QGIS 3.10 User Guide

QGIS Project

09.12.2020

Inhaltsverzeichnis

1 Präambel **1**
1.1 Was ist neu in QGIS 3.10 . . . 2

2 Vorwort **3**

3 Gebrauch der Dokumentation **5**
3.1 GUI Schreibstile . . . 5
3.2 Text oder Tastatur Schreibstile . . . 6
3.3 Betriebssystemspezifische Anweisungen . . . 6

4 Funktionalitäten **7**
4.1 Daten visualisieren . . . 7
4.2 Daten erkunden, abfragen und Karten layouten . . . 8
4.3 Daten erstellen, editieren, verwalten und exportieren . . . 8
4.4 Daten analysieren . . . 9
4.5 Karten im Internet veröffentlichen . . . 9
4.6 Erweiterte QGIS Funktionalität durch Erweiterungen . . . 9
4.6.1 Kern Plugins . . . 9
4.6.2 Externe Python Plugins . . . 10
4.7 Python-Konsole . . . 10
4.8 Bekannte Probleme . . . 10
4.8.1 Begrenzung der Anzahl von geöffneten Dateien . . . 10

5 Der erste Einstieg **11**
5.1 QGIS installieren . . . 11
5.1.1 Installation mit Binärdateien . . . 11
5.1.2 Kompilieren des Quellcodes . . . 11
5.1.3 Auf externen Medien installieren . . . 12
5.1.4 Beispieldaten herunterladen . . . 12
5.2 QGIS starten und beenden . . . 13
5.3 Beispielsitzung: Laden von Raster- und Vektorlayern . . . 13

6 Arbeiten mit Projektdateien **19**
6.1 Vorstellung von QGIS-Projekten . . . 19
6.2 Ausgaben erzeugen . . . 21

7 QGIS GUI **23**
7.1 Menüleiste . . . 24
7.1.1 Projekt . . . 24
7.1.2 Bearbeiten . . . 25
7.1.3 Ansicht . . . 28

7.1.4 Layer . . . 31
7.1.5 Einstellungen . . . 33
7.1.6 Erweiterungen . . . 34
7.1.7 Vektor . . . 34
7.1.8 Raster . . . 35
7.1.9 Datenbank . . . 36
7.1.10 Web . . . 36
7.1.11 Mesh . . . 37
7.1.12 Verarbeitung . . . 37
7.1.13 Hilfe . . . 37
7.1.14 QGIS . . . 37
7.2 Bedienfelder und Werkzeugkästen . . . 38
7.2.1 Werkzeugkästen . . . 38
7.2.2 Bedienfelder . . . 38
7.3 Kartenfenster . . . 41
7.3.1 Exploring the map view . . . 41
7.3.2 Setting additional map views . . . 41
7.3.3 Exporting the map view . . . 42
7.4 3D-Ansicht . . . 45
7.4.1 Navigation options . . . 46
7.4.2 Creating an animation . . . 47
7.4.3 Scene Configuration . . . 47
7.4.4 3D vector layers . . . 50
7.5 Statusleiste . . . 50

8 The Browser panel **53**
8.1 Resources that can be opened / run from the Browser . . . 56
8.2 Browser panel top-level entries . . . 56
8.2.1 Favorites . . . 56
8.2.2 Räumliche Lesezeichen . . . 56
8.2.3 Home . . . 56
8.2.4 / . . . 57
8.2.5 Geopackage . . . 57
8.2.6 SpatiaLite . . . 57
8.2.7 PostGIS . . . 57
8.2.8 MSSQL . . . 58
8.2.9 DB2 . . . 58
8.2.10 WMS/WMTS . . . 58
8.2.11 Vector Tiles . . . 59
8.2.12 XYZ Tiles . . . 59
8.2.13 WCS . . . 59
8.2.14 WFS / OGC API - Features . . . 59
8.2.15 OWS . . . 60
8.2.16 ArcGIS Map Service . . . 60
8.2.17 ArcGIS Features Service . . . 60
8.2.18 GeoNode . . . 60
8.3 Resources . . . 60

9 QGIS anpassen **63**
9.1 Optionen . . . 63
9.1.1 Allgemeine Einstellungen . . . 64
9.1.2 Systemeinstellungen . . . 65
9.1.3 CRS Settings . . . 67
9.1.4 Data Sources Settings . . . 68
9.1.5 Rendering Settings . . . 71
9.1.6 Canvas and Legend Settings . . . 73
9.1.7 Map tools Settings . . . 73
9.1.8 Colors Settings . . . 75

9.1.9 Digitizing Settings . . . 76
9.1.10 Layouts Settings . . . 78
9.1.11 GDAL Settings . . . 78
9.1.12 Variables Settings . . . 80
9.1.13 Authentication Settings . . . 81
9.1.14 Network Settings . . . 81
9.1.15 Locator Settings . . . 82
9.1.16 Advanced Settings . . . 85
9.1.17 Acceleration Settings . . . 85
9.1.18 Processing Settings . . . 86
9.2 Working with User Profiles . . . 86
9.3 Projekteigenschaften . . . 88
9.3.1 General Properties . . . 88
9.3.2 Metadata Properties . . . 90
9.3.3 CRS Properties . . . 90
9.3.4 Default Styles Properties . . . 90
9.3.5 Data Sources Properties . . . 91
9.3.6 Relations Properties . . . 92
9.3.7 Variables Properties . . . 93
9.3.8 Macros Properties . . . 93
9.3.9 QGIS Server Properties . . . 93
9.4 Anpassung . . . 93
9.5 Tastenkürzel . . . 97
9.6 Running QGIS with advanced settings . . . 98
9.6.1 Command line and environment variables . . . 98
9.6.2 Deploying QGIS within an organization . . . 102

10 Arbeiten mit Projektionen **105**
10.1 Überblick zur Projektionsunterstützung . . . 105
10.2 Layer Coordinate Reference Systems . . . 105
10.3 Project Coordinate Reference Systems . . . 107
10.4 Coordinate Reference System Selector . . . 109
10.5 Eigenes Koordinatenbezugssystem definieren . . . 110
10.5.1 Integrate an NTv2-transformation in QGIS . . . 110
10.6 Datum Transformations . . . 110

11 Allgemeine Werkzeuge **113**
11.1 Hilfe . . . 113
11.2 Bedienfelder . . . 113
11.2.1 Layerfenster . . . 113
11.2.2 Layer Styling Panel . . . 118
11.2.3 Layer Order Panel . . . 119
11.2.4 Overview Panel . . . 119
11.2.5 Log Messages Panel . . . 119
11.2.6 Undo/Redo Panel . . . 119
11.2.7 Statistical Summary Panel . . . 119
11.3 Layer/Gruppen einbinden . . . 122
11.4 Working with the map canvas . . . 123
11.4.1 Layeranzeige kontrollieren . . . 123
11.4.2 Zoomen und Karte verschieben . . . 125
11.4.3 Räumliche Lesezeichen . . . 125
11.4.4 Dekorationen . . . 127
11.4.5 Beschriftungstools . . . 132
11.4.6 Messen . . . 136
11.5 Interacting with features . . . 138
11.5.1 Selecting features . . . 138
11.5.2 Identifying Features . . . 141
11.6 Save and Share Layer Properties . . . 144

11.6.1 Managing Custom Styles . . . 144
11.6.2 Storing Styles in a File or a Database . . . 145
11.6.3 Layer definition file . . . 147
11.7 Storing values in Variables . . . 147
11.8 Authentifizierung . . . 147
11.9 Common widgets . . . 149
11.9.1 Farbauswahl . . . 149
11.9.2 Symbol Widget . . . 152
11.9.3 Font Selector . . . 152
11.9.4 Unit Selector . . . 153
11.9.5 Mischmodi . . . 154
11.9.6 Datendefinierte Übersteuerung Setup . . . 154

12 Die Stilbibliothek **157**
12.1 The Style Manager . . . 157
12.1.1 The Style Manager dialog . . . 157
12.1.2 Setting a Color Ramp . . . 162
12.2 The Symbol Selector . . . 163
12.2.1 The symbol layer tree . . . 165
12.2.2 Configuring a symbol . . . 165
12.3 Setting a label . . . 173
12.3.1 Formatting the label text . . . 173
12.3.2 Callouts tab . . . 180
12.3.3 Placement tab . . . 181
12.3.4 Rendering tab . . . 183

13 Verwaltung von Datenquellen **185**
13.1 Öffnen von Daten . . . 185
13.1.1 The Browser Panel . . . 187
13.1.2 The DB Manager . . . 190
13.1.3 Provider-based loading tools . . . 190
13.1.4 QGIS Custom formats . . . 206
13.1.5 QLR - QGIS Layer Definition File . . . 206
13.1.6 Connecting to web services . . . 206
13.1.7 Handling broken file paths . . . 208
13.2 Erstellung von Layern . . . 208
13.2.1 Neue Vektorlayer erstellen . . . 208
13.2.2 Creating new layers from an existing layer . . . 214
13.2.3 Creating new DXF files . . . 216
13.2.4 Creating new layers from the clipboard . . . 218
13.2.5 Creating virtual layers . . . 219
13.3 Exploring Data Formats and Fields . . . 221
13.3.1 Raster data . . . 221
13.3.2 Vektordaten . . . 222

14 Arbeiten mit Vektordaten **231**
14.1 Vektorlayereigenschaften . . . 231
14.1.1 Information Properties . . . 232
14.1.2 Source Properties . . . 232
14.1.3 Symbology Properties . . . 235
14.1.4 Labels Properties . . . 256
14.1.5 Diagrams Properties . . . 265
14.1.6 3D View Properties . . . 271
14.1.7 Fields Properties . . . 272
14.1.8 Attributes Form Properties . . . 273
14.1.9 Joins Properties . . . 279
14.1.10 Auxiliary Storage Properties . . . 281
14.1.11 Actions Properties . . . 288
14.1.12 Display Properties . . . 293

14.1.13 Rendering Properties . . . 294
14.1.14 Variables Properties . . . 296
14.1.15 Metadata Properties . . . 296
14.1.16 Dependencies Properties . . . 296
14.1.17 Legend Properties . . . 296
14.1.18 QGIS Server Properties . . . 297
14.1.19 Digitizing Properties . . . 297
14.2 Ausdrücke . . . 301
14.2.1 Der Ausdruckseditor . . . 301
14.2.2 Liste der Funktionen . . . 302
14.2.3 Funktions Editor . . . 321
14.3 Mit Attributtabelle arbeiten . . . 324
14.3.1 Foreword: Spatial and non-spatial tables . . . 324
14.3.2 Introducing the attribute table interface . . . 324
14.3.3 Interacting with features in an attribute table . . . 329
14.3.4 Using action on features . . . 331
14.3.5 Editiere Attributwerte . . . 332
14.3.6 Creating one or many to many relations . . . 336
14.4 Editierfunktionen . . . 344
14.4.1 Einstellen der Fangtoleranz und des Suchradius . . . 344
14.4.2 Topologisches Editieren . . . 346
14.4.3 Einen vorhandenen Layer editieren . . . 348
14.4.4 Erweiterte Digitalisierung . . . 356
14.4.5 Formen digitalisieren . . . 363
14.4.6 Das Bedienfeld Erweiterte Digitalisierung . . . 366
14.4.7 Mit Verarbeitung Objekte innerhalb eines Layers verändern . . . 372

15 Arbeiten mit Rasterdaten **375**
15.1 Dialogfenster Rasterlayereigenschaften . . . 375
15.1.1 Information Properties . . . 376
15.1.2 Source Properties . . . 376
15.1.3 Symbology Properties . . . 377
15.1.4 Transparency Properties . . . 384
15.1.5 Histogram Properties . . . 385
15.1.6 Rendering Properties . . . 385
15.1.7 Pyramids Properties . . . 385
15.1.8 Metadata Properties . . . 387
15.1.9 Legend Properties . . . 387
15.1.10 QGIS Server Properties . . . 389
15.2 Rasteranalyse . . . 389
15.2.1 Rasterrechner . . . 389
15.2.2 Raster Ausrichtung . . . 392

16 Working with Mesh Data **395**
16.1 What's a mesh? . . . 395
16.2 Supported formats . . . 397
16.3 Mesh Dataset Properties . . . 397
16.3.1 Information Properties . . . 397
16.3.2 Source Properties . . . 398
16.3.3 Symbology Properties . . . 398

17 Kartenlayout **403**
17.1 Überblick über das Drucklayout . . . 403
17.1.1 Sample Session for beginners . . . 403
17.1.2 Layoutverwaltung . . . 404
17.1.3 Menüs, Werkzeuge und Bedienfelder des Drucklayouts . . . 405
17.2 Layout-Elemente . . . 418
17.2.1 Layout Items Common Options . . . 418
17.2.2 Das Kartenelement . . . 423

17.2.3 The 3D Map Item 431
17.2.4 Das Beschriftungselement 432
17.2.5 Das Legendenelement 434
17.2.6 Das Maßstabselement 439
17.2.7 Das Attributtabellenelement 443
17.2.8 The Picture and the North Arrow Items 449
17.2.9 Das HTML-Rahmen Element 451
17.2.10 The Shape Items 455
17.3 Eine Ausgabe erzeugen 458
17.3.1 Exporteinstellungen 458
17.3.2 Speichern als Rasterbild 459
17.3.3 Als SVG exportieren 460
17.3.4 Als PDF exportieren 461
17.3.5 Einen Atlas erzeugen 462
17.4 Creating a Report 466
17.4.1 What is it? 466
17.4.2 Get started 467
17.4.3 Layout Report Workspace 467
17.4.4 Exporteinstellungen 480

18 Arbeiten mit OGC Daten **483**
18.1 QGIS als OGC Datenclient 483
18.1.1 WMS/WMTS Client 484
18.1.2 WCS Client 492
18.1.3 WFS und WFS-T Klient 492
18.2 QGIS as OGC Data Server 496
18.2.1 Der erste Einstieg 496
18.2.2 Services 509
18.2.3 Plugins 541
18.2.4 Advanced configuration 542
18.2.5 Containerized deployment 546

19 Arbeiten mit GPS Daten **555**
19.1 GPS Plugin 555
19.1.1 Was ist GPS? 555
19.1.2 GPS-Daten aus einer Datei laden 555
19.1.3 GPSBabel 556
19.1.4 GPS-Daten importieren 556
19.1.5 GPS-Daten von einem Empfänger herunterladen 557
19.1.6 GPS-Daten auf einen Empfänger hochladen 557
19.1.7 Neues GPS-Gerät definieren 558
19.1.8 Downloaden von Punkten/Spuren von GPS Einheiten 558
19.2 Live GPS tracking 560
19.2.1 Positionskoordinaten 560
19.2.2 GPS Signalstärke 560
19.2.3 GPS Optionen 560
19.2.4 Ein Bluetooth GPS fürs Live Tracking anbinden 563
19.2.5 GPSMAP 60cs verwenden 563
19.2.6 BTGP-38KM Datenlogger verwenden (nur Bluetooth) 564
19.2.7 BlueMax GPS-4044 Datenlogger verwenden (sowohl BT als auch USB) 564

20 Authentifizierungssystem **565**
20.1 Authentifizierungssystem Übersicht 565
20.1.1 Authentifikationsdatenbank 566
20.1.2 Hauptpasswort 566
20.1.3 Authentifikationskonfiguration 567
20.1.4 Authentifizierungsmethoden 567
20.1.5 Hauptpasswort und Auth Konfig Hilfsmittel 573
20.1.6 Gebrauche Authentifikationskonfiguration 574

20.1.7 Python-Bindung . . . 574
20.2 Benutzerauthentifizierung Workflows . . . 575
20.2.1 HTTP(S) Authentifikation . . . 575
20.2.2 Datenbank Authentifikation . . . 575
20.2.3 PKI Authentifikation . . . 575
20.2.4 Defekte Layer behandeln . . . 581
20.2.5 Kennung der Authentifikationskonfiguration ändern . . . 583
20.2.6 QGIS Server Unterstützung: . . . 584
20.2.7 SSL Serverausnahme . . . 584
20.3 Sicherheitsüberprüfung . . . 587
20.3.1 Einschränkungen . . . 588

21 GRASS GIS Integration **589**
21.1 Beispieldatensatz . . . 589
21.2 GRASS Layer visualisieren . . . 589
21.3 Daten in eine GRASS LOCATION importieren via drag and drop . . . 590
21.4 GRASS Daten im QGIS Browser verwalten . . . 590
21.5 GRASS Optionen . . . 590
21.6 GRASS Plugin starten . . . 590
21.7 GRASS Mapset öffnen . . . 591
21.8 Information zur GRASS-Datenbank . . . 591
21.9 Daten in eine GRASS LOCATION importieren . . . 592
21.9.1 Eine neue GRASS LOCATION erstellen . . . 592
21.9.2 Eine neue GRASS MAPSET erstellen . . . 594
21.10 Das GRASS Vektormodell . . . 594
21.11 Einen neuen GRASS Vektorlayer erstellen . . . 595
21.12 Digitalisieren und Editieren eines GRASS Vektorlayers . . . 595
21.13 Einstellung der GRASS Region . . . 598
21.14 Die GRASS Werkzeugkiste . . . 598
21.14.1 Arbeiten mit GRASS Modulen . . . 599
21.14.2 GRASS Beispielanwendung . . . 601
21.14.3 Anpassen der Module . . . 605

22 QGIS Verarbeitung Umgebung **607**
22.1 Einführung . . . 607
22.2 Configuring the Processing Framework . . . 610
22.3 The Toolbox . . . 611
22.3.1 Der Algorithmus Dialog . . . 612
22.3.2 Von Algorithmen erstellte Datenobjekte . . . 618
22.4 Das Protokoll . . . 619
22.4.1 Das Verarbeitung Protokoll . . . 619
22.4.2 Das Verarbeitung Log . . . 620
22.5 Die Grafische Modellierung . . . 620
22.5.1 Definition von Eingaben . . . 620
22.5.2 Definition des Arbeitsablaufs . . . 622
22.5.3 Speichern und laden von Modellen . . . 626
22.5.4 Ein Modell editieren . . . 627
22.5.5 Editieren der Modell-Hilfe Datei und der Metainformationen . . . 628
22.5.6 Exportieren eines Modells als Python Skript . . . 628
22.5.7 Zu den verfügbaren Algorithmen . . . 630
22.6 Die Batch Processing Schnittstelle . . . 630
22.6.1 Einführung . . . 630
22.6.2 Die Parameter-Tabelle . . . 631
22.6.3 Füllen der Parameter-Tabelle . . . 631
22.6.4 Ausführen eines Batch-Prozesses . . . 632
22.7 Verarbeitung Algorithmen von der Konsole aus verwenden . . . 632
22.7.1 Algorithmen von der Python Konsole aus aufrufen . . . 633
22.7.2 Scripte erstellen und diese aus der Werkzeugkiste starten. . . . 637

22.7.3 Pre- und Post-execution Script Hooks 641
22.8 Writing new Processing algorithms as Python scripts 641
22.8.1 Extending QgsProcessingAlgorithm 641
22.8.2 The @alg decorator 645
22.8.3 Input and output types for Processing Algorithms 647
22.8.4 Handing algorithm output 648
22.8.5 Die Kommunikation mit dem Benutzer 648
22.8.6 Ihre Scripte dokumentieren 649
22.8.7 Flags 649
22.8.8 Bewährte Verfahren für das Schreiben von Skript-Algorithmen 649
22.9 Konfiguration externer Anwendungen 649
22.9.1 Ein Hinweis für Windows Anwender 650
22.9.2 Eine Bemwerkung zu den Datenformaten 650
22.9.3 Eine Bemerkung zu Vektorlayern mit ausgewählten Objekten 650
22.9.4 SAGA 650
22.9.5 R scripts 652
22.9.6 R libraries 658
22.9.7 GRASS 658
22.9.8 LAStools 659
22.9.9 OTB Applications 659

23 Datenanbieter und Algorithmen in Verarbeitung 661
23.1 GDAL Algorithmus-Anbieter 661
23.1.1 Cartography 661
23.1.2 Datenbank 668
23.1.3 File tools 674
23.1.4 Graphics 674
23.1.5 Interpolation 680
23.1.6 Layer tools 689
23.1.7 Modeler tools 691
23.1.8 Network analysis 693
23.1.9 Raster analysis 703
23.1.10 Raster terrain analysis 720
23.1.11 Raster tools 731
23.1.12 Vektoranalyse 737
23.1.13 Vector creation 755
23.1.14 Vector general 773
23.1.15 Vector geometry 798
23.1.16 Vector overlay 906
23.1.17 Vector selection 918
23.1.18 Vector table 930
23.2 GDAL-Algorithmen Bereitstellung 944
23.2.1 Raster analysis 944
23.2.2 Raster conversion 967
23.2.3 Raster extraction 973
23.2.4 Raster miscellaneous 978
23.2.5 Rasterprojektionen 992
23.2.6 Vector conversion 995
23.2.7 Vector geoprocessing 1000
23.2.8 Vector miscellaneous 1006
23.3 LAStools algorithm provider 1014
23.3.1 blast2dem 1014
23.3.2 blast2iso 1016
23.3.3 las2dem 1017
23.3.4 las2iso 1019
23.3.5 las2las_filter 1021
23.3.6 las2las_project 1025
23.3.7 las2las_transform 1031

23.3.8 las2txt 1034
23.3.9 lasindex 1035
23.3.10 lasgrid 1036
23.3.11 lasinfo 1038
23.3.12 lasmerge 1041
23.3.13 lasprecision 1042
23.3.14 lasquery 1043
23.3.15 lasvalidate 1044
23.3.16 laszip 1044
23.3.17 txt2las 1045
23.4 TauDEM Algorithmus-Anbieter 1049
23.4.1 Grundlegende Gitteranalyse 1050
23.4.2 Spezialisierte Gitteranalyse 1062
23.4.3 Stromnetzwerkanalyse 1081
23.5 OTB Anwendungs Bereitstellung 1095

24 Plugins **1097**
24.1 QGIS Plugins 1097
24.1.1 Kernerweiterungen und externe Erweiterungen 1097
24.1.2 Der Erweiterungen Dialog 1098
24.2 Using QGIS Core Plugins 1103
24.2.1 Coordinate Capture Plugin 1103
24.2.2 DB Manager Plugin 1103
24.2.3 eVis Plugin 1106
24.2.4 Geometry Checker Plugin 1116
24.2.5 Plugin „Georeferenzierung“ 1119
24.2.6 MetaSearch Catalog Client 1124
24.2.7 Offline Editing Plugin 1130
24.2.8 Topology Checker Plugin 1130
24.3 QGIS Python Konsole 1133
24.3.1 The Interactive Console 1133
24.3.2 The Code Editor 1135
24.3.3 Optionen 1135

25 Hilfe und Support **1137**
25.1 Mailinglisten 1137
25.1.1 QGIS Users 1137
25.1.2 QGIS Developers 1137
25.1.3 QGIS Community Team 1137
25.1.4 QGIS Translations 1138
25.1.5 QGIS Project Steering Committee (PSC) 1138
25.1.6 QGIS-Anwendergruppen 1138
25.2 IRC 1138
25.3 Commercial support 1138
25.4 BugTracker 1138
25.5 Blog 1139
25.6 Plugins 1139
25.7 Wiki 1139

26 Contributors **1141**
26.1 Autoren 1141
26.2 Translators 1142

27 Anhänge **1143**
27.1 Anhang A: GNU General Public License 1143
27.2 Appendix B: GNU Free Documentation License 1147
27.3 Anhang C: QGIS-Dateiformate 1153
27.3.1 QGS/QGZ - Das QGIS-Projekt-Dateiformat 1153
27.3.2 QLR - The QGIS Layer Definition file 1153

27.3.3 QML - Das QGIS-Stil-Dateiformat 1154
27.4 Appendix D: QGIS R script syntax 1154
27.4.1 Eingaben 1158
27.4.2 Ausgaben 1158
27.4.3 Syntax Summary for QGIS R scripts 1158
27.4.4 Beispiele 1160

28 Literatur und Internetreferenzen 1163

KAPITEL 1

Präambel

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 *Contributors*.

Internet: https://www.qgis.org

Lizenz des Dokuments

1.1 Was ist neu in QGIS 3.10

Diese Version von QGIS enthält Hunderte von Fehlerkorrekturen und viele neue Funktionen und Verbesserungen. Wir empfehlen, diese Version statt früherer Versionen zu verwenden. Eine Liste der neuen Funktionen finden Sie in den Änderungsprotokollen unter https://qgis.org/en/site/forusers/visualchangelogs.html.

KAPITEL 2

Vorwort

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

QGIS ist ein freies (Open Source), geografisches Informationssystem. Die Idee zu dem Projekt wurde im Mai 2002 geboren und bereits im Juni desselben Jahres bei SourceForge etabliert. Wir haben hart daran gearbeitet, GIS Software (die bisher üblicherweise teuer und proprietär ist) kostenfrei für jeden, der Zugang zu einem PC hat, bereitzustellen. QGIS kann unter den meisten Unix Systemen, Windows und MacOSX betrieben werden. QGIS wurde mit Hilfe des Qt Toolkit (https://www.qt.io) und C++ entwickelt. Dadurch ist QGIS sehr benutzerfreundlich und besitzt eine einfach und intuitiv zu bedienende grafische Benutzeroberfläche.

QGIS aims to be a user-friendly GIS, providing common functions and features. The initial goal of the project was to provide a GIS data viewer. QGIS has reached the point in its evolution where it is being used for daily GIS data-viewing needs, for data capture, for advanced GIS analysis, and for presentations in the form of sophisticated maps, atlases and reports. QGIS supports a wealth of raster and vector data formats, with new format support easily added using the plugin architecture.

QGIS is released under the GNU General Public License (GPL). Developing QGIS under this license means that you can inspect and modify the source code, and guarantees that you, our happy user, will always have access to a GIS program that is free of cost and can be freely modified. You should have received a full copy of the license with your copy of QGIS, and you can also find it in Appendix *Anhang A: GNU General Public License*.

Tipp: Aktuellste Dokumentation

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

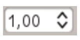
KAPITEL 3

Gebrauch der Dokumentation

Dieser Abschnitt beschreibt die in diesem Handbuch benutzten einheitlichen Schreibstile.

3.1 GUI Schreibstile

Die GUI Schreibstile der sollen das Erscheinungsbild der Grafischen Benutzeroberfläche nachahmen. Im Allgemeinen gibt ein Stil das einfache Erscheinungsbild wieder, so dass der Benutzer die GUI nach etwas das wie die Instruktionen 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 Stile die sich auf Texte, Tastaturbefehle und Codes beziehen um verschiedene Einheiten, wie Klassen und Methoden anzuzeigen. Diese Stile entsprechen 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`

Kodezeilen 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* *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
- Oder jenes

oder als Paragraph

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.

KAPITEL 4

Funktionalitäten

QGIS offers a wealth of GIS functions, provided by core features and plugins. The locator bar makes it easy to search for functions, datasets and more.

A short summary of six general categories of features and plugins is presented below, followed by first insights into the integrated Python console.

4.1 Daten visualisieren

Es ist möglich, Vektor- und Rasterdaten (in 2D und 3D) in unterschiedlichen Formaten und mit verschiedenen Projektionen anzuschauen, 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 wiie z.B. GeoTIFF, ERDAS IMG, ArcInfo ASCII GRID, JPEG, PNG und viele mehr. Siehe Kapitel *Arbeiten mit Rasterdaten*.
- Mesh data (TINs and regular grids are supported). See *Working with Mesh Data*.
- GRASS Raster- und Vektordaten aus GRASS Datenbanken (Location/Mapset). Siehe Kapitel *GRASS GIS Integration*.
- Online Geodaten welche als OGC Webdienst, darunter WMS, WMTS, WCS, WFS und WFS-T, bereitgestellt werden. Siehe Kapitel *Arbeiten mit OGC Daten*.

  The QGIS authentication infrastructure helps you manage user/password, certificates and keys for web services and other resources.
- Spreadsheets (ODS / XLSX)

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
- DB Manager
- Drucklayout
- Report
- Kartenübersichtsfenster
- Räumliche Bookmarks
- 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 die folgenden an:

- Vector digitizing tools
- Erzeugen und Bearbeiten vieler verschiedener Dateiformate und GRASS Vektorlayer
- Georeferenzierungsplugin um Bilder zu geocodieren
- 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 Vektorattributtabellen 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

You can perform spatial data analysis on spatial databases and other OGR-supported formats. QGIS currently offers vector analysis, raster analysis, sampling, geoprocessing, geometry and database management tools. You can also use the integrated GRASS tools, which include the complete GRASS functionality of more than 400 modules (see section *GRASS GIS Integration*). Or, you can work with the Processing plugin, which provides a powerful geospatial analysis framework to call native and third-party algorithms from QGIS, such as GDAL, SAGA, GRASS, R, and more (see section *Einführung*). All analysis functions are run in the background, allowing you to continue your work before the processing has finished.

The graphical modeller allows you to combine / chain functions into a complete workflow in an intuitive graphical environment.

4.5 Karten im Internet veröffentlichen

QGIS can be used as a WMS, WMTS, WMS-C or WFS and WFS-T client, and as a WMS, WCS or WFS server (see section *Arbeiten mit OGC Daten*). Additionally, you can publish your data on the Internet using a webserver with QGIS Server, UMN MapServer or GeoServer installed.

4.6 Erweiterte QGIS Funktionalität durch Erweiterungen

QGIS kann an Ihre speziellen Bedürfnisse mit seiner erweiterbaren Pluginarchitektur 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

Kernplugins sind:

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

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 jedoch werden 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

KAPITEL 5

Der erste Einstieg

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

If you need to build QGIS from source, please refer to the installation instructions. They are distributed with the QGIS source code in a file called `INSTALL`. You can also find them online at https://github.com/qgis/QGIS/blob/master/INSTALL.md.

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 *Systemeinstellungen* für weitere Informationen.`

5.1.4 Beispieldaten herunterladen

Dieses Benutzerhandbuch enthält Beispiele, die auf dem QGIS-Beispieldatensatz (auch `Alaska-Datensatz` genannt) basieren.

Der Windows Installer bietet die Möglichkeit, den QGIS-Beispieldatensatz herunterzuladen. Wenn diese Option aktiviert ist, werden die Daten in den Ordner `Eigene Dateien` heruntergeladen und in den Ordner `GIS Database` verschoben. Sie können diesen Ordner mit dem Windows Explorer an einen beliebigen Ort verschieben. Wenn Sie das Kontrollkästchen zur Installation des Beispieldatensatzes bei der Erstinstallation von QGIS nicht aktiviert haben, können Sie eine der folgenden Aktionen durchführen:

- bereits auf Ihrem Rechner vorhandene GIS Daten verwenden
- Download sample data from https://github.com/qgis/QGIS-Sample-Data/archive/master.zip and unzip the archive on any convenient location on your system.
- QGIS deinstallieren, wieder neu installieren und dabei die entsprechende Option auswählen(nur empfohlen, wenn die oben genannten Lösungen fehlgeschlagen sind).

For GNU/Linux and macOS, there are no dataset installation packages available as rpm, deb or dmg. To use the sample dataset, download it from https://github.com/qgis/QGIS-Sample-Data/archive/master.zip and unzip the archive on any convenient location on your 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 QGIS überwiegend als grafische Benutzeroberfläche für GRASS GIS verwenden möchten, finden Sie weitere GRASS GIS Beispiel-Locations (z.B. Spearfish oder South Dakota) auf der offiziellen GRASS Website unter der URL: https://grass.osgeo.org/download/sample-data/.

5.2 QGIS starten und beenden

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

- 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 *sample 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. Laden Sie die Dateien 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.img` und doppelklicken Sie darauf. Der Landbedeckungslayer wird im Hintergrund geladen während das Fenster Datenquellenverwaltung geöffnet bleibt.
   4. Um die Daten mit den Seen zu laden, navigieren Sie zum Ordner `qgis_sample_data/gml/`, und doppelklicken Sie auf die Datei `lakes.gml`, um sie zu öffnen.
   5. Ein Dialog *Koordinatenbezugssystemsauswahl* öffnet sich. Im Menü *Filter* tippen Sie `2964`, um die darunter befindliche Liste der Koordinatenbezugssysteme zu filtern.
   6. Wählen Sie den Eintrag *NAD27 / Alaska Alberts* aus
   7. Klicken Sie auf *OK*
   8. Schließen Sie das Fenster Datenquellenverwaltung

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 auf 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

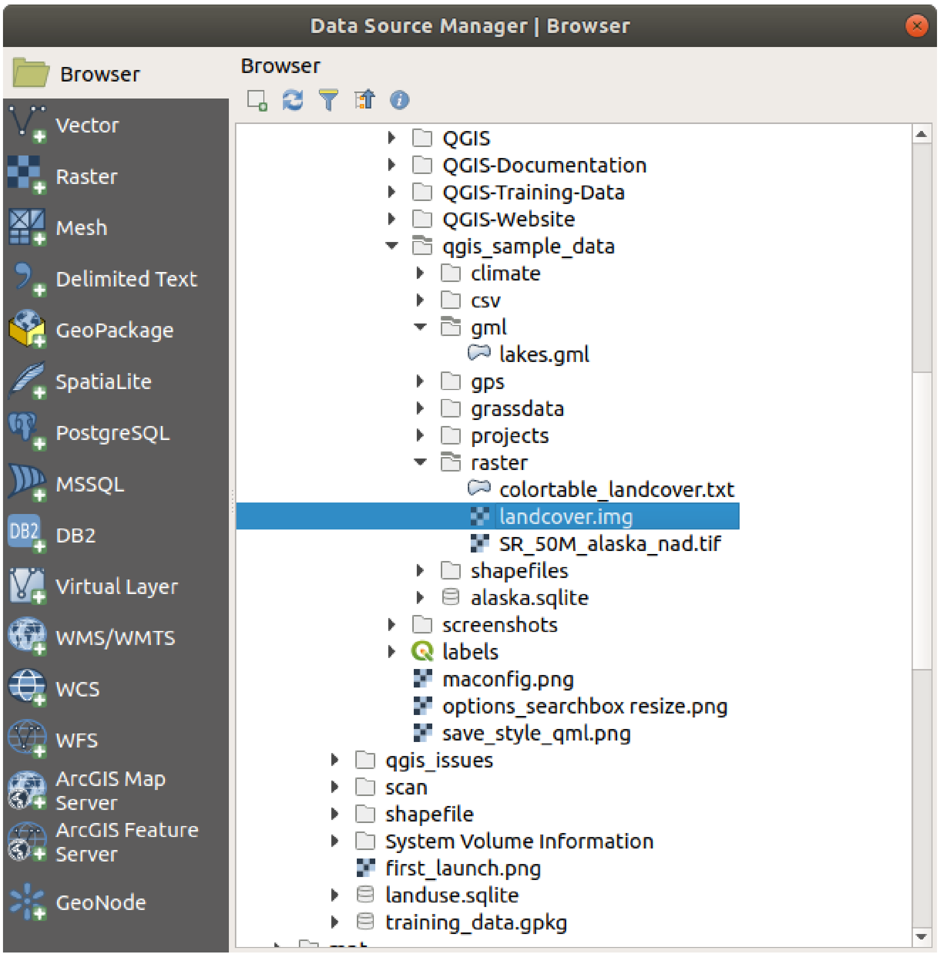


Abb. 5.1: Daten zu einem neuen Projekt in QGIS hinzufügen

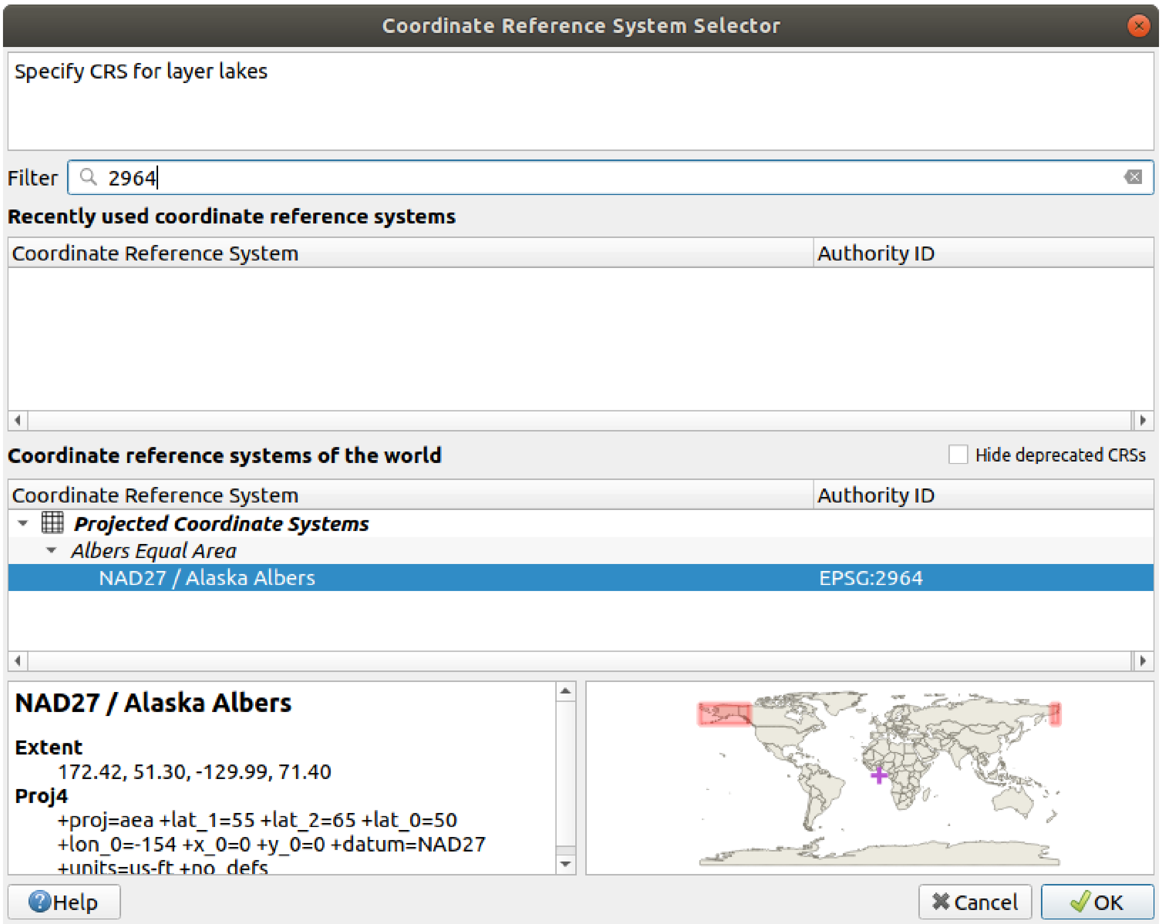


Abb. 5.2: Wählen Sie das Koordinatenbezugssystem der Daten aus

4. So ändern Sie die Farbe der Seen.

    1. Klicken Sie auf den Reiter *Symbolisierung*

    2. Wählen Sie blau als Füllfarbe aus.

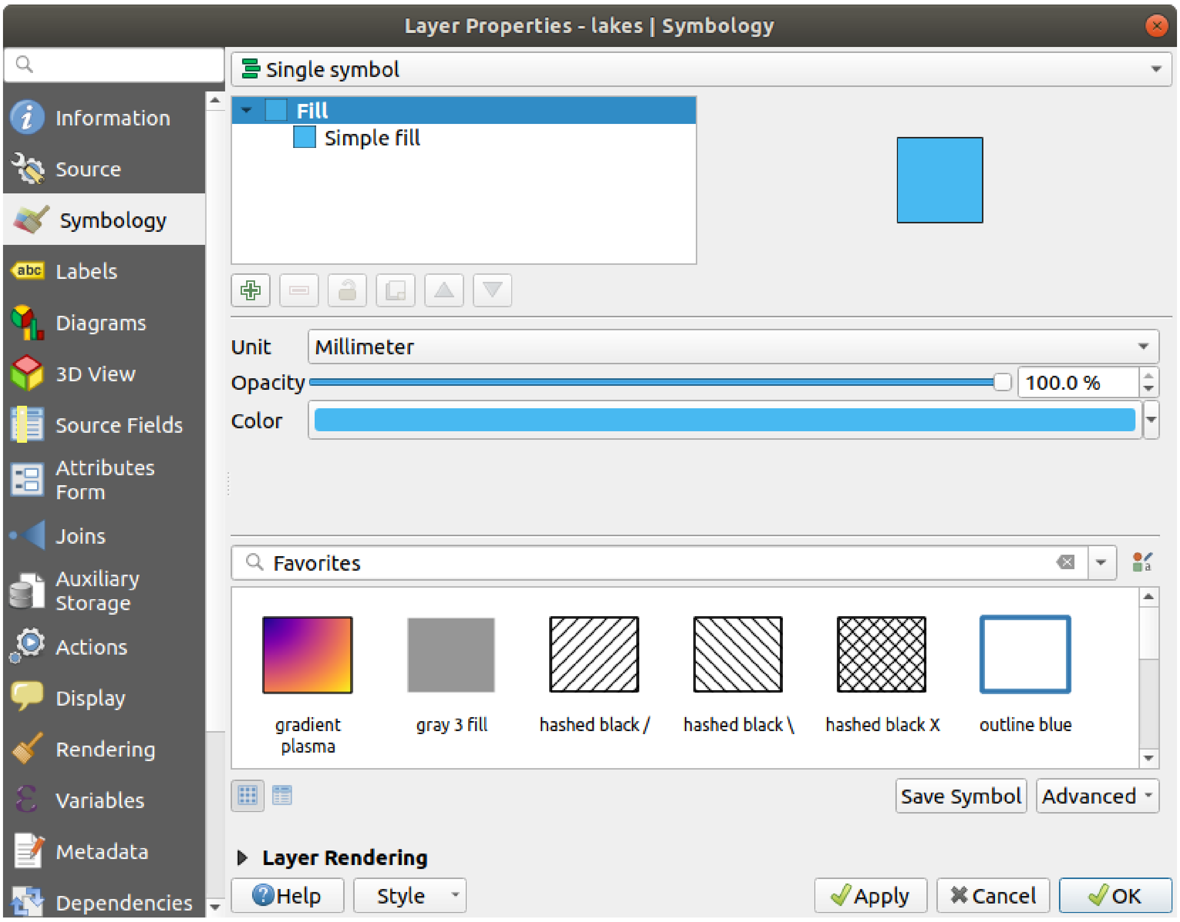


Abb. 5.3: Farbe für die Seen auswählen

    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 *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`.

    5. Drücken Sie auf *Anwenden*. Die Namen werden nun über den Umgriffen dargestellt.

6. Sie können die Lesbarkeit der Beschriftungen verbessern, indem Sie einen weißen Puffer darum herum 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. Finally click *OK* to close the *Layer Properties* dialog and apply the changes.

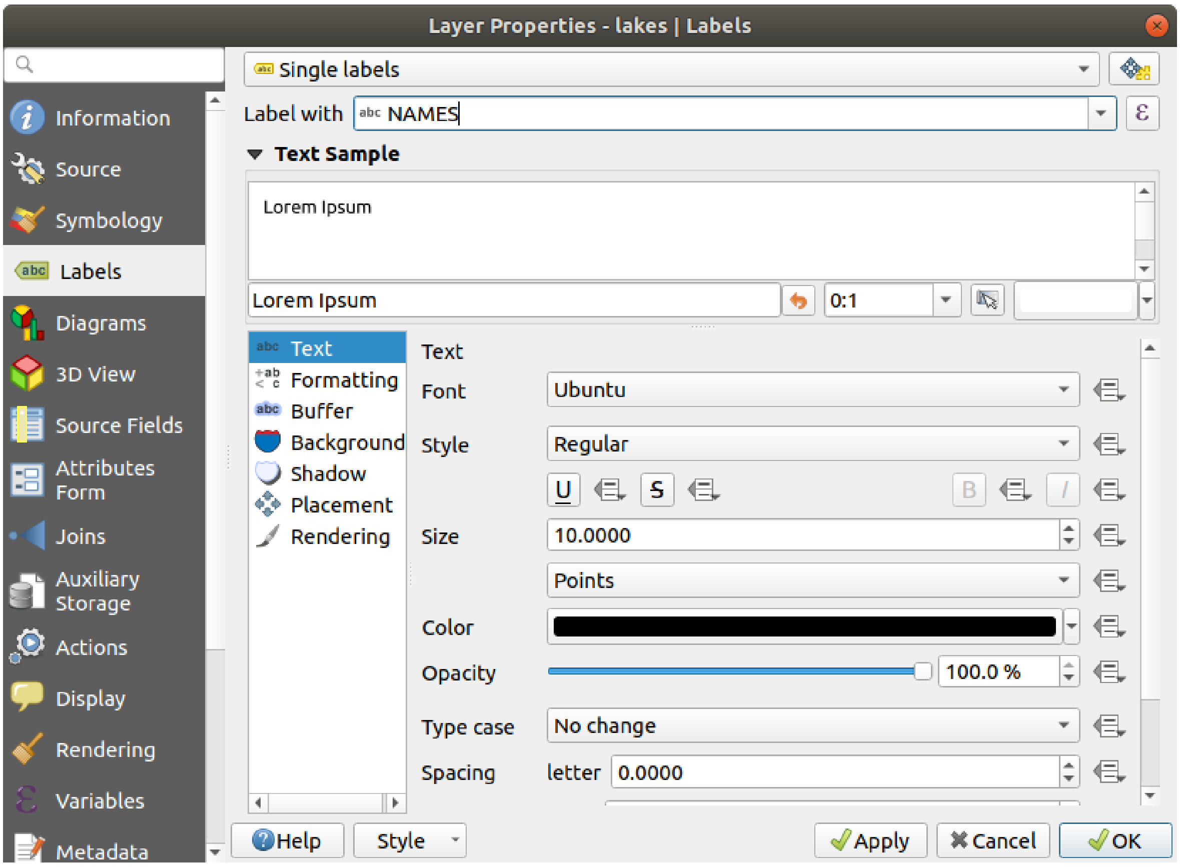


Abb. 5.4: Die Namen der Seen anzeigen

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 menuselection:*Projekt –>* *Speichern unter…*, um Ihre Änderungen als Projektdatei `.qgz` zu speichern.

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 Training manual, um mit schrittweisen Übungen beim Lernen von QGIS weiterzumachen.

KAPITEL 6

Arbeiten mit Projektdateien

6.1 Vorstellung von QGIS-Projekten

The state of your QGIS session is called a project. QGIS works on one project at a time. A settings can be project-specific or an application-wide default for new projects (see section *Optionen*). QGIS can save the state of your workspace into a *QGIS project file* using the menu options *Project ▸ Save* or *Project ▸ Save As…*.

Bemerkung: If the project has been modified the `*` symbol will appear in the title bar and QGIS will, by default, ask you if you would like to save the changes. This behavior is controlled by the *Prompt to save project and data source changes when required* setting under *Settings ▸ Options ▸ General*.

You can load existing projects into QGIS from the Browser panel or by through *Project ▸ Open…*, *Project ▸ New from template* or *Project ▸ Open Recent ▸*.

At startup, a list of *Project Templates* and *Recent Projects* are displayed, including screenshots, names and file paths (for up to ten projects). The *Recent Projects* list is handy to access recently used projects. Double-click an entry to open the project or project template. You can also add a layer to create a new project automatically. The lists will then disappear, giving way to the map canvas.

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

When you open a fresh project, the title bar will show `Untitled Project` until you save it.

In einer Projektdatei sind folgenden Informationen gespeichert:

- Hinzugefügte Layer
- Welche Layer können abgefragt werden
- Layereigenschaften, inklusive Symbologie und Styles
- Projektion für das Kartenfenster
- Zuletzt gewählte Ausdehnung im Kartenfenster
- Drucklayouts

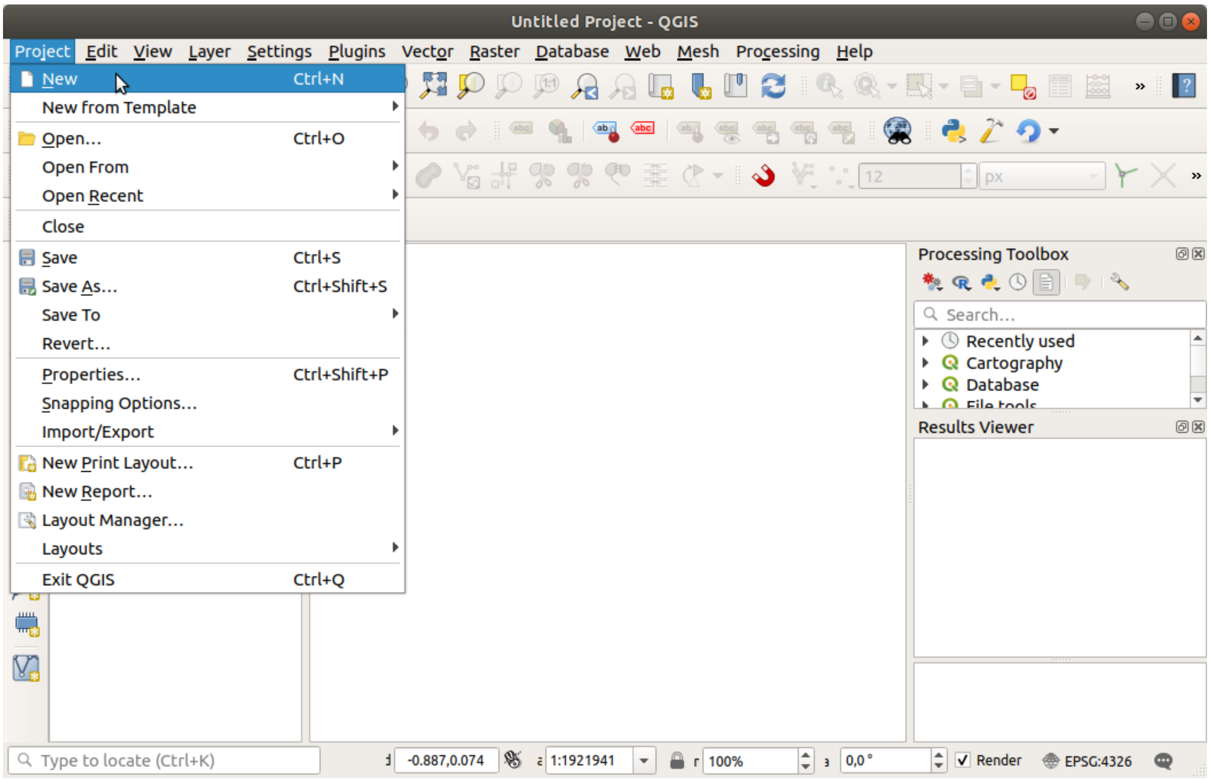


Abb. 6.1: Start eines neuen Projekts in QGIS

- 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

The project file is saved in XML format (see *QGS/QGZ - Das QGIS-Projekt-Dateiformat*). This means that it is possible to edit the file outside of QGIS if you know what you are doing. The project file format has been updated several times. Project files from older QGIS versions may not work properly any more.

Bemerkung: By default, QGIS will warn you of version differences. This behavior is controlled in the *General* tab of *Settings ⍰ Options* (*Warn when opening a project file saved with an older version of QGIS*).

Whenever you save a `.qgs` project file in QGIS, a backup of the file is created in the same directory as the project file, with the extension `.qgs~`.

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: The *Auxiliary Storage Properties* mechanism makes a zipped project particularly useful, since it embeds auxiliary data.

Projekte können auch über die folgenden Menüeinträge des Projekts in eine PostgreSQL-Datenbank gespeichert bzw. geladen werden:

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

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

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

6.2 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, size, … Georeferencing the image is possible. See *Exporting the map view* for more details.
- Exporting to PDF files: *Project Import/Export Export Map to PDF…* outputs the map canvas rendering to PDF at custom scale, resolution, and with some advanced settings (simplification, georeferencing, …). See *Exporting the map view* for more details.
- Exporting to DXF files: *Project Import/Export Export Project to DXF…* opens a dialog where you can define the ‚Symbology mode', the ‚Symbology scale' and vector layers you want to export to DXF. Through the ‚Symbology mode', symbols from the original QGIS Symbology can be exported with high fidelity (see section *Creating new DXF files*).
- Designing maps: *Project New Print Layout…* opens a dialog where you can layout and print the current map canvas (see section *Kartenlayout*).

KAPITEL 7

QGIS GUI

The QGIS graphical user interface (GUI) is shown in the figure below (the numbers 1 through 5 in yellow circles indicate important elements of the QGIS GUI, and are discussed below).

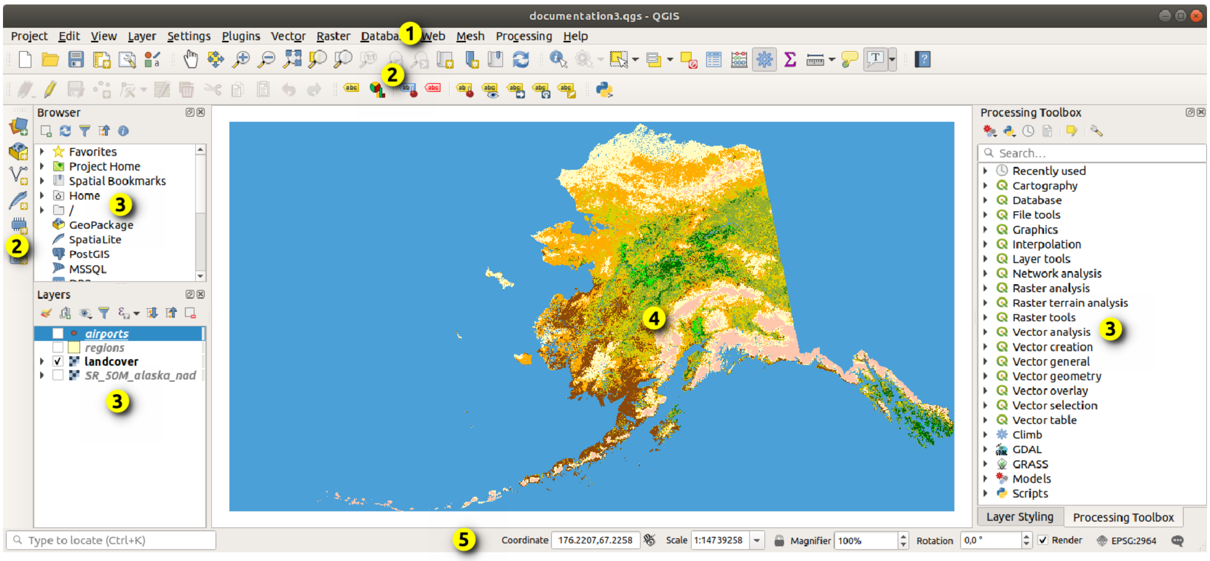


Abb. 7.1: QGIS GUI mit Alaskabeispieldatensatz

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

The main QGIS GUI (Abb. 7.1) consists of five components / component types:

1. *Menu Bar*
2. *Toolbars*
3. *Panels*
4. *Map View*
5. *Status Bar*

Scroll down for detailed explanations of these.

7.1 Menüleiste

The Menu bar provides access to QGIS functions using standard hierarchical menus. The Menus, their options, associated icons and keyboard shortcuts are described below. The keyboard shortcuts can be reconfigured (*Settings ▸ Keyboard Shortcuts*).

Most menu options have a corresponding tool and vice-versa. However, the Menus are not organized exactly like the toolbars. The locations of menu options in the toolbars are indicated below in the table. Plugins may add new options to Menus. For more information about tools and toolbars, see *Werkzeugkästen*.

Bemerkung: QGIS is a cross-platform application. Tools are generally available on all platforms, but they may be placed in different menus, depending on the operating systems. The lists below show the most common locations, including known variations.

7.1.1 Projekt

The *Project* menu provides access and exit points for *project files*. It provides tools to:

- Create a *New* project file from scratch or use another project file as a template (see *Project files options* for template configuration)
- *Open…* a project from a file, a GeoPackage or a PostgreSQL database
- *Close* a project or revert it to its last saved state
- *Save* a project in `.qgs` or `.qgz` file format, either as a file or within a GeoPackage or PostgreSQL database
- Export the map canvas to different formats or use a *print layout* for more complex output
- Set project properties and snapping options for geometry editing.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Neu	`Ctrl+N`	*Projekt*	*Vorstellung von QGIS-Projekten*
Neu aus Vorlage ▸			*Vorstellung von QGIS-Projekten*
Open…	`Strg+O`	*Projekt*	*Vorstellung von QGIS-Projekten*
Open from ▸			
▸ *GeoPackage…*			*Vorstellung von QGIS-Projekten*
▸ *PostgreSQL…*			*Vorstellung von QGIS-Projekten*
Zuletzt verwendetes ▸	`Alt+J` + `R`		*Vorstellung von QGIS-Projekten*
Schließen			*Vorstellung von QGIS-Projekten*
Speichern	`Strg+S`	*Projekt*	*Vorstellung von QGIS-Projekten*
Speichern als…	`Strg+Shift+S`	*Projekt*	*Vorstellung von QGIS-Projekten*
Save to ▸			
▸ *Templates…*			*Vorstellung von QGIS-Projekten*
▸ *GeoPackage…*			*Vorstellung von QGIS-Projekten*
▸ *PostgreSQL…*			*Vorstellung von QGIS-Projekten*
Revert…			
Eigenschaften…	`Strg+Shift+P`		*Projekteigenschaften*
Fangoptionen…			*Einstellen der Fangtoleranz und des Suchradius*
Import/Export ▸			
▸ *Export Map to Image…*			*Exporting the map view*
▸ *Export Map to PDF…*			*Exporting the map view*
▸ *Export Project to DXF…*			*Creating new DXF files*
▸ *Import Layers from DWG/DXF…*			*Importing a DXF or DWG file*
New Print Layout…	`Strg+P`	*Projekt*	*Kartenlayout*
New Report…			*Creating a Report*
Layout-Verwaltung…		*Projekt*	*Kartenlayout*
Layouts ▸			*Kartenlayout*
QGIS beenden	`Strg+Q`		

Under macOS, the *Exit QGIS* command corresponds to *QGIS* ▸ *Quit QGIS* (`Cmd+Q`).

7.1.2 Bearbeiten

The *Edit* menu provides most of the native tools needed to edit layer attributes or geometry (see *Editierfunktionen* for details).

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Rückgängig	`Strg+Z`	*Digitalisierung*	*Rückgängig und Wiederholen*
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*

Fortsetzung auf der nächsten Seite

Tab. 7.1 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Objekte einfügen	Strg+V	*Digitalisierung*	*Objekte ausschneiden, kopieren und einfügen*
Paste Features as			*Mit Attributtabelle arbeiten*
New Vector Layer…			*Mit Attributtabelle arbeiten*
Temporary Scratch Layer…	Ctrl+Alt+V		*Mit Attributtabelle arbeiten*
Auswahl		*Attribute*	*Selecting features*
Select Feature(s)		*Attribute*	*Selecting features*
Select Features by Polygon		*Attribute*	*Selecting features*
Select Features by Freehand		*Attribute*	*Selecting features*
Select Features by Radius		*Attribute*	*Selecting features*
Select Features by Value…	F3	*Attribute*	*Selecting features*
Select Features by Expression…	Ctrl+F3	*Attribute*	*Selecting features*
Deselect Features from All Layers	Strg+Umschalt+A	*Attribute*	*Selecting features*
Reselect Features		*Attribute*	*Selecting features*
Select All Features	Strg+A	*Attribute*	*Selecting features*
Invert Feature Selection		*Attribute*	*Selecting features*
Add Record	Strg+.	*Digitalisierung*	
Add Point Feature	Strg+.	*Digitalisierung*	*Objekte digitalisieren*
Add Line Feature	Strg+.	*Digitalisierung*	*Objekte digitalisieren*
Add Polygon Feature	Strg+.	*Digitalisierung*	*Objekte digitalisieren*
Add Circular String		*Shape Digitizing*	*Kreisbogen hinzufügen*
Add Circular String by Radius		*Shape Digitizing*	*Kreisbogen hinzufügen*
Add Circle		*Shape Digitizing*	*Draw Circles*
Add Circle from 2 Points		*Shape Digitizing*	*Draw Circles*
Add Circle from 3 Points		*Shape Digitizing*	*Draw Circles*
Add Circle from 3 Tangents		*Shape Digitizing*	*Draw Circles*
Add Circle from 2 Tangents and a Point		*Shape Digitizing*	*Draw Circles*
Add Circle by a Center Point and Another Point		*Shape Digitizing*	*Draw Circles*
Add Rectangle		*Shape Digitizing*	*Draw Rectangles*

Fortsetzung auf der nächsten Seite

Tab. 7.1 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
☐ *Add Rectangle from Extent*		*Shape Digitizing*	*Draw Rectangles*
☐ *Add Rectangle from Center and a Point*		*Shape Digitizing*	*Draw Rectangles*
☐ *Add Rectangle from 3 Points (Distance from 2nd and 3rd point)*		*Shape Digitizing*	*Draw Rectangles*
☐ *Add Rectangle from 3 Points (Distance from projected point on segment p1 and p2)*		*Shape Digitizing*	*Draw Rectangles*
Add Regular Polygon ☐		*Shape Digitizing*	*Draw Regular Polygons*
☐ *Add Regular Polygon from Center and a Point*		*Shape Digitizing*	*Draw Regular Polygons*
☐ *Add Regular Polygon from Center and a Corner*		*Shape Digitizing*	*Draw Regular Polygons*
☐ *Add Regular Polygon from 2 Points*		*Shape Digitizing*	*Draw Regular Polygons*
Add Ellipse ☐		*Shape Digitizing*	*Draw Ellipses*
☐ *Add Ellipse from Center and 2 Points*		*Shape Digitizing*	*Draw Ellipses*
☐ *Add Ellipse from Center and a Point*		*Shape Digitizing*	*Draw Ellipses*
☐ *Add Ellipse from Extent*		*Shape Digitizing*	*Draw Ellipses*
☐ *Add Ellipse from Foci*		*Shape Digitizing*	*Draw Ellipses*
Objekt(e) verschieben		*Advanced Digitizing*	*Objekt(e) verschieben*
Copy and Move Feature(s)		*Advanced Digitizing*	*Objekt(e) verschieben*
Delete Selected		*Digitalisierung*	*Ausgewählte Objekte löschen*
Modify Attributes of Selected Features		*Digitalisierung*	*Editiere Attributwerte*
Objekt(e) drehen		*Advanced Digitizing*	*Objekt(e) drehen*
Objekt vereinfachen		*Advanced Digitizing*	*Objekt vereinfachen*
Ring hinzufügen		*Advanced Digitizing*	*Ring hinzufügen*
Teil hinzufügen		*Advanced Digitizing*	*Teil hinzufügen*
Ring füllen		*Advanced Digitizing*	*Ring füllen*

Fortsetzung auf der nächsten Seite

Tab. 7.1 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Ring löschen		*Advanced Digitizing*	*Ring löschen*
Teil löschen		*Advanced Digitizing*	*Teil löschen*
Objekte überarbeiten		*Advanced Digitizing*	*Objekte überarbeiten*
Linie versetzen		*Advanced Digitizing*	*Linie versetzen*
Objekte trennen		*Advanced Digitizing*	*Objekte trennen*
Teile zerlegen		*Advanced Digitizing*	*Teile zerlegen*
Gewählte Objekte verschmelzen		*Advanced Digitizing*	*Gewählte Objekte verschmelzen*
Merge Attributes of Selected Features		*Advanced Digitizing*	*Attribute gewählter Objekte vereinen*
Vertex Tool (All Layers)		*Digitalisierung*	*Knotenwerkzeug*
Vertex Tool (Current Layer)		*Digitalisierung*	*Knotenwerkzeug*
Punktsymbole drehen		*Advanced Digitizing*	*Punktsymbole drehen*
Offset Point Symbols		*Advanced Digitizing*	*Punktsymbolversatz*
Reverse Line		*Advanced Digitizing*	*Reverse Line*
Trim/extend Feature		*Advanced Digitizing*	*Trim/Extend Feature*

Tools that depend on the selected layer geometry type i.e. point, polyline or polygon, are activated accordingly:

Menüleiste	Punkt	Polyline	Polygon
Move Feature(s)			
Copy and Move Feature(s)			

7.1.3 Ansicht

The map is rendered in map views. You can interact with these views using the *View* tools (see *Working with the map canvas* for more information). For example, you can:

- Create new 2D or 3D map views next to the main map canvas
- *Zoom or pan* to any place
- Query displayed features‘ attributes or geometry
- Enhance the map view with preview modes, annotations or decorations
- Access any panel or toolbar

The menu also allows you to reorganize the QGIS interface itself using actions like:

- *Toggle Full Screen Mode*: covers the whole screen while hiding the title bar
- *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
- *Toggle Map Only*: hides panels, toolbars, menus and status bar and only shows the map canvas. Combined with the full screen option, it makes your screen display only the map

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
New Map View	`Ctrl+M`		*Kartenfenster*
New 3D Map View	`Strg+Alt+M`		*3D-Ansicht*
Karte verschieben		*Map Navigation*	*Zoomen und Karte verschieben*
Karte zur Auswahl verschieben		*Map Navigation*	
Hineinzoomen	`Ctrl+Alt++`	*Map Navigation*	*Zoomen und Karte verschieben*
Herauszoomen	`Ctrl+Alt+-`	*Map Navigation*	*Zoomen und Karte verschieben*
Objekte abfragen	`Strg+Shift+I`	*Attribute*	*Identifying Features*
Messen ⍰		*Attribute*	*Messen*
⍰ *Measure Line*	`Ctrl+Shift+M`	*Attribute*	*Messen*
⍰ *Measure Area*	`Ctrl+Shift+J`	*Attribute*	*Messen*
⍰ *Measure Angle*		*Attribute*	*Messen*
Statistische Zusammenfassung		*Attribute*	*Statistical Summary Panel*
Volle Ausdehnung	`Strg+Shift+F`	*Map Navigation*	
Zur Auswahl zoomen	`Strg+J`	*Map Navigation*	
Auf den Layer zoomen		*Map Navigation*	
Zoom To Native Resolution (100%)		*Map Navigation*	
Zoom zurück		*Map Navigation*	
Zoom vor		*Map Navigation*	
Dekorationen ⍰	`Alt+V` + `D`		*Dekorationen*
⍰ *Grid…*			*Gitter*
⍰ *Scale Bar…*			*Maßstab*
⍰ *Image…*			*Image Decoration*
⍰ *North Arrow…*			*Nordpfeil*

Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
⍰ *Title Label…*			*Title Label*
⍰ *Copyright Label…*			*Copyright Label*
⍰ *Layout Extents…*			*Layout Extents*
Vorschaumodus ⍰			
⍰ *Normal*			
⍰ *Simulate Photocopy (Grayscale)*			
⍰ *Simulate Fax (Mono)*			
⍰ *Simulate Color Blindness (Protanope)*			
⍰ *Simulate Color Blindness (Deuteronope)*			
Show Map Tips		*Attribute*	*Display Properties*
New Spatial Bookmark…	`Strg+B`	*Map Navigation*	*Räumliche Lesezeichen*
Show Spatial Bookmarks	`Strg+Shift+B`	*Map Navigation*	*Räumliche Lesezeichen*
Show Spatial Bookmark Manager			*Räumliche Lesezeichen*
Refresh	`F5`	*Map Navigation*	
Alle Layer anzeigen	`Strg+Shift+U`		*Layerfenster*
Alle Layer ausblenden	`Strg+Shift+H`		*Layerfenster*
Show Selected Layers			*Layerfenster*
Hide Selected Layers			*Layerfenster*
Hide Deselected Layers			*Layerfenster*
Bedienfelder ⍰			*Bedienfelder und Werkzeugkästen*
⍰ *Advanced Digitizing*			*Das Bedienfeld Erweiterte Digitalisierung*
⍰ *Browser*			*The Browser Panel*
⍰ *Browser (2)*			*The Browser Panel*
⍰ *GPS Information*			*Live GPS tracking*
⍰ *GRASS Tools*			*GRASS GIS Integration*
⍰ *Layer Order*			*Layer Order Panel*
⍰ *Layer Styling*			*Layer Styling Panel*
⍰ *Layers*			*Layerfenster*
⍰ *Log Messages*			*Log Messages Panel*
⍰ *Overview*			*Overview Panel*
⍰ *Processing Toolbox*			*The Toolbox*
⍰ *Results Viewer*			*The Toolbox*
⍰ *Snapping and Digitizing Options*			*Einstellen der Fangtoleranz und des Suchradius*
⍰ *Spatial Bookmark Manager*			*Räumliche Lesezeichen*
⍰ *Statistics*			*Statistical Summary Panel*
⍰ *Tile Scale*			*Tilesets*
⍰ *Undo/Redo*			*Undo/Redo Panel*
Werkzeugkästen ⍰			*Bedienfelder und Werkzeugkästen*

Fortsetzung auf der nächsten Seite

Tab. 7.2 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
⍰ *Advanced Digitizing Toolbar*			*Erweiterte Digitalisierung*
⍰ *Attributes Toolbar*			
⍰ *Data Source Manager Toolbar*			*Verwaltung von Datenquellen*
⍰ *Database Toolbar*			
⍰ *Digitizing Toolbar*			*Einen vorhandenen Layer editieren*
⍰ *Help Toolbar*			
⍰ *Label Toolbar*			*The Label Toolbar*
⍰ *Manage Layers Toolbar*			*Verwaltung von Datenquellen*
⍰ *Map Navigation Toolbar*			
⍰ *Plugins Toolbar*			*Plugins*
⍰ *Project Toolbar*			
⍰ *Raster Toolbar*			
⍰ *Shape Digitizing Toolbar*			*Formen digitalisieren*
⍰ *Snapping Toolbar*			*Einstellen der Fangtoleranz und des Suchradius*
⍰ *Vector Toolbar*			
⍰ *Web Toolbar*			
⍰ *GRASS*			*GRASS GIS Integration*
Volle Ausdehnung	`F11`		
Toggle Panel Visibility	`Ctrl+Tab`		
Toggle Map Only	`Ctrl+Shift+Tab`		

Under Linux KDE, *Panels* ⍰, *Toolbars* ⍰ and *Toggle Full Screen Mode* are in the *Settings* menu.

7.1.4 Layer

The *Layer* menu provides a large set of tools to *create* new data sources, *add* them to a project or *save modifications* to them. Using the same data sources, you can also:

- *Duplicate* a layer to generate a copy where you can modify the name, style (symbology, labels, …), joins, … The copy uses the same data source as the original.
- *Copy* and *Paste* layers or groups from one project to another as a new instance whose properties can be modified independently. As for *Duplicate*, the layers are still based on the same data source.
- or *Embed Layers and Groups…* from another project, as read-only copies which you cannot modify (see *Layer/Gruppen einbinden*)

The *Layer* menu also contains tools to configure, copy or paste layer properties (style, scale, CRS…).

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Data Source Manager	`Ctrl+L`	*Data Source Manager*	*Opening Data*
Layer erzeugen ⍰			*Neue Vektorlayer erstellen*
⍰ *New GeoPackage Layer…*	`Ctrl+Shift+N`	*Data Source Manager*	*Einen neuen GeoPackage-Layer erstellen*
⍰ *New Shapefile Layer…*		*Data Source Manager*	*Einen neuen Shapedatei-Layer erstellen*

Fortsetzung auf der nächsten Seite

Tab. 7.3 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
? *New SpatiaLite Layer…*		*Data Source Manager*	*Creating a new SpatiaLite layer*
? *New Temporary Scratch Layer…*		*Data Source Manager*	*Creating a new Temporary Scratch Layer*
? *New Virtual Layer…*		*Data Source Manager*	*Creating virtual layers*
Layer hinzufügen ?			*Öffnen von Daten*
? *Add Vector Layer……*	`Strg+Shift+V`	*Manage Layers*	*Loading a layer from a file*
? *Add Raster Layer…*	`Ctrl+Shift+R`	*Manage Layers*	*Loading a layer from a file*
? *Add Mesh Layer…*		*Manage Layers*	*Loading a mesh layer*
? *Add Delimited Text Layer…*	`Ctrl+Shift+T`	*Manage Layers*	*Importing a delimited text file*
? *Add PostGIS Layer…*	`Ctrl+Shift+D`	*Manage Layers*	*Database related tools*
? *Add SpatiaLite Layer…*	`Ctrl+Shift+L`	*Manage Layers*	*SpatiaLite Layers*
? *Add MSSQL Spatial Layer…*		*Manage Layers*	*Database related tools*
? *Add Oracle Spatial Layer…*		*Manage Layers*	*Database related tools*
? *Add DB2 Spatial Layer…*	`Ctrl+Shift+2`	*Manage Layers*	*Database related tools*
? *Add/Edit Virtual Layer…*		*Manage Layers*	*Creating virtual layers*
? *Add WMS/WMTS Layer…*	`Ctrl+Shift+W`	*Manage Layers*	*WMS/WMTS Layer laden*
? *Add ArcGIS MapServer Layer…*		*Manage Layers*	
? *Add WCS Layer…*		*Manage Layers*	*WCS Client*
? *Add WFS Layer…*		*Manage Layers*	*WFS und WFS-T Klient*
? *Add ArcGIS FeatureServer Layer…*		*Manage Layers*	
Eingebettete Layer und Gruppen…			*Layer/Gruppen einbinden*
Aus Layerdefinitionsdatei hinzufügen…			*Layer definition file*
Copy Style			*Save and Share Layer Properties*
Paste Style			*Save and Share Layer Properties*
Copy Layer			
Paste Layer/Group			
Attributtabelle öffnen	`F6`	*Attribute*	*Mit Attributtabelle arbeiten*
Bearbeitungsstatus umschalten		*Digitalisierung*	*Einen vorhandenen Layer editieren*
Layeränderungen speichern		*Digitalisierung*	*Änderungen speichern*
Aktuelle Änderungen ?		*Digitalisierung*	*Änderungen speichern*
? *Save for Selected Layer(s)*		*Digitalisierung*	*Änderungen speichern*
? *Rollback for Selected Layer(s)*		*Digitalisierung*	*Änderungen speichern*
? *Cancel for Selected Layer(s)*		*Digitalisierung*	*Änderungen speichern*

Fortsetzung auf der nächsten Seite

Tab. 7.3 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Save for all Layers		*Digitalisierung*	*Änderungen speichern*
Rollback for all Layers		*Digitalisierung*	*Änderungen speichern*
Cancel for all Layers		*Digitalisierung*	*Änderungen speichern*
Save As…			*Creating new layers from an existing layer*
Save As Layer Definition File…			*Layer definition file*
Layer/Gruppe löschen	`Strg+D`		
Duplicate Layer(s)			
Set Scale Visibility of Layer(s)			
KBS von Layer(n) setzen	`Strg+Shift+C`		*Layer Coordinate Reference Systems*
Set Project CRS from Layer			*Project Coordinate Reference Systems*
Layer Properties…			*Vektorlayereigenschaften*, *Dialogfenster Rasterlayereigenschaften*, *Mesh Dataset Properties*
Filter…	`Ctrl+F`		*Abfrageeditor*
Beschriftung			*Labels Properties*
Show in Overview			*Overview Panel*
Show All in Overview			*Overview Panel*
Hide All from Overview			*Overview Panel*

7.1.5 Einstellungen

Menüleiste	Referenz
User Profiles	*Working with User Profiles*
default	*Working with User Profiles*
Open Active Profile Folder	*Working with User Profiles*
New Profile…	*Working with User Profiles*
Style Manager…	*The Style Manager*
Custom Projections…	*Eigenes Koordinatenbezugssystem definieren*
Keyboard Shortcuts…	*Tastenkürzel*
Interface Customization…	*Anpassung*
Optionen…	*Optionen*

Under Linux KDE, you'll find more tools in the *Settings* menu such as *Panels*, *Toolbars* and *Toggle Full Screen Mode*.

