaften* and can be extended by custom properties introduced by third-party plugins.

-  *Symbology* and  *3D View* properties for mesh layer. These options are the same as in the [Mesh Dataset Properties](#).
-  *Symbology*,  *3D View* and  *Elevation* properties for point cloud layer. These options are the same as in the [Point Clouds Properties](#).
- Enable and configure *global map shading* properties
- Manage the associated style(s) in the  *Style Manager* (more details at [Verwalten von benutzerdefinierten Stilen](#)).
- See the  *History* of changes you applied to the layer style in the current project: you can therefore cancel or restore to any state by selecting it in the list and clicking *Apply*.

Another powerful feature of this panel is the  *Live update* checkbox. Tick it to render your changes immediately on the map canvas: you no longer need to click the *Apply* button.

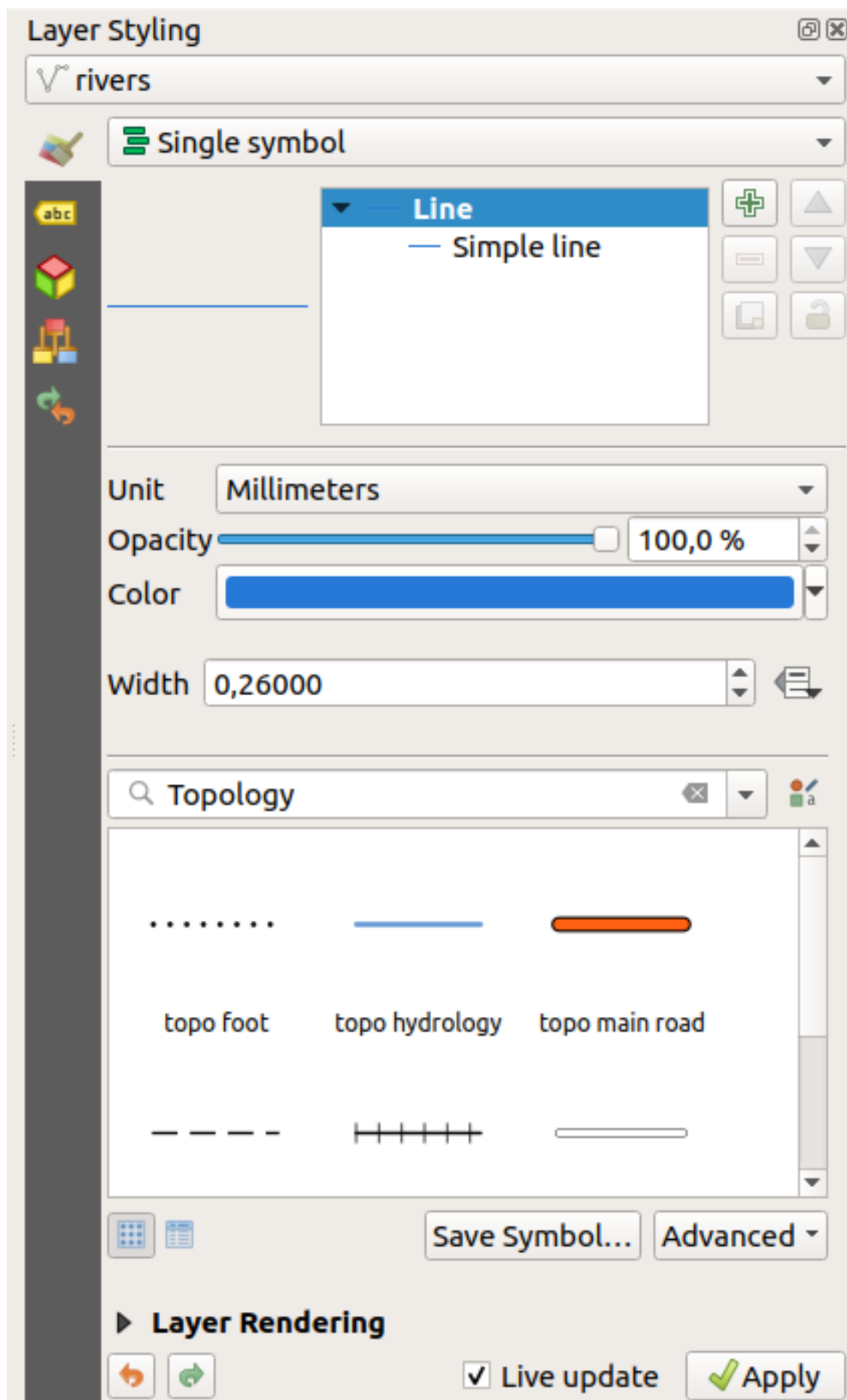


Abb. 12.3: Defining a layer's symbology from the layer styling panel

### 12.2.3 Layer Order Panel

By default, layers shown on the QGIS map canvas are drawn following their order in the *Layers* panel: the higher a layer is in the panel, the higher (hence, more visible) it'll be in the map view.

You can define a drawing order for the layers independent of the order in the layers panel with the *Layer Order* panel enabled in *View ► Panels ►* menu or with `Ctrl+9`. Check ☒ *Control rendering order* underneath the list of layers and reorganize the layers in the panel as you want. This order becomes the one applied to the map canvas. For example, in Abb. 12.4, you can see that the *airports* features are displayed over the *alaska* polygon despite those layers' respective placement in the Layers panel.

Unchecking ☒ *Control rendering order* will revert to default behavior.

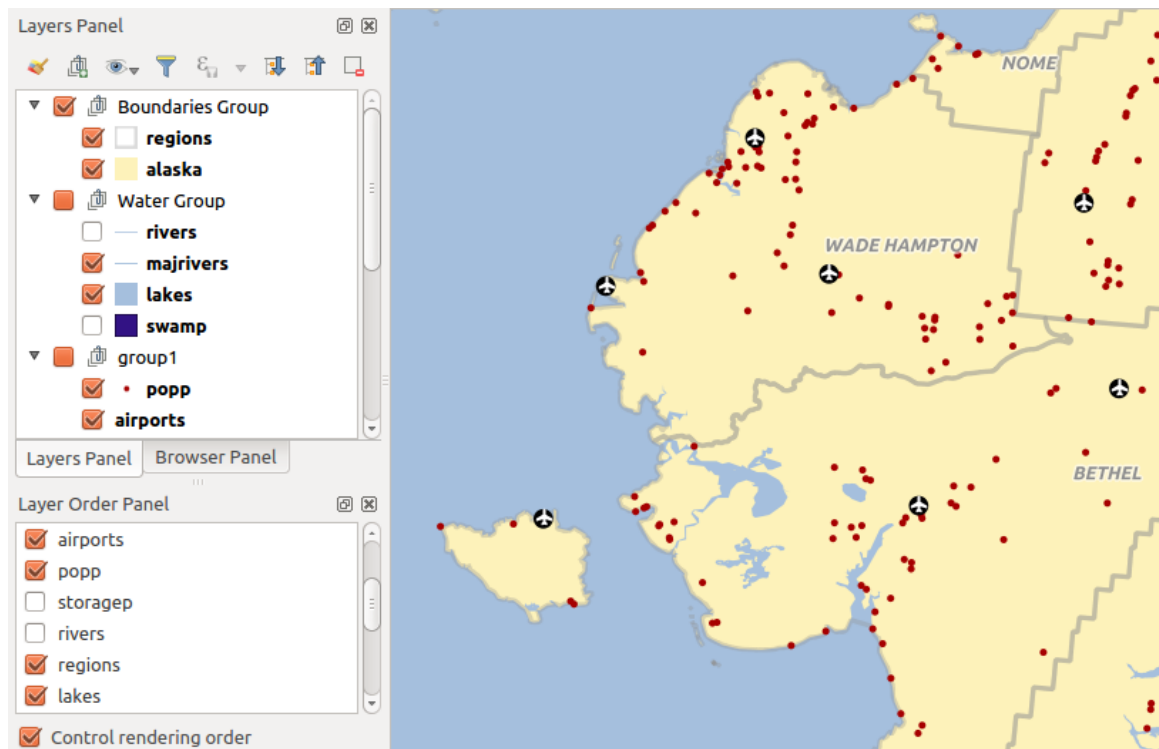



Abb. 12.4: Define a layer order independent of the legend

### 12.2.4 Overview Panel

The *Overview* panel (`Ctrl+8`) displays a map with a full extent view of some of the layers. The Overview map is filled with layers using the *Show in Overview* option from the *Layer* menu or in the layer contextual menu. Within the view, a red rectangle shows the current map canvas extent, helping you quickly to determine which area of the whole map you are currently viewing. If you click-and-drag the red rectangle in the overview frame, the main map view extent will update accordingly.

Note that labels are not rendered to the map overview even if the layers used in the map overview have been set up for labeling.

## 12.2.5 Log Messages Panel


When loading or processing some operations, you can track and follow messages that appear in different tabs using the  *Log Messages Panel*. It can be activated using the most right icon in the bottom status bar.

## 12.2.6 Undo/Redo Panel
















































For each layer being edited, the *Undo/Redo* (Ctrl+5) panel shows the list of actions carried out, allowing you quickly to undo a set of actions by selecting the action listed above. More details at [Undo and Redo edits](#).

## 12.2.7 Statistical Summary Panel




The *Statistics* panel (Ctrl+6) provides summarized information on any vector layer. This panel allows you to select:

- the vector layer to compute the statistics on: it can be selected from the top drop-down menu, or synced with the active layer in the *Layers* panel using the *Follow selected layer* checkbox at the bottom of the statistics drop-down list
- the field or  *expression* to use: for each layer, the last entry is remembered and automatically computed upon layer reselection.
- the statistics to return using the drop-down button at the bottom-right of the dialog. Depending on the field's (or expression's values) type, available statistics are:

Tab. 12.2: Statistics available for each field type

Statistics	Text	Integer	Float	Datum
Count				
Count Distinct Value				
Count Missing value				
Sum				
Mean				
Standard Deviation				
Standard Deviation on Sample				
Minimal value				
Maximal value				
Bereich				
Minority				
Majority				
Variety				
First Quartile				
Third Quartile				
Inter Quartile Range				
Minimum Length				
Maximum Length				
Mean Length				

The statistical summary can be:

- returned for the whole layer or  *Selected features only*
-  copied to the clipboard and pasted as a table in another application
- recalculated using the  button when the underlying data source changes (eg, new or removed features/fields, attribute modification)

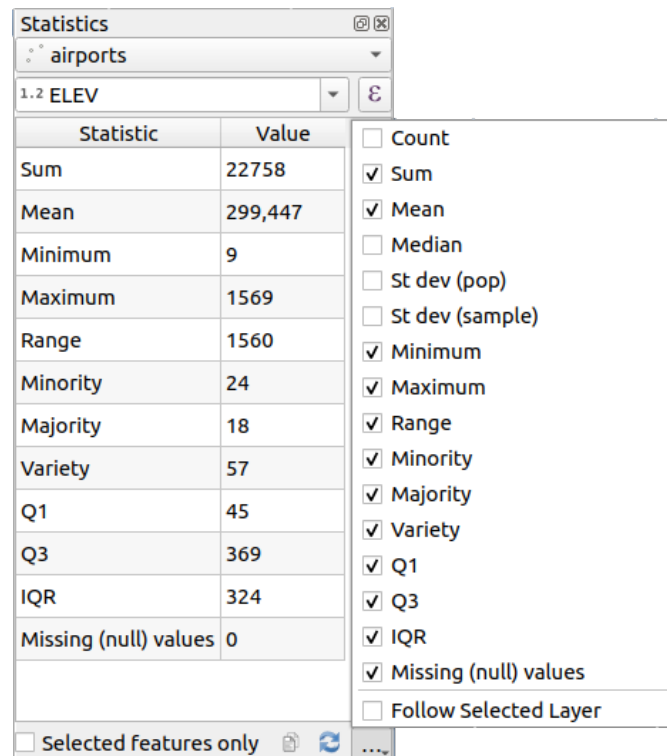





Abb. 12.5: Show statistics on a field


## 12.2.8 Debugging/Development Tools Panel

The *Debugging/Development Tools* panel (F12) provides a unified place for handling and debugging actions within QGIS. Available tools are organized under following tabs:







-  *Network Logger*
-  *Query Logger*
-  *Profiler* - under this tab you can choose one out of these categories:
  - *Startup* helps you diagnose (and fix) occasional long startup times of QGIS.
  - *Project Load* allows you to get a breakdown of the various stages of project load, in order to identify the causes of slow project load times.
  - *Map Render* tool allows you to identify pain points in your map rendering and track down exactly which layers are causing long map redraws.

**Bemerkung:** Plugin authors can extend the panel with custom tabs for debugging and developing their own plugins. This is done using `registerDevToolWidgetFactory` method.

## Network Logger

The  *Network Logger* tab helps you record and display network requests, along with a whole load of useful details like request and reply status, header, errors, SSL configuration errors, timeouts, cache status, etc.

From its top toolbar, you can:

-  *Record Log*: which will start or stop the logging.
-  *Clear Log*: will clear the log history.
-  *Save Log...*: will first show a big warning that the log is sensitive and should be treated as confidential and then allow you to save the log.
- Press the  *Settings* drop-down menu to select whether to *Show Successful Requests*, *Show Timeouts* and *Show Replies Served from Cache*.
-  *Disable cache*: will disable the cache so that every request has to be performed.
-  *Filter requests* based on URL string subsets or request status

By right clicking on a request you can:

- *Open URL* which will open the URL in your default browser.
- *Copy URL*
- *Copy As cURL* to use it in the terminal.
- *Copy as JSON*: copies the tree values as a json string to the clipboard, for easy pasting in bug reports or for remote assistance.

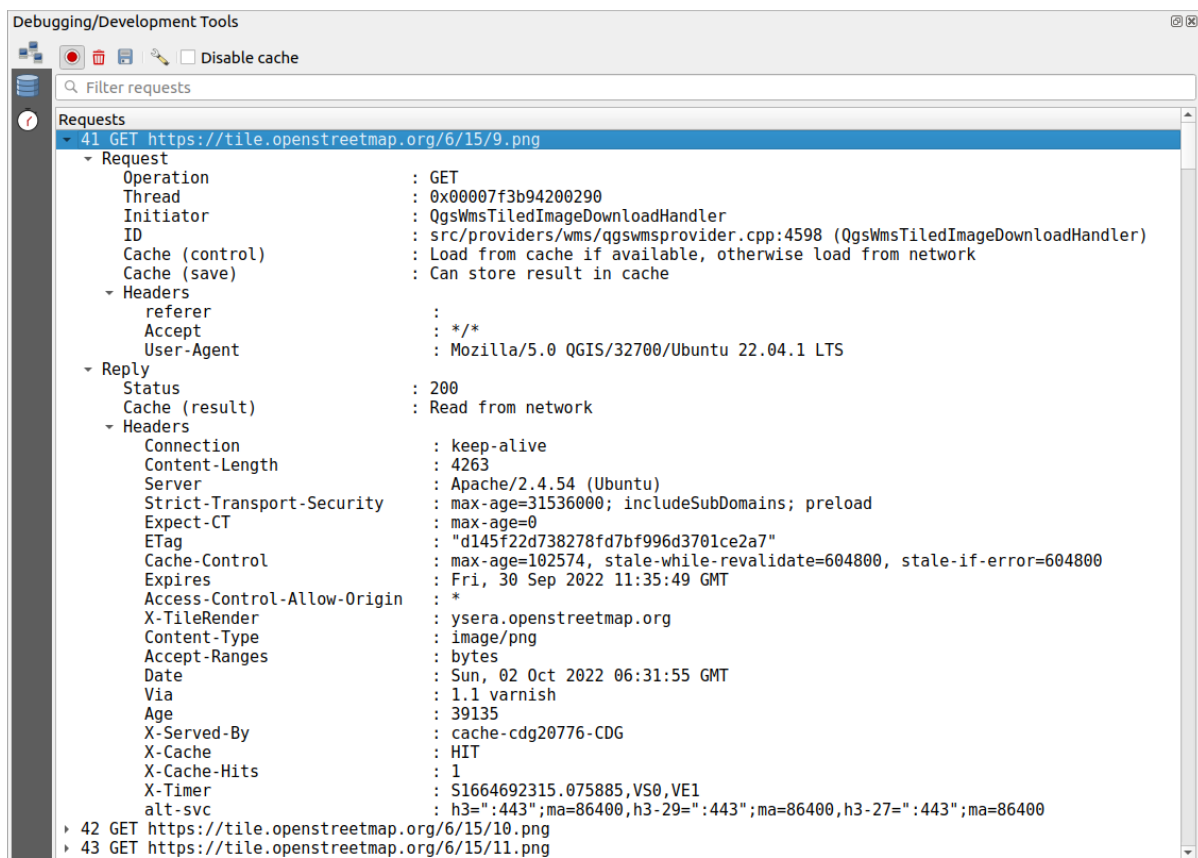







Abb. 12.6: Network Logger output for GET Request



## Query Logger

The  *Query Logger* is the place to log the SQL commands sent by the data provider and the connections API to the backend database, together with their execution time as measured by QGIS (i.e. in the client that sent the commands). This can be useful when investigating performances of a particular layer during debugging or development of a QGIS algorithm or plugin.

From its top toolbar, you can:

-  *Record Log*: which will start or stop the logging.
-  *Clear Log*: will clear the log history.
-  *Save Log...*: will first show a big warning that the log is sensitive and should be treated as confidential and then allow you to save the log.
-  *Filter queries* based on the query string subsets or details such as provider type, start time, initiator, ...

Right-click on a reported query, and you can:

- *Copy SQL* command called by QGIS on the database
- *Copy as JSON*: copies the tree values as a json string to the clipboard, for easy pasting in bug reports or for remote assistance.

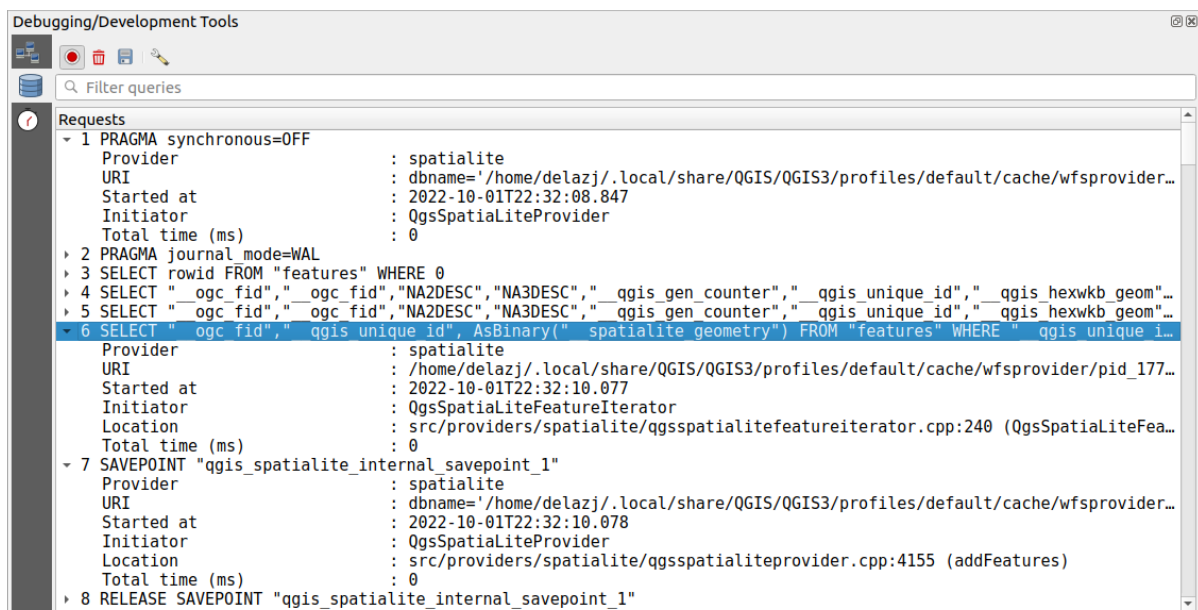



Abb. 12.7: Query Logger output

## Profiler

The  *Profiler* tab allows to get load times for every single operation involved in actions requested by the user. Depending on the context, these operations can be settings reading, menu, map canvas or 3D views creation, resolving map layers reference, bookmark or layout loading, ... This helps identify causes of slow load times.

Default supported actions can be selected from the *Categories* drop-down menu:

- *QGIS Startup*
- *Project Load*

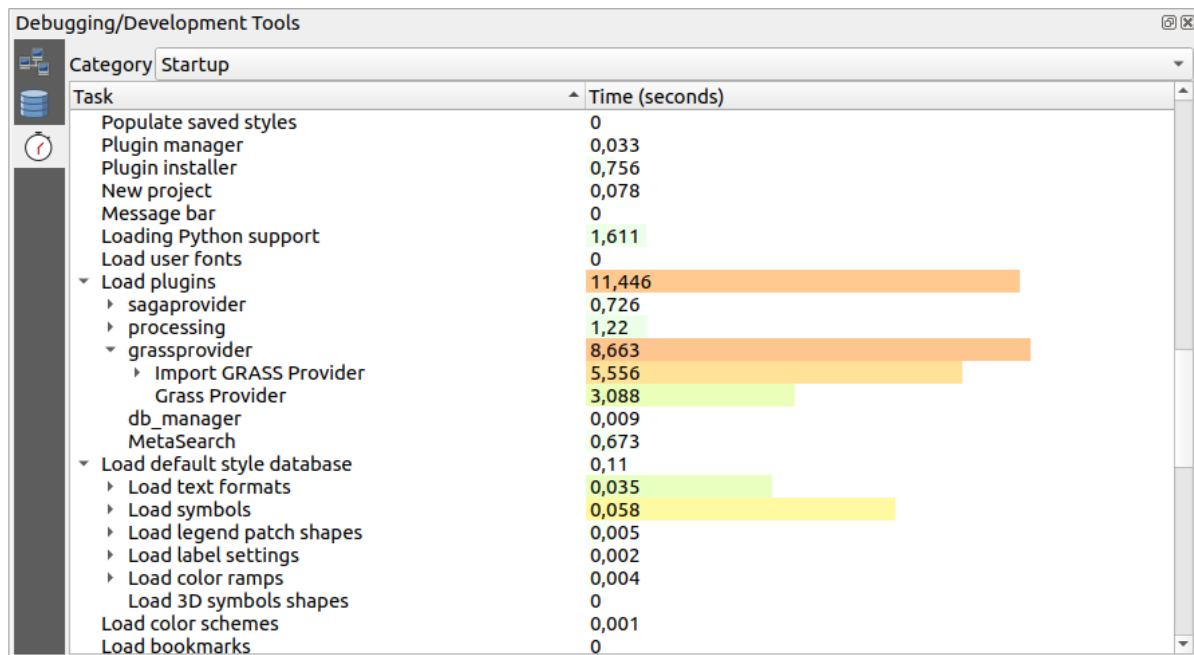


Abb. 12.8: Profiler for QGIS Startup

## 12.3 Embedding layers from external projects


Sometimes, you'd like to keep some layers in different projects, but with the same style. You can either create a *default style* for these layers or embed them from another project to save time and effort.

Embed layers and groups from an existing project has some advantages over styling:

- All types of layers (vector or raster, local or online...) can be added
- Fetching groups and layers, you can keep the same tree structure of the „background“ layers in your different projects
- While the embedded layers are editable, you can't change their properties such as symbology, labels, forms, default values and actions, ensuring consistency across projects
- Modify the items in the original project and changes are propagated to all the other projects

If you want to embed content from other project files into your project, select *Layer ► Embed Layers and Groups*:

1. Click the ... button to look for a project: you can see the content of the project (see [Abb. 12.9](#))
2. Hold down **Ctrl** (or **X** **Cmd**) and click on the layers and groups you wish to retrieve
3. Klicken Sie auf **OK**

The selected layers and groups are embedded in the *Layers* panel and displayed on the map canvas. An  icon is added next to their name for recognition and hovering over displays a tooltip with the original project file path.

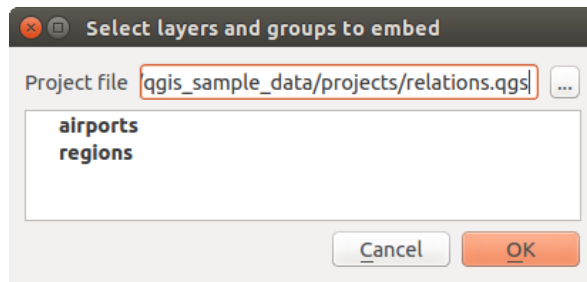


Abb. 12.9: Einzubettende Layer und Gruppen wählen

Like any other layer, an embedded layer can be removed from the project by right-clicking on the layer and clicking




---

### **Tipp:** Change rendering of an embedded layer

It's not possible to change the rendering of an embedded layer, unless you make the changes in the original project file. However, right-clicking on a layer and selecting *Duplicate* creates a layer which is fully-featured and not dependent on the original project. You can then safely remove the linked layer.

---

## 12.4 Interacting with features

### 12.4.1 Selecting features

QGIS provides several tools to select features on the map canvas. Selection tools are available in the *Edit ► Select* menu or in the *Selection Toolbar*.





---

**Bemerkung:** Selection tools work with the currently active layer.


---

#### Selecting manually on the map canvas

To select one or more features with the mouse, you can use one of the following tools:

-  Select Features by area or single click
-  Objekte durch Polygon wählen
-  Objekte freihändig wählen
-  Objekte durch Radius wählen


---

**Bemerkung:** Other than  *Select Features by Polygon*, these manual selection tools allow you to select feature(s) on the map canvas with a single click.

---



---


**Bemerkung:** Use the  *Select Features by Polygon* tool to use an existing polygon feature (from any layer) to select overlapping features in the active layer. Right-click in the polygon and choose it from the context menu that shows a list of all the polygons that contain the clicked point. All the overlapping features from the active layer are selected.

---

---

**Tipp:** Use the *Edit ► Select ► Reselect Features* tool to redo your latest selection. Very useful when you have painstakingly made a selection, and then click somewhere else accidentally and clear your selection.

---









While using the  *Select Feature(s)* tool, holding **Shift** or **Ctrl** toggles whether a feature is selected (ie either adds to the current selection or remove from it).

For the other tools, different behaviors can be performed by holding down:


- **Shift**: add features to the current selection
- **Ctrl**: subtract features from the current selection
- **Ctrl+Shift**: intersect with current selection, ie only keep overlapping features from the current selection
- **Alt**: select features that are totally within the selection shape. Combined with **Shift** or **Ctrl** keys, you can add or subtract features to/from the current selection.

### Automatic selection

The other selection tools, most of them available from the *Attribute table*, perform a selection based on a feature's attribute or its selection state (note that attribute table and map canvas show the same information, so if you select one feature in the attribute table, it will be selected on the map canvas too):

-  *Select By Expression...* select features using expression dialog
-  *Select Features By Value...* or press **F3**
-  *Deselect Features from All Layers* or press **Ctrl+Alt+A** to deselect all selected features in all layers
-  *Deselect Features from the Current Active Layer* or press **Ctrl+Shift+A**
-  *Select All Features* or press **Ctrl+A** to select all features in the current layer
-  *Invert Feature Selection* to invert the selection in the current layer
-  *Select by Location* to select the features based on their spatial relationship with other features (in the same or another layer - see *Auswahl nach der Lage*)
-  *Select within distance* to select features wherever they are within the specified maximum distance from referenced features - see *Select within distance*)

For example, if you want to find regions that are boroughs from `regions.shp` of the QGIS sample data, you can:

1. Use the  icon
2. Expand the *Fields and Values* group
3. Double-click the field that you want to query („TYPE\_2“)
4. Click *All Unique* in the panel that shows up on the right
5. From the list, double-click ‚Borough‘. In the *Expression* editor field, write the following query:

```
"TYPE_2" = 'Borough'
```

6. Click *Select Features*

From the expression builder dialog, you can also use *Function list ► Recent (Selection)* to make a selection that you have used before. The dialog remembers the last 20 expressions used. See *Ausdrücke* for more information and examples.

---

**Tipp:** Save your selection into a new file

Users can save selected features into a **New Temporary Scratch Layer** or a **New Vector Layer** using *Edit ► Copy Features* and *Edit ► Paste Features as* in the desired format.

### Select Features By Value

This selection tool opens the layer's feature form allowing the user to choose which value to look for for each field, whether the search should be case-sensitive, and the operation that should be used. The tool has also autocompletes, automatically filling the search box with existing values.

The dialog box titled "Select Features by Value" contains the following elements:

- Fields: `fid`, `id`, `fk_region`, `elev` (value: 100), `name`, `use` (value: M).
- Buttons: "Exclude field" for each field.
- Dropdowns: "Equal to (=)" for `elev`, "Contains" for `use`.
- Checkboxes: "Case sensitive" for `name` and `use`.
- List box: Shows "Joint Military/Civilian" and "Military" (selected).
- Bottom buttons: "Reset form", "Flash features", "Zoom to features", "Select features" (dropdown), "Close".

Abb. 12.10: Filter/Select features using form dialog

Alongside each field, there is a drop-down list with options to control the search behaviour:

Tab. 12.3: Query operators per data type

Field search option	Text	Numeric	Datum
<i>Exclude Field from the search</i>			
<i>Equal to (=)</i>			
<i>Not equal to (<math>\neq</math>)</i>			
<i>Greater than (<math>&gt;</math>)</i>			
<i>Less than (<math>&lt;</math>)</i>			
<i>Greater than or equal to (<math>\geq</math>)</i>			
<i>Less than or equal to (<math>\leq</math>)</i>			
<i>Between (inclusive)</i>			
<i>Not between (inclusive)</i>			
<i>Contains</i>			
<i>Does not contain</i>			
<i>Is missing (null)</i>			
<i>Is not missing (not null)</i>			
<i>Starts with</i>			
<i>Ends with</i>			

For string comparisons, it is also possible to use the *Case sensitive* option.

After setting all search options, click *Select features* to select the matching features. The drop-down options are:

- *Select features*
- *Add to current selection*
- *Remove from current selection*
- *Filter current selection*

You can also clear all search options using the *Reset form* button.

Once the conditions are set, you can also either:

- *Zoom to features* on the map canvas without the need of a preselection
- *Flash features*, highlighting the matching features. This is a handy way to identify a feature without selection or using the Identify tool. Note that the flash does not alter the map canvas extent and would be visible only if the feature is within the bounds of the current map canvas.

## 12.4.2 Identifying Features

The *Identify Features* tool allows you to interact with the map canvas and get information on features or pixels in a pop-up window. It can be used to query most of the layer types supported by QGIS (vector, raster, mesh, point cloud, wms, wfs, ...). To identify an element, use either:

- *View ► Identify Features*
- **Ctrl+Shift+I** (or **X** **Cmd+Shift+I**),
- *Identify Features* button on the *Attributes* toolbar

Then click on a feature or pixel of the active layer. The identified item gets highlighted in the map canvas while the *Identify Results* dialog opens with detailed information on it. The dialog also shows a set of buttons for advanced configuration.

### The Identify Results dialog

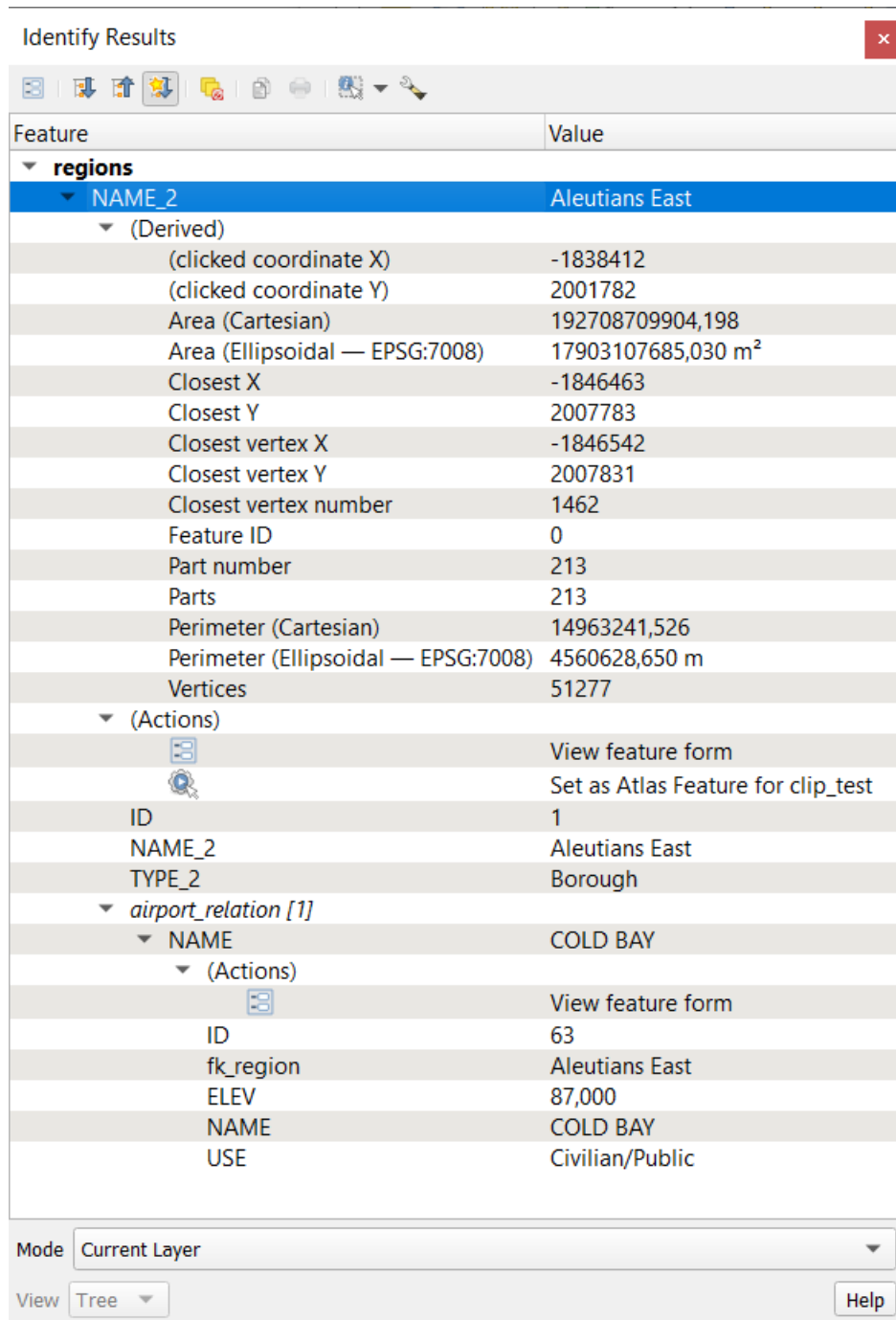














Abb. 12.11: Identify Results dialog


From bottom to top:

- The *View* controls the general aspect of the dialog and the formatting of the results; it can be set as:
  - **Tree**: this is the default view, and returns the results in a tree-structure





- **Table:** available only for raster-based layers, it allows to display the results as a table whose columns are Layer, FID, Attribute and Value
- or **Graph:** available only for raster-based layers
- The *Mode* helps you select the layers from which results could be returned. These layers should be set visible, displaying data in the map canvas, and set *identifiable* from the *Project properties* ► *Data Sources* ► *Layers capabilities*. Available modes are:
  - **Current layer:** only the layer(s) selected in the *Layers* panel return results. If a group is selected, then results are picked from its leaf layers.
  - **Top down, stop at first:** results are from the layer of the top most feature or pixel under the mouse.
  - **Top down:** results are from the layers with feature or pixel under the mouse.
  - **Layer selection:** opens a contextual menu where the user selects the layer to identify features from. If only a single feature is under the mouse, then the results are automatically displayed.
- In the upper part of the *Identify Results* dialog, a frame shows the *information* returned by features as a table, a graph or a tree, depending on the *selected view*. When in a tree view, you have a handful of tools above the results:
  -  Open Form of the current feature
  -  Expand tree
  -  Collapse tree
  -  Expand New Results by Default to define whether the next identified feature's information should be collapsed or expanded
  -  Ergebnisse leeren
  -  Copy the identified feature to clipboard, suitable for pasting in a spreadsheet.
  -  Print selected HTML response: a text-based formatting of the results to print on paper or save as a .PDF file
  - the *interactive identifying tools*: a drop-down menu with tools for selecting on the map canvas features or pixels to identify
  - Under  Identify Settings, you can activate whether to:
    - \*  *Auto open form for single feature results*: If checked, each time a single feature is identified, a form opens showing its attributes. This is a handy way to quickly edit a feature's attributes.
    - \*  *Hide derived attributes from results* to only show fields actually defined in the layer
    - \*  *Hide NULL values from results*
  -  Help to access the current documentation

## Using the Identify Features tool

In its default display (*View: Tree*), the *Identify Results* panel offers several tools to interact with the layers to query. A smart combination of these tools with the *target layers selector* may greatly improve identification operations:


-  Identify Feature(s) by single click or click-and-drag
  - single click or click-and-drag: overlaying features in the target layers are returned
  - right-click: overlaying features from target layers are listed in the contextual menu, grouped by layers. You can then choose to:
    - \* display the result for a specific feature,



- \* display the result for all the features of a specific layer,
  - \* for vector layers, it is also possible to open its attribute table filtered to the returned features
  - \* or show all of the returned features.
-  **Identify Features on Mouse over**: move over the map canvas and hovered items in the target layers get highlighted and returned in the results panel.
  -  **Identify Features by Polygon**: returns items overlapping a drawn or selected polygon.
    - Draw a polygon (left click to add point, right click to close the polygon) and all the overlaying features from target layers are highlighted and returned in the results panel.
    - Right-click and you get the list of all visible polygon features in the project under the click. Pick an entry and QGIS will return all the features from the target layers that overlap the selected polygon.
  -  **Identify Features by Freehand**: returns items overlapping a polygon drawn by freehand. Draw a polygon (left-click to start, move the pointer to shape the area and right-click to close the polygon). All the overlaying features from target layers are highlighted and returned in the results panel.
  -  **Identify Features by Radius** returns items overlapping a drawn circle. Draw a circle (left-click to indicate the center point, move the pointer to shape the area or enter the radius in the pop-up text box and left-click or press `Enter` to validate the circle). All the overlaying features from target layers are highlighted and returned in the results panel.

---

**Tip: Filter the layers to query with the Identify Features tool**

Under *Project ► Properties... ► Data Sources ► Layer Capabilities*, uncheck the *Identifiable* column next to a layer to avoid it being queried when using the  **Identify Features** tool. This is a handy way to return features from only layers that are of interest to you.

---

## Feature information

When you identify a data in the map canvas, the *Identify Results* dialog will list information about the items clicked (or hovered over, depending on the tool in use). The default view is a tree view in which the first item is the name of the layer and its children are its identified feature(s). Each feature is described by the name of a field along with its value. This field is the one set in *Layer Properties ► Display*. All the other information about the feature follows.

The feature information displayed by the identify tool will depend on the type of layer you have selected, whether it is a vector layer (including vector tiles or point cloud data) or raster layer. If your layer is raster, clicking on a location on the map canvas with identify tool will highlight the identified raster pixel. The Identify Results dialog can be customized to display custom fields, but by default it will display the following information:

- The feature *display name*;
- **Actions**: Actions can be added to the identify feature windows. The action is run by clicking on the action label. By default, only one action is added, namely *View feature form* for editing. You can define more actions in the layer's properties dialog (see *Eigenschaften Aktionen*).
- **Derived**: This information is calculated or derived from other information. It includes:
  - general information about the feature's geometry:
    - \* depending on the geometry type, the cartesian measurements of length, perimeter or area in the layer's CRS units. For 3D line vectors the cartesian line length is available.
    - \* depending on the geometry type and if an ellipsoid is set in the project properties dialog for *Measurements*, the ellipsoidal values of length, perimeter or area using the specified units
    - \* the count of geometry parts in the feature and the number of the part clicked

- \* the count of vertices in the feature
- coordinate information, using the project properties *Coordinates display* settings:
  - \* X and Y coordinate values of the point clicked
  - \* the number of the closest vertex to the point clicked
  - \* X and Y coordinate values of the closest vertex (and Z/M if applicable)
  - \* if you click on a curved segment, the radius of that section is also displayed.
- **Datenattribute:** Dies ist die Liste von Attributfeldern und Werten für das Objekt, welches angeklickt wurde.
- information about the related child feature if you defined a *relation*:
  - the name of the relation
  - the entry in reference field, e.g. the name of the related child feature
  - **Actions:** lists actions defined in the layer's properties dialog (see *Eigenschaften Aktionen*) and the default action is `View feature form`.
  - **Data attributes:** This is the list of attributes fields and values of the related child feature.

---

**Bemerkung:** Links in the feature's attributes are clickable from the *Identify Results* panel and will open in your default web browser.

---

### Results contextual menu

Andere Funktionen können im Kontextmenü des abgefragten Objekts gefunden werden. Im Kontextmenü können Sie z.B.:

- Das Objektformular anzeigen
- Zum Objekt zoomen
- Objekt kopieren: Kopieren Sie alle Objektgeometrien und -attribute
- Toggle feature selection: Add identified feature to selection
- Attributwert kopieren: Kopieren Sie nur den Wert des Attributes auf das Sie klicken
- Objektattribute kopieren: Kopieren Sie die Attribute des Objekts
- Select features by attribute value: Select all features in the layer that match the selected attribute
- Ergebnisse löschen: Löschen Sie Ergebnisse im Fenster
- Hervorhebungen löschen: Entfernen Sie in der Karte hervorgehobene Objekte
- Alle hervorheben
- Layer hervorheben
- Layer aktivieren: Wählen Sie einen Layer der aktiviert werden soll
- Layereigenschaften ... : Öffnen Sie das Layereigenschaften Fenster
- Alles ausklappen
- Alles zusammenfalten

## 12.5 Save and Share Layer Properties

### 12.5.1 Verwalten von benutzerdefinierten Stilen

When a vector layer is added to the map canvas, QGIS by default uses a random symbol/color to render its features. However, you can set a default symbol in *Project ► Properties... ► Default styles* that will be applied to each newly added layer according to its geometry type.

Most of the time, though, you'd rather have a custom and more complex style that can be applied automatically or manually to the layers (with less effort). You can achieve this by using the *Style* menu at the bottom of the Layer Properties dialog. This menu provides you with functions to create, load and manage styles.

A style stores any information set in the layer properties dialog to render or interact with the layer (including symbology, labeling, fields and form definitions, actions, diagrams...) for vector layers, or the pixels (band or color rendering, transparency, pyramids, histogram ...) for raster.

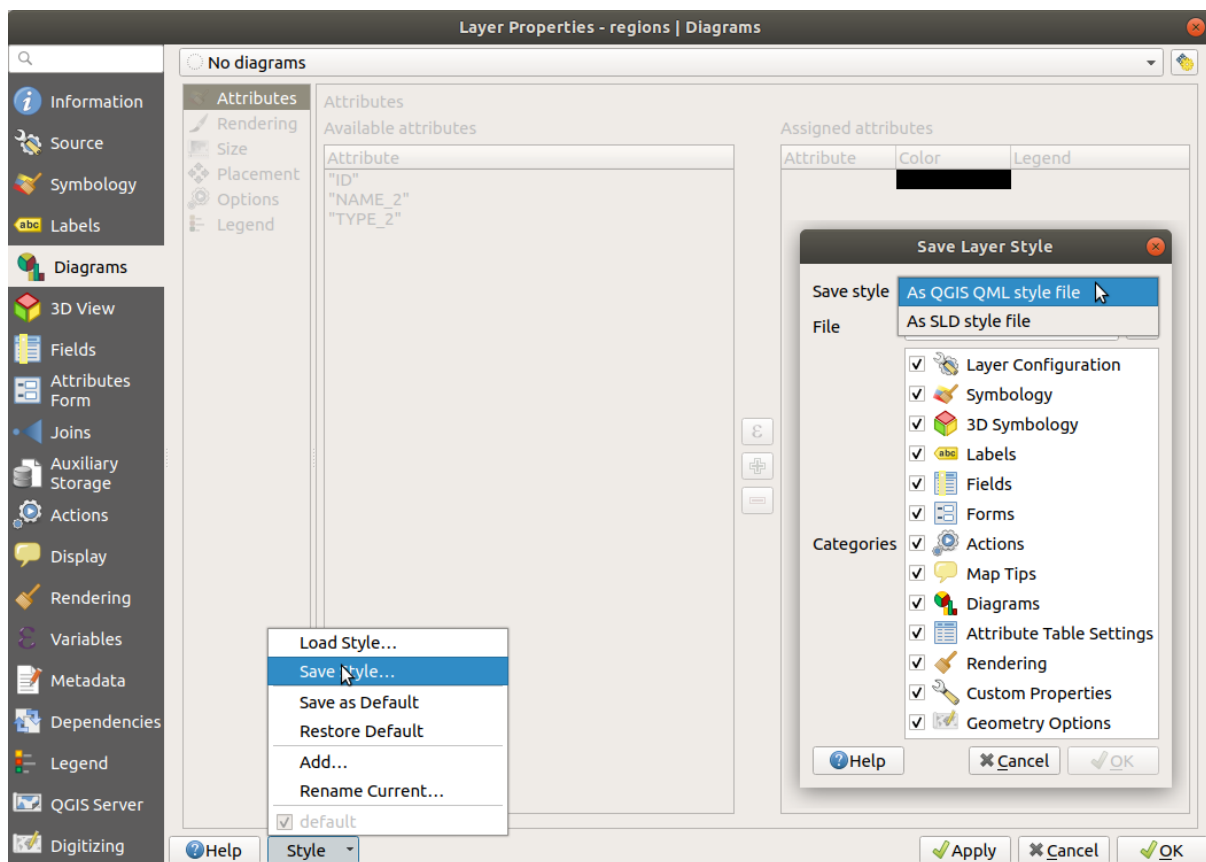


Abb. 12.12: Vector layer style combo box options

By default, the style applied to a loaded layer is named *default*. Once you have got the ideal and appropriate rendering for your layer, you can save it by clicking the *Style* combo box and choosing:

- **Rename Current:** The active style is renamed and updated with the current options
- **Add:** A new style is created using the current options. By default, it will be saved in the QGIS project file. See below to save the style in another file or a database
- **Remove:** Delete unwanted style, in case you have more than one style defined for the layer.

At the bottom of the Style drop-down list, you can see the styles set for the layer with the active one checked.

Note that each time you validate the layer properties dialog, the active style is updated with the changes you've made.

You can create as many styles as you wish for a layer but only one can be active at a time. In combination with *Map Themes*, this offers a quick and powerful way to manage complex projects without the need to duplicate any layer in the map legend.

---

**Bemerkung:** Given that whenever you apply modifications to the layer properties, changes are stored in the active style, always ensure you are editing the right style to avoid mistakenly altering a style used in a *map theme*.

---

---

### **Tipp: Manage styles from layer context menu**

Right-click on the layer in the *Layers* panel to copy, paste, add or rename layer styles.

---

## 12.5.2 Storing Styles in a File or a Database

While styles created from the *Style* combo box are by default saved inside the project and can be copied and pasted from layer to layer in the project, it's also possible to save them outside the project so that they can be loaded in another project.

### Save as text file

Clicking the  *Style* ► *Save Style*, you can save the style as a:

- QGIS layer style file (.qml)
- SLD file (.sls), only available for vector layers

Used on file-based format layers (.shp, .tab...), *Save as Default* generates a .qml file for the layer (with the same name). SLDs can be exported from any type of renderer – single symbol, categorized, graduated or rule-based – but when importing an SLD, either a single symbol or rule-based renderer is created. This means that categorized or graduated styles are converted to rule-based. If you want to preserve those renderers, you have to use the QML format. On the other hand, it can be very handy sometimes to have this easy way of converting styles to rule-based.

### Save in database

Vector layer styles can also be stored in a database if the layer datasource is a database provider. Supported formats are PostGIS, GeoPackage, SpatiaLite, MS SQL Server and Oracle. The layer style is saved inside a table (named `layer_styles`) in the database. Click on *Save Style...* ► *Save in database* then fill in the dialog to define a style name, add a description, a .ui file if applicable and to check if the style should be the default style.

You can save several styles for a single table in the database. However, each table can have only one default style. Default styles can be saved in the layer database or in `qgis.db`, a local SQLite database in the active *user profile* directory.

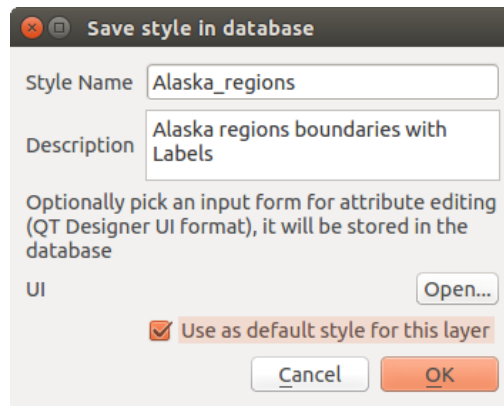


Abb. 12.13: Save Style in database Dialog

**Tipp: Sharing style files between databases**

You can only save your style in a database if the layer comes from such a database. You can't mix databases (layer in Oracle and style in MS SQL Server for instance). Use instead a plain text file if you want the style to be shared among databases.

**Bemerkung:** You may encounter issues restoring the `layer_styles` table from a PostgreSQL database backup. Follow [QGIS layer\\_style table and database backup](#) to fix that.

**Load style**

When loading a layer in QGIS, if a default style already exists for this layer, QGIS loads the layer with this style. Also *Style ► Restore Default* looks for and loads that file, replacing the layer's current style.

*Style ► Load Style* helps you apply any saved style to a layer. While text-file styles (`.sld` or `.qml`) can be applied to any layer whatever its format, loading styles stored in a database is only possible if the layer is from the same database or the style is stored in the QGIS local database.

The *Database Styles Manager* dialog displays a list of styles related to the layer found in the database and all the other styles saved in it, with name and description.

**Tipp: Quickly share a layer style within the project**

You can also share layer styles within a project without importing a file or database style: right-click on the layer in the *Layers Panel* and, from the *Styles* combo box, copy the style of a layer and paste it to a group or a selection of layers: the style is applied to all the layers that are of the same type (vector vs raster) as the original layer and, in the case of vector layers, have the same geometry type (point, line or polygon).

### 12.5.3 Layer definition file


Layer definitions can be saved as a *Layer Definition File (.qlr)* using *Export ► Save As Layer Definition File...* in the active layers' context menu. A layer definition file (.qlr) includes references to the data source of the layers and their styles. .qlr files are shown in the Browser Panel and can be used to add the layers (with the saved style) to the Layers Panel. You can also drag and drop .qlr files from the system file manager into the map canvas.

## 12.6 Documenting your data

In addition to displaying and symbolizing the data in the layers, QGIS allows you to fill:

- **metadata:** information to help people find and understand the dataset, how they can access and use it... these are properties of the datasource and can live out of the QGIS project.
- **notes:** instructions and comments regarding the layer in the current project

### 12.6.1 Metadata

In the layer properties dialog, the  *Metadata* tab provides you with options to create and edit a metadata report on your layer.

Information to fill concern:

- the data *Identification*: basic attribution of the dataset (parent, identifier, title, abstract, language...);
- the *Categories* the data belongs to. Alongside the **ISO** categories, you can add custom ones;
- the *Keywords* to retrieve the data and associated concepts following a standard based vocabulary;
- the *Access* to the dataset (licenses, rights, fees, and constraints);
- the *Extent* of the dataset, either spatial one (CRS, map extent, altitudes) or temporal;
- the *Contact* of the owner(s) of the dataset;
- the *Links* to ancillary resources and related information;
- the *History* of the dataset.

A summary of the filled information is provided in the *Validation* tab and helps you identify potential issues related to the form. You can then either fix them or ignore them.

Metadata are saved in the project file by default, the *Metadata* drop-down offers options for loading/saving metadata from .qmd file and for loading/saving metadata in the „Default“ location.

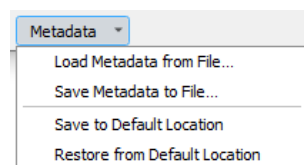


Abb. 12.14: Metadata load/save options

The „Default“ location used by *Save to Default Location* and *Restore from Default Location* changes depending on the underlying data source and on its configuration:

- For PostgreSQL data sources if the configuration option *Allow saving/loading QGIS layer metadata in the database* is checked the metadata are stored inside a dedicated table in the database.

- For GeoPackage data sources *Save to Default Location* always saves the metadata in the internal metadata tables of the GeoPackage.

When metadata are saved into the internal tables of PostgreSQL or GeoPackage they become available for search and filtering in the browser and in the [layer metadata search panel](#).

- For all other file based data sources *Save to Default Location* saves the metadata in a `.qmd` file alongside the file.
- In all other cases *Save to Default Location* saves the metadata in a local `.sqlite` database.

## 12.6.2 Bemerkungen zu Layern

Layer notes allow you to document the layer within the current project. They can be place to store important messages for users of the project like to do lists, instructions, warnings, ...

From the layer's contextual menu in *Layers* panel, select *Add layer notes...* and fill the open dialog with necessary texts.

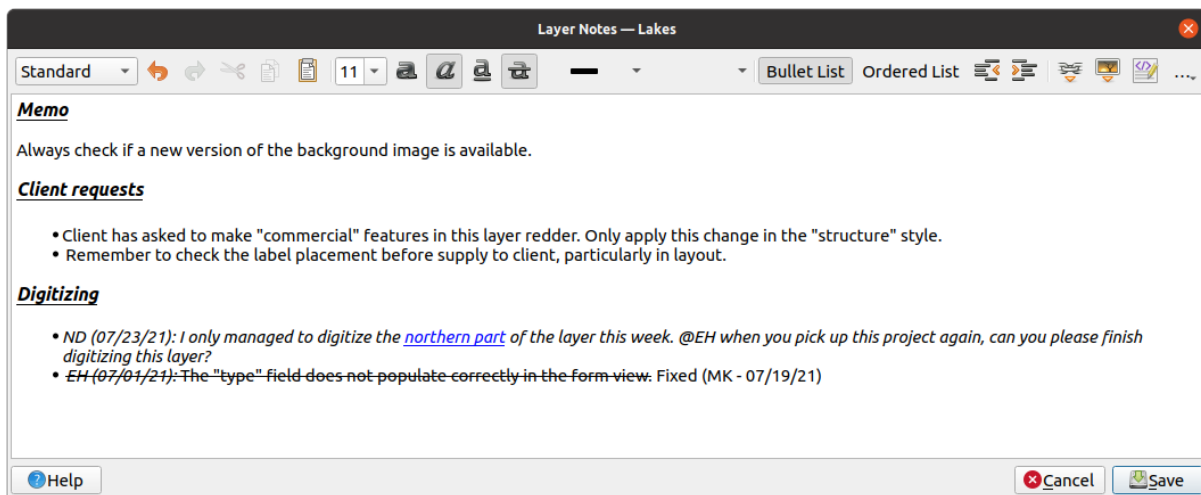


Abb. 12.15: Adding notes to a layer

The *Add layer notes* dialog provides a html-based multiline text box with a complete set of tools for:

- text manipulation: cut, copy, paste, undo, redo
- characters formatting, applied to all or parts of the contents: font size and color, bold, italic, underline, strikethrough, background color, URL highlighting
- paragraph structuring: bullet and numbered lists, indentation, predefined headings
- file insertion, even with drag-and-drop
- editing with HTML coding

From the ... drop-down at the far right of the toolbar, you can:

- *Remove all formatting*
- *Remove character formatting*
- *Clear all content*

In the *Layers* panel, a layer with a note is assigned the  icon which, upon hover, displays the note. Click the icon to edit the note. You can as well right-click the layer and *Edit layer note...* or *Remove layer note*.

**Bemerkung:** Notes are part of the *layer style* and can be saved in the `.qml` or `.qlr` file. They can also be transferred from one layer to another while copy-pasting the layer style.

## 12.7 Storing values in Variables

In QGIS, you can use variables to store useful recurrent values (e.g. the project's title, or the user's full name) that can be used in expressions. Variables can be defined at the application's global level, project level, layer level, processing modeler level, layout level, and layout item's level. Just like CSS cascading rules, variables can be overwritten - e.g., a project level variable will overwrite any application global level variables set with the same name. You can use these variables to build text strings or other custom expressions using the `@` character before the variable name. For example in print layout creating a label with this content:

This `map` was made using QGIS [% @qgis\_version %]. The project file `for` this `map is`: [% @project\_path %]

Wird diesen Namen wie folgt darstellen:

This `map` was made using QGIS 3.4.4-Madeira. The project file `for` this `map is`:  
/gis/qgis-user-conference-2019.qgs

Besides the *preset read-only variables*, you can define your own custom variables for any of the levels mentioned above. You can manage:

- **global variables** from the *Settings ► Options* menu
- **project variables** from the *Project Properties* dialog (see *Projekteigenschaften*)
- **vector layer variables** from the *Layer Properties* dialog (see *Vektorlayereigenschaften*);
- **modeler variables** from the *Model Designer* dialog (see *The model designer*);
- **layout variables** from the *Layout* panel in the Print layout (see *The Layout Panel*);
- and **layout item variables** from the *Item Properties* panel in the Print layout (see *Layout Items Common Options*).

To differentiate from editable variables, read-only variable names and values are displayed in italic. On the other hand, higher level variables overwritten by lower level ones are strike through.



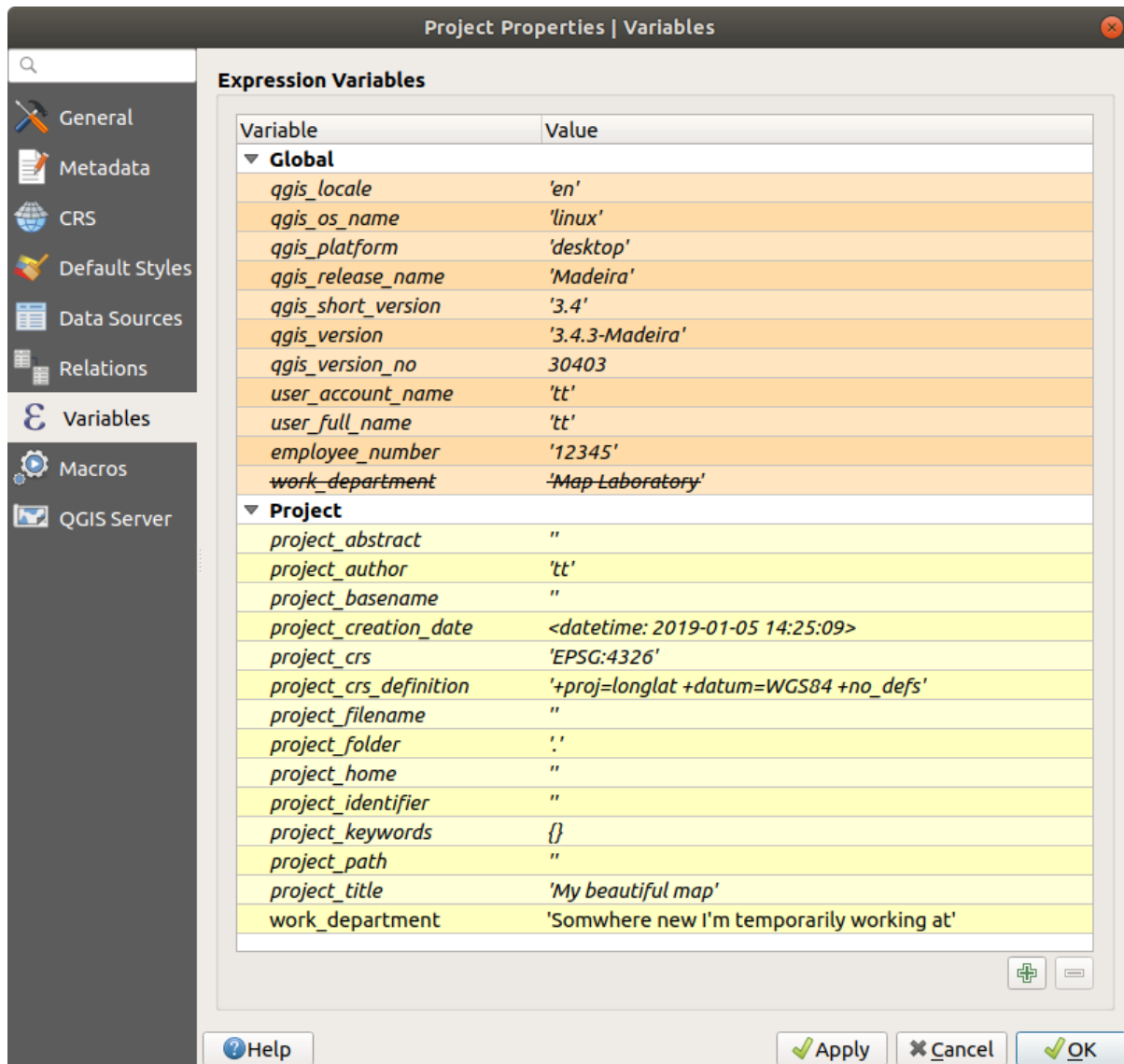


Abb. 12.16: Variables editor at the project level

**Bemerkung:** You can read more about variables and find some examples in Nyall Dawson's [Exploring variables](#) in QGIS 2.12, [part 1](#), [part 2](#) and [part 3](#) blog posts.

## 12.8 Authentifizierung

QGIS has the facility to store/retrieve authentication credentials in a secure manner. Users can securely save credentials into authentication configurations, which are stored in a portable database, can be applied to server or database connections, and are safely referenced by their ID tokens in project or settings files. For more information see [Authentifizierungssystem](#).


Ein Hauptpasswort muss eingerichtet werden, wenn das Authentifizierungssystem und seine tragbare Datenbank initialisiert werden sollen.





## 12.9 Common widgets

In QGIS, there are some options you'll often have to work with. For convenience, QGIS provides you with special widgets that are presented below.

### 12.9.1 Farbauswahl

#### The color dialog

The *Select Color* dialog will appear whenever you click the  icon to choose a color. The features of this dialog depend on the state of the *Use native color chooser dialogs* parameter checkbox in *Settings ► Options... ► General*. When checked, the color dialog used is the native one of the OS on which QGIS is running. Otherwise, the QGIS custom color chooser is used.

The custom color chooser dialog has four different tabs which allow you to select colors by  Color ramp,  Color wheel,  Color swatches or  Color picker. With the first two tabs, you can browse to all possible color combinations and apply your choice to the item.

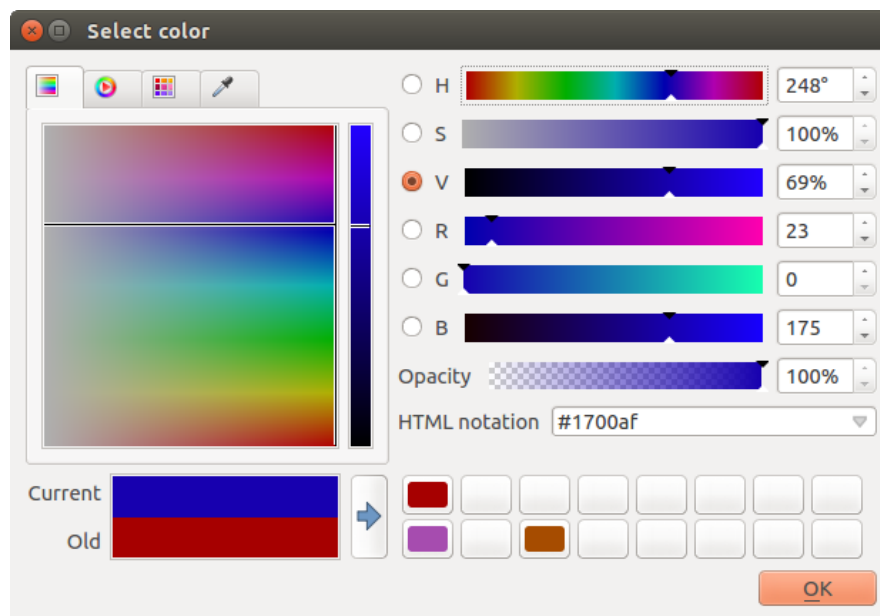





Abb. 12.17: Farbauswahl Verlaufstyp

In the  Color swatches tab, you can choose from a list of color palettes (see [Farben](#) for details). All but the *Recent colors* palette can be modified with the  Add current color and  Remove selected color buttons at the bottom of the frame.

The ... button next to the palette combo box also offers several options to:

- copy, paste, import or export colors
- create, import or remove color palettes
- add the custom palette to the color selector widget with the *Show in Color Buttons* item (see [Abb. 12.19](#))

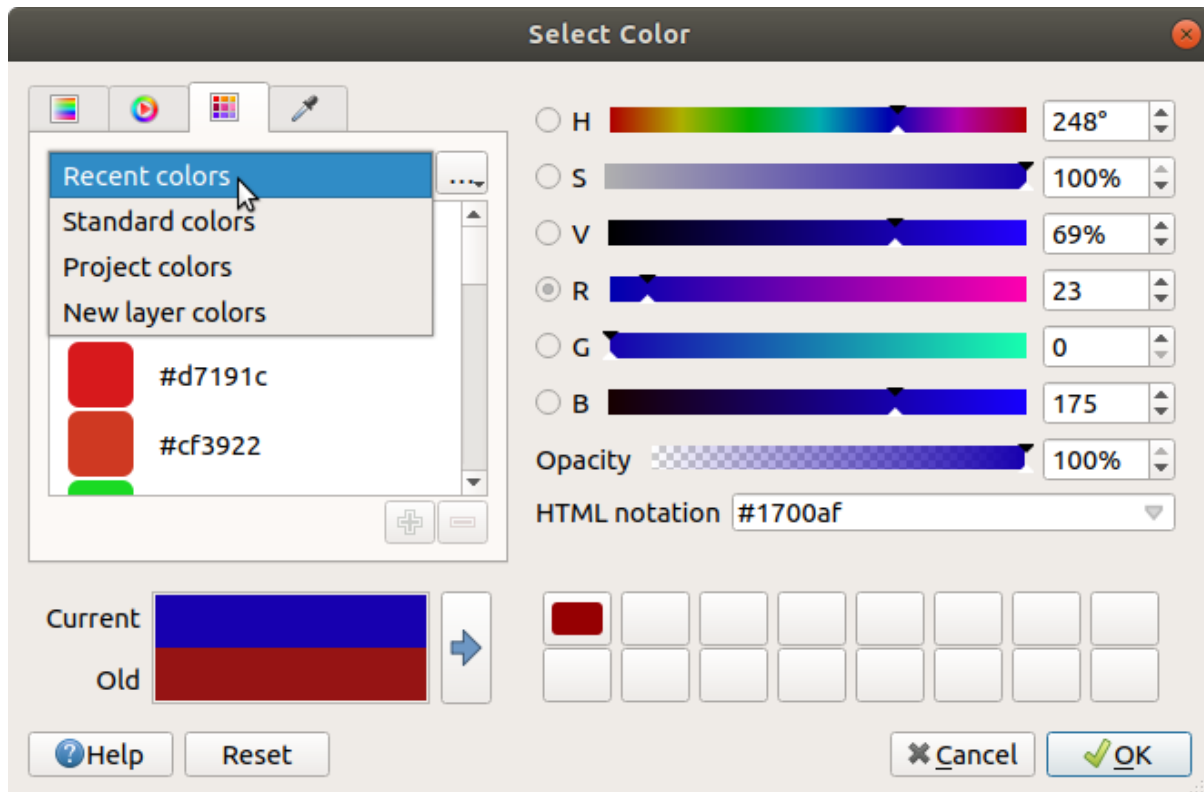




Abb. 12.18: Color selector swatches tab

Another option is to use the  **Color picker** which allows you to sample a color from under your mouse cursor at any part of the QGIS UI or even from another application: press the space bar while the tab is active, move the mouse over the desired color and click on it or press the space bar again. You can also click the *Sample Color* button to activate the picker.

Whatever method you use, the selected color is always described through color sliders for HSV (Hue, Saturation, Value) and RGB (Red, Green, Blue) values. The color is also identifiable in *HTML notation*.

Modifying a color is as simple as clicking on the color wheel or ramp or on any of the color parameters sliders. You can adjust such parameters with the spinbox beside or by scrolling the mouse wheel over the corresponding slider. You can also type the color in HTML notation. Finally, there is an *Opacity* slider to set transparency level.

The dialog also provides a visual comparison between the *Old* color (applied to object) and the *Current* one (being selected). Using drag-and-drop or pressing the  **Add color to swatch** button, any of these colors can be saved in a slot for easy access.

---

#### **Tipp: Quick color modification**

Drag-and-drop a color selector widget onto another one to apply its color.

---

### The color drop-down shortcut

Click the drop-down arrow to the right of the  color button to display a widget for quick color selection. This shortcut provides access to:

- a color wheel to pick a color from
- an alpha slider to change color opacity
- the color palettes previously set to *Show in Color Buttons*
- copy the current color and paste it into another widget
- pick a color from anywhere on your computer display
- choose a color from the color selector dialog
- drag-and-drop the color from one widget to another for quick modification

---

**Tipp:** Scroll the mouse wheel over a color selector widget to quickly modify the opacity of the associated color.

---



---

**Bemerkung:** When the color widget is set to a *project color* through the data-defined override properties, the above functions for changing the color are unavailable. You'd first need to *Unlink color* or *Clear* the definition.

---

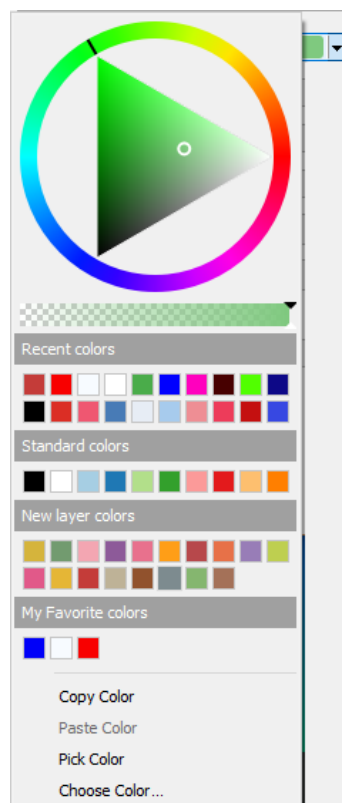



Abb. 12.19: Schnelle Farbwahl Menü

## The color ramp drop-down shortcut

Color ramps are a practical way to apply a set of colors to one or many features. Their creation is described in the [Setting a Color Ramp](#) section. As for the colors, pressing the  color ramp button opens the corresponding color ramp type dialog allowing you to change its properties.

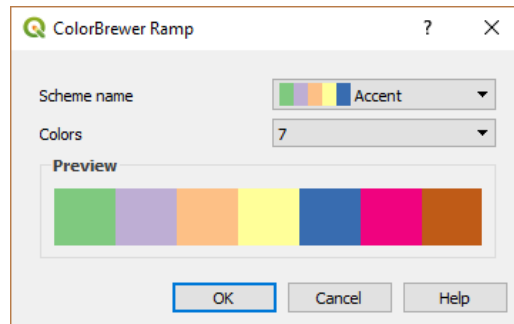



Abb. 12.20: Customizing a colorbrewer ramp

The drop-down menu to the right of the button gives quick access to a wider set of color ramps and options:

- *Invert Color Ramp*
- *Clear Current Ramp* to unset any assigned color ramp to the widget (available only in some contexts)
-  *Random Colors*: available only in some contexts (e.g., when a color ramp is being used for a layer symbology), checking this entry creates and applies a color ramp with random colors. It also enables a *Shuffle random colors* entry to regenerate a new random color ramp if the current one is not satisfactory.
- a preview of the gradient or catalog: `cpt-city` color ramps flagged as **Favorites** in the *Style Manager* dialog
- *All Color Ramps* to access the compatible color ramps database
- *Create New Color Ramp...* of any supported type that could be used in the current widget (note that this color ramp will not be available elsewhere unless you save it in the library)
- *Edit Color Ramp...*, the same as clicking the whole color ramp button
- *Save Color Ramp...*, to save the current color ramp with its customizations in the style library

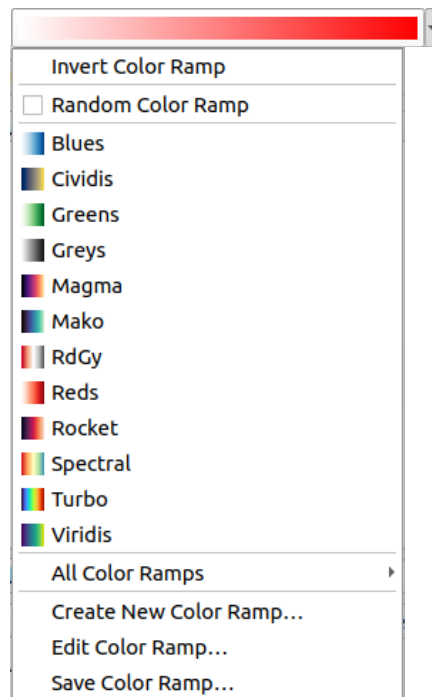


Abb. 12.21: Quick color ramp selection widget

## 12.9.2 Symbol Widget

The *Symbol* selector widget is a convenient shortcut when you want to set symbol properties of a feature. Clicking the drop-down arrow shows the following symbol options, together with the features of the *color drop-down widget*:

- *Configure Symbol...*: the same as pressing the symbol selector widget. It opens a dialog to set the *symbol parameters*.
- *Copy Symbol* from the current item
- *Paste Symbol* to the current item, speeding configuration
- *Clear Current Symbol* to unset any assigned symbol to the widget (available only in some contexts)

---

**Tipp:** Scroll the mouse wheel over a marker or line symbol widget to quickly modify the size of the associated symbol.

---

## 12.9.3 Remote or embedded file selector

Along with the file selector widget, the ... button will sometimes show a drop-down arrow. This is usually available when using:

- an SVG file in a symbol or a label
- a raster image to customize symbols, labels, textures or decorations


Pressing the arrow will provide you with a menu to:


- *Select File...*, to load the file from the file system. The file is identified through the file path and QGIS needs to resolve the path in order to display the corresponding image
- *From URL...*, to load the file from a remote URL. As above, the image will only be loaded on successful retrieval of the remote resource

- *Embed File...*, to embed the file inside the current project, style database, or print layout template. The file is then always rendered as part of the item. This is a convenient way to create self-contained projects with custom symbols which can be easily shared amongst different users and installations of QGIS.
- *Extract Embedded File...*, to extract the embedded file from the widget and save it on disk.

### 12.9.4 Visibility Scale Selector

The visibility scale selector provides options to control the scales at which an element will be made visible in the map canvas. Out of the specified range of scales, the elements are not displayed. It can be applied e.g. to layers, labels or diagrams, from their *Rendering* properties tab.

1. Tick the  *Scale dependent visibility* box
2. Fill the *Minimum (exclusive)* box with the most zoomed out desired scale, typing the value or selecting it from the *predefined scales*
3. and/or fill the *Maximum (inclusive)* box with the most zoomed in desired scale

The  Set to current canvas scale button next to the scale boxes sets the current map canvas scale as boundary of the range visibility. Press the arrow next to the button to access scales from layouts' maps and reuse them to fill the box.

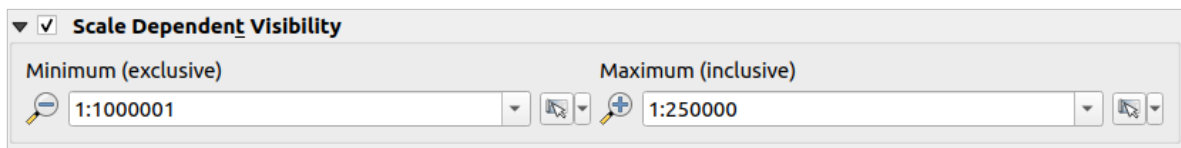


Abb. 12.22: Visibility scale selector widget

### 12.9.5 Spatial Extent Selector

The *Extent* selector widget is a convenient shortcut when you want to select a spatial extent to assign to a layer or to limit the actions to run on. Depending on the context, it offers selection between:

- *Current Layer Extent*: e.g. when exporting a layer
- *Calculate from Layer* ►: uses extent of a layer loaded in the current project
- *Use current Map Canvas Extent*
- *Draw on Canvas*: a rectangle whose coordinates are then used
- *Calculate from Bookmark*: uses extent of a saved *bookmark*
- *Calculate from Layout Map*: uses extent of a *layout map*
- Enter or edit the coordinates as `xmin`, `xmax`, `ymin`, `ymax`

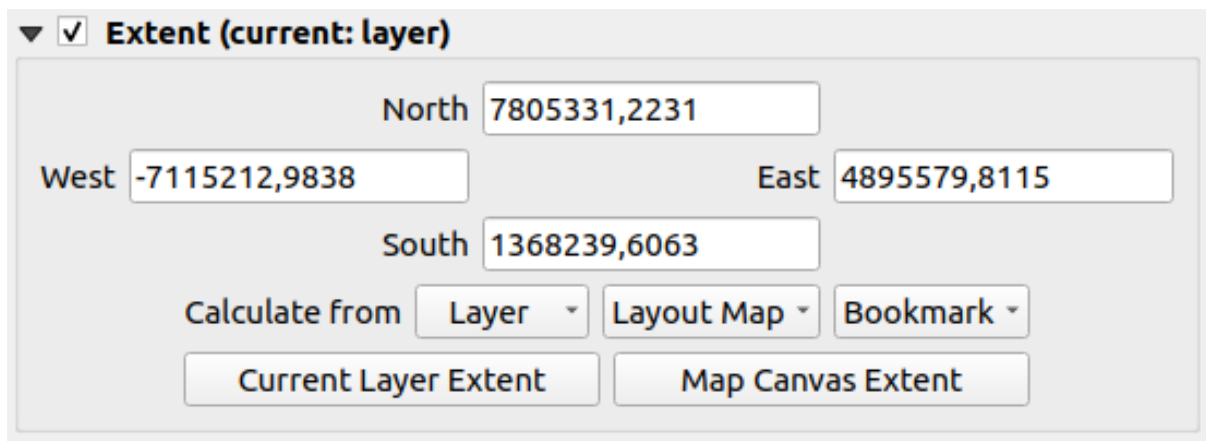


Abb. 12.23: Extent selector widget

## 12.9.6 Font Selector

The *Font* selector widget is a convenient shortcut when you want to set font properties for textual information (feature labels, decoration labels, map legend text, ...). Clicking the drop-down arrow shows some or all of the following options:



Abb. 12.24: Font selector drop-down menu

- *Clear Current Text Format* to unset any assigned text format to the widget (available only in some contexts)
- *Font Size* in the associated unit
- *Recent Fonts* ► menu with the active font checked (at the top)
- *Configure Format...*: same as pressing the font selector widget. It opens the *Text format* dialog, providing advanced formatting options such as color, opacity, orientation, HTML notation, buffer, background, shadow, ...




- *Copy Format* of the text
- *Paste Format* to the text, speeding configuration
- the *color widget* for quick color setting

**Tipp:** Scroll the mouse wheel over a font selector widget to quickly modify the font size of the associated text.

## 12.9.7 Unit Selector

Size properties of the items (labels, symbols, layout elements, ...) in QGIS are not necessarily bound to either the project units or the units of a particular layer. For a large set of properties, the *Unit* selector drop-down menu allows you to tweak their values according to the rendering you want (based on screen resolution, paper size, or the terrain). Available units are:

- *Millimeters*
- *Points*
- *Pixels*
- *Inches*
- *Percentage*: allows you to set some properties as a percent of another one. For example, this is useful for creation of text formats where the components (buffer size, shadow radius...) nicely scale as the text size is changed, instead of having constant buffer/shadow sizes. So you don't need to adjust those sizes, when the text size changes.
- *Meters at Scale*: This allows you to always set the size in meters, regardless of what the underlying map units are (e.g. they can be in inches, feet, geographic degrees, ...). The size in meters is calculated based on the current project ellipsoid setting and a projection of the distances in meters at the center of the current map extent. For maps in a projected coordinate system this is calculated using projected units. For maps in a geographic (latitude/longitude) based system the size is approximated by calculating meter sizes using ellipsoidal calculations for the vertical scale of the map.
- and *Map Units*: The size is scaled according to the map view scale. Because this can lead to too big or too small values, use the  button next to the entry to constrain the size to a range of values based on:
  - The *Minimum scale* and the *Maximum scale*: The value is scaled based on the map view scale until you reach any of these scale limits. Out of the range of scale, the value at the nearest scale limit is kept.
  - and/or The *Minimum size* and the *Maximum size* in mm: The value is scaled based on the map view scale until it reaches any of these limits; Then the limit size is kept.

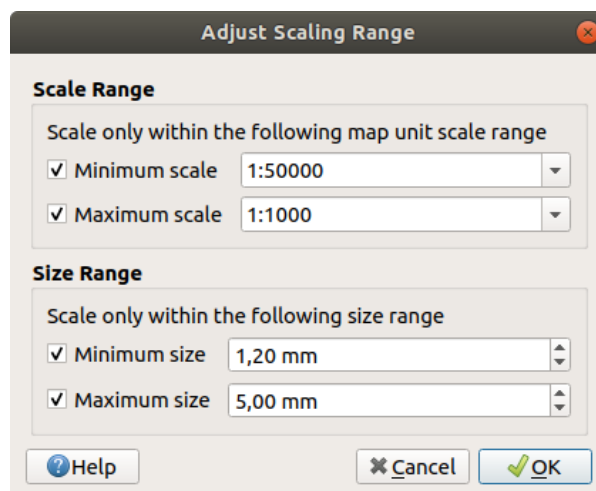


Abb. 12.25: Adjust scaling range dialog

## 12.9.8 Number Formatting

Numeric formatters allow formatting of numeric values for display, using a variety of different formatting techniques (for instance scientific notation, currency values, percentage values, etc). One use of this is to set text in a layout scale bar or fixed table.

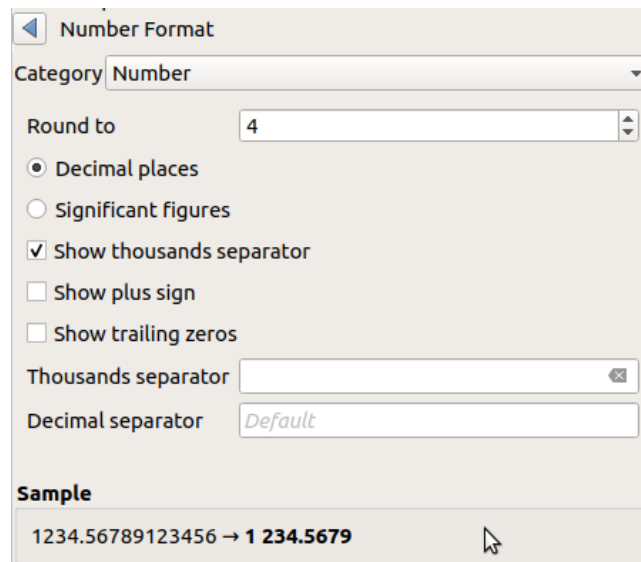



Abb. 12.26: Formatting numeric value

Different categories of formats are supported. For most of them, you can set part or all of the following numeric options:

- ☒ *Show thousands separator*
- ☐ *Show plus sign*
- ☐ *Show trailing zeros*

But they can also have their custom settings. Provided categories are:

- *General*, the default category: has no setting and displays values as set in the parent widget properties or using the global settings.
- *Number*
  - The value can be *Round to* a self defined number of *Decimal places* or their *Significant figures*
  - customize the *Thousands separator* and *Decimal separator*
- *Bearing* for a text representation of a direction/bearing using:
  - *Format*: possible ranges of values are 0 to 180°, with E/W suffix, -180 to +180° and 0 to 360°
  - number of *Decimal places*
- *Currency* for a text representation of a currency value.
  - *Prefix*
  - *Suffix*
  - number of *Decimal places*
- *Fraction* for a vulgar fractional representation of a decimal value (e.g. 1/2 instead of 0.5)
  - ☐ *Use unicode super/subscript to show*. For example <sup>1</sup>/<sub>2</sub> instead of 1/2

-  Use dedicated Unicode characters
- customize the *Thousands separator*
- *Percentage* - appends % to the values, with setting of:
  - number of *Decimal places*
  - *Scaling* to indicate whether the actual values already represent percentages (then they will be kept as is) or fractions (then they are converted)
- *Scientific* notation in the form 2.56e+03. The number of *Decimal places* can be set.

A live preview of the settings is displayed under the *Sample* section.

### 12.9.9 Mischmodi

QGIS offers different options for special rendering effects with these tools that you may previously only know from graphics programs. Blending modes can be applied on layers and features, and also on print layout items:

- *Normal*: This is the standard blend mode, which uses the alpha channel of the top pixel to blend with the pixel beneath it. The colors aren't mixed.
- *Lighten*: This selects the maximum of each component from the foreground and background pixels. Be aware that the results tend to be jagged and harsh.
- *Screen*: Light pixels from the source are painted over the destination, while dark pixels are not. This mode is most useful for mixing the texture of one item with another item (such as using a hillshade to texture another layer).
- *Dodge*: Brighten and saturate underlying pixels based on the lightness of the top pixel. Brighter top pixels cause the saturation and brightness of the underlying pixels to increase. This works best if the top pixels aren't too bright. Otherwise the effect is too extreme.
- *Addition*: Adds pixel values of one item to the other. In case of values above the maximum value (in the case of RGB), white is displayed. This mode is suitable for highlighting features.
- *Darken*: Retains the lowest values of each component of the foreground and background pixels. Like lighten, the results tend to be jagged and harsh.
- *Multiply*: Pixel values of the top item are multiplied with the corresponding values for the bottom item. The results are darker.
- *Burn*: Darker colors in the top item cause the underlying items to darken. Burn can be used to tweak and colorize underlying layers.
- *Overlay*: Combines multiply and screen blending modes. Light parts become lighter and dark parts become darker.
- *Soft light*: Very similar to overlay, but instead of using multiply/screen it uses color burn/dodge. This is supposed to emulate shining a soft light onto an image.
- *Hard light*: Hard light is also very similar to the overlay mode. It's supposed to emulate projecting a very intense light onto an image.
- *Difference*: Subtracts the top pixel from the bottom pixel, or the other way around, in order always to get a positive value. Blending with black produces no change, as the difference with all colors is zero.
- *Subtract*: Subtracts pixel values of one item from the other. In the case of negative values, black is displayed.

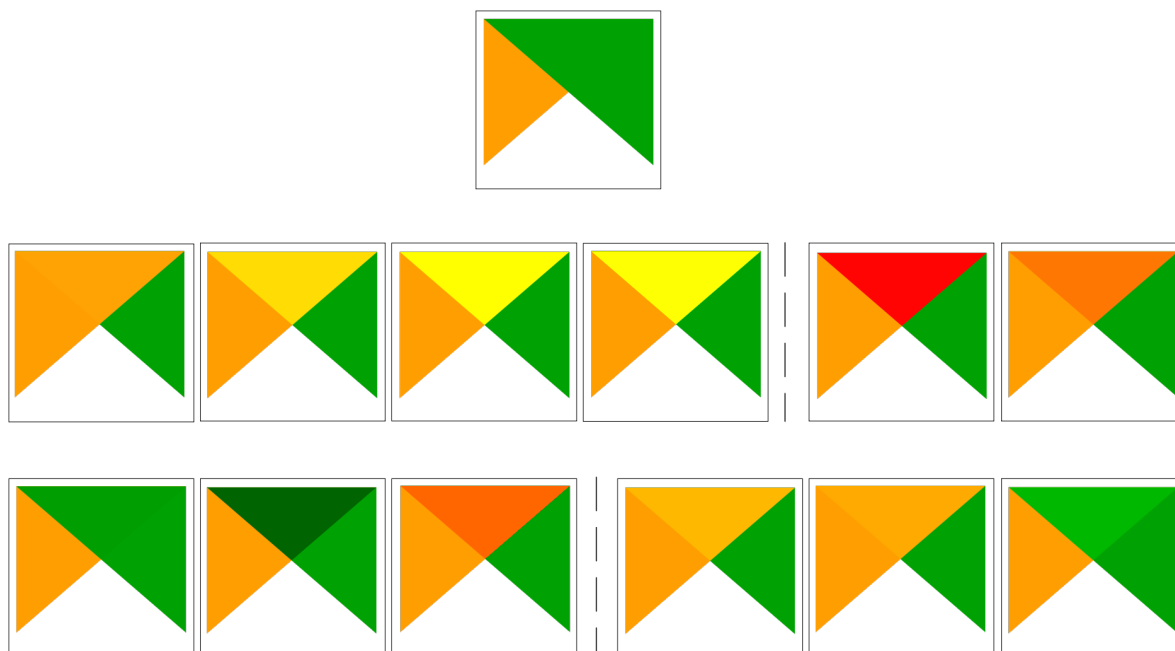


Abb. 12.27: Examples of blend modes applied to a green feature over an orange  
 From top to bottom, left to right: Normal – Lighten, Screen, Dodge, Addition – Difference, Subtract – Darken, Multiply, Burn – Overlay, Soft light, Hard light

When a layer is part of a group that *renders layers as a group*, additional blending modes are available for the rendering. They provide methods to clip the render of one layer's content by the content in a second „mask“ layer.

- *Masked By Below*: The output is the top pixel, where the opacity is reduced by that of the bottom pixel.
- *Mask Below*: The output is the bottom pixel, where the opacity is reduced by that of the top pixel.
- *Inverse Masked By Below*: The output is the top pixel, where the opacity is reduced by the inverse of the bottom pixel.
- *Inverse Mask Below*: The output is the bottom pixel, where the opacity is reduced by the inverse of the top pixel.
- *Paint Inside Below*: The top pixel is blended on top of the bottom pixel, with the opacity of the top pixel reduced by the opacity of the bottom pixel.
- *Paint Below Inside*: The bottom pixel is blended on top of the top pixel, with the opacity of the bottom pixel reduced by the opacity of the top pixel.

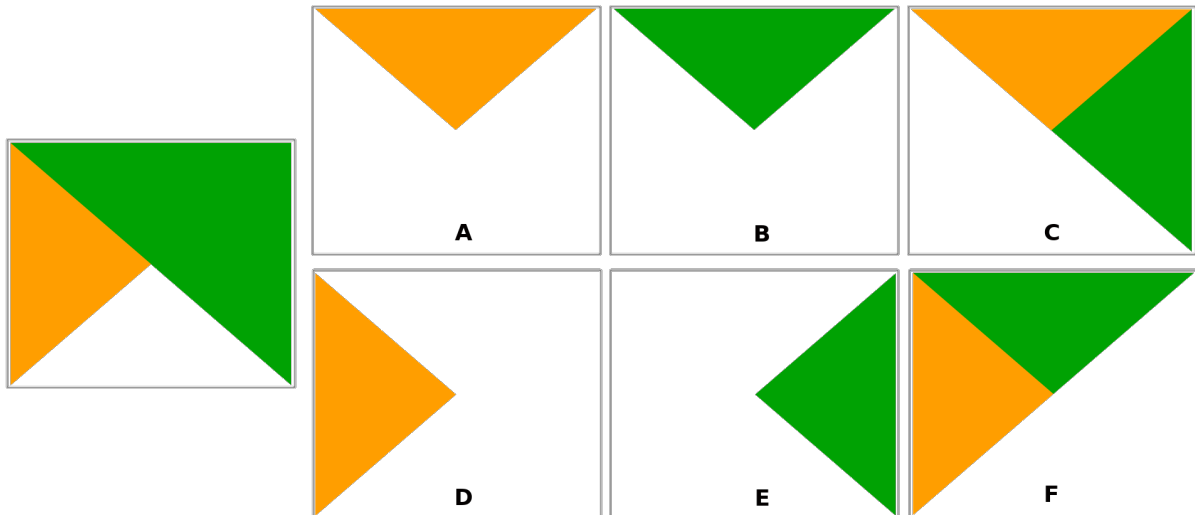




Abb. 12.28: Examples of blend clipping mode applied to top green layer in a group  
 A: Mask Below B: Masked By Below C: Paint Below Inside D: Inverse Mask Below E: Inverse Masked By Below F: Paint Inside Below

### 12.9.10 Datendefinierte Übersteuerung Setup





Next to many options in the vector layer properties dialog or settings in the print layout, you will find a  icon. Using *expressions* based on layer attributes or item settings, prebuilt or custom functions and *variables*, this tool allows you to set dynamic values for parameters. When enabled, the value returned by this widget is applied to the parameter regardless of its normal value (checkbox, textbox, slider...).


#### The data defined override widget

Clicking the  Data defined override icon shows the following entries:

- *Description...* that indicates if the option is enabled, which input is expected, the valid input type and the current definition. Hovering over the widget also pops up this information.
- *Store data in the project*: a button allowing the property to be stored using the *Auxiliary Storage Properties* mechanism.
- *Field type*: an entry to select from the layer's fields that match the valid input type.
- *Color*: when the widget is linked to a color property, this menu gives access to the colors defined as part of the current *project's colors* scheme.
- *Variable*: a menu to access the available user-defined *variables*
- *Edit...* button to create or edit the expression to apply, using the *Expression String Builder* dialog. To help you correctly fill in the expression, a reminder of the expected output's format is provided in the dialog.
- *Paste* and *Copy* buttons.
- *Löschen* Schaltfläche, um das Setup zu entfernen.
- For numeric and color properties, *Assistant...* to rescale how the feature data is applied to the property (more details *below*)


**Tipp:** Use right-click to (de)activate the data override




When the data-defined override option is set up correctly the icon is yellow  or . If it is broken, the icon is red  or .

You can enable or disable a configured  Data-defined override button by simply clicking the widget with the right mouse button.

---

### Using the data-defined assistant interface

When the  Data-defined override button is associated with a size, a rotation, an opacity or a color property, it has an *Assistant...* option that helps you change how the data is applied to the parameter for each feature. The assistant allows you to:

- Define the *Input* data, ie:
  - *Source*: the attribute to represent, using a field or an  *expression*
  - the range of values to represent: you can manually enter the values or use the  Fetch value range from layer button to fill these fields automatically with the minimum and maximum values returned by the *Source* expression applied to your data
-  *Apply transform curve*: by default, output values (see below for setting) are applied to input features following a linear scale. You can override this logic: enable the transform option, click on the graphic to add break point(s) and drag the point(s) to apply a custom distribution.
- Define the *Output* values: the options vary according to the parameter to define. You can globally set:
  - for a color setting, the *color ramp* to apply to values and the single color to use for NULL values
  - for the others, the minimum and maximum values to apply to the selected property as well as the size/angle/opacity value for ignored or NULL source features
  - for size properties, the *Scale method* of representation which can be **Flannery**, **Exponential**, **Surface**, **Radius** or **Linear**
  - the *Exponent* to use for data scaling when the *Scale method* is of exponential type or when tweaking the opacity

When compatible with the property, a live-update preview is displayed in the right-hand side of the dialog to help you control the value scaling.

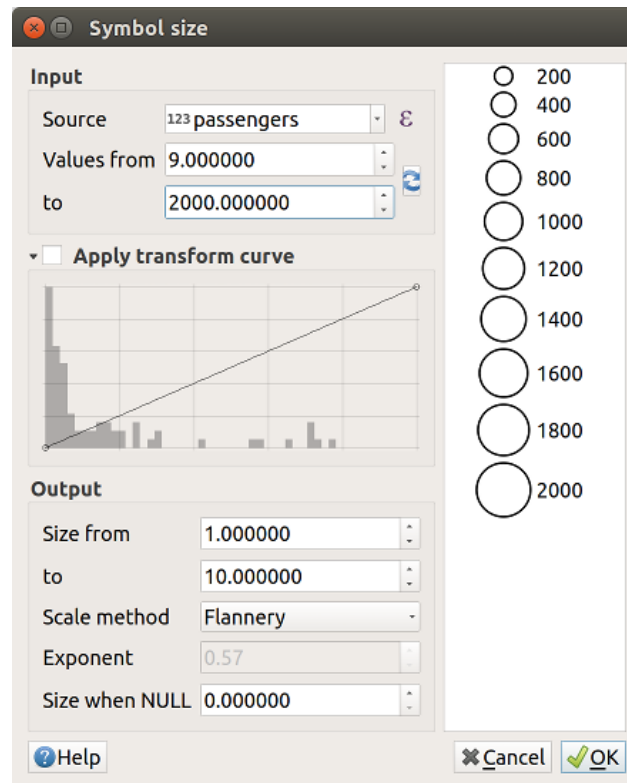


Abb. 12.29: Scaling feature size based on passengers field's value

The values presented in the varying size assistant above will set the size 'Data-defined override' with:

```
coalesce(scale_exp("passengers", 9, 2000, 1, 10, 0.57), 0)
```





---

## Level up mit Ausdrücken

---

### 13.1 Ausdrücke

Based on layer data and prebuilt or user defined functions, **Expressions** offer a powerful way to manipulate attribute value, geometry and variables in order to dynamically change the geometry style, the content or position of the label, the value for diagram, the height of a layout item, select some features, create virtual field, ...





---

**Bemerkung:** A list of the default functions and variables for writing expressions can be found at [Liste der Funktionen](#), with detailed information and examples.

---

#### 13.1.1 Der Ausdruckseditor

Der Hauptdialog zur Erstellung von Ausdrücken, der *Ausdruckeditor* ist an vielen Stellen in QGIS verfügbar. Er kann z.B. hier aufgerufen werden:

- durch Klick auf den  Knopf;
- *Auswahl von Objekten* mit Hilfe des Tools  Objekte über Ausdruck wählen;
- *Attribute bearbeiten* z.B. mit Hilfe des Tools  Feldrechner;
- Änderung der Symbolisierung, der Beschriftung oder der Parameter von Layoutelementen mit Hilfe des Tools  Datendefinierte Übersteuerung (see *Datendefinierte Übersteuerung Setup*);
- Erstellung eines Layers mit Symbolisierung mit Hilfe des *Geometriegenerators*;
- Ausführung einer *Geoprozessierung*.

Der Ausdruckseditor erlaubt den Zugang zu:

- *Expression* tab which, thanks to a list of *predefined functions*, helps to write and check the expression to use;
- den *Funktionseditor* der bei der Erstellung angepasster Funktionen und damit der Erweiterung der Liste der Funktionen unterstützt.

## The Interface

The *Expression* tab provides the main interface to write expressions using functions, layer fields and values. It contains the following widgets:

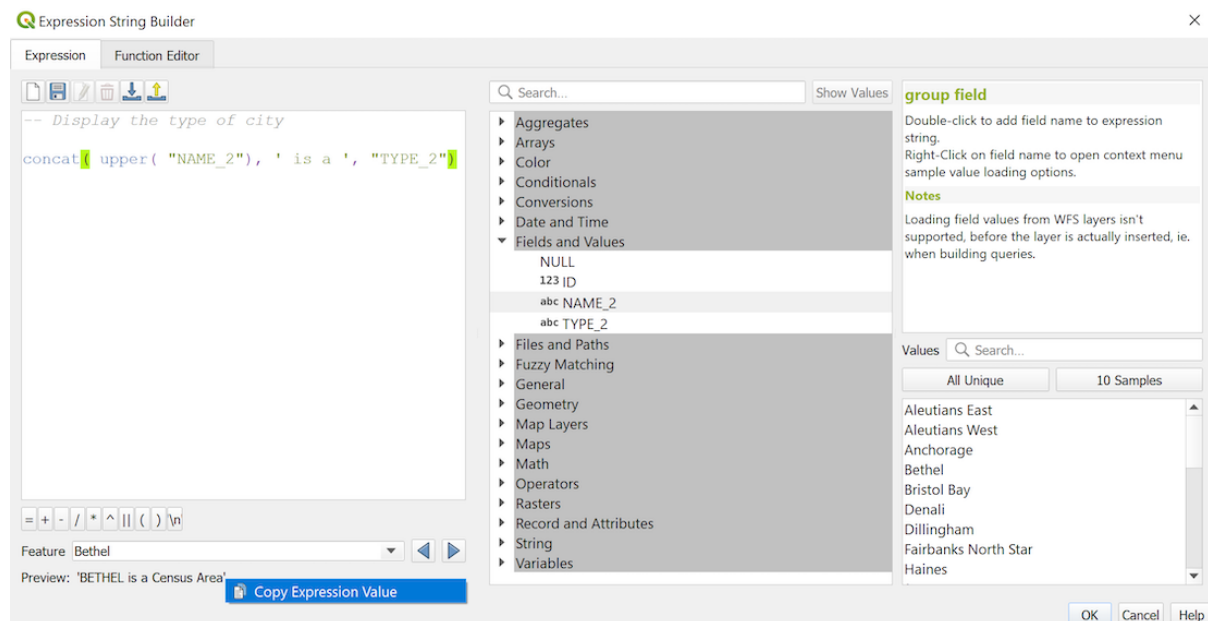


Abb. 13.1: Der Ausdruck Reiter

- An expression editor area for typing or pasting expressions. Autocompletion is available to speed expression writing:
  - Die zum Eingabetext passenden Variablen, Funktionsnamen und Feldnamen werden unterhalb der Eingabe angezeigt. Nutzen Sie die Up und Down Tasten, um die Einträge durchzublätern und die Taste Tab, um einen Eintrag in den Ausdruck zu übernehmen oder klicken Sie einfach doppelt auf den gewünschten Eintrag.
  - Die Parameter von Funktionen werden beim Ausfüllen angezeigt.
- QGIS prüft die Richtigkeit des Ausdrucks und hinterlegt alle Fehler farbig:
  - *Unterstreich*ung: unbekannte Funktionen, falsche oder ungültige Funktionsargumente;
  - *Markierung*: alle weiteren Fehler (z.B. fehlende Klammern, unerwartete Zeichen) an der jeweiligen Stelle.

### **Tipp: Dokumentieren Sie Ihre Ausdrücke mit Kommentaren**

Bei Nutzung komplexer Ausdrücke, ist es eine gute Praxis mehrzeilige Kommentare bzw. im Ausdruck eingeschlossene Kommentare als Erinnerungshilfen zu verwenden.

```
/*
Labels each region with its highest (in altitude) airport(s)
and altitude, eg 'AMBLER : 264m' for the 'Northwest Artic' region
*/
with_variable(
  'airport_alti', -- stores the highest altitude of the region
  aggregate(
    'airports',
    'max',
    "ELEV", -- the field containing the altitude
    -- and limit the airports to the region they are within
    filter := within( $geometry, geometry( @parent ) )
  )
)
```

(Fortsetzung auf der nächsten Seite)


(Fortsetzung der vorherigen Seite)

```


),
aggregate( -- finds airports at the same altitude in the region
  'airports',
  'concatenate',
  "NAME",
  filter := within( $geometry, geometry( @parent ) )
    and "ELEV" = @airport_alti
)
|| ' : ' || @airport_alti || 'm'
-- using || allows regions without airports to be skipped
)

```

- Above the expression editor, a set of tools helps you:

-  Clear the expression editor
- Create and manage *user expressions*

- Under the expression editor, you find:

- A set of basic operators to help you build the expression
- An indication of the expected format of output when you are data-defining feature properties
- A live *Output preview* of the expression (up to 60 characters), evaluated on the first feature of the Layer by default. To view output preview text exceeding 60 characters, you can hover your cursor over the text to display a tooltip pop-up containing the entire output preview. To copy the output preview text onto your clipboard, right-click on the output preview text and select  *Copy Expression Value*.

You can browse and evaluate other features of the layer using the *Feature* combobox (the values are taken from the *display name* property of the layer).

In case of error, it indicates it and you can access the details with the provided hyperlink.

- In einem Auswahlfeld sind die Funktionen, Variablen, Felder, ... in Gruppen organisiert aufgelistet. Ein Suchfeld dient zur Filterung der Gesamtliste, um schnell eine bestimmte Funktion oder Feld zu finden. Mit Hilfe eines Doppelklicks wird das Element in den Ausdruckseditor eingefügt.
- Ein Bedienfeld, das Hilfe für jedes ausgewählte Element aus der Funktionsauwahl anzeigt.


---

**Tipp:** Drücke **Ctrl+Click** beim Überfahren eines Funktionsnamens, um automatisch die Hilfe zu dieser Funktion im Hilfe-Dialog anzuzeigen.

---

A field's values widget shown when a field is selected in the function selector helps to fetch features attributes:

- Look for a particular field value
- Display the list of *All Unique* or *10 Samples* values. Also available from right-click.

When the field is mapped with another layer or a set of values, i.e. if the *field widget* is of *RelationReference*, *ValueRelation* or *ValueMap* type, it's possible to list all the values of the mapped field (from the referenced layer, table or list). Moreover, you can filter this list to  *Only show values in use* in the current field.

Double-clicking a field value in the widget adds it to the expression editor.



---

**Tipp:** Das rechte Bedienfeld mit der Hilfe zu Funktionen und den Feldwerten kann ausgeblendet werden (durch nach rechts ziehen der Bedienfeldbegrenzung). Durch Drücken des Knopfes *Werte anzeigen* oder *Hilfe anzeigen* wird er wieder angezeigt.

---

## Writing an expression

QGIS expressions are used to select features or set values. Writing an expression in QGIS follows some rules:

1. **The dialog defines the context:** if you are used to SQL, you probably know queries of the type *select features from layer where condition* or *update layer set field = new\_value where condition*. A QGIS expression also needs all these information but the tool you use to open the expression builder dialog provides parts of them. For example, giving a layer (buildings) with a field (height):
  - pressing the  Select by expression tool means that you want to „select features from buildings“. The **condition** is the only information you need to provide in the expression text widget, e.g. type "height" > 20 to select buildings that are higher than 20.
  - with this selection made, pressing the  Field calculator button and choosing „height“ as *Update existing field*, you already provide the command „update buildings set height = ??? where height > 20“. The only remaining bits you have to provide in this case is the **new value**, e.g. just enter 50 in the expression editor textbox to set the height of the previously selected buildings.
2. **Pay attention to quotes:** single quotes return a literal, so a text placed between single quotes ('145') is interpreted as a string. Double quotes will give you the value of that text so use them for fields ("myfield"). Fields can also be used without quotes (myfield). No quotes for numbers (3.16).

**Bemerkung:** Functions normally take as argument a string for field name. Do:

```
attribute( @atlas_feature, 'height' ) -- returns the value stored in the
↪ "height" attribute of the current atlas feature
```

And not:

```
attribute( @atlas_feature, "height" ) -- fetches the value of the attribute
↪ named "height" (e.g. 100), and use that value as a field
                                -- from which to return the atlas
↪ feature value. Probably wrong as a field named "100" may not exist.
```

### Tipp: Use named parameters to ease expression reading

Some functions require many parameters to be set. The expression engine supports the use of named parameters. This means that instead of writing the cryptic expression `clamp( 1, 2, 9)`, you can use `clamp( min:=1, value:=2, max:=9)`. This also allows arguments to be switched, e.g. `clamp( value:=2, max:=9, min:=1)`. Using named parameters helps clarify what the arguments for an expression function refer to, which is helpful when you are trying to interpret an expression later!

## Some use cases of expressions

- From the Field Calculator, calculate a „pop\_density“ field using the existing „total\_pop“ and „area\_km2“ fields:

```
"total_pop" / "area_km2"
```

- Label or categorize features based on their area:

```
CASE WHEN $area > 10 000 THEN 'Larger' ELSE 'Smaller' END
```

- Aktualisieren Sie das Feld „density\_level“ mit zugehörigen Kategorien zu den „pop\_density“ Werten:

```
CASE WHEN "pop_density" < 50 THEN 'Low population density'
      WHEN "pop_density" >= 50 and "pop_density" < 150 THEN 'Medium population_
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```

↩density'
  WHEN "pop_density" >= 150 THEN 'High population density'
END

```

- Wenden Sie einen kategorisierten Stil auf alle Objekte an, je nachdem, ob der durchschnittliche Hauspreis kleiner oder größer als 10000€ pro Quadratmeter ist:

```
"price_m2" > 10000
```

- Mit dem „Durch Ausdruck wählen...“ Werkzeug, wählen Sie alle Objekte die eine Fläche mit „Hoher Bevölkerungsdichte“ und einem durchschnittlichen Hauspreis größer als 10000€ pro Quadratmeter darstellen:

```
"density_level" = 'High population density' and "price_m2" > 10000
```

The previous expression could also be used to define which features to label or show on the map.

- Create a different symbol (type) for the layer, using the geometry generator:

```
point_on_surface( $geometry )
```

- Given a point feature, generate a closed line (using make\_line) around its geometry:

```

make_line(
  -- using an array of points placed around the original
  array_foreach(
    -- list of angles for placing the projected points (every 90°)
    array:=generate_series( 0, 360, 90 ),
    -- translate the point 20 units in the given direction (angle)
    expression:=project( $geometry, distance:=20, azimuth:=radians( @element )
  )
)

```


- In a print layout label, display the name of the „airports“ features that are within the layout „Map 1“ item:

```





with_variable( 'extent',
  map_get( item_variables( 'Map 1' ), 'map_extent' ),
  aggregate( 'airports', 'concatenate', "NAME",
    intersects( $geometry, @extent ), ' , '
  )
)


```

## Saving Expressions

Using the  Add current expression to user expressions button above the expression editor frame, you can save important expressions you want to have quick access to. These are available from the **User expressions** group in the middle panel. They are saved under the *user profile* (<userprofile>/QGIS/QGIS3.ini file) and available in all expression dialogs inside all projects of the current user profile.

A set of tools available above the expression editor frame helps you manage the user expressions:

-  Add the current expression to user expressions: store the expression in the user profile. A label and a help text can be added for easy identification.
-  Edit selected expression from user expressions, as well as their help and label
-  Remove selected expression from user expressions
-  Import user expressions from a .json file into the active user profile folder

-  Export user expressions as a .json file; all the user expressions in the user profile QGIS3.ini file are shared

## 13.1.2 Funktions Editor

Mit dem Reiter *Funktionseditor* ist man in der Lage, eigene Funktionen mit Python zu schreiben. Dies bietet einen einfachen und komfortablen Weg, um besondere Anforderungen zu erfüllen, die nicht durch die enthaltenen Funktionen abgedeckt werden.

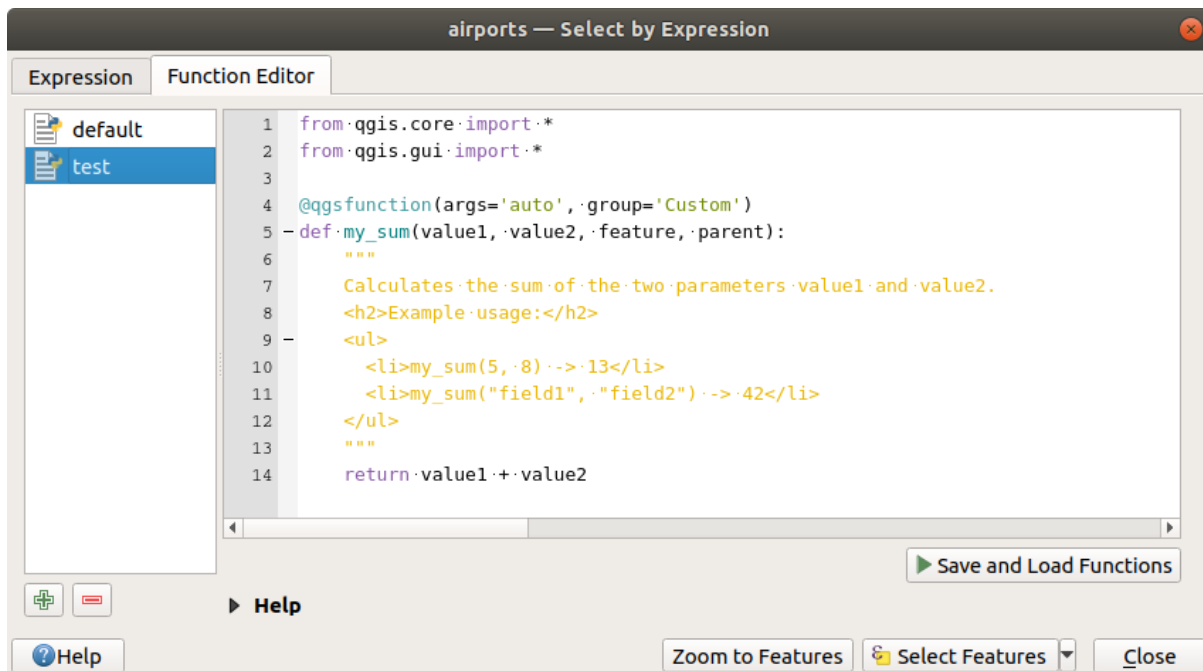






Abb. 13.2: Der Funktionseditor Reiter

Um eine neue Funktion zu erstellen:

1. Drücken Sie auf die  **New File** Schaltfläche.
2. Geben Sie einen Namen in das sich öffnende Formular ein und drücken dann auf **OK**.  
Ein neues Element mit dem vergebenen Namen wird nun im linken Bedienfeld des *Funktionseditor* bereitgestellt; dabei handelt es sich um eine Python .py Datei, basierend auf einer QGIS Template-Datei, gespeichert im Ordner /python/expressions im Verzeichnis des aktiven :ref:`user profile`.
3. Das rechte Bedienfeld stellt den Inhalt der Datei dar: ein Python Skript Template. Aktualisieren den Code und die Hilfeeinträge entsprechend ihrer Erfordernisse.
4. Drücken Sie auf die Schaltfläche  **Lade- und Speicherfunktionen**. Die von ihnen geschriebene Funktion wird in der Liste der Funktionen im Reiter *Ausdruck* in der Voreinstellung in der Gruppe *Custom* hinzugefügt.
5. Haben Sie Spaß mit ihrer neuen Funktion.
6. Wenn die Funktion verbessert werden soll, geht man zum Reiter *Funktionseditor*, nimmt die Änderungen vor und drückt erneut auf die Schaltfläche  **Lade- und Speicherfunktionen**. Damit werden die Änderungen in der Datei und somit auch im Reiter *Ausdruck* verfügbar.

Benutzerdefinierte Pythonfunktionen werden im Profilverzeichnis des Nutzer gespeichert. Das heißt das QGIS beim Start automatische alle Funktionen des aktuellen Nutzerprofils lädt. Beachten Sie, dass neue Funktionen nur im Ordner /python/expressions gespeichert werden und nicht in der Projektdatei. Wenn Sie ein Projekt mit benutzerdefinierten Funktionen verteilen möchten, müssen Sie auch die Datei .py im Ordner /python/expressions mitliefern.

To delete a custom function:

1. Enable the *Function Editor* tab
2. Select the function in the list
3. Press the  Remove selected function. The function is removed from the list and the corresponding .py file deleted from the user profile folder.

### Example

Here's a short example on how to create your own `my_sum` function that will operate with two values.

```
from qgis.core import *
from qgis.gui import *

@qgsfunction(args='auto', group='Custom')
def my_sum(value1, value2, feature, parent):
    """
    Calculates the sum of the two parameters value1 and value2.
    <h2>Example usage:</h2>
    <ul>
        <li>my_sum(5, 8) -> 13</li>
        <li>my_sum("field1", "field2") -> 42</li>
    </ul>
    """
    return value1 + value2
```

The `@qgsfunction` decorator accepts the following arguments:

- `args`: the number of arguments. When using the `args='auto'` argument the number of function arguments required will be calculated by the number of arguments the function has been defined with in Python (minus 2 - `feature`, and `parent`). With `args = -1`, any number of arguments are accepted.
- The `group` argument indicates the group in which the function should be listed in the Expression dialog.
- `usesgeometry=True` if the expression requires access to the features geometry. By default `False`.
- `handlesnull=True` if the expression has custom handling for NULL values. If `False` (default), the result will always be NULL as soon as any parameter is NULL.
- `referenced_columns=[list]`: An array of attribute names that are required to the function. Defaults to `[QgsFeatureRequest.ALL_ATTRIBUTES]`.

The function itself allows following arguments:

- any number and type of parameters you want to pass to your function, set before the following arguments.
- `feature`: the current feature
- `parent`: the `QgsExpression` object
- `context`: If there is an argument called `context` found at the last position, this variable will contain a `QgsExpressionContext` object, that gives access to various additional information like expression variables. E.g. `context.variable( 'layer_id' )`

The previous example function can then be used in expressions:

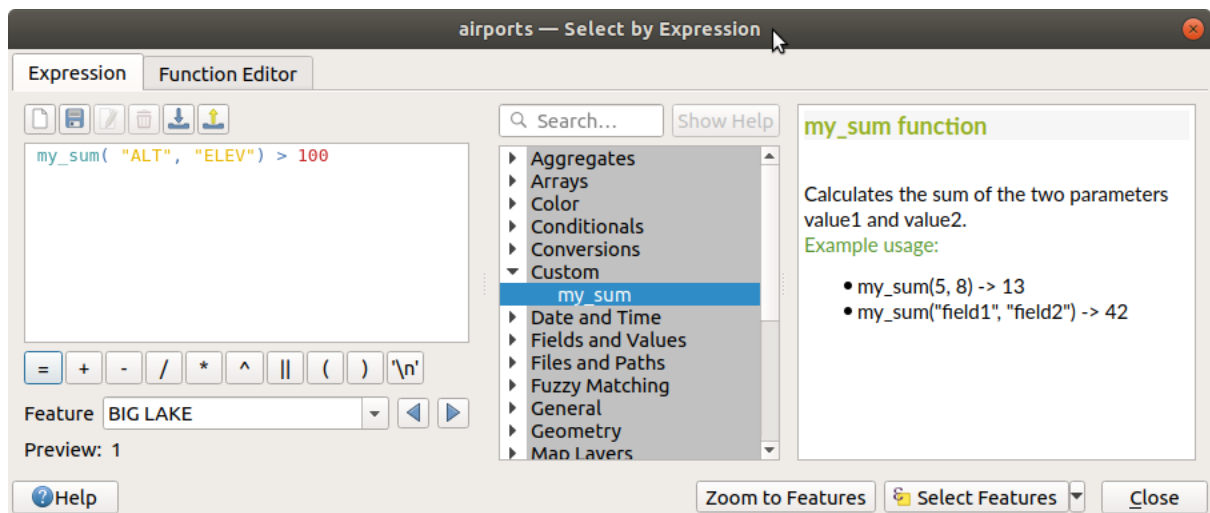


Abb. 13.3: Benutzerdefinierte Funktion im Reiter Ausdruck

Weitere Informationen über das Erstellen von Python-Kodes können in PyQGIS-Developer-Cookbook gefunden werden.

## 13.2 Liste der Funktionen

The functions, operators and variables available in QGIS are listed below, grouped by categories.

### 13.2.1 Aggregatsfunktionen

Diese Gruppe enthält Funktionen, die Werte über Layern und Feldern aggregieren.

#### **aggregate**

Returns an aggregate value calculated using features from another layer.



Syntax	aggregate(layer, aggregate, expression, [filter], [concatenator=""], [order_by]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - a string, representing either a layer name or layer ID</li> <li>• <b>aggregate</b> - a string corresponding to the aggregate to calculate. Valid options are: <ul style="list-style-type: none"> <li>- count</li> <li>- count_distinct</li> <li>- count_missing</li> <li>- min</li> <li>- max</li> <li>- sum</li> <li>- mean</li> <li>- median</li> <li>- stdev</li> <li>- stdevsample</li> <li>- range</li> <li>- minority</li> <li>- majority</li> <li>- q1: first quartile</li> <li>- q3: third quartile</li> <li>- iqr: inter quartile range</li> <li>- min_length: minimum string length</li> <li>- max_length: maximum string length</li> <li>- concatenate: join strings with a concatenator</li> <li>- concatenate_unique: join unique strings with a concatenator</li> <li>- collect: create an aggregated multipart geometry</li> <li>- array_agg: create an array of aggregated values</li> </ul> </li> <li>• <b>expression</b> - sub expression or field name to aggregate</li> <li>• <b>filter</b> - optional filter expression to limit the features used for calculating the aggregate. Fields and geometry are from the features on the joined layer. The source feature can be accessed with the variable @parent.</li> <li>• <b>concatenator</b> - optional string to use to join values for ,concatenate' and ,concatenate_unique' aggregates</li> <li>• <b>order_by</b> - optional filter expression to order the features used for calculating the aggregate. Fields and geometry are from the features on the joined layer. By default, the features will be returned in an unspecified order.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• aggregate(layer:='rail_stations', aggregate:='sum', expression:="passengers") → sum of all values from the passengers field in the rail_stations layer</li> <li>• aggregate('rail_stations', 'sum', "passengers"/7) → calculates a daily average of „passengers“ by dividing the „passengers“ field by 7 before summing the values</li> <li>• aggregate(layer:='rail_stations', aggregate:='sum', expression:="passengers", filter:="class"&gt;3) → sums up all values from the „passengers“ field from features where the „class“ attribute is greater than 3 only</li> <li>• aggregate(layer:='rail_stations', aggregate:='concatenate', expression:="name", concatenator:=',') → comma separated list of the name field for all features in the rail_stations layer</li> <li>• aggregate(layer:='countries', aggregate:='max', expression:="code", filter:=intersects(@geometry, geometry(@parent))) → The country code of an intersecting country on the layer ,countries'</li> <li>• aggregate(layer:='rail_stations', aggregate:='sum', expression:="passengers", filter:=contains(@atlas_geometry, @geometry)) → sum of all values from the passengers field in the rail_stations within the current atlas feature</li> <li>• aggregate(layer:='rail_stations', aggregate:='collect', expression:=centroid(@geometry), filter:="region_name" attribute(@parent, 'name')) → aggregates centroid geometries of the rail_stations of the same region as current feature</li> </ul>

## array\_agg

Returns an array of aggregated values from a field or expression.

Syntax	<code>array_agg(expression, [group_by], [filter], [order_by])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> <li>• <b>order_by</b> - optional expression to use to order features used to calculate aggregate. By default, the features will be returned in an unspecified order.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_agg("name", group_by:="state")</code> → list of name values, grouped by state field</li> </ul>

## collect

Liefert eine mehrteilige Geometrie der aggregierten Geometrien des Ausdrucks

Syntax	<code>collect(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - geometry expression to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>collect(@geometry)</code> → multipart geometry of aggregated geometries</li> <li>• <code>collect(centroid(@geometry), group_by:="region", filter:="use" = 'civilian')</code> → aggregated centroids of the civilian features based on their region value</li> </ul>

## concatenate

Returns all aggregated strings from a field or expression joined by a delimiter.

Syntax	<code>concatenate(expression, [group_by], [filter], [concatenator], [order_by])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> <li>• <b>concatenator</b> - optional string to use to join values. Empty by default.</li> <li>• <b>order_by</b> - optional expression to use to order features used to calculate aggregate. By default, the features will be returned in an unspecified order.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>concatenate("town_name", group_by:="state", concatenator:',')</code> → comma separated list of town_names, grouped by state field</li> </ul>

### concatenate\_unique

Returns all unique strings from a field or expression joined by a delimiter.

Syntax	<code>concatenate_unique(expression, [group_by], [filter], [concatenator], [order_by])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> <li>• <b>concatenator</b> - optional string to use to join values. Empty by default.</li> <li>• <b>order_by</b> - optional expression to use to order features used to calculate aggregate. By default, the features will be returned in an unspecified order.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>concatenate_unique("town_name", group_by:="state", concatenator:=", ")</code> → comma separated list of unique town_names, grouped by state field</li> </ul>

### count

Returns the count of matching features.

Syntax	<code>count(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>count("stations", group_by:="state")</code> → count of stations, grouped by state field</li> </ul>

### count\_distinct

Returns the count of distinct values.

Syntax	<code>count_distinct(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>count_distinct("stations", group_by:="state")</code> → count of distinct stations values, grouped by state field</li> </ul>

### count\_missing

Returns the count of missing (NULL) values.

Syntax	count_missing(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>count_missing("stations", group_by:="state")</code> → count of missing (NULL) station values, grouped by state field</li> </ul>

### iqr

Returns the calculated inter quartile range from a field or expression.

Syntax	iqr(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>iqr("population", group_by:="state")</code> → inter quartile range of population value, grouped by state field</li> </ul>

### majority

Returns the aggregate majority of values (most commonly occurring value) from a field or expression.

Syntax	majority(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>majority("class", group_by:="state")</code> → most commonly occurring class value, grouped by state field</li> </ul>

## max\_length

Returns the maximum length of strings from a field or expression.

Syntax	<code>max_length(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>max_length("town_name", group_by:="state")</code> → maximum length of town_name, grouped by state field</li> </ul>

## maximum

Returns the aggregate maximum value from a field or expression.

Syntax	<code>maximum(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>maximum("population", group_by:="state")</code> → maximum population value, grouped by state field</li> </ul>

## mean

Returns the aggregate mean value from a field or expression.

Syntax	<code>mean(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>mean("population", group_by:="state")</code> → mean population value, grouped by state field</li> </ul>

### median

Returns the aggregate median value from a field or expression.

Syntax	median(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• median("population", group_by:="state") → median population value, grouped by state field</li> </ul>

### min\_length

Returns the minimum length of strings from a field or expression.

Syntax	min_length(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• min_length("town_name", group_by:="state") → minimum length of town_name, grouped by state field</li> </ul>

### minimum

Returns the aggregate minimum value from a field or expression.

Syntax	minimum(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• minimum("population", group_by:="state") → minimum population value, grouped by state field</li> </ul>

## minority

Returns the aggregate minority of values (least occurring value) from a field or expression.

Syntax	<code>minority(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>minority("class", group_by:="state")</code> → least occurring class value, grouped by state field</li> </ul>

## q1

Returns the calculated first quartile from a field or expression.

Syntax	<code>q1(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>q1("population", group_by:="state")</code> → first quartile of population value, grouped by state field</li> </ul>

## q3

Returns the calculated third quartile from a field or expression.

Syntax	<code>q3(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>q3("population", group_by:="state")</code> → third quartile of population value, grouped by state field</li> </ul>

### range

Returns the aggregate range of values (maximum - minimum) from a field or expression.

Syntax	<code>range(expression, [group_by], [filter])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"><li>• <b>expression</b> - sub expression of field to aggregate</li><li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li><li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>range("population", group_by:="state")</code> → range of population values, grouped by state field</li></ul>

### relation\_aggregate

Returns an aggregate value calculated using all matching child features from a layer relation.



Syntax	<code>relation_aggregate(relation, aggregate, expression, [concatenator=""], [order_by])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>relation</b> - a string, representing a relation ID</li> <li>• <b>aggregate</b> - a string corresponding to the aggregate to calculate. Valid options are: <ul style="list-style-type: none"> <li>- count</li> <li>- count_distinct</li> <li>- count_missing</li> <li>- min</li> <li>- max</li> <li>- sum</li> <li>- mean</li> <li>- median</li> <li>- stdev</li> <li>- stdevsample</li> <li>- range</li> <li>- minority</li> <li>- majority</li> <li>- q1: first quartile</li> <li>- q3: third quartile</li> <li>- iqr: inter quartile range</li> <li>- min_length: minimum string length</li> <li>- max_length: maximum string length</li> <li>- concatenate: join strings with a concatenator</li> <li>- concatenate_unique: join unique strings with a concatenator</li> <li>- collect: create an aggregated multipart geometry</li> <li>- array_agg: create an array of aggregated values</li> </ul> </li> <li>• <b>expression</b> - sub expression or field name to aggregate</li> <li>• <b>concatenator</b> - optional string to use to join values for ,concatenate' aggregate</li> <li>• <b>order_by</b> - optional expression to order the features used for calculating the aggregate. Fields and geometry are from the features on the joined layer. By default, the features will be returned in an unspecified order.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>relation_aggregate(relation:='my_relation', aggregate:='mean', expression:="passengers")</code> → mean value of all matching child features using the ,my_relation' relation</li> <li>• <code>relation_aggregate('my_relation', 'sum', "passengers"/7)</code> → sum of the passengers field divided by 7 for all matching child features using the ,my_relation' relation</li> <li>• <code>relation_aggregate('my_relation', 'concatenate', "towns", concatenator:=', ')</code> → comma separated list of the towns field for all matching child features using the ,my_relation' relation</li> <li>• <code>relation_aggregate('my_relation', 'array_agg', "id")</code> → array of the id field from all matching child features using the ,my_relation' relation</li> </ul>

Further reading: *Setting relations between multiple layers*

## stdev

Returns the aggregate standard deviation value from a field or expression.

Syntax	stdev(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>stdev("population", group_by:="state")</code> → standard deviation of population value, grouped by state field</li> </ul>

## sum

Returns the aggregate summed value from a field or expression.

Syntax	sum(expression, [group_by], [filter]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - sub expression of field to aggregate</li> <li>• <b>group_by</b> - optional expression to use to group aggregate calculations</li> <li>• <b>filter</b> - optional expression to use to filter features used to calculate aggregate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>sum("population", group_by:="state")</code> → summed population value, grouped by state field</li> </ul>

## 13.2.2 Matrixfunktionen

Diese Gruppe enthält Funktionen zur Erzeugung und Manipulation von Arrays (auch als Listen bekannt). Die Reihenfolge der Werte im Array ist wichtig. Eine Ausnahme bildet die ‚map‘ Datenstruktur, bei der die Reihenfolge der Schlüssel-Werte-Paare unwichtig ist. Die Werte werden in diesem Fall durch den zugehörigen Schlüssel identifiziert.

## array

Returns an array containing all the values passed as parameter.

Syntax	array(value1, value2, ...)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array(2, 10)</code> → [ 2, 10 ]</li> <li>• <code>array(2, 10) [0]</code> → 2</li> </ul>

### array\_all

Returns TRUE if an array contains all the values of a given array.

Syntax	<code>array_all(array_a, array_b)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array_a</b> - an array</li> <li>• <b>array_b</b> - the array of values to search</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_all(array(1,2,3), array(2,3)) → TRUE</code></li> <li>• <code>array_all(array(1,2,3), array(1,2,4)) → FALSE</code></li> </ul>

### array\_append

Returns an array with the given value added at the end.

Syntax	<code>array_append(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the value to add</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_append(array(1,2,3), 4) → [ 1, 2, 3, 4 ]</code></li> </ul>

### array\_cat

Returns an array containing all the given arrays concatenated.

Syntax	<code>array_cat(array1, array2, ...)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_cat(array(1,2), array(2,3)) → [ 1, 2, 2, 3 ]</code></li> </ul>

### array\_contains

Returns TRUE if an array contains the given value.

Syntax	<code>array_contains(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the value to search</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_contains(array(1,2,3), 2) → TRUE</code></li> </ul>

### array\_count

Counts the number of occurrences of a given value in an array.

Syntax	<code>array_count(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the value to count</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_count(array('a', 'b', 'c', 'b'), 'b') → 2</code></li> </ul>

### array\_distinct

Returns an array containing distinct values of the given array.

Syntax	<code>array_distinct(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_distinct(array(1, 2, 3, 2, 1)) → [ 1, 2, 3 ]</code></li> </ul>

### array\_filter

Returns an array with only the items for which the expression evaluates to true.

Syntax	<code>array_filter(array, expression, [limit=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>expression</b> - an expression to evaluate on each item. The variable <i>@element</i> will be replaced by the current value.</li> <li>• <b>limit</b> - maximum number of elements to be returned. Use 0 to return all values.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_filter(array(1, 2, 3), @element &lt; 3) → [ 1, 2 ]</code></li> <li>• <code>array_filter(array(1, 2, 3), @element &lt; 3, 1) → [ 1 ]</code></li> </ul>

### array\_find

Returns the lowest index (0 for the first one) of a value within an array. Returns -1 if the value is not found.

Syntax	<code>array_find(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the value to search</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_find(array('a', 'b', 'c'), 'b') → 1</code></li> <li>• <code>array_find(array('a', 'b', 'c', 'b'), 'b') → 1</code></li> </ul>

### array\_first

Returns the first value of an array.

Syntax	<code>array_first(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_first(array('a','b','c')) → ,a‘</code></li> </ul>

### array\_foreach

Returns an array with the given expression evaluated on each item.

Syntax	<code>array_foreach(array, expression)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>expression</b> - an expression to evaluate on each item. The variable <i>@element</i> will be replaced by the current value.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_foreach(array('a','b','c'), upper(@element)) → [ ,A‘, ,B‘, ,C‘ ]</code></li> <li>• <code>array_foreach(array(1,2,3), @element + 10) → [ 11, 12, 13 ]</code></li> </ul>

### array\_get

Returns the Nth value (0 for the first one) or the last -Nth value (-1 for the last one) of an array.

Syntax	<code>array_get(array, pos)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>pos</b> - the index to get (0 based)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_get(array('a','b','c'), 1) → ,b‘</code></li> <li>• <code>array_get(array('a','b','c'), -1) → ,c‘</code></li> </ul>

**Hinweis:** You can also use the *index operator* (`[]`) to get a value from an array.

### array\_insert

Returns an array with the given value added at the given position.

Syntax	<code>array_insert(array, pos, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>pos</b> - the position where to add (0 based)</li> <li>• <b>value</b> - the value to add</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_insert(array(1,2,3),1,100) → [ 1, 100, 2, 3 ]</code></li> </ul>

### array\_intersect

Returns TRUE if at least one element of array1 exists in array2.

Syntax	<code>array_intersect(array1, array2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array1</b> - an array</li> <li>• <b>array2</b> - another array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_intersect(array(1,2,3,4),array(4,0,2,5)) → TRUE</code></li> </ul>

### array\_last

Returns the last value of an array.

Syntax	<code>array_last(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_last(array('a','b','c')) → ,c'</code></li> </ul>

### array\_length

Returns the number of elements of an array.

Syntax	<code>array_length(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_length(array(1,2,3)) → 3</code></li> </ul>

## array\_majority

Returns the most common values in an array.

Syntax	<code>array_majority(array, [option='all'])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>option='all'</b> - a string specifying the return values handling. Valid options are: <ul style="list-style-type: none"> <li>– all: Default, all most common values are returned in an array.</li> <li>– any: Returns one of the most common values.</li> <li>– median: Returns the median of the most common values. Non arithmetic values are ignored.</li> <li>– real_majority: Returns the value which occurs more than half the size of the array.</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_majority(array(0,1,42,42,43), 'all') → [ 42 ]</code></li> <li>• <code>array_majority(array(0,1,42,42,43,1), 'all') → [ 42, 1 ]</code></li> <li>• <code>array_majority(array(0,1,42,42,43,1), 'any') → 1 or 42</code></li> <li>• <code>array_majority(array(0,1,1,2,2), 'median') → 1.5</code></li> <li>• <code>array_majority(array(0,1,42,42,43), 'real_majority') → NULL</code></li> <li>• <code>array_majority(array(0,1,42,42,43,42), 'real_majority') → NULL</code></li> <li>• <code>array_majority(array(0,1,42,42,43,42,42), 'real_majority') → 42</code></li> </ul>

## array\_max

Returns the maximum value of an array.

Syntax	<code>array_max(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_max(array(0,42,4,2)) → 42</code></li> </ul>

## array\_mean

Returns the mean of arithmetic values in an array. Non numeric values in the array are ignored.

Syntax	<code>array_mean(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_mean(array(0,1,7,66.6,135.4)) → 42</code></li> <li>• <code>array_mean(array(0,84,'a','b','c')) → 42</code></li> </ul>

## array\_median

Returns the median of arithmetic values in an array. Non arithmetic values in the array are ignored.

Syntax	<code>array_median(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_median(array(0,1,42,42,43)) → 42</code></li> <li>• <code>array_median(array(0,1,2,42,'a','b')) → 1.5</code></li> </ul>

## array\_min

Returns the minimum value of an array.

Syntax	<code>array_min(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_min(array(43,42,54)) → 42</code></li> </ul>

## array\_minority

Returns the less common values in an array.

Syntax	<code>array_minority(array, [option='all'])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>option='all'</b> - a string specifying the return values handling. Valid options are: <ul style="list-style-type: none"> <li>– all: Default, all less common values are returned in an array.</li> <li>– any: Returns one of the less common values.</li> <li>– median: Returns the median of the less common values. Non arithmetic values are ignored.</li> <li>– real_minority: Returns values which occur less than half the size of the array.</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_minority(array(0,42,42), 'all') → [ 0 ]</code></li> <li>• <code>array_minority(array(0,1,42,42), 'all') → [ 0, 1 ]</code></li> <li>• <code>array_minority(array(0,1,42,42,43,1), 'any') → 0 or 43</code></li> <li>• <code>array_minority(array(1,2,3,3), 'median') → 1.5</code></li> <li>• <code>array_minority(array(0,1,42,42,43), 'real_minority') → [ 42, 43, 0, 1 ]</code></li> <li>• <code>array_minority(array(0,1,42,42,43,42), 'real_minority') → [ 42, 43, 0, 1 ]</code></li> <li>• <code>array_minority(array(0,1,42,42,43,42,42), 'real_minority') → [ 43, 0, 1 ]</code></li> </ul>



### array\_prepend

Returns an array with the given value added at the beginning.

Syntax	<code>array_prepend(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the value to add</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_prepend(array(1, 2, 3), 0) → [ 0, 1, 2, 3 ]</code></li> </ul>

### array\_prioritize

Returns an array sorted using the ordering specified in another array. Values which are present in the first array but are missing from the second array will be added to the end of the result.

Syntax	<code>array_prioritize(array, array_prioritize)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>array_prioritize</b> - an array with values ordered by priority</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_prioritize(array(1, 8, 2, 5), array(5, 4, 2, 1, 3, 8)) → [ 5, 2, 1, 8 ]</code></li> <li>• <code>array_prioritize(array(5, 4, 2, 1, 3, 8), array(1, 8, 6, 5)) → [ 1, 8, 5, 4, 2, 3 ]</code></li> </ul>

### array\_remove\_all

Returns an array with all the entries of the given value removed.

Syntax	<code>array_remove_all(array, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>value</b> - the values to remove</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_remove_all(array('a', 'b', 'c', 'b'), 'b') → [ ,a', ,c' ]</code></li> </ul>

### array\_remove\_at

Returns an array with the item at the given index removed. Supports positive (0 for the first element) and negative (the last -Nth value, -1 for the last element) index.

Syntax	<code>array_remove_at(array, pos)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>pos</b> - the position to remove (0 based)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_remove_at(array(1, 2, 3), 1) → [1, 3]</code></li> <li>• <code>array_remove_at(array(1, 2, 3), -1) → [1, 2]</code></li> </ul>

### array\_replace

Returns an array with the supplied value, array, or map of values replaced.

#### Value & array variant

Returns an array with the supplied value or array of values replaced by another value or an array of values.

Syntax	<code>array_replace(array, before, after)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - the input array</li> <li>• <b>before</b> - the value or array of values to replace</li> <li>• <b>after</b> - the value or array of values to use as a replacement</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_replace(array('QGIS', 'SHOULD', 'ROCK'), 'SHOULD', 'DOES') → [ ,QGIS', ,DOES', ,ROCK' ]</code></li> <li>• <code>array_replace(array(3, 2, 1), array(1, 2, 3), array(7, 8, 9)) → [ 9, 8, 7 ]</code></li> <li>• <code>array_replace(array('Q', 'G', 'I', 'S'), array('Q', 'S'), '-') → [ ,-', ,G', ,I', ,-' ]</code></li> </ul>

#### Map variant

Returns an array with the supplied map keys replaced by their paired values.

Syntax	<code>array_replace(array, map)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - the input array</li> <li>• <b>map</b> - the map containing keys and values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_replace(array('APP', 'SHOULD', 'ROCK'), map('APP', 'QGIS', 'SHOULD', 'DOES')) → [ ,QGIS', ,DOES', ,ROCK' ]</code></li> </ul>

### array\_reverse

Returns the given array with array values in reversed order.

Syntax	<code>array_reverse(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_reverse(array(2, 4, 0, 10)) → [ 10, 0, 4, 2 ]</code></li> </ul>

## array\_slice

Returns a portion of the array. The slice is defined by the start\_pos and end\_pos arguments.

Syntax	array_slice(array, start_pos, end_pos)
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>start_pos</b> - the index of the start position of the slice (0 based). The start_pos index is included in the slice. If you use a negative start_pos, the index is counted from the end of the list (-1 based).</li> <li>• <b>end_pos</b> - the index of the end position of the slice (0 based). The end_pos index is included in the slice. If you use a negative end_pos, the index is counted from the end of the list (-1 based).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_slice(array(1,2,3,4,5), 0, 3) → [ 1, 2, 3, 4 ]</code></li> <li>• <code>array_slice(array(1,2,3,4,5), 0, -1) → [ 1, 2, 3, 4, 5 ]</code></li> <li>• <code>array_slice(array(1,2,3,4,5), -5, -1) → [ 1, 2, 3, 4, 5 ]</code></li> <li>• <code>array_slice(array(1,2,3,4,5), 0, 0) → [ 1 ]</code></li> <li>• <code>array_slice(array(1,2,3,4,5), -2, -1) → [ 4, 5 ]</code></li> <li>• <code>array_slice(array(1,2,3,4,5), -1, -1) → [ 5 ]</code></li> <li>• <code>array_slice(array('Dufour', 'Valmiera', 'Chugiak', 'Brighton'), 1, 2) → [ ,Valmiera', ,Chugiak' ]</code></li> <li>• <code>array_slice(array('Dufour', 'Valmiera', 'Chugiak', 'Brighton'), -2, -1) → [ ,Chugiak', ,Brighton' ]</code></li> </ul>

## array\_sort

Returns the provided array with its elements sorted.

Syntax	array_sort(array, [ascending=true]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> <li>• <b>ascending</b> - set this parameter to false to sort the array in descending order</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_sort(array(3,2,1)) → [ 1, 2, 3 ]</code></li> </ul>

## array\_sum

Returns the sum of arithmetic values in an array. Non numeric values in the array are ignored.

Syntax	array_sum(array)
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - an array</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_sum(array(0,1,39.4,1.6, 'a')) → 42.0</code></li> </ul>

## array\_to\_string

Concatenates array elements into a string separated by a delimiter and using optional string for empty values.

Syntax	<code>array_to_string(array, [delimiter=','], [empty_value=''])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - the input array</li> <li>• <b>delimiter</b> - the string delimiter used to separate concatenated array elements</li> <li>• <b>empty_value</b> - the optional string to use as replacement for empty (zero length) matches</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_to_string(array('1', '2', '3')) → ,1,2,3'</code></li> <li>• <code>array_to_string(array(1,2,3), '-') → ,1-2-3'</code></li> <li>• <code>array_to_string(array('1', '', '3'), ', ', '0') → ,1,0,3'</code></li> </ul>

## generate\_series

Creates an array containing a sequence of numbers.

Syntax	<code>generate_series(start, stop, [step=1])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>start</b> - first value of the sequence</li> <li>• <b>stop</b> - value that ends the sequence once reached</li> <li>• <b>step</b> - value used as the increment between values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>generate_series(1,5) → [ 1, 2, 3, 4, 5 ]</code></li> <li>• <code>generate_series(5,1,-1) → [ 5, 4, 3, 2, 1 ]</code></li> </ul>

## geometries\_to\_array

Splits a geometry into simpler geometries in an array.

Syntax	<code>geometries_to_array(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - the input geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geometries_to_array(geom_from_wkt('GeometryCollection (Polygon ((5 8, 4 1, 3 2, 5 8)), LineString (3 2, 4 2))')) → an array of a polygon and a line geometries</code></li> <li>• <code>geom_to_wkt(geometries_to_array(geom_from_wkt('GeometryCollection (Polygon ((5 8, 4 1, 3 2, 5 8)), LineString (3 2, 4 2))')))[0] → ,Polygon ((5 8, 4 1, 3 2, 5 8))'</code></li> <li>• <code>geometries_to_array(geom_from_wkt('MULTIPOLYGON(((5 5,0 0, 0 10,5 5)),((5 5,10 10,10 0,5 5))')) → an array of two polygon geometries</code></li> </ul>

## regexp\_matches

Returns an array of all strings captured by capturing groups, in the order the groups themselves appear in the supplied regular expression against a string.

Syntax	<code>regexp_matches(string, regex, [empty_value=""])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to capture groups from against the regular expression</li> <li>• <b>regex</b> - the regular expression used to capture groups</li> <li>• <b>empty_value</b> - the optional string to use as replacement for empty (zero length) matches</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>regexp_matches('QGIS=&gt;rocks', '(.*)=&gt;(.*') → [ ,QGIS', ,rocks' ]</code></li> <li>• <code>regexp_matches('key=&gt;', '(.*)=&gt;(.*', 'empty value') → [ ,key', ,empty value' ]</code></li> </ul>

## string\_to\_array

Splits string into an array using supplied delimiter and optional string for empty values.

Syntax	<code>string_to_array(string, [delimiter=','], [empty_value=""])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the input string</li> <li>• <b>delimiter</b> - the string delimiter used to split the input string</li> <li>• <b>empty_value</b> - the optional string to use as replacement for empty (zero length) matches</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>string_to_array('1,2,3', ',') → [ ,1', ,2', ,3' ]</code></li> <li>• <code>string_to_array('1,,3', ',') → [ ,1', ,', ,3' ]</code></li> </ul>

## 13.2.3 Farbfunktionen

Diese Gruppe enthält Funktionen zur Farbmanipulation.

### color\_cmyk

Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Cyan-, Magenta-, Gelb- und Schwarzkomponenten

Syntax	<code>color_cmyk(cyan, magenta, yellow, black)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>cyan</b> - cyan component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>magenta</b> - magenta component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>yellow</b> - yellow component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>black</b> - black component of the color, as a percentage integer value from 0 to 100</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_cmyk(100, 50, 0, 10) → ,0,115,230'</code></li> </ul>

## color\_cmyka

Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Cyan-, Magenta-, Gelb-, Schwarz- und Alpha-(Transparenz-)komponenten

Syntax	color_cmyka(cyan, magenta, yellow, black, alpha)
Arguments	<ul style="list-style-type: none"> <li>• <b>cyan</b> - cyan component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>magenta</b> - magenta component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>yellow</b> - yellow component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>black</b> - black component of the color, as a percentage integer value from 0 to 100</li> <li>• <b>alpha</b> - alpha component as an integer value from 0 (completely transparent) to 255 (opaque).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• color_cmyka(100, 50, 0, 10, 200) → ,0,115,230,200‘</li> </ul>

## color\_grayscale\_average

Applies a grayscale filter and returns a string representation from a provided color.

Syntax	color_grayscale_average(color)
Arguments	<ul style="list-style-type: none"> <li>• <b>color</b> - a color string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• color_grayscale_average('255,100,50') → ,135,135,135,255‘</li> </ul>

## color\_hsl

Returns a string representation of a color based on its hue, saturation, and lightness attributes.

Syntax	color_hsl(hue, saturation, lightness)
Arguments	<ul style="list-style-type: none"> <li>• <b>hue</b> - hue of the color, as an integer value from 0 to 360</li> <li>• <b>saturation</b> - saturation percentage of the color as an integer value from 0 to 100</li> <li>• <b>lightness</b> - lightness percentage of the color as an integer value from 0 to 100</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• color_hsl(100, 50, 70) → ,166,217,140‘</li> </ul>

## color\_hsla

Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Farb-, Sättigungs-, Helligkeits- und Alpha-(Transparenz-)Attributen

Syntax	<code>color_hsla(hue, saturation, lightness, alpha)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>hue</b> - hue of the color, as an integer value from 0 to 360</li> <li>• <b>saturation</b> - saturation percentage of the color as an integer value from 0 to 100</li> <li>• <b>lightness</b> - lightness percentage of the color as an integer value from 0 to 100</li> <li>• <b>alpha</b> - alpha component as an integer value from 0 (completely transparent) to 255 (opaque).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_hsla(100, 50, 70, 200) → ,166,217,140,200‘</code></li> </ul>

### color\_hsv

Returns a string representation of a color based on its hue, saturation, and value attributes.

Syntax	<code>color_hsv(hue, saturation, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>hue</b> - hue of the color, as an integer value from 0 to 360</li> <li>• <b>saturation</b> - saturation percentage of the color as an integer value from 0 to 100</li> <li>• <b>value</b> - value percentage of the color as an integer from 0 to 100</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_hsv(40, 100, 100) → ,255,170,0‘</code></li> </ul>

### color\_hsva

Returns a string representation of a color based on its hue, saturation, value and alpha (transparency) attributes.

Syntax	<code>color_hsva(hue, saturation, value, alpha)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>hue</b> - hue of the color, as an integer value from 0 to 360</li> <li>• <b>saturation</b> - saturation percentage of the color as an integer value from 0 to 100</li> <li>• <b>value</b> - value percentage of the color as an integer from 0 to 100</li> <li>• <b>alpha</b> - alpha component as an integer value from 0 (completely transparent) to 255 (opaque)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_hsva(40, 100, 100, 200) → ,255,170,0,200‘</code></li> </ul>

### color\_mix\_rgb

Returns a string representing a color mixing the red, green, blue, and alpha values of two provided colors based on a given ratio.

Syntax	<code>color_mix_rgb(color1, color2, ratio)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>color1</b> - a color string</li> <li>• <b>color2</b> - a color string</li> <li>• <b>ratio</b> - a ratio</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_mix_rgb('0,0,0','255,255,255',0.5) → ,127,127,127,255'</code></li> </ul>

## color\_part

Returns a specific component from a color string, e.g., the red component or alpha component.

Syntax	<code>color_part(color, component)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>color</b> - a color string</li> <li>• <b>component</b> - a string corresponding to the color component to return. Valid options are: <ul style="list-style-type: none"> <li>– red: RGB red component (0-255)</li> <li>– green: RGB green component (0-255)</li> <li>– blue: RGB blue component (0-255)</li> <li>– alpha: alpha (transparency) value (0-255)</li> <li>– hue: HSV hue (0-360)</li> <li>– saturation: HSV saturation (0-100)</li> <li>– value: HSV value (0-100)</li> <li>– hsl_hue: HSL hue (0-360)</li> <li>– hsl_saturation: HSL saturation (0-100)</li> <li>– lightness: HSL lightness (0-100)</li> <li>– cyan: CMYK cyan component (0-100)</li> <li>– magenta: CMYK magenta component (0-100)</li> <li>– yellow: CMYK yellow component (0-100)</li> <li>– black: CMYK black component (0-100)</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_part('200,10,30','green') → 10</code></li> </ul>

## color\_rgb

Returns a string representation of a color based on its red, green, and blue components.

Syntax	<code>color_rgb(red, green, blue)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>red</b> - red component as an integer value from 0 to 255</li> <li>• <b>green</b> - green component as an integer value from 0 to 255</li> <li>• <b>blue</b> - blue component as an integer value from 0 to 255</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_rgb(255,127,0) → ,255,127,0'</code></li> </ul>



## color\_rgba

Returns a string representation of a color based on its red, green, blue, and alpha (transparency) components.

Syntax	<code>color_rgba(red, green, blue, alpha)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>red</b> - red component as an integer value from 0 to 255</li> <li>• <b>green</b> - green component as an integer value from 0 to 255</li> <li>• <b>blue</b> - blue component as an integer value from 0 to 255</li> <li>• <b>alpha</b> - alpha component as an integer value from 0 (completely transparent) to 255 (opaque).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>color_rgba(255, 127, 0, 200)</code> → <code>,255,127,0,200'</code></li> </ul>

## create\_ramp

Returns a gradient ramp from a map of color strings and steps.

Syntax	<code>create_ramp(map, [discrete=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map of color strings and steps</li> <li>• <b>discrete</b> - set this parameter to true to create a discrete color ramp</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ramp_color(create_ramp(map(0, '0,0,0', 1, '255,0,0')), 1)</code> → <code>,255,0,0,255'</code></li> </ul>

## darker

Liefert eine dunklere (oder hellere) Farbzeichenkette

Syntax	<code>darker(color, factor)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>color</b> - a color string</li> <li>• <b>factor</b> - an integer corresponding to the darkening factor: <ul style="list-style-type: none"> <li>– if the factor is greater than 100, this function returns a darker color (e.g., setting factor to 200 returns a color that is half the brightness);</li> <li>– if the factor is less than 100, the return color is lighter, but using the <code>lighter()</code> function for this purpose is recommended;</li> <li>– if the factor is 0 or negative, the return value is unspecified.</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>darker('200,10,30', 200)</code> → <code>,100,5,15,255'</code></li> </ul>

Further reading: [lighter](#)

## lighter

Liefert eine hellere (oder dunklere) Farbzeichenkette

Syntax	<code>lighter(color, factor)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>color</b> - a color string</li> <li>• <b>factor</b> - an integer corresponding to the lightening factor: <ul style="list-style-type: none"> <li>– if the factor is greater than 100, this function returns a lighter color (e.g., setting factor to 150 returns a color that is 50% brighter);</li> <li>– if the factor is less than 100, the return color is darker, but using the <code>darker()</code> function for this purpose is recommended;</li> <li>– if the factor is 0 or negative, the return value is unspecified.</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>lighter('200,10,30', 200) → ,255,158,168,255'</code></li> </ul>

Further reading: *darker*

## project\_color

Returns a color from the project's color scheme.

Syntax	<code>project_color(name)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>name</b> - a color name</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>project_color('Logo color') → ,20,140,50'</code></li> </ul>

Further reading: *setting project colors*

## ramp\_color

Returns a string representing a color from a color ramp.

### Saved ramp variant

Returns a string representing a color from a saved ramp

Syntax	<code>ramp_color(ramp_name, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>ramp_name</b> - the name of the color ramp as a string, for example 'Spectral'</li> <li>• <b>value</b> - the position on the ramp to select the color from as a real number between 0 and 1</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ramp_color('Spectral', 0.3) → ,253,190,115,255'</code></li> </ul>

**Bemerkung:** The color ramps available vary between QGIS installations. This function may not give the expected results if you move your QGIS project between installations.

### Expression-created ramp variant

Returns a string representing a color from an expression-created ramp

Syntax	<code>ramp_color(ramp, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>ramp</b> - the color ramp</li> <li>• <b>value</b> - the position on the ramp to select the color from as a real number between 0 and 1</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ramp_color(create_ramp(map(0, '0,0,0', 1, '255,0,0')), 1)</code> → <code>,255,0,0,255'</code></li> </ul>

Further reading: *Setting a Color Ramp*, *The color ramp drop-down shortcut*

## set\_color\_part

Sets a specific color component for a color string, e.g., the red component or alpha component.

Syntax	<code>set_color_part(color, component, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>color</b> - a color string</li> <li>• <b>component</b> - a string corresponding to the color component to set. Valid options are: <ul style="list-style-type: none"> <li>– red: RGB red component (0-255)</li> <li>– green: RGB green component (0-255)</li> <li>– blue: RGB blue component (0-255)</li> <li>– alpha: alpha (transparency) value (0-255)</li> <li>– hue: HSV hue (0-360)</li> <li>– saturation: HSV saturation (0-100)</li> <li>– value: HSV value (0-100)</li> <li>– hsl_hue: HSL hue (0-360)</li> <li>– hsl_saturation: HSL saturation (0-100)</li> <li>– lightness: HSL lightness (0-100)</li> <li>– cyan: CMYK cyan component (0-100)</li> <li>– magenta: CMYK magenta component (0-100)</li> <li>– yellow: CMYK yellow component (0-100)</li> <li>– black: CMYK black component (0-100)</li> </ul> </li> <li>• <b>value</b> - new value for color component, respecting the ranges listed above</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>set_color_part('200,10,30', 'green', 50)</code> → <code>,200,50,30,255'</code></li> </ul>

## 13.2.4 Funktionen für Bedingungen

Diese Gruppe enthält Funktionen um bedingte Prüfungen in Ausdrücken zu handhaben.

### CASE

CASE is used to evaluate a series of conditions and return a result for the first condition met. The conditions are evaluated sequentially, and if a condition is true, the evaluation stops, and the corresponding result is returned. If none of the conditions are true, the value in the ELSE clause is returned. Furthermore, if no ELSE clause is set and none of the conditions are met, NULL is returned.

CASE

WHEN *condition* THEN *result*

[ ...n ]

[ ELSE *result* ]

END

[ ] marks optional components

Arguments	<ul style="list-style-type: none"> <li>• <b>WHEN condition</b> - A condition expression to evaluate</li> <li>• <b>THEN result</b> - If <i>condition</i> evaluates to True then <i>result</i> is evaluated and returned.</li> <li>• <b>ELSE result</b> - If none of the above conditions evaluated to True then <i>result</i> is evaluated and returned.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• CASE WHEN "name" IS NULL THEN 'None' END → Returns the string ‚None‘ if the „name“ field is NULL</li> <li>• CASE WHEN \$area &gt; 10000 THEN 'Big property' WHEN \$area &gt; 5000 THEN 'Medium property' ELSE 'Small property' END → Returns the string ‚Big property‘ if the area is bigger than 10000, ‚Medium property‘ if the area is between 5000 and 10000, and ‚Small property‘ for others</li> </ul>

### coalesce

Returns the first non-NULL value from the expression list.

This function can take any number of arguments.

Syntax	coalesce(expression1, expression2, ...)
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - any valid expression or value, regardless of type.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• coalesce(NULL, 2) → 2</li> <li>• coalesce(NULL, 2, 3) → 2</li> <li>• coalesce(7, NULL, 3*2) → 7</li> <li>• coalesce("fieldA", "fallbackField", 'ERROR') → value of fieldA if it is non-NULL else the value of „fallbackField“ or the string ‚ERROR‘ if both are NULL</li> </ul>

**if**

Tests a condition and returns a different result depending on the conditional check.

Syntax	if(condition, result_when_true, result_when_false)
Arguments	<ul style="list-style-type: none"> <li>• <b>condition</b> - the condition which should be checked</li> <li>• <b>result_when_true</b> - the result which will be returned when the condition is true or another value that does not convert to false.</li> <li>• <b>result_when_false</b> - the result which will be returned when the condition is false or another value that converts to false like 0 or ,. NULL will also be converted to false.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>if( 1+1=2, 'Yes', 'No' ) → ,Yes‘</code></li> <li>• <code>if( 1+1=3, 'Yes', 'No' ) → ,No‘</code></li> <li>• <code>if( 5 &gt; 3, 1, 0) → 1</code></li> <li>• <code>if( '', 'It is true (not empty)', 'It is false (empty)' ) → ,It is false (empty)‘</code></li> <li>• <code>if( ' ', 'It is true (not empty)', 'It is false (empty)' ) → ,It is true (not empty)‘</code></li> <li>• <code>if( 0, 'One', 'Zero' ) → ,Zero‘</code></li> <li>• <code>if( 10, 'One', 'Zero' ) → ,One‘</code></li> </ul>

**nullif**

Returns a NULL value if value1 equals value2; otherwise it returns value1. This can be used to conditionally substitute values with NULL.

Syntax	nullif(value1, value2)
Arguments	<ul style="list-style-type: none"> <li>• <b>value1</b> - The value that should either be used or substituted with NULL.</li> <li>• <b>value2</b> - The control value that will trigger the NULL substitution.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>nullif(' (none) ', ' (none) ') → NULL</code></li> <li>• <code>nullif('text', ' (none) ') → ,text‘</code></li> <li>• <code>nullif("name", ' ') → NULL</code>, if name is an empty string (or already NULL), the name in any other case.</li> </ul>

**regexp\_match**

Return the first matching position matching a regular expression within an unicode string, or 0 if the substring is not found.

Syntax	regexp_match(input_string, regex)
Arguments	<ul style="list-style-type: none"> <li>• <b>input_string</b> - the string to test against the regular expression</li> <li>• <b>regex</b> - The regular expression to test against. Backslash characters must be double escaped (e.g., „\\s“ to match a white space character or „\\b“ to match a word boundary).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>regexp_match('QGIS ROCKS', '\\sROCKS') → 5</code></li> <li>• <code>regexp_match('Budač', 'udač\\b') → 2</code></li> </ul>

## try

Tries an expression and returns its value if error-free. If the expression returns an error, an alternative value will be returned when provided otherwise the function will return NULL.

Syntax	try(expression, [alternative]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - the expression which should be run</li> <li>• <b>alternative</b> - the result which will be returned if the expression returns an error.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>try( to_int( '1' ), 0 ) → 1</code></li> <li>• <code>try( to_int( 'a' ), 0 ) → 0</code></li> <li>• <code>try( to_date( 'invalid_date' ) ) → NULL</code></li> </ul>

## 13.2.5 Funktionen zum Umwandlung

This group contains functions to convert one data type to another (e.g., string from/to integer, binary from/to string, string to date, ...).

### from\_base64

Decodes a string in the Base64 encoding into a binary value.

Syntax	from_base64(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to decode</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>from_base64( 'U UdJUw==' ) → ,QGIS'</code></li> </ul>

## hash

Creates a hash from a string with a given method. One byte (8 bits) is represented with two hex 'digits', so 'md4' (16 bytes) produces a  $16 * 2 = 32$  character long hex string and 'keccak\_512' (64 bytes) produces a  $64 * 2 = 128$  character long hex string.

Syntax	hash(string, method)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to hash</li> <li>• <b>method</b> - The hash method among 'md4', 'md5', 'sha1', 'sha224', 'sha384', 'sha512', 'sha3_224', 'sha3_256', 'sha3_384', 'sha3_512', 'keccak_224', 'keccak_256', 'keccak_384', 'keccak_512'</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>hash('QGIS', 'md4')</code> → 'c0fc71c241cdebb6e888cbac0e2b68eb'</li> <li>• <code>hash('QGIS', 'md5')</code> → '57470aaa9e22adaefac7f5f342f1c6da'</li> <li>• <code>hash('QGIS', 'sha1')</code> → 'f87cfb2b74cdd5867db913237024e7001e62b114'</li> <li>• <code>hash('QGIS', 'sha224')</code> → '4093a619ada631c770f44bc643ead18fb393b93d6a6af1861fcfece0'</li> <li>• <code>hash('QGIS', 'sha256')</code> → 'eb045cba7a797aaa06ac58830846e40c8e8c780bc0676d3393605fae50c05309'</li> <li>• <code>hash('QGIS', 'sha384')</code> → '91c1de038cc3d09fdd512e99f9dd9922efadc39ed21d3922e69a4305cc2'</li> <li>• <code>hash('QGIS', 'sha512')</code> → 'c2c092f2ab743bf8edbebd028a745f30fc720408465ed369421f0a4e20'</li> <li>• <code>hash('QGIS', 'sha3_224')</code> → '467f49a5039e7280d5d42fd433e80d203439e338eaabd701f0d6c17d'</li> <li>• <code>hash('QGIS', 'sha3_256')</code> → '540f7354b6b8a6e735f2845250f15f4f3ba4f666c55574d9e9354575'</li> <li>• <code>hash('QGIS', 'sha3_384')</code> → '96052da1e77679e9a65f60d7ead961b287977823144786386eb4364'</li> <li>• <code>hash('QGIS', 'sha3_512')</code> → '900d079dc69761da113980253aa8ac0414a8bd6d09879a916228f87'</li> <li>• <code>hash('QGIS', 'keccak_224')</code> → '5b0ce6acef8b0a121d4ac4f3eaa8503c799ad4e26a3392d1fb2014'</li> <li>• <code>hash('QGIS', 'keccak_256')</code> → '991c520aa6815392de24087f61b2ae0fd56abbfee4a8ca019c101'</li> <li>• <code>hash('QGIS', 'keccak_384')</code> → 'c57a3aed9d856fa04e5eeee9b62b6e027cca81ba574116d3cc1f0d'</li> </ul>

## md5

Creates a md5 hash from a string.

Syntax	md5(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to hash</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>md5('QGIS')</code> → '57470aaa9e22adaefac7f5f342f1c6da'</li> </ul>

## sha256

Creates a sha256 hash from a string.

Syntax	sha256(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to hash</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>sha256('QGIS')</code> → 'eb045cba7a797aaa06ac58830846e40c8e8c780bc0676d3393605fae50c05309'</li> </ul>

### to\_base64

Encodes a binary value into a string, using the Base64 encoding.

Syntax	to_base64(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - the binary value to encode</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• to_base64('QGIS') → ,U UdJ U w==‘</li> </ul>

### to\_date

Converts a string into a date object. An optional format string can be provided to parse the string; see [QDate::fromString](#) or the documentation of the `format_date` function for additional documentation on the format. By default the current QGIS user locale is used.

Syntax	to_date(string, [format], [language]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a date value</li> <li>• <b>format</b> - format used to convert the string into a date</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a date. By default the current QGIS user locale is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• to_date('2012-05-04') → 2012-05-04</li> <li>• to_date('June 29, 2019', 'MMMM d, yyyy') → 2019-06-29, if the current locale uses the name ‚June‘ for the sixth month, otherwise an error occurs</li> <li>• to_date('29 juin, 2019', 'd MMMM, yyyy', 'fr') → 2019-06-29</li> </ul>

### to\_datetime

Converts a string into a datetime object. An optional format string can be provided to parse the string; see [QDate::fromString](#), [QTime::fromString](#) or the documentation of the `format_date` function for additional documentation on the format. By default the current QGIS user locale is used.

Syntax	to_datetime(string, [format], [language]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a datetime value</li> <li>• <b>format</b> - format used to convert the string into a datetime</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a datetime. By default the current QGIS user locale is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• to_datetime('2012-05-04 12:50:00') → 2012-05-04T12:50:00</li> <li>• to_datetime('June 29, 2019 @ 12:34', 'MMMM d, yyyy @ HH:mm') → 2019-06-29T12:34, if the current locale uses the name ‚June‘ for the sixth month, otherwise an error occurs</li> <li>• to_datetime('29 juin, 2019 @ 12:34', 'd MMMM, yyyy @ HH:mm', 'fr') → 2019-06-29T12:34</li> </ul>



## to\_decimal

Converts a degree, minute, second coordinate to its decimal equivalent.

Syntax	<code>to_decimal(value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - A degree, minute, second string.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_decimal('6°21\'16.445') → 6.3545680555</code></li> </ul>

## to\_dm

Converts a coordinate to degree, minute.

Syntax	<code>to_dm(coordinate, axis, precision, [formatting=])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>coordinate</b> - A latitude or longitude value.</li> <li>• <b>axis</b> - The axis of the coordinate. Either <code>'x'</code> or <code>'y'</code>.</li> <li>• <b>precision</b> - Number of decimals.</li> <li>• <b>formatting</b> - Designates the formatting type. Acceptable values are NULL (default), <code>'aligned'</code> or <code>'suffix'</code>.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_dm(6.1545681, 'x', 3) → 6°9.274'</code></li> <li>• <code>to_dm(6.1545681, 'y', 4, 'aligned') → 6°09.2741'N</code></li> <li>• <code>to_dm(6.1545681, 'y', 4, 'suffix') → 6°9.2741'N</code></li> </ul>

## to\_dms

Converts a coordinate to degree, minute, second.

Syntax	<code>to_dms(coordinate, axis, precision, [formatting=])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>coordinate</b> - A latitude or longitude value.</li> <li>• <b>axis</b> - The axis of the coordinate. Either <code>'x'</code> or <code>'y'</code>.</li> <li>• <b>precision</b> - Number of decimals.</li> <li>• <b>formatting</b> - Designates the formatting type. Acceptable values are NULL (default), <code>'aligned'</code> or <code>'suffix'</code>.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_dms(6.1545681, 'x', 3) → 6°9'16.445''</code></li> <li>• <code>to_dms(6.1545681, 'y', 4, 'aligned') → 6°09'16.4452''N</code></li> <li>• <code>to_dms(6.1545681, 'y', 4, 'suffix') → 6°9'16.4452''N</code></li> </ul>

### to\_int

Converts a string to integer number. Nothing is returned if a value cannot be converted to integer (e.g. '123asd' is invalid).

Syntax	to_int(string)
Arguments	<ul style="list-style-type: none"> <li><b>string</b> - string to convert to integer number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>to_int('123') → 123</li> </ul>

### to\_interval

Converts a string to an interval type. Can be used to take days, hours, month, etc of a date.

Syntax	to_interval(string)
Arguments	<ul style="list-style-type: none"> <li><b>string</b> - a string representing an interval. Allowable formats include {n} days {n} hours {n} months.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>to_interval('1 day 2 hours') → interval: 1.08333 days</li> <li>to_interval('0.5 hours') → interval: 30 minutes</li> <li>to_datetime('2012-05-05 12:00:00') - to_interval('1 day 2 hours') → 2012-05-04T10:00:00</li> </ul>

### to\_real

Converts a string to a real number. Nothing is returned if a value cannot be converted to real (e.g. '123.56asd' is invalid). Numbers are rounded after saving changes if the precision is smaller than the result of the conversion.

Syntax	to_real(string)
Arguments	<ul style="list-style-type: none"> <li><b>string</b> - string to convert to real number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>to_real('123.45') → 123.45</li> </ul>

### to\_string

Converts a number to string.

Syntax	to_string(number)
Arguments	<ul style="list-style-type: none"> <li><b>number</b> - Integer or real value. The number to convert to string.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>to_string(123) → '123'</li> </ul>

## to\_time

Converts a string into a time object. An optional format string can be provided to parse the string; see [QTime::fromString](#) for additional documentation on the format.

Syntax	<code>to_time(string, [format], [language])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a time value</li> <li>• <b>format</b> - format used to convert the string into a time</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a time</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_time('12:30:01') → 12:30:01</code></li> <li>• <code>to_time('12:34', 'HH:mm') → 12:34:00</code></li> <li>• <code>to_time('12:34', 'HH:mm', 'fr') → 12:34:00</code></li> </ul>

## 13.2.6 Benutzerdefinierte Funktionen

Diese Gruppe enthält vom Nutzer erstellte Funktionen. Unter [Funktions Editor](#) findet man weitere Details.

## 13.2.7 Datum und Zeit Funktionen

This group contains functions for handling date and time data. This group shares several functions with the [Funktionen zur Umwandlung](#) (`to_date`, `to_time`, `to_datetime`, `to_interval`) and [Zeichenkettenfunktionen](#) (`format_date`) groups.

### Bemerkung: Storing date, datetime and intervals on fields

The ability to store *date*, *time* and *datetime* values directly on fields depends on the data source's provider (e.g., Shapefile accepts *date* format, but not *datetime* or *time* format). The following are some suggestions to overcome this limitation:

- *date*, *datetime* and *time* can be converted and stored in text type fields using the [format\\_date\(\)](#) function.
- *Intervals* can be stored in integer or decimal type fields after using one of the date extraction functions (e.g., [day\(\)](#) to get the interval expressed in days)

## age

Returns the difference between two dates or datetimes.

The difference is returned as an `Interval` and needs to be used with one of the following functions in order to extract useful information:

- `year`
- `month`
- `week`
- `day`
- `hour`
- `minute`
- `second`

Syntax	<code>age(datetime1, datetime2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>datetime1</b> - a string, date or datetime representing the later date</li> <li>• <b>datetime2</b> - a string, date or datetime representing the earlier date</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>day (age ('2012-05-12', '2012-05-02')) → 10</code></li> <li>• <code>hour (age ('2012-05-12', '2012-05-02')) → 240</code></li> </ul>

### **datetime\_from\_epoch**

Returns a datetime whose date and time are the number of milliseconds, msec, that have passed since 1970-01-01T00:00:00.000, Coordinated Universal Time (Qt.UTC), and converted to Qt.LocalTime.

Syntax	<code>datetime_from_epoch(int)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>int</b> - number (milliseconds)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>datetime_from_epoch(1483225200000) → 2017-01-01T00:00:00</code></li> </ul>

### **day**

Extracts the day from a date, or the number of days from an interval.

#### **Date variant**

Extracts the day from a date or datetime.

Syntax	<code>day(date)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>date</b> - a date or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>day ('2012-05-12') → 12</code></li> </ul>

#### **Interval variant**

Calculates the length in days of an interval.

Syntax	<code>day(interval)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of days from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>day(to_interval('3 days')) → 3</code></li> <li>• <code>day(to_interval('3 weeks 2 days')) → 23</code></li> <li>• <code>day(age('2012-01-01', '2010-01-01')) → 730</code></li> </ul>

### day\_of\_week

Returns the day of the week for a specified date or datetime. The returned value ranges from 0 to 6, where 0 corresponds to a Sunday and 6 to a Saturday.

Syntax	<code>day_of_week(date)</code>
Arguments	<ul style="list-style-type: none"><li>• <b>date</b> - date or datetime value</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>day_of_week(to_date('2015-09-21')) → 1</code></li></ul>

### epoch

Returns the interval in milliseconds between the unix epoch and a given date value.

Syntax	<code>epoch(date)</code>
Arguments	<ul style="list-style-type: none"><li>• <b>date</b> - a date or datetime value</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>epoch(to_date('2017-01-01')) → 1483203600000</code></li></ul>

### format\_date

Formats a date type or string into a custom string format. Uses Qt date/time format strings. See [QDateTime::toString](#).

Syntax	<code>format_date(datetime, format, [language])</code> [] marks optional arguments																																																
Arguments	<ul style="list-style-type: none"> <li>• <b>datetime</b> - date, time or datetime value</li> <li>• <b>format</b> - String template used to format the string.</li> </ul> <table border="1"> <thead> <tr> <th>Ausdruck</th><th>Output</th></tr> </thead> <tbody> <tr><td>d</td><td>the day as number without a leading zero (1 to 31)</td></tr> <tr><td>dd</td><td>the day as number with a leading zero (01 to 31)</td></tr> <tr><td>ddd</td><td>the abbreviated localized day name (e.g. ‚Mon‘ to ‚Sun‘)</td></tr> <tr><td>dddd</td><td>the long localized day name (e.g. ‚Monday‘ to ‚Sunday‘)</td></tr> <tr><td>M</td><td>the month as number without a leading zero (1-12)</td></tr> <tr><td>MM</td><td>the month as number with a leading zero (01-12)</td></tr> <tr><td>MMM</td><td>the abbreviated localized month name (e.g. ‚Jan‘ to ‚Dec‘)</td></tr> <tr><td>MMMM</td><td>the long localized month name (e.g. ‚January‘ to ‚December‘)</td></tr> <tr><td>yy</td><td>the year as two digit number (00-99)</td></tr> <tr><td>yyyy</td><td>the year as four digit number</td></tr> </tbody> </table> <p>These expressions may be used for the time part of the format string:</p> <table border="1"> <thead> <tr> <th>Ausdruck</th><th>Output</th></tr> </thead> <tbody> <tr><td>h</td><td>the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)</td></tr> <tr><td>hh</td><td>the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)</td></tr> <tr><td>H</td><td>the hour without a leading zero (0 to 23, even with AM/PM display)</td></tr> <tr><td>HH</td><td>the hour with a leading zero (00 to 23, even with AM/PM display)</td></tr> <tr><td>m</td><td>the minute without a leading zero (0 to 59)</td></tr> <tr><td>mm</td><td>the minute with a leading zero (00 to 59)</td></tr> <tr><td>s</td><td>the second without a leading zero (0 to 59)</td></tr> <tr><td>ss</td><td>the second with a leading zero (00 to 59)</td></tr> <tr><td>z</td><td>the milliseconds without trailing zeroes (0 to 999)</td></tr> <tr><td>zzz</td><td>the milliseconds with trailing zeroes (000 to 999)</td></tr> <tr><td>AP or A</td><td>interpret as an AM/PM time. <i>AP</i> must be either ‚AM‘ or ‚PM‘.</td></tr> <tr><td>ap or a</td><td>Interpret as an AM/PM time. <i>ap</i> must be either ‚am‘ or ‚pm‘.</td></tr> </tbody> </table> <ul style="list-style-type: none"> <li>• <b>language</b> - language (lowercase, two- or three-letter, <a href="#">ISO 639 language code</a>) used to format the date into a custom string. By default the current QGIS user locale is used.</li> </ul>	Ausdruck	Output	d	the day as number without a leading zero (1 to 31)	dd	the day as number with a leading zero (01 to 31)	ddd	the abbreviated localized day name (e.g. ‚Mon‘ to ‚Sun‘)	dddd	the long localized day name (e.g. ‚Monday‘ to ‚Sunday‘)	M	the month as number without a leading zero (1-12)	MM	the month as number with a leading zero (01-12)	MMM	the abbreviated localized month name (e.g. ‚Jan‘ to ‚Dec‘)	MMMM	the long localized month name (e.g. ‚January‘ to ‚December‘)	yy	the year as two digit number (00-99)	yyyy	the year as four digit number	Ausdruck	Output	h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)	hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)	H	the hour without a leading zero (0 to 23, even with AM/PM display)	HH	the hour with a leading zero (00 to 23, even with AM/PM display)	m	the minute without a leading zero (0 to 59)	mm	the minute with a leading zero (00 to 59)	s	the second without a leading zero (0 to 59)	ss	the second with a leading zero (00 to 59)	z	the milliseconds without trailing zeroes (0 to 999)	zzz	the milliseconds with trailing zeroes (000 to 999)	AP or A	interpret as an AM/PM time. <i>AP</i> must be either ‚AM‘ or ‚PM‘.	ap or a	Interpret as an AM/PM time. <i>ap</i> must be either ‚am‘ or ‚pm‘.
Ausdruck	Output																																																
d	the day as number without a leading zero (1 to 31)																																																
dd	the day as number with a leading zero (01 to 31)																																																
ddd	the abbreviated localized day name (e.g. ‚Mon‘ to ‚Sun‘)																																																
dddd	the long localized day name (e.g. ‚Monday‘ to ‚Sunday‘)																																																
M	the month as number without a leading zero (1-12)																																																
MM	the month as number with a leading zero (01-12)																																																
MMM	the abbreviated localized month name (e.g. ‚Jan‘ to ‚Dec‘)																																																
MMMM	the long localized month name (e.g. ‚January‘ to ‚December‘)																																																
yy	the year as two digit number (00-99)																																																
yyyy	the year as four digit number																																																
Ausdruck	Output																																																
h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)																																																
hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)																																																
H	the hour without a leading zero (0 to 23, even with AM/PM display)																																																
HH	the hour with a leading zero (00 to 23, even with AM/PM display)																																																
m	the minute without a leading zero (0 to 59)																																																
mm	the minute with a leading zero (00 to 59)																																																
s	the second without a leading zero (0 to 59)																																																
ss	the second with a leading zero (00 to 59)																																																
z	the milliseconds without trailing zeroes (0 to 999)																																																
zzz	the milliseconds with trailing zeroes (000 to 999)																																																
AP or A	interpret as an AM/PM time. <i>AP</i> must be either ‚AM‘ or ‚PM‘.																																																
ap or a	Interpret as an AM/PM time. <i>ap</i> must be either ‚am‘ or ‚pm‘.																																																
Beispiele	<ul style="list-style-type: none"> <li>• <code>format_date('2012-05-15', 'dd.MM.yyyy')</code> → ‚15.05.2012‘</li> <li>• <code>format_date('2012-05-15', 'd MMMM yyyy', 'fr')</code> → ‚15 mai 2012‘</li> <li>• <code>format_date('2012-05-15', 'dddd')</code> → ‚Tuesday‘, if the current locale is an English variant</li> <li>• <code>format_date('2012-05-15 13:54:20', 'dd.MM.yy')</code> → ‚15.05.12‘</li> <li>• <code>format_date('13:54:20', 'hh:mm AP')</code> → ‚01:54 PM‘</li> </ul>																																																

## hour

Extracts the hour part from a datetime or time, or the number of hours from an interval.

### Time variant

Extracts the hour part from a time or datetime.

Syntax	hour(datetime)
Arguments	<ul style="list-style-type: none"> <li>• <b>datetime</b> - a time or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>hour( to_datetime('2012-07-22 13:24:57') ) → 13</code></li> </ul>

### Interval variant

Calculates the length in hours of an interval.

Syntax	hour(interval)
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of hours from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>hour(to_interval('3 hours')) → 3</code></li> <li>• <code>hour(age('2012-07-22T13:00:00','2012-07-22T10:00:00')) → 3</code></li> <li>• <code>hour(age('2012-01-01','2010-01-01')) → 17520</code></li> </ul>

## make\_date

Creates a date value from year, month and day numbers.

Syntax	make_date(year, month, day)
Arguments	<ul style="list-style-type: none"> <li>• <b>year</b> - Year number. Years 1 to 99 are interpreted as is. Year 0 is invalid.</li> <li>• <b>month</b> - Month number, where 1=January</li> <li>• <b>day</b> - Day number, beginning with 1 for the first day in the month</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>make_date(2020, 5, 4) → date value 2020-05-04</code></li> </ul>

## make\_datetime

Creates a datetime value from year, month, day, hour, minute and second numbers.

Syntax	<code>make_datetime(year, month, day, hour, minute, second)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>year</b> - Year number. Years 1 to 99 are interpreted as is. Year 0 is invalid.</li> <li>• <b>month</b> - Month number, where 1=January</li> <li>• <b>day</b> - Day number, beginning with 1 for the first day in the month</li> <li>• <b>hour</b> - Hour number</li> <li>• <b>minute</b> - Minutes</li> <li>• <b>second</b> - Seconds (fractional values include milliseconds)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>make_datetime(2020, 5, 4, 13, 45, 30.5)</code> → datetime value 2020-05-04 13:45:30.500</li> </ul>

## make\_interval

Creates an interval value from year, month, weeks, days, hours, minute and seconds values.

Syntax	<code>make_interval([years=0], [months=0], [weeks=0], [days=0], [hours=0], [minutes=0], [seconds=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>years</b> - Number of years (assumes a 365.25 day year length).</li> <li>• <b>months</b> - Number of months (assumes a 30 day month length)</li> <li>• <b>weeks</b> - Number of weeks</li> <li>• <b>days</b> - Number of days</li> <li>• <b>hours</b> - Number of hours</li> <li>• <b>minutes</b> - Number of minutes</li> <li>• <b>seconds</b> - Number of seconds</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>make_interval(hours:=3)</code> → interval: 3 hours</li> <li>• <code>make_interval(days:=2, hours:=3)</code> → interval: 2.125 days</li> <li>• <code>make_interval(minutes:=0.5, seconds:=5)</code> → interval: 35 seconds</li> </ul>

## make\_time

Creates a time value from hour, minute and second numbers.

Syntax	<code>make_time(hour, minute, second)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>hour</b> - Hour number</li> <li>• <b>minute</b> - Minutes</li> <li>• <b>second</b> - Seconds (fractional values include milliseconds)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>make_time(13, 45, 30.5)</code> → time value 13:45:30.500</li> </ul>



## minute

Extracts the minutes part from a datetime or time, or the number of minutes from an interval.

### Time variant

Extracts the minutes part from a time or datetime.

Syntax	minute(datetime)
Arguments	<ul style="list-style-type: none"> <li>• <b>datetime</b> - a time or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>minute( to_datetime('2012-07-22 13:24:57') ) → 24</code></li> </ul>

### Interval variant

Calculates the length in minutes of an interval.

Syntax	minute(interval)
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of minutes from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>minute(to_interval('3 minutes')) → 3</code></li> <li>• <code>minute(age('2012-07-22T00:20:00', '2012-07-22T00:00:00')) → 20</code></li> <li>• <code>minute(age('2012-01-01', '2010-01-01')) → 1051200</code></li> </ul>

## month

Extracts the month part from a date, or the number of months from an interval.

### Date variant

Extracts the month part from a date or datetime.

Syntax	month(date)
Arguments	<ul style="list-style-type: none"> <li>• <b>date</b> - a date or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>month('2012-05-12') → 05</code></li> </ul>

### Interval variant

Calculates the length in months of an interval.

Syntax	month(interval)
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of months from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>month(to_interval('3 months')) → 3</code></li> <li>• <code>month(age('2012-01-01', '2010-01-01')) → 4.03333</code></li> </ul>

### now

Returns the current date and time. The function is static and will return consistent results while evaluating. The time returned is the time when the expression is prepared.

Syntax	now()
Beispiele	<ul style="list-style-type: none"> <li>• <code>now()</code> → 2012-07-22T13:24:57</li> </ul>

### second

Extracts the seconds part from a datetime or time, or the number of seconds from an interval.

#### Time variant

Extracts the seconds part from a time or datetime.

Syntax	second(datetime)
Arguments	<ul style="list-style-type: none"> <li>• <b>datetime</b> - a time or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>second( to_datetime('2012-07-22 13:24:57') ) → 57</code></li> </ul>

#### Interval variant

Calculates the length in seconds of an interval.

Syntax	second(interval)
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of seconds from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>second(to_interval('3 minutes')) → 180</code></li> <li>• <code>second(age('2012-07-22T00:20:00','2012-07-22T00:00:00')) → 1200</code></li> <li>• <code>second(age('2012-01-01','2010-01-01')) → 63072000</code></li> </ul>

### to\_date

Converts a string into a date object. An optional format string can be provided to parse the string; see [QDate::fromString](#) or the documentation of the `format_date` function for additional documentation on the format. By default the current QGIS user locale is used.

Syntax	<code>to_date(string, [format], [language])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a date value</li> <li>• <b>format</b> - format used to convert the string into a date</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a date. By default the current QGIS user locale is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_date('2012-05-04') → 2012-05-04</code></li> <li>• <code>to_date('June 29, 2019', 'MMMM d, yyyy') → 2019-06-29</code>, if the current locale uses the name ‚June‘ for the sixth month, otherwise an error occurs</li> <li>• <code>to_date('29 juin, 2019', 'd MMMM, yyyy', 'fr') → 2019-06-29</code></li> </ul>

### to\_datetime

Converts a string into a datetime object. An optional format string can be provided to parse the string; see [QDate::fromString](#), [QTime::fromString](#) or the documentation of the `format_date` function for additional documentation on the format. By default the current QGIS user locale is used.

Syntax	<code>to_datetime(string, [format], [language])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a datetime value</li> <li>• <b>format</b> - format used to convert the string into a datetime</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a datetime. By default the current QGIS user locale is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_datetime('2012-05-04 12:50:00') → 2012-05-04T12:50:00</code></li> <li>• <code>to_datetime('June 29, 2019 @ 12:34', 'MMMM d, yyyy @ HH:mm') → 2019-06-29T12:34</code>, if the current locale uses the name ‚June‘ for the sixth month, otherwise an error occurs</li> <li>• <code>to_datetime('29 juin, 2019 @ 12:34', 'd MMMM, yyyy @ HH:mm', 'fr') → 2019-06-29T12:34</code></li> </ul>

### to\_interval

Converts a string to an interval type. Can be used to take days, hours, month, etc of a date.

Syntax	<code>to_interval(string)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - a string representing an interval. Allowable formats include {n} days {n} hours {n} months.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_interval('1 day 2 hours') → interval: 1.08333 days</code></li> <li>• <code>to_interval('0.5 hours') → interval: 30 minutes</code></li> <li>• <code>to_datetime('2012-05-05 12:00:00') - to_datetime('2012-05-04 10:00:00') → 2012-05-04T10:00:00</code></li> </ul>

### to\_time

Converts a string into a time object. An optional format string can be provided to parse the string; see [QTime::fromString](#) for additional documentation on the format.

Syntax	<code>to_time(string, [format], [language])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string representing a time value</li> <li>• <b>format</b> - format used to convert the string into a time</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, ISO 639 language code) used to convert the string into a time</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_time('12:30:01') → 12:30:01</code></li> <li>• <code>to_time('12:34', 'HH:mm') → 12:34:00</code></li> <li>• <code>to_time('12:34', 'HH:mm', 'fr') → 12:34:00</code></li> </ul>

### week

Extracts the week number from a date, or the number of weeks from an interval.

#### Date variant

Extracts the week number from a date or datetime.

Syntax	<code>week(date)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>date</b> - a date or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>week('2012-05-12') → 19</code></li> </ul>

#### Interval variant

Calculates the length in weeks of an interval.

Syntax	<code>week(interval)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of months from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>week(to_interval('3 weeks')) → 3</code></li> <li>• <code>week(age('2012-01-01', '2010-01-01')) → 104.285</code></li> </ul>

## year

Extracts the year part from a date, or the number of years from an interval.

### Date variant

Extracts the year part from a date or datetime.

Syntax	<code>year(date)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>date</b> - a date or datetime value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>year('2012-05-12')</code> → 2012</li> </ul>

### Interval variant

Calculates the length in years of an interval.

Syntax	<code>year(interval)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>interval</b> - interval value to return number of years from</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>year(to_interval('3 years'))</code> → 3</li> <li>• <code>year(age('2012-01-01', '2010-01-01'))</code> → 1.9986</li> </ul>

### Einige Beispiele:

Neben diesen Funktionen führt das Subtrahieren von Datums- oder Zeitangaben mit dem – (minus) Operator zur Ausgabe eines Zeitintervalls.

Die Addition oder Subtraktion von Zeitintervallen zu Datums- oder Zeitangaben mit Hilfe des + (plus) oder – (minus) Operators gibt eine Datums- oder Zeitangabe zurück.

- Ausgabe der Anzahl der Tage bis zur Herausgabe von QGIS 3.0:

```
to_date('2017-09-29') - to_date(now())
-- Returns <interval: 203 days>
```

- Analog mit time:

```
to_datetime('2017-09-29 12:00:00') - now()
-- Returns <interval: 202.49 days>
```

- Ausgabe der Datums- und Zeitangabe für 100 Tage in der Zukunft:

```
now() + to_interval('100 days')
-- Returns <datetime: 2017-06-18 01:00:00>
```

### 13.2.8 Felder und Werte

Contains a list of fields from the active layer, and special values. Fields list includes the ones stored in the dataset, *virtual* and *auxiliary* ones as well as from *joins*.

Doppelklick auf einen Feldnamen fügt das Feld im Ausdruck ein. Man kann auch den Feldnamen (vorzugsweise in Anführungszeichen) oder seinen *alias* verwenden.

Um Feldwerte, die man in einem Ausdruck verwenden will anzuzeigen, klickt man im Vorschau-Widget auf *10 Stichproben* oder *Alle eindeutigen*. Die angeforderten Werte werden dann angezeigt und man kann das Ergebnis mit Hilfe des *Suche* Kastens über der Liste filtern. Beispielwerte können außerdem durch Klick mit der rechten Maustaste auf ein Feld angezeigt werden.

Um einen Wert in den erstellten Ausdruck zu übernehmen, klickt man doppelt auf den Wert in der Liste. Wenn der Wert eine Zeichenkette ist, sollte er mit einfachen Anführungszeichen ausgezeichnet sein. Ansonsten sind keine Anführungszeichen erforderlich.

#### NULL

Equates to a NULL value.

Syntax	NULL
Beispiele	<ul style="list-style-type: none"><li>• NULL → a NULL value</li></ul>

---

**Bemerkung:** To test for NULL use an *IS NULL* or *IS NOT NULL* expression.

---

### 13.2.9 Files and Paths Functions

This group contains functions which manipulate file and path names.

#### base\_file\_name

Returns the base name of the file without the directory or file suffix.

Syntax	base_file_name(path)
Arguments	<ul style="list-style-type: none"><li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>base_file_name('/home/qgis/data/country_boundaries.shp')</code> → <code>,country_boundaries'</code></li></ul>

**exif**

Retrieves exif tag values from an image file.

Syntax	<code>exif(path, [tag])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - An image file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> <li>• <b>tag</b> - The tag to return. If empty, a map with all exif tag values will be returned.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>exif('/my/photo.jpg', 'Exif.Image.Orientation') → 0</code></li> </ul>

**file\_exists**

Returns TRUE if a file path exists.

Syntax	<code>file_exists(path)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>file_exists('/home/qgis/data/country_boundaries.shp') → TRUE</code></li> </ul>

**file\_name**

Returns the name of a file (including the file extension), excluding the directory.

Syntax	<code>file_name(path)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>file_name('/home/qgis/data/country_boundaries.shp') → ,country_boundaries.shp'</code></li> </ul>

**file\_path**

Returns the directory component of a file path. This does not include the file name.

Syntax	<code>file_path(path)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>file_path('/home/qgis/data/country_boundaries.shp') → ,/home/qgis/data'</code></li> </ul>

### file\_size

Returns the size (in bytes) of a file.

Syntax	file_size(path)
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>file_size('/home/qgis/data/country_boundaries.geojson')</code> → 5674</li> </ul>

### file\_suffix

Returns the file suffix (extension) from a file path.

Syntax	file_suffix(path)
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>file_suffix('/home/qgis/data/country_boundaries.shp')</code> → ,shp‘</li> </ul>

### is\_directory

Returns TRUE if a path corresponds to a directory.

Syntax	is_directory(path)
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_directory('/home/qgis/data/country_boundaries.shp')</code> → FALSE</li> <li>• <code>is_directory('/home/qgis/data/')</code> → TRUE</li> </ul>

### is\_file

Returns TRUE if a path corresponds to a file.

Syntax	is_file(path)
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - a file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_file('/home/qgis/data/country_boundaries.shp')</code> → TRUE</li> <li>• <code>is_file('/home/qgis/data/')</code> → FALSE</li> </ul>



### 13.2.10 Form Functions

This group contains functions that operate exclusively under the attribute form context. For example, in *field's widgets* settings.

#### current\_parent\_value

Only usable in an embedded form context, this function returns the current, unsaved value of a field in the parent form currently being edited. This will differ from the parent feature's actual attribute values for features which are currently being edited or have not yet been added to a parent layer. When used in a value-relation widget filter expression, this function should be wrapped into a ,coalesce()' that can retrieve the actual parent feature from the layer when the form is not used in an embedded context.

Syntax	current_parent_value(field_name)
Arguments	<ul style="list-style-type: none"> <li>• <b>field_name</b> - a field name in the current parent form</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• current_parent_value( 'FIELD_NAME' ) → The current value of a field ,FIELD_NAME' in the parent form.</li> </ul>

#### current\_value

Returns the current, unsaved value of a field in the form or table row currently being edited. This will differ from the feature's actual attribute values for features which are currently being edited or have not yet been added to a layer.

Syntax	current_value(field_name)
Arguments	<ul style="list-style-type: none"> <li>• <b>field_name</b> - a field name in the current form or table row</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• current_value( 'FIELD_NAME' ) → The current value of field ,FIELD_NAME'.</li> </ul>

### 13.2.11 Unscharfer Vergleich Funktion

Diese Gruppe enthält unscharfe Vergleichsfunktionen zwischen Werten.

#### hamming\_distance

Returns the Hamming distance between two strings. This equates to the number of characters at corresponding positions within the input strings where the characters are different. The input strings must be the same length, and the comparison is case-sensitive.

Syntax	hamming_distance(string1, string2)
Arguments	<ul style="list-style-type: none"> <li>• <b>string1</b> - a string</li> <li>• <b>string2</b> - a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>hamming_distance('abc', 'xec') → 2</code></li> <li>• <code>hamming_distance('abc', 'ABc') → 2</code></li> <li>• <code>hamming_distance(upper('abc'), upper('ABC')) → 0</code></li> <li>• <code>hamming_distance('abc', 'abcd') → NULL</code></li> </ul>

## levenshtein

Returns the Levenshtein edit distance between two strings. This equates to the minimum number of character edits (insertions, deletions or substitutions) required to change one string to another.

The Levenshtein distance is a measure of the similarity between two strings. Smaller distances mean the strings are more similar, and larger distances indicate more different strings. The distance is case sensitive.

Syntax	levenshtein(string1, string2)
Arguments	<ul style="list-style-type: none"> <li>• <b>string1</b> - a string</li> <li>• <b>string2</b> - a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>levenshtein('kittens', 'mitten') → 2</code></li> <li>• <code>levenshtein('Kitten', 'kitten') → 1</code></li> <li>• <code>levenshtein(upper('Kitten'), upper('kitten')) → 0</code></li> </ul>

## longest\_common\_substring

Returns the longest common substring between two strings. This substring is the longest string that is a substring of the two input strings. For example, the longest common substring of „ABABC“ and „BABCA“ is „BABC“. The substring is case sensitive.

Syntax	longest_common_substring(string1, string2)
Arguments	<ul style="list-style-type: none"> <li>• <b>string1</b> - a string</li> <li>• <b>string2</b> - a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>longest_common_substring('ABABC', 'BABCA') → ‚BABC‘</code></li> <li>• <code>longest_common_substring('abcDeF', 'abcdef') → ‚abc‘</code></li> <li>• <code>longest_common_substring(upper('abcDeF'), upper('abcdex')) → ‚ABCDE‘</code></li> </ul>

**soundex**

Returns the Soundex representation of a string. Soundex is a phonetic matching algorithm, so strings with similar sounds should be represented by the same Soundex code.

Syntax	soundex(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>soundex('robert') → ,R163'</code></li> <li>• <code>soundex('rupert') → ,R163'</code></li> <li>• <code>soundex('rubin') → ,R150'</code></li> </ul>

**13.2.12 Allgemeine Funktionen**

This group contains general assorted functions.

**env**

Gets an environment variable and returns its content as a string. If the variable is not found, NULL will be returned. This is handy to inject system specific configuration like drive letters or path prefixes. Definition of environment variables depends on the operating system, please check with your system administrator or the operating system documentation how this can be set.

Syntax	env(name)
Arguments	<ul style="list-style-type: none"> <li>• <b>name</b> - The name of the environment variable which should be retrieved.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>env('LANG') → ,en_US.UTF-8'</code></li> <li>• <code>env('MY_OWN_PREFIX_VAR') → ,Z:'</code></li> <li>• <code>env('I_DO_NOT_EXIST') → NULL</code></li> </ul>

**eval**

Evaluates an expression which is passed in a string. Useful to expand dynamic parameters passed as context variables or fields.

Syntax	eval(expression)
Arguments	<ul style="list-style-type: none"> <li>• <b>expression</b> - an expression string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>eval('\nice\') → ,nice'</code></li> <li>• <code>eval(@expression_var) → [whatever the result of evaluating @expression_var might be...]</code></li> </ul>

### eval\_template

Evaluates a template which is passed in a string. Useful to expand dynamic parameters passed as context variables or fields.

Syntax	eval_template(template)
Arguments	<ul style="list-style-type: none"> <li><b>template</b> - a template string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>eval_template('QGIS [% upper(\'rocks\') %]') → QGIS ROCKS</li> </ul>

### is\_layer\_visible

Returns TRUE if a specified layer is visible.

Syntax	is_layer_visible(layer)
Arguments	<ul style="list-style-type: none"> <li><b>layer</b> - a string, representing either a layer name or layer ID</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>is_layer_visible('baseraster') → TRUE</li> </ul>

### mime\_type

Returns the mime type of the binary data.

Syntax	mime_type(bytes)
Arguments	<ul style="list-style-type: none"> <li><b>bytes</b> - the binary data</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>mime_type('&lt;html&gt;&lt;body&gt;&lt;/body&gt;&lt;/html&gt;') → text/html</li> <li>mime_type(from_base64('R0lGODlhAQABAAAAACH5BAEKAAEALAAAAABAAEAAAIAGw==')) → image/gif</li> </ul>

### var

Returns the value stored within a specified variable.

Syntax	var(name)
Arguments	<ul style="list-style-type: none"> <li><b>name</b> - a variable name</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>var('qgis_version') → ,2.12‘</li> </ul>

Further reading: List of default *variables*

### with\_variable

This function sets a variable for any expression code that will be provided as 3rd argument. This is only useful for complicated expressions, where the same calculated value needs to be used in different places.

Syntax	with_variable(name, value, expression)
Arguments	<ul style="list-style-type: none"> <li>• <b>name</b> - the name of the variable to set</li> <li>• <b>value</b> - the value to set</li> <li>• <b>expression</b> - the expression for which the variable will be available</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>with_variable('my_sum', 1 + 2 + 3, @my_sum * 2 + @my_sum * 5) → 42</code></li> </ul>

## 13.2.13 Geometriefunktionen

This group contains functions that operate on geometry objects (e.g. buffer, transform, \$area).

### affine\_transform

Returns the geometry after an affine transformation. Calculations are in the Spatial Reference System of this geometry. The operations are performed in a scale, rotation, translation order. If there is a Z or M offset but the coordinate is not present in the geometry, it will be added.

Syntax	affine_transform(geometry, delta_x, delta_y, rotation_z, scale_x, scale_y, [delta_z=0], [delta_m=0], [scale_z=1], [scale_m=1]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>delta_x</b> - x-axis translation</li> <li>• <b>delta_y</b> - y-axis translation</li> <li>• <b>rotation_z</b> - rotation around z-axis in degrees counter-clockwise</li> <li>• <b>scale_x</b> - x-axis scale factor</li> <li>• <b>scale_y</b> - y-axis scale factor</li> <li>• <b>delta_z</b> - z-axis translation</li> <li>• <b>delta_m</b> - m-axis translation</li> <li>• <b>scale_z</b> - z-axis scale factor</li> <li>• <b>scale_m</b> - m-axis scale factor</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(affine_transform(geom_from_wkt('LINESTRING(1 1, 2 2)'), 2, 2, 0, 1, 1)) → ,LineString(3 3,4 4)</code></li> <li>• <code>geom_to_wkt(affine_transform(geom_from_wkt('POLYGON((0 0, 0 3, 2 2, 0 0))'), 0, 0, -90, 1, 2)) → ,Polygon((0 0,6 0,4 -2,0 0))</code></li> <li>• <code>geom_to_wkt(affine_transform(geom_from_wkt('POINT(3 1)'), 0, 0, 0, 1, 1, 5, 0)) → ,PointZ(3 1 5)</code></li> </ul>

## angle\_at\_vertex

Returns the bisector angle (average angle) to the geometry for a specified vertex on a linestring geometry. Angles are in degrees clockwise from north.

Syntax	<code>angle_at_vertex(geometry, vertex)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring geometry</li> <li>• <b>vertex</b> - vertex index, starting from 0; if the value is negative, the selected vertex index will be its total count minus the absolute value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>angle_at_vertex(geometry:=geom_from_wkt('LineString(0 0, 10 0, 10 10)'), vertex:=1) → 45.0</code></li> </ul>

## apply\_dash\_pattern

Applies a dash pattern to a geometry, returning a MultiLineString geometry which is the input geometry stroked along each line/ring with the specified pattern.

Syntax	<code>apply_dash_pattern(geometry, pattern, [start_rule=no_rule], [end_rule=no_rule], [adjustment=both], [pattern_offset=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry (accepts (multi)linestrings or (multi)polygons).</li> <li>• <b>pattern</b> - dash pattern, as an array of numbers representing dash and gap lengths. Must contain an even number of elements.</li> <li>• <b>start_rule</b> - optional rule for constraining the start of the pattern. Valid values are ,no_rule', ,full_dash', ,half_dash', ,full_gap', ,half_gap'.</li> <li>• <b>end_rule</b> - optional rule for constraining the end of the pattern. Valid values are ,no_rule', ,full_dash', ,half_dash', ,full_gap', ,half_gap'.</li> <li>• <b>adjustment</b> - optional rule for specifying which part of patterns are adjusted to fit the desired pattern rules. Valid values are ,both', ,dash', ,gap'.</li> <li>• <b>pattern_offset</b> - Optional distance specifying a specific distance along the pattern to commence at.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(apply_dash_pattern(geom_from_wkt('LINESTRING(1 1, 10 1)'), array(3, 1))) → MultiLineString((1 1, 4 1),(5 1, 8 1),(9 1, 10 1, 10 1))</code></li> <li>• <code>geom_to_wkt(apply_dash_pattern(geom_from_wkt('LINESTRING(1 1, 10 1)'), array(3, 1), start_rule='half_dash')) → MultiLineString((1 1, 2.5 1),(3.5 1, 6.5 1),(7.5 1, 10 1, 10 1))</code></li> </ul>

## \$area

Returns the area of the current feature. The area calculated by this function respects both the current project's ellipsoid setting and area unit settings. For example, if an ellipsoid has been set for the project then the calculated area will be ellipsoidal, and if no ellipsoid is set then the calculated area will be planimetric.

Syntax	\$area
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$area</code> → 42</li> </ul>

## area

Returns the area of a geometry polygon object. Calculations are always planimetric in the Spatial Reference System (SRS) of this geometry, and the units of the returned area will match the units for the SRS. This differs from the calculations performed by the \$area function, which will perform ellipsoidal calculations based on the project's ellipsoid and area unit settings.

Syntax	area(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - polygon geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>area(geom_from_wkt('POLYGON((0 0, 4 0, 4 2, 0 2, 0 0))'))</code> → 8.0</li> </ul>

## azimuth

Returns the north-based azimuth as the angle in radians measured clockwise from the vertical on point\_a to point\_b.

Syntax	azimuth(point_a, point_b)
Arguments	<ul style="list-style-type: none"> <li>• <b>point_a</b> - point geometry</li> <li>• <b>point_b</b> - point geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>degrees( azimuth( make_point(25, 45), make_point(75, 100) ) )</code> → 42.273689</li> <li>• <code>degrees( azimuth( make_point(75, 100), make_point(25, 45) ) )</code> → 222.273689</li> </ul>

## bearing

Returns the north-based bearing as the angle in radians measured clockwise on the ellipsoid from the vertical on point\_a to point\_b.

Syntax	bearing(point_a, point_b, [source_crs], [ellipsoid]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>point_a</b> - point geometry</li> <li>• <b>point_b</b> - point geometry</li> <li>• <b>source_crs</b> - an optional string representing the source CRS of the points. By default the current layer's CRS is used.</li> <li>• <b>ellipsoid</b> - an optional string representing the acronym or the authority:ID (eg ,EPSG:7030') of the ellipsoid on which the bearing should be measured. By default the current project's ellipsoid setting is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>degrees( bearing( make_point(16198544, -4534850), make_point(18736872, -1877769), 'EPSG:3857', 'EPSG:7030') ) → 49.980071</code></li> <li>• <code>degrees( bearing( make_point(18736872, -1877769), make_point(16198544, -4534850), 'EPSG:3857', 'WGS84') ) → 219.282386</code></li> </ul>

## boundary

Returns the closure of the combinatorial boundary of the geometry (ie the topological boundary of the geometry). For instance, a polygon geometry will have a boundary consisting of the linestrings for each ring in the polygon. Some geometry types do not have a defined boundary, e.g., points or geometry collections, and will return NULL.

Syntax	boundary(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(boundary(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))')) → ,LineString(1 1,0 0,-1 1,1 1)'</code></li> <li>• <code>geom_to_wkt(boundary(geom_from_wkt('LineString(1 1,0 0,-1 1)')) → ,MultiPoint((1 1),(-1 1))'</code></li> </ul>

Further reading: *Boundary* algorithm

## bounds

Returns a geometry which represents the bounding box of an input geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	bounds(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>bounds(@geometry) → bounding box of the current feature's geometry</code></li> <li>• <code>geom_to_wkt(bounds(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))')) → ,Polygon((-1 0, 1 0, 1 1, -1 1, -1 0))'</code></li> </ul>



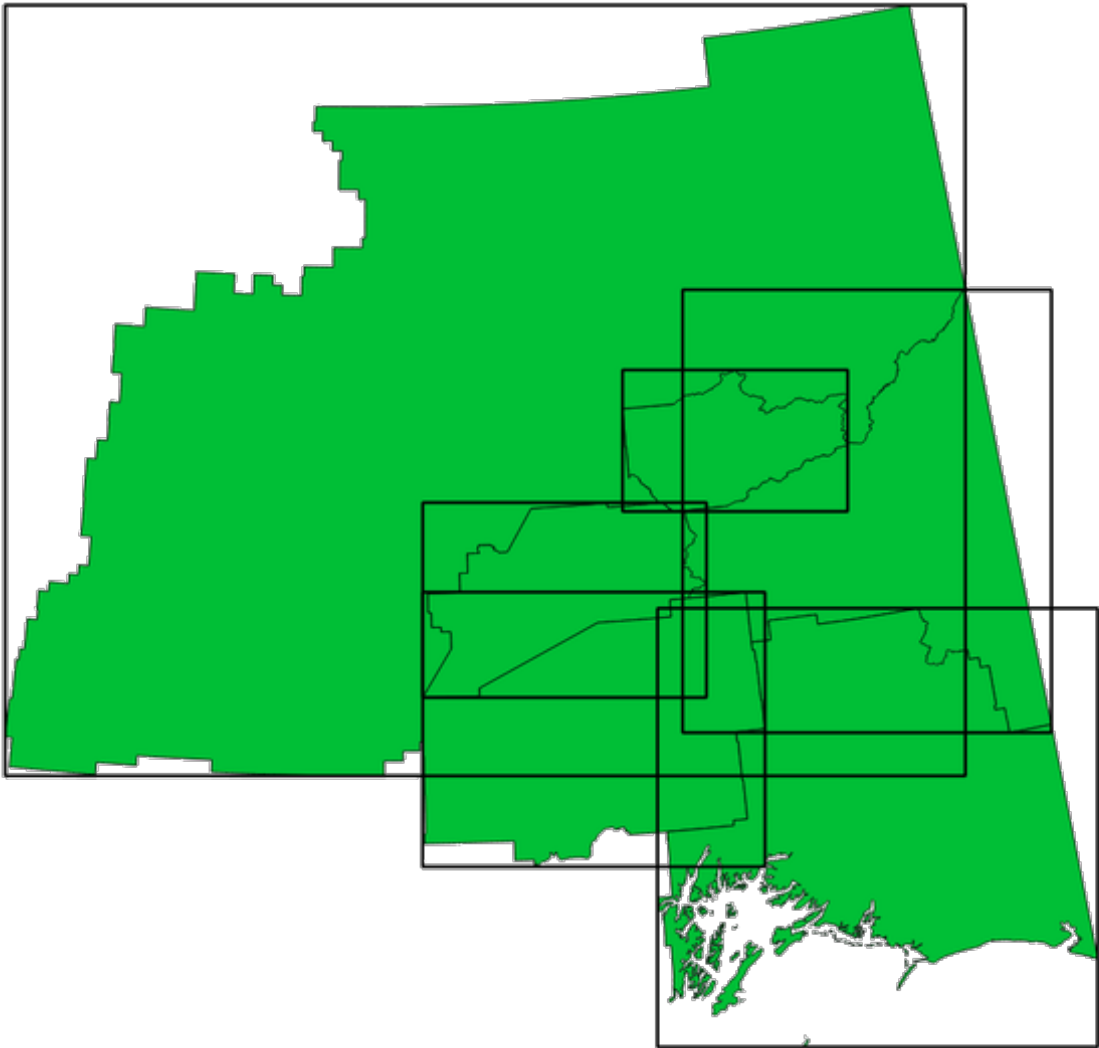


Abb. 13.4: Black lines represent the bounding boxes of each polygon feature

Further reading: *Bounding boxes* algorithm

**bounds\_height**

Returns the height of the bounding box of a geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	bounds_height(geometry)
Arguments	<ul style="list-style-type: none"><li>• <b>geometry</b> - a geometry</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>bounds_height(@geometry)</code> → height of bounding box of the current feature's geometry</li><li>• <code>bounds_height(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))'))</code> → 1</li></ul>

## bounds\_width

Returns the width of the bounding box of a geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	<code>bounds_width(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>bounds_width(@geometry)</code> → width of bounding box of the current feature's geometry</li> <li>• <code>bounds_width(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))'))</code> → 2</li> </ul>

## buffer

Returns a geometry that represents all points whose distance from this geometry is less than or equal to distance. Calculations are in the Spatial Reference System of this geometry.

Syntax	<code>buffer(geometry, distance, [segments=8], [cap='round'], [join='round'], [miter_limit=2])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>distance</b> - buffer distance in layer units</li> <li>• <b>segments</b> - number of segments to use to represent a quarter circle when a round join style is used. A larger number results in a smoother buffer with more nodes.</li> <li>• <b>cap</b> - end cap style for buffer. Valid values are <code>'round'</code>, <code>'flat'</code> or <code>'square'</code></li> <li>• <b>join</b> - join style for buffer. Valid values are <code>'round'</code>, <code>'bevel'</code> or <code>'miter'</code>.</li> <li>• <b>miter_limit</b> - miter distance limit, for use when the join style is set to <code>'miter'</code></li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>buffer(@geometry, 10.5)</code> → polygon of the current feature's geometry buffered by 10.5 units</li> </ul>

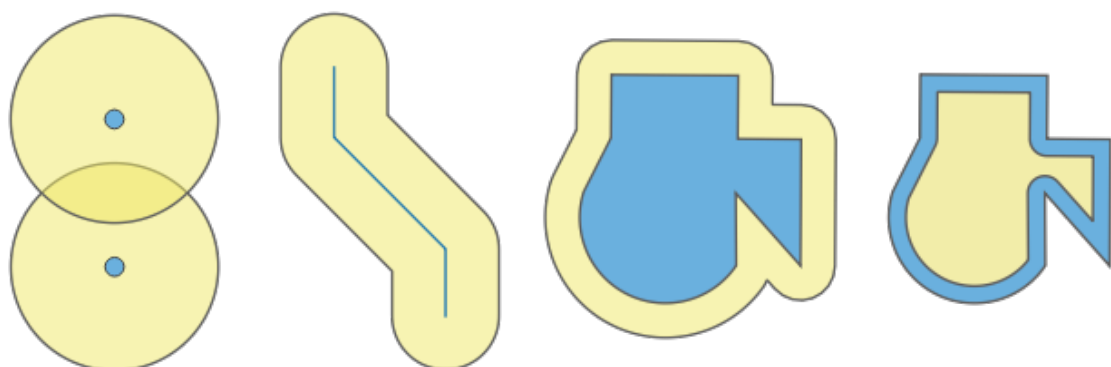


Abb. 13.5: Buffer (in yellow) of points, line, polygon with positive buffer, and polygon with negative buffer

Further reading: [Buffer algorithm](#)

**buffer\_by\_m**

Creates a buffer along a line geometry where the buffer diameter varies according to the m-values at the line vertices.

Syntax	<code>buffer_by_m(geometry, [segments=8])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - input geometry. Must be a (multi)line geometry with m values.</li> <li>• <b>segments</b> - number of segments to approximate quarter-circle curves in the buffer.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>buffer_by_m(geometry:=geom_from_wkt('LINESTRINGM(1 2 0.5, 4 2 0.2)'), segments:=8)</code> → A variable width buffer starting with a diameter of 0.5 and ending with a diameter of 0.2 along the linestring geometry.</li> </ul>

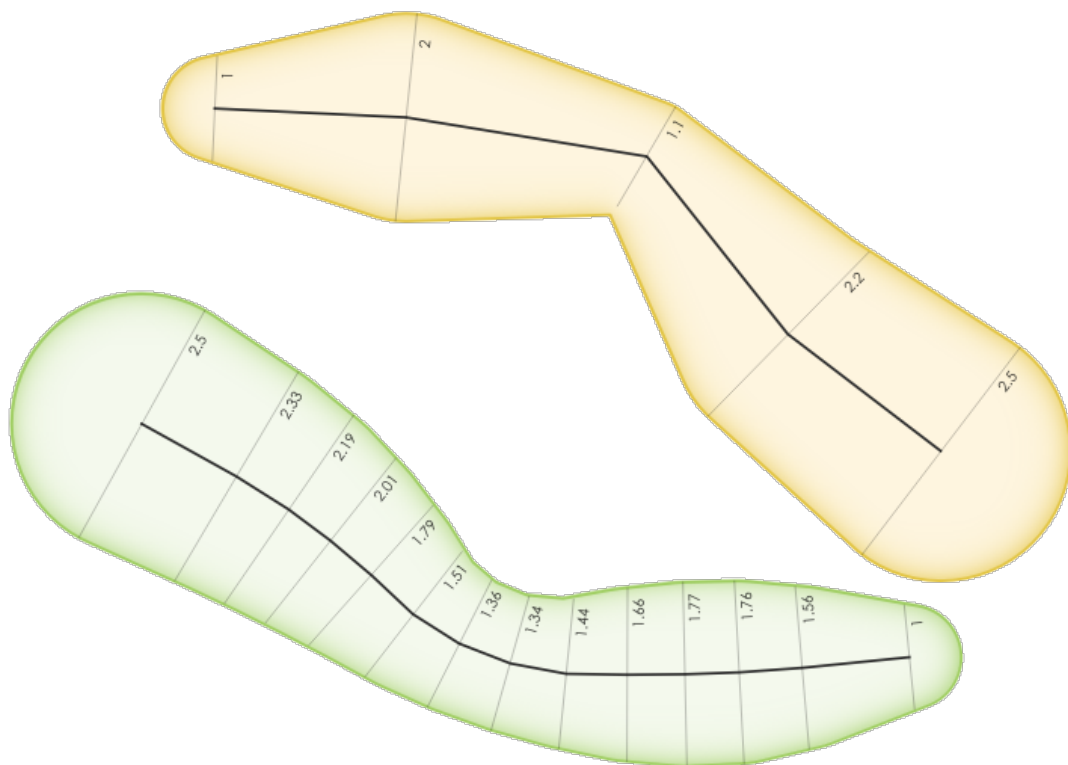


Abb. 13.6: Buffering line features using the m value on the vertices

Further reading: *Variable width buffer (by M value) algorithm*

## centroid

Returns the geometric center of a geometry.

Syntax	<code>centroid(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>centroid(@geometry) → a point geometry</code></li> </ul>

Further reading: [Centroids](#) algorithm

## close\_line

Returns a closed line string of the input line string by appending the first point to the end of the line, if it is not already closed. If the geometry is not a line string or multi line string then the result will be NULL.

Syntax	<code>close_line(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a line string geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(close_line(geom_from_wkt('LINESTRING(0 0, 1 0, 1 1)')))</code> → <code>,LineString(0 0, 1 0, 1 1, 0 0)'</code></li> <li>• <code>geom_to_wkt(close_line(geom_from_wkt('LINESTRING(0 0, 1 0, 1 1, 0 0)')))</code> → <code>,LineString(0 0, 1 0, 1 1, 0 0)'</code></li> </ul>

## closest\_point

Returns the point on geometry1 that is closest to geometry2.

Syntax	<code>closest_point(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - geometry to find closest point on</li> <li>• <b>geometry2</b> - geometry to find closest point to</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(closest_point(geom_from_wkt('LINESTRING (20 80, 98 190, 110 180, 50 75)'), geom_from_wkt('POINT(100 100)')))</code> → <code>,Point(73.0769 115.384)'</code></li> </ul>

## collect\_geometries

Collects a set of geometries into a multi-part geometry object.

### List of arguments variant

Geometry parts are specified as separate arguments to the function.

Syntax	<code>collect_geometries(geometry1, geometry2, ...)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(collect_geometries(make_point(1,2), make_point(3,4), make_point(5,6))) → ,MultiPoint ((1 2),(3 4),(5 6))</code></li> </ul>

### Array variant

Geometry parts are specified as an array of geometry parts.

Syntax	<code>collect_geometries(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - array of geometry objects</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(collect_geometries(array(make_point(1,2), make_point(3,4), make_point(5,6)))) → ,MultiPoint ((1 2),(3 4),(5 6))</code></li> </ul>

Further reading: *Collect geometries* algorithm

### combine

Returns the combination of two geometries.

Syntax	<code>combine(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( combine( geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 2 1)' ) ) ) → ,MULTILINESTRING((4 4, 2 1), (3 3, 4 4), (4 4, 5 5))</code></li> <li>• <code>geom_to_wkt( combine( geom_from_wkt( 'LINESTRING(3 3, 4 4)' ), geom_from_wkt( 'LINESTRING(3 3, 6 6, 2 1)' ) ) ) → ,LINESTRING(3 3, 4 4, 6 6, 2 1)</code></li> </ul>

### concave\_hull

Returns a possibly concave polygon that contains all the points in the geometry

Syntax	<code>concave_hull(geometry, target_percent, [allow_holes=False])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>target_percent</b> - the percentage of area of the convex hull the solution tries to approach. A target_percent of 1 gives the same result as the convex hull. A target_percent between 0 and 0.99 produces a result that should have a smaller area than the convex hull.</li> <li>• <b>allow_holes</b> - optional argument specifying whether to allow holes within the output geometry. Defaults to FALSE, set to TRUE to avoid including holes in the output geometry.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(concave_hull(geom_from_wkt('MULTILINESTRING((106 164,30 112,74 70,82 112,130 94,130 62,122 40,156 32,162 76,172 88),(132 178,134 148,128 136,96 128,132 108,150 130,170 142,174 110,156 96,158 90,158 88),(22 64,66 28,94 38,94 68,114 76,112 30,132 10,168 18,178 34,186 52,184 74,190 100,190 122,182 148,178 170,176 184,156 164,146 178,132 186,92 182,56 158,36 150,62 150,76 128,88 118))'), 0.99))</code> → <code>,Polygon((30 112, 36 150, 92 182, 132 186, 176 184, 190 122, 190 100, 186 52, 178 34, 168 18, 132 10, 112 30, 66 28, 22 64, 30 112))'</code></li> </ul>

Further reading: [convex\\_hull](#)

## contains

Tests whether a geometry contains another. Returns TRUE if and only if no points of geometry2 lie in the exterior of geometry1, and at least one point of the interior of geometry2 lies in the interior of geometry1.

Syntax	<code>contains(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>contains( geom_from_wkt( 'POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))' ), geom_from_wkt( 'POINT(0.5 0.5 )' ) )</code> → <b>TRUE</b></li> <li>• <code>contains( geom_from_wkt( 'POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) )</code> → <b>FALSE</b></li> </ul>

Further reading: [overlay\\_contains](#)

## convex\_hull

Returns the convex hull of a geometry. It represents the minimum convex geometry that encloses all geometries within the set.

Syntax	<code>convex_hull(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( convex_hull( geom_from_wkt( 'LINESTRING(3 3, 4 4, 4 10)' ) ) ) → ,POLYGON((3 3, 4 10, 4 4, 3 3))</code></li> </ul>

Further reading: [concave\\_hull](#), [Convex hull](#) algorithm

## crosses

Tests whether a geometry crosses another. Returns TRUE if the supplied geometries have some, but not all, interior points in common.

Syntax	<code>crosses(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>crosses( geom_from_wkt( 'LINESTRING(3 5, 4 4, 5 3)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) ) → TRUE</code></li> <li>• <code>crosses( geom_from_wkt( 'POINT(4 5)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) ) → FALSE</code></li> </ul>

Further reading: [overlay\\_crosses](#)

## densify\_by\_count

Takes a polygon or line layer geometry and generates a new one in which the geometries have a larger number of vertices than the original one.

Syntax	<code>densify_by_count(geometry, vertices)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry (accepts (multi)linestrings or (multi)polygons).</li> <li>• <b>vertices</b> - number of vertices to add (per segment)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( densify_by_count( geom_from_wkt( 'LINESTRING(1 1, 10 1)' ), 3 ) ) → LineString(1 1, 3.25 1, 5.5 1, 7.75 1, 10 1)</code></li> </ul>

Further reading: [Densify by count](#) algorithm

## densify\_by\_distance

Takes a polygon or line layer geometry and generates a new one in which the geometries are densified by adding additional vertices on edges that have a maximum distance of the specified interval distance.

Syntax	<code>densify_by_distance(geometry, distance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry (accepts (multi)linestrings or (multi)polygons).</li> <li>• <b>distance</b> - maximum interval distance between vertices in output geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(densify_by_distance(geom_from_wkt('LINESTRING(1 1, 10 1)'), 4)) → LineString(1 1, 4 1, 7 1, 10 1)</code></li> </ul>

Further reading: *Densify by interval* algorithm

## difference

Returns a geometry that represents that part of geometry1 that does not intersect with geometry2.

Syntax	<code>difference(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(difference(geom_from_wkt('LINESTRING(3 3, 4 4, 5 5)'), geom_from_wkt('LINESTRING(3 3, 4 4)')) → ,LINESTRING(4 4, 5 5)'</code></li> </ul>

Further reading: *Difference* algorithm

## disjoint

Tests whether geometries do not spatially intersect. Returns TRUE if the geometries do not share any space together.

Syntax	<code>disjoint(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>disjoint(geom_from_wkt('POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))'), geom_from_wkt('LINESTRING(3 3, 4 4, 5 5)')) → TRUE</code></li> <li>• <code>disjoint(geom_from_wkt('LINESTRING(3 3, 4 4, 5 5)'), geom_from_wkt('POINT(4 4)')) → FALSE</code></li> </ul>

Further reading: *overlay\_disjoint*



## distance

Returns the minimum distance (based on spatial reference) between two geometries in projected units.

Syntax	<code>distance(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>distance( geom_from_wkt( 'POINT(4 4)' ), geom_from_wkt( 'POINT(4 8)' ) ) → 4</code></li> </ul>

## distance\_to\_vertex

Returns the distance along the geometry to a specified vertex.

Syntax	<code>distance_to_vertex(geometry, vertex)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring geometry</li> <li>• <b>vertex</b> - vertex index, starting from 0; if the value is negative, the selected vertex index will be its total count minus the absolute value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>distance_to_vertex(geometry:=geom_from_wkt('LineString(0 0, 10 0, 10 10)'),vertex:=1) → 10.0</code></li> </ul>

## end\_point

Returns the last node from a geometry.

Syntax	<code>end_point(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(end_point(geom_from_wkt('LINESTRING(4 0, 4 2, 0 2)')) ) → ,Point (0 2)‘</code></li> </ul>

Further reading: *Extract specific vertices* algorithm

## exif\_geotag

Creates a point geometry from the exif geotags of an image file.

Syntax	<code>exif_geotag(path)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>path</b> - An image file path or a map layer value. If a map layer value is specified then the file source of the layer will be used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(exif_geotag('/my/photo.jpg')) → ,Point (2 4)‘</code></li> </ul>

## extend

Extends the start and end of a linestring geometry by a specified amount. Lines are extended using the bearing of the first and last segment in the line. For a multilinestring, all the parts are extended. Distances are in the Spatial Reference System of this geometry.

Syntax	<code>extend(geometry, start_distance, end_distance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a (multi)linestring geometry</li> <li>• <b>start_distance</b> - distance to extend the start of the line</li> <li>• <b>end_distance</b> - distance to extend the end of the line.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(extend(geom_from_wkt('LineString(0 0, 1 0, 1 1)'), 1, 2)) → ,LineString (-1 0, 1 0, 1 3)'</code></li> <li>• <code>geom_to_wkt(extend(geom_from_wkt('MultiLineString((0 0, 1 0, 1 1), (2 2, 0 2, 0 5))'), 1, 2)) → ,MultiLineString ((-1 0, 1 0, 1 3),(3 2, 0 2, 0 7))'</code></li> </ul>

Further reading: [Extend lines](#) algorithm

## exterior\_ring

Returns a line string representing the exterior ring of a polygon geometry. If the geometry is not a polygon then the result will be NULL.

Syntax	<code>exterior_ring(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a polygon geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(exterior_ring(geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1),( 0.1 0.1, 0.1 0.2, 0.2 0.2, 0.2, 0.1, 0.1 0.1))')) → ,LineString (-1 -1, 4 0, 4 2, 0 2, -1 -1)'</code></li> </ul>

## extrude

Returns an extruded version of the input (Multi-)Curve or (Multi-)Linestring geometry with an extension specified by x and y.

Syntax	<code>extrude(geometry, x, y)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a curve or linestring geometry</li> <li>• <b>x</b> - x extension, numeric value</li> <li>• <b>y</b> - y extension, numeric value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(extrude(geom_from_wkt('LineString(1 2, 3 2, 4 3)'), 1, 2)) → ,Polygon ((1 2, 3 2, 4 3, 5 5, 4 4, 2 4, 1 2))'</code></li> <li>• <code>geom_to_wkt(extrude(geom_from_wkt('MultiLineString((1 2, 3 2), (4 3, 8 3))'), 1, 2)) → ,MultiPolygon (((1 2, 3 2, 4 4, 2 4, 1 2)),((4 3, 8 3, 9 5, 5 5, 4 3)))'</code></li> </ul>

## flip\_coordinates

Returns a copy of the geometry with the x and y coordinates swapped. Useful for repairing geometries which have had their latitude and longitude values reversed.

Syntax	flip_coordinates(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(flip_coordinates(make_point(1, 2))) → ,Point (2 1)‘</code></li> <li><code>geom_to_wkt(flip_coordinates(geom_from_wkt('LineString(0 2, 1 0, 1 6)')) → ,LineString (2 0, 0 1, 6 1)‘</code></li> </ul>

Further reading: *Swap X and Y coordinates* algorithm

## force\_polygon\_ccw

Forces a geometry to respect the convention where exterior rings are counter-clockwise, interior rings are clockwise.

Syntax	force_polygon_ccw(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry. Any non-polygon geometries are returned unchanged.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(force_polygon_ccw(geometry:=geom_from_wkt('Polygon((-1 -1, 0 2, 4 2, 4 0, -1 -1))')) → ,Polygon((-1 -1, 4 0, 4 2, 0 2, -1 -1))‘</code></li> </ul>

Further reading: *force\_polygon\_cw*, *force\_rhr*

## force\_polygon\_cw

Forces a geometry to respect the convention where exterior rings are clockwise, interior rings are counter-clockwise.

Syntax	force_polygon_cw(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry. Any non-polygon geometries are returned unchanged.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(force_polygon_cw(geometry:=geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1))')) → ,Polygon((-1 -1, 0 2, 4 2, 4 0, -1 -1))‘</code></li> </ul>

Further reading: *force\_polygon\_ccw*, *force\_rhr*

## force\_rhr

Forces a geometry to respect the Right-Hand-Rule, in which the area that is bounded by a polygon is to the right of the boundary. In particular, the exterior ring is oriented in a clockwise direction and the interior rings in a counter-clockwise direction. Due to the inconsistency in the definition of the Right-Hand-Rule in some contexts it is recommended to use the explicit `force_polygon_cw` function instead.

Syntax	<code>force_rhr(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry. Any non-polygon geometries are returned unchanged.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(force_rhr(geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1)))') → ,Polygon((-1 -1, 0 2, 4 2, 4 0, -1 -1))'</code></li> </ul>

Further reading: *Force right-hand-rule* algorithm, *force\_polygon\_ccw*, *force\_polygon\_cw*

## geom\_from\_gml

Returns a geometry from a GML representation of geometry.

Syntax	<code>geom_from_gml(gml)</code>
Arguments	<ul style="list-style-type: none"> <li><b>gml</b> - GML representation of a geometry as a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_from_gml('&lt;gml:LineString srsName="EPSG:4326"&gt;&lt;gml:coordinates&gt;4, 4 5, 5 6, 6&lt;/gml:coordinates&gt;&lt;/gml:LineString&gt;') → a line geometry object</code></li> </ul>

## geom\_from\_wkb

Returns a geometry created from a Well-Known Binary (WKB) representation.

Syntax	<code>geom_from_wkb(binary)</code>
Arguments	<ul style="list-style-type: none"> <li><b>binary</b> - Well-Known Binary (WKB) representation of a geometry (as a binary blob)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_from_wkb( geom_to_wkb( make_point(4,5) ) ) → a point geometry object</code></li> </ul>

## geom\_from\_wkt

Returns a geometry created from a Well-Known Text (WKT) representation.

Syntax	<code>geom_from_wkt(text)</code>
Arguments	<ul style="list-style-type: none"> <li><b>text</b> - Well-Known Text (WKT) representation of a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_from_wkt('POINT(4 5)') → a geometry object</code></li> </ul>

### geom\_to\_wkb

Returns the Well-Known Binary (WKB) representation of a geometry

Syntax	geom_to_wkb(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• geom_to_wkb( @geometry ) → binary blob containing a geometry object</li> </ul>

### geom\_to\_wkt

Returns the Well-Known Text (WKT) representation of the geometry without SRID metadata.

Syntax	geom_to_wkt(geometry, [precision=8]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>precision</b> - numeric precision</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• geom_to_wkt( make_point(6, 50) ) → ,POINT(6 50)‘</li> <li>• geom_to_wkt(centroid(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))')) → ,POINT(0 0.66666667)‘</li> <li>• geom_to_wkt(centroid(geom_from_wkt('Polygon((1 1, 0 0, -1 1, 1 1))')), 2) → ,POINT(0 0.67)‘</li> </ul>

### \$geometry

Returns the geometry of the current feature. Can be used for processing with other functions. **WARNING: This function is deprecated. It is recommended to use the replacement @geometry variable instead.**

Syntax	\$geometry
Beispiele	<ul style="list-style-type: none"> <li>• geom_to_wkt( \$geometry ) → ,POINT(6 50)‘</li> </ul>

### geometry

Returns a feature's geometry.

Syntax	<code>geometry(feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - a feature object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geometry( @feature )</code> → the geometry of the current feature. Prefer using <code>@geometry</code>.</li> <li>• <code>geom_to_wkt( geometry( get_feature_by_id( 'streets', 1 ) ) )</code> → the geometry in WKT of the feature with the id 1 on the layer „streets“, e.g. <code>,POINT(6 50)‘</code></li> <li>• <code>intersects( @geometry, geometry( get_feature( 'streets', 'name', 'Main St.' ) ) )</code> → <code>TRUE</code> if the current feature spatially intersects the <code>,Main St.‘</code> named feature in the „streets“ layer</li> </ul>

### geometry\_n

Returns a specific geometry from a geometry collection, or NULL if the input geometry is not a collection. Also returns a part from a multipart geometry.

Syntax	<code>geometry_n(geometry, index)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry collection</li> <li>• <b>index</b> - index of geometry to return, where 1 is the first geometry in the collection</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( geometry_n( geom_from_wkt( 'GEOMETRYCOLLECTION (POINT (0 1), POINT(0 0), POINT(1 0), POINT(1 1)) ' ), 3 ) )</code> → <code>,Point (1 0)‘</code></li> </ul>

### geometry\_type

Returns a string value describing the type of a geometry (Point, Line or Polygon)

Syntax	<code>geometry_type(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geometry_type( geom_from_wkt( 'LINESTRING(2 5, 3 6, 4 8)' ) )</code> → <code>,Line‘</code></li> <li>• <code>geometry_type( geom_from_wkt( 'MULTILINESTRING((2 5, 3 6, 4 8), (1 1, 0 0))' ) )</code> → <code>,Line‘</code></li> <li>• <code>geometry_type( geom_from_wkt( 'POINT(2 5)' ) )</code> → <code>,Point‘</code></li> <li>• <code>geometry_type( geom_from_wkt( 'POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1))' ) )</code> → <code>,Polygon‘</code></li> </ul>

## hausdorff\_distance

Returns the Hausdorff distance between two geometries. This is basically a measure of how similar or dissimilar 2 geometries are, with a lower distance indicating more similar geometries.

The function can be executed with an optional densify fraction argument. If not specified, an approximation to the standard Hausdorff distance is used. This approximation is exact or close enough for a large subset of useful cases. Examples of these are:

- computing distance between Linestrings that are roughly parallel to each other, and roughly equal in length. This occurs in matching linear networks.
- Testing similarity of geometries.

If the default approximate provided by this method is insufficient, specify the optional densify fraction argument. Specifying this argument performs a segment densification before computing the discrete Hausdorff distance. The parameter sets the fraction by which to densify each segment. Each segment will be split into a number of equal-length subsegments, whose fraction of the total length is closest to the given fraction. Decreasing the densify fraction parameter will make the distance returned approach the true Hausdorff distance for the geometries.

Syntax	hausdorff_distance(geometry1, geometry2, [densify_fraction]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> <li>• <b>densify_fraction</b> - densify fraction amount</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• hausdorff_distance( geometry1:= geom_from_wkt('LINESTRING (0 0, 2 1)'), geometry2:=geom_from_wkt('LINESTRING (0 0, 2 0)')) → 2</li> <li>• hausdorff_distance( geom_from_wkt('LINESTRING (130 0, 0 0, 0 150)'), geom_from_wkt('LINESTRING (10 10, 10 150, 130 10)')) → 14.142135623</li> <li>• hausdorff_distance( geom_from_wkt('LINESTRING (130 0, 0 0, 0 150)'), geom_from_wkt('LINESTRING (10 10, 10 150, 130 10)'), 0.5) → 70.0</li> </ul>

## inclination

Liefert die Neigung gemessen vom Zenit (0) zum Fußpunkt (180) von point\_a zu point\_b.

Syntax	inclination(point_a, point_b)
Arguments	<ul style="list-style-type: none"> <li>• <b>point_a</b> - point geometry</li> <li>• <b>point_b</b> - point geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• inclination( make_point( 5, 10, 0 ), make_point( 5, 10, 5 ) ) → 0.0</li> <li>• inclination( make_point( 5, 10, 0 ), make_point( 5, 10, 0 ) ) → 90.0</li> <li>• inclination( make_point( 5, 10, 0 ), make_point( 50, 100, 0 ) ) → 90.0</li> <li>• inclination( make_point( 5, 10, 0 ), make_point( 5, 10, -5 ) ) → 180.0</li> </ul>

## interior\_ring\_n

Returns a specific interior ring from a polygon geometry, or NULL if the geometry is not a polygon.

Syntax	<code>interior_ring_n(geometry, index)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - polygon geometry</li> <li>• <b>index</b> - index of interior to return, where 1 is the first interior ring</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( interior_ring_n( geom_from_wkt( 'POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1), (-0.1 -0.1, 0.4 0, 0.4 0.2, 0 0.2, -0.1 -0.1), (-1 -1, 4 0, 4 2, 0 2, -1 -1))' ), 1)) → ,LineString(-0.1 -0.1, 0.4 0, 0.4 0.2, 0 0.2, -0.1 -0.1)'</code></li> </ul>

## intersection

Returns a geometry that represents the shared portion of two geometries.

Syntax	<code>intersection(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( intersection( geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4)' ) ) ) → ,LINESTRING(3 3, 4 4)'</code></li> <li>• <code>geom_to_wkt( intersection( geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ), geom_from_wkt( 'MULTIPOINT(3.5 3.5, 4 5)' ) ) ) → ,POINT(3.5 3.5)'</code></li> </ul>

Further reading: [Intersection](#) algorithm

## intersects

Tests whether a geometry intersects another. Returns TRUE if the geometries spatially intersect (share any portion of space) and false if they do not.

Syntax	<code>intersects(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>intersects( geom_from_wkt( 'POINT(4 4)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) ) → TRUE</code></li> <li>• <code>intersects( geom_from_wkt( 'POINT(4 5)' ), geom_from_wkt( 'POINT(5 5)' ) ) → FALSE</code></li> </ul>

Further reading: [overlay\\_intersects](#)



### intersects\_bbox

Tests whether a geometry's bounding box overlaps another geometry's bounding box. Returns TRUE if the geometries spatially intersect the bounding box defined and false if they do not.

Syntax	<code>intersects_bbox(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>intersects_bbox( geom_from_wkt( 'POINT(4 5)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) ) → TRUE</code></li> <li>• <code>intersects_bbox( geom_from_wkt( 'POINT(6 5)' ), geom_from_wkt( 'POLYGON((3 3, 4 4, 5 5, 3 3))' ) ) → FALSE</code></li> </ul>

### is\_closed

Returns TRUE if a line string is closed (start and end points are coincident), or false if a line string is not closed. If the geometry is not a line string then the result will be NULL.

Syntax	<code>is_closed(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a line string geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_closed(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')) → FALSE</code></li> <li>• <code>is_closed(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2, 0 0)')) → TRUE</code></li> </ul>

### is\_empty

Returns TRUE if a geometry is empty (without coordinates), false if the geometry is not empty and NULL if there is no geometry. See also `is_empty_or_null`.

Syntax	<code>is_empty(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_empty(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')) → FALSE</code></li> <li>• <code>is_empty(geom_from_wkt('LINESTRING EMPTY')) → TRUE</code></li> <li>• <code>is_empty(geom_from_wkt('POINT(7 4)')) → FALSE</code></li> <li>• <code>is_empty(geom_from_wkt('POINT EMPTY')) → TRUE</code></li> </ul>

Further reading: [\*is\\_empty\\_or\\_null\*](#)

## is\_empty\_or\_null

Returns TRUE if a geometry is NULL or empty (without coordinates) or false otherwise. This function is like the expression `,@geometry IS NULL` or `is_empty(@geometry)`

Syntax	is_empty_or_null(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>is_empty_or_null(NULL) → TRUE</code></li> <li><code>is_empty_or_null(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')) → FALSE</code></li> <li><code>is_empty_or_null(geom_from_wkt('LINESTRING EMPTY')) → TRUE</code></li> <li><code>is_empty_or_null(geom_from_wkt('POINT(7 4)')) → FALSE</code></li> <li><code>is_empty_or_null(geom_from_wkt('POINT EMPTY')) → TRUE</code></li> </ul>

Further reading: [is\\_empty](#), [NULL](#)

## is\_multipart

Returns TRUE if the geometry is of Multi type.

Syntax	is_multipart(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>is_multipart(geom_from_wkt('MULTIPOINT ((0 0), (1 1), (2 2))')) → TRUE</code></li> <li><code>is_multipart(geom_from_wkt('POINT (0 0)')) → FALSE</code></li> </ul>

## is\_valid

Returns TRUE if a geometry is valid; if it is well-formed in 2D according to the OGC rules.

Syntax	is_valid(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>is_valid(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2, 0 0)')) → TRUE</code></li> <li><code>is_valid(geom_from_wkt('LINESTRING(0 0)')) → FALSE</code></li> </ul>

Further reading: [make\\_valid](#), [Check validity](#) algorithm

## \$length

Returns the length of a linestring. If you need the length of a border of a polygon, use \$perimeter instead. The length calculated by this function respects both the current project's ellipsoid setting and distance unit settings. For example, if an ellipsoid has been set for the project then the calculated length will be ellipsoidal, and if no ellipsoid is set then the calculated length will be planimetric.

Syntax	\$length
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$length</code> → 42.4711</li> </ul>

## length

Returns the number of characters in a string or the length of a geometry linestring.

### String variant

Returns the number of characters in a string.

Syntax	length(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to count length of</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>length('hello')</code> → 5</li> </ul>

### Geometry variant

Calculate the length of a geometry line object. Calculations are always planimetric in the Spatial Reference System (SRS) of this geometry, and the units of the returned length will match the units for the SRS. This differs from the calculations performed by the \$length function, which will perform ellipsoidal calculations based on the project's ellipsoid and distance unit settings.

Syntax	length(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - line geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>length(geom_from_wkt('LINESTRING(0 0, 4 0)'))</code> → 4.0</li> </ul>

Further reading: [\*straight\\_distance\\_2d\*](#)

## length3D

Calculates the 3D length of a geometry line object. If the geometry is not a 3D line object, it returns its 2D length. Calculations are always planimetric in the Spatial Reference System (SRS) of this geometry, and the units of the returned length will match the units for the SRS. This differs from the calculations performed by the \$length function, which will perform ellipsoidal calculations based on the project's ellipsoid and distance unit settings.

Syntax	<code>length3D(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - line geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>length3D(geom_from_wkt('LINESTRINGZ(0 0 0, 3 0 4)')) → 5.0</code></li> </ul>

### line\_interpolate\_angle

Liefert den Winkel parallel zu einer Liniengeometrie für eine vorgebenen Strecke entlang der Linie. Die Winkelangabe erfolgt in Grad im Uhrzeigersinn von Nord ausgehend.

Syntax	<code>line_interpolate_angle(geometry, distance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring geometry</li> <li>• <b>distance</b> - distance along line to interpolate angle at</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>line_interpolate_angle(geometry:=geom_from_wkt('LineString(0 0, 10 0)'), distance:=5) → 90.0</code></li> </ul>

### line\_interpolate\_point

Returns the point interpolated by a specified distance along a linestring geometry.

Syntax	<code>line_interpolate_point(geometry, distance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring geometry</li> <li>• <b>distance</b> - distance along line to interpolate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(line_interpolate_point(geometry:=geom_from_wkt('LineString(0, 8 0)'), distance:=5)) → ,Point(5 0)'</code></li> <li>• <code>geom_to_wkt(line_interpolate_point(geometry:=geom_from_wkt('LineString(0, 1 1, 2 0)'), distance:=2.1)) → ,Point(1.48492424 0.51507576)'</code></li> <li>• <code>geom_to_wkt(line_interpolate_point(geometry:=geom_from_wkt('LineString(0, 1 0)'), distance:=2)) → NULL</code></li> </ul>

Further reading: *Interpolate point on line* algorithm

### line\_locate\_point

Liefert den Abstand des nächsten Punktes auf einer Linie zu einem gegebenen Punkt.

Syntax	<code>line_locate_point(geometry, point)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring geometry</li> <li>• <b>point</b> - point geometry to locate closest position on linestring to</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>line_locate_point(geometry:=geom_from_wkt('LineString(0 0, 10 0)'),point:=geom_from_wkt('Point(5 0)')) → 5.0</code></li> </ul>

### line\_merge

Returns a LineString or MultiLineString geometry, where any connected LineStrings from the input geometry have been merged into a single linestring. This function will return NULL if passed a geometry which is not a LineString/MultiLineString.

Syntax	<code>line_merge(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a LineString/MultiLineString geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(line_merge(geom_from_wkt('MULTILINESTRING((0 0, 1 1),(1 1, 2 2))')) → ,LineString(0 0,1 1,2 2)'</code></li> <li>• <code>geom_to_wkt(line_merge(geom_from_wkt('MULTILINESTRING((0 0, 1 1),(11 1, 21 2))')) → ,MultiLineString((0 0, 1 1),(11 1, 21 2)'</code></li> </ul>

### line\_substring

Returns the portion of a line (or curve) geometry which falls between the specified start and end distances (measured from the beginning of the line). Z and M values are linearly interpolated from existing values.

Syntax	<code>line_substring(geometry, start_distance, end_distance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a linestring or curve geometry</li> <li>• <b>start_distance</b> - distance to start of substring</li> <li>• <b>end_distance</b> - distance to end of substring</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(line_substring(geometry:=geom_from_wkt('LineString(0 0, 10 0)'),start_distance:=2,end_distance:=6)) → ,LineString(2 0,6 0)'</code></li> </ul>

Further reading: [Line substring](#) algorithm

## m

Returns the m (measure) value of a point geometry.

Syntax	m(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a point geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>m( geom_from_wkt( 'POINTM(2 5 4)' ) ) → 4</li> </ul>

## m\_at

Retrieves a m coordinate of the geometry, or NULL if the geometry has no m value.

Syntax	m_at(geometry, vertex)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - geometry object</li> <li><b>vertex</b> - index of the vertex of the geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>m_at(geom_from_wkt('LineStringZM(0 0 0 0, 10 10 0 5, 10 10 0 0)'), 1) → 5</li> </ul>

## m\_max

Returns the maximum m (measure) value of a geometry.

Syntax	m_max(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry containing m values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>m_max( make_point_m( 0,0,1 ) ) → 1</li> <li>m_max(make_line( make_point_m( 0,0,1 ), make_point_m( -1, -1,2 ), make_point_m( -2,-2,0 ) ) ) → 2</li> </ul>

## m\_min

Returns the minimum m (measure) value of a geometry.

Syntax	m_min(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry containing m values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>m_min( make_point_m( 0,0,1 ) ) → 1</li> <li>m_min(make_line( make_point_m( 0,0,1 ), make_point_m( -1, -1,2 ), make_point_m( -2,-2,0 ) ) ) → 0</li> </ul>

## main\_angle

Returns the angle of the long axis (clockwise, in degrees from North) of the oriented minimal bounding rectangle, which completely covers the geometry.

Syntax	<code>main_angle(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>main_angle(geom_from_wkt('Polygon ((321577 129614, 321581 129618, 321585 129615, 321581 129610, 321577 129614))'))</code> → 38.66</li> </ul>

## make\_circle

Creates a circular polygon.

Syntax	<code>make_circle(center, radius, [segments=36])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>center</b> - center point of the circle</li> <li>• <b>radius</b> - radius of the circle</li> <li>• <b>segments</b> - optional argument for polygon segmentation. By default this value is 36</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_circle(make_point(10,10), 5, 4))</code> → <code>,Polygon ((10 15, 15 10, 10 5, 5 10, 10 15))'</code></li> <li>• <code>geom_to_wkt(make_circle(make_point(10,10,5), 5, 4))</code> → <code>,PolygonZ ((10 15 5, 15 10 5, 10 5 5, 5 10 5, 10 15 5))'</code></li> <li>• <code>geom_to_wkt(make_circle(make_point(10,10,5,30), 5, 4))</code> → <code>,PolygonZM ((10 15 5 30, 15 10 5 30, 10 5 5 30, 5 10 5 30, 10 15 5 30))'</code></li> </ul>

## make\_ellipse

Creates an elliptical polygon.

Syntax	<code>make_ellipse(center, semi_major_axis, semi_minor_axis, azimuth, [segments=36])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>center</b> - center point of the ellipse</li> <li>• <b>semi_major_axis</b> - semi-major axis of the ellipse</li> <li>• <b>semi_minor_axis</b> - semi-minor axis of the ellipse</li> <li>• <b>azimuth</b> - orientation of the ellipse</li> <li>• <b>segments</b> - optional argument for polygon segmentation. By default this value is 36</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_ellipse(make_point(10,10), 5, 2, 90, 4))</code> → <code>,Polygon ((15 10, 10 8, 5 10, 10 12, 15 10))'</code></li> <li>• <code>geom_to_wkt(make_ellipse(make_point(10,10,5), 5, 2, 90, 4))</code> → <code>,PolygonZ ((15 10 5, 10 8 5, 5 10 5, 10 12 5, 15 10 5))'</code></li> <li>• <code>geom_to_wkt(make_ellipse(make_point(10,10,5,30), 5, 2, 90, 4))</code> → <code>,PolygonZM ((15 10 5 30, 10 8 5 30, 5 10 5 30, 10 12 5 30, 15 10 5 30))'</code></li> </ul>

## make\_line

Creates a line geometry from a series of point geometries.

### List of arguments variant

Line vertices are specified as separate arguments to the function.

Syntax	<code>make_line(point1, point2, ...)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>point</b> - a point geometry (or array of points)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_line(make_point(2,4),make_point(3,5))) → ,LineString(2 4, 3 5)‘</code></li> <li>• <code>geom_to_wkt(make_line(make_point(2,4),make_point(3,5),make_point(9,7))) → ,LineString(2 4, 3 5, 9 7)‘</code></li> </ul>

### Array variant

Line vertices are specified as an array of points.

Syntax	<code>make_line(array)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>array</b> - array of points</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_line(array(make_point(2,4),make_point(3,5),make_point(9,7)))) → ,LineString(2 4, 3 5, 9 7)‘</code></li> </ul>

## make\_point

Creates a point geometry from an x and y (and optional z and m) value.

Syntax	<code>make_point(x, y, [z], [m])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>x</b> - x coordinate of point</li> <li>• <b>y</b> - y coordinate of point</li> <li>• <b>z</b> - optional z coordinate of point</li> <li>• <b>m</b> - optional m value of point</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_point(2,4)) → ,Point(2 4)‘</code></li> <li>• <code>geom_to_wkt(make_point(2,4,6)) → ,PointZ(2 4 6)‘</code></li> <li>• <code>geom_to_wkt(make_point(2,4,6,8)) → ,PointZM(2 4 6 8)‘</code></li> </ul>



### make\_point\_m

Creates a point geometry from an x, y coordinate and m value.

Syntax	<code>make_point_m(x, y, m)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>x</b> - x coordinate of point</li> <li>• <b>y</b> - y coordinate of point</li> <li>• <b>m</b> - m value of point</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_point_m(2, 4, 6)) → ,PointM (2 4 6)‘</code></li> </ul>

### make\_polygon

Creates a polygon geometry from an outer ring and optional series of inner ring geometries.

Syntax	<code>make_polygon(outerRing, [innerRing1], [innerRing2], ...)</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>outerRing</b> - closed line geometry for polygon's outer ring</li> <li>• <b>innerRing</b> - optional closed line geometry for inner ring</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_polygon(geom_from_wkt('LINESTRING( 0 0, 0 1, 1 1, 1 0, 0 0 )')))) → ,Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))‘</code></li> <li>• <code>geom_to_wkt(make_polygon(geom_from_wkt('LINESTRING( 0 0, 0 1, 1 1, 1 0, 0 0 )'), geom_from_wkt('LINESTRING( 0.1 0.1, 0.1 0.2, 0.2 0.2, 0.2 0.1, 0.1 0.1 )'), geom_from_wkt('LINESTRING( 0.8 0.8, 0.8 0.9, 0.9 0.9, 0.9 0.8, 0.8 0.8 )')))) → ,Polygon ((0 0, 0 1, 1 1, 1 0, 0 0),(0.1 0.1, 0.1 0.2, 0.2 0.2, 0.2 0.1, 0.1 0.1),(0.8 0.8, 0.8 0.9, 0.9 0.9, 0.9 0.8, 0.8 0.8))‘</code></li> </ul>

### make\_rectangle\_3points

Creates a rectangle from 3 points.

Syntax	<code>make_rectangle_3points(point1, point2, point3, [option=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>point1</b> - First point.</li> <li>• <b>point2</b> - Second point.</li> <li>• <b>point3</b> - Third point.</li> <li>• <b>option</b> - An optional argument to construct the rectangle. By default this value is 0. Value can be 0 (distance) or 1 (projected). Option distance: Second distance is equal to the distance between 2nd and 3rd point. Option projected: Second distance is equal to the distance of the perpendicular projection of the 3rd point on the segment or its extension.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_rectangle_3points(make_point(0, 0), make_point(0,5), make_point(5, 5), 0)) → ,Polygon ((0 0, 0 5, 5 5, 5 0, 0 0))'</code></li> <li>• <code>geom_to_wkt(make_rectangle_3points(make_point(0, 0), make_point(0,5), make_point(5, 3), 1)) → ,Polygon ((0 0, 0 5, 5 5, 5 0, 0 0))'</code></li> </ul>

## make\_regular\_polygon

Creates a regular polygon.

Syntax	<code>make_regular_polygon(center, radius, number_sides, [circle=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>center</b> - center of the regular polygon</li> <li>• <b>radius</b> - second point. The first if the regular polygon is inscribed. The midpoint of the first side if the regular polygon is circumscribed.</li> <li>• <b>number_sides</b> - Number of sides/edges of the regular polygon</li> <li>• <b>circle</b> - Optional argument to construct the regular polygon. By default this value is 0. Value can be 0 (inscribed) or 1 (circumscribed)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_regular_polygon(make_point(0,0), make_point(0,5), 5)) → ,Polygon ((0 5, 4.76 1.55, 2.94 -4.05, -2.94 -4.05, -4.76 1.55, 0 5))'</code></li> <li>• <code>geom_to_wkt(make_regular_polygon(make_point(0,0), project(make_point(0,0), 4.0451, radians(36)), 5)) → ,Polygon ((0 5, 4.76 1.55, 2.94 -4.05, -2.94 -4.05, -4.76 1.55, 0 5))'</code></li> </ul>

## make\_square

Creates a square from a diagonal.

Syntax	<code>make_square(point1, point2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>point1</b> - First point of the diagonal</li> <li>• <b>point2</b> - Last point of the diagonal</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_square(make_point(0,0), make_point(5,5))) → ,Polygon ((0 0, -0 5, 5 5, 5 0, 0 0))</code></li> <li>• <code>geom_to_wkt(make_square(make_point(5,0), make_point(5,5))) → ,Polygon ((5 0, 2.5 2.5, 5 5, 7.5 2.5, 5 0))</code></li> </ul>

## make\_triangle

Creates a triangle polygon.

Syntax	<code>make_triangle(point1, point2, point3)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>point1</b> - first point of the triangle</li> <li>• <b>point2</b> - second point of the triangle</li> <li>• <b>point3</b> - third point of the triangle</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_triangle(make_point(0,0), make_point(5,5), make_point(0,10))) → ,Triangle ((0 0, 5 5, 0 10, 0 0))</code></li> <li>• <code>geom_to_wkt(boundary(make_triangle(make_point(0,0), make_point(5,5), make_point(0,10)))) → ,LineString (0 0, 5 5, 0 10, 0 0)</code></li> </ul>

## make\_valid

Returns a valid geometry or an empty geometry if the geometry could not be made valid.

Syntax	<code>make_valid(geometry, [method=structure], [keep_collapsed=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>method</b> - repair algorithm. May be either <code>structure</code> or <code>linework</code>. The <code>linework</code> option combines all rings into a set of noded lines and then extracts valid polygons from that linework. The <code>structure</code> method first makes all rings valid and then merges shells and subtracts holes from shells to generate valid result. Assumes that holes and shells are correctly categorized.</li> <li>• <b>keep_collapsed</b> - if set to true, then components that have collapsed into a lower dimensionality will be kept. For example, a ring collapsing to a line, or a line collapsing to a point.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(make_valid(geom_from_wkt('POLYGON((3 2, 4 1, 5 8, 3 2, 4 2))')))</code> → <code>,Polygon((3 2, 5 8, 4 1, 3 2))</code></li> <li>• <code>geom_to_wkt(make_valid(geom_from_wkt('POLYGON((3 2, 4 1, 5 8, 3 2, 4 2))'), 'linework'))</code> → <code>,GeometryCollection(Polygon((5 8, 4 1, 3 2, 5 8)),LineString(3 2, 4 2))</code></li> <li>• <code>geom_to_wkt(make_valid(geom_from_wkt('POLYGON((3 2, 4 1, 5 8))'), method='linework'))</code> → <code>,Polygon((3 2, 4 1, 5 8, 3 2))</code></li> <li>• <code>make_valid(geom_from_wkt('LINESTRING(0 0)'))</code> → An empty geometry</li> </ul>

Further reading: [is\\_valid](#), [Fix geometries](#) algorithm

### minimal\_circle

Returns the minimal enclosing circle of a geometry. It represents the minimum circle that encloses all geometries within the set.

Syntax	<code>minimal_circle(geometry, [segments=36])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>segments</b> - optional argument for polygon segmentation. By default this value is 36</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(minimal_circle(geom_from_wkt('LINESTRING(0 5, 0 -5, 2 1)'), 4))</code> → <code>,Polygon((0 5, 5 -0, -0 -5, -5 0, 0 5))</code></li> <li>• <code>geom_to_wkt(minimal_circle(geom_from_wkt('MULTIPOINT(1 2, 3 4, 3 2)'), 4))</code> → <code>,Polygon((3 4, 3 2, 1 2, 1 4, 3 4))</code></li> </ul>

Further reading: [Minimum enclosing circles](#) algorithm

## nodes\_to\_points

Returns a multipoint geometry consisting of every node in the input geometry.

Syntax	<code>nodes_to_points(geometry, [ignore_closing_nodes=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> <li>• <b>ignore_closing_nodes</b> - optional argument specifying whether to include duplicate nodes which close lines or polygons rings. Defaults to false, set to true to avoid including these duplicate nodes in the output collection.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(nodes_to_points(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')))</code> → <code>,MultiPoint((0 0),(1 1),(2 2))</code></li> <li>• <code>geom_to_wkt(nodes_to_points(geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1)')),true))</code> → <code>,MultiPoint((-1 -1),(4 0),(4 2),(0 2))</code></li> </ul>

Further reading: [Extract vertices](#) algorithm

## num\_geometries

Returns the number of geometries in a geometry collection, or the number of parts in a multi-part geometry. The function returns NULL if the input geometry is not a collection.

Syntax	<code>num_geometries(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry collection or multi-part geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>num_geometries(geom_from_wkt('GEOMETRYCOLLECTION(POINT(0 1), POINT(0 0), POINT(1 0), POINT(1 1))'))</code> → 4</li> <li>• <code>num_geometries(geom_from_wkt('MULTIPOINT((0 1), (0 0), (1 0))'))</code> → 3</li> </ul>

## num\_interior\_rings

Returns the number of interior rings in a polygon or geometry collection, or NULL if the input geometry is not a polygon or collection.

Syntax	<code>num_interior_rings(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - input geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>num_interior_rings(geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1), (-0.1 -0.1, 0.4 0, 0.4 0.2, 0 0.2, -0.1 -0.1))'))</code> → 1</li> </ul>

## num\_points

Returns the number of vertices in a geometry.

Syntax	<code>num_points(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>num_points(@geometry)</code> → number of vertices in the current feature's geometry</li> </ul>

## num\_rings

Returns the number of rings (including exterior rings) in a polygon or geometry collection, or NULL if the input geometry is not a polygon or collection.

Syntax	<code>num_rings(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - input geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>num_rings(geom_from_wkt('POLYGON((-1 -1, 4 0, 4 2, 0 2, -1 -1), (-0.1 -0.1, 0.4 0, 0.4 0.2, 0 0.2, -0.1 -0.1))'))</code> → 2</li> </ul>

## offset\_curve

Returns a geometry formed by offsetting a linestring geometry to the side. Distances are in the Spatial Reference System of this geometry.

Syntax	<code>offset_curve(geometry, distance, [segments=8], [join=1], [miter_limit=2.0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a (multi)linestring geometry</li> <li>• <b>distance</b> - offset distance. Positive values will be buffered to the left of lines, negative values to the right</li> <li>• <b>segments</b> - number of segments to use to represent a quarter circle when a round join style is used. A larger number results in a smoother line with more nodes.</li> <li>• <b>join</b> - join style for corners, where 1 = round, 2 = miter and 3 = bevel</li> <li>• <b>miter_limit</b> - limit on the miter ratio used for very sharp corners (when using miter joins only)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>offset_curve(@geometry, 10.5)</code> → line offset to the left by 10.5 units</li> <li>• <code>offset_curve(@geometry, -10.5)</code> → line offset to the right by 10.5 units</li> <li>• <code>offset_curve(@geometry, 10.5, segments:=16, join:=1)</code> → line offset to the left by 10.5 units, using more segments to result in a smoother curve</li> <li>• <code>offset_curve(@geometry, 10.5, join:=3)</code> → line offset to the left by 10.5 units, using a beveled join</li> </ul>

Further reading: [Offset lines](#) algorithm

## order\_parts

Sortiert die Teile einer Multi-Geometrie nach gegebenem Kriterium

Syntax	<code>order_parts(geometry, orderby, [ascending=true])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a multi-type geometry</li> <li>• <b>orderby</b> - an expression string defining the order criteria</li> <li>• <b>ascending</b> - boolean, True for ascending, False for descending</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(order_parts(geom_from_wkt('MultiPolygon (((1 1, 5 1, 5 5, 1 5, 1 1)), ((1 1, 9 1, 9 9, 1 9, 1 1)))'), 'area(@geometry)', False)) → ,MultiPolygon(((1 1, 9 1, 9 9, 1 9, 1 1)),((1 1, 5 1, 5 5, 1 5, 1 1)))</code></li> <li>• <code>geom_to_wkt(order_parts(geom_from_wkt('LineString(1 2, 3 2, 4 3)'), '1', True)) → ,LineString(1 2, 3 2, 4 3)</code></li> </ul>

## oriented\_bbox

Returns a geometry which represents the minimal oriented bounding box of an input geometry.

Syntax	<code>oriented_bbox(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(oriented_bbox(geom_from_wkt('MULTIPOINT(1 2, 3 4, 3 2)')) → ,Polygon((3 2, 3 4, 1 4, 1 2, 3 2))</code></li> </ul>

Further reading: *Oriented minimum bounding box* algorithm

## overlaps

Tests whether a geometry overlaps another. Returns TRUE if the geometries share space, are of the same dimension, but are not completely contained by each other.

Syntax	<code>overlaps(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlaps(geom_from_wkt('LINESTRING(3 5, 4 4, 5 5, 5 3)'), geom_from_wkt('LINESTRING(3 3, 4 4, 5 5)')) → TRUE</code></li> <li>• <code>overlaps(geom_from_wkt('LINESTRING(0 0, 1 1)'), geom_from_wkt('LINESTRING(3 3, 4 4, 5 5)')) → FALSE</code></li> </ul>

## overlay\_contains

Returns whether the current feature spatially contains at least one feature from a target layer, or an array of expression-based results for the features in the target layer contained in the current feature.

Read more on the underlying GEOS „Contains“ predicate, as described in PostGIS [ST\\_Contains](#) function.

Syntax	<code>overlay_contains(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_contains('regions')</code> → TRUE if the current feature spatially contains a region</li> <li>• <code>overlay_contains('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature spatially contains a region with a population greater than 10000</li> <li>• <code>overlay_contains('regions', name)</code> → an array of names, for the regions contained in the current feature</li> <li>• <code>array_to_string(overlay_contains('regions', name))</code> → a string as a comma separated list of names, for the regions contained in the current feature</li> <li>• <code>array_sort(overlay_contains(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions contained in the current feature and with a population greater than 10000</li> <li>• <code>overlay_contains(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions contained in the current feature</li> </ul>

Further reading: [contains](#), [array manipulation](#), [Auswahl nach der Lage](#) algorithm

## overlay\_crosses

Returns whether the current feature spatially crosses at least one feature from a target layer, or an array of expression-based results for the features in the target layer crossed by the current feature.

Read more on the underlying GEOS „Crosses“ predicate, as described in PostGIS [ST\\_Crosses](#) function.



Syntax	<code>overlay_crosses(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_crosses('regions')</code> → TRUE if the current feature spatially crosses a region</li> <li>• <code>overlay_crosses('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature spatially crosses a region with a population greater than 10000</li> <li>• <code>overlay_crosses('regions', name)</code> → an array of names, for the regions crossed by the current feature</li> <li>• <code>array_to_string(overlay_crosses('regions', name))</code> → a string as a comma separated list of names, for the regions crossed by the current feature</li> <li>• <code>array_sort(overlay_crosses(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions crossed by the current feature and with a population greater than 10000</li> <li>• <code>overlay_crosses(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions crossed by the current feature</li> </ul>

Further reading: [crosses](#), [array manipulation](#), [Auswahl nach der Lage](#) algorithm

## overlay\_disjoint

Returns whether the current feature is spatially disjoint from all the features of a target layer, or an array of expression-based results for the features in the target layer that are disjoint from the current feature.

Read more on the underlying GEOS „Disjoint“ predicate, as described in PostGIS [ST\\_Disjoint](#) function.

Syntax	<code>overlay_disjoint(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_disjoint('regions')</code> → TRUE if the current feature is spatially disjoint from all the regions</li> <li>• <code>overlay_disjoint('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature is spatially disjoint from all the regions with a population greater than 10000</li> <li>• <code>overlay_disjoint('regions', name)</code> → an array of names, for the regions spatially disjoint from the current feature</li> <li>• <code>array_to_string(overlay_disjoint('regions', name))</code> → a string as a comma separated list of names, for the regions spatially disjoint from the current feature</li> <li>• <code>array_sort(overlay_disjoint(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions spatially disjoint from the current feature and with a population greater than 10000</li> <li>• <code>overlay_disjoint(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions spatially disjoint from the current feature</li> </ul>

Further reading: [disjoint](#), [array manipulation](#), [Auswahl nach der Lage](#) algorithm

## overlay\_equals

Returns whether the current feature spatially equals to at least one feature from a target layer, or an array of expression-based results for the features in the target layer that are spatially equal to the current feature.

Read more on the underlying GEOS „Equals“ predicate, as described in PostGIS [ST\\_Equals](#) function.

Syntax	<code>overlay_equals(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_equals('regions')</code> → TRUE if the current feature is spatially equal to a region</li> <li>• <code>overlay_equals('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature is spatially equal to a region with a population greater than 10000</li> <li>• <code>overlay_equals('regions', name)</code> → an array of names, for the regions spatially equal to the current feature</li> <li>• <code>array_to_string(overlay_equals('regions', name))</code> → a string as a comma separated list of names, for the regions spatially equal to the current feature</li> <li>• <code>array_sort(overlay_equals(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions spatially equal to the current feature and with a population greater than 10000</li> <li>• <code>overlay_equals(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions spatially equal to the current feature</li> </ul>

Further reading: [array manipulation](#), [Auswahl nach der Lage](#) algorithm

## overlay\_intersects

Returns whether the current feature spatially intersects at least one feature from a target layer, or an array of expression-based results for the features in the target layer intersected by the current feature.

Read more on the underlying GEOS „Intersects“ predicate, as described in PostGIS [ST\\_Intersects](#) function.

Syntax	<code>overlay_intersects(layer, [expression], [filter], [limit], [cache=false], [min_overlap], [min_inscribed_circle_radius], [return_details], [sort_by_intersection_size])</code> [] marks optional arguments
--------	--

#### Arguments

- **layer** - the layer whose overlay is checked
- **expression** - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.
- **filter** - an optional expression to filter the target features to check. If not set, all the features will be checked.
- **limit** - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.
- **cache** - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)
- **min\_overlap** - defines an optional exclusion filter:
  - for polygons, a minimum area in current feature squared units for the intersection. If the intersection results in multiple polygons the intersection will be returned if at least one of the polygons has an area greater or equal to the value
  - for lines, a minimum length in current feature units. If the intersection results in multiple lines the intersection will be returned if at least one of the lines has a length greater or equal to the value.
- **min\_inscribed\_circle\_radius** - defines an optional exclusion filter (for polygons only): minimum radius in current feature units for the maximum inscribed circle of the intersection. If the intersection results in multiple polygons the intersection will be returned if at least one of the polygons has a radius for the maximum inscribed circle greater or equal to the value.  
Read more on the underlying GEOS predicate, as described in [PostGIS ST\\_MaximumInscribedCircle](#) function.  
This argument requires GEOS >= 3.9.
- **return\_details** - Set this to true to return a list of maps containing (key names in quotes) the feature ,id', the expression ,result' and the ,overlap' value. The ,radius' of the maximum inscribed circle is also returned when the target layer is a polygon. Only valid when used with the expression parameter
- **sort\_by\_intersection\_size** - only valid when used with an expression, set this to ,des' to return the results ordered by the overlap value in descending order or set this to ,asc' for ascending order.

#### Beispiele

- `overlay_intersects('regions')` → TRUE if the current feature spatially intersects a region
- `overlay_intersects('regions', filter:= population > 10000)` → TRUE if the current feature spatially intersects a region with a population greater than 10000
- `overlay_intersects('regions', name)` → an array of names, for the regions intersected by the current feature
- `array_to_string(overlay_intersects('regions', name))` → a string as a comma separated list of names, for the regions intersected by the current feature
- `array_sort(overlay_intersects(layer:='regions', expression:="name", filter:= population > 10000))` → an ordered array of names, for the regions intersected by the current feature and with a population greater than 10000
- `overlay_intersects(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)` → an array of geometries (in WKT), for up to two regions intersected by the current feature
- `overlay_intersects(layer:='regions', min_overlap:=0.54)` → TRUE if the current feature spatially intersects a region and the intersection area (of at least one of the parts in case of multipolygons) is greater or equal to 0.54
- `overlay_intersects(layer:='regions', min_inscribed_circle_radius:=0.54)` → TRUE if the current feature spatially intersects a region and the intersection area maximum inscribed circle's radius (of at least one of the parts in case of multipart) is greater or equal to 0.54

- `overlay_intersects(layer:='regions', expression:= geom_to_wkt(@geometry), return_details:=true)` → an array of maps containing ,id', ,result', ,overlap' and ,radius'
- `overlay_intersects(layer:='regions', expression:=`

Further reading: [intersects](#), [array manipulation](#), [Auswahl nach der Lage](#) algorithm

## overlay\_nearest

Returns whether the current feature has feature(s) from a target layer within a given distance, or an array of expression-based results for the features in the target layer within a distance from the current feature.

Note: This function can be slow and consume a lot of memory for large layers.

Syntax	<code>overlay_nearest(layer, [expression], [filter], [limit=1], [max_distance], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the target layer</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features in the target layer will be used.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, only the nearest feature will be returned. If set to -1, returns all the matching features.</li> <li>• <b>max_distance</b> - an optional distance to limit the search of matching features. If not set, all the features in the target layer will be used.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_nearest('airports')</code> → TRUE if the „airports“ layer has at least one feature</li> <li>• <code>overlay_nearest('airports', max_distance:= 5000)</code> → TRUE if there is an airport within a distance of 5000 map units from the current feature</li> <li>• <code>overlay_nearest('airports', name)</code> → the name of the closest airport to the current feature, as an array</li> <li>• <code>array_to_string(overlay_nearest('airports', name))</code> → the name of the closest airport to the current feature, as a string</li> <li>• <code>overlay_nearest(layer:='airports', expression:= name, max_distance:= 5000)</code> → the name of the closest airport within a distance of 5000 map units from the current feature, as an array</li> <li>• <code>overlay_nearest(layer:='airports', expression:="name", filter:= "Use"='Civilian', limit:=3)</code> → an array of names, for up to the three closest civilian airports ordered by distance</li> <li>• <code>overlay_nearest(layer:='airports', expression:="name", limit:= -1, max_distance:= 5000)</code> → an array of names, for all the airports within a distance of 5000 map units from the current feature, ordered by distance</li> </ul>

Further reading: [array manipulation](#), [Join attributes by nearest](#) algorithm

## overlay\_touches

Returns whether the current feature spatially touches at least one feature from a target layer, or an array of expression-based results for the features in the target layer touched by the current feature.

Read more on the underlying GEOS „Touches“ predicate, as described in PostGIS [ST\\_Touches](#) function.

Syntax	<code>overlay_touces(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_touces('regions')</code> → TRUE if the current feature spatially touches a region</li> <li>• <code>overlay_touces('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature spatially touches a region with a population greater than 10000</li> <li>• <code>overlay_touces('regions', name)</code> → an array of names, for the regions touched by the current feature</li> <li>• <code>string_to_array(overlay_touces('regions', name))</code> → a string as a comma separated list of names, for the regions touched by the current feature</li> <li>• <code>array_sort(overlay_touces(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions touched by the current feature and with a population greater than 10000</li> <li>• <code>overlay_touces(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions touched by the current feature</li> </ul>

Further reading: *[touches](#)*, *[array manipulation](#)*, *[Auswahl nach der Lage](#)* algorithm

## overlay\_within

Returns whether the current feature is spatially within at least one feature from a target layer, or an array of expression-based results for the features in the target layer that contain the current feature.

Read more on the underlying GEOS „Within“ predicate, as described in PostGIS [ST\\_Within](#) function.

Syntax	<code>overlay_within(layer, [expression], [filter], [limit], [cache=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the layer whose overlay is checked</li> <li>• <b>expression</b> - an optional expression to evaluate on the features from the target layer. If not set, the function will just return a boolean indicating whether there is at least one match.</li> <li>• <b>filter</b> - an optional expression to filter the target features to check. If not set, all the features will be checked.</li> <li>• <b>limit</b> - an optional integer to limit the number of matching features. If not set, all the matching features will be returned.</li> <li>• <b>cache</b> - set this to true to build a local spatial index (most of the time, this is unwanted, unless you are working with a particularly slow data provider)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>overlay_within('regions')</code> → TRUE if the current feature is spatially within a region</li> <li>• <code>overlay_within('regions', filter:= population &gt; 10000)</code> → TRUE if the current feature is spatially within a region with a population greater than 10000</li> <li>• <code>overlay_within('regions', name)</code> → an array of names, for the regions containing the current feature</li> <li>• <code>array_to_string(overlay_within('regions', name))</code> → a string as a comma separated list of names, for the regions containing the current feature</li> <li>• <code>array_sort(overlay_within(layer:='regions', expression:="name", filter:= population &gt; 10000))</code> → an ordered array of names, for the regions containing the current feature and with a population greater than 10000</li> <li>• <code>overlay_within(layer:='regions', expression:= geom_to_wkt(@geometry), limit:=2)</code> → an array of geometries (in WKT), for up to two regions containing the current feature</li> </ul>

Further reading: *within*, *array manipulation*, *Auswahl nach der Lage* algorithm

## \$perimeter

Returns the perimeter length of the current feature. The perimeter calculated by this function respects both the current project's ellipsoid setting and distance unit settings. For example, if an ellipsoid has been set for the project then the calculated perimeter will be ellipsoidal, and if no ellipsoid is set then the calculated perimeter will be planimetric.

Syntax	<code>\$perimeter</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$perimeter</code> → 42</li> </ul>

### perimeter

Returns the perimeter of a geometry polygon object. Calculations are always planimetric in the Spatial Reference System (SRS) of this geometry, and the units of the returned perimeter will match the units for the SRS. This differs from the calculations performed by the \$perimeter function, which will perform ellipsoidal calculations based on the project's ellipsoid and distance unit settings.

Syntax	perimeter(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - polygon geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>perimeter(geom_from_wkt('POLYGON((0 0, 4 0, 4 2, 0 2, 0 0))'))</code> → 12.0</li> </ul>

### point\_n

Returns a specific node from a geometry.

Syntax	point_n(geometry, index)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - geometry object</li> <li><b>index</b> - index of node to return, where 1 is the first node; if the value is negative, the selected vertex index will be its total count minus the absolute value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(point_n(geom_from_wkt('POLYGON((0 0, 4 0, 4 2, 0 2, 0 0))'), 2))</code> → ,Point (4 0)‘</li> </ul>

Further reading: [Extract specific vertices](#) algorithm

### point\_on\_surface

Returns a point guaranteed to lie on the surface of a geometry.

Syntax	point_on_surface(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>point_on_surface(@geometry)</code> → a point geometry</li> </ul>

Further reading: [Point on Surface](#) algorithm



## pole\_of\_inaccessibility

Calculates the approximate pole of inaccessibility for a surface, which is the most distant internal point from the boundary of the surface. This function uses the 'polylabel' algorithm (Vladimir Agafonkin, 2016), which is an iterative approach guaranteed to find the true pole of inaccessibility within a specified tolerance. More precise tolerances require more iterations and will take longer to calculate.

Syntax	pole_of_inaccessibility(geometry, tolerance)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>tolerance</b> - maximum distance between the returned point and the true pole location</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(pole_of_inaccessibility(geom_from_wkt('POLYGON((0 1, 0 9, 3 10, 3 3, 10 3, 10 1, 0 1))'), 0.1)) → ,Point(1.546875 2.546875)'</code></li> </ul>

Further reading: *Pole of inaccessibility* algorithm

## project

Returns a point projected from a start point using a distance, a bearing (azimuth) and an elevation in radians.

Syntax	project(point, distance, azimuth, [elevation]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>point</b> - start point</li> <li>• <b>distance</b> - distance to project</li> <li>• <b>azimuth</b> - azimuth in radians clockwise, where 0 corresponds to north</li> <li>• <b>elevation</b> - angle of inclination in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(project(make_point(1, 2), 3, radians(270))) → ,Point(-2, 2)'</code></li> </ul>

Further reading: *Project points (Cartesian)* algorithm

## relate

Tests the Dimensional Extended 9 Intersection Model (DE-9IM) representation of the relationship between two geometries.

### Relationship variant

Returns the Dimensional Extended 9 Intersection Model (DE-9IM) representation of the relationship between two geometries.

Syntax	relate(geometry, geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>relate(geom_from_wkt('LINESTRING(40 40,120 120)'), geom_from_wkt('LINESTRING(40 40,60 120)')) → ,FF1F00102'</code></li> </ul>

### Pattern match variant

Tests whether the DE-9IM relationship between two geometries matches a specified pattern.

Syntax	relate(geometry, geometry, pattern)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>geometry</b> - a geometry</li> <li>• <b>pattern</b> - DE-9IM pattern to match</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>relate( geom_from_wkt( 'LINESTRING(40 40,120 120)' ), geom_from_wkt( 'LINESTRING(40 40,60 120)' ), '**1F001**' ) → TRUE</code></li> </ul>

### reverse

Reverses the direction of a line string by reversing the order of its vertices.

Syntax	reverse(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(reverse(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')) ) → ,LINESTRING(2 2,1 1,0 0)‘</code></li> </ul>

Further reading: *Reverse line direction* algorithm

### rotate

Returns a rotated version of a geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	rotate(geometry, rotation, [center=NULL], [per_part=false]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>rotation</b> - clockwise rotation in degrees</li> <li>• <b>center</b> - rotation center point. If not specified, the center of the geometry's bounding box is used.</li> <li>• <b>per_part</b> - apply rotation per part. If true, then rotation will apply around the center of each part's bounding box when the input geometry is multipart and an explicit rotation center point is not specified.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>rotate(@geometry, 45, make_point(4, 5)) → geometry rotated 45 degrees clockwise around the (4, 5) point</code></li> <li>• <code>rotate(@geometry, 45) → geometry rotated 45 degrees clockwise around the center of its bounding box</code></li> </ul>

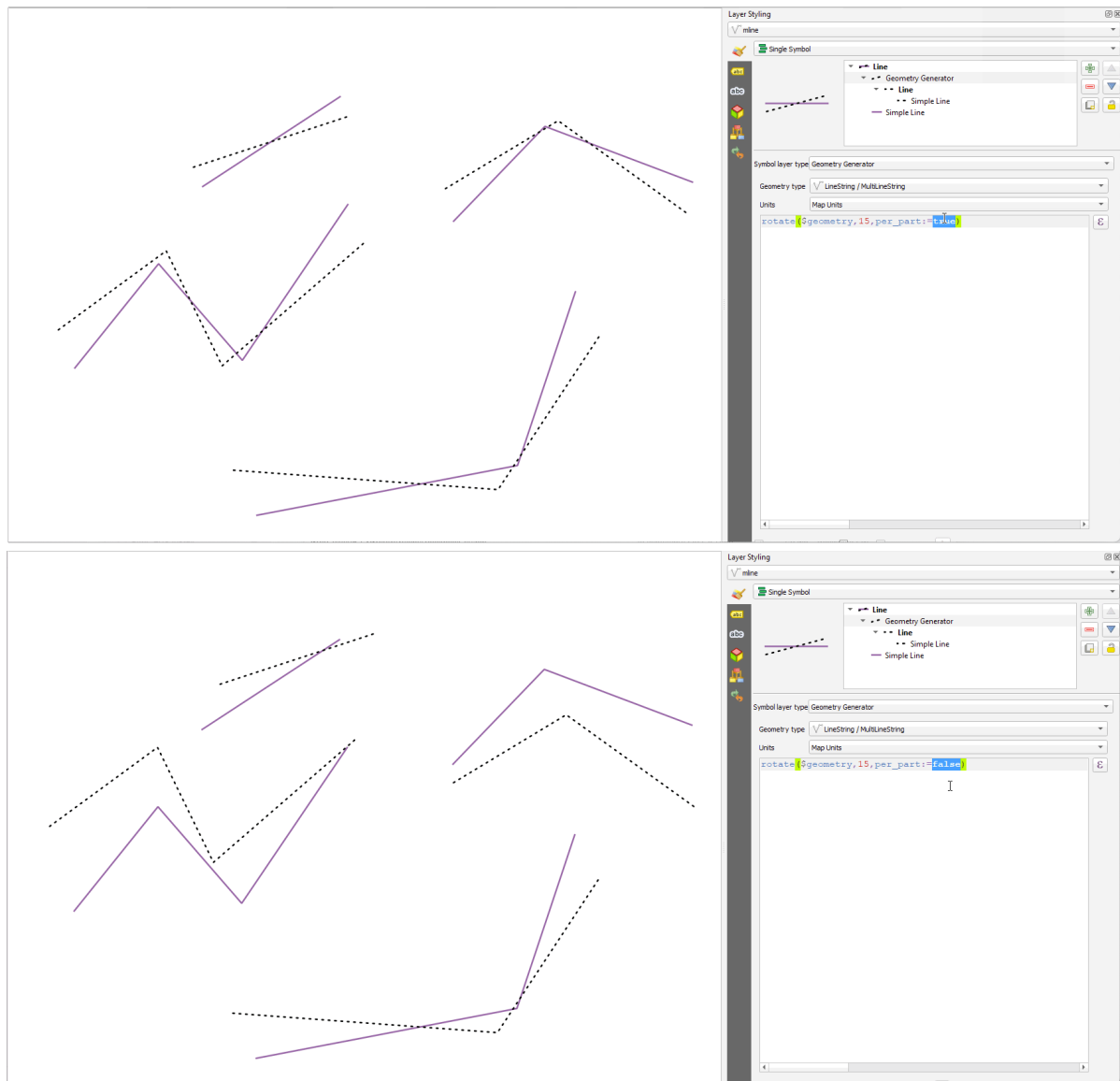


Abb. 13.7: Rotating features

## roundness

Calculates how close a polygon shape is to a circle. The function Returns TRUE when the polygon shape is a perfect circle and 0 when it is completely flat.

Syntax	roundness(geometry)
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a polygon</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>round(roundness(geom_from_wkt('POLYGON(( 0 0, 0 1, 1 1, 1 0, 0 0))'), 3) → 0.785</code></li> <li><code>round(roundness(geom_from_wkt('POLYGON(( 0 0, 0 0.1, 1 0.1, 1 0, 0 0))'), 3) → 0.260</code></li> </ul>

Further reading: [Roundness algorithm](#)

### scale

Returns a scaled version of a geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	<code>scale(geometry, x_scale, y_scale, [center])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>x_scale</b> - x-axis scaling factor</li> <li>• <b>y_scale</b> - y-axis scaling factor</li> <li>• <b>center</b> - scaling center point. If not specified, the center of the geometry's bounding box is used.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>scale(@geometry, 2, 0.5, make_point(4, 5))</code> → geometry scaled twice horizontally and halved vertically, around the (4, 5) point</li> <li>• <code>scale(@geometry, 2, 0.5)</code> → geometry twice horizontally and halved vertically, around the center of its bounding box</li> </ul>

### segments\_to\_lines

Returns a multi line geometry consisting of a line for every segment in the input geometry.

Syntax	<code>segments_to_lines(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(segments_to_lines(geom_from_wkt('LINESTRING(0 0, 1 1, 2 2)')))</code> → <code>MultiLineString((0 0, 1 1),(1 1, 2 2))</code></li> </ul>

Further reading: [Explode lines](#) algorithm

### shared\_paths

Returns a collection containing paths shared by the two input geometries. Those going in the same direction are in the first element of the collection, those going in the opposite direction are in the second element. The paths themselves are given in the direction of the first geometry.

Syntax	<code>shared_paths(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a LineString/MultiLineString geometry</li> <li>• <b>geometry2</b> - a LineString/MultiLineString geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(shared_paths(geom_from_wkt('MULTILINESTRING((26 125,26 200,126 200,126 125,26 125),(51 150,101 150,76 175,51 150))'),geom_from_wkt('LINESTRING(151 100,126 156.25,126 125,90 161, 76 175)')))</code> → <code>,GeometryCollection (MultiLineString ((126 156.25, 126 125),(101 150, 90 161),(90 161, 76 175)),MultiLineString EMPTY)</code></li> <li>• <code>geom_to_wkt(shared_paths(geom_from_wkt('LINESTRING(76 175,90 161,126 125,126 156.25,151 100)'),geom_from_wkt('MULTILINESTRING((26 125,26 200,126 200,126 125,26 125),(51 150,101 150,76 175,51 150))')))</code> → <code>,GeometryCollection (MultiLineString EMPTY,MultiLineString ((76 175, 90 161),(90 161, 101 150),(126 125, 126 156.25)))</code></li> </ul>

### shortest\_line

Returns the shortest line joining geometry1 to geometry2. The resultant line will start at geometry1 and end at geometry2.

Syntax	<code>shortest_line(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - geometry to find shortest line from</li> <li>• <b>geometry2</b> - geometry to find shortest line to</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(shortest_line(geom_from_wkt('LINESTRING (20 80, 98 190, 110 180, 50 75)'),geom_from_wkt('POINT(100 100)')))</code> → <code>,LineString(73.0769 115.384, 100 100)</code></li> </ul>

### simplify

Simplifies a geometry by removing nodes using a distance based threshold (ie, the Douglas Peucker algorithm). The algorithm preserves large deviations in geometries and reduces the number of vertices in nearly straight segments.

Syntax	<code>simplify(geometry, tolerance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>tolerance</b> - maximum deviation from straight segments for points to be removed</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(simplify(geometry:=geom_from_wkt('LineString(0 0, 5 0.1, 10 0)'),tolerance:=5))</code> → <code>,LineString(0 0, 10 0)</code></li> </ul>

Further reading: *Simplify* algorithm

## simplify\_vw

Simplifies a geometry by removing nodes using an area based threshold (ie, the Visvalingam-Whyatt algorithm). The algorithm removes vertices which create small areas in geometries, e.g., narrow spikes or nearly straight segments.

Syntax	<code>simplify_vw(geometry, tolerance)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>tolerance</b> - a measure of the maximum area created by a node for the node to be removed</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(simplify_vw(geometry:=geom_from_wkt('LineString(0 0, 5 0, 5.01 10, 5.02 0, 10 0)'), tolerance:=5)) → ,LineString(0 0, 10 0)'</code></li> </ul>

Further reading: *Simplify* algorithm

## single\_sided\_buffer

Returns a geometry formed by buffering out just one side of a linestring geometry. Distances are in the Spatial Reference System of this geometry.

Syntax	<code>single_sided_buffer(geometry, distance, [segments=8], [join=1], [miter_limit=2.0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a (multi)linestring geometry</li> <li>• <b>distance</b> - buffer distance. Positive values will be buffered to the left of lines, negative values to the right</li> <li>• <b>segments</b> - number of segments to use to represent a quarter circle when a round join style is used. A larger number results in a smoother buffer with more nodes.</li> <li>• <b>join</b> - join style for corners, where 1 = round, 2 = miter and 3 = bevel</li> <li>• <b>miter_limit</b> - limit on the miter ratio used for very sharp corners (when using miter joins only)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>single_sided_buffer(@geometry, 10.5)</code> → line buffered to the left by 10.5 units</li> <li>• <code>single_sided_buffer(@geometry, -10.5)</code> → line buffered to the right by 10.5 units</li> <li>• <code>single_sided_buffer(@geometry, 10.5, segments:=16, join:=1)</code> → line buffered to the left by 10.5 units, using more segments to result in a smoother buffer</li> <li>• <code>single_sided_buffer(@geometry, 10.5, join:=3)</code> → line buffered to the left by 10.5 units, using a beveled join</li> </ul>

Further reading: *Single sided buffer* algorithm

## sinuosity

Returns the sinuosity of a curve, which is the ratio of the curve length to the straight (2D) distance between its endpoints.

Syntax	<code>sinuosity(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - Input curve (circularstring, linestring)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>round(sinuosity(geom_from_wkt('LINESTRING(2 0, 2 2, 3 2, 3 3)'), 3) → 1.265</code></li> <li><code>sinuosity(geom_from_wkt('LINESTRING( 3 1, 5 1)')) → 1.0</code></li> </ul>

## smooth

Smooths a geometry by adding extra nodes which round off corners in the geometry. If input geometries contain Z or M values, these will also be smoothed and the output geometry will retain the same dimensionality as the input geometry.

Syntax	<code>smooth(geometry, [iterations=1], [offset=0.25], [min_length=-1], [max_angle=180])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li><b>geometry</b> - a geometry</li> <li><b>iterations</b> - number of smoothing iterations to apply. Larger numbers result in smoother but more complex geometries.</li> <li><b>offset</b> - value between 0 and 0.5 which controls how tightly the smoothed geometry follow the original geometry. Smaller values result in a tighter smoothing, larger values result in looser smoothing.</li> <li><b>min_length</b> - minimum length of segments to apply smoothing to. This parameter can be used to avoid placing excessive additional nodes in shorter segments of the geometry.</li> <li><b>max_angle</b> - maximum angle at node for smoothing to be applied (0-180). By lowering the maximum angle intentionally sharp corners in the geometry can be preserved. For instance, a value of 80 degrees will retain right angles in the geometry.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li><code>geom_to_wkt(smooth(geometry:=geom_from_wkt('LineString(0 0, 5 0, 5 5)'), iterations:=1, offset:=0.2, min_length:=-1, max_angle:=180)) → ,LineString(0 0, 4 0, 5 1, 5 5)'</code></li> </ul>

Further reading: [Smooth](#) algorithm

## square\_wave

Constructs square/rectangular waves along the boundary of a geometry.

Syntax	<code>square_wave(geometry, wavelength, amplitude, [strict=False])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>wavelength</b> - wavelength of square waveform</li> <li>• <b>amplitude</b> - amplitude of square waveform</li> <li>• <b>strict</b> - By default the wavelength argument is treated as a „maximum wavelength“, where the actual wavelength will be dynamically adjusted so that an exact number of square waves are created along the boundaries of the geometry. If the strict argument is set to true then the wavelength will be used exactly and an incomplete pattern may be used for the final waveform.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>square_wave(geom_from_wkt('LineString(0 0, 10 0)'), 3, 1)</code> → Square waves with wavelength 3 and amplitude 1 along the linestring</li> </ul>

### square\_wave\_randomized

Constructs randomized square/rectangular waves along the boundary of a geometry.

Syntax	<code>square_wave_randomized(geometry, min_wavelength, max_wavelength, min_amplitude, max_amplitude, [seed=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>min_wavelength</b> - minimum wavelength of waves</li> <li>• <b>max_wavelength</b> - maximum wavelength of waves</li> <li>• <b>min_amplitude</b> - minimum amplitude of waves</li> <li>• <b>max_amplitude</b> - maximum amplitude of waves</li> <li>• <b>seed</b> - specifies a random seed for generating waves. If the seed is 0, then a completely random set of waves will be generated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>square_wave_randomized(geom_from_wkt('LineString(0 0, 10 0)'), 2, 3, 0.1, 0.2)</code> → Randomly sized square waves with wavelengths between 2 and 3 and amplitudes between 0.1 and 0.2 along the linestring</li> </ul>

### start\_point

Returns the first node from a geometry.

Syntax	<code>start_point(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt(start_point(geom_from_wkt('LINESTRING(4 0, 4 2, 0 2)')))</code> → <code>Point(4 0)</code></li> </ul>

Further reading: *Extract specific vertices* algorithm



### straight\_distance\_2d

Returns the direct/euclidean distance between the first and last vertex of a geometry. The geometry must be a curve (circularstring, linestring).

Syntax	straight_distance_2d(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - The geometry.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>straight_distance_2d(geom_from_wkt('LINESTRING(1 0, 1 1)'))</code> → 1</li> <li>• <code>round(straight_distance_2d(geom_from_wkt('LINESTRING(1 4, 3 5, 5 0)')), 3)</code> → 5.657</li> </ul>

Further reading: *length*

### sym\_difference

Returns a geometry that represents the portions of two geometries that do not intersect.

Syntax	sym_difference(geometry1, geometry2)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( sym_difference( geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ), geom_from_wkt( 'LINESTRING(3 3, 8 8)' ) ) )</code> → 'LINESTRING(5 5, 8 8)'</li> </ul>

Further reading: *Symmetrical difference* algorithm

### tapered\_buffer

Creates a buffer along a line geometry where the buffer diameter varies evenly over the length of the line.

Syntax	tapered_buffer(geometry, start_width, end_width, [segments=8]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - input geometry. Must be a (multi)line geometry.</li> <li>• <b>start_width</b> - width of buffer at start of line,</li> <li>• <b>end_width</b> - width of buffer at end of line.</li> <li>• <b>segments</b> - number of segments to approximate quarter-circle curves in the buffer.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>tapered_buffer(geometry:=geom_from_wkt('LINESTRING(1 2, 4 2)'), start_width:=1, end_width:=2, segments:=8)</code> → A tapered buffer starting with a diameter of 1 and ending with a diameter of 2 along the linestring geometry.</li> </ul>

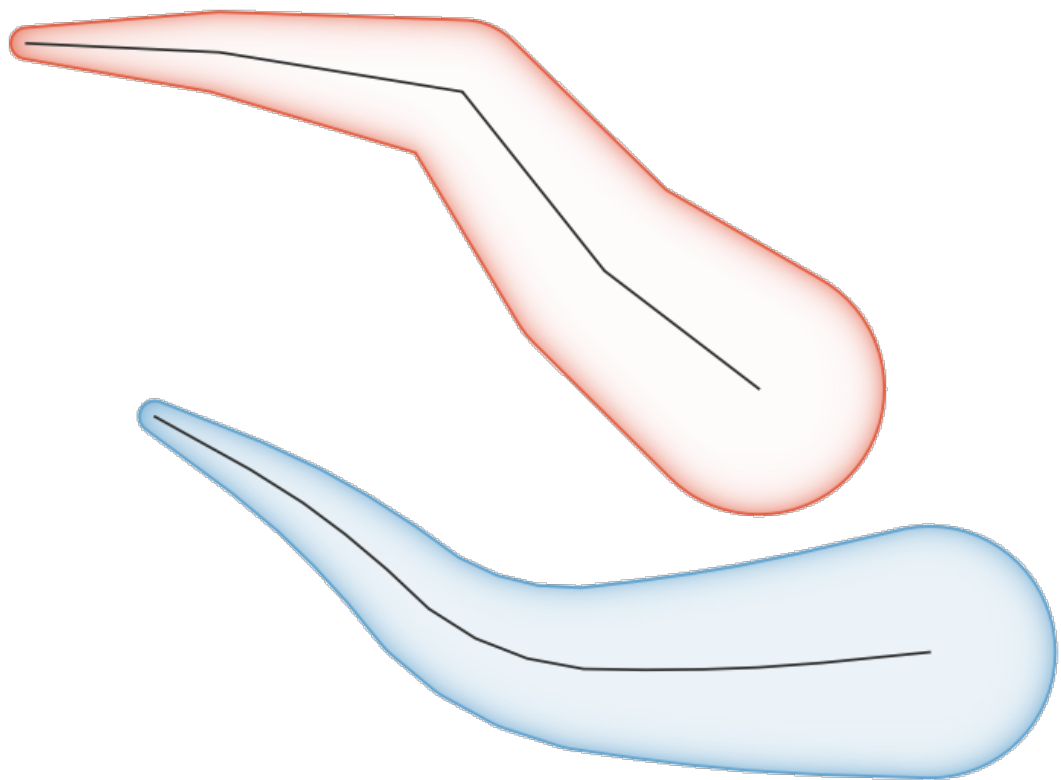


Abb. 13.8: Tapered buffer on line features

Further reading: *Tapered buffers* algorithm

**touches**

Tests whether a geometry touches another. Returns TRUE if the geometries have at least one point in common, but their interiors do not intersect.

Syntax	touches(geometry1, geometry2)
Arguments	<ul style="list-style-type: none"><li>• <b>geometry1</b> - a geometry</li><li>• <b>geometry2</b> - a geometry</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• touches( geom_from_wkt( 'LINESTRING(5 3, 4 4)' ), geom_from_wkt( 'LINESTRING(3 3, 4 4, 5 5)' ) ) → <b>TRUE</b></li><li>• touches( geom_from_wkt( 'POINT(4 4)' ), geom_from_wkt( 'POINT(5 5)' ) ) → <b>FALSE</b></li></ul>

Further reading: *overlay\_touches*

## transform

Returns the geometry transformed from a source CRS to a destination CRS.

Syntax	<code>transform(geometry, source_auth_id, dest_auth_id)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>source_auth_id</b> - the source auth CRS ID</li> <li>• <b>dest_auth_id</b> - the destination auth CRS ID</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( transform( make_point(488995.53240249, 7104473.38600835), 'EPSG:2154', 'EPSG:4326' ) ) → ,POINT(0 51)‘</code></li> </ul>

Further reading: *Reproject layer* algorithm

## translate

Returns a translated version of a geometry. Calculations are in the Spatial Reference System of this geometry.

Syntax	<code>translate(geometry, dx, dy)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>dx</b> - delta x</li> <li>• <b>dy</b> - delta y</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>translate(@geometry, 5, 10) → a geometry of the same type like the original one</code></li> </ul>

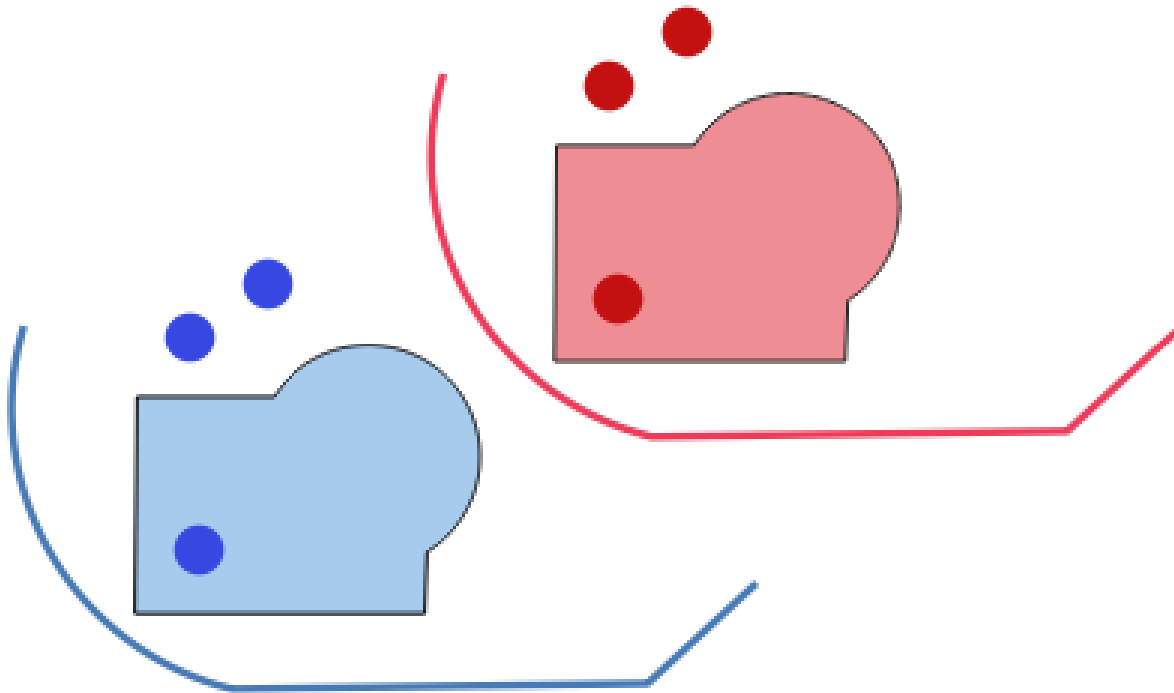


Abb. 13.9: Translating features

Further reading: *Translate* algorithm

### triangular\_wave

Constructs triangular waves along the boundary of a geometry.

Syntax	<code>triangular_wave(geometry, wavelength, amplitude, [strict=False])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>wavelength</b> - wavelength of triangular waveform</li> <li>• <b>amplitude</b> - amplitude of triangular waveform</li> <li>• <b>strict</b> - By default the wavelength argument is treated as a „maximum wavelength“, where the actual wavelength will be dynamically adjusted so that an exact number of triangular waves are created along the boundaries of the geometry. If the strict argument is set to true then the wavelength will be used exactly and an incomplete pattern may be used for the final waveform.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>triangular_wave(geom_from_wkt('LineString(0 0, 10 0)'), 3, 1)</code> → Triangular waves with wavelength 3 and amplitude 1 along the linestring</li> </ul>

## triangular\_wave\_randomized

Constructs randomized triangular waves along the boundary of a geometry.

Syntax	<code>triangular_wave_randomized(geometry, min_wavelength, max_wavelength, min_amplitude, max_amplitude, [seed=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>min_wavelength</b> - minimum wavelength of waves</li> <li>• <b>max_wavelength</b> - maximum wavelength of waves</li> <li>• <b>min_amplitude</b> - minimum amplitude of waves</li> <li>• <b>max_amplitude</b> - maximum amplitude of waves</li> <li>• <b>seed</b> - specifies a random seed for generating waves. If the seed is 0, then a completely random set of waves will be generated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>triangular_wave_randomized(geom_from_wkt('LineString(0 0, 10 0)'), 2, 3, 0.1, 0.2)</code> → Randomly sized triangular waves with wavelengths between 2 and 3 and amplitudes between 0.1 and 0.2 along the linestring</li> </ul>

## union

Returns a geometry that represents the point set union of the geometries.

Syntax	<code>union(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt( union( make_point(4, 4), make_point(5, 5) ) )</code> → <code>,MULTIPOINT(4 4, 5 5)'</code></li> </ul>

## wave

Constructs rounded (sine-like) waves along the boundary of a geometry.

Syntax	<code>wave(geometry, wavelength, amplitude, [strict=False])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>wavelength</b> - wavelength of sine-like waveform</li> <li>• <b>amplitude</b> - amplitude of sine-like waveform</li> <li>• <b>strict</b> - By default the wavelength argument is treated as a „maximum wavelength“, where the actual wavelength will be dynamically adjusted so that an exact number of waves are created along the boundaries of the geometry. If the strict argument is set to true then the wavelength will be used exactly and an incomplete pattern may be used for the final waveform.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>wave(geom_from_wkt('LineString(0 0, 10 0)'), 3, 1)</code> → Sine-like waves with wavelength 3 and amplitude 1 along the linestring</li> </ul>

## wave\_randomized

Constructs randomized curved (sine-like) waves along the boundary of a geometry.

Syntax	<code>wave_randomized(geometry, min_wavelength, max_wavelength, min_amplitude, max_amplitude, [seed=0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> <li>• <b>min_wavelength</b> - minimum wavelength of waves</li> <li>• <b>max_wavelength</b> - maximum wavelength of waves</li> <li>• <b>min_amplitude</b> - minimum amplitude of waves</li> <li>• <b>max_amplitude</b> - maximum amplitude of waves</li> <li>• <b>seed</b> - specifies a random seed for generating waves. If the seed is 0, then a completely random set of waves will be generated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>wave_randomized(geom_from_wkt('LineString(0 0, 10 0)'), 2, 3, 0.1, 0.2)</code> → Randomly sized curved waves with wavelengths between 2 and 3 and amplitudes between 0.1 and 0.2 along the linestring</li> </ul>

## wedge\_buffer

Returns a wedge shaped buffer originating from a point geometry.

Syntax	<code>wedge_buffer(center, azimuth, width, outer_radius, [inner_radius=0.0])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>center</b> - center point (origin) of buffer. Must be a point geometry.</li> <li>• <b>azimuth</b> - angle (in degrees) for the middle of the wedge to point.</li> <li>• <b>width</b> - buffer width (in degrees). Note that the wedge will extend to half of the angular width either side of the azimuth direction.</li> <li>• <b>outer_radius</b> - outer radius for buffers</li> <li>• <b>inner_radius</b> - optional inner radius for buffers</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>wedge_buffer(center:=geom_from_wkt('POINT(1 2)'), azimuth:=90,width:=180,outer_radius:=1)</code> → A wedge shaped buffer centered on the point (1,2), facing to the East, with a width of 180 degrees and outer radius of 1.</li> </ul>

Further reading: [Create wedge buffers](#) algorithm

## within

Tests whether a geometry is within another. Returns TRUE if the geometry1 is completely within geometry2.

Syntax	<code>within(geometry1, geometry2)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry1</b> - a geometry</li> <li>• <b>geometry2</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>within( geom_from_wkt( 'POINT( 0.5 0.5)' ), geom_from_wkt( 'POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))' ) ) → TRUE</code></li> <li>• <code>within( geom_from_wkt( 'POINT( 5 5 )' ), geom_from_wkt( 'POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))' ) ) → FALSE</code></li> </ul>

Further reading: [overlay\\_within](#)

### \$x

Returns the x coordinate of the current point feature. If the feature is a multipoint feature, then the x-coordinate of the first point will be returned. **WARNING: This function is deprecated. It is recommended to use the replacement `x()` function with `@geometry` variable instead.**

Syntax	<code>\$x</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$x → 42</code></li> </ul>

Further reading: [x](#)

### x

Returns the x coordinate of a point geometry, or the x coordinate of the centroid for a non-point geometry.

Syntax	<code>x(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>x( geom_from_wkt( 'POINT(2 5)' ) ) → 2</code></li> <li>• <code>x( @geometry ) → x coordinate of the current feature's centroid</code></li> </ul>

### \$x\_at

Retrieves a x coordinate of the current feature's geometry. **WARNING: This function is deprecated. It is recommended to use the replacement `x_at` function with `@geometry` variable instead.**

Syntax	<code>\$x_at(vertex)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>vertex</b> - index of the vertex of the current geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$x_at(1) → 5</code></li> </ul>

Further reading: [x\\_at](#)

### x\_at

Retrieves a x coordinate of the geometry.

Syntax	x_at(geometry, vertex)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> <li>• <b>vertex</b> - index of the vertex of the geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>x_at( geom_from_wkt( 'POINT(4 5)' ), 0 ) → 4</code></li> </ul>

### x\_max

Returns the maximum x coordinate of a geometry. Calculations are in the spatial reference system of this geometry.

Syntax	x_max(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>x_max( geom_from_wkt( 'LINESTRING(2 5, 3 6, 4 8)' ) ) → 4</code></li> </ul>

### x\_min

Returns the minimum x coordinate of a geometry. Calculations are in the spatial reference system of this geometry.

Syntax	x_min(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>x_min( geom_from_wkt( 'LINESTRING(2 5, 3 6, 4 8)' ) ) → 2</code></li> </ul>

### \$y

Returns the y coordinate of the current point feature. If the feature is a multipoint feature, then the y-coordinate of the first point will be returned. **WARNING: This function is deprecated. It is recommended to use the replacement `y()` function with `@geometry` variable instead.**

Syntax	\$y
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$y → 42</code></li> </ul>

Further reading: [y](#)



## y

Returns the y coordinate of a point geometry, or the y coordinate of the centroid for a non-point geometry.

Syntax	y(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>y( geom_from_wkt( 'POINT(2 5)' ) ) → 5</code></li> <li>• <code>y( @geometry ) → y coordinate of the current feature's centroid</code></li> </ul>

## \$y\_at

Retrieves a y coordinate of the current feature's geometry. **WARNING: This function is deprecated. It is recommended to use the replacement y\_at function with @geometry variable instead.**

Syntax	\$y_at(vertex)
Arguments	<ul style="list-style-type: none"> <li>• <b>vertex</b> - index of the vertex of the current geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$y_at(1) → 2</code></li> </ul>

Further reading: [y\\_at](#)

## y\_at

Retrieves a y coordinate of the geometry.

Syntax	y_at(geometry, vertex)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> <li>• <b>vertex</b> - index of the vertex of the geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>y_at( geom_from_wkt( 'POINT(4 5)' ), 0 ) → 5</code></li> </ul>

## y\_max

Returns the maximum y coordinate of a geometry. Calculations are in the spatial reference system of this geometry.

Syntax	y_max(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>y_max( geom_from_wkt( 'LINESTRING(2 5, 3 6, 4 8)' ) ) → 8</code></li> </ul>

## y\_min

Returns the minimum y coordinate of a geometry. Calculations are in the spatial reference system of this geometry.

Syntax	y_min(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>y_min( geom_from_wkt( 'LINESTRING(2 5, 3 6, 4 8)' ) ) → 5</code></li> </ul>

## \$z

Returns the z value of the current point feature if it is 3D. If the feature is a multipoint feature, then the z value of the first point will be returned. **WARNING: This function is deprecated. It is recommended to use the replacement z() function with @geometry variable instead.**

Syntax	\$z
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$z → 123</code></li> </ul>

## z

Returns the z coordinate of a point geometry, or NULL if the geometry has no z value.

Syntax	z(geometry)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a point geometry</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>z( geom_from_wkt( 'POINTZ(2 5 7)' ) ) → 7</code></li> </ul>

## z\_at

Retrieves a z coordinate of the geometry, or NULL if the geometry has no z value.

Syntax	z_at(geometry, vertex)
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - geometry object</li> <li>• <b>vertex</b> - index of the vertex of the geometry (indices start at 0; negative values apply from the last index, starting at -1)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>z_at(geom_from_wkt('LineStringZ(0 0 0, 10 10 5, 10 10 0)'), 1) → 5</code></li> </ul>

## z\_max

Returns the maximum z coordinate of a geometry, or NULL if the geometry has no z value.

Syntax	<code>z_max(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry with z coordinate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>z_max( geom_from_wkt( 'POINT ( 0 0 1 )' ) ) → 1</code></li> <li>• <code>z_max( geom_from_wkt( 'MULTIPOINT ( 0 0 1 , 1 1 3 )' ) ) → 3</code></li> <li>• <code>z_max( make_line( make_point( 0,0,0 ), make_point( -1,-1,-2 ) ) ) → 0</code></li> <li>• <code>z_max( geom_from_wkt( 'LINESTRING( 0 0 0, 1 0 2, 1 1 -1 )' ) ) → 2</code></li> <li>• <code>z_max( geom_from_wkt( 'POINT ( 0 0 )' ) ) → NULL</code></li> </ul>

## z\_min

Returns the minimum z coordinate of a geometry, or NULL if the geometry has no z value.

Syntax	<code>z_min(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - a geometry with z coordinate</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>z_min( geom_from_wkt( 'POINT ( 0 0 1 )' ) ) → 1</code></li> <li>• <code>z_min( geom_from_wkt( 'MULTIPOINT ( 0 0 1 , 1 1 3 )' ) ) → 1</code></li> <li>• <code>z_min( make_line( make_point( 0,0,0 ), make_point( -1,-1,-2 ) ) ) → -2</code></li> <li>• <code>z_min( geom_from_wkt( 'LINESTRING( 0 0 0, 1 0 2, 1 1 -1 )' ) ) → -1</code></li> <li>• <code>z_min( geom_from_wkt( 'POINT ( 0 0 )' ) ) → NULL</code></li> </ul>

## 13.2.14 Layout Funktionen

Diese Gruppe enthält Funktionen zur Beeinflussung der Eigenschaften von Drucklayoutelementen

### item\_variables

Returns a map of variables from a layout item inside this print layout.

Syntax	<code>item_variables(id)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>id</b> - layout item ID</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_get( item_variables('Map 0'), 'map_scale' ) → scale of the item ,Map 0' in the current print layout</code></li> </ul>

Further reading: List of default *variables*

## map\_credits

Returns a list of credit (usage rights) strings for the layers shown in a layout, or specific layout map item.

Syntax	<code>map_credits([id], [include_layer_names=false], [layer_name_separator=: ,])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>id</b> - Map item ID. If not specified, the layers from all maps in the layout will be used.</li> <li>• <b>include_layer_names</b> - Set to true to include layer names before their credit strings</li> <li>• <b>layer_name_separator</b> - String to insert between layer names and their credit strings, if <code>include_layer_names</code> is true</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array_to_string( map_credits() )</code> → comma separated list of layer credits for all layers shown in all map items in the layout, e.g. 'CC-BY-NC, CC-BY-SA'</li> <li>• <code>array_to_string( map_credits( 'Main Map' ) )</code> → comma separated list of layer credits for layers shown in the 'Main Map' layout item, e.g. 'CC-BY-NC, CC-BY-SA'</li> <li>• <code>array_to_string( map_credits( 'Main Map', include_layer_names := true, layer_name_separator := ': ' ) )</code> → comma separated list of layer names and their credits for layers shown in the 'Main Map' layout item, e.g. 'Railway lines: CC-BY-NC, Basemap: CC-BY-SA'</li> </ul>

This function requires the *Access metadata properties* of the layers to have been filled.

### 13.2.15 Kartenlayer

This group contains a list of the available layers in the current project and, for each layer, their fields (stored in the dataset, virtual or auxiliary ones as well as from joins). The fields can be interacted the same way as mentioned in *Felder und Werte*, except that a double-click will add the name as a string (single quoted) to the expression instead of as a field reference given that they do not belong to the active layer. This offers a convenient way to write expressions referring to different layers, such as when performing *aggregates*, *attribute* or *spatial* queries.

It also provides some convenient functions to manipulate layers.

## decode\_uri

Takes a layer and decodes the uri of the underlying data provider. It depends on the dataprovider, which data is available.

Syntax	<code>decode_uri(layer, [part])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer for which the uri should be decoded.</li> <li>• <b>part</b> - The part of the uri to return. If unspecified, a map with all uri parts will be returned.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>decode_uri(@layer)</code> → <code>{,layerId: ,0', ,layerName: ,', ,path: ,/home/qgis/shapefile.shp'}</code></li> <li>• <code>decode_uri(@layer)</code> → <code>{,layerId: NULL, ,layerName: ,layer', ,path: ,/home/qgis/geopackage.gpkg'}</code></li> <li>• <code>decode_uri(@layer, 'path')</code> → <code>,C:\my_data\qgis\shape.shp'</code></li> </ul>

## layer\_property

Returns a matching layer property or metadata value.

Syntax	layer_property(layer, property)
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - a string, representing either a layer name or layer ID</li> <li>• <b>property</b> - a string corresponding to the property to return. Valid options are: <ul style="list-style-type: none"> <li>– name: layer name</li> <li>– id: layer ID</li> <li>– title: metadata title string</li> <li>– abstract: metadata abstract string</li> <li>– keywords: metadata keywords</li> <li>– data_url: metadata URL</li> <li>– attribution: metadata attribution string</li> <li>– attribution_url: metadata attribution URL</li> <li>– source: layer source</li> <li>– min_scale: minimum display scale for layer</li> <li>– max_scale: maximum display scale for layer</li> <li>– is_editable: if layer is in edit mode</li> <li>– crs: layer CRS</li> <li>– crs_definition: layer CRS full definition</li> <li>– crs_description: layer CRS description</li> <li>– crs_ellipsoid: acronym of the layer CRS ellipsoid</li> <li>– extent: layer extent (as a geometry object)</li> <li>– distance_units: layer distance units</li> <li>– type: layer type, e.g., Vector or Raster</li> <li>– storage_type: storage format (vector layers only)</li> <li>– geometry_type: geometry type, e.g., Point (vector layers only)</li> <li>– feature_count: approximate feature count for layer (vector layers only)</li> <li>– path: File path to the layer data source. Only available for file based layers.</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• layer_property('streets', 'title') → 'Basemap Streets'</li> <li>• layer_property('airports', 'feature_count') → 120</li> <li>• layer_property('landsat', 'crs') → 'EPSG:4326'</li> </ul>

Further reading: [vector](#), [raster](#) and [mesh](#) layer properties

## load\_layer

Loads a layer by source URI and provider name.

Syntax	load_layer(uri, provider)
Arguments	<ul style="list-style-type: none"> <li>• <b>uri</b> - layer source URI string</li> <li>• <b>provider</b> - layer data provider name</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• layer_property(load_layer('c:/data/roads.shp', 'ogr'), 'feature_count') → count of features from the c:/data/roads.shp vector layer</li> </ul>

## 13.2.16 Kartenfunktionen

This group contains functions to create or manipulate keys and values of map data structures (also known as dictionary objects, key-value pairs, or associative arrays). Unlike the *list data structure* where values order matters, the order of the key-value pairs in the map object is not relevant and values are identified by their keys.

### from\_json

Loads a JSON formatted string.

Syntax	from_json(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - JSON string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• from_json('{"qgis": "rocks"}') → { ,qgis': ,rocks' }</li> <li>• from_json('[1,2,3]') → [1,2,3]</li> </ul>

### hstore\_to\_map

Creates a map from a hstore-formatted string.

Syntax	hstore_to_map(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the input string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• hstore_to_map('qgis=&gt;rocks') → { ,qgis': ,rocks' }</li> </ul>

### map

Returns a map containing all the keys and values passed as pair of parameters.

Syntax	map(key1, value1, key2, value2, ...)
Arguments	<ul style="list-style-type: none"> <li>• <b>key</b> - a key (string)</li> <li>• <b>value</b> - a value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• map('1', 'one', '2', 'two') → { ,1': ,one', ,2': ,two' }</li> <li>• map('1', 'one', '2', 'two')[ '1' ] → ,one'</li> </ul>

### map\_akeys

Returns all the keys of a map as an array.

Syntax	<code>map_akeys(map)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_akeys (map ('1', 'one', '2', 'two')) → [ ,1', ,2' ]</code></li> </ul>

### map\_aval

Returns all the values of a map as an array.

Syntax	<code>map_aval(map)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_aval (map ('1', 'one', '2', 'two')) → [ ,one', ,two' ]</code></li> </ul>

### map\_concat

Returns a map containing all the entries of the given maps. If two maps contain the same key, the value of the second map is taken.

Syntax	<code>map_concat(map1, map2, ...)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_concat (map ('1', 'one', '2', 'overridden'), map ('2', 'two', '3', 'three')) → { ,1': ,one', ,2': ,two', ,3': ,three' }</code></li> </ul>

### map\_delete

Returns a map with the given key and its corresponding value deleted.

Syntax	<code>map_delete(map, key)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> <li>• <b>key</b> - the key to delete</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_delete (map ('1', 'one', '2', 'two'), '2') → { ,1': ,one' }</code></li> </ul>

## map\_exist

Returns TRUE if the given key exists in the map.

Syntax	map_exist(map, key)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> <li>• <b>key</b> - the key to lookup</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• map_exist (map ('1', 'one', '2', 'two'), '3') → FALSE</li> </ul>

## map\_get

Returns the value of a map, given its key. Returns NULL if the key does not exist.

Syntax	map_get(map, key)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> <li>• <b>key</b> - the key to lookup</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• map_get (map ('1', 'one', '2', 'two'), '2') → ,two‘</li> <li>• map_get ( item_variables ('Map 0'), 'map_scale') → scale of the item ,Map 0‘ (if it exists) in the current print layout</li> </ul>

**Hinweis:** You can also use the *index operator ([ ])* to get a value from a map.

## map\_insert

Returns a map with an added key/value. If the key already exists, its value is overridden.

Syntax	map_insert(map, key, value)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> <li>• <b>key</b> - the key to add</li> <li>• <b>value</b> - the value to add</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• map_insert (map ('1', 'one'), '3', 'three') → { ,1‘: ,one‘, ,3‘: ,three‘ }</li> <li>• map_insert (map ('1', 'one', '2', 'overridden'), '2', 'two') → { ,1‘: ,one‘, ,2‘: ,two‘ }</li> </ul>



### map\_prefix\_keys

Returns a map with all keys prefixed by a given string.

Syntax	map_prefix_keys(map, prefix)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map</li> <li>• <b>prefix</b> - a string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_prefix_keys(map('1', 'one', '2', 'two'), 'prefix-') → { ,prefix-1': ,one', ,prefix-2': ,two' }</code></li> </ul>

### map\_to\_hstore

Merge map elements into a hstore-formatted string.

Syntax	map_to_hstore(map)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - the input map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_to_hstore(map('qgis', 'rocks')) → „qgis“=&gt;“rocks“</code></li> </ul>

### map\_to\_html\_dl

Merge map elements into a HTML definition list string.

Syntax	map_to_html_dl(map)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - the input map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_to_html_dl(map('qgis', 'rocks')) → &lt;dl&gt;&lt;dt&gt;qgis&lt;/dt&gt;&lt;dd&gt;rocks&lt;/dd&gt;&lt;/dl&gt;</code></li> </ul>

### map\_to\_html\_table

Merge map elements into a HTML table string.

Syntax	map_to_html_table(map)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - the input map</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>map_to_html_table(map('qgis', 'rocks')) → &lt;table&gt;&lt;thead&gt;&lt;th&gt;qgis&lt;/th&gt;&lt;/thead&gt;&lt;tbody&gt;&lt;tr&gt;&lt;td&gt;rocks&lt;/td&gt;&lt;/tr&gt;&lt;/tbody&gt;&lt;/table&gt;</code></li> </ul>

### to\_json

Create a JSON formatted string from a map, array or other value.

Syntax	to_json(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - The input value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_json(map('qgis', 'rocks'))</code> → <code>{„qgis“:„rocks“}</code></li> <li>• <code>to_json(array(1, 2, 3))</code> → <code>[1,2,3]</code></li> </ul>

### url\_encode

Returns an URL encoded string from a map. Transforms all characters in their properly-encoded form producing a fully-compliant query string.

Note that the plus sign `,+‘` is not converted.

Syntax	url_encode(map)
Arguments	<ul style="list-style-type: none"> <li>• <b>map</b> - a map.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>url_encode(map('a&amp;b', 'a and plus b', 'a=b', 'a equals b'))</code> → <code>,a%26+b=a%20and%20plus%20b&amp;a%3Db=a%20equals%20b‘</code></li> </ul>

## 13.2.17 Mathematische Funktionen

Diese Gruppe enthält mathematische Funktionen (z.B. sqrt, sin und cos).

### abs

Returns the absolute value of a number.

Syntax	abs(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>abs(-2)</code> → <code>2</code></li> </ul>

**acos**

Returns the inverse cosine of a value in radians.

Syntax	acos(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - cosine of an angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>acos(0.5) → 1.0471975511966</code></li> </ul>

**asin**

Returns the inverse sine of a value in radians.

Syntax	asin(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - sine of an angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>asin(1.0) → 1.5707963267949</code></li> </ul>

**atan**

Returns the inverse tangent of a value in radians.

Syntax	atan(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - tan of an angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>atan(0.5) → 0.463647609000806</code></li> </ul>

**atan2**

Returns the inverse tangent of dy/dx by using the signs of the two arguments to determine the quadrant of the result.

Syntax	atan2(dy, dx)
Arguments	<ul style="list-style-type: none"> <li>• <b>dy</b> - y coordinate difference</li> <li>• <b>dx</b> - x coordinate difference</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>atan2(1.0, 1.732) → 0.523611477769969</code></li> </ul>

### ceil

Rounds a number upwards.

Syntax	ceil(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ceil(4.9) → 5</code></li> <li>• <code>ceil(-4.9) → -4</code></li> </ul>

### clamp

Restricts an input value to a specified range.

Syntax	clamp(minimum, input, maximum)
Arguments	<ul style="list-style-type: none"> <li>• <b>minimum</b> - the smallest value <i>input</i> is allowed to take.</li> <li>• <b>input</b> - a value which will be restricted to the range specified by <i>minimum</i> and <i>maximum</i></li> <li>• <b>maximum</b> - the largest value <i>input</i> is allowed to take</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>clamp(1, 5, 10) → 5</code> <i>input</i> is between 1 and 10 so is returned unchanged</li> <li>• <code>clamp(1, 0, 10) → 1</code> <i>input</i> is less than minimum value of 1, so function returns 1</li> <li>• <code>clamp(1, 11, 10) → 10</code> <i>input</i> is greater than maximum value of 10, so function returns 10</li> </ul>

### cos

Returns cosine of an angle.

Syntax	cos(angle)
Arguments	<ul style="list-style-type: none"> <li>• <b>angle</b> - angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>cos(1.571) → 0.000796326710733263</code></li> </ul>

### degrees

Converts from radians to degrees.

Syntax	degrees(radians)
Arguments	<ul style="list-style-type: none"> <li>• <b>radians</b> - numeric value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>degrees(3.14159) → 180</code></li> <li>• <code>degrees(1) → 57.2958</code></li> </ul>

**exp**

Returns exponential of an value.

Syntax	exp(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - number to return exponent of</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>exp(1.0) → 2.71828182845905</code></li> </ul>

**floor**

Rounds a number downwards.

Syntax	floor(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>floor(4.9) → 4</code></li> <li>• <code>floor(-4.9) → -5</code></li> </ul>

**ln**

Returns the natural logarithm of a value.

Syntax	ln(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - numeric value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ln(1) → 0</code></li> <li>• <code>ln(2.7182818284590452354) → 1</code></li> </ul>

**log**

Returns the value of the logarithm of the passed value and base.

Syntax	log(base, value)
Arguments	<ul style="list-style-type: none"> <li>• <b>base</b> - any positive number</li> <li>• <b>value</b> - any positive number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>log(2, 32) → 5</code></li> <li>• <code>log(0.5, 32) → -5</code></li> </ul>

### log10

Returns the value of the base 10 logarithm of the passed expression.

Syntax	log10(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - any positive number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>log10 (1) → 0</code></li> <li>• <code>log10 (100) → 2</code></li> </ul>

### max

Returns the largest value in a set of values.

Syntax	max(value1, value2, ...)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>max (2, 10.2, 5.5) → 10.2</code></li> <li>• <code>max (20.5, NULL, 6.2) → 20.5</code></li> </ul>

### min

Returns the smallest value in a set of values.

Syntax	min(value1, value2, ...)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>min (20.5, 10, 6.2) → 6.2</code></li> <li>• <code>min (2, -10.3, NULL) → -10.3</code></li> </ul>

### pi

Returns value of pi for calculations.

Syntax	pi()
Beispiele	<ul style="list-style-type: none"> <li>• <code>pi () → 3.14159265358979</code></li> </ul>

## radians

Converts from degrees to radians.

Syntax	<code>radians(degrees)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>degrees</b> - numeric value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>radians(180) → 3.14159</code></li> <li>• <code>radians(57.2958) → 1</code></li> </ul>

## rand

Returns a random integer within the range specified by the minimum and maximum argument (inclusive). If a seed is provided, the returned will always be the same, depending on the seed.

Syntax	<code>rand(min, max, [seed=NULL])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>min</b> - an integer representing the smallest possible random number desired</li> <li>• <b>max</b> - an integer representing the largest possible random number desired</li> <li>• <b>seed</b> - any value to use as seed</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>rand(1, 10) → 8</code></li> </ul>

## randf

Returns a random float within the range specified by the minimum and maximum argument (inclusive). If a seed is provided, the returned will always be the same, depending on the seed.

Syntax	<code>randf([min=0.0], [max=1.0], [seed=NULL])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>min</b> - an float representing the smallest possible random number desired</li> <li>• <b>max</b> - an float representing the largest possible random number desired</li> <li>• <b>seed</b> - any value to use as seed</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>randf(1, 10) → 4.59258286403147</code></li> </ul>

## round

Rounds a number to number of decimal places.

Syntax	round(value, [places=0]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - decimal number to be rounded</li> <li>• <b>places</b> - Optional integer representing number of places to round decimals to. Can be negative.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• round(1234.567, 2) → 1234.57</li> <li>• round(1234.567) → 1235</li> <li>• round(1234.567, -1) → 1230</li> </ul>

## scale\_exponential

Transforms a given value from an input domain to an output range using an exponential curve. This function can be used to ease values in or out of the specified output range.

Syntax	scale_exponential(value, domain_min, domain_max, range_min, range_max, exponent)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - A value in the input domain. The function will return a corresponding scaled value in the output range.</li> <li>• <b>domain_min</b> - Specifies the minimum value in the input domain, the smallest value the input value should take.</li> <li>• <b>domain_max</b> - Specifies the maximum value in the input domain, the largest value the input value should take.</li> <li>• <b>range_min</b> - Specifies the minimum value in the output range, the smallest value which should be output by the function.</li> <li>• <b>range_max</b> - Specifies the maximum value in the output range, the largest value which should be output by the function.</li> <li>• <b>exponent</b> - A positive value (greater than 0), which dictates the way input values are mapped to the output range. Large exponents will cause the output values to ,ease in‘, starting slowly before accelerating as the input values approach the domain maximum. Smaller exponents (less than 1) will cause output values to ,ease out‘, where the mapping starts quickly but slows as it approaches the domain maximum.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• scale_exp(5, 0, 10, 0, 100, 2) → 25 easing in, using an exponent of 2</li> <li>• scale_exp(3, 0, 10, 0, 100, 0.5) → 54.772 easing out, using an exponent of 0.5</li> </ul>



## scale\_linear

Transforms a given value from an input domain to an output range using linear interpolation.

Syntax	<code>scale_linear(value, domain_min, domain_max, range_min, range_max)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - A value in the input domain. The function will return a corresponding scaled value in the output range.</li> <li>• <b>domain_min</b> - Specifies the minimum value in the input domain, the smallest value the input value should take.</li> <li>• <b>domain_max</b> - Specifies the maximum value in the input domain, the largest value the input value should take.</li> <li>• <b>range_min</b> - Specifies the minimum value in the output range, the smallest value which should be output by the function.</li> <li>• <b>range_max</b> - Specifies the maximum value in the output range, the largest value which should be output by the function.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>scale_linear(5, 0, 10, 0, 100) → 50</code></li> <li>• <code>scale_linear(0.2, 0, 1, 0, 360) → 72</code> scaling a value between 0 and 1 to an angle between 0 and 360</li> <li>• <code>scale_linear(1500, 1000, 10000, 9, 20) → 9.6111111</code> scaling a population which varies between 1000 and 10000 to a font size between 9 and 20</li> </ul>

## scale\_polynomial

Transforms a given value from an input domain to an output range using a polynomial curve. This function can be used to ease values in or out of the specified output range.

Syntax	<code>scale_polynomial(value, domain_min, domain_max, range_min, range_max, exponent)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - A value in the input domain. The function will return a corresponding scaled value in the output range.</li> <li>• <b>domain_min</b> - Specifies the minimum value in the input domain, the smallest value the input value should take.</li> <li>• <b>domain_max</b> - Specifies the maximum value in the input domain, the largest value the input value should take.</li> <li>• <b>range_min</b> - Specifies the minimum value in the output range, the smallest value which should be output by the function.</li> <li>• <b>range_max</b> - Specifies the maximum value in the output range, the largest value which should be output by the function.</li> <li>• <b>exponent</b> - A positive value (greater than 0), which dictates the way input values are mapped to the output range. Large exponents will cause the output values to ,ease in‘, starting slowly before accelerating as the input values approach the domain maximum. Smaller exponents (less than 1) will cause output values to ,ease out‘, where the mapping starts quickly but slows as it approaches the domain maximum.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>scale_polynomial(5, 0, 10, 0, 100, 2) → 25</code> easing in, using an exponent of 2</li> <li>• <code>scale_polynomial(3, 0, 10, 0, 100, 0.5) → 54.772</code> easing out, using an exponent of 0.5</li> </ul>

### sin

Returns the sine of an angle.

Syntax	sin(angle)
Arguments	<ul style="list-style-type: none"> <li>• <b>angle</b> - angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>sin(1.571) → 0.999999682931835</code></li> </ul>

### sqrt

Returns square root of a value.

Syntax	sqrt(value)
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - a number</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>sqrt(9) → 3</code></li> </ul>

### tan

Returns the tangent of an angle.

Syntax	tan(angle)
Arguments	<ul style="list-style-type: none"> <li>• <b>angle</b> - angle in radians</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>tan(1.0) → 1.5574077246549</code></li> </ul>

## 13.2.18 Meshes Functions

This group contains functions which calculate or return mesh related values.

### \$face\_area

Returns the area of the current mesh face. The area calculated by this function respects both the current project's ellipsoid setting and area unit settings. For example, if an ellipsoid has been set for the project then the calculated area will be ellipsoidal, and if no ellipsoid is set then the calculated area will be planimetric.

Syntax	\$face_area
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$face_area → 42</code></li> </ul>

### \$face\_index

Returns the index of the current mesh face.

Syntax	\$face_index
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$face_index</code> → 4581</li> </ul>

### \$vertex\_as\_point

Returns the current vertex as a point geometry.

Syntax	\$vertex_as_point
Beispiele	<ul style="list-style-type: none"> <li>• <code>geom_to_wkt ( \$vertex_as_point )</code> → ,POINT(800 1500 41)‘</li> </ul>

### \$vertex\_index

Returns the index of the current mesh vertex.

Syntax	\$vertex_index
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$vertex_index</code> → 9874</li> </ul>

### \$vertex\_x

Returns the X coordinate of the current mesh vertex.

Syntax	\$vertex_x
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$vertex_x</code> → 42.12</li> </ul>

### \$vertex\_y

Returns the Y coordinate of the current mesh vertex.

Syntax	\$vertex_y
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$vertex_y</code> → 12.24</li> </ul>

## \$vertex\_z

Returns the Z value of the current mesh vertex.

Syntax	\$vertex_z
Beispiele	<ul style="list-style-type: none"> <li>• \$vertex_z → 42</li> </ul>

## 13.2.19 Operatoren

Diese Gruppe enthält Operatoren (z. B. ., +, -, \*). Beachten Sie, dass für die meisten mathematischen Funktionen unten gilt, ist eine Eingabe NULL, dann ist das Ergebnis auch NULL.

### %

Remainder of division. Takes the sign of the dividend.

Syntax	a % b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 9 % 2 → 1</li> <li>• 9 % -2 → 1</li> <li>• -9 % 2 → -1</li> <li>• 5 % NULL → NULL</li> </ul>

### \*

Multiplication of two values

Syntax	a * b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 5 * 4 → 20</li> <li>• 5 * NULL → NULL</li> </ul>

+

Addition of two values. If one of the values is NULL the result will be NULL.

Syntax	a + b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 5 + 4 → 9</li> <li>• 5 + NULL → NULL</li> <li>• 'QGIS ' + 'ROCKS' → 'QGIS ROCKS'</li> <li>• to_datetime('2020-08-01 12:00:00') + '1 day 2 hours' → 2020-08-02T14:00:00</li> </ul>

Further reading: [concat](#), [||](#)

-

Subtraction of two values. If one of the values is NULL the result will be NULL.

Syntax	a - b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 5 - 4 → 1</li> <li>• 5 - NULL → NULL</li> <li>• to_datetime('2012-05-05 12:00:00') - to_interval('1 day 2 hours') → 2012-05-04T10:00:00</li> </ul>

/

Division of two values

Syntax	a / b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 5 / 4 → 1.25</li> <li>• 5 / NULL → NULL</li> </ul>

//

Floor division of two values

Syntax	<code>a // b</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>9 // 2 → 4</code></li> </ul>

<

Compares two values and evaluates to 1 if the left value is less than the right value.

Syntax	<code>a &lt; b</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>5 &lt; 4 → FALSE</code></li> <li>• <code>5 &lt; 5 → FALSE</code></li> <li>• <code>4 &lt; 5 → TRUE</code></li> </ul>

<=

Compares two values and evaluates to 1 if the left value is less or equal than the right value.

Syntax	<code>a &lt;= b</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>5 &lt;= 4 → FALSE</code></li> <li>• <code>5 &lt;= 5 → TRUE</code></li> <li>• <code>4 &lt;= 5 → TRUE</code></li> </ul>

<>

Compares two values and evaluates to 1 if they are not equal.

Syntax	$a \lt \gt b$
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <math>5 \lt \gt 4 \rightarrow \text{TRUE}</math></li> <li>• <math>4 \lt \gt 4 \rightarrow \text{FALSE}</math></li> <li>• <math>5 \lt \gt \text{NULL} \rightarrow \text{NULL}</math></li> <li>• <math>\text{NULL} \lt \gt \text{NULL} \rightarrow \text{NULL}</math></li> </ul>

**=**

Compares two values and evaluates to 1 if they are equal.

Syntax	$a = b$
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <math>5 = 4 \rightarrow \text{FALSE}</math></li> <li>• <math>4 = 4 \rightarrow \text{TRUE}</math></li> <li>• <math>5 = \text{NULL} \rightarrow \text{NULL}</math></li> <li>• <math>\text{NULL} = \text{NULL} \rightarrow \text{NULL}</math></li> </ul>

**>**

Compares two values and evaluates to 1 if the left value is greater than the right value.

Syntax	$a > b$
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <math>5 &gt; 4 \rightarrow \text{TRUE}</math></li> <li>• <math>5 &gt; 5 \rightarrow \text{FALSE}</math></li> <li>• <math>4 &gt; 5 \rightarrow \text{FALSE}</math></li> </ul>

**>=**

Compares two values and evaluates to 1 if the left value is greater or equal than the right value.

Syntax	<code>a &gt;= b</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>5 &gt;= 4 → TRUE</code></li> <li>• <code>5 &gt;= 5 → TRUE</code></li> <li>• <code>4 &gt;= 5 → FALSE</code></li> </ul>

### AND

Returns TRUE when conditions a and b are true.

Syntax	<code>a AND b</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - condition</li> <li>• <b>b</b> - condition</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>TRUE AND TRUE → TRUE</code></li> <li>• <code>TRUE AND FALSE → FALSE</code></li> <li>• <code>4 = 2+2 AND 1 = 1 → TRUE</code></li> <li>• <code>4 = 2+2 AND 1 = 2 → FALSE</code></li> </ul>

### BETWEEN

Returns TRUE if value is within the specified range. The range is considered inclusive of the bounds. To test for exclusion NOT BETWEEN can be used.

Syntax	<code>value BETWEEN lower_bound AND higher_bound</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - the value to compare with a range. It can be a string, a number or a date.</li> <li>• <b>lower_bound AND higher_bound</b> - range bounds</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>'B' BETWEEN 'A' AND 'C' → TRUE</code></li> <li>• <code>2 BETWEEN 1 AND 3 → TRUE</code></li> <li>• <code>2 BETWEEN 2 AND 3 → TRUE</code></li> <li>• <code>'B' BETWEEN 'a' AND 'c' → FALSE</code></li> <li>• <code>lower('B') BETWEEN 'a' AND 'b' → TRUE</code></li> </ul>

---

**Bemerkung:** `value BETWEEN lower_bound AND higher_bound` is the same as „`value >= lower_bound AND value <= higher_bound`“.

---

Further reading: [NOT BETWEEN](#)



## ILIKE

Returns TRUE if the first parameter matches case-insensitive the supplied pattern. LIKE can be used instead of ILIKE to make the match case-sensitive. Works with numbers also.

Syntax	string/number ILIKE pattern
Arguments	<ul style="list-style-type: none"> <li>• <b>string/number</b> - string to search</li> <li>• <b>pattern</b> - pattern to find, you can use <code>%</code> as a wildcard, <code>_</code> as a single char and <code>\</code> to escape these special characters.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'A' ILIKE 'A' → TRUE</li> <li>• 'A' ILIKE 'a' → TRUE</li> <li>• 'A' ILIKE 'B' → FALSE</li> <li>• 'ABC' ILIKE 'b' → FALSE</li> <li>• 'ABC' ILIKE 'B' → FALSE</li> <li>• 'ABC' ILIKE '_b_' → TRUE</li> <li>• 'ABC' ILIKE '_B_' → TRUE</li> <li>• 'ABCD' ILIKE '_b_' → FALSE</li> <li>• 'ABCD' ILIKE '_B_' → FALSE</li> <li>• 'ABCD' ILIKE '_b%' → TRUE</li> <li>• 'ABCD' ILIKE '_B%' → TRUE</li> <li>• 'ABCD' ILIKE '%b%' → TRUE</li> <li>• 'ABCD' ILIKE '%B%' → TRUE</li> <li>• 'ABCD%' ILIKE 'abcd\\%' → TRUE</li> <li>• 'ABCD' ILIKE '%B\\%' → FALSE</li> </ul>

## IN

Returns TRUE if value is found within a list of values.

Syntax	a IN b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - list of values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'A' IN ('A', 'B') → TRUE</li> <li>• 'A' IN ('C', 'B') → FALSE</li> </ul>

## IS

Returns TRUE if a is the same as b.

Syntax	a IS b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - any value</li> <li>• <b>b</b> - any value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'A' IS 'A' → TRUE</li> <li>• 'A' IS 'a' → FALSE</li> <li>• 4 IS 4 → TRUE</li> <li>• 4 IS 2+2 → TRUE</li> <li>• 4 IS 2 → FALSE</li> <li>• @geometry IS NULL → 0, if your geometry is not NULL</li> </ul>

## IS NOT

Returns TRUE if a is not the same as b.

Syntax	a IS NOT b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'a' IS NOT 'b' → TRUE</li> <li>• 'a' IS NOT 'a' → FALSE</li> <li>• 4 IS NOT 2+2 → FALSE</li> </ul>

## LIKE

Returns TRUE if the first parameter matches the supplied pattern. Works with numbers also.

Syntax	string/number LIKE pattern
Arguments	<ul style="list-style-type: none"> <li>• <b>string/number</b> - value</li> <li>• <b>pattern</b> - pattern to compare value with, you can use ,%' as a wildcard, ,_ as a single char and ,\ to escape these special characters.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'A' LIKE 'A' → TRUE</li> <li>• 'A' LIKE 'a' → FALSE</li> <li>• 'A' LIKE 'B' → FALSE</li> <li>• 'ABC' LIKE 'B' → FALSE</li> <li>• 'ABC' LIKE '_B_' → TRUE</li> <li>• 'ABCD' LIKE '_B_' → FALSE</li> <li>• 'ABCD' LIKE '_B%' → TRUE</li> <li>• 'ABCD' LIKE '%B%' → TRUE</li> <li>• '1%' LIKE '1\\%' → TRUE</li> <li>• '1_' LIKE '1\\%' → FALSE</li> </ul>

## NOT

Negates a condition.

Syntax	NOT a
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - condition</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• NOT 1 → FALSE</li> <li>• NOT 0 → TRUE</li> </ul>

## NOT BETWEEN

Returns TRUE if value is not within the specified range. The range is considered inclusive of the bounds.

Syntax	value NOT BETWEEN lower_bound AND higher_bound
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - the value to compare with a range. It can be a string, a number or a date.</li> <li>• <b>lower_bound AND higher_bound</b> - range bounds</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'B' NOT BETWEEN 'A' AND 'C' → FALSE</li> <li>• 1.0 NOT BETWEEN 1.1 AND 1.2 → TRUE</li> <li>• 2 NOT BETWEEN 2 AND 3 → FALSE</li> <li>• 'B' NOT BETWEEN 'a' AND 'c' → TRUE</li> <li>• lower('B') NOT BETWEEN 'a' AND 'b' → FALSE</li> </ul>

**Bemerkung:** *value NOT BETWEEN lower\_bound AND higher\_bound* is the same as „*value < lower\_bound OR value > higher\_bound*“.

Further reading: [BETWEEN](#)

## OR

Returns TRUE when condition a or b is true.

Syntax	a OR b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - condition</li> <li>• <b>b</b> - condition</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 4 = 2+2 OR 1 = 1 → TRUE</li> <li>• 4 = 2+2 OR 1 = 2 → TRUE</li> <li>• 4 = 2 OR 1 = 2 → FALSE</li> </ul>

## []

Index operator. Returns an element from an array or map value.

Syntax	[index]
Arguments	<ul style="list-style-type: none"> <li>• <b>index</b> - array index or map key value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>array(1,2,3)[0] → 1</code></li> <li>• <code>array(1,2,3)[2] → 3</code></li> <li>• <code>array(1,2,3)[-1] → 3</code></li> <li>• <code>map('a',1,'b',2)['a'] → 1</code></li> <li>• <code>map('a',1,'b',2)['b'] → 2</code></li> </ul>

Further reading: [array\\_get](#), [map\\_get](#)

## ^

Power of two values.

Syntax	a ^ b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>5 ^ 4 → 625</code></li> <li>• <code>5 ^ NULL → NULL</code></li> </ul>

## ||

Joins two values together into a string.

If one of the values is NULL the result will be NULL. See the CONCAT function for a different behavior.

Syntax	a    b
Arguments	<ul style="list-style-type: none"> <li>• <b>a</b> - value</li> <li>• <b>b</b> - value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>'Here'    ' and '    'there' → ,Here and there‘</code></li> <li>• <code>'Nothing'    NULL → NULL</code></li> <li>• <code>'Dia: '    "Diameter" → ,Dia: 25‘</code></li> <li>• <code>1    2 → ,12‘</code></li> </ul>

Further reading: [concat](#), [+](#)

~

Performs a regular expression match on a string value. Backslash characters must be double escaped (e.g., „\\s“ to match a white space character).

Syntax	string ~ regex
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - A string value</li> <li>• <b>regex</b> - A regular expression. Slashes must be escaped, eg \\d.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• 'hello' ~ 'll' → TRUE</li> <li>• 'hello' ~ '^ll' → FALSE</li> <li>• 'hello' ~ 'llo\$' → TRUE</li> <li>• 'abc123' ~ '\\d+' → TRUE</li> </ul>

Further reading: [regex\\_match](#)

### 13.2.20 Processing Functions

This group contains functions that operate on processing algorithms.

- [parameter](#)

#### parameter

Returns the value of a processing algorithm input parameter.

Syntax	parameter(name)
Arguments	<ul style="list-style-type: none"> <li>• <b>name</b> - name of the corresponding input parameter</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>parameter('BUFFER_SIZE') → 5.6</code></li> </ul>

### 13.2.21 Rasterfunktionen

Diese Gruppe enthält Funktionen zur Verarbeitung von Rasterlayern.

- [raster\\_attributes](#)
- [raster\\_statistic](#)
- [raster\\_value](#)

## raster\_attributes

Returns a map with the fields names as keys and the raster attribute table values as values from the attribute table entry that matches the given raster value.

Syntax	<code>raster_attributes(layer, band, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the name or id of a raster layer</li> <li>• <b>band</b> - the band number for the associated attribute table lookup.</li> <li>• <b>value</b> - raster value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>raster_attributes('vegetation', 1, raster_value('vegetation', 1, make_point(1,1)))</code> → <code>{,class':,Vegetated',,subclass':,Trees'}</code></li> </ul>

## raster\_statistic

Returns statistics from a raster layer.

Syntax	<code>raster_statistic(layer, band, property)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - a string, representing either a raster layer name or layer ID</li> <li>• <b>band</b> - integer representing the band number from the raster layer, starting at 1</li> <li>• <b>property</b> - a string corresponding to the property to return. Valid options are: <ul style="list-style-type: none"> <li>– min: minimum value</li> <li>– max: maximum value</li> <li>– avg: average (mean) value</li> <li>– stdev: standard deviation of values</li> <li>– range: range of values (max - min)</li> <li>– sum: sum of all values from raster</li> </ul> </li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>raster_statistic('lc', 1, 'avg')</code> → Average value from band 1 from 'lc' raster layer</li> <li>• <code>raster_statistic('ac2010', 3, 'min')</code> → Minimum value from band 3 from 'ac2010' raster layer</li> </ul>

## raster\_value

Returns the raster value found at the provided point.

Syntax	<code>raster_value(layer, band, point)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - the name or id of a raster layer</li> <li>• <b>band</b> - the band number to sample the value from.</li> <li>• <b>point</b> - point geometry (for multipart geometries having more than one part, a NULL value will be returned)</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>raster_value('dem', 1, make_point(1,1))</code> → 25</li> </ul>

### 13.2.22 Funktionen zu Datensätzen und Attributen

Diese Gruppe enthält Funktionen die sich auf datensatzbezeichner beziehen.

#### attribute

Returns an attribute from a feature.

##### Variant 1

Returns the value of an attribute from the current feature.

Syntax	attribute(attribute_name)
Arguments	<ul style="list-style-type: none"> <li>• <b>attribute_name</b> - name of attribute to be returned</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>attribute( 'name' )</code> → value stored in ,name‘ attribute for the current feature</li> </ul>

##### Variant 2

Allows the target feature and attribute name to be specified.

Syntax	attribute(feature, attribute_name)
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - a feature</li> <li>• <b>attribute_name</b> - name of attribute to be returned</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>attribute( @atlas_feature, 'name' )</code> → value stored in ,name‘ attribute for the current atlas feature</li> </ul>

#### attributes

Returns a map containing all attributes from a feature, with field names as map keys.

##### Variant 1

Returns a map of all attributes from the current feature.

Syntax	attributes()
Beispiele	<ul style="list-style-type: none"> <li>• <code>attributes() [ 'name' ]</code> → value stored in ,name‘ attribute for the current feature</li> </ul>

##### Variant 2

Allows the target feature to be specified.

Syntax	attributes(feature)
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - a feature</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>attributes( @atlas_feature ) [ 'name' ]</code> → value stored in ,name‘ attribute for the current atlas feature</li> </ul>

Further reading: *Kartenfunktionen*

### \$currentfeature

Returns the current feature being evaluated. This can be used with the ‚attribute‘ function to evaluate attribute values from the current feature. **WARNING: This function is deprecated. It is recommended to use the replacement @feature variable instead.**

Syntax	\$currentfeature
Beispiele	<ul style="list-style-type: none"><li>• <code>attribute( \$currentfeature, 'name' )</code> → value stored in ‚name‘ attribute for the current feature</li></ul>

### display\_expression

Returns the display expression for a given feature in a layer. The expression is evaluated by default. Can be used with zero, one or more arguments, see below for details.

#### No parameters

If called with no parameters, the function will evaluate the display expression of the current feature in the current layer.

Syntax	display_expression()
Beispiele	<ul style="list-style-type: none"><li>• <code>display_expression()</code> → The display expression of the current feature in the current layer.</li></ul>

#### One ‚feature‘ parameter

If called with a ‚feature‘ parameter only, the function will evaluate the specified feature from the current layer.

Syntax	display_expression(feature)
Arguments	<ul style="list-style-type: none"><li>• <b>feature</b> - The feature which should be evaluated.</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>display_expression(@atlas_feature)</code> → The display expression of the current atlas feature.</li></ul>

#### Layer and feature parameters

If the function is called with both a layer and a feature, it will evaluate the specified feature from the specified layer.



Syntax	<code>display_expression(layer, feature, [evaluate=true])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer (or its ID or name)</li> <li>• <b>feature</b> - The feature which should be evaluated.</li> <li>• <b>evaluate</b> - If the expression must be evaluated. If false, the expression will be returned as a string literal only (which could potentially be later evaluated using the <code>eval</code> function).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>display_expression( 'streets', get_feature_by_id('streets', 1) )</code> → The display expression of the feature with the ID 1 on the layer <code>streets</code>.</li> <li>• <code>display_expression('a_layer_id', @feature, 'False')</code> → The display expression of the given feature not evaluated.</li> </ul>

## feature\_id

Returns a feature's unique ID, or NULL if the feature is not valid.

Syntax	<code>feature_id(feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - a feature object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>feature_id( @feature )</code> → the ID of the current feature</li> </ul>

Further reading: [get\\_feature\\_by\\_id](#)

## get\_feature

Returns the first feature of a layer matching a given attribute value.

### Single value variant

Along with the layer ID, a single column and value are specified.

Syntax	<code>get_feature(layer, attribute, value)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - layer name or ID</li> <li>• <b>attribute</b> - attribute name to use for the match</li> <li>• <b>value</b> - attribute value to match</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>get_feature('streets', 'name', 'main st')</code> → first feature found in „streets“ layer with „main st“ value in the „name“ field</li> </ul>

### Map variant

Along with the layer ID, a map containing the columns (key) and their respective value to be used.

Syntax	<code>get_feature(layer, attribute)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - layer name or ID</li> <li>• <b>attribute</b> - Map containing the column and value pairs to use</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>get_feature('streets', map('name', 'main st', 'lane_num', '4'))</code> → first feature found in „streets“ layer with „main st“ value in the „name“ field and „4“ value in the „lane_num“ field</li> </ul>

### `get_feature_by_id`

Returns the feature with an id on a layer.

Syntax	<code>get_feature_by_id(layer, feature_id)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - layer, layer name or layer id</li> <li>• <b>feature_id</b> - the id of the feature which should be returned</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>get_feature_by_id('streets', 1)</code> → the feature with the id 1 on the layer „streets“</li> </ul>

Further reading: [\*feature\\_id\*](#)

### `$id`

Returns the feature id of the current row. **WARNING: This function is deprecated. It is recommended to use the replacement `@id` variable instead.**

Syntax	<code>\$id</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>\$id</code> → 42</li> </ul>

Further reading: [\*feature\\_id\*](#), [\*get\\_feature\\_by\\_id\*](#)

### `is_attribute_valid`

Returns TRUE if a specific feature attribute meets all constraints.

Syntax	<code>is_attribute_valid(attribute, [feature], [layer], [strength])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>attribute</b> - an attribute name</li> <li>• <b>feature</b> - A feature. If not set, the current feature will be used.</li> <li>• <b>layer</b> - A vector layer. If not set, the current layer will be used.</li> <li>• <b>strength</b> - Set to 'hard' or 'soft' to narrow down to a specific constraint type. If not set, the function will return FALSE if either a hard or a soft constraint fails.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_attribute_valid('HECTARES')</code> → TRUE if the current feature's value in the „HECTARES“ field meets all constraints.</li> <li>• <code>is_attribute_valid('HOUSES', get_feature('my_layer', 'FID', 10), 'my_layer')</code> → FALSE if the value in the „HOUSES“ field from the feature with „FID“=10 in 'my_layer' fails to meet all constraints.</li> </ul>

Further reading: *Beschränkungen*

### is\_feature\_valid

Returns TRUE if a feature meets all field constraints.

Syntax	<code>is_feature_valid([feature], [layer], [strength])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - A feature. If not set, the current feature will be used.</li> <li>• <b>layer</b> - A vector layer. If not set, the current layer will be used.</li> <li>• <b>strength</b> - Set to 'hard' or 'soft' to narrow down to a specific constraint type. If not set, the function will return FALSE if either a hard or a soft constraint fails.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_feature_valid(strength:='hard')</code> → TRUE if all fields from the current feature meet their hard constraints.</li> <li>• <code>is_feature_valid(get_feature('my_layer', 'FID', 10), 'my_layer')</code> → FALSE if all fields from feature with „FID“=10 in 'my_layer' fails to meet all constraints.</li> </ul>

Further reading: *Beschränkungen*

### is\_selected

Returns TRUE if a feature is selected. Can be used with zero, one or two arguments, see below for details.

#### No parameters

If called with no parameters, the function will return TRUE if the current feature in the current layer is selected.

Syntax	<code>is_selected()</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_selected()</code> → TRUE if the current feature in the current layer is selected.</li> </ul>

#### One 'feature' parameter

If called with a `,feature'` parameter only, the function returns TRUE if the specified feature from the current layer is selected.

Syntax	<code>is_selected(feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - The feature which should be checked for selection.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_selected(@atlas_feature)</code> → TRUE if the current atlas feature is selected.</li> <li>• <code>is_selected(get_feature('streets', 'name', 'Main St.'))</code> → TRUE if the unique named „Main St.“ feature on the active „streets“ layer is selected.</li> <li>• <code>is_selected(get_feature_by_id('streets', 1))</code> → TRUE if the feature with the id 1 on the active „streets“ layer is selected.</li> </ul>

### Two parameters

If the function is called with both a layer and a feature, it will return TRUE if the specified feature from the specified layer is selected.

Syntax	<code>is_selected(layer, feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer (its ID or name) on which the selection will be checked.</li> <li>• <b>feature</b> - The feature which should be checked for selection.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>is_selected('streets', get_feature('streets', 'name', 'street_name'))</code> → TRUE if the current building's street is selected (assuming the building layer has a field named <code>,street_name'</code> and the <code>,streets'</code> layer has a field called <code>,name'</code> with unique values).</li> <li>• <code>is_selected('streets', get_feature_by_id('streets', 1))</code> → TRUE if the feature with the id 1 on the „streets“ layer is selected.</li> </ul>

### maptip

Returns the maptip for a given feature in a layer. The expression is evaluated by default. Can be used with zero, one or more arguments, see below for details.

#### No parameters

If called with no parameters, the function will evaluate the maptip of the current feature in the current layer.

Syntax	<code>maptip()</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>maptip()</code> → The maptip of the current feature in the current layer.</li> </ul>

#### One `,feature'` parameter

If called with a `,feature'` parameter only, the function will evaluate the specified feature from the current layer.

Syntax	<code>maptip(feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - The feature which should be evaluated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>maptip(@atlas_feature)</code> → The maptip of the current atlas feature.</li> </ul>

### Layer and feature parameters

If the function is called with both a layer and a feature, it will evaluate the specified feature from the specified layer.

Syntax	<code>maptip(layer, feature, [evaluate=true])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer (or its ID or name)</li> <li>• <b>feature</b> - The feature which should be evaluated.</li> <li>• <b>evaluate</b> - If the expression must be evaluated. If false, the expression will be returned as a string literal only (which could potentially be later evaluated using the <code>eval_template</code> function).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>maptip('streets', get_feature_by_id('streets', 1))</code> → The maptip of the feature with the ID 1 on the layer 'streets'.</li> <li>• <code>maptip('a_layer_id', @feature, 'False')</code> → The maptip of the given feature not evaluated.</li> </ul>

### num\_selected

Returns the number of selected features on a given layer. By default works on the layer on which the expression is evaluated.

Syntax	<code>num_selected([layer=current layer])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer (or its id or name) on which the selection will be checked.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>num_selected()</code> → The number of selected features on the current layer.</li> <li>• <code>num_selected('streets')</code> → The number of selected features on the layer streets</li> </ul>

### represent\_attributes

Returns a map with the attribute names as keys and the configured representation values as values. The representation value for the attributes depends on the configured widget type for each attribute. Can be used with zero, one or more arguments, see below for details.

#### No parameters

If called with no parameters, the function will return the representation of the attributes of the current feature in the current layer.

Syntax	<code>represent_attributes()</code>
Beispiele	<ul style="list-style-type: none"> <li>• <code>represent_attributes()</code> → The representation of the attributes for the current feature.</li> </ul>

#### One 'feature' parameter

If called with a 'feature' parameter only, the function will return the representation of the attributes of the specified feature from the current layer.

Syntax	<code>represent_attributes(feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>feature</b> - The feature which should be evaluated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>represent_attributes(@atlas_feature)</code> → The representation of the attributes for the specified feature from the current layer.</li> </ul>

### Layer and feature parameters

If called with a `,layer‘` and a `,feature‘` parameter, the function will return the representation of the attributes of the specified feature from the specified layer.

Syntax	<code>represent_attributes(layer, feature)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>layer</b> - The layer (or its ID or name).</li> <li>• <b>feature</b> - The feature which should be evaluated.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>represent_attributes('atlas_layer', @atlas_feature)</code> → The representation of the attributes for the specified feature from the specified layer.</li> </ul>

Further reading: [represent\\_value](#)

### represent\_value

Returns the configured representation value for a field value. It depends on the configured widget type. Often, this is useful for `,Value Map‘` widgets.

Syntax	<code>represent_value(value, [fieldName])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>value</b> - The value which should be resolved. Most likely a field.</li> <li>• <b>fieldName</b> - The field name for which the widget configuration should be loaded.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>represent_value("field_with_value_map")</code> → Description for value</li> <li>• <code>represent_value('static value', 'field_name')</code> → Description for static value</li> </ul>

Further reading: [widget types](#), [represent\\_attributes](#)

### sqlite\_fetch\_and\_increment

Manage autoincrementing values in sqlite databases.

SQLite default values can only be applied on insert and not prefetched.

This makes it impossible to acquire an incremented primary key via `AUTO_INCREMENT` before creating the row in the database. Sidenote: with postgres, this works via the option *evaluate default values*.

When adding new features with relations, it is really nice to be able to already add children for a parent, while the parents form is still open and hence the parent feature uncommitted.

To get around this limitation, this function can be used to manage sequence values in a separate table on sqlite based formats like gpkg.

The sequence table will be filtered for a sequence id (`filter_attribute` and `filter_value`) and the current value of the `id_field` will be incremented by 1 and the incremented value returned.

If additional columns require values to be specified, the `default_values` map can be used for this purpose.

#### Note

This function modifies the target sqlite table. It is intended for usage with default value configurations for attributes.

When the database parameter is a layer and the layer is in transaction mode, the value will only be retrieved once during the lifetime of a transaction and cached and incremented. This makes it unsafe to work on the same database from several processes in parallel.

Syntax	<code>sqlite_fetch_and_increment(database, table, id_field, filter_attribute, filter_value, [default_values])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>database</b> - Path to the sqlite file or geopackage layer</li> <li>• <b>table</b> - Name of the table that manages the sequences</li> <li>• <b>id_field</b> - Name of the field that contains the current value</li> <li>• <b>filter_attribute</b> - Name the field that contains a unique identifier for this sequence. Must have a UNIQUE index.</li> <li>• <b>filter_value</b> - Name of the sequence to use.</li> <li>• <b>default_values</b> - Map with default values for additional columns on the table. The values need to be fully quoted. Functions are allowed.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>sqlite_fetch_and_increment(@layer, 'sequence_table', 'last_unique_id', 'sequence_id', 'global', map('last_change', 'date('now')', 'user', ''    @user_account_name    '')) → 0</code></li> <li>• <code>sqlite_fetch_and_increment(layer_property(@layer, 'path'), 'sequence_table', 'last_unique_id', 'sequence_id', 'global', map('last_change', 'date('now')', 'user', ''    @user_account_name    '')) → 0</code></li> </ul>

Further reading: [Data Sources Properties](#), [Setting relations between multiple layers](#)

## uuid

Generates a Universally Unique Identifier (UUID) for each row using the Qt `QUuid::createUuid` method.

Syntax	<code>uuid([format='WithBraces'])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>format</b> - The format, as the UUID will be formatted. <code>'WithBraces'</code>, <code>'WithoutBraces'</code> or <code>'Id128'</code>.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>uuid() → ,{0bd2f60f-f157-4a6d-96af-d4ba4cb366a1}‘</code></li> <li>• <code>uuid('WithoutBraces') → ,0bd2f60f-f157-4a6d-96af-d4ba4cb366a1‘</code></li> <li>• <code>uuid('Id128') → ,0bd2f60ff1574a6d96afd4ba4cb366a1‘</code></li> </ul>

### 13.2.23 Relations

This group contains the list of the *relations* available in the current project, with their description. It provides a quick access to the relation ID for writing an expression (with e.g. the *relation\_aggregate* function) or customizing a form.

### 13.2.24 Sensors Functions

This group contains functions to interact with sensors.

#### sensor\_data

Returns the last captured value (or values as a map for sensors which report multiple values) from a registered sensor.

Syntax	sensor_data(name, [expiration]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"><li>• <b>name</b> - the sensor name</li><li>• <b>expiration</b> - maximum millisecond since last captured value allowed</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>sensor_data('geiger_1') → ,2000'</code></li></ul>

### 13.2.25 Zeichenkettenfunktionen

Diese Gruppe enthält Funktionen für Zeichenketten (z.B. Ersetzen und in Großbuchstaben umwandeln).

#### ascii

Returns the unicode code associated with the first character of a string.

Syntax	ascii(string)
Arguments	<ul style="list-style-type: none"><li>• <b>string</b> - the string to convert to unicode code</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>ascii('Q') → 81</code></li></ul>

#### char

Returns the character associated with a unicode code.