7.1.6 Erweiterungen

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Erweiterungen verwalten und installieren…			*Der Erweiterungen Dialog*
„ *Python Console*	`Strg+Alt+P`	*Plugins*	*QGIS Python Konsole*

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

7.1.7 Vektor

This is what the *Vector* menu looks like if all core plugins are enabled.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Coordinate Capture		*Vector*	*Coordinate Capture Plugin*
Check Geometries…			*Geometry Checker Plugin*
GPS Tools	`Alt+O` + `G`	*Vector*	*GPS Plugin*
Topology Checker		*Vector*	*Topology Checker Plugin*
Geoprocessing Tools ⍰	`Alt+O` + `G`		
⍰ Buffer…			*Buffer*
⍰ Clip…			*Clip*
⍰ Convex Hull…			*Convex hull*
⍰ Difference…			*Difference*
⍰ Dissolve…			*Dissolve*
⍰ Intersection…			*Intersection*
⍰ Symmetrical Difference…			*Symmetrical difference*
⍰ Union…			*Union*
⍰ Eliminate Selected Polygons…			*Eliminate selected polygons*
Geometry Tools ⍰	`Alt+O` + `E`		
⍰ Centroids…			*Centroids*
⍰ Collect Geometries…			*Collect geometries*
⍰ Extract Vertices…			*Extract vertices*
⍰ Multipart to Singleparts…			*Multipart to singleparts*
⍰ Polygons to Lines…			*Polygons to lines*
⍰ Simplify…			*Simplify*
⍰ Check Validity…			*Check validity*
⍰ Delaunay Triangulation…			*Delaunay triangulation*
⍰ Densify by Count…			*Densify by count*
⍰ Add Geometry Attributes…			*Add geometry attributes*
⍰ Lines to Polygons…			*Lines to polygons*
⍰ Voronoi Polygons…			*Voronoi polygons*
Analysis Tools ⍰	`Alt+O` + `A`		
⍰ Line Intersection…			*Line intersections*
⍰ Mean Coordinate(s)…			*Mean coordinate(s)*
⍰ Basic Statistics for Fields…			*Grundstatistik für Felder*
⍰ Count Points in Polygon…			*Count points in polygon*
⍰ Distance Matrix…			*Distance matrix*
⍰ List Unique Values…			*List unique values*
⍰ Nearest Neighbour Analysis…			*Nearest neighbour analysis*
⍰ Sum Line Lengths…			*Sum line lengths*

Fortsetzung auf der nächsten Seite

Tab. 7.4 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Data Management Tools ⍰	`Alt+O` + `D`		
⍰ Merge Vector Layers…			*Merge vector layers*
⍰ Reproject Layer…			*Reproject layer*
⍰ Create Spatial Index…			*Create spatial index*
⍰ Join Attributes by Location…			*Join attributes by location*
⍰ Split Vector Layer…			*Split vector layer*
Research Tools ⍰	`Alt+O` + `R`		
⍰ Select by Location…			*Auswahl nach der Lage*
⍰ Extract Layer Extent…			*Extract layer extent*
⍰ Random Points in Extent…			*Random points in extent*
⍰ Random Points in Layer Bounds…			*Random points in layer bounds*
⍰ Random Points Inside Polygons…			*Random points inside polygons*
⍰ Random Selection…			*Random selection*
⍰ Random Selection Within Subsets…			*Random selection within subsets*
⍰ Regular Points…			*Regular points*

By default, QGIS adds *Processing* algorithms to the *Vector* menu, grouped by sub-menus. This provides shortcuts for many common vector-based GIS tasks from different providers. If not all these sub-menus are available, enable the Processing plugin in *Plugins ⍰ Manage and Install Plugins…*.

Note that the list of the *Vector* menu tools can be extended with any Processing algorithms or some external *plugins*.

7.1.8 Raster

This is what the *Raster* menu looks like if all core plugins are enabled.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Raster calculator…			*Rasterrechner*
Raster ausrichten…			*Raster Ausrichtung*
Georeferencer	`Alt+R` + `G`	*Raster*	*Plugin „Georeferenzierung“*
Analysis ⍰			
⍰ Aspect…			*Aspect*
⍰ Fill nodata…			*Fill nodata*
⍰ Grid (Moving Average)…			*Grid (Moving average)*
⍰ Grid (Data Metrics)…			*Grid (Data metrics)*
⍰ Grid (Inverse Distance to a Power)…			*Grid (Inverse distance to a power)*
⍰ Grid (Nearest Neighbor)…			*Grid (IDW with nearest neighbor searching)*
⍰ Hillshade…			*Hillshade*
⍰ Proximity (Raster Distance)…			*Proximity (raster distance)*
⍰ Roughness…			*Roughness*
⍰ Sieve…			*Sieve*
⍰ Slope…			*Slope*
⍰ Topographic Position Index (TPI)…			*Topographic Position Index (TPI)*
⍰ Terrain Ruggedness Index (TRI)…			*Terrain Ruggedness Index (TRI)*
Projections ⍰			
⍰ Assign Projection…			*Projektion zuweisen*
⍰ Extract Projection…			*Projektion extrahieren*
⍰ Warp (Reproject)…			*Warp (reproject)*
Miscellaneous ⍰			
⍰ Build Virtual Raster…			*Build virtual raster*
⍰ Raster Information…			*Raster information*

Fortsetzung auf der nächsten Seite

Tab. 7.5 – Fortsetzung der vorherigen Seite

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
⍰ *Merge…*			*Merge*
⍰ *Build Overviews (Pyramids)…*			*Build overviews (pyramids)*
⍰ *Tile Index…*			*Tile index*
Extraction ⍰			
⍰ *Clip Raster by Extent…*			*Clip raster by extent*
⍰ *Clip Raster by Mask Layer…*			*Clip raster by mask layer*
⍰ *Contour…*			*Contour*
Conversion ⍰			
⍰ *PCT to RGB…*			*PCT to RGB*
⍰ *Polygonize (Raster to Vector)…*			*Polygonize (raster to vector)*
⍰ *Rasterize (Vector to Raster)…*			*Rasterize (vector to raster)*
⍰ *RGB to PCT…*			*RGB to PCT*
⍰ *Translate (Convert Format)…*			*Translate (convert format)*

By default, QGIS adds *Processing* algorithms to the *Raster* menu, grouped by sub-menus. This provides a shortcut for many common raster-based GIS tasks from different providers. If not all these sub-menus are available, enable the Processing plugin in *Plugins* ⍰ *Manage and Install Plugins…*.

Note that the list of the *Raster* menu tools can be extended with any Processing algorithms or some external *plugins*.

7.1.9 Datenbank

This is what the *Database* menu looks like if all the core plugins are enabled. If no database plugins are enabled, there will be no *Database* menu.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
eVis ⍰	`Alt+D + E`		*eVis Plugin*
⍰ *eVis Database Connection*		*Datenbank*	*eVis Plugin*
⍰ *eVis Event Id Tool*		*Datenbank*	*eVis Plugin*
⍰ *eVis Event Browser*		*Datenbank*	*eVis Plugin*
Offline editing…	`Alt+D + O`		*Offline Editing Plugin*
⍰ *Convert to Offline Project…*		*Datenbank*	*Offline Editing Plugin*
⍰ *Synchronize*		*Datenbank*	*Offline Editing Plugin*
DB Manager…		*Datenbank*	*DB Manager Plugin*

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

7.1.10 Web

This is what the *Web* menu looks like if all the core plugins are enabled. If no web plugins are enabled, there will be no *Web* menu.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
MetaSearch ⍰	`Alt+W + M`		*MetaSearch Catalog Client*
⍰ *Metasearch*		*Web*	*MetaSearch Catalog Client*
⍰ *Help*			*MetaSearch Catalog Client*

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

7.1.11 Mesh

The *Mesh* menu provides tools needed to manipulate *mesh layers*.

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Mesh Calculator…			

7.1.12 Verarbeitung

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Toolbox	Ctrl+Alt+T		*The Toolbox*
Grafische Modellierung…	Ctrl+Alt+G		*Die Grafische Modellierung*
History…	Ctrl+Alt+H		*Das Protokoll*
Results Viewer	Ctrl+Alt+R		*Konfiguration externer Anwendungen*
Edit Features In-Place			*Mit Verarbeitung Objekte innerhalb eines Layers verändern*

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

7.1.13 Hilfe

Menüleiste	Tastenkürzel	Werkzeugleiste	Referenz
Hilfe-Übersicht	F1	*Direkthilfe*	
API Dokumentation			
Plugins ▯			
Ein Problem melden			
Brauchen Sie professionelle Unterstützung?			
QGIS-Homepage	Strg+H		
Check QGIS Version			
About			
QGIS Sustaining Members			

7.1.14 QGIS

This menu is only available under macOS and contains some OS related commands.

Menüleiste	Tastenkürzel
Preferences	
Über QGIS	
QGIS verbergen	
Show All	
Andere verbergen	
QGIS beenden	Cmd+Q

Preferences correspond to *Settings* ▯ *Options*, *About QGIS* corresponds to *Help* ▯ *About* and *Quit QGIS* corresponds to *Project* ▯ *Exit QGIS* for other platforms.

7.2 Bedienfelder und Werkzeugkästen

From the *View* menu (or *Settings*), you can switch QGIS widgets (*Panels* ▸) and toolbars (*Toolbars* ▸) on and off. To (de)activate any of them, right-click the menu bar or toolbar and choose the item you want. Panels and toolbars can be moved and placed wherever you like within the QGIS interface. The list can also be extended with the activation of *Core or external plugins*.

7.2.1 Werkzeugkästen

The toolbars provide access to most of the functions in the menus, plus additional tools for interacting with the map. Each toolbar item has pop-up help available. Hover your mouse over the item and a short description of the tool's purpose will be displayed.

Jede Werkzeugleiste kann nach eigenen Wünschen verschoben werden und kann auch an bzw. ausgeschaltet werden, indem Sie mit der Maus in einen freien Bereich der Werkzeugleiste fahren und auf den rechten Mausknopf drücken.

Tipp: Werkzeugleiste wiederherstellen

If you have accidentally hidden a toolbar, you can get it back using *View* ▸ *Toolbars* ▸ (or *Settings* ▸ *Toolbars* ▸). If, for some reason, a toolbar (or any other widget) totally disappears from the interface, you'll find tips to get it back at *restoring initial GUI*.

7.2.2 Bedienfelder

QGIS provides many panels. Panels are special widgets that you can interact with (selecting options, checking boxes, filling values…) to perform more complex tasks.

Below is a list of the default panels provided by QGIS:

- das *Erweiterte Digitalisierungsfenster*
- the *Browser Panel*
- das *GPS Informationsfenster*
- das *Identifizierenfenster*
- the *Layer Order Panel*
- the *Layer Styling Panel*
- the *Layers Panel*
- the *Log Messages Panel*
- the *Overview Panel*
- the *Processing Toolbox*
- the *Result Viewer Panel*
- the *Spatial Bookmark Manager Panel*
- the *Statistics Panel*
- das *Kachelmaßstabsfenster*
- the *Undo/Redo Panel*

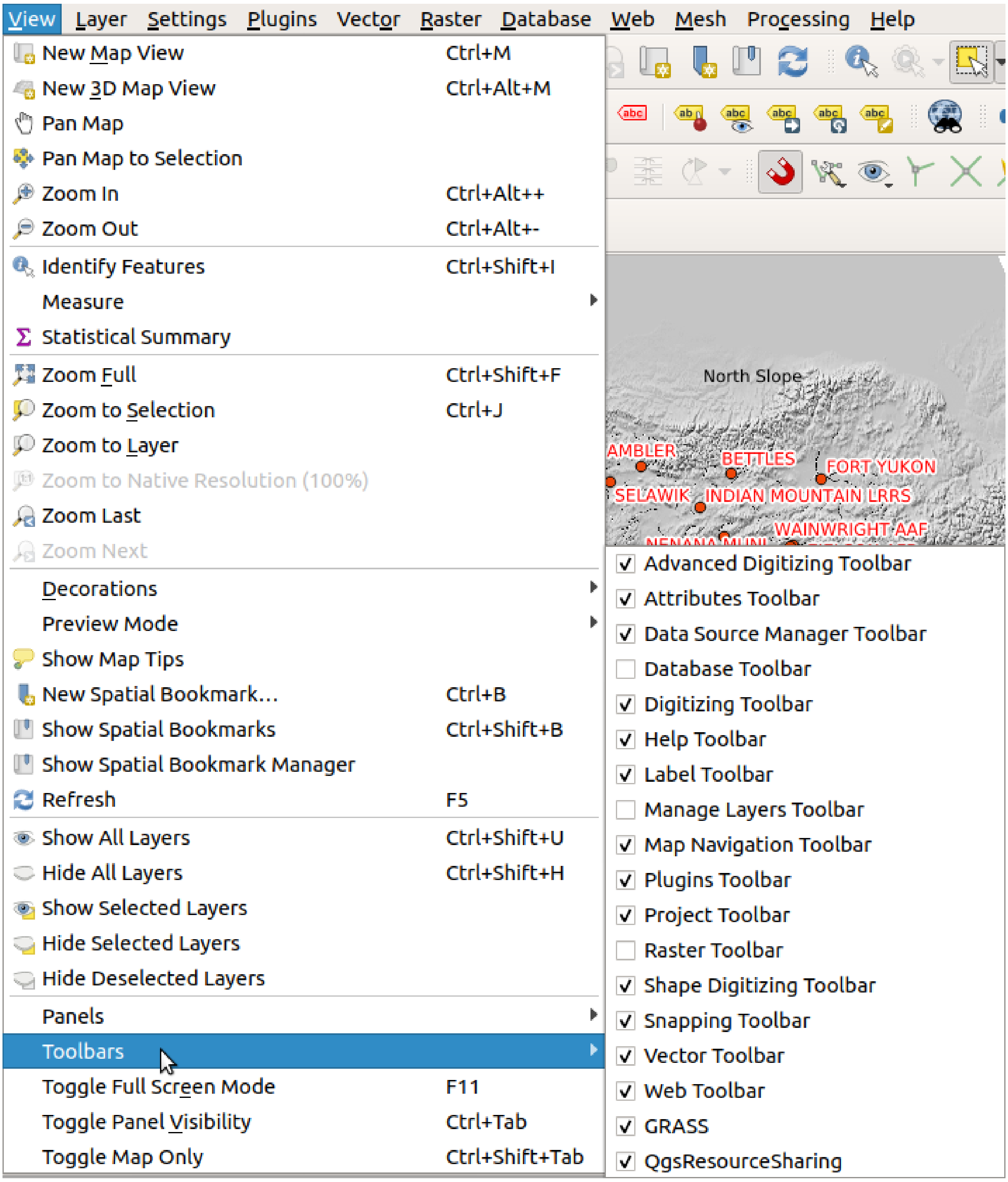


Abb. 7.2: Das Werkzeugkästen Menü

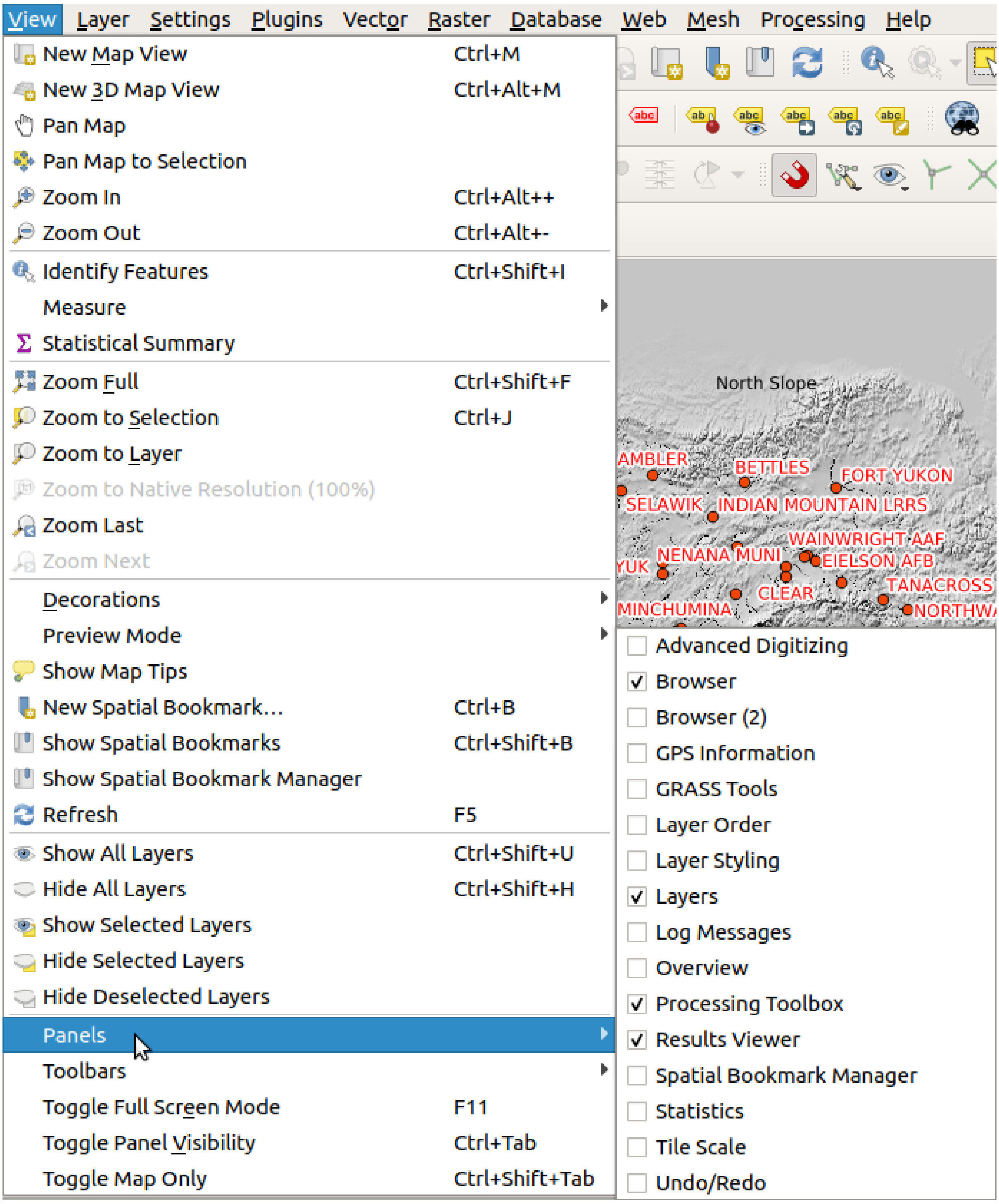


Abb. 7.3: Das Bedienfelder Menü

7.3 Kartenfenster

7.3.1 Exploring the map view

The map view (also called **Map canvas**) is the „business end" of QGIS — maps are displayed in this area. The map displayed in this window will reflect the rendering (symbology, labeling, visibilities…) you applied to the layers you have loaded. It also depends on the layers and the project's Coordinate Reference System (CRS).

When you add a layer (see e.g. *Öffnen von Daten*), QGIS automatically looks for its CRS. If a different CRS is set by default for the project (see *Project Coordinate Reference Systems*) then the layer extent is „on-the-fly" translated to that CRS, and the map view is zoomed to that extent if you start with a blank QGIS project. If there are already layers in the project, no map canvas resize is performed, so only features falling within the current map canvas extent will be visible.

While the focus is over the map view:

- it can be panned, shifting the display to another region of the map: this is performed using the Pan Map tool, the arrow keys, moving the mouse while any of the `Space` key, the middle mouse button or the mouse wheel is held down.
- it can be zoomed in and out, with the dedicated Zoom In and Zoom Out tools. This is also performed by rolling the wheel forward to zoom in and backwards to zoom out. The zoom is centered on the mouse cursor position.

  You can customize the *Zoom factor* under the *Settings ▸ Options ▸ Map tools* menu.
- it can be zoomed to the full extent of loaded layers (Zoom Full), to a layer extent (Zoom to Layer) or to the extent of selected features (Zoom to Selection)
- you can navigate back/forward through canvas view history with the Zoom Last and Zoom Next buttons or using the back/forward mouse buttons.

By default, QGIS opens a single map view (so called „main map"), which is tightly bound to the *Layers* panel; the main map *automatically* reflects the changes you make in the *Layers* panel area. But it's possible to have additional map views to preview different renderings of your dataset, side by side; while still relying on the layers properties as set in the *Layers* panel, each map view can display a different set of layers at different scale and extent.

7.3.2 Setting additional map views

To add a new map view, go to *View ▸ New Map View*. A new floating widget displaying the layers 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.

At the top of an additional map canvas, there's a toolbar with following capabilities:

- Zoom Full, Zoom to Selection and Zoom to Layer to navigate within the view
- Set View Theme to select the *map theme* to display in the map view. If set to `(none)`, the view will follow the *Layers* panel changes.
- View settings to configure the map view:
  - *Synchronize view center with main map*: syncs the center of the map views without changing the scale. This allows you to have an overview style or magnified map which follows the main canvas center.
  - *Synchronize view to selection*: same as zoom to selection
  - *Scale*
  - *Rotation*

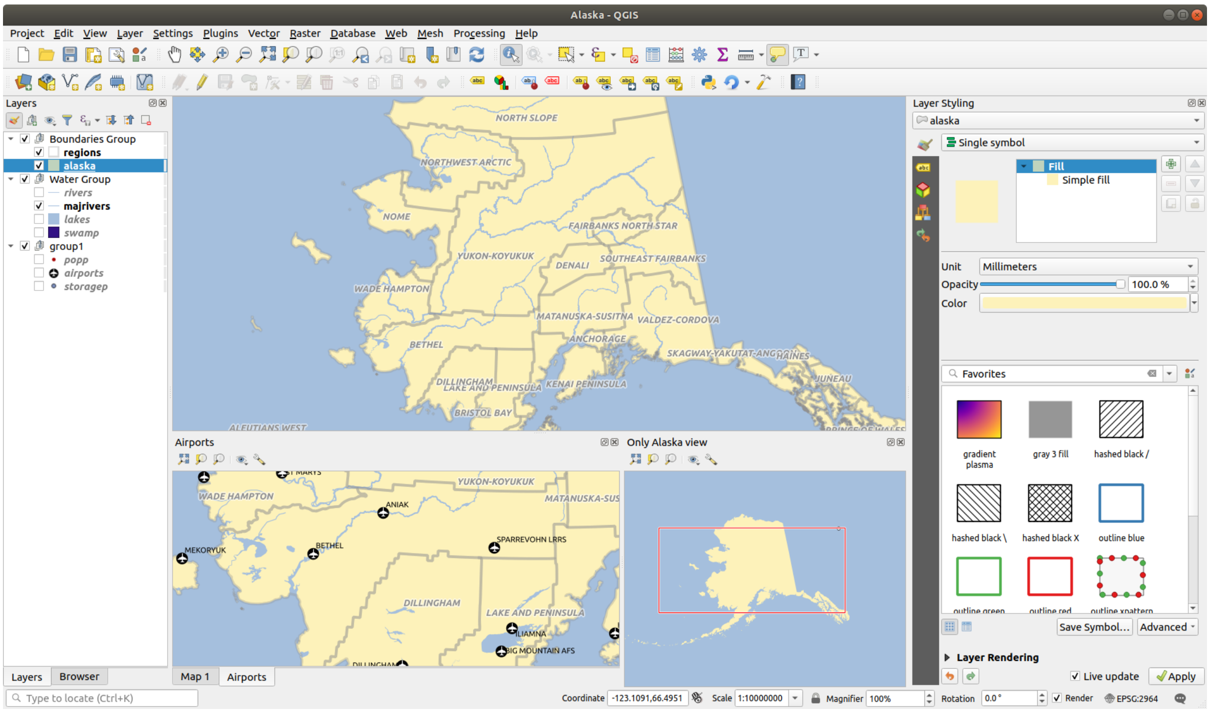


Abb. 7.4: Multiple map views with different settings

- *Magnification*
- *Synchronize scale* with the main map scale. A *Scale factor* can then be applied, allowing you to have a view which is e.g. always 2x the scale of the main canvas.
- *Show annotations*
- *Show cursor position*
- *Show main canvas extent*
- *Show labels*: allows to hide labels regardless they are set in the displayed layers' properties
- *Change map CRS…*
- *Rename view…*

7.3.3 Exporting the map view

Maps you make can be layout and exported to various formats using the advanced capabilities of the *print layout or report*. It's also possible to directly export the current rendering, without a layout. This quick „screenshot" of the map view has some convenient features.

To export the map canvas with the current rendering:

1. Go to *Project ▯ Import/Export*
2. Depending on your output format, select either
   - *Export Map to Image…*
   - or *Export Map to PDF…*

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

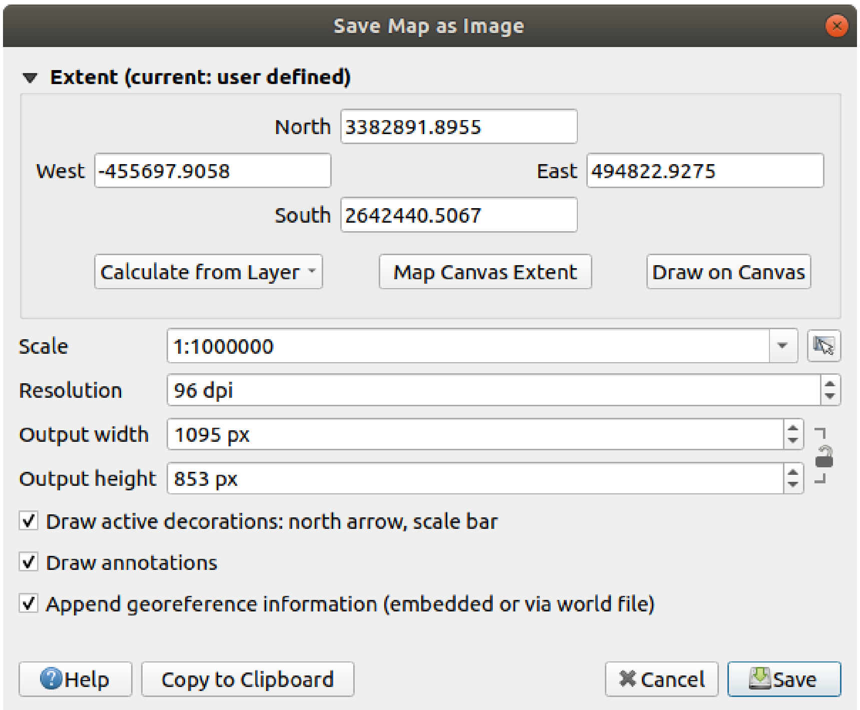


Abb. 7.5: The Save Map as Image dialog

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:
   - *Export RDF metadata* of the document such as the title, author, date, description…
   - *Create Geospatial PDF (GeoPDF)*: Generate a georeferenced PDF file (requires GDAL version 3 or later). 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: Since QGIS 3.10, with GDAL 3 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/.

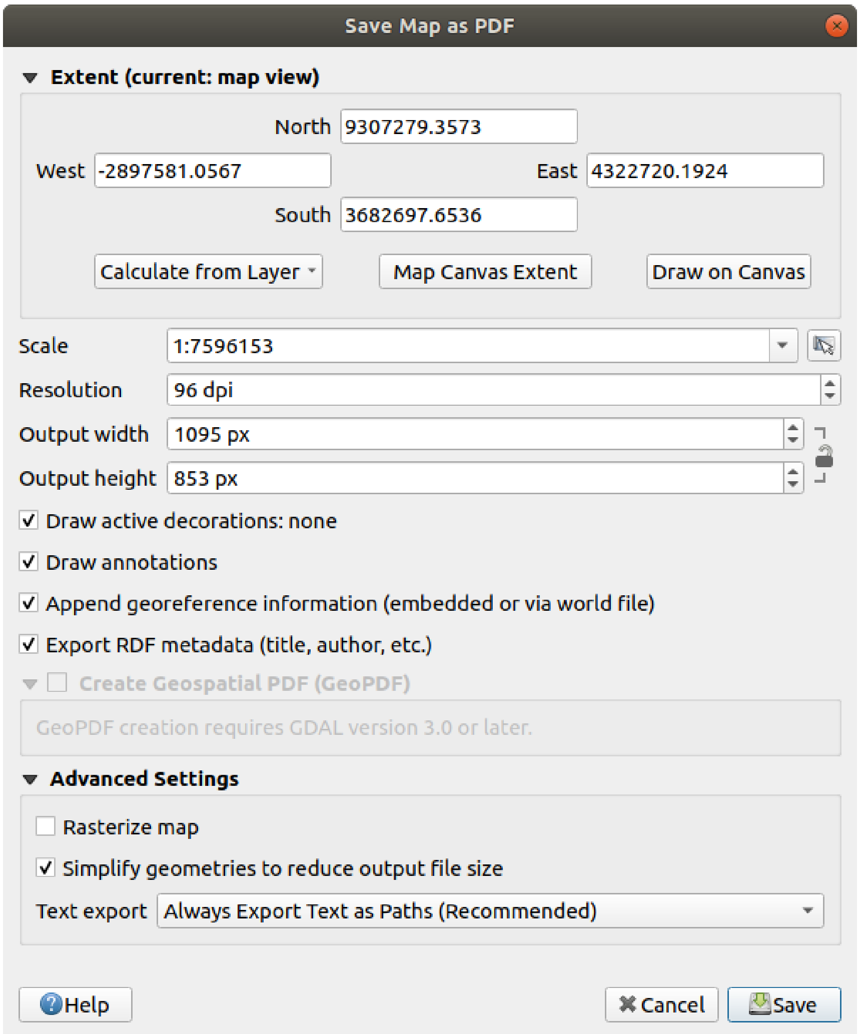


Abb. 7.6: The Save Map as PDF dialog

- *Rasterize map*
- *Simplify geometries to reduce output file size*: Geometries will be simplified while exporting the map 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).
- 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…

7.4 3D-Ansicht

3D visualization support is offered through the 3D map view. You create and open a 3D map view via *View ⍰ New 3D Map View*. A floating QGIS panel will appear. The panel can be docked.

To begin with, the 3D map view has the same extent and view as the 2D main map canvas. A set of navigation tools are available to turn the view into 3D.

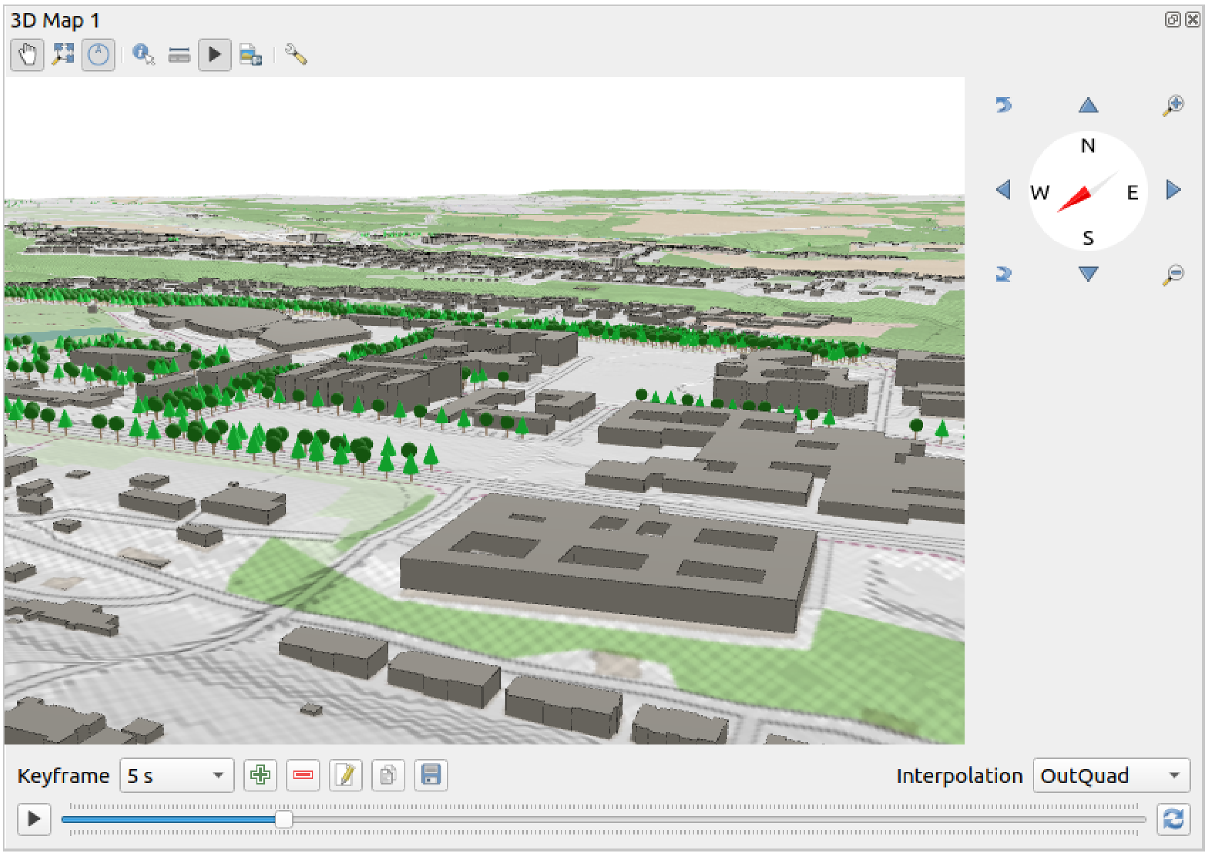


Abb. 7.7: The 3D Map View dialog

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
- 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
- Configure the map view *settings*

7.4.1 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.

7.4.2 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 Repeat button pressed will repeatedly run the animation while clicking stops a running animation.

It is possible to browse the different views of the camera, using the *Keyframe* list. Whenever a time is active, changing the map view will automatically update the associated position. You can also Edit keyframe (time only) or Remove keyframe.

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*.

7.4.3 Scene Configuration

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

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

- Camera's *Field of view*: allowing to create panoramic scenes. Default value is 45°.
- *Terrain*: Before diving into the details, it is worth noting that 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/
  - *Elevation*: Raster layer to be used for generation of the terrain. This layer must contain a band that represents elevation.
  - *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.

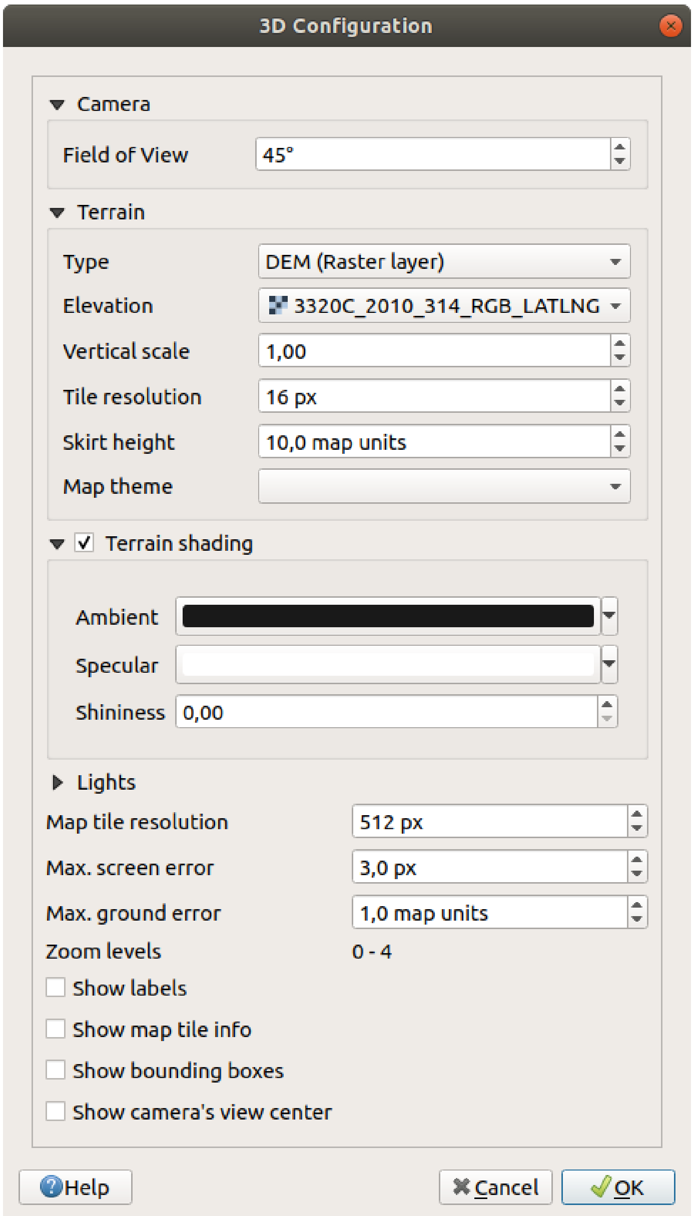


Abb. 7.8: The 3D Map Configuration dialog

  - *Map theme*: Allows you to select the set of layers to display in the map view from predefined *map themes*.
- *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*: You can add up to eight point lights, each with a particular position (in *X*, *Y* and *Z*), *Color*, *Intensity* and *Attenuation*.

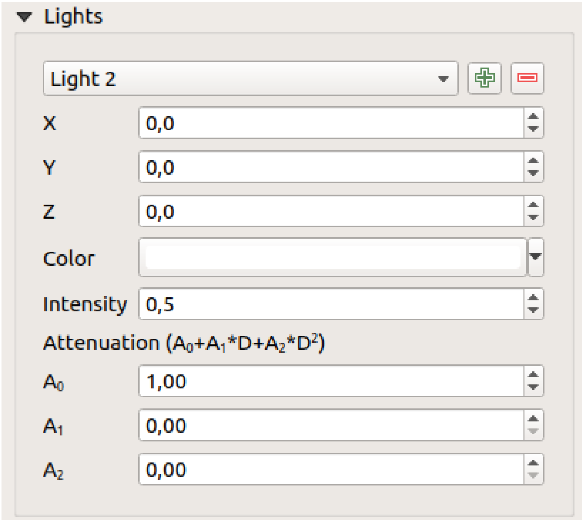


Abb. 7.9: The 3D Map Lights Configuration dialog

- *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 labels*: 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*

7.4.4 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.

7.5 Statusleiste

The status bar provides you with general information about the map view and processed or available actions, and offers you tools to manage the map view.

On the left side of the status bar, the locator bar, a quick search widget, helps you find and run any feature or options in QGIS. Simply type text associated with the item you are looking for (name, tag, keyword…) and you get a list that updates as you write. You can also limit the search scope using *locator filters*. Click the button to select any of them and press the *Configure* entry for global settings.

In the area next to the locator bar, a summary of actions you've carried out will be shown when needed (such as selecting features in a layer, removing layer) or a long description of the tool you are hovering over (not available for all tools).

In case of lengthy operations, such as gathering of statistics in raster layers, executing Processing algorithms or rendering several layers in the map view, a progress bar is displayed in the status bar.

The *Coordinate* option shows the current position of the mouse, following it while moving across the map view. You can set the units (and precision) in the *Project ⍰ Properties…* ⍰ *General* tab. Click on the small button at the left of the textbox to toggle between the Coordinate option and the *Extents* option that displays the coordinates of the current bottom-left and top-right corners of the map view in map units.

Next to the coordinate display you will find the *Scale* display. It shows the scale of the map view. There is a scale selector, which allows you to choose between *predefined and custom scales*.

On the right side of the scale display, press the button to lock the scale to use the magnifier to zoom in or out. The magnifier allows you to zoom in to a map without altering the map scale, making it easier to tweak the positions of labels and symbols accurately. The magnification level is expressed as a percentage. If the *Magnifier* has a level of 100%, then the current map is not magnified. Additionally, a default magnification value can be defined within *Settings ⍰ Options ⍰ Rendering ⍰ Rendering behavior*, which is very useful for high-resolution screens to enlarge small symbols.

To the right of the magnifier tool you can define a current clockwise rotation for your map view in degrees.

On the right side of the status bar, there is a small checkbox which can be used temporarily to prevent layers being rendered to the map view (see section *Layeranzeige kontrollieren*).

To the right of the render functions, you find the *EPSG:code* button showing the current project CRS. Clicking on this opens the *Project Properties* dialog and lets you apply another CRS to the map view.

The Messages button next to it opens the *Log Messages Panel* which has information on underlying processes (QGIS startup, plugins loading, processing tools…)

Depending on the *Plugin Manager settings*, the status bar can sometimes show icons to the right to inform you about availability of new or upgradeable plugins. Click the icon to open the Plugin Manager dialog.

Tipp: Die richtige Maßstabseinheit im Kartenfenster einstellen

When you start QGIS, the default CRS is `WGS 84 (EPSG 4326)` and units are degrees. This means that QGIS will interpret any coordinate in your layer as specified in degrees. To get correct scale values, you can either manually change this setting in the *General* tab under *Project ⍰ Properties…* (e.g. to meters), or you can use the EPSG:code icon seen above. In the latter case, the units are set to what the project projection specifies (e.g., `+units=us-ft`).

Beachten Sie, dass die KBS Wahl beim Start unter *Einstellungen ▸ Optionen ▸ KBS* eingestellt werden kann.

KAPITEL 8

The Browser panel

The QGIS Browser panel is a great tool for browsing, searching, inspecting, copying and loading QGIS resources. Only resources that QGIS knows how to handle are shown in the browser.

Using the Browser panel you can locate, inspect and add data, as described in *The Browser Panel*. In addition, the Browser panel supports drag and drop of many QGIS resources, such as project files, Python scripts, Processing scripts and Processing models.

Python scripts, Processing scripts and Processing models can also be opened for editing in an external editor and the graphical modeller.

You can drag and drop layers from the *Layers* panel to the *Browser* panel, for instance into a GeoPackage or a PostGIS database.

The browser panel (Abb. 8.1) is organised as an expandable hierarchy with some fixed top-level entries that organise the resources handled by the browser. Node entries are expanded by clicking on ▶ to the left of the entry name. A branch is collapsed by clicking on ▼. The Collapse All button collapses all top-level entries.

A filter (Filter Browser) can be used for searching based on entry names (both leaf entries and node entries in the hierarchy). Using the Options pull-down menu next to the filter text field, you can

- toggle *Case Sensitive* search
- set the *Filter pattern syntax* to one of
  - *Normal*
  - *Wildcard(s)*
  - *Regular Expressions*

The *Properties widget*, showing useful information about some entries / resources, can be enabled / disabled using the Enable/disable properties widget button. When enabled, it opens at the bottom of the browser panel, as shown in Abb. 8.2.

A second browser panels can be opened by activating the *Browser (2)* panel in *View ▸ Panels*. Having two browser panels can be useful when copying layers between resources that are locationed deep down in different branches of the browser hierarcy.

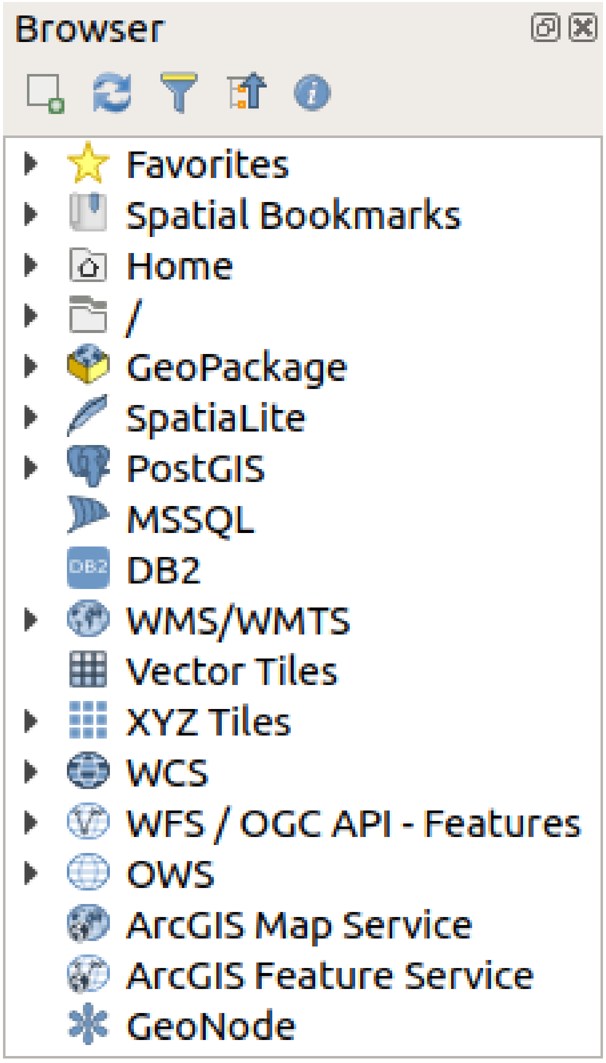


Abb. 8.1: The Browser panel

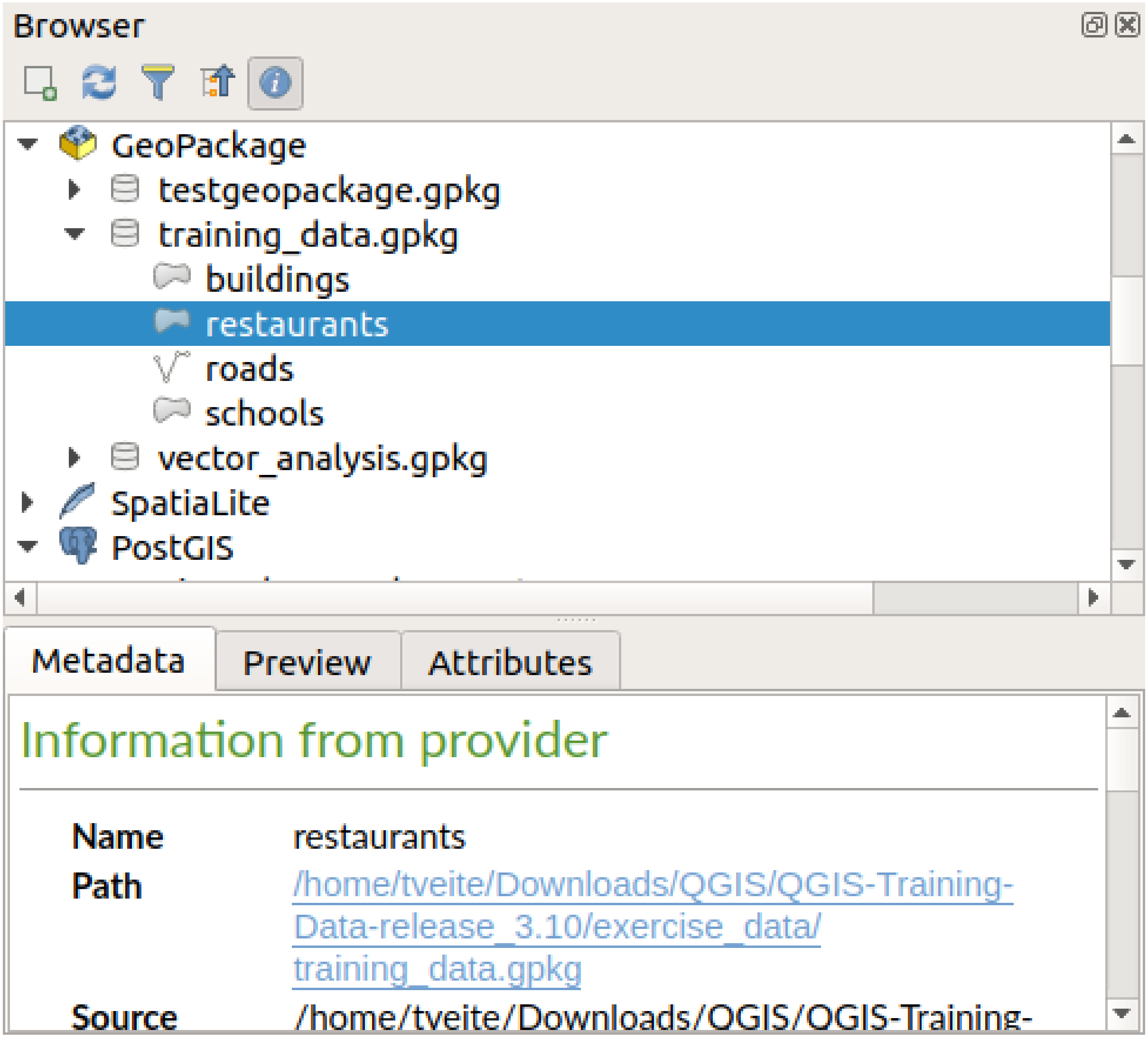


Abb. 8.2: The properties widget

8.1 Resources that can be opened / run from the Browser

A lot can be accomplished in the Browser panel

- Add vector, raster and mesh layers to your map by double-clicking, dragging onto the map canvas or clicking the Add Selected Layers button (after selecting layers)
- Run Python scripts (including Processing algorithms) by double-clicking or dragging onto the map canvas
- Run models by double-clicking or dragging onto the map canvas
- *Extract Symbols…* from QGIS Project files using the context menu
- Copy entries

Resource specific actions are listed for the different resource groups sorted under the top-level entries listed below.

8.2 Browser panel top-level entries

8.2.1 Favorites

Often used file system locations can be tagged as favorites. The ones you have tagged will appear here.

In addition to the operations described under *Home*, the context menu allows you to *Rename Favorite…* and *Remove Favourite*.

8.2.2 Räumliche Lesezeichen

This is where you will find your spatial bookmarks, organised into *Project Bookmarks* and *User Bookmarks*.

From the top level context menu, you can create a bookmark (*New Spatial Bookmark…*), *Show the Spatial Bookmark Manager*, *Import Spatial Bookmarks…* and *Export Spatial Bookmarks…*,

For bookmark entries you can *Zoom to Bookmark*, *Edit Spatial Bookmark…* and *Delete Spatial Bookmark*

8.2.3 Home

Your file system home directory / folder. By right-clicking on an entry, and choosing *Add as a Favorite*, the location will be added to *Favorites*. From the context menu, you can also

- add a directory, Geopackage or ESRI Shapefile format dataset (*Add*)
- hide the directory (*Hide from Browser*)
- toggle *Fast Scan this Directory*
- open the directory in your file manager (*Open Directory*)
- open the directory in a terminal window (*Open in Terminal*)
- inspect properties (*Properties…*, *Directory Properties…*)

8.2.4 /

Your file system root directory / folder.

8.2.5 Geopackage

Geopackage files / databases. From the top level context menu, you can create a Geopackage file / database (*Create Database…*) or add an existing Geopackage file / database (*New Connection…*).

The context menu of each Geopackage lets you remove it from the list (*Remove connection…*), add a new layer or table to the Geopackage (*Create new Layer or Table…*), delete the Geopackage (*Delete <name of geopackage>*) and *Compact Database (VACUUM)*.

For layer/table entries you can

- rename it (*Rename Layer <layer name>…*)
- export it (*Export Layer ▯ To file*)
- add it to the project *Add Layer to Project*
- delete it (*Delete Layer*)
- inspect properties (*Layer Properties…*, *File Properties…*)

8.2.6 SpatiaLite

SpatiaLite database connections.

From the top level context menu, you can create a SpatiaLite file / database (*Create Database…*) or add an existing SpatiaLite file / database (*New Connection…*).

The context menu of each SpatiaLite file lets you delete it (*Delete*).

For layer/table entries you can

- export it (*Export Layer ▯ To file*)
- add it to the project *Add Layer to Project*
- delete it (*Delete Layer*)
- inspect properties (*Layer Properties…*)

8.2.7 PostGIS

PostGIS database connections.

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each connection lets you *Refresh* it, edit it *Edit connection…*, delete it (*Delete connection*) or *Create Schema…*.

The context menu of each schema lets you *Refresh*, *Rename Schema…* or *Delete Schema*.

For layers/tables you can

- rename it (*Rename Table…*)
- remove its contents (*Truncate Table…*)
- export it (*Export Layer ▯ To file*)
- add it to the project (*Add Layer to Project*)
- delete it (*Delete Layer*)

- inspect its properties (*Layer Properties…*)

8.2.8 MSSQL

Microsoft SQL Server connections.

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each connection lets you *Refresh* it, edit it *Edit connection…*, delete it (*Delete connection*) or *Create Schema…*.

The context menu of each schema lets you *Refresh*, *Rename Schema…* or *Delete Schema*.

For layers/tables you can

- rename it (*Rename Table…*)
- remove its contents (*Truncate Table…*)
- export it (*Export Layer ► To file*)
- add it to the project (*Add Layer to Project*)
- delete it (*Delete Layer*)
- inspect its properties (*Layer Properties…*)

8.2.9 DB2

IBM DB2 database connections.

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each connection lets you *Refresh* it, edit it *Edit connection…*, delete it (*Delete connection*) or *Create Schema…*.

The context menu of each schema lets you *Refresh*, *Rename Schema…* or *Delete Schema*.

For layers/tables you can

- rename it (*Rename Table…*)
- remove its contents (*Truncate Table…*)
- export it (*Export Layer ► To file*)
- add it to the project (*Add Layer to Project*)
- delete it (*Delete Layer*)
- inspect its properties (*Layer Properties…*)

8.2.10 WMS/WMTS

Web Map Services (WMS) and Web Map Tile Services (WMTS)

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each WSM/WMTS service lets you *Refresh* it, *Edit…* it and delete it (*Delete*).

Group layers can be added by dragging them onto the map canvas.

For WMS/WMTS layer entries you can

- export it (*Export Layer ► To file*)
- add it to the project (*Add Layer to Project*)
- inspect properties (*Layer Properties…*)

8.2.11 Vector Tiles

Vector tile services

From the top level context menu, you add an existing service (*New Connection…*), and you can *Save Connections…* or *Load Connections…* to / from XML files.

8.2.12 XYZ Tiles

XYZ tile services

From the top level context menu, you add an existing service (*New Connection…*), and you can *Save Connections…* or *Load Connections…* to / from XML files.

For the XYZ tile service entries you can

- edit it (*Edit…*)
- delete it (*Delete*)
- export it (*Export Layer ▸ To file*)
- add it to the project *Add Layer to Project*
- inspect properties (*Layer Properties…*)

8.2.13 WCS

Web Coverage Services

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each WCS lets you *Refresh* it, *Edit…* it and delete it (*Delete*).

For WCS layer entries you can

- export it (*Export Layer ▸ To file*)
- add it to the project (*Add Layer to Project*)
- inspect properties (*Layer Properties…*)

8.2.14 WFS / OGC API - Features

Web Feature Services (WFS) and *OGC API - Features services* (aka WFS3)

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each WFS lets you *Refresh* it, *Edit…* it and delete it (*Delete*).

For WFS layer entries you can

- export it (*Export Layer ▸ To file*)
- add it to the project (*Add Layer to Project*)
- inspect properties (*Layer Properties…*)

8.2.15 OWS

Here you will find a read-only list of all your Open Web Services (OWS) - WMS / WCS / WFS / …

8.2.16 ArcGIS Map Service

8.2.17 ArcGIS Features Service

8.2.18 GeoNode

From the top level context menu, you can add a new connection (*New Connection…*).

The context menu of each service lets you *Refresh* it, *Edit…* it and delete it (*Delete*).

For the service layer entries you can

- export it (*Export Layer ‣ To file*)
- add it to the project (*Add Layer to Project*)
- inspect properties (*Layer Properties…*)

8.3 Resources

- Project files. The context menu for QGIS project files allows you to:
  - open it (*Open Project*)
  - extract symbols (*Extract Symbols…*) - open the style manager that allows you to export symbols to an XML file, add symbols to the default style or export as PNG or SVG.
  - inspect properties (*File Properties…*)

  You can expand the project file to see its layers. The context menu of a layers offers the same actions as elsewhere in the browser.
- QGIS Layer Definition files (QLR) The following actions are available from the context menu:
  - export it (*Export Layer ‣ To file*)
  - add it to the project (*Add Layer to Project*)
  - inspect properties (*Layer Properties…*)
- Processing models (.model3). The following actions are available from the context menu:
  - *Run Model…*)
  - *Edit Model…*)
- QGIS print composer templates (QPT) The following action is available from the context menu:
  - (*New Layout from Template*)
- Python scripts (.py) The following actions are available from the context menu:
  - (*Run script…*)
  - (*Open in External Editor*)
- Recognized raster formats. The following actions are available from the context menu:
  - delete it (*Delete File <dataset name>*)
  - export it (*Export Layer ‣ To file*)
  - add it to the project (*Add Layer to Project*)

  - inspect properties (*Layer Properties…*, *File Properties…*)
- Recognized vector formats. The following actions are available from the context menu:
  - delete it (*Delete File <dataset name>*)
  - export it (*Export Layer To file*)
  - add it to the project (*Add Layer to Project*)
  - inspect properties (*Layer Properties…*, *File Properties…*)

KAPITEL 9

QGIS anpassen

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

- *Style Manager…*: create and manage *symbols, styles and color ramps*.
- *Benutzerprojektionen…*: definieren Sie Ihre eigenen *coordinate reference systems*.
- *Keyboard Shortcuts…*: define your own set of *keyboard shortcuts*. Also, they can be overridden during each QGIS session by the *project properties* (accessible under *Project* menu).
- *Oberflächenanpassung…*: passen Sie die *application interface* an, indem Sie Dialoge oder Werkzeuge verstecken, die Sie wahrscheinlich nicht brauchen.
- *Options…*: set global *options* to apply in different areas of the software. These preferences are saved in the active *User profile* settings and applied by default whenever you open a new project with this profile.

9.1 Optionen

Einige grundlegende Optionen für QGIS können Sie mit dem Dialog *Optionen* einstellen. Wählen Sie den Menüpunkt *Einstellungen* ► *Optionen*. Sie können die Optionen an Ihre Bedürfnisse anpassen. Einige Änderungen können einen Neustart von QGIS erfordern um wirksam zu werden.

Im Nachfolgenden werden die Registerkarten beschrieben, in denen Sie Optionen anpassen können.

Bemerkung: Plugins können Ihre Einstellungen in den Dialog Optionen einbetten

Beachten Sie bitte, dass, während im Folgenden nur Einstellungen der Kernanwendung beschrieben werden, *installed plugins* ihre eigenen Optionen in den Standarddialog einfügen können. Damit wird verhindert, dass jede Erweiterung einen eigenen Konfigurationsdialog mit eigenen Menüeinträgen mitbringt…

9.1.1 Allgemeine Einstellungen

Override System Locale

By default, QGIS relies on your Operating System configuration to set language and manipulate numerical values. Enabling this group allows you to customize the behavior.

- Select from *User interface translation* the language to apply to the GUI
- Select in *Locale (number, date and currency formats)* the system on which date and numeric values should be input and rendered
- *Show group (thousand) separator*

A summary of the selected settings and how they would be interpreted is displayed at the bottom of the frame.

Anwendung

- Select the *Style (QGIS restart required)* ie, the widgets look and placement in dialogs. Possible values depend on your Operating System.
- Define the *UI theme (QGIS restart required)* . It can be ‚default', ‚Night Mapping', or ‚Blend of Gray'
- Define the *Icon size*
- Define the *Font* and its *Size*. The font can be *Qt default* or a user-defined one
- Change the *Timeout for timed messages or dialogs*
- *Hide splash screen at startup*
- *Show QGIS news feed on welcome page*: displays a curated QGIS news feed on the welcome page, giving you a direct way to be aware of project news (user/developer meetings date and summary, community surveys, releases announcements, various tips…)
- *Check QGIS version at startup* to keep you informed if a newer version is released
- *Use native color chooser dialogs* (see *Farbauswahl*)
- *Modeless data source manager dialog* to keep the *data source manager* dialog opened and allow interaction with QGIS interface while adding layers to project

Projektdateien

- *Open project on launch*
  - ‚Welcome Page' (default): can display the „News" feed, the project template(s) and the most recent projects (with thumbnails) of the *user profile*. No project is opened by default.
  - ‚New': opens a new project, based on the default template
  - ‚Most recent': reopens the last saved project
  - and ‚Specific': opens a particular project. Use the … button to define the project to use by default.
- *Neues Projekt aus Vorgabeprojekt erstellen*. Sie haben die Möglichkeit auf *Aktuelles Projekt als Vorgabe speichern* oder *Vorgabe zurücksetzen* zu drücken. Sie können durch Ihre Dateien gehen und ein Verzeichnis definieren in dem Sie ihre benutzerdefinierten Projektvorlagen festlegen. Diese werden dann dem Menü *Projekt ▸ Neu aus Vorlage* hinzugefügt wenn Sie erst *Neues Projekt aus Vorgabeprojekt erstellen* aktivieren und dann ein Projekt in den Projektvorlagenordner speichern.
- 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 wurde erstellt um Makros zu handhaben die geschrieben wurden um eine Aktion auf Projektereignisse auszuführen. Sie können zwischen ‚Nie', ‚Fragen', ‚Nur in dieser Sitzung' und ‚Immer (nicht empfohlen)' wählen.

9.1.2 Systemeinstellungen

SVG-Pfade

Add or Remove *Path(s) to search for Scalable Vector Graphic (SVG) symbols*. These SVG files are then available to symbolize or label the features or decorate your map composition.

When using an SVG file in a symbol or a label, QGIS allows you 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
- load the file from a remote URL: as above, the image will only be loaded on successful retrieval of the remote resource
- embed the SVG file into the item: the file is embedded inside the current project, style database, or print layout template. The SVG file is then always rendered as part of the item. This is a convenient way to create self-contained projects with custom SVG symbols which can be easily shared amongst different users and installations of QGIS.

  It is also possible to extract the embedded SVG file from a symbol or label and save it on disk.

Bemerkung: The above mentioned options for loading and storing an SVG file in a project are also applicable to raster images you may want to use for customizing symbols, labels or decorations.

Erweiterungspfade

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

Dokumentationspfade

Fügen Sie einen neuen *Pfad(e) um nach QGIS-Hilfe zu suchen* ein oder entfernen Sie einen, der für die QGIS-Hilfe benutzt wird. 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

Dient dazu *Benutzeroberflächeneinstellungen zurücksetzen (Neustart erforderlich)* falls Sie eigene *customization* vorgenommen haben.

Umgebung

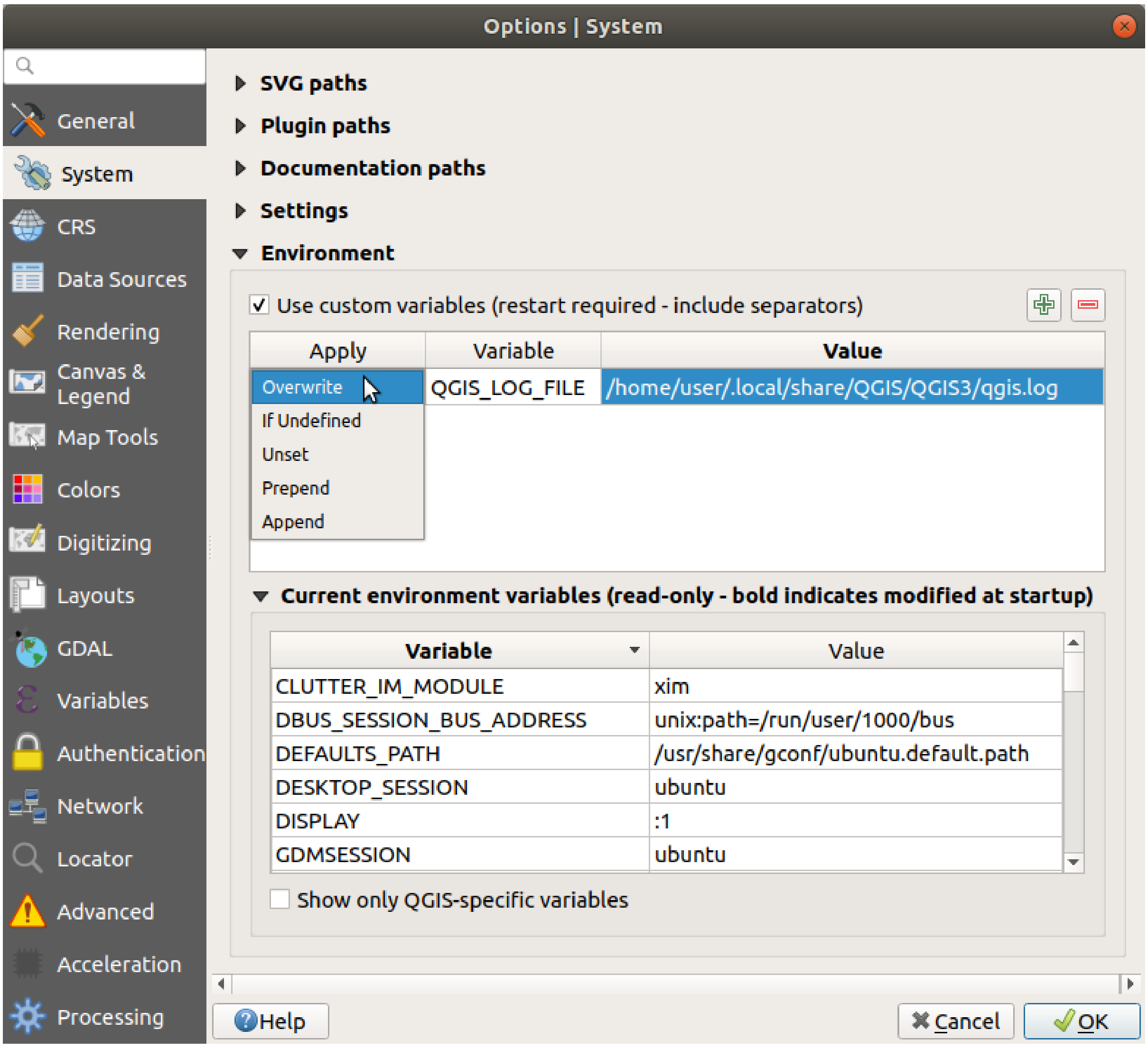


Abb. 9.1: Systemumgebungsvariablen in QGIS

System environment variables can be viewed, and many configured, in the **Environment** group. This is useful for platforms, such as Mac, where a GUI application does not necessarily inherit the user's shell environment. It's also useful for setting and viewing environment variables for the external tool sets controlled by the Processing toolbox (e.g., SAGA, GRASS), and for turning on debugging output for specific sections of the source code.

Use custom variables (restart required - include separators). You can *Add* and *Remove* variables. Already defined environment variables are displayed in *Current environment variables*, and it's possible to filter them by activating *Show only QGIS-specific variables*.

9.1.3 CRS Settings

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

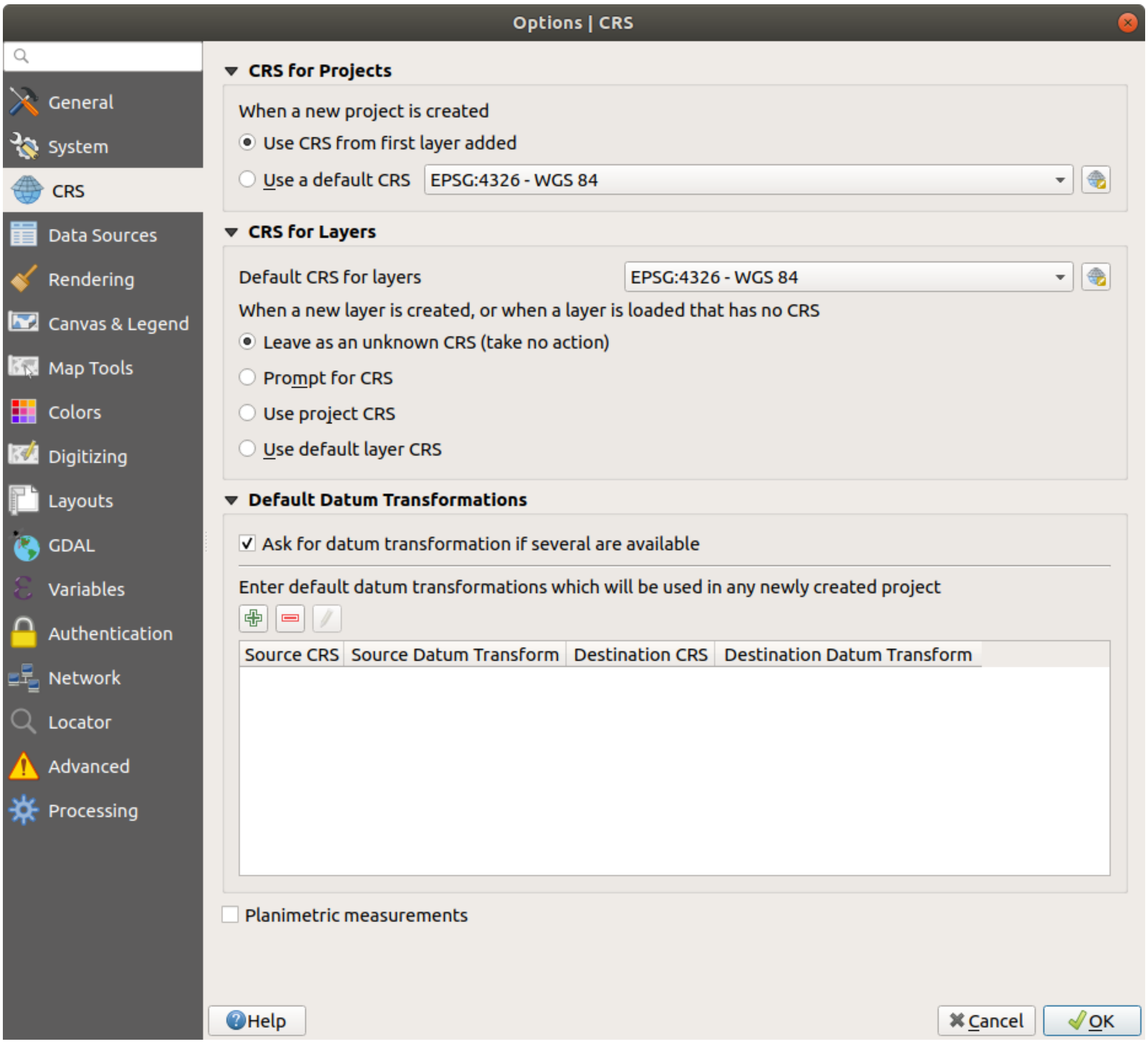


Abb. 9.2: CRS Settings in QGIS

CRS for projects

There is an option to automatically set new project's CRS:

- *Use CRS from first layer added*: the CRS of the project will be set to the CRS of the first layer loaded into it
- *Use a default CRS*: a preselected CRS is applied by default to any new project and is left unchanged when adding layers to the project.

The choice will be saved for use in subsequent QGIS sessions. The Coordinate Reference System of the project can still be overridden from the *Project ▸ Properties… ▸ CRS* tab.

CRS for layers

Default CRS for layers: select a default CRS to use when you create a layer

You can also define the action to take when a new layer is created, or when a layer without a CRS is loaded.

- *Leave as unknown CRS (take no action)*
- *Prompt for CRS*
- *Use project CRS*
- *Use a default CRS*

Datumtransformationsvorgaben

In this group, you can control whether reprojecting layers to another CRS should be:

- automatically processed using QGIS default transformations settings;
- and/or more controlled by you with custom preferences such as:
  - *Ask for datum transformation if several are available*
  - a predefined list of datum transformations to apply by default. See *Datum Transformations* for more details.
- *Planimetric measurements*: sets the default for the „planimetric measurements" property for newly created projects.

9.1.4 Data Sources Settings

Objektattribute und -tabelle

- *Open new attribute tables as docked windows*
- *Copy features as* ‚Plain text, no geometry', ‚Plain text, WKT geometry', or ‚GeoJSON' when pasting features in other applications.
- *Attribute table behavior* : set filter on the attribute table at the opening. There are three possibilities: ‚Show all features', ‚Show selected features' and ‚Show features visible on map'.
- *Default view*: define the view mode of the attribute table at every opening. It can be ‚Remember last view', ‚Table view' or ‚Form view'.
- *Attributtabellenzeilencache* 1,00. Dieser Zeilencache erlaubt es, die N zuletzt geladenen Attributzeilen im Cache zu speichern. Dadurch wird das Arbeiten mit der Attributtabelle schneller. Der Cache wird wieder gelöscht, wenn man die Attributtabelle schliesst.
- *Repräsentation für NULL-Werte*. Hier können Sie einen Wert für Datenfelder die einen NULL-Wert enthalten festlegen.

Tipp: Improve opening of big data attribute table

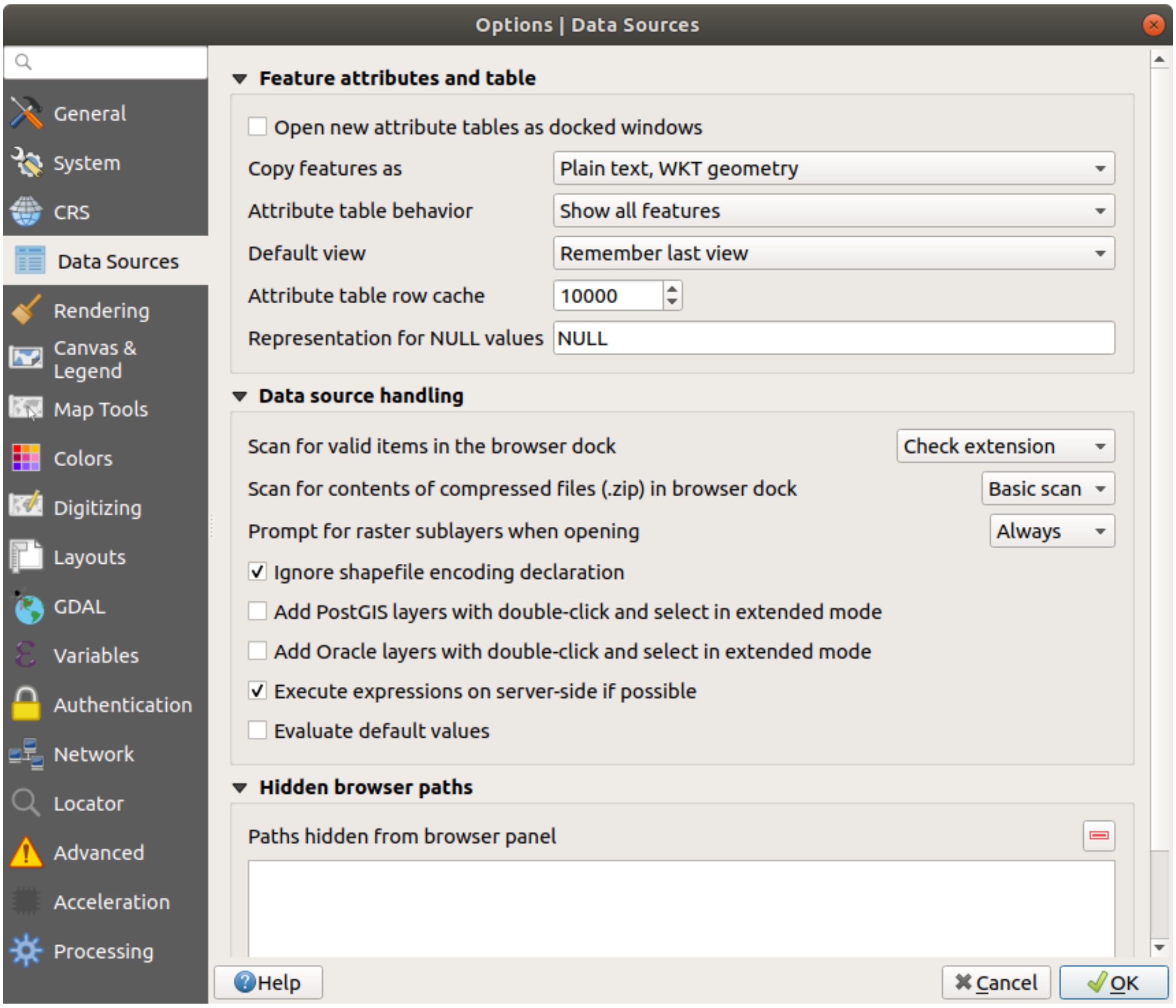


Abb. 9.3: Data Sources Settings in QGIS

When working with layers with big amount of records, opening the attribute table may be slow as the dialog request all the rows in the layer. Setting the *Attribute table behavior* to **Show features visible on map** will make QGIS request only the features in the current map canvas when opening the table, allowing a quick data loading.

Note that data in this attribute table instance will be always tied to the canvas extent it was opened with, meaning that selecting **Show All Features** within such a table will not display new features. You can however update the set of displayed features by changing the canvas extent and selecting **Show Features Visible On Map** option in the attribute table.

Datenquellenbehandlung

- *Nach gültigen Element im Browser suchen* . Sie können zwischen ‚Erweiterung prüfen' und ‚Dateiinhalt prüfen' wählen.
- *Scan for contents of compressed files (.zip) in browser dock* defines how detailed is the widget information at the bottom of the Browser panel when querying such files. ‚No', ‚Basic scan' and ‚Full scan' are possible options.
- *Beim Öffnen nach Rasterunterlayern fragen*. Einige Rasterformate unterstützen Unterlayer - sie werden Subdataset in GDAL genannt. Ein Beispiel sind netCDF-Dateien - wenn es viele netCDF-Variablen gibt interpretiert GDAL jede Variable als Subdataset. Die Option ermöglicht es Ihnen zu steuern wie mit Unterlayern umgegangen wird wenn eine Datei mit Unterlayern geöffnet wird. Sie haben die folgenden Wahlmöglichkeiten:
  - ‚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
- *Shapefile-Kodierungsangabe ignorieren*. Wenn eine Shapedatei Kodierungsinformationen enthält wird dieses von QGIS ignoriert.
- *Execute expressions on server-side if possible*: When requesting features from a datasource, QGIS will try to optimize requests by sending filter criteria directly to the server and only download the features which match the criteria. For example, if for a list on the user interface only the farmers which live in Bern should be listed, QGIS will send a `WHERE "hometown" = 'Bern'` to the database. In some cases, filter criteria are too complex to be translated from QGIS Expressions to database compatible SQL. In those cases, QGIS will download the whole data and filter locally to be on the safe side, which is much less performant.

  By disabling this option, QGIS can be forced to always download the whole data and filter locally, at the expense of a performance penalty. This option is meant as a safety break and should only be deactivated if you identify a misbehavior of the QGIS expression translation engine.
- *Evaluate default values* defines whether default values from database provider should be calculated when digitizing the new feature (checked status) or when saving the changes.

Ausgeblendeter Browserpfad

This widget lists all the folders you chose to hide from the *Browser panel*. Removing a folder from the list will make it available in the *Browser* panel.

9.1.5 Rendering Settings

Zeichenverhalten

- *By default new layers added to the map should be displayed*: unchecking this option can be handy when loading multiple layers to avoid each new layer being rendered in the canvas and slow down the process
- *Wo möglich den Darstellungscache benutzen, um das Neuzeichnen zu beschleunigen*
- *Layer mit vielen CPU-Kernen parallel zeichnen*
- *Max. zu benutzende Kerne*
- *Kartenaktualisierungsintervall*
- *Geometrievereinfachung für neue Layer voreinstellen*
- *Vereinfachungsschwelle*
- *Simplification algorithm*: This option performs a local „on-the-fly" simplification on feature's and speeds up geometry rendering. It doesn't change the geometry fetched from the data providers. This is important when you have expressions that use the feature geometry (e.g. calculation of area) - it ensures that these calculations are done on the original geometry, not on the simplified one. For this purpose, QGIS provides three algorithms: ‚Distance' (default), ‚SnapToGrid' and ‚Visvalingam'.
- *Simplify on provider side if possible*: the geometries are simplified by the provider (PostGIS, Oracle…) and unlike the local-side simplification, geometry-based calculations may be affected
- *Größter Maßstab bis zu dem der Layer vereinfacht werden soll*
- *Magnification level* (see the *magnifier*)

Bemerkung: Besides the global setting, feature simplification can be set for any specific layer from its *Layer properties* ▸ *Rendering* menu.

Zeichenqualität

- *Linie auf Kosten der Zeichengeschwindigkeit weniger gezackt zeichnen*

Curve segmentation

- *Segmentation tolerance*: this setting controls the way circular arcs are rendered. **The smaller** maximum angle (between the two consecutive vertices and the curve center, in degrees) or maximum difference (distance between the segment of the two vertices and the curve line, in map units), the **more straight line** segments will be used during rendering.
- *Tolerance type*: it can be *Maximum angle* or *Maximum difference* between approximation and curve.

Raster

- Mit *RGB Kanalauswahl* können Sie Nummer für den Roten Kanal, Grünen Kanal 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' and ‚Cubic'. 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).

Kontrasverbesserung

Contrast enhancement options can be applied to *Single band gray*, *Multi band color (byte/band)* or *Multi band color (>byte/band)*. For each, you can set:

- the *Algorithm* to use, whose values can be ‚No stretch', ‚Stretch to MinMax', ‚Stretch and Clip to MinMax' or ‚Clip to MinMax'

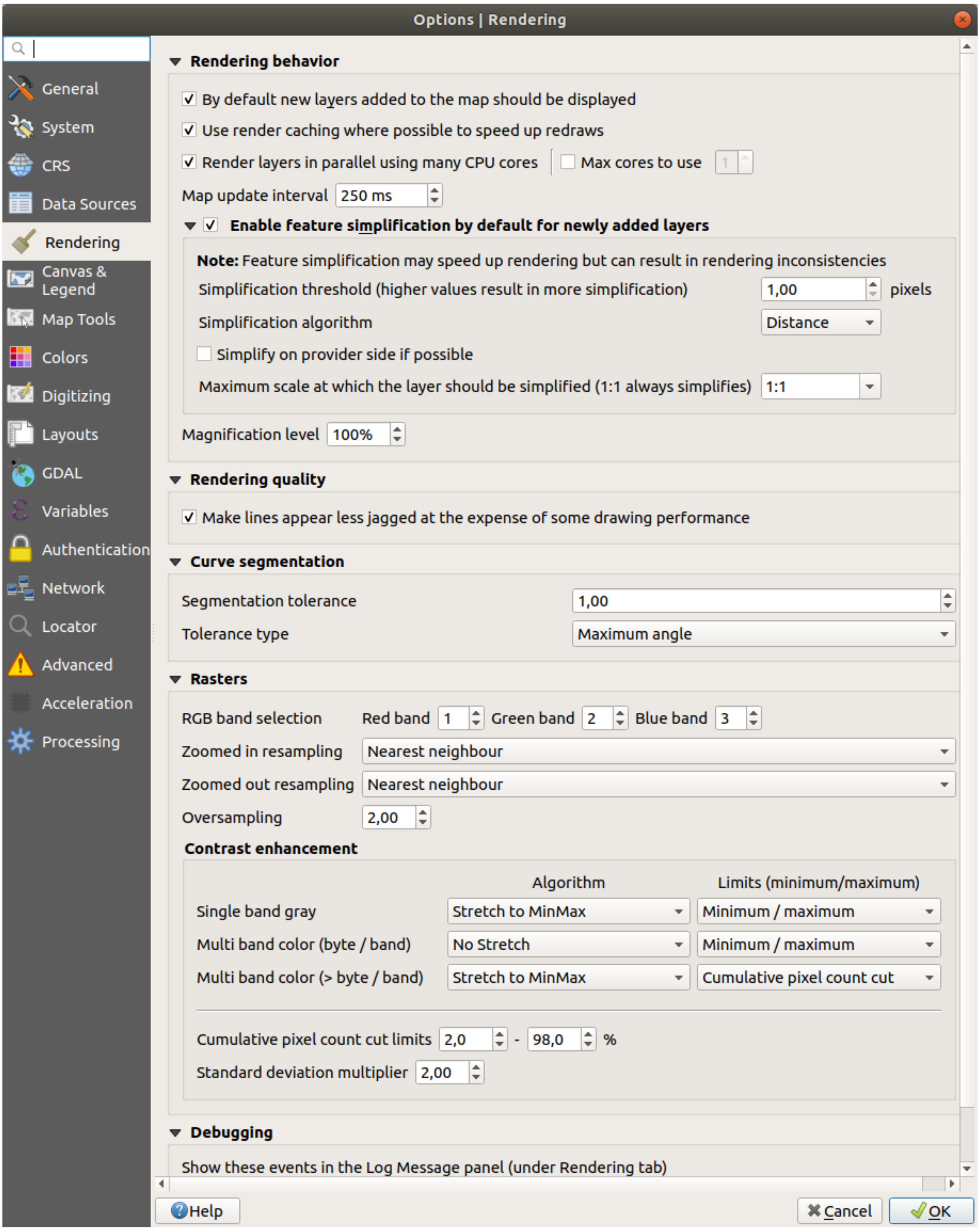


Abb. 9.4: Rendering tab of Project Properties dialog

- the *Limits (minimum/maximum)* to apply, with values such as ‚Cumulative pixel count cut', ‚Minimum/Maximum', ‚Mean +/- standard deviation'.

For rasters rendering, you can also define the following options:

- *Kommulative Pixelanzahl-Schnittgrenzen*
- *Standardabweichungsfaktor*

Fehlersuche

- *Map canvas refresh* to debug rendering duration in the *Log Messages* panel.

9.1.6 Canvas and Legend Settings

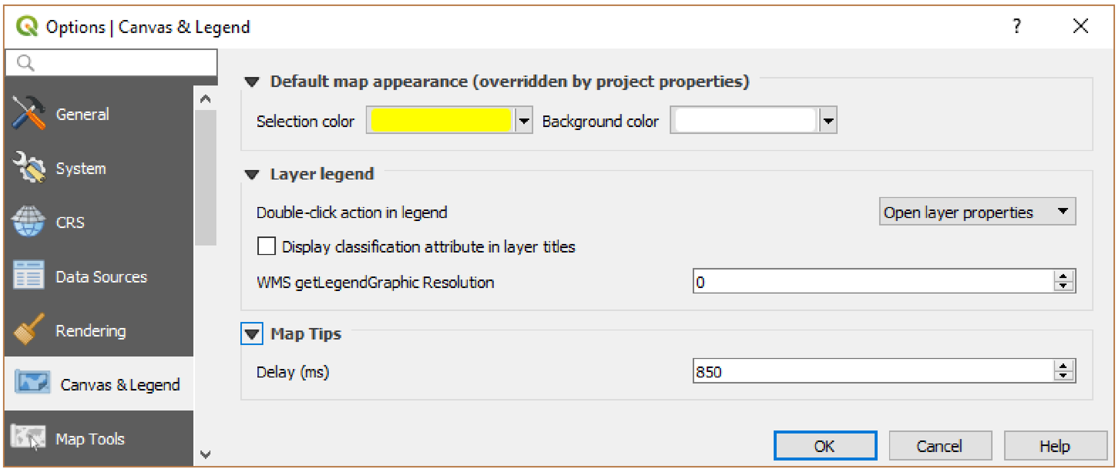


Abb. 9.5: Canvas and Legend Settings

These properties let you set:

- the **Default map appearance (overridden by project properties)**: the *Selection color* and *Background color*.
- **Layer legend** interaction:
  - *Double click action in legend* . You can either ‚Open layer properties', ‚Open attribute table' or ‚Open layer styling dock' with the double click.
  - *Display classification attribute names* in the Layers panel, e.g. when applying a categorized or rule-based renderer (see *Symbology Properties* for more information).
  - the *WMS getLegendGraphic Resolution*
- the *Delay* in milliseconds of layers *map tips* display

9.1.7 Map tools Settings

This tab offers some options regarding the behavior of the *Identify tool*.

- *Search radius for identifying features and displaying map tips* is a tolerance distance within which the identify tool will depict results as long as you click within this tolerance.
- *Highlight color* allows you to choose with which color features being identified should be highlighted.
- *Puffer*, legt eine Pufferdistanz, die durch die Umrisslinie der Hervorhebung dargestellt wird, fest.
- *Minimalbreite*, legt fest wie dick die Umrisslinie eines hervorgehobenen Objektes sein soll.

Messwerkzeug

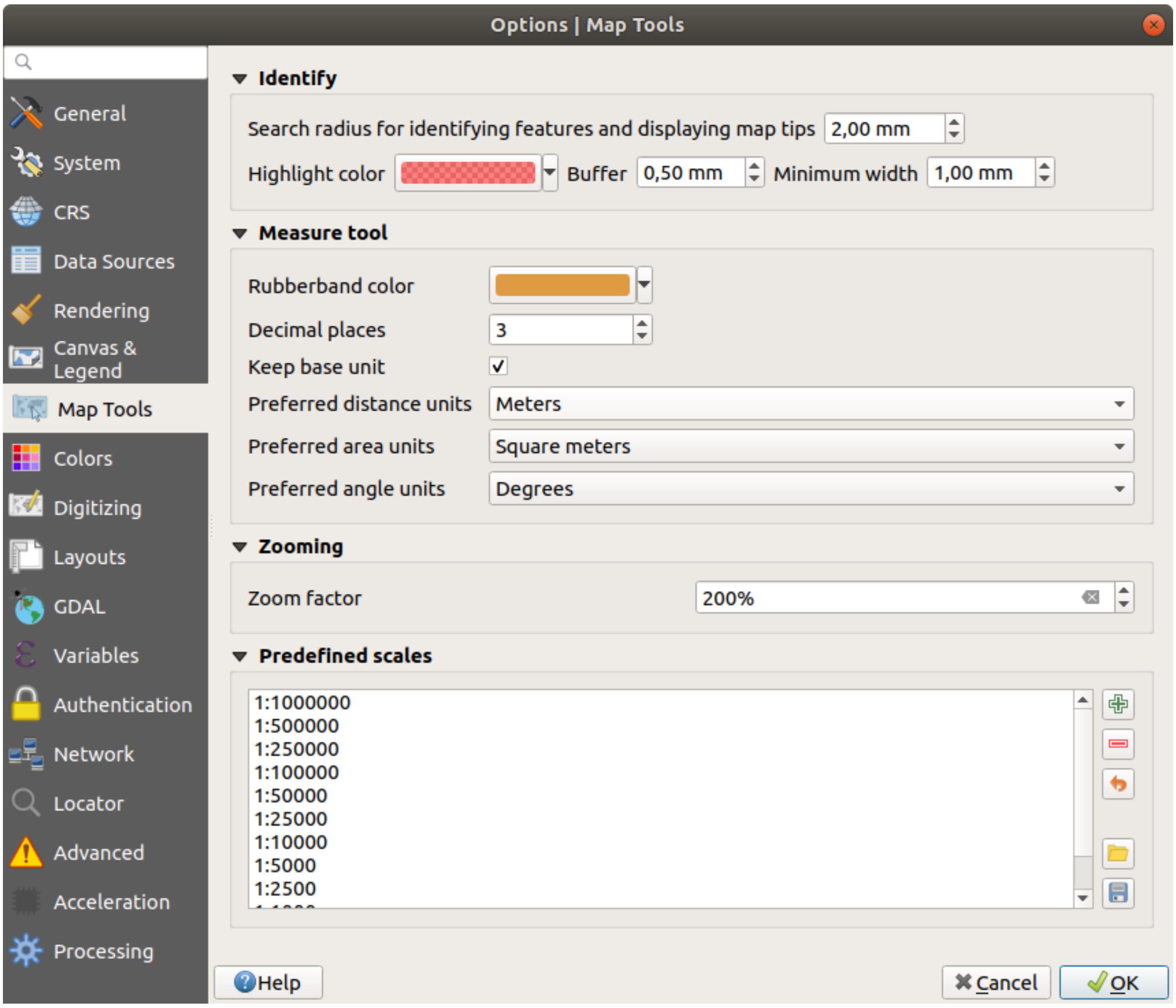


Abb. 9.6: Map tools Settings in QGIS

- Definieren Sie *Gummibandfarbe* für das Meßwerkzeug
- Definieren Sie *Dezimalstellen*
- *Keep base unit* nicht automatisch große Zahle umzuwandeln (z. B. Meter zu Kilometer)
- *Preferred distance units*: options are ‚Meters', ‚Kilometers', ‚Feet', ‚Yards', ‚Miles', ‚Nautical Miles', ‚Centimeters', ‚Millimeters', ‚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 degrees' or ‚Map Units'
- *Preferred angle units*: options are ‚Degrees', ‚Radians', ‚Gon/gradians', ‚Minutes of arc', ‚Seconds of arc', ‚Turns/revolutions', milliradians (SI definition) or mil (NATO/military definition)

Verschieben und Zoomen

- Define a *Zoom factor* for zoom tools or wheel mouse

Vordefinierte Maßstäbe

Hier können Sie eine Liste von vordefinierten Skalen finden. Mit dem und Knopf fügen Sie personalisierte hinzu oder löschen diese. Sie können auch Skalen aus/zu einer `.XML` Datei exportieren/importieren. Beachten Sie, dass Sie immer noch die Möglichkeit haben, um die Änderungen zu entfernen und auf die vordefinierte Liste zurückzusetzen.

9.1.8 Colors Settings

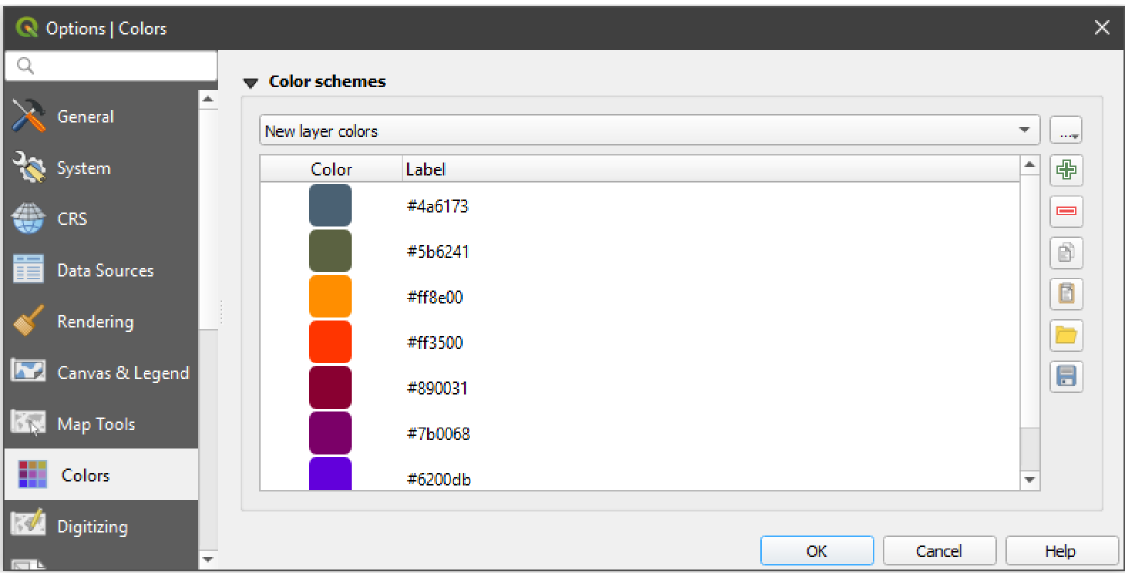


Abb. 9.7: Colors Settings

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 *Default 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.

9.1.9 Digitizing Settings

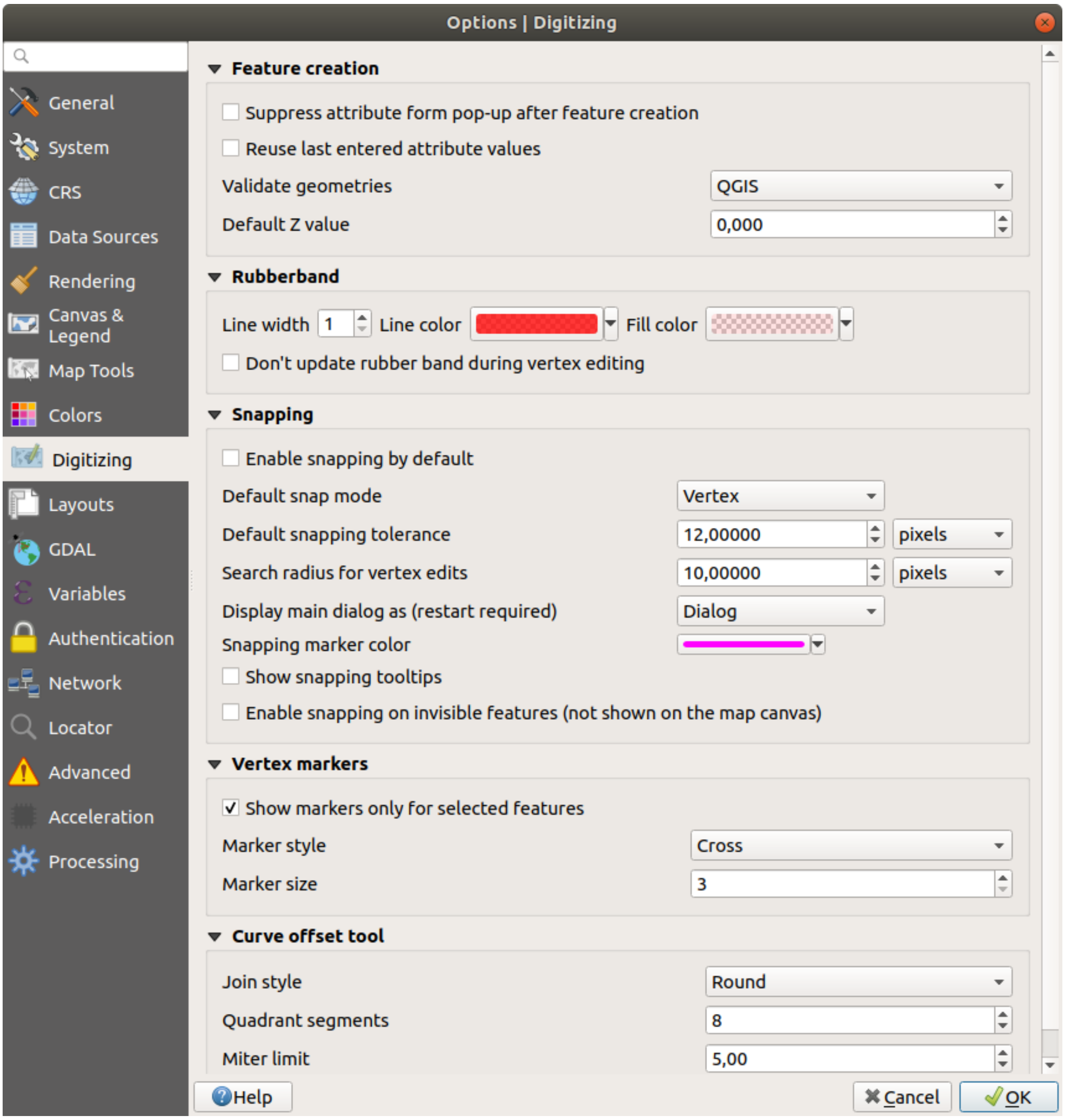


Abb. 9.8: Digitizing Settings in QGIS

This tab helps you configure general settings when *editing vector layer* (attributes and geometry).

Objekterzeugung

- *Suppress attribute form pop-up after feature creation*: this choice can be overridden in each layer properties dialog.
- *Reuse last entered attribute values*.
- *Geometrien prüfen*. Das Bearbeiten komplexer Linien/Polygone mit vielen Stützpunkten kann zu einer erheblichen Verlangsamung der Darstellung führen. Das liegt an den Standard-Validierungsverfahren, die viel Zeit benötigen. Um die Darstellung zu beschleunigen ist es möglich die Geometrieüberprüfung von GEOS (von GEOS 3.3.an) zu wählen oder die Validierung ganz auszuschalten. Die GEOS Geometrieüberprüfung ist viel schneller, aber der Nachteil ist dass nur das erste Geometrieproblem gemeldet wird.

  Note that depending on the selection, reports of geometry errors may differ (see *Types of error messages and their meanings*)
- *Default Z value* to use when creating new 3D features.

Gummiband

- Define Rubberband *Line width*, *Line color* and *Fill color*.
- *Don't update rubberband during vertex editing*.

Objektfang

- *Enable snapping by default* activates snapping when a project is opened
- Define *Default snap mode* (‚Vertex', ‚Vertex and segment', ‚Segment')
- Definieren Sie die *Voreingestellte Fangtoleranz* in Karteneinheiten oder Pixeln
- Definieren Sie den *Suchradius für die Stützpunktbearbeitung* in Karteneinheiten oder Pixeln.
- *Display main dialog as (restart required)*: set whether the Advanced Snapping dialog should be shown as ‚Dialog' or ‚Dock'.
- *Snapping marker color*
- *Show snapping tooltips* such as name of the layer whose feature you are about to snap. Helpful when multiple features overlap.
- *Enable snapping on invisible features (not shown on the map canvas)*

Stützpunktmarken

- *Markierungen nur für gewählte Objekte anzeigen*
- Legen Sie für die Stützpunktmarken den *Markierungsstil* (‚Kreuz' (standard), ‚Teiltransparenter Kreis' oder ‚Keine') fest
- Define vertex *Marker size (in millimeter)*

Werkzeug zum Linien versetzen

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.

- *Join style*: ‚Round', ‚Mitre' or ‚Bevel'
- *Quadrantensegmente*
- *Eckengrenze*

9.1.10 Layouts Settings

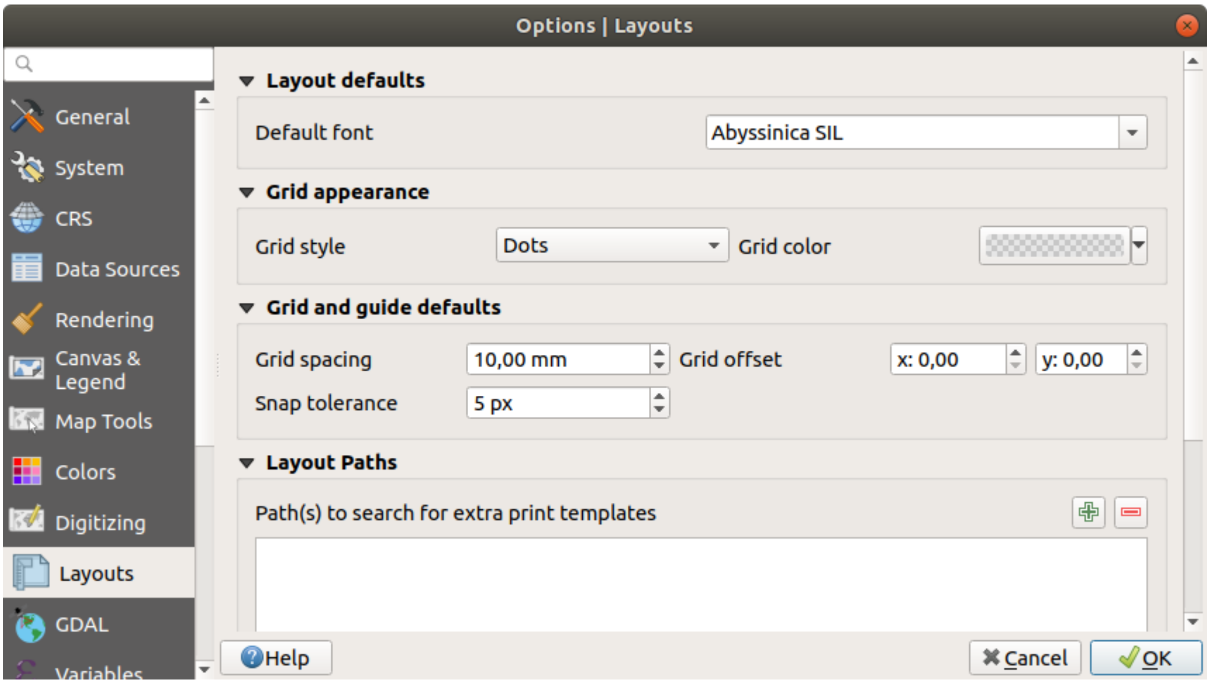


Abb. 9.9: Layouts Settings in QGIS

Zusammenstellungsvoreinstellungen

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

Gitterdarstellung

- Definieren Sie den *Gitterstil* (‚Ausgefüllt', ‚Punkte', ‚Kreuze')
- Definieren Sie *Gitterfarbe*

Gitter- und Führungsvoreinstellungen

- Definieren Sie *Zwischenräume* 1,00
- Define the *Grid offset* 1,00 for X and Y
- Legen Sie die *Fangtoleranz* 1,00 fest

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.

9.1.11 GDAL Settings

GDAL is a data exchange library for vector and raster files. It provides drivers to read and or write data in different formats. The *GDAL* tab currently exposes the drivers for raster formats with their capabilities.

GDAL driver options

This frame provides ways to customize the behavior of drivers that support read and write access:

- *Edit create options*: allows you to edit or add different profiles of file transformation, i.e. a set of predefined combinations of parameters (type and level of compression, blocks size, overview, colorimetry, alpha…) to use when outputting raster files. The parameters depend on the driver.

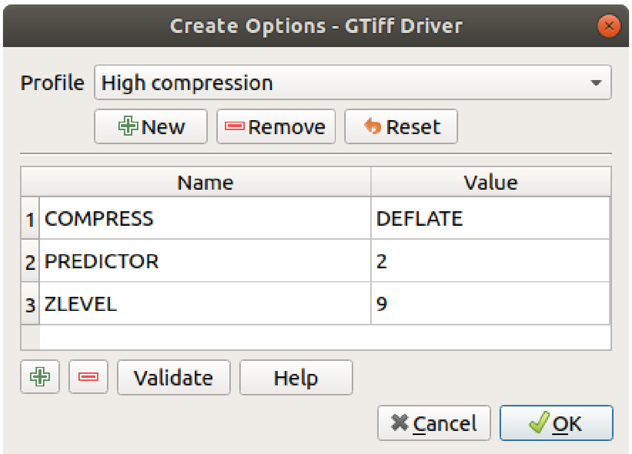


Abb. 9.10: Sample of create options profile (for GeoTiff)

The upper part of the dialog lists the current profile(s) and allows you to add new ones or remove any of them. You can also reset the profile to its default parameters if you have changed them. Some drivers (eg, GeoTiff) have some sample of profiles you can work with.

At the bottom of the dialog:

  – The button lets you add rows to fill with the parameter name and value

  – The button deletes the selected parameter

  – Click the *Validate* button to check that the creation options entered for the given format are valid

  – Use the *Help* button to find the parameters to use, or refer to the GDAL raster drivers documentation.

- *Edit Pyramids Options*

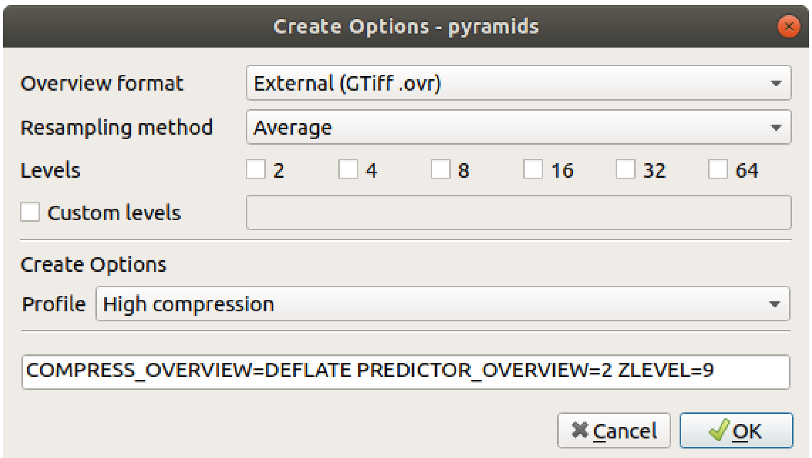


Abb. 9.11: Sample of Pyramids profile

GDAL drivers

In this frame, you can define which GDAL driver is to use to read and/or write files, as in some cases more than one GDAL driver is available.

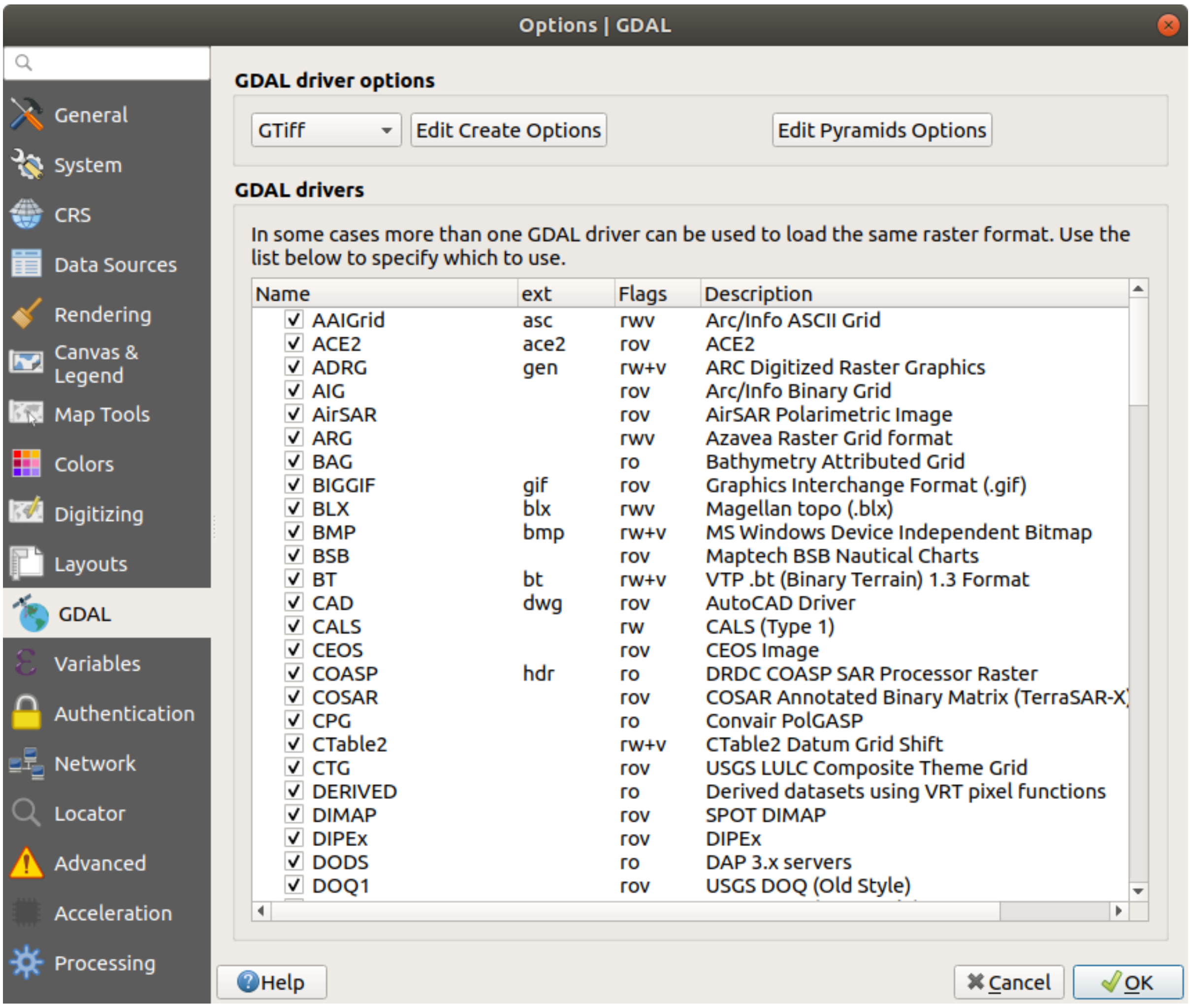


Abb. 9.12: GDAL Settings in QGIS

Tipp: Double-click a driver that allows read and write access (`rw+(v)`) opens the *Edit Create options* dialog for customization.

9.1.12 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.

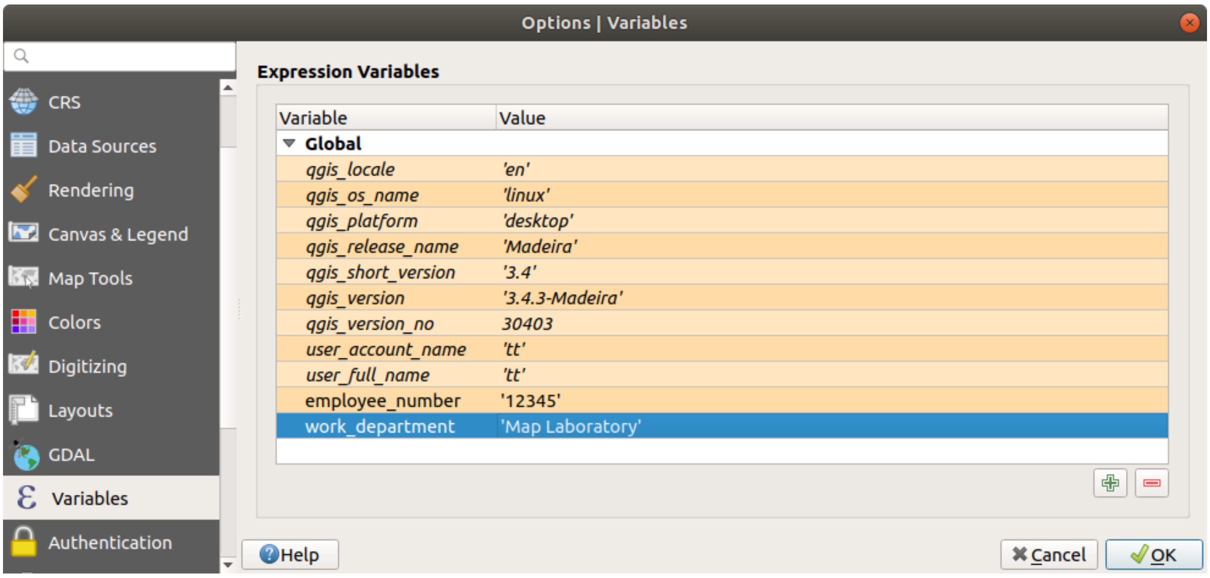


Abb. 9.13: Variables Settings in QGIS

9.1.13 Authentication Settings

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

9.1.14 Network Settings

Allgemein

- *WMS Suchadresse* - Standard ist `http://geopole.org/wms/search?search=%1&type=rss`
- Definieren Sie *Zeitüberschreitung für Netzwerkanfragen (ms)* - Standard ist 60000
- Define *Default expiration period for WMS Capabilities (hours)* - default is 24
- Definieren Sie *Verfallszeitraumvorgabe für WMS-C/WMTS Kacheln (Stunden)* - Standard ist 24
- Define *Max retry in case of tile or feature request errors*
- Definieren Sie *User-Agent*

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.

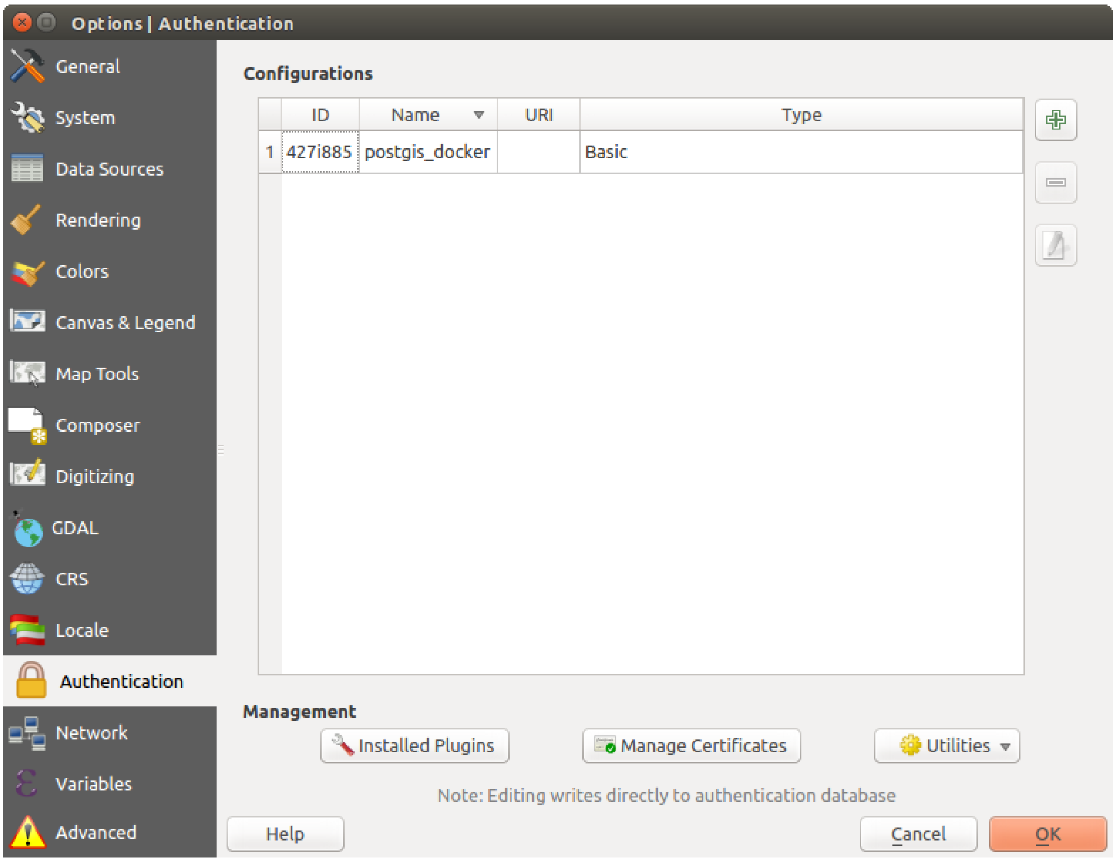


Abb. 9.14: Authentication Settings in QGIS

- *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 *Figure_Network_Tab*). 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/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.

9.1.15 Locator Settings

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

- 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.

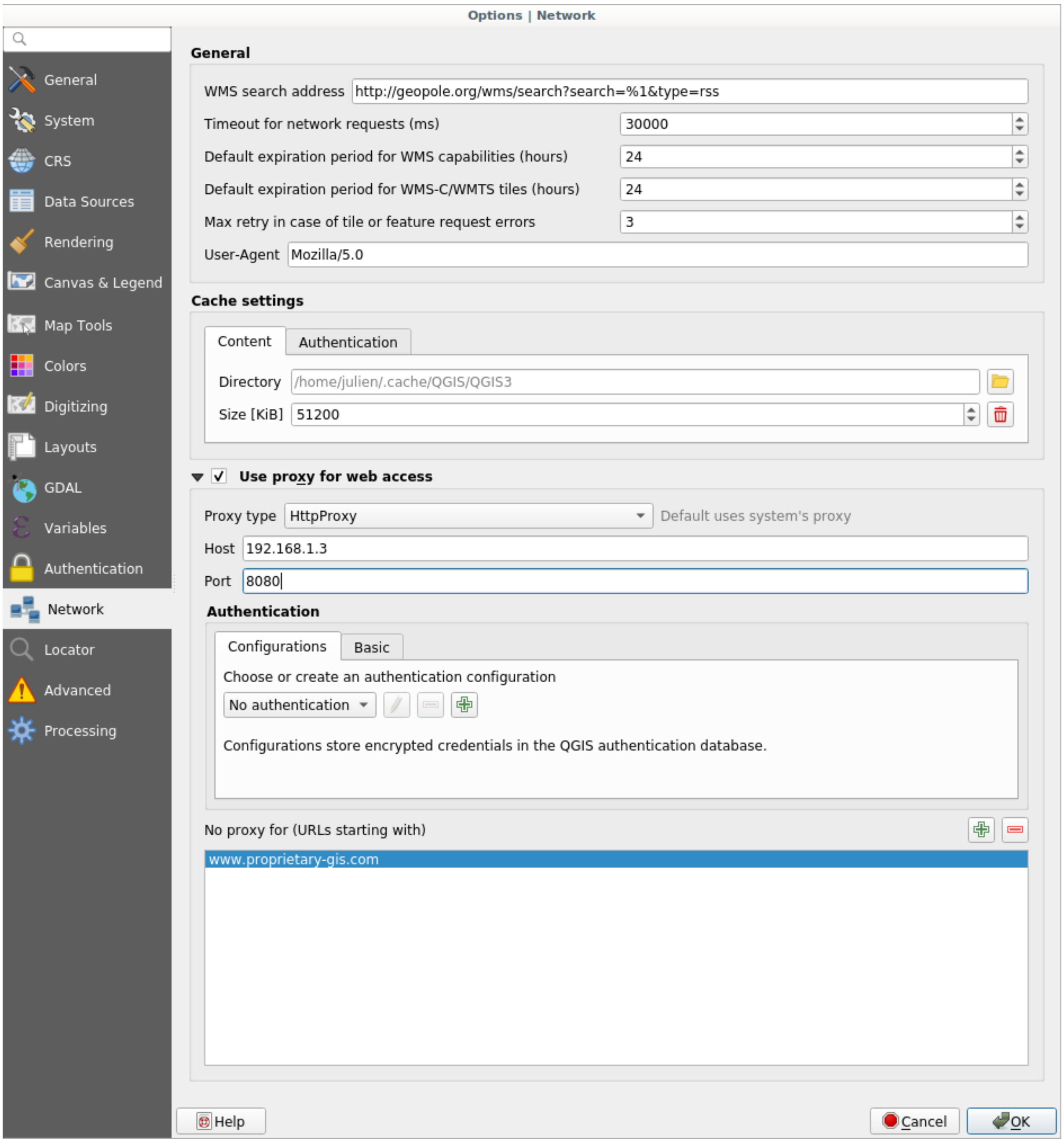


Abb. 9.15: Proxy-Einstellungen in QGIS

- Features in all layers (`af`): searches for matching attributes in the *display name* of each *searchable layers* and zooms to the selected feature.
- 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.
- Processing (`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.

For each filter, you can customize the filter, set whether it is enabled by default or not. The set of default locator filters can be extended by plugins, eg for OSM nominatim searches, direct database searching, layer catalog searches.

The locator search bar can be activated pressing `Ctrl+K`. Type your text to perform a search. By default, results are returned for all enabled locator filters but you can limit the search to a certain filter by prefixing your text with the locator filter prefix, ie. typing `l cad` will return only the layers whose name contains `cad`. Click on the result to execute the corresponding action, depending on the type of item.

Searching is handled using threads, so that results always become available as quickly as possible, regardless of whether any slow search filters may be installed. They also appear as soon as each result is encountered by each filter, which means that e.g. a file search filter will show results one by one as the file tree is scanned. This ensures that the UI is always responsive even if a very slow search filter is present (e.g. one which uses an online service).

Tipp: Quick access to the locator's configurations

Click on the icon inside the locator widget on the status bar to display the list of filters you can use and a *Configure* entry that opens the *Locator* tab of the *Settings ⍰ Options…* menu.

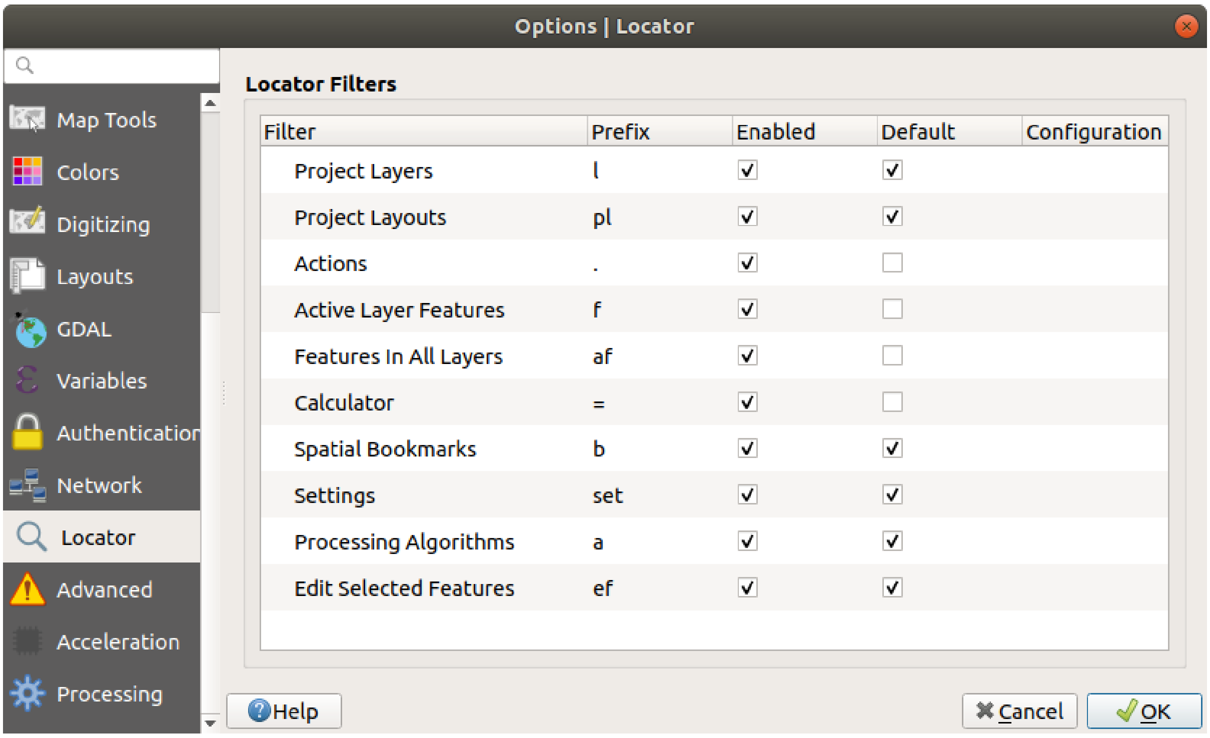


Abb. 9.16: Locator Settings in QGIS

9.1.16 Advanced Settings

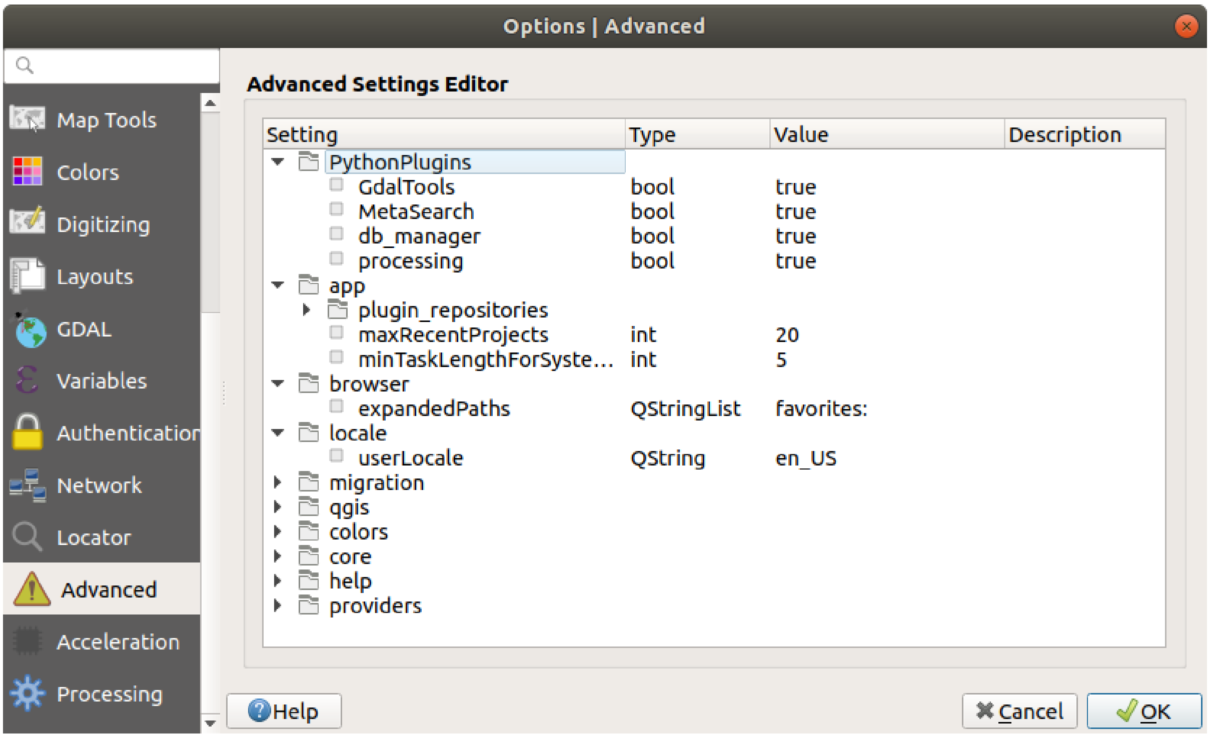


Abb. 9.17: Advanced Settings tab in QGIS

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.

9.1.17 Acceleration Settings

OpenCL acceleration settings.

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

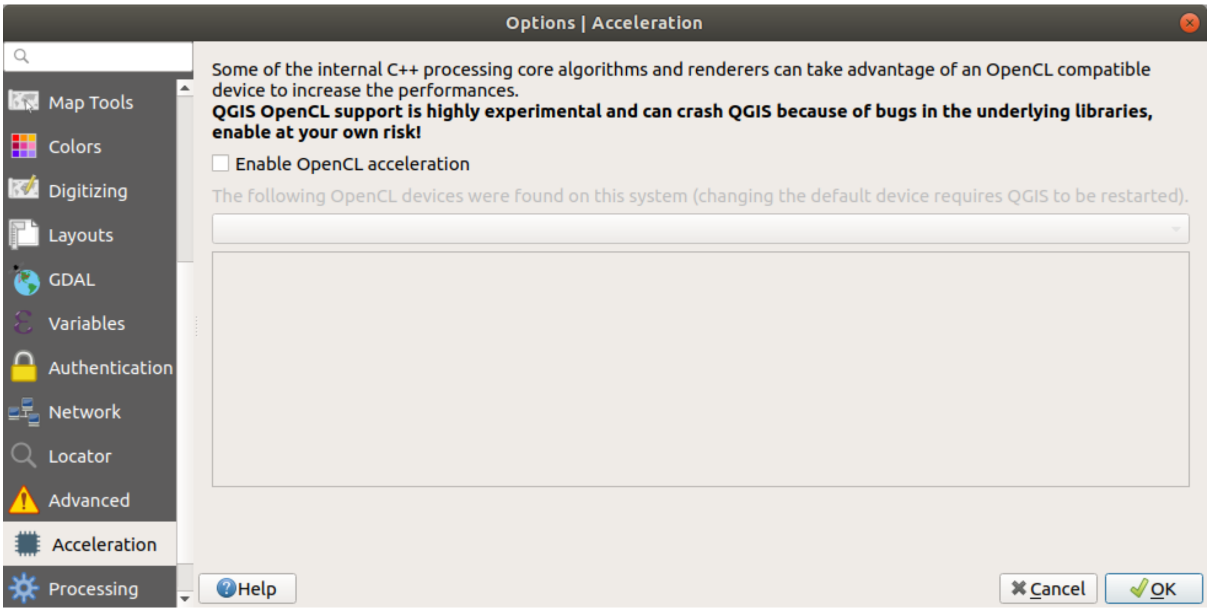


Abb. 9.18: Acceleration tab

9.1.18 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*.

9.2 Working with User Profiles

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*
- 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\(user)`.
   - 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*.

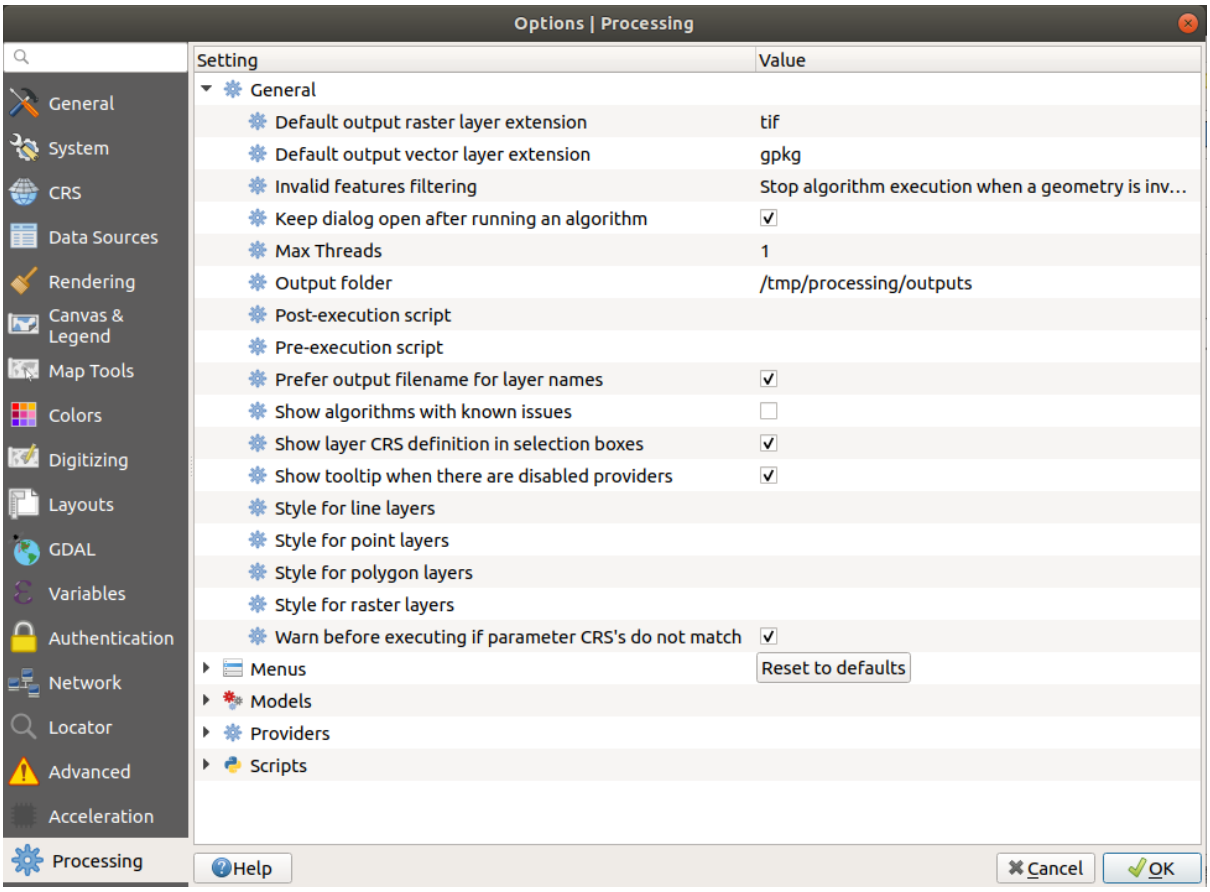


Abb. 9.19: Processing Settings tab in QGIS

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*.

Unless changed, the profile of the last closed QGIS session will be used in the following QGIS sessions.

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.

9.3 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 in the browser). 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.
- 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.

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.
  - 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…).
- Die *Einheiten für Entfernungsmessung* für Länge und Umfang und die *Einheiten für Flächenmessung*. Diese Einstellungen, die standardmäßig über die festgelegten Einheiten in den QGIS Optionen definiert sind, werden dann aber überschrieben für das aktuelle Projekt, werden eingesetzt in:
  - Attributtabellen Feld mit Aktualisierungsleiste

- Feldrechner Berechnungen
- Abfragewerkzeug abgeleitet von Länge, Umfang und Flächenwerten
- Standardeinheiten gezeigt im Messen Dialog

The *Coordinate display* allows you to choose and customize the format of units to use to display the mouse coordinate in the status bar and the derived coordinates shown via the identify tool.

Finally, you can set a *Project predefined scales* list, which overrides the global predefined scales.

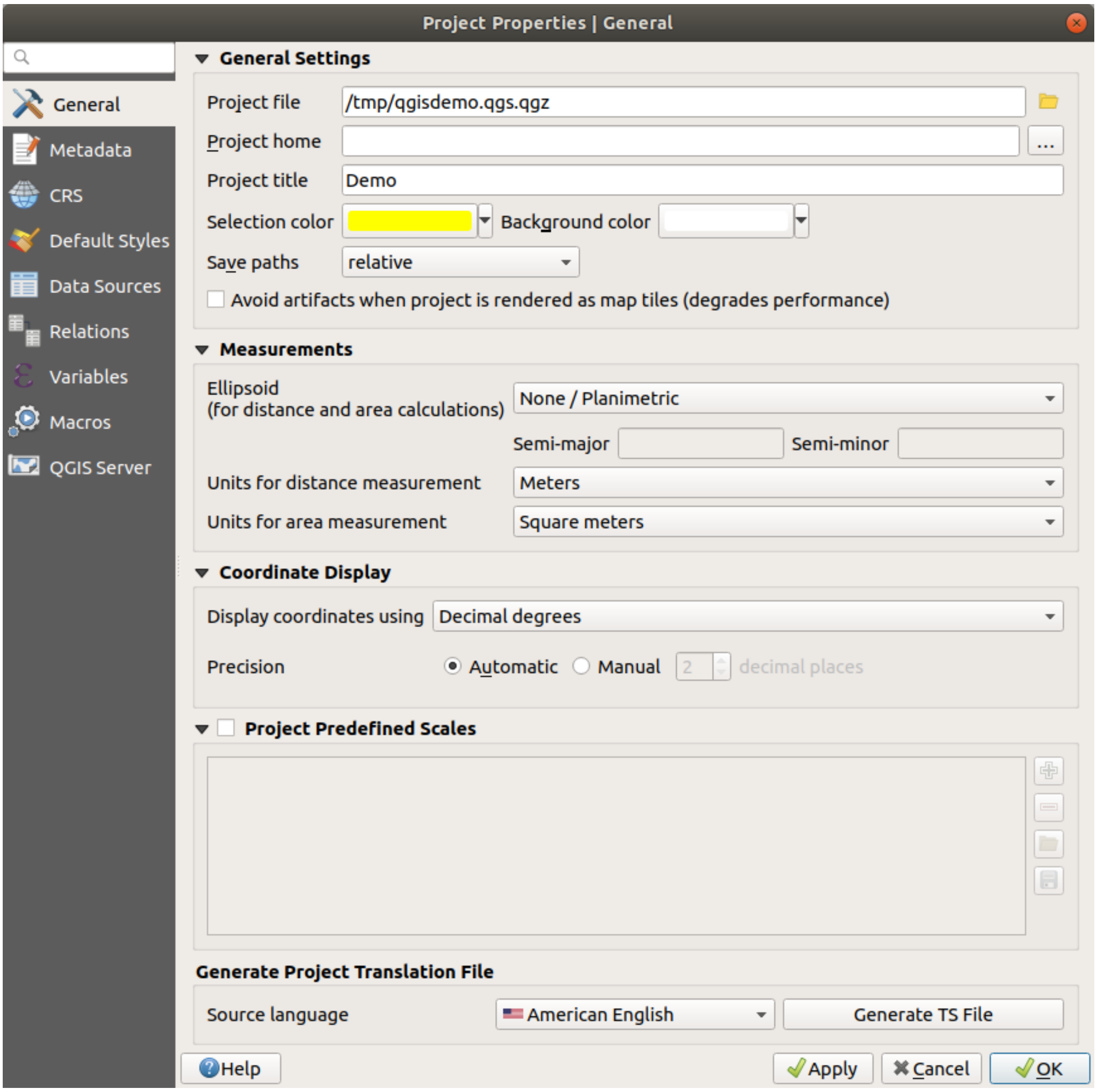


Abb. 9.20: General tab of the Project Properties dialog

9.3.2 Metadata Properties

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 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 projection (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.

The *CRS* tab also 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.

9.3.4 Default Styles Properties

The *Default Styles* tab 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 default symbols (*Marker*, *Line*, *Fill*) to apply depending on the layer geometry type as well as a default *Color Ramp*
- Apply a default *Opacity* to new layers
- *Assign random colors to symbols*, modifying the symbols fill colors, hence avoiding same rendering for all layers.

Using the *Style Manager* button, you can also quickly access the *Style Manager* dialog and configure symbols and color ramps.

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*.

Tipp: Use project colors to quickly assign and update color widgets

Project colors can be refered 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

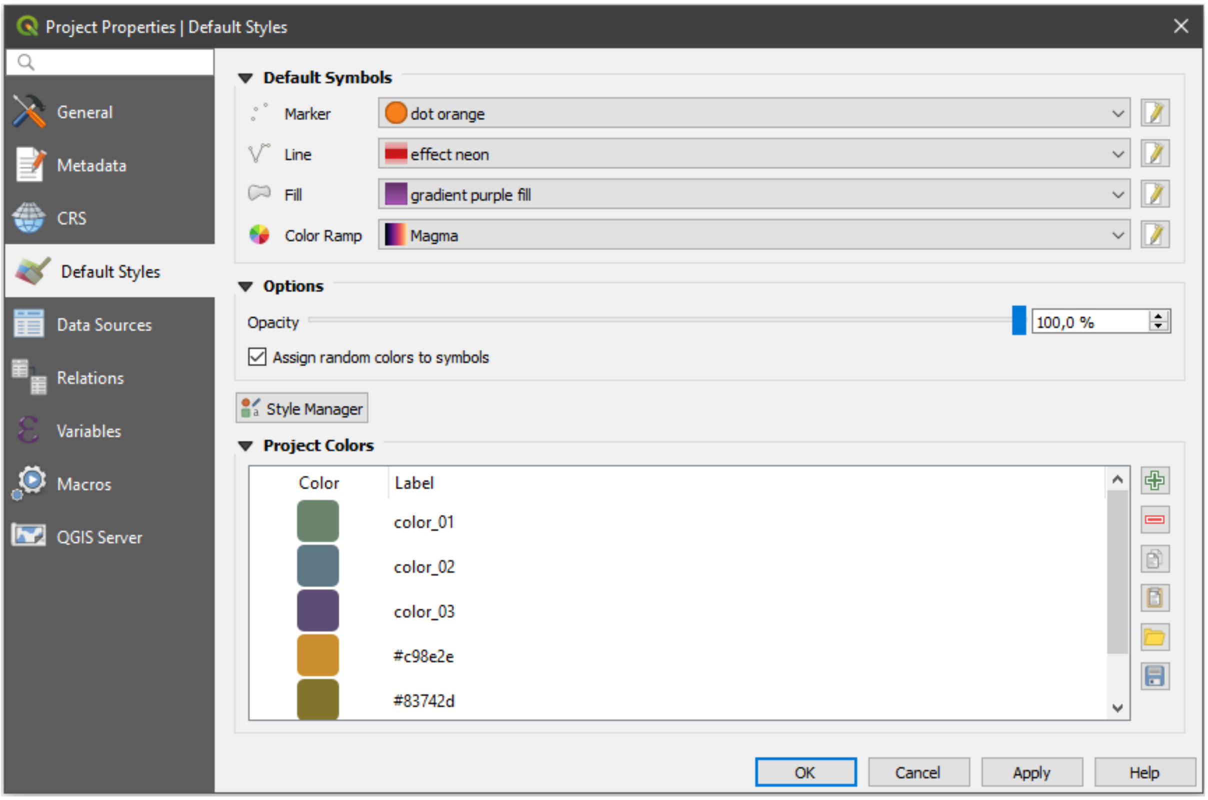


Abb. 9.21: Default Styles tab

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.

9.3.5 Data Sources Properties

In the *Data Sources* tab, you can:

- *Automatically create transaction groups where possible*: When this mode is turned on, all (postgres) layers from the same database are synchronised in their edit state, i.e. when one layer is put into edit state, all are, when one layer is committed or one layer is rolled back, so are the others. Also, instead of buffering edit changes locally, they are directly sent to a transaction in the database which gets committed when the user clicks save layer. Note that you can (de)activate 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`).
- *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.
- 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.

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.

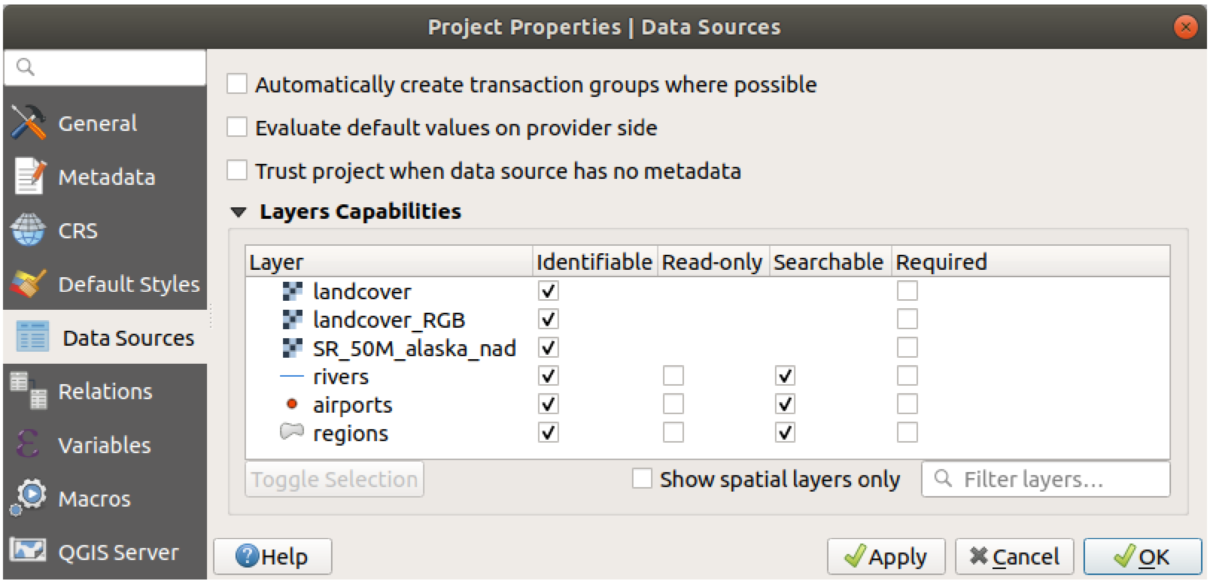


Abb. 9.22: Data Sources tab

9.3.6 Relations Properties

The *Relations* tab is used to define 1:n relations. The relations 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 1:n relations support in Section *Creating one or many to many relations*.

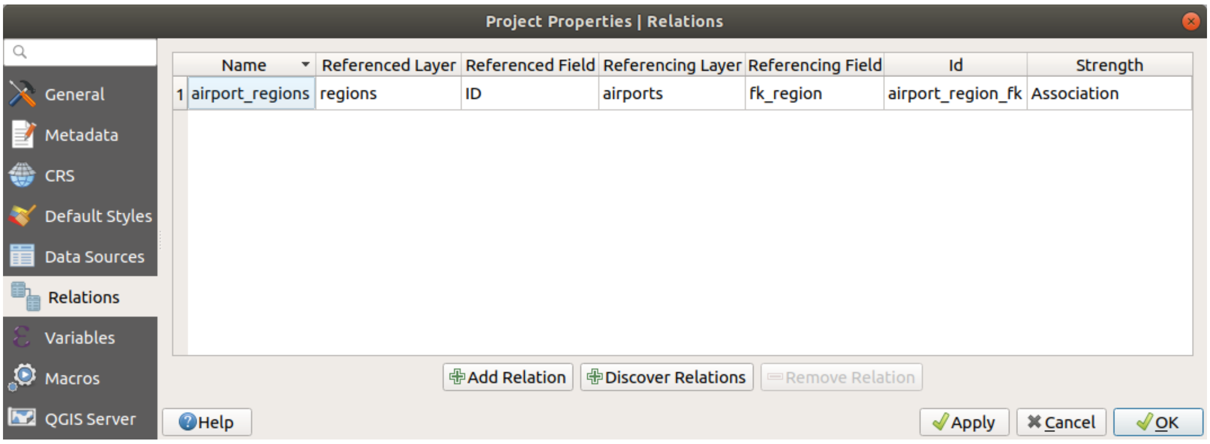


Abb. 9.23: Relations tab

9.3.7 Variables Properties

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.

9.3.8 Macros Properties

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

9.3.9 QGIS Server Properties

The tab *QGIS Server* 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 *Configure your project* and subsequent.

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.

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

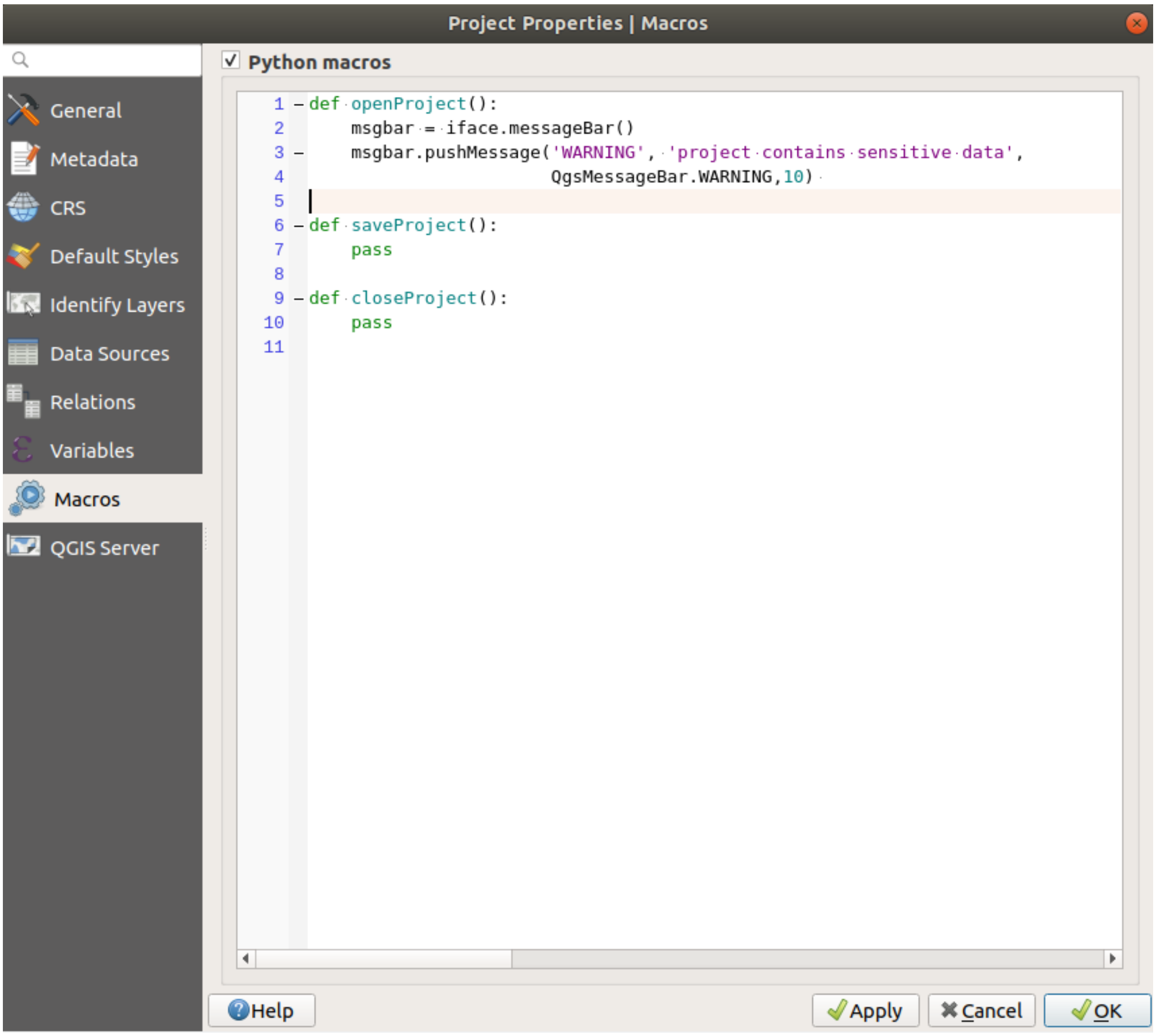


Abb. 9.24: Makroeinstellungen in QGIS

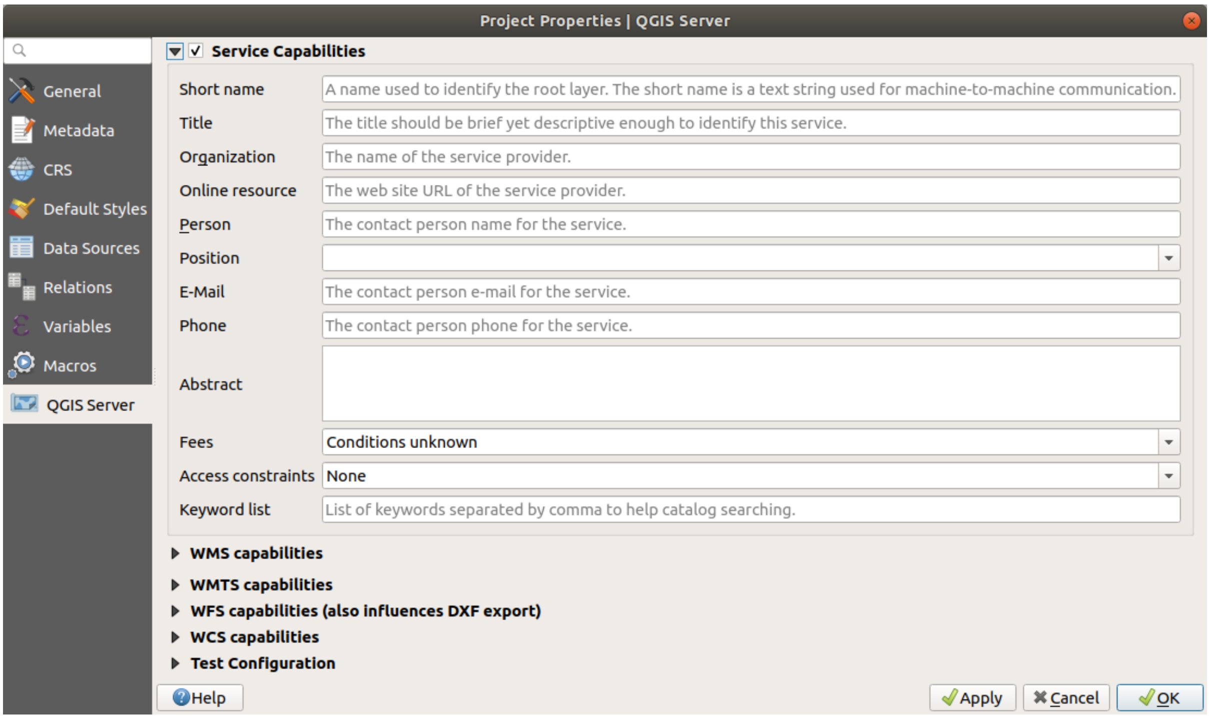


Abb. 9.25: QGIS Server settings tab

- 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 **QSettings** 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.