Syntax	char(code)
Arguments	<ul style="list-style-type: none"><li>• <b>code</b> - a unicode code number</li></ul>
Beispiele	<ul style="list-style-type: none"><li>• <code>char(81) → ,Q'</code></li></ul>



## concat

Concatenates several strings to one. NULL values are converted to empty strings. Other values (like numbers) are converted to strings.

Syntax	concat(string1, string2, ...)
Arguments	<ul style="list-style-type: none"> <li><b>string</b> - a string value</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>concat('sun', 'set') → ,sunset‘</li> <li>concat('a', 'b', 'c', 'd', 'e') → ,abcde‘</li> <li>concat('Anno ', 1984) → ,Anno 1984‘</li> <li>concat('The Wall', NULL) → ,The Wall‘</li> </ul>

## Verkettung von Feldern

You can also concatenate strings or field values using either || or + operators, with some special characteristics:

- The + operator also means sum up expression, so if you have an integer (field or numeric value) operand, this can be error prone and you better use the others:

```
'My feature id is: ' + "gid" => triggers an error as gid returns an integer
```

- When any of the arguments is a NULL value, either || or + will return a NULL value. To return the other arguments regardless the NULL value, you may want to use the concat function:

```
'My feature id is: ' + NULL ==> NULL
'My feature id is: ' || NULL => NULL
concat('My feature id is: ', NULL) => 'My feature id is: '
```

further reading: ||, +

## format

Format a string using supplied arguments.

Syntax	format(string, arg1, arg2, ...)
Arguments	<ul style="list-style-type: none"> <li><b>string</b> - A string with placeholders %1, %2, etc., for the arguments. Placeholders can be repeated. The lowest numbered placeholder is replaced by arg1, the next by arg2, etc.</li> <li><b>arg</b> - any type. Any number of arguments.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>format('This %1 a %2', 'is', 'test') → ,This is a test‘</li> <li>format('This is %2', 'a bit unexpected but 2 is lowest number in string', 'normal') → ,This is a bit unexpected but 2 is lowest number in string‘</li> </ul>

## format\_date

Formats a date type or string into a custom string format. Uses Qt date/time format strings. See [QDateTime::toString](#).

Syntax	<code>format_date(datetime, format, [language])</code> [] marks optional arguments
--------	---

### Arguments

- **datetime** - date, time or datetime value
- **format** - String template used to format the string.

Ausdruck	Output
d	the day as number without a leading zero (1 to 31)
dd	the day as number with a leading zero (01 to 31)
ddd	the abbreviated localized day name (e.g. ‚Mon‘ to ‚Sun‘)
dddd	the long localized day name (e.g. ‚Monday‘ to ‚Sunday‘)
M	the month as number without a leading zero (1-12)
MM	the month as number with a leading zero (01-12)
MMM	the abbreviated localized month name (e.g. ‚Jan‘ to ‚Dec‘)
MMMM	the long localized month name (e.g. ‚January‘ to ‚December‘)
yy	the year as two digit number (00-99)
yyyy	the year as four digit number

These expressions may be used for the time part of the format string:

Ausdruck	Output
h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)
hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)
H	the hour without a leading zero (0 to 23, even with AM/PM display)
HH	the hour with a leading zero (00 to 23, even with AM/PM display)
m	the minute without a leading zero (0 to 59)
mm	the minute with a leading zero (00 to 59)
s	the second without a leading zero (0 to 59)
ss	the second with a leading zero (00 to 59)
z	the milliseconds without trailing zeroes (0 to 999)
zzz	the milliseconds with trailing zeroes (000 to 999)
AP or A	interpret as an AM/PM time. <i>AP</i> must be either ‚AM‘ or ‚PM‘.
ap or a	Interpret as an AM/PM time. <i>ap</i> must be either ‚am‘ or ‚pm‘.

- **language** - language (lowercase, two- or three-letter, [ISO 639 language code](#)) used to format the date into a custom string. By default the current QGIS user locale is used.

### Beispiele

- `format_date('2012-05-15', 'dd.MM.yyyy')` → ‚15.05.2012‘
- `format_date('2012-05-15', 'd MMMM yyyy', 'fr')` → ‚15 mai 2012‘
- `format_date('2012-05-15', 'dddd')` → ‚Tuesday‘, if the current locale is an English variant
- `format_date('2012-05-15 13:54:20', 'dd.MM.yy')` → ‚15.05.12‘
- `format_date('13:54:20', 'hh:mm AP')` → ‚01:54 PM‘

## format\_number

Returns a number formatted with the locale separator for thousands. By default the current QGIS user locale is used. Also truncates the decimal places to the number of supplied places.

Syntax	<code>format_number(number, [places=0], [language], [omit_group_separators=false], [trim_trailing_zeroes=false])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>number</b> - number to be formatted</li> <li>• <b>places</b> - integer representing the number of decimal places to truncate the string to.</li> <li>• <b>language</b> - language (lowercase, two- or three-letter, <a href="#">ISO 639 language code</a>) used to format the number into a string. By default the current QGIS user locale is used.</li> <li>• <b>omit_group_separators</b> - if set to true then group separators will not be included in the string</li> <li>• <b>trim_trailing_zeroes</b> - if set to true then trailing zeros following the decimal point will be trimmed from the string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>format_number(10000000.332, 2)</code> → <code>,10,000,000.33</code> if e.g. the current locale is an English variant</li> <li>• <code>format_number(10000000.332, 2, 'fr')</code> → <code>,10 000 000,33</code></li> </ul>

## left

Returns a substring that contains the *n* leftmost characters of the string.

Syntax	<code>left(string, length)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - a string</li> <li>• <b>length</b> - integer. The number of characters from the left of the string to return.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>left('Hello World', 5)</code> → <code>,Hello'</code></li> </ul>

## length

Returns the number of characters in a string or the length of a geometry linestring.

### String variant

Returns the number of characters in a string.

Syntax	<code>length(string)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to count length of</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>length('hello')</code> → <code>5</code></li> </ul>

### Geometry variant

Calculate the length of a geometry line object. Calculations are always planimetric in the Spatial Reference System (SRS) of this geometry, and the units of the returned length will match the units for the SRS. This differs from the

calculations performed by the `$length` function, which will perform ellipsoidal calculations based on the project's ellipsoid and distance unit settings.

Syntax	<code>length(geometry)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>geometry</b> - line geometry object</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>length(geom_from_wkt('LINESTRING(0 0, 4 0)')) → 4.0</code></li> </ul>

### lower

Converts a string to lower case letters.

Syntax	<code>lower(string)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to convert to lower case</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>lower('HELLO World') → ,hello world'</code></li> </ul>

### lpad

Returns a string padded on the left to the specified width, using a fill character. If the target width is smaller than the string's length, the string is truncated.

Syntax	<code>lpad(string, width, fill)</code>
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to pad</li> <li>• <b>width</b> - length of new string</li> <li>• <b>fill</b> - character to pad the remaining space with</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>lpad('Hello', 10, 'x') → ,xxxxxHello'</code></li> <li>• <code>lpad('Hello', 3, 'x') → ,Hel'</code></li> </ul>

### ltrim

Removes the longest string containing only the specified characters (a space by default) from the start of string.

Syntax	<code>ltrim(string, [characters=' '])</code> [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to trim</li> <li>• <b>characters</b> - characters to trim</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>ltrim(' hello world ') → ,hello world ,</code></li> <li>• <code>ltrim('zzzytest', 'xyz') → ,test'</code></li> </ul>

Further reading: [rtrim](#), [trim](#)

## regexp\_match

Return the first matching position matching a regular expression within an unicode string, or 0 if the substring is not found.

Syntax	regexp_match(input_string, regex)
Arguments	<ul style="list-style-type: none"> <li>• <b>input_string</b> - the string to test against the regular expression</li> <li>• <b>regex</b> - The regular expression to test against. Backslash characters must be double escaped (e.g., „\\s“ to match a white space character or „\\b“ to match a word boundary).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>regexp_match('QGIS ROCKS', '\\sROCKS') → 5</code></li> <li>• <code>regexp_match('Budač', 'udač\\b') → 2</code></li> </ul>

## regexp\_replace

Returns a string with the supplied regular expression replaced.

Syntax	regexp_replace(input_string, regex, replacement)
Arguments	<ul style="list-style-type: none"> <li>• <b>input_string</b> - the string to replace matches in</li> <li>• <b>regex</b> - The regular expression to replace. Backslash characters must be double escaped (e.g., „\\s“ to match a white space character).</li> <li>• <b>replacement</b> - The string that will replace any matching occurrences of the supplied regular expression. Captured groups can be inserted into the replacement string using \\1, \\2, etc.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>regexp_replace('QGIS SHOULD ROCK', '\\sSHOULD\\s', ' DOES ') → ,QGIS DOES ROCK‘</code></li> <li>• <code>regexp_replace('ABC123', '\\d+', '') → ,ABC‘</code></li> <li>• <code>regexp_replace('my name is John', '(.*) is (.*)', '\\2 is \\1') → ,John is my name‘</code></li> </ul>

## regexp\_substr

Returns the portion of a string which matches a supplied regular expression.

Syntax	regexp_substr(input_string, regex)
Arguments	<ul style="list-style-type: none"> <li>• <b>input_string</b> - the string to find matches in</li> <li>• <b>regex</b> - The regular expression to match against. Backslash characters must be double escaped (e.g., „\\s“ to match a white space character).</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>regexp_substr('abc123', '(\\d+)') → ,123‘</code></li> </ul>

## replace

Returns a string with the supplied string, array, or map of strings replaced.

### String & array variant

Returns a string with the supplied string or array of strings replaced by a string or an array of strings.

Syntax	replace(string, before, after)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the input string</li> <li>• <b>before</b> - the string or array of strings to replace</li> <li>• <b>after</b> - the string or array of strings to use as a replacement</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>replace('QGIS SHOULD ROCK', 'SHOULD', 'DOES') → ,QGIS DOES ROCK‘</code></li> <li>• <code>replace('QGIS ABC', array('A', 'B', 'C'), array('X', 'Y', 'Z')) → ,QGIS XYZ‘</code></li> <li>• <code>replace('QGIS', array('Q', 'S'), '') → ,GI‘</code></li> </ul>

### Map variant

Returns a string with the supplied map keys replaced by paired values. Longer map keys are evaluated first.

Syntax	replace(string, map)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the input string</li> <li>• <b>map</b> - the map containing keys and values</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>replace('APP SHOULD ROCK', map('APP', 'QGIS', 'SHOULD', 'DOES')) → ,QGIS DOES ROCK‘</code></li> <li>• <code>replace('forty two', map('for', '4', 'two', '2', 'forty two', '42')) → ,42‘</code></li> </ul>

## right

Returns a substring that contains the *n* rightmost characters of the string.

Syntax	right(string, length)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - a string</li> <li>• <b>length</b> - integer. The number of characters from the right of the string to return.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>right('Hello World', 5) → ,World‘</code></li> </ul>

## rpadd

Returns a string padded on the right to the specified width, using a fill character. If the target width is smaller than the string's length, the string is truncated.

Syntax	rpadd(string, width, fill)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to pad</li> <li>• <b>width</b> - length of new string</li> <li>• <b>fill</b> - character to pad the remaining space with</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>rpadd('Hello', 10, 'x') → ,Helloxxxxx‘</code></li> <li>• <code>rpadd('Hello', 3, 'x') → ,Hel‘</code></li> </ul>

## rtrim

Removes the longest string containing only the specified characters (a space by default) from the end of string.

Syntax	rtrim(string, [characters=' ']) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to trim</li> <li>• <b>characters</b> - characters to trim</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>rtrim(' hello world ') → ,hello world‘</code></li> <li>• <code>rtrim('testxxxz', 'xyz') → ,test‘</code></li> </ul>

Further reading: [ltrim](#), [trim](#)

## strpos

Return the first matching position of a substring within another string, or 0 if the substring is not found.

Syntax	strpos(haystack, needle)
Arguments	<ul style="list-style-type: none"> <li>• <b>haystack</b> - string that is to be searched</li> <li>• <b>needle</b> - string to search for</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>strpos('HELLO WORLD', 'WORLD') → 7</code></li> <li>• <code>strpos('HELLO WORLD', 'GOODBYE') → 0</code></li> </ul>

### substr

Returns a part of a string.

Syntax	substr(string, start, [length]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the full input string</li> <li>• <b>start</b> - integer representing start position to extract beginning with 1; if start is negative, the return string will begin at the end of the string minus the start value</li> <li>• <b>length</b> - integer representing length of string to extract; if length is negative, the return string will omit the given length of characters from the end of the string</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>substr('HELLO WORLD', 3, 5) → ,LLO W'</code></li> <li>• <code>substr('HELLO WORLD', 6) → ,WORLD'</code></li> <li>• <code>substr('HELLO WORLD', -5) → ,WORLD'</code></li> <li>• <code>substr('HELLO', 3, -1) → ,LL'</code></li> <li>• <code>substr('HELLO WORLD', -5, 2) → ,WO'</code></li> <li>• <code>substr('HELLO WORLD', -5, -1) → ,WORL'</code></li> </ul>

### title

Converts all words of a string to title case (all words lower case with leading capital letter).

Syntax	title(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to convert to title case</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>title('hello wOrld') → ,Hello World'</code></li> </ul>

### to\_string

Converts a number to string.

Syntax	to_string(number)
Arguments	<ul style="list-style-type: none"> <li>• <b>number</b> - Integer or real value. The number to convert to string.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>to_string(123) → ,123'</code></li> </ul>



## trim

Removes all leading and trailing whitespace (spaces, tabs, etc) from a string.

Syntax	trim(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - string to trim</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>trim(' hello world ')</code> → <code>,hello world'</code></li> </ul>

Further reading: [ltrim](#), [rtrim](#)

## upper

Converts a string to upper case letters.

Syntax	upper(string)
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to convert to upper case</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>upper('hello World')</code> → <code>,HELLO WORLD'</code></li> </ul>

## wordwrap

Returns a string wrapped to a maximum/minimum number of characters.



Syntax	wordwrap(string, wrap_length, [delimiter_string]) [] marks optional arguments
Arguments	<ul style="list-style-type: none"> <li>• <b>string</b> - the string to be wrapped</li> <li>• <b>wrap_length</b> - an integer. If wrap_length is positive the number represents the ideal maximum number of characters to wrap; if negative, the number represents the minimum number of characters to wrap.</li> <li>• <b>delimiter_string</b> - Optional delimiter string to wrap to a new line.</li> </ul>
Beispiele	<ul style="list-style-type: none"> <li>• <code>wordwrap('UNIVERSITY OF QGIS', 13)</code> → <code>,UNIVERSITY OF&lt;br&gt;QGIS'</code></li> <li>• <code>wordwrap('UNIVERSITY OF QGIS', -3)</code> → <code>,UNIVERSITY&lt;br&gt;OF QGIS'</code></li> </ul>

## 13.2.26 User Expressions

This group contains the expressions saved as *user expressions*.

### 13.2.27 Variables

This group contains dynamic variables related to the application, the project file and other settings. The availability of variables depends on the context:

- von dem  **Objekte über Ausdruck wählen** Dialog
- von dem  **Feldrechner** Dialog
- von dem Layereigenschaften Dialog
- vom Drucklayout

To use these variables in an expression, they should be preceded by the @ character (e.g, @row\_number).

Variable	Beschreibung
algorithm_id	The unique ID of an algorithm
animation_end_time	End of the animation's overall temporal time range (as a datetime value)
animation_interval	Duration of the animation's overall temporal time range (as an interval value)
animation_start_time	Start of the animation's overall temporal time range (as a datetime value)
atlas_feature	The current atlas feature (as feature object)
atlas_featureid	The current atlas feature ID
atlas_featurenumber	The current atlas feature number in the layout
atlas_filename	The current atlas file name
atlas_geometry	The current atlas feature geometry
atlas_layerid	The current atlas coverage layer ID
atlas_layername	The current atlas coverage layer name
atlas_pagename	The current atlas page name
atlas_totalfeatures	The total number of features in atlas
canvas_cursor_point	The last cursor position on the canvas in the project's geographical coordinates
cluster_color	The color of symbols within a cluster, or NULL if symbols have mixed colors
cluster_size	The number of symbols contained within a cluster
current_feature	The feature currently being edited in the attribute form or table row
current_geometry	The geometry of the feature currently being edited in the form or the table row
current_parent_feature	represents the feature currently being edited in the parent form. Only usable in an embedded form context.
current_parent_geometry	represents the geometry of the feature currently being edited in the parent form. Only usable in an embedded form context.
form_mode	What the form is used for, like AddFeatureMode, SingleEditMode, MultiEditMode, SearchMode, AggregateSearchMode or IdentifyMode as string.
feature	The current feature being evaluated. This can be used with the ,attribute' function to evaluate attribute values from the current feature.
frame_duration	Temporal duration of each animation frame (as an interval value)
frame_number	Current frame number during animation playback
frame_rate	Number of frames per second during animation playback
fullextent_maxx	Maximum x value from full canvas extent (including all layers)
fullextent_maxy	Maximum y value from full canvas extent (including all layers)
fullextent_minx	Minimum x value from full canvas extent (including all layers)
fullextent_miny	Minimum y value from full canvas extent (including all layers)
geometry	The geometry of the current feature being evaluated
geometry_part_count	The number of parts in rendered feature's geometry
geometry_part_num	The current geometry part number for feature being rendered
geometry_point_count	The number of points in the rendered geometry's part
geometry_point_num	The current point number in the rendered geometry's part
geometry_ring_num	Current geometry ring number for feature being rendered (for polygon features only). The exterior ring has a value of 0.
grid_axis	The current grid annotation axis (eg, ,x' for longitude, ,y' for latitude)

Fortsetzung auf der nächsten Seite

Tab. 13.1 – Fortsetzung der vorherigen Seite

Variable	Beschreibung
grid_number	The current grid annotation value
id	The ID of the current feature being evaluated
item_id	The layout item user ID (not necessarily unique)
item_uuid	The layout item unique ID
layer	The current layer
layer_crs	The Coordinate Reference System Authority ID of the current layer
layer_crs_ellipsoid	The ellipsoid Authority ID of the current layer CRS
layer_cursor_point	Point geometry under the mouse position in map canvas, in active layer's CRS
layer_id	The ID of current layer
layer_ids	The IDs of all the map layers in the current project as a list
layer_name	The name of current layer
layers	All the map layers in the current project as a list
layout_dpi	The composition resolution (DPI)
layout_name	The layout name
layout_numpages	The number of pages in the layout
layout_page	The page number of the current item in the layout
layout_pageheight	The active page height in the layout (in mm for standard paper sizes, or whatever unit was used for custom paper size)
layout_pageoffsets	Array of Y coordinate of the top of each page. Allows to dynamically position items on pages in a context where page sizes may change
layout_pagewidth	The active page width in the layout (in mm for standard paper sizes, or whatever unit was used for custom paper size)
legend_column_count	The number of columns in the legend
legend_filter_by_map	Indicates if the content of the legend is filtered by the map
legend_filter_out_atlas	Indicates if the atlas is filtered out of the legend
legend_split_layers	Indicates if layers can be split in the legend
legend_title	The title of the legend
legend_wrap_string	The character(s) used to wrap the legend text
map_crs	The Coordinate reference system of the current map
map_crs_acronym	The acronym of the Coordinate reference system of the current map
map_crs_definition	The full definition of the Coordinate reference system of the current map
map_crs_description	The name of the Coordinate reference system of the current map
map_crs_ellipsoid	The acronym of the ellipsoid of the Coordinate reference system of the current map
map_crs_proj4	The Proj4 definition of the Coordinate reference system of the current map
map_crs_projection	The descriptive name of the projection method used by the Coordinate reference system of the map (e.g. 'Albers Equal Area')
map_crs_wkt	The WKT definition of the Coordinate reference system of the current map
map_end_time	The end of the map's temporal time range (as a datetime value)
map_extent	The geometry representing the current extent of the map
map_extent_center	The point feature at the center of the map
map_extent_height	The current height of the map
map_extent_width	The current width of the map
map_id	The ID of current map destination. This will be 'canvas' for canvas renders, and the item ID for layout map renders
map_interval	The duration of the map's temporal time range (as an interval value)
map_layer_ids	The list of map layer IDs visible in the map
map_layers	The list of map layers visible in the map
map_rotation	The current rotation of the map
map_scale	The current scale of the map
map_start_time	The start of the map's temporal time range (as a datetime value)
map_units	The units of map measurements
model_path	Full path (including file name) of current model (or project path if model is embedded in a project).
model_folder	Folder containing current model (or project folder if model is embedded in a project).

Fortsetzung auf der nächsten Seite

Tab. 13.1 – Fortsetzung der vorherigen Seite

Variable	Beschreibung
model_name	Name of current model
model_group	Group for current model
notification_message	Content of the notification message sent by the provider (available only for actions triggered by provider notifications).
parent	Refers to the current feature in the parent layer, providing access to its attributes and geometry when filtering an <i>aggregate</i> function
project_abstract	The project abstract, taken from project metadata
project_area_units	The area unit for the current project, used when calculating areas of geometries
project_author	The project author, taken from project metadata
project_basename	The basename of current project's filename (without path and extension)
project_creation_date	The project creation date, taken from project metadata
project_crs	The Coordinate reference system of the project
project_crs_arconym	The acronym of the Coordinate reference system of the project
project_crs_definition	The full definition of the Coordinate reference system of the project
project_crs_description	The description of the Coordinate reference system of the project
project_crs_ellipsoid	The ellipsoid of the Coordinate reference system of the project
project_crs_proj4	The Proj4 representation of the Coordinate reference system of the project
project_crs_wkt	The WKT (well known text) representation of the coordinate reference system of the project
project_distance_units	The distance unit for the current project, used when calculating lengths of geometries and distances
project_ellipsoid	The name of the ellipsoid of the current project, used when calculating geodetic areas or lengths of geometries
project_filename	The filename of the current project
project_folder	The folder of the current project
project_home	The home path of the current project
project_identifier	The project identifier, taken from the project's metadata
project_keywords	The project keywords, taken from the project's metadata
project_last_saved	Date/time when project was last saved.
project_path	The full path (including file name) of the current project
project_title	The title of current project
project_units	The units of the project's CRS
qgis_locale	The current language of QGIS
qgis_os_name	The current Operating system name, eg ,windows', ,linux' or ,osx'
qgis_platform	The QGIS platform, eg ,desktop' or ,server'
qgis_release_name	The current QGIS release name
qgis_short_version	The current QGIS version short string
qgis_version	The current QGIS version string
qgis_version_no	The current QGIS version number
row_number	Speichert die Zahl der aktuellen Reihe
snapping_results	Gibt Zugriff auf die Einrasthinweise beim Digitalisieren eines Objektes (nur beim Hinzufügen von Objekten verfügbar)
scale_value	The current scale bar distance value
selected_file_path	Selected file path from file widget selector when uploading a file with an external storage system
symbol_angle	The angle of the symbol used to render the feature (valid for marker symbols only)
symbol_color	The color of the symbol used to render the feature
symbol_count	The number of features represented by the symbol (in the layout legend)
symbol_frame	The frame number (for animated symbols only)
symbol_id	The Internal ID of the symbol (in the layout legend)
symbol_label	The label for the symbol (either a user defined label or the default autogenerated label - in the layout legend)
symbol_layer_count	Total number of symbol layers in the symbol
symbol_layer_index	Current symbol layer index
symbol_marker_column	Column number for marker (valid for point pattern fills only).

Fortsetzung auf der nächsten Seite

Tab. 13.1 – Fortsetzung der vorherigen Seite

Variable	Beschreibung
symbol_marker_row	Row number for marker (valid for point pattern fills only).
user_account_name	The current user's operating system account name
user_full_name	The current user's operating system user name
value	The current value
vector_tile_zoom	Exact vector tile zoom level of the map that is being rendered (derived from the current map scale). Normally in interval [0, 20]. Unlike @zoom_level, this variable is a floating point value which can be used to interpolate values between two integer zoom levels.
with_variable	Erlaubt das Setzen einer Variable für die Verwendung in einem Ausdruck und vermeidet so die wiederholte Neuberechnung desselben Wertes
zoom_level	Vector tile zoom level of the map that is being rendered (derived from the current map scale). Normally in interval [0, 20].

**Einige Beispiele:**

- Return the X coordinate of a map item center in layout:

```
x( map_get( item_variables( 'map1' ), 'map_extent_center' ) )
```

- Return, for each feature in the current layer, the number of overlapping airport features:

```
aggregate( layer:='airport', aggregate:='count', expression:="code",
           filter:=intersects( $geometry, geometry( @parent ) ) )
```

- Get the object\_id of the first snapped point of a line:

```
with_variable(
  'first_snapped_point',
  array_first( @snapping_results ),
  attribute(
    get_feature_by_id(
      map_get( @first_snapped_point, 'layer' ),
      map_get( @first_snapped_point, 'feature_id' )
    ),
    'object_id'
  )
)
```

**13.2.28 Kürzlich verwendete Funktionen**

This group contains recently used functions. Depending on the context of its usage (feature selection, field calculator, generic), recently applied expressions are added to the corresponding list (up to ten expressions), sorted from more to less recent. This makes it easy to quickly retrieve and reapply previously used expressions.






## 14.1 The Style Manager

### 14.1.1 The Style Manager dialog

The *Style Manager* is the place where you can manage and create generic style items. These are symbols, color ramps, text formats or label settings that can be used to symbolize features, layers or print layouts. They are stored in the `symbolology-style.db` database under the active *user profile* and shared with all the project files opened with that profile. Style items can also be shared with others thanks to the export/import capabilities of the *Style Manager* dialog.

You can open that modeless dialog either:

- from the *Settings* ►  *Style Manager...* menu
- with the  *Style Manager* button from the Project toolbar
- or with the  *Style Manager* button from a vector *Layer Properties* ► menu (while *configuring a symbol* or *formatting a text*).

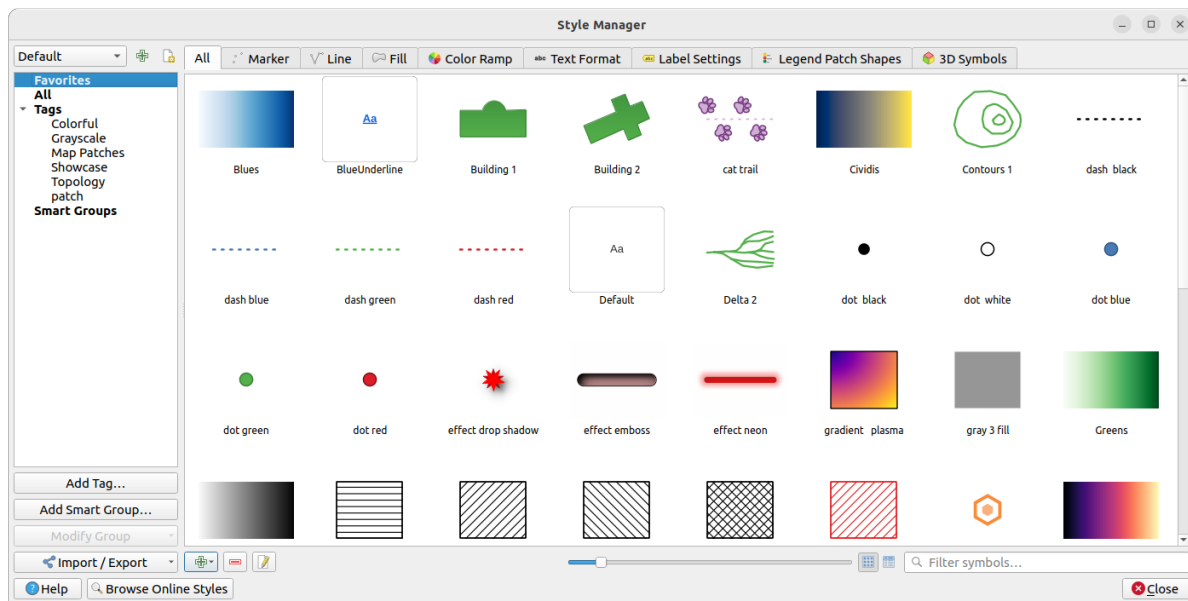












Abb. 14.1: The Style Manager

## Organizing style items

The *Style Manager* dialog displays in its center a frame with previewed items organized into tabs:


- *All* for a complete collection of point, linear and surface symbols and label settings as well as predefined color ramps and text formats;
-  *Marker* for point symbols only;
-  *Line* for linear symbols only;
-  *Fill* for surface symbols only;
-  *Color ramp*;
-  *Text format* to manage *text formats*, which store the font, color, buffers, shadows, and backgrounds of texts (i.e. all the formatting parts of the label settings, which for instance can be used in layouts);
-  *Label settings* to manage *label settings*, which include the text formats and some layer-type specific settings such as label placement, priority, callouts, rendering...
-  *Legend Patch Shapes* to manage custom legend patch shapes, which include *Marker*, *Line* and *Fill* geometries.
-  *3D Symbols* to configure symbols with *3D properties* (extrusion, shading, altitude, ...) for the features to render in a *3D Map view*

You can arrange the Styles in  *Icon View* or in  *List View* on the bottom right side. In both views the tooltip shows a larger instance of the style. The thumbnail size slider at the left of the icons helps you adjust the actual thumbnail sizes in the dialog, for a better preview of the symbols!

For each family of items, you can organize the elements into different categories, listed in the panel on the left:

- **Favorites:** displayed by default when configuring an item, it shows an extensible set of items;
- **All:** lists all the available items for the active type;



- **Tags:** shows a list of labels you can use to identify the items. An item can be tagged more than once. Select a tag in the list and the tabs are updated to show only their items that belong to it. To create a new tag you could later attach to a set of items, use the *Add Tag...* button or select the  *Add Tag...* from any tag contextual menu;
- **Smart Group:** a smart group dynamically fetches its symbols according to conditions set (see eg, [Abb. 14.2](#)). Click the *Add Smart Group...* button to create smart groups. The dialog box allows you to enter an expression to filter the items to select (has a particular tag, have a string in its name, etc.). Any symbol, color ramp, text format or label setting that satisfies the entered condition(s) is automatically added to the smart group.

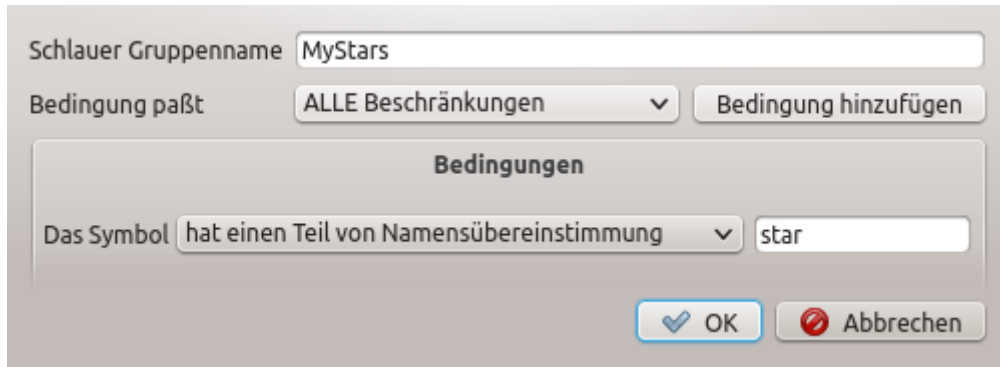



Abb. 14.2: Creating a Smart Group




Tags and smart groups are not mutually exclusive: they are simply two different ways to organize your style elements. Unlike the smart groups that automatically fetch their belonged items based on the input constraints, tags are filled by the user. To edit any of those categories, you can either:

- select the items, right-click and choose *Add to Tag* ► and then select the tag name or create a new tag;
- select the tag and press *Modify group...* ► *Attach Selected Tag to Symbols*. A checkbox appears next to each item to help you select or deselect it. When selection is finished, press *Modify group...* ► *Finish Tagging*.
- select the smart group, press *Modify group...* ► *Edit smart group...* and configure a new set of constraints in the *Smart Group Editor* dialog. This option is also available in the contextual menu of the smart group.

To remove a tag or a smart group, right-click on it and select the  *Remove* button. Note that this does not delete the items grouped in the category.

### Adding, editing or removing an item

As seen earlier, style elements are listed under different tabs whose contents depend on the active category (tag, smart group, favorites...). When a tab is enabled, you can:

- Add new items: press the  *Add item* button and configure the item following *symbols*, *color ramps* or *text format and label* builder description.
- Modify an existing item: select an item and press  *Edit item* button and configure as mentioned above.
- Delete existing items: to delete an element you no longer need, select it and click  *Remove item* (also available through right-click). The item will be deleted from the local database.


Note that the *All* tab provides access to these options for every type of item.

Right-clicking over a selection of items also allows you to:

- *Add to Favorites*;
- *Remove from Favorites*;



- *Add to Tag* ► and select the appropriate tag or create a new one to use; the currently assigned tags are checked;
- *Clear Tags*: detaching the symbols from any tag;
- *Remove Item(s)*;
- *Edit Item*: applies to the item you right-click over;
- *Copy Item*;
- *Paste Item ...*: pasting to one of the categories of the style manager or elsewhere in QGIS (symbol or color buttons)
- *Export Selected Symbol(s) as PNG...* (only available with symbols);
- *Export Selected Symbol(s) as SVG...* (only available with symbols);

### Sharing style items

The  *Import/Export* tool, at the left bottom of the Style Manager dialog, offers options to easily share symbols, color ramps, text formats and label settings with others. These options are also available through right-click over the items.

### Exporting items

You can export a set of items to an .XML file:

1. Expand the  *Import/Export* drop-down menu and select  *Export Item(s)...*
2. Choose the items you'd like to integrate. Selection can be done with the mouse or using a tag or a group previously set.
3. Press *Export* when ready. You'll be prompted to indicate the destination of the saved file. The XML format generates a single file containing all the selected items. This file can then be imported in another user's style library.

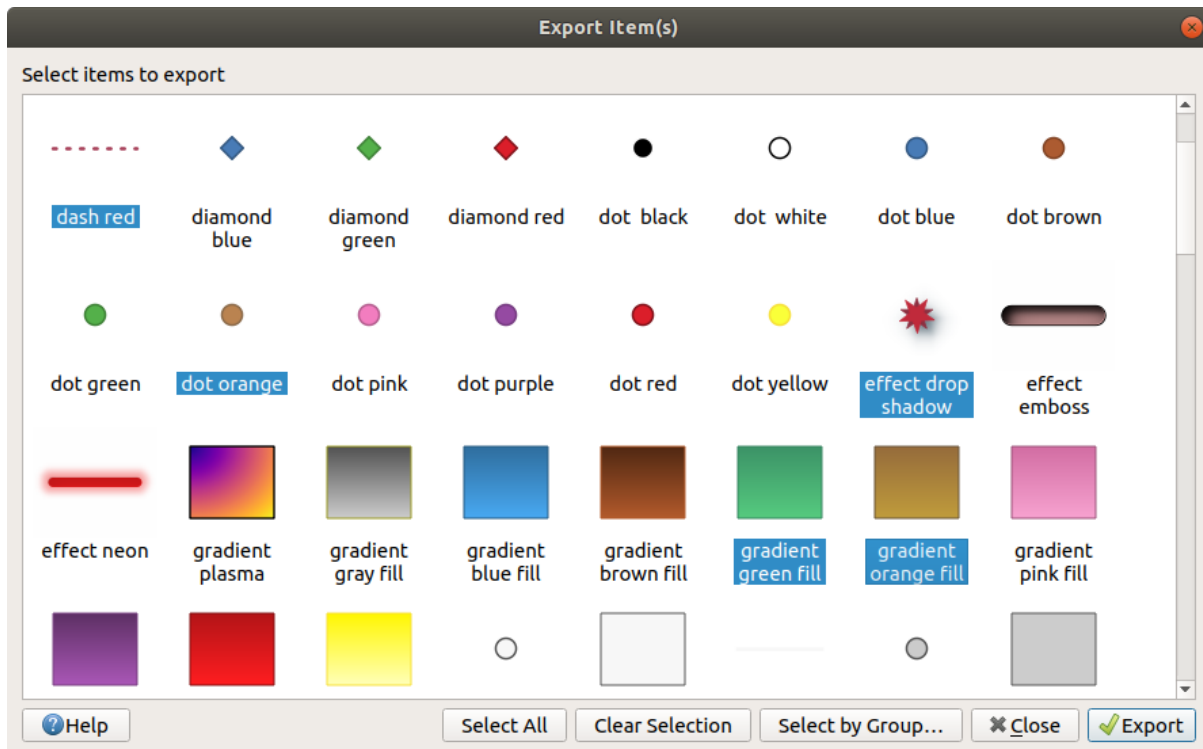




Abb. 14.3: Exporting style items

When symbols are selected, you can also export them to `.PNG` or `.SVG`. Exporting to `.PNG` or `.SVG` (both not available for other style item types) creates a file for each selected symbol in a given folder. The `SVG` folder can be added to the *SVG paths* in *Settings ► Options ► System* menu of another user, allowing him direct access to all these symbols.

## Importing items

You can extend your style library by importing new items:

1. Expand the  *Import/Export* drop-down menu and select  *Import Item(s)* at the left bottom of the dialog.
2. In the new dialog, indicate the source of the style items (it can be an `.xml` file on the disk or a url).
3. Set whether to ☐ *Add to favorites* the items to import.
4. Check ☐ *Do not import embedded tags* to avoid the import of tags associated to the items being imported.
5. Give the name of any *Additional tag(s)* to apply to the new items.
6. Select from the preview the symbols you want to add to your library.
7. And press *Import*.

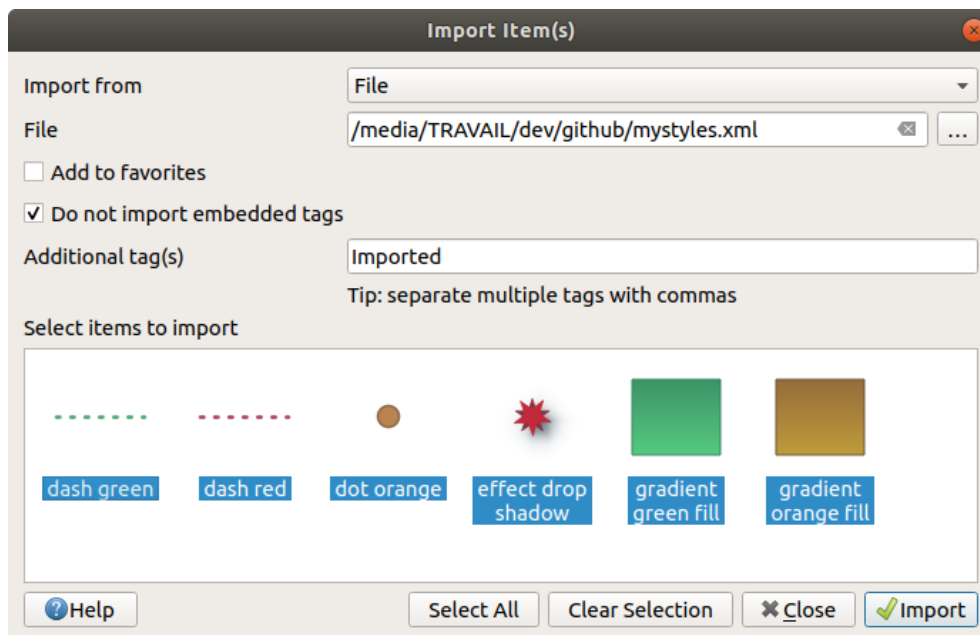


Abb. 14.4: Importing style items

## Using the Browser panel

It's also possible to import style items into the active user profile style database directly from the *Browser* panel:

1. Select the style .xml file in the browser
2. Drag-and-drop it over the map canvas or right-click and select *Import Style...*
3. Fill the *Import Items* dialog following [Importing items](#)
4. Press *Import* and the selected style items are added to the style database

Double-clicking the style file in the browser opens the *Style Manager* dialog showing the items in the file. You can select them and press *Copy to Default Style...* to import them into the active style database. Tags can be assigned to items. Also available through right-click, *Open Style...* command.

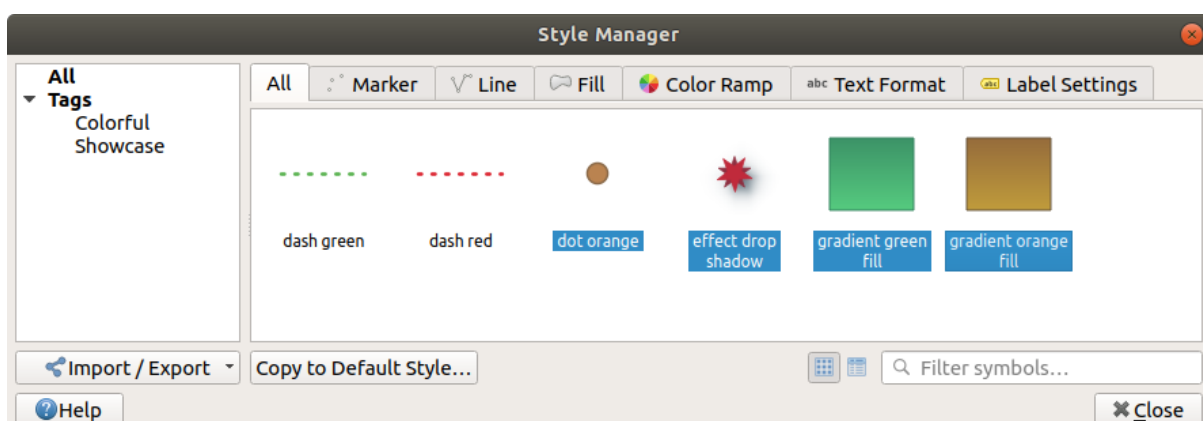



Abb. 14.5: Opening a style items file

The dialog also allows to export single symbols as .PNG or .SVG files.

## Using the online repository


The QGIS project maintains a repository with a collection of styles shared by QGIS users. This is available at <https://plugins.qgis.org/styles> and can be accessed from the *Style Manager* dialog, pressing the  *Browse Online Styles* button at the bottom.

From that repository, you can:

1. Browse and search for any style items, based on their type or name
2. Download the style file and unzip it
3. Load the `.xml` based file into your style database in QGIS, using any of the aforementioned import methods.

### 14.1.2 Setting a Color Ramp

The Color ramp tab in the *Style Manager* dialog helps you preview different color ramps based on the category selected in the left panel.

To create a custom color ramp, activate the Color ramp tab and click the  *Add item* button. The button reveals a drop-down list to choose the ramp type:

- *Gradient*: given a start and end colors, generates a color ramp which can be **continuous** or **discrete**. With double-clicking the ramp preview, you can add as many intermediate color stops as you want. Click on the color stop indicator and under *Gradient stop* you can:
  - adjust its *Relative position* from the color ramp start. Also possible dragging the indicator with the mouse, or pressing the arrow keys (combine with `Shift` key for a larger move)
  - specify the color model to use when interpolating between colors: it can be *RGB*, *HSL* or *HSV*. In some circumstances, this option can help avoid desaturated mid tones, resulting in more visually pleasing gradients.
  - set the direction which the interpolation should follow for the **Hue** component of a *HSL* or *HSV* color specification. It can be *Clockwise* or *Counterclockwise*.
  - set the *color properties*
  - remove the color stop pressing *Delete stop* or `DEL`

The *Plots* group provides another graphical way to design the color ramp, changing the position or the opacity and HSL components of the color stops.

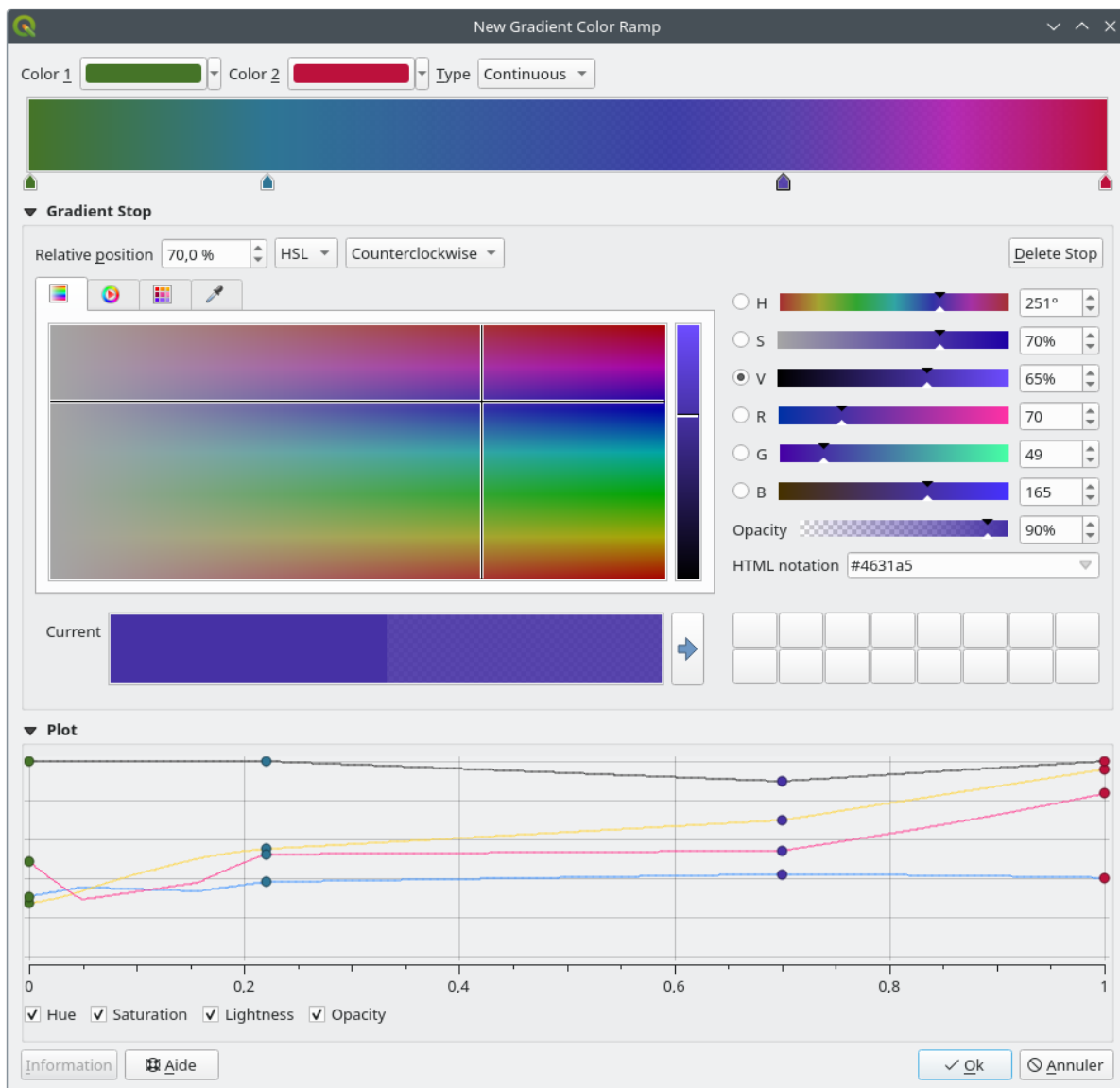


Abb. 14.6: Example of custom gradient color ramp with multiple stops

---

**Hinweis:** Drag-and-drop a color from a color spot onto the gradient ramp preview adds a new color stop.

---

- *Color presets*: allows to create a color ramp consisting of a list of colors selected by the user;
- *Random*: creates a random set of colors based on range of values for *Hue*, *Saturation*, *Value* and *Opacity* and a number of colors (*Classes*);
- *Catalog: ColorBrewer*: a set of predefined discrete color gradients you can customize the number of colors in the ramp;
- or *Catalog: cpt-city*: an access to a whole catalog of color gradients to locally *save as standard gradient*. The *cpt-city* option opens a new dialog with hundreds of themes included ,out of the box‘.

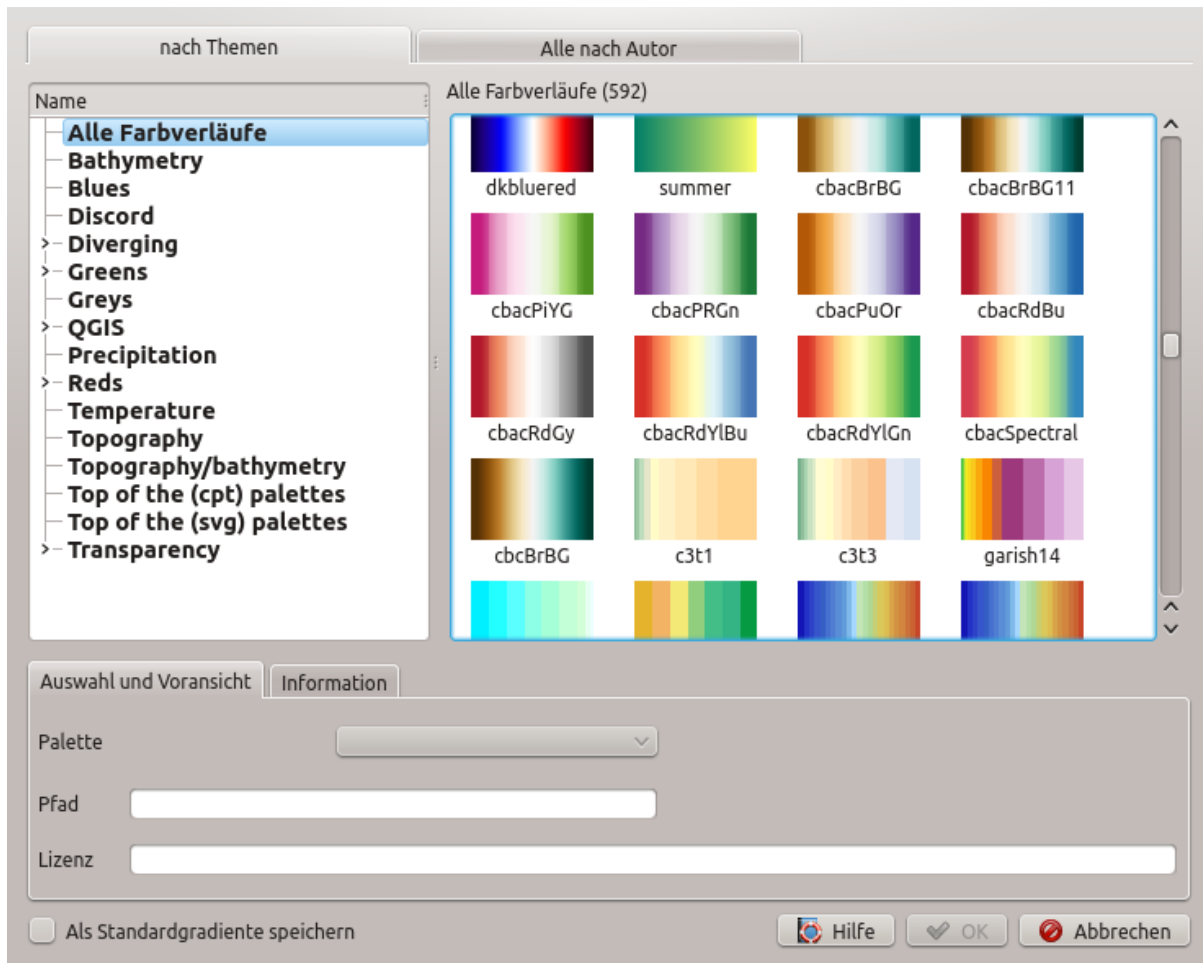



Abb. 14.7: cpt-city dialog with hundreds of color ramps

### 14.1.3 Creating a Legend Patch Shape

To create a new Legend Patch Shape, activate the *Legend Patch Shapes* tab and click the  Add item button. The button reveals a drop-down list to choose the geometry type:

- *Marker Legend Patch Shape...*: to use with point geometries.
- *Line Legend Patch Shape...*: to use with line geometries.
- *Fill Legend Patch Shape...*: to use with polygon geometries.

All three options will show the same dialog.

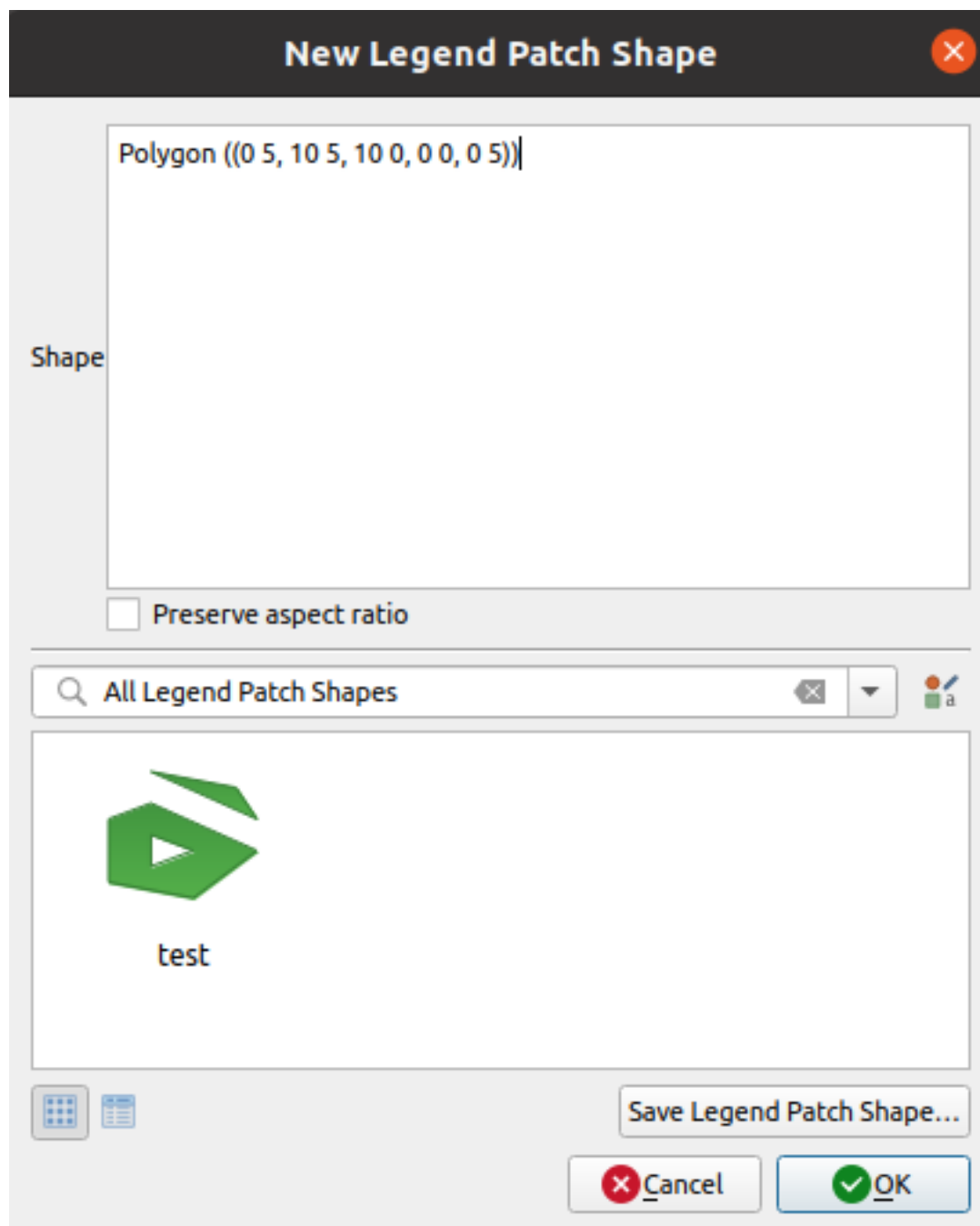


Abb. 14.8: Create a new Legend Patch Shape

Only the shape type and displayed legend patch shapes will differ regarding to the chosen geometry type. The following options will be available:

- *Shape*: define the shape of the legend patch shape as a WKT string. Single and multipart geometries may be used, but no GeometryCollection.
- ☒ *Preserve aspect ratio*
- ☒ *Icon View* or ☐ *List View* of available legend patch shapes, filtered by tags.

When the new Shape is defined you can *Save Legend Patch Shape...* or press *OK*, which will both lead to the same dialog.



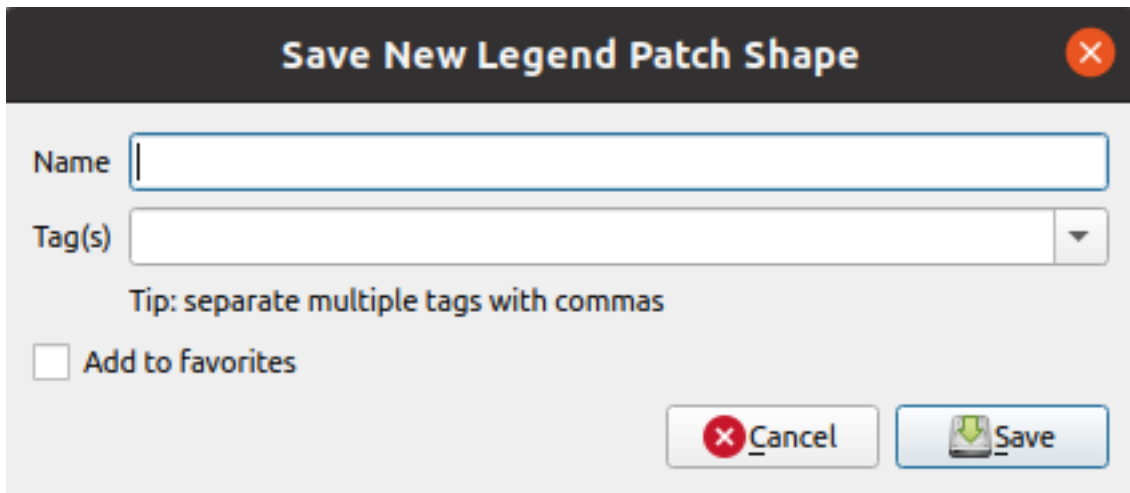


Abb. 14.9: Save a new Legend Patch Shape

Here you have to choose a name, tags to describe the shape and if it should be added to favorites.

If you press *Save...*, the shape is added to the list and you are directed back to the *New Legend Patch Shape* dialog to keep creating new shapes.

## 14.2 The Symbol Selector

The Symbol selector is the main dialog to design a symbol. You can create or edit Marker, Line or Fill Symbols.

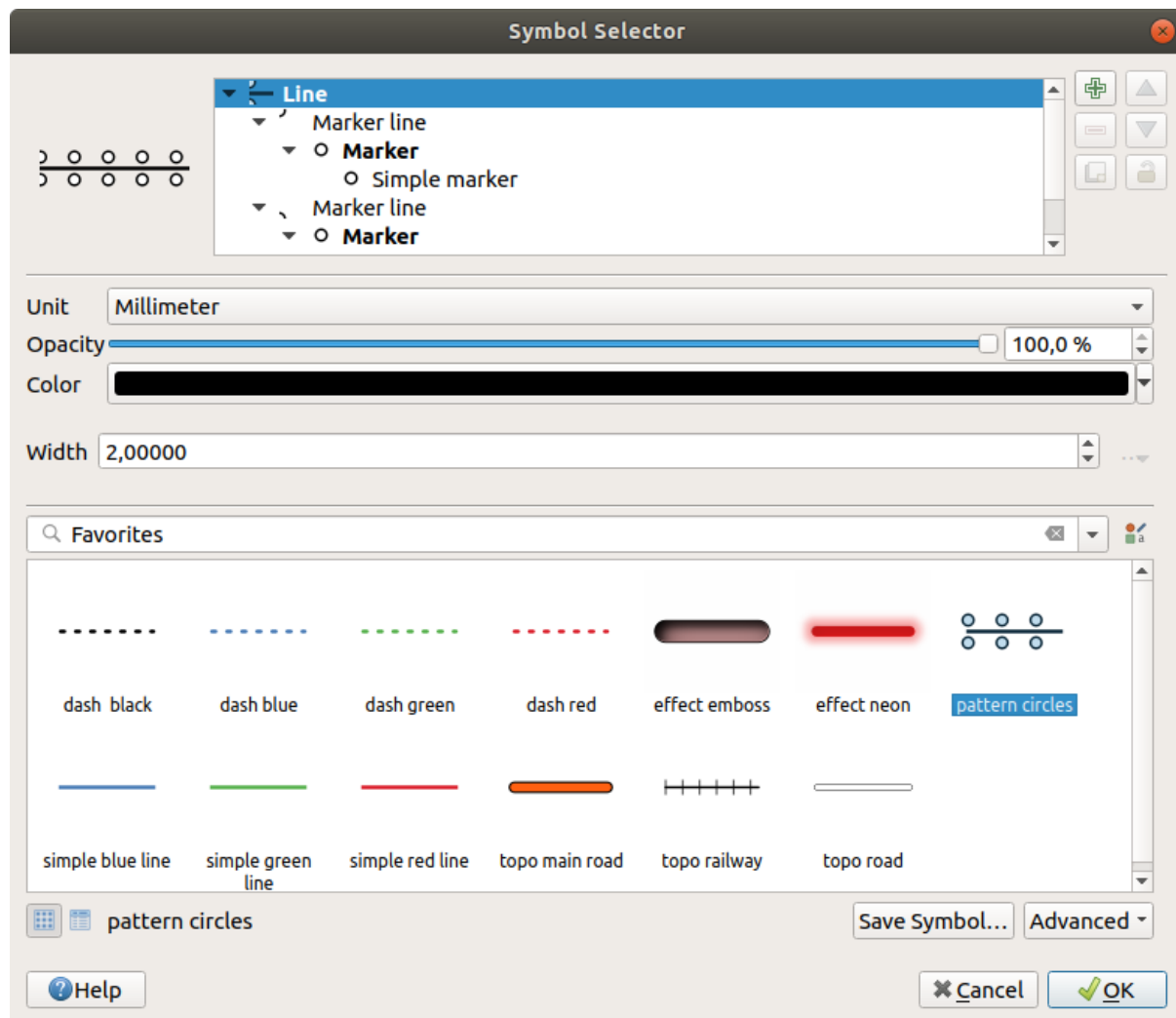


Abb. 14.10: Designing a Line symbol





Two main components structure the symbol selector dialog:

- the symbol tree, showing symbol layers that are combined afterwards to shape a new global symbol
- and settings to configure the selected symbol layer in the tree.

### 14.2.1 The symbol layer tree

A symbol can consist of several *Symbol layers*. The symbol tree shows the overlay of these symbol layers that are combined afterwards to shape a new global symbol. Besides, a dynamic symbol representation is updated as soon as symbol properties change.

Depending on the level selected in the symbol tree items, various tools are made available to help you manage the tree:

-  add new symbol layer: you can stack as many symbols as you want
-  remove the selected symbol layer
- lock colors of symbol layer: a  locked color stays unchanged when user changes the color at the global (or upper) symbol level
-  duplicate a (group of) symbol layer(s)

- move up or down the symbol layer

## 14.2.2 Configuring a symbol

In QGIS, configuring a symbol is done in two steps: the symbol and then the symbol layer.

### The symbol

At the top level of the tree, it depends on the layer geometry and can be of **Marker**, **Line** or **Fill** type. Each symbol can embed one or more symbols (including, of any other type) or symbol layers.

You can setup some parameters that apply to the global symbol:

- *Unit*: it can be **Millimeters**, **Points**, **Pixels**, **Meters at Scale**, **Map units** or **Inches** (see [Unit Selector](#) for more details)
- *Opacity*
- *Color*: when this parameter is changed by the user, its value is echoed to all unlocked sub-symbols color
- *Size* and *Rotation* for marker symbols
- *Width* for line symbols

---

**Tip:** Use the *Size* (for marker symbols) or the *Width* (for line symbols) properties at the symbol level to proportionally resize all of its embedded *symbol layers* dimensions.


---






---

**Bemerkung:** The *Data-defined override* button next to the width, size or rotation parameters is inactive when setting the symbol from the Style manager dialog. When the symbol is connected to a map layer, this button helps you create *proportional or multivariate analysis* rendering.

---

- A preview of the *symbols library*: Symbols of the same type are shown and, through the editable drop-down list just above, can be filtered by free-form text or by *categories*. You can also update the list of symbols using the  Style Manager button and open the eponym dialog. There, you can use any capabilities as exposed in [The Style Manager](#) section.

The symbols are displayed either:

- in an icon list (with thumbnail, name and associated tags) using the  List View button below the frame;
- or as icon preview using the  Icon View button.
- Press the *Save Symbol* button to add the symbol being edited to the symbols library.
- With the *Advanced*  option, you can:
  - for line and fill symbols, *Clip features to canvas extent*.
  - for fill symbols, *Force right-hand rule orientation*: allows forcing rendered fill symbols to follow the standard „right hand rule“ for ring orientation (i.e, polygons where the exterior ring is clockwise, and the interior rings are all counter-clockwise).

The orientation fix is applied while rendering only, and the original feature geometry is unchanged. This allows for creation of fill symbols with consistent appearance, regardless of the dataset being rendered and the ring orientation of individual features.

- Depending on the *symbolology* of the layer a symbol is being applied to, additional settings are available in the *Advanced* menu:

- \* *Symbol levels...* to define the order of symbols rendering

- \* *Data-defined Size Legend*
- \* *Match to Saved Symbols...* and *Match to Symbols from File...* to automatically *assign symbols to classes*
- \* *Animation settings*




### The symbol layer

At a lower level of the tree, you can customize the symbol layers. The available symbol layer types depend on the upper symbol type. You can apply on the symbol layer  *paint effects* to enhance its rendering.

Because describing all the options of all the symbol layer types would not be possible, only particular and significant ones are mentioned below.

### Common parameters

Some common options and widgets are available to build a symbol layer, regardless it's of marker, line or fill sub-type:

- the *color selector* widget to ease color manipulation
- *Units*: it can be **Millimeters**, **Points**, **Pixels**, **Meters at Scale**, **Map units** or **Inches** (see *Unit Selector* for more details)
- the  *Data-defined override* widget near almost all options, extending capabilities of customizing each symbol (see *Datendefinierte Übersteuerung Setup* for more information)
- the  *Enable symbol layer* option controls the symbol layer's visibility. Disabled symbol layers are not drawn when rendering the symbol but are saved in the symbol. Being able to hide symbol layers is convenient when looking for the best design of your symbol as you don't need to remove any for the testing. The data-defined override then makes it possible to hide or display different symbol layers based on expressions (using, for instance, feature attributes).
- the  *Draw effects* button for *effects rendering*.

---

**Bemerkung:** While the description below assumes that the symbol layer type is bound to the feature geometry, keep in mind that you can embed symbol layers in each others. In that case, the lower level symbol layer parameter (placement, offset...) might be bound to the upper-level symbol, and not to the feature geometry itself.

---

### Marker Symbols

Appropriate for point geometry features, marker symbols have several *Symbol layer types*:

- **Simple marker** (default)

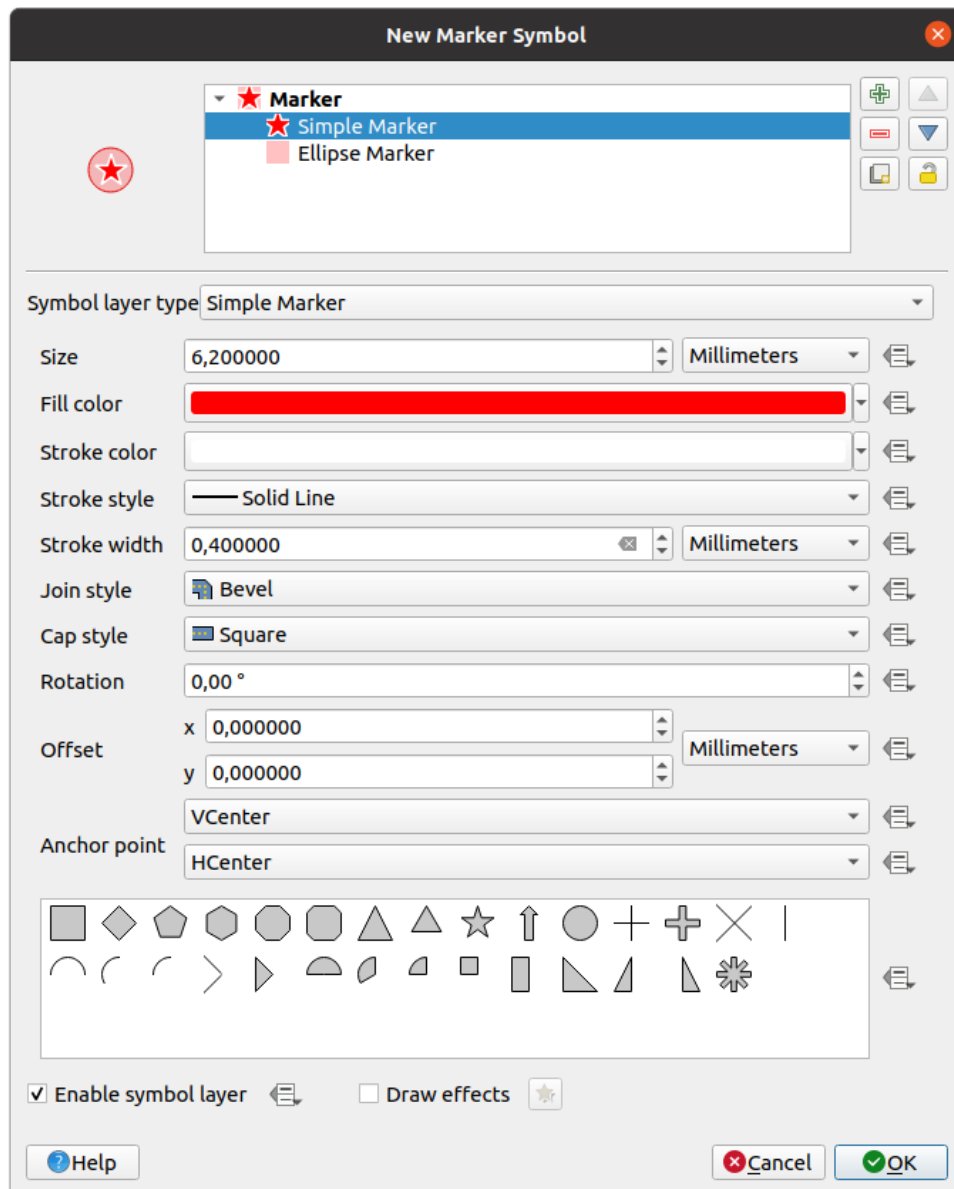




Abb. 14.11: Designing a Simple Marker Symbol

The simple marker symbol layer type has the following properties:

- *Size* in various supported units
- *Fill color*
- *Stroke color*, *Stroke style* from a predefined list and *Stroke size*
- *Join style*: it can be **Bevel**, **Miter** or **Round**
- *Cap style*: it can be **Square**, **Flat** or **Round**
- *Drehung*
- *Offset* in *X* and *Y* directions from the feature
- *Anchor point*: defining the quadrant point on the symbol to settle as placement origin. This is the point the *Offset* is applied on.
- **Animated marker** (see [Animated marker](#))
- **Ellipse marker**: a simple marker symbol layer, with customizable width and height

- **Filled marker:** similar to the simple marker symbol layer, except that it uses a *fill sub symbol* to render the marker. This allows use of all the existing QGIS fill (and stroke) styles for rendering markers, e.g. gradient or shapeburst fills.
- **Font marker:** similar to the simple marker symbol layer, except that it uses installed fonts to render the marker. Its additional properties are:
  - *Font family*
  - *Font style*
  - *Character(s)*, representing the text to display as symbol. They can be typed in or selected from the font characters collection widget and you can live *Preview* them with the selected settings.
- **Geometry generator** (see *The Geometry Generator*)
- **Mask:** its sub-symbol defines a mask shape whose color property will be ignored and only the opacity will be used. This is convenient when the marker symbol overlaps with labels or other symbols whose colors are close, making it hard to decipher. More details at *Eigenschaften Masken*.
- **Raster image marker:** use an image (PNG, JPG, BMP ...) as marker symbol. The image can be a file on the disk, a remote URL, embedded in the style database (*more details*) or it can be encoded as a base64 string.  
Width and height of the image can be set independently or using the  Lock aspect ratio. The size can be set using any of the *common units* or as a percentage of the image's original size (scaled by the width).
- **Vector Field marker** (see *The Vector Field Marker*)
- **SVG marker:** provides you with images from your SVG paths (set in *Settings ► Options... ► System* menu) to render as marker symbol. Width and height of the symbol can be set independently or using the  Lock aspect ratio. Each SVG file colors and stroke can also be adapted. The image can be a file on the disk, a remote URL, embedded in the style database (*more details*) or it can be encoded as a base64 string.  
The symbol can also be set with *Dynamic SVG parameters*. See *Parametrizable SVG* section to parametrize an SVG symbol.

---

### **Bemerkung:** SVG version requirements

QGIS renders SVG files that follow the *SVG Tiny 1.2 profile*, intended for implementation on a range of devices, from cellphones and PDAs to laptop and desktop computers, and thus includes a subset of the features included in SVG 1.1 Full, along with new features to extend the capabilities of SVG.

Some features not included in these specifications might not be rendered correctly in QGIS.

---

## Line Symbols

Appropriate for line geometry features, line symbols have the following symbol layer types:

- **Simple line** (default)

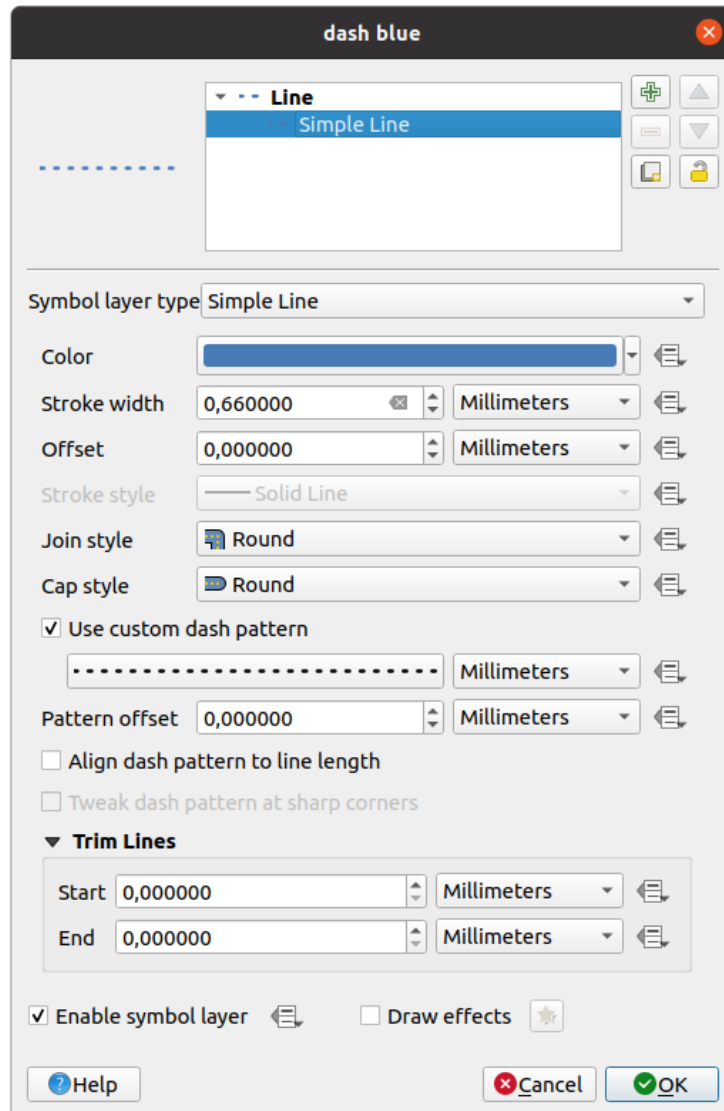




Abb. 14.12: Designing a Simple Line Symbol




The simple line symbol layer type has many of the same properties as the *simple marker symbol*, and in addition:

- ☒ *Use custom dash pattern*: overrides the *Stroke style* setting with a custom dash. You would need to define length of consecutive dashes and spaces shaping the model, in the chosen unit. The total length of the pattern is displayed at the bottom of the dialog.
- *Pattern offset*: the positioning of the dashes/spaces in the line can be tweaked, so that they can be placed at nicer positions to account for corners in the line (also can be used potentially to „align“ adjacent dash pattern borders)
- ☒ *Align dash pattern to line length*: the dash pattern length will be adjusted so that the line will end with a complete dash element, instead of a gap.
- ☒ *Tweak dash pattern at sharp corners*: dynamically adjusts the dash pattern placement so that sharp

corners are represented by a full dash element coming into and out of the sharp corner. Dependent on *Align dash pattern to line length*.

- *Trim lines from Start and/or End*: allows for the line rendering to trim off the first x mm and last y mm from the actual line string when drawing the line. It supports a range of *units*, including percentage of the overall line length, and can be data defined for extra control. The start/end trim distance can be used e.g. when creating complex symbols where a line layer should not overlap marker symbol layers placed at the start and end of the line.
- **Arrow**: draws lines as curved (or not) arrows with a single or a double head with configurable (and data-defined):
  - *Head type*
  - *Arrow type*
  - *Arrow width*
  - *Arrow width at start*
  - *Head length*
  - *Head thickness*
  - *Offset*

It is possible to create  *Curved arrows* (the line feature must have at least three vertices) and  *Repeat arrow on each segment*. It also uses a *fill symbol* such as gradients or shapeburst to render the arrow body. Combined with the geometry generator, this type of layer symbol helps you representing flow maps.

- **Geometry generator** (see *The Geometry Generator*)
- **Interpolated line**: allows to render a line whose *Stroke width* and/or *Color* may be constant (given a *Fixed width* and *Single color* parameters) or vary along the geometry. When varying, necessary inputs are:
  - *Start value* and *End value*: Values that will be used for interpolation at the extremities of the features geometry. They can be fixed values, feature's attributes or based on an expression.
  - *Min. value* and *Max. value*: Values between which the interpolation is performed. Press the  Load button to automatically fill them based on the minimum and maximum start/end values applied to the layer.
  - Only available for the stroke option:
    - \* *Min. width* and *Max. width*: define the range of the varying width. *Min. width* is assigned to the *Min. value* and *Max. width* to the *Max. value*. A *unit* can be associated.
    - \*  *Use absolute value*: only consider absolute value for interpolation (negative values are used as positive).
    - \*  *Ignore out of range*: by default, when the [start value – end value] range of a feature is not included in the [min. value – max. value] range, the out-of-bounds parts of the feature's geometry are rendered with the min or max width. Check this option to not render them at all.
  - For varying color, you can use any of the interpolation methods of *color ramp classification*



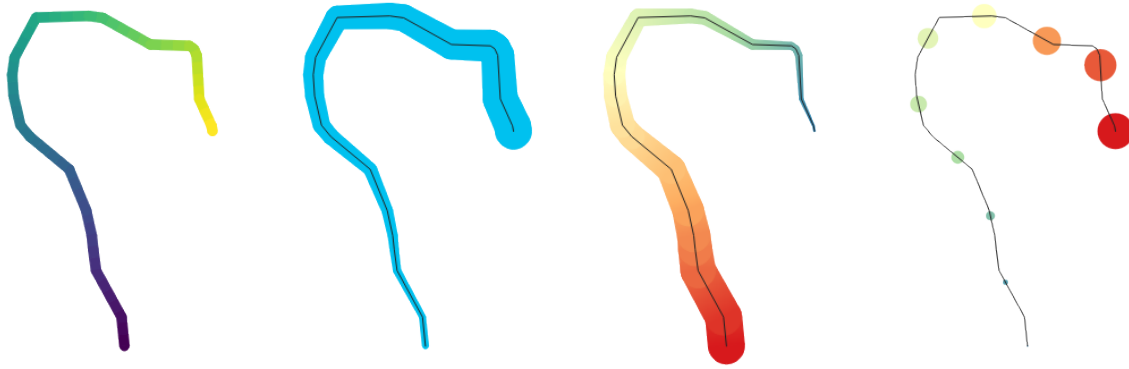


Abb. 14.13: Examples of interpolated lines

- **Marker line:** repeats a *marker symbol* over the length of a line.
  - The *Marker placement* can be set using a regular interval setting (starting from the first vertex) and/or the line geometry property (on first or last vertex, inner vertices, the central point of the line or of each segment, or on every curve point).
  - When first or last vertex placement is enabled, the ☒ *Place on every part extremity* option will make the markers render also at the first or last vertex for every part of multipart geometries.
  - *Offset along line:* the markers placement can also be given an offset along the line, in the `ref:unit` `<unit_selector>` of your choice (millimeters, points, map unit, meters at scale, percentage, ...):
    - \* A positive value offsets the markers symbols in the line direction (with *On first vertex* and *With interval* placements) and backwards (with *On last vertex* placement).
    - \* A negative value on a not closed line will result in no offset (for *On first vertex* and *On last vertex* placements) or backwards offset of the symbols (from the last vertex).
    - \* With a closed ring, QGIS treats the offset as continuing to loop around the ring (forward or backward). E.g. setting the offset to 150% (resp. -10% or -110%) results in the offset being treated as 50% (resp. 90%) of the length of the closed ring.
  - The ☒ *Rotate marker to follow line direction* option sets whether each marker symbol should be oriented relative to the line direction or not.
 

Because a line is often a succession of segments of different directions, the rotation of the marker is calculated by averaging over a specified distance along the line. For example, setting the *Average angle over* property to 4mm means that the two points along the line that are 2mm before and after the symbol placement are used to calculate the line angle for that marker symbol. This has the effect of smoothing (or removing) any tiny local deviations from the overall line direction, resulting in much nicer visual orientations of the marker line symbols.
  - *Line offset:* the marker symbols can also be offset from the line feature.
- **Hashed line:** repeats a line segment (a hash) over the length of a line symbol, with a line sub-symbol used to render each individual segment. In other words, a hashed line is like a marker line in which marker symbols are replaced with segments. As such, the hashed lines have the *same properties* as marker line symbols, along with:
  - *Hash length*
  - *Hash rotation*

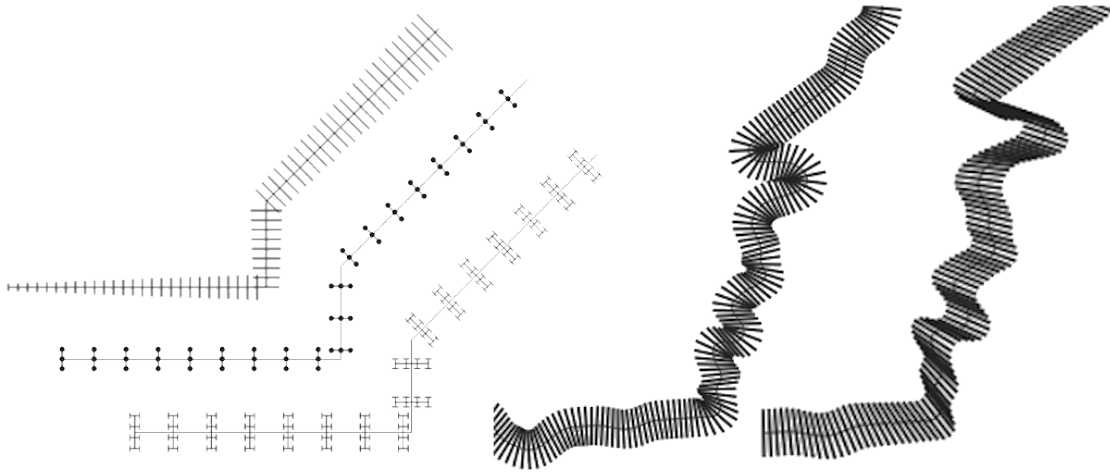


Abb. 14.14: Examples of hashed lines

- **Raster line:** renders and repeats a raster image following the length of a line feature shape. The *Stroke width*, *Offset*, *Join style*, *Cap style* and *Opacity* can be adjusted.

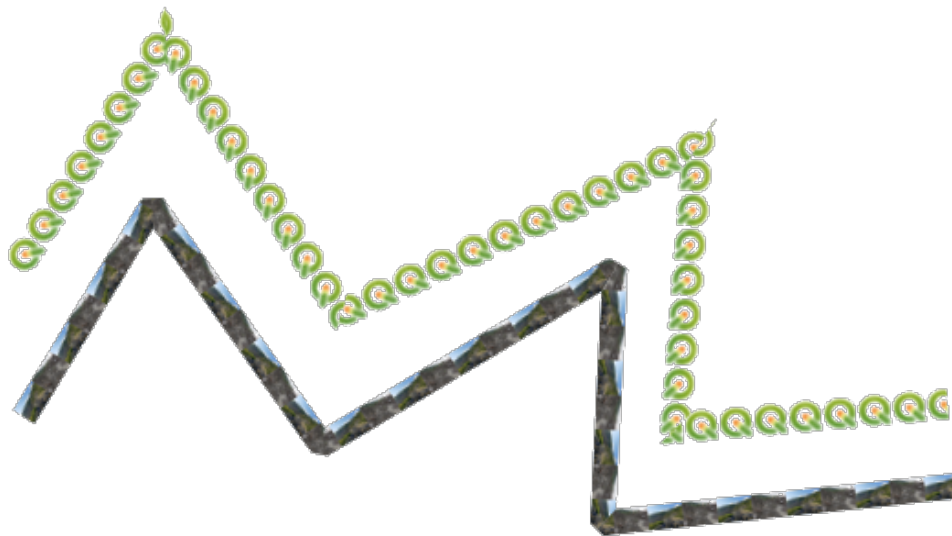


Abb. 14.15: Examples of raster lines

- **Lineburst:** renders a gradient along the width of a line. You can choose between *Two color* or *Color ramp* and the *Stroke width*, *Offset*, *Join style*, *Cap style* can be adjusted.



Abb. 14.16: Examples of lineburst lines

## Fill Symbols

Appropriate for polygon geometry features, fill symbols have also several symbol layer types:

- **Simple fill** (default): fills a polygon with a uniform color

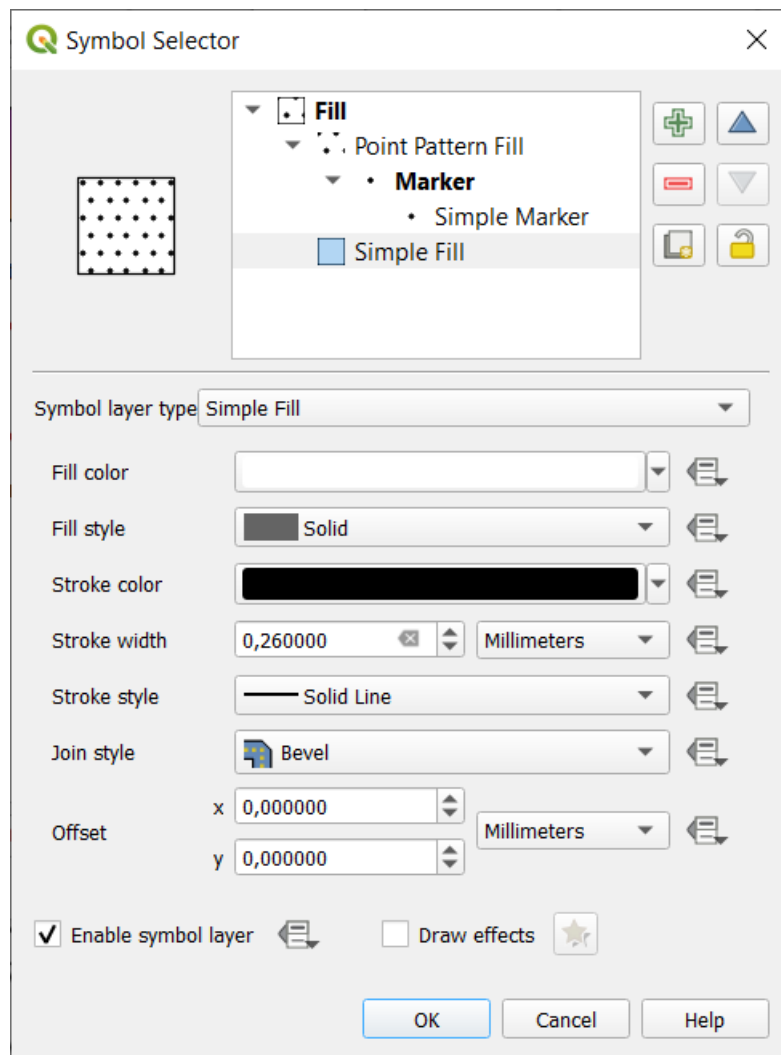


Abb. 14.17: Designing a Simple Fill Symbol

- **Centroid fill:** places a *marker symbol* at the centroid of the visible feature. The position of the marker may not be the real centroid of the feature, because calculation takes into account the polygon(s) clipped to area visible in map canvas for rendering and ignores holes. Use the *geometry generator symbol* if you want the exact centroid.

Sie können:

- *Force placement of markers inside polygons*
- *Draw markers on every part of multi-part features* or place the point only on its biggest part
- display the marker symbol(s) in whole or in part, keeping parts overlapping the current feature geometry (*Clip markers to polygon boundary*) or the geometry part the symbol belongs to (*Clip markers to current part boundary only*)

- **Geometry generator** (see *The Geometry Generator*)
- **Gradient fill:** uses a radial, linear or conical gradient, based on either simple two color gradients or a predefined *gradient color ramp* to fill polygons. The gradient can be rotated and applied on a single feature basis or across the whole map extent. Also start and end points can be set via coordinates or using the centroid (of feature or map). A data-defined offset can be defined.
- **Line pattern fill:** fills the polygon with a hatching pattern of *line symbol layer*. You can set:
  - *Alignment:* defines how the pattern is positioned relative to the feature(s):

- \* *Align pattern to feature*: lines are rendered within each feature
- \* *Align pattern to map extent*: a pattern is rendered over the whole map extent, allowing lines to align nicely across features
- *Rotation* of the lines, counter-clockwise
- *Spacing*: distance between consecutive lines
- *Offset* distance of the lines from the feature boundary
- *Clipping*: allows to control how lines in the fill should be clipped to the polygon shape. Options are:
  - \* *Clip During Render Only*: lines are created covering the whole bounding box of the feature and then clipped while drawing. Line extremities (beginning and end) will not be visible.
  - \* *Clip Lines Before Render*: lines are clipped to the exact shape of the polygon prior to rendering. Line extremities (including cap styles, start/end marker line objects, ...) will be visible, and may sometimes extend outside of the polygon (depending on the line symbol settings).
  - \* *No Clipping*: no clipping at all is done - lines will cover the whole bounding box of the feature
- **Point pattern fill**: fills the polygon with a grid pattern of *marker symbol*. You can set:
  - *Alignment*: defines how the pattern is positioned relative to the feature(s):
    - \* *Align pattern to feature*: marker lines are rendered within each feature
    - \* *Align pattern to map extent*: a pattern is rendered over the whole map extent, allowing markers to align nicely across features

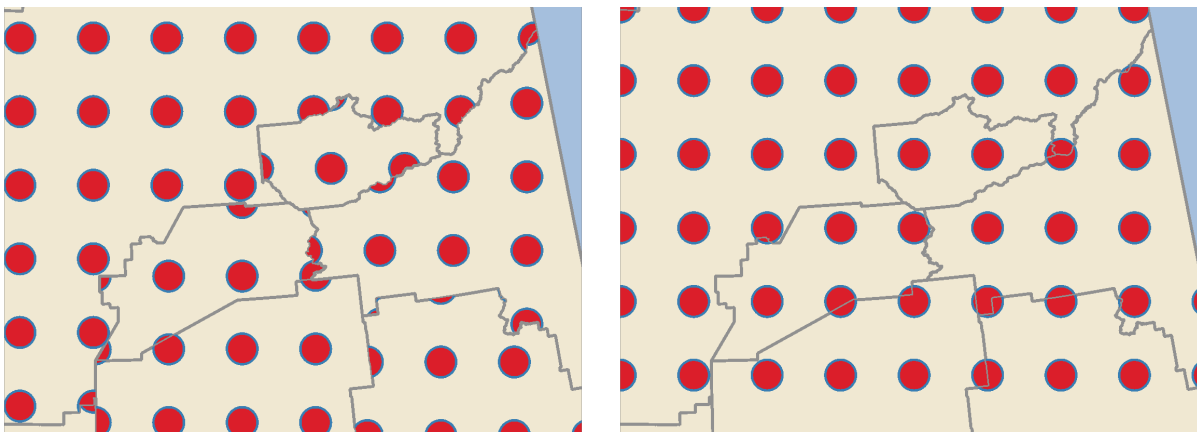


Abb. 14.18: Aligning point pattern to feature (left) and to map extent (right)

- *Distance*: *Horizontal* and *Vertical* distances between consecutive markers
- *Displacement*: a *Horizontal* (resp. *Vertical*) offset of alignment between consecutive markers in a column (resp. in a row)
- *Offset*: *Horizontal* and *Vertical* distances from the feature boundary
- *Clipping*: allows to control how markers in the fill should be clipped to the polygon shape. Options are:
  - \* *Clip to shape*: markers are clipped so that only the portions inside the polygon are visible
  - \* *Marker centroid within shape*: only markers where the center of the marker falls inside the polygon are drawn, but these markers won't be clipped to the outside of the polygon
  - \* *Marker completely within shape*: only markers which fall completely within the polygon are shown
  - \* *No clipping*: any marker which intersects at all with the polygon will be completely rendered (strictly speaking its the „intersects with the bounding box of the marker“)

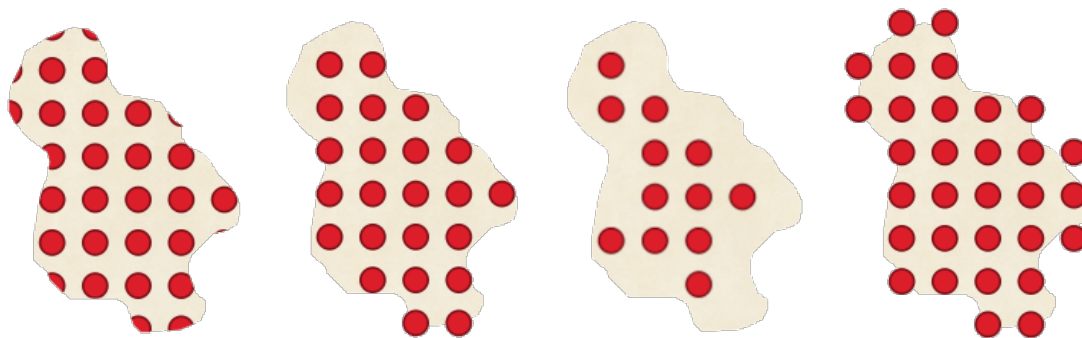


Abb. 14.19: Clipping markers in fill - From left to right: Clip to shape, Marker centroid within shape, Marker completely within shape, No clipping

- *Rotation* of the whole pattern, clockwise
- The *Randomize pattern* group setting allows each point in a point pattern fill to be randomly shifted up to the specified maximum distance *Horizontally* or *Vertically*. You can specify the maximum offset in any supported units, such as millimeters, points, map units, or even „percentage“ (where percentage is relative to the pattern width or height).

You can set an optional random number seed to avoid the symbol patterns „jumping“ around between map refreshes. Data defined overrides are also supported.

---

**Bemerkung:** The main difference between the *Randomize pattern* and the *random marker fill* symbol type is that the random offset with a point pattern allows for quasi-„regular“ placement of markers – because the points in the pattern are effectively constrained to a grid, this allows creation of semi-random fills which don’t have empty areas or overlapping markers. (As opposed to the random marker fill, which will always place points completely randomly... sometimes resulting in visual clusters of points or unwanted empty areas).

---

- **Random marker fill:** fills the polygon with a *marker symbol* placed at random locations within the polygon boundary. You can set:
  - *Count method:* whether the number of marker symbols to render is considered as an absolute count or density-based
  - *Point count:* the number of marker symbols to render,
  - an optional random number *seed*, to give consistent placement
  - *Density area:* in case of density-based count method, ensures the fill density of markers remains the same on different scale / zoom levels of markers whenever maps are refreshed (also allows random placement to play nice with QGIS server and tile-based rendering)
  - *Clip markers to polygon boundary:* whether markers rendered near the edges of polygons should be clipped to the polygon boundary or not
- **Raster image fill:** fills the polygon with tiles from a raster image (PNG JPG, BMP ...). The image can be a file on the disk, a remote URL or an embedded file encoded as a string (*more details*). Options include (data defined) opacity, image width, coordinate mode (object or viewport), rotation and offset. The image width can be set using any of the *common units* or as a percentage of the original size.
- **SVG fill:** fills the polygon using *SVG markers* of a given size (*Texture width*).
- **Shapeburst fill:** buffers a gradient fill, where a gradient is drawn from the boundary of a polygon towards the polygon’s centre. Configurable parameters include distance from the boundary to shade, use of color ramps or simple two color gradients, optional blurring of the fill and offsets.
- **Outline: Arrow:** uses a line *arrow symbol* layer to represent the polygon boundary. The settings for the outline arrow are the same as for arrow line symbols.

- **Outline: Hashed line:** uses a *hash line symbol* layer to represent the polygon boundary (*Rings*) which can be the interior rings only, the exterior ring only or all the rings). The other settings for the outline hashed line are the same as for hashed line symbols.
- **Outline: Marker line:** uses a *marker line symbol* layer to represent the polygon boundary (*Rings*) which can be the interior rings only, the exterior ring only or all the rings). The other settings for the outline marker line are same as for marker line symbols.
- **Outline: simple line:** uses a *simple line symbol* layer to represent the polygon boundary (*Rings*) which can be the interior rings only, the exterior ring only or all the rings). The *Draw line only inside polygon* option displays the polygon borders inside the polygon and can be useful to clearly represent adjacent polygon boundaries. The other settings for the outline simple line are the same as for simple line symbols.

---

**Bemerkung:** When geometry type is polygon, you can choose to disable the automatic clipping of lines/polygons to the canvas extent. In some cases this clipping results in unfavourable symbology (e.g. centroid fills where the centroid must always be the actual feature's centroid).

---

## Parametrizable SVG

You have the possibility to change the colors of a *SVG marker*. You have to add the placeholders `param(fill)` for fill color, `param(fill-opacity)` for fill opacity, `param(outline)` and `param(outline-opacity)` for stroke color and opacity respectively, and `param(outline-width)` for stroke width. These placeholders can optionally be followed by a default value, e.g.:

```
<svg width="100%" height="100%">
<rect fill="param(fill) #ff0000" fill-opacity="param(fill-opacity) 1" stroke=
↪ "param(outline) #00ff00" stroke-opacity="param(outline-opacity) 1" stroke-width=
↪ "param(outline-width) 10" width="100" height="100">
</rect>
</svg>
```

More generally, SVG can be freely parametrized using `param(param_name)`. This param can either be used as an attribute value or a node text:

```
<g stroke-width=".265" text-anchor="middle" alignment-baseline="param(align)">
  <text x="98" y="147.5" font-size="6px">param(text1) </text>
  <text x="98" y="156.3" font-size="4.5px">param(text2) </text>
</g>
```

The parameters can then be defined as expressions in the *Dynamic SVG parameters* table.

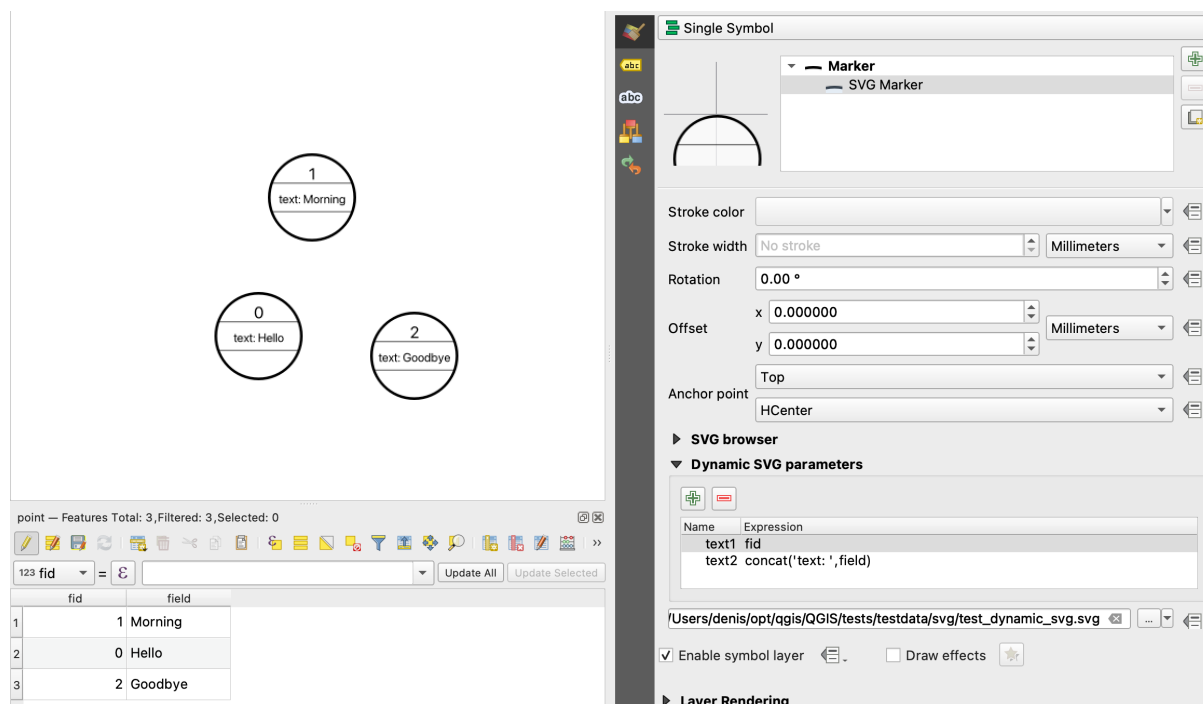


Abb. 14.20: Dynamic SVG parameters table

**Bemerkung:** QGIS is looking for a complete SVG node. So if your parameter is within a more complex node, you need to inject the complete node with the expression. For instance, `transform="rotate(param(angle))` will not work. Instead, you need to do `transform="param(rotation)"` and rotation parameter will be defined with the expression `'rotate(' || coalesce(my_field, 0) || '')`.

## The Geometry Generator

Available with all types of symbols, the *geometry generator* symbol layer allows to use *expression syntax* to generate a geometry on the fly during the rendering process. The resulting geometry does not have to match with the original *Geometry type* and you can add several differently modified symbol layers on top of each other.

A *Units* property can be set: when the geometry generator symbol is not applied to a layer (e.g., it is used on a layout item), this allows more control over the generated output.

Some examples:

- Render symbol as the centroid of a feature

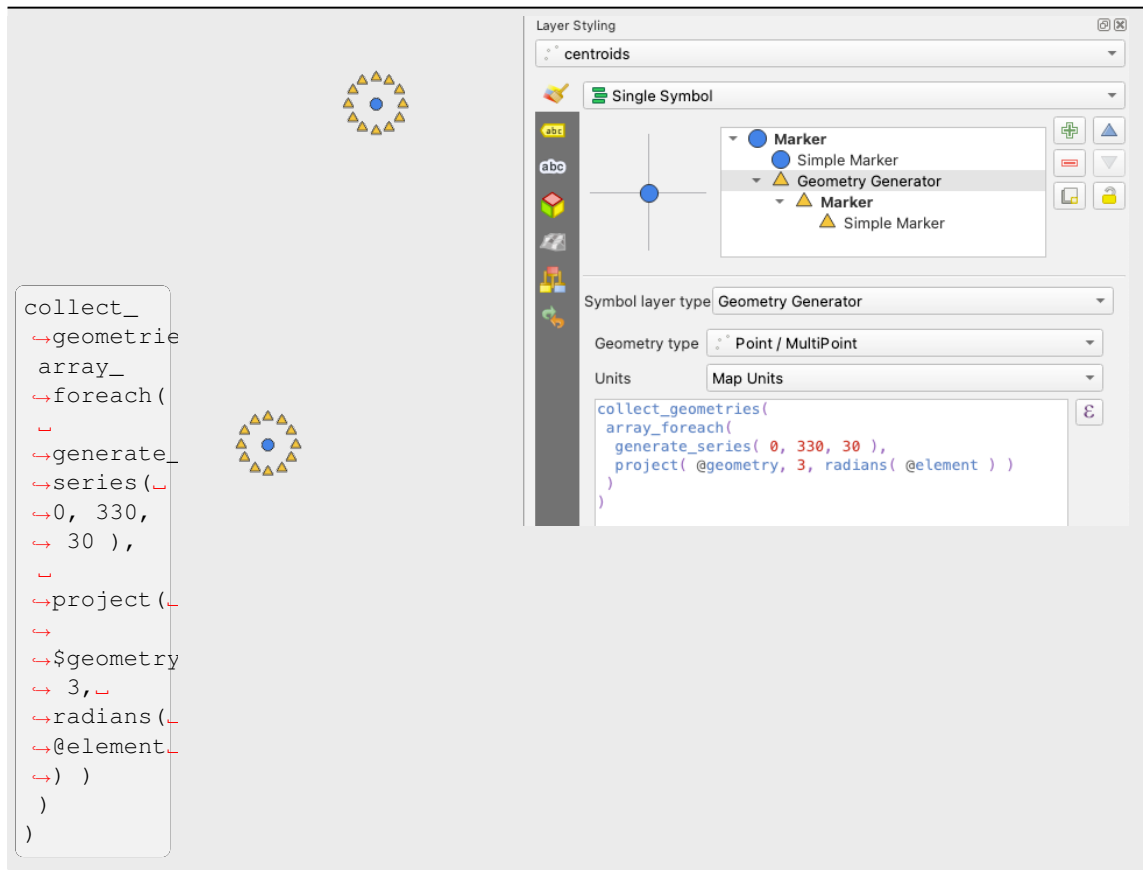
```
centroid( $geometry )
```

- Visually overlap features within a 100 map units distance from a point feature, i.e generate a 100m buffer around the point

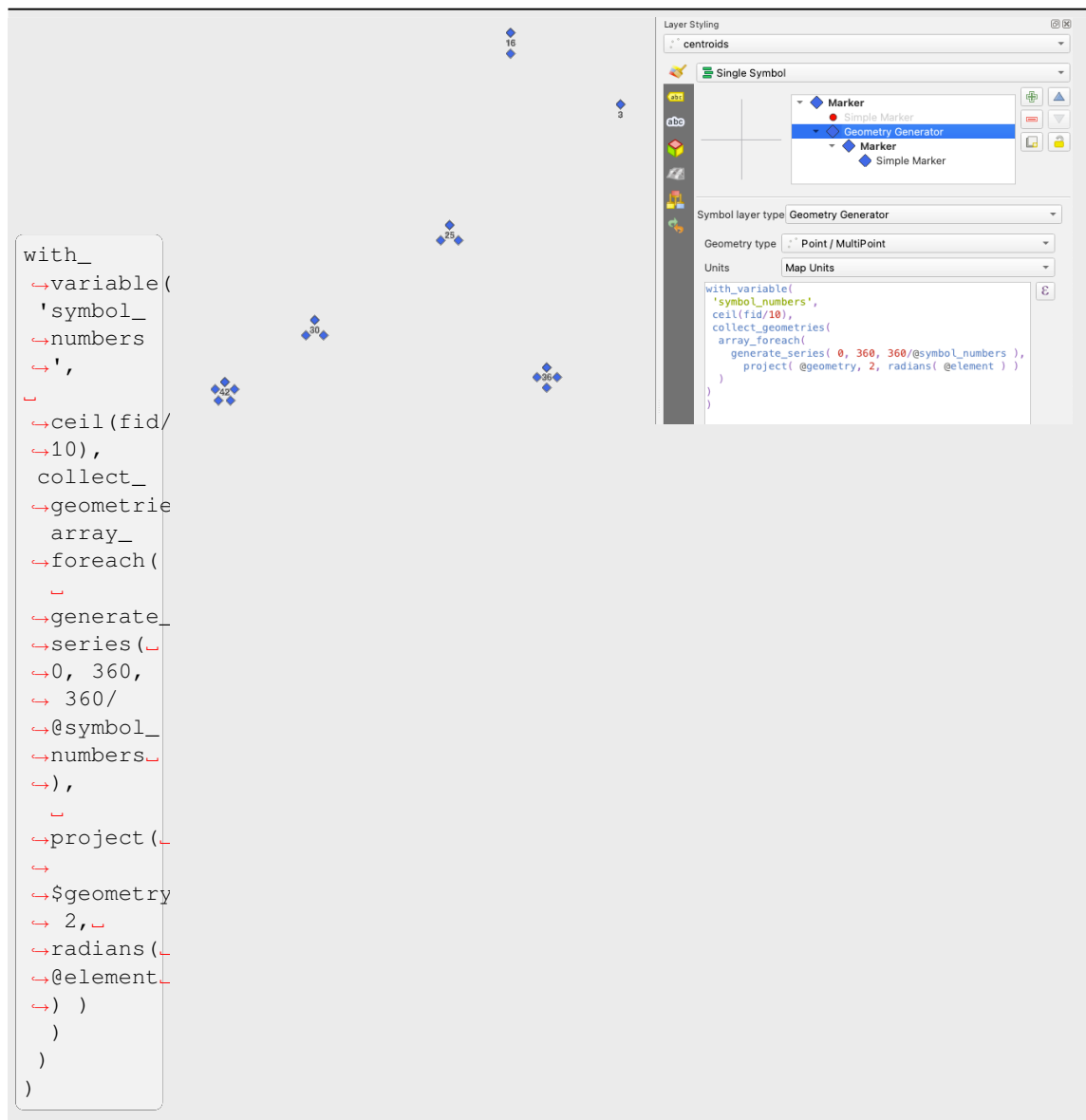
```
buffer( $geometry, 100 )
```

- Create a radial effect of points surrounding the central feature point when used as a MultiPoint geometry generator





- Create a radial effect of points surrounding the central feature point. The number of points varies based on a field.



- Create a curved arrow line connecting features of two layers based on their *relation*

```

collect_geometries(
  with_variable(
    'destination_points',
    relation_aggregate(
      'the_relation_id',
      'array_agg',
      centroid(
        $geometry
      ),
      array_foreach(
        @destination_points,
        make_line(
          centroid(
            @geometry
          ),
          project(
            centroid(
              make_line(
                centroid(
                  @geometry
                ),
                @element
              ),
              10,
              50
            ),
            @element
          )
        )
      )
    )
  )
)

```

Layer Styling

layer1

Single Symbol

Geometry Generator

Geometry type: LineString / MultiLineString

Units: Map Units

Expression:

```

collect_geometries(
  with_variable(
    'destination_points',
    relation_aggregate(
      'the_relation_id',
      'array_agg',
      centroid(
        $geometry
      ),
      array_foreach(
        @destination_points,
        make_line(
          centroid(
            @geometry
          ),
          project(
            centroid(
              make_line(
                centroid(
                  @geometry
                ),
                @element
              ),
              10,
              50
            ),
            @element
          )
        )
      )
    )
  )
)

```

## The Vector Field Marker

The vector field marker is used to display vector field data such as earth deformation, tidal flows, and the like. It displays the vectors as lines (preferably arrows) that are scaled and oriented according to selected attributes of data points. It can only be used to render point data; line and polygon layers are not drawn by this symbology.

The vector field is defined by attributes in the data, which can represent the field either by:

- **cartesian** components ( $x$  and  $y$  components of the field)
- or **polar** coordinates: in this case, attributes define `Length` and `Angle`. The angle may be measured either clockwise from north, or Counterclockwise from east, and may be either in degrees or radians.
- or as **height only** data, which displays a vertical arrow scaled using an attribute of the data. This is appropriate for displaying the vertical component of deformation, for example.

The magnitude of field can be scaled up or down to an appropriate size for viewing the field.

## Animated marker

Animated marker symbol type allows you to use a `.GIF`, `.WebP`, `.MNG`, etc. animation file to represent points on your map. You can specify:

- *File path*,
- *Frame rate*: number of steps that are shown per second, indicating how fast the animation is played,
- *Size* in any *supported unit*,
- *Opacity*,
- *Rotation*,
- *Offset* in  $x$  and  $y$  directions from the marker position,
- *Anchor point*

There are two ways to handle animated symbols:

- **When your map is not configured as an animation** (i.e. it's a standard QGIS project without animations), the frame for the animated markers will be determined solely by the current timestamp.
- **When your map is *configured as an animation***, the animated markers will sync with the animation's timeline. This means that animated markers will pause when the animation is paused, progress with the animation, and so forth. The map will also be redrawn according to the frame rate established for temporal animation. This mode is also applied when exporting an animation using the temporal controller.

## 14.3 Setting a label

Labels are textual information you can display on vector features or maps. They add details you could not necessarily represent using symbols. Two types of text-related items are available in QGIS:

- *Text Format*: defines the appearance of the text, including *font*, *size*, *colors*, *shadow*, *background*, *buffer*, ...

They can be used to render texts over the map (layout/map title, decorations, scale bar, ...), usually through the *font* widget.

To create a *Text Format* item:

1. Open the  *Style Manager* dialog

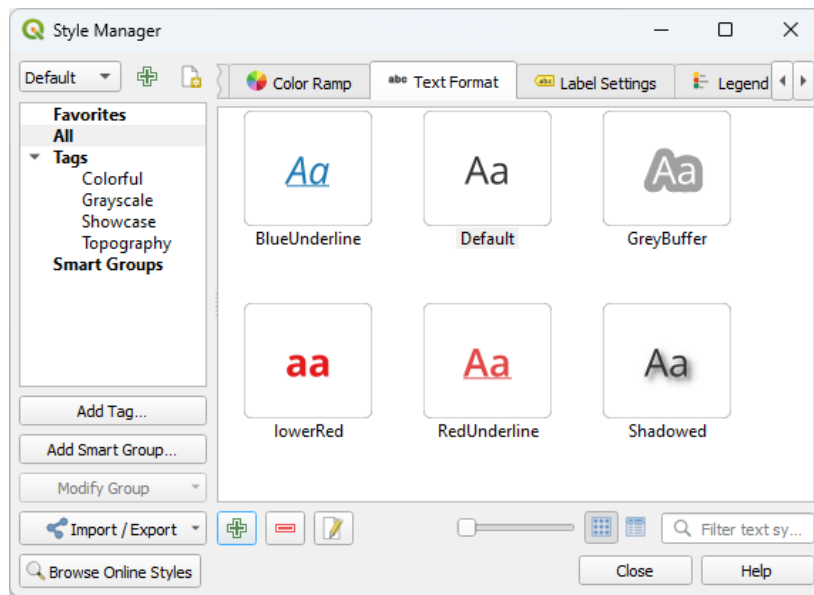

2. Activate the *Text format* tab

Abb. 14.21: Text formats in Style Manager dialog

3. Press the  Add item button. The *Text Format* dialog opens for *configuration*. As usual, these properties are *data-definable*.
- *Label Settings*: extend the text format settings with properties related to the location or the interaction with other texts or features (*callouts*, *placement*, *overlay*, *scale visibility*, mask ...).

They are used to configure smart labelling for vector layers through the  *Labels* tab of the vector *Layer Properties* dialog or *Layer Styling* panel or using the  Layer Labeling Options button of the *Label toolbar*.

To create a *Label Settings* item:

1. Open the  *Style Manager* dialog
2. Activate the *Label Settings* tab

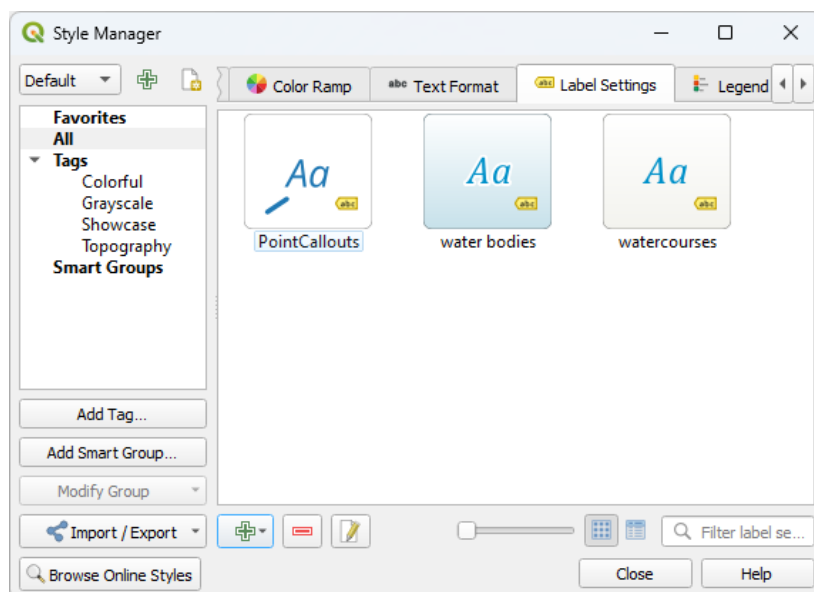

















Abb. 14.22: Label Settings in Style Manager dialog

- Press the  Add item menu and select the entry corresponding to the geometry type of the features you want to label.

The *Label Settings* dialog opens with the following properties. As usual, these properties are *data-definable*.

### 14.3.1 Formatting the label text

Whether you are configuring a *Text Format* or *Label Settings* item, you will be given the following options:

Properties tab	Text format	Label settings
<i>Text</i>		
<i>Formatting</i>		
<i>Buffer</i>		
<i>Mask</i>		
<i>Background</i>		
<i>Shadow</i>		
<i>Callout</i>		
<i>Placement</i>		
<i>Rendering</i>		

## Text tab

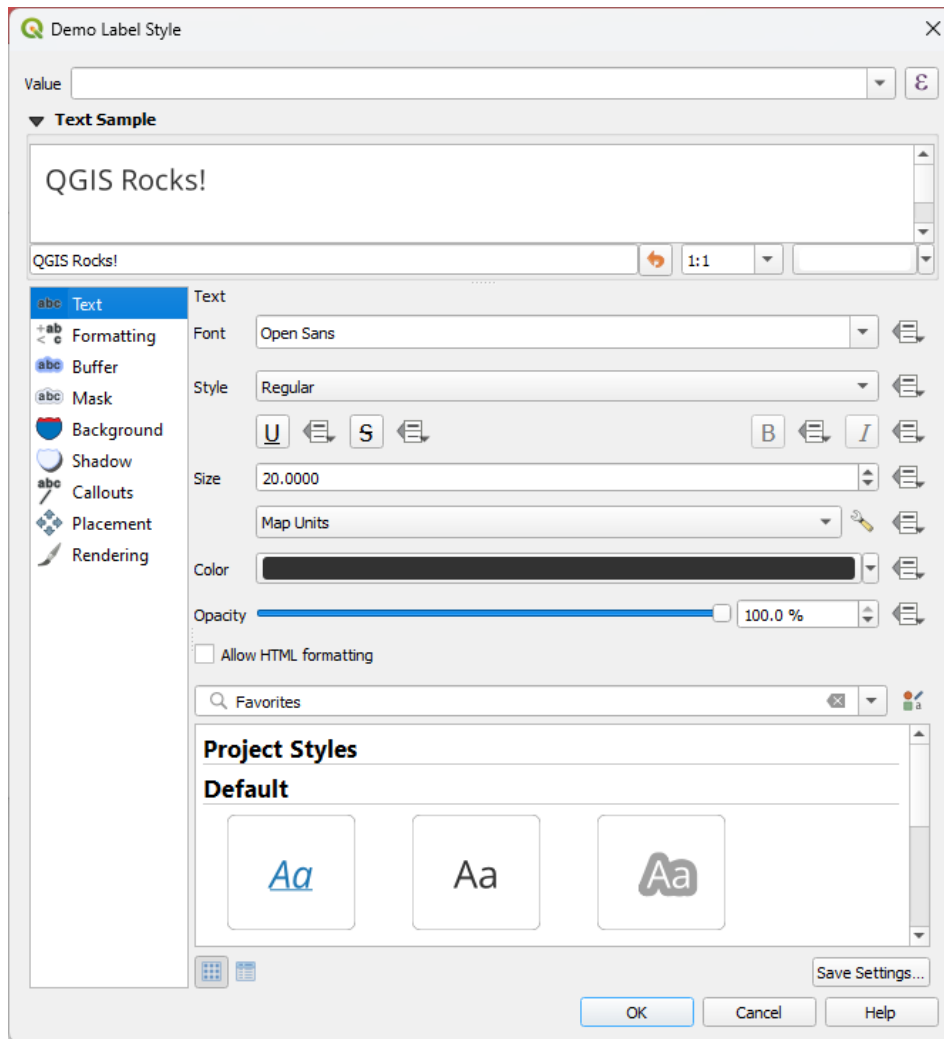


Abb. 14.23: Labels settings - Text tab

In the **abc Text** tab, you can set:

- the *Font*, from the ones available on your machine
- the *Style*: along with the common styles of the font, you can set whether the text should be underlined or strikethrough
- the *Size* in any *supported unit*
- the *Color*
- the *Opacity*
- and *Allow HTML Formatting*: The HTML formatting option enables the proper rendering of some HTML tags to customize the label. The supported HTML tags are:
  - Color, applicable to text, underline, strikethrough, and overline
  - Font properties (font family, font size, bold and italic)
  - Superscript and subscript components in text, where the text will be vertically <sup>super</sup> or <sub>sub</sub> aligned and automatically sized to 2/3 of the parent font size. You can also set a fixed font size for the superscript/subscript by including css rules, e.g.:

```
<sup style="font-size:33pt">my superscript text</sup>
```

The CSS formatting rules `vertical-align: super` or `vertical-align: sub` are also available in any other HTML element (annotation, layout label or HTML items, ...).

In order to use the HTML formatting, you need to provide the HTML code in the *Value* field. The expression is parsed and any supported HTML tag overrides its corresponding setting in the labels properties. They also combine well with other background, shadow, buffer... properties of labels.

Below an example of a HTML-based expression and rendering (applies different colors and underline to the same label):

```
format (
  '<span style="color:blue">%1</span> ( <span style="color:red"><u>%2 ft</u></span> ) ',
  title( lower( "Name" ) ),
  round($length)
)
```

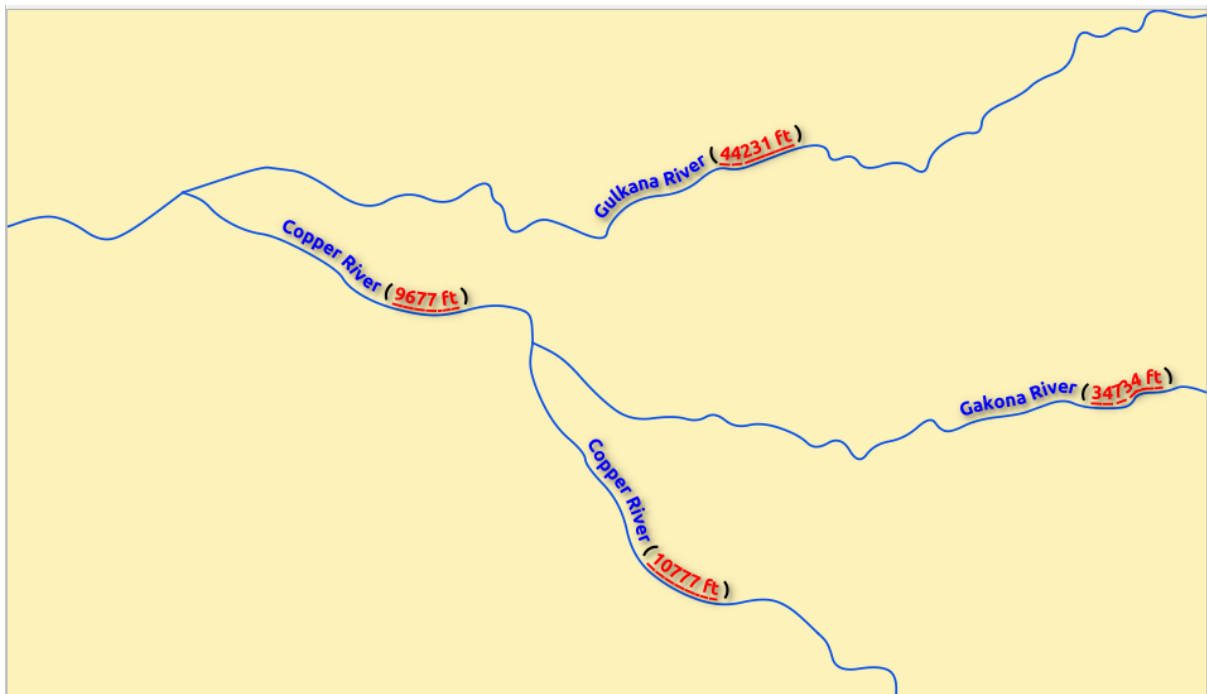


Abb. 14.24: Labeling with HTML formatting enabled

At the bottom of the tab, a widget shows a filterable list of compatible items stored in your *style manager database*. This allows you to easily configure the current text format or label setting based on an existing one, and also save a new item to the style database: Press the *Save format...* or *Save settings...* button and provide a name and tag(s).

**Bemerkung:** When configuring a *Label Settings* item, text format items are also available in this widget. Select one to quickly overwrite the current *textual properties* of the label. Likewise, you can create/overwrite a text format from there.



## Formatting tab

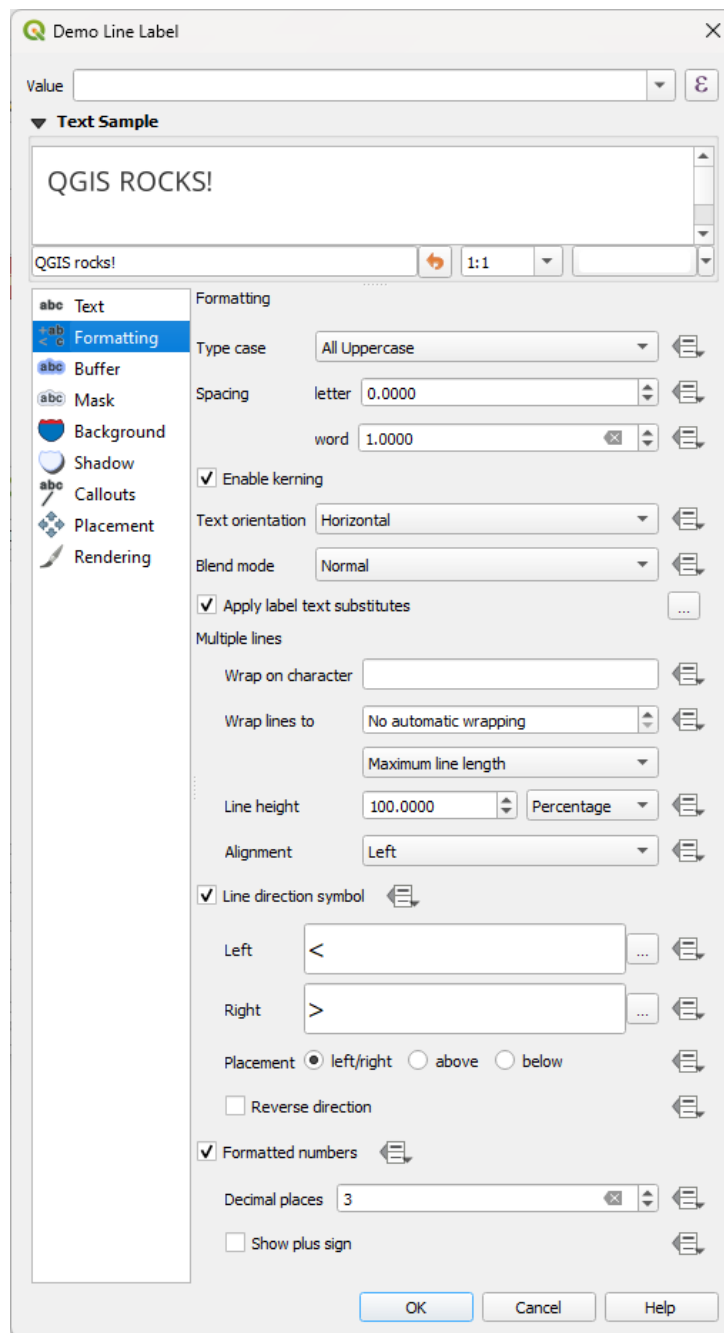




Abb. 14.25: Label settings - Formatting tab

In the  **Formatting** tab, you can:

- Use the *Type case* option to change the capitalization style of the text. You have the possibility to render the text as:
  - *No change*
  - *All uppercase*
  - *All lowercase*
  - *Title case*: modifies the first letter of each word into capital, and turns the other letters into lower case if

the original text is using a single type case. In case of mixed type cases in the text, the other letters are left untouched.




- *Force first letter to capital*: modifies the first letter of each word into capital and leaves the other letters in the text untouched.
- Under *Spacing*, change the space between words and between individual letters.
- *Stretch* ratio: allows text to be horizontally stretched or condensed by a factor. Handy for tweaking the widths of fonts to fit a bit of extra text into labels.
-  *Enable kerning* of the text font
- Set the *Text orientation* which can be *Horizontal* or *Vertical*. It can also be *Rotation-based* when setting a label (e.g., to properly label line features in *parallel* placement mode).
- Use the *Blend mode* option to determine how your labels will mix with the map features below them (more details at *Mischmodi*).
- The  *Apply label text substitutes* option allows you to specify a list of texts to substitute to texts in feature labels (e.g., abbreviating street types). Replacement texts are used when displaying labels on the map. Users can also export and import lists of substitutes to make reuse and sharing easier.
- Configure *Multiple lines*:
  - Set a character that will force a line break in the text with the *Wrap on character* option
  - Set an ideal line size for auto-wrapping using the *Wrap lines to* option. The size can represent either the *Maximum line length* or the *Minimum line length*.
  - Decide the *Line Height*: values can be set to be in *Millimeters*, *Points*, *Pixels*, *Percentage*, or *Inches*. When line height is set to percentage it is the percentage of the default text line spacing of that font family. Typically 1.2 to 1.5 times the text size.
  - Format the *Alignment*: typical values available are *Left*, *Right*, *Justify* and *Center*.

When setting point labels properties, the text alignment can also be *Follow label placement*. In that case, the alignment will depend on the final placement of the label relative to the point. E.g., if the label is placed to the left of the point, then the label will be right aligned, while if it is placed to the right, it will be left aligned.

---

**Bemerkung:** The *Multiple lines* formatting is not yet supported by curve based *label placement*. The options will then be deactivated.

---

- For line labels you can include *Line direction symbol* to help determine the line directions, with symbols to use to indicate the *Left* or *Right*. They work particularly well when used with the *curved* or *Parallel* placement options from the *Placement* tab. There are options to set the symbols position, and to  *Reverse direction*.
- Use the  *Formatted numbers* option to format numeric texts. You can set the number of *Decimal places*. By default, 3 decimal places will be used. Use the  *Show plus sign* if you want to show the plus sign for positive numbers.

## Buffer tab

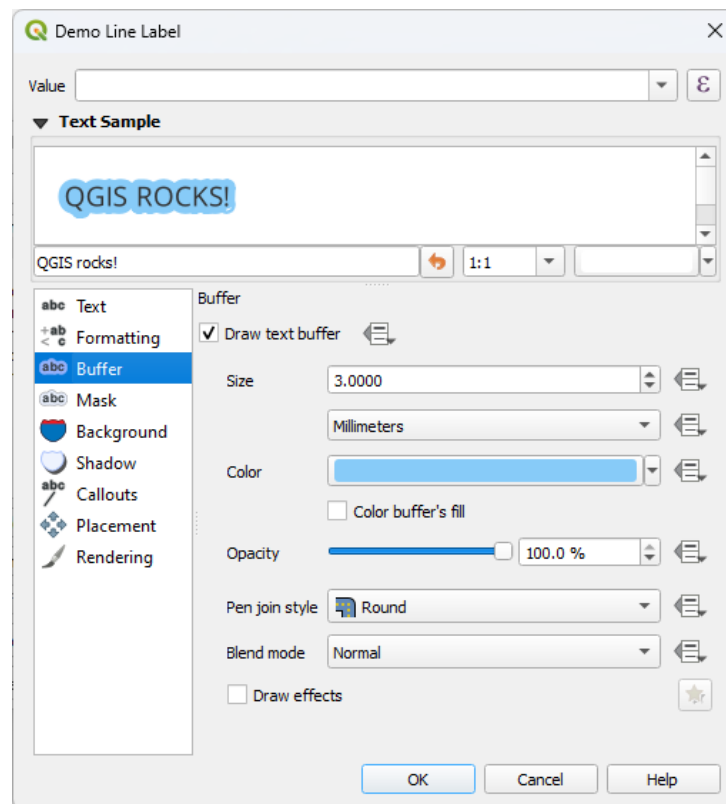









Abb. 14.26: Label settings - Buffer tab

To create a buffer around the label, activate the  *Draw text buffer* checkbox in the  *Buffer* tab. Then you can:

- Set the buffer's *Size* in any *supported unit*
- Select the buffer's *Color*
-  *Color buffer's fill*: The buffer expands from the label's outline, so, if the option is activated, the label's interior is filled. This may be relevant when using partially transparent labels or with non-normal blending modes, which will allow seeing behind the label's text. Unchecking the option (while using totally transparent labels) will allow you to create outlined text labels.
- Define the buffer's *Opacity*
- Apply a *Pen join style*: it can be *Round*, *Miter* or *Bevel*
- Use the *Blend mode* option to determine how your label's buffer will mix with the map components below them (more details at *Mischmodi*).
- Check  *Draw effects* to add advanced  *paint effects* for improving text readability, eg through outer glows and blurs.

## Background tab

The  *Background* tab allows you to configure a shape that stays below each label. To add a background, activate the  *Draw Background* checkbox and select the *Shape* type. It can be:

- a regular shape such as *Rectangle*, *Square*, *Circle* or *Ellipse* using full properties of a *fill symbol*
- an *SVG* symbol from a file, a URL or embedded in the project or style database (*more details*)
- or a *Marker Symbol* you can create or select from the *symbol library*.

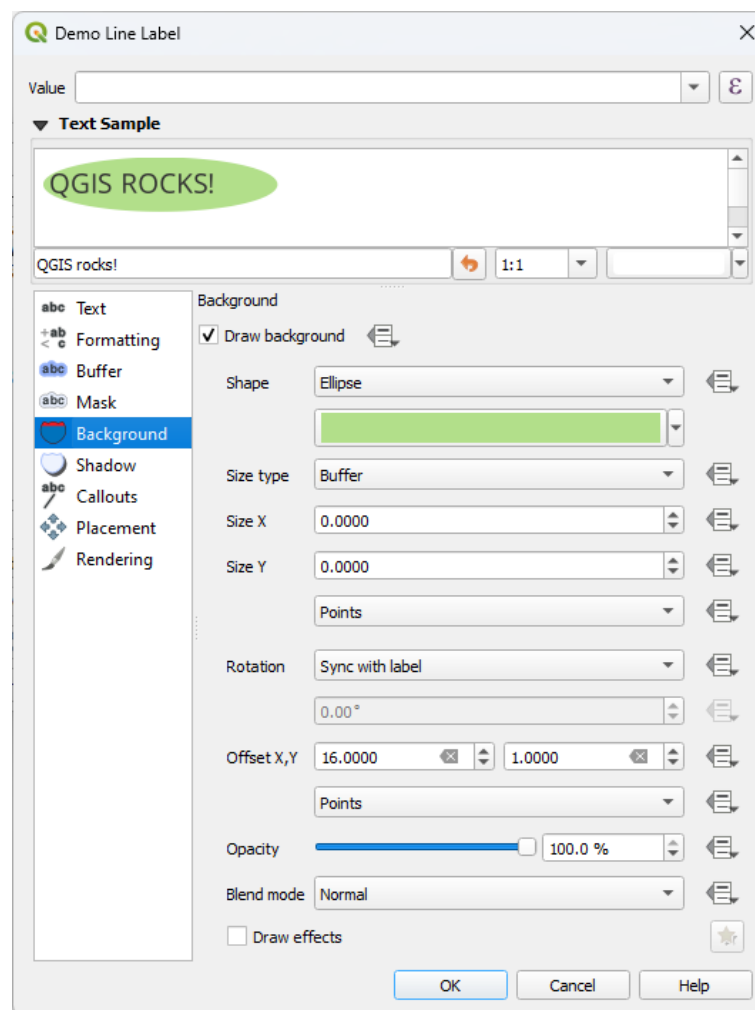




Abb. 14.27: Label settings - Background tab

Depending on the selected shape, you need to configure some of the following properties:

- The *Size type* of the frame, which can be:
  - *Fixed*: using the same size for all the labels, regardless the size of the text
  - or a *Buffer* over the text's bounding box
- The *Size* of the frame in X and Y directions, using any *supported units*
- A *Rotation* of the background, between *Sync with label*, *Offset of label* and *Fixed*. The last two require an angle in degrees.
- An *Offset X,Y* to shift the background item in the X and/or Y directions
- A *Radius X,Y* to round the corners of the background shape (applies to rectangle and square shapes only)

- An *Opacity* of the background
- A *Blend mode* to mix the background with the other items in the rendering (see [Mischmodi](#)).
- For SVG symbol, you can use its default properties (*Load symbol parameters*) or set a custom *Fill color*, *Stroke color* and *Stroke width*.
-  *Draw effects* to add advanced  *paint effects* for improving text readability, eg through outer glows and blurs.

## Shadow tab

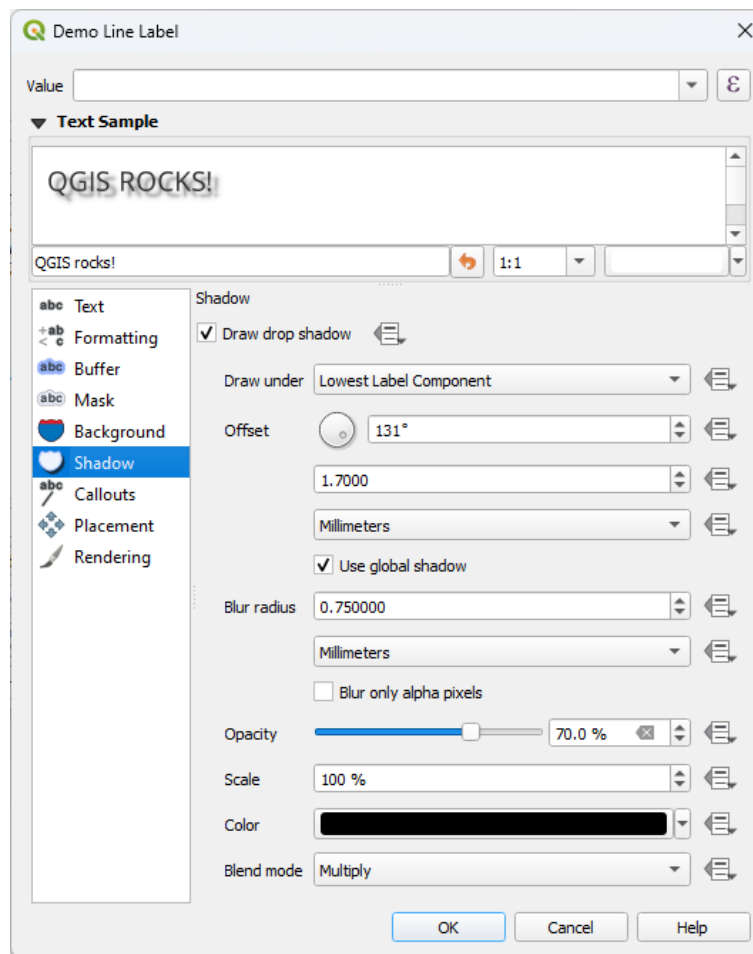





Abb. 14.28: Label settings - Shadow tab

To add a shadow to the text, enable the  *Shadow* tab and activate the  *Draw drop shadow*. Then you can:

- Indicate the item used to generate the shadow with *Draw under*. It can be the *Lowest label component* or a particular component such as the *Text* itself, the *Buffer* or the *Background*.
- Set the shadow's *Offset* from the item being shadowed, ie:
  - The angle: clockwise, it depends on the underlying item orientation
  - The distance of offset from the item being shadowed
  - The units of the offset


If you tick the  *Use global shadow* checkbox, then the zero point of the angle is always oriented to the north and doesn't depend on the orientation of the label's item.

- Influence the appearance of the shadow with the *Blur radius*. The higher the number, the softer the shadows, in the units of your choice.
- Define the shadow's *Opacity*
- Rescale the shadow's size using the *Scale* factor
- Choose the shadow's *Color*
- Use the *Blend mode* option to determine how your label's shadow will mix with the map components below them (more details at [Mischmodi](#)).

### 14.3.2 Configuring interaction with labels

Other than the text formatting settings exposed above, you can also set how labels interact with each others or with the features.

#### Mask tab

The  *Mask* tab allows you to define a mask area around the labels. This feature is very useful when you have overlapping symbols and labels with similar colors, and you want to make the labels visible. A label mask prevents specified features from drawing within the boundary set for the mask. For example, you could set a label mask so that a specified layer does not draw within 2mm of the label, but allow features from another layer to still show. Label masks are similar to label buffers in that they allow control of the legibility of labels that cover other features. The label buffer draws on top of any underlying features, while the label mask selectively stops other layers from drawing.

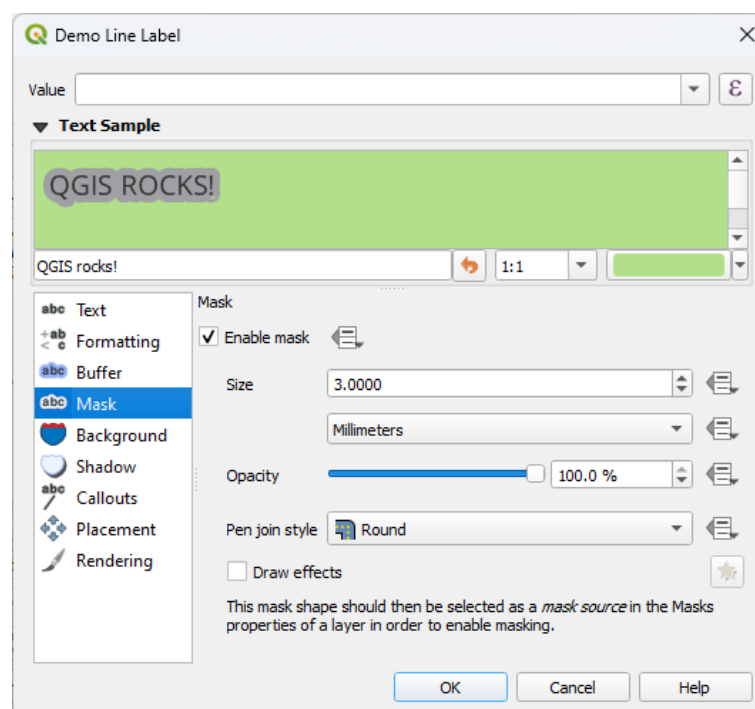






Abb. 14.29: Labels settings - Mask tab (with the text sample showing a green background representing another layer being excluded)

To create masking effects on labels:

1. Activate the  *Enable mask* checkbox in the  tab.
2. Then you can set:



- the mask's *Size* in the *supported units*
  - the *Opacity* of the mask area around the label
  - a *Pen Join Style*
  - *paint effects* through the  *Draw effects* checkbox.
3. Select this mask shape as a mask source in the overlapping layer properties  *Mask* tab (see *Eigenschaften Masken*).

### Callouts tab


A common practice when placing labels on a crowded map is to use **callouts** - labels which are placed outside (or displaced from) their associated feature are identified with a dynamic line connecting the label and the feature. If one of the two endings (either the label or the feature) is moved, the shape of the connector is recomputed.




Abb. 14.30: Labels with various callouts settings


To add a callout to a label, enable the  *Callouts* tab and activate the  *Draw callouts*. Then you can:

1. Select the *Style* of connector, one of:
  - *Simple lines*: a straight line, the shortest path
  - *Manhattan style*: a 90° broken line
  - *Curved lines*: a curved line
  - *Balloons*: a speech bubble surrounding the label and pointing to the feature. It can have rounded corners.
2. For a line-based callout:
  1. Select the *Line style* with full capabilities of a *line symbol* including layer effects, and data-defined settings
  2. If curved, you also define:
    - the percentage of *Curvature* of the connection line
    - and its *Orientation*: starting from the label to the feature, it can be *Clockwise* or *Counter-clockwise*, or *Automatic* (determining an optimal orientation for each label)
  3. Set the *Minimum length* of callout lines

4. Check whether to  *Draw lines to all feature parts* from the feature's label
  5. Set the *Label anchor point*: controls where the connector line should join to the label text. Available options:
    - *Closest point*
    - *Centroid*
    - Fixed position at the edge (*Top left, Top center, Top right, Left middle, Right middle, Bottom left, Bottom center* and *Bottom right*).
  6. Set the *Offset from label area* option: controls the distance from the label anchor point (where the callout line ends). This avoids drawing lines right up against the text.
3. For a balloon callout, you'd need to set:
- the *Fill style* with full capabilities of a *fill symbol* including layer effects, and data-defined settings
  - the *Corner radius* of the speech bubble
  - the *Wedge width*: how large the bubble speech connection with feature's pointer should be
  - the *Margins* around the label's text
4. Set the *Offset from feature* option: controls the distance from the feature (or its anchor point if a polygon) where callout lines end. Eg, this avoids drawing lines right up against the edges of the features.
  5. Set the *Feature anchor point* for the (polygon) feature (the end point of the connector line). Available options:
    - *Pole of inaccessibility*
    - *Point on exterior*
    - *Point on surface*
    - *Centroid*
  6. Set the *Blend mode*: controls the *blending* of the callout.

Under the *Data defined placement* group, coordinates of the *Origin* (on the label side) and/or *Destination* (on the feature side) points of the callout can be controlled. Callouts can also be controlled manually by using the  tool in the *Labeling Toolbar*. The start and end points of each callout can be moved this way. The nodes should be highlighted when the mouse pointer is nearby. If needed the *Shift Key* can be held during the movement. This will snap the point in a way that the angle between the two callout points increments by 15 degrees.

## Placement tab

Choose the  *Placement* tab for configuring label placement and labeling priority. Note that the placement options differ according to the type of vector layer, namely point, line or polygon, and are affected by the global *PAL setting*.

## Placement for point layers

Point labels placement modes available are:

- *Cartographic*: point labels are generated with a better visual relationship with the point feature, following ideal cartographic placement rules. Labels can be placed:
  - at a set *Distance* in *supported units*, either from the point feature itself or from the bounds of the symbol used to represent the feature (set in *Distance offset from*). The latter option is especially useful when the symbol size isn't fixed, e.g. if it's set by a data defined size or when using different symbols in a *categorized* renderer.



- following a *Position priority* that can be customized or set for an individual feature using a data defined list of prioritised positions. This also allows only certain placements to be used, so e.g. for coastal features you can prevent labels being placed over the land.

By default, cartographic mode placements are prioritised in the following order (respecting the [guidelines from Krygier and Wood \(2011\)](#) and other cartographic textbooks):

1. top right
  2. top left
  3. bottom right
  4. bottom left
  5. middle right
  6. middle left
  7. top, slightly right
  8. bottom, slightly left.
- *Around Point*: labels are placed in a circle around the feature. equal radius (set in *Distance*) circle around the feature. The placement priority is clockwise from the „top right“. The position can be constrained using the data-defined *Quadrant* option.
  - *Offset from Point*: labels are placed at an *Offset X,Y* distance from the point feature, in various units, or preferably over the feature. You can use a data-defined *Quadrant* to constrain the placement and can assign a *Rotation* to the label.

## Placement for line layers

Label modes for line layers include:

- *Parallel*: draws the label parallel to a generalised line representing the feature, with preference for placement over straighter portions of the line. You can define:
  - *Allowed positions*: *Above line*, *On line*, *Below line* and *Line orientation dependent position* (placing the label at the left or the right of the line). It's possible to select several options at once. In that case, QGIS will look for the optimal label position.
  - *Distance* between the label and the line
- *Curved*: draws the label following the curvature of the line feature. In addition to the parameters available with the *Parallel* mode, you can set the *Maximum angle between curved characters*, either inside or outside.
- *Horizontal*: draws labels horizontally along the length of the line feature.

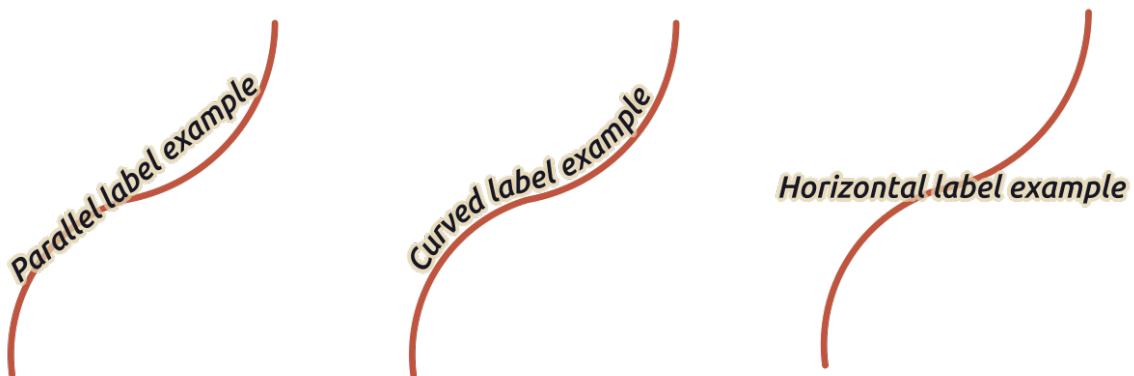






Abb. 14.31: Label placement examples for lines

Next to placement modes, you can set:

- *Repeating Labels Distance* to display multiple times the label over the length of the feature. The distance can be in *Millimeters, Points, Pixels, Meters at scale, Map Units and Inches*.
- A *Label Overrun Distance* (not available for horizontal mode): specifies the maximal allowable distance a label may run past the end (or start) of line features. Increasing this value can allow for labels to be shown for shorter line features.
- *Label Anchoring*: controls the placement of the labels along the line feature they refer to. Click on *Settings ...* to choose:
  - the position along the line (as a ratio) which labels will be placed close to. It can be data-defined and possible values are:
    - \*  *Center of Line*
    - \*  *Start of Line*
    - \*  *End of Line*
    - \* or  *Custom....*
  - *Clipping*: Determines how the label placement on a line is calculated. By default only the visible extent of the line is used but the whole extent can be used to have more consistent results.
  - *Anchor text*: controls which part of the text (start, center or end) will line up with the anchor point. Using *Automatic* anchoring means that:
    - \* For labels anchored near the start of the line (0-25%), the anchor placement will be the **start** of the label text
    - \* For labels anchored near the end of the line (75-100%), the anchor placement will be the **end** of the label text
    - \* For labels anchored near the center of the line (25-75%), the anchor placement will be the **center** of the label text
  - *Placement Behavior*: use *Preferred Placement Hint* to treat the label anchor only as a hint for the label placement. By choosing *Strict*, labels are placed exactly on the label anchor.

## Placement for polygon layers

You can choose one of the following modes for placing labels of polygons:

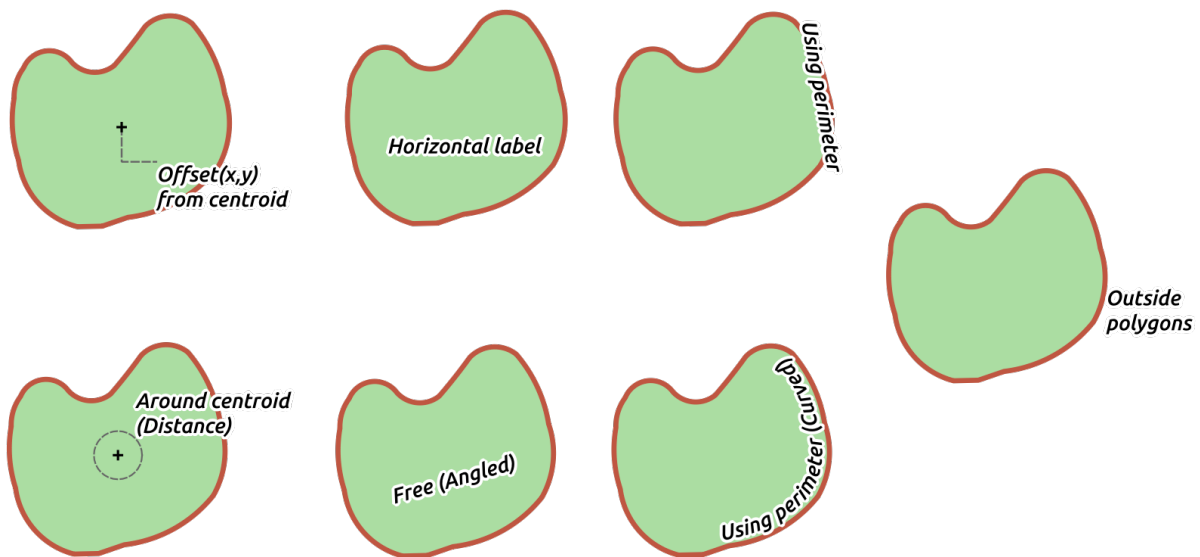


Abb. 14.32: Label placement examples for polygons

- *Offset from Centroid*: labels are placed over the feature centroid or at a fixed *Offset X, Y* distance (in *supported units*) from the centroid. The reference centroid can be determined based on the part of the polygon rendered in the map canvas (*visible polygon*) or the *whole polygon*, no matter if you can see it. You can also:
  - force the centroid point to lay inside their polygon
  - place the label within a specific quadrant
  - assign a rotation
  - *Allow placing labels outside of polygons* when it is not possible to place them inside the polygon. Thanks to data-defined properties, this makes possible to either allow outside labels, prevent outside labels, or force outside labels on a feature-by-feature basis.
- *Around Centroid*: places the label within a preset distance around the centroid, with a preference for the placement directly over the centroid. Again, you can define whether the centroid is the one of the *visible polygon* or the *whole polygon*, and whether to force the centroid point inside the polygon.
- *Horizontal*: places at the best position a horizontal label inside the polygon. The preferred placement is further from the edges of the polygon. It's possible to *Allow placing labels outside of polygons*.
- *Free (Angled)*: places at the best position a rotated label inside the polygon. The rotation respects the polygon's orientation and the preferred placement is further from the edges of the polygon. It's possible to *Allow placing labels outside of polygons*.
- *Using Perimeter*: draws the label parallel to a generalised line representing the polygon boundary, with preference for straighter portions of the perimeter. You can define:
  - *Allowed positions*: *Above line*, *On line*, *Below line* and *Line orientation dependent position* (placing the label at the left or the right of the polygon's boundary). It's possible to select several options at once. In that case, QGIS will look for the optimal label position.
  - *Distance* between the label and the polygon's outline
  - the *Repeating Labels Distance* to display multiple times the label over the length of the perimeter.
- *Using Perimeter (Curved)*: draws the label following the curvature of the polygon's boundary. In addition to the parameters available with the *Using Perimeter* mode, you can set the *Maximum angle between curved characters polygon*, either inside or outside.

- *Outside Polygons*: always places labels outside the polygons, at a set *Distance*


### Common placement settings

Some label placement settings are available for all layer geometry types:

### Geometry Generator

The *Geometry Generator* section allows a user to alter the underlying geometry used to place and render the label, by using *expressions*. This can be useful to perform displacement of the geometry dynamically or to convert it to another geometry (type).

In order to use the geometry generator:

1. Check the  *Geometry generator* option
2. Enter the expression generating the geometry to rely on
3. If relevant, select the geometry type of the expression output: the label geometry-based settings such as placement or rendering are updated to match the new geometry type capabilities.

Some use cases include:

- Use a geometry which is saved in another field „label\_position“
- Use the *generated geometry* from the symbology also for labeling
- Use the @map\_scale variable to calculate distances / sizes be zoom level independent.
- Combined with the curved placement mode, creates a circular label around a point feature:

```
exterior_ring(make_circle($geometry, 20))
```

- Add a label at the start and the end of a line feature:

```
collect_geometries( start_point($geometry), end_point($geometry) )
```

- Rely on a smoothed line of a river to get more room for label placement:

```
smooth( $geometry, iterations:=30, offset:=0.25, min_length:=10 )
```

### Data Defined

The *Data Defined* group provides direct control on labels placement, on a feature-by-feature basis. It relies on their attributes or an expression to set:

- the *X* and *Y* coordinate
- the text alignment over the custom position set above:
  - *Horizontal*: it can be **Left**, **Center** or **Right**
  - the text *Vertical*: it can be **Bottom**, **Base**, **Half**, **Cap** or **Top**
- the text *Rotation*. Different units can be defined for the labeling rotation (e.g. degrees, minutes of arc, turns). Check the *Preserve data rotation values* entry if you want to keep the rotation value in the associated field and apply it to the label, whether the label is pinned or not. If unchecked, unpinning the label rotation is reset and its value cleared from the attribute table.

---

**Bemerkung:** Data-defined rotation with polygon features is currently supported only with the *Around centroid* placement mode.

---

---

**Bemerkung:** Expressions can not be used in combination with the labels map tools (ie the *Rotate label* and *Move label* tools) to *data-define* labels placement. The widget will be reset to the corresponding *auxiliary storage field*.

---

## Priority


In the *Priority* section you can define the placement priority rank of each label, ie if there are different diagrams or labels candidates for the same location, the item with the higher priority will be displayed and the others could be left out.


The priority rank is also used to evaluate whether a label could be omitted due to a greater weighted *obstacle feature*.

## Obstacles

In some contexts (eg, high density labels, overlapping features...), the labels placement can result in labels being placed over unrelated features.


An obstacle is a feature over which QGIS avoids placing other features' labels or diagrams. This can be controlled from the *Obstacles* section:

1. Activate the  *Features act as obstacles* option to decide that features of the layer should act as obstacles for any label and diagram (including items from other features in the same layer).

Instead of the whole layer, you can select a subset of features to use as obstacles, using the  Data-defined override control next to the option.

2. Use the *Settings* button to tweak the obstacle's weighting.
  - For every potential obstacle feature you can assign an *Obstacle weight*: any *label* or *diagram* whose placement priority rank is greater than this value can be placed over. Labels or diagrams with lower rank will be omitted if no other placement is possible.  
This weighting can also be data-defined, so that within the same layer, certain features are more likely to be covered than others.
  - For polygon layers, you can choose the kind of obstacle the feature is:
    - **over the feature's interior**: avoids placing labels over the interior of the polygon (prefers placing labels totally outside or just slightly inside the polygon)
    - or **over the feature's boundary**: avoids placing labels over the boundary of the polygon (prefers placing labels outside or completely inside the polygon). This can be useful for layers where the features cover the whole area (administrative units, categorical coverages, ...). In this case, it is impossible to avoid placing labels within these features, and it looks much better when placing them over the boundaries between features is avoided.

## Rendering tab

In the  *Rendering* tab, you can tune when the labels can be rendered and their interaction with other labels and features.

### Label options

Under *Label options*:

- You find the *scale-based* and the *Pixel size-based* visibility settings.
- The *Label z-index* determines the order in which labels are rendered, as well in relation with other feature labels in the layer (using data-defined override expression), as with labels from other layers. Labels with a higher z-index are rendered on top of labels (from any layer) with lower z-index.

Additionally, the logic has been tweaked so that if two labels have matching z-indexes, then:

- if they are from the same layer, the smaller label will be drawn above the larger label
- if they are from different layers, the labels will be drawn in the same order as their layers themselves (ie respecting the order set in the map legend).

---

**Bemerkung:** This setting doesn't make labels to be drawn below the features from other layers, it just controls the order in which labels are drawn on top of all the layers' features.

---

- *Allow inferior fallback placements*: By default QGIS tries to render labels at their best placement, following your settings. Check this mode to allow features to fallback to worse placement options when there's no other choice, e.g. when a line is too short to fit a curved label text then the label may be placed horizontally just over the feature's center point.
- With data-defined expressions in *Show label* and *Always Show* you can fine tune which labels should be rendered.
- Allow to *Show upside-down labels*: alternatives are **Never**, **when rotation defined** or **always**.
- The *Overlapping labels* group allows you to control whether overlapping labels are permitted for features in the layer and how each of them should be handled:
  - *Never overlap*: never ever place overlapping labels for the layer, even if it means some labels will be missing
  - *Allow overlaps if required*: if the label can't otherwise be placed, draw an overlapping label. This mode will cause the label to be moved to a less ideal placement if possible, e.g. moving the label further from the center of a line or polygon, IF doing so will avoid overlapping labels. But if there's no other positions possible, then draw the label overlapping.
  - *Allow overlaps without penalty*: It doesn't matter at all if the label overlaps other labels or obstacles, that's fine to do and the best placement (e.g most central placement) should always be used even if an alternate further placement is possible which avoids overlaps entirely.

Allowing both overlapping labels and fallback placements options will guarantee that all features in the layer are labeled... not necessarily at their best rendering!



### Feature options

Under *Feature options*:

- You can choose to *Label every part of a multi-part features* and *Limit number of features to be labeled to*.
- Both line and polygon layers offer the option to set a minimum size for the features to be labeled, using *Suppress labeling of features smaller than*.
- For polygon features, you can also filter the labels to show according to whether they completely fit within their feature or not.
- For line features, you can choose to *Merge connected lines to avoid duplicate labels*, rendering a quite airy map in conjunction with the *Distance* or *Repeat* options in the *Placement* tab.

## 14.4 Creating 3D Symbols

The *Style Manager* helps you create and store 3D symbols for every geometry type to render in the *3D map view*.

As of the other items, enable the  3D Symbols tab and expand the  button menu to create:

- 3D point symbols
- 3D line symbols
- 3D polygon symbols

### 14.4.1 Point Layers

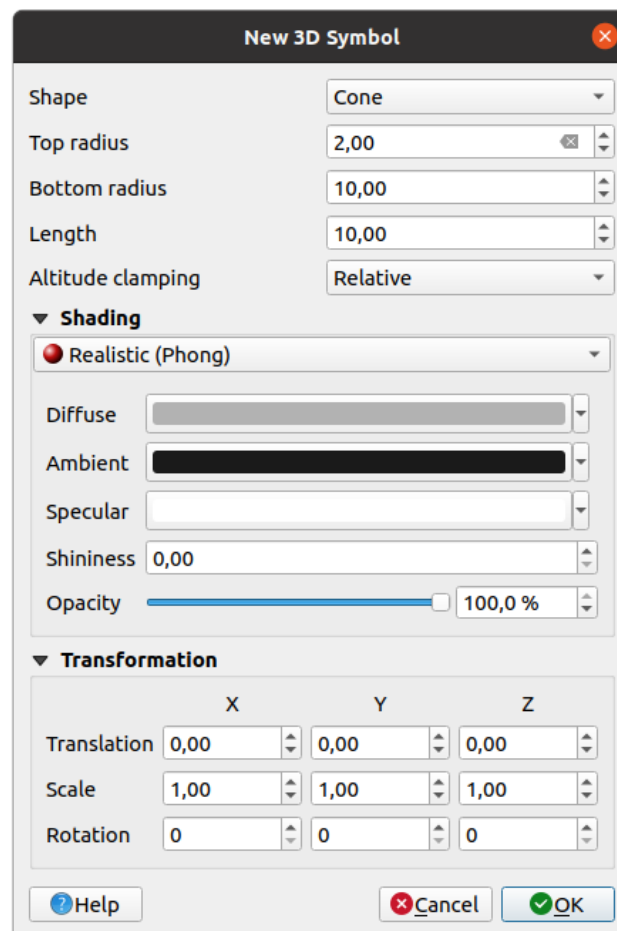


Abb. 14.33: Properties of a 3D point symbol

- You can define different types of 3D *Shape* to use for point symbols. They are mainly defined by their dimensions whose unit refers to the CRS of the project. Available types are:
  - *Sphere* defined by a *Radius*
  - *Cylinder* defined by a *Radius* and *Length*
  - *Cube* defined by a *Size*
  - *Cone* defined by a *Top radius*, a *Bottom radius* and a *Length*
  - *Plane* defined by a *Size*

- *Torus* defined by a *Radius* and a *Minor radius*
- *3D Model*, using a 3D model file: supported formats include wavefront `.obj`, `.glTF` and `.fbx`. Models can be a file on disk, a remote URL or *embedded in the project*. Community-created models are shared on the QGIS Hub at <https://plugins.qgis.org/wavefronts>.
- *Billboard*, defined by the *Billboard height* and the *Billboard symbol* (usually based on a *marker symbol*). The symbol will have a stable size. Convenient for visualizing 3D point clouds Shapes.
- The *Altitude clamping* can be set to *Absolute*, *Relative* or *Terrain*. The *Absolute* setting can be used when height values of the 3d vectors are provided as absolute measures from 0. *Relative* and *Terrain* add given elevation values to the underlying terrain elevation.
- The *shading* properties can be defined.
- Under the *Transformations* frame, you can apply affine transformation to the symbol:
  - *Translation* to move objects in x, y and z axis.
  - *Scale* to resize the 3D shapes
  - *Rotation* around the x-, y- and z-axis.

## 14.4.2 Line layers

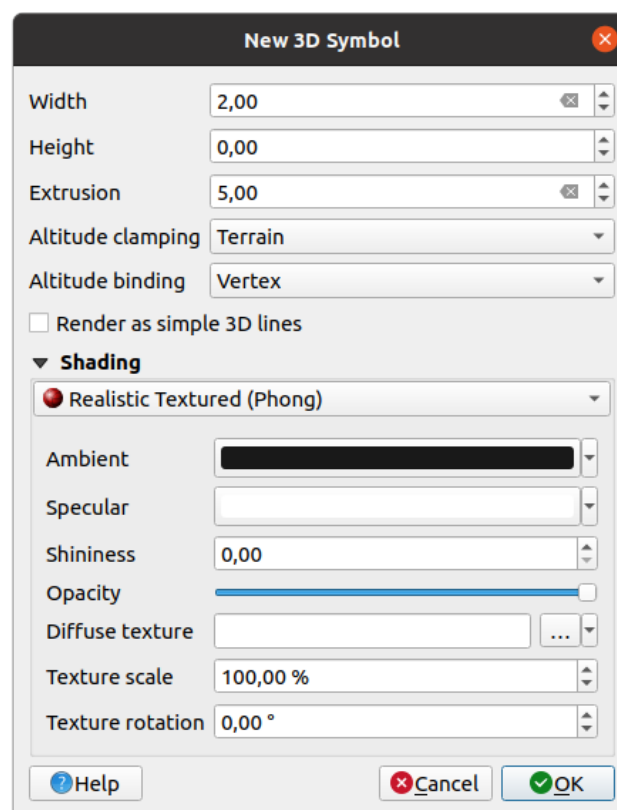


Abb. 14.34: Properties of a 3D line symbol

- Beneath the *Width* and *Height* settings you can define the *Extrusion* of the vector lines. If the lines do not have z-values, you can define the 3d volumes with this setting.
- With the *Altitude clamping* you define the position of the 3D lines relative to the underlying terrain surface, if you have included raster elevation data or other 3D vectors.



- The *Altitude binding* defines how the feature is clamped to the terrain. Either every *Vertex* of the feature will be clamped to the terrain or this will be done by the *Centroid*.
- It is possible to ☒ *Render as simple 3D lines*.
- The *shading* properties can be defined.

### 14.4.3 Polygon Layers

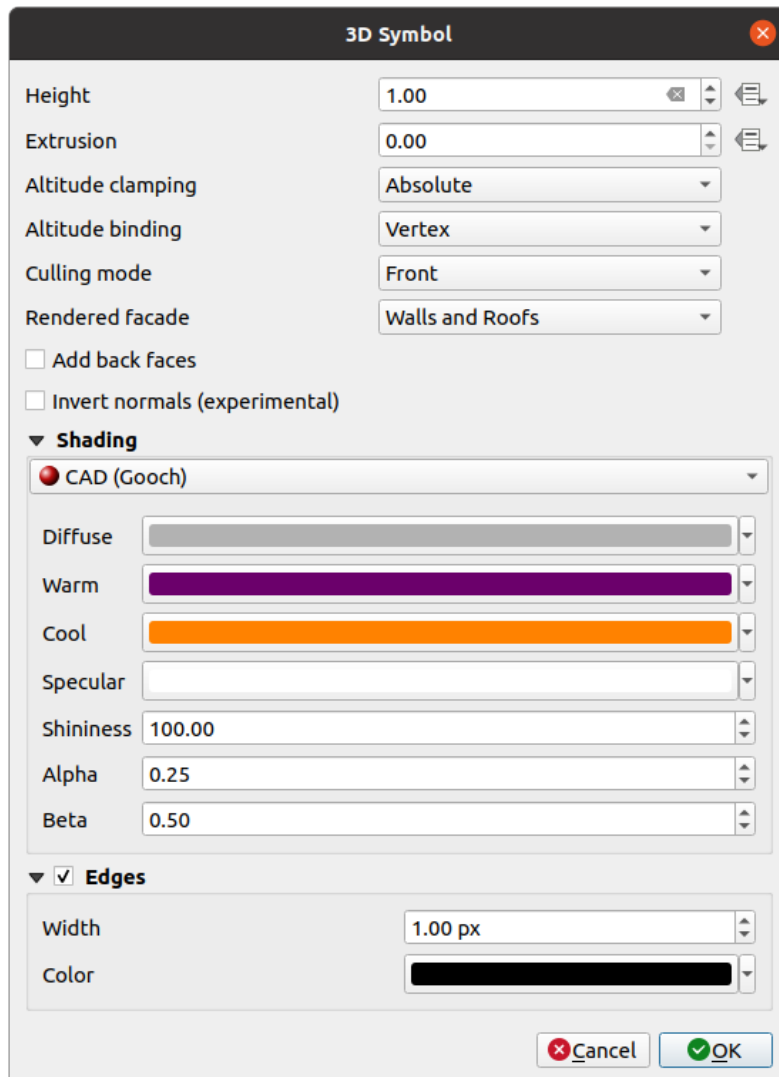




Abb. 14.35: Properties of a 3D polygon symbol

- As for the other ones, *Height* can be defined in CRS units. You can also use the  button to overwrite the value with a custom expression, a variable or an entry of the attribute table
- Again, *Extrusion* is possible for missing z-values. Also for the extrusion you can use the  button in order to use the values of the vector layer and have different results for each polygon:

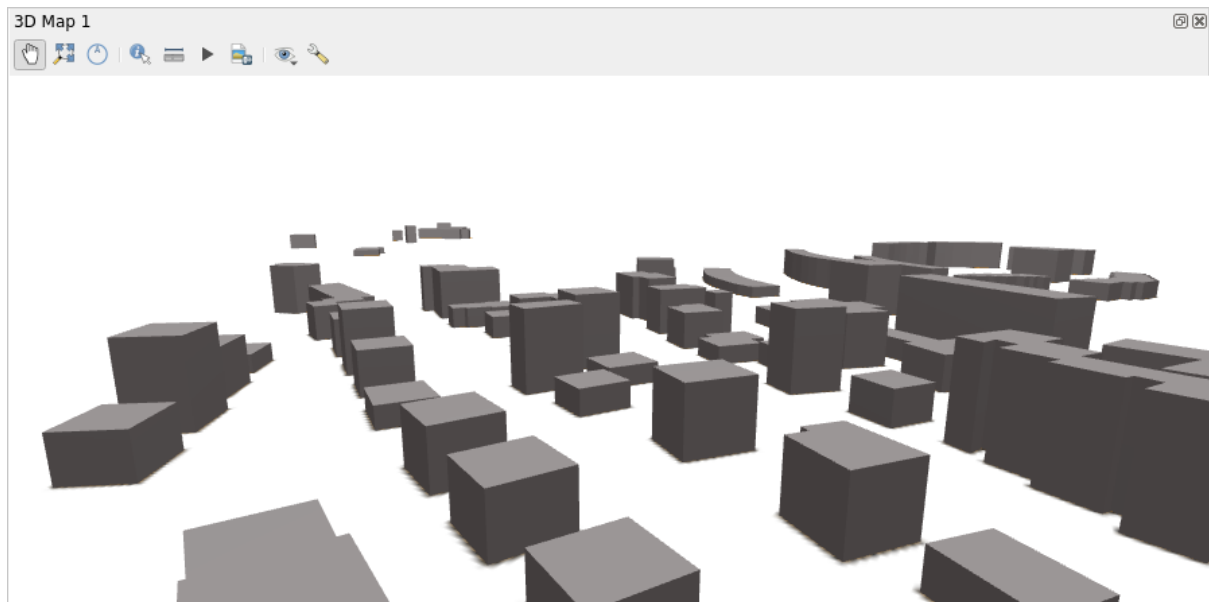


Abb. 14.36: Data Defined Extrusion

- The *Altitude clamping*, *Altitude binding* can be defined as explained above.
- The *Culling mode* to apply to the symbol; it can be:
  - *No Culling*: this can help to avoid seemingly missing surfaces when polygonZ/multipatch data do not have consistent ordering of vertices (e.g. all clock-wise or counter clock-wise)
  - *Front*
  - or *Back*
- The *Rendered facade* determines the faces to display. Possible values are *No facades*, *Walls*, *Roofs*, or *Walls and roofs*
- ☒ *Add back faces*: for each triangle, creates both front and back face with correct normals - at the expense of increased number of vertex data. This option can be used to fix shading issues (e.g., due to data with inconsistent order of vertices).
- ☒ *Invert normals (experimental)*: can be useful for fixing clockwise/counter-clockwise face vertex orders
- The *shading* properties can be defined.
- Display of the ☒ *Edges* of the symbols can be enabled and assigned a *Width* and *Color*.

---

#### Hinweis: Combination for best rendering of 3D data

*Culling mode*, *Add back faces* and *Invert normals* are all meant to fix the look of 3D data if it does not look right. Typically when loading some data, it is best to first try `culling mode=back` and `add back faces=disabled` - it is the most efficient. If the rendering does not look correct, try `add back faces=enabled` and keep `culling mode=no culling`. Other combinations are more advanced and useful only in some scenarios based on how mixed up is the input dataset.

---

#### 14.4.4 Shading the texture

Shading helps you reveal 3d details of objects which may otherwise be hidden due to the scene's lighting. Ultimately, it's an easier material to work with as you don't need to worry about setting up appropriate scene lighting in order to visualise features.

Various techniques of shading are used in QGIS and their availability depends on the geometry type of the symbol:

- *Realistic (Phong)*: describes the way a surface reflects light as a combination of the *Diffuse* reflection of rough surfaces with the *Specular* reflection of shiny surfaces (*Shininess*). It also includes an *Ambient* option to account for the small amount of light that is scattered about the entire scene. Use the *Opacity* slider to render semi-transparent objects in 3D. Read more at [https://en.wikipedia.org/wiki/Phong\\_reflection\\_model#Description](https://en.wikipedia.org/wiki/Phong_reflection_model#Description)
- *Realistic Textured (Phong)*: same as the *Realistic (Phong)* except that an image is used as *Diffuse Texture*. The image can be a file on disk, a remote URL or *embedded in the project*. The *Texture scale* and *Texture rotation* are required. Use the *Opacity* slider to render semi-transparent objects in 3D.
- *CAD (Gooch)*: this technique allows shading to occur only in mid-tones so that edge lines and highlights remain visually prominent. Along with the *Diffuse*, *Specular*, *Shininess* options, you need to provide a *Warm* color (for surface facing toward the light) and a *Cool* color (for the ones facing away). Also, the relative contributions to the cool and warm colors by the diffuse color are controlled by *Alpha* and *Beta* properties respectively. See also [https://en.wikipedia.org/wiki/Gooch\\_shading](https://en.wikipedia.org/wiki/Gooch_shading)
- *Embedded Textures* with 3D models shape

#### 14.4.5 Application example

To go through the settings explained above you can have a look at <https://app.merginmaps.com/projects/saber/luxembourg/tree>.



## Verwaltung von Datenquellen

### 15.1 Öffnen von Daten

Als Teil eines Open-Source-Software-Ökosystems baut QGIS auf verschiedenen Bibliotheken auf, die in Kombination mit den eigenen Diensten die Fähigkeiten bieten, viele Formate zu lesen und oft zu schreiben:


- Vector data formats include GeoPackage, GML, GeoJSON, GPX, KML, Comma Separated Values, ESRI formats (Shapefile, Geodatabase...), MapInfo and MicroStation file formats, AutoCAD DWG/DXF, GRASS and many more... Read the complete list of [supported vector formats](#).
- Raster data formats include GeoTIFF, JPEG, ASCII Gridded XYZ, MBTiles, R or Idrisi rasters, GDAL Virtual, SRTM, Sentinel Data, ERDAS IMAGINE, ArcInfo Binary Grid, ArcInfo ASCII Grid, and many more... Read the complete list of [supported raster formats](#).
- Database formats include PostgreSQL/PostGIS, SQLite/Spatialite, Oracle, MS SQL Server, SAP HANA, MySQL...
- Web map and data services (WM(T)S, WFS, WCS, CSW, XYZ tiles, ArcGIS services, ...) are also handled by QGIS providers. See [Arbeiten mit OGC / ISO-Protokollen](#) for more information about some of these.
- You can read supported files from archived folders and use QGIS native formats such as QML files (*QML - Das QGIS-Stil-Dateiformat*) and virtual and memory layers.

More than 80 vector and 140 raster formats are supported by [GDAL](#) and QGIS native providers.

---

**Bemerkung:** Not all of the listed formats may work in QGIS for various reasons. For example, some require external proprietary libraries, or the GDAL/OGR installation of your OS may not have been built to support the format you want to use. To see the list of available formats, run the command line `ogrinfo --formats` (for vector) and `gdalinfo --formats` (for raster), or check the *Settings ► Options ► GDAL* menu in QGIS.

---

In QGIS, depending on the data format, there are different tools to open a dataset, mainly available in the *Layer ► Add Layer ►* menu or from the *Manage Layers* toolbar (enabled through *View ► Toolbars* menu). However, all these tools point to a unique dialog, the *Data Source Manager* dialog, that you can open with the  Open Data Source Manager button, available on the *Data Source Manager Toolbar*, or by pressing `Ctrl+L`. The *Data Source Manager* dialog ([Abb. 15.1](#)) offers a unified interface to open file-based data as well as databases or web services supported by QGIS.

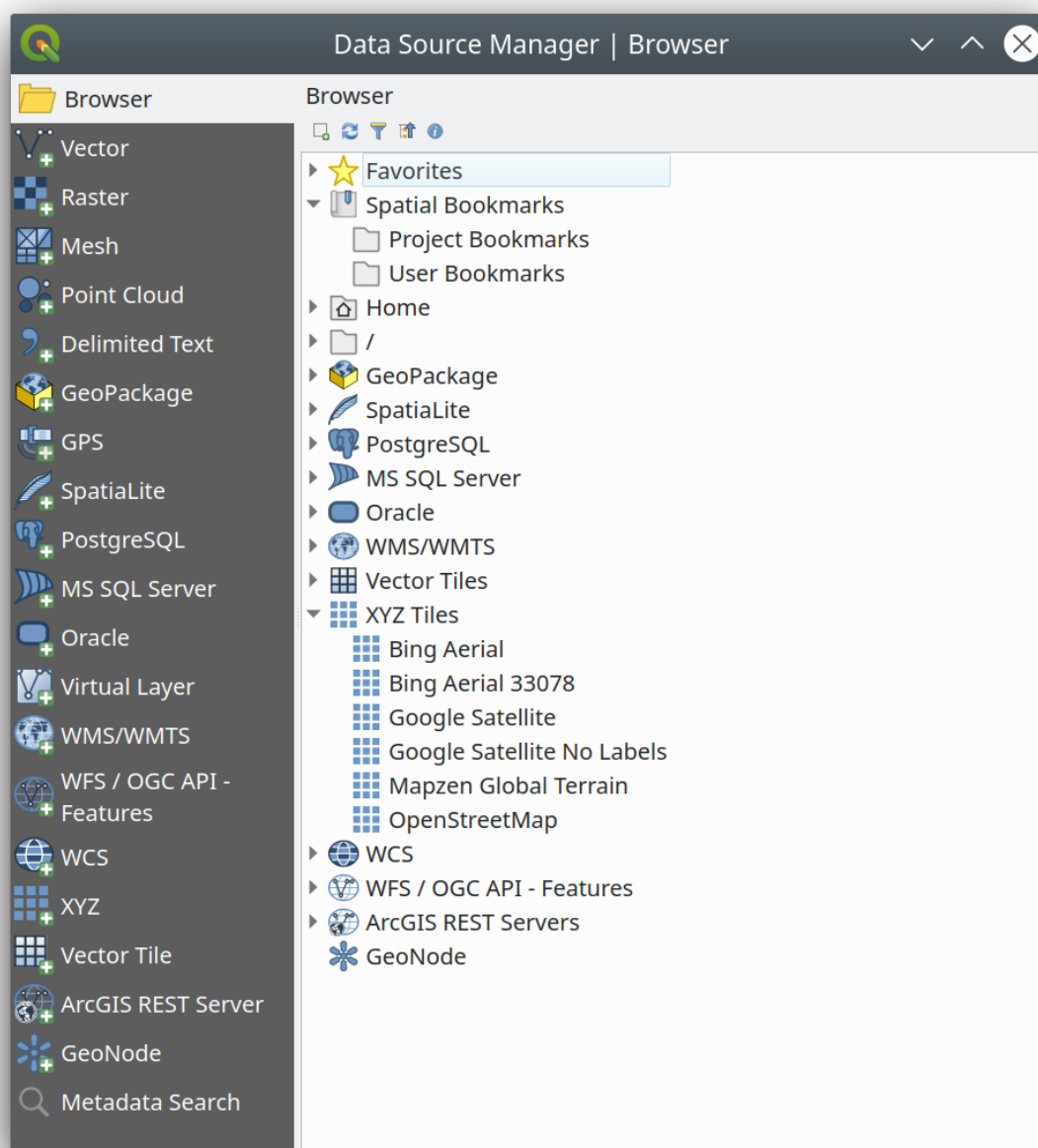



Abb. 15.1: QGIS Data Source Manager dialog

Beside this main entry point, you also have the  *DB Manager* plugin that offers advanced capabilities to analyze and manipulate connected databases. More information on DB Manager capabilities can be found in [DB Manager Plugin](#).

There are many other tools, native or third-party plugins, that help you open various data formats.

This chapter will describe only the tools provided by default in QGIS for loading data. It will mainly focus on the *Data Source Manager* dialog but more than describing each tab, it will also explore the tools based on the data provider or format specificities.

## 15.1.1 The Browser Panel






The *Browser* is one of the main ways to quickly and easily add your data to projects. It's available as:

- a *Data Source Manager* tab, enabled pressing the  Open Data Source Manager button (Ctrl+L);
- as a QGIS panel you can open from the menu *View ► Panels* (or  *Settings ► Panels*) or by pressing Ctrl+2.

In both cases, the *Browser* helps you navigate in your file system and manage geodata, regardless the type of layer (raster, vector, table), or the datasource format (plain or compressed files, databases, web services).







### Exploring the Interface

At the top of the Browser panel, you find some buttons that help you to:

-  Add Selected Layers: you can also add data to the map canvas by selecting **Add selected layer(s)** from the layer's context menu;
-  Refresh the browser tree;
-  Filter Browser to search for specific data. Enter a search word or wildcard and the browser will filter the tree to only show paths to matching DB tables, filenames or folders – other data or folders won't be displayed. See the Browser Panel(2) example in [Abb. 15.2](#). The comparison can be case-sensitive or not. It can also be set to:
  - *Normal*: show items containing the search text
  - *Wildcard(s)*: fine tune the search using the ? and/or \* characters to specify the position of the search text
  - *Regular expression*
-  Collapse All the whole tree;
-  Enable/disable properties widget: when toggled on, a new widget is added at the bottom of the panel showing, if applicable, metadata for the selected item.

The entries in the *Browser* panel are organised hierarchically, and there are several top level entries:

1. *Favorites* where you can place shortcuts to often used locations
2. *Spatial Bookmarks* where you can store often used map extents (see [Bookmarking extents on the map](#))
3. *Project Home*: for a quick access to the folder in which (most of) the data related to your project are stored. The default value is the directory where your project file resides.
4. *Home* directory in the file system and the filesystem root directory.
5. Connected local or network drives
6. Then comes a number of container / database types and service protocols, depending on your platform and underlying libraries:

-  *GeoPackage*
-  *SpatiaLite*
-  *PostgreSQL*
-  *SAP HANA*
-  *MS SQL Server*
-  *Oracle*

-  *WMS/WMF*
-  *Vector Tiles*
-  *XYZ Tiles*
-  *WCS*
-  *WFS/OGC API-Features*
-  *ArcGIS REST Server*

### Interacting with the Browser items

The browser supports drag and drop within the browser, from the browser to the canvas and *Layers* panel, and from the *Layers* panel to layer containers (e.g. GeoPackage) in the browser.

Project file items inside the browser can be expanded, showing the full layer tree (including groups) contained within that project. Project items are treated the same way as any other item in the browser, so they can be dragged and dropped within the browser (for example to copy a layer item to a geopackage file) or added to the current project through drag and drop or double click.

The context menu for an element in the *Browser* panel is opened by right-clicking on it.

For file system directory entries, the context menu offers the following:

- *New ►* to create in the selected entry a:
  - *Directory...*
  - *GeoPackage...*
  - *ShapeFile...*
- *Add as a Favorite*: favorite folders can be renamed (*Rename favorite...*) or removed (*Remove favorite*) any time.
- *Hide from Browser*: hidden folders can be toggled to visible from the *Settings ► Options ► Data Sources ► Hidden browser paths* setting
- *Fast Scan this Directory*
- *Open Directory*
- *Open in Terminal*
- *Eigenschaften...*
- *Directory Properties...*

For leaf entries that can act as layers in the project, the context menu will have supporting entries. For example, for non-database, non-service-based vector, raster and mesh data sources:

- *Layer exportieren ► in Datei...*
- *Add Layer to Project*
- *Layer Properties*
- *Manage ► Rename „<name of file>“... or Delete „<name of file>“...*
- *Show in Files*
- *File Properties*

In the *Layer properties* entry, you will find (similar to what you will find in the *vector* and *raster* layer properties once the layers have been added to the project):



- *Metadata* for the layer. Metadata groups: *Information from provider* (if possible, *Path* will be a hyperlink to the source), *Identification*, *Extent*, *Access*, *Fields* (for vector layers), *Bands* (for raster layers), *Contacts*, *Links* (for vector layers), *References* (for raster layers), *History*.
- A *Preview* panel
- The attribute table for vector sources (in the *Attributes* panel).

To add a layer to the project using the *Browser*:

1. Enable the *Browser* as described above. A browser tree with your file system, databases and web services is displayed. You may need to connect databases and web services before they appear (see dedicated sections).
2. Find the layer in the list.
3. Use the context menu, double-click its name, or drag-and-drop it into the *map canvas*. Your layer is now added to the *Layers panel* and can be viewed on the map canvas.

---

**Tip:** Open a QGIS project directly from the browser

You can also open a QGIS project directly from the Browser panel by double-clicking its name or by drag-and-drop into the map canvas.

---

Once a file is loaded, you can zoom around it using the map navigation tools. To change the style of a layer, open the *Layer Properties* dialog by double-clicking on the layer name or by right-clicking on the name in the legend and choosing *Properties* from the context menu. See section *Eigenschaften Symbolisierung* for more information on setting symbology for vector layers.

Right-clicking an item in the browser tree helps you to:

- for a file or a table, display its metadata or open it in your project. Tables can even be renamed, deleted or truncated.
- for a folder, bookmark it into your favourites or hide it from the browser tree. Hidden folders can be managed from the *Settings ► Options ► Data Sources* tab.
- manage your *spatial bookmarks*: bookmarks can be created, exported and imported as XML files.
- create a connection to a database or a web service.
- refresh, rename or delete a schema.

You can also import files into databases or copy tables from one schema/database to another with a simple drag-and-drop. There is a second browser panel available to avoid long scrolling while dragging. Just select the file and drag-and-drop from one panel to the other.

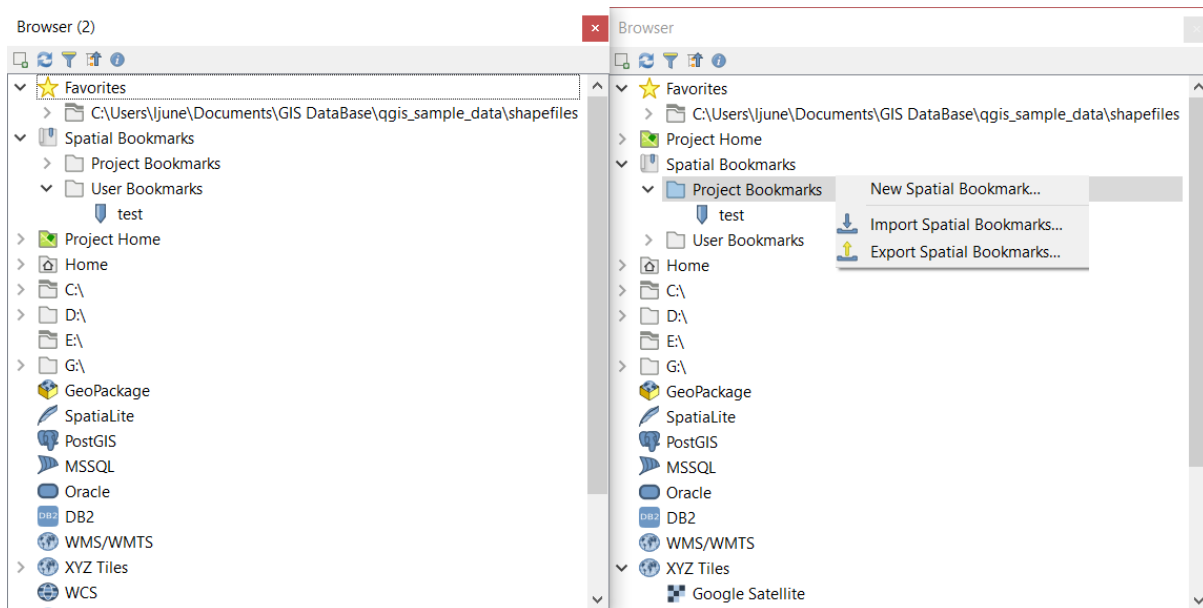



Abb. 15.2: QGIS Browser panels side-by-side

### **Tipp:** Add layers to QGIS by simple drag-and-drop from your OS file browser

You can also add file(s) to the project by drag-and-dropping them from your operating system file browser to the *Layers Panel* or the map canvas.

## 15.1.2 The DB Manager

The *DB Manager* Plugin is another tool for integrating and managing spatial database formats supported by QGIS (PostGIS, SpatiaLite, GeoPackage, Oracle Spatial, MS SQL Server, Virtual layers). It can be activated from the *Plugins ► Manage and Install Plugins...* menu.

The  **DB Manager** Plugin provides several features:

- connect to databases and display their structure and contents
- preview tables of databases
- add layers to the map canvas, either by double-clicking or drag-and-drop.
- add layers to a database from the QGIS Browser or from another database
- create SQL queries and add their output to the map canvas
- create *virtual layers*

More information on DB Manager capabilities is found in *DB Manager Plugin*.

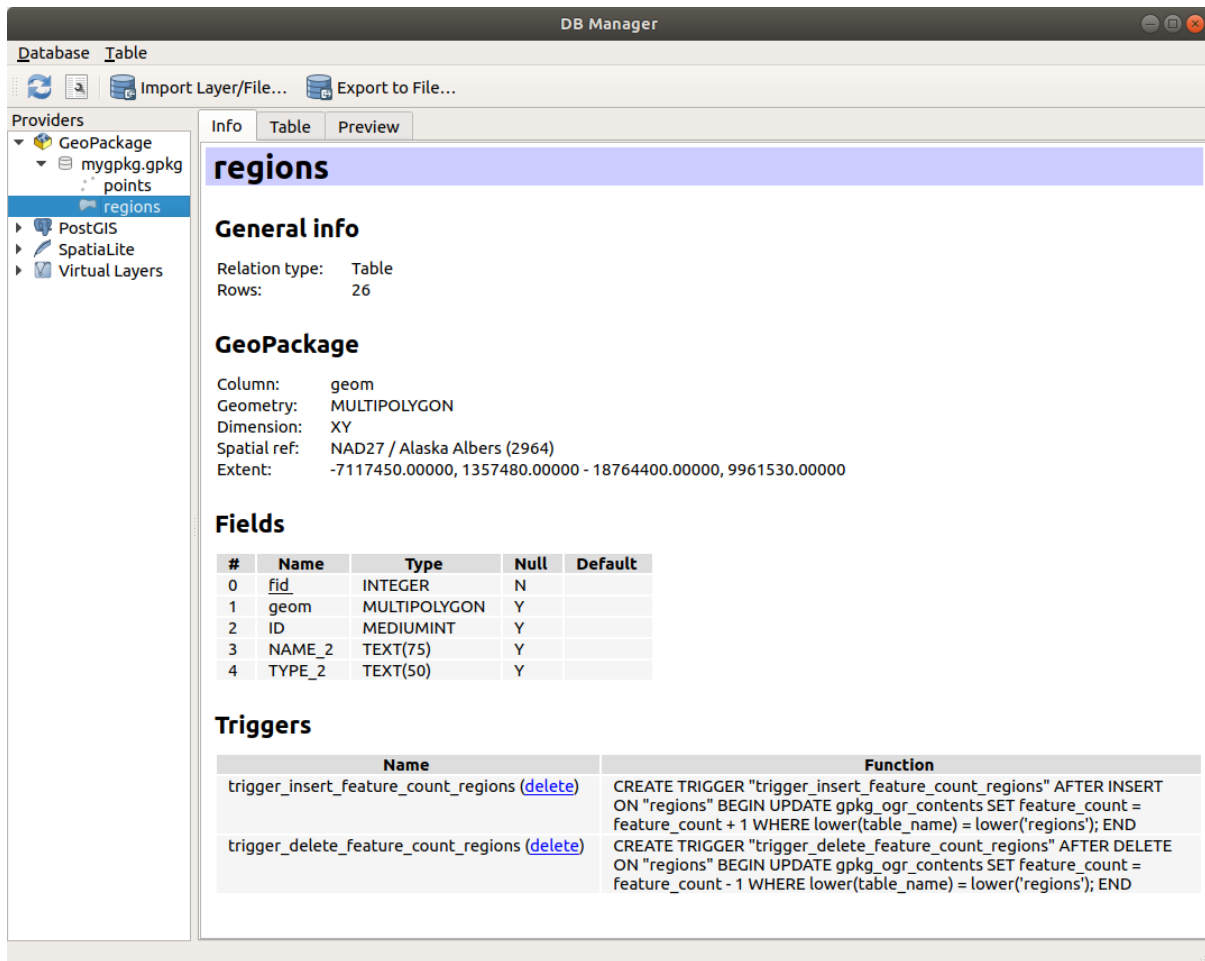


Abb. 15.3: DB Manager dialog

### 15.1.3 Provider-based loading tools

Beside the Browser Panel and the DB Manager, the main tools provided by QGIS to add layers, you'll also find tools that are specific to data providers.




---

**Bemerkung:** Some *external plugins* also provide tools to open specific format files in QGIS.

---

#### Loading a layer from a file

To load a layer from a file:

1. Open the layer type tab in the *Data Source Manager* dialog, ie click the  Open Data Source Manager button (or press **Ctrl+L**) and enable the target tab or:
  - for vector data (like GML, ESRI Shapefile, Mapinfo and DXF layers): press **Ctrl+Shift+V**, select the *Layer ► Add Layer ►  Add Vector Layer* menu option or click on the  Add Vector Layer toolbar button.

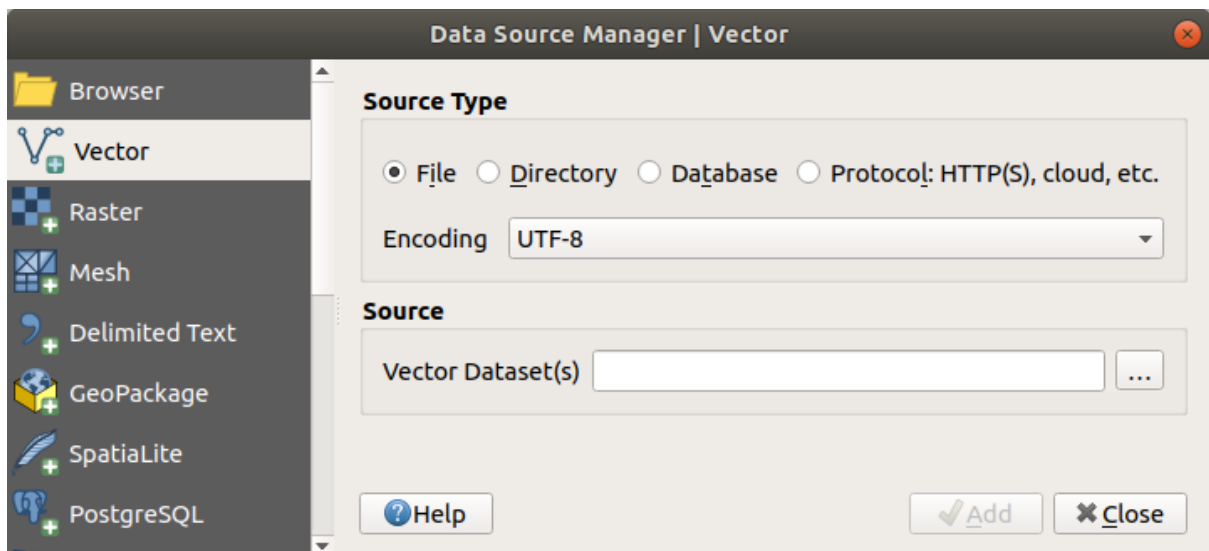




Abb. 15.4: Add Vector Layer Dialog

- for raster data (like GeoTiff, MBTiles, GRIdDED Binary and DWG layers): press `Ctrl+Shift+R`, select the *Layer* ► *Add Layer* ►  *Add Raster Layer* menu option or click on the  Add Raster Layer toolbar button.

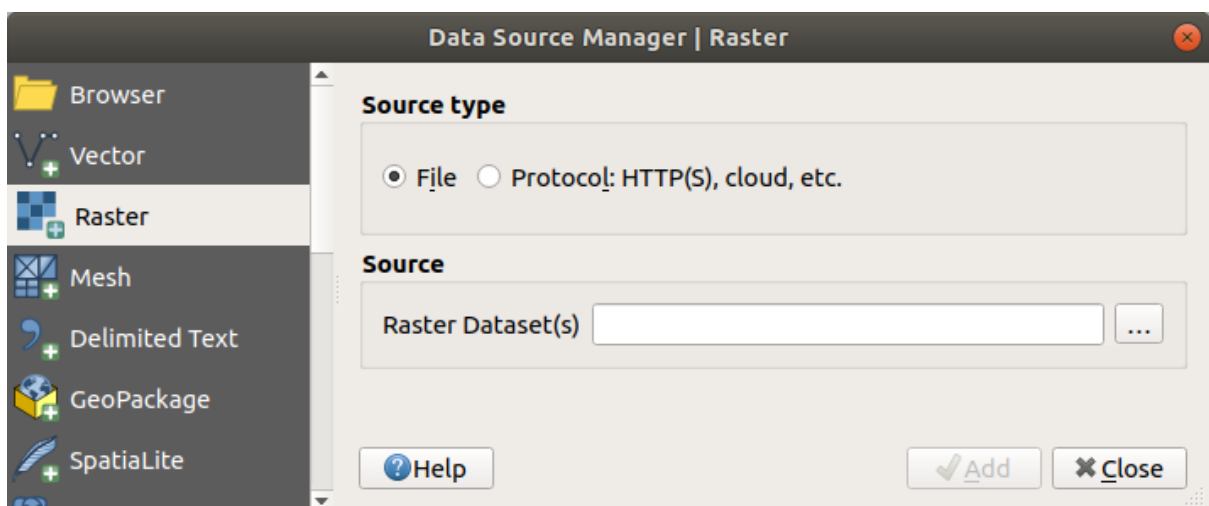



Abb. 15.5: Add Raster Layer Dialog

2. Check  *File* source type
3. Click on the ... <sup>Browse</sup> button
4. Navigate the file system and load a supported data source. More than one layer can be loaded at the same time by holding down the `Ctrl` key and clicking on multiple items in the dialog or holding down the `Shift` key to select a range of items by clicking on the first and last items in the range. Only formats that have been well tested appear in the formats filter. Other formats can be loaded by selecting *All files* (the top item in the pull-down menu).
5. Press *Open* to load the selected file into *Data Source Manager* dialog

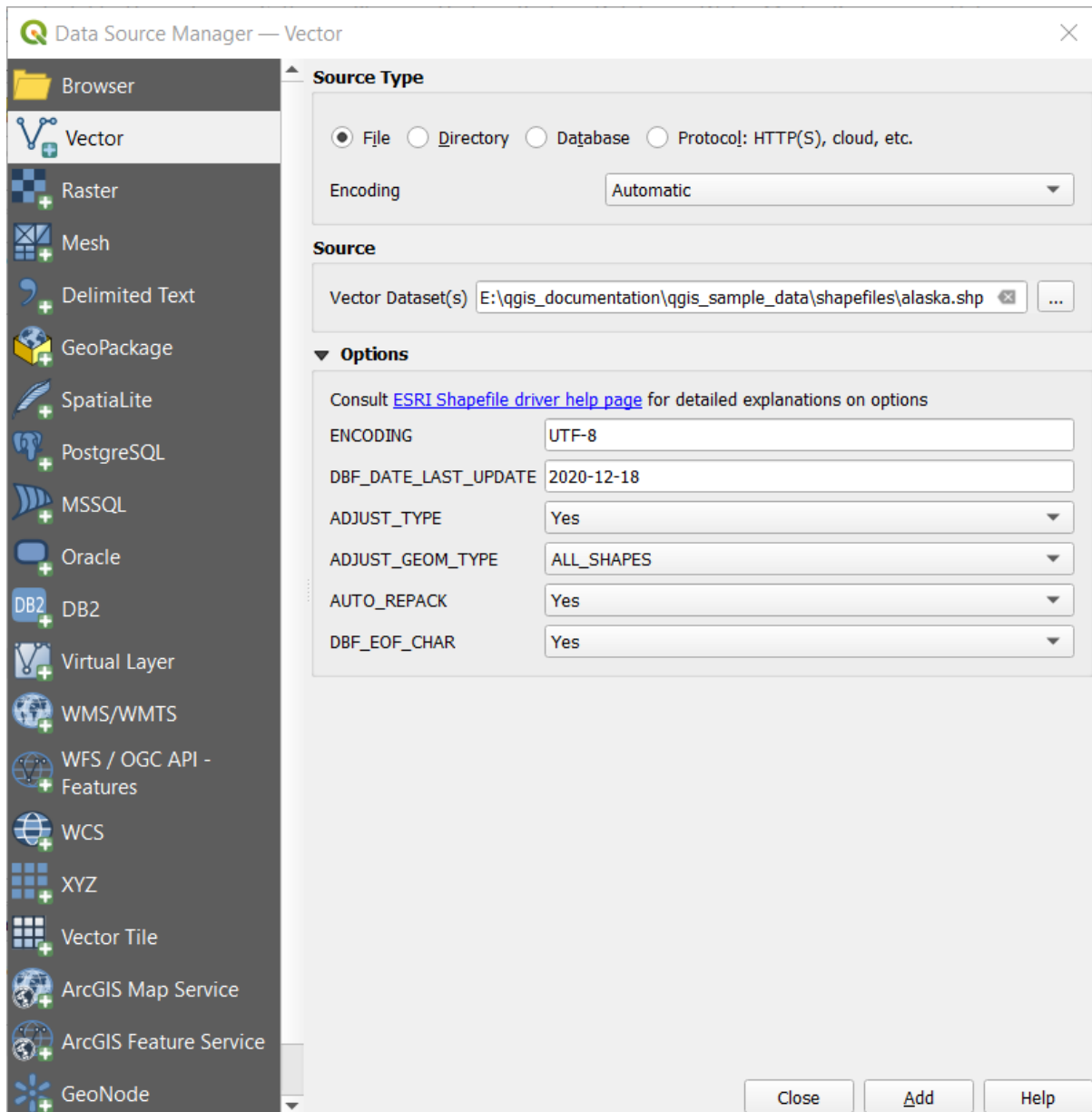


Abb. 15.6: Loading a Shapefile with open options

6. Press *Add* to load the file in QGIS and display them in the map view. When adding vector datasets containing multiple layers, the *Select Items to Add* dialog will appear. In this dialog, you can choose the specific layers from your dataset that you want to add. Also, under *Options* you can choose to:

- ☒ *Add layers to a group*
- ☒ *Show system and internal tables*
- ☒ *Show empty vector layers.*

Abb. 15.7 shows QGIS after loading the `alaska.shp` file.

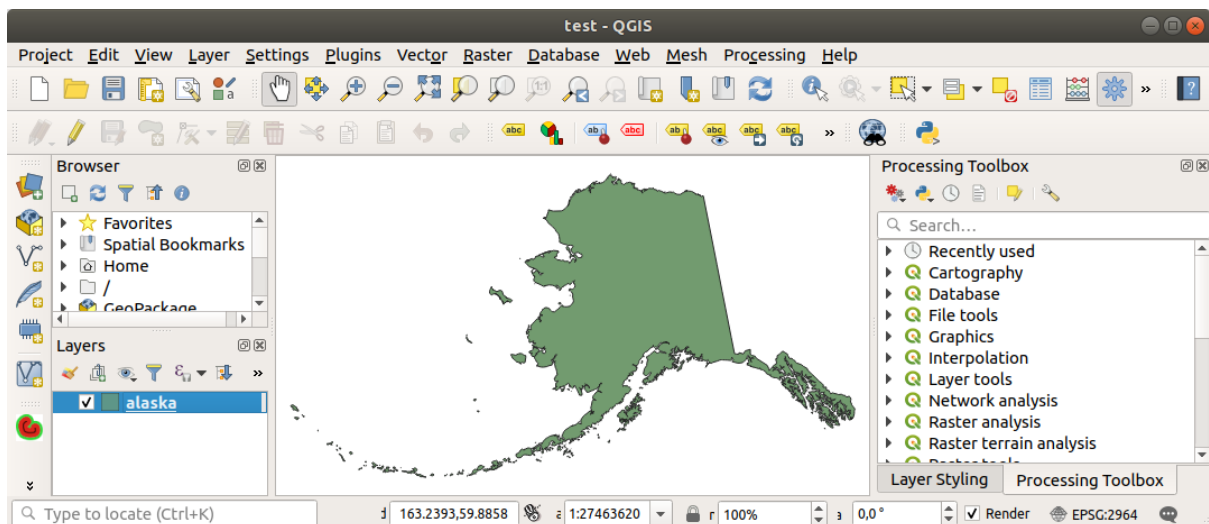




Abb. 15.7: QGIS with Shapefile of Alaska loaded

**Bemerkung:** For loading vector and raster files the GDAL driver offers to define open actions. These will be shown when a file is selected. Options are described in detail on <https://gdal.org/drivers/vector/>, <https://gdal.org/drivers/raster/> and if a file is selected in QGIS, a text with hyperlink will directly lead to the documentation of the selected file type.


**Bemerkung:** Because some formats like MapInfo (e.g., .tab) or Autocad (.dxf) allow mixing different types of geometry in a single file, loading such datasets opens a dialog to select geometries to use in order to have one geometry per layer.

The  Add Vector Layer and  Add Raster Layer tabs allow loading of layers from source types other than *File*:

- You can load specific vector formats like ArcInfo Binary Coverage, UK. National Transfer Format, as well as the raw TIGER format of the US Census Bureau or OpenfileGDB. To do that, you select  *Directory* as *Source type*. In this case, a directory can be selected in the dialog after pressing ... Browse.

- With the  *Database* source type you can select an existing database connection or create one to the selected database type. Some possible database types are ODBC, Esri Personal Geodatabase, MS SQL Server as well as PostgreSQL or MySQL.



Pressing the *New* button opens the *Create a New OGR Database Connection* dialog whose parameters are among the ones you can find in [Creating a stored Connection](#). Pressing *Open* lets you select from the available tables, for example of PostGIS enabled databases.

- The  *Protocol: HTTP(S), cloud, etc.* source type opens data stored locally or on the network, either publicly accessible, or in private buckets of commercial cloud storage services. Supported protocol types are:
  - HTTP/HTTPS/FTP, with a *URI* and, if required, an *authentication*.
  - Cloud storage such as AWS S3, Google Cloud Storage, Microsoft Azure Blob, Alibaba OSS Cloud, Open Stack Swift Storage. You need to fill in the *Bucket or container* and the *Object key*.
  - service supporting OGC WFS 3 (still experimental), using GeoJSON or GEOJSON - Newline Delimited format or based on CouchDB database. A *URI* is required, with optional *authentication*.
  - For all vector source types it is possible to define the *Encoding* or to use the *Automatic* ► setting.

## Loading a mesh layer

A mesh is an unstructured grid usually with temporal and other components. The spatial component contains a collection of vertices, edges and faces in 2D or 3D space. More information on mesh layers at [Working with Mesh Data](#).

To add a mesh layer to QGIS:

1. Open the *Data Source Manager* dialog, either by selecting it from the *Layer* ► menu or clicking the  button.
2. Enable the  *Mesh* tab on the left panel
3. Press the ... <sup>Browse</sup> button to select the file. *Various formats* are supported.
4. Select the file and press *Add*. The layer will be added using the native mesh rendering.
5. If the selected file contains many mesh layers, then you'll be prompted with a dialog to choose the sublayers to load. Do your selection and press *OK* and the layers are loaded with the native mesh rendering. It's also possible to load them within a group.

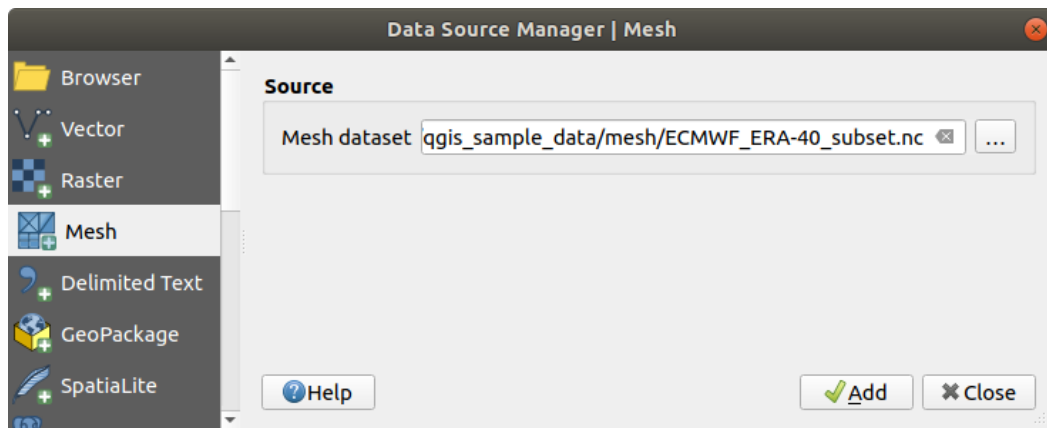





Abb. 15.8: Mesh tab in Data Source Manager

## Importing a delimited text file

Delimited text files (e.g. .txt, .csv, .dat, .wkt) can be loaded using the tools described above. This way, they will show up as simple tables. Sometimes, delimited text files can contain coordinates / geometries that you could want to visualize. This is what  *Add Delimited Text Layer* is designed for.

1. Click the  *Open Data Source Manager* icon to open the *Data Source Manager* dialog
2. Enable the  *Delimited Text* tab
3. Select the delimited text file to import (e.g., `qgis_sample_data/csv/elevp.csv`) by clicking on the ... <sup>Browse</sup> button.
4. In the *Layer name* field, provide the name to use for the layer in the project (e.g. *Elevation*).
5. Configure the settings to meet your dataset and needs, as explained below.

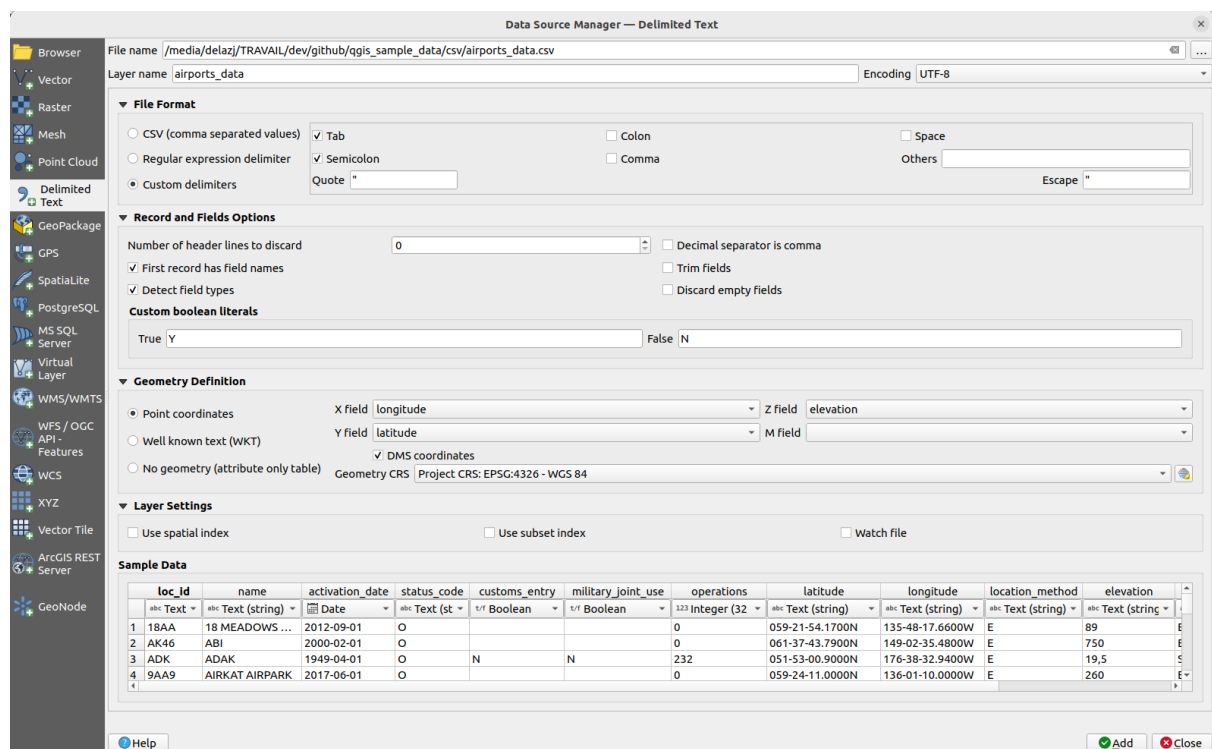


Abb. 15.9: Delimited Text Dialog

## File format

Once the file is selected, QGIS attempts to parse the file with the most recently used delimiter, identifying fields and rows. To enable QGIS to correctly parse the file, it is important to select the right delimiter. You can specify a delimiter by choosing between:


- ☒ **CSV (comma separated values)** to use the comma character.
- ☐ **Regular expression delimiter** and enter text into the *Expression* field. For example, to change the delimiter to tab, use `\t` (this is used in regular expressions for the tab character).
- ☐ **Custom delimiters**, choosing among some predefined delimiters like comma, space, tab, semicolon, ...

## Records and fields


Some other convenient options can be used for data recognition:

- **Number of header lines to discard**: convenient when you want to avoid the first lines in the file in the import, either because those are blank lines or with another formatting.
- ☒ **First record has field names**: values in the first line are used as field names, otherwise QGIS uses the field names `field_1`, `field_2`...
- ☒ **Detect field types**: automatically recognizes the field type. If unchecked then all attributes are treated as text fields.
- ☒ **Decimal separator is comma**: you can force decimal separator to be a comma.
- ☒ **Trim fields**: allows you to trim leading and trailing spaces from fields.



-  *Discard empty fields.*
- *Custom boolean literals:* allows you to add a custom couple of string that will be detected as boolean values.

## Field type detection

QGIS tries to detect the field types automatically (unless  *Detect field types* is not checked) by examining the content of an optional sidcar CSV file (see: [GeoCSV specification](#)) and by scanning the whole file to make sure that all values can actually be converted without errors, the fall-back field type is text.







The detected field type appears under the field name in sample data preview table and can be manually changed if necessary.

The following field types are supported:

- Boolean case-insensitive literal couples that are interpreted as boolean values are 1/0, true/false, t/f, yes/no
- Whole Number (integer)
- Whole Number (integer – 64 bit)
- Decimal Number: double precision floating point number
- Date
- Time
- Date and Time
- Text




## Geometry definition

Once the file is parsed, set *Geometry definition* to

-  *Point coordinates* and provide the *X field*, *Y field*, *Z field* (for 3-dimensional data) and *M field* (for the measurement dimension) if the layer is of point geometry type and contains such fields. If the coordinates are defined as degrees/minutes/seconds, activate the  *DMS coordinates* checkbox. Provide the appropriate *Geometry CRS* using the  *Select CRS* widget.
-  *Well known text (WKT)* option if the spatial information is represented as WKT: select the *Geometry field* containing the WKT geometry and choose the appropriate *Geometry field* or let QGIS auto-detect it. Provide the appropriate *Geometry CRS* using the  *Select CRS* widget.
- If the file contains non-spatial data, activate  *No geometry (attribute only table)* and it will be loaded as an ordinary table.

### Layer settings

Additionally, you can enable:

-  *Use spatial index* to improve the performance of displaying and spatially selecting features.
-  *Use subset index* to improve performance of *subset filters* (when defined in the layer properties).
-  *Watch file* to watch for changes to the file by other applications while QGIS is running.

At the end, click *Add* to add the layer to the map. In our example, a point layer named *Elevation* is added to the project and behaves like any other map layer in QGIS. This layer is the result of a query on the *.csv* source file (hence, linked to it) and would require *to be saved* in order to get a spatial layer on disk.

### Sample Data

As you set the parser properties, the sample data preview updates regarding to the applied settings.

Also in the Sample Data Table it is possible to override the automatically determined column types.

### Importing a DXF or DWG file

DXF and DWG files can be added to QGIS by simple drag-and-drop from the Browser Panel. You will be prompted to select the sublayers you would like to add to the project. Layers are added with random style properties.

---


**Bemerkung:** For DXF files containing several geometry types (point, line and/or polygon), the name of the layers will be generated as *<filename.dxf> entities <geometry type>*.

---

To keep the dxf/dwg file structure and its symbology in QGIS, you may want to use the dedicated *Project ► Import/Export ► Import Layers from DWG/DXF...* tool which allows you to:

1. import elements from the drawing file into a GeoPackage database.
2. add imported elements to the project.

In the *DWG/DXF Import* dialog, to import the drawing file contents:

1. Input the location of the *Source drawing*, i.e. the DWG/DXF drawing file to import.
2. Specify the coordinate reference system of the data in the drawing file.
3. Input the location of the *Target package*, i.e. the GeoPackage file that will store the data. If an existing file is provided, then it will be overwritten.
4. Choose how to import *blocks* with the dedicated combobox:
  - *Expand Block Geometries*: imports the blocks in the drawing file as normal elements.
  - *Expand Block Geometries and Add Insert Points*: imports the blocks in the drawing file as normal elements and adds the insertion point as a point layer.
  - *Add Only Insert Points*: adds the blocks insertion point as a point layer.
5. Check  *Use curves* to promote the imported layers to a *curved* geometry type.
6. Use the *Import* button to import the drawing into the destination GeoPackage file. The GeoPackage database will be automatically populated with the drawing file content. Depending on the size of the file, this can take some time.

After the *.dwg* or *.dxf* data has been imported into the GeoPackage database, the frame in the lower half of the dialog is populated with the list of layers from the imported file. There you can select which layers to add to the QGIS project:

1. At the top, set a *Group name* to group the drawing files in the project. By default this is set to the filename of the source drawing file.
2. Check layers to show: Each selected layer is added to an ad hoc group which contains vector layers for the point, line, label and area features of the drawing layer. The style of the layers will resemble the look they originally had in \*CAD.
3. Choose if the layer should be visible at opening.
4. Checking the ☒ *Merge layers* option places all layers in a single group.
5. Press *OK* to open the layers in QGIS.

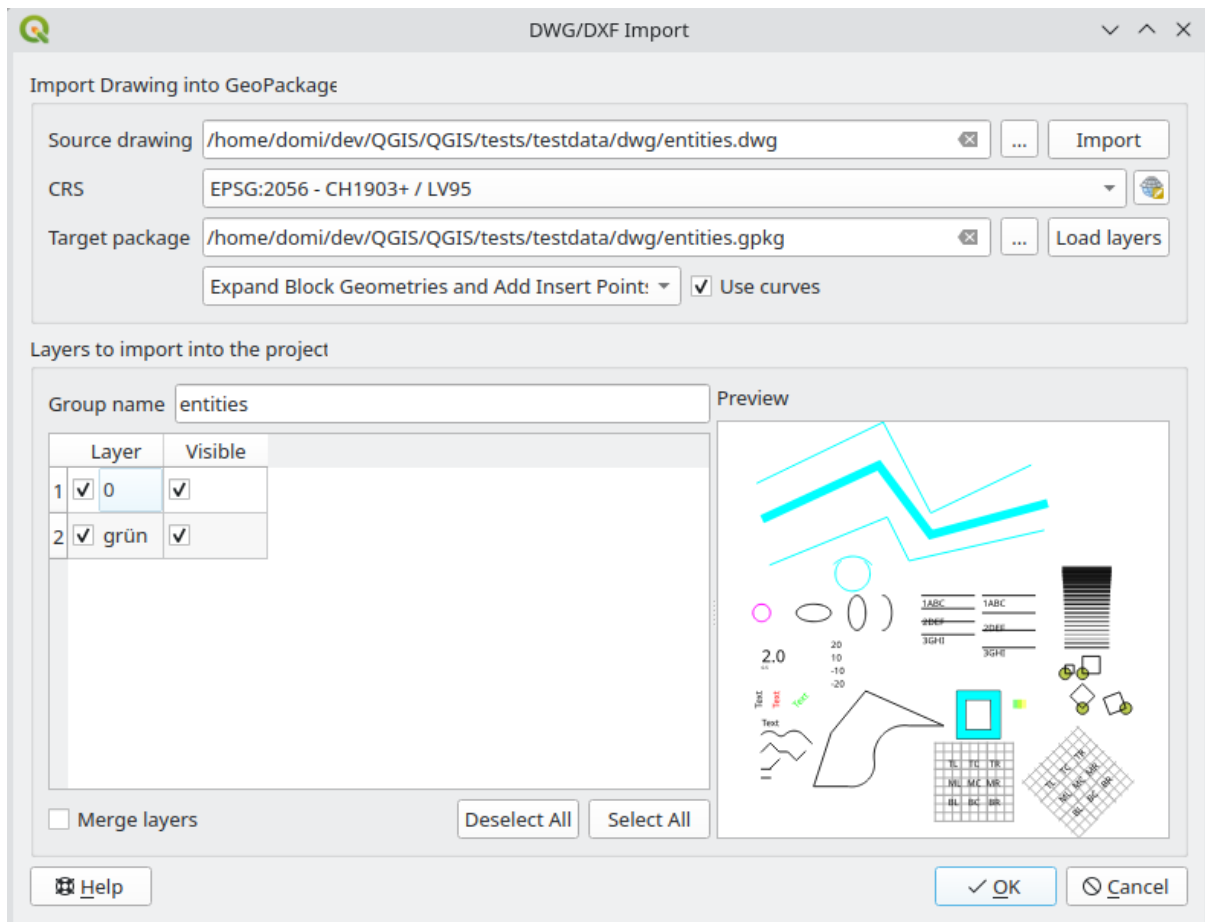



Abb. 15.10: Import dialog for DWG/DXF files



## Importing OpenStreetMap Vectors

The OpenStreetMap project is popular because in many countries no free geodata such as digital road maps are available. The objective of the OSM project is to create a free editable map of the world from GPS data, aerial photography and local knowledge. To support this objective, QGIS provides support for OSM data.

Using the *Browser Panel*, you can load an `.osm` file to the map canvas, in which case you'll get a dialog to select sublayers based on the geometry type. The loaded layers will contain all the data of that geometry type in the `.osm` file, and keep the `osm` file data structure.

### Spatialite Layers

 The first time you load data from a Spatialite database, begin by:

- clicking on the  Add Spatialite Layer toolbar button
- selecting the  Add Spatialite Layer... option from the *Layer ► Add Layer* menu
- or by typing `Ctrl+Shift+L`

This will bring up a window that will allow you either to connect to a Spatialite database already known to QGIS (which you choose from the drop-down menu) or to define a new connection to a new database. To define a new connection, click on *New* and use the file browser to point to your Spatialite database, which is a file with a `.sqlite` extension.

QGIS also supports editable views in Spatialite.

### GPS

Es gibt viele verschiedene Datenformate zum Speichern von GPS-Daten. Das von QGIS unterstützte Format ist GPX (GPS eXchange format), ein Standardformat, welches Wegpunkte, Routen und Spuren in einer Datei enthalten kann.

Use the ... <sup>Browse</sup> button to select the GPX file, then use the check boxes to select the feature types you want to load from that GPX file. Each feature type will be loaded in a separate layer.

More on GPS data manipulation at [Arbeiten mit GPS Daten](#).

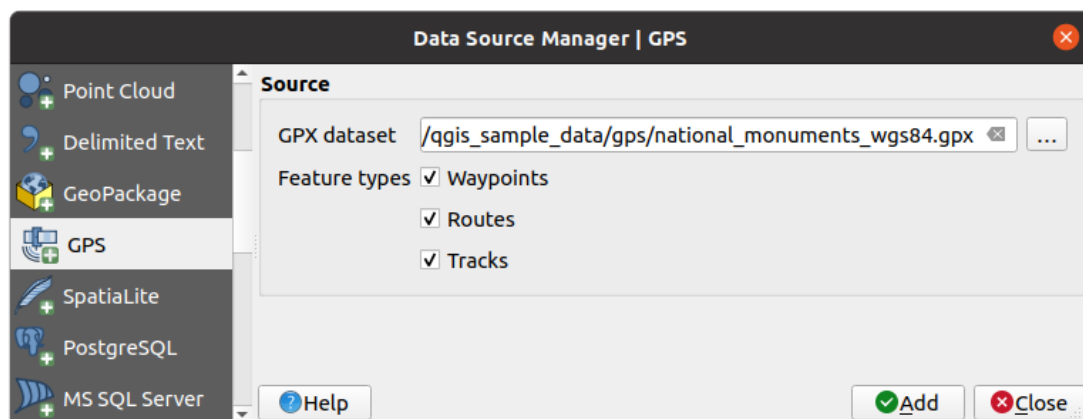


Abb. 15.11: Loading GPS Data dialog





### GRASS

Working with GRASS vector data is described in section [GRASS GIS Integration](#).

## Database related tools

### Creating a stored Connection

In order to read and write tables from a database format QGIS supports you have to create a connection to that database. While *QGIS Browser Panel* is the simplest and recommended way to connect to and use databases, QGIS provides other tools to connect to each of them and load their tables:

-  *Add PostGIS Layer...* or by typing `Ctrl+Shift+D`
-  *Add MS SQL Server Layer*
-  *Add Oracle Spatial Layer...* or by typing `Ctrl+Shift+O`
-  *Add SAP HANA Spatial Layer...* or by typing `Ctrl+Shift+G`

These tools are accessible either from the *Manage Layers Toolbar* and the *Layer ► Add Layer ►* menu. Connecting to SpatiaLite database is described at *SpatiaLite Layers*.

---

#### **Tipp: Create connection to database from the QGIS Browser Panel**

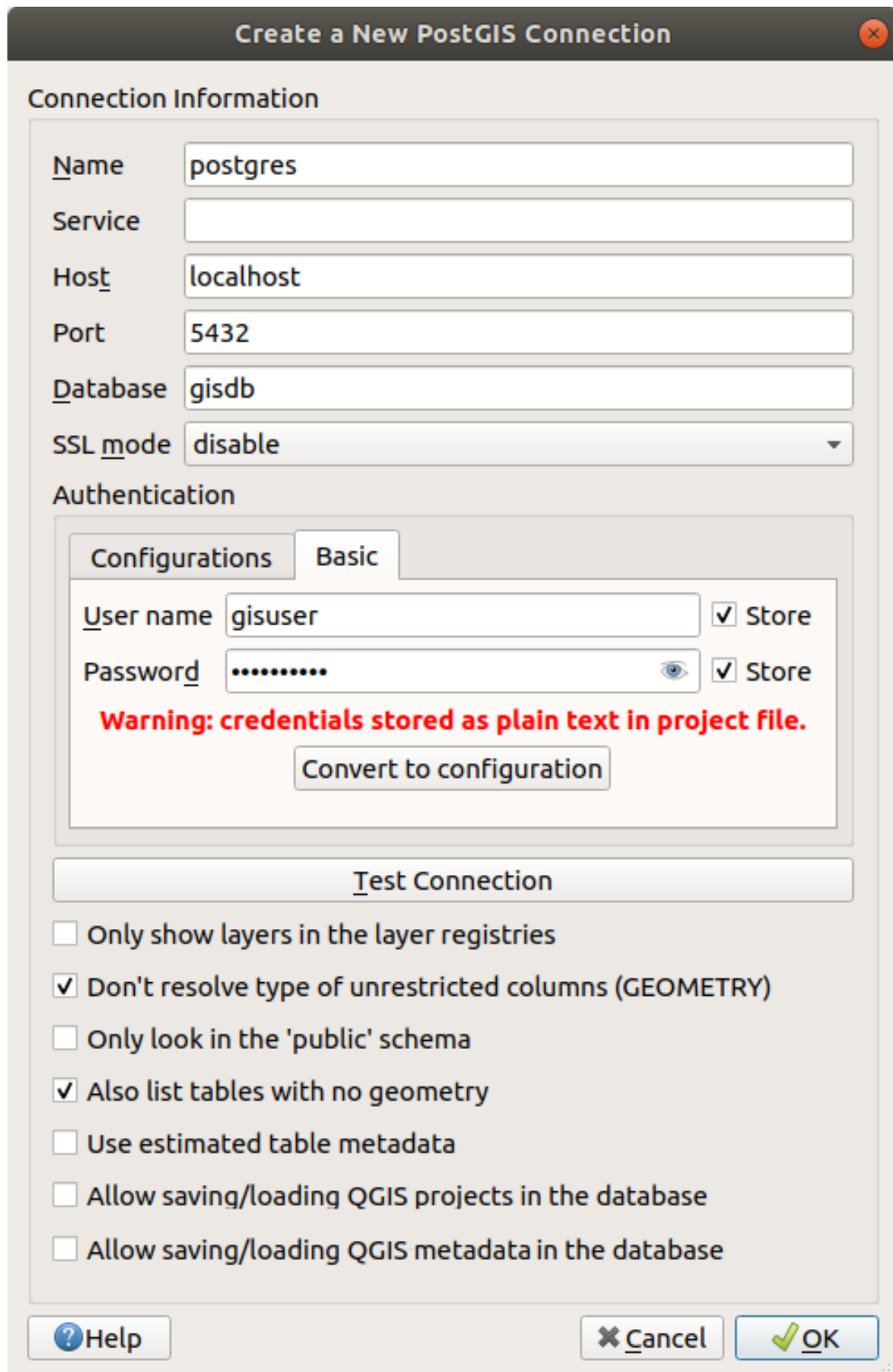
Selecting the corresponding database format in the Browser tree, right-clicking and choosing connect will provide you with the database connection dialog.

---

Most of the connection dialogs follow a common basis that will be described below using the PostgreSQL database tool as an example. For additional settings specific to other providers, you can find corresponding descriptions at:

- *Connecting to MS SQL Server*;
- *Connecting to Oracle Spatial*;
- *Connecting to SAP HANA*.

The first time you use a PostGIS data source, you must create a connection to a database that contains the data. Begin by clicking the appropriate button as exposed above, opening an *Add PostGIS Table(s)* dialog (see [Abb. 15.15](#)). To access the connection manager, click on the *New* button to display the *Create a New PostGIS Connection* dialog.



**Create a New PostGIS Connection**

**Connection Information**

Name: postgres

Service:

Host: localhost

Port: 5432

Database: gisdb

SSL mode: disable

**Authentication**

Configurations Basic

User name: gisuser ☒ Store

Password: ..... ☒ Store

**Warning: credentials stored as plain text in project file.**

Convert to configuration

Test Connection

☐ Only show layers in the layer registries

☒ Don't resolve type of unrestricted columns (GEOMETRY)

☐ Only look in the 'public' schema

☒ Also list tables with no geometry

☐ Use estimated table metadata

☐ Allow saving/loading QGIS projects in the database

☐ Allow saving/loading QGIS metadata in the database

Help Cancel OK

Abb. 15.12: Create a New PostGIS Connection Dialog


The parameters required for a PostgreSQL connection are explained below. For the other database types, see their differences at [Particular Connection requirements](#).

- *Name*: A name for this connection. It can be the same as *Database*.
- *Service*: Service parameter to be used alternatively to hostname/port (and potentially database). This can be defined in `pg_service.conf`. Check the [PostgreSQL Service connection file](#) section for more details.
- *Host*: Name of the database host. This must be a resolvable host name such as would be used to open a TCP/IP connection or ping the host. If the database is on the same computer as QGIS, simply enter *localhost* here.
- *Port*: Port number the PostgreSQL database server listens on. The default port for PostgreSQL is 5432.
- *Database*: Name of the database.
- *SSL mode*: SSL encryption setup The following options are available:
  - *Prefer* (the default): I don't care about encryption, but I wish to pay the overhead of encryption if the server supports it.
  - *Require*: I want my data to be encrypted, and I accept the overhead. I trust that the network will make sure I always connect to the server I want.
  - *Verify CA*: I want my data encrypted, and I accept the overhead. I want to be sure that I connect to a server that I trust.
  - *Verify Full*: I want my data encrypted, and I accept the overhead. I want to be sure that I connect to a server I trust, and that it's the one I specify.
  - *Allow*: I don't care about security, but I will pay the overhead of encryption if the server insists on it.
  - *Disable*: I don't care about security, and I don't want to pay the overhead of encryption.
- *Authentication*, basic.
  - *User name*: User name used to log in to the database.
  - *Password*: Password used with *Username* to connect to the database.


You can save any or both of the `User name` and `Password` parameters, in which case they will be used by default each time you need to connect to this database. If not saved, you'll be prompted to supply the credentials to connect to the database in next QGIS sessions. The connection parameters you entered are stored in a temporary internal cache and returned whenever a username/password for the same database is requested, until you end the current QGIS session.







#### Warning: QGIS User Settings and Security

In the *Authentication* tab, saving **username** and **password** will keep unprotected credentials in the connection configuration. Those **credentials will be visible** if, for instance, you share the project file with someone. Therefore, it is advisable to save your credentials in an *Authentication configuration* instead (*Configurations* tab - See [Authentifizierungssystem](#) for more details) or in a service connection file (see [PostgreSQL Service connection file](#) for example).

- *Authentication*, configurations. Choose an authentication configuration. You can add configurations using the  button. Choices are:
  - Basic authentication
  - PKI PKCS#12 authentication
  - PKI paths authentication
  - PKI stored identity certificate

Optionally, depending on the type of database, you can activate the following checkboxes:

-  *Only show layers in the layer registries*

-  *Don't resolve type of unrestricted columns (GEOMETRY)*
-  *Only look in the ,public' schema*
-  *Also list tables with no geometry*
-  *Use estimated table metadata*
-  *Allow saving/loading QGIS projects in the database - more details [here](#)*
-  *Allow saving/loading QGIS layer metadata in the database - more details [here](#)*

---

### **Tipp:** Use estimated table metadata to speed up operations

When initializing layers, various queries may be needed to establish the characteristics of the geometries stored in the database table. When the *Use estimated table metadata* option is checked, these queries examine only a sample of the rows and use the table statistics, rather than the entire table. This can drastically speed up operations on large datasets, but may result in incorrect characterization of layers (e.g. the feature count of filtered layers will not be accurately determined) and may even cause strange behaviour if columns that are supposed to be unique actually are not.

---

Once all parameters and options are set, you can test the connection by clicking the *Test Connection* button or apply it by clicking the *OK* button. From *Add PostGIS Table(s)*, click now on *Connect*, and the dialog is filled with tables from the selected database (as shown in [Abb. 15.15](#)).

### **Particular Connection requirements**

Because of database type particularities, provided options are not the same. Database specific options are described below.

### **PostgreSQL Service connection file**

The service connection file allows PostgreSQL connection parameters to be associated with a single service name. That service name can then be specified by a client and the associated settings will be used.

It's called `.pg_service.conf` under \*nix systems (GNU/Linux, macOS etc.) and `pg_service.conf` on Windows.

The service file can look like this:

```
[water_service]
host=192.168.0.45
port=5433
dbname=gisdb
user=paul
password=paulspass

[wastewater_service]
host=dbserver.com
dbname=water
user=waterpass
```

---

**Bemerkung:** There are two services in the above example: `water_service` and `wastewater_service`. You can use these to connect from QGIS, pgAdmin, etc. by specifying only the name of the service you want to connect to (without the enclosing brackets). If you want to use the service with `psql`, you can do `psql service=water_service`.

You can find all the PostgreSQL parameters [here](#)

---



---

**Bemerkung:** If you don't want to save the passwords in the service file you can use the `.pg_pass` option.

---



---

**Bemerkung: QGIS Server and service**

When using a service file and QGIS Server, you must configure the service on the server side as well. You can follow the QGIS Server documentation.

---

On \*nix operating systems (GNU/Linux, macOS etc.) you can save the `.pg_service.conf` file in the user's home directory and PostgreSQL clients will automatically be aware of it. For example, if the logged user is `web`, `.pg_service.conf` should be saved in the `/home/web/` directory in order to directly work (without specifying any other environment variables).

You can specify the location of the service file by creating a `PGSERVICEFILE` environment variable (e.g. run the `export PGSERVICEFILE=/home/web/.pg_service.conf` command under your \*nix OS to temporarily set the `PGSERVICEFILE` variable)

You can also make the service file available system-wide (all users) either by placing the `.pg_service.conf` file in `pg_config --sysconfdir` or by adding the `PGSYSCONFDIR` environment variable to specify the directory containing the service file. If service definitions with the same name exist in the user and the system file, the user file takes precedence.

**Warnung:** There are some caveats under Windows:




- The service file should be saved as `pg_service.conf` and not as `.pg_service.conf`.
- The service file should be saved in Unix format in order to work. One way to do it is to open it with `Notepad++` and `Edit ► EOL Conversion ► UNIX Format ► File save`.
- You can add environmental variables in various ways; a tested one, known to work reliably, is `Control Panel ► System and Security ► System ► Advanced system settings ► Environment Variables` adding `PGSERVICEFILE` with the path - e.g. `C:\Users\John\pg_service.conf`
- After adding an environment variable you may also need to restart the computer.




## Connecting to Oracle Spatial

The spatial features in Oracle Spatial aid users in managing geographic and location data in a native type within an Oracle database. In addition to some of the options in [Creating a stored Connection](#), the connection dialog proposes:

- **Database:** SID or SERVICE\_NAME of the Oracle instance;
- **Port:** Port number the Oracle database server listens on. The default port is 1521;
- **Options:** Oracle connection specific options (e.g. `OCI_ATTR_PREFETCH_ROWS`, `OCI_ATTR_PREFETCH_MEMORY`). The format of the options string is a semicolon separated list of option names or option=value pairs;
- **Workspace:** Workspace to switch to;
- **Schema:** Schema in which the data are stored

Optionally, you can activate the following checkboxes:

-  *Only look in metadata table:* restricts the displayed tables to those that are in the `all_sdo_geom_metadata` view. This can speed up the initial display of spatial tables.
-  *Only look for user's tables:* when searching for spatial tables, restricts the search to tables that are owned by the user.
-  *Also list tables with no geometry:* indicates that tables without geometry should also be listed by default.

-  *Use estimated table statistics for the layer metadata:* when the layer is set up, various metadata are required for the Oracle table. This includes information such as the table row count, geometry type and spatial extents of the data in the geometry column. If the table contains a large number of rows, determining this metadata can be time-consuming. By activating this option, the following fast table metadata operations are done: Row count is determined from `all_tables.num_rows`. Table extents are always determined with the `SDO_TUNE.EXTENTS_OF` function, even if a layer filter is applied. Table geometry is determined from the first 100 non-null geometry rows in the table.
-  *Only existing geometry types:* only lists the existing geometry types and don't offer to add others.
-  *Include additional geometry attributes.*

### Tipp: Oracle Spatial Layers

Normally, an Oracle Spatial layer is defined by an entry in the `USER_SDO_METADATA` table.

To ensure that selection tools work correctly, it is recommended that your tables have a **primary key**.

## Connecting to MS SQL Server

As mentioned in *Creating a stored Connection* QGIS allows you to create MS SQL Server connection through *Data Source Manager*.

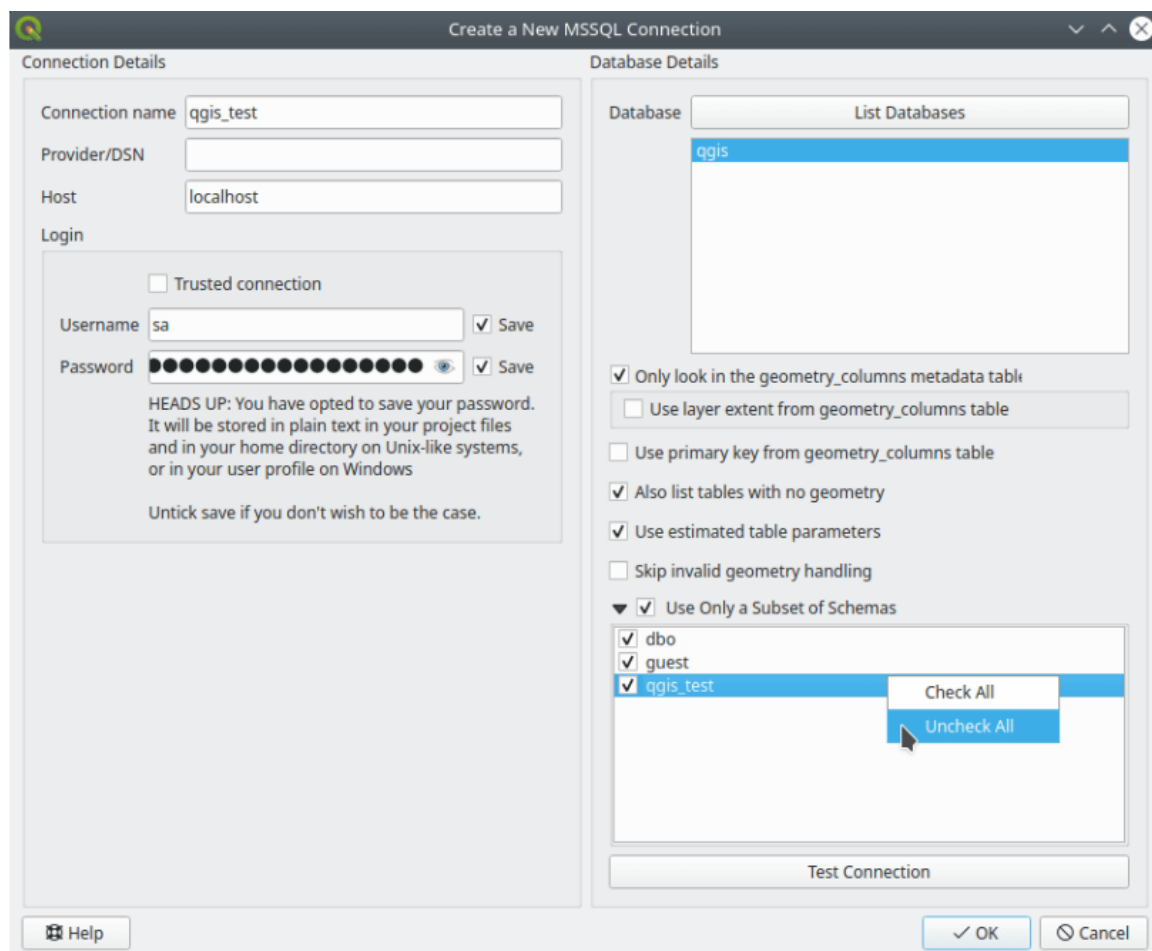










Abb. 15.13: MS SQL Server Connection

To create a new MS SQL Server connection, you need to provide some of the following information in the *Connection Details* dialog:

- *Connection name*
- *Provider/DNS*
- *Host*
- *Login information*. You can choose to  *Save* your credentials.

Navigate to the *Database Details* section and click the *List Databases* button to view the available datasets. Select datasets that you want, then press *OK*. Optionally, you can also perform a *Test Connection*. Once you click *OK* the *Create a New MS SQL Server Connection* dialog will close and in the *Data Source Manager* press *Connect*, select a layer and then click *Add*.

Optionally, you can activate the following options:

-  *Only look in the geometry\_columns metadata table.*
-  *Use layer extent from geometry\_columns table*, this checkbox is dependent on the first one; it remains disabled unless the first option is checked.
-  *Use primary key from geometry\_columns table*
-  *Also list table with no geometry*: tables without a geometry column attached will also be shown in the available table list.
-  *Use estimated table parameters*: only estimated table metadata will be used. This avoids a slow table scan, but may result in incorrect layer properties such as layer extent.
-  *Skip invalid geometry handling*: all handling of records with invalid geometry will be disabled. This speeds up the provider, however, if any invalid geometries are present in a table then the result is unpredictable and may include missing records. Only check this option if you are certain that all geometries present in the database are valid, and any newly added geometries or tables will also be valid.
-  *Use only a Subset of Schemas* will allow you to filter schemas for MS SQL connection. If enabled, only checked schemas will be displayed. You can right-click to *Check* or *Uncheck* any schema in the list.

## Connecting to SAP HANA

---

**Bemerkung:** You require the SAP HANA Client to connect to a SAP HANA database. You can download the SAP HANA Client for your platform at the [SAP Development Tools website](#).

---

Create a New SAP HANA Connection

Connection Information

Name

Driver

/usr/sap/hdbclient/libodbcHDB.so

Host

Identifier

Instance Number

00

Mode

Single container

Multiple containers

Tenant database

Name:

System database

Schema

Authentication

SSL Settings

Configurations

Basic

User name

Store

Password

Optional

Store

Warning: credentials stored as plain text in project file.

Convert to configuration

Test Connection

Only look for user's tables







Also list tables with no geometry

Help

Cancel

OK

The following parameters can be entered:

- *Name*: A name for this connection.
- *Driver* : The name of the HANA ODBC driver. It is `HDBODBC` if you are using 64-bit QGIS, `HDBODBC32` if you are using 32-bit QGIS. The appropriate driver name is entered automatically.
- *Driver*  : Either the name under which the SAP HANA ODBC driver has been registered in `/etc/odbcinst.ini` or the full path to the SAP HANA ODBC driver. The SAP HANA Client installer will install the ODBC driver to `/usr/sap/hdbclient/libodbcHDB.so` by default.
- *Host*: The name of the database host.
- *Identifier*: Identifies the instance to connect to on the host. This can be either *Instance Number* or *Port Number*. Instance numbers consist of two digits, port numbers are in the range from 1 to 65,535.
- *Mode*: Specifies the mode in which the SAP HANA instance runs. This setting is only taken into account if *Identifier* is set to *Instance Number*. If the database hosts multiple containers, you can either connect to a tenant with the name given at *Tenant database* or you can connect to the system database.
- *Schema*: This parameter is optional. If a schema name is given, QGIS will only search for data in that schema. If this field is left blank, QGIS will search for data in all schemas.
- *Authentication* ► *Basic*.
  - *User name*: User name used to connect to the database.
  - *Password*: Password used to connect to the database.
- *SSL Settings*
  - ☒ *Enable TLS/SSL encryption*: Enables TLS 1.1 - TLS1.2 encryption. The server will choose the highest available.
  - *Provider*: Specifies the cryptographic library provider used for SSL communication. *sapcrypto* should work on all platforms, *openssl* should work on  , *mscrypto* should work on  and *commoncrypto* requires CommonCryptoLib to be installed.
  - ☒ *Validate SSL certificate*: If checked, the SSL certificate will be validated using the truststore given in *Trust store file with public key*.
  - *Override hostname in certificate*: Specifies the host name used to verify server's identity. The host name specified here verifies the identity of the server instead of the host name with which the connection was established. If you specify `*` as the host name, then the server's host name is not validated. Other wildcards are not permitted.
  - *Keystore file with private key*: Currently ignored. This parameter might allow to authenticate via certificate instead via user and password in future.
  - *Trust store file with public key*: Specifies the path to a trust store file that contains the server's public certificates if using OpenSSL. Typically, the trust store contains the root certificate or the certificate of the certification authority that signed the server's public certificates. If you are using the cryptographic library CommonCryptoLib or msCrypto, then leave this property empty.
- ☒ *Only look for user's tables*: If checked, QGIS searches only for tables and views that are owned by the user that connects to the database.
- ☒ *Also list tables with no geometries*: If checked, QGIS searches also for tables and views that do not contain a spatial column.

---

#### **Tipp: Connecting to SAP HANA Cloud**

If you'd like to connect to an SAP HANA Cloud instance, you usually must set *Port Number* to 443 and check *Enable TLS/SSL encryption*.

---

## Loading a Database Layer

Once you have one or more connections defined to a database (see section [Creating a stored Connection](#)), you can load layers from it. Of course, this requires that data are available. See section [Importing Data into PostgreSQL](#) for a discussion on importing data into a PostGIS database.

To load a layer from a database, you can perform the following steps:

1. Open the „Add <database> table(s)“ dialog (see [Creating a stored Connection](#)).
2. Choose the connection from the drop-down list and click *Connect*.
3. Select or unselect ☒ *Also list tables with no geometry*.
4. Optionally, use some ☒ *Search Options* to reduce the list of tables to those matching your search. You can also set this option before you hit the *Connect* button, speeding up the database fetching.
5. Find the layer(s) you wish to add in the list of available layers.
6. Select it by clicking on it. You can select multiple layers by holding down the Shift or Ctrl key while clicking.
7. If applicable, use the *Set Filter* button (or double-click the layer) to start the *Query Builder* dialog (see section [Abfrageeditor](#)) and define which features to load from the selected layer. The filter expression appears in the sql column. This restriction can be removed or edited in the *Layer Properties* ► *General* ► *Provider Feature Filter* frame.
8. The checkbox in the *Select at id* column that is activated by default gets the feature ids without the attributes and generally speeds up the data loading.
9. Click on the *Add* button to add the layer to the map.

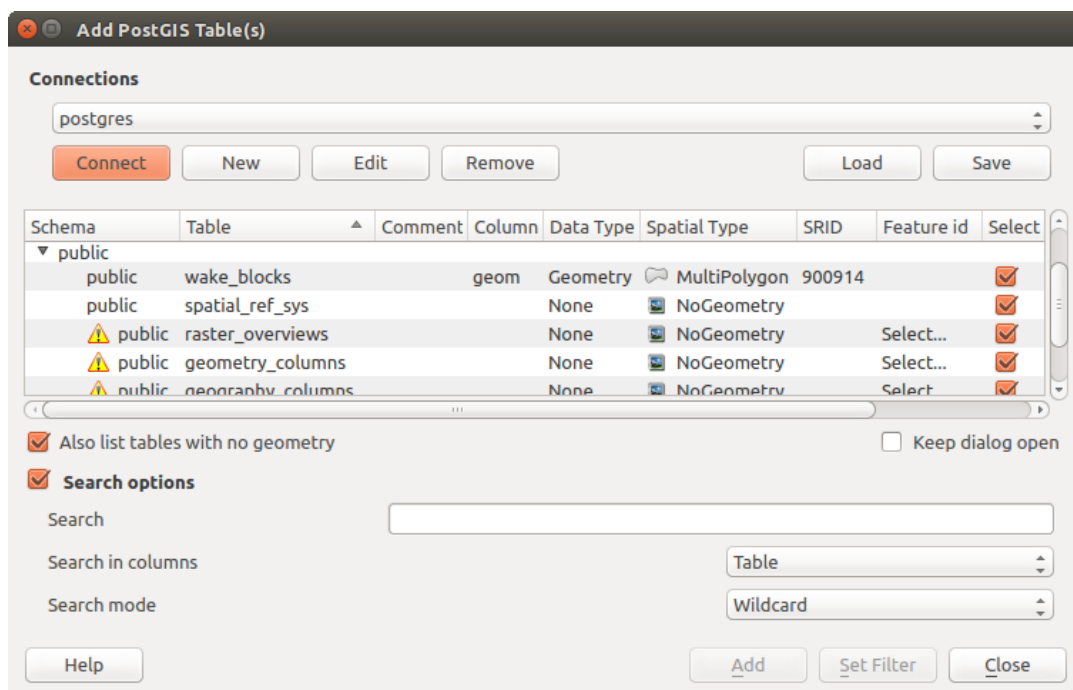


Abb. 15.15: Add PostGIS Table(s) Dialog

### **Tipp:** Use the Browser Panel to speed up loading of database table(s)

Adding DB tables from the *Data Source Manager* may sometimes be time consuming as QGIS fetches statistics and properties (e.g. geometry type and field, CRS, number of features) for each table beforehand. To avoid this, once [the](#)

*connection is set*, it is better to use the *Browser Panel* or the *DB Manager* to drag and drop the database tables into the map canvas.

### 15.1.4 The Layer Metadata Search Panel

By default, QGIS can retrieve layers metadata from the connections or data providers that allow metadata storage (more details on *saving metadata to the database*). The *Metadata search* panel allows to browse the layers by their metadata and add them to the project (either with a double-click or the *Add* button). The list can be filtered:

- by text, watching a set of metadata properties (identifier, title, abstract)
- by spatial extent, using the current *project extent* or the map canvas extent
- by the layer (geometry) type

**Bemerkung:** The sources of metadata are implemented through a layer metadata provider system that can be extended by plugins.

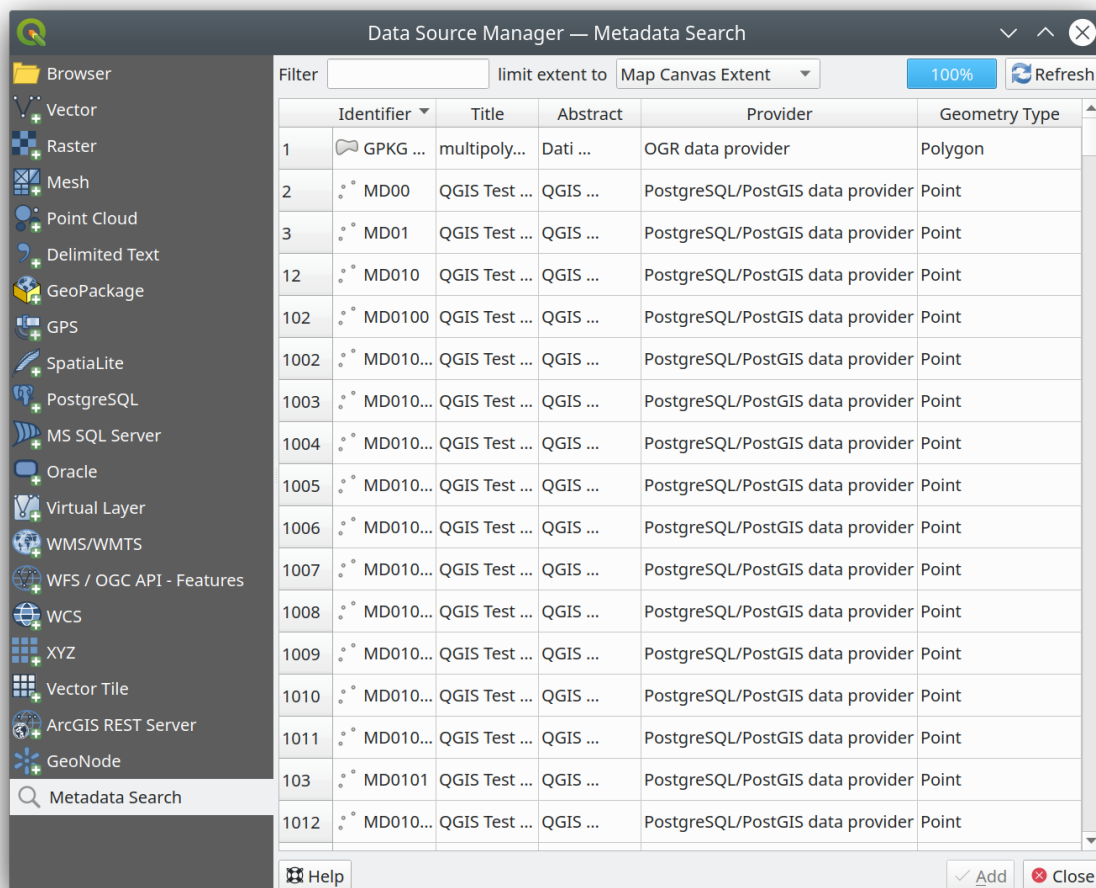


Abb. 15.16: Layer Metadata Search Panel



## 15.1.5 QGIS Custom formats

QGIS proposes two custom formats:

- Temporary Scratch Layer: a memory layer that is bound to the project (see [Creating a new Temporary Scratch Layer](#) for more information)
- Virtual Layers: a layer resulting from a query on other layer(s) (see [Creating virtual layers](#) for more information)

## 15.1.6 QLR - QGIS Layer Definition File

Layer definitions can be saved as a *Layer Definition File* (QLR - .qlr) using *Export ► Save As Layer Definition File...* in the layer context menu.


The QLR format makes it possible to share „complete“ QGIS layers with other QGIS users. QLR files contain links to the data sources and all the QGIS style information necessary to style the layer.

QLR files are shown in the Browser Panel and can be used to add layers (with their saved styles) to the Layers Panel. You can also drag and drop QLR files from the system file manager into the map canvas.



## 15.1.7 Connecting to web services

With QGIS you can get access to different types of OGC web services (WM(T)S, WFS(-T), WCS, CSW, ...). Thanks to QGIS Server, you can also publish such services. QGIS-Server-manual contains descriptions of these capabilities.

### Using Vector Tiles services

Vector Tile services can be added via the  *Vector Tiles* tab of the *Data Source Manager* dialog or the contextual menu of the *Vector Tiles* entry in the *Browser* panel. Services can be either a *New Generic Connection...* or a *New ArcGIS Vector Tile Service Connection....*

You set up a service by adding:

- einen *Namen*
- the *URL*: of the type `http://example.com/{z}/{x}/{y}.pbf` for generic services and `http://example.com/arcgis/rest/services/Layer/VectorTileServer` for ArcGIS based services. The service must provide tiles in .pbf format.
- the  *Min. Zoom Level* and the  *Max. Zoom Level*. Vector Tiles have a pyramid structure. By using these options you have the opportunity to individually generate layers from the tile pyramid. These layers will then be used to render the Vector Tile in QGIS.

For Mercator projection (used by OpenStreetMap Vector Tiles) Zoom Level 0 represents the whole world at a scale of 1:500.000.000. Zoom Level 14 represents the scale 1:35.000.

- a *Style URL*: a URL to a MapBox GL JSON style configuration. If provided, then that style will be applied whenever the layers from the connection are added to QGIS. In the case of Arcgis vector tile service connections, the URL overrides the default style configuration specified in the server configuration.
- the *authentication* configuration if necessary
- a *Referer*

Abb. 15.17 shows the dialog with the MapTiler planet Vector Tiles service configuration.



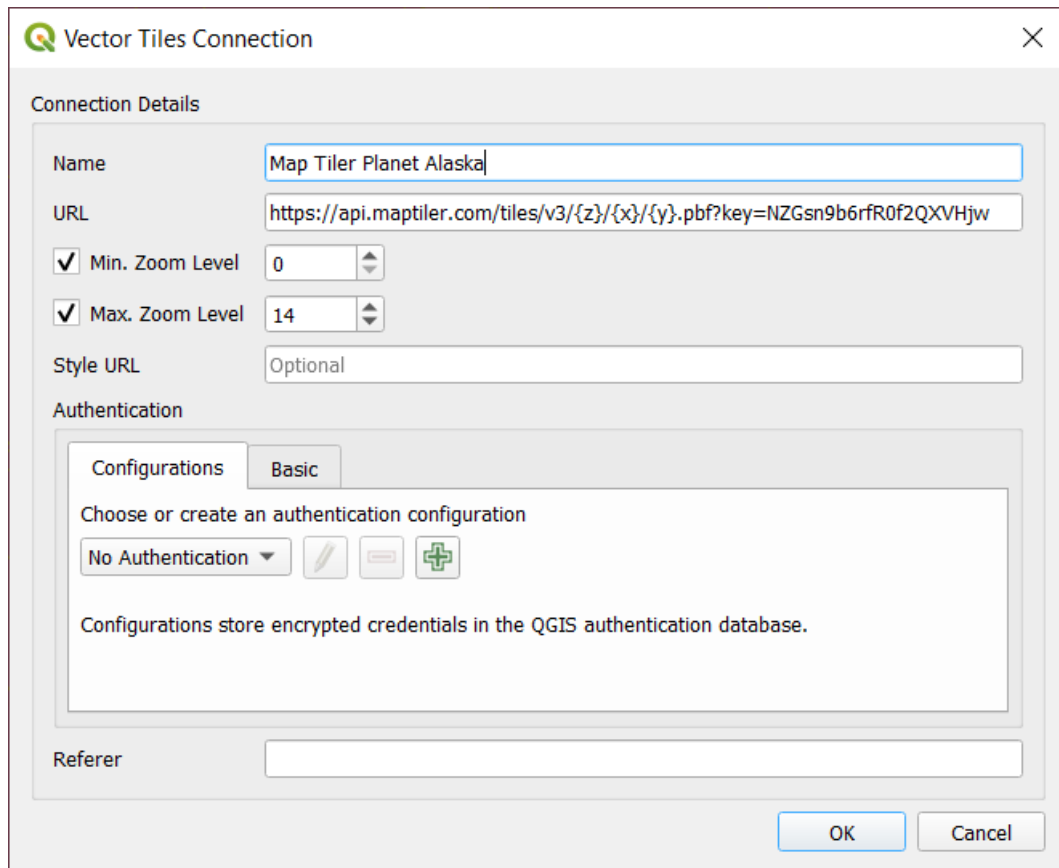



Abb. 15.17: Vector Tiles - Maptiler Planet configuration



Configurations can be saved to .XML file (*Save Connections*) through the *Vector Tiles* entry in *Data Source Manager* dialog or its context menu in the *Browser* panel. Likewise, they can be added from a file (*Load Connections*).

Once a connection to a vector tile service is set, it's possible to:

- *Edit* the vector tile connection settings
- *Remove* the connection
- From the *Browser* panel, right-click over the entry and you can also:
  - *Add layer to project*: a double-click also adds the layer
  - *View the Layer Properties...* and get access to metadata and a preview of the data provided by the service. More settings are available when the layer has been loaded into the project.

## Using XYZ Tile services

XYZ Tile services can be added via the  XYZ tab of the *Data Source Manager* dialog or the contextual menu of the *XYZ Tiles* entry in the *Browser* panel. By default, QGIS provides some default and ready-to-use XYZ Tiles services:

-  *Mapzen Global Terrain*, allowing an immediate access to global DEM source for the projects. More details and resources at <https://registry.opendata.aws/terrain-tiles/>
-  *OpenStreetMap* to access the world 2D map. Abb. 15.18 shows the dialog with the OpenStreetMap XYZ Tile service configuration.

To add a new service, press *New* (respectively *New Connection* from the *Browser* panel) and provide:

The screenshot shows the 'XYZ Connection' dialog box in QGIS. The 'Name' field is set to 'OpenStreetMap'. Under 'Connection Details', the 'URL' field contains 'https://tile.openstreetmap.org/{z}/{x}/{y}.png'. The 'Authentication' section has two tabs: 'Configurations' and 'Basic'. The 'Basic' tab is active, showing a dropdown menu set to 'No Authentication' with edit, delete, and add icons. Below this, a text box states: 'Configurations store encrypted credentials in the QGIS authentication database.' At the bottom, there are checkboxes for 'Min. Zoom Level' (set to 0) and 'Max. Zoom Level' (set to 19). Other fields include 'Referer' (empty), 'Tile Resolution' (set to 'Unknown (not scaled)'), and 'Interpretation' (set to 'Default'). At the bottom right are 'Cancel' and 'OK' buttons, and at the bottom left is a 'Help' button.

Abb. 15.18: XYZ Tiles - OpenStreetMap configuration

- einen *Namen*
- the *URL*, you can add `http://example.com/{z}/{x}/{y}.png` or `file:///local_path/{z}/{x}/{y}.png`
- the *authentication* configuration if necessary
- the *Min. Zoom level* and *Max. Zoom level*
- a *Referer*
- the *Tile Resolution*: possible values are *Unknown (not scaled)*, *Standard (256x256 / 96DPI)* and *High (512x512 / 192DPI)*
- *Interpretation*: converts WMTS/XYZ raster datasets to a raster layer of single band float type following a predefined encoding scheme. Supported schemes are *Default* (no conversion is done), *MapTiler Terrain RGB* and *Terrarium Terrain RGB*. The selected converter will translate the RGB source values to float values for each pixel. Once loaded, the layer will be presented as a single band floating point raster layer, ready for styling using QGIS usual *raster renderers*.

Press *OK* to establish the connection. It will then be possible to:

- Add the new layer to the project; it is loaded with the name given in the settings.
- Edit the XYZ connection settings
- Remove the connection
- From the *Browser* panel, right-click over the entry and you can also:
  - *Export layer... ► To File, saving it as a raster*
  - *Add layer to project*: a double-click also adds the layer
  - View the *Layer Properties...* and get access to metadata and a preview of the data provided by the service. More settings are available when the layer has been loaded into the project.

Configurations can be saved to .XML file (*Save Connections*) through the XYZ entry in *Data Source Manager* dialog or its contextual menu in the *Browser* panel. Likewise, they can be added from a file (*Load Connections*).

The XML file for OpenStreetMap looks like this:

```
<!DOCTYPE connections>
<qgsXYZTilesConnections version="1.0">
  <xyztiles url="https://tile.openstreetmap.org/{z}/{x}/{y}.png"
    zmin="0" zmax="19" tilePixelRatio="0" password="" name="OpenStreetMap"
    username="" authcfg="" referer="" />
</qgsXYZTilesConnections>
```

---

#### **Tip:** Loading XYZ tiles without creating a connection


It is also possible to add XYZ tiles to a project without necessarily storing its connection settings in your user profile (e.g. for a dataset you may need once). In the *Data Source Manager ► XYZ* tab, edit any properties in the *Connection Details* group. The *Name* field above should turn into *Custom*. Press *Add* to load the layer in the project. It will be named by default *XYZ Layer*.

---

Examples of XYZ Tile services:

- OpenStreetMap Monochrome: *URL*: `http://tiles.wmflabs.org/bw-mapnik/{z}/{x}/{y}.png`, *Min. Zoom Level*: 0, *Max. Zoom Level*: 19.
- Google Maps: *URL*: `https://mt1.google.com/vt/lyrs=m&x={x}&y={y}&z={z}`, *Min. Zoom Level*: 0, *Max. Zoom Level*: 19.
- Open Weather Map Temperature: *URL*: `http://tile.openweathermap.org/map/temp_new/{z}/{x}/{y}.png?appid={api_key}` *Min. Zoom Level*: 0, *Max. Zoom Level*: 19.

## Using ArcGIS REST Servers

ArcGIS REST Servers can be added via the  *ArcGIS REST Server* tab of the *Data Source Manager* dialog or the contextual menu of the *ArcGIS REST Servers* entry in the *Browser* panel. Press *New* (respectively *New Connection*) and provide:


- einen *Namen*
- the *URL*
- a *Community endpoint URL*
- a *Content endpoint URL*
- the *authentication* configuration if necessary
- a *Referer*

**Bemerkung:** ArcGIS Feature Service connections which have their corresponding Portal endpoint URLS set can be explored by content groups in the browser panel.

If a connection has the Portal endpoints set, then expanding out the connection in the browser will show a “Groups” and “Services” folder, instead of the full list of services usually shown. Expanding out the groups folder will show a list of all content groups that the user is a member of, each of which can be expanded to show the service items belonging to that group.

Configurations can be saved to .XML file (*Save Connections*) through the *ArcGIS REST Server* entry in *Data Source Manager* dialog. Likewise, they can be added from a file (*Load Connections*).

Once a connection to an ArcGIS REST Server is set, it's possible to:

- *Edit* the ArcGIS REST Server connection settings
- *Remove* the connection
- *Refresh* the connection
- use a filter for the available layers
- choose from a list of available layers with the option to  *Only request features overlapping the current view extent*
- From the *Browser* panel, right-click over the connection entry and you can:
  - *Refresh*
  - *Edit connection...*
  - *Remove connection...*
  - *View Service Info* which will open the default web browser and display the Service Info.
- Right-click over the layer entry and you can also:
  - *View Service Info* which will open the default web browser and display the Service Info.
  - *Export layer... ► To File*
  - *Add layer to project*: a double-click also adds the layer
  - View the *Layer Properties...* and get access to metadata and a preview of the data provided by the service. More settings are available when the layer has been loaded into the project.

### Using 3D tiles services

To load a 3D tiles into QGIS, use the  *Scene* tab in the *Data Source Manager* dialog.

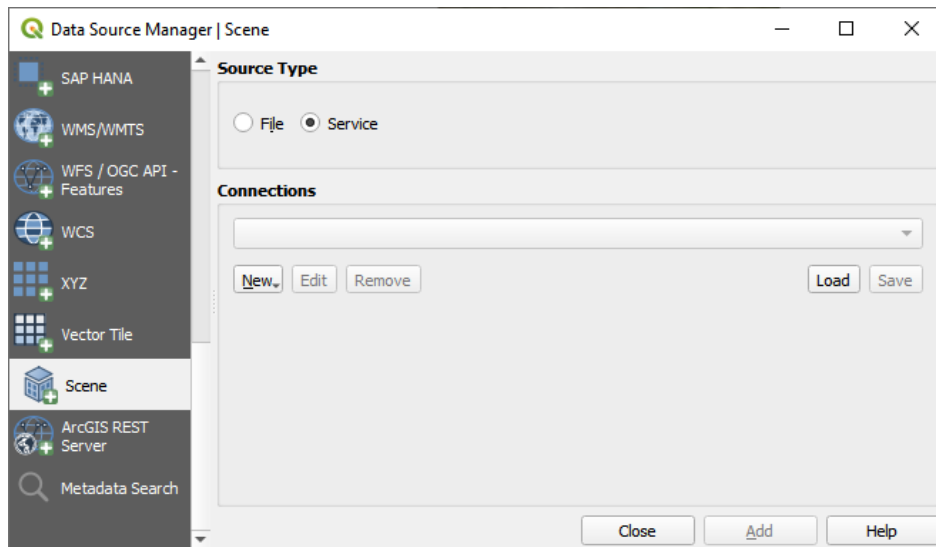


Abb. 15.19: Data Source Manager - Scene

Create a *New Cesium 3D Tiles Connection* by clicking on *New*. Add *Name* and *URL* or add local tileset file.

Support for 3D tiles:

- Remote source - `http://example.com/tileset.json`
- Local files - `file:///path/to/tiles/tileset.json`

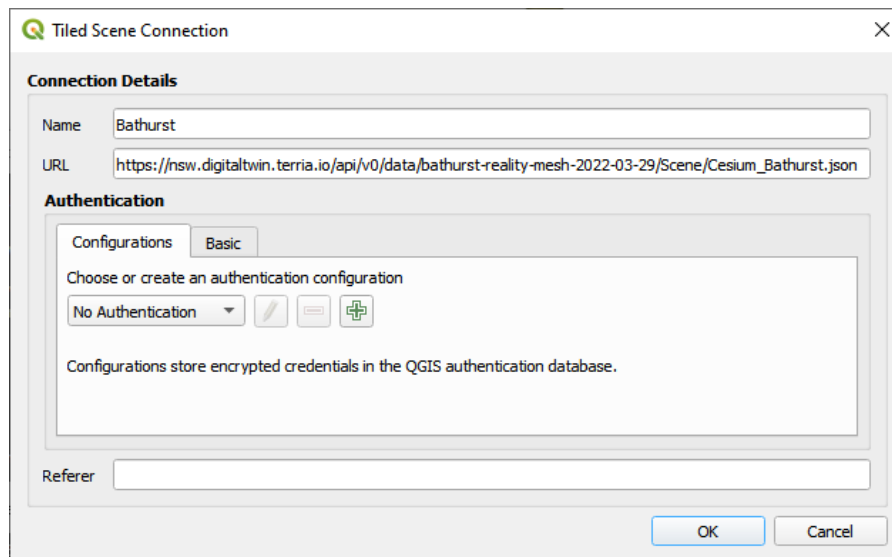


Abb. 15.20: Tiled Scene Connection

You can also add the service from *Browser Panel*.

After creating new connection you are able to *Add* the new layer to your map.

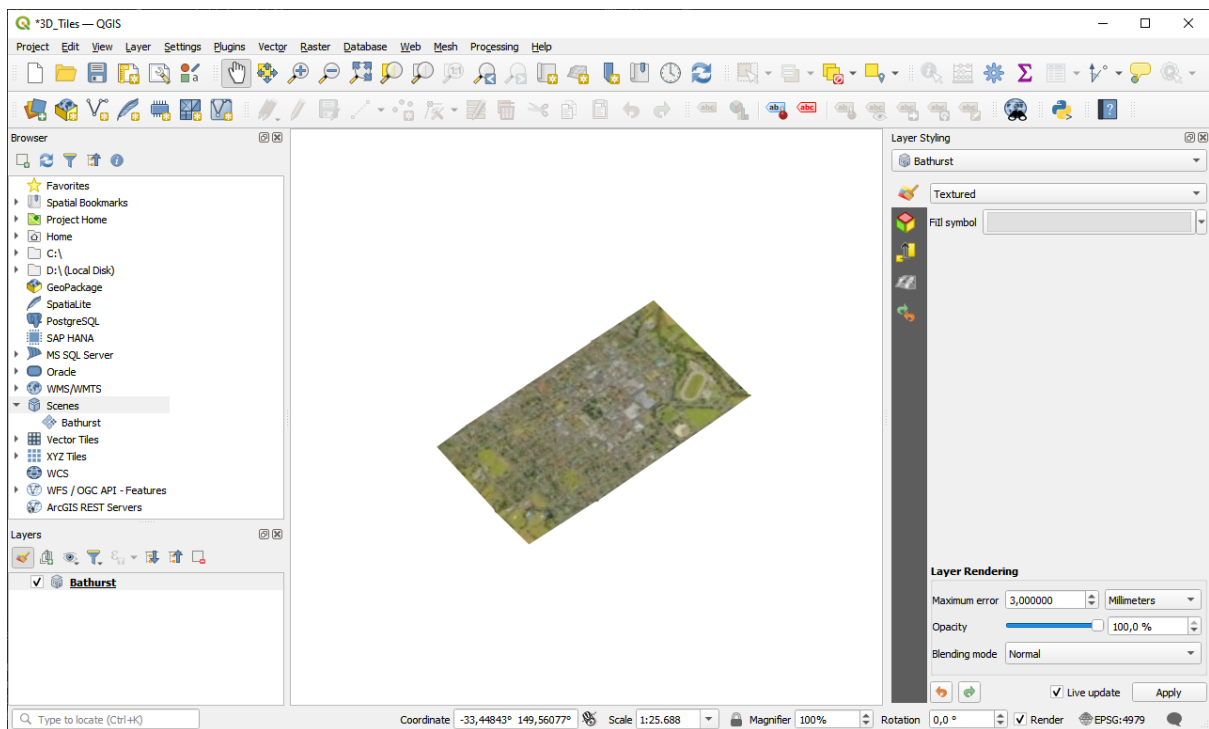


Abb. 15.21: 3D Tiles Layer - Textured

## 15.2 Erstellung von Layern

Layer können auf viele Arten erstellt werden, unter anderem:


- leere Layer von Grund auf
- Layer von vorhandenen Layern
- Layer aus der Zwischenablage
- Layer als Ergebnis einer SQL-ähnlichen Abfrage, die auf einem oder mehreren Layern basiert: die (*virtuellen Layer*)

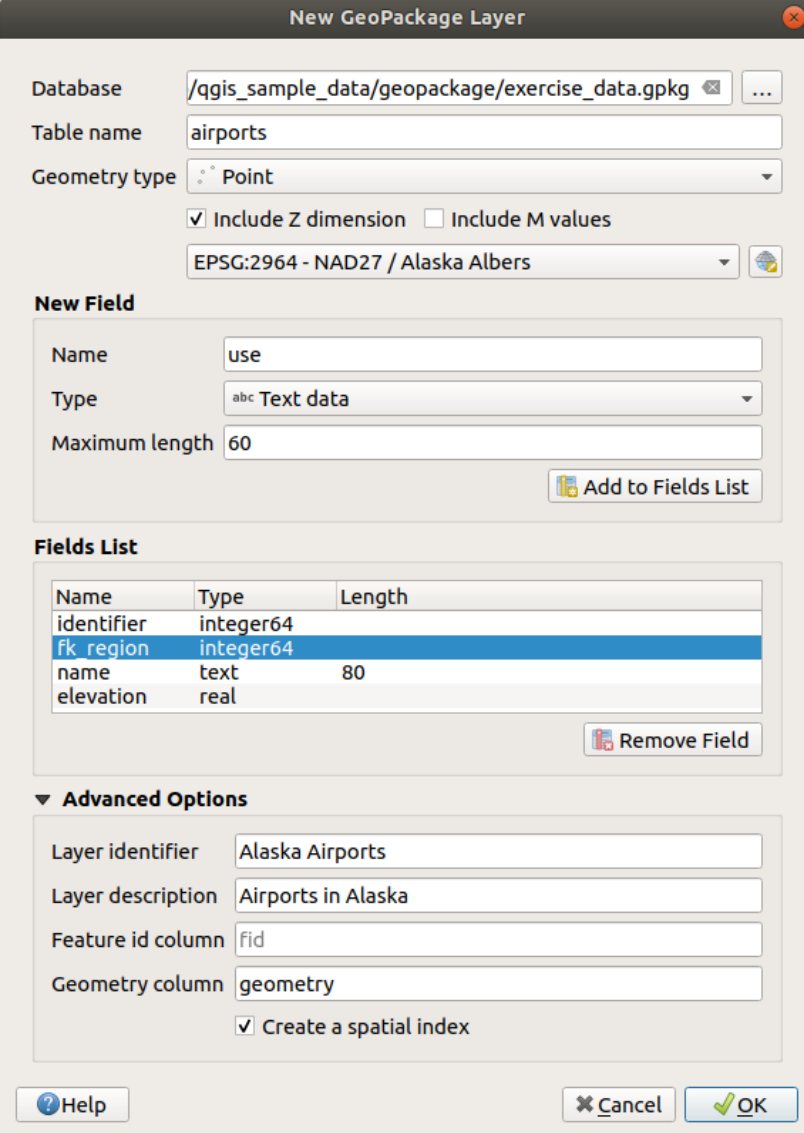
QGIS bietet auch Werkzeuge zum Importieren/Exportieren verschiedener Formate.

### 15.2.1 Neue Vektorlayer erstellen

QGIS erlaubt es Ihnen, neue Layer in verschiedenen Formaten zu erstellen. Es bietet Werkzeuge für die Erstellung von GeoPackage-, Shapefile-, SpatiaLite-, GPX-Format und flüchtige Layer (Temporärlayer). Die Erstellung eines *neuen GRASS-Layer* wird innerhalb des GRASS-Plugins unterstützt.

## Einen neuen GeoPackage-Layer erstellen

To create a new GeoPackage layer, press the  *New GeoPackage Layer...* button in the *Layer ► Create Layer ►* menu or from the *Data Source Manager* toolbar. The *New GeoPackage Layer* dialog will be displayed as shown in Abb. 15.22.



**New GeoPackage Layer**

Database: /qgis\_sample\_data/geopackage/exercise\_data.gpkg

Table name: airports

Geometry type: Point

☒ Include Z dimension ☐ Include M values

EPSG:2964 - NAD27 / Alaska Albers

**New Field**

Name: use

Type: Text data

Maximum length: 60

Add to Fields List

**Fields List**

Name	Type	Length
identifier	integer64	
fk_region	integer64	
name	text	80
elevation	real	

Remove Field

**Advanced Options**

Layer identifier: Alaska Airports

Layer description: Airports in Alaska


Feature id column: fid

Geometry column: geometry

☒ Create a spatial index


Help Cancel OK

Abb. 15.22: Dialogfenster zum Erstellen eines neuen GeoPackage-Layers

1. Der erste Schritt besteht darin, den Speicherort der Datenbankdatei anzugeben. Dies kann durch Drücken der Schaltfläche .... rechts neben dem Feld *Datenbank* erfolgen und eine vorhandene GeoPackage-Datei auswählen oder eine neue erstellen. QGIS wird automatisch die richtige Erweiterung zu dem von Ihnen angegebenen Namen hinzufügen.
2. Geben Sie dem neuen Layer/Tabelle einen Namen (*Tabellenname*)
3. Definieren Sie den *Geometriertyp*. Wenn es sich nicht um einen geometriellosen Layer handelt, können Sie angeben, ob er *Z-Dimension einschließen* und/oder *M- Werte einschließen* soll.
4. Geben Sie das Koordinatenreferenzsystem mit der Schaltfläche  an

Zum Hinzufügen von Feldern zu dem Layer, den Sie erstellen:


1. Geben Sie *Name* des Feldes ein


2. Wählen Sie die Daten *Typ*. Unterstützte Typen sind *Text*, *Ganze Zahl* (sowohl Integer als auch Integer 64 bit), *Dezimalzahl*, *Datum* und *Datum & Zeit*, *Binärobjekt (BLOB)* und *Boolean*.
3. Je nach gewähltem Datenformat geben Sie die *Maximallänge* der Werte ein.
4. Klicken Sie auf die Schaltfläche  *Zur Feldliste hinzufügen*.
5. Wiederholen Sie die obigen Schritte für jedes Feld, das Sie hinzufügen müssen
6. Wenn Sie mit den Attributen zufrieden sind, klicken Sie auf *OK*. QGIS wird den neuen Layer zur Legende hinzufügen, und Sie können ihn wie in Abschnitt *Einen vorhandenen Layer editieren* beschrieben bearbeiten.

Standardmäßig erzeugt QGIS beim Erstellen eines GeoPackage-Layers eine *Objektkennungsspalte* namens *fid*, die als Primärschlüssel des Layers fungiert. Der Name kann geändert werden. Das Geometriefeld, falls verfügbar, heißt *geometry*, und Sie können wählen, ob Sie darauf einen räumlichen Index erstellen wollen. Diese Optionen sind unter dem *Erweiterte Optionen* zusammen mit dem :guilabel: *Layerkennung* (kurzer menschenlesbarer Name des Layers) und der :guilabel: *Layerbeschreibung* zu finden.

Die weitere Verwaltung von GeoPackage-Layern kann mit dem *DB-Verwaltung* erfolgen.

### Einen neuen Shapedatei-Layer erstellen

To create a new ESRI Shapefile format layer, press the  *New Shapefile Layer...* button in the *Layer ► Create Layer ►* menu or from the *Data Source Manager* toolbar. The *New Shapefile Layer* dialog will be displayed as shown in *Abb. 15.23*.

1. Provide a path and file name using the ... button next to *File name*. QGIS will automatically add the right extension to the name you provide.
2. Next, indicate the *File encoding* of the data
3. Choose the *Geometry type* of the layer: No Geometry (resulting in a .DBF format file), point, multipoint, line or polygon
4. Specify whether the geometry should have additional dimensions: *None*, *Z (+ M values)* or *M values*
5. Geben Sie das Koordinatenreferenzsystem mit der Schaltfläche  an



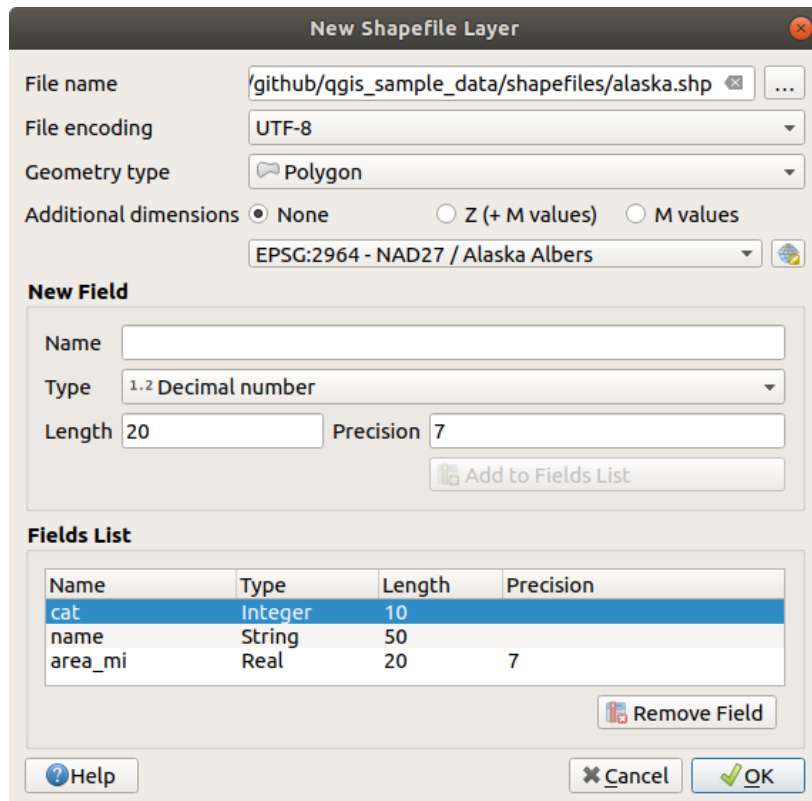



Abb. 15.23: Dialogfenster zum Erstellen eines neuen Shapedatei-Layers

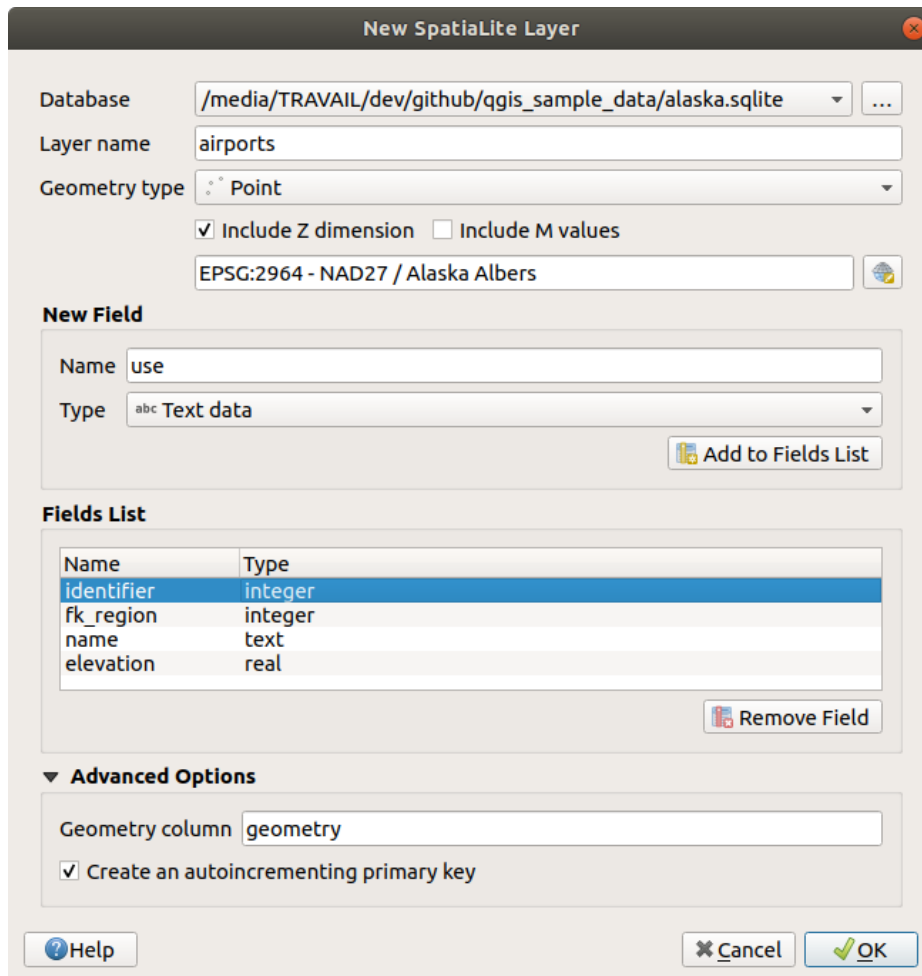
Zum Hinzufügen von Feldern zu dem Layer, den Sie erstellen:

1. Geben Sie *Name* des Feldes ein
2. Select the data *Type*. Only *Decimal number*, *Whole number*, *Text data* and *Date* attributes are supported.
3. Depending on the selected data format, enter the *Length* and *Precision*.
4. Klicken Sie auf die Schaltfläche  *Zur Feldliste hinzufügen*.
5. Wiederholen Sie die obigen Schritte für jedes Feld, das Sie hinzufügen müssen
6. Wenn Sie mit den Attributen zufrieden sind, klicken Sie auf *OK*. QGIS wird den neuen Layer zur Legende hinzufügen, und Sie können ihn wie in Abschnitt *Einen vorhandenen Layer editieren* beschrieben bearbeiten.

By default, a first integer *id* column is added but can be removed.

## Creating a new SpatiaLite layer

To create a new SpatiaLite layer, press the  *New SpatiaLite Layer...* button in the *Layer ► Create Layer ►* menu or from the *Data Source Manager* toolbar. The *New SpatiaLite Layer* dialog will be displayed as shown in Abb. 15.24.




**New SpatiaLite Layer**

Database:  ...

Layer name:

Geometry type:


☒ Include Z dimension ☐ Include M values



**New Field**


Name:

Type:

 Add to Fields List

**Fields List**

Name	Type
identifier	integer
fk_region	integer
name	text
elevation	real

 Remove Field

**Advanced Options**

Geometry column:

☒ Create an autoincrementing primary key






 Help  Cancel  OK


Abb. 15.24: Creating a New SpatiaLite layer dialog

1. The first step is to indicate the database file location. This can be done by pressing the ... button to the right of the *Database* field and select an existing SpatiaLite file or create a new one. QGIS will automatically add the right extension to the name you provide.
2. Provide a name (*Layer name*) for the new layer
3. Definieren Sie den *Geometriertyp*. Wenn es sich nicht um einen geometriellosen Layer handelt, können Sie angeben, ob er *Z-Dimension einschließen* und/oder *M-Werte einschließen* soll.
4. Specify the coordinate reference system using the  button.

Zum Hinzufügen von Feldern zu dem Layer, den Sie erstellen:


1. Geben Sie *Name* des Feldes ein
2. Select the data *Type*. Supported types are *Text data*, *Whole number*, *Decimal number*, *Date* and *Date time*.
3. Klicken Sie auf die Schaltfläche  *Zur Feldliste hinzufügen*.
4. Wiederholen Sie die obigen Schritte für jedes Feld, das Sie hinzufügen müssen

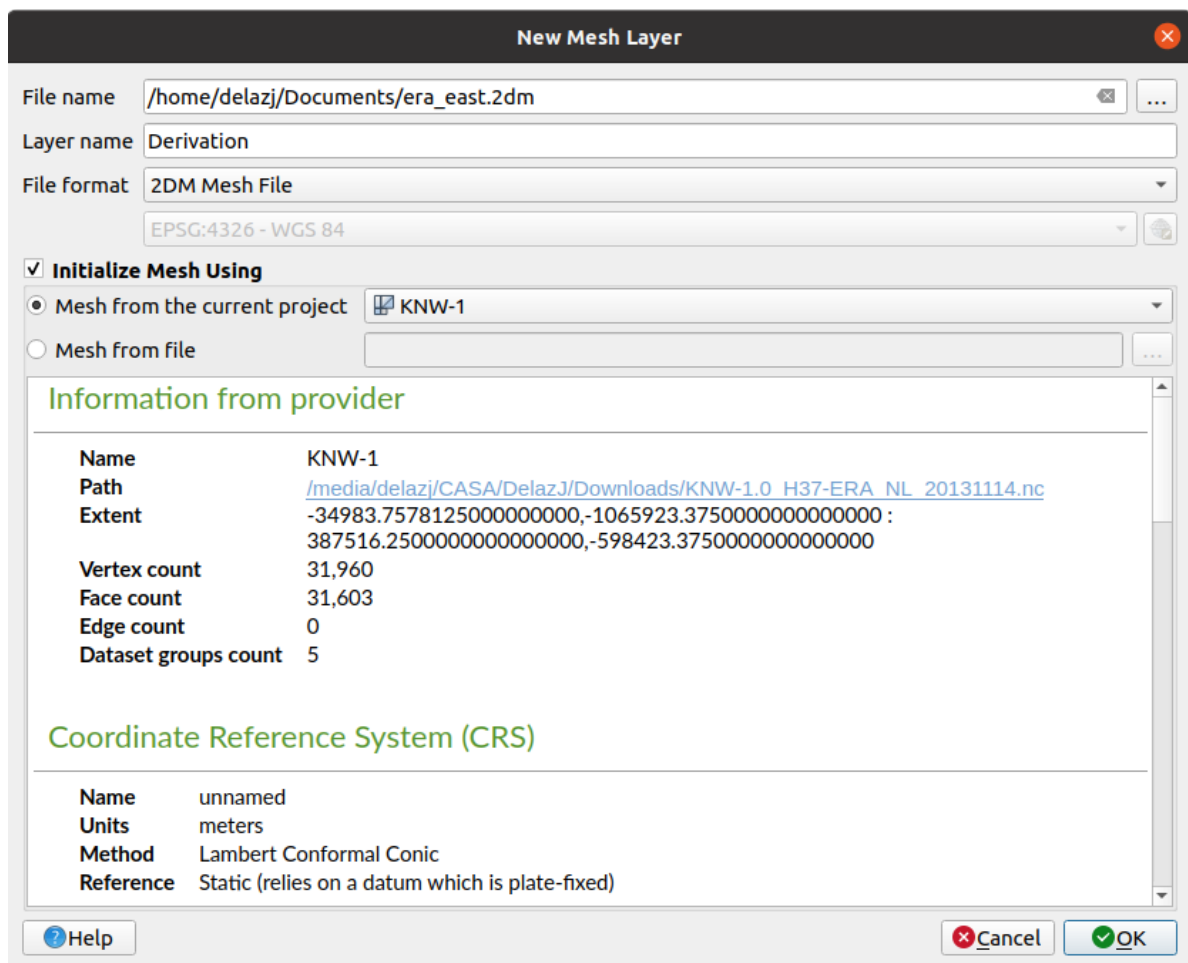
- Wenn Sie mit den Attributen zufrieden sind, klicken Sie auf *OK*. QGIS wird den neuen Layer zur Legende hinzufügen, und Sie können ihn wie in Abschnitt *Einen vorhandenen Layer editieren* beschrieben bearbeiten.

If desired, you can select  *Create an autoincrementing primary key* under the guilabel:Advanced Options section. You can also rename the *Geometry column* (geometry by default).



Further management of SpatiaLite layers can be done with *DB Manager*.

## Creating a new Mesh layer

To create a new Mesh layer, press the  *New Mesh Layer...* button in the *Layer ► Create Layer ►* menu or from the *Data Source Manager* toolbar. The *New Mesh Layer* dialog will be displayed as shown in Abb. 15.25.




**New Mesh Layer**

File name:   


Layer name:

File format:



☒ **Initialize Mesh Using**

☒ Mesh from the current project

☐ Mesh from file  

**Information from provider**

Name	KNW-1
Path	<a href="#">/media/delazj/CASA/DelazJ/Downloads/KNW-1.0_H37-ERA_NL_20131114.nc</a>
Extent	-34983.7578125000000000,-1065923.3750000000000000 : 387516.2500000000000000,-598423.3750000000000000
Vertex count	31,960
Face count	31,603
Edge count	0
Dataset groups count	5

**Coordinate Reference System (CRS)**

Name	unnamed
Units	meters
Method	Lambert Conformal Conic
Reference	Static (relies on a datum which is plate-fixed)


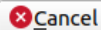


 Help  

Abb. 15.25: Creating a New Mesh layer dialog


- The first step is to indicate the mesh file location. This can be done by pressing the ... button to the right of the *File name* field and select an existing mesh file or create a new one.
- Provide a name (*Layer name*), i.e. the name the layer is displayed with in the *Layers* panel
- Select the *File format*: currently supported mesh file formats are 2DM Mesh File (\*.2dm), Selafin File (\*.slf) and UGRID (\*.nc).
- Indicate the *Coordinate Reference System* to assign to the dataset
- The above steps will generate an empty layer that you can afterwards digitize vertices and add dataset groups to. It's however also possible to initialize the layer with an existing mesh layer, i.e. populate the new layer with vertices or faces from the other. To do so:

1. Check  *Initialize Mesh using*
2. and select either a *Mesh from the current project* or *Mesh from a file*. Informations on the selected mesh file are displayed for checkup.

Note that only the frame of the mesh layer is transferred to the new layer; their datasets are not copied.

### Creating a new GPX layer



To create a new GPX file:

1. Select *Create Layer* ►  *New GPX Layer...* from the *Layer* menu.
2. In the dialog, choose where to save the new file, name it and press *Save*.
3. Three new layers are added to the *Layers Panel*:
  - a point layer to digitize locations (waypoints) with fields storing the name, elevation, comment, description, source, url and url name
  - a line layer to digitize sequences of locations that make up a planned route (routes) with fields storing the name, symbol, number, comment, description, source, url, url name
  - and a line layer to track the receiver's movement over time (tracks) with fields storing the name, symbol, number, comment, description, source, url, url name.
4. You can now edit any of them as described in section *Einen vorhandenen Layer editieren*.

### Creating a new Temporary Scratch Layer

Temporary Scratch Layers are in-memory layers, meaning that they are not saved on disk and will be discarded when QGIS is closed. They can be handy for storing features you temporarily need or as intermediate layers during geoprocessing operations.

To create a new Temporary Scratch layer, choose the  *New Temporary Scratch Layer...* entry in the *Layer* ► *Create Layer* ► menu or in the *Data Source Manager* toolbar. The *New Temporary Scratch Layer* dialog will be displayed as shown in Abb. 15.26. Then:

1. Provide the *Layer name*
2. Select the *Geometry type*. Here you can create a:
  - No geometry type layer, served as simple table,
  - Point or MultiPoint layer,
  - LineString/CompoundCurve or MultiLineString/MultiCurve layer,
  - Polygon/CurvePolygon or MultiPolygon/MultiSurface layer.
3. For geometric types, specify the dimensions of the dataset: check whether it should *Include Z dimension* and/or *Include M values*
4. Specify the coordinate reference system using the  button.
5. Add fields to the layer. Note that unlike many formats, temporary layers can be created without any fields. This step is thus optional.
  1. Geben Sie *Name* des Feldes ein
  2. Select the data *Type*: *Text*, *Whole number*, *Decimal number*, *Boolean*, *Date*, *Time*, *Date & Time* and *Binary (BLOB)* are supported.
  3. Depending on the selected data format, enter the *Length* and *Precision*
  4. Klicken Sie auf die Schaltfläche  *Zur Feldliste hinzufügen*.

5. Repeat the steps above for each field you need to add
6. Once you are happy with the settings, click *OK*. QGIS will add the new layer to the *Layers* panel, and you can edit it as described in section *Einen vorhandenen Layer editieren*.

**New Temporary Scratch Layer**

Layer name:

Geometry type:

☒ Include Z dimension ☐ Include M values

EPSG:4326 - WGS 84

**New Field**

Name:

Type:

Length:  Precision:

**Fields List**

Name	Type	Length	Precision
id	integer		
name	string	20	


**Warning:** Temporary scratch layers are not saved and will be discarded when QGIS is closed.

Abb. 15.26: Creating a new Temporary Scratch layer dialog

You can also create prepopulated temporary scratch layers using e.g. the clipboard (see *Creating new layers from the clipboard*) or as a result of a *Processing algorithm*.

### **Tipp: Permanently store a memory layer on disk**

To avoid data loss when closing a project with temporary scratch layers, you can save these layers to any vector format supported by QGIS:

- clicking the  indicator icon next to the layer;
- selecting the *Make permanent* entry in the layer contextual menu;
- using the *Export ►* entry from the contextual menu or the *Layer ► Save As...* menu.

Each of these commands opens the *Save Vector Layer as* dialog described in the *Creating new layers from an existing layer* section and the saved file replaces the temporary one in the *Layers* panel.

## 15.2.2 Creating new layers from an existing layer

Both raster and vector layers can be saved in a different format and/or reprojected to a different coordinate reference system (CRS) using the *Layer ► Save As...* menu or right-clicking on the layer in the *Layers panel* and selecting:

- *Export ► Save As...* for raster layers
- *Export ► Save Features As...* or *Export ► Save Selected Features As...* for vector layers.
- Drag and drop the layer from the layer tree to the PostGIS entry in the *Browser Panel*. Note that you must have a PostGIS connection in the *Browser Panel*.

### Common parameters


The *Save Layer as...* dialog shows several parameters to change the behavior when saving the layer. Among the common parameters for raster and vector are:

- *File name*: the location of the file on the disk. It can refer to the output layer or to a container that stores the layer (for example database-like formats such as GeoPackage, SpatiaLite or Open Document Spreadsheets).
- *CRS*: can be changed to reproject the data
- *Extent*: restricts the extent of the input that is to be exported using the *extent\_selector* widget
- *Add saved file to map*: to add the new layer to the canvas

However, some parameters are specific to raster and vector formats:

### Raster specific parameters

Depending on the format of export, some of these options may not be available:

- *Output mode* (it can be **raw data** or **rendered image**)
- *Format*: exports to any raster format GDAL can write to, such as GeoTiff, GeoPackage, MBTiles, Geospatial PDF, SAGA GIS Binary Grid, Intergraph Raster, ESRI .hdr Labelled...
- *Resolution*
- *Create Options*: use advanced options (file compression, block sizes, colorimetry...) when generating files, either from the *predefined create profiles* related to the output format or by setting each parameter.
- *Pyramids creation*
- *VRT Tiles* in case you opted to  *Create VRT*
- *No data values*

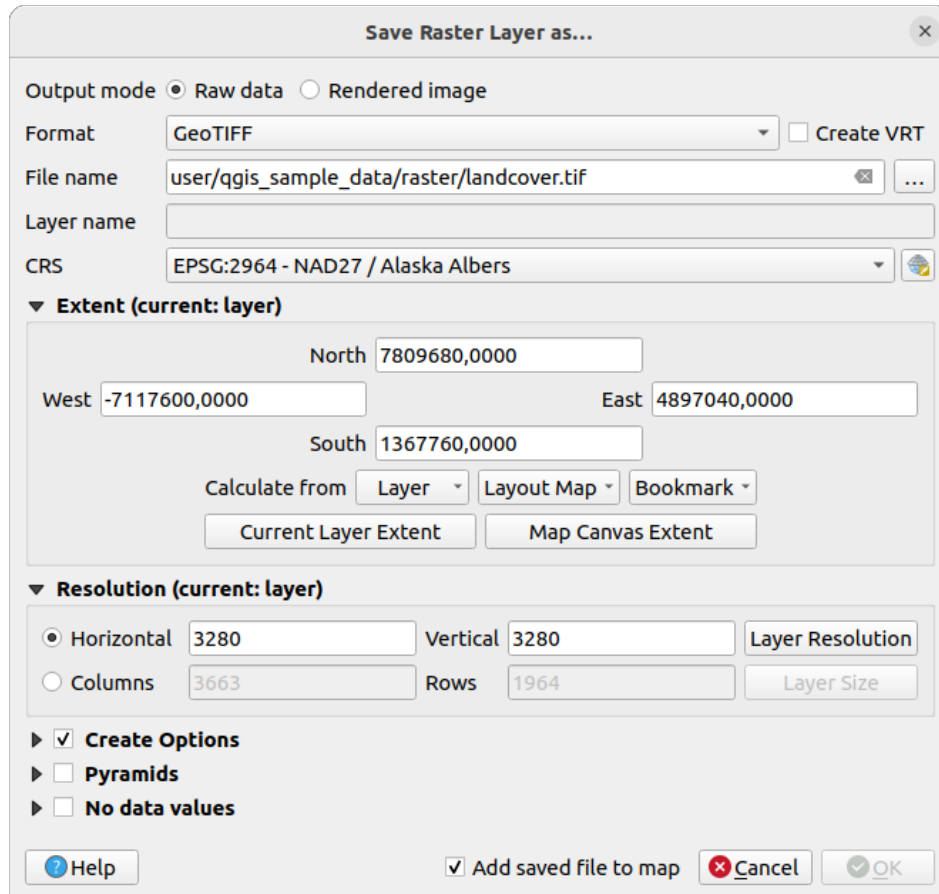


Abb. 15.27: Saving as a new raster layer

## Vector specific parameters

Depending on the format of export, some of these options may be available:

- *Format*: exports to any vector format GDAL can write to, such as GeoPackage, GML, ESRI Shapefile, AutoCAD DXF, ESRI FileGDB, Mapinfo TAB or MIF, SpatiaLite, CSV, KML, ODS, ...
- *Layer name*: available when the *File name* refers to a container-like format, this entry represents the output layer.
- *Encoding*
- *Save only selected features*
- *Select fields to export and their export options*: provides means to export fields with custom names and *form widget* settings:
  - Check rows under the *Name* column to choose fields to keep in the output layer, or press *Select All* or *Deselect All* buttons
  - Toggle the *Use aliases for exported name* checkbox to populate the *Export name* column with corresponding field aliases or reset to the original field name. Double-clicking a cell will also edit the name.
  - Depending on whether attribute form custom widgets are in use, you can *Replace all selected raw field values by displayed values*. E.g. if a *value map* widget is applied to a field, the output layer will contain the description values instead of the original values. The replacement can also be done on a field by field basis, in the *Replace with displayed values* column.
- *Persist layer metadata*: ensures that any layer *metadata* present in the source layer will be copied and stored:

- in the newly created layer, if the output is of GeoPackage format
- as a `.qmd` file along with the output layer, for other formats. Note that file-based formats supporting more than one dataset (e.g. SpatiaLite, DXF,...) may have unintended behavior.
- *Symbology export*: can be used mainly for DXF export and for all file formats who manage OGR feature styles (see note below) as DXF, KML, tab file formats:
  - **No symbology**: default style of the application that reads the data
  - **Feature symbology**: save style with OGR Feature Styles (see note below)
  - **Symbol Layer symbology**: save with OGR Feature Styles (see note below) but export the same geometry multiple times if there are multiple symbology symbol layers used
  - A **Scale** value can be applied to the latest options


---

**Bemerkung:** *OGR Feature Styles* are a way to store style directly in the data as a hidden attribute. Only some formats can handle this kind of information. KML, DXF and TAB file formats are such formats. For advanced details, you can read the [OGR Feature Styles specification](#) document.

---

- *Geometry*: you can configure the geometry capabilities of the output layer
  - *geometry type*: keeps the original geometry of the features when set to **Automatic**, otherwise removes or overrides it with any type. You can add an empty geometry column to an attribute table and remove the geometry column of a spatial layer.
  - *Force multi-type*: forces creation of multi-geometry features in the layer.
  - *Include z-dimension* to geometries.

---

**Tipp:** Overriding layer geometry type makes it possible to do things like save a geometryless table (e.g. `.csv` file) into a shapefile WITH any type of geometry (point, line, polygon), so that geometries can then be manually added to rows with the  Add Part tool.

---

- *Datasource Options*, *Layer Options* or *Custom Options* which allow you to configure advanced parameters depending on the output format. Some are described in [Exploring Data Formats and Fields](#) but for full details, see the [GDAL](#) driver documentation. Each file format has its own custom parameters, e.g. for the GeoJSON format have a look at the [GDAL GeoJSON](#) documentation.



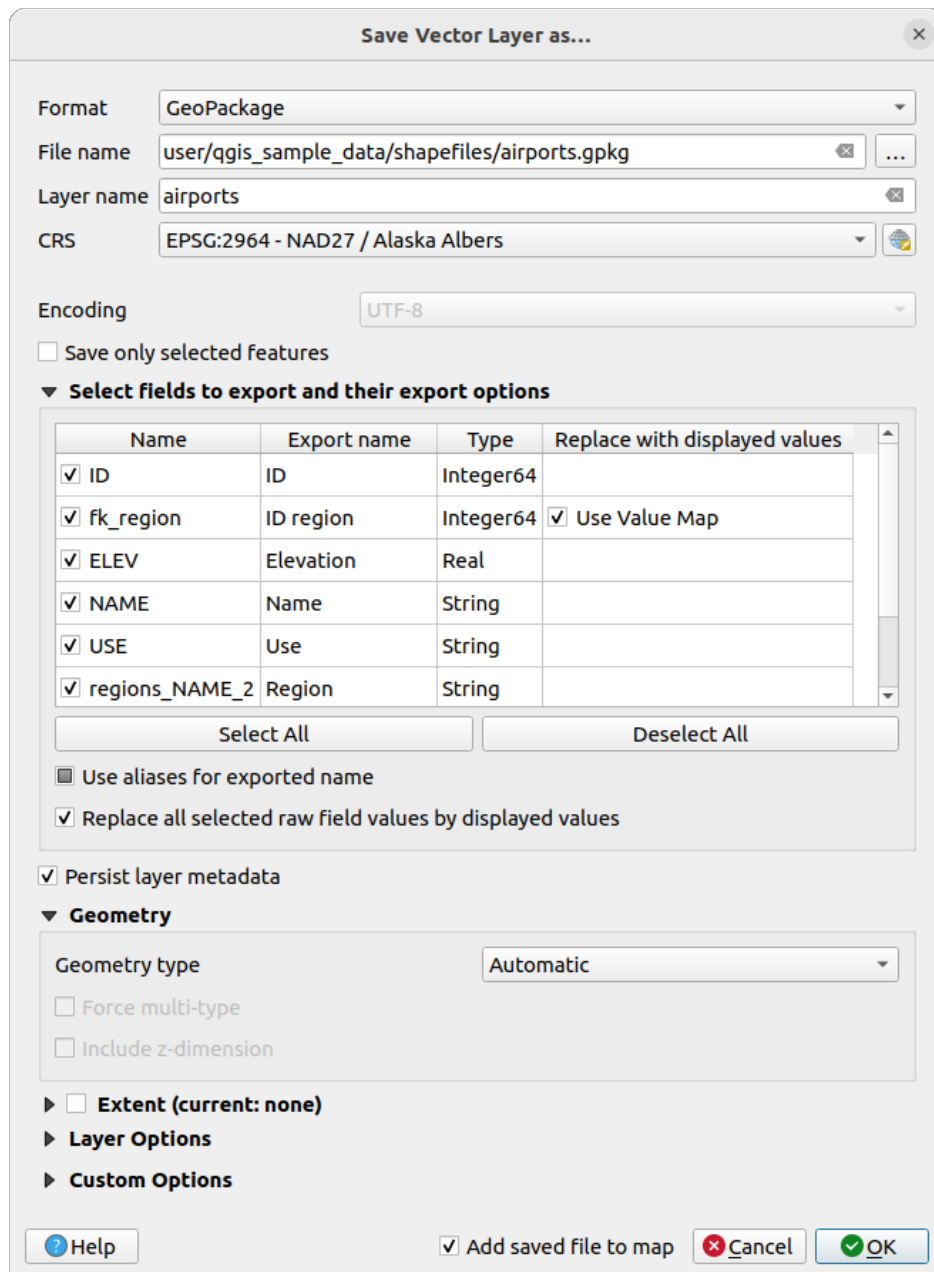


Abb. 15.28: Saving as a new vector layer

When saving a vector layer into an existing file, depending on the capabilities of the output format (Geopackage, SpatiaLite, FileGDB...), the user can decide whether to:

- overwrite the whole file
- overwrite only the target layer (the layer name is configurable)
- append features to the existing target layer
- append features, add new fields if there are any.

For formats like ESRI Shapefile, MapInfo .tab, feature append is also available.

### 15.2.3 Creating new DXF files

Besides the *Save As...* dialog which provides options to export a single layer to another format, including \*.DXF, QGIS provides another tool to export multiple layers as a single DXF layer. It's accessible in the *Project ► Import/Export ► Export Project to DXF...* menu.

In the *DXF Export* dialog:

1. Provide the destination file.
2. Choose the symbology mode and scale (see the *OGR Feature Styles* note), if applicable.
3. Select the data *Encoding*.
4. Select the *CRS* to apply: the selected layers will be reprojected to the given CRS.
5. Select the layers to include in the DXF files either by checking them in the table widget or automatically picking them from an existing *map theme*. The *Select All* and *Deselect All* buttons can help to quickly set the data to export.

For each layer, you can choose whether to export all the features in a single DXF layer or rely on a field whose values are used to split the features into layers in the DXF output.

Optionally, you can also choose to:

- ☒ *Use the layer title as name if set* instead of the layer name itself;
- ☒ *Export features intersecting the current map extent*;
- ☐ *Force 2d output (eg. to support polyline width)*;
- ☒ *Export label as MTEXT elements* or TEXT elements.

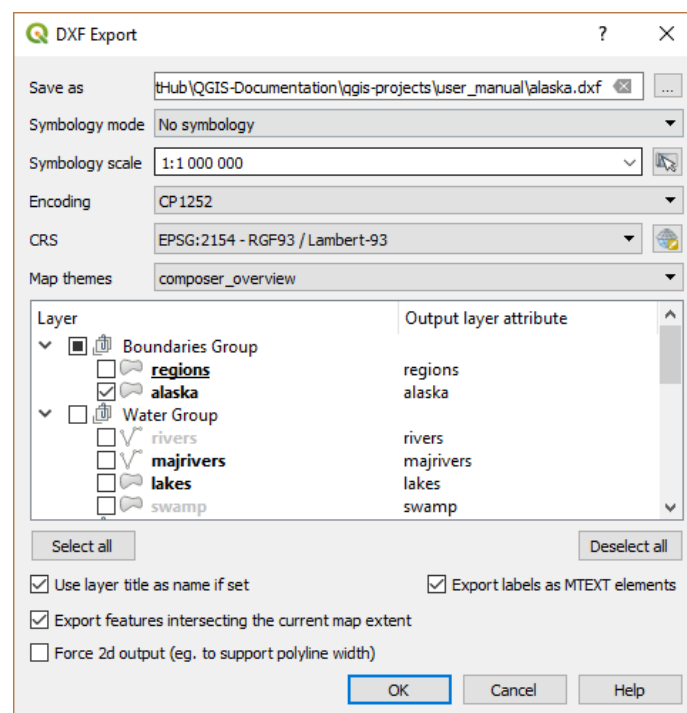


Abb. 15.29: Exporting a project to DXF dialog

## 15.2.4 Creating new layers from the clipboard

Features that are on the clipboard can be pasted into a new layer. To do this, Select some features, copy them to the clipboard, and then paste them into a new layer using *Edit ► Paste Features as ►* and choosing:

- *New Vector Layer...*: the *Save vector layer as...* dialog appears (see *Creating new layers from an existing layer* for parameters)
- or *Temporary Scratch Layer...*: you need to provide a name for the layer

A new layer, filled with selected features and their attributes is created (and added to map canvas).

---



**Bemerkung:** Creating layers from the clipboard is possible with features selected and copied within QGIS as well as features from another application, as long as their geometries are defined using well-known text (WKT).

---

## 15.2.5 Creating virtual layers

A virtual layer is a special kind of vector layer. It allows you to define a layer as the result of an SQL query involving any number of other vector layers that QGIS is able to open. Virtual layers do not carry data by themselves and can be seen as views.

To create a virtual layer, open the virtual layer creation dialog by:

- choosing the  *Add/Edit Virtual Layer* entry in the *Layer ► Add Layer ►* menu;
- enabling the  *Add Virtual Layer* tab in the *Data Source Manager* dialog;
- or using the *DB Manager* dialog tree.

The dialog allows you to specify a *Layer name* and an *SQL Query*. The query can use the name (or id) of loaded vector layers as tables, as well as their field names as columns.

For example, if you have a layer called `airports`, you can create a new virtual layer called `public_airports` with an SQL query like:

```
SELECT *
FROM airports
WHERE USE = "Civilian/Public"
```

The SQL query will be executed, regardless of the underlying provider of the `airports` layer, even if this provider does not directly support SQL queries.

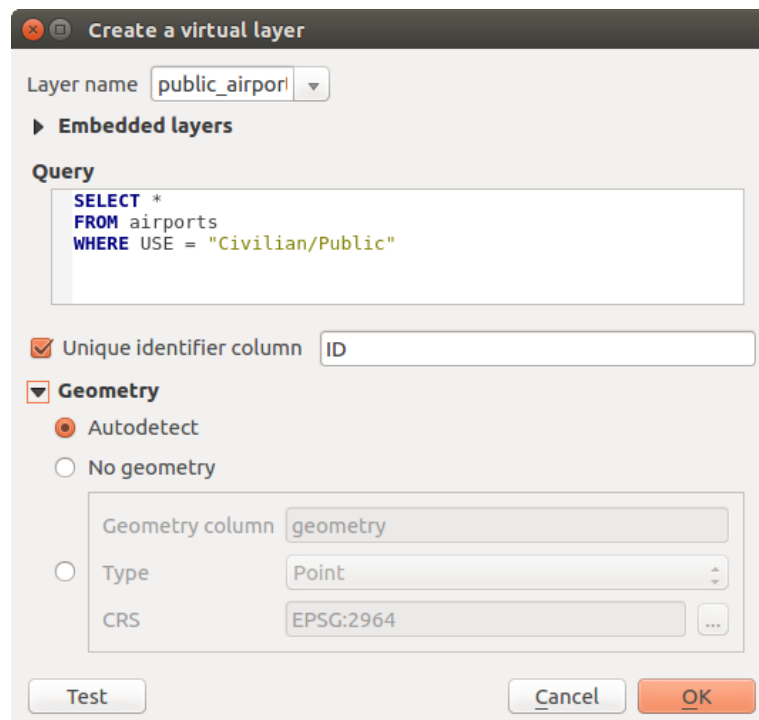


Abb. 15.30: Create virtual layers dialog

Joins and complex queries can also be created, for example, to join airports and country information:

```
SELECT airports.*, country.population
FROM airports
JOIN country
ON airports.country = country.name
```

**Bemerkung:** It's also possible to create virtual layers using the SQL window of *DB Manager Plugin*.

## Embedding layers for use in queries

Besides the vector layers available in the map canvas, the user can add layers to the *Embedded layers* list, which can be used in queries without the need to have them showing in the map canvas or Layers panel.

To embed a layer, click *Add* and provide the *Local name*, *Provider*, *Encoding* and the path to the *Source*.

The *Import* button allows adding layers in the map canvas into the Embedded layers list. Those layers can then be removed from the Layers panel without breaking existent queries.

## Supported query language

The underlying engine uses SQLite and SpatiaLite to operate.

It means you can use all of the SQL your local installation of SQLite understands.

Functions from SQLite and spatial functions from SpatiaLite can also be used in a virtual layer query. For instance, creating a point layer out of an attribute-only layer can be done with a query similar to:

```
SELECT id, MakePoint(x, y, 4326) as geometry
FROM coordinates
```

*Functions of QGIS expressions* can also be used in a virtual layer query.

To refer the geometry column of a layer, use the name `geometry`.

Contrary to a pure SQL query, all the fields of a virtual layer query must be named. Don't forget to use the `as` keyword to name your columns if they are the result of a computation or a function call.

## Performance issues

With default parameters, the virtual layer engine will try its best to detect the type of the different columns of the query, including the type of the geometry column if one is present.

This is done by introspecting the query when possible or by fetching the first row of the query (LIMIT 1) as a last resort. Fetching the first row of the result just to create the layer may be undesirable for performance reasons.

The creation dialog parameters:

- *Unique identifier column*: specifies a field of the query that represents unique integer values that QGIS can use as row identifiers. By default, an autoincrementing integer value is used. Defining a unique identifier column speeds up the selection of rows by id.
- *No geometry*: forces the virtual layer to ignore any geometry field. The resulting layer is an attribute-only layer.
- *Geometry Column*: specifies the name of the geometry column.
- *Geometry Type*: specifies the type of the geometry.
- *Geometry CRS*: specifies the coordinate reference system of the virtual layer.

## Special comments

The virtual layer engine tries to determine the type of each column of the query. If it fails, the first row of the query is fetched to determine column types.

The type of a particular column can be specified directly in the query by using some special comments.

The syntax is the following: `/*:type*/`. It has to be placed just after the name of a column. `type` can be either `int` for integers, `real` for floating point numbers or `text`.

For instance:

```
SELECT id+1 as nid /*:int*/
FROM table
```

The type and coordinate reference system of the geometry column can also be set thanks to special comments with the following syntax `/*:gtype:srid*/` where `gtype` is the geometry type (`point`, `linestring`, `polygon`, `multipoint`, `multilinestring` or `multipolygon`) and `srid` an integer representing the EPSG code of a coordinate reference system.

## Use of indexes

When requesting a layer through a virtual layer, the source layer indices will be used in the following ways:

- if an `=` predicate is used on the primary key column of the layer, the underlying data provider will be asked for a particular id (FilterFid)
- for any other predicates (`>`, `<=`, `!=`, etc.) or on a column without a primary key, a request built from an expression will be used to request the underlying vector data provider. It means indexes may be used on database providers if they exist.

A specific syntax exists to handle spatial predicates in requests and triggers the use of a spatial index: a hidden column named `_search_frame_` exists for each virtual layer. This column can be compared for equality to a bounding box. Example:

```
SELECT *
FROM vtab
WHERE _search_frame_=BuildMbr(-2.10,49.38,-1.3,49.99,4326)
```

Spatial binary predicates like `ST_Intersects` are sped up significantly when used in conjunction with this spatial index syntax.

## 15.3 Exploring Data Formats and Fields

### 15.3.1 Raster data

GIS raster data are matrices of discrete cells that represent features / phenomena on, above or below the earth's surface. Each cell in the raster grid has the same size, and cells are usually rectangular (in QGIS they will always be rectangular). Typical raster datasets include remote sensing data, such as aerial photography, or satellite imagery and modelled data, such as elevation or temperature.

Unlike vector data, raster data typically do not have an associated database record for each cell. They are geocoded by pixel resolution and the X/Y coordinate of a corner pixel of the raster layer. This allows QGIS to position the data correctly on the map canvas.

The GeoPackage format is convenient for storing raster data when working with QGIS. The popular and powerful GeoTiff format is a good alternative.

QGIS makes use of georeference information inside the raster layer (e.g., GeoTiff) or an associated *world file* to properly display the data.

### 15.3.2 Vektordaten

Many of the features and tools available in QGIS work the same, regardless the vector data source. However, because of the differences in format specifications (GeoPackage, ESRI Shapefile, MapInfo and MicroStation file formats, AutoCAD DXF, PostGIS, SpatiaLite, Oracle Spatial, MS SQL Server, SAP HANA Spatial databases and many more), QGIS may handle some of their properties differently. Support is provided by the [GDAL vector drivers](#). This section describes how to work with these specifics.

---

**Bemerkung:** QGIS supports (multi)point, (multi)line, (multi)polygon, CircularString, CompoundCurve, CurvePolygon, MultiCurve, MultiSurface feature types, all optionally with Z and/or M values.

You should also note that some drivers don't support some of these feature types, like CircularString, CompoundCurve, CurvePolygon, MultiCurve, MultiSurface feature type. QGIS will convert them.

---

### GeoPackage

The [GeoPackage](#) (GPKG) format is platform-independent, and is implemented as a SQLite database container, and can be used to store both vector and raster data. The format was defined by the Open Geospatial Consortium (OGC), and was published in 2014.

GeoPackage can be used to store the following in a SQLite database:

- **vector** features
- **tile matrix sets of imagery** and **raster** maps
- attributes (non-spatial data)
- extensions

Since QGIS version 3.8, GeoPackage can also store QGIS projects. GeoPackage layers can have JSON fields. GeoPackage is the default format for vector data in QGIS.

## ESRI Shapefile format

The ESRI Shapefile format is still one of the most used vector file formats, even if it has some limitations compared to for instance GeoPackage and Spatialite.

An ESRI Shapefile format dataset consists of several files. The following three are required:

1. `.shp` file containing the feature geometries
2. `.dbf` file containing the attributes in dBase format
3. `.shx` index file

An ESRI Shapefile format dataset can also include a file with a `.prj` suffix, which contains projection information. While it is very useful to have a projection file, it is not mandatory. A Shapefile format dataset can contain additional files. For further details, see the the [ESRI technical specification](#).

GDAL has read-write support for compressed ESRI Shapefile format (`shz` and `shp.zip`).


### Improving Performance for ESRI Shapefile format datasets

To improve the drawing performance for an ESRI Shapefile format dataset, you can create a spatial index. A spatial index will improve the speed of both zooming and panning. Spatial indexes used by QGIS have a `.qix` extension.

Use these steps to create the index:

1. Load an ESRI Shapefile format dataset (see [The Browser Panel](#))
2. Open the *Layer Properties* dialog by double-clicking on the layer name in the legend or by right-clicking and choosing *Properties...* from the context menu
3. In the *Source* tab, click the *Create Spatial Index* button

### Problem loading a `.prj` file

If you load an ESRI Shapefile format dataset with a `.prj` file and QGIS is not able to read the coordinate reference system from that file, you will need to define the proper projection manually in the *Layer Properties* ► *Source* tab of the layer by clicking the  **Select CRS** button. This is due to the fact that `.prj` files often do not provide the complete projection parameters as used in QGIS and listed in the *CRS* dialog.

For the same reason, if you create a new ESRI Shapefile format dataset with QGIS, two different projection files are created: a `.prj` file with limited projection parameters, compatible with ESRI software, and a `.qpj` file, providing all the parameters of the CRS. Whenever QGIS finds a `.qpj` file, it will be used instead of the `.prj`.

## Delimited Text Files

Delimited text files are very common and widely used because of their simplicity and readability – data can be viewed and edited in a plain text editor. A delimited text file is tabular data with columns separated by a defined character and rows separated by line breaks. The first row usually contains the column names. A common type of delimited text file is a CSV (Comma Separated Values), with columns separated by commas. Delimited text files can also contain positional information (see [Storing geometry information in delimited text files](#)).

QGIS allows you to load a delimited text file as a layer or an ordinary table (see [The Browser Panel](#) or [Importing a delimited text file](#)). First check that the file meets the following requirements:

1. The file must have a delimited header row of field names. This must be the first line of the data (ideally the first row in the text file).
2. If geometry should be enabled, the file must contain field(s) that define the geometry. These field(s) can have any name.

3. The X and Y coordinates fields (if geometry is defined by coordinates) must be specified as numbers. The coordinate system is not important.
4. If you have a CSV file with non-string columns, you can have an accompanying CSVT file (see section [Using CSVT file to control field formatting](#)).

The elevation point data file `elevp.csv` in the QGIS sample dataset (see section [Beispieldaten herunterladen](#)) is an example of a valid text file:

```
X;Y;ELEV
-300120;7689960;13
-654360;7562040;52
1640;7512840;3
[...]
```

Some things to note about the text file:

1. The example text file uses `;` (semicolon) as delimiter (any character can be used to delimit the fields).
2. The first row is the header row. It contains the fields X, Y and ELEV.
3. No quotes (") are used to delimit text fields
4. The X coordinates are contained in the X field
5. The Y coordinates are contained in the Y field

### Storing geometry information in delimited text files

Delimited text files can contain geometry information in two main forms:

- As coordinates in separate columns (eg. `Xcol,Ycol...`), for point geometry data;
- As well-known text (WKT) representation of geometry in a single column, for any geometry type.

Features with curved geometries (CircularString, CurvePolygon and CompoundCurve) are supported. Here are some examples of geometry types in a delimited text file with geometries coded as WKT:

```
Label;WKT_geom
LineString;LINESTRING(10.0 20.0, 11.0 21.0, 13.0 25.5)
CircularString;CIRCULARSTRING(268 415,227 505,227 406)
CurvePolygon;CURVEPOLYGON(CIRCULARSTRING(1 3, 3 5, 4 7, 7 3, 1 3))
CompoundCurve;COMPOUNDCURVE((5 3, 5 13), CIRCULARSTRING(5 13, 7 15,
9 13), (9 13, 9 3), CIRCULARSTRING(9 3, 7 1, 5 3))
```

Delimited text files also support Z and M coordinates in geometries:

```
LINESTRINGZ(10.0 20.0 30.0, 11.0 21.0 31.0, 11.0 22.0 30.0)
```

### Using CSVT file to control field formatting

When loading CSV files, the GDAL driver assumes all fields are strings (i.e. text) unless it is told otherwise. You can create a CSVT file to tell GDAL (and QGIS) the data type of the different columns:



Typ	Name	Example
Whole number	Integer	4
Boolean	Integer(Boolean)	true
Decimal number	Real	3.456
Datum	Date (YYYY-MM-DD)	2016-07-28
Time	Time (HH:MM:SS+nn)	18:33:12+00
Date & Time	DateTime (YYYY-MM-DD HH:MM:SS+nn)	2016-07-28 18:33:12+00
CoordX	CoordX	8.8249
CoordY	CoordY	47.2274
Point(X)	Point(X)	8.8249
Point(Y)	Point(Y)	47.2274
WKT	WKT	POINT(15 20)

The CSVT file is a **ONE line** plain text file with the data types in quotes and separated by commas, e.g.:

```
"Integer", "Real", "String"
```

You can even specify width and precision of each column, e.g.:

```
"Integer(6)", "Real(5.5)", "String(22)"
```

This file is saved in the same folder as the .csv file, with the same name, but .csvt as the extension.

*You can find more information at [GDAL CSV Driver](#).*

---

### **Tipp: Detect Field Types**

Instead of using a CSVT file to tell the data types, QGIS provides the possibility to automatically detect the field types and to change the assumed field types.

---

## **PostGIS Layers**

PostGIS layers are stored in a PostgreSQL database. The advantages of PostGIS are spatial indexing, filtering and querying capabilities. Using PostGIS, vector functions such as select and identify work more accurately than they do with GDAL layers in QGIS.

---

### **Tipp: PostGIS Layers**

Normally, a PostGIS layer is identified by an entry in the geometry\_columns table. QGIS can load layers that do not have an entry in the geometry\_columns table. This includes both tables and views. Refer to your PostgreSQL manual for information on creating views.

---

This section contains some details on how QGIS accesses PostgreSQL layers. Most of the time, QGIS should simply provide you with a list of database tables that can be loaded, and it will load them on request. However, if you have trouble loading a PostgreSQL table into QGIS, the information below may help you understand QGIS messages and give you directions for modifying the PostgreSQL table or view definition to allow QGIS to load it.

---

**Bemerkung:** A PostgreSQL database can also store QGIS projects.

---

### Primary key

QGIS requires that PostgreSQL layers contain a column that can be used as a unique key for the layer. For tables, this usually means that the table needs a primary key, or a column with a unique constraint on it. In QGIS, this column needs to be of type int4 (an integer of size 4 bytes). Alternatively, the ctid column can be used as primary key. If a table lacks these items, the oid column will be used instead. Performance will be improved if the column is indexed (note that primary keys are automatically indexed in PostgreSQL).

QGIS offers a checkbox **Select at id** that is activated by default. This option gets the ids without the attributes, which is faster in most cases.

### Ansicht

If the PostgreSQL layer is a view, the same requirement exists, but views do not always have primary keys or columns with unique constraints on them. You have to define a primary key field (has to be integer) in the QGIS dialog before you can load the view. If a suitable column does not exist in the view, QGIS will not load the layer. If this occurs, the solution is to alter the view so that it does include a suitable column (a type of integer and either a primary key or with a unique constraint, preferably indexed).

As for table, a checkbox **Select at id** is activated by default (see above for the meaning of the checkbox). It can make sense to disable this option when you use expensive views.

---

### Bemerkung: PostgreSQL foreign table

PostgreSQL foreign tables are not explicitly supported by the PostgreSQL provider and will be handled like a view.


---

### QGIS layer\_style table and database backup

If you want to make a backup of your PostGIS database using the `pg_dump` and `pg_restore` commands, and the default layer styles as saved by QGIS fail to restore afterwards, you need to set the XML option to DOCUMENT before the restore command:

1. Make a PLAIN backup of the `layer_style` table
2. Open the file within a text editor
3. Change the line `SET xmloption = content;` into `SET XML OPTION DOCUMENT;`
4. Save the file
5. Use `psql` to restore the table in the new database

### Filter database side

QGIS allows to filter features already on server side. Check *Settings ► Options ► Data Sources ►  Execute expressions on server-side if possible* to do so. Only supported expressions will be sent to the database. Expressions using unsupported operators or functions will gracefully fallback to local evaluation.


## Support of PostgreSQL data types

Data types supported by the PostgreSQL provider include: integer, float, boolean, binary object, varchar, geometry, timestamp, array, hstore and json.

## Importing Data into PostgreSQL

Data can be imported into PostgreSQL/PostGIS using several tools, including the DB Manager plugin and the command line tools `shp2pgsql` and `ogr2ogr`.

### DB Manager

QGIS comes with a core plugin named  **DB Manager**. It can be used to load data, and it includes support for schemas. See section [DB Manager Plugin](#) for more information.

### shp2pgsql

PostGIS includes a utility called **shp2pgsql**, that can be used to import Shapefile format datasets into a PostGIS-enabled database. For example, to import a Shapefile format dataset named `lakes.shp` into a PostgreSQL database named `gis_data`, use the following command:

```
shp2pgsql -s 2964 lakes.shp lakes_new | psql gis_data
```

This creates a new layer named `lakes_new` in the `gis_data` database. The new layer will have a spatial reference identifier (SRID) of 2964. See section [Arbeiten mit Projektionen](#) for more information about spatial reference systems and projections.

### Tipp: Exporting datasets from PostGIS

There is also a tool for exporting PostGIS datasets to Shapefile format: **pgsql2shp**. It is shipped within your PostGIS distribution.


### ogr2ogr

In addition to **shp2pgsql** and **DB Manager**, there is another tool for feeding geographical data in PostGIS: **ogr2ogr**. It is part of your GDAL installation.



To import a Shapefile format dataset into PostGIS, do the following:

```
ogr2ogr -f "PostgreSQL" PG:"dbname=postgis host=myhost.de user=postgres  
password=topsecret" alaska.shp
```

This will import the Shapefile format dataset `alaska.shp` into the PostGIS database `postgis` using the user `postgres` with the password `topsecret` on the host server `myhost.de`.

Note that GDAL must be built with PostgreSQL to support PostGIS. You can verify this by typing (in ):

```
ogrinfo --formats | grep -i post
```

If you prefer to use the PostgreSQL's **COPY** command instead of the default **INSERT INTO** method, you can export the following environment variable (at least available on  and ):

```
export PG_USE_COPY=YES
```

**ogr2ogr** does not create spatial indexes like **shp2pgsql** does. You need to create them manually, using the normal SQL command **CREATE INDEX** afterwards, as an extra step (as described in the next section *Improving Performance*).

### Improving Performance

Retrieving features from a PostgreSQL database can be time-consuming, especially over a network. You can improve the drawing performance of PostgreSQL layers by ensuring that a PostGIS spatial index exists on each layer in the database. PostGIS supports creation of a GiST (Generalized Search Tree) index to speed up spatial searching (GiST index information is taken from the PostGIS documentation available at <https://postgis.net>).

---

**Tip:** You can use the DBManager to create an index for your layer. You should first select the layer and click on *Table ► Edit table*, go to *Indexes* tab and click on *Add Spatial Index*.

---

The syntax for creating a GiST index is:

```
CREATE INDEX [indexname] ON [tablename]
  USING GIST ( [geometryfield] GIST_GEOMETRY_OPS );
```

Note that for large tables, creating the index can take a long time. Once the index is created, you should perform a **VACUUM ANALYZE**. See the PostGIS documentation (POSTGIS-PROJECT in *Literatur und Internetreferenzen*) for more information.

The following example creates a GiST index:

```
gsherman@madison:~/current$ psql gis_data
Welcome to psql 8.3.0, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
        \h for help with SQL commands
        \? for help with psql commands
        \g or terminate with semicolon to execute query
        \q to quit

gis_data=# CREATE INDEX idx_alaska_lakes ON alaska_lakes
gis_data=# USING GIST (the_geom GIST_GEOMETRY_OPS);
CREATE INDEX
gis_data=# VACUUM ANALYZE alaska_lakes;
VACUUM
gis_data=# \q
gsherman@madison:~/current$
```

### Spatialite Layers

If you want to save a vector layer using the Spatialite format, you can do this by following instructions at *Creating new layers from an existing layer*. You select Spatialite as *Format* and enter both *File name* and *Layer name*.

Also, you can select SQLite as format and then add **SPATIALITE=YES** in the *Custom Options ► Data source* field. This tells GDAL to create a Spatialite database. See also <https://gdal.org/drivers/vector/sqlite.html>.

QGIS also supports editable views in Spatialite. For Spatialite data management, you can also use the core plugin *DB Manager*.

If you want to create a new Spatialite layer, please refer to section *Creating a new Spatialite layer*.

## GeoJSON specific parameters

When *exporting layers* to GeoJSON, there are some specific *Layer Options* available. These options come from GDAL which is responsible for the writing of the file:

- *COORDINATE\_PRECISION* the maximum number of digits after the decimal separator to write in coordinates. Defaults to 15 (note: for Lat Lon coordinates 6 is considered enough). Truncation will occur to remove trailing zeros.
- *RFC7946* by default GeoJSON 2008 will be used. If set to YES, the updated RFC 7946 standard will be used. Default is NO (thus GeoJSON 2008). See <https://gdal.org/drivers/vector/geojson.html#rfc-7946-write-support> for the main differences, in short: only EPSG:4326 is allowed, other crs's will be transformed, polygons will be written such as to follow the right-hand rule for orientation, values of a „bbox“ array are [west, south, east, north], not [minx, miny, maxx, maxy]. Some extension member names are forbidden in FeatureCollection, Feature and Geometry objects, the default coordinate precision is 7 decimal digits
- *WRITE\_BBOX* set to YES to include the bounding box of the geometries at the feature and feature collection level

Besides GeoJSON there is also an option to export to „GeoJSON - Newline Delimited“ (see <https://gdal.org/drivers/vector/geojsonseq.html>). Instead of a FeatureCollection with Features, you can stream one type (probably only Features) sequentially separated with newlines.

GeoJSON - Newline Delimited has some specific Layer options available too:

- *COORDINATE\_PRECISION* see above (same as for GeoJSON)
- *RS* whether to start records with the RS=0x1E character. The difference is how the features are separated: only by a newline (LF) character (Newline Delimited JSON, geojsonl) or by also prepending a record-separator (RS) character (giving GeoJSON Text Sequences, geojsons). Default to NO. Files are given the .json extension if extension is not provided.

## SAP HANA Spatial Layers

This section contains some details on how QGIS accesses SAP HANA layers. Most of the time, QGIS should simply provide you with a list of database tables and views that can be loaded, and it will load them on request. However, if you have trouble loading an SAP HANA table or view into QGIS, the information below may help you understand the root cause and assist in resolving the issue.

### Feature Identification

If you'd like to use all of QGIS' feature editing capabilities, QGIS must be able to unambiguously identify each feature in a layer. Internally, QGIS uses a 64-bit signed integer to identify features, whereas the negative range is reserved for special purposes.

Therefore, the SAP HANA provider requires a unique key that can be mapped to a positive 64-bit integer to fully support QGIS' feature editing capabilities. If it is not possible to create such a mapping, you might still view the features, but editing might not work.

### Adding tables

When adding a table as a layer, the SAP HANA provider uses the table's primary key to map it to a unique feature id. Therefore, to have full feature editing support, you need to have a primary key to your table definition.

The SAP HANA provider supports multi-column primary keys, but if you'd like to get the best performance, your primary key should be a single column of type `INTEGER`.

### Adding views

When adding a view as a layer, the SAP HANA provider cannot automatically identify columns that unambiguously identify a feature. Furthermore, some views are read-only and cannot be edited.

To have full feature editing support, the view must be updatable (check column `IS_READ_ONLY` in system view `SYS.VIEWS` for the view in question) and you must manually provide QGIS with one or more columns that identify a feature. The columns can be given by using *Layer ► Add Layer ► Add SAP HANA Spatial Layer* and then selecting the columns in the *Feature id* column. For best performance, the *Feature id* value should be a single `INTEGER` column.

### 15.3.3 Layers crossing 180° longitude

Many GIS packages don't wrap layers with a geographic reference system (lat/lon) crossing the 180 degrees longitude line. As result, if we open such a layer in QGIS, we could see two widely separated locations, that should appear near each other. In [Abb. 15.31](#), the tiny point on the far left of the map canvas (Chatham Islands) should be within the grid, to the right of the New Zealand main islands.



Abb. 15.31: Map in lat/lon crossing the 180° longitude line

## Solving in PostGIS

A work-around is to transform the longitude values using PostGIS and the `ST_ShiftLongitude` function. This function reads every point/vertex in every component of every feature in a geometry, and shifts its longitude coordinate from  $-180..0^\circ$  to  $180..360^\circ$  and vice versa if between these ranges. This function is symmetrical so the result is a  $0..360^\circ$  representation of a  $-180..180^\circ$  data and a  $-180..180^\circ$  representation of a  $0..360^\circ$  data.

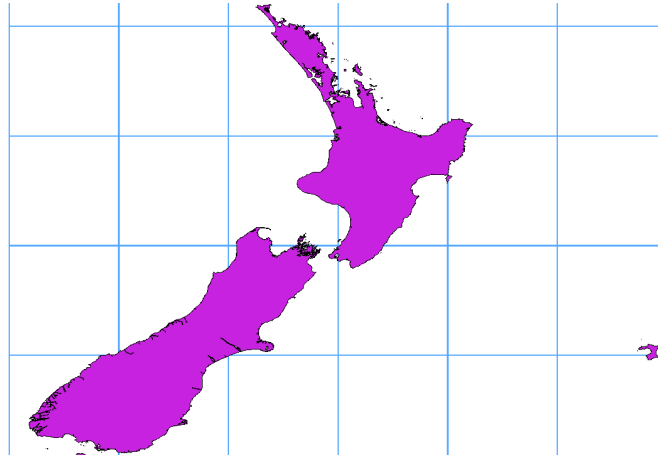


Abb. 15.32: Crossing  $180^\circ$  longitude applying the `ST_ShiftLongitude` function

1. Import data into PostGIS (*Importing Data into PostgreSQL*) using, for example, the DB Manager plugin.
2. Use the PostGIS command line interface to issue the following command:

```
-- In this example, "TABLE" is the actual name of your PostGIS table  
update TABLE set geom=ST_ShiftLongitude(geom);
```

3. If everything went well, you should receive a confirmation about the number of features that were updated. Then you'll be able to load the map and see the difference (*Figure\_vector\_crossing\_map*).



























### 16.1 Vektorlayereigenschaften

Der *Layereigenschaften* Dialog für Vektorlayer enthält generelle Einstellungen, um das Aussehen der Layerobjekte in der Karte (Symbologie, Beschriftung, Diagramme) und die Interaktion mit der Maus (Aktionen, Kartentipps, ...) zu verwalten. Er bietet auch Informationen über den Layer.

Sie können den *Layereigenschaften* Dialog öffnen, indem Sie:

- im Layer `Bedienfeld einen Doppelklick auf den Layer machen oder einen Rechtsklick und im Kontextmenü `:guilabel:'Eigenschaften ...` auswählen
- in der Menüleiste *Layer* ► *Layereigenschaften ...* auswählen wenn der Layer aktiv ist

Der *Layereigenschaften* Dialog für Vektorlayer bietet die folgenden Bereiche:

 <i>Information</i>	 <i>Quelle</i>	 <i>Symbolisierung<sup>[1]</sup></i>
 <i>Beschriftungen<sup>[1]</sup></i>	 <i>Masken<sup>[1]</sup></i>	 <i>3D-Ansicht<sup>[1]</sup></i>
 <i>Diagramme</i>	 <i>Felder</i>	 <i>Attributformular</i>
 <i>Joins</i>	 <i>Hilfsspeicher</i>	 <i>Aktionen</i>
 <i>Anzeige</i>	 <i>Darstellung</i>	 <i>Zeitlich</i>
 <i>Variablen</i>	 <i>Elevation</i>	 <i>Metadaten</i>
 <i>Abhängigkeiten</i>	 <i>Legende</i>	 <i>QGIS Server</i>
 <i>Digitalisieren</i>	<i>Externe Plugins<sup>[2]</sup> Reiter</i>	

<sup>[1]</sup> Auch im Bedienfeld *Layergestaltung* verfügbar

Durch das Installieren von <sup>[2]</sup> *Plugins* können hier im Dialog *Layereigenschaften* weitere Tabs hinzugefügt werden. Diese werden nicht in dieser Dokumentation behandelt, bitte schauen Sie in der entsprechenden Dokumentation nach.

---

### **Tipp:** Layereigenschaften speichern und teilen

Mit dem *Stil* Menü unten im Dialogfenster ist es möglich die Layereigenschaften (oder Teile davon) zu speichern und zu laden. Für mehr Informationen s. *Verwalten von benutzerdefinierten Stilen*.


---

---

**Bemerkung:** Da die Eigenschaften von eingebetteten Layern (see *Embedding layers from external projects*) aus der ursprünglichen QGIS-Projektdatei stammen und um Änderungen zu vermeiden, die mit diesen Einstellungen Konflikte verursachen könnten, ist der Ebeneneigenschaften-Dialog für eingebetteten Ebenen nicht verfügbar.

---

### 16.1.1 Eigenschaften Information

The  *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata on the current layer. Provided information are:

- general such as name in the project, source path, list of auxiliary files, last save time and size, the used provider
- based on the provider of the layer: format of storage, geometry type, data source encoding, extent, feature count...
- das Koordinatenbezugssystem: Name, Einheiten, Methode, Genauigkeit, Referenz (d.h., ob es statisch oder dynamisch ist)
- picked from the *filled metadata*: access, extents, links, contacts, history...
- and related to its geometry (spatial extent, CRS...) or its attributes (number of fields, characteristics of each...).

### 16.1.2 Eigenschaften Quelle



Auf dieser Registerkarte können Sie allgemeine Einstellungen für die Vektorebene festlegen.

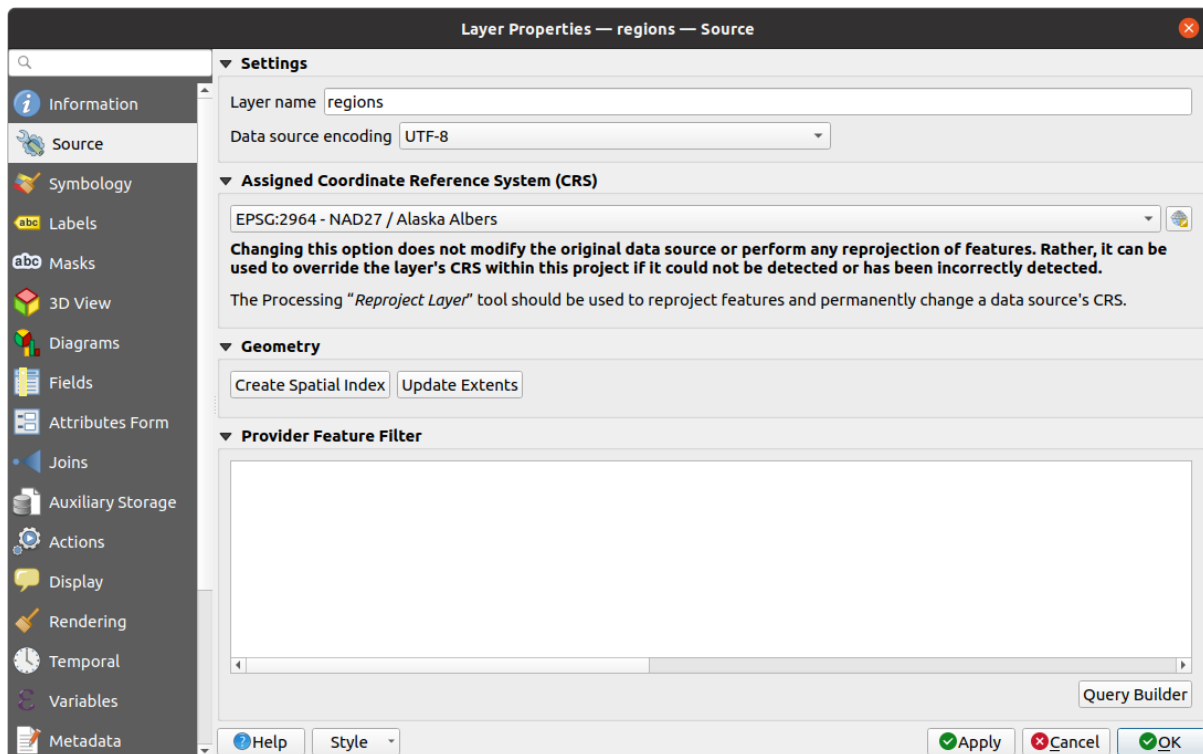



Abb. 16.1: Registerkarte Quelle im Menu Vektorlayereigenschaften

## Einstellungen

- Lege einen *layernamen* fest, der sich vom Dateinamen der Ebene unterscheidet und zur Identifizierung der Ebene im Projekt verwendet wird (im *Layerbedienfeld*, mit Ausdrücken, in der Drucklayoutlegende, ...)
- Wählen Sie je nach Datenformat die *Datenquellenkodierung*, wenn sie von QGIS nicht korrekt erkannt wird.

## Koordinatenbezugssystem und Geometrie

- Displays the layer's *Assigned Coordinate Reference System (CRS)*. You can change the layer's CRS, selecting a recently used one in the drop-down list or clicking on  **Select CRS** button (see *Coordinate Reference System Selector*). Use this process only if the CRS applied to the layer is a wrong one or if none was applied. If you wish to reproject your data into another CRS, rather use layer reprojection algorithms from Processing or *Save it into another layer*.
- *Create spatial index* (only for OGR-supported formats).
- *Update extents* information for a layer.

## Abfrageeditor

The Query Builder dialog is accessible through the *Query Builder* button at the bottom of the *Source* tab in the Layer Properties dialog, under the *Provider feature filter* group.

The Query Builder provides an interface that allows you to define a subset of the features in the layer using a SQL-like WHERE clause and to display the result in the main window. As long as the query is active, only the features corresponding to its result are available in the project.

You can use one or more layer attributes to define the filter in the *Query Builder*. The use of more than one attribute is shown in [Abb. 16.2](#). In the example, the filter combines the attributes

- toa (DateTime field: cast("toa" as character) > '2017-05-17' and cast("toa" as character) < '2019-12-24T18:00:00'),
- name (String field: "name" > 'S') and
- FID (Integer field: FID > 10)

using the AND, OR and NOT operators and parenthesis. This syntax (including the DateTime format for the toa field) works for GeoPackage datasets.

The filter is made at the data provider (OGR, PostgreSQL, MS SQL Server...) level. So the syntax depends on the data provider (DateTime is for instance not supported for the ESRI Shapefile format). The complete expression:

```
cast("toa" as character) > '2017-05-17' AND
cast("toa" as character) < '2019-12-24T18:00:00' AND
NOT ("name" > 'S' OR FID > 10)
```

You can also open the *Query Builder* dialog using the *Filter...* option from the *Layer* menu or the layer contextual menu. The *Fields*, *Values* and *Operators* sections in the dialog help you to construct the SQL-like query exposed in the *Provider specific filter expression* box.

The screenshot shows the 'Set provider filter on testdata' dialog. It is divided into several sections:

- Fields:** A list box containing 'fid', 'name', and 'toa'.
- Values:** A search box with 'Search...', a list box for values, and buttons for 'Sample' and 'All'. There is also a checkbox for 'Use unfiltered layer'.
- Operators:** A grid of buttons for logical and comparison operators: '=', '<', '>', 'LIKE', '%', 'IN', 'NOT IN', '<=', '>=', '!=', 'ILIKE', 'AND', 'OR', and 'NOT'.
- Provider specific filter expression:** A text area containing the SQL query: `cast("toa" as character) > '2017-05-17' AND cast("toa" as character) < '2019-12-24T18:00:00' AND NOT ("name" > 'S' OR FID > 10)`.
- Buttons:** At the bottom, there are buttons for 'Help', 'Test', 'Clear', 'Cancel', and 'OK'.

Abb. 16.2: Abfrageeditor

The **Fields** list contains all the fields of the layer. To add an attribute column to the SQL WHERE clause field, double-click its name or just type it into the SQL box.

The **Values** frame lists the values of the currently selected field. To list all unique values of a field, click the *All* button. To instead list the first 25 unique values of the column, click the *Sample* button. To add a value to the SQL

WHERE clause field, double click its name in the Values list. You can use the search box at the top of the Values frame to easily browse and find attribute values in the list.


The **Operators** section contains all usable operators. To add an operator to the SQL WHERE clause field, click the appropriate button. Relational operators (=, >, ...), string comparison operator (LIKE), and logical operators (AND, OR, ...) are available.

The *Test* button helps you check your query and displays a message box with the number of features satisfying the current query. Use the *Clear* button to wipe the SQL query and revert the layer to its original state (ie, fully load all the features). It is possible to *Save...* the query as a .QQF file, or *Load...* the query from a file into the dialog.

When a filter is applied, QGIS treats the resulting subset acts as if it were the entire layer. For example if you applied the filter above for 'Borough' ("TYPE\_2" = 'Borough'), you can not display, query, save or edit *Anchorage*, because that is a 'Municipality' and therefore not part of the subset.


---

**Tipp: Filtered layers are indicated in the Layers Panel**

In the *Layers* panel, filtered layer is listed with a  <sup>Filter</sup> icon next to it indicating the query used when the mouse hovers over the button. Double-click the icon opens the *Query Builder* dialog for edit. This can also be achieved through the *Layer ► Filter...* menu.

---

### 16.1.3 Eigenschaften Symbolisierung

The  *Symbolology* tab provides you with a comprehensive tool for rendering and symbolizing your vector data. You can use tools that are common to all vector data, as well as special symbolizing tools that were designed for the different kinds of vector data. However all types share the following dialog structure: in the upper part, you have a widget that helps you prepare the classification and the symbol to use for features and at the bottom the *Layerdarstellung* widget.

---

**Tipp: Wechseln Sie schnell zwischen verschiedenen Layerdarstellungen**

Using the *Styles ► Add* menu at the bottom of the *Layer Properties* dialog, you can save as many styles as needed. A style is the combination of all properties of a layer (such as symbology, labeling, diagram, fields form, actions...) as you want. Then, simply switch between styles from the context menu of the layer in *Layers Panel* to automatically get different representations of your data.

---



---

**Tipp: Symbologie exportierten**

Sie haben die Option die Vektorsymbologie von QGIS nach Google \*.kml, \*.dxf und MapInfo \*.tab Dateien zu exportieren. Öffnen Sie einfach das Rechte-Maustasten-Menü des Layers und klicken Sie auf *Speichern als ...* `um den Namen der Ausgabedatei und ihr Format festzulegen. Verwenden Sie im Dialog das *:guilabel:'Darstellungsexport* Menü um die Symbologie entweder als *Objektdarstellung ►* oder *Symbollayerdarstellung ►* zu speichern. Wenn Sie Symbollayer verwendet haben wird empfohlen die zweite Einstellung zu benutzen.

---

### Objekt Darstellung


The renderer is responsible for drawing a feature together with the correct symbol. Regardless layer geometry type, there are four common types of renderers: single symbol, categorized, graduated and rule-based. For point layers, there are point displacement, point cluster and heatmap renderers available while polygon layers can also be rendered with the merged features, inverted polygons and 2.5 D renderers.

Es gibt keinen kontinuierliche Farbe Renderer da es in der Tat einfach ein spezieller Fall des Abgestuft Renderers ist. Die Kategorisiert und Abgestuft Renderer können erstellt werden indem ein Symbol und ein Farbverlauf festgelegt werden - Sie werden die Farben für Symbole angemessen einsetzen. Für jeden Datentyp (Punkte, Linien und

Polygone) sind Symbol Layertypen erhältlich. Abhängig vom ausgesuchten Renderer gibt es verschiedene zusätzliche Bereiche.

**Bemerkung:** Wenn Sie den Darstellungstyp beim Einstellen des Stils eines Vektorlayers ändern werden die Einstellungen für das Symbol beibehalten. Beachten Sie dass dieses Vorgehen nur für eine Änderung funktioniert. Wenn Sie den Darstellungstyp wiederholt ändern gehen die Einstellungen für das Symbol verloren.

### Einzel symbol Darstellung

The  *Single Symbol* renderer is used to render all features of the layer using a single user-defined symbol. See [The Symbol Selector](#) for further information about symbol representation.

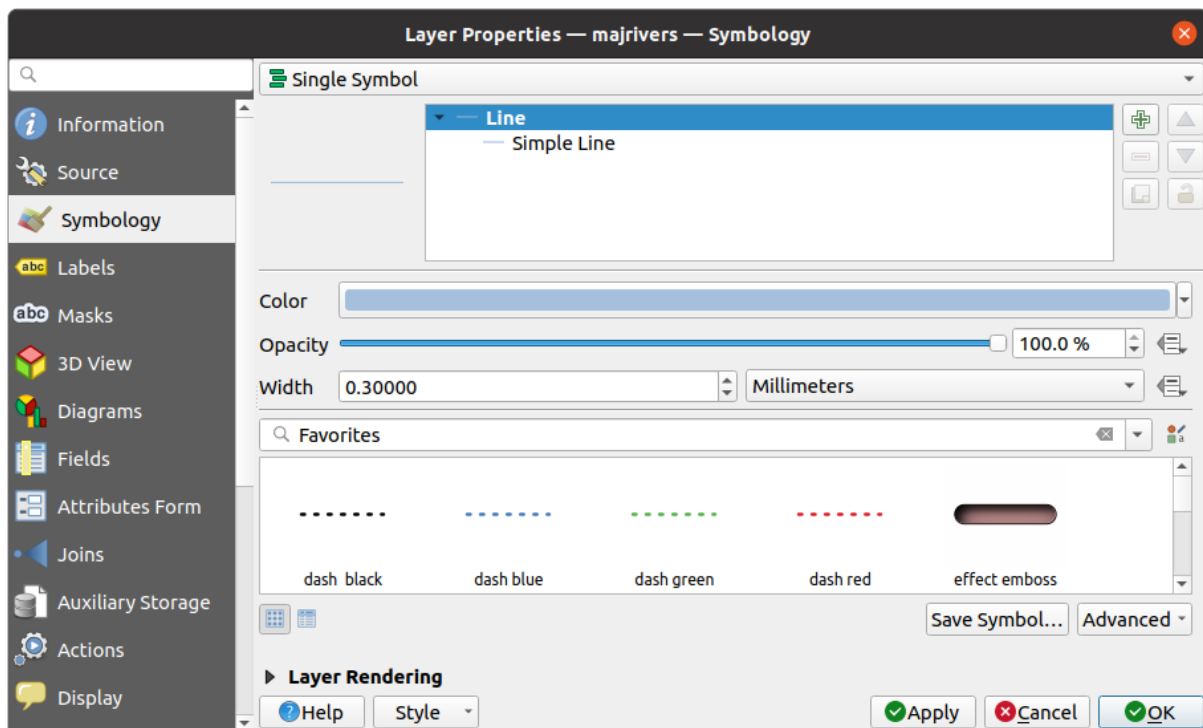



Abb. 16.3: Linieneigenschaften Einzelsymbol


### No Symbols Renderer

The  *No Symbols* renderer is a special use case of the Single Symbol renderer as it applies the same rendering to all features. Using this renderer, no symbol will be drawn for features, but labeling, diagrams and other non-symbol parts will still be shown.

Selections can still be made on the layer in the canvas and selected features will be rendered with a default symbol. Features being edited will also be shown.

This is intended as a handy shortcut for layers which you only want to show labels or diagrams for, and avoids the need to render symbols with totally transparent fill/border to achieve this.

## Kategorisierte Darstellung

The  *Categorized* renderer is used to render the features of a layer, using a user-defined symbol whose aspect reflects the discrete values of a field or an expression.

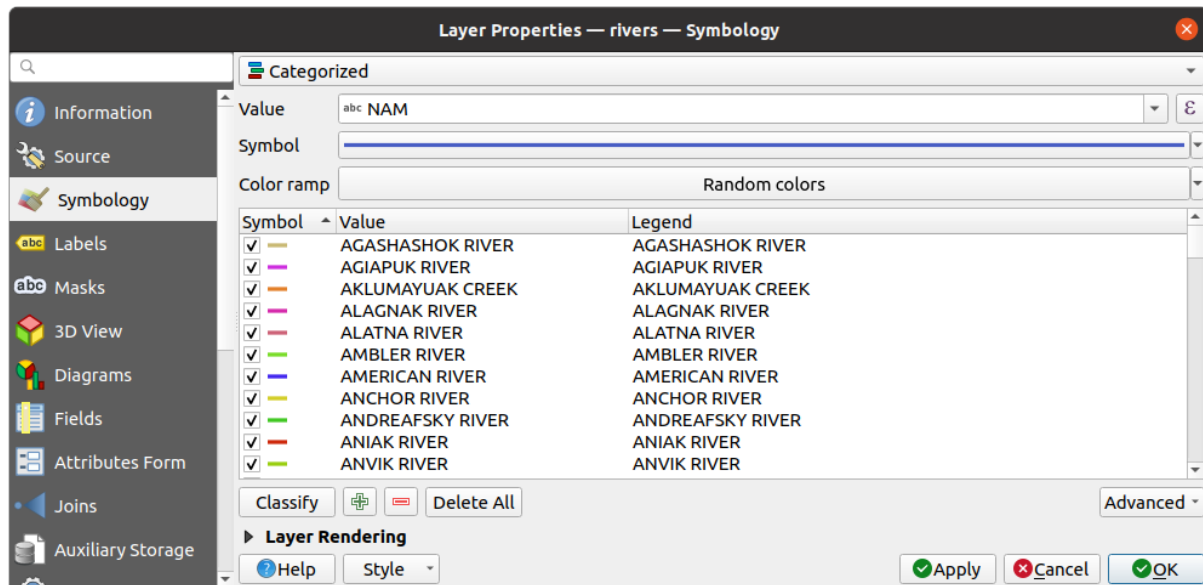



Abb. 16.4: Kategorisierte Symbolisierungsoptionen

To use categorized symbology for a layer:

1. Select the *Value* of classification: it can be an existing field or an *expression* you can type in the box or build using the associated  button. Using expressions for categorizing avoids the need to create a field for symbology purposes only (eg, if your classification criteria are derived from one or more attributes).

The expression used to classify features can be of any type; eg, it can:

- be a comparison. In this case, QGIS returns values 1 (**True**) and 0 (**False**). Some examples:

```
myfield >= 100
$id = @atlas_featureid
myfield % 2 = 0
within( $geometry, @atlas_geometry )
```

- combine different fields:

```
concat( field_1, ' ', field_2 )
```

- be a calculation on fields:

```
myfield % 2
year( myfield )
field_1 + field_2
substr( field_1, -3 )
```

- be used to transform linear values to discrete classes, e.g.:


```
CASE WHEN x > 1000 THEN 'Big' ELSE 'Small' END
```

- combine several discrete values into a single category, e.g.:

```
CASE
WHEN building IN ('residence', 'mobile home') THEN 'residential'
WHEN building IN ('commercial', 'industrial') THEN 'Commercial and
↪Industrial'
END
```

**Tip:** While you can use any kind of expression to categorize features, for some complex expressions it might be simpler to use *rule-based rendering*.



2. Configure the *Symbol*, which will be used as base symbol for all the classes;
3. Indicate the *Color ramp*, ie the range of colors from which the color applied to each symbol is selected.

Besides the common options of the *color ramp widget*, you can apply a  *Random Color Ramp* to the categories. You can click the *Shuffle Random Colors* entry to regenerate a new set of random colors if you are not satisfied.

4. Then click on the *Classify* button to create classes from the distinct values of the provided field or expression.
5. *Apply* the changes if the *live update* is not in use and each feature on the map canvas will be rendered with the symbol of its class.