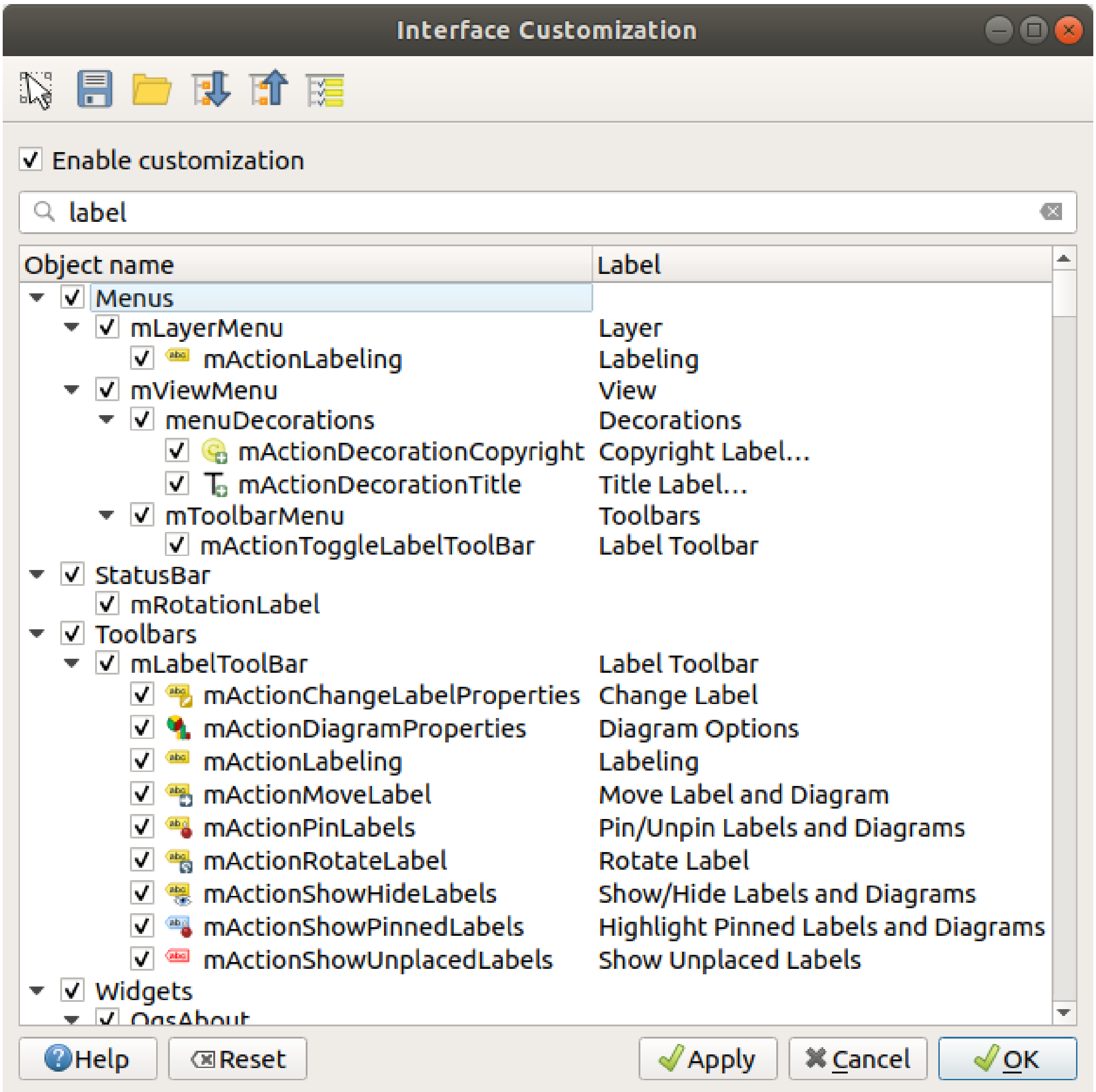


Abb. 9.26: Der Anpassung Dialog

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.

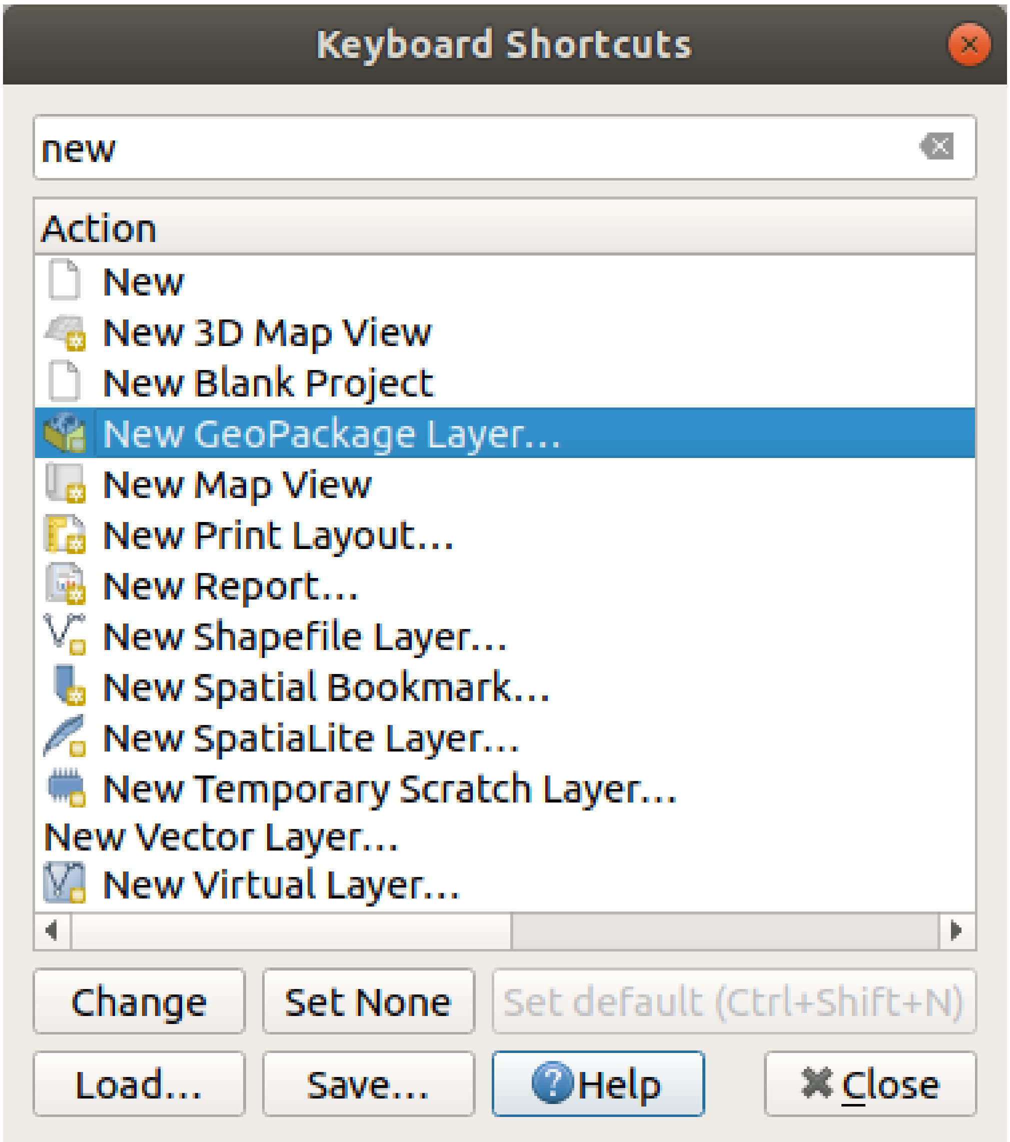


Abb. 9.27: Define shortcut options

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 as an `.XML` file 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:
        [--version]     display version information and exit
        [--snapshot filename]   emit snapshot of loaded datasets to given file
        [--width width] width of snapshot to emit
        [--height height]       height of snapshot to emit
        [--lang language]       use language for interface text (changes existing␣
↪override)
        [--project projectfile] load the given QGIS project
        [--extent xmin,ymin,xmax,ymax]  set initial map extent
        [--nologo]      hide splash screen
        [--noversioncheck]      don't check for new version of QGIS at startup
        [--noplugins]   don't restore plugins on startup
        [--nocustomization]     don't apply GUI customization
        [--customizationfile path]      use the given ini file as GUI customization
        [--globalsettingsfile path]     use the given ini file as Global Settings␣
↪(defaults)
        [--authdbdirectory path] use the given directory for authentication␣
↪database
        [--code path]   run the given python file on load
        [--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|symbollayer|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.
        [--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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        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.

--noversioncheck

Skip searching for a new version of QGIS at startup.

--noplugins

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.

--customizationfile

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

--globalsettingsfile

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. The default global settings is located in `your_QGIS_PKG_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.

The equivalent environment variable is `QGIS_GLOBAL_SETTINGS_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. Unless changed, the selected profile will be used in the following QGIS sessions.

--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_PKG_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"
```

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. A lot of settings can be set using the `INI` file such as WMS/WMTS, PostGIS connections, proxy settings, maptips…

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.

KAPITEL 10

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 OGR or 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*

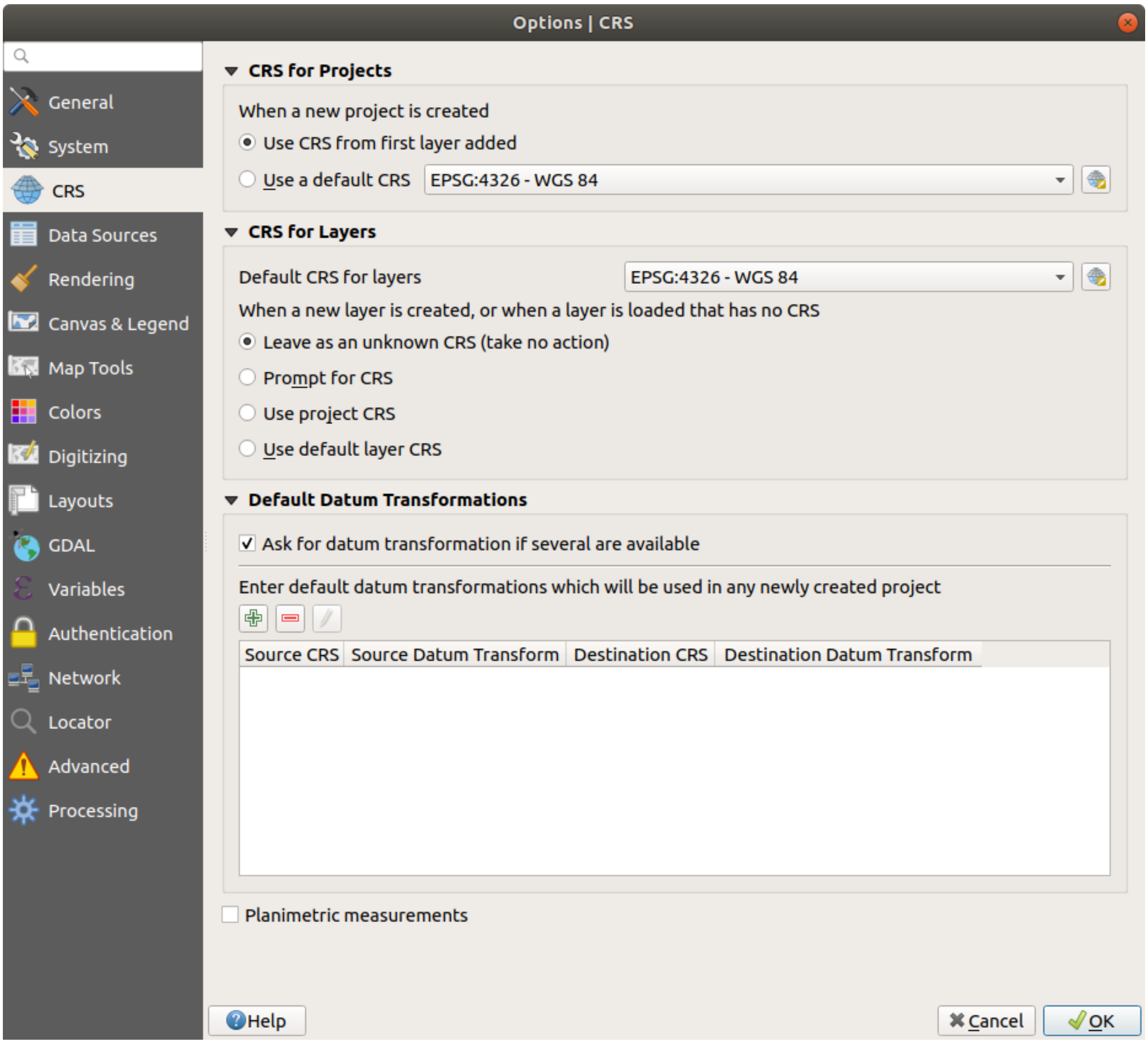


Abb. 10.1: The CRS tab in the QGIS Options Dialog

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, defering 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.

Tipp: 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 *figure_projection_options*). 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.

Available options are:

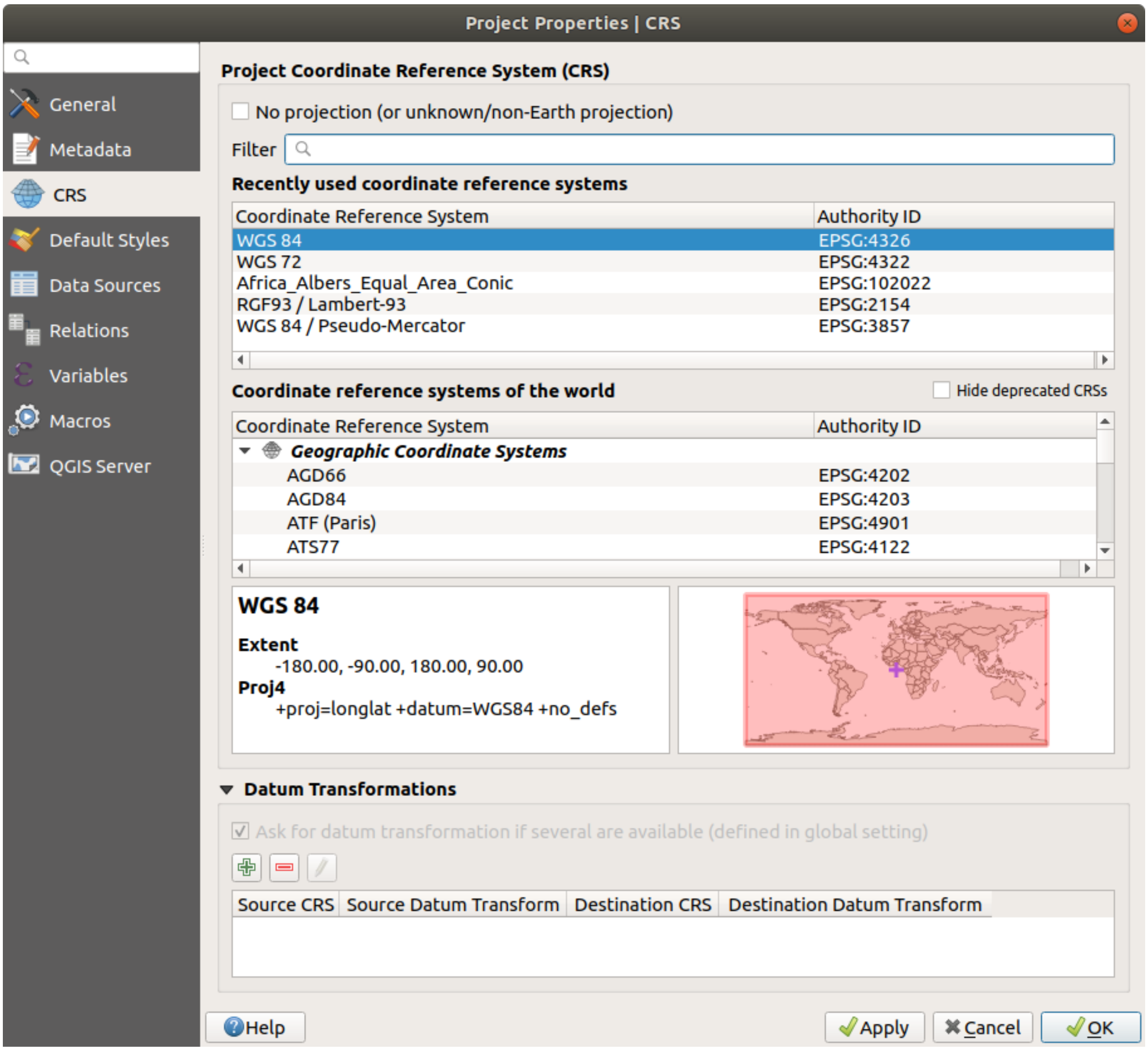


Abb. 10.2: Dialog Projekteigenschaften

- *No projection (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 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

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.

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, click the Add new CRS button, enter a descriptive name, select the format, and the CRS parameters.

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

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…*).

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* ► *CRS* tab menu under the *Default datum transformations* group:

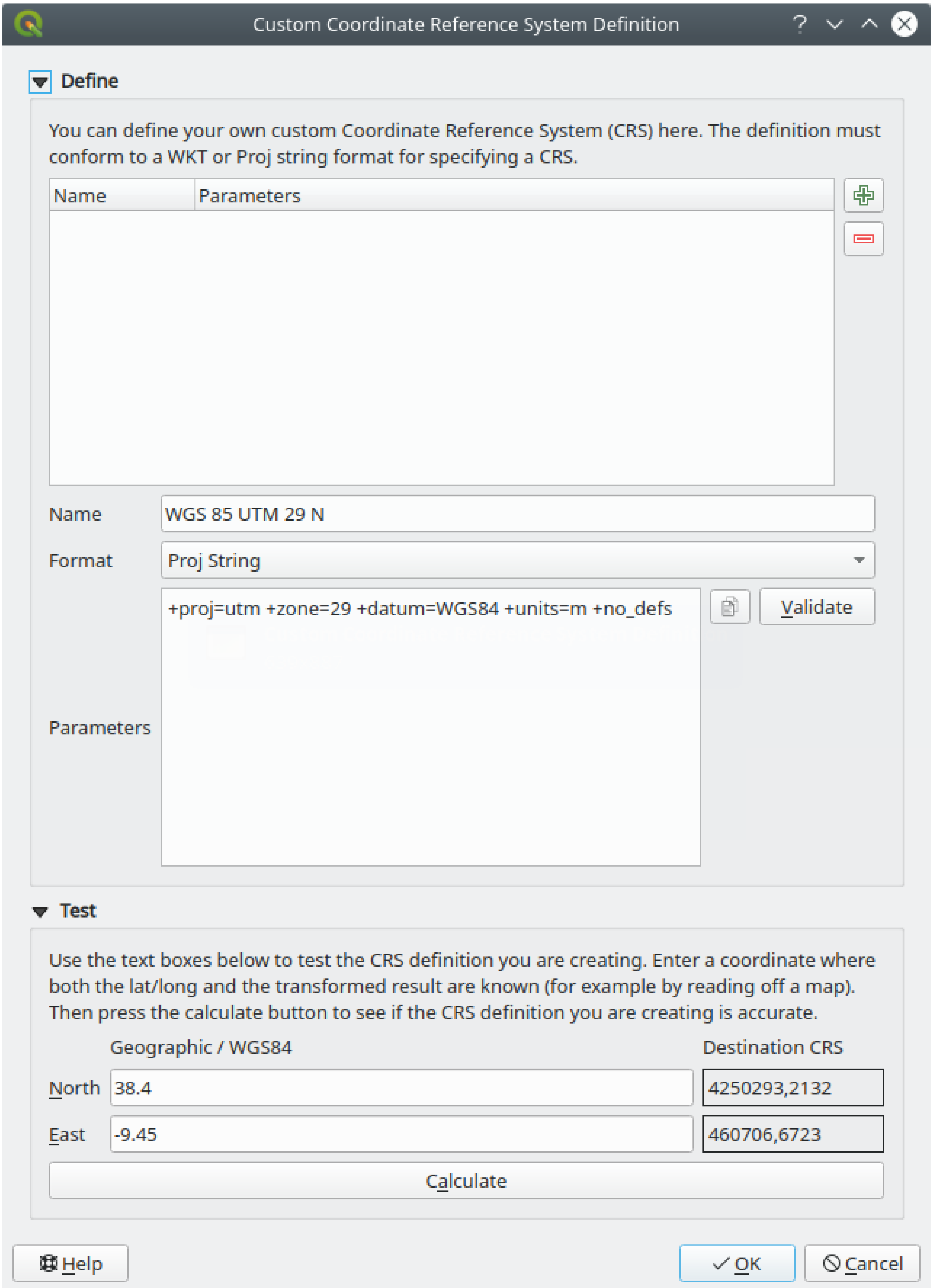


Abb. 10.3: Der Benutzerkoordinatensystem Dialog

Abb. 10.4: Setting an NTv2 transformation

- using *Ask for datum transformation if several are available*: when more than one appropriate datum transformation exists 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 of the transformation.

   In some cases a transformation may not be available for use on your system. In this case, the transformation will still be shown in this list but will not be selectable.
4. Find your preferred transformation, select it and click *OK*.

   A row is added to the table under *Default Datum Transformations* with information about *Source CRS* and *Destination CRS* as well as the datum transformations (*Source Datum Transformation* and *Destination Datum Transformation*) to be used to transform between the CRSs.

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.

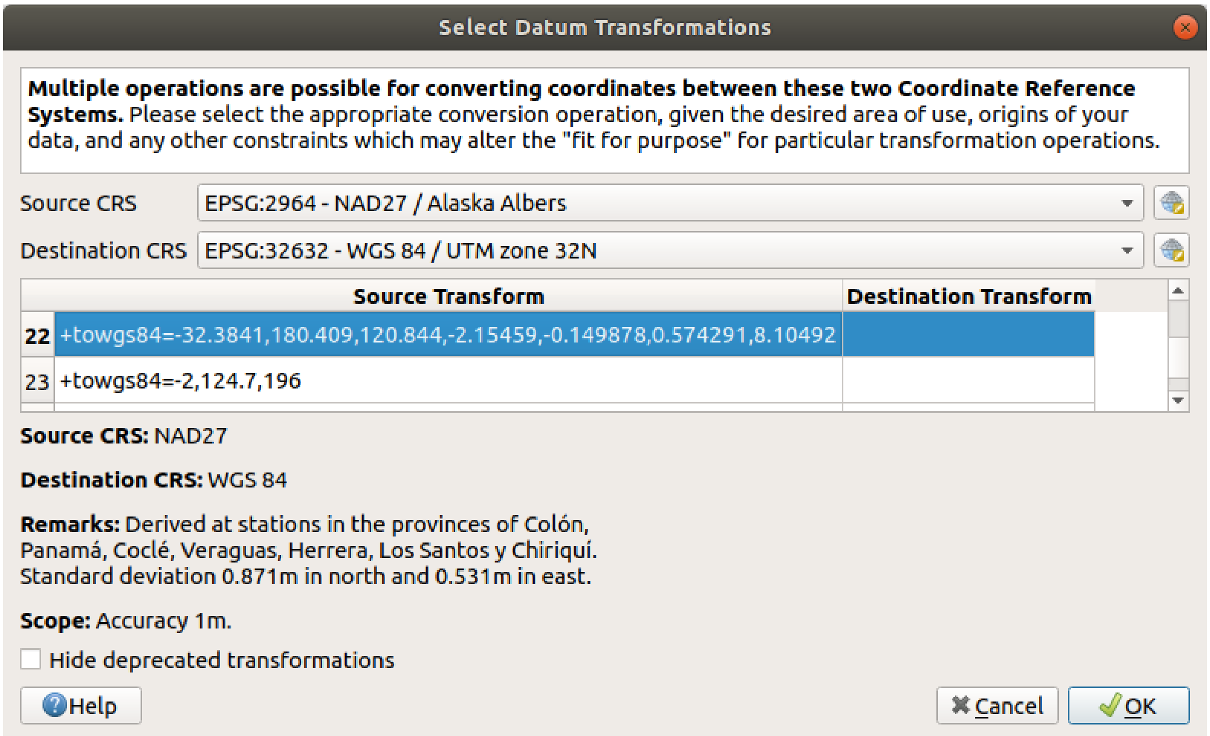


Abb. 10.5: Selecting a preferred default datum transformation

Datum transformations set in the *Settings ▸ Options ▸ CRS* 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.

KAPITEL 11

Allgemeine Werkzeuge

11.1 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.

11.2 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*.

11.2.1 Layerfenster

Das Bedienfeld *Layers* (auch als `Map-Legende` bezeichnet) listet alle Ebenen im Projekt auf und hilft Ihnen, ihre Sichtbarkeit zu verwalten. Sie können es ein- oder ausblenden, indem Sie `Ctrl+1` drücken. Ein Layer kann ausgewählt und in der Legende nach oben oder unten gezogen werden, um die Z-Reihenfolge zu ändern. Z-Reihenfolge bedeutet, dass Schichten, die näher oben in der Legende aufgelistet sind, über Schichten gezeichnet werden, die unten in der Legende aufgelistet sind.

Bemerkung: The Z-ordering behavior can be overridden by the *Layer Order* panel.

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
- Manage Map Themes: control visibility of layers and arrange them in different map themes.
- 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.

- 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. 11.1: Layer Toolbar in Layers Panel

Bemerkung: Tools to manage the layers panel are also available for map and legend items in print layouts

---

Configuring map themes

The Manage Map Themes drop-down button provides access to convenient shortcuts to manipulate visibility of the layers in the *Layers* panel:

- *Alle Layer anzeigen*
- *Alle Layer ausblenden*
- *Show Selected Layers*
- *Hide Selected Layers*
- *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 *symbologies* 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 (symbology, 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. Use the *Remove Current Theme* button to delete the active theme.

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 vector or raster layers added to the project, optionally organized in groups. Depending on the item selected in the panel, a right-click shows a dedicated set of options presented below.

Option	Vektorlayer	Rasterlayer	Group
Zoom to Layer/Group	✓	✓	✓
Zoom to Selection	✓		
Show in Overview	✓	✓	
Show Feature Count	✓		
Copy Layer/Group	✓	✓	✓
Rename Layer/Group	✓	✓	✓
Zoom to Native Resolution (100%)		✓	
Stretch Using Current Extent		✓	
Update SQL Layer…	✓		
Add Group			✓
Duplicate Layer	✓	✓	
Remove Layer/Group…	✓	✓	✓
Move Out of Group	✓	✓	
Move to Top	✓	✓	✓
Check and all its Parents	✓	✓	
Group Selected	✓	✓	
Attributtabelle öffnen	✓		

Fortsetzung auf der nächsten Seite

Tab. 11.1 – Fortsetzung der vorherigen Seite

Option	Vektorlayer	Rasterlayer	Group
Bearbeitungsstatus umschalten	✓		
Aktuelle Änderungen ⍰	✓		
Filter…	✓		
Set Layer Scale Visibility…	✓	✓	
Zoom to Visible Scale	✓	✓	
Set CRS ⍰	✓	✓	
⍰ Set Layer/Group CRS…	✓	✓	✓
⍰ Set Project CRS from Layer	✓	✓	
Set Group WMS Data…			✓
Mutually Exclusive Group			✓
Check and all its children (Ctrl-click)			✓
Uncheck and all its children (Ctrl-click)			✓
Make Permanent	✓		
Export ⍰	✓	✓	✓
⍰ Save As…		✓	
⍰ Save Features As…	✓		
⍰ Save Selected Features As…	✓		
⍰ Save As Layer Definition File…	✓	✓	✓
⍰ Save As QGIS Layer Style File…	✓	✓	
Styles ⍰	✓	✓	
⍰ Copy Style	✓	✓	
⍰ Paste Style	✓	✓	✓
⍰ Add…	✓	✓	
⍰ Rename Current…	✓	✓	
⍰ Edit symbol…	✓		
⍰ Copy Symbol	✓		
⍰ Paste Symbol	✓		
Eigenschaften…	✓	✓	

Table: Context menu from Layers Panel items

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 two 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 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 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, icons appears next to the layer or group in the *Layers* panel to give more information about the layer/group. These symbols 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 setting
- 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. Click the icon to update the source path.
- 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 OGR vector formats supported by QGIS.
- to identify a layer that has no/unknown CRS

Editing vector layer style

From the Layers panel, you have shortcuts to change the layer rendering quickly and easily. Right-click on a vector layer and select *Styles* ▷ in the list in order to:

- see the *styles* currently applied to 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

> **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) as the original layer and, for vector layers, have the same geometry type (point, line or polygon).

- rename the current style, add a new style (which is actually a copy of the current one) or delete the current style (when multiple styles are available).

Bemerkung: The previous options are also available for raster or mesh layers.

- 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…*: open the *Symbol Selector* dialog and change feature symbol (symbol, size, color…).

When using a classification symbology type (based on *categorized*, *graduated* or *rule-based*), the aforementioned symbol-level options are available from the class entry context menu. Also provided are the *Toggle Items*, *Show All Items* and *Hide All Items* entries to switch the visibility of all the classes of features. These avoid (un)checking items one by one.

Tipp: Double-clicking a class leaf entry also opens the *Symbol Selector* dialog.

11.2.2 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:

- Set raster layer *Symbology*, *Transparency*, and *Histogram* properties. These options are the same as those in the *Dialogfenster Rasterlayereigenschaften*.
- Set vector layer *Symbology*, *3D View* and *Labels* properties. These options are the same as those in the *Vektorlayereigenschaften*.
- Manage the associated style(s) in the *Style Manager* (more details at *Managing Custom Styles*).
- 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.

Tipp: Add custom tabs to the Layer Styling panel

Using PyQGIS, you can set new tabs to manage layer properties in the Layer Styling Panel. See https://nathanw.net/2016/06/29/qgis-style-dock-part-2-plugin-panels/ for an example.

11.2.3 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 *figure_layer_order*, 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.

11.2.4 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.

11.2.5 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.

11.2.6 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*.

11.2.7 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
- the column to use, or an *expression*
- 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:

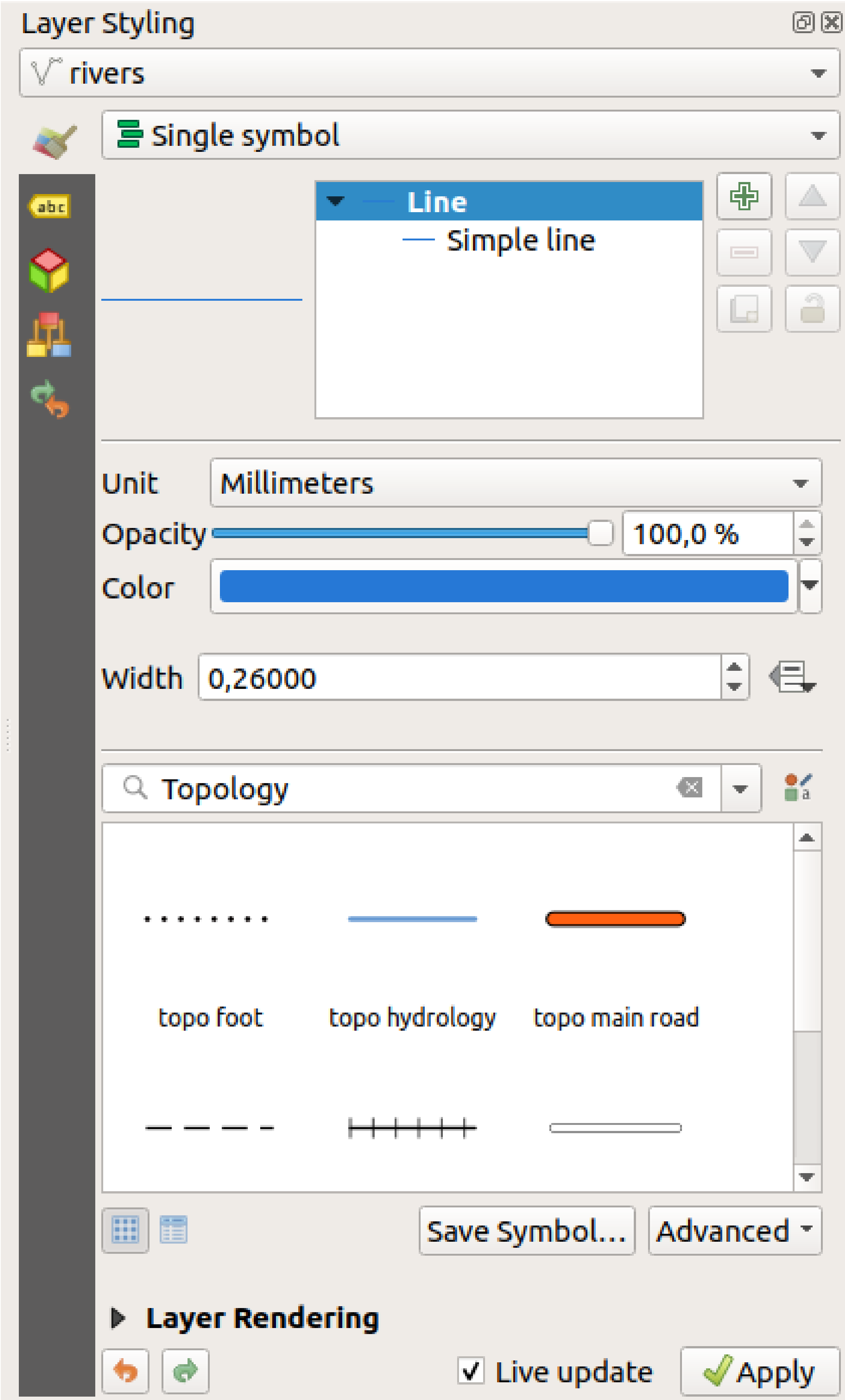


Abb. 11.2: Defining a layer's symbology from the layer styling panel

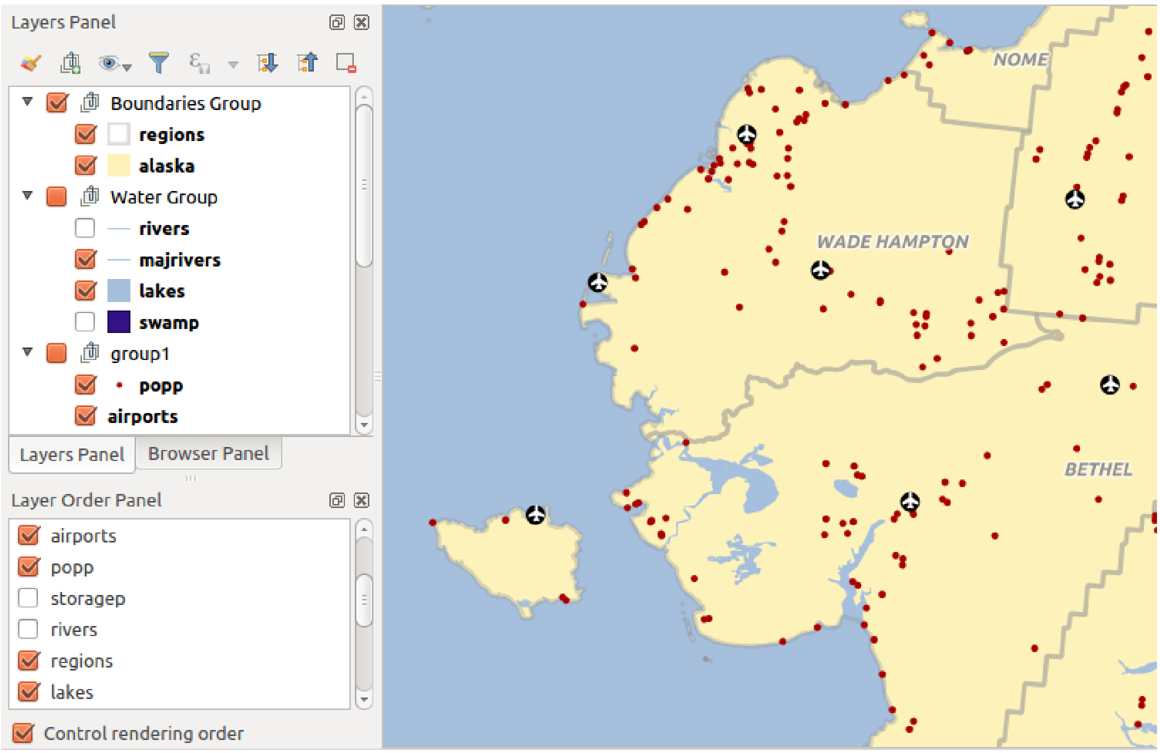


Abb. 11.3: Define a layer order independent of the legend

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	✓			

Table: Statistics available for each field type

The statistical summary can be:

- returned for the whole layer or *selected features only*
- recalculated using the button when the underlying data source changes (eg, new or removed features/fields, attribute modification)
- copied to the clipboard and pasted as a table in another application

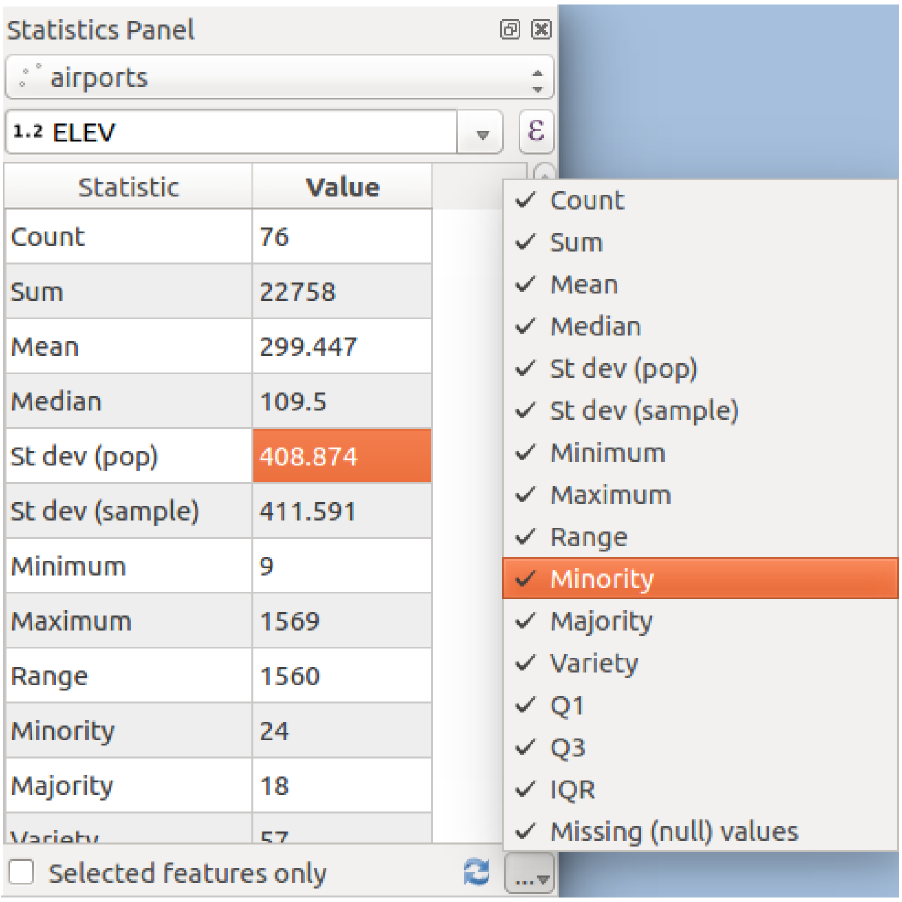


Abb. 11.4: Show statistics on a field

11.3 Layer/Gruppen einbinden

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 *figure_embed_dialog*)
2. Hold down `Ctrl` (or `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.

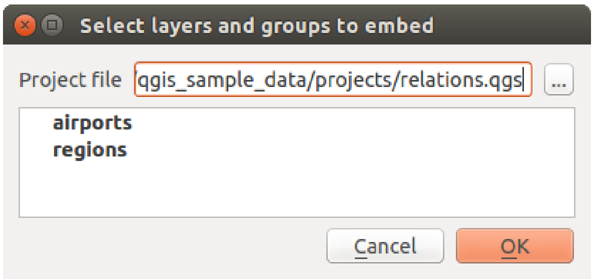


Abb. 11.5: Einzubettende Layer und Gruppen wählen

Like any other layer, an embedded layer can be removed from the project by right-clicking on the layer and clicking Remove.

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.

11.4 Working with the map canvas

11.4.1 Layeranzeige kontrollieren

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

- adding a layer
- panning or zooming
- resizing the QGIS window
- changing the visibility of a layer or layers

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

Maßstabsabhängige Layeranzeige

Scale-dependent rendering allows you to specify the minimum and maximum scales at which a layer (raster or vector) will be visible. To set scale-dependent rendering, open the *Properties* dialog by double-clicking on the layer in the legend. On the *Rendering* tab, tick *Scale dependent visibility* and enter the *Minimum (exclusive)* and *Maximum (inclusive)* scale values.

You can also activate scale dependent visibility on a layer from the Layers panel. Right-click on the layer and in the context menu, select *Set Layer Scale Visibility*.

The Set to current canvas scale button allow you to use the current map canvas scale as boundary of the range visibility.

Bemerkung: When a layer is not rendered in the map canvas because the map scale is out of its visibility scale range, the layer is greyed in the Layers panel and a new option *Zoom to Visible Scale* appears in the layer context menu. Select it and the map is zoomed to the layer's nearest visibility scale.

Layeranzeige kontrollieren

Die Kartendarstellung kann wie unten beschrieben auf verschiedene Art und Weise gesteuert werden.

Wiedergabe unterdrücken

To suspend rendering, click the *Render* checkbox in the bottom-right corner of the status bar. When *Render* is not checked, QGIS does not redraw the canvas in response to any of the events described in the section *Layeranzeige kontrollieren*. 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.

Option für Layer hinzufügen

You can set an option to always load new layers without drawing them. This means the layer will be added to the map, but its visibility checkbox in the legend will be unchecked by default. To set this option, choose menu option *Settings* *Options* and click on the *Rendering* tab. Uncheck *By default new layers added to the map should be displayed*. Any layer subsequently added to the map will be off (invisible) by default.

Zeichnen stoppen

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 take a bit of time between pressing `Esc` for the map drawing to halt.

Die Qualität der Wiedergabe beeinflussen

Um die Wiedergabequalität der Layer zu beeinflussen gibt es drei Möglichkeiten. Öffnen Sie den Reiter *Darstellung* im Menü *Einstellungen* *Optionen* und aktivieren bzw. deaktivieren Sie *Linien auf Kosten der Zeichengeschwindigkeit weniger gezackt zechnen*.

Darstellung beschleunigen

Es gibt zwei Einstellungen die es Ihnen ermöglichen die Darstellungsgeschwindigkeit zu erhöhen. Öffnen Sie den QGIS *Optionen* Dialog indem Sie *Einstellungen ⍰ Optionen* benutzen, gehen Sie zum *Darstellung* Menü und aktivieren oder deaktivieren Sie die folgenden Kontrollkästchen:

- *Use render caching where possible to speed up redraws*.
- *Layer mit vielen CPU-Kernen parallel zeichnen* und legen Sie dann die *Max. zu benutzende Kerne* fest.
- Die Karte rendert im Hintergrund auf einem separaten Bild und jedes *Map Update interval*, der Inhalt von diesem (Off-Screen) Bild wird durch die sichtbare Bildschirmdarstellung getroffen werden, um aktuell zu bleiben. Wenn das Rendering jedoch schneller ist, als diese Dauer beendet ist, wird es sofort angezeigt.
- With *Enable Feature simplification by default for newly added layers*, you simplify features' geometry (fewer nodes) and as a result, they display more quickly. Be aware that this can cause rendering inconsistencies.

11.4.2 Zoomen und Karte verschieben

QGIS bietet Werkzeuge zum zoomen oder verschieben des Kartenfensters zu einem Ausschnitt Ihrer Wahl.

Apart from using the pan and zoom-in/ zoom-out icons on the toolbar with the mouse, you can also navigate with the mouse wheel, spacebar and arrow keys. A *Zoom factor* can be set under the *Settings ⍰ Options ⍰ Map tools* menu to define the scale behavior while zooming.

With the mouse wheel

You can press the mouse wheel to pan inside of the main window (on macOS, you may need to hold down the `cmd` key). You can roll the mouse wheel to zoom in and out on the map; the mouse cursor position will be the center of the zoomed area of interest. Holding down `Ctrl` while rolling the mouse wheel results in a finer zoom.

With the arrow keys

Panning the map is possible with the arrow keys. Place the mouse cursor inside the map area, and click on the arrow keys to pan up, down, left and right.

You can also use the space bar to cause mouse movements temporarily to pan the map. The `PgUp` and `PgDown` keys on your keyboard will cause the map display to zoom in or out following the zoom factor set. Pressing `Ctrl++` or `Ctrl+-` also performs an immediate zoom in/out on the map canvas.

When certain map tools are active (Identify, Measure…), you can perform a zoom by holding down `Shift` and dragging a rectangle on the map to zoom to that area. This is not enabled for selection tools (since they use `Shift` for adding to selection) or edit tools.

11.4.3 Räumliche Lesezeichen

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.

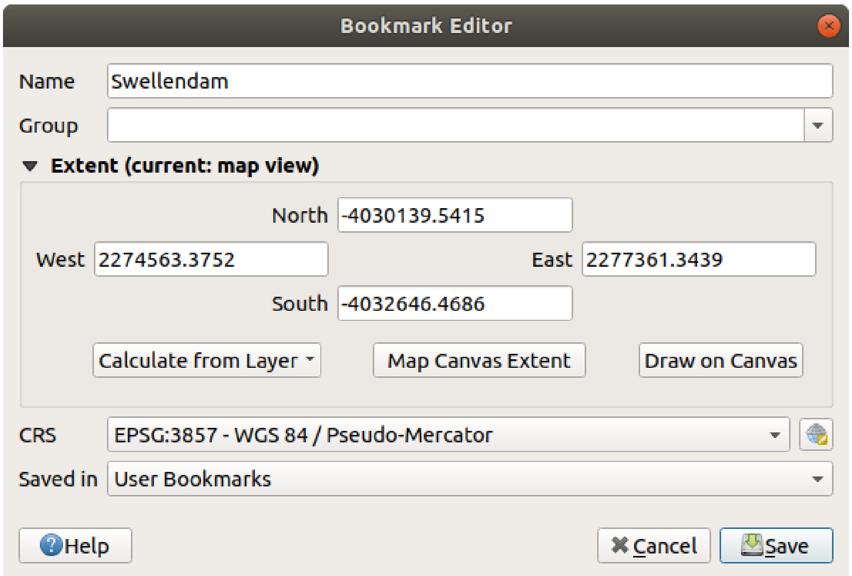


Abb. 11.6: The Bookmark Editor Dialog

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; the extent can be calculated from a loaded layer extent, the current map canvas or drawn over the current map canvas.
6. Indicate the *CRS* to use for the extent
7. Select whether the bookmark will be *Saved in User Bookmarks* or *Project Bookmarks*
8. 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:

Task	Spatial Bookmark Manager	Browser
Zoom to a Bookmark	Double-click on it, or select the bookmark and press the *Zoom to bookmark* button.	Double-click on it, drag and drop it to the map canvas, or right-click the bookmark and select *Zoom to Bookmark*.
Delete a bookmark	Select the bookmark and click the *Delete bookmark* button. Confirm your choice.	Right-click the bookmark and select *Delete Spatial Bookmark*. Confirm your choice.
Export bookmarks to XML	Click the *Import/Export Bookmarks* button and select *Export*. 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 *Export Spatial Bookmarks…*. The selected bookmark subset is saved.
Import bookmarks from XML	Click the *Import/Export Bookmarks* button and select *Import*. All bookmarks in the XML file are imported as user bookmarks.	Right-click the *Spatial Bookmarks* entry or one of its folders (user or project) or subfolders (groups) to determine where to import the bookmarks, then select *Import Spatial Bookmarks*. If performed on the *Spatial Bookmarks* entry, the bookmarks are added to *User Bookmarks*.
Edit bookmark	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 *Edit Spatial Bookmark…*. The *Bookmark Editor* 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 also zoom to bookmarks by typing the bookmark name in the locator.

11.4.4 Dekorationen

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.
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:

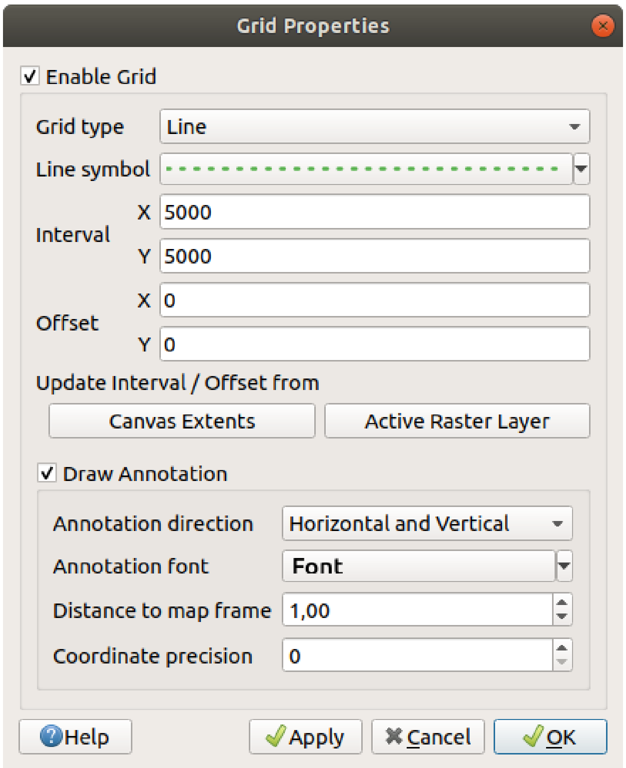


Abb. 11.7: Der Gitter Dialog

- 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* using the OS *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

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.
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 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*.

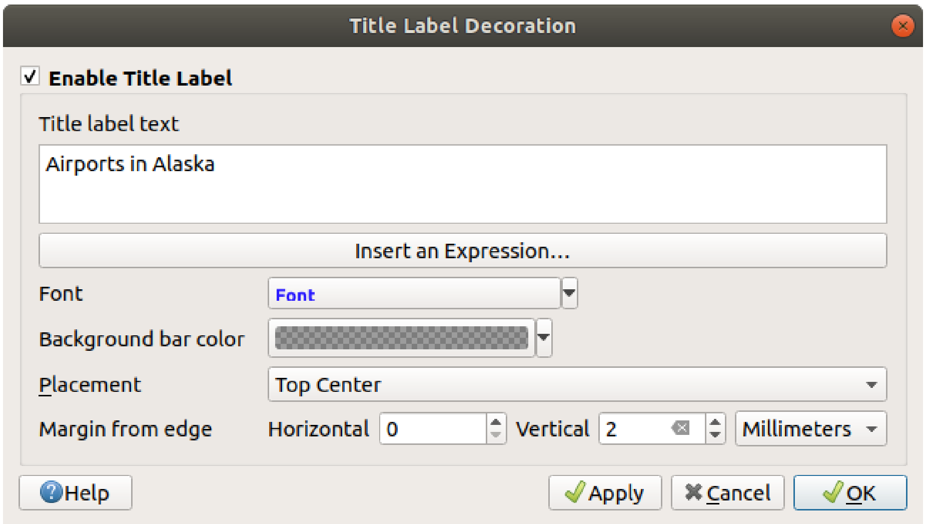


Abb. 11.8: The Title Decoration Dialog

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.

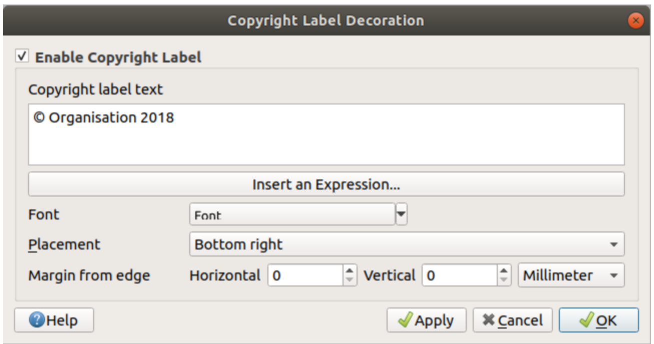


Abb. 11.9: The Copyright Decoration Dialog

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 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.

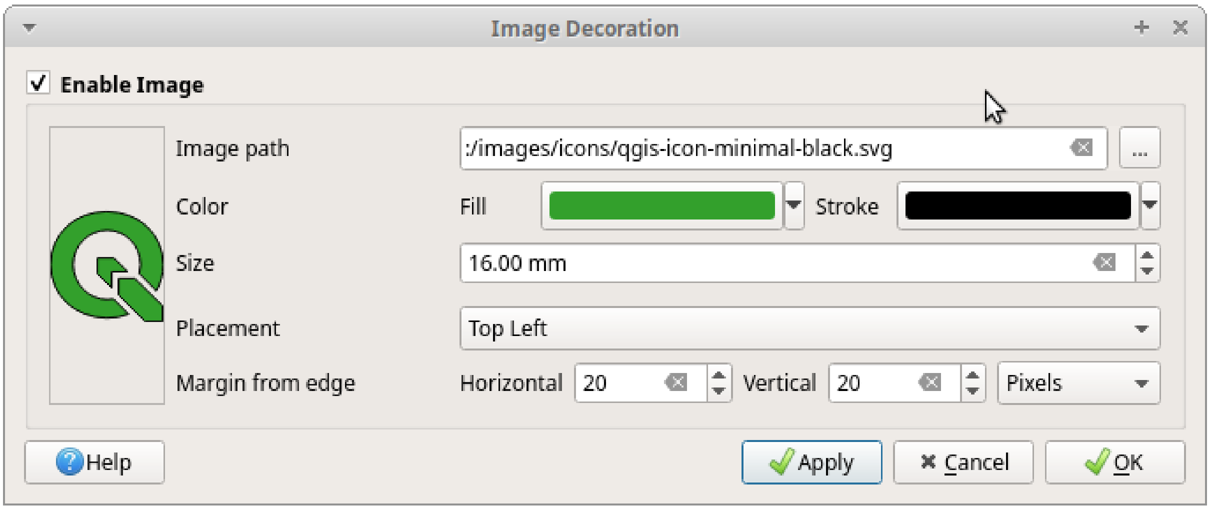


Abb. 11.10: The Image Decoration Dialog

2. Make sure *Enable Image* is checked
3. Select a bitmap (e.g. png or jpg) or SVG image using the … Browse 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.
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

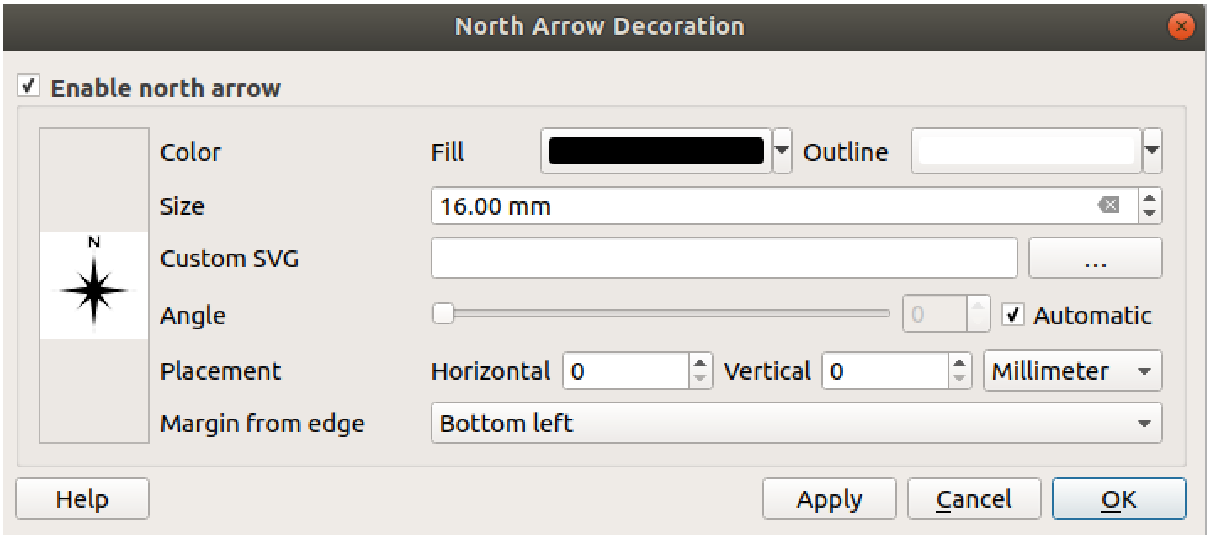


Abb. 11.11: Der Nordpfeil Dialog

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.

QGIS only supports displaying the scale in the same units as your map frame. So, if the units of your project's CRS are meters, you can't create a scale bar in feet. Likewise, if you are using decimal degrees, you can't create a scale bar to display distance in meters.

Das Hinzufügen eines Maßstabbalkens geht folgendermaßen:

1. Select menu option *View ⍰ Decorations ⍰ Scale Bar…* to open the 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* 1,00
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.

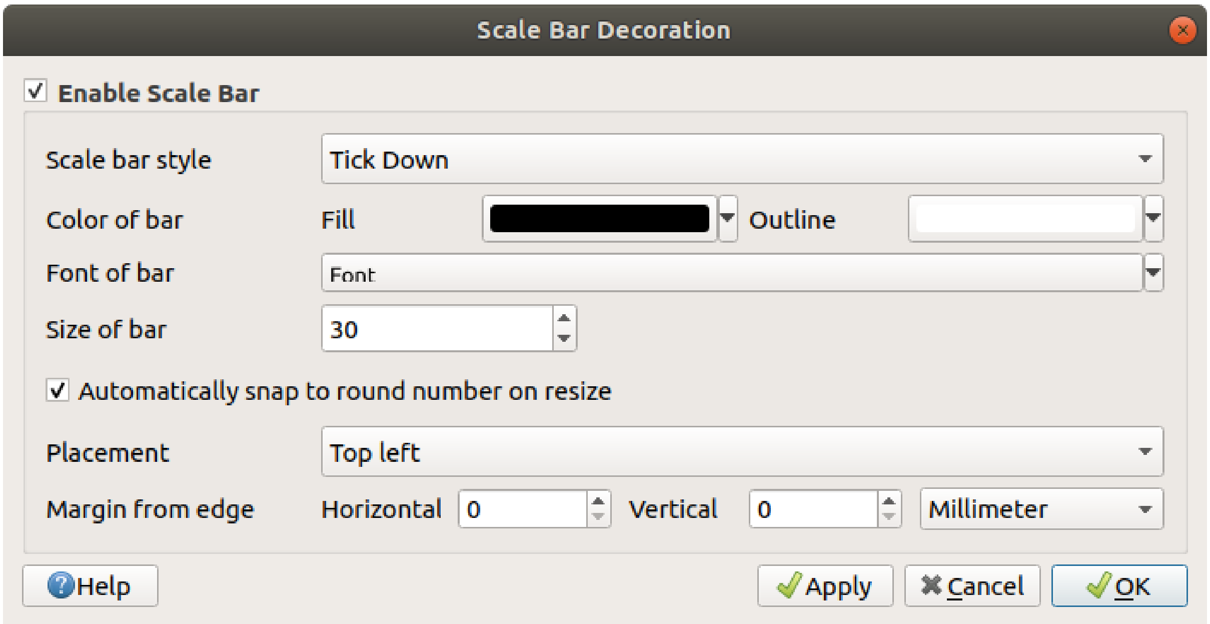


Abb. 11.12: Der Maßstab Dialog

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.

To add layout extent(s):

1. Select *View* ⍰ *Decorations* ⍰ *Layout Extents* to open the 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.

11.4.5 Beschriftungstools

Annotations are information added to the map canvas and shown within a balloon. This information can be of different types and annotations are added using the corresponding tools in the *Attributes Toolbar*:

- Text Annotation for custom formatted text
- HTML Annotation to place the content of an `html` file
- SVG Annotation to add an `SVG` symbol

Abb. 11.13: 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.

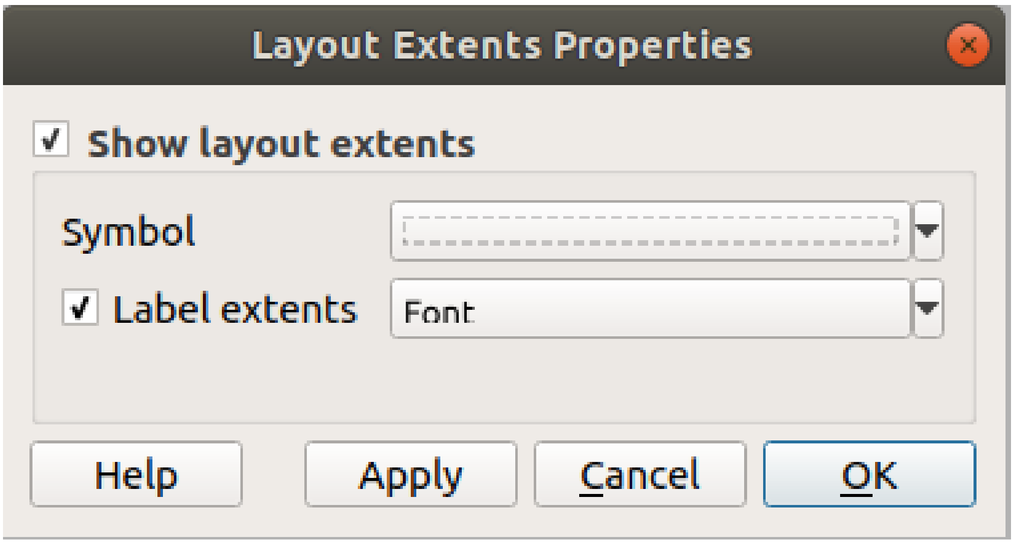


Abb. 11.14: The Layout Extents Dialog

- Form Annotation: useful to display attributes of a vector layer in a customized `ui` file (see *figure_custom_annotation*). 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.

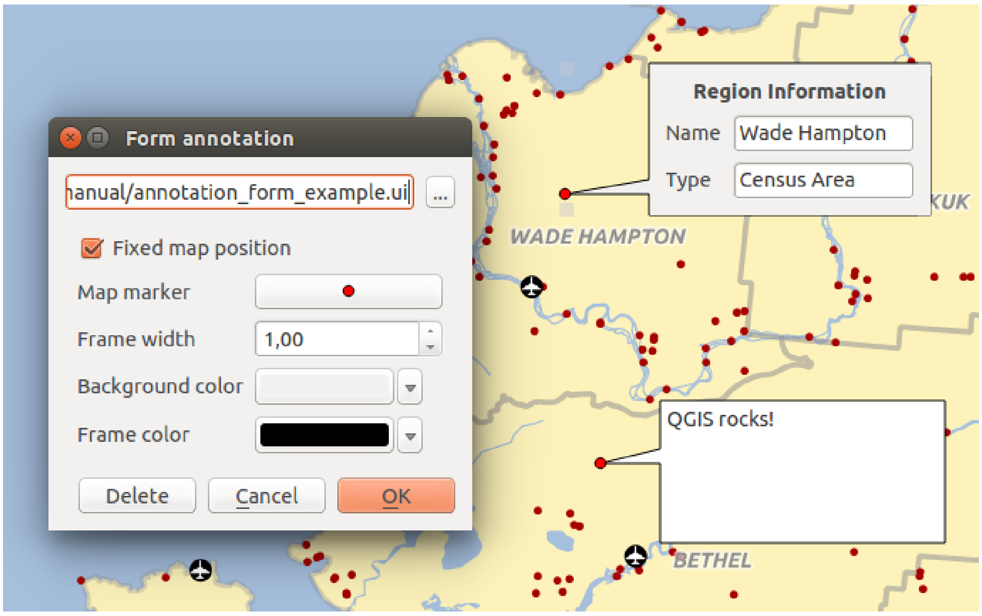


Abb. 11.15: Customized QT Designer annotation form

To add an 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.

Annotations can be selected when an annotation tool is enabled. They can then be moved by map position (by dragging the map marker) or by moving only the balloon. The Move Annotation tool also allows you to move the balloon on the map canvas.

To 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.

Bemerkung: If you press `Ctrl+T` while an *Annotation* tool (move annotation, text annotation, form annotation) is active, the visibility states of the items are inverted.

Tipp: Layout the map with annotations

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

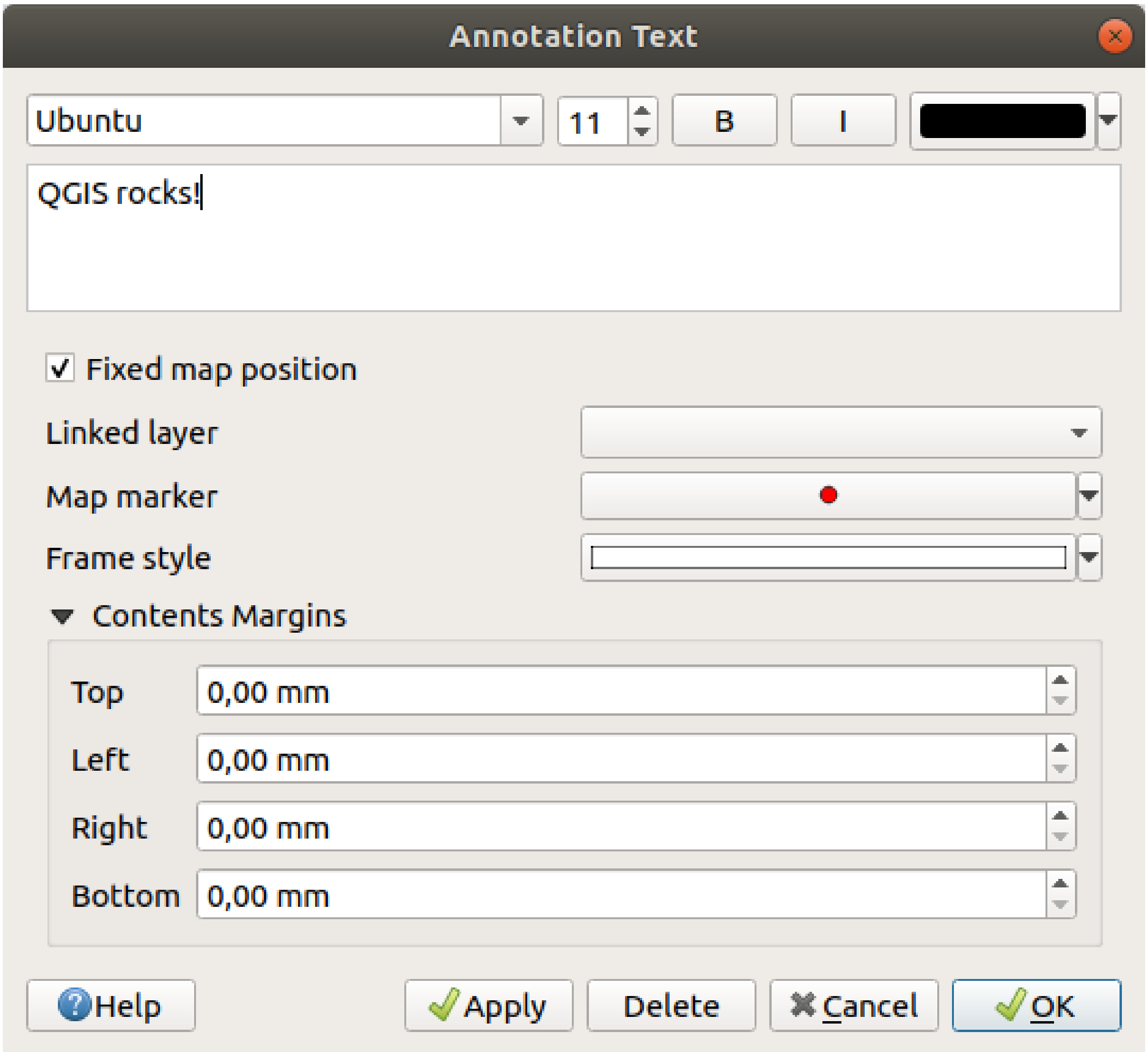


Abb. 11.16: Anmerkungstext Dialog

- 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

11.4.6 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 and angles interactively

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

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 *Einstellen der Fangtoleranz und des Suchradius*). 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.

By default, Measure Line measures real distances between given points according to a defined ellipsoid. 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. 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 (‚Meters', ‚Kilometers', ‚Feet', ‚Yards', ‚Miles', ‚Nautical miles', ‚Centimeters', ‚Millimeters', ‚Degrees', ‚Map units'). 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.

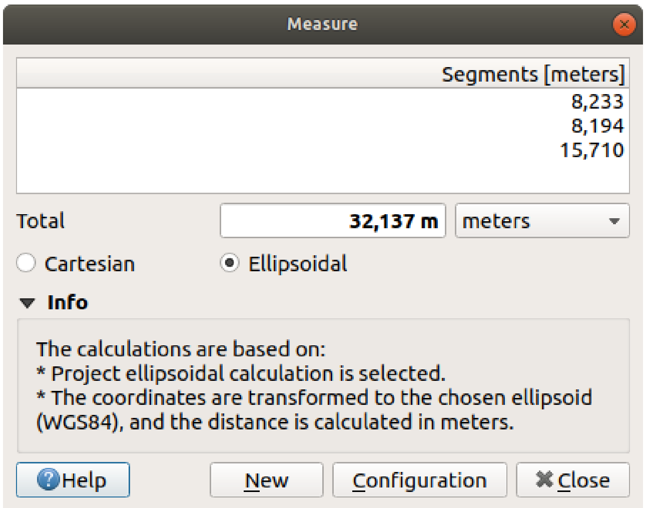


Abb. 11.17: Entfernung messen

 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 (‚Square meters', ‚Square kilometers', ‚Square feet', ‚Square yards', ‚Square miles', ‚Hectares', ‚Acres', ‚Square centimeters', ‚Square millimeters', ‚Square nautical miles', ‚Square degrees', ‚Map units').

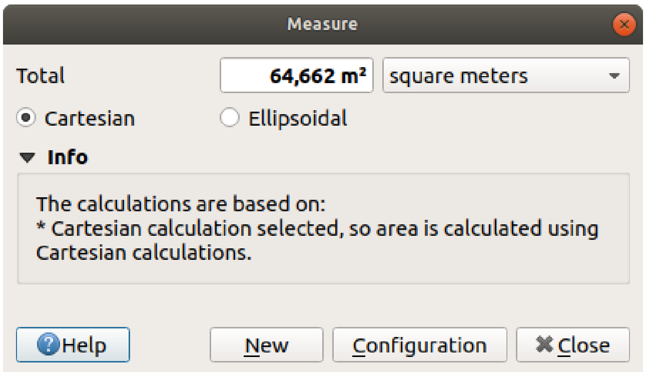


Abb. 11.18: Fläche messen

 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.

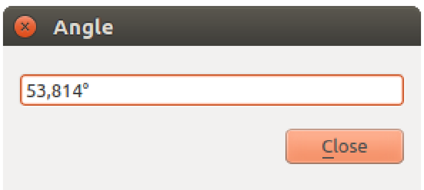


Abb. 11.19: Winkel messen

11.5 Interacting with features

11.5.1 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 *Attributes 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 to select overlapping features. 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`: substract 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 substract 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+Shift+A` to deselect all selected features in all layers
- 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

For example, if you want to find regions that are boroughs from `regions.shp` of the QGIS sample data, you can:

1. Use the Select features using an Expression 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.

Alongside each field, there is a drop-down list with options to control the search behaviour:

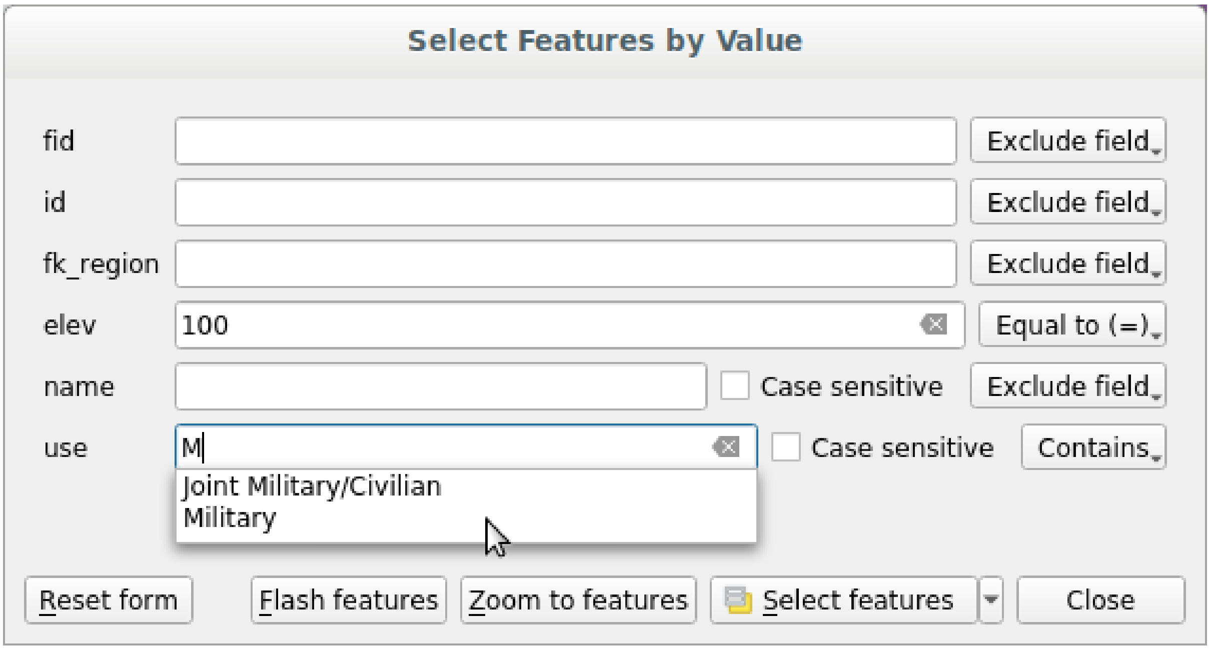


Abb. 11.20: Filter/Select features using form dialog

Field search option	Text	Numeric	Datum
Exclude Field from the search	✓	✓	✓
Equal to (=)	✓	✓	✓
Not equal to (≠)	✓	✓	✓
Greater than (>)		✓	✓
Less than (<)		✓	✓
Greater than or equal to (≥)		✓	✓
Less than or equal to (≤)		✓	✓
Between (inclusive)		✓	✓
Not between (inclusive)		✓	✓
Contains	✓		
Does not contain	✓		
Is missing (null)	✓	✓	✓
Is not missing (not null)	✓	✓	✓
Starts with	✓		
Ends with	✓		

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.

11.5.2 Identifying Features

The Identify tool allows you to interact with the map canvas and get information on features in a pop-up window. To identify features, use:

- *View ▸ Identify Features*
- Ctrl+Shift+I (or Cmd+Shift+I),
- Identify Features icon on the Attributes toolbar

Using the Identify Features tool

QGIS offers several ways to identify features with the Identify Features tool:

- **left click** identifies features according to the *selection mode* and the *selection mask* set in the *Identify Results* panel
- **right click** with *Identify Feature(s)* as *selection mode* set in the *Identify Results* panel fetches all snapped features from all visible layers. This opens a context menu, allowing the user to choose more precisely the features to identify or the action to execute on them.
- **right click** with *Identify Features by Polygon* as *selection mode* in the *Identify Results* panel identifies the features that overlap with the chosen existing polygon, according to the *selection mask* set in the *Identify Results* panel

Tipp: Filter the layers to query with the Identify Features tool

Under *Layer Capabilities* in *Project ▸ Properties… ▸ Data Sources*, uncheck the *Identifiable* column next to a layer to avoid it being queried when using the Identify Features tool in a mode other than **Current Layer**. This is a handy way to return features from only layers that are of interest for you.

If you click on feature(s), the *Identify Results* dialog will list information about the feature(s) clicked. 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.

Feature information

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 *Actions Properties*).
- **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
    * 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.

Bemerkung: Links in the feature's attributes are clickable from the *Identify Results* panel and will open in your default web browser.

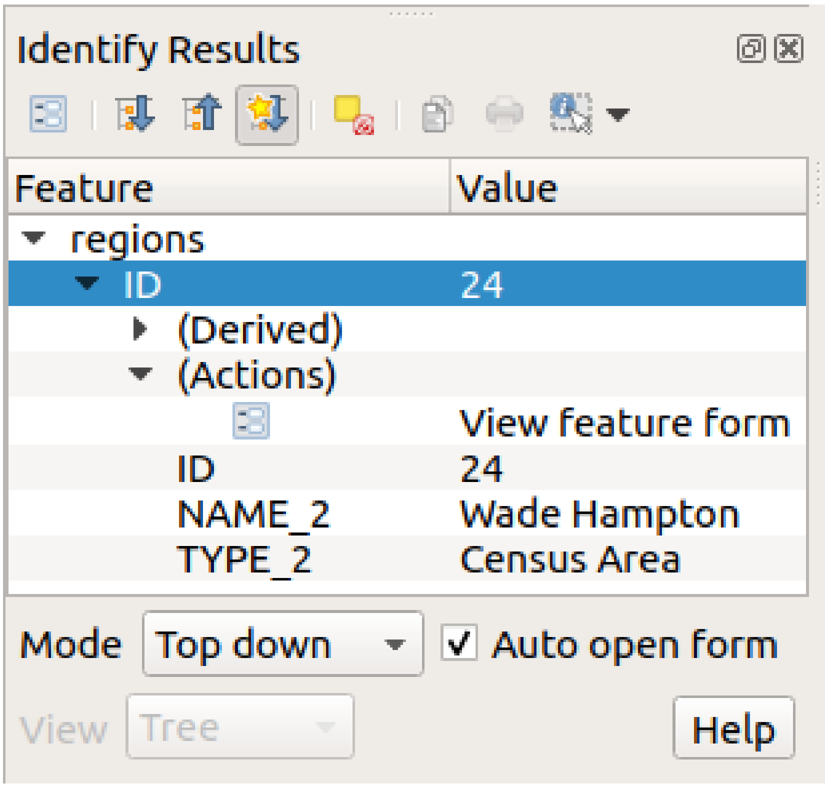


Abb. 11.21: Identify Results dialog

The Identify Results dialog

At the top of the window, you have a handful of tools:

- 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 selected feature to clipboard
- Print selected HTML response
- selection mode to use to fetch features to identify:
  - Identify Features by area or single click
  - Identify Features by Polygon
  - Identify Features by Freehand
  - Identify Features by Radius

Bemerkung: When using Identify Features by Polygon, you can right-click any existing polygon and use it to identify overlapping features in another layer.

At the bottom of the window are the *Mode* and *View* combo boxes. *Mode* defines from which layers features should be identified:

- **Current layer**: only features from the selected layer are identified. The layer need not be visible in the canvas.
- **Top down, stop at first**: only features from the upper visible layer.
- **Top down**: all features from the visible layers. The results are shown in the panel.
- **Layer selection**: opens a context menu where the user selects the layer to identify features from, similar to a right-click. Only the chosen features will be shown in the result panel.

The *View* can be set as **Tree**, **Table** or **Graph**. ‚Table' and ‚Graph' views can only be set for raster layers.

The identify tool allows you to *Auto open form for single feature results*, found under Identify Settings. 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.

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

- 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

11.6 Save and Share Layer Properties

11.6.1 Managing Custom Styles

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.

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.

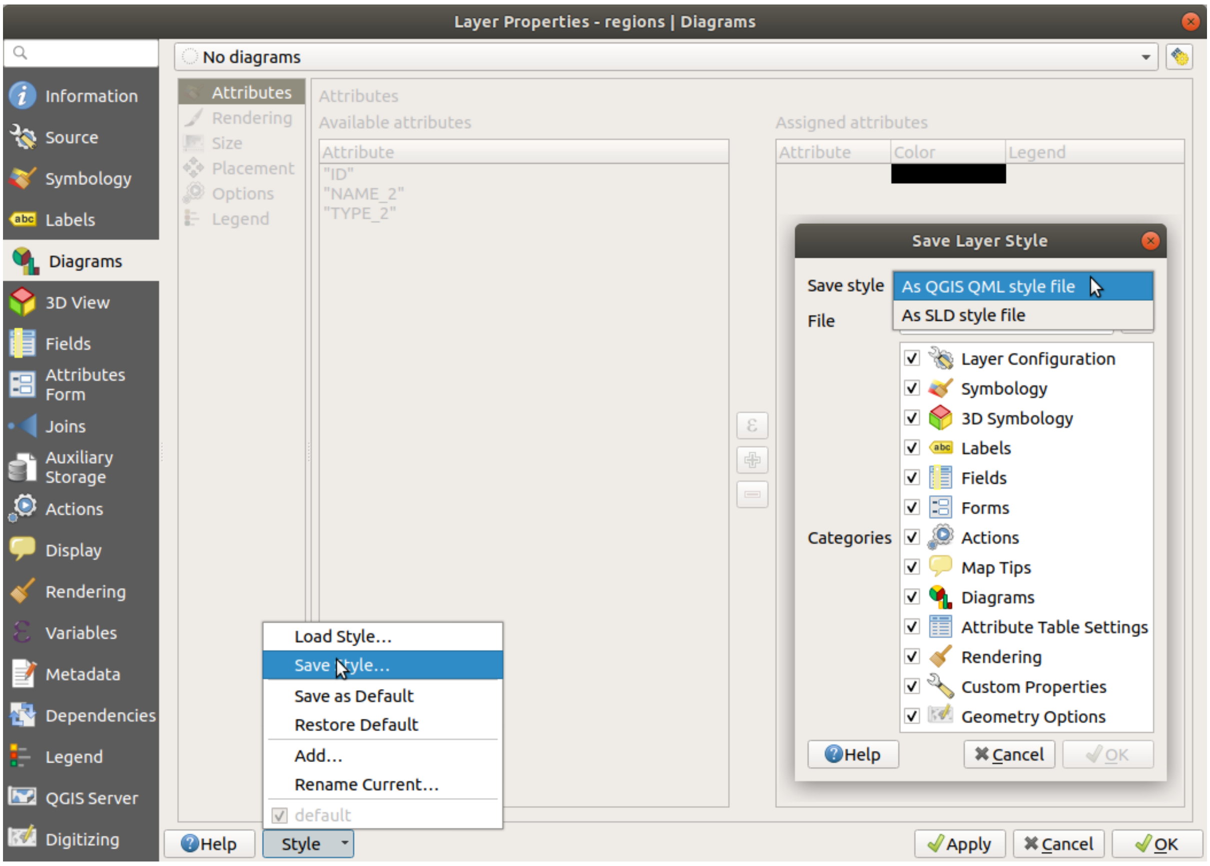


Abb. 11.22: Vector layer style combo box options

11.6.2 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 (`.sld`), 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, MSSQL 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 the QGIS local database, a SQLite database in the `~/.qgis2/` directory (where QGIS stores its local settings).

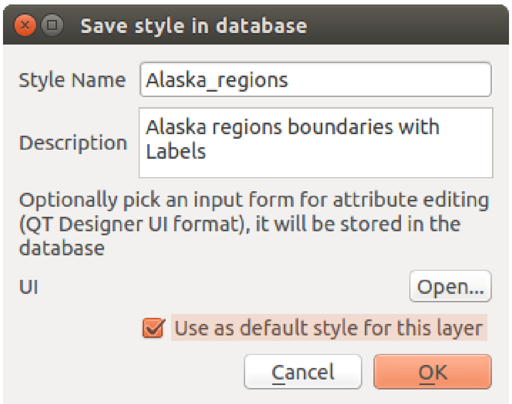


Abb. 11.23: Save Style in database Dialog

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 MSSQL 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).

11.6.3 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.

11.7 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, 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*);
- **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.

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.

11.8 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.

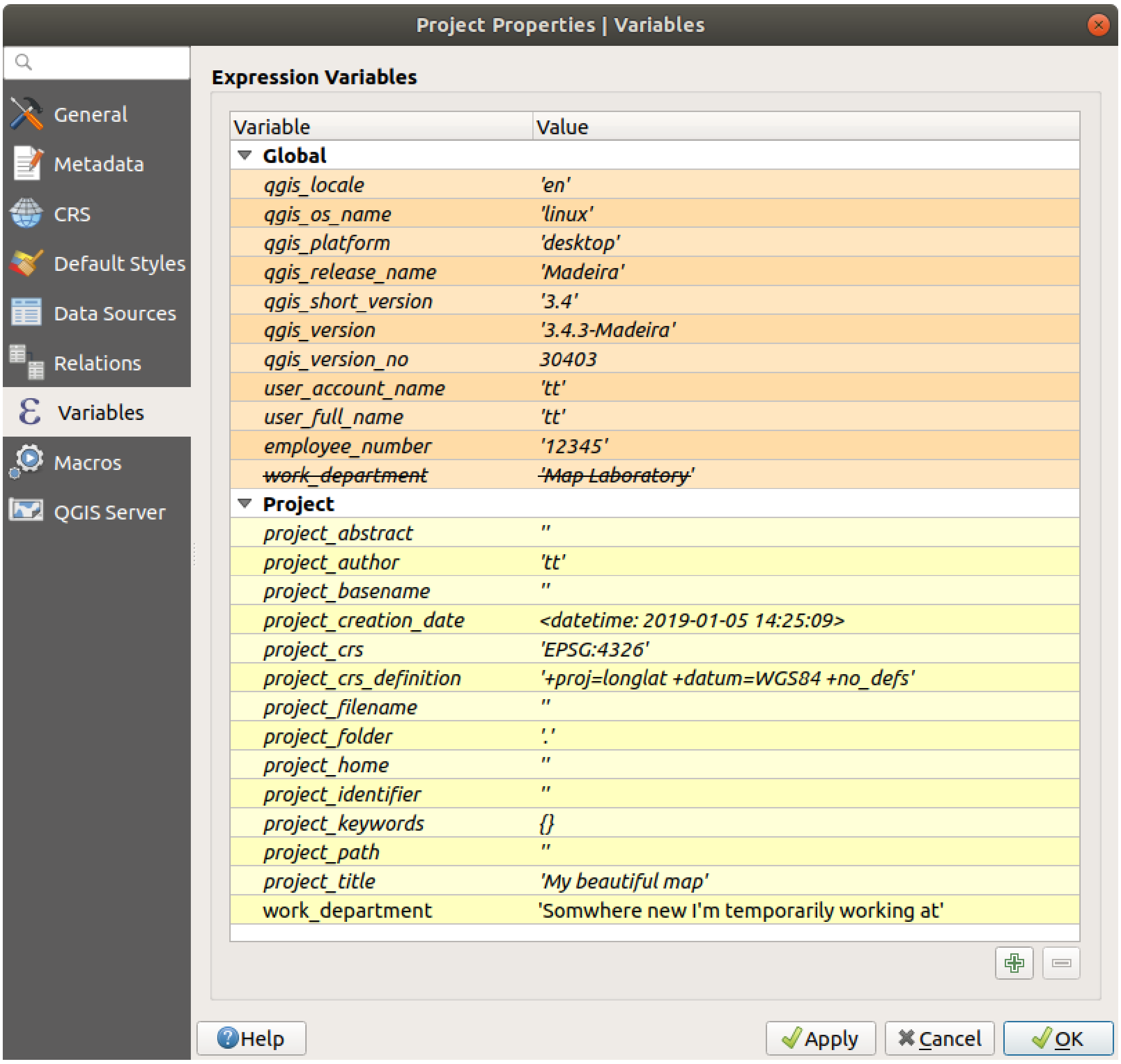


Abb. 11.24: Variables editor at the project level

11.9 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.

11.9.1 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.

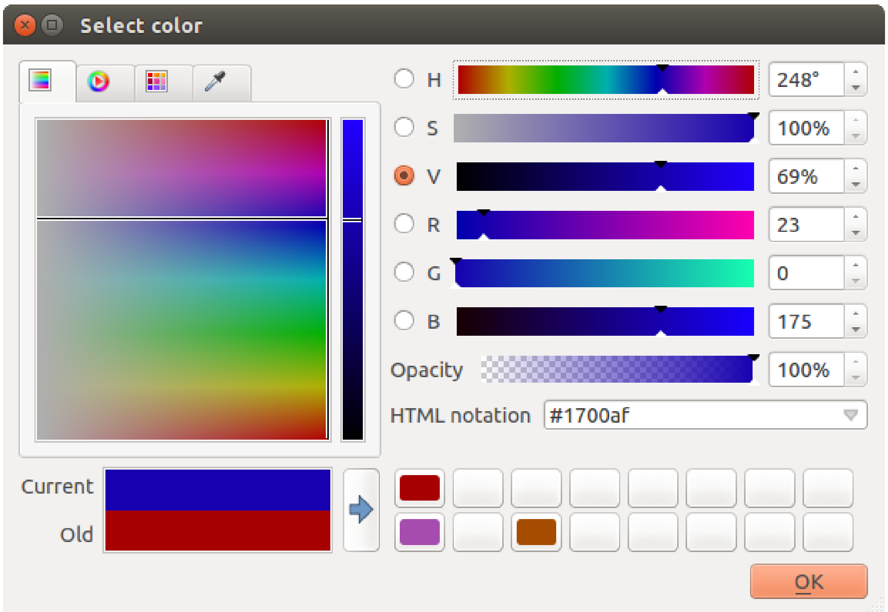


Abb. 11.25: Farbauswahl Verlaufstyp

In the Color swatches tab, you can choose from a list of color palettes (see *Colors Settings* 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 *figure_color_selector*)

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

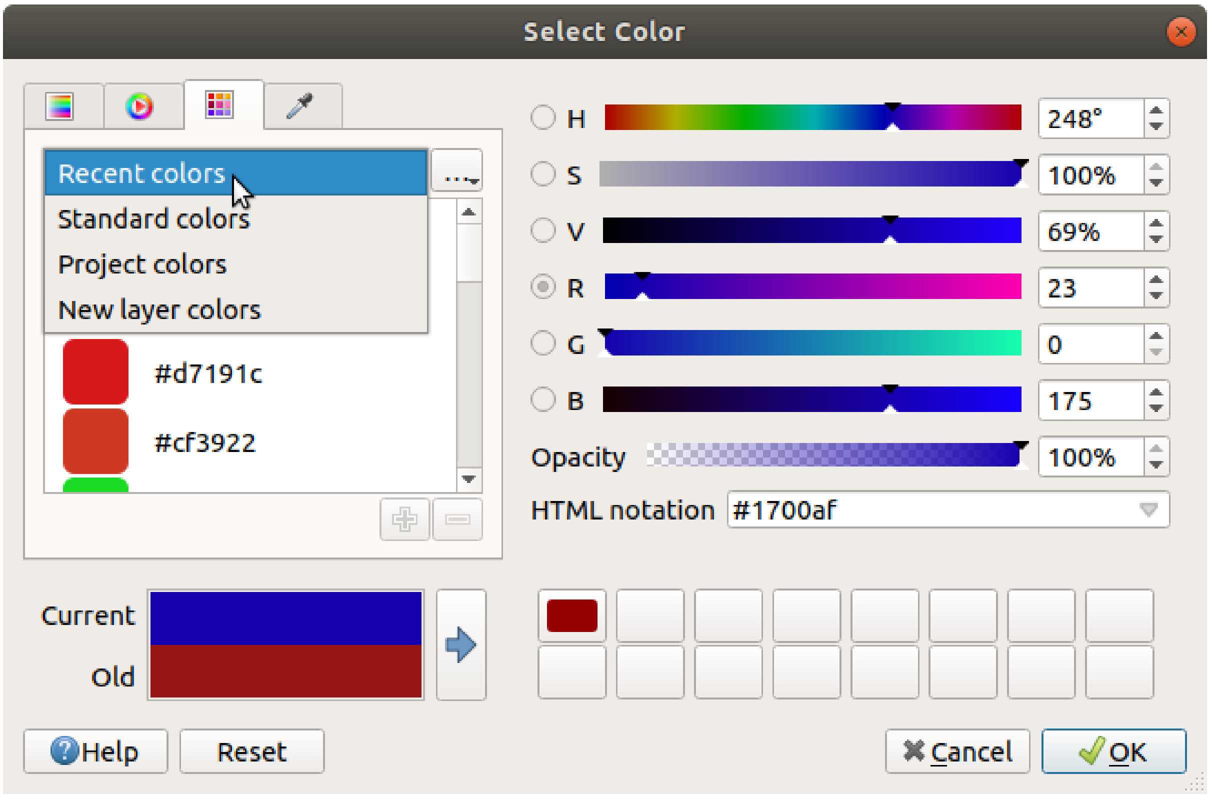


Abb. 11.26: Color selector swatches tab

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

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.

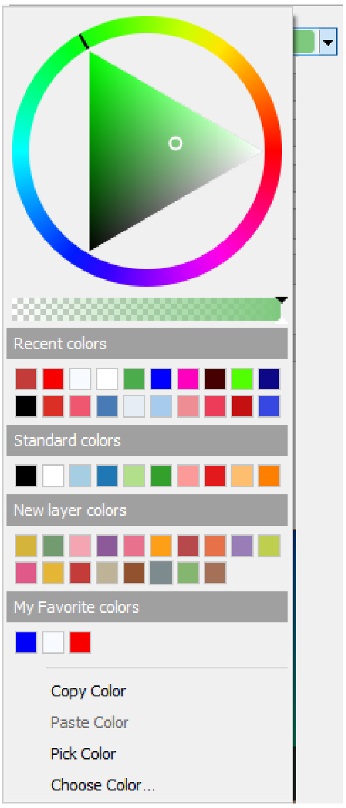


Abb. 11.27: Schnelle Farbwahl Menü

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.

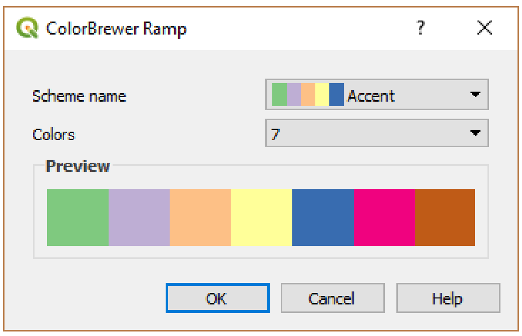


Abb. 11.28: Customizing a colorbrewer ramp

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*
- 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

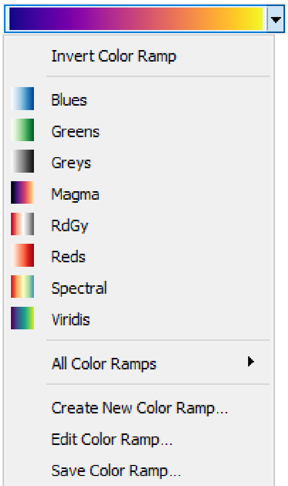


Abb. 11.29: Quick color ramp selection widget

11.9.2 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

11.9.3 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 the following options:

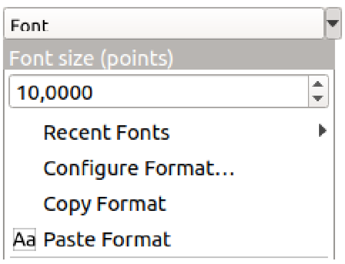


Abb. 11.30: Font selector drop-down menu

- *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 a dialog to set text format parameters. Depending on the context, it can be the OS default *Text format* dialog or the QGIS custom dialog with advanced formatting options (opacity, orientation, buffer, background, shadow, …) as described in section *Formatting the label text*.
- *Copy Format* of the text
- and *Paste Format* to the text, speeding configuration.

11.9.4 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*
- *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.
- 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.

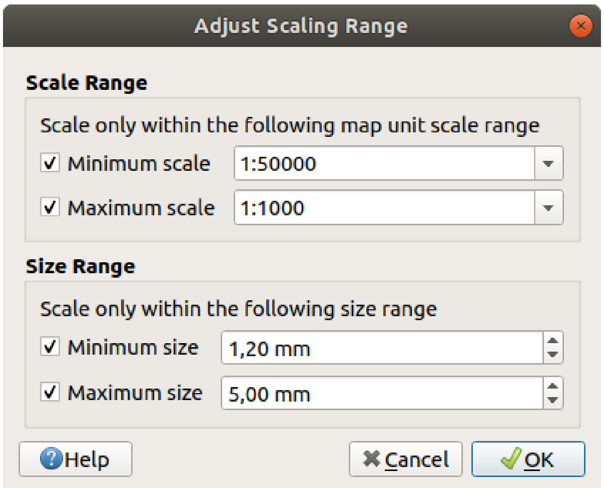


Abb. 11.31: Adjust scaling range dialog

11.9.5 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**: Dies ist der Standardmischmodus, der den Alphakanal des oben liegenden Pixels mit dem darunter liegenden Pixel vermischt. Die Farben werden nicht vermischt.
- **Heller**: Dies wählt das Maximum jeder Komponente der Vordergrund- und Hintergrundpixel. Seien Sie sich bewusst dass die Ergebnisse zackig und hart aussehen können.
- **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.
- **Hartes Licht**: Auch Hartes Licht ist dem Überlagerungsmodus sehr ähnlich. Hier soll die Projektion eines sehr intensiven Lichts auf ein Bild nachgeahmt werden.
- **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.

11.9.6 Datendefinierte Übersteuerung Setup

Next to many options in the vector layer properties dialog or settings in the print layout, you will find a Data defined override 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 to 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* Knopf, 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 numeric or color parameter, it has an *Assistant…* option that allows you to change how the data is applied to the parameter for each feature. The assistant allows you to:

- Define the *Input* data, ie:
  - the attribute to represent, using the Field listbox or the Set column expression function (see *Ausdrücke*)
  - 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 chosen attribute or the 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:
  - the minimum and maximum values to apply to the selected property (n case of a color setting, you'll need to provide a *color ramp*)
  - the *Scale method* of representation which can be **Flannery**, **Exponential**, **Surface** or **Radius**
  - the *Exponent* to use for data scaling
  - the output value or *color* to represent features with NULL values

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.

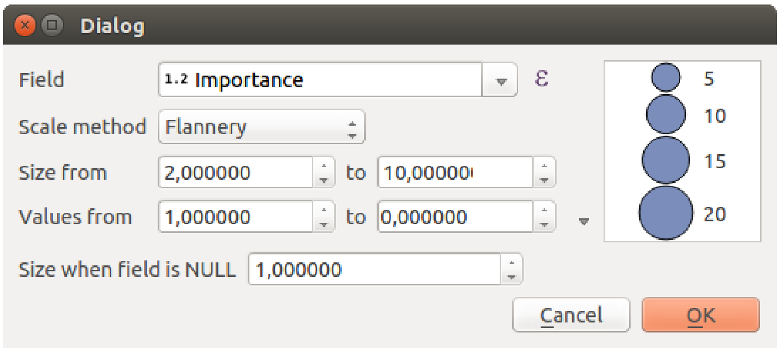


Abb. 11.32: The data-defined size assistant

The values presented in the varying size assistant above will set the size ‚Data-defined override' with:

```
coalesce(scale_exp(Importance, 1, 20, 2, 10, 0.57), 1)
```

KAPITEL 12

Die Stilbibliothek

12.1 The Style Manager

12.1.1 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 `symbology-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*).

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*;

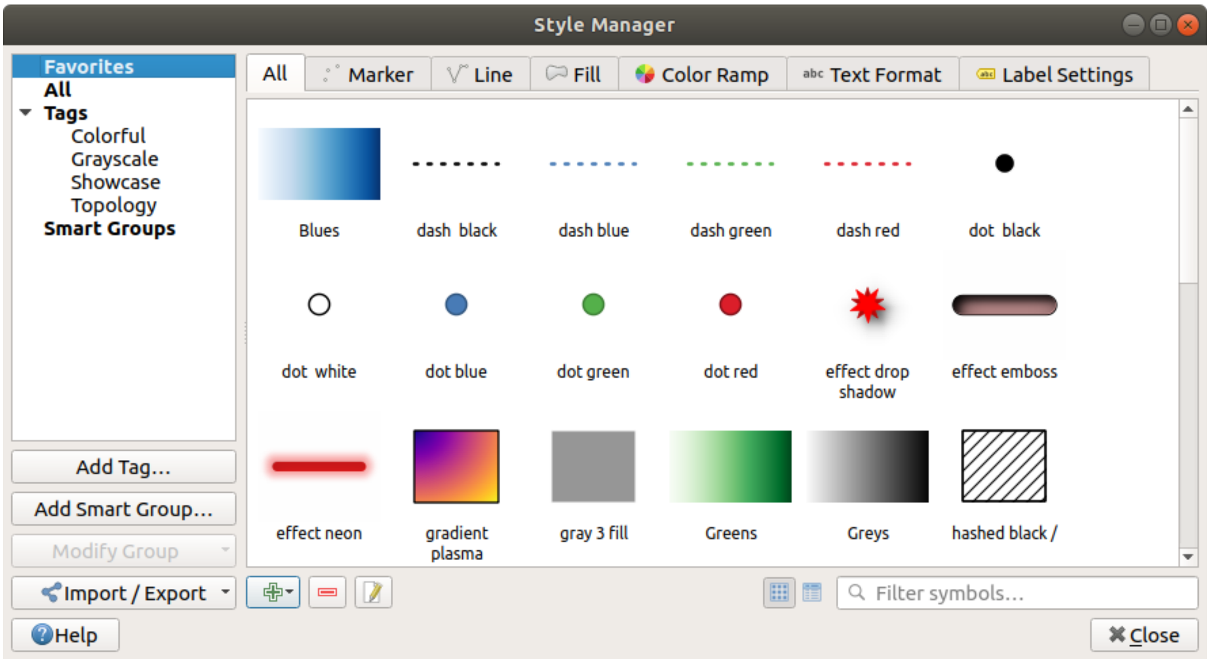


Abb. 12.1: The Style Manager

- abc *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…

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, *figure_smart_group*). 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.

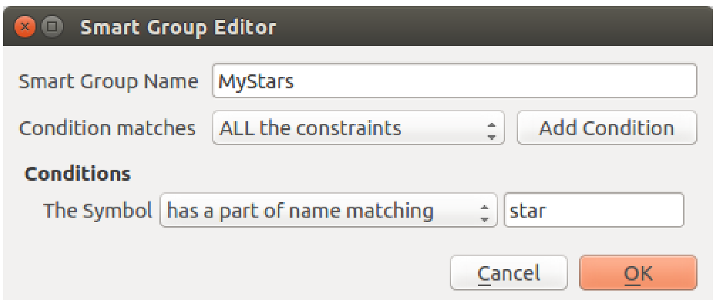


Abb. 12.2: Creating a Smart Group

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.

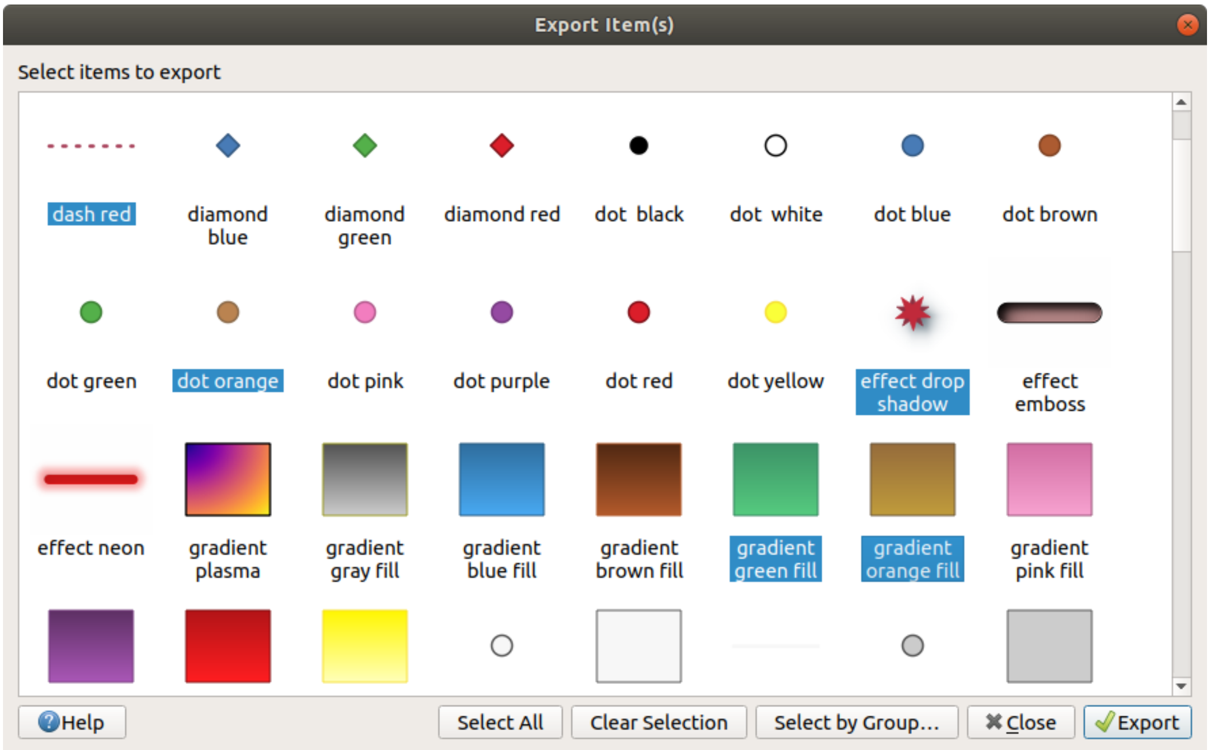


Abb. 12.3: Exporting style items

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*.

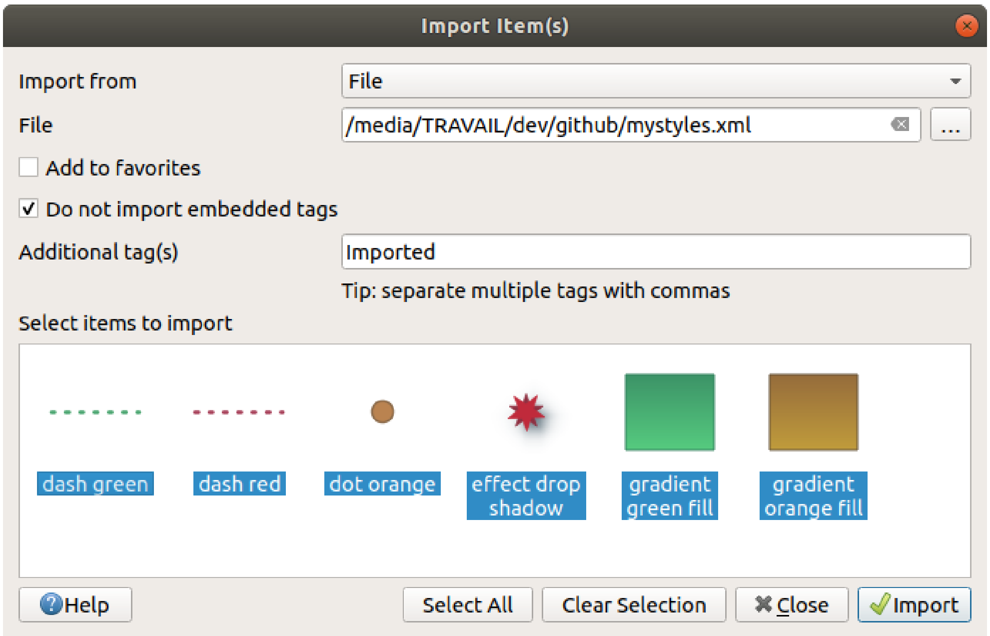


Abb. 12.4: Importing style items

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.

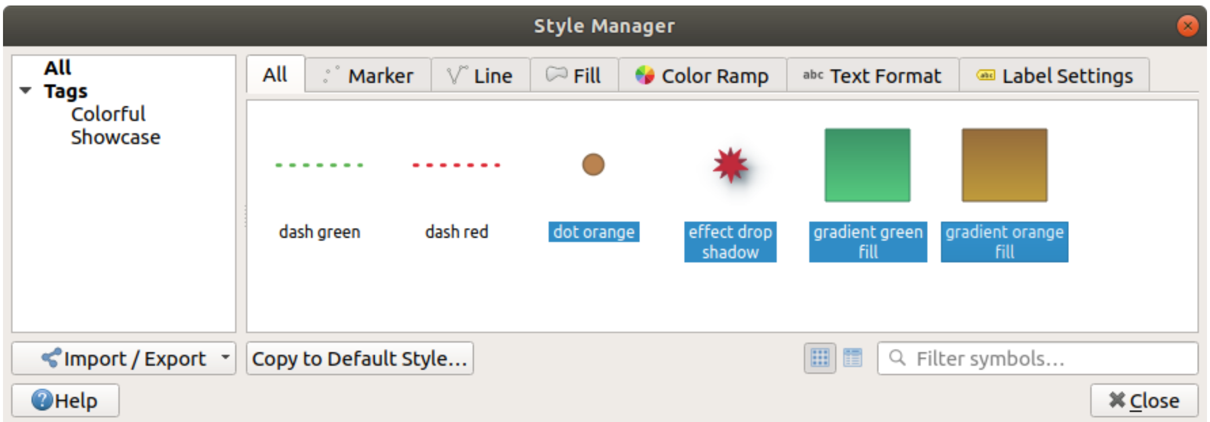


Abb. 12.5: Opening a style items file

The dialog also allows to export single symbols as `.PNG` or `.SVG` files.

12.1.2 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, generate 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.

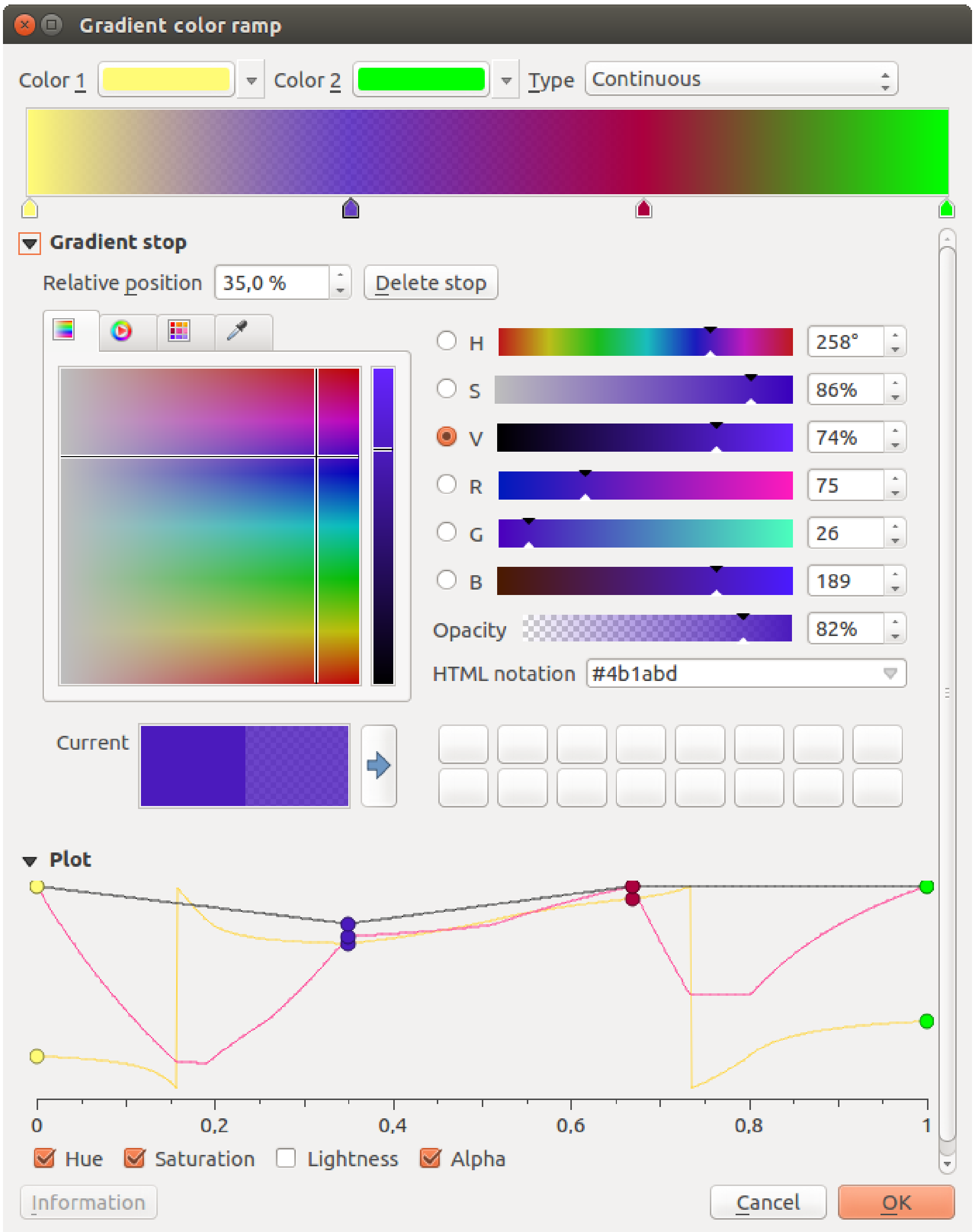


Abb. 12.6: Example of custom gradient color ramp with multiple stops

- *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‘.

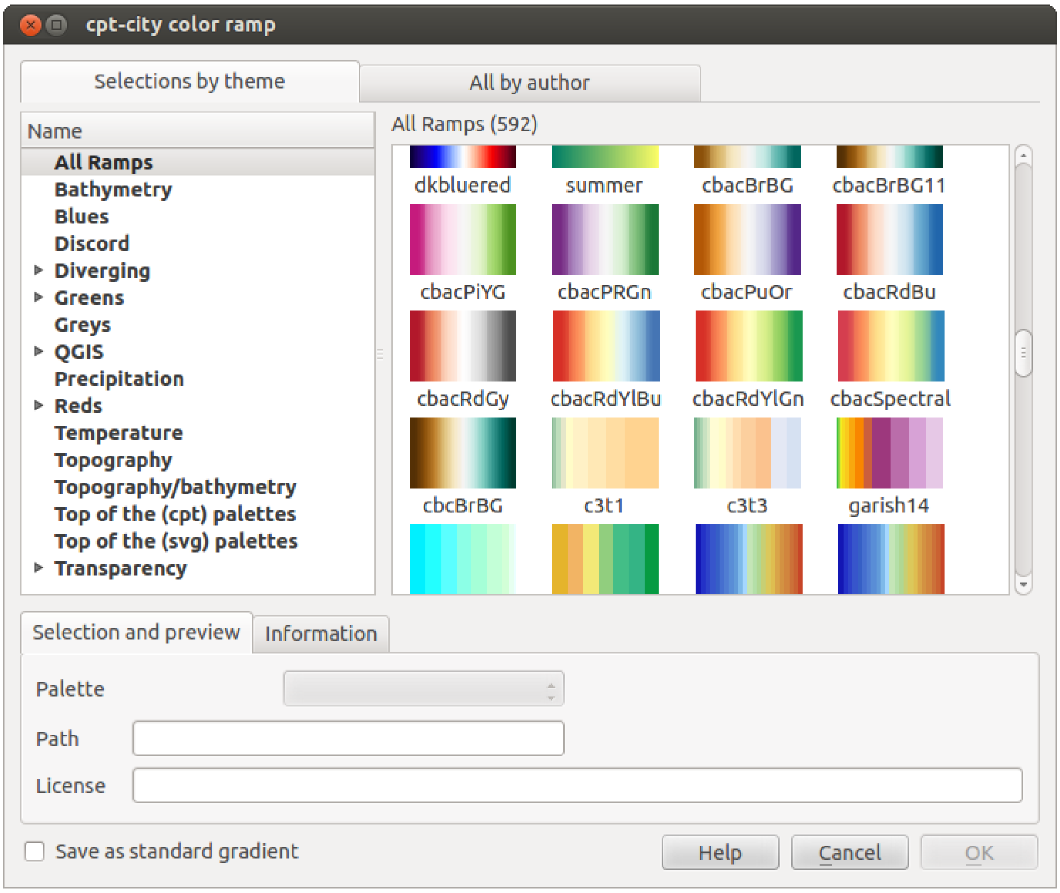


Abb. 12.7: cpt-city dialog with hundreds of color ramps

Tipp: Easily adjust the color stops of the gradient color ramp

Double-clicking the ramp preview or drag-and-drop a color from the color spot onto the ramp preview adds a new color stop. Each color stop can be tweaked using the *Farbauswahl* widgets or by plotting each of its parameters. You can also reposition it using the mouse, the arrow keys (combine with `Shift` key for a larger move) or the *Relative position* spinbox. Pressing *Delete stop* as well as `DEL` key removes the selected color stop.

12.2 The Symbol Selector

The Symbol selector is the main dialog to design a symbol. You can create or edit Marker, Line or Fill Symbols.

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.

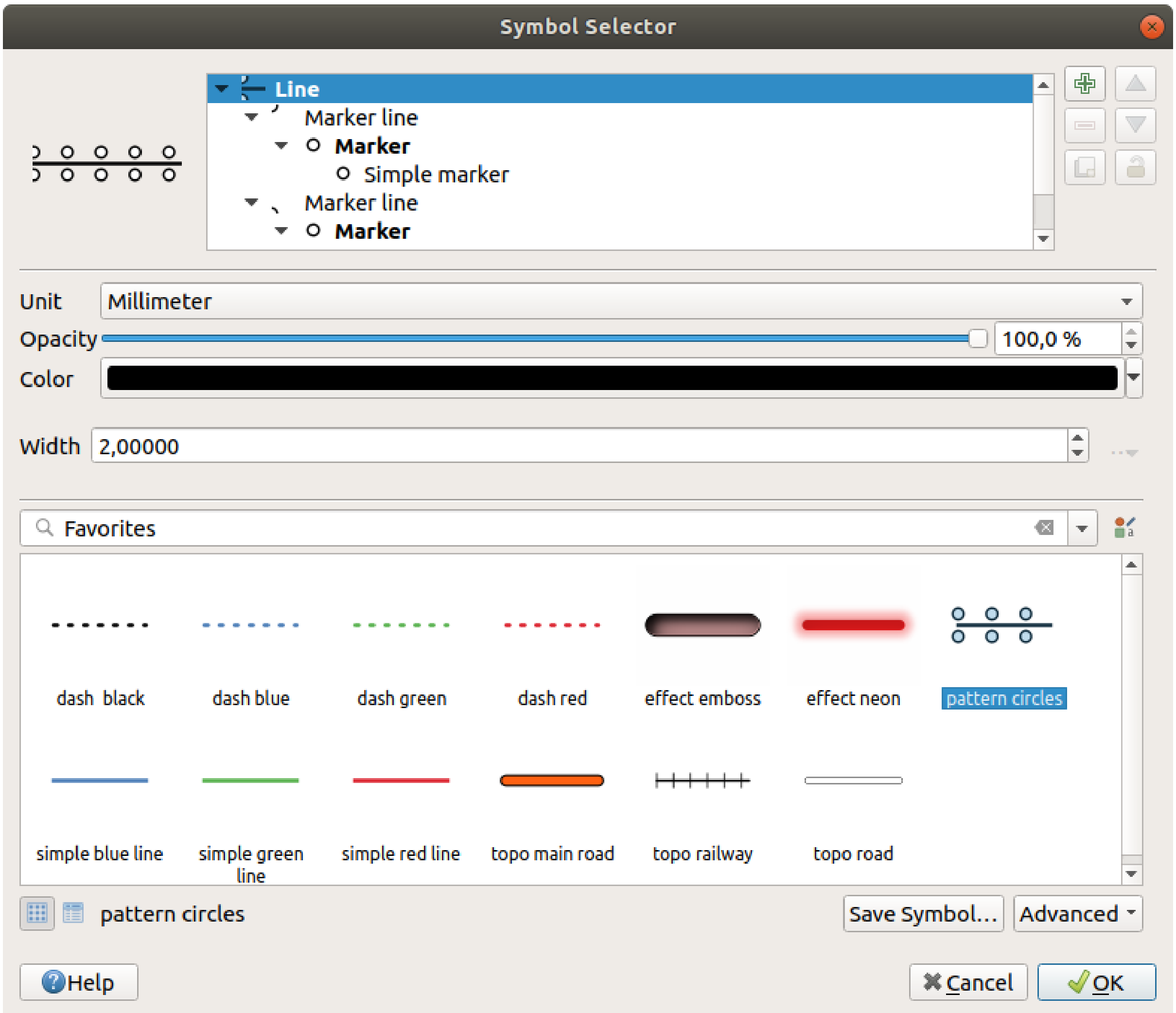


Abb. 12.8: Designing a Line symbol

12.2.1 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

12.2.2 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

Tipp: 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 *symbology* 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*

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 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)

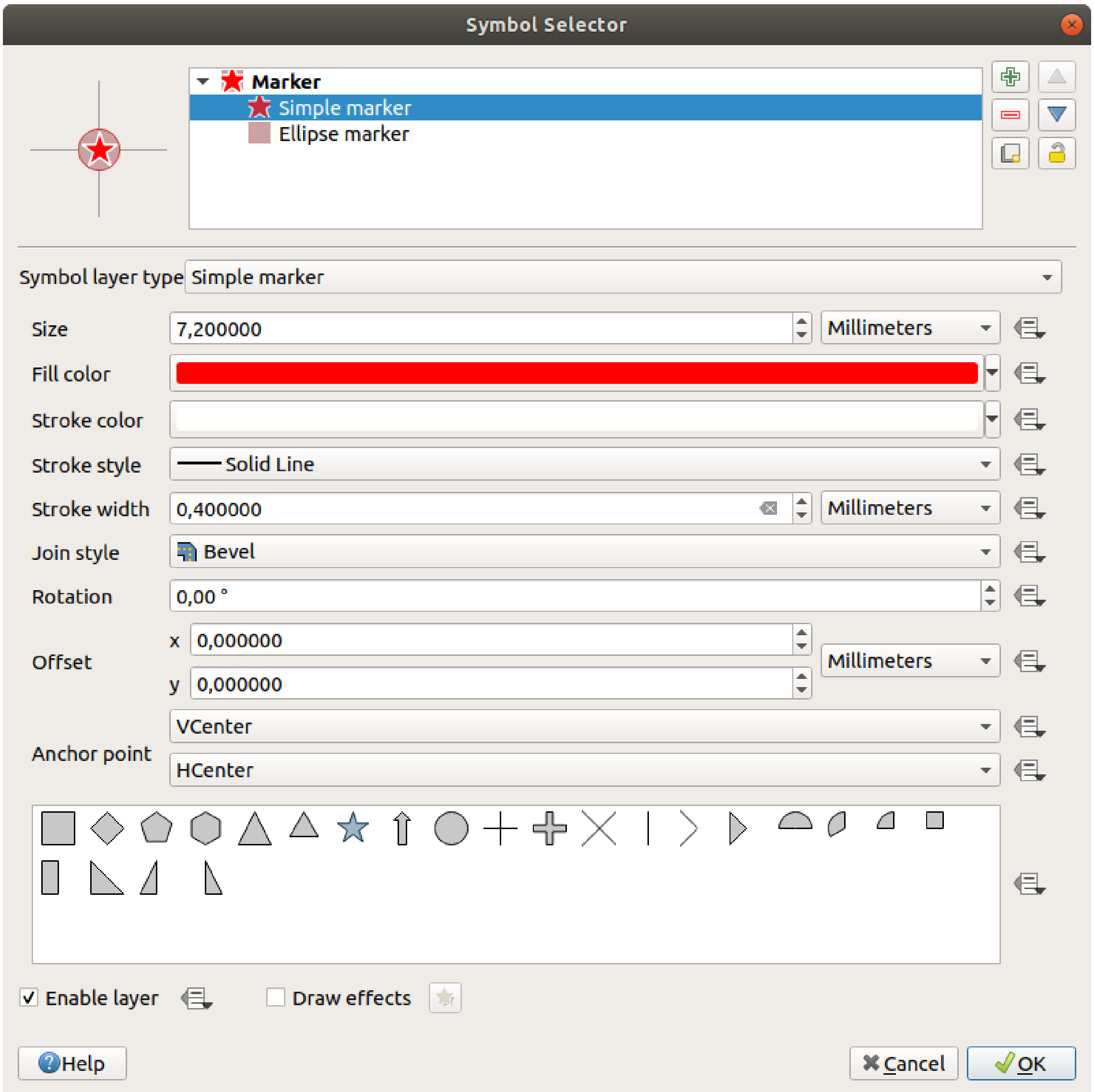


Abb. 12.9: Designing a Simple Marker Symbol

- **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**: use installed fonts as marker symbols
- **Geometry generator** (see *The Geometry Generator*)
- **Raster image marker**: use an image (`PNG`, `JPG`, `BMP` ...) as marker symbol. The image can be a file on the disk, a remote URL or embedded in the style database (*more details*). Width and height of the image can be set independently or using the Lock aspect ratio.
- **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 or embedded in the style database (*more details*).

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.

Tipp: Enable SVG marker symbol customization

To have the possibility to change the colors of a *SVG marker*, you have to add the placeholders `param(fill)` for fill color, `param(outline)` for stroke color 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" stroke="param(outline) #00ff00" stroke-width=
↪"param(outline-width) 10" width="100" height="100">
</rect>
</svg>
```

Line Symbols

Appropriate for line geometry features, line symbols have following symbol layer types:

- **Simple line** (default): available settings are:

  The simple line symbol layer type has many of the same properties as the *simple marker symbol*, and in addition:

  - *Cap style*
  - *Use custom dash pattern*: overrides the *Stroke style* setting with a custom dash.

- **Arrow**: draws lines as curved (or not) arrows with a single or a double head with configurable width, length and thickness. To create a curved arrow the line feature must have at least three vertices. 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*)
- **Marker line**: repeats a *marker symbol* over the length of a line.
  - The markers placement can be at a regular distance or based on the line geometry: first, last or each vertex, on the central point of the line or of each segment, or on every curve point.
  - The markers placement can also be given an offset along the line
  - The *Rotate marker* option allows you to set 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.

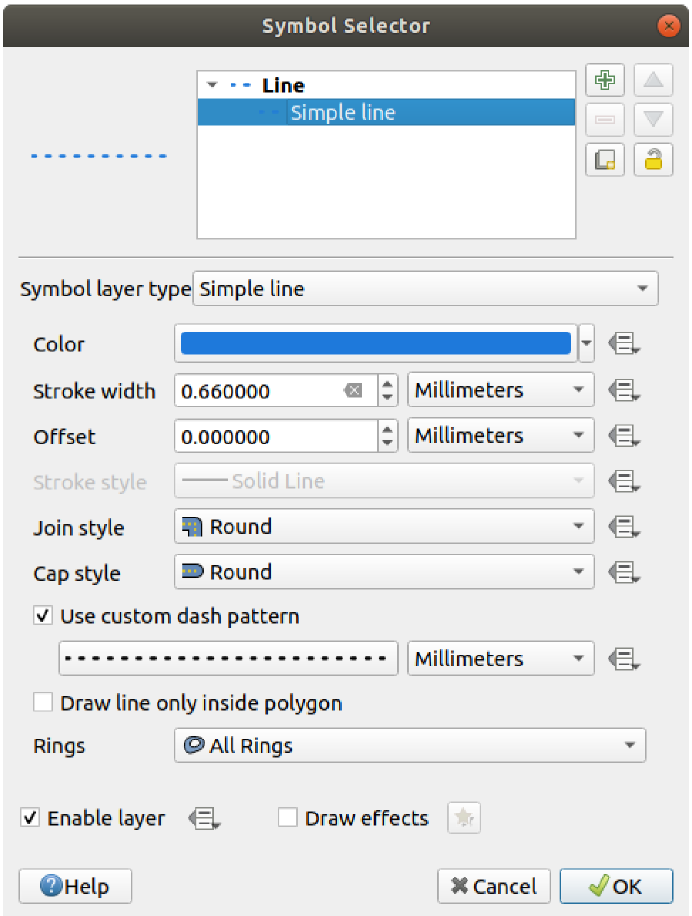


Abb. 12.10: Designing a Simple Line Symbol

- The marker line can also be offset from the line itself.

- **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*

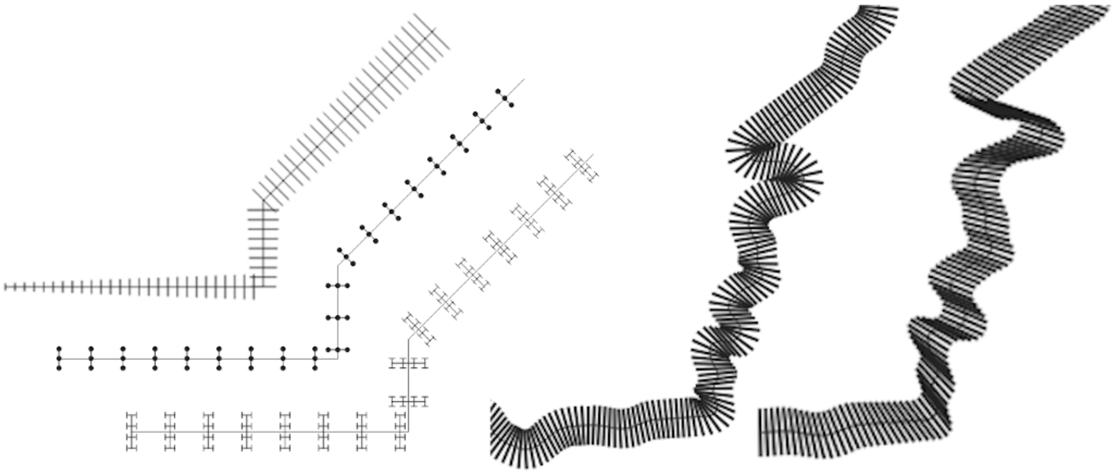


Abb. 12.11: Examples of hashed lines

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
- **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.

  The marker(s) can be placed on every part of a multi-part feature or only on its biggest part, and forced to be inside the polygon.
- **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);
- **Line pattern fill**: fills the polygon with a hatching pattern of *line symbol layer*. You can set a rotation, the spacing between lines and an offset from the feature boundary;
- **Point pattern fill**: fills the polygon with a hatching pattern of *marker symbol layer*. You can set the distance and a displacement between rows of markers, and an offset from the feature boundary;
- **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.
- **SVG fill**: fills the polygon using *SVG markers*;

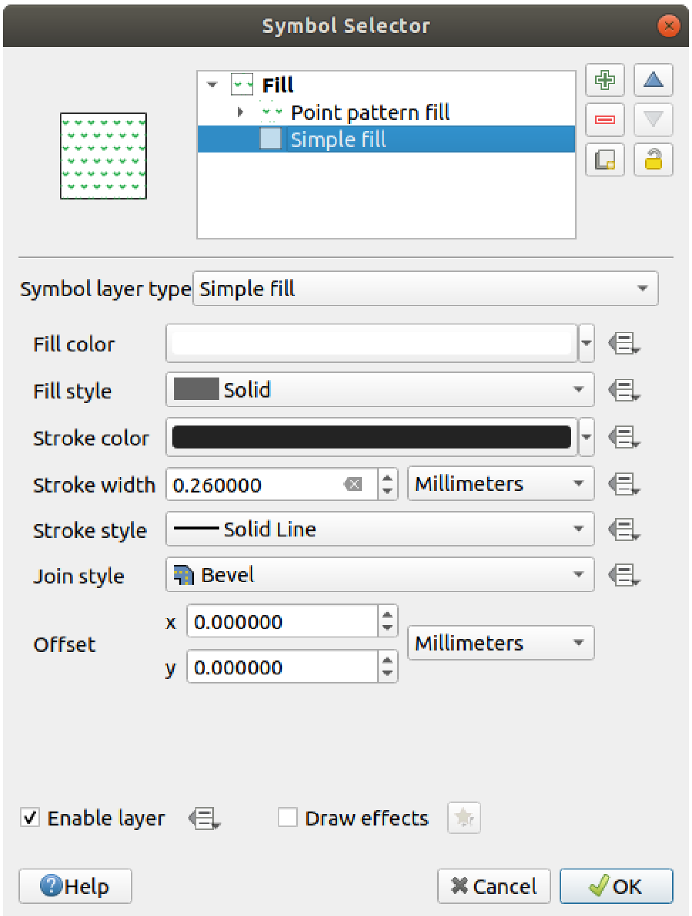


Abb. 12.12: Designing a Simple Fill Symbol

- **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;
- **Outline: Hashed line**: uses a *hash line symbol* layer to represent the polygon boundary (the interior rings, the exterior ring or all the rings).
- **Outline: Marker line**: uses a marker line symbol layer to represent the polygon boundary (the interior rings, the exterior ring or all the rings).
- **Outline: simple line**: uses a simple line symbol layer to represent the polygon boundary (the interior rings, the exterior ring 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.

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).

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.