By default, QGIS appends an *all other values* class to the list. While empty at the beginning, this class is used as a default class for any feature not falling into the other classes (eg, when you create features with new values for the classification field / expression).

Further tweaks can be done to the default classification:

- You can  Add new categories,  Remove selected categories or *Delete All* of them.
- A class can be disabled by unchecking the checkbox to the left of the class name; the corresponding features are hidden on the map.
- Drag-and-drop the rows to reorder the classes
- To change the symbol, the value or the legend of a class, double click the item.

Right-clicking over selected item(s) shows a contextual menu to:

- *Copy Symbol* and *Paste Symbol*, a convenient way to apply the item's representation to others
- *Change Color...* of the selected symbol(s)
- *Change Opacity...* of the selected symbol(s)
- *Change Output Unit...* of the selected symbol(s)
- *Change Width...* of the selected line symbol(s)
- *Change Size...* of the selected point symbol(s)
- *Change Angle...* of the selected point symbol(s)
- *Merge Categories*: Groups multiple selected categories into a single one. This allows simpler styling of layers with a large number of categories, where it may be possible to group numerous distinct categories into a smaller and more manageable set of categories which apply to multiple values.

**Tip:** Since the symbol kept for the merged categories is the one of the topmost selected category in the list, you may want to move the category whose symbol you wish to reuse to the top before merging.

- *Unmerge Categories* that were previously merged


The created classes also appear in a tree hierarchy in the *Layers* panel. Double-click an entry in the map legend to edit the assigned symbol. Right-click and you will get some *more options*.



The *Advanced* menu gives access to options to speed classification or fine-tune the symbols rendering:


- *Match to saved symbols*: Using the *symbols library*, assigns to each category a symbol whose name represents the classification value of the category
- *Match to symbols from file...*: Provided a file with symbols, assigns to each category a symbol whose name represents the classification value of the category
- *Symbol levels...* to define the order of symbols rendering.

## Abgestufte Darstellung

The  *Graduated* renderer is used to render all the features from a layer, using an user-defined symbol whose color or size reflects the assignment of a selected feature's attribute to a class.

Like the Categorized Renderer, the Graduated Renderer allows you to define rotation and size scale from specified columns.

Genauso können Sie -analog zum Kategorisierten Renderer - im Menü auswählen:

- The *Value* of classification: it can be an existing field or an *expression* you can type in the box or build using the associated  button. Using expressions for graduating avoids the need to create a field for symbology purposes only (eg, if your classification criteria are derived from one or more attributes).
- Das Symbol (nutzen des Symbolauswahl Dialogs)
- Die Legendenformatierung und die Genauigkeit
- Die Methode, um das Symbol zu ändern: Farbe oder Größe
- Die Farben (indem man die Farbverlaufsliste verwendet), wenn die Farbmethode ausgewählt ist
- The size (using the size domain and its unit)

Then you can use the Histogram tab which shows an interactive histogram of the values from the assigned field or expression. Class breaks can be moved or added using the histogram widget.

---

**Bemerkung:** Sie können die statistische Zusammenfassung nutzen, um mehr Informationen über Ihren Vektorlayer zu erhalten. Siehe *Statistical Summary Panel*.

---

Zurück zu dem Klassen Reiter, Sie können die Anzahl der Klassen und auch den Modus für das Klassifizieren von Objekten innerhalb der Klassen (indem man die Modus-Liste verwendet) einstellen. Die möglichen Modi sind:

- Equal Count (Quantile): each class will have the same number of elements (the idea of a boxplot).
- Equal Interval: each class will have the same size (e.g. with the values from 1 to 16 and four classes, each class will have a size of four).
- Logarithmic scale: suitable for data with a wide range of values. Narrow classes for low values and wide classes for large values (e.g. for decimal numbers with range [0..100] and two classes, the first class will be from 0 to 10 and the second class from 10 to 100).
- Natural Breaks (Jenks): the variance within each class is minimized while the variance between classes is maximized.
- Pretty Breaks: computes a sequence of about  $n+1$  equally spaced nice values which cover the range of the values in  $x$ . The values are chosen so that they are 1, 2 or 5 times a power of 10. (based on pretty from the R statistical environment <https://www.rdocumentation.org/packages/base/topics/pretty>).
- Standard Deviation: classes are built depending on the standard deviation of the values.

The listbox in the center part of the *Symbology* tab lists the classes together with their ranges, labels and symbols that will be rendered.

Klicken Sie auf den **Klassifizieren** Knopf um Klassen anhand des ausgewählten Modus zu erstellen. Jede Klasse kann anhand des Deaktivierens des Kontrollkästchens links neben dem Klassennamen ausgeschaltet werden.

Sie können das Symbol, den Wert oder die Beschriftung verändern, klicken Sie einfach auf das Element, das Sie ändern wollen.

Right-clicking over selected item(s) shows a contextual menu to:

- *Copy Symbol* and *Paste Symbol*, a convenient way to apply the item's representation to others
- *Change Color...* of the selected symbol(s)
- *Change Opacity...* of the selected symbol(s)
- *Change Output Unit...* of the selected symbol(s)
- *Change Width...* of the selected line symbol(s)
- *Change Size...* of the selected point symbol(s)
- *Change Angle...* of the selected point symbol(s)

The example in Abb. 16.5 shows the graduated rendering dialog for the `major_rivers` layer of the QGIS sample dataset.

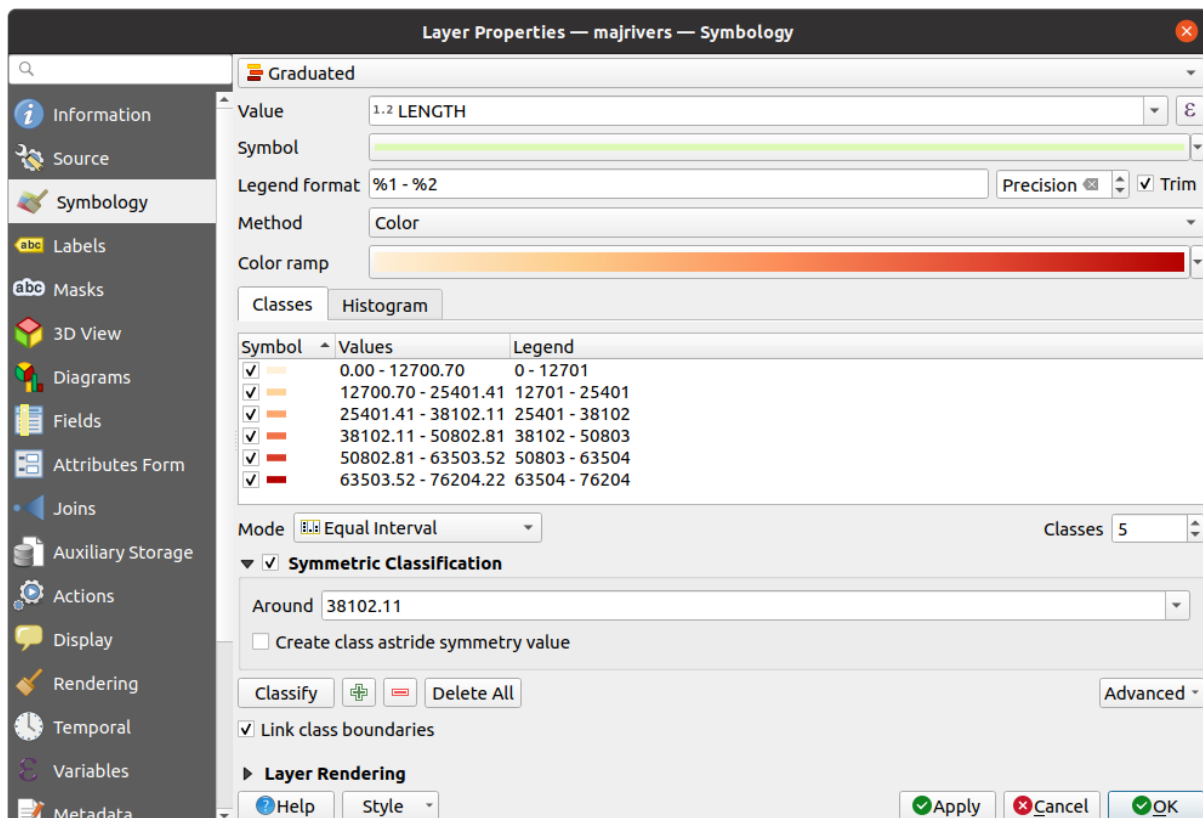


Abb. 16.5: Abgestufte Symbolisierungsoptionen


The created classes also appear in a tree hierarchy in the *Layers* panel. Double-click an entry in the map legend to edit the assigned symbol. Right-click and you will get some *more options*.

## Proportionale Symbole und mehrdimensionale Analysen

Proportional Symbol and Multivariate Analysis are not rendering types available from the Symbology rendering drop-down list. However with the *data-defined override* options applied over any of the previous rendering options, QGIS allows you to display your point and line data with such representation.

### Proportionale Symbole erstellen

To apply a proportional rendering:

1. First apply to the layer the *single symbol renderer*.
2. Then set the symbol to apply to the features.
3. Select the item at the upper level of the symbol tree, and use the  Data-defined override *button* next to the *Size* (for point layer) or *Width* (for line layer) option.
4. Select a field or enter an expression, and for each feature, QGIS will apply the output value to the property and proportionally resize the symbol in the map canvas.


If need be, use the *Size assistant...* option of the  menu to apply some transformation (exponential, flannery...) to the symbol size rescaling (see *Using the data-defined assistant interface* for more details).

You can choose to display the proportional symbols in the *Layers panel* and the *print layout legend item*: unfold the *Advanced* drop-down list at the bottom of the main dialog of the *Symbology* tab and select **Data-defined size legend...** to configure the legend items (see *Data-defined size legend* for details).

### Mehrdimensionale Analyse erzeugen

Eine mehrdimensionale Analyse hilft Ihnen, die Beziehungen zwischen zwei oder mehr Variablen auszuwerten z. B. kann eine als Farbverlauf und die andere als Größe dargestellt werden.

The simplest way to create multivariate analysis in QGIS is to:

1. First apply a categorized or graduated rendering on a layer, using the same type of symbol for all the classes.
2. Then, apply a proportional symbology on the classes:
  1. Click on the *Change* button above the classification frame: you get the *The Symbol Selector* dialog.
  2. Rescale the size or width of the symbol layer using the  *data defined override* widget as seen above.

Like the proportional symbol, the scaled symbology can be added to the layer tree, on top of the categorized or graduated classes symbols using the *data defined size legend* feature. And both representation are also available in the print layout legend item.

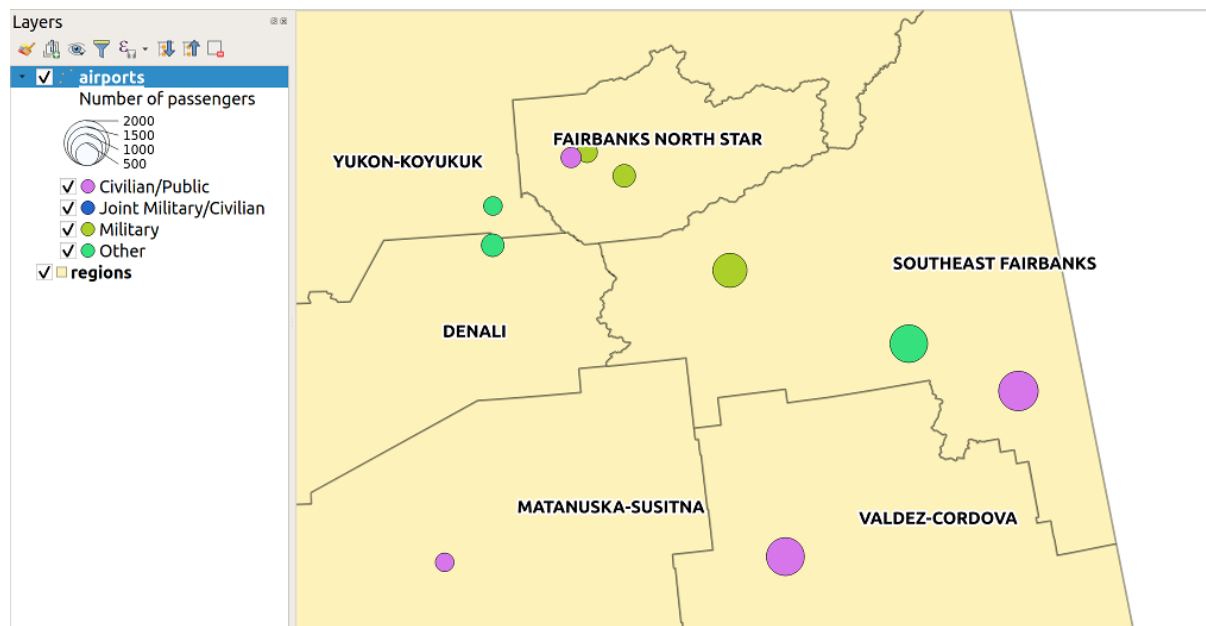








Abb. 16.6: Multivariate example with scaled size legend

## Rule-based Renderer

Rules are QGIS *expressions* used to discriminate features according to their attributes or properties in order to apply specific rendering settings to them. Rules can be nested, and features belong to a class if they belong to all the upper nesting level(s).

The  *Rule-based* renderer is thus designed to render all the features from a layer, using symbols whose aspect reflects the assignment of a selected feature to a fine-grained class.

To create a rule:

1. Activate an existing row by double-clicking it (by default, QGIS adds a symbol without a rule when the rendering mode is enabled) or click the  Edit rule or  Add rule button.
2. In the *Edit Rule* dialog that opens, you can define a label to help you identify each rule. This is the label that will be displayed in the *Layers Panel* and also in the print composer legend.
3. Manually enter an expression in the text box next to the  *Filter* option or press the  button next to it to open the expression string builder dialog.
4. Use the provided functions and the layer attributes to build an *expression* to filter the features you'd like to retrieve. Press the *Test* button to check the result of the query.
5. You can enter a longer label to complete the rule description.
6. You can use the  *Scale Range* option to set scales at which the rule should be applied.
7. Finally, configure the *symbol to use* for these features.
8. And press *OK*.

A new row summarizing the rule is added to the Layer Properties dialog. You can create as many rules as necessary following the steps above or copy pasting an existing rule. Drag-and-drop the rules on top of each other to nest them and refine the upper rule features in subclasses.

The rule-based renderer can be combined with categorized or graduated renderers. Selecting a rule, you can organize its features in subclasses using the *Refine selected rules* drop-down menu. Refined classes appear like sub-items of the

rule, in a tree hierarchy and like their parent, you can set the symbology and the rule of each class. Automated rule refinement can be based on:

- **scales:** given a list of scales, this option creates a set of classes to which the different user-defined scale ranges apply. Each new scale-based class can have its own symbology and expression of definition. This can e.g. be a convenient way to display the same features with various symbols at different scales, or display only a set of features depending on the scale (e.g. local airports at large scale vs international airports at small scale).
- **categories:** applies a *categorized renderer* to the features falling in the selected rule.
- or **ranges:** applies a *graduated renderer* to the features falling in the selected rule.

Refined classes appear like sub-items of the rule, in a tree hierarchy and like above, you can set symbology of each class. Symbols of the nested rules are stacked on top of each other so be careful in choosing them. It is also possible to uncheck ☐ *Symbols* in the *Edit rule* dialog to avoid rendering a particular symbol in the stack.

In the *Edit rule* dialog, you can avoid writing all the rules and make use of the ☐ *Else* option to catch all the features that do not match any of the other rules, at the same level. This can also be achieved by writing `ELSE` in the *Rule* column of the *Layer Properties* ► *Symbology* ► *Rule-based* dialog.

Right-clicking over selected item(s) shows a contextual menu to:

- *Copy* and *Paste*, a convenient way to create new item(s) based on existing item(s)
- *Copy Symbol* and *Paste Symbol*, a convenient way to apply the item's representation to others
- *Change Color...* of the selected symbol(s)
- *Change Opacity...* of the selected symbol(s)
- *Change Output Unit...* of the selected symbol(s)
- *Change Width...* of the selected line symbol(s)
- *Change Size...* of the selected point symbol(s)
- *Change Angle...* of the selected point symbol(s)
- *Refine Current Rule:* open a submenu that allows to refine the current rule with **scales**, **categories** or **Ranges**. Same as selecting the *corresponding menu* at the bottom of the dialog.

Unchecking a row in the rule-based renderer dialog hides in the map canvas the features of the specific rule and the nested ones.

The created rules also appear in a tree hierarchy in the map legend. Double-click an entry in the map legend to edit the assigned symbol. Right-click and you will get some *more options*.

The example in Abb. 16.7 shows the rule-based rendering dialog for the rivers layer of the QGIS sample dataset.

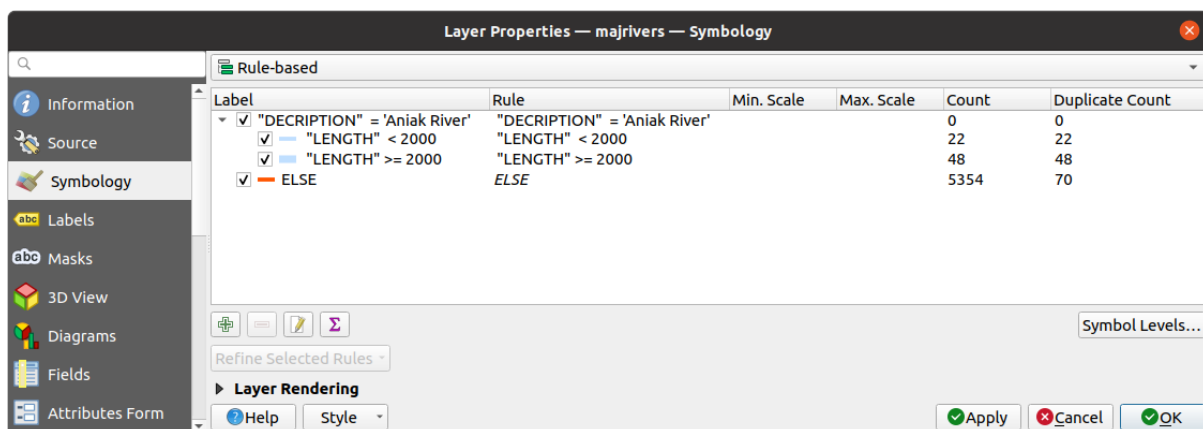



Abb. 16.7: Regelbasierte Symbolisierungsoptionen

## Point displacement Renderer

The  *Point Displacement* renderer takes the point features falling in a given distance tolerance from each other and places their symbols around their barycenter, following different placement methods. This can be a convenient way to visualize all the features of a point layer, even if they have the same location (e.g. amenities in a building).

To configure a point displacement renderer, you have to:

1. Set the *Center symbol*: how the virtual point at the center will look like
2. Select the *Renderer* type: how you want to classify features in the layer (single, categorized, rule-based...)
3. Press the *Renderer Settings...* button to configure features' symbology according to the selected renderer
4. Indicate the *Distance* tolerance in which close features are considered overlapping and then displaced over the same virtual point. Supports common symbol units.
5. Configure the *Placement methods*:
  - **Ring**: places all the features on a circle whose radius depends on the number of features to display.
  - **Concentric rings**: uses a set of concentric circles to show the features.
  - **Grid**: generates a regular grid with a point symbol at each intersection.
6. Displaced symbols are placed on the *Displacement lines*. While the minimal spacing of the displacement lines depends on the point symbols renderer, you can still customize some of their settings such as the *Stroke width*, *Stroke color* and *Size adjustment* (e.g., to add more spacing between the rendered points).
7. Use the *Labels* group options to perform points labeling: the labels are placed near the displaced symbol, and not at the feature real position.
  1. Select the *Label attribute*: a field of the layer to use for labeling
  2. Indicate the *Label font* properties and size
  3. Pick a *Label color*
  4. Set a *Label distance factor*: for each point feature, offsets the label from the symbol center proportionally to the symbol's diagonal size.
  5. Turn on ☐ *Use scale dependent labeling* if you want to display labels only on scales larger than a given *Minimum map scale*.

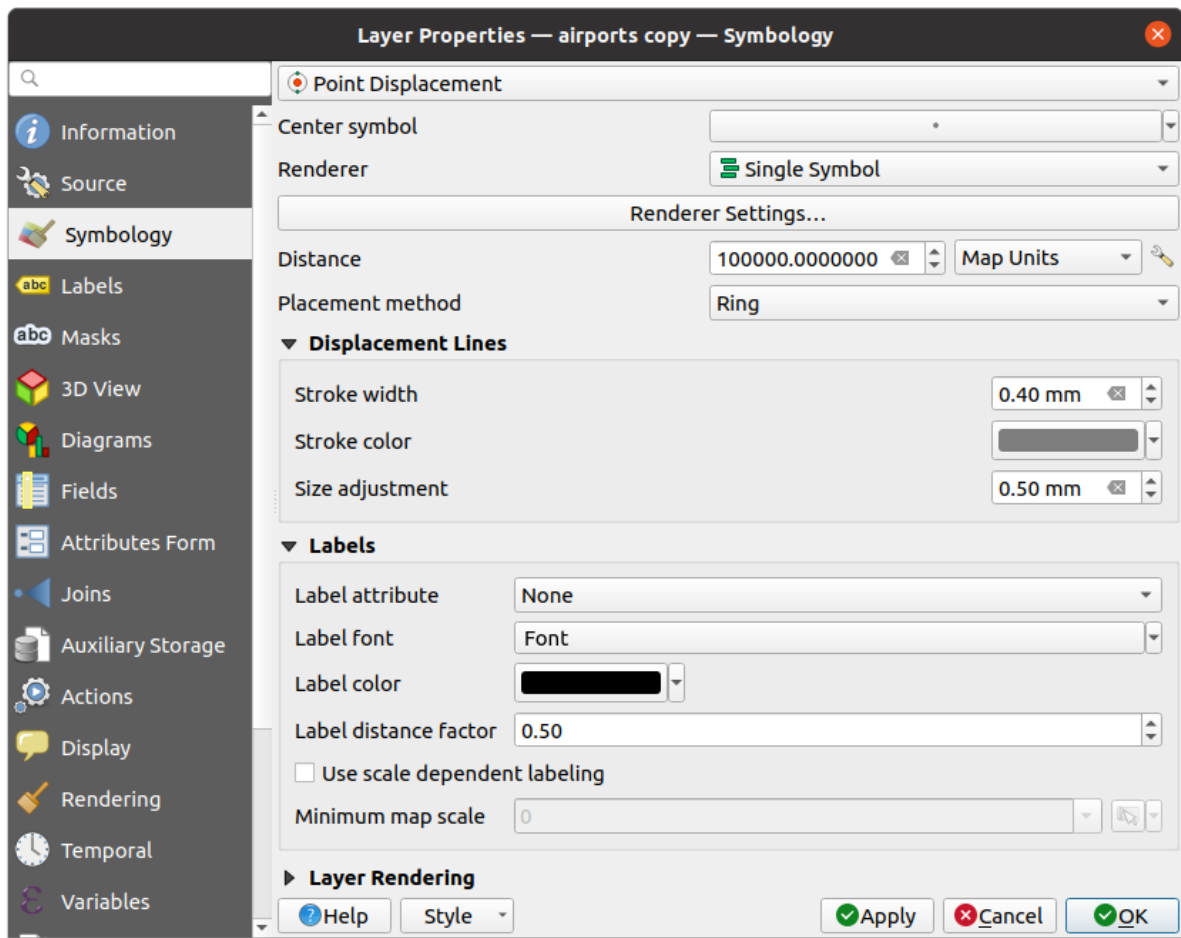




Abb. 16.8: Dialog Punktverdrängung

**Bemerkung:** Point Displacement renderer does not alter feature geometry, meaning that points are not moved from their position. They are still located at their initial place. Changes are only visual, for rendering purpose. Use instead the Processing [Points displacement](#) algorithm if you want to create displaced features.

## Point Cluster Renderer

Unlike the  *Point Displacement* renderer which blows up nearest or overlaid point features placement, the  *Point Cluster* renderer groups nearby points into a single rendered marker symbol. Points that fall within a specified distance from each others are merged into a single symbol. Points aggregation is made based on the closest group being formed, rather than just assigning them the first group within the search distance.

From the main dialog, you can:

1. Set the symbol to represent the point cluster in the *Cluster symbol*; the default rendering displays the number of aggregated features thanks to the `@cluster_size` [variable](#) on Font marker symbol layer.
2. Select the *Renderer* type, i.e. how you want to classify features in the layer (single, categorized, rule-based...)
3. Press the *Renderer Settings...* button to configure features' symbology as usual. Note that this symbology is only visible on features that are not clustered, the *Cluster symbol* being applied otherwise. Also, when all the point features in a cluster belong to the same rendering class, and thus would be applied the same color, that color represents the `@cluster_color` variable of the cluster.
4. Indicate the maximal *Distance* to consider for clustering features. Supports common symbol units.

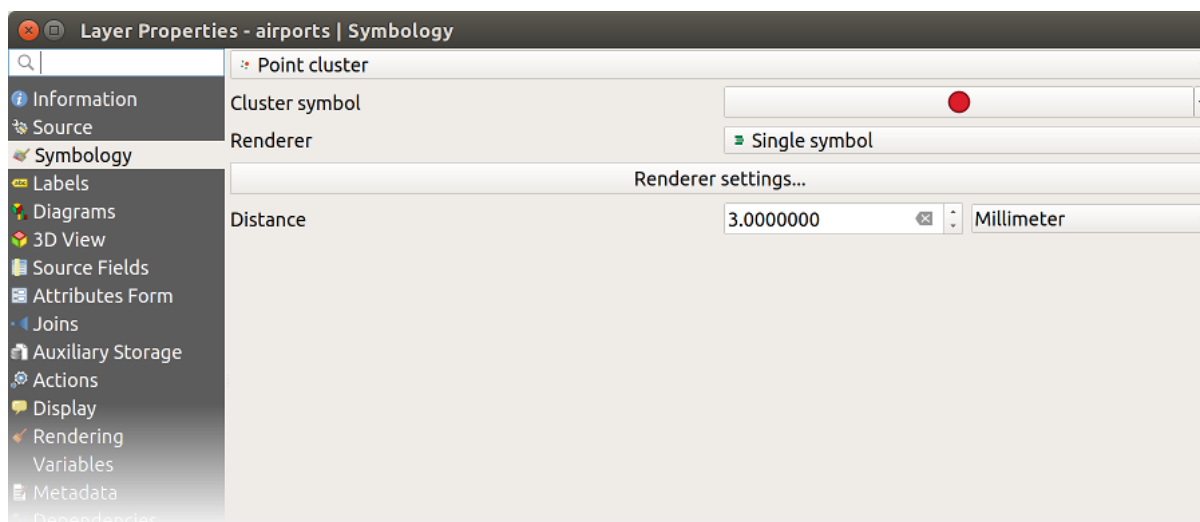




Abb. 16.9: Point Cluster dialog

**Bemerkung:** Point Cluster renderer does not alter feature geometry, meaning that points are not moved from their position. They are still located at their initial place. Changes are only visual, for rendering purpose. Use instead the Processing *K-means clustering* or *DBSCAN clustering* algorithm if you want to create cluster-based features.

### Merged Features Renderer

The  *Merged Features* renderer allows area and line features to be “dissolved” into a single object prior to rendering to ensure that complex symbols or overlapping features are represented by a uniform and contiguous cartographic symbol.

### Inverted Polygon Renderer

The  *Inverted Polygon* renderer allows user to define a symbol to fill in outside of the layer’s polygons. As above you can select subrenderers, namely Single symbol, Graduated, Categorized, Rule-Based or 2.5D renderer.



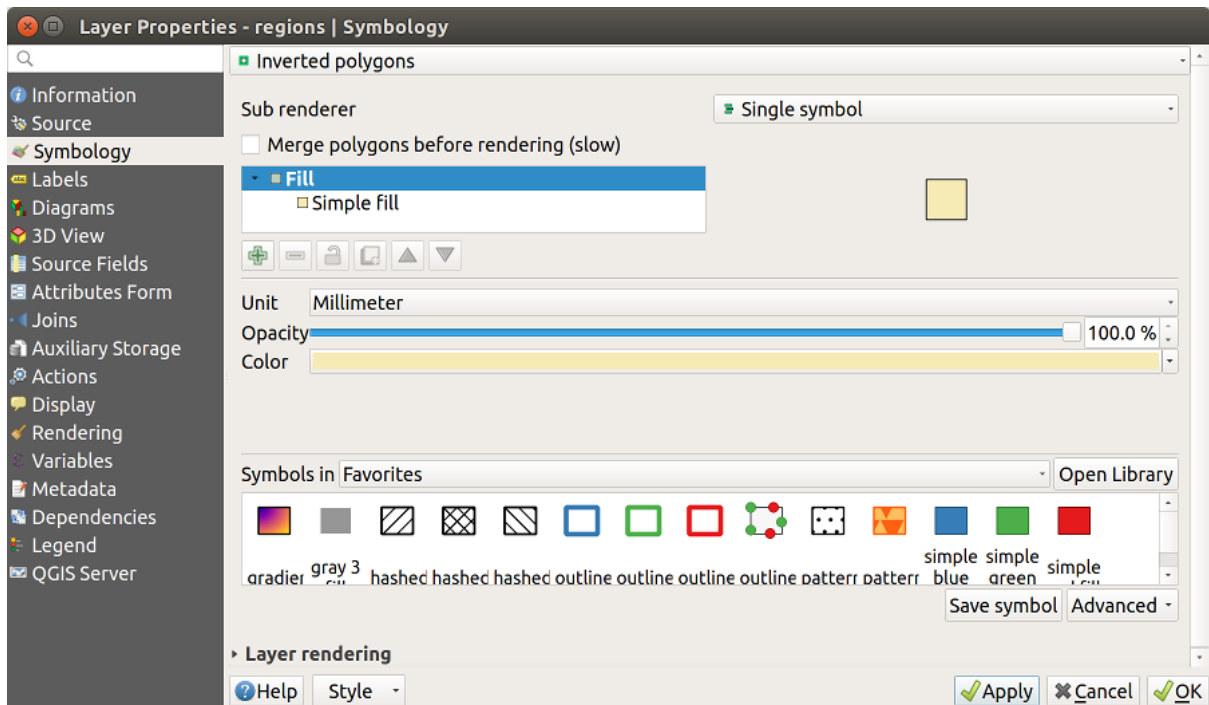



Abb. 16.10: Umgekehrte Polygone Dialog

## Heatmap Renderer

With the  *Heatmap* renderer you can create live dynamic heatmaps for (multi)point layers. You can specify the heatmap radius in millimeters, points, pixels, map units or inches, choose and edit a color ramp for the heatmap style and use a slider for selecting a trade-off between render speed and quality. You can also define a maximum value limit and give a weight to points using a field or an expression. When adding or removing a feature the heatmap renderer updates the heatmap style automatically.

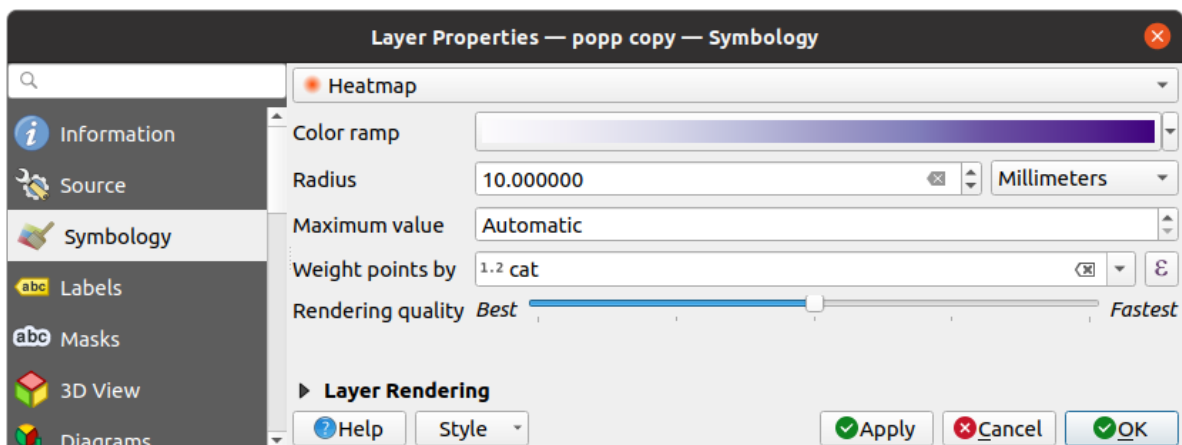




Abb. 16.11: Der Heatmap-Erweiterung Dialog

## 2.5D Renderer

Using the  2.5D renderer it's possible to create a 2.5D effect on your layer's features. You start by choosing a *Height* value (in map units). For that you can use a fixed value, one of your layer's fields, or an expression. You also need to choose an *Angle* (in degrees) to recreate the viewer position (0° means west, growing in counter clock wise). Use advanced configuration options to set the *Roof Color* and *Wall Color*. If you would like to simulate solar radiation on the features walls, make sure to check the  *Shade walls based on aspect* option. You can also simulate a shadow by setting a *Color* and *Size* (in map units).

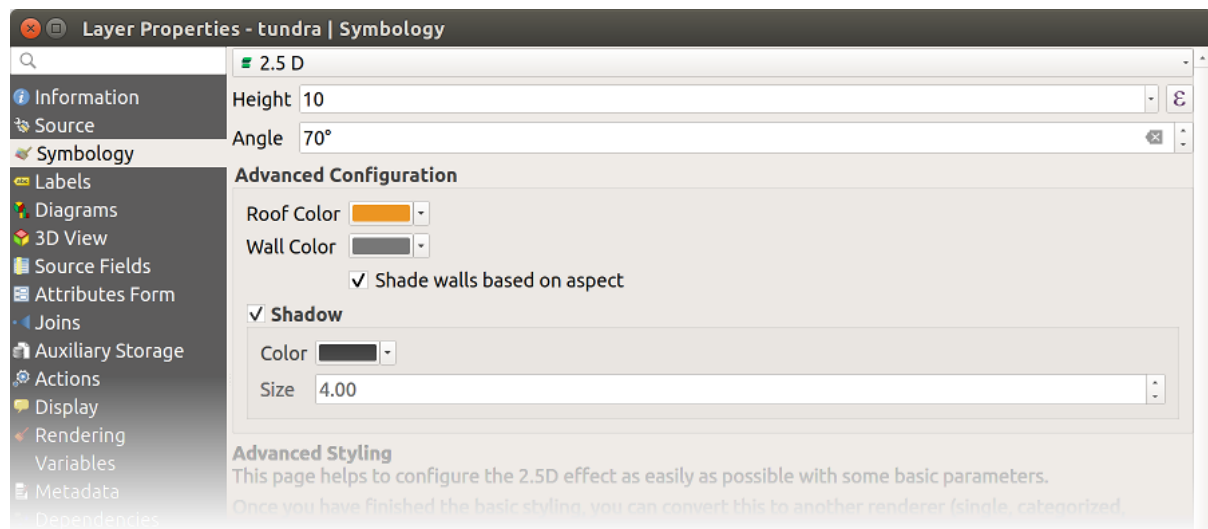


Abb. 16.12: 2.5D dialog

### Tipp: Using 2.5D effect with other renderers



Once you have finished setting the basic style on the 2.5D renderer, you can convert this to another renderer (single, categorized, graduated). The 2.5D effects will be kept and all other renderer specific options will be available for you to fine tune them (this way you can have for example categorized symbols with a nice 2.5D representation or add some extra styling to your 2.5D symbols). To make sure that the shadow and the „building“ itself do not interfere with other nearby features, you may need to enable Symbols Levels ( *Advanced ► Symbol levels...*). The 2.5D height and angle values are saved in the layer's variables, so you can edit it afterwards in the variables tab of the layer's properties dialog.

## Embedded Renderer

The *Embedded Symbols* renderer allows to display the ‚native‘ symbology of a provided datasource. This is mostly the case with KML and TAB datasets that have predefined symbology.

## Layerdarstellung

From the Symbology tab, you can also set some options that invariably act on all features of the layer:

- **Opacity** : You can make the underlying layer in the map canvas visible with this tool. Use the slider to adapt the visibility of your vector layer to your needs. You can also make a precise definition of the percentage of visibility in the menu beside the slider.
- **Blending mode** at the *Layer* and *Feature* levels: You can achieve special rendering effects with these tools that you may previously only know from graphics programs. The pixels of your overlaying and underlying layers are mixed through the settings described in *Mischmodi*.
- Wenden Sie *Zeicheneffekte* auf alle Layerobjekte an, mit dem *Zeicheneffekte* Knopf.
- **Control feature rendering order** allows you, using features attributes, to define the z-order in which they shall be rendered. Activate the checkbox and click on the  button beside. You then get the *Define Order* dialog in which you:
  1. Choose a field or build an expression to apply to the layer features.
  2. Set in which order the fetched features should be sorted, i.e. if you choose **Ascending** order, the features with lower value are rendered under those with higher value.
  3. Define when features returning NULL value should be rendered: **first** (bottom) or **last** (top).
  4. Repeat the above steps as many times as rules you wish to use.

The first rule is applied to all the features in the layer, z-ordering them according to their returned value. Then, within each group of features with the same value (including those with NULL value) and thus the same z-level, the next rule is applied to sort them. And so on...

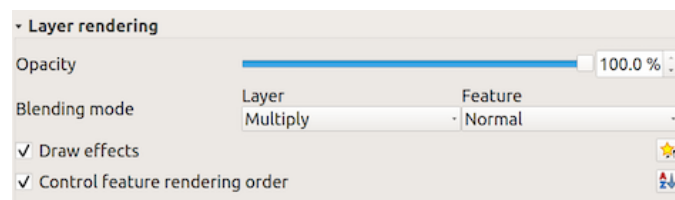



Abb. 16.13: Layerdarstellungsoptionen

## Andere Einstellungen

### Symbol levels

Für Renderer, die Symbollayer gestapelt ermöglichen (nur Heatmap nicht) gibt es eine Option, um die Darstellungsreihenfolge der einzelnen Symbolebenen zu steuern.

For most of the renderers, you can access the Symbols levels option by clicking the *Advanced* button below the saved symbols list and choosing *Symbol levels*. For the *Rule-based Renderer* the option is directly available through *Symbols Levels...* button, while for *Point displacement Renderer* renderer the same button is inside the *Rendering settings* dialog.

Um die Symbolebenen zu aktivieren, wählen Sie  *Symbolebenen aktivieren*. Jede Reihe zeigt eine kleine Vorschau des kombinierten Symbols, seiner Beschriftung und die individuellen Symbollayer, unterteilt in verschiedene Spalten mit einer Nummer. Die Nummern zeigen die Darstellungsreihenfolge, in der die Symbollayer gezeichnet werden.

Niedrige Werteebenen werden zuerst gezeichnet, liegen ganz unten, während höhere Werte als letztes gezeichnet werden und über den anderen liegen.

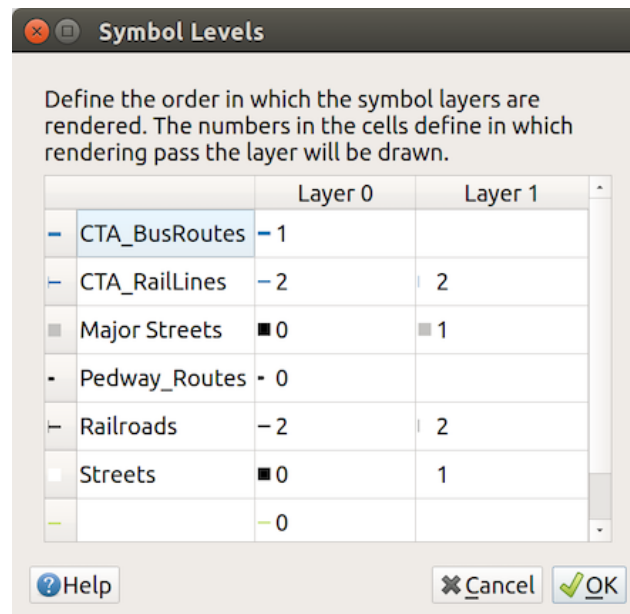


Abb. 16.14: Symbolebenen Dialog

**Bemerkung:** Wenn Symbolebenen deaktiviert werden, werden alle Symbole entsprechend ihren jeweiligen Objektreihenfolge gezeichnet. Überlappende Symbole werden einfach zu anderen darunter verschleiert. Außerdem „verschmelzen“ ähnliche Symbole nicht mit anderen.

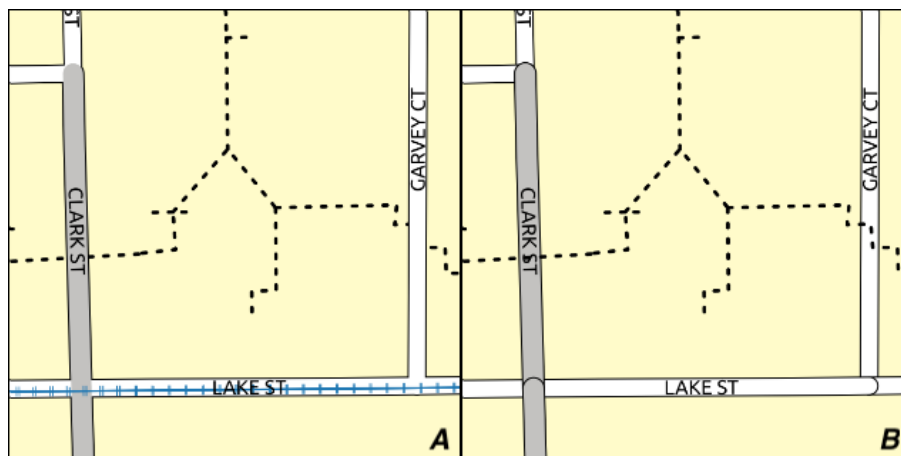








Abb. 16.15: Unterschied von aktivierten (A) und deaktivierten (B) Symbolebenen

## Data-defined size legend

When a layer is rendered with the *proportional symbol* or the *multivariate rendering* or when a *scaled size diagram* is applied to the layer, you can allow the display of the scaled symbols in both the *Layers panel* and the *print layout legend*.

To enable the *Data-defined Size Legend* dialog to render symbology, select the eponym option in the *Advanced* button below the saved symbols list. For diagrams, the option is available under the *Legend* tab. The dialog provides the following options to:

- select the type of legend:  *Legend not enabled*,  *Separated legend items* and  *Collapsed legend*. For the latter option, you can select whether the legend items are aligned at the **Bottom** or at the **Center**;
- preview the *symbol to use* for legend representation;
- insert the title in the legend;
- resize the classes to use: by default, QGIS provides you with a legend of five classes (based on natural pretty breaks) but you can apply your own classification using the  *Manual size classes* option. Use the  and  buttons to set your custom classes values and labels.
- For collapsed legend, it's possible to:
  - *Align symbols* in the center or the bottom
  - configure the horizontal leader *Line symbol* from the symbol to the corresponding legend text.

A preview of the legend is displayed in the right panel of the dialog and updated as you set the parameters.

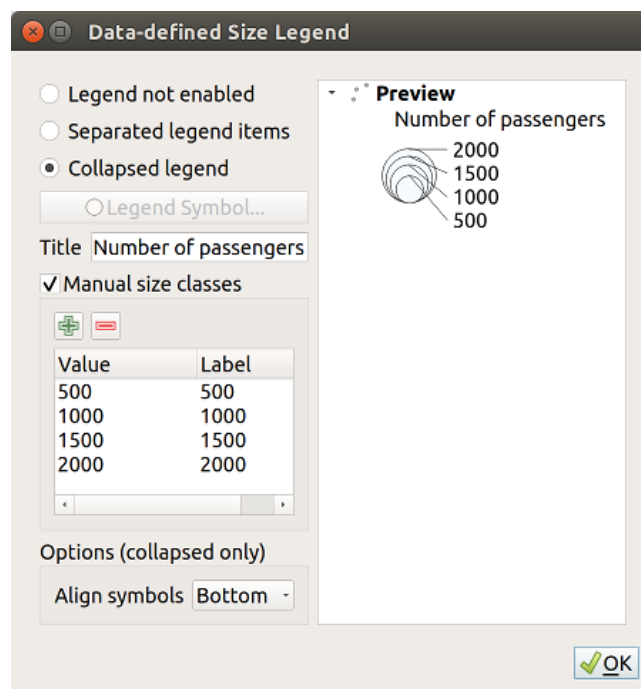


Abb. 16.16: Setting size scaled legend

**Bemerkung:** Currently, data-defined size legend for layer symbology can only be applied to point layer using single, categorized or graduated symbology.


## Animation settings

To allow any symbol to become an *animated symbol*, you can utilize *Animation settings* panel. In this panel, you can enable animation for the symbol and set a specific frame rate for the symbol's redrawing.


1. Start by going to the top symbol level and select *Advanced* menu in the bottom right of the dialog
2. Find *Animation settings* option
3. Check ☒ *Is Animated* to enable animation for the symbol
4. Configure the *Frame rate*, i.e. how fast the animation would be played
5. You can now use `@symbol_frame` variable in any sub-symbol data defined property in order to animate that property.

For example, setting the symbol's rotation to data defined expression `@symbol_frame % 360` will cause the symbol to rotate over time, with rotation speed dictated by the symbol's frame rate:

## Zeicheneffekte

Damit die Layerdarstellung verbessert und vermieden (oder zumindest reduziert) wird, dass andere Software auf die endgültige Darstellung der Karte umsortiert, bietet QGIS eine weitere leistungsfähige Funktionalität: Die Option  *Zeicheneffekte*, die zur Anpassung Zeicheneffekte für die Visualisierung von Vektorlayern hinzufügt.

The option is available in the *Layer Properties* ► *Symbology* dialog, under the *Layer rendering* group (applying to the whole layer) or in *symbol layer properties* (applying to corresponding features). You can combine both usage.

Paint effects can be activated by checking the ☒ *Draw effects* option and clicking the  *Customize effects* button. That will open the *Effect Properties* Dialog (see Abb. 16.17). The following effect types, with custom options are available:

- **Source:** Draws the feature's original style according to the configuration of the layer's properties. The *Opacity* of its style can be adjusted as well as the *Blend mode* and *Draw mode*. These are common properties for all types of effects.

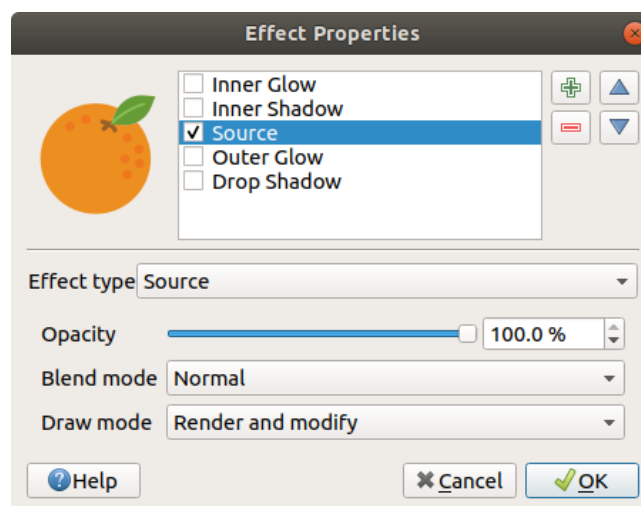


Abb. 16.17: Zeicheneffekte: Dialog Quelle

- **Blur:** Adds a blur effect on the vector layer. The custom options that you can change are the *Blur type* (*Stack blur* (fast) or *Gaussian blur* (quality)) and the *Blur strength*.

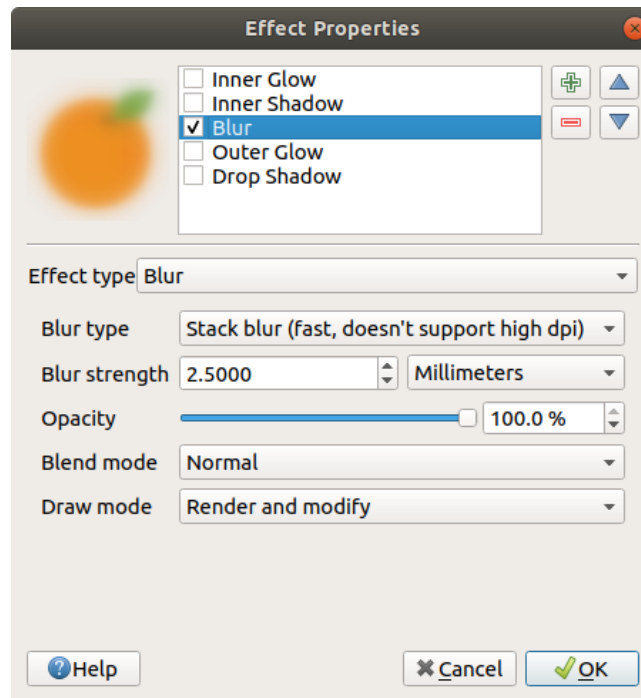




Abb. 16.18: Zeicheneffekte: Dialog verwischen

- **Colorise:** This effect can be used to make a version of the style using one single hue. The base will always be a grayscale version of the symbol and you can:
  - Use the  *Grayscale* to select how to create it: options are ‚By lightness‘, ‚By luminosity‘, ‚By average‘ and ‚Off‘.
  - If  *Colorise* is selected, it will be possible to mix another color and choose how strong it should be.
  - Control the *Brightness*, *Contrast* and *Saturation* levels of the resulting symbol.

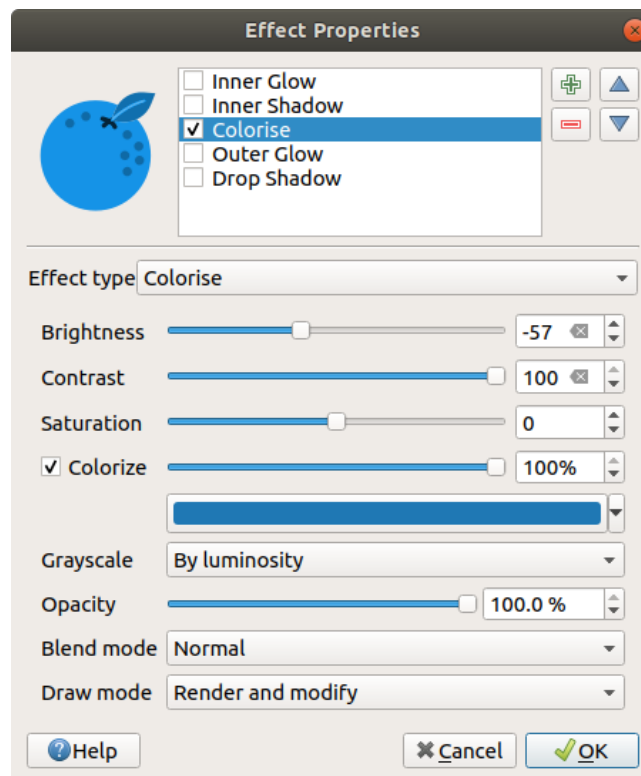


Abb. 16.19: Zeicheneffekte: Dialog einfärben

- **Drop Shadow:** Using this effect adds a shadow on the feature, which looks like adding an extra dimension. This effect can be customized by changing the *Offset* angle and distance, determining where the shadow shifts towards to and the proximity to the source object. *Drop Shadow* also has the option to change the *Blur radius* and the *Color* of the effect.

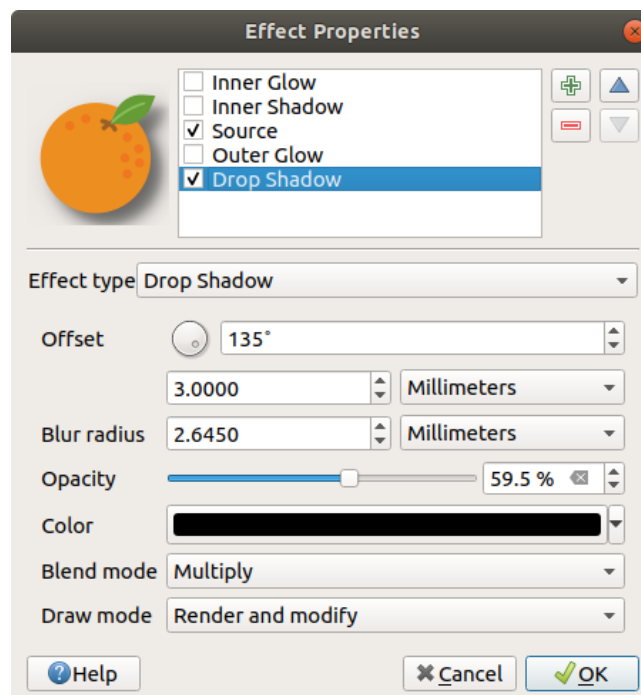


Abb. 16.20: Zeicheneffekte: Dialog Schattenwurf



- **Inner Shadow:** This effect is similar to the *Drop Shadow* effect, but it adds the shadow effect on the inside of the edges of the feature. The available options for customization are the same as the *Drop Shadow* effect.

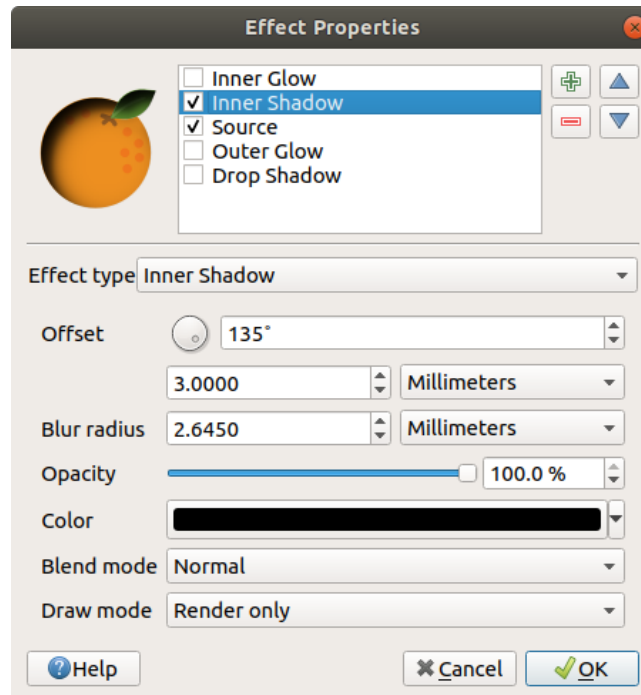


Abb. 16.21: Zeicheneffekte: Dialog Innerer Schatten

- **Inner Glow:** Adds a glow effect inside the feature. This effect can be customized by adjusting the *Spread* (width) of the glow, or the *Blur radius*. The latter specifies the proximity from the edge of the feature where you want any blurring to happen. Additionally, there are options to customize the color of the glow using a *Single color* or a *Color ramp*.

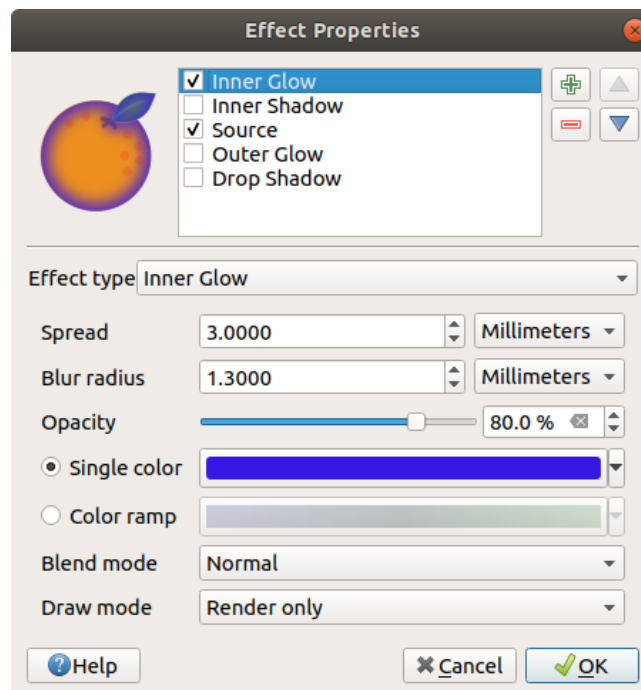


Abb. 16.22: Zeicheneffekte: Dialog Inneres Glühen

- **Outer Glow:** This effect is similar to the *Inner Glow* effect, but it adds the glow effect on the outside of the edges of the feature. The available options for customization are the same as the *Inner Glow* effect.

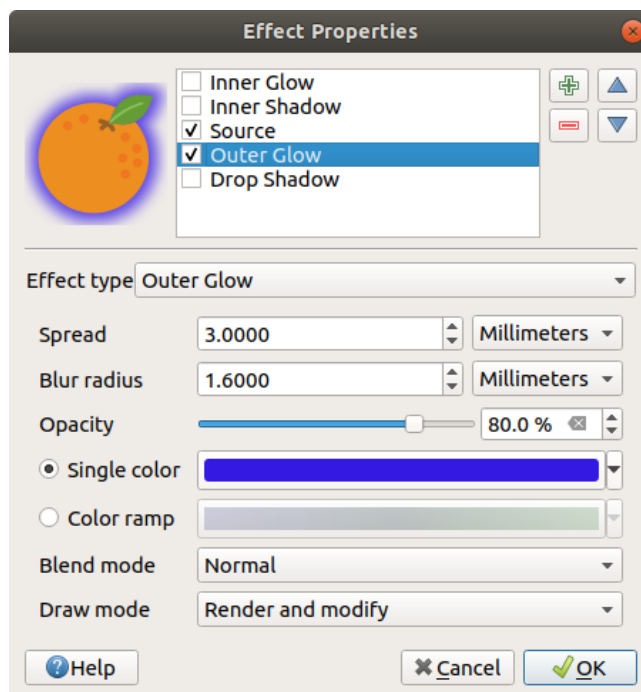


Abb. 16.23: Zeicheneffekte: Dialog Äußeres Glühen

- **Transform:** Adds the possibility of transforming the shape of the symbol. The first options available for customization are the *Reflect horizontal* and *Reflect vertical*, which actually create a reflection on the horizontal and/or vertical axes. The other options are:
  - *Shear X,Y*: Slants the feature along the X and/or Y axis.
  - *Scale X,Y*: Enlarges or minimizes the feature along the X and/or Y axis by the given percentage.
  - *Rotation*: Turns the feature around its center point.
  - and *Translate X,Y* changes the position of the item based on a distance given on the X and/or Y axis.

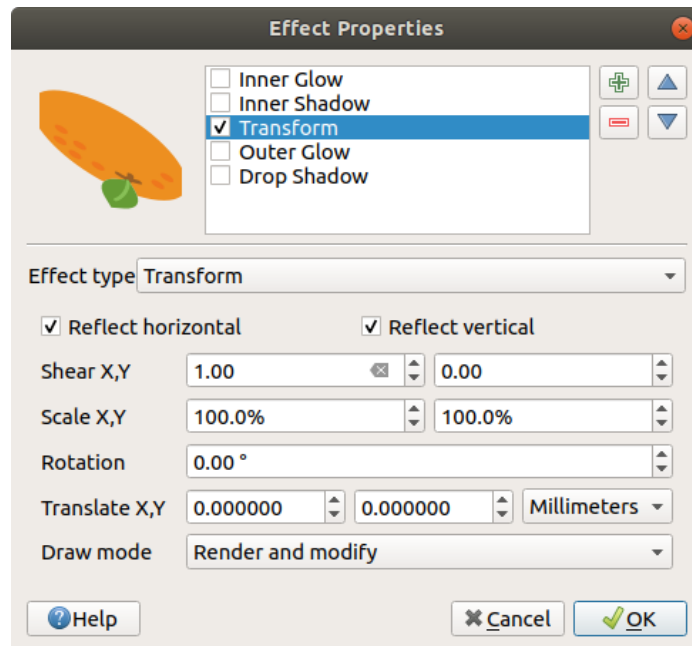


Abb. 16.24: Zeicheneffekte: Dialog transformieren

One or more effect types can be used at the same time. You (de)activate an effect using its checkbox in the effects list. You can change the selected effect type by using the *Effect type* option. You can reorder the effects using and buttons, and also add/remove effects using the *Add new effect* and *Remove effect* buttons.

There are some common options available for all draw effect types. *Opacity* and *Blend mode* options work similar to the ones described in *Layerdarstellung* and can be used in all draw effects except for the transform one.

There is also a *Draw mode* option available for every effect, and you can choose whether to render and/or modify the symbol, following some rules:


- Effects render from top to bottom.
- *Render only* mode means that the effect will be visible.
- *Modifier only* mode means that the effect will not be visible but the changes that it applies will be passed to the next effect (the one immediately below).
- The *Render and Modify* mode will make the effect visible and pass any changes to the next effect. If the effect is at the top of the effects list or if the immediately above effect is not in modify mode, then it will use the original source symbol from the layers properties (similar to source).

### 16.1.4 Labels Properties

The *Labels* properties provides you with all the needed and appropriate capabilities to configure smart labeling on vector layers. This dialog can also be accessed from the *Layer Styling* panel, or using the *Layer Labeling Options* button of the **Labels toolbar**.

The first step is to choose the labeling method from the drop-down list. Available methods are:

- *No labels*: the default value, showing no labels from the layer
- *Single labels*: Show labels on the map using a single attribute or an expression
- *Rule-based labeling*

- and  *Blocking*: allows to set a layer as just an obstacle for other layer's labels without rendering any labels of its own.

The next steps assume you select the  *Single labels* option, opening the following dialog.

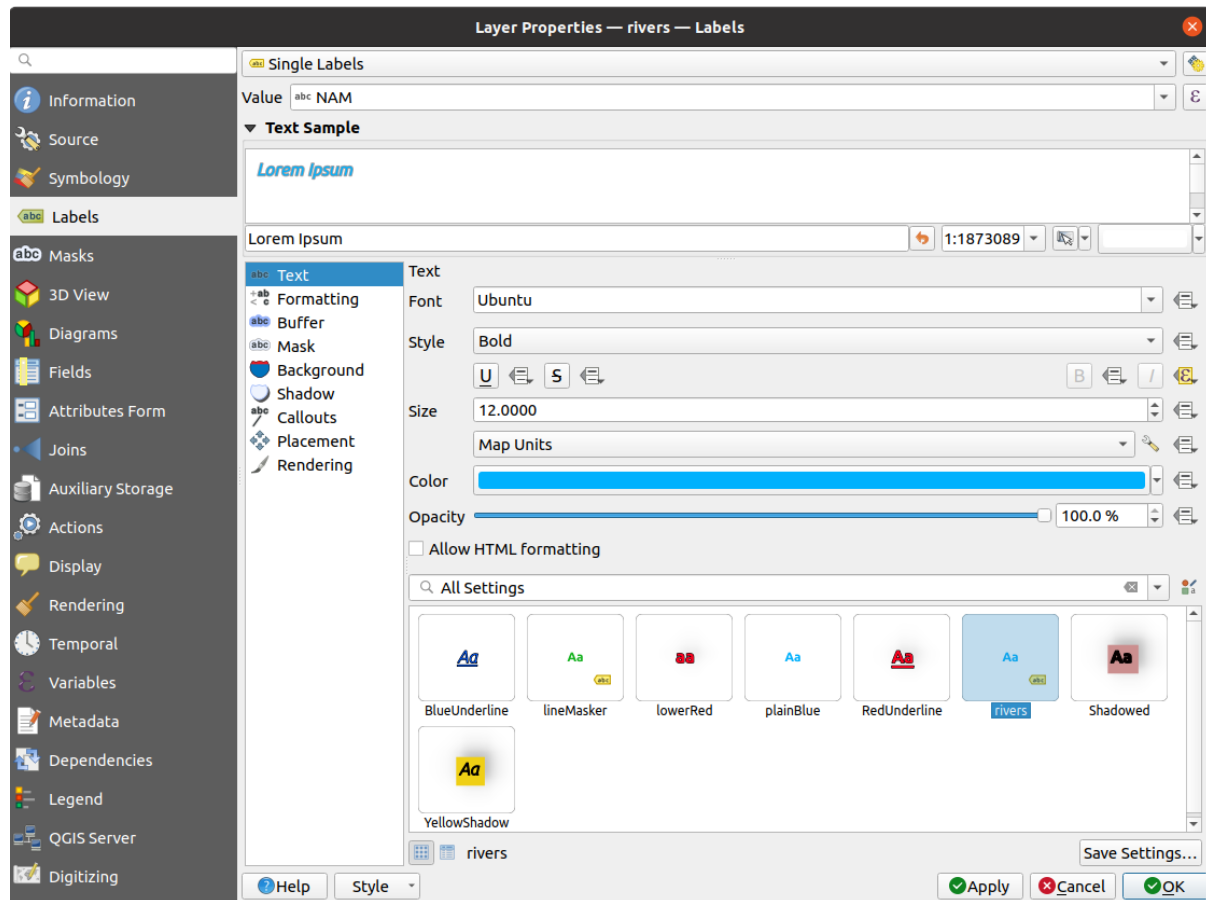



Abb. 16.25: Layer labeling settings - Single labels







At the top of the dialog, a *Value* drop-down list is enabled. You can select an attribute column to use for labeling. By default, the *display field* is used. Click  if you want to define labels based on expressions - See [Ausdrucksbasierte Beschriftungen definieren](#).

---

**Bemerkung:** Labels with their formatting can be displayed as entries in the legends, if enabled in the [Legend](#) tab.

---


Below are displayed options to customize the labels, under various tabs:

-  *Text*
-  *Formatting*
-  *Buffer*
-  *Mask*
-  *Background*
-  *Shadow*
-  *Callouts*

-  *Placement*
-  *Rendering*

Description of how to set each property is exposed at *Setting a label*.

## Setting the automated placement engine

You can use the automated placement settings to configure a project-level automated behavior of the labels. In the top right corner of the *Labels* tab, click the  Automated placement settings (applies to all layers) button, opening a dialog with the following options:

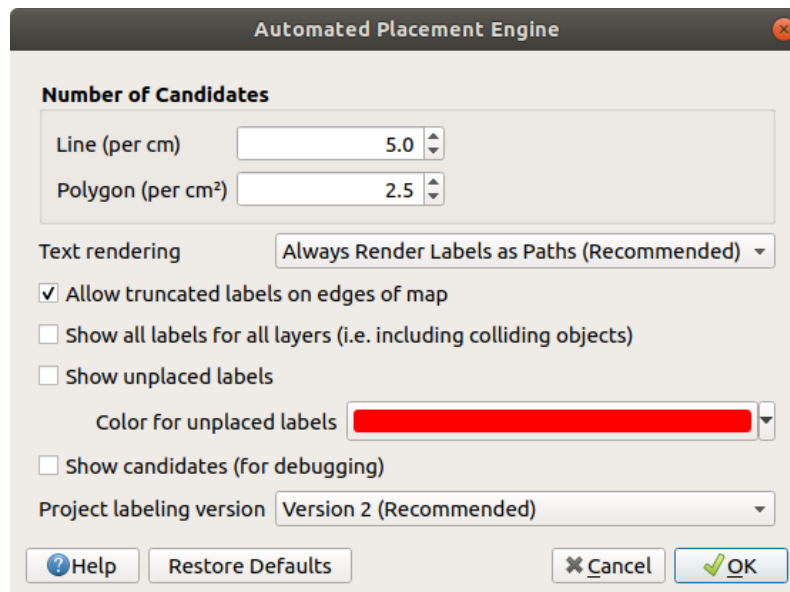


Abb. 16.26: The labels automated placement engine



- *Number of candidates*: calculates and assigns to line and polygon features the number of possible labels placement based on their size. The longer or wider a feature is, the more candidates it has, and its labels can be better placed with less risk of collision.
- *Text rendering*: sets the default value for label rendering widgets when *exporting a map canvas* or *a layout* to PDF or SVG. If *Always render labels as text* is selected then labels can be edited in external applications (e.g. Inkscape) as normal text. BUT the side effect is that the rendering quality is decreased, and there are issues with rendering when certain text settings like buffers are in place. That's why *Always render labels as paths (recommended)* which exports labels as outlines, is recommended.
- ☒ *Allow truncated labels on edges of map*: controls whether labels which fall partially outside of the map extent should be rendered. If checked, these labels will be shown (when there's no way to place them fully within the visible area). If unchecked then partially visible labels will be skipped. Note that this setting has no effects on labels' display in the *layout map item*.
- ☐ *Show all labels for all layers (i.e. including colliding objects)*. Note that this option can be also set per layer (see *Rendering tab*)
- ☐ *Show unplaced labels*: allows to determine whether any important labels are missing from the maps (e.g. due to overlaps or other constraints). They are displayed using a customizable color.
- ☐ *Show candidates (for debugging)*: controls whether boxes should be drawn on the map showing all the candidates generated for label placement. Like the label says, it's useful only for debugging and testing the effect different labeling settings have. This could be handy for a better manual placement with tools from the *label toolbar*.

- *Project labeling version*: QGIS supports two different versions of label automatic placement:
  - *Version 1*: the old system (used by QGIS versions 3.10 and earlier, and when opening projects created in these versions in QGIS 3.12 or later). Version 1 treats label and obstacle priorities as „rough guides“ only, and it’s possible that a low-priority label will be placed over a high-priority obstacle in this version. Accordingly, it can be difficult to obtain the desired labeling results when using this version and it is thus recommended only for compatibility with older projects.
  - *Version 2 (recommended)*: this is the default system in new projects created in QGIS 3.12 or later. In version 2, the logic dictating when labels are allowed to overlap *obstacles* has been reworked. The newer logic forbids any labels from overlapping any obstacles with a greater obstacle weight compared to the label’s priority. As a result, this version results in much more predictable and easier to understand labeling results.

### Regelbasierte Beschriftung

With rule-based labeling multiple label configurations can be defined and applied selectively on the base of expression filters and scale range, as in *Rule-based rendering*.

To create a rule:

1. Select the  **Rule-based labeling** option in the main drop-down list from the *Labels* tab
2. Click the  **Add rule** button at the bottom of the dialog.
3. Fill the new dialog with:
  - *Description*: a text used to identify the rule in the *Labels* tab and as a *label legend entry* in the print layout legend
  - *Filter*: an expression to select the features to apply the label settings to
  - If there are rules already set, the *Else* option can be used to select all the features not matching any filter of the rules in the same group.
4. You can set a *scale range* in which the label rule should be applied.
5. The options available under the *Labels* group box are the usual *label settings*. Configure them and press *OK*.

**Edit Rule**

Description

☒ Filter

☐ Else Catch-all for other features

☒ Scale range

Minimum (exclusive)   Maximum (inclusive)

☒ Labels

Label with

▼ Text Sample

abc Text  
+ab < c Formatting  
abc Buffer  
Background  
Shadow  
**Placement**  
Rendering

**Placement**

☐ Offset from centroid ☐ Horizontal (slow)  
☒ Around centroid ☐ Free (slow)  
☐ Using perimeter ☐ Using perimeter (curved)

Centroid ☒ visible polygon ☐ whole polygon

☐ Force point inside polygon

Distance

▼ Data defined

Coordinate X  Y



Alignment

Rotation  ☒ Preserve data rotation values

▼ Priority

Low  High

Abb. 16.27: Regeleigenschaften

A summary of existing rules is shown in the main dialog (see Abb. 16.28). You can add multiple rules, reorder or imbricate them with a drag-and-drop. You can as well remove them with the  button or edit them with  button or a double-click.

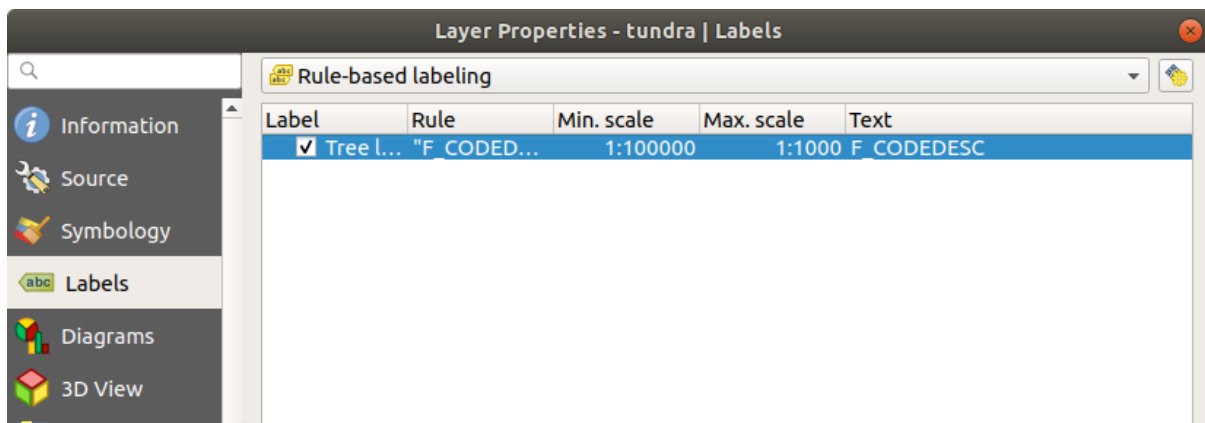




Abb. 16.28: Regelbasierte Beschriftung Bedienfelder

## Ausdrucksbasierte Beschriftungen definieren

Whether you choose single or rule-based labeling type, QGIS allows using expressions to label features.

Assuming you are using the *Single labels* method, click the  button near the *Value* drop-down list in the  *Labels* tab of the properties dialog.

In Abb. 16.29, you see a sample expression to label the alaska trees layer with tree type and area, based on the field 'VEGDESC', some descriptive text, and the function `$area` in combination with `format_number()` to make it look nicer.

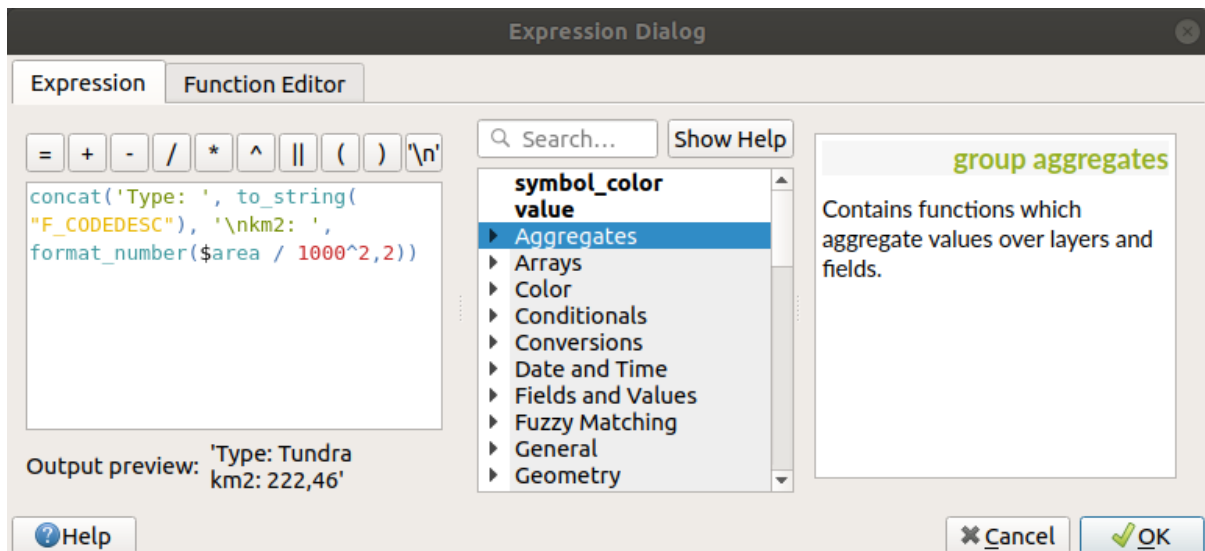


Abb. 16.29: Ausdrücke für das Beschriften verwenden

Expression based labeling is easy to work with. All you have to take care of is that:

- You may need to combine all elements (strings, fields, and functions) with a string concatenation function such as `concat`, `+` or `||`. Be aware that in some situations (when null or numeric value are involved) not all of these tools will fit your need.



- Strings are written in ,single quotes‘.
- Fields are written in „double quotes“ or without any quote.

Schauen wir uns einige Beispiele an:

1. Label based on two fields ,name‘ and ,place‘ with a comma as separator:

```
"name" || ', ' || "place"
```

Returns:

```
John Smith, Paris
```

2. Label based on two fields ,name‘ and ,place‘ with other texts:

```
'My name is ' + "name" + 'and I live in ' + "place"
'My name is ' || "name" || 'and I live in ' || "place"
concat('My name is ', name, ' and I live in ', "place")
```

Returns:

```
My name is John Smith and I live in Paris
```

3. Label based on two fields ,name‘ and ,place‘ with other texts combining different concatenation functions:

```
concat('My name is ', name, ' and I live in ' || place)
```

Returns:

```
My name is John Smith and I live in Paris
```

Or, if the field ,place‘ is NULL, returns:

```
My name is John Smith
```

4. Multi-line label based on two fields ,name‘ and ,place‘ with a descriptive text:

```
concat('My name is ', "name", '\n', 'I live in ' , "place")
```

Returns:

```
My name is John Smith
I live in Paris
```

5. Label based on a field and the \$area function to show the place’s name and its rounded area size in a converted unit:

```
'The area of ' || "place" || ' has a size of '
|| round($area/10000) || ' ha'
```

Returns:

```
The area of Paris has a size of 10500 ha
```

6. Create a CASE ELSE condition. If the population value in field *population* is  $\leq 50000$  it is a town, otherwise it is a city:

```
concat('This place is a ',
CASE WHEN "population" <= 50000 THEN 'town' ELSE 'city' END)
```

Returns:

This place **is** a town

7. Display name for the cities and no label for the other features (for the „city“ context, see example above):


```
CASE WHEN "population" > 50000 THEN "NAME" END
```

Returns:

Paris

Wie Sie im Ausdruckeditor sehen können stehen Ihnen hunderte von Funktionen zur Verfügung um einfache und sehr komplexe Ausdrücke zum Beschriften Ihrer Daten in QGIS zu erstellen. Siehe das [Ausdrücke](#) Kapitel für weitere Informationen und ein Beispiel zu Ausdrücken.


### Datendefinierte Übersteuerung für das Beschriften

With the  Data defined override function, the settings for the labeling are overridden by entries in the attribute table or expressions based on them. This feature can be used to set values for most of the labeling options described above.



For example, using the Alaska QGIS sample dataset, let's label the `airports` layer with their name, based on their military USE, i.e. whether the airport is accessible to :

- military people, then display it in gray color, size 8;
- others, then show in blue color, size 10.


To do this, after you enabled the labeling on the NAME field of the layer (see [Setting a label](#)):

1. Activate the *Text* tab.
2. Click on the  icon next to the *Size* property.
3. Select *Edit...* and type:

```
CASE
  WHEN "USE" like '%Military%' THEN 8 -- because compatible values are
  ↳ 'Military'                                -- and 'Joint Military/Civilian'
  ELSE 10
END
```

4. Press *OK* to validate. The dialog closes and the  button becomes  meaning that an rule is being run.
5. Then click the button next to the color property, type the expression below and validate:

```
CASE
  WHEN "USE" like '%Military%' THEN '150, 150, 150'
  ELSE '0, 0, 255'
END
```

Likewise, you can customize any other property of the label, the way you want. See more details on the  Data-define override widget's description and manipulation in [Datendefinierte Übersteuerung Setup](#) section.

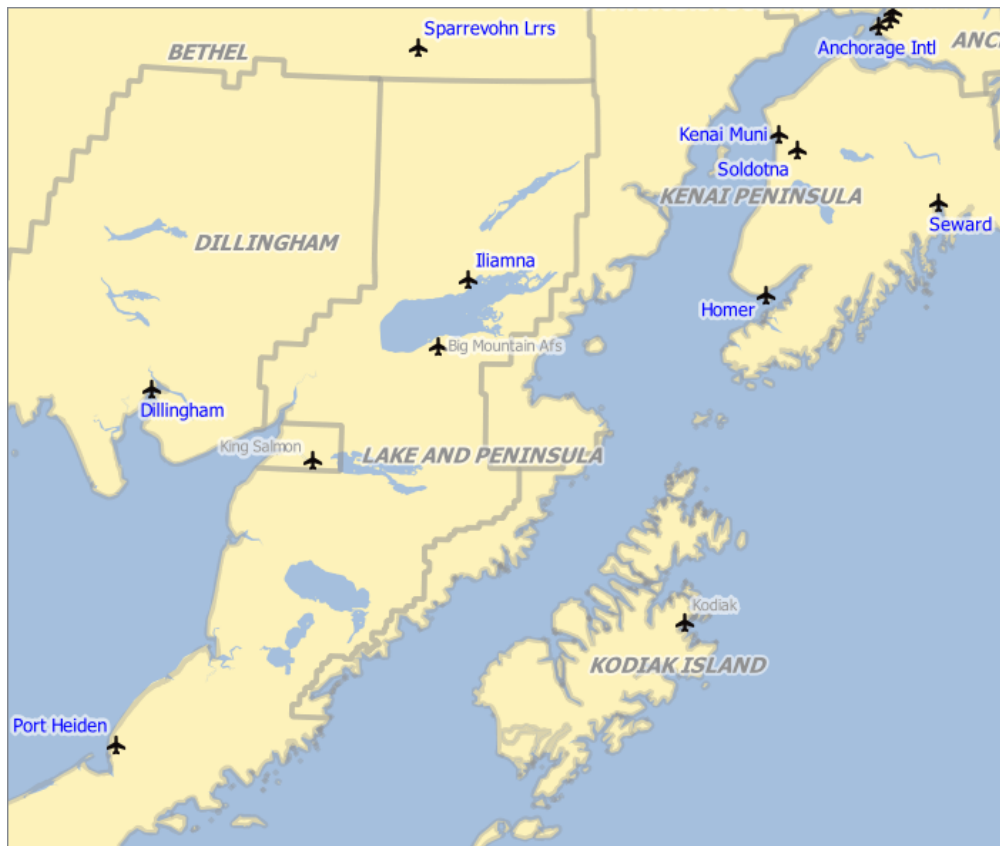


Abb. 16.30: Airports labels are formatted based on their attributes

### Tipp: Use the data-defined override to label every part of multi-part features




There is an option to set the labeling for multi-part features independently from your label properties. Choose the  *Rendering*, *Feature* options, go to the  *Data-define override* button next to the checkbox ☐ *Label every part of multipart-features* and define the labels as described in *Datendefinierte Übersteuerung Setup*.

## The Label Toolbar





The *Label Toolbar* provides some tools to manipulate  *label* (including their *callout*) or  *diagram* properties:



Abb. 16.31: The Label toolbar

-  *Highlight Pinned Labels, Diagrams and Callouts*. If the vector layer of the item is editable, then the highlighting is green, otherwise it's blue.
-  *Toggle Display of Unplaced Labels*. Allows to determine whether any important labels are missing from the maps (e.g. due to overlaps or other constraints). They are displayed with a customizable color (see *Setting the automated placement engine*).
-  *Pin/Unpin Labels and Diagrams*. By clicking or dragging an area, you pin overlaid items. If you click or drag an area

holding **Shift**, the items are unpinned. Finally, you can also click or drag an area holding **Ctrl** to toggle their pin status.

-  **Show/Hide Labels and Diagrams**. If you click on the items, or click and drag an area holding **Shift**, they are hidden. When an item is hidden, you just have to click on the feature to restore its visibility. If you drag an area, all the items in the area will be restored.
-  **Move a Label, Diagram or Callout**. Click to select the item and click to move it to the desired place. The new coordinates are stored in *auxiliary fields*. Selecting the item with this tool and hitting the **Delete** key will delete the stored position value.
-  **Rotate a Label**. Click to select the label and click again to apply the desired rotation. Likewise, the new angle is stored in an auxiliary field. Selecting a label with this tool and hitting the **Delete** key will delete the rotation value of this label.
-  **Change Label Properties**. It opens a dialog to change the clicked label properties; it can be the label itself, its coordinates, angle, font, size, multiline alignment ... as long as this property has been mapped to a field. Here you can set the option to ☒ *Label every part of a feature*.


### Warnung: Label tools overwrite current field values



Using the *Label toolbar* to customize the labeling actually writes the new value of the property in the mapped field. Hence, be careful to not inadvertently replace data you may need later!

**Bemerkung:** The *Auxiliary Storage Properties* mechanism may be used to customize labeling (position, and so on) without modifying the underlying data source.

## Customize the labels from the map canvas

Combined with the *Label Toolbar*, the data defined override setting helps you manipulate labels in the map canvas (move, edit, rotate). We now describe an example using the data-defined override function for the

 **Move Label, Diagram or Callout** function (see Abb. 16.32).

1. Importieren Sie `lakes.shp` aus dem QGIS Beispieldatensatz.
2. Doppelklicken Sie den Layer um die Layereigenschaften zu öffnen. Klicken Sie auf *Beschriftungen* und *Platzierung*. Wählen Sie  *Abstand vom Punkt*.
3. Look for the *Data defined* entries. Click the  icon to define the field type for the *Coordinate*. Choose `xlabel` for X and `ylabel` for Y. The icons are now highlighted in yellow.

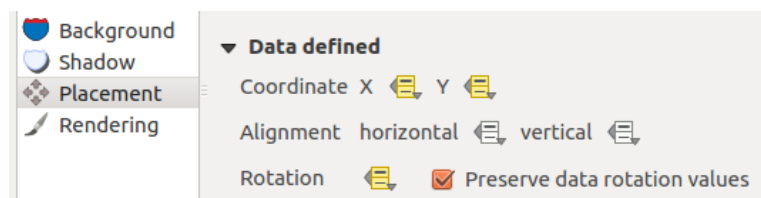




Abb. 16.32: Das Beschriften von Polygonlayern mit datendefinierter Übersteuerung

4. Zoomen Sie auf einen See.
5. Set editabel the layer using the  **Toggle Editing** button.

6. Go to the Label toolbar and click the  icon. Now you can shift the label manually to another position (see Abb. 16.33). The new position of the label is saved in the `xlabel` and `ylabel` columns of the attribute table.

7. It's also possible to add a line connecting each lake to its moved label using:

- the label's *callout property*
- or the *geometry generator symbol layer* with the expression below:

```
make_line( centroid( $geometry ), make_point( "xlabel", "ylabel" ) )
```

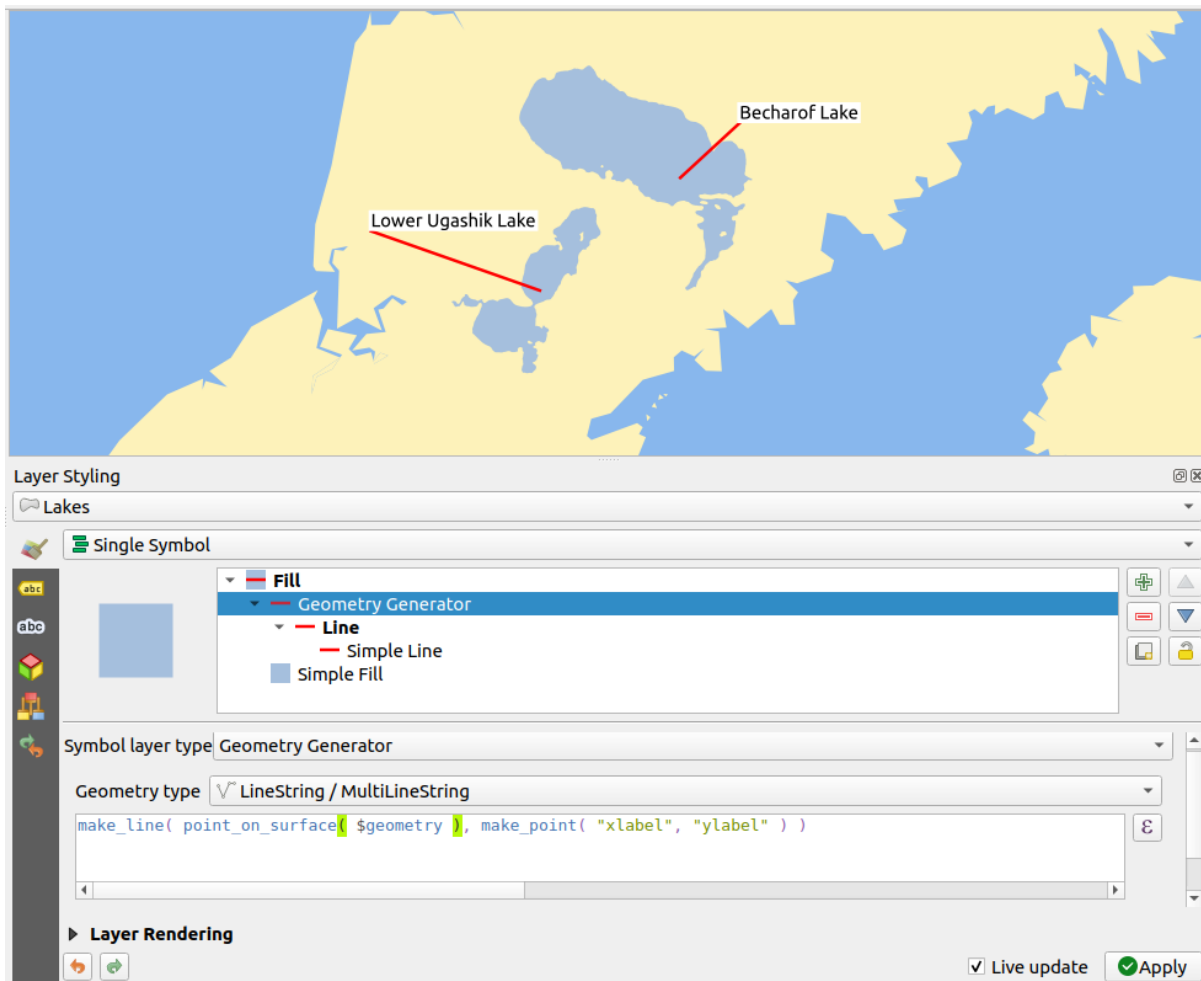







Abb. 16.33: Moved labels


**Bemerkung:** The *Auxiliary Storage Properties* mechanism may be used with data-defined properties without having an editable data source.

## 16.1.5 Diagrammeigenschaften

The  *Diagrams* tab allows you to add a graphic overlay to a vector layer (see [Abb. 16.34](#)).

Die aktuelle Kernimplementation von Diagrammen bietet Unterstützung von:

-  *No diagrams*: the default value with no diagram displayed over the features;
-  *Pie chart*, a circular statistical graphic divided into slices to illustrate numerical proportion. The arc length of each slice is proportional to the quantity it represents;
-  *Text diagram*, a horizontally divided circle showing statistics values inside;
-  *Histogram*, bars of varying colors for each attribute aligned next to each other
-  *Stacked bars*, Stacks bars of varying colors for each attribute on top of each other vertically or horizontally

In the top right corner of the *Diagrams* tab, the  Automated placement settings (applies to all layers) button provides means to control diagram *labels placement* on the map canvas.

---

### Tip: Switch quickly between types of diagrams

Given that the settings are almost common to the different types of diagram, when designing your diagram, you can easily change the diagram type and check which one is more appropriate to your data without any loss.

---

For each type of diagram, the properties are divided into several tabs:

- *Attributes*
- *Rendering*
- *Size*
- *Placement*
- *Options*
- *Legend*

### Attribute

*Attribute* definiert, welche Variablen in dem Diagramm gezeigt werden. Nutzen Sie den *:sup:Objekt hinzufügen* Knopf um das gewünschte Feld in das *„Zugeordnete Attribut“* Bedienfeld zu schieben. Erzeugte Attribute mit den *Ausdrücke* können auch genutzt werden.

You can move up and down any row with click and drag, sorting how attributes are displayed. You can also change the label in the *„Legend“* column or the attribute color by double-clicking the item.

This label is the default text displayed in the legend of the print layout or of the layer tree.

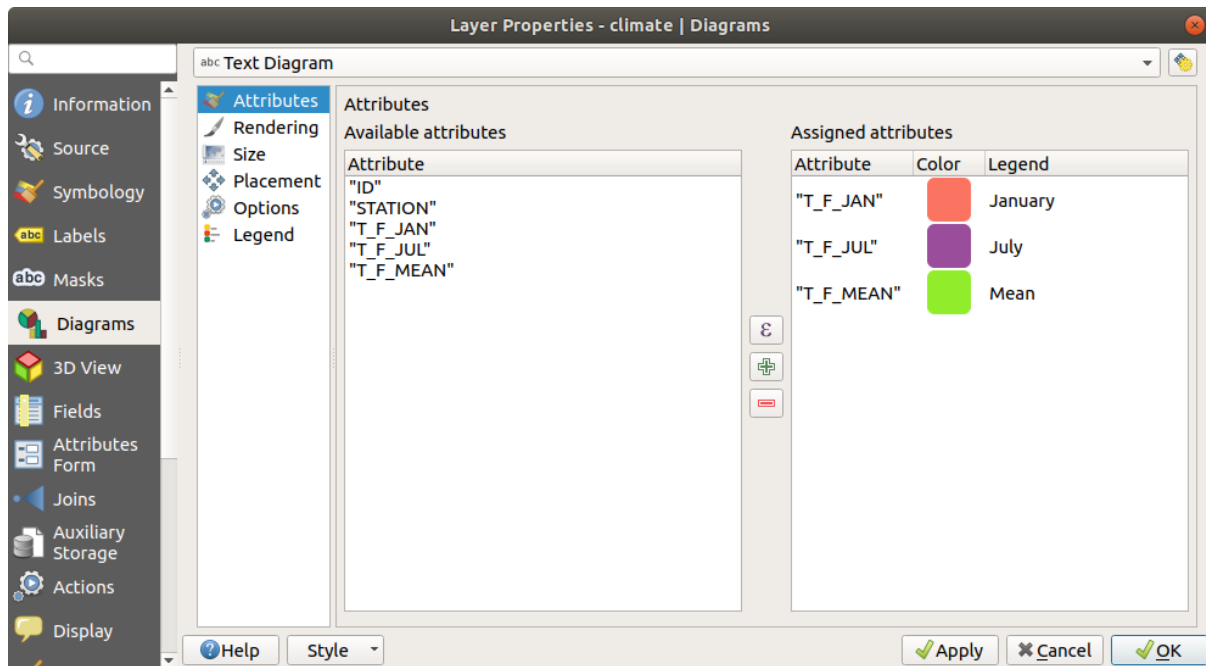


Abb. 16.34: Diagram properties - Attributes tab

## Layeranzeige kontrollieren

*Rendering* defines how the diagram looks like. It provides general settings that do not interfere with the statistic values such as:

- the graphic's opacity, its outline width and color;
- depending on the type of diagram:
  - for histogram and stacked bars, the width of the bar and the spacing between the bars. You may want to set the spacing to 0 for stacked bars. Moreover, the *Axis line symbol* can be made visible on the map canvas and customized using *line symbol properties*.
  - for text diagram, the circle background color and the *font* used for texts;
  - for pie charts, the *Start angle* of the first slice and their *Direction* (clockwise or not).
- the use of *paint effects* on the graphics.

In this tab, you can also manage and fine tune the diagram visibility with different options:

- *Diagram z-index*: controls how diagrams are drawn on top of each other and on top of labels. A diagram with a high index is drawn over diagrams and labels;
- ☒ *Show all diagrams*: shows all the diagrams even if they overlap each other;
- *Show diagram*: allows only specific diagrams to be rendered;
- *Always Show*: selects specific diagrams to always render, even when they overlap other diagrams or map labels;
- setting the *Scale dependent visibility*;

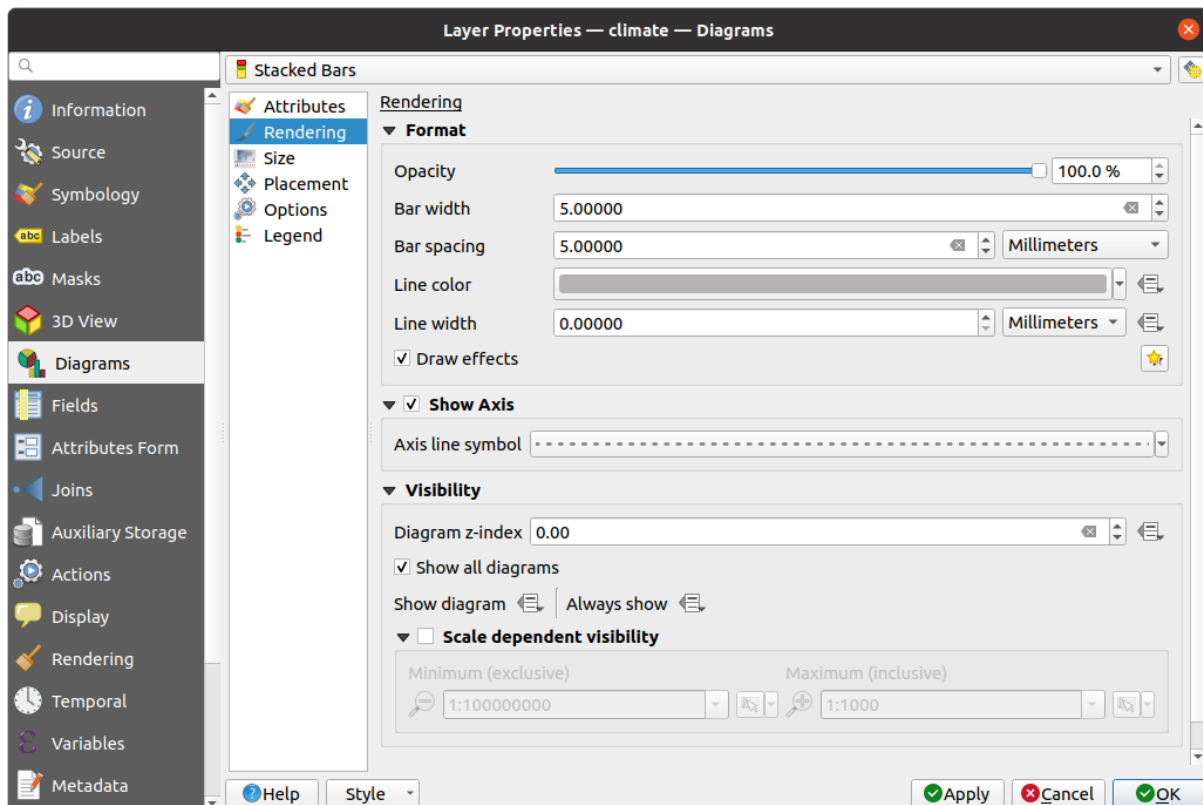


Abb. 16.35: Diagram properties - Rendering tab

## Größe

*Size* is the main tab to set how the selected statistics are represented. The diagram size *units* can be ‚Millimeters‘, ‚Points‘, ‚Pixels‘, ‚Map Units‘ or ‚Inches‘. You can use:

- *Fixed size*, a unique size to represent the graphic of all the features (not available for histograms)
- or *Scaled size*, based on an expression using layer attributes:
  1. In *Attribute*, select a field or build an expression
  2. Press *Find* to return the *Maximum value* of the attribute or enter a custom value in the widget.
  3. For histogram and stacked bars, enter a *Bar length* value, used to represent the *Maximum value* of the attributes. For each feature, the bar length will then be scaled linearly to keep this matching.
  4. For pie chart and text diagram, enter a *Size* value, used to represent the *Maximum value* of the attributes. For each feature, the circle area or diameter will then be scaled linearly to keep this matching (from 0). A *Minimum size* can however be set for small diagrams.



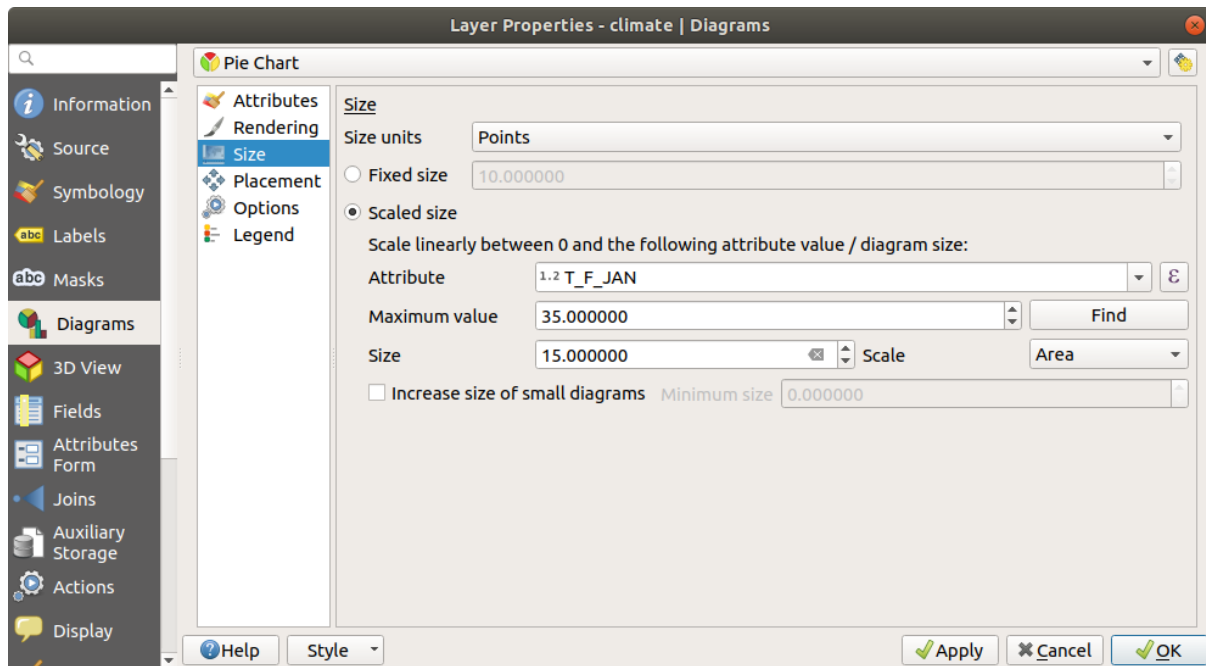


Abb. 16.36: Diagram properties - Size tab

## Platzierung

*Placement* defines the diagram position. Depending on the layer geometry type, it offers different options for the placement (more details at [Placement](#)):

- *Around point* or *Over point* for point geometry. The former variable requires a radius to follow.
- *Around line* or *Over line* for line geometry. Like point feature, the first variable requires a distance to respect and you can specify the diagram placement relative to the feature (‘above’, ‘on’ and/or ‘below’ the line) It’s possible to select several options at once. In that case, QGIS will look for the optimal position of the diagram. Remember that you can also use the line orientation for the position of the diagram.
- *Around centroid* (at a set *Distance*), *Over centroid*, *Using perimeter* and *Inside polygon* are the options for polygon features.

The *Coordinate* group provides direct control on diagram placement, on a feature-by-feature basis, using their attributes or an expression to set the X and Y coordinate. The information can also be filled using the [Move labels and diagrams](#) tool.

In the *Priority* section, you can define the placement priority rank of each diagram, ie if there are different diagrams or labels candidates for the same location, the item with the higher priority will be displayed and the others could be left out.

*Discourage diagrams and labels from covering features* defines features to use as *obstacles*, ie QGIS will try to not place diagrams nor labels over these features. The priority rank is then used to evaluate whether a diagram could be omitted due to a greater weighted obstacle feature.

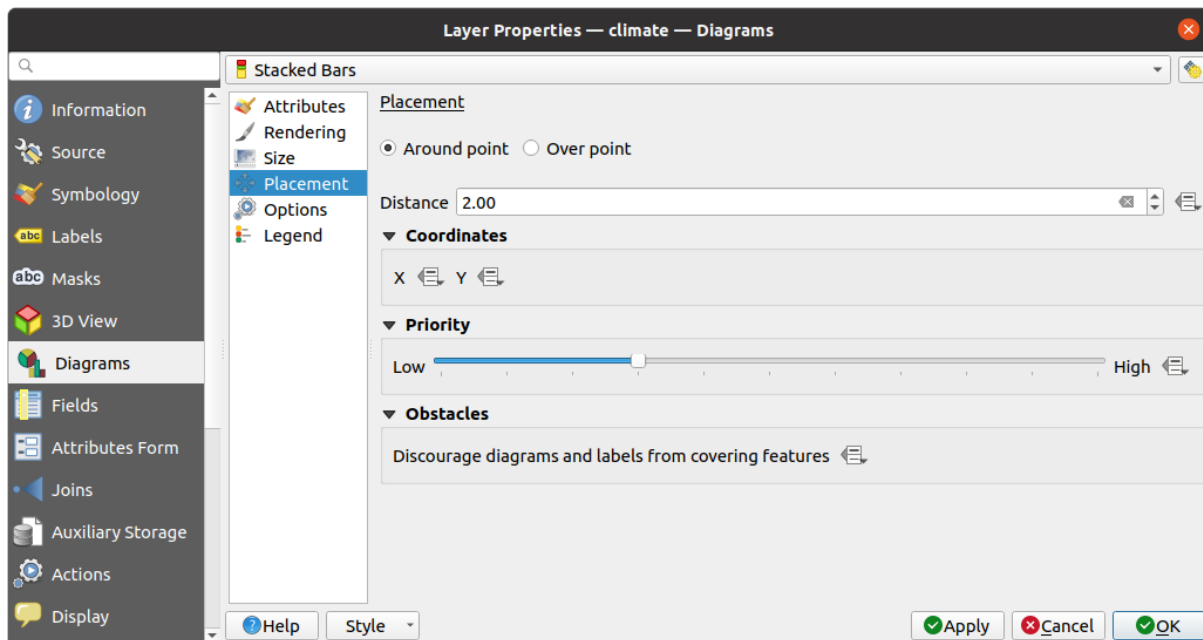


Abb. 16.37: Vector properties dialog with diagram properties, Placement tab

## Optionen

The *Options* tab has settings for histograms and stacked bars. You can choose whether the *Bar orientation* should be *Up*, *Down*, *Right* or *Left*, for horizontal and vertical diagrams.

## Legende

From the *Legend* tab, you can choose to display items of the diagram in the *Layers panel*, and in the *print layout legend*, next to the layer symbology:

- check *Show legend entries for diagram attributes* to display in the legends the *Color* and *Legend* properties, as previously assigned in the *Attributes* tab;
- and, when a *scaled size* is being used for the diagrams, push the *Legend Entries for Diagram Size...* button to configure the diagram symbol aspect in the legends. This opens the *Data-defined Size Legend* dialog whose options are described in *Data-defined size legend*.

When set, the diagram legend items (attributes with color and diagram size) are also displayed in the print layout legend, next to the layer symbology.

### 16.1.6 Eigenschaften Masken

The *Masks* tab helps you configure the current layer symbols overlay with other symbol layers or labels, from any layer. This is meant to improve the readability of symbols and labels whose colors are close and can be hard to decipher when overlapping; it adds a custom and transparent mask around the items to „hide“ parts of the symbol layers of the current layer.

To apply masks on the active layer, you first need to enable in the project either *mask symbol layers* or *mask labels*. Then, from the *Masks* tab, check:

- the *Masked symbol layers*: lists in a tree structure all the symbol layers of the current layer. There you can select the symbol layer item you would like to transparently „cut out“ when they overlap the selected mask sources
- the *Mask sources* tab: list all the mask labels and mask symbol layers defined in the project. Select the items that would generate the mask over the selected masked symbol layers

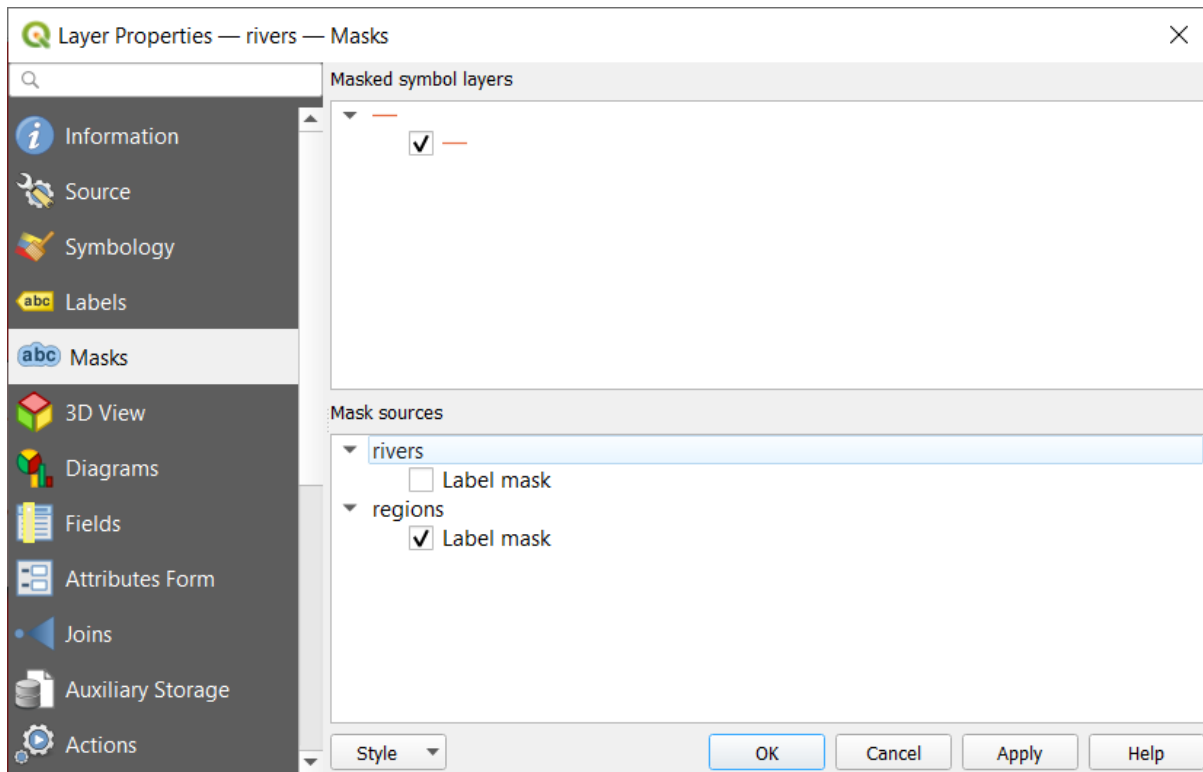



Abb. 16.38: Layer properties - Masks tab

### 16.1.7 3D-Ansicht Eigenschaften

The  **3D View** tab provides settings for vector layers that should be depicted in the *3D Map view* tool.

To display a layer in 3D, select from the combobox at the top of the tab, either:

- *Single symbol*: features are rendered using a common 3D symbol whose properties can be *data-defined* or not. Read details on [setting a 3D symbol](#) for each layer geometry type.
- *Rule-based*: multiple symbol configurations can be defined and applied selectively based on expression filters and scale range. More details on how-to at [Rule-based rendering](#).

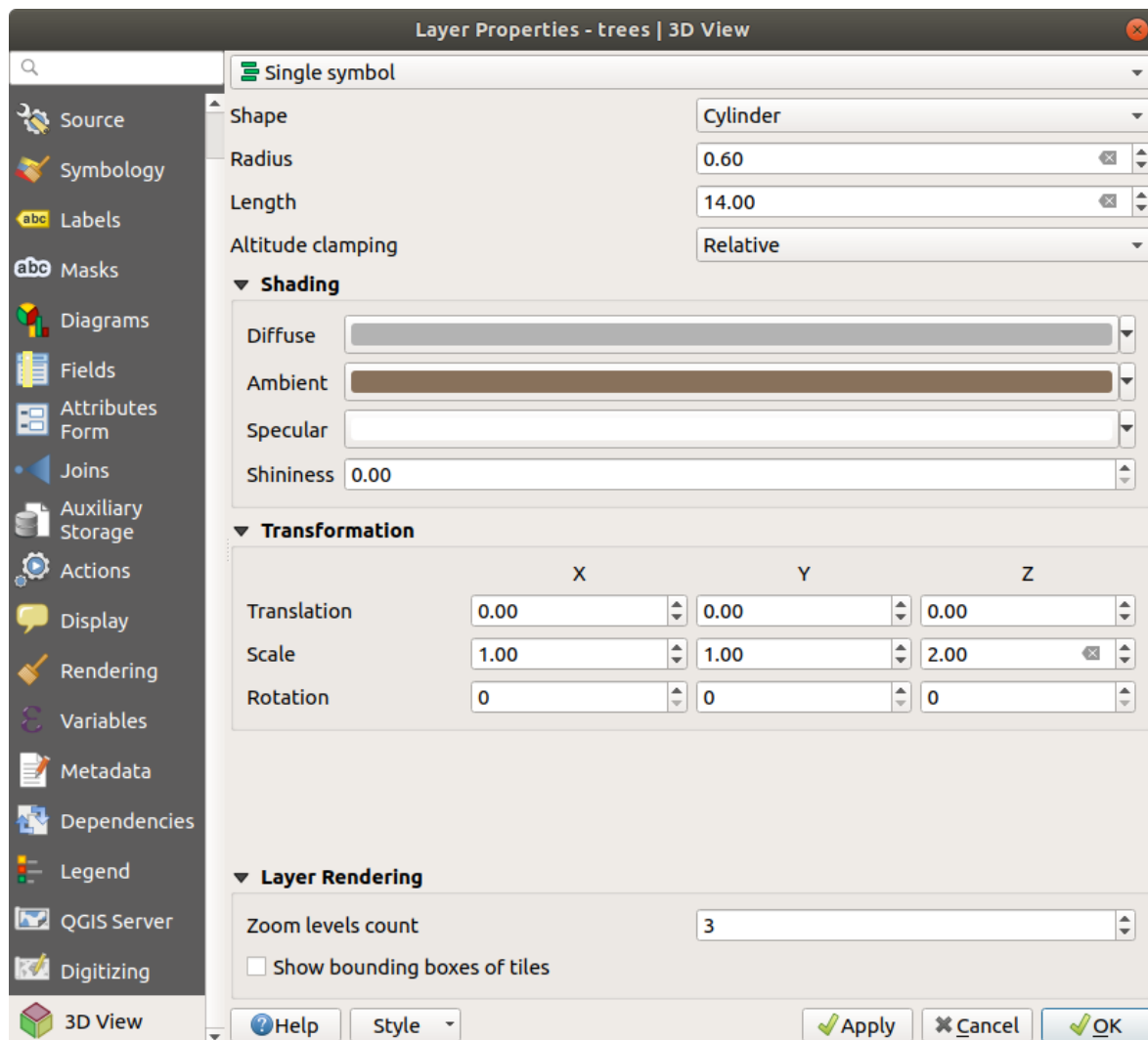


Abb. 16.39: 3D properties of a point layer


**Achtung: Prefer the *Elevation* tab for symbol elevation and terrain settings**




Features' elevation and altitude related properties (*Altitude clamping*, *Altitude binding*, *Extrusion* or *Height*) in the *3D View* tab inherit their default values from the layer's *Elevation properties* and should preferably be set from within the *Elevation* tab.


For better performance, data from vector layers are loaded in the background, using multithreading, and rendered in tiles whose size can be controlled from the *Layer rendering* section of the tab:

- *Zoom levels count*: determines how deep the quadtree will be. For example, one zoom level means there will be a single tile for the whole layer. Three zoom levels means there will be 16 tiles at the leaf level (every extra zoom level multiplies that by 4). The default is 3 and the maximum is 8.
- ☒ *Show bounding boxes of tiles*: especially useful if there are issues with tiles not showing up when they should.

## 16.1.8 Eigenschaften Felder

The  *Fields* tab provides information on fields related to the layer and helps you organize them.

The layer can be made *editable* using the  *Toggle editing mode*. At this moment, you can modify its structure using the  *New field* and  *Delete field* buttons.

When creating  *New field*, the *Comment* option is available only for data sources that allow editing comments (See *Datenbankeinträge* for more details). You can also set aliases within *Add Field* dialog, for supported OGR formats (GeoPackage and ESRI File Geodatabase).

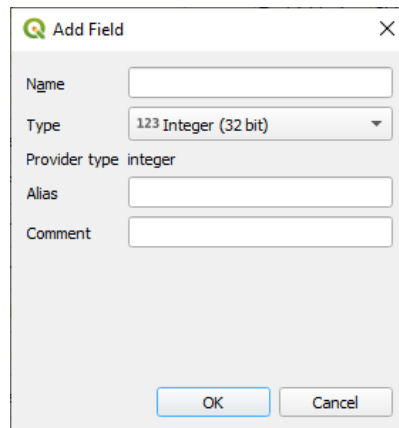


Abb. 16.40: Add Field Dialog

You can also rename fields by double-clicking its name. This is only supported for data providers like PostgreSQL, Oracle, Memory layer and some GDAL layers depending on the GDAL version.

If set in the underlying data source or in the *forms properties*, the field's alias is also displayed. An alias is a human readable field name you can use in the feature form or the attribute table. Aliases are saved in the project file.

Other than the fields contained in the dataset, *virtual fields* and *Auxiliary Storage* included, the *Fields* tab also lists fields from any *joined layers*. Depending on the origin of the field, a different background color is applied to it.

For each listed field, the dialog also lists read-only characteristics such as its *Type*, *Type name*, *Length* and *Precision*.

Depending on the data provider, you can associate a comment with a field, for example at its creation. This information is retrieved and shown in the *Comment* column and is later displayed when hovering over the field label in a feature form.

Under the *Configuration* column, you can set how the field should behave in certain circumstances:

- *Not searchable*: check this option if you do not want this field to be queried by the *search locator bar*
- *Do not expose via WMS*: check this option if you do not want to display this field if the layer is served as WMS from QGIS server
- *Do not expose via WFS*: check this option if you do not want to display this field if the layer is served as WFS from QGIS server

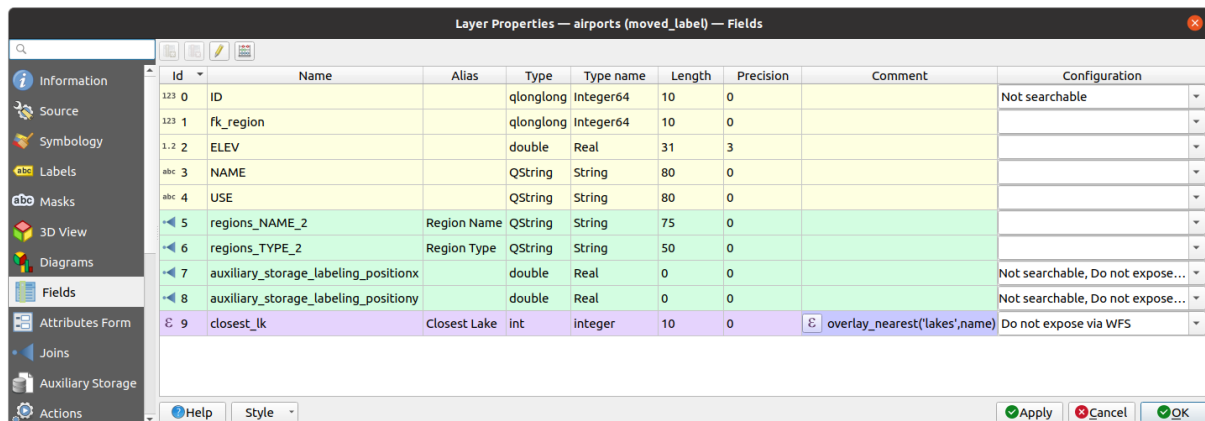


Abb. 16.41: Fields properties tab

## 16.1.9 Eigenschaften Attributformular

The *Attributes Form* tab helps you set up the form to display when creating new features or querying existing one. You can define:

- Aussehen und Funktionsweise für jedes Feld im Eingabeformular oder in der Attributtabelle (Beschriftung, Funktion, Beschränkungen ...);
- die Struktur des Formulars (benutzerdefiniert oder automatisch generiert);
- zusätzliche Logik in Python für die Interaktion mit dem Formular oder den Feldelementen.

Oben rechts im Dialog können Sie einstellen, ob das Formular bei der Erstellung neuer Features standardmäßig geöffnet wird. Dies kann pro Ebene oder global mit der Option *Attributformular-Popup nach Feature-Erstellung unterdrücken* im Menü *Einstellungen ► Optionen ► Digitalisieren* konfiguriert werden.

### Customizing a form for your data

By default, when you click on a feature with the *Identify Features* tool or switch the attribute table to the *form view* mode, QGIS displays a basic form with predefined widgets (generally spinboxes and textboxes — each field is represented on a dedicated row by its label next to the widget). If *relations* are set on the layer, fields from the referencing layers are shown in an embedded frame at the bottom of the form, following the same basic structure.

This rendering is the result of the default *Autogenerate* value of the *Attribute editor layout* setting in the *Layer properties ► Attributes Form* tab. This property holds three different values:

- *Autogenerate*: keeps the basic structure of „one row - one field“ for the form but allows to customize each corresponding widget.
- *Drag-and-drop designer*: other than widget customization, the form structure can be made more complex eg, with widgets embedded in groups and tabs.
- *Provide ui file*: allows to use a Qt designer file, hence a potentially more complex and fully featured template, as feature form.

## The autogenerated form




When the `Autogenerate` option is on, the *Available widgets* panel shows lists of fields (from the layer and its relations) that would be shown in the form. Select a field and you can configure its appearance and behavior in the right panel:

- adding *custom label and automated checks* to the field;
- setting a *particular widget* to use.

## Der Drag-and-drop-Designer

The drag and drop designer allows you to create a form with several containers (tabs or groups) to present the attribute fields or other widgets that are not directly linked to a particular field (like the HTML/QML widgets or the *actions* defined for the layer), as shown for example in Abb. 16.42.

Abb. 16.42: Das Ergebnis eines integrierten Formulars mit Reitern und benannten Gruppen

1. Choose `Drag and drop designer` from the *Select attribute layout editor* combobox. This enables the *Form Layout* panel next to the *Available widgets* panel, filled with existing fields. The selected field displays its *properties* in a third panel.
2. Select fields you do not want to use in your *Form Layout* panel and hit the  button to remove them. You can also toggle the selection with the  `Invert selection` button.
3. Drag and drop fields from the first panel to the `:guilabel'Form Layout'` one to re-add them. The same field can be added multiple times.
4. Drag and drop fields within the *Form Layout* panel to reorder their position.
5. Add containers to associate fields that belong to the same category and better structure the form.
  1. The first step is to use the  `Add a new tab or group to the form layout` icon. Fields and other groups will be displayed in it.

2. Then set the properties of the container, ie:

- the *Label*: the title that will be used for the container
- the *Container Type*: it can be a *Tab*, *Group box in container* (a collapsible group box inside a tab or another group) or a *Row* (a container type that allows you to arrange your widgets in a horizontal row, automatically determining the number of columns based on the number of widgets),
- the *Within*: this optional feature allows you to select an existing container in which the new container (*Group box in container* or *Row*) will be embedded.
- and the *Number of columns* the embedded fields should be distributed over

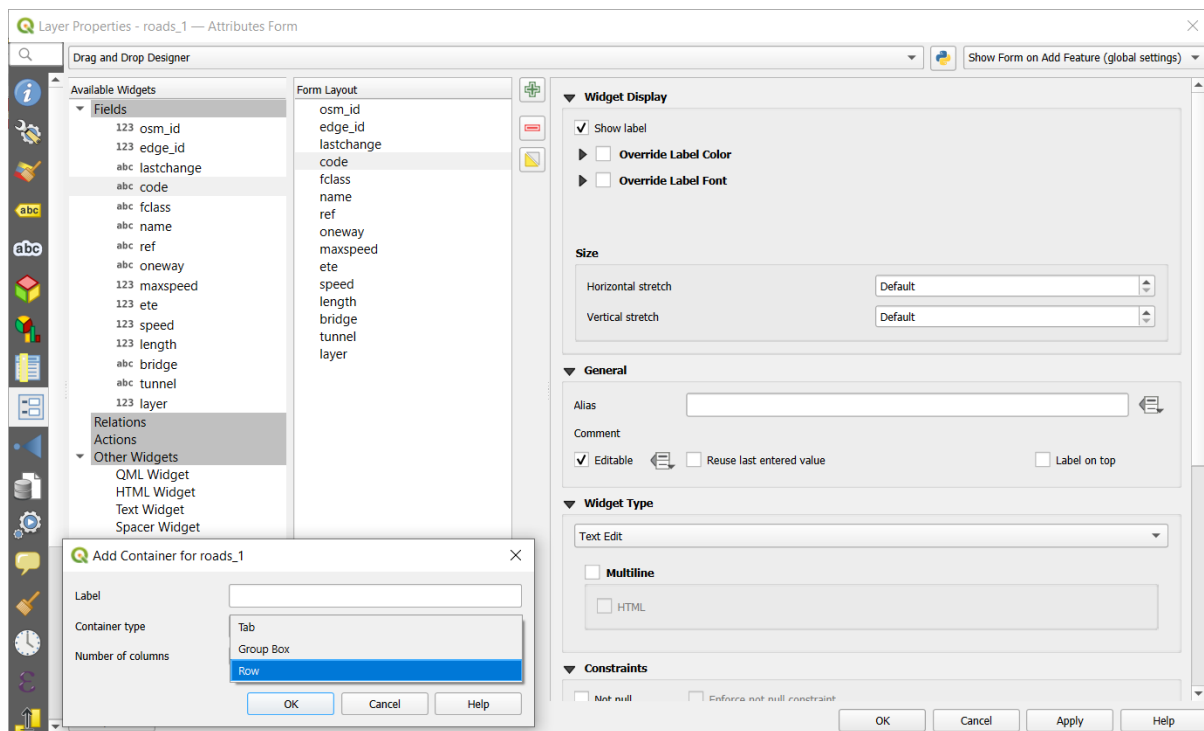



Abb. 16.43: Dialog to create containers with the **Attribute editor layout**


These, and other properties can later be updated by selecting the item and, from the third panel:

- hide or show the container's label
- rename the container
- set the number of columns
- enter an expression to control the container's visibility. The expression will be re-evaluated every time values in the form change, and the tab or group box shown/hidden accordingly
-  *Show as Group Box*: converts a tab into a top-level group box and vice versa
- in case of a group box, set whether at the form opening, it should show as *Collapsed* for every features, or only for those matching an expression (*Control collapsed by expression*).
- configure the *Style* look of the container, with custom *Background color*, label color and font properties

3. You can create and embed as many containers as you want by pressing again the  Add a new tab or group to the form layout icon.

6. The next step is to assign the relevant fields to each container, by simple drag and drop. Group boxes and tabs can also be moved the same way.




7. *Customize the widget* of the fields in use
8. In case the layer is involved in a *one or many to many relation*, drag-and-drop the relation name from the *Available Widgets* panel to the *Form Layout* panel. The associated layer attribute form will be embedded at the chosen place in the current layer's form. As for the other items, select the relation label to configure some properties:
  - hide or show the relation label
  - show the link button
  - show the unlink button
9. In case the layer has one or more *actions* enabled for *Layer* or *Feature* scope, the actions will be listed under *Actions* and you can drag and drop them as with the other fields. The associated action will be embedded at the chosen place in the current layer's form.
10. Further customize the form by adding one or more widgets from *Other Widgets* (see *Andere Widgets*)
11. Apply the layer's properties dialog
12. Open a feature attribute form (eg, using the  Identify features tool) and it should display the new form.

## Andere Widgets

The drag and drop designer offers a number of widgets that are not connected to a particular field. They can be used to enhance the appearance of the form or to display dynamically calculated values.


- *HTML Widget*: embeds an HTML page, the HTML source may contain the result of dynamically calculated expressions.

HTML widgets can be used for example to display images stored as BLOB in a field (let's call it `photo`):

1. In the Drag-and-drop designer mode, add a *HTML Widget* to your *Form Layout*.
2. Double-click on the *HTML Widget* to configure it.
3. Change the default *Title* or hide it.
4. Press the  button and enter the following QGIS expression:

```
'<img src= "data:image/png;base64,' || to_base64("photo") || '>'
```

Ensure that you replace *photo* with your own BLOB field name. The above expression creates a string with HTML image tag in which the BLOB file is encoded.

5. Apply the dialog and then press the  button.
6. QGIS automatically applies HTML formatting and functions to evaluate your expression, resulting in following code:

```
<script>document.write(expression.evaluate("<img src=\"data:image/png;
base64,' || to_base64(\"photo\") || '\">\""));</script>
```

A preview of your image is displayed on the right.

- *QML Widget*: embeds a QML page, the QML source may contain the result of dynamically calculated expressions.
- *Text Widget*: displays a text widget which supports basic HTML markup and may contain the result of dynamically calculated expressions.
- *Spacer Widget*: inserts an empty transparent rectangle, increasing the vertical distance between two widgets.

---

### Tipp: Display Dynamic Content

The widgets mentioned above (except the *Spacer Widget*) support expressions that can be used to display content that dynamically changes whenever another field in the form changes. This can be achieved using the `current_value('field_name')` function in the expression to examine the value of another field.

---

### Using custom ui-file

The `Provide ui-file` option allows you to use complex dialogs made with Qt-Designer. Using a UI-file allows a great deal of freedom in creating a dialog. Note that, in order to link the graphical objects (textbox, combobox...) to the layer's fields, you need to give them the same name.

Use the *Edit UI* to define the path to the file to use.

UI-files can also be hosted on a remote server. In this case, you provide the URL of the form instead of the file path in *Edit UI*.

You'll find some example in the *Creating a new form* lesson of the *QGIS-training-manual-index-reference*. For more advanced information, see <https://woostuff.wordpress.com/2011/09/05/qgis-tips-custom-feature-forms-with-python-logic/>.

### Enhance your form with custom functions

QGIS forms can have a Python function that is called when the dialog is opened. Use this function to add extra logic to your dialogs. The form code can be specified in three different ways:

- `load from the environment`: use a function, for example in `startup.py` or from an installed plugin
- `load from an external file`: a file chooser will let you select a Python file from your filesystem or enter a URL for a remote file.
- `provide code in this dialog`: a Python editor will appear where you can directly type the function to use.

In all cases you must enter the name of the function that will be called (`open` in the example below).

Ein Beispiel ist (im Modul `MyForms.py`):

```
def open(dialog, layer, feature):  
    geom = feature.geometry()  
    control = dialog.findChild(QWidget, "My line edit")
```

Reference in Python Init Function like so: `open`

### Einstellung der Feldfunktionsweisen

Der Hauptteil der Registerkarte *Attributformular* hilft Ihnen dabei festzulegen, wie das Ausfüllen oder Anzeigen der Werte des Feldes in der Attributtabelle oder im Feature-Formular ausgeführt wird: Sie können festlegen, wie der Benutzer mit jedem Feld interagiert und welche Werte oder Wertebereiche jeweils hinzugefügt werden dürfen.

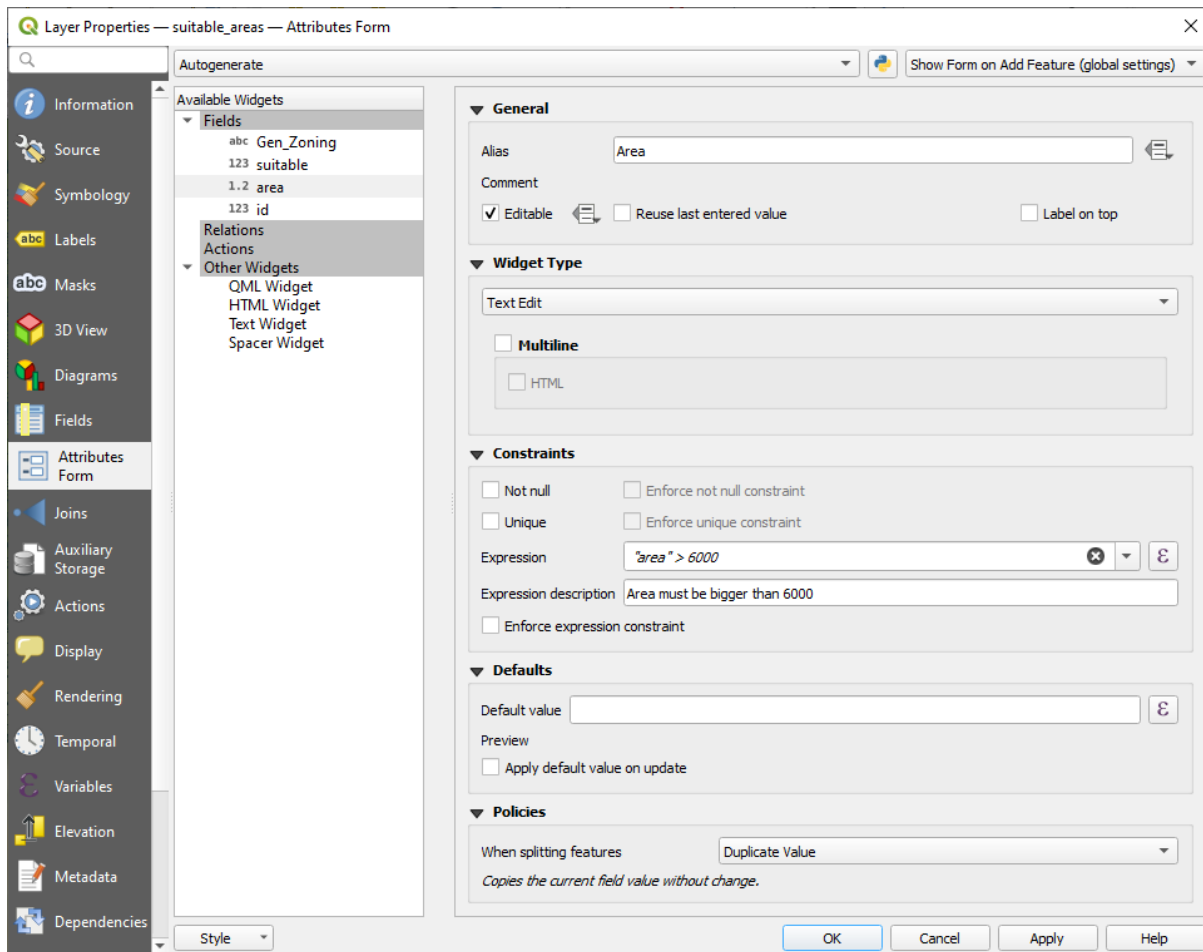


Abb. 16.44: Dialog um ein Bearbeitungselement für eine Attributspalte auszuwählen

## Allgemeine Einstellungen





Regardless the type of widget applied to the field, there are some common properties you can set to control whether and how a field can be edited.

### Widget display

Only available for the *Drag and drop* designer mode, this group helps you configure the look of the widget assigned to the field:



- *Show label*: indicates whether the field name should be displayed in the form
- *Override label color*: applies specific color to the field's label
- *Override label font*: applies specific font properties (bold, italic, underline, strikethrough, or font family) to the field's label

## Allgemeine Optionen


- *Alias*: a human readable name to use for fields. The alias will be displayed in the feature form, the attribute table, or in the *Identify results* panel. It can also be used as field name replacement in the *expression builder*, easing expressions understanding and reviews. Aliases are saved in project file.
- *Comment*: displays the field's comment as shown in the *Fields* tab, in a read-only state. This information is shown as tooltip when hovering over the field label in a feature form.
-  *Editable*: uncheck this option to set the field read-only (not manually modifiable) even when the layer is in edit mode. Note that checking this setting doesn't override any edit limitation from the provider. This option can be controlled by a data-defined property thanks to the  Data-defined override button.
-  *Reuse last entered value*: remembers the last value entered in this field and uses it as default for the next feature being edited in the layer.
-  *Label on top*: places the field name above or beside the widget in the feature form.

## Beschränkungen

You can constrain the value to insert in the field. This constraint can be:

-  *Not null*: requires the user to provide a value;
-  *Unique*: guarantee the inserted value to be unique throughout the field;
- based on a custom *expression*: e.g. `not regexp_match(col0, '^[A-Za-z]')` will ensure that the value of the field *col0* has only alphabet letters. A short description can be added to help you remember the constraint.

Whenever a value is added or edited in a field, it's submitted to the existing constraints and:

- if it meets all the requirements, a green check is shown beside the field in the form;
- if it does not meet all the requirements, then the field is colored in yellow or orange and a corresponding cross is displayed next to the widget. You can hover over the cross to remind which constraints are applied to the field and fix the value:
  - A yellow cross appears when the unmet constraint is an unenforced one (*soft constraint*) and it does not prevent you to save the changes with the „wrong“ values;
  - An orange cross can not be ignored and does not allow you to save your modifications until they meet the constraints. It appears when the  *Enforce constraint* option is checked (*hard constraint*).

## Vorgabewerte


- *Default value*: for new features, automatically populates by default the field with a predefined value or an *expression-based one*. For example, you can:
  - use `$x`, `$length`, `$area` to automatically populate a field with the feature's X coordinate, length, area or any geometric information at its creation;
  - increment a field by 1 for each new feature using `maximum("field")+1`;
  - save the feature creation datetime using `now()`;
  - use *variables* in expressions, making it easier to e.g. insert the operator name (`@user_full_name`), the project file path (`@project_path`), ...

A preview of the resulting default value is displayed at the bottom of the widget.

---

**Bemerkung:** The `Default value` option is not aware of the values in any other field of the feature being created so it won't be possible to use an expression combining any of those values i.e using an expression like `concat (field1, field2)` may not work.

---

-  *Apply default value on update:* whenever the feature attribute or geometry is changed, the default value is recalculated. This could be handy to save values like last user that modifies data, last time it was changed...

## Policies

*Policies* allows you to determine how values are assigned to the field when *Splitting features*:

- *Duplicate Values:* Keeps the existing value of the field for the resulting split features.
- *Use Default Value:* Resets the field by recalculating its *default value*. If no default value clause exists, the existing value is kept for the resulting split features.
- *Remove Value:* Clears the field to an unset state.
- *Use Ratio Geometries:* Recalculates the field value for all split portions by multiplying the existing value by ratio of the split parts lengths or areas.

## Funktionsauswahl

Based on the field type, QGIS automatically determines and assigns a default widget type to it. You can then replace the widget with any other compatible with the field type. The available widgets are:

- **Binary (BLOB):** Available only for binary fields, it displays by default a label with the size of the embedded data, if not empty. A drop-down button next to the label allows to:
  - *Embed file*, replacing or filling the field
  - *Clear contents*, removing any data in the field
  - *Save contents to file*, exporting the data as a file on disk

It is also possible to preview the embedded binary file in the field, if combined in a drag-and-drop form with e.g. a *QML or HTML widget*.

- **Checkbox:** Displays a checkbox whose state defines the value to insert.
- **Classification:** Only available when a *categorized symbology* is applied to the layer, displays a combo box with the values of the classes.
- **Color:** Displays a *color widget* allowing to select a color; the color value is stored as a html notation in the attribute table.
- **Datum/Zeit:** Stellt ein Linienfeld dar, das ein Kalender-Widget zum Öffnen eines Datums, einer Zeit oder beidem, darstellt. Der Spaltentyp muss Text sein. Sie können ein benutzerdefiniertes Format auswählen, einen Pop-up-Kalender, etc.
- **Enumeration:** Opens a combo box with predefined values fetched from the database. This is currently only supported by the PostgreSQL provider, for fields of `enum` type.
- **Attachment:** Uses a „Open file“ dialog to store file path in a relative or absolute mode. It can be used to display a hyperlink (to document path), a picture or a web page. User can also configure an *external storage system* to fetch/store resources.

---

**Tipp:** Relative Path in Attachment widget


If the path which is selected with the file browser is located in the same directory as the `.qgs` project file or below, paths are converted to relative paths. This increases portability of a `.qgs` project with multimedia information attached.

---

- **Versteckt:** Ein verstecktes Attribut ist unsichtbar. Der Anwender kann den Inhalt nicht sehen.
- **Key/Value:** Displays a two-columns table to store sets of key/value pairs within a single field. This is currently supported by the PostgreSQL provider, for fields of `hstore` type.
- **JSON View:** Displays JSON data in a syntax highlighted text edit or in tree view. This widget is currently read only. Several options are available to change how the data is displayed. ‚Default view‘ specify if the widget should appear in Text or Tree mode. ‚Format JSON‘ has three options which are related to the tree view only:
  - Indented: Display data in a human readable form with newlines and four space characters for indentation.
  - Compact: Display data in a one-line size optimized string without newlines or spaces.
  - Disabled: Display data as it comes from the provider.
- **List:** Displays a single column table to add different values within a single field. This is currently supported by the PostgreSQL provider, for fields of `array` type.
- **Bereich:** Erlaubt Ihnen numerische Werte eines bestimmten Wertebereichs festzulegen. Das Bearbeitungselement kann entweder ein Schieber oder ein Drehfeld sein.
- **Relation Reference:** This is the default widget assigned to the referencing field (i.e., the foreign key in the child layer) when a *relation* is set. It provides direct access to the parent feature’s form which in turn embeds the list and form of its children.
- **Text Edit** (default): This opens a text edit field that allows simple text or multiple lines to be used. If you choose multiple lines you can also choose html content.
- **Unique Values:** You can select one of the values already used in the attribute table. If ‚Editable‘ is activated, a line edit is shown with autocompletion support, otherwise a combo box is used.
- **Uuid Generator:** Generates a read-only UUID (Universally Unique Identifiers) field, if empty.
- **Value Map:** A combo box with predefined items. The value is stored in the attribute, the description is shown in the combo box. You can define values manually or load them from a layer or a CSV file.
- **Value Relation:** Offers values from a related table in a combobox. You can select layer, key column and value column. Several options are available to change the standard behaviors: allow null value, order by value, allow multiple selections and use of auto-completer. The forms will display either a drop-down list or a line edit field when completer checkbox is enabled.

If a layer that is stored in PostgreSQL, GeoPackage or SpatiaLite is configured to use a value relation widget, but the required layer is not already loaded into the project, QGIS will automatically search for the layer in the same database/connection.

### 16.1.10 Eigenschaften Verknüpfungen

The  *Joins* tab allows you to associate features of the current layer to features from another loaded vector layer (or table). The join is based on an attribute that is shared by the layers, in a one-to-one relationship. For more details on joins, please read *Joining features between two layers*.

### 16.1.11 Auxiliary Storage Properties

The regular way to customize styling and labeling is to use data-defined properties as described in *Datendefinierte Übersteuerung Setup*. However, it may not be possible if the underlying data is read only. Moreover, configuring these data-defined properties may be very time consuming or not desirable! For example, if you want to fully use map tools coming with *The Label Toolbar*, then you need to add and configure more than 20 fields in your original data source (X and Y positions, rotation angle, font style, color and so on).

The Auxiliary Storage mechanism provides the solution to these limitations and awkward configurations. Auxiliary fields are a roundabout way to automatically manage and store these data-defined properties (labels, diagram, symbology...) in a SQLite database thanks to editable joins. This allows you to store properties for layers that aren't editable.

A tab is available in vector layer properties dialog to manage auxiliary storage:

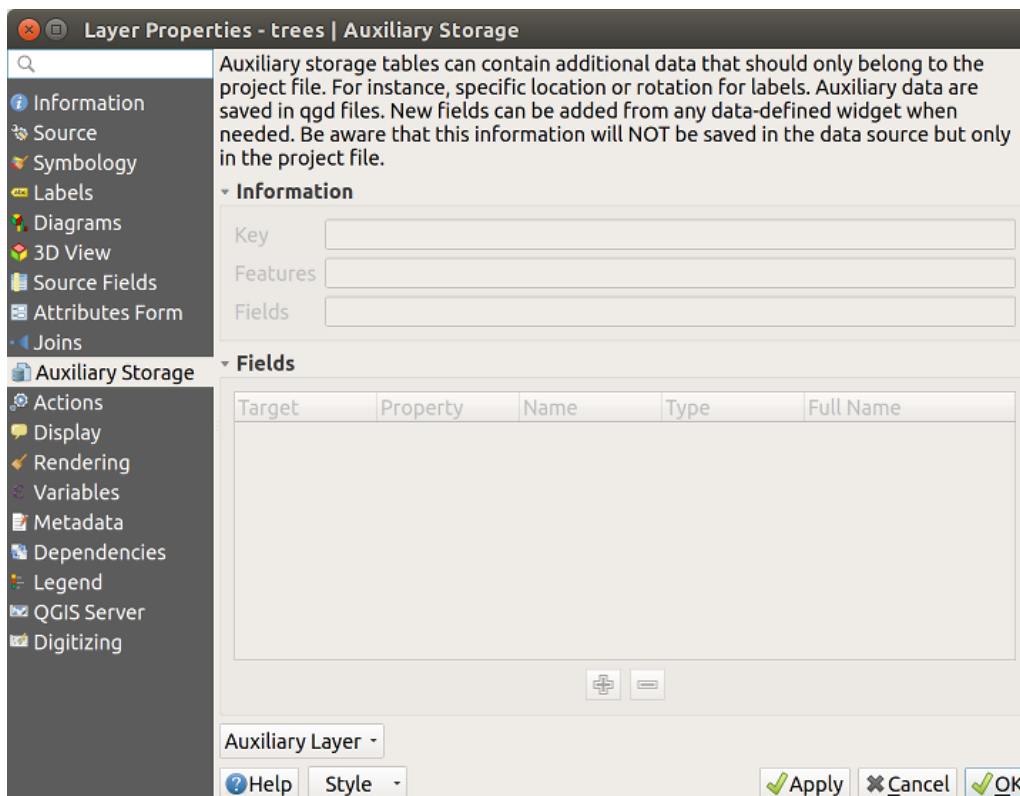


Abb. 16.45: Auxiliary Storage tab

### Beschriftung

Considering that the data source may be customized thanks to data-defined properties without being editable, labeling map tools described in *The Label Toolbar* are always available as soon as labeling is activated.

Actually, the auxiliary storage system needs an auxiliary layer to store these properties in a SQLite database (see *Auxiliary storage database*). Its creation process is run the first time you click on the map while a labeling map tool is currently activated. Then, a window is displayed, allowing you to select the primary key to use for joining (to ensure that features are uniquely identified):

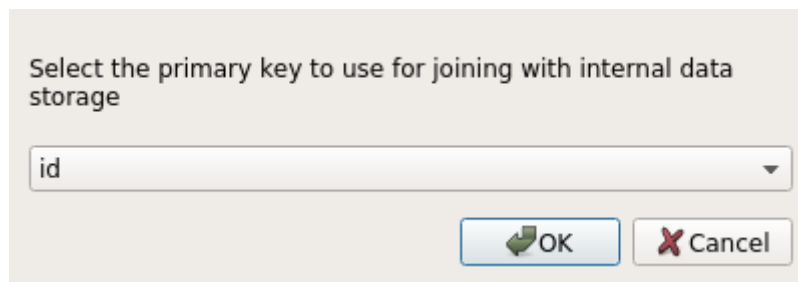


Abb. 16.46: Auxiliary Layer creation dialog

As soon as an auxiliary layer is configured for the current data source, you can retrieve its information in the tab:

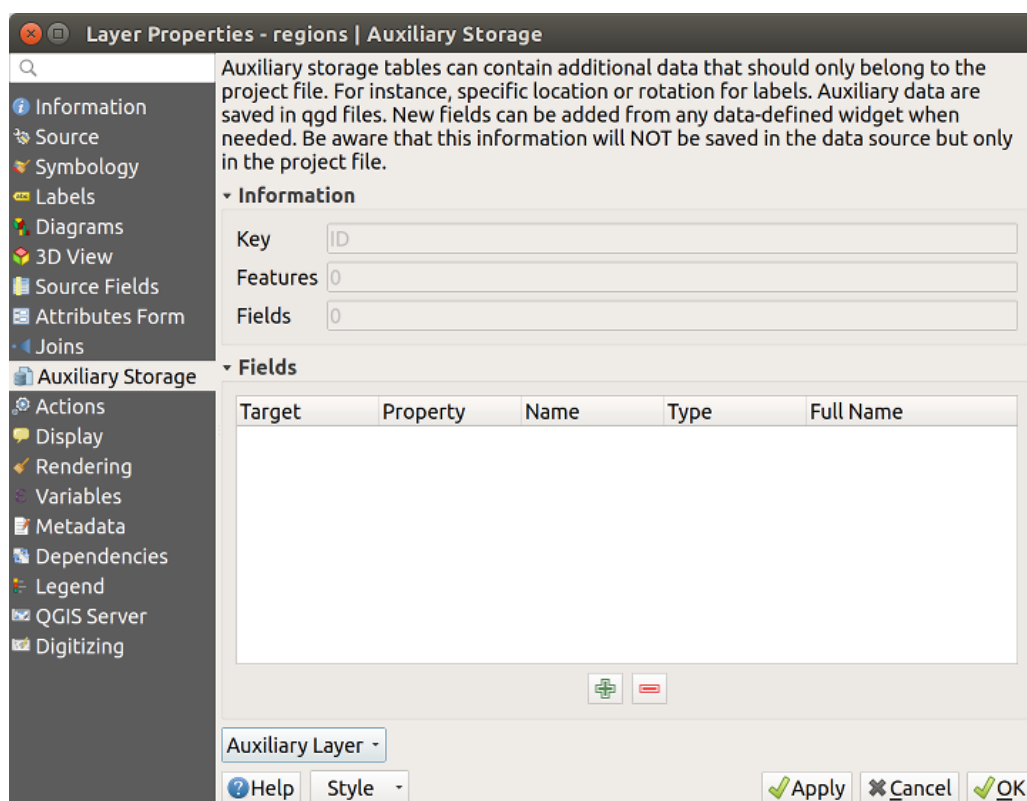



Abb. 16.47: Auxiliary Layer key

The auxiliary layer now has these characteristics:

- the primary key is ID,
- there are 0 features using an auxiliary field,
- there are 0 auxiliary fields.

Now that the auxiliary layer is created, you can edit the layer labels. Click on a label while the  Change Label map tool is activated, then you can update styling properties like sizes, colors, and so on. The corresponding data-defined properties are created and can be retrieved:



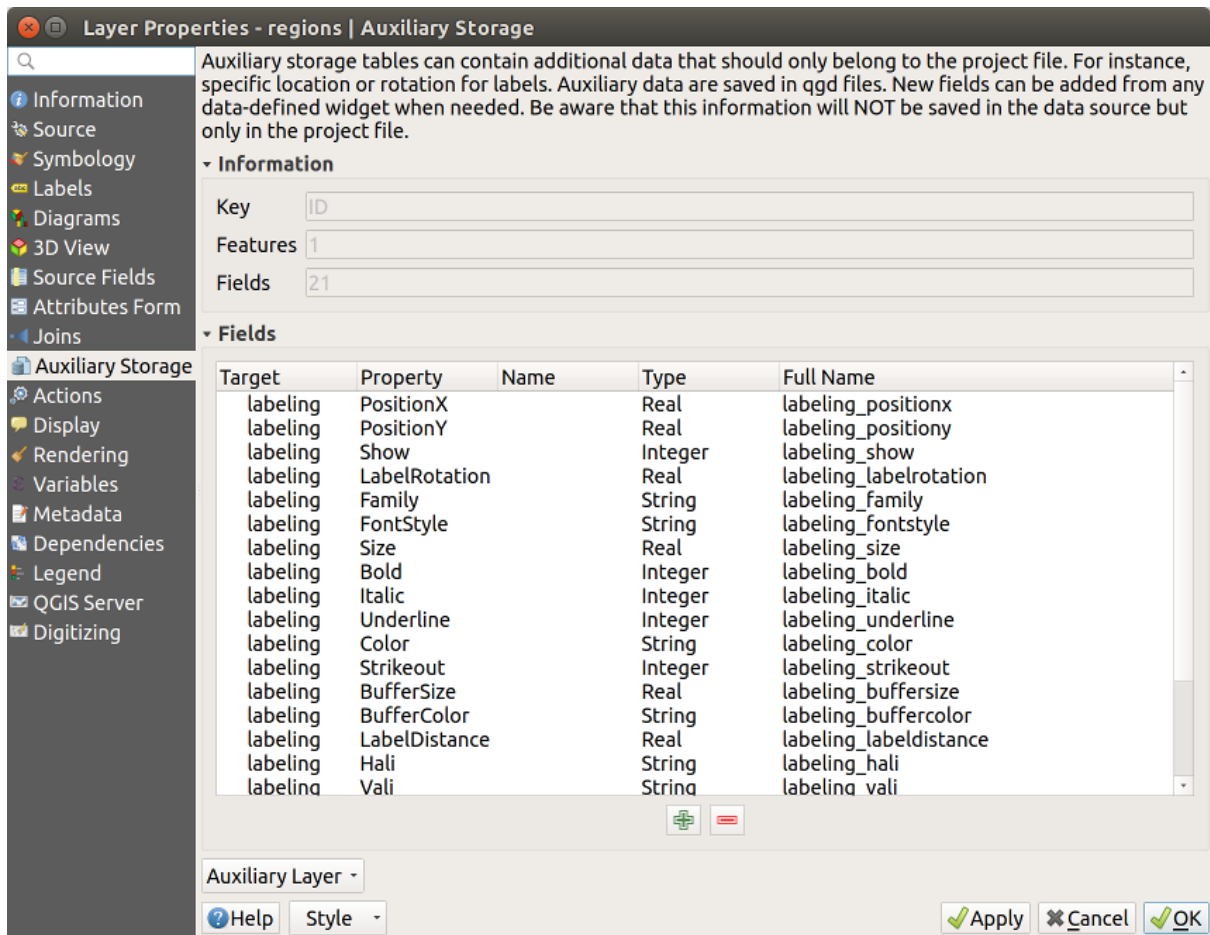



Abb. 16.48: Auxiliary Fields

As you can see in the figure above, 21 fields are automatically created and configured for labeling. For example, the `FontStyle` auxiliary field type is a `String` and is named `labeling_fontstyle` in the underlying SQLite database. There is also 1 feature which is currently using these auxiliary fields.

Notice that the icon  is displayed in the *Labels* properties tab indicating that the data-defined override options are set correctly:

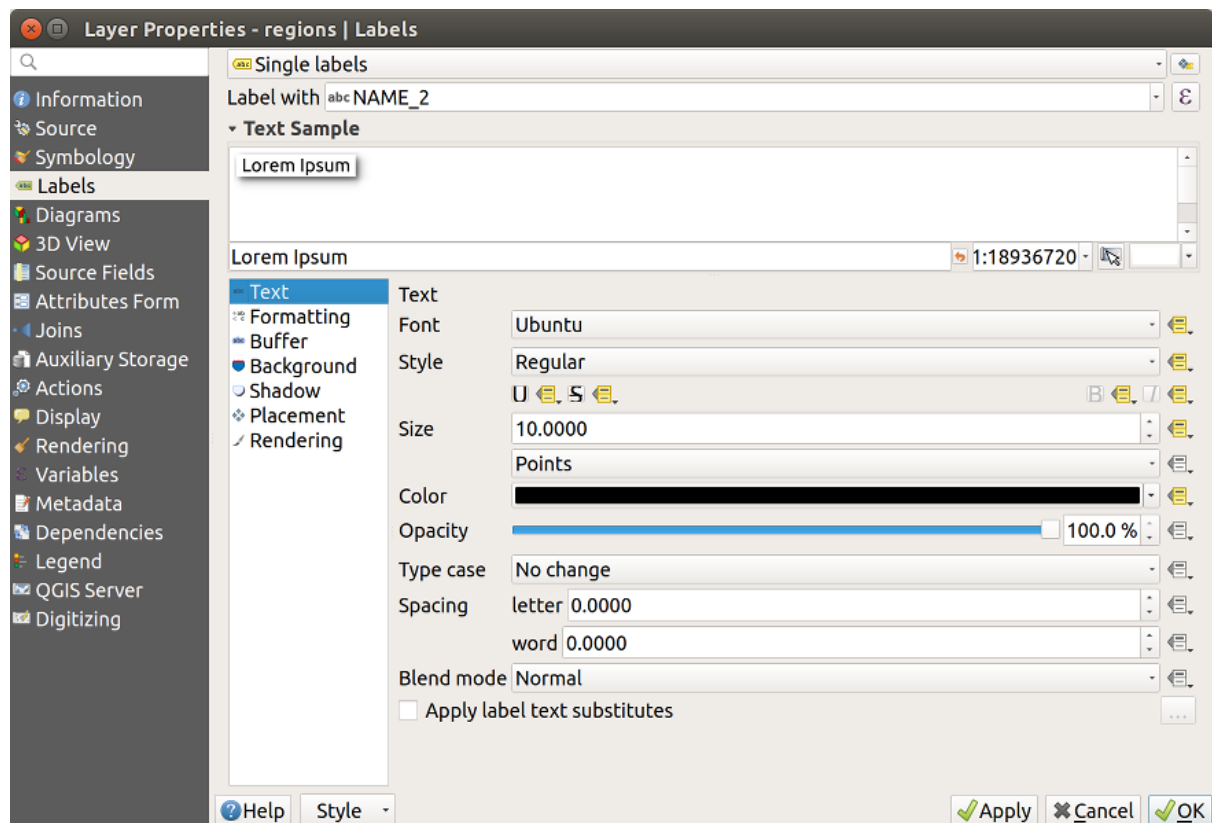




Abb. 16.49: Data-defined properties automatically created

Otherwise, there's another way to create an auxiliary field for a specific property thanks to the  Data-defined override button. By clicking on *Store data in the project*, an auxiliary field is automatically created for the *Opacity* field. If you click on this button and the auxiliary layer is not created yet, a window (Abb. 16.46) is first displayed to select the primary key to use for joining.

## Symbolisierung

Like the method described above for customizing labels, auxiliary fields can also be used to stylize symbols and diagrams. To do this, click on  Data-defined override and select *Store data in the project* for a specific property. For example, the *Fill color* field:

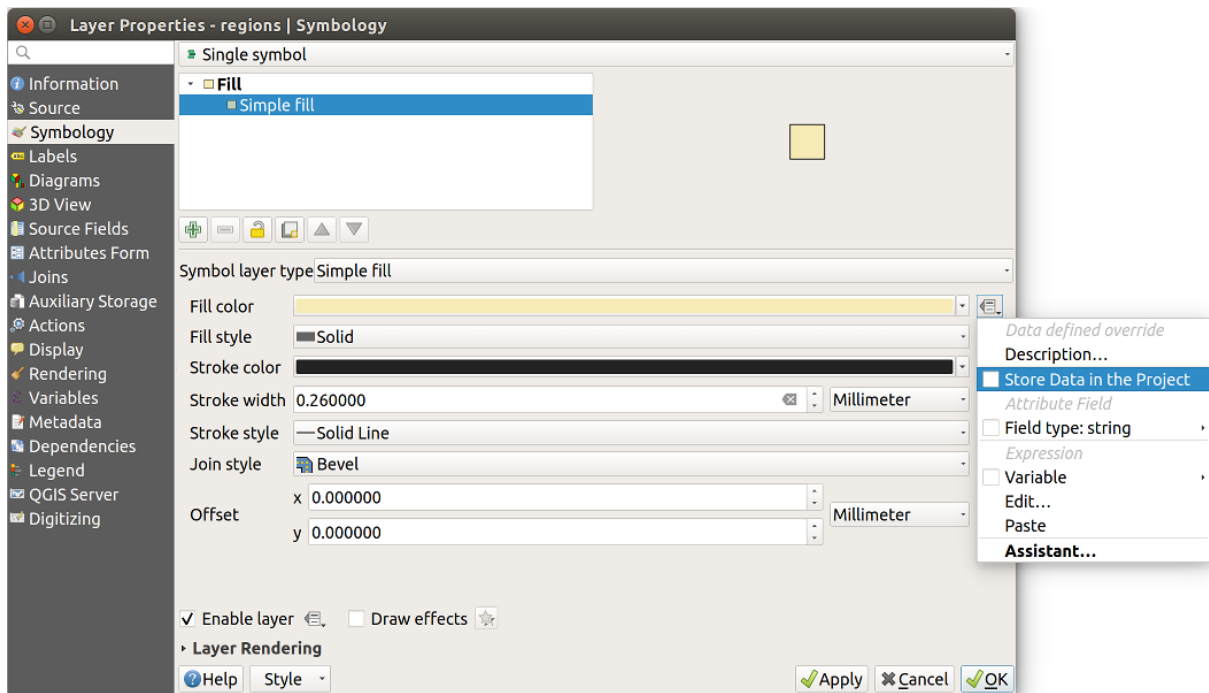


Abb. 16.50: Data-defined property menu for symbol

There are different attributes for each symbol (e.g. fill style, fill color, stroke color, etc...), so each auxiliary field representing an attribute requires a unique name to avoid conflicts. After selecting *Store data in the project*, a window opens and displays the *Type* of the field and prompts you to enter a unique name for the auxiliary field. For example, when creating a *Fill color* auxiliary field the following window opens:

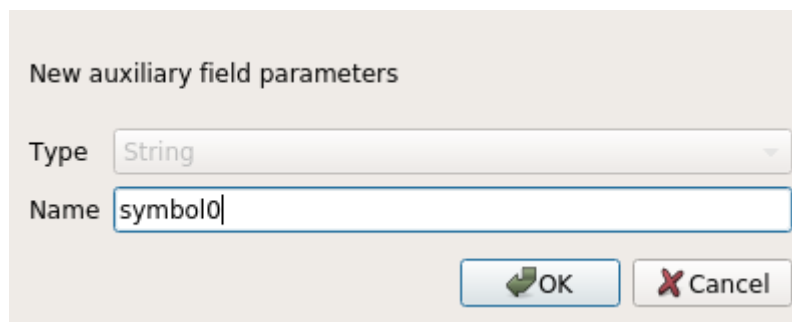


Abb. 16.51: Name of the auxiliary field for a symbol

Once created, the auxiliary field can be retrieved in the auxiliary storage tab:

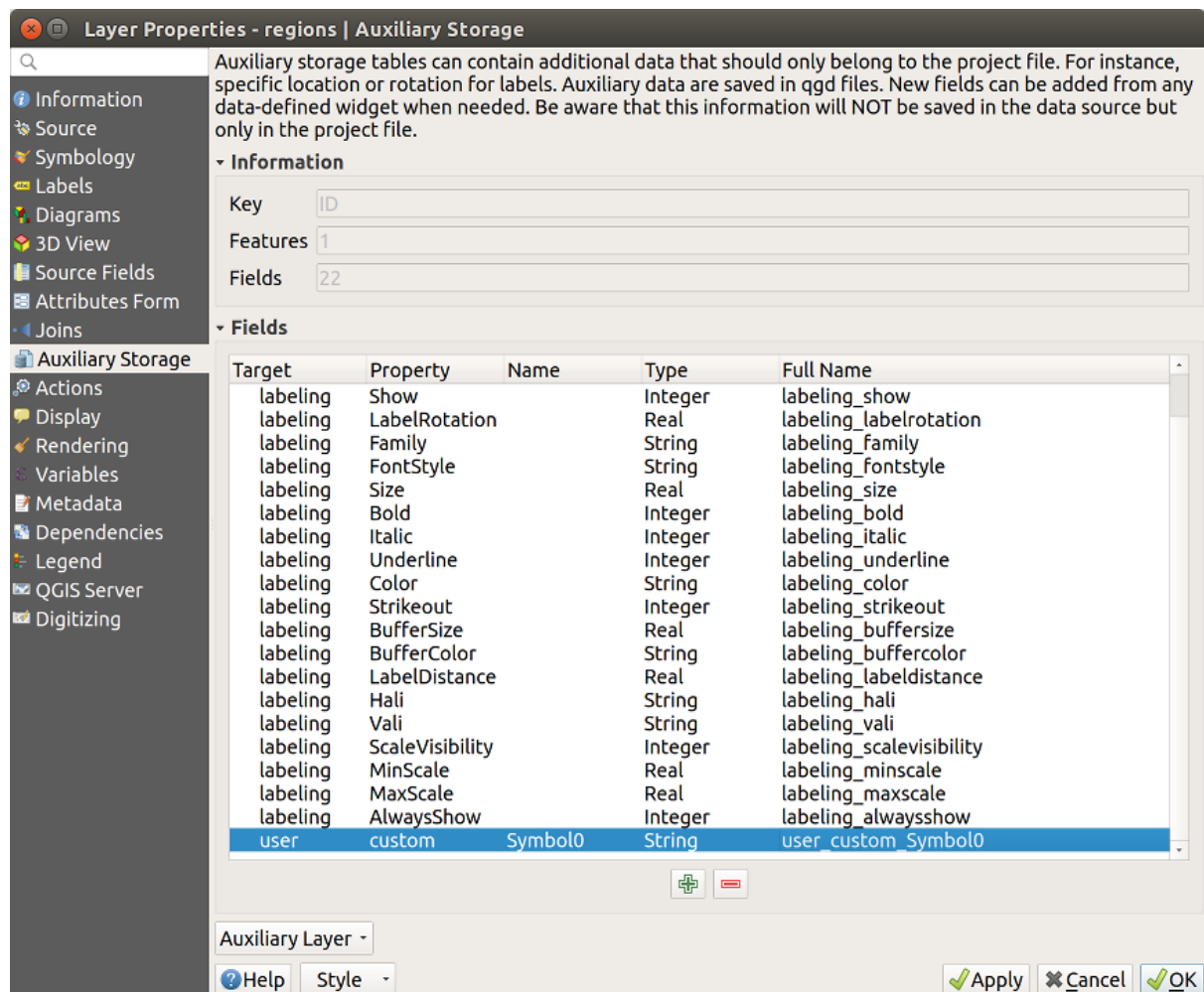


Abb. 16.52: Auxiliary field symbol

## Attributtabelle und Widgets

Auxiliary fields can be edited using the *attribute table*. However, not all auxiliary fields are initially visible in the attribute table.

Auxiliary fields representing attributes of a layer's symbology, labeling, appearance, or diagrams will appear automatically in the attribute table. The exception are attributes that can be modified using the *Label Toolbar* which are hidden by default. Auxiliary fields representing a `Color` have a widget **Color** set by default, otherwise auxiliary fields default to the **Text Edit** widget.

Auxiliary fields that represent attributes that can be modified using the *Label toolbar* are **Hidden** in the attribute table by default. To make a field visible, open the *Attribute Form properties tab* and change the value of an auxiliary field *Widget Type* from **Hidden** to another relevant value. For example, change the **auxiliary\_storage\_labeling\_size** to **Text Edit** or change **auxiliary\_storage\_labeling\_color** to the **Color** widget. Those fields will now be visible in the attribute table.

Auxiliary fields in the attribute table will appear like the following image:

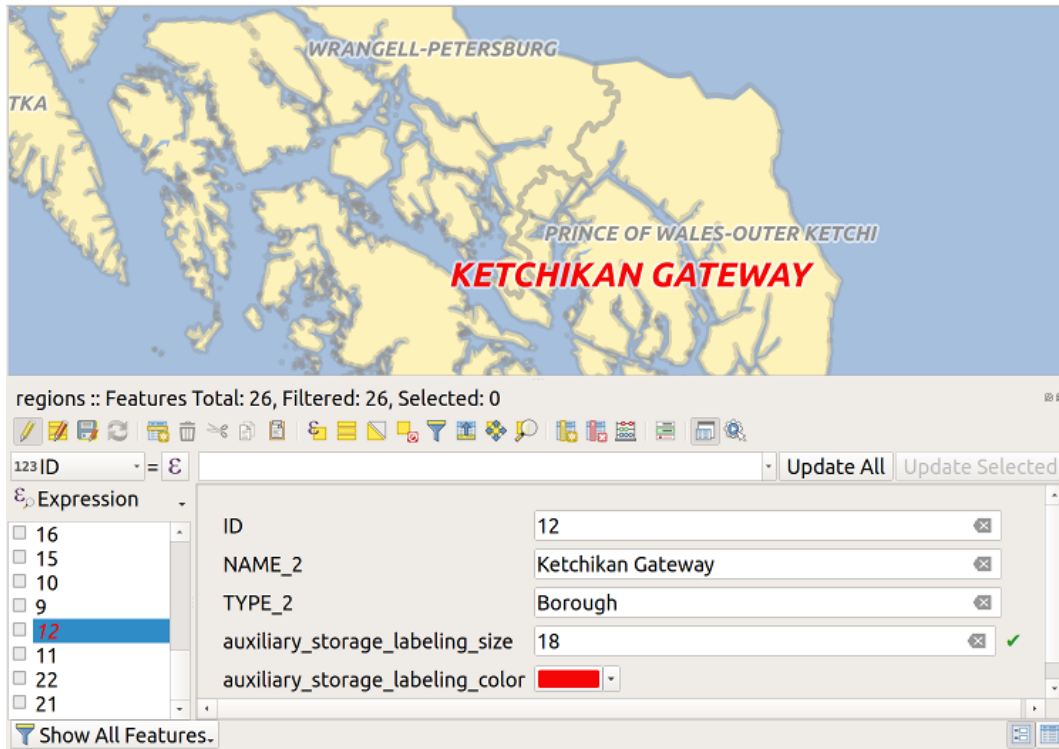


Abb. 16.53: Form with auxiliary fields

## Management

The *Auxiliary Layer* menu allows you to manage the auxiliary fields:

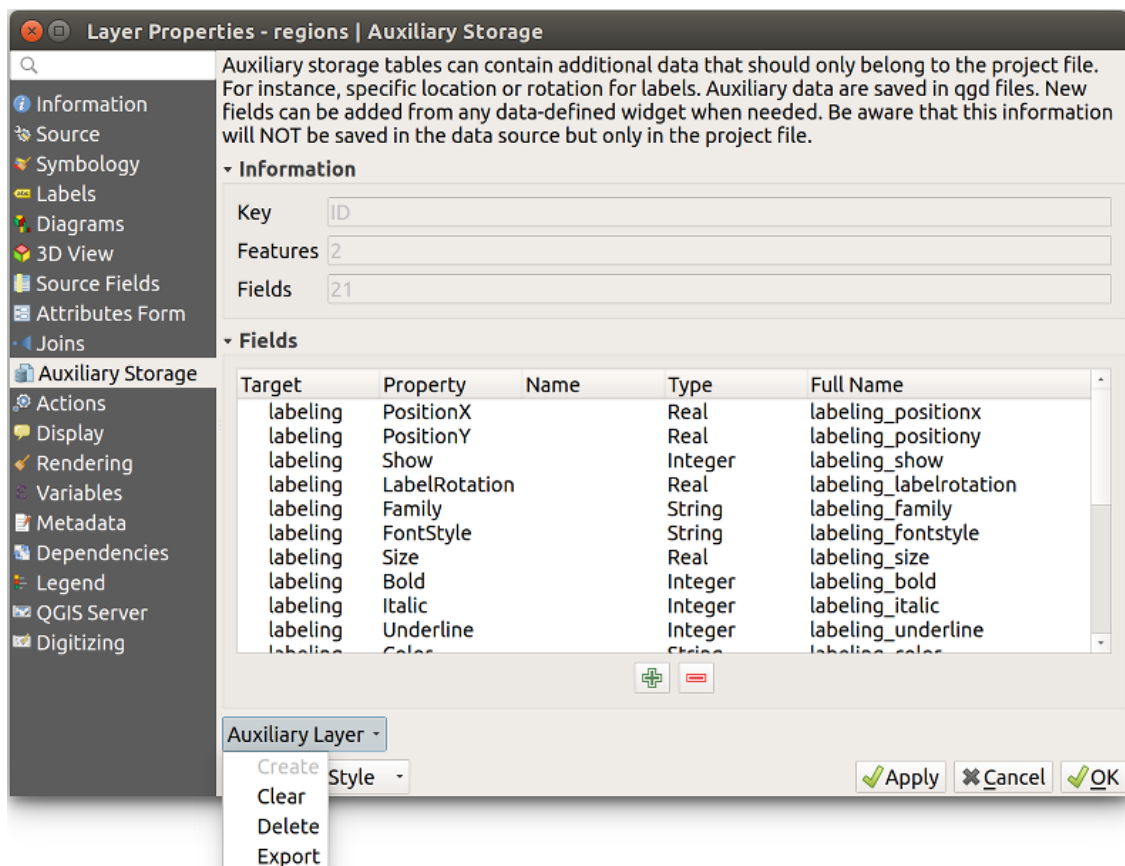


Abb. 16.54: Auxiliary layer management

The first item *Create* is disabled in this case because the auxiliary layer is already created. But in case of a fresh work, you can use this action to create an auxiliary layer. As explained in *Beschriftung*, a primary key will be needed then.

The *Clear* action allows to keep all auxiliary fields, but remove their contents. This way, the number of features using these fields will fall to 0.

The *Delete* action completely removes the auxiliary layer. In other words, the corresponding table is deleted from the underlying SQLite database and properties customization are lost.


Finally, the *Export* action allows to save the auxiliary layer as a *new vector layer*. Note that geometries are not stored in auxiliary storage. However, in this case, geometries are exported from the original data source too.

## Auxiliary storage database

When you save your project with the `.qgs` format, the SQLite database used for auxiliary storage is saved at the same place but with the extension `.qgd`.

For convenience, an archive may be used instead thanks to the `.qgz` format. In this case, `.qgd` and `.qgs` files are both embedded in the archive.

### 16.1.12 Eigenschaften Aktionen

The  *Actions* tab provides the ability to perform an action based on the attributes of a feature. This can be used to perform any number of actions, for example, running a program with arguments built from the attributes of a feature or passing parameters to a web reporting tool.

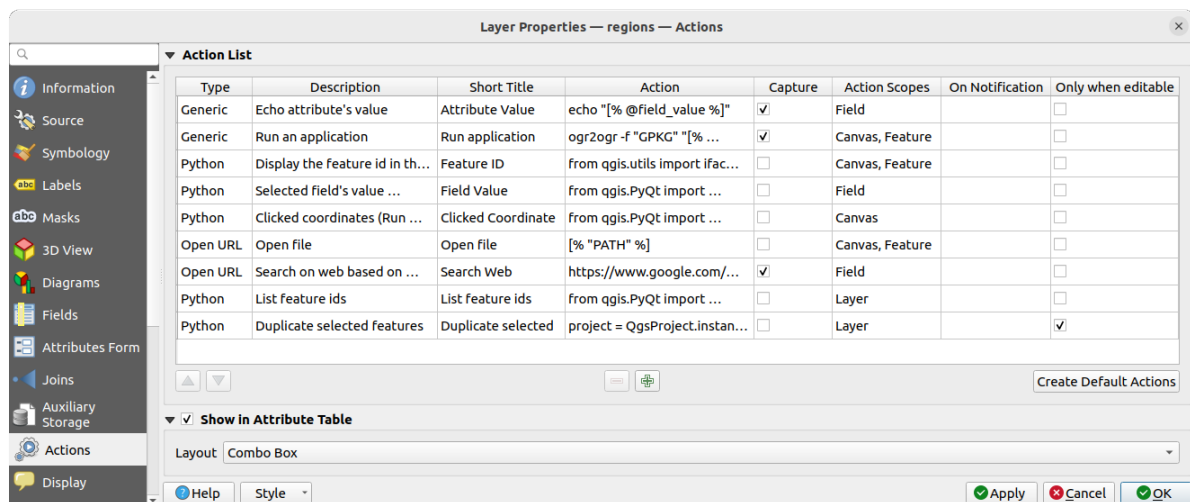


Abb. 16.55: Überblick über den Dialog Aktionen mit einigen Beispielaktionen

Actions are useful when you frequently want to run an external application or view a web page based on one or more values in your vector layer. They are of different types and can be used like this:

- *Generic*, *macOS*, *Windows* and *Unix* actions start an external process.
- *Python* actions execute a Python expression.
- *Generic* and *Python* actions are visible everywhere.
- *macOS*, *Windows* and *Unix* actions are visible only on the respective platform (i.e., you can define three „Edit“ actions to open an editor and the users can only see and execute the one „Edit“ action for their platform to run the editor).
- *Open URL*: Uses a HTTP GET request to open a provided URL.
- *Submit URL (urlencoded or JSON)*: Same as the *Open URL* action but using a HTTP POST request. Data are posted to a URL, using „application/x-www-form-urlencoded“ or „application/json“ if the body is a valid JSON.


An example of action call could be:

```
http://localhost:8000?[% url_encode(map('file', 'index.php')) %]
```


- *Submit URL (multipart)*: Same as the *Open URL* action but using a HTTP POST request. Data are posted to a URL, using „multipart/form-data“.



There are several examples included in the dialog. You can load them by clicking on *Create Default Actions*. To edit any of the examples, double-click its row. One example is performing a search based on an attribute value. This concept is used in the following discussion.

The  *Show in Attribute Table* allows you to display in the attribute table dialog the checked feature-scoped actions, either as *Combo Box* or as *Separate Buttons* (see [Configuring the columns](#)).

### Aktionen definieren

To define an attribute action, open the vector *Layer Properties* dialog and click on the *Actions* tab. In the *Actions* tab, click the  *Add a new action* to open the *Edit Action* dialog.

Select the action *Type* and provide a descriptive name for the action. The action itself must contain the name of the application that will be executed when the action is invoked. You can add one or more attribute field values as arguments to the application. When the action is invoked, any set of characters that start with a % followed by the name of a field will be replaced by the value of that field. The special characters %% will be replaced by the value of the field that was selected from the identify results or attribute table (see [Aktionen anwenden](#)). Double quote marks can be used to group text into a single argument to the program, script or command. Double quotes will be ignored if preceded by a backslash.

The *Action Scopes* allows you to define where the action should be available. You have following choices:

1. *Field*: action is available when right click in the cell within the attribute table, in the feature form and in the default action button of the main toolbar.
2. *Feature*: action is available when right click in the cell within the attribute table.
3. *Canvas*: action is available in the main action button in the toolbar.
4. *Form*: action is available only in a feature form designed using the *drag-and-drop* mode.
5. *Layer*: action is available in the action button in the attribute table toolbar. Be aware that this type of action involves the entire layer and not the single features.

Wenn Sie Feldnamen vorfinden die Substrings anderer Feldnamen sind (z.B., `col1` und `col10`) sollten Sie das angeben indem Sie den Feldnamen (und den % Buchstaben) mit eckigen Klammern umrunden (z.B. `[%col10]`). Hiermit wird vermieden dass der `%col10` Feldname mit dem, `%col1` Feldnamen mit einem 0 am Ende verwechselt wird. Die Klammern werden von QGS entfernt wenn es im Wert des Feldes ersetzt. Wenn Sie wollen dass das zu ersetzende Feld von eckigen Klammern umrandet wird verwenden Sie ein zweites Set wie hier: `[[%col10]]`.

Wenn Sie das *Objekte abfragen* Werkzeug verwenden, können Sie den *Identifikationsergebnis* Dialog öffnen. Es enthält eine (*Abgeleitet*) Item das layertyprelevante Informationen enthält. Die Werte in diesem Element können auf ähnliche Weise den anderen Feldern zugeordnet werden indem dem Abgeleitet Feldnamen ein (*Derived*) vorangeht. Zum Beispiel hat ein Punktlayer ein *X* und *Y* Feld und die Werte dieser Felder können in der Aktion mit `%(Derived) . X` und `%(Derived) . Y` verwendet werden. Die Abgeleitet Attribute sind nur in der *Objekte abfragen* Dialog Box erhältlich, jedoch nicht in der *Attributtabelle* Dialogbox.

Two example actions are shown below:

- `konqueror https://www.google.com/search?q=%nam`
- `konqueror https://www.google.com/search?q=%%`



In the first example, the web browser `konqueror` is invoked and passed a URL to open. The URL performs a Google search on the value of the `nam` field from our vector layer. Note that the application or script called by the action must be in the path, or you must provide the full path. To be certain, we could rewrite the first example as: `/opt/kde3/bin/konqueror https://www.google.com/search?q=%nam`. This will ensure that the `konqueror` application will be executed when the action is invoked.

Das zweite Beispiel nutzt den Ausdruck `%%`, welcher unabhängig ist von einem speziellen Feld. Beim Ausführen des Befehls wird der Ausdruck `%%` durch den Wert des jeweils selektierten Feldes aus *Objekte abfragen* oder der *Attributtabelle* ersetzt.





## Aktionen anwenden

QGIS offers many ways to execute actions you enabled on a layer. Depending on their settings, they can be available:

- in the drop-down menu of  Run Feature Action button from the *Attributes toolbar* or *Attribute table* dialog;
- when right-clicking a feature with the  Identify Features tool (see *Identifying Features* for more information);
- from the *Identify Results* panel, under the *Actions* section;
- as items of an *Actions* column in the *Attribute Table* dialog.

Wenn Sie eine Aktion mit %% Notation verwenden, machen Sie einen Rechtsklick auf den Feldwert im *Objekte abfragen* Dialog oder im *Attributtabelle* Dialog den Sie der Anwendung oder dem Skript übergeben wollen.


In einem weiteren Beispiel soll gezeigt werden, wie Attributwerte eines Vektorlayers abgefragt und in eine Textdatei mit Hilfe der Bash und des `echo` Kommandos geschrieben werden (funktioniert also nur unter  und evtl. ). Der Abfragelayer enthält die Felder Art `taxon_name`, Latitude `lat` und Longitude `long`. Wir möchten jetzt eine räumliche Selektion von Örtlichkeiten machen und diese Feldwerte in eine Textdatei für den ausgewählten Datensatz (in der QGIS Kartenansicht in gelb gezeigt) exportieren. Hier ist die Aktion, um dies zu erreichen:

```
bash -c "echo \"%taxon_name %lat %long\" >> /tmp/species_localities.txt"
```

Nachdem ein paar Orte auf dem Bildschirm ausgewählt wurden (diese erscheinen gelb hinterlegt), starten wir die Aktion mit der rechten Maustaste über den Dialog *Abfrageergebnisse* und können danach in der Textdatei die Ergebnisse ansehen:

```
Acacia mearnsii -34.0800000000 150.0800000000
Acacia mearnsii -34.9000000000 150.1200000000
Acacia mearnsii -35.2200000000 149.9300000000
Acacia mearnsii -32.2700000000 150.4100000000
```

As an exercise, we can create an action that does a Google search on the `lakes` layer. First, we need to determine the URL required to perform a search on a keyword. This is easily done by just going to Google and doing a simple search, then grabbing the URL from the address bar in your browser. From this little effort, we see that the format is <https://www.google.com/search?q=QGIS>, where `QGIS` is the search term. Armed with this information, we can proceed:

1. Laden Sie den Layer file:`lakes.shp`.
2. Öffnen Sie den *Layereigenschaften* Dialog indem Sie einen Doppelklick auf den Layer in der Legende machen und wählen Sie *Eigenschaften* aus dem Popup-Menü.
3. Click on the *Actions* tab.
4. Click  Add a new action.
5. Choose the *Open URL* action type,
6. Geben Sie einen Namen für die Aktion ein, z.B. `Google Search`.
7. Additionally you can add a *Short Name* or even an *Icon*.
8. Choose the *Action Scopes*. See *Aktionen definieren* for further information. Leave the default settings for this example.
9. For the action, add the URL used for doing a Google search, up to but not including the search term: `https://www.google.com/search?q=`
10. The text in the *Action* field should now look like this:

```
https://www.google.com/search?q=
```

11. Click on the drop-down box containing the field names for the `lakes` layer. It's located just to the left of the *Insert* button.

12. From the drop-down box, select *NAMES* and click *Insert*.
13. Your action text now looks like this:

```
https://www.google.com/search?q=[%NAMES%]
```

14. To finalize and add the action, click the *OK* button.

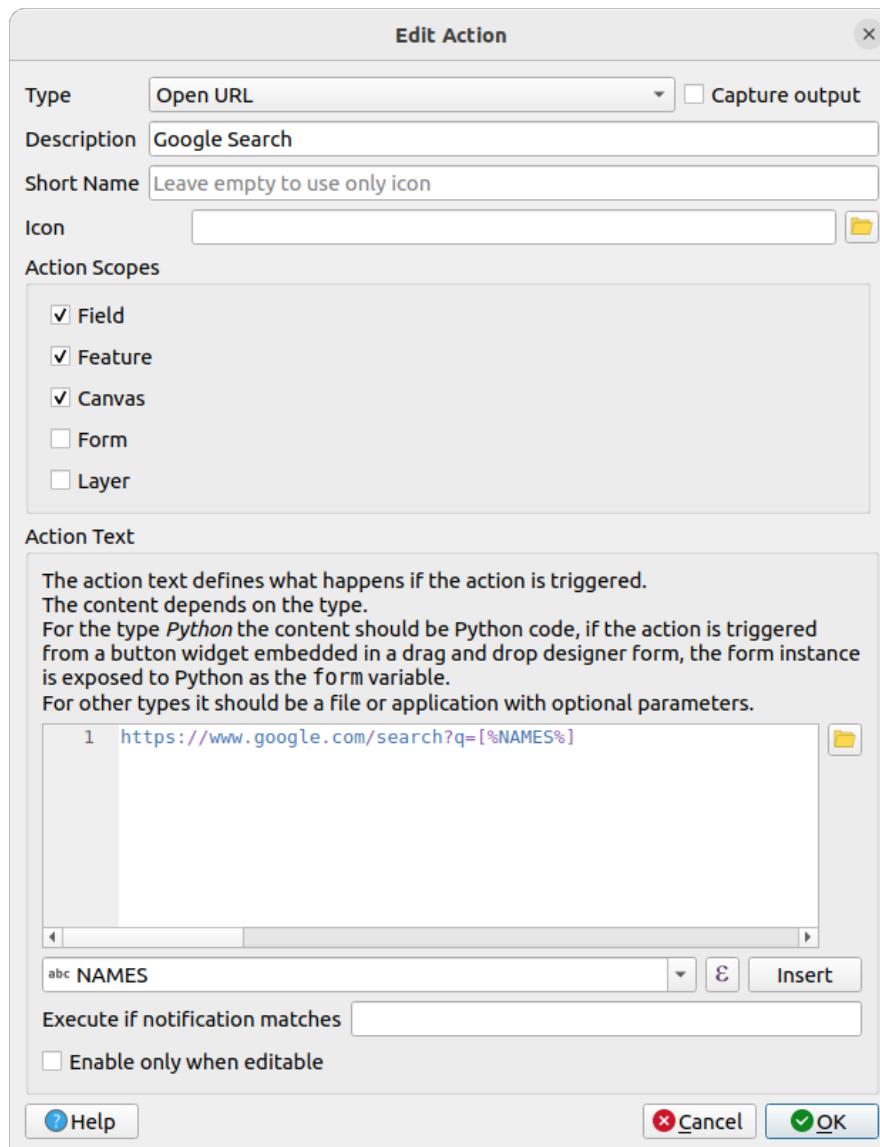


Abb. 16.56: Edit action dialog configured with the example

Damit ist der Vorgang abgeschlossen und das Gerät ist einsatzbereit.

Close the *Layer Properties* dialog and zoom in to an area of interest. Make sure the `lakes` layer is active and identify a lake. In the result box you'll now see that our action is visible:

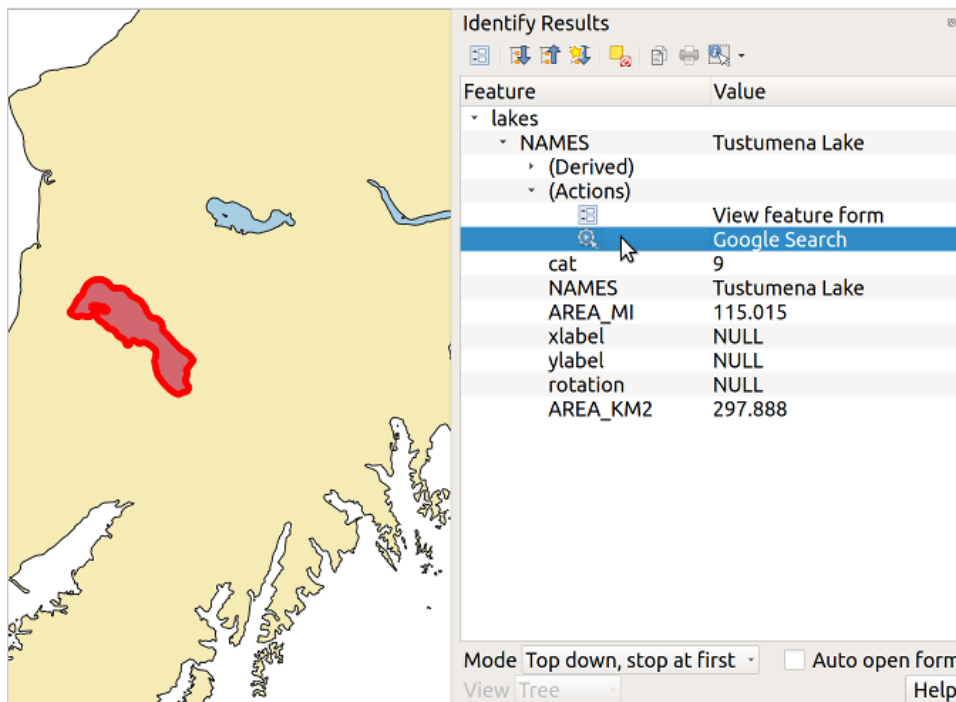


Abb. 16.57: Wählen Sie ein Objekt und eine Aktion aus

When we click on the action, it brings up our default browser and navigates to the URL <https://www.google.com/search?q=Tustumena>. It is also possible to add further attribute fields to the action. Therefore, you can add a + to the end of the action text, select another field and click on *Insert Field*. In this example, there is just no other field available that would make sense to search for.

Sie können auch mehrere Aktionen für einen Layer definieren. Sie alle werden dann bei der Abfrage von Objekten im *Identifikationsergebnis* Dialog angezeigt.

Sie können auch die aus der Attributtabelle aufrufen, indem Sie eine Zeile mit der rechten Maustaste auswählen und dann die Aktion aus dem Pop-up-Menü wählen.

Sie sehen, man kann sich eine Vielzahl interessanter Aktionen ausdenken. Wenn Sie z.B. einen Punktlayer mit einzelnen Punkten haben, an denen Photos geschossen wurden, dann können Sie eine Aktion erstellen, über die Sie dann das entsprechende Foto anzeigen lassen können, wenn Sie auf den Punkt in der Karte klicken. Man kann auch zu bestimmten Attributen webbasierte Information ablegen (z.B. in einer HTML-Datei) und diese dann über eine Aktion anzeigen lassen, etwa so wie in dem Google Beispiel.

Wir können auch komplexere Beispiele erstellen, indem wir z.B. **Python** Aktionen verwenden.

Usually, when we create an action to open a file with an external application, we can use absolute paths, or eventually relative paths. In the second case, the path is relative to the location of the external program executable file. But what about if we need to use relative paths, relative to the selected layer (a file-based one, like Shapefile or Spatialite)? The following code will do the trick:

```
command = "firefox"
imagerelpath = "images_test/test_image.jpg"
layer = qgis.utils.iface.activeLayer()
import os.path
layerpath = layer.source() if layer.providerType() == 'ogr'
else (qgis.core.QgsDataSourceURI(layer.source()).database()
      if layer.providerType() == 'spatialite' else None)
path = os.path.dirname(str(layerpath))
image = os.path.join(path, imagerelpath)
import subprocess
subprocess.Popen( [command, image] )
```

Wir müssen uns nur ins Gedächtnis rufen dass es sich um eine *Python* Aktion handelt und dass das Ändern der *command* und *imagerelpath* Variablen auf unsere Bedürfnisse angepasst wird.

Was aber wenn der relative Pfad relativ zur (gespeicherten) Projektdatei sein muss? Der Code der Python Aktion würde wie folgt lauten:

```
command = "firefox"
imagerelpath = "images_test/test_image.jpg"
projectpath = qgis.core.QgsProject.instance().fileName()
import os.path
path = os.path.dirname(str(projectpath)) if projectpath != '' else None
image = os.path.join(path, imagerelpath)
import subprocess
subprocess.Popen( [command, image] )
```


Ein anderes Python Aktion Beispiel ist das mit wir dem Projekt neue Layer hinzufügen können. Z.B. wird in den folgenden Beispielen dem Projekt ein Vektorlayer beziehungsweise ein Rasterlayer hinzugefügt. Die Namen der Dateien, die dem Projekt hinzugefügt werden sollen, und die Namen, die den Layern gegeben werden, sind datengesteuert (*filename* und *layername* sind Spaltennamen der Attributtabelle des Vektorlayers in dem die Aktion erstellt wurde):




```
qgis.utils.iface.addVectorLayer('/yourpath/[% "filename" %].shp',
    ' [% "layername" %]', 'ogr')
```

Um eine Rasterdatei hinzuzufügen (ein TIF-Bild in diesem Beispiel) wird daraus:

```
qgis.utils.iface.addRasterLayer('/yourpath/[% "filename" %].tif',
    ' [% "layername" %]')
```

### 16.1.13 Eigenschaften Anzeige

The  *Display* tab helps you configure fields to use for feature identification:

- The *Display name*: based on a field or an *expression*. By default, this is set to the first field in the layer if no field with <name> component exists. This is used as:
  - the label shown on top of the feature information in the *Identify tool* results
  - the field used in the *locator bar* when looking for features in all layers
  - the feature identifier in the attribute table *form view*
  - the feature identifier when the map or layout is exported to a layered output format such as GeoPDF
  - the map tip information, i.e. the message displayed in the map canvas when hovering over a feature of the active layer with the  *Show Map Tips* icon pressed. Applicable when  *Enable Map Tips* is active and no *HTML Map Tip* is set.
-  *Enable Map Tips* controls whether to display map tips for the layer
- The *HTML Map Tip* provides a complex and full HTML text editor for map tips, mixing QGIS expressions and html styles and tags (multiline, fonts, images, hyperlink, tables, ...). You can check the result of your code sample in the *Preview* frame (also convenient for previewing the *Display name* output).

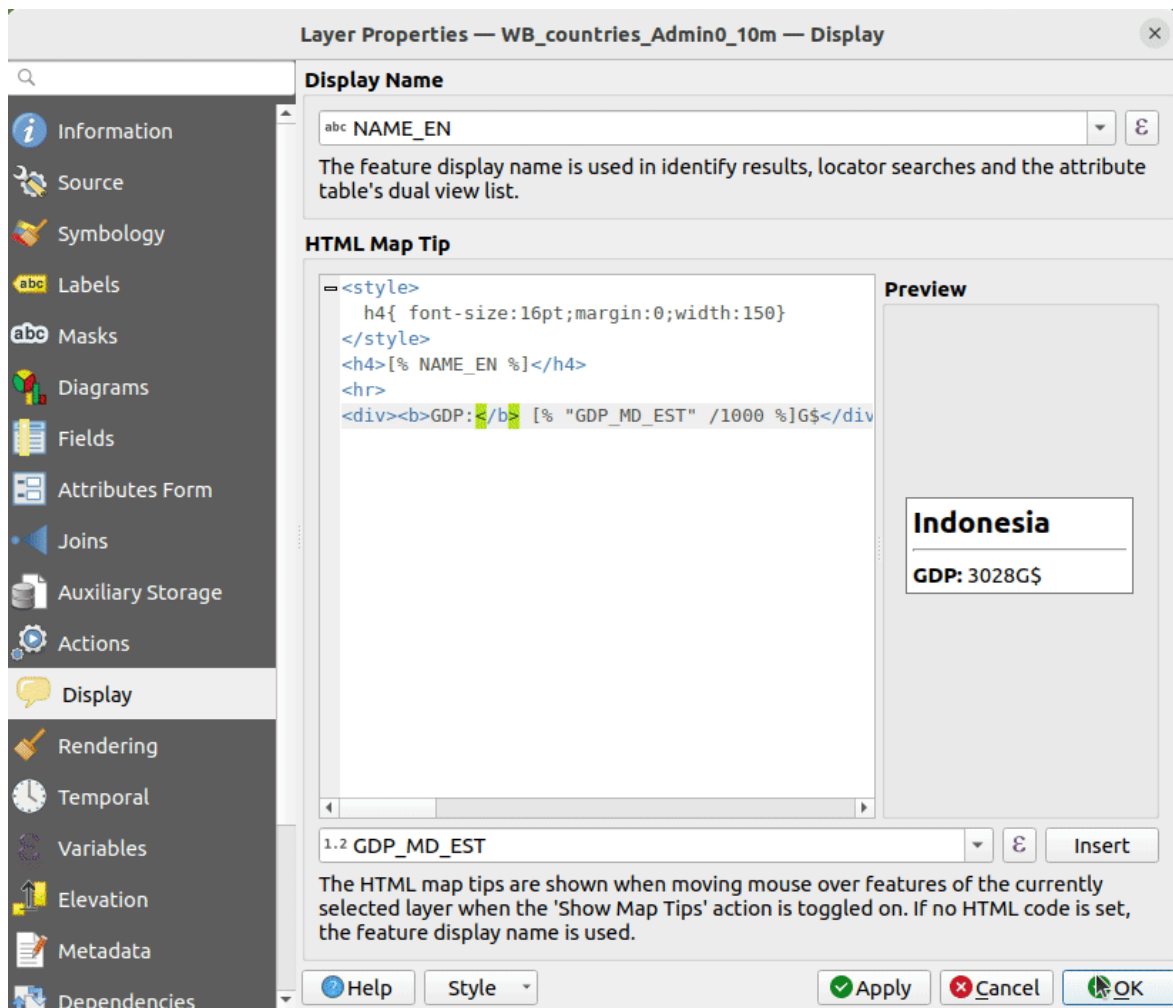




Abb. 16.58: HTML-Code für Kartenhinweis

To display map tips:

1. Select the menu option *View ► Show Map Tips* or click on the  Show Map Tips icon of the *Attributes Toolbar*.
2. Make sure that the layer you target is active and has the  *Enable Map Tips* property checked.
3. Move over a feature, and the corresponding information will be displayed over.

Map tip is a cross-layer feature meaning that once activated, it stays on and applies to any map tip enabled layer in the project until it's toggled off.



Abb. 16.59: Kartenhinweis erstellt mit HTML-Code

## 16.1.14 Eigenschaften Darstellung

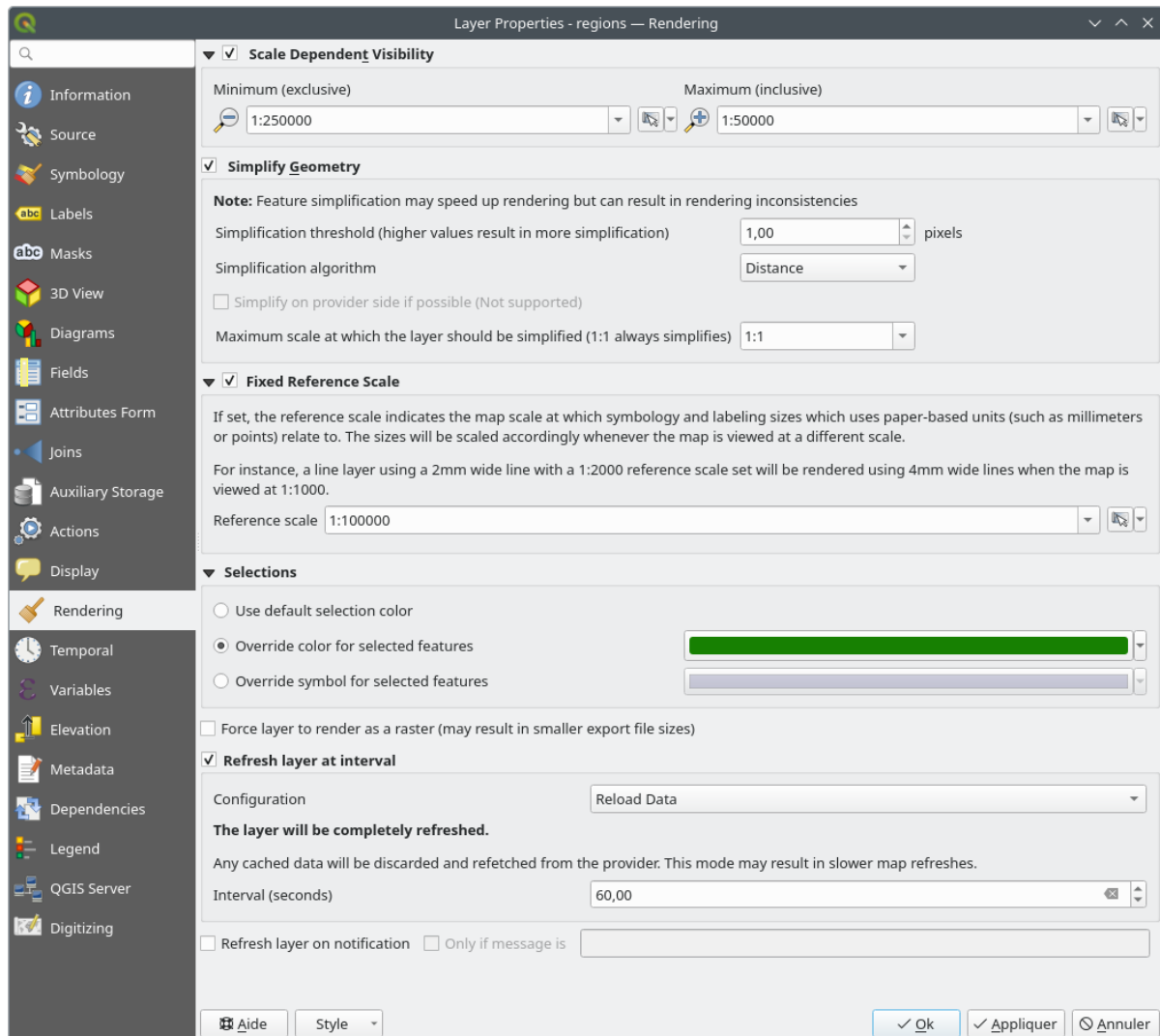





Abb. 16.60: Layer Rendering Properties dialog


The  **Rendering** tab offers following properties:

- Under  **Scale dependent visibility**, you can set the *Maximum (inclusive)* and *Minimum (exclusive)* scales, defining a range of scales in which features will be visible. Out of this range, they are hidden. The  **Set to current canvas scale** button helps you use the current map canvas scale as boundary of the range visibility. See [Visibility Scale Selector](#) for more information.

---

**Bemerkung:** You can also activate scale dependent visibility on a layer from within the *Layers* panel: right-click on the layer and in the contextual menu, select *Set Layer Scale Visibility*.


---

- QGIS offers support for on-the-fly feature generalisation. This can improve rendering times when drawing many complex features at small scales. This feature can be enabled or disabled in the layer settings using the  **Simplify geometry** option. There is also a global setting that enables generalisation by default for newly added layers (see [global simplification](#) for more information).

---


**Bemerkung:** Objektgeneralisierung, kann in einigen Fällen, Artefakte in Ihre gerenderte Ausgabe einführen. Dies können Splitter zwischen Polygonen und ungenau Darstellung bei der Verwendung von Offset-basierten Symbol Layern sein.

---


- The  *Fixed reference scale* indicates the map scale at which symbology and labeling sizes which uses paper-based units (such as millimeters or points) relate to. The sizes will be scaled accordingly whenever the map is viewed at a different scale.

For instance, a line layer using a 2mm wide line with a 1:2000 *Reference scale* set will be rendered using 4mm wide lines when the map is viewed at 1:1000.



- The *Selections* group allows you to control whether a specific color or symbol should be used in place of the defaults (*Project properties* ► *General* ► *Selection color*) for a particular layer. This is useful to improve the visibility of selected features with certain symbology:
  - *Use default selection color*
  - *Override color for selected features*: e.g. if the layer uses a yellow color by default and the standard yellow selection is not visible.
  - *Override symbol for selected features*: e.g. if a line layer uses a thin symbol, and coloring the lines might not make them visible enough, overriding the symbol with a thicker line can help. Also, if the layer uses raster symbology or gradient fills/lines/shapburst with color ramp symbology, then the default selection color is not applied at all; being able to set a specific simpler symbol to use for selected features in the layer can help.
- Rendering extremely detailed layers (e.g. polygon layers with a huge number of nodes), can cause layout exports in PDF/SVG format to be huge as all nodes are included in the exported file. This can also make the resultant file very slow to work with/open in other programs.

Aktivieren von  *Force layer to render as raster* zwingt diese Layer dazu gerastert zu werden, so dass die exportierten Dateien nicht alle enthaltenen Knoten in diesen Layern enthalten müssen und die Wiedergabe daher beschleunigt wird.

You can also do this by forcing the layout to export as a raster, but that is an all-or-nothing solution, given that the rasterisation is applied to all layers. Alternatively, you can rely on geometry simplification in [layout export settings](#).


-  *Refresh layer at interval*: controls whether and how regular a layer can be refreshed. Available *Configuration* options are:
  - *Reload data*: the layer will be completely refreshed. Any cached data will be discarded and refetched from the provider. This mode may result in slower map refreshes.
  - *Redraw layer only*: this mode is useful for animation or when the layer's style will be updated at regular intervals. Canvas updates are deferred in order to avoid refreshing multiple times if more than one layer has an auto update interval set.

It is also possible to set the *Interval (seconds)* between consecutive refreshments.

- Depending on the data provider (e.g. PostgreSQL), notifications can be sent to QGIS when changes are applied to the data source, out of QGIS. Use the  *Refresh layer on notification* option to trigger an update. You can also limit the layer refresh to a specific message set in the  *Only if message is* text box.



### 16.1.15 Eigenschaften Zeitlich

The  *Temporal* tab provides options to control the rendering of the layer over time. Such dynamic rendering requires the *temporal navigation* to be enabled over the map canvas.

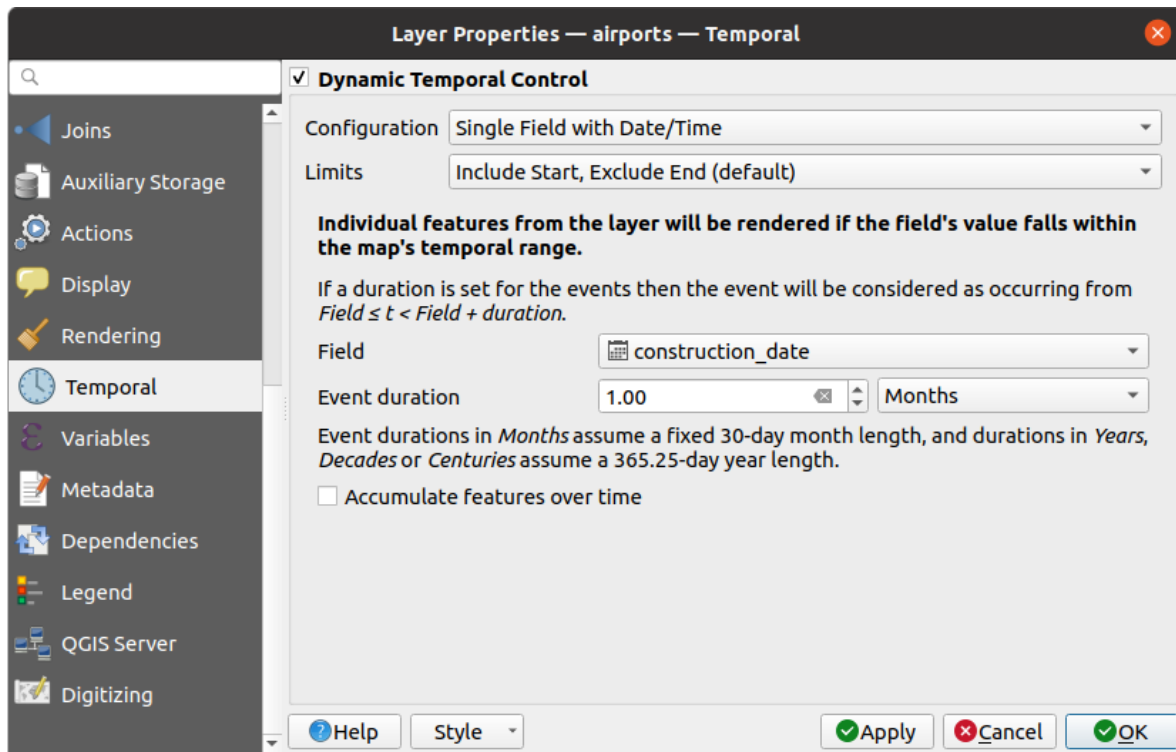



Abb. 16.61: Vector layer temporal properties dialog


Check the  *Dynamic Temporal Control* option to configure the vector layer temporal rendering. Depending on the structure of your dataset, you may want to use one of the provided *Configuration* options:

- *Fixed time range*: all the features are rendered if the map canvas temporal frame overlaps the given *Start date* and *End date* range.
- *Single field with date/time*: features are rendered if their *Field's* value falls within the map canvas temporal frame. An *Event duration* can be set. With checking the *Accumulate features over time* option, all features which occur before or within the map's temporal range will continue to be rendered. The event duration is thus ignored.
- *Separate fields for start and end date/time*: features are rendered if the range specified by their *Start field* and *End field* values overlaps the map canvas temporal.
- *Separate fields for start and event duration*: features are rendered if the range defined by their *Start field* and *Event duration field* values overlaps the map canvas temporal.
- *Start and end date/time from expressions*: features are rendered if the time range specified by the fields *Start expression* and *End expression* overlaps the map canvas temporal.
- *Redraw layer only*: the layer is redrawn at each new animation frame but no time-based filtering is applied to the features. It's useful when the layer uses time-based expression values for renderer settings (e.g. data-defined symbology).

It is also possible to set the *Limits* of the features time range as:

- *Include start, exclude end*
- *Include start, include end*

### 16.1.16 Eigenschaften Variablen


The  *Variables* tab lists all the variables available at the layer's level (which includes all global and project's variables).

It also allows the user to manage layer-level variables. Click the  button to add a new custom layer-level variable.

Likewise, select a custom layer-level variable from the list and click the  button to remove it.

More information on variables usage in the General Tools *Storing values in Variables* section.

### 16.1.17 Elevation Properties

The  *Elevation* tab provides options to control the layer elevation properties within a *3D map view* and its appearance in the *profile tool charts*. Specifically, you can set:

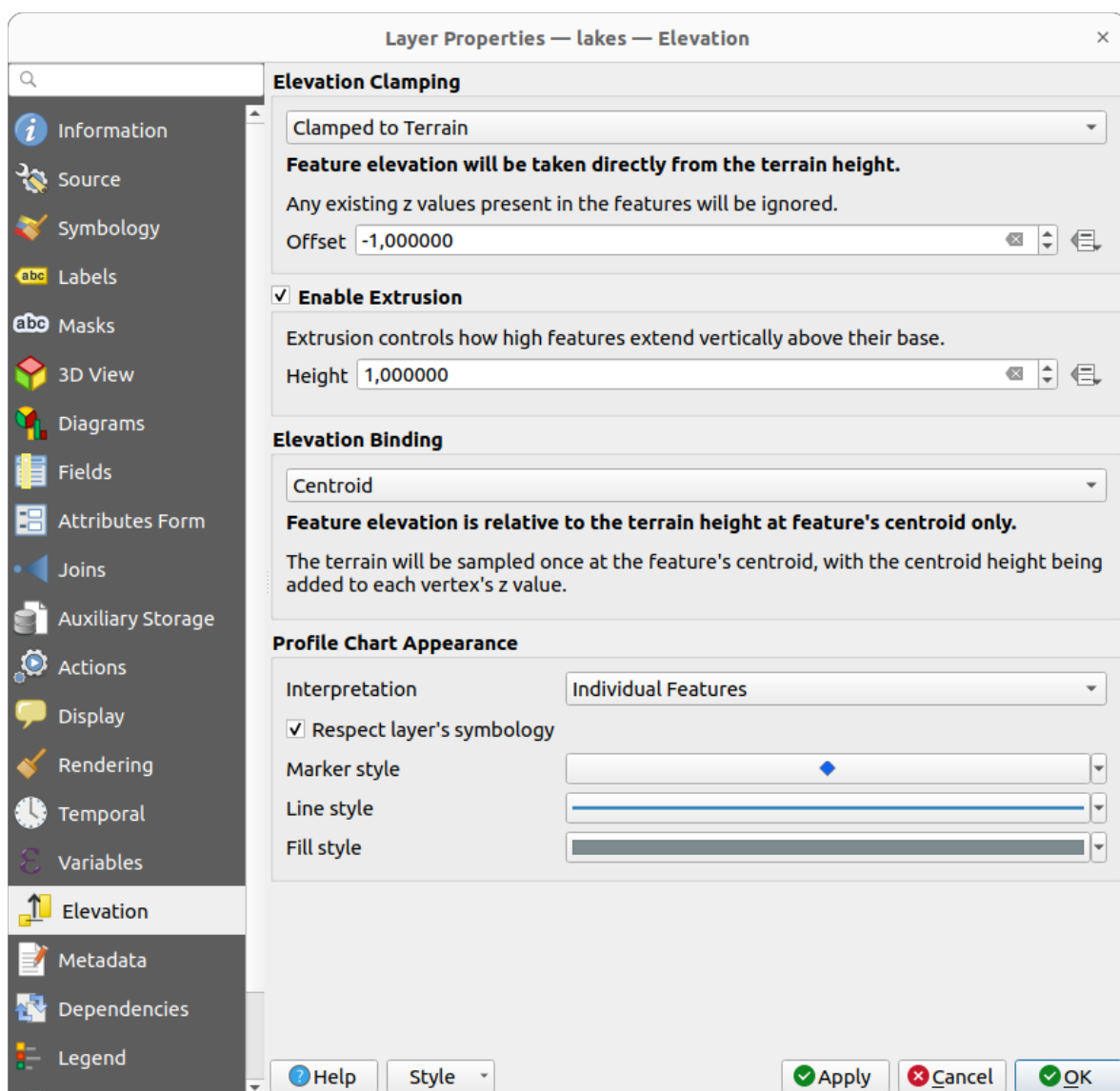




Abb. 16.62: Vector layer elevation properties dialog


- *Elevation Clamping*: defines how and whether the features altitude should be:

- *Clamped to terrain*: takes elevation directly from the terrain height and ignores any existing Z values in the features. A data-defined *Offset* value from the terrain can also be filled.
- *Relative to terrain*: any existing Z values in the features are added to the terrain height. A *Scale* factor followed by a data-defined *Offset* can be used to adjust the elevation. This option is not available for 2D geometry layers.
- *Absolute*: ignores the terrain height and directly takes Z values from the features for the elevation. A *Scale* factor followed by a data-defined *Offset* can be used to adjust the elevation. For 2D geometry layers (with no Z values), a data-defined *Base height* can instead be set.
-  *Enable extrusion*: you can set a *Height* to control how high features vertically extend above their base. This is convenient to indicate that a 2D geometry layers, e.g. a polygon building footprints layer, actually represents 3D objects.
- *Elevation Binding*: only relevant when combining an *Elevation clamping* relying on the terrain with a line or polygon layer, this option controls how feature elevation is set relative to the terrain height. The terrain can be sampled:
  - at the feature's *Centroid*, with the centroid height being added to each vertex's z value
  - at every individual *Vertex* before being added to the vertex's z value
- *Profile Chart Appearance*: controls how features are rendered when drawing a profile chart. Two main *Interpretation* modes are available:
  - as *Individual features*: samples discrete positions where the cross section profile line intersects the vector features. That intersection can be represented as point, line or surface depending on the layer type and whether an extrusion is applied.


With checking  *Respect layer symbology*, features will be rendered on the profile chart with their corresponding *layer styling* (allowing e.g. categorized classes to be visible on the profile chart). If the profile symbol type does not match the layer's renderer symbol types, only the symbol color from the renderer is applied to the profile symbol.

Depending on the layer settings, profile symbols can be represented with a custom style, using:


- \* *Marker style*: for non-extruded point and line features, and for non-extruded polygon features touched by the profile line
- \* *Line style*: for extruded point and line features, and for non-extruded polygon features intersected by the profile line
- \* *Fill style*: for extruded polygon features
- as *Continuous Surface* (e.g. *contours*): the elevation chart will be rendered as a surface instead of separate features by joining the sampled elevation results into a continuous line. This can enhance visualisation and is designed for vector layers which represent a continuous elevation surface, e.g. contour lines or surveyed elevation points. The profile *Style* can be set as:
  - \* a *Line* with a specific *Line style*
  - \* an elevation surface rendered using a fill symbol either above (*Fill above*) or below (*Fill below*) the elevation curve line. The surface symbology is represented using:
    - a *Fill style*
    - and a *Limit*: the maximum (respectively minimum) altitude determining how high the fill surface will be

Moreover, you can check  *Show markers at sampled points* to make them visible over the interpretation line and assign them a *Marker style*.

## 16.1.18 Eigenschaften Metadaten


Der Tab  *Metadaten* ermöglicht es Ihnen, Metadaten für Ihren Layer zu erstellen oder zu editieren. Siehe [Metadata](#) für weitere Informationen.



## 16.1.19 Eigenschaften Abhängigkeiten

The  *Dependencies* tab allows to declare data dependencies between layers. A data dependency occurs when a data modification in a layer, not by direct user manipulation, may modify data of other layers. This is the case for instance when geometry of a layer is updated by a database trigger or custom PyQGIS scripting after modification of another layer's geometry.

In the *Dependencies* tab, you can select any layers which may externally alter the data in the current layer. Correctly specifying dependent layers allows QGIS to invalidate caches for this layer when the dependent layers are altered.

## 16.1.20 Eigenschaften Legende

The  *Legend* properties tab provides you with advanced settings for the *Layers panel* and/or the *print layout legend*. These options include:

- Depending on the symbology applied to the layer, you may end up with several entries in the legend, not necessarily readable/useful to display. The *Legend placeholder image* helps you [select an image](#) for replacement, displayed both in the *Layers* panel and the print layout legend.
-  *Show label legend*: Displays overviews of the different label settings as entries in the legends. The *label style* is previewed along with the description.
-  *Text on symbols*: In some cases it can be useful to add extra information to the symbols in the legend. With this frame, you can affect to any of the symbols used in the layer symbology a text that is displayed over the symbol, in both *Layers* panel and print layout legend. This mapping is done by typing each text next to the symbol in the table widget or filling the table using the *Set Labels from Expression* button. Text appearance is handled through the font and color selector widgets of the *Text Format* button.

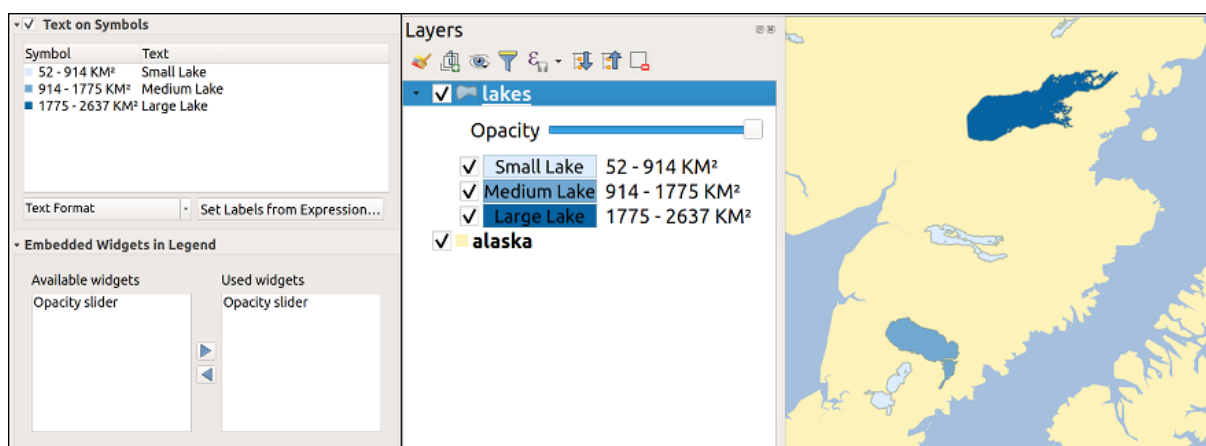


Abb. 16.63: Setting text on symbols (left) and its rendering in the *Layers* panel (right)

- a list of widgets you can embed within the layer tree in the *Layers* panel. The idea is to have a way to quickly access some actions that are often used with the layer (setup transparency, filtering, selection, style or other stuff...).

By default, QGIS provides transparency widget but this can be extended by plugins registering their own widgets and assign custom actions to layers they manage.

### 16.1.21 Eigenschaften QGIS Server

The  **QGIS Server** tab consists of *Description*, *Attribution*, *Metadata URL*, and *Legend URL* sections.

From the *Description* section, you can change the *Short name* used to reference the layer in requests (to learn more about short names, read `services_basics_short_name`). You can also add or edit a *Title* and *Abstract* for the layer, or define a *Keyword list* here. These keyword lists can be used in a metadata catalog. If you want to use a title from an XML metadata file, you have to fill in a link in the *Data URL* field.

Use *Attribution* to get attribute data from an XML metadata catalog.

In *Metadata URL*, you can add the general paths to the XML metadata catalog. This information will be saved in the QGIS project file for subsequent sessions and will be used for QGIS Server.

In the *Legend URL* section, you can provide the url of a legend image in the url field. You can use the Format drop-down option to apply the appropriate format of the image. Currently png, jpg and jpeg image formats are supported.

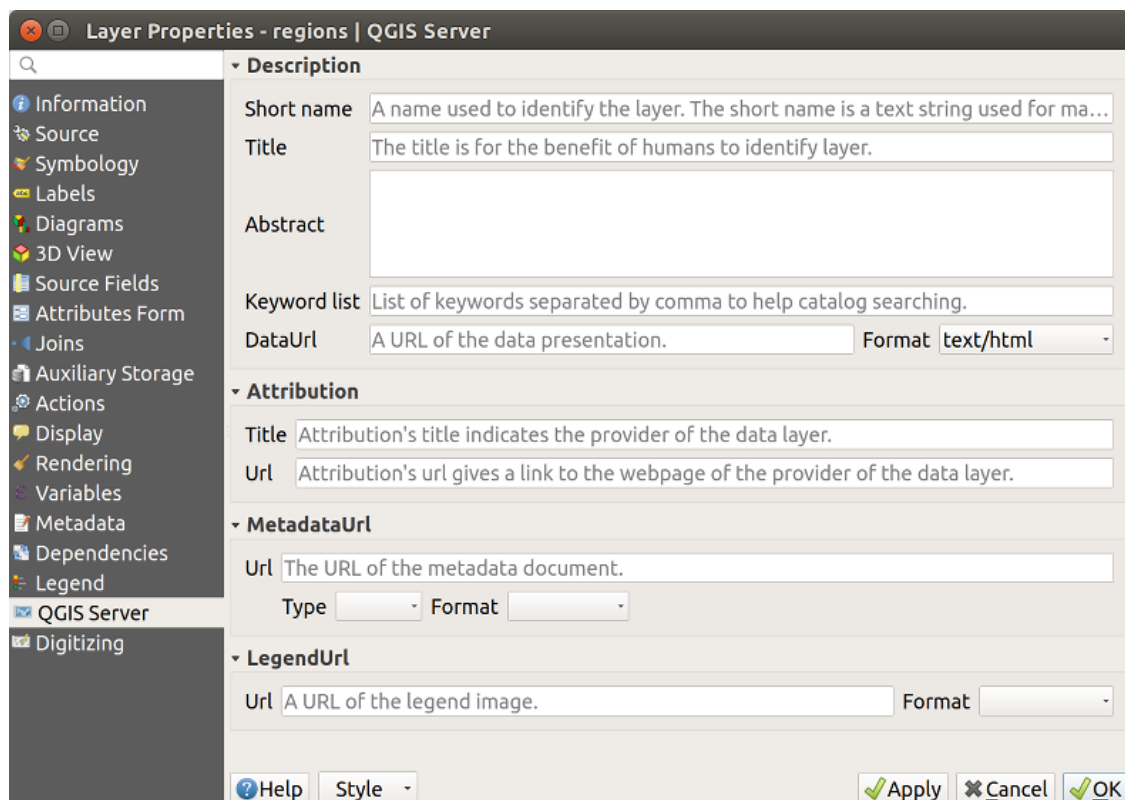



Abb. 16.64: QGIS Server tab in vector layers properties dialog

Um mehr über QGIS Server zu erfahren, lesen Sie die QGIS-Server-manual.

### 16.1.22 Eigenschaften Digitalisieren

The  **Digitizing** tab gives access to options that help to ensure the quality of digitized geometries.

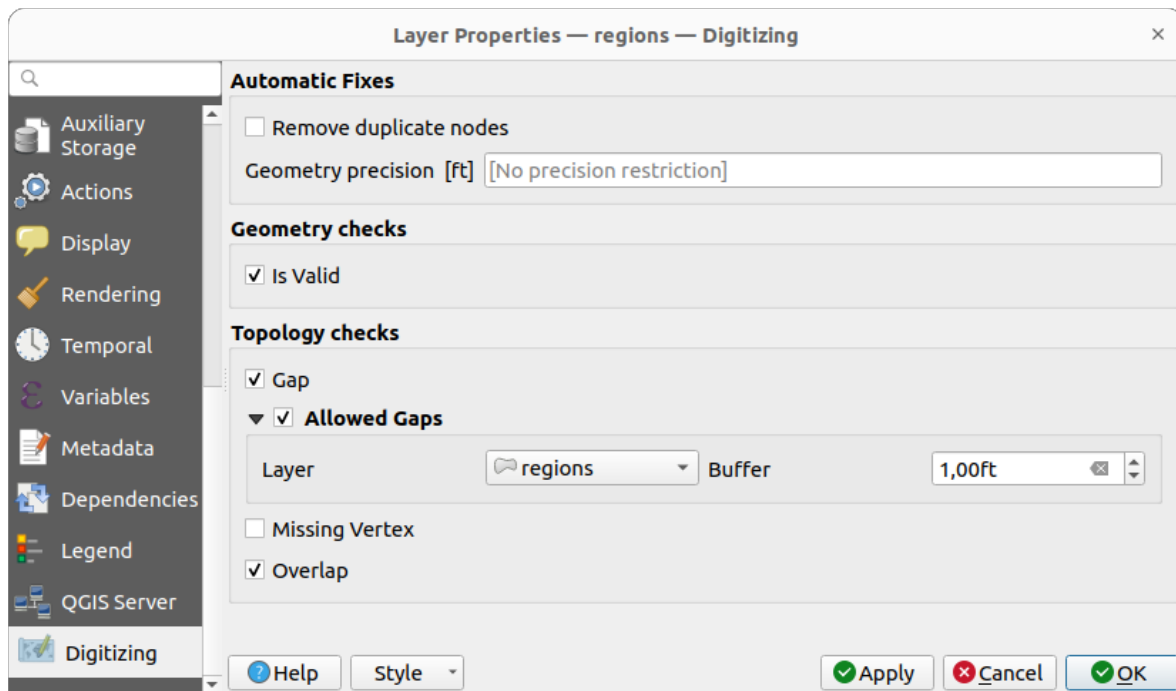


Abb. 16.65: The QGIS Digitizing tab in the vector layers properties dialog

## Automatic Fixes

Options in the *Automatic Fixes* section will directly affect the vertices of any geometry which is added or modified. If the ☒ *Remove duplicate nodes* option is checked, any two subsequent vertices with exactly the same coordinates will be removed. If the *Geometry precision* is set, all vertices will be rounded to the closest multiple of the configured geometry precision. The rounding will happen in the layer coordinate reference system. Z and M values are not rounded. With many map tools, a grid is shown on the canvas while digitizing.

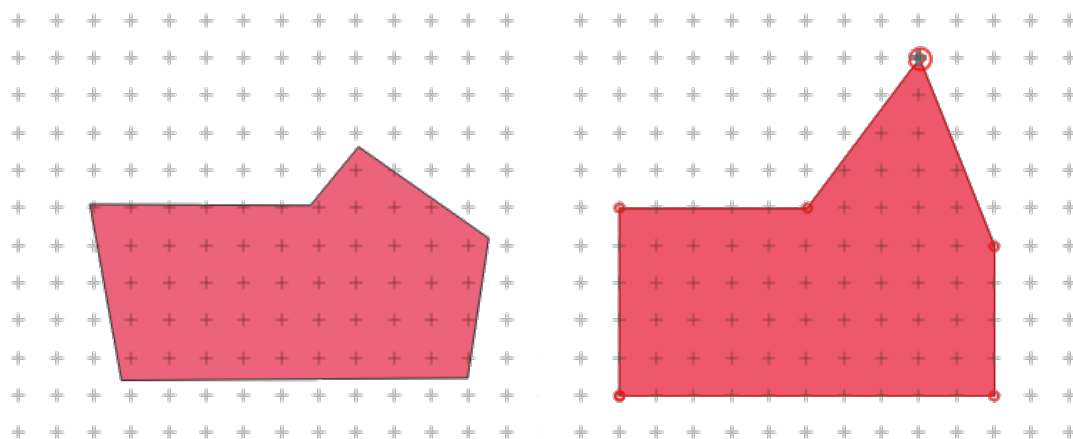




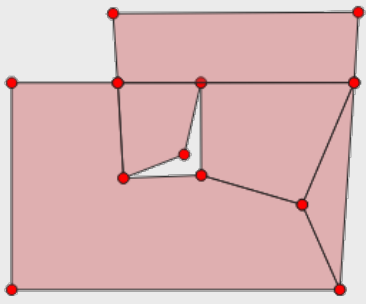

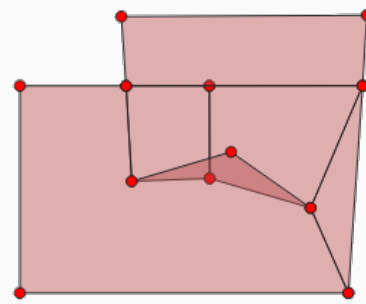

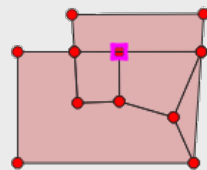
Abb. 16.66: Wird der oberste Stützpunkt verschoben, werden alle Stützpunkte auf das Gitter eingerastet

## Geometry Checks

In the *Geometry checks* section, additional validations on a per geometry basis can be activated. Immediately after any geometry modification, failures in these checks are reported to the user in the *Geometry validation* panel. As long as a check is failing, it is not possible to save the layer. The  *Is valid* check will run basic validity checks like self intersection on geometries.

## Topology Checks

In the *Topology checks* section, additional topology validation checks can be activated. Topology checks will be executed when the user saves the layer. Check errors will be reported in the *Geometry validation* panel. As long as validation errors are present, the layer can not be saved. Topology checks are executed in the area of the bounding box of the modified features. Since other features may be present in the same area, topological errors concerning these features are reported as well as errors introduced in the current edit session.

Topology check option	Illustration
<p>The  <i>Gap</i> check will check for gaps between neighbouring polygons.</p>	
<p>The  <i>Overlap</i> check will check for overlaps between neighbouring polygons.</p>	
<p>The  <i>Missing vertex</i> check will check for shared boundaries of neighbouring polygons where one border misses a vertex which is present on the other one.</p>	

## Gap check exceptions

Sometimes it is desirable to keep gaps inside an area in a polygon layer that otherwise is fully covered by polygons. For example, a land use layer may have acceptable holes for lakes. It is possible to define areas that are ignored in the gap check. Since gaps inside these areas are allowed, we will refer to them as *Allowed Gaps* areas.

In the options for the gap checks under *Allowed Gaps*, an *Allowed Gaps layer* can be configured.

Whenever the gap check is executed, gaps which are covered by one or more polygons in the *Allowed Gaps Layer* are not reported as topology errors.

It is also possible to configure an additional *Buffer*. This buffer is applied to each polygon on the *Allowed Gaps Layer*. This makes it possible to make the tests less susceptible to small changes in the outlines at the borders of gaps.



When *Allowed Gaps* are enabled, an additional button (*Add Allowed Gap*) for detected gap errors is available in the geometry validation dock, where gaps are reported during digitizing. If the *Add Allowed Gap* button is pushed, a new polygon with the geometry of the detected gap is inserted into the *Allowed Gaps Layer*. This makes it possible to quickly flag gaps as allowed.

## Geometry validation panel

The *Geometry Validation* panel is triggered when any of the abovementioned digitizing checks finds an error. The dialog provides you with the list of errors and their description, and you can to browse the list using the keyboard arrows or dedicated arrows.

You'll need to address all the issues before you can save edits to the layer. To do so:

1. Select an error, and it's possible to:

-  *Zoom to Feature(s)*
-  *Zoom to problem*

2. Pick the usual *digitizing tools* to fix the issue.

## 16.2 Mit Attributtabelle arbeiten

The attribute table displays information on features of a selected layer. Each row in the table represents a feature (with or without geometry), and each column contains a particular piece of information about the feature. Features in the table can be searched, selected, moved or even edited.




### 16.2.1 Foreword: Spatial and non-spatial tables

QGIS allows you to load spatial and non-spatial layers. This currently includes tables supported by GDAL and delimited text, as well as the PostgreSQL, MS SQL Server, SpatiaLite and Oracle providers. All loaded layers are listed in the *Layers* panel. Whether a layer is spatially enabled or not determines whether you can interact with it on the map.

Non-spatial tables can be browsed and edited using the attribute table view. Furthermore, they can be used for field lookups. For example, you can use columns of a non-spatial table to define attribute values, or a range of values that are allowed, to be added to a specific vector layer during digitizing. Have a closer look at the edit widget in section *Eigenschaften Attributformular* to find out more.



## 16.2.2 Introducing the attribute table interface

To open the attribute table for a vector layer, activate the layer by clicking on it in the *Layerfenster*. Then, from the main *Layer* menu, choose  *Open Attribute Table*. It is also possible to right-click on the layer and choose  *Open Attribute Table* from the drop-down menu, or to click on the  *Open Attribute Table* button in the Attributes toolbar. If you prefer shortcuts, **F6** will open the attribute table. **Shift+F6** will open the attribute table filtered to selected features and **Ctrl+F6** will open the attribute table filtered to visible features.

This will open a new window that displays the feature attributes for the layer (*figure\_attributes\_table*). According to the setting in *Settings ► Options ► Data sources* menu, the attribute table will open in a docked window or a regular window. The total number of features in the layer and the number of currently selected/filtered features are shown in the attribute table title, as well as if the layer is spatially limited.

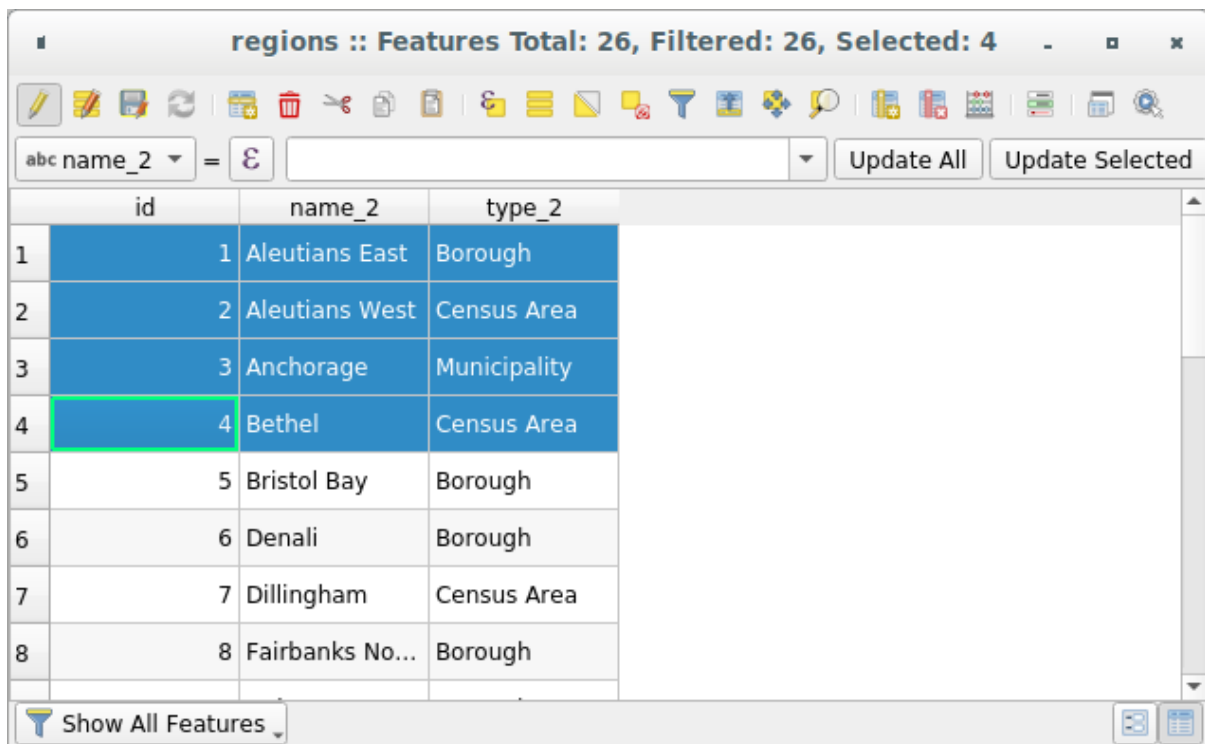









Abb. 16.67: Attributtabelle des regions Layer


















Die anderen Knöpfe im Kopf der Attributtabelle stellen die folgenden Funktionalitäten zur Verfügung:

Tab. 16.1: Available Tools


Icon	Bezeichnung	Funktion	Default Shortcut
	Toggle editing mode	Enable editing functionalities	Ctrl+E
	Toggle multi edit mode	Update multiple fields of many features	
	Save Edits	Save current modifications	
	Reload the table		
	Add feature	Add new geometryless feature	
	Delete selected features	Remove selected features from the layer	
	Cut selected features to clipboard		Strg+X

Fortsetzung auf der nächsten Seite

Tab. 16.1 – Fortsetzung der vorherigen Seite


Icon	Bezeichnung	Funktion	Default Shortcut
	Copy selected features to clipboard		Strg+C
	Paste features from clipboard	Insert new features from copied ones	Strg+V
	Select features using an Expression		
	Select All	Select all features in the layer	Strg+A
	Invert selection	Invert the current selection in the layer	Ctrl+R
	Deselect all	Deselect all features in the current layer	Strg+Umschalt+A
	Filter/Select features using form		Strg+F
	Move selected to top	Move selected rows to the top of the table	
	Pan map to the selected rows		Strg+P
	Zoom map to the selected rows		Strg+J
	New field	Add a new field to the data source	Ctrl+W
	Delete field	Remove a field from the data source	
	Organize columns	Show/hide fields from the attribute table	
	Open field calculator	Update field for many features in a row	Ctrl+I
	Conditional formatting	Enable table formatting	
	Dock attribute table	Allows to dock/undock the attribute table	
	Actions	Lists the actions related to the layer	

**Bemerkung:** Depending on the format of the data and the GDAL library built with your QGIS version, some tools may not be available.


Below these buttons is the Quick Field Calculation bar (enabled only in *edit mode*), which allows to quickly apply calculations to all or part of the features in the layer. This bar uses the same *expressions* as the  Field Calculator (see *Editieren Attributwerte*).

## Table view vs Form view

QGIS provides two view modes to easily manipulate data in the attribute table:




- The  Table view, displays values of multiple features in a tabular mode, each row representing a feature and each column a field. A right-click on the column header allows you to *configure the table display* while a right-click on a cell provides *interaction with the feature*.

The attribute table supports Shift+Mouse Wheel scrolling in table view mode to switch between vertical and horizontal scrolling movements. This can also be achieved replacing the mouse with the trackpad on macOS.

- The  Form view shows *feature identifiers* in a first panel and displays only the attributes of the clicked identifier in the second one. There is a pull-down menu at the top of the first panel where the „identifier“ can be specified using an attribute (*Column preview*) or an *Expression*. The pull-down also includes the last 10 expressions for re-use. Form view uses the layer fields configuration (see *Eigenschaften Attributformular*).

You can browse through the feature identifiers with the arrows on the bottom of the first panel. The features attributes update in the second panel as you go. It's also possible to identify or move to the active feature in the

map canvas with pushing down any of the button at the bottom:

-  Highlight current feature if visible in the map canvas
-  Automatically pan to current feature
-  Zoom to current feature

You can switch from one mode to the other by clicking the corresponding icon at the bottom right of the dialog.

You can also specify the *Default view* mode at the opening of the attribute table in *Settings ► Options ► Data Sources* menu. It can be ,Remember last view', ,Table view' or ,Form view'.

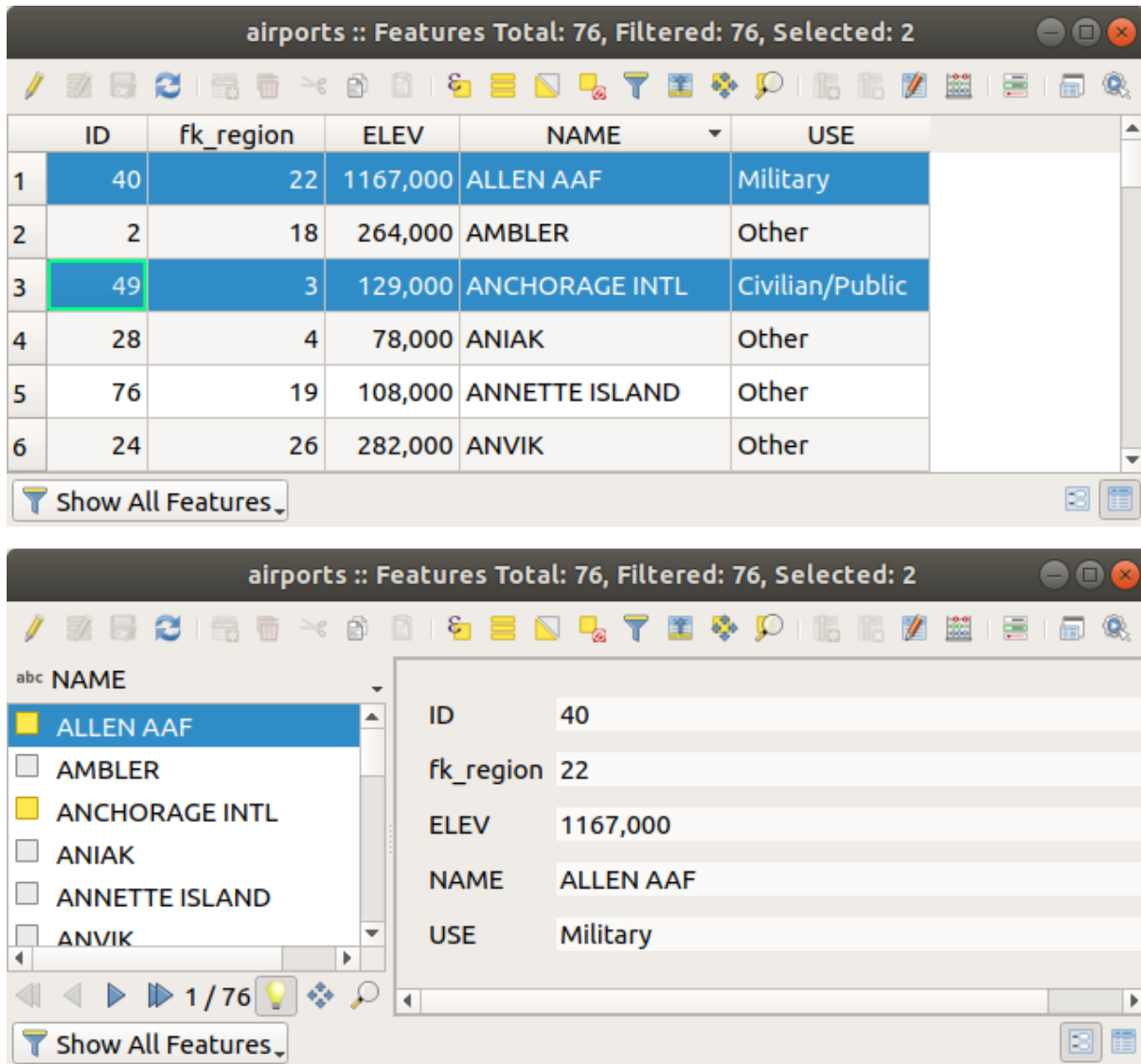


Abb. 16.68: Attribute table in table view (top) vs form view (bottom)

## Configuring the columns

Right-click in a column header when in table view to have access to tools that help you control:

- the *column(s) size*
- the *column(s) visibility and order*
- the *sort order of the data*


## Resizing columns widths

Columns width can be set through a right-click on the column header and select either:

- *Set width...* to enter the desired value. By default, the current value is displayed in the widget
- *Set all column widths...* to the same value
- *Autosize* to resize at the best fit the column.
- *Autosize all columns*

A column size can also be changed by dragging the boundary on the right of its heading. The new size of the column is maintained for the layer, and restored at the next opening of the attribute table.


## Hiding and organizing columns and enabling actions

By right-clicking in a column header, you can choose to *Hide column* from the attribute table (in „table view“ mode). For more advanced controls, press the  *Organize columns...* button from the dialog toolbar or choose *Organize columns...* in a column header contextual menu. In the new dialog, you can:

- check/uncheck columns you want to show or hide: a hidden column will disappear from every instances of the attribute table dialog until it is actively restored.
- drag-and-drop items to reorder the columns in the attribute table. Note that this change is for the table rendering and does not alter the fields order in the layer datasource
- add a new virtual *Actions* column that displays in each row a drop-down box or a button list of enabled actions. See *Eigenschaften Aktionen* for more information about actions.

## Sorting columns

The table can be sorted by any column, by clicking on the column header. A small arrow indicates the sort order (downward pointing means descending values from the top row down, upward pointing means ascending values from the top row down). You can also choose to sort the rows with the *Sort...* option of the column header context menu and write an expression. E.g. to sort the rows using multiple columns you can write `concat (col0, col1)`.

In form view, features identifier can be sorted using the  *Sort by preview expression* option.

---

### **Tipp: Sorting based on columns of different types**


Trying to sort an attribute table based on columns of string and numeric types may lead to unexpected result because of the `concat ("USE", "ID")` expression returning string values (ie, 'Borough105' < 'Borough6'). You can workaround this by using eg `concat ("USE", lpad ("ID", 3, 0))` which returns 'Borough105' > 'Borough006'.



---

## Formatting of table cells using conditions

Conditional formatting settings can be used to highlight in the attribute table features you may want to put a particular focus on, using custom conditions on feature's:

- geometry (e.g., identifying multi-parts features, small area ones or in a defined map extent...);
- or field value (e.g., comparing values to a threshold, identifying empty cells, duplicates, ...).

You can enable the conditional formatting panel clicking on  Conditional formatting button at the top right of the attributes window in table view (not triggered in form view).

The new panel allows user to add new rules to format rendering of  *Field* or  *Full row*. Adding new rule opens a form to define:

- the name of the rule;
- a condition using any of the *expression builder* functions;
- the formatting: it can be chosen from a list of predefined formats or created based on properties like:
  - background and text colors;
  - use of icon;
  - bold, italic, underline, or strikeout;
  - Schriftart.

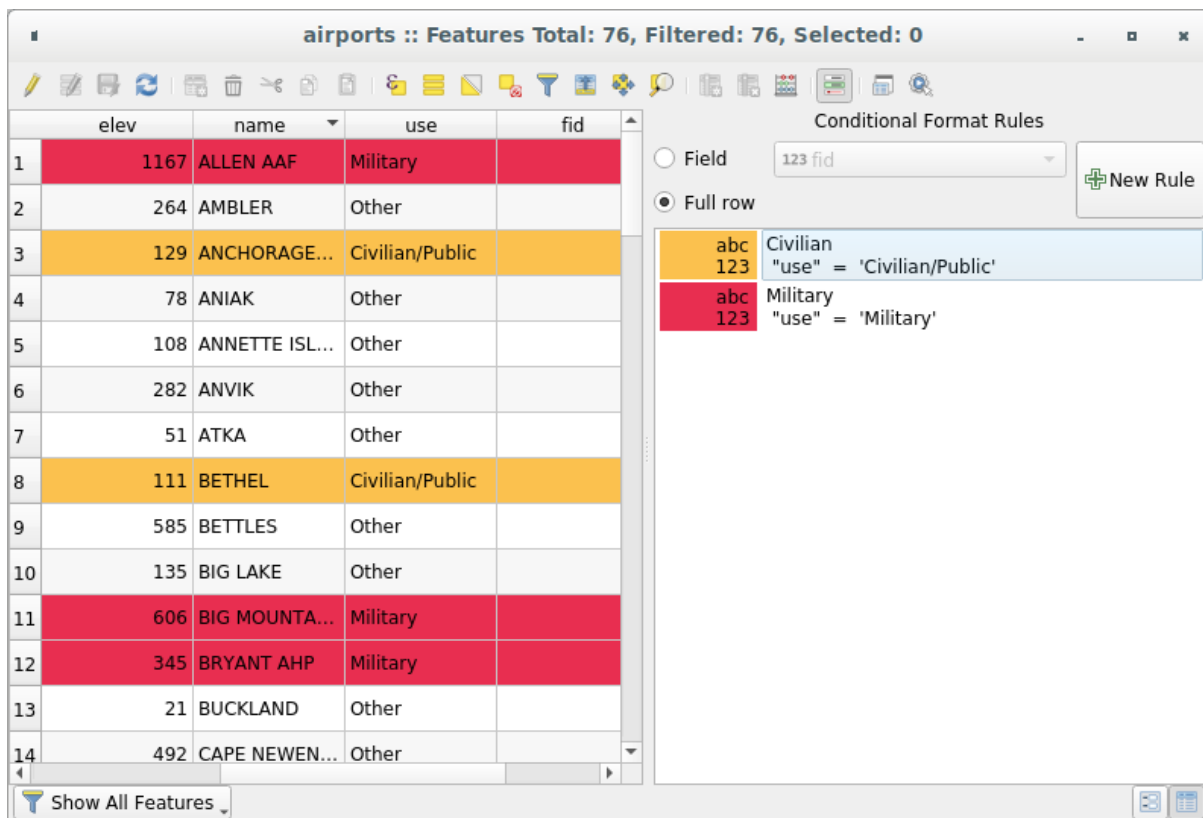


Abb. 16.69: Bedingte Formatierungsregeln für Attributtabellen

## 16.2.3 Interacting with features in an attribute table

### Selecting features






In table view, each row in the attribute table displays the attributes of a unique feature in the layer. Selecting a row selects the feature and likewise, selecting a feature in the map canvas (in case of geometry enabled layer) selects the row in the attribute table. If the set of features selected in the map canvas (or attribute table) is changed, then the selection is also updated in the attribute table (or map canvas) accordingly.

Zeilen können ausgewählt werden, indem Sie auf die Zeilennummer links neben der Zeile klicken. **Mehrere Zeilen** können ausgewählt werden, indem die `Strg` Taste während der Auswahl gedrückt wird. Eine **kontinuierliche Auswahl** ist möglich, indem Sie bei der Selektion die `Umschalt` Taste gedrückt halten, während Sie die Zeilennummern auswählen. Alle Zeilen zwischen der aktuell ausgewählten Zeile und der Mausfeilposition werden dadurch selektiert. Bewegt man den Mauszeiger in der Attributtabelle indem man in eine Zelle in der Tabelle klickt ändert die Zeilenauswahl nicht. Verändert man die Auswahl im Kartenfenster bewegt sich der Mauszeiger in der Attributtabelle nicht.

In form view of the attribute table, features are by default identified in the left panel by the value of their displayed field (see [Eigenschaften Anzeige](#)). This identifier can be replaced using the drop-down list at the top of the panel, either by selecting an existing field or using a custom expression. You can also choose to sort the list of features from the drop-down menu.

Click a value in the left panel to display the feature's attributes in the right one. To select a feature, you need to click inside the square symbol at the left of the identifier. By default, the symbol turns into yellow. Like in the table view, you can perform multiple feature selection using the keyboard combinations previously exposed.





Beyond selecting features with the mouse, you can perform automatic selection based on feature's attribute using tools available in the attribute table toolbar, such as (see section [Automatic selection](#) and following one for more information and use case):





-  *Select By Expression...*
-  *Select Features By Value...*
-  *Deselect All Features from the Layer*
-  *Select All Features*
-  *Invert Feature Selection.*

It is also possible to select features using the [Filtering and selecting features using forms](#).

### Filtering features

Once you have selected features in the attribute table, you may want to display only these records in the table. This can be easily done using the *Show Selected Features* item from the drop-down list at the bottom left of the attribute table dialog. This list offers the following filters:

-  *Show All Features*
-  *Show Selected Features* - same as using *Open Attribute Table (Selected Features)* from the *Layer* menu or the *Attributes Toolbar* or pressing `Shift+F6`
-  *Show Features visible on map* - same as using *Open Attribute Table (Visible Features)* from the *Layer* menu or the *Attributes Toolbar* or pressing `Ctrl+F6`
-  *Show Features with Failing Constraints* - features will be filtered to only show the ones which have failing [constraints](#). Depending on whether the unmet constraint is hard or soft, failing field values are displayed in respectively dark or light orange cells.

-  *Show Edited and New Features* - same as using *Open Attribute Table (Edited and New Features)* from the *Layer* menu or the *Attributes Toolbar*
- *Field Filter* - allows the user to filter based on value of a field: choose a column from a list, type or select a value and press Enter to filter. Then, only the features matching `num_field = value` or `string_field ilike '%value%'` expression are shown in the attribute table. You can check  *Case sensitive* to be less permissive with strings.
-  *Advanced filter (Expression)* - Opens the expression builder dialog. Within it, you can create *complex expressions* to match table rows. For example, you can filter the table using more than one field. When applied, the filter expression will show up at the bottom of the form.
-  *Stored filter expressions* ►: a shortcut to *saved expressions* frequently used for filtering your attribute table.

It is also possible to *filter features using forms*.

---

**Bemerkung:** Filtering records out of the attribute table does not filter features out of the layer; they are simply momentarily hidden from the table and can be accessed from the map canvas or by removing the filter. For filters that do hide features from the layer, use the *Query Builder*.

---



---

**Tipp:** Update datasource filtering with Show Features Visible on Map



---

When for performance reasons, features shown in attribute table are spatially limited to the canvas extent at its opening (see *Data Source Options* for a how-to), selecting *Show Features Visible on Map* on a new canvas extent updates the spatial restriction.

---


## Storing filter expressions

Expressions you use for attribute table filtering can be saved for further calls. When using *Field Filter* or *Advanced Filter (expression)* entries, the expression used is displayed in a text widget in the bottom of the attribute table dialog.

Press the  Save expression with text as name next to the box to save the expression in the project. Pressing the drop-down menu next to the button allows to save the expression with a custom name (*Save expression as...*). Once a saved expression is displayed, the  button is triggered and its drop-down menu allows you to *Edit the expression* and name if any, or *Delete stored expression*.

Saved filter expressions are saved in the project and available through the *Stored filter expressions* menu of the attribute table. They are different from the *user expressions*, shared by all projects of the active user profile.

## Filtering and selecting features using forms

Clicking the  Filter/Select features using form or pressing `Ctrl+F` will make the attribute table dialog switch to form view and replace each widget with its search variant.

From this point onwards, this tool functionality is similar to the one described in *Select Features By Value*, where you can find descriptions of all operators and selecting modes.

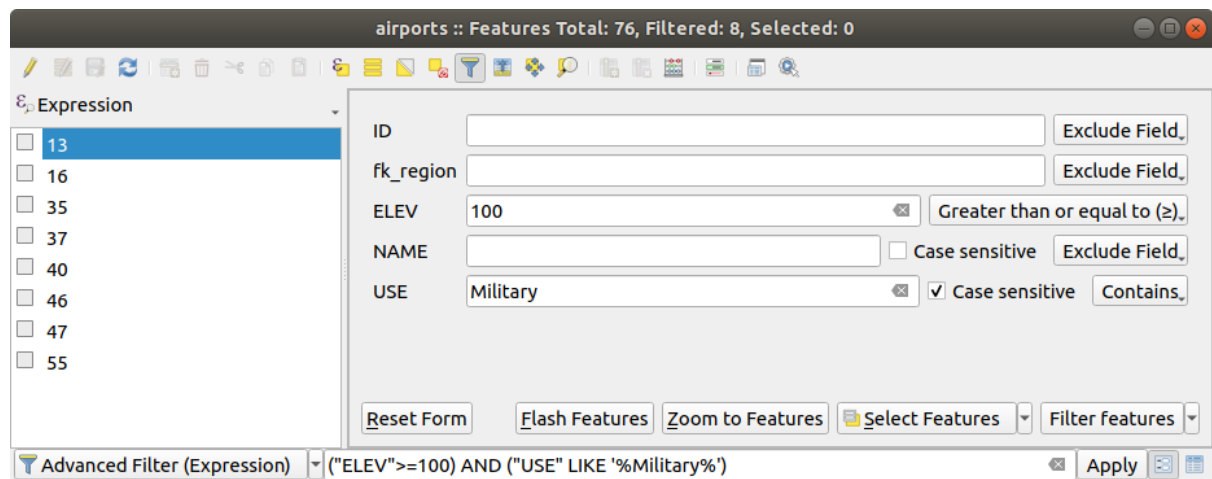


Abb. 16.70: Attribute table filtered by the filter form

When selecting / filtering features from the attribute table, there is a *Filter features* button that allows defining and refining filters. Its use triggers the *Advanced filter (Expression)* option and displays the corresponding filter expression in an editable text widget at the bottom of the form.

If there are already filtered features, you can refine the filter using the drop-down list next to the *Filter features* button. The options are:

- *Filter within („AND“)*
- *Extend filter („OR“)*

To clear the filter, either select the *Show all features* option from the bottom left pull-down menu, or clear the expression and click *Apply* or press **Enter**.

## 16.2.4 Using action on features

Users have several possibilities to manipulate feature with the contextual menu like:

- *Select all* (**Ctrl+A**) the features;
- Copy the content of a cell in the clipboard with *Copy cell content*;
- *Zoom to feature* without having to select it beforehand;
- *Pan to feature* without having to select it beforehand;
- *Flash feature*, to highlight it in the map canvas;
- *Open form*: it toggles attribute table into form view with a focus on the clicked feature.



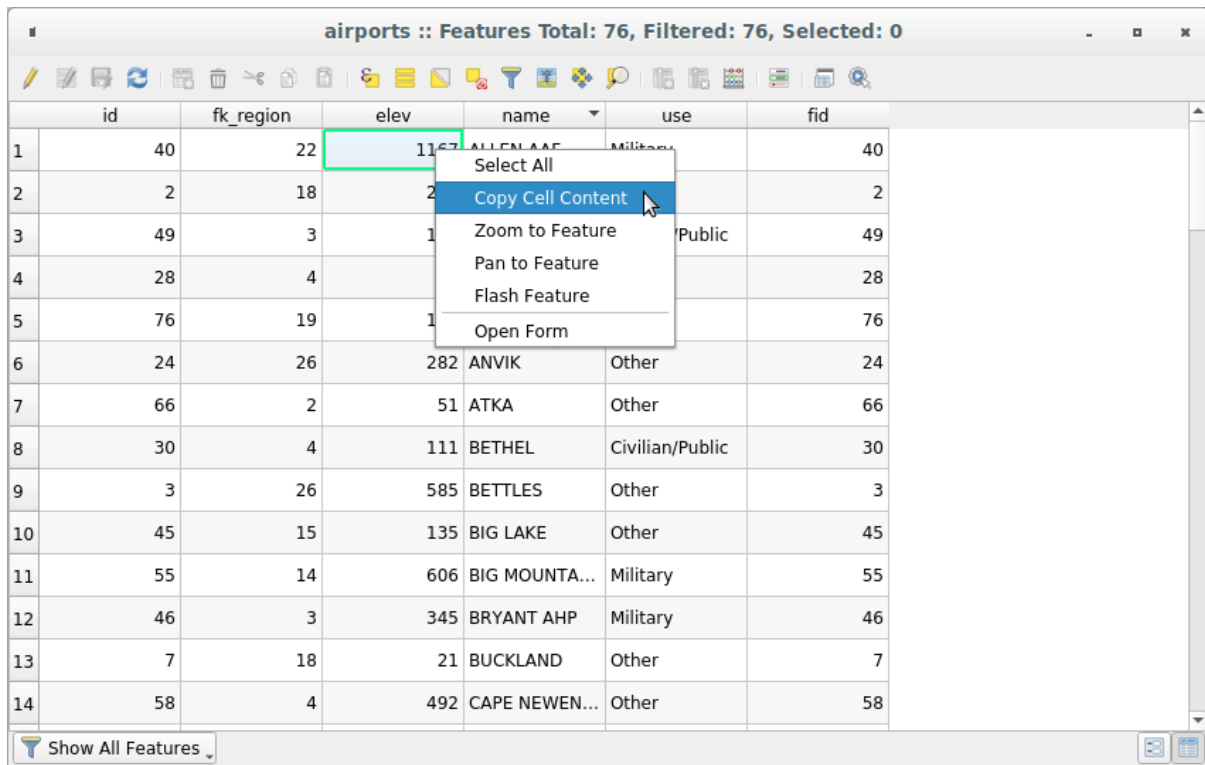



Abb. 16.71: Copy cell content button


If you want to use attribute data in external programs (such as Excel, LibreOffice, QGIS or a custom web application), select one or more row(s) and use the  Copy selected rows to clipboard button or press **Ctrl+C**.

In *Settings ► Options ► Data Sources* menu you can define the format to paste to with *Copy features as* dropdown list:

- Plain text, no geometry,
- Plain text, WKT geometry,
- GeoJSON

You can also display a list of actions in this contextual menu. This is enabled in the *Layer properties ► Actions* tab. See [Eigenschaften Aktionen](#) for more information on actions.

### Saving selected features as new layer


The selected features can be saved as any OGR-supported vector format and also transformed into another coordinate reference system (CRS). In the contextual menu of the layer, from the *Layers* panel, click on *Export ► Save selected features as...* to define the name of the output dataset, its format and CRS (see section [Creating new layers from an existing layer](#)). You'll notice that  *Save only selected features* is checked. It is also possible to specify GDAL creation options within the dialog.

## 16.2.5 Editiere Attributwerte

Editing attribute values can be done by:

- typing the new value directly in the cell, whether the attribute table is in table or form view. Changes are hence done cell by cell, feature by feature;
- using the *field calculator*: update in a row a field that may already exist or to be created but for multiple features. It can be used to create virtual fields;
- using the quick field *calculation bar*: same as above but for only existing field;
- or using the *multi edit* mode: update in a row multiple fields for multiple features.

### Using the Field Calculator

The  Field Calculator button in the attribute table allows you to perform calculations on the basis of existing attribute values or defined functions, for instance, to calculate length or area of geometry features. The results can be used to update an existing field, or written to a new field (that can be a *virtual* one).

The field calculator is available on any layer that supports edit. When you click on the field calculator icon the dialog opens (see [Abb. 16.72](#)). If the layer is not in edit mode, a warning is displayed and using the field calculator will cause the layer to be put in edit mode before the calculation is made.

Based on the *Expression Builder* dialog, the field calculator dialog offers a complete interface to define an expression and apply it to an existing or a newly created field. To use the field calculator dialog, you must select whether you want to:

1. apply calculation on the whole layer or on selected features only
2. create a new field for the calculation or update an existing one.

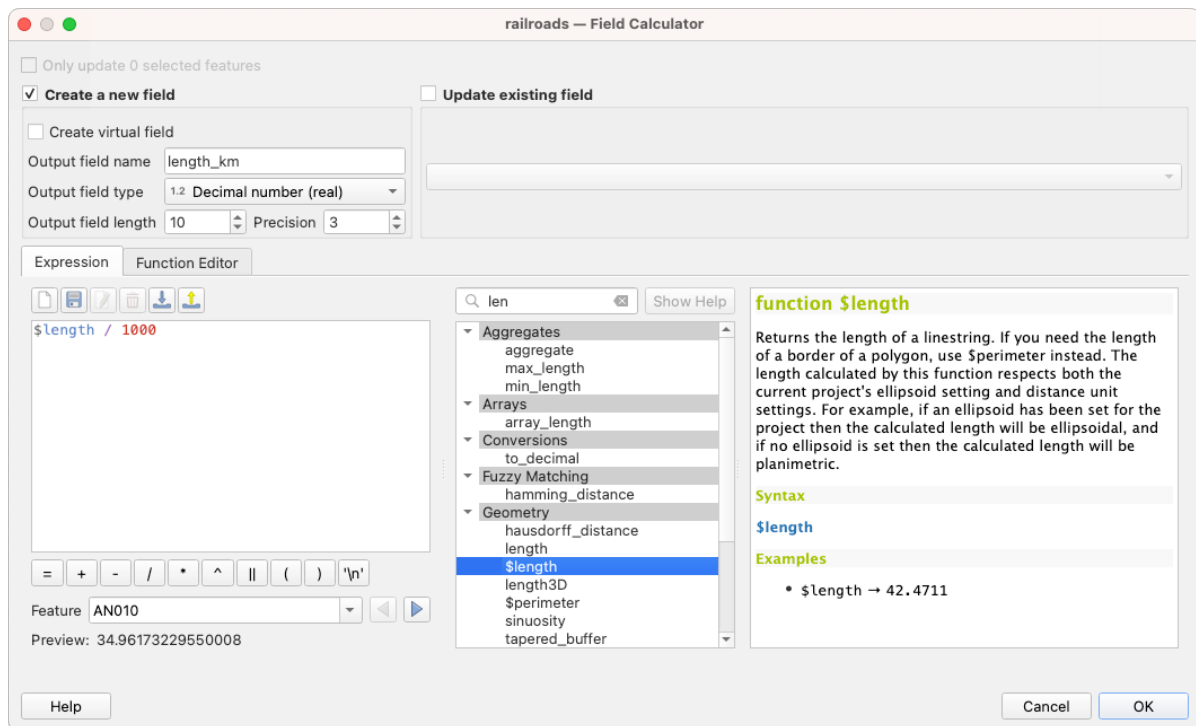


Abb. 16.72: Feldrechner

If you choose to add a new field, you need to enter a field name, a field type (integer, real, date or string) and if needed, the total field length and the field precision. For example, if you choose a field length of 10 and a field precision of 3, it means you have 7 digits before the dot, and 3 digits for the decimal part.

Ein kurzes Beispiel zeigt wie der Feldrechner arbeitet, wenn Sie den Reiter *Ausdruck* verwenden. Wir wollen die Länge des ‚railroads‘ Layers aus dem QGIS Beispieldatensatz in km berechnen:

1. Laden Sie das Shape `railroads.shp` in QGIS und öffnen Sie die den Dialog Attributtabelle öffnen.
2. Klicken Sie auf Bearbeitungsmodus umschalten und öffnen Sie den Feldrechner Dialog.
3. Wählen Sie das *Neues Feld anlegen* Kontrollkästchen um die Berechnungen in ein neues Feld zu speichern.
4. Set *Output field name* to `length_km`
5. Select `Decimal number (real)` as *Output field type*
6. Set the *Output field length* to 10 and the *Precision* to 3
7. Double click on `$length` in the *Geometry* group to add the length of the geometry into the Field calculator expression box (you will begin to see a preview of the output, up to 60 characters, below the expression box updating in real-time as the expression is assembled).
8. Complete the expression by typing `/ 1000` in the Field calculator expression box and click *OK*.
9. You can now find a new `length_km` field in the attribute table.

## Creating a Virtual Field


A virtual field is a field based on an expression calculated on the fly, meaning that its value is automatically updated as soon as an underlying parameter changes. The expression is set once; you no longer need to recalculate the field each time underlying values change. For example, you may want to use a virtual field if you need area to be evaluated as you digitize features or to automatically calculate a duration between dates that may change (e.g., using `now()` function).

### Bemerkung: Virtuelle Felder benutzen

- Virtuelle Felder sind nicht dauerhaft in den Layerattributen, was bedeutet, dass sie in der Projektdatei gespeichert und verfügbar sind, in der sie gespeichert wurden.
- A field can be set virtual only at its creation. Virtual fields are marked with a purple background in the fields tab of the layer properties dialog to distinguish them from regular physical or joined fields. Their expression can be edited later by pressing the expression button in the Comment column. An expression editor window will be opened to adjust the expression of the virtual field.

## Using the Quick Field Calculation Bar

While Field calculator is always available, the quick field calculation bar on top of the attribute table is only visible if the layer is in edit mode. Thanks to the expression engine, it offers a quicker access to edit an already existing field:

1. Select the field to update in the drop-down list.
2. Fill the textbox with a value, an expression you directly write or build using the  expression button.
3. Click on *Update All*, *Update Selected* or *Update Filtered* button according to your need.

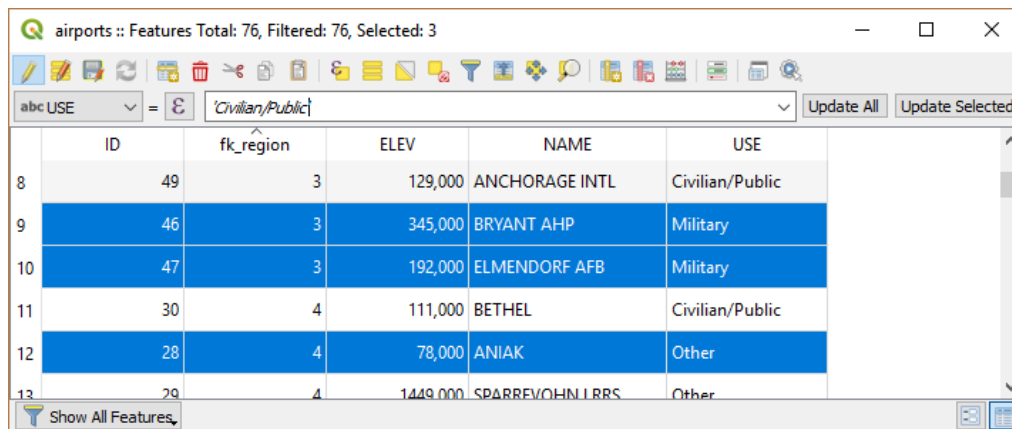




Abb. 16.73: Quick Field Calculation Bar





## Editing multiple fields

Unlike the previous tools, multi edit mode allows multiple attributes of different features to be edited simultaneously. When the layer is toggled to edit, multi edit capabilities are accessible:

- using the  Toggle multi edit mode button from the toolbar inside the attribute table dialog;
- or selecting *Edit* ►  *Modify attributes of selected features* menu.

**Bemerkung:** Unlike the tool from the attribute table, hitting the *Edit ► Modify Attributes of Selected Features* option provides you with a modal dialog to fill attributes changes. Hence, features selection is required before execution.

In order to edit multiple fields in a row:

1. Select the features you want to edit.
2. From the attribute table toolbar, click the  button. This will toggle the dialog to its form view. Feature selection could also be made at this step.
3. At the right side of the attribute table, fields (and values) of selected features are shown. New widgets appear next to each field allowing for display of the current multi edit state:
  -  The field contains different values for selected features. It's shown empty and each feature will keep its original value. You can reset the value of the field from the drop-down list of the widget.
  -  All selected features have the same value for this field and the value displayed in the form will be kept.
  -  The field has been edited and the entered value will be applied to all the selected features. A message appears at the top of the dialog, inviting you to either apply or reset your modification.

Clicking any of these widgets allows you to either set the current value for the field or reset to original value, meaning that you can roll back changes on a field-by-field basis.

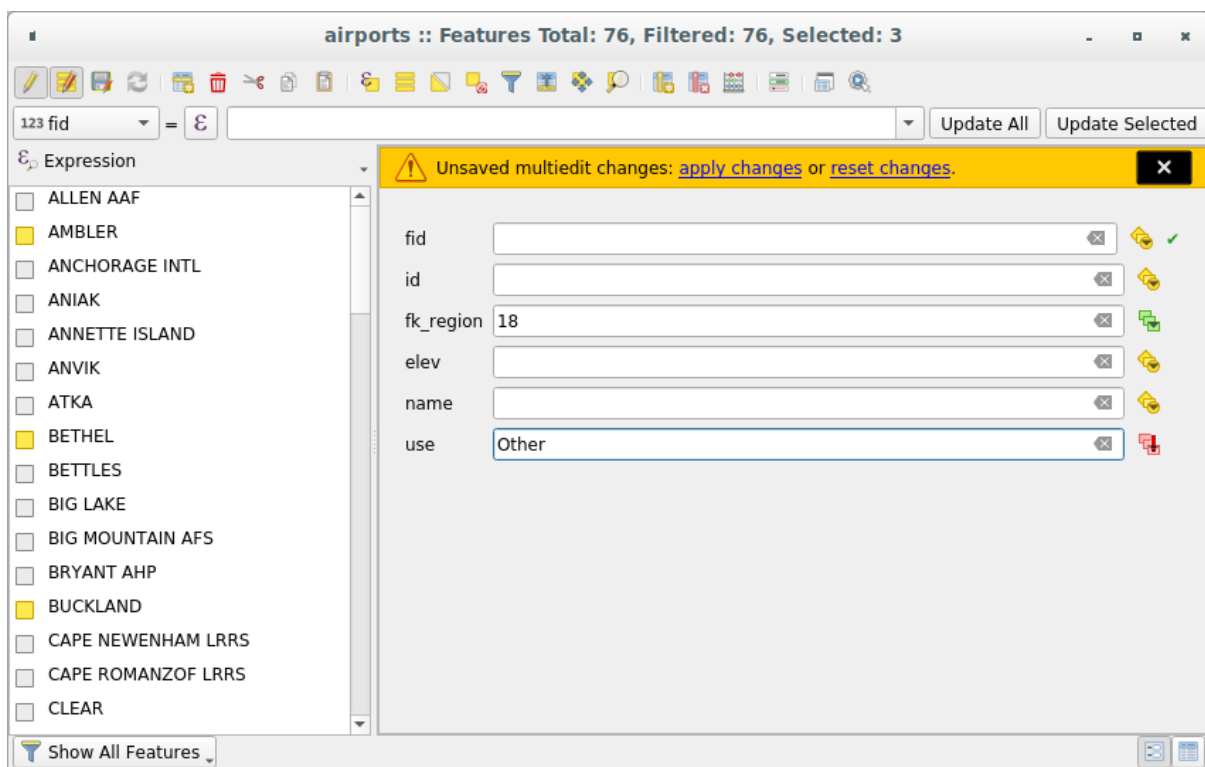



Abb. 16.74: Editing fields of multiple features

4. Make the changes to the fields you want.
5. Click on **Apply changes** in the upper message text or any other feature in the left panel.

Changes will apply to **all selected features**. If no feature is selected, the whole table is updated with your changes. Modifications are made as a single edit command. So pressing  Undo will rollback the attribute changes for all


selected features at once.

---

**Bemerkung:** Multi edit mode is only available for auto generated and drag and drop forms (see *Customizing a form for your data*); it is not supported by custom ui forms.

---

### 16.2.6 Exploring features attributes through the Identify Tool

The  *Identify features* tool can be used to display all attributes of a feature in the map canvas. It is a quick way to view and verify all data without having to search for it in the attribute table.

To use the *Identify features* tool for vector layers, follow these steps:

1. Select the vector layer in the Layers panel.
2. Click on the *Identify features* tool in the toolbar or press `Ctrl+Shift+I`.
3. Click on a feature in the map view.

The *Identify results* panel will display different features information depending on the layer type. There are two columns in the panel, on the left side you can see *Feature* and on the right side *Value*. Under the *Feature* column, panel will display following information:

- **Derived** section - those are the information calculated or derived from other information in the layer. For example, the area of a polygon or the length of a line. General information that can be found in this section:
  - Depending on the geometry type, cartesian measurements of length, perimeter, or area in the layer's CRS units. For 3D line vectors, the cartesian line length is available.
  - Depending on the geometry type and if an ellipsoid is set in the *Project Properties* dialog (*General* → *Measurements*), ellipsoidal values of length, perimeter, or area using the specified units.
  - The count of geometry parts in the feature and the number of the part clicked.
  - The count of vertices in the feature.
- Coordinate information that can be found in this section:
  - X and Y coordinate values of the clicked point.
  - The number of the closest vertex to the clicked point.
  - X and Y coordinate values of the closest vertex.
  - If you click on a curved segment, the radius of that section is also displayed.
- **Data attributes:** This is the list of attribute fields and values for the feature that has been clicked.
- information about the related child feature if you defined a *relation*:
  - the name of the relation
  - the entry in reference field, e.g. the name of the related child feature
  - **Actions:** lists actions defined in the layer's properties dialog (see *Eigenschaften Aktionen*) and the default action is `View feature form`.
  - **Data attributes:** This is the list of attributes fields and values of the related child feature.

## 16.2.7 Storing and fetching an external resource

A field may target a resource stored on an external storage system. Attribute forms can be configured so they act as a client to an external storage system in order to store and fetch those resources, on users demand, directly from the forms.

### Configuring an external storage

In order to setup an external storage, you have to first configure it from the vector *attribute form properties* and select the *Attachment* widget.



Abb. 16.75: Editing a WebDAV external storage for a given field

From the *Attachment* widget, you have to first select the *Storage type*:

- *Select Existing File*: The target URL already exists. When you select a resource, no store operation is achieved, the attribute is simply updated with the URL.
- *Simple Copy*: Stores a copy of the resource on a file disk destination (which could be a local or network shared file system) and the attribute is updated with the path to the copy.
- *WebDAV Storage*: The resource is pushed to a HTTP server supporting the [WebDAV](#) protocol and the attribute is updated with its URL. [Nextcloud](#), [Pydio](#) or other file hosting software support this protocol.
- *AWS S3*: The resource is pushed to a server supporting [AWS Simple Storage Service](#) protocol and the attribute is updated with its URL. Amazon Web Service and [MinIO](#) hosting software support this protocol.

Then, you have to set up the *Store URL* parameter, which provides the URL to be used when a new resource needs to be stored. It's possible to set up an expression using the *data defined override widget* in order to have specific values according to feature attributes.

The variable `@selected_file_path` could be used in that expression and represent the absolute file path of the user selected file (using the file selector or drag'n drop).

---

**Bemerkung:** Using the **WebDAV** or **AWS S3** external storage, if the URL ends with a „/“, it is considered as a folder and the selected file name will be appended to get the final URL.

---

If the external storage system needs to, it's possible to configure an *authentication*.

---

**Bemerkung:** To use the **AWS S3** external storage, you must use an **AWS S3** authentication type.

---

## Using an external storage

Once configured, you can select a local file using the button ... when editing a feature's attribute. Depending on the configured *storage type*, the file will be stored on the external storage system (except if *Select existing file* has been selected) and the field will be updated with the new resource URL.

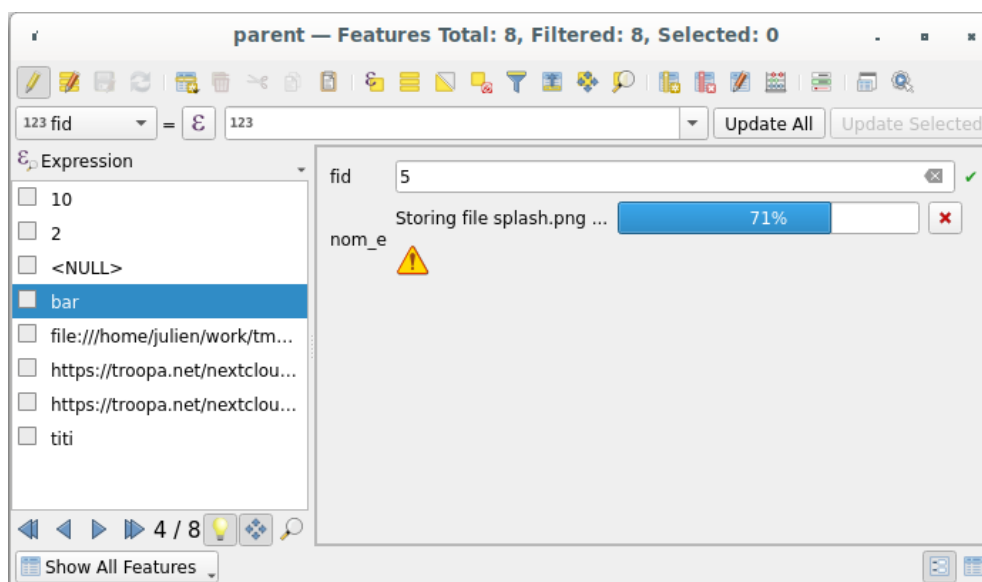




Abb. 16.76: Storing a file to a WebDAV external storage

---

**Bemerkung:** User can also achieve the same result if he drags and drops a file on the whole attachment widget.

---

Use the  button to abort the storing process. It's possible to configure a viewer using the *Integrated document viewer* so the resource will be automatically fetched from the external storage system and displayed directly below the URL. The above  icon indicates that the resource cannot be fetched from the external storage system. In that case, more details might appear in the *Log Messages Panel*.

## 16.3 Editierfunktionen

QGIS has various capabilities for editing OGR, SpatiaLite, PostGIS, MS SQL Server and Oracle Spatial vector layers and tables. They can be of 2D or 3D geometry type.

---

**Bemerkung:** Die Vorgehensweise GRASS Layer zu bearbeiten ist anders - siehe Abschnitt *Digitalisieren und Editieren eines GRASS Vektorlayers* für Details.

---



**Achtung: Zeitgleiches Editieren**

QGIS does not track if somebody else is editing the same feature at the same time as you are. The last person to save the edits wins.

**Tipp: Bearbeitungen validieren**

Sie können die fortwährende Validierung für einzelne Layer über den Tab *Layereigenschaften* ► *Digitalisieren* aktivieren. Mehr dazu unter *Eigenschaften Digitalisieren*.

### 16.3.1 Einrasttoleranz und Suchradius einstellen

Unter dem Menüpunkt *Einstellungen* ► *Optionen...* ► *Digitalisieren* bietet QGIS eine Anzahl von Parametern, mit der Sie das Standardverhalten der Editierwerkzeuge einstellen können. Weitere Informationen finden Sie unter *Digitalisieren*.

Damit Sie die Geometrien von Vektorlayern optimal und akkurat bearbeiten können, ist es notwendig, dass Sie geeignete Werte für die Einrasttoleranz und den Suchradius für die Stützpunkte eines Objekts einstellen. Die Gruppe *Einrasten* bietet entsprechende Optionen, nämlich die Einstellung der Einrasttoleranz und des Suchradius für die Stützpunktbearbeitung.

- **Einrasttoleranz:** Wenn Sie einen neuen Stützpunkt digitalisieren oder einen vorhandenen verschieben, ist die Einrasttoleranz diejenige Distanz, innerhalb derer QGIS nach dem nächsten Stützpunkt oder Segment sucht, um Ihren Stützpunkt damit zu verbinden. Wenn Sie sich außerhalb der Einrasttoleranz befinden, wird QGIS den Stützpunkt an derjenigen Stelle platzieren, an der Sie die Maustaste loslassen, anstatt ihn an einem vorhandenen Stützpunkt oder Segment einzurasten.


Die Einstellung für die Einrasttoleranz beeinflusst alle Werkzeuge, die mit Einrasten arbeiten und wird standardmäßig auf neue Layer und Projekte angewendet. Auf Layerebene kann sie jedoch geändert werden (siehe *Optionen für das Einrasten und Digitalisieren*).

- **Suchradius:** Der *Suchradius für Stützpunktbearbeitung* ist diejenige Distanz, die QGIS benutzt, um nach dem zu selektierenden Stützpunkt zu suchen, wenn Sie in die Karte klicken. Befindet sich der Klick nicht innerhalb des Suchradius, wird QGIS keinen Stützpunkt für die Bearbeitung auswählen.

Einrasttoleranz und Suchradius stellen Sie in Karteneinheiten oder Pixeln ein, so dass Sie vielleicht experimentieren müssen, um sie passend einzustellen. Wenn Sie eine zu große Toleranz angeben, fängt QGIS vielleicht den falschen Stützpunkt, besonders wenn Sie es mit einer großen Anzahl von Stützpunkten innerhalb kurzer Distanz zu tun haben. Stellen Sie den Suchradius zu klein ein, wird nichts zum Verschieben gefunden.

### 16.3.2 Optionen für das Einrasten und Digitalisieren

Global *snapping and digitizing settings* (snapping mode, tolerance value, and units...) can be overridden in the project from the *Project* ► *Snapping Options...* menu. In the *Snapping and Digitizing Options*, you can also configure some other properties (snapping layers, scale limit, topology...) The *Snapping Toolbar* gives access to most of these features.







Standardmäßig ist Einrasten in einem Projekt deaktiviert, bis Sie auf den Knopf  *Einrasten einschalten* oder auf die Taste **S** drücken. Sie können Einrastmodus, Toleranz und Einheiten auch in dieser Werkzeugleiste konfigurieren.

## Einrastoptionen

Sie haben drei Möglichkeiten, um die Layer, auf die eingrastet werden soll, auszuwählen:

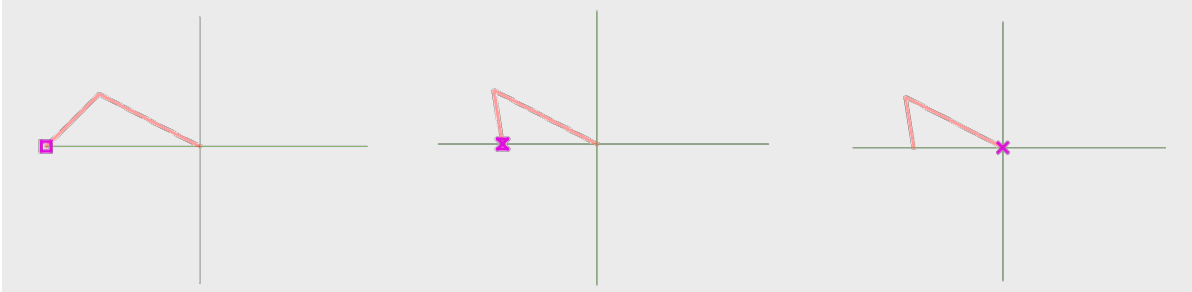
- *Alle Layer*: eine schnelle und einfache Einstellung für alle sichtbaren Layer im Projekt, so dass der Zeiger alle Stützpunkte und/oder Segmente fängt. In den meisten Fällen reicht es aus, diesen Fangmodus zu verwenden; passen Sie aber auf, wenn Sie diesen Modus in Projekten mit vielen Vektorlayern benutzen, weil es die Anwendung deutlich verlangsamen kann.
- *Aktueller Layer*: nur der aktuelle Layer wird genutzt, eine praktische Möglichkeit die Topologie des Layers abzusichern, der bearbeitet wird.
- Mit *Erweiterte Konfiguration*: können Sie für jeden Layer den Typ, die Toleranz und die Einheiten des Einrastens, die Überlappung sowie den Einrastmaßstab einschalten und anpassen (siehe Abb. 16.77). Wenn Sie einen Layer editieren und möchten seine Stützpunkte an einem anderen Layer einrasten, stellen Sie bitte sicher, dass der andere Layer angehakt ist und erhöhen Sie die Einrasttoleranz. Ist ein Layer in den Projekteinrasteeinstellungen nicht angehakt, können Sie auf ihn nicht einrasten.

When moving or creating vertex, you can opt for the following snapping modes:

-  *Vertex*
-  *Segment*: snaps along a line or a polygon perimeter. If topological editing is enabled, then a new vertex is added at the snapping location.
-  *Area*: guarantees that the snap point lies anywhere on a polygon's area, not necessarily on its boundary
-  *Centroid*: snaps to the centroid of the geometry of a feature. In case of a multipart geometry, the target point may be distinct from the existing parts.
-  *Middle of Segments* on line or polygon feature
-  *Line Endpoints*: snaps to the first or last vertex of every part of a line or polygon feature.

QGIS wird die fangbaren Punkte unterschiedlich darstellen, je nachdem welcher Einrastmodus aktiv ist:

Tab. 16.2: Snapping icons

		
Einrasten auf einen Stützpunkt: Quadrat	Einrasten auf Segment: Sanduhr	Einrasten auf Schnittpunkte: Kreuz

Sie können die Farbe dieser Symbole in den Einstellungen im Bereich *Digitalisieren* ändern.

Die Toleranz können Sie entweder in Karteneinheiten des Projekts oder in Pixel einstellen. Der Vorteil von Pixel ist, dass damit der Fangradius bei unterschiedlichen Kartenmassstäben konstant bleibt. 10 bis 12 px ist normalerweise ein guter Wert, der aber von der DPI-Zahl Ihres Bildschirms abhängig ist. Mit Karteneinheiten benutzen Sie für die Toleranz Realwelteinheiten. Wenn Sie z.B. eine Mindestdistanz zwischen Ihren Elementen sicherstellen wollen, hilft Ihnen diese Einstellung dabei, die Stützpunkte nicht zu dicht beieinander zu setzen.

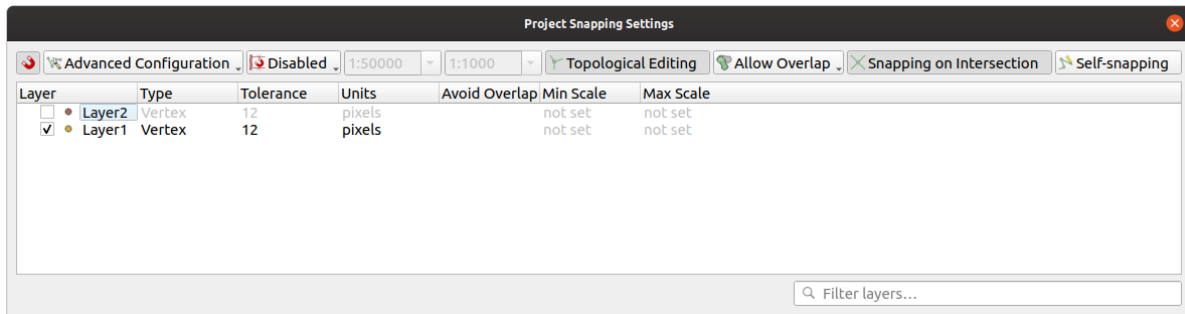




Abb. 16.77: Einrasteeinstellungen (Erweiterte Konfiguration)

**Bemerkung:** In der Voreinstellung können nur sichtbare Objekte gefangen werden (d.h. Objekte, deren Stil dargestellt wird, mit Ausnahme von Layern, deren Symbolisierung „Keine Symbole“ ist). Sie können das Fangen unsichtbarer Objekte durch Anhängen von  *Einrasten auf verborgene Objekte aktivieren* unter *Einstellungen* ► *Optionen* ► *Digitalisieren* anschalten.

### Tipp: Einrasten voreinstellen

Sie können das Einrasten als Voreinstellung für alle neuen Projekte im Reiter *Einstellungen* ► *Optionen* ► *Digitalisierung* anschalten. Sie können auch den Einrastmodus, die Einrasttoleranz und die Einrasteeinheiten voreinstellen, die dann im Dialog *Einrasteeinstellungen* vorausgewählt sind.

### Fang auf Schnittpunkte aktivieren

Eine andere Möglichkeit stellt das  *Einrasten auf Schnittpunkte aktivieren* dar. Sie können damit auf Schnittpunkte der Geometrien von Layern einrasten, für die Einrasten aktiviert ist, selbst wenn es keinen Stützpunkt auf dem Schnittpunkt gibt.

### Maßstabsabhängiges Einrasten setzen

In einigen Fällen kann das Einrasten sehr langsam werden. Dies wird häufig durch die Menge an Features in einigen Layern verursacht, für deren Berechnung und Verwaltung ein umfangreicher Index erforderlich ist. Einige Parameter sind vorhanden, um das Fangen nur zu ermöglichen, wenn sich die Kartenansicht innerhalb eines relevanten Maßstabsbereichs befindet. Dies erlaubt, die kostspielige Indexberechnung im Zusammenhang mit dem Fangen nur in einem Maßstab durchzuführen, der für das Zeichnen relevant ist.


Sie können die Maßstabsabhängigkeit beim Einrasten in *Projekt* ► *Einrastoptionen...* setzen. Die Maßstabsabhängigkeit ist nur in der *Erweiterten Konfiguration* möglich.

Es stehen drei Modi für die Maßstabsabhängigkeit beim Einrasten zur Verfügung:

- *deaktiviert*: Das Einrasten ist aktiv, unabhängig davon, wie der aktuelle Kartenmaßstab ist. Dies ist die Voreinstellung.
- *Global*: Das Einrasten ist beschränkt und nur aktiv wenn der aktuelle Kartenmaßstab zwischen einem globalen Minimal- und Maximalwert liegt. Wenn Sie diesen Modus auswählen, werden zwei Eingabefelder aktiv, mit denen Sie die Maßstabsbereiche, in denen das Einrasten aktiv ist, konfigurieren können.
- *Je Layer*: Legen Sie den Maßstabsbereich für das Einrasten für jeden Layer einzeln fest. Wenn Sie diesen Modus auswählen, werden zwei Spalten aktiv, mit denen Sie die Maßstabsbereiche für jeden Layer einzeln konfigurieren können.

Bitte beachten Sie, dass der Minimal- und der Maximalmaßstab den QGIS-Konventionen folgt: Der Minimalmaßstab ist am weitesten „rausgezoozt“, während der Maximalmaßstab am weitesten „reingezoomt“ ist. Sollten Sie den Minimal- oder Maximalmaßstab auf „0“ oder „nicht gesetzt“ einstellen, wird die Einstellung nicht beachtet.

### Selbsteinrasten

Mit der Einstellung  **Selbsteinrasten** können Sie auf diejenige Geometrie einrasten, die Sie gerade bearbeiten. Zusammen mit dem *Bedienfeld erweiterte Digitalisierung*, haben Sie damit eine praktische Möglichkeit, neue Kanten relativ zu vorhandenen Kanten oder Stützpunkten zu digitalisieren. Bitte benutzen Sie Selbsteinrasten mit Vorsicht, da es ungültige Geometrien erzeugen kann.


### Auf ein eigenes Gitter einrasten

Sie können die Einrastentfernung für jeden Layer individuell in den Layereigenschaften im Tab *Digitalisieren* einstellen. Indem Sie die Entfernung für die *Geometriegenauigkeit* einstellen aktivieren Sie ein Punktraster, das sichtbar wird, sobald der Kartenmaßstab sich in einem für die Darstellung sinnvollen Bereich bewegt. Sie können nun auf die Gitterpunkte einrasten: alle Stützpunkte einer neuen oder geänderten Geometrie werden automatisch auf den nächstgelegenen Gitterpunkt eingerastet. Weitere Informationen bietet *Eigenschaften Digitalisieren*.

## 16.3.3 Topologisches Editieren

Im Dialog *Einrasteeinstellungen...* (*Projekt ► Einrastoptionen*) und in der *Einrastwerkzeugleiste* können Sie einige topologische Funktionalitäten ein- und ausschalten.

### Topologisches Editieren aktivieren




Die Schaltfläche  **Topologisches Editieren aktivieren** unterstützt Sie beim Editieren und Pflegen von Objekten mit gemeinsamen Grenzen. Wenn diese Option eingeschaltet ist, „erkennt“ QGIS gemeinsame Objektgrenzen; wenn Sie gemeinsame Stützpunkte oder Segmente verschieben, wird QGIS sie auch in den benachbarten Objektgeometrien verschieben.

Das topologische Editieren funktioniert mit Objekten aus verschiedenen Layern, solange die Layer sichtbar sind und sich im Bearbeitungsmodus befinden.

In layer with Z or M values, topological editing will interpolate the Z or M value of the vertex based on the value of the edge used for the connection.

### Kontrolle über die Überlappungen

Durch die Steuerung der Überlappung können Sie neue Objekte zeichnen, die bestehende im ausgewählten Layer nicht überlappen, dies beschleunigt die Digitalisierung von aneinander angrenzenden Polygonen. Das Verhalten können Sie mit der Überlappungseinstellung steuern. Drei Modi stehen zu Auswahl:

1.  *Überlappung zulassen* (Standardeinstellung)
2.  *Überlappung auf aktiven Layer vermeiden*: verhindert jegliche Überlappung mit anderen Objekten des Bearbeitungslayers. Wenn Sie die neuen Geometrien so digitalisieren, dass sie ihre Nachbarflächen überlappen, wird QGIS überlappende Teile der neuen Geometrien abschneiden und sie auf die Grenzen der vorhandenen Objekte einrasten. Dadurch müssen Sie die gemeinsamen Stützpunkte auf der Grenzlinie nicht eigens digitalisieren.
3.  *Erweiterter Konfiguration folgen*: damit können Sie die Überlappungseinstellungen für jeden Layer individuell im Modus *Erweiterte Konfiguration* setzen.


**Bemerkung:** Wenn die neue Geometrie vollständig durch vorhandene abgedeckt ist, wird sie gelöscht und QGIS wird eine Fehlermeldung anzeigen.

**Warnung: Benutzen Sie die Option *Schnittpunkte vermeiden* mit Vorsicht**

Da diese Option neue überlappende Geometrien in jedem Polygonlayer zuschneidet, können Sie unerwartete Ergebnisse erhalten, wenn Sie es vergessen, sie auszuschalten, wenn Sie sie nicht mehr benötigen.


## Automatische Spurverfolgung

Wenn Sie Erfassungs-Kartenwerkzeuge (Objekt hinzufügen, Teil hinzufügen, Ring hinzufügen, Objekte überarbeiten oder Objekte zerteilen) benutzen, müssen Sie normalerweise jeden einzelnen Stützpunkt des Objekts anklicken. Mit der automatischen Spurverfolgung können Sie den Digitalisierungsprozess beschleunigen, weil sie nicht mehr jeden Stützpunkt manuell setzen müssen:

1. Schalten Sie das Werkzeug  *Spurverfolgung* über die Schaltfläche ein oder drücken Sie die Taste T.
2. *Fangen* Sie einen Stützpunkt oder ein Segment des Objekts, an dessen Grenze entlang Sie digitalisieren wollen.
3. Bewegen Sie die Maus über einen andern Stützpunkt oder ein anderes Segment, das sie fangen möchten. Anstelle der geraden Linie, die Sie üblicherweise erhalten, folgt das Gummiband dem Weg vom letztgefangenen Punkt bis zur aktuellen Mausposition.

QGIS benutzt die Topologie des zugrundeliegenden Objekts, um den kürzesten Weg zwischen den beiden Punkten zu bestimmen. Die Spurverfolgung erfordert, dass das Einrasten in dem Layer, in dem sich das zu verfolgende Objekte befindet, eingeschaltet ist. Während Sie digitalisieren sollten Sie einen vorhanden Stützpunkt oder ein vorhandenes Segment fangen und sicherstellen, dass Anfangs- und Endpunkt durch vorhandene Objektgrenzen topologisch verbunden sind. Ansonsten ist QGIS nicht in der Lage sie zu verbinden und erzeugt eine gerade Verbindungslinie zwischen ihnen.

4. Wenn Sie klicken, platziert QGIS alle Zwischenstützpunkte entlang des dargestellten Weges.

Klappen Sie die  *Spurverfolgung einschalten* Schaltfläche auf und setzen Sie die *Offset*-Option, um einen Weg parallel zu den Objekten zu digitalisieren anstatt auf ihrer Grenze; ein positiver Wert verschiebt die neue Linie in Spurverfolgungsrichtung nach links und ein negativer Wert nach rechts.

**Bemerkung: Justieren Sie den Kartenmaßstab oder die Fangeinstellungen für eine optimale Verfolgung**

Wenn zu viele Objekte in der Karte gezeigt werden, ist Verfolgung möglicherweise deaktiviert, um potentielle langsame Verfolgung und große Speicherzuschläge zu vermeiden. Nach dem hereinzoomen oder dem ausblenden einiger Layer, ist die Funktion wieder aktiviert.

**Bemerkung: Kein topologisches Editieren**

Dieses Werkzeug fügt bestehenden Polygonen keine neuen Punkte hinzu, auch wenn *Topologisches Editieren* eingeschaltet ist. Falls die Geometriegenauigkeit des Bearbeitungslayers gesetzt ist, kann die sich ergebende Geometrie von der bestehenden abweichen.

**Tipp: Schalten Sie die automatische Spurverfolgung schnell an oder aus, indem Sie T drücken**

Durch Drücken der T Taste, können Sie die Spurverfolgung jederzeit an- und ausschalten, sogar während Sie ein Objekt digitalisieren. Damit ist es möglich, einige Teile eines Objekts mit aktivierter Spurverfolgung zu digitalisieren und andere nicht. Alle Werkzeuge verhalten sich normal, wenn die Spurverfolgung ausgeschaltet ist.

**Tipp: Die Spurverfolgung in Kurvengeometrien umwandeln**

Über *Einstellungen* ► *Optionen* ► *Digitalisieren* ► *Spurverfolgung* können Sie während des Digitalisierens Kurvengeometrien erzeugen. Siehe *Optionen Digitalisieren*.

## 16.3.4 Einen vorhandenen Layer editieren

In der Voreinstellung lädt QGIS Vektorlayer im reinen Lesemodus, um ungewolltes Editieren zu verhindern wenn Sie versehentlich die Maus verrutschen. Ansonsten können Sie aber alle Ebenen editieren, vorausgesetzt der Datenanbieter unterstützt das (siehe *Exploring Data Formats and Fields*) und die Datenquelle ist überhaupt schreibbar (d.h. ihre Dateien sind schreibbar).






















**Tipp: Beschränkung der Editierbarkeit von Layern innerhalb eines Projekts**

Sie können jeden Layer in der Tabelle *Projekt* ► *Eigenschaften...* ► *Datenquellen* ► *Layereigenschaften* unabhängig vom Datenanbieter auf schreibgeschützt setzen. In einer Mehrbenutzerumgebung kann das praktisch sein, um zu verhindern, dass Nutzer fälschlicherweise einen Layer (z.B. eine Shapedatei) editieren, und damit die Daten möglicherweise beschädigen. Beachten Sie aber, dass die Einstellung nur im aktuellen Projekt gültig ist.

Die Werkzeuge zum Bearbeiten von Vektorlayern sind in die beiden Werkzeugleisten Digitalisierung und Erweiterte Digitalisierung, die im Abschnitt *Erweiterte Digitalisierung* beschrieben wird, aufgeteilt. Sie können beides unter *Ansicht* ► *Werkzeugkästen* ► an- oder abwählen. Wenn Sie die Digitalisierungswerkzeugleiste verwenden, können Sie folgende Funktionen ausführen:

In den grundlegenden Digitalisierfunktionen sind folgende Funktionen verfügbar:

Tab. 16.3: Vector layer basic editing toolbar


Werkzeug	Funktion	Werk	Funktion
	Änderungen in allen ausgewählten Layern gleichzeitig speichern, rückgängig machen oder abbrechen		Den Bearbeitungsstatus des ausgewählten Layers oder der ausgewählten Layer auf der Grundlage ihres aktuellen Status an- oder ausschalten.
	Die Änderungen im aktiven Layer speichern		
	Digitize using straight segments		Digitize using curve lines
	Enable freehand digitizing		Digitize polygon of regular shape
	Datensatz hinzufügen		Objekt hinzufügen: Punkt hinzufügen
	Objekt hinzufügen: Linie hinzufügen		Objekt hinzufügen: Polygon hinzufügen
	Knotenwerkzeug (alle Layer)		Knotenwerkzeug (aktueller Layer)
	Set whether the vertex editor panel should auto-open		Attribut aller gewählten Objekten gleichzeitig ändern
	Ausgewählte Objekte des aktiven Layers löschen		Ausgewählte Objekte des aktiven Layers ausschneiden
	Ausgewählte Objekte des aktiven Layers kopieren		Objekte in den aktiven Layer einfügen
	Änderungen im aktiven Layer rückgängig machen		Änderungen im aktiven Layer wiederholen

Beachten Sie, dass Sie während der Nutzung des Digitalisierungswerkzeugs, weiterhin in dem Kartenfenster *zoom or pan* können, ohne den Fokus auf das Werkzeug zu verlieren.

Beginnen Sie jeden Editiervorgang mit einem Click auf den Eintrag  *Bearbeitungsmodus umschalten* den Sie entweder im Kontextmenü eines Layers, in der Attributtabelle, der Digitalisierungswerkzeugleiste oder dem Menü *Layer* finden.





Sobald der Layer im Bearbeitungsmodus ist, werden zusätzliche Werkzeuge verfügbar und Markierungen an den Eckpunkten aller Objekte erscheinen, bis Sie die *Markierungen nur für gewählte Objekte anzeigen* Option unter *Einstellungen ► Optionen... ► Digitalisierung* aktiviert haben.

### Tipp: Regelmäßiges Sichern der Daten

Denken Sie daran  *Layeränderungen* regelmäßig zu speichern. Dies überprüft auch dass Ihre Datenquelle alle Änderungen akzeptiert.





## Geometry editing techniques


When a geometry drawing tool (mainly the ones that add, split, reshape features) is enabled for a line or polygon based layer, you can select the technique for adding new vertices:

- The  *Digitize with Segment*: draws straight segment whose start and end points are defined by left clicks.
- The  *Digitize with Curve*: draws curve line based on three consecutive nodes defined by left clicks (start, point along the arc, end). If the geometry type does not support curves, then consecutive smaller segments are used to approximate the curvature.
- The  *Stream Digitizing*: draws lines in freehand mode, i.e. nodes are added following cursor movement in the map canvas and a *Streaming Tolerance*. The streaming tolerance defines the spacing between consecutive vertices. Currently, the only supported unit is pixels (px). Only the starting left click and the ending right click are necessary in this mode.
- The  *Digitize Shape*: triggers tools on the *Shape Digitizing Toolbar* to draw a polygon of a regular shape.


The selected technique remains while switching among the digitizing tools. You can combine any of the first three methods while drawing the same geometry.

## Objekte digitalisieren

Abhängig von der Art des Layers benutzen Sie eine der Schaltflächen  *Objekt hinzufügen*,  *Punktobjekt hinzufügen*,  *Linienobjekt hinzufügen* oder  *Polygonobjekt hinzufügen* in der Werkzeugleiste, um neue Objekte zum aktiven Layer hinzuzufügen.

Fügen Sie ein geometrieloses Objekt hinzu, indem Sie auf den Knopf  *Objekt hinzufügen* klicken, danach tragen Sie im sich öffnenden Objektformular die Attributwerte ein.



Erzeugen Sie Objekte mit den Geometriewerkzeugen, indem Sie zunächst die Geometrie digitalisieren, danach geben Sie die Attributwerte ein. So digitalisieren Sie die Geometrie:

1. (Optional as it is the default) Select the  *Digitize With Segment* geometry drawing method
2. Klicken Sie mit der linken Maustaste in den Bereich der Karte, an dem Sie den ersten Punkt Ihres neuen Objekts erzeugen möchten. Für Punktobjekte sollte das schon genügen und, falls benötigt, das Objektformular öffnen, in das Sie die Attributwerte eintragen.



3. Machen Sie bei Linien- oder Polygonobjekten weitere Klicks mit der linken Maustaste für jeden weiteren Punkt, den Sie erfassen möchten. Sie können sich auf die Optionen für das *Einrasten*, das *Einrasten auf ein eigenes Gitter* oder das Bedienfeld *Erweiterte Digitalisierung* stützen, um jeden Stützpunkt genau zu platzieren.

Along with drawing straight segments between nodes you click one by one, lines and polygons can be:

- *traced automatically*, accelerating the digitization. This will create consecutive straight lines between the vertices you place, following existing features.
- free-hand digitized, pressing R or activating  Stream Digitizing.
- drawn as curve, pressing Ctrl+Shift+G or activating  Digitize with Curve.

---

**Bemerkung:** While digitizing line or polygon geometries, you can switch back and forth between the geometry drawing methods, allowing you to create features mixing straight segments, free-hand ones and curved parts.

---

4. Press Delete or Backspace key to revert the last node(s) you may wrongly add.
5. Wenn Sie alle Punkte digitalisiert haben, klicken Sie mit der rechten Maustaste auf eine beliebige Stelle, um die Digitalisierung der Geometrie dieses Objektes abzuschließen.



---

#### **Tipp: Das Digitalisierungsgummiband anpassen**

Während des Erfassens von Polygonen kann das voreingestellte rote Band unterliegende Objekte oder Orte verstecken, um einen Punkt zu erfassen. Dies kann durch Einstellen einer geringeren Opazität (oder Alpha-Kanal) des Bands ausgebessert werden *Farbe füllen* im Menü *Einstellungen ► Optionen ► Digitalisierung*. Sie können auch die Verwendung des Gummibands vermeiden *Don't update rubber band during node editing*.

---

6. Wenn Sie ein Linienobjekt digitalisieren, schließt Umschalt + Rechtsklick die Linie automatisch.
7. Das Attributformular wird sich öffnen, in das Sie die Informationen für das neue Objekt eingeben können. Abb. 16.78 zeigt die Attributeingabe für einen neuen Fluss. Im Menü *Einstellungen ► Optionen* können Sie im Bereich *Digitalisieren* auch folgende Einstellungen vornehmen:

-  *Attributeingabeformular bei der Objekterstellung unterdrücken* um das Öffnen des Formulars zu vermeiden
- oder  *Letzte Attributwerteingaben wiederverwenden* um Felder beim Öffnen automatisch gefüllt zu haben und nur ändernde Werte zu korrigieren.

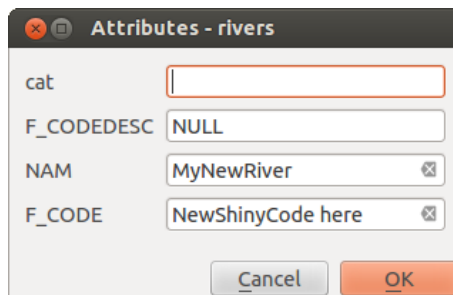




Abb. 16.78: Das Attributformular nach dem Digitalisieren eines neuen Vektorobjekts




## Das Knotenwerkzeug

QGIS bietet Ihnen zwei Werkzeuge zum Bearbeiten der Stützpunkte von Objekten in Vektorlayern an:


-  Knotenwerkzeug (Aktueller Layer): nur übereinanderliegende Objekte im aktiven Layer (im Bedienfeld *Layer*) werden bearbeitet
-  Knotenwerkzeug (alle Layer): übereinanderliegende Objekte in allen editierbaren Layern werden bearbeitet. Damit können Sie Objekte bearbeiten ohne den aktiven Layer wechseln zu müssen oder mehrere Layer gleichzeitig (z.B. Grenzen eines Landes und seiner Regionen)

Die Knotenwerkzeuge bieten Ihnen für jeden editierbaren Vektorlayer Bearbeitungsmöglichkeiten für Stützpunkte, ähnlich wie sie in CAD-Programmen vorhanden sind. Sie können mehrere Stützpunkte auf einmal auswählen und sie verschieben, neue hinzufügen oder sie alle zusammen löschen. Die Knotenwerkzeuge unterstützen Sie auch beim topologischen Editieren. Ihre Auswahl bleibt bestehen, also wenn Sie eine Operation ausführen bleibt die Auswahl für das jeweilige Objekt und Werkzeug aktiv.

Es ist wichtig die Eigenschaften unter *Einstellungen* ►  *Optionen* ► *Digitalisierung* ► *Suchradius*:  auf eine Zahl größer als Null einzustellen. Andernfalls ist QGIS nicht in der Lage mitzuteilen welcher Stützpunkt bearbeitet werden soll und wird einen Warnhinweis zeigen.

---

### Tipp: Stützpunktmarkierungen

QGIS unterstützt verschiedene Arten von Stützpunktmarkierungen: ‚Teiltransparenter Kreis‘, ‚Kreuz‘ und ‚Keine‘. Um die Markierung zu ändern wählen Sie den Menüpunkt  *Optionen* im Menü *Einstellungen* und klicken Sie auf den Tab *Digitalisieren*. Dort wählen Sie den passenden Eintrag.

---

## Eine einfache Übung

Gegeben sei ein Layer im Bearbeitungsmodus. Aktivieren Sie das Knotenwerkzeug. Rote Kreise erscheinen, wenn Sie die Maus über Stützpunkte bewegen.

- **Stützpunkte auswählen:** Wählen Sie Stützpunkte auf eine der folgenden Weisen aus:
  - Klicken Sie sie einzeln an, während Sie `Umschalt` gedrückt halten
  - Klicken Sie und ziehen Sie ein Rechteck auf, das die gewünschten Stützpunkte enthält
  - Zeichnen Sie ein Polygon, das die gewünschten Stützpunkte enthält: Halten Sie die Taste `Alt` gedrückt und klicken Sie mit dem Knotenwerkzeug, um mit dem Digitalisieren des Polygons zu beginnen. Jeder folgende Klick fügt einen neuen Stützpunkt zum Polygon hinzu; es wird mit dem Gummiband dargestellt. Die Tasten `Zurück` oder `Entfernen` löschen den letzthinzugefügten Stützpunkt. Mit `Esc` brechen Sie die Auswahl über ein Polygon ab; das geht auch, indem Sie alle Stützpunkte löschen, die sie bisher erfasst haben. Mit Rechtsklick beenden Sie die Digitalisierung des Polygons und alle innerhalb gelegenen Stützpunkte sind ausgewählt.

Sobald ein Stützpunkt ausgewählt ist, wechselt seine Farbe auf blau. Fügen Sie weitere Stützpunkte zur aktuellen Auswahl hinzu, indem Sie `Umschalt` gedrückt halten, während Sie bei der Auswahl weiterhin so vorgehen, wie beschrieben. Halten Sie `Strg` gedrückt, um Stützpunkte aus der Auswahl zu entfernen.

---

### Tipp: Eine Objektauswahl beschränkt das Knotenwerkzeug

Sie können Stützpunkte über mehrere verschiedene Objekte (und Layer) auswählen. Interessieren Sie sich aber für die Stützpunkte eines bestimmten Objekts an einer Stelle mit vielen Objekten, sollten Sie zuerst dieses Objekt auswählen. Benutzen Sie danach die Stützpunktauswahl über ein Polygon oder ein Rechteck um die Sie interessierenden Stützpunkte: nun sind nur die Stützpunkte des vorher ausgewählten Objekts ausgewählt.

Genauso verhält es sich, wenn Sie das Objekt im Bedienfeld *Stützpunktbearbeitung* anzeigen.

---

- **Stützpunkte im Stapelbetrieb auswählen:** Der Stapelbetriebsmodus wird durch drücken von Umschalt+R aktiviert. Wählen Sie einen ersten Stützpunkt durch einen einzelnen Klick aus und bewegen Sie den Mauszeiger **ohne zu klicken** über einen anderen Stützpunkt. Diese Aktion wählt alle Stützpunkte zwischen den beiden aus, wobei (für Polygone) der kürzeste Weg zwischen beiden benutzt wird.

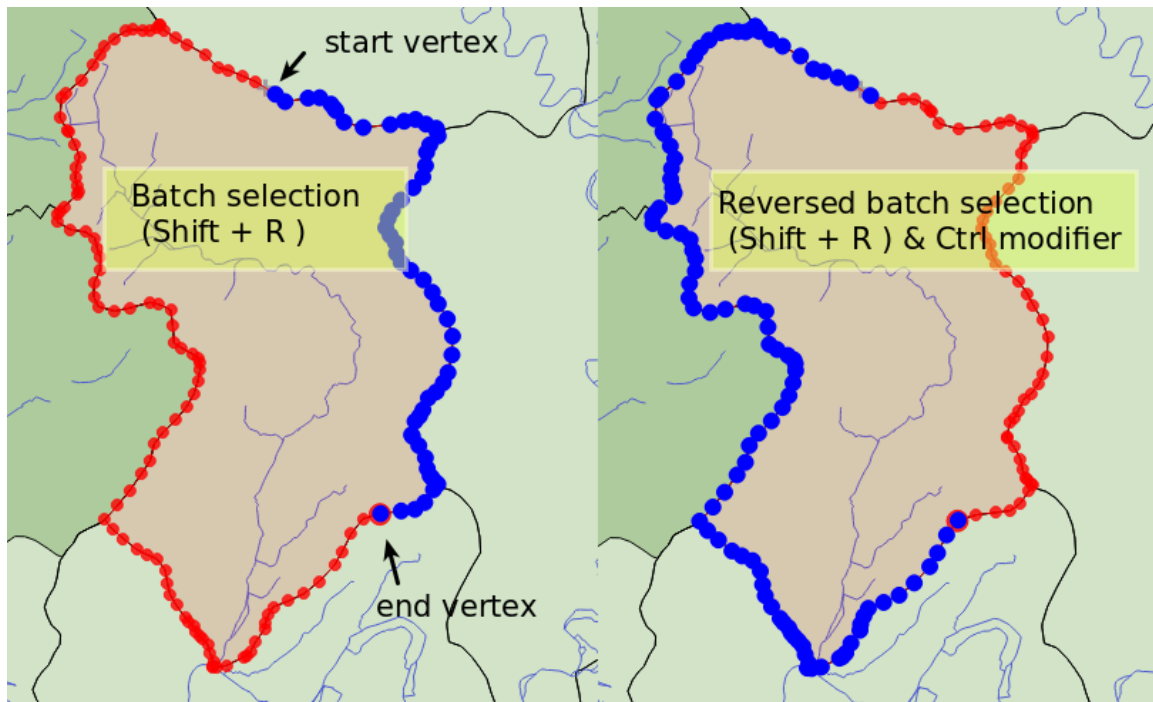


Abb. 16.79: Stützpunkte im Stapelbetrieb mit Umschalt+R auswählen

Wenn Sie **Strg** drücken, wird die Auswahl umgekehrt, so dass der längste Weg entlang der Objektgrenze benutzt wird. Die Stützpunktauswahl im Stapelbetrieb wird mit einem zweiten Klick abgeschlossen, oder mit **Esc** abgebrochen.

- **Stützpunkte hinzufügen:** Halten Sie **Umschalt** gedrückt und doppelklicken Sie auf die entsprechende Stelle des Segments, an dem Sie einer Linie oder einem Polygon einen neuen Stützpunkt hinzufügen wollen.

Während Sie die Maus über ein Segment bewegen, erscheint ein virtueller Stützpunkt in der Mitte des Segments. Sie können darauf klicken, den Mauszeiger an einen anderen Ort bewegen und dort erneut klicken, um einen neuen Stützpunkt hinzuzufügen. Bei Linien erscheint der virtuelle Stützpunkt an beiden Endpunkten: klicken Sie darauf, klicken Sie weitere Male und beenden Sie die Eingabe mit einem Rechtsklick; mit dieser Methode können Sie einen vorhandenen Linienzug verlängern.

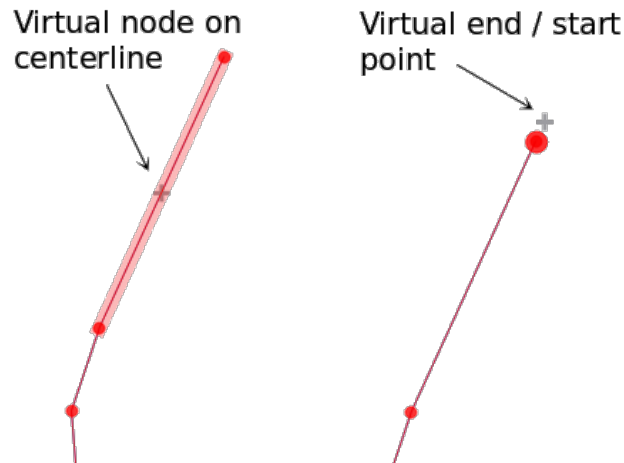



Abb. 16.80: Virtuelle Knoten um Stützpunkte hinzuzufügen

- **Stützpunkte löschen:** Wählen Sie die Stützpunkte und drücken Sie auf **Entfernen**. Das Löschen aller Stützpunkte erzeugt ein geometrieloses Objekt, sofern das mit der Datenquelle vereinbar ist. Beachten Sie, dass das nicht das gesamte Objekt löscht, sondern nur den Geometrieteil; um ein Objekt in Gänze zu löschen, müssen Sie das Werkzeug  **Ausgewähltes löschen** benutzen.
- **Stützpunkte verschieben:** Wählen Sie zunächst alle Stützpunkte aus, die Sie verschieben wollen, klicken Sie auf einen davon und dann auf die gewünschte Zielposition. Benutzen Sie die *Einrastoptionen* und die Einschränkungen für Entfernung, Winkel sowie exakte X- und Y-Positionen im Bedienfeld *Erweiterte Digitalisierung* bevor Sie das zweite mal klicken. Alle ausgewählten Stützpunkte werden verschoben.

Sollten Sie aber die Option *auf Gitter einrasten* aktiviert haben, werden die ausgewählten Stützpunkte auf denjenigen Gitterpunkt eingerastet, der der neuen Position am nächsten ist. Auch nicht ausgewählte Stützpunkte werden zum nächstgelegenen Gitterpunkt verschoben. Es gibt kein einfaches Verschieben.

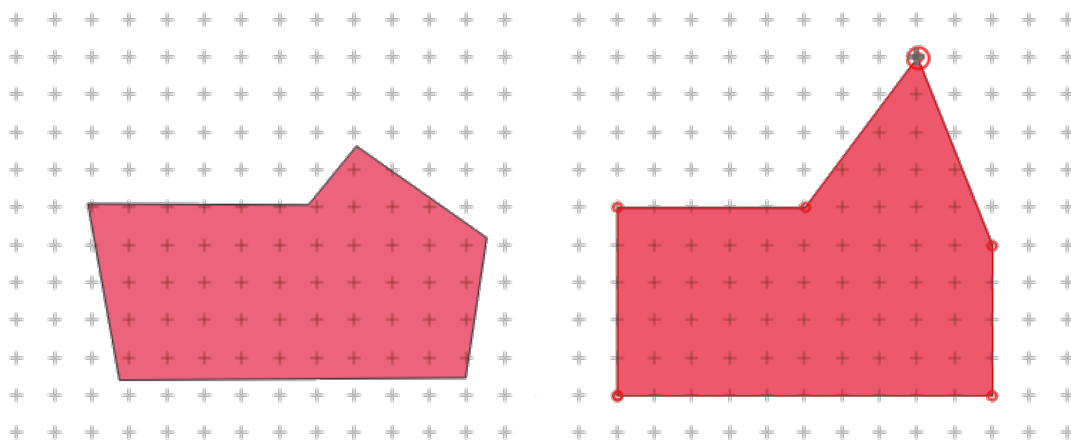


Abb. 16.81: Wird der oberste Stützpunkt verschoben, werden alle Stützpunkte auf das Gitter eingerastet

- **Converting adjacent segments to/from curve:** Select the center vertex of the segment you want to convert, hit the **O** letter key. If the vertex was in a curve, the curve is converted into straight lines. If the vertex was between two straight lines, they are converted into a curve. A first or a last vertex of a line can't be converted to a center vertex curve. The layer must be compatible with curve geometry type.

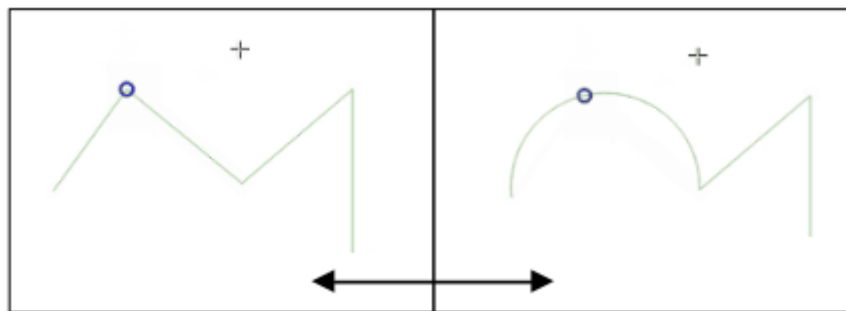



Abb. 16.82: Switch from curve to straight lines with O letter

Jede Änderung, die Sie mit dem Knotenwerkzeug vornehmen wird als eigener Eintrag im Bedienfeld *Rücknahme* gespeichert. Denken Sie daran, dass alle Operationen sowohl topologisches Editieren als auch das Umprojizieren in Echtzeit unterstützen.

### Das Bedienfeld Stützpunktbearbeitung

With enabling a vertex tool, you also open the *Vertex Editor* panel. Right-clicking over a feature fills the panel with the list of all the vertices of the feature with their  $x$ ,  $y$  ( $z$ ,  $m$  if applicable) coordinates and  $r$  (for the radius, in case of circular geometry). The feature is also made exclusive for editing, meaning that the edit of any other features is disabled:

- Selecting a row in the table does select the corresponding vertex in the map canvas, and vice versa.
- Clicking or dragging over the map canvas will only select or move vertices and segments of that feature
- Change a coordinate in the table and the vertex position is updated. This is a convenient way to edit Z coordinate or M value on vertices.
- You can also select multiple rows and delete them altogether.
- New vertices can only be added to the bound feature

If you do not want the *Vertex Editor* panel to immediately show each time you interact with vertex tools (and potentially hide other panels or disturb panels placement), uncheck the *Auto-open table* entry in the  Options menu at the top of the panel. You can then also close the panel. To reopen the panel, you would need to right-click over a panel or toolbar and select it in the list or tick the *Show vertex editor* entry in the *Digitizing toolbar*.

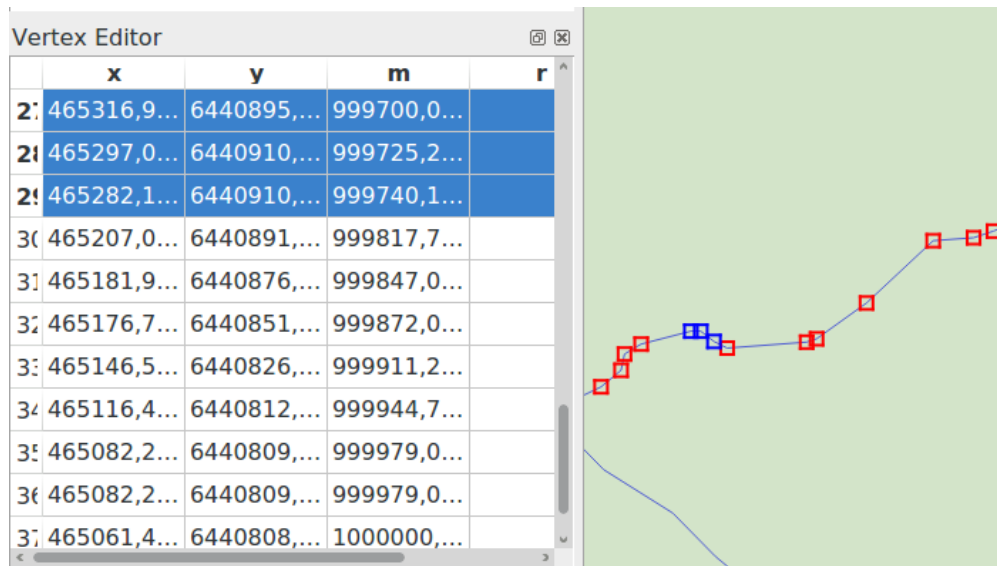



Abb. 16.83: Das Bedienfeld Stützpunktbearbeitung mit ausgewählten Knoten

### Rules of Z coordinate or M value assignment


Digitizing 3D vector features or features with M value is not that different from (X,Y) 2D layers. Tools and options described in this chapter are still available and help you place the vertex or point in a planar environment. Then you may need to handle the Z coordinate (or M value) assignment:

- By default, QGIS will assign to new vertices the *Default Z value* (respectively *Default M value*) set in the *Settings ► Options ► Digitizing* tab. If the *Advanced Digitizing Panel* is in use, then the value is taken from its *z* (respectively *m*) widget.
- When snapping to a vertex, the new or moved vertex takes the snapped one's Z or M value.
- When snapping to a segment while the topological editing is on, then the new vertex Z or M value is interpolated along the segment.
- If the *z* (respectively *m*) widget of the *Advanced Digitizing Panel* is  locked, then its value is applied to the vertex, taking precedence over any snapped vertex or segment Z or M value.

To edit Z or M values of an existing feature, you can use the *Vertex editor panel*. To create features with custom Z or M values you may want to rely on the *Advanced Digitizing Panel*.

### Objekte ausschneiden, kopieren und einfügen

Ausgewählte Objekte können ausgeschnitten, kopiert und an andere Ebenen im aktuellen QGIS-Projekt übergeben (eingefügt) werden, wenn sich der Ziellayer auch im Editiermodus befindet, indem Sie nach Auswahl des Ziellayers

auf den Knopf  Bearbeitungsstatus umschalten klicken.




#### **Tipp: Polygone in Linien und umgekehrt transformieren mit copy/paste**

Kopieren Sie ein Linienobjekt und fügen Sie ihn in ein Polygonlayer ein: QGIS fügt ihn in den Ziellayer dessen Begrenzung am nächsten der Geometrie des Linienobjekts entspricht. Dies ist ein einfacher Weg, um verschiedene Geometrien der selben Daten zu erzeugen.

Objekte können an externe Anwendungen als Text übergeben werden. Dabei werden die Objekte im CSV-Format dargestellt und die Geometriedaten im OGC Well-Know Text (WKT) Format. WKT und GeoJSON-Objekte von außerhalb können ebenso in einen Layer innerhalb von QGIS eingefügt werden.

Aber wann macht es Sinn, Objekte zum kopieren, auszuschneiden und einzufügen? Ein Beispiel ist, wenn Sie parallel an mehreren Layern arbeiten und Objekte zwischen den Layern hin- und herkopieren möchten. Ein Szenario könnte sein, dass Sie einen neuen Layer erstellen möchten, in dem aber nur einige Objekte aus einem bereits existierenden Layer verwendet werden sollen, wie etwa 5 Seen aus der Karte `lakes.shp`, die insgesamt aber tausende Seen enthält.

Als Beispiel werden wir einige Seen in einen neuen Layer kopieren:

1. Laden Sie den Layer, von dem Sie einige Objekte kopieren wollen (Quelle)
2. Laden oder erstellen Sie einen Layer, in den die kopierten Objekte eingefügt werden sollen (Ziel)
3. Schalten Sie für den Ziel Layer den Bearbeitungsstatus ein
4. Stellen Sie die Quelle aktiv, indem Sie es in der Legende anklicken
5.  Objekte über Fläche oder Einzelklick wählen um Objekte aus dessen Quelllayer zu wählen
6. Klicken Sie auf das Icon  Objekte kopieren
7. Stellen Sie das ‚Ziel‘ aktiv, indem Sie es in der Legende anklicken
8. Klicken Sie auf das Icon  Objekte einfügen
9. Beenden Sie den Bearbeitungsstatus für beide Layer und speichern Sie das Ergebnis ab

Was passiert, wenn der Quell- und Ziellayer ein unterschiedliches Schema enthält (Spaltennamen und -typen unterscheiden sich)? QGIS verwendet die Einträge, die gleich sind und ignoriert den Rest. Wenn es Ihnen egal ist, ob die Attribute korrekt übernommen werden, dann ist es egal, wie Sie die Spaltennamen und -typen der Attributtabelle erstellen. Wenn auch die Attributdaten korrekt übernommen werden sollen, dann stellen Sie sicher, dass auch die Spaltennamen und -typen beider Layer zueinander passen.

---

### Bemerkung: Deckungsgleichheit eingefügter Objekte

Wenn Ihre Quell- und Ziellayer die gleiche Projektion verwenden dann haben die eingefügten Objekte die identische Geometrie wie der Quelllayer. Wenn der Ziellayer jedoch eine andere Projektion hat dann kann QGIS nicht garantieren dass die Geometrie identisch ist. Dies ist einfach aus dem Grund so, dass sich kleine Rundungsfehler ergeben wenn zwischen Projektionen konvertiert wird.

---



---



### Tipp: Zeichenketten Attribut in ein anderes kopieren

If you have created a new column in your attribute table with type ‚string‘ and want to paste values from another attribute column that has a greater length the length of the column size will be extended to the same amount. This is because the GDAL Shapefile driver knows to auto-extend string and integer fields to dynamically accommodate for the length of the data to be inserted.



---

## Ausgewählte Objekte löschen

Wenn Sie ein gesamtes Objekt (Attribute und Geometrie) löschen wollen, können Sie das tun, indem Sie zunächst die Geometrie der  Objekte über Fläche oder Einzelklick wählen auswählen. Die Auswahl kann auch über die Attributtabelle erfolgen. Sobald Sie die Auswahl festgelegt haben, drücken Sie **Entfernen** oder **Rücktaste** oder nutzen Sie  **Ausgewähltes löschen** um die Objekte zu löschen. Mehrfachauswahlen können ebenso auf einmal gelöscht werden.

Das Werkzeug  **Ausgewählte Objekte ausschneiden** kann auch benutzt werden, um Objekte zu löschen. Die Objekte werden gelöscht aber zusätzlich noch im ‚spatial clipboard‘ abgelegt. In diesem Fall könnte man dann den letzten Schritt, falls ein Fehler unterlaufen ist, wieder rückgängig machen, indem wir auf das Werkzeug  **Objekte einfügen** drücken. Ausschneiden, kopieren und übergeben von Objekten funktioniert mit den gerade ausgewählten Objekten und können nach Bedarf kombiniert verwendet werden.

## Rückgängig und Wiederholen

Mit dem  Rückgängig und  Wiederholen Werkzeug können Sie Vektorbearbeitungsoperationen rückgängig machen oder wiederholen. Es gibt auch ein andockbares Bedienfeld das alle Operationen im Rücknahme-/Wiederholungsverlauf zeigt (siehe [Figure\\_edit\\_undo](#)). Dieses Bedienfeld wird nicht standardmäßig angezeigt; es kann angezeigt werden, indem Sie mit rechts auf die Werkzeugleiste klicken und das Rücknahme/Wiederholungs-Kontrollkästchen aktivieren. Rücknahme/Wiederholung ist auch dann aktiv wenn das Bedienfeld nicht angezeigt wird.

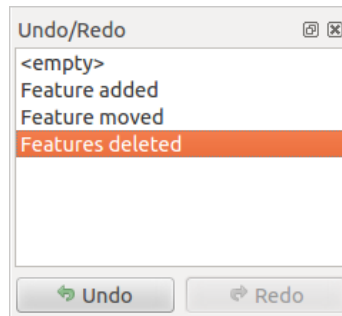




Abb. 16.84: Rückgängig und Wiederholen von Digitalisierschritten

Wenn Rückgängig oder `Strg+z` (oder `Cmd+z`) gedrückt wird, wird der Status aller Objekte und Attribute auf den Status zurückgesetzt, den sie vor der zurückgenommenen Aktion hatten. Es kann sein, dass Änderungen die keine normalen Vektorbearbeitungsoperationen sind (z.B. Änderungen, die durch ein Plugin durchgeführt wurden) nicht rückgängig gemacht werden können. Dies hängt davon ab, wie die Änderungen durchgeführt werden.

Wenn Sie Rücknahme/Wiederholung verwenden wollen klicken Sie einfach auf eine Operation in der History; alle Objekte werden dann auf den Stand vor der ausgewählten Operation zurückgesetzt.

## Änderungen speichern

Wenn ein Layer im Bearbeitungsmodus ist behält QGIS alle Änderungen im Speicher. Aus diesem Grund werden diese nicht umgehend der Datenquelle oder -platte übermittelt. Wenn Sie Bearbeitungen in dem aktuellen Layer speichern wollen aber mit dem Bearbeiten fortfahren wollen ohne den Bearbeitungsmodus verlassen zu wollen können Sie den  Layeränderungen speichern Knopf klicken. Wenn Sie den Editiermodus mit  Bearbeitungsstatus umschalten ausschalten wollen werden Sie ebenfalls gefragt ob Sie Ihre Änderungen speichern oder verwerfen wollen.

Wenn die Änderungen nicht gespeichert werden können (z.B. weil die Festplatte voll ist oder Attribute Werte aufweisen, die außerhalb der Wertespanne liegen), bleiben die Änderungen erstmal im QGIS Arbeitsspeicher. Dies ermöglicht es, Änderungen vorzunehmen und dann nochmals die Daten zu speichern.




---

### Tipp: Datenintegrität

Es ist immer gut ein Backup von Ihren Daten zu machen bevor Sie mit dem Bearbeiten starten. Während die Autoren von QGIS sich bemühen die Integrität Ihrer Daten zu bewahren bieten wir keine Garantie in dieser Hinsicht.

---

## Mehrere Layer auf einmal speichern

Mit dieser neuen Funktion können Sie mehrere Layer digitalisieren. Wählen Sie  *Layeränderungen speichern* um alle Änderungen, die Sie in mehreren Layern gemacht haben, zu speichern. Sie haben auch die Möglichkeit  *Verwerfen für gewählte Layer* zu benutzen so dass die Digitalisierung für alle selektierten Layer rückgängig gemacht werden kann. Wenn Sie das Bearbeiten der selektierten Layer beenden wollen kann man das einfach mit der Funktion  *Abbruch für gewählte Layer* erreichen.






















Die gleichen Funktionen sind für das Bearbeiten aller Layer des Projektes zugänglich.

**Tipp: Transaktionsgruppen benutzen, um Änderungen in mehreren Layern zu bearbeiten, zu speichern oder zurückzunehmen**

Aktivieren Sie Option *Wenn möglich automatisch Transaktionsgruppen erstellen* im Menü *Projekt ► Einstellungen... ► Datenquellen* beim Arbeiten mit Layern aus der selben PostGreSQL-Datenbank, um ihr Verhalten zu synchronisieren (den Bearbeitungsstatus zusammen umschalten, Änderungen gleichzeitig speichern oder verwerfen).


## 16.3.5 Erweiterte Digitalisierung

Tab. 16.4: Vector layer advanced editing toolbar

Icon	Funktion	Icon	Funktion
	Erweiterte Digitalisierungswerkzeuge einschalten		
	Objekt(e) verschieben		Objekt(e) kopieren und verschieben
	Objekt(e) drehen		Objekt vereinfachen
	Objekt skalieren		
	Ring hinzufügen		Teil hinzufügen
	Ring füllen		Linie umkehren
	Ring löschen		Teil löschen
	Linie versetzen		Objekte überarbeiten
	Teile zerlegen		Objekte trennen
	Attribute gewählter Objekte vereinen		Gewählte Objekte verschmelzen
	Punktsymbole drehen		Punktsymbolversatz
	Objekt abschneiden/ausdehnen		






## Objekt(e) verschieben

Mit dem  Objekt(e) verschieben Werkzeug können Sie bestehende Objekte verschieben.

1. Wählen Sie die Objekte aus, die Sie verschieben möchten.
2. Klicken Sie in das Kartenfenster, um den Quellpunkt der Verschiebung zu markieren; Sie können das Einrasten benutzen, um einen exakten Punkt auszuwählen.


Sie können auch die Einstellungen im Bedienfeld *advanced digitizing constraints* benutzen, um die exakten Koordinaten des Quellpunktes zu setzen. In diesem Falle:


1. Klicken Sie zunächst auf die Schaltfläche , um das Bedeinfeld zu aktivieren.
2. Tippen Sie  $x$  und tragen Sie den entsprechenden Wert für den Quellpunkt ein, den Sie benutzen wollen.  
Drücken Sie dann die Schaltfläche  neben der Eingabezeile, um den Wert zu sperren.
3. Genauso für die  $y$ -Koordinate.
4. Klicken Sie ins Kartenfenster und Ihr Quellpunkt wird an den eingegebenen Koordinaten platziert.
3. Bewegen Sie den Mauszeiger über das Kartenfenster, um den Zielpunkt der Verschiebung zu markieren, wobei Sie weiterhin den Einrastmodus benutzen können oder, wie oben beschrieben, benutzen Sie das Bedeinfeld Erweiterte Digitalisierung, das Ihnen zusätzliche Abstand- und "Winkel"-Einschränkungen anbietet, um den Zielpunkt der Verschiebung zu platzieren.
4. Klicken Sie ins Kartenfenster: Alle Objekte werden an ihren neuen Ort verschoben.

Mit dem Werkzeug  Objekt(e) kopieren und verschieben können Sie gleichermaßen eine verschobene Kopie der Objekte erzeugen.

**Bemerkung:** Wenn kein Objekt ausgewählt ist, während Sie das erste mal mit einem der Werkzeuge *Objekt(e) verschieben* oder *Objekt(e) kopieren und verschieben* ins Kartenfenster klicken, wird nur das Objekt unter dem Mauszeiger von der Aktion betroffen. Sollten Sie also mehrere Objekte verschieben wollen, sollten sie vorher ausgewählt worden sein.


## Objekt(e) drehen

Mit dem Werkzeug  Objekt drehen können Sie ein oder mehrere Objekte im Kartenfenster drehen.


1. Drücken Sie auf die Schaltfläche  Objekt drehen
2. Klicken Sie auf das Objekt, das sie drehen möchten. Der Flächenschwerpunkt des Objekts wird als Drehmittelpunkt markiert und ein Eingabefeld erscheint, das den aktuellen :guilabel: `Rotation`swinkel anzeigt.
3. Klicken Sie ins Kartenfenster, wenn Sie mit der neuen Position zufrieden sind oder geben Sie den Drehwinkel händisch in das Eingabefeld ein. Sie können auch die Eingabe *Einrasten auf*  $^{\circ}$  benutzen, um den Winkel einzuschränken.
4. Wollen Sie mehrere Objekte auf einmal drehen, sollten Sie sie zuerst auswählen. Der Drehmittelpunkt ist dann der Schwerpunkt ihrer verbundenen Geometrien.

Sie können auch einen anderen Drehmittelpunkt als den Flächenschwerpunkt benutzen: klicken Sie ins Kartenfenster während Sie die Taste `Strg` drücken und der angeklickte Punkt wird der neue Drehmittelpunkt.


Wenn Sie `Umschalt` vor dem Klick auf die Karte halten, wird die Drehung in 45 Grad Schritten erfolgen, was anschließend in einem Eingabefeld bearbeitet werden kann.


Zum Abbrechen der Objektdrehung, drücken Sie die Taste `Esc` oder klicken Sie auf die Schaltfläche  Objekt(e) drehen.

## Objekt skalieren

Das Werkzeug  Objekt skalieren ist ähnlich wie das Werkzeug Objekt drehen. Aber anstatt das ausgewählte Objekt zu drehen skaliert es dessen Geometrie. Die Änderung wird in Abhängigkeit vom Ankerpunkt ausgeführt und den Skalierungsfaktor können Sie manuell in das Fenster eingeben, das in der oberen Ecke der Karte erscheint.

## Objekt vereinfachen


Mit dem Werkzeug  Objekt vereinfachen können Sie die Form einer Linie oder eines Polygons interaktiv verändern, indem Sie die Anzahl der Stützpunkte verringern oder verdichten, so lange die Geometrie dabei gültig bleibt:

1. Wählen Sie das Werkzeug  Objekt vereinfachen.
2. Klicken Sie auf das Objekt oder ziehen Sie ein Rechteck über die Objekte auf.
3. Ein Eingabefeld öffnet sich, in dem Sie die zu verwendende *Methode* festlegen können, d.h. ob Sie:
  - die *simplify the geometry* wollen, so dass sie weniger Stützpunkte als die Ausgangsgeometrie enthält. Die zur Verfügung stehenden Methoden sind Nach Abstand vereinfachen, Durch Einrasten auf Gitter vereinfachen oder Durch Fläche vereinfachen (Visvalingam). Dann müssten Sie noch die *Toleranz* in Layereinheiten, Pixel oder Karteneinheiten angeben, die für die Vereinfachung benutzt werden soll. Je höher die Toleranz, desto mehr Stützpunkte werden entfernt.
  - oder die *densify the geometries* wollen, indem mit der Option Glatt neue Stützpunkte eingefügt werden: für jeden bestehenden Stützpunkt werden zwei Stützpunkte auf jedes von ihm ausgehende Segment gesetzt, und zwar in der mit Versatz definierten Entfernung, wobei der Versatz dem Prozentanteil der Segmentlänge entspricht. Mit Wiederholungen legen Sie fest, wie oft die Platzierung wiederholt werden soll: je mehr Wiederholungen Sie einstellen, desto mehr Stützpunkte werden eingefügt und desto regelmäßiger wird das Objekt.

Wenn Sie ein Projekt schließen oder eine Editiersitzung beenden, werden Einstellungen, die Sie benutzt haben, gespeichert. Sie können also die selben Parameter wieder benutzen, wenn Sie das nächste mal ein Objekt vereinfachen.

4. Unten im Eingabefenster wird eine Zusammenfassung der Veränderungen angezeigt, die sich ergäben; dabei wird die Anzahl der Objekte und Stützpunkte aufgelistet (vor und nach der Operation sowie das Verhältnis daraus). Zusätzlich wird im Kartenfenster die zu erwartende Geometrie über die bestehende mithilfe der Gummibandfarbe dargestellt.
5. Sobald die zu erwartende Geometrie Ihren Vorstellungen entspricht, klicken Sie *OK*, um die Änderung anzuwenden. Sollten Sie die Operation abbrechen wollen, klicken Sie auf *Abbrechen* oder klicken Sie mit rechts in das Kartenfenster.



---

**Bemerkung:** im Gegensatz zur Objektvereinfachung im Menü *Einstellungen* ► *Optionen* ► *Darstellung*, welches die Geometrie nur zur Darstellung vereinfacht, bearbeitet das Werkzeug  Objekt vereinfachen die Geometrie in der Datenquelle.


---

## Teil hinzufügen

Sie können zu einem ausgewählten Objekt einen  Teil hinzufügen, das erzeugt ein Multipunkt-, Multilinie- oder Multipolygonobjekt. Der neue Teil muss außerhalb des bereits vorhandenen digitalisiert werden, der vorher ausgewählt werden sollte.

Die Schaltfläche  Teil hinzufügen kann ebenfalls benutzt werden, um eine Geometrie zu einem geometrieloosen Objekt hinzuzufügen. Zuerst wählen Sie das Objekt in der Attributtabelle, dann digitalisieren Sie eine neue Geometrie mit dem Werkzeug .


## Teil löschen




Mit dem Werkzeug  Teil löschen können Teile eines Multiobjekts gelöscht werden (z.B. ein Polygon eines Multipolygonobjekts). Das Werkzeug funktioniert mit allen Multigeometrien: Punkt, Linie und Polygon. Weiterhin kann es dazu verwendet werden, die Geometrie eines Objekts vollständig zu entfernen. Um einen Teil zu löschen, klicken Sie einfach in dieses Teil.

## Ring hinzufügen

Sie können Löcher in Polygonen mit dem Werkzeug  Ring hinzufügen herstellen. Das bedeutet, dass Sie innerhalb eines bestehenden Polygons weitere Polygone digitalisieren. Diese erscheinen dann als ‚Loch‘, so dass nur der Zwischenraum des inneren und des äußeren Polygons als Ringpolygon übrig bleibt.


## Ring füllen

Das Werkzeug  Ring füllen hilft Ihnen dabei, ein Polygonobjekt zu erzeugen, das komplett innerhalb eines anderen Polygons liegt und so gestaltet ist dass sich beide nirgendwo überlappen; das neue Objekt liegt innerhalb eines Lochs im bestehenden Objekt. So gehen Sie vor:


1. Wählen Sie das Werkzeug .
2. Zeichnen Sie ein Polygon über das vorhandene Objekt: QGIS fügt der Geometrie ein neues Loch hinzu (wie wenn Sie das Werkzeug  Ring hinzufügen benutzt hätten) und erzeugt ein neues Objekt, dessen Geometrie dem Loch entspricht (wie wenn Sie *Spurverfolgung* über die innere Grenze mit dem Werkzeug  Polygonobjekt hinzufügen angewendet hätten).
3. Falls das Loch im Objekt bereits existiert, können Sie alternativ den Mauszeiger darüber positionieren und mit links klicken, während Sie die :kbd:`Umschalt`-taste gedrückt halten: ein neues Objekt, das das Loch ausfüllt, wird an dieser Stelle gezeichnet.

Das Formular *Objektattribute* des neuen Objekts öffnet sich und ist mit Werten des „Eltern“-objekts und/oder *fields constraints* vorausgefüllt.

## Ring löschen

Das Werkzeug  Ring löschen ermöglicht es, Löcher innerhalb einer existierenden Fläche zu löschen. Das Werkzeug funktioniert nur mit Polygon- und Multipolygonobjekten. Es findet keine Veränderung statt, wenn es auf den äußeren Ring eines Polygons angewendet wird.

## Objekte überarbeiten

Mit dem Werkzeug  Objekte überarbeiten können Sie Linien- und Polygonobjekte umformen. Bei Linien ersetzt es den Teil der Linie vom ersten bis zum letzten Schnittpunkt mit der ursprünglichen Linie.

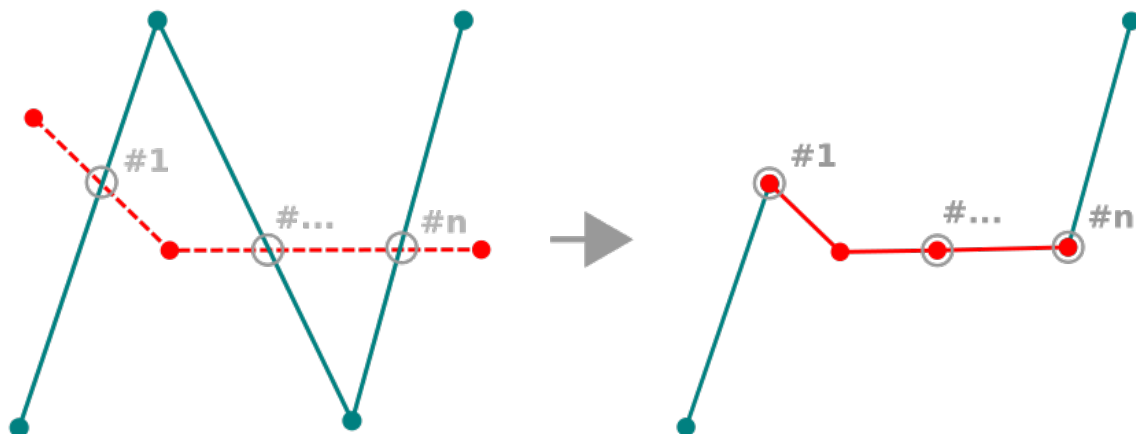



Abb. 16.85: Eine Linie umformen

### **Tipp: Liniengeometrien mit dem Werkzeug Objekte überarbeiten verlängern**

Benutzen Sie das Werkzeug  Objekte überarbeiten, um vorhandene Liniengeometrien zu verlängern: Fangen Sie den ersten oder letzten Stützpunkt der Linie und zeichnen sie eine neue. Schließen Sie die Digitalisierung ab und die neue Objektgeometrie ist die vereinigte Geometrie der beiden Linien.

Bei Polygonen wird die Polygongrenze umgeformt. Damit das funktioniert, muss die Linie des Umformwerkzeugs die Polygongrenze mindestens zwei mal kreuzen. Um diese Linie zu zeichnen, klicken Sie in das Kartenfenster, um ihre Stützpunkte zu erzeugen. Die Zeichnung schließen Sie mit einem Rechtsklick ab. Genauso wie bei Linienobjekten wird nur das Segment zwischen dem ersten und dem letzten Schnittpunkt der Linie mit der Polygongrenze betrachtet. Diejenigen Teile der Linie, die innerhalb des Polygons liegen, schneiden es dort ab, diejenigen, die außerhalb liegen erweitern es.

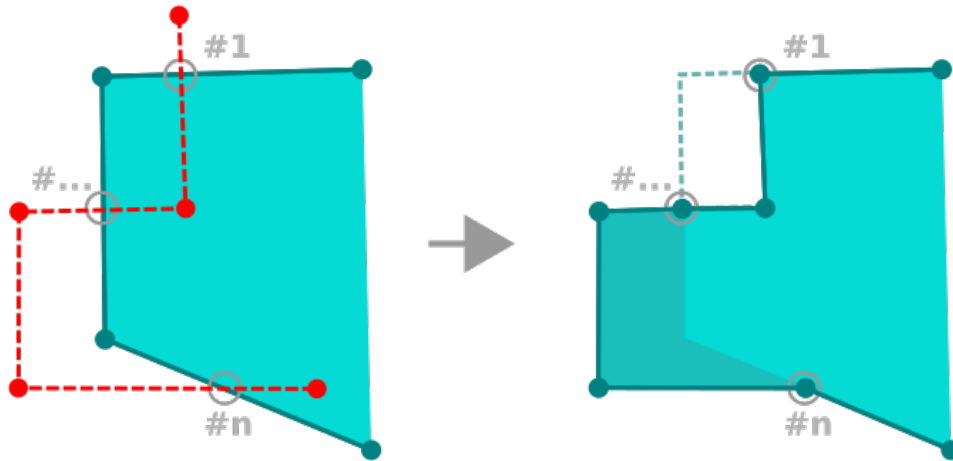





Abb. 16.86: Ein Polygon umformen

Bei Polygonen kann das Umformen manchmal zu unerwarteten Ergebnissen führen. Es ist in erster Linie dafür geeignet, kleinere Teile eines Polygons zu ersetzen und nicht für größere Überarbeitungen, außerdem ist es nicht erlaubt, dass die Umformlinie mehrere Ringe eines Polygons kreuzt, weil das zu einem ungültigen Polygon führen würde.

**Bemerkung:** Das Objekte überarbeiten Werkzeug kann die Startposition eines Polygonringes oder einer geschlossenen Linie verändern. Der Punkt, der zweimal abgebildet ist wird also nicht mehr der gleiche sein. Dies mag kein Problem für die meisten Anwendungen sein, sollte aber beachtet werden.

## Linie versetzen

Das  **Linie versetzen** Werkzeug erstellt parallele Linien von Linienlayern. Das Werkzeug kann auf den bearbeiteten Layer (die Geometrien werden verändert) und auch auf Hintergrundlayer (in diesem Fall erstellt es Kopien von Linien/ Ringen und fügt Sie dem bearbeiteten Layer hinzu) angewendet werden. Es ist auf diese Weise ideal geeignet Abstandslinienlayer zu erstellen. Der *Benutzereingabe* Dialog erscheint und zeigt den Versatz.


Um eine Linie zu versetzen müssen Sie zunächst in den Editiermodus wechseln und das Werkzeug  **Linie versetzen** aktivieren. Klicken Sie danach auf ein Objekt, um es zu versetzen. Bewegen Sie den Mauszeiger und klicken Sie erneut am Zielpunkt oder geben Sie die gewünschte Entfernung in die Eingabemaske ein. Halten Sie **Strg** während des zweiten Klicks gedrückt, um eine versetzte Kopie zu erzeugen. Danach können Sie Ihre Änderungen mit dem Werkzeug  **Layeränderungen speichern** speichern.

Der QGIS Optionen Dialog (Digitalisierung Reiter dann **Werkzeug zum Linien versetzen** Abschnitt) ermöglicht es Ihnen einige Parameter wie **Verbindungsstil**, **Quadrantsegmente**, **Eckengrenze** zu konfigurieren.


### Linie umkehren

Für kartografische Zwecke oder wenn ein Datensatz für Netzwerkanalysen vorbereitet wird, kann es sinnvoll sein, die Digitalisierrichtung einer Linie umzukehren.


So ändern Sie die Digitalisierrichtung einer Linie:

1. Aktivieren Sie das Werkzeug LinieUmkehren, indem Sie auf  Linie umkehren klicken.
2. Klicken Sie auf die Linie. Die Digitalisierrichtung wird umgekehrt.

### Objekte trennen


Mit dem Werkzeug  Objekte zerteilen teilen Sie ein Objekt in zwei oder mehr unabhängige Objekte, d.h. jede Geometrie gehört zu einer neuen Zeile in der Attributtabelle.

So teilen Sie Linien oder Polygonobjekte:

1. Wählen Sie das Werkzeug  Objekte zerteilen.
2. Draw a line across the feature(s) you want to split. If a selection is active, only selected features are split. Fields of resulting features are filled according to their *splitting policy*.
3. You can then as usual modify any of the attributes of any resulting feature.

---


#### **Tipp: Eine Polylinie mit einem Klick in neue Objekte zerteilen**

Fangen und Klicken Sie mit dem Werkzeug  Objekte zerteilen auf einen vorhandenen Stützpunkt einer Polylinie, um sie in zwei neue Objekte zu zerteilen.

---


### Teile zerlegen

In QGIS ist es möglich die Teile eines Multi-Part Features zu zerlegen so dass die Anzahl der Teile sich erhöht.

Zeichnen Sie einfach eine Linie über den Teil den Sie zerlegen wollen indem Sie dafür das  Teile zerlegen Icon verwenden.


---


#### **Tipp: Eine Polylinie mit einem Klick in neue Teile zerlegen**

Fangen und Klicken Sie mit dem Werkzeug  Teile zerlegen auf einen vorhandenen Stützpunkt einer Polylinie, um sie in zwei neue Polylinien zu zerlegen, die weiterhin zum selben Objekt gehören.

---

### Gewählte Objekte verschmelzen

Mit dem Werkzeug  Gewählte Objekte verschmelzen erzeugen Sie ein neues Objekt durch das verschmelzen vorhandener Objekte: Ihre Geometrien werden verschmolzen, um ein neues zu bilden. Wenn die Objekte keine gemeinsame Grenze haben, wird ein Multipolygon-/Multipolylinien-/Multipunktobjekt erzeugt.

1. Wählen Sie zunächst die Objekte aus, die Sie verschmelzen wollen.
2. Drücken Sie dann auf die Schaltfläche  Gewählte Objekte verschmelzen.
3. Die Zeile the *Verschmelzen* unten in der Tabelle des sich öffnenden Eingabefensters zeigt die Attribute des neuen Objekts. Sie können jeden dieser Werte folgendermaßen ändern:
  - ändern des Wertes in der entsprechenden Zelle;

- eine Zeile in der Tabelle auswählen und auf die Schaltfläche *Attribute des gewählten Objekts übernehmen* klicken. Damit übernehmen Sie alle Werte dieses Objekts;
- klicken Sie auf *Attribute des Objekts mit der größten Fläche verwenden*, um die Attribute des längsten Linienobjekts, des größten Polygonobjekts oder des Multipointobjekts mit den meisten Teilen zu übernehmen;
- auf die Schaltfläche *Alle Felder überspringen* klicken, um leere Attribute zu benutzen;
- klicken Sie auf das kleine Dreieck im Kopf der Tabelle und wählen Sie eine der Möglichkeiten aus, die dann ausschließlich auf das jeweilige Feld angewendet wird. Sie können auch die Werte der Ausgangsobjekte aggregieren (Minimum, Maximum, Median, Summe, Anzahl, Verknüpfung..., jeweils abhängig vom Feldtyp. Sehen Sie im Abschnitt *Statistical Summary Panel* für eine Liste aller Funktionen nach).


---

**Bemerkung:** Falls der Layer Standardwerte oder -klauseln für Felder definiert hat, werden diese als Startwert für das verschmolzene Objekt benutzt.


---


4. Drücken Sie auf *OK* um die Änderungen anzuwenden. Ein einzelnes (Multi-)Objekt wird im Layer angelegt und ersetzt die vorher ausgewählten Objekte.


### Attribute gewählter Objekte vereinen

Mit dem Werkzeug  *Attribute gewählter Objekte vereinen* können Sie die selben Attribute in Objekte speichern, ohne ihre Grenzen zu vereinen. Die Eingabemaske ist die selbe wie beim Werkzeug *Gewählte Objekte verschmelzen*, außer dass die ausgewählten Objekte mit ihrer Geometrie bestehen bleiben, während einige ihrer Attributwerte identisch werden.

### Punktsymbole drehen

Das Werkzeug  *Punktsymbole drehen* ermöglicht es Ihnen, den Drehwinkel eines Punktsymbols im Kartenfenster zu ändern.

1. Zuerst müssen Sie das Feld angeben, in dem der Rotationswert gespeichert werden soll. Dies geschieht, indem dem Symbol data-defined rotation property: ein Feld zugewiesen wird:
  1. Im Dialog *Layereigenschaften* ► *Symbolisierung* gehen Sie zu den Einstellungen für das Symbol.
  2. Klicken Sie auf die Schaltfläche  *Datendefinierte Übersteuerung* neben der Einstellung *Drehung* (bevorzugt) in der obersten Ebene *Markierung* der Symbolebenen.
  3. Wählen Sie ein Feld Auswahl *Feldtype*. Die Werte in diesem Feld werden nun dazu benutzt, jedes Symbol eines Objekts entsprechend zu drehen.

Sie können auch  *Daten im Projekt speichern* anhaken, um ein Feld im *Hilfsspeicher* zu erzeugen, das die Drehungswerte aufnimmt.

---

**Bemerkung:** Stellen Sie sicher, dass in allen Symbolebenen das selbe Feld eingestellt ist

Wenn Sie das datendefinierte Feld für die Drehung in der obersten Ebene der Symbolebenen einstellen, ist es automatisch in allen Symbolebenen eingestellt. Das ist die Voraussetzung für die graphische Symboldrehung mit dem Werkzeug *Punktsymbole drehen*. Falls auch nur eine Symbolebene nicht das selbe Feld für die Drehung eingestellt hat, wird das Werkzeug nicht funktionieren.

---

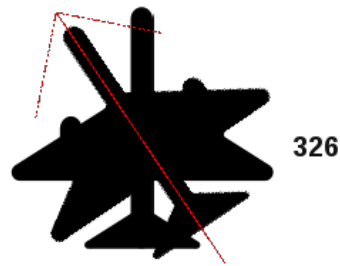






Abb. 16.87: Ein Punktsymbol drehen

2. Klicken Sie danach mit dem Werkzeug  Punktsymbole drehen auf ein Punktsymbol in der Karte
3. Bewegen Sie die Maus. Ein roter Pfeil mit dem Drehwinkel wird Ihnen angezeigt (siehe Abb. 16.87). Wenn Sie die Taste `Strg` gedrückt halten, während sie die Maus bewegen, verändert sich der Drehwinkel in Schritten von 15 Grad.
4. Klicken Sie nochmals, wenn der gewünschte Drehwinkel erreicht ist. Das Symbol wird dann mit diesem neuen Drehwinkel dargestellt und das Feld entsprechend gefüllt.

Klicken Sie mit der rechten Maustaste, um die Drehung des Punktsymbols abzubrechen.




## Punktsymbolversatz

Das Werkzeug  Punktsymbolversatz ermöglicht es Ihnen, die Darstellungsposition von Punktsymbolen im Kartenfenster interaktiv zu verändern. Es funktioniert ähnlich wie das Werkzeug  Punktsymbole drehen außer dass Sie ein Feld für den datendefinierten *Versatz* ( $X, Y$ ) des Symbols einstellen müssen. Dieses Feld wird dann für diejenigen Objekte mit den Versatzkoordinaten gefüllt, deren Symbol Sie im Kartenfenster hin- und herschieben.

1. Stellen Sie ein feld für die datendefinierte Übersteuerung der Symboleigenschaft *Versatz* ( $X, Y$ ) ein. Sollte das Symbol aus mehreren Layern bestehen, müssen Sie das Feld für jeden einzelnen davon einstellen
2. Aktivieren Sie das Werkzeug  Punktsymbolversatz
3. Klicken Sie auf einen Punkt
4. Verschieben Sie ihn auf eine neue Position
5. Klicken Sie nochmals. Das Punktsymbol wird auf die neue Position bewegt. Die Werte für den Versatz werden im verknüpften Feld gespeichert.

Klicken Sie mit der rechten Maustaste, um die Verschiebung des Punktsymbols abzubrechen.


---

**Bemerkung:** Das Werkzeug  Punktsymbolversatz verschiebt nicht das Punktobjekt selbst; dafür sollten Sie das Werkzeug  Knotenwerkzeug (aktueller Layer) oder das Werkzeug  Objekt verschieben benutzen.

---





## Objekt abschneiden/ausdehnen

The  Trim/Extend tool allows you to shorten or lengthen segments of a (multi)line or (multi)polygon geometry to converge with a selected segment (the cutting line). This results in a modified geometry with a vertex snapped to the target segment or in its prolongation. Depending on how the selected geometries are placed in relation to each other, the tool will either:


- **Trim:** removes parts of the line segment or polygon boundary, beyond the cutting line
- **Extend:** extends polygon boundaries or line segments so that they can snap to the cutting line.








In order to trim or extend existing geometries:

1. Enable appropriate *snapping settings* on segment for the involved layer(s)
2. Select the  Trim/Extend tool
3. Click the target limit segment, i.e. the segment with respect to which you want to extend or trim another segment. It appears highlighted.
4. Move to the segment you want to trim or extend. It does not need to be the last segment of the geometry, but has to be on the active layer.
5. Hover over the segment, and QGIS displays a preview of what the feature's geometry would be. If OK, click the segment. In the case of a trim, you must select the part that should be shortened.
6. Falls beide Segmente in 3D sind, führt das Werkzeug eine Interpolation zum Zielsegment aus, um den Z-Wert zu ermitteln.


**Achtung:** Pay attention to the modified geometry while using the  Trim/Extend tool. Depending on the inputs, it can create invalid geometries, potentially resulting in failure at layer saving.