Some examples:

```
-- render 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 )

-- Given polygon layer1( id1, layer2_id, ...) and layer2( id2, fieldn...)
-- render layer1 with a line joining centroids of both where layer2_id = id2
make_line( centroid( $geometry ),
           centroid( geometry( get_feature( 'layer2', 'id2', attribute(
               $currentfeature, 'layer2_id') ) )
         )

-- Create a nice radial effect of points surrounding the central feature
-- point when used as a MultiPoint geometry generator
collect_geometries(
  array_foreach(
    generate_series( 0, 330, 30 ),
      project( $geometry, .2, radians( @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.

12.3 Setting a label

Labels are textual information you can display on vector features. They add details you could not necessarily represent using symbols.

The *Style Manager* dialog allows you to create a set of labels or text formats (ie the appearance of the text, including font, size, colors, shadow, background…). Each of these items could later be applied to layers in the *Labels* tab of the vector *Layer Properties* dialog or *Layer Styling* panel or using the Layer Labeling Options button of the **Labels toolbar**. You can also directly configure them in the abovementioned dialogs.

The *Label Settings* dialog allows you to configure smart labeling for vector layers. Setting a label includes configuring the *text format*, and how the label relates with the features or other labels (through *placement*, *rendering* and *callout*).

12.3.1 Formatting the label text

To create text formats, you can:

1. Open the *Style Manager* dialog
2. Activate the *Text format* tab

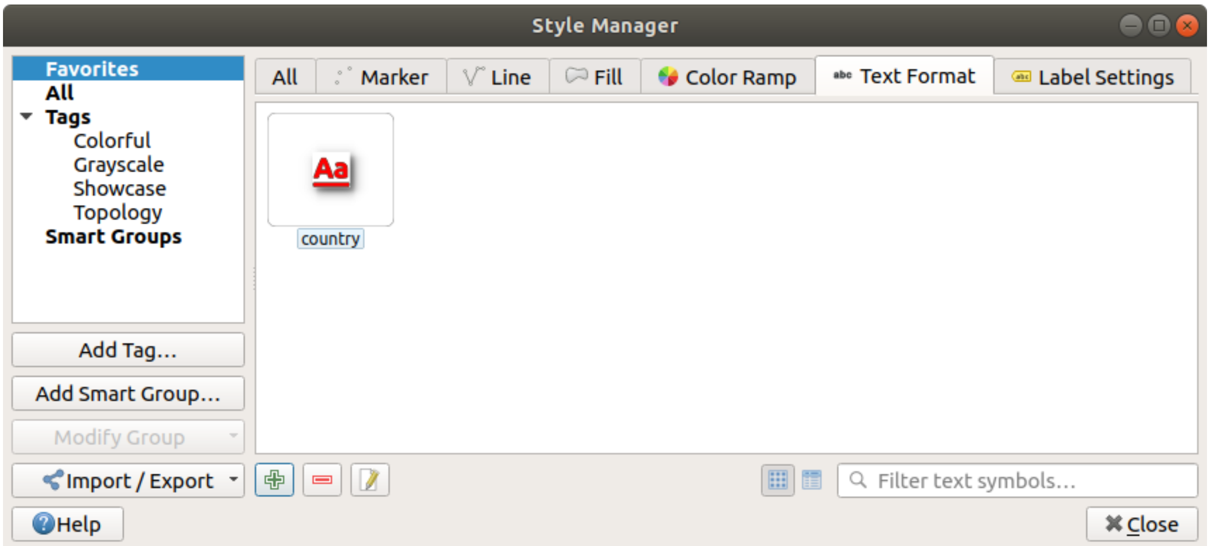


Abb. 12.13: Text formats in Style Manager dialog

3. Press the Add item button. The *Text Settings* dialog opens with the following properties. As usual, these properties are *data-definable*.

Text tab

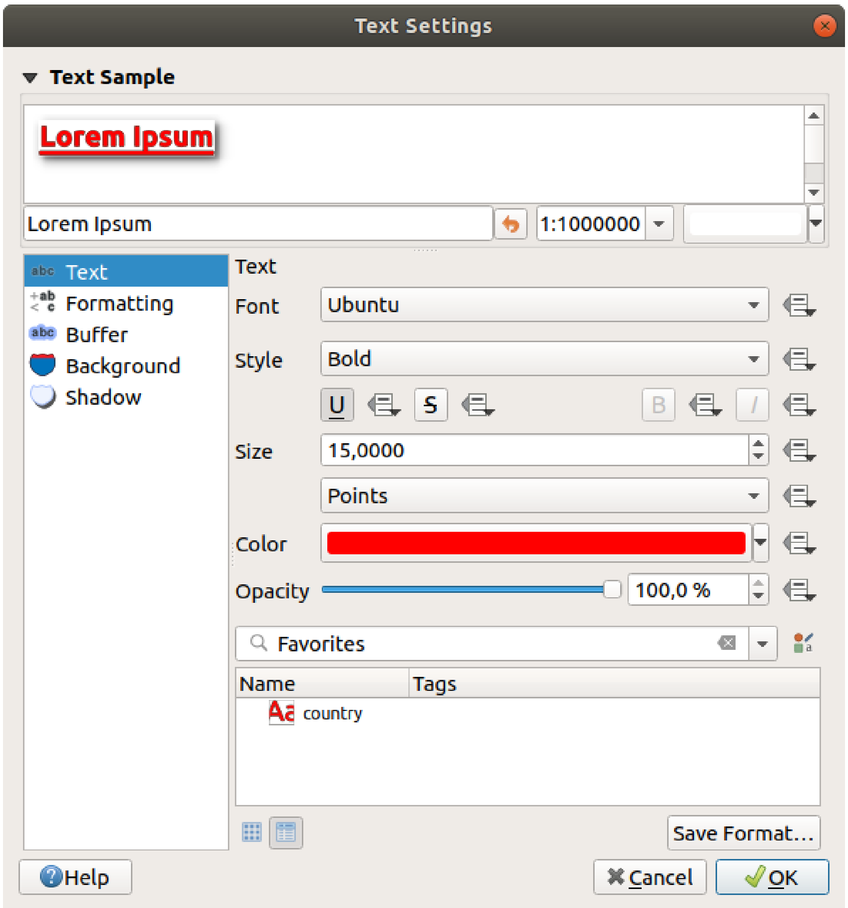


Abb. 12.14: Text settings - Text tab

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 striked through
- the *Size* in any *supported unit*
- the *Color*
- and the *Opacity*.

At the bottom, a text formats list widget shows a filterable list of text formats stored within your *style manager database*. This allows you to easily set text formats to match styles saved in the local style database, and also to add a new text format to the style database based on the current settings. Press the *Save format…* button to store the current text format in the *Style Manager*, providing a name and tag(s).

Likewise, a label settings list widget is shown when configuring labels, allowing you to pick from the Style Manager widget or to add new styles to it.

Formatting tab

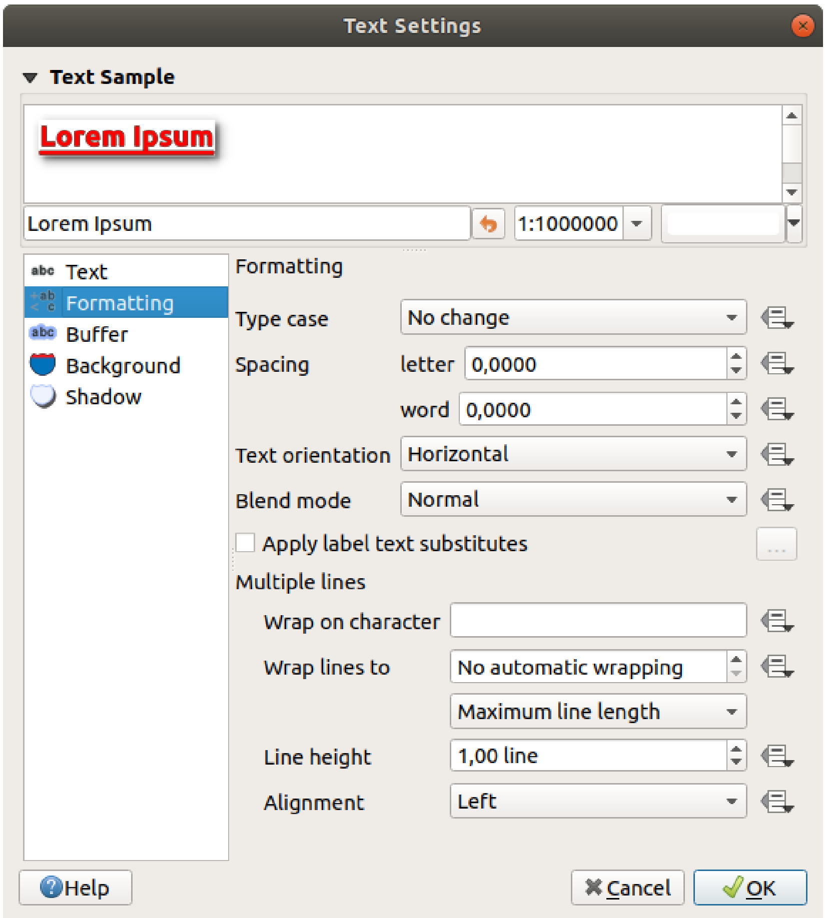


Abb. 12.15: Text settings - Formatting tab

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 *All uppercase*, *All lowercase* or *Capitalize first letter*. Note that the last option modifies only the first letter of each word and leaves the other letters in the text untouched.
- Under *Spacing*, change the space between words and between individual letters.
- *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.
- 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*

  – Format the *Alignment*: typical values available are *Left*, *Right* 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.

- 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

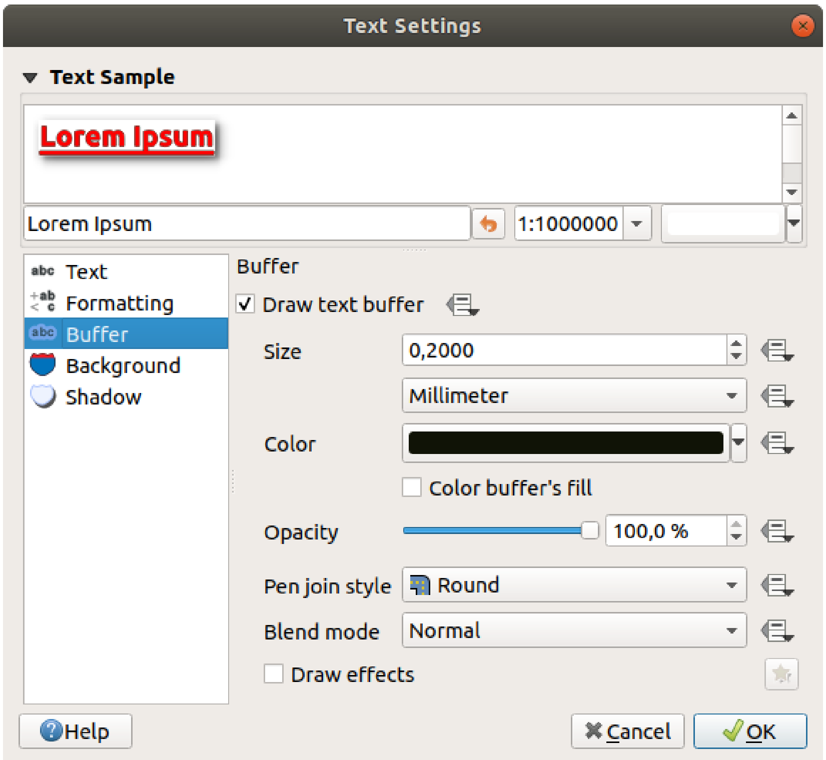


Abb. 12.16: Text settings - Buffer tab

To create a buffer around the label, activate the *Draw text buffer* checkbox in the abc *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*
- 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*.

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*).
- The *Fill color*, *Stroke color* and *Stroke width* for shape types other than the marker symbol. Use the *Load symbol parameters* to revert changes on an SVG symbol to its default settings.
- A *Pen join style*: it can be *Round*, *Miter* or *Bevel* (applies to rectangle and square shapes only)
- *Draw effects* to add advanced *paint effects* for improving text readability, eg through outer glows and blurs.

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 shadowded, ie:
  - The angle: clockwise, it depends on the underlying item orientation
  - The distance of offset from the item being shadowded
  - 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.

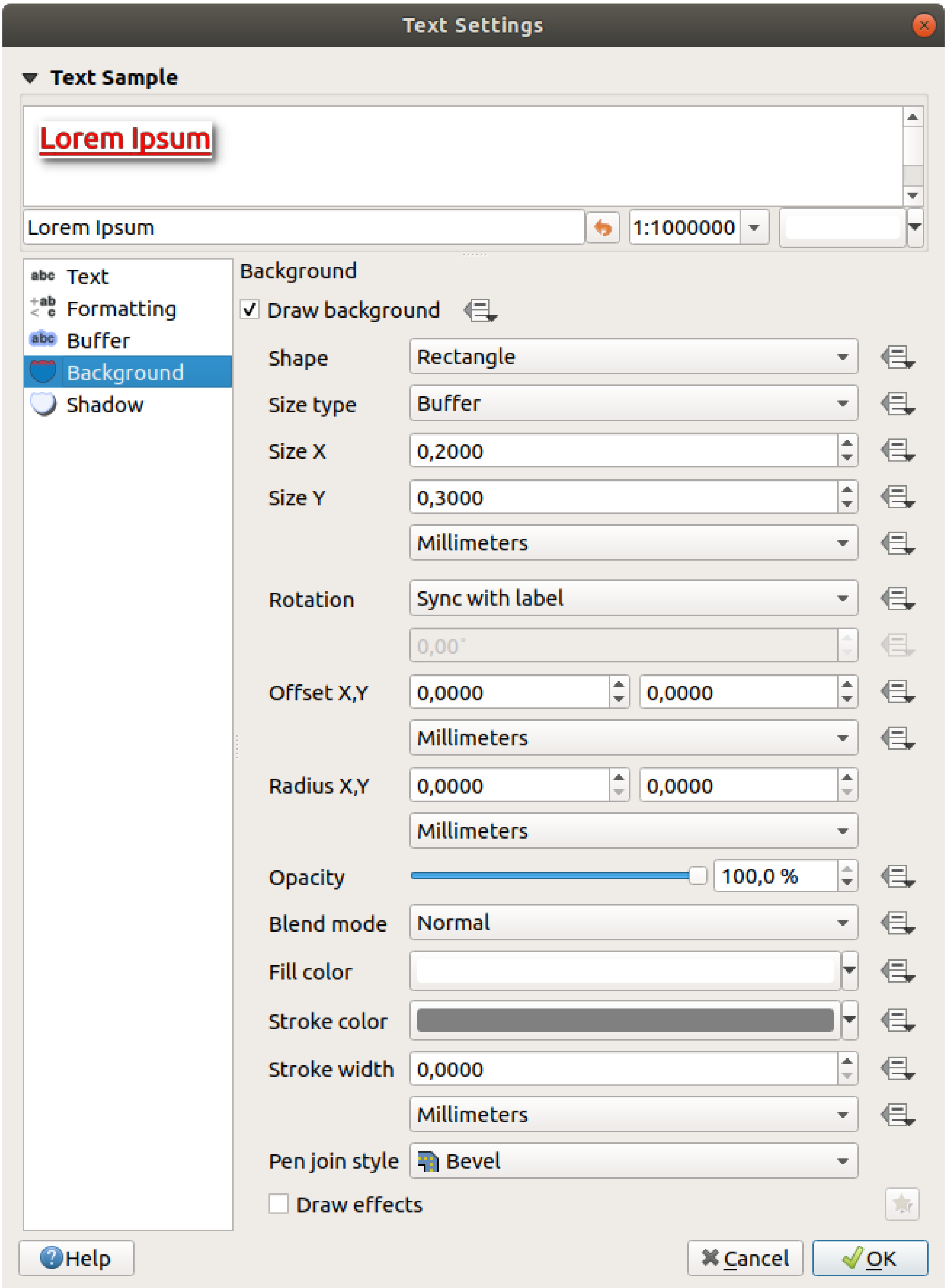


Abb. 12.17: Text settings - Background tab

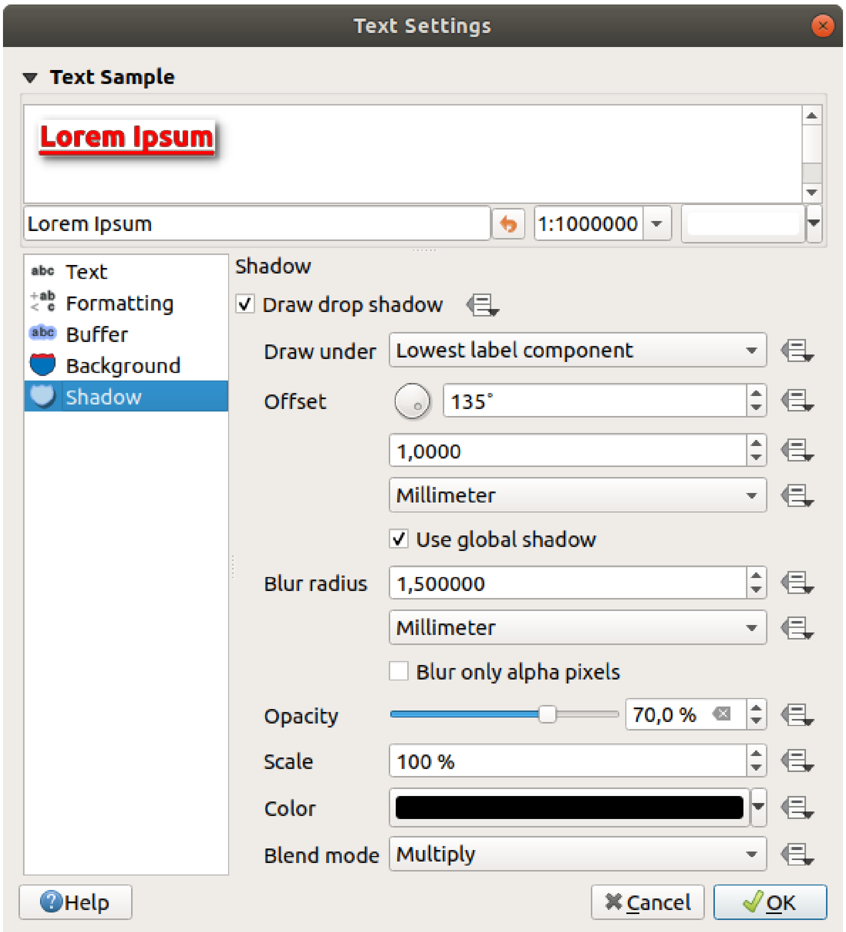


Abb. 12.18: Text settings - Shadow tab

- 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*).

12.3.2 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. 12.19: Labels with various callouts settings

To add a callout to a label, enable the abc *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
2. Select the *Line style* with full capabilities of a *line symbol* including layer effects, and data-defined settings
3. Set the *Minimum length* of callout lines
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 *Offset from label area* option: controls the distance from the label closest corner where callout lines end. This avoids drawing lines right up against the text.
6. *Draw lines to all features parts* from the feature's label
7. Set an *Anchor point* on the (polygon) feature to use as end point of the connector line : available options are:

- *Pole of inaccessibility*
- *Point on exterior*
- *Point on surface*
- *Centroid*

12.3.3 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

With the *Cartographic* placement mode, 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* either from the point feature itself or from the bounds of the symbol used to represent the feature. 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.

By default, placements are prioritised in the following order:

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.

Placement priority can, however, 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.

The *Around point* setting places the label in an equal radius (set in *Distance*) circle around the feature. The placement of the label can even be constrained using the *Quadrant* option.

With the *Offset from point*, labels are placed at a fixed offset from the point feature. You can select the *Quadrant* in which to place your label. You are also able to set the *Offset X,Y* distances between the points and their labels and can alter the angle of the label placement with the *Rotation* setting. Thus, placement in a selected quadrant with a defined rotation is possible.

Placement for line layers

Label options for line layers include *Parallel*, *Curved* or *Horizontal*. For the *Parallel* and *Curved* options, you can set the position to *Above line*, *On line* and *Below line*. It's possible to select several options at once. In that case, QGIS will look for the optimal label position. For Parallel and curved placement options, you can also use the line orientation for the position of the label. Additionally, you can define a *Maximum angle between curved characters* when selecting the *Curved* option (see Figure_labels_placement_line).

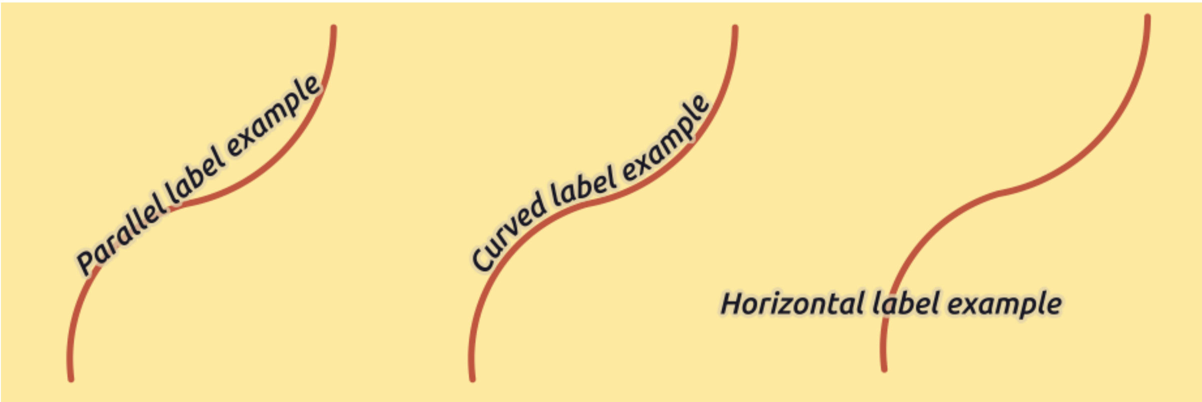


Abb. 12.20: Label placement examples in lines

For all three placement options, in *Repeat*, you can set up a minimum distance for repeating labels. The distance can be in `mm` or in `map units`.

Placement for polygon layers

You can choose one of the following options for placing labels in polygons (see *figure_labels_placement_polygon*):

- *Offset from centroid*,
- *Horizontal (slow)*,
- *Around centroid*,
- *Free (slow)*,
- *Using perimeter*,
- and *Using perimeter (curved)*.

In the *Offset from centroid* settings you can specify if the centroid is of the *visible polygon* or *whole polygon*. That means that either the centroid is used for the polygon you can see on the map or the centroid is determined for the whole polygon, no matter if you can see the whole feature on the map. You can place your label within a specific quadrant, and define offset and rotation.

The *Around centroid* setting places the label at a specified distance around the centroid. Again, you can define *visible polygon* or *whole polygon* for the centroid.

With the *Horizontal (slow)* or *Free (slow)* options, QGIS places at the best position either a horizontal or a rotated label inside the polygon.

With the *Using perimeter* option, the label will be drawn next to the polygon boundary. The label will behave like the parallel option for lines. You can define a position and a distance for the label. For the position, *Above line*, *On line*, ☑ *Below line* and ☑ *Line orientation dependent position* are possible. You can specify the distance between the label and the polygon outline, as well as the repeat interval for the label.

The *Using perimeter (curved)* option helps you draw the label along the polygon boundary, using a curved labeling. In addition to the parameters available with *Using perimeter* setting, you can set the *Maximum angle between curved characters polygon*, either inside or outside.

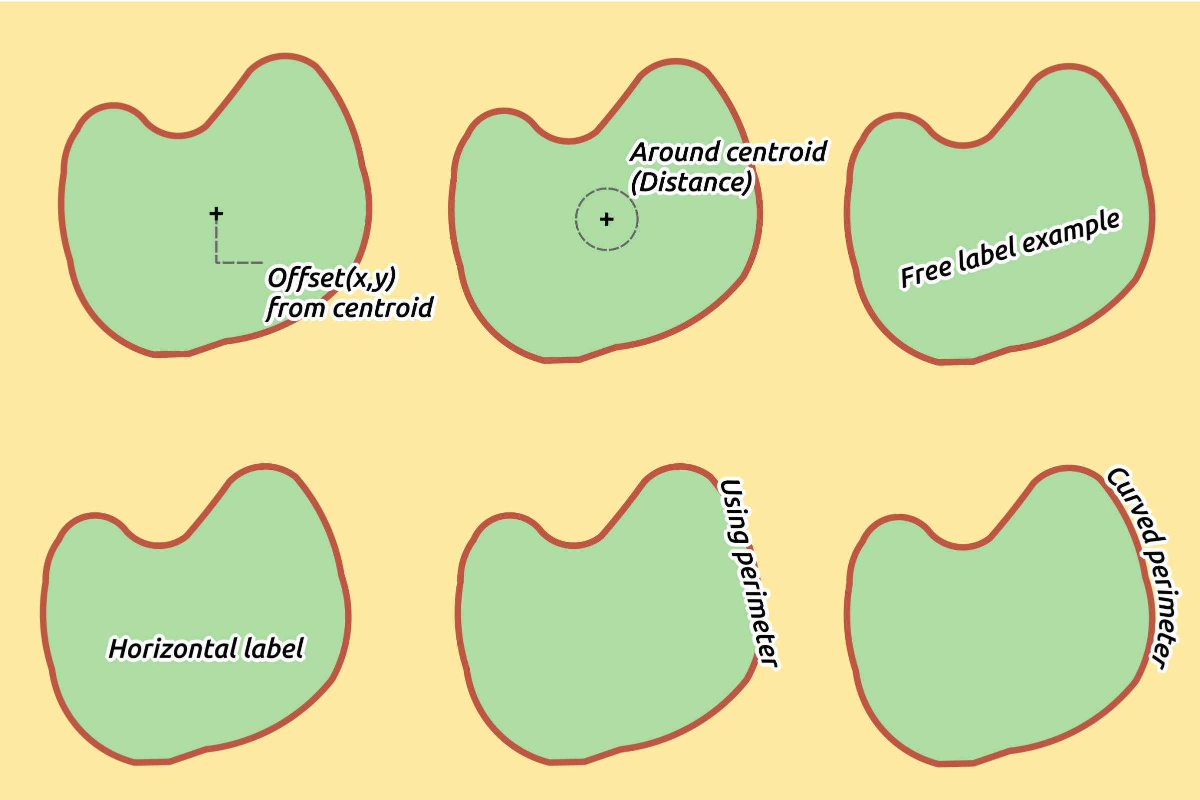


Abb. 12.21: Label placement examples in polygons

In the *priority* section you can define the priority with which labels are rendered for all three vector layer types (point, line, polygon). This placement option interacts with the labels from other vector layers in the map canvas. If there are labels from different layers in the same location, the label with the higher priority will be displayed and the others will be left out.

12.3.4 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.

- While rendering labels and in order to display readable labels, QGIS automatically evaluates the position of the labels and can hide some of them in case of collision. You can however choose to *Show all labels for this layer (including colliding labels)* in order to manually fix their placement (see *The Label Toolbar*).
- 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**.

Feature options

Under *Feature options*:

- You can choose to *label every part of a multi-part feature* and *limit the number of features to be labeled*.
- 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.

Obstacles

An obstacle is a feature QGIS tries as far as possible to not place labels over. From the *Obstacles* frame, you can manage the covering relation between labels and features:

- Activate the *Discourage labels from covering features* option to decide whether features of the layer should act as obstacles for any label (including labels from other features in the same layer).

  Instead of the whole layer, you can define a subset of features to use as obstacles, using the data-defined override control next to the option.
- The priority control slider for obstacles allows you to make labels prefer to overlap features from certain layers rather than others. A **Low weight** obstacle priority means that features of the layer are less considered as obstacles and thus more likely to be covered by labels. This priority 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 type of obstacle the features could be, by minimising the labels placement:
  – **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 boundary of the polygon (prefers placing labels outside or completely inside the polygon). E.g., it can be useful for regional boundary layers, where the features cover an entire area. In this case, it's impossible to avoid placing labels within these features, and it looks much better to avoid placing them over the boundaries between features.

KAPITEL 13

Verwaltung von Datenquellen

13.1 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.
- Zu den Datenbankformaten gehören PostgreSQL/PostGIS, SQLite/SpatiaLite, Oracle, DB2 oder MSSQL Spatial, MySQL…
- Web map and data services (WM(T)S, WFS, WCS, CSW, XYZ tiles, ArcGIS services, …) are also handled by QGIS providers. See *QGIS als OGC Datenclient* 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 *Settings ▸ Options ▸ GDAL* menu (for raster) 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 offers a unified interface to open vector or raster file-based data as well as databases or web services supported by QGIS. It can be set modal or not with the *Modeless data source manager dialog* in the *Settings ▸ Options ▸ General* menu.

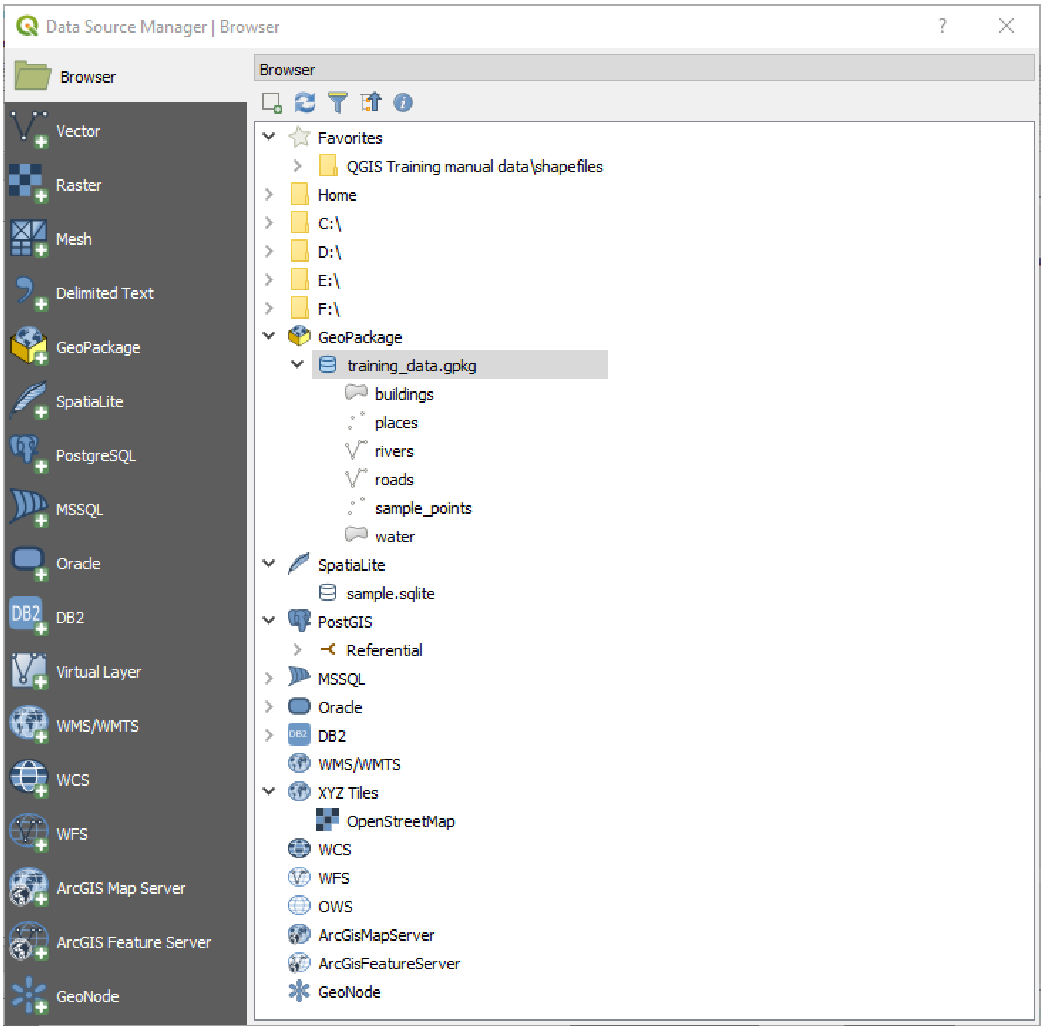


Abb. 13.1: QGIS Data Source Manager dialog

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.

13.1.1 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 *figure_browser_panels*. 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 *Räumliche Lesezeichen*)
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*
- *PostGIS*
- *MSSQL*
- *Oracle*
- *DB2*
- *WMS/WMTS*
- *XYZ Tiles*
- *WCS*
- *WFS*
- *OWS*
- *ArcGISMapServer*
- *ArcGISFeatureServer*
- *GeoNode*

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 –>*
  - *Directory…*
  - *GeoPackage…*
  - *ShapeFile…*
- *Add as a Favorite*
- *Hide from Browser*
- *Fast Scan this Directory*
- *Open Directory*
- *Open in Terminal*
- *Eigenschaften…*
- *Directory Properties…*

Favourites, can also be removed and renamed:

- *Rename favourite…*
- *Remove favourite*

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:

- *Delete File „<name of file>“…*
- *Export Layer –> To File…*
- *Add Layer to Project*
- *Layer Properties*
- *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.

> **Tipp: 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 *Symbology Properties* 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.

Tipp: Add layers to QGIS by simple drag-and-drop from your OS file browser

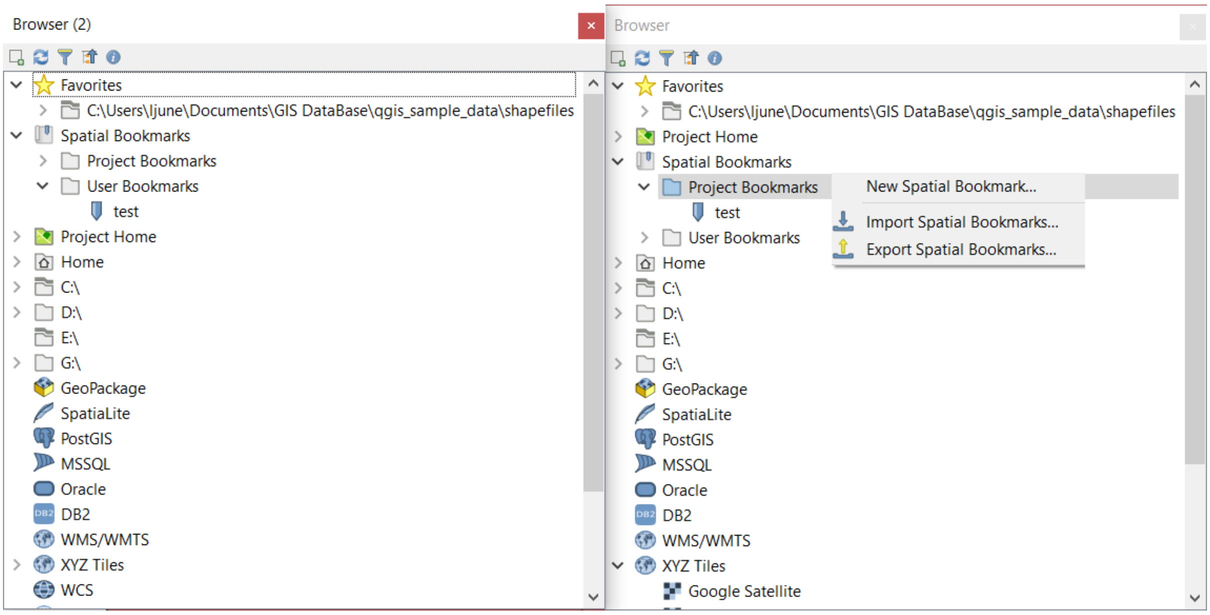


Abb. 13.2: QGIS Browser panels side-by-side

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.

13.1.2 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, MSSQL, DB2, 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*.

13.1.3 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.

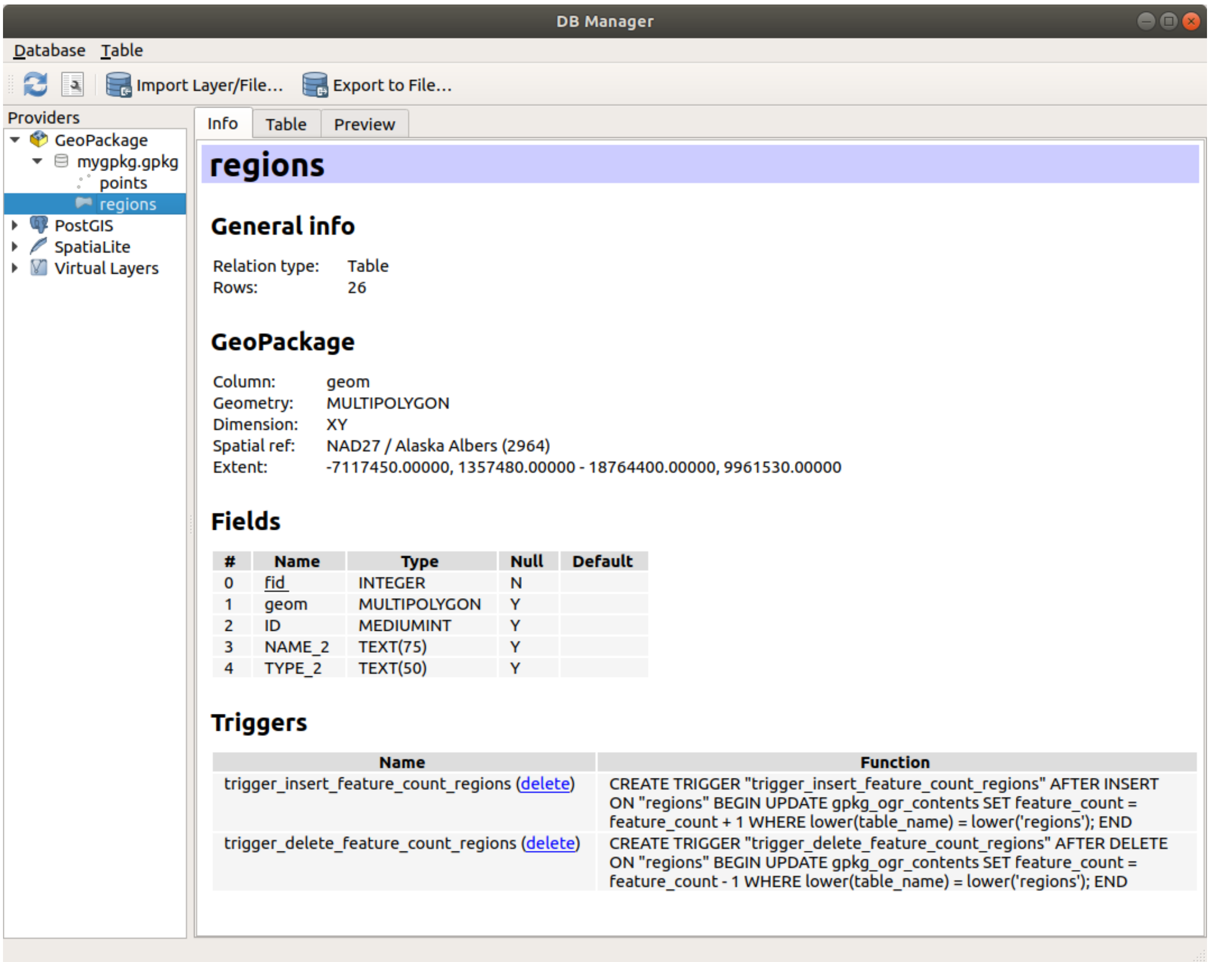


Abb. 13.3: DB Manager dialog

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.

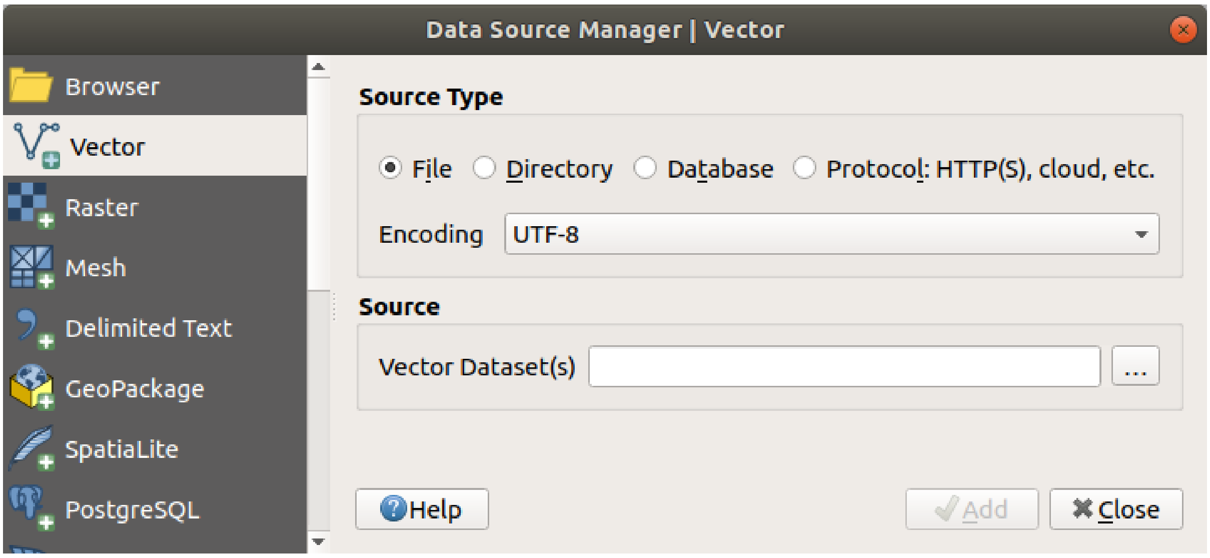


Abb. 13.4: Add Vector Layer Dialog

   - 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.

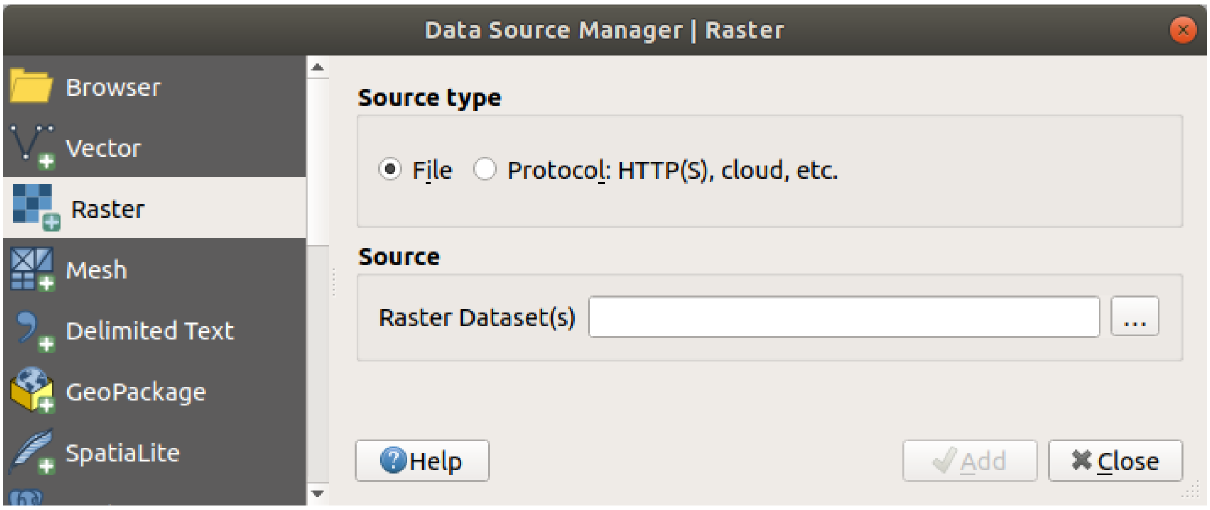


Abb. 13.5: Add Raster Layer Dialog

2. Check *File* source type
3. Click on the … Browse 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
6. You can specify the encoding for vector file if desired
7. Press *Add* to load the file in QGIS and display them in the map view. *figure_vector_loaded* shows QGIS after loading the `alaska.shp` file.

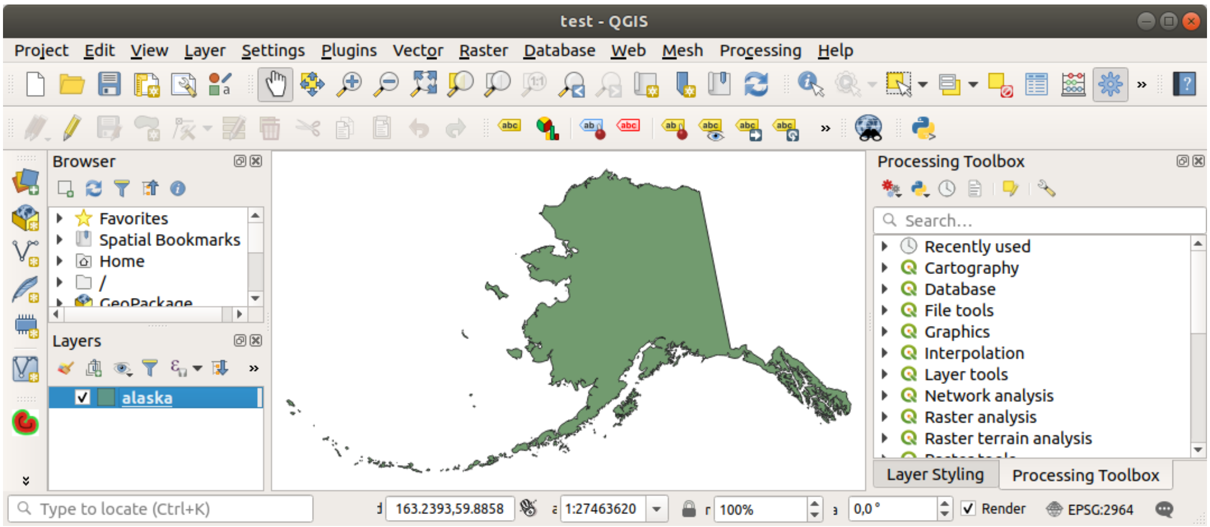


Abb. 13.6: QGIS with Shapefile of Alaska loaded

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`, `MSSQL` 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*.

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 Open Data Source Manager button.
2. Enable the *Mesh* tab on the left panel
3. Press the … Browse button to select the file. *Various formats* are supported.
4. Select the layer and press *Add*. The layer will be added using the native mesh rendering.

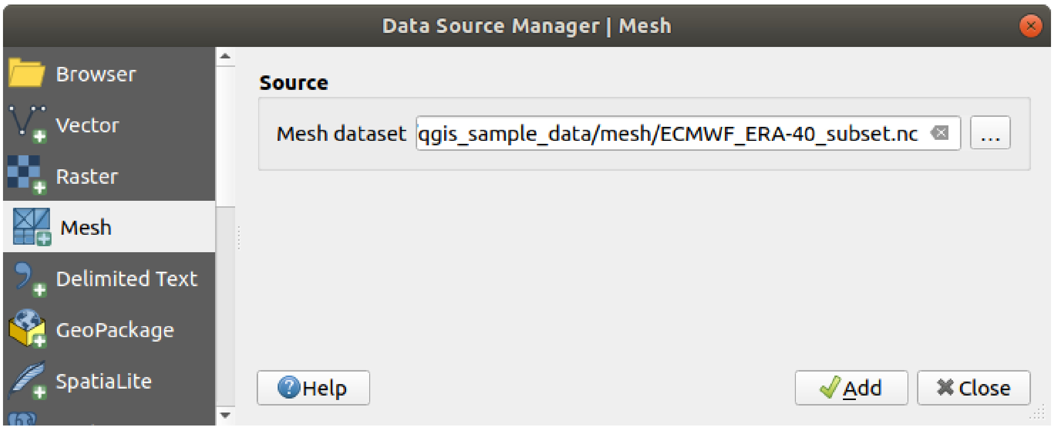


Abb. 13.7: Mesh tab in Data Source Manager

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 … Browse 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.

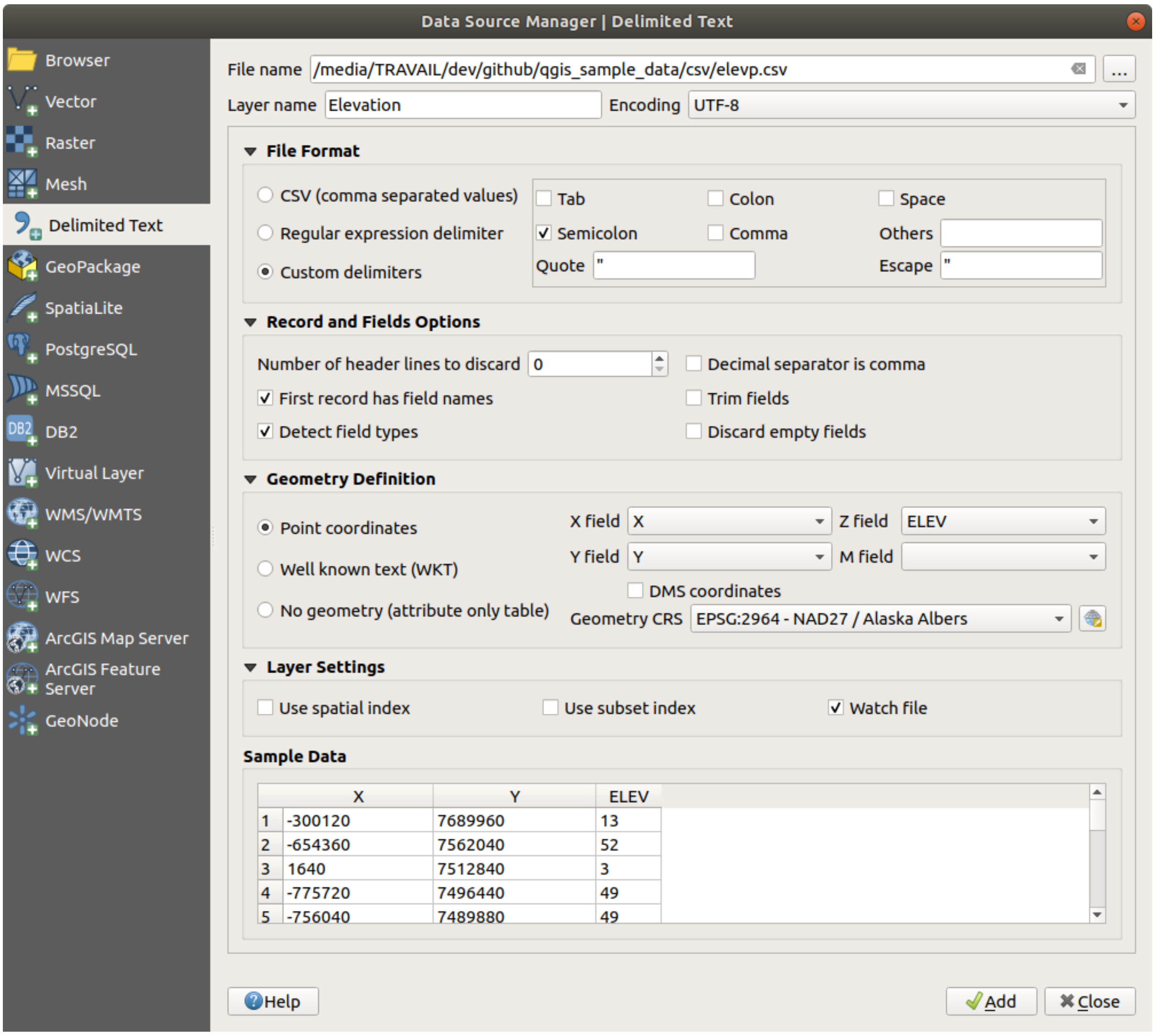


Abb. 13.8: Delimited Text Dialog

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*.

As you set the parser properties, a sample data preview updates at the bottom of the dialog.

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 approriate *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.

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 *Target package*, i.e. the new GeoPackage file that will store the data. If an existing file is provided, then it will be overwritten.
2. Specify the coordinate reference system of the data in the drawing file.
3. Check *Expand block references* to import the blocks in the drawing file as normal elements.
4. Check *Use curves* to promote the imported layers to a `curved` geometry type.
5. Use the *Import* button to select the DWG/DXF file to use (one per geopackage). 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.
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.

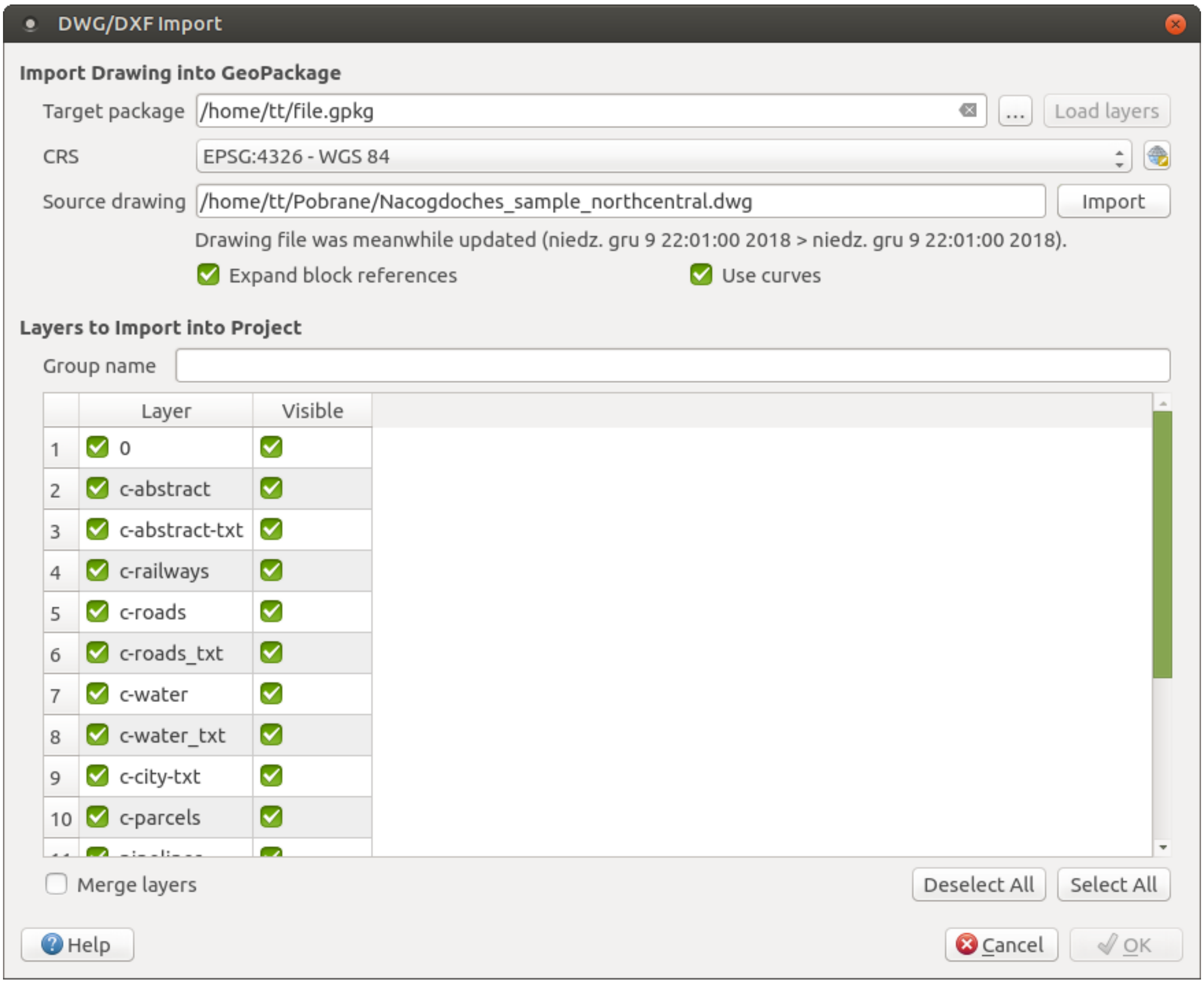


Abb. 13.9: Import dialog for DWG/DXF files

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

Loading GPS data in QGIS can be done using the core plugin `GPS Tools`. Instructions are found in section *GPS Plugin*.

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 recommanded 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 MSSQL Spatial Layer*
- *Add Oracle Spatial Layer…* or by typing `Ctrl+Shift+O`
- *Add DB2 Spatial Layer…* or by typing `Ctrl+Shift+2`

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 MSSQL Spatial*;
- *Connecting to Oracle Spatial*;
- *Connecting to DB2 Spatial*.

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 *figure_add_postgis_tables*). To access the connection manager, click on the *New* button to display the *Create a New PostGIS Connection* dialog.

The parameters required for a PostGIS 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 PostGIS 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.

  Warnung: 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

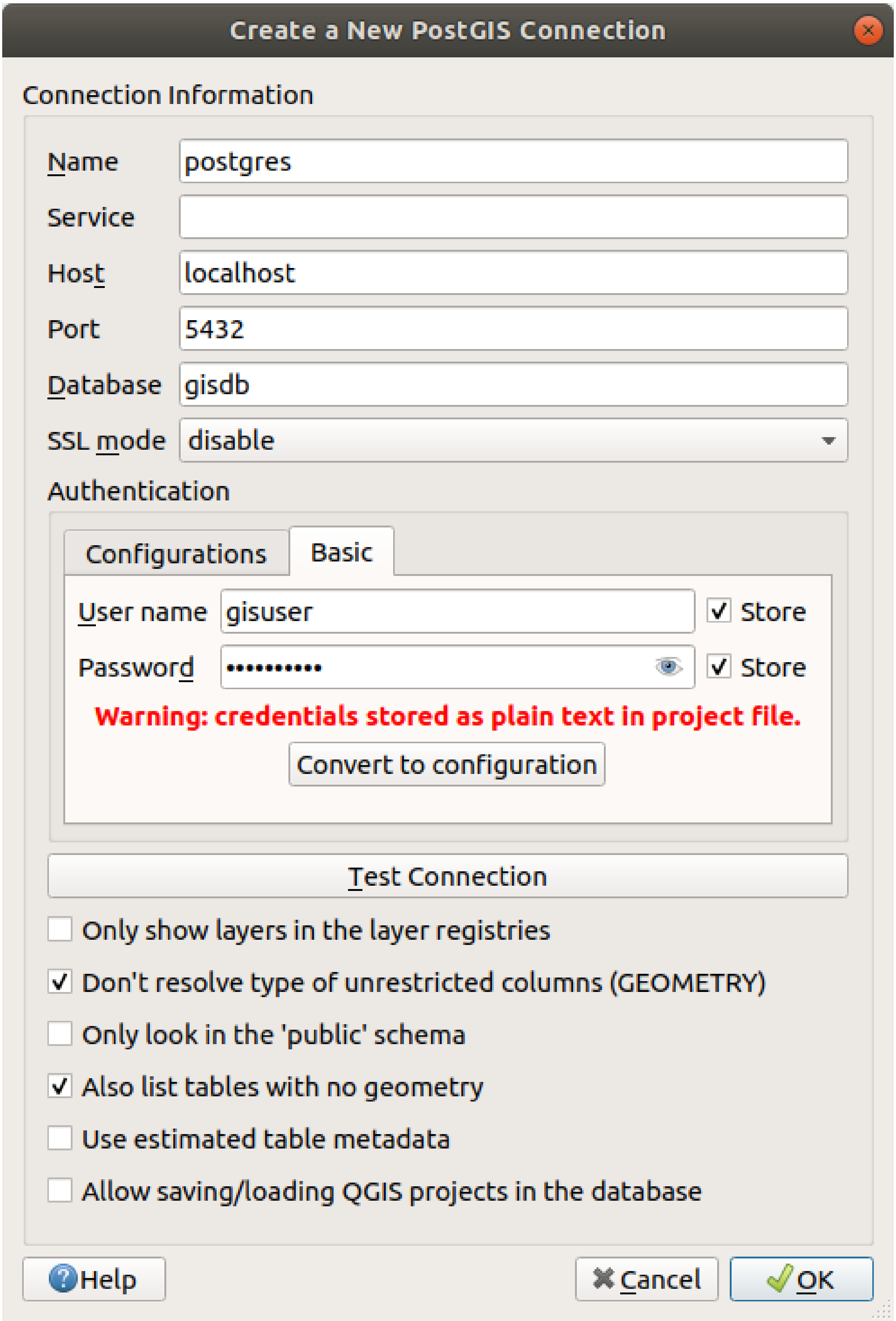


Abb. 13.10: Create a New PostGIS Connection Dialog

> (*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*

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 *figure_add_postgis_tables*).

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 need to do something like `export PGSERVICE=water_service` before doing your psql commands.

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.

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`;
- **Workspace**: Workspace to switch to.

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 DB2 Spatial

In addition to some of the options described in *Creating a stored Connection*, the connection to a DB2 database (see *DB2 Spatial Layers* for more information) can be specified using either a *Service/DSN* name defined to ODBC or *Driver*, *Host* and *Port*.

An ODBC **Service/DSN** connection requires the service name defined to ODBC.

A driver/host/port connection requires:

- **Driver**: Name of the DB2 driver. Typically this would be IBM DB2 ODBC DRIVER.
- **DB2 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.
- **DB2 Port**: Port number the DB2 database server listens on. The default DB2 LUW port is `50000`. The default DB2 z/OS port is `446`.

Tipp: DB2 Spatial Layers

A DB2 Spatial layer is defined by a row in the **DB2GSE.ST_GEOMETRY_COLUMNS** view.

Bemerkung: In order to work effectively with DB2 spatial tables in QGIS, it is important that tables have an INTEGER or BIGINT column defined as PRIMARY KEY and if new features are going to be added, this column should also have the GENERATED characteristic.

It is also helpful for the spatial column to be registered with a specific spatial reference identifier (most often `4326` for WGS84 coordinates). A spatial column can be registered by calling the `ST_Register_Spatial_Column` stored procedure.

Connecting to MSSQL Spatial

In addition to some of the options in *Creating a stored Connection*, creating a new MSSQL connection dialog proposes you to fill a **Provider/DSN** name. You can also display available databases.

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.

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.

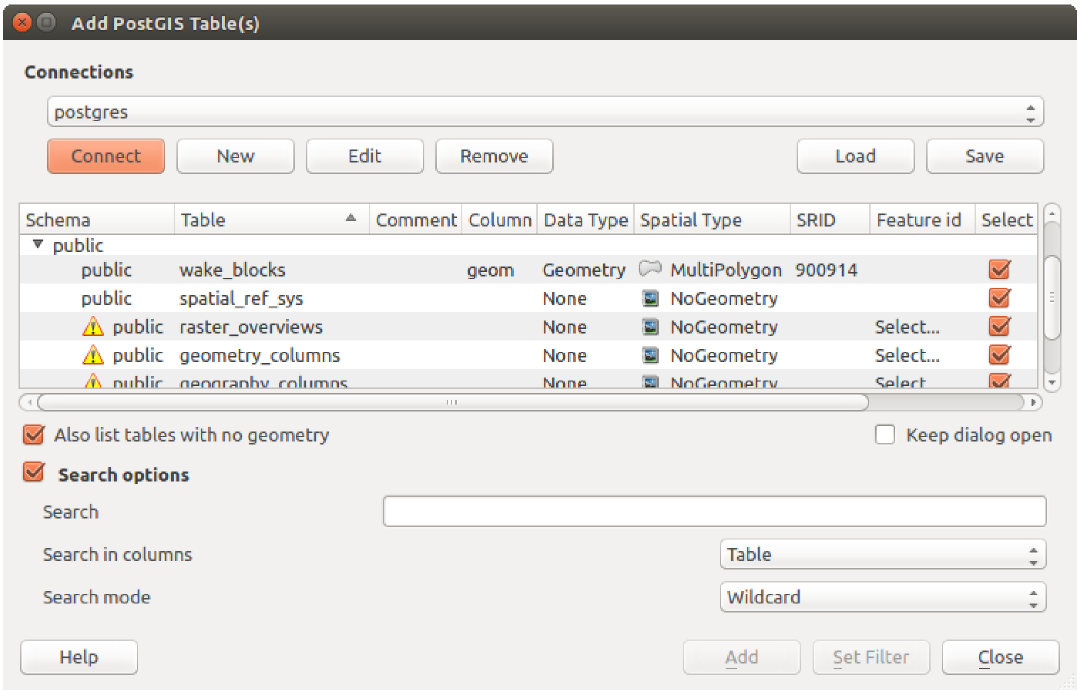


Abb. 13.11: Add PostGIS Table(s) Dialog

13.1.4 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)

13.1.5 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.

13.1.6 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. Chapter *Arbeiten mit OGC Daten* contains descriptions of these capabilities.

Using XYZ Tile services

XYZ Tile services can be found in the *XYZ Tiles* top level entry in the *Browser*. By default, the OpenStreetMap XYZ Tile service is configured. You can add other services that use the XYZ Tile protocol by choosing *New Connection* in the XYZ Tiles context menu (right-click to open). *figure_xyz_tiles_openstreetmap* shows the dialog with the OpenStreetMap XYZ Tile service configuration.

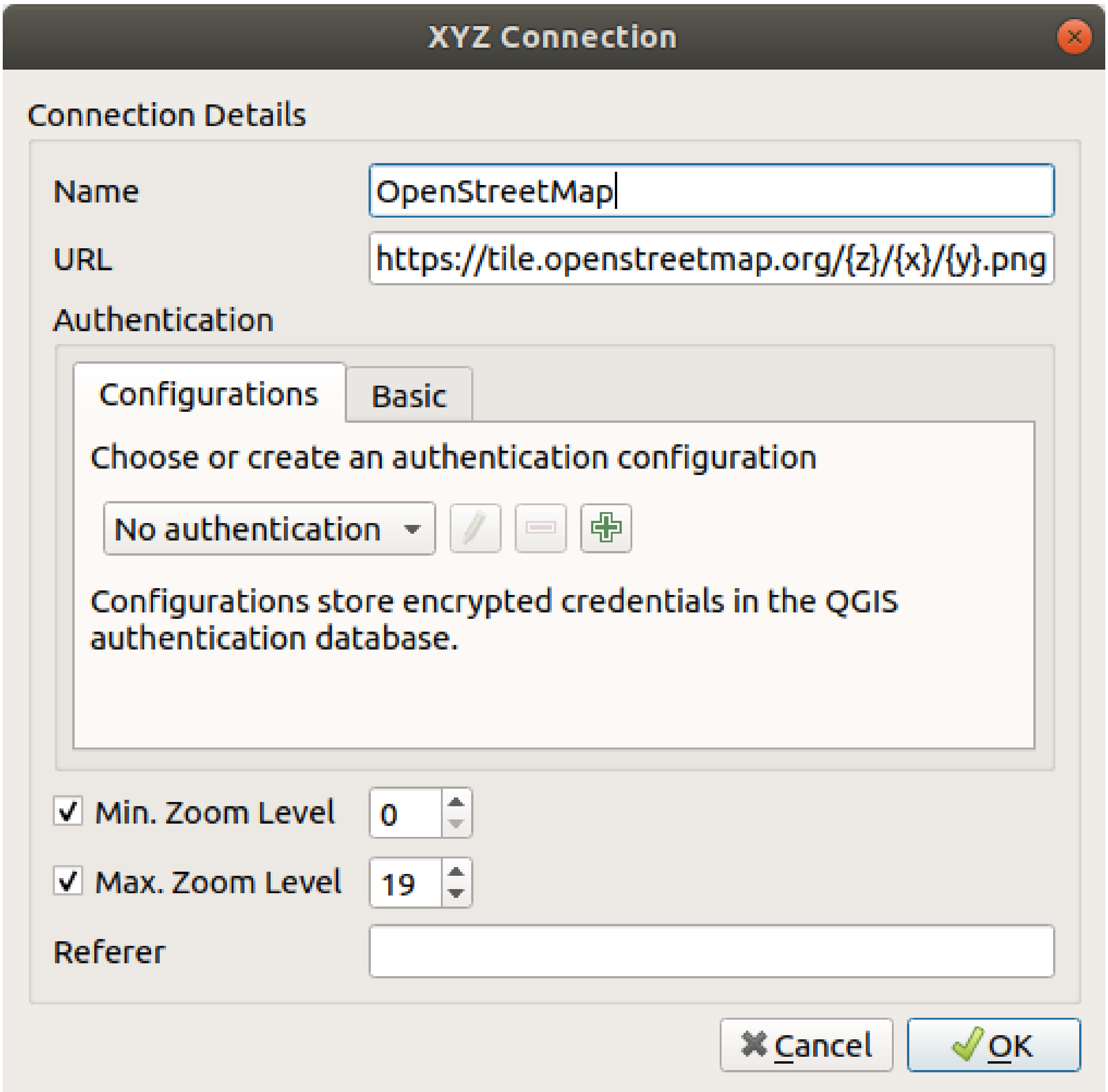


Abb. 13.12: XYZ Tiles - OpenStreetMap configuration

Configurations can be saved (*Save Connections*) to XML and loaded (*Load Connections*) through the context menu. Authentication configuration is supported. 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" password="" name="OpenStreetMap" username=""
   authcfg="" referer=""/>
</qgsXYZTilesConnections>
```

Once a connection to a XYZ tile service is set, right-click over the entry to:

- *Edit…* the XYZ connection settings
- *Delete* the connection
- *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.

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.

13.1.7 Handling broken file paths

When the path to a data source is wrong, QGIS opens the *Handle Unavailable Layers* dialog. You can double-click in the *Datasource* field or click *Browse* to fix the path. It is possible to continue working with your project with the broken path by clicking *Keep Unavailable Layers*. Your layer is then displayed in the *Layers* panel, but without any data until you fix the path using the Unavailable layer! icon next to it in the *Layers* panel, or *Change Data Source…* in the layer contextual menu. Another possibility is to *Remove Unavailable Layers*. As the last step, click *Apply changes*.

When a layer path has been fixed, QGIS scans through all other broken paths and tries to auto-fix those that have the same broken file path.

13.2 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.

13.2.1 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

Um einen neuen GeoPackage-Layer zu erstellen, drücken Sie die Schaltfläche *Neuer GeoPackage Layer…* im Menü *Layer ‣ Layer erstellen ‣* oder aus der Symbolleiste *Datenquellen-Verwaltung*. Das Dialogfeld *Neuer GeoPackage Layer* wird angezeigt, wie in *figure_create_geopackage* gezeigt.

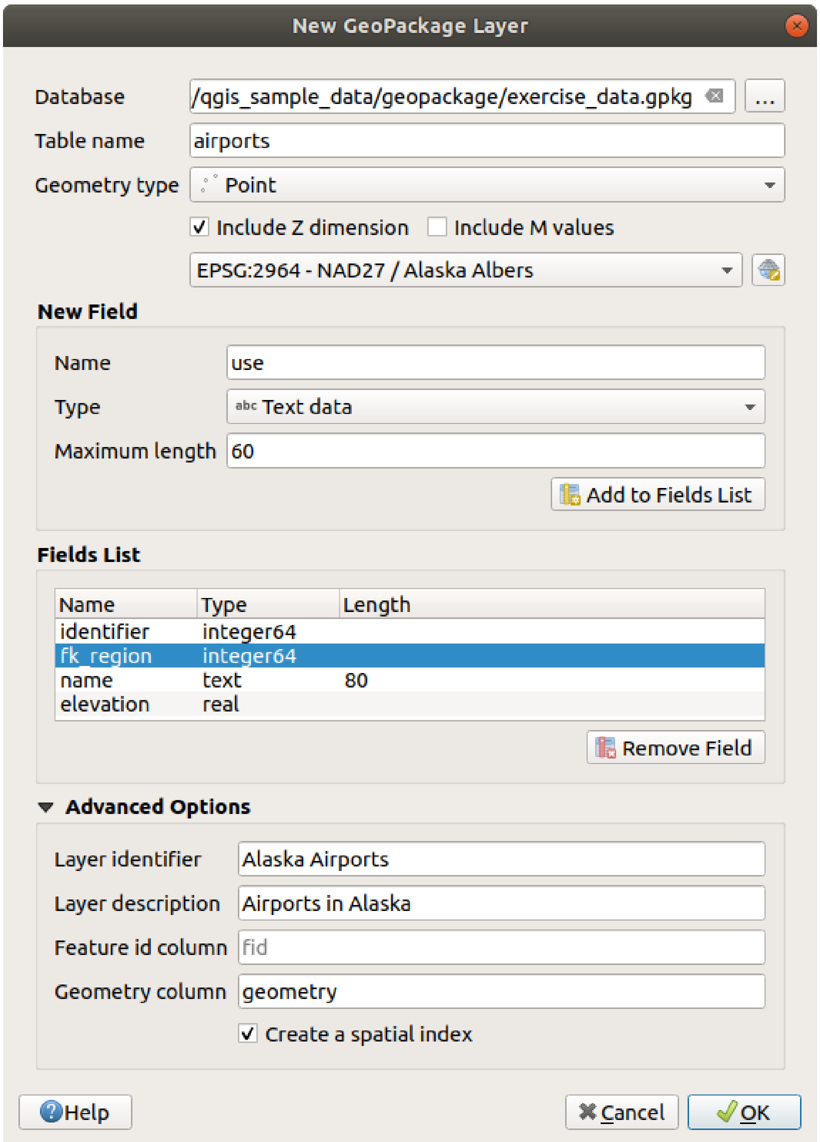


Abb. 13.13: Dialogfenster zum Erstellen eines neuen GeoPackage-Layers

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 *Geometrietyp*. Wenn es sich nicht um einen geometrielosen 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

Um einen neuen Layer im ESRI Shapefile-Format zu erstellen, drücken Sie die Schaltfläche *Neuer Shapedatei Layer…* im Menü *Layer ▸ Layer erstellen ▸* oder aus der Symbolleiste *Datenquellenverwaltung*. Das Dialogfeld *Neuer Shapedatei Layer* wird wie in *figure_create_shapefile* angezeigt.

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 (point, multipoint, line or polygon)
4. Specify whether the geometry should have *Z (+ M values)* or *M values*
5. 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. 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.

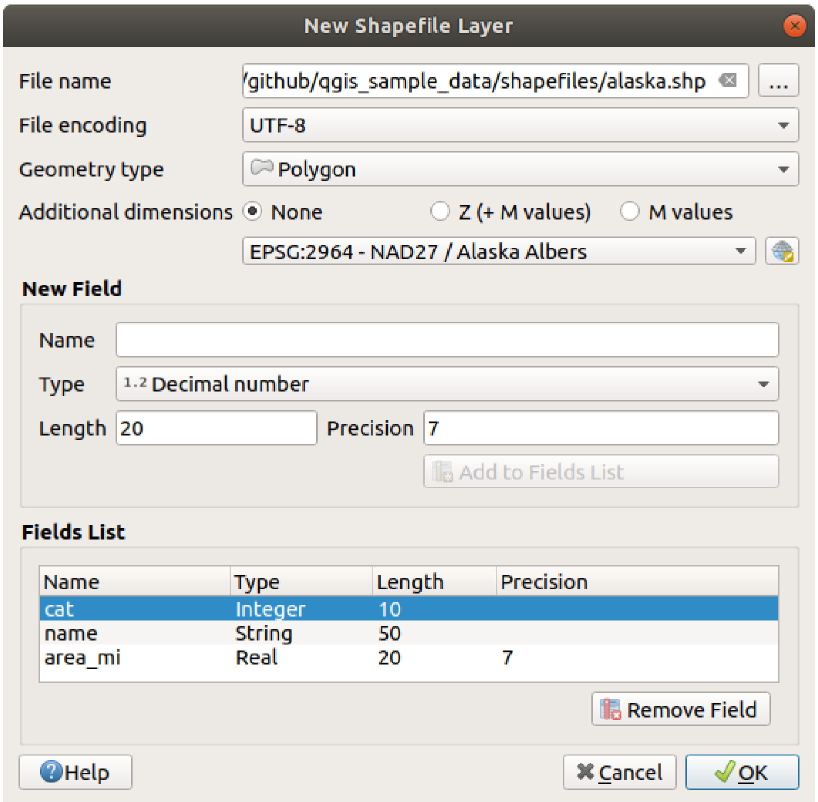


Abb. 13.14: Dialogfenster zum Erstellen eines neuen Shapedatei-Layers

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 *Figure_create_spatialite*.

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 *Geometrietyp*. Wenn es sich nicht um einen geometrielosen 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* and *Decimal number*.
3. Klicken Sie auf die Schaltfläche *Zur Feldliste hinzufügen*.
4. Wiederholen Sie die obigen Schritte für jedes Feld, das Sie hinzufügen müssen
5. 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).

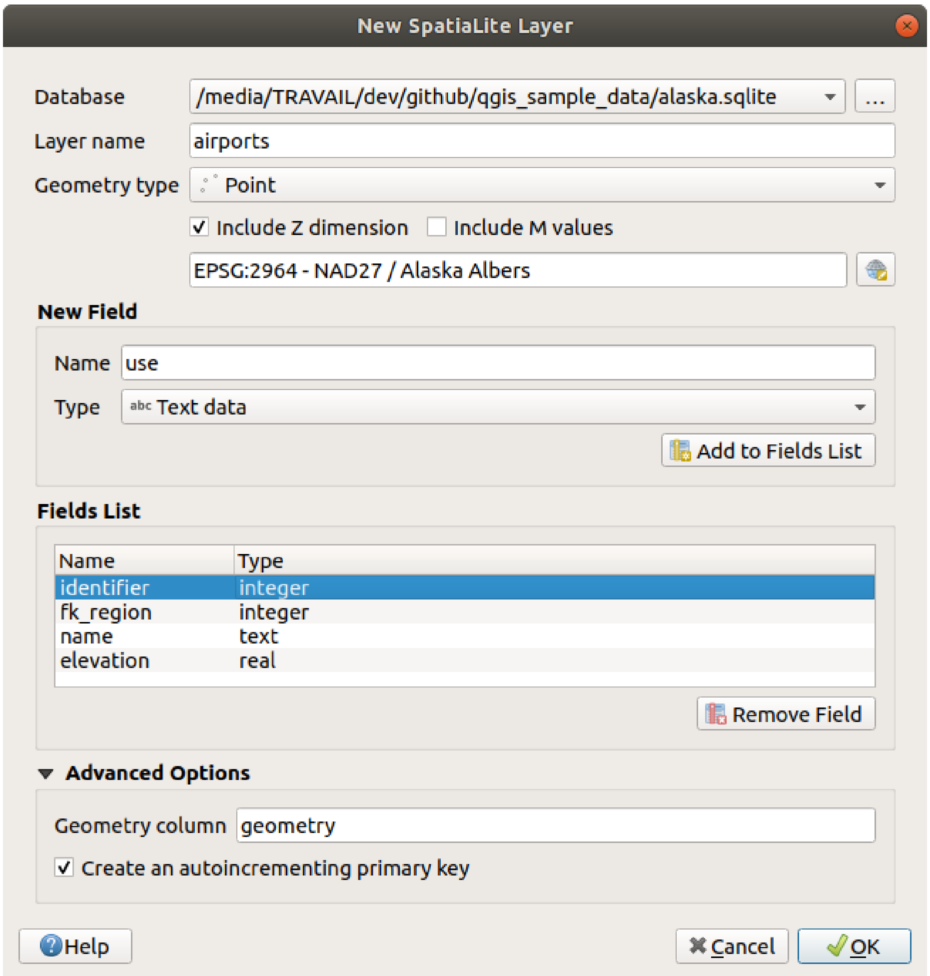


Abb. 13.15: Creating a New SpatiaLite layer dialog

Further management of SpatiaLite layers can be done with *DB Manager*.

Creating a new GPX layer

To create a new GPX file, you first need to load the GPS plugin. *Plugins* → *Plugin Manager…* opens the Plugin Manager Dialog. Activate the *GPS Tools* checkbox.

When this plugin is loaded, choose *Create Layer* → *Create new GPX Layer…* from the *Layer* menu. In the dialog, choose where to save the new file and press *Save*. Three new layers are added to the *Layers Panel*: `waypoints`, `routes` and `tracks`.

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 *figure_create_temporary*. 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. Specify the coordinate reference system using the button.

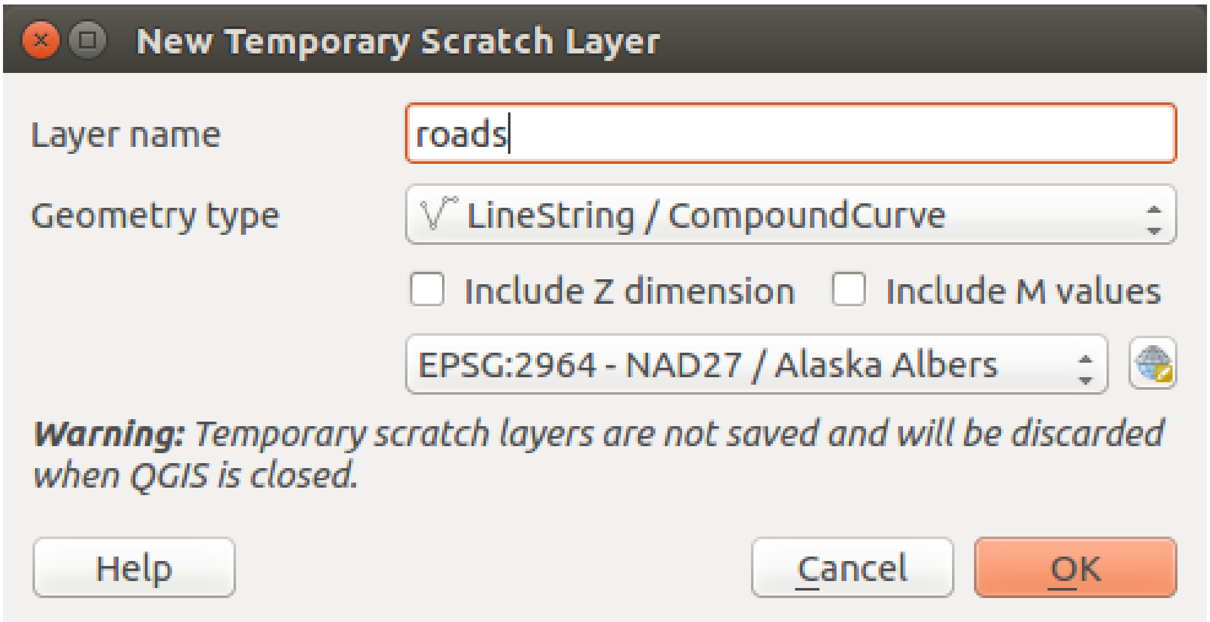


Abb. 13.16: Creating a new Temporary Scratch layer dialog

By default, a new temporary scratch layer is created without any attributes. You can later add them using the New Field button in the layer's attribute table dialog or the *Fields* tab of its properties 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.

13.2.2 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* (possible values are **layer**, **Map view** or **user-defined** extent)
- *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*

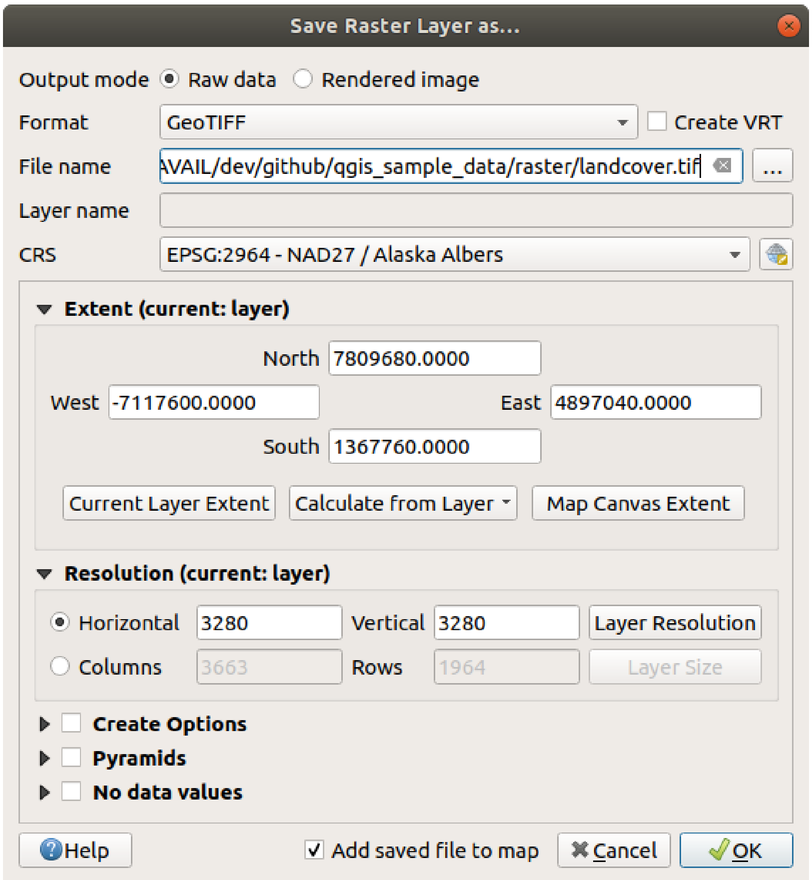


Abb. 13.17: Saving as a new raster layer

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*. In case you set your fields behavior with some *Edit widgets*, e.g. `value map`, you can keep the displayed values in the layer by checking *Replace all selected raw fields values by displayed values*.
- *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.

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.

13.2.3 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:

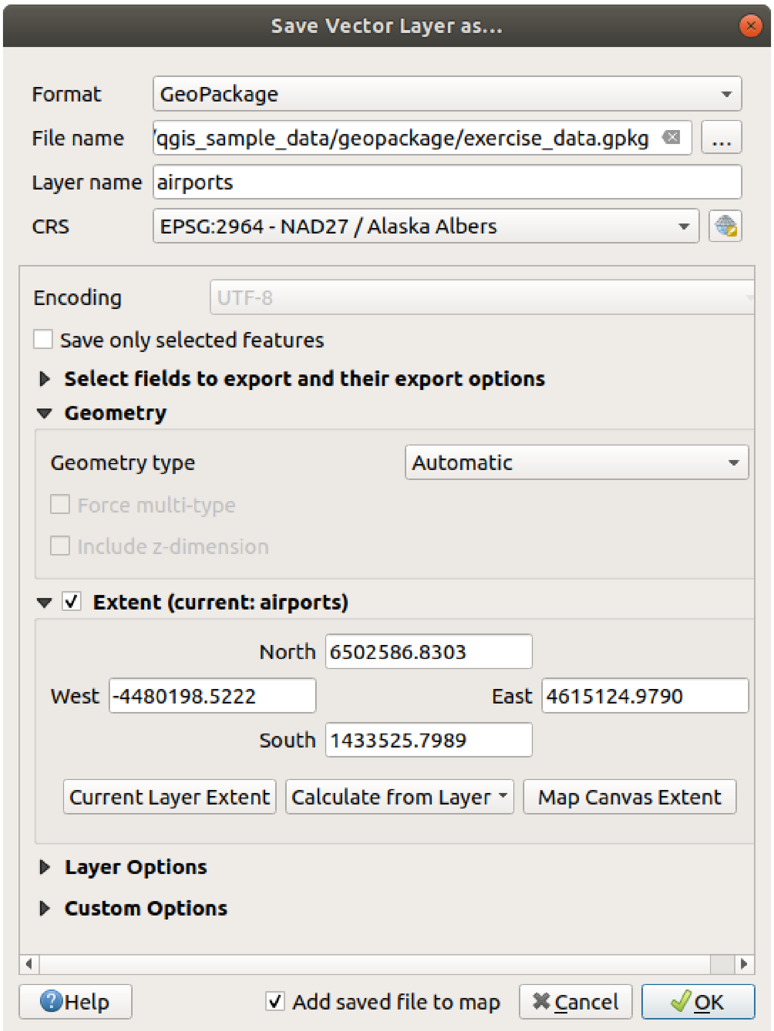


Abb. 13.18: Saving as a new vector layer

- *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.

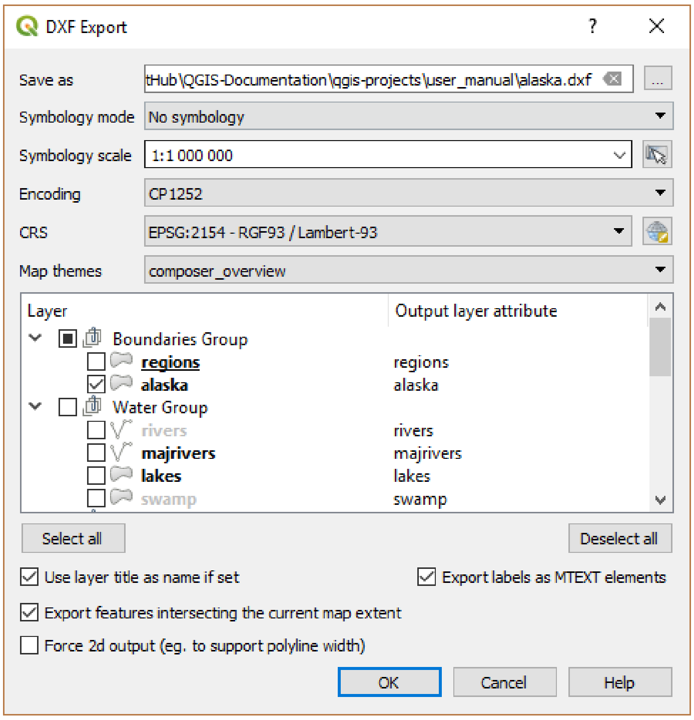


Abb. 13.19: Exporting a project to DXF dialog

13.2.4 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).

13.2.5 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.

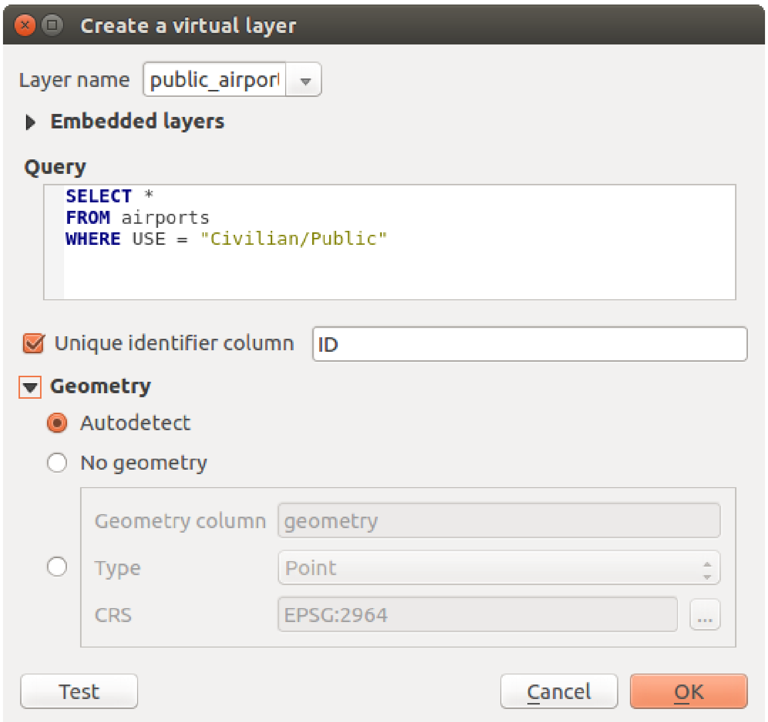


Abb. 13.20: Create virtual layers dialog

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.

13.3 Exploring Data Formats and Fields

13.3.1 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.

13.3.2 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, DB2, Oracle Spatial, MSSQL Spatial databases, and many more), QGIS may handle some of their properties differently. Support is provided by the OGR Simple Feature Library. This section describes how to work with these specificities.

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 at https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

GDAL 3.1 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 must 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 OGR driver assumes all fields are strings (i.e. text) unless it is told otherwise. You can create a CSVT file to tell OGR (and QGIS) the data type of the different columns:

Type	Name	Example
Whole number	Integer	4
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

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.

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 OGR 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.

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.

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:

```
SET XML OPTION DOCUMENT;
```

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 OGR 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 **shp2pgsl** 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).