## 16.3.6 Formen digitalisieren

The *Shape Digitizing* toolbar offers a set of tools to draw lines or polygons features of regular shape. It is synchronized with the  Digitize Shape *geometry drawing method* you can select on the *Digitizing Toolbar*. To use it:

1. Display the toolbar: **View ► Toolbars ► Shape Digitizing**
2. Select a tool that creates or modifies the shape of a geometry, e.g.  Add line feature,  Add polygon feature,  Add part,  Add ring,  Reshape Features, ...
3. The  Digitize with segment button on the *Digitizing Toolbar* is enabled. The first time, you may need to switch it to the  Digitize Shape in order to enable tools on the *Shape Digitizing* toolbar.
4. Pick a shape digitizing tool and draw.

### Circular string by radius

The  **Circular string by radius** button allows to add line or polygon features with a circular geometry, given two nodes on the curve and a radius:

1. Left click twice to place the two points on the geometry.
2. A *Radius* widget in the top right corner of the map canvas displays current radius (corresponding to distance between the points). Edit that field to the value you want.
3. An overview of the arcs matching these constraints is displayed while moving around the cursor. Right-click to validate when the expected arc is shown.
4. Add a new point to start shaping another arc.

---

#### Bemerkung: Gebogene Geometrien sind als solche nur in kompatiblen Datenanbieter gespeichert






Obwohl QGIS es ermöglicht, innerhalb jedes editierbaren Datenformats gebogene Geometrien zu digitalisieren, müssen Sie, um gebogene Geometrien zu speichern, einen Datenanbieter verwenden, der gebogene Objekte unterstützt (z. B. PostGIS, GML oder WFS), ansonsten teilt QGIS die Bögen in Einzelsegmente auf.

---

### Kreise zeichnen

Um Kreise zu zeichnen, gibt es einen Satz von Werkzeugen. Sie werden im folgenden beschrieben.



Kreise werden in Kreisbögen umgewandelt. Falls das die jeweilige Datenquelle zulässt, werden sie als gebogene Geometrien gespeichert. Falls nicht, wird QGIS die Kreisbögen in Segmente aufteilen. Dies wird im Abschnitt in *Circular string by radius* beschrieben.



-  **Circle from 2 points**: The two points define the diameter and the orientation of the circle. (Left-click, right-click)
-  **Circle from 3 points**: Draws a circle from three known points on the circle. (Left-click, left-click, right-click)
-  **Circle by a center point and another point**: Draws a circle with a given center and a point on the circle (Left-click, right-click). When used with the *Das Bedienfeld Erweiterte Digitalisierung* this tool can become a „Add circle from center and radius“ tool by setting and locking the distance value after first click.
-  **Circle from 3 tangents**: Draws a circle that is tangential to three segments. **Note that you must activate snapping to segments** (See *Einrasttoleranz und Suchradius einstellen*). Click on a segment to add a tangent. If two tangents are parallel, the coordinates of the click on the first parallel tangent are used to determine the positioning of the circle. If three tangents are parallel, an error message appears and the input is cleared. (Left-click, left-click, right-click)
-  **Circle from 2 tangents and a point**: Similar to circle from 3 tangents, except that you have to select two tangents, enter a radius and select the desired center.

### Ellipsen zeichnen

Um Ellipsen zu zeichnen, gibt es einen Satz von Werkzeugen. Sie werden im folgenden beschrieben.





Ellipsen können nicht in bogenförmige Geometrien umgewandelt werden, so dass sie immer segmentiert werden.

-  **Ellipse from center and two points**: Draws an ellipse with a given center, major axis and minor axis. (Left-click, left-click, right-click)
-  **Ellipse from center and a point**: Draws an ellipse into a bounding box with the center and a corner. (Left-click, right-click)

-  Ellipse from extent: Draws an ellipse into a bounding box with two opposite corners. (Left-click, right-click)
-  Ellipse from foci: Draws an ellipse by 2 points for foci and a point on the ellipse. (Left-click, left-click, right-click)

## Rechtecke zeichnen

Ihnen stehen eine Reihe von Werkzeugen zum Zeichnen von Rechtecken zur Verfügung, sie werden nachfolgend beschrieben.

-  Rechteck aus Zentrum und einem Punkt hinzufügen: Zeichnet ein Rechteck, dass über seinen Mittelpunkt und eine Ecke definiert wird. (Linksklick, Rechtsklick)
-  Rechteck nach Ausdehnung hinzufügen: Zeichnet ein Rechteck, das über zwei gegenüberliegende Ecken definiert wird. (Linksklick, Rechtsklick)
-  Rechteck über drei Punkte hinzufügen (Abstand vom 2. und 3. Punkt): Zeichnet ein orientiertes Rechteck aus drei Punkten. Der erste und der zweite Punkt definieren Länge und Winkel der ersten Seite. Der dritte Punkt legt die Länge der anderen Seite fest. Sie können das Bedienfeld *Das Bedienfeld Erweiterte Digitalisierung* benutzen, um die Länge der Seiten festzulegen. (Linksklick, Linksklick, Rechtsklick)
-  Rechteck über drei Punkte hinzufügen (Abstand vom projizierten Punkt auf Segment und 1. und 2. Punkt): Die Funktion ist genau so wie im vorstehenden Werkzeug, allerdings wird die Länge der zweiten Seite über die Projektion des dritten Punktes auf die erste Seite berechnet. (Linksklick, Linksklick, Rechtsklick)

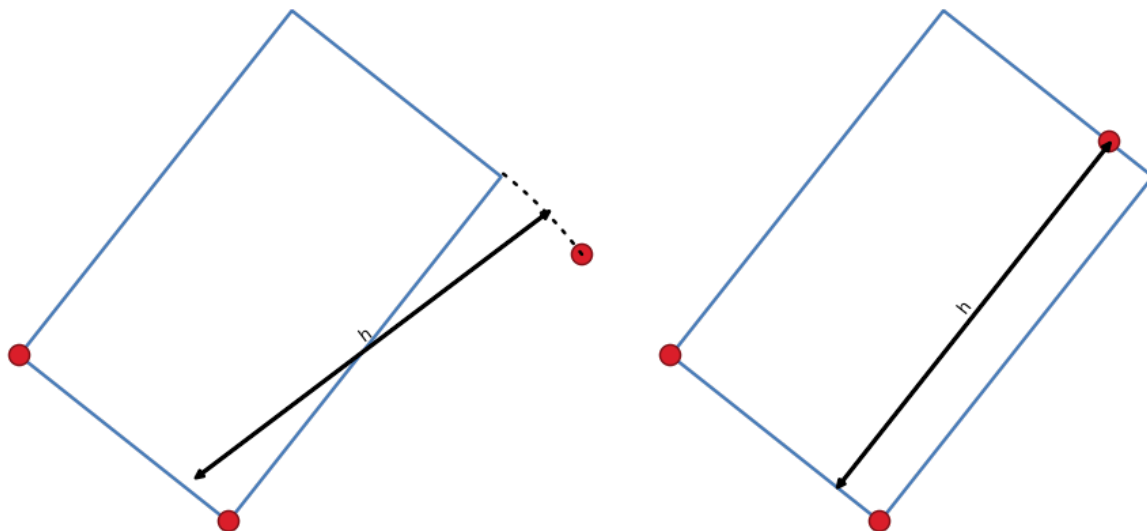





Abb. 16.88: Ein Rechteck über drei Punkte hinzufügen, mittels Abstand (rechts) und projiziertem Punkt (links)

## Regelmäßige Polygone zeichnen

Ihnen stehen eine Reihe von Werkzeugen zum Zeichnen von regelmäßigen Polygonen zur Verfügung, sie werden nachfolgend beschrieben. Klicken Sie mit links, um den ersten Punkt zu setzen. In dem danach erscheinenden Eingabefeld können Sie die Anzahl der Seiten festlegen. Klicken Sie mit rechts, um das regelmäßige Polygon abzuschließen.

-  **Regelmäßiges Polygon aus zwei Punkten hinzufügen:** Zeichnet ein regelmäßiges Polygon, bei dem Länge und Winkel der ersten Seite über zwei Punkte definiert werden.
-  **Regelmäßiges Polygon aus Zentrum und einem Punkt hinzufügen:** Zeichnet ein regelmäßiges Polygon über seinen Mittelpunkt. Der zweite Punkt bestimmt Winkel und Abstand einer Seite zu diesem Mittelpunkt.
-  **Regelmäßiges Polygon aus Zentrum und einer Ecke hinzufügen:** Die selbe Funktion wie das vorhergehende Werkzeug, jedoch bestimmt der zweite Punkt Winkel und Abstand zu einem Eckpunkt des Polygons.

### 16.3.7 Das Bedienfeld Erweiterte Digitalisierung

When capturing, reshaping, splitting new or existing geometries you also have the possibility to use the Advanced Digitizing panel. You can digitize lines exactly parallel or perpendicular to a particular angle or lock lines to specific angles. Furthermore, you can make a precise definition of your new geometry by entering X and Y coordinates as well as Z for 3D features, or M values.

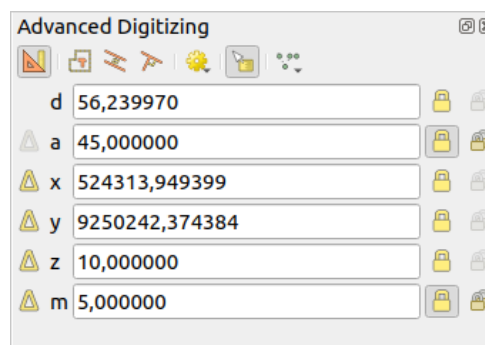


Abb. 16.89: Das Bedienfeld Erweiterte Digitalisierung

The *Advanced Digitizing* panel can be opened either with a right-click on the toolbar, from *View ► Panels ►* menu or pressing **Ctrl+4**. Once the panel is visible, click the  **Enable advanced digitizing tools** button to activate the set of tools.

---

**Bemerkung:** Die Werkzeuge sind nicht aktiviert, wenn die Kartenansicht geographische Koordinaten zeigt.






---

Die Werkzeuge zielen darauf ab, Koordinaten, Längen und Winkel zu sperren, während Sie den Mauszeiger beim Digitalisieren über das Kartenfenster bewegen.



Sie können auch Festsetzungen mit relativer oder absoluter Referenz vornehmen. Relative Referenz bedeutet, dass die Festsetzungen für den folgenden Stützpunkt relativ zum vorhergehenden Stützpunkt oder Segment sind.

## Die Werkzeuge




Oben im Bedienfeld *Erweiterte Digitalisierung* finden Sie folgende Schaltflächen:

-  Erweiterte Digitalisierungswerkzeuge einschalten
-  Konstruktionsmodus: ermöglicht es Ihnen, Ihre Klickpunkte zu erfassen, um sie als Referenzpunkte für das Sperren der Entfernung, des Winkels oder der Relativwerte für X, Y, Z oder M zu benutzen. Mehr dazu unter [Konstruktionsmodus](#).
-  Parallel ermöglicht es Ihnen, eine Linie parallel zu einer vorhandenen zu zeichnen (mehr dazu unter [Parallele und senkrechte Linien](#))
-  Senkrecht ermöglicht es Ihnen, eine Linie senkrecht zu einer vorhandenen zu zeichnen (mehr unter [Parallele und senkrechte Linien](#))
-  Snap to common angles: when moving the cursor, displays a virtual line that you can snap to to add the next vertex. The snapping line is defined by the last added vertex and an (absolute or relative to previous segment) angle from a preset list (following steps of 0.1°, 0.5°, 1°, 5°, 10°, 15°, 18°, 22.5°, 30°, 45° or 90°). Choose *Do not snap to common angles* to disable this feature.

*Snapping to features* can be used along with snapping to common angles for accurate digitizing. For a fine-grained control on how the target element to snap to is retained, you can indicate whether to prioritize snapping to features over common angles, and vice-versa under the *Snapping priority* entry. You can switch from one method to the other during the digitizing operation, and this avoids disabling any of the snapping options in the meantime. Press N (or Shift+N) during a digitizing operation to cycle through the angles list.

-  Floater settings: if the *Show floater* item is checked, a contextual menu with digitizing information follows the cursor during digitizing. The values can be accessed using the [panel's shortcuts](#), edited and  Locked after validation (pressing Enter). The type of information to display can be selected in the bottom part of the menu:

- *Show distance*
- *Show angle*
- *Show XY coordinates*
- *Show Z value*
- *Show M value*
- *Show bearing/azimuth*
- *Show common snapping angle*

-  Construction Tools provides a couple of options that constrain the vertices placement based on extrapolated coordinates of existing elements:
  -  *Line Extension*: hover over a segment and you get a purple dotted line extending the segment across the map canvas. You can snap the vertex anywhere on this virtual line.
  -  *X/Y Point*: hover over a vertex and you get a purple dotted line along its X or Y coordinate, across the map canvas. You can snap the vertex anywhere on this virtual line. It is even possible to hover over two different vertices, generating virtual coordinate lines for both, and snap to their intersection.

Below the toolbar, you will find a number of text boxes whose value reflects by default the position or movement of the cursor in the map canvas. Editing these values helps you constrain the position of the items you edit:

- *d* for the distance from a reference position, usually the last edited vertex
- *a* for the angle (absolute or relative) from a reference position, usually the last edited segment
- *x* for the X coordinate of the pointer

- $y$  for the Y coordinate of the pointer
- $z$  for the default Z value or the Z coordinate of the vertex or segment under the pointer
- $m$  for the default M value or the M value of the vertex or segment under the pointer

## Tastenkürzel

Um die Arbeit mit dem Bedienfeld Erweiterte Digitalisierung zu beschleunigen, gibt es eine Reihe von Tastenkürzeln:


Tab. 16.5: Keyboard shortcuts of the Advanced Digitizing Panel tools




Taste	nur Taste	Strg+ or Alt+	Umschalt+
D	Abstand einstellen	Abstand beibehalten	
A	Winkel einstellen	Winkel beibehalten	Winkel relativ zum letzten Segment an-/ausschalten
X	X-Koordinate einstellen	X-Koordinate beibehalten	X relativ zum letzten Stützpunkt an-/ausschalten
Y	Y-Koordinate einstellen	Y-Koordinate beibehalten	Y relativ zum letzten Stützpunkt an-/ausschalten
Z	Z-Koordinate einstellen	Z-Koordinate beibehalten	Z relativ zum letzten Stützpunkt an-/ausschalten
M	M-Wert einstellen	M-Wert beibehalten	M relativ zum letzten Stützpunkt an-/ausschalten
C	Konstruktionsmodus an-/ausschalten		
P	Zwischen senkrechtem und parallelem Modus umschalten		

**Bemerkung:** Z-Koordinate und M-Wert sind nur verfügbar, wenn die Geometriedimension des Layers dies hergibt.

## Mit absoluter Referenz digitalisieren

Sobald Sie eine neue Geometrie von Grund auf neu zeichnen, kann es hilfreich sein, den ersten Stützpunkt an festgelegten Koordinaten zu setzen.

Wenn Sie z.B. ein neues Objekt zu einem Polygonlayer hinzufügen, klicken Sie auf die Schaltfläche . Sie können die genauen Koordinaten angeben, an denen Sie die Digitalisierung beginnen möchten, d.h.:

1. klicken Sie auf das Eingabefeld  $x$  (oder benutzen das Tastenkürzel X),
2. geben Sie die X-Koordinate ein und drücken Sie **Eingabe** oder klicken Sie auf die Schaltfläche  rechts davon, um die Mausposition auf der X-Achse im Kartenfenster zu beizubehalten,
3. klicken Sie auf das Eingabefeld  $y$  (oder benutzen das Tastenkürzel Y),
4. geben Sie die Y-Koordinate ein und drücken Sie **Eingabe** oder klicken Sie auf die Schaltfläche  rechts davon, um die Mausposition auf der Y-Achse im Kartenfenster zu beizubehalten.
5. If the layer has Z coordinate or M values, the corresponding  $z$  or  $m$  widget is enabled and displays its default value, as set in *Settings* ► *Options* ► *Digitizing* tab.
  1. Click the  $z$  or  $m$  text box (or use respectively the Z or M keyboard shortcut).
  2. Type the coordinate value you want and press **Enter** or click the  button to their right to lock the value in the widget.

**Bemerkung:** Read *Rules of Z coordinate or M value assignment* for details on how Z coordinate and M values are automatically determined from existing features.

6. Two blue dotted lines and a green cross identify the exact coordinates you entered. Click on the map canvas to add a vertex at the green cross position.

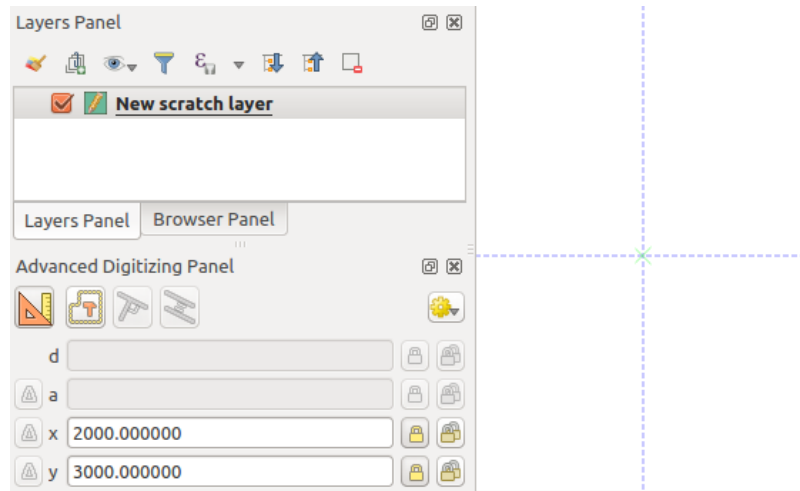



Abb. 16.90: Eine Zeichnung mit vorgegebenen Koordinaten beginnen

7. You can proceed as above, adding a new set of coordinates for the next vertex, or switch to another *mode of digitizing* (e.g. segment, curve or stream).
8. Falls Sie ein Segment bestimmter Länge zeichnen möchten:
  1. Klicken Sie auf das Eingabefeld *d* (*Entfernung*) (Tastenkürzel *D*, Entfernung = engl. distance)
  2. Tragen Sie die Entfernung ein (in Karteneinheiten)
  3. Drücken Sie *Eingabe* oder klicken Sie auf die Schaltfläche  rechts, um den Mauszeiger im Kartenfenster auf die Länge des Segments zu sperren (also die Länge beizubehalten). Um den letztgesetzten Stützpunkt wird ein Kreis gezeichnet, dessen Radius der eingegebenen Entfernung entspricht. Ein Kreuz auf der Kreislinie zeigt Ihnen die Position des nächsten Stützpunktes an, falls Sie nun klicken.

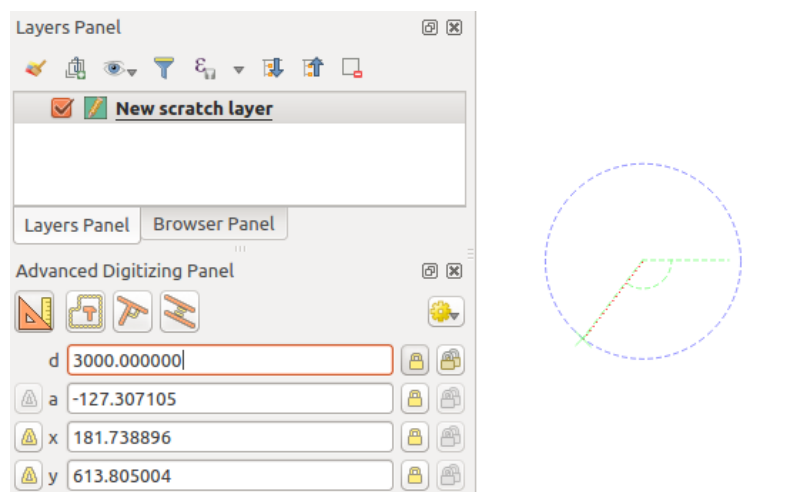



Abb. 16.91: Segment mit einer vorgegebenen Länge

9. Sie können auch die Position des Stützpunktes beschränken, indem Sie den Winkel des Segments setzen wie oben beschrieben:
  1. Klicken Sie auf das Eingabefeld *a* (*Winkel*) (Tastenkürzel A , Winkel = engl. angle)
  2. Geben Sie den Winkel (in Grad) ein
  3. Drücken Sie **Eingabe** oder klicken Sie auf die Schaltfläche  rechts, um den Winkel zu sperren. Durch den letztgesetzten Stützpunkt wird eine Linie gezeichnet, die um den eingegebenen Winkel gedreht ist. Ein Kreuz auf dieser Linie zeigt Ihnen die Position des nächsten Stützpunktes an, falls Sie nun klicken.

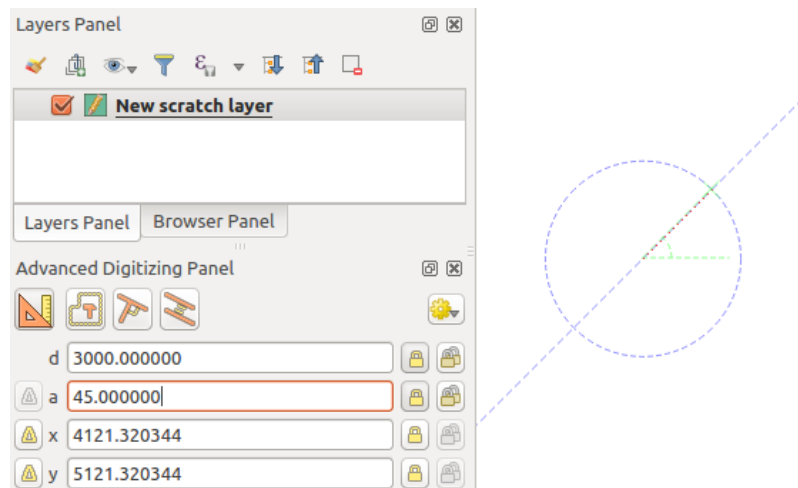





Abb. 16.92: Segment mit vorgegebenem Winkel

**Hinweis:** Drücken Sie **Strg+<key>** oder **Alt+<key>**, um die die Eingabe zu sperren den Wert automatisch bearbeitbar zu machen. Nun müssen Sie nur noch **Eingabe** drücken. Zusammen mit der  Anzeige über Karten umschalten kann das Digitalisieren über die Tastatur sehr zeitsparend sein.

### Mit relativer Referenz digitalisieren


Anstatt absolute Werte für Winkel oder Koordinaten können Sie Werte relativ zum zuletzt eingegebenen Stützpunkt oder Segment benutzen.

Bei Winkeln klicken Sie auf die Schaltfläche  links der Eingabezeile *a* (oder drücken Sie **Umschalt+A**) um Winkleingabe relativ zum zuletzt eingegebenen Segment ein- bzw. auszuschalten. Ist diese Option eingeschaltet, werden Winkel zwischen dem zuletzt eingegebenen Segment und der Position des Mauszeigers gemessen.



Klicken Sie auf die Schaltflächen  links der Eingabefelder *x*, *y*, *z* oder *m* (oder drücken Sie **Umschalt+<key>**), um die Koordinateneingabe relativ zum zuletzt gesetzten Stützpunkt ein- oder auszuschalten. Sind diese Optionen eingeschaltet, werden die Koordinaten so berechnet, als sei der letztgesetzte Stützpunkt der Koordinatenursprung.




## Laufendes Beibehalten

Sowohl beim Digitalisieren mit absoluter als auch mit relativer Referenz können Sie den Winkel, die Entfernung, X, Y, Z und M Einschränkungen laufend beibehalten, indem auf die Schaltflächen  *laufend beibehalten* klicken. Damit können Sie mehrere Punkte oder Stützpunkte digitalisieren und dabei die selben Einschränkungen benutzen.

## Parallele und senkrechte Linien

Alle bisher beschriebenen Werkzeuge können Sie mit den Werkzeugen  Senkrecht und  Parallel zusammen benutzen. Mit diese beiden Werkzeuge zeichnen Sie Segmente exakt rechtwinklig oder parallel zu anderen Segmenten. Das entsprechende Segment kann zu einem anderen Layer, zu einem anderen Objekt im selben Layer oder auch zu dem Objekt, an dem Sie gerade arbeiten, gehören (letzteres benötigt *Selbsteinrasten*).

So zeichnen Sie ein *senkrecht*es Segment:

1. Fügen Sie zunächst einen der beiden Stützpunkte des Segments hinzu.
2. Zum Aktivieren klicken Sie auf die Schaltfläche  Senkrecht (Tastenkürzel P , senkrecht engl. perpendicular).
3. Klicken Sie dann auf das Segment, auf dem Ihr neues senkrecht stehen soll.
4. Eine gepunktete Hilfslinie senkrecht zu diesem Segment durch den zuletzt gesetzten Stützpunkt erscheint. Die Winkeleinstellung ist gesperrt, wodurch der nächste Stützpunkt auf dieser Linie liegen muss und ein Kreuz zeigt Ihnen die zukünftige Position an. Klicken Sie, um einen neuen Stützpunkt zu setzen.

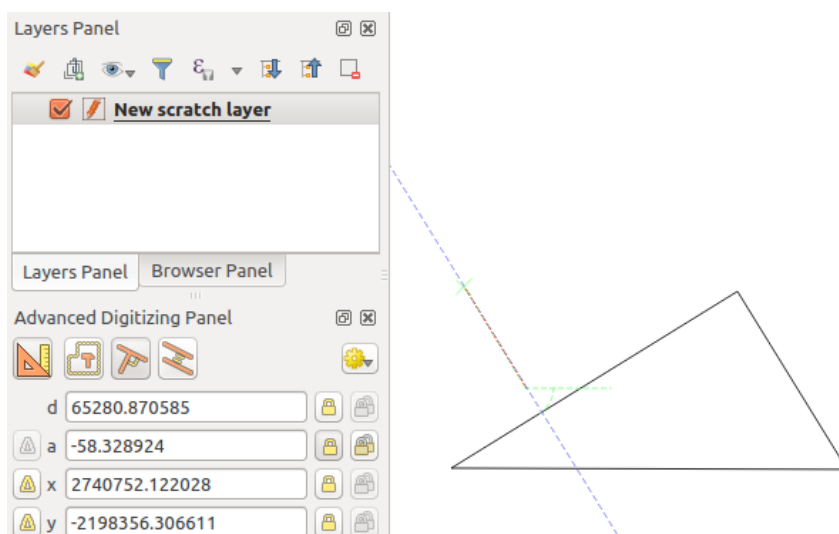



Abb. 16.93: Senkrecht Digitalisieren

Ein *paralleles* Segment zeichnen Sie genau so mit der Ausnahme, dass Sie auf die Schaltfläche  Parallel (Tastenkürzel zweimal P).

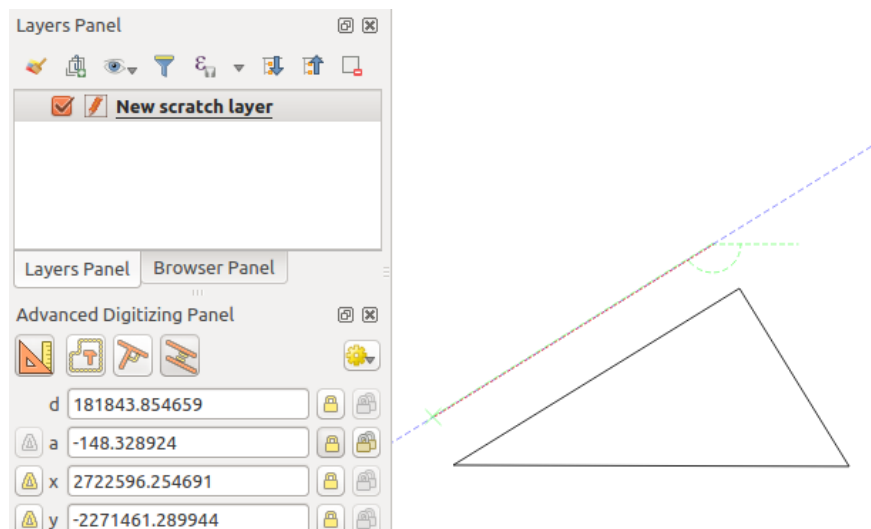





Abb. 16.94: Paralleles Digitalisieren

Beide Werkzeuge stellen den richtigen Winkel für paralleles und senkrecht arbeiten fest und sperren ihn, während Sie digitalisieren. Entsperren Sie den Winkel, um die Benutzung zwischendurch abubrechen.

## Konstruktionsmodus

Sie können den *Konstruktionsmodus* an- und abschalten, indem Sie auf die Schaltfläche  *Konstruktionsmodus* klicken oder mit Tastenkürzel **C**. Während Sie sich im *Konstruktionsmodus* befinden, werden durch Klicks in die Kartenansicht keine neuen Stützpunkte hinzugefügt; stattdessen werden die Klickpositionen erfasst, so dass Sie sie als Referenzpunkte benutzen können, um die relativen Werte der Entfernung, des Winkels oder von X, Y, Z, M zu sperren.

Der *Konstruktionsmodus* kann z.B. benutzt werden, um einen Punkt in einer genau festgelegten Entfernung von einem bestehenden Punkt zu zeichnen.

Haben Sie einen Punkt in der Kartenansicht und der *Einrastmodus* ist richtig eingeschaltet, können Sie sehr einfach andere Punkte in einer vorgegebenen Entfernung und im vorgegebenen Winkel dazu zeichnen. Zusätzlich zur Schaltfläche  müssen Sie den *Konstruktionsmodus* durch Klick auf die Schaltfläche  *Konstruktionsmodus* oder Tastenkürzel **C** aktivieren.

Klicken Sie in die Nähe des Punktes, von dem aus Sie den Abstand berechnen wollen und klicken Sie auf die Eingabe *d* (Tastenkürzel **D**). Geben Sie den gewünschten Abstand ein und drücken Sie **Eingabe**, um die Mausposition im Kartenfenster beizubehalten:

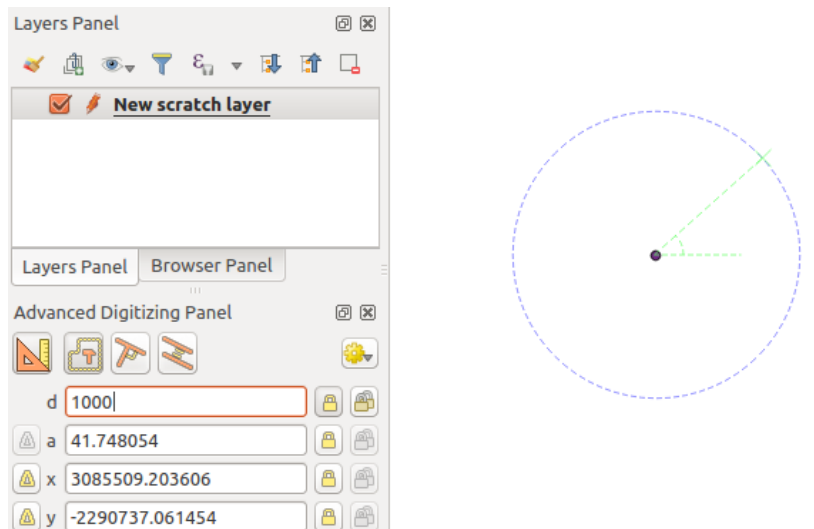



Abb. 16.95: Abstand von einem Punkt

Bevor Sie den neuen Punkt eingeben, drücken Sie **C**, um den Konstruktionsmodus zu verlassen. Jetzt können Sie ins Kartenfenster klicken und der Punkt wird mit dem eingegebenen Abstand platziert.

Sie können zusätzlich die Winkeleinstellung benutzen, um beispielsweise einen zweiten Punkt in der selben Entfernung von Ihrem Originalpunkt aber in einem bestimmten Winkel zu Ihrem eben neu gesetzten Punkt zu erzeugen. Klicken Sie auf die Schaltfläche  Konstruktionsmodus oder Tastenkürzel **C**, um in den Konstruktionsmodus zu gelangen. Klicken Sie auf den eben neu gesetzten Punkt und dann auf den anderen, um ein Richtungssegment zu erhalten. Klicken Sie dann in die Texteingabe *d* (Tastenkürzel **D**), geben Sie die gewünschte Entfernung ein und drücken Sie **Eingabe**. Klicken Sie auf die Texteingabe *a* (Tastenkürzel **A**), geben Sie den gewünschten Winkel ein und drücken Sie **Eingabe**. Die Mausposition wird in Entfernung und Winkel gesperrt.

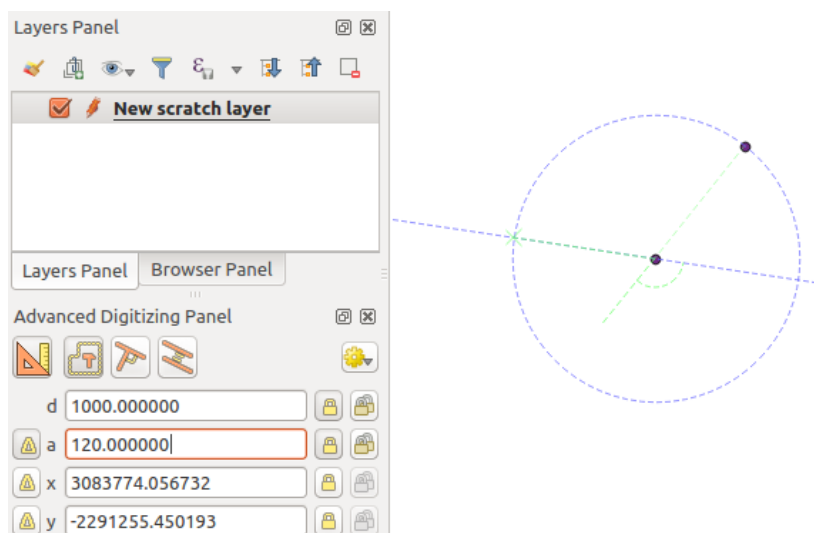


Abb. 16.96: Abstand von und Winkel zu einem Punkt

Bevor Sie den neuen Punkt hinzufügen, drücken Sie **C**, um den Konstruktionsmodus zu verlassen. Jetzt können Sie in das Kartenfenster klicken und der Punkt wird in dem eingegebenen Abstand und Winkel erzeugt. Sie können den Vorgang wiederholen, um mehrere Punkte zu erzeugen.

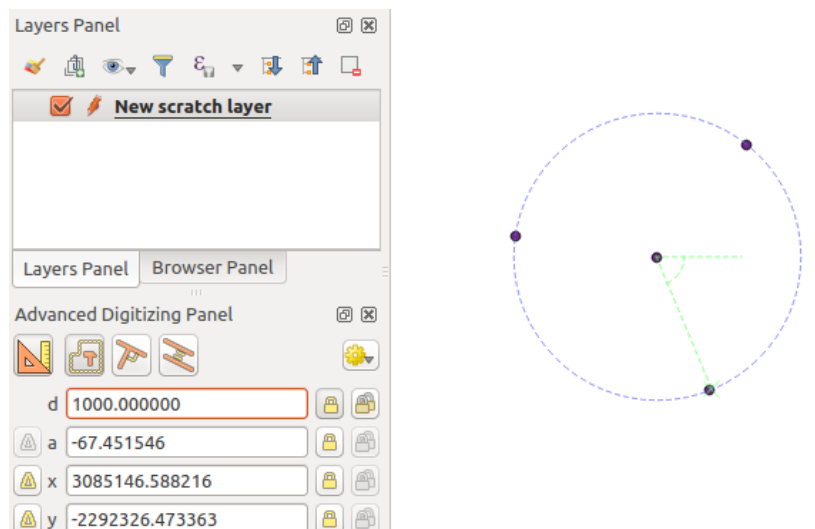



Abb. 16.97: Punkte in festgelegtem Abstand und Winkel

### 16.3.8 Mit Verarbeitung Objekte innerhalb eines Layers verändern

Das Menü *Verarbeitung* bietet eine große Zahl von Werkzeugen zur Analyse und Erzeugung neuer Objekte auf der Basis der Eigenschaften vorhandener Objekte oder ihrer Beziehung zu anderen Objekten (innerhalb oder außerhalb des selben Layers). Während üblicherweise als Ergebnis neue Layer erzeugt werden, erlauben es manche Algorithmen, den Eingabelayer selbst zu verändern. Das ist praktisch, um mehrfache Objektänderungen mittels fortschrittlicher und komplexer Methoden zu automatisieren.

So ändern Sie Objekte insitu:

1. Wählen Sie den Layer, den Sie bearbeiten möchten, im Bedienfeld *Layer* aus.
2. Wählen Sie die entsprechenden Objekte aus. Diesen Schritt können Sie überspringen, dann werden die Änderungen auf den ganzen Layer angewendet.
3. Drücken Sie die Schaltfläche  *Objekt insitu ändern* oben in *Processing toolbox*. Die Liste der Algorithmen ist gefiltert, es werden nur diejenigen angezeigt, die mit der insitu-Verarbeitung funktionieren, d.h.:
  - Sie arbeiten an der Objektquelle und nicht auf Ebene des Layers.
  - Sie ändern die Struktur des Layers nicht, z.B. Felder anlegen oder entfernen.
  - Sie ändern den Geometrietyp nicht, z.B. von Linie zu Punkt.

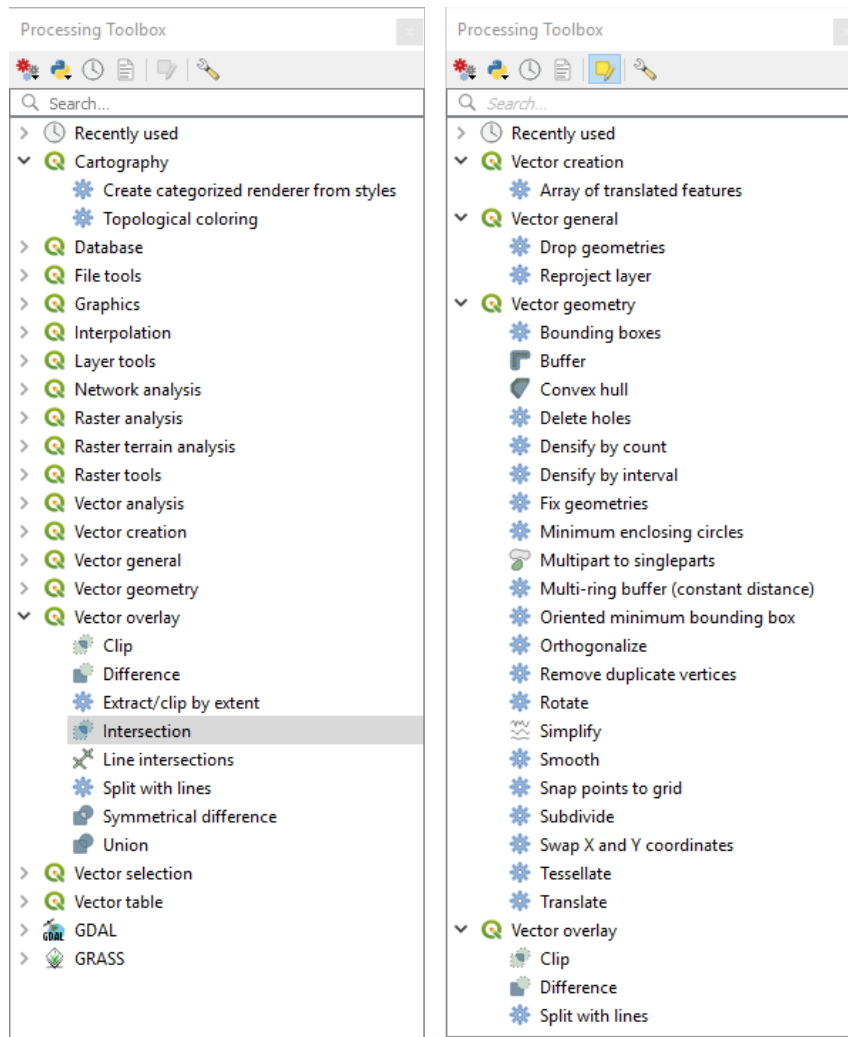





Abb. 16.98: Verarbeitungswerkzeuge: alle (links) und insitu-Änderung (rechts)

4. Wählen Sie den Algorithmus, den Sie ausführen wollen und doppelklicken Sie darauf.

**Bemerkung:** Falls der Algorithmus keine weiteren Nutzereingaben erfordert (abgesehen von Eingabelayer und Ausgabelayer), wird er sofort ausgeführt, ohne dass sich das Eingabefenster öffnet.

1. Falls der Algorithmus andere Nutzereingaben als Ein- und Ausgabelayer erfordert, öffnet sich sein Eingabefenster. Geben Sie alle Pflichtangaben ein.
2. Klicken Sie auf *Gewählte Objekte ändern* oder *Alle Objekte ändern* je nachdem, ob es eine aktive Auswahl gibt.

Die Änderungen werden auf den Layer angewendet und im Editierspeicher festgehalten: tatsächlich wird der Bearbeitungsstatus des Layers angeschaltet, wie Sie am Symbol  neben dem Layernamen erkennen können.

5. Drücken Sie wie üblich  Layeränderungen speichern, um die Änderungen im Layer zu speichern. Sie können stattdessen auch auf  Rückgängig klicken, um alle Änderungen rückgängig zu machen.

## 16.4 Connecting and Editing Data Across Layers

Ability to connect data from different layers is one of the duties of a GIS software. Such a connection can be based on the spatial relationship between the features, or on their shared attributes. QGIS provides tools to handle any of these associations, such as:



- Processing algorithms that can create a new layer as a result of the connection, namely *Join attributes by location*, *Join attributes by nearest*, *Join attributes by field value*, ...
- SQL queries to create a new layer from the *DB Manager* or as a *virtual layer*
- *Joins properties* or *relations settings* that temporarily extend attributes of features in a given layer, with those of features in another layer based on some matching attribute(s).

Joins and relations are technical concepts borrowed from databases to get the most out of data stored in tables by combining their contents. The idea is that features (rows) of different layers (tables) can be associated to each other. The number of rows which are matching each other can be of any value (zero, one, many).

### 16.4.1 Joining features between two layers

**Joins** in QGIS allow you to associate features of the current layer to features from another loaded vector layer. Whether they are spatially enabled and the type of geometry do not matter. The join is based on an attribute that is shared by the layers, in a one-to-one relationship.

To create a join on a layer (identified below as `target layer`):

1. Go to the layer *Properties* ►  *Joins* tab
2. Click the  *Add new join* button. The *Add vector join* dialog appears.
3. Select the *Join layer* you want to connect with the target vector layer
4. Specify the *Join field* (from the `join layer`) and the *Target field* (from the `target layer`). These are the fields that are used to find matching feature in both layers hence they should have values in common.
5. Press *OK* and a summary of selected parameters is added to the *Join* panel.

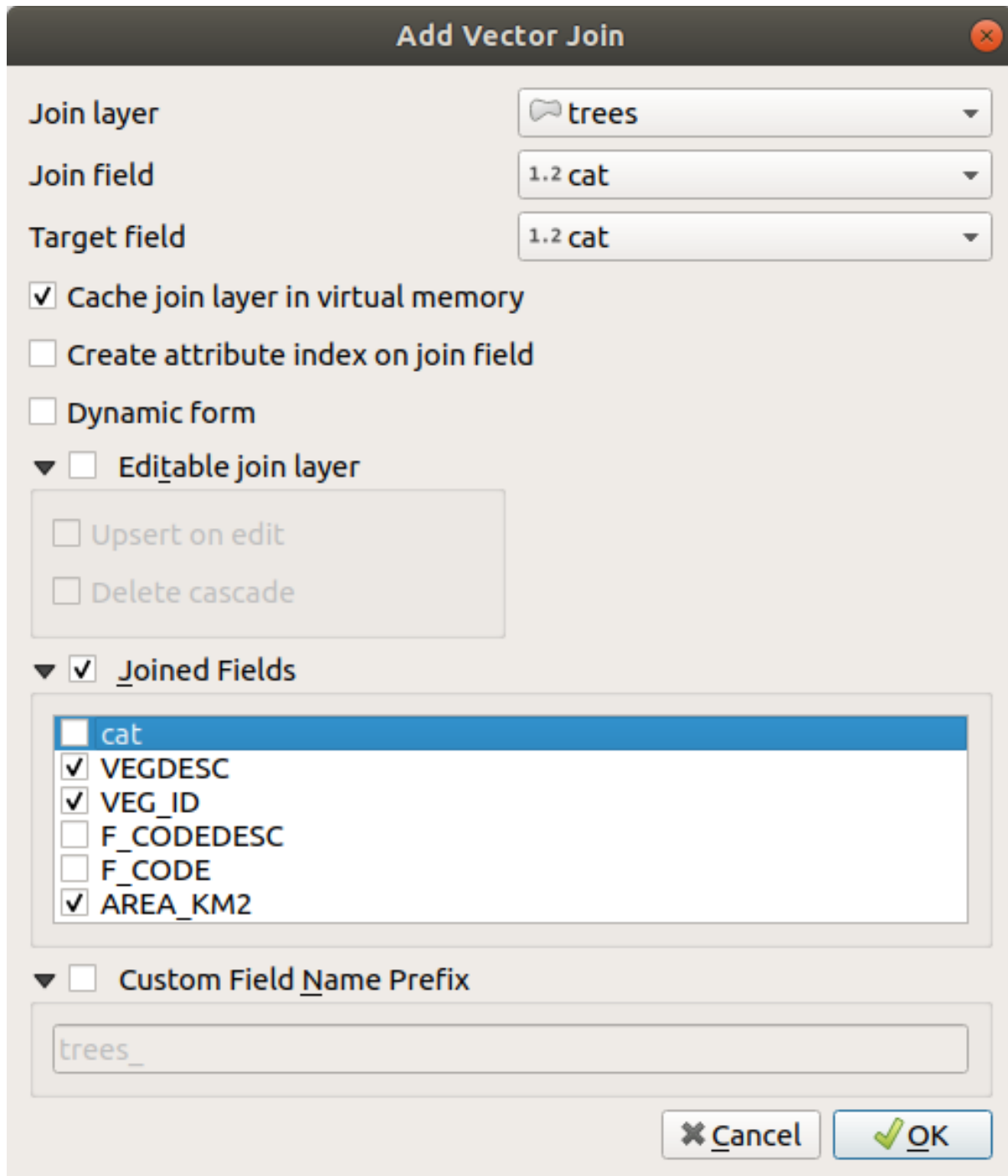


Abb. 16.99: Join an attribute table to an existing vector layer

The steps above will create a join, where **ALL** the attributes of the **first matching feature** in the join layer is added to the target layer's feature. The following logic is used to pair features during a join process:












- All the features in the target layer are returned, regardless they have a match
- If the target field contains duplicate values, these features are assigned the same feature from the join layer.
- If the join field contains duplicate matching values, only the first fetched feature is picked.

**Bemerkung:** Joins in QGIS are based on a single field matching so most of the times, you would want to make sure

that values in the matchable fields are unique.

---


QGIS provides some more options to tweak the join:

-  *Cache join layer in virtual memory*: allows you to cache values in memory (without geometries) from the joined layer in order to speed up lookups.
-  *Create attribute index on the join field* to speed up lookups
-  *Dynamic form*: helps to synchronize join fields on the fly, according to the *Target field*. This way, constraints for join fields are also correctly updated. Note that it's deactivated by default because it may be very time consuming if you have a lot of features or a myriad of joins.
- If the target layer is editable, then some icons will be displayed in the attribute table next to fields, in order to inform about their status:
  - : the join layer is not configured to be editable. If you want to be able to edit join features from the target attribute table, then you have to check the option  *Editable join layer*.
  - : the join layer is well configured to be editable, but its current status is read only.
  - : the join layer is editable, but synchronization mechanisms are not activated. If you want to automatically add a feature in the join layer when a feature is created in the target layer, then you have to check the option  *Upsert on edit*. Symmetrically, the option  *Delete cascade* may be activated if you want to automatically delete join features.
-  *Joined fields*: instead of adding all the fields from the joined layer, you can specify a subset.
-  *Custom field name prefix* for joined fields, in order to avoid name collision

### 16.4.2 Setting relations between multiple layers

Unlike joins that define a one-to-one link between features across two layers, relations help you build interconnections between multiple features across two or more layers. As such, relations are project level settings and are set in *Project*



► *Properties* ►  *Relations* tab. From there, you can:

-  *Add relation* whose type can be:
  - *one to many relation*
  - *many to many relation*
  - *polymorphic relation* that you can add or edit with the dedicated tools in the action drop-down menu.

---

**Bemerkung:** There is no simple way yet to edit a non-polymorphic relation once it has been created. Only the name can be edited with a double-click. For any other parameters of such a relation you will have to remove and recreate it.

---

-  *Discover relations*: QGIS is able to discover existing relations from supported database formats (PostgreSQL, GeoPackage, ESRI File Geodatabase, ...). This can be a good way to ease the relations definition.
-  *Remove relation*



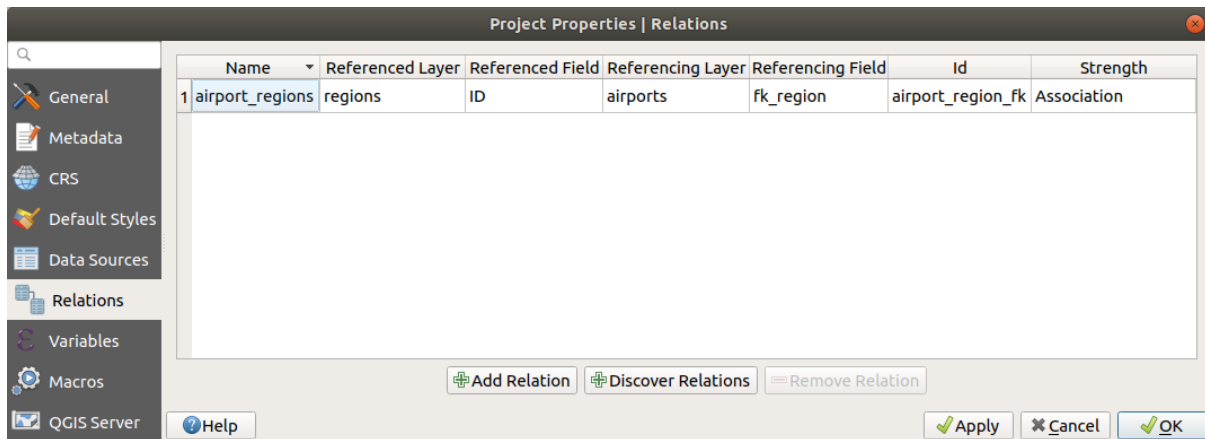


Abb. 16.100: Relations tab

### One to many (1-N) relations

As an example you have a layer with all regions of Alaska (polygon) which provides some attributes about its name and region type and a unique id (which acts as primary key).

Then you get another point layer or table with information about airports that are located in the regions and you also want to keep track of these. If you want to add them to the regions layer, you need to create a one to many relation using foreign keys, because there are several airports in most regions.



Abb. 16.101: Alaska region with airports

### Layers and keys

QGIS makes no difference between a table and a vector layer. Basically, a vector layer is a table with a geometry. So you can add your table as a vector layer. To demonstrate the 1-n relation, you can load the *regions* and *airports* layers in the sample dataset. In practice, each airport belongs to exactly one region while each region can have any number of airports (a typical one to many relation).


which has a foreign key field (*fk\_region*) to the layer *regions*.

In addition to the attributes describing the airports, the airports layer has another field *fk\_region* which acts as a foreign key (if you have a database, you will probably want to define a constraint on it). This *fk\_region* field will always contain an id of a region. It can be seen like a pointer to the region it belongs to.


All you have to do is to tell QGIS the relation between the layers so that you can design a custom edit form for editing and QGIS takes care of the setup. It works with different providers (so you can also use it with shape and csv files).

## Defining 1-N relations

The first thing we are going to do is to let QGIS know about the relations between the layers. This is done in *Project*

► *Properties...* Open the *Relations* tab and click on  *Add Relation*.

- **Name** is going to be used as a title. It should be a human readable string describing what the relation is used for. We will just call say **airport\_relation** in this case.
- **Referenced Layer (Parent)** also considered as parent layer, is the one with the primary key, pointed to, so here it is the `regions` layer. You need to define the primary key of the referenced layer, so it is `ID`.
- **Referencing Layer (Child)** also considered as child layer, is the one with the foreign key field on it. In our case, this is the `airports` layer. For this layer you need to add a referencing field which points to the other layer, so this is `fk_region`.

**Bemerkung:** Sometimes, you need more than a single field to uniquely identify features in a layer. Creating a relation with such a layer requires a **composite key**, i.e. more than a single pair of matching fields. Use the  Add new field pair as part of a composite foreign key button to add as many pairs as necessary.

- **Id** will be used for internal purposes and has to be unique. You may need it to build *custom forms*. If you leave it empty, one will be generated for you but you can assign one yourself to get one that is easier to handle
- **Relationship strength** sets the strength of the relation between the parent and the child layer. The default *Association* type means that the parent layer is *simply* linked to the child one while the *Composition* type allows you to duplicate also the child features when duplicating the parent ones and on deleting a feature the children are deleted as well, resulting in cascade over all levels (means children of children of... are deleted as well).

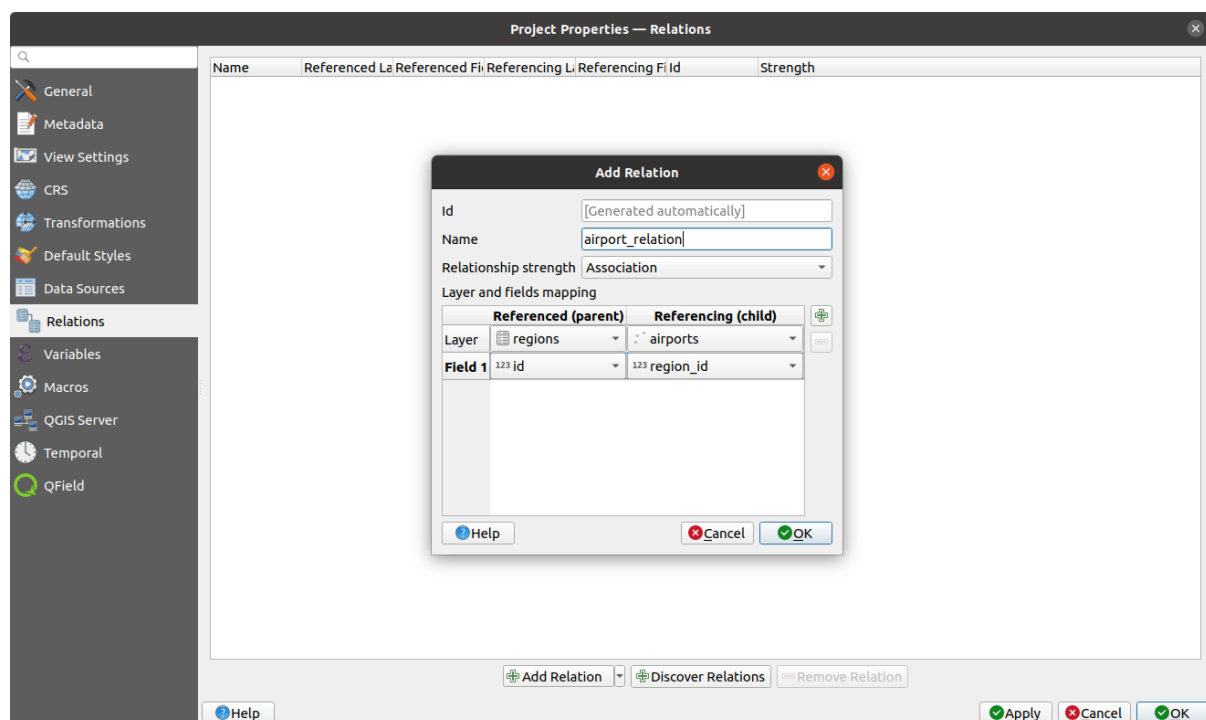


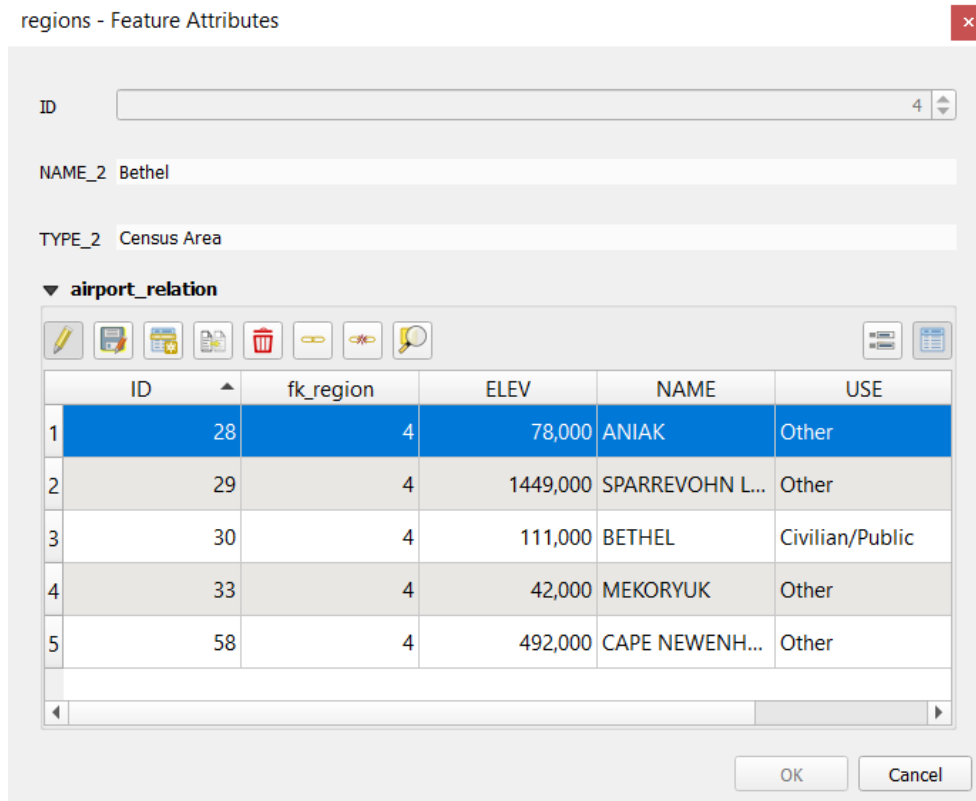
Abb. 16.102: Adding a relation between regions and airports layers

From the *Relations* tab, you can also press the  *Discover Relation* button to fetch the relations available from the

providers of the loaded layers. This is possible for layers stored in data providers like PostgreSQL or SpatiaLite.

## Forms for 1-N relations

Now that QGIS knows about the relation, it will be used to improve the forms it generates. As we did not change the default form method (autogenerated), it will just add a new widget in our form. So let's select the layer region in the legend and use the identify tool. Depending on your settings, the form might open directly or you will have to choose to open it in the identification dialog under actions.



regions - Feature Attributes

ID: 4

NAME\_2: Bethel

TYPE\_2: Census Area







▼ airport\_relation






	ID	fk_region	ELEV	NAME	USE
1	28	4	78,000	ANIAK	Other
2	29	4	1449,000	SPARREVOHN L...	Other
3	30	4	111,000	BETHEL	Civilian/Public
4	33	4	42,000	MEKORYUK	Other
5	58	4	492,000	CAPE NEWENH...	Other

OK Cancel

Abb. 16.103: Identification dialog regions with relation to airports

As you can see, the airports assigned to this particular region are all shown in a table. And there are also some buttons available. Let's review them shortly:

- The  button is for toggling the edit mode. Be aware that it toggles the edit mode of the airport layer, although we are in the feature form of a feature from the region layer. But the table is representing features of the airport layer.
- The  button is for saving all the edits in the child layer (airport).
- The  button lets you digitize the airport geometry in the map canvas and assigns the new feature to the current region by default. Note that the icon will change according to the geometry type.
- The  button adds a new record to the airport layer attribute table and assigns the new feature to the current region by default. The geometry can be drawn later with the *Add part* digitizing tool.
- The  button allows you to copy and paste one or more child features within the child layer. They can later be assigned to a different parent feature or have their attributes modified.
- The  button deletes the selected airport(s) permanently.

- The  symbol opens a new dialog where you can select any existing airport which will then be assigned to the current region. This may be handy if you created the airport on the wrong region by accident.
- The  symbol unlinks the selected airport(s) from the current region, leaving them unassigned (the foreign key is set to NULL) effectively.
- With the  button you can zoom the map to the selected child features.
- The two buttons  and  to the right switch between the *table view* and *form view* of the related child features.

If you use the *Drag and Drop Designer* for the regions feature, you can select which tools are available. You can even decide whether to open a new form when a new feature is added using *Force hide form on add feature* option. Be aware that this option implies that not null attributes must take a valid default value to work correctly.

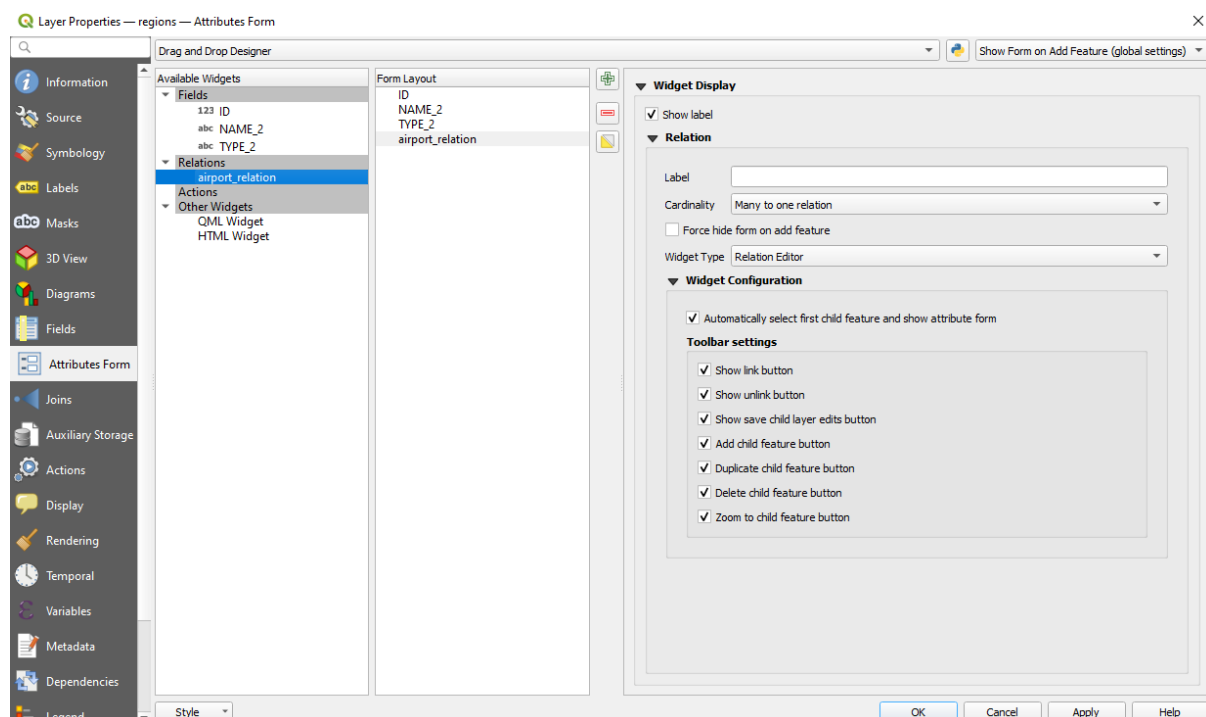





Abb. 16.104: Drag and Drop Designer for configure regions-airports relation tools

In the above example the referencing layer has geometries (so it isn't just an alphanumeric table) so the above steps will create an entry in the layer attribute table that has no corresponding geometric feature. To add the geometry:

1. Choose  *Open Attribute Table* for the referencing layer.
2. Select the record that has been added previously within the feature form of the referenced layer.
3. Use the  *Add Part* digitizing tool to attach a geometry to the selected attributes table record.

If you work on the airport table, the widget Relation Reference is automatically set up for the `fk_region` field (the one used to create the relation), see *Relation Reference widget*.

In the airport form you will see the  button at the right side of the `fk_region` field: if you click on the button the form of the region layer will be opened. This widget allows you to easily and quickly open the forms of the linked parent features.



airports - Feature Attributes

ID 28

fk\_region Bethel

ELEV 78,000

NAME ANIAC

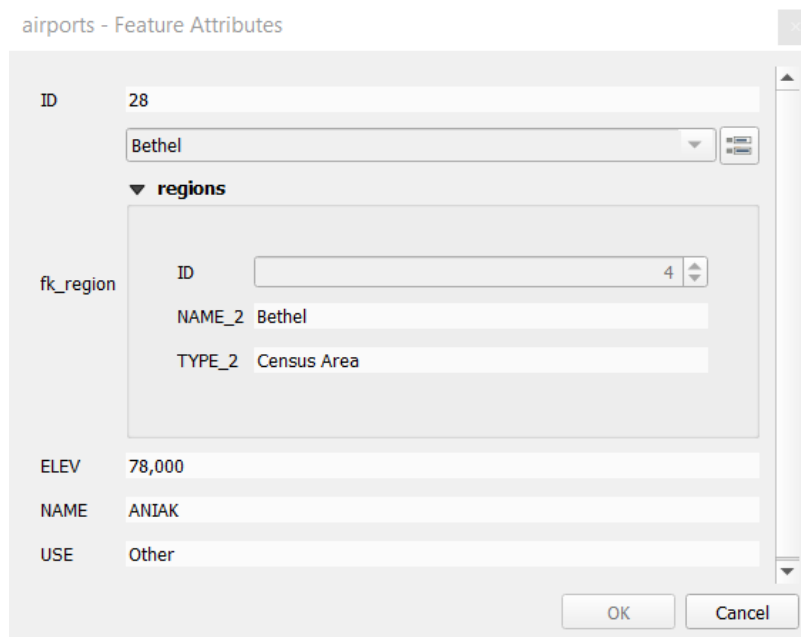
USE Other

OK Cancel

Abb. 16.105: Identification dialog airport with relation to regions

The Relation Reference widget has also an option to embed the form of the parent layer within the child one. It is available in the *Properties* ► *Attributes Form* menu of the airport layer: select the `fk_region` field and check the *Show embedded form* option.

If you look at the feature dialog now, you will see that the form of the region is embedded inside the airports form and will even have a combobox, which allows you to assign the current airport to another region.



airports - Feature Attributes

ID 28

Bethel

▼ regions

fk\_region

ID 4

NAME\_2 Bethel


TYPE\_2 Census Area

ELEV 78,000

NAME ANIAC

USE Other

OK Cancel

Moreover if you toggle the editing mode of the airport layer, the `fk_region` field has also an autocompleter function: while typing you will see all the values of the `id` field of the region layer. Here it is possible to digitize a polygon for the region layer using the  button if you chose the option *Allow adding new features* in the *Properties* ► *Attributes Form* menu of the airport layer.

The child layer can also be used in the *Select Features By Value* tool in order to select features of the parent layer based on attributes of their children.

In Abb. 16.106, all the regions where the mean altitude of the airports is greater than 500 meters above sea level are selected.

You will find that many different aggregation functions are available in the form.

Abb. 16.106: Select parent features with child values

## Many-to-many (N-M) relations

N-M relations are many-to-many relations between two tables. For instance, the `airports` and `airlines` layers: an airport receives several airline companies and an airline company flies to several airports.

This SQL code creates the three tables we need for an N-M relationship in a PostgreSQL/PostGIS schema named `locations`. You can run the code using the *Database ► DB Manager...* for PostGIS or external tools such as [pgAdmin](#). The `airports` table stores the `airports` layer and the `airlines` table stores the `airlines` layer. In both tables few fields are used for clarity. The *tricky* part is the `airports_airlines` table. We need it to list all airlines for all airports (or vice versa). This kind of table is known as a *pivot table*. The *constraints* in this table force that an airport can be associated with an airline only if both already exist in their layers.

```
CREATE SCHEMA locations;

CREATE TABLE locations.airports
(
  id serial NOT NULL,
  geom geometry(Point, 4326) NOT NULL,
  airport_name text NOT NULL,
  CONSTRAINT airports_pkey PRIMARY KEY (id)
);

CREATE INDEX airports_geom_idx ON locations.airports USING gist (geom);

CREATE TABLE locations.airlines
(
  id serial NOT NULL,
  geom geometry(Point, 4326) NOT NULL,
  airline_name text NOT NULL,
  CONSTRAINT airlines_pkey PRIMARY KEY (id)
);

CREATE INDEX airlines_geom_idx ON locations.airlines USING gist (geom);
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```

CREATE TABLE locations.airports_airlines
(
    id serial NOT NULL,
    airport_fk integer NOT NULL,
    airline_fk integer NOT NULL,
    CONSTRAINT airports_airlines_pkey PRIMARY KEY (id),
    CONSTRAINT airports_airlines_airport_fk_fkey FOREIGN KEY (airport_fk)
        REFERENCES locations.airports (id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
        DEFERRABLE INITIALLY DEFERRED,
    CONSTRAINT airports_airlines_airline_fk_fkey FOREIGN KEY (airline_fk)
        REFERENCES locations.airlines (id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
        DEFERRABLE INITIALLY DEFERRED
);

```

Instead of PostgreSQL you can also use GeoPackage. In this case, the three tables can be created manually using the *Database ► DB Manager....* In GeoPackage there are no schemas so the *locations* prefix is not needed.

Foreign key constraints in *airports\_airlines* table can't be created using *Table ► Create Table...* or *Table ► Edit Table...* so they should be created using *Database ► SQL Window....* GeoPackage doesn't support *ADD CONSTRAINT* statements so the *airports\_airlines* table should be created in two steps:

1. Set up the table only with the *id* field using *Table ► Create Table...*
2. Using *Database ► SQL Window...*, type and execute this SQL code:

```

ALTER TABLE airports_airlines
    ADD COLUMN airport_fk INTEGER
    REFERENCES airports (id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    DEFERRABLE INITIALLY DEFERRED;

ALTER TABLE airports_airlines
    ADD COLUMN airline_fk INTEGER
    REFERENCES airlines (id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    DEFERRABLE INITIALLY DEFERRED;

```

Then in QGIS, you should set up two *one-to-many relations* as explained above:

- a relation between *airlines* table and the pivot table;
- and a second one between *airports* table and the pivot table.

An easier way to do it (only for PostgreSQL) is using the *Discover Relations* in *Project ► Properties ► Relations*. QGIS will automatically read all relations in your database and you only have to select the two you need. Remember to load the three tables in the QGIS project first.

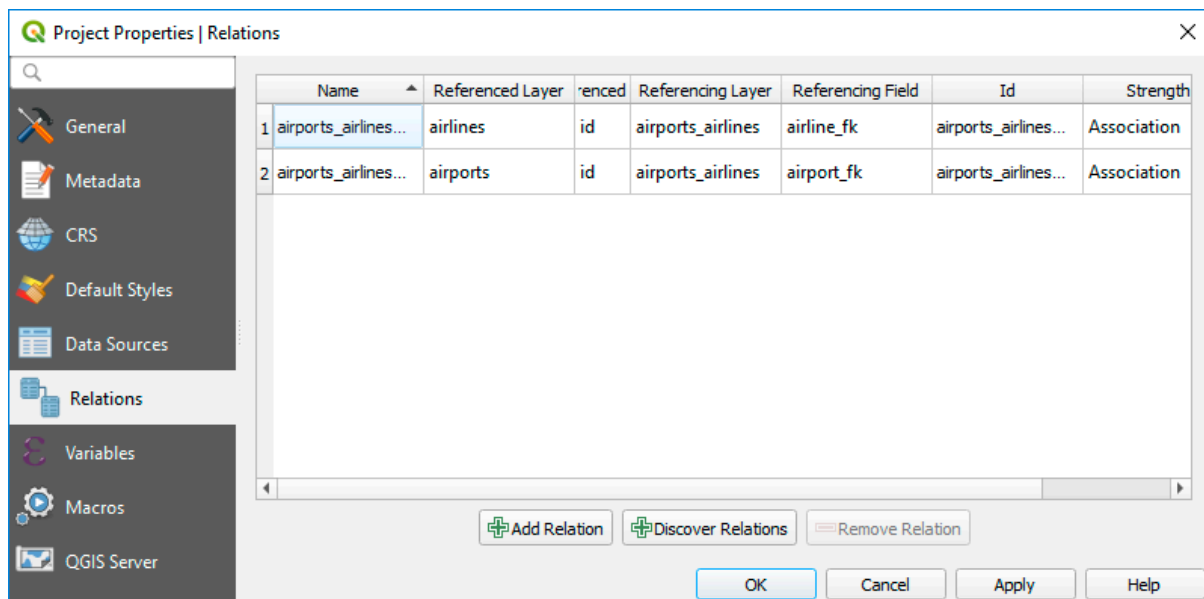


Abb. 16.107: Relations and autodiscover

In case you want to remove an airport or an airline, QGIS won't remove the associated record(s) in `airports_airlines` table. This task will be made by the database if we specify the right *constraints* in the pivot table creation as in the current example.

#### Bemerkung: Combining N-M relation with automatic transaction group

You should enable the transaction mode in *Project Properties* ► *Data Sources* ► when working on such context. QGIS should be able to add or update row(s) in all tables (airlines, airports and the pivot tables).

Finally we have to select the right cardinality in the *Layer Properties* ► *Attributes Form* for the `airports` and `airlines` layers. For the first one we should choose the **airlines (id)** option and for the second one the **airports (id)** option.

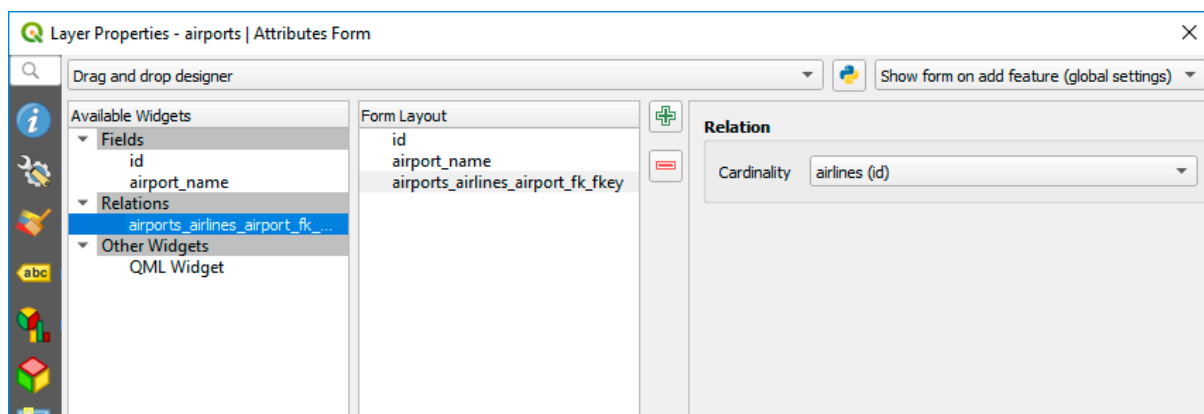


Abb. 16.108: Set relationship cardinality

Now you can associate an airport with an airline (or an airline with an airport) using *Add child feature* or *Link existing child feature* in the subforms. A record will automatically be inserted in the `airports_airlines` table.



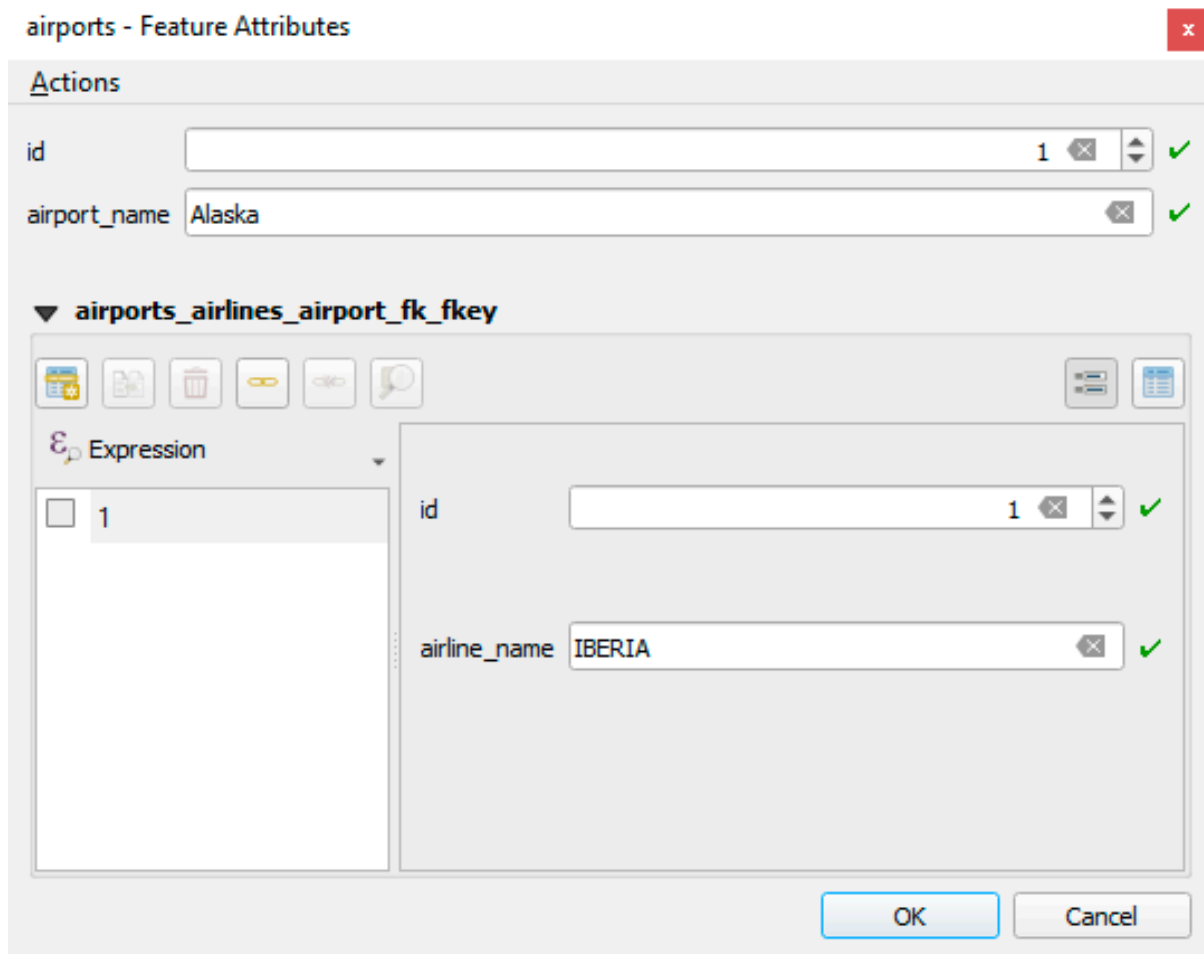


Abb. 16.109: N-M relationship between airports and airlines

**Bemerkung:** Using **Many to one relation** cardinality

Sometimes hiding the pivot table in an N-M relationship is not desirable. Mainly because there are attributes in the relationship that can only have values when a relationship is established. If your tables have a geometry field, it could be interesting to activate the *On map identification* option (*Layer Properties* ► *Attributes Form* ► *Available widgets* ► *Fields*) for the foreign key fields in the pivot table.

**Bemerkung:** Pivot table primary key

Avoid using multiple fields in the primary key in a pivot table. QGIS assumes a single primary key so a constraint like `constraint airports_airlines_pkey primary key (airport_fk, airline_fk)` will not work.

### Polymorphic relations

#### The purpose


Polymorphic relations are special case of 1-N relations, where a single referencing (document) layer contains the features for multiple referenced layers. This differs from normal relations which require different referencing layer for each referenced layer. A single referencing (document) layer is achieved by adding an additional `layer_field` column in the referencing (document) layer that stores information to identify the referenced layer. In its most simple form, the referencing (document) layer will just insert the layer name of the referenced layer into this field.

To be more precise, a polymorphic relation is a set of normal relations having the same referencing layer but having the referenced layer dynamically defined. The polymorphic setting of the layer is solved by using an expression which has to match some properties of the referenced layer like the table name, layer id, layer name.

Imagine we are going to the park and want to take pictures of different species of `plants` and `animals` we see there. Each plant or animal has multiple pictures associated with it, so if we use the normal 1:N relations to store pictures, we would need two separate tables, `animal_images` and `plant_images`. This might not be a problem for 2 tables, but imagine if we want to take separate pictures for mushrooms, birds etc.

Polymorphic relations solve this problem as all the referencing features are stored in the same table `documents`. For each feature the referenced layer is stored in the `referenced_layer` field and the referenced feature id in the `referenced_fk` field.

#### Defining polymorphic relations

First, let QGIS know about the polymorphic relations between the layers. This is done in *Project ► Properties....* Open the *Relations* tab and click on the little down arrow next to the  *Add Relation* button, so you can select the *Add Polymorphic Relation* option from the newly appeared dropdown.

**Add Polymorphic Relation**

Id: [Generated automatic...]

Referencing layer: documents

Layer field: abc referenced\_layer

Layer field expression: @layer\_name  $\epsilon$

Relationship strength: Association

Referenced layers: plants, animals

	Referenced (parent)	Referencing (child)
Field 1	fid	abc referenced_id

Polymorphic relations are for advanced usage where a set of standard relations have the same referencing layer but the referenced layer is calculated from an expression.

Buttons: ? Help, Cancel, OK

Abb. 16.110: Adding a polymorphic relation using documents layer as referencing and animals and plants as referenced layers.

- **Id** will be used for internal purposes and has to be unique. You may need it to build *custom forms*. If you leave it empty, one will be generated for you but you can assign one yourself to get one that is easier to handle
- **Referencing Layer (Child)** also considered as child layer, is the one with the foreign key field on it. In our case, this is the documents layer. For this layer you need to add a referencing field which points to the other layer, so this is referenced\_fk.

**Bemerkung:** Sometimes, you need more than a single field to uniquely identify features in a layer. Creating a relation with such a layer requires a **composite key**, i.e. more than a single pair of matching fields. Use the



Add new field pair as part of a composite foreign key button to add as many pairs as necessary.

- **Layer Field** is the field in the referencing table that stores the result of the evaluated layer expression which is the referencing table that this feature belongs to. In our example, this would be the referenced\_layer field.
- **Layer expression** evaluates to a unique identifier of the layer. This can be the layer name @layer\_name, the layer id @layer\_id, the layer's table name decode\_uri(@layer, 'table') or anything that can uniquely identifies a layer.

- **Relationship strength** sets the strength of the generated relations between the parent and the child layer. The default *Association* type means that the parent layer is *simply* linked to the child one while the *Composition* type allows you to duplicate also the child features when duplicating the parent ones and on deleting a feature the children are deleted as well, resulting in cascade over all levels (means children of children of ... are deleted as well).
- **Referenced Layers** also considered as parent layers, are those with the primary key, pointed to, so here they would be `plants` and `animals` layers. You need to define the primary key of the referenced layers from the dropdown, so it is `fid`. Note that the definition of a valid primary key requires all the referenced layers to have a field with that name. If there is no such field you cannot save a polymorphic relation.

Once added, the polymorphic relation can be edited via the *Edit Polymorphic Relation* menu entry.

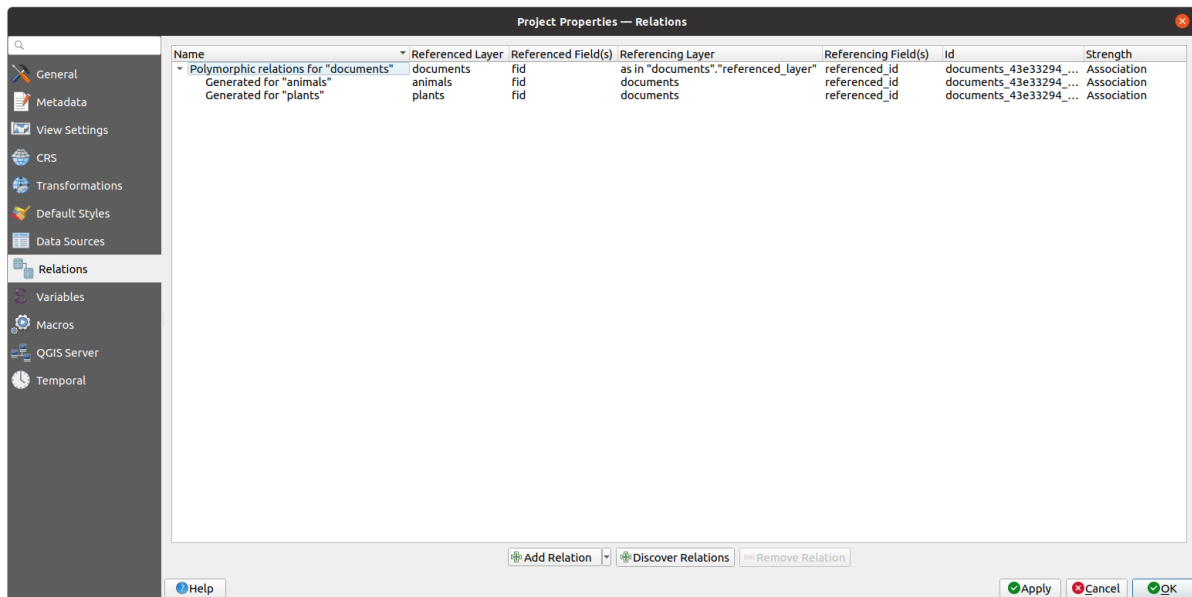


Abb. 16.111: Preview of the newly created polymorphic relation and its child relations for animals and plants.

The example above uses the following database schema:

```
CREATE SCHEMA park;

CREATE TABLE park.animals
(
    fid serial NOT NULL,
    geom geometry(Point, 4326) NOT NULL,
    animal_species text NOT NULL,
    CONSTRAINT animals_pkey PRIMARY KEY (fid)
);

CREATE INDEX animals_geom_idx ON park.animals USING gist (geom);

CREATE TABLE park.plants
(
    fid serial NOT NULL,
    geom geometry(Point, 4326) NOT NULL,
    plant_species text NOT NULL,
    CONSTRAINT plants_pkey PRIMARY KEY (fid)
);

CREATE INDEX plants_geom_idx ON park.plants USING gist (geom);

CREATE TABLE park.documents
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
(  
  fid serial NOT NULL,  
  referenced_layer text NOT NULL,  
  referenced_fk integer NOT NULL,  
  image_filename text NOT NULL,  
  CONSTRAINT documents_pkey PRIMARY KEY (fid)  
);
```



### 17.1 Dialogfenster Rasterlayereigenschaften

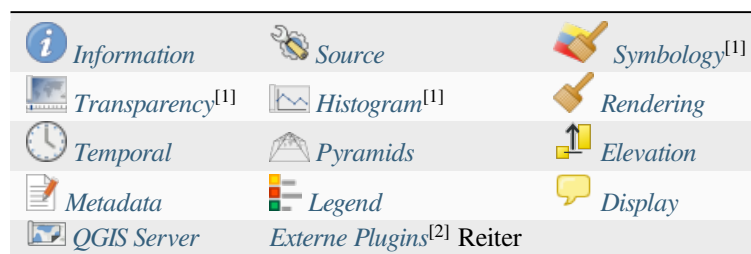
Raster data is made up of pixels (or cells), and each pixel has a value. It is commonly used to store various types of data, including:

- Imagery, such as satellite images, digital aerial photographs, scanned maps
- Elevation data, such as digital elevation models (DEMs), digital terrain models (DTMs)
- Other types of data, such as land cover, soil types, rainfall and many others.

Raster data can be stored in several supported formats, including GeoTIFF, ERDAS Imagine, ArcInfo ASCII GRID, PostGIS Raster and others. See more at [Öffnen von Daten](#).

To view and set the properties for a raster layer, double click on the layer name in the map legend, or right click on the layer name and choose *Properties* from the context menu. This will open the *Raster Layer Properties* dialog.

There are several tabs in the dialog:



<sup>[1]</sup> Auch im Bedienfeld *Layergestaltung* verfügbar

Durch das Installieren von <sup>[2]</sup> *Plugins* können hier im Dialog *Layereigenschaften* weitere Tabs hinzugefügt werden. Diese werden nicht in dieser Dokumentation behandelt, bitte schauen Sie in der entsprechenden Dokumentation nach.

**Tipp:** Live update rendering


The *Layer Styling Panel* provides you with some of the common features of the Layer properties dialog and is a good modeless widget that you can use to speed up the configuration of the layer styles and view your changes on the map canvas.

---

**Bemerkung:** Because properties (symbolology, label, actions, default values, forms...) of embedded layers (see *Embedding layers from external projects*) are pulled from the original project file, and to avoid changes that may break this behavior, the layer properties dialog is made unavailable for these layers.


---


### 17.1.1 Eigenschaften Information

The  *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata for the current layer. Provided information are:

- general such as name in the project, source path, list of auxiliary files, last save time and size, the used provider
- based on the provider of the layer: extent, width and height, data type, GDAL driver, bands statistics
- das Koordinatenbezugssystem: Name, Einheiten, Methode, Genauigkeit, Referenz (d.h., ob es statisch oder dynamisch ist)
- read from layer properties: data type, extent, width/height, compression, pixel size, statistics on bands, number of columns, rows and no-data values of the raster...
- picked from the *filled metadata*: access, extents, links, contacts, history...

### 17.1.2 Eigenschaften Quelle

The  *Source* tab displays basic information about the selected raster, including:

- the *Layer name* to display in the *Layers Panel*;
- the *Coordinate Reference System*: Displays the layer's *Coordinate Reference System (CRS)*. You can change the layer's CRS, by selecting a recently used one in the drop-down list or clicking on the  *Select CRS* button (see *Coordinate Reference System Selector*). Use this process only if the layer CRS is a wrong or not specified. If you wish to reproject your data, use a reprojection algorithm from Processing or *Save it as new dataset*.



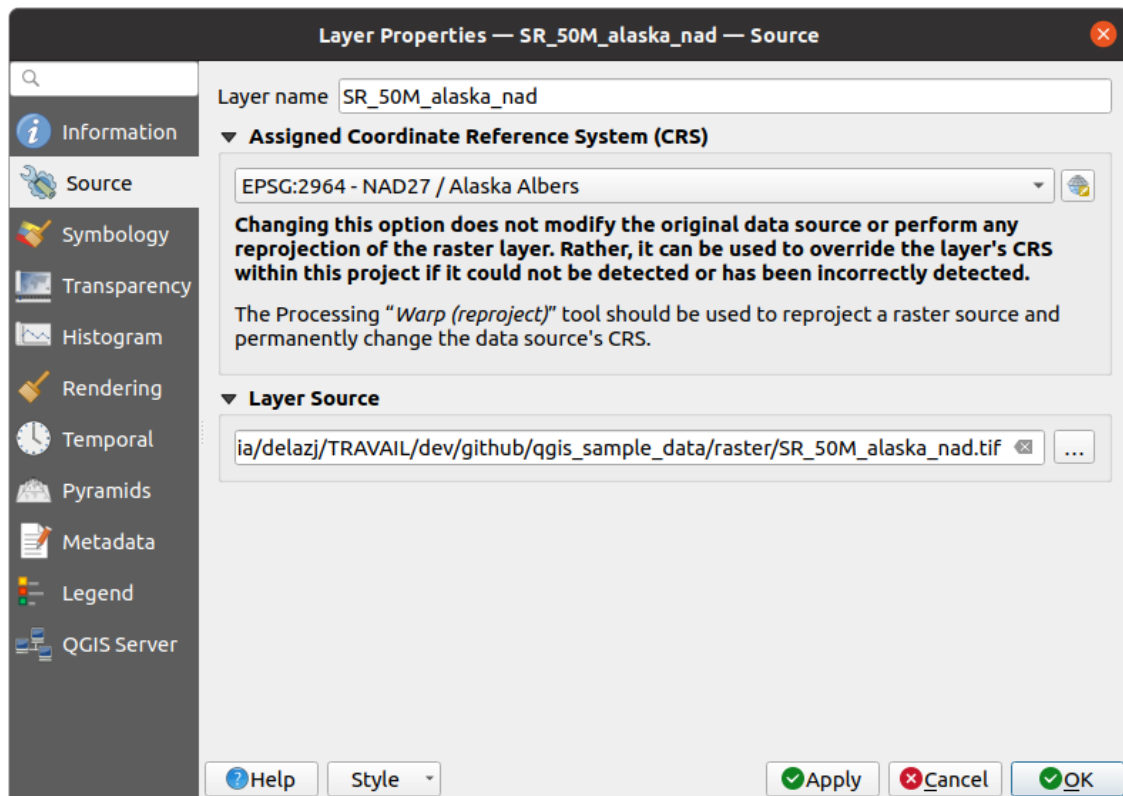


Abb. 17.1: Raster Layer Properties - Source Dialog

### 17.1.3 Eigenschaften Symbolisierung

The raster layer symbology tab is made of three different sections:

- The *Band rendering* where you can control the renderer type to use
- The *Layer rendering* to apply effects on rendered data
- The *Resampling* methods to optimize rendering on map

#### Kanaldarstellung

QGIS offers many different *Render types*. The choice of renderer depends on the data type and the information you'd like to highlight.

1. *Multiband color* - if the file comes with several bands (e.g. a satellite image with several bands).
2. *Paletted/Unique values* - for single band files that come with an indexed palette (e.g. a digital topographic map) or for general use of palettes for rendering raster layers.
3. *Singleband gray* - (one band of) the image will be rendered as gray. QGIS will choose this renderer if the file is neither multiband nor paletted (e.g. a shaded relief map).
4. *Singleband pseudocolor* - this renderer can be used for files with a continuous palette or color map (e.g. an elevation map).
5. *Hillshade* - Creates hillshade from a band.
6. *Contours* - Generates contours on the fly for a source raster band.

### Multiband color

With the multiband color renderer, three selected bands from the image will be used as the red, green or blue component of the color image. QGIS automatically fetches *Min* and *Max* values for each band of the raster and scales the coloring accordingly. You can control the value ranges in the [Min/Max Value Settings](#) section.

A *Contrast enhancement* method can be applied to the values: ‚No enhancement‘, ‚Stretch to MinMax‘, ‚Stretch and clip to MinMax‘ and ‚Clip to min max‘.

#### Bemerkung: Kontrastverbesserung

When adding GRASS rasters, the option *Contrast enhancement* will always be set automatically to *stretch to min max*, even if this is set to another value in the QGIS general options.

The screenshot shows the 'Band Rendering' dialog in QGIS. It is configured for 'Multiband color' rendering. Three bands are selected: Band 1 (Red), Band 2 (Green), and Band 3 (Blue). Each band has its own 'Min' and 'Max' value range set. The 'Contrast enhancement' dropdown is currently set to 'No Enhancement'. At the bottom, there is a button labeled 'Min / Max Value Settings'.

Abb. 17.2: Raster Symbology - Multiband color rendering

#### Tipp: Einen einzelnen Kanal eines Mehrkanal-Rasterlayers anzeigen

If you want to view a single band of a multiband image (for example, Red), you might think you would set the Green and Blue bands to *Not Set*. But the preferred way of doing this is to set the image type to *Singleband gray*, and then select Red as the *Gray band* to use.

### Paletted/Unique values

This is the standard render option for singleband files that include a color table, where a certain color is assigned to each pixel value. In that case, the palette is rendered automatically.

It can be used for all kinds of raster bands, assigning a color to each unique raster value.

If you want to change a color, just double-click on the color and the *Select color* dialog appears.

It is also possible to assign labels to the colors. The label will then appear in the legend of the raster layer.

Right-clicking over selected rows in the color table shows a contextual menu to:

- *Change Color...* for the selection
- *Change Opacity...* for the selection
- *Change Label...* for the selection

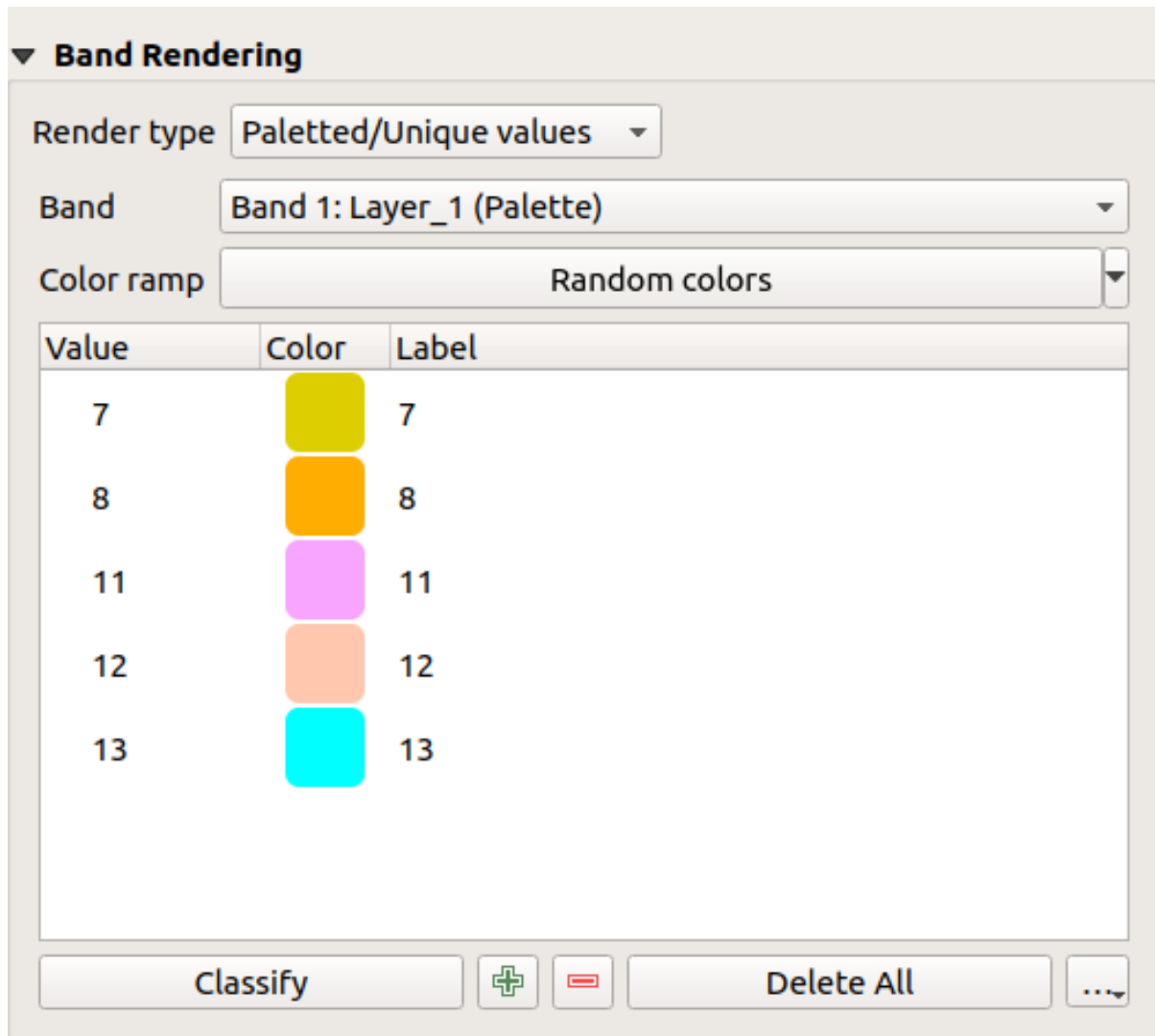


Abb. 17.3: Raster Symbolology - Paletted unique value rendering

The pulldown menu, that opens when clicking the ... (Advanced options) button below the color map to the right, offers color map loading (*Load Color Map from File...*) and exporting (*Export Color Map to File...*), and loading of classes (*Load Classes from Layer*).

### Singleband gray

This renderer allows you to render a layer using only one band with a *Color gradient*: ‚Black to white‘ or ‚White to black‘. You can change the range of values to color (*Min* and *Max*) in the [Min/Max Value Settings](#).

A *Contrast enhancement* method can be applied to the values: ‚No enhancement‘, ‚Stretch to MinMax‘, ‚Stretch and clip to MinMax‘ and ‚Clip to min max‘.

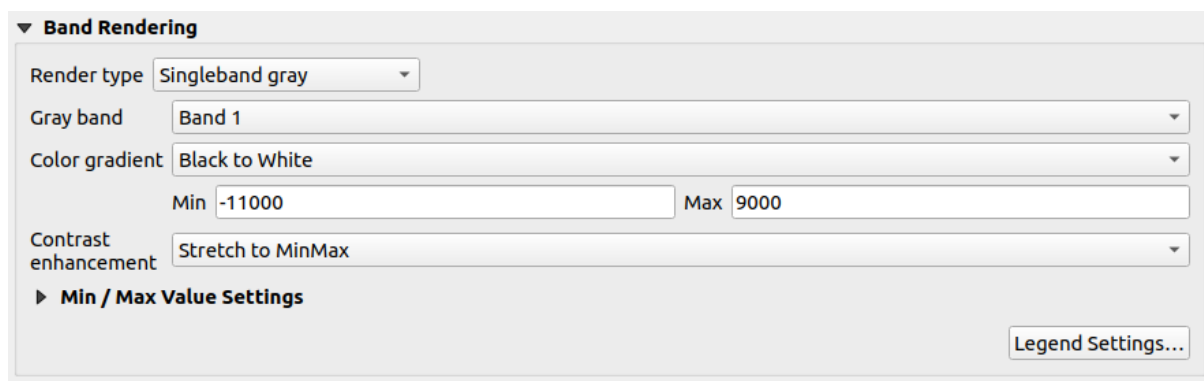


Abb. 17.4: Raster Symbolology - Singleband gray rendering

Pixels are assigned a color based on the selected color gradient and the layer's legend (in the *Layers* panel and the layout *legend item*) is displayed using a continuous color ramp. Press *Legend settings...* if you wish to tweak the settings. More details at [Customize raster legend](#).

## Singleband pseudocolor

This is a render option for single-band files that include a continuous palette. You can also create color maps for a band of a multiband raster.

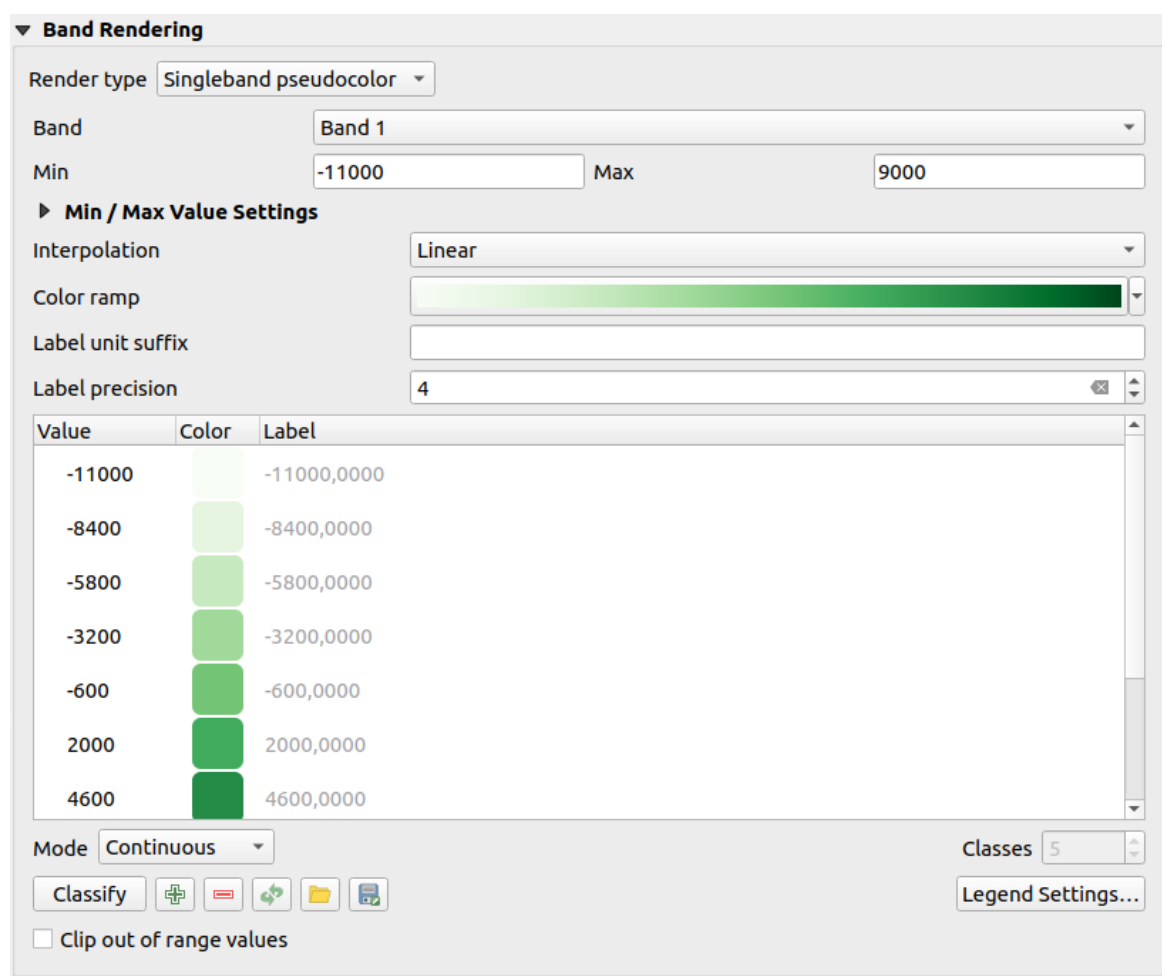


Abb. 17.5: Raster Symbolology - Singleband pseudocolor rendering

Using a *Band* of the layer and a *values range*, you can now interpolate and assign representation color to pixels within classes. More at [Color ramp shader classification](#).

Pixels are assigned a color based on the selected color ramp and the layer's legend (in the *Layers* panel and the layout *legend item*) is displayed using a continuous color ramp. Press *Legend settings...* if you wish to tweak the settings or instead use a legend with separated classes (and colors). More details at [Customize raster legend](#).

## Hillshade

Render a band of the raster layer using hillshading.

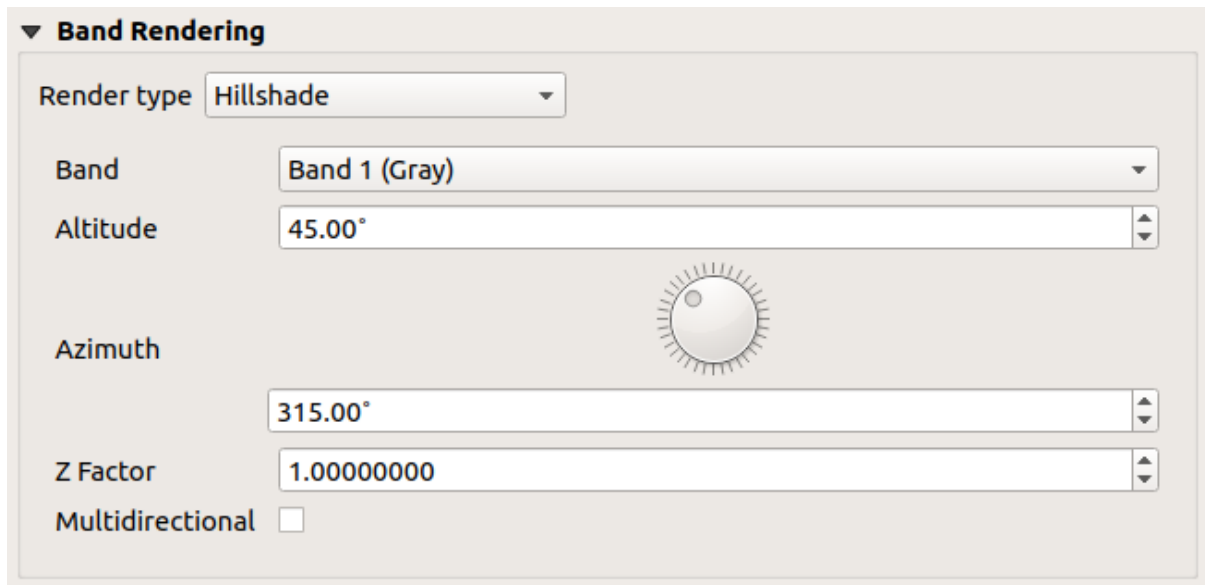


Abb. 17.6: Raster Symbolology - Hillshade rendering

Optionen:

- *Band*: The raster band to use.
- *Altitude*: The elevation angle of the light source (default is 45°).
- *Azimuth*: The azimuth of the light source (default is 315°).
- *Z Factor*: Scaling factor for the values of the raster band (default is 1).
- ☒ *Multidirectional*: Specify if multidirectional hillshading is to be used (default is off).

## Contours

This renderer draws contour lines that are calculated on the fly from the source raster band.

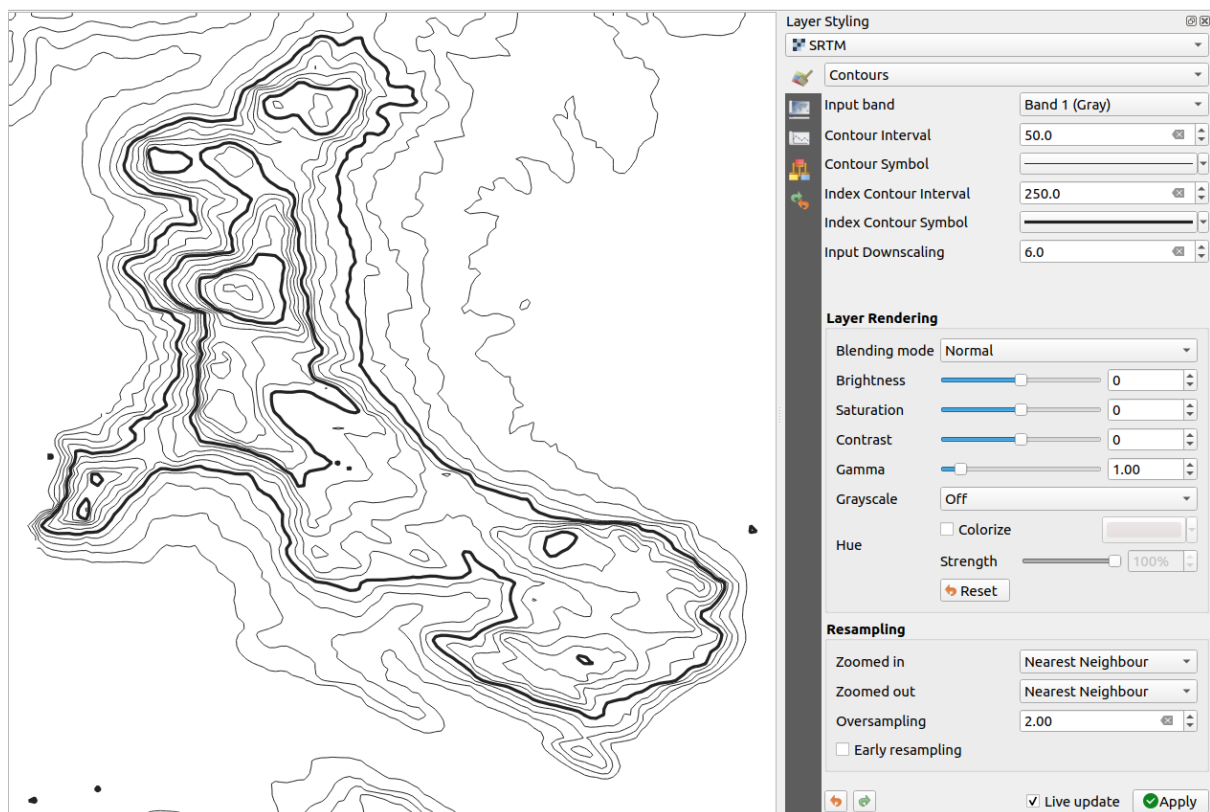


Abb. 17.7: Raster Symbology - Contours rendering

Optionen:

- *Input band*: the raster band to use.
- *Contour interval*: the distance between two consecutive contour lines
- *Contour symbol*: the [symbol](#) to apply to the common contour lines.
- *Index contour interval*: the distance between two consecutive **index contours**, that is the lines shown in a distinctive manner for ease of identification, being commonly printed more heavily than other contour lines and generally labeled with a value along its course.
- *Index contour symbol*: the symbol to apply to the index contour lines
- *Input downscaling*: Indicates by how much the renderer will scale down the request to the data provider (default is 4.0).

For example, if you generate contour lines on input raster block with the same size as the output raster block, the generated lines would contain too much detail. This detail can be reduced by the „downscale“ factor, requesting lower resolution of the source raster. For a raster block 1000x500 with downscale 10, the renderer will request raster 100x50 from provider. Higher downscale makes contour lines more simplified (at the expense of losing some detail).

## Setting the min and max values

By default, QGIS reports the *Min* and *Max* values of the band(s) of the raster. A few very low and/or high values can have a negative impact on the rendering of the raster. The *Min/Max Value Settings* frame helps you control the rendering.

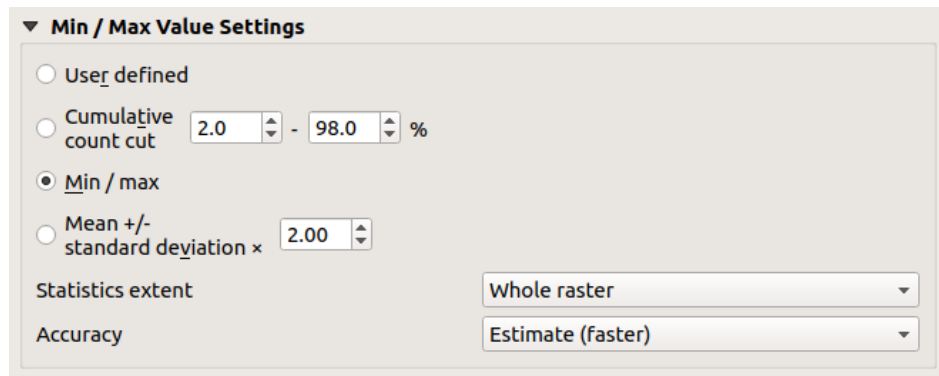


Abb. 17.8: Raster Symbology - Min and Max Value Settings

Available options are:

- *User defined*: The default *Min* and *Max* values of the band(s) can be overridden
- *Cumulative count cut*: Removes outliers. The standard range of values is 2% to 98%, but it can be adapted manually.
- *Min / max*: Uses the whole range of values in the image band.
- *Mean +/- standard deviation x*: Creates a color table that only considers values within the standard deviation or within multiple standard deviations. This is useful when you have one or two cells with abnormally high values in a raster layer that impact the rendering of the raster negatively.

Calculations of the min and max values of the bands are made based on the:

- *Statistics extent*: it can be *Whole raster*, *Current canvas* or *Updated canvas*. *Updated canvas* means that min/max values used for the rendering will change with the canvas extent (dynamic stretching).
- *Accuracy*, which can be either *Estimate (faster)* or *Actual (slower)*.

---

**Bemerkung:** For some settings, you may need to press the *Apply* button of the layer properties dialog in order to display the actual min and max values in the widgets.

---

## Color ramp shader classification

This method can be used to classify and represent scalar dataset (raster or mesh contour) based on their values. Given a *color ramp* and a number of classes, it generates intermediate color map entries for class limits. Each color is mapped with a value interpolated from a range of values and according to a classification mode. The scalar dataset elements are then assigned their color based on their class.

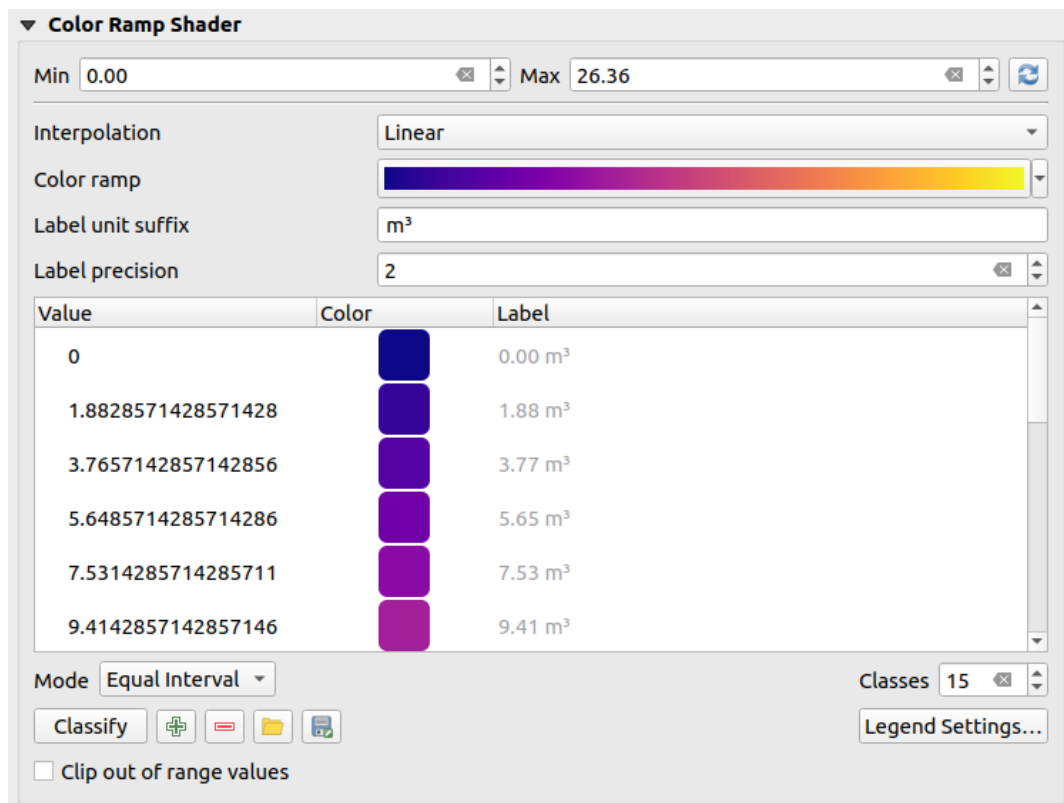






Abb. 17.9: Classifying a dataset with a color ramp shader


1. A *Min* and *Max* values must be defined and used to interpolate classes bounds. By default QGIS detects them from the dataset but they can be modified.
2. The *Interpolation* entry defines how scalar elements are assigned their color :
  - *Discrete* (a  $\leq$  symbol appears in the header of the *Value* column): The color is taken from the closest color map entry with equal or higher value
  - *Linear*: The color is linearly interpolated from the color map entries above and below the pixel value, meaning that to each dataset value corresponds a unique color
  - *Exact* (a  $=$  symbol appears in the header of the *Value* column): Only pixels with value equal to a color map entry are applied a color; others are not rendered.
3. The *Color ramp* widget helps you select the color ramp to assign to the dataset. As usual with [this widget](#), you can create a new one and edit or save the currently selected one. The name of the color ramp will be saved in the configuration.
4. The *Label unit suffix* adds a label after the value in the legend, and the *Label precision* controls the number of decimals to display.
5. The classification *Mode* helps you define how values are distributed across the classes:
  - *Equal interval*: Provided the *Number of classes*, limits values are defined so that the classes all have the same magnitude.
  - *Continuous*: Classes number and color are fetched from the color ramp stops; limits values are set following stops distribution in the color ramp.
  - *Quantile*: Provided the *Number of classes*, limits values are defined so that the classes have the same number of elements. Not available with [mesh layers](#).
6. You can then *Classify* or tweak the classes:
  - The button  Add values manually adds a value to the table.



- The button  Remove selected row deletes selected values from the table.
- Double clicking in the *Value* column lets you modify the class value.
- Double clicking in the *Color* column opens the dialog *Change color*, where you can select a color to apply for that value.
- Double clicking in the *Label* column to modify the label of the class, but this value won't be displayed when you use the identify feature tool.
- Right-clicking over selected rows in the color table shows a contextual menu to *Change Color...* and *Change Opacity...* for the selection.

You can use the buttons  Load color map from file or  Export color map to file to load an existing color table or to save the color table for later use.

7. With linear *Interpolation*, you can also configure:

-  *Clip out of range values*: By default, the linear method assigns the first class (respectively the last class) color to values in the dataset that are lower than the set *Min* (respectively greater than the set *Max*) value. Check this setting if you do not want to render those values.
- *Legend settings*, for display in the *Layers* panel and the layout *legend item*. More details at [Customize raster legend](#).

## Customize raster legend

When applying a color ramp to a raster or a mesh layer, you may want to display a legend showing the classification. By default, QGIS displays a continuous color ramp with min and max values in the *Layers* panel and the layout *legend item*. This can be customized using the *Legend settings* button in the classification widget.

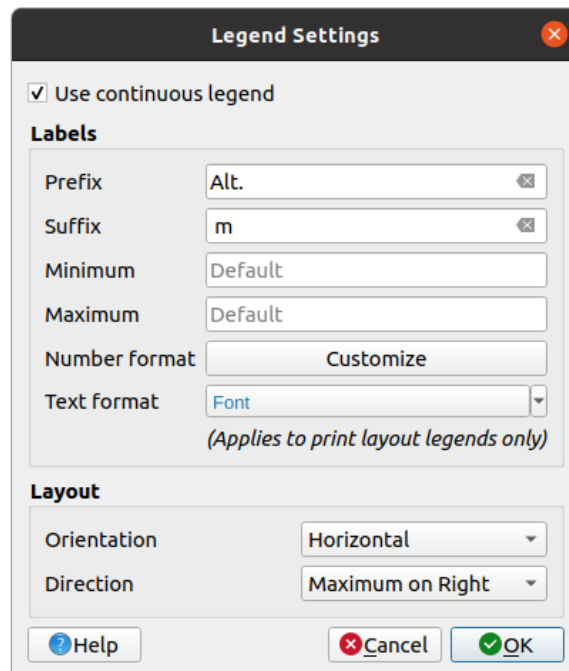



Abb. 17.10: Modifying a raster legend

In this dialog, you can set whether to  *Use continuous legend*: if unchecked, the legend displays separated colors corresponding to the different classes applied. This option is not available for raster *singleband gray* symbology.

Checking the *Use continuous legend* allows you to configure both the labels and layout properties of the legend.

## Labels

- Add a *Prefix* and a *Suffix* to the labels
- Modify the *Minimum* and a *Maximum* values to show in the legend
- *Customize* the *Number format*
- *Customize* the *Text format* to use in the print layout legend.

## Layout

- Control the *Orientation* of the legend color ramp; it can be **Vertical** or **Horizontal**
- Control the *Direction* of the values depending on the orientation:
  - If vertical, you can display the **Maximum on top** or the **Minimum on top**
  - If horizontal, you can display the **Maximum on right** or the **Minimum on right**

## Layerdarstellung

Over the symbology type applied to the layer band(s), you can achieve special rendering effects for the whole raster file(s):

- Use one of the blending modes (see *Mischmodi*)
- Set custom *Brightness*, *Saturation*, *Gamma* and *Contrast* to colors.
- With the ☒ *Invert colors*, the layer is rendered with opposite colors. Handy, for example, to switch out-of-the box OpenStreetMap tiles to dark mode.
- Turn the layer to *Grayscale* option either ‚By lightness‘, ‚By luminosity‘ or ‚By average‘.
- *Colorize* and adjust the *Strength* of *Hue* in the color table

Press *Reset* to remove any custom changes to the layer rendering.

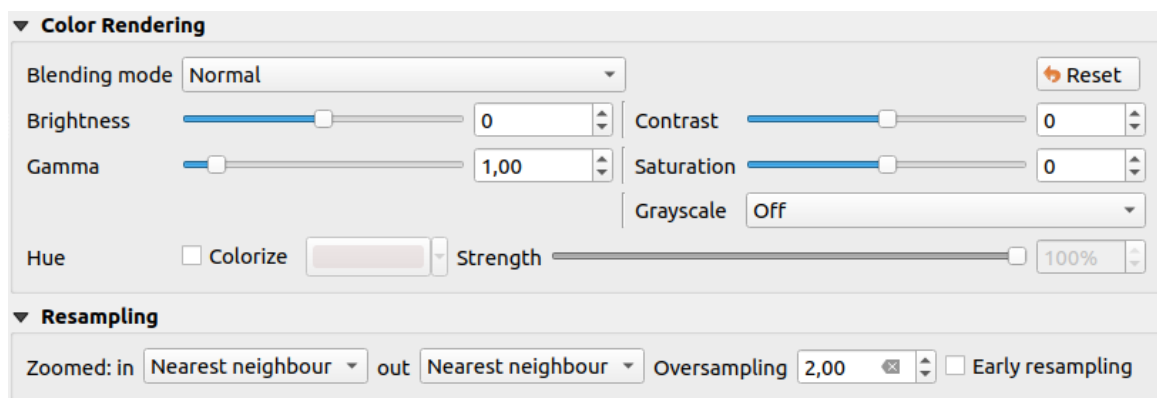



Abb. 17.11: Raster Symbolology - Layer rendering and Resampling settings

## Abtastung

The *Resampling* option has effect when you zoom in and out of an image. Resampling modes can optimize the appearance of the map. They calculate a new gray value matrix through a geometric transformation.

When applying the ‚Nearest neighbour‘ method, the map can get a pixelated structure when zooming in. This appearance can be improved by using the ‚Bilinear (2x2 kernel)‘ or ‚Cubic (4x4 kernel)‘ method, which cause sharp edges to be blurred. The effect is a smoother image. This method can be applied to for instance digital topographic raster maps.

 *Early resampling*: allows to calculate the raster rendering at the provider level where the resolution of the source is known, and ensures a better zoom in rendering with QGIS custom styling. Really convenient for tile rasters loaded using an *interpretation method*.

### 17.1.4 Transparency Properties

QGIS provides capabilities to set the  *Transparency* level of a raster layer.

Use the *Global opacity* slider to set to what extent the underlying layers (if any) should be visible through the current raster layer. This is very useful if you overlay raster layers (e.g., a shaded relief map overlayed by a classified raster map). This will make the look of the map more three dimensional. The opacity of the raster can be data-defined, and vary e.g. depending on the visibility of another layer, by temporal variables, on different pages of an atlas, ...

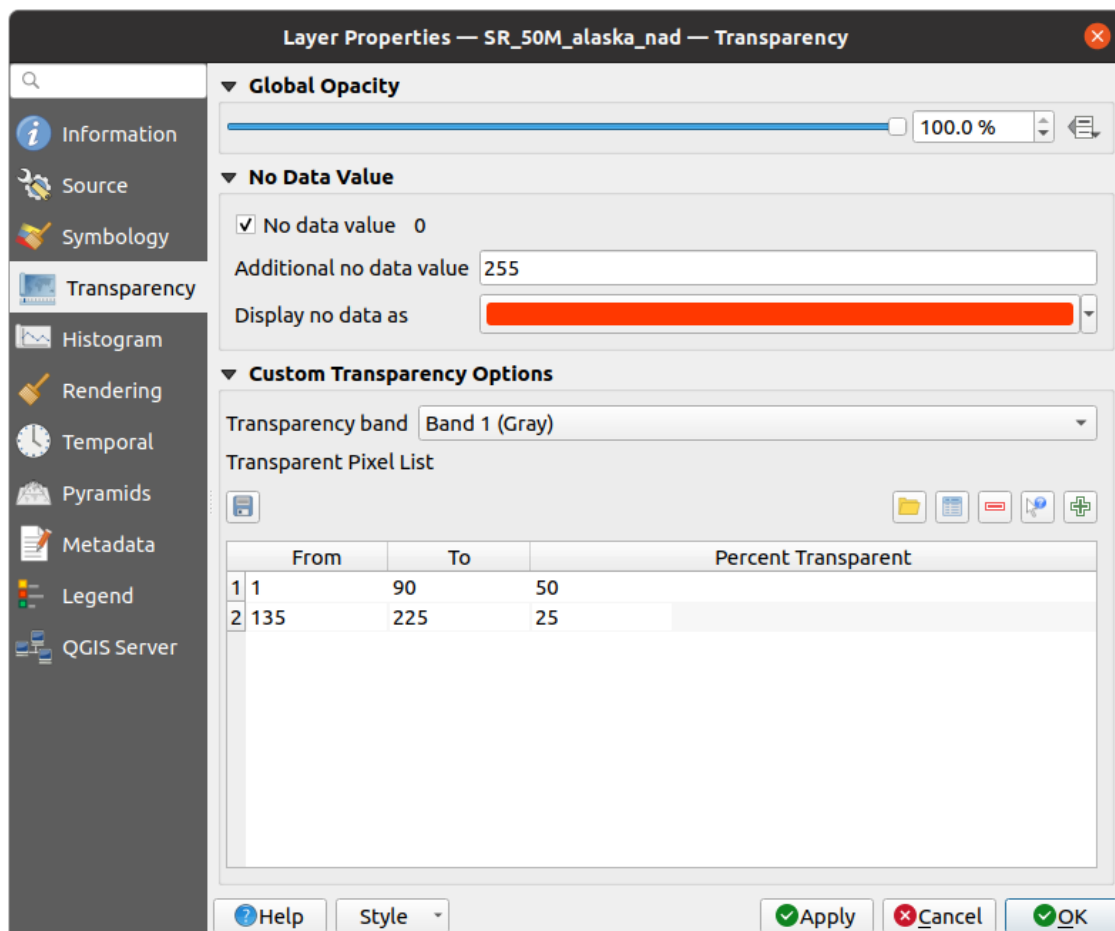







Abb. 17.12: Raster Transparency

With  *No data value* QGIS reports the original source no data value (if defined) which you can consider as is in the rendering. Additionally, you can enter a raster value that should be treated as an *Additional no data value*. The



*Display no data as color selector* allows you to apply a custom color to no data pixels, instead of the default transparent rendering.


An even more flexible way to customize the transparency is available in the *Custom transparency options* section:


- Use *Transparency band* to apply transparency for an entire band.
- Provide a list of pixels to make transparent with corresponding levels of transparency:
  1. Klicken Sie den  *Werte manuell hinzufügen* Knopf. Eine neue Zeile erscheint in der Pixelliste.
  2. Enter the **Red**, **Green** and **Blue** values of the pixel and adjust the **Percent Transparent** to apply.
  3. Alternatively, you can fetch the pixel values directly from the raster using the  *Add values from display* button. Then enter the transparency value.
  4. Repeat the steps to adjust more values with custom transparency.
  5. Press the *Apply* button and have a look at the map.

Wie Sie sehen können ist es recht einfach die benutzerdefinierte Transparenz einzustellen, aber es kann ganz schön viel Arbeit sein. Deswegen können Sie den Knopf  *In Datei exportieren* benutzen um Ihre Transparenzliste in eine Datei zu speichern. Der Knopf  *Aus Datei importieren* lädt Ihre Transparenzeinstellungen und wendet sie auf den aktuellen Rasterlayer an.

### 17.1.5 Histogram Properties

The  *Histogram* tab allows you to view the distribution of the values in your raster. The histogram is generated when you press the *Compute Histogram* button. All existing bands will be displayed together. You can save the histogram as an image with the  button.

At the bottom of the histogram, you can select a raster band in the drop-down menu and *Set min/max style for it*. The  *Prefs/Actions* drop-down menu gives you advanced options to customize the histogram:

- With the *Visibility* option, you can display histograms for individual bands. You will need to select the option  *Show selected band*.
- The *Min/max options* allow you to ,Always show min/max markers‘, to ,Zoom to min/max‘ and to ,Update style to min/max‘.
- The *Actions* option allows you to ,Reset‘ or ,Recompute histogram‘ after you have changed the min or max values of the band(s).

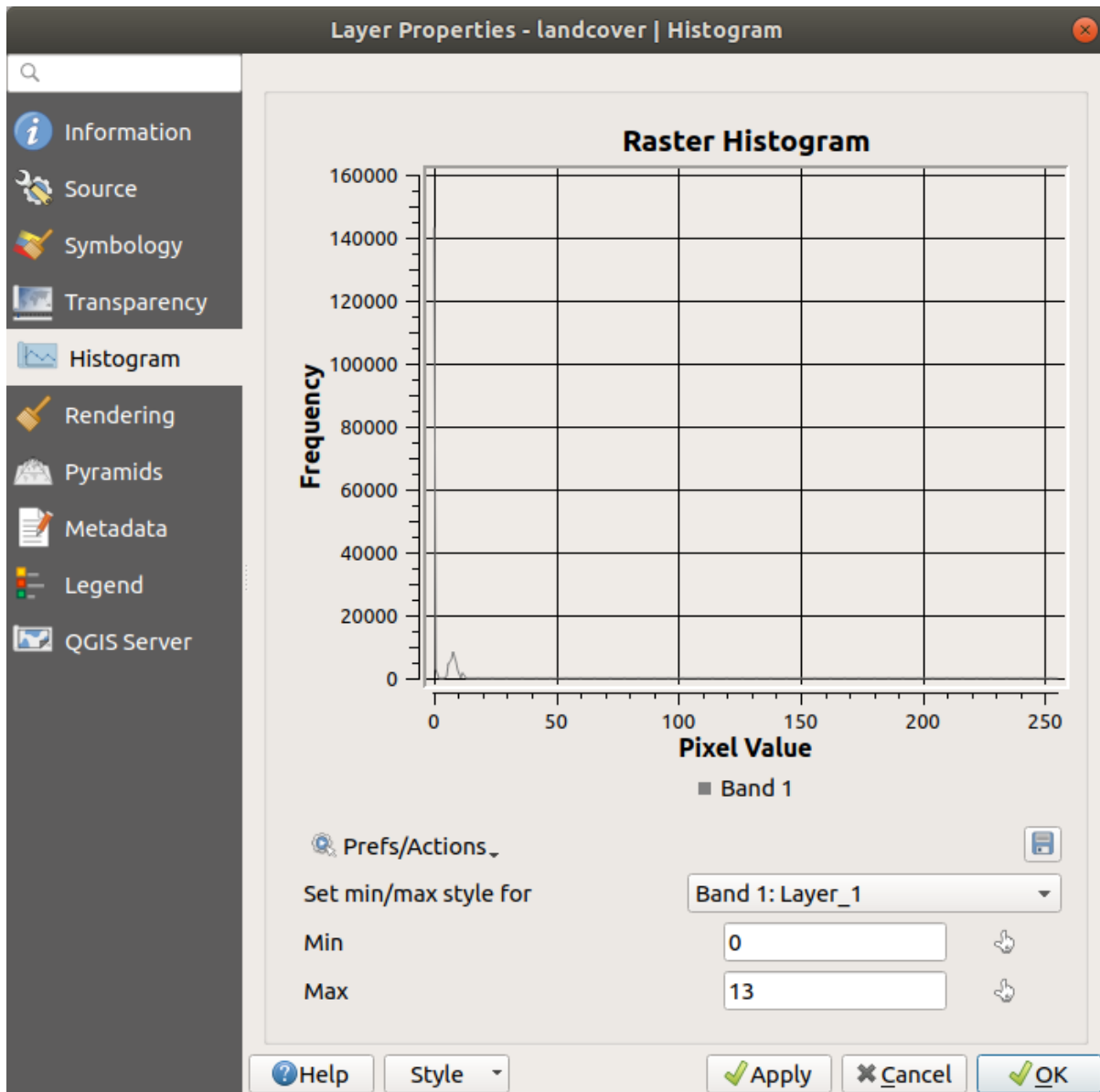




Abb. 17.13: Rasterhistogramm

### 17.1.6 Eigenschaften Darstellung


In the  *Rendering* tab, it's possible to:

- set *Scale dependent visibility* for the layer: You can set the *Maximum (inclusive)* and *Minimum (exclusive)* scales, defining a range of scales in which the layer will be visible. It will be hidden outside this range. The  Set to current canvas scale button helps you use the current map canvas scale as a boundary. See [Visibility Scale Selector](#) for more information.

---

**Bemerkung:** You can also activate scale dependent visibility on a layer from within the *Layers* panel: right-click on the layer and in the contextual menu, select *Set Layer Scale Visibility*.

---

-  *Refresh layer at interval*: controls whether and how regular a layer can be refreshed. Available *Configuration* options are:

- *Reload data*: the layer will be completely refreshed. Any cached data will be discarded and refetched from the provider. This mode may result in slower map refreshes.
- *Redraw layer only*: this mode is useful for animation or when the layer's style will be updated at regular intervals. Canvas updates are deferred in order to avoid refreshing multiple times if more than one layer has an auto update interval set.

It is also possible to set the *Interval (seconds)* between consecutive refreshments.

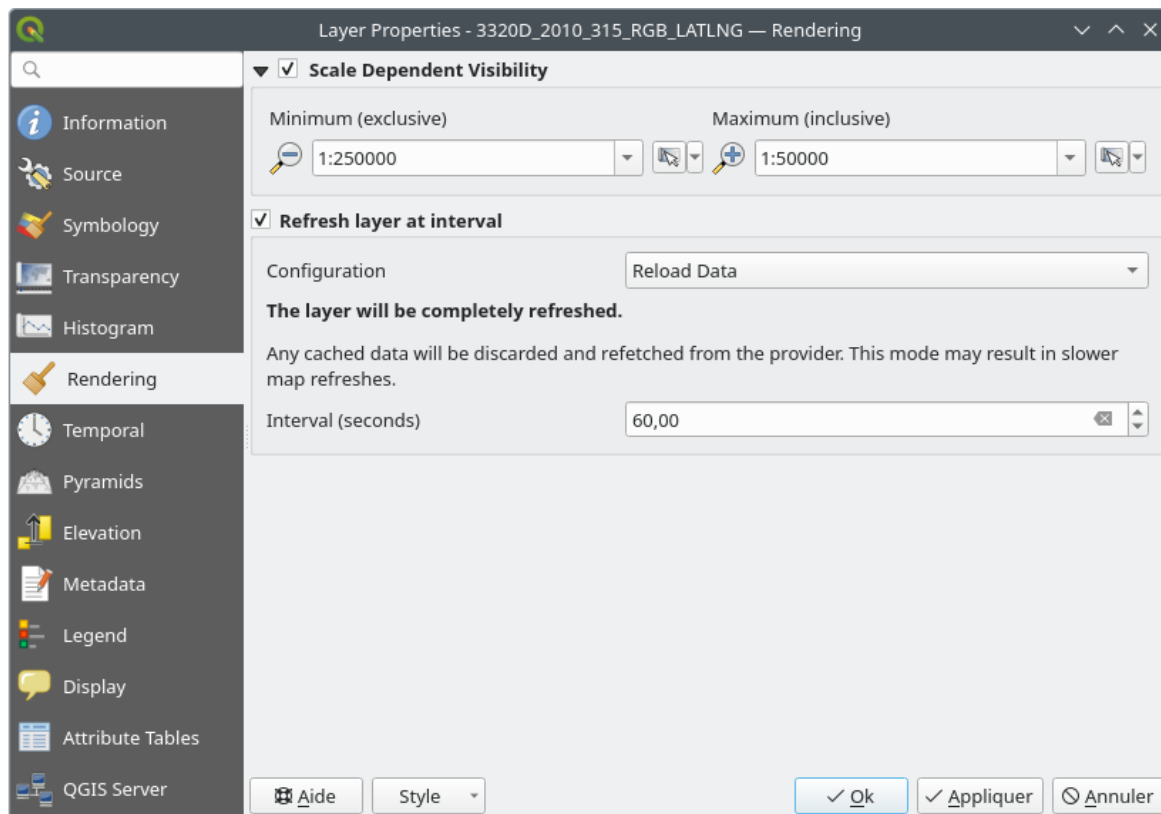



Abb. 17.14: Raster Rendering Properties

### 17.1.7 Eigenschaften Zeitlich

The  *Temporal* tab provides options to control the rendering of the layer over time. Such dynamic rendering requires the *temporal navigation* to be enabled over the map canvas.

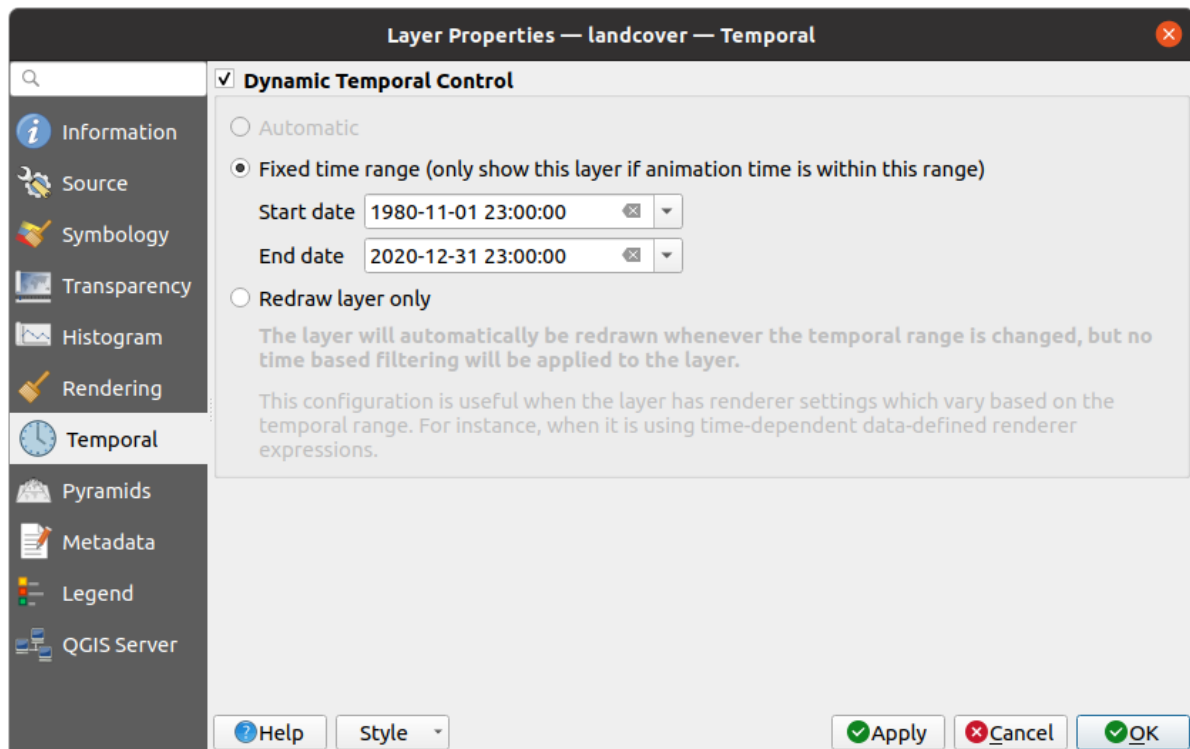



Abb. 17.15: Raster Temporal Properties

Check the  *Dynamic Temporal Control* option and set whether the layer redraw should be:

- *Automatic*: the rendering is controlled by the underlying data provider if it supports temporal data handling. E.g. this can be used with WMS-T layers or PostGIS rasters.
- *Fixed time range*: only show the raster layer if the animation time is within a *Start date* and *End date* range
- *Redraw layer only*: the layer is redrawn at each new animation frame. It's useful when the layer uses time-based expression values for renderer settings (e.g. data-defined renderer opacity, to fade in/out a raster layer).

### 17.1.8 Pyramids Properties

High resolution raster layers can slow navigation in QGIS. By creating lower resolution copies of the data (pyramids), performance can be considerably improved, as QGIS selects the most suitable resolution to use depending on the zoom level.

Sie brauchen dazu Schreibrecht in dem Ordner, in dem sich sie Originaldaten befinden.

From the *Resolutions* list, select resolutions at which you want to create pyramid levels by clicking on them.

If you choose **Internal (if possible)** from the *Overview format* drop-down menu, QGIS tries to build pyramids internally.

---

**Bemerkung:** Please note that building pyramids may alter the original data file, and once created they cannot be removed. If you wish to preserve a 'non-pyramided' version of your raster, make a backup copy prior to pyramid building.

---

If you choose **External** and **External (Erdas Imagine)** the pyramids will be created in a file next to the original raster with the same name and a `.ovr` extension.

Several *Resampling methods* can be used for pyramid calculation:

- Nächster Nachbar

- Durchschnitt
- Gauß
- Kubisch
- Cubic Spline
- Laczos
- Modus
- Keine

Finally, click *Build Pyramids* to start the process.

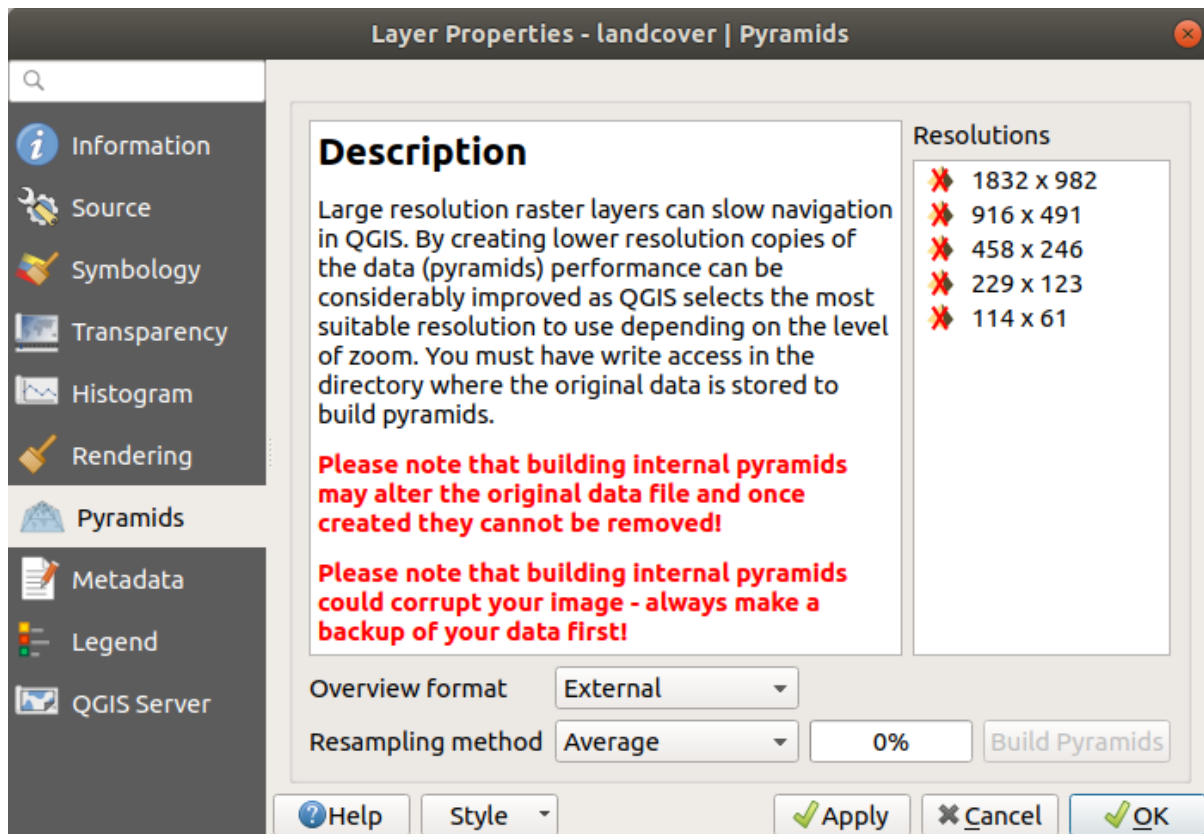



Abb. 17.16: Raster Pyramids



## 17.1.9 Elevation Properties

The  *Elevation* tab provides options to control the layer elevation properties within a *3D map view* and its appearance in the *profile tool charts*. Specifically, you can set:

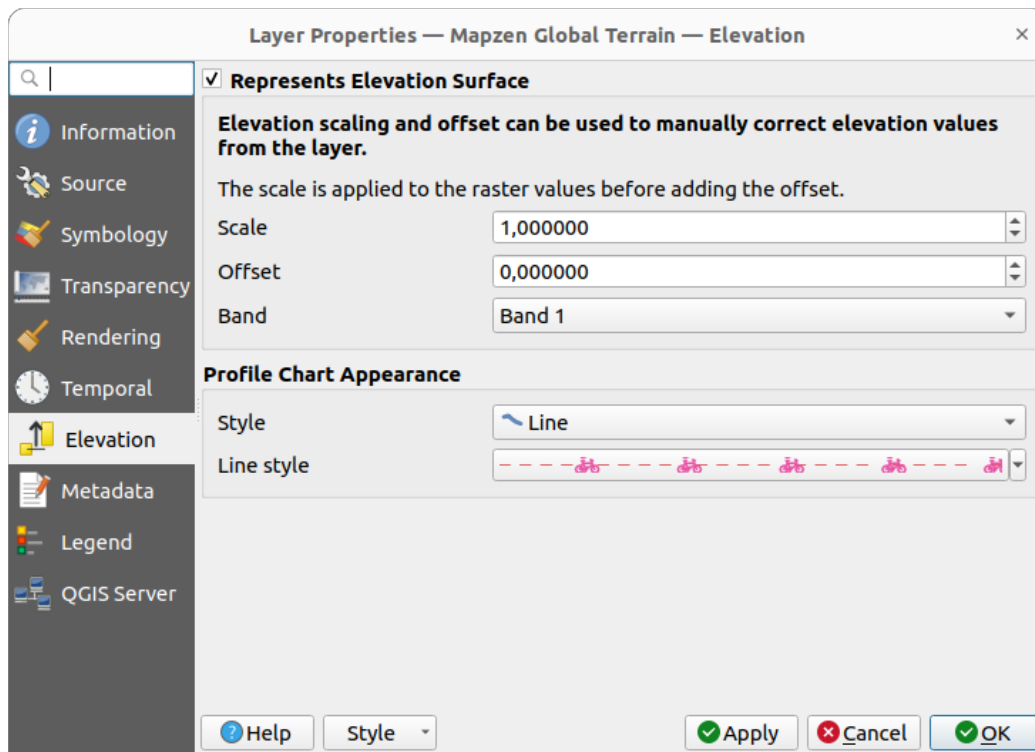




Abb. 17.17: Raster Elevation Properties

-  *Represents Elevation Surface*: whether the raster layer represents a height surface (e.g. DEM) and the pixel values should be interpreted as elevations. Check this option if you want to display a raster in an *elevation profile view*. You will also need to fill in the *Band* to pick values from and can apply a *Scale* factor and an *Offset*.
- *Profile Chart Appearance*: controls the rendering of the raster elevation data in the profile chart. The profile *Style* can be set as:
  - a *Line* with a specific *Line style*
  - an elevation surface rendered using a fill symbol either above (*Fill above*) or below (*Fill below*) the elevation curve line. The surface symbology is represented using:
    - \* a *Fill style*
    - \* and a *Limit*: the maximum (respectively minimum) altitude determining how high the fill surface will be

### 17.1.10 Eigenschaften Metadaten

Der Tab  *Metadaten* ermöglicht es Ihnen, Metadaten für Ihren Layer zu erstellen oder zu editieren. Siehe [Metadaten](#) für weitere Informationen.

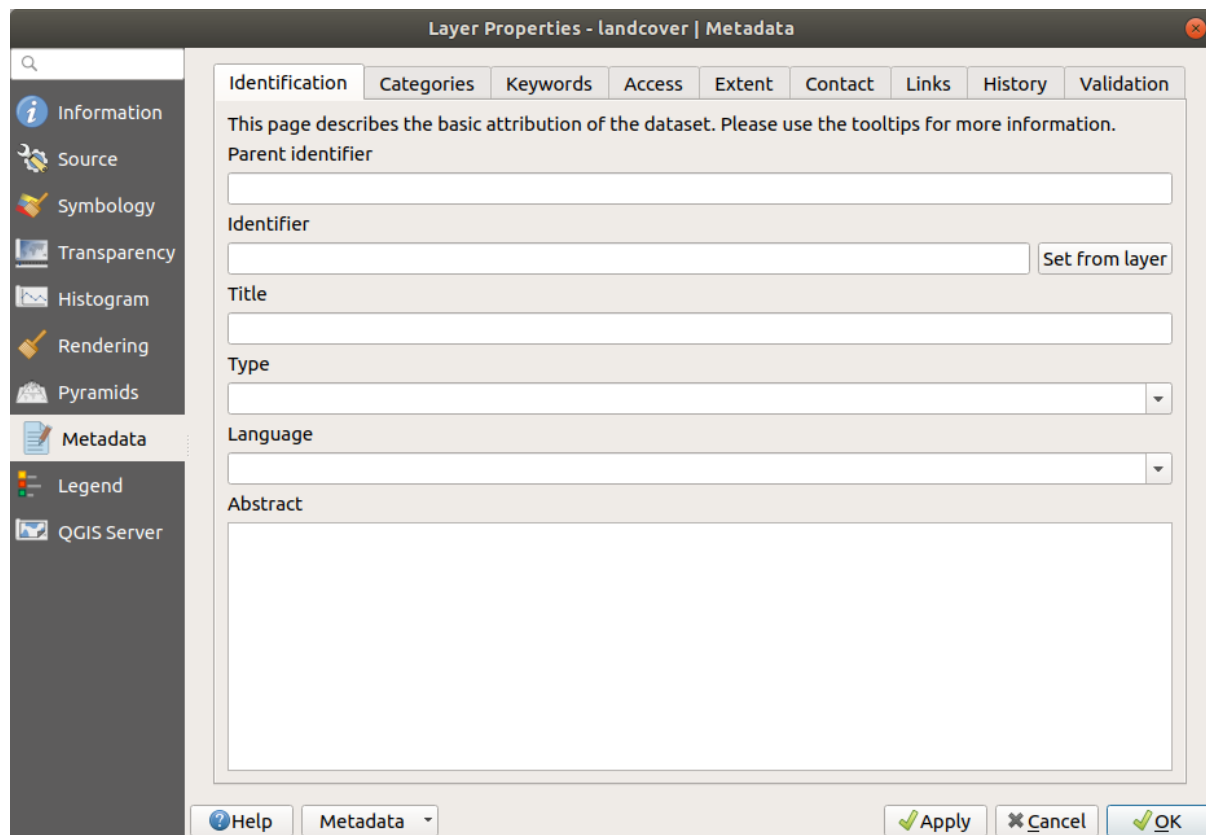



Abb. 17.18: Raster Metadaten

### 17.1.11 Eigenschaften Legende

Der Tab  *Legende* provides you with advanced settings for the *Layers panel* and/or the *print layout legend*. These options include:

- Depending on the symbology applied to the layer, you may end up with several entries in the legend, not necessarily readable/useful to display. The *Legend placeholder image* helps you *select an image* for replacement, displayed both in the *Layers panel* and the *print layout legend*.
- The *Embedded widgets in Legend* provides you with a list of widgets you can embed within the layer tree in the *Layers panel*. The idea is to have a way to quickly access some actions that are often used with the layer (setup transparency, filtering, selection, style or other stuff...).

By default, QGIS provides a transparency widget but this can be extended by plugins that register their own widgets and assign custom actions to layers they manage.

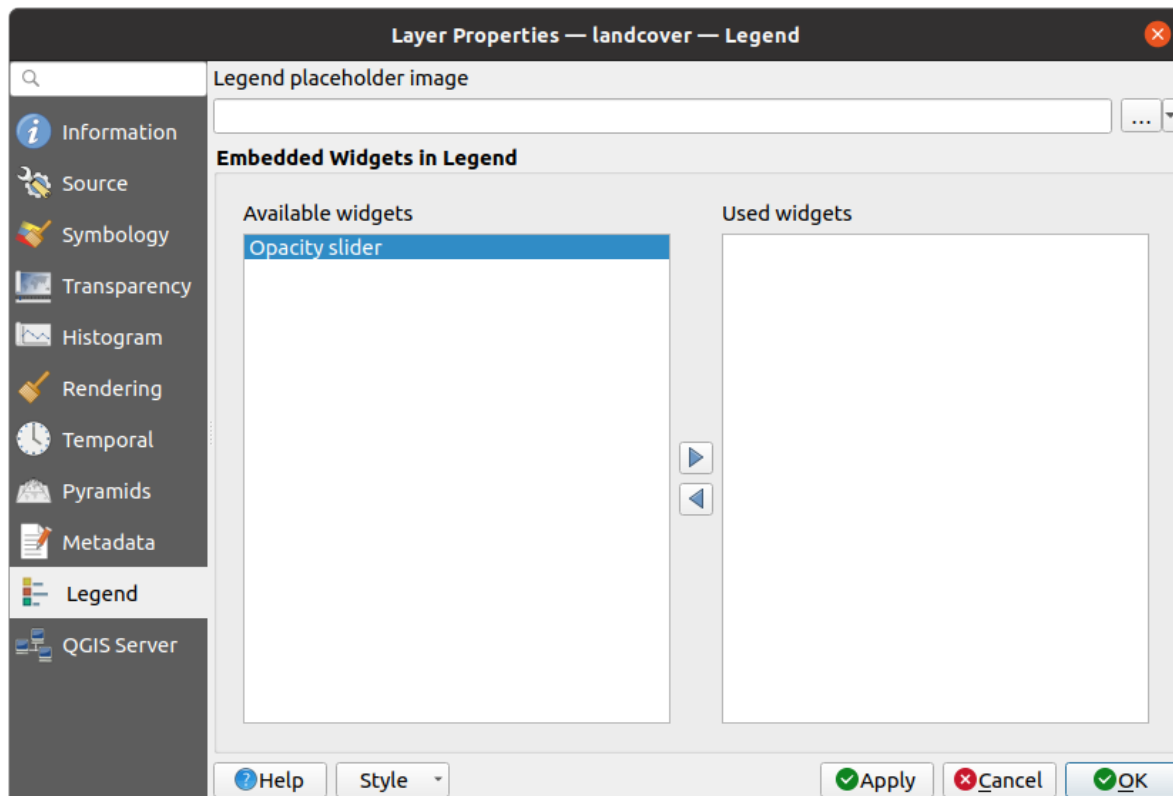



Abb. 17.19: Raster Legend

### 17.1.12 Display Properties

The  *Display* tab helps you configure HTML map tips to use for pixels identification:

-  *Enable Map Tips* controls whether to display map tips for the layer
- The *HTML Map Tip* provides a complex and full HTML text editor for map tips, mixing QGIS expressions and html styles and tags (multiline, fonts, images, hyperlink, tables, ...). You can check the result of your code sample in the *Preview* frame.

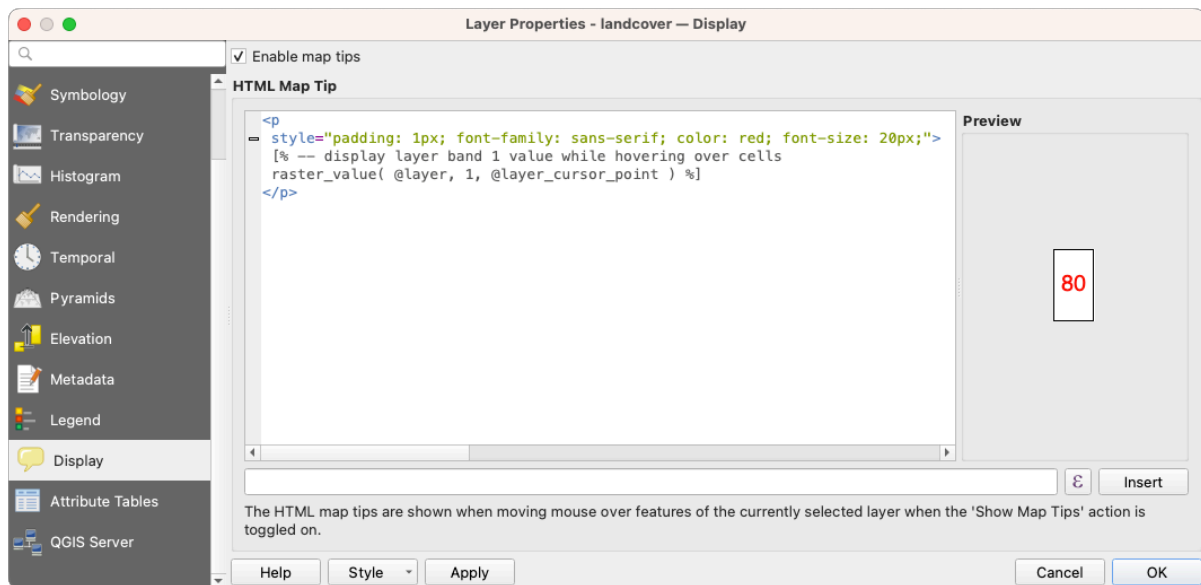





Abb. 17.20: Map tips with raster layer

To display map tips:

1. Select the menu option *View ► Show Map Tips* or click on the  Show Map Tips icon of the *Attributes Toolbar*.
2. Make sure that the layer you target is active and has the  *Enable Map Tips* property checked.
3. Move over a pixel, and the corresponding information will be displayed over.

Map tip is a cross-layer feature meaning that once activated, it stays on and applies to any map tip enabled layer in the project until it is toggled off.

### 17.1.13 Eigenschaften QGIS Server

The  *QGIS Server* tab helps you configure settings of the data when published by QGIS Server. The configuration concerns:

- *Description*: provides information to describe the data, such as *Short name*, *Title*, *Summary*, a *List of Keywords*, and a *Data URL* whose *Type* can be in *text/html*, *text/plain* or *application/pdf*.
- *Attribution*: a *Title* and *URL* to identify who provides the data
- *Metadata URL*: a list of *URL* for the metadata that can be of *FGDC* or *TC211 Type*, and in *text/plain* or *text/xml Format*
- *Legend URL*: a *URL* for the legend, in either *image/png* or *image/jpeg Format*

---

**Bemerkung:** When the raster layer you want to publish is already provided by a web service, further *properties* are available for setting.

---

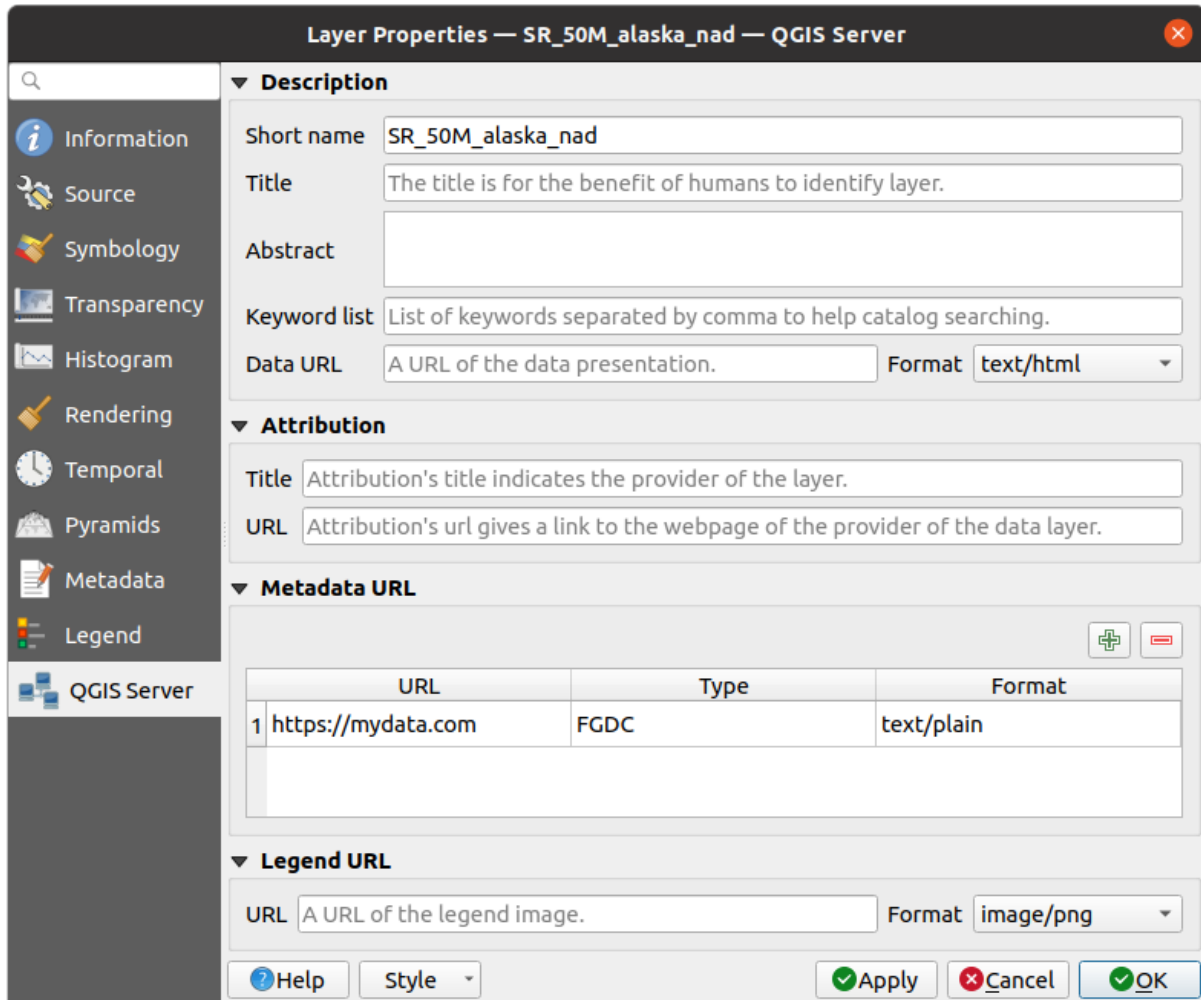



Abb. 17.21: QGIS Server in Raster Properties

### 17.1.14 Identify raster cells

The  *identify features* tool allows you to get information about specific points in a raster layer.

To use the  *Identify features* tool:

1. Select the raster layer in the Layers panel.
2. Click on the *Identify features* tool in the toolbar or press `Ctrl+Shift+I`.
3. Click on the point in the raster layer that you want to identify.

The Identify Results panel will open in its default *Tree* view and display information about the clicked point. Below the name of the raster layer, you have on the left the band(s) of the clicked pixel, and on the right their respective value. These values can also be rendered (from the *View* menu located at the bottom of the panel) in:

- a *Table* view - organizes the information about the identified features and their values in a table.
- a *Graph* view - organizes the information about the identified features and their values in a graph.

Under the pixel attributes, you will find the *Derived* information, such as:

- X and Y coordinate values of the point clicked
- Column and row of the point clicked (pixel)

## 17.2 Rasteranalyse

### 17.2.1 Rasterrechner

The *Raster Calculator* in the *Raster* menu allows you to perform calculations on the basis of existing raster pixel values (see Abb. 17.22). The results are written to a new raster layer in a GDAL-supported format.

**Raster Calculator**

**Raster Bands**

SR\_50M\_alaska\_nad@1  
landcover@1

**Result Layer**

☐ Create on-the-fly raster instead of writing layer to disk

Output layer:

Output format:

**Spatial Extent**

X min:  X max:

Y min:  Y max:

**Resolution**

Columns:  Rows:

Output CRS:

☒ Add result to project

**▼ Operators**

+	*	(	min	IF	cos	acos
-	/	)	max	AND	sin	asin
<	>	=	abs	OR	tan	atan
<=	>=	!=	^	sqrt	log10	ln



**Raster Calculator Expression**

Expression valid

Abb. 17.22: Rasterrechner

The *Raster bands* list contains all loaded raster layers that can be used. To add a raster to the raster calculator expression field, double click its name in the Fields list. You can then use the operators to construct calculation expressions, or you can just type them into the box.

In the *Result layer* section, you will need to define an output layer. You can:

-  *Create on-the-fly raster instead of writing layer to disk:*
  - If unchecked, the output is stored on the disk as a new plain file. An *Output layer* path and an *Output format* are required.
  - If checked, a virtual raster layer, i.e. a raster layer defined by its URI and whose pixels are calculated on-the-fly, is created. It's not a new file on disk; the virtual layer is still connected to the rasters used in the calculation meaning that deleting or moving these rasters would break it. A *Layer name* can be provided, otherwise the calculation expression is used as such. Removing the virtual layer from the project deletes it, and it can be made persistent in file using the layer *Export ► Save as...* contextual menu.
- Define the *Spatial extent* of the calculation based on an input raster layer extent, or on custom X,Y coordinates
- Set the *Resolution* of the layer using columns and rows number. If the input layer has a different resolution, the values will be resampled with the nearest neighbor algorithm.
- With the  *Add result to project* checkbox, the result layer will automatically be added to the legend area and can be visualized. Checked by default for virtual rasters.

The *Operators* section contains all available operators. To add an operator to the raster calculator expression box, click the appropriate button. Mathematical calculations (+, -, \*, ...) and trigonometric functions (sin, cos, tan, ...) are available. Conditional expressions (=, !=, <, >=, ...) return either 0 for false or 1 for true, and therefore can be used with other operators and functions.

**Siehe auch:**

*Raster calculator* and *Raster calculator (virtual)* algorithms

## Raster calculator expression

### The dialog

The *Raster calculator expression* dialog provides means to write expressions for pixels calculations between a set of raster layers.

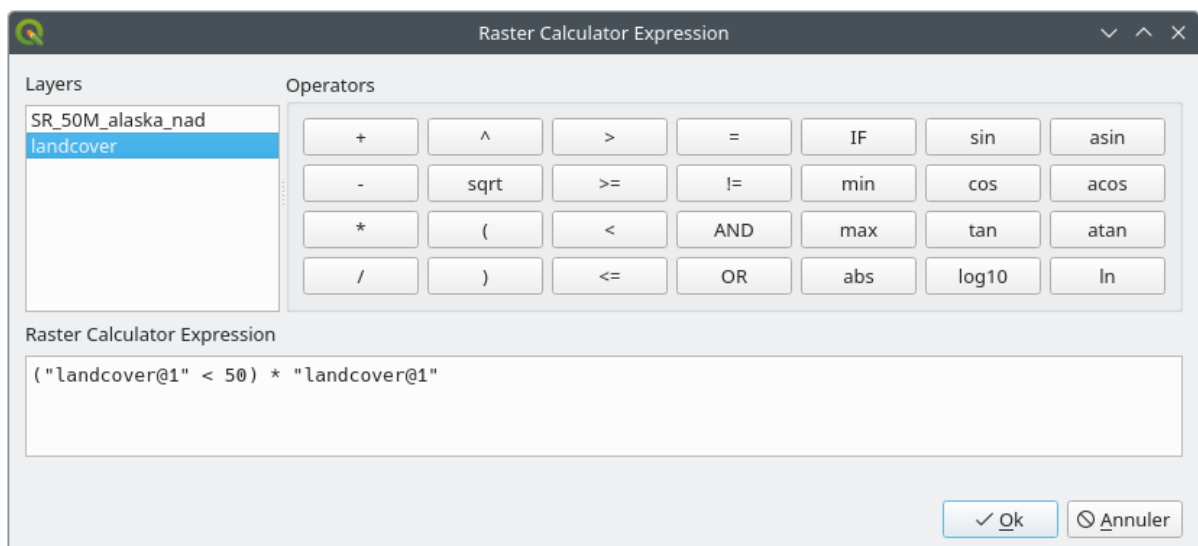


Abb. 17.23: Raster Expression Calculator

- *Layers*: Shows the list of all raster layers loaded in the legend. These can be used to fill the expression box (double click to add). Raster layers are referred by their name and the number of the band:

`layer_name@band_number`. For instance, the first band from a layer named DEM will be referred as `DEM@1`.

- **Operators:** contains a number of calculation operators for pixels manipulation:

- Arithmetic: `+`, `-`, `*`, `sqrt`, `abs`, `ln`, ...
- Trigonometric: `sin`, `cos`, `tan`, ...
- Comparison: `=`, `!=`, `<`, `>=`, ...
- Logical: `IF`, `AND`, `OR`, `(, )`
- Statistical: `min`, `max`

To add an operator to the raster calculator expression box, click the appropriate button.

- *Raster calculator expression* is the area in which the expression is composed

### Beispiele

#### Höhenwerte von Meter zu Fuß konvertieren

Für das Erstellen eines Rasterlayers in Fuß aus einem Rasterlayer in Metern müssen Sie den Konvertierungsfaktor von Metern zu Fuß benutzen: 3.28. Der Ausdruck lautet:

```
"elevation@1" * 3.28
```

#### Eine Maske verwenden

Wenn Sie Teile des Rasterlayers ausmaskieren wollen, weil Sie nur an Höhenwerten über 0 Metern interessiert sind, können Sie den folgenden Ausdruck zum Erstellen einer Maske und zum gleichzeitigen Anwenden auf den Rasterlayer verwenden.

```
("elevation@1" >= 0) * "elevation@1"
```

Mit anderen Worten, für jede Zelle größer oder gleich 0 wird der bedingte Ausdruck mit 1 ausgewertet, was den ursprünglichen Wert durch Multiplikation mit 1 beibehält, ansonsten wird der bedingte Ausdruck zu 0 ausgewertet, was den Rasterwert auf 0 setzt und die Maske im laufenden Betrieb erzeugt.

#### Classify a Raster

Wenn Sie ein Raster klassifizieren wollen, sagen wir beispielsweise in zwei Höhenklassen, können Sie die folgenden Ausdrücke verwenden um ein Raster mit zwei Werten 1 und 2 in einem Schritt erstellen.

```
("elevation@1" < 50) * 1 + ("elevation@1" >= 50) * 2
```


In anderen Worten stelle den Wert für jede Zelle mit einem Wert kleiner 50 auf 1 ein. Für jede Zelle mit einem Wert größer als oder gleich 50 stelle den Wert 2 ein.

Or you can use the IF operator.

```
if ( elevation@1 < 50 , 1 , 2 )
```



## 17.3 Georeferencer

The  Georeferencer is a tool for generating world files for layers. It allows you to reference rasters or vectors to geographic or projected coordinate systems by creating a new GeoTiff or by adding a world file to the existing image. The basic approach to georeferencing a layer is to locate points on it for which you can accurately determine coordinates.

Funktionen





















Icon	Funktion	Icon	Funktion
	Raster öffnen		Open vector
	Georeferenzierung durchführen		
	GDAL-Skript erzeugen		Passpunkte laden
	Passpunkte speichern		Transformationseinstellungen
	Passpunkt hinzufügen		Passpunkt löschen
	Passpunkt verschieben		Verschieben
	Hinein zoomen		Heraus zoomen
	Auf den Layer zoomen		Zoom zurück
	Zoom vor		Georeferenzierung mit QGIS verbinden
	QGIS mit Georeferenzierung verbinden		Volle Histogrammstreckung
	Lokale Histogrammstreckung		

Table Georeferenzierung 1: Georeferenzierfunktionen

### 17.3.1 Wie benutzt man den Georeferenzierer

As X and Y coordinates (DMS (dd mm ss.ss), DD (dd.dd) or projected coordinates (mmmm.mm)), which correspond with the selected point on the image, two alternative procedures can be used:

- The raster itself sometimes provides crosses with coordinates „written“ on the image. In this case, you can enter the coordinates manually.
- Using already georeferenced layers. This can be either vector or raster data that contain the same objects/features that you have on the image that you want to georeference and with the projection that you want for your image. In this case, you can enter the coordinates by clicking on the reference dataset loaded in the QGIS map canvas.

The usual procedure for georeferencing an image involves selecting multiple points on the raster, specifying their coordinates, and choosing a relevant transformation type. Based on the input parameters and data, the Georeferencer will compute the world file parameters. The more coordinates you provide, the better the result will be.

The first step is to start QGIS and click on *Layer* ►  *Georeferencer*, which appears in the QGIS menu bar. The Georeferencer dialog appears as shown in [Abb. 17.24](#).

For this example, we are using a topo sheet of South Dakota from SDGS. It can later be visualized together with the data from the GRASS `spearfish60` location. You can download the topo sheet here: [https://grass.osgeo.org/sampled/spearfish\\_toposheet.tar.gz](https://grass.osgeo.org/sampled/spearfish_toposheet.tar.gz).

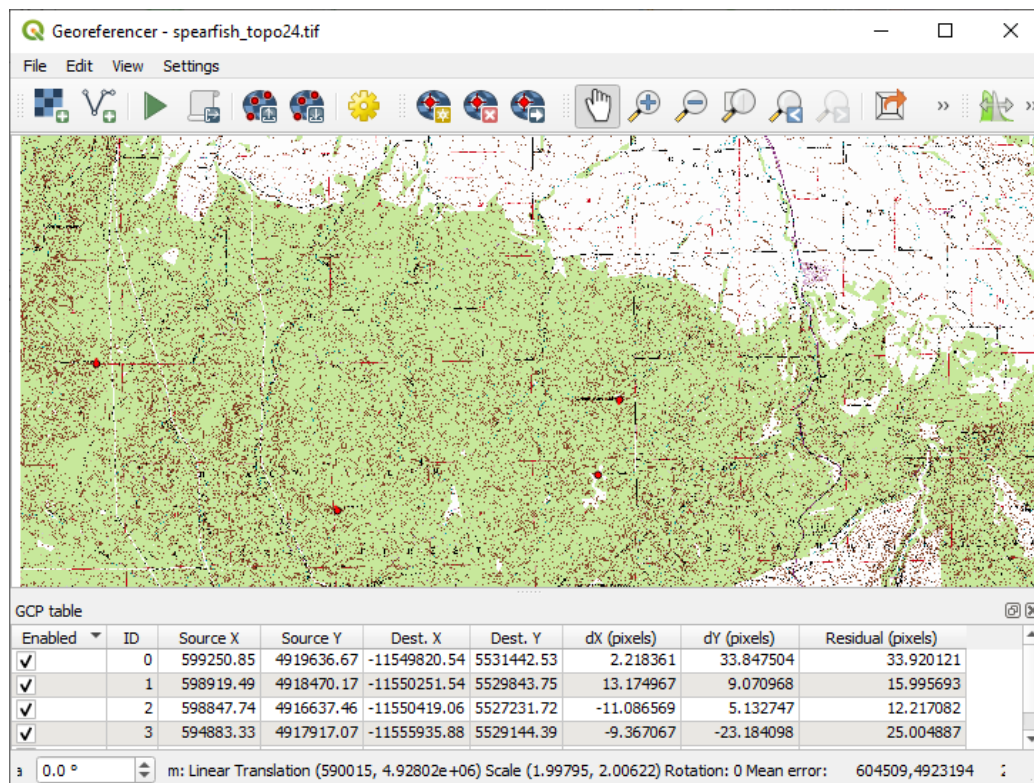








Abb. 17.24: Georeferencer Dialog

## Entering ground control points (GCPs)

1. To start georeferencing an unreferenced raster, we must load it using the  button. The raster will show up in the main working area of the dialog. Once the raster is loaded, we can start to enter reference points.
2. Using the  Add Point button, add points to the main working area and enter their coordinates (see Figure Abb. 17.25). For this procedure you have the following options:
  - Click on a point in the raster image and enter the X and Y coordinates manually, along with the CRS of the point.
  - Click on a point in the raster image and choose the  From map canvas button to add the X and Y coordinates with the help of a georeferenced map already loaded in the QGIS map canvas. The CRS will be set automatically.
  - When entering GCPs from the main map canvas, you have the option to hide the georeferencer window while selecting points from the main canvas. If the  Automatically hide georeferencer window checkbox is ticked, after clicking  From Map Canvas, the main georeferencer window will be hidden until a point is added on the map canvas. The Enter Map Coordinates dialog will remain open. If the box is unchecked, both windows will remain open while selecting a point on the map canvas. This option only takes effect when the georeferencer window is not docked in the main interface.
3. Continue entering points. You should have at least four points, and the more coordinates you can provide, the better the result will be. There are additional tools for zooming and panning the working area in order to locate a relevant set of GCP points.
4. With the  tool, you can move the GCPs in both the canvas and the georeferencing window, if you need to correct them.

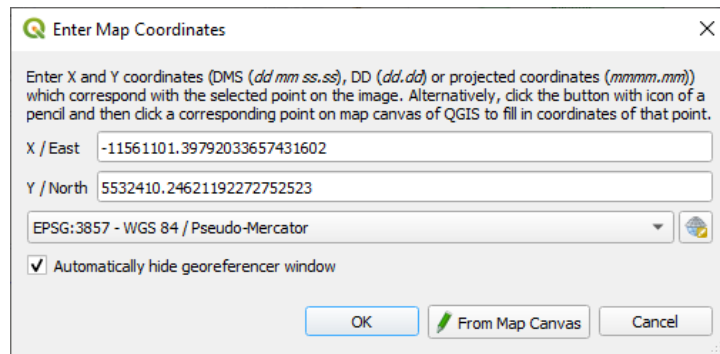




Abb. 17.25: Add points to the raster image

The points that are added to the map will be stored in a separate text file (`[filename].points`) usually together with the raster image. This allows us to reopen the Georeferencer at a later date and add new points or delete existing ones to optimize the result. The points file contains values of the form: `mapX`, `mapY`, `pixelX`, `pixelY`. You can use the  Load GCP points and  Save GCP points as buttons to manage the files.

## Defining the transformation settings

After you have added your GCPs to the raster image, you need to define the transformation settings for the georeferencing process.

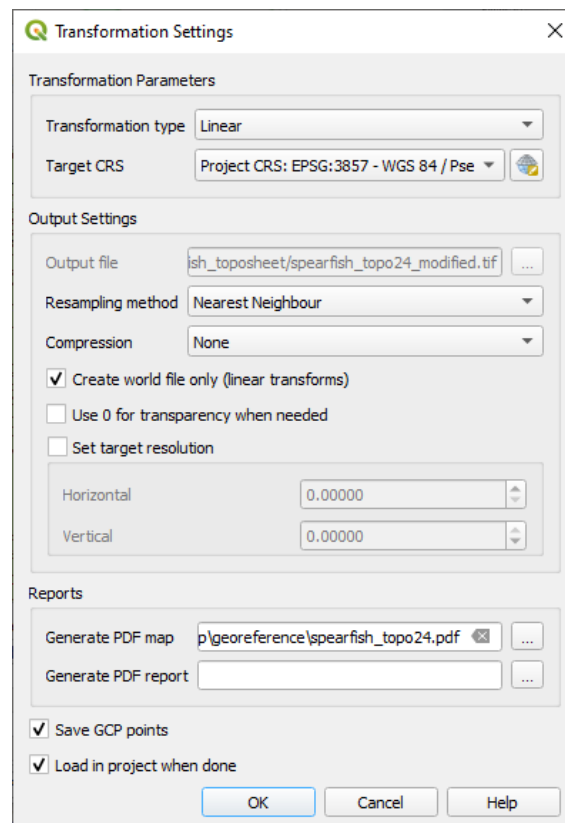


Abb. 17.26: Defining the georeferencer transformation settings

## Available Transformation algorithms

A number of transformation algorithms are available, dependent on the type and quality of input data, the nature and amount of geometric distortion that you are willing to introduce to the final result, and the number of ground control points (GCPs).

Currently, the following *Transformation types* are available:

- The **Linear** algorithm is used to create a world file and is different from the other algorithms, as it does not actually transform the raster pixels. It allows positioning (translating) the image and uniform scaling, but no rotation or other transformations. It is the most suitable if your image is a good quality raster map, in a known CRS, but is just missing georeferencing information. At least 2 GCPs are needed.
- The **Helmert** transformation also allows rotation. It is particularly useful if your raster is a good quality local map or orthorectified aerial image, but not aligned with the grid bearing in your CRS. At least 2 GCPs are needed.
- The **Polynomial 1** algorithm allows a more general affine transformation, in particular also a uniform shear. Straight lines remain straight (i.e., collinear points stay collinear) and parallel lines remain parallel. This is particularly useful for georeferencing data cartograms, which may have been plotted (or data collected) with different ground pixel sizes in different directions. At least 3 GCP's are required.
- The **Polynomial** algorithms 2-3 use more general 2nd or 3rd degree polynomials instead of just affine transformation. This allows them to account for curvature or other systematic warping of the image, for instance photographed maps with curving edges. At least 6 (respectively 10) GCP's are required. Angles and local scale are not preserved or treated uniformly across the image. In particular, straight lines may become curved, and there may be significant distortion introduced at the edges or far from any GCPs arising from extrapolating the data-fitted polynomials too far.
- The **Projective** algorithm generalizes Polynomial 1 in a different way, allowing transformations representing a central projection between 2 non-parallel planes, the image and the map canvas. Straight lines stay straight, but parallelism is not preserved and scale across the image varies consistently with the change in perspective. This transformation type is most useful for georeferencing angled photographs (rather than flat scans) of good quality maps, or oblique aerial images. A minimum of 4 GCPs is required.
- Finally, the **Thin Plate Spline** (TPS) algorithm „rubber sheets“ the raster using multiple local polynomials to match the GCPs specified, with overall surface curvature minimized. Areas away from GCPs will be moved around in the output to accommodate the GCP matching, but will otherwise be minimally locally deformed. TPS is most useful for georeferencing damaged, deformed, or otherwise slightly inaccurate maps, or poorly orthorectified aeriels. It is also useful for approximately georeferencing and implicitly reprojecting maps with unknown projection type or parameters, but where a regular grid or dense set of ad-hoc GCPs can be matched with a reference map layer. It technically requires a minimum of 10 GCPs, but usually more to be successful.

In all of the algorithms except TPS, if more than the minimum GCPs are specified, parameters will be fitted so that the overall residual error is minimized. This is helpful to minimize the impact of registration errors, i.e. slight imprecisions in pointer clicks or typed coordinates, or other small local image deformations. Absent other GCPs to compensate, such errors or deformations could translate into significant distortions, especially near the edges of the georeferenced image. However, if more than the minimum GCPs are specified, they will match only approximately in the output. In contrast, TPS will precisely match all specified GCPs, but may introduce significant deformations between nearby GCPs with registration errors.

## Define the Resampling method






The type of resampling you choose will likely depend on your input data and the ultimate objective of the exercise. If you don't want to change statistics of the raster (other than as implied by nonuniform geometric scaling if using other than the Linear, Helmert, or Polynomial 1 transformations), you might want to choose 'Nearest neighbour'. In contrast, 'cubic resampling', for instance, will usually generate a visually smoother result.

It is possible to choose between five different resampling methods:

1. Nearest neighbour
2. Bilinear (2x2 kernel)
3. Cubic (4x4 kernel)
4. Cubic B-Spline (4x4 kernel)
5. Lanczos (6x6 kernel)

## Define the transformation settings

There are several options that need to be defined for the georeferenced output raster.


- The  *Create world file* checkbox is only available if you decide to use the linear transformation type, because this means that the raster image actually won't be transformed. In this case, the *Output raster* field is not activated, because only a new world file will be created.
- For all other transformation types, you have to define an *Output raster*. As default, a new file ([filename]\_modified) will be created in the same folder together with the original raster image.
- As a next step, you have to define the *Target CRS* (Coordinate Reference System) for the georeferenced raster (see [Arbeiten mit Projektionen](#)).
- If you like, you can **generate a pdf map** and also **a pdf report**. The report includes information about the used transformation parameters, an image of the residuals and a list with all GCPs and their RMS errors.
- Furthermore, you can activate the  *Set Target Resolution* checkbox and define the pixel resolution of the output raster. Default horizontal and vertical resolution is 1.
- The  *Use 0 for transparency when needed* can be activated, if pixels with the value 0 shall be visualized transparent. In our example toposheet, all white areas would be transparent.
- The  *Save GCP Points* will store GCP Points in a file next to the output raster.
- Finally,  *Load in project when done* loads the output raster automatically into the QGIS map canvas when the transformation is done.

## Show and adapt raster properties


Clicking on the *Raster properties* option in the *Settings* menu opens the [Layer properties](#) dialog of the raster file that you want to georeference.

### Configure the georeferencer

You can customize the behavior of the georeferencer in *Settings ► Configure Georeferencer* (or use keyboard shortcut `Ctrl+P`).

- Under *Point Tip* you can use the checkboxes to toggle displaying GCP IDs and X/Y coordinates in both the Georeferencer window and the main map canvas.
- *Residual Units* controls whether residual units are given in pixels or map units
- *PDF Report* allows you to set margin size in mm for the report export
- *PDF Map* allows you to choose a paper size for the map export
- Finally, you can activate to  *Show Georeferencer window docked*. This will dock the Georeferencer window in the main QGIS window rather than showing it as a separate window that can be minimized.

### Running the transformation

After all GCPs have been collected and all transformation settings are defined, just press the  Start georeferencing button to create the new georeferenced raster.

---

## Working with Mesh Data

---

### 18.1 What's a mesh?

A mesh is an unstructured grid usually with temporal and other components. The spatial component contains a collection of vertices, edges and/or faces, in 2D or 3D space:

- **vertices** - XY(Z) points (in the layer's coordinate reference system)
- **edges** - connect pairs of vertices
- **faces** - a face is a set of edges forming a closed shape - typically a triangle or a quadrilateral (quad), rarely polygons with more vertices

Relying on the above, mesh layers can thus have different types of structure:

- 1D Meshes: consist of vertices and edges. An edge connects two vertices and can have assigned data (scalars or vectors) on it. The 1D mesh network can be for example used for modelling of an urban drainage system.
- 2D meshes: consist of faces with triangles, regular or unstructured quads.
- 3D layered meshes: consist of multiple stacked 2D unstructured meshes each extruded in the vertical direction (levels) by means of a vertical coordinate. The vertices and faces have the same topology in each vertical level. The mesh definition (vertical level extrusion) could in general change in time. The data is usually defined in volume centres or by some parametric function.

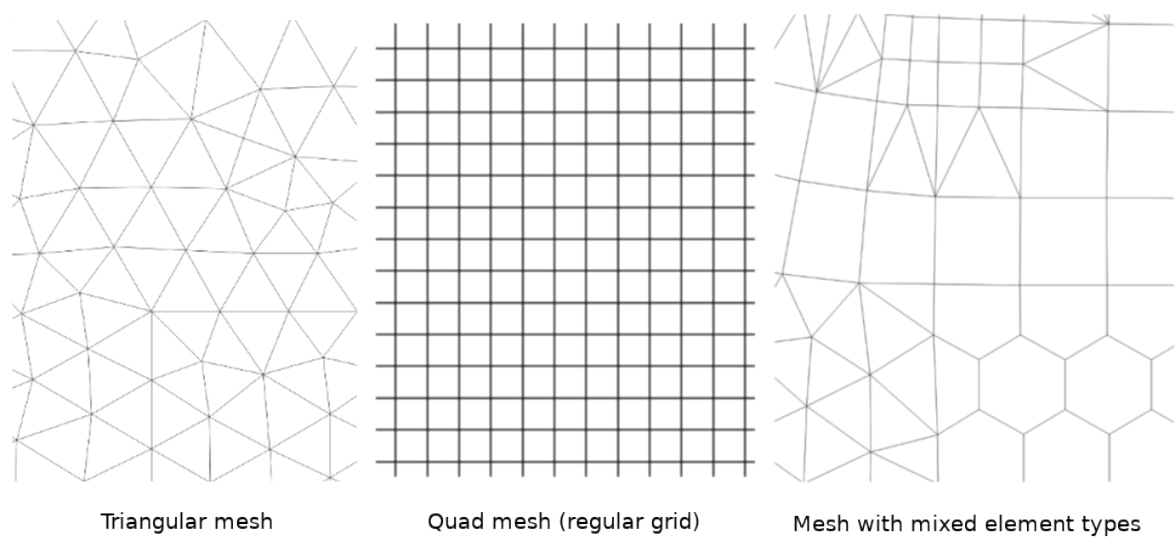


Abb. 18.1: Different mesh types

Mesh provides information about the spatial structure. In addition, the mesh can have datasets (groups) that assign a value to every vertex. For example, having a triangular mesh with numbered vertices as shown in the image below:

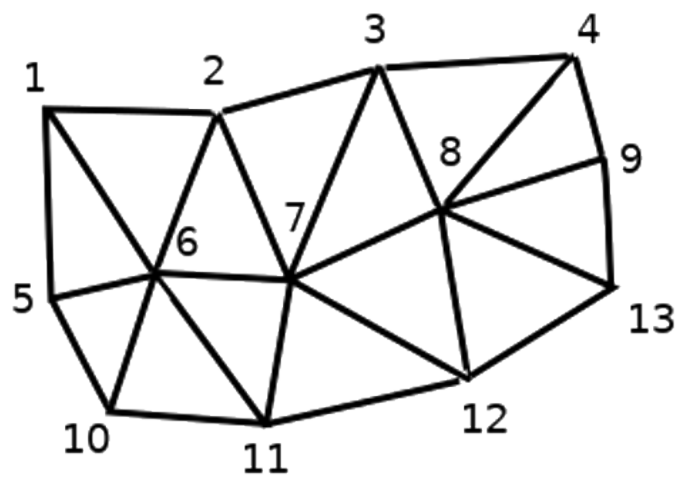


Abb. 18.2: Triangular grid with numbered vertices

Each vertex can store different datasets (typically multiple quantities), and those datasets can also have a temporal dimension. Thus, a single file may contain multiple datasets.

The following table gives an idea about the information that can be stored in mesh datasets. Table columns represent indices of mesh vertices, each row represents one dataset. Datasets can have different datatypes. In this case, it stores wind velocity at 10m at a particular moments in time (t1, t2, t3).

In a similar way, the mesh dataset can also store vector values for each vertex. For example, wind direction vector at the given time stamps:



Tab. 18.1: Example of mesh dataset

10 metre wind	1	2	3	...
10 metre speed at time=t1	17251	24918	32858	...
10 metre speed at time=t2	19168	23001	36418	...
10 metre speed at time=t3	21085	30668	17251	...
...	...	...	...	...
10m wind direction time=t1	[20,2]	[20,3]	[20,4.5]	...
10m wind direction time=t2	[21,3]	[21,4]	[21,5.5]	...
10m wind direction time=t3	[22,4]	[22,5]	[22,6.5]	...
...	...	...	...	...

We can visualize the data by assigning colors to values (similarly to how it is done with *Singleband pseudocolor* raster rendering) and interpolating data between vertices according to the mesh topology. It is common that some quantities are 2D vectors rather than being simple scalar values (e.g. wind direction). For such quantities it is desirable to display arrows indicating the directions.

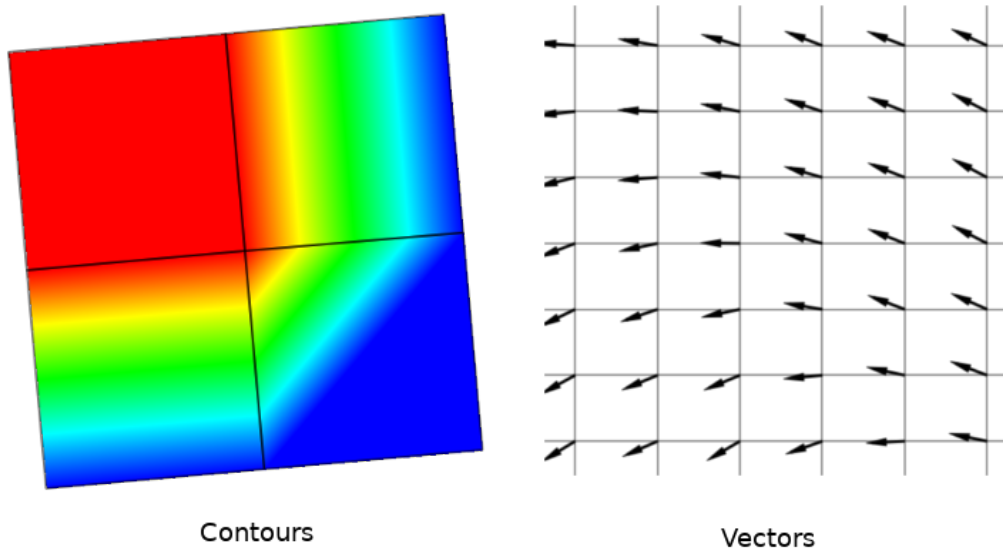



Abb. 18.3: Possible visualisation of mesh data

## 18.2 Supported formats

QGIS accesses mesh data using the *MDAL drivers*, and natively supports a *variety of formats*. Whether QGIS can edit a mesh layer depends on the format and the mesh structure type.

To load a mesh dataset into QGIS, use the  *Mesh* tab in the *Data Source Manager* dialog. Read *Loading a mesh layer* for more details.

## 18.3 Mesh Dataset Properties

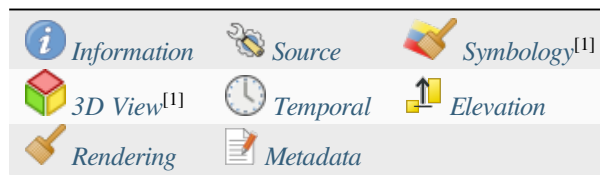
The *Layer Properties* dialog for a mesh layer provides general settings to manage dataset groups of the layer and their rendering (active dataset groups, symbology, 2D and 3D rendering). It also provides information about the layer.

To access the *Layer Properties* dialog:

- In the *Layers* panel, double-click the layer or right-click and select *Properties...* from the pop-up menu;
- Go to *Layer ► Layer Properties...* menu when the layer is selected.

The mesh *Layer Properties* dialog provides the following sections:

Tab. 18.2: Tabs of the Mesh Layer Properties



<sup>[1]</sup> Also available in the *Layer styling panel*

**Bemerkung:** Most of the properties of a mesh layer can be saved to or loaded from a `.qml` using the *Style* menu at the bottom of the dialog. More details at *Verwalten von benutzerdefinierten Stilen*.

### 18.3.1 Information Properties

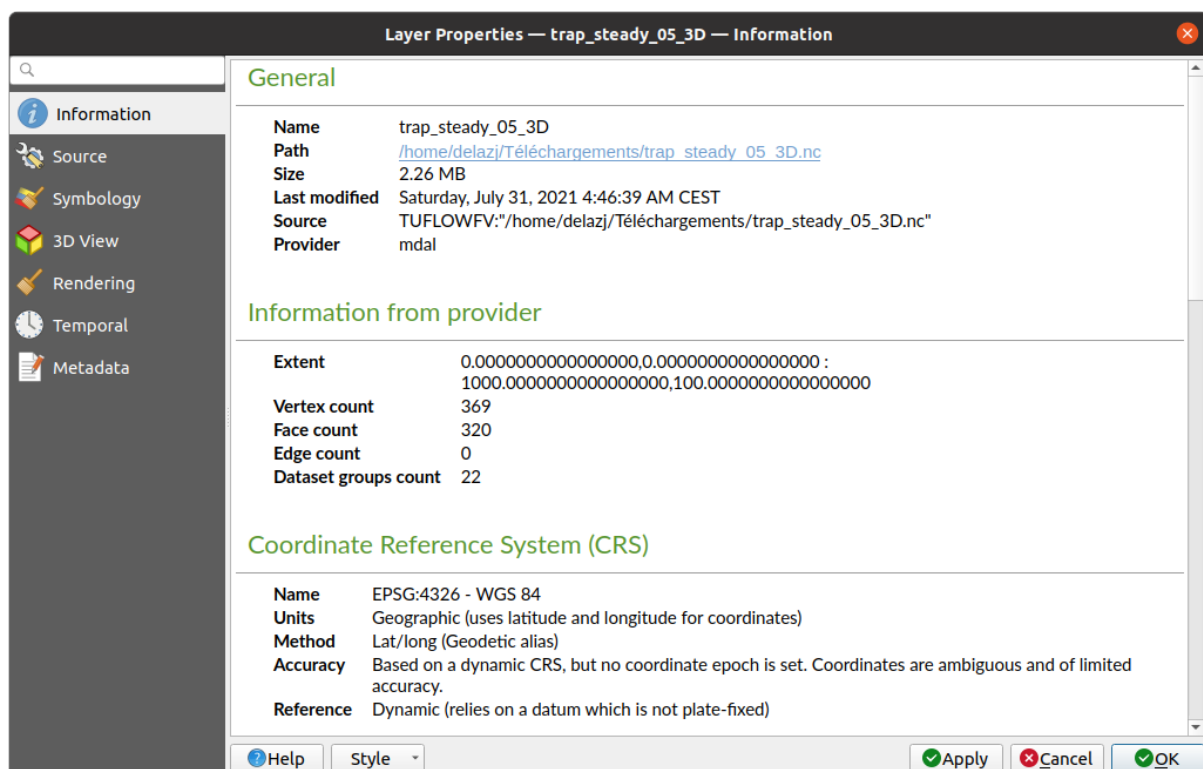




Abb. 18.4: Mesh Layer Information Properties

The  *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata on the current layer. Provided information are:

- general such as name in the project, source path, list of auxiliary files, last save time and size, the used provider
- based on the provider of the layer: extent, vertex, face, edges and/or dataset groups count
- the Coordinate Reference System: name, units, method, accuracy, reference (i.e. whether it's static or dynamic)
- extracted from filled *metadata*: access, extents, links, contacts, history...

### 18.3.2 Source Properties

The  *Source* tab displays basic information about the selected mesh, including:

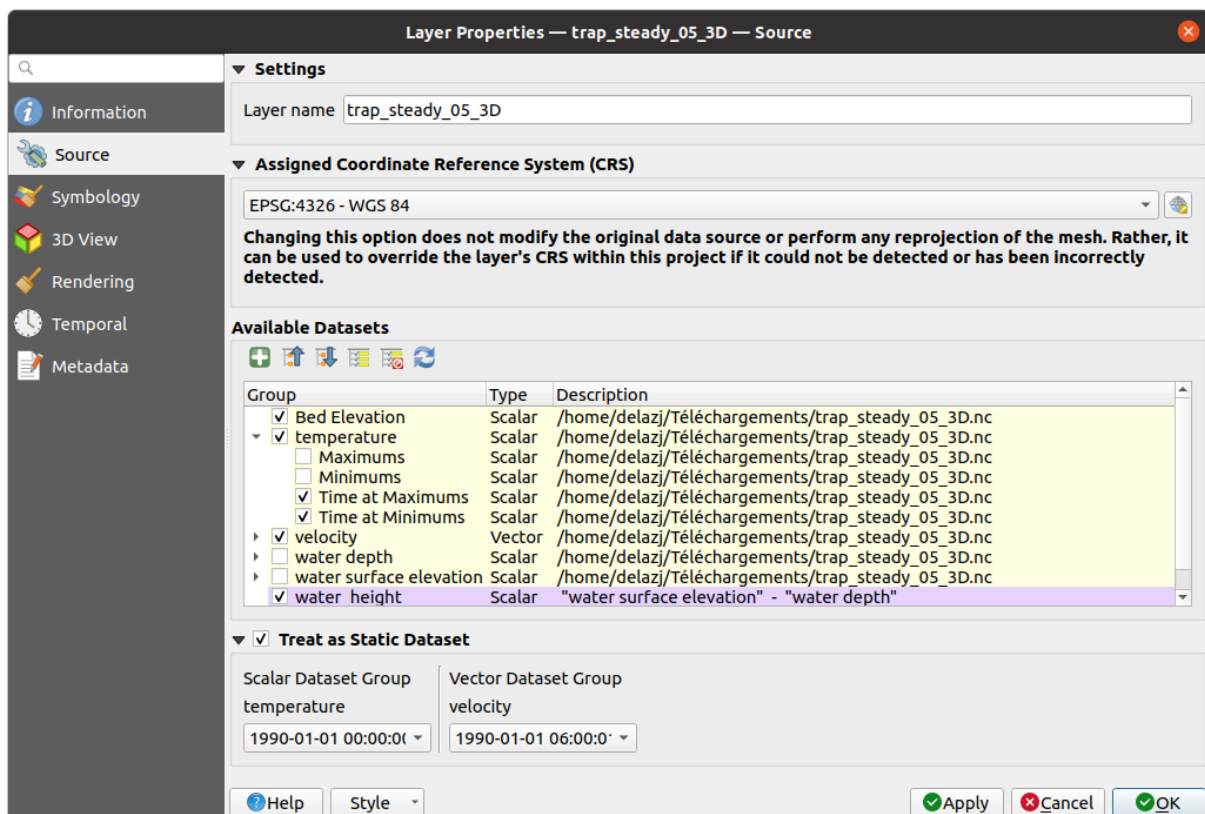











Abb. 18.5: Mesh Layer Source Properties

- the layer name to display in the *Layers* panel
- setting the Coordinate Reference System: Displays the layer's *Assigned Coordinate Reference System (CRS)*. You can change the layer's CRS by selecting a recently used one in the drop-down list or clicking on  *Select CRS* button (see *Coordinate Reference System Selector*). Use this process only if the CRS applied to the layer is wrong or if none was applied.
- The *Available datasets* frame lists all the dataset groups (and subgroups) in the mesh layer, with their type and description in a tree view. Both regular datasets (i.e. their data is stored in the file) and virtual datasets (which are *calculated on the fly*) are listed.
  - Use the  *Assign extra dataset to mesh* button to add more groups to the current mesh layer.
  -  *Collapse all* and  *Expand all* the dataset tree, in case of embedded groups


- If you are interested in few datasets, you can uncheck the others and make them unavailable in the project
- Double-click over a name and you can rename the dataset.
-  *Reset to defaults*: checks all the groups and renames them back to their original name in the provider.
- Right-click over a virtual dataset group and you can:
  - \* *Remove dataset group* from the project
  - \* *Save dataset group as...* a file on disk, to any supported format. The new file is kept assigned to the current mesh layer in the project.
- Checking the  *Treat as static dataset* group allows to ignore the *map temporal navigation* properties while rendering the mesh layer. For each active dataset group (as selected in  *Symbology* ►  *Datasets* tab), you can:
  - set to *None*: the dataset group is not displayed at all
  - *Display dataset*: e.g., for the „bed elevation“ dataset which is not time aware
  - extract a particular date time: the dataset matching the provided time is rendered and stay fixed during map navigation.



### 18.3.3 Symbology Properties

Click the  *Symbology* button to activate the dialog. Symbology properties are divided into several tabs:

- *Datasets*
- *Contours*
- *Vectors*
- *Rendering*
- *Stacked mesh averaging method*

#### Datasets

The tab  *Datasets* is the main place to control and set which datasets will be used for the layer. It presents the following items:

- *Groups* available in the mesh dataset, with whether they provide:
  -  scalar dataset
  - or  vector dataset: by default, each vector dataset has a scalar dataset representing its magnitude automatically generated.

Click on the icon next to the dataset name to select the group and type of data to represent.

- *Selected dataset group(s) metadata*, with details on:
  - the mesh type: edges or faces
  - the data type: vertices, edges, faces or volume
  - whether it's of vector type or not
  - the original name in the mesh layer
  - the unit, if applicable
- *blending mode* available for the selected datasets.

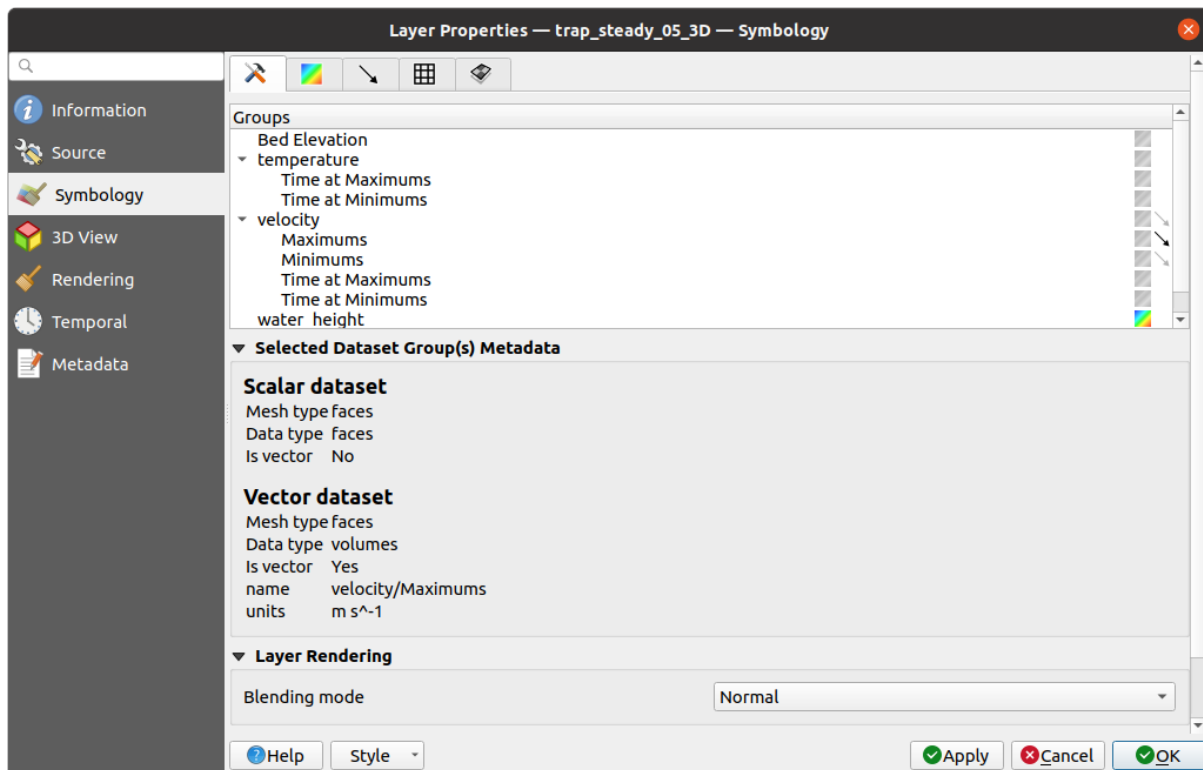





Abb. 18.6: Mesh Layer Datasets

You can apply symbology to the selected vector and/or scalar group using the next tabs.

## Contours Symbology

**Bemerkung:** The  Contours tab can be activated only if a scalar dataset has been selected in the  Datasets tab.

In the  Contours tab you can see and change the current visualization options of contours for the selected group, as shown in Abb. 18.7 below:

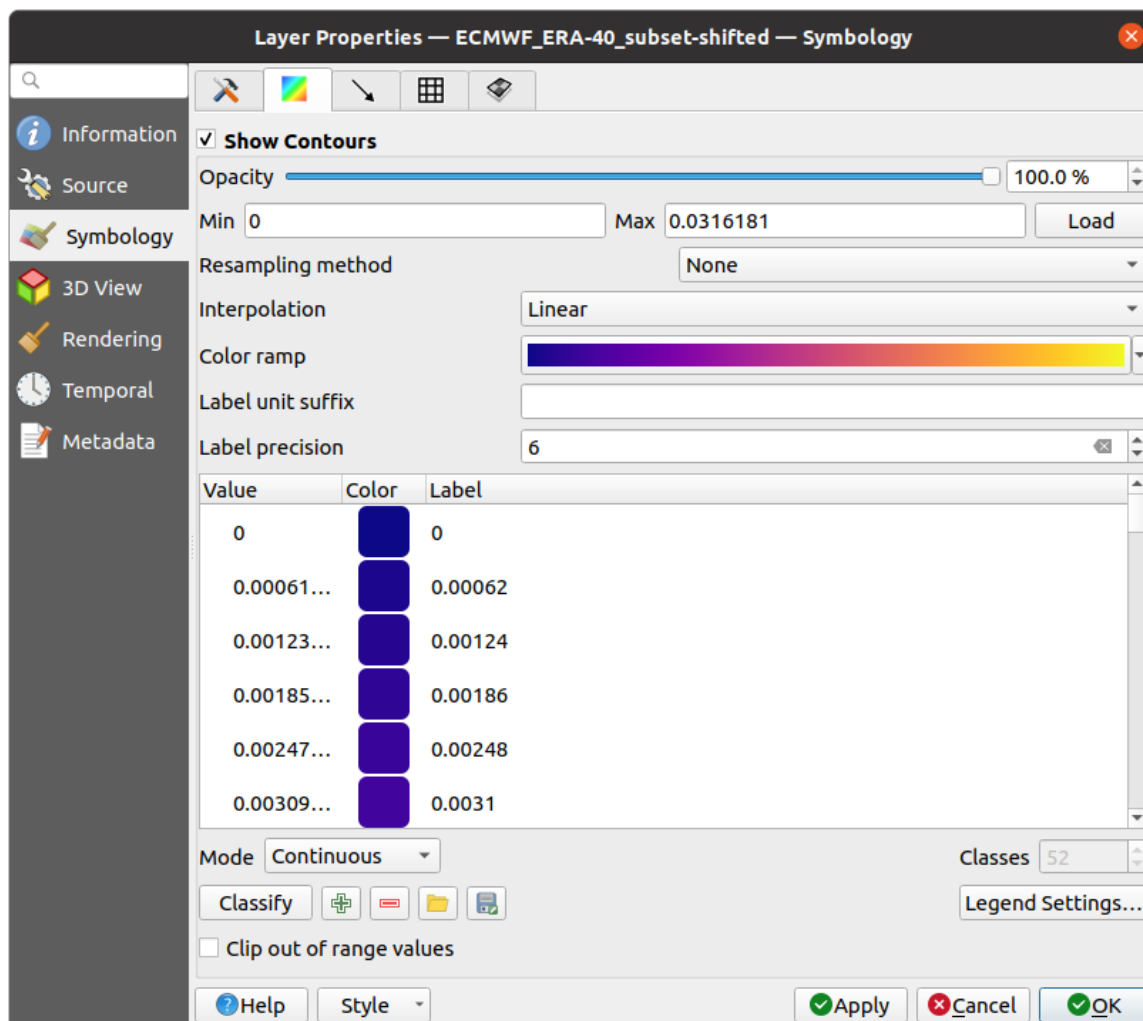






Abb. 18.7: Styling Contours in a Mesh Layer

- For 1D mesh, set the *Stroke width* of the edges. This can be a fixed size for the whole dataset, or vary along the geometry (more details with the *interpolated line renderer*)
- Use the slider or the spinbox to set the *Opacity* of the current group, if of a 2D mesh type.
- Enter the range of values you want to represent on the current group: use  Load to fetch the min and max values of the current group or enter custom values if you want to exclude some.
- For 2D/3D meshes, select the *Resampling method* to interpolate the values on the surrounding vertices to the faces (or from the surrounding faces to the vertices) using the *Neighbour average* method. Depending on whether the dataset is defined on the vertices (respectively on the faces), QGIS defaults this setting to *None* (respectively *Neighbour average*) method in order to use values on vertices and keep the default rendering smooth.
- Classify the dataset using the *color ramp shader* classification.

## Vectors Symbolology

**Bemerkung:** The  **Vectors** tab can be activated only if a vector dataset has been selected in the  **Datasets** tab.

In the  **Vectors** tab you can see and change the current visualization options of vectors for the selected group, as shown in Abb. 18.8:

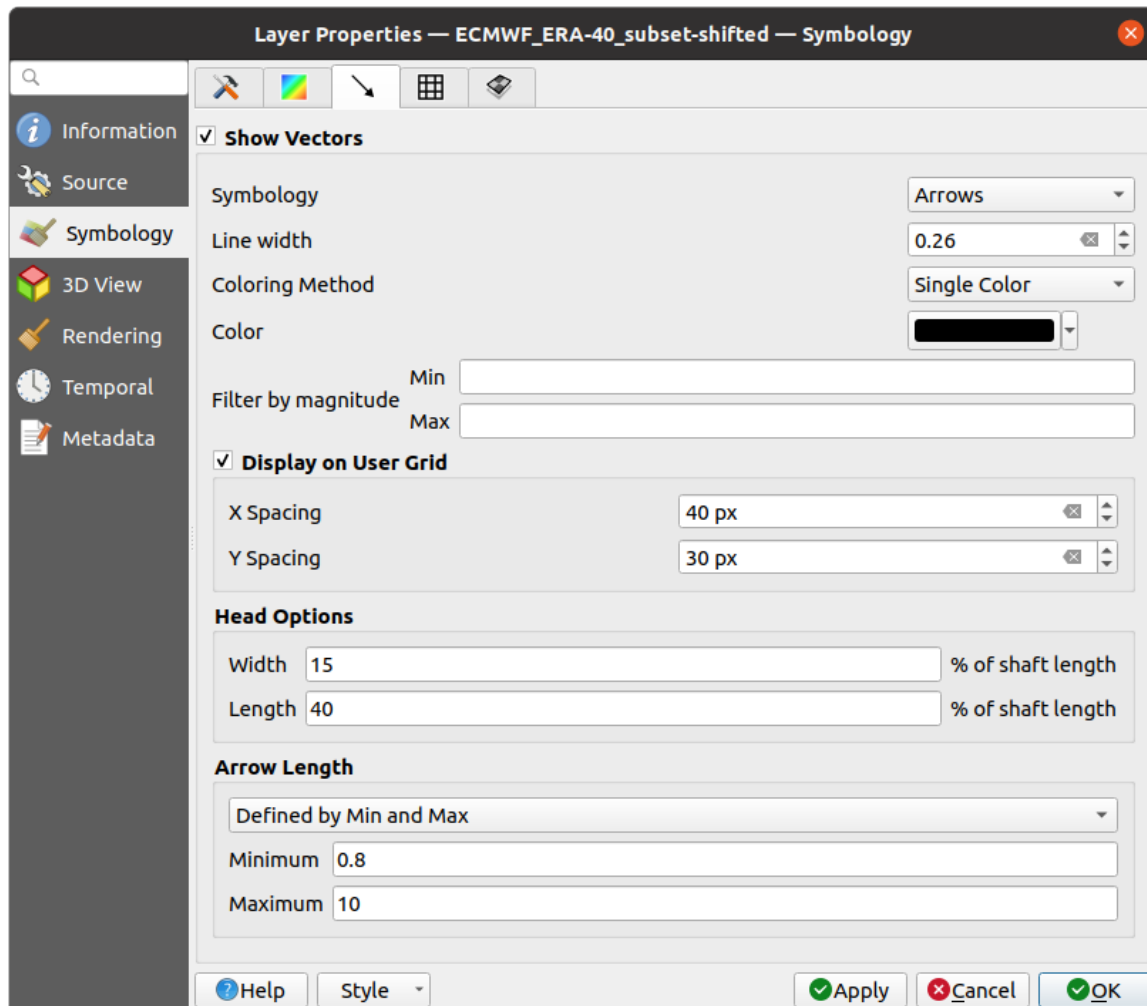














Abb. 18.8: Styling Vectors in a Mesh Layer with arrows

Mesh vector dataset can be styled using various types of *Symbology*:

- **Arrows:** vectors are represented with arrows at the same place as they are defined in the raw dataset (i.e. on the nodes or center of elements) or on a user-defined grid (hence, they are evenly distributed). The arrow length is proportional to the magnitude of the arrow as defined in the raw data but can be scaled by various methods.
- **Streamlines:** vectors are represented with streamlines seeded from start points. The seeding points can start from the vertices of the mesh, from a user grid or randomly.
- **Traces:** a nicer animation of the streamlines, the kind of effect you get when you randomly throws sand in the water and see where the sand items flows.

Available properties depend on the selected symbology as shown in the following table.

Tab. 18.3: Availability and meaning of the vectors symbology properties

Label	Description and Properties	Arrow	Streamlines	Traces
<i>Line width</i>	Width of the vector representation			
<i>Coloring method</i>	<ul style="list-style-type: none"> <li>a <i>Single color</i> assigned to all vectors</li> <li>or a variable color based on vectors magnitude, using a <i>Color ramp shader</i></li> </ul>			
<i>Filter by magnitude</i>	Only vectors whose length for the selected dataset falls between a <i>Min</i> and <i>Max</i> range are displayed			
<i>Display on user grid</i>	Places the vector on a grid with custom <i>X spacing</i> and <i>Y spacing</i> and interpolates their length based on neighbours			
<i>Head options</i>	<i>Length</i> and <i>Width</i> of the arrow head, as a percentage of its shaft length			
<i>Arrow length</i>	<ul style="list-style-type: none"> <li><b>Defined by Min and Max:</b> You specify the minimum and maximum length for the arrows, QGIS will interpolate their size based on the underlying vector's magnitude</li> <li><b>Scale to magnitude:</b> arrow length is proportional to their vector's magnitude</li> <li><b>Fixed:</b> all the vectors are shown with</li> </ul>			



## Rendering

In the  **Rendering** tab, QGIS offers possibilities to display and customize the mesh structure. *Line width* and *Line color* can be set to represent:

- the edges for 1D meshes
- For 2D meshes:
  - *Native mesh rendering*: shows original faces and edges from the layer
  - *Triangular mesh rendering*: adds more edges and displays the faces as triangles

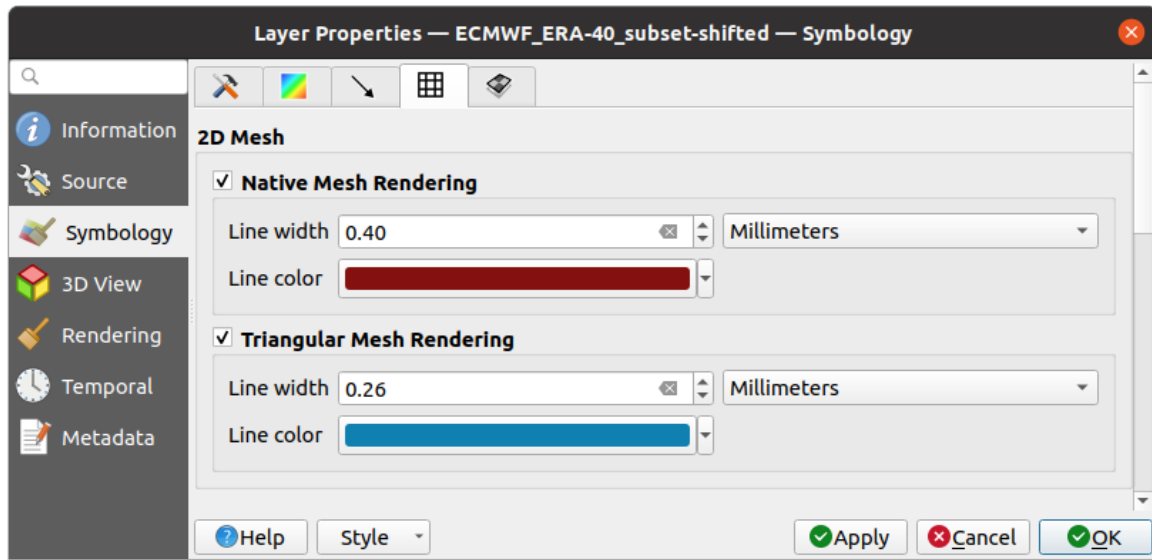


Abb. 18.9: 2D Mesh Rendering

### Stacked mesh averaging method


3D layered meshes consist of multiple stacked 2D unstructured meshes each extruded in the vertical direction (*levels*) by means of a vertical coordinate. The vertices and faces have the same topology in each vertical level. Values are usually stored on the volumes that are regularly stacked over base 2d mesh. In order to visualise them on 2D canvas, you need to convert values on volumes (3d) to values on faces (2d) that can be shown in mesh layer. The



Stacked mesh averaging method provides different averaging/interpolation methods to handle this.

You can select the method to derive the 2D datasets and corresponding parameters (level index, depth or height values). For each method, an example of application is shown in the dialog but you can read more on the methods at [https://fvwiki.tuflow.com/index.php?title=Depth\\_Averaging\\_Results](https://fvwiki.tuflow.com/index.php?title=Depth_Averaging_Results).

### 18.3.4 3D View Properties

Mesh layers can be used as *terrain in a 3D map view* based on their vertices Z values. From the  **3D View** properties tab, it's also possible to render the mesh layer's dataset in the same 3D view. Therefore, the vertical component of the vertices can be set equal to dataset values (for example, level of water surface) and the texture of the mesh can be set to render other dataset values with color shading (for example velocity).

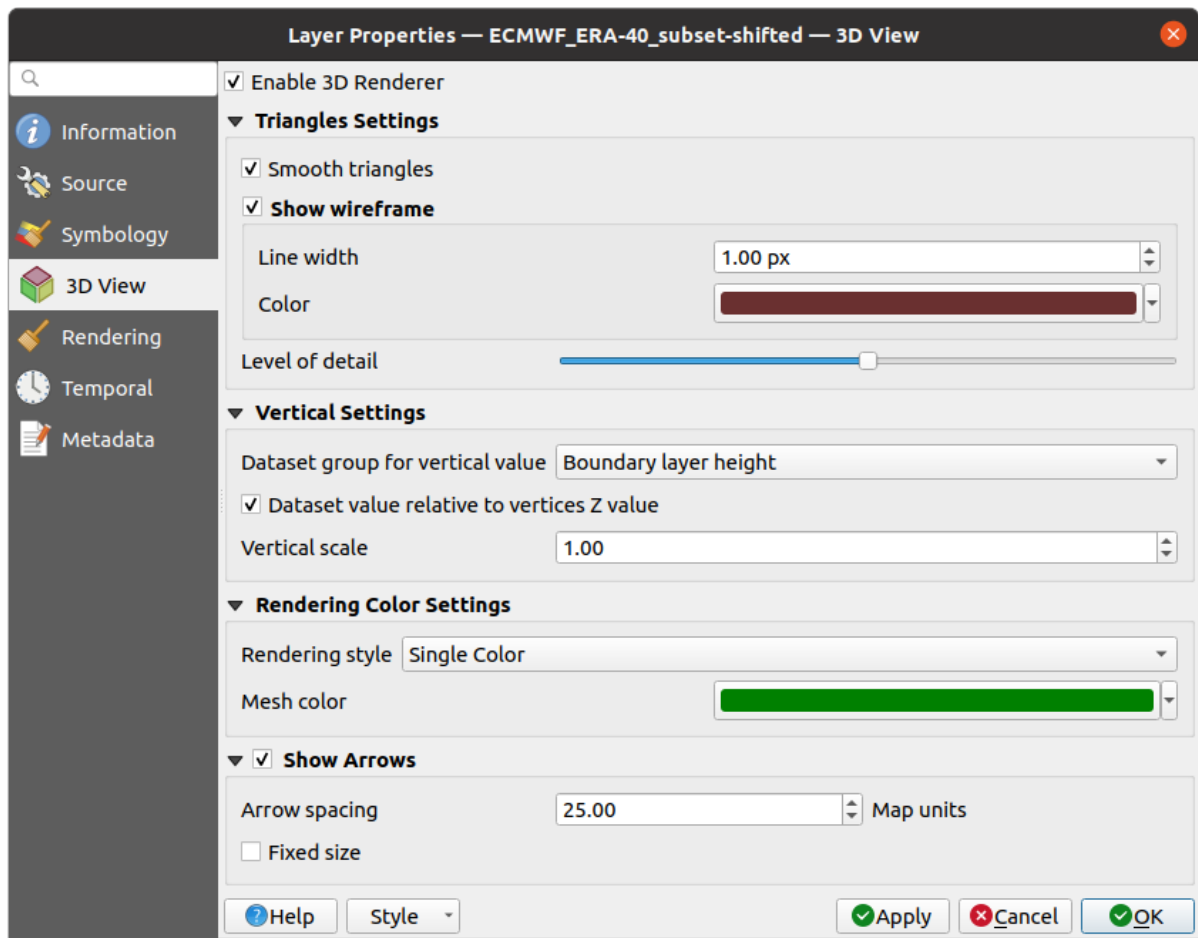


Abb. 18.10: Mesh dataset 3D properties

Check ☒ *Enable 3D Renderer* and you can edit following properties:

- Under *Triangle settings*
  - *Smooth triangles*: Angles between consecutive triangles are smoothed for a better 3D rendering
  - *Show wireframe* whose you can set the *Line width* and *Color*
  - *Level of detail*: Controls how *simplified* the mesh layer to render should be. On the far right, it is the base mesh, and the more you go left, the more the layer is simplified and is rendered with less details. This option is only available if the *Simplify mesh* option under the *Rendering* tab is activated.
- **Vertical settings to control behavior of the vertical component** of vertices of rendered triangles
  - *Dataset group for vertical value*: the dataset group that will be used for the vertical component of the mesh
  - ☐ *Dataset value relative to vertices Z value*: whether to consider the dataset values as absolute Z coordinate or relative to the vertices native Z value
  - *Vertical scale*: the scale factor to apply to the dataset Z values
- *Rendering color settings* with a *Rendering style* that can be based on the color ramp shader set in *Contours Symbolology* (2D contour color ramp shader) or as a *Single color* with an associated *Mesh color*
- *Show arrows*: displays arrows on mesh layer dataset 3D entity, based on the same vector dataset group used in the *vector 2D rendering*. They are displayed using the 2D color setting. It's also possible to define the *Arrow spacing* and, if it's of a *Fixed size* or scaled on magnitude. This spacing setting defines also the max size of arrows because arrows can't overlap.

### 18.3.5 Rendering Properties

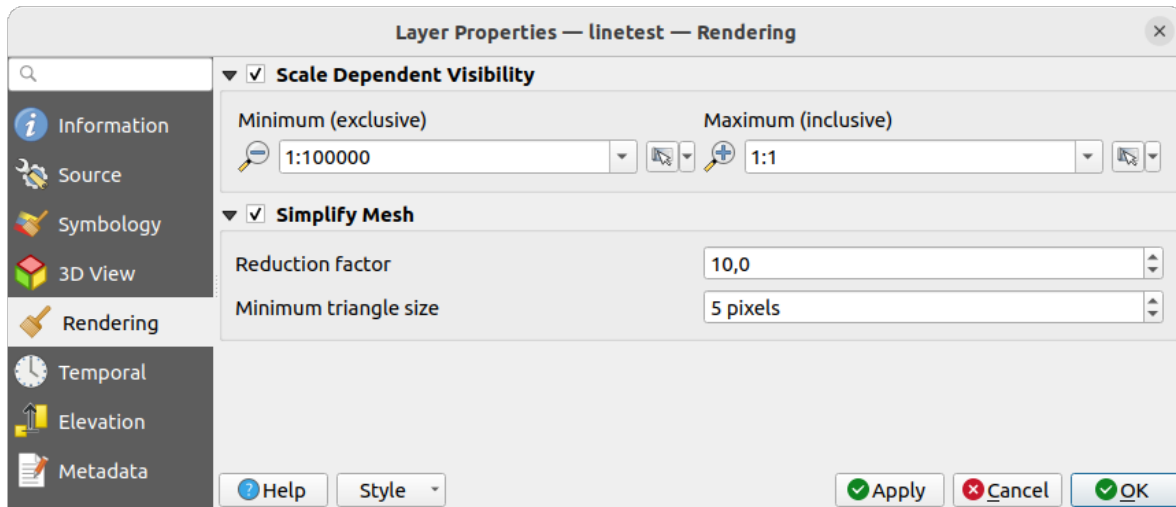





Abb. 18.11: Mesh rendering properties

Under the *Scale dependent visibility* group box, you can set the *Maximum (inclusive)* and *Minimum (exclusive)* scale, defining a range of scale in which mesh elements will be visible. Out of this range, they are hidden. The  button helps you use the current map canvas scale as boundary of the range visibility. See [Visibility Scale Selector](#) for more information.


**Bemerkung:** You can also activate scale dependent visibility on a layer from within the *Layers* panel: right-click on the layer and in the contextual menu, select *Set Layer Scale Visibility*.

As mesh layers can have millions of faces, their rendering can sometimes be very slow, especially when all the faces are displayed in the view whereas they are too small to be viewed. To speed up the rendering, you can simplify the mesh layer, resulting in one or more meshes representing different *levels of detail* and select at which level of detail you would like QGIS to render the mesh layer. Note that the simplify mesh contains only triangular faces.

From the  *Rendering* tab, check  *Simplify mesh* and set:

- a *Reduction factor*: Controls generation of successive levels of simplified meshes. For example, if the base mesh has 5M faces, and the reduction factor is 10, the first simplified mesh will have approximately 500 000 faces, the second 50 000 faces, the third 5000,... If a higher reduction factor leads quickly to simpler meshes (i.e. with triangles of bigger size), it produces also fewer levels of detail.
- *Minimum triangle size*: the average size (in pixels) of the triangles that is permitted to display. If the average size of the mesh is lesser than this value, the rendering of a lower level of details mesh is triggered.

### 18.3.6 Temporal Properties

The  *Temporal* tab provides options to control the rendering of the layer over time. It allows to dynamically display temporal values of the enabled dataset groups. Such a dynamic rendering requires the *temporal navigation* to be enabled over the map canvas.

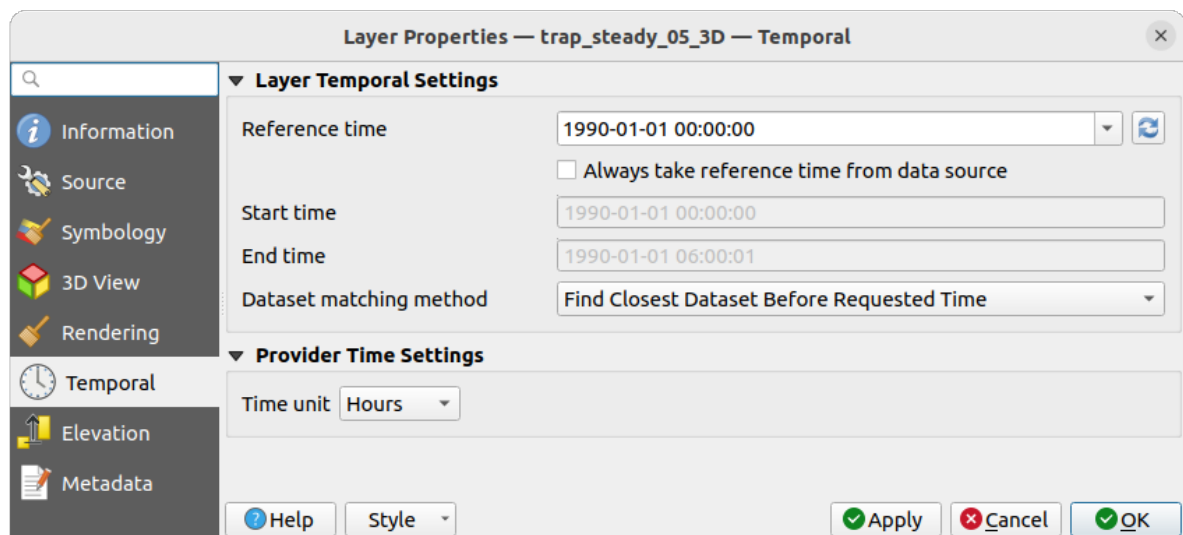




Abb. 18.12: Mesh Temporal properties

### Layer temporal settings

- *Reference time* of the dataset group, as an absolute date time. By default, QGIS parses the source layer and returns the first valid reference time in the layer's dataset group. If unavailable, the value will be set by the project time range or fall back to the current date. The *Start time* and *End time* to consider are then calculated based on the internal timestamp step of the dataset.


It is possible to set a custom *Reference time* (and then the time range), and revert the changes using the  Reload from provider button. With  *Always take reference time from data source* checked, you ensure that the time properties are updated from the file each time the layer is reloaded or the project reopened.

- *Dataset matching method*: determines the dataset to display at the given time. Options are *Find closest dataset before requested time* or *Find closest dataset from requested time (after or before)*.

### Provider time settings

- *Time unit* extracted from the raw data, or user defined. This can be used to align the speed of the mesh layer with other layers in the project during map time navigation. Supported units are *Seconds*, *Minutes*, *Hours* and *Days*.

## 18.3.7 Elevation Properties

The  *Elevation* tab provides options to control the layer elevation properties within a *3D map view* and its appearance in the *profile tool charts*. Specifically, you can set:

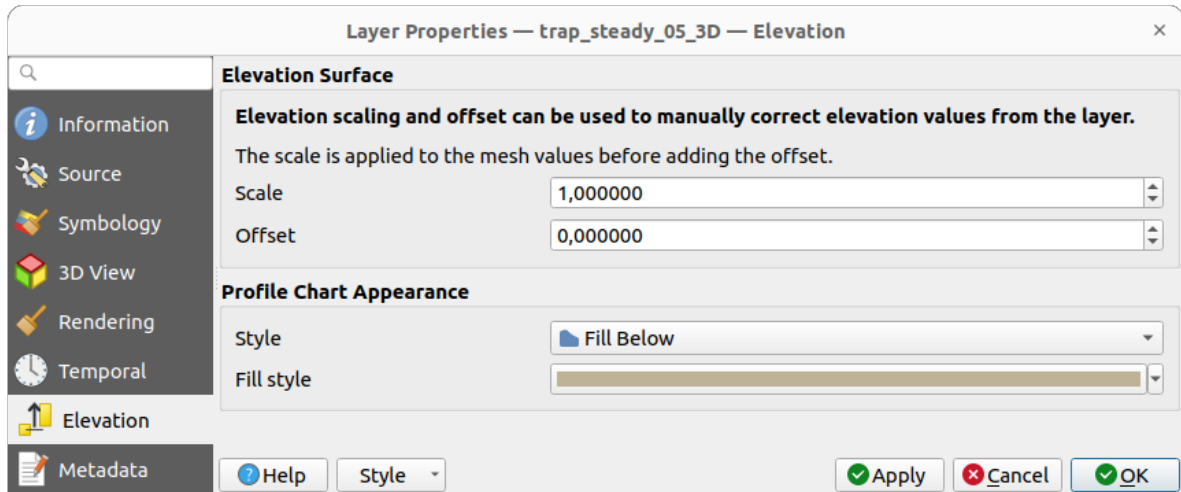



Abb. 18.13: Mesh Elevation properties

- *Elevation Surface*: how the mesh layer vertices Z values should be interpreted as terrain elevation. You can apply a *Scale* factor and an *Offset*.
- *Profile Chart Appearance*: controls the rendering of the mesh elements elevation in the profile chart. The profile *Style* can be set as:
  - a *Line* with a specific *Line style*
  - an elevation surface rendered using a fill symbol either above (*Fill above*) or below (*Fill below*) the elevation curve line. The surface symbology is represented using:
    - \* a *Fill style*
    - \* and a *Limit*: the maximum (respectively minimum) altitude determining how high the fill surface will be

### 18.3.8 Metadata Properties

The  *Metadata* tab provides you with options to create and edit a metadata report on your layer. See [Metadata](#) for more information.

## 18.4 Editing a mesh layer

QGIS allows to *create a mesh layer* from scratch or based on an existing one. You can create/modify the geometries of the new layer whom you can assign datasets afterwards. It's also possible to edit an existing mesh layer. Because the editing operation requires a frames-only layer, you will be asked to either remove any associated datasets first (make sure you have them available if they still are necessary) or create a copy (only geometries) of the layer.

---













**Bemerkung:** QGIS does not allow to digitize edges on mesh layers. Only vertices and faces are mesh elements that can be created. Also not all supported mesh formats can be edited in QGIS (see [permissions](#)).

---

### 18.4.1 Overview of the mesh digitizing tools


To interact with or edit a base mesh layer element, following tools are available.

Tab. 18.4: Tools for mesh digitizing

Label	Purpose	Location	
 Current Edits	Access to save, rollback or cancel changes in all or selected layers simultaneously	Digitizing toolbar	
 Toggle to Edit	Turn on/off the layer in edit mode	Digitizing toolbar	
 Save Edits	Save changes done to the layer	Digitizing toolbar	
 Undo	Undo the last change(s) - Ctrl+Z	Digitizing toolbar	
 Redo	Redo the last undone action(s) - Ctrl+Shift+Z	Digitizing toolbar	
 Enable Advanced Digitizing tools	Turn on/off the <i>Advanced Digitizing Panel</i>	Advanced	Digitizing toolbar
 Reindex Faces and Vertices	Recreate index and renumber the mesh elements for optimization	Mesh menu	
 Digitize Mesh Elements	Select/Create vertices and faces	Mesh toolbar	Digitizing
 Select Mesh Elements by Polygon	Select vertices and faces overlapped by a drawn polygon	Mesh toolbar	Digitizing
 Select Mesh Elements by Expression	Select vertices and faces using an expression	Mesh toolbar	Digitizing
 Transform Vertices Coordinates	Modify coordinates of a selection of vertices	Mesh toolbar	Digitizing
 Force by Selected Geometries	Split faces and constrain Z value using a linear geometry	Mesh toolbar	Digitizing

### 18.4.2 Exploring the Z value assignment logic



When a mesh layer is turned into edit mode, a *Vertex Z value* widget opens at the top right of the map canvas. By default, its value corresponds to the *Default Z value* set in *Settings ► Options ► Digitizing* tab. When there are selected vertices, the widget displays the average Z value of the selected vertices.

During editing, the *Vertex Z value* is assigned to new vertices. It is also possible to set a custom value: edit the widget, press **Enter** and you will override the default value and make use of this new value in the digitizing process. Click the  icon in the widget to reset its value to the Options default value.

#### Rules of assignment

When **creating** a new vertex, its Z value definition may vary depending on the active selection in the mesh layer and its location. The following table displays the various combinations.

Tab. 18.5: Matrix of Z value assignment to new vertex

Vertex creation	Are there selected vertices in mesh layer?	Source of assigned value	Assigned Z Value
„Free“ vertex, not connected to any face or edge of a face	No	<i>Vertex Z value</i>	Default or user defined
		<i>Advanced Digitizing Panel</i> (if <i>z</i> widget is in  Locked state)	<i>z</i> widget if in  Locked state
	Yes	<i>Vertex Z value</i>	Average of the selected vertices
Vertex on an edge	—	Mesh layer	Interpolated from the edge's vertices
Vertex on a face	—	Mesh layer	Interpolated from the face's vertices
Vertex snapped to a 2D vector feature	—	<i>Vertex Z value</i>	Default or user defined
Vertex snapped to a 3D vector vertex	—	Vector layer	Vertex
Vertex snapped to a 3D vector segment	—	Vector layer	Interpolated along the vector segment

**Bemerkung:** The *Vertex Z value* widget is deactivated if the *Advanced Digitizing Panel* is enabled and no mesh element is selected. The latter's *z* widget then rules the Z value assignment.

## Modifying Z value of existing vertices

To modify the Z value of vertices, the most straightforward way is:


1. Select one or many vertices. The *Vertex Z value* widget will display the average height of the selection.
2. Change the value in the widget.
3. Press **Enter**. The entered value is assigned to the vertices and becomes the default value of next vertices.

Another way to change the Z value of a vertex is to move and snap it on a vector layer feature with the Z value capability. If more than one vertex are selected, the Z value can't be changed in this way.

The *Transform mesh vertices* dialog also provides means to modify the Z value of a selection of vertices (along with their X or Y coordinates).

## 18.4.3 Selecting mesh elements

### Using *Digitize Mesh Elements*

Activate the  *Digitize Mesh Elements* tool. Hover over an element and it gets highlighted, allowing you to select it.

- Click on a vertex, and it is selected.
- Click on the small square at the center of a face or an edge, and it gets selected. Connected vertices are also selected. Conversely, selecting all the vertices of an edge or a face also selects that element.
- Drag a rectangle to select overlapping elements (a selected face comes with all their vertices). Press **Alt** key if you want to select only completely contained elements.
- To add elements to a selection, press **Shift** while selecting them.


- To remove an element from the selection, press **Ctrl** and reselect it. A deselected face will also deselect all their vertices.




### Using *Select Mesh Elements by Polygon*

Activate the  *Select Mesh Elements by Polygon* tool and:

- Draw a polygon (left-click to add vertex, **Backspace** to undo last vertex, **Esc** to abort the polygon and right-click to validate it) over the mesh geometries. Any partially overlapping vertices and faces will get selected. Press **Alt** key while drawing if you want to select only completely contained elements.
- Right-click over the geometry of a vector layer's feature, select it in the list that pops up and any partially overlapping vertices and faces of the mesh layer will get selected. Use **Alt** while drawing to select only completely contained elements.
- To add elements to a selection, press **Shift** while selecting them.
- To remove an element from the selection, press **Ctrl** while drawing over the selection polygon.

### Using *Select Mesh Elements by Expression*


Another tool for mesh elements selection is  *Select Mesh Elements by Expression*. When pressed, the tool opens the mesh *expression selector dialog* from which you can:

1. Select the method of selection:
  - *Select by vertices*: applies the entered expression to vertices, and returns matching ones and their eventually associated edges/faces
  - *Select by faces*: applies the entered expression to faces, and returns matching ones and their associated edges/vertices
2. Write the expression of selection. Depending on the selected method, available functions in the *Mesches group* will be filtered accordingly.
3. Run the query by setting how the selection should behave and pressing:
  -  *Select*: replaces any existing selection in the layer
  -  *Add to current selection*
  -  *Remove from current selection*

## 18.4.4 Modifying mesh elements

### Adding vertices

To add vertices to a mesh layer:


1. Press the  *Digitize mesh elements* button
2. A *Vertex Z value* widget appears on the top right corner of the map canvas. Set this value to the Z coordinate you would like to assign to the subsequent vertices
3. Then double-click:
  - outside a face: adds a „free vertex“, that is a vertex not linked to any face. This vertex is represented by a red dot when the layer is in editing mode.
  - on the edge of existing face(s): adds a vertex on the edge, splits the touching face(s) into triangles connected to the new vertex.



- inside a face: splits the face into triangles whose edges connect the surrounding vertices to the new vertex.

## Adding faces

To add faces to a mesh layer:

1. Press the  Digitize mesh elements button
2. A *Vertex Z value* widget appears on the top right corner of the map canvas. Set this value to the Z coordinate you would like to assign to the subsequent vertices.
3. Hover over a vertex and click the small triangle that appears next it.
4. Move the cursor to the next vertex position; you can snap to existing vertex or left-click to add a new one.
5. Proceed as above to add as many vertices you wish for the face. Press `Backspace` button to undo the last vertex.
6. While moving the mouse, a rubberband showing the shape of the face is displayed. If it is shown in green, then the expected face is valid and you can right-click to add it to the mesh. If it is red, the face is not valid (e.g. because it self-intersects, overlaps an existing face or vertex, creates a hole, ...) and can't be added. You'd need to undo some vertices and fix the geometry.
7. Press `ESC` to abort the face digitizing.
8. Right-click to validate the face.

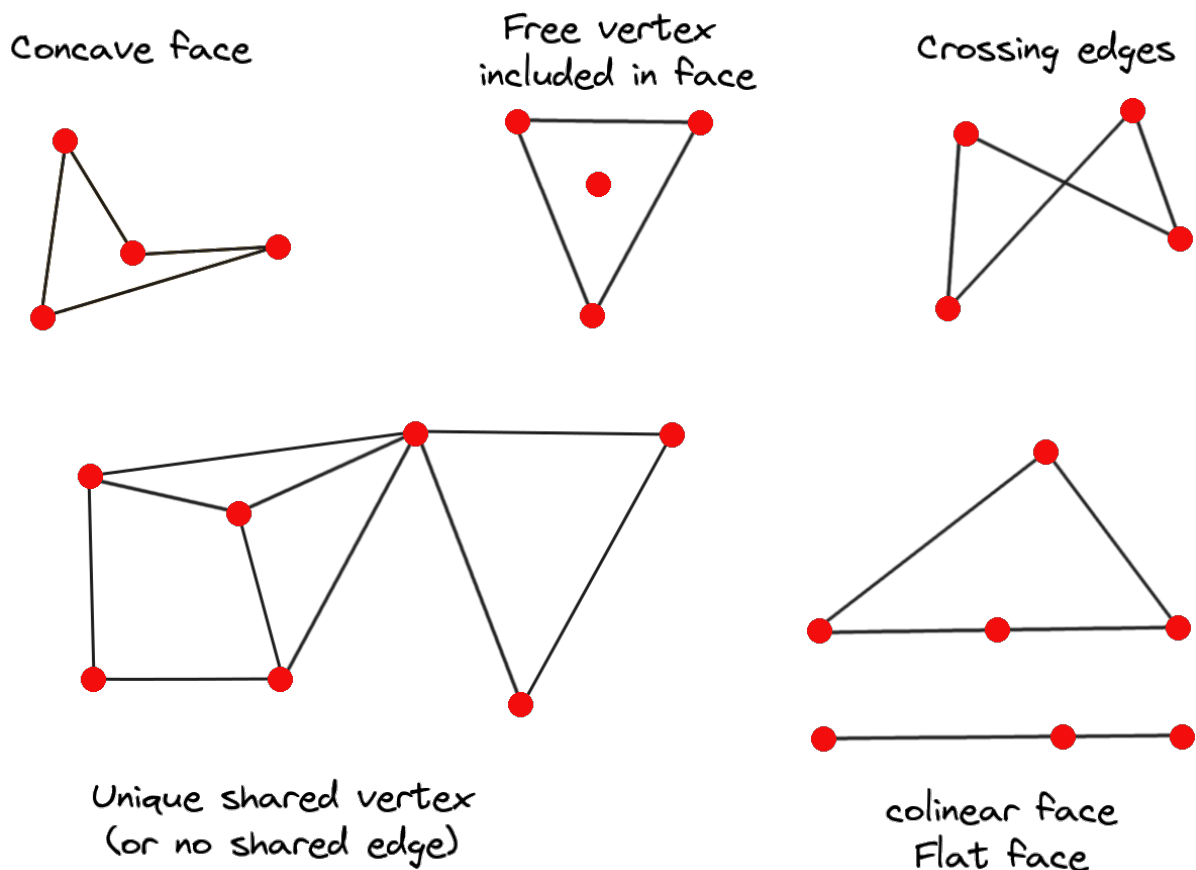



Abb. 18.14: Examples of invalid mesh


## Removing mesh elements

1. *Select the target elements*
2. Enable the  Digitize mesh elements tool
3. Right-click and select:
  - *Remove Selected Vertices and Fill Hole(s)* or press `Ctrl+Del`: removes vertices and linked faces and fills the hole(s) by triangulating from the neighbor vertices
  - *Remove Selected Vertices Without Filling Hole(s)* or press `Ctrl+Shift+Del`: removes vertices and linked faces and do not fill hole(s)
  - *Remove Selected Face(s)* or press `Shift+Del`: removes faces but keeps the vertices


These options are also accessible from the contextual menu when hovering over a single item without selecting.




## Moving mesh elements

To move vertices and faces of a mesh layer:

1. *Select the target elements*
2. Enable the  Digitize mesh elements tool
3. To start moving the element, click on a vertex or the centroid of a face/edge
4. Move the cursor to the target location (snapping to vector features is supported).
5. If the new location does not generate an *invalid mesh*, the moved elements appear in green. Click again to release them at this location. Faces whose vertices are all selected are translated, their neighbors are reshaped accordingly.


## Transforming mesh vertices

The  Transform Vertices Coordinates tool gives a more advanced way to move vertices, by editing their X, Y and/or Z coordinates thanks to expressions.


1. Select the vertices you want to edit the coordinates
2. Press  Transform Vertices Coordinates. A dialog opens with a mention of the number of selected vertices. You can still add or remove vertices from the selection.
3. Depending on the properties you want to modify, you need to check the *X coordinate*, *Y coordinate* and/or *Z value*.
4. Then enter the target position in the box, either as a numeric value or an expression (using the  Expression dialog)
5. With the  Import Coordinates of the Selected Vertex pressed, the X, Y and Z boxes are automatically filled with its coordinates whenever a single vertex is selected. A convenient and quick way to adjust vertices individually.
6. Press *Preview Transform* to simulate the vertices new location and preview the mesh with transformation.
  - If the preview is green, transformed mesh is valid and you can apply the transformation.
  - If the preview is red, the transformed mesh is invalid and you can not apply the transformation until it is corrected.
7. Press *Apply Transform* to modify the selected coordinates for the set of vertices.

## Reshaping mesh geometry

### The contextual menu


1. Enable the  Digitize mesh elements
2. Select mesh item(s), or not
3. Hover over a mesh element, it gets highlighted.
4. Right-click and you can:
  - *remove the item(s)*
  - *Split Selected Face(s) (Split Current Face)*: splits the face you are hovering over or each selected quad mesh faces into two triangles
  - *Delaunay Triangulation with Selected vertices*: builds triangular faces using selected free vertices.
  - *Refine Selected Face(s) (Refine Current Face)*: splits the face into four faces, based on vertices added at the middle of each edge (a triangle results into triangles, a quad into quads). Also triangulates adjacent faces connected to the new vertices.


### The edge markers


When the  Digitize mesh elements is active and you hover over an edge, the edge is highlighted and it is possible to interact with it. Depending on the context, following markers may be available:


- a **square**, at the center of the edge: click on it to select extremity vertices.
- a **cross** if the two faces on either side can be merged: click on it to delete the edge and merge the faces.
- a **circle** if the edge is between two triangles: Click on it to flip the edge, i.e. connect it instead to the two other „free“ vertices of the faces

### The *Force by Selected Geometries* tool

The  Force by Selected Geometries tool provides advanced ways to apply break lines using lines geometry. A break line will force the mesh to have edges along the line. Note that the break line will not be considered persistent once the operation is done; resulting edges will not act as constraints anymore and can be modified like any other edge. This can be used for example to locally modify a mesh layer with accurate lines, as river banks or border of road embankments.

1. Enable the  Force by Selected Geometries tool
2. Indicate the geometry to use as „forcing line“; it can be:
  - picked from a line or polygon feature in the map canvas: right-click over the vector feature and select it from the list in the contextual menu.
  - a virtual line drawn over the mesh frame: left-click to add vertices, right-click for validation. Vertices Z value is set through the *Vertex Z value* widget or the *z* widget if the *Advanced Digitizing Panel* is on. If the line is snapped to a mesh vertex or a 3D vector feature's vertex or segment, the new vertex takes the snapped element Z value.

Mesh faces that overlap the line geometry or the polygon's boundary will be affected in a way that depends on options you can set from the  Force by Selected Geometries tool drop-down menu:

-  *Add new vertex on intersecting edges*: with this option, a new vertex is added each time the forcing line intersect an edge. This option leads to split along the line each encountered faces.

Without this option, encountered faces are removed and replaced by faces coming from a triangulation with only the existing vertices plus the vertices of the forcing lines (new vertices are also added on the boundary edge intersecting the forcing lines).

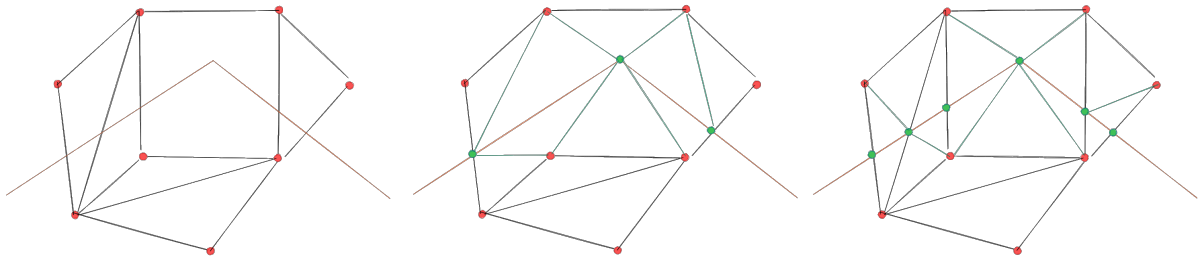


Abb. 18.15: Force Mesh using a line geometry - Results without (middle) and with (right) new vertex on edges intersection

- *Interpolate Z value from*: set how the new vertices Z value is calculated. It can be from:
  - the *Mesh* itself: the new vertices Z value is interpolated from vertices of the face they fall within
  - or the *Forcing line*: if the line is defined by a 3D vector feature or a drawn line then the new vertices Z value is derived from its geometry. In case of 2D line feature, the new vertices Z value is the *Vertex Z value*.
- *Tolerance*: when an existing mesh vertex is closer to the line than the tolerance value, do not create new vertex on the line but use the existing vertex instead. The value can be set in *Meters at Scale* or in *Map Units* (more details at [Unit Selector](#)).

## 18.4.5 Reindexing meshes

During edit, and in order to allow quick undo/redo operations, QGIS keeps empty places for deleted elements, which may lead to growing memory use and inefficient mesh structuring. The *Mesh ► <sup>182</sup>3 Reindex Faces and Vertices* tool is designed to remove these holes and renumber the indices of faces and vertices so that they are continuous and somewhat reasonably ordered. This optimizes relation between faces and vertices and increases the efficiency of calculation.

---

**Bemerkung:** The *<sup>182</sup>3 Reindex Faces and Vertices* tool saves the layer and clear the undo/redo stacks, disabling any rollback.

---

## 18.5 Mesh Calculator

The *Mesh Calculator* tool from the top *Mesh* menu allows you to perform arithmetic and logical calculations on existing dataset groups to generate a new dataset group (see [Abb. 18.16](#)).

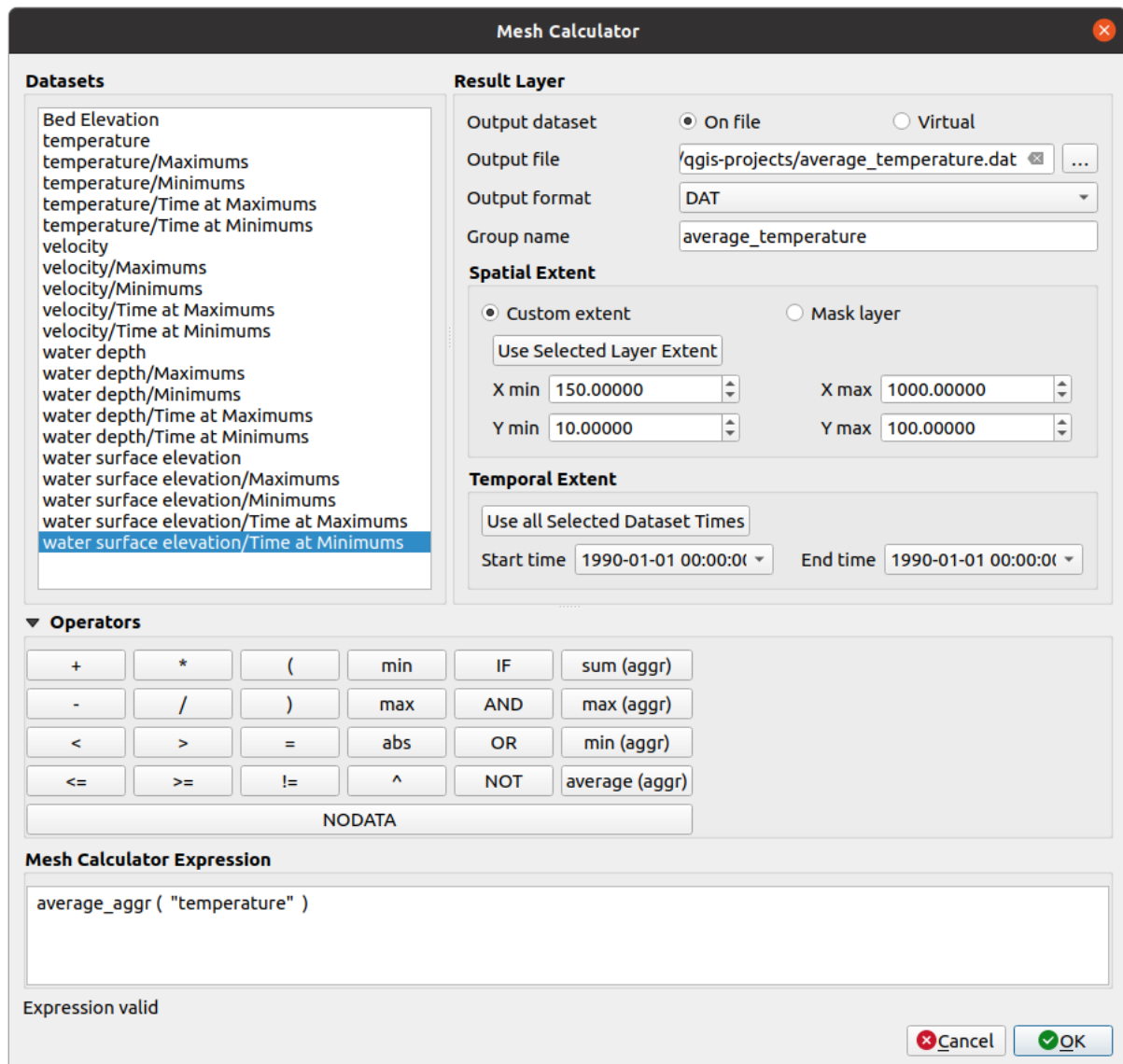



Abb. 18.16: Mesh Calculator

The *Datasets* list contains all dataset groups in the active mesh layer. To use a dataset group in an expression, double click its name in the list and it will be added to the *Mesh calculator expression* field. You can then use the operators to construct calculation expressions, or you can just type them into the box.

The *Result Layer* helps you configure properties of the output layer:

-  *Create on-the-fly dataset group instead of writing layer to disk:*
  - If unchecked, the output is stored on disk as a new plain file. An *Output File* path and an *Output Format* are required.
  - If checked, a new dataset group will be added to the mesh layer. Values of the dataset group are not stored in memory but each dataset is calculated when needed with the formula entered in the mesh calculator. That virtual dataset group is saved with the project, and if needed, it can be removed or made persistent in file from the layer *Source* properties tab.

In either case, you should provide a *Group Name* for the output dataset group.

- The *Spatial extent* to consider for calculation can be:
  - a *Custom extent*, manually filled with the *X min*, *X max*, *Y min* and *Y max* coordinate, or extracted from an existing dataset group (select it in the list and press *Use selected layer extent* to fill the abovementioned

coordinate fields)

- defined by a polygon layer (*Mask layer*) of the project: the polygon features geometry are used to clip the mesh layer datasets
- The *Temporal extent* to take into account for datasets can be set with the *Start time* and *End time* options, selected from the existing dataset groups timesteps. They can also be filled using the *Use all selected dataset times* button to take the whole range.

The *Operators* section contains all available operators. To add an operator to the mesh calculator expression box, click the appropriate button. Mathematical calculations (+, -, \*, ...) and statistical functions (min, max, sum (aggr), average (aggr), ...) are available. Conditional expressions (=, !=, <, >=, IF, AND, NOT, ...) return either 0 for false and 1 for true, and therefore can be used with other operators and functions. The NODATA value can also be used in the expressions.

The *Mesh Calculator Expression* widget shows and lets you edit the expression to execute.

---

## Working with Vector Tiles

---

### 19.1 What are Vector Tiles?

Vector tiles are packets of geographic data, packaged into pre-defined roughly-square shaped „tiles“ for transfer over the web. They combine pre-rendered raster map tiles and vector map tiles. The vector tile server returns vector map data, which has been clipped to the boundaries of each tile, instead of a pre-rendered map image. The clipped tiles represent the zoom-levels of the vector tile service, derived from a pyramid approach. Using this structure, the data-transfer is reduced in comparison to un-tiled vector maps. Only data within the current map view, and at the current zoom level need to be transferred. Also, compared to a tiled raster map, data transfer is also greatly reduced, as vector data is typically much smaller than a rendered bitmap. Vector tiles do not have any styling information assigned so QGIS needs to apply a cartographic style in order to display the data.

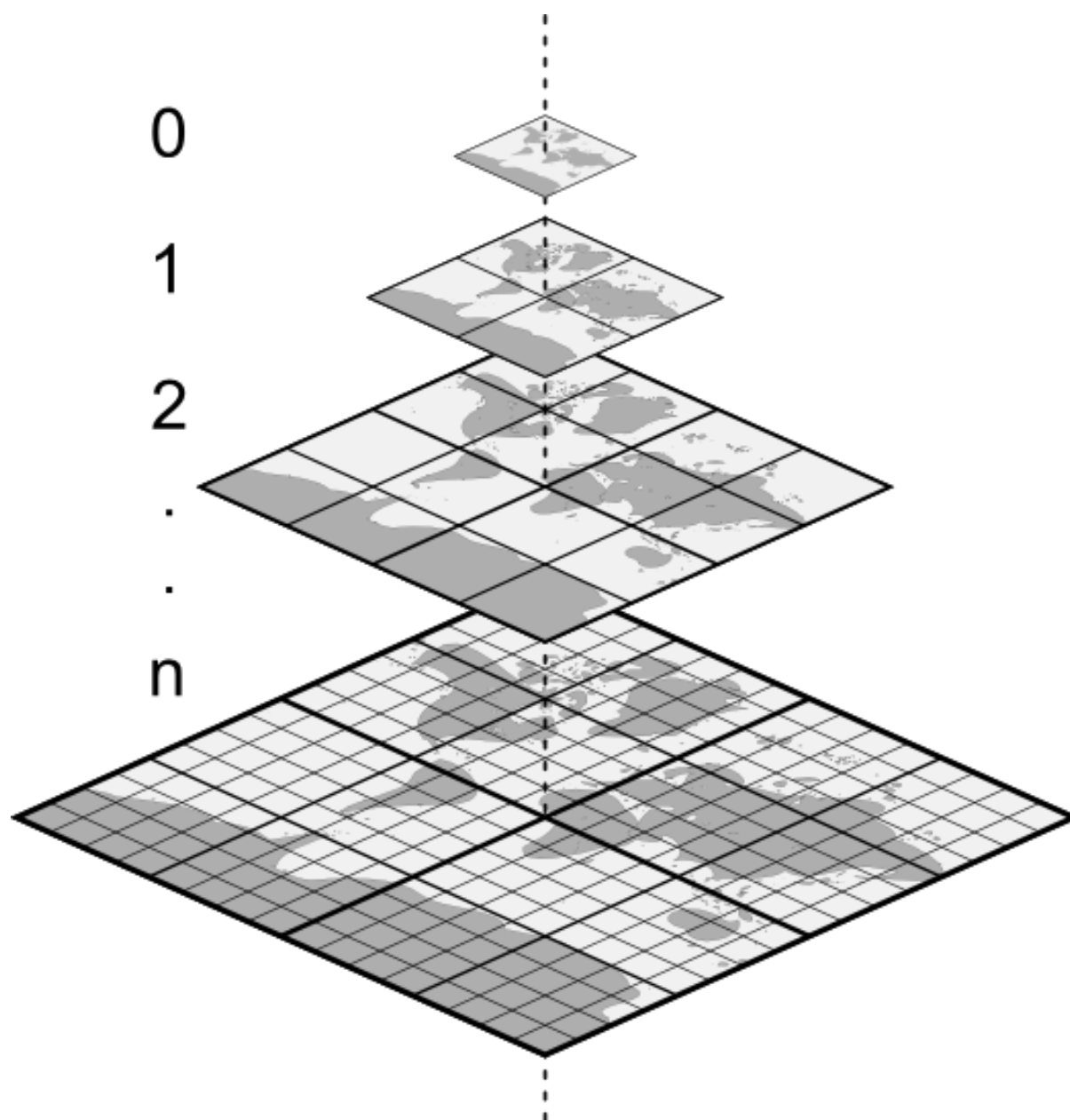



Abb. 19.1: Pyramid structure of vector tiles with zoom-levels

## 19.2 Supported Formats

There is support for vector tiles through:

- remote sources (HTTP/S) - with XYZ template - `type=xyz&url=http://example.com/{z}/{x}/{y}.pbf`
- local files - with XYZ template - e.g. `type=xyz&url=file:///path/to/tiles/{z}/{x}/{y}.pbf`
- local MBTiles database - e.g. `type=mbtiles&url=file:///path/to/file.mbtiles`

To load a vector tiles dataset into QGIS, use the  **Vector Tile** tab in the *Data Source Manager* dialog. Read [Using Vector Tiles services](#) for more details.



## 19.3 Vector Tiles Dataset Properties

The vector tiles *Layer Properties* dialog provides the following sections:




<sup>[1]</sup> Also available in the *Layer styling panel*


### 19.3.1 Information Properties

The *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata on the current layer. Provided information are:

- based on the provider of the layer: name, URI, source type and path, number of zoom levels
- the Coordinate Reference System: name, units, method, accuracy, reference (i.e. whether it's static or dynamic)
- picked from the *filled metadata*: access, extents, links, contacts, history...

### 19.3.2 Source Properties

The  *Source* tab displays basic information about the selected vector tile, including:

- the *Layer name* to display in the *Layers Panel*;
- the *Coordinate Reference System*: Displays the layer's *Coordinate Reference System (CRS)*. You can change the layer's CRS, by selecting a recently used one in the drop-down list or clicking on the  *Select CRS* button (see *Coordinate Reference System Selector*). Use this process only if the layer CRS is wrong or not specified.

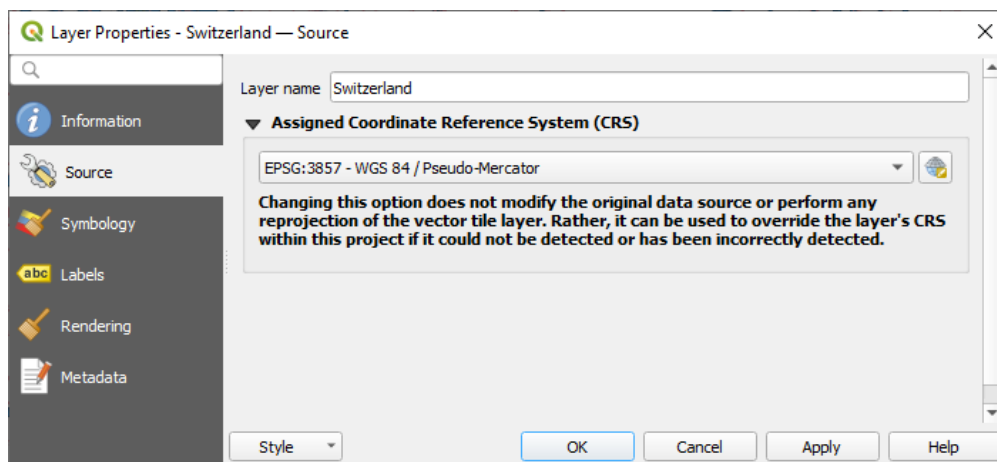


Abb. 19.2: Vector Tiles Properties - Source Dialog

## 19.3.3 Symbology and Labels Properties

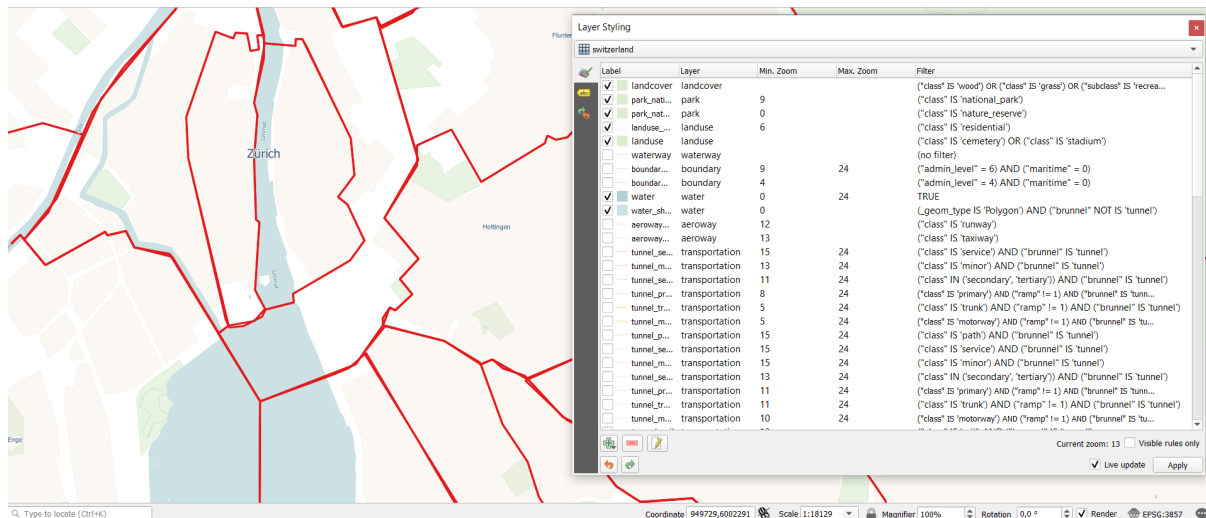



Abb. 19.3: Vector Tile Layer Symbology




### Setting rules



As vector tiles consist of point, line and polygon geometries, the respective symbols are available. To apply a cartographic style (with symbology and/or labels), you can either:

- Use a *Style URL* when creating the *Vector Tiles Connection*. The symbology will be shown immediately in the  *Symbology* tab after the layer is loaded in QGIS.
- Or build your own symbology and labeling in the corresponding tabs of the layer properties. By default, QGIS assigns an identical symbol to the features based on their geometry type.

In both cases, setting a style for a vector tile relies on a set of *rules* applied to the features, indicating:

- a vector *symbol* or *label*
- a *Label*, a title for comprehensive identification of the rule
- the name of a particular *Layer* the rule should apply to, if not applied to (*all layers*)
- a *Min. Zoom* and a *Max. Zoom*, for the range of display. symbology and labelling can be dependent on the zoom level.
- a *Filter*, a QGIS expression to identify the features to apply the style to

Each rule is added pressing the  *Add rule* button and selecting the type of symbols (*Marker*, *Line*, *Fill*) corresponding to the features geometry type. You can as well  *Remove selected rules* or  *Edit current rule*.

At the bottom the *Current Zoom* is shown. Check the  *Visible rules only* option at the top of the dialog to filter the list of rules to only those that are visible at the current zoom level. This makes it easier to work with complex vector styling and to locate troublesome rules. The  *Filter rules* text box also helps you easily find a rule, by searching the *Label*, *Layer* and *Filter* fields.

In *Abb. 19.3* we set up style for the OpenStreetMap *landuse* layer. For better visibility most of the rules are deselected.

## Layer rendering

From the *Symbology* tab, you can also set some options that invariably act on all features of the layer:

- *Opacity*: You can make the underlying layer in the map canvas visible with this tool. Use the slider to adapt the visibility of your vector layer to your needs. You can also make a precise definition of the percentage of visibility in the menu beside the slider.
- *Blending mode*: You can achieve special rendering effects with these tools that you may previously only know from graphics programs. The pixels of your overlaying and underlaying layers are mixed through the settings described in *Mischmodi*.



## Styles

Available at the bottom of most of the tabs, the *Styles* ► menu provides shortcuts to save, load, create, switch styles to apply to the vector tiles. Vector tiles can have their style saved from QGIS as *QML* files and they can be imported as:

- *QML* files (*QML - Das QGIS-Stil-Dateiformat*)
- *MapBox GL Json* style configuration files

More details at *Save and Share Layer Properties*.

### 19.3.4 Rendering Properties

Under  *Scale dependent visibility*, you can set the *Maximum (inclusive)* and *Minimum (exclusive)* scales, defining a range of scales in which features will be visible. Out of this range, they are hidden. The  Set to current canvas scale button helps you use the current map canvas scale as boundary of the range visibility. See *Visibility Scale Selector* for more information.

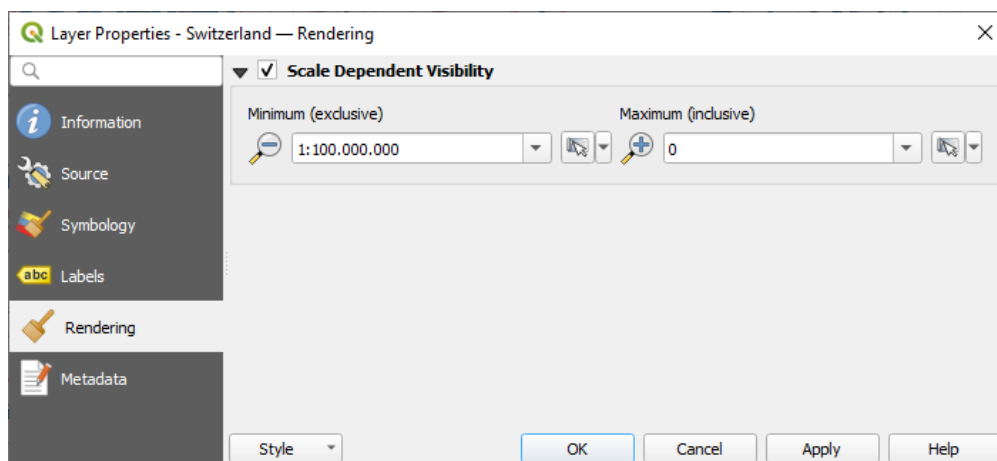



Abb. 19.4: Vector Tiles Properties - Rendering Dialog

### 19.3.5 Metadata Properties

The  *Metadata* tab provides you with options to create and edit a metadata report on your layer. See [Metadata](#) for more information.

---

## Working with Point Clouds

---

### 20.1 Introduction to Point Clouds

#### What is A Point Cloud?

A point cloud is a three-dimensional image of a space made up of many individual of data points (up to billions, even trillions). Each of the points has an x, y and z coordinate. Depending on the capture method, point clouds usually also have additional attributes that came from the capture, such as color values or intensity. These attributes can be used, for instance, to display point clouds in different colors. In QGIS, a point cloud can be used to generate a three-dimensional image of the landscape (or of another space).


#### Supported Formats

QGIS supports the data formats Entwine Point Tile (EPT) and LAS/LAZ. To work with point clouds, QGIS always saves the data in EPT. EPT is a storage format that consists of several files stored in a common folder. To allow quick access to the data, EPT uses indexing. For more information on the EPT format, see [entwine homepage](#)

If the data is in LAS or LAZ format, QGIS will convert it to EPT when it is loaded for the first time. Depending on the size of the file, this may take some time. In this process, a subfolder is created in the folder in which the LAS/LAZ file is located according to the scheme `ept_ + name_LAS/LAZ_file`. If such a subfolder already exists, QGIS loads the EPT immediately (which leads to a reduced loading time).

#### Worth Knowing

In QGIS it is not (yet) possible to edit point clouds. If you want to manipulate your point cloud, you can use [CloudCompare](#), an open source point cloud processing tool. Also the [Point Data Abstraction Library](#) (PDAL - similar to GDAL) offers you options to edit point clouds (PDAL is command line only).

Due to the large number of data points, it is not possible to display an attribute table of point clouds in QGIS. However, the  [Identify tool](#) supports point clouds, so you can display all attributes, even of a single data point.

## 20.2 Point Clouds Properties

The *Layer Properties* dialog for a point cloud layer offers general settings for the layer and its rendering. It also provides information about the layer.

Sie können den *Layereigenschaften* Dialog öffnen, indem Sie:

- In the *Layers* panel, double-click the layer or right-click and select *Properties...* from the context menu;
- in der Menüleiste *Layer ► Layereigenschaften ...* auswählen wenn der Layer aktiv ist

The point cloud *Layer Properties* dialog provides the following sections:




<sup>[1]</sup> Auch im Bedienfeld *Layergestaltung* verfügbar


---

**Bemerkung:** Most of the properties of a point cloud layer can be saved to or loaded from a `.qml` file using the *Style* menu at the bottom of the properties dialog. More details at [Save and Share Layer Properties](#)

---

### 20.2.1 Eigenschaften Information

The  *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata on the current layer. Provided information are:

- General such as name in the project, source path, last save time and size, the used provider
- Based on the provider of the layer: extent and number of points
- The Coordinate Reference System: name, units, method, accuracy, reference (i.e. whether it's static or dynamic)
- Metadata delivered by the provider: creation date, version, data format, scale X/Y/Z, ...
- Picked from the  *Metadata* tab (where they can be edited): access, extents, links, contacts, history...

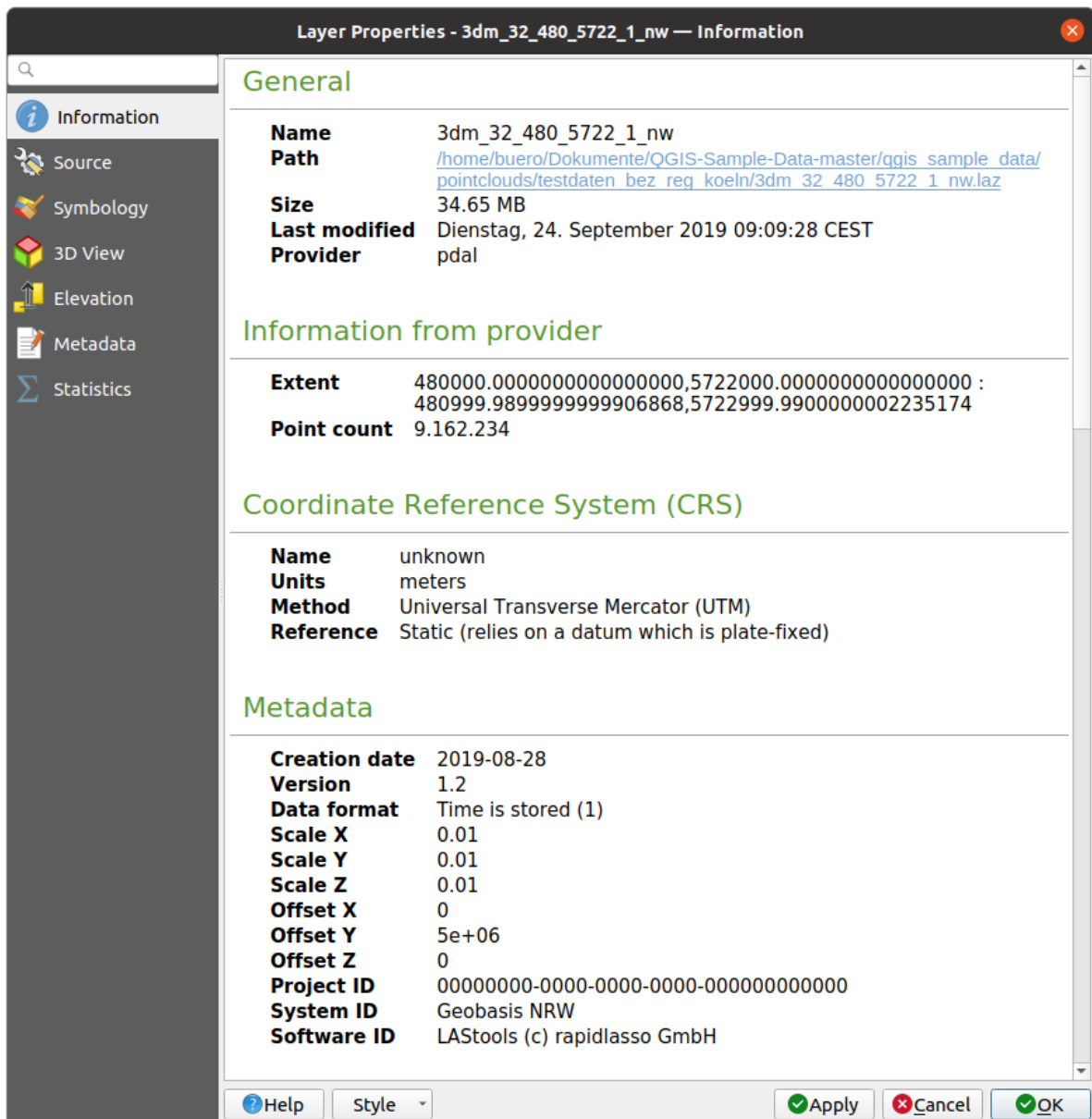



Abb. 20.1: Point cloud information tab

## 20.2.2 Eigenschaften Quelle

In the  *Source* tab you can see and edit basic information about the point cloud layer:

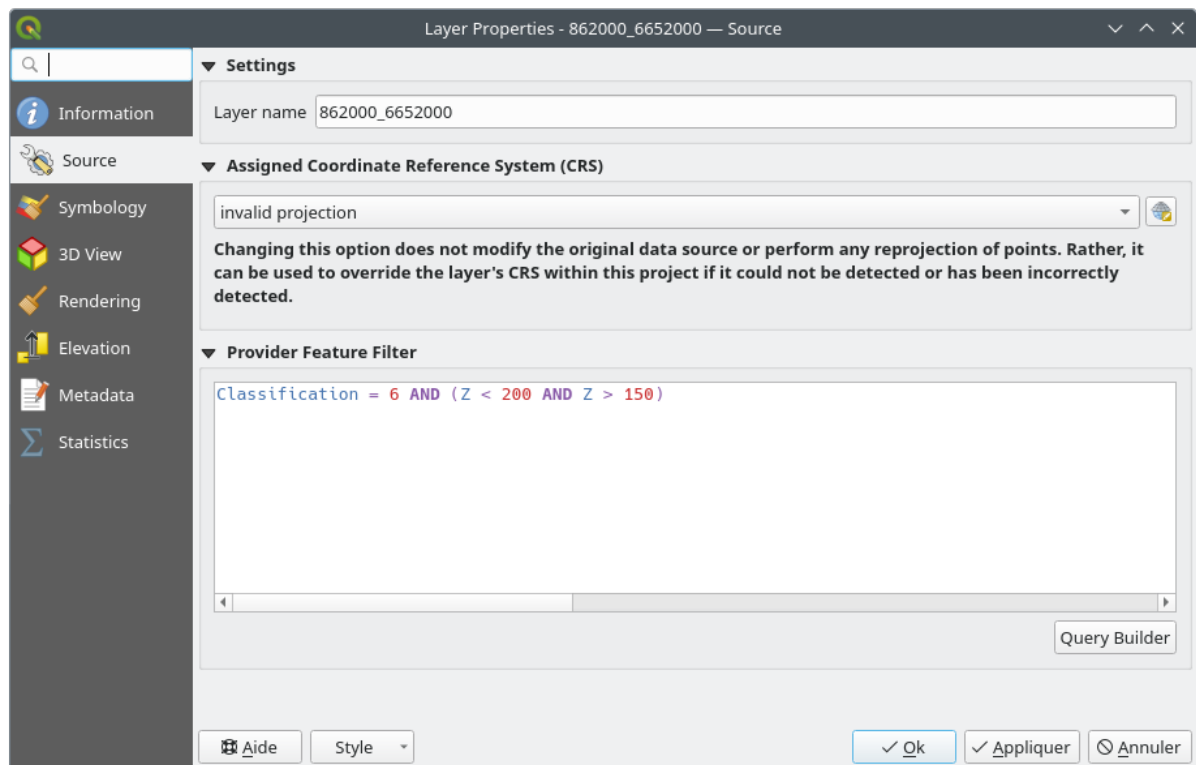



Abb. 20.2: Point cloud source tab

- *Settings*: Set a Layer name different from the layer filename that will be used to identify the layer in the project (in the Layers Panel, with expressions, in print layout legend,...)
- *Assigned Coordinate Reference System (CRS)*: Here you can change the layer's assigned *Coordinate Reference System*, selecting a recently used one in the drop-down list or clicking on  set Projection Select CRS button (see *Coordinate Reference System Selector*). Use this process only if the CRS applied to the layer is a wrong one or if none was applied.
- *Provider Feature Filter*: allows to restrict data to use in the current project to a subset, using functions and expression at the PDAL data provider level. Press the *Query Builder* button at the bottom to start setting the filter.



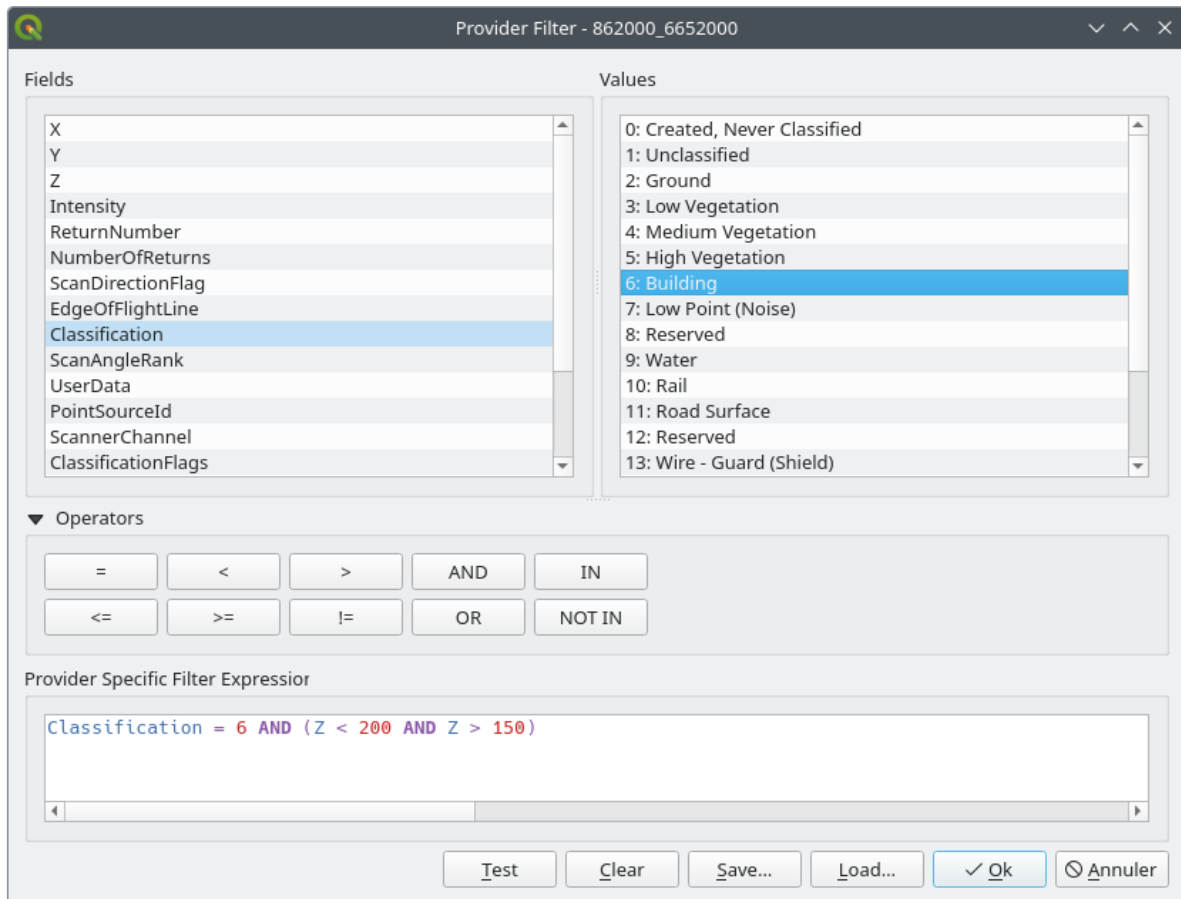


Abb. 20.3: Filtering point cloud layer to buildings at specific altitude

In the bottom part of the dialog, you can type a *Provider specific filter expression*. Such an expression can be built thanks to:


- *Fields*: the list contains all the attributes of the point cloud layer. To add an attribute to the expression, double-click its name or just type it into the text box.
- *Values*: frame lists values or statistics of the currently selected attribute, as a `key:value` pair. To add a value to the expression field, double click its name in the list: depending on the type of attribute, either the key or the value will be added to the expression. Or type the value in the expression text box.
- *Operators*: this toolbar contains all usable operators. To add an operator to the expression field, click the appropriate button. Relational operators (`=`, `>`, `<`, ...) and logical operators (`AND`, `OR`, ...) are available.

The *Test* button helps you check the syntax of your query. Use the *Clear* button to wipe the query and revert the layer to its original state (i.e., fully load all the points in the layer). It is possible to *Save...* the query as a `.QQF` file, or *Load...* the query from a file into the dialog.

When a filter is applied, QGIS treats the resulting subset as if it were the entire layer. For example if you applied the *filter above* for filtering buildings, you can not e.g. display, query, save or edit points that are of vegetation classification because they are not part of the subset.


---

**Tip:** Filtered layers are indicated in the Layers Panel

In the *Layers* panel, a filtered layer is listed with a  *Filter* icon next to it indicating the query used when the mouse hovers over the icon. Double-click the icon to open the *Query Builder* dialog for edit. This can also be achieved through the *Layer ► Filter...* menu.





---

### 20.2.3 Eigenschaften Symbolisierung

In the  *Symbology* tab the settings for the rendering of the point cloud are made. In the upper part, the settings of the different feature renderers can be found. In the lower part, there are sections with which general settings for the entire layer can be made and which apply over feature renderers.


#### Feature Rendering types

There are different options for rendering point clouds that can be selected using the drop-down menu at the top of the *Symbology* tab (see [Abb. 20.4](#)):

-  *Extent Only*: Only a bounding box of the extent of the data is displayed; convenient for overviewing the data extent. As usual, the *Symbol widget* helps you configure any properties (color, stroke, opacity, sub-layers, ...) you'd like for the box.
-  *Attribute by Ramp*: The data is drawn over a color gradient. See [Attribute by Ramp Renderer](#)
-  *RGB*: Draw the data using red, green and blue color values. See [RGB Renderer](#)
-  *Classification*: The data is drawn using different colors for different classes. See [Classification Renderer](#)

When a point cloud is loaded, QGIS follows a logic to select the best renderer:

- if the dataset contains color information (red, green, blue attributes), the RGB renderer will be used
- else if the dataset contains a `Classification` attribute, the classified renderer will be used
- else it will fall back to rendering based on Z attribute

If you do not know the attributes of the point cloud, the  *Statistics tab* provides a good overview of which attributes are contained in the point cloud and in which ranges the values are located.

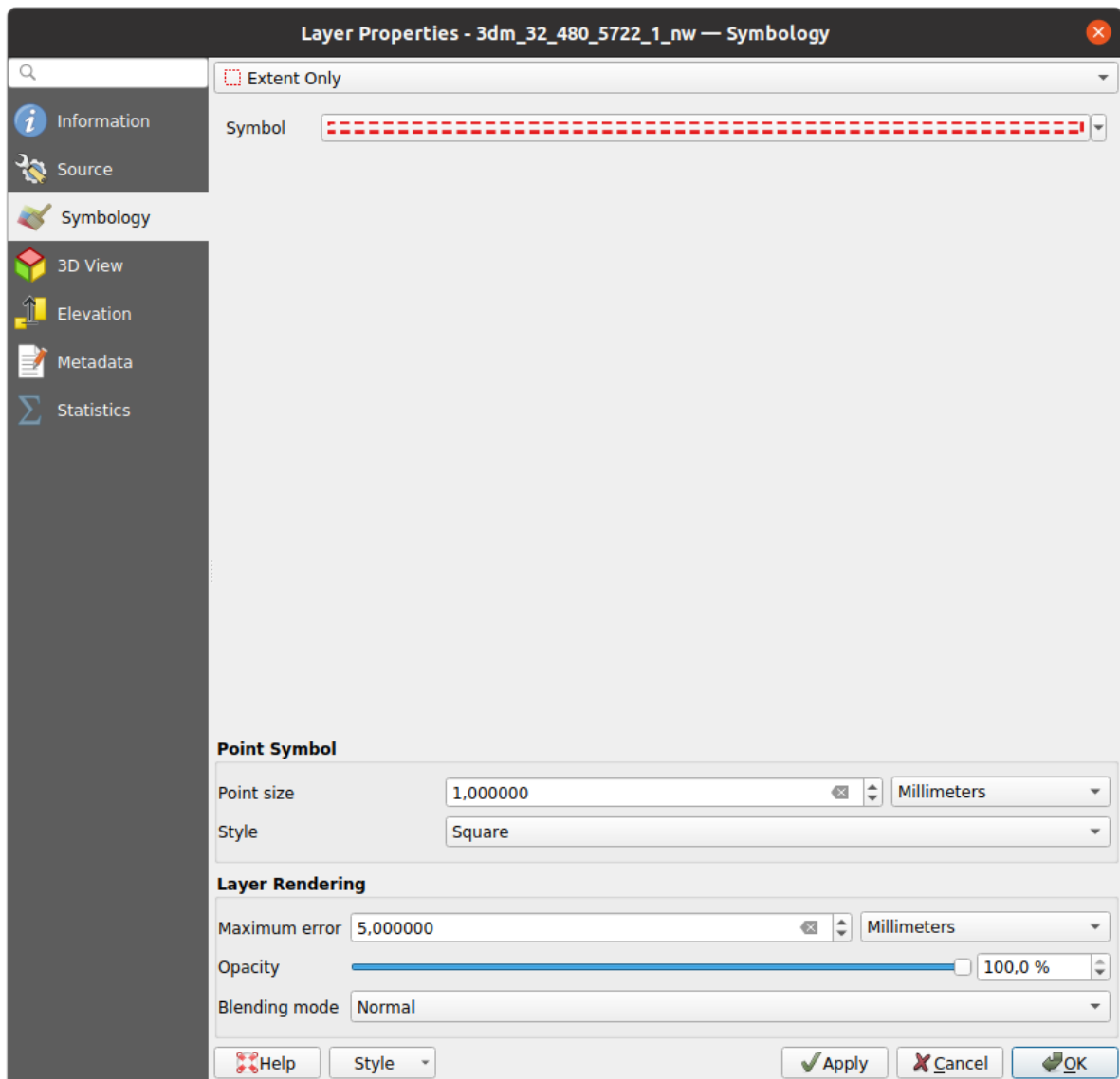



Abb. 20.4: Point cloud symbology tab

### Attribute by Ramp Renderer

With  *Attribute by Ramp*, the data can be displayed by numerical values over a color gradient. Such numerical values can be, for example, an existing intensity attribute or the Z-value. Depending on a minimum and a maximum value, the other values are spread to the color gradient via interpolation. The distinct values and their assignment to a certain color are called „color map“ and are shown in the table. There are various setting options, which are described below the figure.

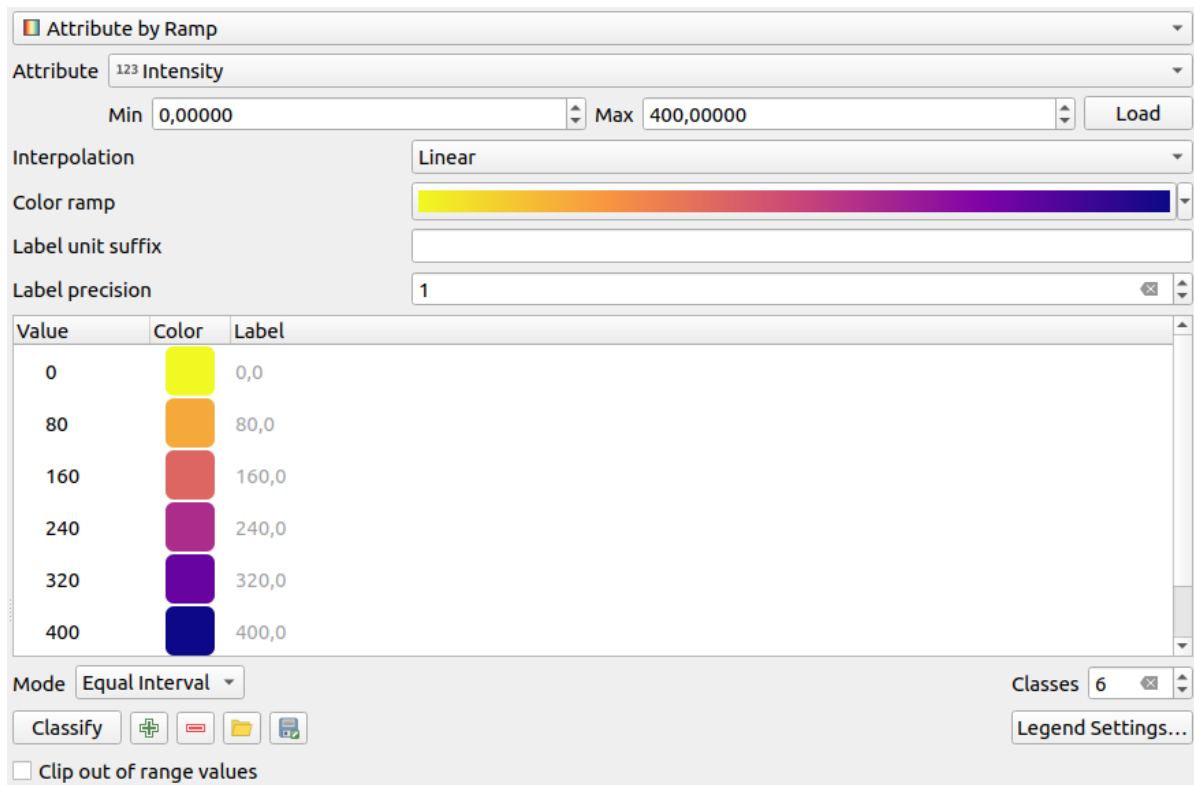


Abb. 20.5: Point cloud symbology tab: Attribute by Ramp

- *Min* and *Max* define the range that is applied to the color ramp: the *Min* value represents the left, the *Max* value the right end of the color ramp, the values in between are interpolated. By default QGIS detects the minimum and the maximum from the selected attribute but they can be modified. Once you have changed the values, you can restore the defaults by clicking on the *Load* button.
- The *Interpolation* entry defines how values are assigned their color:
  - *Discrete* (a  $\leq$  symbol appears in the header of the *Value* column): The color is taken from the closest color map entry with equal or higher value
  - *Linear* The color is linearly interpolated from the color map entries above and below the pixel value, meaning that to each dataset value corresponds a unique color
  - *Exact* (a  $=$  symbol appears in the header of the *Value* column): Only pixels with value equal to a color map entry are applied a color; others are not rendered.
- The *Color ramp* widget helps you select the color ramp to assign to the dataset. As usual with [this widget](#), you can create a new one and edit or save the currently selected one.
- The *Label unit suffix* adds a label after the value in the legend, and the *Label precision* controls the number of decimals to display.



The classification *Mode* helps you define how values are distributed across the classes:



- *Continuous*: Classes number and color are fetched from the color ramp stops; limits values are set following stops distribution in the color ramp (you can find more information on stops in [Setting a Color Ramp](#)).
- *Equal interval*: The number of classes is set by the *Classes* field at the end of the line; limits values are defined so that the classes all have the same magnitude.

The classes are determined automatically and shown in the color map table. But you can also edit these classes manually:


- Double clicking in a *Value* in the table lets you modify the class value

- Double clicking in the *Color* column opens the *Farbauswahl* widget, where you can select a color to apply for that value
- Double clicking in the *Label* column to modify the label of the class
- Right-clicking over selected rows in the color table shows a contextual menu to *Change Color...* and *Change Opacity...* for the selection


Below the table there are the options to restore the default classes with *Classify* or to manually  Add values or  Delete selected values from the table.

Since a customized color map can be very complex, there is also the option to  Load an existing color map or to  Save it for use in other layers (as a `txt` file).

If you have selected *Linear* for *Interpolation*, you can also configure:

-  *Clip out of range values* By default, the linear method assigns the first class (respectively the last class) color to values in the dataset that are lower than the set *Min* (respectively greater than the set *Max*) value. Check this setting if you do not want to render those values.
- *Legend settings*, for display in the *Layers* panel and in the *layout legend*. Customization works the same way as with a raster layer (find more details at *Customize raster legend*).

## RGB Renderer

With the  *RGB* renderer, three selected attributes from the point cloud will be used as the red, green and blue component. If the attributes are named accordingly, QGIS selects them automatically and fetches *Min* and *Max* values for each band and scales the coloring accordingly. However, it is also possible to modify the values manually.

A *Contrast enhancement* method can be applied to the values: *No Enhancement*, *Stretch to MinMax*, *Stretch and Clip to MinMax* and *Clip to MinMax*

---

**Bemerkung:** The *Contrast enhancement* tool is still under development. If you have problems with it, you should use the default setting *Stretch to MinMax*.

---

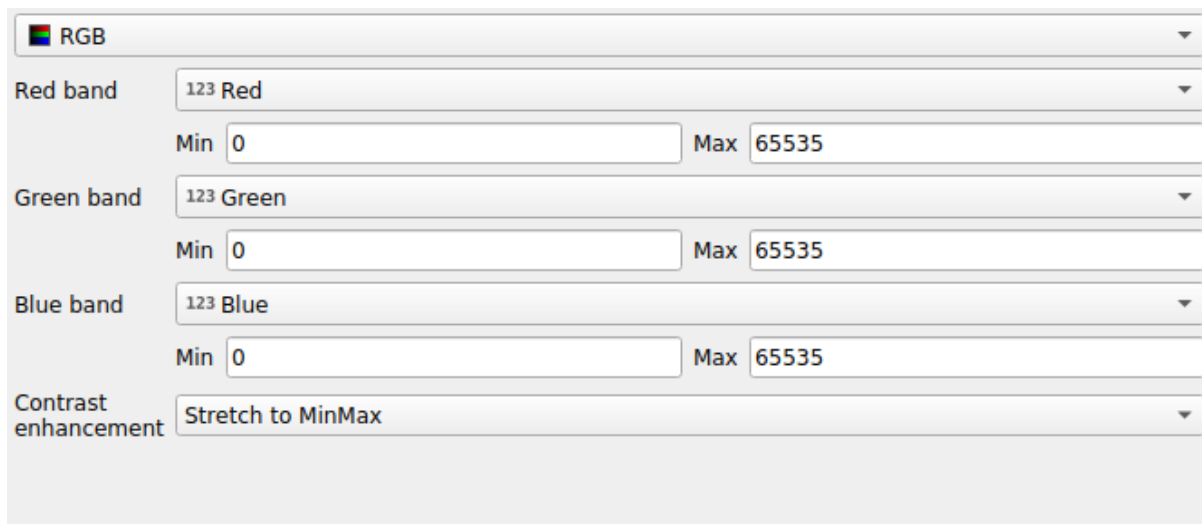



Abb. 20.6: The point cloud RGB renderer

## Classification Renderer

In the  *Classification* rendering, the point cloud is shown differentiated by color on the basis of an attribute. Any type of attribute can be used (numeric, string, ...). Point cloud data often includes a field called `Classification`. This usually contains data determined automatically by post-processing, e.g. about vegetation. With *Attribute* you can select the field from the attribute table that will be used for the classification. By default, QGIS uses the definitions of the LAS specification (see table 'ASPRS Standard Point Classes' in the PDF on [ASPRS home page](#)). However, the data may deviate from this schema; in case of doubt, you have to ask the person or institution from which you received the data for the definitions.

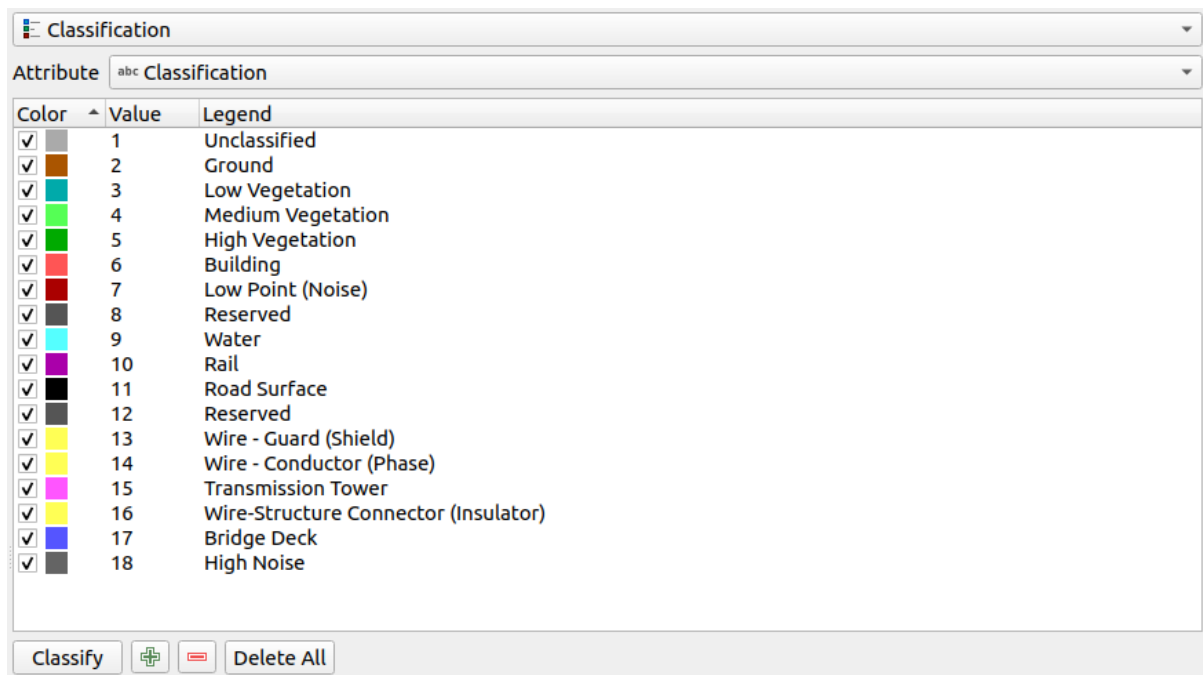





Abb. 20.7: The point cloud classification renderer

In the table all used values are displayed with the corresponding color and legend. At the beginning of each row there is a  check box; if it is unchecked, this value is no longer shown on the map. With double click in the table, the *Color*, the *Value* and the *Legend* can be modified (for the color, the *Farbauswahl* widget opens).

Below the table there are buttons with which you can change the default classes generated by QGIS:

- With the *Classify* button the data can be classified automatically: all values that occur in the attributes and are not yet present in the table are added
- With  *Add* and  *Delete*, values can be added or removed manually
- *Delete All* removes all values from the table

---

**Hinweis:** In the *Layers* panel, you can right-click over a class leaf entry of a layer to quickly configure visibility of the corresponding features.

---

## Point Symbol


Under *Point Symbol*, the size and the unit (e.g. millimeters, pixels, inches) with which each data point is displayed can be set. Either *Circle* or *Square* can be selected as the style for the points.

## Layer Rendering

In the *Layer Rendering* section you have the following options to modify the rendering of the layer:





- *Draw order*: allows to control whether point clouds rendering order on 2d map canvas should rely on their Z value. It is possible to render :
  - with the *Default* order in which the points are stored in the layer,
  - from *Bottom to top* (points with larger Z values cover lower points giving the looks of a true ortho photo),
  - or from *Top to bottom* where the scene appears as viewed from below.
- *Maximum error*: Point clouds usually contains more points than are needed for the display. By this option you set how dense or sparse the display of the point cloud will be (this can also be understood as ,maximum allowed gap between points'). If you set a large number (e.g. 5 mm), there will be visible gaps between points. Low value (e.g. 0.1 mm) could force rendering of unnecessary amount of points, making rendering slower (different units can be selected).
- *Opacity*: You can make the underlying layer in the map canvas visible with this tool. Use the slider to adapt the visibility of your layer to your needs. You can also make a precise definition of the percentage of visibility in the menu beside the slider.
- *Blending mode*: You can achieve special rendering effects with this tool. The pixels of your overlaying and underlying layers are mixed through the settings described in *Mischmodi*.
- *Eye dome lighting*: this applies shading effects to the map canvas for a better depth rendering. Rendering quality depends on the *draw order* property; the *Default* draw order may give sub-optimal results. Following parameters can be controlled:
  - *Strength*: increases the contrast, allowing for better depth perception
  - *Distance*: represents the distance of the used pixels off the center pixel and has the effect of making edges thicker.

### 20.2.4 3D-Ansicht Eigenschaften

In the  *3D View* tab you can make the settings for the rendering of the point cloud in 3D maps.

#### 3D Rendering modes

Following options can be selected from the drop down menu at the top of the tab:

- *No Rendering*: Data are not displayed
- *Follow 2D Symbology*: Syncs features rendering in 3D with *symbology assigned in 2D*
-  *Single Color*: All points are displayed in the same *color* regardless of attributes
-  *Attribute by Ramp*: Interpolates a given attribute over a color ramp and assigns to features their matching color. See *Attribute by Ramp Renderer*.
-  *RGB*: Use different attributes of the features to set the Red, Green and Blue color components to assign to them. See *RGB Renderer*.
-  *Classification*: differentiates points by color on the basis of an attribute. See *Classification Renderer*.

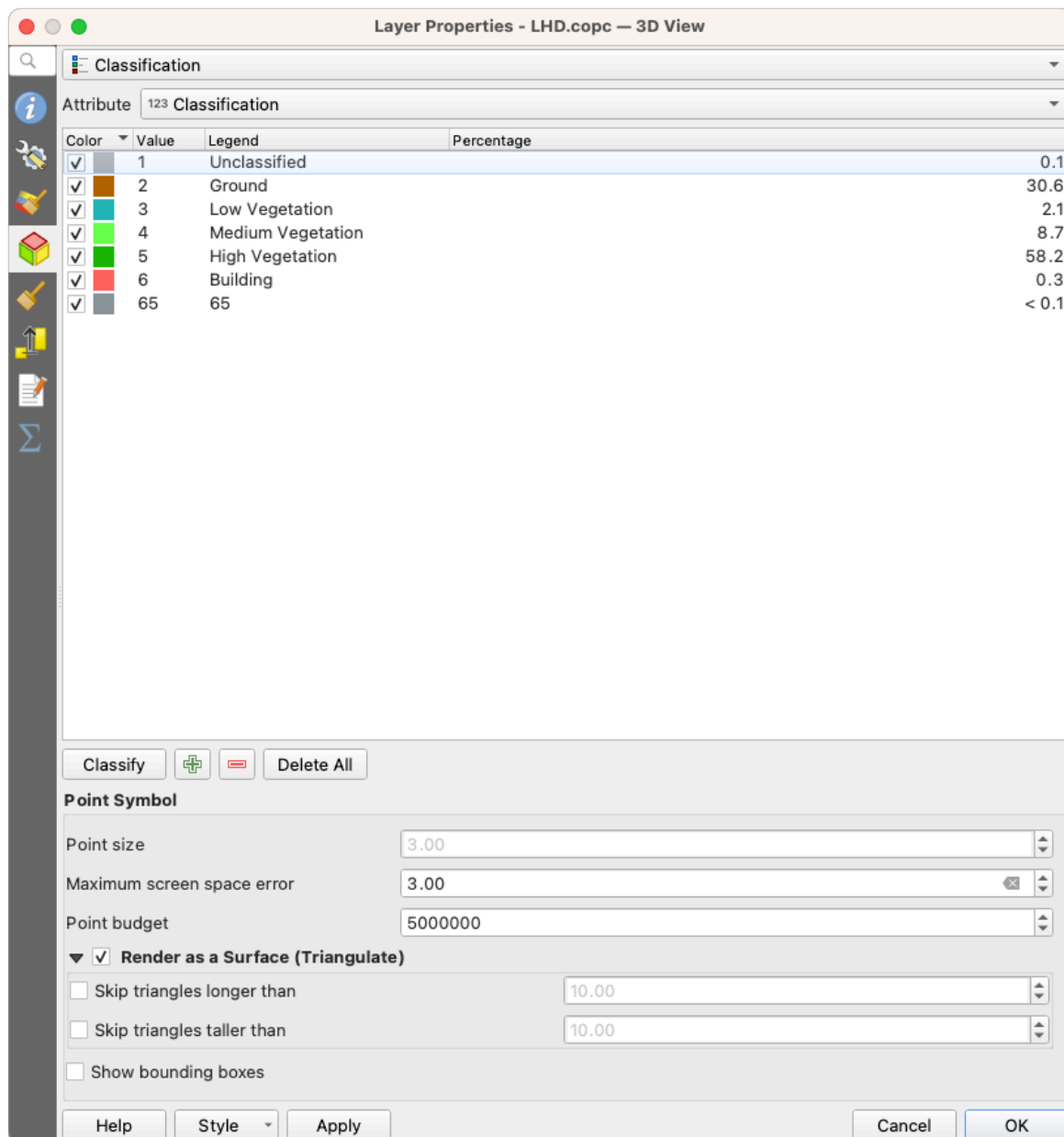







Abb. 20.8: The point cloud 3D view tab with the classification renderer

### 3D Point Symbol


In the lower part of the  *3D View* tab you can find the *Point Symbol* section. Here you can make general settings for the entire layer which are the same for all renderers. There are the following options:

- *Point size*: The size (in pixels) with which each data point is displayed can be set
- *Maximum screen space error*: By this option you set how dense or sparse the display of the point cloud will be (in pixels). If you set a large number (e.g. 10), there will be visible gaps between points; low value (e.g. 0) could force rendering of unnecessary amount of points, making rendering slower (you can find more details at [Symbolology Maximum error](#)).
- *Point budget*: To avoid long rendering, you can set the maximum number of points that will be rendered
- Check  *Render as surface (Triangulate)* to render the point cloud layer in the 3D view with a solid surface obtained by triangulation. You can control dimensions of the computed triangles:



-  *Skip triangles longer than* a threshold value: sets in the horizontal plan, the maximum length of a side of the triangles to consider
-  *Skip triangles taller than* a threshold value: sets in the vertical plan, the maximum height of a side of the triangles to consider
-  *Show bounding boxes*: Especially useful for debugging, shows bounding boxes of nodes in hierarchy

## 20.2.5 Eigenschaften Darstellung

Under the *Scale dependent visibility* group box, you can set the *Maximum (inclusive)* and *Minimum (exclusive)* scale, defining a range of scale in which features will be visible. Out of this range, they are hidden. The  button helps you use the current map canvas scale as boundary of the range visibility. See [Visibility Scale Selector](#) for more information.

**Bemerkung:** You can also activate scale dependent visibility on a layer from within the *Layers* panel: right-click on the layer and in the contextual menu, select *Set Layer Scale Visibility*.

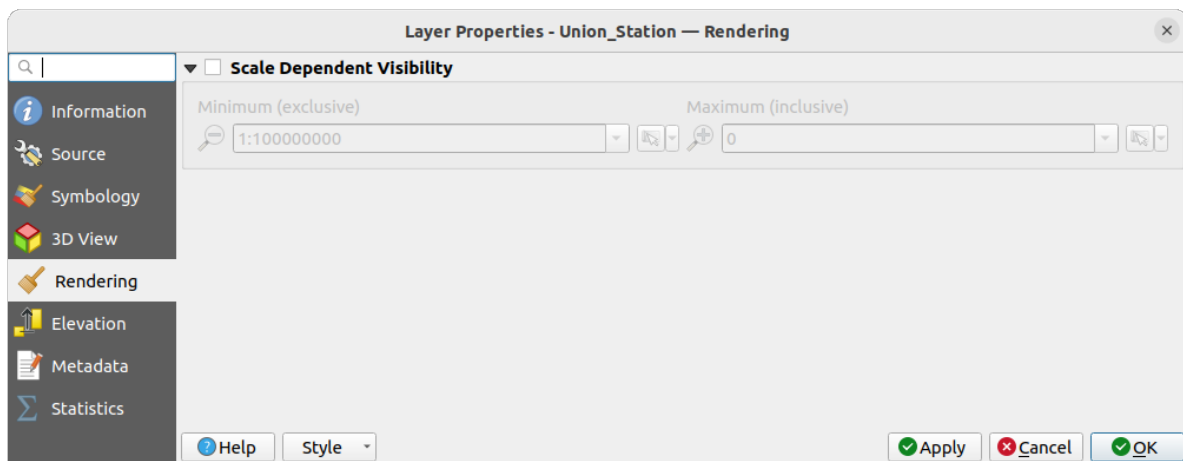




Abb. 20.9: The point cloud rendering tab

## 20.2.6 Elevation Properties

In the  *Elevation* tab, you can set corrections for the Z-values of the data. This may be necessary to adjust the elevation of the data in 3D maps and its appearance in the *profile tool charts*. There are following options:

- Under *Elevation* group:
  - You can set a *Scale*: If 10 is entered here, a point that has a value  $Z = 5$  is displayed at a height of 50.
  - An *offset* to the z-level can be entered. This is useful to match different data sources in its height to each other. By default, the lowest z-value contained in the data is used as this value. This value can also be restored with the  *Refresh* button at the end of the line.
- Under *Profile Chart Accuracy*, the *Maximum error* helps you control how dense or sparse the points will be rendered in the elevation profile. Larger values result in a faster generation with less points included.
- Under *Profile Chart Appearance*, you can control the point display:
  - *Point size*: the size to render the points with, in supported units (millimeters, map units, pixels, ...)
  - *Style*: whether to render the points as *Circle* or *Square*

- Apply a single *Color* to all the points visible in the profile view
- Check ☒ *Respect layer's coloring* to instead show the points with the color assigned via their *2D symbology*
- ☐ *Apply opacity by distance from curve effect*, reducing the opacity of points which are further from the profile curve

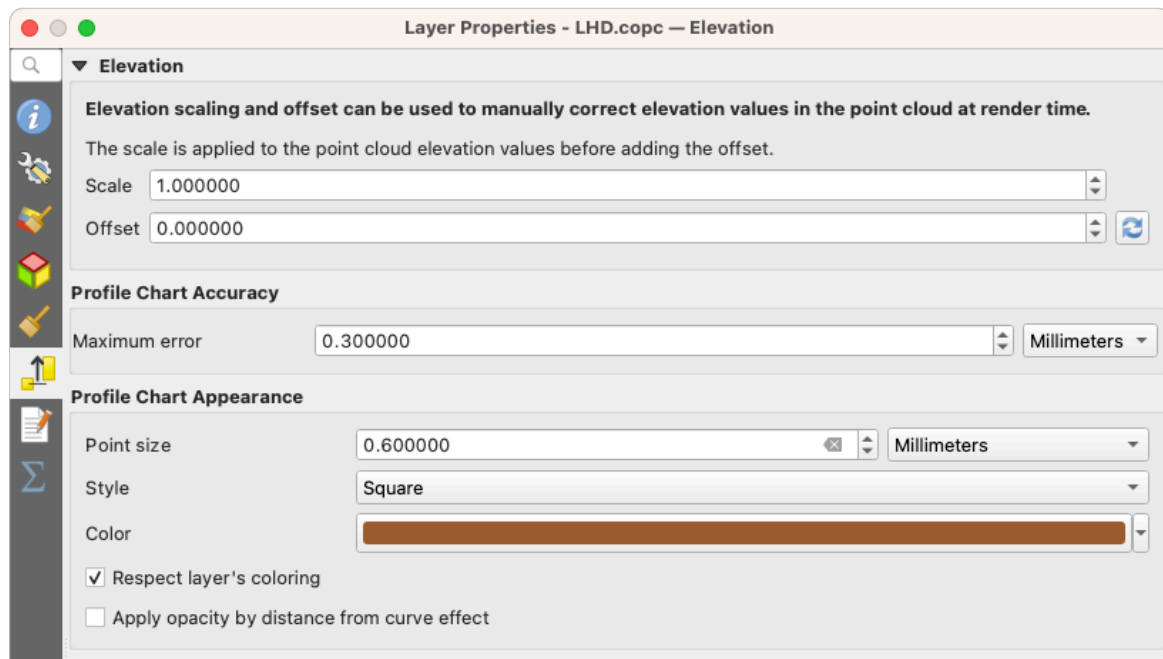




Abb. 20.10: The point cloud elevation tab

## 20.2.7 Eigenschaften Metadaten

Der Tab  *Metadaten* ermöglicht es Ihnen, Metadaten für Ihren Layer zu erstellen oder zu editieren. Siehe [Metadata](#) für weitere Informationen.

## 20.2.8 Statistics Properties

In the  *Statistics* tab you can get an overview of the attributes of your point cloud and their distribution.

At the top you will find the section *Attribute Statistics*. Here all attributes contained in the point cloud are listed, as well as some of their statistical values: *Minimum*, *Maximum*, *Mean*, *Standard Deviation*

If there is an attribute *Classification*, then there is another table in the lower section. Here all values contained in the attribute are listed, as well as their absolute *Count* and relative *% abundance*.

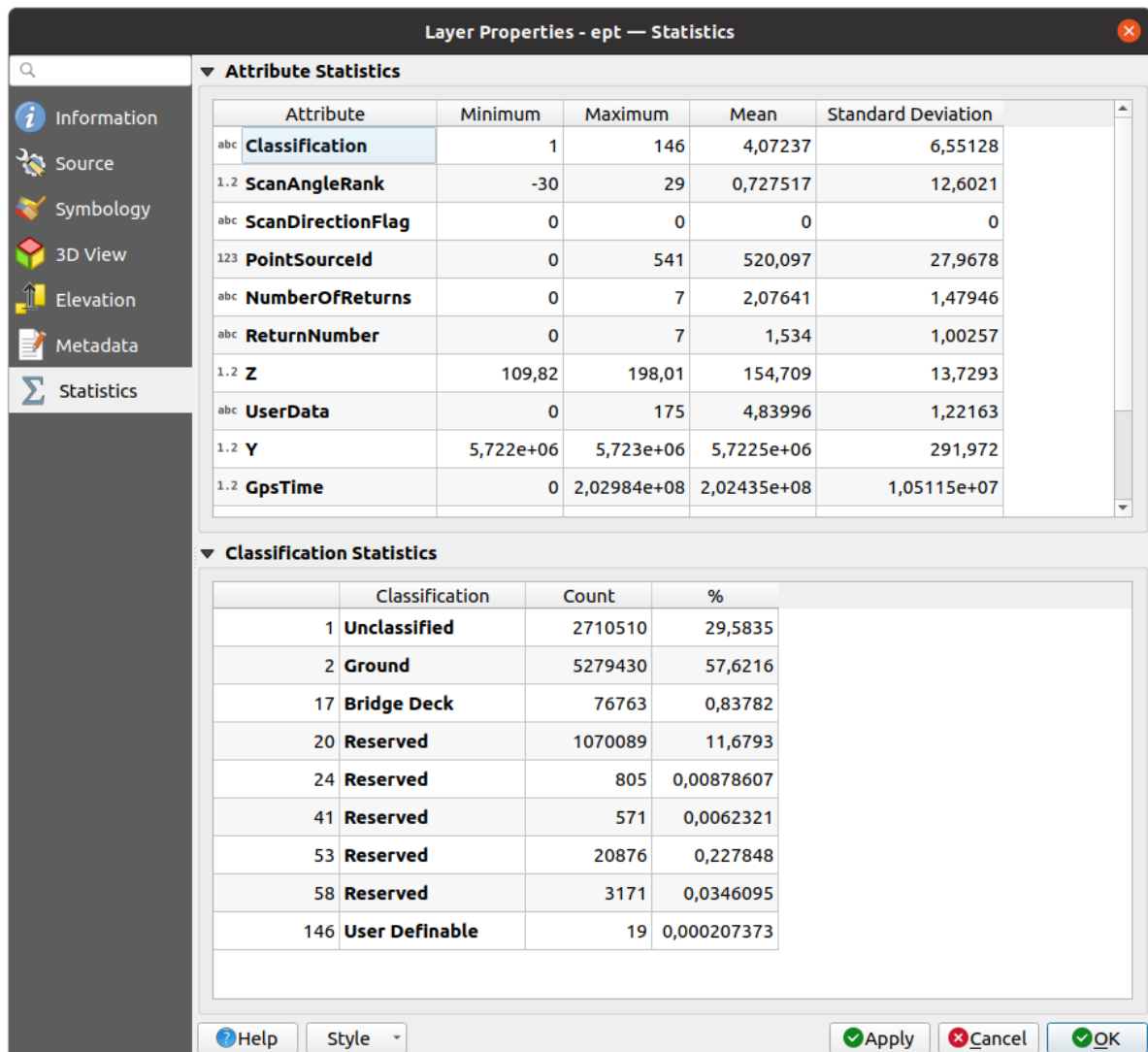


Abb. 20.11: The point cloud statistics tab

## 20.3 Virtual point cloud

Lidar surveys of larger areas are often multi-terabyte datasets with many billions of points. Having such large datasets represented as a single point cloud file is not practical due to the difficulties of storage, transfer, display and analysis. Point cloud data are therefore typically stored and distributed split into square tiles (e.g. 1km x 1km), each tile having a more manageable file size (e.g. ~200 MB when compressed).

Tiling of data solves the problems with size of data, but it introduces issues when processing or viewing an area of interest that does not fit entirely into a single tile. Users need to develop workflows that take into account multiple tiles and special care needs to be taken to deal with data near edges of tiles to avoid unwanted artefacts in outputs. Similarly, when viewing point cloud data, it becomes cumbersome to load many individual files and apply the same symbology.

Here is an example of several point cloud tiles loaded in QGIS. Each tile is styled based on min/max Z values of the tile, creating visible artefacts on tile edges. The styling has to be adjusted for each layer separately:



Abb. 20.12: Individual point cloud tiles loaded, with artefacts on edges

In the GIS world, many users are familiar with the concept of virtual rasters. A virtual raster is a file that simply references other raster files with actual data. In this way, GIS software then treats the whole dataset comprising many files as a single raster layer, making the display and analysis of all the rasters listed in the virtual file much easier.

Borrowing the concept of virtual rasters from GDAL, **virtual point cloud (VPC)** is a file format that references other point cloud files. Software supporting virtual point clouds handles the whole tiled dataset as a single data source.



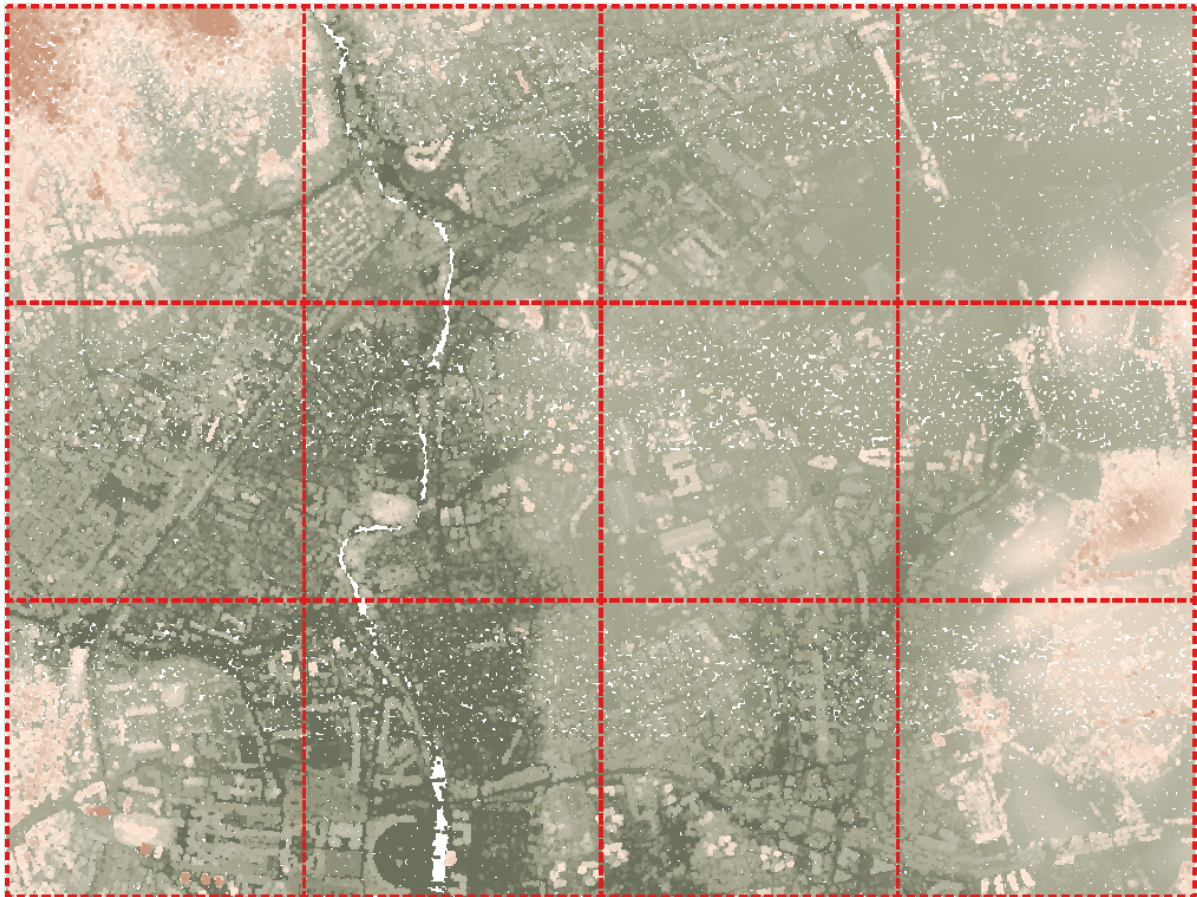


Abb. 20.13: The virtual point cloud

Displaying and manipulating virtual point cloud is much more fluent and easy.

At the core, a virtual point cloud file is a simple JSON file with `.vpc` extension, containing references to actual data files (e.g. `.LAS`, `.LAZ` or `.COPC` files) and additional metadata extracted from the files. Even though it is possible to write VPC files by hand, it is strongly recommended to create them using an automated tool, such as:

- The Processing *Build virtual point cloud (VPC)* algorithm
- The `build_vpc` command of `PDAL wrench` tool

For more details, please refer to the [VPC specification](#) that also contains best practices and optional extensions (such as overviews).



---

## Working with 3D Tiles

---

### 21.1 What are 3D Tiles?

3D tiles are specification for streaming and rendering large-scale 3D geospatial datasets. They use a hierarchical structure to efficiently manage and display 3D content, optimising performance by dynamically loading appropriate levels of detail. This technology is widely used in urban planning, architecture, simulation, gaming, and virtual reality, providing a standardised and interoperable solution for visualising complex geographical data. Read [Using 3D tiles services](#) for more details.



Abb. 21.1: Example of 3D tiles

## 21.2 Data Sources

Support for 3D tiles:

- Remote source - `http://example.com/tileset.json`
- Local files - `file:///path/to/tiles/tileset.json`

### 21.2.1 Using data from Cesium ion

*Cesium ion* is a cloud-based platform for managing and streaming 3D geospatial data. It simplifies data management, visualisation and sharing. To add 3D tiles from Cesium ion, you need to first sign up to their service here: <https://ion.cesium.com>.

You can also use Cesium ion plugin from the QGIS repository to add the data to QGIS.



## 21.2.2 Using Google 3D data

To use Google Photorealistic 3D tiles in QGIS, get an API key for 3D tiles. Once you have the key, simply add Google tiles using the same connection process.

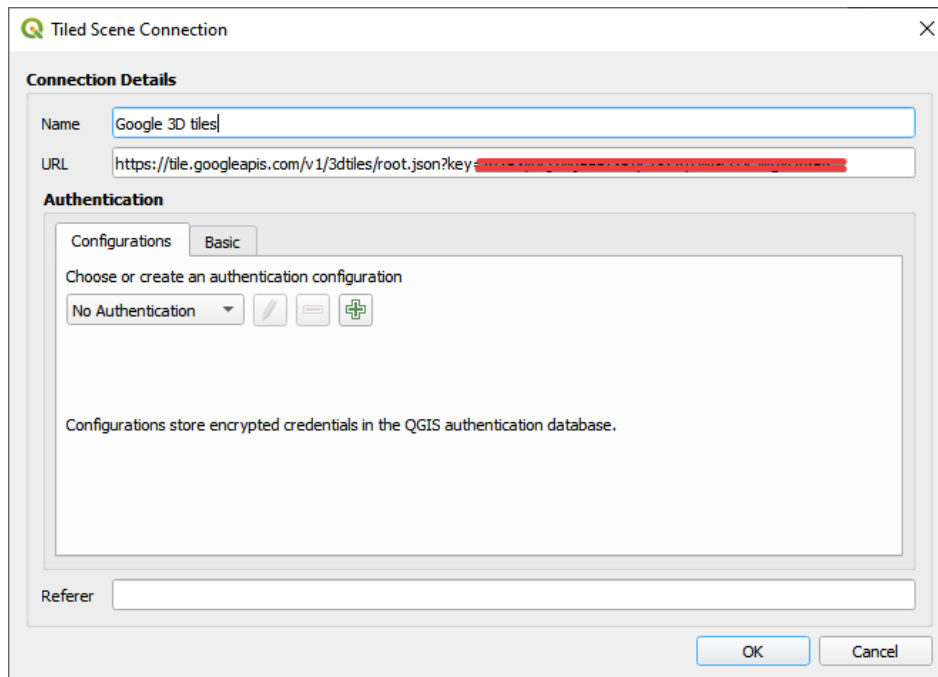


Abb. 21.2: Google 3D Tiles connection

## 21.3 3D Tiles Properties

The 3D tiles *Layer Properties* dialog provides the following sections:




<sup>[1]</sup> Also available in the *Layer styling panel*


### 21.3.1 Information Properties

The *Information* tab is read-only and represents an interesting place to quickly grab summarized information and metadata on the current layer. Provided information are:

- based on the provider of the layer: name, URL, source type and path, number of zoom levels
- the Coordinate Reference System: name, units, method, accuracy, reference (i.e. whether it's static or dynamic)
- picked from the *filled metadata*: access, extents, links, contacts, history...

## 21.3.2 Source Properties

The  *Source* tab displays basic information about the selected 3D tile, including:

- the *Layer name* to display in the *Layers Panel*;
- the *Coordinate Reference System*: Displays the layer's *Coordinate Reference System (CRS)*. You can change the layer's CRS, by selecting a recently used one in the drop-down list or clicking on the  *Select CRS* button (see *Coordinate Reference System Selector*). Use this process only if the layer CRS is wrong or not specified.

## 21.3.3 Symbology Properties

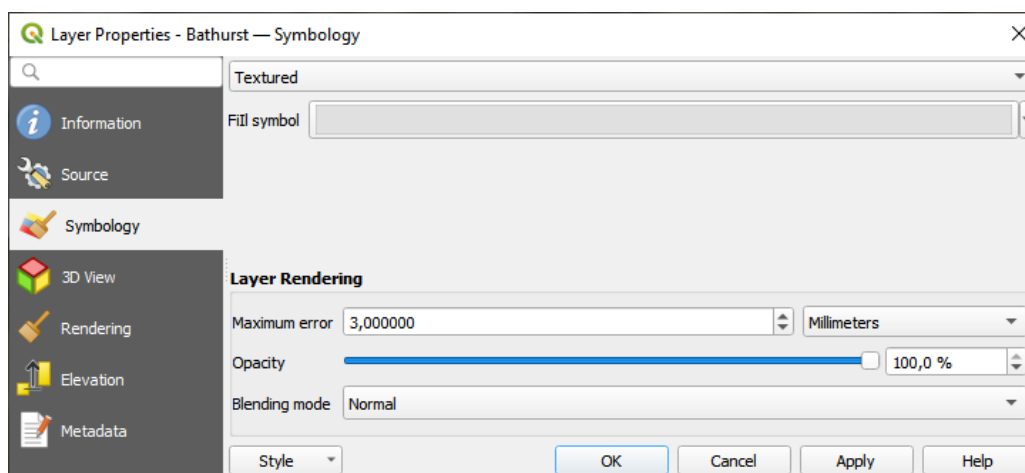


Abb. 21.3: 3D Tile Layer Symbology

By default, the layer is styled using texture, but you can change it to see the wireframe mesh behind the scene by choosing *Wireframe* in the drop-down menu. You can also, change the mesh fill and line symbols similar to the vector polygons. Checking ☒ *Use texture colors* will render each mesh element with the average value of the full texture. This is a good option to try when dealing with a large dataset and want to get a quick overview of the data.

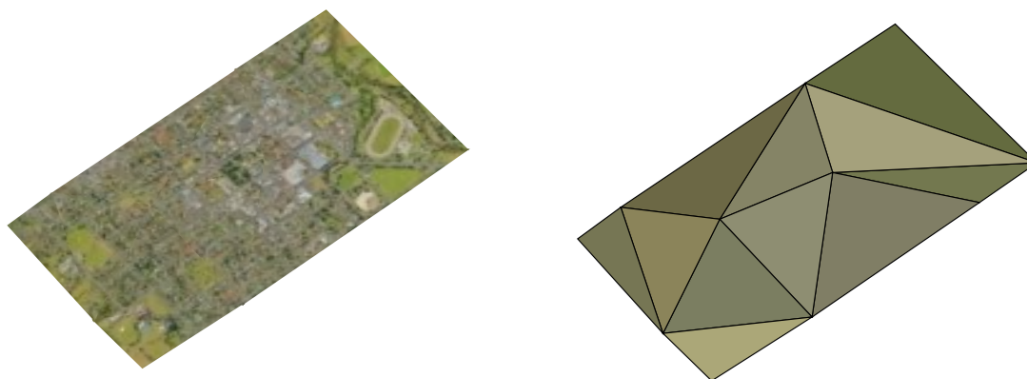



Abb. 21.4: 3D Tiles - textured and wireframe

To view the data you can open  *New 3D map view*.

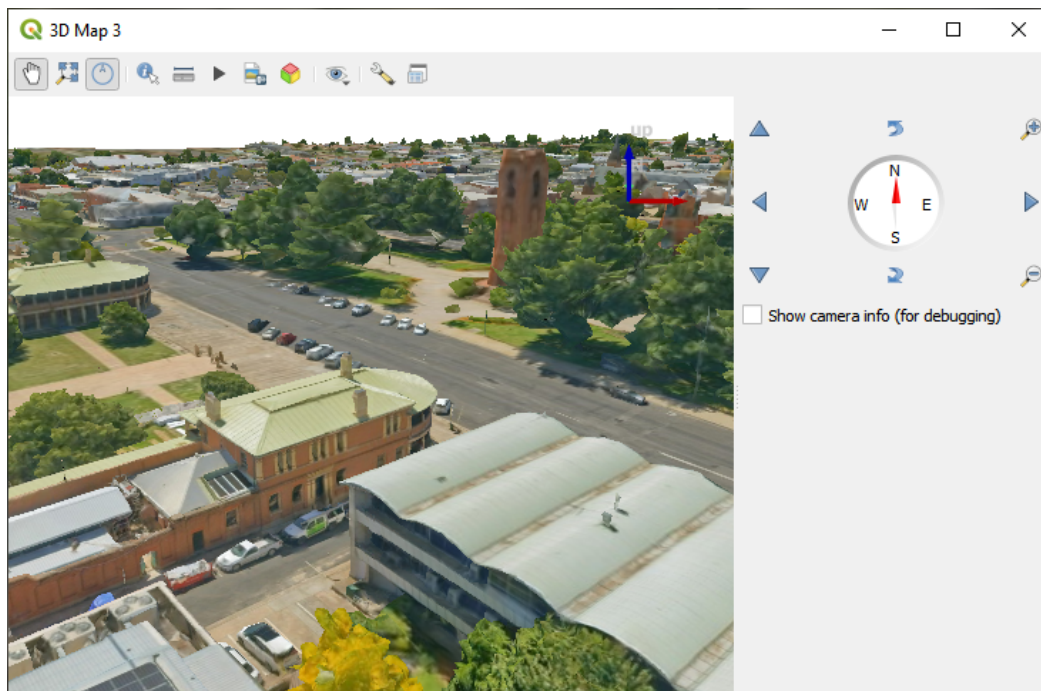


Abb. 21.5: 3D Map View



From the *Symbology* tab, you can also set some options that invariably act on all features of the layer:

- *Maximum error*: This parameter determines the level of detail displayed in the 3D model. Similar to point clouds, 3D tiles often contain more information than necessary for visual representation. By adjusting this setting, you control the balance between display density and rendering speed. A larger value (e.g., 5 mm) may introduce noticeable gaps between elements, while a smaller value (e.g., 0.1 mm) could lead to the rendering of an excessive number of details, potentially slowing down the rendering process. Different units can be selected to tailor the setting to your specific needs.
- *Opacity*: Adjust the visibility of the underlying layer on the map canvas using this tool. Use slider to tailor the visibility of your scene layer according to your preferences. Alternatively, specify the exact percentage of visibility through the menu next to the slider.
- *Blending mode*: You can achieve special rendering effects with these tools that you may previously only know from graphics programs. The pixels of your overlaying and underlying layers are mixed through the settings described in *Mischmodi*.


### 21.3.4 3D View Properties

- *Maximum screen space error* determines the threshold for swapping terrain tiles with more detailed ones (and vice versa) - i.e. how soon the 3D view will use higher quality tiles. Lower numbers mean more details in the scene at the expenses of increased rendering complexity.
- ☐ *Show bounding boxes* Show 3D bounding boxes of the terrain tiles (useful for troubleshooting terrain issues).

### 21.3.5 Rendering Properties


Under  *Scale dependent visibility*, you can set the *Maximum (inclusive)* and *Minimum (exclusive)* scales, defining a range of scales in which features will be visible. Out of this range, they are hidden. The  Set to current canvas scale button helps you use the current map canvas scale as boundary of the range visibility. See [Visibility Scale Selector](#) for more information.

### 21.3.6 Elevation Properties

The  *Elevation* tab provides options to control the layer elevation properties within a *3D map view*. Specifically, you can set:

- *Elevation Surface*: how the 3D layer vertices Z values should be interpreted as terrain elevation. You can apply a *Scale* factor and an *Offset*.

### 21.3.7 Metadata Properties



The  *Metadata* tab provides you with options to create and edit a metadata report on your layer. See [Metadata](#) for more information.

Mit Drucklayouts und Berichten können Sie Karten und Kartenserien (Atlas) erzeugen und sie entweder ausdrucken oder als Bild, PDF- oder SVG-Datei speichern.





### 22.1 Überblick über das Drucklayout


The print layout provides growing layout and printing capabilities. It allows you to add elements such as the QGIS 2D or 3D map canvas, text labels, images, legends, scale bars, basic shapes, arrows, attribute and simple tables, elevation profiles and HTML frames... You can size, group, align, position and rotate each element and adjust their properties to create your layout. The layout can be printed or exported to image formats, PostScript, PDF or to SVG. You can save the layout as a template and load it again in another session. Finally, generating several maps based on a template can be done through the atlas generator.

#### 22.1.1 Sample Session for beginners

Before you start to work with the print layout, you need to load some raster or vector layers in the QGIS map canvas and adapt their properties to suit your own convenience. After everything is rendered and symbolized to your liking, click the  New Print Layout icon in the *Project* toolbar or choose *Project* ►  *New Print Layout*. You will be prompted to choose a title for the new layout.

Um zu zeigen wie man eine Karte erstellt folgen Sie den nächsten Anweisungen.

1. On the left side, select the  Add map toolbar button and draw a rectangle on the canvas holding down the left mouse button. Inside the drawn rectangle the QGIS map view to the canvas.
2. Select the  Add scalebar toolbar button and click with the left mouse button on the print layout canvas. A scalebar will be added to the canvas.
3. Select the  Add legend toolbar button and draw a rectangle on the canvas holding down the left mouse button. Inside the drawn rectangle the legend will be drawn.
4. Wählen Sie das  Den Elementinhalt verschieben Icon um die Karte auszuwählen und verschieben Sie es ein bisschen.

5. Während das Kartenelement noch ausgewählt ist, können Sie auch seine Größe verändern. Dazu klicken Sie in eins der weißen kleinen Rechtecke in einer der Ecken des Kartenelements, halten Sie die linke Maustaste gedrückt und verschieben Sie das Rechteck an einen neuen Ort.
6. Click the *Item Properties* panel on the left down side and find the setting for the orientation. Change the value of the setting *Map orientation* to ,15.00°. You should see the orientation of the map item change.
7. Jetzt können Sie Ihr Drucklayout mit den Exportwerkzeugen im Menü *Layout* ausdrucken oder in ein Bildformat, nach PDF oder SVG exportieren.
8. Schließlich können Sie Ihr Drucklayout mit der Schaltfläche  *Projekt speichern* innerhalb der Projektdatei speichern.



You can add multiple elements to the print layout. It is also possible to have more than one map view or legend or scale bar in the print layout canvas, on one or several pages. Each element has its own properties and, in the case of the map, its own extent. If you want to remove any elements from the layout canvas, you can do that with the *Delete* or the *Backspace* key.

## 22.1.2 Layoutverwaltung

Die *Layoutverwaltung* ist das Hauptfenster, um Drucklayouts innerhalb des Projekts zu verwalten. Sie gibt Ihnen eine Übersicht über die vorhandenen Layouts und Berichte im Projekt und bietet folgende Werkzeuge:

- search for a layout;
- ein neues Drucklayout oder einen neuen Bericht auf der Basis der leeren oder einer bestimmten Vorlage erzeugen oder ein bereits vorhandenes duplizieren;
- eins davon umbenennen oder löschen;
- um sie im Projekt zu öffnen.

So öffnen Sie die Maske Layout-Verwaltung:

- Im QGIS-Hauptfenster wählen Sie im Menü *Project* ► *Layout-Verwaltung...* oder klicken Sie auf die Schaltfläche  *Layout-Verwaltung* in der *Projektwerkzeugleiste*;
- im Drucklayout oder im Berichtsfenster wählen Sie im Menü *Layout* ► *Layout-Verwaltung...* oder klicken Sie auf die Schaltfläche  *Layoutverwaltung* in der *Layout-Werkzeugleiste*.

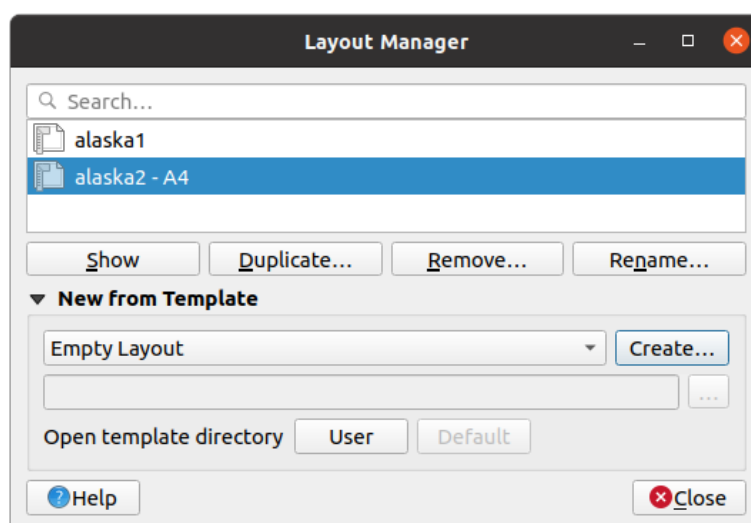


Abb. 22.1: Die Layout-Verwaltung

Die Layout-Verwaltung listet im oberen Teil alle im Projekt verfügbaren Drucklayouts oder Berichte auf und bietet folgende Werkzeuge:

- Anzeigen: Sie können mehrere Drucklayouts und/oder Berichte auswählen und sie mit einem Klick öffnen. Ein Doppelklick auf den Namen öffnet auch;
- das ausgewählte Drucklayout oder den ausgewählten Bericht duplizieren (setzt voraus, dass ein einzelner Eintrag ist ausgewählt ist): erzeugt ein neues Fenster und benutzt das ausgewählte Element als Vorlage. Sie werden dazu aufgefordert, einen neuen Titel für das Layout zu vergeben;
- das ausgewählte Drucklayout oder den ausgewählten Bericht umbenennen (setzt voraus, dass ein einzelner Eintrag ist ausgewählt ist): Sie werden dazu aufgefordert, einen neuen Titel für das Layout zu vergeben;
- das Layout entfernen: das (oder die) ausgewählte(n) Layout(s) wird (werden) aus dem Projekt gelöscht.

In the lower part, it's possible to create new print layouts or reports from scratch or a template. By default, QGIS will look for templates in the *user profile* folder and the application template directories (accessible with the two buttons at the bottom of the frame) but also in any folder declared as *Path(s) to search for extra print templates* in *Settings ► Options ► Layouts*. Found templates are listed in the combobox. Select an item and press the *Create* button to generate a new report or print layout.

Sie können auch Vorlagen aus einem eigenen Verzeichnis benutzen; in diesem Fall wählen Sie *Bestimmtes* aus dem Auswahlfeld, gehen Sie im Dateisystem zur Vorlage und drücken Sie *Erzeugen....*

---

**Tipp: So erzeugen Sie Drucklayouts im Bedienfeld Browser aus Vorlagen heraus**

Das Einfügen einer Vorlagendatei `.qpt` per drag and drop aus dem Dateimanager in das Kartenfenster oder ein Doppelklick darauf im *Bedienfeld Browser* erzeugt ein neues Drucklayout aus der Vorlagendatei.

---

### 22.1.3 Menüs, Werkzeuge und Bedienfelder des Drucklayouts

Opening the print layout provides you with a blank canvas that represents the paper surface when using the print option. Initially you find buttons on the left beside the canvas to add print layout items: 2D or 3D map canvases, text labels, images, legends, scale bars, basic shapes, arrows, attribute or simple tables, HTML frames, elevation profiles,... In this toolbar you also find buttons to navigate, zoom in on an area and pan the view on the layout as well as buttons to select any layout item and to move the extents of the map items.

Abb. 22.2 shows the initial view of the print layout before any elements are added.

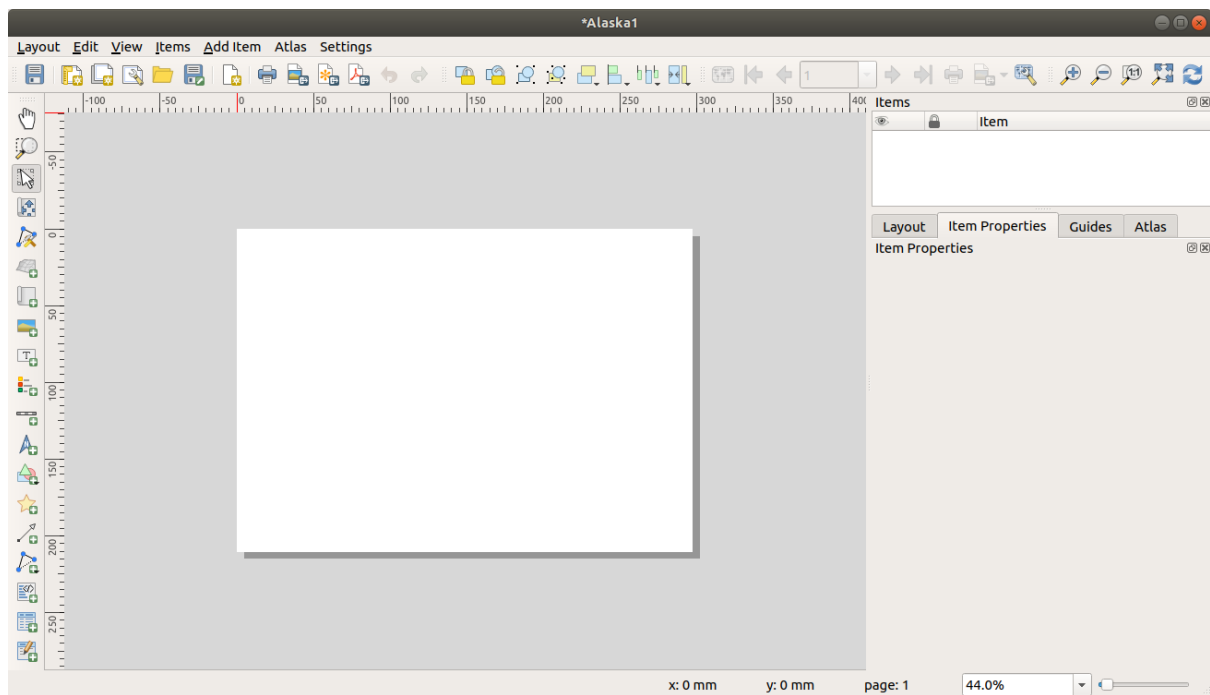



Abb. 22.2: Drucklayout

Rechts von der Zeichenfläche finden Sie zwei Bedienfelder. Das obere enthält die Reiter *Elemente* und *Rücknahmeprotokoll* und das untere Bedienfeld enthält die Reiter *Layout*, *Elementeigenschaften* und *Atlas*.

- Der Reiter *Elemente* enthält eine Liste aller Elemente, die der Zeichenfläche hinzugefügt wurden (siehe [The Items Panel](#) für weitere Informationen).
- The *Undo History* panel displays the history of all changes applied to the layout. With a mouse click, it is possible to undo and redo layout steps back and forth to a certain status.
- Im Reiter *Layout* können Sie allgemeine Parameter einstellen, die auf das Layout angewendet werden, wenn Sie es exportieren oder darin arbeiten (siehe [The Layout Panel](#) für eine genauere Beschreibung);
- Der Reiter *Elementeigenschaften* enthält die Eigenschaften für das ausgewählte Element. Klicken Sie auf die Schaltfläche  *Eintrag wählen/verschieben*, um ein Element in der Zeichenfläche auszuwählen (z.B. eine Legende, einen Maßstab oder eine Beschriftung). Klicken Sie dann den Reiter *Elementeigenschaften* an und passen Sie die Einstellungen für das ausgewählte Element an (siehe [Layout-Elemente](#) für eine genauere Beschreibung über die Einstellungen jedes Elements).
- Der Reiter *Atlas* ermöglicht es Ihnen, einen Atlas für das aktuelle Drucklayout zu erzeugen und gibt Ihnen Zugang zu seinen Einstellungen (siehe [Einen Atlas erzeugen](#) für eine genauere Beschreibung der Erzeugung eines Atlas).

Im unteren Teil des Drucklayoutfensters können Sie eine Statusleiste mit der Mausposition, der aktuellen Seitenanzahl, ein Auswahlfeld zum Einstellen der Zoomstufe, die Anzahl der gewählten Objekte (falls eine Auswahl vorhanden ist) und im Falle einer Atlaserzeugung, die Anzahl der Atlasobjekte, finden.

Im oberen Teil des Drucklayoutfensters finden Sie Menüs und Werkzeugleisten. Alle Werkzeuge des Drucklayouts sind über Menüs und Schaltflächen in der Werkzeugleiste verfügbar.





The toolbars and the panels can be switched off and on using the right mouse button over any toolbar, or through *View ► Toolbars ►* or *View ► Panels ►*.









## Menüs und Werkzeuge

### Das Menü Layout













Mit den Einträgen im Menü *Layout* können Sie das Layout verwalten:

- Speichern Sie die Projektdatei direkt aus dem Drucklayout heraus.
- Erzeugen Sie mit dem Eintrag  *Neues Layout...* ein neues leeres Drucklayout.
-  *Layout duplizieren...* : Erzeugen Sie ein neues Drucklayout als Duplikat des aktuellen.
- Löschen Sie mit dem Eintrag  *Layout löschen...* das aktuelle Layout.
- Öffnen Sie die Layout-Verwaltung mit  *Layout-Verwaltung...*
- Öffnen Sie ein vorhandenes Drucklayout mit *Layouts* ►.

Wenn Sie das Layout fertig gestaltet haben, speichern Sie seinen aktuellen Zustand mit  *Als Vorlage speichern* in eine .qpt Vorlagendatei und laden Sie seine Elemente mit  *Elemente zur Vorlage hinzufügen...* (Übersetzungshinweis: muss richtig heißen „aus Vorlage“) in einer anderen Sitzung/in ein anderes Drucklayout wieder hinein.

Im Menü *Layout* gibt es auch sehr mächtige Werkzeuge, um geographische Informationen, die mit QGIS erzeugt wurden, verfügbar zu machen, so dass sie in Berichte eingebunden oder publiziert werden können. Diese Werkzeuge sind  *Als Bild exportieren...*,  *Als PDF exportieren...*,  *Als SVG exportieren...* und  *Drucken...*

Unten ist eine Liste aller verfügbaren Werkzeuge in diesem Menü mit einigen hilfreichen Informationen.

Werkzeug	Tastenkürzel	Werkzeuggestreife	Referenz
 <i>Projekt speichern</i>	Strg+S	<i>Layout</i>	<i>Vorstellung von QGIS-Projekten</i>
 <i>Neues Layout</i>	Strg+N	<i>Layout</i>	<i>Layoutverwaltung</i>
 <i>Layout duplizieren</i>		<i>Layout</i>	<i>Layoutverwaltung</i>
 <i>Layout löschen</i>			
 <i>Layout-Verwaltung...</i>		<i>Layout</i>	<i>Layoutverwaltung</i>
<i>Layouts</i> ►			
<i>Layouteigenschaften...</i>			<i>The Layout Panel</i>
<i>Layout umbenennen...</i>			
 <i>Seiten hinzufügen...</i>		<i>Layout</i>	<i>Working with the page properties</i>
 <i>Elemente aus Vorlage hinzufügen</i>		<i>Layout</i>	<i>Creating a layout item</i>
 <i>Als Vorlage speichern...</i>		<i>Layout</i>	<i>Layoutverwaltung</i>
 <i>Als Bild exportieren...</i>		<i>Layout</i>	<i>Speichern als Rasterbild</i>
 <i>Als SVG exportieren...</i>		<i>Layout</i>	<i>Als SVG exportieren</i>
 <i>Als PDF exportieren...</i>		<i>Layout</i>	<i>Als PDF exportieren</i>
<i>Seiteneinstellungen...</i>	Strg+Umschalt+P		
 <i>Drucken...</i>	Strg+P	<i>Layout</i>	<i>Eine Ausgabe erzeugen</i>
<i>Schließen</i>	Strg+Q		















## Menü Bearbeiten

Das Menü *Bearbeiten* bietet Ihnen Werkzeuge, um Drucklayouts zu bearbeiten. Das beinhaltet gängige Aufgaben wie Auswahlwerkzeuge, Kopieren/Ausschneiden/Einfügen und die Funktionen Rückgängig/Wiederherstellen (siehe *The Undo History Panel: Revert and Restore actions*) für Elemente im Layout.

Wenn Sie die Aktion Einfügen benutzen, werden die Elemente an der aktuellen Position des Mauszeigers eingefügt. Wenn Sie den Menüpunkt *Bearbeiten ► Durch Einfügen ersetzen* (Übersetzungshinweis: muss richtig „An der Originalposition einfügen“ o.ä. heißen) oder das Tastenkürzel `Strg+Umschalt+V` drücken, werden die Elemente auf die aktuelle Seite an der Position eingefügt, an der sie auch auf ihrer Originalseite platziert waren. Damit kann sichergestellt werden, dass kopierte Elemente von Seite zu Seite an genau die selbe Stelle eingefügt werden.


Unten ist eine Liste aller verfügbaren Werkzeuge in diesem Menü mit einigen hilfreichen Informationen.

Tab. 22.1: Available Tools

Werkzeug	Tastenkürzel	Werkzeug	Referenz
 Rückgängig (letzte Änderung)	<code>Strg+Z</code>	<i>Layout</i>	<i>The Undo History Panel: Revert and Restore actions</i>
 Wiederherstellen (der letzten rückgängig gemachten Änderung)	<code>Strg+Y</code>	<i>Layout</i>	<i>The Undo History Panel: Revert and Restore actions</i>
 Löschen	<code>Entf</code>		
 Ausschneiden	<code>Strg+X</code>		
 Kopieren	<code>Strg+C</code>		
 Einfügen	<code>Strg+V</code>		
<i>Durch Einfügen ersetzen</i>	<code>Strg+Shift+V</code>		
 Alles wählen	<code>Strg+A</code>		
 Alles abwählen	<code>Strg+Umschalt</code>		
 Auswahl umkehren			
<i>Nächstes Element darunter wählen</i>	<code>Strg+Alt+[</code>		
<i>Nächstes Element darüber wählen</i>	<code>Strg+Alt+]</code>		
 Layout verschieben	<code>P</code>	<i>Werkzeugki</i>	
 Zoom	<code>Z</code>	<i>Werkzeugki</i>	
 Eintrag wählen/verschieben	<code>V</code>	<i>Werkzeugki</i>	<i>Interacting with layout items</i>
 Inhalt verschieben	<code>C</code>	<i>Werkzeugki</i>	<i>Das Kartenelement</i>
 Knotenelement bearbeiten		<i>Werkzeugki</i>	<i>Das Knotenelement</i>


## Menü Ansicht


The *View* menu gives access to navigation tools and helps to configure general behavior of the print layout. Beside the common zoom tools, you have means to:
















-  Refresh view (if you find the view in an inconsistent state);
- enable a *grid* you could snap items to when moving or creating them. Grids setting is done in *Settings ► Layout Options...* or in the *Layout Panel*;
- enable *guides* you could snap items to when moving or creating them. Guides are red lines that you can create by clicking in the ruler (above or at the left side of the layout) and drag and drop to the desired location;
- *Smart Guides*: uses other layout items as guides to dynamically snap to as you move or reshape an item;

- *Clear Guides* to remove all current guides;
- *Show Bounding box* around the items to better identify your selection;
- *Show Rules* around the layout;
- *Show Pages* or set up pages to transparent. Often layout is used to create non-print layouts, e.g. for inclusion in presentations or other documents, and it's desirable to export the composition using a totally transparent background. It's sometimes referred to as „infinite canvas“ in other editing packages.

In the print layout, you can change the zoom level using the mouse wheel or the slider and combo box in the status bar. If you need to switch to pan mode while working in the layout area, you can hold the **Spacebar** or the mouse wheel. With **Ctrl+Spacebar**, you can temporarily switch to **Zoom In** mode, and with **Ctrl+Alt+Spacebar**, to **Zoom Out** mode.

Panels and toolbars can be enabled from the **View ►** menu. To maximise the space available to interact with a composition you can check the  **View ► Toggle Panel Visibility** option or press **Ctrl+Tab**; all panels are hidden and only previously visible panels are restored when unchecked.

It's also possible to switch to a full screen mode to have more space to interact with by pressing **F11** or using **View ►  Toggle Full Screen**.

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
 Aktualisieren	F5	Navigation	
Preview ►			
 Hineinzoomen	Ctrl++	Navigation	
 Herauszoomen	Ctrl+-	Navigation	
 Zoom to 100%	Strg+1	Navigation	
 Volle Ausdehnung	Strg+0	Navigation	
Zoom to Width			
 Show Grid	Ctrl+'		Guides and Grid
 Snap to Grid	Ctrl+Shift+'		Guides and Grid
 Show Guides	Ctrl+;		Guides and Grid
 Snap to Guides	Ctrl+Shift+;		Guides and Grid
 Smart Guides	Ctrl+Alt+;		
Manage Guides...			The Guides Panel
Clear Guides			
			The Guides Panel
 Show Rulers	Ctrl+R		
 Show Bounding Boxes	Strg+Shift+B		
 Show Pages			
Werkzeugkästen ►			Bedienfelder und Werkzeugkästen
Bedienfelder ►			Bedienfelder und Werkzeugkästen
 Toggle Full Screen	F11		Ansicht
 Toggle Panel Visibility	Strg+Tab		Ansicht

## Items menu

The *Items* helps you configure items' position in the layout and the relations between them (see *Interacting with layout items*).

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
 <i>Group</i>	Ctrl+G	<i>Actions</i>	<i>Grouping items</i>
 <i>Ungroup</i>	Ctrl+Shift+G	<i>Actions</i>	<i>Grouping items</i>
 <i>Raise</i>	Ctrl+]	<i>Actions</i>	<i>Alignment and distribution</i>
 <i>Lower</i>	Ctrl+[	<i>Actions</i>	<i>Alignment and distribution</i>
 <i>Bring to Front</i>	Ctrl+Shift+]	<i>Actions</i>	<i>Alignment and distribution</i>
 <i>Send to Back</i>	Ctrl+Shift+[	<i>Actions</i>	<i>Alignment and distribution</i>
 <i>Lock Selected Items</i>	Strg+L	<i>Actions</i>	<i>Locking items</i>
 <i>Unlock All</i>	Strg+Umschalt+L	<i>Actions</i>	<i>Locking items</i>
<i>Align Items ►</i>		<i>Actions</i>	<i>Alignment and distribution</i>
<i>Distribute Items ►</i>		<i>Actions</i>	<i>Moving and resizing items</i>
<i>Resize ►</i>		<i>Actions</i>	<i>Moving and resizing items</i>

## Druck-Werkzeugkiste

Dies sind Werkzeuge zum Erstellen von Kartenelementen. Jedes dieser Werkzeuge wird im Kapitel *Layout-Elemente* ausführlich beschrieben.

Werkzeug	Werkzeugleiste	Referenz
 Karte hinzufügen	Werkzeugkiste	<i>Das Kartenelement</i>
 3D-Karte hinzufügen	Werkzeugkiste	<i>Das 3D Kartenelement</i>
 Bild hinzufügen	Werkzeugkiste	<i>Das Bildelement</i>
 Beschriftung hinzufügen Add Dynamic Text ►	Werkzeugkiste	<i>Das Beschriftungselement</i> <i>The Label Item</i>
 Legende hinzufügen	Werkzeugkiste	<i>Das Legendenelement</i>
 Maßstab hinzufügen	Werkzeugkiste	<i>Das Maßstabsselement</i>
 Nordpfeil hinzufügen	Werkzeugkiste	Das Nordpfeilelement
 Form hinzufügen	Werkzeugkiste	<i>Das Rechteckelement</i>
 ► Rechteck hinzufügen	Werkzeugkiste	<i>Das Rechteckelement</i>
 ► Ellipse hinzufügen	Werkzeugkiste	<i>Das Rechteckelement</i>
 ► Dreieck hinzufügen	Werkzeugkiste	<i>Das Rechteckelement</i>
 Markierung hinzufügen	Werkzeugkiste	<i>The Marker Item</i>
 Pfeil hinzufügen	Werkzeugkiste	Das Pfeilelement
 Knotenelement hinzufügen ►	Werkzeugkiste	<i>Das Knotenelement</i>
 ► Polygon hinzufügen	Werkzeugkiste	<i>Das Knotenelement</i>
 ► Polylinie hinzufügen	Werkzeugkiste	<i>Das Knotenelement</i>
 HTML hinzufügen	Werkzeugkiste	<i>Das HTML-Rahmen Element</i>
 Attributtabelle hinzufügen	Werkzeugkiste	<i>Das Attributtablennenelement</i>
 Feste Tabelle hinzufügen	Werkzeugkiste	<i>Das Feste Tabellenelement</i>
 Add Elevation Profile	Werkzeugkiste	<i>The Elevation Profile Item</i>

## Atlas menu

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
 Preview Atlas	Ctrl+Alt+/,	Atlas	<i>Preview and generate an atlas</i>
 First Feature	Ctrl+<	Atlas	<i>Preview and generate an atlas</i>
 Previous Feature	Ctrl+,	Atlas	<i>Preview and generate an atlas</i>
 Next Feature	Strg+.	Atlas	<i>Preview and generate an atlas</i>
 Last feature	Ctrl+>	Atlas	<i>Preview and generate an atlas</i>
 Print Atlas...		Atlas	<i>Preview and generate an atlas</i>
 Export Atlas as Images...		Atlas	<i>Preview and generate an atlas</i>
 Export Atlas as SVG...		Atlas	<i>Preview and generate an atlas</i>
 Export Atlas as PDF...		Atlas	<i>Preview and generate an atlas</i>
 Atlas Settings		Atlas	<i>Einen Atlas erzeugen</i>

### Menü Einstellungen



The *Settings ► Layout Options...* menu is a shortcut to *Settings ► Options ► Layouts* menu of QGIS main canvas. Here, you can set some options that will be used as default on any new print layout:

- *Layout defaults* let you specify the default font to use;
- With *Grid appearance*, you can set the grid style and its color. There are three types of grid: **Dots**, **Solid** lines and **Crosses**;
- *Grid and guide defaults* defines spacing, offset and tolerance of the grid (see [Guides and Grid](#) for more details);
- *Layout Paths*: to manage list of custom paths to search print templates.

The *Settings ► Keyboard Shortcuts...* menu allows you to use the [shortcuts manager](#) in the print layout interface.

### Contextual menus

Depending on where you right-click in the print layout dialog, you open a contextual menu with various features:

- Right-click on the menu bar or any toolbar and you get the list of layout panels and toolbars you can enable or disable in one-click.
- Right-click over a ruler and you can  *Show Guides*,  *Snap to Guides*, *Manage Guides...* opening the [Guides panel](#) or *Clear Guides*. It's also possible to hide the rulers.
- Right-click in the print layout canvas and:
  - You'll be able to *Undo* and *Redo* recent changes, or *Paste* any copied item (only available if no item is selected).
  - If you click over a page, you can additionally access the current [Page Properties](#) panel or *Remove Page*.
  - If you click on a selected item then you can cut or copy it as well as open the [Item Properties](#) panel.
  - If more than one item are selected, then you can either group them and/or ungroup if at least one group is already in the selection.
- Right-click inside a text box or spinbox widget of any layout panel provides edit options to manipulate its content.

## The Layout Panel

In the *Layout* panel, you can define the global settings of your print layout.

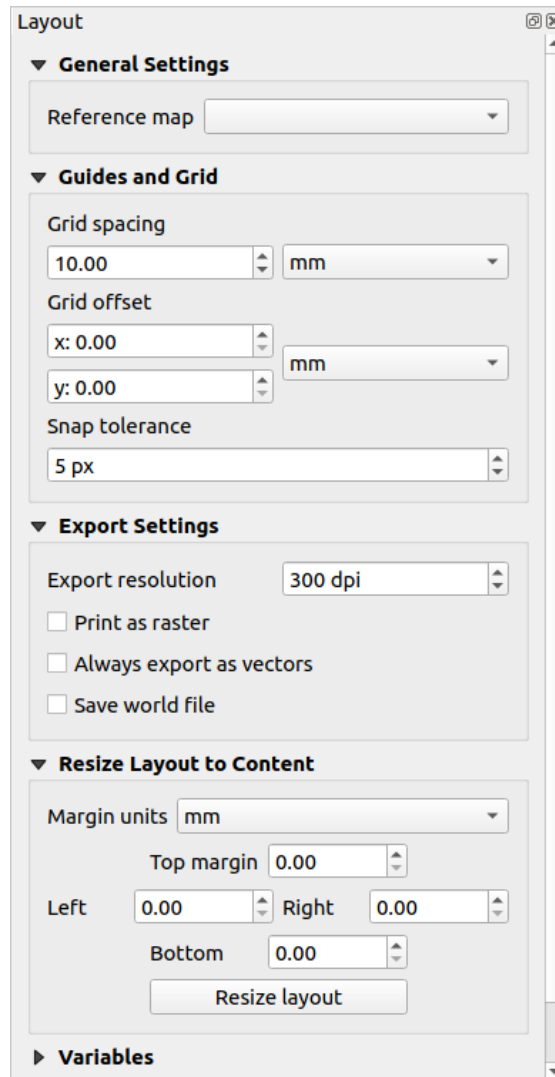


Abb. 22.3: Layout Settings in the Print Layout

## General settings

In a print layout, you can use more than one map item. The *Reference map* represents the map item to use as the layout's master map. It's assigned as long as there's a map item in the layout. The layout will use this map in any of their properties and variables calculating units or scale. This includes exporting the print layout to georeferenced formats.

Moreover, new layout items such as scale bar, legend or north arrow have by default their settings (orientation, displayed layers, scale, ...) bound to the map item they are drawn over, and fall back to the reference map if there is no overlapping map.

### Guides and Grid

You can put some reference marks on your paper sheet to help you accurately place some items. These marks can be:

- simple horizontal or vertical lines (called **Guides**) put at the position you want (see [The Guides Panel](#) for guides creation).
- or regular **Grid**: a network of horizontal and vertical lines superimposed over the layout.

Settings like *Grid spacing* or *Grid offset* can be adjusted in this group as well as the *Snap tolerance* to use for items. The tolerance is the maximum distance below which the mouse cursor is snapped to a grid or a guide, while moving, resizing or creating an item.

Whether grid or guides should be shown is set in *View* menu. There, you can also decide if they might be used to snap layout items. When both a grid line and a guide line are within tolerance of a point, guides will always take precedence - since they have been manually set (hence, assumption that they have been explicitly placed at highly desirable snapping locations, and should be selected over the general grid).

---


**Bemerkung:** In the *Settings ► Layout Options* menu, you can also set the grid and guides parameters exposed above. However, these options will only apply as defaults to new print layouts.


---

### Exporteinstellungen

You can define a resolution to use for all exported maps in *Export resolution*. This setting can then be overridden each time you export a map.

Because of some advanced rendering options (*blending mode, effects...*), a layout item may need rasterization in order to be exported correctly. QGIS will individually rasterize it without forcing every other item to also be rasterized. This allows printing or saving as PostScript or PDF to keep items as much as possible as vectors, e.g. a map item with layer opacity won't force labels, scale bars, etc to be rasterized too. You can however:

- force all the items to be rasterized checking the  *Print as raster* box;
- or use the opposite option, i.e. *Always export as vectors*, to force the export to keep items as vectors when exported to a compatible format. Note that in some cases, this could cause the output to look different to layout.

Where the format makes it possible (e.g., .TIF, .PDF) exporting a print layout results by default in a georeferenced file (based on the *Reference map* item in the *General settings* group). For other formats, georeferenced output requires you to generate a world file by checking  *Save world file*. The world file is created beside the exported map(s), has the name of the page output with the reference map item and contains information to georeference it easily.

### Resize layout to content

Using the *Resize page* tool in this group, you create a unique page composition whose extent covers the current contents of the print layout (with some optional *margins* around the cropped bounds).


Note that this behavior is different from the *crop to content* option in that all the items are placed on a real and unique page in replacement of all the existing pages.



## Variables

The *Variables* lists all the variables available at the layout's level (which includes all global and project's variables).

It also allows the user to manage layout-level variables. Click the  button to add a new custom layout-level variable.

Likewise, select a custom layout-level variable from the list and click the  button to remove it.

More information on variables usage in the [General Tools](#) section.

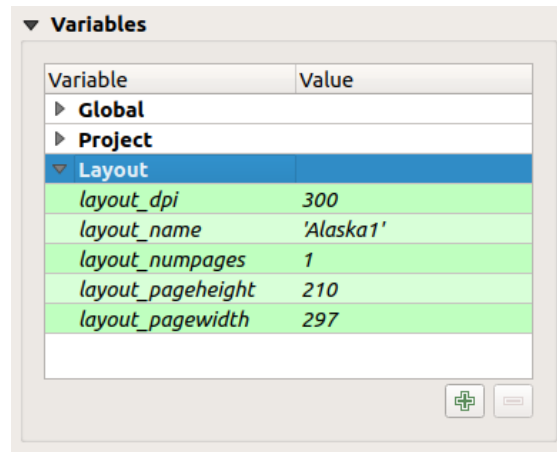



Abb. 22.4: Variables Editor in the Print Layout

## Working with the page properties

A layout can be composed of several pages. For instance, a first page can show a map canvas, and a second page can show the attribute table associated with a layer, while a third one shows an HTML frame linking to your organization website. Or you can add many types of items on each page.

### Adding a new page

Futhermore, a layout can be made using different size and/or orientation of pages. To add a page, select the  *Add Pages...* tool from the *Layout* menu or *Layout Toolbar*. The *Insert Pages* dialog opens and you are asked to fill:

- the number of pages to insert;
- the position of the page(s): before or after a given page or at the end of the print layout;
- The *Page size*: it could be of a preset format page (A4, B0, Legal, Letter, ANSI A, Arch A and their derivatives as well as a resolution type, such as 1920x1080 or 1024x768) with associated *Orientation* (Portrait or Landscape).

The page size can also be of a custom format; In that case, you'd need to enter its *Width* and *Height* (with locked size ratio if needed) and select the unit to use among mm, cm, px, pt, in, ft... Conversion of entered values is automatically applied when switching from one unit to another.

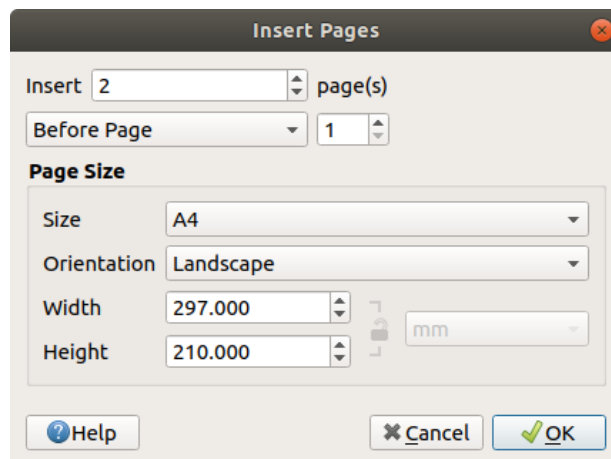


Abb. 22.5: Creating a new page in the Print Layout

### Updating page properties

Any page can be later customized through the *Page Item Properties* panel. To access a page's properties, left-click on an empty section of the page or right-click on a page and select *Page Properties...*. The *Item Properties* panel opens with settings such as:

- for layouts with several pages, it displays the active page number and the count of available pages
- the *Page size* frame described above. You can modify each property using the data defined override options (see [Explore Data-defined override buttons with atlas](#) for a use case);
- the ☐ *Exclude page from exports* to control whether the current page with its content should be included in the *layout output*;
- the *Background* of the current page using the *color* or *symbol* you want.

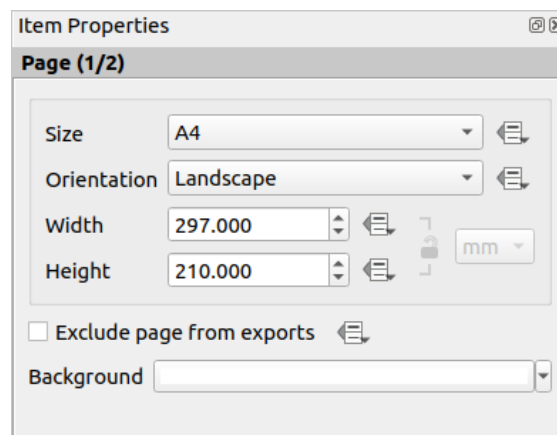


Abb. 22.6: Page properties dialog

## The Guides Panel

Guides are vertical or horizontal line references you can place on a layout page to assist you on items placement, when creating, moving or resizing them. To be active, guides require the *View ► Show Guides* and *View ► Snap to Guides* options to be checked. To create a guide, there are two different methods:

- if the *View ► Show Rulers* option is set, drag out a ruler and release the mouse button within the page area, at the desired position.
- for more precision, use the *Guides* panel from the *View ► Toolbox ►* or by selecting *Manage guides for page...* from the page's contextual menu.

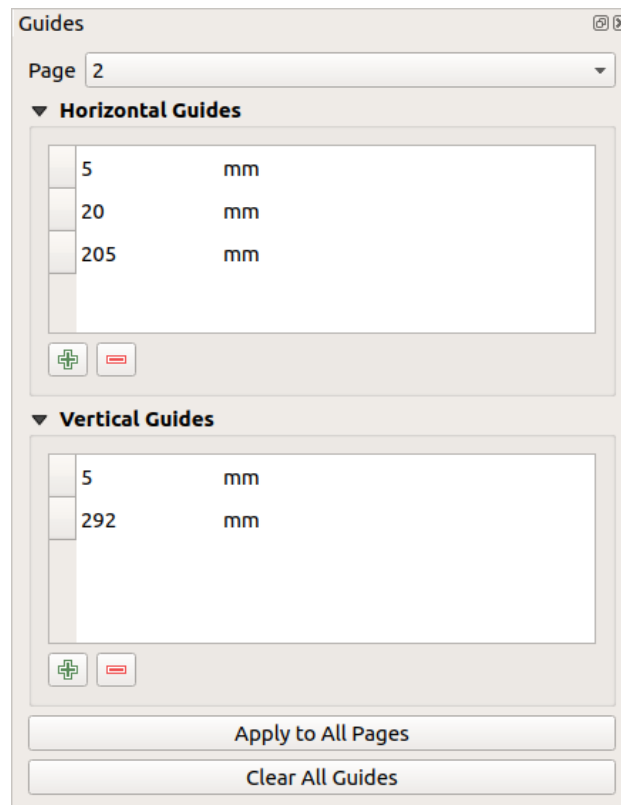




Abb. 22.7: The Guides panel

The *Guides* panel allows creation of snap lines at specific locations:

1. Select the *Page* you'd like to add the guides to
2. Click the  Add new guide button and enter the coordinates of the horizontal or vertical line. The origin is at the top left corner. Different units are available for this.  
The panel also allows adjusting the position of existing guides to exact coordinates: double-click and replace the value.
3. The *Guides* panel lists only the items for the current page. It allows creation or removal of guides only in the current page. However, you can use the *Apply to All Pages* button to replicate the guide configuration of the current page to the other pages in the layout.
4. To delete a guide, select it and press the  Remove selected guide button. Use *Clear All Guides* to remove all the guides in the current page.

---

**Tipp: Snapping to existing layout items**



Other than guides and grids, you can use existing items as snapping references when moving, resizing or creating new items; these are called **smart guides** and require *View ► Smart Guides* option to be checked. Anytime the mouse pointer is close to an item's bound, a snapping cross appears.


---

### The Items Panel

The *Items* panel offers some options to manage selection and visibility of items. All the items added to the print layout canvas (including *items group*) are shown in a list and selecting an item makes the corresponding row selected in the list as well as selecting a row does select the corresponding item in the print layout canvas. This is thus a handy way to select an item placed behind another one. Note that a selected row is shown as bold. Multiple items selection is also possible holding **Shift** for contiguous items, and **Ctrl** key for non-contiguous ones.



For any available item, you can:

-  set it visible or not;
-  lock or unlock its position;
- sort its Z position. You can move up and down each item in the list with a click and drag. The upper item in the list will be brought to the foreground in the print layout canvas. By default, a newly created item is placed in the foreground.
- change the item ID by double-clicking the text;
- right-click an item and select whether to copy or delete it or open its *properties panel*.

Once you have found the correct position for an item, you can lock it by ticking the box in  column. Locked items are **not** selectable on the canvas. Locked items can be unlocked by selecting the item in the *Items* panel and unchecking the tickbox or you can use the icons on the toolbar.

### The Undo History Panel: Revert and Restore actions

During the layout process, it is possible to revert and restore changes. This can be done with the revert and restore tools available in the *Edit* menu, the *Layout* toolbar or the contextual menu any time you right-click in the print layout area:

-  Letzte Änderung zurücknehmen
-  Letzte Änderung wiederherstellen

This can also be done by mouse click within the *Undo history* panel (see [Abb. 22.8](#)). The History panel lists the last actions done within the print layout. Select the point you want to revert to and once you do a new action all the actions done after the selected one will be removed.

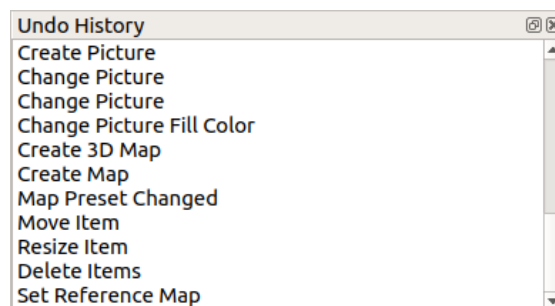


Abb. 22.8: Undo History in the Print Layout

## 22.2 Layout-Elemente

### 22.2.1 Layout Items Common Options

QGIS provides a large set of items to layout a map. They can be of map, legend, scale bar, picture, table, north arrow, image type... They however share some common options and behavior that are exposed below.

#### Creating a layout item

Items can be created using different tools, either from scratch or based on existing items.

To create a layout item from scratch:

1. Select the corresponding tool either from the *Add Item* menu or the *Toolbox* bar.
2. Then:
  - Click on the page and fill the size and placement information requested in the *New Item Properties* dialog that pops up (for details, see [Position and Size](#));

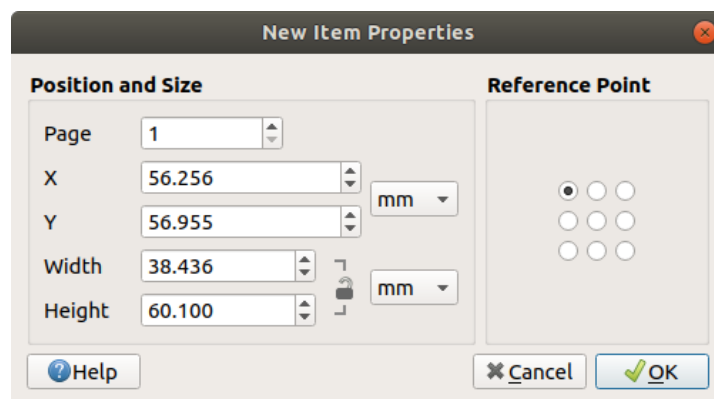



Abb. 22.9: New Item properties dialog

- Or click-and-drag to define the initial size and placement of the item. You can rely on [grids and guides](#) snapping for a better position.

**Bemerkung:** Because they can have particular shapes, drawing node or arrow items does not work with one-click nor click-and-drag methods; you need to click and place each node of the item. See [Das Knotenelement](#) for more details.

You can also:

1. Select an existing item with the  Select/Move item button from the *Toolbox* toolbar
2. Use the contextual menu or the *Edit* menu tools to copy/cut the item and paste it at the mouse position as a new item.

You can also use the *Paste in Place* (Ctrl+Shift+V) command to duplicate an item from one page to another and place it in the new page at the same coordinates as the original.


Moreover, you can create items using a print layout template (for details, see [Layoutverwaltung](#)) through the *Layout* ► *Add Items from Template...* command.


### Tip: Add layout items using the file browser

From your file browser or using the *Browser* panel, drag-and-drop a print layout template (.qpt file) onto a print layout dialog and QGIS automatically adds all items from that template to the layout.

---

## Interacting with layout items

Each item inside the print layout can be moved and resized to create a perfect layout. For both operations the first step is to activate the  Select/Move item tool and click on the item.

You can select multiple items with the  Select/Move item button: click and drag over the items or hold the *Shift* button and click on each of the items you want. To deselect an item, click on it holding the *Shift* button.






Each time there's a selection, count of selected items is displayed on the status bar. Inside the *Edit* menu, you can find actions to select all the items, clear all selections, invert the current selection and more...

## Moving and resizing items

Unless *View ► Show Bounding Boxes* option is unchecked, a selected item will show squares on its boundaries; moving one of them with the mouse will resize the item in the corresponding direction. While resizing, holding *Shift* will maintain the aspect ratio. Holding *Alt* will resize from the item center.

To move a layout item, select it with the mouse and move while holding the left button. If you need to constrain the movements to the horizontal or vertical axis, hold the *Shift* button on the keyboard while moving the mouse. You can also move a selected item using the *Arrow* keys on the keyboard; if the movement is too slow, you can speed it up by holding *Shift*. If you need better precision, use the *Position and size* properties, or grid/guides snapping as explained above for item's creation.

Resizing or moving several items at once is made the same way as for a single item. QGIS however provides some advanced tools to automatically resize a selection of items following different rules:


- each item height matches the  tallest or the  shortest selected item;
- each item width matches the  widest or the  narrowest selected item;
- resizes items to  squares: each item is enlarged to shape a square.

Likewise, there are *tools* available to organize multiple items position by distributing equidistantly:


- edges (left, right, top or bottom) of items;
- centers of items horizontally or vertically;
- spacing between items horizontally or vertically.

## Grouping items

Grouping items allows you to manipulate a set of items like a single one: you can easily resize, move, delete, copy the items as a whole.


To create a group of items, select more than one and press the  *Group* button on the *Items* menu or the *Actions* toolbar, or from the right-click menu. A row named *Group* is added to the *Items* panel and can be locked or hidden like any other *Items panel's object*. Grouped items are **not individually** selectable on the canvas; use the *Items* panel for direct selection and access the item's properties panel.

## Locking items

Once you have found the correct position for an item, you can lock it by using the  *Lock selected items* button in the *Items* menu or the *Actions* toolbar, or ticking the box next to the item in the *Items* panel. Locked items are **not** selectable on the canvas.

Locked items can be unlocked by selecting the item in the *Items* panel and unchecking the tickbox or you can use the icons on the toolbar.

## Alignment and distribution

Tools for raising or lowering the Z position of items in the layout are inside the  *Raise selected items* pull-down menu. Choose an element on the print layout canvas and select the matching functionality to raise or lower the selected element over the other elements. This order is shown in the *Items* panel. You can also raise or lower objects in the *Items* panel by clicking and dragging an object's label in this list.

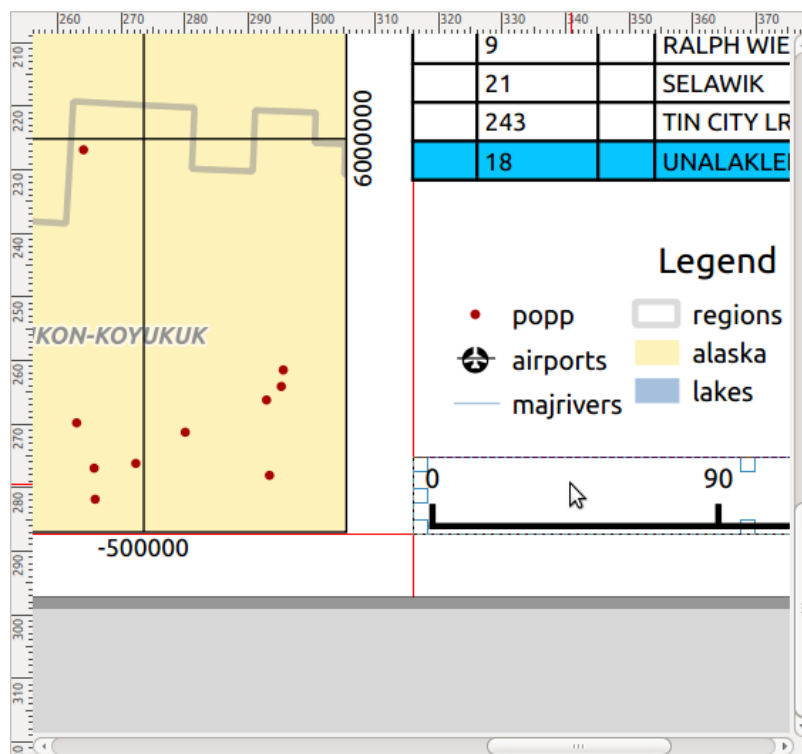


















Abb. 22.10: Alignment helper lines in the print layout

There are several alignment options available within the  *Align selected items* pull-down menu (see Abb. 22.10). To use an alignment function, you first select the elements and then click on one of the alignment icons:

-  *Align Left* or  *Align Right*;
-  *Align Top* or  *Align Bottom*;
-  *Align Center horizontally* or  *Align Center Vertical*.

All selected elements will then be aligned to their common bounding box. When moving items on the layout canvas, alignment helper lines appear when borders, centers or corners are aligned.

Another way to improve layout items placement is by adjusting the spacing between them, over the layout page. This can be done by selecting the items and press the  *Distribute Left Edges* drop-down menu to:

-  *Distribute Left Edges* or  *Distribute Right Edges* of items equidistantly
-  *Distribute Top edges* or  *Distribute Bottom Edges* of items equidistantly
-  *Distribute Horizontal Centers* or  *Distribute Vertical Centers* of items equidistantly
- Add equal space between items:  *Distribute Horizontal Spacing Equally* or  *Distribute Vertical Spacing Equally*

### Items Common Properties

Layout items have a set of common properties you will find at the bottom of the *Item Properties* panel: Position and size, Rotation, Frame, Background, Item ID, Variables and Rendering (see Abb. 22.11).

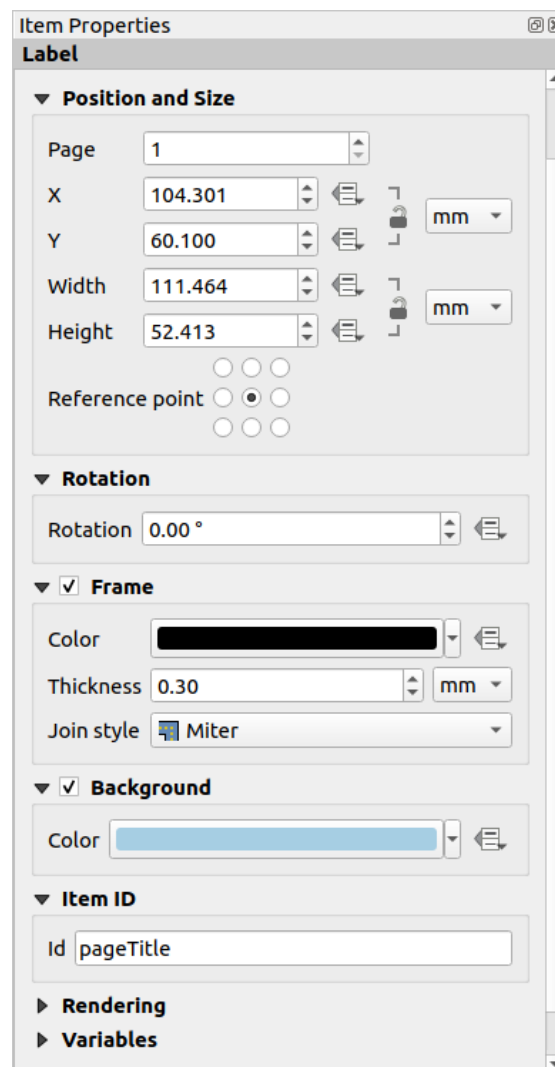




Abb. 22.11: Common Item Properties groups

**Bemerkung:** The  Data defined override icon next to most of the options means that you can associate that property



with a layer, features attributes, geometry or with any other layout item's property, using *expressions* or *variables*. For more information see *Datendefinierte Übersteuerung Setup*.

- The *Position and size* group lets you define the size and position of the frame which contains the item (see *Position and Size* for more information).
- Die *Drehung* stellt die Drehung des Elements (in Grad) ein.
- The  *Frame* shows or hides the frame around the item. Use the *Color*, *Thickness* and *Join style* widgets to adjust those properties.
- In the *Background* group you can create or pick a background *Color*. Transparency can be adjusted through altering the alpha field settings.
- Use the *Item ID* to create a relationship to other print layout items. This is used with QGIS server and other potential web clients. You can set an ID on an item (for example, a map or a label), and then the web client can send data to set a property (e.g., label text) for that specific item. The `GetProjectSettings` command will list the items and IDs which are available in a layout.
- *Rendering mode* helps you set whether and how the item can be displayed: you can, for instance, apply *blending mode*, adjust the opacity of the item or *Exclude item from exports*.

## Position and Size

Extending the features of the *New Item Properties* dialog with data-defined capabilities, this group allows you to place the items accurately.

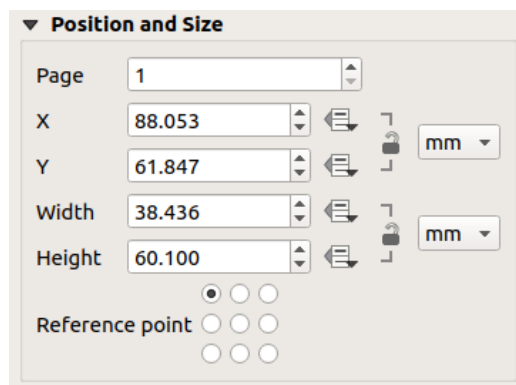




Abb. 22.12: Position and size

- the actual number of the page to place the item on;
- the reference point of the item;
- the *X* and *Y* coordinates of the *Reference point* of the item on the chosen page. The ratio between these values can be locked by clicking on the  button. Changes made to a value using the widget or the  Select/Move item tool will be reflected in both of them;
- the *Width* and *Height* of the item bounding box. As for coordinates, the ratio between width and height can be locked.

## Darstellung

QGIS allows advanced rendering for layout items just like vector and raster layers.

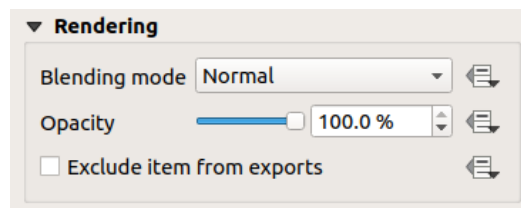





Abb. 22.13: Darstellung

- *Mischmodus*: Sie können spezielle Darstellungseffekte mit diesen Tools, die Sie vorher vielleicht nur von Grafikprogrammen kannten, erzielen. Die Pixel Ihrer oben liegenden und darunterliegenden Elemente werden durch den unten beschriebenen Modus gemischt (siehe *Mischmodi* für Beschreibungen jeden Effekts).
- *Opacity*: You can make the underlying item in the layout visible with this tool. Use the slider to adapt the visibility of your item to your needs. You can also make a precise definition of the percentage of visibility in the menu beside the slider.
-  *Exclude item from exports*: Combined with data-defined expressions, you can decide to make an item invisible in some exports. After activating this checkbox, the item will not be included in export to PDF, print etc..


## Variables

The *Variables* lists all the variables available at the layout item's level (which includes all global, project and layout's variables). Layout map items have an additional *Map settings* section for variables that provide easy access to values like the map's scale, extent, and so on.

In *Variables*, it's also possible to manage layout item level variables. Click the  button to add a new custom variable. Likewise, select any custom item-level variable from the list and click the  button to remove it.

More information on variables usage in the *Storing values in Variables* section.

### 22.2.2 Das Kartenelement

The map item is the main frame that displays the map you've designed in the map canvas. Use the  *Add Map* tool following *items creation instructions* to add a new map item that you can later manipulate the same way as exposed in *Interacting with layout items*.

By default, a new map item shows the current status of the *map canvas* with its extent and visible layers. You can customize it thanks to the *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities:

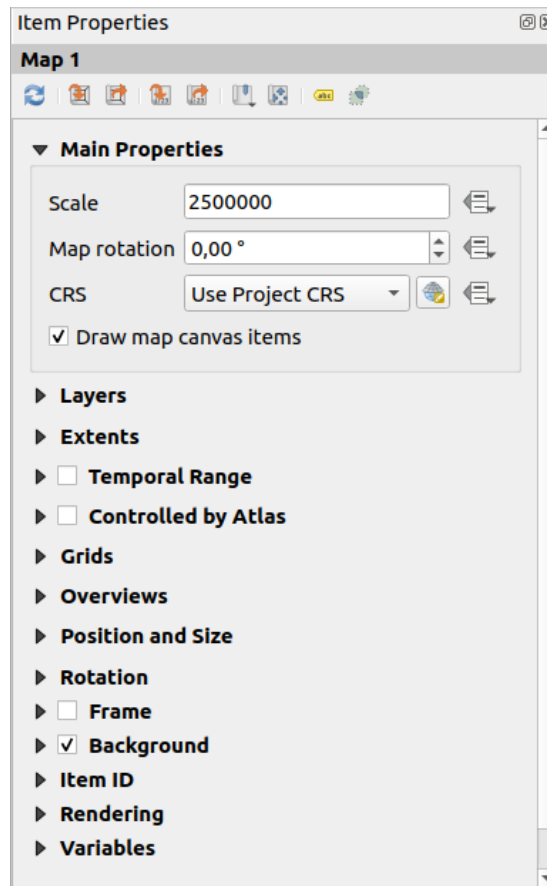











Abb. 22.14: Map Item Properties Panel

## The Toolbar


The Map *Item Properties* panel embeds a toolbar with the following functionalities:


-  Update map preview
-  Set map canvas to match main canvas extent
-  View current map extent in main canvas
-  Set map scale to match main canvas scale
-  Set main canvas to match current map scale
-  Bookmarks: set the map item extent to match an existing spatial bookmark
-  Interactively edit map extent: pan and zoom interactively within the map item
-  Labeling settings: control feature label behaviour (placement, visibility...) in the layout map item extent:
  - set a *Margin from map edges*, a data definable distance from the map item's limits inside which no label should be displayed
  -  *Allow truncated labels on edges of map*: controls whether labels which fall partially outside of the map item allowed extent should be rendered. If checked, these labels will be shown (when there's no way to place them fully within the visible area). If unchecked then partially visible labels will be skipped.

- *Label blocking items*: allows other layout items (such as scalebars, north arrows, inset maps, etc) to be marked as a blockers for the map labels in the **active** map item. This prevents any map labels from being placed under those items - causing the labeling engine to either try alternative placement for these labels or discard them altogether.

If a *Margin from map edges* is set, the map labels are not placed closer than the specified distance from the checked layout items.




- *Show unplaced labels*: can be used to determine whether labels are missing from the layout map (e.g. due to conflicts with other map labels or due to insufficient space to place the label) by highlighting them in a *predefined color*.


-  Clipping settings: allows to clip the map item to the atlas feature and to shape and polygon items:

-  *Clip to atlas feature*: you can determine that the layout map item will be clipped automatically to the current *atlas feature*.

There are different clipping modes available:

- \* *Clip During Render Only*: applies a painter based clip, so that portions of vector features which sit outside the atlas feature become invisible
- \* *Clip Feature Before Render*: applies the clip before rendering features, so borders of features which fall partially outside the atlas feature will still be visible on the boundary of the atlas feature
- \* *Render Intersecting Features Unchanged*: renders all features which intersect the current atlas feature, but without clipping their geometry.

You can  *Force labels inside atlas feature*. If you don't want to  *Clip all layers* to the atlas feature you can use the  *Clip selected layers* option.

-  *Clip to item*: it is possible to change the shape of the map item by using a *shape* or *polygon* item from the print layout. When you enable this option the map will be automatically clipped to the selected shape in the combobox. Again, the above mentioned clipping modes are available and labels can be forced to display only inside the clipping shape.

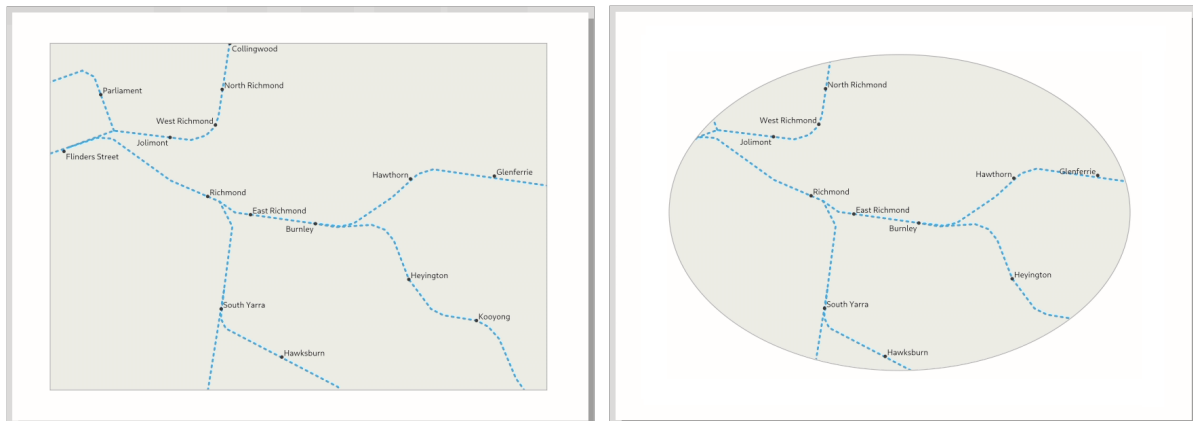


Abb. 22.15: Clipping a layout map item to shapes

## Haupteigenschaften

In the *Main properties* group (see Abb. 22.14) of the map *Item Properties* panel, available options are:

- The *Update Preview* button to refresh the map item rendering if the view in map canvas has been modified. Note that most of the time, the map item refresh is automatically triggered by the changes;
- The *Scale* to manually set the map item scale;
- The *Map rotation* allows you to rotate the map item content clockwise in degrees. The rotation of the map canvas can be imitated here;
- The *CRS* allows you to display the map item content in any *CRS*. It defaults to *Use project CRS*;
- ☒ *Draw map canvas items* lets you show in the print layout *annotations* that are placed on the main map canvas.

## Layers

By default, map item appearance is synced with the map canvas rendering meaning that toggling visibility of the layers or modifying their style in the *Layers Panel* is automatically applied to the map item. Because, like any other item, you may want to add multiple map items to a print layout, there's a need to break this synchronization in order to allow showing different areas, layer combinations, at different scales... The *Layers* properties group (see Abb. 22.16) helps you do that.

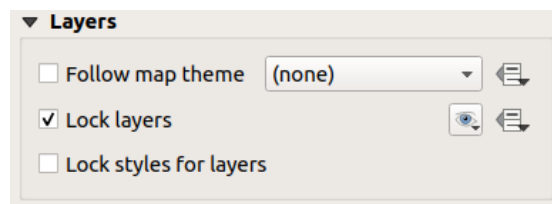





Abb. 22.16: Map Layers group

If you want to keep the map item consistent with an existing *map theme*, check ☒ *Follow map theme* and select the desired theme in the drop-down list. Any changes applied to the theme in QGIS' main window (using the replace theme function) will automatically affect the map item. If a map theme is selected, the *Lock styles for layers* option is disabled because *Follow map theme* also updates the style (symbolology, labels, diagrams) of the layers.

To lock the layers shown in a map item to the current map canvas visibility, check ☒ *Lock layers*. When this option is enabled, any changes on the layers' visibility in QGIS' main window will not affect the layout's map item. Nevertheless, style and labels of locked layers are still refreshed according to QGIS' main window. You can prevent this by using *Lock styles for layers*.

Instead of using the current map canvas, you can also lock the layers of the map item to those of an existing map theme: select a map theme from the  Set layer list from a map theme drop-down button, and the ☒ *Lock layers* is activated. The set of visible layers in the map theme is from now on used for the map item until you select another map theme or uncheck the ☒ *Lock layers* option. You then may need to refresh the view using the  Refresh view button of the *Navigation* toolbar or the *Update Preview* button seen above.

Note that, unlike the *Follow map theme* option, if the *Lock layers* option is enabled and set to a map theme, the layers in the map item will not be refreshed even if the map theme is updated (using the replace theme function) in QGIS' main window.

Locked layers in the map item can also be *data-defined*, using the  icon beside the option. When used, this overrides the selection set in the drop-down list. You need to pass a list of layers separated by | character. The following example locks the map item to use only layers `layer 1` and `layer 2`:

```
concat ('layer 1', '|', 'layer 2')
```

### Ausdehnung

The *Extents* group of the map item properties panel provides the following functionalities (see Abb. 22.17):

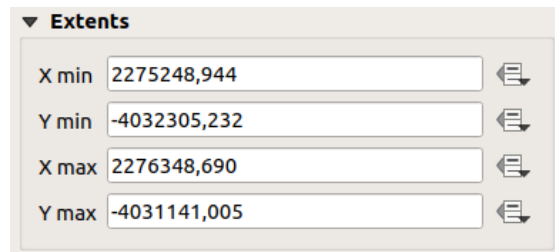






Abb. 22.17: Map Extents group

The **Extents** area displays X and Y coordinates of the area shown in the map item. Each of these values can be manually replaced, modifying the map canvas area displayed and/or map item size. The extent can also be modified using tools at the top of the map item panel such as:

-  Set map canvas to match main canvas extent
-  Set map scale to match main canvas scale


You can also alter a map item extent using the  Move item content tool: click-and-drag within the map item to modify its current view, keeping the same scale. With the  tool enabled, use the mouse wheel to zoom in or out, modifying the scale of the shown map. Combine the movement with **Ctrl** key pressed to have a smaller zoom.



### Temporal range


The *Temporal range* group of the map item properties panel provides the options to control layers rendering in the map item based on a temporal range. Only layers whose temporal properties overlap with the time range set by the *Start* and *End* dates are displayed in the map item.

The associated data-defined widgets help make the time range dynamic, and allow outputting temporal *atlases*, i.e. automated maps with fixed spatial extent and whose contents vary based on time. For example, using as coverage layer a csv file with a start and end pair of fields and a number of rows representing date ranges, enable both the temporal range and control by atlas in the map item properties and hit atlas export.

### Controlled by atlas





The  *Controlled by atlas* group properties is available only if an *atlas* is active in the print layout. Check this option if you want the map item being ruled by the atlas; when iterating over the coverage layer, the map item extent is panned/zoomed to the atlas feature following:

-  *Margin around features*: zooms to the feature at the best scale, keeping around each a margin representing a percentage of the map item width or height. The margin can be the same for all features or *set variable*, e.g., depending on map scale;
-  *Predefined scale (best fit)*: zooms to the feature at the project *predefined scale* where the atlas feature best fits;

-  *Fixed scale*: atlas features are panned from one to another, keeping the same scale of the map item. Ideal when working with features of same size (e.g., a grid) or willing to highlight size differences among atlas features.

## Gitter

With grids, you can add, over your map, information relative to its extent or coordinates, either in the map item projection or a different one. The *Grids* group provides the possibility to add several grids to a map item.

- With the  and  buttons you can add or remove a selected grid;
- With the  and  buttons you can move up and down a grid in the list, hence move it on top or bottom of another one, over the map item.

Double-click the added grid to rename it.

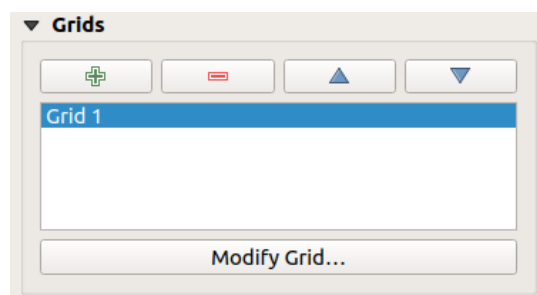


Abb. 22.18: Kartengitterdialog

To modify a grid, select it and press the *Modify Grid...* button to open the *Map Grid Properties* panel and access its configuration options.


## Grid Appearance

In the *Map Grid Properties* panel, check  *Grid enabled* to show the grid on the map item.

As grid type, you can specify to use a:

- **Solid**: shows a line across the grid frame. The *Line style* can be customized using *color* and *symbol* selector widget;
- **Cross**: displays segment at the grid lines intersection for which you can set the *Line style* and the *Cross width*;
- **Markers**: only displays customizable markers symbol at grid lines intersection;
- or **Frame and annotations only**.

Other than the grid type, you can define:

- the *CRS* of the grid: by default, it will follow the map item CRS. Press  *Select CRS* button to set it to a different CRS.
- the *Interval* type to use for the grid references:
  - **Map Units**: you set a distance within the map (in the unit of the grid CRS) between consecutive grid references in the *X* and *Y* directions. The number of grid ticks will vary depending on the map scale.
  - choosing **Fit Segment Width** will dynamically select the grid interval based on the map extent to a „pretty“ interval. That optimal interval is calculated within a range of distances whose *Minimum* and *Maximum* values can be customized.

- With **Millimeters** or **Centimeters**, you set a distance on the paper between consecutive grid references in the *X* and *Y* directions. The number of grid ticks will be the same whatever the map scale.
- the *Offset* from the map item edges, in the *X* and/or the *Y* direction
- and the *Blend mode* of the grid (see *Mischmodi*) when compatible.

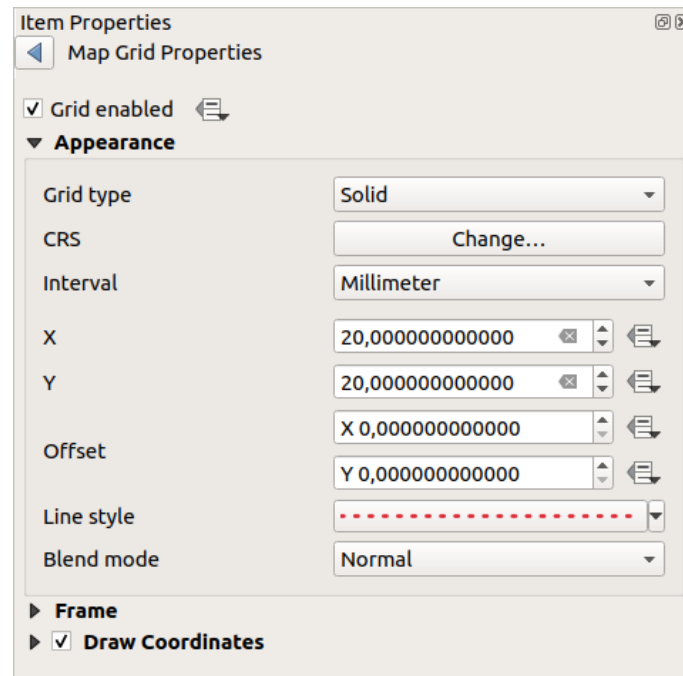


Abb. 22.19: Grid Appearance Dialog

## Grid Frame

There are different options to style the frame that holds the map. The following options are available: No Frame, Zebra, Zebra (nautical), Interior ticks, Exterior ticks, Interior and Exterior ticks, Line border and Line border (nautical).

When compatible, it's possible to set the *Frame size*, a *Frame margin*, the *Frame line thickness* with associated color and the *Frame fill colors*.

Using Latitude/*Y* only and Longitude/*X* only values in the divisions section you can prevent a mix of latitude/*Y* and longitude/*X* coordinates showing on each side when working with rotated maps or reprojected grids. Also you can choose to set visible or not each side of the grid frame.

When the map item extent is rotated (from the *Main properties* group) or the grid has a different CRS applied, grid lines may not be orthogonal to the map item sides. This can result in bad looking of the grid when styled with interior and/or exterior ticks. Checking ☒ *Follow grid rotation* will align the ticks with grid lines. Moreover, you can adjust some more properties:

- *Ticks alignment*: The interior and/or exterior ticks will be parallel to their corresponding grid line. Their alignment can be:
  - **Orthogonal**: ticks on the same side end at one line, parallel to the side. This can result e.g. in some ticks getting longer when with a low angle to the frame.
  - **Fixed length**: all ticks have the same length, so they may not align
- *Skip below angle*: prevents displaying ticks for grid lines intersecting the frame border below a specified threshold



- *Margin from map corner*: prevents displaying ticks too close to the map corners, because they could overlap and/or be out of bounds.

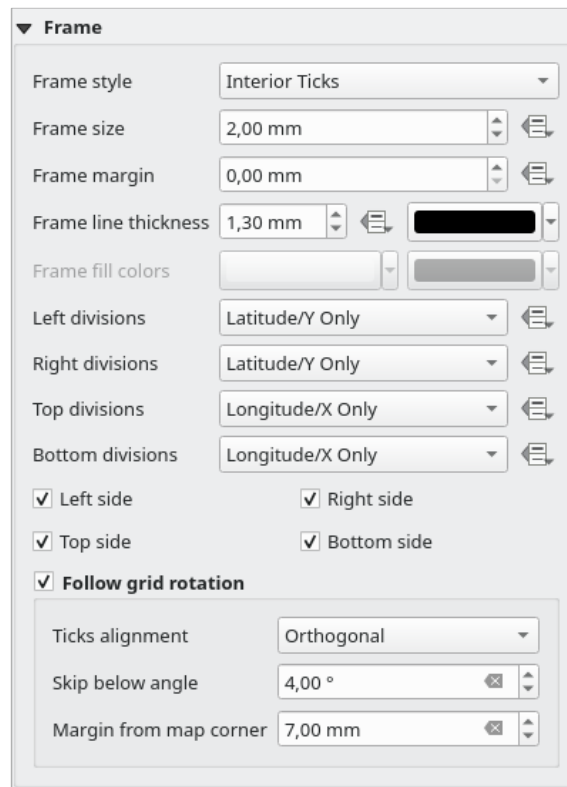



Abb. 22.20: Gitterrahmendialog


## Coordinates

The  *Draw coordinates* checkbox allows you to add coordinates to the map frame. Displayed values relate to the chosen *grid interval* unit. You can choose the annotation numeric format, the options range from decimal to degrees, minute and seconds, with or without suffix, aligned or not and a custom format using the expression dialog.

For each of the *Left*, *Right*, *Top* and *Bottom* sides of the grid frame, you can indicate:

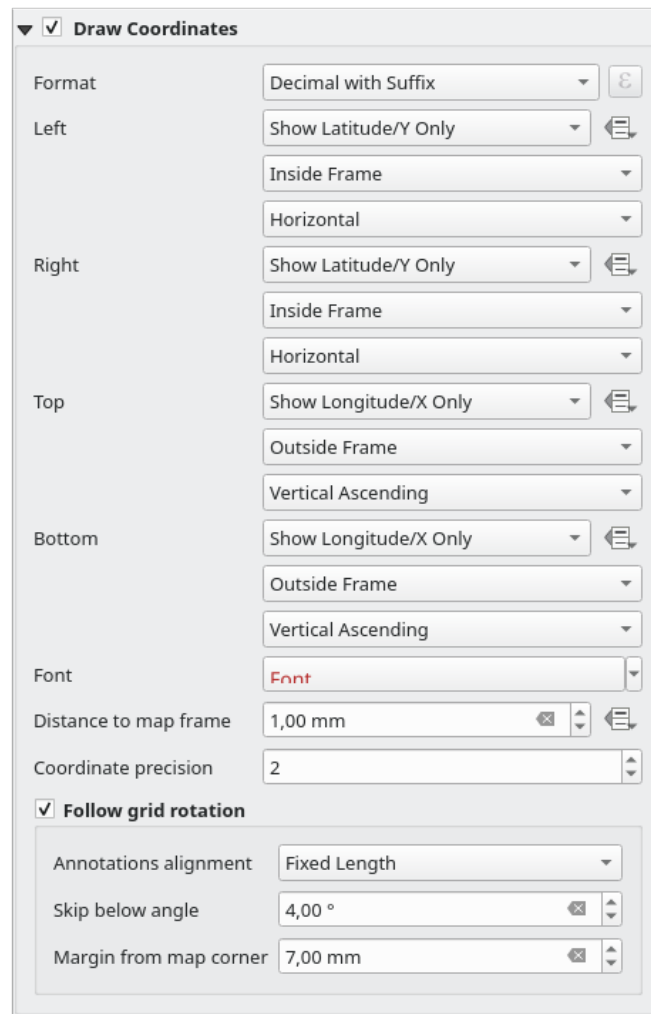
- whether to render the coordinates: **Show all**, **Show latitude/Y only**, **Show longitude/X only**, **Disabled**. Showing only Latitude/Y or Longitude/X values in the divisions helps prevent a mix of latitude/Y and longitude/X coordinates showing on each side when working with rotated maps or reprojected grids.
- the relative position of the text to the grid frame: **Outside frame** or **Inside frame**
- the placement and orientation of the annotation:
  - **Horizontal**
  - **Vertical ascending**, **Vertical descending**
  - **Boundary direction**
  - **Above tick**, **On tick**, **Under tick** when a tick-based frame is used

You can also define the *Font properties* (font, size, color, buffer,...) the *Distance to the map frame* and the *Coordinate precision* (number of decimals) for the drawn annotations.

 *Follow grid rotation*: available when the map extent is rotated or the grid is reprojected, it helps you adjust the annotations placement. Depending on the selected placement mode, the annotations are also rotated:

- *Annotations alignment*: it can be **Orthogonal** or of **Fixed length**

- *Skip below angle*: prevents displaying annotations for grid lines intersecting the frame border below a specified threshold
- *Margin from map corner*: prevents displaying annotations too close to the map corners, because they could overlap and/or be out of bounds.



**Draw Coordinates**

Format: Decimal with Suffix

Left: Show Latitude/Y Only, Inside Frame, Horizontal

Right: Show Latitude/Y Only, Inside Frame, Horizontal

Top: Show Longitude/X Only, Outside Frame, Vertical Ascending

Bottom: Show Longitude/X Only, Outside Frame, Vertical Ascending

Font: Font

Distance to map frame: 1,00 mm

Coordinate precision: 2

☒ Follow grid rotation

Annotations alignment: Fixed Length

Skip below angle: 4,00 °

Margin from map corner: 7,00 mm

Abb. 22.21: Gitter-Koordinaten-Zeichnen-Dialog

## Übersichten

Sometimes you may have more than one map in the print layout and would like to locate the study area of one map item on another one. This could be for example to help map readers identify the area in relation with its larger geographic context shown in the second map.

The *Overviews* group of the map panel helps you create the link between two different maps extent and provides the following functionalities:

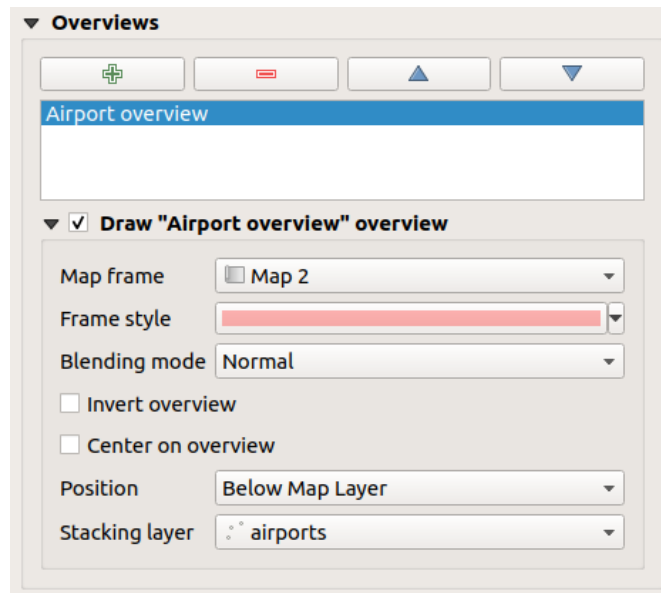


Abb. 22.22: Map Overviews group

To create an overview, select the map item on which you want to show the other map item's extent and expand the *Overviews* option in the *Item Properties* panel. Then press the button to add an overview.


Initially this overview is named 'Overview 1' (see Abb. 22.22). You can:

- Rename it with a double-click
- With the and buttons, add or remove overviews
- With the and buttons, move an overview up and down in the list, placing it above or below other overviews in the map item (when they are at the same *stack position*).

Then select the overview item in the list and check the *Draw „<name\_overview>“ overview* to enable the drawing of the overview on the selected map frame. You can customize it with:

- The *Map frame* selects the map item whose extents will be shown on the present map item.
- The *Frame Style* uses the *symbol properties* to render the overview frame.
- Der *Mischmodus* ermöglicht verschiedene Transparenzmischmodi einzustellen.
- The *Invert overview* creates a mask around the extents when activated: the referenced map extents are shown clearly, whereas the rest of the map item is blended with the frame fill color (if a fill color is used).
- The *Center on overview* pans the map item content so that the overview frame is displayed at the center of the map. You can only use one overview item to center, when you have several overviews.
- The *Position* controls exactly where in the map item's layer stack the overview will be placed, e.g. allowing an overview extent to be drawn below some feature layers such as roads whilst drawing it above other background layers. Available options are:
  - *Below map*
  - *Below map layer* and *Above map layer*: place the overview frame below and above the geometries of a layer, respectively. The layer is selected in the *Stacking layer* option.
  - *Below map labels*: given that labels are always rendered above all the feature geometries in a map item, places the overview frame above all the geometries and below any label.
  - *Above map labels*: places the overview frame above all the geometries and labels in the map item.

## 22.2.3 Das 3D Kartenelement

The 3D Map item is used to display a *3D map view*. Use the  *Add 3D Map* button, and follow *items creation instructions* to add a new 3D Map item that you can later manipulate the same way as demonstrated in *Interacting with layout items*.

By default, a new 3D Map item is empty. You can set the properties of the 3D view and customize it in the *Item Properties* panel. In addition to the *common properties*, this feature has the following functionalities (Abb. 22.23):

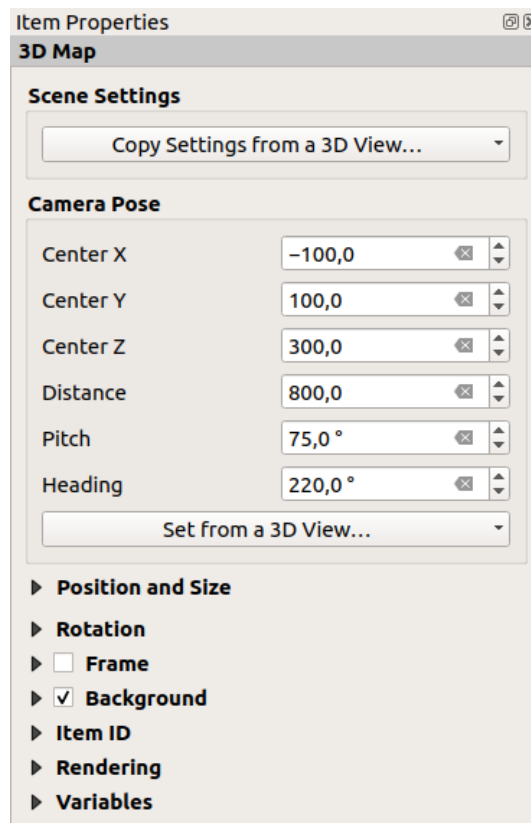


Abb. 22.23: 3D Map Item Properties

### Scene settings

Press *Copy Settings from a 3D View...* to choose the 3D map view to display.

The 3D map view is rendered with its current configuration (layers, terrain, lights, camera position and angle...).


### Camera pose

- *Center X* sets the X coordinate of the point the camera is pointing at
- *Center Y* sets the Y coordinate of the point the camera is pointing at
- *Center Z* sets the Z coordinate of the point the camera is pointing at
- *Distance* sets the distance from the camera center to the point the camera is pointing at
- *Pitch* sets the rotation of the camera around the X-axis (vertical rotation). Values from 0 to 360 (degrees). 0°: terrain seen straight from above; 90°: horizontal (from the side); 180°: straight from below; 270°: horizontal, upside down; 360°: straight from above.

- *Heading* sets the rotation of the camera around the Y-axis (horizontal rotation - 0 to 360 degrees). 0°/360°: north; 90°: west; 180°: south; 270°: east.

The *Set from a 3D View...* pull-down menu lets you populate the items with the parameters of a 3D View.

## 22.2.4 Das Beschriftungselement

The *Label* item is a tool that helps decorate your map with texts that would help to understand it; it can be the title, author, data sources or any other information... You can add a label with the  *Add Label* tool following [items creation instructions](#) and manipulate it the same way as exposed in [Interacting with layout items](#).

By default, the label item provides a default text that you can customize using its *Item Properties* panel. Other than the [items common properties](#), this feature has the following functionalities (see [Abb. 22.24](#)):

Label

▼ Main Properties

Airports of [%"NAME"%] region|

☐ Render as HTML

Insert/Edit Expression...

▼

Dynamic Text

▼

▼ Appearance

Font

▼

Horizontal margin

0,00 mm

▲▼

Vertical margin

0,00 mm

▲▼

Horizontal alignment

☐ Left
 ☐ Center
 ☐ Right
 ☒ Justify

Vertical alignment

☒ Top
 ☐ Middle
 ☐ Bottom

Abb. 22.24: Label Item Properties Panel

778

Kapitel 22. Kartenlayout

## Haupteigenschaften

The *Main properties* group is the place to provide the text of the label. The text can be static, dynamic with *expression* functions and variables, and/or formatted with HTML. Dynamic parts of a label need to be surrounded by [% and %] in order to be interpreted and evaluated as such.

- To use expressions in labels, you can click on *Insert/Edit Expression...* button, write your formula as usual and when the dialog is applied, QGIS automatically adds the surrounding characters.

---

**Hinweis:** Clicking the *Insert/Edit Expression...* button when no selection is made in the textbox will append the new expression to the existing text. If you want to modify an existing expression, you need to first select the part of interest.

---


Because maps are usually filled with some common textual information (date, author, title, page number, ...), QGIS provides a direct access to the corresponding expressions or variables: press the *Dynamic text* button to select and insert them into your label.

---

**Tipp:** The top menu *Add Item ► Add Dynamic Text ►* can be used to create a new label item filled with the selected predefined expression.

---

It's possible to turn a dynamic label into static: press the drop-down arrow next to the *Insert/Edit Expression...* button and select *Convert to Static*. Any dynamic parts of the label's contents will be evaluated and replaced with their current values. You can then manually tweak the resulting text when needed.

- Labels can be interpreted as HTML code: check  *Render as HTML*. You can now insert HTML tags or styles, URL, a clickable image that links to a web page, or something more complex...

The following code combines HTML rendering with expressions, for an advanced labeling and will output [Abb. 22.25](#):

```
<html>
<head>
  <style>
    /* Define some custom styles, with attribute-based size */
    name {color:red; font-size: [% ID %]px; font-family: Verdana; text-shadow: 1px 1px 0px grey 1px 0 10px;}
    use {color:blue;}
  </style>
</head>

<body>
  <!-- Information to display -->
  <u>Feature Information</u>
  <ul style="list-style-type:disc">
    <li>Feature Id: [% ID %]</li>
    <li>Airport: <name>[% NAME %]</name></li>
    <li>Main use: <use>[% USE %]</use></li>
  </ul>
  Last check: [% concat( format_date( "control_date", 'yyyy-MM-dd'), ' by <b><i>',
  @user_full_name, '</i></b>' ) %]

  <!-- Insert an image -->
  <p align=center></p>
</body>
</html>
```

#### Feature Information

- Feature number: 36

• Airport name: **FAIRBANKS INTL**

- Main use: **Civilian/Public**

Last check: 2021-01-26 by **John McClane**



Abb. 22.25: Leveraging a label with HTML styling

## Darstellung

- Define font and style of the text by clicking on the *Font* button. In the *Label Font* menu you can use some of the options for *Formatting the label text*.
- You can specify different horizontal and vertical margins in mm. This is the margin from the edge of the layout item. The label can be positioned outside the bounds of the label e.g. to align label items with other items. In this case you have to use negative values for the margin.
- Using the text alignment is another way to position your label. It can be:
  - *Left, Center, Right* or *Justify* for *Horizontal alignment*
  - and *Top, Middle, Bottom* for *Vertical alignment*.

## Exploring expressions in a label item

Below some examples of expressions you can use to populate the label item with interesting information - remember that the code, or at least the calculated part, should be surrounded by [% and %] in the *Main properties* frame:

- Display a title with the current atlas feature value in „field1“:

```
'This is the map for ' || "field1"
```

or, written in the *Main properties* section:

```
This is the map for [% "field1" %]
```

- Add a pagination for processed atlas features (eg, Page 1/10):

```
concat( 'Page ', @atlas_featurenumber, '/', @atlas_totalfeatures )
```

- Return the name of the airports of the current atlas region feature, based on their common attributes:

```
aggregate( layer := 'airports',
  aggregate := 'concatenate',
  expression := "NAME",
  filter := fk_regionId = attribute( @atlas_feature, 'ID' ),
  concatenator := ', '
)
```

Or, if an *attributes relation* is set:

```
relation_aggregate( relation := 'airports_in_region_relation',
  aggregate := 'concatenate',
  expression := "NAME",
  concatenator := ', '
)
```



- Return the name of the airports contained in the current atlas region feature, based on their spatial relationship:

```
aggregate( layer := 'airports',
           aggregate := 'concatenate',
           expression := "NAME",
           filter := contains( geometry( @parent ), $geometry ),
           concatenator := ', '
         )
```

OR:

```
array_to_string( array:= overlay_contains( layer := 'airports',
                                           expression := "NAME" ),
                 delimiter:= ', '
               )
```

- Return the lower X coordinate of the Map 1 item's extent:

```
x_min( map_get( item_variables( 'Map 1' ), 'map_extent' ) )
```


- Retrieve the name of the layers in the current layout Map 1 item, and formats in one name by line:

```
array_to_string(
  array_foreach(
    map_get( item_variables( 'Map 1' ), 'map_layers' ), -- retrieve the layers_
    ↪list
    layer_property( @element, 'name' ) -- retrieve each layer name
  ),
  '\n' -- converts the list to string separated by breaklines
)
```

- Display the list of layers with their license strings (usage rights) in a layout Map 1 item. You need to fill the layers' *Access metadata* properties first.

```
array_to_string( map_credits( 'Map 1', true ) )
```

## 22.2.5 Das Legendenelement

The *Legend* item is a box or a table that explains the meanings of the symbols used on the map. A legend is then bound to a map item. You can add a legend item with the  *Add Legend* tool following *items creation instructions* and manipulate it the same way as exposed in *Interacting with layout items*.

By default, the legend item displays all available layers and can be refined using its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see Abb. 22.26):

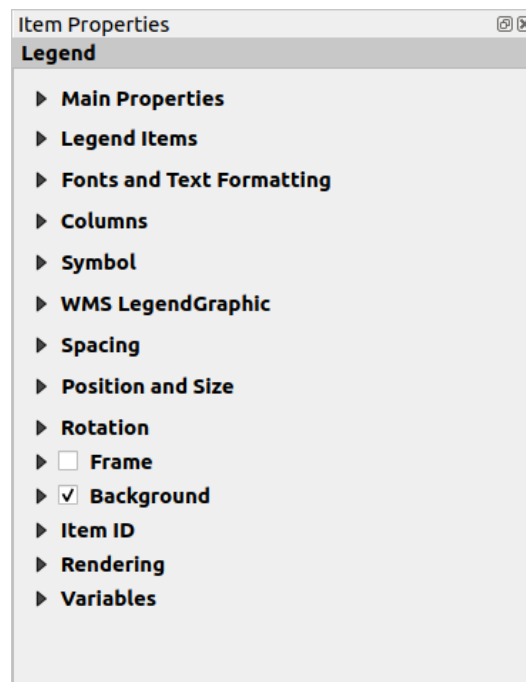


Abb. 22.26: Legend Item Properties Panel

## Haupteigenschaften

The *Main properties* group of the legend *Item Properties* panel provides the following functionalities (see [Abb. 22.27](#)):

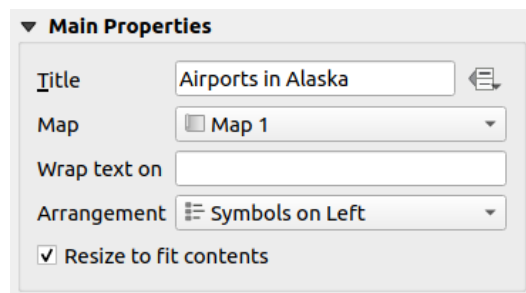


Abb. 22.27: Einstellung der Haupteigenschaften der Legende

In den Haupteigenschaften können Sie:


- den Legenden *Titel* ändern. Er kann durch die Einstellung von *Daten definierter Überschreibung* dynamisch verwendet werden, was z.B. bei der Atlaserstellung hilfreich sein kann;
- Choose which *Map* item the current legend will refer to. By default, the map over which the legend item is drawn is picked. If none, then it falls back to the *reference map*.

---

**Bemerkung:** *Variables* of the linked map item (@map\_id, @map\_scale, @map\_extent...) are also accessible from data-defined properties of the legend.

---

- Wrap the text of the legend on a given character: each time the character appears, it's replaced with a line break;
- Set the symbols and text placement in the legend: the *Arrangement* can be *Symbols on left* or *Symbols on right*. The default value depends on the locale in use (right-to-left based or not).

- Use  *Resize to fit contents* to control whether or not a legend should be automatically resized to fit its contents. If unchecked, then the legend will never resize and instead just stick to whatever size the user has set. Any content which doesn't fit the size is cropped out.

## Legendenelemente

The *Legend items* group of the legend *Item Properties* panel provides the following functionalities (see Abb. 22.28):

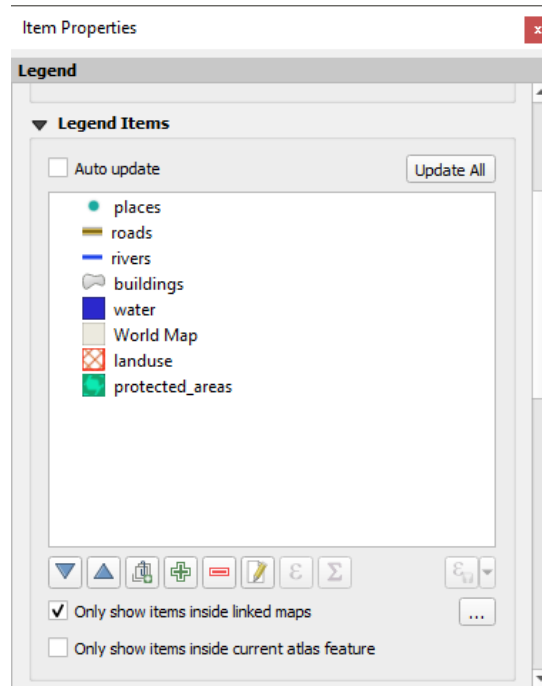















Abb. 22.28: Legend Items group


- The legend will be updated automatically if  *Auto update* is checked. When *Auto update* is unchecked this will give you more control over the legend items. All the icons below the legend items list will be activated.
- The legend items window lists all legend items and allows you to change item order, group layers, remove and restore items in the list, edit layer names and symbology and add a filter.
  - The item order can be changed using the  and  buttons or with 'drag-and-drop' functionality. The order can not be changed for WMS legend graphics.
  - Use the  button to add a legend group.
  - Use the  button to add layers and  button to remove groups, layers or symbol classes.
  - The  button is used to edit the layer, group name or title. First you need to select the legend item. Double-clicking the item also opens the text box to rename it.
  - The  button uses expressions to customize each symbol label of the selected layer (see *Daten verknüpfte Legendenbeschriftung*)
  - The  button adds a feature count for each class of vector layer.
  - The  *Filter legend by expression* helps you filter which of the legend items of a layer will be displayed, i.e. using a layer that has different legend items (e.g., from a rule-based or categorized symbology), you can specify a boolean expression to remove from the legend tree, styles that have no feature satisfying a condition. Note that the features are nevertheless kept and shown in the layout map item.

While the default behavior of the legend item is to mimic the *Layers* panel tree, displaying the same groups, layers and classes of symbology, right-click any item offers you options to hide layer's name or raise it as a group or subgroup. In case you have made some changes to a layer, you can revert them by choosing *Reset to defaults* from the contextual menu of the legend entry.


After changing the symbology in the QGIS main window, you can click on *Update All* to adapt the changes in the legend element of the print layout.

- With the  *Only show items inside linked maps*, only the legend items visible in the linked map will be listed in the legend. If you have more than one map you can click on ... and select other maps from your layout. This tool remains available when  *Auto-update* is active.
- While generating an atlas with polygon features, you can filter out legend items that lie outside the current atlas feature. To do that, check the  *Only show items inside current atlas feature* option.

### Daten verknüpfte Legendenbeschriftung

 allows you to add *expressions* to each symbol label of a given layer. New variables (@symbol\_label, @symbol\_id and @symbol\_count) help you interact with the legend entry.

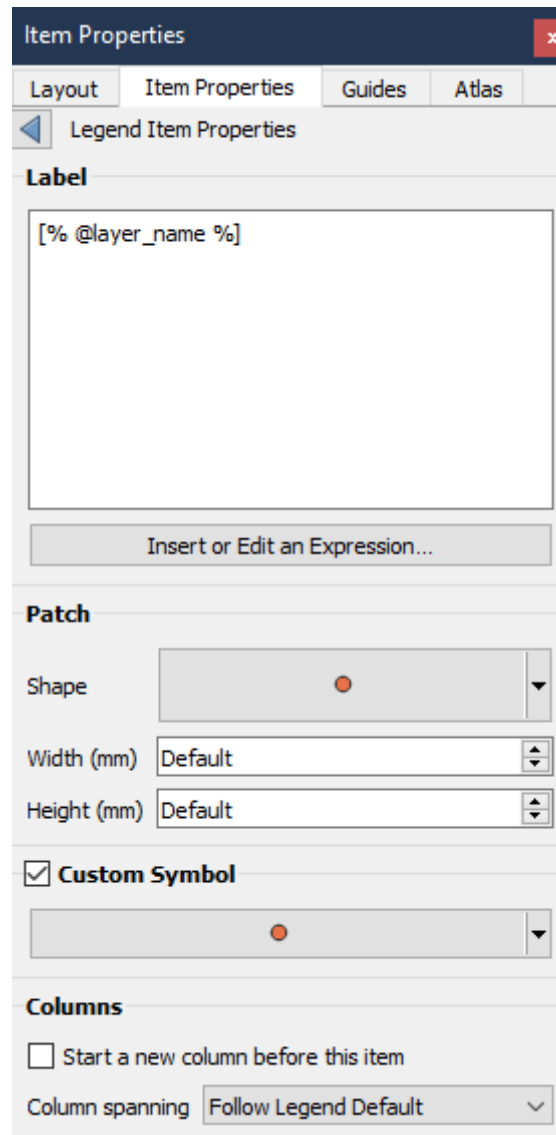
For example, given a *regions* layer categorized by its *type* field, you can append to each class in the legend their number of features and total area, e.g. *Borough (3) - 850ha*:


1. Select the layer entry in the legend tree
2. Press the  button, opening the *Expression String Builder* dialog
3. Enter the following expression (*assuming symbol labels have not been edited*):


```
format( '%1 (%2) - %3ha',  
        @symbol_label,  
        @symbol_count,  
        round( aggregate(@layer, 'sum', $area, filter:= "type"=@symbol_label) /  
→10000 )  
        )
```

4. Klicken Sie auf *OK*


## Customizing legend items



Legend items can also be customized individually in the *Legend Items Properties*. But these customization can only be done with  *Auto update* disabled.

Double-clicking on an item or pressing  *Edit selected item properties* allows for further customization.

### Label

For all item types it allows to modify the label text by typing in or by inserting expressions using the  *Insert or Edit an Expression*. Expressions can also be added directly anywhere in the item's label by using the [% expression %] notation.

### Columns

The Legend Item Property also allows you to control the column splitting behaviour by forcing the column split to occur after a specific item or all symbols of a layer. Automatic splitting of a layer and its child can also be allowed or blocked on a layer-basis in this widget.

### Patch

For items with a symbol, the Legend Item Property allows you to specify the maximum height and width that a symbol can occupy.

For vector symbols, you can specify a custom shape for the symbol. The shapes are usually defined by an expression to represent the geometry in a simple plane, but those symbols can also be saved in the style manager and imported later. The default symbol for each geometry type can also be controlled via the style manager.

### Custom Symbol

A custom symbol can also be specified for vector symbols. This can be useful to tweak the render of a specific symbol, to enhance it in the legend or have a symbol independent from its true symbol preview. This custom symbol will override the legend symbol, but will take into account the symbol *Patch* specified.

## Fonts and text formatting

The *Fonts and text formatting* group of the legend *Item Properties* panel provides the following functionalities:

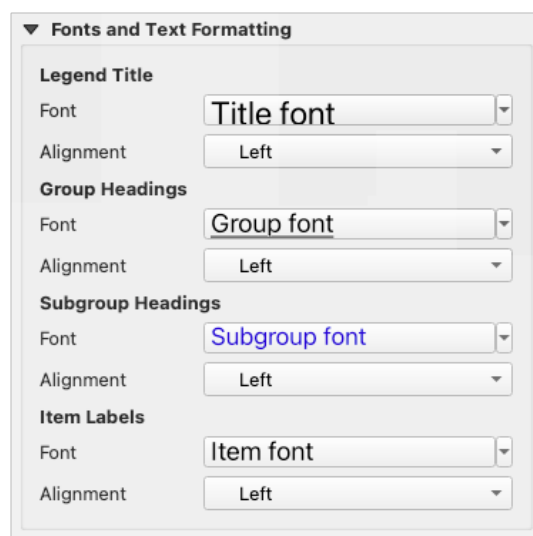


Abb. 22.29: Legend Fonts properties

- You can change the font of the legend title, group, subgroup and item (feature) in the legend item using the *font selector* widget, providing all the capabilities of *text formatting* (font spacing control, mixed HTML formatting, coloring, blending, background, text buffer, shadow, ...)
- For each of these levels you can set the text *Alignment*: it can be *Left* (default for left-to-right based locales), *Center* or *Right* (default for right-to-left based locales).

## Columns

Under the *Columns* group of the legend *Item Properties* panel, legend items can be arranged over several columns:

- Set the number of columns in the *Count*  field. This value can be made dynamic e.g., following atlas features, legend contents, the frame size...
- ☒ *Gleiche Spaltenbreite* stellt ein, wie Legendenspalten angepasst werden sollen.
- Die ☒ *Layer aufteilen* Option ermöglicht es eine kategorisierte oder abgestufte Layerlegende in Spalten aufzuteilen.

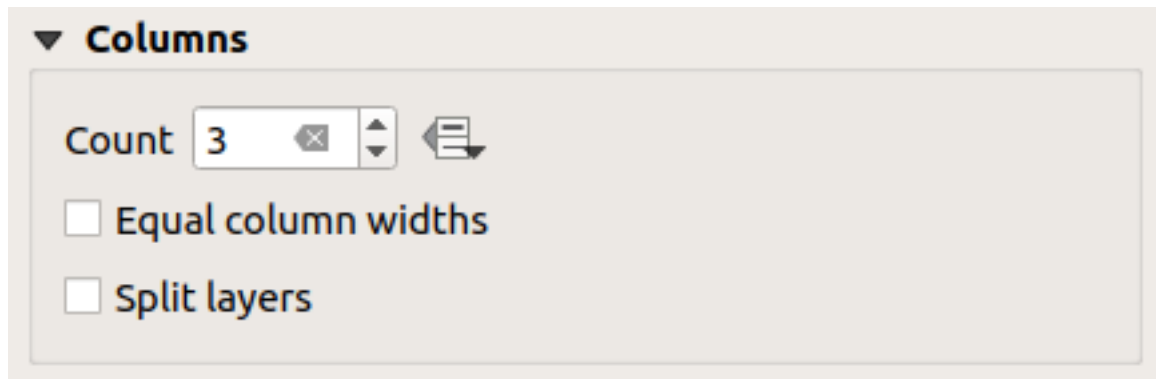


Abb. 22.30: Legend Columns settings

## Symbol

The *Symbol* group of the legend *Item Properties* panel configures the size of symbols displayed next to the legend labels. You can:

- Set the *Symbol width* and *Symbol height*
- Set the markers' *Min symbol size* and *Max symbol size*: 0 . 00mm means there is no value set.
- ☒ *Draw stroke for raster symbols*: this adds an outline to the symbol representing the band color of the raster layer; you can set both the *Stroke color* and *Thickness*.

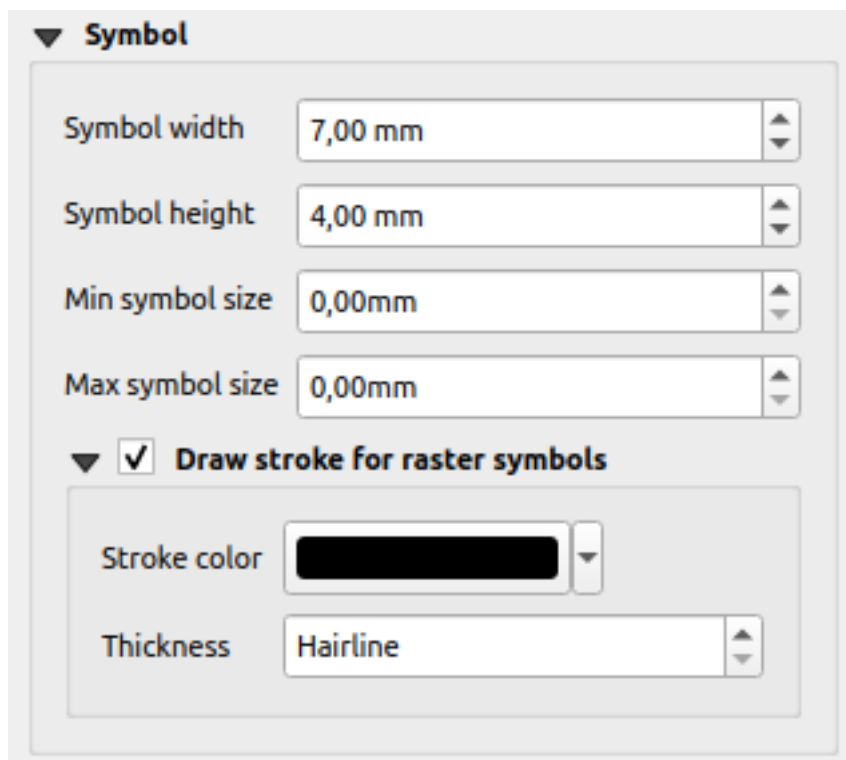


Abb. 22.31: Legend Symbol configuration

## WMS LegendGraphic

The *WMS LegendGraphic* section of the legend *Item Properties* panel provide the following functionalities (see Abb. 22.32):

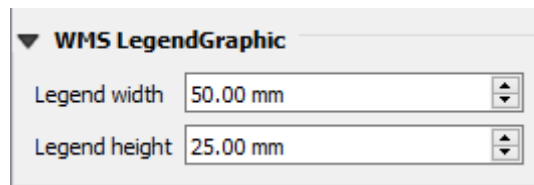
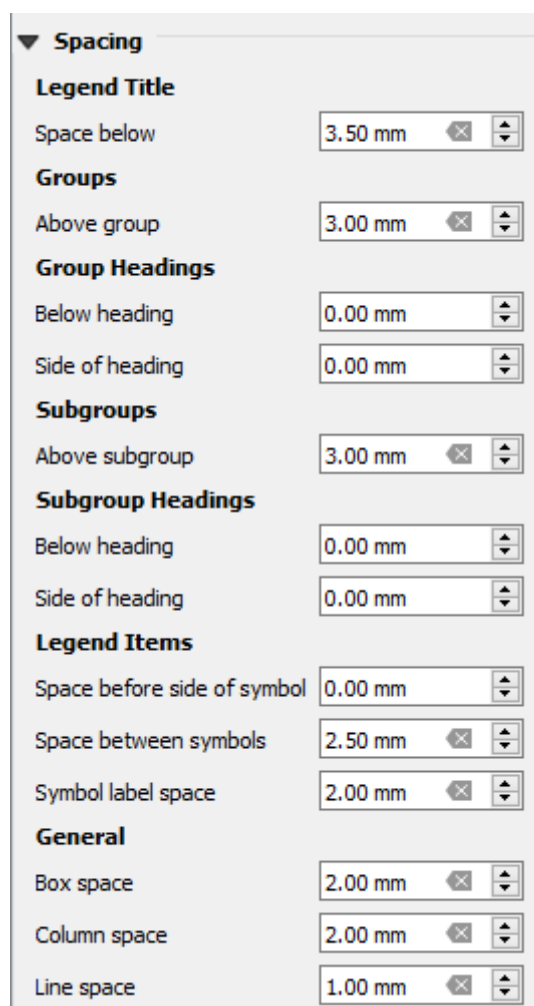


Abb. 22.32: WMS LegendGraphic

When you have added a WMS layer and you insert a legend item, a request will be sent to the WMS server to provide a WMS legend. This Legend will only be shown if the WMS server provides the GetLegendGraphic capability. The WMS legend content will be provided as a raster image.

*WMS LegendGraphic* wird verwendet um die *Legendenbreite* und die *Legendenhöhe* des WMS Legendengrasterbildes anzupassen.

## Spacing






The *Spacing* section allows you to customize the spacing within the legend. Spacing can greatly help denote the groupement of items in the legend and their relation.

*Spacing* around and before title, groups, subgroups, symbols, labels, boxes, columns and lines can be customized through this dialog.

## 22.2.6 Das Maßstabselement

Scale bars provide a visual indication of the size of features, and distance between features, on the map item. A scale bar item requires a map item. Use the  *Add Scale Bar* tool following *items creation instructions* to add a new scale bar item that you can later manipulate the same way as exposed in *Interacting with layout items*.

By default, a new scale bar item shows the scale of the map item over which it is drawn. If there is no map item below, the *reference map* is used. You can customize it in the *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see Abb. 22.33):

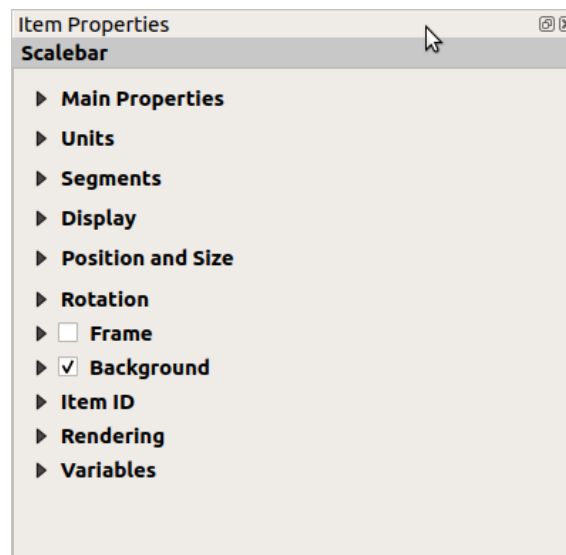


Abb. 22.33: Scale Bar Item Properties Panel

### Haupteigenschaften

The *Main properties* group of the scale bar *Item Properties* panel provides the following functionalities (see Abb. 22.34):

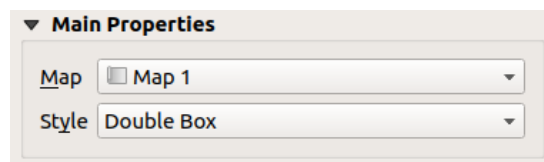


Abb. 22.34: Scale Bar Main properties group

1. First, choose the map the scale bar will be attached to
2. Then, choose the style of the scale bar. Available styles are:
  - **Single box** and **Double box** styles, which contain one or two lines of boxes alternating colors;
  - **Middle**, **Up** or **Down** line ticks;

- **Stepped line** style that draws a stepped line representation of a scalebar
  - **Hollow** style that draws a single box with alternating color for the segments, with horizontal lines through alternating segments
  - **Numeric**, where the scale ratio is printed (e.g., 1 : 50000).
3. Set properties as appropriate

## Units

The *Units* group of the scale bar *Item Properties* panel provides the functionalities to set the units of display and some text formatting (see Abb. 22.35):

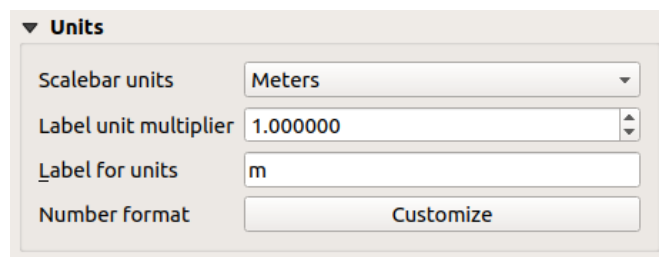


Abb. 22.35: Scale Bar Units group

- Select the units you want to use with *Scalebar units*. There are many possible choices: **Map Units** (the default one), **Meters**, **Feet**, **Miles** or **Nautical Miles**... and some derivatives. Units conversion is handled automatically.
- The *Label unit multiplier* specifies how many scale bar units per labeled unit. Eg, if your scale bar units are set to „meters“, a multiplier of 1000 will result in the scale bar labels in „kilometers“.
- The *Label for units* field defines the text used to describe the units of the scale bar, eg m or km. This should be matched to reflect the multiplier above.
- Press *Customize* next to *Number format* to have control over all the formatting properties for the numbers in the scale bar, including thousand separators, decimal places, scientific notation, etc. (see [Number Formatting](#) for more details). Very useful in the case of making maps for audiences outside of the current QGIS locale, or when you would like to vary the style from the locale defaults (e.g. adding thousands separators when the locale default is to hide them).

## Segments

The *Segments* group of the scale bar *Item Properties* panel provides the functionalities to configure the number and size of segments and subdivisions (see [Abb. 22.36](#)):

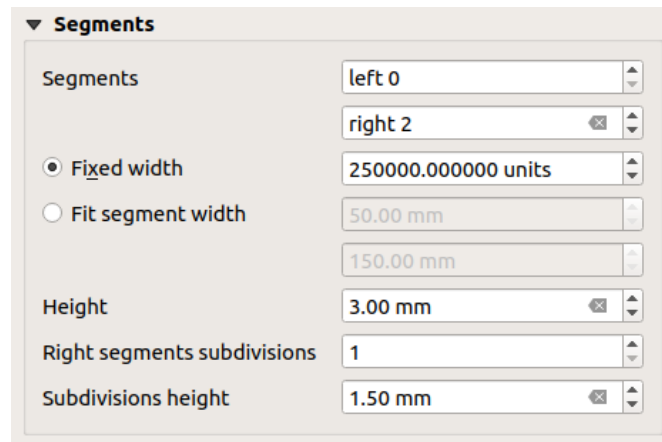


Abb. 22.36: Scale Bar Segments group

- You can define the number of *Segments* that will be drawn at the left and right sides of the 0 of the scale bar:
  - number of subdivisions of a unique segment on the *Left* side
  - number of segments on the *Right* side
- You can set the width of a segment or a range for the total length of the scale bar:
  - Set how long a segment will be in your scale bar units (*Fixed width*)
  - Or limit the total scale bar size in mm with *Fit segment width* option. In the latter case, each time the map scale changes, the scale bar is resized (and its label updated) to fit between the upper and lower range set.
- *Höhe* wird benutzt, um die Höhe der Leiste einzustellen.
- *Right segment subdivisions* is used to define the number of sections the right-side segments of the scale bar can have (for *Line Ticks Down*, *Line Ticks Middle* and *Line Ticks Up* scale bar styles) .
- *Subdivision height* is used to define the height of the subdivision segment.

## Anzeigen

The *Display* group of the scale bar *Item Properties* panel provides the following functionalities:

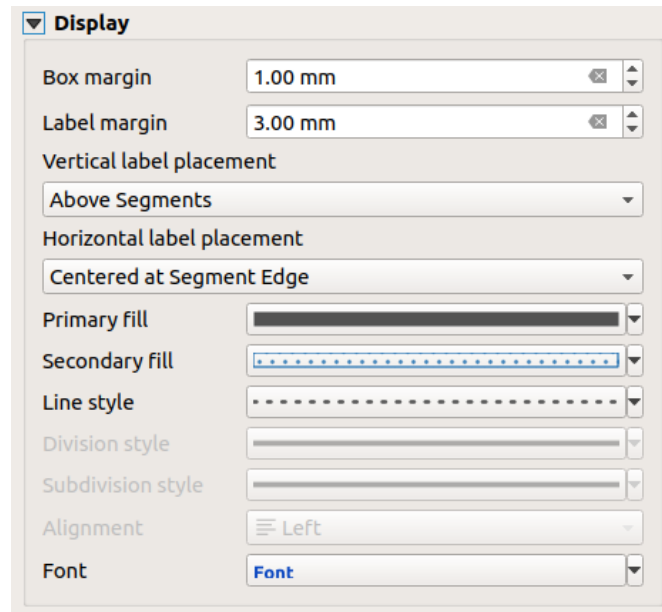


Abb. 22.37: Scale Bar Display group

Sie können festlegen wie der Maßstab in seinem Rahmen dargestellt wird.

- *Rahmenrand*: Zwischenraum zwischen Text- und Rahmengrenzen
- *Label margin* : space between text and scale bar drawing
- *Vertical label placement*: it can be above or below the scale bar segment
- *Horizontal label placement*: which would be centered at the scale bar segment's edge or center
- *Primary fill* and *Secondary fill* of the scale bar drawing using *fill symbols properties* (color, opacity, patterns, effects...) — for *Single Box*, *Double Box* and *Hollow* styles
- *Line style* of the scale bar drawing using *line symbols properties* (color, stroke, join, cap style, patterns, effects...) — for all but *Numeric* style
- *Division style* and *Subdivision style* respectively for division and subdivision segments in *Line Ticks Up*, *Line Ticks Middle* and *Line Ticks Down* scale bar styles using *line symbols properties* (color, stroke, join, cap style, patterns, effects...)
- *Alignment* puts text on the left, center or right side of the frame (only for *Numeric* scale bar style)
- *Font* to set the *properties* (size, font, color, letter spacing, shadow, background...) of the scale bar label.

Since most of the display properties of the scale bar rely on symbols whose properties can be data-defined, it's possible to render data-defined scale bars.

**Example:** The following code applied to the bold property of the scale labels will display numbers in bold when they are a multiple of 500:

```
-- returns True (or 1) if the value displayed on the bar
-- is a multiple of 500


@scale_value % 500 = 0
```

## 22.2.7 Das Tabellenelement

You can use table items to decorate and explain your map:

- *Attribute table*: shows a subset of the attributes of a layer, based on predefined rules
- *Fixed table*: inserts a manual text table where information can be independent from the layers.

### Das Attributtabellenelement

Any layer in the project can have its attributes shown in the print layout. Use the  *Add Attribute Table* tool following *items creation instructions* to add a new table item that you can later manipulate the same way as exposed in *Interacting with layout items*.

By default, a new attribute table item loads first rows of the first (alphabetically sorted) layer, with all the fields. You can however customize the table thanks to its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see [Abb. 22.38](#)):

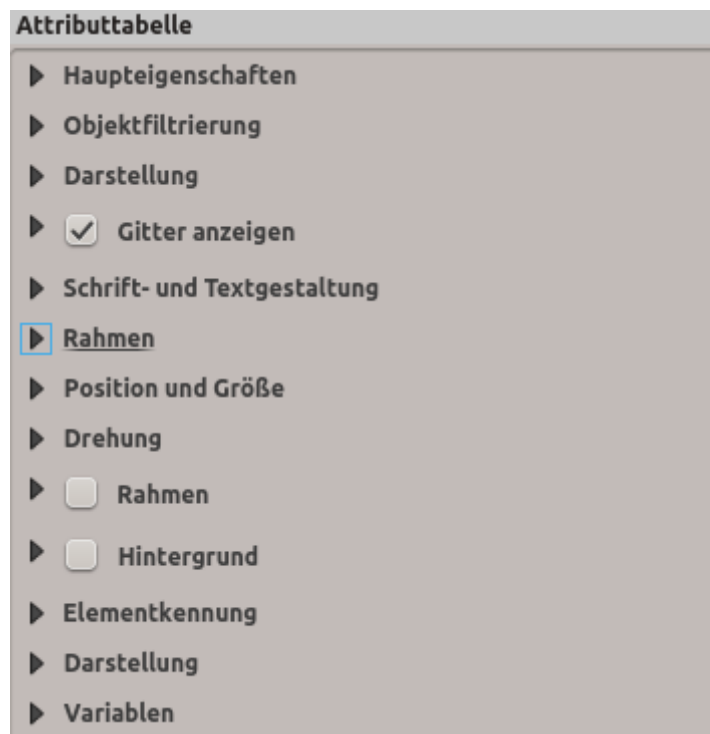


Abb. 22.38: Attribute table Item Properties Panel

### Haupteigenschaften

The *Main properties* group of the attribute table provides the following functionalities (see [Abb. 22.39](#)):

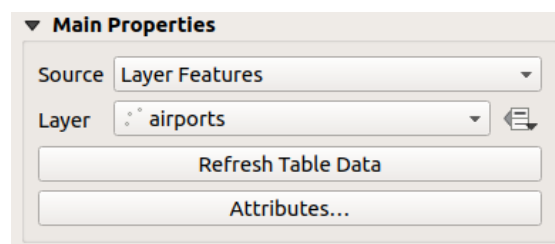




Abb. 22.39: Attribute table Main properties Group

- For *Source* you can by default only select **Layer features** allowing you to select a *Layer* from the vector layers loaded in the project.

The  Data-defined override button near the layer list allows you to dynamically change the layer which is used to populate the table, e.g. you could fill the attribute table with different layer attributes per atlas page. Note that the table structure used (Abb. 22.42) is the one of the layer shown in the *Layer* drop-down list and it is left intact, meaning that setting a data defined table to a layer with different field(s) will result in empty column(s) in the table.

In case you activate the  *Generate an atlas* option in the *Atlas* panel (see *Einen Atlas erzeugen*), there are two additional *Source* possible:

- **Current atlas feature** (see Abb. 22.40): you won't see any option to choose the layer, and the table item will only show a row with the attributes from the current feature of the atlas coverage layer.
  - and **Relation children** (see Abb. 22.41): an option with the relation names will show up. This feature can only be used if you have defined a *relation* using your atlas coverage layer as parent, and the table will show the children rows of the atlas coverage layer's current feature.
- The button *Refresh Table Data* can be used to refresh the table when the actual contents of the table has changed.

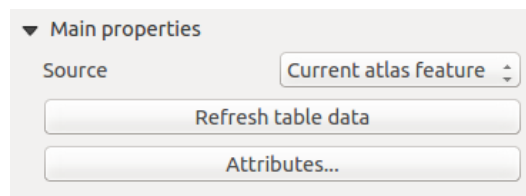


Abb. 22.40: Attribute table Main properties for 'Current atlas feature'

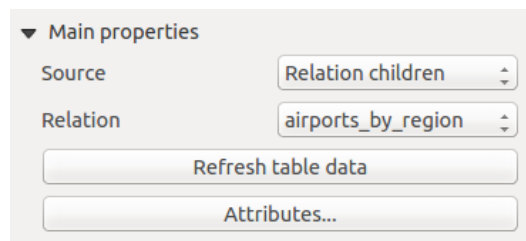


Abb. 22.41: Attribute table Main properties for 'Relation children'

- The button *Attributes...* starts the *Select Attributes* dialog, (see Abb. 22.42) that can be used to change the visible contents of the table. The upper part of the window shows the list of the attributes to display and the lower part helps you sort the data.

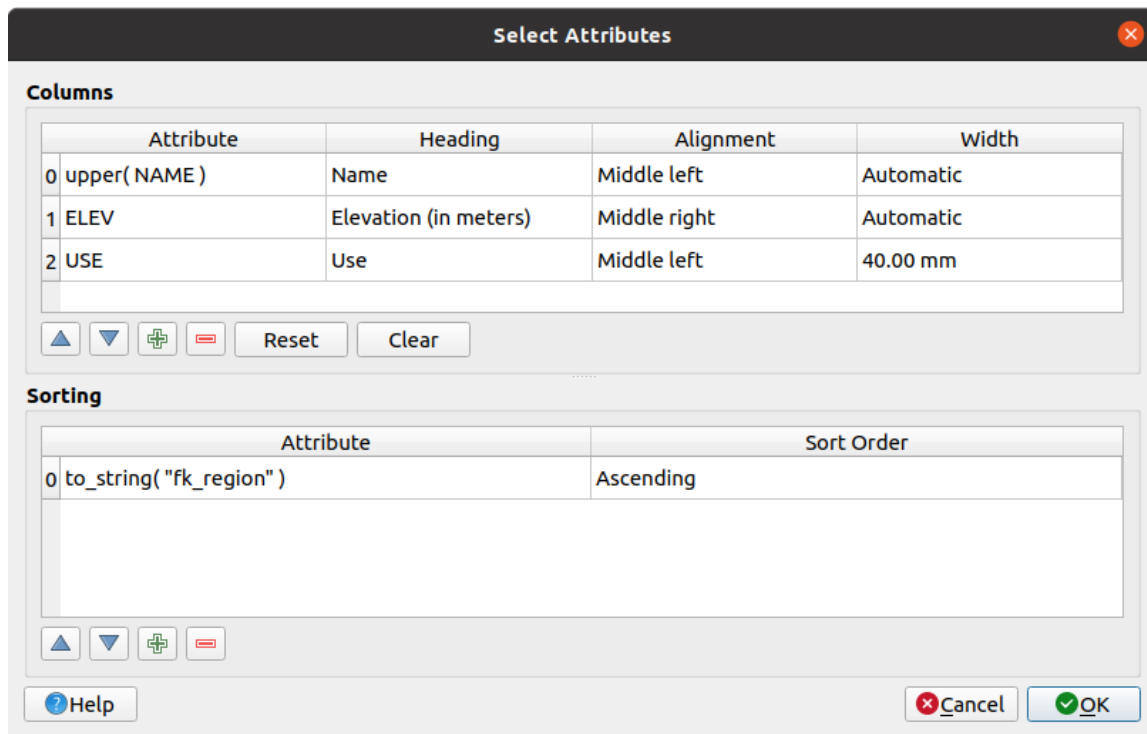


Abb. 22.42: Attribute table Select attributes Dialog

Im *Spalten* Abschnitt können Sie:

- Move attributes up or down the list by selecting the rows and then using the and buttons to shift the rows. Multiple rows can be selected and moved at any one time.
- Add an attribute with the button. This will add an empty row at the bottom of the table where you can select a field to be the attribute value or create an attribute via a regular expression.
- Remove an attribute with the button. Multiple rows can be selected and removed at any one time.
- Reset the attribute table back to its default state with the *Reset* button.
- Clear the table using the *Clear* button. This is useful when you have a large table but only want to show a small number of attributes. Instead of manually removing each row, it may be quicker to clear the table and add the rows needed.
- Cell headings can be altered by adding the custom text in the *Heading* column.
- Cell alignment can be managed with the *Alignment* column which will dictate the texts position within the table cell.
- Cell width can be manually managed by adding custom values to the *width* column.

Im *Sortierung* Abschnitt können Sie:

- Add an attribute to sort the table with: press the button and a new empty row is added. Insert a field or an expression in the *Attribute* column and set the *Sort order* to **Ascending** or **Descending**.
- Select a row in the list and use the and buttons to change the sort priority on attribute level. Selecting a cell in the *Sort Order* column helps you change the sorting order of the attribute field.
- Use the button to remove an attribute from the sorting list.

## Objektfiltrierung

The *Feature filtering* group of the attribute table provides the following functionalities (see [Abb. 22.43](#)):

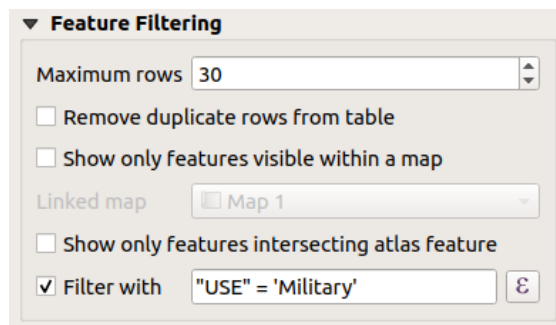


Abb. 22.43: Attribute table Feature filtering Group

Sie können:

- Die *Maximalen Zeilen* , die dargestellt werden sollen, definieren.
- ☒ *Doppelte Zeilen aus der Tabelle entfernen* aktivieren um nur eindeutige Datensätze zu zeigen.
- Activate ☒ *Show only visible features within a map* and select the corresponding *Linked map* whose visible features attributes will be displayed.
- Activate ☒ *Show only features intersecting Atlas feature* is only available when ☒ *Generate an atlas* is activated. When activated it will show a table with only the features which intersect the current atlas feature.
- Aktivieren Sie ☒ *Filtern nach* und schaffen Sie einen Filter indem Sie eine Eingabezeile eingeben oder einen regulären Ausdruck über den gegebenen Ausdrucksknopf einfügen. Einige Beispiele von Filteranweisungen, die Sie verwenden können wenn Sie den airports Layer aus dem Beispieldatensatz geladen haben
  - `ELEV > 500`
  - `NAME = 'ANIAK'`
  - `NAME NOT LIKE 'AN%'`
  - `regexp_match( attribute( $currentfeature, 'USE' ) , '[i]')`

Der letzte reguläre Ausdruck wird nur die Flughäfen einfügen die einen Buchstaben ‚i‘ in der Attributspalte ‚USE‘ haben.

## Darstellung

The *Appearance* group of the attribute table provides the following functionalities (see [Abb. 22.44](#)):



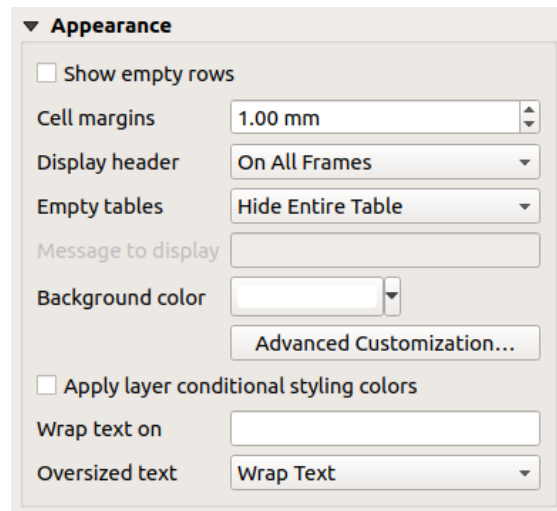




Abb. 22.44: Attribute table appearance Group

- Klicken Sie  *Zeige leere Zeilen* um die Attributtabelle mit leeren Zellen zu füllen, diese Option kann auch verwendet werden um zusätzliche leere Zellen zu schaffen, wenn Sie ein Ergebnis anzeigen wollen!
- Mit *Zellenränder* können Sie die Ränder um den Text in jeder Zelle der Tabelle definieren.
- Mit *Kopf anzeigen* können Sie von einer Liste eine der voreingestellten Optionen ‚Im ersten Rahmen‘, ‚In allen Rahmen‘ oder ‚Kein Kopf‘ auswählen.
- Die Option *Leere Tabellen* kontrolliert was dargestellt wird wenn die Ergebnisauswahl leer ist.
  - **Nur Kopf zeichnen** zeichnet nur den Kopf ausser Sie haben ‚Kein Kopf‘ bei *Kopf anzeigen* ausgewählt.
  - **Ganze Tabelle ausblenden** zeichnet nur den Hintergrund der Tabelle. Sie können  *Hintergrund nicht anzeigen wenn Rahmen leer ist* in *Rahmen* aktivieren um die Tabelle komplett auszublenden.
  - **Eingestellte Nachricht anzeigen** zeichnet den Kopf und fügt eine Zelle ein, die sich über alle Spalten erstreckt und eine Nachricht wie ‚Kein Ergebnis‘ kann in der Option *Anzuzeigende Nachricht* bereitgestellt werden
- Die Option *Anzuzeigende Nachricht* ist nur aktiviert wenn Sie **Eingestellte Nachricht anzeigen** bei *Leere Tabelle* gewählt haben. Die angegebene Nachricht wird in der Tabelle in der ersten Zeile gezeigt wenn das Ergebnis eine leere Tabelle ist.
- With *Background color* you can set the background color of the table using the *color selector* widget. The *Advanced customization* option helps you define different background colors for each cell (see [Abb. 22.45](#))

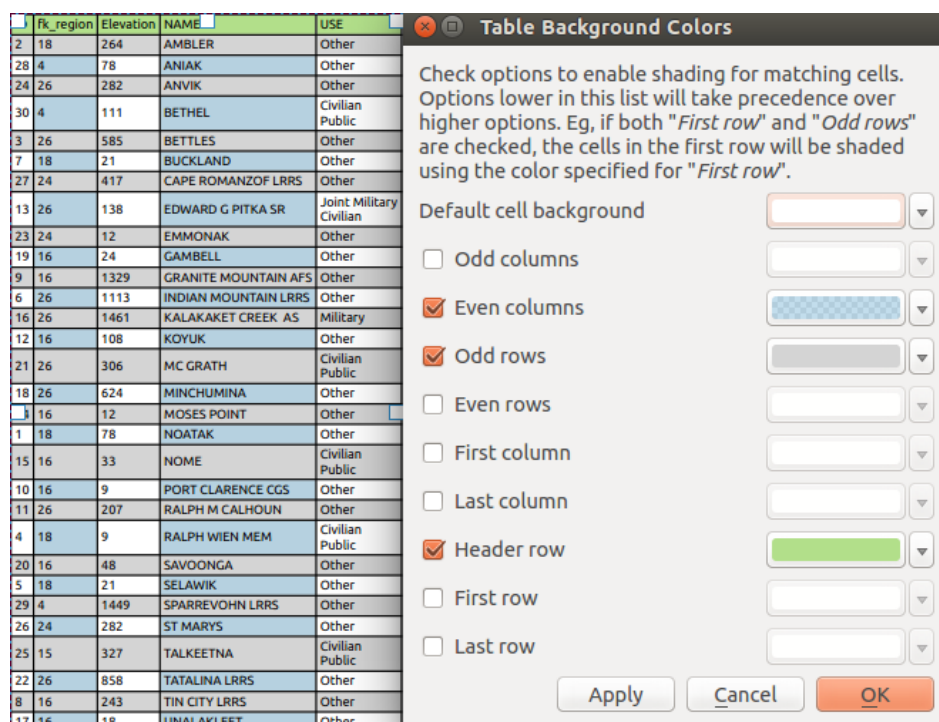


Abb. 22.45: Attribute table Advanced Background Dialog

- **Apply layer conditional styling colors:** the *conditional table formatting* present in the layer is applied inside the layout attribute table (background color, font family and properties such as bold, italic, strikeout, underline, color, ...). Conditional formatting rules take precedence over other layout table formatting settings, e.g. they will override other cell background color settings such as alternating row colors.
- Mit der **:gui-Label: `Wrap Text on`** Option können Sie ein Zeichen definieren, auf dem der Zellinhalt wird eingewickelt wird, jedes Mal, wenn er erfüllt ist
- With *Oversized text* you define the behavior when the width set for a column is smaller than its content's length. It can be **Wrap text** or **Truncate text**.

---

**Bemerkung:** More properties of the attribute table item are described in the *Tables common functionalities* section.

---

## Das Feste Tabellenelement

Additional information about the map can be inserted manually into a table by choosing *Add Fixed Table* and by following *items creation instructions* to add a new table item that you can later manipulate the same way as exposed in *Interacting with layout items*.

By default, an empty table with two minimized columns and rows appears in the map layout. You have to customize the table in the *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities:

## Haupteigenschaften

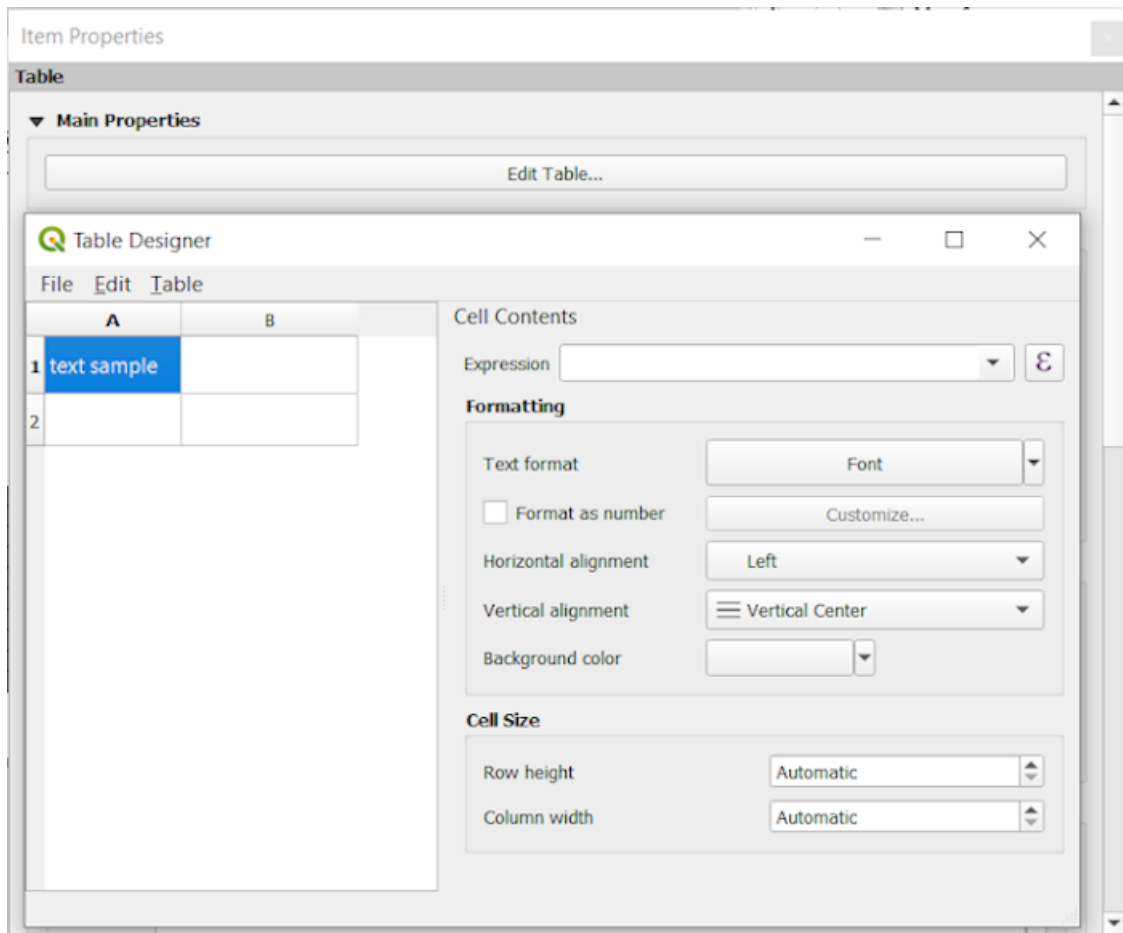





Abb. 22.46: Fixed table Item Properties Panel with Table designer


In *Main properties* you can work with the *Table designer* when clicking the *Edit table ...*:

- You can click into the table and insert texts manually.
- Through the menus on top it is possible to:
  - *Import Content From Clipboard* by going to *File* (it overrides given inputs).
  - work with selection functionalities for rows and columns by going to *Edit*.
  - *Insert rows*, *Insert columns*, *Delete Rows*, *Delete Columns* as well as using the option to  *Include Header Row*.
- You can work with the *Cell Contents* section on the right and:
  - Define the text format of selected cells in *Formatting*
    - \* by clicking on the given  expression button and using a regular expression for the input of the cell
    - \* by choosing the *Text format*
    - \* by  *Format as number* (several formats are available)
    - \* by defining the *Horizontal alignment* and the *Vertical alignment*
    - \* by choosing a *Background color*

- Define the *Cell Size* with *Row height* and *Column width*.

### Darstellung

The *Appearance* group of the fixed table provides the following functionalities:

- Click  *Show empty rows* to fill the attribute table with empty cells.
- Mit *Zellenränder* können Sie die Ränder um den Text in jeder Zelle der Tabelle definieren.
- Mit *Kopf anzeigen* können Sie von einer Liste eine der voreingestellten Optionen ‚Im ersten Rahmen‘, ‚In allen Rahmen‘ oder ‚Kein Kopf‘ auswählen.
- With *Background color* you can set the background color of the table using the *color selector* widget. The *Advanced customization* option helps you define different background colors for each cell.
- With *Oversized text* you define the behavior when the width set for a column is smaller than its content's length. It can be **Wrap text** or **Truncate text**.

---

**Bemerkung:** More properties of the fixed table item are described in the *Tables common functionalities* section.

---

### Tables common functionalities

#### Gitter anzeigen

The *Show grid* group of the table items provides the following functionalities (see [Abb. 22.47](#)):

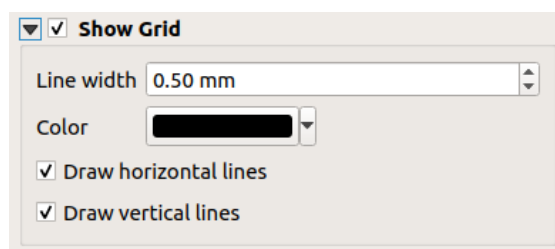



Abb. 22.47: Attribute table Show grid Group

- Activate  *Show grid* when you want to display the grid, the outlines of the table cells. You can also select to either *Draw horizontal lines* or *Draw vertical lines* or both.
- Mit *Strichbreite* können Sie die Dicke der Linien, die im Gitter verwendet werden, festlegen.
- The *Color* of the grid can be set using the color selection widget.

### Schrift- und Textgestaltung

The *Fonts and text styling* group of the table items provides the following functionalities (see [Abb. 22.48](#)):



Abb. 22.48: Attribute table Fonts and text styling Group

- You can define *Font* properties for *Table heading* and *Table contents*, using the advanced *text settings* widget (with buffer, shadow, paint effects, transparency, background, coloring, ...). Note that these changes do not affect the cells that have custom font assigned, either from the *Appearance* section or the *Table Designer* dialog. Only cells with the default rendering are overwritten.
- For *Table heading* you can additionally set the *Alignment* to *Follow column alignment* or override this setting by choosing *Left*, *Center* or *Right*. The column alignment is set using the *Select Attributes* dialog (see Abb. 22.42 ).

## Rahmen

The *Frames* group of the table item properties provides the following functionalities (see Abb. 22.49):

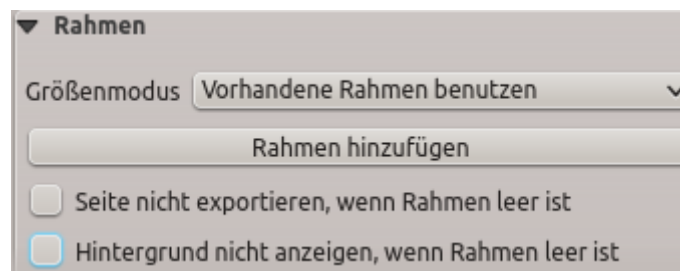


Abb. 22.49: Attribute table Frames Group

- Mit dem *Größenmodus* können Sie auswählen wie der Inhalt der Attributtabelle gerendert wird:
  - *Use existing frames* displays the result in the first frame and added frames only.
  - *Extend to next page* will create as many frames (and corresponding pages) as necessary to display the full selection of attribute table. Each frame can be moved around on the layout. If you resize a frame, the resulting table will be divided up between the other frames. The last frame will be trimmed to fit the table.
  - *Repeat until finished* will also create as many frames as the *Extend to next page* option, except all frames will have the same size.
- Use the *Add Frame* button to add another frame with the same size as selected frame. The result of the table that will not fit in the first frame will continue in the next frame when you use the *Resize mode Use existing frames*.
- Activate ☒ *Don't export page if frame is empty* prevents the page to be exported when the table frame has no contents. This means all other layout items, maps, scalebars, legends etc. will not be visible in the result.
- Das Aktivieren von ☒ *Hintergrund nicht anzeigen wenn Rahmen leer ist* verhindert dass der Hintergrund gezeichnet wird wenn der Rahmen keinen Inhalt hat.


## 22.2.8 The Marker, Picture and North Arrow Items


- *Das Bildelement*
  - *Haupteigenschaften*
  - *Size and placement*
  - *Image rotation*
- *Das Nordpfeilelement*
- *The Marker Item*

Along with the map or legend items in the print layout, you may want to decorate your realization with images or annotations. QGIS provides different tools to achieve this:

- the *picture item*: decorates the layout with an image raster or SVG file (e.g. logos, pictures, north arrows, ...)
- the *north arrow item*: a picture item predefined with a north arrow image
- the *marker item*: decorates the layout with QGIS vector *symbols*. It can be used to place markers over a map item or for creation of advanced custom legends.

### Das Bildelement

You can add a picture by dragging it from your file manager onto the canvas, pasting it directly into the layout by using `Ctrl+V` or *Edit ► Paste* and by using the  Add Picture, following *items creation instructions*. Then you can manipulate it, as explained in *Interacting with layout items*.

When using  Add Picture, the picture item will be a blank frame that you can customize using its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities:

## Haupteigenschaften

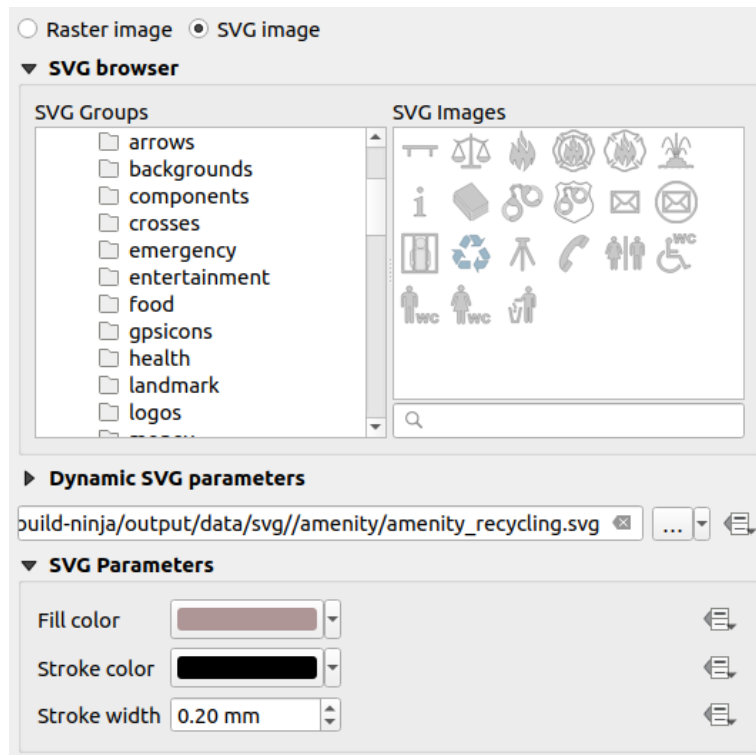



Abb. 22.50: Elementeigenschaften für Bilder

The picture item supports two types of images:

- *Raster Image*: a file selector widget can be used to fetch the data. Use the ... <sup>Browse</sup> button to select a file on your computer or enter the path directly in the text field. You can even provide a remote URL that points to a picture. The associated image can also be *embedded* in the layout.

Benutzen Sie die Schaltfläche  *Datendefinierte Übersteuerung*, um die Bildquelle aus einem Objektattribut zu lesen oder über einen Ausdruck einzustellen.

- *SVG Image*: using by default the SVG libraries provided in *Settings ► Options ► System ► SVG Paths*. You can however use any other file, and the file selection follows the same rules as for the raster image. The SVG parameters can as well be set dynamic.

The QGIS provided (default) .SVG files are customizable, meaning that you can easily apply other *Fill color*, *Stroke color* (including opacity) and *Stroke width* than the original, using their corresponding feature in the *SVG Parameters* group. These properties can also be *data-defined*.

If you add an .SVG file that does not enable these properties, you may need to add the following tags to the file in order to add support e.g. for transparency:

- *fill-opacity*="param(fill-opacity)"
- *stroke-opacity*="param(outline-opacity)"

More details at [Parametrizable SVG](#).

---

**Bemerkung:** Drag-and-drop an image file (raster or SVG) into the layout page will create a layout picture item with corresponding settings.

---

## Size and placement




Abb. 22.51: Layout pictures size and placement properties

Mit der Einstellung *Größenmodus* können Sie festlegen, wie das Bild dargestellt wird, wenn sich die Größe des Rahmens ändert:

- **Zoom:** enlarges/reduces the image to the frame while maintaining the aspect ratio of picture
- **Stretch:** stretches the image to fit inside the frame
- **Clip:** use this mode for raster images only, it sets the size of the image to the original image size without scaling, and the frame is used to clip the image. So only the part of the image that is inside the frame will be visible.
- **Zoom and resize frame:** enlarges the image to fit the frame, and then resizes frame to fit the resulting image dimensions
- **Resize frame to image size:** sets the size of the frame to match the original size of the image (no scaling)

Depending on the selected *Resize mode*, the *Placement* and *Image rotation* options may be disabled. *Placement* lets you select the position of the image inside its frame (top/middle/bottom and left/center/right).

## Image rotation

Images can be rotated with the *Image rotation* field. Activating the  *Sync with map* checkbox synchronizes the rotation of the image with the rotation applied to the selected map item. This is a convenient feature to make any picture behave as a north arrow. The *North alignment* can be:

- **Grid north:** the direction of a grid line which is parallel to the central meridian of the national/local grid
- **True north:** direction of a meridian of longitude.

Sie können der Bilddrehung auch einen *Versatz* für die Missweisung hinzufügen.

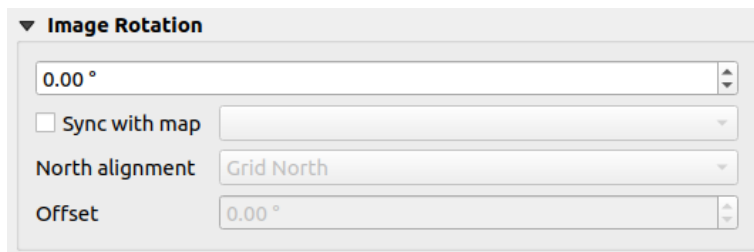


Abb. 22.52: Layout pictures image rotation properties



## Das Nordpfeilelement

You can add a north arrow with the  Add North Arrow button, following *items creation instructions* and manipulate it the same way as exposed in *Interacting with layout items*.

Since north arrows are images, the *North Arrow* item has the same properties as the *picture item*. The main differences are:

- A default north arrow is used when adding the item, instead of a blank frame
- The north arrow item is synced with a map item by default: the *Sync with map* property is the map over which the north arrow item is drawn. If none, it falls back to the *reference map*.

**Bemerkung:** Many of the north arrows do not have an ‚N‘ added in the north arrow. This is done on purpose, since there are languages that do not use an ‚N‘ for North.

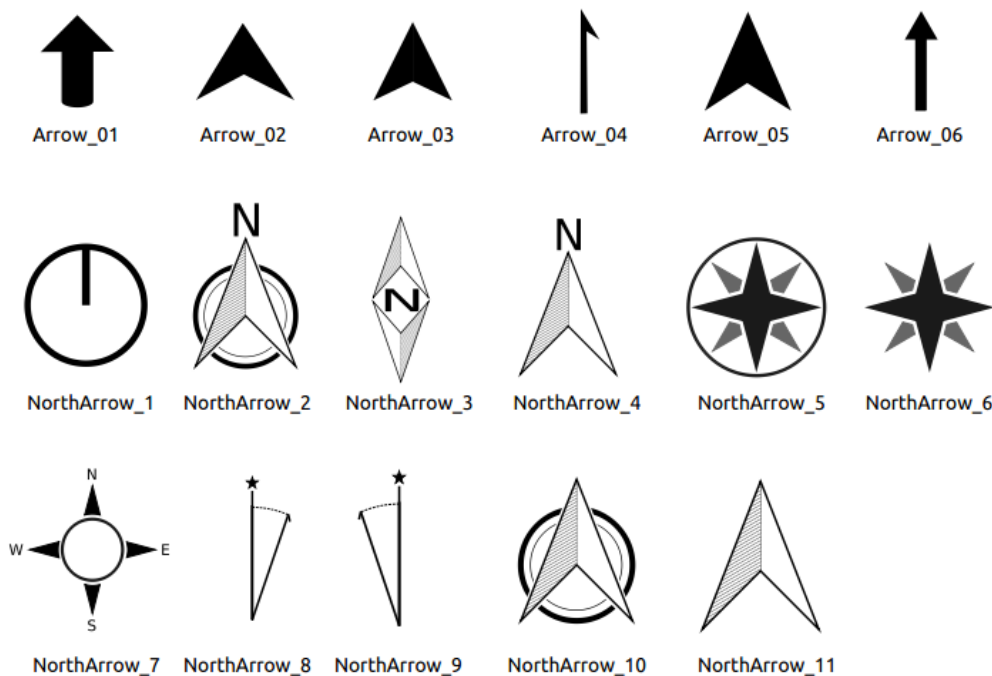



Abb. 22.53: Nordpfeile, die von der SVG-Bibliothek zur Verfügung gestellt werden.

## The Marker Item

To add a marker item, select the  Add Marker button, and click on the page. A default point marker symbol is added. Then you can manipulate it, as explained in *Interacting with layout items*. But note that unlike most of the other items, you resize the item given that its size is controlled by the embedded symbols properties.

The marker item can be customized from the *Item Properties* panel. Other than the *items common properties*, you can also:

- modify the *Symbol*, relying on all the symbol *widget capabilities*
- sync the marker item rotation with the map's (see *Image rotation*), acting as a north arrow. The map rotation is added to any existing marker symbol level rotation (so .e.g if you have to rotate the triangle marker 90° to get it pointing straight up, it will still work nicely in north arrow mode!)

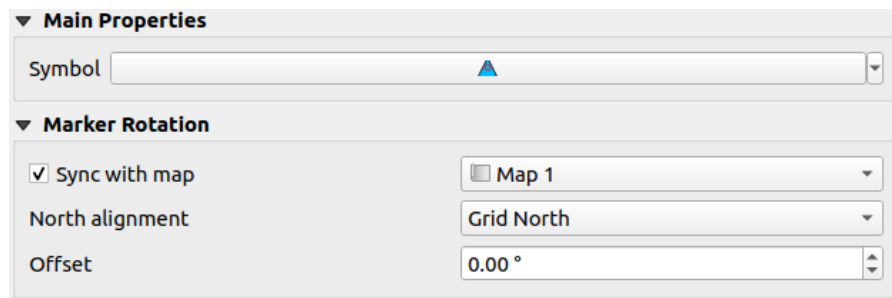





Abb. 22.54: The marker item custom properties

## 22.2.9 The Elevation Profile Item

The Elevation Profile item is used to display an *elevation profile view* in a layout. Use the  **Add Elevation Profile** button, and follow *items creation instructions* to add a new elevation profile item that you can later manipulate the same way as demonstrated in *Interacting with layout items*.

A new elevation profile item has default settings that make it render an empty chart. You can customize its properties in the *Item Properties* panel. In addition to the *common properties*, this feature has the following functionalities:


The Elevation profile *Item Properties* panel embeds a top toolbar with the following functionalities:

-  **Update elevation profile** to refresh the item rendering
-  **Copy from elevation profile**: a drop-down menu to select an elevation profile view from. The view settings are applied to the layout elevation profile item, and can be later modified.

## Layers

Under the *Layers* group, check in the tree view the layers you would like to render in the profile item. Remember to properly configure the *Elevation* properties of the selected layers.

## Profile curve

-  **Controlled by atlas**: the *profile curve* will be taken from the current atlas feature and the elevation profile view updated as you walk through the atlas features. This is currently supported for an active layout atlas or report, using a line geometry type coverage layer.
- The *Tolerance* distance, which can be data-defined, helps you control how far a feature of the visible layers should be from the profile curve in order to display in the layout elevation view. Only point features are currently returned.

## Chart ranges

A layout elevation profile item does not necessarily display the full extent of the elevation profile view it is based on. You can limit the area to render, providing:

- on the X axis, the *Minimum distance* and *Maximum distance* from the profile curve starting point
- On the Y axis, the *Minimum elevation* and *Maximum elevation*

## Distance and elevation axes

The *Distance axis* and *Elevation axis* groups give options to tweak the grid over the elevation profile item, respectively on the X and Y axes:


- the graduation on the axis with both a *Major interval* and *Minor interval*
- the line symbols to apply to the corresponding *Major grid lines* and *Minor grid lines*
- how regular the graduation items should be labeled (*Label interval*) as well as their *Label format* and *Label font*

## Chart area

Under *Chart area*, you can configure the rendering of the area in which the elevation profile plot is actually displayed:

- a *Background* fill symbol
- a *Border* line symbol
- the margins from the elevation profile item border

## 22.2.10 Das HTML-Rahmen Element

It is possible to add a frame that displays the contents of a website or even create and style your own HTML page and display it! You can add a picture with the  *Add HTML* following *items creation instructions* and manipulate it the same way as exposed in *Interacting with layout items*. Note that the HTML scale is controlled by the layout export resolution at the time the HTML frame is created.

The HTML item can be customized using its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see [Abb. 22.55](#)):

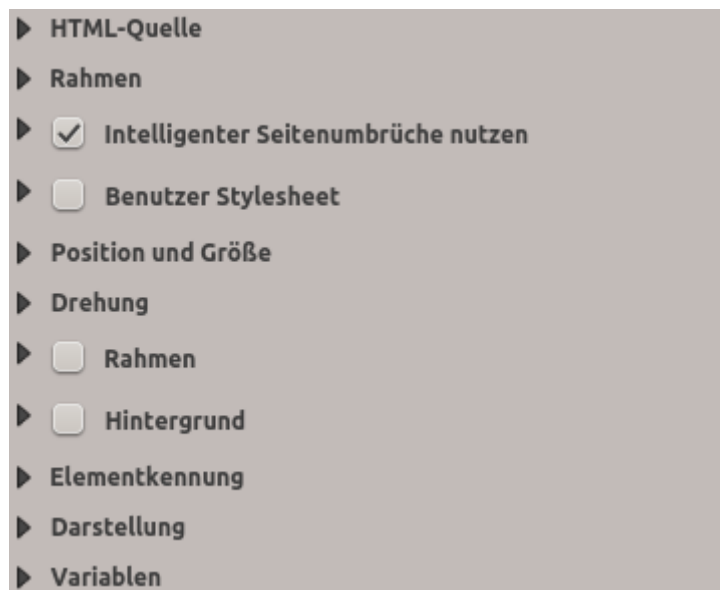


Abb. 22.55: HTML Frame, the Item Properties Panel

## HTML-Quelle

The *HTML Source* group of the HTML frame *Item Properties* panel provides the following functionalities (see [Abb. 22.56](#)):

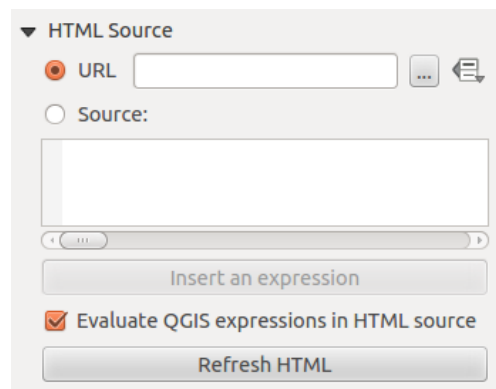




Abb. 22.56: HTML frame, the HTML Source properties

- In *URL* you can enter the URL of a webpage you copied from your Internet browser or select an HTML file using the ... <sup>Browse</sup> button. There is also the option to use the  *Data-defined override* button, to provide a URL from the contents of an attribute field of a table or using a regular expression.
- Unter *Quelle* können Sie Text mit einigen HTML-Tags ins Textfenster eingeben oder eine ganze HTML-Seite zur Verfügung zu stellen.
- The *Insert or Edit an Expression...* button can be used to add an expression like [%Year(\$now) %] in the Source textbox to display the current year. This button is only activated when radiobutton *Source* is selected. After inserting the expression click somewhere in the textbox before refreshing the HTML frame, otherwise you will lose the expression.
- Aktivieren Sie  *QGIS-Ausdrücke in HTML-Quelle auswerten* um das Ergebnis des Ausdrucks, den Sie eingefügt haben, zu sehen, andernfalls sehen Sie den Ausdruck.
- Use the *Refresh HTML* button to refresh the HTML frame(s) and see the result of changes.

## Rahmen

The *Frames* group of the HTML frame *Item Properties* panel provides the following functionalities (see [Abb. 22.57](#)):

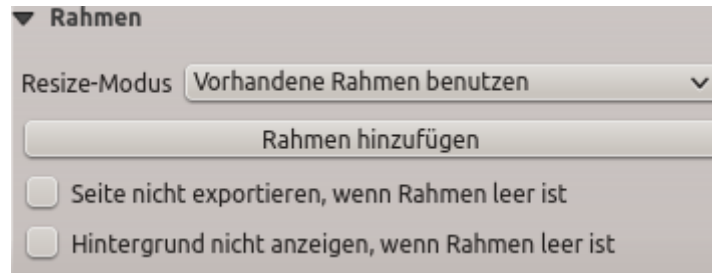


Abb. 22.57: HTML frame, the Frames properties

- Mit dem *Resize-Modus* können Sie auswählen wie der HTML Inhalt gerendert wird:
  - `Use existing frames` displays the result in the first frame and added frames only.
  - `Extend to next page` will create as many frames (and corresponding pages) as necessary to render the height of the web page. Each frame can be moved around on the layout. If you resize a frame, the webpage will be divided up between the other frames. The last frame will be trimmed to fit the web page.
  - `Repeat on every page` will repeat the upper left of the web page on every page in frames of the same size.
  - `Repeat until finished` will also create as many frames as the `Extend to next page` option, except all frames will have the same size.
- Use the *Add Frame* button to add another frame with the same size as selected frame. If the HTML page does not fit in the first frame it will continue in the next frame when you use *Resize mode* or *Use existing frames*.
- Activate ☒ *Don't export page if frame is empty* prevents the page from being exported when the frame has no HTML contents. This means all other layout items, maps, scale bars, legends etc. will not be visible in the result.
- Das Aktivieren von ☒ *Hintergrund nicht anzeigen, wenn Rahmen leer ist* verhindert, dass der HTML-Rahmen gezeichnet wird wenn der Rahmen leer ist.

## Intelligente Seitenumbrüche und Benutzer Stylesheet nutzen

The *Use smart page breaks* dialog and *User style sheet* dialog of the HTML frame *Item Properties* panel provides the following functionalities (see [Abb. 22.58](#)):

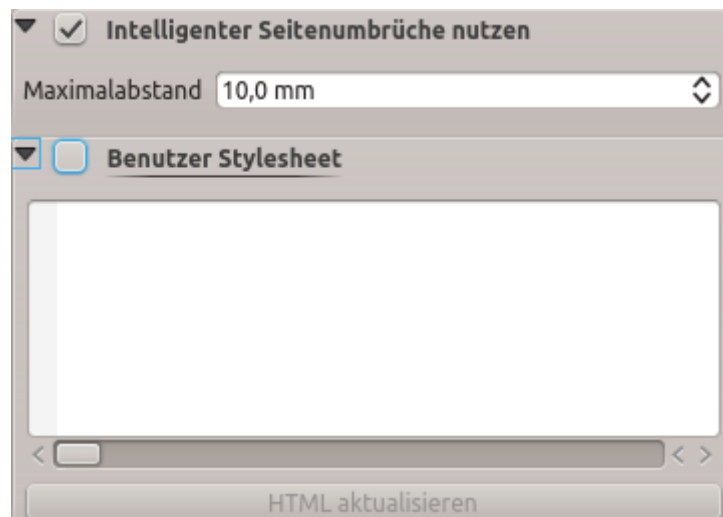


Abb. 22.58: HTML frame, Use smart page breaks and User style sheet properties

- Aktivieren Sie ☒ *Intelligente Seitenumbrüche* um zu verhindern, dass der HTML-Rahmen inmitten einer Textzeile umbricht so dass er weiter schön und glatt im nächsten Rahmen weitergeht.
- Setzen Sie den erlaubten *Maximalabstand* wenn ausgerechnet werden soll wo die Seitenumbrüche im HTML platziert werden sollen. Dieser Abstand ist der maximale Größe des leeren Raums am Fuß eines Rahmens nachdem der optimale Platz für den Seitenumbruch berechnet wurde. Das Setzen eines größeren Wertes resultiert in einer besseren Auswahl des Platzes für den Seitenumbruch, es resultiert aber auch in mehr nicht genutztem Platz am Fuß des Rahmens. Dies wird nur genutzt wenn *Intelligenter Seitenumbrüche nutzen* aktiviert ist.
- Aktivieren Sie ☒ *User style sheet* to apply HTML styles that often is provided in cascading style sheets. An example of style code is provided below to set the color of <h1> header tag to green and set the font and font size of text included in paragraph tags <p>.

```
h1 {color: #00ff00;
}
p {font-family: "Times New Roman", Times, serif;
  font-size: 20px;
}
```





- Use the *Update HTML* button to see the result of the style sheet settings.

## 22.2.11 Das Formelement

QGIS provides a couple of tools to draw regular or more complex shapes over the print layout.

**Bemerkung:** Unlike other print layout items, you can not style the frame nor the background color of the shapes bounding frame (set to transparent by default).

### Das Rechtekelement

The *Shape* item is a tool that helps to decorate your map with regular shapes like triangle, rectangle, ellipse... You can add a regular shape using the  Add Shape tool which gives access to particular tools like  Add Rectangle,  Add Ellipse and  Add Triangle. Once you have selected the appropriate tool, you can draw the item following *items creation instructions*. Like other layout items, a regular shape can be manipulated the same way as exposed in *Interacting with layout items*.

**Bemerkung:** Holding down the **Shift** key while drawing the basic shape with the click and drag method helps you create a perfect square, circle or triangle.

The default shape item can be customized using its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see [Abb. 22.59](#)):

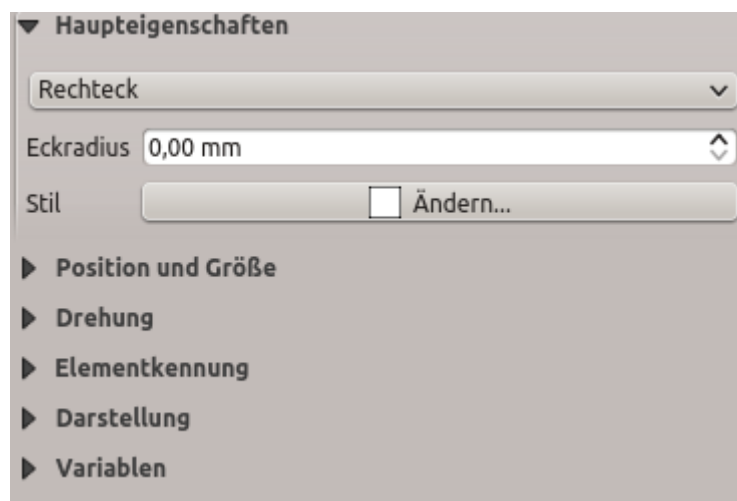





Abb. 22.59: Shape Item Properties Panel

The *Main properties* group shows and allows you to switch the type of the shape item (**Ellipse**, **Rectangle** or **Triangle**) inside the given frame.




You can set the style of the shape using the advanced *symbol* and *color* selector widget...

For the rectangle shape, you can set in different units the value of the *Corner radius* to round of the corners.

## Das Knotenelement

While the  *Add Shape* tool provides way to create simple and predefined geometric item, the  *Add Node Item* tool helps you create a custom and more advanced geometric item. For polylines or polygons, you can draw as many lines or sides as you want and vertices of the items can be independently and directly manipulated using the  *Edit Nodes Item*. The item itself can be manipulated as exposed in *Interacting with layout items*.

To add a node-based shape:

1. Click the  *Add Node Item* icon
2. Select either  *Add Polygon* or  *Add Polyline* tool
3. Perform consecutive left clicks to add nodes of your item. If you hold down the `Shift` key while drawing a segment, it is constrained to follow an orientation multiple of  $45^\circ$ .
4. When you're done, right-click to terminate the shape.

You can customize the appearance of the shape in the *Item Properties* panel.

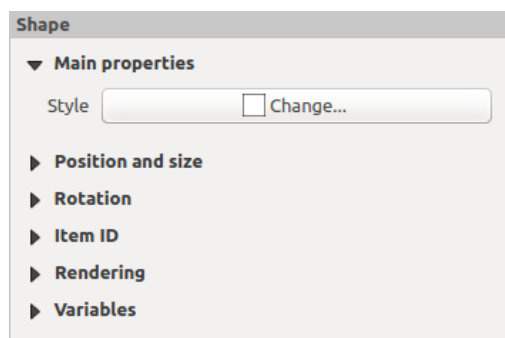


Abb. 22.60: Polygon Node Shape Item Properties Panel

In the *Main properties*, you can set the style of the shape using the advanced *symbol* and *color* selector widget...

For polyline node items, you can also parameterize the *Line markers* i.e. add:

- start and/or end markers with options:
  - *None*: draws a simple polyline.
  - *Arrow*: adds a regular triangular arrow head that you can customize.
  - *SVG marker*: uses an *SVG* file as arrow head of the item.
- customize the arrow head:
  - *Arrow stroke color*: sets the stroke color of the arrow head.
  - *Arrow fill color*: sets the fill color of the arrow head.
  - *Arrow stroke width*: sets the stroke width of the arrow head.
  - *Arrow head width*: sets the size of the arrow head.

SVG images are automatically rotated with the line. Stroke and fill colors of QGIS predefined SVG images can be changed using the corresponding options. Custom SVG may require some tags following this *instruction*.



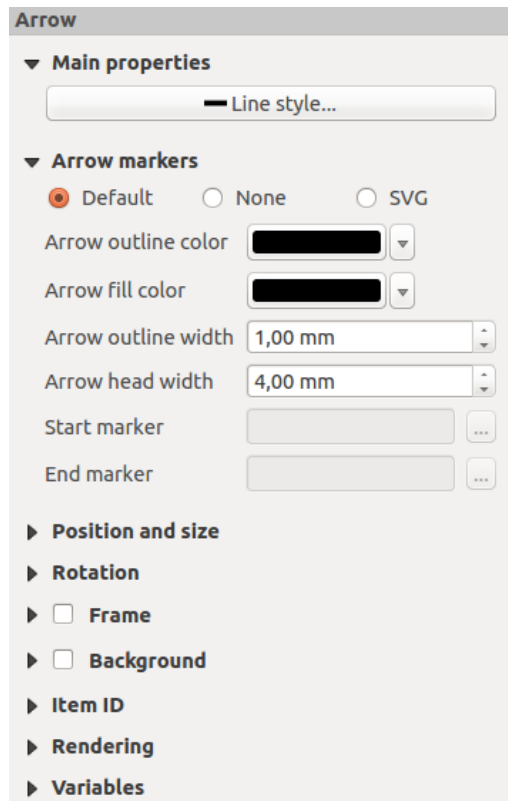




Abb. 22.61: Polyline Node Shape Item Properties Panel

## Das Pfeilelement

The  **Add Arrow** tool is a shortcut to create an arrow-enabled polyline by default and thus has the same properties and behavior as a *polyline node item*.

Actually, the arrow item can be used to add a simple arrow, for example, to show the relation between two different print layout items. However, to create a north arrow, the *image item* should be considered first as it gives access to a set of north arrows in *.SVG* format that you can sync with a map item so that it rotates automatically with it.

## Editing a node item geometry

A specific tool is provided to edit node-based shapes through  **Edit Nodes Item**. Within this mode, you can select a node by clicking on it (a marker is displayed on the selected node). A selected node can be moved either by dragging it or by using the arrow keys. Moreover, in this mode, you are able to add nodes to an existing shape: double-click on a segment and a node is added at the place you click. Finally, you can remove the currently selected node by hitting the **Del** key.

## 22.3 Eine Ausgabe erzeugen

Abb. 22.62 shows an example print layout including all the types of layout items described in the previous section.

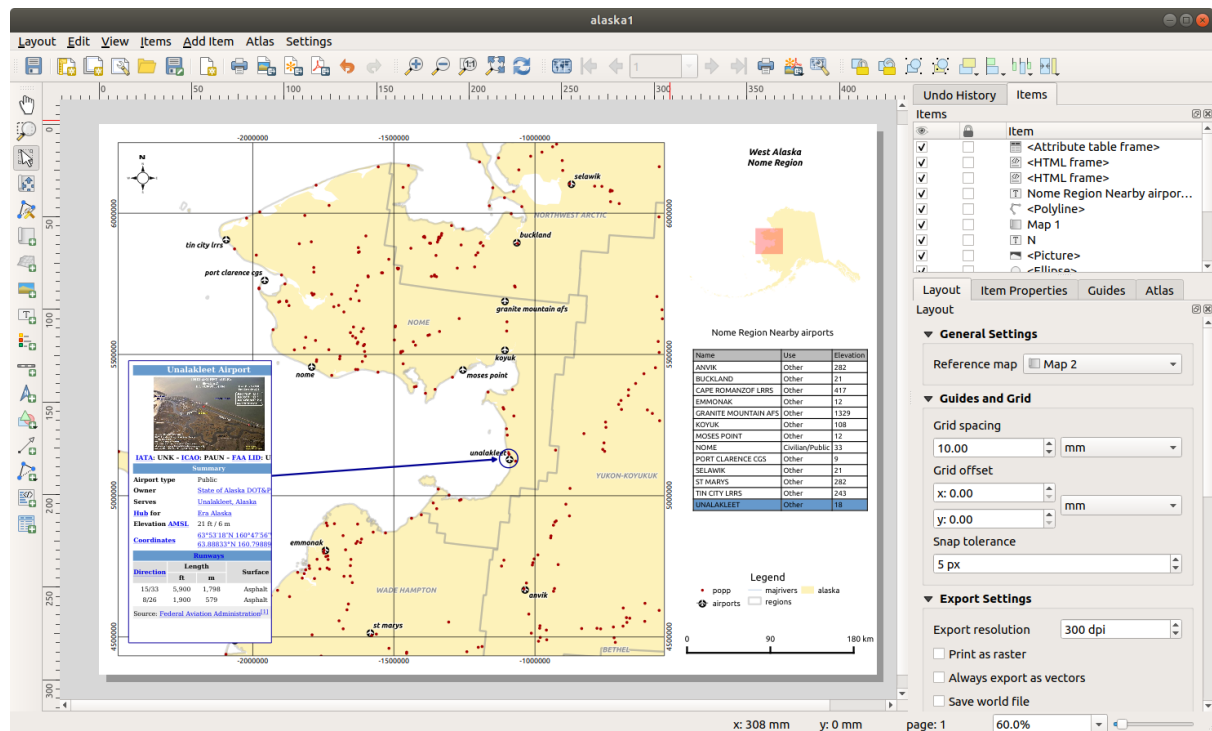






Abb. 22.62: Print Layout with map view, legend, image, scale bar, coordinates, text and HTML frame added

From the *Layout* menu or toolbar, you can output the print layout to different file formats, and it is possible to modify the resolution (print quality) and paper size:

- The  **Print** icon allows you to print the layout to a connected printer or a PostScript file, depending on the installed printer drivers.
- The  **Export as image** icon exports the print layout image formats such as PNG, BMP, TIF, JPG, and many others...
- The  **Export as SVG** icon saves the print layout as an SVG (Scalable Vector Graphic).
- The  **Export as PDF** icon saves the defined print layout directly as a PDF (Portable Document Format) file.

### 22.3.1 Exporteinstellungen


Whenever you export a print layout, there are a selection of export settings QGIS needs to check in order to produce the most appropriate output. These configurations are:

- The *Export settings* of the *Layout* panel, such as *Export resolution*, *Print as raster* *Always export as vectors* or *Save world file*
- *Exclude page from exports* in the *page item properties* panel
- *Exclude item from exports* in the *item properties* panel

Moreover, a number of predefined checks are automatically applied to the layout. Currently these checks include testing that scalebars are correctly linked to map items, and that map overview items are also correctly linked to a map. If the checks fail, you are shown a nice warning advising you of the issue.

## 22.3.2 Speichern als Rasterbild

To export a layout as an image:

1. Click the  Export as image icon
2. Select the image format, the folder and filename (e.g. `myill.png`) to use. If the layout contains more than one page, each page will be exported to a file with the given filename with the page number appended (e.g. `myill_2.png`).
3. In the next (*Image Export Options*) dialog:
  - You can override the print layout *Export resolution* and the exported page dimensions (as set in *Layout* panel).
  - Image rendering can also be improved with the *Enable antialiasing* option.
  - If you want to export your layout as a **georeferenced image** (e.g., to share with other projects), check the ☐ *Generate world file* option, and an *ESRI World File* with the same name as the exported image, but a different extension (`.tifw` for TIFF, `.pnw` for PNG, `.jgw` for JPEG, ...) will be created when exporting. This option can also be checked by default in the *layout panel*.

---

**Bemerkung:** For multi-page output, only the page that contains the *reference map* will get a world file (assuming that the *Generate world file* option is checked).

---

- By checking ☒ *Crop to content* option, the image output by the layout will include the minimal area enclosing all the items (map, legend, scale bar, shapes, label, image...) of each page of the composition:
  - If the composition includes a single page, then the output is resized to include EVERYTHING on the composition. The page can then be reduced or extended to all items depending on their position (on, above, below, left or right of the page).
  - In case of a multi-page layout, each page will be resized to include items in its area (left and right sides for all pages, plus top for the first page and bottom for the last page). Each resized page is exported to a separate file.

The *Crop to content* dialog also lets you add margins around the cropped bounds.

- By checking ☒ *Open file after exporting* the exported file will automatically open in the default image viewer.

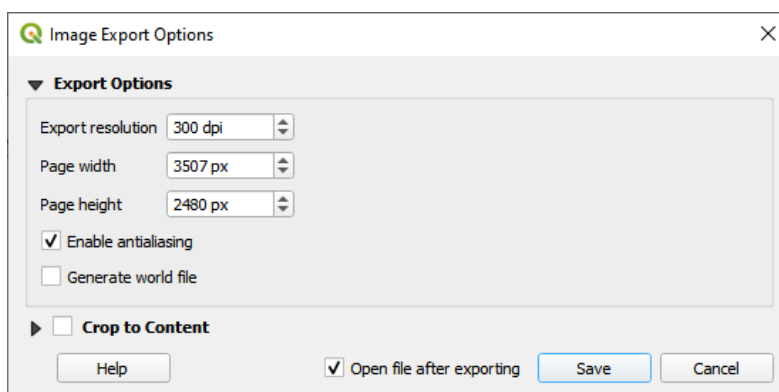


Abb. 22.63: Image Export Options

---

**Tipp:** Use image formats that support transparency when items extend beyond the paper extent

Layout items may be placed outside the paper extent. When exporting with the *Crop to content* option, the resulting image may therefore extend beyond the paper extent. Since the background outside of the paper extent will be

transparent, for image formats that do not support transparency (e.g. BMP and JPG) the transparent background will be rendered as full black, „corrupting“ the image. Use transparency-compatible formats (e.g. TIFF and PNG) in such cases.


---

**Bemerkung:** When supported by the format (e.g. PNG) and the underlying Qt library, the exported image may include *project metadata* (author, title, date, description...)

---

### 22.3.3 Als SVG exportieren

To export a layout as SVG:

1. Click the  Export as SVG icon
2. Fill in the path and filename (used as a base name for all the files in case of multi-page composition, as for image export)
3. In the next *SVG Export Options* dialog, you can override the layout default *export settings* or configure new ones:
  - ☐ *Export map layers as SVG groups*: exported items are grouped within layers whose name matches the layer names from QGIS, making it much easier to understand the contents of the document.
  - ☐ *Always export as vectors*: some rendering options require items to be rasterized for a better rendering. Check this option to keep the objects as vectors with the risk that the appearance of the output file may not match the print layout preview (for more details, see *Exporteinstellungen*).
  - ☒ *Export RDF metadata* of the document such as the title, author, date, description...
  - ☒ *Simplify geometries to reduce output file size*: this avoids exporting ALL geometry vertices, which can result in a ridiculously complex and large export file size that could fail to load in other applications. Geometries will be simplified while exporting the layout in order to remove any redundant vertices which are not discernably different at the export resolution (e.g. if the export resolution is 300 dpi, vertices that are less than 1/600 inch apart will be removed).
  - Set the *Text export*: controls whether text labels are exported as proper text objects (*Always export texts as text objects*) or as paths only (*Always export texts as paths*). If they are exported as text objects, they can be edited in external applications (e.g. Inkscape) as normal text. BUT the side effect is that the rendering quality is reduced, AND there are issues with rendering when certain text settings like buffers are in place. That's why exporting as paths is recommended.
  - Apply ☒ *Crop to content option*
  - ☐ *Disable tiled raster layer exports*: When exporting files, QGIS uses a built-in raster layer tiled rendering that saves memory. Sometimes, this can cause visible „seams“ in the rasters for generated files. Checking this option would fix that, at the cost of a higher memory usage during exports.
  - By checking ☒ *Open file after exporting* the exported file will automatically open in the default SVG viewer.

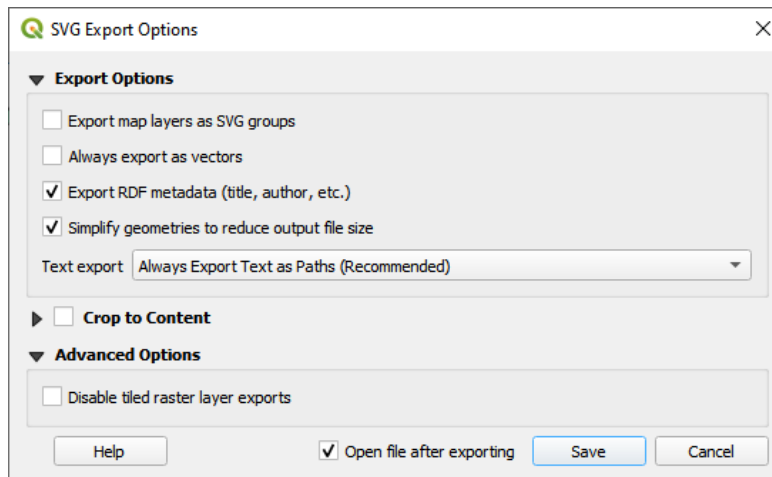





Abb. 22.64: SVG Exportoptionen

**Bemerkung:** Zur Zeit ist die SVG Ausgabe sehr einfach gehalten. Dies ist kein QGIS Problem sondern ein Problem mit der darunterliegenden Qt Bibliothek. Dies wird hoffentlich in zukünftigen Versionen behoben.

### 22.3.4 Als PDF exportieren

To export a layout as PDF:

1. Click the  Export as PDF icon
2. Fill in the path and filename: unlike for image and SVG export, all the pages in the layout are exported to a single PDF file.
3. In the next *PDF Export Options* dialog, you can override the layout default *export settings* or configure new ones:
  - ☐ *Always export as vectors*: some rendering options require items to be rasterized for a better rendering. Check this option to keep the objects as vectors with the risk that the appearance of the output file may not match the print layout preview (for more details, see *Exporteinstellungen*).
  - ☒ *Append georeference information*: available only if the *reference map*, from which the information is taken, is on the first page.
  - ☒ *Export RDF metadata* of the document such as the title, author, date, description...
  - Set the *Text export*: controls whether text labels are exported as proper text objects (*Always export texts as text objects*) or as paths only (*Always export texts as paths*). If they are exported as text objects then they can be edited in external applications (e.g. Inkscape) as normal text. BUT the side effect is that the rendering quality is decreased, AND there are issues with rendering when certain text settings like buffers are in place. That's why exporting as paths is recommended.
  - Control the PDF *Image compression* using:
    - *Lossy (JPEG)*, which is the default compression mode
    - or *Lossless*, which creates bigger files in most cases, but is much more suitable for printing outputs or for post-production in external applications (requires Qt 5.13 or later).
  - ☐ *Create Geospatial PDF (GeoPDF)*: Generate a georeferenced PDF file.
  - ☐ *Disable tiled raster layer exports*: When exporting files, QGIS uses tiled based rendering that saves memory. Sometimes, this can cause visible „seams“ in the rasters for generated files. Checking this option would fix that, at the cost of a higher memory usage during exports.

-  *Simplify geometries to reduce output file size:* Geometries will be simplified while exporting the layout by removing vertices that are not discernably different at the export resolution (e.g. if the export resolution is 300 dpi, vertices that are less than 1/600 inch apart will be removed). This can reduce the size and complexity of the export file (very large files can fail to load in other applications).
- By checking  *Open file after exporting* the exported file will automatically open in the default PDF viewer.

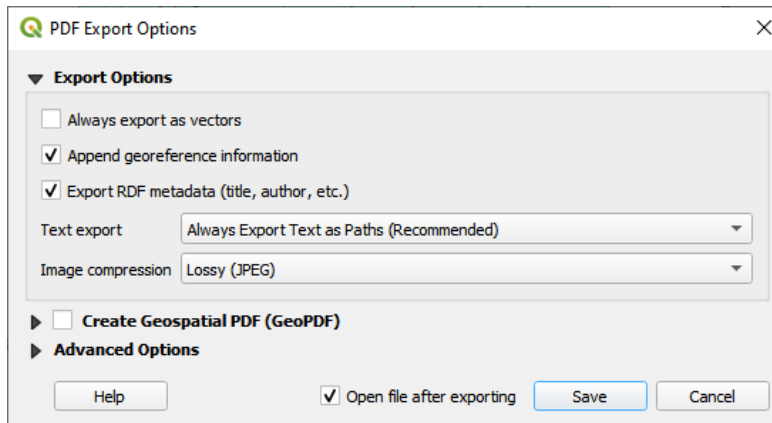


Abb. 22.65: PDF Export Options

---

**Bemerkung:** GeoPDF export is supported, and a number of GeoPDF specific options are available:

- *Format* (GeoPDF format - there are some GeoPDF variations),
  - *Include multiple map themes* (specify map themes to include),
  - *Include vector feature information* (choose the layers and group them into logical PDF groups).
- 

---

**Bemerkung:** Exporting a print layout to formats that supports georeferencing (e.g. PDF and TIFF) creates a georeferenced output by default.

---

### 22.3.5 Einen Atlas erzeugen

Atlas functions allow you to create map books in an automated way. Atlas uses the features of a table or vector layer (*Coverage layer*) to create an output for each feature (**atlas feature**) in the table / layer. The most common usage is to zoom a map item to the current atlas feature. Further use cases include:

- a map item showing, for another layer, only features that share the same attribute as the atlas feature or are within its geometry.
- a label or HTML item whose text is replaced as features are iterated over
- a table item showing attributes of associated *parent or children* features of the current atlas feature...

For each feature, the output is processed for all pages and items according to their exports settings.

---

**Tipp:** Use variables for more flexibility

QGIS provides a large panel of functions and *variables*, including atlas related ones, that you can use to manipulate the layout items, but also the symbology of the layers, according to atlas status. Combining these features gives you a lot of flexibility and helps you easily produce advanced maps.

---

To enable the generation of an atlas and access atlas parameters, refer to the *Atlas* panel. This panel contains the following (see Abb. 22.66):

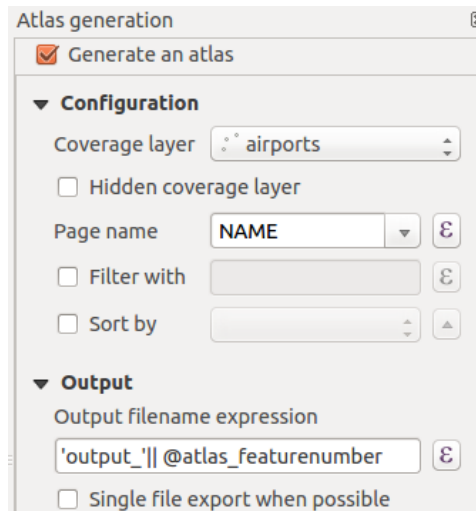









Abb. 22.66: Atlas Panel

-  *Generate an atlas* enables or disables atlas generation.
- *Configuration*
  - A *Coverage layer*  combo box that allows you to choose the table or vector layer containing the features to iterate over.
  - An optional  *Hidden coverage layer* that, if checked, will hide the coverage layer (but not the other layers) during the generation.
  - An optional *Page name* combo box to specify the name for the feature page(s). You can select a field of the coverage layer or set an *expression*. If this option is empty, QGIS will use an internal ID, according to the filter and/or the sort order applied to the layer.
  - An optional  *Filter with* text area that allows you to specify an expression for filtering features from the coverage layer. If the expression is not empty, only features that evaluate to `True` will be processed.
  - An optional  *Sort by* that allows you to sort features of the coverage layer (and the output), using a field of the coverage layer or an expression. The sort order (either ascending or descending) is set by the two-state *Sort direction* button that displays an up or a down arrow.
- *Output* - this is where the output of the atlas can be configured:
  - An *Output filename expression* textbox that is used to generate a filename for each atlas feature. It is based on expressions, is meaningful only for rendering to multiple files.
  - A  *Single file export when possible* that allows you to force the generation of a single file if this is possible with the chosen output format (PDF, for instance). If this field is checked, the value of the *Output filename expression* field is meaningless.
  - An *Image export format* drop-down list to select the output format when using the  *Export atlas as Images...* button.

### Control map by atlas

The most common usage of atlas is with the map item, zooming to the current atlas feature, as iteration goes over the coverage layer. This behavior is set in the *Controlled by atlas* group properties of the map item. See *Controlled by atlas* for different settings you can apply on the map item.

### Customize labels with expression

In order to adapt labels to the feature the atlas iterates over, you can include expressions. Make sure that you place the expression part (including functions, fields or variables) between [% and %] (see *Das Beschriftungselement* for more details).

For example, for a city layer with fields CITY\_NAME and ZIPCODE, you could insert this:

```
The area of [% concat( upper(CITY_NAME), ', ', ZIPCODE, ' is ',
format_number($area/1000000, 2) ) %] km2
```


oder, andere Kombinationen:

```
The area of [% upper(CITY_NAME)%], [%ZIPCODE%] is
[%format_number($area/1000000,2) %] km2
```


The information [% concat( upper(CITY\_NAME), ', ', ZIPCODE, ' is ', format\_number(\$area/1000000, 2) ) %] is an expression used inside the label. Both expressions would result in the following type of label in the generated atlas:


```
The area of PARIS,75001 is 1.94 km2
```

### Explore Data-defined override buttons with atlas


There are several places where you can use a  Data defined override button to override the selected setting. This is particularly useful with atlas generation. See *Datendefinierte Übersteuerung Setup* for more details on this widget.

For the following examples the Regions layer of the QGIS sample dataset is used and selected as *Coverage layer* for the atlas generation. We assume that it is a single page layout containing a map item and a label item.

When the height (north-south) of a region extent is greater than its width (east-west), you should use *Portrait* instead of *Landscape* orientation to optimize the use of paper. With a  Data Defined Override button you can dynamically set the paper orientation.

Right-click on the page and select *Page Properties* to open the panel. We want to set the orientation dynamically, using an expression depending on the region geometry, so press the  button of field *Orientation*, select *Edit...* to open the *Expression string builder* dialog and enter the following expression:

```
CASE WHEN bounds_width(@atlas_geometry) > bounds_height(@atlas_geometry)
THEN 'Landscape' ELSE 'Portrait' END
```

Now if you *preview the atlas*, the paper orients itself automatically, but item placements may not be ideal. For each Region you need to reposition the location of the layout items as well. For the map item you can use the  button of its *Width* property to set it dynamic using the following expression:


```
@layout_pagewidth - 20
```

Likewise, use the  button of the *Height* property to provide the following expression to constrain map item size:

```
@layout_pageheight - 20
```



To ensure the map item is centered in the page, set its *Reference point* to the upper left radio button and enter 10 for its *X* and *Y* positions.

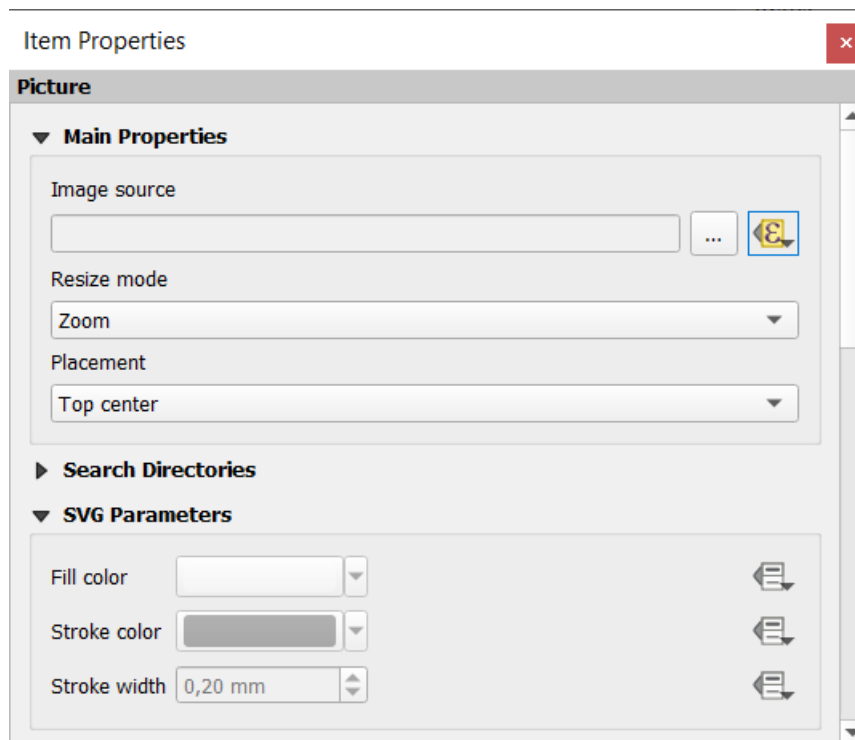
Let's add a title above the map in the center of the page. Select the label item and set the horizontal alignment to  *Center*. Next move the label to the right position, choose the middle button for the *Reference point*, and provide the following expression for field *X*:

```
@layout_pagewidth / 2
```

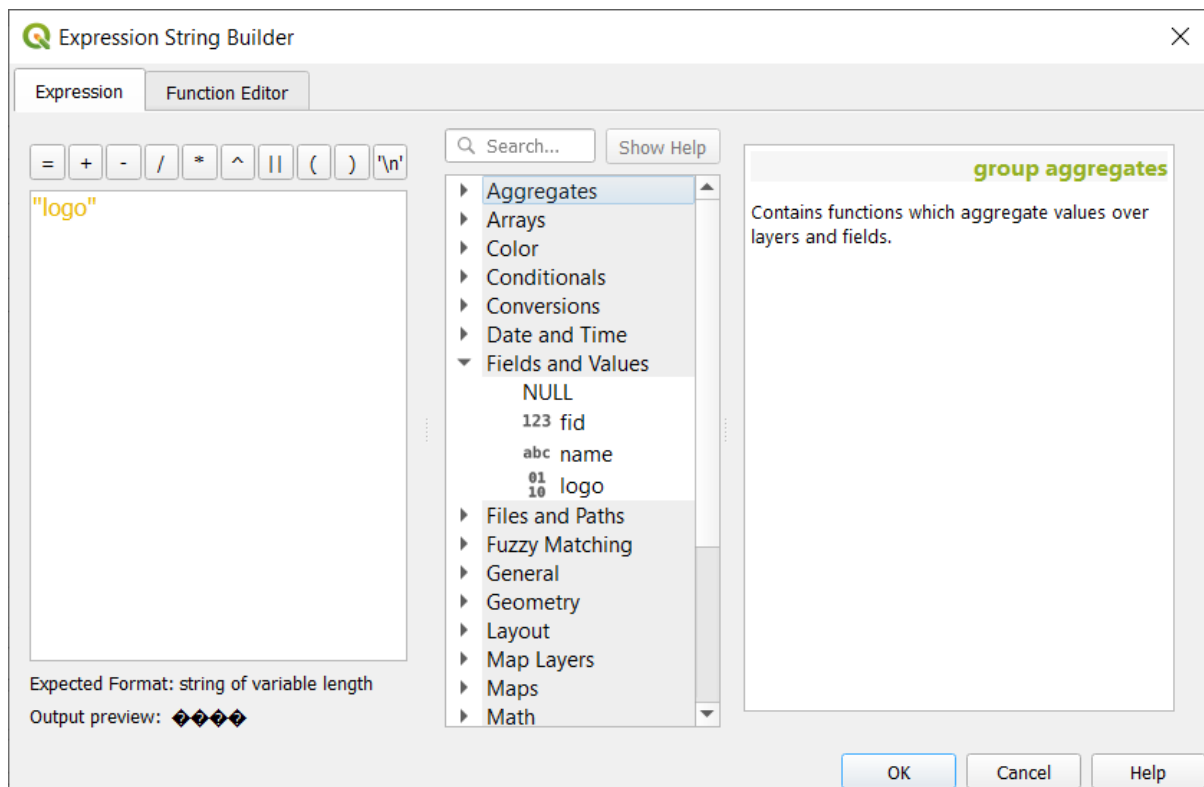
For all other layout items you can set the position in a similar way so they are correctly positioned both for portrait and landscape. You can also do more tweaks such as customizing the title with feature attributes (see [Customize labels with expression](#) example), changing images, resizing the number of legend columns number according to page orientation, ...

The information provided here is an update of the excellent blog (in English and Portuguese) on the Data Defined Override options [Multiple\\_format\\_map\\_series\\_using\\_QGIS\\_2.6](#) .

Another example for using data-defined override buttons is the usage of a dynamic picture. For the following examples we use a geopackage layer containing a BLOB field called `logo` with the field type binary (see [Einen neuen GeoPackage-Layer erstellen](#)). For every feature there is defined a different picture so that the atlas can iterate over as described in [Preview and generate an atlas](#). All you need to do is add a picture in the print layout and go to its *Item properties* in the atlas context. There you can find a data-defined override button in the *Image source* section of the *Main Properties*.



In the following window choose *Edit* so that the *Expression String Builder* opens. From the *Fields and values* section you can find the BLOB field that was defined in the geopackage layer. Double-click the field name `logo` and click *OK*.



The atlas iterates over the entries in the BLOB field provided that you choose the geopackage layer as *Coverage layer* (further instructions you can find in [Preview and generate an atlas](#)).

These are just two examples of how you can use some advanced settings with atlas.

### Preview and generate an atlas

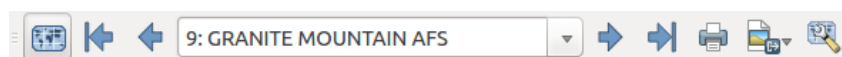








Abb. 22.67: Atlas-Voransichtwerkzeuge

Once the atlas settings have been configured, and layout items (map, table, image...) linked to it, you can create a preview of all the pages by choosing *Atlas ► Preview Atlas* or clicking the  *Preview Atlas* icon. You can then use the arrows to navigate through all the features:

-  Erstes Objekt
-  Vorheriges Objekt
-  Nächstes Objekt
-  Letztes Objekt

You can also use the combo box to select and preview a specific feature. The combo box shows atlas feature names according to the expression set in the atlas *Page name* option.

As for simple compositions, an atlas can be generated in different ways (see [Eine Ausgabe erzeugen](#) for more information - just use tools from the *Atlas* menu or toolbar instead of the *Layout* menu.

This means that you can directly print your compositions with *Atlas ► Print Atlas*. You can also create a PDF using *Atlas ► Export Atlas as PDF...*: You will be asked for a directory to save all the generated PDF files, except if the  *Single file export when possible* has been selected. In that case, you'll be prompted to give a filename.

With *Atlas ► Export Atlas as Images...* or *Atlas ► Export Atlas as SVG...* tool, you're also prompted to select a folder. Each page of each atlas feature composition is exported to the image file format set in *Atlas* panel or to SVG.

---

**Bemerkung:** With multi-page output, an atlas behaves like a layout in that only the page that contains the *General settings* will get a world file (for each feature output).

---



---

### Tipp: Drucken eines bestimmten Atlasobjektes

If you want to print or export the composition of only one feature of the atlas, simply start the preview, select the desired feature in the drop-down list and click on *Layout ► Print* (or *Export...* to any supported file format).

---

## Use project defined relations for atlas creation

For users with HTML and Javascript knowledge it is possible to operate on GeoJSON objects and use project defined relations from the QGIS project. The difference between this approach and using expressions directly inserted into the HTML is that it gives you a full, unstructured GeoJSON feature to work with. This means that you can use existing Javascript libraries and functions that operate on GeoJSON feature representations.

The following code includes all related child features from the defined relation. Using the JavaScript `setFeature` function it allows you to make flexible HTML which represents relations in whatever format you like (lists, tables, etc). In the code sample, we create a dynamic bullet list of the related child features.

```
// Declare the two HTML div elements we will use for the parent feature id
// and information about the children
<div id="parent"></div>
<div id="my_children"></div>

<script type="text/javascript">
  function setFeature(feature)
  {
    // Show the parent feature's identifier (using its "ID" field)
    document.getElementById('parent').innerHTML = feature.properties.ID;
    //clear the existing relation contents
    document.getElementById('my_children').innerHTML = '';
    feature.properties.my_relation.forEach(function(child_feature) {
      // for each related child feature, create a list element
      // with the feature's name (using its "NAME" field)
      var node = document.createElement("li");
      node.appendChild(document.createTextNode(child_feature.NAME));
      document.getElementById('my_children').appendChild(node);
    });
  }
</script>
```

During atlas creation there will be an iteration over the coverage layer containing the parent features. On each page, you will see a bullet list of the related child features following the parent's identifier.

## 22.4 Creating a Report

This section will help you set up a report in QGIS.

### 22.4.1 What is it?

By definition, a GIS report is a document containing information organized in a narrative way, containing maps, text, graphics, tables, etc. A report can be prepared ad hoc, periodic, recurring, regular, or as required. Reports may refer to specific periods, events, occurrences, subjects or locations.

In QGIS, a *Report* is an extension of a *Layouts*.

Reports allow users to output their GIS projects in a simple, quick and structured way.

A report can be created with *Project ► New Report* or inside the *Project ► Layout Manager*.

---

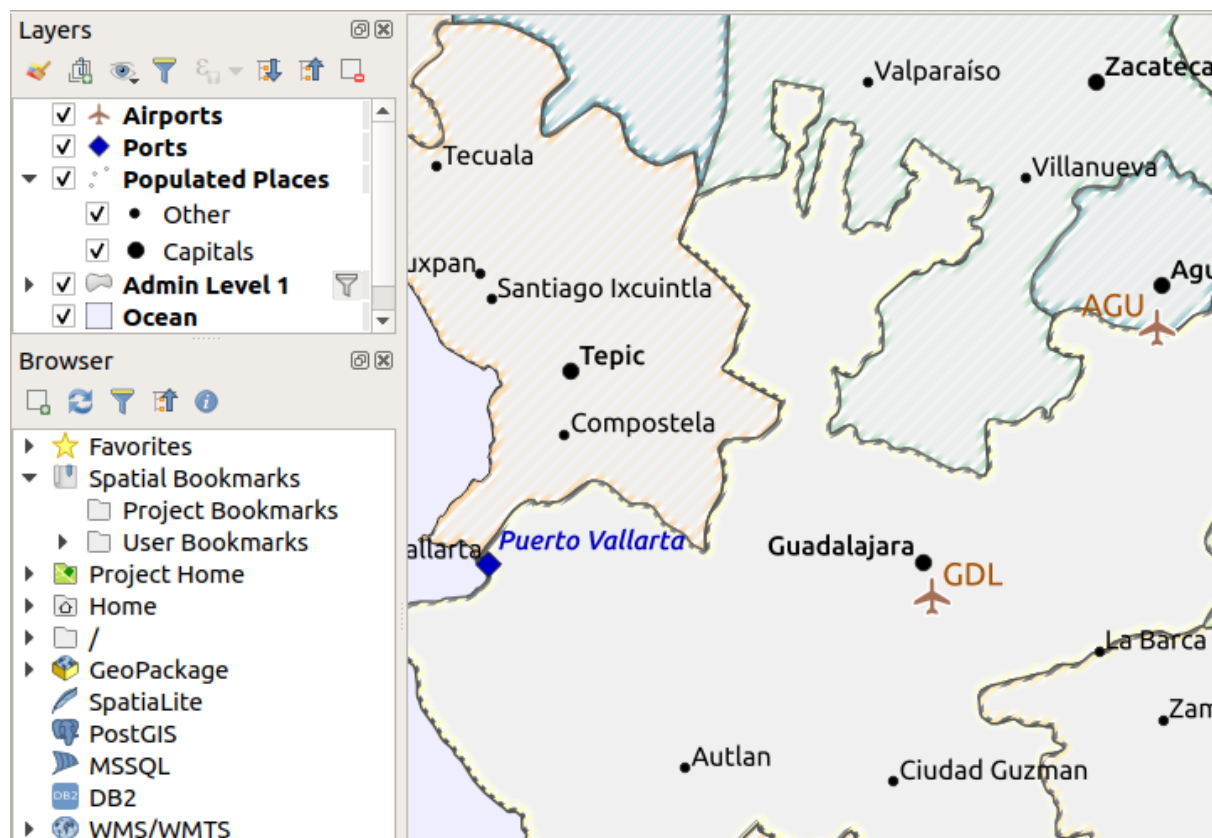
**Bemerkung:** The maps in QGIS reports behave in the same way as maps in print layouts and atlases. We will concentrate on the specifics of QGIS reports. For details on map handling, see the sections on [print layouts](#) and [atlases](#).

---

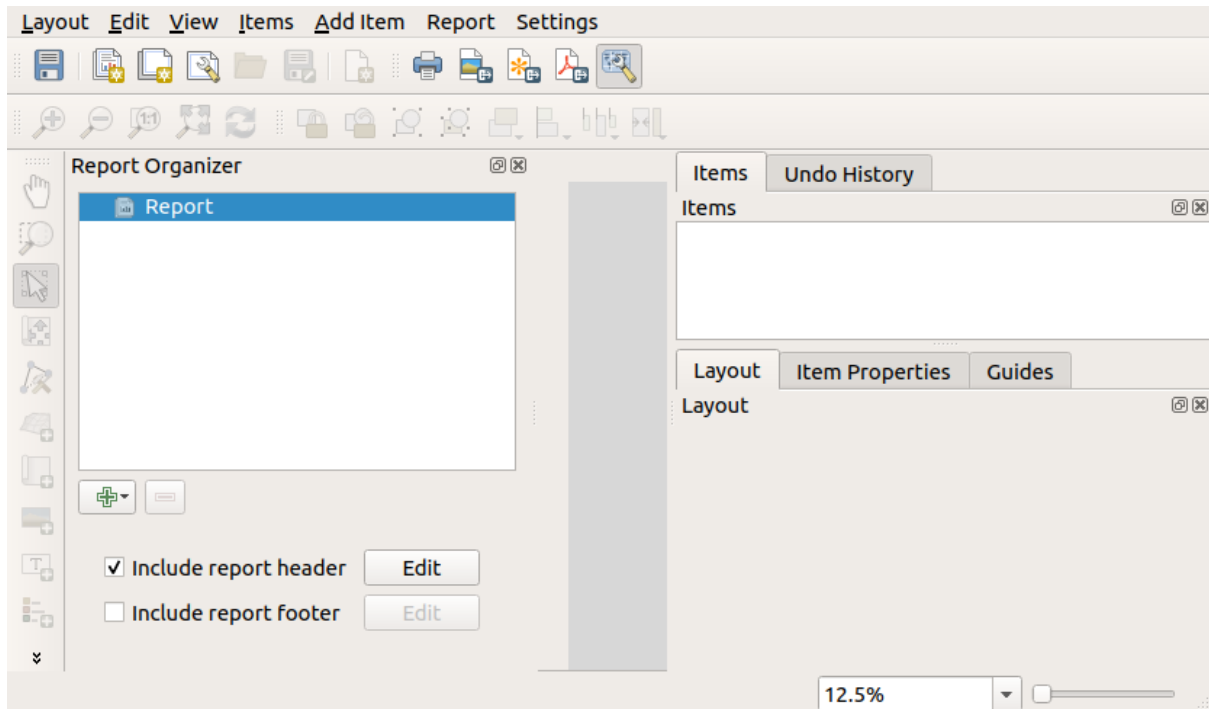
### 22.4.2 Get started

In the *Layout Manager* dialog a report can be created through *New from template* by selecting the dropdown option *Empty Report* and hitting the *Create...* button.

For this example, we use some administrative boundaries, populated places, ports and airports from the [Natural Earth dataset](#) (1:10M).

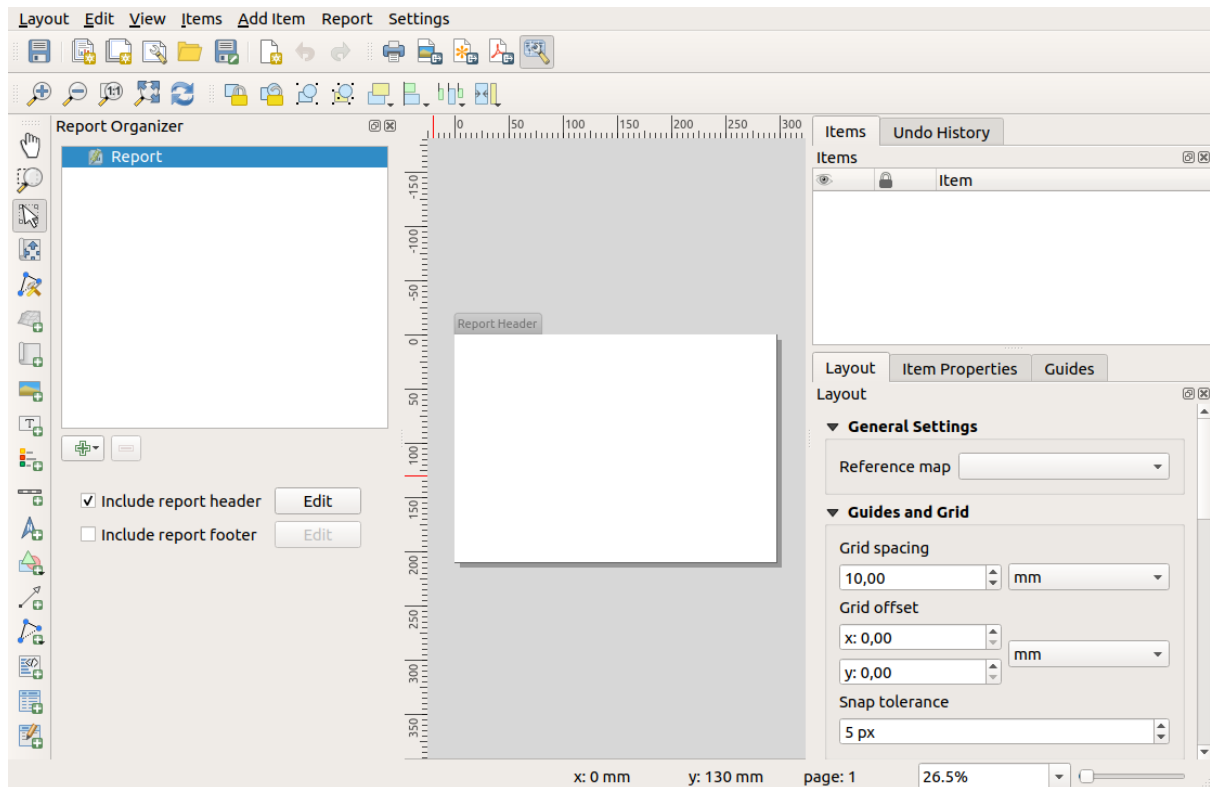


Using the *Project ► New Report* command, we create a blank report. Initially, there is not much to look at – the dialog which is displayed looks much like the print layout designer, except for the *Report Organizer* panel to the left:



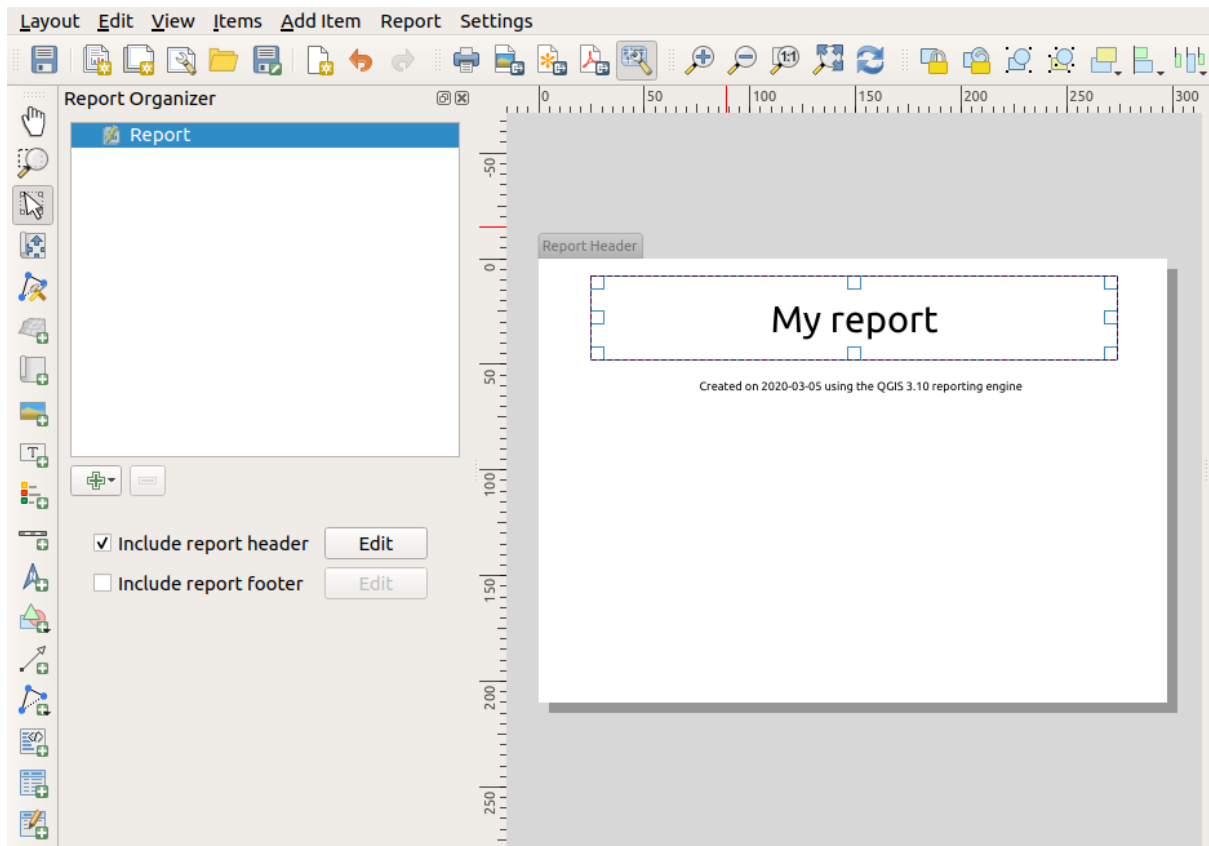
### 22.4.3 Layout Report Workspace

QGIS reports can consist of multiple, nested sections. In our new blank report we initially only have the main report section. The only options for this report section is *Include report header* and *Include report footer*. If we enable these options, a header will be included as the first page(s) (individual parts of reports can be multi-page if desired) in the report, and a footer will constitute the last page(s). Enable the header (*Include report header*), and hit the *Edit* button next to it:

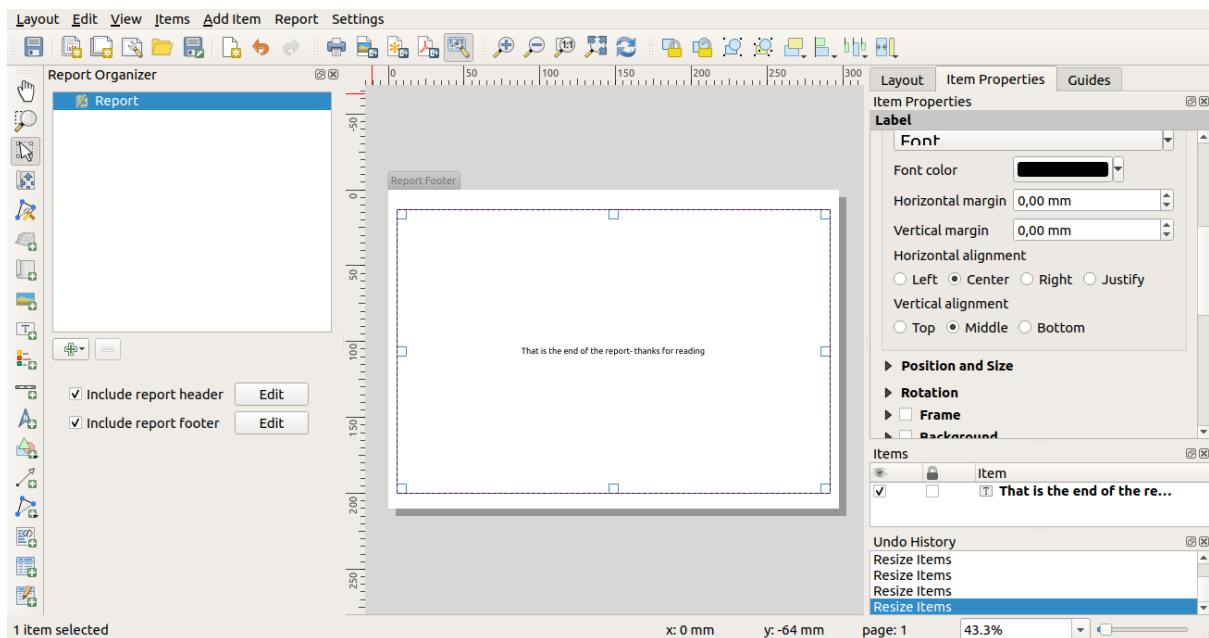


A few things happen as a result. Firstly, an edit pencil is shown next to *Report* in the *Report Organizer*, indicating that the report section is currently being edited in the designer. We also see a new page with a small *Report Header* title. The page has *landscape* orientation by default, but this (and other properties of the page) can be changed by right-clicking on the page and choosing *Page properties*. This will bring up the *Item properties* tab for the page, and page *Size*, *Width*, *Height*, and more can be specified.

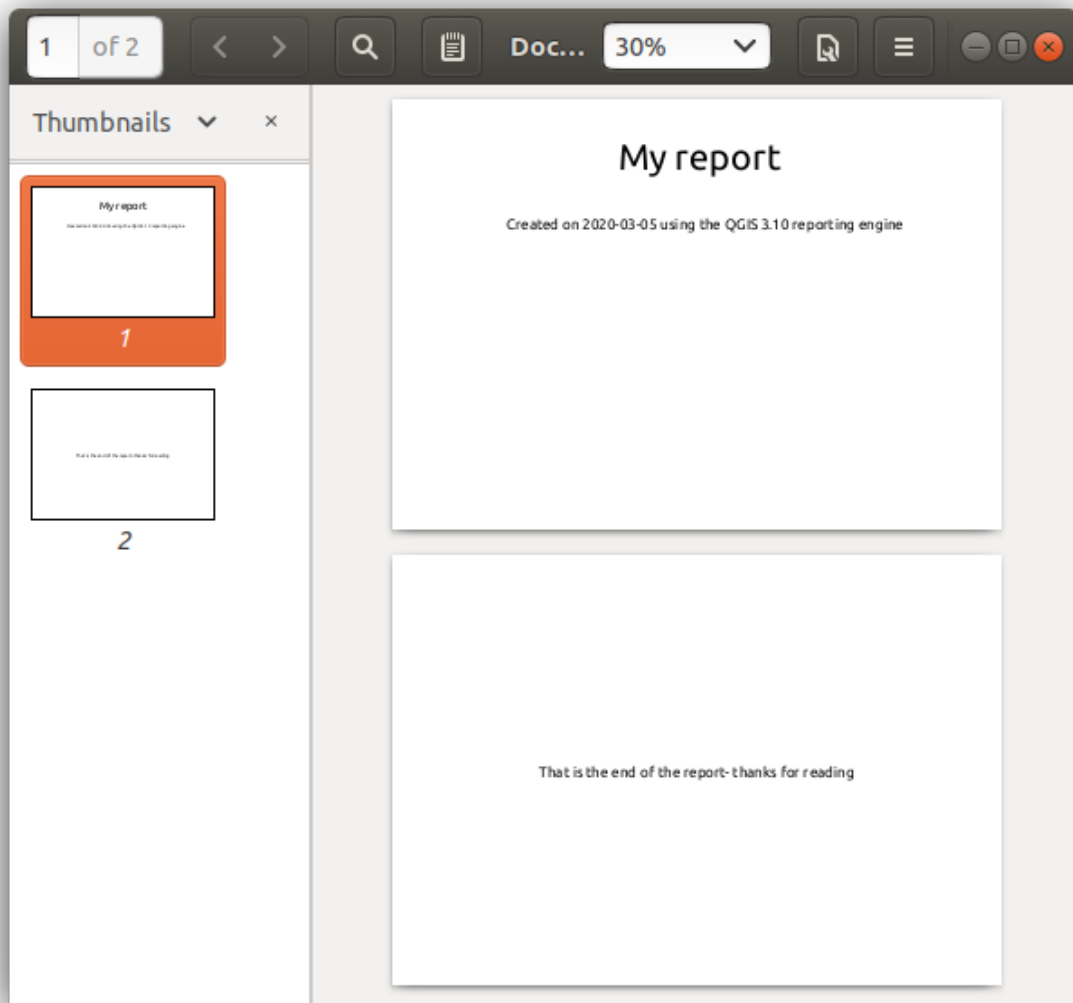
In QGIS reports, every component of the report is made up of individual layouts. They can be created and modified using the same tools as for standard print layouts – so you can use any desired combination of labels, pictures, maps, tables, etc. Let us add some items to our report header to demonstrate:



We will also create a simple footer for the report by checking the *Include report footer* option and hitting *Edit*.

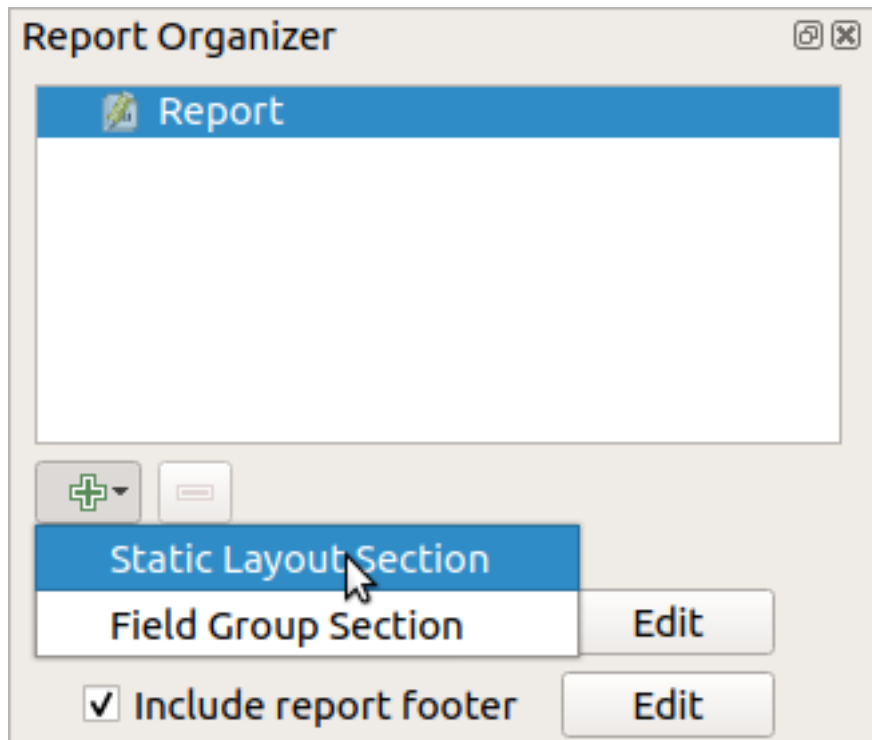


Before proceeding further, let us export this report and see what we get. Exporting is done from the *Report* menu – in this case we select *Export Report as PDF...* to render the whole report to a PDF file. Here is the not-very-impressive result – a two page PDF consisting of our header and footer:



Let us make things more interesting. By hitting the  **Add Section** button in the *Report Organizer*, we are given a choice of new sections to add to our report.



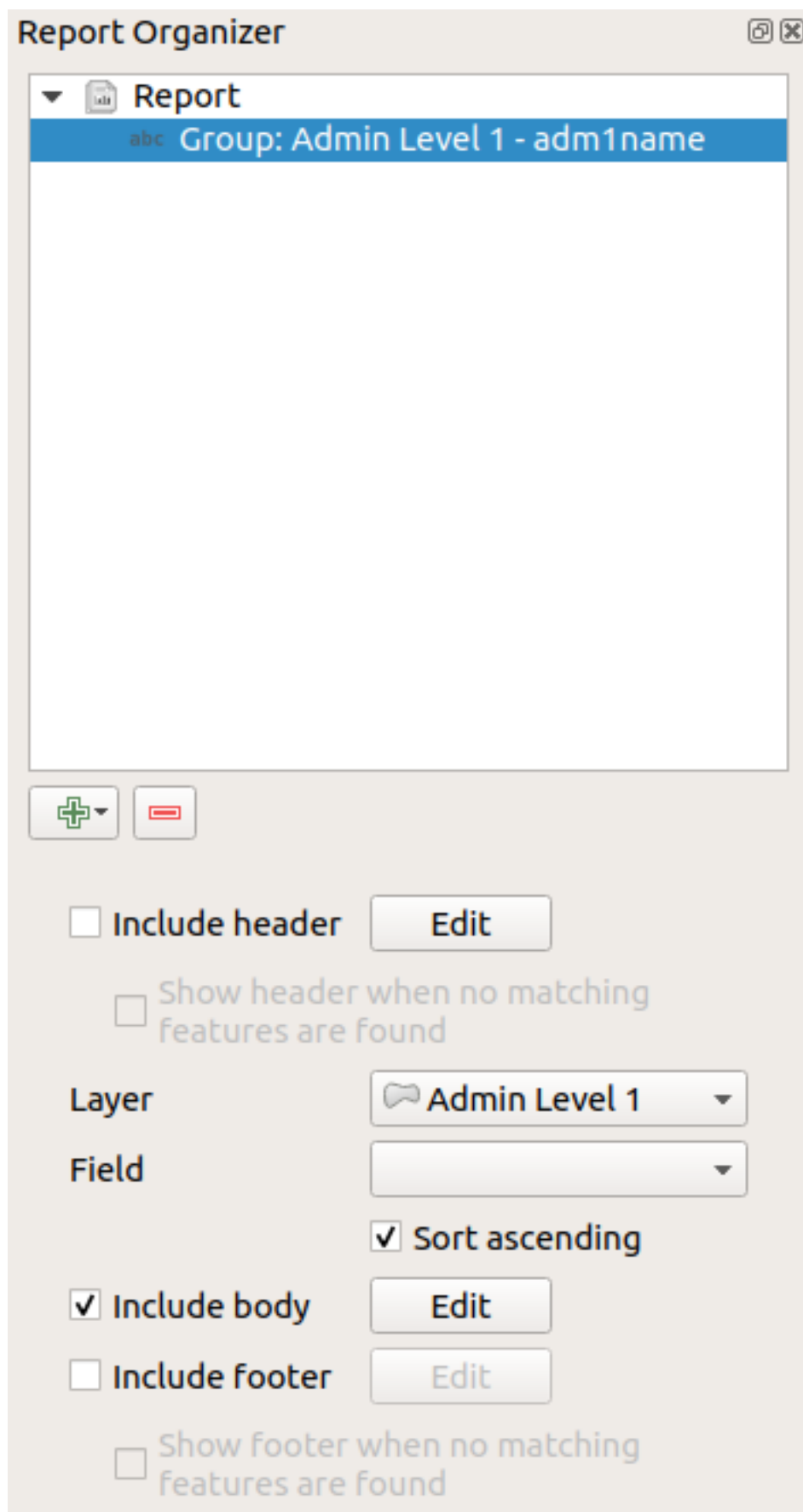


There are two options: *Static Layout Section* and *Field Group Section*.

The *Add Static Layout Section* is a single, static body layout. This can be used to embed static layouts mid-way through a report.

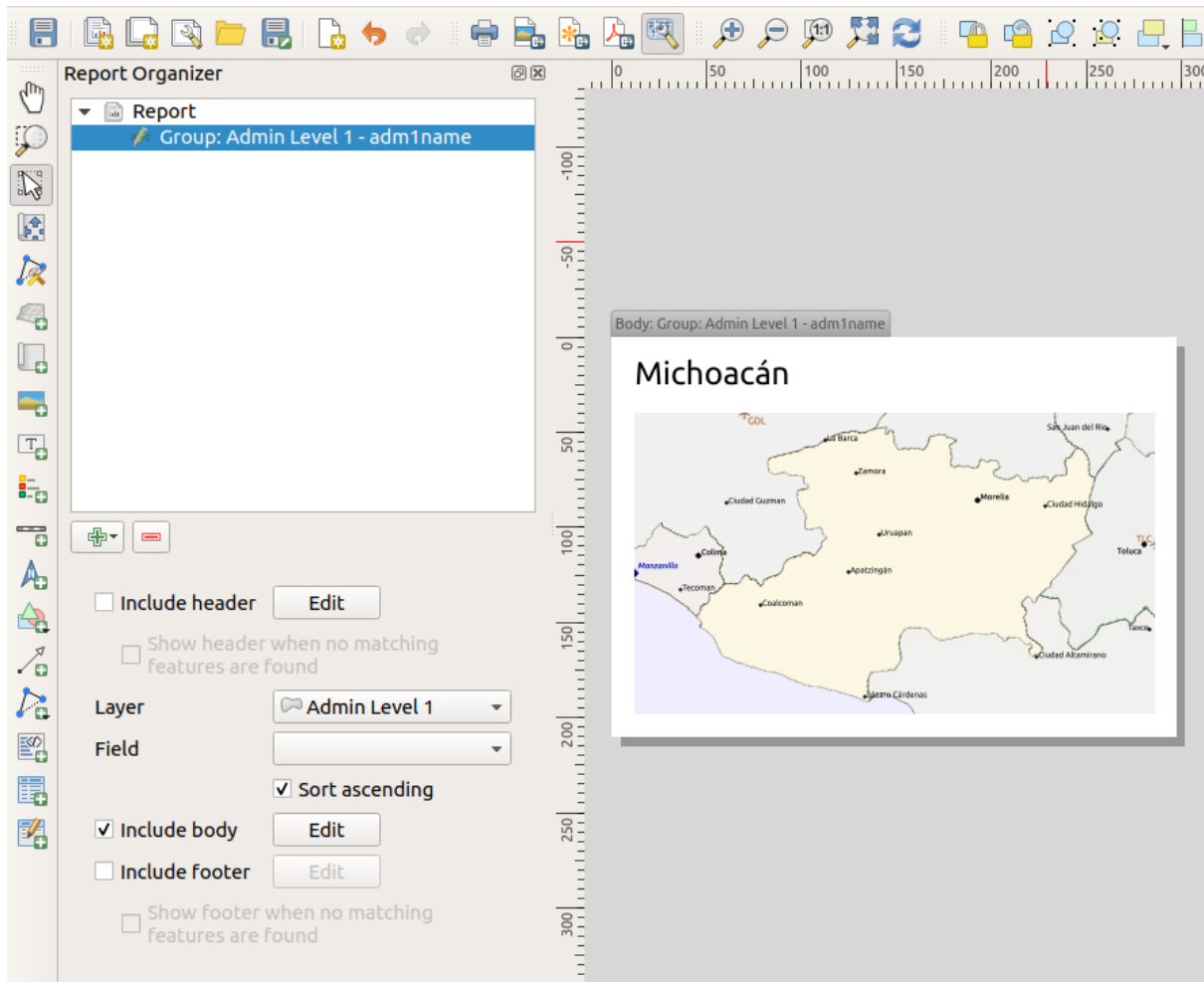
The *Field Group Section* repeats its body layout for every feature of a layer. The features are sorted by the selected grouping feature (with an option for ascending/descending sort). If a field group section has child sections (e.g. another field group section with a different field), then only features with unique values for the group feature are iterated over. This allows nested reports.

For now we will add a *Field Group Section* to our report. At its most basic level, you can think of a *Field Group Section* as the equivalent of a *print atlas*: you select a layer to iterate over, and the report will insert a section for each feature found. Selecting the new *Field Group Section* reveals a number of new related settings:



In this case we've setup our Field Group so that we iterate over all the states from the *Admin Level 1* layer, using

the values from the *adm1name* field. The same options to include header and footer are present, together with a new option to include a *body* for this section. We'll do that, and edit the body:

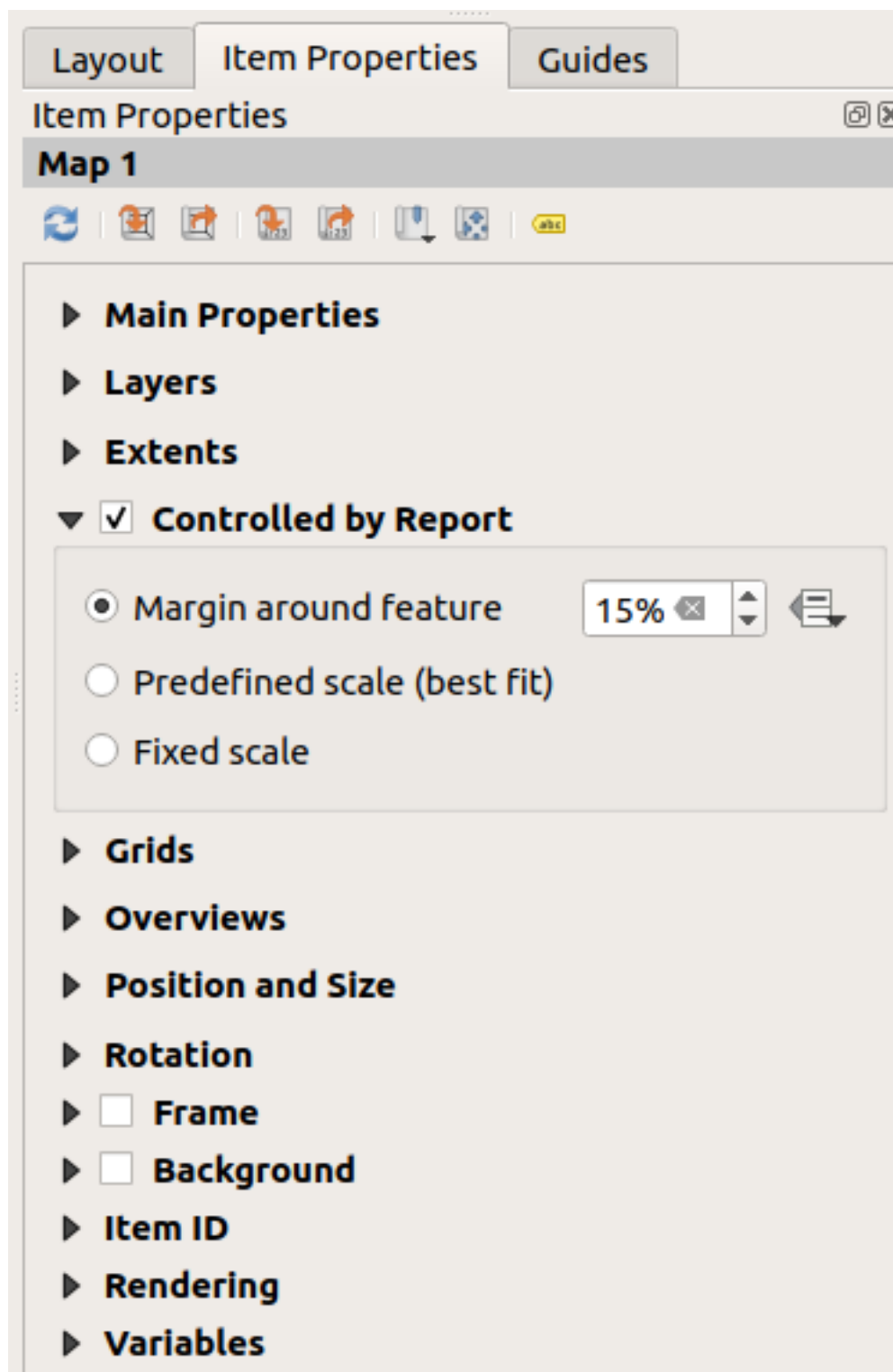


Our body now consists of a map and a label showing the name of the state. To include the name of the state, we selected *Add Item ► Add Label* and data defined the text under *Main Properties* with the help of *Insert or Edit an Expression...*

The result was the following expression (*name* is the name of the attribute in the *Admin Level 1* layer that contains the name of the state):

```
[ % "name" % ]
```

The map is set to follow the current report feature (enabled by checking *Controlled by Report* – just like a map item in an atlas will follow the current atlas feature when *Controlled by Atlas* is checked):



If we went ahead and exported our report now, we'd get something like this:

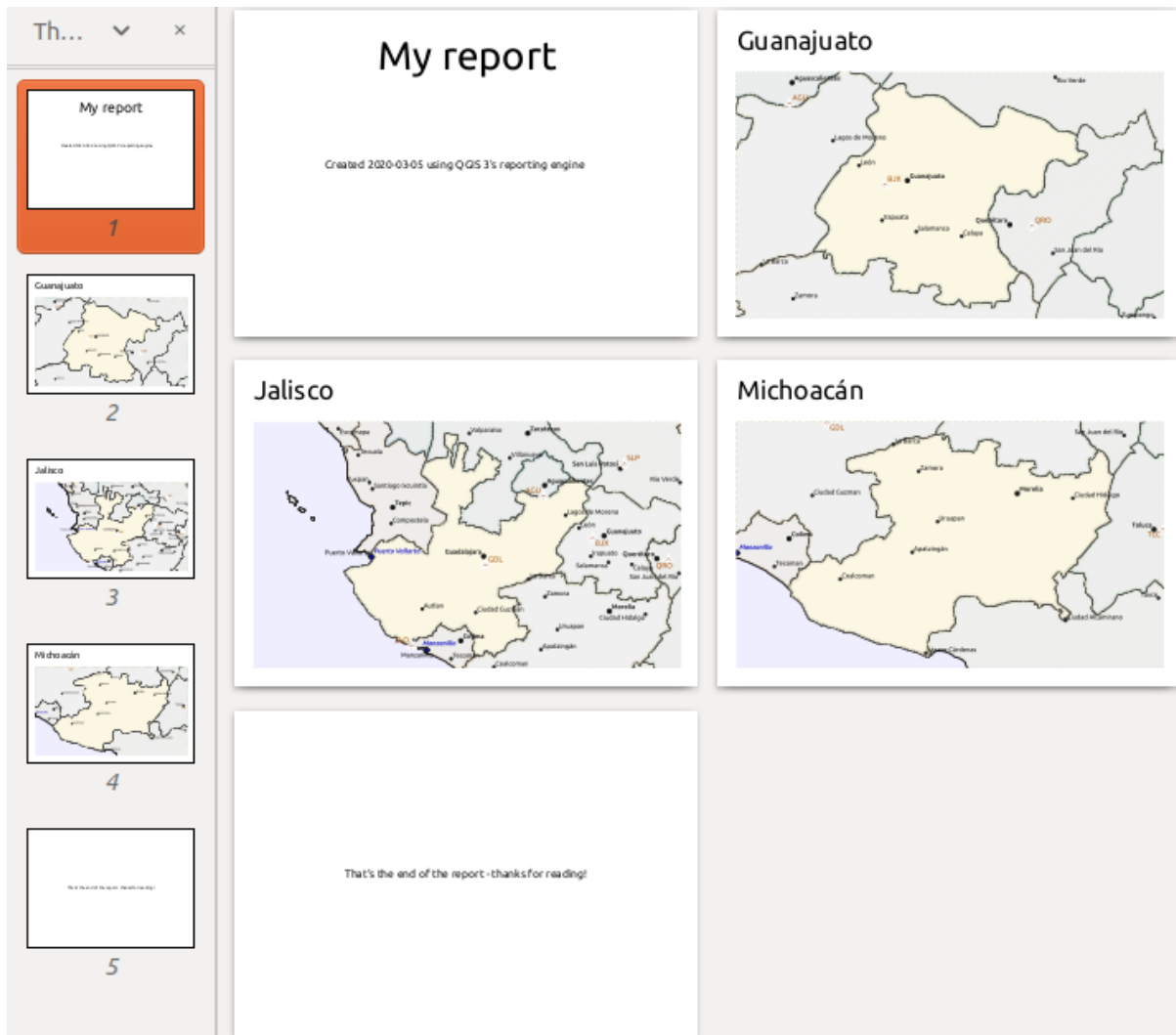
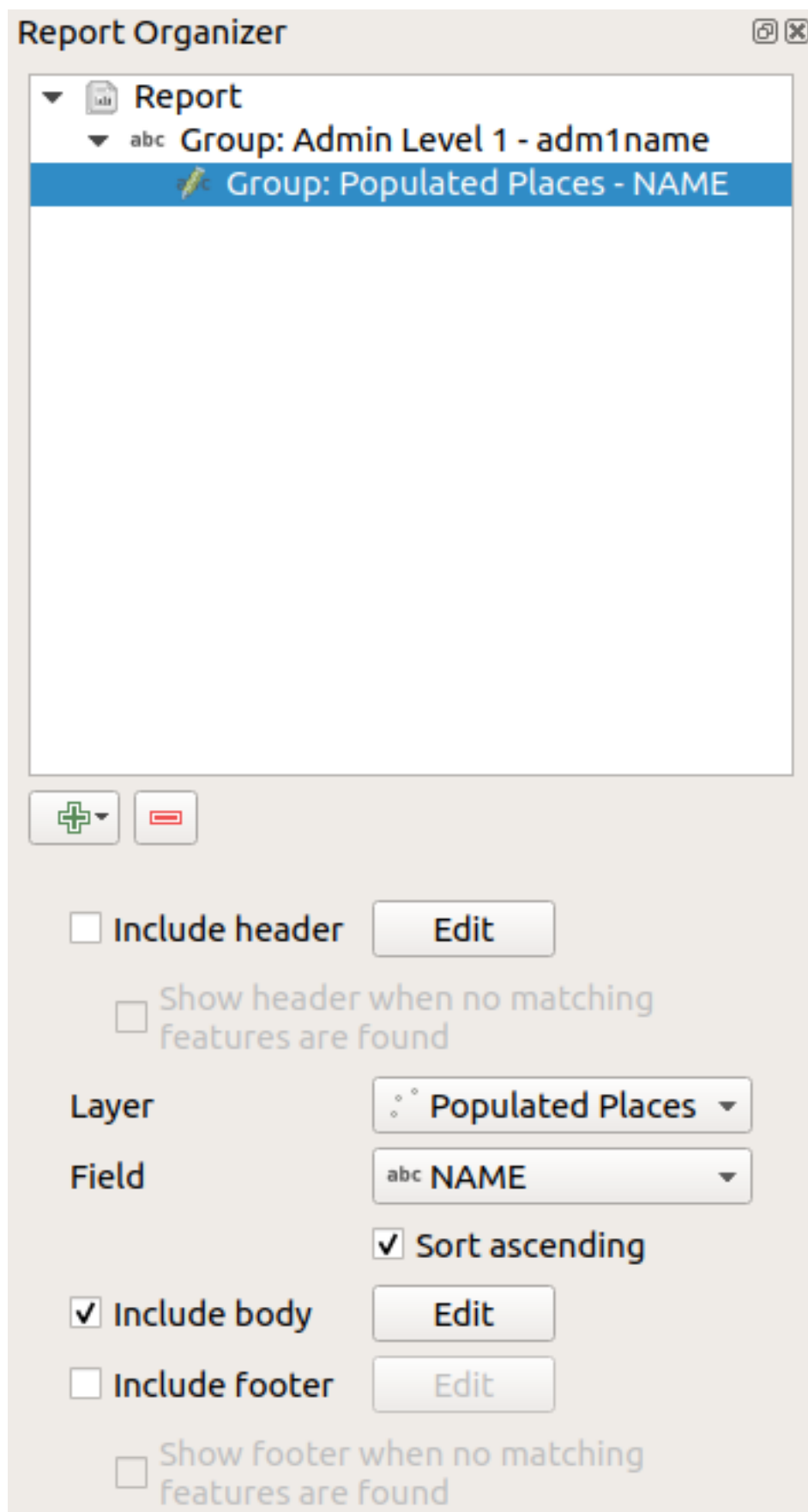


Abb. 22.68: The report header, a page for each state, and the report footer.

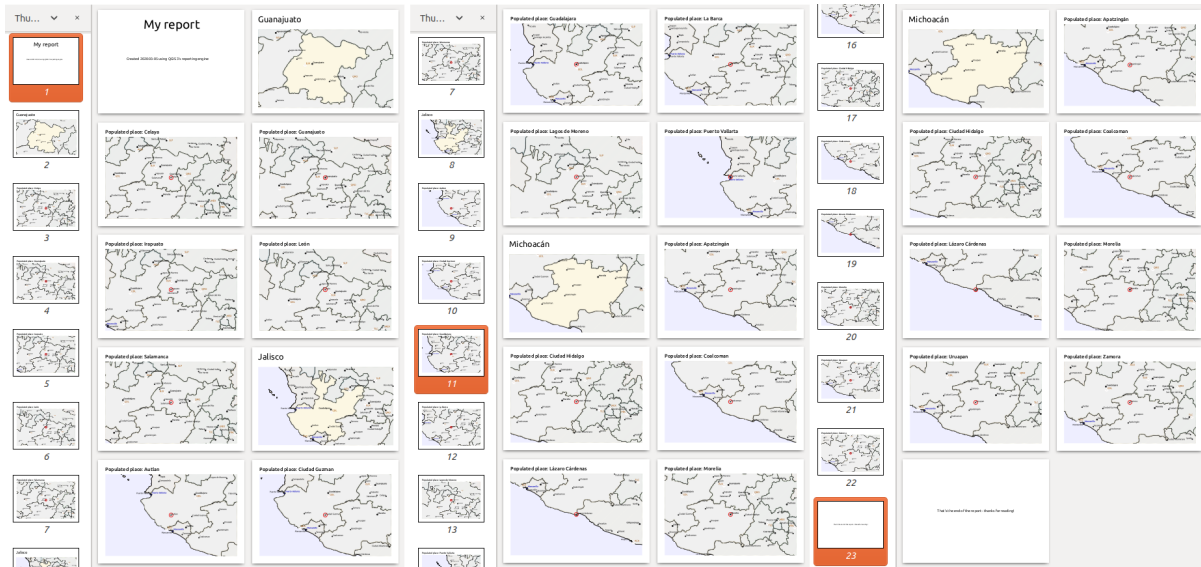
So more or less an atlas, but with a header and footer page.

Let us make things more interesting by adding a subsection to our state group. We do this by first selecting the *Admin Level 1* field group in the organizer, then hitting the  Add Field button and adding a new *Field Group Section*:



When iterating over the features of a *Field Group Section*, the features will be filtered to match the defining field

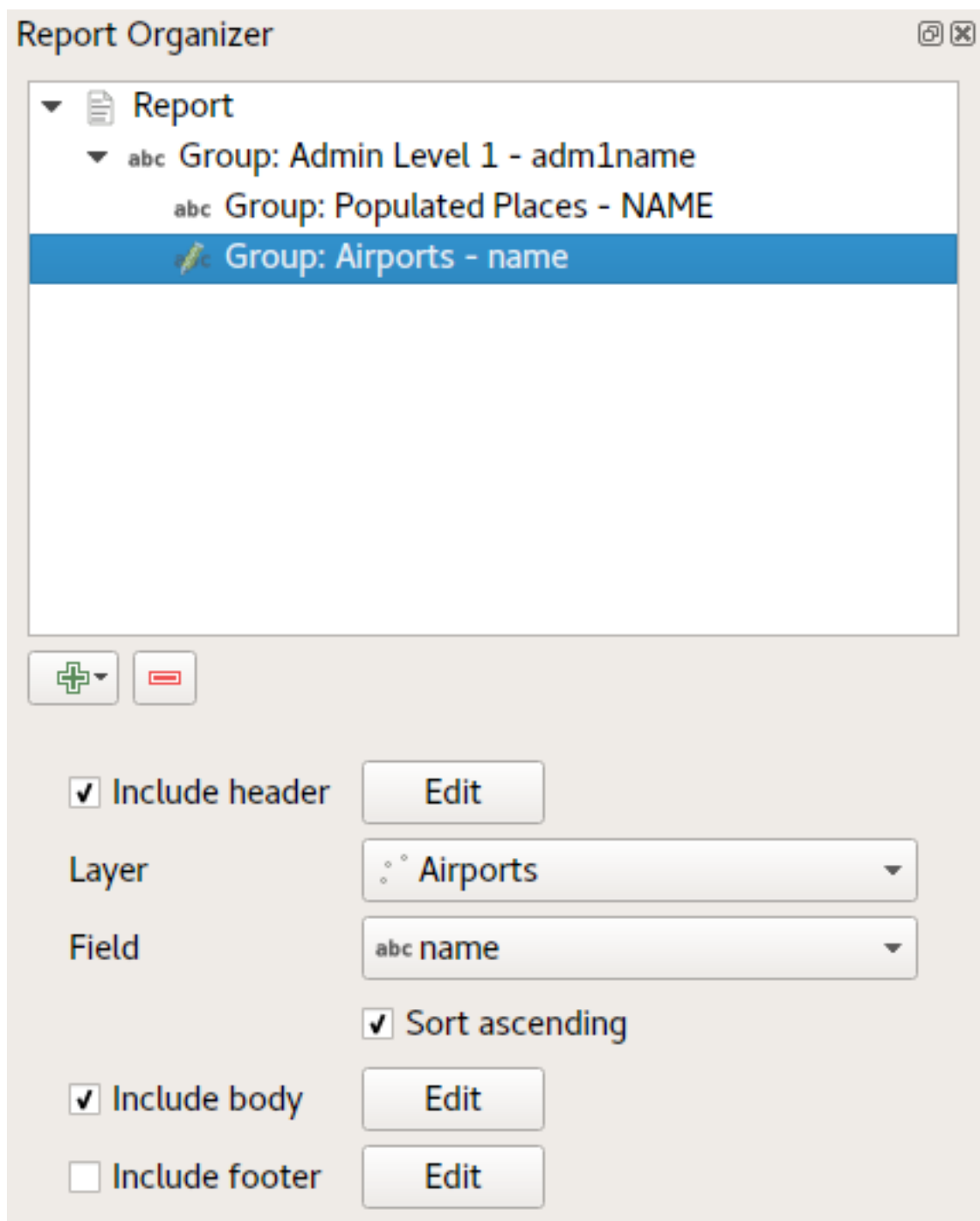
of its parent group (`adm1name` in this case). Here, the subsection we added will iterate over a *Populated Places* layer, including a body section for each place encountered. The magic here is that the *Populated Places* layer has an attribute with the same name as the defining field in the parent layer, `adm1name`, tagging each place with the state it is contained within (if you're lucky your data will already be structured like this – if not, run the [Join Attributes by Location](#) Processing algorithm and create your own field). When we export this report, QGIS will grab the first state from the *Admin Level 1* layer, and then iterate over all the *Populated Places* with a matching `adm1name` value. Here's what we get:



Here we created a basic body for the *Populated Places* group, including a map of the place and a table of some place attributes. So our report is now a report header, a page for the first state, followed by a page for every populated place within that state, then the rest of the states with their populated places, and finally the report footer. If we were to add a header for the *Populated Places* group, it would be included just before listing the populated places for each state, as shown in the illustration below.

Similarly, a footer for the *Populated Places* group would be inserted after the final place for each state is included.

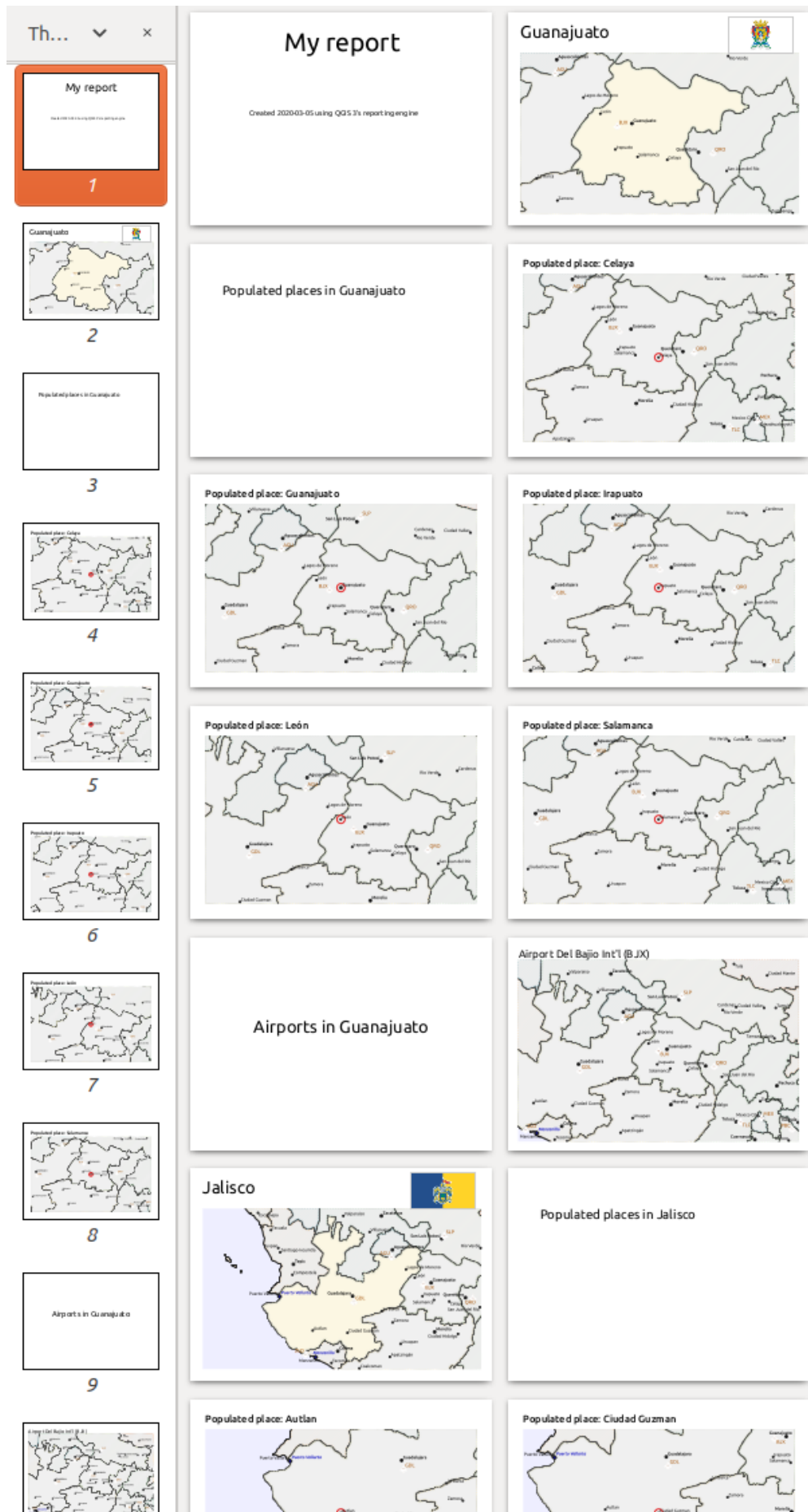
In addition to nested subsections, subsections in a report can also be included consecutively. If we add a second subsection to the *Admin Level 1* group for *Airports*, then (if the *Airports* layer has an attribute `adm1name` that can link it to the parent group) our report will first list ALL the populated places for each state, followed by all the airports within that state, before proceeding to the next state.



The key point here is that our *Airports group* is a subsection of the *Admin Level 1 group* – not the *Populated Places group*.

In this case our report would be structured like this (note that state flags have also been included - the procedure for adding feature specific pictures in this way is described below):






## Including pictures in a report

Pictures can be quite useful in reports, and QGIS allows pictures in both the static and dynamic parts of a report. Pictures are added in the same way as for standard print layouts, and for the static report parts (and static pictures in dynamic parts) there is not more to it.

But if you want illustrations that are tailored to the report features, your layer must have an attribute that can be used to define the picture to include.

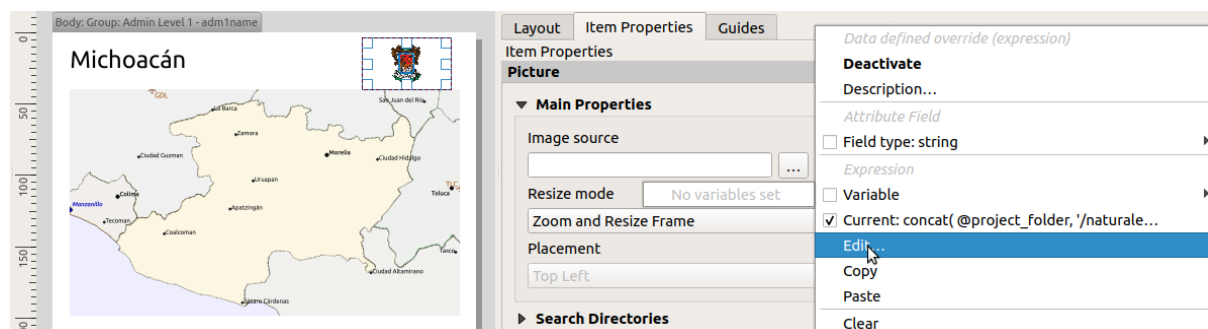
QGIS depends on absolute file names for images in reports.

For dynamic pictures, you first add a picture to the body part of the group, as usual. In the *Item properties* of the picture, you set the *Image Source* using the  Data defined override button, and either select an attribute that contains the absolute path of the images or *Edit...* (to enter an expression that generates the absolute image path).

Below is an example expression that uses string concatenation to specify the absolute path to the pictures, using the directory where the project file is located (@project\_path) and an attribute (admlname) from which the file name is generated (in this case by transforming the string in the admlname attribute to uppercase, and appending \_flag.png):

```
concat(@project_folder, '/natureearth/pictures/' ,
      upper("admlname"), '_flag.png')
```

This means that the pictures are located in the natureearth/pictures subdirectory of the project file directory.



## Highlighting the current report feature in a map

In the above report, the report features are emphasized in the maps using highlighting (state) and circles (populated places). To emphasize the report features in the maps (apart from placing them at the centre of the maps), you must data define the style using a comparison between its @id and the @atlas\_featureid, as for atlases.

For instance, if you would like to use a thicker line / border for the report feature than the other features you can data define the line width:

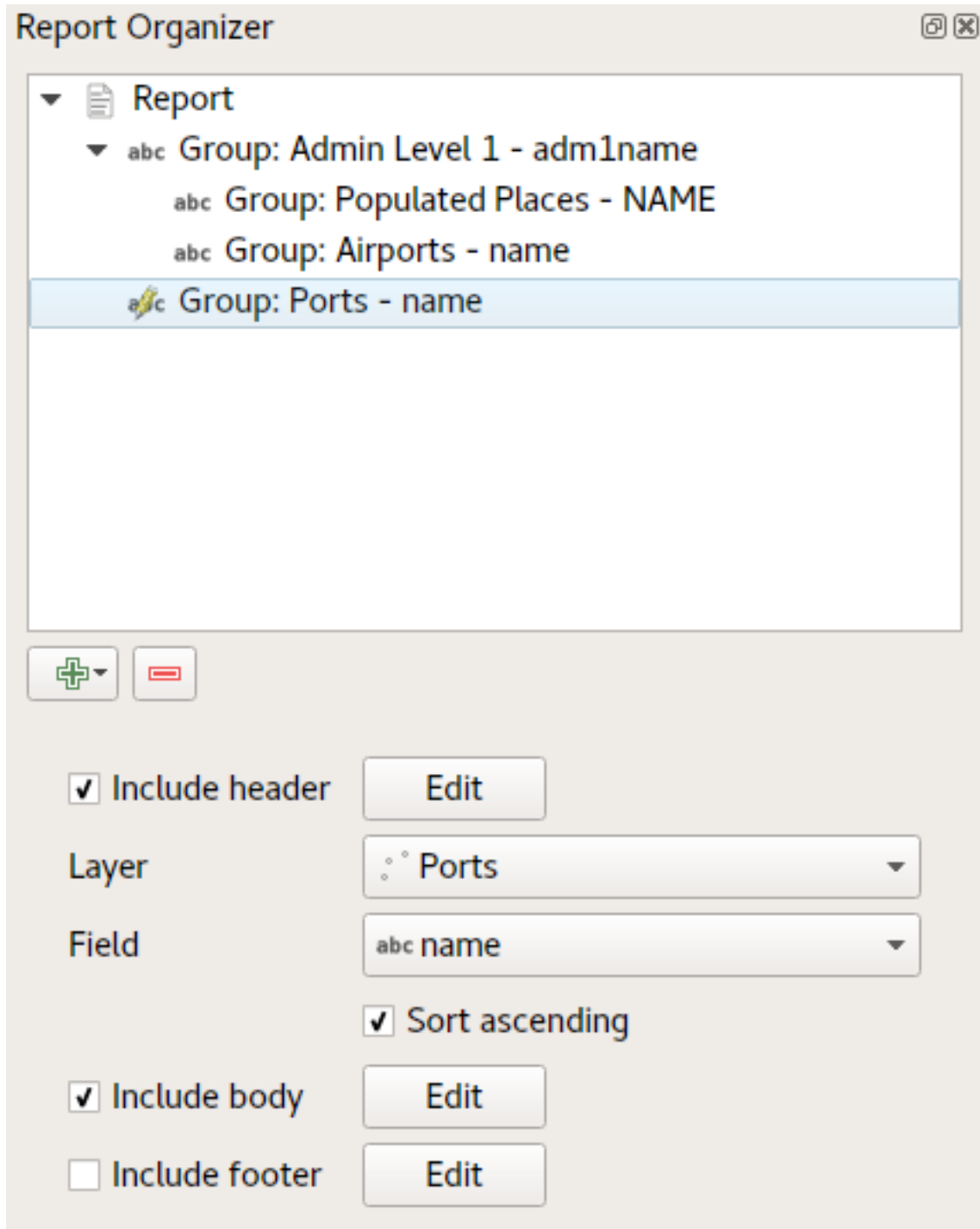
```
if($id=@atlas_featureid, 2.0, 0.1)
```

The report feature will get a 2 units wide polygon outline, while all other features will get a 0.1 units wide line. It is also possible to data define the colour (non-transparent dark magenta for the report feature and semi-transparent light gray for the other features):

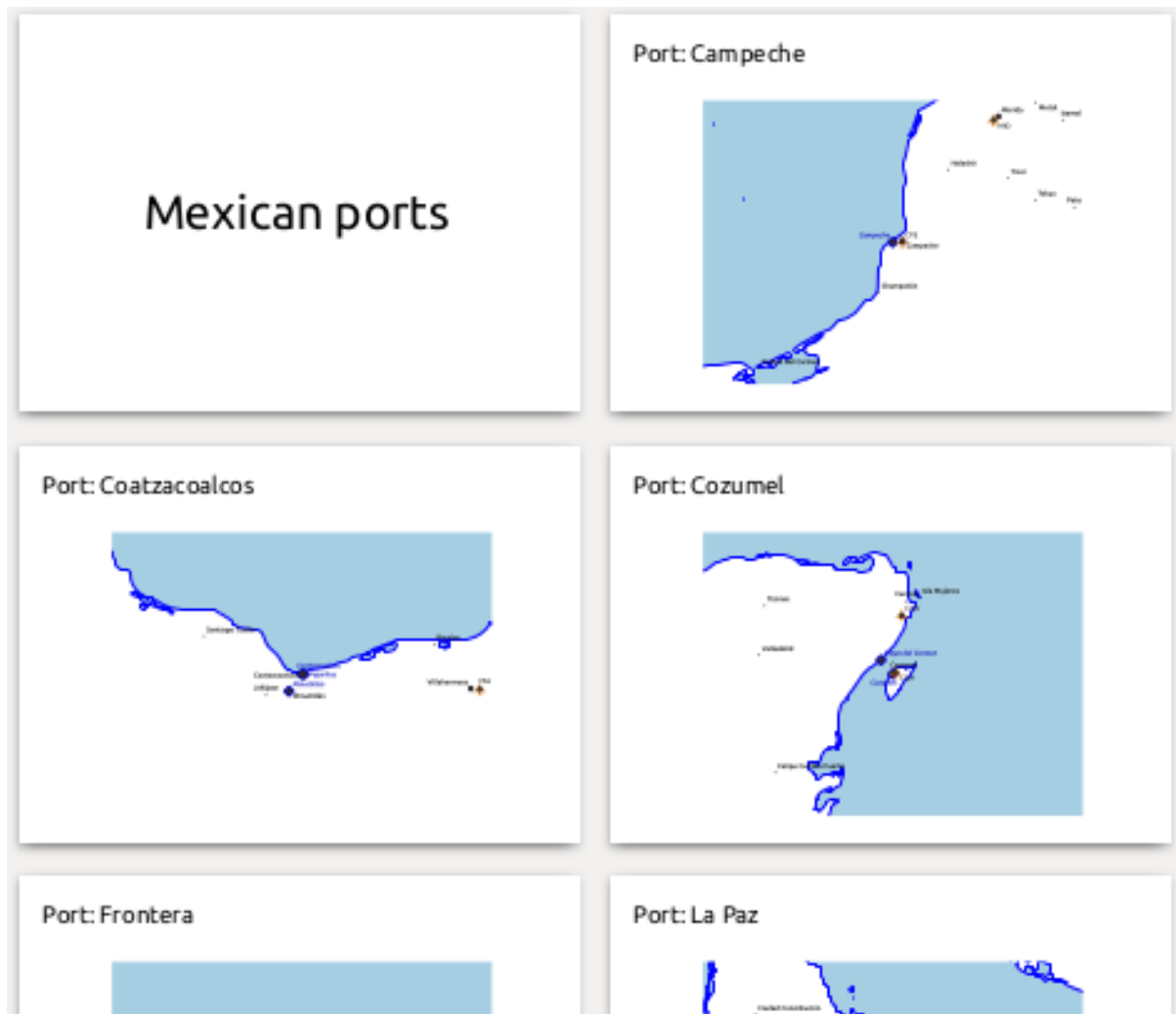
```
if($id=@atlas_featureid, '#FF880088', '#88CCCCC')
```

### More level 1 groups

Combining nested and consecutive sections, together with section headers and footers allows for tons of flexibility. For instance, in the below report we add another field group as a child of the main report for the :guiabel`Ports` layer. Now, after listing the states together with their populated places and airports, we'll get a summary list of all the ports in the region:



This results in the last part of our report exporting as:



## 22.4.4 Exporteinstellungen

When you export a report (*Report ► Export Report as Images... / SVG... / PDF...*), you will be asked for a file name, and then you get the opportunity to tune the export settings to get the most appropriate output.

As you see, reports in QGIS are extremely powerful and flexible!

---

**Bemerkung:** The current information was adapted from a North Road blog, [Exploring Reports in QGIS 3.0 - the Ultimate Guide!](#)

---

---

## Arbeiten mit OGC / ISO-Protokollen

---

Das Open Geospatial Consortium (OGC) ist eine internationale Organisation mit mehr als 300 Mitgliedern aus kommerziellen, behördlichen Bereichen, aus der Forschung sowie aus Non-Profit Organisationen Vereinen. Die Mitglieder entwickeln und implementieren Standards für den Austausch räumlicher Daten, GIS-Datenprocessing und standardisierte Bereitstellung von Geodaten.

Describing a basic data model for geographic features, an increasing number of specifications are developed by OGC to serve specific needs for interoperable location and geospatial technology, including GIS. Further information can be found at <https://www.ogc.org/>.

Wichtige von QGIS unterstützte OGC Spezifikationen sind:

- **WMS** — Web Map Service (*WMS/WMTS Client*)
- **WMTS** — Web Map Tile Service (*WMS/WMTS Client*)
- **WFS** — Web Feature Service (*WFS und WFS-T Klient*)
- **WFS-T** — Web Feature Service - Transactional (*WFS und WFS-T Klient*)
- **WCS** — Web Coverage Service (*WCS Client*)
- **WPS** — Web Processing Service
- **CSW** — Catalog Service for the Web
- **SFS** — Simple Features for SQL (*PostGIS Layers*)
- **GML** — Geography Markup Language

OGC-Dienste werden vermehrt zum Austausch von geographischen Daten zwischen unterschiedlichen GIS-Systemen und -implementierungen verwendet. QGIS unterstützt mittlerweile die oben genannten Spezifikationen als Client in Form von **SFS** (durch den Postgresql/PostGIS Datenprovider, vgl. *PostGIS Layers*).

Sie können Ihre Karten und Daten auch über die Protokolle WMS, WMTS, WFS, WFS-T und WCS teilen, indem Sie einen Webserver benutzen, auf dem QGIS Server, UMN MapServer oder GeoServer installiert ist.

## 23.1 WMS/WMTS Client

### 23.1.1 Übersicht über die WMS-Unterstützung

Derzeit kann QGIS als WMS-Klient eingesetzt werden. Es unterstützt die Versionen 1.1, 1.1.1 und 1.3 der WMS-Server. Gut getestet wurden die öffentlich verfügbaren Server wie beispielsweise DEMIS.

WMS-Server liefern Daten aufgrund einer Anfrage eines Klienten (hier QGIS) als Rasterbild aus. Dabei spielen Ausdehnung, Anzahl der angefragten Layer, Symbolisierungen und Transparenz eine Rolle. Der WMS-Server holt die benötigten Daten dann aus seiner Datenquelle hervor, rendert diese in eine Rasterkarte und sendet das fertige Bild zurück zum Klienten. Das für QGIS typische Rasterformat ist in aller Regel JPEG oder PNG.

WMS ist ein komplett auf Übertragung ausgelegter Dienst (REST = Representational State Transfer). Daraus resultiert die Tatsache, dass die von QGIS generierte URL für das Bild auch in einem Browser eingesetzt werden kann. Das Resultat dieser Anfrage sieht in der Regel genauso aus wie in QGIS. Das ist besonders hilfreich, wenn es beim Einsatz von WMS Probleme geben sollte. Da es sehr viele unterschiedle WMS-Server-Anbieter am Markt gibt (und alle die WMS-Spezifikation etwas unterschiedlich interpretieren), ist eine Überprüfung im Browser sehr hilfreich.

WMS-Layer können sehr einfach hinzugefügt werden, solange man die URL des Servers kennt, eine Verbindung über HTTP zu diesem Server besteht und der angefragte Server auch HTTP versteht.

Additionally, QGIS will cache your WMS responses (i.e. images) for 24h as long as the GetCapabilities request is not triggered. The GetCapabilities request is triggered every time the *Connect* button in the *WMS/WMTS* dialog is used to retrieve the WMS server capabilities. This is an automatic feature meant to optimize project loading time. If a project is saved with a WMS layer, the corresponding WMS tiles will be loaded from the cache the next time the project is opened as long as they are not older than 24h.

### 23.1.2 Überblick über die WMTS Unterstützung

**lqgl** kann auch als WMTS Client fungieren. WMTS ist ein OGC Standard zum Bereitstellen von Tile Sets von räumlichen Daten. Dies ist ein schnellerer und effizienterer Weg als die Bereitstellung über einen WMS weil bei WMTS die Tile Sets vorgeneriert werden und der Client nur die Übermittlung von Tiles, nicht Ihre Erstellung abfragt. Ein WMS beinhaltet typischerweise sowohl die Erstellung als auch die Übermittlung der Daten. Ein bekanntes Beispiel eines nicht-OGC Standards zum Darstellen von gekachelten räumlichen Daten ist Google Maps.

Um die Daten in einer Vielzahl von Maßstäben je nach Anforderung darzustellen werden die WMTS Tile Sets in mehreren verschiedenen Maßstäben erstellt und dem GIS Client zur Abfrage bereitgestellt.

Das Diagramm veranschaulicht das Konzept der Tile Sets:

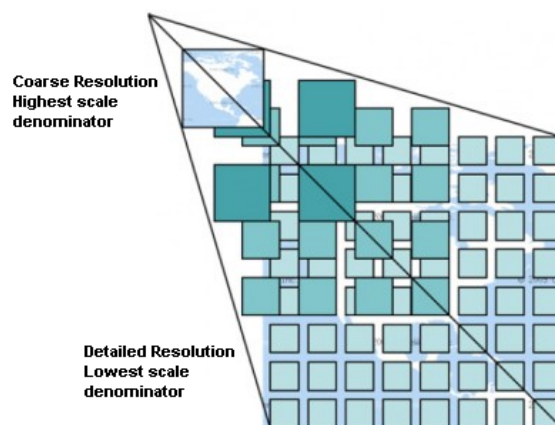


Abb. 23.1: Konzept der WMTS Tile Sets

Die zwei Typen von WMTS Schnittstellen die QGIS unterstützt sind die über Key-Value-Pairs (KVP) und RESTful. Diese zwei Schnittstellen sind unterschiedlich und Sie müssen diese für QGIS unterschiedlich spezifizieren.

1. Um einen **WMTS KVP** Service anzubinden muss ein QGIS Benutzer die WMS/WMTS Schnittstelle öffnen und die folgenden Zeichenkette zu der URL des WMTS Tile Service hinzufügen:

```
"?SERVICE=WMTS&REQUEST=GetCapabilities"
```

An example of this type of address is:

```
https://opencache.statkart.no/gatekeeper/gk/gk.open_wmts?service=WMTS&
→request=GetCapabilities
```

Zu Testzwecken funktioniert der topo2 Layer in diesem WMTS gut. Indem man diese Zeichenfolge hinzufügt gibt man an dass ein WMTS Web Service anstatt eines WMS Service benutzt werden soll.

2. Der **RESTful WMTS** Service erfordert eine andere Form, eine einfache URL. Das von der OGC empfohlene Format ist:

```
{WMTSBaseURL}/1.0.0/WMTSCapabilities.xml
```

This format helps you to recognize that it is a RESTful address. A RESTful WMTS is accessed in QGIS by simply adding its address in the WMS setup in the URL field of the form. An example of this type of address for the case of an Austrian basemap is:

```
https://maps.wien.gv.at/basemap/1.0.0/WMTSCapabilities.xml
```





**Bemerkung:** Es gibt immer noch einige alte WMS-C-Dienste. Diese sind ähnlich wie WMTS (d.h. sie dienen dem selben Zweck, funktionieren aber anders). Sie können Sie genau so verwalten wie WMTS-Dienste. Fügen Sie einfach `?tiled=true` an das Ende der URL an. Siehe [https://wiki.osgeo.org/wiki/Tile\\_Map\\_Service\\_Specification](https://wiki.osgeo.org/wiki/Tile_Map_Service_Specification) für weitere Informationen über diese Spezifikation.


Wenn Sie WMTS lesen können Sie auch an WMS-C denken.

### 23.1.3 WMS/WMTS Server auswählen

The first time you use the WMS/WMTS feature in QGIS, there are no servers defined.

Dann müssen Sie Verbindungen zu den Servern, auf die Sie zugreifen wollen, definieren:

1. Gehen Sie zum Tab  **WMS/WMTS** des Dialogs *Datenquellenverwaltung* indem Sie:
  - entweder auf die Schaltfläche  *Datenquellenverwaltung öffnen* klicken (oder Tastenkürzel `Strg+L`) und dann den Tab aktivieren
  - oder auf die Schaltfläche  *WMS/WMTS-Layer hinzufügen* in der *Layerverwaltungswerkzeugleiste* klicken
  - oder im Menü *Layer* ► *Layer hinzufügen* ►  den Menüpunkt *WMS/WMTS-Layer hinzufügen...* anwählen
2. Klicken Sie auf *Neu* im Tab *Layer*. Der Dialog *Neue WMS/WMTS-Verbindung anlegen* erscheint.

**Tipp:** Ein Rechtsklick auf den Eintrag  *WMS/WMTS* innerhalb des *Bedienfelds Browser* und die Auswahl *Neue Verbindung...* öffnet ebenfalls den Dialog *Neue WMS/WMTS-Verbindung anlegen*.

3. Geben Sie dort die Verbindungsparameter zu Ihrem gewünschten WMS-Server ein, so wie im Folgenden beschrieben:

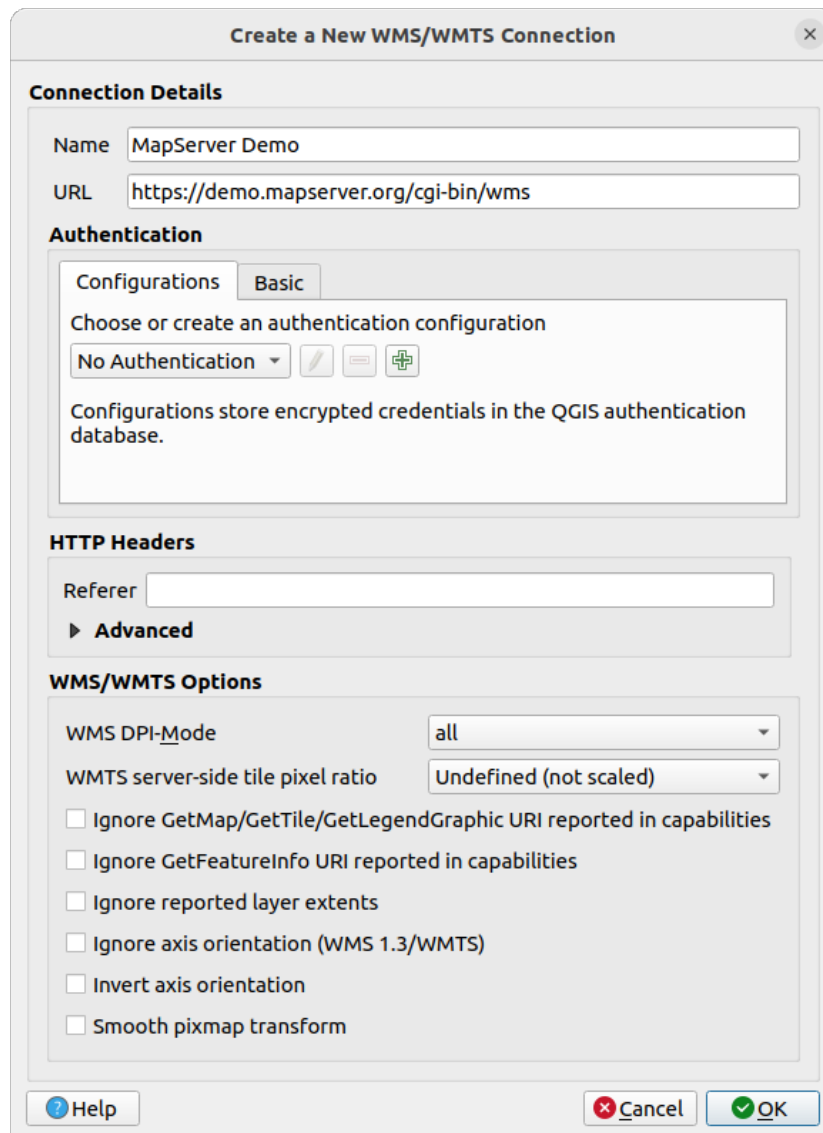


Abb. 23.2: Verbindung mit einem WMS-Server anlegen

- *Name*: Der Name der Verbindung. Der Name wird in der Auswahlliste der Serververbindungen benutzt, so dass Sie die WMS-Server unterscheiden können.
- *URL*: Die URL des Servers. Es muss sich um einen auflösbaren Hostnamen handeln – im selben Format, wie Sie es für eine Telnetverbindung benutzen würden oder um einen Host anzupingen, d.h. nur die Basis-URL. Ihre URL sollte also keine Teile wie `request=GetCapabilities` oder `version=1.0.0` enthalten.
- *Authentifizierung* (optional): Benutzen Sie eine *gespeicherte Konfiguration* oder eine Basic-Authentifizierung mit *Benutzername* und *Passwort*.

**Warnung:** Entering **username** and **password** in the *Authentication* tab will keep unprotected credentials in the connection configuration. Those **credentials will be visible** if, for instance, you shared the project file with someone. Therefore, it's advisable to save your credentials in an *Authentication configuration* instead (*Configurations* tab). See [Authentifizierungssystem](#) for more details.


- HTTP *Referer*



- *WMS DPI-Mode*: Available options are **all**, **off**, **QGIS**, **UMN** and **GeoServer**
- *WMTS server-side tile pixel ratio*: When rendering WMTS layers, allows to scale up or down the tiles based on the device screen DPI. Available options are *Undefined (not scaled)*, *Standard (96 DPI)* and *High (192 DPI)*.
- ☐ *Ignore GetMap/GetTile/GetLegendGraphic URI reported in capabilities*: if checked, use given URI from the *URL* field above.
- ☐ *Gemeldete GetFeatureInfo-URI ignorieren*: falls Sie das anhaken, wird die im *URL* Feld eingegebene URI benutzt.
- ☐ *Berichtete Layerausmaße ignorieren*: die von Rasterlayern berichteten Layerausmaße können kleiner sein, als die Fläche, die tatsächlich gerendert wird (dies ist insbesondere bei WMS-Servern der Fall, die Symbole enthalten, die mehr Platz brauchen als die Daten selbst). Haken Sie diese Option an, um zu verhindern, dass Rasterlayer am Rand abgeschnitten werden, was zu abgeschnittenen Symbolen in den Randbereichen dieser Layer führen kann
- ☐ *Achsenorientierung ignorieren (WMS 1.3/WMTS)*
- ☐ *Achsenorientierung invertieren*
- ☐ *Pixmap-Transformation glätten*

4. Klicken Sie auf *OK*

Once the new WMS/WMTS server connection has been created, it will be preserved for future QGIS sessions. Note that it is also possible to *Load* the connection parameters from a *.XML* file or *Save* them to a *.XML* file.

Wenn Sie einen Proxyserver konfigurieren müssen, um WMS-Dienste aus dem Internet zu beziehen, können Sie Ihren Proxyserver in den Optionen hinzufügen. Wählen Sie im Menü *Einstellungen* ► *Optionen* und klicken Sie auf den Tab *Netzwerk*. Dort können Sie Ihre Proxyeinstellungen einfügen und sie durch Anhaken von ☒ *Proxy für Webzugriff verwenden* aktivieren. Stellen Sie sicher, dass Sie den richtigen *Proxytyp*  aus der Auswahl eingestellt haben.

### 23.1.4 WMS/WMTS Layer laden

Sobald Sie einmal erfolgreich Ihre Parameter eingegeben haben, können Sie die Schaltfläche *Verbinden* benutzen, um die Eigenschaften des ausgewählten Servers herunterzuladen. Dazu gehören die Bildkodierung, die Layer und ihre Stile sowie die Bezugssysteme. Da es sich um eine Netzwerkanfrage handelt, hängt die Antwortgeschwindigkeit von der Qualität Ihrer Netzwerkverbindung zum WMS-Server ab. Während Daten vom WMS-Server heruntergeladen werden, wird der Downloadfortschritt in der unteren linken Ecke des QGIS-Hauptfensters dargestellt.

Ihr Bildschirm sollte nun so ähnlich wie im Bild [Abb. 23.3](#) aussehen, das die Antwort eines WMS-Servers darstellt.

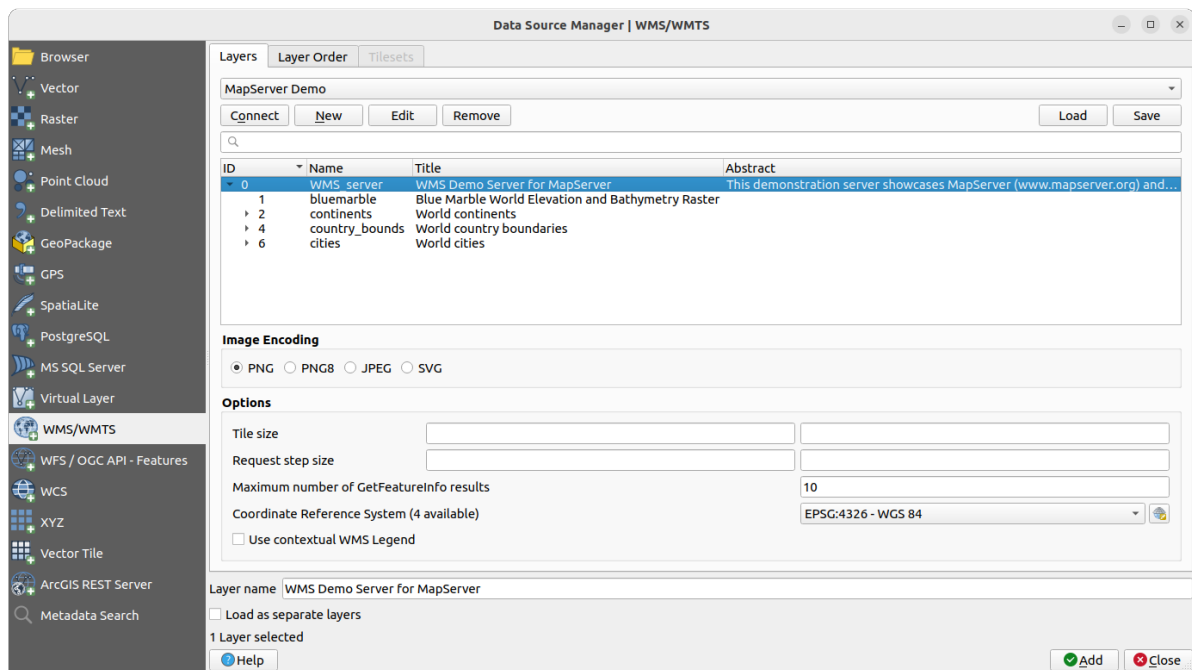


Abb. 23.3: Daten von einem WMS-Server hinzufügen, mit Auflistung der verfügbaren Layer

Der obere Teil des Tabs *Layer* im Dialog zeigt eine Baumstruktur, die Layergruppen mit ihren enthaltenen Layern und dem (bzw. den) zugehörigen Stil(en) enthält, die vom Server angeboten werden. Jeder Eintrag wird durch:

- eine *ID*
- einen *Namen*
- einen *Titel*
- und eine *Zusammenfassung* identifiziert.

Sie können die Liste mit dem Suchfeld  rechts oben filtern.

## Bildkodierung

Der *Bildkodierung* Bereich listet die Formate die sowohl vom Client als auch vom Server unterstützt werden auf. Wählen Sie eines abhängig von Ihren Anforderungen an die Bildqualität aus.

### Tipp: Bildkodierung

In der Regel bieten WMS-Server JPEG oder PNG als Bildkodierung an. JPEG hat eine bildverschlechternde Kompression, während PNG zumeist die Qualität der ursprünglichen Rasterdaten widerspiegelt.



Verwenden Sie JPEG wenn Sie damit rechnen das die WMS Daten photographischen Charakter haben und/oder Sie ein Verlust an Bildqualität nicht stört. Dieser Kompromiss reduziert typischerweise die Datentransferbedingungen um das fünffache verglichen mit PNG.

Verwenden Sie PNG wenn Sie eine genaue Wiedergabe der Originaldaten erzielen wollen und die erhöhten Datentransferbedingungen Sie nicht stören.


## Optionen

Der Dialogbereich Optionen enthält Möglichkeiten, die WMS-Abfrage zu konfigurieren. Sie können folgendes einstellen:

- *Kachelgröße* wenn Sie die Kachelgröße (bspw., 256x256) einstellen wollen, um die WMS-Abfrage in viele einzelne Abfragen aufzuteilen.

- *Request step size*: if you want to reduce the effect of cut labels at tile borders, increasing the step size creates larger requests, fewer tiles and fewer borders. The default value is 2000.
  - Die *Maximale Anzahl von GetFeatureInfo-Ergebnissen* vom Server
  - Each WMS layer can be presented in multiple CRSs, depending on the capability of the WMS server. If you select a WMS from the list, a field with the default projection provided by the web server appears. Press the  **Select CRS** widget to replace the default projection of the WMS with another CRS supported by the WMS server.
- Sie sehen einen Dialog ähnlich wie im Bild [Abb. 10.4](#). Der Hauptunterschied zur WMS-Version des Dialogs ist, dass er nur KBS anzeigt, die vom WMS-Server unterstützt werden.
- Schließlich können Sie noch  *Kontextbezogene WMS-Legende verwenden* anhängen, falls der WMS-Server dies unterstützt. Dann wird nur die für Ihren aktuellen Kartenausschnitt relevante Legende gezeigt, wodurch Legendenelemente für Objekte ausgeblendet werden, die Sie im aktuellen Kartenausschnitt gar nicht sehen.

Ganz unten im Dialog sehen Sie ein Eingabefeld *Layername*, das den *Titel* des ausgewählten Eintrags enthält. Sie können den Namen nach Belieben ändern. Dieser Name wird im Bedienfeld *Layer* erscheinen, nachdem Sie auf die Schaltfläche *Hinzufügen* geklickt und damit den oder die Layer nach QGIS geladen haben.

You can select several layers at once, but only one image style per layer. When several layers are selected, they will be combined at the WMS server and transmitted to QGIS in one go, as a single layer. The default name is a slash (/) separated list of their original title. You can however opt to  *Load as separate layers*.

### Layerreihenfolge

Der Tab *Layerreihenfolge* listet die ausgewählten Layer des verbundenen WMS-Server auf.

Die WMS-Layers werden vom Server in der Reihenfolge übereinander gerendert, wie sie in der Liste im Tab *Layer* erscheinen, von oben nach unten. Möchten Sie die Reihenfolge ändern, können Sie die Schaltflächen *Auf* und *Ab* im Tab *Layerreihenfolge* verwenden.

### Transparenz


Die Einstellung für die *Globale Deckkraft* in den *Layerseigenschaften* ist hartkodiert, um stets aktiv zu sein, falls verfügbar.

## 23.1.5 Tilesets


When using WMTS (Cached WMS) services you are able to browse through the *Tilesets* tab given by the server. Additional information like tile size, formats and supported CRS are listed in this table.

```
# example of WMTS service
https://opencache.statkart.no/gatekeeper/gk/gk.open_wmts?service=WMTS&
  ↪ request=GetCapabilities
```

Selecting a layer to load, it is also possible to apply an *Interpretation method*, converting it into a raster layer of single band float type, ready for styling using QGIS usual *raster renderers*.

In combination with this feature, you can use the tile scale slider by selecting *View ► Panels* ( or  *Settings ► Panels*), then choosing *Tile Scale Panel*. This gives you the available scales from the tile server with a nice slider docked in.


## 23.1.6 Das Objekte abfragen Werkzeug

Nachdem Sie einen Layer von einem WMS-Server geladen haben, können Sie die Layer mit dem Werkzeug  Objekte Abfragen abfragen, sofern der WMS-Server diese Funktion unterstützt. Ein Klick auf einen Pixel stellt dann eine Abfrage an den WMS-Server für diesen Pixel. Das Ergebnis wird in Textform geliefert. Die Formatierung hängt von dem jeweilig verwendeten WMS-Server ab.

### Formatauswahl

Wenn durch den Server mehrere Formate unterstützt werden wird dem Objekte abfragen Dialog automatisch eine Kombobox mit den unterstützten Formaten hinzugefügt und das ausgewählte Format kann im Projekt für den Layer gespeichert werden.

### GML Formatunterstützung

Das  Objekte abfragen Werkzeug unterstützt WMS Server Response (GetFeatureInfo) im GML Format (es wird Feature in der QGIS GUI in diesem Zusammenhang genannt). Wenn das „Feature“ Format vom Server unterstützt wird und ausgewählt ist, sind die Ergebnisse des Objekte abfragen Werkzeugs Vektorobjekte, wie bei einem normalen Vektorlayer. Wenn ein einzelnes Objekt im Baum ausgewählt wird wird es in der Karte hervorgehoben und kann in die Zwischenablage kopiert werden und in einen anderen Vektorlayer eingefügt werden. Sehen Sie sich für die Unterstützung von GetFeatureInfo im GML Format das Beispielsetup des UMN Mapservers unten an.

```
# in layer METADATA add which fields should be included and define geometry_
↪ (example):

"gml_include_items"    "all"
"ows_geometries"       "mygeom"
"ows_mygeom_type"      "polygon"

# Then there are two possibilities/formats available, see a) and b):

# a) basic (output is generated by Mapserver and does not contain XSD)
# in WEB METADATA define formats (example):
"wms_getfeatureinfo_formatlist" "application/vnd.ogc.gml,text/html"

# b) using OGR (output is generated by OGR, it is sent as multipart and contains_
↪ XSD)
# in MAP define OUTPUTFORMAT (example):
OUTPUTFORMAT
  NAME "OGRGML"
  MIMETYPE "ogr/gml"
  DRIVER "OGR/GML"
  FORMATOPTION "FORM=multipart"
END

# in WEB METADATA define formats (example):
"wms_getfeatureinfo_formatlist" "OGRGML,text/html"
```

## 23.1.7 Viewing Properties

Once you have added a WMS server, you can view its properties by right-clicking on it in the legend and selecting *Properties*. The WMS/WMTS layer properties is much like the raster layer properties so you will find detailed description at *Dialogfenster Rasterlayereigenschaften*. However, there are some differences, which will be explained below.

## Information properties

### Reiter Metadaten

Der Reiter *Metadaten* im Kontextmenü zeigt eine Vielzahl von Informationen über den WMS-Server. Diese Infos sind dem Capabilities-Dokument des Servers entnommen. Viele Definitionen können reduziert werden indem man den WMS Standard liest (siehe OPEN-GEOSPATIAL-CONSORTIUM *Literatur und Internetreferenzen*), hier sind dazu einige praktische Definitionen:


- **Servereigenschaften**

- **WMS Version**— Die WMS-Version, die vom Server unterstützt wird.
- **Bildformate** — Eine Liste der MIME-Typen mit denen der Server antworten kann. QGIS unterstützt jedes Format, welches die darunterliegende Bibliothek QT unterstützt, mindestens aber `image/png` und `image/jpeg`.
- **Abfrageformate** — Eine Liste der MIME-Typen mit denen der Server auf Pixel-Abfragen antworten kann. Derzeit wird von QGIS nur der Typ `text-plain` unterstützt.

- **Layereigenschaften**

- **Ausgewählt** — Gibt an, ob dieser Layer während des Hinzufügens des Server ausgewählt war.
- **Sichtbar** — Gibt an, ob der Layer in der Legende angezeigt wird oder nicht. (noch nicht verwendet in der aktuellen Version von QGIS.)
- **Kann abfragen** — Gibt an, ob der Layer auf Abfragen Ergebnisse zurückgibt.
- **Kann Transparenz** — Gibt an, ob der Layer transparent gezeichnet werden kann. Diese QGIS Version verwendet ein hardcodiertes `Ja`, sofern die Bildkodierung Transparenz bietet
- **Kann reingezoomt werden** —Gibt an, ob dieser Layer gezoomt werden kann. Diese Version von QGIS verwendet standardmäßig `Ja`. Daher kann es sein, dass einige Layer komisch aussehen, die diese Funktion nicht unterstützen.
- **Kaskadierend** — WMS-Server können als Proxy zwischen anderen WMS-Servern agieren, um Rasterdaten für einen Layer anzufordern. Dieser Eintrag gibt an, wieviele WMS-Server angefragt werden müssen, um die Daten zu bekommen.
- **Fixierte Höhe, Fixierte Breite** —Gibt an, ob der Layer eine feste Pixeldimension hat. Diese Version von QGIS nimmt an, dass alle WMS-Layer diesen Wert nicht gesetzt haben. Daher kann es sein, dass einige Layer komisch aussehen, die diese Funktion nicht unterstützen.
- **WGS 84 Boundingbox** —Gibt die Boundingbox eines Layers in WGS84-Koordinaten an. Einige WMS-Server setzen diese Werte nicht korrekt (z.B. stehen darin manchmal UTM-Koordinaten), sodass bei solchen Layern in QGIS der Eindruck entsteht, sehr weit herausgezoomt zu sein. Der Webmaster des WMS-Servers sollte dann auf dieses Problem aufmerksam gemacht werden. Das WMS XML-Element ist `LatLonBoundingBox`, `EX_GeographicBoundingBox` oder die `CRS:84 BoundingBox`.
- **Verfügbare Koordinatensysteme** — Die Projektionen, in denen dieser Layer dargestellt werden kann. Diese sind dem Capabilities-Dokument des Servers entnommen.
- **Verfügbare Stile** — Die Bildstile, in denen dieser Layer dargestellt werden kann.

## Temporal properties

Raster *temporal properties* (namely *Dynamic Temporal Control*) can be set for WMS and WMTS layers. By default, when a time-dimension enabled WMS or WMTS layer is added to the project, it is indicated in the *Layers* panel with the  Temporal Layer icon next to it. Its *Temporal* properties default to the *Automatic* temporal mode, meaning that the layer will follow the temporal controller's current time range by default.

You can then opt to show a specific static time value for the layer by unchecking *Dynamic Temporal Control* and picking an option under *Static WMS-T Temporal Range*:

- *Server default*
- *Predefined date* with a server exposing data for non-contiguous temporal ranges or *Predefined range* with a server exposing a range of available dates. A *Start date* and *End date* are necessary in the latter case. Their expected formatting can be deduced from the reference time option (see below). depending on whether the provider has data for contiguous period or not
- *Follow project's temporal range* as defined in the project's properties dialog

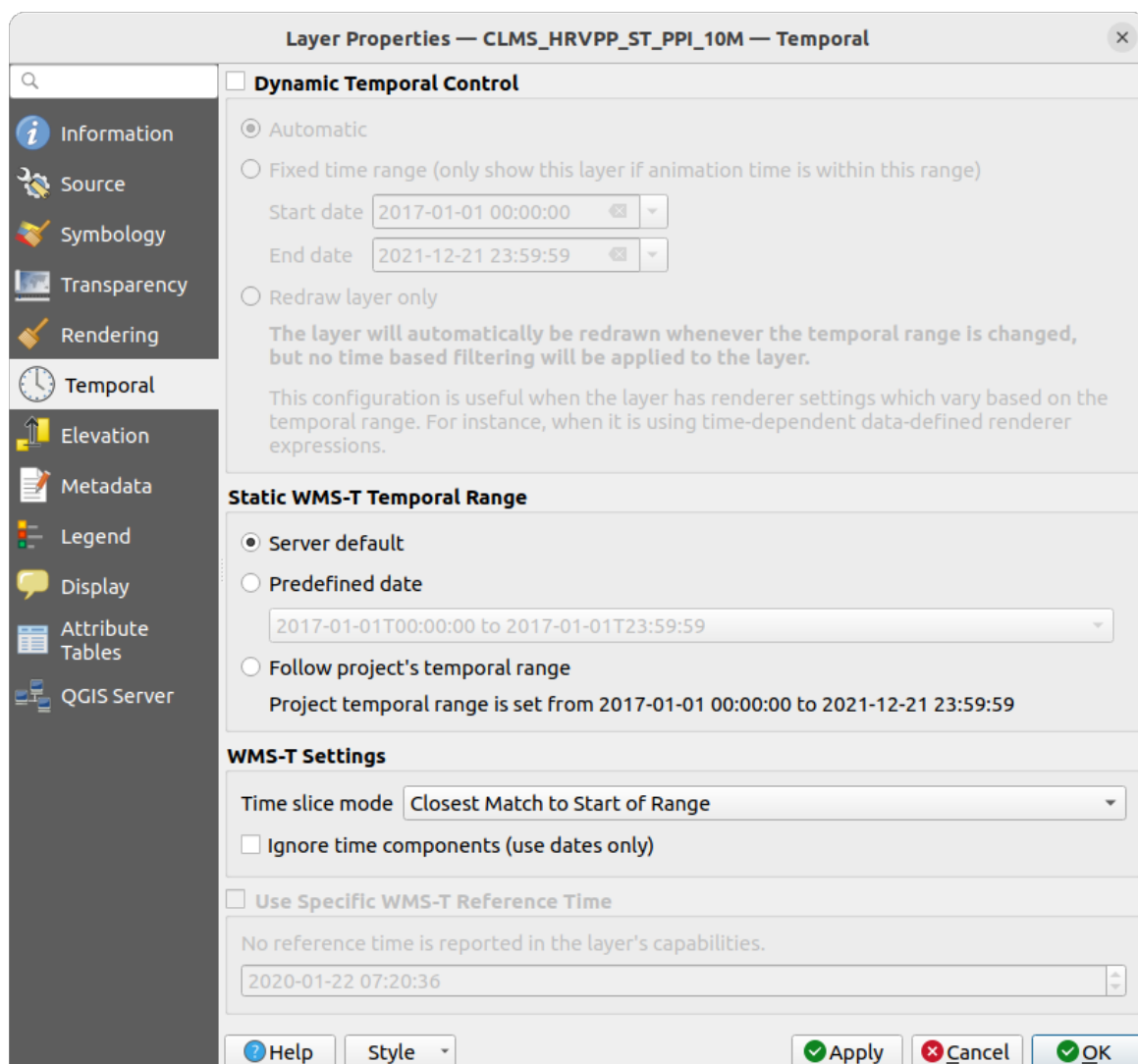



Abb. 23.4: Temporal properties of a WMTS layer

Whatever temporal data control is in use, there are some *WMS-T Settings* to help display the correct temporal data:



- *Time slice mode* which can be:

- *Use whole temporal range*
- *Match to start of range*
- *Match to end of range*
- *Closest match to start of range*
- *Closest match to end of range*
- *Ignore time components (use dates only)*: If checked, the time component of temporal queries will be discarded and only the date component will be used in server requests.

You can also  *Use Specific WMS-T Reference Time* picked from times reported in the layer's capabilities. Convenient for servers which expose a non-contiguous set of date time instances (instead of a range of dates).

### QGIS Server properties

In addition to the raster layer [properties](#), publishing a WMS/WMTS layer with QGIS Server will display following options:

- *WMS Print layer*: Allows to set alternative WMS layer that should be used for printing (in the GetProjectSettings reply). Convenient for WMTS layers that are generally not suitable for printing.
-  *Publish WMS/WMTS data source uri*: Allows a web client to fetch the WMS/WMTS data directly
-  *Advertise as background layer*

**Layer Properties — CLMS\_HRVPP\_ST\_PPI\_10M — QGIS Server**

**Description**

Short name: A name used to identify the layer. The short name is a text string used for ma...

Title: The title is for the benefit of humans to identify layer.

Abstract:

Keyword list: List of keywords separated by comma to help catalog searching.

Data URL: A URL of the data presentation. Format: [dropdown]

**Attribution**

Title: Attribution's title indicates the provider of the layer.

URL: Attribution's url gives a link to the webpage of the provider of the data layer.

**Metadata URL**

URL	Type	Format

**Legend URL**

URL: A URL of the legend image. Format: [dropdown]

**WMS Print layer**

☐ Publish WMS/WMTS data source uri

☐ Advertise as background layer

Buttons: Help, Style, Apply, Cancel, OK

Abb. 23.5: QGIS Server properties of a WMS/WMTS layer

### 23.1.8 Show WMS legend graphic in table of contents and layout

The QGIS WMS data provider is able to display a legend graphic in the table of contents' layer list and in the print layout. The WMS legend will be shown only if the WMS server has GetLegendGraphic capability and the layer has getCapability url specified, so you additionally have to select a styling for the layer.

Wenn eine legendGraphic verfügbar ist, wird diese unter dem Layer angezeigt. Sie ist klein und Sie müssen darauf klicken um Sie in Ihrer tatsächlichen Größe (gemäß den QgsLegendInterface Architekturlimitationen) zu öffnen. Das Klicken auf die Legende des Layers öffnet einen Rahmen mit einer Legende in voller Auflösung.

In the print layout, the legend will be integrated at it's original (downloaded) dimension. Resolution of the legend graphic can be set in the item properties under *Legend ► WMS LegendGraphic* to match your printing requirements.

Die Legende wird Kontextinformationen basiert auf dem aktuellen Maßstab darstellen. Die WMS Legende wird nur gezeigt wenn der WMS Server eine GetLegendGraphic Capability hat und für dem Layer eine getCapability URL angegeben wurde, also müssen Sie einen Stil auswählen.



## 23.2 WCS Client



Ein Web Coverage Service (WCS) stellt eine Anbindung zu Rasterdaten in Formaten die nützlich für die client-seitige Darstellung, als Input für wissenschaftliche Modelle und für andere Clients zur Verfügung. Der WCS ist vergleichbar zu WFS und WMS. Als WMS und WFS Service Instanz erlaubt der WCS den Clients Teile von Serverinformationsbeständen basierend auf räumlichen Einschränkungen und Abfragekriterien auszuwählen.

QGIS hat einen nativen WCS Provider und unterstützt sowohl Version 1.0 und 1.1 (welche deutliche Unterschiede aufweisen), aktuell jedoch wird 1.0 vorgezogen da es mit 1.1 viele Probleme gibt (z.B. implementiert jeder Server es auf eine andere Art und Weise mit zahlreichen Besonderheiten).

Der native WCS Provider kümmert sich um alle Netzwerkanfragen und verwendet alle voreingestellten QGIS Netzwerkanfragen (insbesondere proxy). Es ist auch möglich einen Cache auszuwählen („Immer cachén“, „Cache vorziehen“, „Netzwerk vorziehen“, „Immer Netzwerk“) und der Provider unterstützt ebenfalls das Auswählen der Zeitposition wenn Temporal Domain vom Server angeboten wird.

**Warnung:** Wenn Sie **Benutzername** und **Passwort** im Tab *Authentifizierung* speichern, werden diese Geheimnisse ungeschützt in der Verbindungskonfiguration gespeichert. Die **Geheimnisse sind sichtbar** wenn Sie z.B. die Projektdatei an andere weitergeben. Deshalb ist es ratsam, die Geheimnisse stattdessen in einer *Authentifizierungskonfiguration* (Tab *Konfigurationen*). Schlagen Sie Details in [Authentifizierungssystem](#) nach.

## 23.3 WFS und WFS-T Klient

In QGIS, a WFS layer behaves pretty much like any other vector layer. You can identify and select features, and view the attribute table. QGIS supports WFS 1.0.0, 1.1.0, 2.0 and OGC API - Features (OAPIF), including editing (through WFS-T). QGIS also supports background download and progressive rendering, on-disk caching of downloaded features and version autodetection.

Layers of servers implementing [OGC API - Features - Part 4: Create, Replace, Update and Delete](#) can be turned into editing mode to allow creating, modifying and deleting features. Note that each created/modified/deleted feature requires a dedicated network request, so performance might suffer in case of simultaneous modification of hundreds or more features at a time.



In general, adding a WFS layer is very similar to the procedure used with WMS. There are no default servers defined, so you have to add your own. You can find WFS servers by using the [MetaSearch plugin](#) or your favourite web search engine. There are a number of lists with public URLs, some of them maintained and some not.

### Einen WFS-Layer laden

As an example, we use the Gateway Geomatics WFS server and display a layer.

```
https://demo.gatewaygeomatics.com/cgi-bin/wfs_gateway?REQUEST=GetCapabilities&
→VERSION=1.0.0&SERVICE=WFS
```

To be able to load a WFS Layer, first create a connection to the WFS server:

1. Open the *Data Source Manager* dialog by pressing the  **Open Data Source Manager** button
2. Enable the  **WFS / OGC API - Features** tab
3. Click on *New...* to open the *Create a New WFS Connection* dialog
4. Enter **Gateway Geomatics** as name
5. Enter the URL (see above)

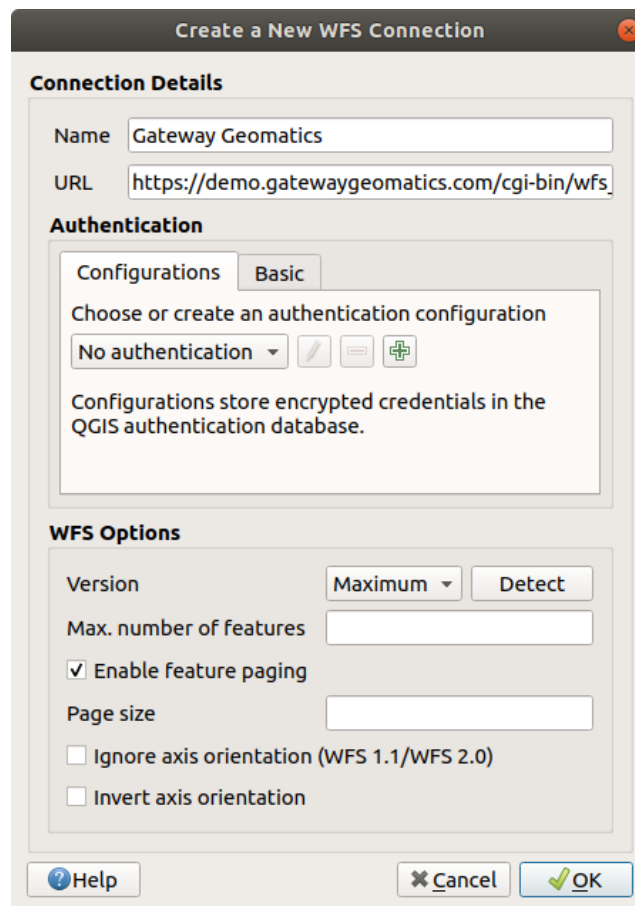


Abb. 23.6: Creating a connection to a WFS server

---

**Bemerkung:** In case of an OGC API - Features (OAPIF), the URL to provide should be the landing page, ie the main page from which it is possible to navigate to all the available service endpoints.

---

6. In the WFS settings dialog, you can:

- Indicate the WFS version of the server. If unknown, press the *Detect* button to automatically retrieve it.
- Define the *maximum number of features* retrieved in a single GetFeature request. If empty, no limit is set.
- And depending on the WFS version, indicate whether to:
  - *Enable feature paging* and specify the maximum number of features to retrieve with *Page size*. If no limit is defined, then the server default is applied.
  - Force to *Ignore axis orientation (WFS 1.1/WFS 2.0)*
  - *Invert axis orientation*.
  - *Use GML2 encoding for transactions*.


**Warnung:** Entering **username** and **password** in the *Authentication* tab will keep unprotected credentials in the connection configuration. Those **credentials will be visible** if, for instance, you shared the project file with someone. Therefore, it's advisable to save your credentials in an *Authentication configuration* instead (*Configurations* tab). See [Authentifizierungssystem](#) for more details.

7. Press *OK* to create the connection.

Note that any proxy settings you may have set in your preferences are also recognized. Also note that it is possible to *Load* the connection parameters from a .XML file or *Save* them to a .XML file.

Now we are ready to load WFS layers from the above connection.

1. Choose ‚Gateway Geomatics‘ from the *Server Connections* drop-down list.
2. Click *Connect*
3. Select the *Parks* layer in the list
4. You can also choose whether to:
  - ☐ *Use title for layer name*, showing the layer’s title as defined on the server in the *Layers* panel instead of its *Name*
  - ☒ *Only request features overlapping the view extent*
  - *Change...* the layer’s CRS to any other supported by the service
  - or build a query to specify particular features to retrieve from the service: double-click on the row of the layer to open the *SQL Query Composer* dialog. That dialog provides widgets to write an advanced SQL query relying on available tables and columns of the service, with sorting and filtering and a bunch of SQL functions, spatial predicates and operators.

The query you build will appear after validation in the *SQL* column within the *WFS / OGC API - Features* table, and the filtered layer will display the  icon next to it in the *Layers* panel. It is thus possible to adjust the query at any moment.

5. Click *Add* to add the layer to the map.

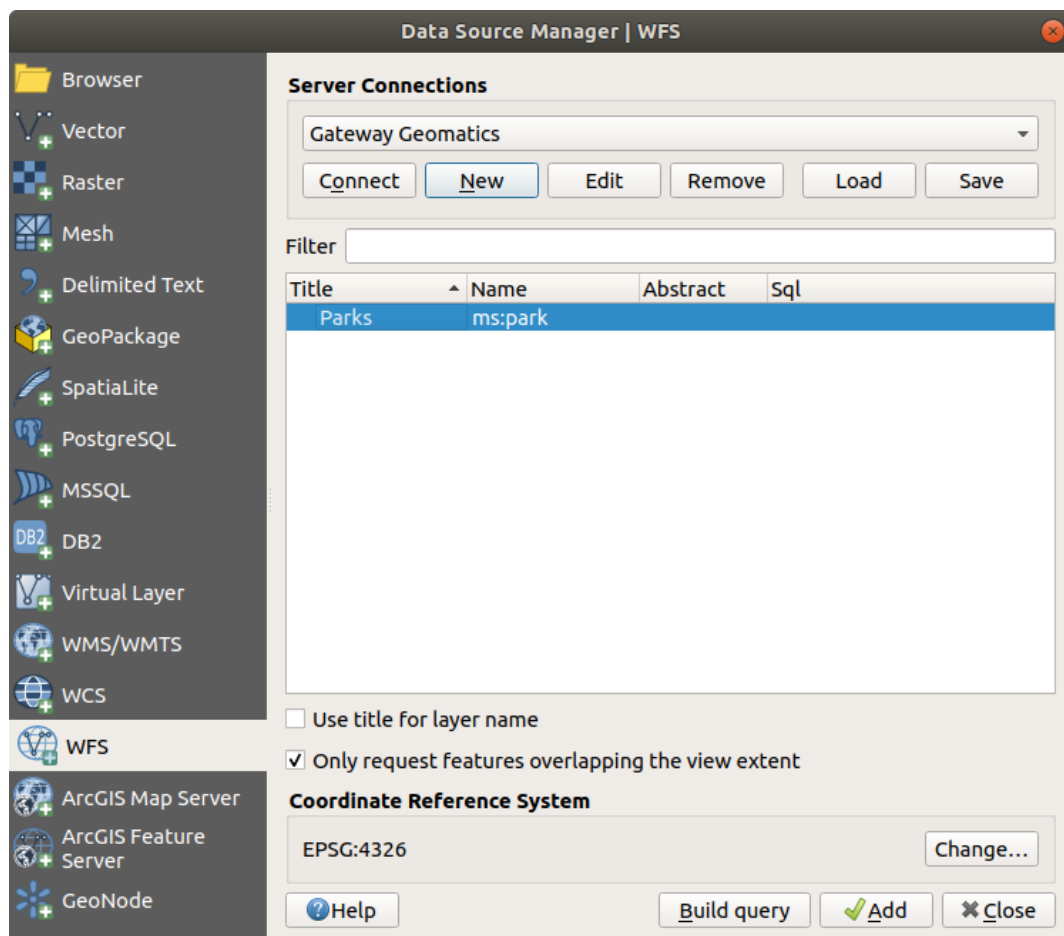


Abb. 23.7: Einen WFS Layer hinzufügen

You'll notice the download progress is visualized in the lower left of the QGIS main window. Once the layer is loaded, you can identify and select a couple of features and view the attribute table.

### 24.1 Introducing GNSS/GPS Data

#### 24.1.1 Was ist GPS?

GPS, the **Global Positioning System**, is a satellite-based system that allows anyone with a GPS receiver to find their exact position anywhere in the world. GPS is used as an aid in navigation, for example in airplanes, in boats and by hikers. The GPS receiver uses the signals from the satellites to calculate its latitude, longitude and (sometimes) elevation. Most receivers also have the capability to store:

- locations (known as **waypoints**)
- sequences of locations that make up a planned **route**
- and a **track** log of the receiver's movement over time.

Waypoints, routes and tracks are the three basic feature types in GPS data. QGIS displays waypoints in point layers, while routes and tracks are displayed in linestring layers.

---

**Bemerkung:** QGIS supports also GNSS receivers. But we keep using the term GPS in this documentation.

---


#### 24.1.2 Transferring or loading GPS data


##### Loading a GPX file

Es gibt viele verschiedene Datenformate zum Speichern von GPS-Daten. Das von QGIS unterstützte Format ist GPX (GPS eXchange format), ein Standardformat, welches Wegpunkte, Routen und Spuren in einer Datei enthalten kann.

To load a GPX file:

1. Open the *GPS* tab in the *Data Source Manager* dialog, i.e.:

- Click the  Open Data Source Manager button on the toolbar (or press `Ctrl+L`) and enable the target tab

- Or select *Layer ► Add Layer ►  Add GPX Layer...*
2. Use the ... <sup>Browse</sup> button next to the *GPX dataset* option to select the GPX file
  3. Use the check boxes to select the *Feature types* you want to load from the file. Each feature type (*Waypoints*, *Tracks* or *Routes*) will be loaded in a separate layer.

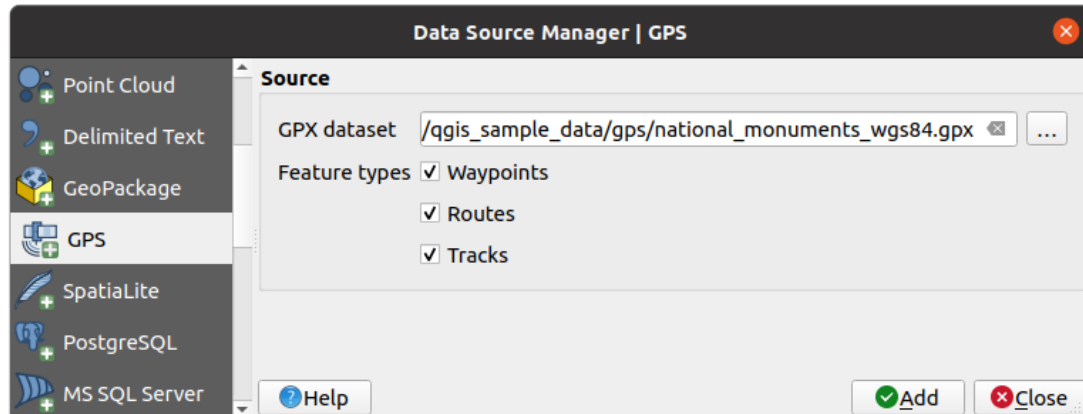





Abb. 24.1: Loading GPS Data dialog

### Loading to or from a device

There are lots of different types of GPS devices and formats. Since QGIS uses GPX files, you need a way to convert other GPS file formats to GPX. QGIS can do that using the free program [GPSBabel](#). GPSBabel can help you convert waypoints, tracks, and routes between popular GPS receivers such as Garmin or Magellan and mapping programs like Google Earth or Basecamp. Literally hundreds of GPS receivers and programs are supported. It can also transfer GPS data between your computer and a GPS device.

Under *Settings ►  Options ►  GPS ►  GPSBabel*, QGIS allows you to define your own device type and set parameters of conversion that could later be used by the *Processing GPS algorithms*.

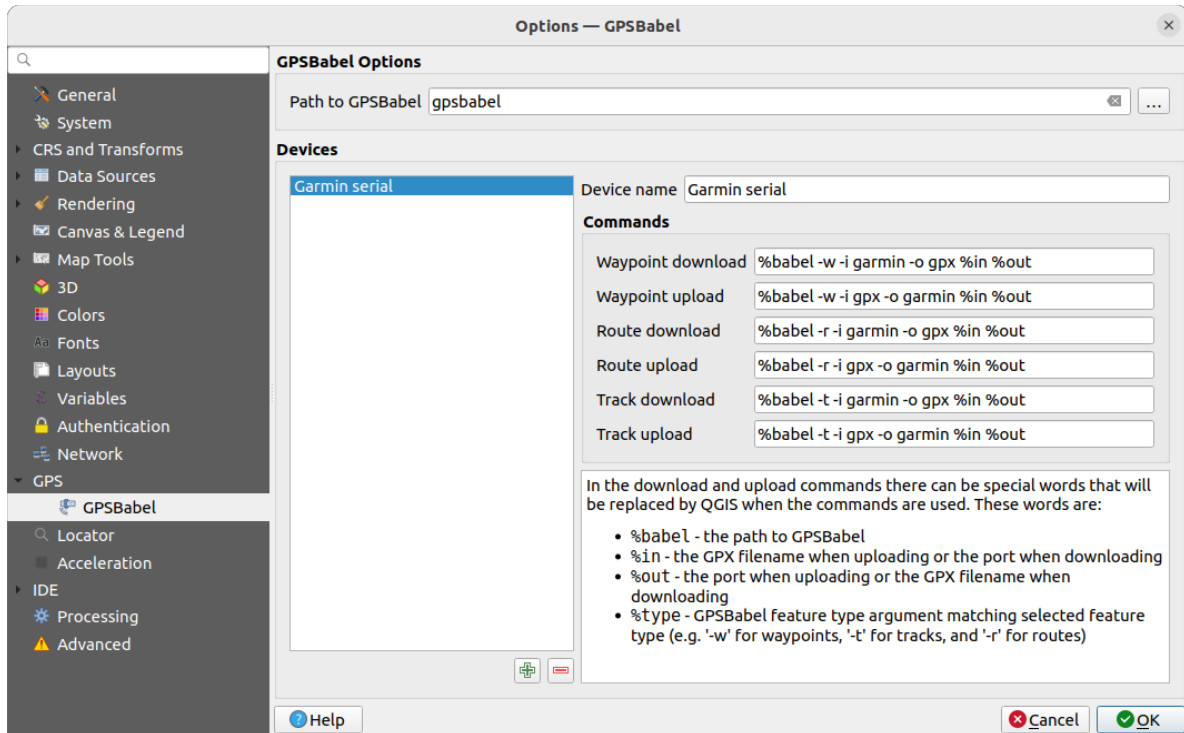




Abb. 24.2: GPS Babel settings

1. First you have to define the *Path to GPSBabel* binaries.
2. Then you may want to add your device. You can update devices list using  Add new device or  Remove device button.
3. For each device:
  - you provide a *Device name*
  - you configure different *Commands* QGIS will use while interacting with it, such as:
    - *Waypoint download* from the device
    - *Waypoint upload* to the device
    - *Route download* from the device
    - *Route upload* to the device
    - *Track download* from the device
    - *Track upload* to the device

While the commands are usually GPSBabel commands, you can also use any other command line program that can create a GPX file. QGIS will replace the keywords %type, %in, and %out when it runs the command.

As an example, if you create a device type with the download command `gpsbabel %type -i garmin -o gpx %in %out` and then use it to download waypoints from port `/dev/ttyS0` to the file `output.gpx`, QGIS will replace the keywords and run the command `gpsbabel -w -i garmin -o gpx /dev/ttyS0 output.gpx`.

Read the GPSBabel manual for the command line options that may be specific to your use case.

Once you have created a new device type, it will appear in the device lists for the GPS download and upload algorithms.

**Bemerkung:** GPS units allow you to store data in different coordinate systems. When downloading a GPX file (from your GPS unit or a web site) and then loading it in QGIS, be sure that the data stored in the GPX file uses WGS 84

(latitude/longitude). QGIS expects this, and it is the official GPX specification. See [GPX 1.1 Schema Documentation](#).





---

## 24.2 Live GPS tracking






QGIS can help you do field mapping with a GPS receiver. Such a live tracking operation is done using the *GPS Toolbar*. Some *device configuration* may be needed before connecting QGIS and the GPS receiver.

### 24.2.1 GPS toolbar











The *GPS Toolbar* provides the main tools to control a live tracking session. It can be activated from *View ► Toolbars ► GPS*. It follows the state of the project, GPS, and current GPS track, and enables actions only when they make sense. Once a device is detected by QGIS, you will be able to interact with it:

-  Connect to GPS
-  Disconnect from GPS
-  Recenter map on GPS location: the map will immediately recenter on the current GPS position regardless of the *automatic recentering parameter*.
-  Set destination layer for GPS digitized features: By default, QGIS will *Follow active layer* for feature digitizing, meaning that GPS digitizing tools will adapt to the layer selected in *Layers* panel and created features will be stored in it accordingly. While this may be useful in some contexts, it also implies careful interaction with other layers to avoid storing inadvertently features in undesired layers. This option allows you to explicitly indicate a layer for the data storage during the live tracking session, and you could switch depending on your needs. Exposed tools will adapt as well to the selected layer type.

The GPS destination layer is made automatically editable when a feature is being created, and the user is notified accordingly.



- actions for feature creation:
  -  Add vertex to GPS track using GPS current location
  -  Create feature from track
  -  Create a new point feature at current GPS location
  -  Reset track
-  Show GPS information: opens the *GPS Information* panel
- a display box for quick access to some GPS information components you may need:
  - ☐ Show location
  - ☐ Show altitude (Geoid)
  - ☐ Show altitude (WGS 84 ellipsoid)
  - ☐ Show ground speed
  - ☐ Show bearing
  - ☐ Show total track length
  - ☐ Show distance from start of track



- The  **Settings** button opens a drop-down menu with common settings which are expected to be modified mid-session:
  -  *Show location marker*
  -  *Show bearing line*
  -  *Rotate map to match GPS direction*
  - Options to control map recentering:
    - \*  *Always recenter map* when the GPS position is offset from the map canvas center by a certain distance (as a ratio of the map canvas extent)
    - \*  *Recenter map when leaving extent*
    - \*  *Never recenter*
  -  *Automatically add track vertices* whenever new locations are received from the GPS device.
  -  *Automatically save added features*: features created from GPS locations are immediately committed to their target layers (skipping the usual layer edit buffer)
  - *Time stamp destination* to adjust the field for time fix storage
  -  *Log to GeoPackage/Spatialite...*: When activated, the user will be prompted to select an existing GeoPackage/Spatialite file or enter a new file name. A `gps_points` and `gps_tracks` table will be created in the file with a predefined structure.

All incoming GPS messages will be logged to the `gps_points` layer, along with speed, bearing, altitude and accuracy information from the GPS.

When the GPS is disconnected (or QGIS closed), the entire recorded GPS track will be added to the `gps_tracks` table (along with some calculated information like track length, start and end times).

  -  *Log NMEA sentences...*: enables logging all raw NMEA strings from the device to a text file
  -  *GPS settings...* to access the GPS *global options* dialog

---

**Tipp: Live status bar information on position**

When a GPS device is connected and the user moves the cursor over the map canvas, a live status bar message displays the distance and bearing from the cursor to the GPS position. Project distance and bearing settings are respected in this display. On a touch screen device, use a tap-and-hold event to trigger the live status bar message.




---

## 24.2.2 GPS Information Panel


For full monitoring of a live GPS tracking in QGIS, you may need to enable the *GPS Information Panel* (*View ► Panels ►* or press `Ctrl+0`).

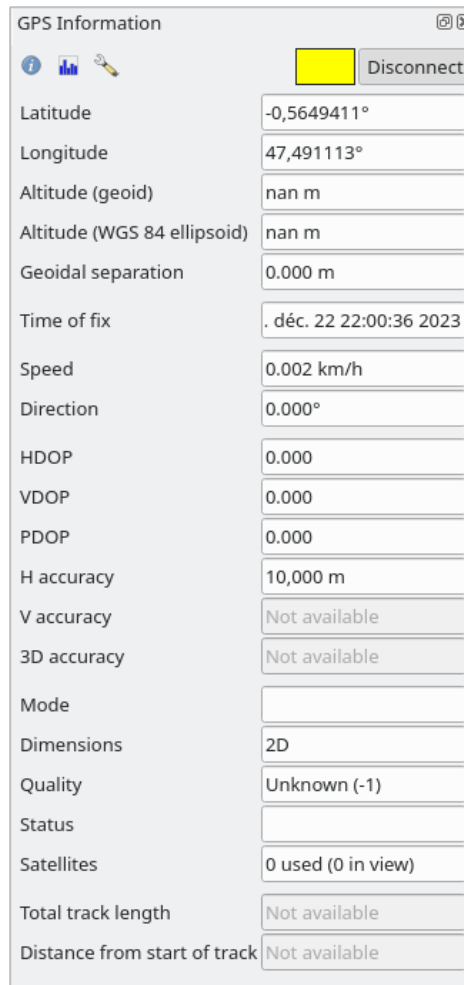
In the top right corner of the *GPS Information Panel*, you press *Connect* to initiate connection between QGIS and a plugged-in GPS receiver, or *Disconnect* them.

In the top left of the panel, following buttons are accessible:

-  **Position**: live details on the GPS position and sensors
-  **Signal**: signal strength of satellite connections
-  **Settings**: drop-down menu to *live tracking options* you may need to modify during a session

## Positionskoordinaten

In the  Position tab, if the GPS is receiving signals from satellites, you will see your position in latitude, longitude and altitude together with additional attributes.




GPS Information	
Latitude	-0,5649411°
Longitude	47,491113°
Altitude (geoid)	nan m
Altitude (WGS 84 ellipsoid)	nan m
Geoidal separation	0.000 m
Time of fix	. déc. 22 22:00:36 2023
Speed	0.002 km/h
Direction	0.000°
HDOP	0.000
VDOP	0.000
PDOP	0.000
H accuracy	10,000 m
V accuracy	Not available
3D accuracy	Not available
Mode	
Dimensions	2D
Quality	Unknown (-1)
Status	
Satellites	0 used (0 in view)
Total track length	Not available
Distance from start of track	Not available

Abb. 24.3: GPS Tracking Position und zusätzliche Attribute

- *Latitude*
- *Longitude*
- *Altitude (Geoid)*: Altitude/elevation above or below the mean sea level
- *Altitude (WGS 84 ellipsoid)*: Altitude/elevation above or below the WGS-84 Earth ellipsoid
- *Geoidal separation*: Difference between the WGS-84 Earth ellipsoid and mean sea level (geoid), – means mean sea level is below ellipsoid
- *Time of fix*
- *Speed*: Ground speed
- *Direction*: Bearing measured in degrees clockwise from true north to the direction of travel
- *HDOP*: Horizontal dilution of precision
- *VDOP*: Vertical dilution of precision
- *PDOP*: Dilution of precision
- *H accuracy*: Horizontal accuracy in meters

- *V accuracy*: Vertical accuracy in meters
- *3D accuracy*: 3D Root Mean Square (RMS) in meters
- *Mode*: GPS receiver configuration 2D/3D mode, can be `automatic` or `manual`
- *Dimensions*: Position fix dimension, can be 2D, 3D or `No fix`
- *Quality*: Positioning quality indicator
- *Status*: Position fix status, can be `Valid` or `Invalid`
- *Satellites*: Count of satellites used in obtaining the fix
- *Total track length*: Total distance of current GPS track
- *Distance from start of track*: Direct distance from first vertex in current GPS track to latest vertex

## Signal

With  `Signal` tab, you can see the signal strength of the satellites you are receiving signals from.

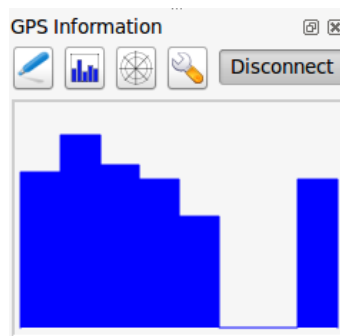


Abb. 24.4: GPS Tracking Signalstärke

### 24.2.3 Ein Bluetooth GPS fürs Live Tracking anbinden


Mit **lqgl** können Sie ein Bluetooth GPS für das Aufnehmen von Felddaten anbinden. Um dies durchzuführen benötigen Sie ein Bluetooth Gerät und einen Bluetooth Empfänger auf Ihrem Computer.

Als erstes müssen Sie Ihr GPS Gerät erkennen lassen und mit dem Computer verbinden. Stellen Sie das GPS an, gehen Sie zum Bluetooth Icon in Ihrem Infobereich und suchen Sie nach einem Neuen Gerät.

On the right side of the Device selection mask make sure that all devices are selected so your GPS unit will probably appear among those available. In the next step a serial connection service should be available, select it and click on *Configure* button.

Denken Sie daran dass die an die GPS Verbindung angebotenen COM Ports aus den Bluetooth Eigenschaften resultieren.

Machen Sie die Kopplung für die Verbindung nachdem das GPS erkannt wurde. Normalerweise ist der Authorisationscode 0000.

Now open *GPS information* panel and switch to  GPS options screen. Select the COM port assigned to the GPS connection and click the *Connect*. After a while a cursor indicating your position should appear.

Wenn **lqgl** keine GPS Daten empfangen kann sollten Sie Ihr GPS Gerät neustarten, 5-10 Sekunden warten und dann wieder eine Verbindung versuchen. Normalerweise funktioniert diese Lösung. Wenn Sie wieder einen Verbindungsfehler erhalten vergewissern Sie sich dass kein anderer Bluetoothempfänger, der an die gleiche GPS Einheit gekoppelt ist, in Ihrer Nähe ist.


## 24.2.4 Examples of GPS devices connection in QGIS

### GPSPMAP 60cs verwenden

#### MS Windows

Easiest way to make it work is to use a middleware (freeware, not open) called [GPSPGate](#).

Launch the program, make it scan for GPS devices (works for both USB and BT ones) and then in QGIS:

1. In *Settings* ► *Options* ► *GPS*, detect the plugged-in device. You can use the  *Autodetect* mode.
2. Press *Connect* in the *GPS Information* panel

#### Ubuntu/Mint GNU/Linux

Wie unter Windows ist der einfachste Weg einen Server, in diesem Fall GPSPD, dazwischen zu benutzen, also

1. Install the program

```
sudo apt install gpsd
```

2. Laden Sie dann das `garmin_gps` Kernelmodul

```
sudo modprobe garmin_gps
```

#. And then connect the unit. Then check with `dmesg` the actual device being used by the unit, for example `/dev/ttyUSB0`.

1. Now you can launch `gpsd`

```
gpsd /dev/ttyUSB0
```


2. Und verbinden Sie sich zuletzt mit dem **lqgl** Live Tracking Tool.

### BTGP-38KM Datenlogger verwenden (nur Bluetooth)

Sie können GPSPD (unter Linux) oder GPSPGate (unter Windows) mühelos verwenden.

### BlueMax GPS-4044 Datenlogger verwenden (sowohl BT als auch USB)

#### MS Windows

The live tracking works for both USB and BT modes, by using GPSPGate or even without it, just use the  *Autodetect* mode, or point the tool to the right port.

## Ubuntu/Mint GNU/Linux

### Für USB

Das Live Tracking funktioniert sowohl mit GPSD

```
gpsd /dev/ttyACM3
```

oder ohne es, indem man das **lqgl** Live Tracking Tool direkt mit dem Gerät verbindet (zum Beispiel /dev/ttyACM3).

### Für Bluetooth

Das Live Tracking funktioniert sowohl mit GPSD

```
gpsd /dev/rfcomm0
```

oder ohne es, indem man das **lqgl** Live Tracking Tool direkt mit dem Gerät verbindet (zum Beispiel /dev/rfcomm0).



## 25.1 Authentifizierungssystem Übersicht

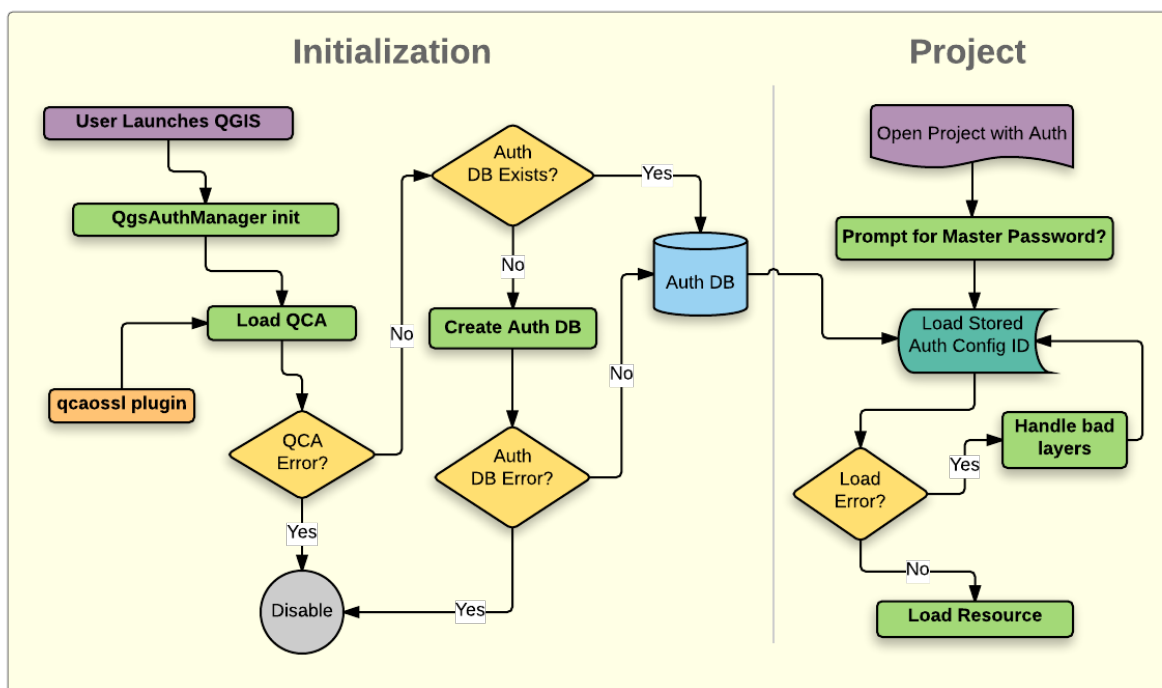


Abb. 25.1: Anatomie des Authentifizierungssystems

## 25.1.1 Authentifikationsdatenbank

The new authentication system stores authentication configurations in an SQLite database file located, by default, at `<profile directory>/qgis-auth.db`.

Diese Authentifizierungsdatenbank kann ohne Auswirkung auf andere aktuelle QGIS Benutzereinstellungen zwischen QGIS Installationen bewegt werden, da sie völlig unabhängig von normalen QGIS-Einstellungen ist. Eine Konfigurations-ID (eine zufällige 7-stellige alphanumerische Zeichenfolge) wird erzeugt, wenn eine Konfiguration zu der Datenbank gespeichert wird. Dies stellt die Konfiguration dar, wodurch die ID im Klartext als Anwendungskomponenten gespeichert wird (z. B. Projekt-, Plugin- oder Einstellungsdateien) ohne Offenlegung ihrer zugehörigen Anmeldeinformationen.

---

**Bemerkung:** Das übergeordnete Verzeichnis des *qgis-auth.db* kann mit der folgenden Umgebungsvariablen gesetzt werden, `QGIS_AUTH_DB_DIR_PATH`, oder mit der `--authdbdirectory` Option während des Starts auf der Kommandozeile.

---

## 25.1.2 Hauptpasswort

To store or access sensitive information within the database, a user must define a *master password*. A new master password is requested and verified when initially storing any encrypted data to the database. When sensitive information is accessed, the user is prompted for the master password. The password is then cached for the remainder of the session (until application is quit), unless the user manually chooses an action to clear its cached value. Some instances of using the authentication system do not require input of the master password, such as when selecting an existing authentication configuration, or applying a configuration to a server configuration (such as when adding a WMS layer).

You can choose to save the password in the Wallet/Keyring of your computer.

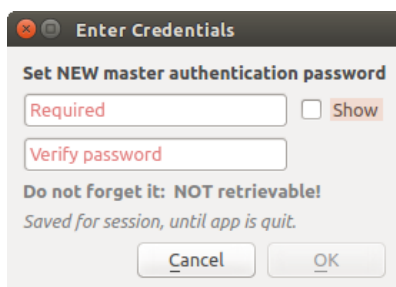


Abb. 25.2: Neues Hauptpasswort eingeben

---

**Bemerkung:** Ein Pfad zu einer Datei, die das Hauptpasswort beinhaltet, kann mit Hilfe der Umgebungsvariablen eingestellt werden `QGIS_AUTH_PASSWORD_FILE`.

---



## Hauptpasswort verwalten

Einmal eingestellt, kann das Hauptpasswort zurückgesetzt werden, das aktuelle Hauptpasswort wird vor dem Zurücksetzen jedoch benötigt werden. Während dieses Prozesses ist es eine Option, eine vollständige Sicherung der aktuellen Datenbank vorzunehmen.

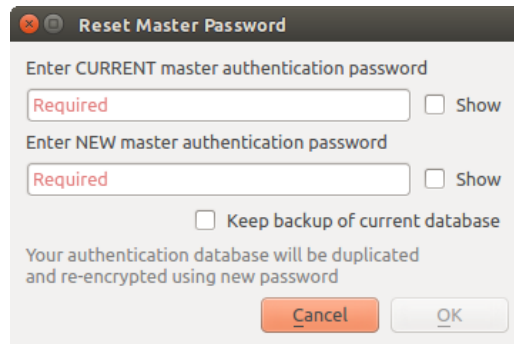


Abb. 25.3: Hauptpasswort zurücksetzen

Wenn der Benutzer das Hauptpasswort vergisst, gibt es keine Möglichkeit es abzurufen oder außer Kraft setzen. Es gibt auch keine Mittel die verschlüsselten Informationen abzurufen ohne das Master-Passwort zu kennen.

Wenn ein Benutzer das bestehende Passwort drei Mal falsch eingibt, wird ein Dialog Ihnen anbieten, die Datenbank zu löschen.

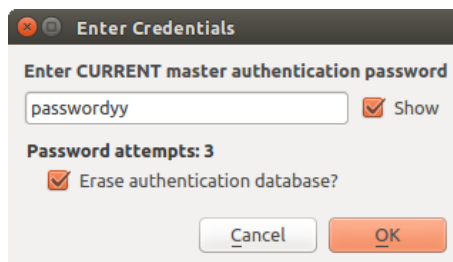


Abb. 25.4: Passwortabfrage nach drei ungültigen Versuchen

### 25.1.3 Authentifikationskonfiguration

Sie können Authentifizierungseinstellungen über *Konfigurationen* in dem *Authentifizierung* Reiter des QGIS Optionen Dialog (*Einstellungen* ► *Optionen*) vornehmen.

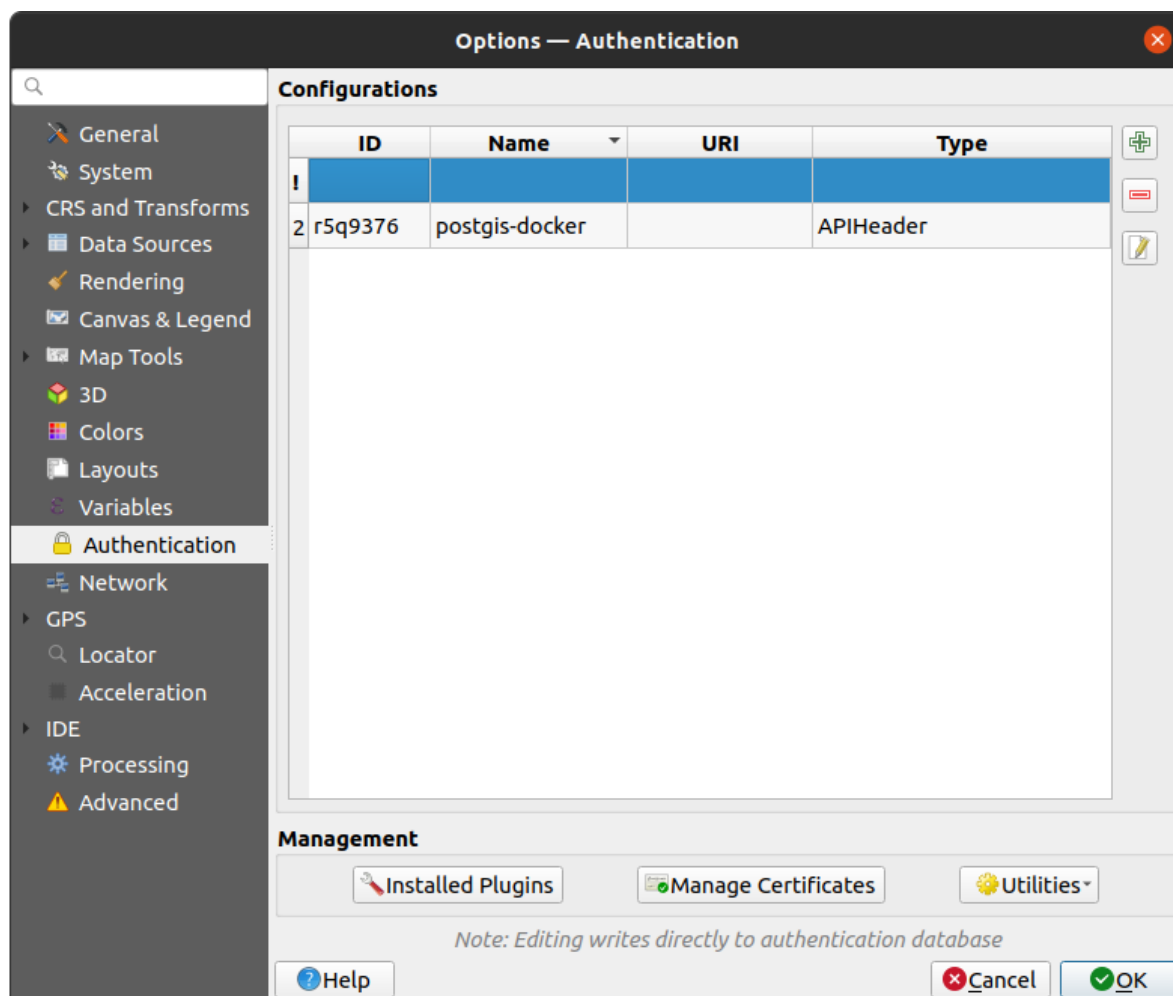





Abb. 25.5: Konfigurationen Editor

Nutzen Sie den  Knopf, um neue Konfigurationen hinzuzufügen, den  Knopf um Sie zu entfernen und den  Knopf um bestehende zu bearbeiten.

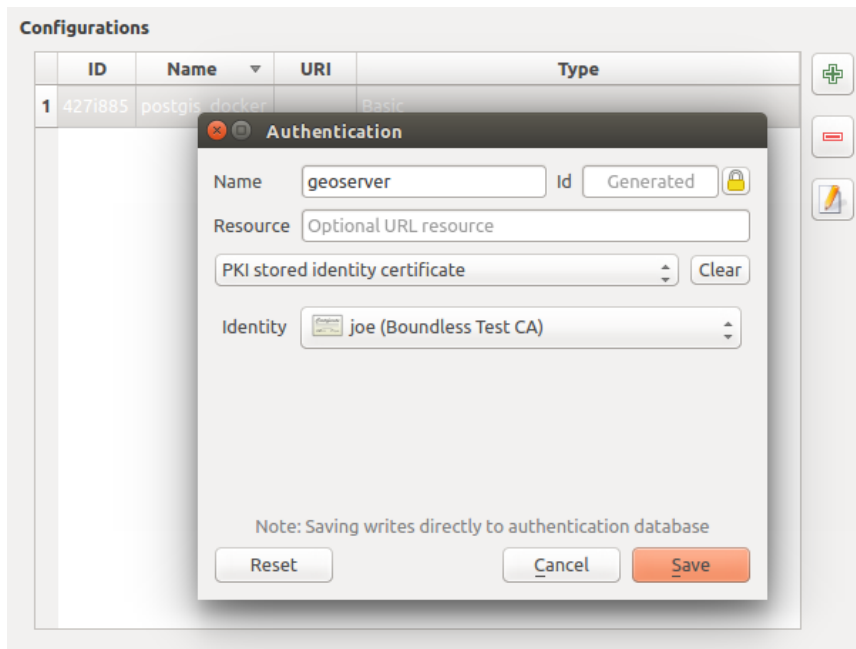


Abb. 25.6: Konfigurationen von innerhalb des Konfigurationseditors hinzufügen

Die selben Arten von Tätigkeiten für Authentifikationskonfigurationen (Hinzufügen, Bearbeiten und Entfernen) können auch bei der Konfiguration einer Service-Verbindung, wie der Konfiguration einer OWS-Service-Verbindung, getätigt werden. Dafür gibt es Aktionsschaltflächen innerhalb der Konfigurationsauswahl für die vollständige Verwaltungskonfigurationen die man innerhalb der Authentifikationsdatenbank findet. In diesem Fall ist es nicht erforderlich in den *Konfigurationen* im *Authentifikation* Reiter zu wechseln, es sei denn Sie müssen weitere umfassende Konfigurationen vornehmen.

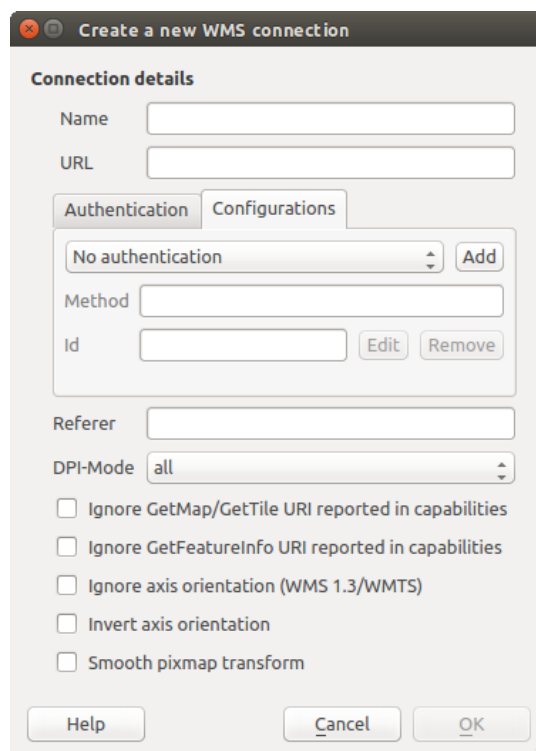



Abb. 25.7: WMS connection dialog showing *Add*, *Edit*, and *Remove* authentication configuration buttons

Beim erstellen oder bearbeiten einer Authentifikationskonfiguration, ist ein Name, eine Authentifizierungsmethode oder eine andere Information, die die Authentifizierungsmethode benötigt, erforderlich (mehr über die verfügbaren Authentifikationstypen in [Authentifizierungsmethoden](#)).

### 25.1.4 Authentifizierungsmethoden

Available authentications are provided by C++ plugins much in the same way data provider plugins are supported by QGIS. The method of authentication that can be selected is relative to the access needed for the resource/provider, e.g. HTTP(S) or database, and whether there is support in both QGIS code and a plugin. As such, some authentication method plugins may not be applicable everywhere an authentication configuration selector is shown. A list of available authentication method plugins and their compatible resource/providers can be accessed going to *Settings ► Options* and, in the *Authentication* tab, click the  *Installed Plugins* button.

Installed authentication method plugins		
Method	Description	Works with
Basic	Basic authentication	postgres, db2, ows, wfs, wcs, wms, ogr, gdal, proxy
EsriToken	ESRI token based authentication	arcgismapserver, arcgisfeatureserver
Identity-Cert	PKI stored identity certificate	ows, wfs, wcs, wms, postgres
OAuth2	OAuth2 authentication	ows, wfs, wcs, wms
PKI-Paths	PKI paths authentication	ows, wfs, wcs, wms, postgres
PKI-PKCS#12	PKI PKCS#12 authentication	ows, wfs, wcs, wms, postgres

Abb. 25.8: Verfügbare Liste von Methodenerweiterungen

Plugins can be created for new authentication methods that do not require QGIS to be recompiled. Since the support for plugins is currently C++-only, QGIS will need to be restarted for the new dropped-in plugin to become available to the user. Ensure your plugin is compiled against the same target version of QGIS if you intend to add it to an existing target install.

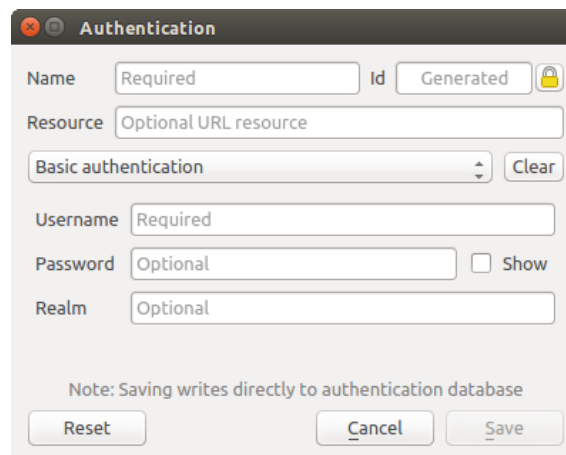


Abb. 25.9: Einfache HTTP Authentifikationskonfiguration

Name Required Id Generated

Resource Optional URL resource

ESRI token based authentication Clear

Token Required

*Note: Saving writes directly to authentication database*

Reset Cancel Save

Abb. 25.10: ESRI Token authentication configs

Name  Id

Resource

OAuth2 authentication

Configure Defined Software S

Grant Flow

Description

Request URL

Token URL

Refresh Token URL

Redirect URL

Client ID

Client Secret

Scope

API Key

*Advanced*

Token Session ☐ Persist between launches

Access Method

Request Timeout

☒ **Extra initial request parameters**

Key	Value (unencoded)
<input type="text"/>	

*Note: Saving writes directly to authentication database*

Abb. 25.11: OAuth2 authentication configs

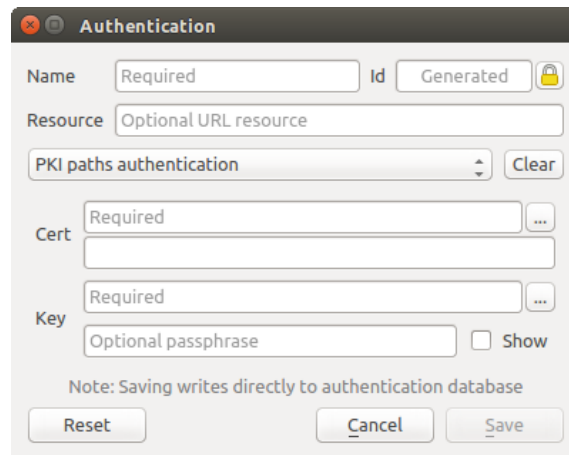


Abb. 25.12: PKI Pfad-Authentifikationskonfiguration

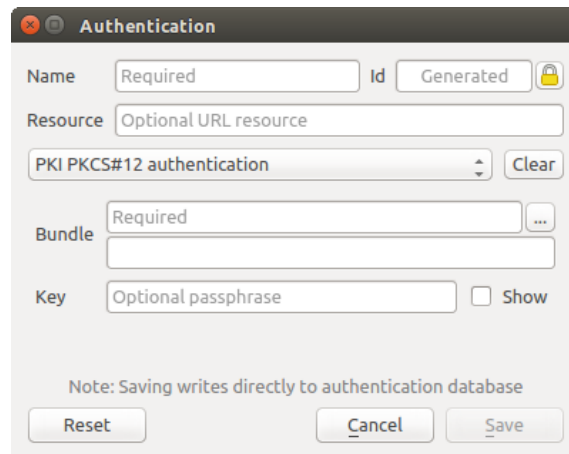


Abb. 25.13: PKI PKCS#12 Dateipfad Authentifikationskonfiguration

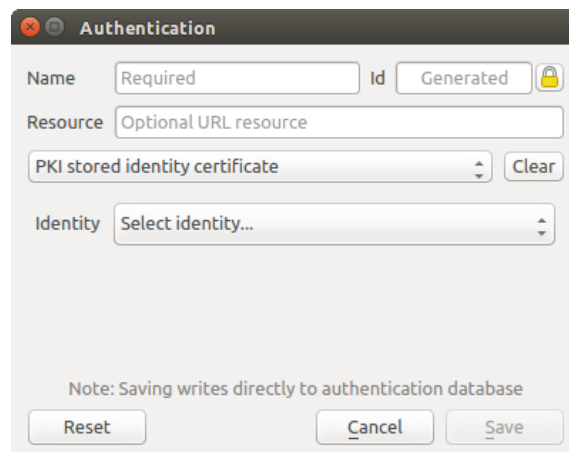


Abb. 25.14: Gespeicherte Identität Authentifikationskonfigurationen

**Bemerkung:** Die URL-Ressource ist derzeit eine *nicht ausführbare* Funktion, die eine bestimmte Konfiguration automatisch wählt, wenn sie sich mit einer Ressource und gegebenen URL verbindet.

## 25.1.5 Hauptpasswort und Auth Konfig Hilfsmittel

Unter dem Optionen-Menü (*Einstellungen ► Optionen*) in der *Authentifikation* Registerkarte gibt es verschiedene Werkzeuge die Authentifikationsdatenbank und -konfigurationen zu verwalten:

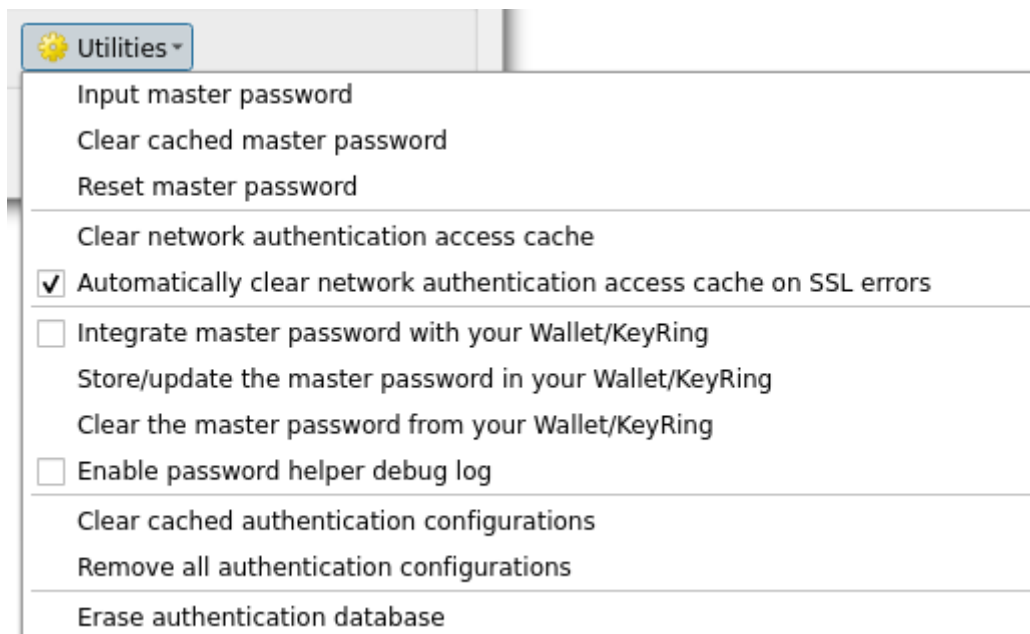


Abb. 25.15: Werkzeug-Menü

- **Input master password:** opens the master password input dialog, independent of performing any authentication database command
- **Clear cached master password:** unsets the master password if it has been set
- **Reset master password:** opens a dialog to change the master password (the current password must be known) and optionally back up the current database
- **Clear network authentication access cache:** clears the authentication cache of all connections
- **Automatically clear network authentication access cache on SSL errors:** the connection cache stores all authentication data for connections, also when the connection fails. If you change authentication configurations or certification authorities, you should clear the authentication cache or restart QGIS. When this option is checked, the authentication cache will be automatically cleared every time an SSL error occurs and you choose to abort the connection
- **Integrate master password with your Wallet/Keyring:** adds the master password to your personal Wallet/Keyring
- **Store/update the master password in your Wallet/Keyring:** updates the changed master password in your Wallet/Keyring
- **Clear the master password from your Wallet/Keyring:** deletes the master password from your Wallet/Keyring
- **Enable password helper debug log:** enables a debug tool that will contain all the log information of the authentication methods
- **Clear cached authentication configurations:** clears the internal lookup cache for configurations, used to speed up network connections. This does not clear QGIS's core network access manager's cache, which requires a relaunch of QGIS.
- **Remove all authentication configurations:** clears the database of all configuration records, without removing other stored records.



- **Erase authentication database:** schedules a backup of the current database and complete rebuild of the database table structure. The actions are scheduled for a later time, to ensure that other operations, like project loading, do not interrupt the operation or cause errors due to a temporarily missing database.

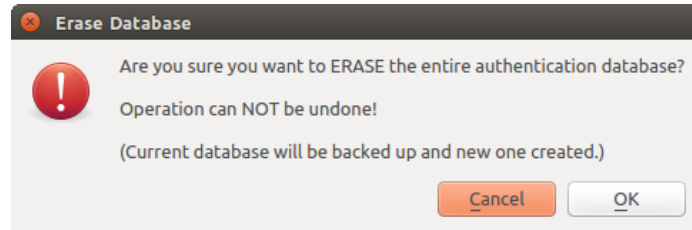


Abb. 25.16: Menü DB Löscherifizierung

### 25.1.6 Gebrauche Authentifikationskonfiguration

Typischerweise wird eine Authentifizierungskonfiguration in einem Konfigurationsdialog für ein Netzwerk-Dienste (wie WMS) ausgewählt. Allerdings kann das Auswahl-Widget überall, wo Authentifizierung benötigt wird, eingebettet werden oder in einer Nicht-Kernfunktionalität, wie in Dritt-Beteiligten PyQGIS oder C ++ Plugins.

Wenn die Auswahl *Keine Authentifizierung* im Pop-up-Menü anzeigt, wenn nichts ausgewählt ist, wenn es keine Konfigurationen zur Auswahl gibt oder wenn eine zuvor zugewiesene Konfiguration nicht mehr in der Datenbank gefunden wurde. Die *Typ* und *ID* Felder sind schreibgeschützt und stellen eine Beschreibung der Authentifizierungsmethode zur Verfügung und der entsprechenden ID der Config.

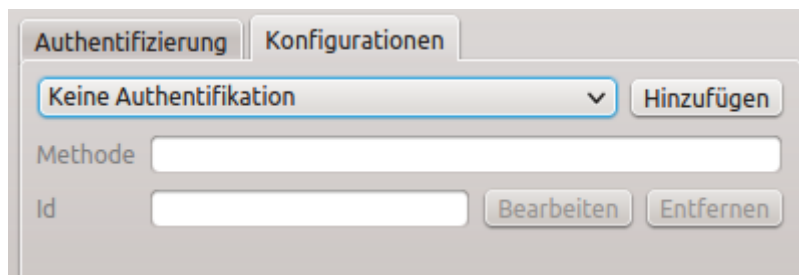


Abb. 25.17: Authentication configuration selector with no authentication

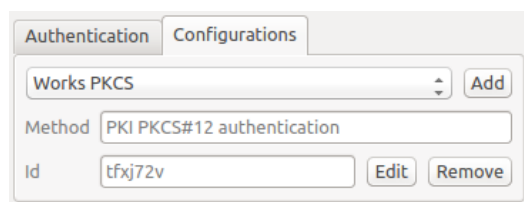


Abb. 25.18: Authentication configuration selector with selected config

### 25.1.7 Python-Bindung

Alle Klassen und öffentlichen Funktionen haben sip Bindungen, außer `QgsAuthCrypto`, da das Management des Master-Passwort-Hashing und die Auth-Datenbank-Verschlüsselung von der Hauptanwendung und nicht über Python behandelt werden sollte. Siehe `:ref:authentication_security_considerations`` über Python-Zugang.

## 25.2 Benutzerauthentifizierung Workflows

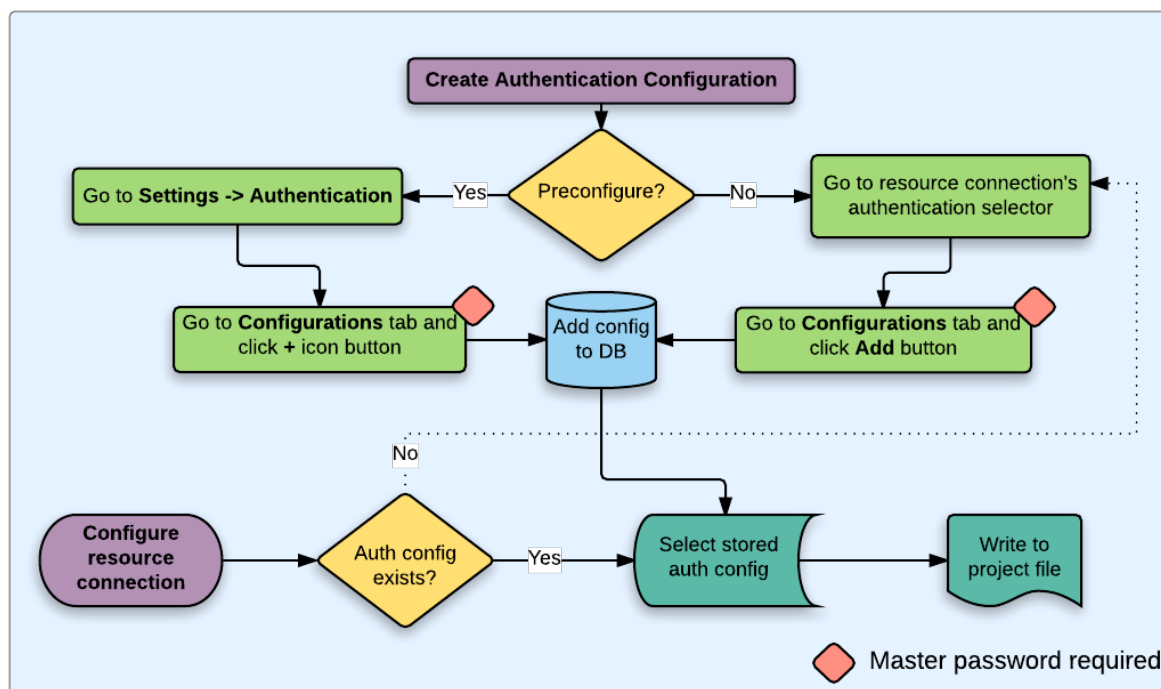


Abb. 25.19: Generischer Benutzer Workflow

### 25.2.1 HTTP(S) Authentifikation

Eine der häufigsten Verbindungen findet mittels HTTP (S) statt, z.B. bei WebGIS Servern und die Authentifizierungsmethoden werden oft für diese Art von Verbindungen genutzt. Die Methoden haben Zugriff auf das HTTP-Request Objekt und können sowohl den Request als auch seinen Header manipulieren. Dies ermöglicht viele Formen von Internet-basierten Authentifizierungen. Wenn über HTTP(S) mit der Standard Benutzername-/Passwort-Authentifizierungsmethode verwendet wird, wird bei der Verbindung eine HTTP-Basisauthentifizierung versucht.

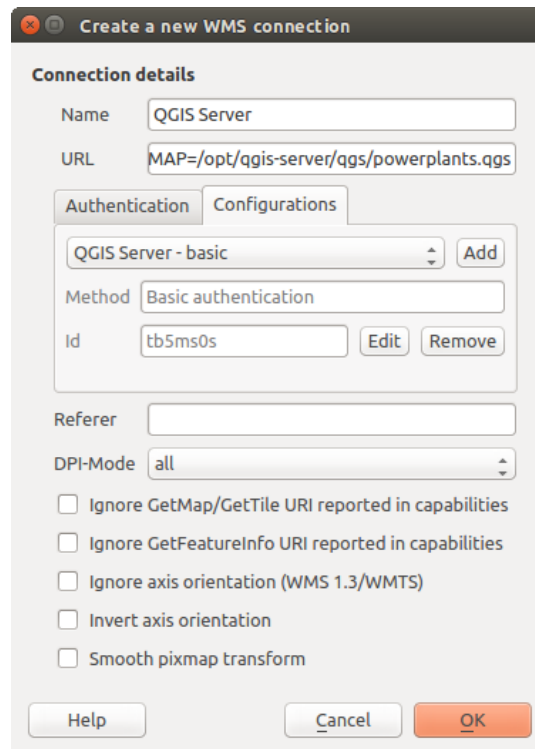


Abb. 25.20: Konfigurieren einer WMS-Verbindung für HTTP BASIC

### 25.2.2 Datenbank Authentifikation

Connections to database resources are generally stored as `key=value` pairs, which will expose usernames and (optionally) passwords, if *not* using an authentication configuration. When configuring with the auth system, the `key=value` will be an abstracted representation of the credentials, e.g. `authfg=81t21b9`.

**Create a New PostGIS connection**

**Connection Information**

Name: SSL PKI Docker Local

Service:

Host: 192.168.117.164

Port: 5432

Database: gis

SSL mode: require

**Authentication** | **Configurations**

Ident - joe Add

Method: Identity certificate authentication

Id: 81t21b9 Edit Remove

Test Connection

☐ Only show layers in the layer registries

☐ Don't resolve type of unrestricted columns (GEOMETRY)

☐ Only look in the 'public' schema

☐ Also list tables with no geometry

☐ Use estimated table metadata

Help Cancel OK

Abb. 25.21: Einstellen einer Postgress-SSL mit PKI-Verbindung

### 25.2.3 PKI Authentifikation

Wenn PKI-Komponenten innerhalb des Authentifikationssystems konfiguriert werden, haben Sie die Möglichkeit, Komponenten in die Datenbank zu importieren oder Dateien, die auf Ihrem System gespeichert sind, zu referenzieren. Letzteres kann nützlich sein, wenn solche Komponenten sich häufig verändern oder wenn die Dateien durch einen Systemadministrator verschoben werden. In beiden Fällen müssen Sie ein beliebiges Passwort speichern um Zugang zu privaten Schlüsseln innerhalb der Datenbank zu haben.

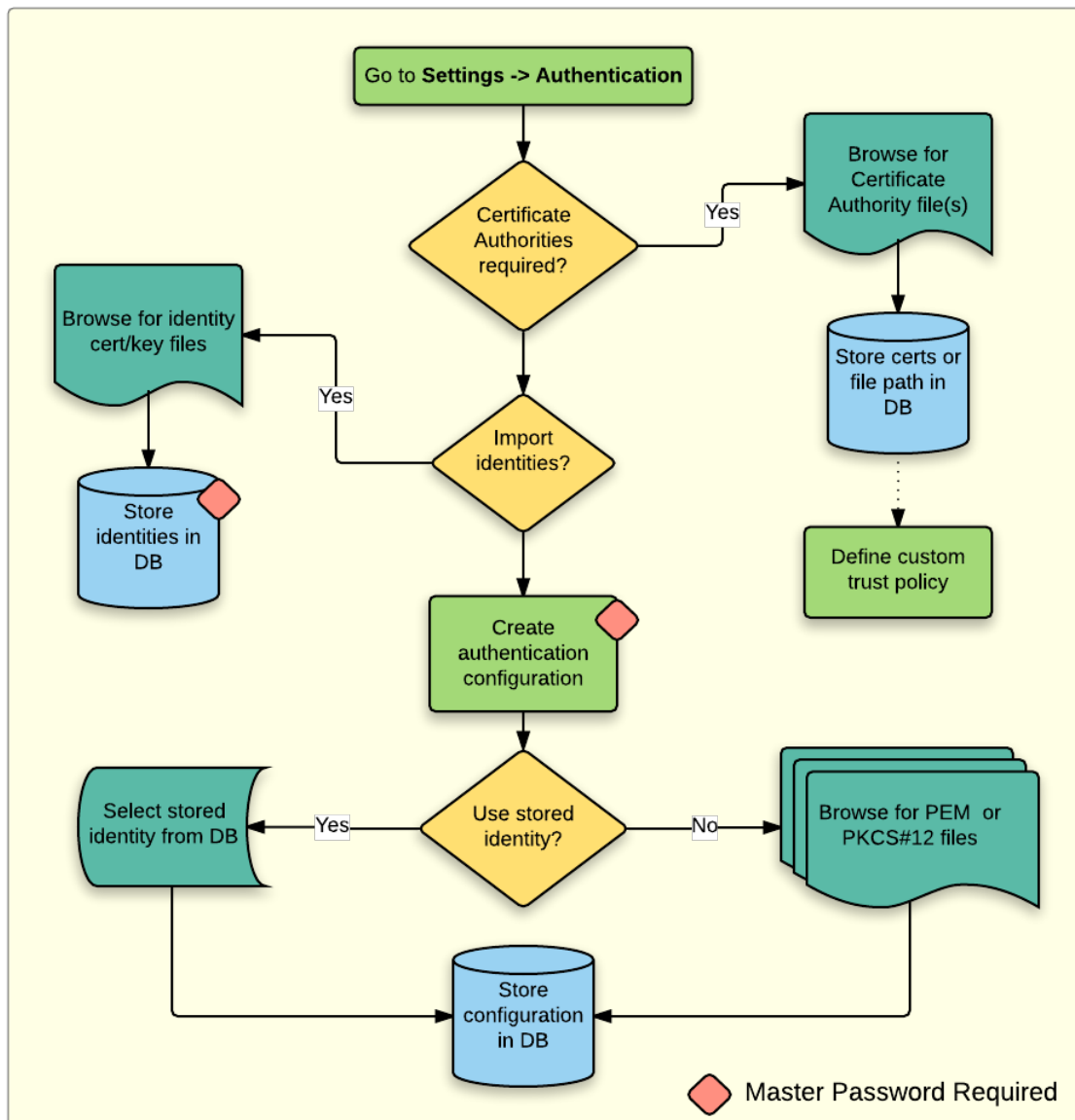


Abb. 25.22: PKI Konfiguration Workflow

All PKI components can be managed in separate editors within the **Certificate Manager**, which can be accessed in the *Authentication* tab in QGIS *Options* dialog (*Settings* ► *Options*) by clicking the *Manage Certificates* button.

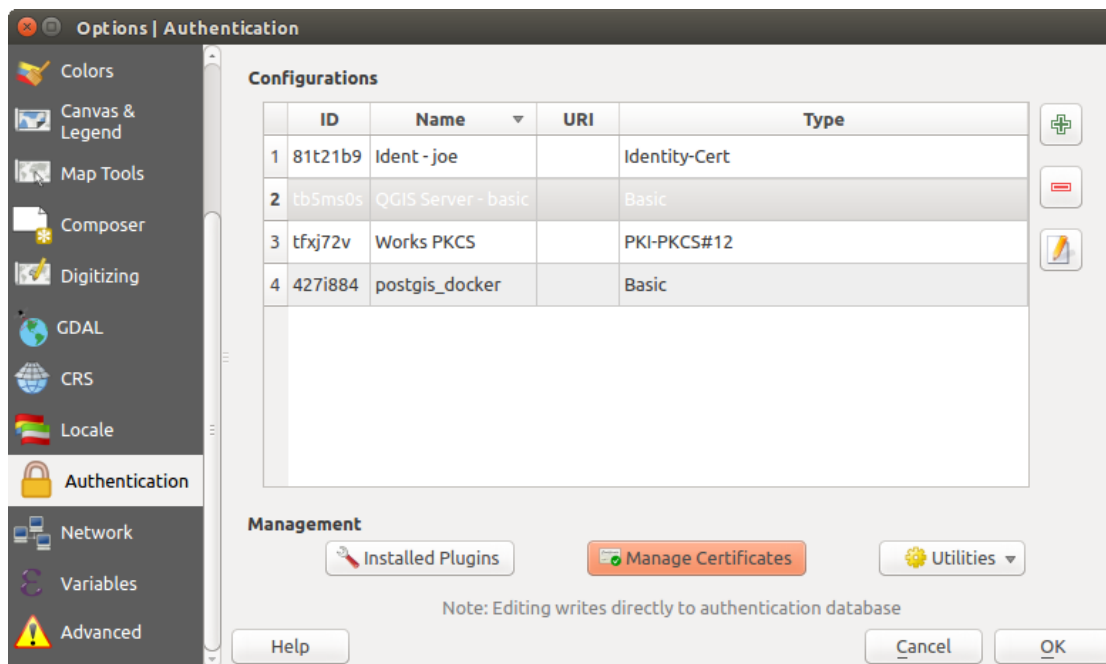


Abb. 25.23: Öffnen der Zertifikatsverwaltung

In the *Certificate Manager*, there are editors for **Identities**, **Servers** and **Authorities**. Each of these are contained in their own tabs, and are described below in the order they are encountered in the workflow chart above. The tab order is relative to frequently accessed editors once you are accustomed to the workflow.

**Bemerkung:** Because all authentication system edits write immediately to the authentication database, there is no need to click the *Options* dialog *OK* button for any changes to be saved. This is unlike other settings in the Options dialog.

## Autoritäten

Sie können verfügbare Zertifikat Autoritäten (Cas) aus dem **Autoritäten** Reiter im **Zertifikate verwalten** aus dem **Authentifikation** Reiter des QGIS **Optionen** Dialog verwalten.

Wie oben in dem Workflow-Diagramm Bezug genommen wird, ist der erste Schritt importieren oder einer Datei eine CAs zuweisen. Dieser Schritt ist optional und nicht notwendig, wenn Ihre PKI Vertrauensketten ihren Ursprung in einer CA hat, welche bereits auf Ihrem Betriebssystem (OS) installiert ist, wie ein Zertifikat eines kommerziellen Zertifizierungsanbieters. Falls Ihre Authentifizierung keine vertrauenswürdige CA ist, müssen Sie eine importieren oder dem Dateisystempfad eine zuweisen. (Kontaktieren Sie Ihren Systemadministrator, falls Sie sich unsicher sind.)

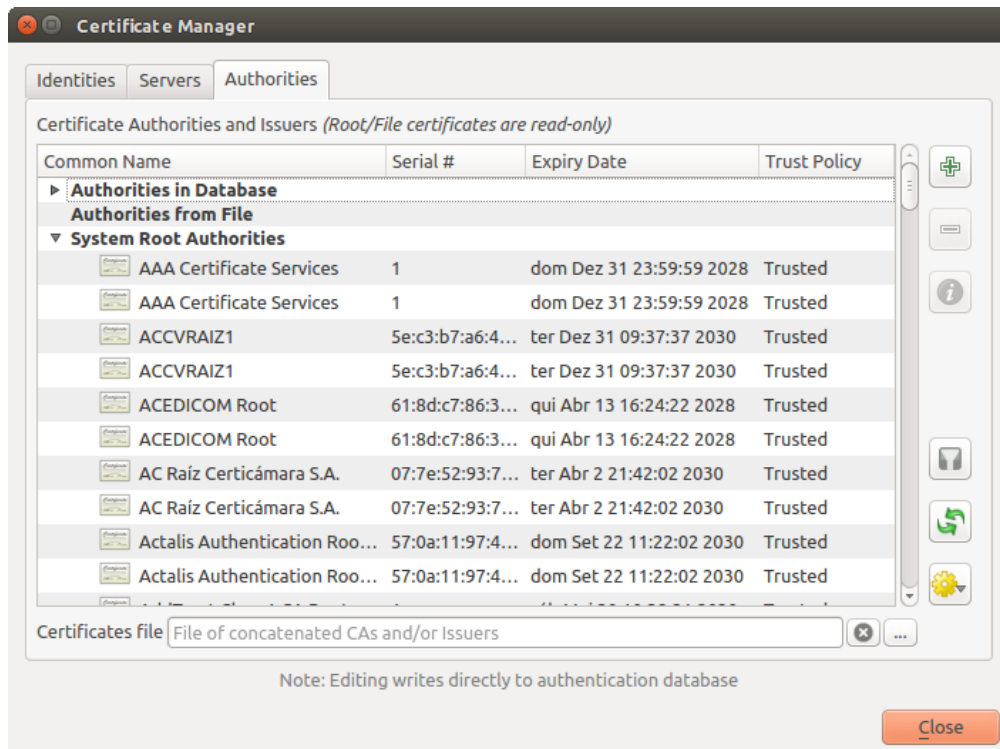



Abb. 25.24: Autoritäteneeditor

By default, the root CAs from your OS are available; however, their trust settings are not inherited. You should review the certificate trust policy settings, especially if your OS root CAs have had their policies adjusted. Any certificate that is expired will be set to untrusted and will not be used in secure server connections, unless you specifically override its trust policy. To see the QGIS-discoverable trust chain for any certificate, select it and click the  Show information for certificate.

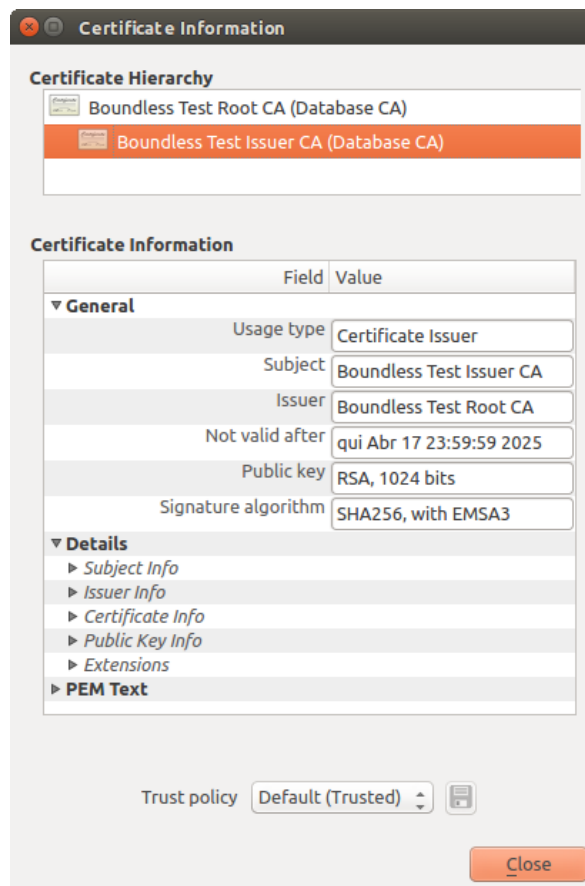




Abb. 25.25: Zertifikatsinformation Dialog

You can edit the *Trust policy*  for any selected certificate within the chain. Any change in trust policy to a selected certificate will not be saved to the database unless the  Save certificate trust policy change to database button is clicked per selected certification. Closing the dialog will **not** apply the policy changes.

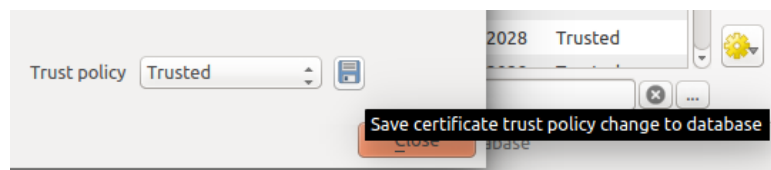



Abb. 25.26: Speichern der Vertrauensrichtlinien Änderungen

Sie können die gefilterten CAs überprüfen, sowohl die Zwischen- als auch die Stammzertifikate, denen durch Klicken auf sichere Verbindungen getraut wird oder ändern Sie die Standard-Vertrauensrichtlinie indem Sie den  **Optionen** Knopf klicken.

**Warnung:** Die Standard-Vertrauensrichtlinie ändern kann zu Problemen mit sicheren Verbindungen führen.



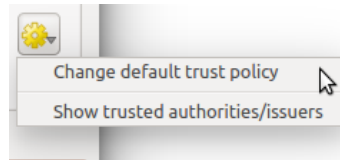


Abb. 25.27: Autoritätenoptionen-Menü

Sie können CAs importieren oder speichern ein Dateisystempfad aus einer Datei, welche mehrere CAs enthält oder importieren einzelne CAs. Das Standard PEM Format für Dateien, welche mehrere CA Kettenzertifikate enthalten, haben das Stammzertifikat am Ende der Datei und alle anschließend unterzeichneten Zertifikate oben, am Anfang der Datei.

Der CA Zertifikat-Importieren-Dialog wird alle CA Zertifikate innerhalb einer Datei finden, unabhängig von der Reihenfolge und gibt auch die Möglichkeit Zertifikate zu importieren, die als ungültig betrachtet werden (falls Sie deren Vertrauensrichtlinien überschreiben). Sie können die Vertrauensrichtlinie nach dem Import außer Kraft setzen, oder tun Sie es später in dem **Autoritäten** Editor.

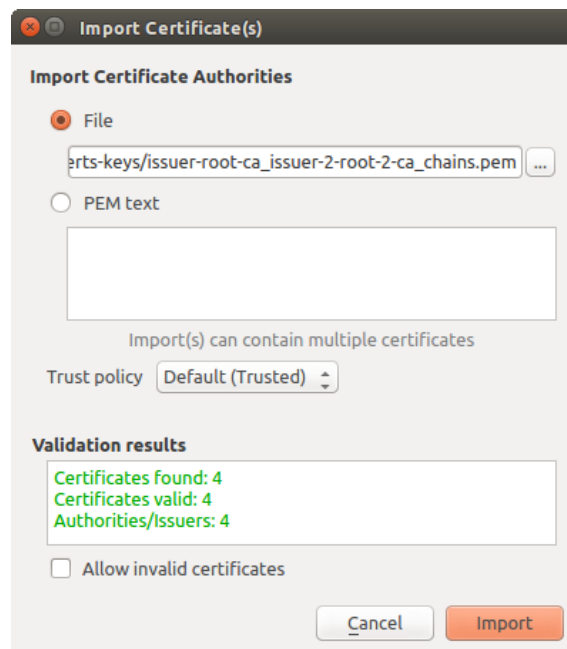


Abb. 25.28: Zertifikat importieren Dialog

**Bemerkung:** Wenn Sie Zertifikatinformationen in das *PEM Text* Feld einfügen, beachten Sie, dass verschlüsselte Zertifikate nicht unterstützt werden.

## Identitäten

Sie können verfügbare Client-Identitäts-Bündel über den Reiter *Identitäten* des *Zertifikat Manager* im Reiter **Authentifizierung** im Dialog **QGIS Optionen**. Eine Identität authentifiziert Sie gegen einen PKI-fähigen Dienst und besteht in der Regel aus einem Client-Zertifikat und einen privaten Schlüssel, entweder als separate Dateien oder kombiniert in einer einzigen „gebündelten“ Datei. Das Bündel oder private Schlüssel sind oft Passphrase geschützt.

Sobald Sie alle Zertifizierungsstellen (CAs) importiert haben können Sie optional alle Identitätsbündel in die Authentifizierungsdatenbank importieren. Wenn Sie die Identitäten nicht speichern möchten, können Sie ihre Komponentendateisystempfade innerhalb einer einzelnen Authentifizierungskonfiguration referenzieren.

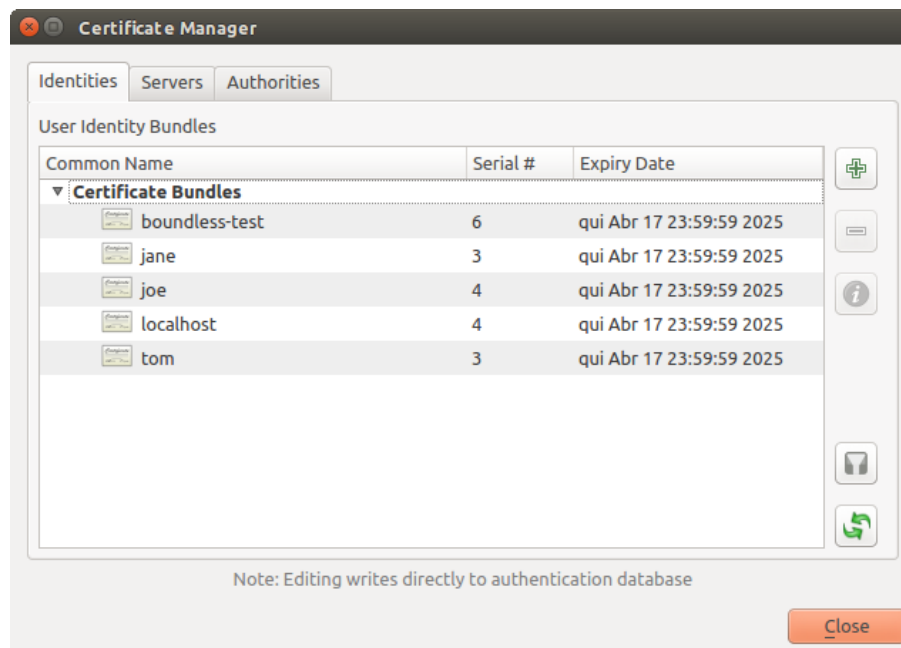


Abb. 25.29: Identitäteneditor

Wenn ein Identitäts-Bündel importieren, können diese Passwort-geschützt oder ungeschützt sein und CA-Zertifikate enthalten, die eine Vertrauenskette bilden. Vertrauenskette Zertifizierungen werden nicht an dieser Stelle importiert; sie können separat unter dem Reiter guilabel: *Authorities* hinzugefügt werden.

Beim Import wird das Zertifikats und der Private Key eines Bündels in der Datenbank gespeichert, wobei der Speicher des Schlüssels über das QGIS Master Passwort verschlüsselt ist. Die anschließende Verwendung des gespeicherten Bündels aus der Datenbank erfordert dann nur Eingabe des QGIS Master Passworts.

Persönliche Identitäts-Bündel bestehend aus PEM/DER (.pem / .der) und PKCS#12 (.p12/.pfx) Komponenten werden unterstützt. Wenn ein Schlüssel oder ein Bündel passwortgeschützt ist, wird das Passwort benötigt, um die Komponente zu validieren, bevor sie importiert wird. Wenn das Client-Zertifikat in dem Bündel ungültig ist (zum Beispiel ist das Datum des Inkrafttretens noch nicht gestartet oder abgelaufen), kann das Bündel nicht importiert werden.

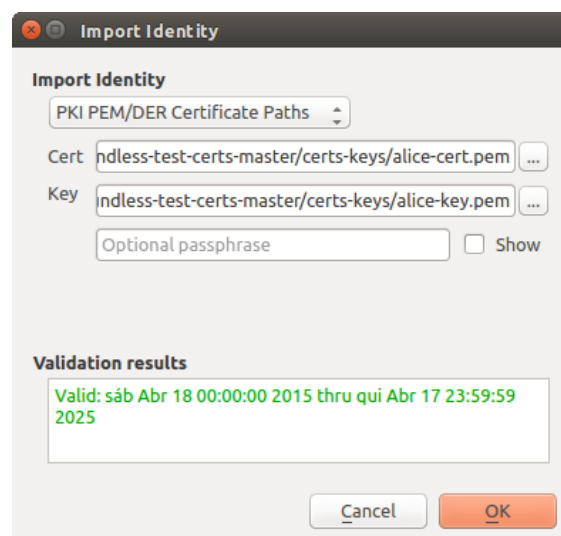


Abb. 25.30: PEM/DER Identität importieren

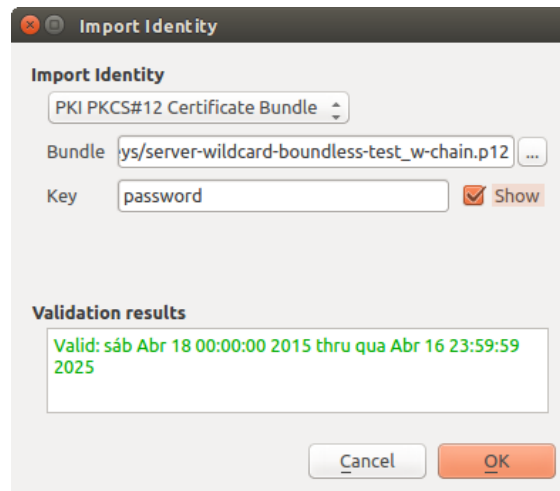


Abb. 25.31: PKCS#12 Identität importieren

## 25.2.4 Defekte Layer behandeln

Gelegentlich kommt es vor, dass die ID der Authentifizierungs-Konfiguration, die mit einer Projektdatei gespeichert wird, nicht mehr gültig ist, möglicherweise, weil die aktuelle Authentifizierungsdatenbank sich geändert hat, seitdem das Projekt zuletzt gespeichert wurde, oder aufgrund einer fehlenden Übereinstimmung der Anmeldeinformationen. In solchen Fällen öffnet sich der Dialog *Defekte Layer behandeln* beim Starten von QGIS.

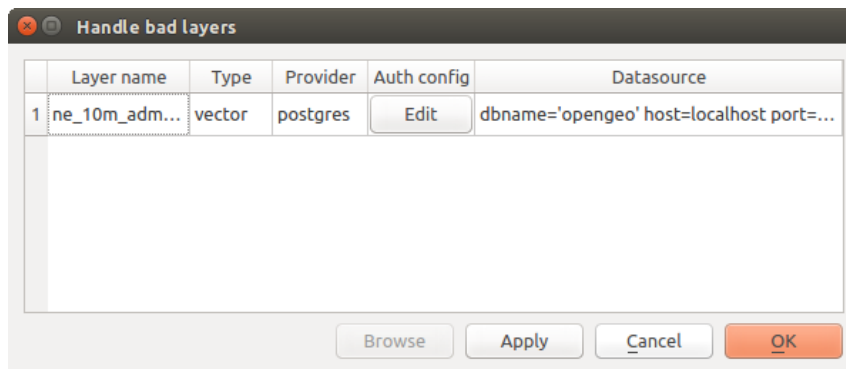


Abb. 25.32: Defekte Layer mit Authentifizierung behandeln

Wenn eine Datenquelle eine ihr zugewiesene Authentifizierungs Konfigurations-ID besitzt, können Sie diese bearbeiten. Dadurch wird der Pfad der Datenquelle automatisch geändert, so wie es beim Öffnen der Projektdatei mit einem Texteditor und dem bearbeiten von Zeichenfolgen passiert.

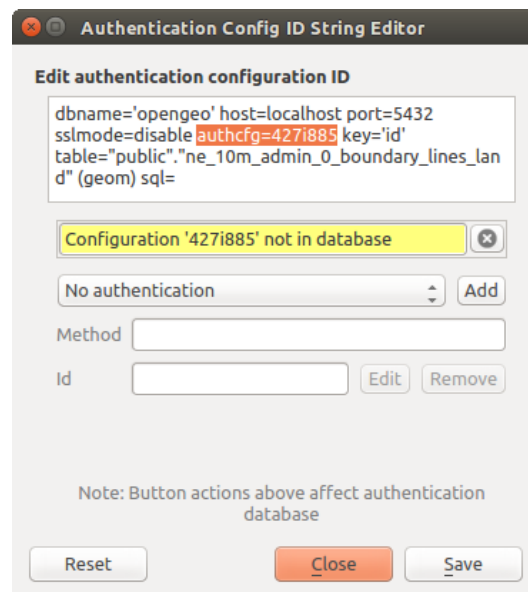


Abb. 25.33: Defekte Layer mit Authentifikationskonfiguration bearbeiten

## 25.2.5 Kennung der Authentifikationskonfiguration ändern

Occasionally, you will need to change the authentication configuration ID that is associated with accessing a resource. There are instances where this is useful:

- **Resource auth config ID is no longer valid:** This can occur when you have switched auth databases and need to *align* a new configuration to the ID already associated with a resource.
- **Shared project files:** If you intended to share projects between users, e.g. via a shared file server, you can *predefine* a 7-character (containing **a-z** and/or **0-9**) that is associated with the resource. Then, individual users change the ID of an authentication configuration that is specific to their credentials of the resource. When the project is opened, the ID is found in the authentication database, but the credentials are different per user.

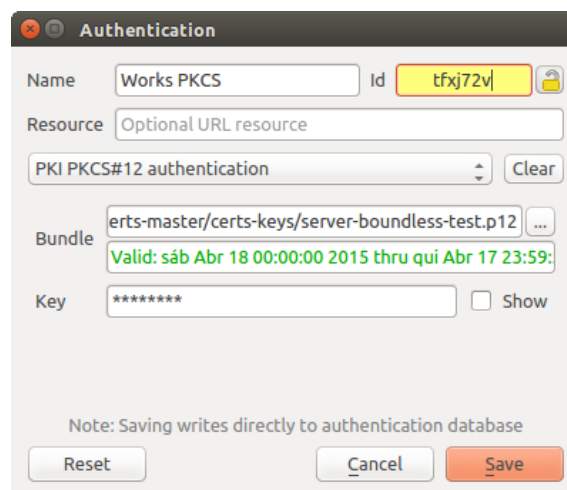


Abb. 25.34: Kennung einer Layer Authentifikationskonfiguration ändern (entsichert gelbes Textfeld)

**Warnung:** Das Ändern der Auth-Config-ID ist eine erweiterte Operation und sollte nur in Betracht gezogen werden, wenn es notwendig ist. Aus diesem Grund gibt es eine Lock-Taste, die geklickt werden muss, um das ID Text-Feld vor der Bearbeitung zu entsperren.

## 25.2.6 QGIS Server Unterstützung:

Wenn eine Projektdatei verwendet wird mit Layern, die eine Authentifizierung konfiguriert haben und vom QGIS Server verwendet werden sollen, gibt es ein paar zusätzliche Schritte zu beachten, um mit QGIS die Ressource zu laden:

- Authentifizierungsdatenbank muss verfügbar sein
- Das Authentifizierungsdatenbank Hauptpasswort muss verfügbar sein

When instantiating the authentication system, Server will create or use `qgis-auth.db` file in the active *user profile*, or the directory defined by the `QGIS_AUTH_DB_DIR_PATH` environment variable. It may be that the Server's user has no HOME directory, in which case, use the environment variable to define a directory that the Server's user has read/write permissions and is not located within the web-accessible directories.

Um das Master-Passwort an den Server zu übergeben, schreiben sie dieses in die erste Zeile einer Datei im Pfad des Dateisystems, der durch den Server Prozess Benutzer lesbar ist und definiert ist durch die `QGIS_AUTH_PASSWORD_FILE` Umgebungsvariable. Stellen Sie sicher, dass die Datei nur lesbar ist durch den Server Prozess Benutzer und nicht in einem über das Web zugänglichen Verzeichnis liegt.

**Bemerkung:** `QGIS_AUTH_PASSWORD_FILE` variable will be removed from the Server environment immediately after accessing.

## 25.2.7 SSL Serverausnahme

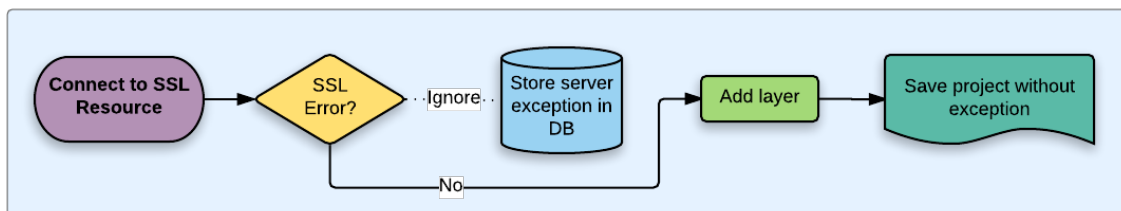



Abb. 25.35: SSL Serverausnahmen

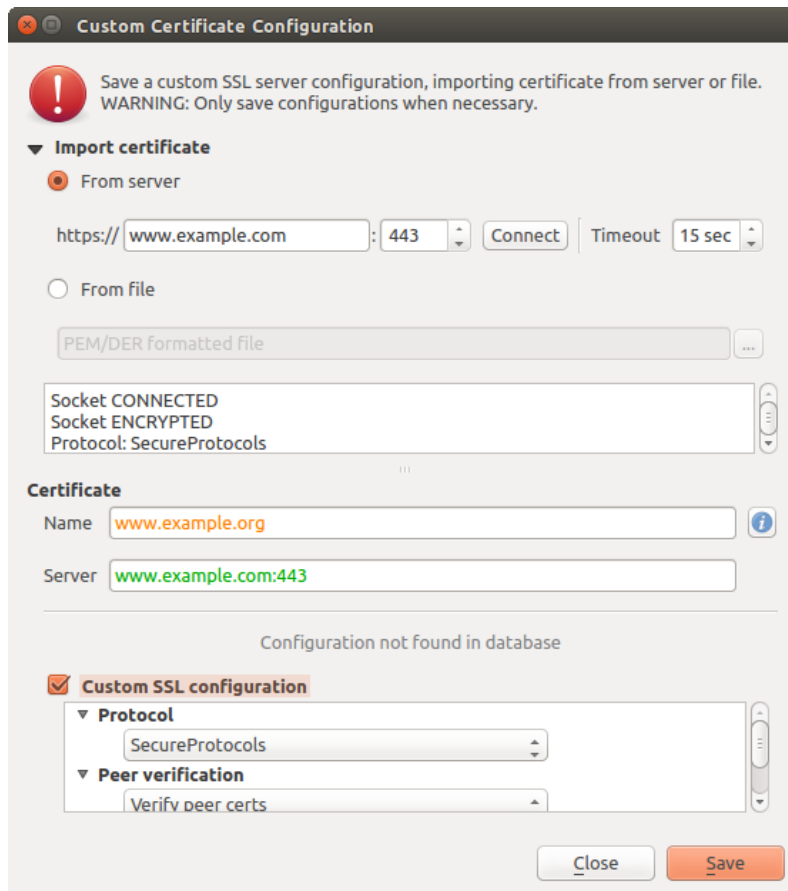
Sie können SSL-Server-Konfigurationen und Ausnahmen über den Reiter **Server** im Abschnitt **Authentifizierung** des QGIS Menüs **Optionen** verwalten.

Manchmal, wenn sie sich mit einem SSL-Server verbinden, gibt es Fehler beim SSL „Handshake“ oder mit dem Zertifikat des Servers. Sie können diese Fehler ignorieren oder eine SSL-Server-Konfiguration als eine Ausnahme erstellen. Dies ist ähnlich wie bei Web-Browsern, wo Sie SSL-Fehler ausser Kraft setzen können, aber mit einer besseren Kontrolle.

**Warnung:** Sie sollten keine SSL-Server-Konfiguration erstellen, wenn Sie keine umfassenden Kenntnisse des gesamten SSL-Setups zwischen dem Server und Client besitzen. Stattdessen wenden Sie sich besser an den Server-Administrator.

**Bemerkung:** Einige PKI-Setups verwenden eine völlig andere CA-Vertrauenskette, um Client-Identitäten zu validieren, als der Ablauf bei der Validierung eines SSL-Server-Zertifikats. Unter solchen Umständen behebt die Erstellung einer neuen Konfiguration für den Verbindungsserver nicht unbedingt das Problem, sodass sich in diesem Fall nur einen Serveradministrator wenden können.

You can pre-configure an SSL server configuration by clicking the  button. Alternatively, you can add a configuration when an SSL error occurs during a connection and you are presented with an **SSL Error** dialog (where the error can be ignored temporarily or saved to the database and ignored):



**Custom Certificate Configuration**

Save a custom SSL server configuration, importing certificate from server or file.  
WARNING: Only save configurations when necessary.

▼ **Import certificate**

☒ From server

https://  :   Timeout

☐ From file

...

Socket CONNECTED  
Socket ENCRYPTED  
Protocol: SecureProtocols

**Certificate**

Name  ⓘ

Server

Configuration not found in database

☒ **Custom SSL configuration**

▼ **Protocol**

▼ **Peer verification**

Abb. 25.36: Manuell eine Konfiguration hinzufügen

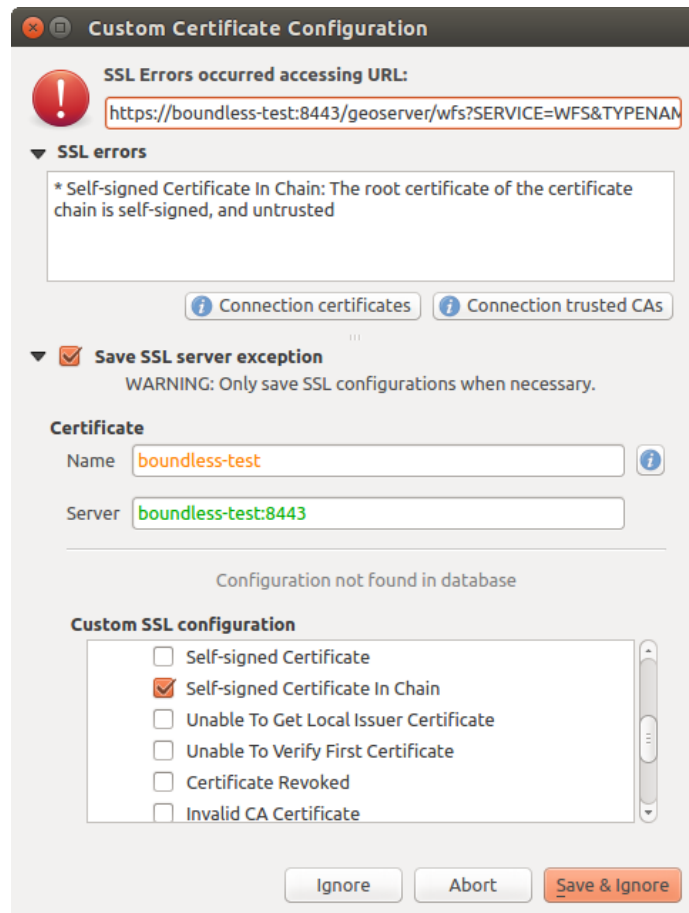


Abb. 25.37: Konfiguration während einem SSL Fehler hinzufügen

Sobald eine SSL-Konfiguration in der Datenbank gespeichert wird, kann sie bearbeitet oder gelöscht werden.

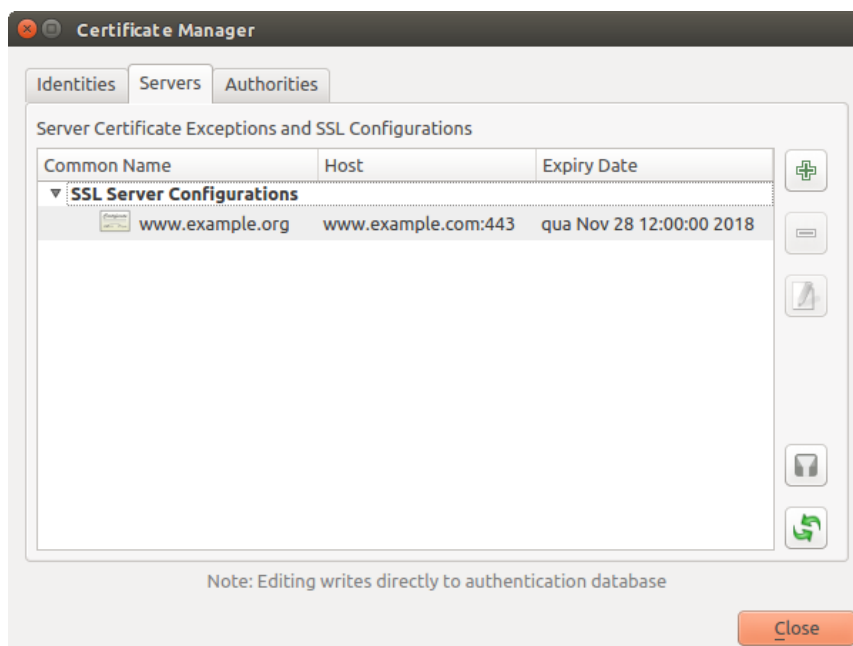


Abb. 25.38: Bestehende SSL Konfiguration

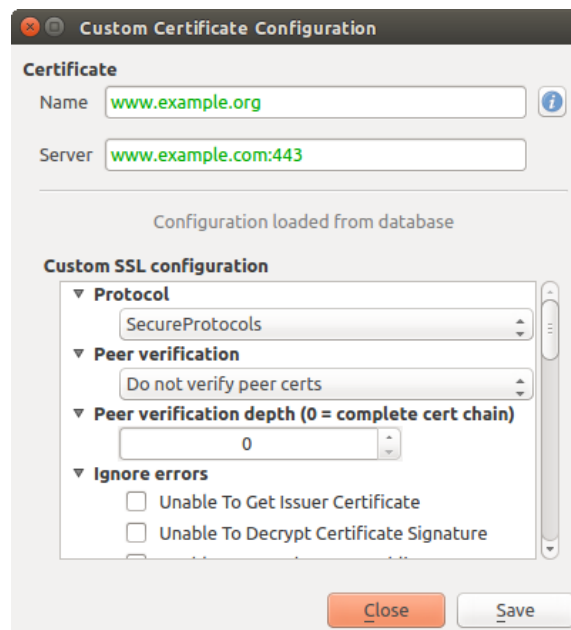


Abb. 25.39: Bestehende SSL Konfiguration bearbeiten

If you want to pre-configure an SSL configuration and the import dialog is not working for your server's connection, you can manually trigger a connection via the **Python Console** by running the following code (replace `https://bugreports.qt-project.org` with the URL of your server):

```
from qgis.PyQt.QtNetwork import QNetworkRequest
from qgis.PyQt.QtCore import QUrl
from qgis.core import QgsNetworkAccessManager

req = QNetworkRequest(QUrl('https://bugreports.qt-project.org'))
reply = QgsNetworkAccessManager.instance().get(req)
```

Dies wird ein SSL-Fehler-Dialog öffnen, wenn Fehler auftreten, wo Sie die Konfiguration in die Datenbank speichern können.

## 25.3 Sicherheitsüberprüfung

Sobald das Hauptpasswort eingegeben wird, ist die API offen für Authentifizierung configs die auf die Authentifizierungs-Datenbank zuzugreifen, so ähnlich wie funktioniert Firefox. Doch in der ersten Implementierung gibt es keinen Schutz gegen PyQGIS Zugang. Dies kann zu Problemen führen, wenn ein Benutzer ein böswilliges PyQGIS Plugin oder eine Standalone-Anwendung installiert, die den Zugriff auf die Anmeldeinformationen zur Authentifizierung erhält.

Die schnelle Lösung für die anfängliche Freisetzung von Funktion ist nicht nur die meisten PyQGIS Bindungen für das Authentifizierungssystem zu umfassen.

Eine weitere einfache, wenn auch nicht robuste Lösung ist eine Kombobox hinzuzufügen in *Einstellungen ► Optionen ► Authentifizierung* (Vorgabe ist „niemals“):

```
"Allow Python access to authentication system"
Choices: [ confirm once per session | always confirm | always allow | never]
```

Eine solche Einstellung der Optionen müsste an einem Ort nicht gespeichert werden, der für Python nicht zugänglich Python ist, z. B. die Authentifizierungsdatenbank und verschlüsselt mit dem Master-Passwort.



- Eine weitere Option kann es sein zu verfolgen, welche Plugins der Benutzer speziell hat
- erlaubt auf Authentifizierungssystem zuzugreifen, obwohl es schwierig sein kann, abzuleiten welches Plugin tatsächlich abrufen.
- Sandboxing Plugins, möglich in ihrer eigenen virtuellen Umgebung, würde „Cross-Plugin“ Hacking von Authentifizierung configs von einem anderen Plugin reduzieren, das autorisiert ist. Dies könnte bedeuten, die Cross-Plugin Kommunikation ist gut, aber vielleicht nur zwischen Plugins von Drittanbietern.
- Eine weitere gute Lösung ist, Code-Signing-Zertifikate zu prüfen. Denn dann kann das Zertifikat des Plugins beim Laden validieren. Bei Bedarf kann der Nutzer auch nicht vertrauenswürdige Politiken für Zertifikate einstellen, mit dem Plugin im Zusammenhang mit bestehenden Zertifikatsverwaltungen.
- Alternativ, Zugriff auf sensible Authentifizierungssystem Daten von Python
- konnte nie erlaubt werden und nur QGIS Kern-Widgets nutzen oder Authentifizierungs-System-Integrationen kopieren, würde es dem Plugin ermöglichen, mit Ressourcen zu arbeiten, die eine Authentifizierungskonfiguration haben, während Hauptpasswort und Authentifizierungs Config im Bereich der wichtigen Bereiche der Hauptapp laden.

Die gleichen Sicherheitsbedenken gelten für C ++ Plugins, obwohl es schwieriger sein wird, den Zugang zu beschränken, da es keine verbindliche Funktion gibt, einfach wie mit Python zu entfernen.

### 25.3.1 Einschränkungen

Die verwirrende [Lizenzierung und Exportierung](#) Probleme sind mit OpenSSL assoziiert. Um mit Qt mit SSL-Zertifikaten zu arbeiten, muss es Zugang zu den OpenSSL-Bibliotheken geben. Je nachdem, wie Qt kompiliert wurde, ist die Standard-dynamische Laufzeit auf OpenSSL-Libs verbunden (Exportbeschränkungen vermeiden).

QCA folgt einer ähnlichen Taktik, wobei durch die Verknüpfung zu QCA keine Einschränkungen entstehen, weil das QCA-openssl (OpenSSL) Plugin zur Laufzeit geladen wird. Das QCA-openssl Plugin ist mit den OpenSSL-Libs direkt verknüpft. Packagers wäre die benötigte OpenSSL-Linking Einschränkung, um sicherzustellen, dass erfüllt sind, wenn sie das Plugin versenden. Könnte sein. Ich weiß es nicht wirklich. Ich bin kein Anwalt.

Das Authentifizierungssystem deaktiviert sicher selbst, wenn `qca-openssl` zur Laufzeit nicht gefunden wird.



---

## GRASS GIS Integration

---

Die GRASS-Integration stellt den Zugang zu GRASS GIS Datenbanken und Funktionalitäten (siehe GRASS-PROJECT in Literatur und Internetreferenzen) bereit. Die Integration besteht aus zwei Teilen: Anbieter und Plugin. Der Anbieter erlaubt das Browsen, Managen und Visualisieren von GRASS Raster- und Vektorlayern. Das Plugin kann verwendet werden, um neue GRASS Locations und Mapsets zu erstellen, die GRASS Region zu ändern, Vektorlayer zu erstellen oder zu verändern und GRASS 2-D und 3-D Daten mit Hilfe von mehr als 400 GRASS Modulen zu analysieren. In diesem Abschnitt führen wir in die Funktionalitäten des Anbieters und des Plugins ein und geben einige Beispiele zur Handhabung und Arbeit mit GRASS-Daten.


Der Anbieter unterstützt GRASS Version 6 und 7, das Plugin unterstützt GRASS 6 und 7 (ab QGIS 2.12). QGIS Distribution kann entweder Provider/Plugin für GRASS 6 oder GRASS 7 oder für beide Versionen gleichzeitig enthalten (Binärdateien haben unterschiedliche Dateinamen). Nur eine Version des Anbieters/Plugin kann zur Laufzeit jedoch geladen werden.

### 26.1 Beispieldatensatz

Als Beispiel nutzen wir den QGIS Alaska Datensatz (siehe Abschnitt *Beispieldaten herunterladen*). Er beinhaltet eine kleine Muster GRASS LOCATION mit drei Vektorlayern und einer Rasterhöhenkarte. Legen Sie einen neuen Ordner `grassdata` an, laden den QGIS ‚Alaska‘ Datensatz `qgis_sample_data.zip` von <https://qgis.org/downloads/data/> herunter und entpacken die Datei nach `grassdata`.

More sample GRASS LOCATIONS are available at the GRASS website at <https://grass.osgeo.org/download/data/>.

### 26.2 GRASS Layer visualisieren

Wenn der Anbieter in QGIS geladen ist, wird für jeden Ordner der eine GRASS Location enthält das GRASS  Icon im QGIS Browser angezeigt. Gehen Sie im QGIS Browser zum Ordner `grassdata` und erweitern den Ordner `alaska` und das Mapset `demo`.

Sie können GRASS Raster- und Vektorebenen wie jeden anderen Layer aus dem Browser entweder durch Doppelklick auf Layer oder per Drag & Drop auf die Karte oder Legende laden.


---

**Tipp:** Probleme beim Laden von GRASS-Layern



Wenn Sie keine GRASS Locationitems sehen, überprüfen Sie *Hilfe* -> **Menüauswahl: Über Anbieter**, ob GRASS Vektor-Anbieter geladen sind.

## 26.3 Daten in eine GRASS LOCATION importieren via drag and drop

In diesem Abschnitt wird Ihnen Beispielhaft gezeigt, wie Sie Raster- oder Vektordateien in eine GRASS Mapset laden.

1. Navigieren Sie im QGIS-Browser zu dem Mapset, aus dem Sie Dateninformationen importieren wollen.
2. Um QGIS Browser einen Layer zu finden, den Sie GRASS importieren möchten finden, beachten Sie, dass Sie eine andere Instanz des Browsers öffnen können (*Browser Panel (2)*), wenn Quelldaten zu weit von der mapset im Baum entfernt sind.
3. Ziehen Sie einen Layer in die Zielkarte. Der Import nimmt unter Umständen für größere Layer etwas Zeit in Anspruch. Man sieht dann ein animiertes Icon  vor dem neuen Layer bis der Import abgeschlossen ist.

Wenn Rasterdaten in einem abweichenden KBS vorliegen, können sie mit Hilfe von *Approximate* (schnell) oder *Exact* (genau) transformiert werden. Wenn eine Verknüpfung zur Quelle des Rasters erstellt ist (mit Hilfe von `r.external`), haben die Quelldaten dasselbe KBS und das Format ist GDAL bekannt. Das KBS der Quelldaten wird verwendet. Man kann diese Optionen im *Browser* Reiter unter *GRASS Optionen* einstellen.

Wenn ein Quellraster mehr Kanäle hat, wird eine neue GRASS Karte für jede Schicht mit dem **<Kanalnummerr>** Suffix und eine Gruppe aller Karten mit  Symbol erzeugt. Externe Raster haben ein anderes Symbol .



## 26.4 GRASS Daten im QGIS Browser verwalten

- Karten kopieren: GRASS Karten können zwischen Mapsets innerhalb der gleichen Stelle mit Drag & Drop kopiert werden.
- Löschen von Karten: Rechtsklick auf eine GRASS Karte und wählen Sie **gui-Label: Löschen** aus dem Kontextmenü.
- Umbenennen von Karten: Rechtsklick auf eine GRASS Karte und wählen Sie: gui-Label: *Umbenennen* aus dem Kontextmenü.

## 26.5 GRASS Optionen



GRASS Optionen werden im *GRASS Optionen* Dialog eingestellt, welchen Sie auch durch Rechtsklick auf die Location oder das Mapset Element im Browser öffnen können, wenn Sie *GRASS Optionen* wählen.

## 26.6 GRASS Plugin starten

Um die GRASS Funktionalitäten in QGIS zu nutzen, muss das GRASS Plugin in den Erweiterungen ausgewählt und geladen werden. Man geht dazu im Menü zu *Erweiterungen* ►  *Erweiterungen verwalten und installieren...*, wählt dort  *GRASS* und klickt auf *OK*.

Die folgenden Haupteigenschaften werden beim Starten des GRASS Plugins im GRASS Menü bereitgestellt (*Erweiterungen* ► *GRASS*):

-  Mapset öffnen

-  Neues Mapset
-  SchlieÙe Mapset
-  GRASS-Werkzeugkiste öffnen
-  Aktuelle GRASS-Region darstellen
-  GRASS Optionen

26.7 GRASS Mapset öffnen

Ein GRASS Mapset muss geöffnet werden, um Zugang zu den GRASS Tools im Plugin zu bekommen (die Werkzeuge sind deaktiviert, wenn kein Mapset geöffnet ist). Sie können ein Mapset aus dem Browser öffnen: Rechtsklick auf das Mapset Element und wählen Sie dann *Mapset öffnen* aus dem Kontextmenü.

26.8 Information zur GRASS-Datenbank

GRASS data are stored in a directory referred to as GISDBASE. This directory, often called `grassdata`, must be created before you start working with the GRASS plugin in QGIS. Within this directory, the GRASS GIS data are organized by projects stored in subdirectories called `LOCATION`s. Each `LOCATION` is defined by its coordinate system, map projection and geographical boundaries. Each `LOCATION` can have several `MAPSET`s (subdirectories of the `LOCATION`) that are used to subdivide the project into different topics or sub-regions, or as workspaces for individual team members (see Neteler & Mitasova 2008 in *Literatur und Internetreferenzen*). In order to analyze vector and raster layers with GRASS modules, you generally have to import them into a GRASS `LOCATION`. (This is not strictly true – with the GRASS modules `r.external` and `v.external` you can create read-only links to external GDAL-supported datasets without importing them. This is not the usual way for beginners to work with GRASS, therefore this functionality will not be described here.)

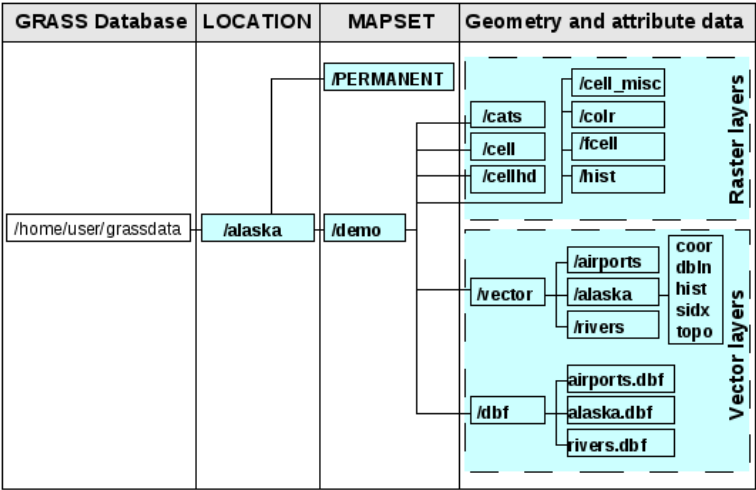




Abb. 26.1: GRASS Daten in der Alaska LOCATION

## 26.9 Daten in eine GRASS LOCATION importieren


Siehe Abschnitt `import_data` und um herauszufinden, wie die Daten einfach importiert per Drag & Drop in den Browser importiert werden können.




Dieser Abschnitt zeigt ein Beispiel wie man Raster- und Vektordaten in die ‚alaska‘ GRASS LOCATION aus dem QGIS ‚Alaska‘ Datensatz in einem üblichen Weg, mit Hilfe von Standard GRASS Modulen, importiert. Daher verwenden wir die Landcover Rasterkarte `landcover.img` und die GML Vektorkarte `lakes.gml` aus dem QGIS ‚Alaska‘ Datensatz (siehe [Beispieldaten herunterladen](#)).

1. Starten Sie QGIS und laden Sie das GRASS Plugin, falls dies noch nicht geschehen ist.
2. Klicken Sie in der GRASS Werkzeugleiste das  **Mapset öffnen** Icon um den *Wählen Sie ein GRASS Mapset* Assistenten zu starten.
3. Wählen Sie als GRASS Datenbank den Ordner `grassdata` im QGIS Alaska Datensatz, als LOCATION ‚alaska‘, als MAPSET ‚demo‘ und klicken auf *OK*.
4. Nun klicken Sie auf das Icon  **GRASS-Werkzeugkiste öffnen**, damit die GRASS Werkzeuge (siehe Abschnitt [Die GRASS Werkzeugkiste](#)) zur Verfügung stehen.
5. Um die Rasterkarte `landcover.img` zu importieren, drücken Sie auf das Modul `r.in.gdal` im *Modulbaum* Reiter. Diese GRASS Module ermöglicht es, GDAL-unterstützte Rasterdateien in eine GRASS LOCATION zu importieren.
6. Browsen Sie zum Ordner `raster` im QGIS ‚Alaska‘ Datensatz und wählen Sie die Datei `landcover.img`.
7. Geben Sie als Ausgabe-Rasterkarte `landcover_grass` ein und klicken auf *Starten*. Im Reiter *Ergebnis* sehen Sie das aktuell ausgeführte GRASS Kommando `r.in.gdal -o input=/path/to/landcover.img output=landcover_grass`.
8. Klicken Sie nach der Ausgabe **Erfolgreich beendet** auf *Ergebnis visualisieren*. Der Rasterlayer `landcover_grass` ist nun in GRASS importiert und wird im QGIS Arbeitsbereich angezeigt.
9. Um die GML Vektordatei `lakes.gml` zu importieren klicken Sie das Modul `v.in.ogr` im *Modulbaum* Reiter. Dieses GRASS Modul ermöglicht den Import von OGR-unterstützten Vektordateien in eine GRASS LOCATION. Der Moduldialog für `v.in.ogr` erscheint.
10. Browsen Sie zum Ordner `gml` im QGIS ‚Alaska‘ Datensatz und wählen Sie die Datei `:file:`lakes.gml` als OGR Datei aus.
11. Vergeben Sie `lakes_grass` als Ausgabe-Vektorkarte und klicken auf *Starten*. Sie brauchen sich um die anderen Optionen in diesem Beispiel nicht kümmern. Im Reiter *Ergebnis* sehen Sie das aktuell ausgeführte GRASS Kommando `v.in.ogr -o dsn=/path/to/lakes.gml output=lakes\_grass`.
12. Klicken Sie nach Erscheinen der Ausgabe **Erfolgreich beendet** auf *Ergebnis visualisieren*. Der Vektorlayer `lakes_grass` ist nun in GRASS importiert und wird im QGIS Arbeitsbereich dargestellt.

### 26.9.1 Eine neue GRASS LOCATION erstellen

Als ein Beispiel möchten wir Ihnen zeigen, wie die GRASS LOCATION des Alaska Beispieldatensatzes erstellt wurde. Das Koordinatenbezugssystem ist Albers Equal Area mit der Einheit ‚feet‘. Diese GRASS `:file:`LOCATION alaska`` wird für alle GRASS GIS Beispiele verwendet. Es ist also sinnvoll, sich diesen Datensatz zu installieren (siehe Abschnitt [Beispieldaten herunterladen](#)).

1. Starten Sie QGIS und laden Sie das GRASS Plugin, falls dies noch nicht geschehen ist.
2. Laden Sie die Shapefile `alaska.shp` aus dem QGIS Alaska Datensatz (see [Beispieldaten herunterladen](#)) in QGIS ein (see section [Loading a layer from a file](#)).
3. Klicken Sie in der GRASS Werkzeugleiste auf das  **Neues Mapset** Icon um den *Neues Mapset* Assistenten zu öffnen.

4. Wählen Sie einen vorhandenen GRASS Datenbankordner (GISDBASE) `grassdata` oder legen Sie für die neue LOCATION mit Hilfe des Dateimanagers einen neuen Ordner an. Klicken Sie dann auf *Weiter*.
5. We can use this wizard to create a new MAPSET within an existing LOCATION (see section [Eine neue GRASS MAPSET erstellen](#)) or to create a new LOCATION altogether. Select  *Create new location* (see [Abb. 26.2](#)).
6. Gebe einen Namen für die LOCATION an – wir verwendeten ‚alaska‘ – und klicke auf *Weiter*.
7. Definieren Sie die Projektion indem Sie den Radiobutton  *Projektion* zum Aktivieren der Projektionsliste klicken.
8. Wir benutzen Albers Equal Area Alaska (Fuß) Projektion. Da wir zufällig wissen dass es durch den EPSG ID 2964 repräsentiert wird geben wir dies in das Suchfenster ein. (Bemerkung: Wenn Sie diesen Ablauf für eine andere LOCATION und Projektion wiederholen wollen und Sie die EPSG ID nicht gespeichert haben, klicken Sie auf das  KBS-Status Icon in der unteren rechten Ecke der Statusleiste (siehe Abschnitt [Arbeiten mit Projektionen](#))).
9. Geben Sie bei *Filter* 2964 ein um die Projektion auszuwählen.
10. Klicken Sie auf *Weiter*.
11. Um die voreingestellte Region zu bestimmen, muss man die Grenzen der LOCATION im Norden, Süden, Osten und Westen angeben. Wir klicken einfach auf den Knopf *Setze aktuelle QGIS-Ausdehnung*, um die Ausdehnung des in QGIS geladenen Layers `alaska.shp` als voreingestellte GRASS-Region zu verwenden.
12. Klicken Sie auf *Weiter*.
13. Wir müssen außerdem ein MAPSET innerhalb unserer neuen LOCATION vorgeben (das ist beim Erstellen einer neuen LOCATION erforderlich). Sie können hier einen beliebigen Namen vergeben, wir verwendeten ‚demo‘. GRASS erstellt automatisch ein spezielles MAPSET mit Namen PERMANENT, das die Hauptdaten des Projekts, die vorgegebene Ausdehnung und die Vorgaben zum Koordinatensystem speichert (siehe auch Neteler & Mitasova 2008 in [Literatur und Internetreferenzen](#)).
14. Sehen Sie sich die Zusammenfassung an und vergewissern sich, dass alles korrekt ist und klicken anschließend auf *Abschließen*.
15. Die neue LOCATION, ‚alaska‘, und zwei MAPSETs, ‚demo‘ und ‚PERMANENT‘ werden erstellt. Die gerade geöffnete Arbeitsumgebung ist ‚demo‘, wie Sie es definiert haben.
16. Beachten Sie, dass einige Werkzeuge des GRASS Plugins grau hinterlegt waren und nun auch zur Verfügung stehen.

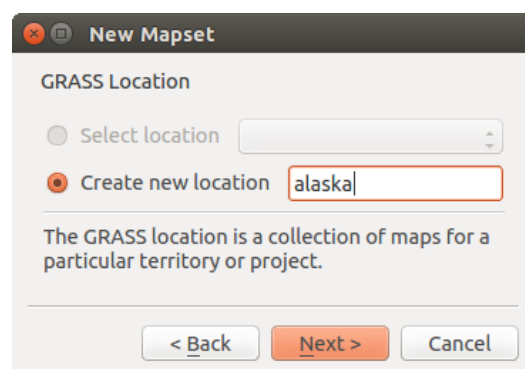




Abb. 26.2: Erstellen einer neuen GRASS LOCATION oder einer neuen MAPSET in QGIS

Wenn das wie eine Menge von Schritten schien ist es doch nicht so schlimm und ein schneller Weg eine LOCATION zu erstellen. Die LOCATION ‚alaska‘ ist jetzt bereit für den Datenimport (siehe Abschnitt [Daten in eine GRASS LOCATION importieren](#)). Sie können auch die bereits bestehenden Vektor- und Rasterdaten in der Beispiel GRASS LOCATION ‚alaska‘ verwenden, die im **lqgl** ‚Alaska‘ Datensatz [Beispieldaten herunterladen](#) enthalten ist und weiter zu Abschnitt [Das GRASS Vektormodell](#) gehen.

## 26.9.2 Eine neue GRASS MAPSET erstellen

Ein Benutzer hat nur Schreibzugriff auf eine MAPSET die er oder sie erstellt hat. Das heißt dass Sie neben dem Zugriff zu Ihrer eigenen MAPSET Sie Karten in MAPSETs anderer Benutzer lesen können (und Sie können Ihre lesen) aber Sie nur die Karten in Ihrer eigenen MAPSET verändern und löschen können.

Alle MAPSETs beinhalten eine WIND Datei die die aktuellen Grenzkordinatenwerte und die aktuell ausgewählte Rasterauflösung speichert (siehe Neteler & Mitasova 2008 in *Literatur und Internetreferenzen*, und Abschnitt *Einstellung der GRASS Region*).

1. Starten Sie QGIS und laden Sie das GRASS Plugin, falls dies noch nicht geschehen ist.
2. Klicken Sie in der GRASS Werkzeugleiste auf das  Icon um den *Neues Mapset* Assistenten zu öffnen.
3. Wählen Sie den GRASS Datenbankordner (GISDBASE) `grassdata` mit der LOCATION `,alaska'` aus wo Sie eine weitere MAPSET genannt `,test'` hinzufügen wollen.
4. Klicken Sie auf *Weiter*.
5. We can use this wizard to create a new MAPSET within an existing LOCATION or to create a new LOCATION altogether. Click on the radio button  *Select location* (see Abb. 26.2) and click *Next*.
6. Geben Sie den Namen `test` ein für ein neues MAPSET. Unten im Wizard sehen Sie eine Liste aller existierenden MAPSETs und ihrer Eigentümer.
7. Klicken Sie auf *Weiter*, vergewissern Sie sich, dass alle Angaben richtig sind und klicken auf *Abschließen*.

## 26.10 Das GRASS Vektormodell

Es ist wichtig, das `:index:` GRASS Vektordaten Modell ,vor der Digitalisierung zu verstehen. Im Allgemeinen verwendet GRASS ein topologisches Vektormodell. Das bedeutet, dass Flächen nicht als geschlossene Polygone vorhanden sind, sondern als ein oder mehrere Umrandungen (Boundaries). Eine Umrandung (Boundary) zwischen zwei aneinander grenzenden Flächen ist nur einmal digitalisiert worden; beide Flächen teilen sich diese Umrandung. Umrandungen dürfen keine Lücken haben. Eine Fläche besteht also aus einer Umrandung und einem Zentroid, der diese Fläche als ein sog. **Labelpunkt** mit einer Attributtabelle verknüpft.

Neben den Umrandungen und Zentroiden kann eine Vektorkarte selbstverständlich auch Punkte und Linien enthalten. Alle diese Geometrieelemente können innerhalb ein und dem selben Datensatz enthalten sein. Sie werden in unterschiedlichen ,Ebenen' innerhalb von QGIS dargestellt. Auch wenn es möglich ist, Geometrieelemente zu mischen, so ist es eigentlich unüblich und wird normalerweise auch in GRASS GIS nur selten verwendet. Etwa bei Netzwerkanalysen. Im Normalfall sollten Sie versuchen, unterschiedliche Geometrietypen in unterschiedlichen Datensätzen (Layern) zu speichern.

Es ist auch möglich, unterschiedliche Inhalte des gleichen Geometrietyps in verschiedenen Ebenen eines Vektorlayers zu speichern. Beispielsweise können Felder, Wälder und Seen in einem Vektordatensatz gespeichert werden. Angrenzende Seen, Felder und Wälder teilen sich dann die gleiche Umrandung, jedoch haben sie separate Attributtabellen, die über ihre Ebene angesprochen wird. Darüber hinaus können Sie auch Attribute für die Umrandungen vergeben, falls eine Umrandung gleichzeitig einen Weg darstellt. In diesem Fall könnte auch die Umrandung eine separate Attributtabelle haben.

Die ,Ebene' eines jeden Objektes wird in GRASS intern als ,layer' bezeichnet. ,Layer' ist die Nummer die definiert ob es mehr als einen Layer innerhalb des Datensatzes gibt (z.B. ob die Geometrie Wald oder See ist). Vorerst kann dies nur eine Nummer sein. In Zukunft wird GRASS auch Namen als Felder in der Benutzeroberfläche unterstützen.

Attribute können innerhalb der GRASS LOCATION als dBase oder SQLite3 oder in externen Datenbanktabellen abgelegt werden, z.B. PostgreSQL, MySQL, SQLITE3, etc.

Die Attribute in den Tabellen werden über ein sog. ,Kategoriefeld' an die Geometrien des Datensatzes gehängt.

Die ,Kategorie' (oder key, ID, etc) ist eine Ganzzahl, über die eine Verknüpfung zwischen den Geometrien und den Spalten in der Datenbanktabelle hergestellt wird.



**Tipp: Das GRASS Vektormodell verstehen**

The best way to learn the GRASS vector model and its capabilities is to download one of the many GRASS tutorials where the vector model is described more deeply. See <https://grass.osgeo.org/learn/manuals/> for more information, books and tutorials in several languages.

## 26.11 Einen neuen GRASS Vektorlayer erstellen

Um einen neuen GRASS Vektorlayer zu erzeugen, wählen Sie eines der folgenden Elemente aus dem Mapset Kontextmenü im Browser:

- Neuer Punktlayer
- Neuer Linienlayer
- Neuer Polygonlayer

und einen Namen in das Dialogfeld eingeben. Eine neue Vektorkarte wird erstellt und der Layer dem Kartenfenster hinzugefügt und die Bearbeitung gestartet. Einen Typ des Layers zu wählen beschränkt nicht den Geometrietypen, der in der Vektorkarte digitalisiert werden kann. In GRASS ist es möglich, alle Arten von Geometrietypen (Punkt, Linie und Polygon) in einer Vektorkarte zu organisieren. Der Typ wird nur verwendet, um die Layer dem Kartenfenster hinzuzufügen, da QGIS von dem Layer fordert eine bestimmte Art zu haben.

Es ist auch möglich, Layer bestehenden Vektorkarten hinzuzufügen, wählen Sie ein Element oben aus dem beschriebenen Kontextmenü der bestehenden Vektorkarte.

GRASS GIS erlaubt es aufgrund des topologischen Datenmodells, die verschiedenen Geometrietypen (Punkt, Linie und Fläche) in einem Vektorlayer abzuspeichern. Aus diesem Grund ist es nicht notwendig im Vorfeld einen Geometrietyp festzulegen. Dies unterscheidet sich von der Erstellung eines neuen Shapefile in QGIS, denn Shapefiles verwenden das Simple Feature Vektormodell (siehe Abschnitt *Neue Vektorlayer erstellen*).

## 26.12 Digitalisieren und Editieren eines GRASS Vektorlayers

GRASS Vektorebenen können mit den Standard QGIS Digitalisierungsfunktionen werden digitalisiert. Es gibt jedoch einige Besonderheiten, die Sie kennen sollten, aufgrund

- GRASS topologischen Modell im Vergleich zu einfachen QGIS Funktion
- Komplexität von GRASS-Modellen
  - mehrere Layer in einer Karte
  - mehrere Geometrietypen in einer Karte
  - Geometrieteilung von mehreren Objekte durch mehrere Layer

Die Besonderheiten sind in den folgenden Abschnitten besprochen.

### Speichern, Änderungen verwerfen, rückgängig machen, wiederholen

**Warnung:** Alle während der Bearbeitung vorgenommenen Änderungen werden sofort auf Vektorkarten und die dazugehörigen Attributtabelle geschrieben.

Die Änderungen werden nach jeder Operation geschrieben, es ist jedoch möglich, Rückgängig/Wiederholen zu wählen oder alle Änderungen zu verwerfen, wenn die Bearbeitung geschlossen wird. Wenn Änderungen rückgängig gemacht werden oder Änderungen verworfen werden, wird der ursprüngliche Zustand in der Vektorkarte neu geschrieben und die der Attributtabelle.

Es gibt zwei Hauptgründe für dieses Verhalten:

- Es ist die Natur von GRASS-Vektoren aus Überzeugung zu kommen, dass der Benutzer tun will, was er tut und es ist besser, dass die Daten gespeichert werden, wenn die Arbeit plötzlich unterbrochen wird (z. B. Stromausfall)
- Die Notwendigkeit für eine effektive Bearbeitung von topologischen Daten visualisiert Informationen über die topologische Korrektheit, solche Informationen können nur von GRASS Vektorkarte erworben werden, wenn Änderungen an der Karte vorgenommen werden.

### Werkzeuggeste

Die ‚Digitalisierungsleiste‘ hat einige spezielle Werkzeuge, wenn Sie einen GRASS Layer bearbeiten:

Icon	Werkzeug	Zweck
	Neuer Punkt	Digitalisiert neuen Punkt
	Neue Linie	Digitalisiert neue Linie
	Neue Grenze	Digitalisiert neue Grenze
	Neuer Zentroid	Digitalisiert neuen Zentroiden (Labelpunkt für eine existierende Fläche)
	Neue geschlossene Grenze	Digitalisiert neue geschlossene Grenze

Tabelle Digitalisierung GRASS: GRASS Digitalisierungstools

### Tipp: Polygone in GRASS digitalisieren

Wenn Sie ein Polygon innerhalb eines GRASS Vektorlayers erstellen wollen, digitalisieren Sie zuerst die Boundary der Fläche. Danach fügen Sie einen Zentroid (Labelpunkt) hinzu. Der Grund ist, dass in topologischen Layern die Attributinformationen einer Fläche immer mit dem Zentroiden und nicht mit der Grenze verknüpft werden.

### Kategorie

Kategorie, die oft als cat bezeichnet wird, ist eine Art von ID. Der Name stammt aus Zeiten, in denen GRASS-Vektoren nur einzelne Attribut „Kategorien“ hatten. Kategorien werden als Bindeglied zwischen Geometrie und Attribute verwendet. Eine einzelne Geometrie kann mehrere Kategorien haben und somit mehrere Funktionen in verschiedenen Layern darstellen. Derzeit ist es möglich, nur eine Kategorie pro Layer mit QGIS Werkzeugen zu bearbeiten. Neue Funktionen haben automatische neue einzigartige Kategorien, ohne Grenzen. Grenzen bilden in der Regel nur Bereiche und stellen keine lineare Funktionen dar, es ist jedoch möglich, Attribute für eine Grenze später zu definieren, beispielsweise in einer anderen Ebene.

Neue Kategorien werden immer nur in derzeit bearbeiteten Layern erstellt.

Es ist nicht möglich mehrere Kategorien zuweisen, um die Geometrie mit QGIS zu bearbeiten, solchen Daten werden wie mehrere Funktionen dargestellt und einzelne Merkmale, auch aus verschiedenen Schichten, können gelöscht werden.

### Attribute

Die Attribute des aktuell bearbeiteten Layer können nur geändert werden, wenn die Vektorkarte mehrere Layer enthält, Merkmale anderer Layer werden alle ‚<nicht editierbar (layer#)>‘ eingestellt, so werden Sie gewarnt, dass solche Attribut nicht editierbar sind. Der Grund dafür ist, dass andere Layer in der Regel andere Gruppe von Feldern haben und haben können, während QGIS nur einen festen Satz von Feldern pro Layer unterstützt.

Wenn einer Geometrie primitive nicht eine Kategorie zugeordnet ist, wird eine neue einzigartige Kategorie automatisch zugeordnet und eine neue Eintragung in der Attributtabelle erstellt, wenn ein Attribut dieser Geometrie verändert wird.

**Tipp:** Wenn Sie Bulk-Aktualisierung von Attributen in der Tabelle machen wollen, zum Beispiel mit dem ‚Feldrechner‘ (*Using the Field Calculator*), und es einige Funktionen ohne Kategorie gibt, die Sie nicht aktualisieren

wollen (in der Regel Grenzen) , können Sie diese in dem „Erweiterten Filter“ auf `cat is not null` einstellen.

### Stil editieren

Die topologische Symbologie ist für eine effektive Bearbeitung von topologischen Daten wesentlich. Bei Beginn der Bearbeitung, eine spezialisierter Renderer „GRASS Bearbeiten“ wird automatisch auf die Ebene gesetzt und der ursprüngliche Renderer wiederhergestellt, wenn die Bearbeitung geschlossen ist. Der Stil kann in den Layereigenschaften im „Stil“ Reiter angepasst werden. Der Stil kann auch in der Projektdatei oder in einer separaten Datei gespeichert werden. Wenn Sie den Stil anpassen, ändern sich nicht seinen Namen, weil er zurückgesetzt wird, wenn das Bearbeiten erneut gestartet wird.

**Tipp:** Speichern Sie die Projektdatei nicht, wenn der Layer bearbeitet wird, dies würde den Layer mit „Edit Style“ speichern, was keine Bedeutung hat, wenn der Layer nicht bearbeitet wird.

Der Stil basiert auf topologischen Informationen, die vorübergehend als Feld „topo\_symbol“ der Tabelle hinzugefügt wird. Das Feld wird automatisch entfernt, wenn die Bearbeitung abgeschlossen ist.

**Tipp:** Entfernen Sie nicht das „topo\_symbol“ Feld aus der Attributtabelle, das würde das Objekt unsichtbar machen, weil die Darstellung basiert auf dieser Spalte.

### Objektfang

Um einen Bereich zu bilden, müssen Ecken verbundener Grenzen **genau** die gleichen Koordinaten haben. Dies kann mit dem Schnappen Werkzeug erreicht werden, wenn Kartenfenster und Vektorkarte das gleiche KBS haben. Ansonsten, koordiniert durch Umwandlung von Karte auf Kartenfenster und wieder zurück, können die Koordinaten etwas unterschiedlich wegen Darstellungsfehler und KBS Transformationen werden.

**Tipp:** Layer KBS verwenden, auch für das Fenster während dem editieren.

### Einschränkungen

Die gleichzeitige Bearbeitung von mehreren Layern innerhalb des gleichen Vektors wird nicht unterstützt. Dies ist vor allem deswegen unmöglich, durch das nicht händeln mehrere Handhabungs Stacks für eine einzelne Datenquelle.



Unter Linux und macOS kann jeweils nur ein GRASS layer bearbeitet werden. Das liegt an einem Fehler in GRASS der es nicht erlaubt, Datenbanktreiber in zufälliger Reihenfolge zu schließen. Der Fehler wird von den GRASS Entwicklern behoben.

### Tipp: GRASS Schreibberechtigung

Sie müssen der Besitzer der GRASS MAPSET, die Sie bearbeiten wollen, sein. Es ist unmöglich Datenlayer in einer MAPSET die Ihnen nicht gehört zu bearbeiten, auch wenn Sie Schreibrechte darauf haben.

## 26.13 Einstellung der GRASS Region


Die Region Einstellung (Einstellung eines räumlichen Arbeitsfensters) in GRASS ist wichtig für das Arbeiten mit Rasterlayern. Vektoranalysen sind standardmäßig nicht auf eine definierte Region Definition begrenzt. Aber alle neu erstellten Raster werden die räumliche Ausdehnung und Auflösung der gerade definierten GRASS Region haben, unabhängig von Ihrer ursprünglichen Ausdehnung und Auflösung. Die aktuelle GRASS Region ist in der `$LOCATION/$MAPSET/WIND` Datei gespeichert und Sie definiert die Nord-, Süd-, Ost- und Westgrenze, die Anzahl von Spalten und Reihen sowie die horizontale und vertikale räumliche Auflösung.

Es ist auch möglich, die Region mit dem Knopf  Aktuelle GRASS-Region darstellen aus- bzw. einzuschalten.

Die Region kann im Reiter ‚Region‘ im ‚GRASS-Werkzeuge‘ Widget angepasst werden. Man kann die neuen Grenzen und die neue Auflösung hier eingeben und auf *Anwenden* drücken. Durch Drücken auf den Knopf *Ausdehnung durch Ziehen auf der Karte wählen* kann die Region interaktiv durch Ziehen eines Rechteckes mit der Maus im QGIS Arbeitsbereich festgelegt werden.

Das GRASS Modul `g.region` bietet viele weitere Optionen zur Einstellung der passenden Ausdehnung und Auflösung der Region für die Rasteranalyse. Das Modul `g.region` können Sie über die GRASS Werkzeugkiste nutzen, wie in Kapitel *Die GRASS Werkzeugkiste* beschrieben.

## 26.14 Die GRASS Werkzeugkiste

Die  GRASS-Werkzeugkiste ermöglicht es, GRASS Module auf Daten innerhalb einer ausgewählten GRASS LOCATION und MAPSET anzuwenden. Dazu muss im Vorfeld eine GRASS LOCATION und MAPSET geöffnet werden, in der Sie Schreibrechte besitzen. Dies ist normalerweise garantiert, wenn Sie die MAPSET selbst erstellt haben und notwendig, damit die Ergebniskarten der Raster- und Vektoranalysen in der ausgewählten LOCATION und `:file:`MAPSET` gespeichert werden können.

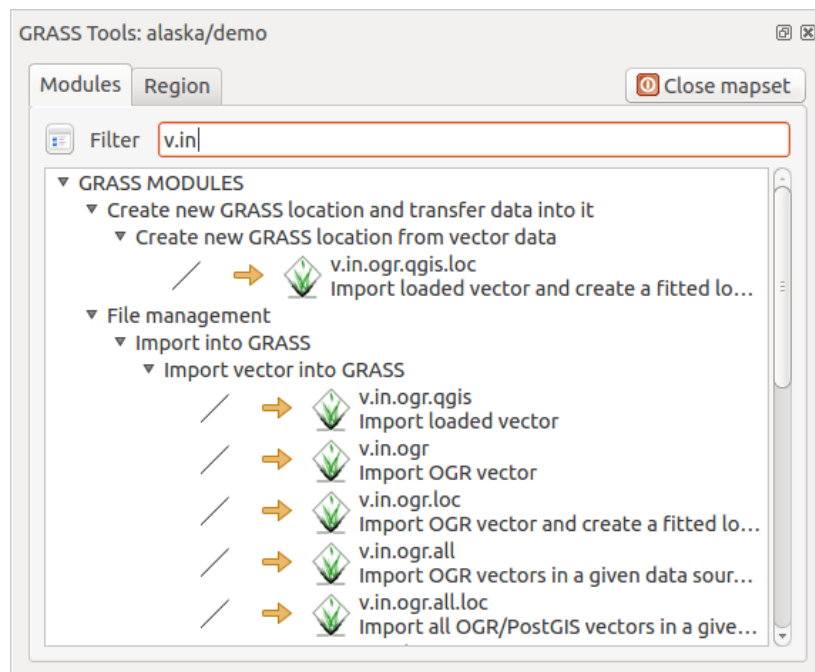


Abb. 26.3: GRASS Werkzeuge und Modulbaum

### 26.14.1 Arbeiten mit GRASS Modulen

Die GRASS Shell der Werkzeugkiste bietet Zugriff auf fast alle (mehr als 300) GRASS Module über die Kommandozeile. Um eine benutzfreundlichere Umgebung zu bieten, sind davon etwa 200 Module graphisch auswählbar und bieten einen Dialog in Form eines zusätzlichen Reiters in der Werkzeugkiste.

Eine vollständige Liste der GRASS-Module, die in der grafischen Toolbox in der QGIS-Version | CURRENT | verfügbar sind ist im GRASS-Wiki unter [https://grasswiki.osgeo.org/wiki/GRASS-QGIS\\_relevant\\_module\\_list](https://grasswiki.osgeo.org/wiki/GRASS-QGIS_relevant_module_list) verfügbar.

Es ist außerdem möglich, die GRASS Werkzeugkiste anzupassen und weitere Module zu integrieren. Die Herangehensweise ist in Abschnitt *Anpassen der Module* beschrieben.

As shown in Abb. 26.3, you can look for the appropriate GRASS module using the thematically grouped *Modules Tree* or the searchable *Modules List* tab.

Wenn Sie auf das grafische Icon eines Modules klicken, öffnet sich ein neuer Moduldialog mit drei Reitern *Optionen*, *Ergebnis* und *Handbuch*.

### Optionen

Der Reiter *Optionen* stellt Ihnen in vereinfachter Form die unbedingt notwendigen Eingabeparameter zur Verfügung, die das Modul zum Laufen benötigt.

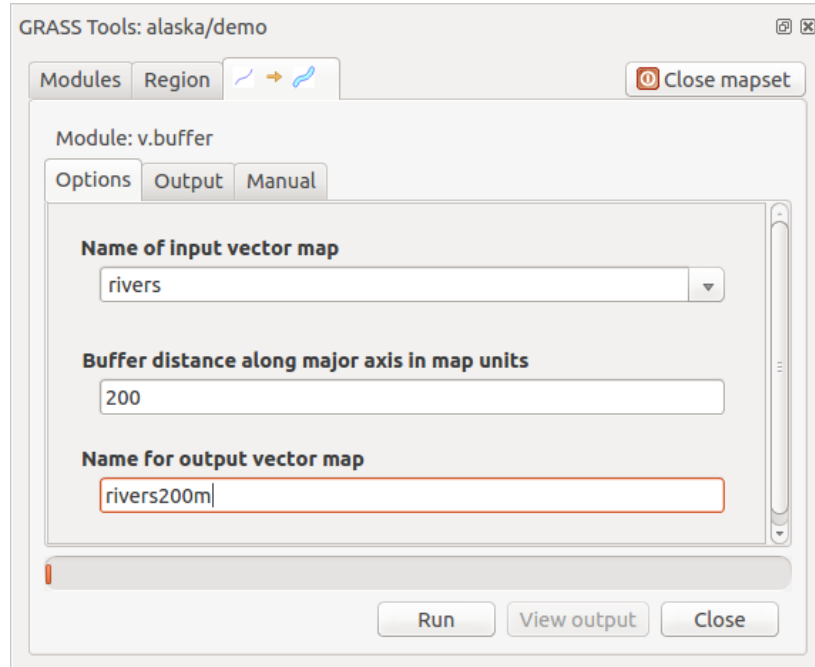


Abb. 26.4: GRASS Werkzeuge Modul Optionen

Die zur Verfügung gestellten Modulparameter sind oftmals nicht vollständig um den Dialog einfach zu halten. Wenn Sie weitergehende Modulparameter und -flags verwenden wollen müssen Sie die GRASS Shell starten und das Programm in der Kommandozeile benutzen.

Eine neue Funktion seit der QGIS Version 1.8.0 ist der Knopf *Fortgeschrittene Optionen einblenden* unterhalb der vereinfachten Moduldialoge im Reiter *Optionen*. Momentan ist diese Funktion nur für das Modul `v.in.ascii` umgesetzt. Es ist aber zukünftig für weitere Module vorgesehen und wird so die kompletten Funktionen der GRASS Module grafisch bereitstellen, ohne die GRASS Shell und somit die Kommandozeile benutzen zu müssen.

### Ergebnis

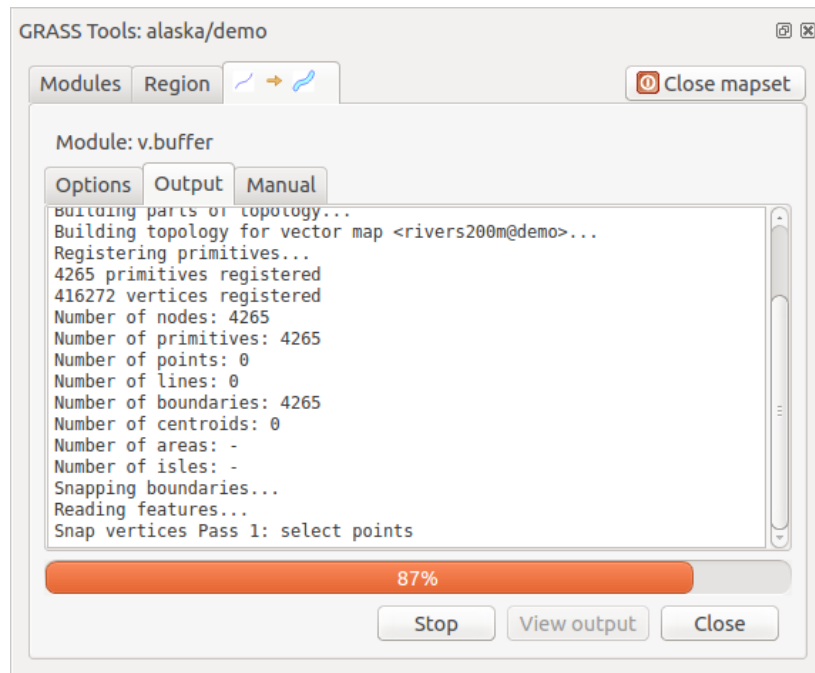


Abb. 26.5: GRASS Werkzeuge Modul Ergebnis

Der Reiter *Ergebnis* stellt Informationen über den Ausgabestatus des Moduls bereit. Nach Klick auf den *Starten* Knopf wechselt das Modul zum Reiter *Ergebnis* und zeigt Informationen über die Abarbeitung des Prozesses. Wenn alles funktioniert, erscheint schlussendlich die Meldung *Erfolgreich beendet*.

## Handbuch

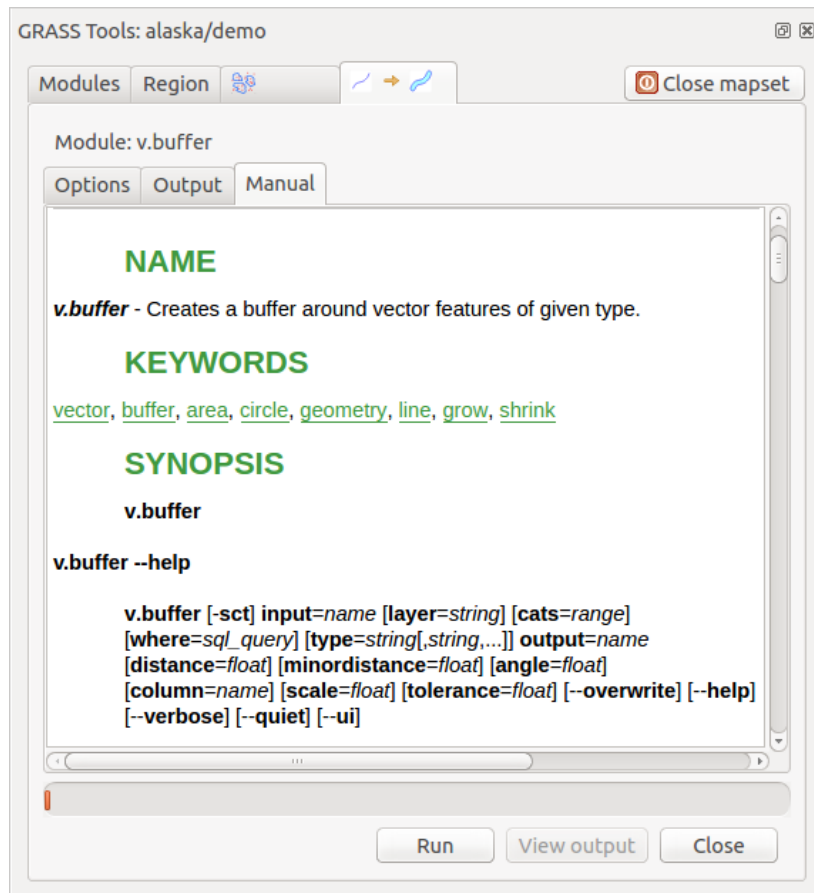


Abb. 26.6: GRASS Werkzeuge Modul Handbuch

Der Reiter *Handbuch* zeigt die HTML Hilfeseite der GRASS Module. Sie können es zum Überprüfen weitergehender Modulparameter oder -flags oder um vertieftere Kenntnisse über den Anwendungszweck der Module zu erwerben benutzen. Am Ende jeder Modul Handbuchseite sehen Sie weiterführende Links zum `Main Help index`, dem `Thematic index` und dem `Full index`. Diese Links stellen die gleiche Information wie das Modul `g.manual` zur Verfügung.

#### **Tipp: Ergebnisse direkt anzeigen**



Wollen Sie Ihre Ergebnisse direkt in der Kartenansicht ansehen, nutzen Sie den Knopf ‚Ergebnis visualisieren‘ im unteren Bereich des jeweiligen Modulreiters

## 26.14.2 GRASS Beispielanwendung

Die folgenden Beispiele sollen die Anwendung verschiedener GRASS Module demonstrieren.

### Höhenlinien aus einem DGM erstellen

Im ersten Beispiel wird eine Höhenlinienkarte aus einem Höhenraster (DEM) erstellt. Hier wird angenommen dass Sie die Alaska LOCATION wie in Abschnitt [Daten in eine GRASS LOCATION importieren](#) eingerichtet haben.

- Als erstes öffnen Sie die Location, indem Sie auf das  Mapset öffnen Icon klicken und dann die Alaska Location auswählen.
- Als nächstes drücken Sie auf das Icon  GRASS-Werkzeugkiste öffnen.
- In der Liste von Werkzeugkategorien doppelklicken Sie auf *Raster ► Oberflächenverwaltung ► Vektorkonturlinien erzeugen*.
- Jetzt wird ein einfacher Klick auf das Werkzeug **r.contour** einen Werkzeugdialog wie oben beschrieben öffnen (siehe [Arbeiten mit GRASS Modulen](#)).
- Geben Sie `gtopo30` in *Name des Eingaberasters* ein.
- Geben Sie im Feld *Abstand zwischen den Kontourintervallen*  den Wert 100 an. (Dadurch werden Höhenlinien in einem Abstand von 100m erstellt.)
- In das Feld *Name der Vektorausgabekarte* geben Sie den Namen `hoehen_100` an.
- Klicken Sie auf *Starten* um die Verarbeitung zu starten. Warten Sie bis die Meldung **Erfolgreich beendet** im Ergebnissenster erscheint. Drücken Sie dann auf *Ergebnis visualisieren* und *Schließen*.

Da die aktuelle GRASS Region ziemlich groß ist, kann es eine Weile dauern, bis der Layer vollständig dargestellt wird. Danach können Sie noch die Layereigenschaften verändern und eine Farbe für die Linien auswählen, die sich deutlich vom Höhenmodell unterscheidet, siehe [Vektorlayereigenschaften](#).

Als nächstes zoomen Sie in einen bergigen Bereich im Zentrum Alaskas. Wenn Sie weit genug in die Karte hineingezoomt sind, werden Sie erkennen, dass die Höhenlinien teilweise sehr eckig erscheinen. Um das Erscheinungsbild zu optimieren, bietet GRASS ein Modul mit dem Namen **v.generalize**. Dabei wird mit Hilfe des Douglas Peuker Algorithmus und der einer Reduktion der Stützpunkte eine Glättung der Linien erreicht, ohne die Geometrien zu zerstören. Da der Ergebnislayer weniger Stützpunkte hat, ist er auch kleiner und kann schneller geladen werden. Die Analyse wird z.B. angewendet, wenn man sehr detaillierte Daten nur in einem kleinen Maßstab anzeigen möchte.

---

#### **Typ:** Geometrien in QGIS vereinfachen

Beachten Sie, dass es in QGIS das Werkzeug *Vektor ► Geometrie-Werkzeuge ► Vereinfachen* gibt, dass ähnlich dem GRASS **v.generalize** Douglas-Peuker-Algorithmus arbeitet.

---

In diesem Beispiel wollen wir nun aber etwas anderes erreichen. Die Höhenlinien, die wir mit **r.contour** erstellt haben, zeigen teilweise sehr scharfe Winkel, die wir glätten möchten. Unter den Algorithmen des Moduls **v.generalize** befindet sich auch der Chaikens-Algorithmus, der exakt das macht, was wir möchten (auch Hermite Splines). Achten Sie aber darauf, dass es passieren kann, dass Stützpunkte nicht nur eliminiert sondern auch **hinzugefügt** werden können. Dadurch kann der Layer wieder langsamer geladen werden.

- Öffnen Sie die GRASS Werkzeugkiste und doppelklicken Sie die Kategorien *Vektor ► Karte entwickeln ► Generalisierung* und klicken Sie dann auf das **v.generalize** Modul um sein Optionsfenster zu öffnen.
- Stellen Sie sicher, dass `hoehen_100` als Vektorlayer in Feld *Name der Vektoreingabekarte* erscheint.
- Wählen Sie Chaiken's Algorithm aus der Liste der Generalisierungs Algorithmen. Lassen Sie alle anderen Optionen auf der Voreinstellung. Geben Sie in der untersten Zeile im Feld *Name der Ausgabe-Vektorkarte* `hoehen_100_smooth` ein und klicken auf *Starten*.



- Die Ausführung dauert einen Moment. Wenn Erfolgreich beendet im Ergebnisfenster erscheint, klicken Sie auf *View Output* und dann auf *Schließen*.
- Ändern Sie nun auch die Farbe des neuen Layers, damit er sich deutlich von dem Höhenmodell und den zuvor berechneten Höhenlinien abhebt. Sie werden erkennen, dass die Kanten der neuen Höhenlinien wesentlich weicher gezeichnet sind.



Abb. 26.7: GRASS Modul v.generalize zum Glätten einer Vektorkarte

#### **Tipp: Vektorlinien glätten mit dem GRASS Modul v.generalize**

Die oben beschriebene Anwendung kann auch in anderen Situationen verwendet werden. Wenn Sie z.B. eine Rasterkarte mit Niederschlagswerten haben, können Sie mit `r.contour` einen Isohyetallayer (konstanter Niederschlag) erstellen.

### **Erstellen eines 3D Schummerungseffekts**

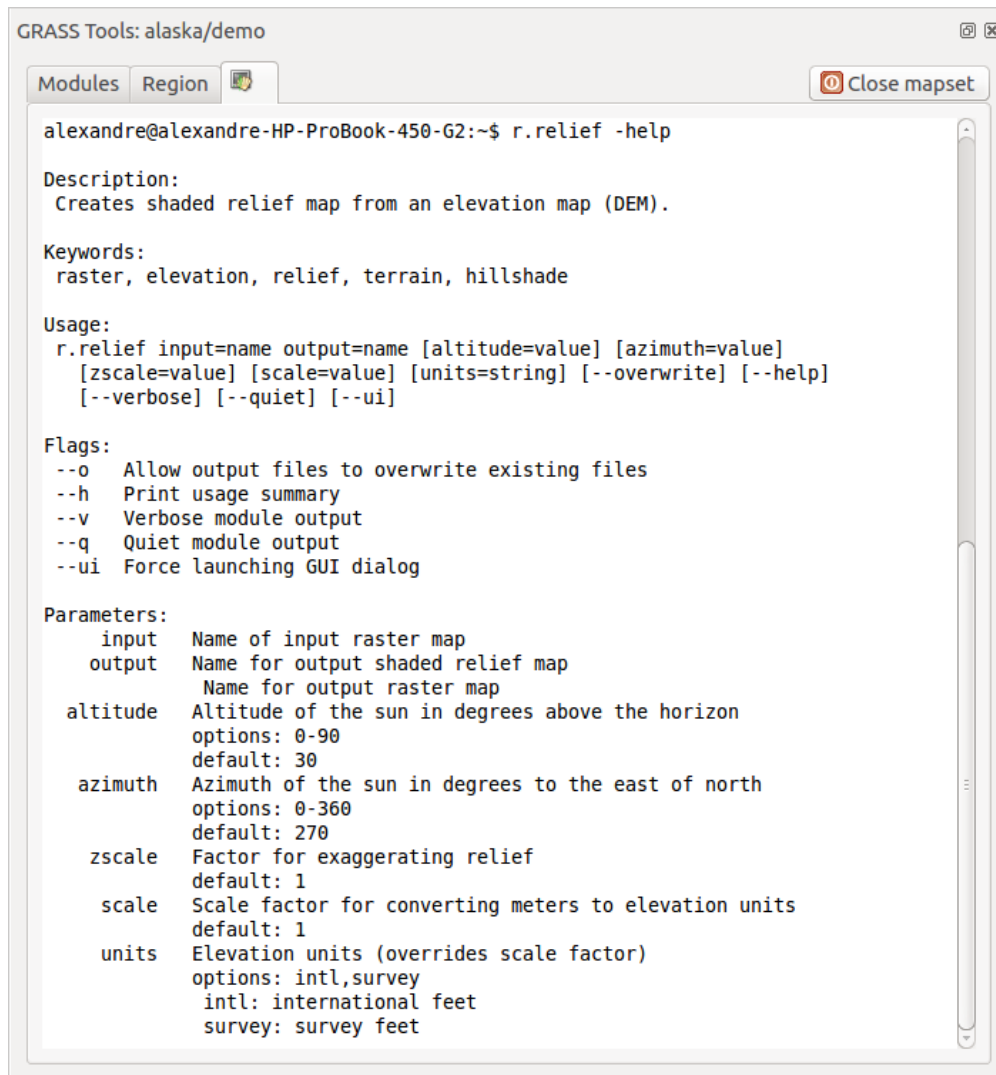
Es gibt verschiedene Methoden, um Höhenlayer anzuzeigen und ihnen einen 3D Schummerungseffekt zu verleihen. Der Gebrauch von Höhenlinien ist eine populäre Methode, die häufig angewendet wird, um topographische Karten zu erstellen. Eine andere Möglichkeit, um einen 3D Effekt zu erzeugen ist, bietet das Hillshading. Der Effekt basiert dabei auf einem Höhenmodell. Dabei wird zuerst die Hangneigung und -richtung der Zellen bestimmt und dann durch die Simulation des Sonnenstandes eine Reflexionswert erzeugt. Dadurch werden der Sonne zugewandte Bereiche aufgehellt und der Sonne abgewandte Bereiche (im Schatten) dunkler dargestellt.

- Fangen Sie mit diesem Beispiel an indem Sie zuerst den Rasterlayer `gtopo30` laden. Öffnen Sie die GRASS-Werkzeugkiste und unter der Raster Kategorie wechseln Sie in den Bereich *Räumliche Analysen* ► *Geländeanalyse*.
- Nun klicken Sie auf **r.shaded.relief**, um den Modulreiter zu öffnen.
- Ändern Sie den Wert im Feld *Winkel der Sonne in Grad östlich von der Nordrichtung*  270 auf 315.
- Geben Sie `gtopo30_shade` für das neu zu erstellende Raster mit Schummerung ein und klicken auf *Starten*.
- Nachdem die Karte berechnet wurde, visualisieren Sie sie und setzen Sie die Farbe des Rasterlayers auf Graustufen.
- Um die Schummerung und das Höhenmodell `gtopo30` zusammen zu sehen, ziehen Sie die Schummerungskarte in der Legende unter das Höhenmodell. Öffnen Sie dann den Dialog Layereigenschaften der Karte `gtopo30`, und ändern Sie die Transparenz im Reiter *Transparenz* auf den Wert 25%.

Die Höhenkarte `gtopo30` wird nun als Farbkarte leicht transparent über der Schummerungskarte angezeigt. Dadurch entsteht ein visueller 3D Effekt. Um den Unterschied besser zu erkennen, wechseln Sie über das Kontrollkästchen den Anzeigemodus der Schummerungskarte in der Legende und wieder zurück.

## Die GRASS Kommandozeile verwenden

Das GRASS Plugin in QGIS stellt die GRASS Module oftmals in vereinfachter Form und auch nicht vollständig bereit. Es ist also grundsätzlich für Anwender gestaltet, die sich nicht so gut mit GRASS und all seinen Fähigkeiten auskennen oder nur einfacher Analysen durchführen wollen. Daher werden in den grafischen Moduldialogen oftmals nicht alle Optionen und Parameter, die das GRASS Modul bieten bereitgestellt, um den Umgang einfacher und intuitiver zu gestalten. Wer tiefer in GRASS einsteigen möchte, er hat die Möglichkeit, sämtliche Funktionalitäten und Module über die GRASS Kommandozeile (GRASS Shell) anzusprechen. In dem folgenden Beispiel soll eine zusätzliche Option des Moduls **r.shaded.relief** angesprochen werden, die nur über die Kommandozeile genutzt werden kann.



```

GRASS Tools: alaska/demo

Modules Region

alexandre@alexandre-HP-ProBook-450-G2:~$ r.relief -help

Description:
  Creates shaded relief map from an elevation map (DEM).

Keywords:
  raster, elevation, relief, terrain, hillshade

Usage:
  r.relief input=name output=name [altitude=value] [azimuth=value]
  [zscale=value] [scale=value] [units=string] [--overwrite] [--help]
  [--verbose] [--quiet] [--ui]

Flags:
  --o Allow output files to overwrite existing files
  --h Print usage summary
  --v Verbose module output
  --q Quiet module output
  --ui Force launching GUI dialog

Parameters:
  input      Name of input raster map
  output     Name for output shaded relief map
             Name for output raster map
  altitude   Altitude of the sun in degrees above the horizon
             options: 0-90
             default: 30
  azimuth    Azimuth of the sun in degrees to the east of north
             options: 0-360
             default: 270
  zscale     Factor for exaggerating relief
             default: 1
  scale      Scale factor for converting meters to elevation units
             default: 1
  units      Elevation units (overrides scale factor)
             options: intl,survey
             intl: international feet
             survey: survey feet
  
```

Abb. 26.8: Die GRASS Shell, Modul r.shaded.relief

Das Modul **r.shaded.relief** stellt einen zusätzliche Parameter **zmult** bereit, über den der Höhenwert relativ zu den X-Y Werten multipliziert werden kann. Dadurch wird der Schummerungseffekt noch prägnanter.

- Laden Sie das Höhenraster **gtopo30** wie zuvor und starten dann die GRASS Toolbox und klicken auf die GRASS Shell. Geben Sie im Shell Fenster das Kommando **r.shaded.relief map=gtopo30 shade=gtopo30\_shade2 azimuth=315 zmult=3** ein und drücken Enter.
- Wenn die Berechnung abgeschlossen ist, wechseln Sie den Reiter *Browser* und Doppelklicken Sie auf die neu erstellte Karte **gtopo30\_shade2**, um Sie in QGIS anzuzeigen.
- Wie oben erklärt verschieben Sie die Schummerungskarte unter das **gtopo30** Raster in der Legende, überprüfen Sie dann die Transparenz des farbigen **gtopo30** Layers. Sie sollten dabei erkennen, dass der Schummerungseffekt stärker verglichen mit der ersten Schummerungskarte ist.

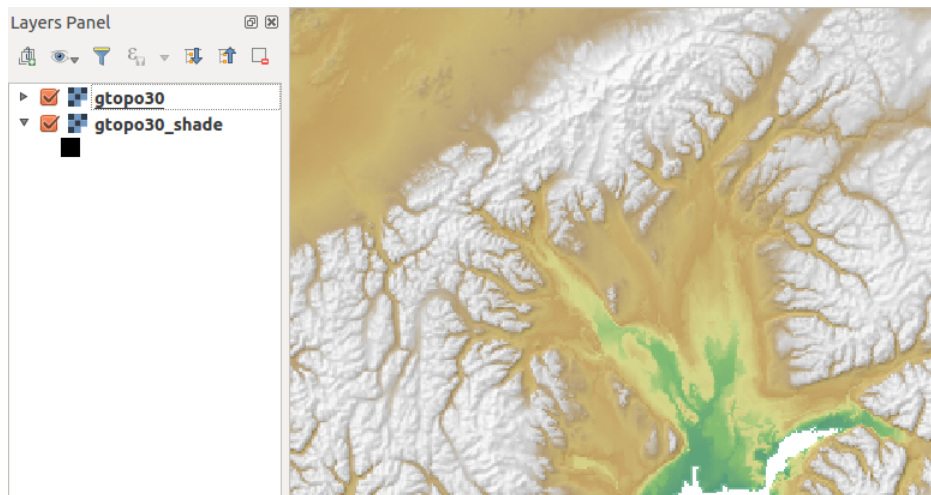



Abb. 26.9: Darstellen einer Schummering erstellt mit dem GRASS Modul `r.shaded.relief`

### Rasterstatistik auf Basis eines Vektorlayer berechnen

Das folgende Beispiel zeigt, wie man univariate Statistik für Rasterwerte innerhalb von Vektorpolygonen berechnen kann und die Werte in neue Attributspalten des Vektorlayers hinzufügt.

- Es werden wieder die Alaska Daten verwendet. Unter *Daten in eine GRASS LOCATION importieren* findet man Hinweise zum Import der Datei `shapefiles/trees.shp` nach GRASS.
- Jetzt ist ein Zwischenschritt nötig: es müssen Zentroiden zu der importierten Baumkarte hinzugefügt werden um es zu einem kompletten GRASS Flächenvektor (der sowohl Umrandungen als auch Zentroide enthält) zu machen.
- Wählen Sie aus der Werkzeugkiste *Vektor ► Karte entwickeln ► Objekte verwalten* und öffnen Sie das Modul **v.centroids**.
- Geben Sie als *Name für die Ausgabe-Vektorkarte* `forest_areas` an und starten Sie das Modul.
- Nun laden Sie den neuen Layer `forest_areas` und visualisieren Sie die verschiedenen Waldtypen in verschiedenen Farben - deciduous, evergreen und mixed. Dazu öffnen Sie den Dialog *Eigenschaften* des Layers, wechseln zum Reiter *Darstellung*, wählen  „Eindeutiger Wert“ und setzen das *Klassifikationsfeld* auf `VEGDESC` (siehe auch die Beschreibung des Reiters *Darstellung* in Abschnitt *Eigenschaften Symbolisierung* des Vektordatenabschnitts).
- Als nächstes öffnen Sie wieder die GRASS-Werkzeugkiste und wechseln nach *Vektor ► Vektor mit anderen Karten aktualisieren*.
- Klicken Sie auf das **v.rast.stats** Modul. Geben Sie `gtopo30` und `forest_areas` ein.
- Ein weiterer Parameter wird benötigt: Geben Sie bei *column prefix* `elev` ein und klicken auf *Starten*. Dies ist eine rechenintensive Operation, die längere Zeit in Anspruch nimmt (wahrscheinlich bis zu zwei Stunden).
- Schließlich öffnen Sie den Layer `forest_areas` erneut und lassen Sie sich die Attributtabelle anzeigen. Dort gibt es nun weitere Spalten einschließlich `elev_min`, `elev_max`, `elev_mean` usw. für jedes Waldpolygon.

### 26.14.3 Anpassen der Module

Nahezu alle GRASS-Module können in die GRASS-Werkzeugkiste integriert werden. Eine XML-Schnittstelle wertet die sehr einfachen XML-Dateien, die die Module beschreiben, aus und übernimmt die Oberflächendarstellung.

Beispielhaft ist hier die XML-Datei zum Modul `v.buffer` (`v.buffer.qgm`) dargestellt:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE qgisgrassmodule SYSTEM "http://mrcc.com/qgisgrassmodule.dtd">

<qgisgrassmodule label="Vector buffer" module="v.buffer">
  <option key="input" typeoption="type" layeroption="layer" />
  <option key="buffer" />
  <option key="output" />
</qgisgrassmodule>
```

Der Parser liest diese Definition aus und erstellt bei Auswahl des Moduls einen neuen Reiter in der Toolbox. Eine detaillierte Beschreibung zum Hinzufügen neuer Module, zum Ändern der Modulgruppe usw. findet man unter <https://qgis.org/en/site/getinvolved/development/addinggrasstools.html>.

---


QGIS Verarbeitung Umgebung

---

## 27.1 Einführung

Dieses Kapitel stellt das QGIS Verarbeitung Umgebung, eine leistungsstarke Analyseumgebung für raumbezogene Daten in QGIS, vor. Verarbeitung ist eine Geoverarbeitungsumgebung, die verwendet wird, um eigene und fremde Algorithmen aus QGIS heraus aufrufen zu können, so dass räumliche Analysen produktiv und einfach zu bewerkstelligen sind.

As a *Core plugin*, Processing is installed by default but you need to activate it:

1. Go to *Plugins ► Manage and install plugins...*
2. Click on the *Installed* tab at the left
3. Check the box next to the  *Processing* entry
4. Close the dialog.

A *Processing* menu is now available in the top menu bar. From there you can reach the main components of this framework.

In den folgenden Abschnitten werden wir betrachten wie die grafischen Elemente dieser Umgebung benutzt werden und wie man das Beste aus jedem einzelnen von ihnen herausholen kann.

There are four basic elements in the framework GUI, which are used to run algorithms for different purposes. Choosing one tool or another will depend on the kind of analysis that is to be performed and the particular characteristics of each user and project. All of them (except for the batch processing interface, which is called from the toolbox or the algorithm execution dialog, as we will see) can be accessed from the *Processing* menu item (you will see more entries; the remaining ones are not used to execute algorithms and will be explained later in this chapter).

- The *Toolbox*: The main element of the GUI, it is used to execute a single algorithm or run a batch process based on that algorithm.

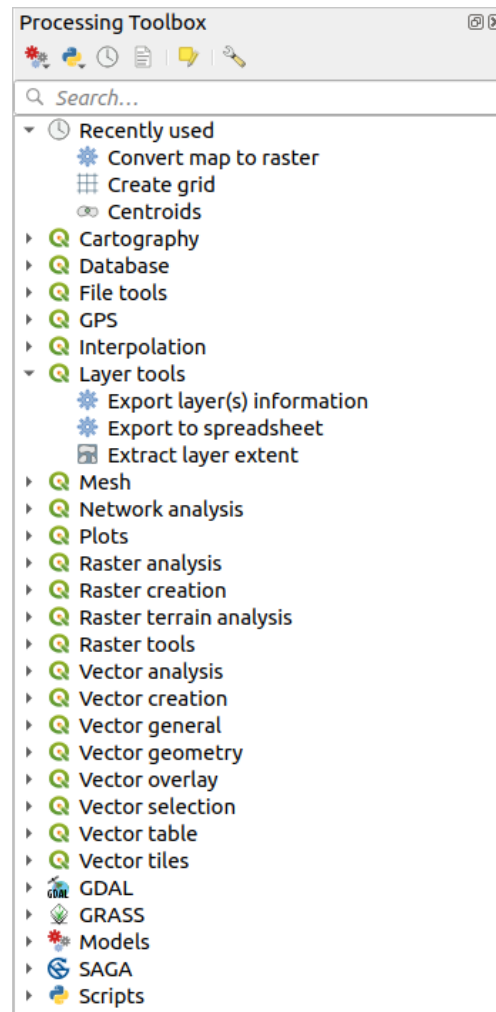


Abb. 27.1: Verarbeitungswerkzeuge

- The *Model Designer*: Several algorithms can be combined graphically using the modeler to define a workflow, creating a single process that involves several subprocesses.

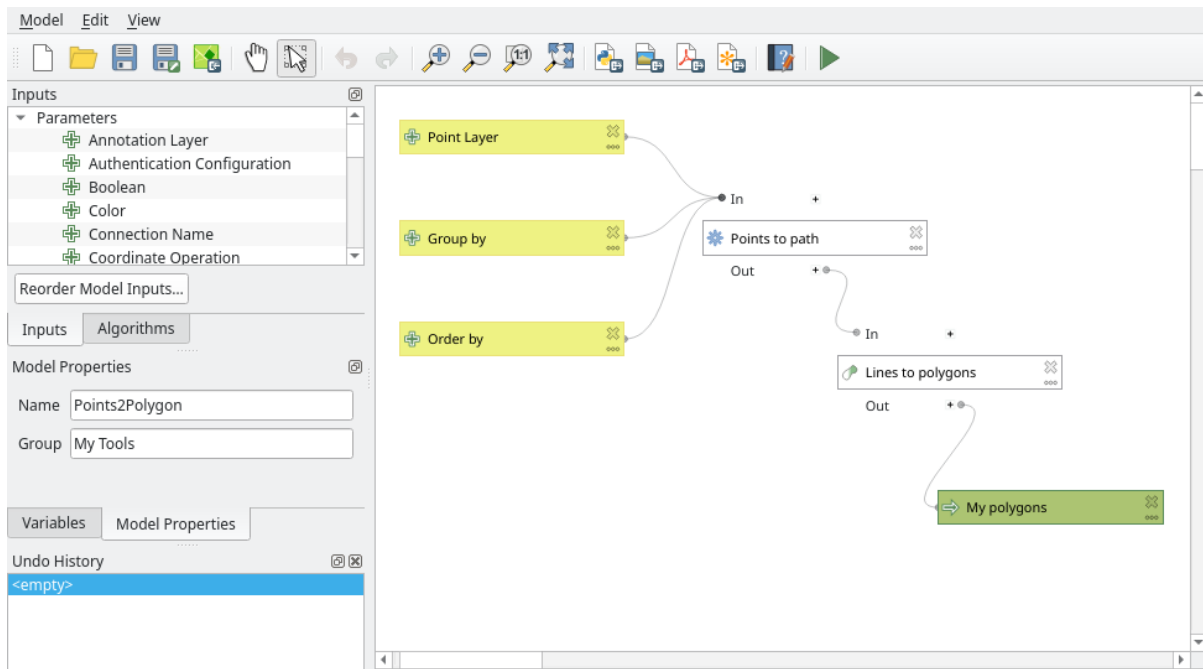


Abb. 27.2: Verarbeitungsmodellierung

- The *History* manager: All actions performed using any of the aforementioned elements are stored in a history file and can be later easily reproduced using the history manager.

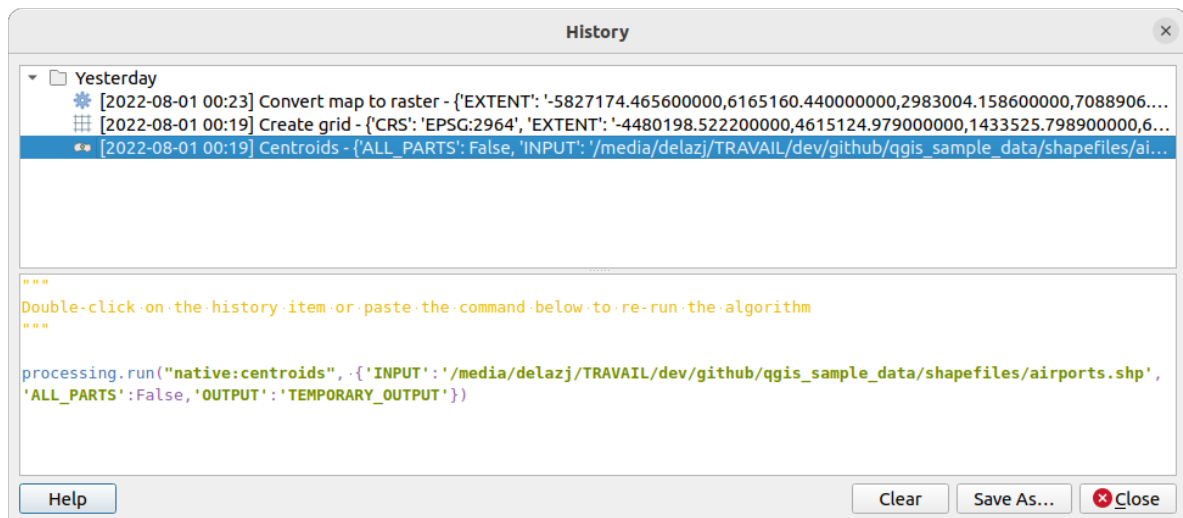


Abb. 27.3: Verarbeitung Protokoll

- The *Batch Processing* interface: This interface allows you to execute batch processes and automate the execution of a single algorithm on multiple datasets.

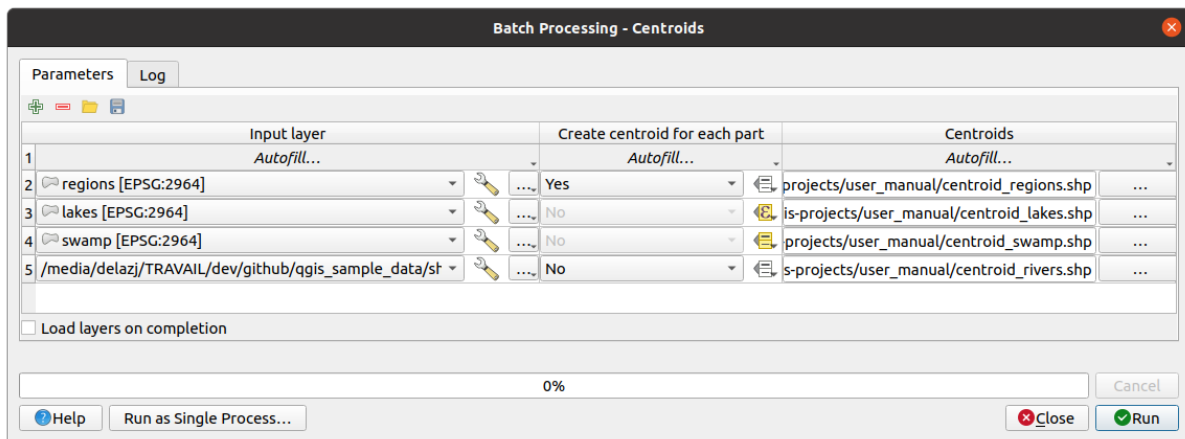



Abb. 27.4: Batch Processing Schnittstelle

In den folgenden Abschnitten werden wir auf jedes dieser Elemente im Detail eingehen.

## 27.2 Configuring the Processing Framework

The Processing Options menu (*Settings* ► *Options* ►  *Processing* tab) allows you to configure how algorithms work. Configuration parameters are structured in separate blocks that you can select on the left-hand side of the dialog.

### 27.2.1 General

The *General* block contains the default settings to control how the algorithm dialog as well as input or output parameters should behave. Some of the settings can however be overridden at the algorithm level, per *algorithm run* or for *individual parameters*.

- *Default output raster layer extension* is by default `tif`
- *Default output vector layer extension* is by default `gpkg`
- *Invalid features filtering* when executing algorithm:
  - *Do not filter (better performance)*: all the features (with valid and invalid geometries) are processed, but the result may be erroneous depending on how the geometry invalidity affects the operations
  - *Skip (ignore) features with invalid geometries*, meaning that only a subset of your dataset (the valid geometry features) will be processed
  - *Stop algorithm execution when a geometry is invalid*: you'll need to track and fix the invalid geometries if you want the algorithm to process the whole layer. Algorithms like *Check validity* or *Fix geometries* can help you achieve this.

The *Invalid features filtering* setting can be overridden on a per-input basis, at algorithm runtime.

- *Keep dialog open after running algorithm*. Once an algorithm has finished execution and its output layers are loaded into the QGIS project, the algorithm dialog is closed. If you want to keep it open (to run the algorithm again with different parameters, or to better check the output that is written to the log tab), check this option.
- *Max Threads*
- *Output folder* for non temporary outputs: If no folder path is provided for the Processing execution outputs, this is the folder in which they will be saved. Default is `processing/outputs` under the active *user profile* directory.



- *Override temporary output folder path*: Temporary outputs are saved by default in the `tmp` folder on the machine. This option helps you set a different place for storage.
- *Pre-execution script* and *Post-execution script*. These parameters point to files that contain scripts written using the processing scripting functionality, explained in the section covering scripting and the console.
- *Prefer output filename for layer names*. The name of each resulting layer created by an algorithm is defined by the algorithm itself. In some cases, a fixed name might be used, meaning that the same output name will be used, no matter which input layer is used. In other cases, the name might depend on the name of the input layer or some of the parameters used to run the algorithm. If this checkbox is checked, the name will be taken from the output filename instead. Notice that, if the output is saved to a temporary file, the filename of this temporary file is usually a long and meaningless one intended to avoid collision with other already existing filenames.
- *Results group name*. If you want to obtain all processing result layers in a group in the *Layers* panel, set a group name for this parameter. The group may exist already or not. QGIS will add all output layers to such a group. By default, this parameter is empty, so all output layers are added to different places in the *Layers* panel, depending on the item that is active when running an algorithm. Note that output layers will be loaded to the *Layers* panel only if *Open output file after running algorithm* is checked in the algorithm dialog.
- *Show algorithms with known issues*: By default, QGIS avoids display of broken algorithms (generally from third-party providers). If checked, they will be available in the Processing toolbox, with a warning icon and a tooltip explaining they have issues. Use at your own risks.
- *Show feature count for output vector layers*. Since calculating the feature count might take some time in certain data formats, this option is off by default.
- *Show layer CRS definition in selection boxes*
- *Show tooltip when there are disabled providers*
- *Style for line layers*, *Style for point layers*, *Style for polygons layers* and *Style for raster layers* are used for setting the default rendering style for output layers (that is, layers generated by processing algorithms). Just create the style you want using QGIS, save it to a file, and then enter the path to that file in the settings so the algorithms can use it. Whenever a layer is loaded by Processing and added to the QGIS canvas, it will be rendered with that style.

Rendering styles can be configured individually for each algorithm and each one of its outputs. Just right-click on the name of the algorithm in the toolbox and select *Edit rendering styles for outputs*. You will see a dialog like the one shown next.

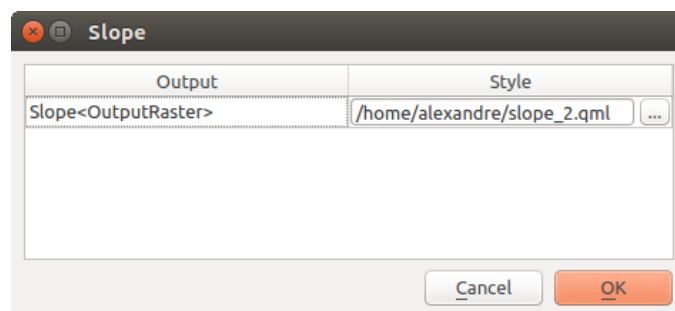




Abb. 27.5: Rendering Styles

Select the style file (`.qml`) that you want for each output and press *OK*.

- *Warn before executing if parameter CRS's do not match*: By default, QGIS native algorithms (i.e. the ones listed under the  *Menu* ► *QGIS (native C++)* group) transparently reproject the input layers to the first one's CRS before execution. Check this option to get a notification from the other tools that do not support reprojection, when the inputs CRS are not identical. Third-party providers are not concerned.



## 27.2.2 Menus

The  *Menus* block controls whether an algorithm, script or model (built-in or provided by plugins) should be made available through a dedicated menu or toolbar (along with the Processing Toolbox). For each item of each provider, you can:


- *Add button in toolbar*, making it available in the *Processing Algorithms* toolbar
- assign an *Icon* to the algorithm
- set a *Menu path*: the algorithm will then be available through an existing or a custom menu, e.g. *Vector/MyTopAlgorithms*

Restart QGIS to apply the settings. At any time, your changes can be *Reset to defaults*.

## 27.2.3 Models and Scripts

In the  *Models* and  *Scripts* blocks, you can set a default folder to store, and look for models and scripts respectively.

## 27.2.4 Providers

You will also find a block for algorithm  *Providers*. This is the place where installed providers expose their settings. For example, built-in providers contain an *Activate* item that you can use to make their algorithms appear or not in the toolbox. Some algorithm providers have their own configuration items, which will be explained when covering particular algorithm providers.

## 27.3 The Toolbox

The *Processing Toolbox* is the main element of the processing GUI, and the one that you are more likely to use in your daily work. It shows the list of all available **algorithms** grouped in different blocks called *Providers*, and custom **models** and **scripts** you can add to extend the set of tools. Hence the toolbox is the access point to run them, whether as a single process or as a batch process involving several executions of the same algorithm on different sets of inputs.

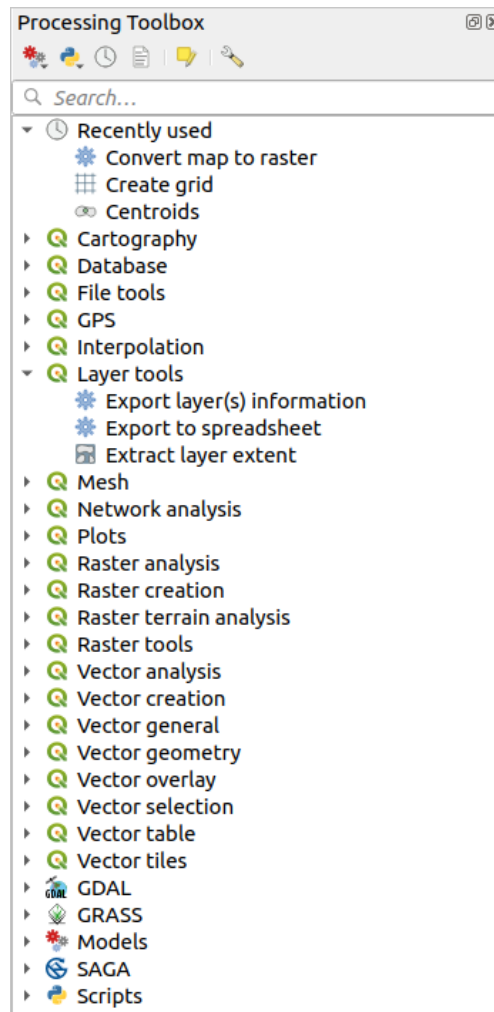









Abb. 27.6: Verarbeitungswerkzeuge

Providers can be (de)activated in the *Processing settings dialog*. By default, only providers that do not rely on third-party applications (that is, those that only require QGIS elements to be run) are active. Algorithms requiring external applications might need additional configuration. Configuring providers is explained in a *later chapter* in this manual.

In the upper part of the toolbox dialog, you will find a set of tools to:

- work with  **Models**: *Create New Model...*, *Open Existing Model...* and *Add Model to Toolbox...*;
- work with  **Scripts**: *Create New Script...*, *Create New Script from Template...*, *Open Existing Script...* and *Add Script to Toolbox...*;
- open the  **History** panel;
- open the  **Results Viewer** panel;
- toggle the toolbox to the *in-place modification mode* using the  **Edit Features In-Place** button: only the algorithms that are suitable to be executed on the active layer without outputting a new layer are displayed;
- open the  **Options** dialog.

Below this toolbar is a  **Search...** box to help you easily find the tools you need. You can enter any word or phrase on the text box. Notice that, as you type, the number of algorithms, models or scripts in the toolbox is reduced to just those that contain the text you have entered in their names or keywords.

**Bemerkung:** At the top of the list of algorithms are displayed the most recent used tools; handy if you want to reexecute any.

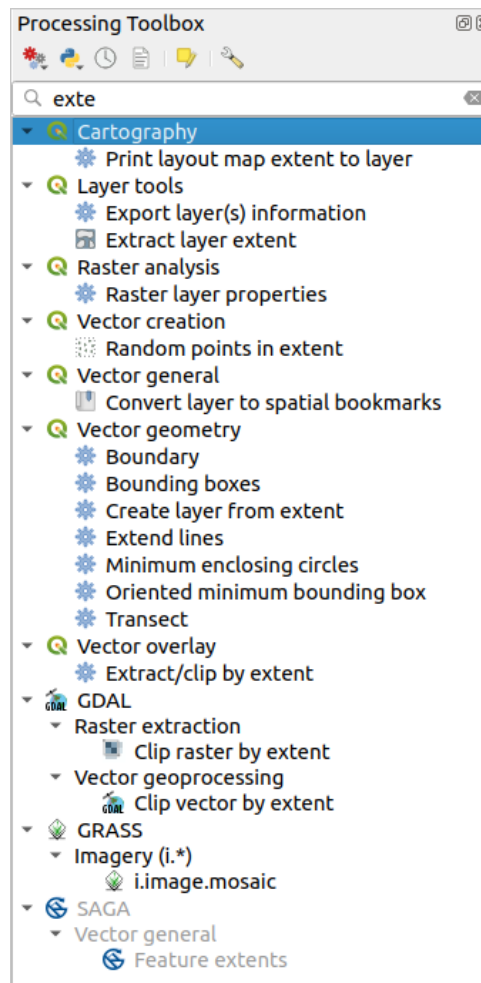


Abb. 27.7: Die Verarbeitungswerkzeuge zeigen Suchergebnisse

To execute a tool, just double-click on its name in the toolbox.

### 27.3.1 Der Algorithmus Dialog

Once you double-click on the name of the algorithm that you want to execute, a dialog similar to that in the [Abb. 27.8](#) below is shown (in this case, the dialog corresponds to the `Centroids` algorithm).

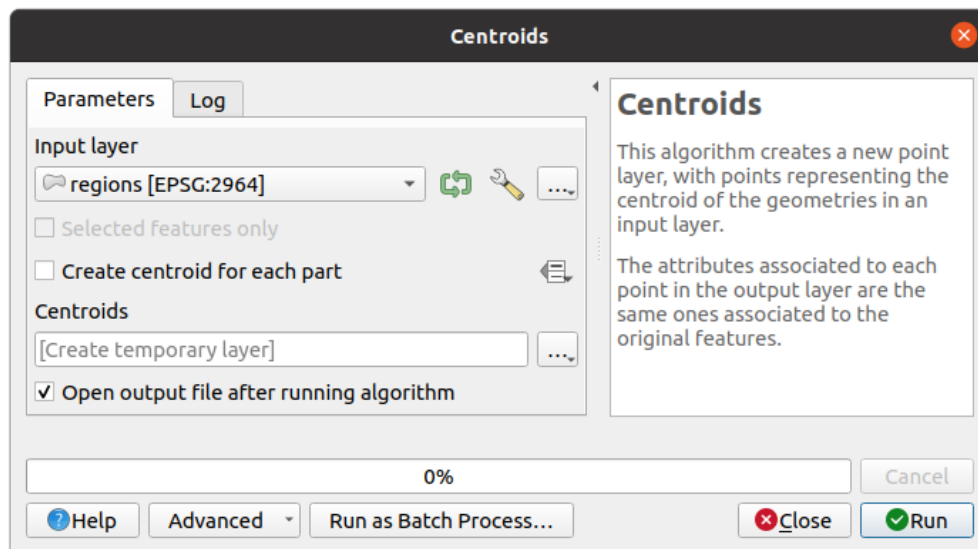


Abb. 27.8: Algorithm Dialog - Parameters

The dialog shows two tabs (*Parameters* and *Log*) on the left part, the algorithm description on the right, and a set of buttons at the bottom.

### Parameter types

The *Parameters* tab is used to set the input values that the algorithm needs to be executed. It shows a list of input values and configuration parameters to be set. It of course has a different content, depending on the requirements of the algorithm to be executed, and is created automatically based on those requirements.

#### Tipp: Setting your own default values for algorithm parameters

Algorithm dialogs open with some parameters prefilled with values from QGIS installation. It is however possible to set *your own default values* for specific algorithm parameters so that they are used at algorithm startup.

Obwohl die Anzahl und Art der Parameter von den Eigenschaften des Algorithmus abhängt, ist die Struktur für alle ähnlich. Die Parameter die Sie in der Tabelle finden können können folgende Typen sein.

- A **vector layer**, to select from a list of all vector layers available (currently opened) in QGIS. You can also use unloaded layers: press the ... button on the widget right-hand side, and select:
  - *Select file...*: selects file on disk using the Operating System file explorer
  - *Browse for layer...*: opens the *Browser panel*, allowing to take the layers directly from database sources (PostgreSQL, SQL Server, Oracle, ...), web services (WFS, AFS, ...) or files on disk.

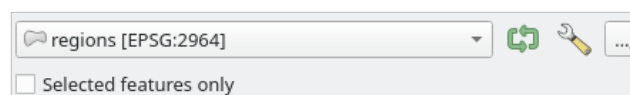




Abb. 27.9: Vector input widget

**Bemerkung:** By default, the layer widget shows the CRS of the layer along with its name. If you do not want to see this additional information, you can disable this functionality in the Processing Settings dialog, unchecking the *General* ► *Show layer CRS definition in selection boxes* option.

The vector input widget also has following features:

- an iterator  button: If toggled, the algorithm will be executed iteratively on each one of its features, instead of just once for the whole layer, producing as many outputs as times the algorithm is executed. This allows for automating the process when all features in a layer have to be processed separately. If the algorithm contains several input vectors you can iterate over, the iteration will be processed only on the first toggled parameter, in the order parameters are declared in the algorithm.
-  Advanced options button to adjust settings to use for that specific parameter. These settings concern:
  - \* *Invalid feature filtering*: allows the *default method* for handling features with invalid geometries to be overridden
  - \* *Limit features processed*: optional limit on number of features processed from the source
  - \* *Feature filter*: allows to enter an expression to subset the layer dynamically when running the tool, avoiding the need for separate steps to set layer filters or create layer subsets.

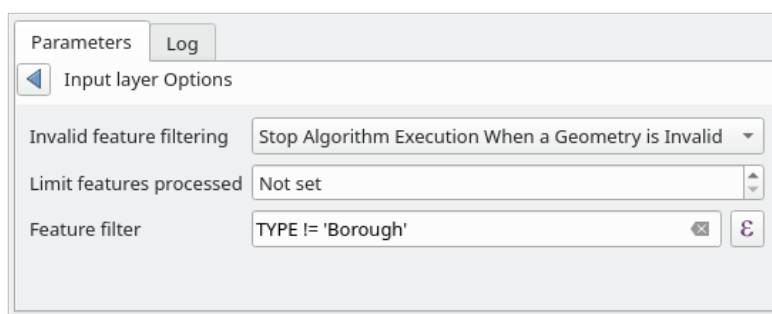


Abb. 27.10: Advanced options for vector input widget

- It is also possible to limit the algorithm execution on the vector layer to its *Selected features only*.
- A **table**, to select from a list of all available in QGIS. Non-spatial tables are loaded into QGIS like vector layers, and use the *same widget*.
- A **raster layer**, to select from a list of all raster layers available in QGIS. The selector contains as well a ... button on its right-hand side, to let you select filenames that represent layers currently not loaded in QGIS.

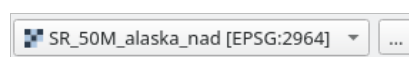



Abb. 27.11: Raster input widget

- An **option**, to choose from a selection list of possible options.
- A **numerical value**, to be introduced in a spin box. In some contexts (when the parameter applies at the feature level and not at the layer's), you will find a  Data-defined override button by its side, allowing you to open the *expression builder* and enter a mathematical expression to generate variable values for the parameter. Some useful variables related to data loaded into QGIS can be added to your expression, so you can select a value derived from any of these variables, such as the cell size of a layer or the northernmost coordinate of another one.

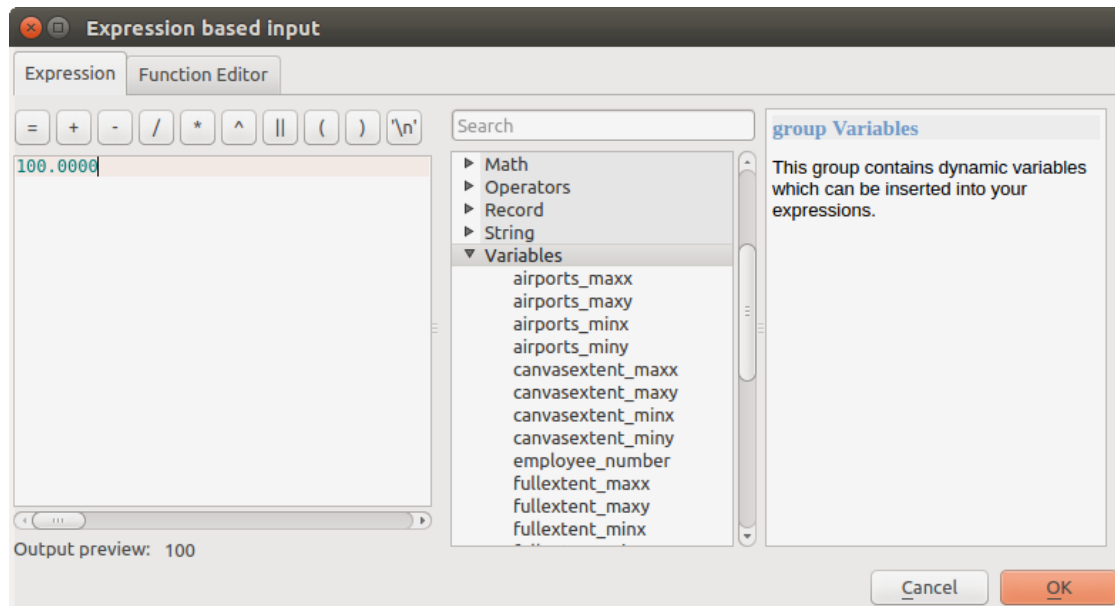




Abb. 27.12: Expression based input

- A **range**, with min and max values to be introduced in two text boxes.
- A **text string**, to be introduced in a text box.
- A **field**, to choose from the attributes table of a vector layer or a single table selected in another parameter.
- A **coordinate reference system**. You can select it among the recently used ones from the drop-down list or from the *CRS selection* dialog that appears when you click on the button on the right-hand side.
- An **extent**, a text box defining a rectangle through its corners coordinate in the format `xmin, xmax, ymin, ymax`. Press the  Set to current map canvas extent button to use the map canvas extent. Clicking the arrow on the right-hand side of the value selector, a pop-up menu will appear, giving you options to:
  - *Calculate from layer* ►: fills the text box with the coordinates of the bounding box of a layer to select among the loaded ones
  - *Calculate from layout map* ►: fills the text box with the coordinates of a map item selected from a layout in the current project
  - *Calculate from bookmark* ►: fills the text box with the coordinates of a saved bookmark
  -  *Use current map canvas extent*
  - *Draw on canvas*: the parameters window will hide itself, so you can click and drag onto the canvas. Once you have defined the extent rectangle, the dialog will reappear, containing the values in the extent text box.

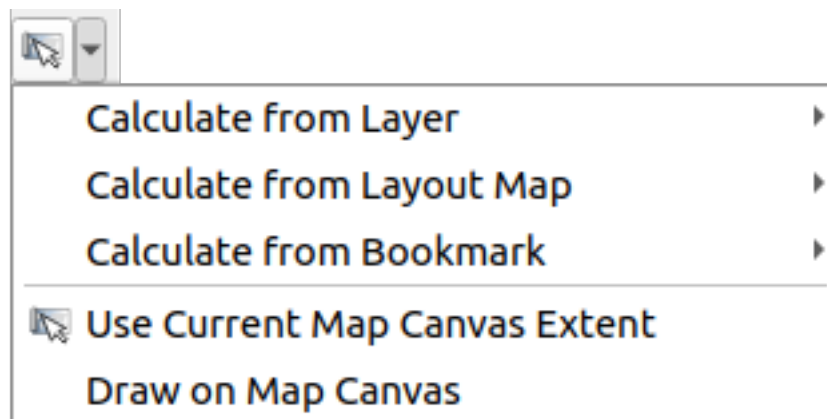


Abb. 27.13: Ausdehnung Auswahl

- A **list of elements** (whether raster or vector layers, tables, fields) to select from. Click on the ... button at the left of the option to see a dialog like the following one. Multiple selection is allowed and when the dialog is closed, number of selected items is displayed in the parameter text box widget.

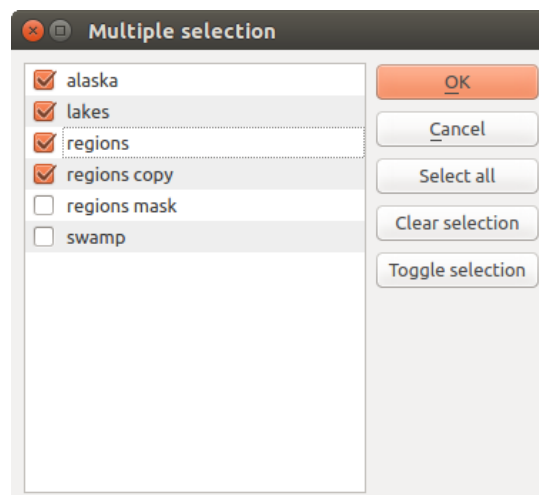


Abb. 27.14: Mehrfachauswahl

- A **small table** to be edited by the user. These are used to define parameters like lookup tables or convolution kernels, among others.

Klicken Sie auf den Knopf auf der rechten Seite, um die Tabelle zu sehen und zu editieren.

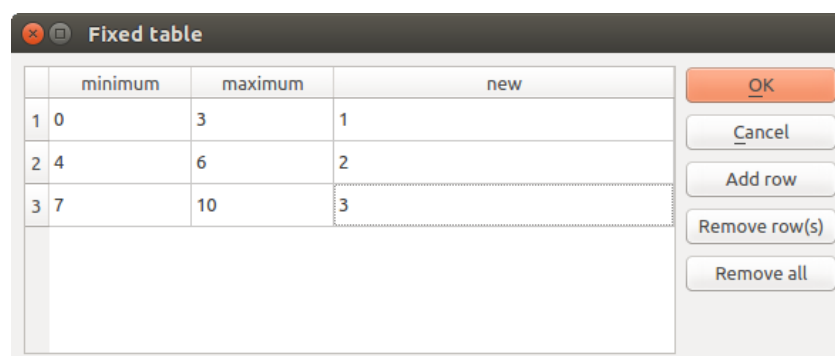


Abb. 27.15: Fixe Tabelle



In Abhängigkeit vom Algorithmus kann die Anzahl der Zeilen verändert werden oder auch nicht, indem Sie auf den Knopf rechts neben dem Fenster klicken.

**Bemerkung:** Some algorithms require many parameters to run, e.g. in the *Raster calculator* you have to specify manually the cell size, the extent and the CRS. You can avoid to choose all the parameters manually when the algorithm has the *Reference layers* parameter. With this parameter you can choose the reference layer and all its properties (cell size, extent, CRS) will be used.

## Logging the execution

Along with the *Parameters* tab, there is another tab named *Log* (see Abb. 27.16 below). Information provided by the algorithm during its execution is written in this tab, allowing you to track the execution as well as being aware and having more details about the algorithm as it runs. Information on algorithm execution is also output in the *View ► Panels ► Log Messages Panel*.

Notice that not all algorithms write information to the *Log* tab, and many of them might run silently without producing any output other than the final files. Check the *Log Messages Panel* in that case.

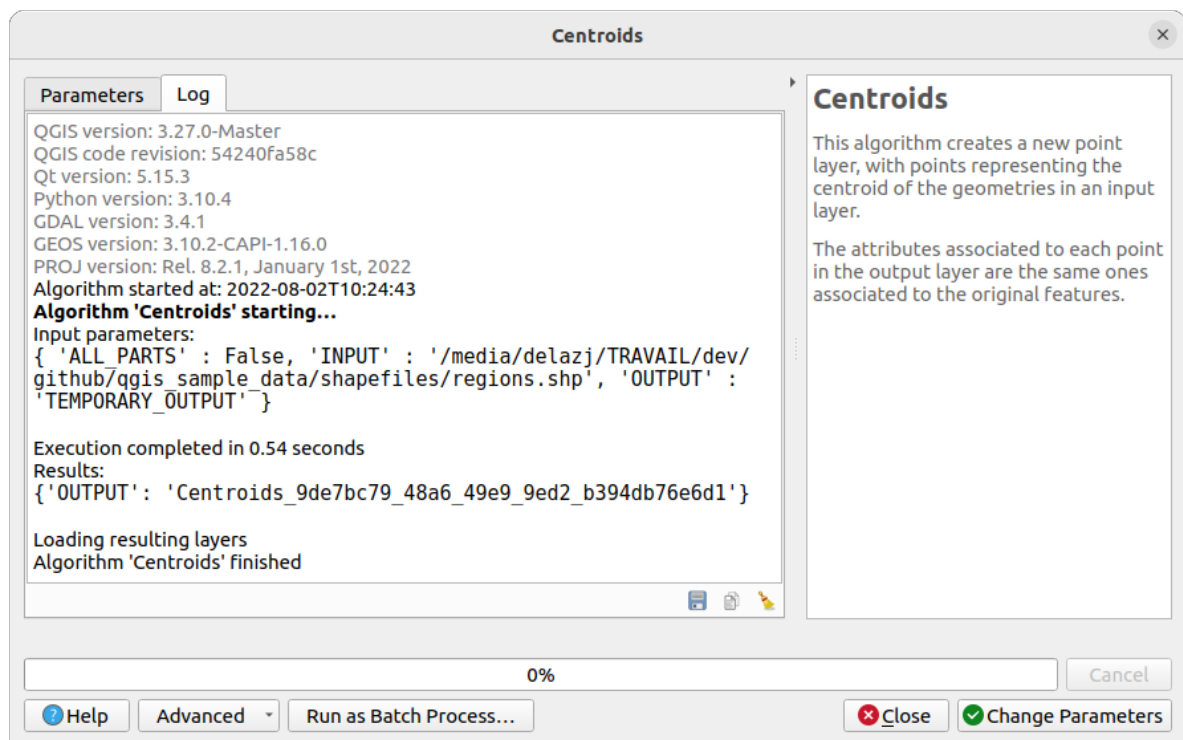





Abb. 27.16: Algorithm Dialog - Log






At the bottom of the *Log* tab you will find buttons to  *Save Log to File*,  *Copy Log to Clipboard* and  *Clear Log*. These are particularly handy when you have checked the *Keep dialog open after running algorithm* in the *General* part of the Processing options.

### Other tools

On the right hand side of the dialog you will find a short description of the algorithm, which will help you understand its purpose and its basic ideas. If such a description is not available, the description panel will not be shown.

For a more detailed help file, which might include description of every parameter it uses, or examples, you will find a *Help* button at the bottom of the dialog bringing you to the [Processing algorithms documentation](#) or to the provider documentation (for some third-party providers).


The *Advanced* ► menu provides functions to reuse the configuration defined in the dialog without running the algorithm:

-  *Algorithm Settings...*: allows to override processing settings for the current algorithm execution. More details at [Override algorithm settings](#).
-  *Copy as Python Command*: allows for easy copying of the equivalent *PyQGIS command* to run the tool using the parameters defined in the dialog
-  *Copy as qgis\_process Command*: allows for easy generation of *qgis\_process command*, including its environment settings like the distance units, area units, ellipsoid, and any tricky parameter values like GeoPackage outputs with specific layers
-  *Copy as JSON*: all the settings of the command are copied in a JSON format, ready to be consumed by *qgis\_process*. This is a convenient way to see the expected format of the commands, even for complex parameters (like TIN interpolation parameters). You can store these easily and then restore them later by pasting the values.
-  *Paste Settings in a JSON format*

The *Run as Batch Process...* button triggers the [batch processing mode](#) allowing to configure and run multiple instances of the algorithm with a variety of parameters. A *Run as Single Process...* helps you switch back from the batch mode.

When an algorithm execution finishes (either successfully or not), a new button *Change Parameters* is shown as long as the *Log* tab is active.

### Override algorithm settings

Triggered from within the *Advanced* drop-down menu at the bottom of an algorithm dialog, the  *Algorithm Settings...* shows a panel allowing users to control general processing settings which apply to that algorithm execution only. It is intended to be a place where a user can override their [global processing settings](#) on an ad-hoc basis without having to change their usual default settings.

Settings that can be overridden are:

- *Invalid feature filtering*: unlike the existing per-parameter setting override for this, setting the handling method here will apply to **ALL inputs** for the algorithm
- *Calculation settings*, such as *Distance units* and *Area units* to use for distance/area measurements
- *Environment settings*, such as *Temporary folder* and *Number of threads to use*

## Bemerkung zum Thema Projektionen

Processing algorithm execution are always performed in the input layer coordinate reference system (CRS). Due to QGIS's on-the-fly reprojecting capabilities, although two layers might seem to overlap and match, that might not be true if their original coordinates are used without reprojecting them onto a common coordinate system. Whenever you use more than one layer as input to a *QGIS native algorithm*, whether vector or raster, the layers will all be reprojected to match the coordinate reference system of the first input layer.

This is however less true for most of the external applications whose algorithms are exposed through the processing framework as they assume that all of the layers are already in a common coordinate system and ready to be analyzed.

By default, the parameters dialog will show a description of the CRS of each layer along with its name, making it easy to select layers that share the same CRS to be used as input layers. If you do not want to see this additional information, you can disable this functionality in the Processing settings dialog, unchecking the *Show layer CRS definition in selection boxes* option.

If you try to execute an algorithm using as input two or more layers with unmatching CRSs, a warning dialog will be shown. This occurs thanks to the *Warn before executing if layer CRS's do not match* option.

Sie können den Algorithmus immer noch ausführen, seien Sie sich aber dessen bewusst dass dies in den meisten Fällen zu falschen Ergebnissen führt, so z.B. leere Layer aufgrunddessen dass Eingabelayer nicht überlappen.

---

### **Tipp: Use Processing algorithms to do intermediate reprojection**

When an algorithm can not successfully perform on multiple input layers due to unmatching CRSs, use QGIS internal algorithm such as *Reproject layer* to perform layers' reprojection to the same CRS before executing the algorithm using these outputs.

---

## 27.3.2 Von Algorithmen erstellte Datenobjekte

Von Algorithmen erstellte Datenobjekte können jeder der folgenden Typen sein:

- Rasterlayer
- Vektorlayer
- Tabelle
- HTML-Datei (wird für Text und grafische Ausgabe verwendet)

These are all saved to disk, and the parameters table will contain a text box corresponding to each one of these outputs, where you can type the output channel to use for saving it. An output channel contains the information needed to save the resulting object somewhere. In the most usual case, you will save it to a file, but in the case of vector layers, and when they are generated by native algorithms (algorithms not using external applications) you can also save to a PostGIS, GeoPackage or Spatialite database, or a memory layer.

Um einen Ausgabekanal auszuwählen, klicken Sie einfach auf den Knopf auf der rechten Seite des Textfelds und Sie werden ein kleines Kontextmenü mit den verfügbaren Optionen sehen.

Im häufigsten Fall werden Sie Speichern in einer Datei auswählen. Wenn Sie diese Option auswählen, werden Sie in einem Datei speichern Dialog aufgefordert werden, den gewünschten Dateipfad auszuwählen. Unterstützte Dateierweiterungen werden in der Dateiformatauswahl des Dialogs gezeigt, abhängig von der Art der Ausgabe und des Algorithmus.

The format of the output is defined by the filename extension. The supported formats depend on what is supported by the algorithm itself. To select a format, just select the corresponding file extension (or add it, if you are directly typing the file path instead). If the extension of the file path you entered does not match any of the supported formats, a default extension will be appended to the file path, and the file format corresponding to that extension will be used to save the layer or table. Default extensions are `.dbf` for tables, `.tif` for raster layers and `.gpkg` for vector layers. These can be modified in the setting dialog, selecting any other of the formats supported by QGIS.

If you do not enter any filename in the output text box (or select the corresponding option in the context menu), the result will be saved as a *temporary file* in the corresponding default file format, and it will be deleted once you exit QGIS (take care with that, in case you save your project and it contains temporary layers).

You can set a default folder for output data objects. Go to the settings dialog (you can open it from the *Settings ► Options ► Processing* menu), and in the *General* group, you will find a parameter named *Output folder*. This output folder is used as the default path in case you type just a filename with no path (i.e., `myfile.shp`) when executing an algorithm.

Wenn Sie einen Algorithmus ausführen, der einen Vektorlayer im iterativen Modus verwendet, wird der eingegebene Dateipfad als Basispfad für alle erstellten Dateien, die mit Hilfe des Basisnamens benannt werden und an die eine Nummer, die den Index der Iteration darstellt, angehängt wird, verwendet. Die Dateierweiterung (und das Format) wird für alle so erstellten Dateien benutzt.

Apart from raster layers and tables, algorithms also generate graphics and text as HTML files. These results are shown at the end of the algorithm execution in a new dialog. This dialog will keep the results produced by any algorithm during the current session, and can be shown at any time by selecting *Processing ► Results Viewer* from the QGIS main menu.

Einige externe Anwendungen können Dateien (ohne besondere Einschränkungen bei der Dateinamen-Erweiterung haben) als Ausgabe erzeugen, die nicht zu einer der oben genannten Kategorien gehören. Diese Ausgabedateien werden nicht von QGIS verarbeitet (weder geöffnet noch in das aktuelle QGIS-Projekt integriert), da diese meistens nicht von QGIS unterstützt werden. Dies ist beispielsweise der Fall beim LAS-Format für LIDAR-Daten. Die Dateien werden erstellt, aber Sie werden nicht in QGIS angezeigt.

Für alle anderen Arten von Ausgaben finden Sie ein Kontrollkästchen, mit dem sie festlegen können, ob die Datei in **lqgl** geladen werden soll oder nicht, wenn sie durch den Algorithmus erzeugt wurde. Standardmäßig werden alle Dateien angezeigt.

Optionale Ausgaben werden nicht unterstützt, alle Ausgaben werden erstellt, aber Sie können über das entsprechende Kontrollkästchen definieren, wenn Sie an einer bestimmten Ausgabe nicht interessiert sind, was es im Wesentlichen zu einer Optionale Ausgabe macht (Der Layer wird zwar erstellt, aber nur temporär im Hintergrund, wenn Sie das Textfeld leer lassen, und wieder gelöscht, sobald Sie QGIS verlassen).

## 27.4 Das Protokoll

### 27.4.1 Das Verarbeitung Protokoll

Jedes Mal, wenn Sie einen Algorithmus ausführen, werden Informationen über den Prozess im Verlaufsmanager gespeichert. Datum und Uhrzeit der Ausführung werden zusammen mit den verwendeten Parametern gespeichert, was es einfach macht, alle Arbeiten, die mit dem Verarbeitungsframework entwickelt wurden, zu verfolgen, zu kontrollieren und zu reproduzieren.

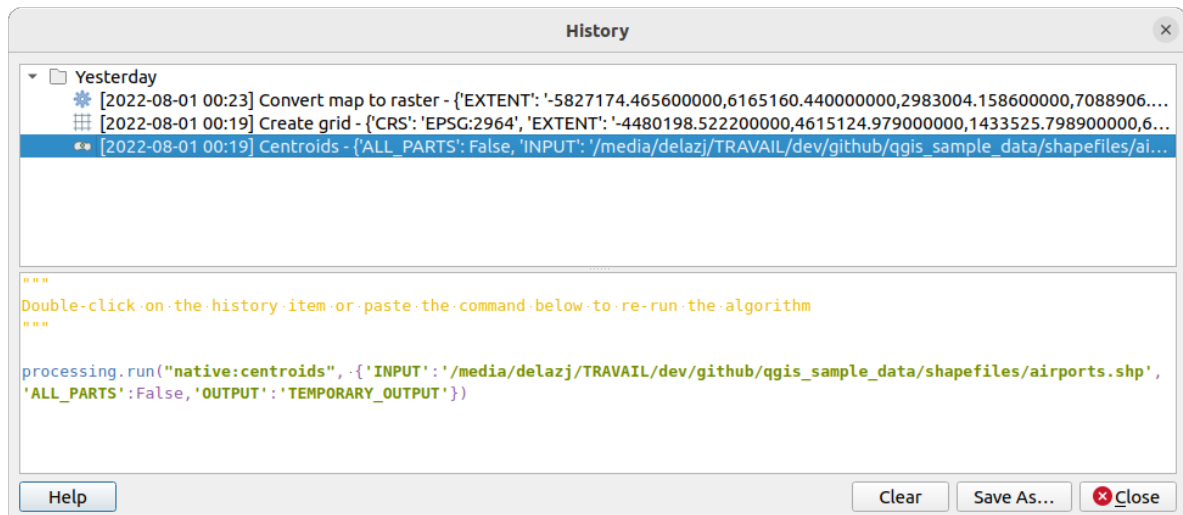


Abb. 27.17: Protokoll

Prozessinformationen werden als Befehlszeilenausdruck gespeichert, selbst wenn der Algorithmus aus der Toolbox gestartet wurde. Dies macht es nützlich für diejenigen, die lernen möchten, wie man die Befehlszeilenschnittstelle verwendet, da sie einen Algorithmus mit der Toolbox aufrufen und dann das Protokoll nutzen können, um zu sehen, wie er von der Befehlszeile aus aufgerufen werden könnte.

Right-click on a row and you should be able to:

- *Copy as Python Command*: allows for easy copying of the equivalent *PyQGIS command* run from the dialog. Same as the code displayed below the commands list.
- *Copy as qgis\_process Command*: allows for easy generation of *qgis\_process command*, including its environment settings like the distance units, area units, ellipsoid, and any tricky parameter values like GeoPackage outputs with specific layers
- *Copy as JSON*: all the settings of the command are copied in a JSON format, ready to be consumed by qgis\_process. This is a convenient way to see the expected format of the commands, even for complex parameters (like TIN interpolation parameters). You can store these easily and then restore them later by pasting the values over an algorithm dialog.
- *Create Test...* using the concerned algorithm and parameters, following instructions at [Processing README file](#).

Neben dem Durchsuchen der Einträge in der Registrierung können Sie Prozesse auch erneut ausführen, indem Sie einfach auf den Eintrag doppelklicken. Das Algorithmus-Dialogfeld wird dann mit bereits festgelegten Parametern geöffnet, und Sie können jeden davon an Ihre Bedürfnisse anpassen und den Algorithmus erneut ausführen.

The *History* dialog also provides a convenient way to contribute to the consolidation of the testing infrastructure of QGIS Processing algorithms and scripts.

## 27.4.2 Das Verarbeitung Log

Der Verlaufsdialog enthält nur die Ausführungsaufrufe, aber nicht die Informationen, die der Algorithmus bei der Ausführung erzeugt. Diese Informationen werden in das QGIS-Log geschrieben (*View ► Panels ► Log Messages Panel*).

Algorithmen von Drittanbietern werden normalerweise mithilfe ihrer Befehlszeilenschnittstellen ausgeführt, die über die Konsole mit dem Benutzer kommunizieren. Obwohl diese Konsole nicht angezeigt wird, wird normalerweise jedes Mal, wenn Sie einen dieser Algorithmen ausführen, ein vollständiger Dump davon in das Protokoll geschrieben. Um das Protokoll nicht mit diesen Informationen zu überladen, können Sie es für jeden Anbieter im Einstellungsdialog deaktivieren.

Einige Algorithmen geben, auch wenn sie mit den gegebenen Eingabedaten ein Ergebnis liefern können, Kommentare oder zusätzliche Informationen zum Protokollieren aus, wenn sie potenzielle Probleme mit den Daten erkennen, um Sie zu warnen. Stellen Sie sicher, dass Sie diese Meldungen im Protokoll überprüfen, wenn Sie unerwartete Ergebnisse erhalten.

## 27.5 The model designer

The *model designer* allows you to create complex models using a simple and easy-to-use interface. When working with a GIS, most analysis operations are not isolated, rather part of a chain of operations. Using the model designer, that chain of operations can be wrapped into a single process, making it convenient to execute later with a different set of inputs. No matter how many steps and different algorithms it involves, a model is executed as a single algorithm, saving time and effort.

The model designer can be opened from the Processing menu (*Processing ► Model Designer*).

### 27.5.1 The model designer interface






















Abb. 27.18: Model designer










In its main part, the modeler has a working canvas where the structure of the model and the workflow it represents can be constructed.

At the top of the dialog, different menus and the *Navigation* toolbar give access to a variety of tools.












## Model menu

Bezeichnung	Tastenkürz	Navigation Toolbar	Beschreibung
 <i>Validate Model</i>			Checks whether the algorithms and inputs used in the model exist. Convenient before releasing a model.
 <i>Run Model...</i>	F5		Executes the model
<i>Reorder Model Inputs...</i>			Sets the order in which inputs are presented to the user in the algorithm dialog.
<i>Reorder Output Layers...</i>			Sets a specific order which the outputs from the model must use when loading the results into a project.
 <i>Open Model...</i>	Strg+O		Opens a .model3 file for edit or execution
 <i>Save Model</i>	Strg+S		Saves the model to disk as a .model3 file
 <i>Save Model as...</i>	Strg+Shi		Saves the model to disk as a new .model3 file
 <i>Save Model in project</i>			Embeds the model file in the project file, making it available when sharing the project file.
 <i>Edit Model Help...</i>			An interface to document the model, the algorithms, the parameters and outputs, as well as the author and versioning
<i>Export ►</i>			
 ► <i>Export as Image...</i>			Saves the model's graphical design to an image file format (for illustration purpose)
 ► <i>Export as PDF...</i>			Saves the model's graphical design to a PDF file format (for illustration purpose)
 ► <i>Export as SVG...</i>			Saves the model's graphical design to an SVG file format (for illustration purpose)
 ► <i>Export as Script Algorithm...</i>			Generates a python script file including the model's instructions

## Menü Bearbeiten

Bezeichnung	Tastenkürz	Navigation Toolbar	Beschreibung
 <i>Alles wählen</i> <i>Snap selected components to Grid</i>	Strg+A		Selects all the model components in the designer snaps and aligns the elements into a grid
 <i>Wiederholen</i>	Strg+Y		Rollback the latest canceled action. See also the <i>Undo/Redo</i> panel.
 <i>Rückgängig</i>	Strg+Z		Cancel the previous change. See also the <i>Undo/Redo</i> panel.
 <i>Ausschneiden</i>	Strg+X		Cuts a selection of components from the model.
 <i>Kopieren</i>	Strg+C		Copies a selection of components from the model.
 <i>Einfügen</i>	Strg+V		Pastes a cut or copied selection of components from a model to another or within the same model. The selected components keep their original properties and comments.
 <i>Delete selected components</i> <i>Add Group Box</i>	Entf		Removes a component from the model.  Adds a box at the background of related components in order to visually group them. Particularly useful in big models to keep the workflow clean.

## Menü Ansicht

Bezeichnung	Tastenkürz	Navigation Toolbar	Beschreibung
<i>Zoom To ►</i>			Zooms to the selected group box extent
 <i>Hineinzoomen</i>	Ctrl++		
 <i>Herauszoomen</i>	Ctrl+-		
 <i>Zoom to 100%</i>	Strg+1		
 <i>Volle Ausdehnung</i>	Strg+0		Displays all the components in the designer current canvas
 <i>Show Comments</i>			Displays comments associated to every algorithm or input in the model designer
 <i>Enable Snapping</i>			
 <i>Toggle Panel Visibility</i>	Strg+Tab		Switches ON or OFF the <i>panels</i> in the designer



## Bedienfelder

The left part of the window is a section with five panels that can be used to add new elements to the model:

1. *Model Properties*: specify the name (required) of the model and the group in which it will be displayed in the *Processing Toolbox*
2. *Inputs*: all the *input parameters* that could shape your model
3. *Algorithms*: the available *Processing algorithms*
4. *Variables*: Models can contain dedicated *variables* that are unique and only available to them. These variables can be accessed by any expression used within the model. They are useful to control algorithms within a model and control multiple aspects of the model by changing a single variable. The variables can be viewed and modified in the *Variables* panel.
5. *Undo History*: this panel will register everything that happens in the modeler, making it easy to cancel things you did wrong.

## Zu den verfügbaren Algorithmen

Some algorithms that can be executed from the toolbox do not appear in the list of available algorithms when you are designing a model. To be included in a model, an algorithm must have the correct semantic. If an algorithm does not have such a well-defined semantic (for instance, if the number of output layers cannot be known in advance), then it is not possible to use it within a model, and it will not appear in the list of algorithms that you can find in the modeler dialog. On the other hand some algorithms are specific to the modeler. Those algorithms are located within the group ‚Modeler Tools‘.

## 27.5.2 Creating a model

Das Erstellen eines Modells beinhaltet zwei grundlegende Schritte:

1. *Definition von erforderlichen Eingaben*. Diese Eingaben werden dem Parameterfenster hinzugefügt, so dass der Anwender Ihre Werte einstellen kann wenn er das Modell ausführt. Das Modell selber ist ein Algorithmus, also wird das Parameterfenster automatisch erstellt, so wie es mit allen Algorithmen, die in der Verarbeiten Umgebung zur Verfügung stehen, passiert.
2. *Definition des Arbeitsablaufs*. Der Arbeitsablauf wird definiert, indem Algorithmen hinzugefügt werden. In diesen wird festgelegt, wie sie die definierten Eingaben oder die von anderen Algorithmen im Modell erzeugten Ausgaben verwenden.

## Definition von Eingaben

Der erste Schritt besteht darin, die Eingaben für das Modell zu definieren. Die folgenden Elemente befinden sich auf der Registerkarte :guilabel: *Eingaben* auf der linken Seite des Modellierfensters:

Tab. 27.1: List of parameter types for model building

Annotation Layer	Authentication Configuration	Boolean	Color
Coordinate Operation	CRS	Database Schema	Database
Distance	Duration	DXF Layers	Enum
Extent	Field Aggregates	Fields Mapper	File/Folder
Map Layer	Map Theme	Matrix	Mesh Data
Mesh Layer	Multiple Input	Number	Point
Point Cloud Layer	Print Layout	Print Layout Item	Range
Raster Layer	Scale	String	TIN Creation
Vector Field	Vector Layer	Vector Tile Writer Layers	

**Bemerkung:** Wenn Sie mit der Maus über die Eingaben fahren, wird ein Tooltip mit zusätzlichen Informationen angezeigt.

Wenn Sie auf ein Element doppelklicken, wird ein Dialog angezeigt, in dem Sie seine Eigenschaften definieren können. Je nach Parameter enthält der Dialog mindestens ein Grundelement (die Beschreibung, die der Benutzer bei der Ausführung des Modells sieht). Wenn Sie einen numerischen Wert hinzufügen, wie in der nächsten Abbildung zu sehen ist, müssen Sie, zusätzlich zur Beschreibung des Parameters, einen Standardwert und den Bereich der gültigen Werte festlegen.

Abb. 27.19: Definition der Modellparameter

Sie können Ihre Eingabe als zwingend für Ihr Modell definieren, indem Sie das Kontrollkästchen ☒ Zwingend ankreuzen. Durch Ankreuzen des Kontrollkästchens ☐ Erweitert können Sie die Eingabe so einstellen, dass sie sich im Abschnitt Erweitert befindet. Dies ist besonders nützlich, wenn das Modell viele Parameter hat und einige davon nicht trivial sind, Sie diese aber dennoch wählen möchten.

Für jede hinzugefügte Eingabe wird der Modellierungsebene ein neues Element hinzugefügt.



Abb. 27.20: Modellparameter

Sie können Eingaben auch hinzufügen, indem Sie den Eingabetyp aus der Liste an die gewünschte Stelle im Modellierungsfenster ziehen. Wenn Sie einen Parameter einer bestehenden Eingabe ändern möchten, doppelklicken Sie einfach darauf, und es erscheint der gleiche Dialog.

When using a model within another model, the inputs and outputs necessary will be displayed in the canvas.

## Definition des Arbeitsablaufs

In the following example we will add two inputs and two algorithms. The aim of the model is to copy the elevation values from a DEM raster layer to a line layer using the *Drape* algorithm, and then calculate the total ascent of the line layer using the *Climb Along Line* algorithm.

In the *Inputs* tab, choose the two inputs as *Vector Layer* for the line and *Raster Layer* for the DEM. We are now ready to add the algorithms to the workflow.

Die Algorithmen finden Sie auf der Registerkarte *Algorithmen*, ähnlich gruppiert wie in der Verarbeitungs-Werkzeugkiste.

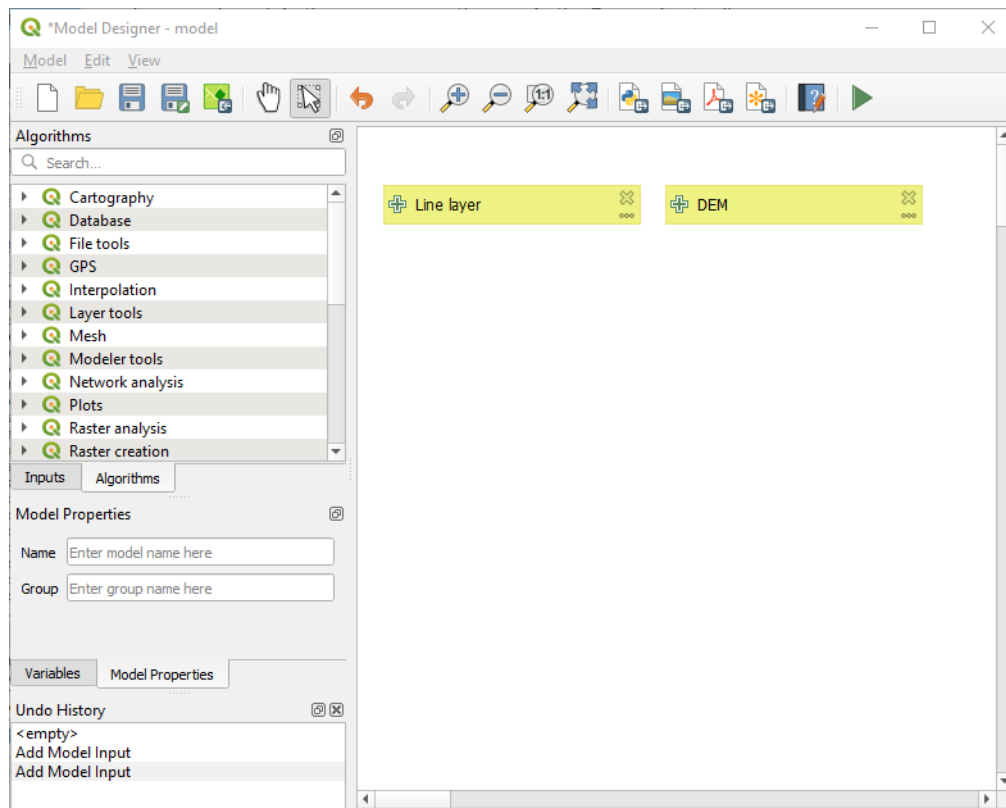


Abb. 27.21: Modell-Eingaben

To add an algorithm to a model, double-click on its name or drag and drop it, just like for inputs. As for the inputs you can change the description of the algorithm and add a comment. When adding an algorithm, an execution dialog will appear, with a content similar to the one found in the execution panel that is shown when executing the algorithm from the toolbox. The following picture shows both the *Drape* (set Z value from raster) and the *Climb along line* algorithm dialogs.

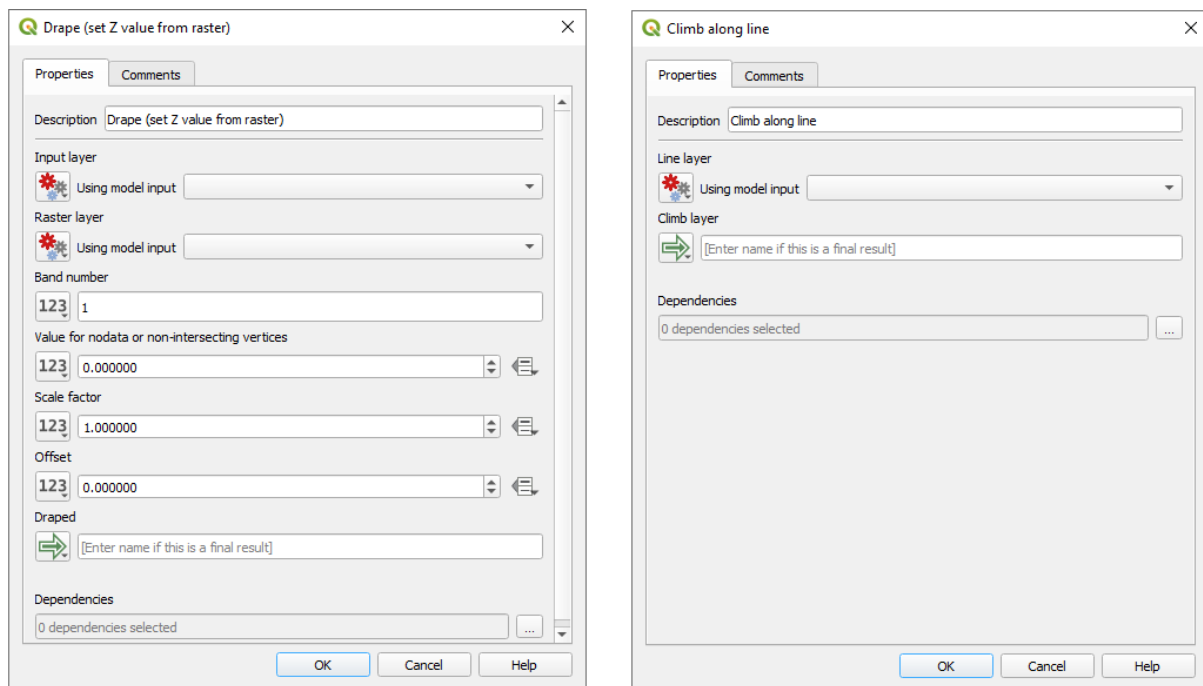







Abb. 27.22: Modell-Algorithmus-Parameter

As you can see, there are however some differences. Each parameter has a drop-down menu next to it allowing to control how it will be served during the workflow:

-  **Value**: allows you to assign a static value to the parameter. Depending on the parameter type, the widget will let you enter a number (5.0), a string (mytext), select layer(s) loaded in the QGIS project or from a folder, pick items from a list, ...
-  **Pre-calculated Value**: opens the *Expression Builder* dialog and lets you define an expression to fill the parameter. Model inputs together with some other layer statistics are available as **variables** and are listed at the top of the Search dialog of the Expression Builder. The expression is evaluated once before the child algorithm is executed and used during the execution of that algorithm.
-  **Model Input**: allows to use an input added to the model as a parameter. Once clicked, this option will list all the suitable inputs for the parameter.
-  **Algorithm Output**: allows to use the output of another algorithm as an input of the current algorithm. As of model inputs, this option will list all the suitable inputs for the parameter.
- The **output parameter** also has the above options in its drop-down menu:
  - add static outputs for child algorithms, e.g. always saving a child algorithm's output to a predefined geopackage or postgres layer
  - use an expression based output values for child algorithms, e.g. generating an automatic file name based on today's date and saving outputs to that file
  - use a model input, e.g. the *File/Folder* model input to specify an output file or folder
  - use another algorithm output, e.g. the output of the *Create directory* algorithm (from *Modeler tools*)
  - an additional  **Model Output** option makes the output of the algorithm available in the model. If a layer generated by the algorithm is only to be used as input to another algorithm, don't edit that text box.

In the following picture you can see the two input parameters defined as *Model Input* and the temporary output layer:

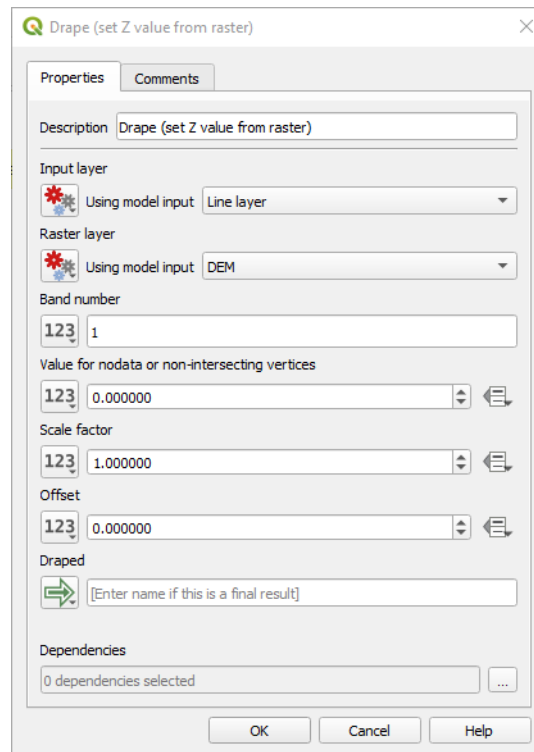



Abb. 27.23: Algorithm Input and Output parameters

You will also find an additional parameter named *Dependencies* that is not available when calling the algorithm from the toolbox. This parameter allows you to define the order in which algorithms are executed, by explicitly defining one algorithm as a *parent* of the current one. This will force the *parent* algorithm to be executed before the current one.

Wenn Sie die Ausgabe eines vorherigen Algorithmus als Eingabe Ihres Algorithmus verwenden, wird der vorherige Algorithmus implizit als übergeordneter Algorithmus des aktuellen festgelegt (und der entsprechende Pfeil im Modellierungsfenster platziert). In einigen Fällen kann ein Algorithmus jedoch von einem anderen Algorithmus abhängen, selbst wenn er kein Ausgabeobjekt von diesem verwendet (z.B. ein Algorithmus, der einen SQL-Satz auf einer PostGIS-Datenbank ausführt und ein anderer, der einen Layer in dieselbe Datenbank importiert). In diesem Fall wählen Sie einfach den vorherigen Algorithmus im Parameter *Dependencies* aus und sie werden in der richtigen Reihenfolge ausgeführt.

Sobald allen Parametern gültige Werte zugewiesen wurden, klicken Sie auf *OK* und der Algorithmus wird der Oberfläche hinzugefügt. Er wird mit den Elementen auf der Oberfläche (Algorithmen oder Eingaben) verknüpft, die Objekte liefern, die als Eingaben für den Algorithmus verwendet werden.

Elements can be dragged to a different position on the canvas using the  *Select/Move Item* tool. This is useful to make the structure of the model clearer and more intuitive. You can also resize the elements, grasping their border. This is particularly useful if the description of the input or algorithm is long. With *View ► Enable snapping* option checked, items resizing or displacement can be bound to a virtual grid, for a more visually structured algorithm design.

Links between elements are updated automatically and you can see a + button at the top and at the bottom of each algorithm. Clicking the button will list all the inputs and outputs of the algorithm so you can have a quick overview.

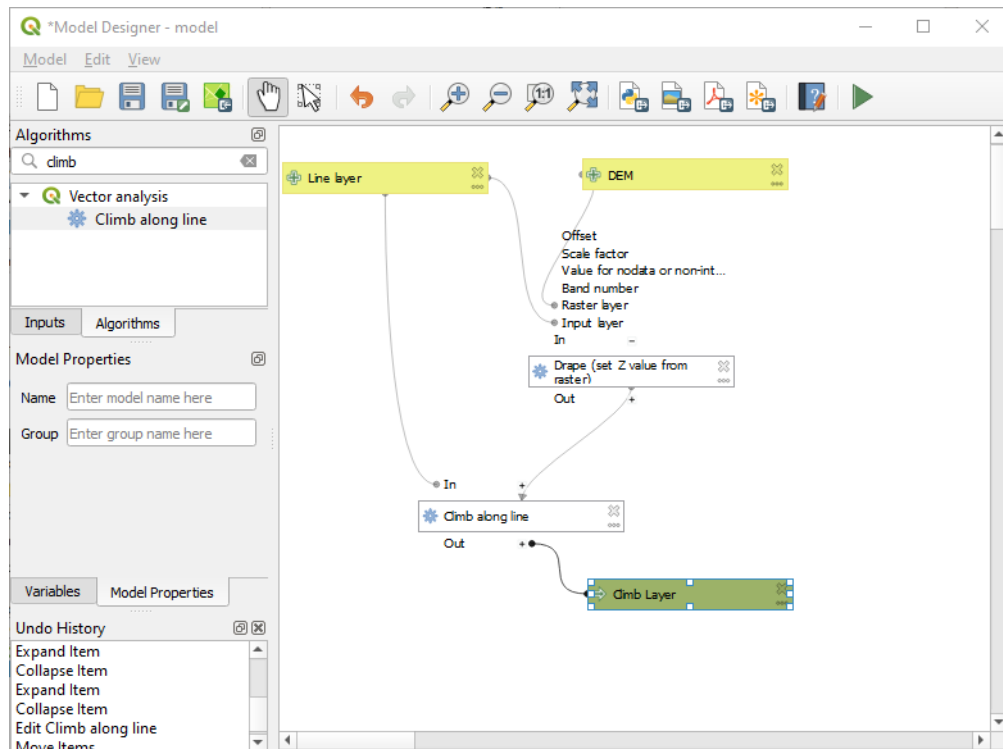


Abb. 27.24: Ein komplettes Modell

With the *Edit ► Add Group Box* tool, you can add a draggable *box* to the canvas. This feature is very useful in big models to group related elements in the modeler canvas and to keep the workflow clean. For example we might group together all the inputs of the example:

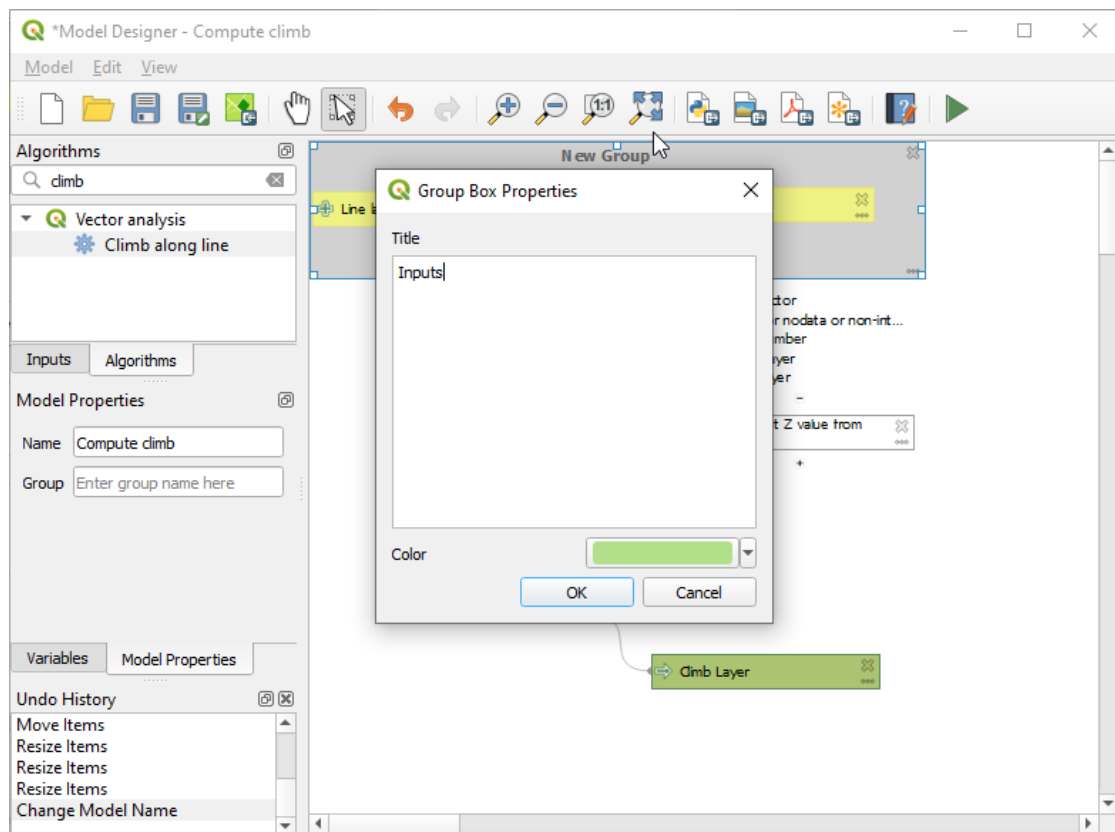


Abb. 27.25: Model Group Box

You can change the name and the color of the boxes. Group boxes are very useful when used together with *View ► Zoom To ►* tool, allowing you to zoom to a specific part of the model. You can also zoom in and out by using the mouse wheel.

You might want to change the order of the inputs and how they are listed in the main model dialog. At the bottom of the *Input* panel you will find the *Reorder Model Inputs . . .* button and by clicking on it a new dialog pops up allowing you to change the order of the inputs:

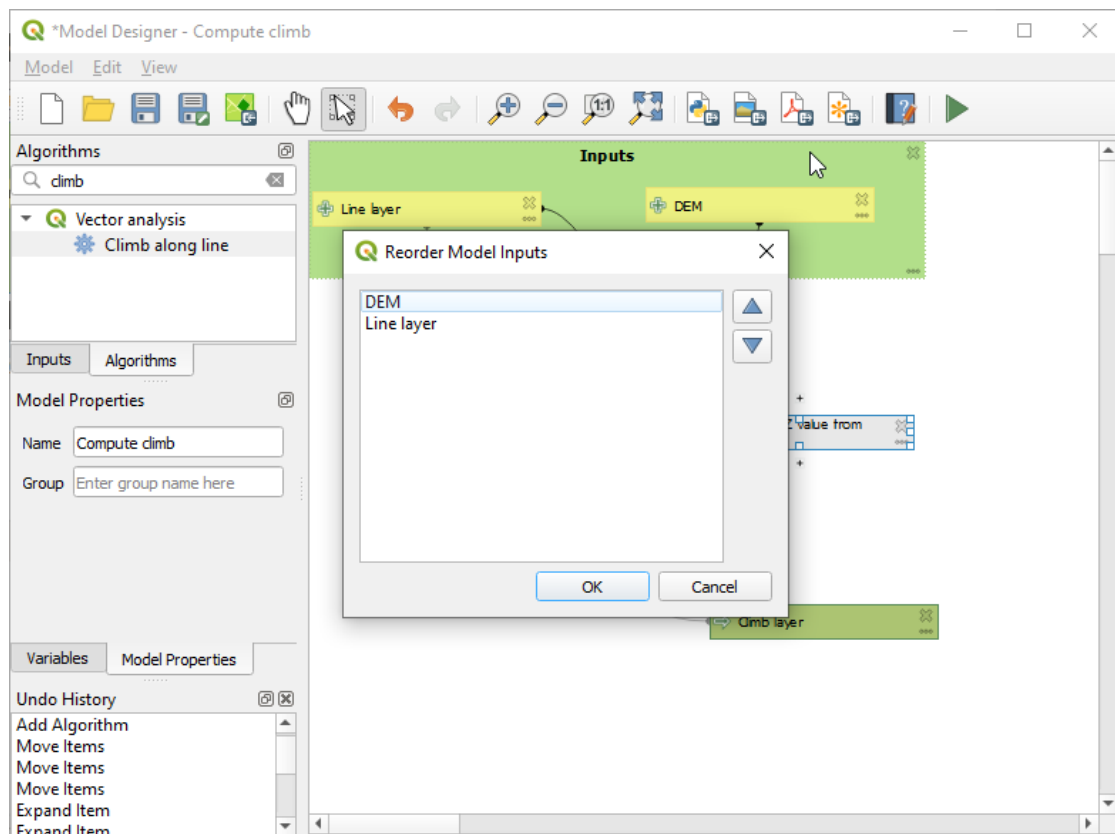


Abb. 27.26: Reorder Model Inputs

There is also the possibility to set a specific order which the outputs from the model must use when loading the results into a project. This gives the model creator a means of ensuring that layers are logically ordered on the canvas when running a model, such as placing a vector layer output over a raster layer output, or a point layer over a polygon layer. The model creator can also set an optional „Group name“ for the outputs for automatically grouping outputs within the layer tree using a new group name or by adding them to an existing group. In the **Model** menu you will find the **Reorder Output Layers...** entry and by clicking on it a new dialog pops up allowing you to change the order of the output layers:



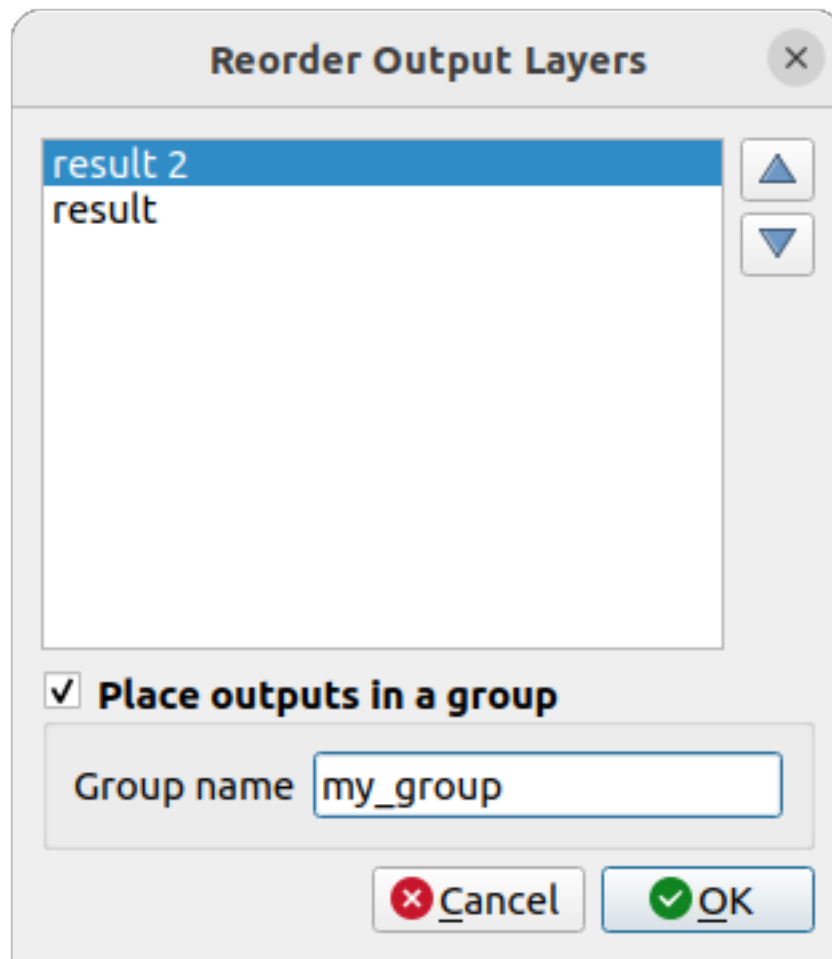




Abb. 27.27: Reorder Output Layers

Comments can also be added to inputs or algorithms present in the modeler. This can be done by going in the *Comment* tab of the item or with a right-click. In the same tab a color can be set manual for individual model comments. Comments are visible only in the modeler canvas and not in the final algorithm dialog; they can be hidden by deactivating *View ► Show Comments*.

You can run your algorithm any time by clicking on the  **Run model** button. When using the editor to execute a model, any non-default values will be saved in the inputs. This means that executing the model at a later time from the editor will have the dialog prefilled with those values on any subsequent run.

In order to use the algorithm from the toolbox, it has to be saved and the modeler dialog closed, to allow the toolbox to refresh its contents.

## Documenting your model

You need to document your model, and this can be done from the modeler itself. Click on the  button, and a dialog like the one shown next will appear.

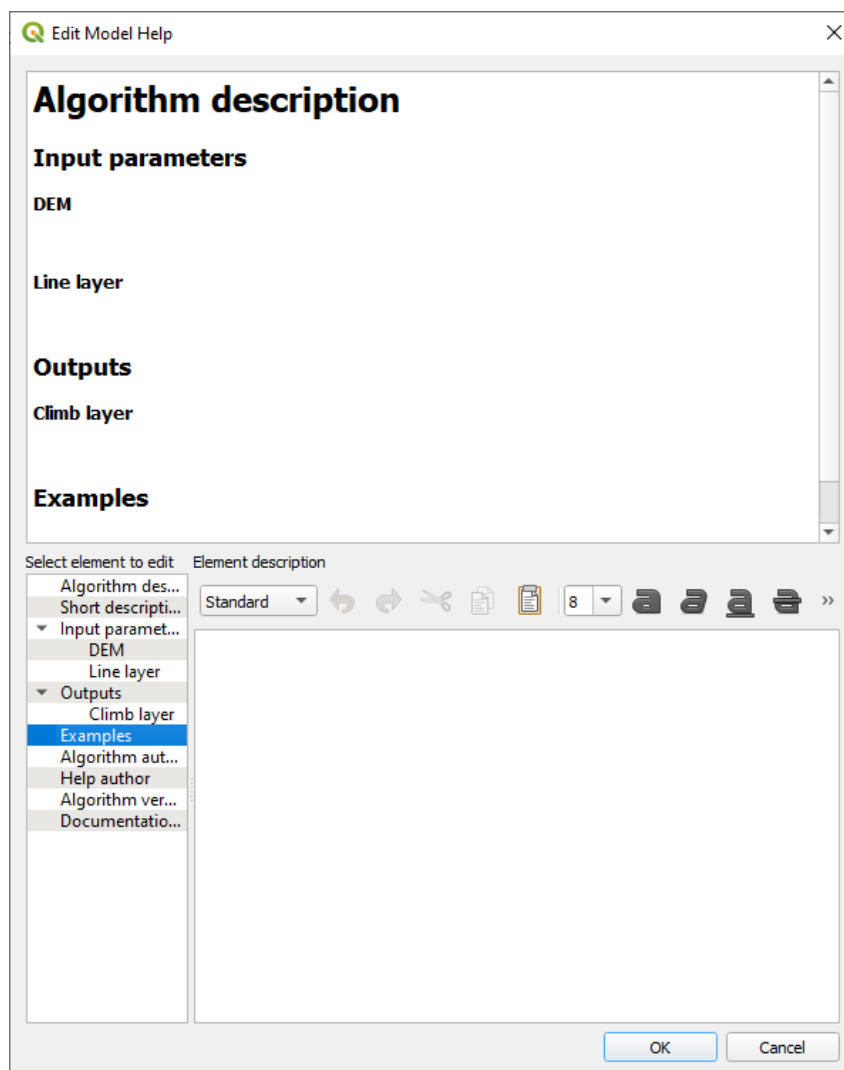




Abb. 27.28: Hilfe bearbeiten

On the right-hand side, you will see a simple HTML page, created using the description of the input parameters and outputs of the algorithm, along with some additional items like a general description of the model or its author. Also, there is an Example section where you can input your own custom examples to help explain the usage of the model. The first time you open the help editor, all these descriptions are empty, but you can edit them using the elements on the left-hand side of the dialog. Select an element on the upper part and then write its description in the text box below.

Modell-Hilfe wird als Teil des Modells selbst gespeichert.


## 27.5.3 Speichern und laden von Modellen


### Saving models

Use the  Save model button to save the current model and the  Open Model button to open a previously saved model. Models are saved with the `.model3` extension. If the model has already been saved from the modeler window, you will not be prompted for a filename. Since there is already a file associated with the model, that file will be used for subsequent saves.

Bevor Sie ein Modell speichern, müssen Sie in den Textfeldern im oberen Teil des Fensters einen Namen und eine Gruppe für das Modell eingeben.

Modelle, die im Ordner `models` gespeichert sind (der Standardordner, wenn Sie zur Eingabe eines Dateinamens zum Speichern des Modells aufgefordert werden), erscheinen in der Toolbox in der entsprechenden Rubrik. Wenn die Toolbox aufgerufen wird, durchsucht sie den Ordner `models` nach Dateien mit der Erweiterung `.model3` und lädt die darin enthaltenen Modelle. Da ein Modell selbst ein Algorithmus ist, kann es wie jeder andere Algorithmus zur Toolbox hinzugefügt werden.

Modelle können auch innerhalb der Projektdatei mit der Schaltfläche  Modell im Projekt speichern gespeichert werden. Modelle, die mit dieser Methode gespeichert werden, werden nicht als `:file:~\model3` Dateien auf die Platte geschrieben, sondern in die Projektdatei eingebettet.

Project models are available in the  *Project models* menu of the toolbox and in the *Project ► Models* menu item.



Die Verarbeitungsmodellierung kann im Verarbeitungskonfigurationsdialog in der *Models* Gruppe eingestellt werden.

Modelle, die aus dem Ordner `models` geladen wurden, erscheinen nicht nur in der Toolbox, sondern auch in der Algorithmenliste im Reiter *Algorithms* des Modellierfensters. Das bedeutet, dass Sie ein Modell als Teil eines größeren Modells einbinden können, genau wie andere Algorithmen.




Die Modelle werden im Panel *Browser* angezeigt und können von dort aus gestartet werden.

### Exportieren eines Modells als Python Skript

As we will see in a later chapter, Processing algorithms can be called from the QGIS Python console, and new Processing algorithms can be created using Python. A quick way to create such a Python script is to create a model and then export it as a Python file.

To do so, click on the  Export as Script Algorithm... in the modeler canvas or right click on the name of the model in the Processing Toolbox and choose  Export Model as Python Algorithm...

### Exportieren eines Modells als Bild, PDF oder SVG

A model can also be exported as an image, SVG or PDF (for illustration purposes) by clicking  Export as image,  Export as PDF or  Export as SVG.

## 27.5.4 Ein Modell editieren

Sie können das Modell, das Sie gerade erstellen, bearbeiten und dabei den Workflow und die Beziehungen zwischen den Algorithmen und Eingaben, die das Modell definieren, ändern.

Wenn Sie mit der rechten Maustaste auf einen Algorithmus in der Leinwand klicken, sehen Sie ein Kontextmenü wie das folgende:

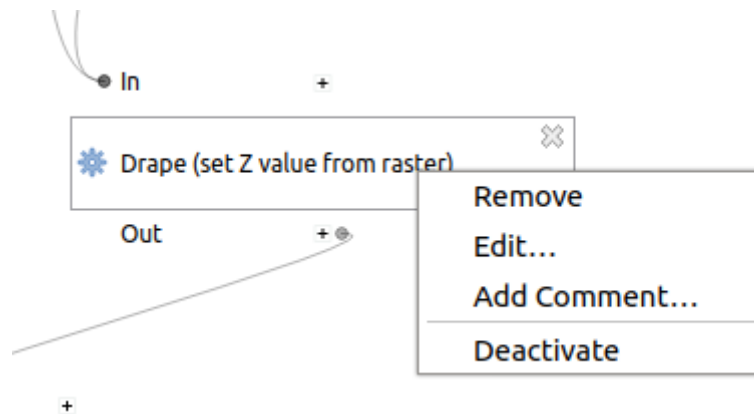


Abb. 27.29: Rechtsklick auf den Modeler

Wird die *Remove* Option gewählt, bewirkt dies, dass der ausgewählte Algorithmus entfernt wird. Ein Algorithmus kann nur entfernt werden wenn keine weiteren Algorithmen von ihm abhängen. Das heißt, wenn keine Ausgabe des Algorithmus in einem anderen als Eingabe verwendet wird. Wenn Sie versuchen einen Algorithmus, von dem andere abhängen, zu entfernen, wird eine Warnmeldung wie die, die Sie unten sehen können, gezeigt:

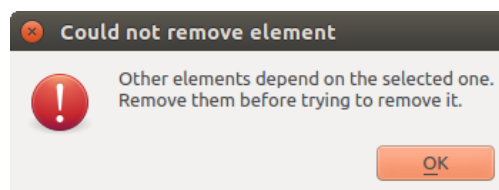


Abb. 27.30: Kann Algorithmus nicht entfernen

Wenn Sie die Option `:guilabel: Bearbeiten` wählen, wird der Parameterdialog des Algorithmus angezeigt, so dass Sie die Eingaben und Parameterwerte ändern können. Es werden nicht alle im Modell verfügbaren Eingabeelemente als verfügbare Eingaben angezeigt. Layer oder Werte, die bei einem weiter entwickelten Schritt des Arbeitsablaufes erzeugt wurden, sind nicht verfügbar, wenn sie zirkuläre Abhängigkeiten verursachen.

Wählen Sie die neuen Werte aus und klicken Sie wie gewohnt auf die Schaltfläche *OK*. Die Verbindungen zwischen den Modellelementen ändern sich in der Modellierungsarbeitsfläche entsprechend.

The *Add comment...* allows you to add a comment to the algorithm to better describe the behavior.

A model can be run partially by deactivating some of its algorithms. To do it, select the *Deactivate* option in the context menu that appears when right-clicking on an algorithm element. The selected algorithm, and all the ones in the model that depend on it will be displayed in grey and will not be executed as part of the model.

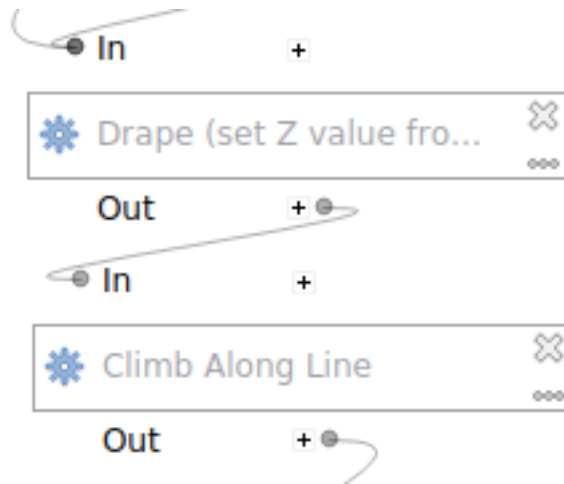


Abb. 27.31: Modell mit deaktivierten Algorithmen

Wenn Sie mit der rechten Maustaste auf einen Algorithmus klicken, der nicht aktiv ist, sehen Sie die Menüoption *Aktivieren*, mit der Sie den Algorithmus reaktivieren können.

## 27.6 Die Batch Processing Schnittstelle

### 27.6.1 Einführung

All algorithms (including models) can be executed as a batch process. That is, they can be executed using not just a single set of inputs, but several of them, executing the algorithm as many times as needed. This is useful when processing large amounts of data, since it is not necessary to launch the algorithm many times from the toolbox.

Um einen Algorithmus als Batch-Prozess ausführen, klicken Sie mit der rechten Maustaste auf seinen Namen in der Toolbox und wählen Sie die Option *Ausführen als Batch-Prozess*.

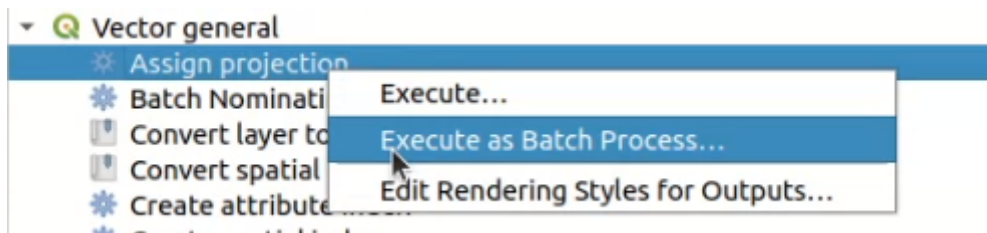


Abb. 27.32: Batch Processing from right-click

Wenn Sie den Ausführungsdialog des Algorithmus geöffnet haben, können Sie die Stapelverarbeitung auch von dort starten, drücken Sie auf den *Als Batch-Prozess ausführen...* Knopf.

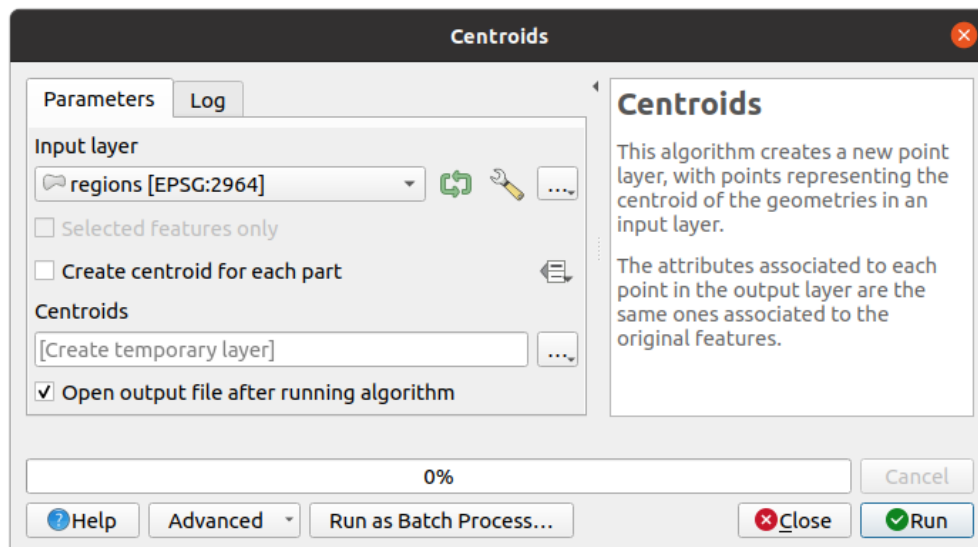


Abb. 27.33: Batch-Prozess vom Algorithmus Dialog

## 27.6.2 Die Parameter-Tabelle

Executing a batch process is similar to performing a single execution of an algorithm. Parameter values have to be defined, but in this case we need not just a single value for each parameter, but a set of them instead, one for each time the algorithm has to be executed. Values are introduced using a table like the one shown next, where each row is an iteration and columns are the parameters of the algorithm.

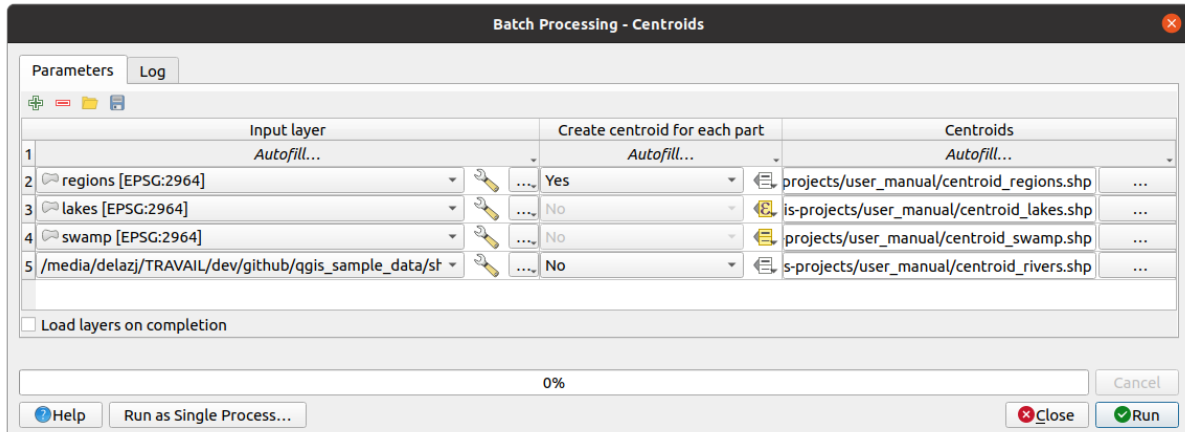



Abb. 27.34: Batch Processing

From the top toolbar, you can:

- Add row: adds a new processing entry for configuration
- Remove row(s): remove selected rows from the table. Row selection is done by clicking the number at the left and allows *keyboard combination* for multi selection.
- Open a batch processing configuration file
- Save the batch processing configuration to a .JSON file that can be run afterwards

By default, the table contains just two rows:

- The first row displays in each cell an *Autofill...* ► drop-down menu with *options* to quickly fill the cells below. Available options depend on the parameter type.
- The second row (as well as each subsequent one) represents a single execution of the algorithm, and each cell contains the value of one of the parameters. It is similar to the parameters dialog that you see when executing an algorithm from the toolbox, but with a different arrangement.


At the bottom of the table, you can set whether to  *Load layers on completion*.

Sobald die Größe der Tabelle gesetzt ist, muss sie mit den gewünschten Werten gefüllt werden.


### 27.6.3 Füllen der Parameter-Tabelle

For most parameters, setting the value is trivial. The appropriate widget, same as in the *single process dialog*, is provided, allowing to just type the value, or select it from a list of possible values, depending on the parameter type. This also includes data-define widget, when compatible.


To automate the batch process definition and avoid filling the table cell by cell, you may want to press down the *Autofill...* menu of a parameter and select any of the following options to replace values in the column:

- *Fill Down* will take the input for the first process and enter it for all other processes.
-  *Calculate by Expression...* will allow you to create a new QGIS expression to use to update all existing values within that column. Existing parameter values (including those from other columns) are available for use inside the expression via *variables*. E.g. setting the number of segments based on the buffer distance of each layer:

```
CASE WHEN @DISTANCE > 20 THEN 12 ELSE 8 END
```

- *Add Values by Expression...* will add new rows using the values from an expression which returns an array (as opposed to *Calculate by Expression...*, which works only on existing rows). The intended use case is to allow populating the batch dialog using complex numeric series. For example adding rows for a batch buffer using the expression `generate_series(100, 1000, 50)` for distance parameter results in new rows with values 100, 150, 200, .... 1000.
- When setting a file or layer parameter, more options are provided:
  - *Add Files by Pattern...*: adds new rows to the table for files matching a *File pattern* in a folder to *Look in*. E.g. \*.shp will add to the list all the SHP files in the folder. Check  *Search recursively* to also browse sub-folders.
  - *Select Files...* individually on disk
  - *Add All Files from a Directory...*
  - *Select from Open Layers...* in the active project

Output data parameter exposes the same capabilities as when executing the algorithm as a single process. Depending on the algorithm, the output can be:

- skipped, if the cell is left empty
- saved as a temporary layer: fill the cell with `TEMPORARY_OUTPUT` and remember to tick the  *Load layers on completion* checkbox.
- saved as a plain file (.SHP, .GPKG, .XML, .PDF, .JPG,...) whose path could be set with the *Autofill* options exposed beforehand. E.g. use *Calculate by Expression...* to set output file names to complex expressions like:

```
'/home/me/stuff/buffer_' || left(@INPUT, 30) || '_' || @DISTANCE || '.shp'
```

You can also type the file path directly or use the file chooser dialog that appears when clicking on the accompanying ... button. Once you select the file, a new dialog is shown to allow for auto-completion of other cells in the same column (same parameter).

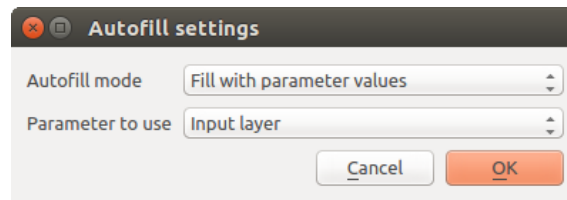


Abb. 27.35: Batch Processing Save

If the default value (*Do not autofill*) is selected, it will just put the selected filename in the selected cell from the parameters table. If any of the other options is selected, all the cells **below** the selected one will be automatically filled based on a defined criteria:

- *Fill with numbers*: incrementally appends a number to the file name
- *Fill with parameter values*: you can select a parameter whose value in the same row is appended to the file name. This is particularly useful for naming output data objects according to input ones.
- saved as a layer within a database container:

```
# Indicate a layer within a GeoPackage file
ogr:dbname='C:/Path/To/Geopackage.gpkg' table="New_Table" (geom)

# Use the "Calculate By Expression" to output to different layers in a
↳ GeoPackage
'ogr:dbname=\' | | @project_folder | | '/Buffers.gpkg\' table=\' | | @INPUT | | \'_
↳ \' | | @DISTANCE | | \' (geom)'
```

## 27.6.4 Ausführen eines Batch-Prozesses

To execute the batch process once you have introduced all the necessary values, just click on *Run*. The *Log* panel is activated and displays details and steps of the execution process. Progress of the global batch task will be shown in the progress bar in the lower part of the dialog.

## 27.7 Verarbeitung Algorithmen von der Konsole aus verwenden

Die Konsole ermöglicht es fortgeschrittenen Anwendern ihre Produktivität zu erhöhen und komplexe Operationen, die nicht anhand eines der GUI Elemente der Verarbeitung Umgebung ausgeführt werden können, durchzuführen. Modelle mit mehreren Algorithmen können anhand der Kommandozeilenschnittstelle definiert werden und zusätzliche Operationen wie Schleifen und Bedingungssätze können hinzugefügt werden, um flexiblere und leistungsfähigere Workflows zu erstellen.

There is not a processing console in QGIS, but all processing commands are available instead from the QGIS built-in *Python console*. That means that you can incorporate those commands into your console work and connect processing algorithms to all the other features (including methods from the QGIS API) available from there.

Der Code den Sie von der Python Konsole aus ausführen können, auch wenn er keine spezifische Verarbeitung Methode aufruft, kann in einen neuen Algorithmus überführt werden den Sie später aus der Werkzeugkiste, der Grafischen Modellierung oder jeder anderen Komponente aufrufen können, so wie Sie es mit jedem anderen Algorithmus tun würden. In der Tat sind einige Algorithmen, die Sie in der Werkzeugkiste finden können einfache Skripte.

In diesem Abschnitt werden wir uns angucken wie man Verarbeitung Algorithmen aus der QGIS Python Konsole heraus verwendet und auch wie man Algorithmen in Python schreibt.



### 27.7.1 Algorithmen von der Python Konsole aus aufrufen

Das erste, was Sie machen müssen, ist die Verarbeitung Funktionen mit der folgenden Zeile importieren:

```
>>> from qgis import processing
```

Now, there is basically just one (interesting) thing you can do with that from the console: execute an algorithm. That is done using the `run()` method, which takes the name of the algorithm to execute as its first parameter, and then a variable number of additional parameters depending on the requirements of the algorithm. So the first thing you need to know is the name of the algorithm to execute. That is not the name you see in the toolbox, but rather a unique command-line name. To find the right name for your algorithm, you can use the `processingRegistry`. Type the following line in your console:

```
>>> for alg in QgsApplication.processingRegistry().algorithms():
    print(alg.id(), "->", alg.displayName())
```

You will see something like this (with some extra dashes added to improve readability).

```
3d:tessellate -----> Tessellate
gdal:aspect -----> Aspect
gdal:assignprojection -----> Assign projection
gdal:bufferectors -----> Buffer vectors
gdal:buildvirtualraster ----> Build Virtual Raster
gdal:cliprasterbyextent ----> Clip raster by extent
gdal:cliprasterbymasklayer -> Clip raster by mask layer
gdal:clipvectorbyextent ----> Clip vector by extent
gdal:clipvectorbypolygon ---> Clip vector by mask layer
gdal:colorrelief -----> Color relief
gdal:contour -----> Contour
gdal:convertformat -----> Convert format
gdal:dissolve -----> Dissolve
...
```

That's a list of all the available algorithm IDs, sorted by provider name and algorithm name, along with their corresponding names.

Once you know the command-line name of the algorithm, the next thing to do is to determine the right syntax to execute it. That means knowing which parameters are needed when calling the `run()` method.

There is a method to describe an algorithm in detail, which can be used to get a list of the parameters that an algorithm requires and the outputs that it will generate. To get this information, you can use the `algorithmHelp(id_of_the_algorithm)` method. Use the ID of the algorithm, not the full descriptive name.

Calling the method with `native:buffer` as parameter (`qgis:buffer` is an alias for `native:buffer` and will also work), you get the following description:

```
>>> processing.algorithmHelp("native:buffer")
Buffer (native:buffer)

This algorithm computes a buffer area for all the features in an
input layer, using a fixed or dynamic distance.

The segments parameter controls the number of line segments to
use to approximate a quarter circle when creating rounded
offsets.

The end cap style parameter controls how line endings are handled
in the buffer.

The join style parameter specifies whether round, miter or
beveled joins should be used when offsetting corners in a line.
```

(Fortsetzung auf der nächsten Seite)

The miter limit parameter is only applicable for miter join styles, and controls the maximum distance from the offset curve to use when creating a mitered join.

-----  
Input parameters  
-----

INPUT: Input layer

Parameter type: QgsProcessingParameterFeatureSource

Accepted data types:

- str: layer ID
- str: layer name
- str: layer source
- QgsProcessingFeatureSourceDefinition
- QgsProperty
- QgsVectorLayer

DISTANCE: Distance

Parameter type: QgsProcessingParameterDistance

Accepted data types:

- int
- float
- QgsProperty

SEGMENTS: Segments

Parameter type: QgsProcessingParameterNumber

Accepted data types:

- int
- float
- QgsProperty

END\_CAP\_STYLE: End cap style

Parameter type: QgsProcessingParameterEnum

Available values:

- 0: Round
- 1: Flat
- 2: Square

Accepted data types:

- int
- str: as string representation of int, e.g. '1'
- QgsProperty

JOIN\_STYLE: Join style

Parameter type: QgsProcessingParameterEnum

Available values:

- 0: Round
- 1: Miter

(Fortsetzung der vorherigen Seite)

```

- 2: Bevel

Accepted data types:
- int
- str: as string representation of int, e.g. '1'
- QgsProperty

MITER_LIMIT: Miter limit

Parameter type: QgsProcessingParameterNumber

Accepted data types:
- int
- float
- QgsProperty

DISSOLVE: Dissolve result

Parameter type: QgsProcessingParameterBoolean

Accepted data types:
- bool
- int
- str
- QgsProperty

OUTPUT: Buffered

Parameter type: QgsProcessingParameterFeatureSink

Accepted data types:
- str: destination vector file, e.g. 'd:/test.shp'
- str: 'memory:' to store result in temporary memory layer
- str: using vector provider ID prefix and destination URI,
      e.g. 'postgres:...' to store result in PostGIS table
- QgsProcessingOutputLayerDefinition
- QgsProperty

-----
Outputs
-----

OUTPUT: <QgsProcessingOutputVectorLayer>
      Buffered

```

Now you have everything you need to run any algorithm. As we have already mentioned, algorithms can be run using: `run()`. Its syntax is as follows:

```
>>> processing.run(name_of_the_algorithm, parameters)
```

Where `parameters` is a dictionary of parameters that depend on the algorithm you want to run, and is exactly the list that the `algorithmHelp()` method gives you.

```

1 >>> processing.run("native:buffer", {'INPUT': '/data/lines.shp',
2     'DISTANCE': 100.0,
3     'SEGMENTS': 10,
4     'DISSOLVE': True,
5     'END_CAP_STYLE': 0,
6     'JOIN_STYLE': 0,
7     'MITER_LIMIT': 10,
8     'OUTPUT': '/data/buffers.shp'})

```

If a parameter is optional and you do not want to use it, then don't include it in the dictionary.

If a parameter is not specified, the default value will be used.

Abhängig vom Parametertyp werden Werte verschieden eingeführt. Die nächste Liste gibt einen kurzen Überblick darüber wie man Werte für jeden Typ von Eingabeparameter einführt:

- Raster Layer, Vector Layer or Table. Simply use a string with the name that identifies the data object to use (the name it has in the QGIS Table of Contents) or a filename (if the corresponding layer is not opened, it will be opened but not added to the map canvas). If you have an instance of a QGIS object representing the layer, you can also pass it as parameter.
- Enumeration. If an algorithm has an enumeration parameter, the value of that parameter should be entered using an integer value. To know the available options, you can use the `algorithmHelp()` command, as above. For instance, the `native:buffer` algorithm has an enumeration called `JOIN_STYLE`:

```
JOIN_STYLE: Join style

Parameter type: QgsProcessingParameterEnum

Available values:
- 0: Round
- 1: Miter
- 2: Bevel

Accepted data types:
- int
- str: as string representation of int, e.g. '1'
- QgsProperty
```

In this case, the parameter has three options. Notice that ordering is zero-based.

- Boolean. Use `True` or `False`.
- Multiple input. Der Wert ist ein String mit Eingabebeschreibungen getrennt durch Semikolons (;). Wie im Fall von einfachen Layern oder Tabellen kann jede Eingabebeschreibung der Datenobjektname oder sein Dateipfad sein.
- Tabellen Feld von XXX. Verwenden Sie einen String mit dem Namen des Feldes, das benutzt werden soll. Dieser Parameter unterscheidet zwischen Groß- und Kleinschreibung.
- Fixed Table. Geben Sie die Liste aller Tabellenwerte, die durch Kommas (,) getrennt sind und zwischen Anführungsstrichen (") eingeschlossen sind. Die Werte beginnen in der oberen Zeile und gehen von links nach rechts. Sie können auch einen 2D-Array von Werten, die die Tabelle repräsentieren, verwenden.
- CRS. Geben Sie den EPSG Code des gewünschten KBS ein.
- Extent. Sie müssen einen String mit `xmin`, `xmax`, `ymin` und `ymax` Werten getrennt durch Kommas (,) eingeben.

Boolean, file, string und numerical parameters brauchen keien zusätzlichen Erläuterungen.

Input parameters such as strings, booleans, or numerical values have default values. The default value is used if the corresponding parameter entry is missing.

For output data objects, type the file path to be used to save it, just as it is done from the toolbox. If the output object is not specified, the result is saved to a temporary file (or skipped if it is an optional output). The extension of the file determines the file format. If you enter a file extension not supported by the algorithm, the default file format for that output type will be used, and its corresponding extension appended to the given file path.

Unlike when an algorithm is executed from the toolbox, outputs are not added to the map canvas if you execute that same algorithm from the Python console using `run()`, but `runAndLoadResults()` will do that.

The `run()` method returns a dictionary with one or more output names (the ones shown in the algorithm description) as keys and the file paths of those outputs as values:

```

1 >>> myresult = processing.run("native:buffer", {'INPUT': '/data/lines.shp',
2         'DISTANCE': 100.0,
3         'SEGMENTS': 10,
4         'DISSOLVE': True,
5         'END_CAP_STYLE': 0,
6         'JOIN_STYLE': 0,
7         'MITER_LIMIT': 10,
8         'OUTPUT': '/data/buffers.shp'})
9 >>> myresult['OUTPUT']
10 /data/buffers.shp

```

You can load feature output by passing the corresponding file paths to the `load()` method. Or you could use `runAndLoadResults()` instead of `run()` to load them immediately.

If you want to open an algorithm dialog from the console you can use the `createAlgorithmDialog` method. The only mandatory parameter is the algorithm name, but you can also define the dictionary of parameters so that the dialog will be filled automatically:

```

1 >>> my_dialog = processing.createAlgorithmDialog("native:buffer", {
2         'INPUT': '/data/lines.shp',
3         'DISTANCE': 100.0,
4         'SEGMENTS': 10,
5         'DISSOLVE': True,
6         'END_CAP_STYLE': 0,
7         'JOIN_STYLE': 0,
8         'MITER_LIMIT': 10,
9         'OUTPUT': '/data/buffers.shp'})
10 >>> my_dialog.show()

```

The `execAlgorithmDialog` method opens the dialog immediately:

```

1 >>> processing.execAlgorithmDialog("native:buffer", {
2         'INPUT': '/data/lines.shp',
3         'DISTANCE': 100.0,
4         'SEGMENTS': 10,
5         'DISSOLVE': True,
6         'END_CAP_STYLE': 0,
7         'JOIN_STYLE': 0,
8         'MITER_LIMIT': 10,
9         'OUTPUT': '/data/buffers.shp'})

```

## 27.7.2 Skripte erstellen und diese aus der Werkzeugkiste starten.

You can create your own algorithms by writing Python code. Processing scripts extend `QgsProcessingAlgorithm`, so you need to add some extra lines of code to implement mandatory functions. You can find *Create new script* (clean sheet) and *Create New Script from Template* (template that includes code for mandatory functions of `QgsProcessingAlgorithm`) under the *Scripts* dropdown menu on the top of the Processing toolbox. The Processing Script Editor will open, and that's where you should type your code. Saving the script from there in the `scripts` folder (the default folder when you open the save file dialog) with a `.py` extension should create the corresponding algorithm.

The name of the algorithm (the one you will see in the toolbox) is defined within the code.

Let's have a look at the following code, which defines a Processing algorithm that performs a buffer operation with a user defined buffer distance on a vector layer that is specified by the user, after first smoothing the layer.

```

1 from qgis.core import (QgsProcessingAlgorithm,
2     QgsProcessingParameterNumber,
3     QgsProcessingParameterFeatureSource,
4     QgsProcessingParameterFeatureSink)

```

(Fortsetzung auf der nächsten Seite)

```

5
6 from qgis import processing
7
8 class algTest(QgsProcessingAlgorithm):
9     INPUT_BUFFERDIST = 'BUFFERDIST'
10    OUTPUT_BUFFER = 'OUTPUT_BUFFER'
11    INPUT_VECTOR = 'INPUT_VECTOR'
12
13    def __init__(self):
14        super().__init__()
15
16    def name(self):
17        return "algTest"
18
19    def displayName(self):
20        return "algTest script"
21
22    def createInstance(self):
23        return type(self)()
24
25    def initAlgorithm(self, config=None):
26        self.addParameter(QgsProcessingParameterFeatureSource(
27            self.INPUT_VECTOR, "Input vector"))
28        self.addParameter(QgsProcessingParameterNumber(
29            self.INPUT_BUFFERDIST, "Buffer distance",
30            QgsProcessingParameterNumber.Double,
31            100.0))
32        self.addParameter(QgsProcessingParameterFeatureSink(
33            self.OUTPUT_BUFFER, "Output buffer"))
34
35    def processAlgorithm(self, parameters, context, feedback):
36        #DO SOMETHING
37        algresult = processing.run("native:smoothgeometry",
38            {'INPUT': parameters[self.INPUT_VECTOR],
39             'ITERATIONS':2,
40             'OFFSET':0.25,
41             'MAX_ANGLE':180,
42             'OUTPUT': 'memory:'},
43            context=context, feedback=feedback, is_child_algorithm=True)
44        smoothed = algresult['OUTPUT']
45        algresult = processing.run('native:buffer',
46            {'INPUT': smoothed,
47             'DISTANCE': parameters[self.INPUT_BUFFERDIST],
48             'SEGMENTS': 5,
49             'END_CAP_STYLE': 0,
50             'JOIN_STYLE': 0,
51             'MITER_LIMIT': 10,
52             'DISSOLVE': True,
53             'OUTPUT': parameters[self.OUTPUT_BUFFER]}},
54            context=context, feedback=feedback, is_child_algorithm=True)
55        buffered = algresult['OUTPUT']
56        return {self.OUTPUT_BUFFER: buffered}

```

After doing the necessary imports, the following `QgsProcessingAlgorithm` functions are specified:

- `name()`: The id of the algorithm (lowercase).
- `displayName()`: A human readable name for the algorithm.
- `createInstance()`: Create a new instance of the algorithm class.
- `initAlgorithm()`: Configure the parameterDefinitions and outputDefinitions.

Here you describe the parameters and output of the algorithm. In this case, a feature source for the input, a

feature sink for the result and a number for the buffer distance.

- `processAlgorithm()` : Do the work.

Here we first run the `smoothgeometry` algorithm to smooth the geometry, and then we run the `buffer` algorithm on the smoothed output. To be able to run algorithms from within another algorithm we have to set the `is_child_algorithm` argument to `True`. You can see how input and output parameters are used as parameters to the `smoothgeometry` and `buffer` algorithms.

There are a number of different parameter types available for input and output. Below is an alphabetically sorted list:

Tab. 27.2: List of input and output parameters

<code>QgsProcessingParameterAggregate</code>	<code>QgsProcessingParameterAnnotationLayer</code>	<code>QgsProcessingParameterAnnotationLayer</code>
<code>QgsProcessingParameterBoolean</code>	<code>QgsProcessingParameterColor</code>	<code>QgsProcessingParameterColor</code>
<code>QgsProcessingParameterDatabaseSchema</code>	<code>QgsProcessingParameterDatabaseTable</code>	<code>QgsProcessingParameterDatabaseTable</code>
<code>QgsProcessingParameterEnum</code>	<code>QgsProcessingParameterExpression</code>	<code>QgsProcessingParameterExpression</code>
<code>QgsProcessingParameterFeatureSource</code>	<code>QgsProcessingParameterField</code>	<code>QgsProcessingParameterField</code>
<code>QgsProcessingParameterFileDestination</code>	<code>QgsProcessingParameterFolderDestination</code>	<code>QgsProcessingParameterFolderDestination</code>
<code>QgsProcessingParameterLayoutItem</code>	<code>QgsProcessingParameterMapLayer</code>	<code>QgsProcessingParameterMapLayer</code>
<code>QgsProcessingParameterMeshLayer</code>	<code>QgsProcessingParameterMultipleLayers</code>	<code>QgsProcessingParameterMultipleLayers</code>
<code>QgsProcessingParameterPointCloudAttribute</code>	<code>QgsProcessingParameterPointCloudLayer</code>	<code>QgsProcessingParameterPointCloudLayer</code>
<code>QgsProcessingParameterRasterDestination</code>	<code>QgsProcessingParameterRasterLayer</code>	<code>QgsProcessingParameterRasterLayer</code>
<code>QgsProcessingParameterVectorDestination</code>	<code>QgsProcessingParameterVectorLayer</code>	<code>QgsProcessingParameterVectorLayer</code>

The first parameter to the constructors is the name of the parameter, and the second is the description of the parameter (for the user interface). The rest of the constructor parameters are parameter type specific.

The input can be turned into QGIS classes using the `parameterAs` functions of `QgsProcessingAlgorithm`. For instance to get the number provided for the buffer distance as a double:

```
self.parameterAsDouble(parameters, self.INPUT_BUFFERDIST, context)).
```

The `processAlgorithm` function should return a dictionary containing values for every output defined by the algorithm. This allows access to these outputs from other algorithms, including other algorithms contained within the same model.

Well behaved algorithms should define and return as many outputs as makes sense. Non-feature outputs, such as numbers and strings, are very useful when running your algorithm as part of a larger model, as these values can be used as input parameters for subsequent algorithms within the model. Consider adding numeric outputs for things like the number of features processed, the number of invalid features encountered, the number of features output, etc. The more outputs you return, the more useful your algorithm becomes!

## Feedback

The `feedback` object passed to `processAlgorithm()` should be used for user feedback / interaction. You can use the `setProgress()` function of the `feedback` object to update the progress bar (0 to 100) to inform the user about the progress of the algorithm. This is very useful if your algorithm takes a long time to complete.

The `feedback` object provides an `isCanceled()` method that should be monitored to enable cancelation of the algorithm by the user. The `pushInfo()` method of `feedback` can be used to send information to the user, and `reportError()` is handy for pushing non-fatal errors to users.

Algorithms should avoid using other forms of providing feedback to users, such as print statements or logging to `QgsMessageLog`, and should always use the feedback object instead. This allows verbose logging for the algorithm, and is also thread-safe (which is important, given that algorithms are typically run in a background thread).

## Handling errors

If your algorithm encounters an error which prevents it from executing, such as invalid input values or some other condition from which it cannot or should not recover, then you should raise a `QgsProcessingException`. E.g.:

```
if feature['value'] < 20:
    raise QgsProcessingException('Invalid input value {}, must be >= 20'.
    ↪format(feature['value']))
```

Try to avoid raising `QgsProcessingException` for non-fatal errors (e.g. when a feature has a null geometry), and instead just report these errors via `feedback.reportError()` and skip the feature. This helps make your algorithm „model-friendly“, as it avoids halting the execution of an entire algorithm when a non-fatal error is encountered.

## Ihre Skripte dokumentieren

As in the case of models, you can create additional documentation for your scripts, to explain what they do and how to use them.

`QgsProcessingAlgorithm` provides the `helpString()`, `shortHelpString()` and `helpUrl()` functions for that purpose. Specify / override these to provide more help to the user.

`shortDescription()` is used in the tooltip when hovering over the algorithm in the toolbox.

## 27.7.3 Pre- und Post-execution Script Hooks

Scripts can also be used as pre- and post-execution hooks that are run before and after an algorithm is run, respectively. This can be used to automate tasks that should be performed whenever an algorithm is executed.

Die Syntax ist identisch zu der oben erklärten Syntax, es steht aber eine zusätzliche Globalvariable genannt `alg` zur Verfügung, die den Algorithmus, der gerade ausgeführt wurde (oder ausgeführt wird), repräsentiert.

In the *General* group of the processing options dialog, you will find two entries named *Pre-execution script* and *Post-execution script* where the filenames of the scripts to be run in each case can be entered.

## 27.8 Using processing from the command line

QGIS comes with a tool called QGIS Processing Executor which allows you to run Processing algorithms and models (built-in or provided by plugins) directly from the command line without starting QGIS Desktop itself.

From a command line tool, run `qgis_process` and you should get:

```
QGIS Processing Executor - 3.27.0-Master 'Master' (3.27.0-Master)
Usage: C:\OSGeo4W\apps\qgis-dev\bin\qgis_process.exe [--help] [--version] [--json]
↪[--verbose] [--no-python] [command] [algorithm id, path to model file, or path
↪to Python script] [parameters]
```

Options:

```
--help or -h      Output the help
--version or -v   Output all versions related to QGIS Process
--json           Output results as JSON objects
--verbose        Output verbose logs
--no-python      Disable Python support (results in faster startup)
```

Available commands:

(Fortsetzung auf der nächsten Seite)



(Fortsetzung der vorherigen Seite)

```

plugins          list available and active plugins
plugins enable   enables an installed plugin. The plugin name must be specified,
↳ e.g. "plugins enable cartography_tools"
plugins disable  disables an installed plugin. The plugin name must be specified,
↳ e.g. "plugins disable cartography_tools"
list            list all available processing algorithms
help            show help for an algorithm. The algorithm id or a path to a
↳ model file must be specified.
run            runs an algorithm. The algorithm id or a path to a model file
↳ and parameter values must be specified.
               Parameter values are specified after -- with PARAMETER=VALUE
↳ syntax.
               Ordered list values for a parameter can be created by
↳ specifying the parameter multiple times,
               e.g. --LAYERS=layer1.shp --LAYERS=layer2.shp
               Alternatively, a '-' character in place of the parameters
↳ argument indicates that the parameters should be read from STDIN as a JSON
↳ object.
               The JSON should be structured as a map containing at least the
↳ "inputs" key specifying a map of input parameter values.
               This implies the --json option for output as a JSON object.
               If required, the ellipsoid to use for distance and area
↳ calculations can be specified via the "--ELLIPSOID=name" argument.
               If required, an existing QGIS project to use during the
↳ algorithm execution can be specified via the "--PROJECT_PATH=path" argument.

```

**Bemerkung:** Only installed plugins that advertise `hasProcessingProvider=yes` in their `metadata.txt` file are recognized and can be activated or loaded by `qgis_process` tool.

**Hinweis:** Before calling `qgis_process` on a system without window manager (e.g. a headless server), you should set:

```
export QT_QPA_PLATFORM=offscreen
```

The command `list` can be used to get a list of all available providers and algorithms.

```
qgis_process list
```

The command `help` can be used to get further information about commands or algorithms.

```
qgis_process help qgis:regularpoints
```

The command `run` can be used to run an algorithm or model. Specify the name of the algorithm or a path to a model as first parameter.

```
qgis_process run native:buffer -- INPUT=source.shp DISTANCE=2 OUTPUT=buffered.shp
```

Where a parameter accepts a list of values, set the same variable multiple times.

```
qgis_process run native:mergevectorlayers -- LAYERS=input1.shp LAYERS=input2.shp
↳ OUTPUT=merged.shp
```

While running an algorithm a text-based feedback bar is shown, and the operation can be cancelled via `CTRL+C`.

The `run` command also supports further parameters.

- `--json` will format stdout output in a JSON structured way.
- `--ellipsoid` will set the ellipsoid to the specified one.

- `--distance_units` will use the specified distance units.
- `--area_units` will use the specified area units.
- `--project_path` will load the specified project for running the algorithm.

Complex input parameters, i.e. parameter types which are themselves specified as a dictionary type object for algorithms, are supported by `qgis_process`. To indicate that parameters will be specified via stdin, the `qgis_process` command must follow the format (with a trailing `-` in place of the usual arguments list).

```
qgis_process run algorithmId -
```

The JSON object must contain an „inputs“ key, which is a map of the input parameter values. E.g.

```
echo '{"inputs": {"INPUT": "my_shape.shp", "DISTANCE": 5}}' | qgis_process run_
↪native:buffer -
```

Additionally, extra settings like the distance units, area units, ellipsoid and project path can be included in this JSON object:

```
{
  'ellipsoid': 'EPSG:7019',
  'distance_units': 'feet',
  'area_units': 'ha',
  'project_path': 'C:/temp/my_project.qgs'
  'inputs': {'DISTANCE': 5, 'SEGMENTS': 8 ... }
}
```

Specifying input parameters via stdin implies automatically the JSON output format for results.

## 27.9 Writing new Processing algorithms as Python scripts

There are two options for writing Processing algorithms using Python.

- *Extending `QgsProcessingAlgorithm`*
- *Using the `@alg decorator`*

Within QGIS, you can use *Create new script* in the *Scripts* menu at the top of the *Processing Toolbox* to open the *Processing Script Editor* where you can write your code. To simplify the task, you can start with a script template by using *Create new script from template* from the same menu. This opens a template that extends `QgsProcessingAlgorithm`.

If you save the script in the `scripts` folder (the default location) with a `.py` extension, the algorithm will become available in the *Processing Toolbox*.

### 27.9.1 Extending `QgsProcessingAlgorithm`

The following code

1. takes a vector layer as input
2. counts the number of features
3. does a buffer operation
4. creates a raster layer from the result of the buffer operation
5. returns the buffer layer, raster layer and number of features

```

1 from qgis.PyQt.QtCore import QApplication
2 from qgis.core import (QgsProcessing,
3                        QgsProcessingAlgorithm,
4                        QgsProcessingException,
5                        QgsProcessingOutputNumber,
6                        QgsProcessingParameterDistance,
7                        QgsProcessingParameterFeatureSource,
8                        QgsProcessingParameterVectorDestination,
9                        QgsProcessingParameterRasterDestination)
10 from qgis import processing
11
12
13 class ExampleProcessingAlgorithm(QgsProcessingAlgorithm):
14     """
15     This is an example algorithm that takes a vector layer,
16     creates some new layers and returns some results.
17     """
18
19     def tr(self, string):
20         """
21         Returns a translatable string with the self.tr() function.
22         """
23         return QApplication.translate('Processing', string)
24
25     def createInstance(self):
26         # Must return a new copy of your algorithm.
27         return ExampleProcessingAlgorithm()
28
29     def name(self):
30         """
31         Returns the unique algorithm name.
32         """
33         return 'bufferrasterextend'
34
35     def displayName(self):
36         """
37         Returns the translated algorithm name.
38         """
39         return self.tr('Buffer and export to raster (extend)')
40
41     def group(self):
42         """
43         Returns the name of the group this algorithm belongs to.
44         """
45         return self.tr('Example scripts')
46
47     def groupId(self):
48         """
49         Returns the unique ID of the group this algorithm belongs
50         to.
51         """
52         return 'examplescripts'
53
54     def shortHelpString(self):
55         """
56         Returns a localised short help string for the algorithm.
57         """
58         return self.tr('Example algorithm short description')
59
60     def initAlgorithm(self, config=None):
61         """
62         Here we define the inputs and outputs of the algorithm.

```

(Fortsetzung auf der nächsten Seite)

```

63     """
64     # 'INPUT' is the recommended name for the main input
65     # parameter.
66     self.addParameter(
67         QgsProcessingParameterFeatureSource(
68             'INPUT',
69             self.tr('Input vector layer'),
70             types=[QgsProcessing.TypeVectorAnyGeometry]
71         )
72     )
73     self.addParameter(
74         QgsProcessingParameterVectorDestination(
75             'BUFFER_OUTPUT',
76             self.tr('Buffer output'),
77         )
78     )
79     # 'OUTPUT' is the recommended name for the main output
80     # parameter.
81     self.addParameter(
82         QgsProcessingParameterRasterDestination(
83             'OUTPUT',
84             self.tr('Raster output')
85         )
86     )
87     self.addParameter(
88         QgsProcessingParameterDistance(
89             'BUFFERDIST',
90             self.tr('BUFFERDIST'),
91             defaultValue = 1.0,
92             # Make distance units match the INPUT layer units:
93             parentParameterName='INPUT'
94         )
95     )
96     self.addParameter(
97         QgsProcessingParameterDistance(
98             'CELLSIZE',
99             self.tr('CELLSIZE'),
100             defaultValue = 10.0,
101             parentParameterName='INPUT'
102         )
103     )
104     self.addOutput(
105         QgsProcessingOutputNumber(
106             'NUMBEROFFEATURES',
107             self.tr('Number of features processed')
108         )
109     )
110
111     def processAlgorithm(self, parameters, context, feedback):
112         """
113         Here is where the processing itself takes place.
114         """
115         # First, we get the count of features from the INPUT layer.
116         # This layer is defined as a QgsProcessingParameterFeatureSource
117         # parameter, so it is retrieved by calling
118         # self.parameterAsSource.
119         input_featuresource = self.parameterAsSource(parameters,
120                                                         'INPUT',
121                                                         context)
122         numfeatures = input_featuresource.featureCount()
123 
```

(Fortsetzung der vorherigen Seite)

```

124     # Retrieve the buffer distance and raster cell size numeric
125     # values. Since these are numeric values, they are retrieved
126     # using self.parameterAsDouble.
127     bufferdist = self.parameterAsDouble(parameters, 'BUFFERDIST',
128                                         context)
129     rastercellsize = self.parameterAsDouble(parameters, 'CELLSIZE',
130                                             context)
131
132     if feedback.isCanceled():
133         return {}
134     buffer_result = processing.run(
135         'native:buffer',
136         {
137             # Here we pass on the original parameter values of INPUT
138             # and BUFFER_OUTPUT to the buffer algorithm.
139             'INPUT': parameters['INPUT'],
140             'OUTPUT': parameters['BUFFER_OUTPUT'],
141             'DISTANCE': bufferdist,
142             'SEGMENTS': 10,
143             'DISSOLVE': True,
144             'END_CAP_STYLE': 0,
145             'JOIN_STYLE': 0,
146             'MITER_LIMIT': 10
147         },
148         # Because the buffer algorithm is being run as a step in
149         # another larger algorithm, the is_child_algorithm option
150         # should be set to True
151         is_child_algorithm=True,
152         # It's important to pass on the context and feedback objects to
153         # child algorithms, so that they can properly give feedback to
154         # users and handle cancelation requests.
155         context=context,
156         feedback=feedback)
157
158     # Check for cancelation
159     if feedback.isCanceled():
160         return {}
161
162     # Run the separate rasterization algorithm using the buffer result
163     # as an input.
164     rasterized_result = processing.run(
165         'qgis:rasterize',
166         {
167             # Here we pass the 'OUTPUT' value from the buffer's result
168             # dictionary off to the rasterize child algorithm.
169             'LAYER': buffer_result['OUTPUT'],
170             'EXTENT': buffer_result['OUTPUT'],
171             'MAP_UNITS_PER_PIXEL': rastercellsize,
172             # Use the original parameter value.
173             'OUTPUT': parameters['OUTPUT']
174         },
175         is_child_algorithm=True,
176         context=context,
177         feedback=feedback)
178
179     if feedback.isCanceled():
180         return {}
181
182     # Return the results
183     return {'OUTPUT': rasterized_result['OUTPUT'],
184           'BUFFER_OUTPUT': buffer_result['OUTPUT'],

```

(Fortsetzung auf der nächsten Seite)

```
'NUMBEROFFEATURES': numfeatures}
```

Processing algorithm standard functions:

- **createInstance (mandatory)**  
Must return a new copy of your algorithm. If you change the name of the class, make sure you also update the value returned here to match!
- **name (mandatory)**  
Returns the unique algorithm name, used for identifying the algorithm.
- **displayName (mandatory)**  
Returns the translated algorithm name.
- **group**  
Returns the name of the group this algorithm belongs to.
- **groupId**  
Returns the unique ID of the group this algorithm belongs to.
- **shortHelpString**  
Returns a localised short help string for the algorithm.
- **initAlgorithm (mandatory)**  
Here we define the inputs and outputs of the algorithm.  
  
INPUT and OUTPUT are recommended names for the main input and main output parameters, respectively.  
  
If a parameter depends on another parameter, `parentParameterName` is used to specify this relationship (could be the field / band of a layer or the distance units of a layer).
- **processAlgorithm (mandatory)**  
This is where the processing takes place.  
  
Parameters are retrieved using special purpose functions, for instance `parameterAsSource` and `parameterAsDouble`.  
  
`processing.run` can be used to run other processing algorithms from a processing algorithm. The first parameter is the name of the algorithm, the second is a dictionary of the parameters to the algorithm. `is_child_algorithm` is normally set to `True` when running an algorithm from within another algorithm. `context` and `feedback` inform the algorithm about the environment to run in and the channel for communicating with the user (catching cancel request, reporting progress, providing textual feedback). When using the (parent) algorithm's parameters as parameters to „child“ algorithms, the original parameter values should be used (e.g. `parameters['OUTPUT']`).  
  
It is good practice to check the feedback object for cancelation as much as is sensibly possible! Doing so allows for responsive cancelation, instead of forcing users to wait for unwanted processing to occur.  
  
The algorithm should return values for all the output parameters it has defined as a dictionary. In this case, that's the buffer and rasterized output layers, and the count of features processed. The dictionary keys must match the original parameter/output names.

## 27.9.2 The @alg decorator

Using the @alg decorator, you can create your own algorithms by writing the Python code and adding a few extra lines to supply additional information needed to make it a proper Processing algorithm. This simplifies the creation of algorithms and the specification of inputs and outputs.

One important limitation with the decorator approach is that algorithms created in this way will always be added to a user's Processing Scripts provider – it is not possible to add these algorithms to a custom provider, e.g. for use in plugins.

The following code uses the @alg decorator to

1. use a vector layer as input
2. count the number of features
3. do a buffer operation
4. create a raster layer from the result of the buffer operation
5. returns the buffer layer, raster layer and number of features

```

1 from qgis import processing
2 from qgis.processing import alg
3 from qgis.core import QgsProject
4
5 @alg(name='bufferrasteralg', label='Buffer and export to raster (alg)',
6      group='examplescripts', group_label='Example scripts')
7 # 'INPUT' is the recommended name for the main input parameter
8 @alg.input(type=alg.SOURCE, name='INPUT', label='Input vector layer')
9 # 'OUTPUT' is the recommended name for the main output parameter
10 @alg.input(type=alg.RASTER_LAYER_DEST, name='OUTPUT',
11            label='Raster output')
12 @alg.input(type=alg.VECTOR_LAYER_DEST, name='BUFFER_OUTPUT',
13            label='Buffer output')
14 @alg.input(type=alg.DISTANCE, name='BUFFERDIST', label='BUFFER DISTANCE',
15            default=1.0)
16 @alg.input(type=alg.DISTANCE, name='CELLSIZE', label='RASTER CELL SIZE',
17            default=10.0)
18 @alg.output(type=alg.NUMBER, name='NUMBEROFFEATURES',
19             label='Number of features processed')
20
21 def bufferrasteralg(instance, parameters, context, feedback, inputs):
22     """
23     Description of the algorithm.
24     (If there is no comment here, you will get an error)
25     """
26     input_featuresource = instance.parameterAsSource(parameters,
27                                                       'INPUT', context)
28     numfeatures = input_featuresource.featureCount()
29     bufferdist = instance.parameterAsDouble(parameters, 'BUFFERDIST',
30                                              context)
31     rastercellsize = instance.parameterAsDouble(parameters, 'CELLSIZE',
32                                                  context)
33     if feedback.isCanceled():
34         return {}
35     buffer_result = processing.run('native:buffer',
36                                   {'INPUT': parameters['INPUT'],
37                                    'OUTPUT': parameters['BUFFER_OUTPUT'],
38                                    'DISTANCE': bufferdist,
39                                    'SEGMENTS': 10,
40                                    'DISSOLVE': True,
41                                    'END_CAP_STYLE': 0,
42                                    'JOIN_STYLE': 0,

```

(Fortsetzung auf der nächsten Seite)

```

43         'MITER_LIMIT': 10
44     },
45     is_child_algorithm=True,
46     context=context,
47     feedback=feedback)
48
49     if feedback.isCanceled():
50         return {}
51     rasterized_result = processing.run('qgis:rasterize',
52                                     {'LAYER': buffer_result['OUTPUT'],
53                                      'EXTENT': buffer_result['OUTPUT'],
54                                      'MAP_UNITS_PER_PIXEL': rastercellsize,
55                                      'OUTPUT': parameters['OUTPUT']
56                                     },
57                                     is_child_algorithm=True, context=context,
58                                     feedback=feedback)
59
60     if feedback.isCanceled():
61         return {}
62     return {'OUTPUT': rasterized_result['OUTPUT'],
63            'BUFFER_OUTPUT': buffer_result['OUTPUT'],
64            'NUMBEROFFEATURES': numfeatures}

```

As you can see, it involves two algorithms (‘native:buffer’ and ‘qgis:rasterize’). The last one (‘qgis:rasterize’) creates a raster layer from the buffer layer that was generated by the first one (‘native:buffer’).

The part of the code where this processing takes place is not difficult to understand if you have read the previous chapter. The first lines, however, need some additional explanation. They provide the information that is needed to turn your code into an algorithm that can be run from any of the GUI components, like the toolbox or the model designer.

These lines are all calls to the `@alg` decorator functions that help simplify the coding of the algorithm.

- The `@alg` decorator is used to define the name and location of the algorithm in the Toolbox.
- The `@alg.input` decorator is used to define the inputs of the algorithm.
- The `@alg.output` decorator is used to define the outputs of the algorithm.

### 27.9.3 Input and output types for Processing Algorithms

Here is the list of input and output types that are supported in Processing with their corresponding `alg` decorator constants (the `algfactory.py` file contains the complete list of `alg` constants). Sorted on class name.

#### Input types

Class	Alg constant	Beschreibung
<code>QgsProcessingParameterAnnotationL</code>	<code>alg.ANNOTATION_LAYER</code>	An annotation layer
<code>QgsProcessingParameterAuthConfig</code>	<code>alg.AUTH_CFG</code>	Allows users to select from available authentication configurations or create new authentication configurations
<code>QgsProcessingParameterBand</code>	<code>alg.BAND</code>	A band of a raster layer
<code>QgsProcessingParameterBoolean</code>	<code>alg.BOOL</code>	A boolean value
<code>QgsProcessingParameterColor</code>	<code>alg.COLOR</code>	A color

Fortsetzung auf der nächsten Seite



Tab. 27.3 – Fortsetzung der vorherigen Seite

Class	Alg constant	Beschreibung
<code>QgsProcessingParameterCoordinateOperation</code>	<code>alg.COORDINATE_OPERATION</code>	A coordinate operation (for CRS transformations)
<code>QgsProcessingParameterCrs</code>	<code>alg.CRS</code>	A Coordinate Reference System
<code>QgsProcessingParameterDatabaseSchema</code>	<code>alg.DATABASE_SCHEMA</code>	A database schema
<code>QgsProcessingParameterDatabaseTable</code>	<code>alg.DATABASE_TABLE</code>	A database table
<code>QgsProcessingParameterDateTime</code>	<code>alg.DATETIME</code>	A datetime (or a pure date or time)
<code>QgsProcessingParameterDistance</code>	<code>alg.DISTANCE</code>	A double numeric parameter for distance values
<code>QgsProcessingParameterEnum</code>	<code>alg.ENUM</code>	An enumeration, allowing for selection from a set of predefined values
<code>QgsProcessingParameterExpression</code>	<code>alg.EXPRESSION</code>	An expression
<code>QgsProcessingParameterExtent</code>	<code>alg.EXTENT</code>	A spatial extent defined by xmin, xmax, ymin, ymax
<code>QgsProcessingParameterField</code>	<code>alg.FIELD</code>	A field in the attribute table of a vector layer
<code>QgsProcessingParameterFile</code>	<code>alg.FILE</code>	A filename of an existing file
<code>QgsProcessingParameterFileDestination</code>	<code>alg.FILE_DEST</code>	A filename for a newly created output file
<code>QgsProcessingParameterFolderDestination</code>	<code>alg.FOLDER_DEST</code>	A folder (destination folder)
<code>QgsProcessingParameterGeometry</code>	<code>alg.GEOMETRY</code>	A geometry
<code>QgsProcessingParameterInteger</code>	<code>alg.INT</code>	An integer
<code>QgsProcessingParameterLayout</code>	<code>alg.LAYOUT</code>	A layout
<code>QgsProcessingParameterLayoutItem</code>	<code>alg.LAYOUT_ITEM</code>	A layout item
<code>QgsProcessingParameterMapLayer</code>	<code>alg.MAPLAYER</code>	A map layer
<code>QgsProcessingParameterMapTheme</code>	<code>alg.MAP_THEME</code>	A project map theme
<code>QgsProcessingParameterMatrix</code>	<code>alg.MATRIX</code>	A matrix
<code>QgsProcessingParameterMeshLayer</code>	<code>alg.MESH_LAYER</code>	A mesh layer
<code>QgsProcessingParameterMultipleLayers</code>	<code>alg.MULTILAYER</code>	A set of layers
<code>QgsProcessingParameterNumber</code>	<code>alg.NUMBER</code>	A numerical value
<code>QgsProcessingParameterPoint</code>	<code>alg.POINT</code>	A point
<code>QgsProcessingParameterPointCloudDestination</code>	<code>alg.POINTCLOUD_LAYER_DEST</code>	A point cloud layer destination parameter, for specifying the destination path for a point cloud layer created by the algorithm
<code>QgsProcessingParameterPointCloudLayer</code>	<code>alg.POINTCLOUD_LAYER</code>	A point cloud layer
<code>QgsProcessingParameterProviderConnection</code>	<code>alg.PROVIDER_CONNECTION</code>	An available connection for a database provider
<code>QgsProcessingParameterRange</code>	<code>alg.RANGE</code>	A number range
<code>QgsProcessingParameterRasterLayer</code>	<code>alg.RASTER_LAYER</code>	Rasterlayer

Fortsetzung auf der nächsten Seite

Tab. 27.3 – Fortsetzung der vorherigen Seite

Class	Alg constant	Beschreibung
<code>QgsProcessingParameterRasterDesti</code>	<code>alg.RASTER_LAYER_DEST</code>	A raster layer destination parameter, for specifying the destination path for a raster layer created by the algorithm
<code>QgsProcessingParameterScale</code>	<code>alg.SCALE</code>	A map scale
<code>QgsProcessingParameterFeatureSink</code>	<code>alg.SINK</code>	A feature sink
<code>QgsProcessingParameterFeatureSour</code>	<code>alg.SOURCE</code>	A feature source
<code>QgsProcessingParameterString</code>	<code>alg.STRING</code>	A text string
<code>QgsProcessingParameterVectorLayer</code>	<code>alg.VECTOR_LAYER</code>	Vektorlayer
<code>QgsProcessingParameterVectorDesti</code>	<code>alg.VECTOR_LAYER_DEST</code>	A vector layer destination parameter, for specifying the destination path for a vector layer created by the algorithm
<code>QgsProcessingParameterVectorTileD</code>		A vector tile layer destination parameter, for specifying the destination path for a vector tile layer created by the algorithm

## Output types

Class	Alg constant	Beschreibung
<code>QgsProcessingOutputBoolean</code>	<code>alg.BOOL</code>	A boolean value
<code>QgsProcessingOutputNumber</code>	<code>alg.DISTANCE</code>	A double numeric parameter for distance values
<code>QgsProcessingOutputFile</code>	<code>alg.FILE</code>	A filename of an existing file
<code>QgsProcessingOutputFolder</code>	<code>alg.FOLDER</code>	A folder
<code>QgsProcessingOutputHtml</code>	<code>alg.HTML</code>	HTML
<code>QgsProcessingOutputNumber</code>	<code>alg.INT</code>	A integer
<code>QgsProcessingOutputLayerDefinition</code>	<code>alg.LAYERDEF</code>	A layer definition
<code>QgsProcessingOutputMapLayer</code>	<code>alg.MAPLAYER</code>	A map layer
<code>QgsProcessingOutputMultipleLayers</code>	<code>alg.MULTILAYER</code>	A set of layers
<code>QgsProcessingOutputNumber</code>	<code>alg.NUMBER</code>	A numerical value
<code>QgsProcessingOutputPointCloudLayer</code>	<code>alg.POINTCLOUD_LAYER</code>	A point cloud layer
<code>QgsProcessingOutputRasterLayer</code>	<code>alg.RASTER_LAYER</code>	Rasterlayer
<code>QgsProcessingOutputString</code>	<code>alg.STRING</code>	A text string
<code>QgsProcessingOutputVectorLayer</code>	<code>alg.VECTOR_LAYER</code>	Vektorlayer
<code>QgsProcessingOutputVectorTileLayer</code>		A vector tile layer

## 27.9.4 Handling algorithm output

When you declare an output representing a layer (raster or vector), the algorithm will try to add it to QGIS once it is finished.

- Raster layer output: `QgsProcessingParameterRasterDestination / alg.RASTER_LAYER_DEST`.
- Vector layer output: `QgsProcessingParameterVectorDestination / alg.VECTOR_LAYER_DEST`.

So even if the `processing.run()` method does not add the layers it creates to the user's current project, the two output layers (buffer and raster buffer) will be loaded, since they are saved to the destinations entered by the user (or to temporary destinations if the user does not specify destinations).

If a layer is created as output of an algorithm, it should be declared as such. Otherwise, you will not be able to properly use the algorithm in the modeler, since what is declared will not match what the algorithm really creates.

You can return strings, numbers and more by specifying them in the result dictionary (as demonstrated for „NUMBEROFFEATURES“), but they should always be explicitly defined as outputs from your algorithm. We encourage algorithms to output as many useful values as possible, since these can be valuable for use in later algorithms when your algorithm is used as part of a model.

## 27.9.5 Die Kommunikation mit dem Benutzer

If your algorithm takes a long time to process, it is a good idea to inform the user about the progress. You can use `feedback` (`QgsProcessingFeedback`) for this.

The progress text and progressbar can be updated using two methods: `setProgressText(text)` and `setProgress(percent)`.

You can provide more information by using `pushCommandInfo(text)`, `pushDebugInfo(text)`, `pushInfo(text)` and `reportError(text)`.

If your script has a problem, the correct way of handling it is to raise a `QgsProcessingException`. You can pass a message as an argument to the constructor of the exception. Processing will take care of handling it and communicating with the user, depending on where the algorithm is being executed from (toolbox, modeler, Python console, ...)

## 27.9.6 Ihre Scripte dokumentieren

You can document your scripts by overloading the `helpString()` and `helpUrl()` methods of `QgsProcessingAlgorithm`.

## 27.9.7 Flags

You can override the `flags()` method of `QgsProcessingAlgorithm` to tell QGIS more about your algorithm. You can for instance tell QGIS that the script shall be hidden from the modeler, that it can be canceled, that it is not thread safe, and more.

---

**Tipp:** By default, Processing runs algorithms in a separate thread in order to keep QGIS responsive while the processing task runs. If your algorithm is regularly crashing, you are probably using API calls which are not safe to do in a background thread. Try returning the `QgsProcessingAlgorithm.FlagNoThreading` flag from your algorithm's `flags()` method to force Processing to run your algorithm in the main thread instead.

---

## 27.9.8 Bewährte Verfahren für das Schreiben von Skript-Algorithmen

Here's a quick summary of ideas to consider when creating your script algorithms and, especially, if you want to share them with other QGIS users. Following these simple rules will ensure consistency across the different Processing elements such as the toolbox, the modeler or the batch processing interface.

- Laden Sie keine Ergebnislayer. Lassen Sie die Verarbeitung Ihrer Ergebnisse abarbeiten und laden Sie Ihre Layer, wenn nötig.
- Always declare the outputs your algorithm creates.
- Do not show message boxes or use any GUI element from the script. If you want to communicate with the user, use the methods of the feedback object (`QgsProcessingFeedback`) or throw a `QgsProcessingException`.

There are already many processing algorithms available in QGIS. You can find code on the [QGIS repo](#).

## 27.10 Konfiguration externer Anwendungen

The processing framework can be extended using additional applications. Algorithms that rely on external applications are managed by their own algorithm providers. Additional providers can be found as separate plugins, and installed using the QGIS Plugin Manager.

This section will show you how to configure the Processing framework to include these additional applications, and it will explain some particular features of the algorithms based on them. Once you have correctly configured the system, you will be able to execute external algorithms from any component like the toolbox or the model designer, just like you do with any other algorithm.

By default, algorithms that rely on an external application not shipped with QGIS are not enabled. You can enable them in the Processing settings dialog if they are installed on your system.

### 27.10.1 Ein Hinweis für Windows Anwender

If you are not an advanced user and you are running QGIS on Windows, you might not be interested in reading the rest of this chapter. Make sure you install QGIS in your system using the standalone installer. That will automatically install SAGA and GRASS in your system and configure them so they can be run from QGIS. All the algorithms from these providers will be ready to be run without needing any further configuration. If installing with the OSGeo4W application, make sure that you also select SAGA and GRASS for installation.

### 27.10.2 Eine Bemerkung zu den Datenformaten

When using external software, opening a file in QGIS does not mean that it can be opened and processed in that other software. In most cases, other software can read what you have opened in QGIS, but in some cases, that might not be true. When using databases or uncommon file formats, whether for raster or vector layers, problems might arise. If that happens, try to use well-known file formats that you are sure are understood by both programs, and check the console output (in the log panel) to find out what is going wrong.

You might for instance get trouble and not be able to complete your work if you call an external algorithm with a GRASS raster layers as input. For this reason, such layers will not appear as available to algorithms.

You should, however, not have problems with vector layers, since QGIS automatically converts from the original file format to one accepted by the external application before passing the layer to it. This adds extra processing time, which might be significant for large layers, so do not be surprised if it takes more time to process a layer from a DB connection than a layer from a Shapefile format dataset of similar size.

Provider, die keine externe Anwendung verwenden können jeden beliebigen Layer, den Sie in QGIS öffnen können, prozessieren, da Sie ihn für Analysen anhand von QGIS öffnen.

All raster and vector output formats produced by QGIS can be used as input layers. Some providers do not support certain formats, but all can export to common formats that can later be transformed by QGIS automatically. As for input layers, if a conversion is needed, that might increase the processing time.

### 27.10.3 Eine Bemerkung zu Vektorlayern mit ausgewählten Objekten

Externe Anwendungen können auch über die Auswahl die in Vektorlayern innerhalb von QGIS vorliegt unterrichtet werden. Das jedoch erfordert das Umschreiben aller Eingabevektorlayer, so als würden Sie ursprünglich in einem Format, das nicht von der externen Anwendung unterstützt wird, vorliegen. Nur wenn eine Auswahl besteht, oder die *Use only selected features* Option in den Verarbeitungsoptionen nicht aktiviert ist, kann ein Layer direkt an eine externe Anwendung übergeben werden.

In other cases, exporting only selected features is needed, which causes longer execution times.

### 27.10.4 Using third-party Providers

#### SAGA

SAGA algorithms can be run from QGIS if SAGA is included with the QGIS installation.

If you are running Windows, both the stand-alone installer and the OSGeo4W installer include SAGA.

#### Über die SAGA KBS Einschränkungen

Die meisten SAGA Algorithmen, die mehrere Eingaberasterlayer benötigen, haben die Anforderung, dass diese das gleiche Grid-System haben. Das heißt, dass Sie das gleiche geografische Gebiet abdecken und die gleiche Zellgröße besitzen müssen, so dass Ihre korrespondierenden Grids zueinander passen. Wenn SAGA Algorithmen von QGIS aus aufgerufen werden, können Sie jeden Layer unabhängig von Zellgröße und Ausdehnung benutzen. Wenn Mehrfachrasterlayer als Eingabe für einen SAGA Algorithmus verwendet werden, resampled QGIS sie zu einem gemeinsamen Grid-System und übergibt diese dann an SAGA (es sei denn der SAGA Algorithmus kann mit Layern aus verschiedenen Grid-Systemen operieren).

Die Definition dieses gemeinsamen Grid-Systems wird durch den Anwender kontrolliert und Sie werden dafür mehrere Parameter in der SAGA Gruppe im Einstellungen Fenster finden. Es gibt zwei Wege das Ziel-Grid-System einzustellen:

- Manuelles Einstellen. Sie definieren das Ausmaß indem Sie die Werte der folgenden Parameter setzen:
  - *Resampling min X*
  - *Resampling max X*
  - *Resampling min Y*
  - *Resampling max Y*
  - *Resampling cellsize*

Beachten Sie, dass QGIS alle Eingabelayer bis zu diesem Ausmaß resampled, auch wenn sie sich nicht damit überschneiden.

- Automatisches Einstellen aus Eingabelayern. Um diese Option auszuwählen, überprüfen Sie einfach die *Use min covering grid system for resampling* Option. Alle anderen Einstellungen werden ignoriert und das minimale Ausmaß, das alle Eingabelayer abdeckt, wird benutzt. Die Zellgröße des Ziellayers ist das Maximum aller Zellgrößen des Eingabelayers.

Für Algorithmen, die nicht mit mehreren Rasterlayern arbeiten, oder für diejenigen, die kein eindeutiges KBS brauchen, wird kein Resampling vor dem Aufruf von SAGA durchgeführt, und die Parameter werden nicht verwendet.

### Einschränkungen für Multi-Band-Layer

Im Gegensatz zu QGIS hat SAGA keine Unterstützung für Multikanallayer. Wenn Sie einen Multikanallayer benutzen wollen (so wie ein RGB oder ein Multispektralbild), müssen Sie ihn erst in Einkanalbilder aufspalten. Um das zu tun, können Sie den ‚SAGA/Grid - Tools/Split RGB image‘ Algorithmus (der drei Bilder aus einem RGB Bild erstellt) oder den ‚SAGA/Grid - Tools/Extract band‘ Algorithmus (um einen einzelnen Kanal zu extrahieren) verwenden.

### Einschränkungen in der Zellgröße

SAGA geht davon aus, dass Rasterlayer die selbe Pixelgröße in X- und Y-Richtung haben. Wenn Sie mit einem Layer mit unterschiedlichen Werten für die horizontale und vertikale Pixelgröße arbeiten, erhalten Sie möglicherweise unerwartete Ergebnisse. In diesem Fall wird eine Warnung im Verarbeitung Protokoll hinzugefügt, das anzeigt, dass ein Layer möglicherweise nicht geeignet ist, von SAGA verarbeitet zu werden.

### Logging

Wenn QGIS SAGA aufruft, findet das über die Kommandozeilen-Schnittstelle statt mit einer Reihe von Befehlen, um alle erforderlichen Operation durchzuführen. SAGA zeigt den Fortschritt, indem es Informationen an die Konsole übergibt, die den Prozentsatz der Verarbeitung beinhaltet, zusammen mit zusätzlichen Inhalten. Diese Ausgabe wird gefiltert und verwendet, um die Fortschrittsanzeige zu aktualisieren, während der Algorithmus läuft.

Both the commands sent by QGIS and the additional information printed by SAGA can be logged along with other processing log messages, and you might find them useful to track what is going on when QGIS runs a SAGA algorithm. You will find two settings, namely *Log console output* and *Log execution commands*, to activate that logging mechanism.

Most other providers that use external applications and call them through the command-line have similar options, so you will find them as well in other places in the processing settings list.

### R scripts and libraries

To enable R in Processing you need to install the **Processing R Provider** plugin and configure R for QGIS.

Configuration is done in *Provider ► R* in the *Processing* tab of *Settings ► Options*.

Depending on your operating system, you may have to use *R folder* to specify where your R binaries are located.

---

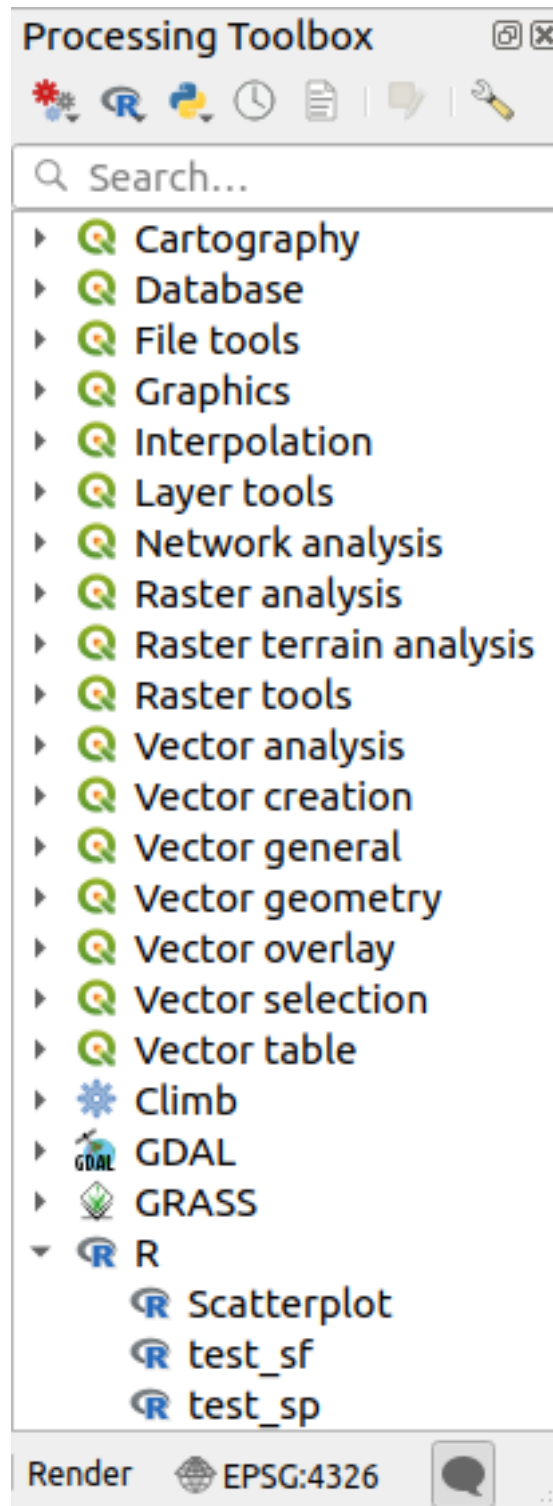
**Bemerkung:** On **Windows** the R executable file is normally in a folder (R-<version>) under C:\Program Files\R\. Specify the folder and **NOT** the binary!

On **Linux** you just have to make sure that the R folder is in the PATH environment variable. If R in a terminal window starts R, then you are ready to go.

---

After installing the **Processing R Provider** plugin, you will find some example scripts in the *Processing Toolbox*:

- *Scatterplot* runs an R function that produces a scatter plot from two numerical fields of the provided vector layer.
- *test\_sf* does some operations that depend on the *sf* package and can be used to check if the R package *sf* is installed. If the package is not installed, R will try to install it (and all the packages it depends on) for you, using the *Package repository* specified in *Provider ► R* in the Processing options. The default is <https://cran.r-project.org/>. Installing may take some time...
- *test\_sp* can be used to check if the R package *sp* is installed. If the package is not installed, R will try to install it for you.



If you have R configured correctly for QGIS, you should be able to run these scripts.

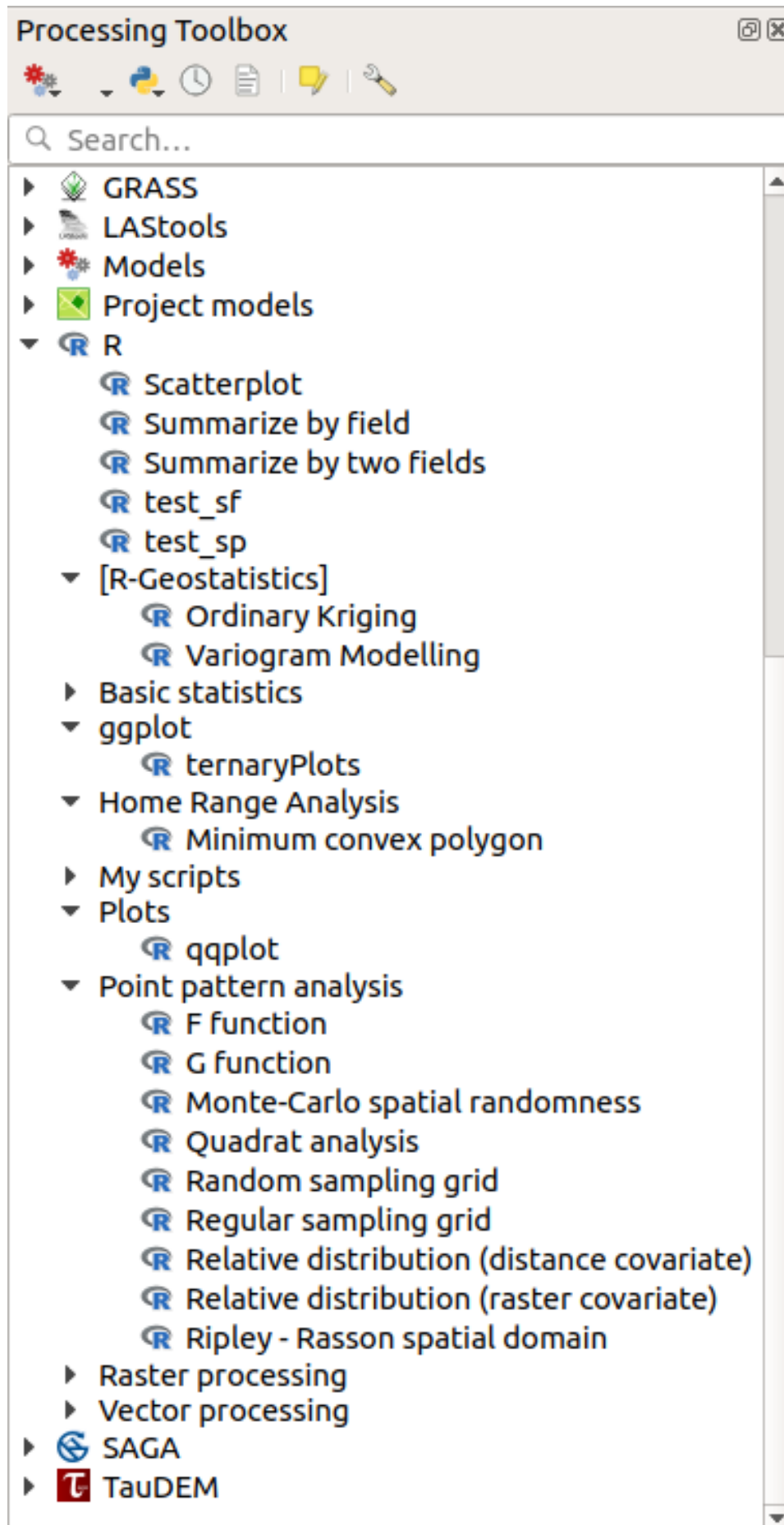
### Adding R scripts from the QGIS collection

R integration in QGIS is different from that of SAGA in that there is not a predefined set of algorithms you can run (except for some example script that come with the *Processing R Provider* plugin).

A set of example R scripts is available in the QGIS Repository. Perform the following steps to load and enable them using the *QGIS Resource Sharing* plugin.

1. Add the *QGIS Resource Sharing* plugin (you may have to enable *Show also experimental plugins* in the Plugin Manager *Settings*)
2. Open it (*Plugins ► Resource Sharing ► Resource Sharing*)
3. Choose the *Settings* tab
4. Click *Reload repositories*
5. Choose the *All* tab
6. Select *QGIS R script collection* in the list and click on the *Install* button
7. The collection should now be listed in the *Installed* tab
8. Close the plugin
9. Open the *Processing Toolbox*, and if everything is OK, the example scripts will be present under R, in various groups (only some of the groups are expanded in the screenshot below).



Abb. 27.36: The *Processing Toolbox* with some R scripts shown

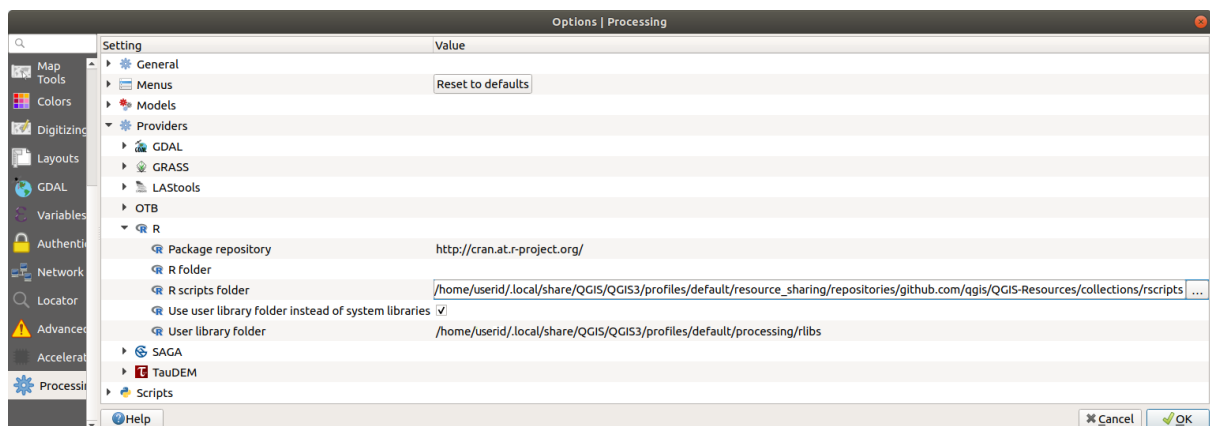
The scripts at the top are the example scripts from the *Processing R Provider* plugin.

10. If, for some reason, the scripts are not available in the *Processing Toolbox*, you can try to:

1. Open the Processing settings (*Settings ► Options ► Processing* tab)
2. Go to *Providers ► R ► R scripts folder*

- On Ubuntu, set the path to (or, better, include in the path):

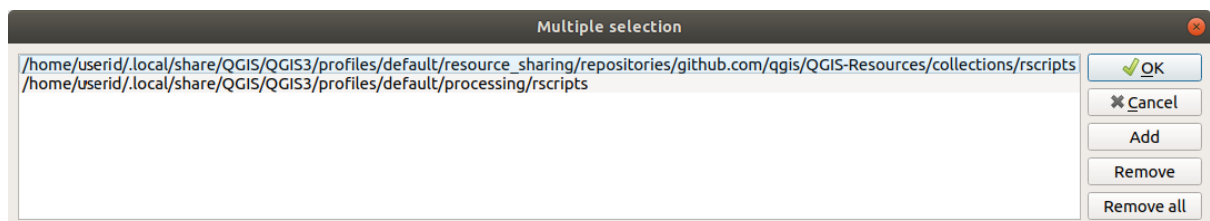
/home/<user>/local/share/QGIS/QGIS3/profiles/default/resource\_sharing/repositories/github.com/qgis/QGIS-Resources/collections/rscripts



- On Windows, set the path to (or, better, include in the path):

C:\Users<user>\AppData\Roaming\QGIS\QGIS3\profiles\default\resource\_sharing\repositories\github.com\qgis-Resources\collections\rscripts

To edit, double-click. You can then choose to just paste / type the path, or you can navigate to the directory by using the ... button and press the *Add* button in the dialog that opens. It is possible to provide several directories here. They will be separated by a semicolon (,;“).



If you would like to get all the R scripts from the QGIS 2 on-line collection, you can select *QGIS R script collection (from QGIS 2)* instead of *QGIS R script collection*. You will probably find that scripts that depend on vector data input or output will not work.

### Creating R scripts

You can write scripts and call R commands, as you would do from R. This section shows you the syntax for using R commands in QGIS, and how to use QGIS objects (layers, tables) in them.

To add an algorithm that calls an R function (or a more complex R script that you have developed and you would like to have available from QGIS), you have to create a script file that performs the R commands.

R script files have the extension *.rsx*, and creating them is pretty easy if you just have a basic knowledge of R syntax and R scripting. They should be stored in the R scripts folder. You can specify the folder (*R scripts folder*) in the *R* settings group in Processing settings dialog).

Let's have a look at a very simple script file, which calls the R method `spsample` to create a random grid within the boundary of the polygons in a given polygon layer. This method belongs to the `maptools` package. Since almost all the algorithms that you might like to incorporate into QGIS will use or generate spatial data, knowledge of spatial packages like `maptools` and `sp/sf`, is very useful.

```
##Random points within layer extent=name
##Point pattern analysis=group
##Vector_layer=vector
##Number_of_points=number 10
##Output=output vector
library(sp)
spatpoly = as(Vector_layer, "Spatial")
pts=spsample(spatpoly,Number_of_points,type="random")
spdf=SpatialPointsDataFrame(pts, as.data.frame(pts))
Output=st_as_sf(spdf)
```

The first lines, which start with a double Python comment sign (`##`), define the display name and group of the script, and tell QGIS about its inputs and outputs.

**Bemerkung:** To find out more about how to write your own R scripts, have a look at the R Intro section in the training manual and consult the [QGIS R Syntax](#) section.

When you declare an input parameter, QGIS uses that information for two things: creating the user interface to ask the user for the value of that parameter, and creating a corresponding R variable that can be used as R function input.

In the above example, we have declared an input of type `vector`, named `Vector_layer`. When executing the algorithm, QGIS will open the layer selected by the user and store it in a variable named `Vector_layer`. So, the name of a parameter is the name of the variable that you use in R for accessing the value of that parameter (you should therefore avoid using reserved R words as parameter names).

Spatial parameters such as vector and raster layers are read using the `st_read()` (or `readOGR`) and `brick()` (or `readGDAL`) commands (you do not have to worry about adding those commands to your description file – QGIS will do it), and they are stored as `sf` (or `Spatial*DataFrame`) objects.

Table fields are stored as strings containing the name of the selected field.

Vector files can be read using the `readOGR()` command instead of `st_read()` by specifying `##load_vector_using_rgdal`. This will produce a `Spatial*DataFrame` object instead of an `sf` object.

Raster files can be read using the `readGDAL()` command instead of `brick()` by specifying `##load_raster_using_rgdal`.

If you are an advanced user and do not want QGIS to create the object for the layer, you can use `##pass_filenames` to indicate that you prefer a string with the filename. In this case, it is up to you to open the file before performing any operation on the data it contains.

With the above information, it is possible to understand the first lines of the R script (the first line not starting with a Python comment character).

```
library(sp)
spatpoly = as(Vector_layer, "Spatial")
pts=spsample(polyg,numpoints,type="random")
```

The `spsample` function is provided by the `sp` library, so the first thing we do is to load that library. The variable `Vector_layer` contains an `sf` object. Since we are going to use a function (`spsample`) from the `sp` library, we must convert the `sf` object to a `SpatialPolygonsDataFrame` object using the `as` function.

Then we call the `spsample` function with this object and the `numpoints` input parameter (which specifies the number of points to generate).

Since we have declared a vector output named `Output`, we have to create a variable named `Output` containing an `sf` object.

We do this in two steps. First we create a `SpatialPolygonsDataFrame` object from the result of the function, using the `SpatialPointsDataFrame` function, and then we convert that object to an `sf` object using the `st_as_sf` function (of the `sf` library).

You can use whatever names you like for your intermediate variables. Just make sure that the variable storing your final result has the defined name (in this case `Output`), and that it contains a suitable value (an `sf` object for vector layer output).

In this case, the result obtained from the `spsample` method had to be converted explicitly into an `sf` object via a `SpatialPointsDataFrame` object, since it is itself an object of class `ppp`, which can not be returned to QGIS.

If your algorithm generates raster layers, the way they are saved will depend on whether or not you have used the `##dontuserasterpackage` option. If you have used it, layers are saved using the `writeGDAL()` method. If not, the `writeRaster()` method from the `raster` package will be used.

If you have used the `##pass_filenames` option, outputs are generated using the `raster` package (with `writeRaster()`).

If your algorithm does not generate a layer, but a text result in the console instead, you have to indicate that you want the console to be shown once the execution is finished. To do so, just start the command lines that produce the results you want to print with the `>` (‘greater than’) sign. Only output from lines prefixed with `>` are shown. For instance, here is the description file of an algorithm that performs a normality test on a given field (column) of the attributes of a vector layer:

```
##layer=vector
##field=field layer
##nortest=group
library(nortest)
>lillie.test(layer[[field]])
```

Die Ausgabe der letzten Zeile wird gedruckt, die Ausgabe der ersten aber nicht (und auch nicht die Ausgaben von anderen Kommandozeilen, die automatisch von QGIS hinzugefügt werden).

If your algorithm creates any kind of graphics (using the `plot()` method), add the following line (`output_plots_to_html` used to be `showplots`):

```
##output_plots_to_html
```

Dies bewirkt, dass QGIS alle grafischen Ausgaben von R in eine temporäre Datei, die geöffnet wird nachdem die Ausführung von R beendet wurde, umleitet.

Both graphics and console results will be available through the processing results manager.

For more information, please check the R scripts in the official QGIS collection (you download and install them using the *QGIS Resource Sharing* plugin, as explained in [Adding R scripts from the QGIS collection](#)). Most of them are rather simple and will greatly help you understand how to create your own scripts.

---

**Bemerkung:** The `sf`, `rgdal` and `raster` libraries are loaded by default, so you do not have to add the corresponding `library()` commands. However, other libraries that you might need have to be explicitly loaded by typing: `library(ggplot2)` (to load the `ggplot2` library). If the package is not already installed on your machine, Processing will try to download and install it. In this way the package will also become available in R Standalone. **Be aware** that if the package has to be downloaded, the script may take a long time to run the first time.

---

## R libraries installed when running sf\_test

The R script `sp_test` tries to load the R packages `sp` and `raster`.

The R script `sf_test` tries to load `sf` and `raster`. If these two packages are not installed, R may try to load and install them (and all the libraries that they depend on).

The following R libraries end up in `~/.local/share/QGIS/QGIS3/profiles/default/processing/rscripts` after `sf_test` has been run from the Processing Toolbox on Ubuntu with version 2.0 of the *Processing R Provider* plugin and a fresh install of R 3.4.4 (*apt* package `r-base-core` only):

```
abind, askpass, assertthat, backports, base64enc, BH, bit, bit64, blob, brew,
↳ callr, classInt, cli, colorspace, covr, crayon, crosstalk, curl, DBI, deldir,
desc, dichromat, digest, dplyr, e1071, ellipsis, evaluate, fansi, farver, fastmap,
↳ gdtools, ggplot2, glue, goftest, gridExtra, gtable, highr, hms,
htmltools, htmlwidgets, httpuv, httr, jsonlite, knitr, labeling, later, lazyeval,
↳ leafem, leaflet, leaflet.providers, leafpop, leafsync, lifecycle, lwgeom,
magrittr, maps, mapview, markdown, memoise, microbenchmark, mime, munsell, odbc,
↳ openssl, pillar, pkgbuild, pkgconfig, pkgload, plogr, plyr, png, polyclip,
praise, prettyunits, processx, promises, ps, purrr, R6, raster, RColorBrewer, Rcpp,
↳ reshape2, rex, rgeos, rlang, rmarkdown, RPostgres, RPostgreSQL,
rprojroot, RSQLite, rstudioapi, satellite, scales, sf, shiny, sourcetools, sp,
↳ spatstat, spatstat.data, spatstat.utils, stars, stringi, stringr, svglite,
sys, systemfonts, tensor, testthat, tibble, tidyselect, tinytex, units, utf8, uuid,
↳ vctrs, viridis, viridisLite, webshot, withr, xfun, XML, xtable
```

## GRASS

Configuring GRASS is not much different from configuring SAGA. First, the path to the GRASS folder has to be defined, but only if you are running Windows.

By default, the Processing framework tries to configure its GRASS connector to use the GRASS distribution that ships along with QGIS. This should work without problems for most systems, but if you experience problems, you might have to configure the GRASS connector manually. Also, if you want to use a different GRASS installation, you can change the setting to point to the folder where the other version is installed. GRASS 7 is needed for algorithms to work correctly.

If you are running Linux, you just have to make sure that GRASS is correctly installed, and that it can be run without problem from a terminal window.

GRASS Algorithmen verwenden eine Region für die Berechnungen. Diese Region kann manuell definiert werden unter Verwendung von Werten ähnlich denen, die in der SAGA Konfiguration stehen oder automatisch, wobei die minimale Ausdehnung aller Eingangslayer verwendet werden, wenn der Algorithmus ausgeführt wird. Wenn dies das Verhalten ist, das Sie bevorzugen, können Sie es über die Option *Verwende min abdecken Region* in den GRASS Konfigurationsparametern definieren.

## LAStools

To use [LAStools](#) in QGIS, you need to download and install LAStools on your computer and install the LAStools plugin (available from the official repository) in QGIS.

On Linux platforms, you will need [Wine](#) to be able to run some of the tools.

LAStools is activated and configured in the Processing options (*Settings ► Options, Processing* tab, *Providers ► LAStools*), where you can specify the location of LAStools (*LAStools folder*) and Wine (*Wine folder*). On Ubuntu, the default Wine folder is `/usr/bin`.

### OTB Applications

OTB applications are fully supported within the QGIS Processing framework.

**OTB** (Orfeo ToolBox) is an image processing library for remote sensing data. It also provides applications that provide image processing functionalities. The list of applications and their documentation are available in [OTB CookBook](#)

---

**Bemerkung:** Note that OTB is not distributed with QGIS and needs to be installed separately. Binary packages for OTB can be found on the [download page](#).

---

To configure QGIS processing to find the OTB library:

1. Open the processing settings: *Settings ► Options ► Processing*
2. You can see OTB under *Providers* menu:
  1. Expand the *OTB* entry
  2. Set the *OTB folder*. This is the location of your OTB installation.
  3. Set the *OTB application folder*. This is the location of your OTB applications (`<PATH_TO_OTB_INSTALLATION>/lib/otb/applications`)
  4. Click *OK* to save the settings and close the dialog.

If settings are correct, OTB algorithms will be available in the *Processing Toolbox*.

### Documentation of OTB settings available in QGIS Processing

- **OTB folder:** This is the directory where OTB is available.
- **OTB application folder:** This is the location(s) of OTB applications.

Multiple paths are allowed.
- **Logger level** (optional): Level of logger to use by OTB applications.

The level of logging controls the amount of detail printed during algorithm execution. Possible values for logger level are `INFO`, `WARNING`, `CRITICAL`, `DEBUG`. This value is `INFO` by default. This is an advanced user configuration.
- **Maximum RAM to use** (optional): by default, OTB applications use all available system RAM.

You can, however, instruct OTB to use a specific amount of RAM (in MB) using this option. A value of 256 is ignored by the OTB processing provider. This is an advanced user configuration.
- **Geoid file** (optional): Path to the geoid file.

This option sets the value of the `elev.dem.geoid` and `elev.geoid` parameters in OTB applications. Setting this value globally enables users to share it across multiple processing algorithms. Empty by default.
- **SRTM tiles folder** (optional): Directory where SRTM tiles are available.

SRTM data can be stored locally to avoid downloading of files during processing. This option sets the value of `elev.dem.path` and `elev.dem` parameters in OTB applications. Setting this value globally enables users to share it across multiple processing algorithms. Empty by default.

## **Compatibility and Troubleshoot**

Starting from OTB 6.6.1, new releases of OTB are made compatible with at least the latest QGIS version available at that time.

If you have issues with OTB applications in QGIS Processing, please open an issue on the [OTB bug tracker](#), using the `qgis` label.

Additional information about OTB and QGIS can be found in [OTB Cookbook](#).





Datenanbieter und Algorithmen in Verarbeitung

Im Folgenden werden die Algorithmen und ihre Parameter beschrieben (wie sie in der Benutzeroberfläche dargestellt werden).

28.1 GDAL Algorithmus-Anbieter

QGIS algorithm provider implements various analysis and geoprocessing operations using mostly only QGIS API. So almost all algorithms from this provider will work „out of the box“ without any additional configuration.

This provider incorporates some algorithms from plugins and also adds its own algorithms.

28.1.1 3D Tiles

NEW in 3.34

Convert B3DM to GLTF

Converts files from the legacy .B3DM format to .GLTF or .GLB.

Parameters

Label	Name	Type	Description
Input B3DM	INPUT	[3D Tile]	Input file to convert.
Output file	OUTPUT	[3D Tile] Default: [Save to temporary file]	Specify the output 3D tile file. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Outputs

Label	Name	Type	Description
<b>Output file</b>	OUTPUT	[3D Tile]	Output layer in .GLTF format.

## Python code

**Algorithm ID:** native:b3dmtogltf

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert GLTF to vector features

Converts .GLTF or .GLB file contents to standard vector layer formats.

## Parameters

Label	Name	Type	Description
<b>Input GLTF</b>	INPUT	[3D Tile]	Input file to convert.
<b>Output polygons</b> Optional	OUTPUT_POLYGON	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer to store polygon features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Output lines</b> Optional	OUTPUT_LINES	[vector: line] Default: [Create temporary layer]	Specify the output vector layer to store line features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Outputs

Label	Name	Type	Description
<b>Output polygons</b>	OUTPUT_POLYGON	[vector: polygon]	A vector layer storing the polygon features in input layer.
<b>Output lines</b>	OUTPUT_LINES	[vector: line]	A vector layer storing the line features in input layer.

## Python code

**Algorithm ID:** native:gltfvector

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.2 Cartography

### Align points to features

Calculates the rotation required to align point features with their nearest feature from another reference layer. A new field is added to the output layer which is filled with the angle (in degrees, clockwise) to the nearest reference feature.

Optionally, the output layer's symbology can be set to automatically use the calculated rotation field to rotate marker symbols. If desired, a maximum distance to use when aligning points can be set, to avoid aligning isolated points to distant features.

---

**Hinweis:** This algorithm is designed for use cases like aligning building point symbols to follow the nearest road direction.

---



Allows *features in-place modification* of point features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Point features to calculate the rotation for
<b>Reference layer</b>	REFERENCE_LAYER	[vector: any]	Layer to find the closest feature from for rotation calculation
<b>Maximum distance to consider</b> Optional	MAX_DISTANCE	[number] Default: Not set	If no reference feature is found within this distance, no rotation is assigned to the point feature.
<b>Angle field name</b>	FIELD_NAME	[string] Default: 'rotation'	Field in which to store the rotation value.
<b>Automatically apply symbology</b>	APPLY_SYMBOLOG	[boolean] Default: True	Rotates the symbol marker of the features using the angle field value
<b>Aligned layer</b>	OUTPUT	[vector: point] Default: [Save to temporary file]	Specify the rotated output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aligned layer</b>	OUTPUT	[vector: point]	The point layer appended with a rotation field. If loaded to QGIS, it is applied by default the input layer symbology, with a data-defined rotation of its marker symbol.

## Pythoncode

**Algorithm ID:** native:angletonearest

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Combine style databases

Combines multiple QGIS style databases into a single style database. If items of the same type with the same name exist in different source databases these will be renamed to have unique names in the output combined database.

### Siehe auch:

*Create style database from project*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input databases</b>	INPUT	[file] [list]	Files containing QGIS style items
<b>Objects to combine</b>	OBJECTS	[enumeration] [list]	Types of style items in the input databases you would like to put in the new database. These can be: <ul style="list-style-type: none"> <li>• 0 — <i>Symbols</i></li> <li>• 1 — <i>Color ramps</i></li> <li>• 2 — <i>Text formats</i></li> <li>• 3 — <i>Label settings</i></li> </ul>
<b>Output style database</b>	OUTPUT	[file] Default: [Save to temporary file]	Output .XML file combining the selected style items. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Color ramp count</b>	COLORRAMPS	[number]	
<b>Label settings count</b>	LABELSETTINGS	[number]	
<b>Output style database</b>	OUTPUT	[file]	Output .XML file combining the selected style items
<b>Symbol count</b>	SYMBOLS	[number]	
<b>Text format count</b>	TEXTFORMATS	[number]	

### Pythoncode

**Algorithm ID:** native:combinestyles

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create categorized renderer from styles

Sets a vector layer's renderer to a categorized renderer using matching symbols from a style database. If no style file is specified, symbols from the user's current *symbol library* are used instead.

A specified expression or field is used to create categories for the renderer. Each category is individually matched to the symbols which exist within the specified QGIS XML style database. Whenever a matching symbol name is found, the category's symbol will be set to this matched symbol.

If desired, outputs can also be tables containing lists of the categories which could not be matched to symbols, and symbols which were not matched to categories.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer to apply a categorized style to
<b>Categorize using expression</b>	FIELD	[expression]	Field or expression to categorize the features
<b>Style database (leave blank to use saved symbols)</b>	STYLE	[file]	File (.XML) containing the symbols to apply to the input layer categories. The file can be obtained from the Style Manager <i>Share symbols</i> tool. If no file is specified, QGIS local symbols library is used.
<b>Use case-sensitive match to symbol names</b>	CASE_SENSITIVE	[boolean] Default: False	If True (checked), applies a case sensitive comparison between the categories and symbols names
<b>Ignore non-alphanumeric characters while matching</b>	TOLERANT	[boolean] Default: False	If True (checked), non-alphanumeric characters in the categories and symbols names will be ignored, allowing greater tolerance during the match.
<b>Non-matching categories</b> Optional	NON_MATCHING_C	[table] Default: [Skip output]	Output table for categories which do not match any symbol in the database. One of: <ul style="list-style-type: none"> <li>Skip Output</li> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Non-matching symbol names</b> Optional	NON_MATCHING_S	[table] Default: [Skip output]	Output table for symbols from the provided style database which do not match any category. One of: <ul style="list-style-type: none"> <li>Skip Output</li> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Non-matching categories</b>	NON_MATCHING_C	[table]	Lists categories which could not be matched to any symbol in the provided style database
<b>Non-matching symbol names</b>	NON_MATCHING_S	[table]	Lists symbols from the provided style database which could not match any category
<b>Categorized layer</b>	OUTPUT	[same as input]	The input vector layer with the categorized style applied. No new layer is output.

## Pythoncode

**Algorithm ID:** native:categorizeusingstyle

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create style database from project

Extracts all style objects (symbols, color ramps, text formats and label settings) from a QGIS project.

The extracted symbols are saved to a QGIS style database (XML format), which can be managed and imported via the [Style Manager](#) dialog.

**Siehe auch:**

[Combine style databases](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input project</b> (leave blank to use current) Optional	INPUT	[file]	A QGIS project file to extract the style items from
<b>Objects to extract</b>	OBJECTS	[enumeration] [list]	Types of style items in the input project you would like to put in the new database. These can be: <ul style="list-style-type: none"> <li>0 — <a href="#">Symbols</a></li> <li>1 — <a href="#">Color ramps</a></li> <li>2 — <a href="#">Text formats</a></li> <li>3 — <a href="#">Label settings</a></li> </ul>
<b>Output style database</b>	OUTPUT	[file] Default: [Save to temporary file]	Specify the output .XML file for the selected style items. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Color ramp count</b>	COLORRAMPS	[number]	Number of color ramps
<b>Label settings count</b>	LABELSETTINGS	[number]	Number of label settings
<b>Output style database</b>	OUTPUT	[file]	Output .XML file for the selected style items
<b>Symbol count</b>	SYMBOLS	[number]	Number of symbols
<b>Text format count</b>	TEXTFORMATS	[number]	Number of text formats

## Pythoncode

**Algorithm ID:** native:stylefromproject

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export atlas layout as image

Exports the atlas of a print layout as image files (e.g. PNG or JPEG images).

If a coverage layer is set, the selected layout's atlas settings exposed in this algorithm will be overwritten. In this case, an empty filter or sort by expression will turn those settings off.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Atlas layout</b>	LAYOUT	[layout]	Layout to export
<b>Coverage layer</b> Optional	COVERAGE_LAYER	[vector: any]	Layer to use to generate the atlas
<b>Filter expression</b>	FILTER_EXPRESS	[expression]	Expression to use to filter out atlas features
<b>Sort expression</b> Optional	SORTBY_EXPRESS	[expression]	Expression to use to sort the atlas features
<b>Reverse sort order</b> Optional	SORTBY_REVERSE	[boolean]	Determines if sorting should be inverted. Used when a sort expression is provided.
<b>Output filename expression</b>	FILENAME_EXPRES	[expression] Default: ,output_'\ @atlas_feat	Expression for use to generate filenames
<b>Output folder</b>	FOLDER	[folder]	Destination folder where the images will be generated



## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map layers to assign to unlocked map item(s)</b> Optional	LAYERS	[enumeration] [layer]	Layers to display in the map item(s) whose contents are not locked
<b>Image format</b>	EXTENSION	[enumeration] Default: png	File format of the generated output(s). The list of available formats varies depending on OS and installed drivers.
<b>DPI</b> Optional	DPI Default: Not set	[number]	DPI of the output file(s). If not set, the value in the print layout settings will be used.
<b>Generate world file</b>	GEOREFERENCE	[boolean] Default: True	Determines if a world file should be generated
<b>Export RDF metadata</b>	INCLUDE_METADATA	[boolean] Default: True	Determines if RDF metadata (title, author, ...) should be generated
<b>Enable antialiasing</b>	ANTIALIAS	[boolean] Default: True	Determines if antialiasing should be enabled

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Image file</b>	OUTPUT	[file]	Image files generated by the atlas layout

## Pythoncode

**Algorithm ID:** native:atlaslayouttoimage

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export atlas layout as PDF (multiple files)

Exports the atlas of a print layout to multiple PDF files.

If a coverage layer is set, the selected layout's atlas settings exposed in this algorithm will be overwritten. In this case, an empty filter or sort by expression will turn those settings off.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Atlas layout</b>	LAYOUT	[layout]	Layout to export
<b>Coverage layer</b> Optional	COVERAGE_LAYER	[vector: any]	Layer to use to generate the atlas
<b>Filter expression</b>	FILTER_EXPRESS	[expression]	Expression to use to filter out atlas features
<b>Sort expression</b> Optional	SORTBY_EXPRESS	[expression]	Expression to use to sort the atlas features
<b>Reverse sort order</b> Optional	SORTBY_REVERSE	[boolean]	Determines if sorting should be inverted. Used when a sort expression is provided.
<b>Output filename</b> Optional	OUTPUT_FILENAM	[expression]	Name pattern of the PDF output files.
<b>Output folder</b>	OUTPUT_FOLDER	[folder]	Destination folder for the output PDF files.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map layers to assign to unlocked map item(s)</b> Optional	LAYERS	[enumeration] [layer]	Layers to display in the map item(s) whose contents are not locked
<b>DPI</b> Optional	DPI Default: Not set	[number]	DPI of the output file(s). If not set, the value in the print layout settings will be used.
<b>Always export as vectors</b>	FORCE_VECTOR	[boolean] Default: False	Determines if vectorial data should be left as vectors
<b>Always export as raster</b>	FORCE_RASTER	[boolean] Default: False	Forces all the items in the map to be rasterized. This parameter takes precedence over the FORCE_VECTOR parameter.
<b>Append georeference information</b>	GEOREFERENCE	[boolean] Default: True	Determines if a world file should be generated
<b>Export RDF metadata</b>	INCLUDE_METADATA	[boolean] Default: True	Determines if RDF metadata (title, author, ...) should be generated
<b>Disable tiled raster layer exports</b>	DISABLE_TILED	[boolean] Default: False	Determines if raster should be tiled
<b>Simplify geometries to reduce output file size</b>	SIMPLIFY	[boolean] Default: True	Determines if geometries should be simplified to reduce output file size
<b>Text export</b>	TEXT_FORMAT	[enumeration] Default: 0	Determines if text should be exported as path or text objects. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Always export text as paths (recommended)</li> <li>• 1 - Always export texts as text objects</li> </ul>
<b>Image compression</b>	IMAGE_COMPRESS	[enumeration] Default: 0	Determines compression level of the image and how suitable the file could be for printing outputs or post-production in external applications. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Lossy (JPEG)</li> <li>• 1 - Lossless</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>PDF file</b>	OUTPUT	[file]	PDF file corresponding to the exported atlas layout

## Pythoncode

**Algorithm ID:** native:atlaslayouttomultiplepdf

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export atlas layout as PDF (single file)

Exports the atlas of a print layout as a single PDF file.

If a coverage layer is set, the selected layout's atlas settings exposed in this algorithm will be overwritten. In this case, an empty filter or sort by expression will turn those settings off.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Atlas layout</b>	LAYOUT	[layout]	Layout to export
<b>Coverage layer</b> Optional	COVERAGE_LAYER	[vector: any]	Layer to use to generate the atlas
<b>Filter expression</b>	FILTER_EXPRESS	[expression]	Expression to use to filter out atlas features
<b>Sort expression</b> Optional	SORTBY_EXPRESS	[expression]	Expression to use to sort the atlas features
<b>Reverse sort order</b> Optional	SORTBY_REVERSE	[boolean]	Determines if sorting should be inverted. Used when a sort expression is provided.
<b>PDF file</b>	OUTPUT	[file] Default: [Save to temporary file]	Name (including path) of the output file. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map layers to assign to unlocked map item(s)</b> Optional	LAYERS	[enumeration] [layer]	Layers to display in the map item(s) whose contents are not locked
<b>DPI</b> Optional	DPI Default: Not set	[number]	DPI of the output file(s). If not set, the value in the print layout settings will be used.
<b>Always export as vectors</b>	FORCE_VECTOR	[boolean] Default: False	Determines if vectorial data should be left as vectors
<b>Always export as raster</b>	FORCE_RASTER	[boolean] Default: False	Forces all the items in the map to be rasterized. This parameter takes precedence over the FORCE_VECTOR parameter.
<b>Append georeference information</b>	GEOREFERENCE	[boolean] Default: True	Determines if a world file should be generated
<b>Export RDF metadata</b>	INCLUDE_METADATA	[boolean] Default: True	Determines if RDF metadata (title, author, ...) should be generated
<b>Disable tiled raster layer exports</b>	DISABLE_TILED	[boolean] Default: False	Determines if raster should be tiled
<b>Simplify geometries to reduce output file size</b>	SIMPLIFY	[boolean] Default: True	Determines if geometries should be simplified to reduce output file size
<b>Text export</b>	TEXT_FORMAT	[enumeration] Default: 0	Determines if text should be exported as path or text objects. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Always export text as paths (recommended)</li> <li>• 1 - Always export texts as text objects</li> </ul>
<b>Image compression</b>	IMAGE_COMPRESS	[enumeration] Default: 0	Determines compression level of the image and how suitable the file could be for printing outputs or post-production in external applications. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Lossy (JPEG)</li> <li>• 1 - Lossless</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>PDF file</b>	OUTPUT	[file]	PDF file corresponding to the exported atlas layout

## Pythoncode

**Algorithm ID:** native:atlaslayouttopdf

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export print layout as image

Exports a print layout as an image file (e.g. PNG or JPEG images)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Print layout</b>	LAYOUT	[layout]	Layout to export
<b>Image file</b>	OUTPUT	[file] Default: [Save to temporary file]	Name (including path) of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map layers to assign to unlocked map item(s)</b> Optional	LAYERS	[enumeration] [layer]	Layers to display in the map item(s) whose contents are not locked
<b>DPI</b> Optional	DPI Default: Not set	[number]	DPI of the output file(s). If not set, the value in the print layout settings will be used.
<b>Generate world file</b>	GEOREFERENCE	[boolean] Default: True	Determines if a world file should be generated
<b>Export RDF metadata</b>	INCLUDE_METADATA	[boolean] Default: True	Determines if RDF metadata (title, author, ...) should be generated
<b>Enable antialiasing</b>	ANTIALIAS	[boolean] Default: True	Determines if antialiasing should be enabled

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Image file</b>	OUTPUT	[file]	Image file corresponding to the exported print layout

## Pythoncode

**Algorithm ID:** native:printlayouttoimage

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export print layout as PDF

Exports a print layout as a PDF file.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Print Layout</b>	LAYOUT	[layout]	Layout to export
<b>PDF file</b>	OUTPUT	[file] Default: [Save to temporary file]	Name (including path) of the output file. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map layers to assign to unlocked map item(s)</b> Optional	LAYERS	[enumeration] [layer]	Layers to display in the map item(s) whose contents are not locked
<b>DPI</b> Optional	DPI Default: Not set	[number]	DPI of the output file(s). If not set, the value in the print layout settings will be used.
<b>Always export as vectors</b>	FORCE_VECTOR	[boolean] Default: False	Determines if vectorial data should be left as vectors
<b>Always export as raster</b>	FORCE_RASTER	[boolean] Default: False	Forces all the items in the map to be rasterized. This parameter takes precedence over the FORCE_VECTOR parameter.
<b>Append georeference information</b>	GEOREFERENCE	[boolean] Default: True	Determines if a world file should be generated
<b>Export RDF metadata</b>	INCLUDE_METADATA	[boolean] Default: True	Determines if RDF metadata (title, author, ...) should be generated
<b>Disable tiled raster layer exports</b>	DISABLE_TILED	[boolean] Default: False	Determines if raster should be tiled
<b>Simplify geometries to reduce output file size</b>	SIMPLIFY	[boolean] Default: True	Determines if geometries should be simplified to reduce output file size
<b>Text export</b>	TEXT_FORMAT	[enumeration] Default: 0	Determines if text should be exported as path or text objects. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Always export text as paths (recommended)</li> <li>• 1 - Always export texts as text objects</li> </ul>
<b>Image compression</b>	IMAGE_COMPRESS	[enumeration] Default: 0	Determines compression level of the image and how suitable the file could be for printing outputs or post-production in external applications. Possible options are: <ul style="list-style-type: none"> <li>• 0 - Lossy (JPEG)</li> <li>• 1 - Lossless</li> </ul>
<b>Export layers as separate PDF files</b>	SEPARATE_LAYER	[boolean] Default: False	If True, then a separate PDF file will be created per layer per map item in the layout. Additionally, separate PDF files may be created for other complex layout items, resulting in a set of PDF files which contain logical atomic components of the layout.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
PDF file	OUTPUT	[file]	PDF file(s) corresponding to the exported print layout

## Pythoncode

**Algorithm ID:** native:printlayouttopdf

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract labels

Extracts label information from a rendered map at a given extent and scale.

If a map theme is provided, the rendered map will match the visibility and symbology of that theme. If left blank, all visible layers from the project will be used. Extracted label information include: position (served as point geometries), the associated layer name and feature ID, label text, rotation (in degree, clockwise), multiline alignment, and font details.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map extent</b>	EXTENT	[extent]	Extent of the map to extract the labels from Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Map scale</b>	SCALE	[scale]	Extracted labels will be rendered using their properties set at this scale.
<b>Map theme</b> Optional	MAP_THEME	[maptheme]	A map theme displaying the layers to extract the labels from. If unset, labels of the currently visible layers are extracted.
<b>Include unplaced labels</b> Optional	INCLUDE_UNPLAC	[boolean] Default: True	Specify whether all overlapping labels should be extracted, including the conflicting (thus unplaced) ones.
<b>Extracted labels</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer for the extent(s). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Map resolution (in DPI)</b> Optional	DPI Default: 96.0	[number]	

Ausgaben

Bezeichnung	Name	Typ	Beschreibung
Extracted labels	OUTPUT	[vector: point]	<p>Point vector layer representing the fetched labels. Each feature has attributes identifying its source (layer, feature ID) and the assigned labeling properties (text, font, size, rotation, ...). A default style with labeling and null symbol is also applied to the layer.</p> <div><p><b>Warnung:</b> Because some of the generated fields have name with more than 10 characters, using the ESRI shapefile format (SHP) to store the output may lead to unexpected rendering while loading the layer in QGIS.</p></div>

## Pythoncode

**Algorithm ID:** native:extractlabels

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Print layout map extent to layer

Creates a polygon layer containing the extent of a print layout map item (or items), with attributes specifying the map size (in layout units, i.e. the *reference map* units), scale and rotation.

If the map item parameter is specified, then only the matching map extent will be exported. If it is not specified, all map extents from the layout will be exported.

Optionally, a specific output CRS can be specified. If it is not specified, the original map item CRS will be used.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Print layout</b>	LAYOUT	[enumeration]	A print layout in the current project
<b>Map item</b> Optional	MAP	[enumeration] Default: <i>All the map items</i>	The map item(s) whose information you want to extract. If none is provided then all the map items are processed.
<b>Extent</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer for the extent(s). One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Override CRS</b> Optional	CRS	[crs] Default: <i>The layout CRS</i>	Select the CRS for the layer in which the information will be reported.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Map height</b>	HEIGHT	[number]	Output polygon vector layer containing extents of all the input layout map item(s)
<b>Extent</b>	OUTPUT	[vector: polygon]	
<b>Map rotation</b>	ROTATION	[number]	
<b>Map scale</b>	SCALE	[number]	
<b>Map width</b>	WIDTH	[number]	

## Pythoncode

**Algorithm ID:** native:printlayoutmapextenttolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Set layer style

Applies a provided style to a layer. The style must be defined in a QML file.

No new output are created: the style is immediately assigned to the layer.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[layer]	Input layer you want to apply the style to
<b>Style file</b>	STYLE	[file]	Path to the .qml file of the style

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
	OUTPUT	[same as input]	The input layer with the new style assigned. No new layer is created.

## Pythoncode

**Algorithm ID:** native:setlayerstyle

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Topological coloring

Assigns a color index to polygon features in such a way that no adjacent polygons share the same color index, whilst minimizing the number of colors required.

The algorithm allows choice of method to use when assigning colors.

A minimum number of colors can be specified if desired. The color index is saved to a new attribute named **color\_id**.

The following example shows the algorithm with four different colors chosen; as you can see each color class has the same amount of features.

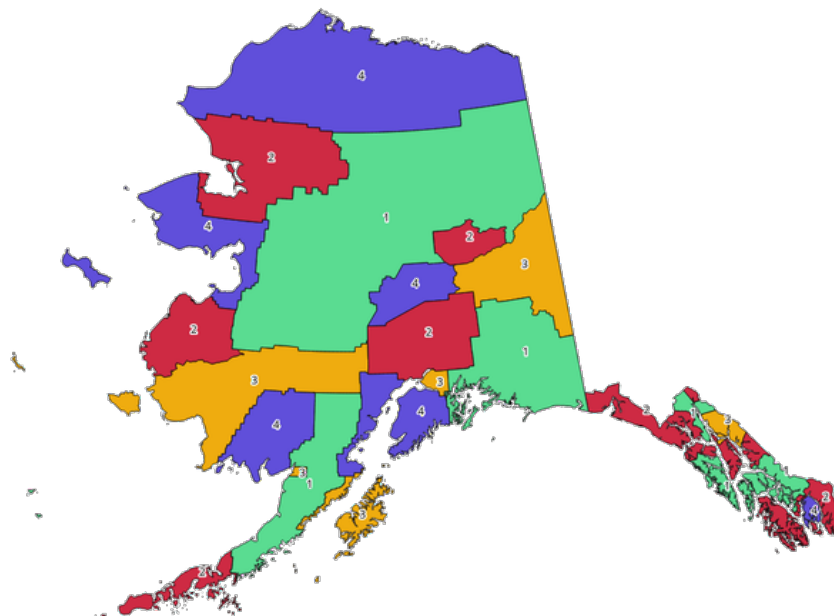


Abb. 28.1: Topological colors example

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	The input polygon layer
<b>Minimum number of colors</b>	MIN_COLORS	[number] Default: 4	The minimum number of colors to assign. Minimum 1, maximum 1000.
<b>Minimum distance between features</b>	MIN_DISTANCE	[number] Default: 0.0	Prevent nearby (but non-touching) features from being assigned equal colors. Minimum 0.0.

Fortsetzung auf der nächsten Seite

Tab. 28.4 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Balance color assignment</b>	BALANCE	[enumeration] Default: 0	Options are: <ul style="list-style-type: none"> <li>• 0 — By feature count Attempts to assign colors so that the count of features assigned to each individual color index is balanced.</li> <li>• 1 — By assigned area Assigns colors so that the total area of features assigned to each color is balanced. This mode can be useful to help avoid large features resulting in one of the colors appearing more dominant on a colored map.</li> <li>• 2 — By distance between colors Assigns colors in order to maximize the distance between features of the same color. This mode helps to create a more uniform distribution of colors across a map.</li> </ul>
<b>Colored</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Colored</b>	OUTPUT	[vector: polygon]	Polygon vector layer with an added color_id column

## Pythoncode

**Algorithm ID:** qgis:topologicalcoloring

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Transfer annotations from main layer

Transfers all *annotations* from the main annotation layer in a project to a new annotation layer. Items placement can then be adjusted within the layer stack.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
New layer name	LAYER_NAME	[string] Default: ,Annotations'	Name of the annotations layer to create

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
New layer name	OUTPUT	[layer]	A layer with items from the main annotation layer

### Pythoncode

**Algorithm ID:** native:transferannotationsfrommain

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.3 Database

### Export to PostgreSQL

Exports a vector layer to a PostgreSQL database, creating a new relation. If a relation with the same name exists, it can be removed before the new relation is created. Prior to this a connection between QGIS and the PostgreSQL database has to be created (see eg [Creating a stored Connection](#)).

### Parameters

Label	Name	Type	Description
Layer to import	INPUT	[vector: any]	Vector layer to add to the database
Database (connection name)	DATABASE	[string]	Name of the database connection (not the database name). Existing connections will be shown in the combobox.

Fortsetzung auf der nächsten Seite



Tab. 28.6 – Fortsetzung der vorherigen Seite

Label	Name	Type	Description
<b>Schema (schema name)</b> Optional	SCHEMA	[string] Default: ‚public‘	Name of the schema to store the data. It can be a new one or already exist.
<b>Table to import to (leave blank to use layer name)</b> Optional	TABLERNAME	[string] Default: ‚‘	Defines a table name for the imported vector file. If nothing is added, the layer name will be used.
<b>Primary key field</b> Optional	PRIMARY_KEY	[tablefield: any]	Sets the primary key field from an existing field in the vector layer. A column with <b>unique</b> values can be used as Primary key for the database.
<b>Geometry column</b>	GEOMETRY_COLUMN	[string] Default: ‚geom‘	Defines the name of the geometry column in the new PostGIS table. Geometry information for the features is stored in this column.
<b>Encoding</b> Optional	ENCODING	[string] Default: ‚UTF-8‘	Defines the encoding of the output layer
<b>Overwrite</b>	OVERWRITE	[boolean] Default: True	If the specified table exists, setting this option to <code>True</code> will make sure that it is deleted and a new table will be created before the features are added. If this option is <code>False</code> and the table exists, the algorithm will throw an exception („relation already exists“).
<b>Create spatial index</b>	CREATEINDEX	[boolean] Default: True	Specifies whether to create a spatial index or not
<b>Convert field names to lowercase</b>	LOWERCASE_NAME	[boolean] Default: True	Converts the field names of the input vector layer to lowercase
<b>Drop length constraint on character fields</b>	DROP_STRING_LE	[boolean] Default: False	Should length constraints on character fields be dropped or not
<b>Create single-part geometries instead of multi-part</b>	FORCE_SINGLEPA	[boolean] Default: False	Should the features of the output layer be single-part instead of multi-part. By default the existing geometries information are preserved.

## Ausgaben

The algorithm has no output.

## Pythoncode

**Algorithm ID:** `qgis:importintopostgis`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export to SpatiaLite

Exports a vector layer to a SpatiaLite database. Prior to this a connection between QGIS and the SpatiaLite database has to be created (see eg [SpatiaLite Layers](#)).

### Parameters

Label	Name	Type	Description
<b>Layer to import</b>	INPUT	[vector: any]	Vector layer to add to the database
<b>File database</b>	DATABASE	[vector: any]	The SQLite/SpatiaLite database file to connect to
<b>Table to import to (leave blank to use layer name)</b> Optional	TABLERNAME	[string] Default: ;	Defines the table name for the imported vector file. If nothing is specified, the layer name will be used.
<b>Primary key field</b> Optional	PRIMARY_KEY	[tablefield: any]	Use a field in the input vector layer as the primary key
<b>Geometry column</b>	GEOMETRY_COLUMN	[string] Default: 'geom'	Defines the name of the geometry column in the new SpatiaLite table. Geometry information for the features is stored in this column.
<b>Encoding</b> Optional	ENCODING	[string] Default: 'UTF-8'	Defines the encoding of the output layer
<b>Overwrite</b>	OVERWRITE	[boolean] Default: True	If the specified table exists, setting this option to <code>True</code> will make sure that it is deleted and a new table will be created before the features of the layer is added. If this option is <code>False</code> and the table exists, the algorithm will throw an exception („table already exists“).
<b>Create spatial index</b>	CREATEINDEX	[boolean] Default: True	Specifies whether to create a spatial index or not
<b>Convert field names to lowercase</b>	LOWERCASE_NAME	[boolean] Default: True	Convert the field names of the input vector layer to lowercase
<b>Drop length constraint on character fields</b>	DROP_STRING_LENGTH	[boolean] Default: False	Should length constraints on character fields be dropped or not
<b>Create single-part geometries instead of multi-part</b>	FORCE_SINGLEPART	[boolean] Default: False	Should the features of the output layer be single-part instead of multi-part. By default the existing geometries information are preserved.

## Ausgaben

The algorithm has no output.

## Pythoncode

**Algorithm ID:** qgis:importintospatialite

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Package layers

Adds layers to a GeoPackage.

If the GeoPackage exists and **Overwrite existing GeoPackage** is checked, it will be overwritten (removed and recreated). If the GeoPackage exists and **Overwrite existing GeoPackage** is not checked, the layer will be appended.

## Parameters

Label	Name	Type	Description
<b>Input layers</b>	LAYERS	[vector: any] [list]	The (vector) layers to import into the GeoPackage. Raster layers are not supported. If a raster layer is added, a <code>QgsProcessingException</code> will be thrown.
<b>Overwrite existing GeoPackage</b>	OVERWRITE	[boolean] Default: False	If the specified GeoPackage exists, setting this option to <code>True</code> will make sure that it is deleted and a new one will be created before the layers are added. If set to <code>False</code> , layers will be appended.
<b>Save layer styles into GeoPackage</b>	SAVE_STYLES	[boolean] Default: True	Save the layer styles
<b>Save only selected features</b>	SELECTED_FEATU	[boolean] Default: False	If a layer has a selection, setting this option to <code>True</code> will result in only selected features being saved. For layers without a selection all features will be saved.
<b>Export related layers following relations defined in the project</b>	EXPORT_RELATED	[boolean] Default: False	If an input layer has <i>relations</i> set in the project, setting this option to <code>True</code> will result in exporting also its related layer(s). If the layer has features selected, then only their related features will be exported unless the related layer was also an input layer.
<b>Destination GeoPackage</b>	OUTPUT	[file] Default: [Save to temporary file]	Specify where to store the GeoPackage file. One of <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Label	Name	Type	Description
Layers within new package	OUTPUT_LAYERS	[string] [list]	The list of layers added to the GeoPackage.

## Pythoncode

**Algorithm ID:** native:package

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## PostgreSQL execute and load SQL

Allows a SQL database query to be performed on a PostgreSQL database connected to QGIS and loads the result. The algorithm **won't** create a new layer: it is designed to run queries on the layer itself.

### Example

1. Set all the values of an existing field to a fixed value. The SQL query string will be:

```
UPDATE your_table SET field_to_update=20;
```

In the example above, the values of the field `field_to_update` of the table `your_table` will be all set to 20.

2. Create a new `area` column and calculate the area of each feature with the `ST_AREA` PostGIS function.

```
-- Create the new column "area" on the table your_table"
ALTER TABLE your_table ADD COLUMN area double precision;
-- Update the "area" column and calculate the area of each feature:
UPDATE your_table SET area=ST_AREA(geom);
```

### Siehe auch:

[PostgreSQL execute SQL](#), [Execute SQL](#), [Spatialite execute SQL](#)

## Parameters

Label	Name	Type	Description
<b>Database (connection name)</b>	DATABASE	[string]	The database connection (not the database name). Existing connections will be shown in the combobox.
<b>SQL query</b>	SQL	[string]	Defines the SQL query, for example 'UPDATE my_table SET field=10'.
<b>Unique ID field name</b>	ID_FIELD	[string] Default: id	Sets the primary key field (a column in the result table)
<b>Geometry field name</b> Optional	GEOMETRY_FIELD	[string] Default: 'geom'	Name of the geometry column (a column in the result table)

## Ausgaben

Label	Name	Type	Description
<b>SQL layer</b>	OUTPUT	[vector: any]	The resulting vector layer to be loaded into QGIS.

## Pythoncode

**Algorithm ID:** qgis:postgisexecuteandloadsql

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## PostgreSQL execute SQL

Allows a SQL database query to be performed on a PostgreSQL database connected to QGIS. The algorithm **won't** create a new layer: it is designed to run queries on the layer itself.

### Example

1. Set all the values of an existing field to a fixed value. The SQL query string will be:

```
UPDATE your_table SET field_to_update=20;
```

In the example above, the values of the field `field_to_update` of the table `your_table` will be all set to 20.

2. Create a new `area` column and calculate the area of each feature with the `ST_AREA` PostGIS function.

```
-- Create the new column "area" on the table your_table
ALTER TABLE your_table ADD COLUMN area double precision;
-- Update the "area" column and calculate the area of each feature:
UPDATE your_table SET area=ST_AREA(geom);
```

**Siehe auch:**

*PostgreSQL execute and load SQL, Execute SQL, SpatiaLite execute SQL*

**Parameters**

Label	Name	Type	Description
<b>Database (connection name)</b>	DATABASE	[string]	The database connection (not the database name). Existing connections will be shown in the combobox.
<b>SQL query</b>	SQL	[string]	Defines the SQL query, for example 'UPDATE my_table SET field=10'.

**Ausgaben**

No output is created. The SQL query is executed in place.

**Pythoncode**

**Algorithm ID:** native:postgisexecutesql

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

**SpatiaLite execute SQL**

Allows a SQL database query to be performed on a SpatiaLite database. The algorithm **won't** create a new layer: it is designed to run queries on the layer itself.

**Siehe auch:**

*PostgreSQL execute SQL, Execute SQL*

For some SQL query examples see *PostGIS SQL Query Examples*.

**Parameters**

Label	Name	Type	Description
<b>File Database</b>	DATABASE	[vector]	The SQLite/SpatiaLite database file to connect to
<b>SQL query</b>	SQL	[string] Default: ;	Defines the SQL query, for example 'UPDATE my_table SET field=10'.

## Ausgaben

No output is created. The SQL query is executed in place.

## Pythoncode

**Algorithm ID:** native:spatialiteexecutesql

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Spatialite execute SQL (registered DB)

Allows a SQL database query to be performed on a Spatialite database connected to QGIS. The algorithm **won't** create a new layer: it is designed to run queries on the layer itself.

**Siehe auch:**

[PostgreSQL execute SQL](#), [Execute SQL](#)

For some SQL query examples see [PostGIS SQL Query Examples](#).

## Parameters

Label	Name	Type	Description
<b>Database</b>	DATABASE	[enumeration] Default: not set	Select a SQLite/Spatialite database connected to the current session
<b>SQL query</b>	SQL	[string] Default: ;	Defines the SQL query, for example 'UPDATE my_table SET field=10'.

## Ausgaben

No output is created. The SQL query is executed in place.

## Pythoncode

**Algorithm ID:** native:spatialiteexecutesqlregistered

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.4 File tools

### Download file

Downloads a file specified using a URL (using for instance `http:` or `file:`). In other words you can copy/paste a URL and download the file.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>URL</b>	URL	[string]	The URL of the file to download.
<b>File destination</b> Optional	OUTPUT	[string] Default: [Save to temporary file]	Specification of the file destination. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Method</b>	METHOD	[enumeration] Default: 0	The HTTP method to use for the request. Options are: <ul style="list-style-type: none"><li>• 0 — GET</li><li>• 1 — POST</li></ul>
<b>Data</b> Optional	DATA	[string]	The data to add in the body if the request is a POST.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>File destination</b>	OUTPUT	[string]	The location of the downloaded file

### Pythoncode

**Algorithm ID:** `qgis:filedownloader`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## 28.1.5 GPS

### Convert GPS data

Uses the [GPSBabel tool](#) to convert a GPS data file from a range of formats to the GPX standard format.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input file</b>	INPUT	[file]	File containing the data to convert
<b>Format</b>	FORMAT	[enumeration]	Format of the file to convert, from <a href="#">this list</a> .
<b>Feature type</b>	FEATURE_TYPE	[enumeration] Default: 0	The type of data to convert <ul style="list-style-type: none"> <li>• 0 — Waypoints</li> <li>• 1 — Routes</li> <li>• 2 — Tracks</li> </ul>
<b>Ergebnis</b>	OUTPUT	[vector: any] Default: [Save to temporary file]	Specification of the output GPX file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT_LAYER	[vector: any]	Output layer with data in GPX standard format

#### Pythoncode

**Algorithm ID:** native:convertgpsdata

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert GPX feature type

Uses the [GPSTBabel](#) tool to convert GPX features from one type to another (e.g. converting all waypoint features to a route).

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input file</b>	INPUT	[file]	File containing the data to convert
<b>Conversion</b>	CONVERSION	[enumeration] Default: 0	The type of conversion to apply <ul style="list-style-type: none"><li>• 0 — Waypoints from a route</li><li>• 1 — Waypoints from a track</li><li>• 2 — Routes from waypoints</li><li>• 3 — Tracks from waypoints</li></ul>
<b>Ergebnis</b>	OUTPUT	[vector: point or line] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[vector: any]	Output layer with converted GPX features

### Pythoncode

**Algorithm ID:** native:convertgpxfeaturetype

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Download GPS data from device

Uses the [GPSTabel tool](#) to download data from a GPS device into the GPX standard format.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Device</b>	DEVICE	[enumeration] Default: <i>Garmin serial</i>	The GPS device used to create the data. Must be declared in <a href="#">GPS Settings</a> dialog.
<b>Port</b>	PORT	[enumeration]	The port the device is connected to. Available ports depend on the OS.
<b>Feature type</b>	FEATURE_TYPE	[enumeration] Default: 0	The type of data to convert <ul style="list-style-type: none"> <li>• 0 — Waypoints</li> <li>• 1 — Routes</li> <li>• 2 — Tracks</li> </ul>
<b>Ergebnis</b>	OUTPUT	[vector: any] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT_LAYER	[vector: any]	Output layer with data in GPX standard format

### Pythoncode

**Algorithm ID:** native:downloadgpsdata

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Upload GPS data to device

Uses the [GPSTBabel tool](#) to upload data to a GPS device from the GPX standard format.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input file</b>	INPUT	[file]	. GPX file containing the data to upload
<b>Device</b>	DEVICE	[enumeration] Default: <i>Garmin serial</i>	The GPS device you would like to upload the data to. Must be declared in <a href="#">GPS Settings</a> dialog.
<b>Port</b>	PORT	[enumeration]	The port the device is connected to. Available ports depend on the OS.
<b>Feature type</b>	FEATURE_TYPE	[enumeration] Default: 0	The type of data to upload <ul style="list-style-type: none"><li>• 0 — Waypoints</li><li>• 1 — Routes</li><li>• 2 — Tracks</li></ul>

### Ausgaben

No output is provided. If successful, data are loaded to the device.

### Pythoncode

**Algorithm ID:** native:uploadgpsdata

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.6 Interpolation

### Heatmap (kernel density estimation)

Creates a density (heatmap) raster of an input point vector layer using kernel density estimation.

The density is calculated based on the number of points in a location, with larger numbers of clustered points resulting in larger values. Heatmaps allow easy identification of *hotspots* and clustering of points.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Point vector layer to use for the heatmap
<b>Radius</b>	RADIUS	[number] Default: 100.0	Heatmap search radius (or kernel bandwidth) in map units. The radius specifies the distance around a point at which the influence of the point will be felt. Larger values result in greater smoothing, but smaller values may show finer details and variation in point density.
<b>Output raster size</b>	PIXEL_SIZE	[number] Default: 0.1	Pixel size of the output raster layer in layer units.  In the GUI, the size can be specified by the number of rows (Number of rows) / columns (Number of columns) <b>or</b> the pixel size (Pixel Size X / Pixel Size Y). Increasing the number of rows or columns will decrease the cell size and increase the file size of the output raster. The values in Rows, Columns, Pixel Size X and Pixel Size Y will be updated simultaneously - doubling the number of rows will double the number of columns, and the cell size will be halved. The extent of the output raster will remain the same (approximately).
<b>Radius from field</b> Optional	RADIUS_FIELD	[tablefield: numeric]	Sets the search radius for each feature from an attribute field in the input layer.
<b>Weight from field</b> Optional	WEIGHT_FIELD	[tablefield: numeric]	Allows input features to be weighted by an attribute field. This can be used to increase the influence certain features have on the resultant heatmap.
<b>Kernel shape</b>	KERNEL	[enumeration] Vorgabe: 0	Controls the rate at which the influence of a point decreases as the distance from the point increases. Different kernels decay at different rates, so a triweight kernel gives features greater weight for distances closer to the point than the Epanechnikov kernel does. Consequently, triweight results in “sharper” hotspots and Epanechnikov results in “smoother” hotspots.  There are many shapes available (please see the <a href="#">Wikipedia page</a> for further information): <ul style="list-style-type: none"> <li>• 0 — Quartic</li> <li>• 1 — Triangular</li> <li>• 2 — Uniform</li> <li>• 3 — Triweight</li> <li>• 4 — Epanechnikov</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.15 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Decay ratio</b> (Triangular kernels only) Optional	DECAY	[number] Default: 0.0	Can be used with Triangular kernels to further control how heat from a feature decreases with distance from the feature. <ul style="list-style-type: none"> <li>• A value of 0 (=minimum) indicates that the heat will be concentrated in the center of the given radius and completely extinguished at the edge.</li> <li>• A value of 0.5 indicates that pixels at the edge of the radius will be given half the heat as pixels at the center of the search radius.</li> <li>• A value of 1 means the heat is spread evenly over the whole search radius circle. (This is equivalent to the 'Uniform' kernel.)</li> <li>• A value greater than 1 indicates that the heat is higher towards the edge of the search radius than at the center.</li> </ul>
<b>Output scaling</b>	value	OUTPUT_VALUE [enumeration] Default: Raw	Allow to change the values of the output heatmap raster. One of: <ul style="list-style-type: none"> <li>• 0 — Raw</li> <li>• 1 — Scaled</li> </ul>
<b>Heatmap</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with kernel density values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Heatmap</b>	OUTPUT	[raster]	Raster layer with kernel density values

## Example: Creating a Heatmap

For the following example, we will use the `airports` vector point layer from the QGIS sample dataset (see [Beispieldaten herunterladen](#)). Another excellent QGIS tutorial on making heatmaps can be found at <http://qgistutorials.com>.

In Abb. 28.2, the airports of Alaska are shown.

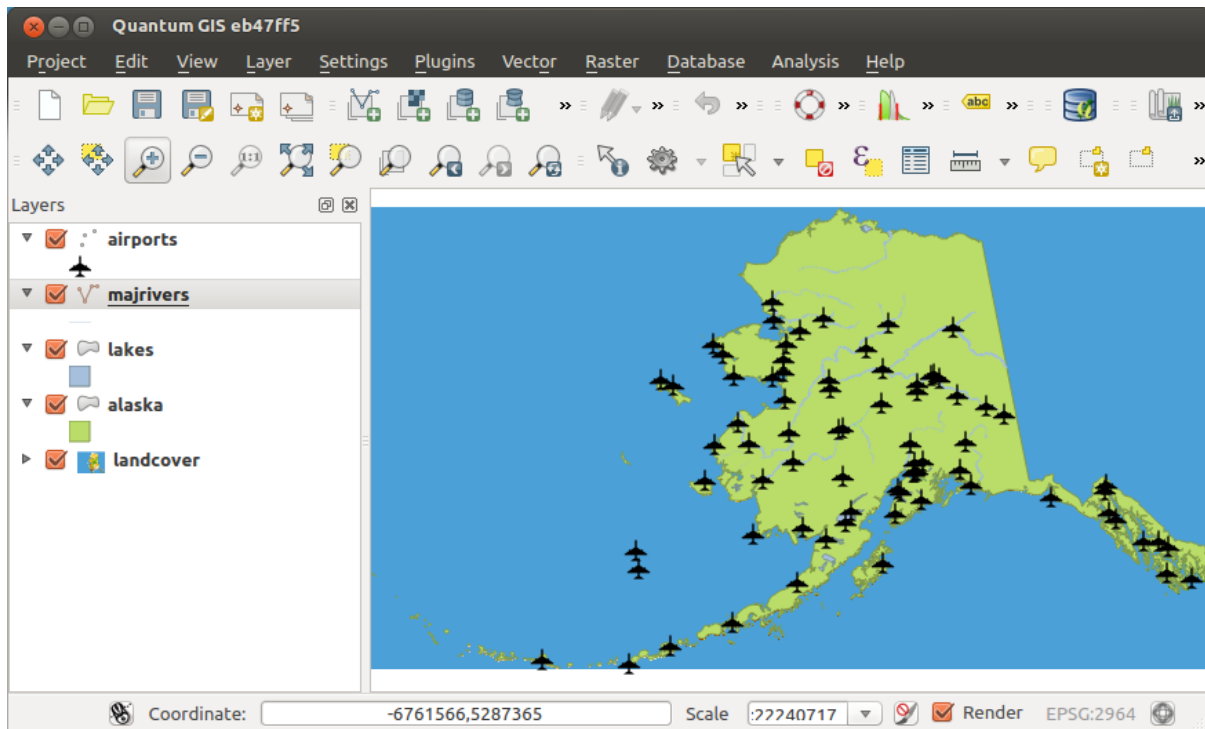



Abb. 28.2: Airports of Alaska

1. Open the *Heatmap (Kernel Density Estimation)* algorithm from the QGIS *Interpolation* group
2. In the *Point layer*  field, select `airports` from the list of point layers loaded in the current project.
3. Change the *Radius* to 1000000 meters.
4. Change the *Pixel size X* to 1000. The *Pixel size Y*, *Rows* and *Columns* will be automatically updated.
5. Click on *Run* to create and load the airports heatmap (see Abb. 28.4).

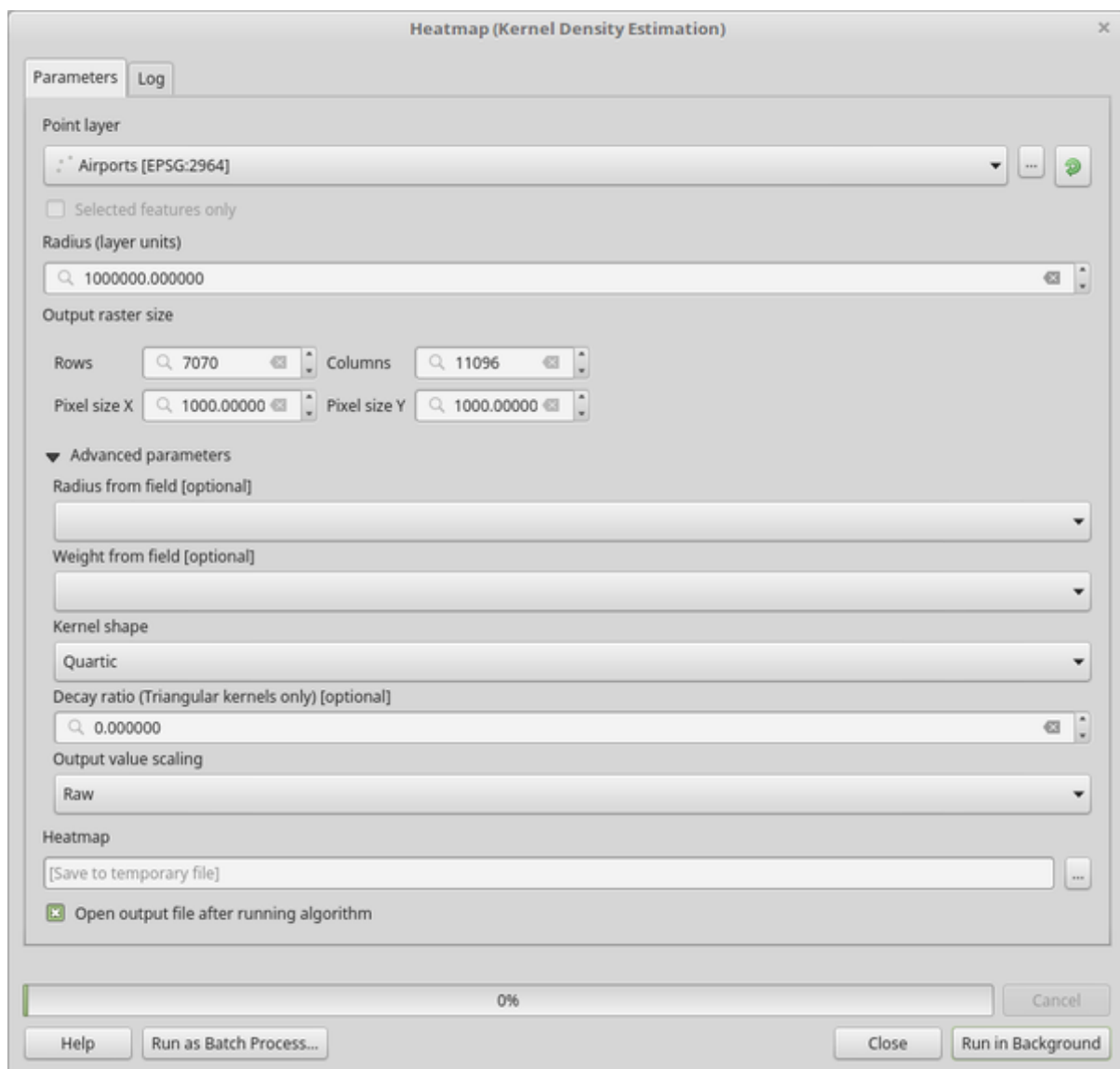


Abb. 28.3: The Heatmap Dialog

QGIS will generate the heatmap and add it to your map window. By default, the heatmap is shaded in greyscale, with lighter areas showing higher concentrations of airports. The heatmap can now be styled in QGIS to improve its appearance.



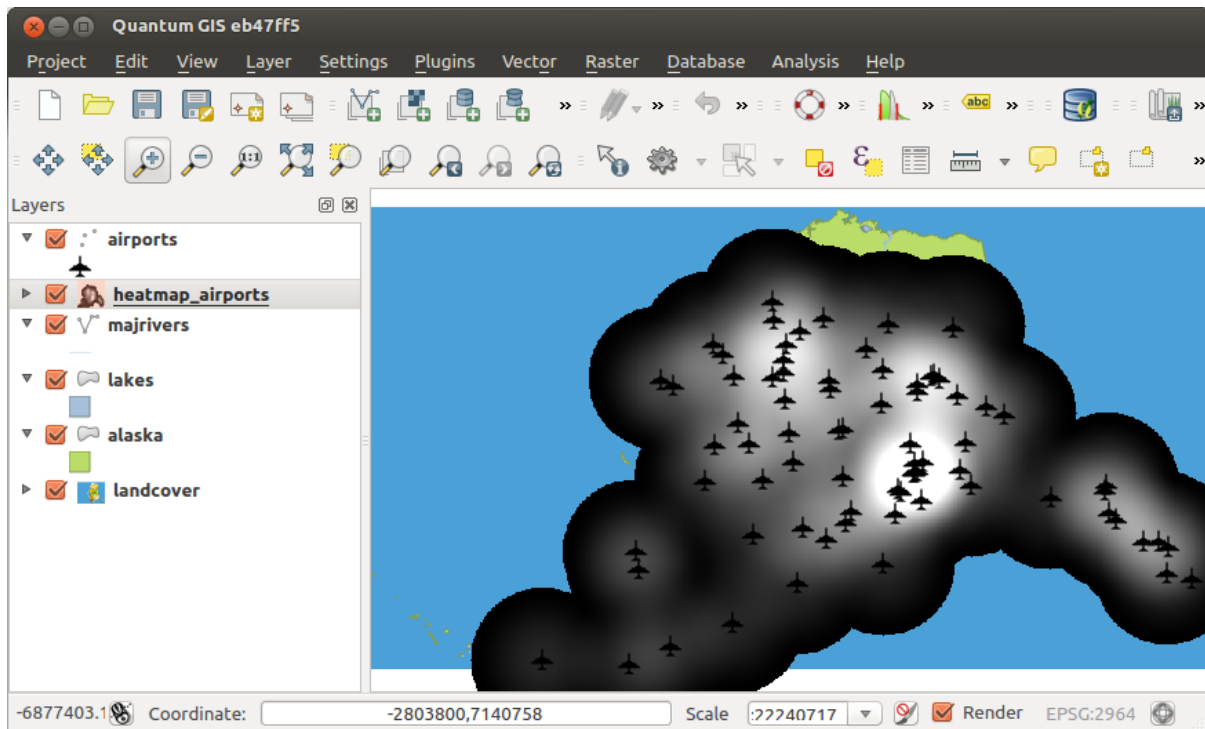




Abb. 28.4: The heatmap after loading looks like a grey surface

1. Open the properties dialog of the `heatmap_airports` layer (select the layer `heatmap_airports`, open the context menu with the right mouse button and select *Properties*).
2. Select the *Symbol* tab.
3. Change the *Render type*  to 'Singleband pseudocolor'.
4. Select a suitable *Color ramp* , for instance `YlOrRd`.
5. Click the *Classify* button.
6. Press *OK* to update the layer.

The final result is shown in [Abb. 28.5](#).

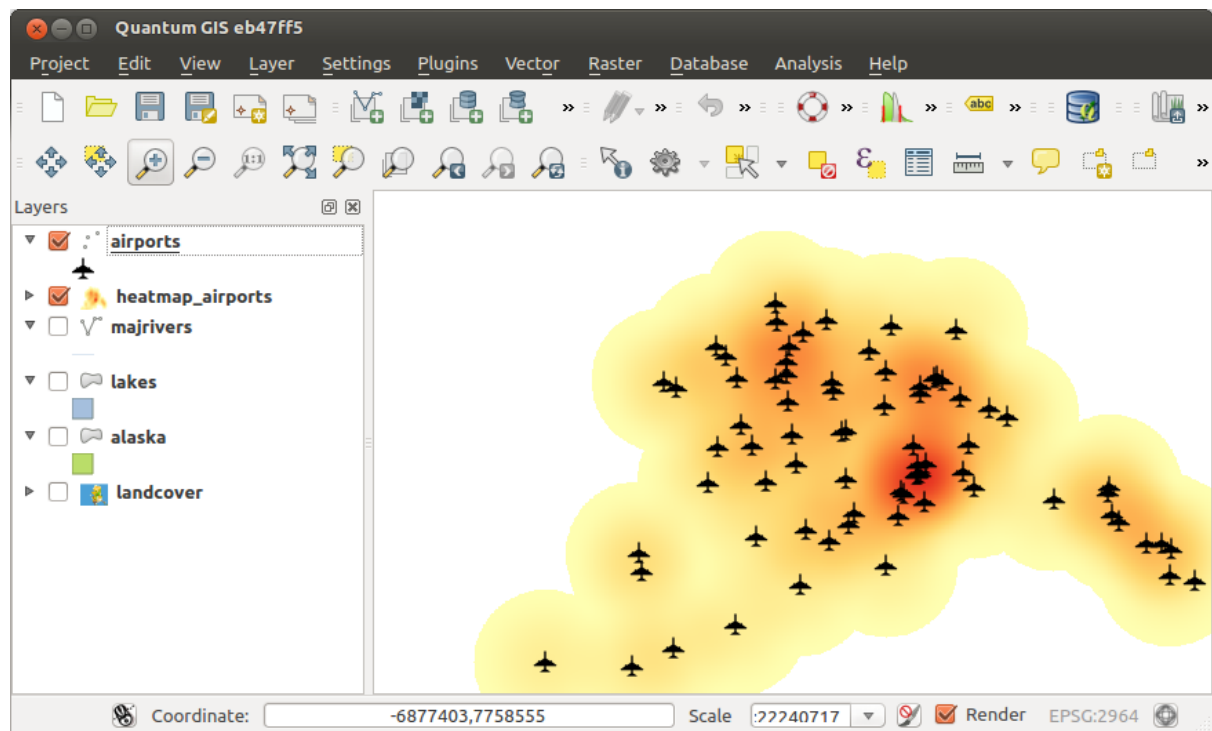


Abb. 28.5: Styled heatmap of airports of Alaska

## Pythoncode

**Algorithm ID:** qgis:heatmapkerneldensityestimation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## IDW Interpolation

Generates an Inverse Distance Weighted (IDW) interpolation of a point vector layer.

Sample points are weighted during interpolation such that the influence of one point relative to another declines with distance from the unknown point you want to create.

The IDW interpolation method also has some disadvantages: the quality of the interpolation result can decrease, if the distribution of sample data points is uneven.

Furthermore, maximum and minimum values in the interpolated surface can only occur at sample data points.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer(s)</b>	INTERPOLATION_	[string]	<p>Vector layer(s) and field(s) to use for the interpolation, coded in a string (see the <code>ParameterInterpolationData</code> class in <a href="#">InterpolationWidgets</a> for more details).</p> <p>The following GUI elements are provided to compose the interpolation data string:</p> <ul style="list-style-type: none"> <li>• <b>Vector layer</b> [vector: any]</li> <li>• <b>Interpolation attribute</b> [tablefield: numeric]: Attribute to use in the interpolation</li> <li>• <b>Use Z-coordinate for interpolation</b> [boolean]: Uses the layer's stored Z values (Default: False)</li> </ul> <p>For each of the added layer-field combinations, a type can be chosen:</p> <ul style="list-style-type: none"> <li>• <i>Points</i></li> <li>• <i>Structured lines</i></li> <li>• <i>Break lines</i></li> </ul> <p>In the string, the layer-field elements are separated by ' : :   : : '. The sub-elements of the layer-field elements are separated by ' : : ~ : : '.</p>
<b>Distance coefficient P</b>	DISTANCE_COEFF	[number] Default: 2.0	Sets the distance coefficient for the interpolation. Minimum: 0.0, maximum: 100.0.
<b>Extent xmin, xmax, ymin, ymax)</b>	(xmin, ymin, EXTENT	[extent]	<p>Extent of the output raster layer.</p> <p>Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.17 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster size</b>	PIXEL_SIZE	[number] Default: 0.1	Pixel size of the output raster layer in layer units.  In the GUI, the size can be specified by the number of rows (Number of rows) / columns (Number of columns) <b>or</b> the pixel size (Pixel Size X / Pixel Size Y). Increasing the number of rows or columns will decrease the cell size and increase the file size of the output raster. The values in Rows, Columns, Pixel Size X and Pixel Size Y will be updated simultaneously - doubling the number of rows will double the number of columns, and the cell size will be halved. The extent of the output raster will remain the same (approximately).
<b>Interpolated</b>	OUTPUT	[raster] Default: [Save to temporary file]	Raster layer of interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated</b>	OUTPUT	[raster]	Raster layer of interpolated values

## Pythoncode

**Algorithm ID:** qgis:gdalinterpolation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Line Density

Calculates for each raster cell, the density measure of linear features within a circular neighbourhood. This measure is obtained by summing all the line segments intersecting the circular neighbourhood and dividing this sum by the area of such neighbourhood. A weighting factor can be applied to the line segments.

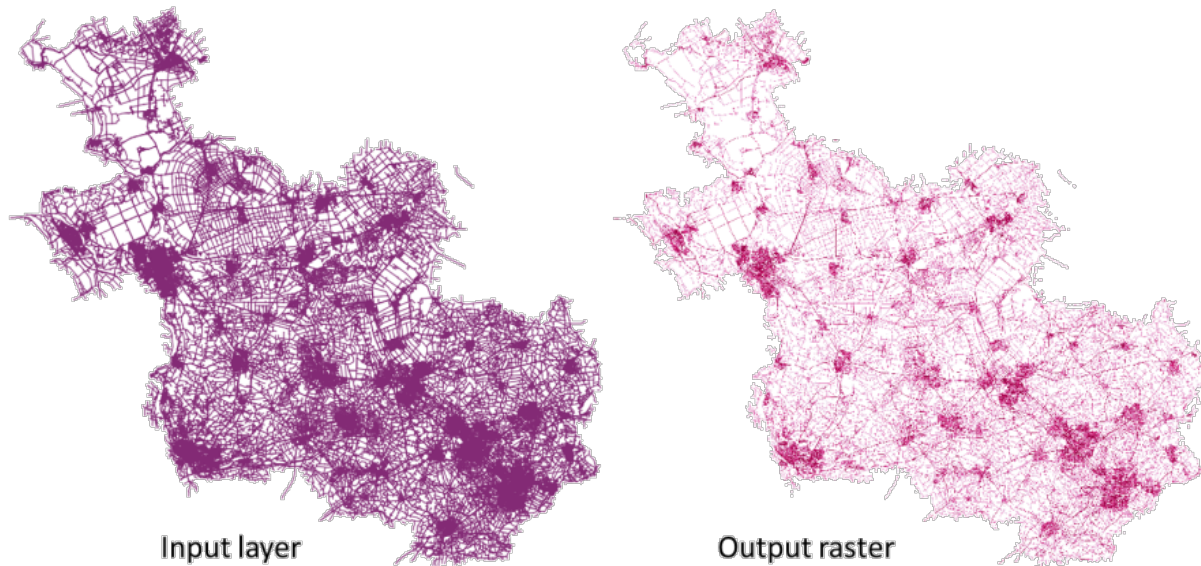


Abb. 28.6: Line density example. Input layer source: Roads Overijssel - The Netherlands (OSM).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input line layer</b>	INPUT	[vector: any]	Input vector layer containing line features
<b>Weight field</b>	WEIGHT	[number]	Field of the layer containing the weight factor to use during the calculation
<b>Search Radius</b>	RADIUS	[number] Default: 10	Radius of the circular neighbourhood. Units can be specified here.
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 10	Pixel size of the output raster layer in layer units. The raster has square pixels.
<b>Line density raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	The output as a raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Line density raster</b>	OUTPUT	[raster]	The output line density raster layer.

## Pythoncode

**Algorithm ID:** native:linedensity

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## TIN Interpolation

Generates a Triangulated Irregular Network (TIN) interpolation of a point vector layer.

With the TIN method you can create a surface formed by triangles of nearest neighbor points. To do this, circumcircles around selected sample points are created and their intersections are connected to a network of non overlapping and as compact as possible triangles. The resulting surfaces are not smooth.

The algorithm creates both the raster layer of the interpolated values and the vector line layer with the triangulation boundaries.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer(s)</b>	INTERPOLATION_	[string]	<p>Vector layer(s) and field(s) to use for the interpolation, coded in a string (see the <code>ParameterInterpolationData</code> class in <a href="#">InterpolationWidgets</a> for more details).</p> <p>The following GUI elements are provided to compose the interpolation data string:</p> <ul style="list-style-type: none"> <li>• <b>Vector layer</b> [vector: any]</li> <li>• <b>Interpolation attribute</b> [tablefield: numeric]: Attribute to use in the interpolation</li> <li>• <b>Use Z-coordinate for interpolation</b> [boolean]: Uses the layer's stored Z values (Default: False)</li> </ul> <p>For each of the added layer-field combinations, a type can be chosen:</p> <ul style="list-style-type: none"> <li>• <i>Points</i></li> <li>• <i>Structured lines</i></li> <li>• <i>Break lines</i></li> </ul> <p>In the string, the layer-field elements are separated by ' : :   : : '. The sub-elements of the layer-field elements are separated by ' : : ~ : : '.</p>
<b>Interpolation method</b>	METHOD	[enumeration] Default: 0	<p>Set the interpolation method to be used. One of:</p> <ul style="list-style-type: none"> <li>• <i>Linear</i></li> <li>• <i>Clough-Toucher (cubic)</i></li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.21 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> <b>xmax,</b> <b>ymin,</b> <b>ymax)</b>	(xmin, EXTENT ymin,	[extent]	Extent of the output raster layer. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Output raster size</b>	PIXEL_SIZE	[number] Default: 0.1	Pixel size of the output raster layer in layer units. In the GUI, the size can be specified by the number of rows (Number of rows) / columns (Number of columns) <b>or</b> the pixel size (Pixel Size X / Pixel Size Y). Increasing the number of rows or columns will decrease the cell size and increase the file size of the output raster. The values in Rows, Columns, Pixel Size X and Pixel Size Y will be updated simultaneously - doubling the number of rows will double the number of columns, and the cell size will be halved. The extent of the output raster will remain the same (approximately).
<b>Interpolated</b>	OUTPUT	[raster] Default: [Save to temporary file]	The output TIN interpolation as a raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>
<b>Triangulation</b>	TRIANGULATION	[vector: line] Default: [Skip output]	The output TIN as a vector layer. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated</b>	OUTPUT	[raster]	The output TIN interpolation as a raster layer
<b>Triangulation</b>	TRIANGULATION	[vector: line]	The output TIN as a vector layer.

## Pythoncode

**Algorithm ID:** qgis:tininterpolation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.7 Layer tools

### Export layer(s) information

Creates a polygon layer with features corresponding to the extent of selected layer(s).

Additional layer details (CRS, provider name, file path, layer name, subset filter, abstract and attribution) are attached as attributes to each feature.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	LAYERS	[vector: any] [list]	Input vector layers to get information on.
<b>Ergebnis</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specification of the output layer with information. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[vector: polygon]	Polygon vector layer showing extent of input layers and associated information in attributes.

## Pythoncode

**Algorithm ID:** native:exportlayersinformation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export to spreadsheet

Exports the attributes of a selection of vector layers into a spreadsheet document or optionally appends them to an existing spreadsheet as additional sheets.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	LAYERS	[vector: any] [list]	Input vector layers. The output spreadsheet will consist of a sheet, for each layer, that contains the attributes of this layer.
<b>Use field aliases as column headings</b>	USE_ALIAS	[boolean] Default: False	Use the field aliases from the attribute table for the spreadsheet.
<b>Export formatted values instead of raw values</b>	FORMATTED_VALU	[boolean] Default: False	If True, exports the formatted, human readable values (e.g., from a <a href="#">value map</a> or <a href="#">value relation</a> ) to the spreadsheet.
<b>Overwrite existing spreadsheet</b>	OVERWRITE	[boolean] Default: True	If the specified spreadsheet exists, setting this option to True will overwrite the existing spreadsheet. If this option is False and the spreadsheet exists, the layers will be appended as additional sheets.
<b>Destination spreadsheet</b>	OUTPUT	[file] Default: [Save to temporary file]	Output spreadsheet with a sheet for every layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Destination spreadsheet</b>	OUTPUT	[file]	Spreadsheet with a sheet for every layer.
<b>Layers within spreadsheet</b>	OUTPUT_LAYERS	[list]	The list of sheets added to the spreadsheet.

## Pythoncode

**Algorithm ID:** native:exporttospreadsheet

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract layer extent

Generates a vector layer with the minimum bounding box (rectangle with N-S orientation) that covers all the input features.

The output layer contains a single bounding box for the whole input layer.

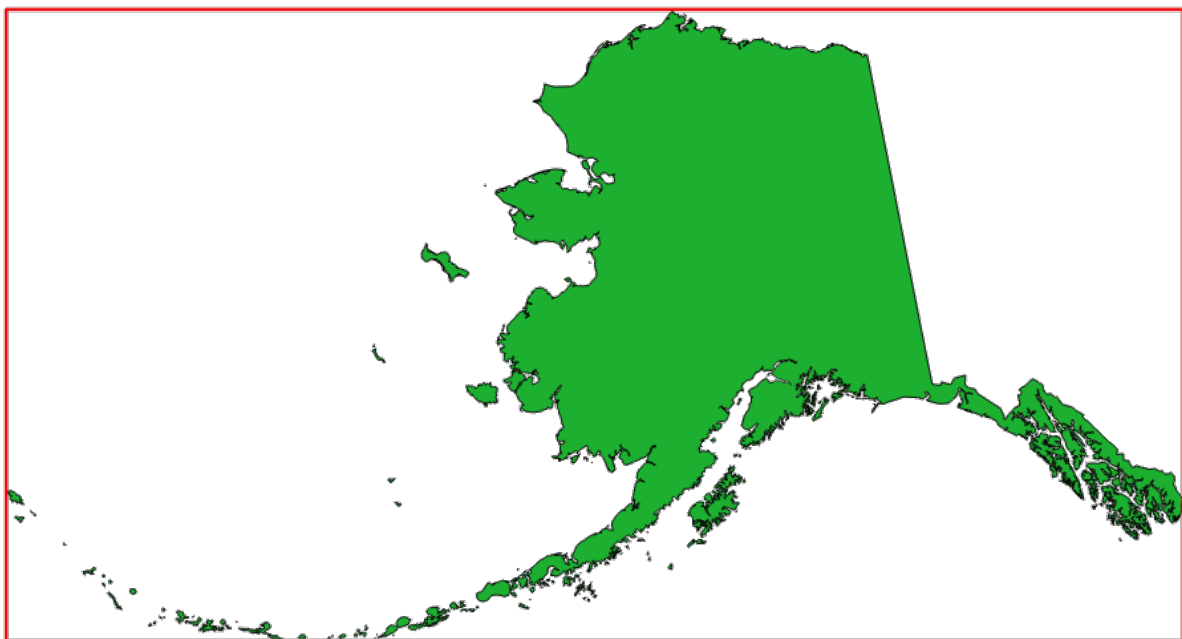


Abb. 28.7: In red the bounding box of the source layer

**Default menu:** *Vector ► Research Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Layer</b>	INPUT	[layer]	Input layer
<b>Extent</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the polygon vector layer for the output extent. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b>	OUTPUT	[vector: polygon]	Output (polygon) vector layer with the extent (minimum bounding box)

## Pythoncode

**Algorithm ID:** qgis:polygonfromlayerextent

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.8 Netz

### Export contours

Creates contours as a vector layer from a mesh scalar dataset.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.23 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Increment between contour levels</b> Optional	INCREMENT	[number] Default: <i>Not set</i>	Interval between generated levels.
<b>Minimum contour level</b> Optional	MINIMUM	[number] Default: <i>Not set</i>	Starting level values of contours.
<b>Maximum contour level</b> Optional	MAXIMUM	[number] Default: <i>Not set</i>	Maximum values of contours, i.e. no generated levels will be greater than this value.
<b>List of contours level</b> Optional	CONTOUR_LEVEL_	[number] Default: <i>Not set</i>	List of wanted levels of contours (separated by commas). If filled, the increment, minimum, and maximum fields will not be considered.
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Exported contour lines</b>	OUTPUT_LINES	[vector: line] Default: [Create temporary layer]	Specify the output line layer representing the contours of the mesh layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Exported contour polygons</b>	OUTPUT_POLYGON	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon layer representing the contours of the mesh layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Exported contour lines</b>	OUTPUT_LINES	[vector: line]	Line layer representing the contours of the mesh layer.
<b>Exported contour polygons</b>	OUTPUT_POLYGON	[vector: polygon]	Polygon layer representing the contours of the mesh layer.

## Pythoncode

**Algorithm ID:** native:meshcontours

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export cross section dataset values on lines from mesh

Extracts a mesh dataset's values from lines contained in a vector layer.

Each line is discretized with a resolution distance parameter for extraction of values on its vertices.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>
<b>Lines for data export</b>	INPUT_LINES	[vector: line]	Lines where the data will be extracted from the dataset mesh
<b>Line segmentation resolution</b>	RESOLUTION	[number] Default: 10.0	The distance between points on the lines where the data will be extracted from the dataset mesh.
<b>Digits count for dataset value</b>	DATASET_DIGITS	[number] Default: 2	Number of digits to round dataset values
<b>Exported data CSV file</b>	OUTPUT	[file] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Exported data CSV file</b>	OUTPUT	[file]	

## Pythoncode

**Algorithm ID:** native:meshexportcrosssection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export mesh edges

Exports a mesh layer's edges to a line vector layer, with the dataset values on edges as attribute values.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Export vector option</b>	VECTOR_OPTION	[enumeration]	Coordinate type of vector value exportation. <ul style="list-style-type: none"> <li>• 0 — Cartesian (x,y)</li> <li>• 1 — Polar (magnitude, degree)</li> <li>• 2 — Cartesian and polar</li> </ul>
<b>Output layer</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: line]	Output vector line layer containing the edges of the input mesh layer with associated dataset values

## Pythoncode

**Algorithm ID:** native:exportmeshedges

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export mesh faces

Exports a mesh layer's faces to a polygon vector layer, with the dataset values on faces as attribute values.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Export option</b>	<b>vector</b> VECTOR_OPTION	[enumeration]	Coordinate type of vector value exportation. <ul style="list-style-type: none"> <li>• 0 — Cartesian (x,y)</li> <li>• 1 — Polar (magnitude, degree)</li> <li>• 2 — Cartesian and polar</li> </ul>
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: polygon]	Output vector polygon layer containing the faces of the input mesh layer with associated dataset values

## Pythoncode

**Algorithm ID:** native:exportmeshfaces

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export mesh on grid

Exports a mesh layer's dataset values to a gridded point vector layer, with the dataset values on this point as attribute values.

For data on volume (3D stacked dataset values), the exported dataset values are averaged on faces using the method defined in [the mesh layer properties](#) (default is Multi level averaging method). 1D meshes are not supported.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>

Fortsetzung auf der nächsten Seite



Tab. 28.27 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> Optional	EXTENT	[extent]	Specify the spatial extent on which to process the data. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Grid spacing</b> Optional	GRID_SPACING	[number] Default: 10.0	Spacing between the sample points to use
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Export option</b>	<b>vector</b> VECTOR_OPTION	[enumeration]	Coordinate type of vector value exportation. <ul style="list-style-type: none"> <li>• 0 — Cartesian (x,y)</li> <li>• 1 — Polar (magnitude, degree)</li> <li>• 2 — Cartesian and polar</li> </ul>
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: point] Default: [Create temporary layer]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: point]	Output vector point layer with dataset values computed from the overlaid face.

## Pythoncode

**Algorithm ID:** native:exportmeshongrid

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export mesh vertices

Exports a mesh layer's vertices to a point vector layer, with the dataset values on vertices as attribute values.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Export vector option</b>	VECTOR_OPTION	[enumeration]	Coordinate type of vector value exportation. <ul style="list-style-type: none"> <li>• 0 — Cartesian (x,y)</li> <li>• 1 — Polar (magnitude, degree)</li> <li>• 2 — Cartesian and polar</li> </ul>
<b>Output layer</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	<b>vector</b> OUTPUT	[vector: point]	Output vector point layer containing the vertices of the input mesh layer with associated dataset values

## Pythoncode

**Algorithm ID:** native:exportmeshvertices

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export time series values from points of a mesh dataset

Extracts a mesh dataset's time series values from points contained in a vector layer.

If the time step is kept to its default value (0 hours), the time step used is the one of the two first datasets of the first selected dataset group.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to extract data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Starting time</b>	STARTING_TIME	[datetime]	The start of the time range to take into account <ul style="list-style-type: none"> <li>0 — Current canvas time</li> <li>1 — Defined date/time</li> <li>2 — Dataset group time step</li> </ul>
<b>Finishing time</b>	FINISHING_TIME	[datetime]	The end of the time range to take into account <ul style="list-style-type: none"> <li>0 — Current canvas time</li> <li>1 — Defined date/time</li> <li>2 — Dataset group time step</li> </ul>
<b>Time step (hours)</b> Optional	TIME_STEP	[number] Default: 0	Time between two consecutive steps to extract. Keep 0 to use time step of the first selected dataset group.
<b>Points for data export</b>	INPUT_POINTS	[vector: point]	Vector layer containing points where the data will be extracted from the dataset mesh
<b>Digits count for coordinates</b>	COORDINATES_DI	[number]	Number of digits to round coordinate values Default: 2
<b>Digits count for dataset value</b>	DATASET_DIGITS	[number] Default: 2	Number of digits to round dataset values

Fortsetzung auf der nächsten Seite

Tab. 28.29 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Exported data CSV file</b>	OUTPUT	[file] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Exported data CSV file</b>	OUTPUT	[file]	. CSV file containing the mesh dataset time series values at the overlaying point features

## Pythoncode

**Algorithm ID:** native:meshexporttimeseries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rasterize mesh dataset

Creates a raster layer from a mesh dataset.

For data on volume (3D stacked dataset values), the exported dataset values are averaged on faces using the method defined in [the mesh layer properties](#) (default is Multi level averaging method). 1D meshes are not supported.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input mesh layer</b>	INPUT	[mesh]	The mesh layer to export data from
<b>Dataset groups</b>	DATASET_GROUPS	[layer] [list]	The dataset groups
<b>Dataset time</b>	DATASET_TIME	[datetime]	The time range to take into account <ul style="list-style-type: none"> <li>• 0 — Current canvas time</li> <li>• 1 — Defined date/time</li> <li>• 2 — Dataset group time step</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.30 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> Optional	EXTENT	[extent]	Specify the spatial extent on which to process the data. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size of the output raster layer.
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Output raster layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster layer</b>	OUTPUT	[raster]	Output raster layer with dataset values computed from the mesh layer.

## Pythoncode

**Algorithm ID:** native:meshrasterize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## TIN mesh creation

Creates a TIN mesh layer from vector layers. The TIN mesh is created using a Delaunay triangulation.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	SOURCE_DATA	[vector: any] [list]	Vector layers to combine to generate the mesh layer
<b>Vector layer</b>	GUI ONLY	[vector: any] [list]	A selector for the vector layers to combine to generate the mesh layer
<b>Value on vertex</b>	GUI ONLY	[tablefield: any]	A selector of the field to use from the selected layer. Each vertex is assigned the corresponding value of its original feature.
<b>Use Z-coordinate for value on vertex</b>	GUI ONLY	[boolean] Default: False	If checked, the Z value of vector layer points or polygons/lines vertices will be used to assign the Z value of the vertex mesh layer. Only available if the input layers are 3D.
<b>Output format</b>	MESH_FORMAT	[enumeration] Default: 2DM	Output format of the generated layer <ul style="list-style-type: none"> <li>• 0 — 2DM</li> <li>• 1 — SELAFIN</li> <li>• 2 — PLY</li> <li>• 3 — Ugrid</li> </ul>
<b>Output coordinate system</b> Optional	CRS_OUTPUT	[crs]	Coordinate Reference System to assign to the output
<b>Output file</b>	OUTPUT_MESH	[mesh] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output file</b>	OUTPUT_MESH	[mesh]	Output mesh layer with dataset values computed from the vector layers.

### Pythoncode

**Algorithm ID:** native:tinmeshcreation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.9 Modellierungswerkzeuge

**Warnung:** Diese Werkzeuge sind nur in der Grafischen Modellierung verfügbar. Sie sind nicht in der Verarbeitungs-Toolbox verfügbar.

### Calculate expression

NEW in 3.34

It calculates the result of a QGIS expression and eliminates the need to use the same expression multiple times throughout a model when the same result is required more than once. Additionally, it enables use cases that would otherwise not be possible. For instance, you can generate a timestamp value once and use it multiple times within the model, if the timestamp were recalculated every time, the values would vary during the model's runtime.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input</b>	INPUT	[expression]	Expression to calculate

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Value</b>	OUTPUT	[Result Value]	Calculated result value, the data type of the output will vary based on the specific expression used in the algorithm.

#### Pythoncode

**Algorithm ID:** native:calculateexpression

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

#### Bedingte Verzweigung

Fügt einem Modell eine bedingte Verzweigung hinzu, so dass Teile des Modells auf der Grundlage des Ergebnisses eines Ausdrucks ausgeführt werden können. Meistens durch die Verwendung von Werkzeugabhängigkeiten, um den Ablauf des Modells zu steuern.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Field</b>	BRANCH	[string]	Name of the condition
<b>Field</b>	CONDITION	[expression]	Expression to evaluate

## Ausgaben

None.

## Pythoncode

**Algorithm ID:** native:condition

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create directory

Creates a new directory on a file system. Directories will be created recursively, creating all required parent directories in order to construct the full specified directory path. No errors will be raised if the directory already exists.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Directory path</b>	PATH	[string]	Folder path to create

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[folder]	Created folder

## Pythoncode

**Algorithm ID:** native:createdirectory

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Feature filter

Filters features from the input layer and redirects them to one or several outputs. If you do not know about any attribute names that are common to all possible input layers, filtering is only possible on the feature geometry and general record mechanisms, such as `$id` and `uuid`.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer.
<b>Outputs and filters</b> (one or more)	OUTPUT_<name of the filter>	[same as input]	The output layers with filters (as many as there are filters).

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b> (one or more)	native:filter_ of filter>	[same as input]	The output layers with filtered features (as many as there are filters).

### Pythoncode

**Algorithm ID:** native:filter

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Filter layers by type

Filters layers by their type. Incoming layers will be directed to different outputs based on whether they are a vector or raster layer.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[layer]	Generic Map Layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vector features</b> Optional	VECTOR	[vector]	A Vector Layer of the input, if compatible
<b>Raster layer</b> Optional	RASTER	[raster]	A Raster Layer of the input, if compatible

## Pythoncode

**Algorithm ID:** native:filterlayersbytype

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Load layer into project

Loads a layer to the current project.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Layer</b>	INPUT	[layer]	Layer to load in the legend
<b>Loaded layer name</b>	NAME	[string]	Name of the loaded layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer</b>	OUTPUT	[same as input]	The (renamed) loaded layer

## Pythoncode

**Algorithm ID:** native:loadlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raise exception

Raises an exception and cancels a model's execution. The exception message can be customized, and optionally an expression based condition can be specified. If an expression condition is used, then the exception will only be raised if the expression result is true. A false result indicates that no exception will be raised, and the model execution can continue uninterrupted.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Message</b>	MESSAGE	[string]	Message to display
<b>Condition</b> Optional	CONDITION	[expression]	Expression to evaluate if true

### Ausgaben

A message in the log panel.

### Pythoncode

**Algorithm ID:** native:raiseexception

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raise message

Raises an information message in the log. The message can be customized, and optionally an expression based condition can be specified. If an expression condition is used, then the message will only be logged if the expression result is true. A false result indicates that no message will be logged.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Information message</b>	MESSAGE	[string]	Message to display
<b>Condition</b> Optional	CONDITION	[expression]	Expression to evaluate if true

## Ausgaben

A message in the log panel.

## Pythoncode

**Algorithm ID:** native:raisemessage

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raise warning

Raises a warning message in the log. The warning message can be customized, and optionally an expression based condition can be specified. If an expression condition is used, then the warning will only be logged if the expression result is true. A false result indicates that no warning will be logged.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Message</b>	MESSAGE	[string]	Message to display
<b>Condition</b> Optional	CONDITION	[expression]	Expression to evaluate if true

## Ausgaben

A message in the log panel.

## Pythoncode

**Algorithm ID:** native:raisewarning

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rename layer

Renames a layer.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Layer</b>	INPUT	[layer]	Layer to rename
<b>New name</b>	NAME	[string]	The new name of the layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer</b>	OUTPUT	[same as input]	The (renamed) output layer

### Pythoncode

**Algorithm ID:** native:renamelayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Save log to file

Saves the model's execution log to a file. Optionally, the log can be saved in a HTML formatted version.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Use HTML</b>	USE_HTML	[Boolean] Default: False	Use HTML formatting

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>File</b>	OUTPUT	[string]	Destination of the log

## Pythoncode

**Algorithm ID:** native:savelog

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Set project variable

Sets an expression variable for the current project.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Variable name</b>	NAME	[string]	Name of the variable
<b>Variable value</b>	VALUE	[string]	Value to be stored

## Ausgaben

None.

## Pythoncode

**Algorithm ID:** native:setprojectvariable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## String concatenation

Concatenates two strings into a single one in the Processing Modeler.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input 1</b>	INPUT_1	[string]	First string
<b>Input 2</b>	INPUT_2	[string]	Second string

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Concatenation</b>	CONCATENATION	[string]	The concatenated string

## Pythoncode

**Algorithm ID:** native:stringconcatenation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Variable distance buffer

**Warnung:** This algorithm is deprecated and can be removed anytime. Prefer using [Buffer](#) algorithm instead.

Computes a buffer area for all the features in an input layer.

The size of the buffer for a given feature is defined by an attribute, so it allows different features to have different buffer sizes.

**Siehe auch:**

[Buffer](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Distance field</b>	DISTANCE	[tablefield: numeric]	Attribute for the distance radius of the buffer
<b>Segments</b>	SEGMENTS	[number] Default: 5	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
<b>Dissolve result</b>	DISSOLVE	[boolean] Default: <i>False</i>	Choose to dissolve the final buffer, resulting in a single feature covering all input features.

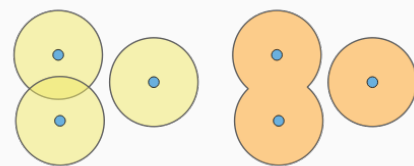


Abb. 28.8: Normal and dissolved buffer

Fortsetzung auf der nächsten Seite

Tab. 28.32 – Fortsetzung der vorherigen Seite


Bezeichnung	Name	Typ	Beschreibung
<b>End cap style</b>	END_CAP_STYLE	[enumeration] Default: Round	Controls how line endings are handled in the buffer. 
<b>Join style</b>	JOIN_STYLE	[enumeration] Default: Round	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line.
<b>Miter limit</b>	MITER_LIMIT	[number] Default: 2.0	Only applicable for mitered join styles, and controls the maximum distance from the offset curve to use when creating a mitered join.

Abb. 28.9: Round, flat and square cap styles

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffer</b>	OUTPUT	[vector: polygon]	Buffer polygon vector layer.

## Pythoncode

**Algorithm ID:** qgis:variabledistancebuffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.10 Network analysis

### Service area (from layer)

Returns all the edges or parts of edges of a network that can be reached within a distance or a time, starting from a point layer. This allows evaluation of accessibility within a network, e.g. what are the places I can navigate to on a road network without spending cost greater than a given value (the cost can be distance or time).



## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector layer representing network</b>	INPUT	[vector: line]	Line vector layer representing the network to be covered
<b>Vector layer with start points</b>	START_POINTS	[vector: point]	Point vector layer whose features are used as start points to generate the service areas
<b>Path type to calculate</b>	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: <ul style="list-style-type: none"> <li>• 0 — Shortest</li> <li>• 1 — Fastest</li> </ul>
<b>Travel cost (distance for „Shortest“, time for „Fastest“)</b>	TRAVEL_COST	[number] Default: 0	The value is estimated as a distance (in the network layer units) when looking for the <i>Shortest</i> path and as time (in hours) for the <i>Fastest</i> path.
<b>Service area (lines)</b>	OUTPUT_LINES	[vector: line] Default: [Create temporary layer]	Specify the output line layer for the service area. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Service area (boundary nodes)</b>	OUTPUT	[vector: point] Default: [Skip output]	Specify the output point layer for the service area boundary nodes. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Direction field</b> Optional	DIRECTION_FIEL	[tablefield: string] Default: 0.0	The field used to specify directions for the network edges. The values used in this field are specified with the three parameters Value for forward direction, Value for backward direction and Value for both directions. Forward and reverse directions correspond to a one-way edge, „both directions“ indicates a two-way edge. If a feature does not have a value in this field, or no field is set then the default direction setting (provided with the Default direction parameter) is used.
<b>Value for forward direction</b> Optional	VALUE_FORWARD	[string] Default: ,‘ (empty string)	Value set in the direction field to identify edges with a forward direction
<b>Value for backward direction</b> Optional	VALUE_BACKWARD	[string] Default: ,‘ (empty string)	Value set in the direction field to identify edges with a backward direction
<b>Value for both directions</b> Optional	VALUE_BOTH	[string] Default: ,‘ (empty string)	Value set in the direction field to identify bidirectional edges
<b>Default direction</b>	DEFAULT_DIRECT	[enumeration] Default: 2	If a feature has no value set in the direction field or if no direction field is set, then this direction value is used. One of: <ul style="list-style-type: none"> <li>• 0 — Forward direction</li> <li>• 1 — Backward direction</li> <li>• 2 — Both directions</li> </ul>
<b>Speed field</b> Optional	SPEED_FIELD	[tablefield: string]	Field providing the speed value (in km/h) for the edges of the network when looking for the fastest path. If a feature does not have a value in this field, or no field is set then the default speed value (provided with the Default speed parameter) is used.
<b>Default speed (km/h)</b>	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
<b>Topology tolerance</b>	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected
<b>Include upper/lower bound points</b>	INCLUDE_BOUNDS	[boolean] Default: False	Creates a point layer output with two points for each edge at the boundaries of the service area. One point is the start of that edge, the other is the end.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Service area (boundary nodes)</b>	OUTPUT	[vector: point]	The output point layer with the service area boundary nodes.
<b>Service area (lines)</b>	OUTPUT_LINES	[vector: line]	Line layer representing the parts of the network that can be serviced by the start points, for the given cost.

## Pythoncode

**Algorithm ID:** `qgis:serviceareafromlayer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Service area (from point)

Returns all the edges or parts of edges of a network that can be reached within a given distance or time, starting from a point feature. This allows the evaluation of accessibility within a network, e.g. what are the places I can navigate to on a road network without spending a cost greater than a given value (the cost can be distance or time).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector layer representing the network</b>	INPUT	[vector: line]	Line vector layer representing the network to be covered
<b>Start point (x, y)</b>	START_POINT	[coordinates]	Coordinate of the point to calculate the service area around.
<b>Path type to calculate</b>	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: <ul style="list-style-type: none"> <li>0 — Shortest</li> <li>1 — Fastest</li> </ul>
<b>Travel cost (distance for „Shortest“, time for „Fastest“)</b>	TRAVEL_COST	[number] Default: 0	The value is estimated as a distance (in the network layer units) when looking for the <i>Shortest</i> path and as time (in hours) for the <i>Fastest</i> path.

Fortsetzung auf der nächsten Seite

Tab. 28.35 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Service area (lines)</b>	OUTPUT_LINES	[vector: line] Default: [Create temporary layer]	Specify the output line layer for the service area. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Service area (boundary nodes)</b>	OUTPUT	[vector: point] Default: [Skip output]	Specify the output point layer for the service area boundary nodes. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Direction field</b> Optional	DIRECTION_FIEL	[tablefield: string] Default: 0.0	The field used to specify directions for the network edges. The values used in this field are specified with the three parameters Value for forward direction, Value for backward direction and Value for both directions. Forward and reverse directions correspond to a one-way edge, „both directions“ indicates a two-way edge. If a feature does not have a value in this field, or no field is set then the default direction setting (provided with the Default direction parameter) is used.
<b>Value for forward direction</b> Optional	VALUE_FORWARD	[string] Default: ; (empty string)	Value set in the direction field to identify edges with a forward direction
<b>Value for backward direction</b> Optional	VALUE_BACKWARD	[string] Default: ; (empty string)	Value set in the direction field to identify edges with a backward direction
<b>Value for both directions</b> Optional	VALUE_BOTH	[string] Default: ; (empty string)	Value set in the direction field to identify bidirectional edges

Fortsetzung auf der nächsten Seite

Tab. 28.36 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Default direction</b>	DEFAULT_DIRECT	[enumeration] Default: 2	If a feature has no value set in the direction field or if no direction field is set, then this direction value is used. One of: <ul style="list-style-type: none"> <li>• 0 — Forward direction</li> <li>• 1 — Backward direction</li> <li>• 2 — Both directions</li> </ul>
<b>Speed field</b> Optional	SPEED_FIELD	[tablefield: string]	Field providing the speed value (in km/h) for the edges of the network when looking for the fastest path. If a feature does not have a value in this field, or no field is set then the default speed value (provided with the Default speed parameter) is used.
<b>Default speed (km/h)</b>	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
<b>Topology tolerance</b>	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected
<b>Include upper/lower bound points</b>	INCLUDE_BOUNDS	[boolean] Default: False	Creates a point layer output with two points for each edge at the boundaries of the service area. One point is the start of that edge, the other is the end.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Service area (boundary nodes)</b>	OUTPUT	[vector: point]	The output point layer with the service area boundary nodes.
<b>Service area (lines)</b>	OUTPUT_LINES	[vector: line]	Line layer representing the parts of the network that can be serviced by the start point, for the given cost.

## Pythoncode

**Algorithm ID:** native:serviceareafrompoint

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Shortest path (layer to point)

Computes the optimal (shortest or fastest) routes from multiple start points defined by a vector layer and a given end point.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector layer representing network</b>	INPUT	[vector: line]	Line vector layer representing the network to be covered
<b>Path type to calculate</b>	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: <ul style="list-style-type: none"> <li>0 — Shortest</li> <li>1 — Fastest</li> </ul>
<b>Vector layer with start points</b>	START_POINTS	[vector: point]	Point vector layer whose features are used as start points of the routes
<b>End point (x, y)</b>	END_POINT	[coordinates]	Point feature representing the end point of the routes
<b>Shortest path</b>	OUTPUT	[vector: line]	Specify the output line layer for the shortest paths. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Direction field</b> Optional	DIRECTION_FIEL	[tablefield: string] Default: 0.0	The field used to specify directions for the network edges. The values used in this field are specified with the three parameters Value for forward direction, Value for backward direction and Value for both directions. Forward and reverse directions correspond to a one-way edge, „both directions“ indicates a two-way edge. If a feature does not have a value in this field, or no field is set then the default direction setting (provided with the Default direction parameter) is used.
<b>Value for forward direction</b> Optional	VALUE_FORWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a forward direction

Fortsetzung auf der nächsten Seite

Tab. 28.38 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Value for backward direction</b> Optional	VALUE_BACKWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a backward direction
<b>Value for both directions</b> Optional	VALUE_BOTH	[string] Default: ‘ (empty string)	Value set in the direction field to identify bidirectional edges
<b>Default direction</b>	DEFAULT_DIRECT	[enumeration] Default: 2	If a feature has no value set in the direction field or if no direction field is set, then this direction value is used. One of: <ul style="list-style-type: none"> <li>• 0 — Forward direction</li> <li>• 1 — Backward direction</li> <li>• 2 — Both directions</li> </ul>
<b>Speed field</b> Optional	SPEED_FIELD	[tablefield: string]	Field providing the speed value (in km/h) for the edges of the network when looking for the fastest path.  If a feature does not have a value in this field, or no field is set then the default speed value (provided with the Default speed parameter) is used.
<b>Default speed (km/h)</b>	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
<b>Topology tolerance</b>	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Shortest path</b>	OUTPUT	[vector: line]	Line layer of the shortest or fastest path from each of the start points to the end point

## Pythoncode

**Algorithm ID:** native:shortestpathlayertopoint

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Shortest path (point to layer)

Computes the optimal (shortest or fastest) routes between a given start point and multiple end points defined by a point vector layer.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector layer representing network</b>	INPUT	[vector: line]	Line vector layer representing the network to be covered
<b>Path type to calculate</b>	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: <ul style="list-style-type: none"> <li>• 0 — Shortest</li> <li>• 1 — Fastest</li> </ul>
<b>Start point (x, y)</b>	START_POINT	[coordinates]	Point feature representing the start point of the routes
<b>Vector layer with end points</b>	END_POINTS	[vector: point]	Point vector layer whose features are used as end points of the routes
<b>Shortest path</b>	OUTPUT	[vector: line]	Specify the output line layer for the shortest paths. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Direction field</b> Optional	DIRECTION_FIEL	[tablefield: string] Default: 0.0	The field used to specify directions for the network edges. The values used in this field are specified with the three parameters Value for forward direction, Value for backward direction and Value for both directions. Forward and reverse directions correspond to a one-way edge, „both directions“ indicates a two-way edge. If a feature does not have a value in this field, or no field is set then the default direction setting (provided with the Default direction parameter) is used.
<b>Value for forward direction</b> Optional	VALUE_FORWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a forward direction

Fortsetzung auf der nächsten Seite



Tab. 28.40 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Value for backward direction</b> Optional	VALUE_BACKWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a backward direction
<b>Value for both directions</b> Optional	VALUE_BOTH	[string] Default: ‘ (empty string)	Value set in the direction field to identify bidirectional edges
<b>Default direction</b>	DEFAULT_DIRECT	[enumeration] Default: 2	If a feature has no value set in the direction field or if no direction field is set, then this direction value is used. One of: <ul style="list-style-type: none"> <li>• 0 — Forward direction</li> <li>• 1 — Backward direction</li> <li>• 2 — Both directions</li> </ul>
<b>Speed field</b> Optional	SPEED_FIELD	[tablefield: string]	Field providing the speed value (in km/h) for the edges of the network when looking for the fastest path.  If a feature does not have a value in this field, or no field is set then the default speed value (provided with the Default speed parameter) is used.
<b>Default speed (km/h)</b>	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
<b>Topology tolerance</b>	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Shortest path</b>	OUTPUT	[vector: line]	Line layer of the shortest or fastest path from each of the start points to the end point

## Pythoncode

**Algorithm ID:** native:shortestpathpointtolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Shortest path (point to point)

Computes the optimal (shortest or fastest) route between a given start point and a given end point.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector layer representing network</b>	INPUT	[vector: line]	Line vector layer representing the network to be covered
<b>Path type to calculate</b>	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: <ul style="list-style-type: none"> <li>• 0 — Shortest</li> <li>• 1 — Fastest</li> </ul>
<b>Start point (x, y)</b>	START_POINT	[coordinates]	Point feature representing the start point of the routes
<b>End point (x, y)</b>	END_POINT	[coordinates]	Point feature representing the end point of the routes
<b>Shortest path</b>	OUTPUT	[vector: line]	Specify the output line layer for the shortest paths. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Direction field</b> Optional	DIRECTION_FIEL	[tablefield: string] Default: 0.0	The field used to specify directions for the network edges. The values used in this field are specified with the three parameters Value for forward direction, Value for backward direction and Value for both directions. Forward and reverse directions correspond to a one-way edge, „both directions“ indicates a two-way edge. If a feature does not have a value in this field, or no field is set then the default direction setting (provided with the Default direction parameter) is used.
<b>Value for forward direction</b> Optional	VALUE_FORWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a forward direction

Fortsetzung auf der nächsten Seite

Tab. 28.42 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Value for backward direction</b> Optional	VALUE_BACKWARD	[string] Default: ‘ (empty string)	Value set in the direction field to identify edges with a backward direction
<b>Value for both directions</b> Optional	VALUE_BOTH	[string] Default: ‘ (empty string)	Value set in the direction field to identify bidirectional edges
<b>Default direction</b>	DEFAULT_DIRECT	[enumeration] Default: 2	If a feature has no value set in the direction field or if no direction field is set, then this direction value is used. One of: <ul style="list-style-type: none"> <li>• 0 — Forward direction</li> <li>• 1 — Backward direction</li> <li>• 2 — Both directions</li> </ul>
<b>Speed field</b> Optional	SPEED_FIELD	[tablefield: string]	Field providing the speed value (in km/h) for the edges of the network when looking for the fastest path.  If a feature does not have a value in this field, or no field is set then the default speed value (provided with the Default speed parameter) is used.
<b>Default speed (km/h)</b>	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
<b>Topology tolerance</b>	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Shortest path</b>	OUTPUT	[vector: line]	Line layer of the shortest or fastest path from each of the start point to the end point

## Pythoncode

**Algorithm ID:** native:shortestpathpointtopoint

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.11 Plots

### Bar plot

Creates a bar plot from a category and a layer field.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Category field name</b>	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the bars (X axis)
<b>Value field</b>	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).
<b>Bar plot</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Bar plot</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

#### Pythoncode

**Algorithm ID:** qgis:barplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Box plot

Creates a box plot from a category field and a numerical layer field.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Category field name</b>	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the boxes (X axis)
<b>Value field</b>	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).

Fortsetzung auf der nächsten Seite

Tab. 28.43 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Additional statistic lines</b>	MSD	[enumeration] Default: 0	Additional statistics information to add to the plot. One of: <ul style="list-style-type: none"> <li>• 0 — Show Mean</li> <li>• 1 — Show Standard Deviation</li> <li>• 2 — Don't show mean and standard deviation</li> </ul>
<b>Box plot</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Box plot</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:boxplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Mean and standard deviation plot

Creates a box plot with mean and standard deviation values.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input table</b>	INPUT	[vector: any]	Input vector layer
<b>Category name field</b>	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the boxes (X axis)
<b>Value field</b>	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).
<b>Plot</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Plot</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:meanandstandarddeviationplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Polar plot

Generates a polar plot based on the value of an input vector layer.

Two fields must be entered as parameters: one that defines the category each feature (to group features) and another one with the variable to plot (this has to be a numeric one).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Category name field</b>	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the features (X axis)
<b>Value field</b>	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).
<b>Polar plot</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Polar plot</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:polarplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster layer histogram

Generates a histogram with the values of a raster layer.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band]	Raster band to use for the histogram
<b>number of bins</b>	BINS	[number] Default: 10	The number of bins to use in the histogram (X axis). Minimum 2.
<b>Histogram</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Histogram</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:rasterlayerhistogram

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Vector layer histogram

Generates a histogram with the values of the attribute of a vector layer.

The attribute to use for computing the histogram must be numeric.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Attribute</b>	FIELD	[tablefield: any]	Value to use for the plot (Y axis).
<b>number of bins</b>	BINS	[number] Default: 10	The number of bins to use in the histogram (X axis). Minimum 2.
<b>Histogram</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Histogram</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

### Pythoncode

**Algorithm ID:** qgis:vectorlayerhistogram

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Vector layer scatterplot

Creates a simple X - Y scatter plot for a vector layer.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>X attribute</b>	XFIELD	[tablefield: any]	Field to use for the X axis
<b>Y attribute</b>	YFIELD	[tablefield: any]	Field to use for the Y axis
<b>Scatterplot</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Scatterplot</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:vectorlayersscatterplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Vector layer scatterplot 3D

Creates a 3D scatter plot for a vector layer.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>X attribute</b>	XFIELD	[tablefield: any]	Field to use for the X axis
<b>Y attribute</b>	YFIELD	[tablefield: any]	Field to use for the Y axis
<b>Z attribute</b>	ZFIELD	[tablefield: any]	Field to use for the Z axis
<b>Histogram</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Histogram</b>	OUTPUT	[html]	HTML file with the plot. Available in the <i>Processing ► Result Viewer</i> .

## Pythoncode

**Algorithm ID:** qgis:scatter3dplot

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.12 Point Cloud Conversion

NEW in 3.32

### Convert format

Converts a point cloud to a different file format, e.g. creates a compressed .LAZ.

### Parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to convert
<b>Converted</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the point cloud file to use as output. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Outputs

Label	Name	Type	Description
<b>Converted</b>	OUTPUT	[point cloud]	Output point cloud layer in a modified file format. Currently supported formats are .LAS, .LAZ, .COPC.LAZ and .VPC.

## Python code

**Algorithm ID:** pdal:convertformat

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Export to raster

Exports point cloud data to a 2D raster grid having cell size of given resolution, writing values from the specified attribute.

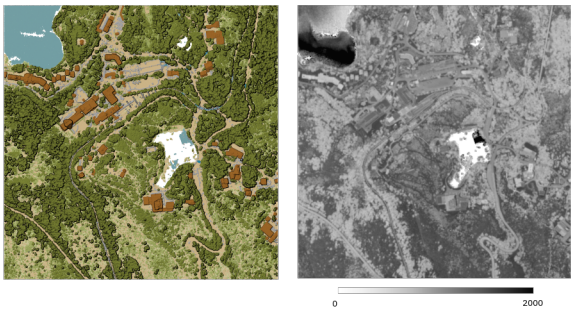


Abb. 28.10: Raster output using Intensity attribute of points

Parameters

Basic parameters

Label	Name	Type	Description
Input layer	INPUT	[point cloud]	Input point cloud layer to export
Attribute	ATTRIBUTE	[field] [enumeration]	A Field of the point cloud layer to extract the values from
Resolution of the density raster	RESOLUTION	[number] Default: 1.0	Cell size of the output raster
Tile size for parallel runs	TILE_SIZE	[number] Default: 1000	
Exported	OUTPUT	[raster] Default: [Save to temporary file]	Specify the raster file to export the data to. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>X origin of a tile for parallel runs</b> Optional	ORIGIN_X	[number]	
<b>Y origin of a tile for parallel runs</b> Optional	ORIGIN_Y	[number]	

## Outputs

Label	Name	Type	Description
<b>Exported</b>	OUTPUT	[raster]	Output raster layer features of the point cloud layer are exported to. Currently supported format is .TIF.

## Python code

**Algorithm ID:** pdal:exportraster

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Export raster (using triangulation)

Exports point cloud data to a 2D raster grid using a triangulation of points and then interpolating cell values from triangles.

**Bemerkung:** Using this algorithm can be slower if you are dealing with a large dataset. If your point cloud is dense, you can export your ground points as a raster using the *Export to raster* algorithm.

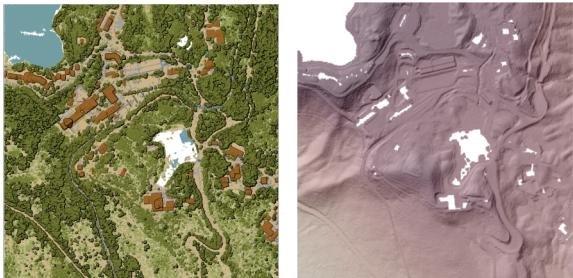


Abb. 28.11: Terrain raster output generated by point cloud triangulation

Parameters

Basic parameters

Label	Name	Type	Description
Input layer	INPUT	[point cloud]	Input point cloud layer to export
Resolution of the density raster	RESOLUTION	[number] Default: 1.0	Cell size of the output raster
Tile size for parallel runs	TILE_SIZE	[number] Default: 1000	
Exported	OUTPUT	[raster] Default: [Save to temporary file]	Specify the raster file to export the data to. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>X origin of a tile for parallel runs</b> Optional	ORIGIN_X	[number]	
<b>Y origin of a tile for parallel runs</b> Optional	ORIGIN_Y	[number]	

## Outputs

Label	Name	Type	Description
<b>Exported (using triangulation)</b>	OUTPUT	[raster]	Output raster layer features of the point cloud layer are exported to. Currently supported format is .TIF.

## Python code

**Algorithm ID:** pdal:exportrastertin

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Export to vector

Exports point cloud data to a vector layer with 3D points (a GeoPackage), optionally with extra attributes.

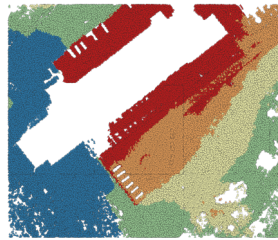


Abb. 28.12: Exporting point cloud (ground points) to a vector layer styled based on the elevation

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to export
<b>Attribute</b> Optional	ATTRIBUTE	[field] [list]	One or more fields of the point cloud layer to export with the points.
<b>Exported</b>	OUTPUT	[vector] Default: [Save to temporary file]	Specify the vector file to export the data to. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Exported</b>	OUTPUT	[vector]	Output vector layer features of the point cloud layer are exported to. Currently supported format is .GPKG.

## Python code

**Algorithm ID:** pdal:exportvector

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.13 Point Cloud Data Management

NEW in 3.32

### Assign projection

Assigns a Coordinate Reference System to a point cloud layer, if it is missing or wrong. A new layer is created.

**Siehe auch:**

[Reproject](#)

### Parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to assign a CRS to
<b>Desired CRS</b>	CRS	[crs]	The CRS to apply to the layer
<b>Output layer</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the point cloud file to use as output. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>



## Outputs

Label	Name	Type	Description
<b>Converted</b>	OUTPUT	[point cloud]	Output point cloud layer with a new CRS. Currently supported formats are .LAS, .LAZ, .COPC.LAZ and .VPC.

## Python code

**Algorithm ID:** pdal:assignprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Build virtual point cloud (VPC)

Creates a *virtual point cloud (VPC)* from input point cloud data.

If you leave the optional parameters unchecked, the VPC file will be built very quickly as the algorithm will only read metadata of input files. With any of the optional parameters set, the algorithm will read all points which can take some time.



Abb. 28.13: Generating Virtual Point Cloud with overview from a set of point cloud tiles

**Siehe auch:**

[Merge](#)

## Parameters

Label	Name	Type	Description
<b>Input layers</b>	LAYERS	[point cloud] [list]	Input point cloud layers to combine inside a virtual point cloud layer
<b>Calculate boundary polygon</b>	BOUNDARY	[boolean] Default: False	Set to True to show the exact boundaries of data (rather than just rectangular extent)
<b>Calculate statistics</b>	STATISTICS	[boolean] Default: False	Set to True to understand ranges of values of the various attributes
<b>Build overview point cloud</b>	OVERVIEW	[boolean] Default: False	Generates a single “thinned” point cloud of all the input data (using only every 1000th point from original data). The overview point cloud will be created next to the VPC file - for example, for <code>mydata.vpc</code> , the overview point cloud would be named <code>mydata-overview.copc.laz</code> .
<b>Virtual point cloud</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the point cloud file to build the data into. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Virtual point cloud</b>	OUTPUT	[raster]	Output point cloud layer combining all the input data, as a virtual file.

## Python code

**Algorithm ID:** `pdal:virtualpointcloud`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Clip

Clips a point cloud layer by a polygon layer so that the resulting point cloud contains only points within the polygons.

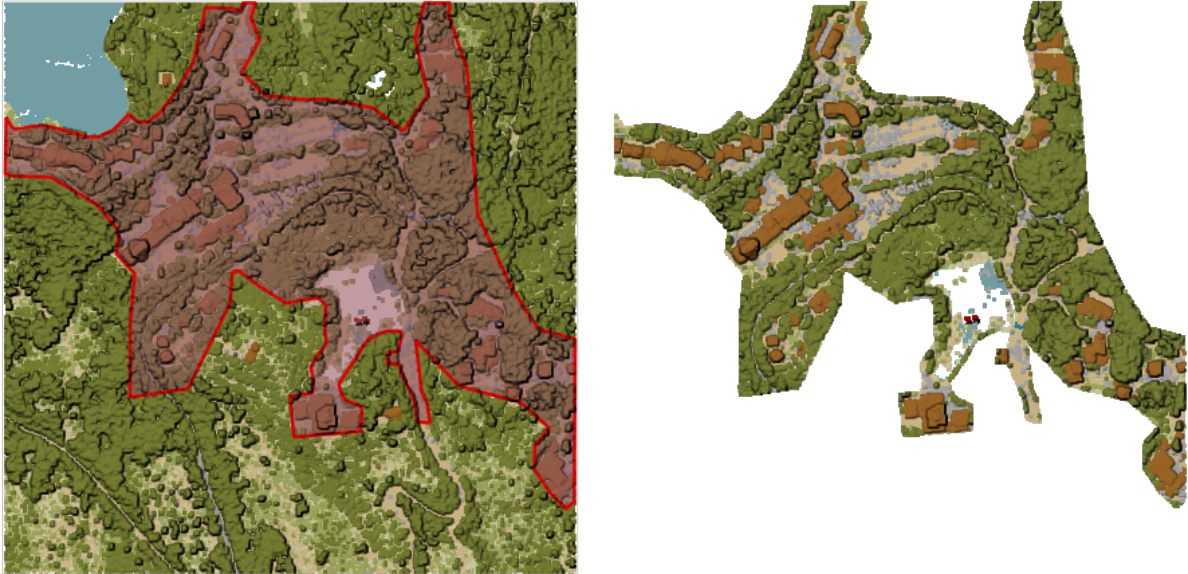


Abb. 28.14: Clipping an input point cloud layer with a polygon coverage

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to clip
<b>Clipping polygons</b>	OVERLAY	[vector: polygon]	Polygon vector layer to use as coverage for clipping the points
<b>Clipped</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the point cloud file to export the clipped points to. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	<p>A map extent for selecting a subset of features in the point cloud data</p> <p>Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Clipped</b>	OUTPUT	[raster]	Output point cloud whose features are the points within the coverage polygon layer.

## Python code

**Algorithm ID:** pdal:clip

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Create COPC

Creates the index for all the input point cloud files in a batch mode.

### Parameters

Label	Name	Type	Description
<b>Input layers</b>	LAYERS	[point cloud] [list]	Input point cloud layers to create an index for
<b>Output directory</b> Optional	OUTPUT	[folder] Default: [Skip output]	Specify the folder to create the new files in. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

### Outputs

Label	Name	Type	Description
<b>Output directory</b>	OUTPUT	[folder]	Output folder containing point cloud layers with accompanying COPC index files.

### Python code

**Algorithm ID:** pdal:createcopc

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Information

Outputs basic metadata from an input point cloud file.

Example of output information:

```
LAS                1.4
point format      6
count            56736130
scale            0.001 0.001 0.001
offset          431749.999 5440919.999 968.898
extent          431250 5440420 424.266
                 432249.999 5441419.999 1513.531
crs              ETRS89 / UTM zone 34N (N-E) (EPSG:3046) (vertical CRS missing!)
units            horizontal=metre vertical=unknown

Attributes:
- X floating 8
- Y floating 8
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
- Z floating 8
- Intensity unsigned 2
- ReturnNumber unsigned 1
- NumberOfReturns unsigned 1
- ScanDirectionFlag unsigned 1
- EdgeOfFlightLine unsigned 1
- Classification unsigned 1
- ScanAngleRank floating 4
- UserData unsigned 1
- PointSourceId unsigned 2
- GpsTime floating 8
- ScanChannel unsigned 1
- ClassFlags unsigned 1
```

## Parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to extract metadata information from
<b>Layer information</b>	OUTPUT	[file] Default: [Save to temporary file]	Specify the file to store the metadata information. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Layer information</b>	OUTPUT	[vector]	HTML file to store the metadata information.

## Python code

**Algorithm ID:** pdal:info

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Merge

Merges multiple point cloud files into a single one.

**Siehe auch:**

*Build virtual point cloud (VPC)*

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layers</b>	LAYERS	[point cloud] [list]	Input point cloud layers to merge into a single one
<b>Merged</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the output point cloud merging input files. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Merged</b>	OUTPUT	[point cloud]	Output point cloud layer merging all the input files.

## Python code

**Algorithm ID:** pdal:merge

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Reproject

Reprojects a point cloud to a different Coordinate Reference System (CRS).

**Siehe auch:**

*Projektion zuweisen*

## Parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to reproject to a different CRS
<b>Target CRS</b>	CRS	[crs]	The CRS to apply to the layer
<b>Reprojected</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the reprojected point cloud file. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Advanced parameters

Label	Name	Type	Description
<b>Coordinate operation</b> Optional	OPERATION	[datum]	The <i>datum transformation</i> to use to reproject the data between the origin and target systems.



## Outputs

Label	Name	Type	Description
<b>REPROJECTED</b>	OUTPUT	[point cloud]	Output point cloud layer in the target CRS.

## Python code

**Algorithm ID:** `pdal:reproject`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Thin (by sampling radius)

Creates a thinned version of the point cloud by performing sampling by distance point (reduces the number of points within a certain radius).

**Siehe auch:**

*Thin (by skipping points)*

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to create a thinned version from
<b>Sampling radius (in map units)</b>	SAMPLING_RADIUS	[number] Default: 1.0	Distance within which points are sampled to a unique point
<b>Thinned (by radius)</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the output point cloud with reduced points. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	<p>A map extent for selecting a subset of features in the point cloud data</p> <p>Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Thinned radius)</b>	(by OUTPUT	[point cloud]	Output point cloud layer with reduced points.

## Python code

**Algorithm ID:** pdal:thinbyradius

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Thin (by skipping points)

Creates a thinned version of the point cloud by keeping only every N-th point (reduces the number of points by skipping nearby points).

### Siehe auch:

*Thin (by sampling radius)*

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to create a thinned version from
<b>Number of points to skip</b>	POINTS_NUMBER	[number] Default: 1	Keep only every N-th point in the input layer
<b>Thinned (by decimation)</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the output point cloud with reduced points. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Thinned decimation)</b>	(by OUTPUT	[point cloud]	Output point cloud layer with reduced points.

## Python code

**Algorithm ID:** pdal:thinbydecimate

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Tile

Creates tiles from input point cloud files, recommended for best performance (in display or analysis) with such datasets in QGIS.

**Siehe auch:**

*Build virtual point cloud (VPC), Create COPC*

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layers</b>	LAYERS	[point cloud] [list]	Input point cloud layers to create tiles from
<b>Tile length</b>	LENGTH	[number] Default: 1000.0	Size of the edge of each generated tile
<b>Output directory</b>	OUTPUT	[folder] Default: [Save to temporary folder]	Specify the folder to store the generated tiles. One of: <ul style="list-style-type: none"><li>• Save to a Temporary Directory</li><li>• Save to Directory</li></ul>

### Advanced parameters

Label	Name	Type	Description
<b>Assign CRS</b> Optional	CRS	[crs]	The CRS to apply to the layer

## Outputs

Label	Name	Type	Description
<b>Output directory</b>	OUTPUT	[folder]	Output folder containing the tiles generated from input files.

## Python code

**Algorithm ID:** pdal:tile

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.14 Point Cloud Extraction

NEW in 3.32

### Boundary

Exports a polygon file containing point cloud layer boundary. It may contain holes and it may be a multi-part polygon.

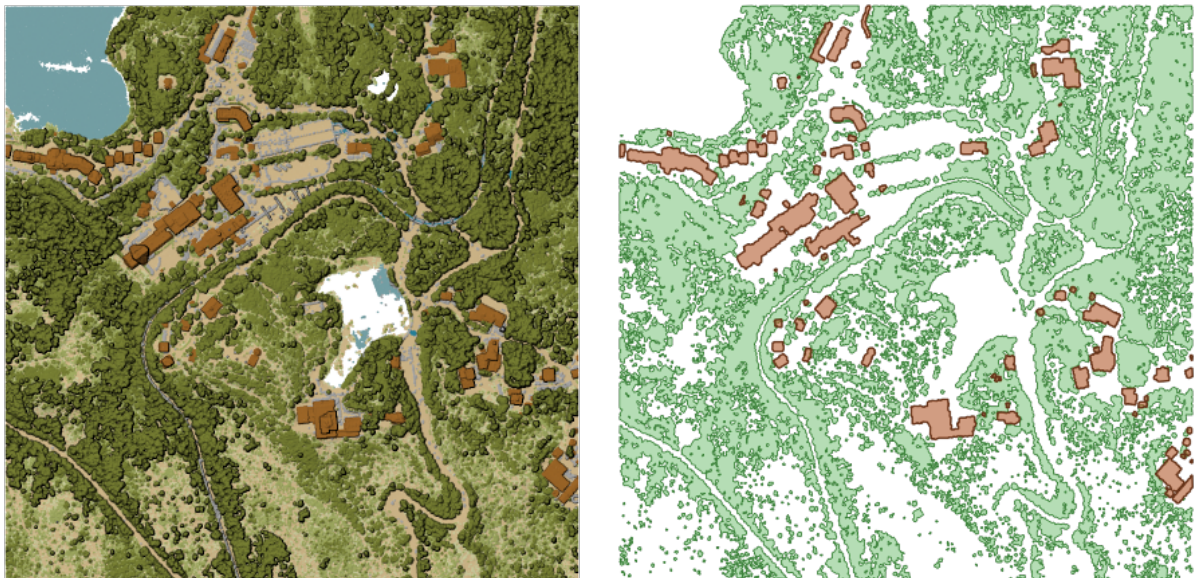


Abb. 28.15: Extracting high vegetation and building polygons from an input point cloud layer

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to calculate boundary for
<b>Resolution of cells used to calculate boundary</b> Optional	RESOLUTION	[number]	Resolution of cells used to calculate boundary
<b>Minimal number of points in a cell to consider cell occupied</b> Optional	THRESHOLD	[number]	Minimal number of points in a cell to consider cell occupied
<b>Boundary</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specify the point cloud file to use as output. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Outputs

Label	Name	Type	Description
<b>Boundary</b>	OUTPUT	[vector: polygon]	Output polygon vector layer with the point cloud boundary. Currently supported format is .GPKG.

## Python code

**Algorithm ID:** pdal:boundary

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Density

Exports a raster file based on the number of points within each raster cell - useful for quality checking of point cloud datasets.

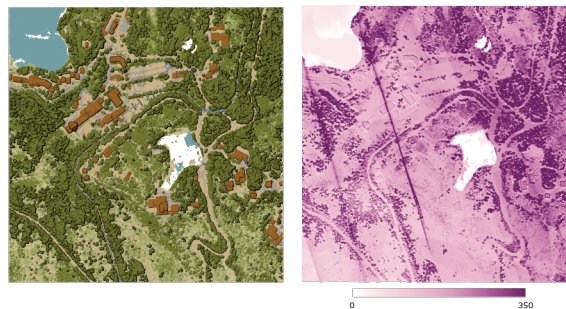


Abb. 28.16: Point density (number of points per 2x2 m) as a raster

## Parameters

### Basic parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to export
<b>Resolution of the density raster</b>	RESOLUTION	[number] Default: 1.0	Cell size of the output raster
<b>Tile size for parallel runs</b>	TILE_SIZE	[number] Default: 1000	Size of the tiles to split the data into for parallel runs
<b>Density</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the raster file to export the data to. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Label	Name	Type	Description
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>X origin of a tile for parallel runs</b> Optional	ORIGIN_X	[number]	
<b>Y origin of a tile for parallel runs</b> Optional	ORIGIN_Y	[number]	

## Outputs

Label	Name	Type	Description
<b>Density</b>	OUTPUT	[raster]	Output raster layer with number of points within each cell. Currently supported format is .TIF.

## Python code

**Algorithm ID:** pdal:density

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.



filter

Extracts point from the input point cloud which match PDAL expression and/or are inside of a cropping rectangle.

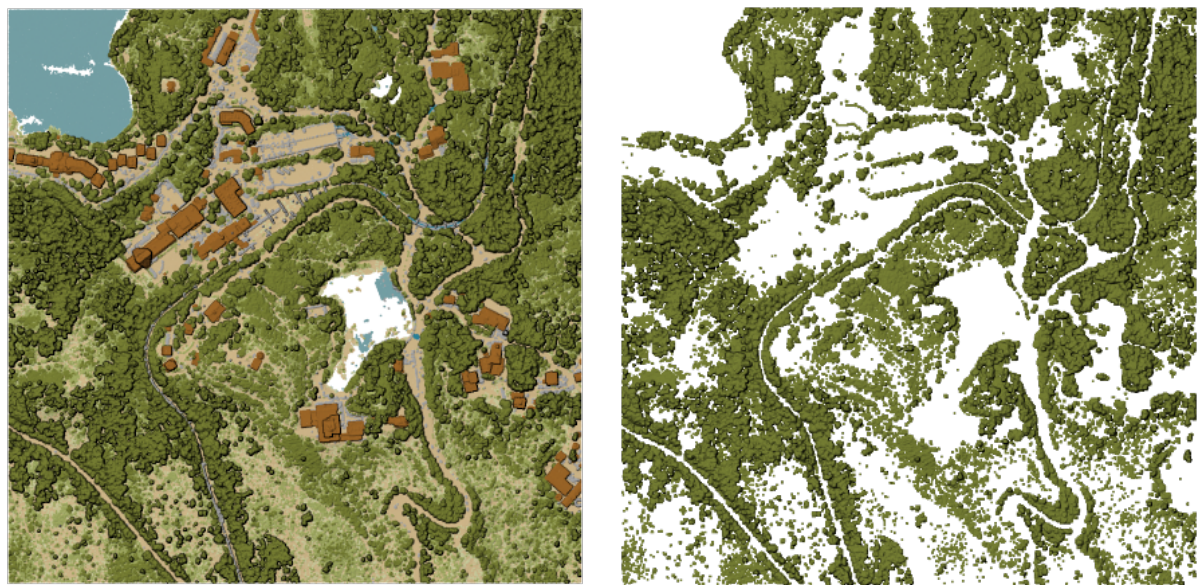


Abb. 28.17: Filtering of high vegetation class from an input point cloud layer

Parameters

Label	Name	Type	Description
<b>Input layer</b>	INPUT	[point cloud]	Input point cloud layer to export
<b>Filter expression</b> Optional	FILTER_EXPRESS	[expression]	A <i>PDAL expression</i> for selecting a subset of features in the point cloud data
<b>Cropping extent</b> Optional	FILTER_EXTENT	[extent]	A map extent for selecting a subset of features in the point cloud data Available methods are: <ul style="list-style-type: none"><li>• Calculate from layer...: uses extent of a layer loaded in the current project</li><li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li><li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li><li>• Use map canvas extent</li><li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li><li>• Enter the coordinates as xmin, xmax, ymin, ymax</li></ul>
<b>Filtered</b>	OUTPUT	[point cloud] Default: [Save to temporary file]	Specify the point cloud file to export the data to. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Outputs

Label	Name	Type	Description
<b>Filtered</b>	OUTPUT	[point cloud]	Output point cloud layer with the filtered features of the input point cloud layer. Currently supported formats are .LAS, .LAZ, .COPC.LAZ and .VPC.

## Python code

**Algorithm ID:** pdal:filter

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### 28.1.15 Raster analysis

#### Cell stack percent rank from value

Calculates the cell-wise percentrank value of a stack of rasters based on a single input value and writes them to an output raster.

At each cell location, the specified value is ranked among the respective values in the stack of all overlaid and sorted cell values from the input rasters. For values outside of the stack value distribution, the algorithm returns NoData because the value cannot be ranked among the cell values.

There are two methods for percentile calculation:

- Inclusive linear interpolation (PERCENTRANK.INC)
- Exclusive linear interpolation (PERCENTRANK.EXC)

The linear interpolation methods return the unique percent rank for different values. Both interpolation methods follow their counterpart methods implemented by [LibreOffice](#) or Microsoft Excel.

The output raster's extent and resolution is defined by a reference raster. Input raster layers that do not match the cell size of the reference raster layer will be resampled using nearest neighbor resampling. NoData values in any of the input layers will result in a NoData cell output if the „Ignore NoData values“ parameter is not set. The output raster data type will always be `Float32`.

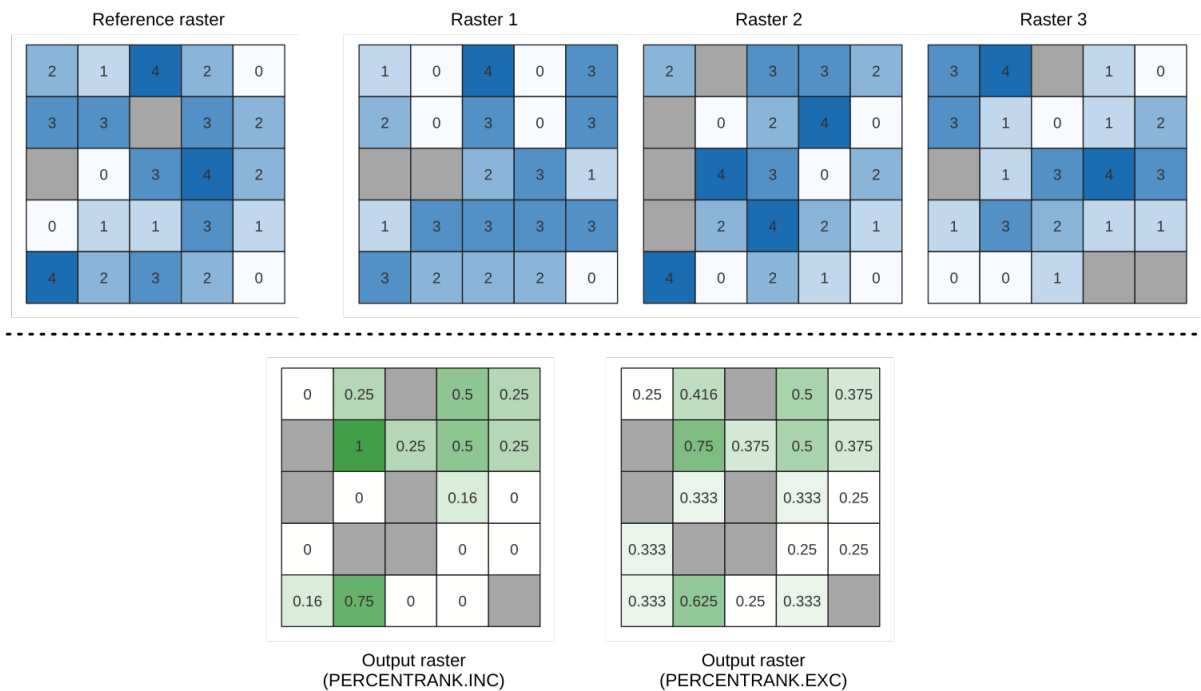


Abb. 28.18: Percent ranking Value = 1. NoData cells (grey) are ignored.

**Siehe auch:**

*Cell stack percentile, Cell stack percentrank from raster layer*

**Parameter****Basic parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters
<b>Method</b>	METHOD	[enumeration] Default: 0	Method for percentile calculation: <ul style="list-style-type: none"> <li>0 — Inclusive linear interpolation (PERCENTRANK.INC)</li> <li>1 — Exclusive linear interpolation (PERCENTRANK.EXC)</li> </ul>
<b>Value</b>	VALUE	[number] Default: 10.0	Value to rank among the respective values in the stack of all overlaid and sorted cell values from the input rasters
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: True	If unchecked, any NoData cells in the input layers will result in a NoData cell in the output raster
<b>Reference layer</b>	REFERENCE_LAYER	[raster]	The reference layer for the output layer creation (extent, CRS, pixel dimensions)

Fortsetzung auf der nächsten Seite

Tab. 28.44 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output value</b>	<b>NoData</b> OUTPUT_NODATA_	[number] Default: -9999.0	Value to use for NoData in the output layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer

### Pythoncode

**Algorithm ID:** native:cellstackpercentrankfromvalue

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Cell stack percentile

Calculates the cell-wise percentile value of a stack of rasters and writes the results to an output raster. The percentile to return is determined by the percentile input value (ranges between 0 and 1). At each cell location, the specified percentile is obtained using the respective value from the stack of all overlaid and sorted cell values of the input rasters.

There are three methods for percentile calculation:

- Nearest rank: returns the value that is nearest to the specified percentile
- Inclusive linear interpolation (PERCENTRANK.INC)
- Exclusive linear interpolation (PERCENTRANK.EXC)

The linear interpolation methods return the unique values for different percentiles. Both interpolation methods follow their counterpart methods implemented by [LibreOffice](#) or Microsoft Excel.

The output raster's extent and resolution is defined by a reference raster. Input raster layers that do not match the cell size of the reference raster layer will be resampled using nearest neighbor resampling. NoData values in any of the input layers will result in a NoData cell output if the „Ignore NoData values“ parameter is not set. The output raster data type will always be `Float32`.

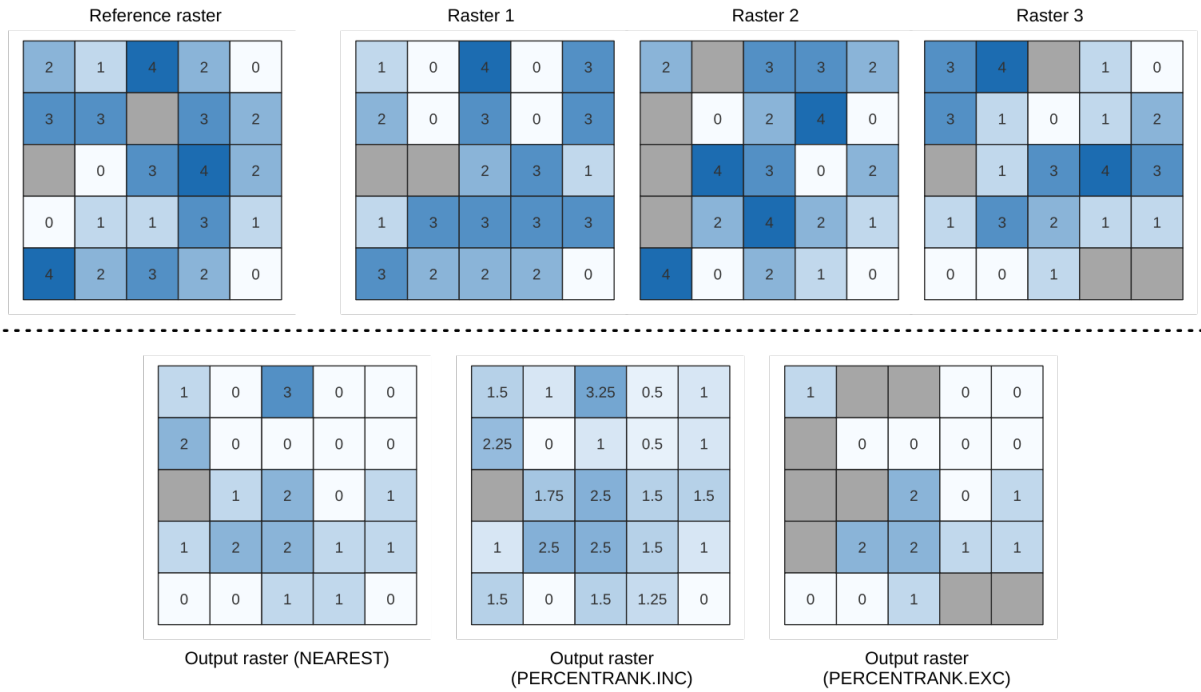


Abb. 28.19: Percentile = 0.25. NoData cells (grey) are ignored.

**Siehe auch:**

*Cell stack percentile, Cell stack percentrank from raster layer*

**Parameter**

**Basic parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters
<b>Method</b>	METHOD	[enumeration] Default: 0	Method for percentile calculation: <ul style="list-style-type: none"><li>• 0 — Nearest rank: returns the value that is nearest to the specified percentile</li><li>• 1 — Inclusive linear interpolation (PERCENTILE.INC)</li><li>• 2 — Exclusive linear interpolation (PERCENTILE.EXC)</li></ul>

Fortsetzung auf der nächsten Seite

Tab. 28.46 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Percentile</b>	VALUE	[number] Default: 0.25	Value to rank among the respective values in the stack of all overlaid and sorted cell values from the input rasters. Between 0 and 1.
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: True	If unchecked, any NoData cells in the input layers will result in a NoData cell in the output raster
<b>Reference layer</b>	REFERENCE_LAYER	[raster]	The reference layer for the output layer creation (extent, CRS, pixel dimensions)
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	OUTPUT_NODATA_	[number] Default: -9999.0	Value to use for NoData in the output layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer

### Pythoncode

**Algorithm ID:** native:cellstackpercentile

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Cell stack percentrank from raster layer

Calculates the cell-wise percentrank value of a stack of rasters based on an input value raster and writes them to an output raster.

At each cell location, the current value of the value raster is ranked among the respective values in the stack of all overlaid and sorted cell values of the input rasters. For values outside of the the stack value distribution, the algorithm returns NoData because the value cannot be ranked among the cell values.

There are two methods for percentile calculation:

- Inclusive linear interpolation (PERCENTRANK.INC)
- Exclusive linear interpolation (PERCENTRANK.EXC)

The linear interpolation methods return the unique values for different percentiles. Both interpolation methods follow their counterpart methods implemented by [LibreOffice](#) or Microsoft Excel.

The output raster's extent and resolution is defined by a reference raster. Input raster layers that do not match the cell size of the reference raster layer will be resampled using nearest neighbor resampling. NoData values in any of the input layers will result in a NoData cell output if the „Ignore NoData values“ parameter is not set. The output raster data type will always be `Float32`.

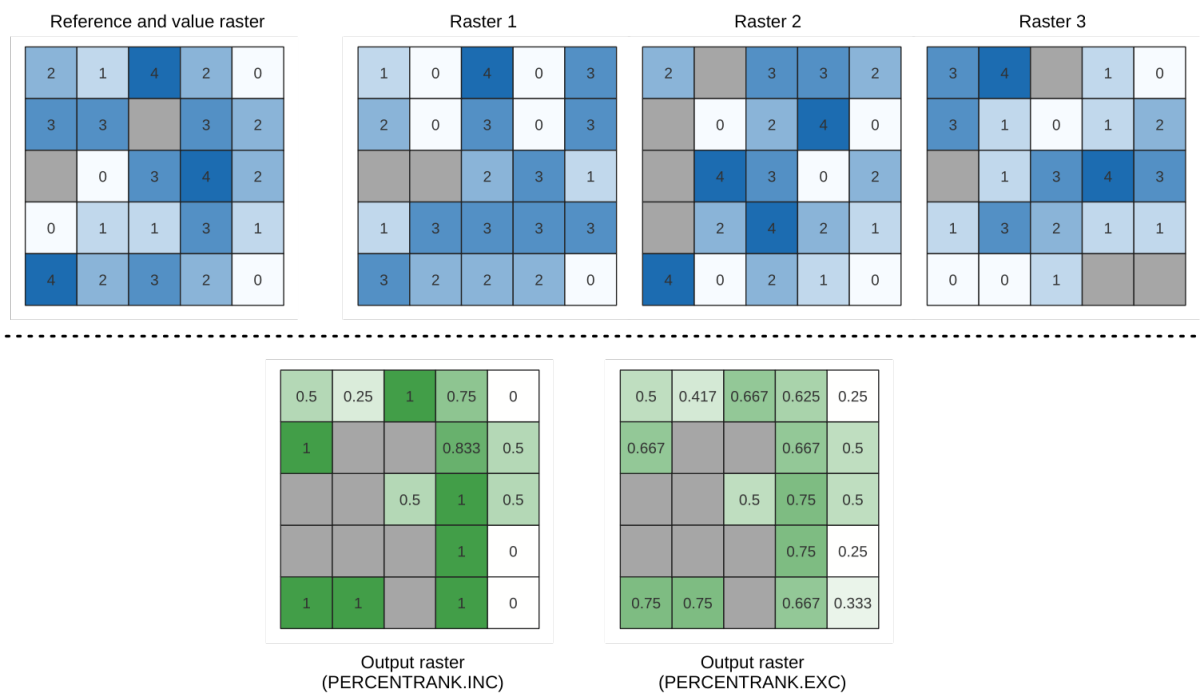


Abb. 28.20: Ranking the value raster layer cells. NoData cells (grey) are ignored.

### Siehe auch:

*Cell stack percentile, Cell stack percent rank from value*



## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters
<b>Value raster layer</b>	INPUT_VALUE_RA	[raster]	The layer to rank the values among the stack of all overlaid layers
<b>Value raster band</b>	VALUE_RASTER_B	[integer] Default: 1	Band of the „value raster layer“ to compare to
<b>Method</b>	METHOD	[enumeration] Default: 0	Method for percentile calculation: <ul style="list-style-type: none"> <li>• 0 — Inclusive linear interpolation (PERCENTRANK.INC)</li> <li>• 1 — Exclusive linear interpolation (PERCENTRANK.EXC)</li> </ul>
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: True	If unchecked, any NoData cells in the input layers will result in a NoData cell in the output raster
<b>Reference layer</b>	REFERENCE_LAYE	[raster]	The reference layer for the output layer creation (extent, CRS, pixel dimensions)
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	OUTPUT_NODATA_	[number] Default: -9999.0	Value to use for NoData in the output layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer



## Pythoncode

**Algorithm ID:** native:cellstackpercentrankfromrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Cell statistics

Computes per-cell statistics based on input raster layers and for each cell writes the resulting statistics to an output raster. At each cell location, the output value is defined as a function of all overlaid cell values of the input rasters.

By default, a NoData cell in ANY of the input layers will result in a NoData cell in the output raster. If the *Ignore NoData values* option is checked, then NoData inputs will be ignored in the statistic calculation. This may result in NoData output for locations where all cells are NoData.

The *Reference layer* parameter specifies an existing raster layer to use as a reference when creating the output raster. The output raster will have the same extent, CRS, and pixel dimensions as this layer.

**Calculation details:** Input raster layers that do not match the cell size of the reference raster layer will be resampled using nearest neighbor resampling. The output raster data type will be set to the most complex data type present in the input datasets except when using the functions Mean, Standard deviation and Variance (data type is always Float32 or Float64 depending on input float type) or Count and Variety (data type is always Int32).

- **Count:** The count statistic will always result in the number of cells without NoData values at the current cell location.
- **Median:** If the number of input layers is even, the median will be calculated as the arithmetic mean of the two middle values of the ordered cell input values.
- **Minority/Majority:** If no unique minority or majority could be found, the result is NoData, except all input cell values are equal.

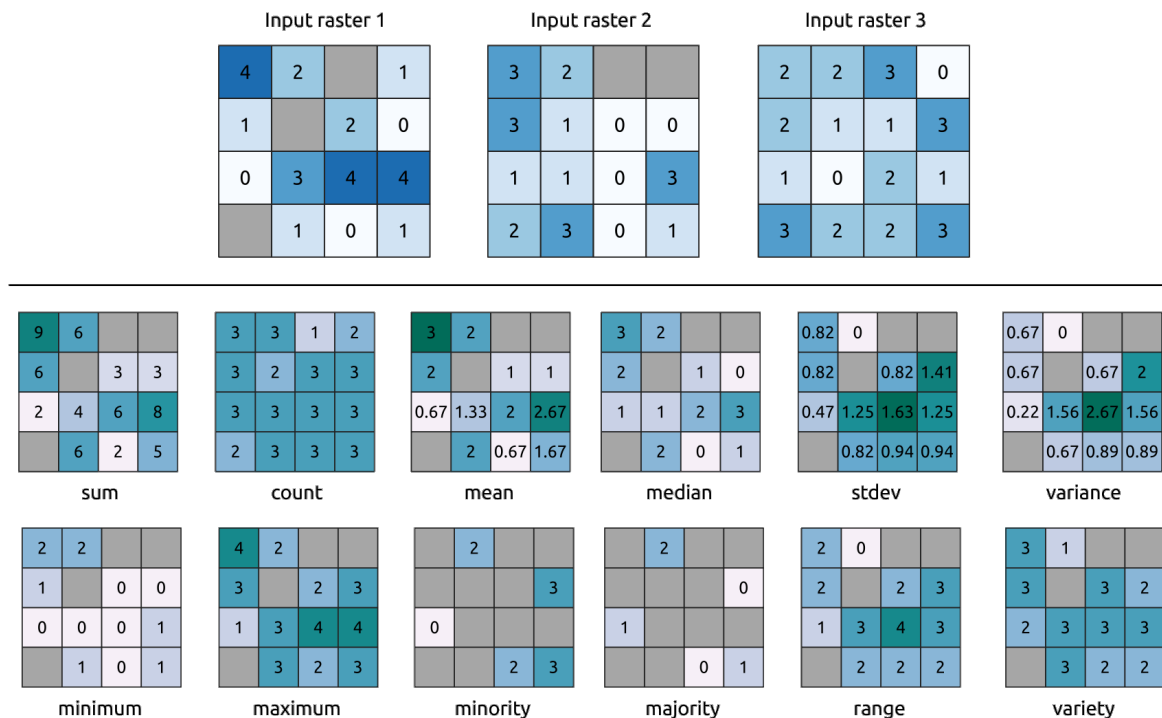


Abb. 28.21: Example with all the statistic functions. NoData cells (grey) are taken into account.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	Input raster layers
<b>Statistic</b>	STATISTIC	[enumeration] Default: 0	Available statistics. Options: <ul style="list-style-type: none"> <li>0 — Sum</li> <li>1 — Count</li> <li>2 — Mean</li> <li>3 — Median</li> <li>4 — Standard deviation</li> <li>5 — Variance</li> <li>6 — Minimum</li> <li>7 — Maximum</li> <li>8 — Minority (least common value)</li> <li>9 — Majority (most common value)</li> <li>10 — Range (max - min)</li> <li>11 — Variety (unique value count)</li> </ul>
<b>Ignore values</b>	IGNORE_NODATA	[boolean] Default: True	Calculate statistics also for all cells stacks, ignoring NoData occurrence.
<b>Reference layer</b>	REF_LAYER	[raster]	The reference layer to create the output layer from (extent, CRS, pixel dimensions)
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output value</b> Optional	<b>NoData</b> OUTPUT_NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[integer]	The number of rows in the output raster layer
<b>Output raster</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer

## Pythoncode

**Algorithm ID:** native:cellstatistics

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Equal to frequency

Evaluates on a cell-by-cell basis the frequency (number of times) the values of an input stack of rasters are equal to the value of a value layer. The output raster extent and resolution are defined by the input raster layer and is always of `Int32` type.

If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters - use GDAL to use other bands in the analysis. The output NoData value can be set manually.

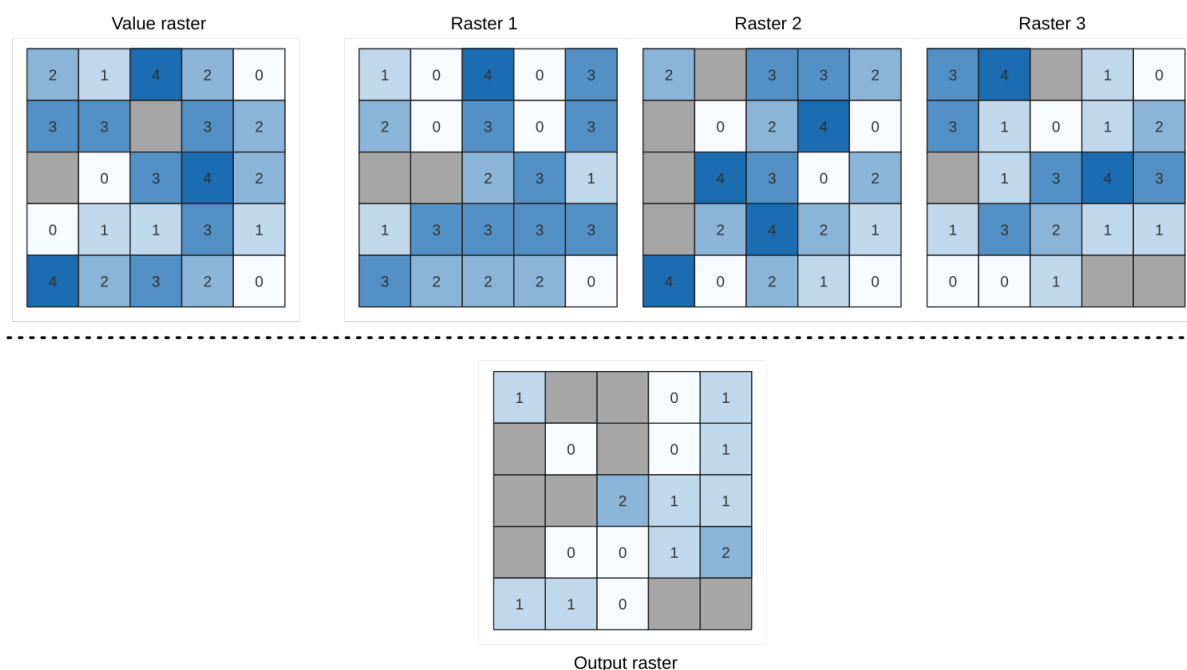


Abb. 28.22: For each cell in the output raster, the value represents the number of times that the corresponding cells in the list of rasters are the same as the value raster. NoData cells (grey) are taken into account.

#### Siehe auch:

*Greater than frequency, Less than frequency*

#### Parameter

##### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input value raster</b>	INPUT_VALUE_RA	[raster]	The input value layer serves as reference layer for the sample layers
<b>Value raster band</b>	INPUT_VALUE_RA	[raster band] Default: The first band of the raster layer	Select the band you want to use as sample
<b>Input raster layers</b>	INPUT_RASTERS	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: False	If unchecked, any NoData cells in the value raster or the data layer stack will result in a NoData cell in the output raster
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output value</b> Optional	NoData OUTPUT_NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Count of cells with equal value occurrences</b>	FOUND_LOCATION	[number]	
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[number]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer
<b>Mean frequency at valid cell locations</b>	MEAN_FREQUENCY	[number]	
<b>Count of value occurrences</b>	OCCURRENCE_COU	[number]	
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer

## Pythoncode

**Algorithm ID:** native:equaltofrequency

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (gaussian membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Gaussian membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The gaussian membership function is defined as  $\mu(x) = e^{-f1*(x-f2)^2}$ , where *f1* is the spread and *f2* the midpoint.

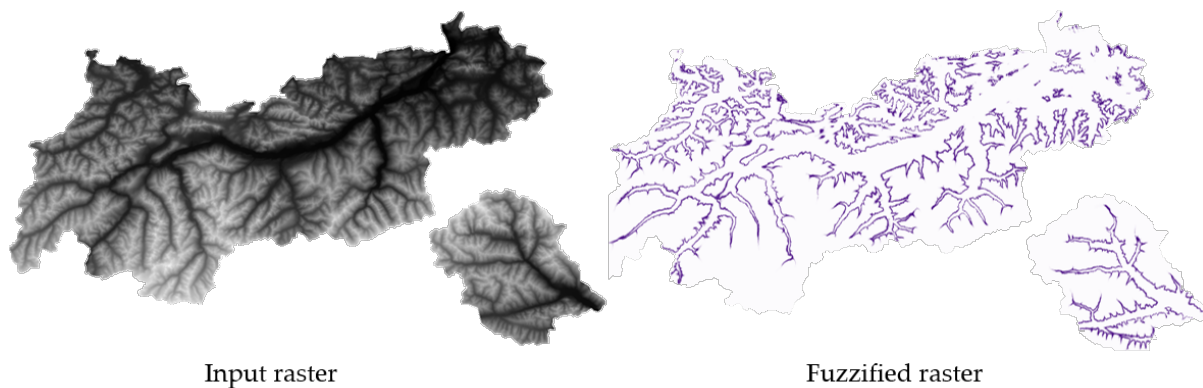


Abb. 28.23: Fuzzify raster example. Input raster source: Land Tirol - data.tirol.gv.at.

#### Siehe auch:

*Fuzzify raster (large membership), Fuzzify raster (linear membership), Fuzzify raster (near membership), Fuzzify raster (power membership), Fuzzify raster (small membership)*

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Function midpoint</b>	FUZZYMIDPOINT	[number] Default: 10	Midpoint of the gaussian function
<b>Function spread</b>	FUZZYSPREAD	[number] Default: 0.01	Spread of the gaussian function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrastergaussianmembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (large membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Large membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The large membership function

is defined as 
$$\mu(x) = \frac{1}{1 + \left(\frac{x}{f2}\right)^{-f1}}$$
, where *f1* is the spread and *f2* the midpoint.

**Siehe auch:**

*Fuzzify raster (gaussian membership), Fuzzify raster (linear membership), Fuzzify raster (near membership), Fuzzify raster (power membership), Fuzzify raster (small membership)*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Function midpoint</b>	FUZZYMIDPOINT	[number] Default: 50	Midpoint of the large function
<b>Function spread</b>	FUZZYSPREAD	[number] Default: 5	Spread of the large function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrasterlargemembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (linear membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Linear membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The linear function is defined as

$$\mu(X) \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a < x < b \\ 1 & x \geq b \end{cases}, \text{ where } a \text{ is the low bound and } b \text{ the high bound.}$$

This equation assigns membership values using a linear transformation for pixel values between the low and high bounds. Pixels values smaller than the low bound are given 0 membership whereas pixel values greater than the high bound are given 1 membership.

### Siehe auch:

*Fuzzify raster (gaussian membership), Fuzzify raster (large membership), Fuzzify raster (near membership), Fuzzify raster (power membership), Fuzzify raster (small membership)*



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Low fuzzy membership bound</b>	FUZZYLOWBOUND	[number] Default: 0	Low bound of the linear function
<b>High fuzzy membership bound</b>	FUZZYHIGHBOUND	[number] Default: 1	High bound of the linear function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrasterlinearmembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (near membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Near membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The near membership function

is defined as  $\mu(x) = \frac{1}{1 + f1 * (x - f2)^2}$ , where  $f1$  is the spread and  $f2$  the midpoint.

**Siehe auch:**

*Fuzzify raster (gaussian membership), Fuzzify raster (large membership), Fuzzify raster (linear membership), Fuzzify raster (power membership), Fuzzify raster (small membership)*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Function midpoint</b>	FUZZYMIDPOINT	[number] Default: 50	Midpoint of the near function
<b>Function spread</b>	FUZZYSPREAD	[number] Default: 0.01	Spread of the near function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrasternearmembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (power membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Power membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The power function is defined as

$$\mu(x) = \begin{cases} 0 & x \leq a \\ \left(\frac{x-a}{b-a}\right)^{f1} & a < x < b \\ 1 & x \geq b \end{cases}$$

, where  $a$  is the low bound,  $b$  is the high bound, and  $f1$  the exponent. This equation assigns membership values using the power transformation for pixel values between the low and high bounds. Pixels values smaller than the low bound are given 0 membership whereas pixel values greater than the high bound are given 1 membership.

**Siehe auch:**

*Fuzzify raster (gaussian membership), Fuzzify raster (large membership), Fuzzify raster (linear membership), Fuzzify raster (near membership), Fuzzify raster (small membership)*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Low fuzzy membership bound</b>	FUZZYLOWBOUND	[number] Default: 0	Low bound of the power function
<b>High fuzzy membership bound</b>	FUZZYHIGHBOUND	[number] Default: 1	High bound of the power function
<b>High fuzzy membership bound</b>	FUZZYEXPONENT	[number] Default: 2	Exponent of the power function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrasterpowermembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fuzzify raster (small membership)

Transforms an input raster to a fuzzified raster by assigning a membership value to each pixel, using a Small membership function. Membership values range from 0 to 1. In the fuzzified raster, a value of 0 implies no membership of the defined fuzzy set, whereas a value of 1 means full membership. The small membership function

is defined as 
$$\mu(x) = \frac{1}{1 + \left(\frac{x}{f2}\right)^{f1}}$$
, where *f1* is the spread and *f2* the midpoint.

**Siehe auch:**

*Fuzzify raster (gaussian membership), Fuzzify raster (large membership) Fuzzify raster (linear membership), Fuzzify raster (near membership), Fuzzify raster (power membership)*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Input raster layer
<b>Band Number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that you want to fuzzify.
<b>Function midpoint</b>	FUZZYMIDPOINT	[number] Default: 50	Midpoint of the small function
<b>Function spread</b>	FUZZYSPREAD	[number] Default: 5	Spread of the small function
<b>Fuzzified raster</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fuzzified raster</b>	OUTPUT	[same as input]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:fuzzifyrastersmallmembership

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Greater than frequency

Evaluates on a cell-by-cell basis the frequency (number of times) the values of an input stack of rasters are equal to the value of a value raster. The output raster extent and resolution is defined by the input raster layer and is always of `Int32` type.

If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters - use GDAL to use other bands in the analysis. The output NoData value can be set manually.

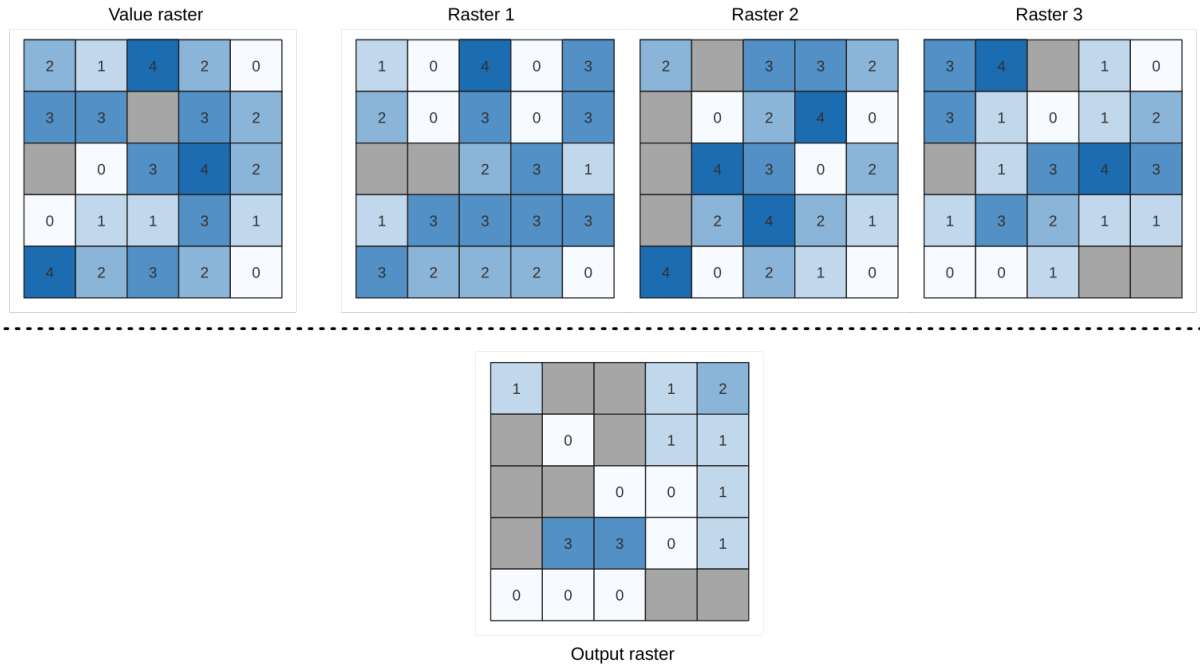


Abb. 28.24: For each cell in the output raster, the value represents the number of times that the corresponding cells in the list of rasters are greater than the value raster. NoData cells (grey) are taken into account.

Siehe auch:

*Equal to frequency, Less than frequency*

Parameter

Basic parameters

Bezeichnung	Name	Typ	Beschreibung
Input value raster	INPUT_VALUE_RA	[raster]	The input value layer serves as reference layer for the sample layers
Value raster band	INPUT_VALUE_RA	[raster band] Default: The first band of the raster layer	Select the band you want to use as sample
Input raster layers	INPUT_RASTERS	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters

Fortsetzung auf der nächsten Seite

Tab. 28.54 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: False	If unchecked, any NoData cells in the value raster or the data layer stack will result in a NoData cell in the output raster
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b> Optional	OUTPUT_NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Count of cells with equal value occurrences</b>	FOUND_LOCATION	[number]	
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[number]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer
<b>Mean frequency at valid cell locations</b>	MEAN_FREQUENCY	[number]	
<b>Count of value occurrences</b>	OCCURRENCE_COU	[number]	
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer

### Pythoncode

**Algorithm ID:** native:greaterthanfrequency

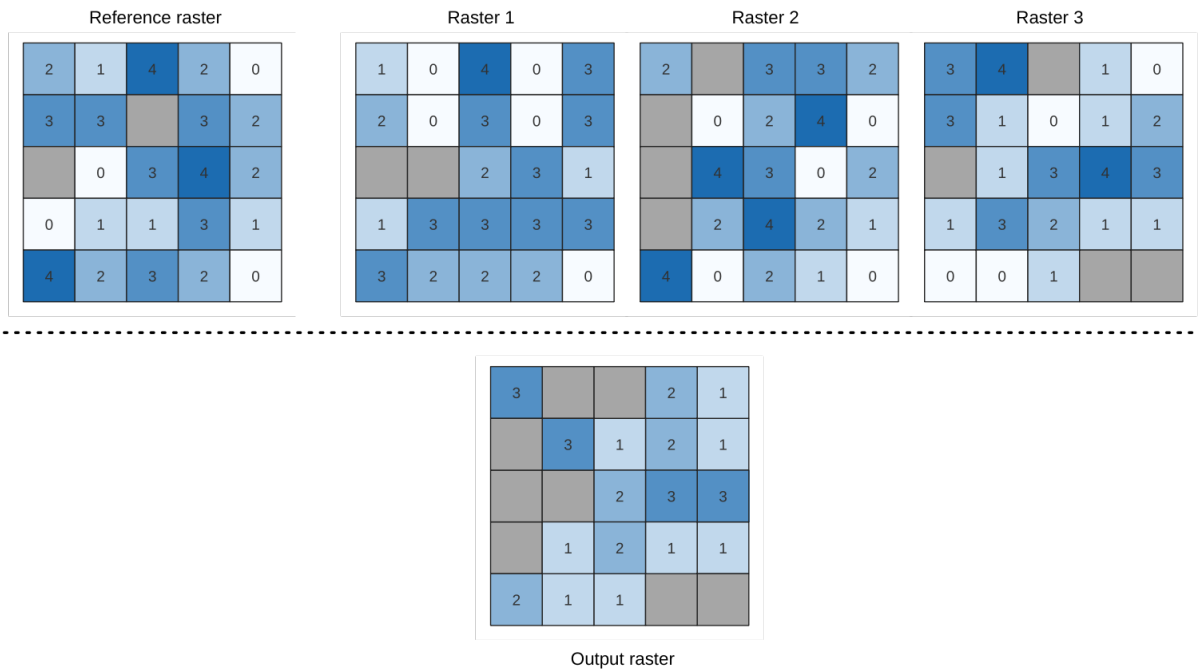
```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Highest position in raster stack

Evaluates on a cell-by-cell basis the position of the raster with the highest value in a stack of rasters. Position counts start with 1 and range to the total number of input rasters. The order of the input rasters is relevant for the algorithm. If multiple rasters feature the highest value, the first raster will be used for the position value.

If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters - use GDAL to use other bands in the analysis. Any NoData cells in the raster layer stack will result in a NoData cell in the output raster unless the „ignore NoData“ parameter is checked. The output NoData value can be set manually. The output rasters extent and resolution is defined by a reference raster layer and is always of `Int32` type.



**Siehe auch:**

*Lowest position in raster stack*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input raster layers</b>	INPUT_RASTERS	[raster] [list]	List of raster layers to compare with
<b>Reference layer</b>	REFERENCE_LAYE	[raster]	The reference layer for the output layer creation (extent, CRS, pixel dimensions)
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: False	If unchecked, any NoData cells in the data layer stack will result in a NoData cell in the output raster
<b>Output layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster containing the result. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>



## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output value</b>	<b>NoData</b> OUTPUT_NODATA_	[number] Default: -9999.0	Value to use for NoData in the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:highestpositioninrasterstack

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Less than frequency

Evaluates on a cell-by-cell basis the frequency (number of times) the values of an input stack of rasters are less than the value of a value raster. The output raster extent and resolution is defined by the input raster layer and is always of `Int32` type.

If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters - use GDAL to use other bands in the analysis. The output NoData value can be set manually.

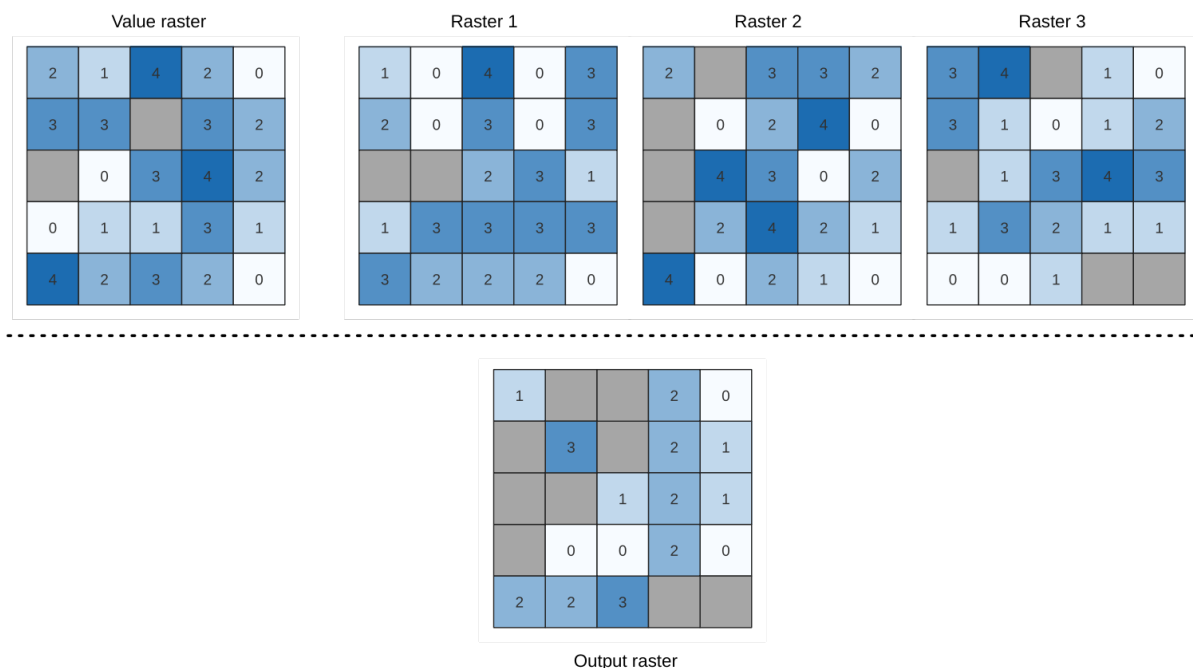


Abb. 28.25: For each cell in the output raster, the value represents the number of times that the corresponding cells in the list of rasters are less than the value raster. NoData cells (grey) are taken into account.

#### Siehe auch:

*Equal to frequency, Greater than frequency*

#### Parameter

##### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input value raster</b>	INPUT_VALUE_RA	[raster]	The input value layer serves as reference layer for the sample layers
<b>Value raster band</b>	INPUT_VALUE_RA	[raster band] Default: The first band of the raster layer	Select the band you want to use as sample
<b>Input raster layers</b>	INPUT_RASTERS	[raster] [list]	Raster layers to evaluate. If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: False	If unchecked, any NoData cells in the value raster or the data layer stack will result in a NoData cell in the output raster
<b>Output layer</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output value</b> Optional	OUTPUT_NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Count of cells with equal value occurrences</b>	FOUND_LOCATION	[number]	
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[number]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer
<b>Mean frequency at valid cell locations</b>	MEAN_FREQUENCY	[number]	
<b>Count of value occurrences</b>	OCCURRENCE_COU	[number]	
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer

## Pythoncode

**Algorithm ID:** native:lessthanfrequency

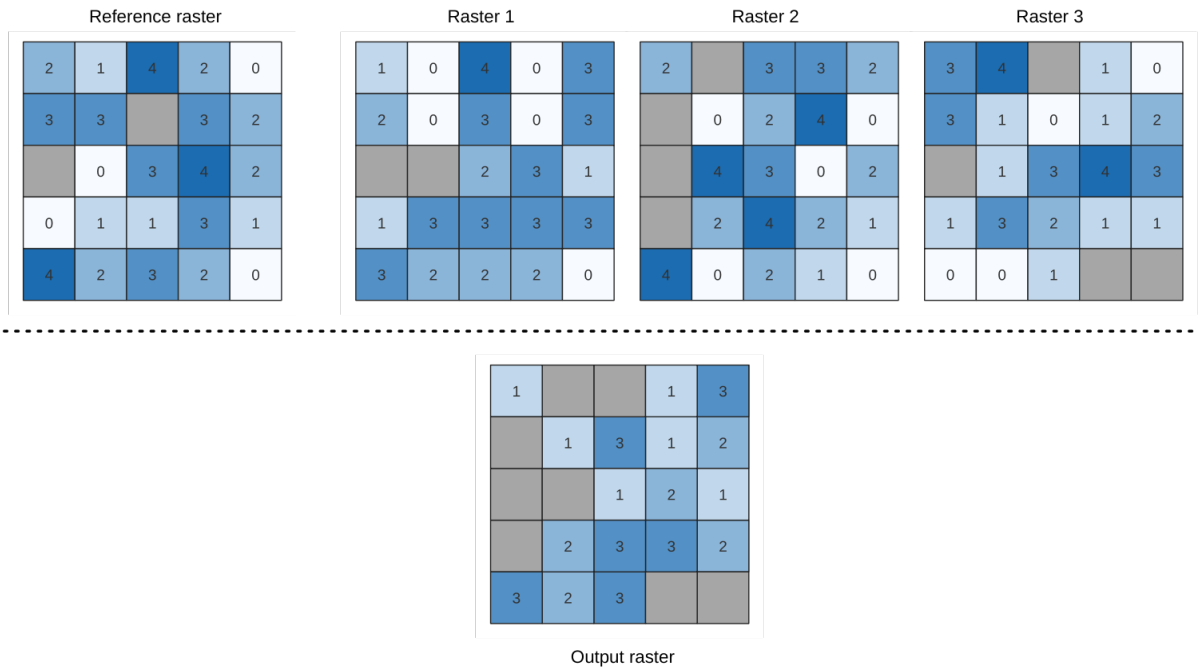
```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Lowest position in raster stack

Evaluates on a cell-by-cell basis the position of the raster with the lowest value in a stack of rasters. Position counts start with 1 and range to the total number of input rasters. The order of the input rasters is relevant for the algorithm. If multiple rasters feature the lowest value, the first raster will be used for the position value.

If multiband rasters are used in the data raster stack, the algorithm will always perform the analysis on the first band of the rasters - use GDAL to use other bands in the analysis. Any NoData cells in the raster layer stack will result in a NoData cell in the output raster unless the „ignore NoData“ parameter is checked. The output NoData value can be set manually. The output rasters extent and resolution is defined by a reference raster layer and is always of `Int32` type.



Siehe auch:

*Highest position in raster stack*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input raster layers</b>	INPUT_RASTERS	[raster] [list]	List of raster layers to compare with
<b>Reference layer</b>	REFERENCE_LAYER	[raster]	The reference layer for the output layer creation (extent, CRS, pixel dimensions)
<b>Ignore NoData values</b>	IGNORE_NODATA	[boolean] Default: False	If unchecked, any NoData cells in the data layer stack will result in a NoData cell in the output raster
<b>Output layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster containing the result. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	OUTPUT_NODATA_	[number] Default: -9999.0	Value to use for NoData in the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXE	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer

## Pythoncode

**Algorithm ID:** native:lowestpositioninrasterstack

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster boolean AND

Calculates the boolean AND for a set of input rasters. If all of the input rasters have a non-zero value for a pixel, that pixel will be set to 1 in the output raster. If any of the input rasters have 0 values for the pixel it will be set to 0 in the output raster.

The reference layer parameter specifies an existing raster layer to use as a reference when creating the output raster. The output raster will have the same extent, CRS, and pixel dimensions as this layer.

By default, a NoData pixel in ANY of the input layers will result in a NoData pixel in the output raster. If the *Treat NoData values as false* option is checked, then NoData inputs will be treated the same as a 0 input value.

**Siehe auch:**

[Raster boolean OR](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	List of input raster layers
<b>Reference layer</b>	REF_LAYER	[raster]	The reference layer to create the output layer from (extent, CRS, pixel dimensions)
<b>Treat NoData values as false</b>	NODATA_AS_FALS	[boolean] Default: False	Treat NoData values in the input files as 0 when performing the operation

Fortsetzung auf der nächsten Seite

Tab. 28.62 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster containing the result. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	Output raster data type. Options: <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help</i> ► <i>About</i> menu)</p>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer

Fortsetzung auf der nächsten Seite

Tab. 28.64 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>NoData pixel count</b>	NODATA_PIXEL_C	[integer]	The count of NoData pixels in the output raster layer
<b>True pixel count</b>	TRUE_PIXEL_COU	[integer]	The count of True pixels (value = 1) in the output raster layer
<b>False pixel count</b>	FALSE_PIXEL_CO	[integer]	The count of False pixels (value = 0) in the output raster layer
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result

## Pythoncode

**Algorithm ID:** native:rasterbooleanand

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster boolean OR

Calculates the boolean OR for a set of input rasters. If all of the input rasters have a zero value for a pixel, that pixel will be set to 0 in the output raster. If any of the input rasters have 1 values for the pixel it will be set to 1 in the output raster.

The reference layer parameter specifies an existing raster layer to use as a reference when creating the output raster. The output raster will have the same extent, CRS, and pixel dimensions as this layer.

By default, a NoData pixel in ANY of the input layers will result in a NoData pixel in the output raster. If the *Treat NoData values as false* option is checked, then NoData inputs will be treated the same as a 0 input value.

**Siehe auch:**

[Raster boolean AND](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	List of input raster layers
<b>Reference layer</b>	REF_LAYER	[raster]	The reference layer to create the output layer from (extent, CRS, pixel dimensions)
<b>Treat NoData values as false</b>	NODATA_AS_FALS	[boolean] Default: False	Treat NoData values in the input files as 0 when performing the operation
<b>Output layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster containing the result. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	NO_DATA	[number] Default: -9999.0	Value to use for NoData in the output layer
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	Output raster data type. Options: <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (quint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <i>Help</i> ► <i>About</i> menu)

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>CRS authority identifier</b>	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_PIXEL	[integer]	The number of rows in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer
<b>NoData pixel count</b>	NODATA_PIXEL_COUNT	[integer]	The count of NoData pixels in the output raster layer
<b>True pixel count</b>	TRUE_PIXEL_COUNT	[integer]	The count of True pixels (value = 1) in the output raster layer
<b>False pixel count</b>	FALSE_PIXEL_COUNT	[integer]	The count of False pixels (value = 0) in the output raster layer
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer containing the result



## Pythoncode

**Algorithm ID:** native:rasterbooleanor

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster calculator

Performs algebraic operations using raster layers.

The resulting layer will have its values computed according to an expression. The expression can contain numerical values, operators and references to any of the layers in the current project.

**Siehe auch:**

*Raster calculator (virtual), Raster calculator, Rasterrechner*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	List of input raster layers
<b>Expression</b>	EXPRESSION	[expression]	<i>Raster-based expression</i> that will be used to calculate the output raster layer.
<b>Output extent</b> Optional	EXTENT	[extent]	Specify the spatial extent of the output raster layer. If the extent is not specified, the minimum extent that covers all the selected reference layers will be used. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Output cell size</b> (leave empty to set automatically) Optional	CELLSIZE	[number]	Cell size of the output raster layer. If the cell size is not specified, the minimum cell size of the selected reference layer(s) will be used. The cell size will be the same for the X and Y axes.
<b>Output CRS</b> Optional	CRS	[crs]	CRS of the output raster layer. If the output CRS is not specified, the CRS of the first reference layer will be used.

Fortsetzung auf der nächsten Seite

Tab. 28.68 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[raster]	Output raster file with the calculated values.

## Pythoncode

**Algorithm ID:** native:rastercalc

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster calculator (virtual)

Performs algebraic operations using raster layers and generates in-memory result.

The resulting layer will have its values computed according to an expression. The expression can contain numerical values, operators and references to any of the layers in the current project.

A virtual raster layer is a raster layer defined by its URI and whose pixels are calculated on-the-fly. It's not a new file on disk; the virtual layer is still connected to the rasters used in the calculation meaning that deleting or moving these rasters would break it. A *Layer name* can be provided, otherwise the calculation expression is used as such. Removing the virtual layer from the project deletes it, and it can be made persistent in file using the layer *Export ► Save as...* contextual menu.

**Siehe auch:**

[Raster calculator](#), [Raster calculator](#), [Rasterrechner](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	List of input raster layers
<b>Expression</b>	EXPRESSION	[expression]	<a href="#">Raster-based expression</a> that will be used to calculate the output raster layer.

Fortsetzung auf der nächsten Seite

Tab. 28.69 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output extent</b> Optional	EXTENT	[extent]	Specify the spatial extent of the output raster layer. If the extent is not specified, the minimum extent that covers all the selected reference layers will be used. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Output cell size</b> (leave empty to set automatically) Optional	CELLSIZE	[number]	Cell size of the output raster layer. If the cell size is not specified, the minimum cell size of the selected reference layer(s) will be used. The cell size will be the same for the X and Y axes.
<b>Output CRS</b> Optional	CRS	[crs]	CRS of the output raster layer. If the output CRS is not specified, the CRS of the first reference layer will be used.
<b>Output layer name</b> Optional	LAYER_NAME	[string]	The name to assign to the generated layer. If not set, the text of the calculation expression is used.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[raster]	Output virtual raster layer with the calculated values.

## Pythoncode

**Algorithm ID:** native:virtualrastercalc

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster layer properties

Returns basic properties of the given raster layer, including the extent, size in pixels and dimensions of pixels (in map units), number of bands, and NoData value.

This algorithm is intended for use as a means of extracting these useful properties to use as the input values to other algorithms in a model - e.g. to allow to pass an existing raster's pixel sizes over to a GDAL raster algorithm.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b> Optional	BAND	[raster band] Default: Not set	Whether to also return properties of a specific band. If a band is specified, the noData value for the selected band is also returned.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Number of bands in raster</b>	BAND_COUNT	[number]	The number of bands in the raster
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The raster layer extent in the CRS
<b>Band has a NoData value set</b>	HAS_NODATA_VAL	[Boolean]	Indicates whether the raster layer has a value set for NoData pixels in the selected band
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[integer]	The number of columns in the raster layer
<b>Band NoData value</b>	NODATA_VALUE	[number]	The value (if set) of the NoData pixels in the selected band
<b>Pixel size (height) in map units</b>	PIXEL_HEIGHT	[integer]	Vertical size in map units of the pixel
<b>Pixel size (width) in map units</b>	PIXEL_WIDTH	[integer]	Horizontal size in map units of the pixel
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of rows in the raster layer
<b>Maximum coordinate</b>	x- X_MAX	[number]	
<b>Minimum coordinate</b>	x- X_MIN	[number]	
<b>Maximum coordinate</b>	y- Y_MAX	[number]	
<b>Minimum coordinate</b>	y- Y_MIN	[number]	

## Pythoncode

**Algorithm ID:** `native:rasterlayerproperties`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster layer statistics

Calculates basic statistics from the values in a given band of the raster layer. The output is loaded in the *Processing* ► *Results viewer* menu.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to get statistics for.
<b>Statistics</b>	OUTPUT_HTML_FI	[html] Default: [Save to temporary file]	Specification of the output file: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Maximum value</b>	MAX	[number]	
<b>Mean value</b>	MEAN	[number]	
<b>Minimum value</b>	MIN	[number]	

Fortsetzung auf der nächsten Seite

Tab. 28.70 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Statistics</b>	OUTPUT_HTML_FI	[html]	The output file contains the following information: <ul style="list-style-type: none"> <li>Analyzed file: path of the raster layer</li> <li>Minimum value: minimum value of the raster</li> <li>Maximum value: maximum value of the raster</li> <li>Range: difference between the maximum and minimum values</li> <li>Sum: total sum of the values</li> <li>Mean value: mean of the values</li> <li>Standard deviation: standard deviation of the values</li> <li>Sum of the squares: sum of the squared differences of each observation from the overall mean</li> </ul>
<b>Range</b>	RANGE	[number]	
<b>Standard deviation</b>	STD_DEV	[number]	
<b>Sum</b>	SUM	[number]	
<b>Sum of the squares</b>	SUM_OF_SQUARES	[number]	

## Pythoncode

**Algorithm ID:** native:rasterlayerstatistics

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster layer unique values report

Returns the count and area of each unique value in a given raster layer.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to get statistics for.

Fortsetzung auf der nächsten Seite

Tab. 28.71 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Unique report</b>	OUTPUT_HTML_FI	[file] Default: [Save to temporary file]	Specification of the output file: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>
<b>Unique table</b>	OUTPUT_TABLE	[table] Default: [Skip output]	Specification of the table for unique values: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[integer]	The number of rows in the output raster layer
<b>NoData pixel count</b>	NODATA_PIXEL_C	[number]	The number of NoData pixels in the output raster layer
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[integer]	The count of pixels in the output raster layer
<b>Unique report</b>	OUTPUT_HTML_FI	[html]	The output HTML file contains the following information: <ul style="list-style-type: none"> <li>• Analyzed file: the path of the raster layer</li> <li>• Extent: xmin, ymin, xmax, ymax coordinates of the extent</li> <li>• Projection: projection of the layer</li> <li>• Width in pixels: number of columns and pixel width size</li> <li>• Height in pixels: number of rows and pixel width size</li> <li>• Total pixel count: count of all the pixels</li> <li>• NoData pixel count: count of pixels with NoData value</li> </ul>
<b>Unique table</b>	OUTPUT_TABLE	[table]	A table with three columns: <ul style="list-style-type: none"> <li>• <i>value</i>: pixel value</li> <li>• <i>count</i>: count of pixels with this value</li> <li>• <i>m²</i>: total area in square meters of pixels with this value.</li> </ul>
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[integer]	The number of columns in the output raster layer

## Pythoncode

**Algorithm ID:** `native:rasterlayeruniquevaluesreport`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster layer zonal statistics

Calculates statistics for a raster layer's values, categorized by zones defined in another raster layer.

**Siehe auch:**

[Zonal statistics](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband choose the band for which you want to calculate the statistics.
<b>Zones layer</b>	ZONES	[raster]	Raster layer defining zones. Zones are given by contiguous pixels having the same pixel value.
<b>Zones band number</b>	ZONES_BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that defines the zones
<b>Statistics</b>	OUTPUT_TABLE	[table] Default: [Create temporary layer]	Specification of the output report. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.



## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Reference layer</b> Optional	REF_LAYER	[enumeration] Default: 0	Raster layer used to calculate the centroids that will be used as reference when determining the zones in the output layer. One of: <ul style="list-style-type: none"> <li>0 — Input layer: zones are determined by sampling the zone raster layer value at the centroid of each pixel from the source raster layer</li> <li>1 — Zones layer: the input raster layer will be sampled at the centroid of each pixel from the zones raster layer</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>CRS authority identifier</b>	CRS_AUTHID	[string]	The coordinate reference system of the output raster layer
<b>Extent</b>	EXTENT	[string]	The spatial extent of the output raster layer
<b>Height in pixels</b>	HEIGHT_IN_Pixe	[integer]	The number of rows in the output raster layer
<b>NoData pixel count</b>	NODATA_PIXEL_C	[number]	The number of NoData pixels in the output raster layer
<b>Statistics</b>	OUTPUT_TABLE	[table]	The output layer contains the following information <b>for each zone</b> : <ul style="list-style-type: none"> <li>Area: the area in square raster units in the zone;</li> <li>Sum: the total sum of the pixel values in the zone;</li> <li>Count: the number of pixels in the zone;</li> <li>Min: the minimum pixel value in the zone;</li> <li>Max: the maximum pixel value in the zone;</li> <li>Mean: the mean of the pixel values in the zone;</li> </ul>
<b>Total pixel count</b>	TOTAL_PIXEL_CO	[number]	The count of pixels in the output raster layer
<b>Width in pixels</b>	WIDTH_IN_PIXEL	[number]	The number of columns in the output raster layer

## Pythoncode

**Algorithm ID:** native:rasterlayerzonalstats

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster surface volume

Calculates the volume under a raster surface relative to a given base level. This is mainly useful for Digital Elevation Models (DEM).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>INPUT layer</b>	INPUT	[raster]	Input raster, representing a surface
<b>Band number</b>	BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that shall define the surface.
<b>Base level</b>	LEVEL	[number] Default: 0.0	Define a base or reference value. This base is used in the volume calculation according to the <i>Method</i> parameter (see below).
<b>Method</b>	METHOD	[enumeration] Default: 0	Define the method for the volume calculation given by the difference between the raster pixel value and the <i>Base level</i> . Options: <ul style="list-style-type: none"> <li>• 0 — Count Only Above Base Level: only pixels above the base level will add to the volume.</li> <li>• 1 — Count Only Below Base Level: only pixels below the base level will add to the volume.</li> <li>• 2 — Subtract Volumes Below Base level: pixels above the base level will add to the volume, pixels below the base level will subtract from the volume.</li> <li>• 3 — Add Volumes Below Base level: Add the volume regardless whether the pixel is above or below the base level. This is equivalent to sum the absolute values of the difference between the pixel value and the base level.</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.76 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Surface report</b>	OUTPUT_HTML_FI	[html] Default: [Save to temporary file]	Specification of the output HTML report. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Surface table</b>	OUTPUT_TABLE	[table] Default: [Skip output]	Specification of the output table. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Volume</b>	VOLUME	[number]	The calculated volume
<b>Area</b>	AREA	[number]	The area in square map units
<b>Pixel_count</b>	PIXEL_COUNT	[number]	The total number of pixels that have been analyzed
<b>Surface report</b>	OUTPUT_HTML_FI	[html]	The output report (containing volume, area and pixel count) in HTML format
<b>Surface table</b>	OUTPUT_TABLE	[table]	The output table (containing volume, area and pixel count)

## Pythoncode

**Algorithm ID:** native:rastersurfacevolume

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Reclassify by layer

Reclassifies a raster band by assigning new class values based on the ranges specified in a vector table.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Raster layer to reclassify
<b>Band number</b>	RASTER_BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band you want to reclassify.
<b>Layer containing class breaks</b>	INPUT_TABLE	[vector: any]	Vector layer containing the values to use for classification.
<b>Minimum class value field</b>	MIN_FIELD	[tablefield: numeric]	Field with the minimum value of the range for the class. Use <code>-inf</code> to capture the lowest possible value.
<b>Maximum class value field</b>	MAX_FIELD	[tablefield: numeric]	Field with the maximum value of the range for the class. Use <code>inf</code> to capture the highest possible value.
<b>Output value field</b>	VALUE_FIELD	[tablefield: numeric]	Field with the value that will be assigned to the pixels that fall in the class (between the corresponding min and max values). Use <code>nan</code> to set the value of the range to NoData.
<b>Reclassified raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	NO_DATA	[number] Default: -9999.0	Value to apply to NoData values.
<b>Range boundaries</b>	RANGE_BOUNDARI	[enumeration] Default: 0	Defines comparison rules for the classification. Options: <ul style="list-style-type: none"> <li>• 0 — <code>min &lt; value &lt;= max</code></li> <li>• 1 — <code>min &lt;= value &lt; max</code></li> <li>• 2 — <code>min &lt;= value &lt;= max</code></li> <li>• 3 — <code>min &lt; value &lt; max</code></li> </ul>
<b>Use NoData when no range matches value</b>	NODATA_FOR_MIS	[boolean] Default: False	Applies the NoData value to band values that do not fall in any class. If False, the original value is kept.

Fortsetzung auf der nächsten Seite

Tab. 28.78 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the format of the output raster file.</p> <p>Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Reclassified raster</b>	OUTPUT	[raster]	Output raster layer with reclassified band values

## Pythoncode

**Algorithm ID:** native:reclassifybylayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Reclassify by table

Reclassifies a raster band by assigning new class values based on the ranges specified in a fixed table.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Raster layer to reclassify
<b>Band number</b>	RASTER_BAND	[raster band] Default: 1	Raster band for which you want to recalculate values.
<b>Reclassification table</b>	TABLE	[table]	A 3-columns table to fill with the values to set the boundaries of each class (Minimum and Maximum) and the new Value to assign to the band values that fall in the class. The value <code>-inf</code> can be used as minimum, <code>inf</code> as maximum, and <code>nan</code> can be used to set the output value to NoData.
<b>Reclassified raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output NoData value</b>	NO_DATA	[number] Default: -9999.0	Value to apply to NoData values.
<b>Range boundaries</b>	RANGE_BOUNDARI	[enumeration] Default: 0	Defines comparison rules for the classification. Options: <ul style="list-style-type: none"> <li>• 0 — <math>\min &lt; \text{value} \leq \max</math></li> <li>• 1 — <math>\min \leq \text{value} &lt; \max</math></li> <li>• 2 — <math>\min \leq \text{value} \leq \max</math></li> <li>• 3 — <math>\min &lt; \text{value} &lt; \max</math></li> </ul>
<b>Use NoData when no range matches value</b>	NODATA_FOR_MIS	[boolean] Default: False	Applies the NoData value to band values that do not fall in any class. If False, the original value is kept.

Fortsetzung auf der nächsten Seite

Tab. 28.80 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the format of the output raster file.</p> <p>Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Reclassified raster</b>	OUTPUT	[raster]	Output raster layer with reclassified band values

## Pythoncode

**Algorithm ID:** native:reclassifybytable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rescale raster

Rescales raster layer to a new value range, while preserving the shape (distribution) of the raster's histogram (pixel values). Input values are mapped using a linear interpolation from the source raster's minimum and maximum pixel values to the destination minimum and maximum pixel range.

By default the algorithm preserves the original NoData value, but there is an option to override it.

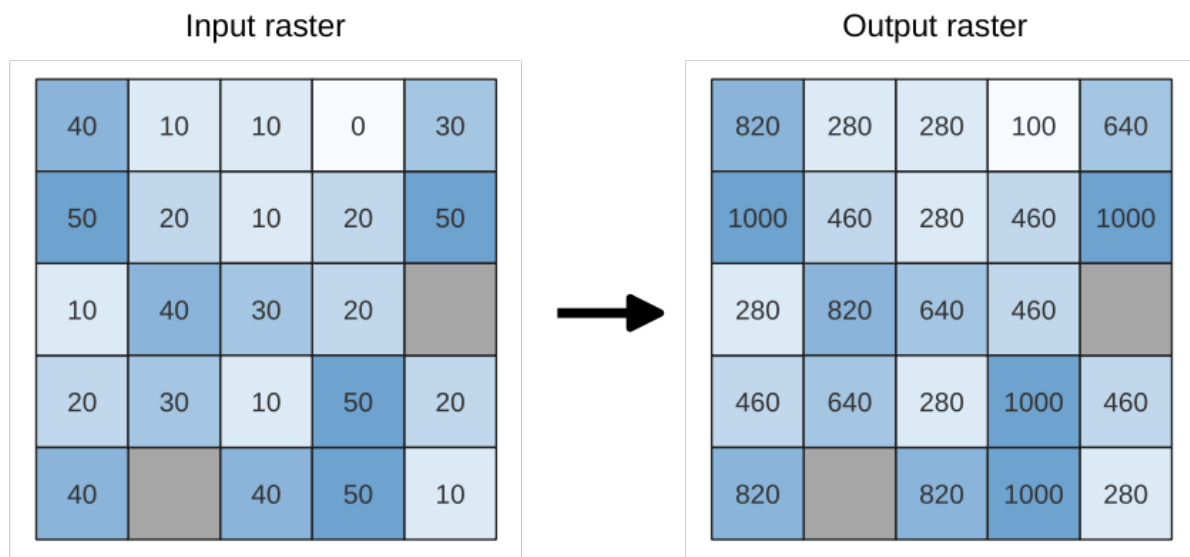


Abb. 28.26: Rescaling values of a raster layer from [0 - 50] to [100 - 1000]

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Raster</b>	INPUT	[raster]	Raster layer to use for rescaling
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose a band.
<b>New minimum value</b>	MINIMUM	[number] Default value: 0.0	Minimum pixel value to use in the rescaled layer
<b>New maximum value</b>	MAXIMUM	[number] Default value: 255.0	Maximum pixel value to use in the rescaled layer
<b>New NoData value</b> Optional	NODATA	[number] Default value: Not set	Value to assign to the NoData pixels. If unset, original NoData values are preserved.
<b>Rescaled</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rescaled</b>	OUTPUT	[raster]	Output raster layer with rescaled band values

## Pythoncode

**Algorithm ID:** native:rescaleraster

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Round raster

Rounds the cell values of a raster dataset according to the specified number of decimals.

Alternatively, a negative number of decimal places may be used to round values to powers of a base n. For example, with a Base value n of 10 and Decimal places of -1, the algorithm rounds cell values to multiples of 10, -2 rounds to multiples of 100, and so on. Arbitrary base values may be chosen, the algorithm applies the same multiplicative principle. Rounding cell values to multiples of a base n may be used to generalize raster layers.

The algorithm preserves the data type of the input raster. Therefore byte/integer rasters can only be rounded to multiples of a base n, otherwise a warning is raised and the raster gets copied as byte/integer raster.

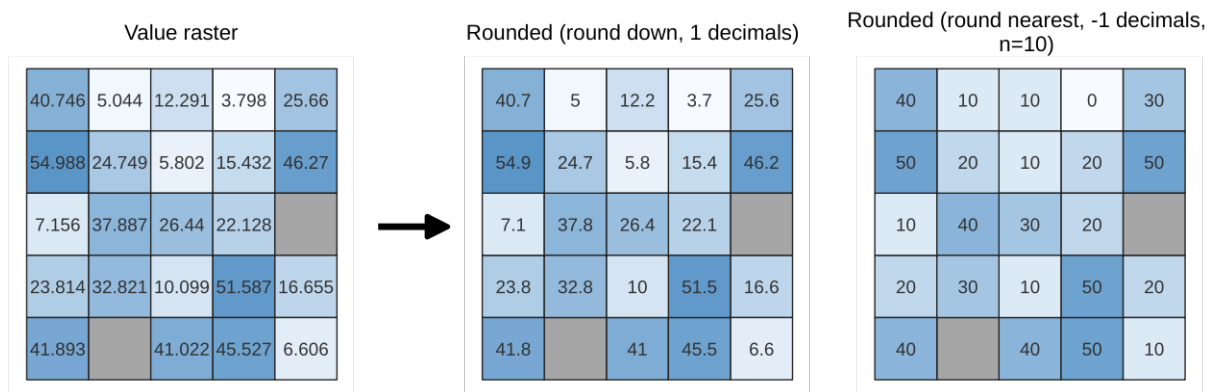


Abb. 28.27: Rounding values of a raster

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input raster</b>	INPUT	[raster]	The raster to process.
<b>Band number</b>	BAND	[number] Default: 1	The band of the raster
<b>Rounding direction</b>	ROUNDING_DIREC	[list] Default: 1	How to choose the target rounded value. Options are: <ul style="list-style-type: none"> <li>• 0 — Round up</li> <li>• 1 — Round to nearest</li> <li>• 2 — Round down</li> </ul>
<b>Number of decimals places</b>	DECIMAL_PLACES	[number] Default: 2	Number of decimals places to round to. Use negative values to round cell values to a multiple of a base n
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Base n for rounding to multiples of n</b>	BASE_N	[number] Default: 10	When the DECIMAL_PLACES parameter is negative, raster values are rounded to multiples of the base n value

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	The output raster layer with values rounded for the selected band.

## Pythoncode

**Algorithm ID:** native:roundrastervalues

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Sample raster values

Extracts raster values at the point locations. If the raster layer is multiband, each band is sampled.

The attribute table of the resulting layer will have as many new columns as the raster layer band count.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: point]	Point vector layer to use for sampling
<b>Raster Layer</b>	RASTERCOPY	[raster]	Raster layer to sample at the given point locations.
<b>Output column prefix</b>	COLUMN_PREFIX	[string] Default: ,SAMPLE_‘	Prefix for the names of the added columns.
<b>Sampled</b> Optional	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output layer containing the sampled values. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Sampled</b>	OUTPUT	[vector: point]	The output layer containing the sampled values.

### Pythoncode

**Algorithm ID:** native:rastersampling

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Zonal histogram

Appends fields representing counts of each unique value from a raster layer contained within polygon features.

The output layer attribute table will have as many fields as the unique values of the raster layer that intersects the polygon(s).

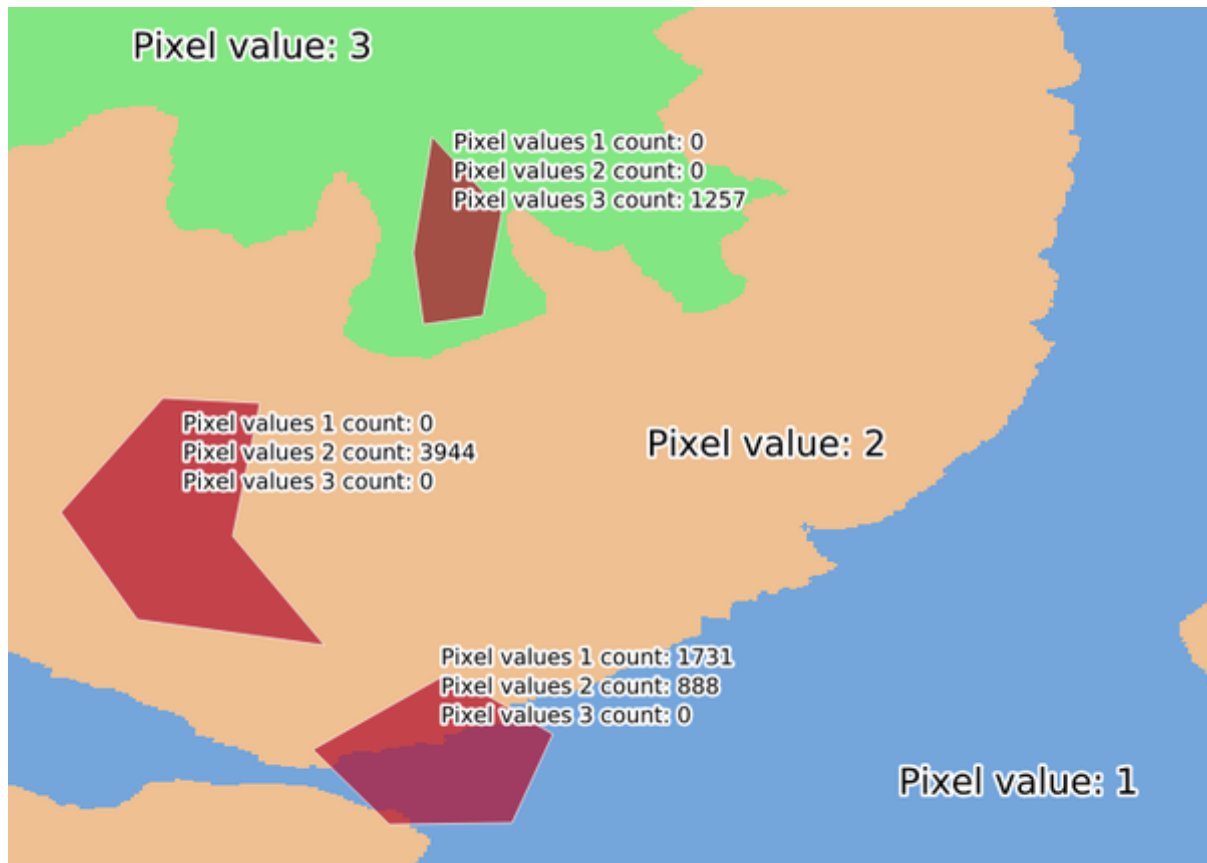


Abb. 28.28: Raster layer histogram example

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer.
<b>Band number</b>	RASTER_BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose a band.
<b>Vector layer containing zones</b>	INPUT_VECTOR	[vector: polygon]	Vector polygon layer that defines the zones.
<b>Output column prefix</b>	COLUMN_PREFIX Optional	[string] Default: ,HISTO_	Prefix for the output columns names.
<b>Output zones</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector polygon layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output zones</b>	OUTPUT	[vector: polygon]	The output vector polygon layer.

## Pythoncode

**Algorithm ID:** native:zonalhistogram

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Zonal statistics

Calculates statistics of a raster layer for each feature of an overlapping polygon vector layer.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Vector polygon layer that contains the zones.
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer.
<b>Raster band</b>	RASTER_BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose a band for the statistics.
<b>Output column prefix</b>	COLUMN_PREFIX	[string] Default: ,_'	Prefix for the output columns names.
<b>Statistics to calculate</b>	STATISTICS	[enumeration] [list] Default: [0,1,2]	List of statistical operator for the output. Options: <ul style="list-style-type: none"> <li>• 0 — Count</li> <li>• 1 — Sum</li> <li>• 2 — Mean</li> <li>• 3 — Median</li> <li>• 4 — St. dev.</li> <li>• 5 — Minimum</li> <li>• 6 — Maximum</li> <li>• 7 — Range</li> <li>• 8 — Minority</li> <li>• 9 — Majority</li> <li>• 10 — Variety</li> <li>• 11 — Variance</li> </ul>
<b>Zonal Statistics</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector polygon layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Zonal Statistics</b>	OUTPUT	[vector: polygon]	The zone vector layer with added statistics.

## Pythoncode

**Algorithm ID:** native:zonalstatisticsfb

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.16 Raster Creation

### Create constant raster layer

Generates raster layer for given extent and cell size filled with the specified value.

Additionally an output data type can be specified. The algorithm will abort if a value has been entered that cannot be represented by the selected output raster data type.

### Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRIS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Constant value</b>	NUMBER	[number] Default: 1	Constant pixel value for the output raster layer.
<b>Constant</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 5	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Byte</li> <li>• 1 — Integer16</li> <li>• 2 — Unsigned Integer16</li> <li>• 3 — Integer32</li> <li>• 4 — Unsigned Integer32</li> <li>• 5 — Float32</li> <li>• 6 — Float64</li> </ul>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Constant</b>	OUTPUT	[raster]	Raster covering the desired extent with the specified pixel size and value.

## Pythoncode

**Algorithm ID:** native:createconstantrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (binomial distribution)

Generates a raster layer for given extent and cell size filled with binomially distributed random values.

By default, the values will be chosen given an N of 10 and a probability of 0.5. This can be overridden by using the advanced parameter for N and probability. The raster data type is set to Integer types (Integer16 by default). The binomial distribution random values are defined as positive integer numbers. A floating point raster will represent a cast of integer values to floating point.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRSS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Integer16</li> <li>• 1 — Unsigned Integer16</li> <li>• 2 — Integer32</li> <li>• 3 — Unsigned Integer32</li> <li>• 4 — Float32</li> <li>• 5 — Float64</li> </ul>
<b>N</b>	N	[number] Default: 10	
<b>Probability</b>	PROBABILITY	[number] Default: 0.5	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with random values

## Pythoncode

**Algorithm ID:** `native:createrandombinomialrasterlayer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (exponential distribution)

Generates a raster layer for given extent and cell size filled with exponentially distributed random values.

By default, the values will be chosen given a lambda of 1.0. This can be overridden by using the advanced parameter for lambda. The raster data type is set to Float32 by default as the exponential distribution random values are floating point numbers.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRSS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Float32</li> <li>• 1 — Float64</li> </ul>
<b>Lambda</b>	LAMBDA	[number] Default: 1.0	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with random values

## Pythoncode

**Algorithm ID:** native:createrandomexponentialrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (gamma distribution)

Generates a raster layer for given extent and cell size filled with gamma distributed random values.

By default, the values will be chosen given an alpha and beta value of 1.0. This can be overridden by using the advanced parameter for alpha and beta. The raster data type is set to Float32 by default as the gamma distribution random values are floating point numbers.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRIS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Float32</li> <li>• 1 — Float64</li> </ul>
<b>Alpha</b>	ALPHA	[number] Default: 1.0	
<b>Beta</b>	BETA	[number] Default: 1.0	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** native:createrandomgammaraSTERlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (geometric distribution)

Generates a raster layer for given extent and cell size filled with geometrically distributed random values.

By default, the values will be chosen given a probability of 0.5. This can be overridden by using the advanced parameter for mean value. The raster data type is set to Integer types (Integer16 by default). The geometric distribution random values are defined as positive integer numbers. A floating point raster will represent a cast of integer values to floating point.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRSS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Integer16</li> <li>• 1 — Unsigned Integer16</li> <li>• 2 — Integer32</li> <li>• 3 — Unsigned Integer32</li> <li>• 4 — Float32</li> <li>• 5 — Float64</li> </ul>
<b>Probability</b>	PROBABILITY	[number] Default: 0.5	



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** native:createrandomgeometricrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (negative binomial distribution)

Generates a raster layer for given extent and cell size filled with negative binomially distributed random values.

By default, the values will be chosen given a distribution parameter *k* of 10.0 and a probability of 0.5. This can be overridden by using the advanced parameters for *k* and probability. The raster data type is set to Integer types (Integer16 by default). The negative binomial distribution random values are defined as positive integer numbers. A floating point raster will represent a cast of integer values to floating point.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Integer16</li> <li>• 1 — Unsigned Integer16</li> <li>• 2 — Integer32</li> <li>• 3 — Unsigned Integer32</li> <li>• 4 — Float32</li> <li>• 5 — Float64</li> </ul>
<b>Distribution parameter k</b>	K_PARAMETER	[number] Default: 10	
<b>Probability</b>	PROBABILITY	[number] Default: 0.5	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** `native:createrandomnegativebinomialrasterlayer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (normal distribution)

Generates a raster layer for given extent and cell size filled with normally distributed random values.

By default, the values will be chosen given a mean of 0.0 and a standard deviation of 1.0. This can be overridden by using the advanced parameters for mean and standard deviation value. The raster data type is set to Float32 by default as the normal distribution random values are floating point numbers.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Float32</li> <li>• 1 — Float64</li> </ul>
<b>Mean of normal distribution</b>	MEAN	[number] Default: 0.0	
<b>Standard deviation of normal distribution</b>	STDDEV	[number] Default: 1.0	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** native:createrandomnormalrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (poisson distribution)

Generates a raster layer for given extent and cell size filled with poisson distributed random values.

By default, the values will be chosen given a mean of 1.0. This can be overridden by using the advanced parameter for mean value. The raster data type is set to Integer types (Integer16 by default). The poisson distribution random values are positive integer numbers. A floating point raster will represent a cast of integer values to floating point.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRIS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 0	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Integer16</li> <li>• 1 — Unsigned Integer16</li> <li>• 2 — Integer32</li> <li>• 3 — Unsigned Integer32</li> <li>• 4 — Float32</li> <li>• 5 — Float64</li> </ul>
<b>Mean</b>	MEAN	[number] Default: 1.0	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** native:createrandompoissonrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create random raster layer (uniform distribution)

Generates a raster layer for given extent and cell size filled with random values.

By default, the values will range between the minimum and maximum value of the specified output raster type. This can be overridden by using the advanced parameters for lower and upper bound value. If the bounds have the same value or both are zero (default) the algorithm will create random values in the full value range of the chosen raster data type. Choosing bounds outside the acceptable range of the output raster type will abort the algorithm.

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Desired extent</b>	EXTENT	[extent]	Specify the spatial extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Target CRS</b>	TARGET_CRS	[crs] Default: Project CRS	CRS for the output raster layer
<b>Pixel size</b>	PIXEL_SIZE	[number] Default: 1.0	Pixel size (X=Y) in map units.
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster data type</b>	OUTPUT_TYPE Default: 5	[enumeration]	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Byte</li> <li>• 1 — Integer16</li> <li>• 2 — Unsigned Integer16</li> <li>• 3 — Integer32</li> <li>• 4 — Unsigned Integer32</li> <li>• 5 — Float32</li> <li>• 6 — Float64</li> </ul>
<b>Lower bound for random number range</b>	LOWER_BOUND	[number] Default: 0.0	
<b>Upper bound for random number range</b>	UPPER_BOUND	[number] Default: 0.0	



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	Raster covering the desired extent with the cell size filled with randomly distributed values

## Pythoncode

**Algorithm ID:** native:createrandomuniformrasterlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.17 Raster terrain analysis

### Aspect

Calculates the aspect of the Digital Terrain Model in input. The final aspect raster layer contains values from 0 to 360 that express the slope direction, starting from north (0°) and continuing clockwise.

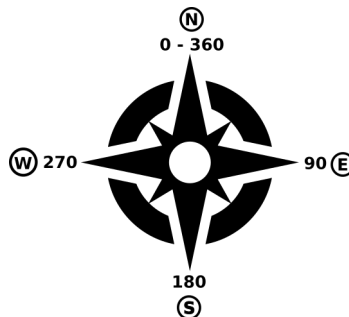


Abb. 28.29: Aspect values

The following picture shows the aspect layer reclassified with a color ramp:

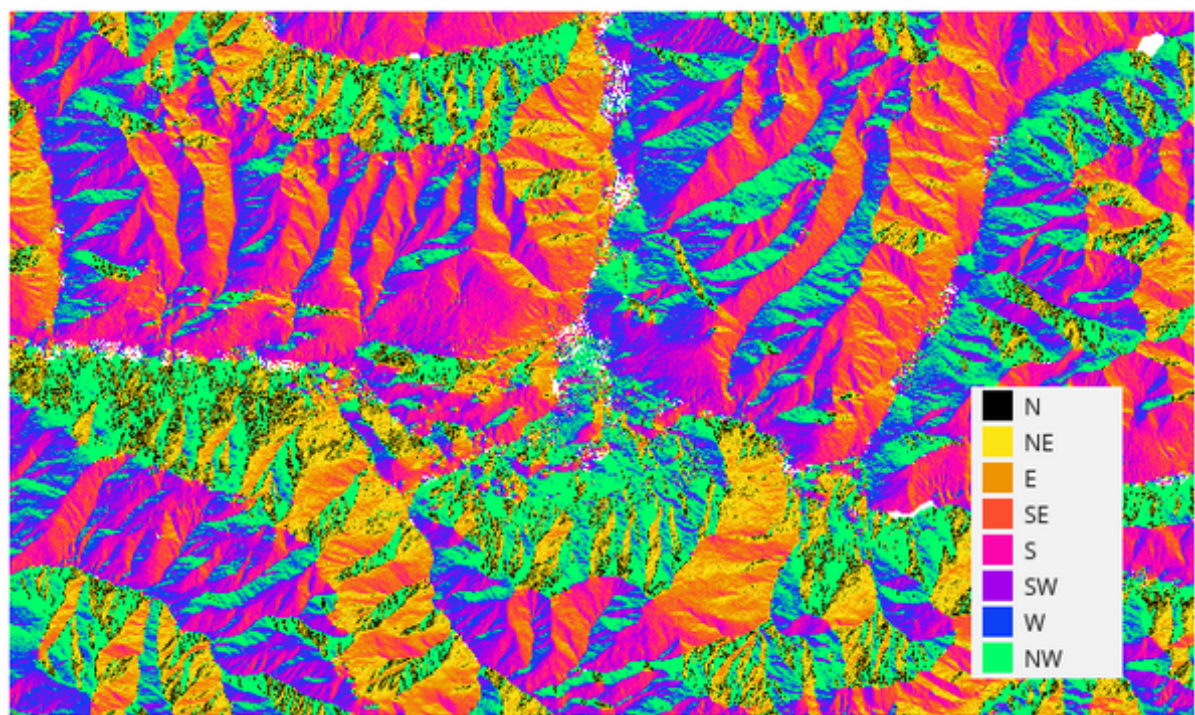


Abb. 28.30: Aspect layer reclassified

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Elevation layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Z factor</b>	Z_FACTOR	[number] Default: 1.0	Vertical exaggeration. This parameter is useful when the Z units differ from the X and Y units, for example feet and meters. You can use this parameter to adjust for this. The default is 1 (no exaggeration).
<b>Aspect</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output aspect raster layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aspect</b>	OUTPUT	[raster]	The output aspect raster layer

## Pythoncode

**Algorithm ID:** qgis:aspect

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## DTM filter (slope-based)

NEW in 3.34

Can be used to filter a digital elevation model in order to classify its cells into ground and object (non-ground) cells.

The tool uses concepts as described by Vosselman (2000) and is based on the assumption that a large height difference between two nearby cells is unlikely to be caused by a steep slope in the terrain. The probability that the higher cell might be non-ground increases when the distance between the two cells decreases. Therefore the filter defines a maximum height difference ( $dz_{max}$ ) between two cells as a function of the distance ( $d$ ) between the cells ( $dz_{max}(d) = d$ ). A cell is classified as terrain if there is no cell within the kernel radius to which the height difference is larger than the allowed maximum height difference at the distance between these two cells.

The approximate terrain slope ( $s$ ) parameter is used to modify the filter function to match the overall slope in the study area ( $dz_{max}(d) = d * s$ ). A 5 % confidence interval ( $ci = 1.65 * \sqrt{2 * stddev}$ ) may be used to modify the filter function even further by either relaxing ( $dz_{max}(d) = d * s + ci$ ) or amplifying ( $dz_{max}(d) = d * s - ci$ ) the filter criterium.

*References: Vosselman, G. (2000): Slope based filtering of laser altimetry data. IAPRS, Vol. XXXIII, Part B3, Amsterdam, The Netherlands, 935-942*

### Siehe auch:

This tool is a port of the SAGA [DTM Filter \(slope-based\)](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Band number</b>	BAND	[number] [list]	The band of the DEM to consider
<b>Kernel radius (pixels)</b>	RADIUS	[number] Default: 5	The radius of the filter kernel (in pixels). Must be large enough to reach ground cells next to non-ground objects.
<b>Terrain slope (% , pixel size/vertical units)</b>	TERRAIN_SLOPE	[number] Default: 30	The approximate terrain slope in %. The terrain slope must be adjusted to account for the ratio of height units vs raster pixel dimensions. Used to relax the filter criterium in steeper terrain.

Fortsetzung auf der nächsten Seite

Tab. 28.84 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Filter modification</b>	<code>FILTER_MODIFIC</code>	[list] Default: 0	Choose whether to apply the filter kernel without modification or to use a confidence interval to relax or amplify the height criterium. <ul style="list-style-type: none"> <li>• 0 - None</li> <li>• 1 - Relax filter</li> <li>• 2 - Amplify</li> </ul>
<b>Standard deviation</b>	<code>STANDARD_DEVIA</code>	[number] Default: 0.1	The standard deviation used to calculate a 5% confidence interval applied to the height threshold.
<b>Output layer (ground)</b> Optional	<code>OUTPUT_GROUND</code>	[raster] Default: [Save to temporary file]	Specify the filtered DEM containing only cells classified as ground. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>
<b>Output layer (non-ground objects)</b> Optional	<code>OUTPUT_NONGROU</code>	[raster] Default: [Skip output]	Specify the non-ground objects removed by the filter. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer (ground)</b>	<code>OUTPUT_GROUND</code>	[raster]	The filtered DEM containing only cells classified as ground.
<b>Output layer (non-ground objects)</b>	<code>OUTPUT_NONGROU</code>	[raster]	The non-ground objects removed by the filter.

## Pythoncode

**Algorithm ID:** `native:dtmslopebasedfilter`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Hillshade

Calculates the hillshade raster layer given an input Digital Terrain Model.

The shading of the layer is calculated according to the sun position: you have the options to change both the horizontal angle (azimuth) and the vertical angle (sun elevation) of the sun.

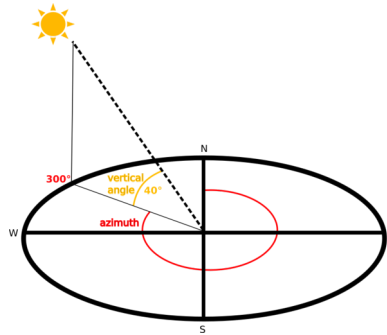


Abb. 28.31: Azimuth and vertical angle

The hillshade layer contains values from 0 (complete shadow) to 255 (complete sun). Hillshade is used usually to better understand the relief of the area.

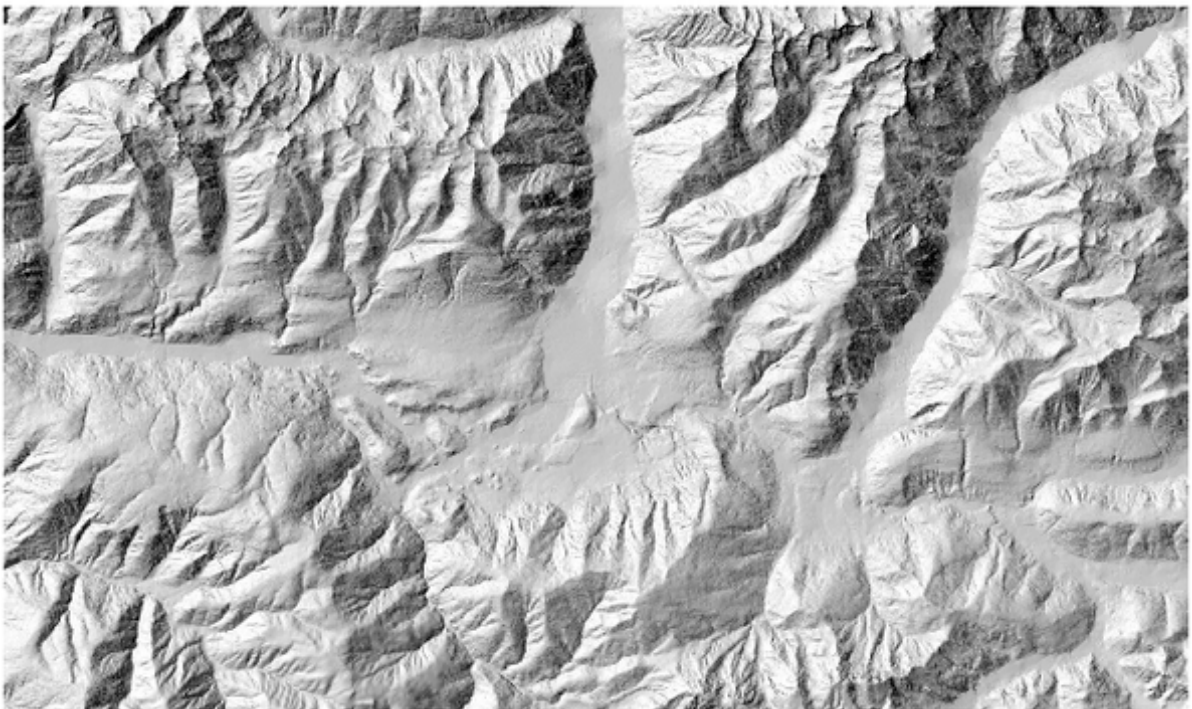


Abb. 28.32: Hillshade layer with azimuth 300 and vertical angle 45

Particularly interesting is to give the hillshade layer a transparency value and overlap it with the elevation raster:



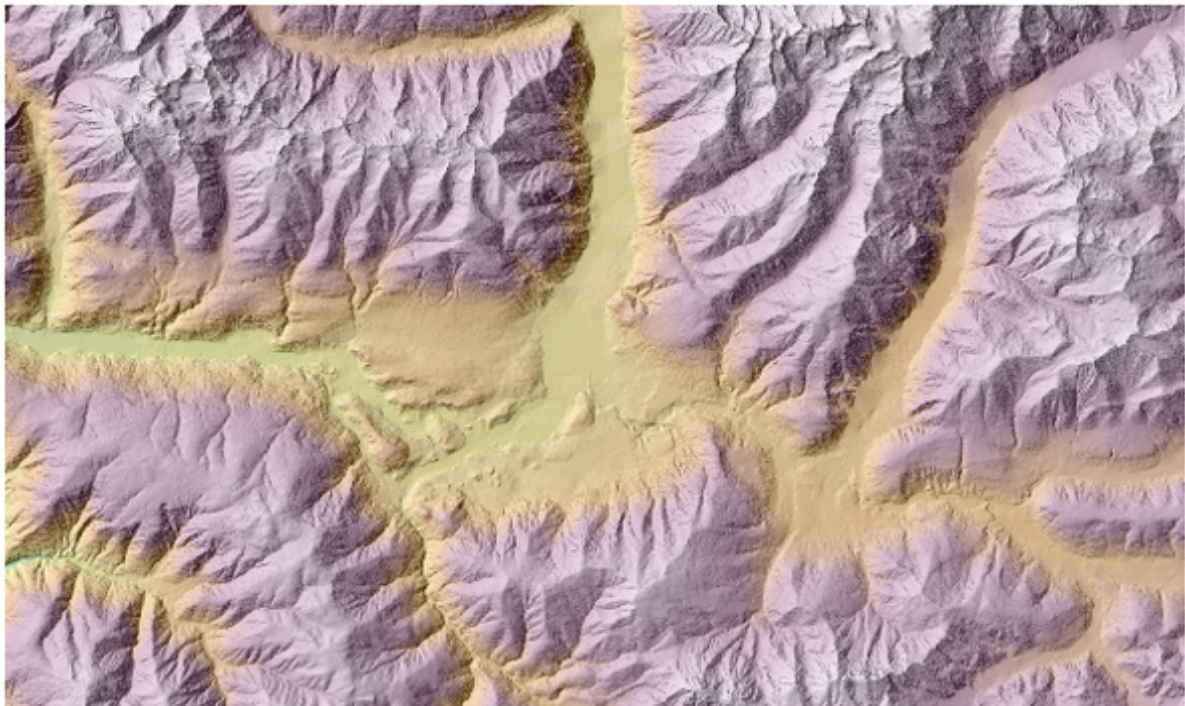


Abb. 28.33: Overlapping the hillshade with the elevation layer

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Elevation layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Z factor</b>	Z_FACTOR	[number] Default: 1.0	Vertical exaggeration. This parameter is useful when the Z units differ from the X and Y units, for example feet and meters. You can use this parameter to adjust for this. Increasing the value of this parameter will exaggerate the final result (making it look more „hilly“). The default is 1 (no exaggeration).
<b>Azimuth (horizontal angle)</b>	AZIMUTH	[number] Default: 300.0	Set the horizontal angle (in degrees) of the sun (clockwise direction). Range: 0 to 360. 0 is north.
<b>Vertical angle</b>	V_ANGLE	[number] Default: 40.0	Set the vertical angle (in degrees) of the sun, that is the height of the sun. Values can go from 0 (minimum elevation) to 90 (maximum elevation).
<b>Hillshade</b>	OUTPUT	[raster] Default: Save to temporary file	Specify the output hillshade raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hillshade</b>	OUTPUT	[raster]	The output hillshade raster layer

## Pythoncode

**Algorithm ID:** qgis:hillshade

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Hypsometric curves

Calculates hypsometric curves for an input Digital Elevation Model. Curves are produced as CSV files in an output folder specified by the user.

A hypsometric curve is a cumulative histogram of elevation values in a geographical area.

You can use hypsometric curves to detect differences in the landscape due to the geomorphology of the territory.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>DEM to analyze</b>	INPUT_DEM	[raster]	Digital Terrain Model raster layer to use for calculating altitudes
<b>Boundary layer</b>	BOUNDARY_LAYER	[vector: polygon]	Polygon vector layer with boundaries of areas used to calculate hypsometric curves
<b>Step</b>	STEP	[number] Default: 100.0	Vertical distance between curves
<b>Use % of area instead of absolute value</b>	USE_PERCENTAGE	[boolean] Default: False	Write area percentage to “Area” field of the CSV file instead of the absolute area
<b>Hypsometric curves</b>	OUTPUT_DIRECTO	[folder]	Specify the output folder for the hypsometric curves. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hypsometric curves</b>	OUTPUT_DIRECTO	[folder]	Directory containing the files with the hypsometric curves. For each feature from the input vector layer, a CSV file with area and altitude values will be created. The file names start with <code>histogram_</code> , followed by layer name and feature ID.

	A	B
1	Area	Elevation
2	177475194.383	307
3	233206029.24	407
4	295553735.793	507
5	394718815.615	607
6	501801102.615	707
7	624399019.792	807
8	828877274.39	907
9	1042693465.68	1007
10	1277373021.81	1107
11	1556443975.41	1207
12	1888617494.27	1307
13	2248520437.31	1407
14	2627916813.17	1507
15	3010880212.04	1607
16	3411087555.34	1707

## Pythoncode

**Algorithm ID:** `qgis:hypsometriccurves`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Relief

Creates a shaded relief layer from digital elevation data. You can specify the relief color manually, or you can let the algorithm choose automatically all the relief classes.



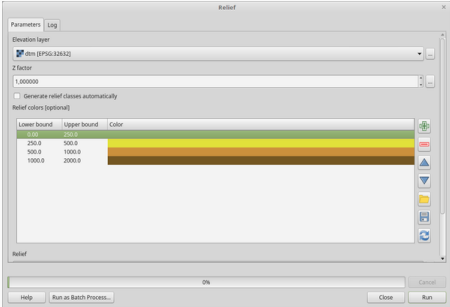
Abb. 28.34: Relief layer

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Elevation layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Z factor</b>	Z_FACTOR	[number] Default: 1.0	Vertical exaggeration. This parameter is useful when the Z units differ from the X and Y units, for example feet and meters. You can use this parameter to adjust for this. Increasing the value of this parameter will exaggerate the final result (making it look more „hilly“). The default is 1 (no exaggeration).
<b>Generate relief classes automatically</b>	AUTO_COLORS	[boolean] Default: False	If you check this option the algorithm will create all the relief color classes automatically

Fortsetzung auf der nächsten Seite

Tab. 28.87 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Relief colors</b> Optional	COLORS	[table widget]	Use the table widget if you want to choose the relief colors manually. You can add as many color classes as you want: for each class you can choose the lower and upper bound and finally by clicking on the color row you can choose the color thanks to the color widget. 
<b>Relief</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output relief raster layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>
<b>Frequency distribution</b> Optional	FREQUENCY_DIST	[table] Default: [Skip output]	Specify the CSV table for the output frequency distribution. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Relief</b>	OUTPUT	[raster]	The output relief raster layer
<b>Frequency distribution</b>	OUTPUT	[table]	The output frequency distribution

## Pythoncode

**Algorithm ID:** qgis:relief

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Ruggedness index

Calculates the quantitative measurement of terrain heterogeneity described by Riley et al. (1999). It is calculated for every location, by summarizing the change in elevation within the 3x3 pixel grid.

Each pixel contains the difference in elevation from a center cell and the 8 cells surrounding it.

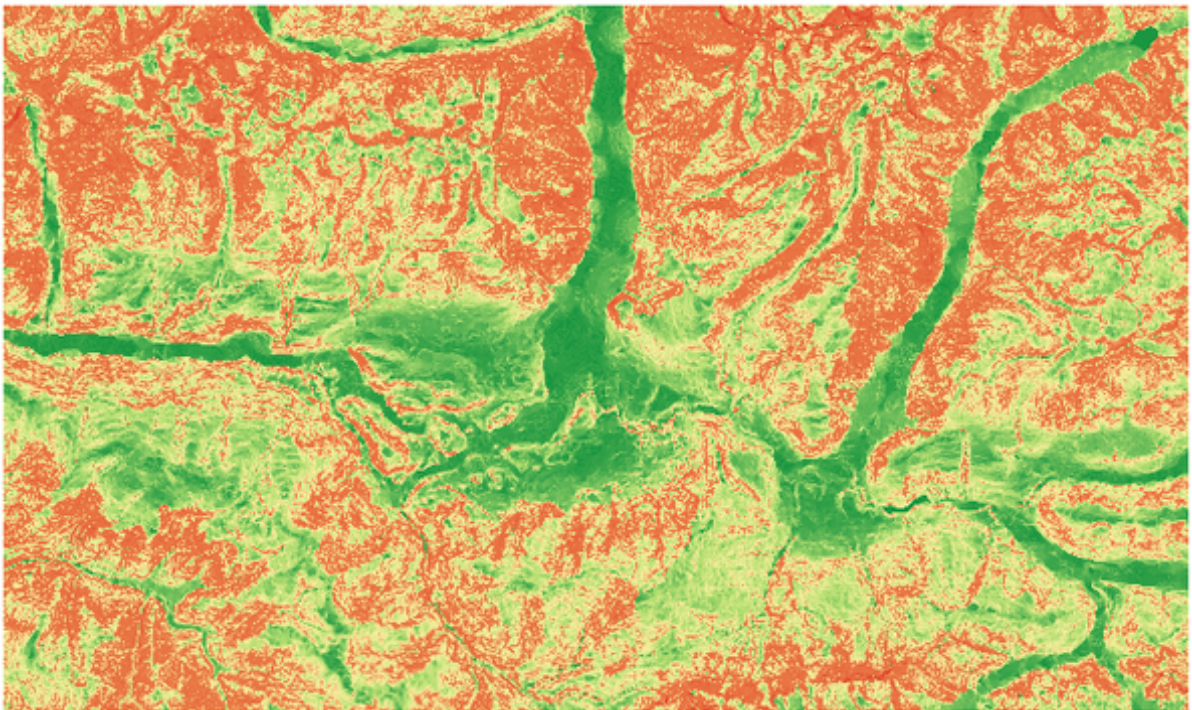


Abb. 28.36: Ruggedness layer from low (red) to high values (green)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Elevation layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Z factor</b>	Z_FACTOR	[number] Default: 1.0	Vertical exaggeration. This parameter is useful when the Z units differ from the X and Y units, for example feet and meters. You can use this parameter to adjust for this. Increasing the value of this parameter will exaggerate the final result (making it look more rugged). The default is 1 (no exaggeration).
<b>Ruggedness</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output ruggedness raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ruggedness</b>	OUTPUT	[raster]	The output ruggedness raster layer

## Pythoncode

**Algorithm ID:** qgis:ruggednessindex

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



Slope

Calculates the slope from an input raster layer. The slope is the angle of inclination of the terrain and is expressed in **degrees**.

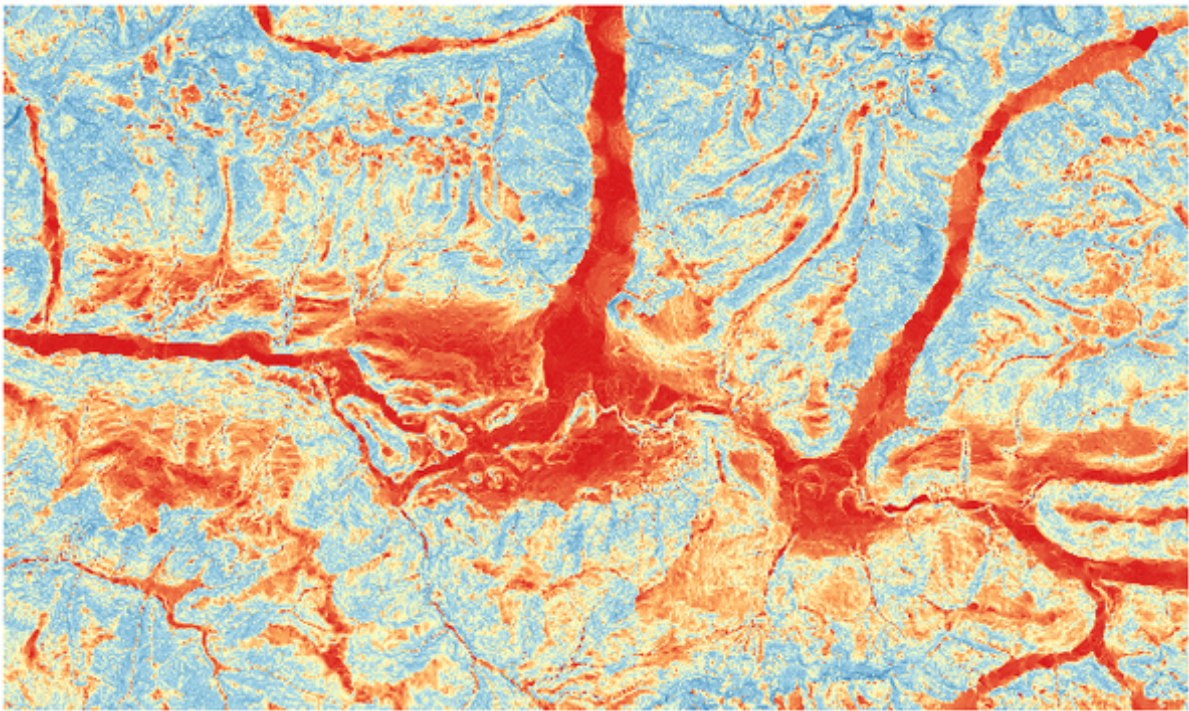


Abb. 28.37: Flat areas in red, steep areas in blue

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Elevation layer</b>	INPUT	[raster]	Digital Terrain Model raster layer
<b>Z factor</b>	Z_FACTOR	[number] Default: 1.0	Vertical exaggeration. This parameter is useful when the Z units differ from the X and Y units, for example feet and meters. You can use this parameter to adjust for this. Increasing the value of this parameter will exaggerate the final result (making it steeper). The default is 1 (no exaggeration).
<b>Slope</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output slope raster layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Slope</b>	OUTPUT	[raster]	The output slope raster layer

## Pythoncode

**Algorithm ID:** qgis:slope

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.18 Raster tools

### Align raster

NEW in 3.34

Aligns raster by resampling it to the same cell size and reprojecting to the same CRS as a reference raster.

**Warnung:** This algorithm is ONLY available in the *Model Designer* context.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer to align
<b>Resampling method</b>	RESAMPLING_MET	[enumeration] Default: 0	Method to use for input layer resampling. Available options are: <ul style="list-style-type: none"> <li>0 — Nearest Neighbour</li> <li>1 — Bilinear (2x2 kernel)</li> <li>2 — Cubic (4x4 kernel)</li> <li>3 — Cubic B-Spline (4x4 kernel)</li> <li>4 — Lanczos (6x6 kernel)</li> <li>5 — Average</li> <li>6 — Mode</li> <li>7 — Maximum</li> <li>8 — Minimum</li> <li>9 — Median</li> <li>10 — First quartile (Q1)</li> <li>11 — Third quartile (Q3)</li> </ul>
<b>Rescale values according to the cell size</b>	RESCALE	[boolean] Default: No	

Fortsetzung auf der nächsten Seite

Tab. 28.88 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Reference layer</b>	REFERENCE_LAYER	[raster]	A raster layer that will be used to fetch extent, cell size and CRS that will be applied to input layers.
<b>Override reference CRS</b> Optional	CRS	[crs]	CRS to be used instead of the reference layer's
<b>Override reference cell size X</b> Optional	CELL_SIZE_X	[number]	Cell size in X direction to be used instead of the reference layer's
<b>Override reference cell size Y</b> Optional	CELL_SIZE_Y	[number]	Cell size in Y direction to be used instead of the reference layer's
<b>Override reference grid offset X</b> Optional	GRID_OFFSET_X	[number]	Offset in X direction to apply to cells grid
<b>Override reference grid offset Y</b> Optional	GRID_OFFSET_Y	[number]	Offset in Y direction to apply to cells grid
<b>Clip to extent</b> Optional	EXTENT	[extent]	Specify the extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Aligned raster</b>	OUTPUT	[raster]	output raster with pixels resampled

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aligned raster</b>	OUTPUT	[raster]	output raster with pixels resampled

## Pythoncode

**Algorithm ID:** native:alignraster

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Align rasters

NEW in 3.34

Aligns rasters by resampling them to the same cell size and reprojecting to the same CRS as a reference raster.

**Warnung:** This algorithm is NOT available in the *Model Designer* context.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	<p>List of input raster layers with resampling options associated (filled as a <code>QgsProcessingParameterAlignRasterLayers</code> item):</p> <p><b>Input layer [string] (inputFile)</b> Full path of the input layer to align</p> <p><b>Output file [string] (outputFile)</b> Full path of the corresponding aligned output layer</p> <p><b>Resampling method [enumeration] (resampleMethod)</b> Method to use for input layer resampling. Available options are:</p> <ul style="list-style-type: none"> <li>• 0 — Nearest Neighbour</li> <li>• 1 — Bilinear (2x2 kernel)</li> <li>• 2 — Cubic (4x4 kernel)</li> <li>• 3 — Cubic B-Spline (4x4 kernel)</li> <li>• 4 — Lanczos (6x6 kernel)</li> <li>• 5 — Average</li> <li>• 6 — Mode</li> <li>• 7 — Maximum</li> <li>• 8 — Minimum</li> <li>• 9 — Median</li> <li>• 10 — First quartile (Q1)</li> <li>• 11 — Third quartile (Q3)</li> </ul> <p><b>Rescale values according to the cell size</b> [boolean] (rescale)</p>

Fortsetzung auf der nächsten Seite



Tab. 28.89 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Reference layer</b>	REFERENCE_LAYER	[raster]	A raster layer that will be used to fetch extent, cell size and CRS that will be applied to input layers.
<b>Override reference CRS</b> Optional	CRS	[crs]	CRS to be used instead of the reference layer's
<b>Override reference cell size X</b> Optional	CELL_SIZE_X	[number]	Cell size in X direction to be used instead of the reference layer's
<b>Override reference cell size Y</b> Optional	CELL_SIZE_Y	[number]	Cell size in Y direction to be used instead of the reference layer's
<b>Override reference grid offset X</b> Optional	GRID_OFFSET_X	[number]	Offset in X direction to apply to cells grid
<b>Override reference grid offset Y</b> Optional	GRID_OFFSET_Y	[number]	Offset in Y direction to apply to cells grid
<b>Clip to extent</b> Optional	EXTENT	[extent]	Specify the extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aligned rasters</b>	OUTPUT_LAYERS	[raster] [list]	output rasters with pixels resampled

## Pythoncode

**Algorithm ID:** native:alignrasters

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert map to raster

Creates a raster image of map canvas content.

A *map theme* can be selected to render a predetermined set of layers with a defined style for each layer.

Alternatively, a single layer can be selected if no map theme is set.

If neither map theme nor layer is set, the current map content will be rendered. The minimum extent entered will internally be extended to be a multiple of the tile size.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Minimum extent to render</b> (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Specify the extent of the output raster layer. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>Use map canvas extent</li> <li>Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Tile size</b>	TILE_SIZE	[number] Default: 1024	Size of the tile of the output raster layer. Minimum value: 64.
<b>Map units per pixel</b>	MAP_UNITS_PER_PIXEL	[number] Default: 100.0	Pixel size (in map units). Minimum value: 0.0
<b>Make background transparent</b>	MAKE_BACKGROUND_TRANSPARENT	[boolean] Default: False	Allows exporting the map with a transparent background. Outputs an RGBA (instead of RGB) image if set to True.
<b>Map theme to render</b> Optional	MAP_THEME	[enumeration]	Use an existing <i>map theme</i> for the rendering.

Fortsetzung auf der nächsten Seite

Tab. 28.90 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Single layer to render</b> Optional	LAYER	[enumeration]	Choose a single layer for the rendering
<b>Output layer</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[raster]	Output raster layer

## Pythoncode

**Algorithm ID:** native:rasterize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fill NoData cells

Resets the NoData values in the input raster to a chosen value, resulting in raster dataset with no NoData pixels.

The algorithm respects the input raster data type, e.g. a floating point fill value will be truncated when applied to an integer raster.

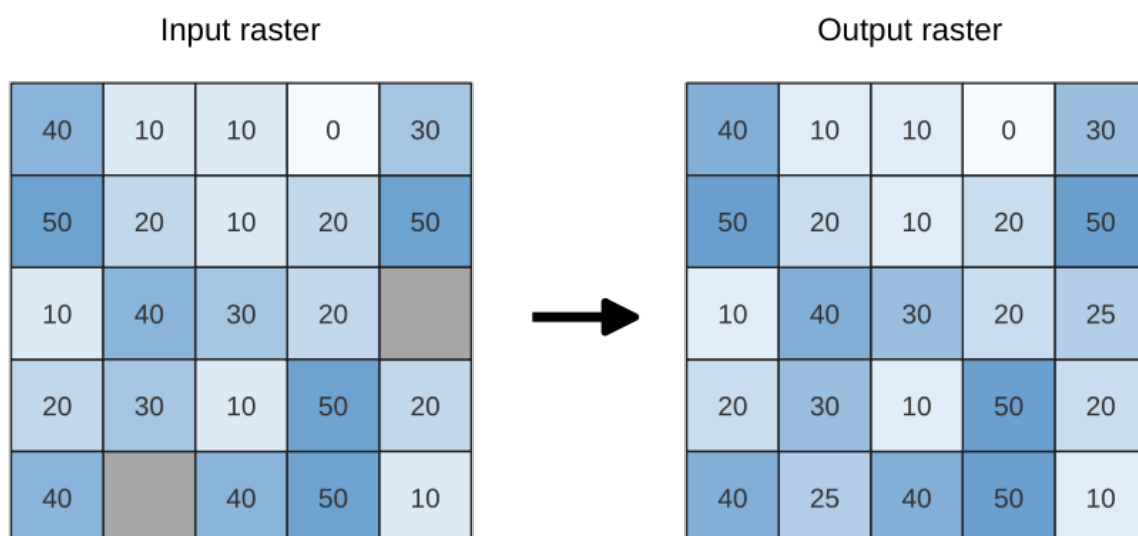


Abb. 28.38: Filling NoData values (in grey) of a raster

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input raster</b>	INPUT	[raster]	The raster to process.
<b>Band number</b>	BAND	[number] Default: 1	The band of the raster
<b>Fill value</b>	FILL_VALUE	[number] Default: 1.0	Set the value to use for the NoData pixels
<b>Output raster</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster</b>	OUTPUT	[raster]	The output raster layer with filled data cells.

## Pythoncode

**Algorithm ID:** native:fillnodata

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Generate XYZ tiles (Directory)

Generates raster “XYZ” tiles using the current QGIS project as individual images to a directory structure.

Optionally, a Leaflet HTML output file using the generated tiles as a map layer could be created.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Specify the extent of the tiles. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Minimum zoom</b>	ZOOM_MIN	[number] Default: 12	Minimum 0, maximum 25.
<b>Maximum zoom</b>	ZOOM_MAX	[number] Default: 12	Minimum 0, maximum 25.
<b>DPI</b>	DPI	[number] Default: 96	Minimum 48, maximum 600.
<b>Background color</b> Optional	BACKGROUND_COL	[color] Default: QColor(0, 0, 0, 0)	Choose the background color for the tiles
<b>Enable antialiasing</b>	ANTIALIAS	[boolean] Default: True	Determines if antialiasing should be enabled
<b>Tile format</b>	TILE_FORMAT	[enumeration] Default: 0	One of: <ul style="list-style-type: none"> <li>• 0 — PNG</li> <li>• 1 — JPG</li> </ul>
<b>Quality (JPG only)</b> Optional	QUALITY	[number] Default: 75	Minimum 1, maximum 100.
<b>Metatile size</b> Optional	METATILESIZE	[number] Default: 4	Specify a custom metatile size when generating XYZ tiles. Larger values may speed up the rendering of tiles and provide better labelling (fewer gaps without labels) at the expense of using more memory. Minimum 1, maximum 20.
<b>Tile width</b> Optional	TILE_WIDTH	[number] Default: 256	Minimum 1, maximum 4096.
<b>Tile height</b> Optional	TILE_HEIGHT	[number] Default: 256	Minimum 1, maximum 4096.
<b>Use inverted tile Y axis (TMS conventions)</b> Optional	TMS_CONVENTION	[boolean] Default: False	

Fortsetzung auf der nächsten Seite

Tab. 28.91 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b> Optional	OUTPUT_DIRECTO	[folder] Default: [Save to temporary folder]	Specification of the output directory (for the tiles). One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>
<b>Output html (Leaflet)</b> Optional	OUTPUT_HTML	[html] Default: [Save to temporary file]	Specification of the output HTML file. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

NEW in 3.30

Bezeichnung	Name	Typ	Beschreibung
<b>Leaflet HTML output title</b> Optional	HTML_TITLE	[string] Default: Not set	HTML <title>-tag used for the Leaflet HTML output file.
<b>Leaflet HTML output attribution</b> Optional	HTML_ATTRIBUTI	[string] Default: Not set	Custom map attribution used for the Leaflet HTML output file. HTML links are possible.
<b>Include OpenStreetMap basemap in Leaflet HTML output</b> Optional	HTML_OSM	[boolean] Default: False	An OpenStreetMap basemap layer (source: <a href="https://tile.openstreetmap.org">https://tile.openstreetmap.org</a> ) is included in the Leaflet HTML output file. Proper map attribution is added automatically.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b>	OUTPUT_DIRECTO	[folder]	Output directory (for the tiles)
<b>Output html (Leaflet)</b>	OUTPUT_HTML	[html]	The output HTML (Leaflet) file

### Pythoncode

**Algorithm ID:** native:tilexyzdirectory

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Generate XYZ tiles (MBTiles)

Generates raster “XYZ” tiles using the current QGIS project as a single file in the “MBTiles” format.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> <b>xmin,</b> <b>xmax,</b> <b>ymin,</b> <b>ymax)</b>	(xmin, xmax, ymin, ymax) EXTENT	[extent]	Specify the extent of the tiles. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Minimum zoom</b>	ZOOM_MIN	[number] Default: 12	Minimum 0, maximum 25.
<b>Maximum zoom</b>	ZOOM_MAX	[number] Default: 12	Minimum 0, maximum 25.
<b>DPI</b>	DPI	[number] Default: 96	Minimum 48, maximum 600.
<b>Background color</b> Optional	BACKGROUND_COLOR	[color] Default: QColor(0, 0, 0, 0)	Choose the background color for the tiles
<b>Enable antialiasing</b>	ANTIALIAS	[boolean] Default: True	Determines if antialiasing should be enabled
<b>Tile format</b>	TILE_FORMAT	[enumeration] Default: 0	One of: <ul style="list-style-type: none"> <li>• 0 — PNG</li> <li>• 1 — JPG</li> </ul>
<b>Quality (JPG only)</b> Optional	QUALITY	[number] Default: 75	Minimum 1, maximum 100.
<b>Metatile size</b> Optional	METATILESIZE	[number] Default: 4	Specify a custom metatile size when generating XYZ tiles. Larger values may speed up the rendering of tiles and provide better labelling (fewer gaps without labels) at the expense of using more memory. Minimum 1, maximum 20.

Fortsetzung auf der nächsten Seite

Tab. 28.93 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output file (for MBTiles)</b>	OUTPUT_FILE	[file] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output file (for MBTiles)</b>	OUTPUT_FILE	[file]	The output file.

## Pythoncode

**Algorithm ID:** native:tilestxyzmbtiles

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.19 Vektoranalyse

### Grundstatistik für Felder

Erzeugt Grundstatistiken für ein Feld in der Attributtabelle eines Vektorlayers.

Numerische, Datums-, Zeit- und Zeichenkettenfelder sind unterstützt

The statistics returned will depend on the field type.

Statistics are generated as an HTML file and are available in the *Processing ► Results viewer*.

**Default menu:** *Vector ► Analysis Tools*



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input vector</b>	INPUT_LAYER	[vector: any]	Vector layer to calculate the statistics on
<b>Field to calculate statistics on</b>	FIELD_NAME	[tablefield: any]	Any supported table field to calculate the statistics
<b>Statistics</b> Optional	OUTPUT_HTML_FI	[html] Default: [Save to temporary file]	Specification of the file for the calculated statistics. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Statistics</b>	OUTPUT_HTML_FI	[html]	HTML file with the calculated statistics
<b>Count</b>	COUNT	[number]	
<b>Number of unique values</b>	UNIQUE	[number]	
<b>Number of empty (null) values</b>	EMPTY	[number]	
<b>Number of non-empty values</b>	FILLED	[number]	
<b>Minimum value</b>	MIN	[same as input]	
<b>Maximum value</b>	MAX	[same as input]	
<b>Minimum length</b>	MIN_LENGTH	[number]	
<b>Maximum length</b>	MAX_LENGTH	[number]	
<b>Mean length</b>	MEAN_LENGTH	[number]	
<b>Coefficient of Variation</b>	CV	[number]	
<b>Sum</b>	SUM	[number]	
<b>Mean value</b>	MEAN	[number]	
<b>Standard deviation</b>	STD_DEV	[number]	
<b>Range</b>	RANGE	[number]	
<b>Median</b>	MEDIAN	[number]	
<b>Minority (rarest occurring value)</b>	MINORITY	[same as input]	
<b>Majority (most frequently occurring value)</b>	MAJORITY	[same as input]	
<b>First quartile</b>	FIRSTQUARTILE	[number]	
<b>Third quartile</b>	THIRDQUARTILE	[number]	
<b>Interquartile Range (IQR)</b>	IQR	[number]	

## Pythoncode

**Algorithm ID:** qgis:basicstatisticsforfields

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Climb along line

Calculates the total climb and descent along line geometries. The input layer must have Z values present. If Z values are not available, the *Drape (set Z value from raster)* algorithm may be used to add Z values from a DEM layer.

The output layer is a copy of the input layer with additional fields that contain the total climb (`climb`), total descent (`descent`), the minimum elevation (`minelev`) and the maximum elevation (`maxelev`) for each line geometry. If the input layer contains fields with the same names as these added fields, they will be renamed (field names will be altered to „name\_2“, „name\_3“, etc, finding the first non-duplicate name).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Line layer</b>	INPUT	[vector: line]	Line layer to calculate the climb for. Must have Z values
<b>Climb layer</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specification of the output (line) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Climb layer</b>	OUTPUT	[vector: line]	Line layer containing new attributes with the results from climb calculations.
<b>Total climb</b>	TOTALCLIMB	[number]	The sum of the climb for all the line geometries in the input layer
<b>Total descent</b>	TOTALDESCENT	[number]	The sum of the descent for all the line geometries in the input layer
<b>Minimum elevation</b>	MINELEVATION	[number]	The minimum elevation for the geometries in the layer
<b>Maximum elevation</b>	MAXELEVATION	[number]	The maximum elevation for the geometries in the layer

## Pythoncode

**Algorithm ID:** qgis:climbalongline

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Count points in polygon

Takes a point and a polygon layer and counts the number of points from the point layer in each of the polygons of the polygon layer.

A new polygon layer is generated, with the exact same content as the input polygon layer, but containing an additional field with the points count corresponding to each polygon.

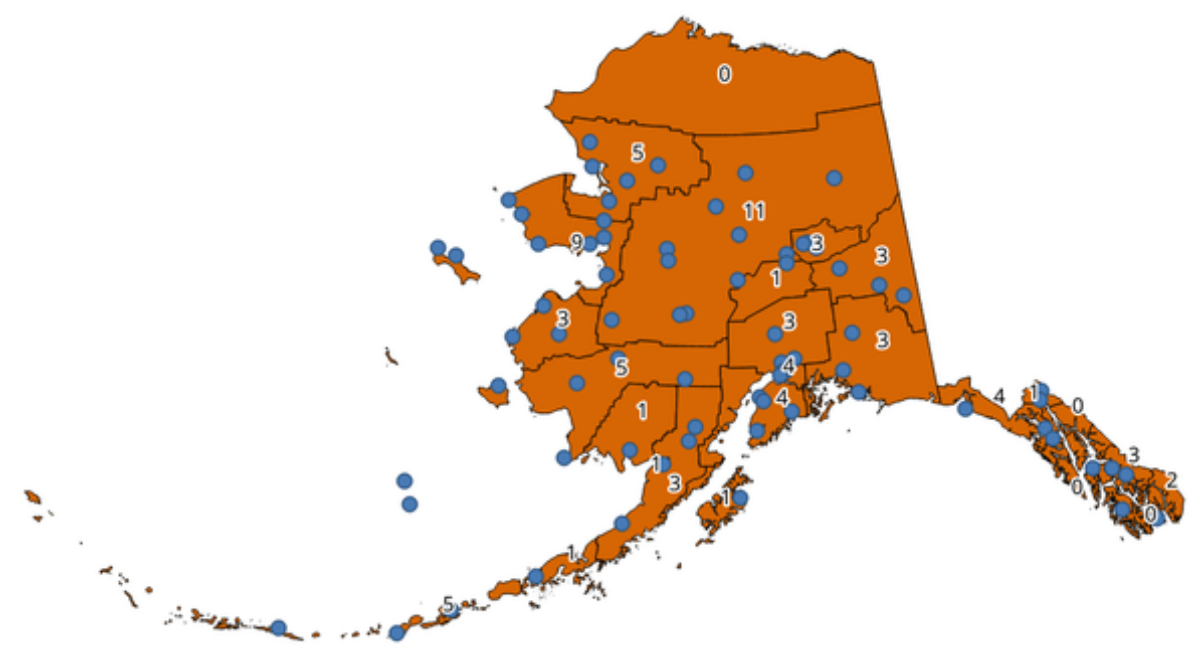


Abb. 28.39: The labels in the polygons show the point count

An optional weight field can be used to assign weights to each point. Alternatively, a unique class field can be specified. If both options are used, the weight field will take precedence and the unique class field will be ignored.

 Allows *features in-place modification* of polygon features

Default menu: *Vector ► Analysis Tools*

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Polygons</b>	POLYGONS	[vector: polygon]	Polygon layer whose features are associated with the count of points they contain
<b>Points</b>	POINTS	[vector: point]	Point layer with features to count
<b>Weight field</b> Optional	WEIGHT	[tablefield: any]	A field from the point layer. The count generated will be the sum of the weight field of the points contained by the polygon. If the weight field is not numeric, the count will be 0.

Fortsetzung auf der nächsten Seite

Tab. 28.95 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Class field</b> Optional	CLASSFIELD	[tablefield: any]	Points are classified based on the selected attribute and if several points with the same attribute value are within the polygon, only one of them is counted. The final count of the points in a polygon is, therefore, the count of different classes that are found in it.
<b>Count field name</b>	FIELD	[string] Default: ‚NUMPOINTS‘	The name of the field to store the count of points
<b>Count</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specification of the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Count</b>	OUTPUT	[vector: polygon]	Resulting layer with the attribute table containing the new column with the points count

## Pythoncode

**Algorithm ID:** native:countpointsinpolygon

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## DBSCAN clustering

Clusters point features based on a 2D implementation of Density-based spatial clustering of applications with noise (DBSCAN) algorithm.

The algorithm requires two parameters, a minimum cluster size, and the maximum distance allowed between clustered points.

**Siehe auch:**

*ST-DBSCAN clustering, K-means clustering*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Layer to analyze
<b>Minimum cluster size</b>	MIN_SIZE	[number] Default: 5	Minimum number of features to generate a cluster
<b>Maximum distance between clustered points</b>	EPS	[number] Default: 1.0	Distance beyond which two features can not belong to the same cluster (eps)
<b>Clusters</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the vector layer for the result of the clustering. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Treat border points as noise (DBSCAN*)</b> Optional	DBSCAN*	[boolean] Default: False	If checked, points on the border of a cluster are themselves treated as unclustered points, and only points in the interior of a cluster are tagged as clustered.
<b>Cluster field name</b>	FIELD_NAME	[string] Default: ,CLUSTER_ID'	Name of the field where the associated cluster number shall be stored
<b>Cluster size field name</b>	SIZE_FIELD_NAME	[string] Default: ,CLUSTER_SIZE'	Name of the field with the count of features in the same cluster

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clusters</b>	OUTPUT	[vector: point]	Vector layer containing the original features with a field setting the cluster they belong to
<b>Number of clusters</b>	NUM_CLUSTERS	[number]	The number of clusters discovered

## Pythoncode

**Algorithm ID:** native:dbscanclustering

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Distance matrix

Calculates for point features distances to their nearest features in the same layer or in another layer.

**Default menu:** *Vector ► Analysis Tools*

**Siehe auch:**

*Join attributes by nearest*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input point layer</b>	INPUT	[vector: point]	Point layer for which the distance matrix is calculated ( <b>from</b> points)
<b>Input unique ID field</b>	INPUT_FIELD	[tablefield: any]	Field to use to uniquely identify features of the input layer. Used in the output attribute table.
<b>Target point layer</b>	TARGET	[vector: point]	Point layer containing the nearest point(s) to search ( <b>to</b> points)
<b>Target unique ID field</b>	TARGET_FIELD	[tablefield: any]	Field to use to uniquely identify features of the target layer. Used in the output attribute table.
<b>Output matrix type</b>	MATRIX_TYPE	[enumeration] Default: 0	Different types of calculation are available: <ul style="list-style-type: none"> <li>• 0 — Linear (<math>N * k \times 3</math>) distance matrix: for each input point, reports the distance to each of the <math>k</math> nearest target points. The output matrix consists of up to <math>k</math> rows per input point, and each row has three columns: <i>InputID</i>, <i>TargetID</i> and <i>Distance</i>.</li> <li>• 1 — Standard (<math>N \times T</math>) distance matrix</li> <li>• 2 — Summary distance matrix (mean, std. dev., min, max): for each input point, reports statistics on the distances to its target points.</li> </ul>
<b>Use only the nearest (k) target points</b>	NEAREST_POINTS	[number] Default: 0	You can choose to calculate the distance to all the points in the target layer (0) or limit to a number ( $k$ ) of closest features.

Fortsetzung auf der nächsten Seite

Tab. 28.98 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Distance matrix</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specification of the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Distance matrix</b>	OUTPUT	[vector: point]	Point (or MultiPoint for the „Linear (N * k x 3)“ case) vector layer containing the distance calculation for each input feature. Its features and attribute table depend on the selected output matrix type.

## Pythoncode

**Algorithm ID:** qgis:distancematrix

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Distance to nearest hub (line to hub)

Creates lines that join each feature of an input vector to the nearest feature in a destination layer. Distances are calculated based on the *center* of each feature.

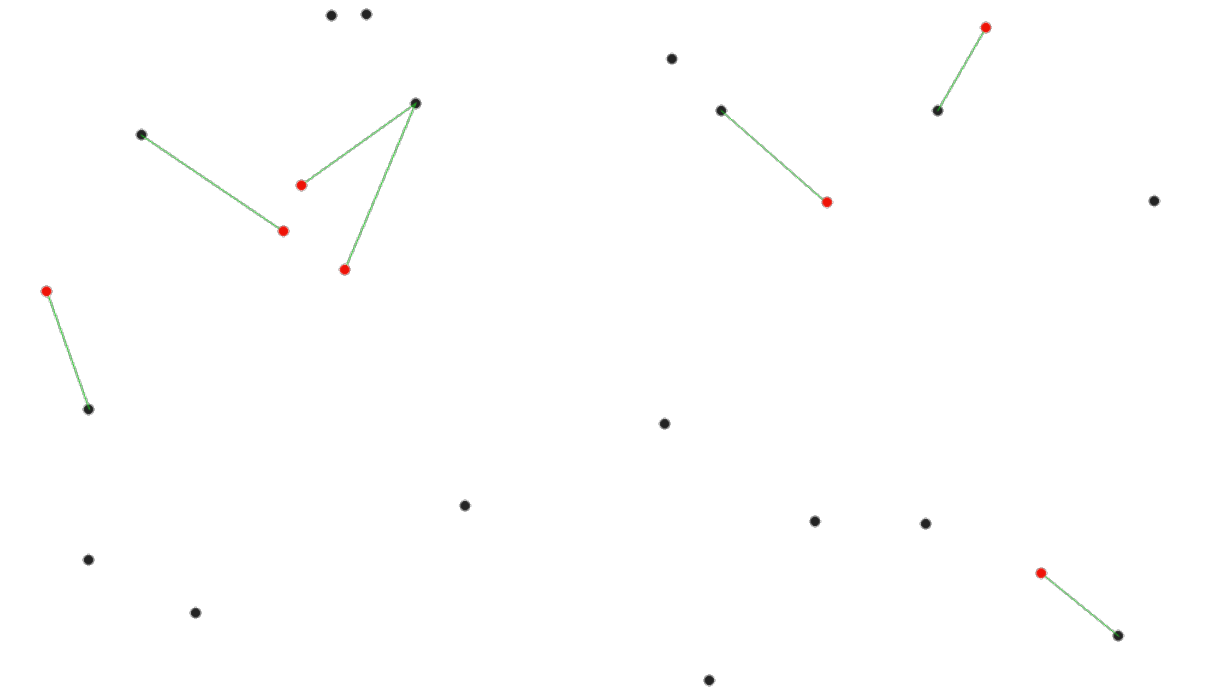


Abb. 28.40: Display the nearest hub for the red input features

**Siehe auch:**

*Distance to nearest hub (points), Join attributes by nearest*

**Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Source layer</b>	<b>points</b>	INPUT	[vector: any]
			Vector layer for which the nearest feature is searched
<b>Destination layer</b>	<b>hubs</b>	HUBS	[vector: any]
			Vector layer containing the features to search for
<b>Hub layer attribute</b>	<b>name</b>	FIELD	[tablefield: any]
			Field to use to uniquely identify features of the destination layer. Used in the output attribute table
<b>Measurement unit</b>	UNIT	[enumeration] Default: 0	Units in which to report the distance to the closest feature: <ul style="list-style-type: none"><li>• 0 — Meters</li><li>• 1 — Feet</li><li>• 2 — Miles</li><li>• 3 — Kilometers</li><li>• 4 — Layer units</li></ul>
<b>Hub distance</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer connecting the matching points. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hub distance</b>	OUTPUT	[vector: line]	Line vector layer with the attributes of the input features, the identifier of their closest feature and the calculated distance.

## Pythoncode

**Algorithm ID:** qgis:distancetonearesthublinetohub

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Distance to nearest hub (points)

Creates a point layer representing the *center* of the input features with the addition of two fields containing the identifier of the nearest feature (based on its center point) and the distance between the points.

### Siehe auch:

*Distance to nearest hub (line to hub), Join attributes by nearest*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Source points layer</b>	INPUT	[vector: any]	Vector layer for which the nearest feature is searched
<b>Destination hubs layer</b>	HUBS	[vector: any]	Vector layer containing the features to search for
<b>Hub layer name attribute</b>	FIELD	[tablefield: any]	Field to use to uniquely identify features of the destination layer. Used in the output attribute table
<b>Measurement unit</b>	UNIT	[enumeration] Default: 0	Units in which to report the distance to the closest feature: <ul style="list-style-type: none"> <li>• 0 — Meters</li> <li>• 1 — Feet</li> <li>• 2 — Miles</li> <li>• 3 — Kilometers</li> <li>• 4 — Layer units</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.99 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Hub distance</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output point vector layer with the nearest hub. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hub distance</b>	OUTPUT	[vector: point]	Point vector layer representing the center of the source features with their attributes, the identifier of their closest feature and the calculated distance.

## Pythoncode

**Algorithm ID:** qgis:distancetonearesthubpoints

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Join by lines (hub lines)

Creates hub and spoke diagrams by connecting lines from points on the Spoke layer to matching points in the Hub layer.

Determination of which hub goes with each point is based on a match between the Hub ID field on the hub points and the Spoke ID field on the spoke points.

If input layers are not point layers, a point on the surface of the geometries will be taken as the connecting location.

Optionally, geodesic lines can be created, which represent the shortest path on the surface of an ellipsoid. When geodesic mode is used, it is possible to split the created lines at the antimeridian ( $\pm 180$  degrees longitude), which can improve rendering of the lines. Additionally, the distance between vertices can be specified. A smaller distance results in a denser, more accurate line.

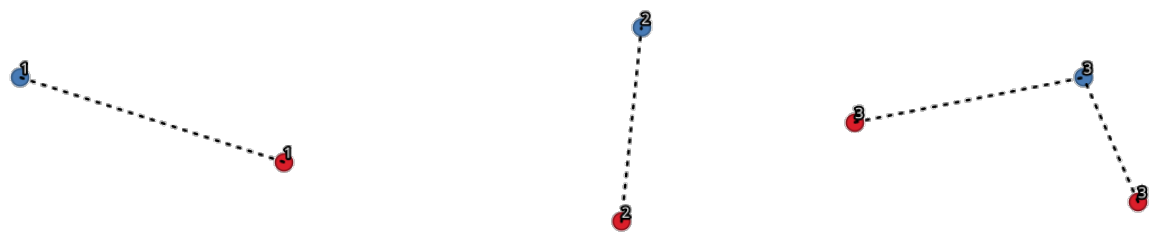


Abb. 28.41: Join points based on a common field / attribute

Parameter

Basic parameters

Bezeichnung	Name	Typ	Beschreibung
Hub layer	HUBS	[vector: any]	Input layer
Hub ID field	HUB_FIELD	[tablefield: any]	Field of the hub layer with ID to join
Hub layer fields to copy (leave empty to copy all fields)	HUB_FIELDS	[tablefield: any] [list]	The field(s) of the hub layer to be copied. If no field(s) are chosen all fields are taken.
Optional			
Spoke layer	SPOKES	[vector: any]	Additional spoke point layer
Spoke ID field	SPOKE_FIELD	[tablefield: any]	Field of the spoke layer with ID to join
Spoke layer fields to copy (leave empty to copy all fields)	SPOKE_FIELDS	[tablefield: any] [list]	Field(s) of the spoke layer to be copied. If no fields are chosen all fields are taken.
Optional			
Create geodesic lines	GEODESIC	[boolean] Default: False	Create geodesic lines (the shortest path on the surface of an ellipsoid)
Hub lines	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output hub line vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
Distance between vertices (geodesic lines only)	GEODESIC_DISTA	[number] Default: 1000.0 (kilometers)	Distance between consecutive vertices (in kilometers). A smaller distance results in a denser, more accurate line

Fortsetzung auf der nächsten Seite

Tab. 28.101 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Split lines at antimeridian (<math>\pm 180</math> degrees longitude)</b>	ANTIMERIDIAN_S	[boolean] Default: False	Split lines at $\pm 180$ degrees longitude (to improve rendering of the lines)

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hub lines</b>	OUTPUT	[vector: line]	The resulting line layer connecting matching points in input layers

## Pythoncode

**Algorithm ID:** native:hublines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## K-means clustering

Calculates the 2D distance based k-means cluster number for each input feature.

K-means clustering aims to partition the features into k clusters in which each feature belongs to the cluster with the nearest mean. The mean point is represented by the barycenter of the clustered features.

If input geometries are lines or polygons, the clustering is based on the centroid of the feature.

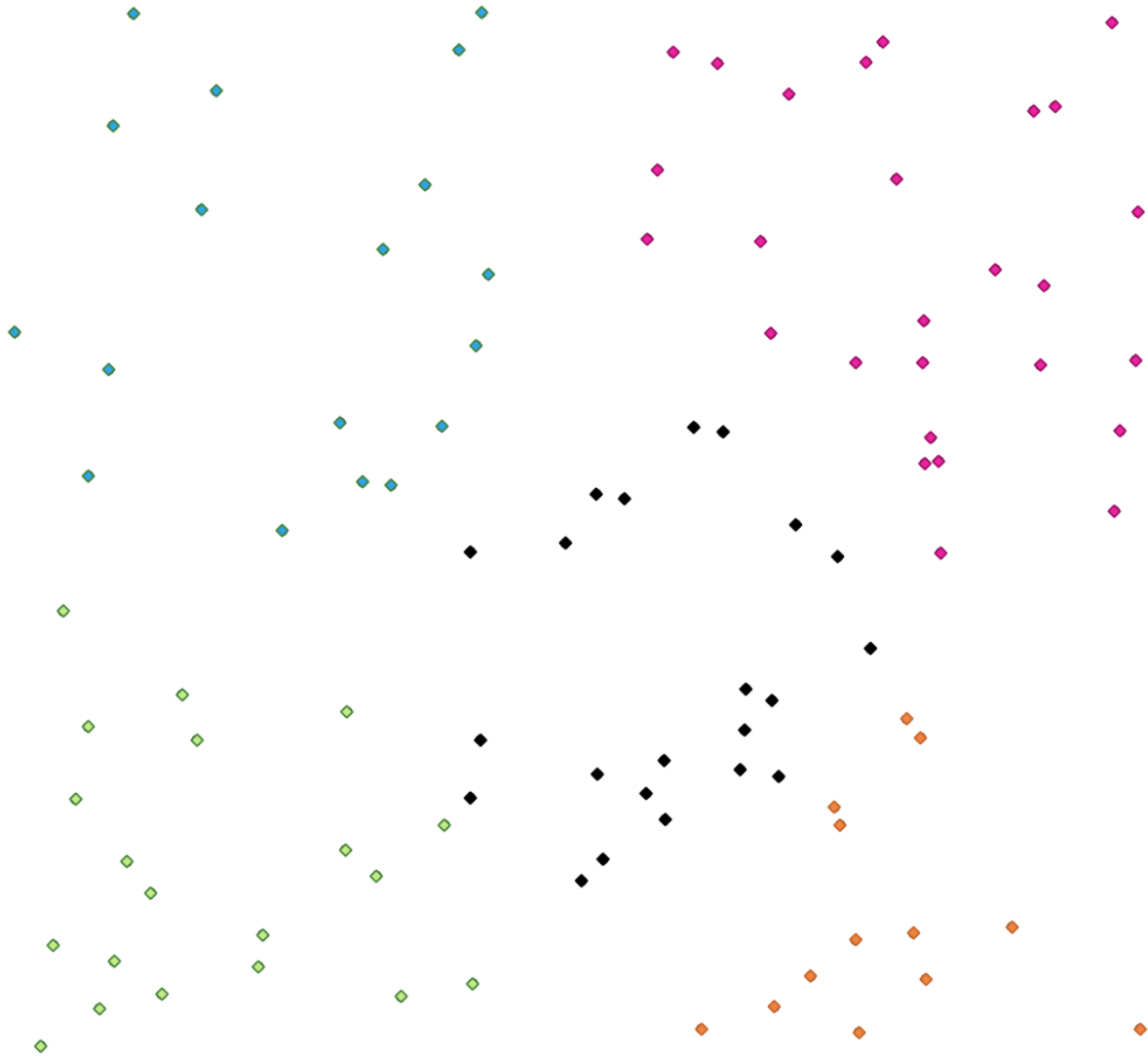


Abb. 28.42: A five class point clusters

**Siehe auch:**

*DBSCAN clustering, ST-DBSCAN clustering*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to analyze
<b>Number of clusters</b>	CLUSTERS	[number] Default: 5	Number of clusters to create with the features
<b>Clusters</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specify the output vector layer for generated the clusters. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Cluster field name</b>	FIELD_NAME	[string] Default: ,CLUSTER_ID‘	Name of the field where the associated cluster number shall be stored
<b>Cluster size field name</b>	SIZE_FIELD_NAME	[string] Default: ,CLUSTER_SIZE‘	Name of the field with the count of features in the same cluster

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clusters</b>	OUTPUT	[vector: any]	Vector layer containing the original features with fields specifying the cluster they belong to and their number in it

## Pythoncode

**Algorithm ID:** native:kmeansclustering

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## List unique values

Lists unique values of an attribute table field and counts their number.

**Default menu:** *Vector ► Analysis Tools*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to analyze
<b>Target field(s)</b>	FIELDS	[tablefield: any]	Field to analyze
<b>Unique values</b> Optional	OUTPUT	[table] Default: [Create temporary layer]	Specify the summary table layer with unique values. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>HTML report</b> Optional	OUTPUT_HTML_FI	[html] Default: [Save to temporary file]	HTML report of unique values in the <i>Processing ► Results viewer</i> . One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Unique values</b>	OUTPUT	[table]	Summary table layer with unique values
<b>HTML report</b>	OUTPUT_HTML_FI	[html]	HTML report of unique values. Can be opened from the <i>Processing ► Results viewer</i>
<b>Total unique values</b>	TOTAL_VALUES	[number]	The number of unique values in the input field
<b>Unique values concatenated</b>	UNIQUE_VALUES	[string]	A string with the comma separated list of unique values found in the input field

### Pythoncode

**Algorithm ID:** `qgis:listuniquevalues`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Mean coordinate(s)

Computes a point layer with the center of mass of geometries in an input layer.

An attribute can be specified as containing weights to be applied to each feature when computing the center of mass.

If an attribute is selected in the parameter, features will be grouped according to values in this field. Instead of a single point with the center of mass of the whole layer, the output layer will contain a center of mass for the features in each category.

**Default menu:** *Vector ► Analysis Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Weight field</b> Optional	WEIGHT	[tablefield: numeric]	Field to use if you want to perform a weighted mean
<b>Unique ID field</b>	UID	[tablefield: numeric]	Unique field on which the calculation of the mean will be made
<b>Mean coordinates</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the (point vector) layer for the result. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Mean coordinates</b>	OUTPUT	[vector: point]	Resulting point(s) layer

## Pythoncode

**Algorithm ID:** native:meancoordinates

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

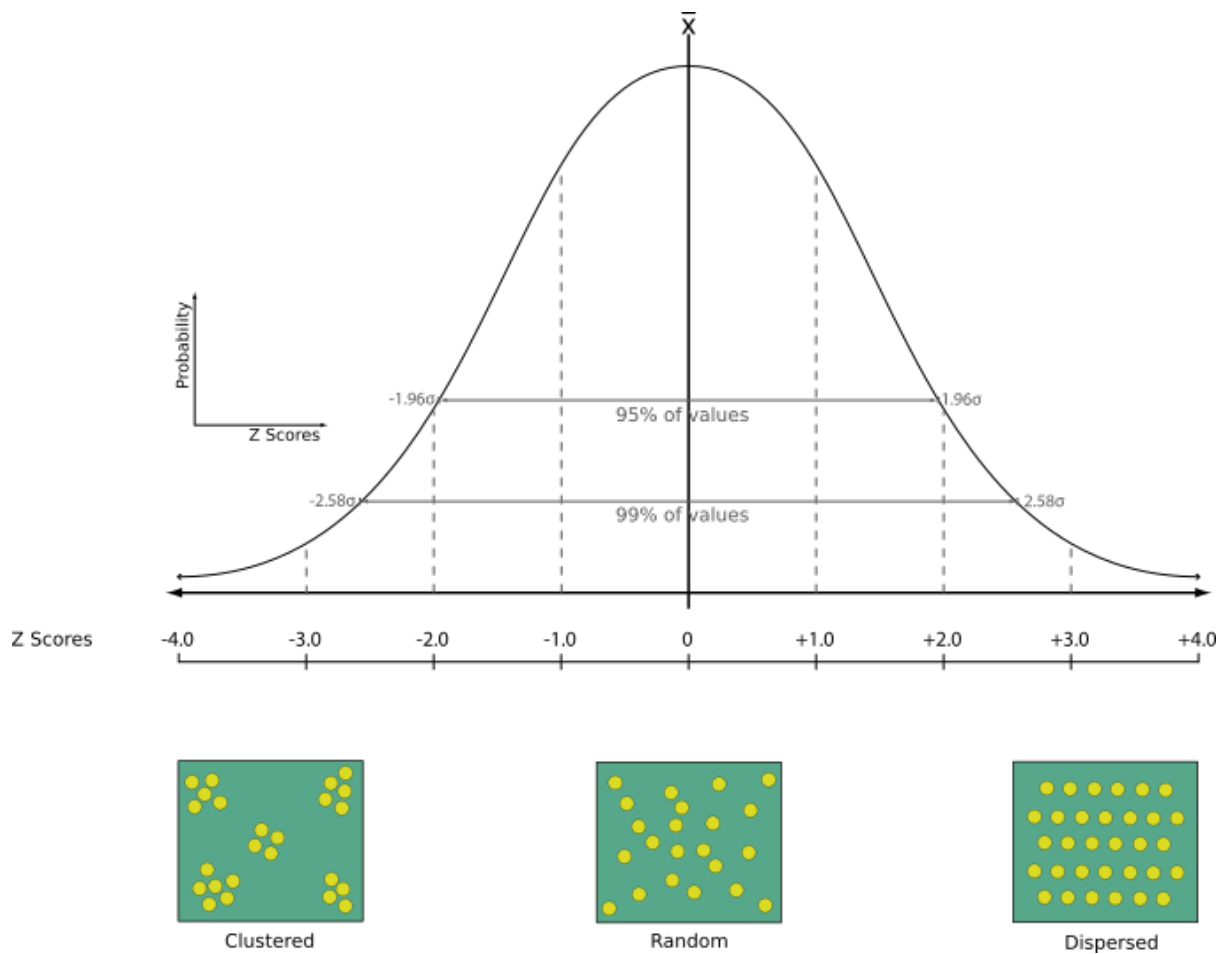


## Nearest neighbour analysis

Performs nearest neighbor analysis for a point layer. The output tells you how your data are distributed (clustered, randomly or distributed).

Output is generated as an HTML file with the computed statistical values:

- Observed mean distance
- Expected mean distance
- Nearest neighbour index
- Number of points
- Z-Score: Comparing the Z-Score with the normal distribution tells you how your data are distributed. A low Z-Score means that the data are unlikely to be the result of a spatially random process, while a high Z-Score means that your data are likely to be a result of a spatially random process.



**Default menu:** *Vector ► Analysis Tools*

**Siehe auch:**

*Join attributes by nearest*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Point vector layer to calculate the statistics on
<b>Nearest neighbour</b> Optional	OUTPUT_HTML_FI	[html] Default: [Save to temporary file]	Specification of the HTML file for the computed statistics. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Nearest neighbour</b>	OUTPUT_HTML_FI	[html]	HTML file with the computed statistics
<b>Observed mean distance</b>	OBSERVED_MD	[number]	Observed mean distance
<b>Expected mean distance</b>	EXPECTED_MD	[number]	Expected mean distance
<b>Nearest neighbour index</b>	NN_INDEX	[number]	Nearest neighbour index
<b>Number of points</b>	POINT_COUNT	[number]	Number of points
<b>Z-Score</b>	Z_SCORE	[number]	Z-Score

## Pythoncode

**Algorithm ID:** native:nearestneighbouranalysis

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Overlap analysis

Calculates the area and percentage cover by which features from an input layer are overlapped by features from a selection of overlay layers.

New attributes are added to the output layer reporting the total area of overlap and percentage of the input feature overlapped by each of the selected overlay layers.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer.
<b>Overlap layers</b>	LAYERS	[vector: any] [list]	The overlay layers.
<b>Overlap</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Grid size</b> Optional	GRID_SIZE	[number] Default: Not set	If provided, the input geometries are snapped to a grid of the given size, and the result vertices are computed on that same grid. Requires GEOS 3.9.0 or higher.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Overlap</b>	OUTPUT	[same as input]	The output layer with additional fields reporting the overlap (in map units and percentage) of the input feature overlapped by each of the selected layers.

## Pythoncode

**Algorithm ID:** native:calculatevectoroverlaps

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Shortest line between features

Creates a line layer as the shortest line between the source and the destination layer. By default only the first nearest feature of the destination layer is taken into account. The n-nearest neighboring features number can be specified. If a maximum distance is specified, then only features which are closer than this distance will be considered.

The output features will contain all the source layer attributes, all the attributes from the n-nearest feature and the additional field of the distance.

---

**Wichtig:** This algorithm uses purely Cartesian calculations for distance, and does not consider geodetic or ellipsoid properties when determining feature proximity. The measurement and output coordinate system is based on the coordinate system of the source layer.

---

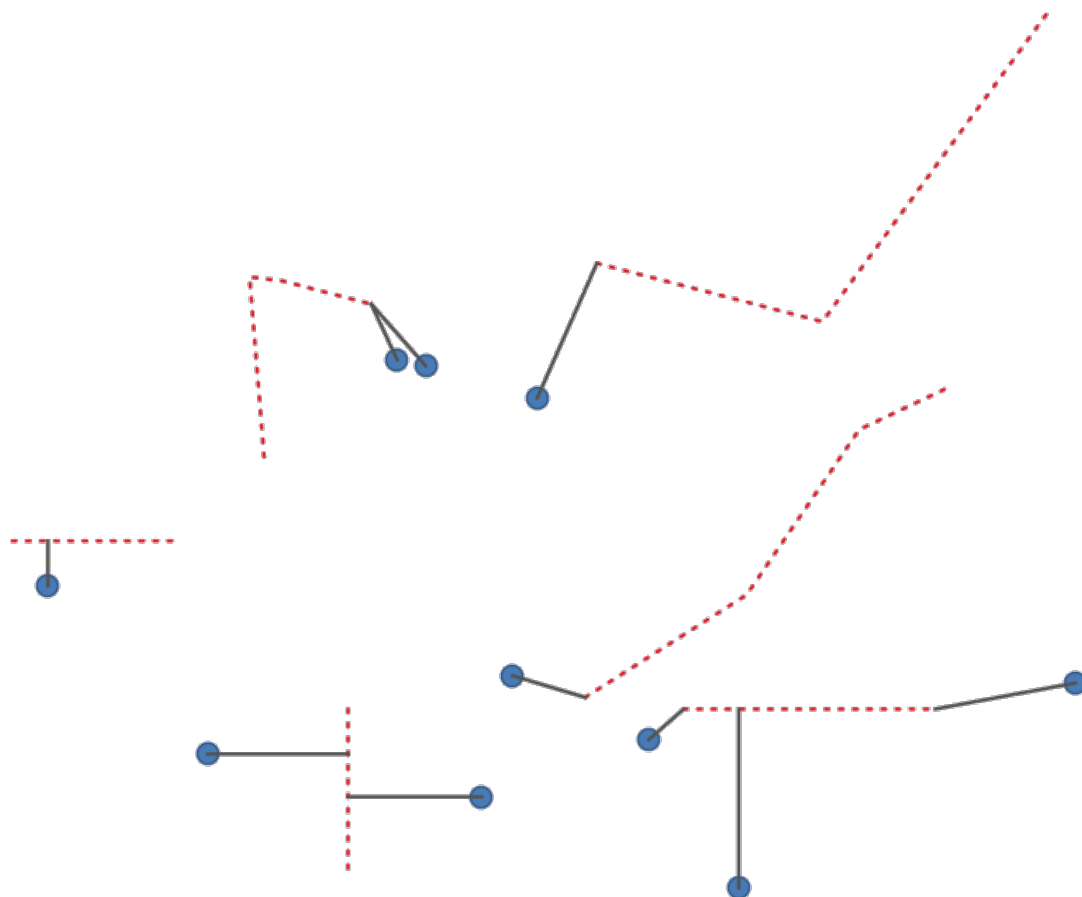


Abb. 28.43: Shortest line from point features to lines

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Source layer</b>	SOURCE	[vector: any]	Origin layer for which to search for nearest neighbors
<b>Destination layer</b>	DESTINATION	[vector: any]	Target Layer in which to search for nearest neighbors
<b>Method</b>	METHOD	[enumeration] Default: 0	Shortest distance calculation method Possible values are: <ul style="list-style-type: none"> <li>• 0 — Distance to nearest point on feature</li> <li>• 1 — Distance to feature centroid</li> </ul>
<b>Maximum number of neighbors</b>	NEIGHBORS	[number] Default: 1	Maximum number of neighbors to look for
<b>Maximum distance</b> Optional	DISTANCE	[number]	Only destination features which are closer than this distance will be considered.
<b>Shortest lines</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[vector: line]	Line vector layer joining source features to their nearest neighbor(s) in the destination layer. Contains all attributes for both source and destination features, and the computed distance.

## Pythoncode

**Algorithm ID:** native:shortestline

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## ST-DBSCAN clustering

Clusters point features based on a 2D implementation of spatiotemporal Density-based clustering of applications with noise (ST-DBSCAN) algorithm.

**Siehe auch:**

*DBSCAN clustering, K-means clustering*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Layer to analyze
<b>Date/time field</b>	DATETIME_FIELD	[tablefield: date]	Field containing the temporal information
<b>Minimum cluster size</b>	MIN_SIZE	[number] Default: 5	Minimum number of features to generate a cluster
<b>Maximum distance between clustered points</b>	EPS	[number] Default: 1.0	Distance beyond which two features can not belong to the same cluster (eps)
<b>Maximum time duration between clustered points</b>	EPS2	[number] Default: 0.0 (days)	Time duration beyond which two features can not belong to the same cluster (eps2). Available time units are milliseconds, seconds, minutes, hours, days and weeks.
<b>Clusters</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the vector layer for the result of the clustering. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Treat border points as noise (DBSCAN*)</b> Optional	DBSCAN*	[boolean] Default: False	If checked, points on the border of a cluster are themselves treated as unclustered points, and only points in the interior of a cluster are tagged as clustered.
<b>Cluster field name</b>	FIELD_NAME	[string] Default: ,CLUSTER_ID'	Name of the field where the associated cluster number shall be stored
<b>Cluster size field name</b>	SIZE_FIELD_NAME	[string] Default: ,CLUSTER_SIZE'	Name of the field with the count of features in the same cluster

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clusters</b>	OUTPUT	[vector: point]	Vector layer containing the original features with a field setting the cluster they belong to
<b>Number of clusters</b>	NUM_CLUSTERS	[number]	The number of clusters discovered

## Pythoncode

**Algorithm ID:** native:stdbscanclustering

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Statistics by categories

Calculates statistics of a field depending on a parent class. The parent class is a combination of values from other fields.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input vector layer</b>	INPUT	[vector: any]	Input vector layer with unique classes and values
<b>Field to calculate statistics on (if empty, only count is calculated)</b>	VALUES_FIELD_N	[tablefield: any]	If empty only the count will be calculated
Optional			
<b>Field(s) with categories</b>	CATEGORIES_FIE	[vector: any] [list]	The fields that (combined) define the categories
<b>Statistics by category</b>	OUTPUT	[table] Default: [Create temporary layer]	Specify the output table for the generated statistics. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Statistics category</b>	<b>by</b> OUTPUT	[table]	Table containing the statistics

Depending on the type of the field being analyzed, the following statistics are returned for each grouped value:

Statistics	Text	Numeric	Datum
Count (COUNT)			
Unique values (UNIQUE)			
Empty (null) values (EMPTY)			
Non-empty values (FILLED)			
Minimal value (MIN)			
Maximal value (MAX)			
Range (RANGE)			
Sum (SUM)			
Mean value (MEAN)			
Median value (MEDIAN)			
Standard Deviation (STD_DEV)			
Coefficient of variation (CV)			
Minority (rarest occurring value - MINORITY)			
Majority (most frequently occurring value - MAJORITY)			
First Quartile (FIRSTQUARTILE)			
Third Quartile (THIRDQUARTILE)			
Inter Quartile Range (IQR)			
Minimum Length (MIN_LENGTH)			
Mean Length (MEAN_LENGTH)			
Maximum Length (MAX_LENGTH)			

## Pythoncode

**Algorithm ID:** qgis:statisticsbycategories

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Sum line lengths

Takes a polygon layer and a line layer and measures the total length of lines and the total number of them that cross each polygon.

The resulting layer has the same features as the input polygon layer, but with two additional attributes containing the length and count of the lines across each polygon.



Allows *features in-place modification* of polygon features

**Default menu:** *Vector ► Analysis Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Lines</b>	LINES	[vector: line]	Input vector line layer
<b>Polygons</b>	POLYGONS	[vector: polygon]	Polygon vector layer
<b>Lines length field name</b>	LEN_FIELD	[string] Default: 'LENGTH'	Name of the field for the lines length
<b>Lines count field name</b>	COUNT_FIELD	[string] Default: 'COUNT'	Name of the field for the lines count
<b>Line length</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon layer with generated statistics. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Line length</b>	OUTPUT	[vector: polygon]	Polygon output layer with fields of lines length and line count

## Pythoncode

**Algorithm ID:** native:sumlinelengths

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

28.1.20 Vector creation

Array of offset (parallel) lines

Creates copies of line features in a layer, by creating multiple offset versions of each feature. Each new version is incrementally offset by a specified distance.

Positive distance will offset lines to the left, and negative distances will offset them to the right.

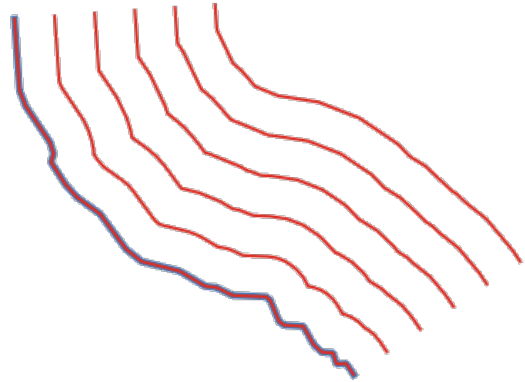


Abb. 28.44: In blue the source layer, in red the offset one



 Allows *features in-place modification* of line features

Siehe auch:

*Offset lines, Array of translated features*

Parameter

Basic parameters

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer to use for the offsets.
Number of features to create	COUNT	[number  ] Default: 10	Number of offset copies to generate for each feature
Offset distance step	OFFSET	[number  ] Default: 1.0	Distance between two consecutive offset copies
Offset lines	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line layer with offset features. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Advanced parameters


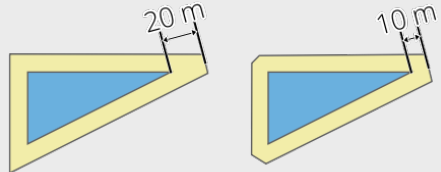
Bezeichnung	Name	Typ	Beschreibung
<b>Segments</b>	SEGMENTS	[number] Default: 8	Number of line segments to use to approximate a quarter circle when creating rounded offsets
<b>Join style</b>	JOIN_STYLE	[enumeration] Default: 0	Specify whether round, miter or beveled joins should be used when offsetting corners in a line. One of: <ul style="list-style-type: none"> <li>• 0 — Round</li> <li>• 1 — Miter</li> <li>• 2 — Bevel</li> </ul> 
<b>Miter limit</b>	MITER_LIMIT	[number] Default: 2.0	Sets the maximum distance from the offset geometry to use when creating a mitered join as a factor of the offset distance (only applicable for miter join styles). Minimum: 1.0 

Abb. 28.45: Round, miter, and bevel join styles

Abb. 28.46: A 10m buffer with a limit of 2 and a 10m buffer with a limit of 1

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Offset lines</b>	OUTPUT	[vector: line]	Output line layer with offset features. The original features are also copied.

## Pythoncode

**Algorithm ID:** native:arrayoffsetlines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Array of translated features

Creates copies of features in a layer by creating multiple translated versions of each. Each copy is incrementally displaced by a preset amount in the X, Y and/or Z axis.

M values present in the geometry can also be translated.

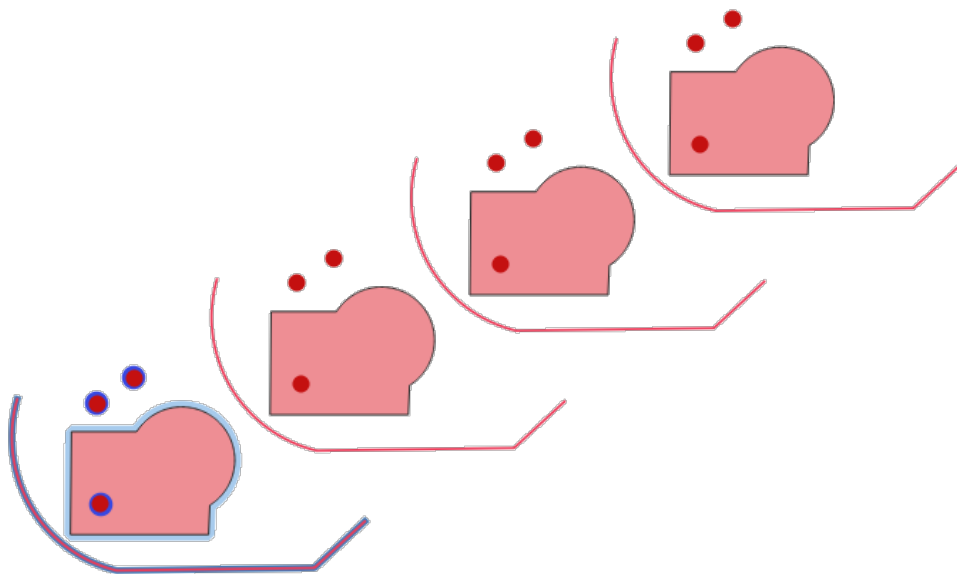







Abb. 28.47: Input layers in blue tones, output layers with translated features in red tones

☒ Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

*Translate, Array of offset (parallel) lines*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer to translate
<b>Number of features to create</b>	COUNT	[number  ] Default: 10	Number of copies to generate for each feature
<b>Step distance (x-axis)</b>	DELTA_X	[number  ] Default: 0.0	Displacement to apply on the X axis
<b>Step distance (y-axis)</b>	DELTA_Y	[number  ] Default: 0.0	Displacement to apply on the Y axis
<b>Step distance (z-axis)</b>	DELTA_Z	[number  ] Default: 0.0	Displacement to apply on the Z axis
<b>Step distance (m values)</b>	DELTA_M	[number  ] Default: 0.0	Displacement to apply on M

Fortsetzung auf der nächsten Seite

Tab. 28.110 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Translated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Output vector layer with translated (moved) copies of the features. The original features are also copied. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Translated</b>	OUTPUT	[same as input]	Output vector layer with translated (moved) copies of the features. The original features are also copied.

## Pythoncode

**Algorithm ID:** native:arraytranslatedfeatures

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create grid

Creates a vector layer with a grid covering a given extent. Grid cells can have different shapes:

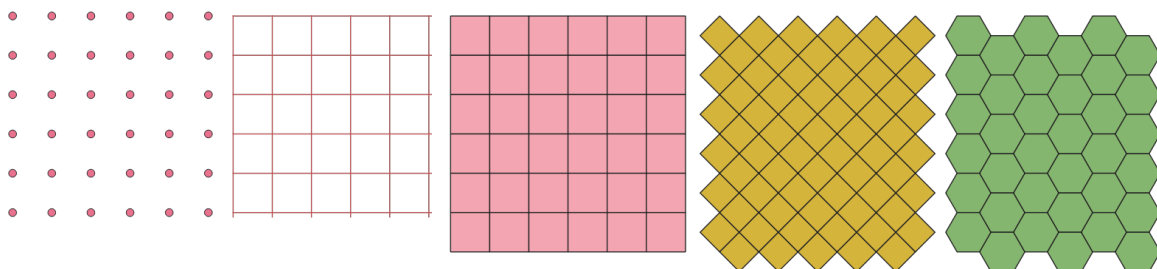


Abb. 28.48: Different grid cell shapes applied to the same extent, without overlaps

The size and/or placement of each element in the grid is defined using a horizontal and vertical spacing. The CRS of the output layer must be defined. The grid extent and the spacing values must be expressed in the coordinates and units of this CRS.

**Default menu:** *Vector ► Research Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Grid type</b>	TYPE	[enumeration] Default: 0	Shape of the grid. One of: <ul style="list-style-type: none"> <li>• 0 — Point</li> <li>• 1 — Line</li> <li>• 2 — Rectangle (polygon)</li> <li>• 3 — Diamond (polygon)</li> <li>• 4 — Hexagon (polygon)</li> </ul>
<b>Grid extent</b>	EXTENT	[extent]	Extent of the grid Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Horizontal spacing</b>	HSPACING	[number] Default: 1.0	Size of a grid cell on the X-axis
<b>Vertical spacing</b>	VSPACING	[number] Default: 1.0	Size of a grid cell on the Y-axis
<b>Horizontal overlay</b>	HOVERLAY	[number] Default: 0.0	Overlay distance between two consecutive grid cells on the X-axis
<b>Vertical overlay</b>	VOVERLAY	[number] Default: 0.0	Overlay distance between two consecutive grid cells on the Y-axis
<b>Grid CRS</b>	CRS	[crs] Default: <i>Project CRS</i>	Coordinate reference system to apply to the grid
<b>Grid</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Resulting vector grid layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Grid</b>	OUTPUT	[vector: any]	Resulting vector grid layer. The output geometry type (point, line or polygon) depends on the <i>Grid type</i> . Features are created from top to bottom, left to right. The attribute table is filled with: <ul style="list-style-type: none"> <li>• an id</li> <li>• coordinates on the left, right, top and bottom sides</li> <li>• and their placement in the grid: row_index and column_index (available for point, rectangle and hexagon grid types)</li> </ul>

## Pythoncode

**Algorithm ID:** native:creategrid

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create points layer from table

Creates points layer from a table with columns that contain coordinates fields.

Besides X and Y coordinates you can also specify Z and M fields.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer or a table.
<b>X field</b>	XFIELD	[tablefield: any]	Field containing the X coordinate
<b>Y field</b>	YFIELD	[tablefield: any]	Field containing the Y coordinate
<b>Z field</b>	ZFIELD	[tablefield: any]	Field containing the Z coordinate
Optional			
<b>M field</b>	MFIELD	[tablefield: any]	Field containing the M value
Optional			
<b>Target CRS</b>	TARGET_CRS	[crs] Default: EPSG:4326	Coordinate reference system to use for layer. The provided coordinates are assumed to be compliant.

Fortsetzung auf der nächsten Seite

Tab. 28.112 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Points from table</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the resulting point layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Points from table</b>	OUTPUT	[vector: point]	The resulting point layer

## Pythoncode

**Algorithm ID:** native:createpointslayerfromtable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Generate points (pixel centroids) along line

Generates a point vector layer from an input raster and line layer.

The points correspond to the pixel centroids that intersect the line layer.

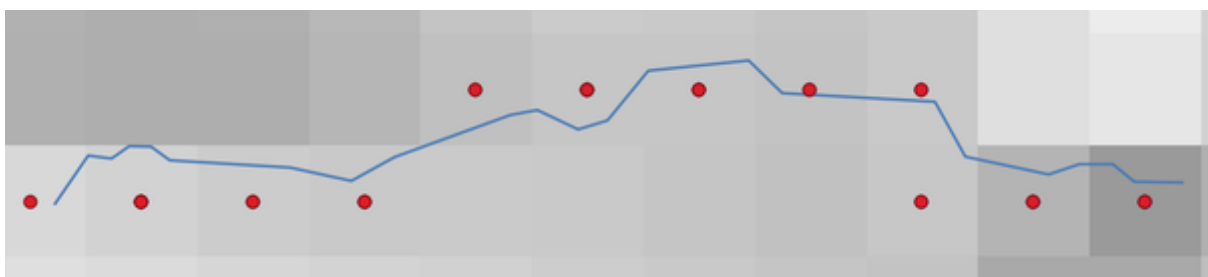


Abb. 28.49: Points of the pixel centroids



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>Vector layer</b>	INPUT_VECTOR	[vector: line]	Input line vector layer
<b>Points along line</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Resulting point layer with pixel centroids. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Points along line</b>	OUTPUT	[vector: point]	Resulting point layer with pixel centroids

## Pythoncode

**Algorithm ID:** qgis:generatepointspixelcentroidsalongline

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Generate points (pixel centroids) inside polygon

Generates a point vector layer from an input raster and polygon layer.

The points correspond to the pixel centroids that intersect the polygon layer.

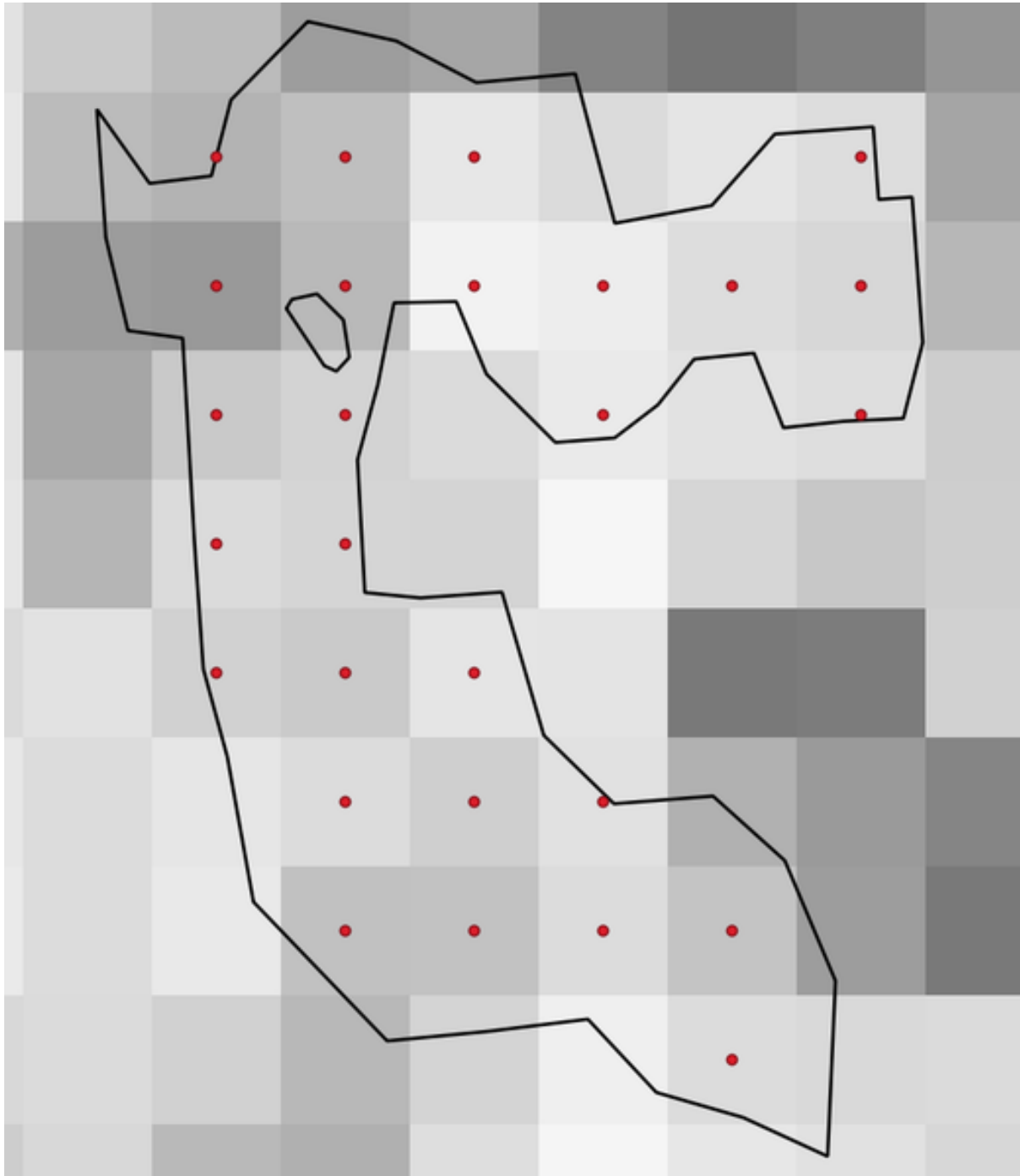


Abb. 28.50: Points of the pixel centroids

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>Vector layer</b>	INPUT_VECTOR	[vector: polygon]	Input polygon vector layer
<b>Points inside polygons</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Resulting point layer of pixel centroids. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Points inside polygons</b>	OUTPUT	[vector: point]	Resulting point layer of pixel centroids

## Pythoncode

**Algorithm ID:** native:generatepointspixelcentroidsinsidepolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Import geotagged photos

Creates a point layer corresponding to the geotagged locations from JPEG or HEIC/HEIF images from a source folder.

The point layer will contain a single PointZ feature per input file from which the geotags could be read. Any altitude information from the geotags will be used to set the point's Z value.

Besides longitude and latitude also altitude, direction and timestamp information, if present in the photo, will be added to the point as attributes.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input folder</b>	FOLDER	[folder]	Path to the source folder containing the geotagged photos
<b>Scan recursively</b>	RECURSIVE	[boolean] Default: False	If checked, the folder and its subfolders will be scanned
<b>Photos</b> Optional	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the point vector layer for the geotagged photos. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Invalid photos table</b> Optional	INVALID	[table] Default: [Skip output]	Specify the table of unreadable or non-geotagged photos. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Photos</b>	OUTPUT	[vector: point]	Point vector layer with geotagged photos. The form of the layer is automatically filled with paths and photo previews settings.
<b>Invalid photos table</b> Optional	INVALID	[table]	Table of unreadable or non-geotagged photos can also be created.

## Pythoncode

**Algorithm ID:** native:importphotos

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Points to path

Converts a point layer to a line layer, by joining points in an order defined by an expression or a field in the input point layer.

Points can be grouped by a field or an expression to distinguish line features.

In addition to the line vector layer, a text file is output that describes the resulting line as a start point and a sequence of bearings / directions (relative to azimuth) and distances.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Create closed paths</b>	CLOSE_PATH	[boolean] Default: False	If checked, the first and last points of the line will be connected and close the generated path
<b>Order expression</b> Optional	ORDER_EXPRESSION	[expression]	Field or expression providing the order to connect the points in the path. If not set, the feature ID ( <code>\$id</code> ) is used.
<b>Sort text containing numbers naturally</b> Optional	NATURAL_SORT	[boolean] Default: False	If checked, naturally sorts the features based on the provided expression (i.e., <code>'a9' &lt; 'a10'</code> ).
<b>Path group expression</b> Optional	GROUP_EXPRESSION	[expression]	Point features of the same value in the field or expression will be grouped in the same line. If not set, a single path is drawn with all the input points.
<b>Paths</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the line vector layer of the path. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Directory for text output</b> Optional	OUTPUT_TEXT_DIRECTORY	[folder] Default: [Skip output]	Specify the directory that will contain the description files of points and paths. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Paths</b>	OUTPUT	[vector: line]	Line vector layer of the path
<b>Directory for text output</b>	OUTPUT_TEXT_DI	[folder]	Directory containing description files of points and paths

## Pythoncode

**Algorithm ID:** native:pointstopath

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points along line

Creates a new point layer, with points placed on the lines of another layer.

For each line in the input layer, a given number of points is added to the resulting layer. The procedure for adding a point is to:

1. randomly select a line feature from the input layer
2. if the feature is multi-part, randomly select a part of it
3. randomly select a segment of that line
4. randomly select a position on that segment.

The procedure means that curved parts of the lines (with relatively short segments) will get more points than straight parts (with relatively long segments), as demonstrated in the illustration below, where the output of the *Random points along lines* algorithm can be compared with the output of the *Random points on lines* algorithm (that produces points with an, on average, even distribution along the lines).

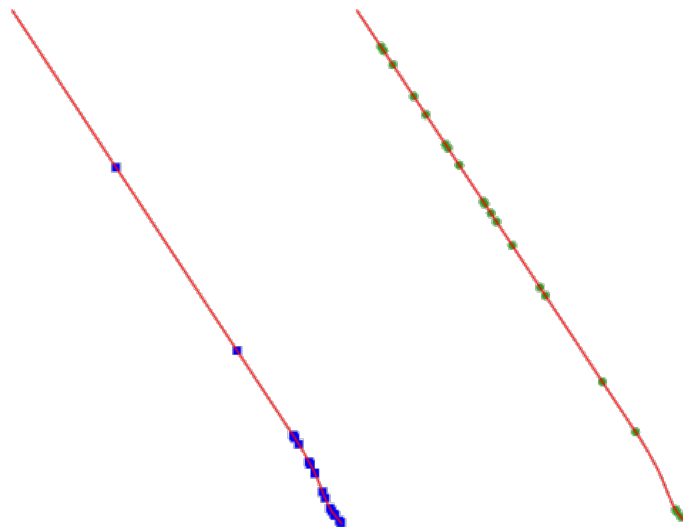


Abb. 28.51: Example algorithm output. Left: *Random points along line*, right: *Random points on lines*

A minimum distance can be specified, to avoid points being too close to each other.

**Siehe auch:**

*Random points on lines*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input point layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Number of points</b>	POINTS_NUMBER	[number] Default: 1	Number of points to create
<b>Minimum distance between points</b>	MIN_DISTANCE	[number] Default: 0.0	The minimum distance between points
<b>Random points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points</b>	OUTPUT	[vector: point]	The output random points layer.

## Pythoncode

**Algorithm ID:** qgis:qgisrandompointsalongline

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points in extent

Creates a new point layer with a given number of random points, all of them within a given extent.

A distance factor can be specified, to avoid points being too close to each other. If the minimum distance between points makes it impossible to create new points, either distance can be decreased or the maximum number of attempts may be increased.

**Default menu:** *Vector ► Research Tools*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input extent</b>	EXTENT	[extent]	Map extent for the random points Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Number of points</b>	POINTS_NUMBER	[number] Default: 1	Number of point to create
<b>Minimum distance between points</b>	MIN_DISTANCE	[number] Default: 0.0	The minimum distance between points
<b>Target CRS</b>	TARGET_CRIS	[crs] Default: <i>Project CRS</i>	CRS of the random points layer
<b>Random points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Maximum number of search attempts given the minimum distance</b>	MAX_ATTEMPTS	[number] Default: 200	Maximum number of attempts to place the points



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points</b>	OUTPUT	[vector: point]	The output random points layer.

## Pythoncode

**Algorithm ID:** `native:randompointsinextent`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points in layer bounds

Creates a new point layer with a given number of random points, all of them within the extent of a given layer.

A minimum distance can be specified, to avoid points being too close to each other.

**Default menu:** *Vector ► Research Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon layer defining the area
<b>Number of points</b>	POINTS_NUMBER	[number] Default: 1	Number of points to create
<b>Minimum distance between points</b>	MIN_DISTANCE	[number] Default: 0.0	The minimum distance between points
<b>Random points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points</b>	OUTPUT	[vector: point]	The output random points layer.

## Pythoncode

**Algorithm ID:** qgis:randompointsinlayerbounds

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points in polygons

Creates a point layer with points placed inside the polygons of another layer.

For each feature (polygon / multi-polygon) geometry in the input layer, the given number of points is added to the result layer.

Per feature and global minimum distances can be specified in order to avoid points being too close in the output point layer. If a minimum distance is specified, it may not be possible to generate the specified number of points for each feature. The total number of generated points and missed points are available as output from the algorithm.

The illustration below shows the effect of per feature and global minimum distances and zero/non-zero minimum distances (generated with the same seed, so at least the first point generated will be the same).

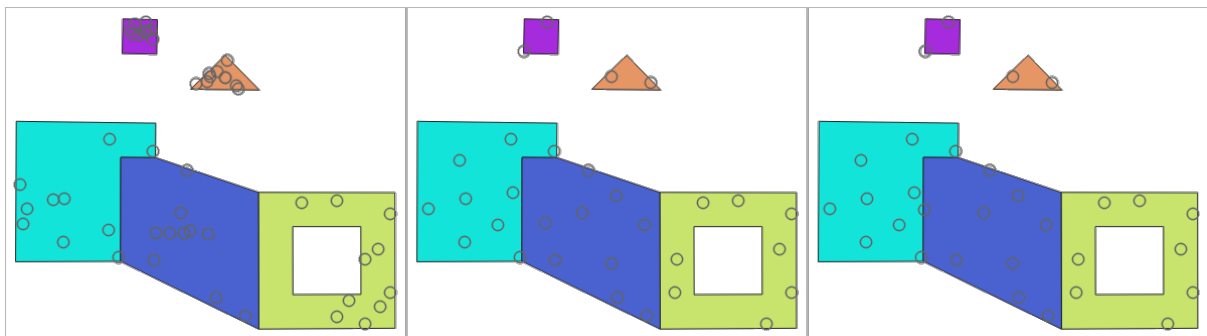


Abb. 28.52: Ten points per polygon feature, *left*: min. distances = 0, *middle*: min.distances = 1, *right*: min. distance = 1, global min. distance = 0

The maximum number of tries per point can be specified. This is only relevant for non-zero minimum distance.

A seed for the random number generator can be provided, making it possible to get identical random number sequences for different runs of the algorithm.

The attributes of the polygon feature on which a point was generated can be included (*Include polygon attributes*).



If you want approximately the same point density for all the features, you can data-define the number of points using the area of the polygon feature geometry.

**Siehe auch:**



[Random points inside polygons](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input polygon vector layer
<b>Number of points for each feature</b>	POINTS_NUMBER	[number]  Default: 1	Number of points to create
<b>Minimum distance between points</b> Optional	MIN_DISTANCE	[number]  Default: 0.0	The minimum distance between points within one polygon feature
<b>Random points in polygons</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Global minimum distance between points</b> Optional	MIN_DISTANCE_G	[number]  Default: 0.0	The global minimum distance between points. Should be smaller than the <i>Minimum distance between points (per feature)</i> for that parameter to have an effect.
<b>Maximum number of search attempts (for Min. dist. &gt; 0)</b> Optional	MAX_TRIES_PER_	[number]  Default: 10	The maximum number of tries per point. Only relevant if the minimum distance between points is set (and greater than 0).
<b>Random seed</b> Optional	SEED	[number] Default: Not set	The seed to use for the random number generator.
<b>Include polygon attributes</b>	INCLUDE_POLYGO	[boolean] Default: True	If set, a point will get the attributes from the line on which it is placed.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points in polygons</b>	OUTPUT	[vector: point]	The output random points layer.
<b>Number of features with empty or no geometry</b>	FEATURES_WITH_	[number]	

Fortsetzung auf der nächsten Seite

Tab. 28.118 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Total number of points generated</b>	OUTPUT_POINTS	[number]	
<b>Number of missed points</b>	POINTS_MISSED	[number]	The number of points that could not be generated due to the minimum distance constraint.
<b>Number of features with missed points</b>	POLYGONS_WITH_	[number]	Not including features with empty or no geometry

## Pythoncode

**Algorithm ID:** native:randompointsinpolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points inside polygons

Creates a new point layer with a given number of random points inside each polygon of the input polygon layer.

Two sampling strategies are available:

- Points count: number of points for each feature
- Points density: density of points for each feature


A minimum distance can be specified, to avoid points being too close to each other.

**Default menu:** *Vector ► Research Tools*

**Siehe auch:**

[Random points in polygons](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Sampling strategy</b>	STRATEGY	[enumeration] Default: 0	Sampling strategy to use. One of: <ul style="list-style-type: none"> <li>• 0 — Points count: number of points for each feature</li> <li>• 1 — Points density: density of points for each feature</li> </ul>
<b>Point count or density</b>	VALUE	[number  ] Default: 1.0	The number or density of points, depending on the chosen <i>Sampling strategy</i> .
<b>Minimum distance between points</b>	MIN_DISTANCE	[number] Default: 0.0	The minimum distance between points

Fortsetzung auf der nächsten Seite

Tab. 28.119 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Random points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points</b>	OUTPUT	[vector: point]	The output random points layer.

## Pythoncode

**Algorithm ID:** qgis:randompointsinsidepolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random points on lines

Creates a point layer with points placed on the lines of another layer.

For each feature (line / multi-line) geometry in the input layer, the given number of points is added to the result layer.

Per feature and global minimum distances can be specified in order to avoid points being too close in the output point layer. If a minimum distance is specified, it may not be possible to generate the specified number of points for each feature. The total number of generated points and missed points are available as output from the algorithm.

The illustration below shows the effect of per feature and global minimum distances and zero/non-zero minimum distances (generated with the same seed, so at least the first point generated will be the same).



Abb. 28.53: Five points per line feature, *left*: min. distances = 0, *middle*: min.distances != 0, *right*: min. distance != 0, global min. distance = 0

The maximum number of tries per point can be specified. This is only relevant for non-zero minimum distance.

A seed for the random number generator can be provided, making it possible to get identical random number sequences for different runs of the algorithm.

The attributes of the line feature on which a point was generated can be included (*Include line attributes*).



If you want approximately the same point density for all the line features, you can data-define the number of points using the length of the line feature geometry.

**Siehe auch:**

*Random points along line*

## Parameter

### Basic parameters



Bezeichnung	Name	Typ	Beschreibung
<b>Input line layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Number of points for each feature</b>	POINTS_NUMBER	[number  ] Default: 1	Number of points to create
<b>Minimum distance between points (per feature)</b> Optional	MIN_DISTANCE	[number  ] Default: 0.0	The minimum distance between points within one line feature

Fortsetzung auf der nächsten Seite

Tab. 28.120 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Random points on lines</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Global minimum distance between points</b> Optional	MIN_DISTANCE_G	[number]  Default: 0.0	The global minimum distance between points. Should be smaller than the <i>Minimum distance between points (per feature)</i> for that parameter to have an effect.
<b>Maximum number of search attempts (for Min. dist. &gt; 0)</b> Optional	MAX_TRIES_PER_	[number]  Default: 10	The maximum number of tries per point. Only relevant if the minimum distance between points is set (and greater than 0).
<b>Random seed</b> Optional	SEED	[number] Default: Not set	The seed to use for the random number generator.
<b>Include line attributes</b>	INCLUDE_LINE_A	[boolean] Default: True	If set, a point will get the attributes from the line on which it is placed.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Random points on lines</b>	OUTPUT	[vector: point]	The output random points layer.
<b>Number of features with empty or no geometry</b>	FEATURES_WITH_	[number]	
<b>Number of features with missed points</b>	LINES_WITH_MIS	[number]	Not including features with empty or no geometry
<b>Total number of points generated</b>	POINTS_GENERAT	[number]	
<b>Number of missed points</b>	POINTS_MISSED	[number]	The number of points that could not be generated due to the minimum distance constraint.

## Pythoncode

**Algorithm ID:** native:randompointsonlines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster pixels to points

Creates a vector layer of points corresponding to each pixel in a raster layer.

Converts a raster layer to a vector layer, by creating point features for each individual pixel's center in the raster layer. Any NoData pixels are skipped in the output.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>Band number</b>	RASTER_BAND	[raster band]	Raster band to extract data from
<b>Field name</b>	FIELD_NAME	[string] Default: 'VALUE'	Name of the field to store the raster band value
<b>Vector points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the resulting point layer of pixels centroids. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vector points</b>	OUTPUT	[vector: point]	Resulting point layer with pixels centroids

## Pythoncode

**Algorithm ID:** native:pixelstopoints

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Raster pixels to polygons

Creates a vector layer of polygons corresponding to each pixel in a raster layer.

Converts a raster layer to a vector layer, by creating polygon features for each individual pixel's extent in the raster layer. Any NoData pixels are skipped in the output.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>Band number</b>	RASTER_BAND	[raster band]	Raster band to extract data from
<b>Field name</b>	FIELD_NAME	[string] Default: 'VALUE'	Name of the field to store the raster band value
<b>Vector polygons</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the resulting polygon layer of pixel extents. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vector polygons</b>	OUTPUT	[vector: polygon]	Resulting polygon layer of pixel extents

### Pythoncode

**Algorithm ID:** native:pixelstopolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Regular points

Creates a new point layer with its points placed in a regular grid within a given extent.

The grid is specified either by the spacing between the points (same spacing for all dimensions) or by the number of points to generate. In the latter case, the spacing will be determined from the extent. In order to generate a full rectangular grid, at least the number of points specified by the user is generated for the latter case.

Random offsets to the point spacing can be applied, resulting in a non-regular point pattern.

**Default menu:** *Vector ► Research Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input extent (xmin, ymin, xmax, ymax)</b>	EXTENT	[extent]	Map extent for the random points Available methods are: <ul style="list-style-type: none"> <li>Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>Use map canvas extent</li> <li>Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Point spacing/count</b>	SPACING	[number] Default: 100	Spacing between the points, or the number of points, depending on whether Use point spacing is checked or not.
<b>Initial inset from corner (LH side)</b>	INSET	[number] Default: 0.0	Offsets the points relative to the upper left corner. The value is used for both the X and Y axis.
<b>Apply random offset to point spacing</b>	RANDOMIZE	[boolean] Default: False	If checked the points will have a random spacing
<b>Use point spacing</b>	IS_SPACING	[boolean] Default: True	If unchecked the point spacing is not taken into account
<b>Output layer CRS</b>	CRS	[crs] Default: <i>Project CRS</i>	CRS of the random points layer
<b>Regular points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output regular point layer. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Regular points</b>	OUTPUT	[vector: point]	The output regular point layer.

## Pythoncode

**Algorithm ID:** qgis:regularpoints

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.21 Vector general

### Projektion zuweisen

Weist einem Vektorlayer eine neue Projektion zu.

Erzeugt einen neuen Layer mit exakt den gleichen Objekten und Geometrien wie die Eingabe, aber mit einem neu zugewiesenen KBS. Die Geometrien werden **nicht** umprojiziert, sondern es wird nur ein neues KBS zugewiesen.

Der Algorithmus kann benutzt werden, um Layer zu reparieren, denen eine falsche Projektion zugewiesen wurde.

Attribute werden von diesem Algorithmus nicht verändert.

**Siehe auch:**

*Define Shapefile projection, Find projection, Reproject layer*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer with wrong or missing CRS
<b>Assigned CRS</b>	CRS	[crs] Default: EPSG:4326 – WGS84	Select the new CRS to assign to the vector layer
<b>Assigned CRS</b> Optional	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Assigned CRS</b>	OUTPUT	[same as input]	Vector layer with assigned projection

## Pythoncode

**Algorithm ID:** native:assignprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Batch Nominatim geocoder

Performs batch geocoding using the Nominatim service against an input layer string field. The output layer will have a point geometry reflecting the geocoded location as well as a number of attributes associated to the geocoded location.



Allows *features in-place modification* of point features

**Bemerkung:** This algorithm is compliant with the [usage policy](#) of the Nominatim geocoding service provided by the OpenStreetMap Foundation.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer to geocode the features
<b>Address field</b>	FIELD	[tablefield: string]	Field containing the addresses to geocode
<b>Geocoded</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output layer containing only the geocoded addresses. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> <li>Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Geocoded</b>	OUTPUT	[vector: point]	Vector layer with point features corresponding to the geocoded addresses

## Pythoncode

**Algorithm ID:** native:batchnominatimgeocoder

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert layer to spatial bookmarks

Creates spatial bookmarks corresponding to the extent of features contained in a layer.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: line, polygon]	The input vector layer
<b>Bookmark destination</b>	DESTINATION	[enumeration] Default: 0	Select the destination for the bookmarks. One of: <ul style="list-style-type: none"> <li>0 — Project bookmarks</li> <li>1 — User bookmarks</li> </ul>
<b>Name field</b>	NAME_EXPRESSION	[expression]	Field or expression that will give names to the generated bookmarks
<b>Group field</b>	GROUP_EXPRESSION	[expression]	Field or expression that will provide groups for the generated bookmarks

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Count of bookmarks added</b>	COUNT	[number]	

## Pythoncode

**Algorithm ID:** `native:layertobookmarks`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert spatial bookmarks to layer

Creates a new layer containing polygon features for stored spatial bookmarks. The export can be filtered to only bookmarks belonging to the current project, to all user bookmarks, or a combination of both.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Bookmark source</b>	SOURCE	[enumeration] [list] Default: [0,1]	Select the source(s) of the bookmarks. One or more of: <ul style="list-style-type: none"> <li>• 0 — Project bookmarks</li> <li>• 1 — User bookmarks</li> </ul>
<b>Output CRS</b>	CRS	[crs] Default: EPSG:4326 – WGS 84	The CRS of the output layer
<b>Ergebnis</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[vector: polygon]	The output (bookmarks) vector layer

## Pythoncode

**Algorithm ID:** native:bookmarkstolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create attribute index

Creates an index against a field of the attribute table to speed up queries. The support for index creation depends on both the layer's data provider and the field type.

No outputs are created: the index is stored on the layer itself.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Select the vector layer you want to create an attribute index for
<b>Attribute to index</b>	FIELD	[tablefield: any]	Field of the vector layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Indexed layer</b>	OUTPUT	[same as input]	A copy of the input vector layer with an index for the specified field

## Pythoncode

**Algorithm ID:** native:createattributeindex

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create spatial index

Creates an index to speed up access to the features in a layer based on their spatial location. Support for spatial index creation is dependent on the layer's data provider.

No new output layers are created.

**Default menu:** *Vector ► Data Management Tools*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
Indexed layer	OUTPUT	[same as input]	A copy of the input vector layer with a spatial index

### Pythoncode

**Algorithm ID:** `native:createspatialindex`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Define Shapefile projection

Sets the CRS (projection) of an existing Shapefile format dataset to the provided CRS. It is very useful when a Shapefile format dataset is missing the `prj` file and you know the correct projection.

Contrary to the [Projektion zuweisen](#) algorithm, it modifies the current layer and will not output a new layer.

---

**Bemerkung:** For Shapefile datasets, the `.prj` and `.qpj` files will be overwritten - or created if missing - to match the provided CRS.

---

**Default menu:** *Vector ► Data Management Tools*

**Siehe auch:**

*Projektion zuweisen, Find projection, Reproject layer*



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer with missing projection information
<b>CRS</b>	CRS	[crs]	Select the CRS to assign to the vector layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
	INPUT	[same as input]	The input vector layer with the defined projection

## Pythoncode

**Algorithm ID:** qgis:definecurrentprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Delete duplicate geometries

Finds and removes duplicated geometries.

Attributes are not checked, so in case two features have identical geometries but different attributes, only one of them will be added to the result layer.

### Siehe auch:

*Drop geometries, Remove null geometries, Delete duplicates by attribute*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The layer with duplicate geometries you want to clean
<b>Cleaned</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Count of discarded duplicate records</b>	DUPLICATE_COUNT	[number]	Count of discarded duplicate records
<b>Cleaned</b>	OUTPUT	[same as input]	The output layer without any duplicated geometries
<b>Count of retained records</b>	RETAINED_COUNT	[number]	Count of unique records

## Pythoncode

**Algorithm ID:** native:deleteduplicategeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Delete duplicates by attribute

Deletes duplicate rows by only considering the specified field / fields. The first matching row will be retained, and duplicates will be discarded.

Optionally, these duplicate records can be saved to a separate output for analysis.

**Siehe auch:**

[Delete duplicate geometries](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer
<b>Fields to match duplicates by</b>	FIELDS	[tablefield: any] [list]	Fields defining duplicates. Features with identical values for all these fields are considered duplicates.
<b>Filtered duplicates) (no</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output layer containing the unique features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 28.124 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Filtered (duplicates)</b> Optional	DUPLICATES	[same as input] Default: [Skip output]	Specify the output layer containing only the duplicates. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Filtered (duplicates)</b> Optional	DUPLICATES	[same as input] Default: [Skip output]	Vector layer containing the removed features. Will not be produced if not specified (left as [Skip output]).
<b>Count of discarded duplicate records</b>	DUPLICATE_COUNT	[number]	Count of discarded duplicate records
<b>Filtered (no duplicates)</b>	OUTPUT	[same as input]	Vector layer containing the unique features.
<b>Count of retained records</b>	RETAINED_COUNT	[number]	Count of unique records

## Pythoncode

**Algorithm ID:** native:removeduplicatesbyattribute

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Detect dataset changes

Compares two vector layers, and determines which features are unchanged, added or deleted between the two. It is designed for comparing two different versions of the same dataset.

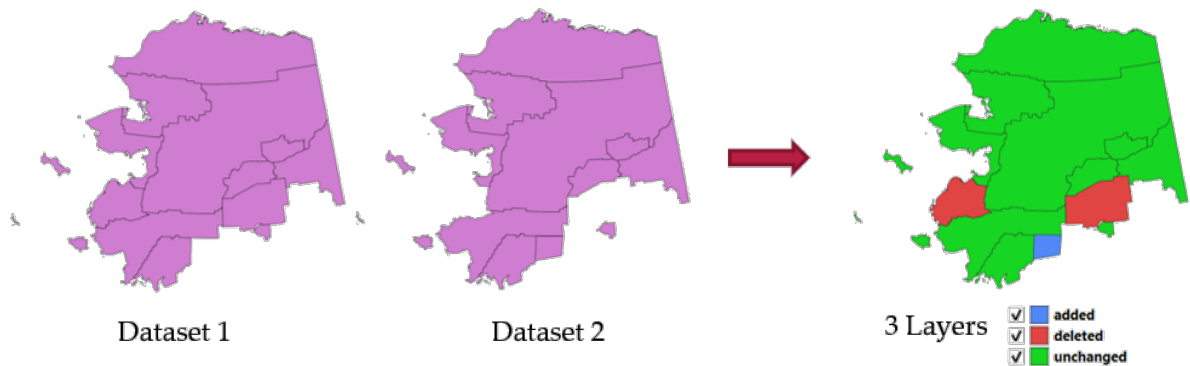


Abb. 28.54: Detect dataset change example

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Original layer</b>	ORIGINAL	[vector: any]	The vector layer considered as the original version
<b>Revised layer</b>	REVISED	[vector: any]	The revised or modified vector layer
<b>Attributes to consider for match</b> Optional	COMPARE_ATTRIB	[tablefield: any] [list]	Attributes to consider for match. By default, all attributes are compared.
<b>Geometry comparison behavior</b> Optional	MATCH_TYPE	[enumeration] Default: 1	Defines the criteria for comparison. Options: <ul style="list-style-type: none"><li>• 0 — Exact Match: includes the order and vertices count of geometries</li><li>• 1 — Tolerant Match (Topological Equality): geometries are considered equal</li></ul>
<b>Unchanged features</b> Optional	UNCHANGED	[vector: same as Original layer]	Specify the output vector layer containing the unchanged features. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 28.126 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Added features</b> Optional	ADDED	[vector: same as Original layer]	Specify the output vector layer containing the added features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Deleted features</b> Optional	DELETED	[vector: same as Original layer]	Specify the output vector layer containing the deleted features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Unchanged features</b>	UNCHANGED	[vector: same as Original layer]	Vector layer containing the unchanged features.
<b>Added features</b>	ADDED	[vector: same as Original layer]	Vector layer containing the added features.
<b>Deleted features</b>	DELETED	[vector: same as Original layer]	Vector layer containing the deleted features.
<b>Count of unchanged features</b>	UNCHANGED_COUNT	[number]	Count of unchanged features.
<b>Count of features added in revised layer</b>	ADDED_COUNT	[number]	Count of features added in revised layer.
<b>Count of features deleted from original layer</b>	DELETED_COUNT	[number]	Count of features deleted from original layer.

## Pythoncode

**Algorithm ID:** native:detectvectorchanges

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Drop geometries

Creates a simple *geometryless* copy of the input layer attribute table. It keeps the attribute table of the source layer.

If the file is saved in a local folder, you can choose between many file formats.



Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

*Delete duplicate geometries, Remove null geometries*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Dropped geometries</b>	OUTPUT	[table]	Specify the output geometryless layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Dropped geometries</b>	OUTPUT	[table]	The output geometryless layer. A copy of the original attribute table.

## Pythoncode

**Algorithm ID:** native:dropgeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Execute SQL

Runs a simple or complex query based only on SELECT with SQL syntax on the source layer.

Input datasources are identified with `input1`, `input2`... `inputN` and a simple query will look like `SELECT * FROM input1`.

Beside a simple query, you can add expressions or variables within the `SQL query` parameter itself. This is particularly useful if this algorithm is executed within a Processing model and you want to use a model input as a parameter of the query. An example of a query will then be `SELECT * FROM [% @table %]` where `@table` is the variable that identifies the model input.

The result of the query will be added as a new layer.

**Siehe auch:**

*SpatiaLite execute SQL, PostgreSQL execute SQL*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Additional input datasources</b> (called <code>input1</code> , ..., <code>inputN</code> in the query)	INPUT_DATASOUR	[vector: any] [list]	List of layers to query. In the SQL editor you can refer these layers with their <b>real</b> name or also with <b>input1</b> , <b>input2</b> , <b>inputN</b> depending on how many layers have been chosen.
<b>SQL query</b>	INPUT_QUERY	[string]	Type the string of your SQL query, e.g. <code>SELECT * FROM input1</code> .
<b>Unique identifier field</b> Optional	INPUT_UID_FIEL	[string]	Specify the column with unique ID
<b>Geometry field</b> Optional	INPUT_GEOMETRY	[string]	Specify the geometry field
<b>Geometry type</b> Optional	INPUT_GEOMETRY	[enumeration] Default: 0	Choose the geometry of the result. By default the algorithm will autodetect it. One of: <ul style="list-style-type: none"> <li>• 0 — Autodetect</li> <li>• 1 — No geometry</li> <li>• 2 — Point</li> <li>• 3 — LineString</li> <li>• 4 — Polygon</li> <li>• 5 — MultiPoint</li> <li>• 6 — MultiLineString</li> <li>• 7 — MultiPolygon</li> </ul>
<b>CRS</b> Optional	INPUT_GEOMETRY	[crs]	The CRS to assign to the output layer
<b>SQL Output</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specify the output layer created by the query. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>SQL Output</b>	OUTPUT	[vector: any]	Vector layer created by the query

## Pythoncode

**Algorithm ID:** qgis:executesql

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export layers to DXF

Exports layers to DXF file. For each layer, you can choose a field whose values are used to split features in generated destination layers in DXF output.

**Siehe auch:**

[Creating new DXF files](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	LAYERS	[vector: any] [list]	Input vector layers to export
<b>Symbology mode</b>	SYMBOLGY_MODE	[enumeration] Default: 0	Type of symbology to apply to output layers. You can choose between: <ul style="list-style-type: none"> <li>0 — No Symbology</li> <li>1 — Feature Symbology</li> <li>2 — Symbol Layer Symbology</li> </ul>
<b>Symbology scale</b>	SYMBOLGY_SCAL	[scale] Default: 1:1 000 000	Default scale of data export.
<b>Encoding</b>	ENCODING	[enumeration]	Encoding to apply to layers.
<b>CRS</b>	CRS	[crs]	Choose the CRS for the output layer.
<b>Use layer title as name</b>	USE_LAYER_TITL	[boolean] Default: False	Name the output layer with the layer title (as set in QGIS) instead of the layer name.
<b>Force 2D</b>	FORCE_2D	[boolean] Default: False	
<b>Export labels as MTEXT elements</b>	MTEXT	[boolean] Default: False	Exports labels as MTEXT or TEXT elements
<b>DXF</b>	OUTPUT	[file] Default: [Save to temporary file]	Specification of the output DXF file. One of: <ul style="list-style-type: none"> <li>Save to a Temporary File</li> <li>Save to File...</li> </ul>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>DXF</b>	OUTPUT	[file]	.DXF file containing the input layers

## Pythoncode

**Algorithm ID:** native:dxfexport

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract selected features

Saves the selected features as a new layer.

**Bemerkung:** If the selected layer has no selected features, the newly created layer will be empty.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Layer to save the selection from
<b>Selected features</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the vector layer for the selected features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Selected features</b>	OUTPUT	[same as input]	Vector layer with only the selected features, or no feature if none was selected.

## Pythoncode

**Algorithm ID:** native:savesselectedfeatures

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract Shapefile encoding

Extracts the attribute encoding information embedded in a Shapefile. Both the encoding specified by an optional .cpg file and any encoding details present in the .dbf LDID header block are considered.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	ESRI Shapefile (.SHP) Layer to extract the encoding information.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Shapefile encoding</b>	ENCODING	[string]	Encoding information specified in the input file
<b>CPG encoding</b>	CPG_ENCODING	[string]	Encoding information specified in any optional .CPG file
<b>LDID encoding</b>	LDID_ENCODING	[string]	Encoding information specified in .dbf LDID header block

## Pythoncode

**Algorithm ID:** native:shpencodinginfo

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Find projection

Creates a shortlist of candidate coordinate reference systems, for instance for a layer with an unknown projection.

The area that the layer is expected to cover must be specified via the target area parameter. The coordinate reference system for this target area must be known to QGIS.

The algorithm operates by testing the layer's extent in every known reference system and then listing any for which the bounds would be near the target area if the layer was in this projection.

### Siehe auch:

*Projektion zuweisen, Define Shapefile projection, Reproject layer*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Layer with unknown projection
<b>Target area for layer (xmin, xmax, ymin, ymax)</b>	TARGET_AREA	[extent]	<p>The area that the layer covers.</p> <p>Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>CRS candidates</b>	OUTPUT	<p>[table]</p> <p>Default: [Create temporary layer]</p>	<p>Specify the table (geometryless layer) for the CRS suggestions (EPSG codes). One of:</p> <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> <p>The file encoding can also be changed here.</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>CRS candidates</b>	OUTPUT	[table]	A table with all the CRS (EPSG codes) of the matching criteria.

## Pythoncode

**Algorithm ID:** qgis:findprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Flatten relationship

Flattens a *relationship* for a vector layer, exporting a single layer containing one parent feature per related child feature. This master feature contains all the attributes for the related features. This allows to have the relation as a plain table that can be e.g. exported to CSV.

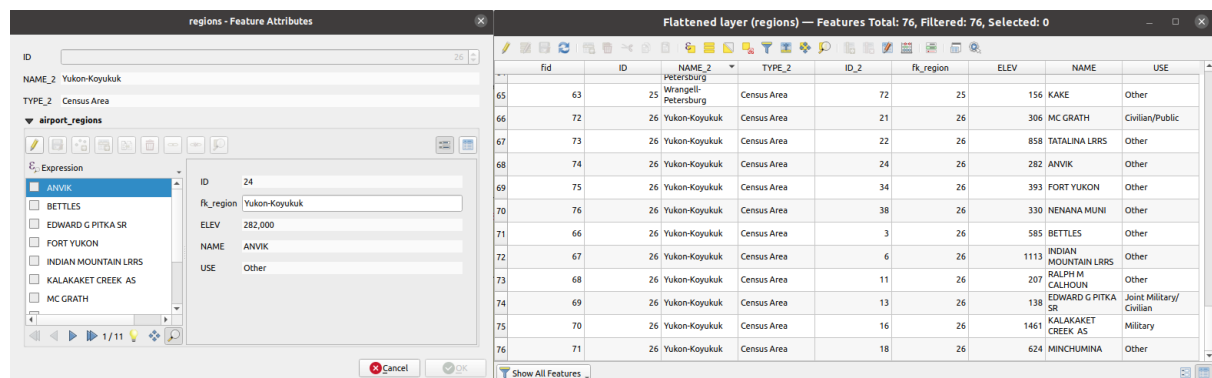


Abb. 28.55: Form of a region with related children (left) - A duplicate region feature for each related child, with joined attributes (right)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Layer with the relationship that should be de-normalized
<b>Flattened Layer</b> Optional	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (flattened) layer. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Flattened layer</b>	OUTPUT	[same as input]	A layer containing master features with all the attributes for the related features

## Pythoncode

**Algorithm ID:** native:flattenrelationships

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Join attributes by field value

Takes an input vector layer and creates a new vector layer that is an extended version of the input one, with additional attributes in its attribute table.

The additional attributes and their values are taken from a second vector layer. An attribute is selected in each of them to define the join criteria.

**Siehe auch:**

*Join attributes by nearest*, *Join attributes by location*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer. The output layer will consist of the features of this layer with attributes from matching features in the second layer.
<b>Table field</b>	FIELD	[tablefield: any]	Field of the source layer to use for the join
<b>Input layer 2</b>	INPUT_2	[vector: any]	Layer with the attribute table to join
<b>Table field 2</b>	FIELD_2	[tablefield: any]	Field of the second (join) layer to use for the join The type of the field must be equal to (or compatible with) the input table field type.
<b>Layer 2 fields to copy</b> Optional	FIELDS_TO_COPY	[tablefield: any] [list]	Select the specific fields you want to add. By default all the fields are added.
<b>Join type</b>	METHOD	[enumeration] Default: 1	The type of the final joined layer. One of: <ul style="list-style-type: none"> <li>0 — Create separate feature for each matching feature (one-to-many)</li> <li>1 — Take attributes of the first matching feature only (one-to-one)</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.132 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Discard records which could not be joined</b>	DISCARD_NONMAT	[boolean] Default: True	Check if you don't want to keep the features that could not be joined
<b>Joined field prefix</b> Optional	PREFIX	[string]	Add a prefix to joined fields in order to easily identify them and avoid field name collision
<b>Joined layer</b> Optional	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the join. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Unjoinable features from first layer</b> Optional	NON_MATCHING	[same as input] Default: [Skip output]	Specify the output vector layer for unjoinable features from first layer. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Number of joined features from input table</b>	JOINED_COUNT	[number]	
<b>Unjoinable features from first layer</b> Optional	NON_MATCHING	[same as input]	Vector layer with the non-matched features
<b>Joined layer</b> Optional	OUTPUT	[same as input]	Output vector layer with added attributes from the join
<b>Number of unjoinable features from input table</b> Optional	UNJOINABLE_COU	[number]	

## Pythoncode

**Algorithm ID:** native:joinattributestable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Join attributes by location

Takes an input vector layer and creates a new vector layer that is an extended version of the input one, with additional attributes in its attribute table.

The additional attributes and their values are taken from a second vector layer. A spatial criteria is applied to select the values from the second layer that are added to each feature from the first layer.

**Default menu:** *Vector ► Data Management Tools*

**Siehe auch:**

*Join attributes by nearest, Join attributes by field value, Join attributes by location (summary)*

## Exploring spatial relations

Geometrische Prädikate sind boolesche Funktionen, die verwendet werden, um die räumliche Beziehung eines Features zu einem anderen zu bestimmen, indem verglichen wird, ob und wie ihre Geometrien einen Teil des Raums teilen.

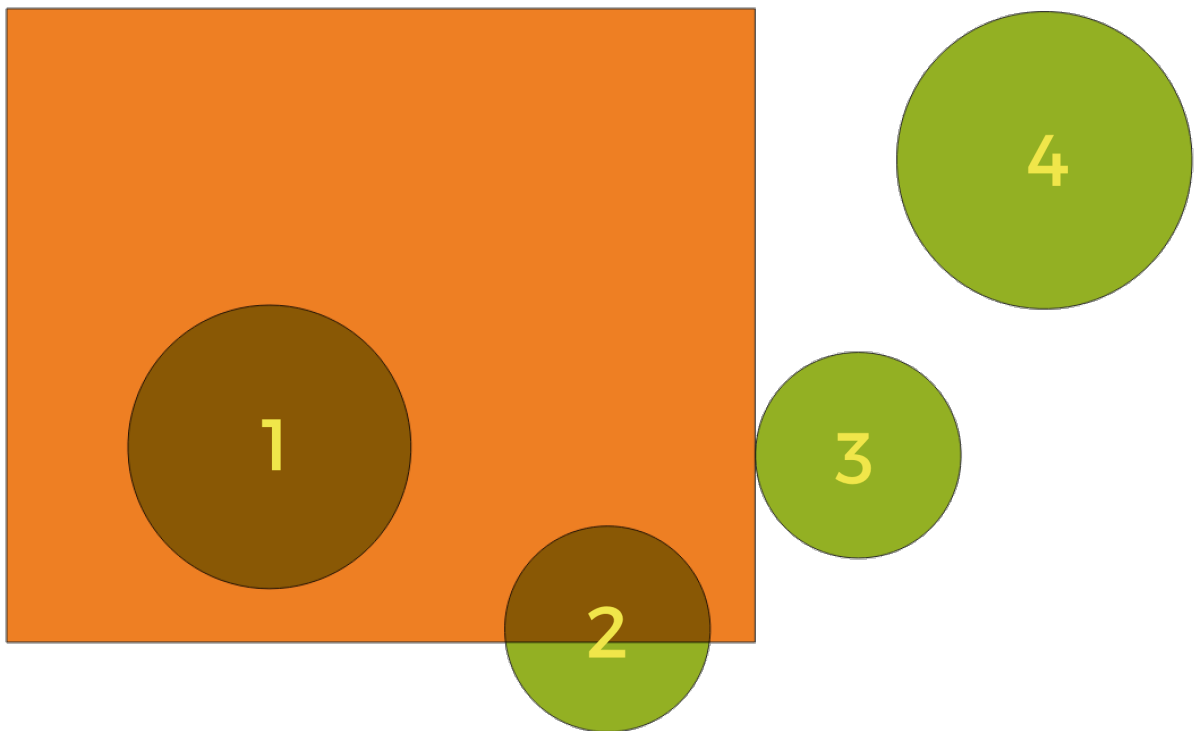


Abb. 28.56: Suche nach räumlichen Beziehungen zwischen Layern

Unter Verwendung der obigen Abbildung suchen wir nach den grünen Kreisen, indem wir sie räumlich mit dem orangefarbenen Rechteck-Feature vergleichen. Verfügbare geometrische Prädikate sind:

### **Schneidet**

Testet, ob eine Geometrie eine andere schneidet. Gibt 1 (wahr) zurück, wenn sich die Geometrien räumlich überschneiden (einen Teil des Raums teilen – überlappen oder berühren) und 0, wenn dies nicht der Fall ist. Im obigen Bild werden die Kreise 1, 2 und 3 zurückgegeben.

### **Enthält**

Gibt 1 (wahr) zurück, wenn und nur wenn keine Punkte von b außerhalb von a liegen und mindestens ein Punkt des Inneren von b innerhalb von a liegt. Im Bild wird kein Kreis zurückgegeben, sondern das Rechteck wäre es, wenn man es umgekehrt suchen würde, da es den Kreis 1 komplett enthält. Das ist das Gegenteil von *sind innerhalb*.

### **Getrennt**

Gibt 1 (wahr) zurück, wenn die Geometrien keinen Teil des Raums teilen (keine Überlappung, keine Berührung). Nur Kreis 4 wird zurückgegeben.

### **Gleich**

Gibt 1 (wahr) zurück, wenn und nur wenn die Geometrien genau gleich sind. Es werden keine Kreise zurückgegeben.

### **Berührt**

Testet, ob eine Geometrie eine andere berührt. Gibt 1 (wahr) zurück, wenn die Geometrien mindestens einen Punkt gemeinsam haben, aber ihre Innenräume sich nicht schneiden. Nur Kreis 3 wird zurückgegeben.

### **Überlappt**

Testet, ob eine Geometrie eine andere überlappt. Gibt 1 (wahr) zurück, wenn die Geometrien sich den Raum teilen, dieselbe Dimension haben, aber nicht vollständig ineinander liegen. Nur Kreis 2 wird zurückgegeben.

### **Sind innerhalb**

Testet, ob sich eine Geometrie in einer anderen befindet. Gibt 1 (wahr) zurück, wenn sich Geometrie a vollständig innerhalb von Geometrie b befindet. Nur Kreis 1 wird zurückgegeben.

### **Kreuzen**

Gibt 1 (wahr) zurück, wenn die bereitgestellten Geometrien einige, aber nicht alle inneren Punkte gemeinsam haben und die tatsächliche Kreuzung eine geringere Dimension als die höchste bereitgestellte Geometrie hat. Beispielsweise wird eine Linie, die ein Polygon kreuzt, als Linie gekreuzt (wahr). Zwei Linien, die sich kreuzen, kreuzen sich als Punkt (wahr). Zwei Polygone kreuzen sich als Polygon (falsch). Im Bild werden keine Kreise zurückgegeben.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Join to features in</b>	INPUT	[vector: any]	Input vector layer. The output layer will consist of the features of this layer with attributes from matching features in the second layer.

Fortsetzung auf der nächsten Seite



Tab. 28.133 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Where the features</b>	PREDICATE	[enumeration] [list] Default: [0]	Type of spatial relation the source feature should have with the target feature so that they could be joined. One or more of: <ul style="list-style-type: none"> <li>• 0 — intersect</li> <li>• 1 — contain</li> <li>• 2 — equal</li> <li>• 3 — touch</li> <li>• 4 — overlap</li> <li>• 5 — are within</li> <li>• 6 — cross</li> </ul> If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.
<b>By comparing to</b>	JOIN	[vector: any]	The join layer. Features of this vector layer will <b>add</b> their attributes to the source layer attribute table if they satisfy the spatial relationship.
<b>Fields to add (leave empty to use all fields)</b> Optional	JOIN_FIELDS	[tablefield: any] [list]	Select the specific fields you want to add from the join layer. By default all the fields are added.
<b>Join type</b>	METHOD	[enumeration]	The type of the final joined layer. One of: <ul style="list-style-type: none"> <li>• 0 — Create separate feature for each matching feature (one-to-many)</li> <li>• 1 — Take attributes of the first matching feature only (one-to-one)</li> <li>• 2 — Take attributes of the feature with largest overlap only (one-to-one)</li> </ul>
<b>Discard records which could not be joined</b>	DISCARD_NONMAT	[boolean] Default: False	Remove from the output the input layer's features which could not be joined
<b>Joined field prefix</b> Optional	PREFIX	[string]	Add a prefix to joined fields in order to easily identify them and avoid field name collision
<b>Joined layer</b> Optional	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the join. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Unjoinable features from first layer</b> Optional	NON_MATCHING	[same as input] Default: [Skip output]	Specify the output vector layer for unjoinable features from first layer. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
Number of joined features from input table	JOINED_COUNT	[number]	
Unjoinable features from first layer	NON_MATCHING	[same as input]	Vector layer of the non-matched features
Optional Joined layer	OUTPUT	[same as input]	Output vector layer with added attributes from the join

## Pythoncode

**Algorithm ID:** native:joinattributesbylocation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Join attributes by location (summary)

Takes an input vector layer and creates a new vector layer that is an extended version of the input one, with additional attributes in its attribute table.

The additional attributes and their values are taken from a second vector layer. A spatial criteria is applied to select the values from the second layer that are added to each feature from the first layer.

The algorithm calculates a statistical summary for the values from matching features in the second layer (e.g. maximum value, mean value, etc).

### Siehe auch:

*[Join attributes by location](#)*

## Exploring spatial relations

Geometrische Prädikate sind boolesche Funktionen, die verwendet werden, um die räumliche Beziehung eines Features zu einem anderen zu bestimmen, indem verglichen wird, ob und wie ihre Geometrien einen Teil des Raums teilen.

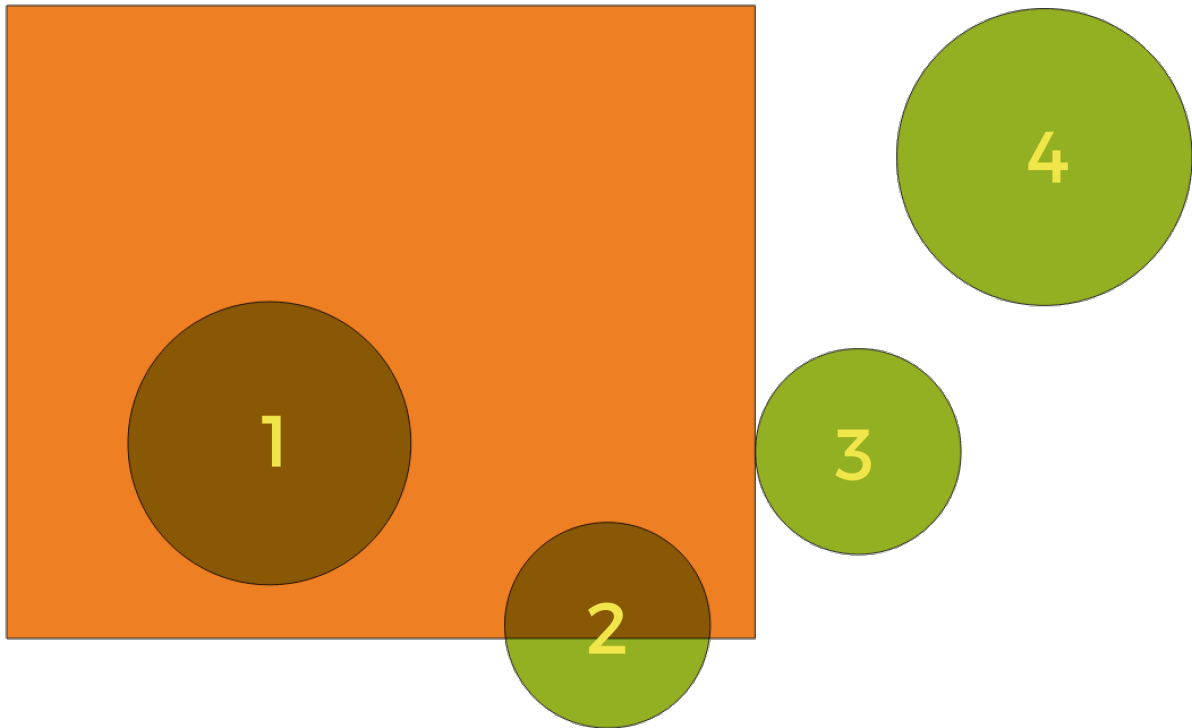


Abb. 28.57: Suche nach räumlichen Beziehungen zwischen Layern

Unter Verwendung der obigen Abbildung suchen wir nach den grünen Kreisen, indem wir sie räumlich mit dem orangefarbenen Rechteck-Feature vergleichen. Verfügbare geometrische Prädikate sind:

**Schneidet**

Testet, ob eine Geometrie eine andere schneidet. Gibt 1 (wahr) zurück, wenn sich die Geometrien räumlich überschneiden (einen Teil des Raums teilen – überlappen oder berühren) und 0, wenn dies nicht der Fall ist. Im obigen Bild werden die Kreise 1, 2 und 3 zurückgegeben.

**Enthält**

Gibt 1 (wahr) zurück, wenn und nur wenn keine Punkte von b außerhalb von a liegen und mindestens ein Punkt des Inneren von b innerhalb von a liegt. Im Bild wird kein Kreis zurückgegeben, sondern das Rechteck wäre es, wenn man es umgekehrt suchen würde, da es den Kreis 1 komplett enthält. Das ist das Gegenteil von *sind innerhalb*.

**Getrennt**

Gibt 1 (wahr) zurück, wenn die Geometrien keinen Teil des Raums teilen (keine Überlappung, keine Berührung). Nur Kreis 4 wird zurückgegeben.

**Gleich**

Gibt 1 (wahr) zurück, wenn und nur wenn die Geometrien genau gleich sind. Es werden keine Kreise zurückgegeben.

**Berührt**

Testet, ob eine Geometrie eine andere berührt. Gibt 1 (wahr) zurück, wenn die Geometrien mindestens einen Punkt gemeinsam haben, aber ihre Innenräume sich nicht schneiden. Nur Kreis 3 wird zurückgegeben.

**Überlappt**

Testet, ob eine Geometrie eine andere überlappt. Gibt 1 (wahr) zurück, wenn die Geometrien sich den Raum teilen, dieselbe Dimension haben, aber nicht vollständig ineinander liegen. Nur Kreis 2 wird zurückgegeben.

**Sind innerhalb**

Testet, ob sich eine Geometrie in einer anderen befindet. Gibt 1 (wahr) zurück, wenn sich Geometrie a vollständig innerhalb von Geometrie b befindet. Nur Kreis 1 wird zurückgegeben.

**Kreuzen**

Gibt 1 (wahr) zurück, wenn die bereitgestellten Geometrien einige, aber nicht alle inneren Punkte gemeinsam haben und die tatsächliche Kreuzung eine geringere Dimension als die höchste bereitgestellte Geometrie hat. Beispielsweise wird eine Linie, die ein Polygon kreuzt, als Linie gekreuzt (wahr). Zwei Linien, die sich kreuzen, kreuzen sich als Punkt (wahr). Zwei Polygone kreuzen sich als Polygon (falsch). Im Bild werden keine Kreise zurückgegeben.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Join to features in</b>	INPUT	[vector: any]	Input vector layer. The output layer will consist of the features of this layer with attributes from matching features in the second layer.
<b>Where the features</b>	PREDICATE	[enumeration] [list] Default: [0]	Type of spatial relation the source feature should have with the target feature so that they could be joined. One or more of: <ul style="list-style-type: none"> <li>• 0 — intersect</li> <li>• 1 — contain</li> <li>• 2 — equal</li> <li>• 3 — touch</li> <li>• 4 — overlap</li> <li>• 5 — are within</li> <li>• 6 — cross</li> </ul> <p>If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.</p>
<b>By comparing to</b>	JOIN	[vector: any]	The join layer. Features of this vector layer will <b>add</b> summaries of their attributes to the source layer attribute table if they satisfy the spatial relationship.
<b>Fields to summarize (leave empty to use all fields)</b> Optional	JOIN_FIELDS	[tablefield: any] [list]	Select the specific fields you want to add from the join layer. By default all the fields are added.

Fortsetzung auf der nächsten Seite

Tab. 28.134 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Summaries to calculate (leave empty to use all fields)</b> Optional	SUMMARIES	[enumeration] [list] Default: []	For each input feature, statistics are calculated on joined fields of their matching features. One or more of: <ul style="list-style-type: none"> <li>• 0 — count</li> <li>• 1 — unique</li> <li>• 2 — min</li> <li>• 3 — max</li> <li>• 4 — range</li> <li>• 5 — sum</li> <li>• 6 — mean</li> <li>• 7 — median</li> <li>• 8 — stddev</li> <li>• 9 — minority</li> <li>• 10 — majority</li> <li>• 11 — q1</li> <li>• 12 — q3</li> <li>• 13 — iqr</li> <li>• 14 — empty</li> <li>• 15 — filled</li> <li>• 16 — min_length</li> <li>• 17 — max_length</li> <li>• 18 — mean_length</li> </ul>
<b>Discard records which could not be joined</b>	DISCARD_NONMAT	[boolean] Default: False	Remove from the output the input layer's features which could not be joined
<b>Joined layer</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the join. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Joined layer</b>	OUTPUT	[same as input]	Output vector layer with summarized attributes from the join

## Pythoncode

**Algorithm ID:** qgis:joinbylocationsummary

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Join attributes by nearest

Takes an input vector layer and creates a new vector layer with additional fields in its attribute table. The additional attributes and their values are taken from a second vector layer. Features are joined by finding the closest features from each layer.

By default only the nearest feature is joined, but the join can also join to the k-nearest neighboring features.

If a maximum distance is specified, only features which are closer than this distance will be matched.

**Siehe auch:**

*Nearest neighbour analysis, Join attributes by field value, Join attributes by location, Distance matrix*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer.
<b>Input layer 2</b>	INPUT_2	[vector: any]	The join layer.
<b>Layer 2 fields to copy (leave empty to copy all fields)</b>	FIELDS_TO_COPY	[fields]	Join layer fields to copy (if empty, all fields will be copied).
<b>Discard records which could not be joined</b>	DISCARD_NONMAT	[boolean] Default: False	Remove from the output the input layer records which could not be joined
<b>Joined field prefix</b>	PREFIX	[string]	Joined field prefix
<b>Maximum nearest neighbors</b>	NEIGHBORS	[number] Default: 1	Maximum number of nearest neighbors
<b>Maximum distance</b>	MAX_DISTANCE	[number]	Maximum search distance
<b>Joined layer</b> Optional	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the vector layer containing the joined features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 28.135 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Unjoinable features from first layer</b>	NON_MATCHING	[same as input] Default: [Skip output]	Specify the vector layer containing the features that could not be joined. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Joined layer</b>	OUTPUT	[same as input]	The output joined layer.
<b>Unjoinable features from first layer</b>	NON_MATCHING	[same as input]	Layer containing the features from first layer that could not be joined to any features in the join layer.
<b>Number of joined features from input table</b>	JOINED_COUNT	[number]	Number of features from the input table that have been joined.
<b>Number of unjoinable features from input table</b>	UNJOINABLE_COU	[number]	Number of features from the input table that could not be joined.

## Pythoncode

**Algorithm ID:** native:joinbynearest

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Merge vector layers

Combines multiple vector layers of the **same geometry** type into a single one.

The attribute table of the resulting layer will contain the fields from all input layers. If fields with the same name but different types are found then the exported field will be automatically converted into a string type field. New fields storing the original layer name and source are also added.

If any input layers contain Z or M values, then the output layer will also contain these values. Similarly, if any of the input layers are multi-part, the output layer will also be a multi-part layer.

Optionally, the destination coordinate reference system (CRS) for the merged layer can be set. If it is not set, the CRS will be taken from the first input layer. All layers will be reprojected to match this CRS.



**Default menu:** *Vector ► Data Management Tools*

**Siehe auch:**

*Split vector layer*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layers</b>	LAYERS	[vector: any] [list]	The layers that are to be merged into a single layer. Layers should be of the same geometry type.
<b>Destination CRS</b> Optional	CRS	[crs]	Choose the CRS for the output layer. If not specified, the CRS of the first input layer is used.
<b>Merged</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Merged</b>	OUTPUT	[same as input]	Output vector layer containing all the features and attributes from the input layers.

## Pythoncode

**Algorithm ID:** `native:mergevectorlayers`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Order by expression

Sorts a vector layer according to an expression: changes the feature index according to an expression.

Be careful, it might not work as expected with some providers, the order might not be kept every time.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer to sort
<b>Expression</b>	EXPRESSION	[expression]	Expression to use for the sorting
<b>Sort ascending</b>	ASCENDING	[boolean] Default: True	If checked the vector layer will be sorted from small to large values.
<b>Sort nulls first</b>	NULLS_FIRST	[boolean] Default: False	If checked, Null values are placed first
<b>Ordered</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ordered</b>	OUTPUT	[same as input]	Output (sorted) vector layer

### Pythoncode

**Algorithm ID:** native:orderbyexpression

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Repair Shapefile

Repairs a broken ESRI Shapefile dataset by (re)creating the SHX file.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Shapefile</b>	INPUT	[file]	Full path to the ESRI Shapefile dataset with a missing or broken SHX file

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Repaired layer</b>	OUTPUT	[vector: any]	The input vector layer with the SHX file repaired

## Pythoncode

**Algorithm ID:** native:repairshapefile

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Reproject layer

Reprojects a vector layer in a different CRS. The reprojected layer will have the same features and attributes of the input layer.



Allows *features in-place modification* of point, line, and polygon features

### Siehe auch:

*Projektion zuweisen, Define Shapefile projection, Find projection*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer to reproject
<b>Target CRS</b>	TARGET_CRS	[crs] Default: EPSG:4326 – WGS 84	Destination coordinate reference system
<b>Convert curved geometries to straight segments</b> NEW in 3.32 Optional	CONVERT_CURVED	[boolean] Default: False	If checked, curved geometries will be converted to straight segments in the process, avoiding potential distortion issues.
<b>Reprojected</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Coordinate Operation</b> Optional	OPERATION	[string]	Specific operation to use for a particular reprojection task, instead of always forcing use of the current project's transformation settings. Useful when reprojecting a particular layer and control over the exact transformation pipeline is required. Requires proj version >= 6. Read more at <a href="#">Datum Transformations</a> .

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Reprojected</b>	OUTPUT	[same as input]	Output (reprojected) vector layer

## Pythoncode

**Algorithm ID:** native:reprojectlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Save vector features to file

Saves vector features to a specified file dataset.

For dataset formats supporting layers, an optional layer name parameter can be used to specify a custom string. Optional GDAL-defined dataset and layer options can be specified. For more information on this, read the online [GDAL documentation](#) on the format.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Vector features</b>	INPUT	[vector: any]	Input vector layer.
<b>Saved features</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specify the file to save the features to. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Layer name</b> Optional	LAYER_NAME	[string]	Name to use for the output layer
<b>GDAL dataset options</b> Optional	DATASOURCE_OPT	[string]	GDAL dataset creation options of the output format. Separate individual options with semicolons.
<b>GDAL layer options</b> Optional	LAYER_OPTIONS	[string]	GDAL layer creation options of the output format. Separate individual options with semicolons.
<b>Action to take on pre-existing file</b>	ACTION_ON_EXIS	[enumeration] Default: 0	How to manage existing features. Valid methods are: 0 — Create or overwrite file 1 — Create or overwrite layer 2 — Append features to existing layer, but do not create new fields 3 — Append features to existing layer, and create new fields if needed

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Saved features</b>	OUTPUT	[same as input]	Vector layer with the saved features.
<b>File name and path</b>	FILE_PATH	[string]	Output file name and path.
<b>Layer name</b>	LAYER_NAME	[string]	Name of the layer, if any.

## Pythoncode

**Algorithm ID:** native:savefeatures

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Set layer encoding

Sets the encoding used for reading a layer's attributes. No permanent changes are made to the layer, rather it affects only how the layer is read during the current session.

---

**Bemerkung:** Changing the encoding is only supported for some vector layer data sources.

---

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Saved features</b>	INPUT	[vector: any]	Vector layer to set the encoding.
<b>Encoding</b>	ENCODING	[string]	Text encoding to assign to the layer in the current QGIS session.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output layer</b>	OUTPUT	[same as input]	Input vector layer with the set encoding.

## Pythoncode

**Algorithm ID:** native:setlayerencoding

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Split features by character

Features are split into multiple output features by splitting a field's value at a specified character. For instance, if a layer contains features with multiple comma separated values contained in a single field, this algorithm can be used to split these values up across multiple output features. Geometries and other attributes remain unchanged in the output. Optionally, the separator string can be a regular expression for added flexibility.



Allows *features in-place modification* of point, line, and polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer
<b>Split using values in the field</b>	FIELD	[tablefield: any]	Field to use for splitting
<b>Split value using character</b>	CHAR	[string]	Character to use for splitting
<b>Use regular expression separator</b>	REGEX	[boolean] Default: False	

Fortsetzung auf der nächsten Seite

Tab. 28.137 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Split</b>	OUTPUT	[same as input] Default: Create temporary layer	Specify output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Split</b>	OUTPUT	[same as input]	The output vector layer.

## Pythoncode

**Algorithm ID:** native:splitfeaturesbycharacter

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Split vector layer

Creates a set of vectors in an output folder based on an input layer and an attribute. The output folder will contain as many layers as the unique values found in the desired field.

The number of files generated is equal to the number of different values found for the specified attribute.

It is the opposite operation of *merging*.

**Default menu:** *Vector ► Data Management Tools*

**Siehe auch:**

*Merge vector layers*

## Parameter

## Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer
<b>Unique ID field</b>	FIELD	[tablefield: any]	Field to use for splitting
<b>Output directory</b>	OUTPUT	[folder] Default: [Save to temporary folder]	Specify the directory for the output layers. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Output file type</b> Optional	FILE_TYPE	[enumeration] Default: gpkg in the dialog window	Select the extension of the output files. If not specified or invalid, the output files format will be the one set in the „Default output vector layer extension“ Processing setting.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b>	OUTPUT	[folder]	The directory for the output layers
<b>Output layers</b>	OUTPUT_LAYERS	[same as input] [list]	The output vector layers resulting from the split.

## Pythoncode

**Algorithm ID:** native:splitvectorlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Truncate table

Truncates a layer, by deleting all features from within the layer.

**Warnung:** This algorithm modifies the layer in place, and deleted features cannot be restored!

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input Layer</b>	INPUT	[vector: any]	Input vector layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Truncated layer</b>	OUTPUT	[folder]	The truncated (empty) layer

## Pythoncode

**Algorithm ID:** native:truncatetable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.22 Vector geometry

### Add geometry attributes

Computes geometric properties of the features in a vector layer and includes them in the output layer.

It generates a new vector layer with the same content as the input one, but with additional attributes, containing geometric measurements based on a selected CRS.

The attributes added to the table depend on the geometry type and dimension of the input layer:

- for **point** layers: X (xcoord), Y (ycoord), Z (zcoord) coordinates and/or M value (mvalue)
- for **line** layers: length and, for the LineString and CompoundCurve geometry types, the feature sinuosity and straight distance (straightdis)
- for **polygon** layers: perimeter and area

**Default menu:** *Vector ► Geometry Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Calculate using</b>	CALC_METHOD	[enumeration] Default: 0	Calculation parameters to use for the geometric properties. One of: <ul style="list-style-type: none"> <li>• 0 — Layer CRS</li> <li>• 1 — Project CRS</li> <li>• 2 — Ellipsoidal</li> </ul>
<b>Added geom info</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (input copy with geometry) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Added geom info</b>	OUTPUT	[same as input]	Copy of the input vector layer with the addition of the geometry fields

## Pythoncode

**Algorithm ID:** qgis:exportaddgeometrycolumns

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Affine transform

Applies an affine transformation to the layer geometries. Affine transformations can include translation, scaling and rotation. The operations are performed in the following order: scale, rotation, and translation.

Z and M values (if present) can be translated and scaled.

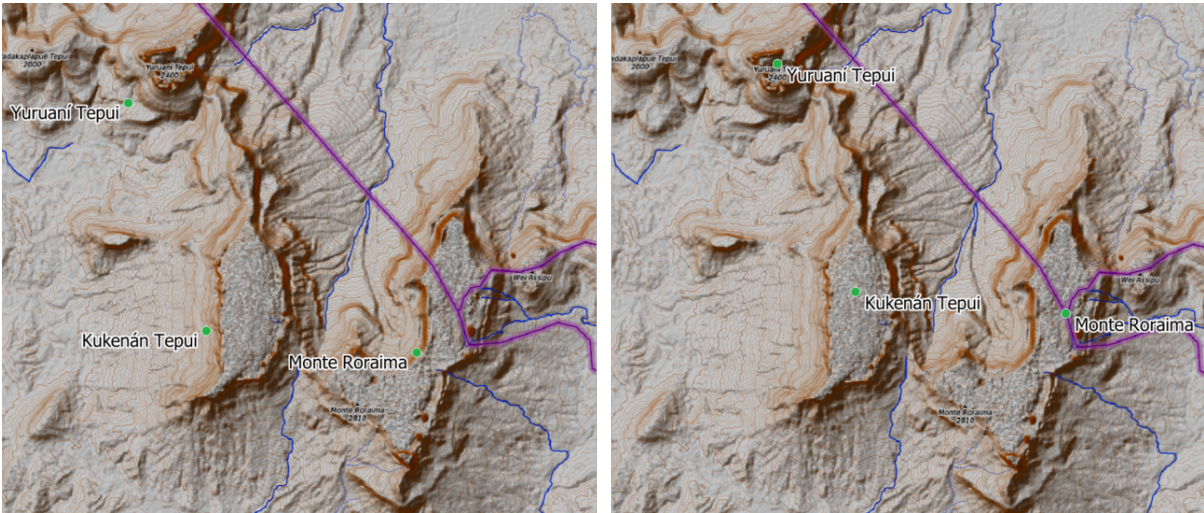









Abb. 28.58: Vector point layer (green dots) before (left), and after (right) an affine transformation (translation).

 Allows *features in-place modification* of point, line, and polygon features

Siehe auch:



[Translate](#)

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Translation axis) (x-	DELTA_X	[number  ] Default: 0	Displacement to apply on the X axis.
Translation axis) (y-	DELTA_Y	[number  ] Default: 0	Displacement to apply on the Y axis.
Translation axis) (z-	DELTA_Z	[number  ] Default: 0	Displacement to apply on the Z axis.
Translation values) (m-	DELTA_M	[number  ] Default: 0	Offset to apply on m values.
Scale factor axis) (x-	SCALE_X	[number  ] Default: 1	Scaling value (expansion or contraction) to apply on the X axis.
Scale factor axis) (y-	SCALE_Y	[number  ] Default: 1	Scaling value (expansion or contraction) to apply on the Y axis.
Scale factor axis) (z-	SCALE_Z	[number  ] Default: 1	Scaling value (expansion or contraction) to apply on the Z axis.

Fortsetzung auf der nächsten Seite

Tab. 28.140 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Scale factor (m-values)</b>	SCALE_M	[number  ] Default: 1	Scaling value (expansion or contraction) to apply on m values.
<b>Rotation around z-axis (degrees counter-clockwise)</b>	ROTATION_Z	[number  ] Default: 0	Angle of the rotation in degrees.
<b>Transformed</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Transformed</b>	OUTPUT	[same as input]	Output (transformed) vector layer.

## Pythoncode

**Algorithm ID:** native:affinetransform

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Aggregate

Takes a vector or table layer and creates a new layer by aggregating features based on a `group by` expression.

Features for which `group by` expression returns the same value are grouped together.

It is possible to group all source features together using constant value in `group by` parameter, example: NULL.

It is also possible to group features by multiple fields using Array function, example: Array(„Field1“, „Field2“).

Geometries (if present) are combined into one multipart geometry for each group. Output attributes are computed depending on each given aggregate definition.

This algorithm allows to use the default *aggregates functions* of the QGIS Expression engine.

**Siehe auch:**






[Collect geometries](#), [Dissolve](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Group by expression</b>	GROUP_BY	[tablefield: any] Default: ‚NULL‘	Choose the grouping field. If <i>NULL</i> all features will be grouped.

Fortsetzung auf der nächsten Seite

Tab. 28.142 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Aggregates</b>	AGGREGATES	[list]	<p>List of output layer field definitions. Example of a field definition:  <pre>{,aggregate: 'sum', ,delimiter: ', ', ,input: ' \$area', ,length: 10, ,name: 'totarea', ,precision: 0, ,type: 6}</pre> By default, the list contains all the fields of the input layer. In the GUI, you can edit these fields and their definitions, and you can also:</p> <ul style="list-style-type: none"> <li>• Click the  button to add a new field.</li> <li>• Click  to delete the selected field.</li> <li>• Use  and  to change order of the fields.</li> <li>• Click  to reset to the default (the fields of the input layer).</li> </ul> <p>For each of the fields you'd like to retrieve information from, you need to define the following:</p> <p><b>Input expression [expression] (input)</b>  Field or expression from the input layer.</p> <p><b>Aggregate function [enumeration] (aggregate)</b>  <i>Function</i> to use on the input expression to return the aggregated value.  Default: <i>concatenate</i> (for string data type), <i>sum</i> (for numeric data type)</p> <p><b>Delimiter [string] (delimiter)</b>  Text string to separate aggregated values, for example in case of concatenation.  Default: ,</p> <p><b>Output field name [string] (name)</b>  Name of the aggregated field in the output layer. By default input field name is kept.</p> <p><b>Type [enumeration] (type)</b>  Data type of the output field. One of:</p> <ul style="list-style-type: none"> <li>• 1 — Boolean</li> <li>• 2 — Integer</li> <li>• 4 — Integer64</li> <li>• 6 — Double</li> <li>• 10 — String</li> <li>• 14 — Date</li> <li>• 16 — DateTime</li> </ul> <p><b>Length [number] (length)</b>  Length of the output field.</p> <p><b>Precision [number] (precision)</b>  Precision of the output field.</p>

Fortsetzung auf der nächsten Seite

Tab. 28.142 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Load fields from layer</b>	GUI only	[vector: any]	You can load fields from another layer and use them for the aggregation
<b>Aggregated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (aggregate) layer One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aggregated</b>	OUTPUT	[same as input]	Multigeometry vector layer with the aggregated values

## Pythoncode

**Algorithm ID:** native:aggregate

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Boundary

Returns the closure of the combinatorial boundary of the input geometries (i.e. the topological boundary of the geometry).

Only for polygon and line layers.

For **polygon geometries**, the boundary consists of all the lines making up the rings of the polygon.

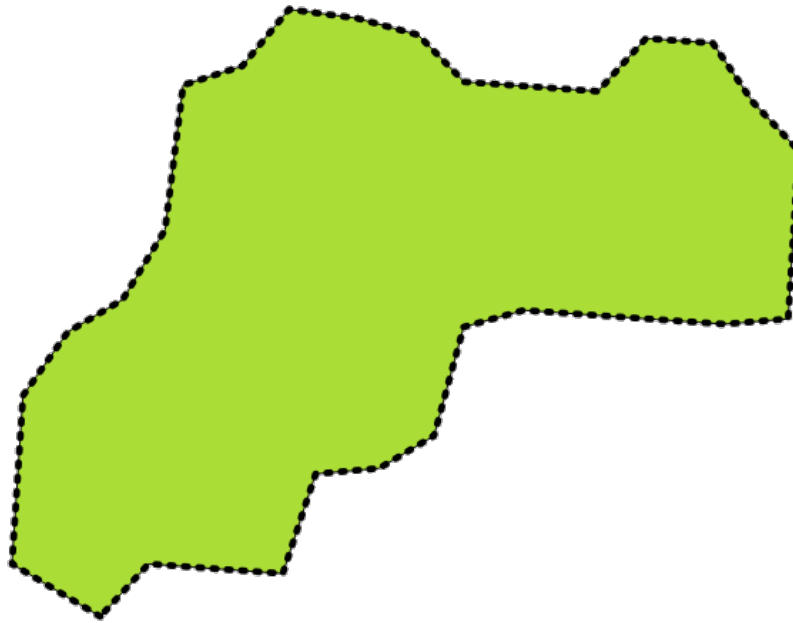


Abb. 28.59: Boundaries (black dashed line) of the source polygon layer

For **lines geometries**, the boundaries are their end points.

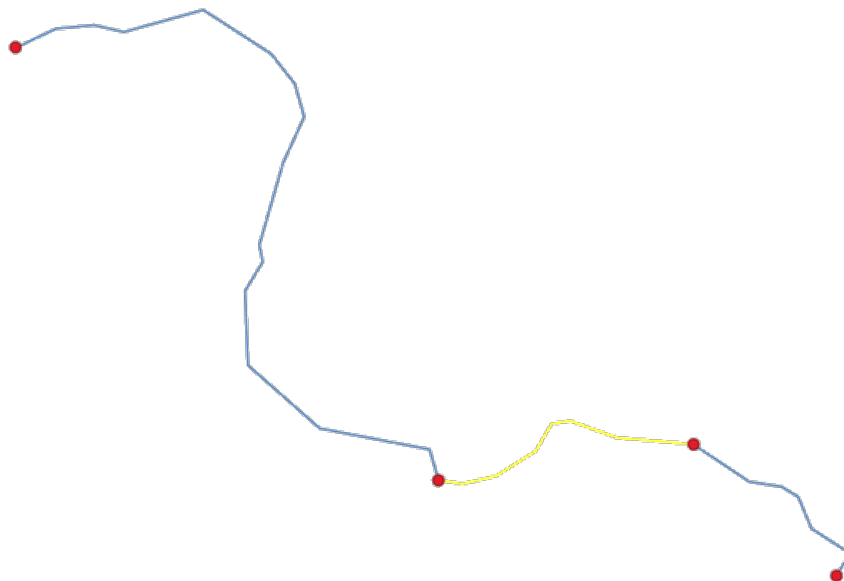


Abb. 28.60: Boundary layer (red points) for lines. In yellow a selected feature.



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Boundary</b>	OUTPUT	[vector: point, line] Default: [Create temporary layer]	Specify the output (boundary) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Boundary</b>	OUTPUT	[vector: point, line]	Boundaries from the input layer (point for line, and line for polygon)

## Pythoncode

**Algorithm ID:** native:boundary

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Bounding boxes

Calculates the bounding box (envelope) of each feature in an input layer. Polygon and line geometries are supported.

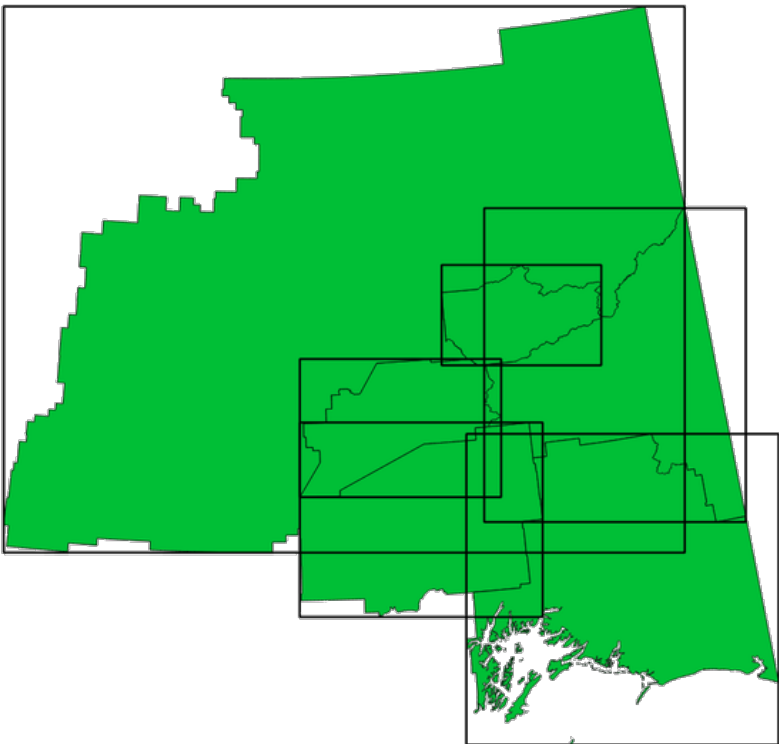


Abb. 28.61: Black lines represent the bounding boxes of each polygon feature

☒ Allows *features in-place modification* of polygon features

Siehe auch:

*Minimum bounding geometry*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line, polygon]	Input line or polygon vector layer
Bounds	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output (bounding box) layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Bounds</b>	OUTPUT	[vector: polygon]	Bounding boxes of input layer. Other than the input attributes, the output layer also contains following fields: width, height, area and perimeter of the generated polygon.

## Pythoncode

**Algorithm ID:** native:boundingboxes

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Buffer

Computes a buffer area for all the features in an input layer, using a fixed or data defined distance.

It is possible to use a negative distance for polygon input layers. In this case the buffer will result in a smaller polygon (setback).

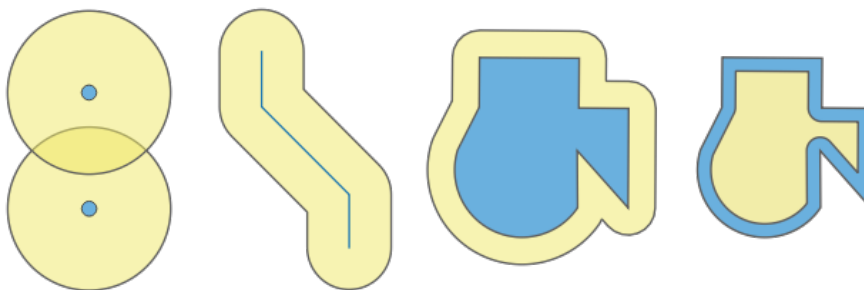


Abb. 28.62: Buffer (in yellow) of points, line, polygon with positive buffer, and polygon with negative buffer

☒ Allows *features in-place modification* of polygon features

**Default menu:** Vector ► Geoprocessing Tools

**Siehe auch:**

*Variable distance buffer, Multi-ring buffer (constant distance), Variable width buffer (by M value)*

## Parameter

### Basic parameters


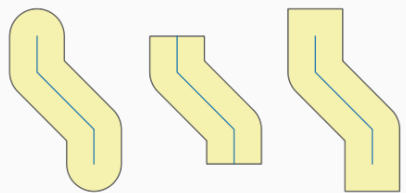

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Distance</b>	DISTANCE	[number  ] Default: 10.0	Buffer distance (from the boundary of each feature). You can use the Data Defined button on the right to choose a field from which the radius will be calculated. This way you can have different radius for each feature (see <a href="#">Variable distance buffer</a> ).
<b>Segments</b>	SEGMENTS	[number] Default: 5	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
<b>End cap style</b>	END_CAP_STYLE	[enumeration] Default: 0	Controls how line endings are handled in the buffer. One of: <ul style="list-style-type: none"> <li>• 0 — Round</li> <li>• 1 — Flat</li> <li>• 2 — Square</li> </ul> 
<b>Join style</b>	JOIN_STYLE	[enumeration] Default: 0	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: <ul style="list-style-type: none"> <li>• 0 — Round</li> <li>• 1 — Miter</li> <li>• 2 — Bevel</li> </ul> 

Abb. 28.63: Round, flat and square cap styles

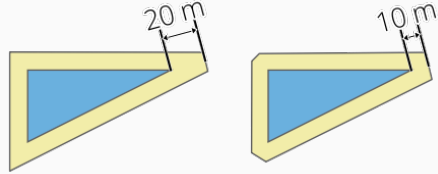
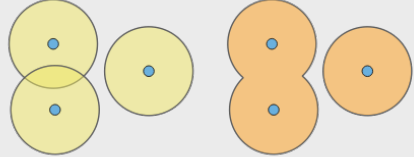
Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are:

- 0 — Round
- 1 — Miter
- 2 — Bevel

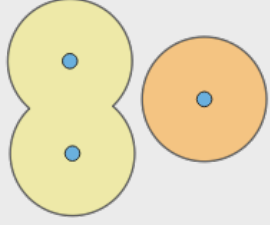
Abb. 28.64: Round, miter, and bevel join styles

Fortsetzung auf der nächsten Seite

Tab. 28.143 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Miter limit</b>	MITER_LIMIT	[number] Default: 2.0	Sets the maximum distance from the offset geometry to use when creating a mitered join as a factor of the offset distance (only applicable for miter join styles). Minimum: 1.0
			 <p>Abb. 28.65: A 10m buffer with a limit of 2 and a 10m buffer with a limit of 1</p>
<b>Dissolve result</b>	DISSOLVE	[boolean] Default: False	Dissolve the final buffer. If True (checked), overlapping buffers will be dissolved (combined) into a single multipart feature.
			 <p>Abb. 28.66: Standard (three single part features - left), dissolved (1 multipart feature with 2 parts - right)</p>
<b>Buffered</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output (buffer) layer. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> <li>Append to Layer...</li> </ul> The file encoding can also be changed here.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Keep disjoint features separate</b>	SEPARATE_DISJO	[boolean] Default: False	If True (checked) and dissolved is checked, features that do not overlap or touch will be exported as separate features (instead of parts of a multipart feature).
			 <p>Abb. 28.67: Results in 2 single part features</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffered</b>	OUTPUT	[vector: polygon]	Output (buffer) polygon layer

## Pythoncode

**Algorithm ID:** native:buffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Centroids

Creates a new point layer, with points representing the centroids of the geometries of the input layer.

The centroid is a single point representing the barycenter (of all parts) of the feature, so it can be outside the feature borders. But can also be a point on each part of the feature.

The attributes of the points in the output layer are the same as for the original features.

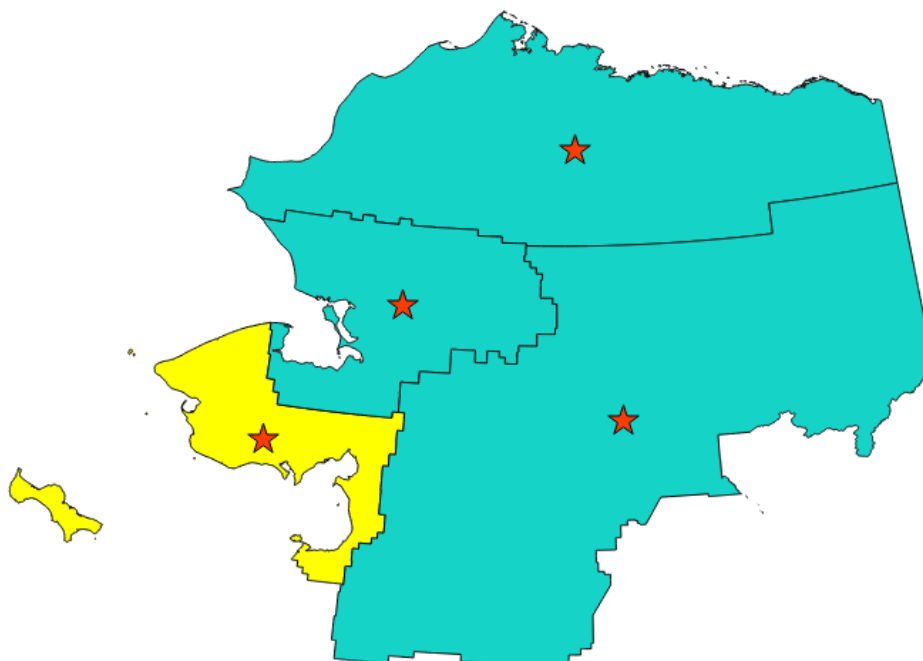


Abb. 28.68: The red stars represent the centroids of the features of the input layer.




Allows *features in-place modification* of point features

**Default menu:** Vector ► Geometry Tools

**Siehe auch:**

[Point on Surface](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Create centroid for each part</b>	ALL_PARTS	[boolean  ] Default: False	If True (checked), a centroid will be created for each part of the geometry
<b>Centroids</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output (centroid) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Centroids</b>	OUTPUT	[vector: point]	Output point vector layer (centroids)

## Pythoncode

**Algorithm ID:** native:centroids

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Check validity

Performs a validity check on the geometries of a vector layer.

The geometries are classified in three groups (valid, invalid and error) and for each group, a vector layer with its features is generated:

- The **Valid output** layer contains only the valid features (without topological errors).
- The **Invalid output** layer contains all the invalid features found by the algorithm.
- The **Error output** layer is a point layer that points to where the invalid features were found.

The attribute tables of the generated layers will contain some additional information („message“ for the **error** layer, „FID“ and „\_errors“ for the **invalid** layer and only „FID“ for the **valid** layer):

The attribute table of each generated vector layer will contain some additional information (number of errors found and types of error):

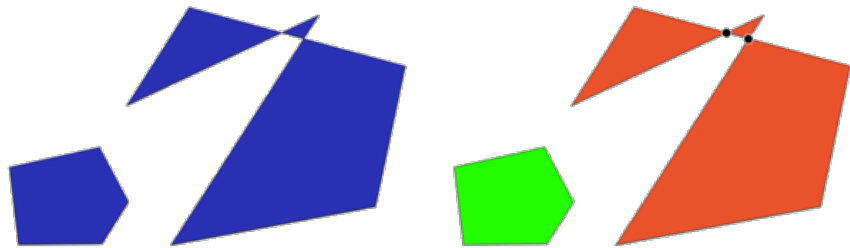


Abb. 28.69: Left: the input layer. Right: the valid layer (green), the invalid layer (orange)

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Fix geometries* and the core plugin *Geometry Checker Plugin*

**Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT_LAYER	[vector: any]	Input vector layer
<b>Method</b>	METHOD	[enumeration] Default: 2	Method to use to check validity. Options: <ul style="list-style-type: none"><li>• 0: The one selected in digitizing settings</li><li>• 1: QGIS</li><li>• 2: GEOS</li></ul>
<b>Ignore ring self intersection</b>	IGNORE_RING_SE	[boolean] Default: False	Ignore self intersecting rings when checking for validity.
<b>Valid output</b>	VALID_OUTPUT	[same as input] Default: [Create temporary layer]	Specify the vector layer to contain a copy of the valid features of the source layer. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.
<b>Invalid output</b>	INVALID_OUTPUT	[same as input] Default: [Create temporary layer]	Vector layer containing copy of the invalid features of the source layer with the field <code>_errors</code> listing the summary of the error(s) found. One of: <ul style="list-style-type: none"><li>• Skip Output</li><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite



Tab. 28.144 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Error output</b>	ERROR_OUTPUT	[vector: point] Default: [Create temporary layer]	Point layer of the exact position of the validity problems detected with the message field describing the error(s) found. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Count of errors</b>	ERROR_COUNT	[number]	The number of geometries that caused errors.
<b>Error output</b>	ERROR_OUTPUT	[vector: point]	Point layer of the exact position of the validity problems detected with the message field describing the error(s) found.
<b>Count of invalid features</b>	INVALID_COUNT	[number]	The number of invalid geometries.
<b>Invalid output</b>	INVALID_OUTPUT	[same as input]	Vector layer containing copy of the invalid features of the source layer with the field <code>_errors</code> listing the summary of the error(s) found.
<b>Count of valid features</b>	VALID_COUNT	[number]	The number of valid geometries.
<b>Valid output</b>	VALID_OUTPUT	[same as input]	Vector layer containing a copy of the valid features of the source layer.

## Pythoncode

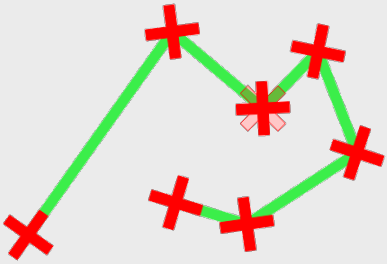
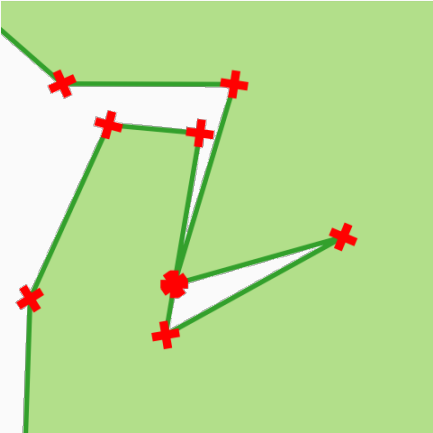
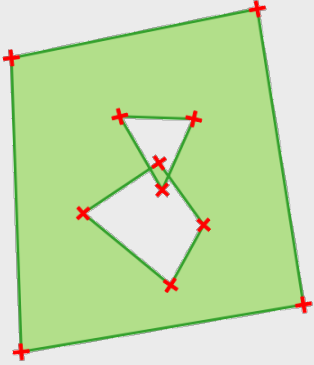
**Algorithm ID:** qgis:checkvalidity

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

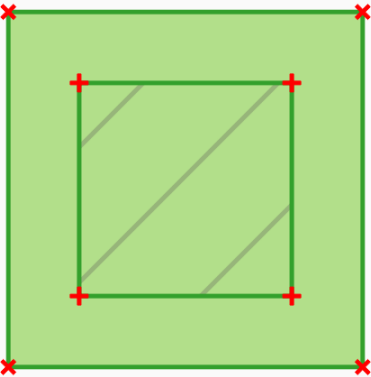
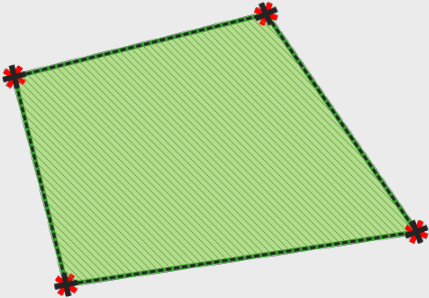
Types of error messages and their meanings

Tab. 28.146: If the GEOS method is used the following error messages can occur:

Error message	Explanation	Example
Repeated point	This error happens when a given vertex is repeated.	
Ring self-intersection	This error happens when a geometry touches itself and generates a ring.	
Self-intersection	This error happens when a geometry touches itself.	
Topology validation error		
Hole lies outside shell		
Holes are nested		
Interior is disconnected		

Fortsetzung auf der nächsten Seite

Tab. 28.146 – Fortsetzung der vorherigen Seite

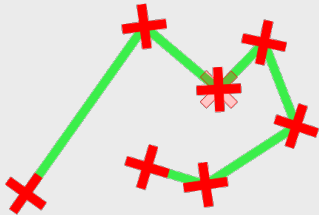
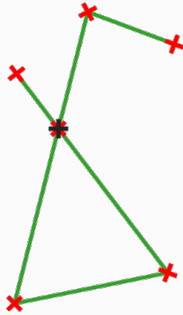
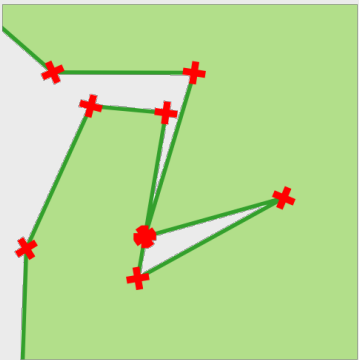
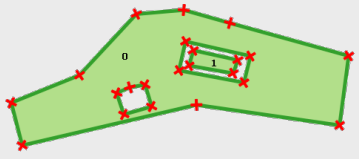
Error message	Explanation	Example
Nested shells	This error happens when a polygon geometry is on top of another polygon geometry.	
Duplicate rings	This error happens when two rings (exterior or interior) of a polygon geometry are identical	
Too few points in geometry component		
Invalid coordinate	For a point geometry, this error happens when the geometry does not have a proper coordinate pair. The coordinate pair does not contain a latitude value and a longitude value in that order.	
Ring is not closed		

Tab. 28.147: If the QGIS method is used the following error messages can occur:

Error message	Explanation	Example
Segment %1 of ring %2 of polygon %3 intersects segment %4 of ring %5 of polygon %6 at %7		
Ring %1 with less than four points		
Ring %1 not closed		
Line %1 with less than two points		

Fortsetzung auf der nächsten Seite

Tab. 28.147 – Fortsetzung der vorherigen Seite

Error message	Explanation	Example
Line %1 contains %n duplicate node(s) at %2	This error happens when consecutive points on a line have the same coordinates.	
Segments %1 and %2 of line %3 intersect at %4	This error happens when a line self intersects (two segments of the line intersect each other).	
Ring self-intersection	This error happens when an outer or inner (island) ring / boundary of a polygon geometry intersects itself.	
Ring %1 of polygon %2 not in exterior ring		
Polygon %1 lies inside polygon %2	This error happens when a part of a MultiPolygon geometry is inside a hole of a MultiPolygon geometry.	

## Collect geometries

Takes a vector layer and collects its geometries into new multipart geometries.

One or more attributes can be specified to collect only geometries belonging to the same class (having the same value for the specified attributes), alternatively all geometries can be collected.

All output geometries will be converted to multi geometries, even those with just a single part. This algorithm does not dissolve overlapping geometries - they will be collected together without modifying the shape of each geometry part.

See the ‚Promote to multipart‘ or ‚Aggregate‘ algorithms for alternative options.

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Aggregate, Promote to multipart, Dissolve*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Unique ID fields</b>	FIELD	[tablefield: any] [list]	Choose one or more attributes to collect the geometries
<b>Collected</b>	OUTPUT	[same as input]	Vector layer with collected geometries

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Collected</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the collected geometries. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Pythoncode

**Algorithm ID:** `native:collect`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Concave hull

Computes the concave hull of the features from an input point layer.

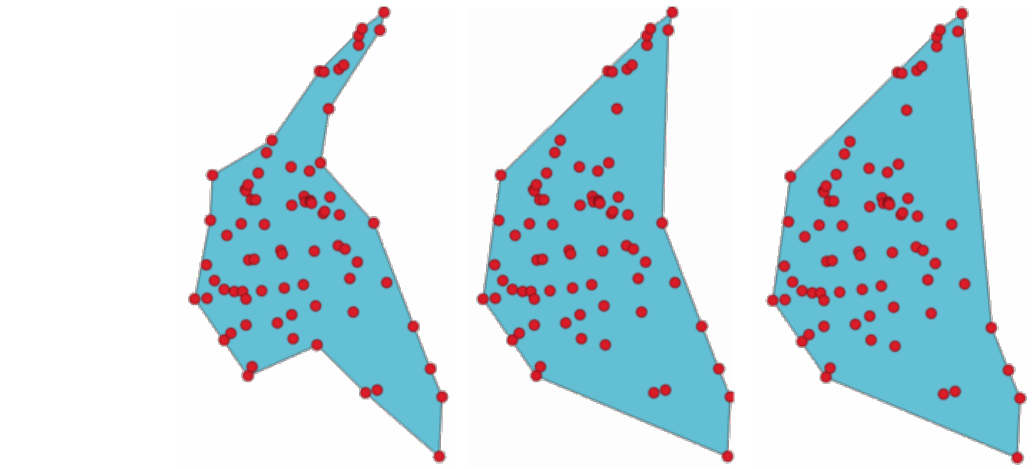


Abb. 28.70: Concave hulls with different thresholds (0.3, 0.6, 0.9)

Siehe auch:

*Convex hull*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input point layer	INPUT	[vector: point]	Input point vector layer
Threshold	ALPHA	[number] Default: 0.3	Number from 0 (maximum concave hull) to 1 (convex hull).
Allow holes	HOLES	[boolean] Default: True	Choose whether to allow holes in the final concave hull
Split multipart geometry into singlepart geometries	NO_MULTIGEOMET	[boolean] Default: False	Check if you want to have singlepart geometries instead of multipart ones.
Concave hull	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Concave hull</b>	OUTPUT	[vector: polygon]	The output vector layer

## Pythoncode

**Algorithm ID:** native:concavehull

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert geometry type

Generates a new layer based on an existing one, with a different type of geometry.

The attribute table of the output layer is the same as the one of the input layer.

Not all conversions are possible. For instance, a line layer can be converted to a point layer, but a point layer cannot be converted to a line layer.

**Siehe auch:**

*Polygonize, Lines to polygons, Polygons to lines, Points to path*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>New geometry type</b>	TYPE	[enumeration] Default: 0	Geometry type to apply to the output features. One of: <ul style="list-style-type: none"> <li>• 0 — Centroids</li> <li>• 1 — Nodes</li> <li>• 2 — Linestrings</li> <li>• 3 — Multilinestrings</li> <li>• 4 — Polygons</li> </ul>
<b>Converted</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Converted</b>	OUTPUT	[vector: any]	Output vector layer - the type depends on the parameters

## Pythoncode

**Algorithm ID:** qgis:convertgeometrytype

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convert to curved geometries

Converts a geometry into its curved geometry equivalent.

Already curved geometries will be retained without change.



Allows *features in-place modification* of line and polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line or polygon]	Input vector layer
<b>Maximum distance tolerance</b>	DISTANCE	[number] Default: 0.000001	The maximum distance allowed between the original location of vertices and where they would fall on the converted curved geometries
<b>Maximum angle tolerance</b>	ANGLE	[number] Default: 0.000001	Segments are considered as suitable for replacing with an arc if the points are all regularly spaced on the candidate arc. This parameter specifies the maximum angular deviation (in degrees) allowed when testing for regular point spacing. Between 0 and 45°.
<b>Curves</b>	OUTPUT	[vector: compoundcurve or curvepolygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> <li>Append to Layer...</li> </ul> The file encoding can also be changed here.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Curves</b>	OUTPUT	[vector: compoundcurve or curvepolygon]	Output vector layer with curved geometries

## Pythoncode

**Algorithm ID:** native:converttocurves

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Convex hull

Calculates the convex hull for each feature in an input layer.

See the 'Minimum bounding geometry' algorithm for a convex hull calculation which covers the whole layer or grouped subsets of features.

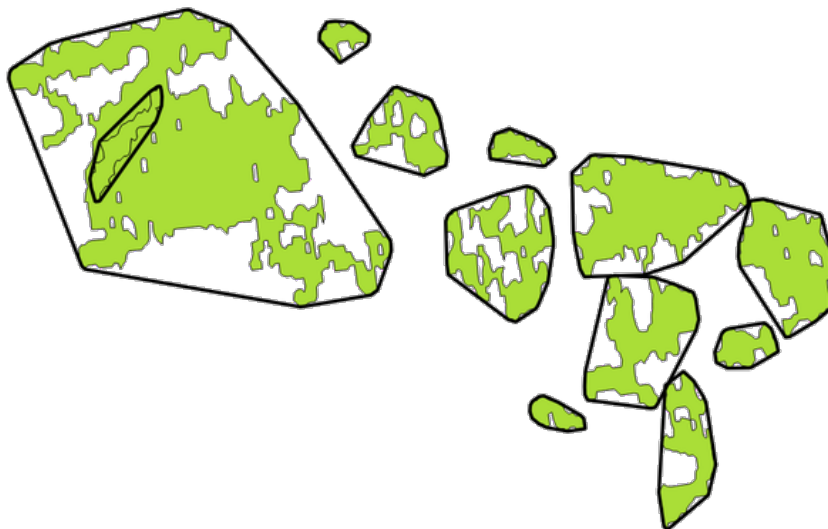


Abb. 28.71: Black lines identify the convex hull for each layer feature

 Allows *features in-place modification* of polygon features

**Default menu:** *Vector ► Geoprocessing Tools*

**Siehe auch:**

*Minimum bounding geometry, Concave hull*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Convex hull</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Convex hull</b>	OUTPUT	[vector: polygon]	The output (convex hull) vector layer

## Pythoncode

**Algorithm ID:** native:convexhull

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Create layer from extent

Creates a new vector layer that contains a single feature with geometry matching the extent of the input layer.

It can be used in models to convert a literal extent (xmin, xmax, ymin, ymax format) into a layer which can be used for other algorithms which require a layer based input.

### Siehe auch:

[Create layer from point](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b> <b>xmax,</b> <b>ymin,</b> <b>ymax)</b>	(xmin, ymin,	INPUT [extent]	Input extent Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Extent</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extent</b>	OUTPUT	[vector: polygon]	The output (extent) vector layer

## Pythoncode

**Algorithm ID:** native:extenttolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Create layer from point

Creates a new vector layer that contains a single feature with geometry matching a point parameter. It can be used in models to convert a point into a point layer for algorithms which require a layer based input.

### Siehe auch:

*Create layer from extent*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Point</b>	INPUT	[coordinates]	Input point, including CRS info (example: 397254, 6214446 [EPSG:32632]). If the CRS is not provided, the Project CRS will be used. The point can be specified by clicking on the map canvas.
<b>Point</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Point</b>	OUTPUT	[vector: point]	The output point vector layer containing the input point.

## Pythoncode

**Algorithm ID:** native:pointtolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Create wedge buffers

Creates wedge shaped buffers from input points.

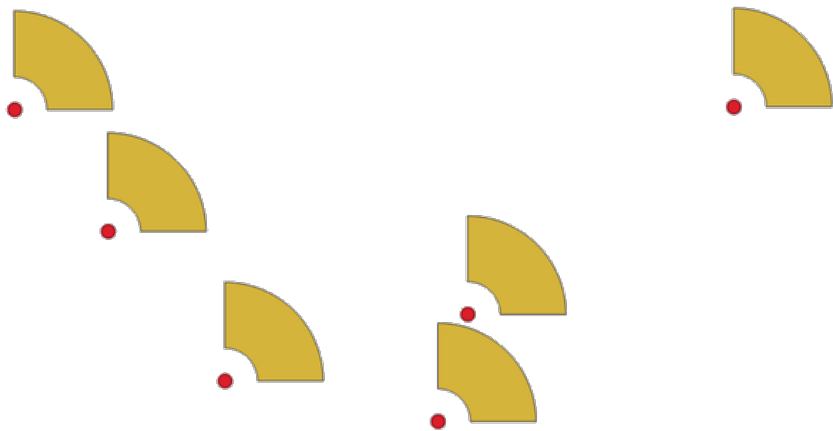



Abb. 28.72: Wedge buffers

The native output from this algorithm are CurvePolygon geometries, but these may be automatically segmentedized to Polygons depending on the output format.

Siehe auch:


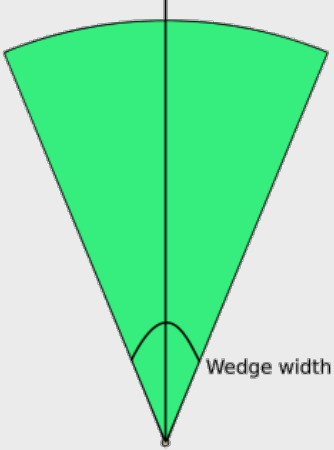


*Buffer, Variable width buffer (by M value), Tapered buffers*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: point]	Input point vector layer
Azimuth (degrees from North)	AZIMUTH	[number  ] Default: 0.0	Angle (in degrees) as the middle value of the wedge

Fortsetzung auf der nächsten Seite

Tab. 28.150 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Wedge width (in degrees)</b>	WIDTH	[number  ] Default: 45.0	Width (in degrees) of the buffer. The wedge will extend to half of the angular width either side of the azimuth direction.  Azimuth = 0° Wedge width = 45°  Abb. 28.73: Azimuth and width values of the wedge buffer
<b>Outer radius</b>	OUTER_RADIUS	[number  ] Default: 1.0	The outer <i>size</i> (length) of the wedge: the size is meant from the source point to the edge of the wedge shape.
<b>Inner radius</b> Optional	INNER_RADIUS	[number  ] Default: 0.0	Inner radius value. If 0 the wedge will begin from the source point.
<b>Buffers</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffers</b>	OUTPUT	[vector: polygon]	The output (wedge buffer) vector layer

## Pythoncode

**Algorithm ID:** native:wedgebuffers

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Delaunay triangulation

Creates a polygon layer with the Delaunay triangulation corresponding to the input point layer.

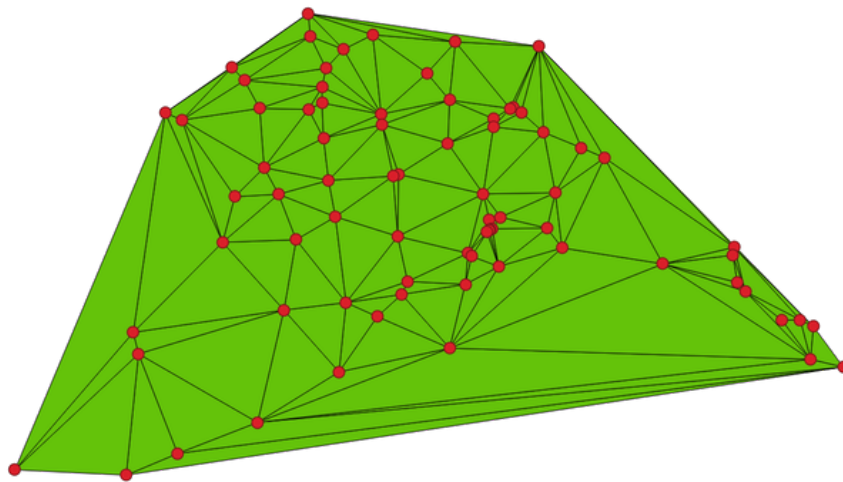


Abb. 28.74: Delaunay triangulation on points

**Default menu:** *Vector ► Geometry Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Tolerance</b> Optional	TOLERANCE	[number] Default: 0.0	Specifies an optional snapping tolerance which can be used to improve the robustness of the triangulation.
<b>Add point IDs to output</b>	ADD_ATTRIBUTES	[boolean] Default: True	Specifies whether fields storing involved point features ID should be added to the output. If False, an id field is used to identify the polygons.
<b>Delaunay triangulation</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Delaunay triangulation</b>	OUTPUT	[vector: polygon]	The output (Delaunay triangulation) vector layer

## Pythoncode

**Algorithm ID:** native:delaunaytriangulation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Delete holes

Takes a polygon layer and removes holes in polygons. It creates a new vector layer in which polygons with holes have been replaced by polygons with only their external ring. Attributes are not modified.

An optional minimum area parameter allows removing only holes which are smaller than a specified area threshold. Leaving this parameter at 0.0 results in all holes being removed.

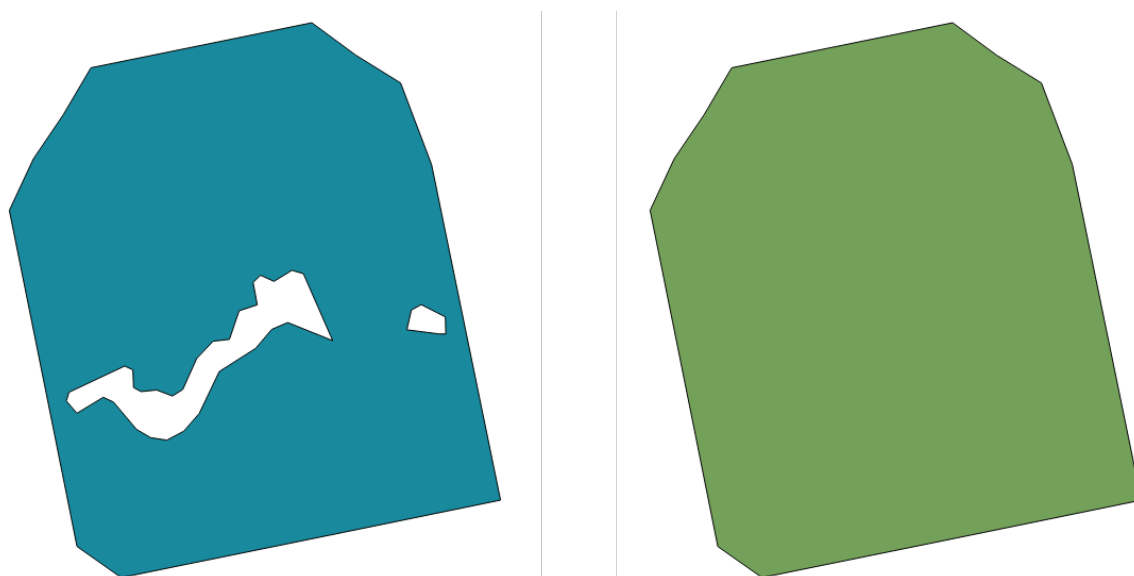



Abb. 28.75: Before and after the cleaning



Allows *features in-place modification* of polygon features



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Remove holes with area less than</b> Optional	MIN_AREA	[number  ] Default: 0.0	Only holes with an area less than this threshold will be deleted. With a value of 0.0, <b>all</b> the holes will be deleted.
<b>Cleaned</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Cleaned</b>	OUTPUT	[same as input]	The output (cleaned) vector layer

## Pythoncode

**Algorithm ID:** native:deleteholes

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Densify by count

Takes a polygon or line layer and generates a new one in which the geometries have a larger number of vertices than the original one.

If the geometries have Z or M values present then these will be linearly interpolated at the added vertices.

The number of new vertices to add to each segment is specified as an input parameter.

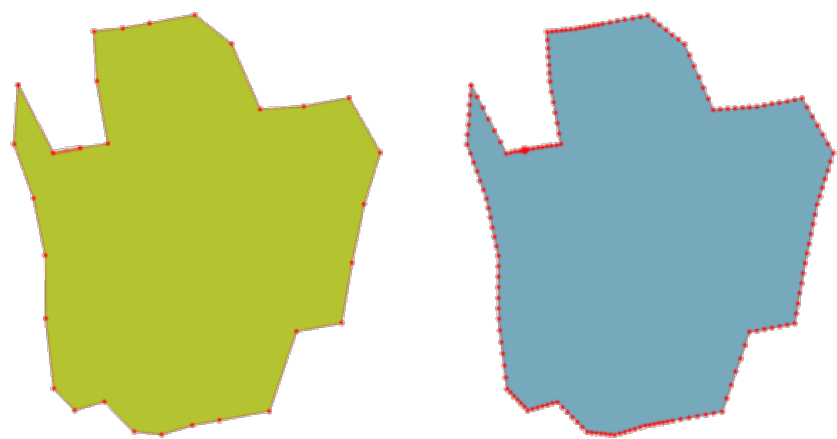


Abb. 28.76: Red points show the vertices before and after the densify

 Allows *features in-place modification* of line and polygon features

Default menu: *Vector ► Geometry Tools*

Siehe auch:

*Densify by interval*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line, polygon]	Input line or polygon vector layer
Vertices to add	VERTICES	[number] Default: 1	Number of vertices to add to each segment
Densified	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

Ausgaben

Bezeichnung	Name	Typ	Beschreibung
Densified	OUTPUT	[same as input]	The output (densified) vector layer

## Pythoncode

**Algorithm ID:** native:densifygeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Densify by interval

Takes a polygon or line layer and generates a new one in which the geometries have a larger number of vertices than the original one.

The geometries are densified by adding regularly placed extra vertices inside each segment so that the maximum distance between any two vertices does not exceed the specified distance.

If the geometries have Z or M values present then these will be linearly interpolated at the added vertices.

### Example

Specifying a distance of 3 would cause the segment  $[0 \ 0] \rightarrow [10 \ 0]$  to be converted to  $[0 \ 0] \rightarrow [2.5 \ 0] \rightarrow [5 \ 0] \rightarrow [7.5 \ 0] \rightarrow [10 \ 0]$ , since 3 extra vertices are required on the segment and spacing these at 2.5 increments allows them to be evenly spaced over the segment.

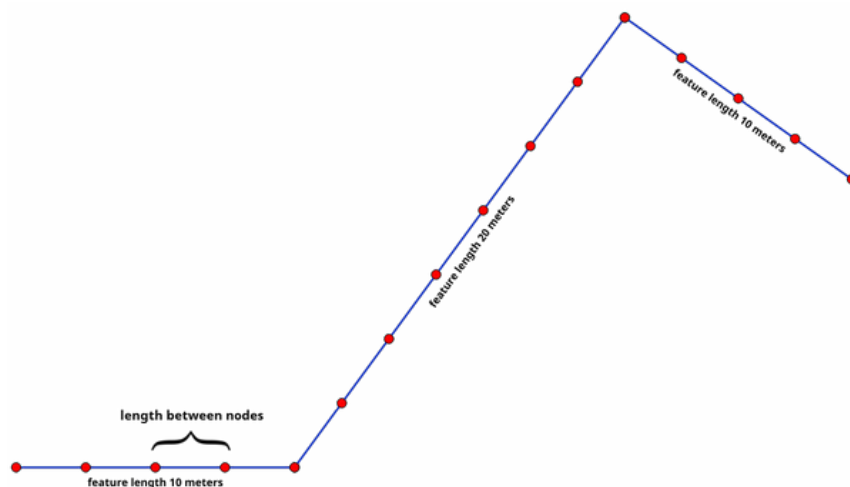


Abb. 28.77: Densify geometry at a given interval




Allows *features in-place modification* of line and polygon features

**Siehe auch:**

*Densify by count*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Interval between vertices to add</b>	INTERVAL	[number  ] Default: 1.0	Maximum distance between two consecutive vertices
<b>Densified</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Densified</b>	OUTPUT	[same as input]	The output (densified) vector layer

## Pythoncode

**Algorithm ID:** native:densifygeometriesgivenaninterval

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Dissolve

Takes a vector layer and combines its features into new features. One or more attributes can be specified to dissolve features belonging to the same class (having the same value for the specified attributes), alternatively all features can be dissolved to a single feature.

All output geometries will be converted to multi geometries. In case the input is a polygon layer, common boundaries of adjacent polygons being dissolved will get erased. If enabled, the optional „Keep disjoint features separate“ setting will cause features and parts that do not overlap or touch to be exported as separate features (instead of parts of a single multipart feature).

The resulting attribute table will have the same fields as the input layer. The values in the output layer's fields are the ones of the first input feature that happens to be processed.

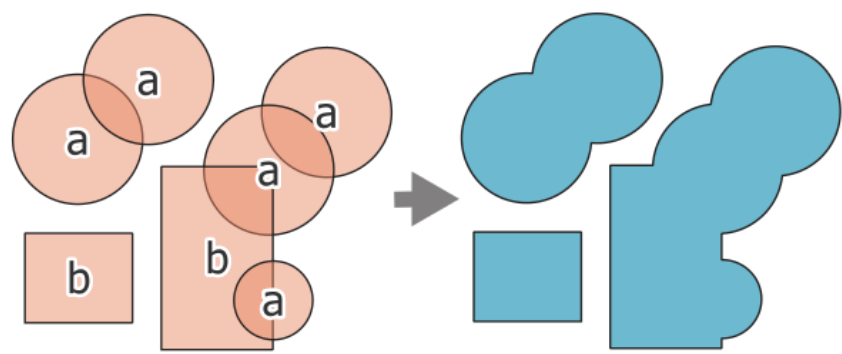


Abb. 28.78: Dissolving a layer into a single multipart feature

Default menu: *Vector ► Geoprocessing Tools*

Siehe auch:

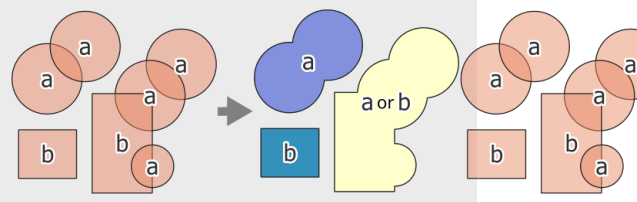
*Aggregate, Collect geometries*

Parameter

Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Dissolve field(s)</b> Optional	FIELD	[tablefield: any] [list] Default: []	Features having the same value for the selected field(s) will be replaced with a single one and their geometries are merged. If no field is provided then all the features are dissolved, resulting in a single (multipart) feature.
Abb. 28.79: Dissolve the polygon layer on a common attribute (2 multipart features)			
<b>Dissolved</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Keep disjoint features separate</b>	SEPARATE_DISJO	[boolean] Default: False	<p>Parts of dissolved features are exported as separate features (instead of parts of a multipart feature).</p>  <p>Abb. 28.80: source (left), dissolve all (3 distinct features - right)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Dissolved</b>	OUTPUT	[same as input]	The output vector layer with dissolved geometries

## Pythoncode

**Algorithm ID:** native:dissolve

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Drape (set Z value from raster)

Uses values sampled from a band within a raster layer to set the Z value for every overlapping vertex in the feature geometry. The raster values can optionally be scaled by a preset amount.




If Z values already exist in the layer, they will be overwritten with the new value. If no Z values exist, the geometry will be upgraded to include the Z dimension.

☒ Allows *features in-place modification* of point, line, and polygon features with Z enabled

**Siehe auch:**

[Set M value from raster](#), [Set Z value](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Raster layer</b>	RASTER	[raster]	Raster layer with Z values
<b>Band number</b>	BAND	[raster band] Default: 1	The raster band to take the Z values from
<b>Value for NoData or non-intersecting vertices</b>	NODATA	[number  Default: 0	Value to use in case the vertex does not intersect (a valid pixel of) the raster
<b>Scale factor</b>	SCALE	[number  Default: 1.0	Scaling value: the band values are multiplied by this value.
<b>Offset</b>	OFFSET	[number  Default: 0.0	Offset value: it is algebraically added to the band values after applying the „Scale factor“.
<b>Updated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with Z values from the raster layer). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Updated</b>	OUTPUT	[same as input]	The output vector layer with Z values from the raster layer

## Pythoncode

**Algorithm ID:** native:setzfromraster

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Drop M/Z values

Removes M (measure) or Z (altitude) values from input geometries.

**Siehe auch:**

*M-Wert einstellen, Set Z value*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer with M or Z values
<b>Drop M Values</b>	DROP_M_VALUES	[boolean] Default: False	Removes the M values from the geometries
<b>Drop Z Values</b>	DROP_Z_VALUES	[boolean] Default: False	Removes the Z values from the geometries
<b>Z/M Dropped</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Z/M Dropped</b>	OUTPUT	[same as input]	The output vector layer (identical to the input layer, except that the M and/or Z dimensions have been removed from the geometries).

### Pythoncode

**Algorithm ID:** native:dropmzvalues

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Eliminate selected polygons

Combines selected polygons of the input layer with certain adjacent polygons by erasing their common boundary. The adjacent polygon can be either the one with the largest or smallest area or the one sharing the largest common boundary with the polygon to be eliminated.

Eliminate is normally used to get rid of sliver polygons, i.e. tiny polygons that are a result of polygon intersection processes where boundaries of the inputs are similar but not identical.

**Default menu:** *Vector ► Geoprocessing Tools*

**Siehe auch:**

*Fix geometries*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Merge selection with the neighboring polygon with the</b>	MODE	[enumeration] Default: None	Choose the parameter to use in order to get rid of the selected polygons: <ul style="list-style-type: none"> <li>• 0 — Largest Area</li> <li>• 1 — Smallest Area</li> <li>• 2 — Largest Common Boundary</li> </ul>
<b>Eliminated</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Eliminated</b>	OUTPUT	[vector: polygon]	The output polygon vector layer.

## Pythoncode

**Algorithm ID:** `qgis:eliminateselectedpolygons`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Explode lines

Takes a lines layer and creates a new one in which each line layer is replaced by a set of lines representing the segments in the original line.

Each line in the resulting layer contains only a start and an end point, with no intermediate vertices between them.



Abb. 28.82: The original line layer and the exploded one

 Allows *features in-place modification* of line features

Siehe auch:

*Subdivide, Line substring*

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Exploded</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Exploded</b>	OUTPUT	[vector: line]	The output line vector layer with features representing each segment of the input layer.

## Pythoncode

**Algorithm ID:** native:explodelines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extend lines

Extends line geometry by a specified amount at the start and end of the line.

Lines are extended using the bearing of the first and last segment in the line.

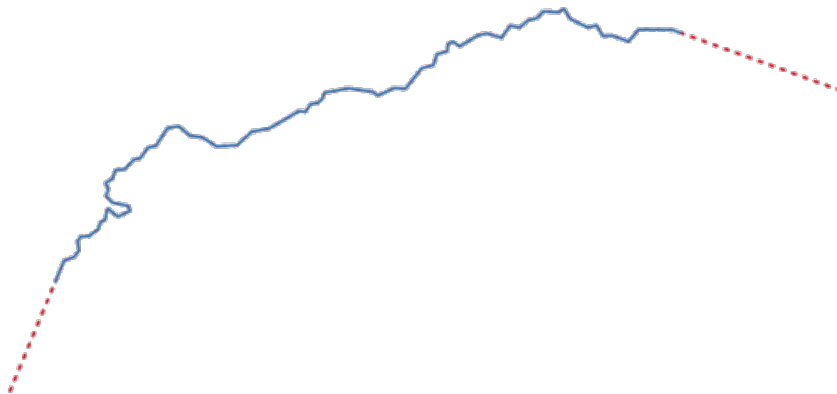




Abb. 28.83: The red dashes represent the initial and final extension of the original layer

☒ Allows *features in-place modification* of line features

**Siehe auch:**

*Line substring*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Start distance</b>	START_DISTANCE	[number  ]	Distance by which to extend the first segment of the line (starting point)
<b>End distance</b>	END_DISTANCE	[number  ]	Distance by which to extend the last segment of the line (ending point)
<b>Extended</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extended</b>	OUTPUT	[vector: line]	The output (extended) line vector layer.

## Pythoncode

**Algorithm ID:** native:extendlines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract M values

Extracts M values from geometries into feature attributes.

By default only the M value from the first vertex of each feature is extracted, however the algorithm can optionally calculate statistics on all of the geometry's M values, including sum, mean, minimum and maximum.

### Siehe auch:

[Extract Z values](#), [M-Wert einstellen](#), [Drop M/Z values](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Summaries to calculate</b>	SUMMARIES	[enumeration] Default: [0]	Statistics on the M values of a geometry. One or more of: <ul style="list-style-type: none"> <li>• 0 — First</li> <li>• 1 — Last</li> <li>• 2 — Count</li> <li>• 3 — Sum</li> <li>• 4 — Mean</li> <li>• 5 — Median</li> <li>• 6 — St.dev (pop)</li> <li>• 7 — Minimum</li> <li>• 8 — Maximum</li> <li>• 9 — Range</li> <li>• 10 — Minority</li> <li>• 11 — Majority</li> <li>• 12 — Variety</li> <li>• 13 — Q1</li> <li>• 14 — Q3</li> <li>• 15 — IQR</li> </ul>
<b>Output column prefix</b>	COLUMN_PREFIX	[string] Default: ,m_‘	The prefix for the output (M) column
<b>Extracted</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted</b>	OUTPUT	[same as input]	The output vector layer (with M values)

## Pythoncode

**Algorithm ID:** native:extractmvalues

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract specific vertices

Takes a vector layer and generates a point layer with points representing specific vertices in the input geometries.

For instance, this algorithm can be used to extract the first or last vertices in the geometry. The attributes associated to each point are the same ones associated to the feature that the vertex belongs to.

The vertex indices parameter accepts a comma separated string specifying the indices of the vertices to extract. The first vertex corresponds to an index of 0, the second vertex has an index of 1, etc. Negative indices can be used to find vertices at the end of the geometry, e.g., an index of -1 corresponds to the last vertex, -2 corresponds to the second last vertex, etc.

Additional fields are added to the vertices indicating the specific vertex position (e.g., 0, -1, etc), the original vertex index, the vertex's part and its index within the part (as well as its ring for polygons), distance along the original geometry and bisector angle of vertex for the original geometry.



Allows *features in-place modification* of point features

**Siehe auch:**

*Extract vertices, Filter vertices by M value, Filter vertices by Z value*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Vertex indices</b>	VERTICES	[string] Default: ,0'	Comma-separated string of the indices of the vertices to extract.
<b>Vertices</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vertices</b>	OUTPUT	[vector: point]	The output (point) vector layer containing the specified vertices from the input layer geometries.

## Pythoncode

**Algorithm ID:** native:extractspecificvertices

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract vertices

Takes a vector layer and generates a point layer with points representing the vertices in the input geometries.

The attributes associated to each point are the same ones associated to the feature that the vertex belongs to.

Additional fields are added to the vertices indicating the vertex index (beginning at 0), the feature's part and its index within the part (as well as its ring for polygons), distance along original geometry and bisector angle of vertex for original geometry.

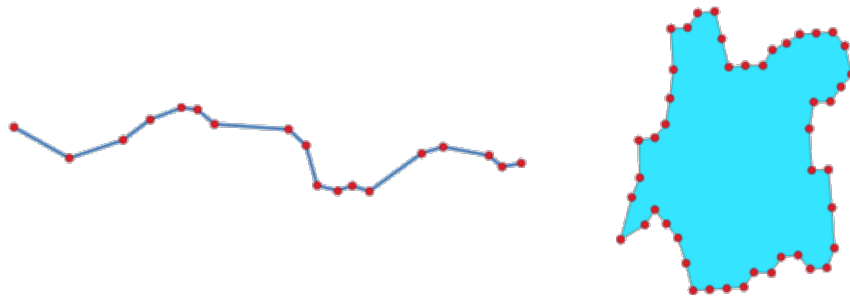


Abb. 28.84: Vertices extracted for line and polygon layer

☒ Allows *features in-place modification* of point features

**Default menu:** Vector ► Geometry Tools

**Siehe auch:**

*Extract specific vertices, Filter vertices by M value, Filter vertices by Z value*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Vertices</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vertices</b>	OUTPUT	[vector: point]	The output (point) vector layer containing the vertices from the input layer geometries.

## Pythoncode

**Algorithm ID:** native:extractvertices

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract Z values

Extracts Z values from geometries into feature attributes.

By default only the Z value from the first vertex of each feature is extracted, however the algorithm can optionally calculate statistics on all of the geometry's Z values, including sum, mean, minimum and maximum.

**Siehe auch:**

*Extract M values, Set Z value, Drop M/Z values*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer

Fortsetzung auf der nächsten Seite



Tab. 28.153 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Summaries to calculate</b>	SUMMARIES	[enumeration] Default: [0]	Statistics on the Z values of a geometry. One or more of: <ul style="list-style-type: none"> <li>• 0 — First</li> <li>• 1 — Last</li> <li>• 2 — Count</li> <li>• 3 — Sum</li> <li>• 4 — Mean</li> <li>• 5 — Median</li> <li>• 6 — St.dev (pop)</li> <li>• 7 — Minimum</li> <li>• 8 — Maximum</li> <li>• 9 — Range</li> <li>• 10 — Minority</li> <li>• 11 — Majority</li> <li>• 12 — Variety</li> <li>• 13 — Q1</li> <li>• 14 — Q3</li> <li>• 15 — IQR</li> </ul>
<b>Output prefix</b>	COLUMN_PREFIX	[string] Default: ,z_‘	The prefix for the output (Z) column
<b>Extracted</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted</b>	OUTPUT	[same as input]	The output vector layer (with Z values)

## Pythoncode

**Algorithm ID:** native:extractzvalues

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Filter vertices by M value

Filters away vertices based on their M value, returning geometries with only vertex points that have a M value greater than or equal to the specified minimum value and/or less than or equal to the maximum value.

If the minimum value is not specified then only the maximum value is tested, and similarly if the maximum value is not specified then only the minimum value is tested.

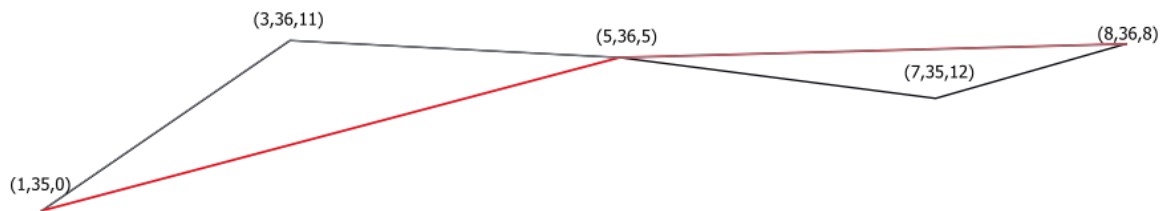


Abb. 28.85: The red line represents the black line with only vertices whose M value is  $\leq 10$ .





Allows *features in-place modification* of line and polygon features with M enabled

**Bemerkung:** Depending on the input geometry attributes and the filters used, the resultant geometries created by this algorithm may no longer be valid.

### Siehe auch:

*Filter vertices by Z value, Extract vertices, Extract specific vertices*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer to remove vertices from
<b>Minimum</b> Optional	MIN	[number  Default: <i>Not set</i>	Minimum of M values allowed
<b>Maximum</b> Optional	MAX	[number  Default: <i>Not set</i>	Maximum of M values allowed
<b>Filtered</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Filtered</b>	OUTPUT	[same as input]	The output vector layer of features with only the filtered vertices.

## Pythoncode

**Algorithm ID:** native:filterverticesbym

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Filter vertices by Z value

Filters away vertices based on their Z value, returning geometries with only vertex points that have a Z value greater than or equal to the specified minimum value and/or less than or equal to the maximum value.

If the minimum value is not specified then only the maximum value is tested, and similarly if the maximum value is not specified then only the minimum value is tested.

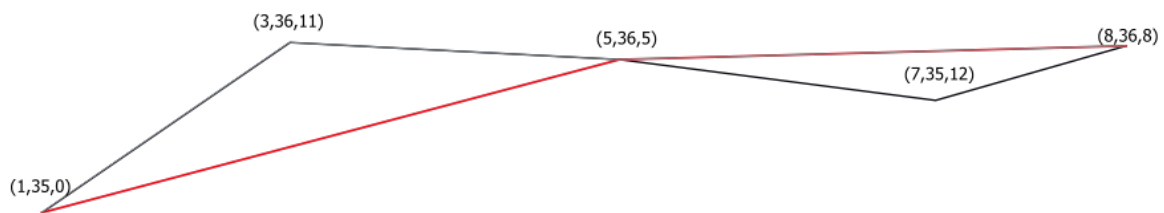


Abb. 28.86: The red line represents the black line with only vertices whose Z value is  $\leq 10$ .





Allows *features in-place modification* of line and polygon features with Z enabled

**Bemerkung:** Depending on the input geometry attributes and the filters used, the resultant geometries created by this algorithm may no longer be valid. You may need to run the [Fix geometries](#) algorithm to ensure their validity.

## Siehe auch:

[Filter vertices by M value](#), [Extract vertices](#), [Extract specific vertices](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer to remove vertices from
<b>Minimum</b> Optional	MIN	[number  ] Default: <i>Not set</i>	Minimum of Z values allowed
<b>Maximum</b> Optional	MAX	[number  ] Default: <i>Not set</i>	Maximum of Z values allowed
<b>Filtered</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Filtered</b>	OUTPUT	[same as input]	The output vector layer of features with only the filtered vertices.

## Pythoncode

**Algorithm ID:** native:filterverticesbyz

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fix geometries

Attempts to create a valid representation of a given invalid geometry without losing any of the input vertices. Already valid geometries are returned without further intervention. Always outputs multi-geometry layer.



Allows *features in-place modification* of point, line, and polygon features without M enabled

**Bemerkung:** M values will be dropped from the output.

**Siehe auch:**

[Check validity](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Repair method</b>	METHOD	[enumeration] Default: 1	Method used to repair the geometries. One of: <ul style="list-style-type: none"> <li>0 — <b>Linework</b>: combines all rings into a set of noded lines and then extracts valid polygons from that linework</li> <li>1 — <b>Structure</b>: first makes all rings valid and then merges shells and subtracts holes from shells to generate valid result. Assumes that holes and shells are correctly categorized. Requires QGIS version built with GEOS 3.10 or later (check <i>Help ► About</i> menu).</li> </ul>
<b>Fixed geometries</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> <li>Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Fixed geometries</b>	OUTPUT	[same as input]	The output vector layer with fixed geometries.

## Pythoncode

**Algorithm ID:** native:fixgeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Force right-hand-rule

Forces polygon geometries to respect the Right-Hand-Rule, in which the area that is bounded by a polygon is to the right of the boundary. In particular, the exterior ring is oriented in a clockwise direction and any interior rings in a counter-clockwise direction.



Allows *features in-place modification* of polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input vector layer
<b>Reoriented</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Reoriented</b>	OUTPUT	[vector: polygon]	The output vector layer with reoriented geometries.

## Pythoncode

**Algorithm ID:** native:forcerhr

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Geodesic line split at antimeridian

Splits a line into multiple geodesic segments, whenever the line crosses the antimeridian ( $\pm 180$  degrees longitude).

Splitting at the antimeridian helps the visual display of the lines in some projections. The returned geometry will always be a multi-part geometry.

Whenever line segments in the input geometry cross the antimeridian, they will be split into two segments, with the latitude of the breakpoint being determined using a geodesic line connecting the points either side of this segment. The current project ellipsoid setting will be used when calculating this breakpoint.

If the input geometry contains M or Z values, these will be linearly interpolated for the new vertices created at the antimeridian.



Allows *features in-place modification* of line features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Split</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Split</b>	OUTPUT	[vector: line]	The output line vector layer split at the antimeridian.

## Pythoncode

**Algorithm ID:** native:antimeridiansplit

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Geometry by expression

Updates existing geometries (or creates new geometries) for input features by use of a QGIS expression.

This allows complex geometry modifications which can utilize all the flexibility of the QGIS expression engine to manipulate and create geometries for output features.

For help with QGIS expression functions, see the inbuilt help available in the [expression builder](#).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer

Fortsetzung auf der nächsten Seite

Tab. 28.154 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output geometry type</b>	OUTPUT_GEOMETR	[enumeration] Default: 0	The output geometry strongly depends on the expression: for instance, if you create a buffer the geometry type has to be polygon. One of: <ul style="list-style-type: none"> <li>• 0 — Polygon</li> <li>• 1 — Line</li> <li>• 2 — Point</li> </ul>
<b>Output geometry has z values</b>	WITH_Z	[boolean] Default: False	Choose if the output geometry should include the Z dimension
<b>Output geometry has m values</b>	WITH_M	[boolean] Default: False	Choose if the output geometry should include the M dimension
<b>Geometry expression</b>	EXPRESSION	[expression] Default: <code>,\$geometry</code>	Add the geometry expression you want to use. You can use the button to open the Expression Dialog. The dialog lists all the relevant expressions, together with their help and guide.
<b>Modified geometry</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Modified geometry</b>	OUTPUT	[vector: any]	The output vector layer

## Pythoncode

**Algorithm ID:** `native:geometrybyexpression`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Interpolate point on line

Creates a point geometry interpolated at a set distance along line or curve geometries.

Z and M values are linearly interpolated from existing values.

If a multipart geometry is encountered, only the first part is considered when calculating the substring.

If the specified distance is greater than the input feature's length, the resultant feature will have a null geometry.




Abb. 28.87: Interpolated point at 500m of the beginning of the line

### Siehe auch:

*Points along geometry*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Distance</b>	DISTANCE	[number  ] Default: 0.0	Distance from the beginning of the line
<b>Interpolated points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated points</b>	OUTPUT	[vector: point]	The output point vector layer with features at a set distance along the line or polygon boundary

Pythoncode

Algorithm ID: native:interpolatepoint

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Keep N biggest parts

Takes a layer with polygons or multipolygons and returns a new layer in which only the *n* largest polygons of each multipolygon feature are kept. If a feature has *n* or fewer parts, the feature will just be copied.



Abb. 28.88: Clockwise from top left: original multipart feature, one, two and three biggest parts kept

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Polygons</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Parts to keep</b>	PARTS	[number] Default: 1	Number of parts to keep. If 1, only the biggest part of the feature will be kept.
<b>Parts</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Parts</b>	OUTPUT	[vector: polygon]	The output polygon vector layer with the N biggest parts of each feature

## Pythoncode

**Algorithm ID:** qgis:keepnbiggestparts

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Line substring

Returns the portion of a line (or curve) which falls between the specified start and end distances (measured from the beginning of the line).

Z and M values are linearly interpolated from existing values.

If a multipart geometry is encountered, only the first part is considered when calculating the substring.





Abb. 28.89: Substring line with starting distance set at 0 meters and the ending distance at 250 meters.

 Allows *features in-place modification* of line features

**Siehe auch:**

[Extend lines](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Start distance</b>	START_DISTANCE	[number  ]	Distance along the input line to the start point of the output feature
<b>End distance</b>	END_DISTANCE	[number  ]	Distance along the input line to the end point of the output feature
<b>Substring</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Substring</b>	OUTPUT	[vector: line]	The output line vector layer.

## Pythoncode

**Algorithm ID:** native:linesubstring

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Lines to polygons

Generates a polygon layer using as polygon rings the lines from an input line layer.

The attribute table of the output layer is the same as the one of the input layer.

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Polygons to lines, Polygonize, Convert geometry type*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Polygons</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Polygons</b>	OUTPUT	[vector: polygon]	The output polygon vector layer.

## Pythoncode

**Algorithm ID:** qgis:linestopolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Merge lines

Joins all connected parts of MultiLineString geometries into single LineString geometries.

If any parts of the input MultiLineString geometries are not connected, the resultant geometry will be a MultiLineString containing any lines which could be merged and any non-connected line parts.



Allows *features in-place modification* of line features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Merged</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Merged</b>	OUTPUT	[vector: line]	The output (merged) line vector layer.

## Pythoncode

**Algorithm ID:** native:mergelines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Minimum bounding geometry

Creates geometries which enclose the features from an input layer. The features can be grouped by a field. The output layer will then contain one feature per group value with a geometry (MBB) that covers the geometries of the features with matching value.

The following enclosing geometry types are supported:

- bounding box (envelope)
- oriented rectangle
- circle
- convex hull

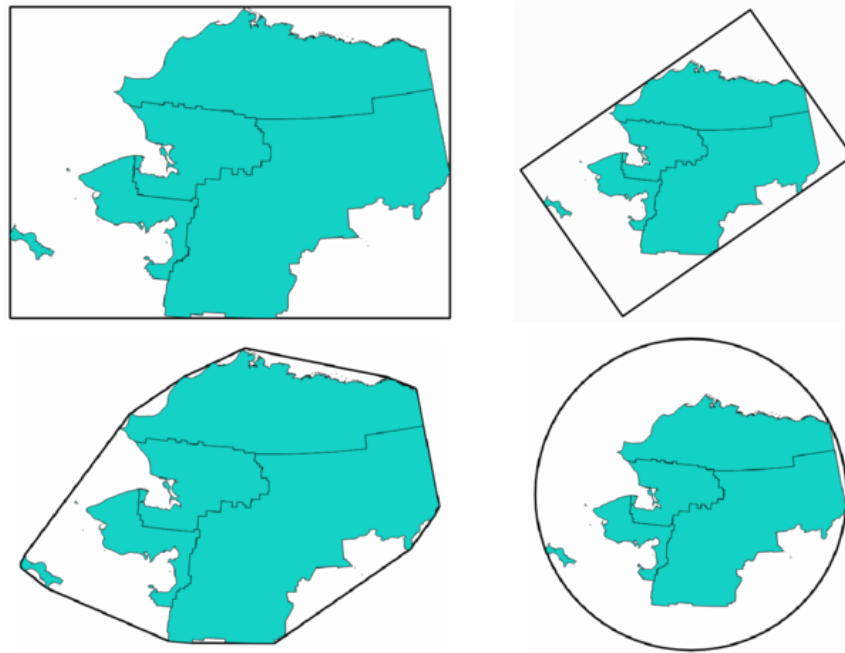


Abb. 28.90: Clockwise from top left: envelope, oriented rectangle, circle, convex hull

#### Siehe auch:

*Minimum enclosing circles*

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Field</b> Optional	FIELD	[tablefield: any]	Features can be grouped by a field. If set, this causes the output layer to contain one feature per grouped value with a minimal geometry covering only the features with matching values.
<b>Geometry type</b>	TYPE	[enumeration] Default: 0	Enclosing geometry types. One of: <ul style="list-style-type: none"> <li>• 0 — Envelope (Bounding Box)</li> <li>• 1 — Minimum Oriented Rectangle</li> <li>• 2 — Minimum Enclosing Circle</li> <li>• 3 — Convex Hull</li> </ul>
<b>Bounding geometry</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Bounding geometry</b>	OUTPUT	[vector: polygon]	The output (bounding) polygon vector layer.

## Pythoncode

**Algorithm ID:** qgis:minimumboundinggeometry

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Minimum enclosing circles

Calculates the minimum enclosing circles of the features in the input layer.

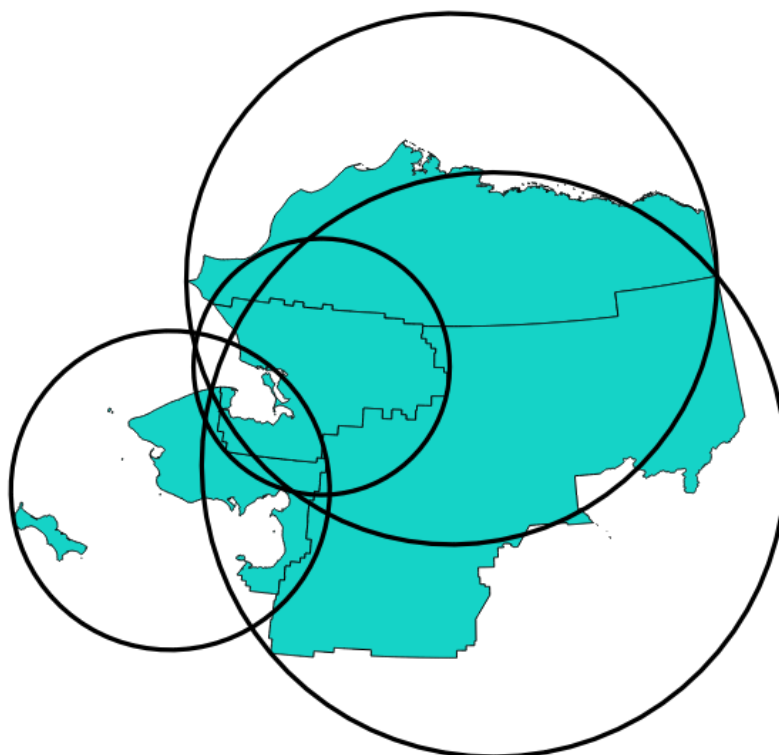


Abb. 28.91: Enclosing circles for each feature

 Allows *features in-place modification* of polygon features

**Siehe auch:**

*Minimum bounding geometry*



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Number of segments in circles</b>	SEGMENTS	[number] Default: 72	The number of segments used to approximate a circle. Minimum 8, maximum 100000.
<b>Minimum enclosing circles</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Minimum enclosing circles</b>	OUTPUT	[vector: polygon]	The output polygon vector layer.

## Pythoncode

**Algorithm ID:** native:minimumentclosingcircle

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Multi-ring buffer (constant distance)

Computes multi-ring (*donut*) buffer for the features of the input layer, using a fixed or dynamic distance and number of rings.

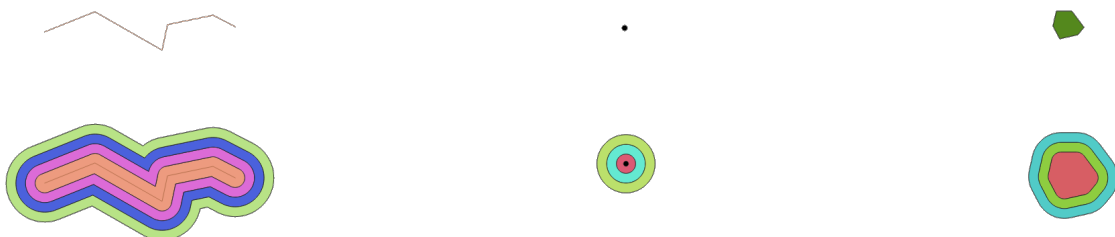




Abb. 28.92: Multi-ring buffer for a line, point and polygon layer

 Allows *features in-place modification* of polygon features

**Siehe auch:**

*Buffer, Variable distance buffer, Rectangles, ovals, diamonds, Single sided buffer*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Number of rings</b>	RINGS	[number  ] Default: 1	The number of rings. It can be a unique value (same number of rings for all the features) or it can be taken from features data (the number of rings depends on feature values).
<b>Distance between rings</b>	DISTANCE	[number  ] Default: 1.0	Distance between the rings. It can be a unique value (same distance for all the features) or it can be taken from features data (the distance depends on feature values).
<b>Multi-ring buffer (constant distance)</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Multi-ring buffer (constant distance)</b>	OUTPUT	[vector: polygon]	The output polygon vector layer.

### Pythoncode

**Algorithm ID:** native:multiringconstantbuffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Multipart to singleparts

Splits multipart features in the input layer into singlepart features.

The attributes of the output layer are the same as the original ones but divided into single features.



Abb. 28.93: Left the multipart source layer and right the single part output result

 Allows *features in-place modification* of point, line, and polygon features

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Collect geometries, Promote to multipart*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Single parts</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Single parts</b>	OUTPUT	[same as input]	The output vector layer.

## Pythoncode

**Algorithm ID:** native:multiparttosingleparts

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Offset lines

Offsets lines by a specified distance. Positive distances will offset lines to the left, and negative distances will offset them to the right.

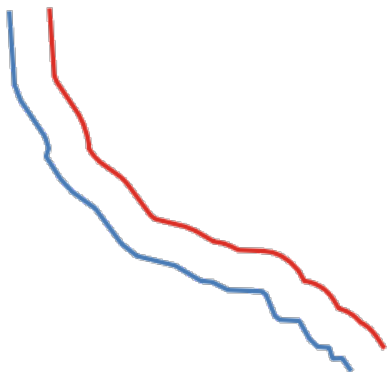


Abb. 28.94: In blue the source layer, in red the offset one

 Allows *features in-place modification* of line features

Siehe auch:

*Array of offset (parallel) lines, Translate*

Parameter



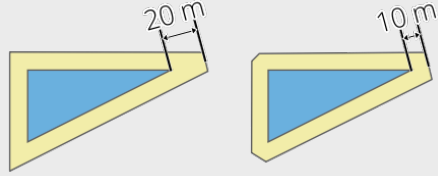
Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer
Distance	DISTANCE	[number  ] Default: 10.0	Offset distance. You can use the Data Defined button on the right to choose a field from which the radius will be calculated. This way you can have different radius for each feature (see <a href="#">Variable distance buffer</a> ).
Segments	SEGMENTS	[number] Default: 8	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
Join style	JOIN_STYLE	[enumeration] Default: 0	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: <ul style="list-style-type: none"><li>• 0 — Round</li><li>• 1 — Miter</li><li>• 2 — Bevel</li></ul> 

Abb. 28.95: Round, miter, and bevel join styles

Fortsetzung auf der nächsten Seite

Tab. 28.158 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Miter limit</b>	MITER_LIMIT	[number] Default: 2.0	Sets the maximum distance from the offset geometry to use when creating a mitered join as a factor of the offset distance (only applicable for miter join styles). Minimum: 1.0
			 <p>Abb. 28.96: A 10m buffer with a limit of 2 and a 10m buffer with a limit of 1</p>
<b>Offset</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	<p>Specify the output (offset) layer. One of:</p> <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> <p>The file encoding can also be changed here.</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Offset</b>	OUTPUT	[vector: line]	Output (offset) line layer

## Pythoncode

**Algorithm ID:** native:offsetline

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Oriented minimum bounding box

Calculates the minimum area rotated rectangle for each feature in the input layer.

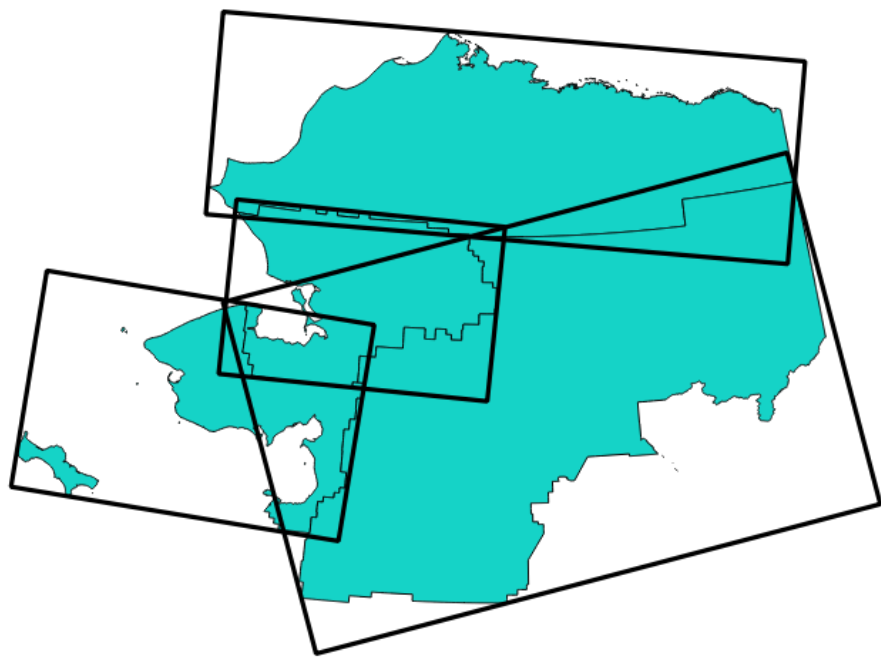


Abb. 28.97: Oriented minimum bounding box

 Allows *features in-place modification* of polygon features

**Siehe auch:**

*Minimum bounding geometry*

**Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Bounding boxes</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Bounding boxes</b>	OUTPUT	[vector: polygon]	The output polygon vector layer.

## Pythoncode

**Algorithm ID:** native:orientedminimumboundingbox

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Orthogonalize

Attempts to orthogonalize the geometries of the input line or polygon layer. This process shifts the vertices in the geometries to try to make every angle in the geometry either a right angle or a straight line.

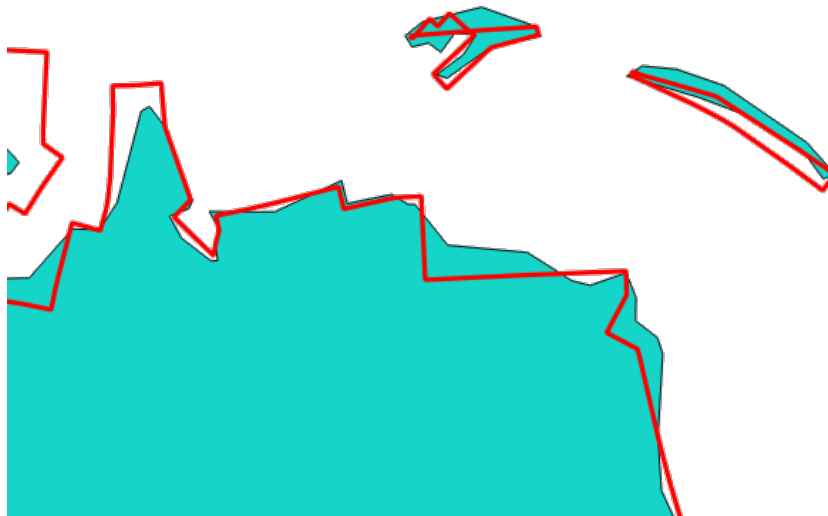


Abb. 28.98: In blue the source layer and in the red orthogonalized result



Allows *features in-place modification* of line and polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Maximum angle tolerance (degrees)</b>	ANGLE_TOLERANCE	[number] Default: 15	Specify the maximum deviation from a right angle or straight line a vertex can have for it to be adjusted. Smaller tolerances mean that only vertices which are already closer to right angles will be adjusted, and larger tolerances mean that vertices which deviate further from right angles will also be adjusted.
<b>Maximum algorithm iterations</b>	MAX_ITERATIONS	[number] Default: 1000	Setting a larger number for the maximum number of iterations will result in a more orthogonal geometry at the cost of extra processing time.
<b>Orthogonalized</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Orthogonalized</b>	OUTPUT	[same as input]	The output polygon vector layer with adjusted angles.

## Pythoncode

**Algorithm ID:** native:orthogonalize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Point on Surface


For each feature of the input layer, returns a point that is guaranteed to lie on the surface of the feature geometry.

 Allows *features in-place modification* of point features

**Siehe auch:**

*Centroids*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Create point on surface for each part</b>	ANGLE_TOLERANC	[boolean  ]	If checked, a point will be created for each part of the geometry.
<b>Point</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output point vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Point</b>	OUTPUT	[vector: point]	The output point vector layer.

## Pythoncode

**Algorithm ID:** native:pointonsurface

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Points along geometry

Creates points at regular intervals along line or polygon geometries. Created points will have new attributes added for the distance along the geometry and the angle of the line at the point.

An optional start and end offset can be specified, which controls how far from the start and end of the geometry the points should be created.

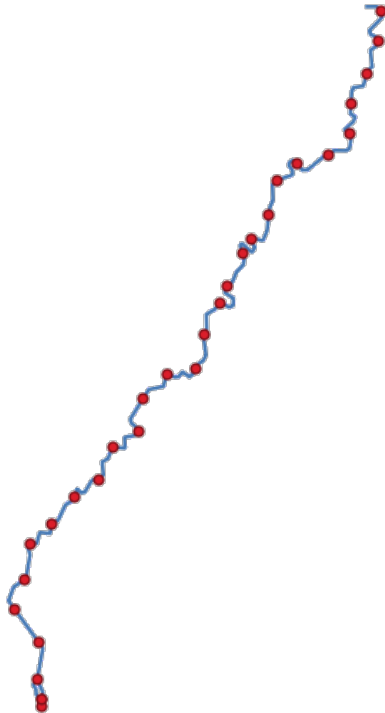





Abb. 28.99: Points created along the source line layer

Siehe auch:

*Interpolate point on line*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: polygon]	line, Input line or polygon vector layer
Distance	DISTANCE	[number  ] Default: 1.0	Distance between two consecutive points along the line
Start offset	START_OFFSET	[number  ] Default: 0.0	Distance from the beginning of the input line, representing the position of the first point.
End offset	END_OFFSET	[number  ] Default: 0.0	Distance from the end of the input line, representing the position beyond which no point feature should be created.

Fortsetzung auf der nächsten Seite

Tab. 28.160 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated points</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated points</b>	OUTPUT	[vector: point]	Point vector layer with features placed along lines or polygon boundaries of the input layer.

## Pythoncode

**Algorithm ID:** native:pointsalonglines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Points displacement

Given a distance of proximity, identifies nearby point features and radially distributes them over a circle whose center represents their barycenter. A convenient tool to scatter overlaid features.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Minimum distance to other points</b>	PROXIMITY	[number] Default: 1.0	Distance below which point features are considered close. Close features are distributed altogether.
<b>Displacement distance</b>	DISTANCE	[number] Default: 1.0	Radius of the circle on which close features are placed
<b>Horizontal distribution for two point case</b>	HORIZONTAL	[boolean] Default: False	When only two points are identified as close, aligns them horizontally on the circle instead of vertically.

Fortsetzung auf der nächsten Seite

Tab. 28.161 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Displaced</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Displaced</b>	OUTPUT	[vector: point]	Output point vector layer

## Pythoncode

**Algorithm ID:** qgis:pointdisplacement

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Pole of inaccessibility

Calculates the pole of inaccessibility for a polygon layer, which is the most distant internal point from the boundary of the surface.

This algorithm uses the 'polylabel' algorithm (Vladimir Agafonkin, 2016), which is an iterative approach guaranteed to find the true pole of inaccessibility within a specified tolerance. A more precise tolerance (lower value) requires more iterations and will take longer to calculate.

The distance from the calculated pole to the polygon boundary will be stored as a new attribute in the output layer.



Abb. 28.100: Pole of inaccessibility

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input vector layer
<b>Tolerance</b>	TOLERANCE	[number] Default: 1.0	Set the tolerance for the calculation
<b>Point</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Point</b>	OUTPUT	[vector: point]	The output point vector layer

## Pythoncode

**Algorithm ID:** native:poleofinaccessibility

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Polygonize

Creates a polygon layer whose features boundaries are generated from a line layer of **closed** features.

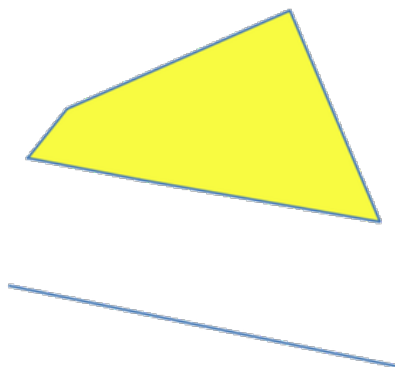


Abb. 28.101: The yellow polygons generated from the closed lines

**Bemerkung:** The line layer must have closed shapes in order to be transformed into a polygon.

**Siehe auch:**

*Polygons to lines, Lines to polygons, Convert geometry type*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Keep fields from the input layer</b>	KEEP_FIELDS	[boolean] Default: False	Check to keep the fields (only the table structure, not the values) of the input layer
Optional <b>Polygons from lines</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Polygons from lines</b>	OUTPUT	[vector: polygon]	The output polygon vector layer from lines

## Pythoncode

**Algorithm ID:** native:polygonize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Polygons to lines

Takes a polygon layer and creates a line layer, with lines representing the boundaries of the polygons in the input layer.

The attribute table of the output layer is the same as the one of the input layer.



Abb. 28.102: Black lines as the result of the algorithm

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Lines to polygons, Polygonize, Convert geometry type*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Lines</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Lines</b>	OUTPUT	[vector: line]	The output line vector layer from polygons

### Pythoncode

**Algorithm ID:** native:polygontolines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.



## Project points (Cartesian)

Projects point geometries by a specified distance and bearing (azimuth).



Allows *features in-place modification* of point features

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Bearing (degrees from North)</b>	BEARING	[number  ] Default: 0.0	Clockwise angle starting from North, in degree (°) unit
<b>Distance</b>	DISTANCE	[number  ] Default: 1.0	Distance to offset geometries, in layer units
<b>Projected</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output point vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Projected</b>	OUTPUT	[vector: point]	The output (projected) point vector layer

### Pythoncode

**Algorithm ID:** native:projectpointcartesian

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Promote to multipart

Takes a vector layer with singlepart geometries and generates a new one in which all geometries are multipart.

Input features which are already multipart features will remain unchanged.

This algorithm can be used to force geometries to multipart types in order to be compatible with data providers that require multipart features.



Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

*Aggregate, Collect geometries*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Multiparts</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output multipart vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Multiparts</b>	OUTPUT	[same as input]	The output multipart vector layer

## Pythoncode

**Algorithm ID:** native:promotetomulti

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Rectangles, ovals, diamonds

Creates a buffer area with a rectangle, oval or diamond shape for each feature of the input point layer. The shape parameters can be fixed for all features or dynamic using a field or an expression.

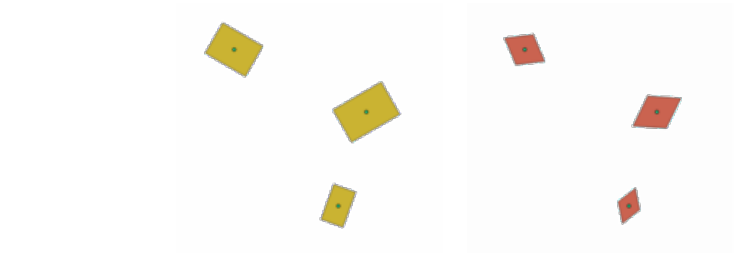





Abb. 28.103: Different buffer shapes with dynamic parameters

Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Buffer shape</b>	SHAPE	[enumeration]	The shape to use. One of: <ul style="list-style-type: none"><li>• 0 — Rectangles</li><li>• 1 — Ovals</li><li>• 2 — Diamonds</li></ul>
<b>Width</b>	WIDTH	[number  ] Default: 1.0	Width of the buffer shape
<b>Height</b>	HEIGHT	[number  ] Default: 1.0	Height of the buffer shape
<b>Rotation</b> Optional	ROTATION	[number  ] Default: None	Rotation of the buffer shape
<b>Number of segments</b>	SEGMENTS	[number] Default: 36	Number of segments for a full circle ( <i>Ovals</i> shape)
<b>Ergebnis</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[vector: polygon]	The output vector layer (with the buffer shapes)

## Pythoncode

**Algorithm ID:** native:rectanglesovalsdiamonds

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Remove duplicate vertices

Removes duplicate vertices from features, wherever removing the vertices does not result in a degenerate geometry.

The tolerance parameter specifies the tolerance for coordinates when determining whether vertices are identical.

By default, Z values are not considered when detecting duplicate vertices. E.g. two vertices with the same X and Y coordinate but different Z values will still be considered duplicate and one will be removed. If the *Use Z Value* parameter is true, then the Z values are also tested and vertices with the same X and Y but different Z will be maintained.





Allows *features in-place modification* of point, line, and polygon features

**Bemerkung:** Duplicate vertices are not tested between different parts of a multipart geometry, e.g. a multipoint geometry with overlapping points will not be changed by this method.

## Siehe auch:

[Extract vertices](#), [Extract specific vertices](#), [Delete duplicate geometries](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Tolerance</b>	TOLERANCE	[number  ] Default: 0.000001	Vertices closer than the specified distance are considered duplicates
<b>Use Z value</b>	USE_Z_VALUE	[boolean  ] Default: False	If the <i>Use Z Value</i> parameter is true, then the Z values are also tested and vertices with the same X and Y but different Z will be maintained.

Fortsetzung auf der nächsten Seite

Tab. 28.164 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Cleaned</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Cleaned</b>	OUTPUT	[same as input]	The output vector layer (without duplicate vertices)

## Pythoncode

**Algorithm ID:** native:removeduplicatevertices

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Remove null geometries

Removes any features which do not have a geometry from a vector layer. All other features will be copied unchanged.

The features with null geometries can be saved to a separate layer.

If *Also remove empty geometries* is checked, the algorithm removes features whose geometries have no coordinates, i.e., geometries that are empty. In that case, also the null output will reflect this option, containing both null and empty geometries.

**Siehe auch:**

[Delete duplicate geometries](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer (with non-NULL geometries)
<b>Also remove empty geometries</b>	REMOVE_EMPTY	[boolean]	

Fortsetzung auf der nächsten Seite

Tab. 28.165 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Non null geometries</b>	OUTPUT Optional	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the non-NULL (and non-empty) geometries. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Null geometries</b> Optional	NULL_OUTPUT	[same as input] Default: [Skip output]	Specify the output vector layer for the NULL (and empty) geometries. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Null geometries</b>	NULL_OUTPUT	[same as input]	Output vector layer (for NULL and, if chosen, empty geometries)
<b>Non null geometries</b>	OUTPUT	[same as input]	The output vector layer (without NULL and, if chosen, empty geometries)

## Pythoncode

**Algorithm ID:** native:removenullgeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Reverse line direction

Inverts the direction of a line layer.

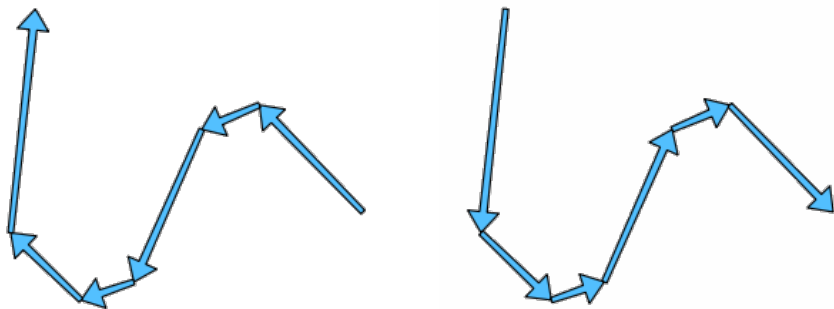


Abb. 28.104: Before and after the direction inversion

☒ Allows *features in-place modification* of line features

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer
Reversed	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.

Ausgaben

Bezeichnung	Name	Typ	Beschreibung
Reversed	OUTPUT	[vector: line]	The output line vector layer (with reversed lines)

Pythoncode

Algorithm ID: native:reverselinedirection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Rotate

Rotates feature geometries by the specified angle clockwise. The rotation occurs around each feature's centroid, or optionally around a unique preset point.




Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

*Translate, Swap X and Y coordinates*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Rotation (degrees clockwise)</b>	ANGLE	[number  ] Default: 0.0	Angle of the rotation in degrees
<b>Rotation anchor point (x, y)</b> Optional	ANCHOR	[point] Default: None	X,Y coordinates of the point to rotate the features around. If not set the rotation occurs around each feature's centroid.
<b>Rotated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with rotated geometries). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rotated</b>	OUTPUT	[same as input]	The output vector layer with rotated geometries

## Pythoncode

**Algorithm ID:** native:rotatefeatures

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Roundness

Calculates the roundness of each feature and stores it as a new field. The input vector layer must contain polygons.

The roundness of a polygon is defined as  $4\pi \times \text{polygon area} / \text{perimeter}^2$ . The roundness value varies between 0 and 1. A perfect circle has a roundness of 1, while a completely flat polygon has a roundness of 0.

---

**Bemerkung:** The algorithm returns NULL for multipart polygon features.

---



Allows *features in-place modification* of polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input vector layer
<b>Roundness</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer (with roundness field). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rotated</b>	OUTPUT	[same as input]	The output vector layer with roundness value in a field

## Pythoncode

**Algorithm ID:** native:roundness

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Segmentize by maximum angle


Segmentizes a geometry by converting curved sections to linear sections.

The segmentization is performed by specifying the maximum allowed radius angle between vertices on the straightened geometry (e.g the angle of the arc created from the original arc center to consecutive output vertices on the linearized geometry). Non-curved geometries will be retained without change.

**Siehe auch:**

*Segmentize by maximum distance, Simplify, Smooth*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Maximum angle between vertices (degrees)</b>	ANGLE	[number  ] Default: 5.0	Maximum allowed radius angle between vertices on the straightened geometry
<b>Segmentized</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with segmentized geometries). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Segmentized</b>	OUTPUT	[same as input]	The output vector layer with segmentized geometries

### Pythoncode

**Algorithm ID:** native:segmentizebymaxangle

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Segmentize by maximum distance


Segmentizes a geometry by converting curved sections to linear sections.

The segmentization is performed by specifying the maximum allowed offset distance between the original curve and the segmentized representation. Non-curved geometries will be retained without change.

**Siehe auch:**

*Segmentize by maximum angle, Simplify, Smooth*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Maximum offset distance</b>	DISTANCE	[number]  Default: 1.0	Maximum allowed offset distance between the original curve and the segmentized representation, in the layer units.
<b>Segmentized</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with segmentized geometries). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Segmentized</b>	OUTPUT	[same as input]	The output vector layer with segmentized geometries

### Pythoncode

**Algorithm ID:** native:segmentizebymaxdistance

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.


## M-Wert einstellen

Sets the M value for geometries in a layer.

If M values already exist in the layer, they will be overwritten with the new value. If no M values exist, the geometry will be upgraded to include M values and the specified value used as the initial M value for all geometries.




Allows *features in-place modification* of point, line, and polygon features with M enabled

**Tip:** Use the  Identify Features button to check the added M value: the results are available in the *Identify Results* dialog.

### Siehe auch:

*Set M value from raster, Set Z value, Drop M/Z values*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>M Value</b>	M_VALUE	[number  ] Default: 0.0	M value to assign to the feature geometries
<b>M Added</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>M Added</b>	OUTPUT	[same as input]	The output vector layer (with M values assigned to the geometries)

## Pythoncode

**Algorithm ID:** native:setmvalue

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

## Set M value from raster

Uses values sampled from a band within a raster layer to set the M value for every overlapping vertex in the feature geometry. The raster values can optionally be scaled by a preset amount.

If M values already exist in the layer, they will be overwritten with the new value. If no M values exist, the geometry will be upgraded to include M values.






Allows *features in-place modification* of point, line, and polygon features with M enabled

**Siehe auch:**

*Drape (set Z value from raster), M-Wert einstellen*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Raster layer</b>	RASTER	[raster]	Raster layer with M values
<b>Band number</b>	BAND	[raster band] Default: 1	The raster band from which the M values are taken
<b>Value for NoData or non-intersecting vertices</b>	NODATA	[number  Default: 0.0	Value to use in case the vertex does not intersect (a valid pixel of) the raster
<b>Scale factor</b>	SCALE	[number  Default: 1.0	Scaling value: the band values are multiplied by this value.
<b>Offset</b>	OFFSET	[number  Default: 0.0	Offset value: it is algebraically added to the band values after applying the „Scale factor“.
<b>Updated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with updated M values). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Updated</b>	OUTPUT	[same as input]	The output vector layer (with updated M values)

## Pythoncode

**Algorithm ID:** `native:setmfromraster`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.


## Set Z value

Sets the Z value for geometries in a layer.

If Z values already exist in the layer, they will be overwritten with the new value. If no Z values exist, the geometry will be upgraded to include Z values and the specified value used as the initial Z value for all geometries.




Allows *features in-place modification* of point, line, and polygon features with Z enabled

**Tipp:** Use the  Identify Features button to check the added Z value: the results are available in the *Identify Results* dialog.

## Siehe auch:

*Drape (set Z value from raster), M-Wert einstellen, Drop M/Z values*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Z Value</b>	Z_VALUE	[number  ] Default: 0.0	Z value to assign to the feature geometries
<b>Z Added</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Z Added</b>	OUTPUT	[same as input]	The output vector layer (with Z values assigned)

## Pythoncode

**Algorithm ID:** native:setzvalue

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Simplify

Simplifies the geometries in a line or polygon layer. It creates a new layer with the same features as the ones in the input layer, but with geometries containing a lower number of vertices.

The algorithm gives a choice of simplification methods, including distance based (the „Douglas-Peucker“ algorithm), area based („Visvalingam“ algorithm) and snapping geometries to grid.



Abb. 28.105: Clockwise from top left: source layer and increasing simplification tolerances


 Allows *features in-place modification* of line and polygon features

**Default menu:** *Vector ► Geometry Tools*

**Siehe auch:**

*Smooth, Denisify by count, Denisify by interval*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Simplification method</b>	METHOD	[enumeration] Default: 0	Simplification method. One of: <ul style="list-style-type: none"> <li>• 0 — Distance (Douglas-Peucker)</li> <li>• 1 — Snap to grid</li> <li>• 2 — Area (Visvalingam)</li> </ul>
<b>Tolerance</b>	TOLERANCE	[number]  Default: 1.0	Threshold tolerance (in units of the layer): if the distance between two nodes is smaller than the tolerance value, the segment will be simplified and vertices will be removed.
<b>Simplified</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (simplified) vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Simplified</b>	OUTPUT	[same as input]	The output (simplified) vector layer

## Pythoncode

**Algorithm ID:** native:simplifygeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Single sided buffer

Computes a buffer on lines by a specified distance on one side of the line only.

Buffer always results in a polygon layer.

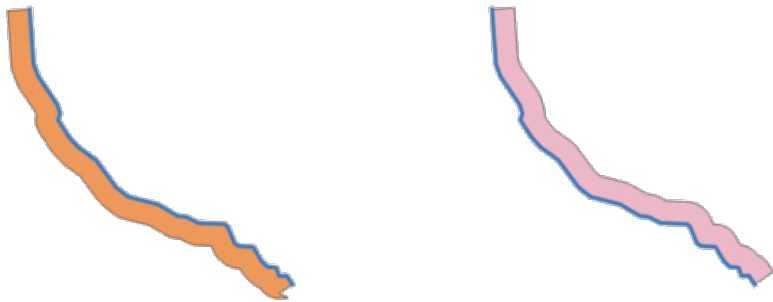


Abb. 28.106: Left versus right side buffer on the same vector line layer

Siehe auch:

Buffer

Parameter


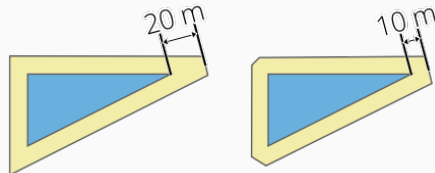
Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Distance</b>	DISTANCE	[number] Default: 10.0	Buffer distance.
<b>Side</b>	SIDE	[enumeration] Default: 0	Which side to create the buffer on. One of: <ul style="list-style-type: none"><li>• 0 – Left</li><li>• 1 – Right</li></ul>
<b>Segments</b>	SEGMENTS	[number] Default: 8	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
<b>Join style</b>	JOIN_STYLE	[enumeration] Default: 0	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: <ul style="list-style-type: none"><li>• 0 — Round</li><li>• 1 — Miter</li><li>• 2 — Bevel</li></ul> <div></div>

Abb. 28.107: Round, miter, and bevel join styles

Fortsetzung auf der nächsten Seite



Tab. 28.168 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Miter limit</b>	MITER_LIMIT	[number] Default: 2.0	Sets the maximum distance from the offset geometry to use when creating a mitered join as a factor of the offset distance (only applicable for miter join styles). Minimum: 1.0
			
<b>Buffer</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	<p>Abb. 28.108: A 10m buffer with a limit of 2 and a 10m buffer with a limit of 1</p> <p>Specify the output (buffer) layer. One of:</p> <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> <p>The file encoding can also be changed here.</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffer</b>	OUTPUT	[vector: polygon]	Output (buffer) polygon layer

## Pythoncode

**Algorithm ID:** native:singlesidedbuffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Smooth

Smooths the geometries in a line or polygon layer by adding more **vertices and corners** to the feature geometries.

The iterations parameter dictates how many smoothing iterations will be applied to each geometry. A higher number of iterations results in smoother geometries with the cost of greater number of nodes in the geometries.

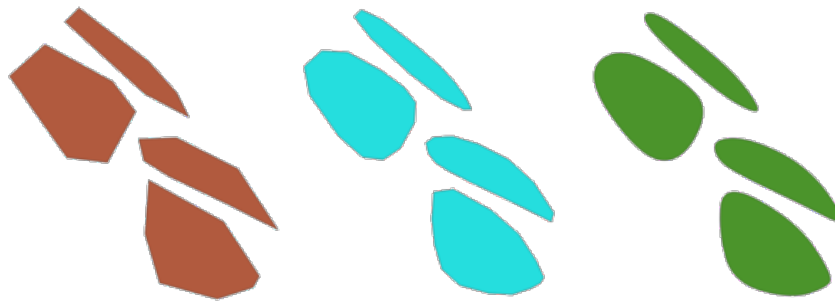


Abb. 28.109: Increasing number of iterations causes smoother geometries

The offset parameter controls how „tightly“ the smoothed geometries follow the original geometries. Smaller values results in a tighter fit, and larger values will create a looser fit.

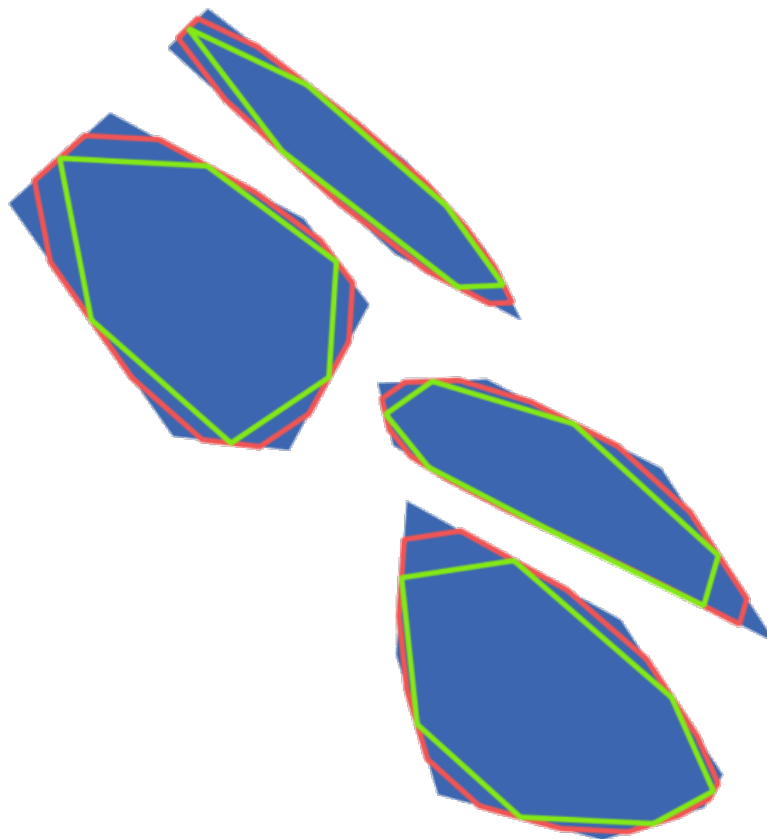


Abb. 28.110: Blue: the input layer. Offset 0.25 gives the red line, while offset 0.50 gives the green line.




The maximum angle parameter can be used to prevent smoothing of nodes with large angles. Any node where the angle of the segments to either side is larger than this will not be smoothed. For example, setting the maximum angle to 90 degrees or lower would preserve right angles in the geometry.

☒ Allows *features in-place modification* of line and polygon features

**Siehe auch:**

*Simplify, Denseify by count, Denseify by interval*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line, polygon]	Input line or polygon vector layer
<b>Iterations</b>	ITERATIONS	[number  ] Default: 1	Increasing the number of iterations will give smoother geometries (and more vertices).
<b>Offset</b>	OFFSET	[number  ] Default: 0.25	Increasing values will <i>move</i> the smoothed lines / boundaries further away from the input lines / boundaries.
<b>Maximum node angle to smooth</b>	MAX_ANGLE	[number  ] Default: 180.0	Every node below this value will be smoothed
<b>Smoothed</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (smoothed) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Smoothed</b>	OUTPUT	[same as input]	Output (smoothed) vector layer

## Pythoncode

**Algorithm ID:** native:smoothgeometry

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Snap geometries to layer

Snaps the geometries in a layer either to the geometries from another layer, or to geometries within the same layer.

Matching is done based on a tolerance distance, and vertices will be inserted or removed as required to make the geometries match the reference geometries.



Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

[Snap points to grid](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Reference layer</b>	REFERENCE_LAYER	[vector: any]	Vector layer to snap to
<b>Tolerance</b>	TOLERANCE	[number] Default: 10.0	Control how close input vertices need to be to the reference layer geometries before they are snapped.
<b>Behavior</b>	BEHAVIOR	[enumeration] Default: 0	<p>Snapping can be done to an existing node or a segment (its closest point to the vertex to move). Available snapping options:</p> <ul style="list-style-type: none"> <li>• 0 — Prefer aligning nodes, insert extra vertices where required Prefer to snap to nodes, even when a segment may be closer than a node. New nodes will be inserted to make geometries follow each other exactly when inside allowable tolerance.</li> <li>• 1 — Prefer closest point, insert extra vertices where required Snap to closest point, regardless of it is a node or a segment. New nodes will be inserted to make geometries follow each other exactly when inside allowable tolerance.</li> <li>• 2 — Prefer aligning nodes, don't insert new vertices Prefer to snap to nodes, even when a segment may be closer than a node. No new nodes will be inserted.</li> <li>• 3 — Prefer closest point, don't insert new vertices Snap to closest point, regardless of it is a node or a segment. No new nodes will be inserted.</li> <li>• 4 — Move end points only, prefer aligning nodes Only snap start/end points of lines (point features will also be snapped, polygon features will not be modified), prefer to snap to nodes.</li> <li>• 5 — Move end points only, prefer closest point Only snap start/end points of lines (point features will also be snapped, polygon features will not be modified), snap to closest point</li> <li>• 6 — Snap end points to end points only Only snap the start/end points of lines to other start/end points of lines</li> <li>• 7 — Snap to anchor nodes (single layer only)</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.170 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Snapped geometry</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (snapped) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Snapped geometry</b>	OUTPUT	[same as input]	Output (snapped) vector layer

## Pythoncode

**Algorithm ID:** native:snapgeometries

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Snap points to grid

Modifies the coordinates of geometries in a vector layer, so that all points or vertices are snapped to the closest point of a grid.

If the snapped geometry cannot be calculated (or is totally collapsed) the feature's geometry will be cleared.

Snapping can be performed on the X, Y, Z or M axis. A grid spacing of 0 for any axis will disable snapping for that axis.







Allows *features in-place modification* of point, line, and polygon features

**Bemerkung:** Snapping to grid may generate an invalid geometry in some corner cases.

## Siehe auch:

*Snap geometries to layer*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>X Grid Spacing</b>	HSPACING	[number  ] Default: 1.0	Grid spacing on the X axis
<b>Y Grid Spacing</b>	VSPACING	[number  ] Default: 1.0	Grid spacing on the Y axis
<b>Z Grid Spacing</b>	ZSPACING	[number  ] Default: 0.0	Grid spacing on the Z axis
<b>M Grid Spacing</b>	MSPACING	[number  ] Default: 0.0	Grid spacing on the M axis
<b>Snapped</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (snapped) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Snapped</b>	OUTPUT	[same as input]	Output (snapped) vector layer

## Pythoncode

**Algorithm ID:** native:snappointstogrid

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.


## Split lines by maximum length

Takes a line (or curve) layer and splits each feature into multiple parts, where each part is of a specified maximum length. Z and M values at the start and end of the new line substrings are linearly interpolated from existing values.



Allows *features in-place modification* of line features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	The input line vector layer
<b>Maximum line length</b>	LENGTH	[number  ] Default: 10.0	The maximum length of a line in the output.
<b>Split</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Split</b>	OUTPUT	[vector: line]	The new line vector layer - the length of the feature geometries is less than or equal to the length specified in the LENGTH parameter.

## Pythoncode

**Algorithm ID:** native:splitlinesbylength

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Subdivide

Subdivides the geometry. The returned geometry will be a collection containing subdivided parts from the original geometry, where no part has more than the specified maximum number of nodes.

This is useful for dividing a complex geometry into less complex parts, easier to spatially index and faster to perform spatial operations. Curved geometries will be segmentized before subdivision.

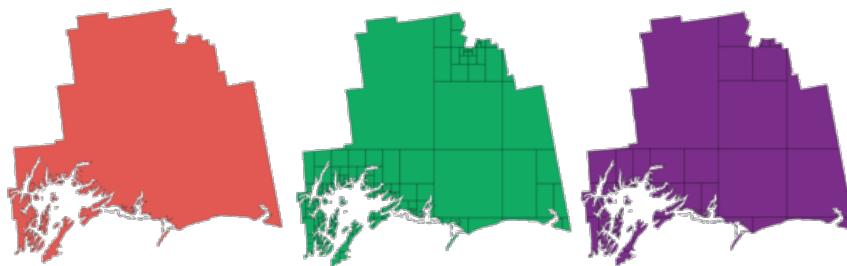


Abb. 28.111: Left the input layer, middle maximum nodes value is 100 and right maximum value is 200


 Allows *features in-place modification* of point, line, and polygon features

**Bemerkung:** Subdividing a geometry can generate geometry parts that may not be valid and may contain self-intersections.

#### Siehe auch:

*Explode lines, Line substring*

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Maximum nodes in parts</b>	MAX_NODES	[number  ] Default: 256	Maximum number of vertices each new geometry part is allowed to have. Fewer <i>sub-parts</i> for higher values.
<b>Subdivided</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (subdivided) vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Subdivided</b>	OUTPUT	[same as input]	Output vector layer



## Pythoncode

**Algorithm ID:** native:subdivide

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Swap X and Y coordinates

Switches the X and Y coordinate values in input geometries.

It can be used to repair geometries which have accidentally had their latitude and longitude values reversed.



Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

[Translate, Rotate](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Swapped</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Swapped</b>	OUTPUT	[same as input]	Output (swapped) vector layer

## Pythoncode

**Algorithm ID:** native:swapxy

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Tapered buffers

Creates tapered buffer along line geometries, using a specified start and end buffer diameter.

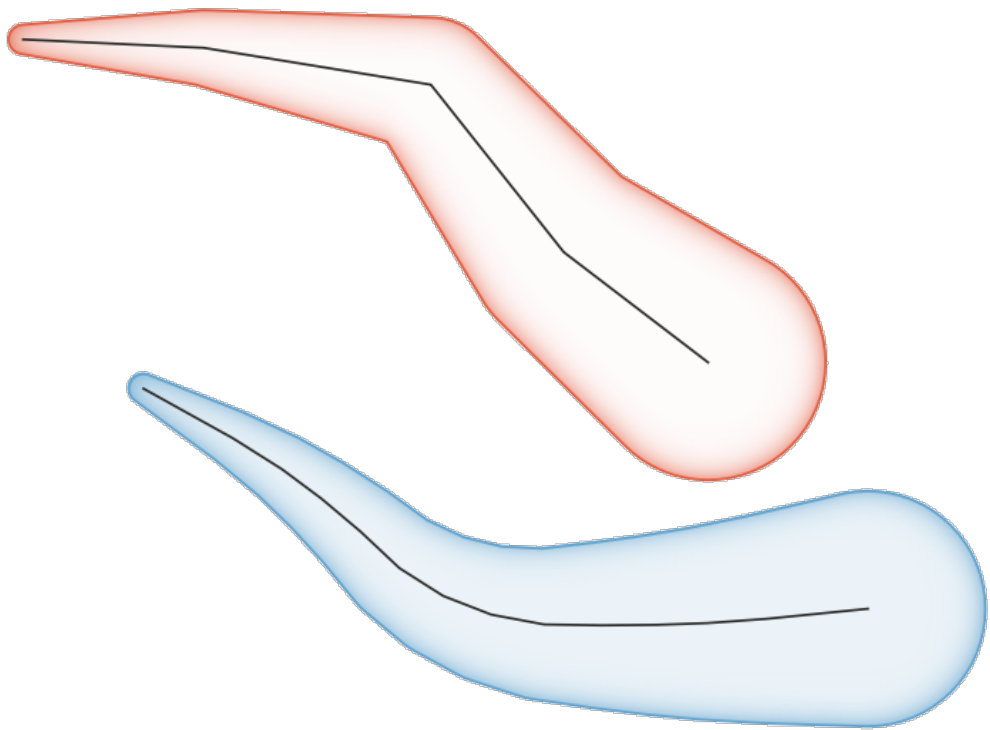





Abb. 28.112: Tapered buffer example

Siehe auch:

*Variable width buffer (by M value), Buffer, Create wedge buffers*

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer
Start width	START_WIDTH	[number  ] Default: 0.0	Represents the radius of the buffer applied at the start point of the line feature
End width	END_WIDTH	[number  ] Default: 0.0	Represents the radius of the buffer applied at the end point of the line feature.
Segments	SEGMENTS	[number  ] Default: 16	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.

Fortsetzung auf der nächsten Seite

Tab. 28.172 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Buffered</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output (buffer) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffered</b>	OUTPUT	[vector: polygon]	Output (buffer) polygon layer

## Pythoncode

**Algorithm ID:** native:taperedbuffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Tessellate

Tessellates a polygon geometry layer, dividing the geometries into triangular components.

The output layer consists of multipolygon geometries for each input feature, with each multipolygon consisting of multiple triangle component polygons.

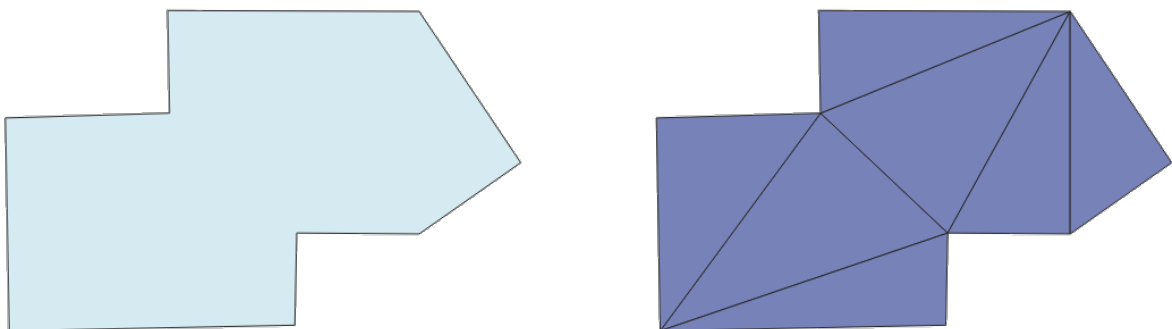


Abb. 28.113: Tessellated polygon (right)

 Allows *features in-place modification* of polygon features

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: polygon]	Input polygon vector layer
<b>Tesselated</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Tesselated</b>	OUTPUT	[vector: polygon]	Output multipolygonZ layer

## Pythoncode

**Algorithm ID:** 3d:tessellate

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Transect

Creates transects on vertices for (multi)linestring.

A transect is a line oriented from an angle (by default perpendicular) to the input polylines (at vertices).

Field(s) from feature(s) are returned in the transect with these new fields:

- TR\_FID: ID of the original feature
- TR\_ID: ID of the transect. Each transect have an unique ID
- TR\_SEGMENT: ID of the segment of the linestring
- TR\_ANGLE: Angle in degrees from the original line at the vertex
- TR\_LENGTH: Total length of the transect returned
- TR\_ORIENT: Side of the transect (only on the left or right of the line, or both side)

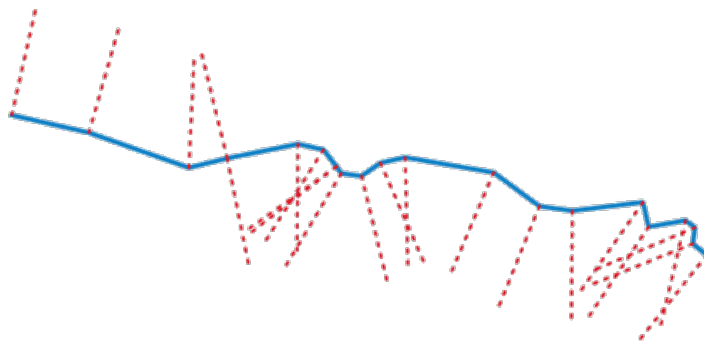




Abb. 28.114: Dashed red lines represent the transect of the input line layer

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Length of the transect</b>	LENGTH	[number]  Default: 5.0	Length in map unit of the transect
<b>Angle in degrees from the original line at the vertices</b>	ANGLE	[number]  Default: 90.0	Change the angle of the transect
<b>Side to create the transect</b>	SIDE	[enumeration]	Choose the side of the transect. Available options are: <ul style="list-style-type: none"> <li>• 0 — Left</li> <li>• 1 — Right</li> <li>• 2 — Both</li> </ul>
<b>Transect</b>	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Transect</b>	OUTPUT	[vector: line]	Output line layer

## Pythoncode

**Algorithm ID:** native:transect

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Translate

Moves the geometries within a layer, by offsetting with a predefined X and Y displacement.

Z and M values present in the geometry can also be translated.

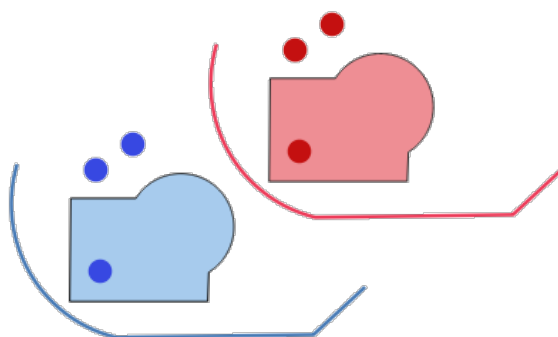


Abb. 28.115: Dashed lines represent the translated geometry of the input layer







Allows *features in-place modification* of point, line, and polygon features

**Siehe auch:**

*Array of translated features, Offset lines, Rotate, Swap X and Y coordinates*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Offset distance (x-axis)</b>	DELTA_X	[number  ] Default: 0.0	Displacement to apply on the X axis
<b>Offset distance (y-axis)</b>	DELTA_Y	[number  ] Default: 0.0	Displacement to apply on the Y axis
<b>Offset distance (z-axis)</b>	DELTA_Z	[number  ] Default: 0.0	Displacement to apply on the Z axis
<b>Offset distance (m values)</b>	DELTA_M	[number  ] Default: 0.0	Displacement to apply on the M axis

Fortsetzung auf der nächsten Seite

Tab. 28.174 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Translated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Translated</b>	OUTPUT	[same as input]	Output vector layer

## Pythoncode

**Algorithm ID:** native:translategeometry

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Variable width buffer (by M value)

Creates variable width buffers along lines, using the M value of the line geometries as the diameter of the buffer at each vertex.

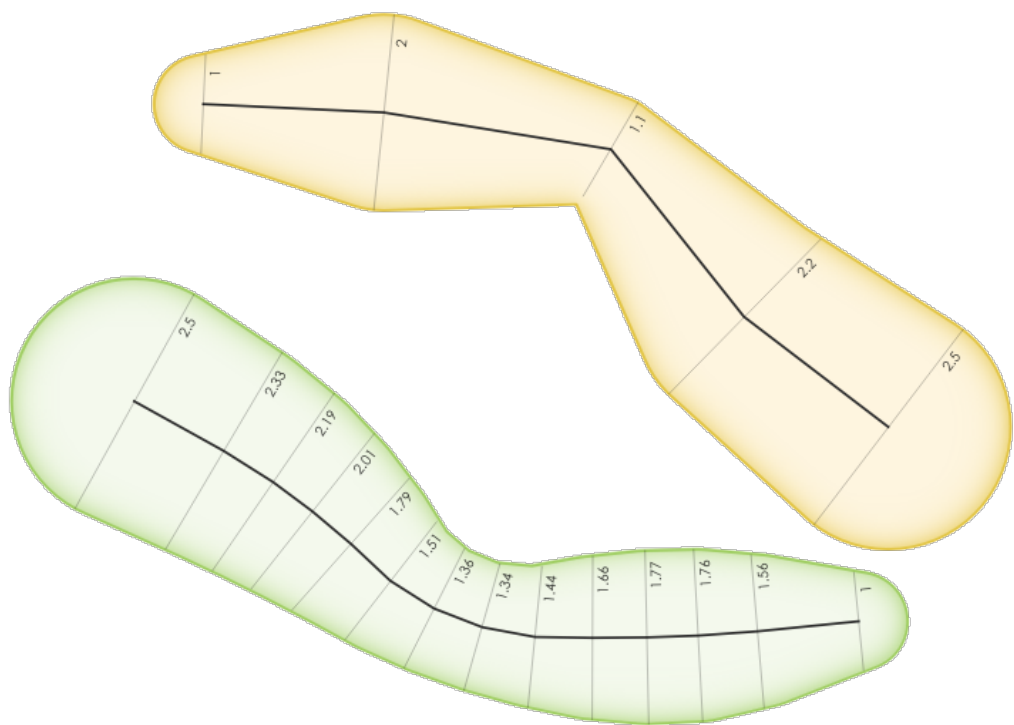



Abb. 28.116: Variable buffer example

**Siehe auch:**

*Tapered buffers, Buffer, M-Wert einstellen, Variable distance buffer*

**Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line vector layer
<b>Segments</b>	SEGMENTS	[number  ] Default: 16	Number of the buffer segments per quarter circle. It can be a unique value (same value for all the features), or it can be taken from features data (the value can depend on feature attributes).
<b>Buffered</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output (buffer) layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li><li>• Append to Layer...</li></ul> The file encoding can also be changed here.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffered</b>	OUTPUT	[vector: polygon]	Variable buffer polygon layer

## Pythoncode

**Algorithm ID:** native:bufferbym

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Voronoi polygons

Takes a point layer and generates a polygon layer containing the Voronoi polygons (known also as Thiessen polygons) corresponding to those input points.

Any location within a Voronoi polygon is closer to the associated point than to any other point.

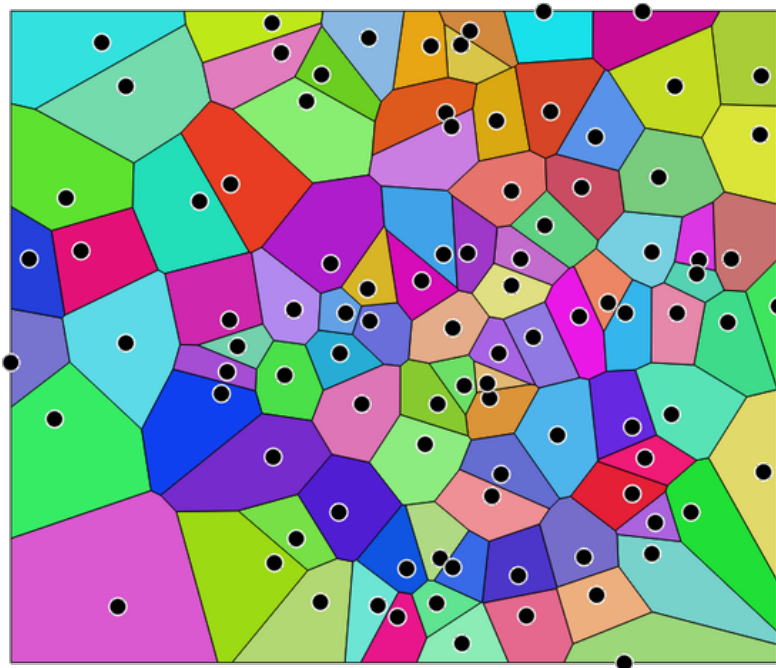


Abb. 28.117: Voronoi polygons

**Default menu:** *Vector ► Geometry Tools*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Buffer region (% of extent)</b>	BUFFER	[number] Default: 0.0	The extent of the output layer will be this much bigger than the extent of the input layer
<b>Tolerance</b> Optional	TOLERANCE	[number] Default: 0.0	Specifies an optional snapping tolerance which can be used to improve the robustness of the voronoi.
<b>Copy attributes from input features</b>	COPY_ATTRIBUTE	[boolean] Default: True	Specifies whether fields storing involved point features ID should be added to the output. If False, an id field is created to identify the polygons.
<b>Voronoi polygons</b>	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output layer (with the Voronoi polygons). One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Voronoi polygons</b>	OUTPUT	[vector: polygon]	Voronoi polygons of the input point vector layer

## Pythoncode

**Algorithm ID:** native:voronoipolygons

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### 28.1.23 Vector overlay

#### Clip

Clips a vector layer using the features of an additional polygon layer.

Only the parts of the features in the input layer that fall within the polygons of the overlay layer will be added to the resulting layer.

#### Warnung: Geometry modification only

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

This algorithm uses spatial indexes on the providers, prepared geometries and apply a clipping operation if the geometry isn't wholly contained by the mask geometry.

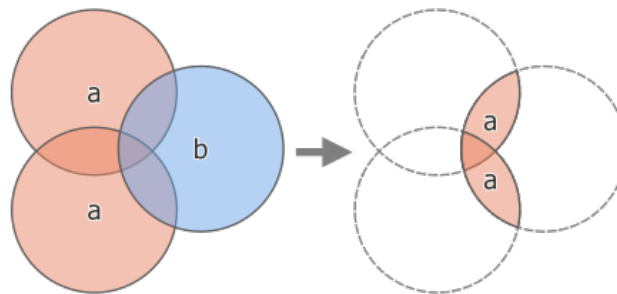


Abb. 28.118: Clipping operation between a two-feature input layer ,a' and a single feature overlay layer ,b' (left) - resulting in a new layer with the modified ,a' features (right)

 Allows *features in-place modification* of point, line, and polygon features

**Default menu:** *Vector ► Geoprocessing Tools*

**Siehe auch:**

*Intersection, Difference*

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer containing the features to be clipped
<b>Overlay layer</b>	OVERLAY	[vector: polygon]	Layer containing the clipping features
<b>Clipped</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the features from the input layer that are inside the overlay (clipping) layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clipped</b>	OUTPUT	[same as input]	Layer containing features from the input layer split by the overlay layer.

## Pythoncode

**Algorithm ID:** qgis:clip

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Difference

Extracts features from the input layer that don't fall within the boundaries of the overlay layer.

Input layer features that partially overlap the overlay layer feature(s) are split along the boundary of those feature(s) and only the portions outside the overlay layer features are retained.

### Warnung: Geometry modification only

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

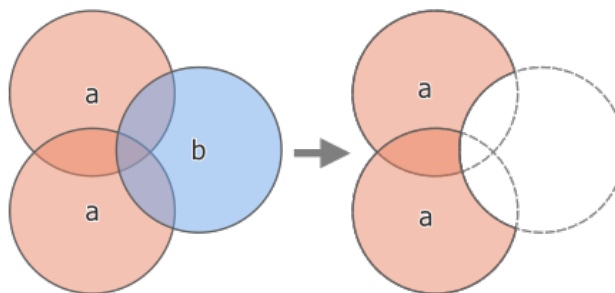


Abb. 28.119: Difference operation between a two-features input layer ,a' and a single feature overlay layer ,b' (left) - resulting in a new layer with the modified ,a' features (right)



Allows *features in-place modification* of point, line, and polygon features

**Default menu:** Vector ► Geoprocessing Tools

**Siehe auch:**

*Difference (multiple), Symmetrical difference, Clip*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to extract (parts of) features from.
<b>Overlay layer</b>	OVERLAY	[vector: any]	Layer containing the geometries that will be subtracted from the input layer geometries. It is expected to have at least as many dimensions (point: 0D, line: 1D, polygon: 2D, volume: 3D) as the input layer geometries.
<b>Difference</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the (parts of) features from the input layer that are not inside the overlay layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Grid size</b> Optional	GRID_SIZE	[number] Default: Not set	If provided, the input geometries are snapped to a grid of the given size, and the result vertices are computed on that same grid. Requires GEOS 3.9.0 or higher.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Difference</b>	OUTPUT	[same as input]	Layer containing (parts of) features from the input layer not overlapping the overlay layer.

## Pythoncode

**Algorithm ID:** qgis:difference

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Difference (multiple)

Extracts features from the input layer that fall completely outside or only partially overlap the features from any of the overlay layer(s).

For each overlay layer the difference is calculated between the result of all previous difference operations and this overlay layer. Input layer features that partially overlap feature(s) in the overlay layer are split along those features' boundary and only the portions outside the overlay layer features are retained.

### Warnung: Geometry modification only

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

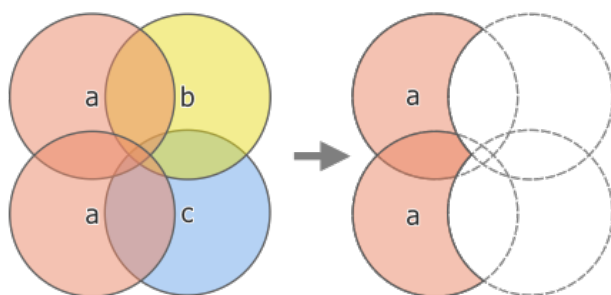


Abb. 28.120: Difference operation between a two-feature input layer ,a' and single feature overlay layers ,b' and ,c' (left) - resulting in a new layer with the modified ,a' features (right)

### Siehe auch:

*Difference, Symmetrical difference, Clip*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to extract (parts of) features from.
<b>Overlay layers</b>	OVERLAYS	[vector: any] [list]	List of layers containing the geometries that will be subtracted from the input layer geometries. They are expected to have at least as many dimensions (point: 0D, line: 1D, polygon: 2D, volume: 3D) as the input layer geometries.
<b>Difference</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the (parts of) features from the input layer that do not overlap features of the overlay layers. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Difference</b>	OUTPUT	[same as input]	Layer containing (parts of) features from the input layer not overlapping features from the overlay layers.

## Pythoncode

**Algorithm ID:** qgis:multidifference

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract/clip by extent

Creates a new vector layer that only contains features which fall within a specified extent.

Any features which intersect the extent will be included.

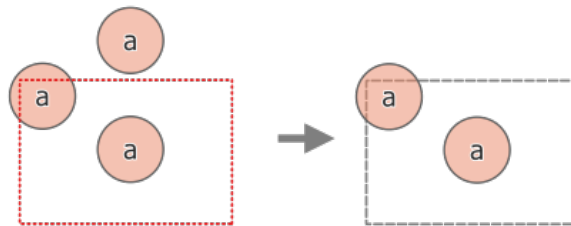
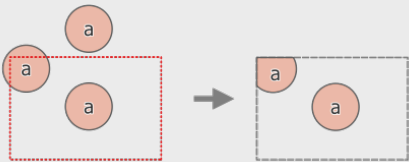


Abb. 28.121: Extract operation between a three-feature input layer ‚a‘ and a dashed extent (left) - resulting features with dashed extent for reference (right)

## Siehe auch:

*Clip*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to extract (parts of) features from.
<b>Extent</b> (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Extent for clipping. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Clip features to extent</b>	CLIP	[boolean] Default: False	<p>If checked, output geometries will be automatically converted to multi geometries to ensure uniform output types. Moreover the geometries will be clipped to the extent chosen instead of taking the whole geometry as output.</p>  <p>Abb. 28.122: Extract operation between a three-feature input layer ,a' and a dashed extent (left) - resulting features with dashed extent for reference (right)</p>
<b>Extracted</b>	OUTPUT	[same as input] Default: [Create temporary layer]	<p>Specify the layer to contain the features from the input layer that are inside the clip extent. One of:</p> <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> <p>The file encoding can also be changed here.</p>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted</b>	OUTPUT	[same as input]	Layer containing the clipped features.

## Pythoncode

**Algorithm ID:** qgis:extractbyextent

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Intersection

Extracts the portions of features from the input layer that overlap features in the overlay layer.

Features in the intersection layer are assigned the attributes of the overlapping features from both the input and overlay layers.

### Warnung: Geometry modification only

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

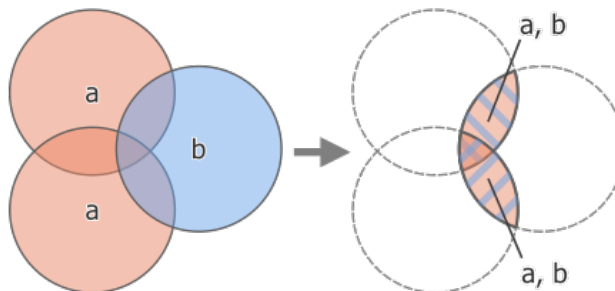


Abb. 28.123: Intersection operation between a two-feature input layer ,a‘ and a single feature overlay layer ,b‘ (left) - overlapping areas become a new two-feature layer with both layers‘ attributes (right)

**Default menu:** *Vector ► Geoprocessing Tools*

**Siehe auch:**

*Intersection (multiple), Clip, Difference*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to extract (parts of) features from.
<b>Overlay layer</b>	OVERLAY	[vector: any]	Layer containing the features to check for overlap. Its features' geometry is expected to have at least as many dimensions (point: 0D, line: 1D, polygon: 2D, volume: 3D) as the input layer's.
<b>Input fields to keep (leave empty to keep all fields)</b> Optional	INPUT_FIELDS	[tablefield: any] [list] Default: None	Field(s) of the input layer to keep in the output. If no fields are chosen all fields are taken.
<b>Overlay fields to keep (leave empty to keep all fields)</b> Optional	OVERLAY_FIELDS	[tablefield: any] [list] Default: None	Field(s) of the overlay layer to keep in the output. If no fields are chosen all fields are taken. Duplicate field names will be appended a count suffix to avoid collision.
<b>Intersection</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain (the parts of) the features from the input layer that overlap one or more features from the overlay layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Overlay fields prefix</b> Optional	OVERLAY_FIELDS_PREFIX	[string]	Add a prefix to identify fields of the overlay layer. Duplicate field names will be appended a count suffix to avoid collision.
<b>Grid size</b> Optional	GRID_SIZE	[number] Default: Not set	If provided, the input geometries are snapped to a grid of the given size, and the result vertices are computed on that same grid. Requires GEOS 3.9.0 or higher.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Intersection</b>	OUTPUT	[same as input]	Layer containing (parts of) features from the input layer that overlap the overlay layer.

Pythoncode

Algorithm ID: qgis:intersection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Intersection (multiple)

Extracts the overlapping portions of features in the input and all overlay layers.

Features in the output layer are assigned the attributes of the overlapping features from both the input and overlay layers.

**Warnung: Geometry modification only**

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

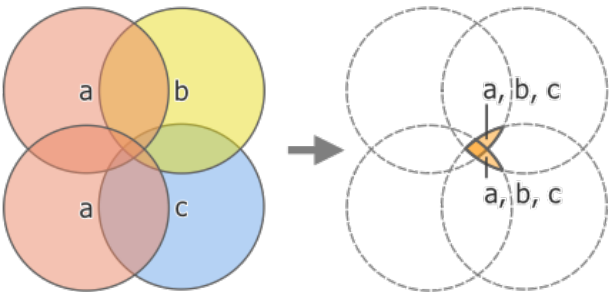


Abb. 28.124: Intersection operation between a two-feature input layer ,a‘ and single feature overlay layers ,b‘ and ,c‘ (left) - overlapping areas become a new two-feature layer with all layers‘ attributes (right)

Siehe auch:

[Intersection](#), [Clip](#), [Difference](#)

Parameter

Basic parameters

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: any]	Layer to extract (parts of) features from.
Overlay layers	OVERLAYS	[vector: any] [list]	Layers containing the features to check for overlap. The features‘ geometry is expected to have at least as many dimensions (point: 0D, line: 1D, polygon: 2D, volume: 3D) as the input layer’s.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Overlay prefix</b> Optional	<b>fields</b> OVERLAY_FIELDS	[string]	Add a prefix to identify fields of the overlay layers. Duplicate field names will be appended a count suffix to avoid collision.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Intersection</b>	OUTPUT	[same as input]	Layer containing (parts of) features from the input layer that overlap all the overlay layers.

## Pythoncode

**Algorithm ID:** qgis:multiintersection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Line intersections

Creates point features where the lines from the two layers intersect.

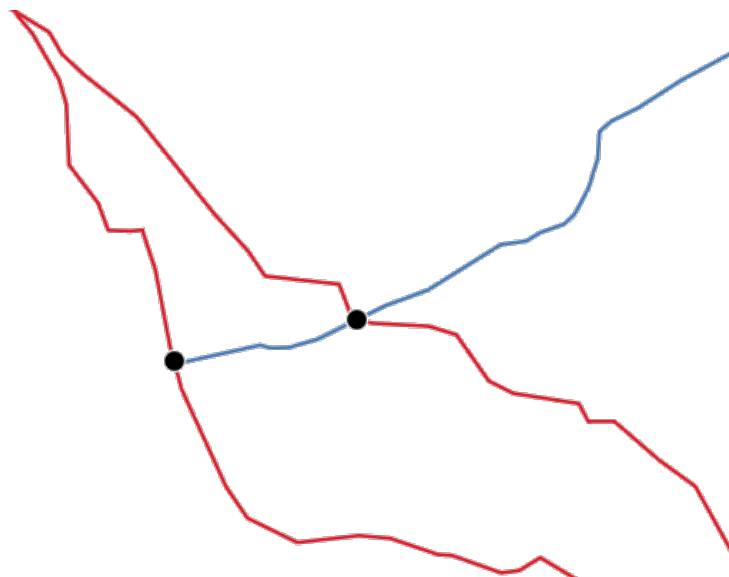


Abb. 28.125: Points of intersection

**Default menu:** *Vector ► Analysis Tools*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	Input line layer.
<b>Intersect layer</b>	INTERSECT	[vector: line]	Layer to use to find line intersections.
<b>Input fields to keep (leave empty to keep all fields)</b> Optional	INPUT_FIELDS	[tablefield: any] [list] Default: None	Field(s) of the input layer to keep in the output. If no fields are chosen all fields are taken.
<b>Intersect fields to keep (leave empty to keep all fields)</b> Optional	INTERSECT_FIEL	[tablefield: any] [list] Default: None	Field(s) of the intersect layer to keep in the output. If no fields are chosen all fields are taken. Duplicate field names will be appended a count suffix to avoid collision.
<b>Intersection</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the layer to contain the intersection points of the lines from the input and overlay layers. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Intersect fields prefix</b> Optional	INTERSECT_FIEL	[string]	Add a prefix to identify fields of the intersect layer.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Intersections</b>	OUTPUT	[vector: point]	Point vector layer of the lines intersections, with both layers' attributes.

## Pythoncode

**Algorithm ID:** qgis:lineintersections

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

Split with lines

Splits the lines or polygons in one layer using the lines or polygon rings in another layer to define the breaking points. Intersection between geometries in both layers are considered as split points.

Output will contain multi geometries for split features.

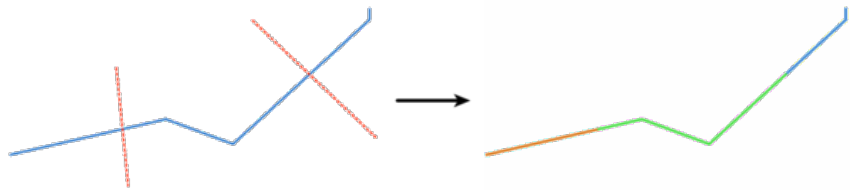


Abb. 28.126: Split lines

 Allows *features in-place modification* of line and polygon features

Parameter

Bezeichnung	Name	Typ	Beschreibung
Input layer	INPUT	[vector: line, polygon]	Layer containing the lines or polygons to split.
Split layer	LINES	[vector: line, polygon]	Layer whose lines or rings are used to define the breaking points.
Split	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the splitted (in case they are intersected by a line in the split layer) line/polygon features from the input layer. One of: <ul style="list-style-type: none"><li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li><li>• Save to File...</li><li>• Save to Geopackage...</li><li>• Save to Database Table...</li></ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Split</b>	OUTPUT	[same as input]	Output vector layer with split lines or polygons from input layer.

## Pythoncode

**Algorithm ID:** qgis:splitwithlines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Symmetrical difference

Creates a layer containing features from both the input and overlay layers but with the overlapping areas between the two layers removed.

The attribute table of the symmetrical difference layer contains attributes and fields from both the input and overlay layers.

### Warnung: Geometry modification only

This operation modifies only the features geometry. The attribute values of the features are **not modified**, although properties such as area or length of the features will be modified by the overlay operation. If such properties are stored as attributes, those attributes will have to be manually updated.

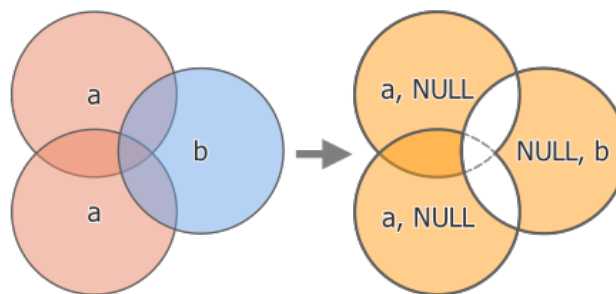


Abb. 28.127: Symmetrical difference operation between a two-features input layer ,a' and a single feature overlay layer ,b' (left) - resulting three-feature layer with both layers' attributes (right)

**Default menu:** *Vector ► Geoprocessing Tools*

**Siehe auch:**

*Difference, Clip, Intersection*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	First layer to extract (parts of) features from.
<b>Overlay layer</b>	OVERLAY	[vector: any]	Second layer to extract (parts of) features from. Ideally the geometry type should be the same as input layer.
<b>Symmetrical difference</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain (the parts of) the features from the input and overlay layers that do not overlap features from the other layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Overlay fields prefix</b> Optional	OVERLAY_FIELDS_	[string]	Add a prefix to identify fields of the overlay layer. Duplicate field names will be appended a count suffix to avoid collision.
<b>Grid size</b> Optional	GRID_SIZE	[number] Default: Not set	If provided, the input geometries are snapped to a grid of the given size, and the result vertices are computed on that same grid. Requires GEOS 3.9.0 or higher.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Symmetrical difference</b>	OUTPUT	[same as input]	Layer containing (parts of) features from each layer not overlapping the other layer, with both layers' attributes.



## Pythoncode

**Algorithm ID:** qgis:symmetricaldifference

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Union

Checks overlaps between features within the input layer and creates separate features for overlapping and non-overlapping parts. The area of overlap will create as many identical overlapping features as there are features that participate in that overlap.

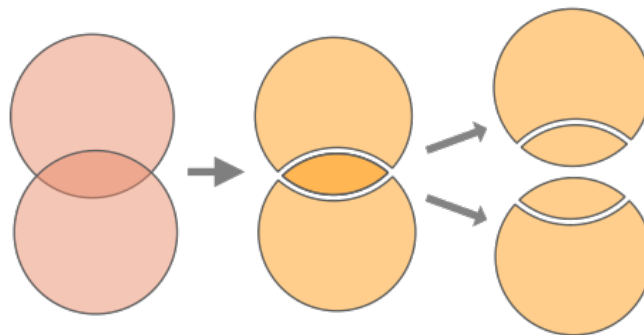


Abb. 28.128: Union operation with a single input layer with two overlapping features (left) - resulting in four features (middle) - features moved for clarity (right)

An overlay layer can also be used, in which case features from each layer are split at their overlap with features from the other one, creating a layer containing all the portions from both input and overlay layers. Features on the same layer will not split each other. The attribute table of the union layer is filled with attribute values from the respective original layer for non-overlapping features, and attribute values from both layers for overlapping features.

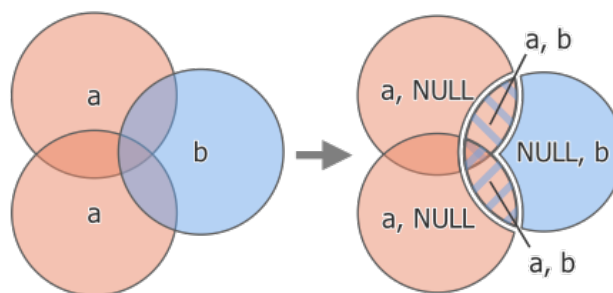


Abb. 28.129: Union operation between a two-feature input layer ,a' and a single feature overlay layer ,b' (left) - resulting five-feature layer with attributes from both layers (right)

**Bemerkung:** With an overlay layer, features on the same layer will not split each other. If you want to split overlaps on the same layer as well as other layers, first run the algorithm with multiple layers then run the algorithm again with only the previous output.

**Default menu:** Vector ► Geoprocessing Tools

**Siehe auch:***Union (multiple), Clip, Difference, Intersection***Parameter****Basic parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer to split at any intersections.
<b>Overlay layer</b> Optional	OVERLAY	[vector: any]	Layer that will be combined to the first one. Ideally the geometry type should be the same as input layer.
<b>Union</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the (split and duplicated) features from the input layer and the overlay layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

**Advanced parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Overlay fields prefix</b> Optional	OVERLAY_FIELDS	[string]	Add a prefix to identify fields of the overlay layer. Duplicate field names will be appended a count suffix to avoid collision.
<b>Grid size</b> Optional	GRID_SIZE	[number] Default: Not set	If provided, the input geometries are snapped to a grid of the given size, and the result vertices are computed on that same grid. Requires GEOS 3.9.0 or higher.

**Ausgaben**

Bezeichnung	Name	Typ	Beschreibung
<b>Union</b>	OUTPUT	[same as input]	Layer containing all the overlapping and non-overlapping parts from the processed layer(s).

## Pythoncode

**Algorithm ID:** qgis:union

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Union (multiple)

Checks overlaps between features within the input layer and creates separate features for overlapping and non-overlapping parts. The area of overlap will create as many identical overlapping features as there are features that participate in that overlap.

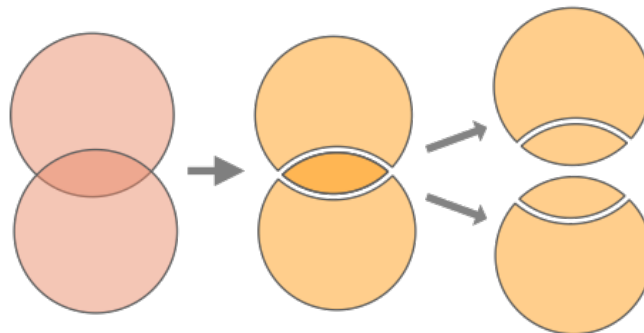


Abb. 28.130: Union operation with a single input layer with two overlapping features (left) - resulting in four features (middle) - features moved for clarity (right)

Multiple overlay layers can also be used, in which case features from each layer are split at their overlap with features from all other layers, creating a layer containing all the portions from both input and overlay layers. Features on the same layer will not split each other. The attribute table of the Union layer is filled with attribute values from the respective original layer for non-overlapping features, and attribute values from overlay layers for overlapping features.

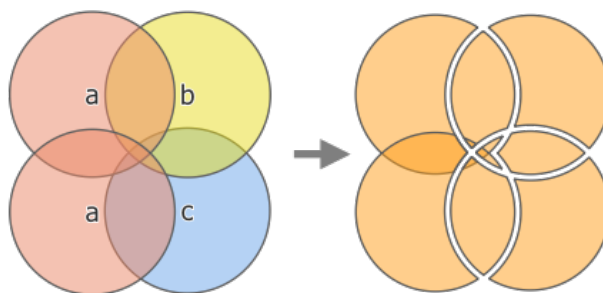


Abb. 28.131: Union operation between a two-feature input layer ,a' and single feature overlay layers ,b' and ,c' (left) - resulting eleven-feature layer with attributes from all layers (right)

**Bemerkung:** With an overlay layer, features on the same layer will not split each other. If you want to split overlaps on the same layer as well as other layers, first run the algorithm with multiple layers then run the algorithm again with only the previous output.

**Siehe auch:**

*Union, Clip, Difference, Intersection*

**Parameter**

**Basic parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer to split at any intersections.
<b>Overlay layers</b> Optional	OVERLAYS	[vector: any] [list]	Layers that will be combined to the first one. Ideally the geometry type should be the same as input layer.
<b>Union</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the layer to contain the (split and duplicated) features from the input layer and the overlay layers. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

**Advanced parameters**

Bezeichnung	Name	Typ	Beschreibung
<b>Overlay fields prefix</b> Optional	OVERLAY_FIELDS_	[string]	Add a prefix to identify fields of the overlay layers. Duplicate field names will be appended a count suffix to avoid collision.

**Ausgaben**

Bezeichnung	Name	Typ	Beschreibung
<b>Union</b>	OUTPUT	[same as input]	Layer containing all the overlapping and non-overlapping parts from the processed layer(s), with all layers' attributes.

## Pythoncode

**Algorithm ID:** qgis:multiunion

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.24 Vector selection

### Extract by attribute

Creates two vector layers from an input layer: one will contain only matching features while the second will contain all the non-matching features.

The criteria for adding features to the resulting layer is based on the values of an attribute from the input layer.

**Siehe auch:**

[Select by attribute](#)

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to extract features from.
<b>Selection attribute</b>	FIELD	[tablefield: any]	Filtering field of the layer
<b>Operator</b>	OPERATOR	[enumeration] Default: 0	Many different operators are available: <ul style="list-style-type: none"> <li>0 — =</li> <li>1 — ≠</li> <li>2 — &gt;</li> <li>3 — &gt;=</li> <li>4 — &lt;</li> <li>5 — &lt;=</li> <li>6 — begins with</li> <li>7 — contains</li> <li>8 — is null</li> <li>9 — is not null</li> <li>10 — does not contain</li> </ul>
<b>Value</b> Optional	VALUE	[string]	Value to be evaluated
<b>Extracted (attribute)</b>	OUTPUT	[same as input] Default: [Create Temporary Layer]	Specify the output vector layer for matching features. One of: <ul style="list-style-type: none"> <li>Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>Save to File...</li> <li>Save to Geopackage...</li> <li>Save to Database Table...</li> </ul> The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 28.188 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (non-matching)</b>	FAIL_OUTPUT	[same as input] Default: [Skip output]	Specify the output vector layer for non-matching features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (attribute)</b>	OUTPUT	[same as input]	Vector layer with matching features from the input layer
<b>Extracted (non-matching)</b>	FAIL_OUTPUT	[same as input]	Vector layer with non-matching features from the input layer

## Pythoncode

**Algorithm ID:** qgis:extractbyattribute

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract by expression

Creates two vector layers from an input layer: one will contain only matching features while the second will contain all the non-matching features.

The criteria for adding features to the resulting layer is based on a QGIS expression. For more information about expressions see the [Ausdrücke](#).

**Siehe auch:**

[Select by expression](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Expression</b>	EXPRESSION	[expression]	Expression to filter the vector layer
<b>Matching features</b>	OUTPUT	[same as input] Default: [Create Temporary Layer]	Specify the output vector layer for matching features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Non-matching</b>	FAIL_OUTPUT	[same as input] Default: [Skip output]	Specify the output vector layer for non-matching features. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Matching features</b>	OUTPUT	[same as input]	Vector layer with matching features from the input layer
<b>Non-matching</b>	FAIL_OUTPUT	[same as input]	Vector layer with non-matching features from the input layer

## Pythoncode

**Algorithm ID:** qgis:extractbyexpression

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract by location

Creates a new vector layer that only contains matching features from an input layer.

The criteria for adding features to the resulting layer is based on the spatial relationship between each feature and the features in an additional layer.

**Siehe auch:**

*Auswahl nach der Lage, Extract within distance*

## Exploring spatial relations

Geometrische Prädikate sind boolesche Funktionen, die verwendet werden, um die räumliche Beziehung eines Features zu einem anderen zu bestimmen, indem verglichen wird, ob und wie ihre Geometrien einen Teil des Raums teilen.

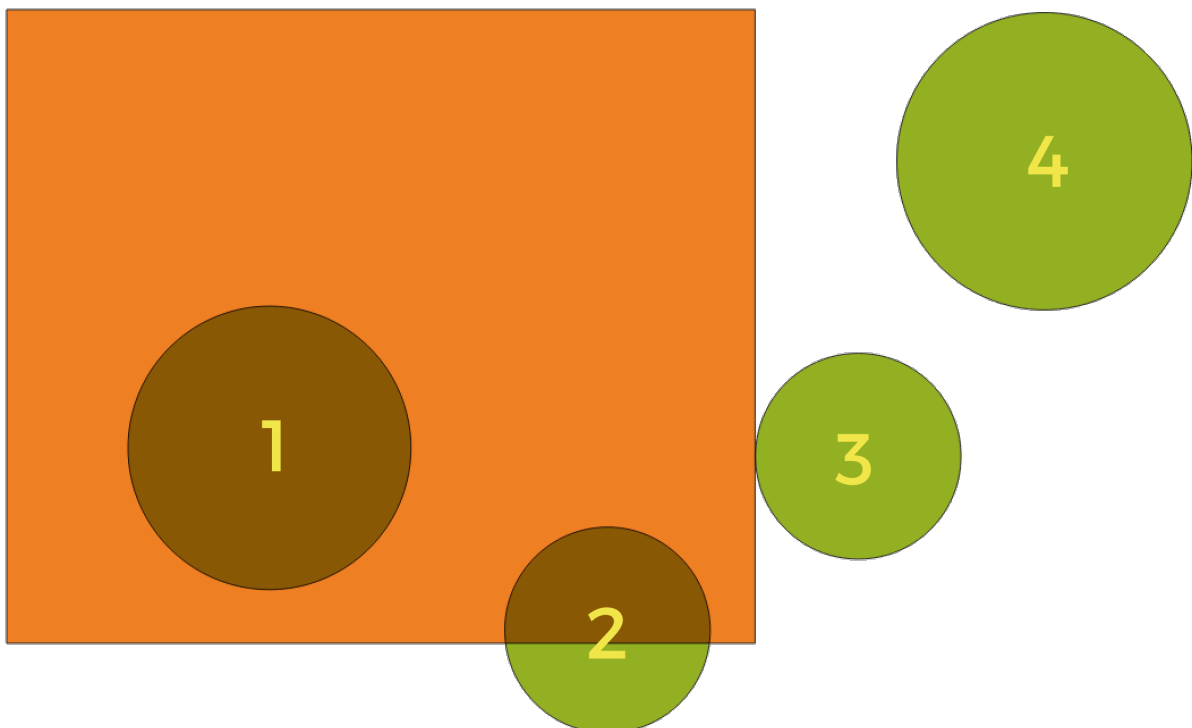


Abb. 28.132: Suche nach räumlichen Beziehungen zwischen Layern

Unter Verwendung der obigen Abbildung suchen wir nach den grünen Kreisen, indem wir sie räumlich mit dem orangefarbenen Rechteck-Feature vergleichen. Verfügbare geometrische Prädikate sind:

### **Schneidet**

Testet, ob eine Geometrie eine andere schneidet. Gibt 1 (wahr) zurück, wenn sich die Geometrien räumlich überschneiden (einen Teil des Raums teilen – überlappen oder berühren) und 0, wenn dies nicht der Fall ist. Im obigen Bild werden die Kreise 1, 2 und 3 zurückgegeben.

### **Enthält**

Gibt 1 (wahr) zurück, wenn und nur wenn keine Punkte von b außerhalb von a liegen und mindestens ein Punkt des Inneren von b innerhalb von a liegt. Im Bild wird kein Kreis zurückgegeben, sondern das Rechteck wäre es, wenn man es umgekehrt suchen würde, da es den Kreis 1 komplett enthält. Das ist das Gegenteil von *sind innerhalb*.

### **Getrennt**

Gibt 1 (wahr) zurück, wenn die Geometrien keinen Teil des Raums teilen (keine Überlappung, keine



Berührung). Nur Kreis 4 wird zurückgegeben.

#### **Gleich**

Gibt 1 (wahr) zurück, wenn und nur wenn die Geometrien genau gleich sind. Es werden keine Kreise zurückgegeben.

#### **Berührt**

Testet, ob eine Geometrie eine andere berührt. Gibt 1 (wahr) zurück, wenn die Geometrien mindestens einen Punkt gemeinsam haben, aber ihre Innenräume sich nicht schneiden. Nur Kreis 3 wird zurückgegeben.

#### **Überlappt**

Testet, ob eine Geometrie eine andere überlappt. Gibt 1 (wahr) zurück, wenn die Geometrien sich den Raum teilen, dieselbe Dimension haben, aber nicht vollständig ineinander liegen. Nur Kreis 2 wird zurückgegeben.

#### **Sind innerhalb**

Testet, ob sich eine Geometrie in einer anderen befindet. Gibt 1 (wahr) zurück, wenn sich Geometrie a vollständig innerhalb von Geometrie b befindet. Nur Kreis 1 wird zurückgegeben.

#### **Kreuzen**

Gibt 1 (wahr) zurück, wenn die bereitgestellten Geometrien einige, aber nicht alle inneren Punkte gemeinsam haben und die tatsächliche Kreuzung eine geringere Dimension als die höchste bereitgestellte Geometrie hat. Beispielsweise wird eine Linie, die ein Polygon kreuzt, als Linie gekreuzt (wahr). Zwei Linien, die sich kreuzen, kreuzen sich als Punkt (wahr). Zwei Polygone kreuzen sich als Polygon (falsch). Im Bild werden keine Kreise zurückgegeben.

### **Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Extract features from</b>	INPUT	[vector: any]	Input vector layer
<b>Where the features (geometric predicate)</b>	PREDICATE	[enumeration] [list] Default: [0]	Type of spatial relation the input feature should have with an intersect feature so that it could be selected. One or more of: <ul style="list-style-type: none"> <li>• 0 — intersect</li> <li>• 1 — contain</li> <li>• 2 — disjoint</li> <li>• 3 — equal</li> <li>• 4 — touch</li> <li>• 5 — overlap</li> <li>• 6 — are within</li> <li>• 7 — cross</li> </ul> <p>If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.</p>
<b>By comparing to the features from</b>	INTERSECT	[vector: any]	Intersection vector layer
<b>Extracted (location)</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the features that have the chosen spatial relationship(s) with one or more features in the comparison layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> <p>The file encoding can also be changed here.</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (location)</b>	OUTPUT	[same as input]	Vector layer with features from the input layer that have the chosen spatial relationship(s) with one or more features in the comparison layer.

## Pythoncode

**Algorithm ID:** qgis:extractbylocation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract within distance

Creates a new vector layer that only contains matching features from an input layer. Features are copied wherever they are within the specified maximum distance from the features in an additional reference layer.

**Siehe auch:**

*Select within distance, Extract by location*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Extract features from</b>	INPUT	[vector: any]	Input vector layer to copy features from
<b>By comparing to the features from</b>	REFERENCE	[vector: any]	Vector layer whose features closeness is used
<b>Where the features are within</b>	DISTANCE	[number] Default: 100	The maximum distance around reference features to select input features within
<b>Modify current selection by</b>	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: <ul style="list-style-type: none"> <li>• 0 — creating new selection</li> <li>• 1 — adding to current selection</li> <li>• 2 — selecting within current selection</li> <li>• 3 — removing from current selection</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.190 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (location)</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the features that are within the set distance from reference features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (location)</b>	OUTPUT	[same as input]	Vector layer with features from the input layer matching the condition of distance from reference features

## Pythoncode

**Algorithm ID:** native:extractwithindistance

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Filter by geometry type

Filters features by their geometry type. Incoming features will be directed to different outputs based on whether they have a point, line or polygon geometry.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Layer to evaluate

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Point features</b> Optional	POINTS	[vector: point]	Layer with points
<b>Line features</b> Optional	LINES	[vector: line]	Layer with lines
<b>Polygon features</b> Optional	POLYGONS	[vector: polygon]	Layer with polygons
<b>Features with no geometry</b> Optional	NO_GEOMETRY	[table]	Geometry-less vector layer

## Pythoncode

**Algorithm ID:** native:filterbygeometry

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random extract

Takes a vector layer and generates a new one that contains only a subset of the features in the input layer.

The subset is defined randomly, based on feature IDs, using a percentage or count value to define the total number of features in the subset.

**Siehe auch:**

[Random selection](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Source vector layer to select the features from
<b>Method</b>	METHOD	[enumeration] Default: 0	Random selection methods. One of: <ul style="list-style-type: none"> <li>0 — Number of selected features</li> <li>1 — Percentage of selected features</li> </ul>
<b>Number/percentage of selected features</b>	NUMBER	[number] Default: 10	Number or percentage of features to select

Fortsetzung auf der nächsten Seite

Tab. 28.191 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (random)</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specify the output vector layer for the randomly selected features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (random)</b>	OUTPUT	[same as input]	Vector layer containing randomly selected features from the input layer

## Pythoncode

**Algorithm ID:** `qgis:randomextract`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random extract within subsets

Takes a vector layer and generates a new one that contains only a subset of the features in the input layer.

The subset is defined randomly, based on feature IDs, using a percentage or count value to define the total number of features in the subset. The percentage/count value is not applied to the whole layer, but instead to each category. Categories are defined according to a given attribute.

**Siehe auch:**

[Random selection within subsets](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer to select the features from
<b>ID field</b>	FIELD	[tablefield: any]	Category of the source vector layer to select the features from
<b>Method</b>	METHOD	[enumeration] Default: 0	Random selection method. One of: <ul style="list-style-type: none"> <li>• 0 — Number of selected features</li> <li>• 1 — Percentage of selected features</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.192 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Number/percentage of selected features</b>	NUMBER	[number] Default: 10	Number or percentage of features to select
<b>Extracted (random stratified)</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer for the randomly selected features. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Extracted (random stratified)</b>	OUTPUT	[same as input]	Vector layer containing randomly selected features from the input layer

## Pythoncode

**Algorithm ID:** qgis:randomextractwithinsubsets

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random selection

Takes a vector layer and selects a subset of its features. No new layer is generated by this algorithm.

The subset is defined randomly, based on feature IDs, using a percentage or count value to define the total number of features in the subset.

**Default menu:** *Vector ► Research Tools*

**Siehe auch:**

*Random extract*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer for the selection
<b>Method</b>	METHOD	[enumeration] Default: 0	Random selection method. One of: <ul style="list-style-type: none"> <li>• 0 — Number of selected features</li> <li>• 1 — Percentage of selected features</li> </ul>
<b>Number/percentage of selected features</b>	NUMBER	[number] Default: 10	Number or percentage of features to select

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** qgis:randomselection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Random selection within subsets

Takes a vector layer and selects a subset of its features. No new layer is generated by this algorithm.

The subset is defined randomly, based on feature IDs, using a percentage or count value to define the total number of features in the subset.

The percentage/count value is not applied to the whole layer, but instead to each category.

Categories are defined according to a given attribute, which is also specified as an input parameter for the algorithm.

No new outputs are created.

**Default menu:** *Vector ► Research Tools*

**Siehe auch:**

[Random extract within subsets](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer to select features in
<b>ID field</b>	FIELD	[tablefield: any]	Category of the input layer to select the features from
<b>Method</b>	METHOD	[enumeration] Default: 0	Random selection method. One of: <ul style="list-style-type: none"> <li>0 — Number of selected features</li> <li>1 — Percentage of selected features</li> </ul>
<b>Number/percentage of selected features</b>	NUMBER	[number] Default: 10	Number or percentage of features to select

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** qgis:randomselectionwithinsubsets

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Select by attribute

Erstellt eine Auswahl in einem Vektorlayer.

The criteria for selecting features is based on the values of an attribute from the input layer.

**Siehe auch:**

*Extract by attribute*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Vector layer to select features in
<b>Selection attribute</b>	FIELD	[tablefield: any]	Filtering field of the layer

Fortsetzung auf der nächsten Seite



Tab. 28.193 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Operator</b>	OPERATOR	[enumeration] Default: 0	Many different operators are available: <ul style="list-style-type: none"> <li>• 0 — =</li> <li>• 1 — ≠</li> <li>• 2 — &gt;</li> <li>• 3 — &gt;=</li> <li>• 4 — &lt;</li> <li>• 5 — &lt;=</li> <li>• 6 — begins with</li> <li>• 7 — contains</li> <li>• 8 — is null</li> <li>• 9 — is not null</li> <li>• 10 — does not contain</li> </ul>
<b>Value</b> Optional	VALUE	[string]	Value to be evaluated
<b>Modify current selection by</b>	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: <ul style="list-style-type: none"> <li>• 0 — creating new selection</li> <li>• 1 — adding to current selection</li> <li>• 2 — removing from current selection</li> <li>• 3 — selecting within current selection</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** qgis:selectbyattribute

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Select by expression

Erstellt eine Auswahl in einem Vektorlayer.

The criteria for selecting features is based on a QGIS expression. For more information about expressions see the [Ausdrücke](#).

**Siehe auch:**

[Extract by expression](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Expression</b>	EXPRESSION	[expression]	Expression to filter the input layer
<b>Modify current selection by</b>	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: <ul style="list-style-type: none"><li>• 0 — creating new selection</li><li>• 1 — adding to current selection</li><li>• 2 — removing from current selection</li><li>• 3 — selecting within current selection</li></ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** qgis:selectbyexpression

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Auswahl nach der Lage

Erstellt eine Auswahl in einem Vektorlayer.

Die Kriterien für die Auswahl von Features basieren auf der räumlichen Beziehung zwischen jedem Feature und den Features in eines zusätzlichen Layers.

**Default menu:** *Vector ► Research Tools*

**Siehe auch:**

[Extract by location](#), [Select within distance](#)

## Exploring spatial relations

Geometrische Prädikate sind boolesche Funktionen, die verwendet werden, um die räumliche Beziehung eines Features zu einem anderen zu bestimmen, indem verglichen wird, ob und wie ihre Geometrien einen Teil des Raums teilen.

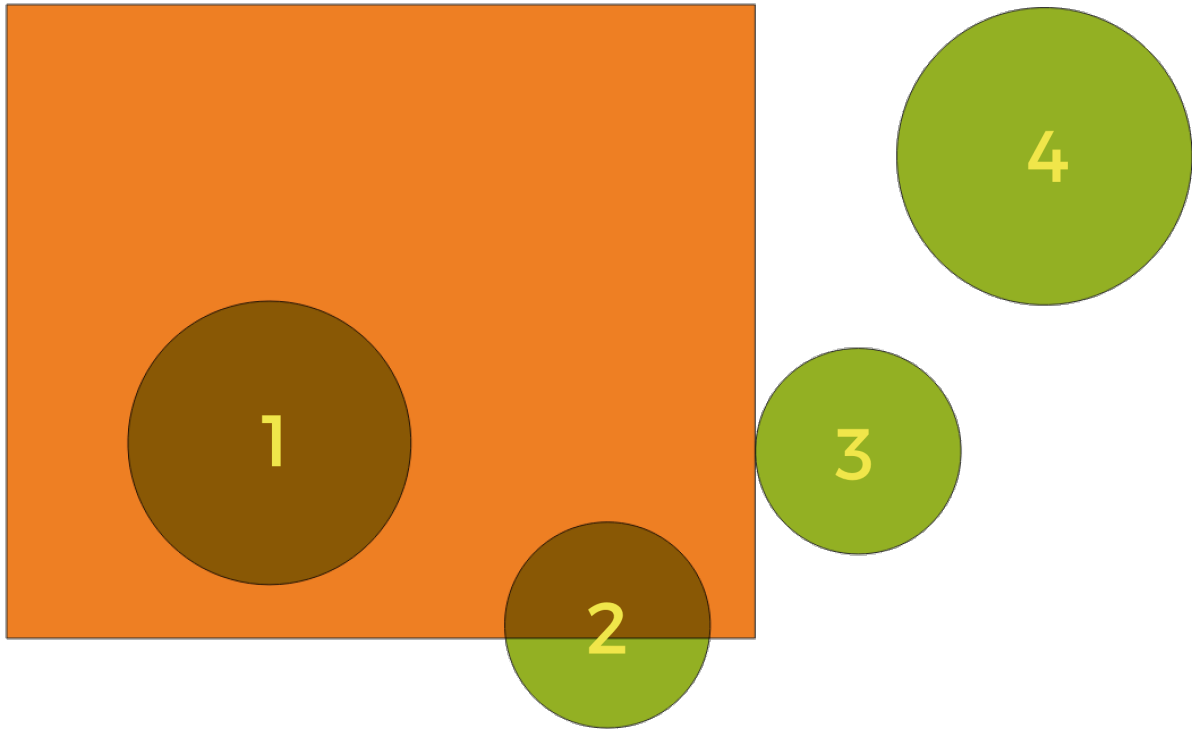


Abb. 28.133: Suche nach räumlichen Beziehungen zwischen Layern

Unter Verwendung der obigen Abbildung suchen wir nach den grünen Kreisen, indem wir sie räumlich mit dem orangefarbenen Rechteck-Feature vergleichen. Verfügbare geometrische Prädikate sind:

### **Schneidet**

Testet, ob eine Geometrie eine andere schneidet. Gibt 1 (wahr) zurück, wenn sich die Geometrien räumlich überschneiden (einen Teil des Raums teilen – überlappen oder berühren) und 0, wenn dies nicht der Fall ist. Im obigen Bild werden die Kreise 1, 2 und 3 zurückgegeben.

### **Enthält**

Gibt 1 (wahr) zurück, wenn und nur wenn keine Punkte von b außerhalb von a liegen und mindestens ein Punkt des Inneren von b innerhalb von a liegt. Im Bild wird kein Kreis zurückgegeben, sondern das Rechteck wäre es, wenn man es umgekehrt suchen würde, da es den Kreis 1 komplett enthält. Das ist das Gegenteil von *sind innerhalb*.

### **Getrennt**

Gibt 1 (wahr) zurück, wenn die Geometrien keinen Teil des Raums teilen (keine Überlappung, keine Berührung). Nur Kreis 4 wird zurückgegeben.

### **Gleich**

Gibt 1 (wahr) zurück, wenn und nur wenn die Geometrien genau gleich sind. Es werden keine Kreise zurückgegeben.

### **Berührt**

Testet, ob eine Geometrie eine andere berührt. Gibt 1 (wahr) zurück, wenn die Geometrien mindestens einen Punkt gemeinsam haben, aber ihre Innenräume sich nicht schneiden. Nur Kreis 3 wird zurückgegeben.

### **Überlappt**

Testet, ob eine Geometrie eine andere überlappt. Gibt 1 (wahr) zurück, wenn die Geometrien sich den Raum

teilen, dieselbe Dimension haben, aber nicht vollständig ineinander liegen. Nur Kreis 2 wird zurückgegeben.

#### ***Sind innerhalb***

Testet, ob sich eine Geometrie in einer anderen befindet. Gibt 1 (wahr) zurück, wenn sich Geometrie a vollständig innerhalb von Geometrie b befindet. Nur Kreis 1 wird zurückgegeben.

#### ***Kreuzen***

Gibt 1 (wahr) zurück, wenn die bereitgestellten Geometrien einige, aber nicht alle inneren Punkte gemeinsam haben und die tatsächliche Kreuzung eine geringere Dimension als die höchste bereitgestellte Geometrie hat. Beispielsweise wird eine Linie, die ein Polygon kreuzt, als Linie gekreuzt (wahr). Zwei Linien, die sich kreuzen, kreuzen sich als Punkt (wahr). Zwei Polygone kreuzen sich als Polygon (falsch). Im Bild werden keine Kreise zurückgegeben.

### **Parameter**

Bezeichnung	Name	Typ	Beschreibung
<b>Select from Where features (geometric predicate)</b>	INPUT	[vector: any]	Input vector layer
	PREDICATE	[enumeration] [list] Default: [0]	Type of spatial relation the input feature should have with an intersect feature so that it could be selected. One or more of: <ul style="list-style-type: none"> <li>• 0 — intersect</li> <li>• 1 — contain</li> <li>• 2 — disjoint</li> <li>• 3 — equal</li> <li>• 4 — touch</li> <li>• 5 — overlap</li> <li>• 6 — are within</li> <li>• 7 — cross</li> </ul> If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.
<b>By comparing to the features from</b>	INTERSECT	[vector: any]	Intersection vector layer
<b>Modify current selection by</b>	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: <ul style="list-style-type: none"> <li>• 0 — creating new selection</li> <li>• 1 — adding to current selection</li> <li>• 2 — selecting within current selection</li> <li>• 3 — removing from current selection</li> </ul>

### **Ausgaben**

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** qgis:selectbylocation

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Select within distance

creates a selection in a vector layer. Features are selected wherever they are within the specified maximum distance from the features in an additional reference layer.

**Siehe auch:**

*Extract within distance, Auswahl nach der Lage*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Select features from</b>	INPUT	[vector: any]	Input vector layer to select features from
<b>By comparing to the features from</b>	REFERENCE	[vector: any]	Vector layer whose features closeness is used
<b>Where the features are within</b>	DISTANCE	[number] Default: 100	The maximum distance around reference features to select input features
<b>Modify current selection by</b>	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: <ul style="list-style-type: none"> <li>• 0 — creating new selection</li> <li>• 1 — adding to current selection</li> <li>• 2 — selecting within current selection</li> <li>• 3 — removing from current selection</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[same as input]	The input layer with features selected

## Pythoncode

**Algorithm ID:** native:selectwithindistance

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.25 Vector table

### Add autoincremental field

Adds a new integer field to a vector layer, with a sequential value for each feature.

This field can be used as a unique ID for features in the layer. The new attribute is not added to the input layer but a new layer is generated instead.

The initial starting value for the incremental series can be specified. Optionally, the incremental series can be based on grouping fields and a sort order for features can also be specified.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer.
<b>Field name</b>	FIELD_NAME	[string] Default: 'AUTO'	Name of the field with autoincremental values
<b>Start values at</b> Optional	START	[number] Default: 0	Choose the initial number of the incremental count
<b>Modulus value</b> Optional	MODULUS	[number] Default: 0	Specifying an optional modulus value will restart the count to START whenever the field value reaches the modulus value. 0 means no restart.
<b>Group values by</b> Optional	GROUP_FIELDS	[tablefield: any] [list]	Select grouping field(s): instead of a single count run for the whole layer, a separate count is processed for each value returned by the combination of these fields.
<b>Sort expression</b> Optional	SORT_EXPRESSION	[expression]	Use an expression to sort the features in the layer either globally or if set, based on group fields.
<b>Sort ascending</b>	SORT_ASCENDING	[boolean] Default: True	When a sort expression is set, use this option to control the order in which features are assigned values.
<b>Sort nulls first</b>	SORT_NULLS_FIRST	[boolean] Default: False	When a sort expression is set, use this option to set whether <i>Null</i> values are counted first or last.

Fortsetzung auf der nächsten Seite

Tab. 28.196 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Incremented</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer with the auto increment field. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Incremented</b>	OUTPUT	[same as input]	Vector layer with auto incremental field

## Pythoncode

**Algorithm ID:** native:addautoincrementalfield

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Add field to attributes table

Adds a new field to a vector layer.

The name and characteristics of the attribute are defined as parameters.

The new attribute is not added to the input layer but a new layer is generated instead.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer
<b>Field name</b>	FIELD_NAME	[string]	Name of the new field
<b>Field type</b>	FIELD_TYPE	[enumeration] Default: 0	Type of the new field. You can choose between: <ul style="list-style-type: none"> <li>• 0 — Integer</li> <li>• 1 — Float</li> <li>• 2 — String</li> </ul>
<b>Field length</b>	FIELD_LENGTH	[number] Default: 10	Length of the field

Fortsetzung auf der nächsten Seite

Tab. 28.197 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Field precision</b>	FIELD_PRECISION	[number] Default: 0	Precision of the field. Useful with Float field type.
<b>Field alias</b> NEW in 3.32 Optional	FIELD_ALIAS	[string]	Set a name to use as alias for the field. Not supported by all format types.
<b>Field comment</b> NEW in 3.32 Optional	FIELD_COMMENT	[string]	Store a comment describing the field. Not supported by all format types.
<b>Added</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Added</b>	OUTPUT	[same as input]	Vector layer with new field added

## Pythoncode

**Algorithm ID:** native:addfieldtoattributetable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Add unique value index field

Takes a vector layer and an attribute and adds a new numeric field.

Values in this field correspond to values in the specified attribute, so features with the same value for the attribute will have the same value in the new numeric field.

This creates a numeric equivalent of the specified attribute, which defines the same classes.

The new attribute is not added to the input layer but a new layer is generated instead.



## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer.
<b>Class field</b>	FIELD	[tablefield: any]	Features that have the same value for this field will get the same index.
<b>Output field name</b>	FIELD_NAME	[string] Default: 'NUM_FIELD'	Name of the new field containing the indexes.
<b>Layer with index field</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Vector layer with the numeric field containing indexes. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.
<b>Class summary</b>	SUMMARY_OUTPUT	[table] Default: [Skip output]	Specify the table to contain the summary of the class field mapped to the corresponding unique value. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer with index field</b>	OUTPUT	[same as input]	Vector layer with the numeric field containing indexes.
<b>Class summary</b>	SUMMARY_OUTPUT	[table]	Table with summary of the class field mapped to the corresponding unique value.

## Pythoncode

**Algorithm ID:** native:adduniquevalueindexfield

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Add X/Y fields to layer

Adds X and Y (or latitude/longitude) fields to a point layer. The X/Y fields can be calculated in a different CRS to the layer (e.g. creating latitude/longitude fields for a layer in a projected CRS).



Allows *features in-place modification* of point features

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: point]	The input layer.
<b>Coordinate system</b>	CRS	[crs] Default: „EPSG:4326“	Coordinate reference system to use for the generated x and y fields.
<b>Field prefix</b> Optional	PREFIX	[string]	Prefix to add to the new field names to avoid name collisions with fields in the input layer.
<b>Added fields</b>	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Added fields</b>	OUTPUT	[vector: point]	The output layer - identical to the input layer but with two new double fields, x and y.

### Pythoncode

**Algorithm ID:** native:addxyfieldstolayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Advanced Python field calculator

Adds a new attribute to a vector layer, with values resulting from applying an expression to each feature.

The expression is defined as a Python function.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Result field name</b>	FIELD_NAME	[string] Default: „NewField“	Name of the new field
<b>Field type</b>	FIELD_TYPE	[enumeration] Default: 0	Type of the new field. One of: <ul style="list-style-type: none"> <li>• 0 — Integer</li> <li>• 1 — Float</li> <li>• 2 — String</li> </ul>
<b>Field length</b>	FIELD_LENGTH	[number] Default: 10	Length of the field
<b>Field precision</b>	FIELD_PRECISIO	[number] Default: 3	Precision of the field. Useful with Float field type.
<b>Global expression</b> Optional	GLOBAL	[string]	The code in the global expression section will be executed only once before the calculator starts iterating through all the features of the input layer. Therefore, this is the correct place to import necessary modules or to calculate variables that will be used in subsequent calculations.
<b>Formula</b>	FORMULA	[string]	The Python formula to evaluate. Example: To calculate the area of an input polygon layer you can add: <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> value = \$geom.area() </div>
<b>Calculated</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the vector layer with the new calculated field. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[same as input]	Vector layer with the new calculated field

## Pythoncode

**Algorithm ID:** qgis:advancedpythonfieldcalculator

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Drop field(s)

Takes a vector layer and generates a new one that has the same features but without the selected columns.

**Siehe auch:**

[Retain fields](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer to drop field(s) from
<b>Fields to drop</b>	COLUMN	[tablefield: any] [list]	The field(s) to drop
<b>Remaining fields</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer with the remaining fields. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Remaining fields</b>	OUTPUT	[same as input]	Vector layer with the remaining fields

## Pythoncode

**Algorithm ID:** native:deletecolumn

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Explode HStore Field

Creates a copy of the input layer and adds a new field for every unique key in the HStore field.

The expected field list is an optional comma separated list. If this list is specified, only these fields are added and the HStore field is updated. By default, all unique keys are added.

The PostgreSQL [HStore](#) is a simple key-value store used in PostgreSQL and GDAL (when reading an [OSM file](#) with the `other_tags` field).

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>HStore field</b>	FIELD	[tablefield: any]	The field(s) to drop
<b>Expected list of fields separated by a comma</b> Optional	EXPECTED_FIELD	[string] Default: ;	Comma-separated list of fields to extract. The HStore field will be updated by removing these keys.
<b>Exploded</b>	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> </ul> The file encoding can also be changed here.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Exploded</b>	OUTPUT	[same as input]	Output vector layer

### Pythoncode

**Algorithm ID:** native:explodehstorefield

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Extract binary field

Extracts contents from a binary field, saving them to individual files. Filenames can be generated using values taken from an attribute in the source table or based on a more complex expression.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer containing the binary data
<b>Binary field</b>	FIELD	[tablefield: any]	Field containing the binary data
<b>File name</b>	FILENAME	[expression]	Field or expression-based text to name each output file
<b>Destination folder</b>	FOLDER	[folder] Default: [Save to temporary folder]	Folder in which to store the output files. One of: <ul style="list-style-type: none"><li>• Save to a Temporary Directory</li><li>• Save to Directory</li></ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Folder</b>	FOLDER	[folder]	The folder that contains the output files.

### Pythoncode

**Algorithm ID:** native:extractbinary

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Field calculator

Opens the field calculator (see [Ausdrücke](#)). You can use all the supported expressions and functions.

A new layer is created with the result of the expression.

The field calculator is very useful when used in [The model designer](#).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The layer to calculate on
<b>Output field name</b>	FIELD_NAME	[string]	The name of the field for the results
<b>Output field type</b>	FIELD_TYPE	[enumeration] Default: 0	The type of the field. One of: <ul style="list-style-type: none"> <li>• 0 — Float</li> <li>• 1 — Integer</li> <li>• 2 — String</li> <li>• 3 — Date</li> </ul>
<b>Output field width</b>	FIELD_LENGTH	[number] Default: 10	The length of the result field (minimum 0)
<b>Field precision</b>	FIELD_PRECISIO	[number] Default: 3	The precision of the result field (minimum 0, maximum 15)
<b>Create new field</b>	NEW_FIELD	[boolean] Default: True	Should the result field be a new field
<b>Formula</b>	FORMULA	[expression]	The formula to use to calculate the result
<b>Output file</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specification of the output layer. <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[vector: any]	Output layer with the calculated field values

## Pythoncode

**Algorithm ID:** native:fieldcalculator

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Refactor fields

Allows editing the structure of the attribute table of a vector layer.

Fields can be modified in their type and name, using a fields mapping.

The original layer is not modified. A new layer is generated, which contains a modified attribute table, according to the provided fields mapping.

---

**Bemerkung:** When using a template layer with *constraints* on fields, the information is displayed in the widget with a coloured background and tooltip. Treat this information as a hint during configuration. No constraints will be added on an output layer nor will they be checked or enforced by the algorithm.

---

The Refactor fields algorithm allows to:

- Change field names and types
- Add and remove fields
- Reorder fields
- Calculate new fields based on expressions
- Load field list from another layer



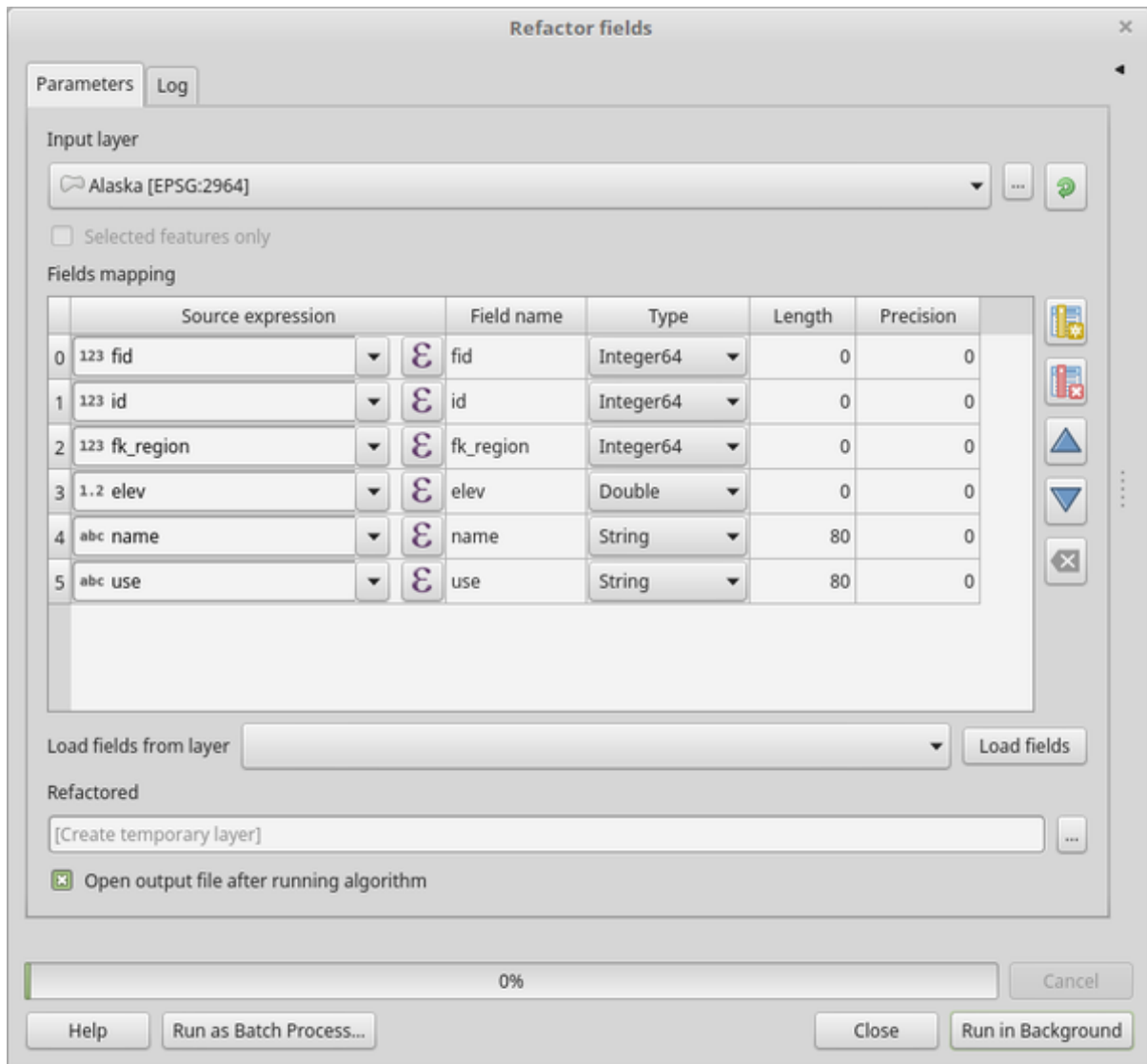








Abb. 28.134: Refactor fields dialog

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The layer to modify

Fortsetzung auf der nächsten Seite

Tab. 28.203 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Fields mapping</b>	FIELDS_MAPPING	[list]	<p>List of output fields with their definitions. The embedded table lists all the fields of the source layer and allows you to edit them:</p> <ul style="list-style-type: none"> <li>Click  to create a new field.</li> <li>Click  to remove a field.</li> <li>Use  and  to change the selected field order.</li> <li>Click  to reset to the default view.</li> <li>Click  to invert the selection in the fields list.</li> </ul> <p>For each of the fields you'd like to reuse, you need to fill the following options:</p> <p><b>Source expression (expression)</b> [expression] Field or expression from the input layer.</p> <p><b>Field name (name) [string]</b> Name of the field in the output layer. By default input field name is kept.</p> <p><b>Type (type) [enumeration]</b> Data type of the output field. Available types depend on the output layer provider.</p> <p><b>Length (length) [number]</b> Length of the output field.</p> <p><b>Precision (precision) [number]</b> Precision of the output field.</p> <p><b>Constraints (constraints) [string]</b> When using a template layer, indicates whether there are constraints applied to the template field. Hover over the cell to display the constraints.</p> <p><b>Field alias (field_alias) [string]</b> <b>NEW in 3.32</b> Set a name to use as alias for the field. Not supported by all format types. Existing aliases are displayed and will be copied to the destination layer if supported.</p> <p><b>Field comment (field_comment) [string]</b> <b>NEW in 3.32</b> Store a comment describing the field. Not supported by all format types. Existing comments are displayed and will be copied to the destination layer if supported.</p> <p><b>Load fields from template layer</b> Allows to select a layer from the current project as a template and (with <i>Load fields</i>) fill the above „Fields mapping“ options with its fields and their definitions.</p>

Fortsetzung auf der nächsten Seite

Tab. 28.203 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Refactored</b>	OUTPUT	[vector: any] Default: [Create temporary layer]	Specification of the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Refactored</b>	OUTPUT	[vector: any]	Output layer with refactored fields

## Pythoncode

**Algorithm ID:** native:refactorfields

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rename field

Renames an existing field from a vector layer.

The original layer is not modified. A new layer is generated where the attribute table contains the renamed field.

**Siehe auch:**

[Refactor fields](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Field to rename</b>	FIELD	[tablefield: any]	The field to be altered
<b>New field name</b>	NEW_NAME	[string]	The new field name

Fortsetzung auf der nächsten Seite

Tab. 28.204 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Renamed</b>	OUTPUT	[vector: same as input] Default: [Create temporary layer]	Specification of the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Renamed</b>	OUTPUT	[vector: same as input]	Output layer with the renamed field

## Pythoncode

**Algorithm ID:** `qgis:renametablefield`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Retain fields

Takes a vector layer and generates a new one that retains only the selected fields. All other fields will be dropped.

**Siehe auch:**

*Drop field(s)*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Fields to retain</b>	FIELDS	[tablefield: any] [list]	List of fields to keep in the layer
<b>Retained fields</b>	OUTPUT	[vector: same as input] Default: [Create temporary layer]	Specification of the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Retained fields</b>	OUTPUT	[vector: same as input]	Output layer with the retained fields

## Pythoncode

**Algorithm ID:** native:retainfields

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Text to float

Modifies the type of a given attribute in a vector layer, converting a text attribute containing numeric strings into a numeric attribute (e.g. '1' to 1.0).

The algorithm creates a new vector layer so the source one is not modified.

If the conversion is not possible the selected column will have NULL values.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer.
<b>Text attribute to convert to float</b>	FIELD	[tablefield: string]	The string field for the input layer that is to be converted to a float field.
<b>Float from text</b>	OUTPUT	[same as input] Default: [Create Temporary Layer]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Create Temporary Layer (TEMPORARY_OUTPUT)</li> <li>• Save to File...</li> <li>• Save to Geopackage...</li> <li>• Save to Database Table...</li> <li>• Append to Layer...</li> </ul> The file encoding can also be changed here.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Float from text</b>	OUTPUT	[same as input]	Output vector layer with the string field converted into a float field

## Pythoncode

**Algorithm ID:** qgis:texttfloat

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.1.26 Vector Tiles

### Download vector tiles

NEW in 3.32

Downloads vector tiles of an input vector tile layer and saves them in a local vector tile file.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector tiles]	A vector tile layer to extract some tiles from
<b>Extent</b>	EXTENT	[extent]	Specify the spatial extent of the downloaded area. It will internally be extended to a multiple of the tile size. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Maximum zoom level to download</b>	MAX_ZOOM	[number] Default: 10	Defines how far to zoom in and fetch data from the tiles
<b>Tile limit</b>	TILE_LIMIT	[number] Default: 100	Maximum number of tiles to download, considering the zoom levels and the extent.
<b>Output</b>	OUTPUT	[vector tiles] Default: [Save to temporary file]	Specification of the output vector tile file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output</b>	OUTPUT	[vector tiles]	A local vector tile file storing the downloaded tiles.

## Pythoncode

**Algorithm ID:** native:downloadvectortiles

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Write vector tiles (MBTiles)

Exports one or more vector layers to vector tiles, a data format optimized for fast map rendering and small data size.

MBTiles is a specification for storing tiled map data in SQLite databases for immediate usage and for transfer. MBTiles files are known as tilesets.

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[vector: any] [list]	A list of layers to combine to generate the vector tiles
<b>Minimum zoom level</b>	MIN_ZOOM	[number] Default: 0	The lowest zoom level for which the tileset provides data. Set between 0 and 24.
<b>Maximum zoom level</b>	MAX_ZOOM	[number] Default: 3	The highest zoom level for which the tileset provides data. Set between 0 and 24.
<b>Extent</b> Optional	EXTENT	[extent] Default: Not set	The maximum extent of the rendered map area. Bounds must define an area covered by all zoom levels.
<b>Metadata: Name</b> Optional	META_NAME	[string]	Name of the tileset
<b>Metadata: Description</b> Optional	META_DESCRIPTION	[string]	A description of the tileset's contents
<b>Metadata: Attribution</b> Optional	META_ATTRIBUTION	[string]	An attribution string, which explains the sources of data and/or style for the map.
<b>Metadata: Version</b> Optional	META_VERSION	[string]	The version of the tileset. This refers to a revision of the tileset itself, not of the MBTiles specification.
<b>Metadata: Type</b> Optional	META_TYPE	[string]	Type of tileset. Possible values are <code>overlay</code> or <code>baselayer</code> .
<b>Metadata: Center</b> Optional	META_CENTER	[string]	The center (string of comma-separated numbers: the longitude, latitude, and zoom level) of the default view of the map. Example: <code>-122.1906, 37.7599, 11</code>
<b>Destination MBTiles</b>	OUTPUT	[vector tiles] Default: [Save to temporary file]	Specification of the output MBTiles file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Destination MBTiles</b>	OUTPUT	[file]	Output vector tiles <code>.mbtiles</code> file.

## Pythoncode

**Algorithm ID:** `native:writevectortiles_mbtiles`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## Write vector tiles (XYZ)

Exports one or more vector layers to vector tiles, a data format optimized for fast map rendering and small data size.

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>File template</b>	XYZ_TEMPLATE	[string] Default: ,{z}/{x}/{y}.pbf	Template to generate the vector tiles url
<b>Input layers</b>	INPUT	[vector: any] [list]	A list of layers to combine to generate the vector tiles
<b>Minimum zoom level</b>	MIN_ZOOM	[number] Default: 0	The lowest zoom level for which the tileset provides data. Set between 0 and 24.
<b>Maximum zoom level</b>	MAX_ZOOM	[number] Default: 3	The highest zoom level for which the tileset provides data. Set between 0 and 24.
<b>Extent</b> Optional	EXTENT	[extent] Default: Not set	The maximum extent of the rendered map area. Bounds must define an area covered by all zoom levels.
<b>Output directory</b>	OUTPUT_DIRECTO	[folder] Default: [Save to temporary folder]	Specification of the output vector tiles folder. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b>	OUTPUT_DIRECTO	[folder]	A folder containing different subsets of the vector tiles files (.pbf) stored in subfolders corresponding to the zoom levels.

### Pythoncode

**Algorithm ID:** native:writevectortiles\_xyz

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2 GDAL-Algorithmen Bereitstellung

**GDAL** (Geospatial Data Abstraction Library) ist eine Übersetzerbibliothek für Raster- und Vektor-Geodatenformate. Die Algorithmen im Verarbeitungsrahmen sind von den **GDAL-Rasterprogrammen** und **GDAL-Vektorprogrammen** abgeleitet.

### 28.2.1 Raster analysis

#### Aspect

Generates an aspect map from any GDAL-supported elevation raster. Aspect is the compass direction that a slope faces. The pixels will have a value from 0-360° measured in degrees from north indicating the azimuth. On the northern hemisphere, the north side of slopes is often shaded (small azimuth from 0°-90°), while the southern side receives more solar radiation (higher azimuth from 180°-270°).

This algorithm is derived from the **GDAL DEM utility**.

**Default menu:** *Raster ► Analysis*

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use as elevation
<b>Return trigonometric angle instead of azimuth</b>	TRIG_ANGLE	[boolean] Default: False	Activating the trigonometric angle results in different categories: 0° (East), 90° (North), 180° (West), 270° (South).
<b>Return 0 for flat instead of -9999</b>	ZERO_FLAT	[boolean] Default: False	Activating this option will insert a 0-value for the value -9999 on flat areas.
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Use Zevenbergen&amp;Thorn formula instead of the Horn's one</b>	ZEVENBERGEN	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
<b>Aspect</b>	OUTPUT	[raster] Default: [Save to temporary file]	Output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Aspect</b>	OUTPUT	[raster]	Output raster with angle values in degrees

## Pythoncode

**Algorithm ID:** gdal:aspect

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Color relief

Generates a color relief map from any GDAL-supported elevation raster. Color reliefs can particularly be used to depict elevations. The Algorithm outputs a 4-band raster with values computed from the elevation and a text-based color configuration file. By default, the colors between the given elevation values are blended smoothly and the result is a nice colorized elevation raster.

This algorithm is derived from the [GDAL DEM utility](#).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use as elevation
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Color configuration file</b>	COLOR_TABLE	[file]	A text-based color configuration file
<b>Matching mode</b>	MATCH_MODE	[enumeration] Default: 2	One of: <ul style="list-style-type: none"> <li>• 0 — Use strict color matching</li> <li>• 1 — Use closest RGBA quadruples</li> <li>• 2 — Use smoothly blended colours</li> </ul>
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
<b>Color relief</b>	OUTPUT	[raster] Default: [Save to temporary file]	Output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Color relief</b>	OUTPUT	[raster]	A 4-band output raster

## Pythoncode

**Algorithm ID:** gdal:colorrelief

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Fill NoData

Fill raster regions with NoData values by interpolation from edges. The values for the NoData regions are calculated by the surrounding pixel values using inverse distance weighting. After the interpolation a smoothing of the results takes place. Input can be any GDAL-supported raster layer. This algorithm is generally suitable for interpolating missing regions of fairly continuously varying rasters (such as elevation models for instance). It is also suitable for filling small holes and cracks in more irregularly varying images (like airphotos). It is generally not so great for interpolating a raster from sparse point data.

This algorithm is derived from the [GDAL fillnodata utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The band to operate on. NoData values must be represented by the value 0.
<b>Maximum distance (in pixels) to search out for values to interpolate</b>	DISTANCE	[number] Default: 10	The number of pixels to search in all directions to find values to interpolate from
<b>Number of smoothing iterations to run after the interpolation</b>	ITERATIONS	[number] Default: 0	The number of 3x3 filter passes to run (0 or more) to smoothen the results of the interpolation.
<b>Do not use default validity mask for the input band</b>	NO_MASK	[boolean] Default: False	Activates the user-defined validity mask
<b>Validity mask</b>	MASK_LAYER	[raster]	A raster layer that defines the areas to fill.
<b>Filled</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Filled</b>	OUTPUT	[raster]	Output raster

## Pythoncode

**Algorithm ID:** gdal:fillnodata

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (Data metrics)

Computes some data metrics using the specified window and output grid geometry.

This algorithm is derived from the [GDAL grid utility](#).

**Default menu:** *Raster ► Analysis*

**Siehe auch:**

[GDAL grid tutorial](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Data metric to use</b>	METRIC	[enumeration] Default: 0	One of: <ul style="list-style-type: none"> <li>• 0 — Minimum, minimum value found in grid node search ellipse</li> <li>• 1 — Maximum, maximum value found in grid node search ellipse</li> <li>• 2 — Range, a difference between the minimum and maximum values found in grid node search ellipse</li> <li>• 3 — Count, a number of data points found in grid node search ellipse</li> <li>• 4 — Average distance, an average distance between the grid node (center of the search ellipse) and all of the data points found in grid node search ellipse</li> <li>• 5 — Average distance between points, an average distance between the data points found in grid node search ellipse. The distance between each pair of points within ellipse is calculated and average of all distances is set as a grid node value</li> </ul>
<b>The first radius of search ellipse</b>	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
<b>The second radius of search ellipse</b>	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
<b>Angle of search ellipse rotation in degrees (counter clockwise)</b>	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
<b>Minimum number of data points to use</b>	MIN_POINTS	[number] Default: 0.0	Minimum number of data points to average. If less amount of points found the grid node considered empty and will be filled with NoData marker.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (data metrics)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <a href="#">Help ► About menu</a>)</p>



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (data metrics)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:griddatametrics

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (IDW with nearest neighbor searching)

Computes the Inverse Distance to a Power gridding combined to the nearest neighbor method. Ideal when a maximum number of data points to use is required.

This algorithm is derived from the [GDAL grid utility](#).

**Siehe auch:**

[GDAL grid tutorial](#)

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Weighting power</b>	POWER	[number] Default: 2.0	Weighting power
<b>Smoothing</b>	SMOOTHING	[number] Default: 0.0	Smoothing parameter
<b>The radius of the search circle</b>	RADIUS	[number] Default: 1.0	The radius of the search circle
<b>Maximum number of data points to use</b>	MAX_POINTS	[number] Default: 12	Do not search for more points than this number.
<b>Minimum number of data points to use</b>	MIN_POINTS	[number] Default: 0	Minimum number of data points to average. If less amount of points found the grid node considered empty and will be filled with NoData marker.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (IDW with NN search)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <a href="#">Help ► About menu</a>)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (IDW with NN search)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:gridinversedistancenearestneighbor

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (Inverse distance to a power)

The Inverse Distance to a Power gridding method is a weighted average interpolator.

You should supply the input arrays with the scattered data values including coordinates of every data point and output grid geometry. The function will compute interpolated value for the given position in output grid.

This algorithm is derived from the [GDAL grid utility](#).

**Default menu:** *Raster ► Analysis*

**Siehe auch:**

[GDAL grid tutorial](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Weighting power</b>	POWER	[number] Default: 2.0	Weighting power
<b>Smoothing</b>	SMOOTHING	[number] Default: 0.0	Smoothing parameter
<b>The first radius of search ellipse</b>	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
<b>The second radius of search ellipse</b>	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
<b>Angle of search ellipse rotation in degrees (counter clockwise)</b>	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.

Fortsetzung auf der nächsten Seite

Tab. 28.215 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Maximum number of data points to use</b>	MAX_POINTS	[number] Default: 0	Do not search for more points than this number.
<b>Minimum number of data points to use</b>	MIN_POINTS	[number] Default: 0	Minimum number of data points to average. If less amount of points found the grid node considered empty and will be filled with NoData marker.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (IDW)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 28.216 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (IDW)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:gridinversedistance

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (Linear)

The Linear method perform linear interpolation by computing a Delaunay triangulation of the point cloud, finding in which triangle of the triangulation the point is, and by doing linear interpolation from its barycentric coordinates within the triangle. If the point is not in any triangle, depending on the radius, the algorithm will use the value of the nearest point or the NoData value.

This algorithm is derived from the [GDAL grid utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>Search distance</b>	RADIUS	[number] Default: -1.0	In case the point to be interpolated does not fit into a triangle of the Delaunay triangulation, use that maximum distance to search a nearest neighbour, or use NoData otherwise. If set to -1, the search distance is infinite. If set to 0, NoData value will be used.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (Linear)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 28.218 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (Linear)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:gridlinear

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (Moving average)

The Moving Average is a simple data averaging algorithm. It uses a moving window of elliptic form to search values and averages all data points within the window. Search ellipse can be rotated by specified angle, the center of ellipse located at the grid node. Also the minimum number of data points to average can be set, if there are not enough points in window, the grid node considered empty and will be filled with specified NoData value.

This algorithm is derived from the [GDAL grid utility](#).

**Default menu:** *Raster ► Analysis*

**Siehe auch:**

[GDAL grid tutorial](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>The first radius of search ellipse</b>	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
<b>The second radius of search ellipse</b>	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
<b>Angle of search ellipse rotation in degrees (counter clockwise)</b>	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
<b>Minimum number of data points to use</b>	MIN_POINTS	[number] Default: 0.0	Minimum number of data points to average. If less amount of points found the grid node considered empty and will be filled with NoData marker.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (moving average)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).

Fortsetzung auf der nächsten Seite



Tab. 28.220 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (moving average)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:gridaverage

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Grid (Nearest neighbor)

The Nearest Neighbor method doesn't perform any interpolation or smoothing, it just takes the value of nearest point found in grid node search ellipse and returns it as a result. If there are no points found, the specified NoData value will be returned.

This algorithm is derived from the [GDAL grid utility](#).

**Default menu:** *Raster ► Analysis*

**Siehe auch:**

[GDAL grid tutorial](#)

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Point layer</b>	INPUT	[vector: point]	Input point vector layer
<b>The first radius of search ellipse</b>	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
<b>The second radius of search ellipse</b>	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
<b>Angle of search ellipse rotation in degrees (counter clockwise)</b>	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
<b>NoData</b>	NODATA	[number] Default: 0.0	NoData marker to fill empty points
<b>Interpolated (Nearest neighbour)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Z value from field</b> Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 28.222 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Interpolated (Nearest neighbour)</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:gridnearestneighbor

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Hillshade

Outputs a raster with a nice shaded relief effect. It's very useful for visualizing the terrain. You can optionally specify the azimuth and altitude of the light source, a vertical exaggeration factor and a scaling factor to account for differences between vertical and horizontal units.

This algorithm is derived from the [GDAL DEM utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input Elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	Band containing the elevation information
<b>Z factor (vertical exaggeration)</b>	Z_FACTOR	[number] Default: 1.0	The factor exaggerates the height of the output elevation raster
<b>Scale (ratio of vert. units to horiz.)</b>	SCALE	[number] Default: 1.0	The ratio of vertical units to horizontal units
<b>Azimuth of the light</b>	AZIMUTH	[number] Default: 315.0	Defines the azimuth of the light shining on the elevation raster in degrees. If it comes from the top of the raster the value is 0, if it comes from the east it is 90 a.s.o.
<b>Altitude of the light</b>	ALTITUDE	[number] Default: 45.0	Defines the altitude of the light, in degrees. 90 if the light comes from above the elevation raster, 0 if it is raking light.
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Use Zevenbergen&amp;Thorn formula (instead of the Horn's one)</b>	ZEVENBERGEN	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
<b>Combined shading</b>	COMBINED	[boolean] Default: False	
<b>Multidirectional shading</b>	MULTIDIRECTION	[boolean] Default: False	
<b>Hillshade</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Hillshade</b>	OUTPUT	[raster]	Output raster with interpolated values

## Pythoncode

**Algorithm ID:** gdal:hillshade

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Near black

Converts nearly black/white borders to black.

This algorithm will scan an image and try to set all pixels that are nearly or exactly black, white or one or more custom colors around the collar to black or white. This is often used to „fix up“ lossy compressed airphotos so that color pixels can be treated as transparent when mosaicking.

This algorithm is derived from the [GDAL nearblack utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input Elevation raster layer
<b>How far from black (white)</b>	NEAR	[number] Default: 15	Select how far from black, white or custom colors the pixel values can be and still considered near black, white or custom color.
<b>Search for nearly white pixels instead of nearly black</b>	WHITE	[boolean] Default: False	Search for nearly white (255) pixels instead of nearly black pixels
<b>Nearblack</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Nearblack</b>	OUTPUT	[raster]	Output raster

## Pythoncode

**Algorithm ID:** gdal:nearblack

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Proximity (raster distance)

Generates a raster proximity map indicating the distance from the center of each pixel to the center of the nearest pixel identified as a target pixel. Target pixels are those in the source raster for which the raster pixel value is in the set of target pixel values.

This algorithm is derived from the [GDAL proximity utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input Elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	Band containing the elevation information
<b>A list of pixel values in the source image to be considered target pixels</b> Optional	VALUES	[string] Default: ‘	A list of target pixel values in the source image to be considered target pixels. If not specified, all non-zero pixels will be considered target pixels.
<b>Distance units</b>	UNITS	[enumeration] Default: 1	Indicate whether distances generated should be in pixel or georeferenced coordinates. One of: <ul style="list-style-type: none"> <li>• 0 — Georeferenced coordinates</li> <li>• 1 — Pixel coordinates</li> </ul>
<b>The maximum distance to be generated</b> Optional	MAX_DISTANCE	[number] Default: 0.0	The maximum distance to be generated. The NoData value will be used for pixels beyond this distance. If a NoData value is not provided, the output band will be queried for its NoData value. If the output band does not have a NoData value, then the value 65535 will be used. Distance is interpreted according to the value of <i>Distance units</i> .

Fortsetzung auf der nächsten Seite

Tab. 28.227 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Value to be applied to all pixels that are within the maxdist of target pixels</b> Optional	REPLACE	[number] Default: 0.0	Specify a value to be applied to all pixels that are closer than the maximum distance from target pixels (including the target pixels) instead of a distance value.
<b>NoData value to use for the destination proximity raster</b> Optional	NODATA	[number] Default: 0.0	Specify the NoData value to use for the output raster
<b>Proximity map</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite



Tab. 28.228 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Proximity map</b>	OUTPUT	[raster]	Output raster

## Pythoncode

**Algorithm ID:** gdal:proximity

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Roughness

Outputs a single-band raster with values computed from the elevation. Roughness is the degree of irregularity of the surface. It's calculated by the largest inter-cell difference of a central pixel and its surrounding cell. The determination of the roughness plays a role in the analysis of terrain elevation data, it's useful for calculations of the river morphology, in climatology and physical geography in general.

This algorithm is derived from the [GDAL DEM utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use as elevation
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Roughness</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Roughness</b>	OUTPUT	[raster]	Single-band output roughness raster. The value -9999 is used as NoData value.

## Pythoncode

**Algorithm ID:** gdal:roughness

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Sieve

Removes raster polygons smaller than a provided threshold size (in pixels) and replaces them with the pixel value of the largest neighbour polygon. It is useful if you have a large amount of small areas on your raster map.

This algorithm is derived from the [GDAL sieve utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Threshold</b>	THRESHOLD	[number] Default: 10	Only raster polygons smaller than this size will be removed
<b>Use 8-connectedness</b>	EIGHT_CONNECTE	[boolean] Default: False	Use eight connectedness instead of four connectedness
<b>Do not use the default validity mask for the input band</b>	NO_MASK	[boolean] Default: False	
<b>Validity mask</b> Optional	MASK_LAYER	[raster]	Validity mask to use instead of the default
<b>Sieved</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Sieved</b>	OUTPUT	[raster]	Output raster layer.

## Pythoncode

**Algorithm ID:** gdal:sieve

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Slope

Generates a slope map from any GDAL-supported elevation raster. Slope is the angle of inclination to the horizontal. You have the option of specifying the type of slope value you want: degrees or percent slope.

This algorithm is derived from the [GDAL DEM utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input Elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	Band containing the elevation information
<b>Ratio of vertical units to horizontal</b>	SCALE	[number] Default: 1.0	The ratio of vertical units to horizontal units
<b>Slope expressed as percent (instead of degrees)</b>	AS_PERCENT	[boolean] Default: False	Express slope as percent instead of degrees
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Use Zevenbergen&amp;Thorn formula (instead of the Horn's one)</b>	ZEVENBERGEN	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
<b>Slope</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Slope</b>	OUTPUT	[raster]	Output raster

## Pythoncode

**Algorithm ID:** gdal:slope

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Terrain Ruggedness Index (TRI)

Outputs a single-band raster with values computed from the elevation. TRI stands for Terrain Ruggedness Index, which is defined as the mean difference between a central pixel and its surrounding cells.

This algorithm is derived from the [GDAL DEM utility](#).

**Default menu:** *Raster ► Analysis*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer

Fortsetzung auf der nächsten Seite

Tab. 28.235 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use as elevation
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Terrain Ruggedness Index</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Terrain Ruggedness Index</b>	OUTPUT	[raster]	Output ruggedness raster. The value -9999 is used as NoData value.

### Pythoncode

**Algorithm ID:** gdal:triterrainruggednessindex

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Topographic Position Index (TPI)

Outputs a single-band raster with values computed from the elevation. TPI stands for Topographic Position Index, which is defined as the difference between a central pixel and the mean of its surrounding cells.

This algorithm is derived from the [GDAL DEM utility](#).

**Default menu:** *Raster ► Analysis*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use for elevation values
<b>Compute edges</b>	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
<b>Terrain Ruggedness Index</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Terrain Ruggedness Index</b>	OUTPUT	[raster]	Output raster.

## Pythoncode

**Algorithm ID:** gdal:tpitopographicpositionindex

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.2 Raster conversion

### gdal2xyz

Converts raster data to XYZ ASCII file format.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Raster layer to convert
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to convert
<b>Source NoData</b> Optional	NODATA_INPUT	[number] Default: None	Input pixel value to treat as „NoData“ (GDAL >= 3.7).
<b>Destination NoData</b> Optional	NODATA_OUTPUT	[number] Default: None	Assign specified „NoData“ value to output (GDAL >= 3.7).
<b>Do not output NoData values</b>	SKIP_NODATA	[boolean] Default: False	Do not output „NoData“ values (GDAL >= 3.3).
<b>Output comma-separated values</b>	CSV	[boolean] Default: False	Sets whether the output file should be of type comma-separated values (csv).
<b>XYZ ASCII file</b>	OUTPUT	[file] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>XYZ ASCII file</b>	INPUT	[table]	Table file containing the values exported from the raster band.



## Pythoncode

**Algorithm ID:** gdal:gdal2xyz

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## PCT to RGB

Converts an 8 bit paletted image to a 24 bit RGB. It will convert a pseudocolor band from the input file to an RGB file of the desired format.

This algorithm is derived from the [GDAL pct2rgb utility](#).

**Default menu:** *Raster ► Conversion*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input 8 bit raster image
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to convert
<b>Generate a RGBA file</b>	RGBA	[boolean] Default: False	Sets whether the output file should be of type RGBA.
<b>PCT to RGB</b>	OUTPUT	[file] Default: [Save to temporary file]	Specification of the output file. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>PCT to RGB</b>	OUTPUT	[raster]	24 bit RGB raster image

## Pythoncode

**Algorithm ID:** gdal:pcttorgb

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Polygonize (raster to vector)

Creates vector polygons for all connected regions of pixels in the raster sharing a common pixel value. Each polygon is created with an attribute indicating the pixel value of that polygon.

This algorithm is derived from the [GDAL polygonize utility](#).

**Default menu:** *Raster ► Conversion*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Band number</b>	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to use
<b>Name of the field to create</b>	FIELD	[string] Default: „DN“	Specify the field name for the attributes of the connected regions.
<b>Use 8-connectedness</b>	EIGHT_CONNECTE	[boolean] Default: False	If not set, raster cells must have a common border to be considered connected ( <i>4-connected</i> ). If set, touching raster cells are also considered connected ( <i>8-connected</i> ).
<b>Vectorized</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specification of the output (polygon) vector layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Vectorized</b>	OUTPUT	[vector: polygon]	Output vector layer

## Pythoncode

**Algorithm ID:** gdal:polygonize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rearrange bands

Creates a new raster using selected band(s) from a given raster layer. The algorithm also makes it possible to reorder the bands for the newly-created raster.

This algorithm is derived from the [GDAL translate utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Selected band(s)</b>	BANDS	[raster band] [list] Default: None	Ordered list of the bands to use to create the new raster
<b>Converted</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 0	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Use Input Layer Data Type</li> <li>• 1 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 2 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 3 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 5 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 6 — Float32 (Thirty two bit floating point (float))</li> <li>• 7 — Float64 (Sixty four bit floating point (double))</li> <li>• 8 — CInt16 (Complex Int16)</li> <li>• 9 — CInt32 (Complex Int32)</li> <li>• 10 — CFloat32 (Complex Float32)</li> <li>• 11 — CFloat64 (Complex Float64)</li> <li>• 12 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About menu</i>)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Converted</b>	OUTPUT	[raster]	Output raster layer with rearranged bands.

## Pythoncode

**Algorithm ID:** gdal:rearrange\_bands

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## RGB to PCT

Converts a 24 bit RGB image into a 8 bit paletted. Computes an optimal pseudo-color table for the given RGB-image using a median cut algorithm on a downsampled RGB histogram. Then it converts the image into a pseudo-colored image using the color table. This conversion utilizes Floyd-Steinberg dithering (error diffusion) to maximize output image visual quality.

If you want to classify a raster map and want to reduce the number of classes it can be helpful to downsample your image with this algorithm before.

This algorithm is derived from the GDAL [rgb2pct](#) utility.

**Default menu:** *Raster ► Conversion*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input (RGB) raster layer
<b>Number of colors</b>	NCOLORS	[number] Default: 2	The number of colors the resulting image will contain. A value from 2-256 is possible.
<b>RGB to PCT</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>RGB to PCT</b>	OUTPUT	[raster]	Output raster layer.

## Pythoncode

**Algorithm ID:** gdal:rgbtompct

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Translate (convert format)

Converts raster data between different formats.

This algorithm is derived from the [GDAL translate utility](#).

**Default menu:** *Raster ► Conversion*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Override the projection of the output file</b> Optional	TARGET_CRS	[crs]	Specify a projection for the output file
<b>Assign a specified NoData value to output bands</b> Optional	NODATA	[number] Default: Not set	Defines the value to use for NoData in the output raster
<b>Copy all subdatasets of this file to individual output files</b>	COPY_SUBDATASE	[boolean] Default: False	Create individual files for subdatasets
<b>Converted</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output (translated) raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 0	<p>Defines the data type of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Use Input Layer Data Type</li> <li>• 1 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 2 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 3 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 5 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 6 — Float32 (Thirty two bit floating point (float))</li> <li>• 7 — Float64 (Sixty four bit floating point (double))</li> <li>• 8 — CInt16 (Complex Int16)</li> <li>• 9 — CInt32 (Complex Int32)</li> <li>• 10 — CFloat32 (Complex Float32)</li> <li>• 11 — CFloat64 (Complex Float64)</li> <li>• 12 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About menu</i>)</p>

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Converted</b>	OUTPUT	[raster]	Output (translated) raster layer.

## Pythoncode

**Algorithm ID:** gdal:translate

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.3 Raster extraction

### Clip raster by extent

Clips any GDAL-supported raster file to a given extent.

This algorithm is derived from the [GDAL translate utility](#).

**Default menu:** *Raster ► Extraction*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	The input raster

Fortsetzung auf der nächsten Seite



Tab. 28.245 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Clipping extent</b>	EXTENT	[extent]	<p>Extent that should be used for the output raster. Only pixels within the specified bounding box will be included in the output. Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Override the projection for the output file</b>	OVERCRS	[boolean] Default: False	If checked, the output file is assigned the input layer CRS.
<b>Assign a specified NoData value to output bands</b> Optional	NODATA	[number] Default: None	Defines a value that should be inserted for the NoData values in the output raster
<b>Clipped (extent)</b>	OUTPUT	[raster] Default: [Save to temporary file]	<p>Specification of the output raster layer. One of:</p> <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 0	Defines the format of the output raster file. Optionen: <ul style="list-style-type: none"> <li>• 0 — Use Input Layer Data Type</li> <li>• 1 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 2 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 3 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 5 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 6 — Float32 (Thirty two bit floating point (float))</li> <li>• 7 — Float64 (Sixty four bit floating point (double))</li> <li>• 8 — CInt16 (Complex Int16)</li> <li>• 9 — CInt32 (Complex Int32)</li> <li>• 10 — CFloat32 (Complex Float32)</li> <li>• 11 — CFloat64 (Complex Float64)</li> <li>• 12 — Int8 (Eight bit signed integer (qint8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <i>Help ► About menu</i> )
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clipped (extent)</b>	OUTPUT	[raster]	Output raster layer clipped by the given extent

## Pythoncode

**Algorithm ID:** gdal:cliprasterbyextent

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Clip raster by mask layer

Clips any GDAL-supported raster by a vector mask layer.

This algorithm is derived from the [GDAL warp utility](#).

**Default menu:** *Raster ► Extraction*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	The input raster
<b>Mask layer</b>	MASK	[vector: polygon]	Vector mask for clipping the raster
<b>Source CRS</b>	SOURCE_CRIS	[crs]	Set the coordinate reference to use for the input raster
<b>Target CRS</b>	TARGET_CRIS	[crs]	Set the coordinate reference to use for the mask layer

Fortsetzung auf der nächsten Seite

Tab. 28.246 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Target extent</b> Optional	TARGET_EXTENT	[extent]	Extent of the output file to be created Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Assign a specified NoData value to output bands</b> Optional	NODATA	[number] Default: None	Defines a value that should be inserted for the NoData values in the output raster
<b>Create an output alpha band</b>	ALPHA_BAND	[boolean] Default: False	Creates an alpha band for the result. The alpha band then includes the transparency values of the pixels.
<b>Match the extent of the clipped raster to the extent of the mask layer</b>	CROP_TO_CUTLIN	[boolean] Default: True	Applies the vector layer extent to the output raster if checked.
<b>Keep resolution of input raster</b>	KEEP_RESOLUTION	[boolean] Default: False	The resolution of the output raster will not be changed
<b>Set output file resolution</b>	SET_RESOLUTION	[boolean] Default: False	Shall the output resolution (cell size) be specified
<b>X Resolution to output bands</b> Optional	X_RESOLUTION	[number] Default: None	The width of the cells in the output raster
<b>Y Resolution to output band</b> Optional	Y_RESOLUTION	[number] Default: None	The height of the cells in the output raster
<b>Use multithreaded warping implementation</b>	MULTITHREADING	[boolean] Default: False	Two threads will be used to process chunks of image and perform input/output operation simultaneously. Note that computation is not multithreaded itself.
<b>Clipped (mask)</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 0	Defines the format of the output raster file. Optionen: <ul style="list-style-type: none"> <li>• 0 — Use Input Layer Data Type</li> <li>• 1 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 2 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 3 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 5 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 6 — Float32 (Thirty two bit floating point (float))</li> <li>• 7 — Float64 (Sixty four bit floating point (double))</li> <li>• 8 — CInt16 (Complex Int16)</li> <li>• 9 — CInt32 (Complex Int32)</li> <li>• 10 — CFloat32 (Complex Float32)</li> <li>• 11 — CFloat64 (Complex Float64)</li> <li>• 12 — Int8 (Eight bit signed integer (qint8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <i>Help ► About menu</i> )
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clipped (mask)</b>	OUTPUT	[raster]	Output raster layer clipped by the vector layer

## Pythoncode

**Algorithm ID:** gdal:cliprasterbymasklayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Contour

Extracts contour lines from any GDAL-supported elevation raster.

This algorithm is derived from the [GDAL contour utility](#).

**Default menu:** *Raster ► Extraction*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster
<b>Band number</b>	BAND	[raster band] Default: 1	Raster band to create the contours from
<b>Interval between contour lines</b>	INTERVAL	[number] Default: 10.0	Defines the interval between the contour lines in the given units of the elevation raster (minimum value 0)
<b>Attribute name (if not set, no elevation attribute is attached)</b> Optional	FIELD_NAME	[string] Default: ‚ELEV‘	Provides a name for the attribute in which to put the elevation.
<b>Offset from zero relative to which to interpret intervals</b> Optional	OFFSET	[number] Default: 0.0	
<b>Contours</b>	OUTPUT	[vector: line] Default: [Save to temporary file]	Specification of the output vector layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Produce 3D vector</b>	CREATE_3D	[boolean] Default: False	Forces production of 3D vectors instead of 2D. Includes elevation at every vertex.
<b>Treat all raster values as valid</b>	IGNORE_NODATA	[boolean] Default: False	Ignores any NoData values in the dataset.
<b>Input pixel value to treat as „NoData“</b> Optional	NODATA	[number] Default: None	Defines a value that should be inserted for the NoData values in the output raster
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options. Refer to the corresponding GDAL utility documentation.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Contours</b>	OUTPUT	[vector: line]	Output vector layer with contour lines

## Pythoncode

**Algorithm ID:** gdal:contour

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Contour Polygons

Extracts contour polygons from any GDAL-supported elevation raster.

This algorithm is derived from the [GDAL contour utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster
<b>Band number</b>	BAND	[raster band] Default: 1	Raster band to create the contours from

Fortsetzung auf der nächsten Seite

Tab. 28.248 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Interval between contour lines</b>	INTERVAL	[number] Default: 10.0	Defines the interval between the contour lines in the given units of the elevation raster (minimum value 0)
<b>Offset from zero relative to which to interpret intervals</b> Optional	OFFSET	[number] Default: 0.0	
<b>Attribute name for minimum elevation of contour polygon</b> Optional	FIELD_NAME_MIN	[string] Default: ,ELEV_MIN‘	Provides a name for the attribute in which to put the minimum elevation of contour polygon. If not provided no minimum elevation attribute is attached.
<b>Attribute name for maximum elevation of contour polygon</b> Optional	FIELD_NAME_MAX	[string] Default: ,ELEV_MAX‘	Provides a name for the attribute in which to put the maximum elevation of contour polygon. If not provided no maximum elevation attribute is attached.
<b>Contours</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specification of the output vector layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Produce 3D vector</b>	CREATE_3D	[boolean] Default: False	Forces production of 3D vectors instead of 2D. Includes elevation at every vertex.
<b>Treat all raster values as valid</b>	IGNORE_NODATA	[boolean] Default: False	Ignores any NoData values in the dataset.
<b>Input pixel value to treat as „NoData“</b> Optional	NODATA	[number] Default: None	Defines a value that should be inserted for the NoData values in the output raster
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options. Refer to the corresponding GDAL utility documentation.



## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Contours</b>	OUTPUT	[vector: polygon]	Output vector layer with contour polygons

## Pythoncode

**Algorithm ID:** gdal:contour\_polygon

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.4 Raster miscellaneous

### Build overviews (pyramids)

To speed up rendering time of raster layers overviews (pyramids) can be created. Overviews are lower resolution copies of the data which QGIS uses depending of the level of zoom.

This algorithm is derived from the [GDAL addo utility](#).

**Default menu:** *Raster ► Miscellaneous*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Remove all existing overviews</b>	CLEAN	[boolean] Default: False	Removes existing overviews from the raster. By default these are not removed.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Overview levels</b>	LEVELS	[string] Default: ,2 4 8 16`	Defines the number of overview levels calculated by the original resolution of the input raster layer. By default 4 levels will be taken into consideration.
<b>Resampling method</b> Optional	RESAMPLING	[enumeration] Default: 0	Calculates the overviews with a defined resampling method. Possible resampling methods are: <ul style="list-style-type: none"> <li>• 0 – Nearest Neighbour (nearest)</li> <li>• 1 – Average (average)</li> <li>• 2 – Gaussian (gauss)</li> <li>• 3 – Cubic (4x4 kernel) (cubic)</li> <li>• 4 – Cubic B-Spline (4x4 kernel) (cubicspline)</li> <li>• 5 – Lanczos (6x6 kernel) (lanczos)</li> <li>• 6 – Average MP (average_mp)</li> <li>• 7 – Average in Mag/Phase Space (average_magphase)</li> <li>• 8 – Mode (mode)</li> </ul>
<b>Overviews format</b> Optional	FORMAT	[enumeration] Default: 0	The overviews can be stored internally, or externally as GTiff or ERDAS Imagine file. By default the overviews are stored in the output raster. Possible formats methods are: <ul style="list-style-type: none"> <li>• 0 – Internal (if possible)</li> <li>• 1 – External (GTiff .ovr)</li> <li>• 2 – External (ERDAS Imagine .aux)</li> </ul>
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Pyramidized</b>	OUTPUT	[raster]	Output raster layer with overviews

## Pythoncode

**Algorithm ID:** gdal:overviews

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Build virtual raster

Builds a VRT (Virtual Dataset) that is a mosaic of the list of input GDAL-supported rasters. With a mosaic you can merge several raster files.

This algorithm is derived from the [GDAL buildvrt utility](#).

**Default menu:** *Raster ► Miscellaneous*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	GDAL-supported raster layers.
<b>Resolution</b>	RESOLUTION	[enumeration] Default: 0	The output resolution of the mosaic. By default the average resolution of the raster files will be chosen. Optionen: <ul style="list-style-type: none"> <li>• 0 — Average (average)</li> <li>• 1 — Highest (highest)</li> <li>• 2 — Lowest (lowest)</li> </ul>
<b>Place each input file into a separate band</b>	SEPARATE	[boolean] Default: False	With ,True' you can define that each raster file goes into a separated stacked band in the VRT band.
<b>Allow projection difference</b>	PROJ_DIFFERENC	[boolean] Default: False	Allows that the output bands have different projections derived from the projection of the input raster layers.
<b>Virtual</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Add alpha mask band to VRT when source raster has none</b>	ADD_ALPHA	[boolean] Default: False	Adds an alpha mask band to the VRT when the source raster has none.
<b>Override projection for the output file</b> Optional	ASSIGN_CRS	[crs] Default: None	Overrides the projection for the output file. No reprojection is done.
<b>Resampling algorithm</b>	RESAMPLING	[enumeration] Default: 0	The <a href="#">resampling algorithm</a> to use. Options: <ul style="list-style-type: none"> <li>• 0 — Nearest Neighbour (<i>nearest</i>)</li> <li>• 1 — Bilinear (2x2 kernel) (<i>bilinear</i>)</li> <li>• 2 — Cubic (4x4 kernel) (<i>cubic</i>)</li> <li>• 3 — Cubic B-Spline (4x4 kernel) (<i>cubicspline</i>)</li> <li>• 4 — Lanczos (6x6 kernel) (<i>lanczos</i>)</li> <li>• 5 — Average (<i>average</i>)</li> <li>• 6 — Mode (<i>mode</i>)</li> </ul>
<b>NoData value(s) for input bands (space separated)</b> Optional	SRC_NODATA	[string] Default: None	Space separated NoData value(s) for input band(s)
<b>Additional command-line parameters</b>	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Virtual</b>	OUTPUT	[raster]	Output raster layer

## Pythoncode

**Algorithm ID:** gdal:buildvirtualraster

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## gdal2tiles

Generates a directory with small tiles and metadata, following the [OSGeo Tile Map Service Specification](#). See also the [OpenGIS Web Map Tile Service Implementation Standard](#). Simple web pages with viewers based on Google Maps, OpenLayers and Leaflet are generated as well. To explore your maps on-line in the web browser, you only need to upload the generated directory onto a web server.

This algorithm also creates the necessary metadata for Google Earth (KML SuperOverlay), in case the supplied map uses EPSG:4326 projection.

ESRI world files and embedded georeferencing is used during tile generation, but you can publish a picture without proper georeferencing too.

This algorithm is derived from the [GDAL gdal2tiles utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	GDAL-supported raster layer.
<b>Tile cutting profile</b>	PROFILE	[enumeration] Default: 0	One of: <ul style="list-style-type: none"> <li>• 0 — Mercator (mercator)</li> <li>• 1 — Geodetic (geodetic)</li> <li>• 2 — Raster (raster)</li> </ul>
<b>Zoom levels to render</b> Optional	ZOOM	[string] Default: ;	
<b>Web viewer to generate</b>	VIEWER	[enumerate] Default: 0	One of: <ul style="list-style-type: none"> <li>• 0 — All (all)</li> <li>• 1 — GoogleMaps (google)</li> <li>• 2 — OpenLayers (openlayers)</li> <li>• 3 — Leaflet (leaflet)</li> <li>• 4 — None (none)</li> </ul>
<b>Title of the map</b> Optional	TITLE	[string] Default: ;	
<b>Copyright of the map</b>	COPYRIGHT	[string] Default: ;	
<b>Output directory</b>	OUTPUT	[folder] Default: [Save to temporary folder]	Specify the output folder for the tiles. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Resampling method</b>	RESAMPLING	[enumeration] Default: 0	The resampling algorithm to use. Options: <ul style="list-style-type: none"><li>• 0 — Average (average)</li><li>• 1 — Nearest Neighbour (near)</li><li>• 2 — Bilinear (2x2 kernel) (bilinear)</li><li>• 3 — Cubic (4x4 kernel) (cubic)</li><li>• 4 — Cubic B-Spline (4x4 kernel) (cubicspline)</li><li>• 5 — Lanczos (6x6 kernel) (lanczos)</li><li>• 6 — Antialias (antialias)</li></ul>
<b>The spatial reference system used for the source input data</b> Optional	SOURCE_CR	[crs] Default: None	
<b>Transparency value to assign to the input data</b> Optional	NODATA	[number] Default: 0.0	
<b>URL address where the generated tiles are going to be published</b> Optional	URL	[string] Default: ,	
<b>Google Maps API key</b> (http://code.google.c Optional	GOOGLE_KEY	[string] Default: ,	Your Google maps API key.
<b>Bing Maps API key</b> (https://www.bingm Optional	BING_KEY	[string] Default: ,	Your Bing maps API key.
<b>Generate only missing files</b>	RESUME	[boolean] Default: False	
<b>Generate KML for Google Earth</b>	KML	[boolean] Default: False	
<b>Avoid automatic generation of KML files for EPSG:4326</b>	NO_KML	[boolean] Default: False	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b>	OUTPUT	[folder]	The output folder (for the tiles)

## Pythoncode

**Algorithm ID:** gdal:gdal2tiles

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Merge

Merges raster files in a simple way. Here you can use a pseudocolor table from an input raster and define the output raster type. All the images must be in the same coordinate system.

This algorithm is derived from the [GDAL merge utility](#).

**Default menu:** *Raster ► Miscellaneous*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layers</b>	INPUT	[raster] [list]	Input raster layers
<b>Grab pseudocolor table from first layer</b>	PCT	[boolean] Default: False	The pseudocolor table from the first layer will be used for the coloring
<b>Place each input file into a separate band</b>	SEPARATE	[boolean] Default: False	Place each input file into a separate band

Fortsetzung auf der nächsten Seite

Tab. 28.252 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	<p>Defines the format of the output raster file.</p> <p>Options:</p> <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (quint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (quint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>
<b>Merged</b>	OUTPUT	[raster] Default: [Save to temporary file]	<p>Specification of the output raster layer. One of:</p> <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>



## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input pixel value to treat as „NoData“</b> Optional	NODATA_INPUT	[number] Default: None	Ignores pixels from files being merged in with this pixel value
<b>Assign specified „NoData“ value to output</b> Optional	NODATA_OUTPUT	[number] Default: None	Assigns the specified NoData value to output bands.
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ‘	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b>	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Merged</b>	OUTPUT	[raster]	Output raster layer

## Pythoncode

**Algorithm ID:** gdal:merge

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Pansharpening

Performs a pan-sharpening operation. It can create a „classic“ output dataset (such as GeoTIFF), or a VRT dataset describing the pan-sharpening operation.

See [GDAL Pansharpen](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Spectral dataset</b>	SPECTRAL	[raster]	Input (spectral) raster layer
<b>Panchromatic dataset</b>	PANCHROMATIC	[raster]	Input (panchromatic) raster layer
<b>Ergebnis</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output (sharpened) raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Resampling algorithm</b>	RESAMPLING	[enumeration] Default: 2	The <a href="#">resampling algorithm</a> to use. Options: <ul style="list-style-type: none"> <li>• 0 — Nearest Neighbour (<i>nearest</i>)</li> <li>• 1 — Bilinear (2x2 kernel) (<i>bilinear</i>)</li> <li>• 2 — Cubic (4x4 kernel) (<i>cubic</i>)</li> <li>• 3 — Cubic B-Spline (4x4 kernel) (<i>cubicspline</i>)</li> <li>• 4 — Lanczos (6x6 kernel) (<i>lanczos</i>)</li> <li>• 5 — Average (<i>average</i>)</li> </ul>
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ‘	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[raster]	Output (sharpened) raster layer

## Pythoncode

**Algorithm ID:** gdal:pansharp

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster calculator

Command line raster calculator with numpy syntax. Use any basic arithmetic supported by numpy arrays, such as +, -, \*, and / along with logical operators, such as >. Note that all input rasters must have the same dimensions, but no projection checking is performed.

See the [GDAL Raster Calculator utility docs](#).

**Siehe auch:**

*Raster calculator, Raster calculator (virtual)*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer A</b>	INPUT_A	[raster]	First input raster layer (mandatory)
<b>Number of raster band for A</b>	BAND_A	[raster band]	Band for input layer A (mandatory)
<b>Input layer B</b>	INPUT_B	[raster]	Second input raster layer
Optional		Default: None	
<b>Number of raster band for B</b>	BAND_B	[raster band]	Band for input layer B
Optional			
<b>Input layer C</b>	INPUT_C	[raster]	Third input raster layer
Optional		Default: None	
<b>Number of raster band for C</b>	BAND_C	[raster band]	Band for input layer C
Optional			
<b>Input layer D</b>	INPUT_D	[raster]	Fourth input raster layer
Optional		Default: None	
<b>Number of raster band for D</b>	BAND_D	[raster band]	Band for input layer D
Optional			
<b>Input layer E</b>	INPUT_E	[raster]	Fifth input raster layer
Optional		Default: None	
<b>Number of raster band for E</b>	BAND_E	[raster band]	Band for input layer E
Optional			
<b>Input layer F</b>	INPUT_F	[raster]	Sixth input raster layer
Optional			

Fortsetzung auf der nächsten Seite

Tab. 28.254 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Number of raster band for F</b> Optional	BAND_F	[raster band] Default: None	Band for input layer F
<b>Calculation in gdalnumeric syntax using +/-* or any numpy array functions (i.e. logical_and())</b>	FORMULA	[string] Default: ;	The calculation formula. Examples: <ul style="list-style-type: none"> <li>• <math>A * (A &gt; 0)</math> — outputs the value of the raster A if the value of A is greater than 0. If not, outputs 0.</li> <li>• <math>A * (A &gt; 0 \text{ and } A &gt; B)</math> — outputs the value of A if that value is bigger than 0 and bigger than the value of B. If not, outputs 0.</li> <li>• <math>A * \text{logical\_or}(A \leq 177, A \geq 185)</math> — outputs the value of A if <math>A \leq 177</math> or <math>A \geq 185</math>. If not, outputs 0.</li> <li>• <math>\text{sqrt}(A * A + B * B)</math> — Outputs the square root of the sum of the value of A squared and the value of B squared.</li> </ul>
<b>Set output NoData value</b> Optional	NO_DATA	[number] Default: None	Value to use for NoData
<b>Handling of extent differences</b>	EXTENT_OPT	[enumeration] Default: 0	Determines how to handle rasters with different extents. Only available with GDAL 3.3+. <a href="#">Supported options</a> are: <ul style="list-style-type: none"> <li>• 0 — Ignore</li> <li>• 1 — Fail</li> <li>• 2 — Union</li> <li>• 3 — Intersect</li> </ul>
<b>Output extent</b> Optional	INPUT	[extent]	Custom extent of the output raster. Only available with GDAL 3.3+. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>

Fortsetzung auf der nächsten Seite

Tab. 28.254 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Output raster type</b>	RTYPE	[enumeration] Default: 5	Defines the data type of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)
<b>Calculated</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output (calculated) raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ‘	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: ‘	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Calculated</b>	OUTPUT	[raster]	Output (calculated) raster layer

## Pythoncode

**Algorithm ID:** gdal:rastercalculator

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Raster information

The gdalinfo program lists various information about a GDAL supported raster dataset.

This algorithm is derived from the [GDAL info utility](#).

**Default menu:** *Raster ► Miscellaneous*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer
<b>Force computation of the actual min/max values for each band</b>	MIN_MAX	[boolean] Default: False	Forces computation of the actual min/max values for each band in the dataset
<b>Read and display image statistics (force computation if necessary)</b>	STATS	[boolean] Default: False	Reads and displays image statistics. Forces computation if no statistics are stored in an image.
<b>Suppress GCP info</b>	NO_GCP	[boolean] Default: False	Suppresses ground control points list printing. It may be useful for datasets with huge amount of GCPs, such as L1B AVHRR or HDF4 MODIS which contain thousands of them.
<b>Suppress metadata info</b>	NO_METADATA	[boolean] Default: False	Suppresses metadata printing. Some datasets may contain a lot of metadata strings.

Fortsetzung auf der nächsten Seite

Tab. 28.256 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Layer information</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for output. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer information</b>	OUTPUT	[html]	The HTML file containing information about the input raster layer

### Pythoncode

**Algorithm ID:** gdal:gdalinfo

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Retile

Retiles a set of input tiles. All the input tiles must be georeferenced in the same coordinate system and have a matching number of bands. Optionally pyramid levels are generated.

This algorithm is derived from the [GDAL Retile utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input files</b>	INPUT	[raster] [list]	The input raster files
<b>Tile width</b>	TILE_SIZE_X	[number] Default: 256	Width of the tiles in pixels (minimum 0)
<b>Tile height</b>	TILE_SIZE_Y	[number] Default: 256	Height of the tiles in pixels (minimum 0)
<b>Overlap in pixels between consecutive tiles</b>	OVERLAP	[number] Default: 0	
<b>Number of pyramid levels to build</b>	LEVELS	[number] Default: 1	Minimum: 0
<b>Output directory</b>	OUTPUT	[folder] Default: [Save to temporary folder]	Specify the output folder for the tiles. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary Directory</li> <li>• Save to Directory</li> </ul>
<b>CSV file containing the georeferencing information</b>	OUTPUT_CSV	[file] Default: [Skip output]	Specify the output file for the tiles. One of: <ul style="list-style-type: none"> <li>• Skip Output</li> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Source coordinate reference system</b> Optional	SOURCE_CRSS	[crs] Default: None	
<b>Resampling method</b>	RESAMPLING	[enumeration] Default: 0	The <a href="#">resampling algorithm</a> to use. Options: <ul style="list-style-type: none"> <li>• 0 — Nearest Neighbour (<i>nearest</i>)</li> <li>• 1 — Bilinear (2x2 kernel) (<i>bilinear</i>)</li> <li>• 2 — Cubic (4x4 kernel) (<i>cubic</i>)</li> <li>• 3 — Cubic B-Spline (4x4 kernel) (<i>cubicspline</i>)</li> <li>• 4 — Lanczos (6x6 kernel) (<i>lanczos</i>)</li> </ul>

Fortsetzung auf der nächsten Seite



Tab. 28.259 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Column delimiter used in the CSV file</b> Optional	DELIMITER	[string] Default: ,	Delimiter to use in the CSV file containing the tile(s) georeferencing information
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ,	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: ,	Add extra GDAL command line options
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	Defines the format of the output raster file. Options: <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (uint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (int16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (uint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (uint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (int32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (int8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <a href="#">Help ► About menu</a> )
<b>Build only the pyramids</b>	ONLY_PYRAMIDS	[boolean] Default: False	
<b>Use separate directory for each tile row</b>	DIR_FOR_ROW	[boolean] Default: False	

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Output directory</b>	OUTPUT	[folder]	The output folder for the tiles.
<b>CSV file containing the georeferencing information</b>	OUTPUT_CSV	[file]	The CSV file with georeferencing information for the tiles.

## Pythoncode

Algorithm ID: `gdal:retile`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Tile index

Builds a vector layer with a record for each input raster file, an attribute containing the filename, and a polygon geometry outlining the raster. This output is suitable for use with MapServer as a raster tileindex.

This algorithm is derived from the [GDAL Tile Index](#) utility.

**Default menu:** *Raster ► Miscellaneous*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input files</b>	LAYERS	[raster] [list]	The input raster files. Can be multiple files.
<b>Field name to hold the file path to the indexed rasters</b>	PATH_FIELD_NAME Optional	[string] Default: 'location'	The output field name to hold the file path/location to the indexed rasters.
<b>Store absolute path to the indexed rasters</b>	ABSOLUTE_PATH	[boolean] Default: False	Set whether the absolute path to the raster files is stored in the tile index file. By default the raster filenames will be put in the file exactly as they are specified in the command.
<b>Skip files with different projection reference</b>	PROJ_DIFFERENC	[boolean] Default: False	Only files with same projection as files already inserted in the tile index will be inserted. Default does not check projection and accepts all inputs.

Fortsetzung auf der nächsten Seite

Tab. 28.260 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Tile index</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specify the polygon vector layer to write the index to. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Transform geometries to the given CRS</b> Optional	TARGET_CRS	[crs]	Geometries of input files will be transformed to the specified target coordinate reference system. Default creates simple rectangular polygons in the same coordinate reference system as the input rasters.
<b>The name of the field to store the SRS of each tile</b> Optional	CRS_FIELD_NAME	[string]	The name of the field to store the SRS of each tile
<b>The format in which the CRS of each tile must be written</b>	CRS_FORMAT	[enumeration] Default: 0	Format for the CRS. One of: <ul style="list-style-type: none"> <li>• 0 – Auto (AUTO)</li> <li>• 1 – Well-known text (WKT)</li> <li>• 2 – EPSG (EPSG)</li> <li>• 3 – Proj.4 (PROJ)</li> </ul>

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Tile index</b>	OUTPUT	[vector: polygon]	The polygon vector layer with the tile index.

### Pythoncode

**Algorithm ID:** gdal:tileindex

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Viewshed

Calculates a viewshed raster from an input raster DEM using method defined in [Wang2000](#) for a user defined point.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input elevation raster layer
<b>Band number</b>	BAND	[raster band] Default: 1	The number of the band to use as elevation
<b>Observer location</b>	OBSERVER	[point]	The location of the observer
<b>Observer height</b>	OBSERVER_HEIGH	[number] Default: 1.0	The altitude of the observer, in the DEM units
<b>Target height</b>	TARGET_HEIGHT	[number] Default: 1.0	The altitude of the target element, in the DEM units
<b>Maximum distance from observer to compute visibility</b>	MAX_DISTANCE	[number] Default: 100.0	Maximum distance from observer to compute visibility, in the DEM units
<b>Ergebnis</b>	OUTPUT	[raster] Default: [Save to temporary file]	Output raster layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ,'	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <a href="#">GDAL driver options section</a> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Additional command-line parameters</b>	EXTRA	[string] Default: None	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Ergebnis</b>	OUTPUT	[raster]	The raster layer displaying the viewshed.

## Pythoncode

**Algorithm ID:** gdal:viewshed

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.5 Rasterprojektionen

### Projektion zuweisen

Wendet ein Koordinatenbezugssystem auf ein Rasterdatensatz an.

This algorithm is derived from the [GDAL edit utility](#).

**Default menu:** *Raster ► Projections*

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT_LAYER	[raster]	Input raster layer
<b>Desired CRS</b>	CRS	[crs]	The projection (CRS) of the output layer

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer with projection</b>	OUTPUT	[raster]	The output raster layer (with the new projection information)

## Pythoncode

**Algorithm ID:** gdal:assignprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Projektion extrahieren

Extracts the projection of a raster file and writes it into a *world* file with extension *.wld*.

This algorithm is derived from the [GDAL srsinfo utility](#).

**Default menu:** *Raster ► Projections*

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input file</b>	INPUT_LAYER	[raster]	Input raster The raster layer has to be file based, as the algorithm uses the path to the raster file as the location of the generated <i>.wld</i> file. Using a non-file raster layer will lead to an error.
<b>Create also .prj file</b>	PRJ_FILE_CREAT	[boolean] Default: False	If this is activated a <i>.prj</i> file containing the projection information is also created.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>World file</b>	WORLD_FILE	[file]	Text file with extension <i>.wld</i> containing transformation parameters for the raster file.
<b>ESRI Shapefile prj file</b>	PRJ_FILE	[file]	Text file with <i>.prj</i> extension that describes the CRS. Will be <i>None</i> if <i>Create also .prj file</i> is False.

## Pythoncode

**Algorithm ID:** gdal:extractprojection

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Warp (reproject)

Reprojects a raster layer into another Coordinate Reference System (CRS). The output file resolution and the resampling method can be chosen.

This algorithm is derived from the [GDAL warp utility](#).

**Default menu:** *Raster ► Projections*

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[raster]	Input raster layer to reproject
<b>Source CRS</b> Optional	SOURCE_CRS	[crs]	Defines the CRS of the input raster layer
<b>Target CRS</b> Optional	TARGET_CRS	[crs] Default: EPSG:4326	The CRS of the output layer
<b>Resampling method to use</b>	RESAMPLING	[enumeration] Default: 0	Pixel value resampling method to use. Options: <ul style="list-style-type: none"> <li>• 0 — Nearest Neighbour</li> <li>• 1 — Bilinear (2x2 kernel)</li> <li>• 2 — Cubic (4x4 kernel)</li> <li>• 3 — Cubic B-Spline (4x4 kernel)</li> <li>• 4 — Lanczos (6x6 kernel)</li> <li>• 5 — Average</li> <li>• 6 — Mode</li> <li>• 7 — Maximum</li> <li>• 8 — Minimum</li> <li>• 9 — Median</li> <li>• 10 — First quartile (Q1)</li> <li>• 11 — Third quartile (Q3)</li> </ul>
<b>NoData value for output bands</b> Optional	NODATA	[number] Default: None	Sets NoData value for output bands. If not provided, then NoData values will be copied from the source dataset.
<b>Output file resolution in target georeferenced units</b> Optional	TARGET_RESOLUTION	[number] Default: None	Defines the output file resolution of reprojection result
<b>Reprojected</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 0	<p>Defines the format of the output raster file. Options:</p> <ul style="list-style-type: none"> <li>• 0 — Use Input Layer Data Type</li> <li>• 1 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 2 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 3 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 4 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 5 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 6 — Float32 (Thirty two bit floating point (float))</li> <li>• 7 — Float64 (Sixty four bit floating point (double))</li> <li>• 8 — CInt16 (Complex Int16)</li> <li>• 9 — CInt32 (Complex Int32)</li> <li>• 10 — CFloat32 (Complex Float32)</li> <li>• 11 — CFloat64 (Complex Float64)</li> <li>• 12 — Int8 (Eight bit signed integer (qint8))</li> </ul> <p>Available options depend on the GDAL version built with QGIS (see <i>Help ► About</i> menu)</p>

Fortsetzung auf der nächsten Seite



Tab. 28.265 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Georeferenced extents of output file to be created</b> Optional	TARGET_EXTENT	[extent]	Sets the georeferenced extent of the output file to be created (in the <i>Target CRS</i> by default. In the <i>CRS of the target raster extent</i> , if specified). Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>CRS of the target raster extent</b> Optional	TARGET_EXTENT_	[crs]	Specifies the CRS in which to interpret the coordinates given for the extent of the output file. This must not be confused with the target CRS of the output dataset. It is instead a convenience e.g. when knowing the output coordinates in a geodetic long/lat CRS, but wanting a result in a projected coordinate system.
<b>Use multithreaded warping implementation</b>	MULTITHREADING	[boolean] Default: False	Two threads will be used to process chunks of the image and perform input/output operations simultaneously. Note that the computation itself is not multithreaded.
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: None	Add extra GDAL command line options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Reprojected</b>	OUTPUT	[raster] Default: [Save to temporary file]	Reprojected output raster layer

## Pythoncode

**Algorithm ID:** gdal:warpreproject

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.6 Vector conversion

### Convert format

Converts any OGR-supported vector layer into another OGR-supported format.

This algorithm is derived from the [ogr2ogr utility](#).

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Convert all layers from dataset</b>	CONVERT_ALL_LA	[boolean] Default: False	Converts the whole dataset. Supported output formats for this option are GPKG and GML.
<b>Converted</b>	OUTPUT	[same as input]	Specification of the output vector layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul> For Save to File, the output format has to be specified. All GDAL vector formats are supported. For Save to a Temporary File the QGIS default vector format will be used.

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ‘ ’ (no additional options)	Additional GDAL creation options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Converted</b>	OUTPUT	[same as input]	The output vector layer

## Pythoncode

**Algorithm ID:** gdal:convertformat

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rasterize (overwrite with attribute)

Overwrites a raster layer with values from a vector layer. New values are assigned based on the attribute value of the overlapping vector feature.

This algorithm is derived from the [GDAL rasterize utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Input raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>Field to use for a burn-in value</b> Optional	FIELD	[tablefield: numeric]	Defines the attribute field to use to set the pixels values

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Add burn in values to existing raster values</b>	ADD	[boolean] Default: False	If False, pixels are assigned the selected field's value. If True, the selected field's value is added to the value of the input raster layer.
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: ;	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rasterized</b>	OUTPUT	[raster]	The overwritten input raster layer

## Pythoncode

**Algorithm ID:** gdal:rasterize\_over

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rasterize (overwrite with fixed value)

Overwrites parts of a raster layer with a fixed value. The pixels to overwrite are chosen based on the supplied (overlapping) vector layer.

This algorithm is derived from the [GDAL rasterize utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Input raster layer</b>	INPUT_RASTER	[raster]	Input raster layer
<b>A fixed value to burn</b>	BURN	[number] Default: 0.0	The value to burn

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Add burn in values to existing raster values</b>	ADD	[boolean] Default: False	If False, pixels are assigned the fixed value. If True, the fixed value is added to the value of the input raster layer.
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: ;	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rasterized</b>	OUTPUT	[raster]	The overwritten input raster layer

## Pythoncode

**Algorithm ID:** gdal:rasterize\_over\_fixed\_value

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Rasterize (vector to raster)

Converts vector geometries (points, lines and polygons) into a raster image.

This algorithm is derived from the [GDAL rasterize utility](#).

**Default menu:** *Raster ► Conversion*

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Field to use for a burn-in value</b> Optional	FIELD	[tablefield: numeric]	Defines the attribute field from which the attributes for the pixels should be chosen
<b>A fixed value to burn</b> Optional	BURN	[number] Default: 0.0	A fixed value to burn into a band for all features.
<b>Burn value extracted from the „Z“ values of the feature</b> Optional	USE_Z	[boolean] Default: False	Indicates that a burn value should be extracted from the „Z“ values of the feature. Works with points and lines (linear interpolation along each segment). For polygons, works properly only if they are flat (same Z value for all vertices)
<b>Output raster size units</b>	UNITS	[enumeration] Default: 0	Units to use when defining the output raster size/resolution. One of: <ul style="list-style-type: none"> <li>0 — Pixels</li> <li>1 — Georeferenced units</li> </ul>
<b>Width/Horizontal resolution</b>	WIDTH	[number] Default: 0.0	Sets the width (if size units is „Pixels“) or horizontal resolution (if size units is „Georeferenced units“) of the output raster. Minimum value: 0.0.

Fortsetzung auf der nächsten Seite

Tab. 28.268 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Height/Vertical resolution</b>	HEIGHT	[number] Default: 0.0	Sets the height (if size units is „Pixels“) or vertical resolution (if size units is „Georeferenced units“) of the output raster.
<b>Output extent</b> Optional	EXTENT	[extent]	Extent of the output raster layer. If the extent is not specified, the minimum extent that covers the selected reference layer(s) will be used. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Assign a specified NoData value to output bands</b> Optional	NODATA	[number] Default: 0.0	Assigns a specified NoData value to output bands
<b>Rasterized</b>	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul> For Save to File, the output format has to be specified. All GDAL raster formats are supported. For Save to a Temporary File the QGIS default raster format will be used.

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ;	For adding one or more creation options that control the raster to be created (colors, block size, file compression...). For convenience, you can rely on predefined profiles (see <i>GDAL driver options section</i> ). Batch Process and Model Designer: separate multiple options with a pipe character ( ).
<b>Output data type</b>	DATA_TYPE	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: <ul style="list-style-type: none"> <li>• 0 — Byte (Eight bit unsigned integer (qint8))</li> <li>• 1 — Int16 (Sixteen bit signed integer (qint16))</li> <li>• 2 — UInt16 (Sixteen bit unsigned integer (qint16))</li> <li>• 3 — UInt32 (Thirty two bit unsigned integer (qint32))</li> <li>• 4 — Int32 (Thirty two bit signed integer (qint32))</li> <li>• 5 — Float32 (Thirty two bit floating point (float))</li> <li>• 6 — Float64 (Sixty four bit floating point (double))</li> <li>• 7 — CInt16 (Complex Int16)</li> <li>• 8 — CInt32 (Complex Int32)</li> <li>• 9 — CFloat32 (Complex Float32)</li> <li>• 10 — CFloat64 (Complex Float64)</li> <li>• 11 — Int8 (Eight bit signed integer (qint8))</li> </ul> Available options depend on the GDAL version built with QGIS (see <i>Help ► About menu</i> )
<b>Pre-initialize the output image with value</b> Optional	INIT	[number]	Pre-initializes the output image bands with this value. Not marked as the NoData value in the output file. The same value is used in all the bands.
<b>Invert rasterization</b>	INVERT	[boolean] Default: False	Burns the fixed burn value, or the burn value associated with the first feature into all parts of the image not inside the provided polygon.
<b>Additional command-line parameters</b> Optional	EXTRA	[string] Default: ;	Add extra GDAL command line options

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Rasterized</b>	OUTPUT	[raster]	Output raster layer

## Pythoncode

**Algorithm ID:** gdal:rasterize

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.2.7 Vector geoprocessing

### Buffer vectors

Create buffers around the features of a vector layer.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Geometry column name</b>	GEOMETRY	[string] Default: 'geometry'	The name of the input layer geometry column to use
<b>Buffer distance</b>	DISTANCE	[number] Default: 10.0	Minimum: 0.0
<b>Dissolve by attribute</b> Optional	FIELD	[tablefield: any] Default: None	Field to use for dissolving
<b>Dissolve results</b>	DISSOLVE	[boolean] Default: False	<b>If set, the result is dissolved.</b> If no field is set for dissolving, all the buffers are dissolved into one feature.
<b>Produce one feature for each geometry in any kind of geometry collection in the source file</b>	EXPLODE_COLLECTION	[boolean] Default: False	

Fortsetzung auf der nächsten Seite



Tab. 28.269 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Buffer</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specify the output buffer layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ; (no additional options)	Additional GDAL creation options.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Buffer</b>	OUTPUT	[vector: polygon]	The output buffer layer

### Pythoncode

**Algorithm ID:** `gdal:buffer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

### Clip vector by extent

Clips any OGR-supported vector file to a given extent.

This algorithm is derived from the [GDAL ogr2ogr](#) utility.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer

Fortsetzung auf der nächsten Seite

Tab. 28.271 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Clip extent</b>	EXTENT	[extent]	<p>Defines the bounding box that should be used for the output vector file. It has to be defined in target CRS coordinates.</p> <p>Available methods are:</p> <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Clipped (extent)</b>	OUTPUT	[same as input] Default: [Save to temporary file]	<p>Specify the output (clipped) layer. One of:</p> <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ; (no additional options)	Additional GDAL creation options.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clipped (extent)</b>	OUTPUT	[same as input]	The output (clipped) layer. The default format is „ESRI Shapefile“.

### Pythoncode

**Algorithm ID:** gdal:clipvectorbyextent

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Clip vector by mask layer

Clips any OGR-supported vector layer by a mask polygon layer.

This algorithm is derived from the [GDAL ogr2ogr utility](#).

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input vector layer
<b>Mask layer</b>	MASK	[vector: polygon]	Layer to be used as clipping extent for the input vector layer.
<b>Clipped (mask)</b>	OUTPUT	[same as input] Default: [Save to temporary file]	The output (masked) layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ';' (no additional options)	Additional GDAL creation options.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Clipped (mask)</b>	OUTPUT	[same as input]	The output (masked) layer. The default format is „ESRI Shapefile“.

### Pythoncode

**Algorithm ID:** gdal:clipvectorbypolygon

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Dissolve

Dissolve (combine) geometries that have the same value for a given attribute / field. The output geometries are multipart.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	The input layer to dissolve
<b>Dissolve field</b> Optional	FIELD	[tablefield: any]	The field of the input layer to use for dissolving
<b>Geometry column name</b>	GEOMETRY	[string] Default: 'geometry'	The name of the input layer geometry column to use for dissolving.
<b>Dissolved</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specify the output layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Produce one feature for each geometry in any kind of geometry collection in the source file</b>	EXPLODE_COLLEC	[boolean] Default: False	Produce one feature for each geometry in any kind of geometry collection in the source file
<b>Keep input attributes</b>	KEEP_ATTRIBUTE	[boolean] Default: False	Keep all attributes from the input layer
<b>Count dissolved features</b>	COUNT_FEATURES	[boolean] Default: False	Count the dissolved features and include it in the output layer.
<b>Compute area and perimeter of dissolved features</b>	COMPUTE_AREA	[boolean] Default: False	<b>Compute the area and perimeter of dissolved features and</b> include them in the output layer
<b>Compute min/max/sum/mean for attribute</b>	COMPUTE_STATIS	[boolean] Default: False	Calculate statistics (min, max, sum and mean) for the numeric attribute specified and include them in the output layer
<b>Numeric attribute to calculate statistics on</b> Optional	STATISTICS_ATT	[tablefield: numeric]	The numeric attribute to calculate statistics on
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ' ' (no additional options)	Additional GDAL creation options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Dissolved</b>	OUTPUT	[same as input]	<b>The output multipart geometry layer (with dissolved geometries)</b>

## Pythoncode

**Algorithm ID:** gdal:dissolve

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Offset curve

Offsets lines by a specified distance. Positive distances will offset lines to the left, and negative distances will offset them to the right.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	The input line layer
<b>Geometry column name</b>	GEOMETRY	[string] Default: 'geometry'	The name of the input layer geometry column to use
<b>Offset distance (left-sided: positive, right-sided: negative)</b>	DISTANCE	[number] Default: 10.0	
<b>Offset curve</b>	OUTPUT	[vector: line] Default: [Save to temporary file]	Specify the output line layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

## Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ' ' (no additional options)	Additional GDAL creation options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Offset curve</b>	OUTPUT	[vector: line]	The output offset curve layer

## Pythoncode

**Algorithm ID:** gdal:offsetcurve

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## One side buffer

Creates a buffer on one side (right or left) of the lines in a line vector layer.

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	The input line layer
<b>Geometry column name</b>	GEOMETRY	[string] Default: 'geometry'	The name of the input layer geometry column to use
<b>Buffer distance</b>	DISTANCE	[number] Default: 10.0	
<b>Buffer side</b>	BUFFER_SIDE	[enumeration] Default: 0	One of: <ul style="list-style-type: none"> <li>0 — Right</li> <li>1 — Left</li> </ul>
<b>Dissolve attribute</b> Optional	by FIELD	[tablefield: any] Default: None	Field to use for dissolving

Fortsetzung auf der nächsten Seite

Tab. 28.275 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Dissolve all results</b>	DISSOLVE	[boolean] Default: False	<b>If set, the result is dissolved.</b> If no field is set for dissolving, all the buffers are dissolved into one feature.
<b>Produce one feature for each geometry in any kind of geometry collection in the source file</b>	EXPLODE_COLLEC	[boolean] Default: False	
<b>One-sided buffer</b>	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specify the output buffer layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ‘ (no additional options)	Additional GDAL creation options.

### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>One-sided buffer</b>	OUTPUT	[vector: polygon]	The output buffer layer

### Pythoncode

**Algorithm ID:** gdal:onesidebuffer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Points along lines

Generates a point on each line of a line vector layer at a distance from start. The distance is provided as a fraction of the line length.

### Parameter

#### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: line]	The input line layer
<b>Geometry column name</b>	GEOMETRY	[string] Default: 'geometry'	The name of the input layer geometry column to use
<b>Distance from line start represented as a fraction of line length</b>	DISTANCE	[number] Default: 0.5 (middle of the line)	
<b>Points along lines</b>	OUTPUT	[vector: point] Default: [Save to temporary file]	Specify the output point layer. One of: <ul style="list-style-type: none"><li>• Save to a Temporary File</li><li>• Save to File...</li></ul>

#### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: '' (no additional options)	Additional GDAL creation options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Points along line</b>	OUTPUT	[vector: point]	The output point layer

## Pythoncode

**Algorithm ID:** gdal:pointsalonglines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.



## 28.2.8 Vector miscellaneous

### Build virtual vector

Creates a virtual vector layer that contains a set of vector layers. The output virtual vector layer will not be opened in the current project.

This algorithm is especially useful in case another algorithm needs multiple layers but accept only one `vrt` in which the layers are specified.

#### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input datasources</b>	INPUT	[vector: any] [list]	Select the vector layers you want to use to build the virtual vector
<b>Create „unioned“ VRT</b>	UNIONED	[boolean] Default: False	Check if you want to unite all the vectors in a single <code>vrt</code> file
<b>Virtual vector</b>	OUTPUT	[same as input] Default: [Save to temporary file]	Specify the output layer containing only the duplicates. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul>

#### Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Virtual vector</b>	OUTPUT	[vector: any]	The output virtual vector made from the chosen sources

#### Pythoncode

**Algorithm ID:** `gdal:buildvirtualvector`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

#### Execute SQL

Runs a simple or complex query with SQL syntax on the source layer. The result of the query will be added as a new layer.

This algorithm is derived from the [GDAL ogr2ogr utility](#).

## Parameter

### Basic parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	OGR-supported input vector layer
<b>SQL expression</b>	SQL	[string]	Defines the SQL query, for example SELECT * FROM my_table WHERE name is not null.
<b>SQL dialect</b>	DIALECT	[enumeration] Default: 0	SQL dialect to use. One of: <ul style="list-style-type: none"> <li>• 0 — None</li> <li>• 1 — OGR SQL</li> <li>• 2 — SQLite</li> </ul>
<b>SQL result</b>	OUTPUT	[vector: any]	Specification of the output layer. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul> For Save to File, the output format has to be specified. All GDAL vector formats are supported. For Save to a Temporary File the default output vector layer format will be used.

### Advanced parameters

Bezeichnung	Name	Typ	Beschreibung
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ' ' (no additional options)	Additional GDAL creation options.

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>SQL result</b>	OUTPUT	[vector: any]	Vector layer created by the query

## Pythoncode

**Algorithm ID:** gdal:executesql

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export to PostgreSQL (available connections)

Imports vector layers inside a PostgreSQL database on the basis of an available connection. The connection has to *be defined properly* beforehand. Be aware that the checkboxes ‚Save Username‘ and ‚Save Password‘ are activated. Then you can use the algorithm.

This algorithm is derived from the [GDAL ogr2ogr utility](#).

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Database (connection name)</b>	DATABASE	[string]	The PostgreSQL database to connect to
<b>Input layer</b>	INPUT	[vector: any]	OGR-supported vector layer to export to the database
<b>Shape encoding</b> Optional	SHAPE_ENCODING	[string] Default: ‘	Sets the encoding to apply to the data
<b>Output geometry type</b>	GTYPE	[enumeration] Default: 0	Defines the output geometry type. One of: <ul style="list-style-type: none"> <li>• 0 —</li> <li>• 1 — NONE</li> <li>• 2 — GEOMETRY</li> <li>• 3 — POINT</li> <li>• 4 — LINESTRING</li> <li>• 5 — POLYGON</li> <li>• 6 — GEOMETRYCOLLECTION</li> <li>• 7 — MULTIPOINT</li> <li>• 8 — MULTIPOLYGON</li> <li>• 9 — MULTILINESTRING</li> </ul>
<b>Assign an output CRS</b> Optional	A_SRS	[crs] Default: None	Defines the output CRS of the database table
<b>Reproject to this CRS on output</b> Optional	T_SRS	[crs] Default: None	Reprojects/transforms to this CRS on output
<b>Override source CRS</b> Optional	S_SRS	[crs] Default: None	Overrides the input layer CRS
<b>Schema (schema name)</b> Optional	SCHEMA	[string] Default: ‚public‘	Defines the schema for the database table
<b>Table to export to (leave blank to use layer name)</b> Optional	TABLE	[string] Default: ‘	Defines a name for the table that will be imported into the database. By default the table name is the name of the input vector file.
<b>Primary Key (new field)</b> Optional	PK	[string] Default: ‚id‘	Defines which attribute field will be the primary key of the database table
<b>Primary Key (existing field, used if the above option is left empty)</b> Optional	PRIMARY_KEY	[tablefield: any] Default: None	Defines which attribute field in the exported layer will be the primary key of the database table

Fortsetzung auf der nächsten Seite

Tab. 28.278 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Geometry column name</b> Optional	GEOCOLUMN	[string] Default: ‚geom‘	Defines in which attribute field of the database there will be the geometry information
<b>Vector dimensions</b> Optional	DIM	[enumeration] Default: 0 (2D)	Defines if the vector file to be imported has 2D or 3D data. One of: <ul style="list-style-type: none"> <li>• 0 — 2</li> <li>• 1 — 3</li> </ul>
<b>Distance tolerance for simplification</b> Optional	SIMPLIFY	[string] Default: ‚‘	Defines a distance tolerance for the simplification of the vector geometries to be imported. By default there is no simplification.
<b>Maximum distance between 2 nodes (densification)</b> Optional	SEGMENTIZE	[string] Default: ‚‘	The maximum distance between two nodes. Used to create intermediate points. By default there is no densification.
<b>Select features by extent (defined in input layer CRS)</b> Optional	SPAT	[extent] Default: None	You can select features from a given extent that will be in the output table. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Clip the input layer using the above (rectangle) extent</b> Optional	CLIP	[boolean] Default: False	The input layer will be clipped by the extent you defined before
<b>Select features using a SQL „WHERE“ statement (Ex: column=“value“)</b> Optional	WHERE	[string] Default: ‚‘	Defines with a SQL „WHERE“ statement which features should be selected from the input layer
<b>Group N features per transaction (Default: 2000)</b> Optional	GT	[string] Default: ‚‘	You can group the input features in transactions where N defines the size. By default N limits the transaction size to 20000 features.
<b>Overwrite existing table</b> Optional	OVERWRITE	[boolean] Default: True	If there is a table with the same name in the database, and if this option is set to True, the table will be overwritten.

Fortsetzung auf der nächsten Seite

Tab. 28.278 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Append to existing table</b> Optional	APPEND	[boolean] Default: False	If checked / True the vector data will be appended to an existing table. New fields found in the input layer are ignored. By default a new table will be created.
<b>Append and add new fields to existing table</b> Optional	ADDFIELDS	[boolean] Default: False	If activated the vector data will be appended to an existing table, there won't be a new table created. New fields found in input layer are added to the table. By default a new table will be created.
<b>Do not launder columns/table names</b> Optional	LAUNDER	[boolean] Default: False	With this option checked you can prevent the default behaviour (converting column names to lowercase, removing spaces and other invalid characters).
<b>Do not create Spatial Index</b> Optional	INDEX	[boolean] Default: False	Prevents a spatial index for the output table from being created. By default, a spatial index is added.
<b>Continue after a failure, skipping the failed feature</b> Optional	SKIPFAILURES	[boolean] Default: False	
<b>Promote to Multipart</b> Optional	PROMOTETOMULTI	[boolean] Default: True	Casts features geometry type to multipart in the output table
<b>Keep width and precision of input attributes</b> Optional	PRECISION	[boolean] Default: True	Avoids modifying column attributes to comply with input data
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ; (no additional options)	Additional GDAL creation options.

## Ausgaben

This algorithm has no output.

## Pythoncode

**Algorithm ID:** gdal:importvectorintopostgisdatabaseavailableconnections

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Export to PostgreSQL (new connection)

Imports vector layers inside a PostgreSQL database. A new connection to the PostGIS database must be created.

This algorithm is derived from the [GDAL ogr2ogr utility](#).

### Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	OGR-supported vector layer to export to the database
<b>Shape encoding</b> Optional	SHAPE_ENCODING	[string] Default: ‘	Sets the encoding to apply to the data
<b>Output geometry type</b>	GTYPE	[enumeration] Default: 0	Defines the output geometry type. One of: <ul style="list-style-type: none"> <li>• 0 —</li> <li>• 1 — NONE</li> <li>• 2 — GEOMETRY</li> <li>• 3 — POINT</li> <li>• 4 — LINESTRING</li> <li>• 5 — POLYGON</li> <li>• 6 — GEOMETRYCOLLECTION</li> <li>• 7 — MULTIPOINT</li> <li>• 8 — MULTIPOLYGON</li> <li>• 9 — MULTILINESTRING</li> </ul>
<b>Assign an output CRS</b> Optional	A_SRS	[crs] Default: None	Defines the output CRS of the database table
<b>Reproject to this CRS on output</b> Optional	T_SRS	[crs] Default: None	Reprojects/transforms to this CRS on output
<b>Override source CRS</b> Optional	S_SRS	[crs] Default: None	Overrides the input layer CRS
<b>Host</b> Optional	HOST	[string] Default: ,localhost‘	Name of the database host
<b>Port</b> Optional	PORT	[string] Default: ,5432‘	Port number the PostgreSQL database server listens on
<b>Username</b> Optional	USER	[string] Default: ‘	User name used to log in to the database
<b>Database name</b> Optional	DBNAME	[string] Default: ‘	Name of the database
<b>Password</b> Optional	PASSWORD	[string] Default: ‘	Password used with Username to connect to the database
<b>Schema (schema name)</b> Optional	SCHEMA	[string] Default: ,public‘	Defines the schema for the database table
<b>Table name, leave blank to use input name</b> Optional	TABLE	[string] Default: ‘	Defines a name for the table that will be imported into the database. By default the table name is the name of the input vector file.
<b>Primary Key (new field)</b> Optional	PK	[string] Default: ,id‘	Defines which attribute field will be the primary key of the database table

Fortsetzung auf der nächsten Seite

Tab. 28.279 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Primary Key</b> (existing field, used if the above option is left empty) Optional	PRIMARY_KEY	[tablefield: any] Default: None	Defines which attribute field in the exported layer will be the primary key of the database table
<b>Geometry column name</b> Optional	GEOCOLUMN	[string] Default: 'geom'	Defines in which attribute field to store the geometry information
<b>Vector dimensions</b> Optional	DIM	[enumeration] Default: 0 (2D)	Defines if the vector file to be imported has 2D or 3D data. One of: <ul style="list-style-type: none"> <li>• 0 — 2D</li> <li>• 1 — 3D</li> </ul>
<b>Distance tolerance for simplification</b> Optional	SIMPLIFY	[string] Default: ''	Defines a distance tolerance for the simplification of the vector geometries to be imported. By default there is no simplification.
<b>Maximum distance between 2 nodes (densification)</b> Optional	SEGMENTIZE	[string] Default: ''	The maximum distance between two nodes. Used to create intermediate points. By default there is no densification.
<b>Select features by extent (defined in input layer CRS)</b> Optional	SPAT	[extent] Default: None	You can select features from a given extent that will be in the output table. Available methods are: <ul style="list-style-type: none"> <li>• Calculate from layer...: uses extent of a layer loaded in the current project</li> <li>• Calculate from layout map...: uses extent of a <i>layout map item</i> in the active project</li> <li>• Calculate from bookmark...: uses extent of a saved <i>bookmark</i></li> <li>• Use map canvas extent</li> <li>• Draw on canvas: click and drag a rectangle delimiting the area to take into account</li> <li>• Enter the coordinates as xmin, xmax, ymin, ymax</li> </ul>
<b>Clip the input layer using the above (rectangle) extent</b> Optional	CLIP	[boolean] Default: False	The input layer will be clipped by the extent you defined before
<b>Fields to include (leave empty to use all fields)</b> Optional	FIELDS	[string] [list] Default: []	Defines fields to keep from the imported vector file. If none is selected, all the fields are imported.

Fortsetzung auf der nächsten Seite

Tab. 28.279 – Fortsetzung der vorherigen Seite

Bezeichnung	Name	Typ	Beschreibung
<b>Select features using a SQL „WHERE“ statement</b> (Ex: column=“value“) Optional	WHERE	[string] Default: ;	Defines with a SQL „WHERE“ statement which features should be selected for the output table
<b>Group N features per transaction</b> (Default: 2000) Optional	GT	[string] Default: ;	You can group the input features in transactions where N defines the size. By default N limits the transaction size to 20000 features.
<b>Overwrite existing table</b> Optional	OVERWRITE	[boolean] Default: True	If there is a table with the same name in the database, and if this option is set to True, the table will be overwritten.
<b>Append to existing table</b> Optional	APPEND	[boolean] Default: False	If checked / True the vector data will be appended to an existing table. New fields found in the input layer are ignored. By default a new table will be created.
<b>Append and add new fields to existing table</b> Optional	ADDFIELDS	[boolean] Default: False	If activated the vector data will be appended to an existing table, there won't be created a new table. New fields found in input layer are added to the table. By default a new table will be created.
<b>Do not launder columns/table names</b> Optional	LAUNDER	[boolean] Default: False	With this option checked you can prevent the default behaviour (converting column names to lowercase, removing spaces and other invalid characters).
<b>Do not create Spatial Index</b> Optional	INDEX	[boolean] Default: False	Prevents a spatial index for the output table from being created. By default, a spatial index is added.
<b>Continue after a failure, skipping the failed feature</b> Optional	SKIPFAILURES	[boolean] Default: False	
<b>Promote to Multipart</b> Optional	PROMOTETOMULTI	[boolean] Default: True	Casts features geometry type to multipart in the output table
<b>Keep width and precision of input attributes</b> Optional	PRECISION	[boolean] Default: True	Avoids modifying column attributes to comply with input data
<b>Additional creation options</b> Optional	OPTIONS	[string] Default: ; (no additional options)	Additional GDAL creation options.



## Ausgaben

This algorithm has no output.

## Pythoncode

**Algorithm ID:** `gdal:importvectorintopostgisdatabasenewconnection`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## Vector Information

Creates an information file that lists information about an OGR-supported data source. The output will be shown in a 'Result' window and can be written into a HTML-file. The information includes the geometry type, feature count, the spatial extent, the projection information and many more.

This algorithm is derived from the [GDAL ogrinfo utility](#).

## Parameter

Bezeichnung	Name	Typ	Beschreibung
<b>Input layer</b>	INPUT	[vector: any]	Input vector layer
<b>Summary output only</b> Optional	SUMMARY_ONLY	[boolean] Default: True	
<b>Suppress metadata info</b> Optional	NO_METADATA	[boolean] Default: False	
<b>Layer information</b>	OUTPUT	[html] Default: [Save to temporary file]	Specify the output HTML file that includes the file information. One of: <ul style="list-style-type: none"> <li>• Save to a Temporary File</li> <li>• Save to File...</li> </ul> If no HTML-file is defined the output will be written to a temporary file

## Ausgaben

Bezeichnung	Name	Typ	Beschreibung
<b>Layer information</b>	OUTPUT	[html]	The output HTML-file that includes the file information.

### Pythoncode

**Algorithm ID:** gdal:ogrinfo

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMES and values. See [Verarbeitung Algorithmen von der Konsole aus verwenden](#) for details on how to run processing algorithms from the Python console.

## 28.3 OTB Anwendungs Bereitstellung

OTB (Orfeo ToolBox) ist eine Bildverarbeitungsbibliothek für Fernerkundungsdaten. Sie stellt auch Anwendungen zur Verfügung, die Bildverarbeitungsfunktionen bereitstellen. Die Liste der Anwendungen und ihre Dokumentation sind im [OTB CookBook](#) verfügbar.

### 29.1 QGIS Plugins


QGIS has been designed with a plugin architecture. This allows many new features and functions to be easily added to the application. Some of the features in QGIS are actually implemented as plugins.

#### 29.1.1 Kernerweiterungen und externe Erweiterungen

QGIS Plugins sind entweder als **Core-Plugins** oder **Externe Plugins** implementiert.

*Core Plugins* are maintained by the QGIS Development Team and are automatically part of every QGIS distribution. They are written in one of two languages: **C++** or **Python**.


Most of External Plugins are currently written in Python. They are stored either in the 'Official' QGIS Repository at <https://plugins.qgis.org/plugins/> or in external repositories and are maintained by the individual authors. Detailed documentation about the usage, minimum QGIS version, home page, authors, and other important information are provided for the plugins in the Official repository. For other external repositories, documentation might be available with the external plugins themselves. External plugins documentation is not included in this manual.




To install or activate a plugin, go to *Plugins* menu and select  *Manage and install plugins....* Installed external python plugins are placed under the `python/plugins` folder of the active *user profile* path.


Pfade zu benutzerdefinierten C++ Plugins Bibliotheken können auch unter *Einstellungen ► Optionen ► System* hinzugefügt werden.

## 29.1.2 Der Erweiterungen Dialog



### The Settings tab

At the bottom of the left panel, the  *Settings* tab is the main place you can configure which plugins can be displayed in your application. You can use the following options:

-  *Check for Updates on Startup*. Whenever an installed plugin has update available, QGIS will inform you *Every Time QGIS starts, Once a Day, Every 3 Days, Every Week, Every 2 Weeks or Every month*.
-  *Show also Experimental Plugins*. QGIS will show you plugins in early stages of development, which are generally unsuitable for production use. For these plugins, you can install either the stable or the experimental version, and at any moment switch from one to the other.
-  *Show also Deprecated Plugins*. These plugins are usually unmaintained because they have replacement functions in QGIS, a lack of maintainers, they rely on functions that are no longer available in QGIS... They are generally unsuitable for production use and appear grayed in the plugins list.

By default, in the *Plugin Repositories* section, QGIS provides you with its official plugin repository with the URL `https://plugins.qgis.org/plugins/plugins.xml?qgis=version` (where `<version>` represents the exact QGIS version you are running). To add external author repositories, click  *Add...* and fill in the *Repository Details* form with a name and the URL. The URL can be of `http://` or `file://` protocol type.

The default QGIS repository is an open repository and you don't need any authentication to access it. You can however deploy your own plugin repository and require an authentication (basic authentication, PKI). You can get more information on QGIS authentication support in *Authentifizierung* chapter.

If you do not want one or more of the added repositories, they can be disabled from the Settings tab via the  *Edit...* button, or completely removed with the  *Delete* button.

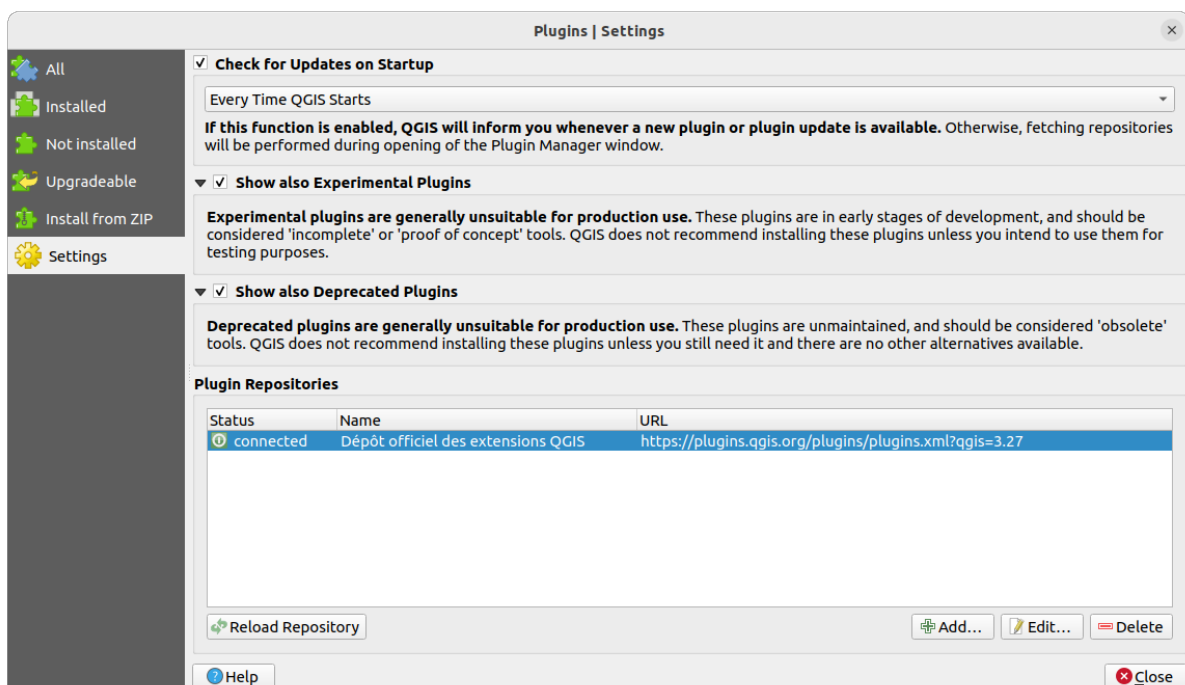








Abb. 29.1: The  *Settings* tab

## Browsing the plugins

### The tabs

The upper tabs in the *Plugins* dialog provide you with lists of plugins based on their install, creation or update status. Depending on the plugins settings, available tabs can be:

-  *All*: shows all the available plugins in the enabled repositories
-  *Installed*: shows both the plugins you installed and the core plugins that are installed by default and you can not uninstall
-  *Not installed*: shows uninstalled or not yet installed plugins in the enabled repositories
-  *New*: shows plugins released since the last *Check for Updates on Startup*
-  *Upgradeable*: shows installed plugins that have published a more recent version in the repository
-  *Invalid*: shows all installed plugins that are currently broken for any reason (missing dependency, errors while loading, incompatible functions with QGIS version...)

At the top of the tabs, a *Search* function helps you find any plugin using metadata information (author, name, description, tag,...).

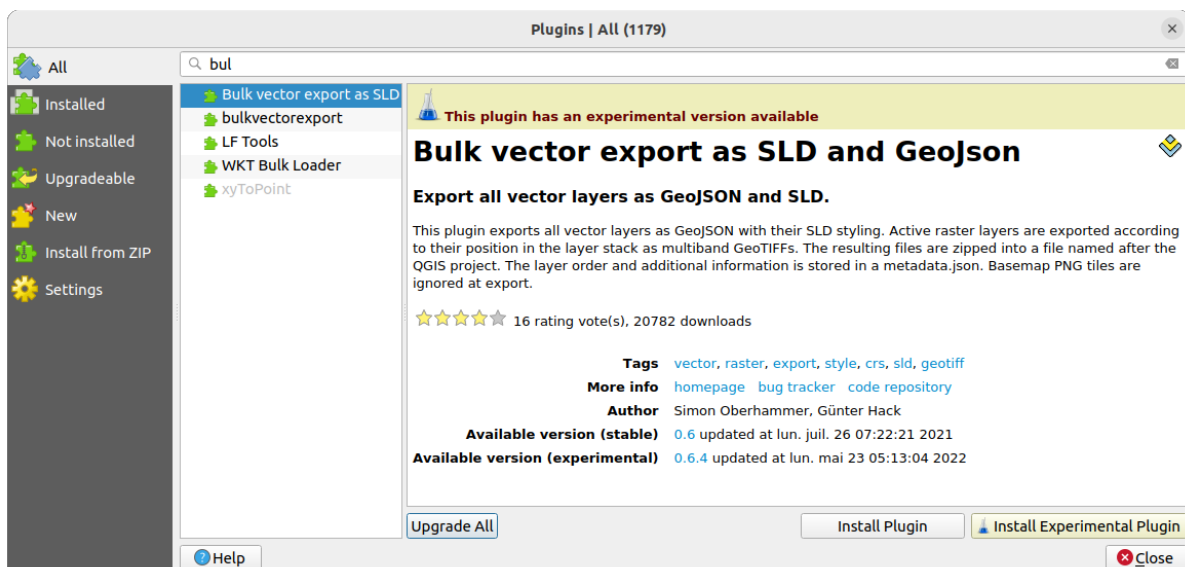



Abb. 29.2: Searching a plugin from the  *All* tab

## The Plugins


Select a plugin and you will have some metadata displayed in the right panel:

- information on whether the plugin is experimental or has an experimental version available (if *Show also Experimental Plugins* is checked)
- summary and description
- Bewertungen (Sie können eine Bewertung für Ihre bevorzugte Erweiterung abgeben!)
- Elemente
- einige nützliche Links wie die Homepage, die Fehlerverfolgung und das Quellcode-Repository

- Autoren
- version(s) available with link to download page in the repository, or path to local folder for installed plugins

The *Plugin Manager* dialog allows you to interact with the latest version of the plugins. When enabled, the experimental version can be shown only if it is more recent than the latest stable version. Depending on the active tab, whether the selected plugin is installed, you will be given some of the following options:


- *Install*: installs the latest stable version of the selected plugin
- *Install Experimental Plugin*: installs the experimental version of the selected plugin
- *Reinstall Plugin*: installs the same stable version of the plugin e.g. after it has failed to load
- *Reinstall Experimental Plugin*: installs the same stable version of the plugin e.g. after it has failed to load
- *Upgrade Plugin*: upgrades selected plugin to its latest stable version
- *Upgrade Experimental Plugin*: upgrades selected plugin to its experimental version
- *Upgrade All*: upgrades all installed plugins to their more recent stable or experimental version (depending on whether their previously installed version was stable or experimental).
- *Downgrade Plugin*: moves from the experimental version of the plugin to its previous stable version
- *Downgrade Experimental Plugin*: moves from an experimental version of the plugin to its latest published experimental version. This may occur when playing with a not yet published version.
- *Uninstall Plugin*: removes the installed plugin from the user profile

An installed plugin displays a  checkbox on its left. Uncheck it to temporarily deactivate the plugin.

Right-click on a plugin in the list and you will be able to sort the plugins list by various metadata. The new order applies to all the tabs. Sort options are:

- *Sort by Name*
- *Sort by Downloads*
- *Sort by Vote*
- *Sort by Status*
- *Sort by Date Created*
- *Sort by Date Updated*

### The Install from ZIP tab

The  *Install from ZIP* tab provides a file selector widget to import plugins in a zipped format, e.g. plugins downloaded directly from their repository. Encrypted files are supported.

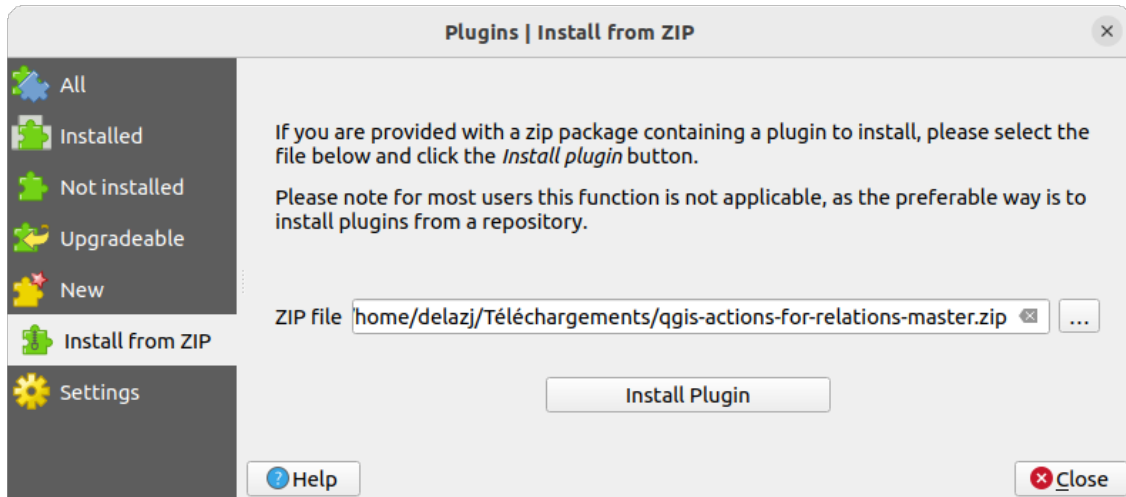



Abb. 29.3: The  *Install from zip* tab

## 29.2 Using QGIS Core Plugins

### 29.2.1 DB Manager Plugin

The DB Manager Plugin is intended to be the main tool to integrate and manage spatial database formats supported by QGIS (PostGIS, SpatiaLite, GeoPackage, Oracle Spatial, Virtual layers) in one user interface. The  DB Manager Plugin provides several features. You can drag layers from the QGIS Browser into the DB Manager, and it will import your layer into your spatial database. You can drag and drop tables between spatial databases and they will get imported.

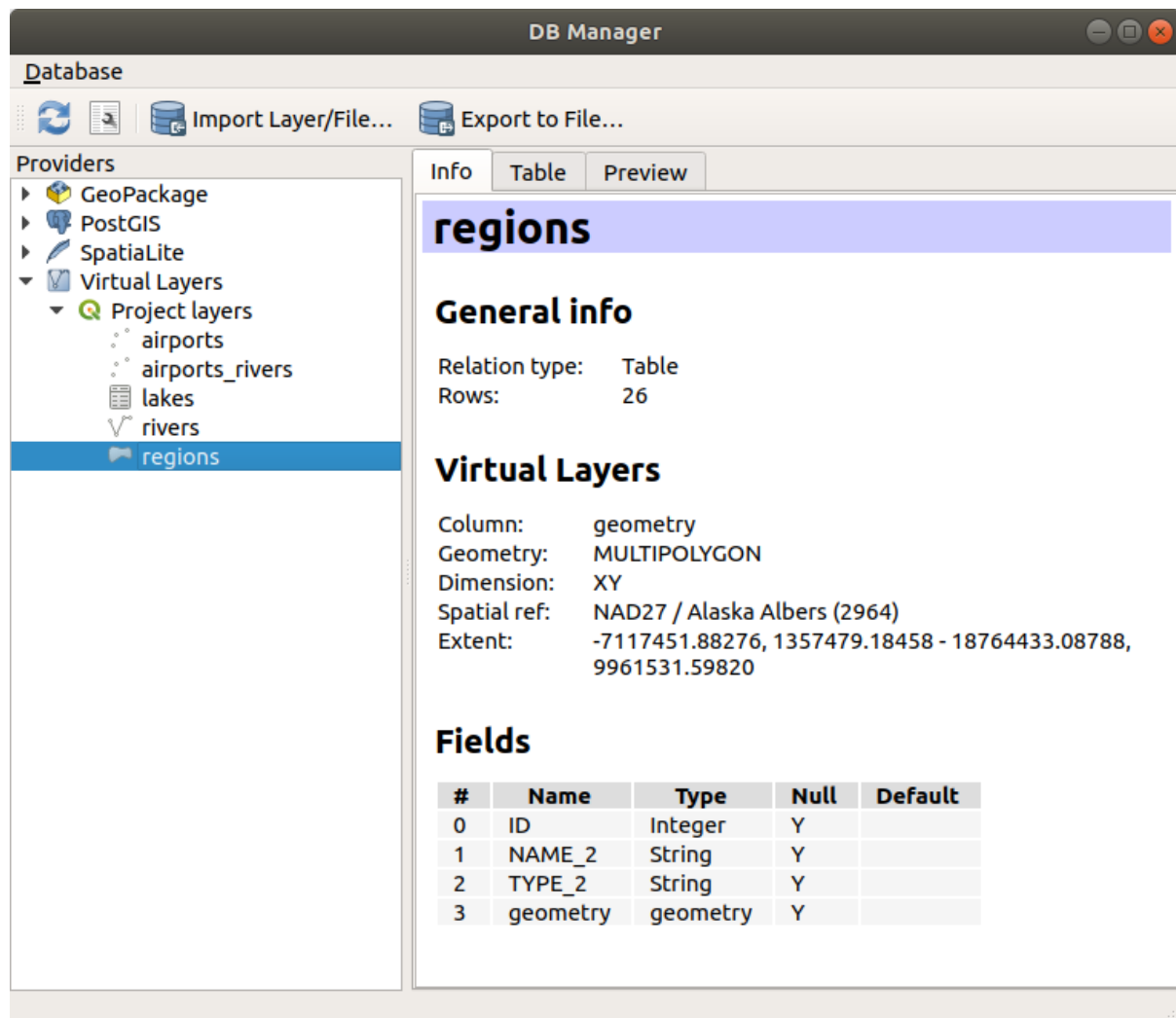


Abb. 29.4: DB Manager dialog

The *Database* menu allows you to connect to an existing database, to start the SQL window and to exit the DB Manager Plugin. Once you are connected to an existing database, the menus *Schema* (relevant for DBMSs, such as PostGIS / PostgreSQL) and *Table* will appear.

The *Schema* menu includes tools to create and delete (only if empty) schemas and, if topology is available (e.g. with PostGIS topology), to start a *TopoViewer*.

The *Table* menu allows you to create and edit tables and to delete tables and views. It is also possible to empty tables and to move tables between schemas. You can *Run Vacuum Analyze* for the selected table. *Vacuum* reclaims space and makes it available for reuse, and *analyze* updates statistics that is used to determine the most efficient way to execute a query. *Change Logging...* allows you to add change logging support to a table. Finally, you can *Import Layer/File...* and *Export to File...*

---

**Bemerkung:** Using the DB Manager it is possible to add comments for tables and columns of a PostgreSQL Database.

---

The *Providers* window lists all existing databases supported by QGIS. With a double-click, you can connect to the database. With the right mouse button, you can rename and delete existing schemas and tables. Tables can also be added to the QGIS canvas with the context menu.

If connected to a database, the **main** window of the DB Manager offers four tabs. The *Info* tab provides information about the table and its geometry, as well as about existing fields, constraints and indexes. It allows you to create a



spatial index on a the selected table. The *Table* tab shows the table, and the *Preview* tab renders the geometries as preview. When you open an *SQL Window*, it will be placed in a new tab.

## Working with the SQL Window

You can use the DB Manager to execute SQL queries against your spatial database. Queries can be saved and loaded, and there the *SQL Query Builder* will help you formulate your queries. You can even view spatial output by checking *Load as new layer* and specifying *Column(s) with unique values* (IDs), *Geometry column* and *Layer name (prefix)*. It is possible to highlight a portion of the SQL to only execute that portion when pressing **Ctrl+R** or clicking the *Execute* button.

The *Query History* button stores the last 20 queries of each database and provider.

Double clicking on an entry will add the string to the SQL window.

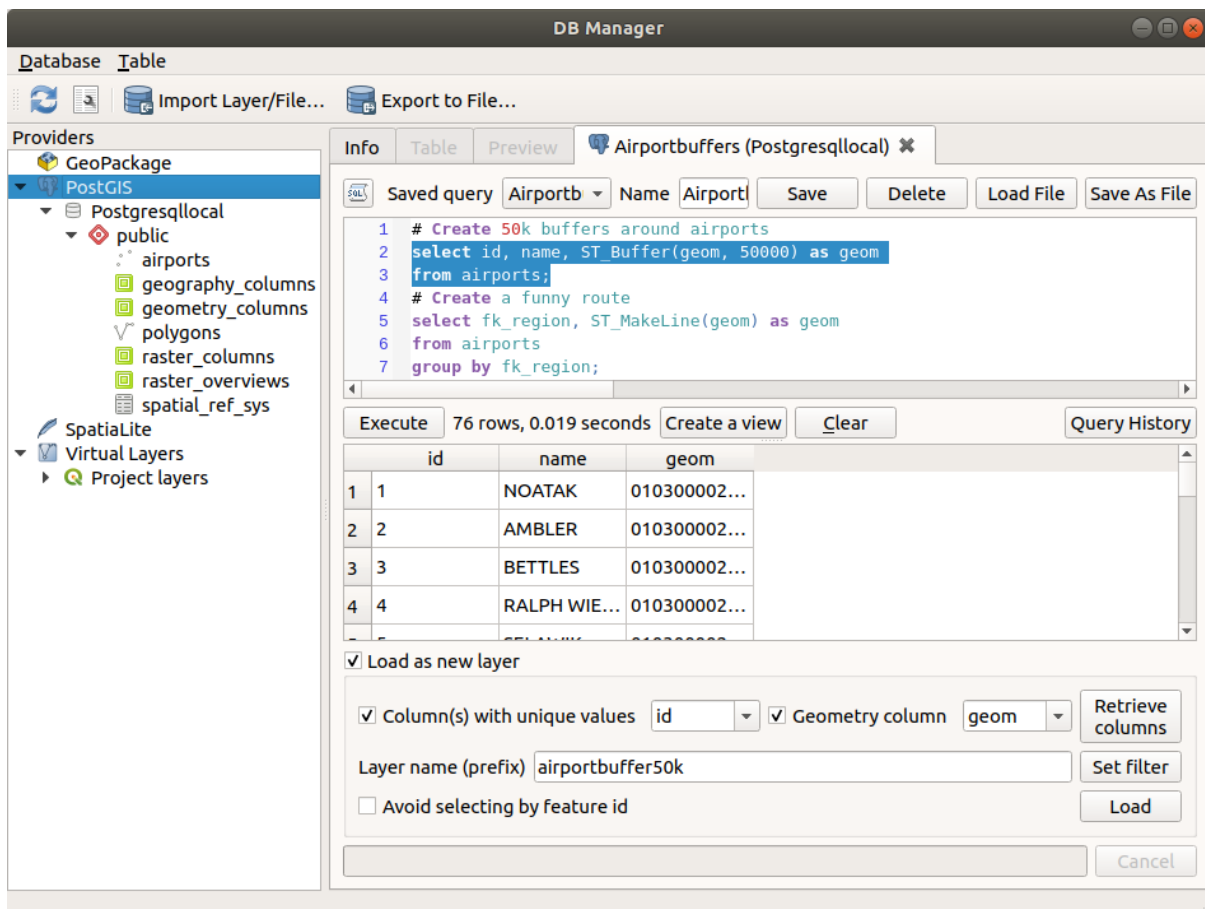



Abb. 29.5: Executing SQL queries in the DB Manager SQL window

**Bemerkung:** The SQL Window can also be used to create Virtual Layers. In that case, instead of selecting a database, select **QGIS Layers** under **Virtual Layers** before opening the SQL Window. See [Creating virtual layers](#) for instructions on the SQL syntax to use.










## 29.2.2 Geometry Checker Plugin

Geometry Checker is a powerful core plugin to check and fix the geometry validity of a layer. It is available from the *Vector* menu ( *Check Geometries...*).

### Configuring the checks

The *Check Geometries* dialog shows different grouped settings in the first tab (*Setup*):

- *Input vector layers*: to select the layers to check. A  *Only selected features* checkbox can be used to restrict the checking to the geometries of the selected features.
- *Allowed geometry types* gives the chance to restrict the geometry type of the input layer(s) to:
  -  Point
  -  Multipoint
  -  Line
  -  Multiline
  -  Polygon
  -  Multipolygon
- *Geometry validity*. Depending on geometry types you can choose between:
  -  *Self intersections*
  -  *Duplicate nodes*
  -  *Self contacts*
  -  *Polygon with less than 3 nodes.*
- *Geometry properties*. Depending on geometry types, different options are available:
  -  *Polygons and multipolygons may not contain any holes*
  -  *Multipart objects must consist of more than one part*
  -  *Lines must not have dangles*
- *Geometry conditions*. Allows you to add some condition to validate the geometries with:
  -  *Minimal segment length (map units)*
  -  *Minimum angle between segment (deg)*
  -  *Minimal polygon area (map units sqr.)*
  -  *No sliver polygons with a Maximum thinness*  and a  *Max. area (map units sqr.)*
- *Topology checks*. Depending on geometry types, many different options are available:
  -  *Checks for duplicates*
  -  *Checks for features within other features*
  -  *Checks for overlaps smaller than*
  -  *Checks for gaps smaller than*

-  *Points must be covered by lines*
-  *Points must properly lie inside a polygon*
-  *Lines must not intersect any other lines*
-  *Lines must not intersect with features of layer* 
-  *Polygons must follow boundaries of layer* 
- *Tolerance.* You can define the tolerance of the check in map layer units.
- *Output vector layer* gives the choice to:
  -  *Modify input layer*
  -  *Create new layers*

When you are happy with the configuration, you can click on the *Run* button.

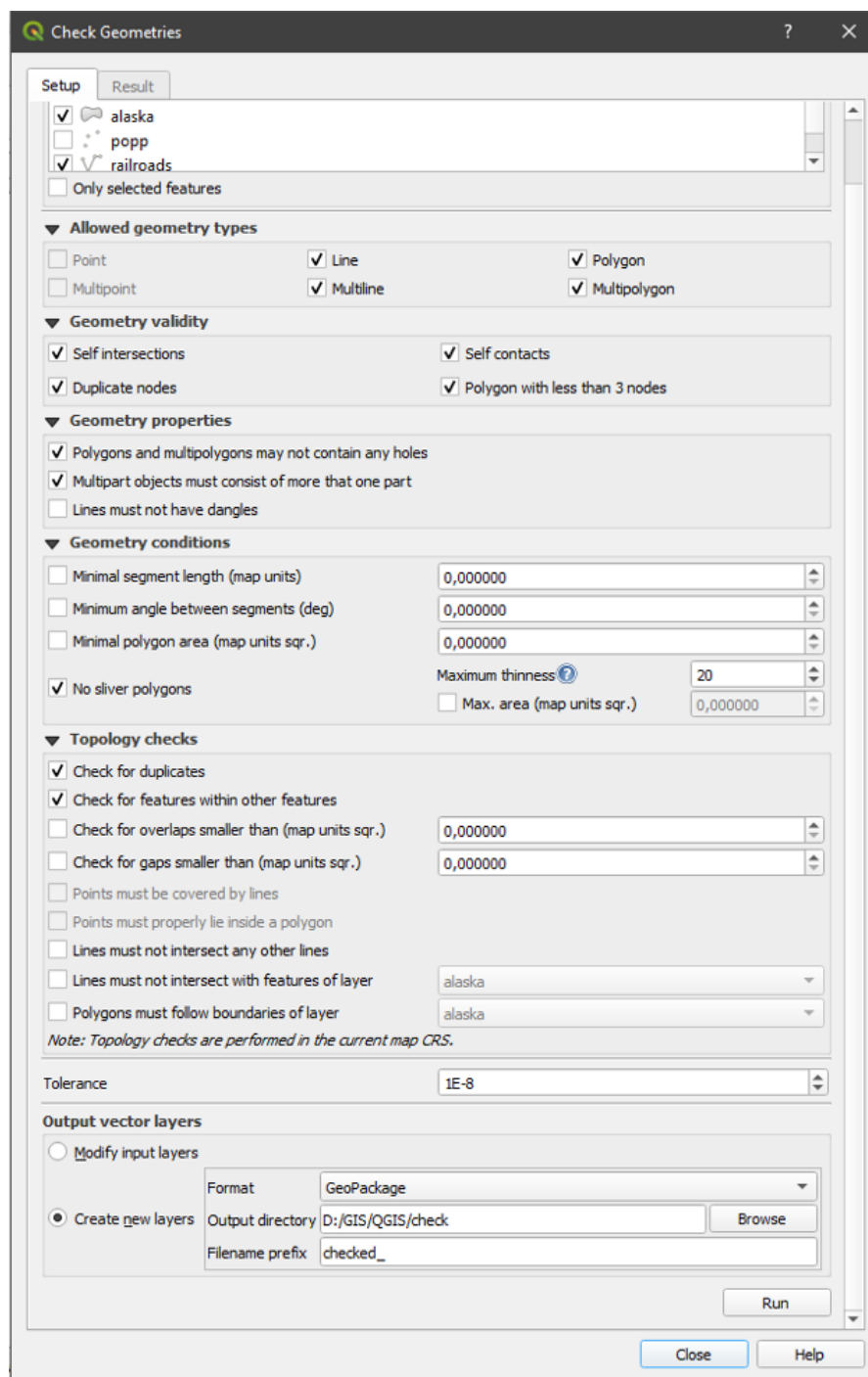


Abb. 29.6: The Geometry Checker Plugin

The *Geometry Checker Plugin* can find the following errors:

- Self intersections: a polygon with a self intersection
- Duplicate nodes: two duplicates nodes in a segment
- Holes: hole in a polygon
- Segment length: a segment length lower than a threshold
- Minimum angle: two segments with an angle lower than a threshold
- Minimum area: polygon area lower than a threshold

- Silver polygon: this error come from very small polygon (with small area) with a large perimeter
- Duplicates features
- Feature within feature
- Overlaps: polygon overlapping
- Gaps: gaps between polygons

The following figure shows the different checks made by the plugin.

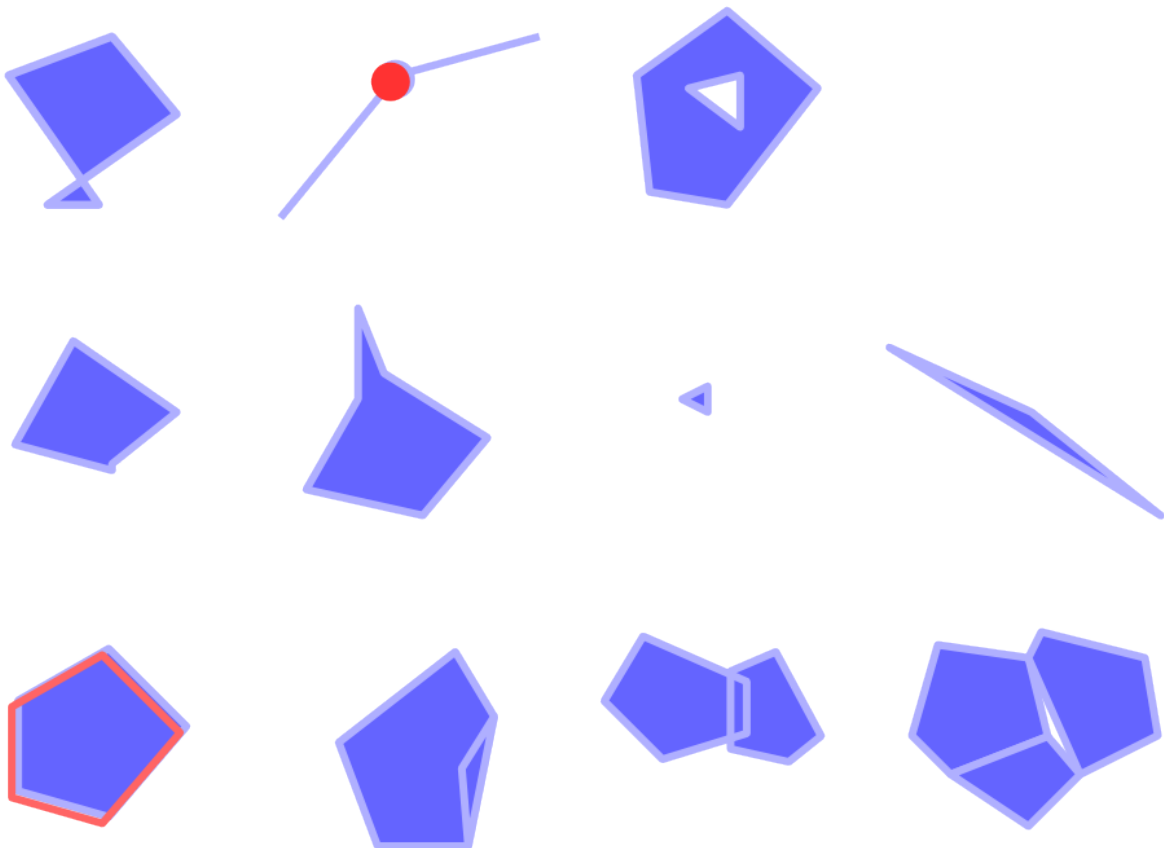


Abb. 29.7: Some checks supported by the plugin

## Analysing the results


The results appear in the second tab (*Result*) and as an overview layer of the errors in the canvas (its name has the default prefix *checked\_*). A table lists the *Geometry check result* with one error per row and columns containing: the layer name, an ID, the error type, then the coordinates of the error, a value (depending on the type of the error) and finally the resolution column which indicates the resolution of the error. At the bottom of this table, you can *Export* the error into different file formats. You also have a counter with the number of total errors and fixed ones.

You can select a row to see the location of the error. You can change this behavior by selecting another action between

☐ *Error* (default), ☐ *Feature*, ☐ *Don't move*, and ☒ *Highlight selected features*.

Below the zoom action when clicking on the table row, you can:

- *Show selected features in attribute table*
- *Fix selected errors using default resolution*
- *Fix selected errors, prompt for resolution method* You will see a window to choose the resolution's method among which:

- Merge with neighboring polygon with longest shared edge
  - Merge with neighboring polygon with largest area
  - Merge with neighboring polygon with identical attribute value, if any, or leave as is
  - Delete feature
  - No action
-  *Error resolution settings* allows you to change the default resolution method depending on the error type

### Tipp: Fix multiple errors

You can fix multiple errors by selecting more than one row in the table with the *CTRL + click* action.

Finally, you can choose which *Attribute to use when merging features by attribute value*.

## 29.2.3 MetaSearch Catalog Client

### Einführung

MetaSearch is a QGIS plugin to interact with metadata catalog services, supporting both the OGC API - Records and OGC Catalog Service for the Web (CSW) standards.

MetaSearch provides an easy and intuitive approach and user-friendly interface to searching metadata catalogs within QGIS.

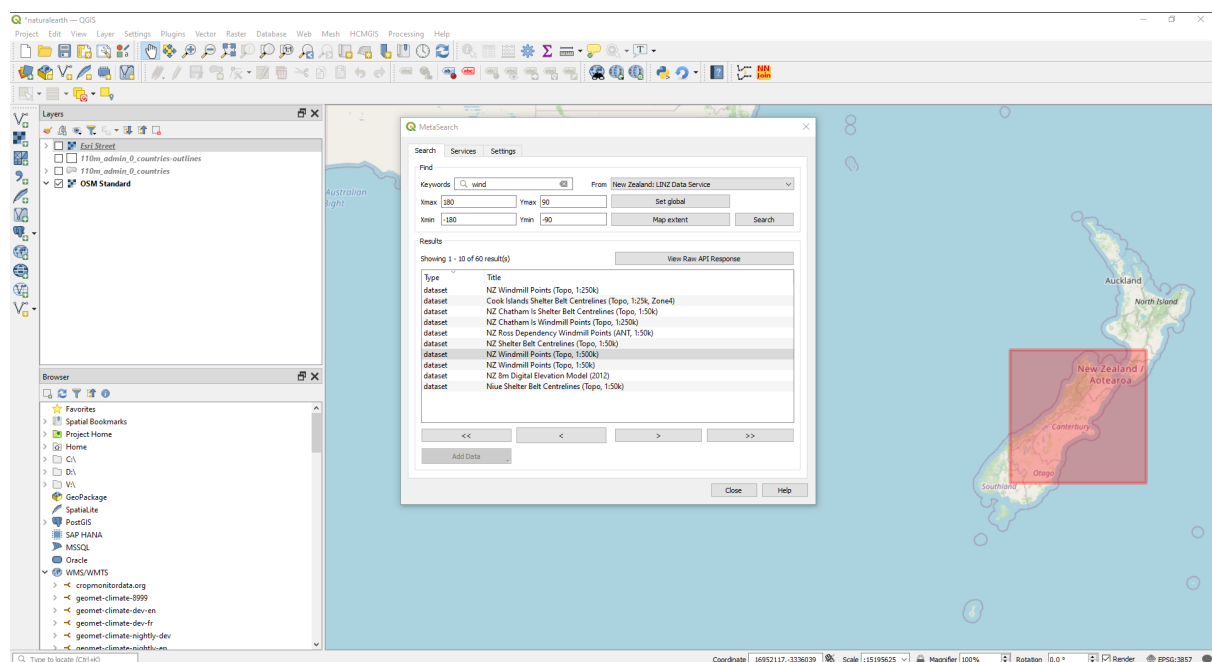


Abb. 29.8: Search and results of Services in MetaSearch

## Working with Metadata Catalogs in QGIS

MetaSearch is included by default in QGIS, with all of its dependencies, and can be enabled from the QGIS Plugin Manager.


### OGC API - Records

OGC API - Records is an OGC (Open Geospatial Consortium) standard for the discovery of geospatial resources on the Web.

### CSW (Catalog Service for the Web)

CSW (Catalog Service for the Web) is an OGC (Open Geospatial Consortium) specification that defines common interfaces to discover, browse and query metadata about data, services, and other potential resources.

### Startup

To start MetaSearch, click the  icon or select *Web ► MetaSearch ► MetaSearch* via the QGIS main menu. The MetaSearch dialog will appear. The main GUI consists of three tabs: *Services*, *Search* and *Settings*.

## Managing Catalog Services

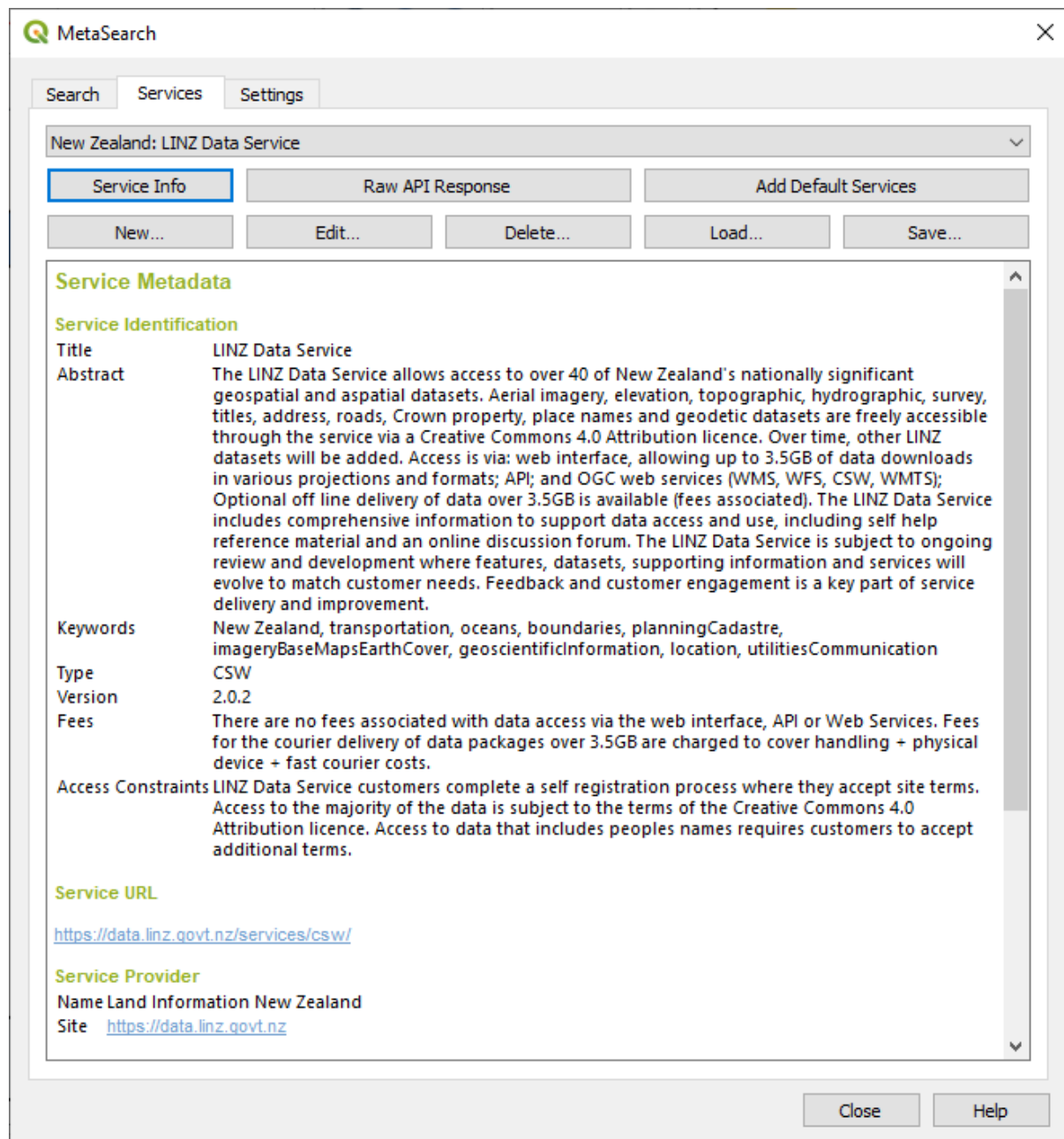


Abb. 29.9: Managing Catalog Services

The *Services* tab allows the user to manage all available catalog services. MetaSearch provides a default list of Catalog Services, which can be added by pressing the *Add Default Services* button.

To find all listed Catalog Service entries, click the dropdown select box.

To add a Catalog Service entry:

1. Click the *New* button
2. Enter a *Name* for the service, as well as the *URL* (endpoint). Note that for OGC CSW 2.0.2 Catalogs, only the base URL is required (not a full GetCapabilities URL). For OGC API - Records Catalogs, the URL should be the path to the collection endpoint
3. If the Catalog requires authentication, enter the appropriate *User name* and *Password* credentials.



4. Click *OK* to add the service to the list of entries.

To edit an existing Catalog Service entry:

1. Select the entry you would like to edit
2. Click the *Edit* button
3. And modify the *Name* or *URL* values
4. Click *OK*.

To delete a Catalog Service entry, select the entry you would like to delete and click the *Delete* button. You will be asked to confirm deleting the entry.

MetaSearch allows loading and saving connections to an XML file. This is useful when you need to share settings between applications. Below is an example of the XML file format.

```
<?xml version="1.0" encoding="UTF-8"?>
<qgsCSWConnections version="1.0">
  <csw type="OGC CSW 2.0.2" name="Data.gov CSW" url="https://catalog.data.gov/
  ↪ csw-all"/>
  <csw type="OGC CSW 2.0.2" name="Geonorge - National CSW service for Norway" ↪
  ↪ url="https://www.geonorge.no/geonetwork/srv/eng/csw"/>
  <csw type="OGC CSW 2.0.2" name="Geoportale Nazionale - Servizio di ricerca ↪
  ↪ Italiano" url="http://www.pcn.minambiente.it/geoportal/csw"/>
  <csw type="OGC CSW 2.0.2" name="LINZ Data Service" url="http://data.linz.govt.
  ↪ nz/feeds/csw"/>
  <csw type="OGC CSW 2.0.2" name="Nationaal Georegister (Nederland)" url="http://
  ↪ www.nationaalgeoregister.nl/geonetwork/srv/eng/csw"/>
  <csw type="OGC CSW 2.0.2" name="RNDT - Repertorio Nazionale dei Dati ↪
  ↪ Territoriali - Servizio di ricerca" url="http://www.rndt.gov.it/RNDT/CSW"/>
  <csw type="OGC CSW 2.0.2" name="UK Location Catalogue Publishing Service" url=
  ↪ "http://csw.data.gov.uk/geonetwork/srv/en/csw"/>
  <csw type="OGC CSW 2.0.2" name="UNEP/GRID-Geneva Metadata Catalog" url="http://
  ↪ metadata.grid.unep.ch:8080/geonetwork/srv/eng/csw"/>
</qgsCSWConnections>
```

To load a list of entries:

1. Click the *Load* button. A new window will appear.
2. Click the *Browse* button and navigate to the XML file of entries you wish to load.
3. Click *Open*. The list of entries will be displayed.
4. Select the entries you wish to add from the list and click *Load*.

Click the *Service Info* button to display information about the selected Catalog Service such as service identification, service provider and contact information. If you would like to view the raw API response, click the *Raw API Response* button. A separate window will open displaying server information in raw JSON or XML format.

## Searching Catalog Services

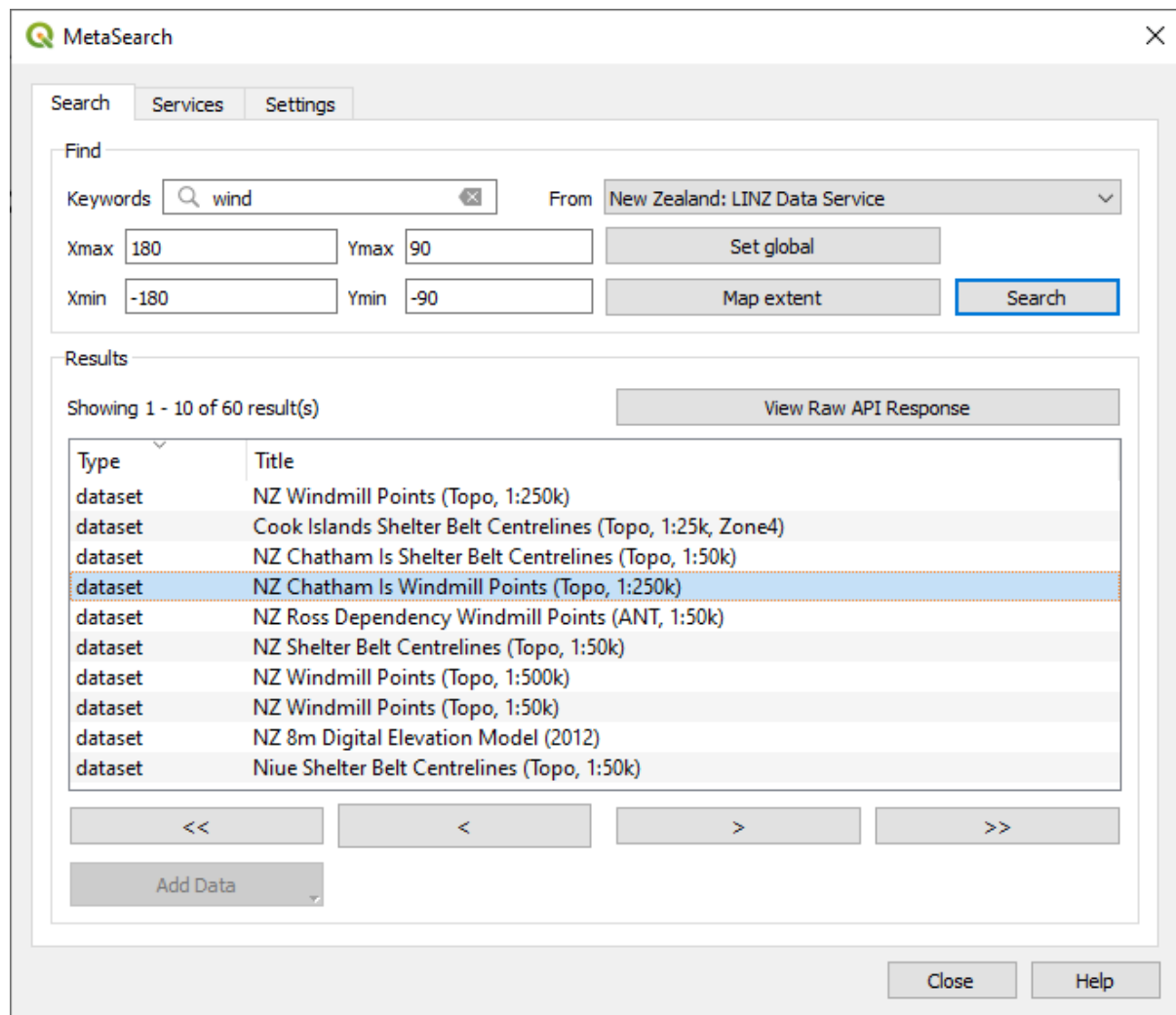


Abb. 29.10: Searching catalog services

The *Search* tab allows the user to query Catalog Services for data and services, set various search parameters and view results.

The following search parameters are available:

- **Keywords:** free text search keywords;
- **From:** the Catalog Service to perform the query against;
- **Bounding box:** the spatial area of interest to filter, defined by *Xmax*, *Xmin*, *Ymax*, and *Ymin*. Click *Set Global* to do a global search, click *Map Extent* to do a search in the visible area, or enter values manually.

Clicking the *Search* button will search the selected Metadata Catalog. Search results are displayed in a list, and can be sorted by clicking on the column header. You can navigate through search results with the directional buttons below the search results.

Select a result and:

- Click the *View Raw API Response* button to open a window with the service response in raw JSON or XML format.
- If the metadata record has an associated bounding box, a footprint of the bounding box will be displayed on the map.

- Double-click the record to display the record metadata with any associated access links. Clicking a link opens the link in the user's web browser.
- If the record is a supported web service (WMS/WMTS, WFS, WCS, ArcGIS REST Service, etc.) or a GIS File, the *Add Data* button will be enabled. When clicking this button, MetaSearch will verify if this is a valid OWS. The service will then be added to the appropriate QGIS connection list, and the appropriate connection dialog will appear. If you choose *Add GIS File* the linked GIS file will be added to the current project.

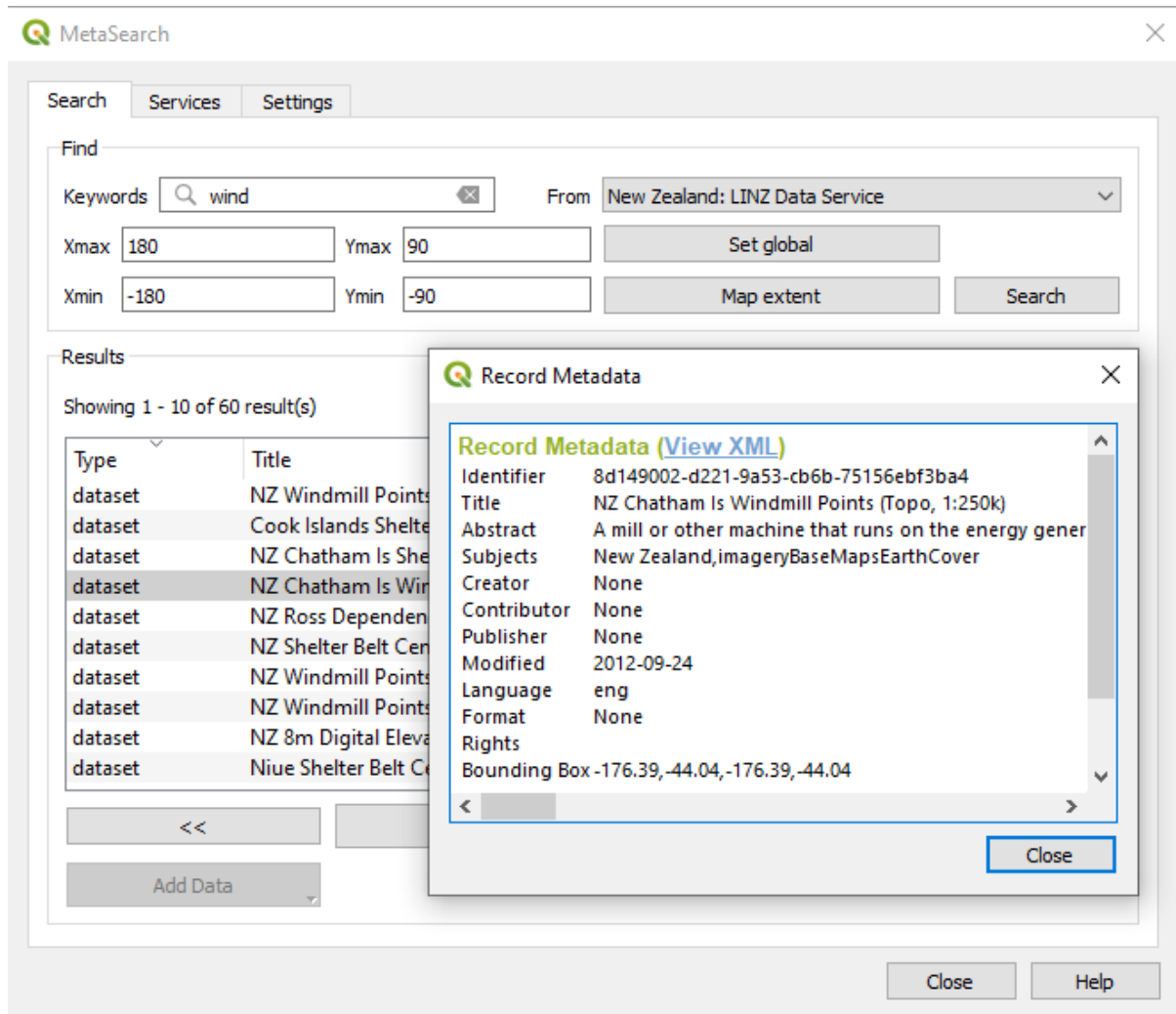


Abb. 29.11: Metadata record display

## Einstellungen

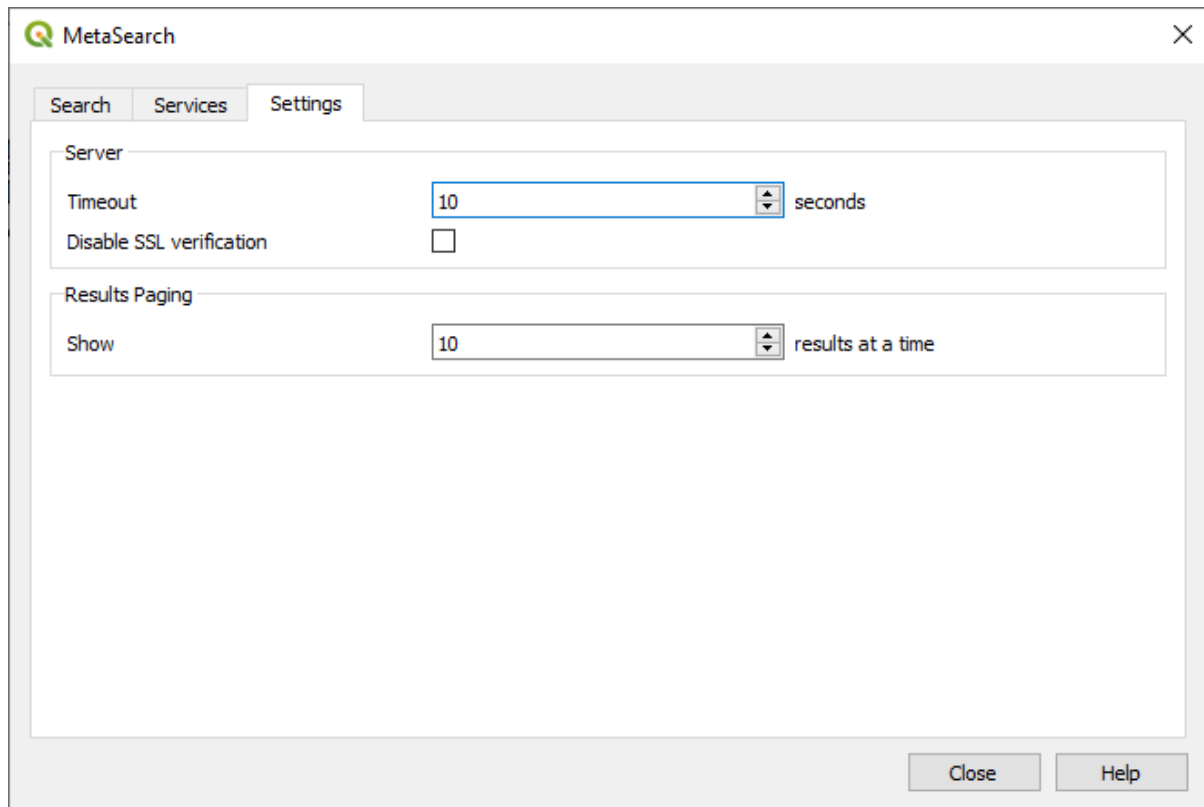


Abb. 29.12: MetaSearch settings

You can fine tune MetaSearch with the following *Settings*:


- *Server Timeout*: when searching metadata catalogs, the number of seconds for blocking connection attempt. Default value is 10.
- *Disable SSL verification*: option to switch off ssl verification.
- *Results paging*: when searching metadata catalogs, the number of results to show per page. Default value is 10.

## Catalog Server Errors


In some cases, the Catalog will work in a web browser, but not in MetaSearch. This may be due to the Catalog server's configuration/setup. Catalog server providers should ensure URLs are consistent and up to date in their configuration (this is common in HTTP -> HTTPS redirection scenarios). Please see the [pycsw FAQ item](#) for a deeper explanation of the issue and fix. Although the FAQ item is pycsw specific it can also apply in general to other Catalog servers.

### 29.2.4 Offline Editing Plugin

For data collection, it is a common situation to work with a laptop or a cell phone offline in the field. Upon returning to the network, the changes need to be synchronized with the master datasource (e.g., a PostGIS database). If several persons are working simultaneously on the same datasets, it is difficult to merge the edits by hand, even if people don't change the same features.

The  Offline Editing Plugin automates the synchronisation by copying the content of the datasource to a SpatiaLite or GeoPackage database and storing the offline edits to dedicated tables. After being connected to the network again, it is possible to apply the offline edits to the master dataset.



To use the plugin:


1. Open a project with some vector layers (e.g., from an Esri Shapefile, PostGIS or WFS-T datasource).
2. Assuming you have already enabled the plugin (see *Kernerweiterungen und externe Erweiterungen*) go to *Database ► Offline Editing ►*  *Convert to offline project*. The eponym dialog opens.
3. Select the *Storage type*. It can be of *GeoPackage* or *SpatiaLite* database type.
4. Use the *Browse* button to indicate the location of the database in which to store the *Offline data*. It can be an existing file or one to create.
5. In the *Select remote layers* section, check the layers you'd like to save. The content of the layers is saved to database tables.

**Bemerkung:** Since target database formats do not have native list support, the offline editing plugin transforms {string, number} list fields into string fields where values are separated by commas. This allows reading and edit of the contents of those fields when offline.

If you would like to handle both the field from the original layer and the offline layer, you can rely on the *try()* and *array* expression functions, e.g.:

```
try(array_contains("field",1),array_contains(string_to_array("field"),1))
```

6. You can check  *Only synchronize selected features if a selection is present* allowing to only save and work on a subset. It can be invaluable in case of large layers.  
This is all!
7. Save your project and bring it on the field.
8. Edit the layers offline.
9. After being connected again, upload the changes using *Database ► Offline Editing ►*  *Synchronize*.

**Bemerkung:** Layers that are used offline are marked with the  icon in the *Layers* panel.

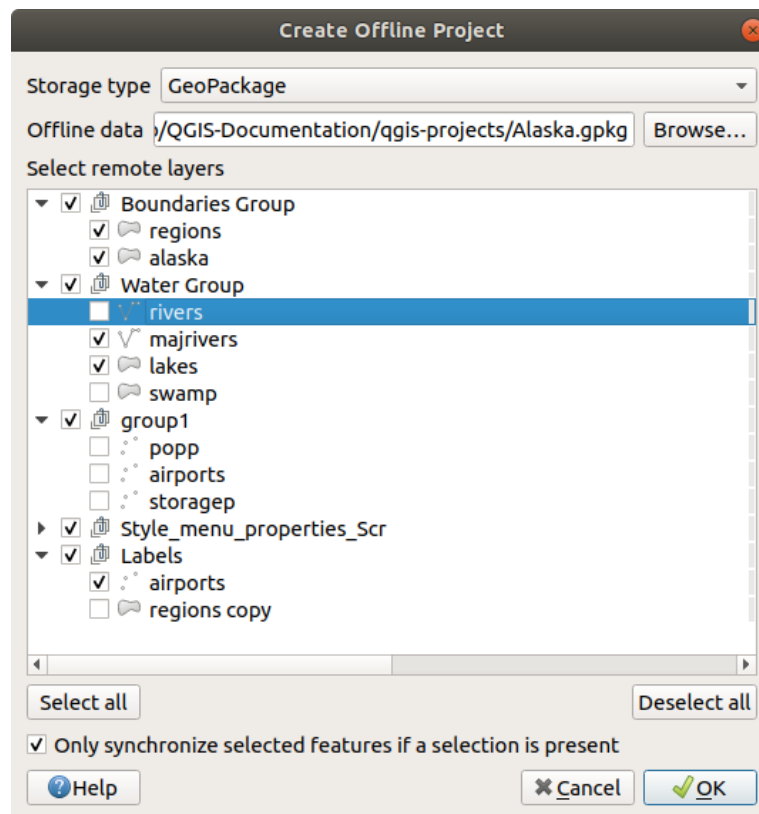


Abb. 29.13: Create an offline project

## 29.2.5 Topology Checker Plugin

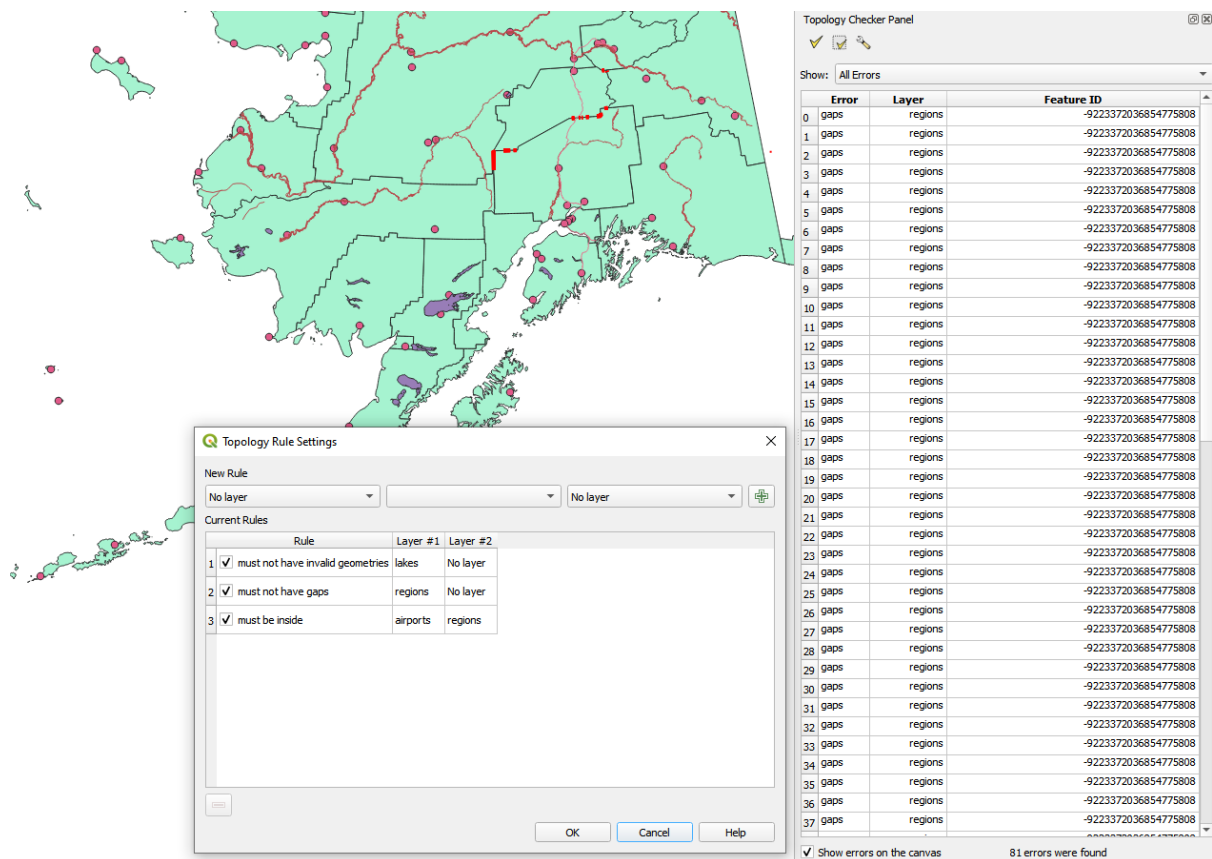








Abb. 29.14: The Topology Checker Plugin

Topology describes the relationships between points, lines and polygons that represent the features of a geographic region. With the Topology Checker plugin, you can look over your vector files and check the topology with several topology rules. These rules check with spatial relations whether your features ,Equal', ,Contain', ,Cover', are ,CoveredBy', ,Cross', are ,Disjoint', ,Intersect', ,Overlap', ,Touch' or are ,Within' each other. It depends on your individual questions which topology rules you apply to your vector data (e.g., normally you won't accept overshoots in line layers, but if they depict dead-end streets you won't remove them from your vector layer).

QGIS has a built-in topological editing feature, which is great for creating new features without errors. But existing data errors and user-induced errors are hard to find. This plugin helps you find such errors through a list of rules.

To enable the *Topology checker* plugin:

1. Go to *Plugins* menu
2. Open  *Manage and Install plugins* and choose  *Installed*
3. Enable  *Topology checker*
4. Close the *Plugin manager* dialog. A  *Topology checker* entry is added to the *Vector* menu.

After you enabled  *Topology checker* open it and choose  *Configure* to create your topology rules.

On **point layers** the following rules are available:

- *Must be covered by*: Here you can choose a vector layer from your project. Points that aren't covered by the given vector layer occur in the ,Error' field.
- *Must be covered by endpoints of*: Here you can choose a line layer from your project.


- *Must be inside:* Here you can choose a polygon layer from your project. The points must be inside a polygon. Otherwise, QGIS writes an ‚Error‘ for the point.
- *Must not have duplicates:* Whenever a point is represented twice or more, it will occur in the ‚Error‘ field.
- *Must not have invalid geometries:* Checks whether the geometries are valid.
- *Must not have multi-part-geometries:* All multi-part points are written into the ‚Error‘ field.


On **line layers**, the following rules are available:

- *End points must be covered by:* Here you can select a point layer from your project.
- *Must not have dangles:* This will show the overshoots in the line layer.
- *Must not have duplicates:* Whenever a line feature is represented twice or more, it will occur in the ‚Error‘ field.
- *Must not have invalid geometries:* Checks whether the geometries are valid.
- *Must not have multi-part geometries:* Sometimes, a geometry is actually a collection of simple (single-part) geometries. Such a geometry is called multi-part geometry. If it contains just one type of simple geometry, we call it multi-point, multi-linestring or multi-polygon. All multi-part lines are written into the ‚Error‘ field.
- *Must not have pseudos:* A line geometry’s endpoint should be connected to the endpoints of two other geometries. If the endpoint is connected to only one other geometry’s endpoint, the endpoint is called a pseudo node.

On **polygon layers**, the following rules are available:



- *Must contain:* Polygon layer must contain at least one point geometry from the second layer.
- *Must not have duplicates:* Polygons from the same layer must not have identical geometries. Whenever a polygon feature is represented twice or more it will occur in the ‚Error‘ field.
- *Must not have gaps:* Adjacent polygons should not form gaps between them. Administrative boundaries could be mentioned as an example (US state polygons do not have any gaps between them...).
- *Must not have invalid geometries:* Checks whether the geometries are valid. Some of the rules that define a valid geometry are:
  - Polygon rings must close.
  - Rings that define holes should be inside rings that define exterior boundaries.
  - Rings may not self-intersect (they may neither touch nor cross one another).
  - Rings may not touch other rings, except at a point.
- *Must not have multi-part geometries:* Sometimes, a geometry is actually a collection of simple (single-part) geometries. Such a geometry is called multi-part geometry. If it contains just one type of simple geometry, we call it multi-point, multi-linestring or multi-polygon. For example, a country consisting of multiple islands can be represented as a multi-polygon.
- *Must not overlap:* Adjacent polygons should not share common area.
- *Must not overlap with:* Adjacent polygons from one layer should not share common area with polygons from another layer.

When you create a *New rule* click on the  **Add rule** to include it to the *Current rules*. You can enable or disable individual rules by clicking on the checkbox. Right-clicking over a rule provides the following options:


- *Select All* the rules
- *Activate* or *Deactivate* the selected rules
- *Toggle activation* of selected rules
- *Delete* selected rules. This can also be achieved with the  **Delete selected rules** button.

Press **OK** and then choose from the *Topology checker* panel:












-  **Validate All**: applies the active rules to all the features of the involved layer(s)
- or  **Validate Extent**: applies the active rules to the features of the involved layer(s), within the current map canvas. The button is kept pushed and the results will update as the map canvas extent changes.

Errors will show up in the table of results containing type of error, layer and feature ID. Use *Filter errors by rule* menu to filter the errors to a specific error type.

Check  **Show errors on the canvas** to show error location on the canvas. Clicking a row in the table will zoom the map canvas to the concerned feature, where you can use *QGIS digitizing tools* to fix the error.

Below is the list of Core plugins provided with QGIS. They are not necessarily enabled by default.

Icon	Plugin	Beschreibung	Manual Reference
	DB Manager	Manage your databases within QGIS	<i>DB Manager Plugin</i>
	Geometrieprüfung	Check and repair errors in vector geometries	<i>Geometry Checker Plugin</i>
	GRASS 7	GRASS functionality	<i>GRASS GIS Integration</i>
	GRASS GIS provider	GRASS GIS Processing functionality	<i>GRASS GIS Integration</i>
	MetaSearch Client	Catalog Interact with metadata catalog services (CSW)	<i>MetaSearch Catalog Client</i>
	Offline Editing	Offline editing and synchronizing with database	<i>Offline Editing Plugin</i>
	OrfeoToolbox provider	OrfeoToolbox Processing provider	<i>OTB Bereitstellung</i> <i>Anwendungs</i>
	Verarbeitung	Spatial data processing framework	<i>QGIS Umgebung</i> <i>Verarbeitung</i>
	Topology Checker	Find topological errors in vector layers	<i>Topology Checker Plugin</i>

**Bemerkung:** To use the Core Plugins  GRASS 7,  GRASS GIS provider, or  OrfeoToolbox provider, they have to be configured. Informations can be found [here](#).

## 29.3 QGIS Python Konsole

As you will see later in this chapter, QGIS has been designed with a plugin architecture. Plugins can be written in Python, a very famous language in the geospatial world.

QGIS brings a Python API (see PyQGIS Developer Cookbook for some code sample) to let the user interact with its objects (layers, feature or interface). QGIS also has a Python console.

The QGIS Python Console is an interactive shell for Python command executions. It also has a Python file editor that allows you to edit and save your Python scripts. Both console and editor are based on PyQScintilla2 package. To open the console go to *Plugins ► Python Console* (Ctrl+Alt+P).

## 29.3.1 The Interactive Console

The console is a Python interpreter that allows you to execute Python commands. Modules from QGIS (analysis, core, gui, server, processing, 3d) and Qt (QtCore, QtGui, QtNetwork, QtWidgets, QtXml) as well as Python's math, os, re and sys modules are already imported and can be used directly.

The interactive console is composed of a toolbar, an input area and an output one.



Abb. 29.15: The Python Console

### Werkzeugeleiste

The toolbar proposes the following tools:

- Clear Console to wipe the output area;
- Run Command available in the input area; same as pressing Enter;
- Show Editor: toggles *The Code Editor* visibility;
- Options...: opens a dialog to configure *console properties*;
- Help... provides a menu to access various documentation:
  - *Python Console Help* (the current page)
  - PyQGIS API documentation
  - PyQGIS Cookbook
- Dock Code Editor to dock or undock the panel in QGIS interface

### Input area

The Console input area main features are:

- Code completion, highlighting syntax and calltips for the following APIs:
  - Python
  - PyQGIS
  - PyQt5
  - QScintilla2
  - osgeo-gdal-ogr

- `Ctrl+Alt+Space` to view the auto-completion list if enabled in the *Python Console settings*;
- Execute code snippets from the input area by typing and pressing `Enter` or *Run Command*;
- Execute code snippets from the output area using the *Enter Selected* from the contextual menu or pressing `Ctrl+E`;
- Browse the command history from the input area using the `Up` and `Down` arrow keys and execute the command you want;
- `Ctrl+Shift+Space` to view the command history: double-clicking a row will execute the command. The *Command History* dialog can also be accessed from context menu of input area;
- Save and clear the command history. The history will be saved into the `console_history.txt` file under the active *user profile* folder;
- Type the following special commands:
  - `?` to show a help of the Python Console
  - `_api` to open **QGIS C++ API** documentation or `_api(object)` for a specific object documentation (in QGIS C++ API or Qt API documentation)
  - `_pyqgis` to open **QGIS Python API** documentation or `_pyqgis(object)` for a specific object documentation (in QGIS Python API or Qt API documentation)
  - `_cookbook` to open PyQGIS Cookbook.
  - `!` followed by a command to execute Shell commands from the Python Console. The console will start a subprocess, and forward its output to the Python Console Output. While the subprocess is running, the Python Console Input switches to STDIN mode and forwards entered character to the child process. This makes it possible to send confirmation when the child program asks for it. When the Console is in STDIN mode, pressing `Ctrl+C` kills the subprocess. It is also possible to affect the result of a command to a variable with the syntax `var = !cmd`

```
>>> !echo QGIS Rocks!
QGIS Rocks

>>> !gdalinfo --version
GDAL 3.6.2, released 2023/01/02

>>> !pip install black
# Install black python formatter using pip (if available)

>>> sql_formats = !ogrinfo --formats | grep SQL
>>> sql_formats
['SQLite -vector- (rw+v): SQLite / Spatialite', ' MSSQLSpatial -vector-
↪(rw+): Microsoft SQL Server Spatial Database', ' PostgreSQL -vector-
↪(rw+): PostgreSQL/PostGIS', ' MySQL -vector- (rw+): MySQL', ' PGDUMP -
↪vector- (w+v): PostgreSQL SQL dump']
```


---



### **Tipp: Reuse executed commands from the output panel**

You can execute code snippets from the output panel by selecting some text and pressing `Ctrl+E`. No matter if selected text contains the interpreter prompt (`>>>`, `...`).

---

## 29.3.2 The Code Editor

Use the  **Show Editor** button to enable the editor widget. It allows editing and saving Python files and offers advanced functionalities to manage your code (comment and uncomment code, check syntax, share the code via GitHub and much more). Main features are:

- Code completion, highlighting syntax and calltips for the following APIs:
  - Python
  - PyQGIS
  - PyQt5
  - QScintilla2
  - osgeo-gdal-ogr
- **Ctrl+Space** to view the auto-completion list.
- Sharing code snippets via [GitHub](#).
- **Ctrl+4** Syntax check.
- Search bar (open it with the default Desktop Environment shortcut, usually **Ctrl+F**):
  - Use the default Desktop Environment shortcut to find next/previous (**Ctrl+G** and **Shift+Ctrl+G**);
  - Automatically find first match when typing in find box;
  - Set initial find string to selection when opening find;
  - Pressing **Esc** closes the find bar.
- Object inspector: a class and function browser;
- Go to an object definition with a mouse click (from Object inspector);
- Execute code snippets with the  **Run Selected** command in contextual menu;
- Execute the whole script with the  **Run Script** command (this creates a byte-compiled file with the extension `.pyc`).

---

**Bemerkung:** Running partially or totally a script from the *Code Editor* outputs the result in the Console output area.

---

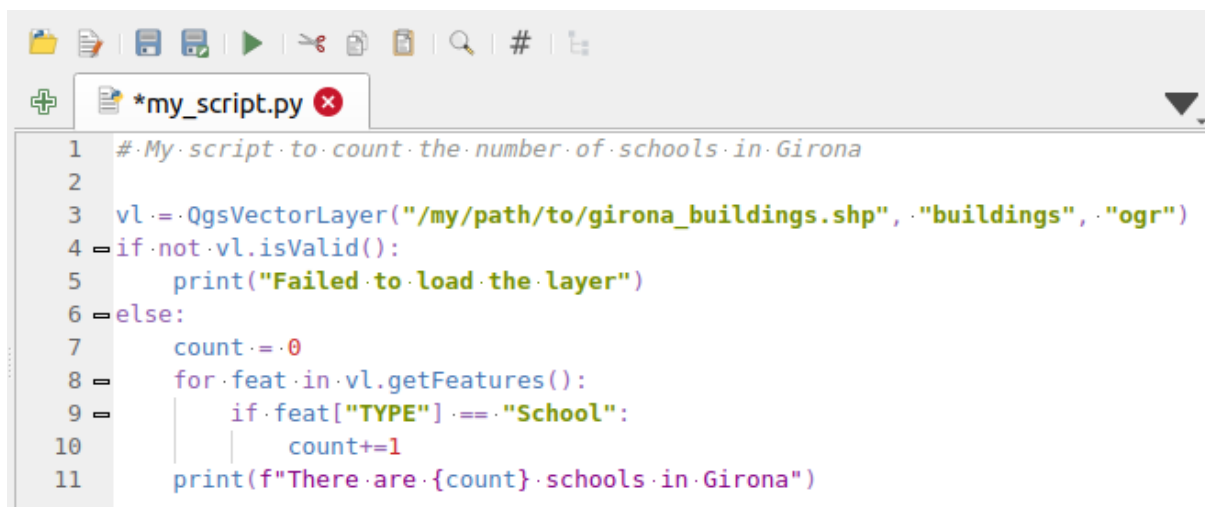


Abb. 29.16: The Python Console editor

---

**Tipp: Save the options**

To save the state of console's widgets you have to close the Python Console from the close button. This allows you to save the geometry to be restored to the next start.

---



### 30.1 Mailinglisten

QGIS entwickelt sich ständig weiter, daher kann es vorkommen, dass es mal nicht so funktioniert, wie erwartet. Die bevorzugte und effektivste Art, Hilfe zu bekommen, besteht darin, sich in die qgis-users Mailingliste einzuschreiben. Ihre Fragen erreichen eine breite Basis von Anwendern und die Antworten auf Ihre Fragen können auch anderen helfen.

#### 30.1.1 QGIS Users

This mailing list is used for discussion about QGIS in general, as well as specific questions regarding its installation and use. You can subscribe to the qgis-users mailing list by visiting the following URL: <https://lists.osgeo.org/mailman/listinfo/qgis-user>

#### 30.1.2 QGIS Developers

Wenn Sie Entwickler sind und eher technische Probleme haben, können Sie sich in die englischsprachige qgis-developer Mailingliste eintragen. Diese Liste ist auch der richtige Ort, an dem Sie sich zu Themen der Bedienbarkeit von QGIS äußern können und die dort gesammelt und diskutiert werden. Die Liste ist unter der URL: <https://lists.osgeo.org/mailman/listinfo/qgis-developer> erreichbar.

#### 30.1.3 QGIS Community Team

This list deals with topics like documentation, context help, user guide, web sites, blog, mailing lists, forums, and translation efforts. If you would like to work on the user guide as well, this list is a good starting point to ask your questions. You can subscribe to this list at: <https://lists.osgeo.org/mailman/listinfo/qgis-community-team>

### 30.1.4 QGIS Translations

This list deals with the translation efforts. If you like to work on the translation of the website, manuals or the graphical user interface (GUI), this list is a good starting point to ask your questions. You can subscribe to this list at: <https://lists.osgeo.org/mailman/listinfo/qgis-tr>

### 30.1.5 QGIS Project Steering Committee (PSC)

This list is used to discuss Steering Committee issues related to overall management and direction of QGIS. You can subscribe to this list at: <https://lists.osgeo.org/mailman/listinfo/qgis-psc>

### 30.1.6 QGIS-Anwendergruppen

Einige QGIS-Gemeinschaften sind als QGIS-Anwendergruppen organisiert, um QGIS vor Ort zu fördern und zu seiner Weiterentwicklung beizutragen. Diese Gruppen sind der richtige Ort, um lokale Themen zu diskutieren, regionale oder nationale Anwendertreffen oder die Unterstützung für neue Programmfunktionen zu organisieren. Die Liste der Anwendergruppen finden Sie unter <https://qgis.org/en/site/forusers/usergroups.html>

You are welcome to subscribe to any of the lists. Please remember to contribute to the list by answering questions and sharing your experiences.

## 30.2 Matrix / IRC

Matrix (<https://matrix.org>) is a project for decentralised chat. QGIS has a room accessible by the #qgis:osgeo.org alias (among others) which bridges to the libera.chat #qgis IRC channel.

To use Matrix:

1. Create an account (at matrix.org is easiest, but if you have an OSGeo account you can also use your OSGeo id as matrix id)
2. Install a client (Elements is easiest, but see [Matrix clients](#) for more), OR point your browser to <https://matrix.to/#/#qgis:osgeo.org>

To use IRC:

1. Install an IRC client
2. Connect to <irc://irc.libera.chat/#qgis> or point your browser to <https://web.libera.chat/?channels=#qgis>

## 30.3 Kommerzieller Support

Commercial support for QGIS is also available. Check the website [https://qgis.org/en/site/forusers/commercial\\_support.html](https://qgis.org/en/site/forusers/commercial_support.html) for more information.



## 30.4 BugTracker

While the qgis-users mailing list is useful for general ,How do I do XYZ in QGIS?-type questions, you may wish to notify us about bugs in QGIS. You can submit bug reports using the [QGIS bug tracker](#).

Denken Sie auch bitte daran, dass ein für Sie wichtiger Fehler nicht immer die gleiche Priorität bei anderen Personen und besonders den Entwicklern hat. Einige Fehler sind sehr aufwendig zu reparieren und daher kann es schon mal ein wenig dauern, bis genügend Zeit vorhanden ist, ein Problem zu lösen.

Feature requests can be submitted as well using the same ticket system as for bugs. Please make sure to select the type `Feature request`.

If you have found a bug and fixed it yourself, you can submit a Pull Request on the [Github QGIS Project](#).

Read [Bugs](#), [Features](#) and [Issues](#) and `submit_patch` for more details.

## 30.5 Blog

The QGIS community also runs a weblog at <https://plugins.qgis.org/planet/>, which has some interesting articles for users and developers. Many other QGIS blogs exist, and you are invited to contribute with your own QGIS blog!

## 30.6 Plugins

The website <https://plugins.qgis.org> is the official QGIS plugins web portal. Here, you find a list of all stable and experimental QGIS plugins available via the ,Official QGIS Plugin Repository‘.

## 30.7 Wiki

Lastly, we maintain a WIKI web site at <https://github.com/qgis/QGIS/wiki> where you can find a variety of useful information relating to QGIS development, release plans, links to download sites, message-translation hints and more. Check it out, there are some goodies inside!



## Mitwirkende

QGIS ist ein Open-Source-Projekt, das von einem Team aus engagierten Freiwilligen und Organisationen entwickelt wird. Wir bemühen uns, eine einladende Gemeinschaft für Menschen aller Rassen, Glaubensrichtungen, Geschlechter und Lebensbereiche zu sein. Sie können sich jederzeit [einbringen](#).

### 31.1 Autoren

Nachfolgend sind die Personen aufgeführt, die ihre Zeit und Energie dem Schreiben, Überprüfen und Aktualisieren der gesamten QGIS-Dokumentation widmen.

Tim Sutton	Yves Jacolin	Jacob Lanstorp	Gary E. Sherman	Richard Duivenvoorde
Tara Athan	Anita Graser	Arnaud Morvan	Gavin Macaulay	Luca Casagrande
K. Koy	Hugo Mercier	Akbar Gumbira	Marie Silvestre	Jürgen E. Fischer
Fran Raga	Eric Goddard	Martin Dobias	Diethard Jansen	Saber Razmjooei
Ko Nagase	Nyall Dawson	Matthias Kuhn	Andreas Neumann	Harrissou Sant-anna
Manel Clos	David Willis	Larissa Junek	Paul Blottière	Sebastian Dietrich
Chris Mayo	Stephan Holl	Magnus Homann	Bernhard Ströbl	Alessandro Pasotti
N. Horning	Radim Blazek	Joshua Arnott	Luca Manganelli	Marco Hugentobler
Andre Mano	Mie Winstrup	Frank Sokolic	Vincent Picavet	Jean-Roc Morreale
Andy Allan	Victor Olaya	Tyler Mitchell	René-Luc D'Hont	Marco Bernasocchi
Ilkka Rinne	Werner Macho	Chris Berkhout	Nicholas Duggan	Jonathan Willits
David Adler	Lars Luthman	Brendan Morely	Raymond Nijssen	Carson J.Q. Farmer
Jaka Kranjc	Mezene Worku	Patrick Sunter	Steven Cordwell	Stefan Blumentrath
Andy Schmid	Vincent Mora	Alexandre Neto	Hien Tran-Quang	Alexandre Busquets
João Gaspar	Tom Kralidis	Alexander Bruy	Paolo Cavallini	Milo Van der Linden
Peter Ersts	Ujaval Gandhi	Dominic Keller	Giovanni Manghi	Maximilian Krambach
Anne Ghisla	Dick Groskamp	Uros Preloznik	Stéphane Brunner	QGIS Korean Translator
Zoltan Siki	Håvard Tveite	Matteo Ghetta	Salvatore Larosa	Konstantinos Nikolaou
Tom Chadwin	Larry Shaffer	Nathan Woodrow	Martina Savarese	Godofredo Contreras
Astrid Emde	Luigi Pirelli	Thomas Gratier	Giovanni Allegri	GiordanoPezzola
Paolo Corti	Tudor Băräscu	Maning Sambale	Claudia A. Engel	Yoichi Kayama
Otto Dassau	Denis Rouzaud	Nick Bearman	embelding	ajazepk
Ramon	Andrei	zstadler	icephale	Rosa Aguilar
Patrice Pineault	Jörn Gutzeit	Felix Feckler	Benoît de Mezzo	Étienne Trimaille

Fortsetzung auf der nächsten Seite

Tab. 31.1 – Fortsetzung der vorherigen Seite

Andrea Giudiceandrea	Julien Cabieces	roya0045	Sebastian Gutwein	Jessica Veenstra
Ryan Welfle	Martin Pergler	Ivan Ivanov	muranamihdk	Loïc Bartoletti
Tomasz Taraś	Ian Maddaus	Jürnjakob Dugge	Roman Bug	Damiano Lombardi
Marc Ducobu	Philip Albrecht	Dennis Milechin	Cody Martin	Savinaud Mickaël
Stefan Uhrig	Ariadni-Karolina Alexiou	Björn Hinkeldey	Benjamin Riley	MorriganR
Thayer Young	Shane Carey	Ian Turton	Emma Hain	Germán Carrillo
Jakob Miksch	Nicolas Boisteault	Bertrand Rix	Jorge Rosales	

## 31.2 Übersetzer

QGIS ist eine mehrsprachige Anwendung und veröffentlicht auch eine in mehrere Sprachen übersetzte Dokumentation. Viele andere Sprachen werden übersetzt und werden veröffentlicht, sobald sie einen angemessenen Übersetzungsanteil erreichen. Wenn Sie helfen möchten, eine Sprache zu verbessern oder eine neue anzufordern, besuchen Sie bitte <https://qgis.org/en/site/getinvolved/index.html>.

Die aktuellen Übersetzungen werden ermöglicht durch:

Sprache	Mitwirkende
Bahasa Indonesia	Emir Hartato, I Made Anombawa, Januar V. Simarmata, Muhammad Iqnaul Haq Siregar, Trias Aditya
Chinese (Traditional)	Calvin Ngei, Zhang Jun, Richard Xie
Chinese (Simplified)	Xu Baocai
Dutch	Carlo van Rijswijk, Dick Groskamp, Diethard Jansen, Raymond Nijssen, Richard Duivenvoorde, Willem Hoffman
Finnish	Matti Mäntynen, Kari Mikkonen
French	Arnaud Morvan, Augustin Roche, Didier Vanden Berghe, Dofabien, Etienne Trimaille, Francis Gasc, Harrissou Sant-anna, Jean-Roc Morreale, Jérémy Garniaux, Loïc Buscoz, Lsam, Marc-André Saia, Marie Silvestre, Mathieu Bossaert, Mathieu Lattes, Mayeul Kauffmann, Médéric Ribreux, Mehdi Semchaoui, Michael Douchin, Nicolas Boisteault, Nicolas Rochard, Pascal Obstetar, Robin Prest, Rod Bera, Stéphane Henriod, Stéphane Possamai, sylther, Sylvain Badey, Sylvain Maillard, Vincent Picavet, Xavier Tardieu, Yann Leveille-Menez, yoda89
Galician	Xan Vieiro
German	Jürgen E. Fischer, Otto Dassau, Stephan Holl, Werner Macho
Hindi	Harish Kumar Solanki
Italian	Alessandro Fanna, Anne Ghisla, Flavio Rigolon, Giuliano Curti, Luca Casagrande, Luca Delucchi, Marco Braida, Matteo Ghetta, Maurizio Napolitano, Michele Beneventi, Michele Ferretti, Roberto Angeletti, Paolo Cavallini, Stefano Campus
Japanese	Baba Yoshihiko, Minoru Akagi, Norihiro Yamate, Takayuki Mizutani, Takayuki Nuimura, Yoichi Kayama
Korean	OSGeo Korean Chapter
Polish	Andrzej Świąder, Borys Jurgiel, Ewelina Krawczak, Jakub Bobrowski, Mateusz Łoskot, Michał Kułach, Michał Smoczyk, Milena Nowotarska, Radosław Pasiok, Robert Szczepanek, Tomasz Paul
Portuguese	Alexandre Neto, Duarte Carreira, Giovanni Manghi, João Gaspar, Joana Simões, Leandro Infantini, Nelson Silva, Pedro Palheiro, Pedro Pereira, Ricardo Sena
Portuguese (Brasil)	Arthur Nanni, Felipe Sodr� Barros, Le�nidas Descovi Filho, Marcelo Soares Souza, Narc�lio de S� Pereira Filho, Sidney Schaberle Goveia
Romanian	Alex B�descu, Bogdan Pacurar, Georgiana Ioanovici, Lonut Losifescu-Enescu, Sorin C�linic�, Tudor B�r�scu
Russian	Alexander Bruy, Artem Popov
Spanish	Carlos D�vila, Diana Galindo, Edwin Amado, Gabriela Awad, Javier C�sar Aldariz, Mayeul Kauffmann, Fran Raga
Ukrainian	Alexander Bruy

## 31.3  bersetzungsstatistiken

Die  bersetzungsbem hungen f r QGIS 3.34 Long Term Release sind unten aufgef hrt. Nur die Sprachen, die zum Zeitpunkt der Ver ffentlichung der Version 5% erreicht haben, werden hier ver ffentlicht.

*Last update: 2024-02-27*

Anzahl der Zeichenfolgen	Anzahl der Zielsprachen	Gesamt-�bersetzungsanteil
<b>29146</b>	<b>58</b>	<b>14.92%</b>

Sprache	Übersetzungsanteil (%)	Sprache	Übersetzungsanteil (%)	Sprache	Übersetzungsanteil (%)
Albanian	1.12	Arabic	4.63	Azerbaijani	0.87
Basque	1.89	Bengali	1.06	Bulgarian	3.41
Burmese	0.97	Catalan	2.03	Chinese Simplified	29.67
Chinese Traditional	2.2	Croatian	0.98	Czech	6.74
Danish	1.59	Dutch	100.0	Estonian	2.25
Finnish	2.22	French	85.75	Galician	1.36
Georgian	0.98	German	22.93	Greek	1.25
Hebrew	1.68	Hindi	1.19	Hungarian	19.22
Igbo	0.87	Indonesian	3.64	Italian	100.0
Japanese	99.93	Kabyle	0.97	Korean	86.8
Lithuanian	11.42	Macedonian	0.99	Malay	0.91
Malayalam	0.97	Marathi	1.01	Mongolian	0.98
N'ko	2.53	Norwegian Bokmål	3.73	Panjabi (Punjabi)	0.0
Persian	0.69	Polish	2.75	Portuguese (Brazil)	70.65
Portuguese (Portugal)	8.61	Romanian	34.92	Russian	15.47
Serbian	0.13	Slovak	2.44	Slovenian	3.66
Spanish	99.96	Swedish	1.95	Tagalog	0.97
Tamil	1.41	Telugu	0.89	Thai	0.98
Turkish	3.37	Ukrainian	3.68	Urdu	0.86
Vietnamese	1.11				

### 32.1 Anhang A: GNU General Public License

Deutsche Übersetzung der Version 2, Juni 1991. Den offiziellen englischen Originaltext finden Sie unter <http://www.gnu.org/licenses/gpl.html>. Diese Übersetzung wird mit der Absicht angeboten, das Verständnis der GNU General Public License (GNU GPL) zu erleichtern. Es handelt sich jedoch nicht um eine offizielle oder im rechtlichen Sinne anerkannte Übersetzung. Diese Übersetzung wurde ursprünglich erstellt von Katja Lachmann. Übersetzungen im Auftrag der S.u.S.E. GmbH - <http://www.suse.de>. Sie wurde überarbeitet von Peter Gerwinski, G-N-U GmbH - <http://www.g-n-u.de> (31. Oktober 1996, 4. Juni 2000)

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Es ist jedermann gestattet, diese Lizenzurkunde zu vervielfältigen und unveränderte Kopien zu verbreiten; Änderungen sind jedoch nicht erlaubt.

#### Vorwort

Die meisten Softwarelizenzen sind daraufhin entworfen worden, Ihnen die Freiheit zu nehmen, die Software weiterzugeben und zu verändern. Im Gegensatz dazu soll Ihnen die GNU General Public License, die Allgemeine Öffentliche GNU-Lizenz, ebendiese Freiheit garantieren. Sie soll sicherstellen, dass die Software für alle Benutzer frei ist. Diese Lizenz gilt für den Großteil der von der Free Software Foundation herausgegebenen Software und für alle anderen Programme, deren Autoren ihr Werk dieser Lizenz unterstellt haben. Auch Sie können diese Möglichkeit der Lizenzierung für Ihre Programme anwenden. (Ein anderer Teil der Software der Free Software Foundation unterliegt stattdessen der GNU Lesser General Public License, der Kleineren Allgemeinen Öffentlichen GNU-Lizenz).

Die Bezeichnung „freie“ Software bezieht sich auf Freiheit, nicht auf den Preis. Unsere Lizenzen sollen Ihnen die Freiheit garantieren, Kopien freier Software zu verbreiten (und etwas für diesen Service zu berechnen, wenn Sie möchten), die Möglichkeit, die Software im Quelltext zu erhalten oder den Quelltext auf Wunsch zu bekommen. Die Lizenzen sollen garantieren, dass Sie die Software ändern oder Teile davon in neuen freien Programmen verwenden dürfen - und dass Sie wissen, dass Sie dies alles tun dürfen.

Um Ihre Rechte zu schützen, müssen wir Einschränkungen machen, die es jedem verbieten, Ihnen diese Rechte zu verweigern oder Sie aufzufordern, auf diese Rechte zu verzichten. Aus diesen Einschränkungen folgen bestimmte Verantwortlichkeiten für Sie, wenn Sie Kopien der Software verbreiten oder sie verändern.

Beispielsweise müssen Sie den Empfängern alle Rechte gewähren, die Sie selbst haben, wenn Sie - kostenlos oder gegen Bezahlung - Kopien eines solchen Programms verbreiten. Sie müssen sicherstellen, dass auch die Empfänger den Quelltext erhalten bzw. erhalten können. Und Sie müssen ihnen diese Bedingungen zeigen, damit sie ihre Rechte kennen.

Wir schützen Ihre Rechte in zwei Schritten: (1) Wir stellen die Software unter ein Urheberrecht (Copyright), und (2) wir bieten Ihnen diese Lizenz an, die Ihnen das Recht gibt, die Software zu vervielfältigen, zu verbreiten und/oder zu verändern.

Um die Autoren und uns zu schützen, wollen wir darüberhinaus sicherstellen, dass jeder erfährt, dass für diese freie Software keinerlei Garantie besteht. Wenn die Software von jemand anderem modifiziert und weitergegeben wird, möchten wir, dass die Empfänger wissen, dass sie nicht das Original erhalten haben, damit irgendwelche von anderen verursachte Probleme nicht den Ruf des ursprünglichen Autors schädigen.

Schließlich und endlich ist jedes freie Programm permanent durch Software-Patente bedroht. Wir möchten die Gefahr ausschließen, dass Distributoren eines freien Programms individuell Patente lizensieren - mit dem Ergebnis, dass das Programm proprietär würde. Um dies zu verhindern, haben wir klargestellt, dass jedes Patent entweder für freie Benutzung durch jedermann lizenziert werden muss oder überhaupt nicht lizenziert werden darf.

Es folgen die genauen Bedingungen für die Vervielfältigung, Verbreitung und Bearbeitung

0. Diese Lizenz gilt für jedes Programm und jedes andere Werk, in dem ein entsprechender Vermerk des Copyright-Inhabers darauf hinweist, dass das Werk unter den Bestimmungen dieser General Public License verbreitet werden darf. Im folgenden wird jedes derartige Programm oder Werk als ‚das Programm‘ bezeichnet; die Formulierung ‚auf dem Programm basierendes Werk‘ bezeichnet das Programm sowie jegliche Bearbeitung des Programms im urheberrechtlichen Sinne, also ein Werk, welches das Programm, auch auszugsweise, sei es unverändert oder verändert und/oder in eine andere Sprache übersetzt, enthält. (Im folgenden wird die Übersetzung ohne Einschränkung als ‚Bearbeitung‘ eingestuft.) Jeder Lizenznehmer wird im folgenden als ‚Sie‘ angesprochen.

Andere Handlungen als Vervielfältigung, Verbreitung und Bearbeitung werden von dieser Lizenz nicht berührt; sie fallen nicht in ihren Anwendungsbereich. Der Vorgang der Ausführung des Programms wird nicht eingeschränkt, und die Ausgaben des Programms unterliegen dieser Lizenz nur, wenn der Inhalt ein auf dem Programm basierendes Werk darstellt (unabhängig davon, dass die Ausgabe durch die Ausführung des Programmes erfolgte). Ob dies zutrifft, hängt von den Funktionen des Programms ab.

1. Sie dürfen auf beliebigen Medien unveränderte Kopien des Quelltextes des Programms, wie sie ihn erhalten haben, anfertigen und verbreiten. Voraussetzung hierfür ist, dass Sie mit jeder Kopie einen entsprechenden Copyright-Vermerk sowie einen Haftungsausschluss veröffentlichen, alle Vermerke, die sich auf diese Lizenz und das Fehlen einer Garantie beziehen, unverändert lassen und desweiteren allen anderen Empfängern des Programms zusammen mit dem Programm eine Kopie dieser Lizenz zukommen lassen.

Sie dürfen für den eigentlichen Kopiervorgang eine Gebühr verlangen. Wenn Sie es wünschen, dürfen Sie auch gegen Entgelt eine Garantie für das Programm anbieten.

2. Sie dürfen Ihre Kopie(n) des Programms oder eines Teils davon verändern, wodurch ein auf dem Programm basierendes Werk entsteht; Sie dürfen derartige Bearbeitungen unter den Bestimmungen von Paragraph 1 vervielfältigen und verbreiten, vorausgesetzt, dass zusätzlich alle im folgenden genannten Bedingungen erfüllt werden:

- a) Sie müssen die veränderten Dateien mit einem auffälligen Vermerk versehen, der auf die von Ihnen vorgenommene Modifizierung und das Datum jeder Änderung hinweist.
- b) Sie müssen dafür sorgen, dass jede von Ihnen verbreitete oder veröffentlichte Arbeit, die ganz oder teilweise von dem Programm oder Teilen davon abgeleitet ist, Dritten gegenüber als Ganzes unter den Bedingungen dieser Lizenz ohne Lizenzgebühren zur Verfügung gestellt wird.
- c) Wenn das veränderte Programm normalerweise bei der Ausführung interaktiv Kommandos einliest, müssen Sie dafür sorgen, dass es, wenn es auf dem üblichsten Wege für solche interaktive Nutzung gestartet wird, eine Meldung ausgibt oder ausdruckt, die einen geeigneten Copyright-Vermerk enthält sowie einen Hinweis, dass es keine Gewährleistung gibt (oder anderenfalls, dass Sie Garantie leisten), und dass die Benutzer das Programm unter diesen Bedingungen weiter verbreiten dürfen. Auch muss der Benutzer darauf hingewiesen werden, wie er eine Kopie dieser Lizenz ansehen kann. (Ausnahme: Wenn



das Programm selbst interaktiv arbeitet, aber normalerweise keine derartige Meldung ausgibt, muss Ihr auf dem Programm basierendes Werk auch keine solche Meldung ausgeben.)

Diese Anforderungen gelten für das bearbeitete Werk als Ganzes. Wenn identifizierbare Teile des Werkes nicht von dem Programm abgeleitet sind und vernünftigerweise als unabhängige und eigenständige Werke für sich selbst zu betrachten sind, dann gelten diese Lizenz und ihre Bedingungen nicht für die betroffenen Teile, wenn Sie diese als eigenständige Werke weitergeben. Wenn Sie jedoch dieselben Abschnitte als Teil eines Ganzen weitergeben, das ein auf dem Programm basierendes Werk darstellt, dann muss die Weitergabe des Ganzen nach den Bedingungen dieser Lizenz erfolgen, deren Bedingungen für weitere Lizenznehmer somit auf das gesamte Ganze ausgedehnt werden - und somit auf jeden einzelnen Teil, unabhängig vom jeweiligen Autor.

Somit ist es nicht die Absicht dieses Abschnittes, Rechte für Werke in Anspruch zu nehmen oder Ihnen die Rechte für Werke streitig zu machen, die komplett von Ihnen geschrieben wurden; vielmehr ist es die Absicht, die Rechte zur Kontrolle der Verbreitung von Werken, die auf dem Programm basieren oder unter seiner auszugsweisen Verwendung zusammengestellt worden sind, auszuüben.

Ferner bringt auch das einfache Zusammenlegen eines anderen Werkes, das nicht auf dem Programm basiert, mit dem Programm oder einem auf dem Programm basierenden Werk auf ein- und demselben Speicher- oder Vertriebsmedium dieses andere Werk nicht in den Anwendungsbereich dieser Lizenz.

3. Sie dürfen das Programm (oder ein darauf basierendes Werk gemäß Paragraph 2) als Objectcode oder in ausführbarer Form unter den Bedingungen der Paragraphen 1 und 2 kopieren und weitergeben - vorausgesetzt, dass Sie außerdem eine der folgenden Leistungen erbringen:
  - a) Liefern Sie das Programm zusammen mit dem vollständigen zugehörigen maschinenlesbaren Quelltext auf einem für den Datenaustausch üblichen Medium aus, wobei die Verteilung unter den Bedingungen der Paragraphen 1 und 2 erfolgen muss. Oder,
  - b) Liefern Sie das Programm zusammen mit einem mindestens drei Jahre lang gültigen schriftlichen Angebot aus, jedem Dritten eine vollständige maschinenlesbare Kopie des Quelltextes zur Verfügung zu stellen - zu nicht höheren Kosten als denen, die durch den physikalischen Kopiervorgang anfallen -, wobei der Quelltext unter den Bedingungen der Paragraphen 1 und 2 auf einem für den Datenaustausch üblichen Medium weitergegeben wird. Oder,
  - c) Liefern Sie das Programm zusammen mit den Informationen, die Sie beim Angebot zur Verfügung stellen des Quelltextes, erhalten haben. (Diese Alternative ist nur für nicht-kommerzielle Verbreitung zulässig und nur, wenn Sie das Programm als Objektcode oder in ausführbarer Form mit einem entsprechenden Angebot erhalten haben, gemäß Absatz b oben.)

Unter dem Quelltext eines Werkes wird diejenige Form des Werkes verstanden, die für Bearbeitungen vorzugsweise verwendet wird. Für ein ausführbares Programm bedeutet „der komplette Quelltext“: Der Quelltext aller im Programm enthaltenen Module einschließlich aller zugehörigen Modulschnittstellen-Definitionsdateien sowie der zur Compilation und Installation verwendeten Skripte. Als besondere Ausnahme jedoch braucht der verteilte Quelltext nichts von dem zu enthalten, was üblicherweise (entweder als Quelltext oder in binärer Form) zusammen mit den Hauptkomponenten des Betriebssystems (Kernel, Compiler usw.) geliefert wird, unter dem das Programm läuft - es sei denn, diese Komponente selbst gehört zum ausführbaren Programm.

Wenn die Verbreitung eines ausführbaren Programms oder von Objectcode dadurch erfolgt, dass der Kopierzugriff auf eine dafür vorgesehene Stelle gewährt wird, so gilt die Gewährung eines gleichwertigen Zugriffs auf den Quelltext als Verbreitung des Quelltextes, auch wenn Dritte nicht dazu gezwungen sind, den Quelltext zusammen mit dem Objectcode zu kopieren.

4. Sie dürfen das Programm nicht vervielfältigen, verändern, weiter lizenzieren oder verbreiten, sofern es nicht durch diese Lizenz ausdrücklich gestattet ist. Jeder anderweitige Versuch der Vervielfältigung, Modifizierung, Weiterlizenzierung und Verbreitung ist nichtig und beendet automatisch Ihre Rechte unter dieser Lizenz. Jedoch werden die Lizenzen Dritter, die von Ihnen Kopien oder Rechte unter dieser Lizenz erhalten haben, nicht beendet, solange diese die Lizenz voll anerkennen und befolgen.
5. Sie sind nicht verpflichtet, diese Lizenz anzunehmen, da Sie sie nicht unterzeichnet haben. Jedoch gibt Ihnen nichts anderes die Erlaubnis, das Programm oder von ihm abgeleitete Werke zu verändern oder zu verbreiten. Diese Handlungen sind gesetzlich verboten, wenn Sie diese Lizenz nicht anerkennen. Indem Sie das Programm (oder ein darauf basierendes Werk) verändern oder verbreiten, erklären Sie Ihr Einverständnis mit dieser Lizenz

und mit allen ihren Bedingungen bezüglich der Vervielfältigung, Verbreitung und Veränderung des Programms oder eines darauf basierenden Werks.

6. Jedesmal, wenn Sie das Programm (oder ein auf dem Programm basierendes Werk) weitergeben, erhält der Empfänger automatisch vom ursprünglichen Lizenzgeber die Lizenz, das Programm entsprechend den hier festgelegten Bestimmungen zu vervielfältigen, zu verbreiten und zu verändern. Sie dürfen keine weiteren Einschränkungen der Durchsetzung der hierin zugestandenen Rechte des Empfängers vornehmen. Sie sind nicht dafür verantwortlich, die Einhaltung dieser Lizenz durch Dritte durchzusetzen.
7. Sollten Ihnen infolge eines Gerichtsurteils, des Vorwurfs einer Patentverletzung oder aus einem anderen Grunde (nicht auf Patentfragen begrenzt) Bedingungen (durch Gerichtsbeschluss, Vergleich oder anderweitig) auferlegt werden, die den Bedingungen dieser Lizenz widersprechen, so befreien Sie diese Umstände nicht von den Bestimmungen dieser Lizenz. Wenn es Ihnen nicht möglich ist, das Programm unter gleichzeitiger Beachtung der Bedingungen in dieser Lizenz und Ihrer anderweitigen Verpflichtungen zu verbreiten, dann dürfen Sie als Folge das Programm überhaupt nicht verbreiten. Wenn zum Beispiel ein Patent nicht die gebührenfreie Weiterverbreitung des Programms durch diejenigen erlaubt, die das Programm direkt oder indirekt von Ihnen erhalten haben, dann besteht der einzige Weg, sowohl das Patentrecht als auch diese Lizenz zu befolgen, darin, ganz auf die Verbreitung des Programms zu verzichten.

Sollte sich ein Teil dieses Paragraphen als ungültig oder unter bestimmten Umständen nicht durchsetzbar erweisen, so soll dieser Paragraph seinem Sinne nach angewandt werden; im übrigen soll dieser Paragraph als Ganzes gelten.

Zweck dieses Paragraphen ist nicht, Sie dazu zu bringen, irgendwelche Patente oder andere Eigentumsansprüche zu verletzen oder die Gültigkeit solcher Ansprüche zu bestreiten; dieser Paragraph hat einzig den Zweck, die Integrität des Vertriebssystems der freien Software zu schützen, das durch die Praxis öffentlicher Lizenzen verwirklicht wird. Viele Leute haben großzügige Beiträge zu dem großen Angebot der mit diesem System verbreiteten Software im Vertrauen auf die konsistente Anwendung dieses Systems geleistet; es liegt am Autor/Geber, zu entscheiden, ob er die Software mittels irgendeines anderen Systems verbreiten will; ein Lizenznehmer hat auf diese Entscheidung keinen Einfluss.

Dieser Paragraph ist dazu gedacht, deutlich klarzustellen, was als Konsequenz aus dem Rest dieser Lizenz betrachtet wird.

8. Wenn die Verbreitung und/oder die Benutzung des Programms in bestimmten Staaten entweder durch Patente oder durch urheberrechtlich geschützte Schnittstellen eingeschränkt ist, kann der Urheberrechtsinhaber, der das Programm unter diese Lizenz gestellt hat, eine explizite geographische Begrenzung der Verbreitung angeben, in der diese Staaten ausgeschlossen werden, so dass die Verbreitung nur innerhalb und zwischen den Staaten erlaubt ist, die nicht ausgeschlossen sind. In einem solchen Fall beinhaltet diese Lizenz die Beschränkung, als wäre sie in diesem Text niedergeschrieben.
9. Die Free Software Foundation kann von Zeit zu Zeit überarbeitete und/oder neue Versionen der General Public License veröffentlichen. Solche neuen Versionen werden vom Grundprinzip her der gegenwärtigen entsprechen, können aber im Detail abweichen, um neuen Problemen und Anforderungen gerecht zu werden.

Jede Version dieser Lizenz hat eine eindeutige Versionsnummer. Wenn in einem Programm angegeben wird, dass es dieser Lizenz in einer bestimmten Versionsnummer oder „jeder späteren Version“ („any later version“) unterliegt, so haben Sie die Wahl, entweder den Bestimmungen der genannten Version zu folgen oder denen jeder beliebigen späteren Version, die von der Free Software Foundation veröffentlicht wurde. Wenn das Programm keine Versionsnummer angibt, können Sie eine beliebige Version wählen, die je von der Free Software Foundation veröffentlicht wurde.

10. Wenn Sie den Wunsch haben, Teile des Programms in anderen freien Programmen zu verwenden, deren Bedingungen für die Verbreitung anders sind, schreiben Sie an den Autor, um ihn um die Erlaubnis zu bitten. Für Software, die unter dem Copyright der Free Software Foundation steht, schreiben Sie an die Free Software Foundation; wir machen zu diesem Zweck gelegentlich Ausnahmen. Unsere Entscheidung wird von den beiden Zielen geleitet werden, zum einen den freien Status aller von unserer freien Software abgeleiteten Werke zu erhalten und zum anderen das gemeinschaftliche Nutzen und Wiederverwenden von Software im allgemeinen zu fördern.

### Keine Gewährleistung

11. Da das Programm ohne jegliche Kosten lizenziert wird, besteht keinerlei Gewährleistung für das Programm, soweit dies gesetzlich zulässig ist. Sofern nicht anderweitig schriftlich bestätigt, stellen die Copyright-Inhaber

und/oder Dritte das Programm so zur Verfügung, 'wie es ist', ohne irgendeine Gewährleistung, weder ausdrücklich noch implizit, einschließlich - aber nicht begrenzt auf - Marktreife oder Verwendbarkeit für einen bestimmten Zweck. Das volle Risiko bezüglich Qualität und Leistungsfähigkeit des Programms liegt bei Ihnen. Sollte sich das Programm als fehlerhaft herausstellen, liegen die Kosten für notwendigen Service, Reparatur oder Korrektur bei Ihnen.

12. In keinem Fall, außer wenn durch geltendes Recht gefordert oder schriftlich zugesichert, ist irgendein Copyright-Inhaber oder irgendein Dritter, der das Programm wie oben erlaubt modifiziert oder verbreitet hat, Ihnen gegenüber für irgendwelche Schäden haftbar, einschließlich jeglicher allgemeiner oder spezieller Schäden, Schäden durch Seiteneffekte (Nebenwirkungen) oder Folgeschäden, die aus der Benutzung des Programms oder der Unbenutzbarkeit des Programms folgen (einschließlich - aber nicht beschränkt auf - Datenverluste, fehlerhafte Verarbeitung von Daten, Verluste, die von Ihnen oder anderen getragen werden müssen, oder dem Unvermögen des Programms, mit irgendeinem anderen Programm zusammenzuarbeiten), selbst wenn ein Copyright-Inhaber oder Dritter über die Möglichkeit solcher Schäden unterrichtet worden war.

### QGIS Qt Ausnahme für die GPL

Zusätzlich, als eine besondere Ausnahme, gibt das QGIS Development Team das Recht, den Quellcode dieses Programms mit der Qt-Bibliothek, einschliesslich aber nicht begrenzt auf die folgenden Versionen (frei und kommerziell): Qt/Non-commercial Windows, Qt/Windows, Qt/X11, Qt/Mac, und Qt/Embedded (oder mit modifizierten Qt-Versionen, welche die gleiche Lizenz wie Qt nutzen) zu verlinken und zu vertreiben. Für jeglichen Quellcode ausser dem Qt-Quellcode, müssen Sie die GNU General Public License in jeder Hinsicht befolgen. Wenn Sie diesen Text ändern, können Sie diese Ausnahmen erweitern zu Ihrer Version dieses Textes, aber Sie sind dazu nicht verpflichtet. Wenn Sie es nicht möchten, löschen Sie diese Ausnahme aus Ihrer Version.

## 32.2 Anhang B: GNU Free Documentation License

Dies ist eine inoffizielle Übersetzung der GNU Free Documentation License (FDLv1.3) ins Deutsche. Sie wurde nicht von der Free Software Foundation veröffentlicht, und legt keine rechtsgültigen Bestimmungen zur Verteilung für Dokumentation fest, welche die GNU FDL verwendet - nur der englische Originaltext der GNU FDL gewährleistet dies. Dennoch hoffen wir, dass diese Übersetzung Deutsch sprechenden dazu verhilft, die GNU FDL besser zu verstehen

Copyright 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc

<https://www.fsf.org/>

Es ist jedermann gestattet, diese Lizenzurkunde zu vervielfältigen und unveränderte Kopien zu verbreiten; Änderungen sind jedoch nicht erlaubt.

### Einleitung

Der Zweck dieser Lizenz ist es, ein Handbuch, Fachbuch, oder ein anderes sachliches sowie nützliches Dokument ‚frei‘ im Sinne von Freiheit anzufertigen: um jedermann die wirkliche Freiheit zuzusichern, es zu kopieren und neu zu verteilen, mit oder ohne Modifikation daran, entweder kommerziell oder nicht kommerziell. Zweitrangig erhält diese Lizenz für den Autor und Herausgeber die Möglichkeit aufrecht, Anerkennung für ihr Werk zu bekommen, während Sie nicht als verantwortlich betrachtet werden fürvorgenommene Modifikationen anderer.

Diese Lizenz ist eine Art ‚Copyleft‘, was bedeutet, dass abgeleitete Werke des Dokuments ihrerseits im selben Sinne frei sein müssen. Sie ergänzt die GNU General Public License, welche eine Copyleft-Lizenz darstellt, die für freie Software vorgesehen ist.

Wir haben diese Lizenz entworfen, um sie für Handbücher für freie Software zu verwenden, denn freie Software braucht kostenlose Dokumentation: Ein kostenloses Programm sollte mit Handbüchern geliefert werden, die die gleichen Freiheiten bieten wie die Software. Diese Lizenz ist jedoch nicht auf Softwarehandbücher beschränkt. Es kann für jedes Textwerk verwendet werden, unabhängig vom Thema oder davon, ob es als gedrucktes Buch veröffentlicht wird. Wir empfehlen diese Lizenz vor allem für Werke, deren Zweck der Anleitung oder Referenz dient.

### 1. Anwendbarkeit und Definitionen

Diese Lizenz trifft auf jedes Handbuch oder sonstiges Werk zu, in beliebiger Form, das einen vom Urheberrechtseinhaber untergebrachten Hinweis mit den Worten enthält, dass es unter den Bestimmungen dieser Lizenz verteilt werden kann. Solch ein Hinweis gewährt eine weltweite, vergütungsfreie Lizenz von unbefristeter Dauer, um dieses Werk unter den hier festgelegten Bedingungen zu verwenden. Das ‚Dokument‘, nachstehend, bezieht sich auf jedes derartige Handbuch oder Werk. Jedes Mitglied der Öffentlichkeit ist ein Lizenznehmer und wird als ‚Sie‘ angesprochen. Sie akzeptieren die Lizenz, wenn Sie die Werke kopieren, modifizieren oder verteilen, was gewissermaßen unter dem Urheberrechtsgesetz die Erlaubnis erfordert.

Eine **modifizierte Version** des Dokumentes bedeutet, dass jedes Werk das Dokument selbst oder einen Teil davon beinhaltet, entweder unverändert kopiert, oder mit Modifikationen und/oder übersetzt in eine andere Sprache.

Ein **untergeordneter Abschnitt** ist ein benannter Anhang oder ein Teilei-Abschnitt des Dokuments, der sich ausschließlich mit dem Verhältnis der Herausgeber oder Autoren des Dokuments zum Gesamtthema des Dokuments befasst (oder damit in Verbindung stehende Bewandnisse), und nichts beinhaltet was direkt innerhalb des Gesamtthemas fallen könnte. (Wenn das Dokument zu einem Fachbuch über Mathematik gehört, kann ein untergeordneter Abschnitt folglich nichts Mathematisches erläutern.) Die Beziehung könnte ein Anliegen mit historischer Verbindung zum Thema oder ähnlicher Angelegenheiten, oder bezüglich ihrer rechtlichen, kommerziellen, philosophischen, ethischen oder politischen Position sein.

Die ‚unveränderlichen Abschnitte‘ sind bestimmte untergeordnete Abschnitte, deren Titel zum Beispiel in dem Hinweis, der besagt, dass das Dokument unter dieser Lizenz freigegeben ist, als jene unveränderlichen Abschnitte gekennzeichnet sind. Wenn ein Abschnitt nicht zur obigen Definition von untergeordnet passt, dann ist es nicht erlaubt ihn als unveränderlich zu kennzeichnen. Das Dokument kann null unveränderliche Abschnitte enthalten. Wenn das Dokument keine unveränderlichen Abschnitte kennzeichnet, dann gibt es keine.

Die ‚Umschlagtexte‘ sind bestimmte kurze Textpassagen die als vordere Umschlagtexte oder hintere Umschlagtexte in dem Hinweis, der besagt, dass das Dokument unter dieser Lizenz freigegeben ist, verzeichnet sind. Ein vorderer Umschlagtext darf höchstens 5 Wörter lang sein, und ein hinterer Umschlagtext darf höchstens 25 Wörter lang sein.

Eine ‚transparente‘ Kopie des Dokumentes bedeutet eine maschinenlesbare Kopie, in einem ansehnlichen Format, dessen Spezifikation für die Allgemeinheit verfügbar ist, welches geeignet ist das Dokument unkompliziert mit allgemeinen Texteditoren oder (für aus Pixeln bestehende Bilder) allgemeinen Malprogrammen oder (für Zeichnungen) irgendeinem weit verbreiteten Zeicheneditor zu überarbeiten, und das geeignet ist zur Eingabe in Textformatierer oder zur automatischen Übersetzung in eine Variante von geeigneten Formaten zur Eingabe in Textformatierer. Eine Kopie, erstellt in einem ansonsten transparenten Dateiformat, dessen Auszeichnung oder fehlende Auszeichnung derart ausgestaltet wurde, um nachträgliche Modifikation durch Leser zu behindern oder zu verhindern, ist nicht transparent. Ein Bildformat ist nicht transparent, wenn es für irgendeine beträchtliche Menge von Text verwendet wird. Eine Kopie, die nicht ‚transparent‘ ist, wird ‚undurchlässig‘ genannt.

Beispiele von geeigneten Formaten für transparente Kopien beinhalten einfachen ASCII ohne Auszeichnung, Texinfo Eingabeformat, LaTeX Eingabeformat, SGML oder XML unter Verwendung einer öffentlich zugänglichen DTD, und standardkonformes einfaches HTML, PostScript oder PDF, vorgesehen für humane Modifikation. Beispiele für transparente Bildformate beinhalten PNG, XCF und JPG. Undurchlässige Formate beinhalten proprietäre Formate die nur mit proprietären Textverarbeitungssystemen gelesen und bearbeitet werden können, SGML oder XML für welche die DTD und/oder Bearbeitungswerkzeuge nicht allgemein verfügbar sind, und das maschinengenerierte HTML, PostScript oder PDF, erzeugt mit irgendwelchen Textverarbeitungssystemen, nur für Ausgabezwecke.

Das ‚Titelblatt‘ bedeutet, für ein gedrucktes Buch, das Titelblatt an sich zzgl. solcher nachfolgenden Seiten die notwendig sind, die Lesbarkeit des Materials beizubehalten, wie von dieser Lizenz erfordert, um im Titelblatt zu erscheinen. Für Werke in Formaten, welche kein Titelblatt als solches haben, bedeutet ‚Titelblatt‘ der Text nahe dem bedeutendsten Auftreten des Titels dieses Werkes, ausgehend vom Anfang des Textkörpers.

Der ‚Herausgeber‘ ist jede Person oder Instanz, welche Kopien des Dokuments an die Öffentlichkeit verteilt.

Ein Abschnitt **Mit dem Titel XYZ** bedeutet eine benannte Untereinheit des Dokuments, dessen Titel entweder genau XYZ ist, oder XYZ in runden Klammern, gefolgt von Text, welcher XYZ in eine andere Sprache übersetzt. (Hier steht XYZ für einen spezifischen Abschnittsnamen, weiter unten erwähnt, wie zum Beispiel ‚Danksagungen‘, ‚Widmungen‘, ‚Befürwortungen‘ oder ‚Verlauf‘.) Den ‚Titel‘ eines solchen Abschnitts ‚zu erhalten‘, wenn Sie das Dokument modifizieren, bedeutet, dass dieser ein Abschnitt ‚Mit dem Titel XYZ‘ gemäß seiner Definition bleibt.

Das Dokument kann neben dem Hinweis, der festlegt, dass diese Lizenz auf das Dokument zutrifft, Garantie-Ausschlussklauseln beinhalten. Diese Garantie-Ausschlussklauseln werden als aufgenommen betrachtet, durch

Verweis in dieser Lizenz, aber nur was die Ausschließung von Garantien betrifft: Jede andere Auswirkung, die diese Garantie-Ausschlussklauseln haben können, ist ungültig und hat keine Wirkung auf die Bedeutung dieser Lizenz.

## 2. Unveränderte Vervielfältigung

Sie dürfen das Dokument in beliebiger Form kopieren und verteilen, entweder kommerziell oder nicht kommerziell, vorausgesetzt, dass diese Lizenz, die Urheberrechtshinweise und der Lizenzhinweis mit den Worten, dass diese Lizenz auf das Dokument zutrifft, in allen Kopien wiedergegeben werden, und dass Sie keine anderen Bedingungen, welcher Art auch immer, zu jenen dieser Lizenz hinzufügen. Sie dürfen keine technischen Maßnahmen anwenden, die das Lesen oder weitere Vervielfältigung von den Kopien, die Sie erstellen oder verteilen, behindern oder kontrollieren. Allerdings können Sie Bezahlung im Austausch für Kopien entgegennehmen. Wenn Sie eine ausreichend große Anzahl von Kopien verteilen, müssen Sie außerdem die Bedingungen in Abschnitt 3 beachten.

Sie dürfen außerdem Kopien verleihen, unter den selben, oben angegebenen Bedingungen, und Sie dürfen Kopien öffentlich auslegen.

## 3. Vervielfältigung grosser Mengen

Wenn Sie, mehr als 100 umfassend, gedruckte Kopien eines Dokuments veröffentlichen (oder Kopien in Medien, die üblicherweise gedruckte Umschläge haben) und der Lizenzhinweis des Dokuments Umschlagtexte erfordert, müssen Sie die Kopien in Umschläge beifügen, welche eindeutig und leserlich alle diese Umschlagtexte tragen: vordere Umschlagtexte auf vordere Umschläge, und hintere Umschlagtexte auf hintere Umschläge. Beide Umschläge müssen, ebenso eindeutig und leserlich, Sie, als den Herausgeber dieser Kopien, identifizieren. Der vordere Umschlag muss den vollständigen Titel, mit allen Wörtern des Titels, in gleicher Weise auffallend und leicht erkennbar darstellen. Sie dürfen darüber hinaus sonstiges Material auf die Umschläge anbringen. Vervielfältigung mit Änderungen, begrenzt auf die Umschläge, sofern sie den Titel des Dokuments erhalten und diese Bedingungen erfüllen, können ansonsten als unveränderte Vervielfältigung behandelt werden.

Wenn die erforderlichen Texte für beide Umschläge zu umfangreich sind, um lesbar zu passen, sollten Sie die ersten verzeichneten (so viele, um angemessen zu passen) auf den aktuellen Umschlag setzen, und den Rest auf den nachfolgenden Seiten fortsetzen.

Wenn Sie, mehr als 100 umfassend, undurchlässige Kopien des Dokuments veröffentlichen oder verteilen, müssen Sie entweder eine maschinenlesbare transparente Kopie zusammen mit jeder undurchlässigen Kopie einbeziehen, oder in bzw. mit jeder undurchlässigen Kopie eine Computer-Netzwerkadresse angeben, von welcher die allgemeine netzwerknutzende Öffentlichkeit, unter Verwendung von Netzwerkprotokollen öffentlicher Standards, Zugang hat, um eine vollständig transparente Kopie, frei von hinzugefügtem Material, herunterzuladen. Falls Sie die letzte Option verwenden, müssen Sie angemessen überlegte Schritte unternehmen, wenn Sie mit der Verteilung von undurchlässigen Kopien in großen Mengen beginnen, um sicherzustellen, dass diese transparente Kopie unter der angegebenen Adresse auf diese Weise zugänglich bleibt, bis mindestens ein Jahr nachdem Sie zum letzten Mal eine undurchlässige Kopie (unmittelbar oder durch Ihre Vertreter oder Einzelhändler) in dieser Ausgabe an die Öffentlichkeit verteilen.

Es ist erwünscht, aber nicht erforderlich, dass Sie in Kontakt mit den Autoren des Dokuments treten, kurz bevor Sie irgendeine große Anzahl von Kopien neu verteilen, um ihnen die Möglichkeit zu geben, Sie mit einer aktualisierten Version des Dokuments zu versorgen.

## 4. Modifikationen

Sie dürfen eine modifizierte Version des Dokuments unter den oben erwähnten Bedingungen der Abschnitte 2 und 3 kopieren und verteilen, vorausgesetzt, dass Sie die modifizierte Version unter genau dieser Lizenz freigeben, mit der modifizierten Version wird die Rolle des Dokuments besetzt, und somit der Lizenzierung von Verteilung und Modifikation der modifizierten Version, für jeden der eine Kopie davon besitzt. Zusätzlich müssen Sie diese Sachen in der modifizierten Version erledigen:

- A. Verwenden Sie auf dem Titelblatt (und gegebenenfalls auf den Umschlägen) einen Titel der sich von dem des Dokuments unterscheidet, und von jenen der vorhergehenden Versionen (die, wenn es irgendwelche gab, in dem Verlaufs-Abschnitt des Dokuments verzeichnet sein sollten). Sie dürfen denselben Titel wie den einer vorhergehenden Version verwenden, wenn der ursprüngliche Herausgeber dieser Version die Erlaubnis gibt.
- B. Verzeichnen Sie auf dem Titelblatt, als Autoren, eine oder mehrere Personen oder Organe, verantwortlich für die Autorschaft der Modifikationen in der modifizierten Version, zusammen mit mindestens fünf der Hauptautoren des Dokuments (alle seine Hauptautoren, wenn es weniger als fünf hat), es sei denn, dass sie Sie von dieser Anforderung befreien.

- C. Geben Sie auf dem Titelblatt den Namen des Herausgebers der modifizierten Version als den des Herausgebers an.
- D. Erhalten Sie alle Urheberrechtshinweise des Dokumentes.
- E. Fügen Sie einen entsprechenden Urheberrechtshinweis für Ihre Modifikationen, angrenzend zu den anderen Urheberrechtshinweisen, hinzu.
- F. Nehmen Sie, direkt nach den Urheberrechtshinweisen, einen Lizenzhinweis auf, der die öffentliche Erlaubnis gibt, die modifizierte Version unter den Bestimmungen dieser Lizenz zu verwenden, in der Form, wie weiter unten im Anhang gezeigt.
- G. Erhalten Sie in diesem Lizenzhinweis die vollständigen Listen der unveränderlichen Abschnitte und erforderlichen Umschlagtexte, aufgeführt in dem Lizenzhinweis des Dokuments.
- H. Nehmen Sie eine ungeänderte Kopie dieser Lizenz auf.
- I. Erhalten Sie den Abschnitt mit dem Titel ‚Verlauf‘, erhalten Sie seinen Titel und fügen Sie ihm einen Punkt hinzu, der mindestens den Titel, das Jahr, neue Autoren und Herausgeber der modifizierten Version angibt, wie auf dem Titelblatt aufgeführt. Wenn es keinen Abschnitt ‚Verlauf‘ in dem Dokument gibt, erzeugen Sie einen, der den Titel, das Jahr, die Autoren und Herausgeber des Dokuments angibt, wie auf seinem Titelblatt aufgeführt, dann fügen Sie einen Punkt hinzu, der die modifizierte Version beschreibt, wie in dem vorhergehenden Satz angegeben.
- J. Erhalten Sie gegebenenfalls die Netzwerkadresse, aufgeführt in dem Dokument, für den öffentlichen Zugang zu einer transparenten Kopie des Dokuments, und ebenso die in dem Dokument aufgeführten Netzwerkadressen, für vorhergehende Versionen, auf dem es beruht. Diese können in den Abschnitt ‚Verlauf‘ untergebracht werden. Sie dürfen eine Netzwerkadresse für ein Werk auslassen, das mindestens vier Jahre vor dem Dokument selbst veröffentlicht wurde, oder der ursprüngliche Herausgeber der Version, auf die es sich bezieht, die Erlaubnis gibt.
- K. Für jeden Abschnitt mit dem Titel ‚Danksagungen‘ oder ‚Widmungen‘, erhalten Sie den Titel des Abschnitts, und erhalten Sie in dem Abschnitt vollständig den wesentlichen Gehalt und Umgangston der, von jeglichen Mitwirkenden darin aufgeführten, Danksagungen und/oder Widmungen.
- L. Erhalten Sie alle unveränderlichen Abschnitte des Dokuments, ungeändert in ihrem Text und ihren Titeln. Abschnittsnummern oder Entsprechendes werden nicht als Teil der Abschnittstitel betrachtet.
- M. Löschen Sie jeden Abschnitt mit dem Titel ‚Befürwortungen‘. Solch ein Abschnitt darf nicht in die modifizierte Version aufgenommen werden.
- N. Betiteln Sie keinen Titel eines vorhandenen Abschnitts neu, der mit ‚Befürwortung‘ betitelt ist oder in Konflikt zum Titel irgendeines unveränderlichen Abschnitts steht.
- O. Erhalten Sie jegliche Garantie-Ausschlussklauseln.

Wenn die modifizierte Version neue Teilei-Abschnitte oder Anhänge beinhaltet, die als untergeordnete Abschnitte qualifiziert sind und kein kopiertes Material aus dem Dokument enthalten, können Sie Ihrer Wahl nach einige oder alle diese Abschnitte als unveränderlich bestimmen. Um dies zu tun, fügen Sie ihre Titel zur Liste der unveränderlichen Abschnitte in den Lizenzhinweis der modifizierten Version hinzu. Diese Titel müssen sich von allen anderen Abschnittstitel unterscheiden.

Sie können einen Abschnitt mit dem Titel ‚Befürwortungen‘ hinzufügen, vorausgesetzt, dass dieser nichts enthält als nur Befürwortungen Ihrer modifizierten Version durch verschiedene Parteien Beispiel Aussagen der Begutachtung oder dass der Text von einer Organisation als maßgebliche Definition eines Standards anerkannt worden ist.

Sie können eine Textpassage von bis zu fünf Wörtern als einen vorderen Umschlagtext, und eine Textpassage von bis zu 25 Wörtern als hinteren Umschlagtext in der modifizierten Version hinzufügen. Nur eine Textpassage des vorderen Umschlagtextes und eine des hinteren Umschlagtextes kann von (oder durch, von ihr, angefertigte Zusammenstellung) irgendeiner Person hinzugefügt werden. Wenn das Dokument bereits einen Umschlagtext für denselben Umschlag beinhaltet, zuvor von Ihnen hinzugefügt oder durch Zusammenstellung, angefertigt von derselben Person, in dessen Namen Sie handeln, können Sie keinen weiteren hinzufügen; aber Sie dürfen den alten ersetzen, mit ausdrücklicher Erlaubnis des vorherigen Herausgebers, welcher den alten hinzufügte.

Der/die Autor(en) und Herausgeber des Dokuments geben durch diese Lizenz für keine modifizierte Version die Erlaubnis ihre Namen für Werbung zu verwenden oder Befürwortung zu behaupten oder anzudeuten.

## 5. Dokumente kombinieren

Sie dürfen das Dokument mit anderen, unter dieser Lizenz freigegeben, Dokumenten kombinieren, unter den Bestimmungen, definiert in Abschnitt 4 für modifizierte Versionen, vorausgesetzt, dass Sie in die Kombination alle unveränderlichen Abschnitte aller Originaldokumente, nicht modifiziert, einbeziehen und sie alle als unveränderliche Abschnitte Ihres kombinierten Werkes in dessen Lizenzhinweis verzeichnen, und dass Sie alle ihre Garantie-Ausschlussklauseln erhalten.

Das kombinierte Werk muss nur eine Kopie dieser Lizenz enthalten, und mehrfach identische unveränderliche Abschnitte können gegen eine einzelne Kopie ausgetauscht werden. Wenn es mehrfach identische unveränderliche Abschnitte mit demselben Namen, aber unterschiedlichem Inhalt gibt, erzeugen Sie den Titel eines jeden solchen Abschnitts an seinem Ende eindeutig durch Hinzufügen, in runden Klammern, des Namens, falls bekannt, des ursprünglichen Autors oder Herausgebers dieses Abschnittes, oder andernfalls einer eindeutigen Nummer. Nehmen Sie dieselbe Anpassung bei den Abschnittstiteln in der Liste von unveränderlichen Abschnitten in dem Lizenzhinweis des kombinierten Werkes vor.

In der Kombination müssen Sie jegliche Abschnitte mit dem Titel ‚Verlauf‘ in den verschiedenen Originaldokumenten kombinieren und einen Abschnitt mit dem Titel ‚Verlauf‘ bilden; ebenso kombinieren Sie jegliche Abschnitte mit dem Titel ‚Danksagungen‘, und jegliche Abschnitte mit dem Titel ‚Widmungen‘. Sie müssen alle Abschnitte mit dem Titel ‚Befürwortungen‘ löschen.

## 6. Sammlungen von Dokumenten

Sie dürfen eine Sammlung erzeugen, bestehend aus dem Dokument und anderen Dokumenten, freigegeben unter dieser Lizenz, und die Einzel-Kopien dieser Lizenz in den verschiedenen Dokumenten gegen eine einzelne Kopie, die in die Sammlung aufgenommen wird, austauschen, vorausgesetzt, dass Sie die Regeln dieser Lizenz für unveränderte Vervielfältigung jedes Dokuments in jeder anderen Hinsicht befolgen.

Sie dürfen ein einzelnes Dokument aus solch einer Sammlung entnehmen, und es individuell unter dieser Lizenz verteilen, vorausgesetzt, dass Sie eine Kopie dieser Lizenz in das extrahierte Dokument einfügen, und diese Lizenz in jeder anderen Hinsicht, bezüglich der unveränderten Vervielfältigung dieses Dokuments, befolgen.

## 7. Gruppierung mit unabhängigen Werken

Eine Zusammentragung des Dokuments oder seiner Ableitungen mit anderen gesonderten und unabhängigen Dokumenten oder Werken, in oder zu einem Band auf einem Speicher- oder Verteilungsmedium, wird eine ‚Gruppierung‘ genannt, wenn das Urheberrecht, das sich aus der Zusammentragung ergibt, nicht verwendet wird, um die gültigen Rechte der Benutzer dieser Zusammentragungen darüber hinaus, was die Einzel-Werke gestatten, zu beschränken. Wenn das Dokument in eine Gruppierung aufgenommen ist, trifft diese Lizenz nicht auf die anderen Werke in der Gruppierung zu, welche nicht selbst abgeleitete Werke des Dokuments sind.

Wenn die Anforderung für Umschlagtext von Abschnitt 3 auf diese Kopien des Dokuments anwendbar ist, wenn außerdem das Dokument weniger als eine Hälfte der gesamten Gruppierung darstellt, kann der Umschlagtext des Dokuments auf Umschläge, die das Dokument innerhalb der Gruppierung einklammern, untergebracht werden, oder der elektronischen Entsprechung von Umschlägen, wenn das Dokument in elektronischer Form vorliegt. Andernfalls müssen sie auf gedruckten Umschlägen vorkommen, welche die vollständige Gruppierung einklammern.

## 8. Übersetzung

Übersetzung wird als eine Art von Modifikation betrachtet, also dürfen Sie Übersetzungen unter den Bestimmungen von Abschnitt 4 verteilen. Das Austauschen unveränderlicher Abschnitte mit Übersetzungen erfordert besondere Erlaubnis von ihren Urheberrechtshabern, aber Sie können Übersetzungen von einigen oder allen unveränderlichen Abschnitten aufnehmen, zusätzlich zu den Originalversionen dieser unveränderlichen Abschnitte. Sie können eine Übersetzung dieser Lizenz aufnehmen, und alle Lizenzhinweise in dem Dokument, und jegliche Garantie-Ausschlussklauseln, vorausgesetzt, dass Sie außerdem die englische Originalversion dieser Lizenz und die Originalversionen jener Hinweise und Ausschlussklauseln aufnehmen. Im Falle eines Widerspruchs zwischen der Übersetzung und der Originalversion dieser Lizenz oder eines Hinweises oder einer Ausschlussklausel, wird sich die Originalversion durchsetzen.

Wenn ein Abschnitt in dem Dokument mit ‚Danksagungen‘, ‚Widmungen‘ oder ‚Verlauf‘ betitelt ist, wird die Anforderung (Abschnitt 4), seinen Titel (Abschnitt 1) zu erhalten, normalerweise die Änderung des tatsächlichen Titels erfordern.

## 9. Schlussbestimmung

Sie dürfen das Dokument nicht kopieren, modifizieren, unterlizenzieren oder verteilen, außer, als ausdrücklich unter dieser Lizenz zur Verfügung gestellt. Jeder andere Versuch es zu kopieren, zu modifizieren, zu unterlizenzieren oder zu verteilen ist ungültig, und wird automatisch Ihre Rechte unter dieser Lizenz aufkündigen.

Jedoch, wenn Sie jeglichen Verstoß gegen diese Lizenz beenden, dann ist ihre Lizenz eines bestimmten Urheberrechtsinhabers wieder (a) vorläufig, sofern und solange der Urheberrechtsinhaber nicht ihre Lizenz beendet, und (b) dauerhaft, wenn der Urheberrechtsinhaber es versäumt, Sie über den Verstoß zu benachrichtigen, mit angemessenen Mitteln, vorzeitig 60 Tage nach der Beendigung.

Darüber hinaus ist Ihre Lizenz eines bestimmten Urheberrechtsinhabers wieder dauerhaft, wenn der Urheberrechtsinhaber Sie über den Verstoß mit angemessenen Mitteln benachrichtigt, es das erste Mal ist, dass Sie eine Benachrichtigung über den Verstoß dieser Lizenz (für jedes Werk) vom Urheberrechtsinhaber erhalten, und Sie den Verstoß vorzeitig 30 Tage nach Erhalt der Benachrichtigung beseitigen.

Die Beendigung Ihrer Rechte unter diesem Abschnitt beendet nicht die Lizenzen der Parteien, welche Kopien oder Rechte von Ihnen unter dieser Lizenz erhalten haben. Wenn Ihre Rechte aufgekündigt und nicht wieder dauerhaft geworden sind, gibt Ihnen der Erhalt einer Kopie von Teilen oder Sämtlichem des gleichen Materials keine Rechte, diese zu verwenden.

### 10. Zukünftige Überarbeitungen dieser Lizenz

Die Free Software Foundation kann von Zeit zu Zeit neue, überarbeitete Versionen der GNU Free Documentation License veröffentlichen. Solche neuen Versionen ähneln im Geiste der aktuellen Version, können jedoch im Detail abweichen, um neuen Problemen oder Bedenken Rechnung zu tragen. Siehe <http://www.gnu.org/copyleft/>.

Jeder Version der Lizenz wird eine charakteristische Versionsnummer gegeben. Wenn das Dokument spezifiziert, dass eine besonders nummerierte Version dieser Lizenz, oder jede spätere Version auf sich zutrifft, haben Sie die Wahl den Bestimmungen und Bedingungen entweder dieser spezifizierten Version oder jeder späteren Version, die (nicht als Entwurf) durch die Free Software Foundation veröffentlicht worden ist, zu folgen. Wenn das Dokument spezifiziert, dass ein Bevollmächtigter entscheiden darf welche zukünftigen Versionen dieser Lizenz verwendet werden können, berechtigt Sie das dauerhaft, durch die öffentliche Aussage desjenigen Bevollmächtigten über die Akzeptanz einer Version, diese Version für das Dokument zu wählen.

### 11. Relizenzierung

„Massive Multiauthor Collaboration Site“ (or „MMC Site“) meint jeglichen World Wide Web Server, welcher urheberrechtsfähige Werke veröffentlicht und ferner bedeutende Möglichkeiten für jeden bietet, solche Werke zu editieren. Ein öffentliches Wiki, welches jeder bearbeiten kann, ist ein Beispiel eines solchen Servers. Eine „Massives Multiautor Collaboration“ (oder „MMC“), enthalten im Standort, bedeutet jegliche Zusammenstellung urheberrechtsfähiger Werke, die folglich auf dem MMC-Standort veröffentlicht werden.

„CC-BY-SA“ meint die Creative Commons Namensnennung-Weitergabe unter gleichen Bedingungen 3.0 Lizenz, veröffentlicht von der Creative Commons Corporation, einem gemeinnützigen Unternehmen mit Hauptgeschäftssitz in San Francisco, Californien, sowie zukünftige Copyleft-Versionen dieser Lizenz, welche durch dieselbe Organisation veröffentlicht wird.

„Einbeziehen“ bedeutet, ein Dokument zu veröffentlichen oder neu zu veröffentlichen, ganz oder teilweise, als Bestandteil eines anderen Dokuments.

Eine MMC ist „geeignet zur Relizenzierung“, wenn sie unter dieser Lizenz lizenziert wird, und wenn alle Werke, die zuerst unter dieser Lizenz irgendwo anders als auf der MMC veröffentlicht, und nachträglich ganz oder teilweise in der MMC einbezogen wurden, (1) keine Umschlagtexte oder unveränderliche Abschnitte hatten, und (2) folglich vor dem 1. November 2008 einbezogen wurden.

Betreiber eines MMC-Standortes kann eine MMC, enthalten im Standort unter CC-BY-SA des gleichen Standortes, jederzeit vor dem 1. August 2009 neu veröffentlichen, vorausgesetzt, die MMC ist für die Relizenzierung geeignet.

### Anhang: Wie Sie diese Lizenz für Ihre Dokumente verwenden

Um diese Lizenz in einem Dokument, das Sie geschrieben haben, zu verwenden, nehmen Sie eine Kopie der Lizenz in das Dokument auf und setzen Sie die folgenden Urheberrechts- und Lizenzhinweise unmittelbar hinter das Titelblatt:

Copyright © YEAR YOUR NAME.  
Permission is granted to copy, distribute and/or modify this document  
under the terms of the GNU Free Documentation License, Version 1.3

(Fortsetzung auf der nächsten Seite)



(Fortsetzung der vorherigen Seite)

```
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

Wenn Sie unveränderliche Abschnitte, vordere Umschlagtexte und hintere Umschlagtexte haben, ersetzen Sie die Zeile ‚ohne ...Umschlagtexte.‘ durch diese:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

Wenn Sie unveränderliche Abschnitte ohne Umschlagtexte haben oder eine andere Kombination der drei, vermischen Sie jene zwei Alternativen zu einer übereinstimmenden Situation.

Wenn Ihr Dokument nicht-triviale Beispiele eines Programmcodes enthält, empfehlen wir, diese Beispiele entsprechend unter einer freien Software-Lizenz ihrer Wahl, wie der GNU General Public License freizugeben, um deren Verwendung in freier Software zu gestatten.

## 32.3 Anhang C: QGIS-Dateiformate

### 32.3.1 QGS/QGZ - Das QGIS-Projekt-Dateiformat

Das **QGS**-Format ist ein XML-Format zur Speicherung von QGIS-Projekten. Das **QGZ**-Format ist ein komprimiertes (Zip-)Archiv, das eine QGS-Datei und eine QGD-Datei enthält. Die **QGD**-Datei ist die zugehörige Sqlite-Datenbank des qgis-Projekts, die Hilfsdaten für das Projekt enthält. Wenn es keine Hilfsdaten gibt, ist die QGD-Datei leer.

Eine QGIS-Datei enthält alles, was zum Speichern eines QGIS-Projekts erforderlich ist, einschließlich:

- Projekttitel
- Projekt-KBS
- der Layer-Baum
- Einrastoptionen
- Beziehungen
- Kartenausschnitt
- Projekt-Modelle
- Legende
- mapview docks (2D and 3D)
- the layers with links to the underlying datasets (data sources) and other layer properties including extent, SRS, joins, styles, renderer, blend mode, opacity and more.
- Projekteigenschaften

The figures below show the top level tags in a QGS file and the expanded `ProjectLayers` tag.

```
-<qgis version="3.4.13-Madeira" projectname="">
  <homePath path=""/>
  <title/>
  <autotransaction active="0"/>
  <evaluateDefaultValues active="0"/>
  <trust active="0"/>
+<projectCrs></projectCrs>
+<layer-tree-group></layer-tree-group>
+<snapping-settings tolerance="12" unit="1" enabled="0" type="1" mode="2" intersection-snapping="0">
  </snapping-settings>
  <relations/>
-<mapcanvas name="theMapCanvas" annotationsVisible="1">
  <units>meters</units>
  +<extent></extent>
  <rotation>0</rotation>
  +<destinationrs></destinationrs>
  <rendermaptile>0</rendermaptile>
</mapcanvas>
<projectModels/>
+<legend updateDrawingOrder="true"></legend>
  <mapViewDocks/>
  <mapViewDocks3D/>
+<projectlayers></projectlayers>
+<layerorder></layerorder>
+<properties></properties>
  <visibility-presets/>
  <transformContext/>
+<projectMetadata></projectMetadata>
  <Annotations/>
  <Layouts/>
</qgis>
```

Abb. 32.1: The top level tags in a QGS file

```

-<projectlayers>
-<maplayer styleCategories="AllStyleCategories" readOnly="0" autoRefreshTime="0" autoRefreshEnabled="0" refreshOnNotifyEnabled="0" maxScale="0"
geometry="Polygon" labelsEnabled="0" type="vector" simplifyDrawingHints="1" hasScaleBasedVisibilityFlag="0" simplifyDrawingTol="1"
simplifyMaxScale="1" minScale="1e+8" simplifyAlgorithm="0" simplifyLocal="1" refreshOnNotifyMessage="">
+<extent></extent>
+<id>watersheds_b62efa19_8809_4406_b6ec_2951ac4c94c5</id>
-<datasource>
./QGIS-Training-Data-2.0/exercise_data/processing/generalize/watersheds.shp
</datasource>
+<keywordList></keywordList>
+<layername>watersheds</layername>
+<srs></srs>
+<resourceMetadata></resourceMetadata>
+<provider encoding="UTF-8">ogr</provider>
+<vectorJoins>
+<layerDependencies>
+<dataDependencies>
+<legend type="default-vector"/>
+<expressionFields>
+<map-layer-style-manager current="default"></map-layer-style-manager>
+<auxiliaryLayer>
+<flags></flags>
+<renderer-v2 symbollevels="0" enableorderby="0" type="singleSymbol" forcercaster="0"></renderer-v2>
+<customProperties></customProperties>
+<blendMode>0</blendMode>
+<featureBlendMode>0</featureBlendMode>
+<layerOpacity>1</layerOpacity>
+<SingleCategoryDiagramRenderer diagramType="Histogram" attributeLegend="1"></SingleCategoryDiagramRenderer>
+<DiagramLayerSettings priority="0" linePlacementFlags="18" dist="0" showAll="1" placement="1" obstacle="0" zIndex="0"></DiagramLayerSettings>
+<geometryOptions removeDuplicateNodes="0" geometryPrecision="0"></geometryOptions>
+<fieldConfiguration></fieldConfiguration>
+<aliases></aliases>
+<excludeAttributesWMS>
+<excludeAttributesWFS>
+<defaults></defaults>
+<constraints></constraints>
+<constraintExpressions></constraintExpressions>
+<expressionFields>
+<attributeactions></attributeactions>
+<attributetableconfig actionWidgetStyle="dropDown" sortExpression="" sortOrder="0"></attributetableconfig>
+<conditionalstyles></conditionalstyles>
+<editform tolerant="1"/>
+<editforminit>
+<editforminitcodesource>0</editforminitcodesource>
+<editforminitfilepath>
+<editforminitcode></editforminitcode>
+<featformsuppress>0</featformsuppress>
+<editorlayout>generatedlayout</editorlayout>
+<editable></editable>
+<labelOnTop></labelOnTop>
+<widgets>
+<previewExpression>ID</previewExpression>
+<mapTip>
</maplayer>
</projectlayers>

```

Abb. 32.2: The expanded top level ProjectLayers tag of a QGS file

### 32.3.2 QLR – die QGIS-Layerdefinitionsdatei

Eine Layerdefinitionsdatei (QLR) ist eine XML-Datei, die zusätzlich zu den QGIS-Stil-Informationen einen Verweis auf die Layer-Datenquelle enthält.

Der Anwendungsfall für diese Datei ist einfach: eine einzelne Datei zu haben, mit der die Datenquelle geöffnet und alle zugehörigen Stil-Informationen geladen werden können. QLR-Dateien erlauben es auch, die zugrundeliegende Datenquelle mit einer leicht zu öffnenden Datei zu maskieren.

Ein Beispiel für die Verwendung von QLR ist das Öffnen von MS SQL-Layern. Anstatt den MS SQL-Verbindungsdialog aufrufen, verbinden, auswählen, laden und schließlich stylen zu müssen, können Sie einfach eine .qlr-Datei hinzufügen, die auf den richtigen MS SQL-Layer mit allen erforderlichen Stilen verweist.

In Zukunft kann eine .qlr-Datei einen Verweis auf mehr als eine Ebene enthalten.

```
-<qlr>
+<layer-tree-group name="" checked="Qt::Checked" expanded="1"></layer-tree-group>
-<maplayers>
-  <maplayer autoRefreshEnabled="0" labelsEnabled="0" autoRefreshTime="0" readOnly="0" refreshOnNotifyMessage=""
    geometry="Line" simplifyDrawingTol="1" simplifyMaxScale="1" styleCategories="AllStyleCategories" simplifyDrawingHints="1"
    maxScale="0" simplifyLocal="1" hasScaleBasedVisibilityFlag="0" type="vector" refreshOnNotifyEnabled="0" minScale="1e+8"
    simplifyAlgorithm="0">
    +<extent></extent>
    <id>inputnew_6740bb2e_0441_4af5_8dcf_305c5c4d8ca7</id>
    +<datasource></datasource>
    +<keywordList></keywordList>
    <layername>inputnew</layername>
    +<srs></srs>
    +<resourceMetadata></resourceMetadata>
    <provider encoding="UTF-8">ogr</provider>
    <vectorjoins/>
    <layerDependencies/>
    <dataDependencies/>
    <legend type="default-vector"/>
    <expressionfields/>
    +<map-layer-style-manager current="default"></map-layer-style-manager>
    <auxiliaryLayer/>
    +<flags></flags>
    +<renderer-v2 enableorderby="0" type="singleSymbol" forceraster="0" symbollevels="0"></renderer-v2>
    +<customproperties></customproperties>
    <blendMode>0</blendMode>
    <featureBlendMode>0</featureBlendMode>
    <layerOpacity>1</layerOpacity>
    +<geometryOptions removeDuplicateNodes="0" geometryPrecision="0"></geometryOptions>
    +<fieldConfiguration></fieldConfiguration>
    +<aliases></aliases>
    <excludeAttributesWMS/>
    <excludeAttributesWFS/>
    +<defaults></defaults>
    +<constraints></constraints>
    +<constraintExpressions></constraintExpressions>
    <expressionfields/>
    +<attributeactions></attributeactions>
    +<attributetableconfig sortExpression="" actionWidgetStyle="dropDown" sortOrder="0"></attributetableconfig>
    +<conditionalstyles></conditionalstyles>
    <editform tolerant="1">../src/qgisplugins/qgisbostaskdeppugin/data</editform>
    <editforminit/>
    <editforminitcodesource>0</editforminitcodesource>
    <editforminitfilepath/>
    <editforminitcode></editforminitcode>
    <featformsuppress>0</featformsuppress>
    <editorlayout>generatedlayout</editorlayout>
    <editable/>
    <labelOnTop/>
    <widgets/>
    <previewExpression>"FID"</previewExpression>
    <mapTip/>
  </maplayer>
</maplayers>
</qlr>
```

Abb. 32.3: Die Tags der obersten Ebene einer QLR-Datei

### 32.3.3 QML - Das QGIS-Stil-Dateiformat

QML ist ein XML-Format zum Speichern von Layer-Stilen.

Eine QML-Datei enthält alle Informationen, die QGIS für das Rendern von Feature-Geometrien verarbeiten kann, einschließlich Symboldefinitionen, Größen und Drehungen, Beschriftung, Deckkraft und Mischmodus und mehr.

The figure below shows the top level tags of a QML file (with only `renderer_v2` and its `symbol` tag expanded).

```

- <qgis version="3.4.13-Madeira" styleCategories="AllStyleCategories" readOnly="0" maxScale="0"
labelsEnabled="0" simplifyDrawingHints="1" hasScaleBasedVisibilityFlag="0" simplifyDrawingTol="1"
simplifyMaxScale="1" minScale="1e+8" simplifyAlgorithm="0" simplifyLocal="1">
+ <flags></flags>
- <renderer-v2 symbollevels="0" enableorderby="0" type="singleSymbol" forceraster="0">
  - <symbols>
    + <symbol clip_to_extent="1" name="0" alpha="1" type="fill" force_rhr="0"></symbol>
  </symbols>
  <rotation/>
  <sizescale/>
</renderer-v2>
+ <customproperties></customproperties>
  <blendMode>0</blendMode>
  <featureBlendMode>0</featureBlendMode>
  <layerOpacity>1</layerOpacity>
+ <SingleCategoryDiagramRenderer diagramType="Histogram" attributeLegend="1">
</SingleCategoryDiagramRenderer>
+ <DiagramLayerSettings priority="0" linePlacementFlags="18" dist="0" showAll="1" placement="1"
obstacle="0" zIndex="0">
</DiagramLayerSettings>
+ <geometryOptions removeDuplicateNodes="0" geometryPrecision="0"></geometryOptions>
+ <fieldConfiguration></fieldConfiguration>
+ <aliases></aliases>
  <excludeAttributesWMS/>
  <excludeAttributesWFS/>
+ <defaults></defaults>
+ <constraints></constraints>
+ <constraintExpressions></constraintExpressions>
  <expressionfields/>
+ <attributeactions></attributeactions>
+ <attributetableconfig actionWidgetStyle="dropDown" sortExpression="" sortOrder="0">
</attributetableconfig>
+ <conditionalstyles></conditionalstyles>
  <editform tolerant="1"/>
  <editforminit/>
  <editforminitcodesource>0</editforminitcodesource>
  <editforminitfilepath/>
+ <editforminitcode></editforminitcode>
  <featformsuppress>0</featformsuppress>
  <editorlayout>generatedlayout</editorlayout>
+ <editable></editable>
+ <labelOnTop></labelOnTop>
  <widgets/>
  <previewExpression>ID</previewExpression>
  <mapTip/>
  <layerGeometryType>2</layerGeometryType>
</qgis>

```

Abb. 32.4: The top level tags of a QML file (only the renderer\_v2 tag with its symbol tag is expanded)

## 32.4 Anhang D: QGIS R Skriptsyntax

Contributed by Matteo Ghetta - funded by [Scuola Superiore Sant'Anna](#)

Writing R scripts in Processing is a bit tricky because of the special syntax.

A Processing R script starts with defining its **Inputs** and **Outputs**, each preceded with double hash characters (##).

Before the inputs, the group to place the algorithm in can be specified. If the group already exists, the algorithm will be added to it, if not, the group will be created. In the example below, the name of the group is *My group*:

```
##My Group=group
```

### 32.4.1 Eingaben

All input data and parameters have to be specified. There are several types of inputs:

- vector: `##Layer = vector`
- vector field: `##F = Field Layer` (where *Layer* is the name of an input vector layer the field belongs to)
- raster: `##r = raster`
- table: `##t = table`
- number: `##Num = number`
- string: `##Str = string`
- boolean: `##Bol = boolean`
- elements in a dropdown menu. The items must be separated with semicolons `;; ##type=selection point;lines;point+lines`

### 32.4.2 Ausgaben

Genau wie die Eingaben müssen alle Ausgaben am Beginn des Skriptes definiert werden:

- vector: `##output= output vector`
- raster: `##output= output raster`
- table: `##output= output table`
- plots: `##output_plots_to_html (##showplots in earlier versions)`
- To show R output in the *Result Viewer*, put `>` in front of the command whose output you would like to show.

### 32.4.3 Syntax Summary for QGIS R scripts

A number of input and output parameter types are offered.

## Input parameter types

Parameter	Syntax Beispiel	Rückgabeobjekte
vector	Layer = vector	sf object (or SpatialDataFrame object, if <code>##load_vector_using_rgdal</code> is specified)
vector point	Layer = vector point	sf object (or SpatialDataFrame object, if <code>##load_vector_using_rgdal</code> is specified)
vector line	Layer = vector line	sf object (or SpatialDataFrame object, if <code>##load_vector_using_rgdal</code> is specified)
vector polygon	Layer = vector polygon	sf object (or SpatialPolygonsDataFrame object, if <code>##load_vector_using_rgdal</code> is used)
multiple vector	Layer = multiple vector	sf object (or SpatialDataFrame objects if <code>##load_vector_using_rgdal</code> is specified)
table	Layer = table	Datenrahmenumwandlung von csv, voreingestelltes Objekt der <code>read.csv</code> Funktion
field	Field = Field Layer	Name des ausgewählten Feldes, z.B. "Area"
raster	Layer = raster	RasterBrick Objekt, voreingestelltes Objekt des <code>raster</code> Pakets
multiple raster	Layer = multiple raster	RasterBrick Objekt, voreingestelltes Objekt des <code>raster</code> Pakets
number	N = number	gewählte Ganzzahl oder Fließkommazahl
string	S = string	in der Box eingefügte Zeichenkette
longstring	LS = longstring	in der Box eingefügte Zeichenkette, kann länger als eine normale Zeichenkette sein
selection	S = selection first;second;third	Zeichenkette des im dropdown-Menü gewählten Objektes
crs	C = crs	Zeichenkette des sich ergebenden KBS im Format: "EPSG:4326"
extent	E = extent	Ausdehnungsobjekt des <code>raster</code> Pakets, man kann Werte als <code>E@xmin`</code> extrahieren
point	P = point	beim Klicken auf die Karte erhält man die Koordinaten des Punktes
file	F = file	Pfad der gewählten Datei, z.B. „/home/matteo/file.txt“
folder	F = folder	Pfad des gewählten Ordners, z.B. „/home/matteo/Downloads“

A parameter can be **OPTIONAL**, meaning that it can be ignored.

In order to set an input as optional, you add the string `optional` **before** the input, e.g:

```
##Layer = vector
##Field1 = Field Layer
##Field2 = optional Field Layer
```

## Output parameter types

Parameter	Syntax Beispiel
vector	Output = output vector
raster	Output = output raster
table	Output = output table
file	Output = output file

**Bemerkung:** You can save plots as png from the *Processing Result Viewer*, or you can choose to save the plot directly from the algorithm interface.

### Skript Hauptteil

The script body follows R syntax and the **Log** panel can help you if there is something wrong with your script.

**Remember** that you have to load all additional libraries in the script:

```
library(sp)
```

### 32.4.4 Beispiele

#### Beispiel mit Vektorausgabe

Sehen wir uns einen Algorithmus aus der Onlinesammlung an, der zufällige Punkte innerhalb der Ausdehnung eines Eingabelayers erstellt:

```
##Point pattern analysis=group
##Layer=vector polygon
##Size=number 10
##Output=output vector
library(sp)
spatpoly = as(Layer, "Spatial")
pts=spsample(spatpoly, Size, type="random")
spdf=SpatialPointsDataFrame(pts, as.data.frame(pts))
Output=st_as_sf(spdf)
```

Explanation (per line in the script):

1. Point pattern analysis ist die Gruppe des Algorithmus
2. Layer ist der **Eingabevektorlayer**
3. Size is a **numerical** parameter with a default value of 10
4. Output ist der **Vektorlayer** der vom Algorithmus erstellt wird
5. library(sp) loads the **sp** library
6. spatpoly = as(Layer, "Spatial") translate to an sp object
7. Call the spsample function of the sp library and run it using the input defined above (Layer and Size)
8. Create a *SpatialPointsDataFrame* object using the SpatialPointsDataFrame function
9. Create the output vector layer using the st\_as\_sf function

That's it! Just run the algorithm with a vector layer you have in the QGIS Legend, choose the number of random point. The resulting layer will be added to your map.

#### Beispiel mit Rasterausgabe

The following script will perform basic ordinary kriging to create a raster map of interpolated values from a specified field of the input point vector layer by using the autoKrige function of the automap R package. It will first calculate the kriging model and then create a raster. The raster is created with the raster function of the raster R package:

```
##Basic statistics=group
##Layer=vector point
##Field=Field Layer
##Output=output raster
##load_vector_using_rgdal
require("automap")
require("sp")
```

(Fortsetzung auf der nächsten Seite)



(Fortsetzung der vorherigen Seite)

```
require("raster")
table=as.data.frame(Layer)
coordinates(table)= ~coords.x1+coords.x2
c = Layer[[Field]]
kriging_result = autoKrige(c~1, table)
prediction = raster(kriging_result$krige_output)
Output<-prediction
```

By using `##load_vector_using_rgdal`, the input vector layer will be made available as a `SpatialDataFrame` objects, so we avoid having to translate it from an `sf` object.

### Beispiel mit Tabellenausgabe

Lassen Sie uns den Algorithmus `Summary Statistics` bearbeiten, so dass als Ausgabe eine Tabelle (csv) erzeugt wird.

Der Hauptteil des Skriptes sieht wie folgt aus:

```
##Basic statistics=group
##Layer=vector
##Field=Field Layer
##Stat=Output table
Summary_statistics<-data.frame(rbind(
  sum(Layer[[Field]]),
  length(Layer[[Field]]),
  length(unique(Layer[[Field]])),
  min(Layer[[Field]]),
  max(Layer[[Field]]),
  max(Layer[[Field]])-min(Layer[[Field]]),
  mean(Layer[[Field]]),
  median(Layer[[Field]]),
  sd(Layer[[Field]]),
  row.names=c("Sum:", "Count:", "Unique values:", "Minimum value:", "Maximum value:",
  ↪ "Range:", "Mean value:", "Median value:", "Standard deviation:"))
colnames(Summary_statistics)<-c(Field)
Stat<-Summary_statistics
```

Die dritte Zeile gibt als Eingabe **Vector Field** vor. In der vierten Zeile wird die Ausgabe des Algorithmus als Tabelle festgelegt.

In der letzten Zeile wird das im Skript erstellte Objekt `Stat` in eine `csv` Tabelle umgewandelt.

### Beispiel mit Konsolenausgabe

We can use the previous example and instead of creating a table, print the result in the **Result Viewer**:

```
##Basic statistics=group
##Layer=vector
##Field=Field Layer
Summary_statistics<-data.frame(rbind(
  sum(Layer[[Field]]),
  length(Layer[[Field]]),
  length(unique(Layer[[Field]])),
  min(Layer[[Field]]),
  max(Layer[[Field]]),
  max(Layer[[Field]])-min(Layer[[Field]]),
  mean(Layer[[Field]]),
  median(Layer[[Field]]),
  sd(Layer[[Field]]), row.names=c("Sum:", "Count:", "Unique values:", "Minimum value:",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
↩ "Maximum value:", "Range:", "Mean value:", "Median value:", "Standard deviation:") )  
colnames(Summary_statistics) <- c(Field)  
>Summary_statistics
```

The script is exactly the same as the one above except for two edits:

1. no output specified (the fourth line has been removed)
2. the last line begins with `>`, telling Processing to make the object available through the result viewer

### Beispiel mit Plot

To create plots, you have to use the `##output_plots_to_html` parameter as in the following script:

```
##Basic statistics=group  
##Layer=vector  
##Field=Field Layer  
##output_plots_to_html  
####output_plots_to_html  
qqnorm(Layer[[Field]])  
qqline(Layer[[Field]])
```

The script uses a field (`Field`) of a vector layer (`Layer`) as input, and creates a *QQ Plot* (to test the normality of the distribution).

The plot is automatically added to the Processing *Result Viewer*.

## 32.5 Anhang E: Netzwerkverbindungen der QGIS-Anwendung

This is a list of the pre-configured/automated network connections that QGIS makes. Some of the connections are user initiated because they require an action from the user before they take place, others happen automatically.

Name	Funktion	UX mode	Server	Information sent to server	Information stored on server
<b>qgis.org</b>					
Python API help	Browser PyQGIS documentation	User initiated	<a href="https://qgis.org/pyqgis/%1/search.html?q=%2">https://qgis.org/pyqgis/%1/search.html?q=%2</a>	IP, QGIS version, OS	IP in server log
<b>version.qgis.org</b>					
New version check	Notify the user on new QGIS versions availability	Automatic	<a href="https://version.qgis.org">https://version.qgis.org</a>	IP, QGIS version, OS	IP in server log
<b>feed.qgis.org</b>					
QGIS Feed	Retrieve QGIS news from the feed	Automatic	<a href="https://feed.qgis.org">https://feed.qgis.org</a>	IP, QGIS version, language code, last download timestamp, OS	IP in server log; QGIS version, OS and IP are aggregated and used to collect some statistics
<b>plugins.qgis.org</b>					
Check for plugin updates	Notify the user about plugin updates	User initiated/automatic (configurable)	<a href="https://plugins.qgis.org">https://plugins.qgis.org</a>	IP, QGIS version, OS	IP in server log
Plugins list	Retrieve the list of plugins	User initiated	<a href="https://plugins.qgis.org">https://plugins.qgis.org</a>	IP, QGIS version, OS	IP in server log
Plugin installation	Download and install a plugin package	User initiated	<a href="https://plugins.qgis.org">https://plugins.qgis.org</a>	IP, QGIS version, OS	Increase plugin download counter by one
Styles	List user contributed styles	User initiated	<a href="https://plugins.qgis.org/styles">https://plugins.qgis.org/styles</a>	IP, QGIS version, OS	Increase download counter by one
<b>3rd party</b>					
Terrain data	Produce a DEM for 3D views	User initiated	<a href="https://s3.amazonaws.com/elevation-tiles-prod/terrarium/{z}/{x}/{y}">https://s3.amazonaws.com/elevation-tiles-prod/terrarium/{z}/{x}/{y}</a>	IP, QGIS version, OS	see Amazon TOS
Google Map Geocode	Geocoding services	User initiated	<a href="https://maps.googleapis.com/maps/api/geocode/json">https://maps.googleapis.com/maps/api/geocode/json</a>	IP, QGIS version, OS	See google maps API TOS
Nominatim Geocode	Geocoding services	User initiated	<a href="https://nominatim.qgis.org/ui/search.html">https://nominatim.qgis.org/ui/search.html</a>	IP, QGIS version, OS	
Geodetic grid	Add a new PROJ grid	User initiated	<a href="https://cdn.proj.org">https://cdn.proj.org</a>	IP, PROJ version	Access logs are permanently deleted after one day



---

### Literatur und Internetreferenzen

---

GDAL-SOFTWARE-SUITE. Geospatial data abstraction library. <https://gdal.org>, 2013.

GRASS-PROJECT. Geographic resource analysis support system. <https://grass.osgeo.org>, 2013.

NETELER, M., AND MITASOVA, H. Open source gis: A grass gis approach, 2008.

OGR-SOFTWARE-SUITE. Geospatial data abstraction library. <https://gdal.org>, 2013.

OPEN-GEOSPATIAL-CONSORTIUM. Web map service (1.1.1) implementation specification. [https://portal.ogc.org/files/?artifact\\_id=1081&version=1&format=pdf](https://portal.ogc.org/files/?artifact_id=1081&version=1&format=pdf), 2002.

OPEN-GEOSPATIAL-CONSORTIUM. Web map service (1.3.0) implementation specification. [https://portal.ogc.org/files/?artifact\\_id=14416&format=pdf](https://portal.ogc.org/files/?artifact_id=14416&format=pdf), 2004.

POSTGIS-PROJECT. Spatial support for postgresql. <http://www.refrations.net/products/postgis/>, 2013.