Tipp: 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 sidx_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$
```

Vector layers crossing 180° longitude

Many GIS packages don't wrap vector maps with a geographic reference system (lat/lon) crossing the 180 degrees longitude line (http://postgis.refractions.net/documentation/manual-2.0/ST_Shift_Longitude.html). As result, if we open such a map in QGIS, we could see two widely separated locations, that should appear near each other. In *Figure_vector_crossing*, 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.

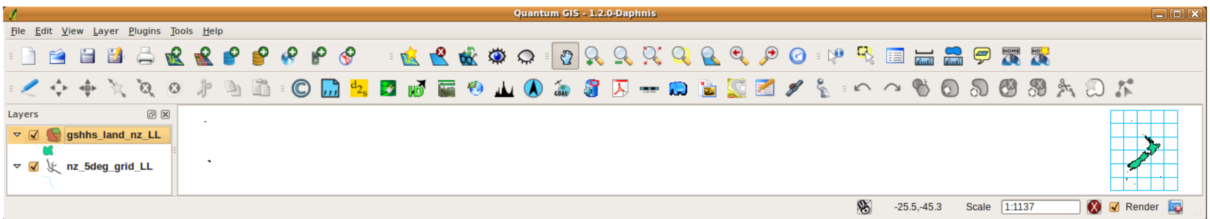


Abb. 13.21: Map in lat/lon crossing the 180° longitude line

A work-around is to transform the longitude values using PostGIS and the **ST_Shift_Longitude** function. This function reads every point/vertex in every component of every feature in a geometry, and if the longitude coordinate is < 0°, it adds 360° to it. The result is a 0° - 360° version of the data to be plotted in a 180°-centric map.

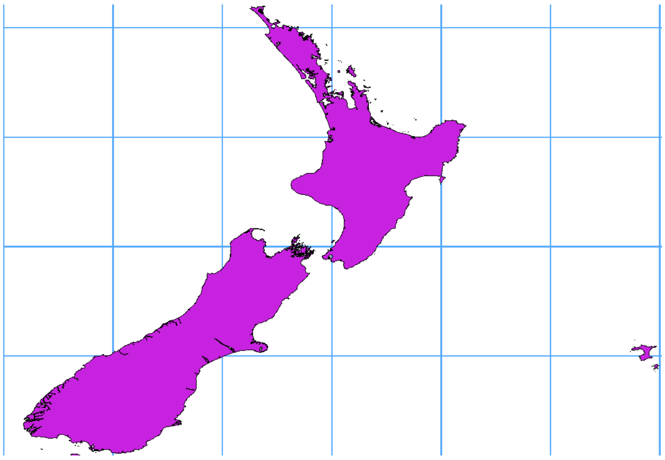


Abb. 13.22: Crossing 180° longitude applying the **ST_Shift_Longitude** function

Anwendung

- Import data into PostGIS (*Importing Data into PostgreSQL*) using, for example, the DB Manager plugin.
- Use the PostGIS command line interface to issue the following command (in this example, „TABLE“ is the actual name of your PostGIS table): `gis_data=# update TABLE set the_geom=ST_Shift_Longitude(the_geom);`
- 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*).

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/drv_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 availabe 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.

DB2 Spatial Layers

IBM DB2 for Linux, Unix and Windows (DB2 LUW), IBM DB2 for z/OS (mainframe) and IBM DashDB products allow users to store and analyse spatial data in relational table columns. The DB2 provider for QGIS supports the full range of visualization, analysis and manipulation of spatial data in these databases.

User documentation on these capabilities can be found at the DB2 z/OS KnowledgeCenter, DB2 LUW KnowledgeCenter and DB2 DashDB KnowledgeCenter.

For more information about working with the DB2 spatial capabilities, check out the DB2 Spatial Tutorial on IBM DeveloperWorks.

The DB2 provider currently only supports the Windows environment through the Windows ODBC driver.

The client running QGIS needs to have one of the following installed:

- DB2 LUW
- IBM Data Server Driver Package
- IBM Data Server Client

To open a DB2 data in QGIS, see the *The Browser Panel* or *Loading a Database Layer* section.

If you are accessing a DB2 LUW database on the same machine or using DB2 LUW as a client, the DB2 executables and supporting files need to be included in the Windows path. This can be done by creating a batch file like the following with the name **db2.bat** and including it in the directory **%OSGEO4W_ROOT%/etc/ini**:

```
@echo off
REM Point the following to where DB2 is installed
SET db2path=C:\Program Files (x86)\sqllib
REM This should usually be ok - modify if necessary
SET gskpath=C:\Program Files (x86)\ibm\gsk8
SET Path=%db2path%\BIN;%db2path%\FUNCTION;%gskpath%\lib64;%gskpath%\lib;%path%
```

KAPITEL 14

Arbeiten mit Vektordaten

14.1 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, form design) zu verwalten. Es bietet auch Informationen über die 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 vector *Layer Properties* dialog provides the following sections:

Information	*Source*	*Symbology*[1]
Labels[1]	*Diagrams*	*3D View*[1]
Fields	*Attributes Form*	*Joins*
Auxiliary Storage	*Actions*	*Display*
Rendering	*Variables*	*Metadata*
Dependencies	*Legend*	*QGIS Server*
Digitizing	*External plugins*[2] tabs	

[1] Also available in the *Layer styling panel*

[2] *External plugins* you install can optionally add tabs to this dialog. Those are not presented in this document. Refer to their documentation.

Tipp: Share full or partial properties of the layer styles

The *Style* menu at the bottom of the dialog allows you to import or export these or part of these properties from/to several destination (file, clipboard, database). See *Managing Custom Styles*.

Bemerkung: Because properties (symbology, label, actions, default values, forms…) of embedded layers (see *Layer/Gruppen einbinden*) 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.

14.1.1 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 (format of storage, path, geometry type, data source encoding, extent…);
- picked from the *filled metadata* (access, links, contacts, history…);
- or related to its geometry (spatial extent, CRS…) or its attributes (number of fields, characteristics of each…).

14.1.2 Source Properties

Use this tab to define general settings for the vector layer.

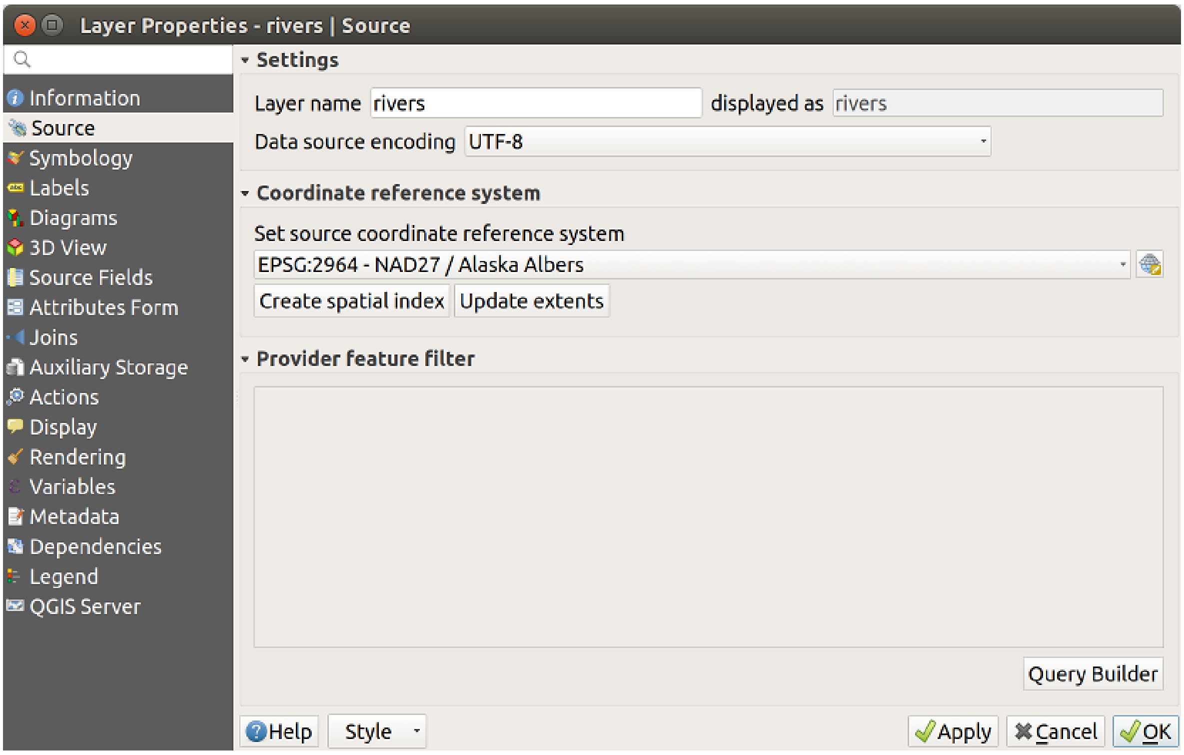


Abb. 14.1: Source tab in vector Layer Properties dialog

Other than setting the *Layer name* to display in the *Layers Panel*, available options include:

Koordinatenbezugssystem

- Displays the layer's *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 eponym 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 *Figure_vector_querybuilder*. 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, MSSQL…) 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 **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).

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.

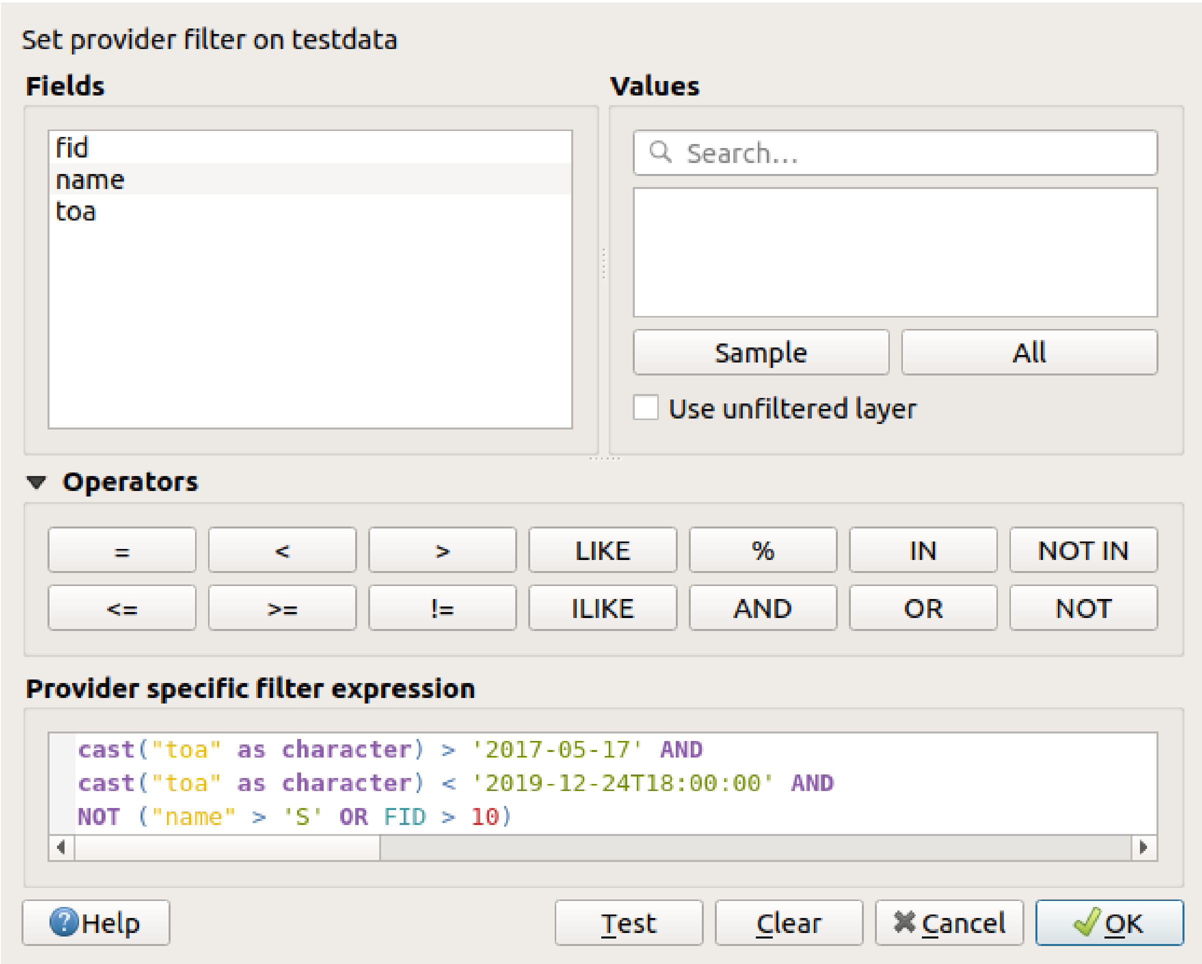


Abb. 14.2: Abfrageeditor

Tipp: Filtered layers are indicated in the Layers Panel

In the *Layers* panel, filtered layer is listed with a Filter 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.

14.1.3 Symbology Properties

The Symbology 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 a point displacement and a heatmap renderers available while polygon layers can also be rendered with the inverted polygons and 2.5 D renderers.

Es gibt keinen kontinuirliche 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 Symbollayertypen 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 Änderunge funktioniert. Wenn Sie den Darstellungstyp wiederholt ändern gehen die Einstellungen für das Symbol verloren.

Einzelsymbol 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.

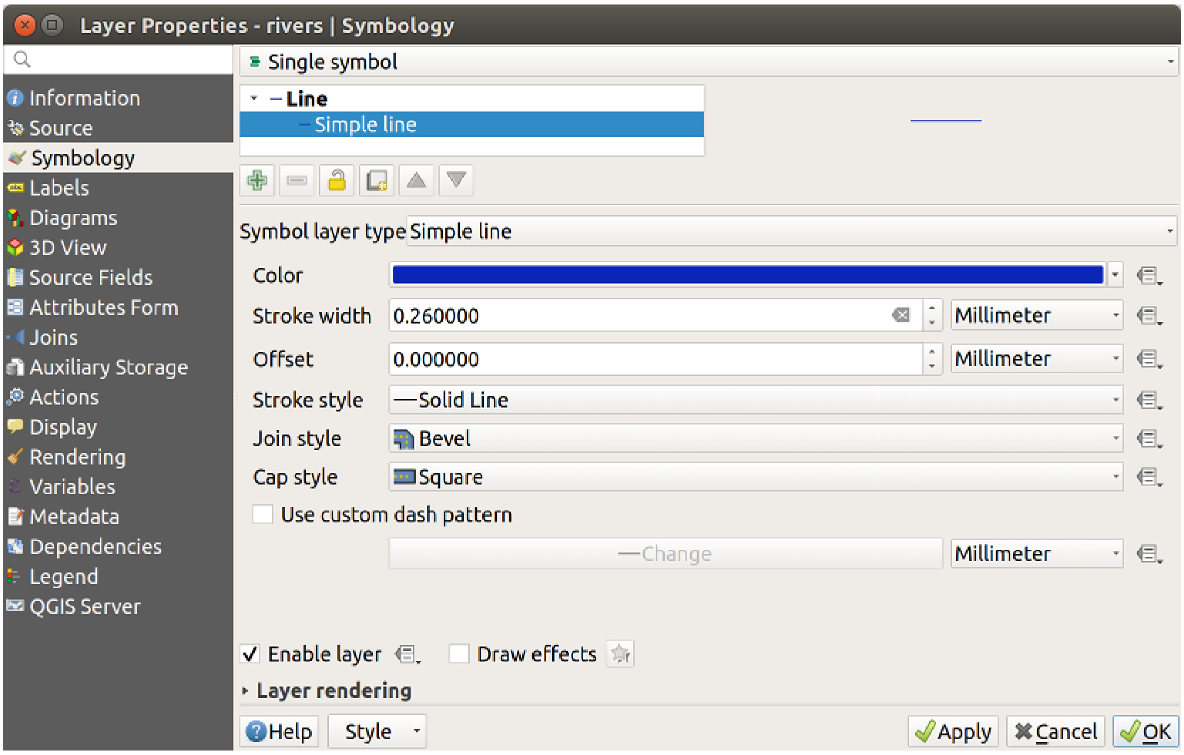


Abb. 14.3: Linieneigenschaften Einzelsymbol

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.

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 ε button. Using expressions for categorizing avoids the need to create an ad hoc field for symbology purposes (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:

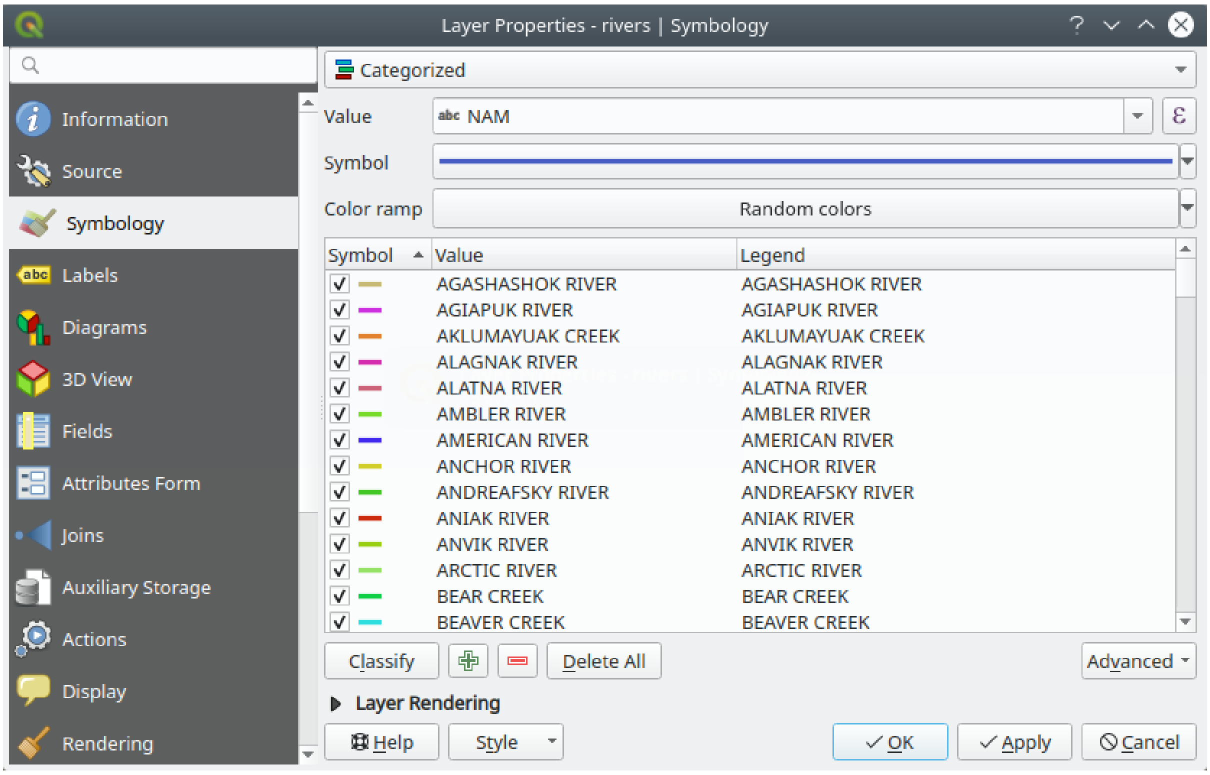


Abb. 14.4: Kategorisierte Symbolisierungsoptionen

- 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
```

Tipp: 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.

  Tipp: 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 *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.

Tipp: Edit categories directly from the *Layers* panel

When a layer symbology is based on a *categorized*, *graduated* or *rule-based* symbology mode, you can edit each of the categories from the *Layers* Panel. Right-click on a sub-item of the layer and you will:

- *Toggle items* visibility
- *Show all items*
- *Hide all items*
- Modify the symbol color thanks to the *color selector* wheel
- *Edit symbol…* from the *symbol selector* dialog
- *Copy symbol*
- *Paste symbol*

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 (using the fields listbox or the Set value expression function)
- 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 Interval: each class has the same size (e.g. values from 0 to 16 and 4 classes, each class has a size of 4).
- Quantile: each class will have the same number of element inside (the idea of a boxplot).
- Natural Breaks (Jenks): the variance within each class is minimal while the variance between classes is maximal.
- Standard Deviation: classes are built depending on the standard deviation of the values.
- 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://astrostatistics.psu.edu/datasets/R/html/base/html/pretty.html)

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 *figure_graduated_symbology* shows the graduated rendering dialog for the major_rivers layer of the QGIS sample dataset.

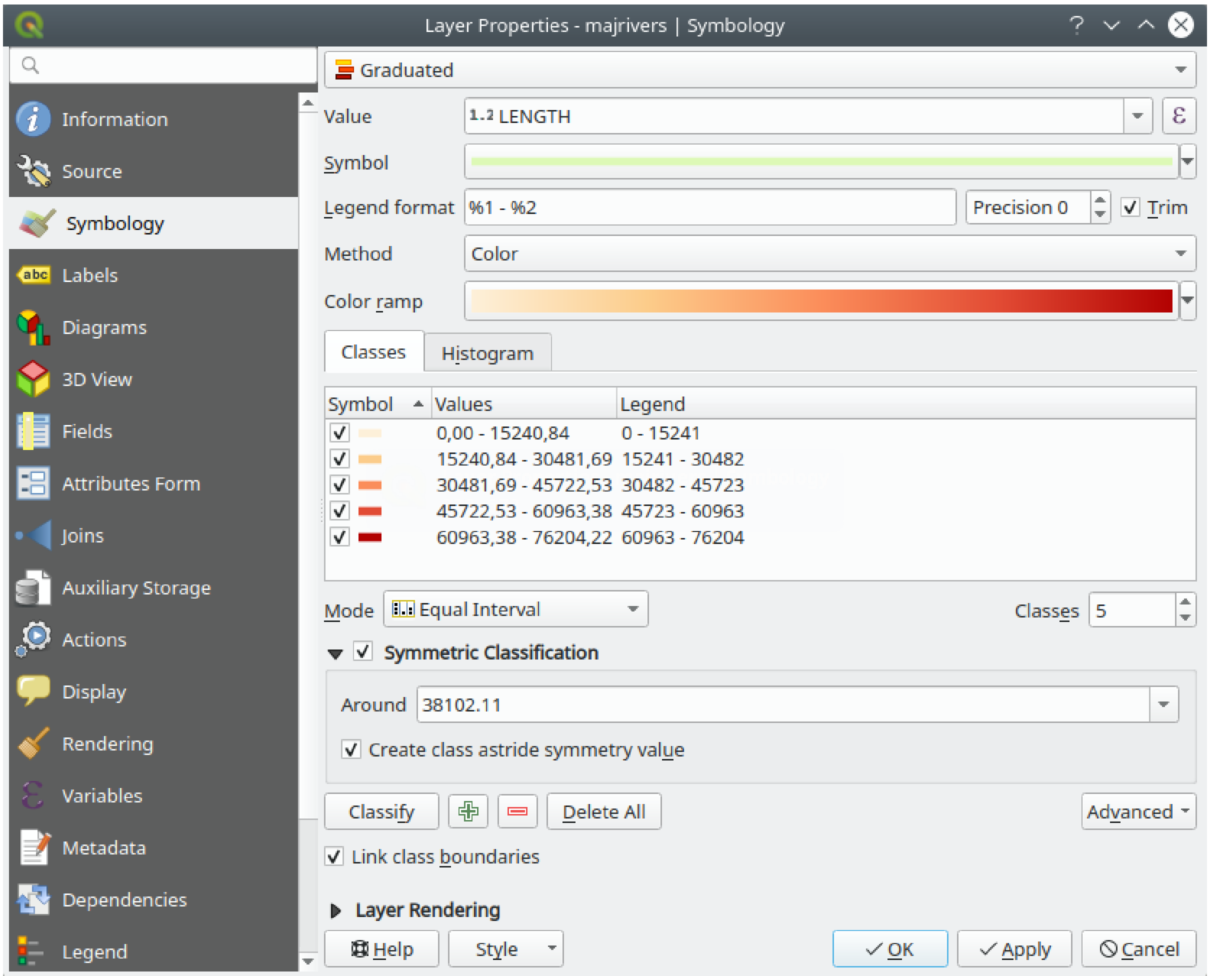


Abb. 14.5: Abgestufte Symbolisierungsoptionen

Tipp: Thematische Karten anhand von Ausdrücken erstellen

Categorized and graduated thematic maps can be created using the result of an expression. In the properties dialog for vector layers, the attribute chooser is extended with a ε Set column expression function. So you don't need to write the classification attribute to a new column in your attribute table if you want the classification attribute to be a composite of multiple fields, or a formula of some sort.

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).

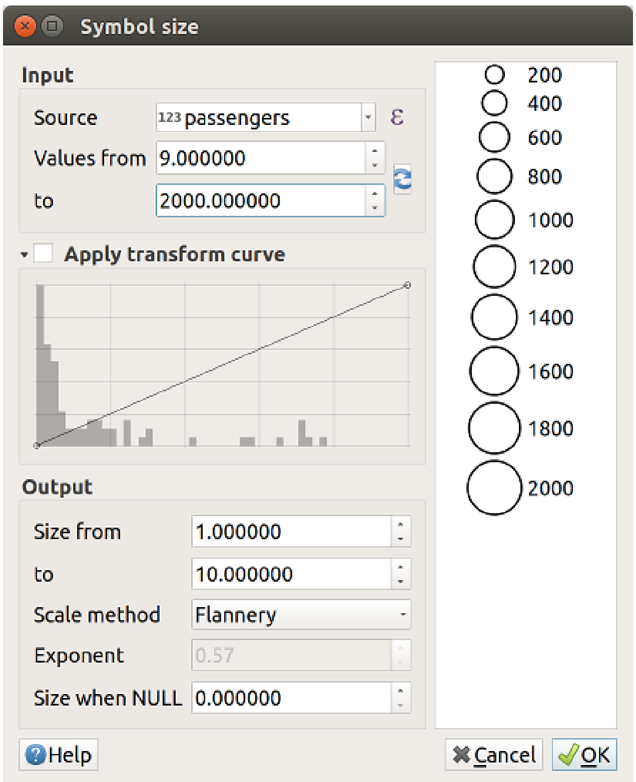


Abb. 14.6: Scaling airports size based on elevation of the airport

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.

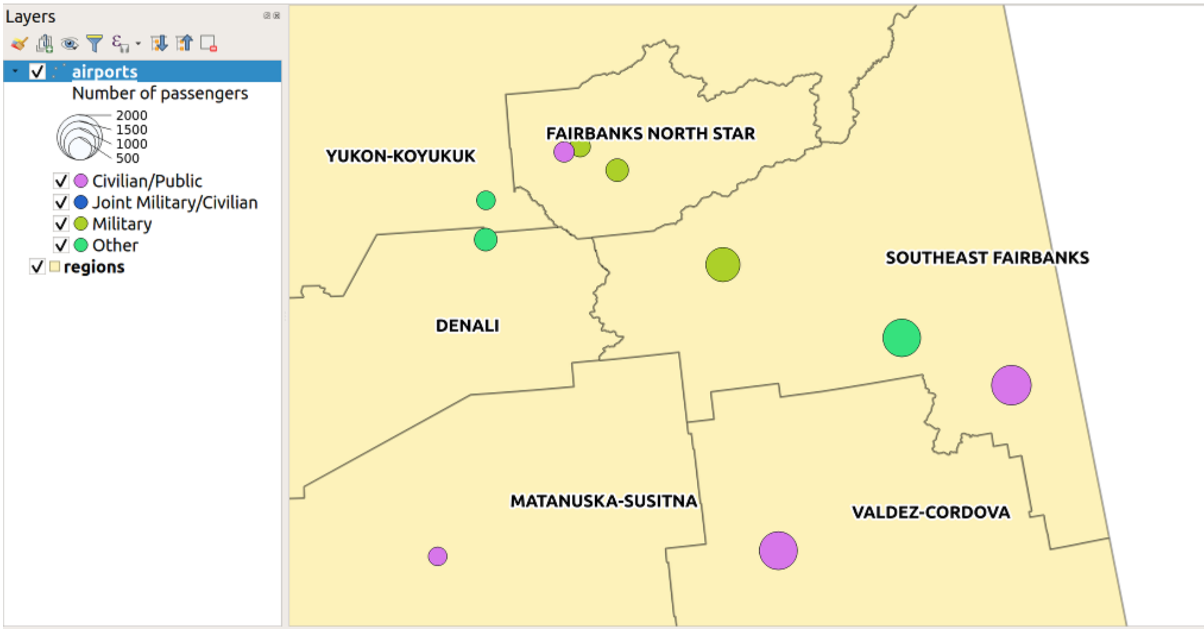


Abb. 14.7: Multivariate example with scaled size legend

Rule-based Renderer

The *Rule-based* renderer is used to render all the features from a layer, using rule-based symbols whose aspect reflects the assignment of a selected feature's attribute to a class. The rules are based on SQL statements and can be nested. The dialog allows rule grouping by filter or scale, and you can decide if you want to enable symbol levels or use only the first-matched rule.

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 ε 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.

Selecting a rule, you can also organize its features in subclasses using the *Refine selected rules* drop-down menu. Automated rule refinement can be based on:

- **scales**;
- **categories**: applying a *categorized renderer*;
- or **ranges**: applying a *graduated renderer*.

Refined classes appear like sub-items of the rule, in a tree hierarchy and like above, you can set symbology of each class.

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** (categorized renderer) or **Ranges** (graduated renderer).

The created rules also appear in a tree hierarchy in the map legend. Double-click the rules in the map legend and the Symbology tab of the layer properties appears showing the rule that is the background for the symbol in the tree.

The example in *figure_rule_based_symbology* shows the rule-based rendering dialog for the rivers layer of the QGIS sample dataset.

Point displacement Renderer

The *Point Displacement* renderer works to visualize all features of a point layer, even if they have the same location. To do this, the renderer takes the points falling in a given *Distance* tolerance from each other and places them around their barycenter following different *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.

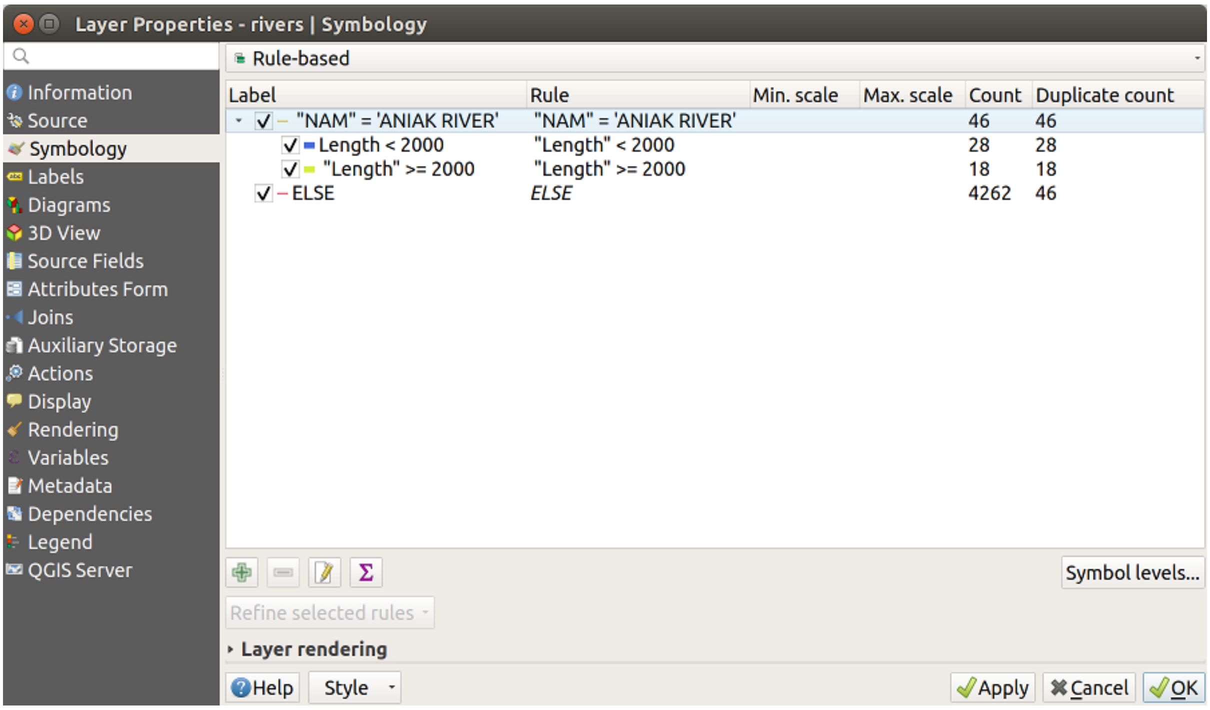


Abb. 14.8: Regelbasierte Symbolisierungsoptionen

The *Center symbol* widget helps you customize the symbol and color of the middle point. For the distributed points symbols, you can apply any of the *No symbols*, *Single symbol*, *Categorized*, *Graduated* or *Rule-based* renderer using the *Renderer* drop-down list and customize them using the *Renderer Settings…* button.

While the minimal spacing of the *Displacement lines* depends on the point symbol renderer's, you can still customize some of its settings such as the *Stroke width*, *Stroke color* and *Size adjustment* (eg, to add more spacing between the rendered points).

Use the *Labels* group options to perform points labeling: the labels are placed near the displaced position of the symbol, and not at the feature real position. Other than the *Label attribute*, *Label font* and *Label color*, you can set the *Minimum map scale* to display the labels.

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. Based on a specified *Distance*, points that fall within 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:

- 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.
- use the *Renderer* drop-down list to apply any of the other feature rendering types to the layer (single, categorized, rule-based…). Then, push the *Renderer Settings…* button to configure features' symbology as usual. Note that this renderer is only visible on features that are not clustered. Also, when the symbol color is the same for all the point features inside a cluster, that color sets the `@cluster_color` variable of the cluster.

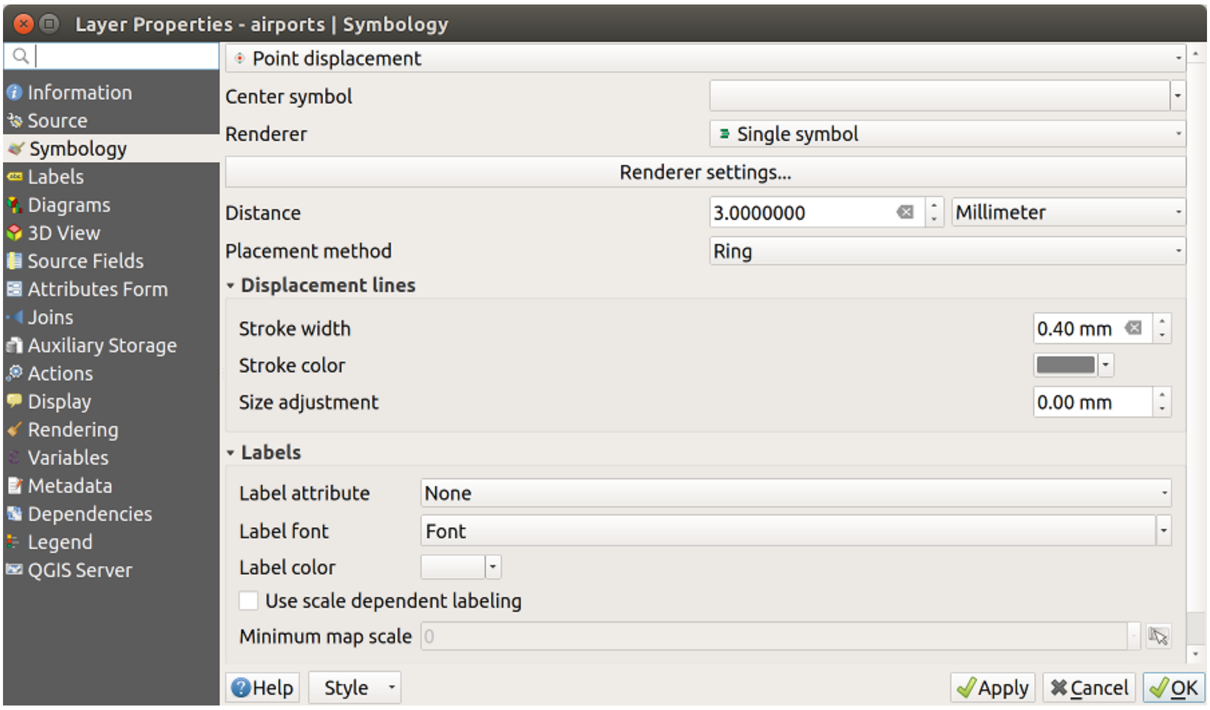


Abb. 14.9: Dialog Punktverdrängung

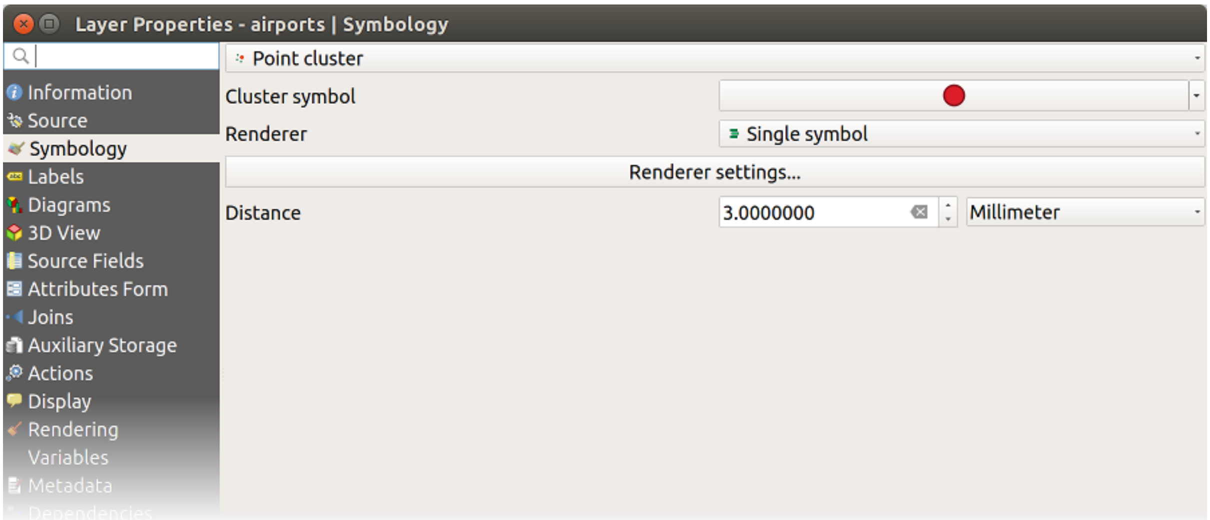


Abb. 14.10: Point Cluster dialog

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.

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.

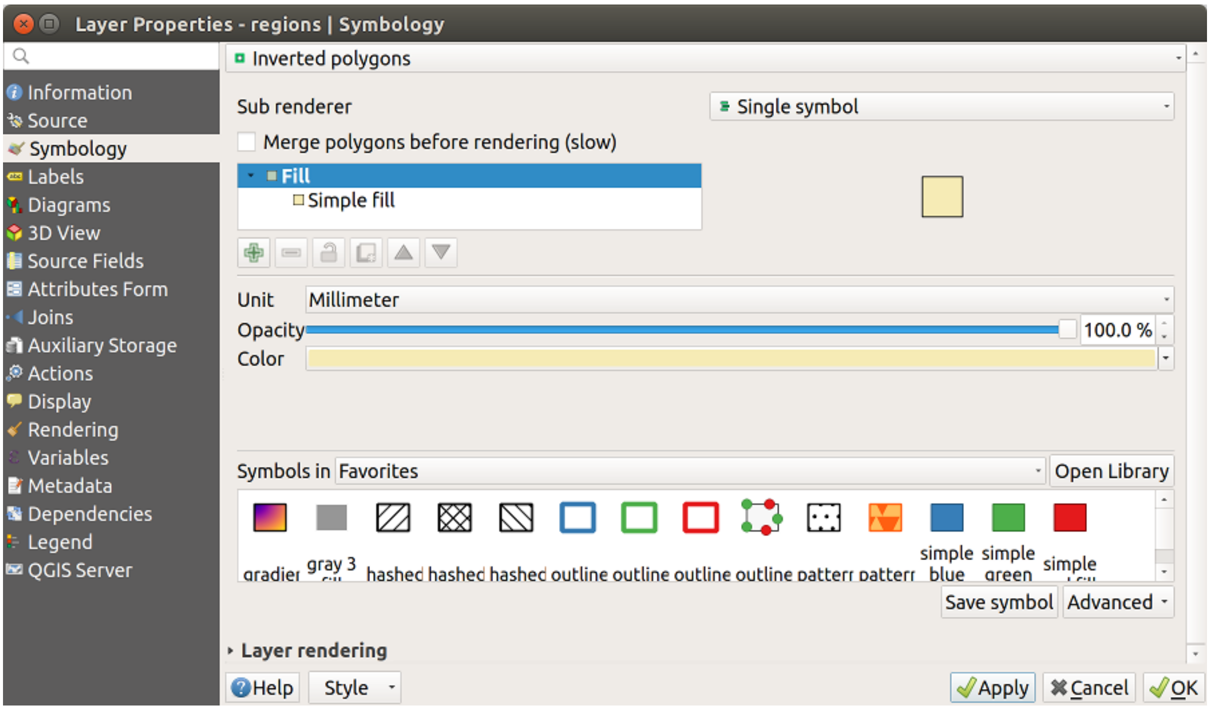


Abb. 14.11: Umgekehrte Polygone Dialog

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.

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).

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

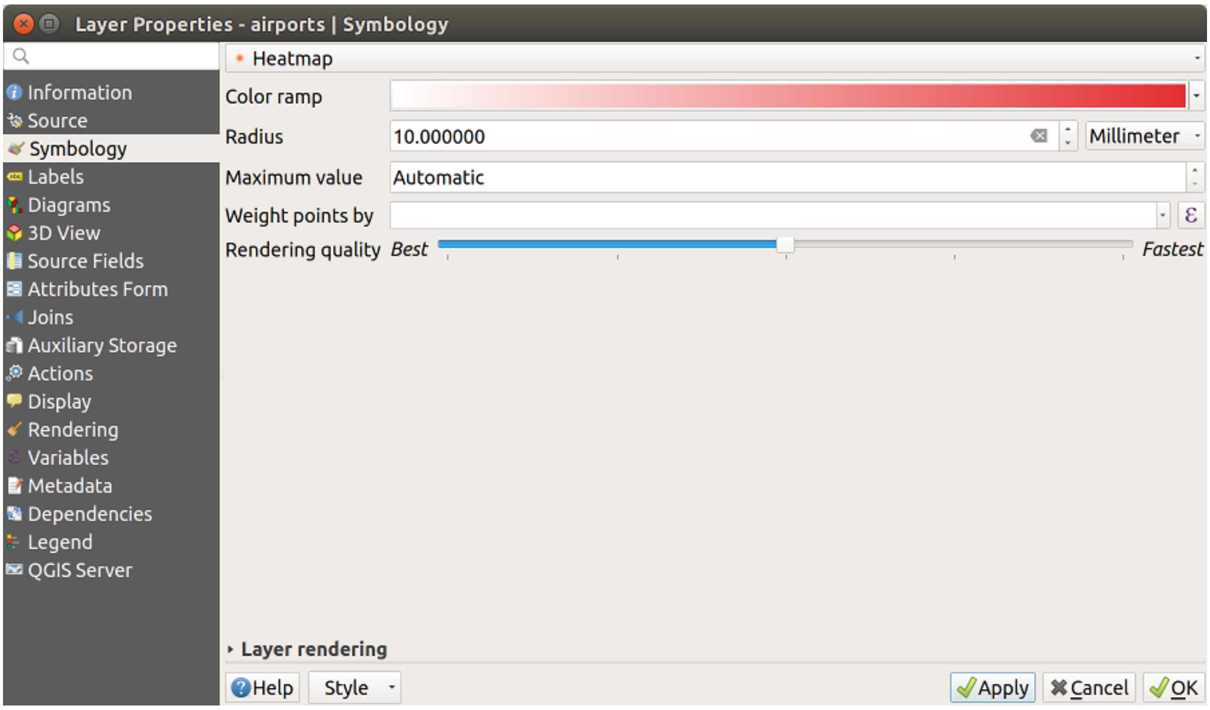


Abb. 14.12: Der Heatmap-Erweiterung Dialog

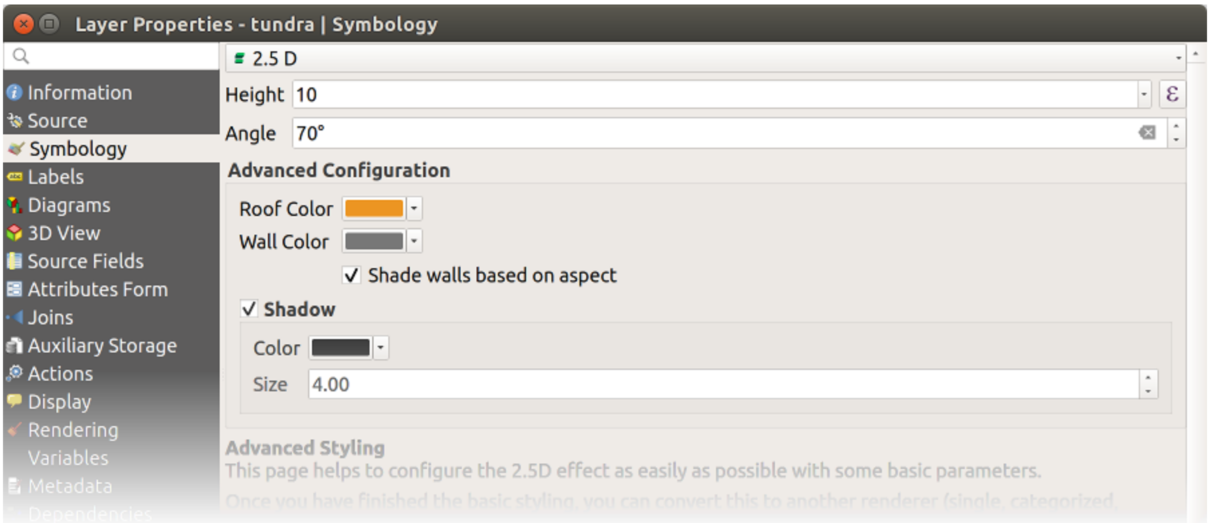


Abb. 14.13: 2.5D dialog

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.

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 underlaying 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…

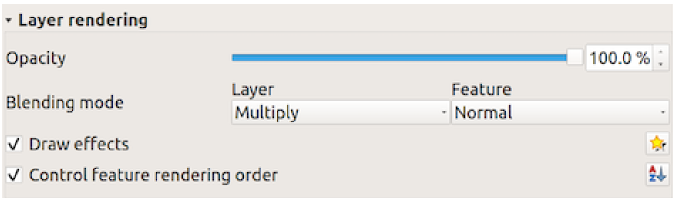


Abb. 14.14: Layerdarstellungsoptionen

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.

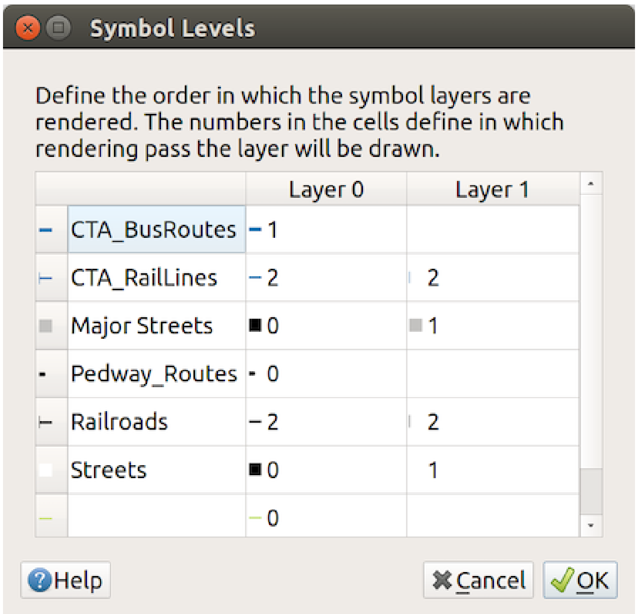


Abb. 14.15: Symbolebenen Dialog

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.

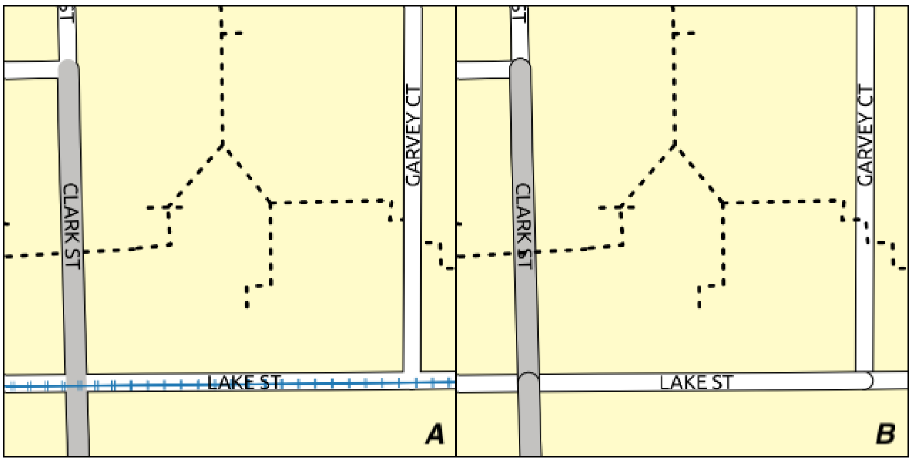


Abb. 14.16: Unterschied von aktivierten (A) und deaktivierten (B) Symbolebenen

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**;
- set 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.

A preview of the legend is displayed in the right panel of the dialog and updated as you set the parameters. For collapsed legend, a leader line from the horizontal center of the symbol to the corresponding legend text is drawn.

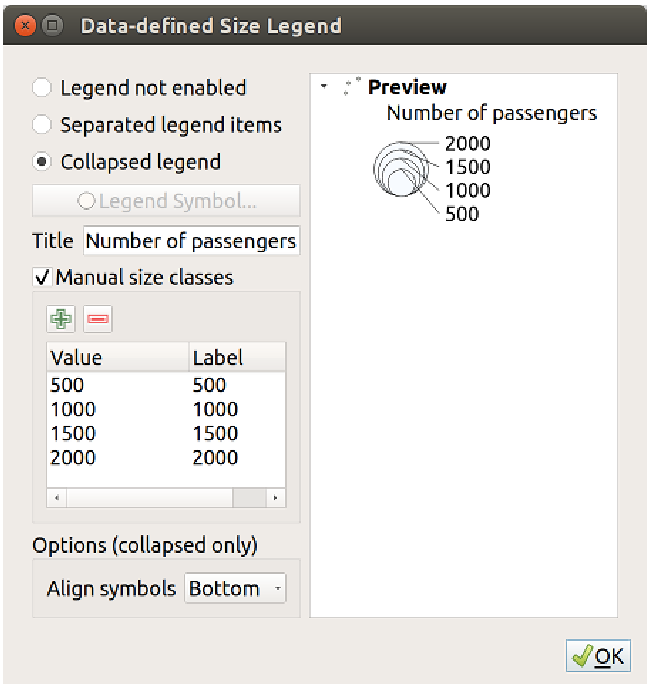


Abb. 14.17: Setting size scaled legend

Bemerkung: Currently, data-defined size legend for layer symbology can only be applied to point layer using single, categorized or graduated symbology.

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 *figure_effects_source*). 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.

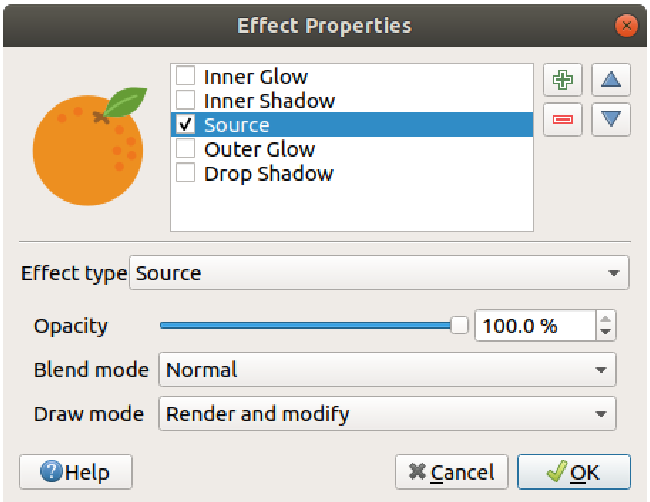


Abb. 14.18: Zeicheneffekte: Dialog Quelle

- **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*.
- **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.
- **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.
- **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.
- **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*.

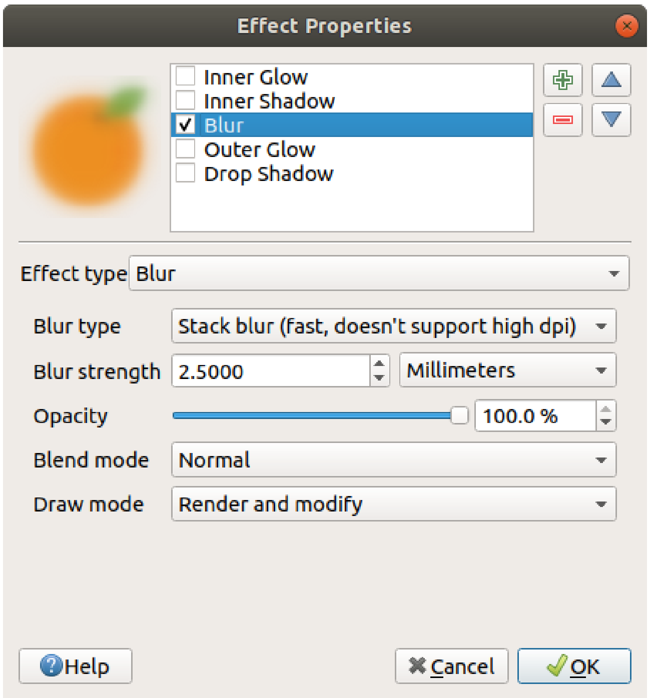


Abb. 14.19: Zeicheneffekte: Dialog verwischen

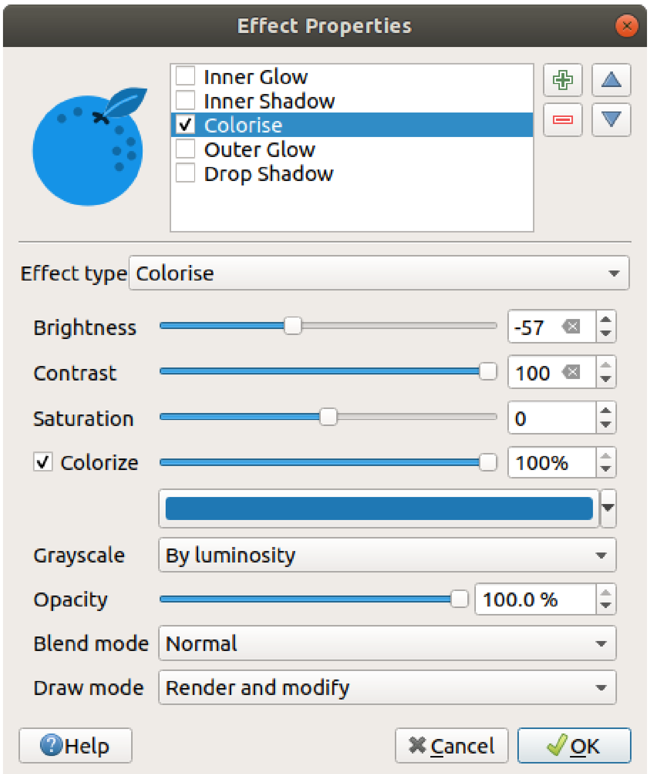


Abb. 14.20: Zeicheneffekte: Dialog einfärben

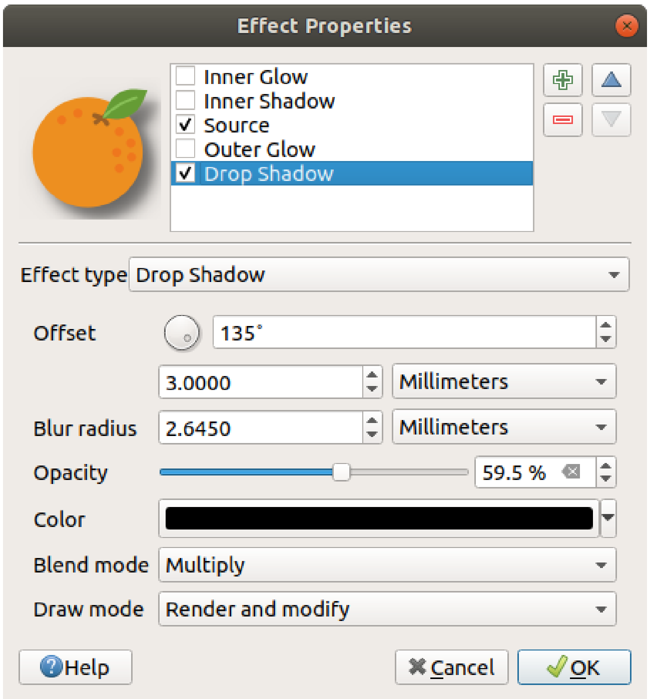


Abb. 14.21: Zeicheneffekte: Dialog Schattenwurf

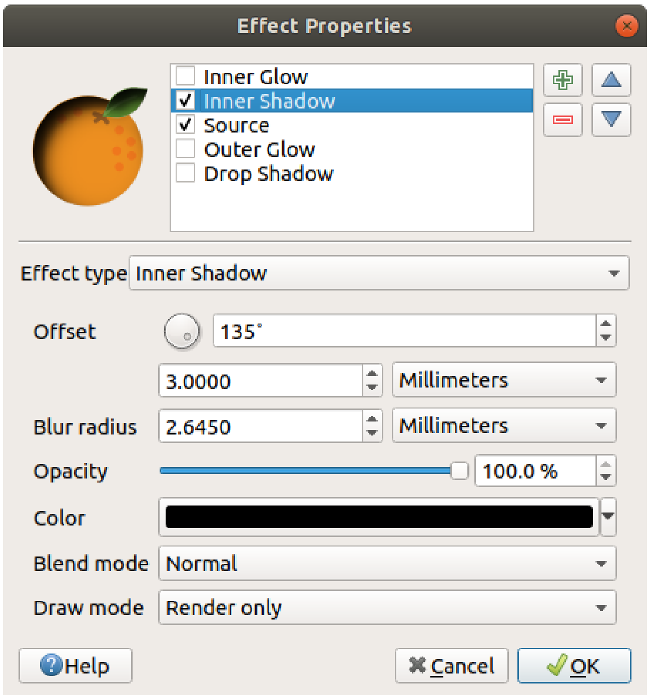


Abb. 14.22: Zeicheneffekte: Dialog Innerer Schatten

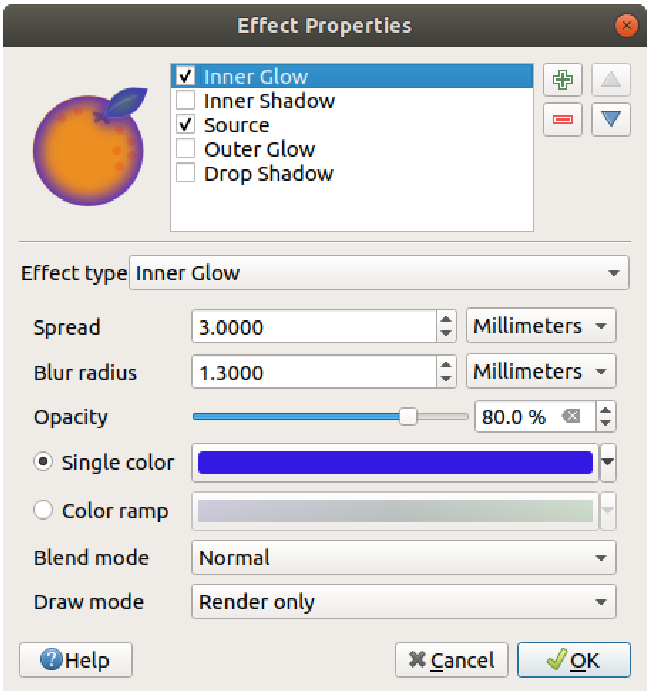


Abb. 14.23: Zeicheneffekte: Dialog Inneres Glühen

- **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.
- **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.

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 Move up and Move down 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).

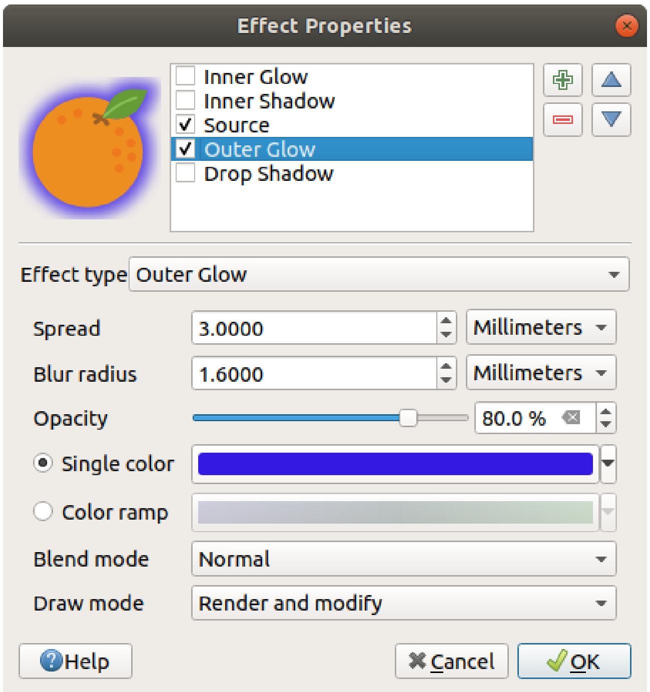


Abb. 14.24: Zeicheneffekte: Dialog Äußeres Glühen

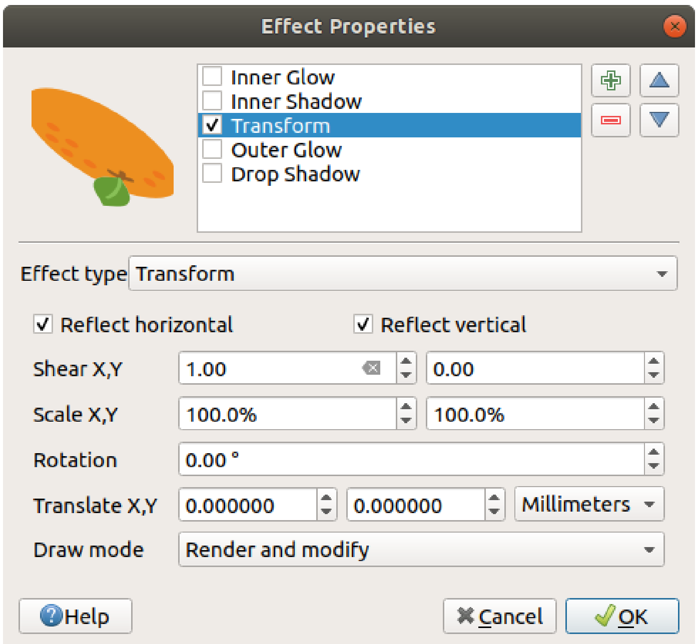


Abb. 14.25: Zeicheneffekte: Dialog transformieren

14.1.4 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.

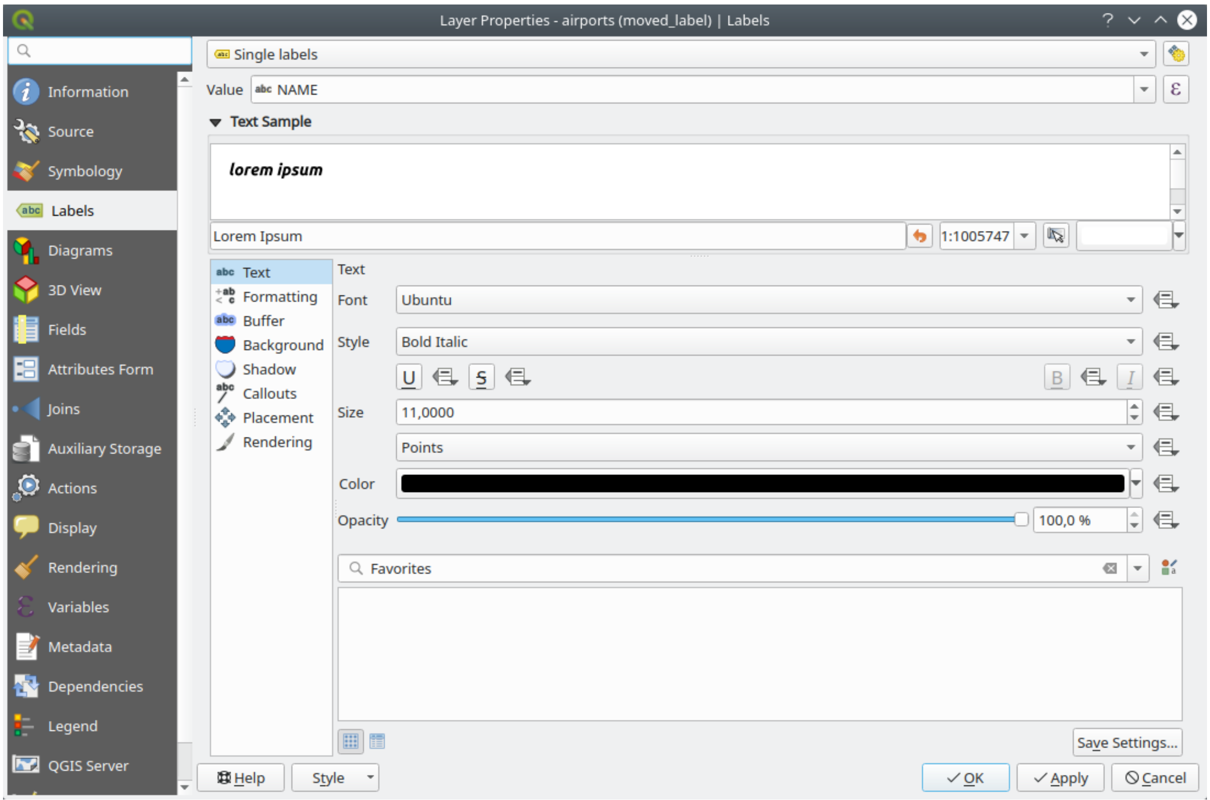


Abb. 14.26: Layer labeling settings - Single labels

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 ε if you want to define labels based on expressions - See *Ausdrucksbasierte Beschriftungen definieren*.

Below are displayed options to customize the labels, under various tabs:

- abc *Text*
- *Formatting*
- *Buffer*
- *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 global and 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:

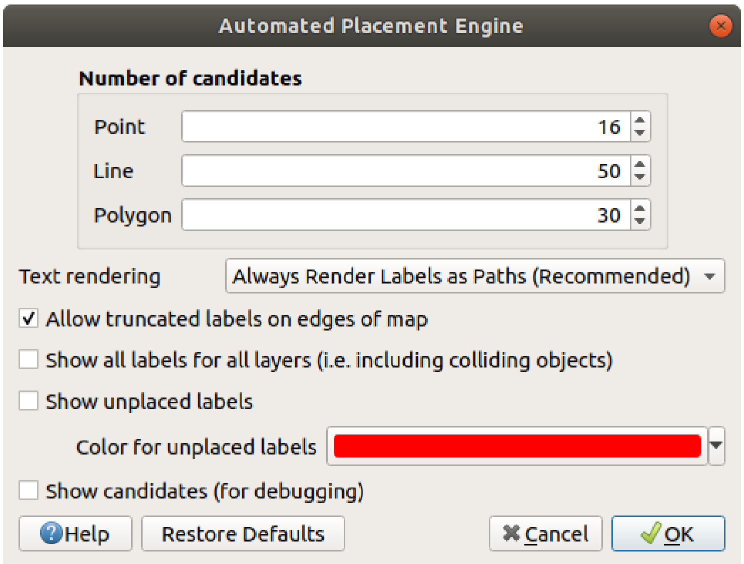


Abb. 14.27: The labels automated placement engine

- The *Number of candidates* controls how many label placement candidates should be generated for each feature type. The more candidates generated, the better the labeling will be - but at a cost of rendering speed. Smaller number of candidates results in less labels placed but faster redraws.
- *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*.

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, select the **Rule-based labeling** option in the main drop-down list from the *Labels* tab and click the button at the bottom of the dialog. Then fill the new dialog with a description and an expression to filter features. You can also set a *scale range* in which the label rule should be applied. The other options available in this dialog are the *common settings* seen beforehand.

A summary of existing rules is shown in the main dialog (see *figure_labels_rule_based*). 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.

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 *figure_labels_expression*, 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.

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:

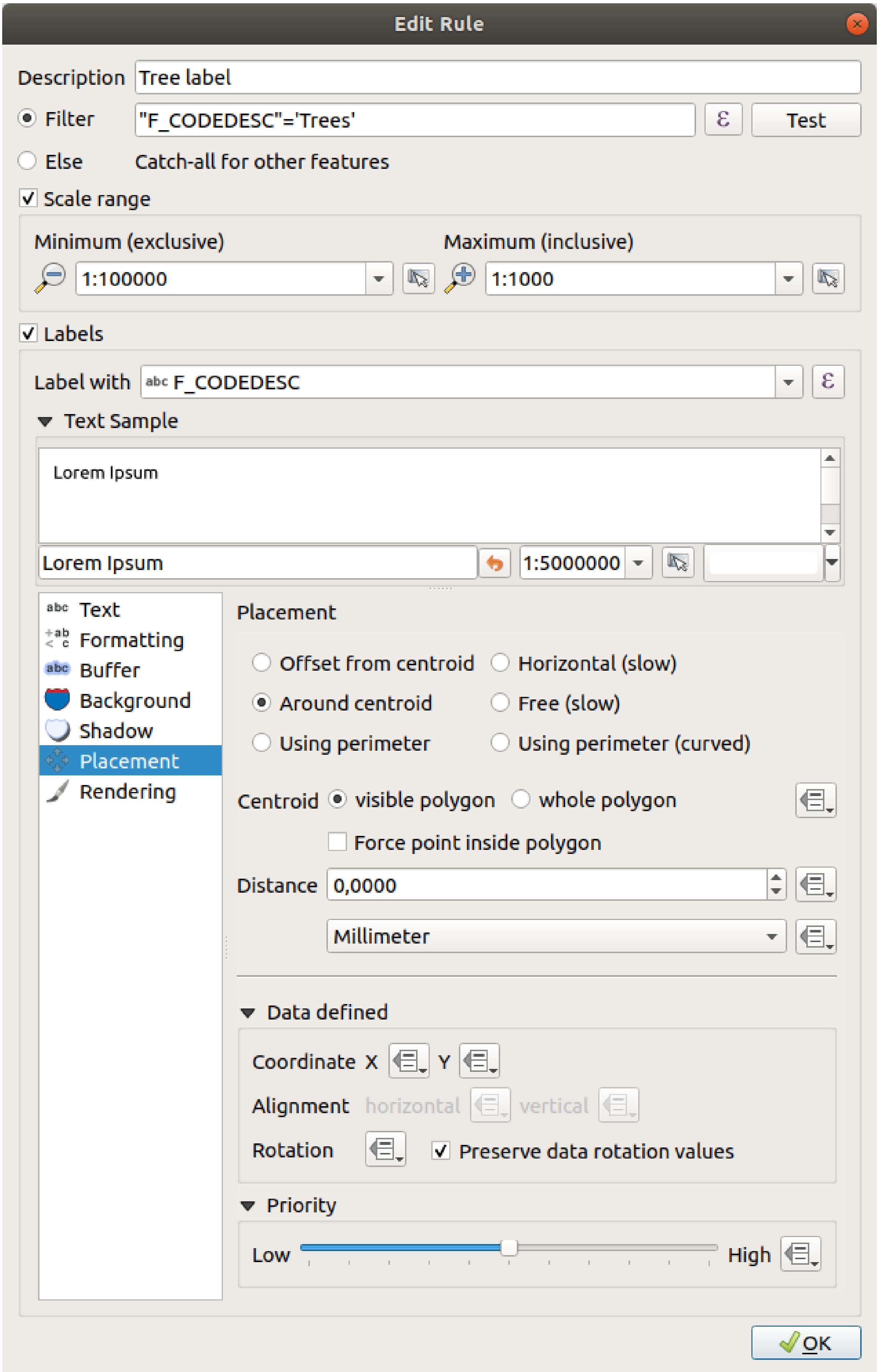


Abb. 14.28: Regeleigenschaften

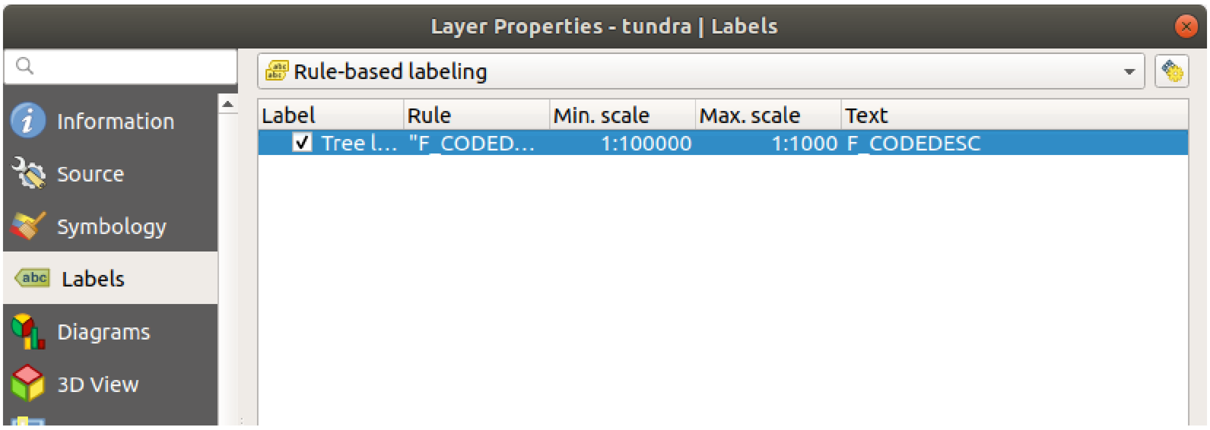


Abb. 14.29: Regelbasierte Beschriftung Bedienfelder

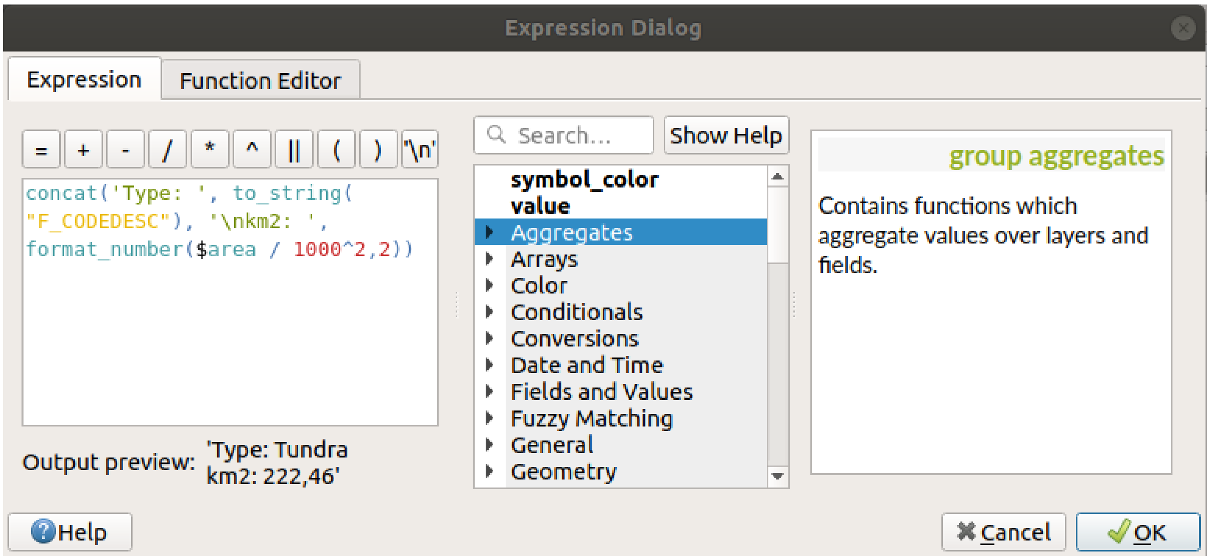


Abb. 14.30: Ausdrücke für das Beschriften verwenden

```
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 <= 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 zumBeschriften 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 militarian `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.

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* or *diagram* properties.

While for readability, `label` has been used below to describe the Label toolbar, note that when mentioned in their name, the tools work almost the same way with diagrams:

- Highlight Pinned Labels and Diagrams. If the vector layer of the label is editable, then the highlighting is green, otherwise it's blue.
- Toggles 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*).

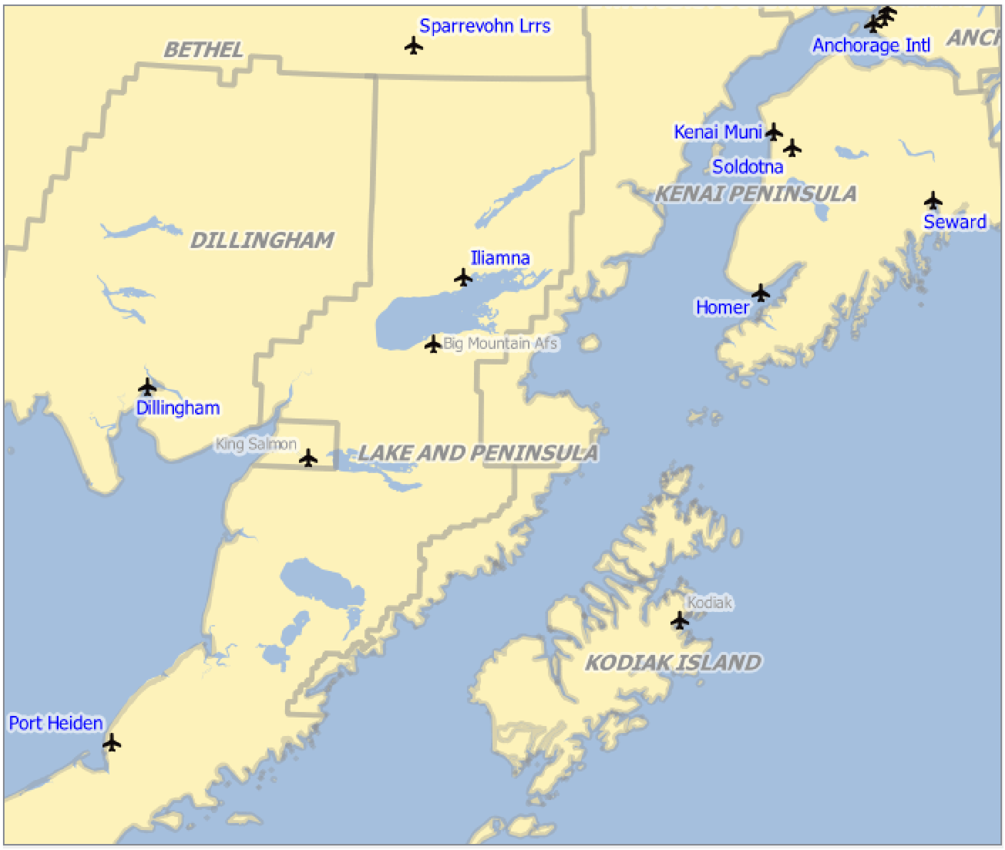


Abb. 14.31: Airports labels are formatted based on their attributes

Abb. 14.32: The Label toolbar

- Pin/Unpin Labels and Diagrams. By clicking or draging an area, you pin label(s). If you click or drag an area holding `Shift`, label(s) are unpinned. Finally, you can also click or drag an area holding `Ctrl` to toggle the pin status of label(s).
- Show/Hide Labels and Diagrams. If you click on the labels, or click and drag an area holding `Shift`, they are hidden. When a label is hidden, you just have to click on the feature to restore its visibility. If you drag an area, all the labels in the area will be restored.
- Moves a Label or Diagram. You just have to drag the label to the desired place.
- Rotates a Label. Click the label and move around and you get the text rotated.
- 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 function (see *figure_labels_coordinate_data_defined*).

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.

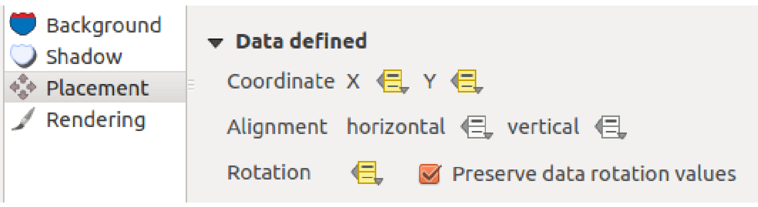


Abb. 14.33: Das Beschriften von Polygonlayern mit datendefinierter Übersteuerung

4. Zoomen Sie auf einen See.
5. Set editable 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 *figure_labels_move*). The new position of the label is saved in the `xlabel` and `ylabel` columns of the attribute table.

7. Using *The Geometry Generator* with the expression below, you can also add a linestring symbol layer to connect each lake to its moved label:

```
make_line( centroid( $geometry ), make_point( "xlabel", "ylabel" ) )
```

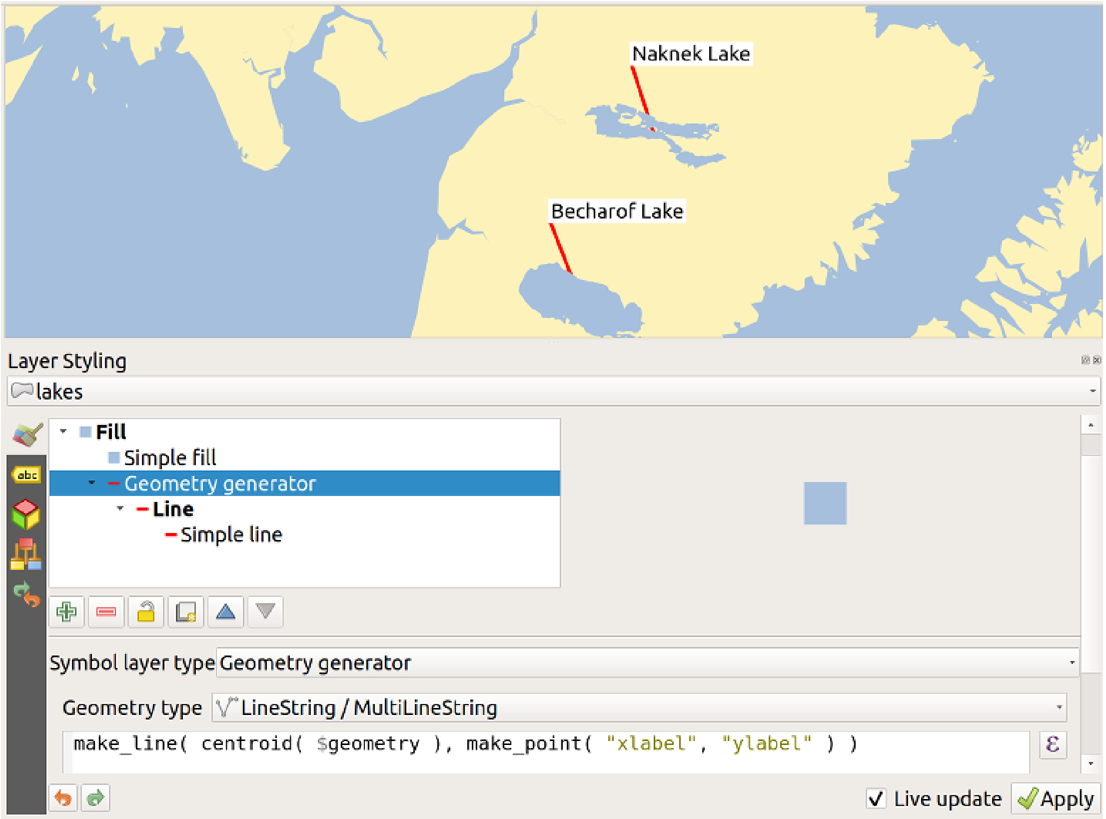


Abb. 14.34: Moved labels

Bemerkung: The *Auxiliary Storage Properties* mechanism may be used with data-defined properties without having an editable data source.

14.1.5 Diagrams Properties

The *Diagrams* tab allows you to add a graphic overlay to a vector layer (see *figure_diagrams_attributes*).

Die aktuelle Kernimplementation von Diagrammen bietet Unterstützung von:

- *No diagrams*: the default value with no diagram displayed over the features;
- *Pie charts*, a circular statistical graphic divided into slices to illustrate numerical proportion. The arc length of each slice is proportional to the quantity it represents;
- abc *Text diagrams*, a horizontaly divided circle showing statistics values inside;
- and *Histograms*.

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.

Tipp: 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.

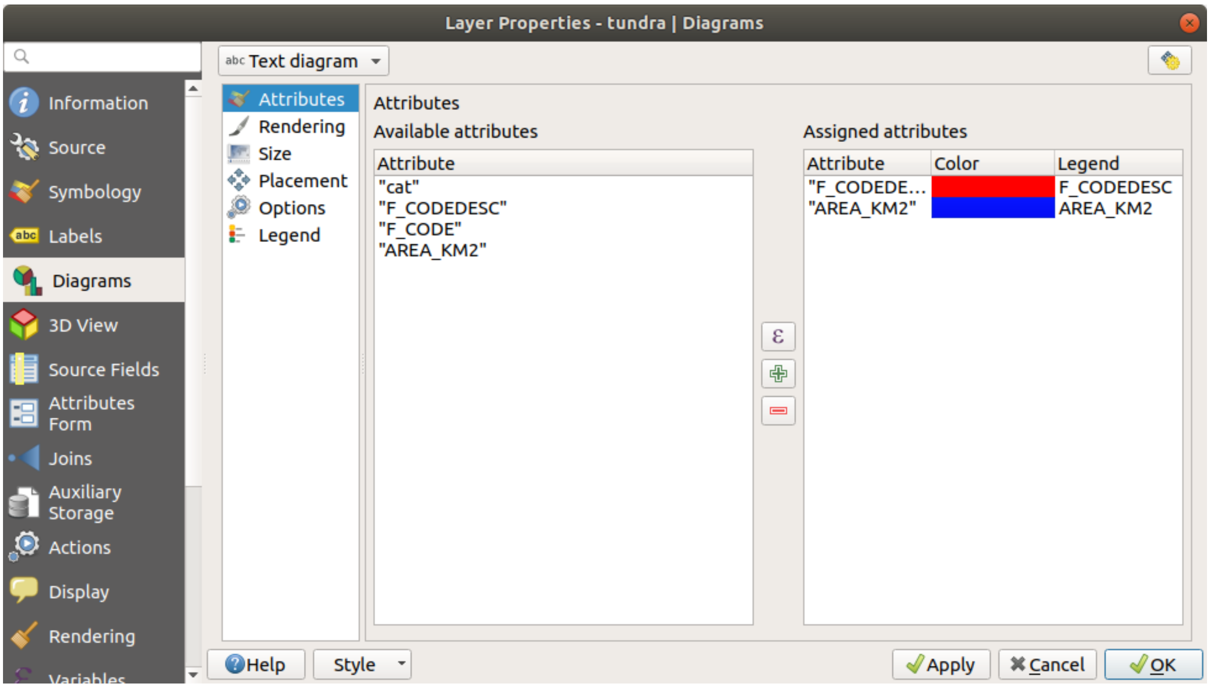


Abb. 14.35: Diagram properties - Attributes tab

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;
- and, depending on the type of diagram:
  - the width of the bar in case of histogram;
  - the circle background color in case of text diagram, and the font used for texts;
  - die Ausrichtung der linken Linie von der ersten Scheibe im Tortendiagramm. Beachten Sie, die Scheiben werden im Uhrzeigersinn angezeigt.

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*;
- *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.

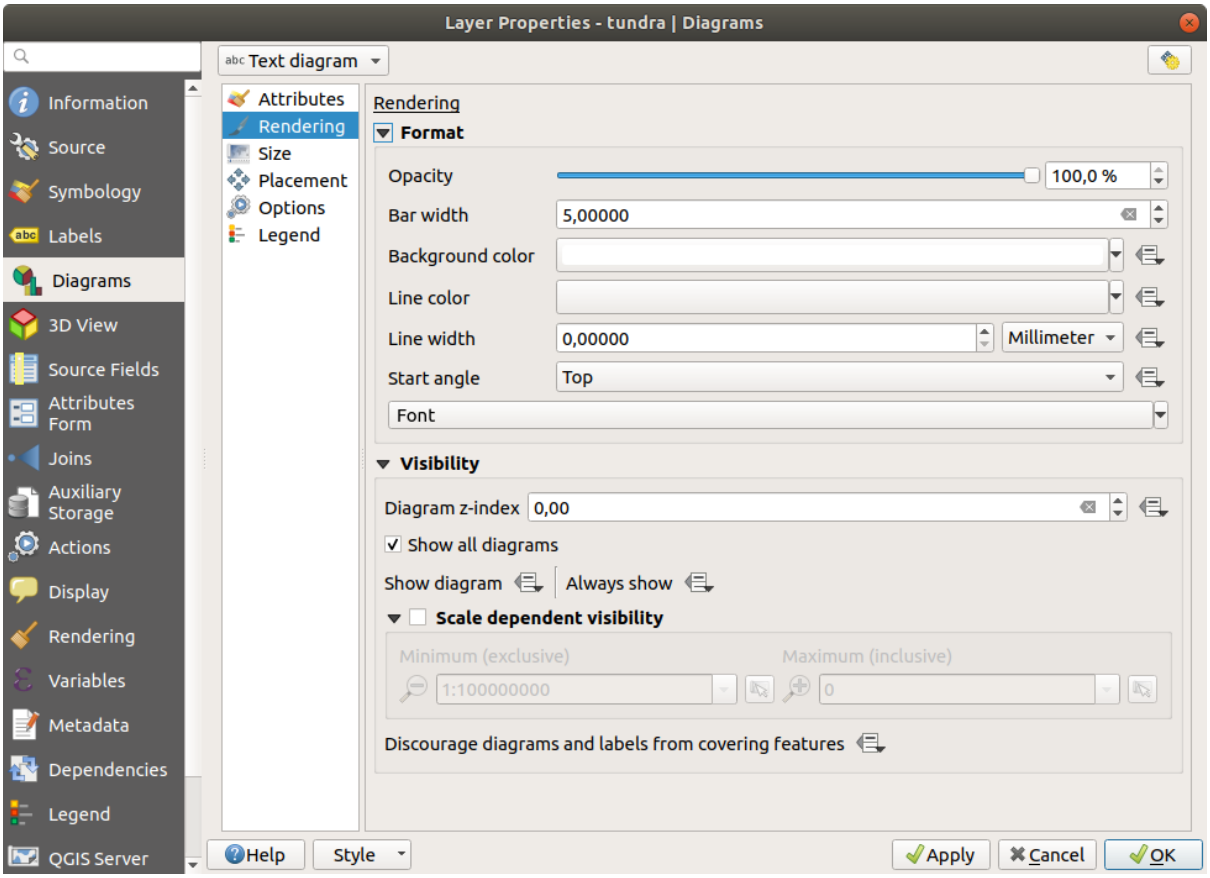


Abb. 14.36: Diagram properties - Rendering tab

Größe

Size is the main tab to set how the selected statistics are represented. The diagram size units can be ‚Millimeter', ‚Points', Pixels, ‚Map Units' or ‚Inches'. You can use :

- *Feste Größe*, eine feste Größe zum Darstellen der Grafik aller Objekte, außer beim Darstellen des Histogramms
- oder *Skalierte Größe*, basiert auf einem Ausdruck, der Layerattribute benutzt.

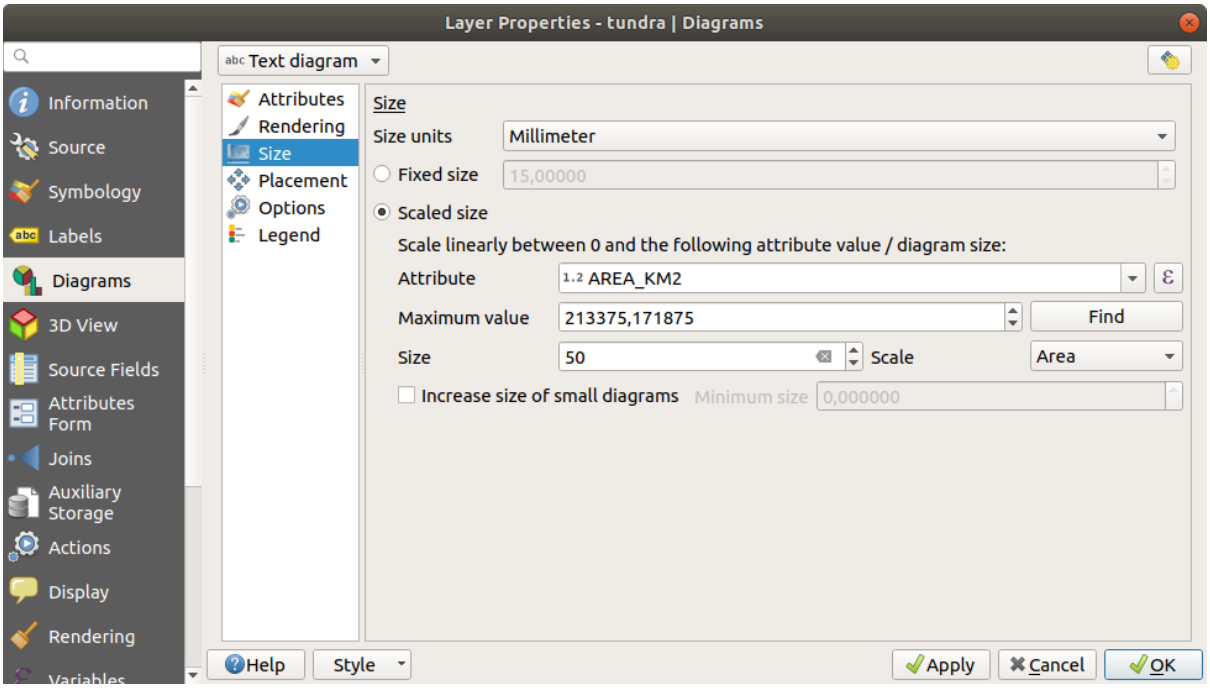


Abb. 14.37: Diagram properties - Size tab

Platzierung

Platzierung hilft die Diagramm Position einzustellen. Entsprechend dem Layergeometrietyp, bietet es verschiedene Optionen zur Platzierung:

- ‚Over the point' or ‚Around the point' for point geometry. The latter variable requires a radius to follow.
- ‚Over the line' or ‚Around the line' for line geometry. Like point feature, the last variable requires a distance to respect and user 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 here you can also use the line orientation for the position of the diagram.
- ‚Über dem Zentrum', ‚Um das Zentrum' (mit eingestellter Distanz), ‚Umfang' und irgendwo ‚Innerhalb des Polygons' sind die Optionen für Polygonobjekte.

The diagram can also be placed using feature data to fill the coordinates `X` and `Y` fields.

The placement of the diagrams can interact with the labeling, so you can detect and solve position conflicts between diagrams and labels by setting the **Priority** slider value.

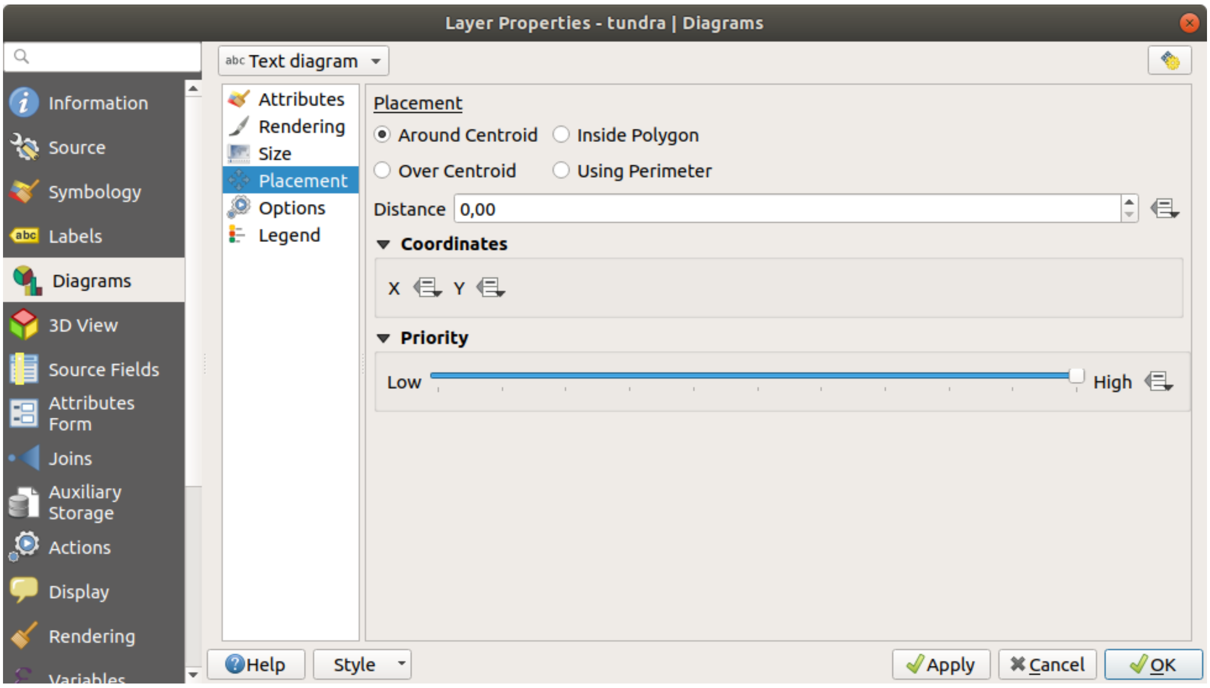


Abb. 14.38: Vector properties dialog with diagram properties, Placement tab

Optionen

Der Reiter *Optionen* hat nur Einstellungen im Fall eines Histrogramms. Sie können wählen, ob die Orientierung der Leiste ‚oben', ‚unten', ‚rechts' und ‚links' ist.

Legend

From the *Legend* tab, you can choose to display items of the diagram in the *Layerfenster*, 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.

Fallstudien

Wir werden ein Beispiel zeigen und dem Alaskagrenzlayer ein Textdiagramm das Temperaturdaten von einem climate Vektorlayer zeigt überlagern. Beide Vektorlayer sind Teil des QGIS Beispieldatensatzes (siehe Abschnitt *Beispieldaten herunterladen*).

1. Klicken Sie erst auf das Vektorlayer hinzufügen Icon, browsen Sie zum QGIS Beispieldatensatzordner und laden Sie die beiden Vektorlayer `alaska.shp` und `climate.shp`.
2. Doppelklicken Sie auf den `climate` Layer in der Kartenlegende um den Dialog *Layereigenschaften* zu öffnen.
3. Click on the *Diagrams* tab and from the *Diagram type* combo box, select ‚Text diagram'.

4. Im *Darstellung* Reiter wählen wir ein Hellblau als Hintergrundfarbe und im Reiter *Größe* stellen wir eine feste Größe von 18 mm ein.
5. Im Reiter *Position* könnte die Platzierung auf ‚Um Punkt' eingestellt werden.
6. Im Diagramm wollen wir die Werte der drei Spalten `T_F_JAN`, `T_F_JUL` und `T_F_MEAN` darstellen. Wählen Sie erst `T_F_JAN` als *Attribute´ und klicken Sie den |signPlus| Knopf, dann ``T_F_JUL`* und schließlich `T_F_MEAN`.
7. Now click *Apply* to display the diagram in the QGIS main window.
8. Sie können die Diagrammgröße im *Größe* Reiter anpassen. Aktivieren Sie *Skalierte Größe* und stellen Sie die Größe des Diagramms auf Basis eines Attributes mit dem *Maximalwert* und dem *Größe* Menü ein. Wenn die Diagramme auf dem Bildschirm zu klein erscheinen können Sie das *Kleine Diagramme vergrößern* Kontrollkästchen aktivieren und die Minimalgröße des Diagramms definieren.
9. Change the attribute colors by double clicking on the color values in the *Assigned attributes* field. *Figure_diagrams_mapped* gives an idea of the result.
10. Finally, click *OK*.

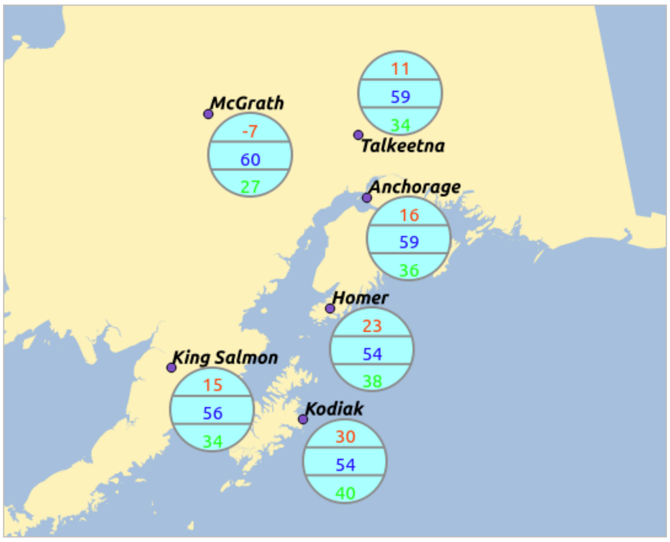


Abb. 14.39: Diagramm aus Temperaturdaten auf einer Karte dargestellt

Behalten Sie im Hinterkopf dass im Reiter *Position* eine *Datendefinierte Position* der Diagramme möglich ist. Sie können hier Attribute verwenden um die Position des Diagramms zu definieren. Sie können auch eine maßstabsabhängige Sichtbarkeit im *Darstellung* Reiter einstellen.

The size and the attributes can also be an expression. Use the button to add an expression. See *Ausdrücke* chapter for more information and example.

Using data-defined override

As mentioned above, you can use some custom data-defined to tune the diagrams rendering:

- position in *Placement* tab by filling `X` and `Y` fields
- visibility in *Appearance* tab by filling the `Visibility` field

See *Datendefinierte Übersteuerung für das Beschriften* for more information.

14.1.6 3D View Properties

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 symbol whose properties can be *data-defined* or not
- *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*.

Depending on the layer geometry type, various properties are available for 3D rendering.

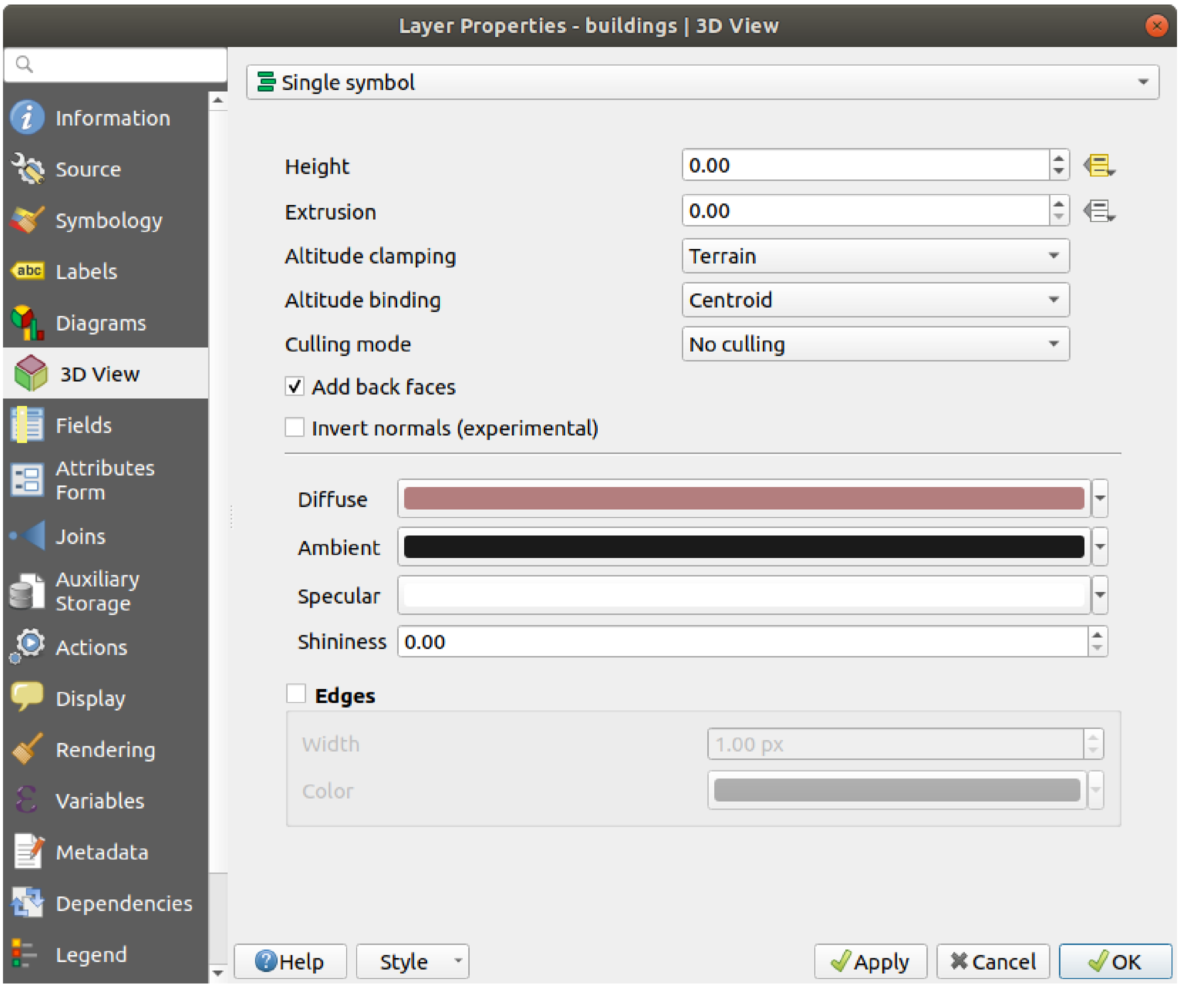


Abb. 14.40: 3D properties of a polygon layer

Point Layers

- You can define different simple 3D shapes like *Sphere*, *Cylinder*, *Cube*, *Cone*, *Plane* and *Torus* defined by their *Radius*, *Size* or *Length*. The unit of size of the 3D shapes refers to the CRS of the project.
- The shading of the 3D shapes can be defined by the menus *Diffuse*, *Ambient*, *Specular* and *Shininess* (see https://en.wikipedia.org/wiki/Phong_reflection_model#Description)
- If you choose *3D Model*, the location will be determined by a simple point coordinate.
- For visualizing 3D point clouds you can use *Billboard* Shapes defined by the *Billboard Height*, *Billboard symbol* and *Altitude clamping*. The symbol will have a stable size.

- *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.
- *Translation* can be used to move objects in x, y and z axis.
- You can define a *Scale factor* for the 3D shape as well as a *Rotation* around the x-, y- and z-axis.

Line layers

- 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 can be defined in the menus *Diffuse*, *Ambient*, *Specular* and *Shininess*.

Polygon Layers

- As for the other ones, *Height* can be defined in CRS units.
- Again, *Extrusion* is possible for missing z-values.
- The *Altitude clamping*, *Altitude binding* can be defined as explained above.
- There is an additional option to *Add back faces* and *Invert normals*.
- You can define *Edges* by *Width* and *Color*.

Application example

To go through the settings explained above you can have a look at https://public.cloudmergin.com/projects/saber/luxembourg/tree.

14.1.7 Fields Properties

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.

You can also rename fields by double-clicking its name. This is only supported for data providers like PostgreSQL, Oracle, Memory layer and some OGR layer depending on the OGR data format and 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.

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.

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`. When serving the layer as `WMS` or `WFS`, you can also check here which fields could be retrieved.

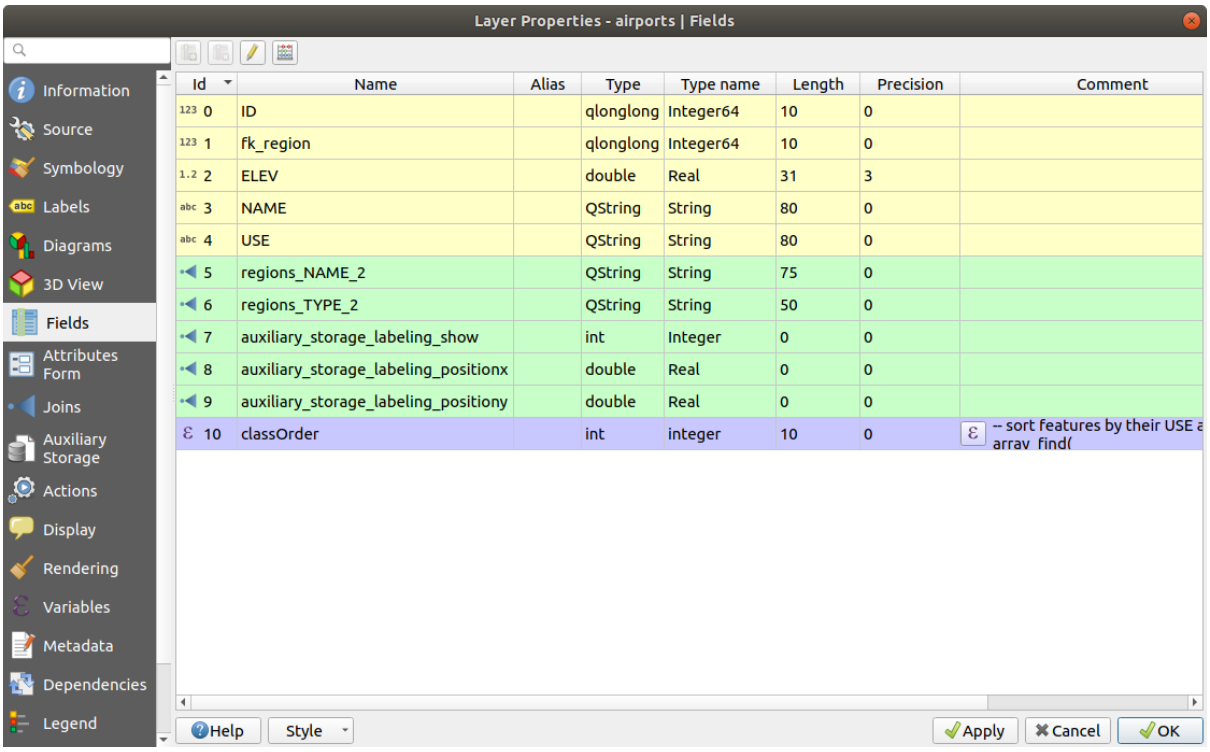


Abb. 14.41: Fields properties tab

14.1.8 Attributes Form Properties

The *Attributes Form* tab helps you set up the form to display when creating new features or querying existing one. You can define:

- the look and the behavior of each field in the feature form or the attribute table (label, widget, constraints…);
- the form's structure (custom or autogenerated):
- extra logic in Python to handle interaction with the form or field widgets.

At the top right of the dialog, you can set whether the form is opened by default when creating new features. This can be configured per layer or globally with the *Suppress attribute form pop-up after feature creation* option in the *Settings > Options > Digitizing* menu.

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.

The 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, as shown for example in *figure_fields_form*.

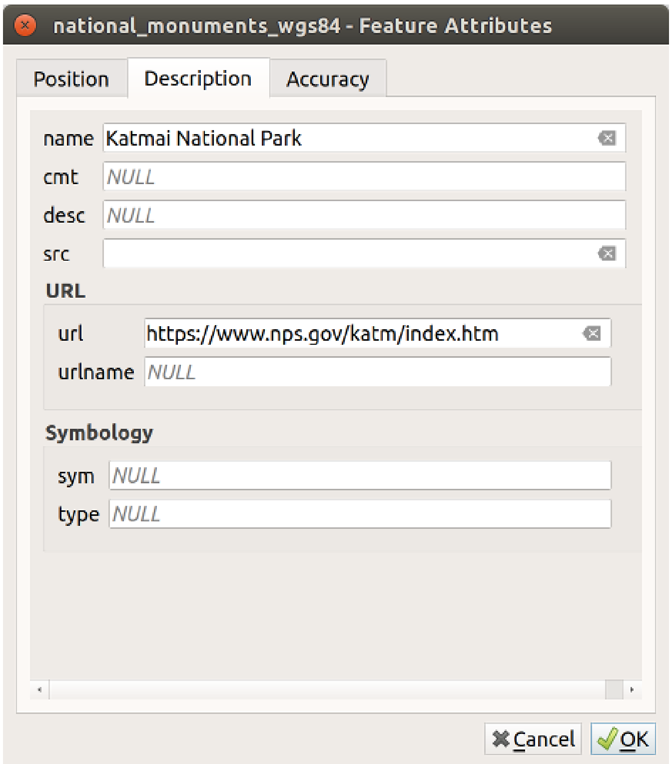


Abb. 14.42: Das Ergebnis eines integrierten Formulars mit Reitern und benannten Gruppen

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* (that you can customize) 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. Drag and drop fields from the other panel to re-add them. The same field can be added multiple times.
3. Drag and drop fields within the *Form Layout* panel to reorder their position.
4. Add containers (tab or group frames) to associate fields that belong to the same category and better structure the form.

1. The first step is to use the icon to create a tab in which fields and groups will be displayed
2. Then set the properties of the container, ie:
   - the name
   - the type, ie a *tab* or a *group in container* (a group inside a tab or another group)
   - and the *number of columns* the embedded fields should be distributed over

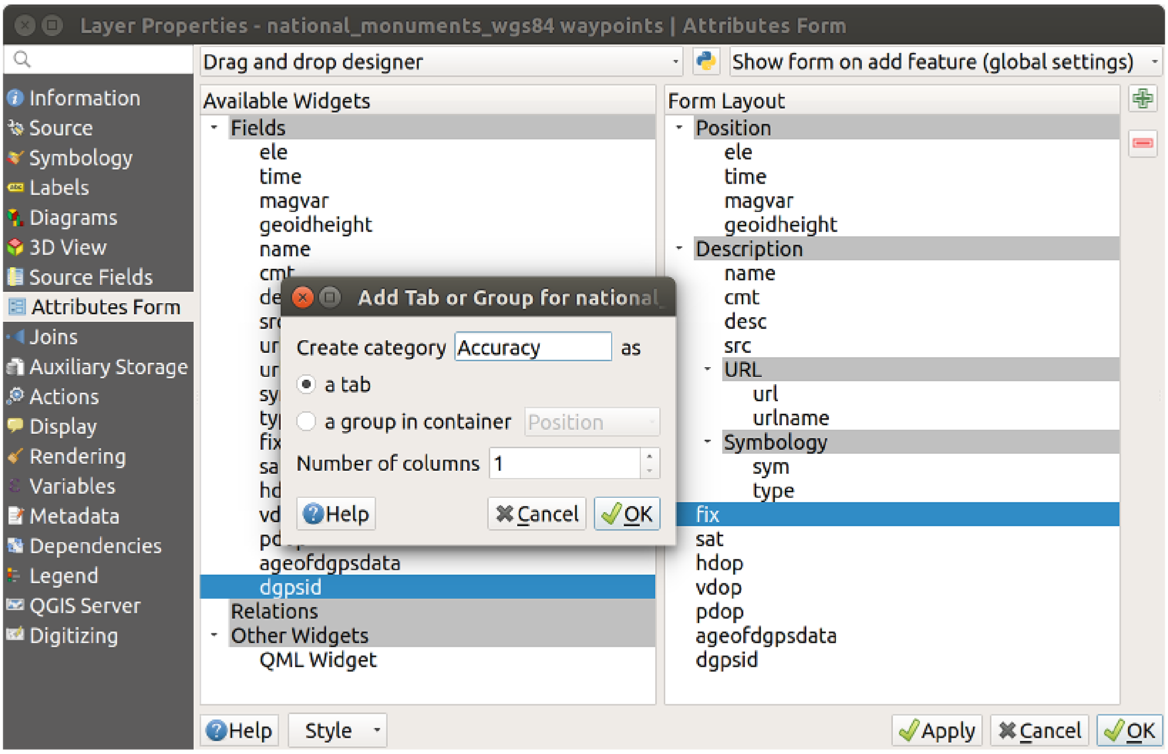


Abb. 14.43: Dialog to create containers with the **Attribute editor layout**

These, and other properties can later be updated by selecting the item and, from the third panel:

- hide or show the container's label
- display the container as a group box (only available for tabs).
- 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
- add a background color

3. You can create as many containers as you want; press the icon again to create another tab or a group frame under an existing tab.

5. The next step is to assign the relevant fields to each container, by simple drag and drop. Groups and tabs can also be moved in the same way.
6. 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

7. Apply the layer's properties dialog
8. Open a feature attribute form (eg, using the Identify features tool) and it should display the new form.

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(QWidged,"My line edit")
```

Reference in Python Init Function like so: `open`

Configure the field behavior

The main part of the *Attributes Form* tab helps you set the type of widget used to fill or display values of the field, in the attribute table or the feature form: you can define how user interacts with each field and the values or range of values that are allowed to be added to each.

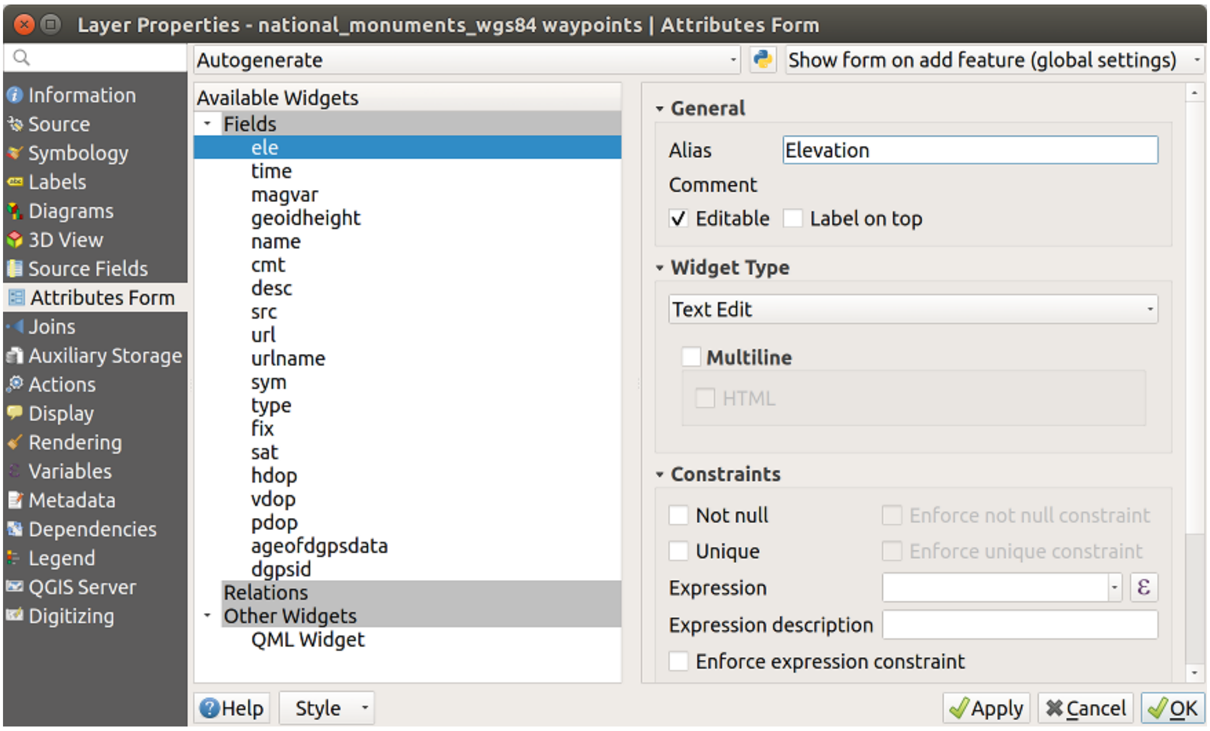


Abb. 14.44: Dialog um ein Bearbeitungselement für eine Attributspalte auszuwählen

Common settings

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

Show label: indicates whether the field name should be displayed in the form.

General options

- *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.
- *Label on top*: places the field name above or beside the widget in the feature form.

Default values

- *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…

Constraints

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. `regexp_match(col0,'A-Za-z')` to ensure that the value of the field *col0* has only alphabetical letter. 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 a yellow or red cross is displayed near the field. 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 and it does not prevent you to save the changes with the „wrong“ values;
  - A red 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.

Edit widgets

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:

- **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 könnne 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 also be used to display a hyperlink (to document path), a picture or a web page.
- **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.
- **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 Bearbeitungsele-ment kann entweder ein Schieber oder ein Drehfeld sein.
- **Relation Reference**: This widget lets you embed the feature form of the referenced layer on the feature form of the actual layer. See *Creating one or many to many relations*.
- **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.

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.

14.1.9 Joins Properties

The *Joins* tab allows you to associate features of the current layer (called `Target layer`) to features from another loaded vector layer (or table). The join is based on an attribute that is shared by the layers. The layers can be geometryless (tables) or not but their join attribute should be of the same type.

To create a join:

1. Click the Add new join button. The *Add vector join* dialog appears.
2. Select the *Join layer* you want to connect with the target vector layer
3. Specify the *Join field* and the *Target field* that are common to both the join layer and the target layer
4. Press *OK* and a summary of selected parameters is added to the *Join* panel.

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. QGIS provides more options to tweak the join:

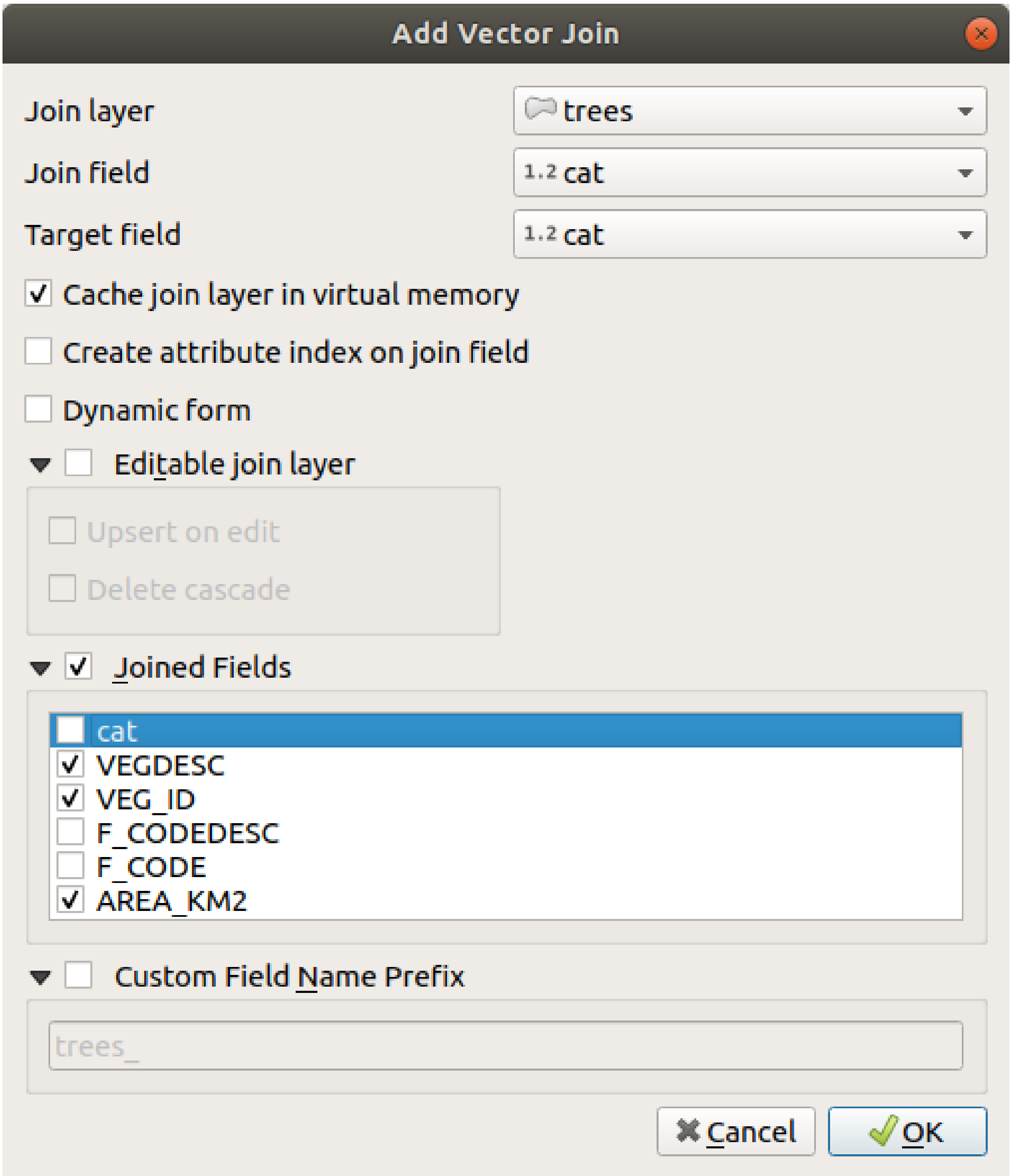


Abb. 14.45: Verknüpfe eine Attributtabelle zu einem vorhandenen Vektorlayer

- *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*
- *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

QGIS currently has support for joining non-spatial table formats supported by OGR (e.g., CSV, DBF and Excel), delimited text and the PostgreSQL provider.

14.1.10 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:

Labeling

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):

As soon as an auxiliary layer is configured for the current data source, you can retrieve its information in the tab:

The auxiliary layer now has these characteristics:

- the primary key is `ID`,
- there are `0` features using an auxiliary field,

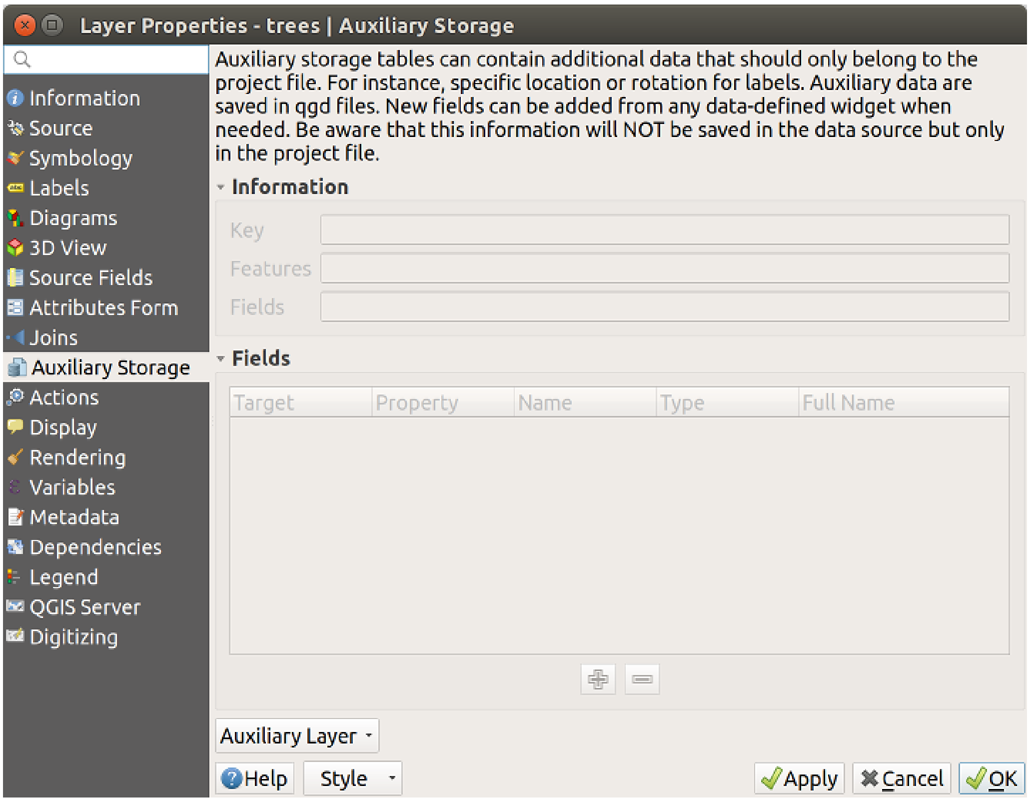


Abb. 14.46: Auxiliary Storage tab

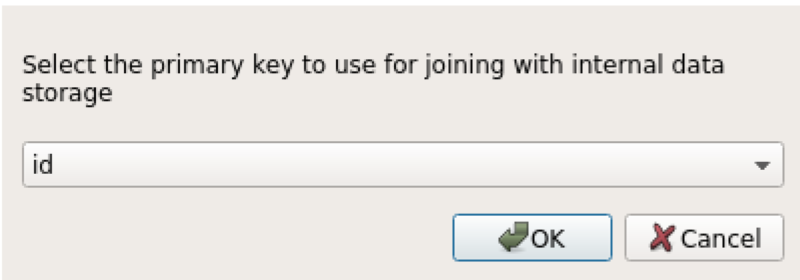


Abb. 14.47: Auxiliary Layer creation dialog

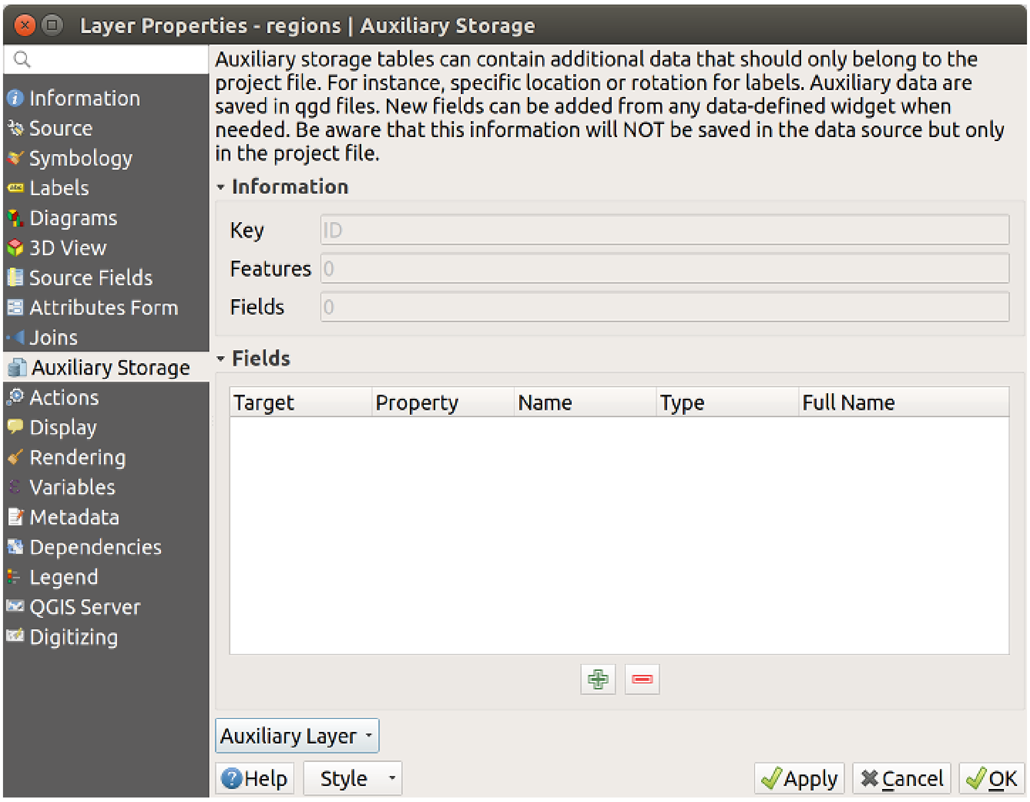


Abb. 14.48: Auxiliary Layer key

- 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:

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:

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. 14.47) 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:

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:

Once created, the auxiliary field can be retrieved in the auxiliary storage tab:

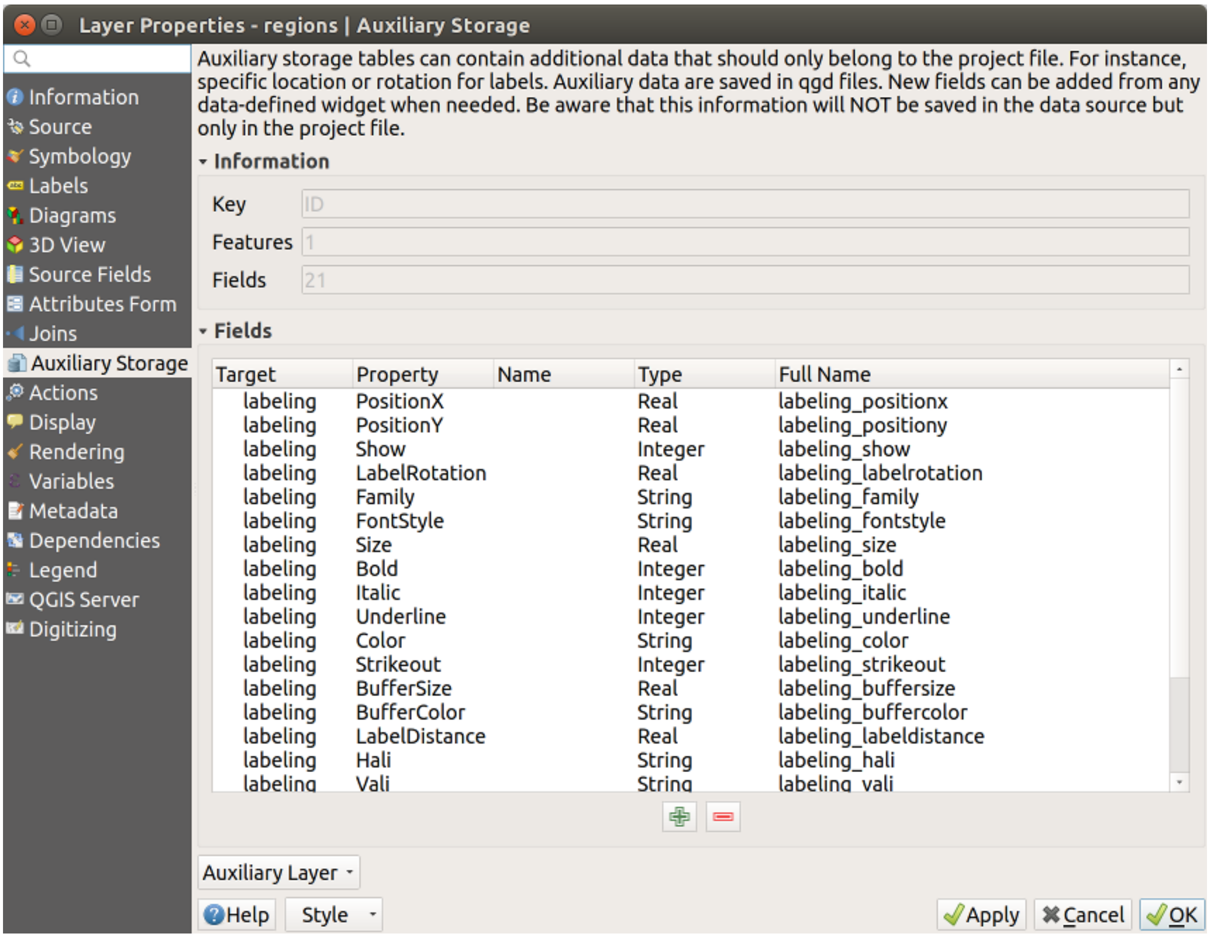


Abb. 14.49: Auxiliary Fields

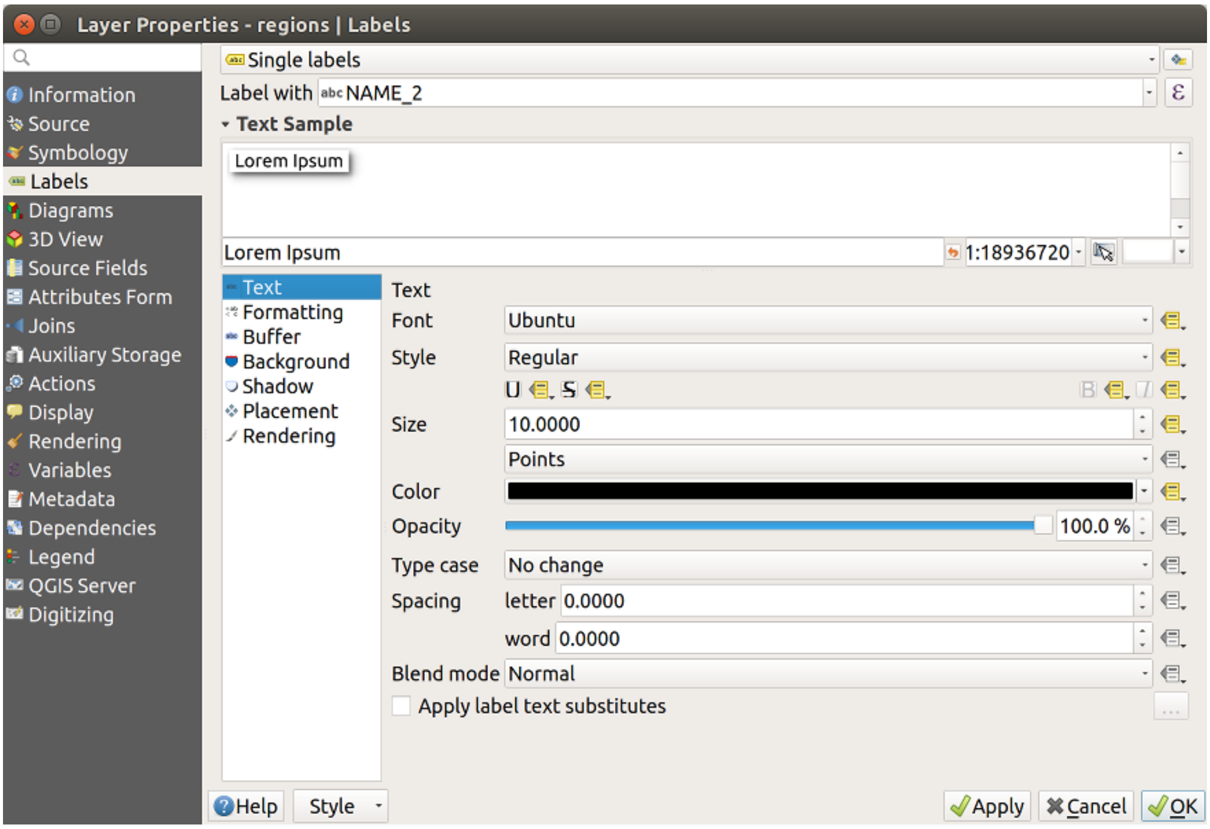


Abb. 14.50: Data-defined properties automatically created

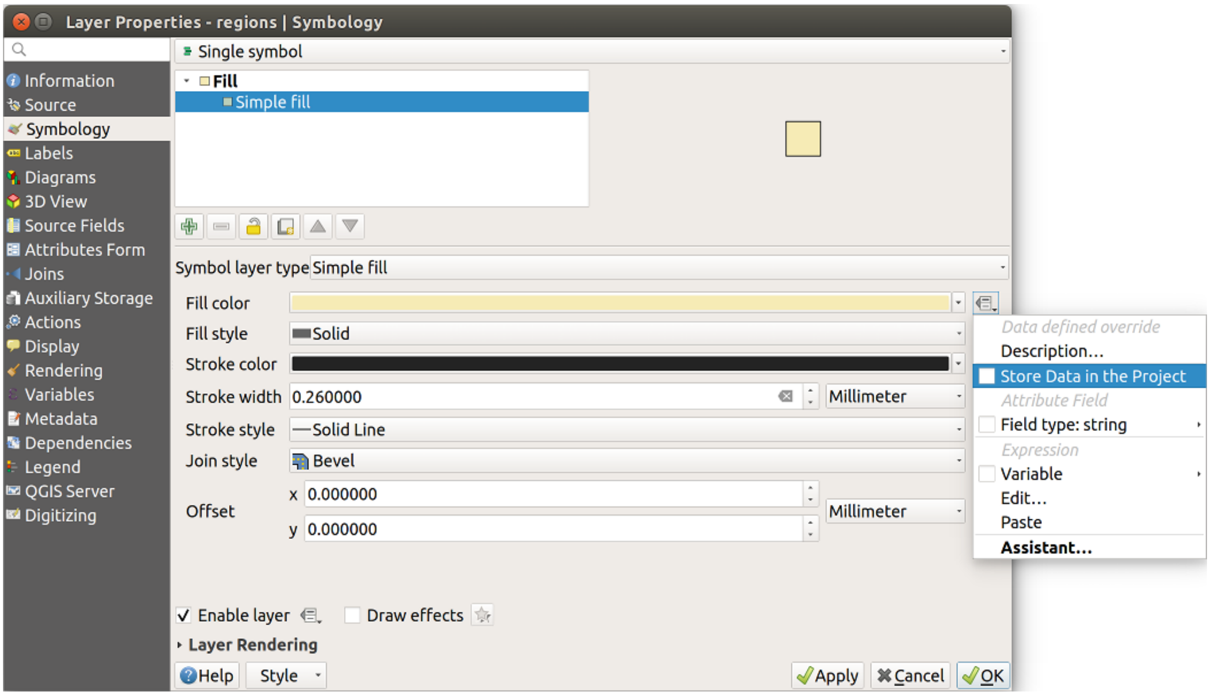


Abb. 14.51: Data-defined property menu for symbol

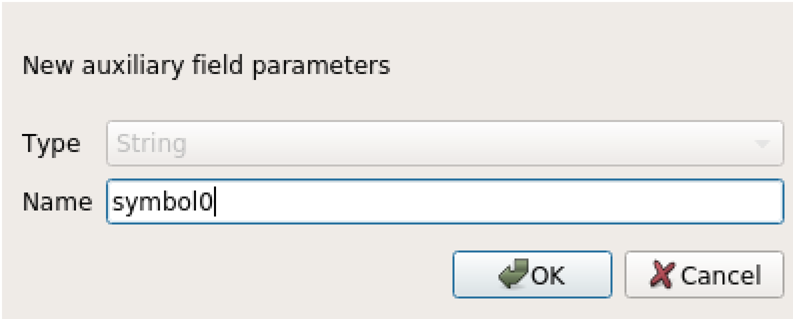


Abb. 14.52: Name of the auxiliary field for a symbol

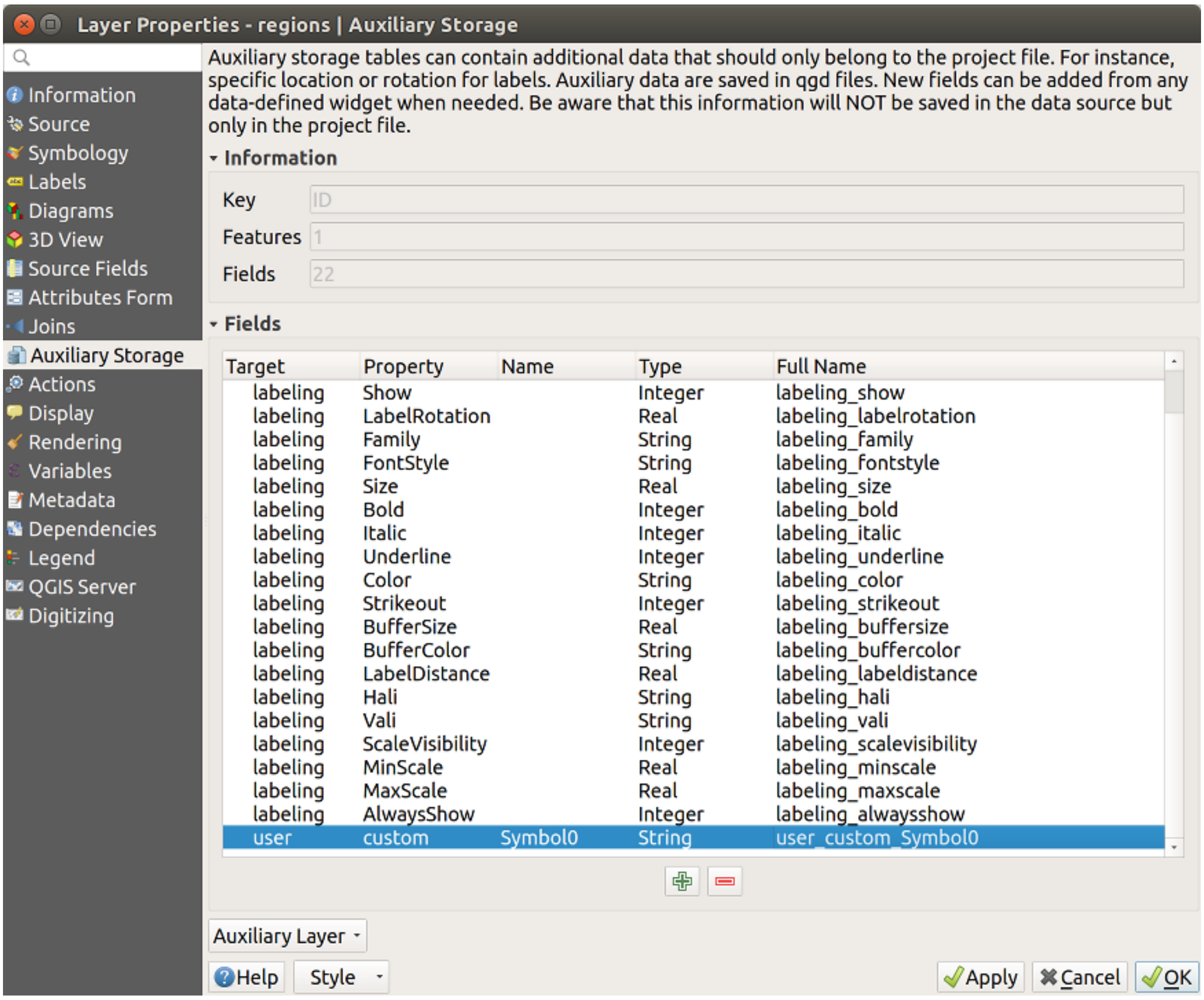


Abb. 14.53: Auxiliary field symbol

Attribute table and 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:

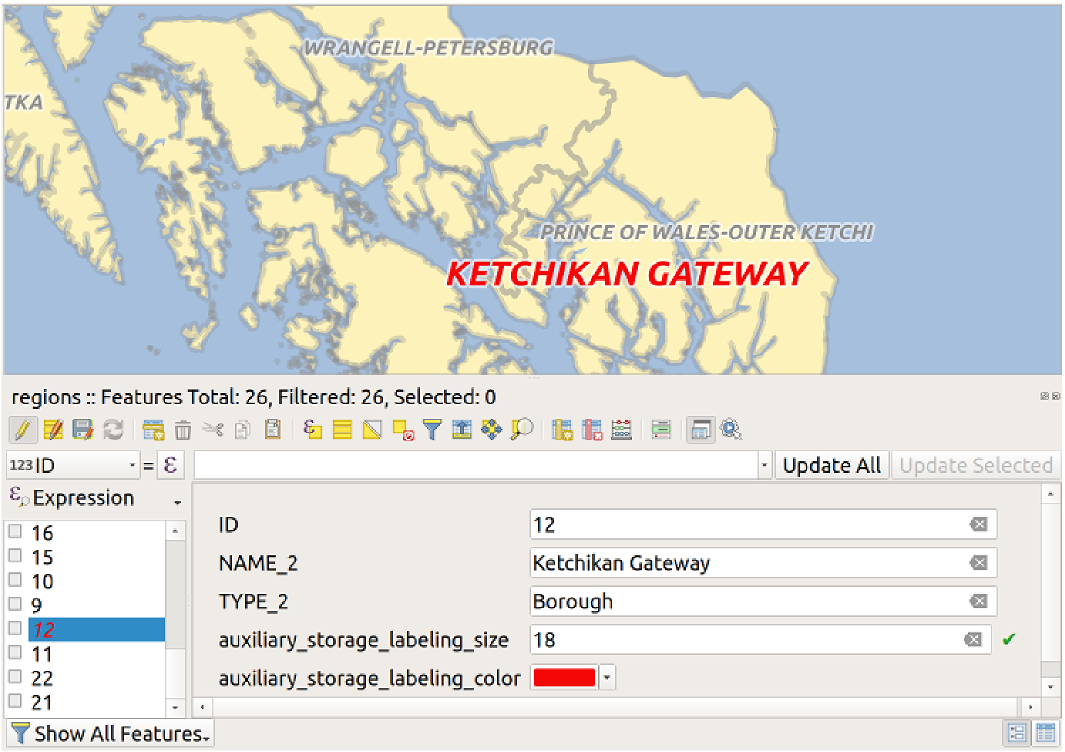


Abb. 14.54: Form with auxiliary fields

Management

The *Auxiliary Layer* menu allows you to manage the auxiliary fields:

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 *Labeling*, 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.

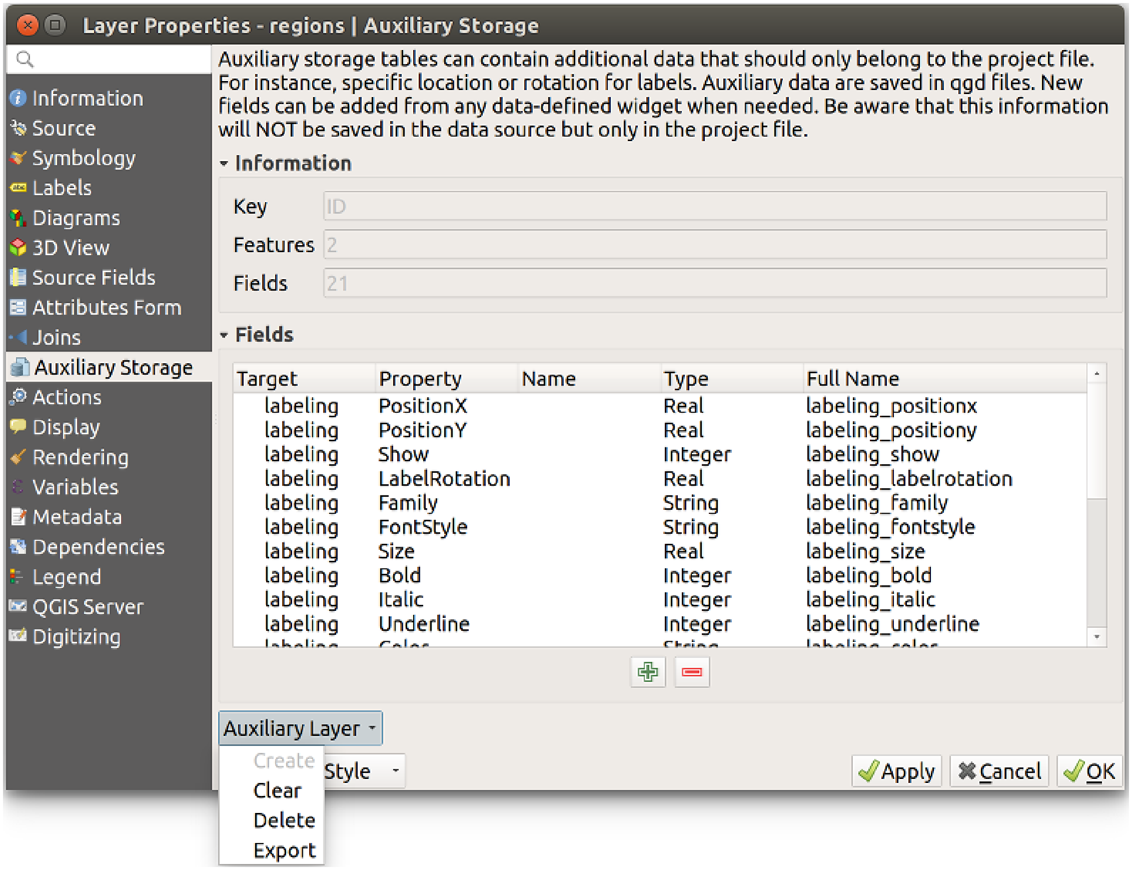


Abb. 14.55: Auxiliary layer management

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.

14.1.11 Actions Properties

QGIS bietet die Möglichkeit, Aktionen auf Basis von Attributen einer Ebene durchzuführen. Dies kann für eine Vielzahl von Aktionen genutzt werden, z.B. um ein Programm mit Abfragen aus der Attributdatenbank zu füttern oder um Parameter an ein Web-Reporting-Tool weiterzugeben.

Aktionen auf Basis von Attributen sind sinnvoll wenn sie häufig eine externe Anwendung starten oder eine Internetseite auf Basis von einem oder mehreren Werten in Ihrem Vektorlayer visualisieren wollen. Sie sind in 6 Typen aufgeteilt und können wie folgt verwendet werden:

- Allgemein, Mac, Windows und Unix Aktionen starten einen externen Prozess.
- Python Aktionen führen einen Python-Ausdruck aus.
- Allgemeine und Pythonaktionen sind überall sichtbar.
- Mac, Windows und Unix Aktionen sind nur sichtbar auf der entsprechenden Plattform (z.B. können Sie drei ‚Bearbeiten' Aktionen definieren um einen Editor zu öffnen und die Benutzer können nur die eine ‚Bearbeiten' Aktion für Ihr Betriebssystem sehen und ausführen um den Editor zu starten).

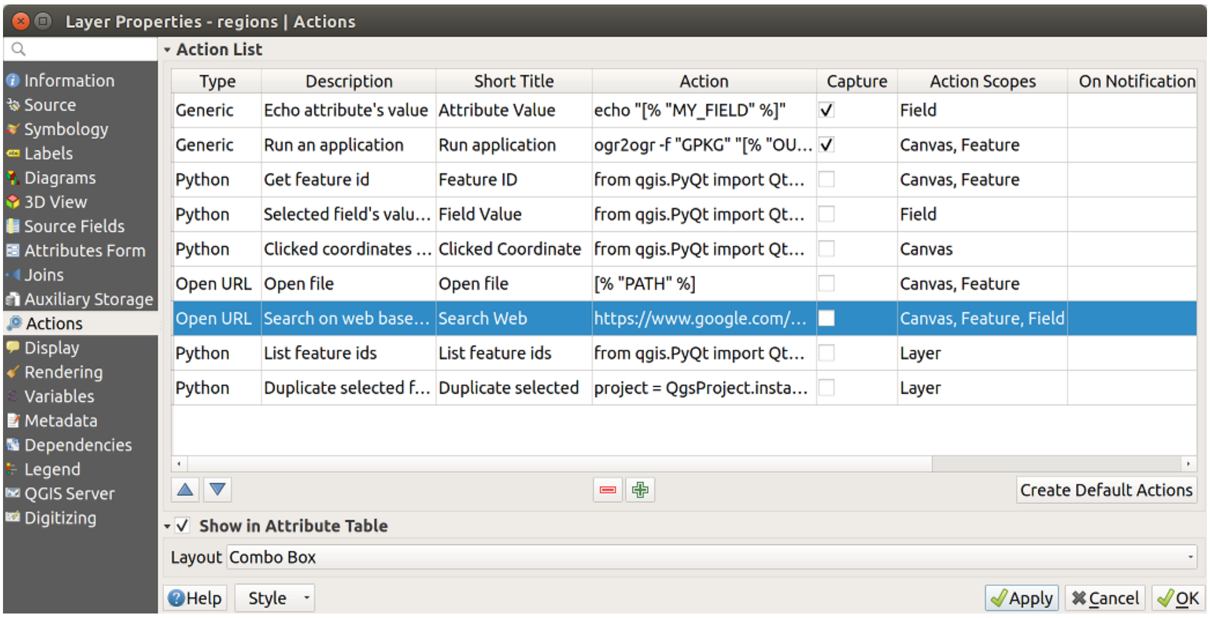


Abb. 14.56: Überblick über den Dialog Aktionen mit einigen Beispielaktionen

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 *using_actions* below). 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 4 different choices:

1. *Feature Scope*: action is available when right click in the cell within the attribute table.
2. *Field Scope*: 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.
3. *Layer Scope*: 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.
4. *Canvas*: action is available in the main action button in the toolbar.

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 de, `%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* 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 *Scope*. See *Aktionen definieren* for further information. Leave the default settings for this example.
9. Für diese Aktion ist es notwendig den Namen des externen Programms anzugeben. In diesem Fall können wir Firefox verwenden. Wenn das Programm sich nicht im Pfad befindet müssen Sie den vollständigen Pfad angeben.
10. Following the name of the external application, add the URL used for doing a Google search, up to but not including the search term: `https://www.google.com//search?q=`
11. The text in the *Action* field should now look like this: `https://www.google.com//search?q=`
12. 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.
13. From the drop-down box, select ‚NAMES' and click *Insert*.
14. Die Aktion sieht nun so aus:

    `https://www.google.com//search?q=[%NAMES%]`
15. To finalize and add the action, click the *OK* button.

Damit ist die Aktion fertig für den Einsatz. Der gesamte Befehl der Aktion sollte folgendermaßen aussehen:

```
https://www.google.com//search?q=[%NAMES%]
```

Damit ist die Aktion fertig für den Einsatz. Schließen Sie den *Eigenschaften* Dialog und zoomen Sie in einen Bereich Ihrer Wahl. Stellen Sie sicher, dass der Layer `lakes` in der Legende aktiviert ist. Nun identifizieren Sie einen See. In der Ergebnisanzeige sollte nun die Aktion sichtbar sein:

When we click on the action, it brings up Firefox 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:

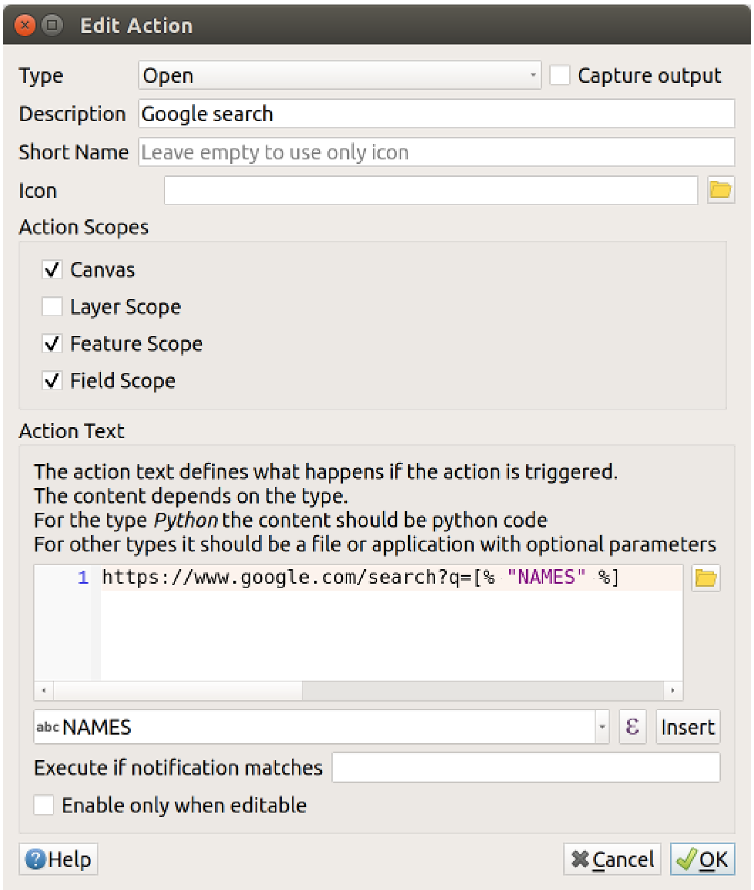


Abb. 14.57: Edit action dialog configured with the example

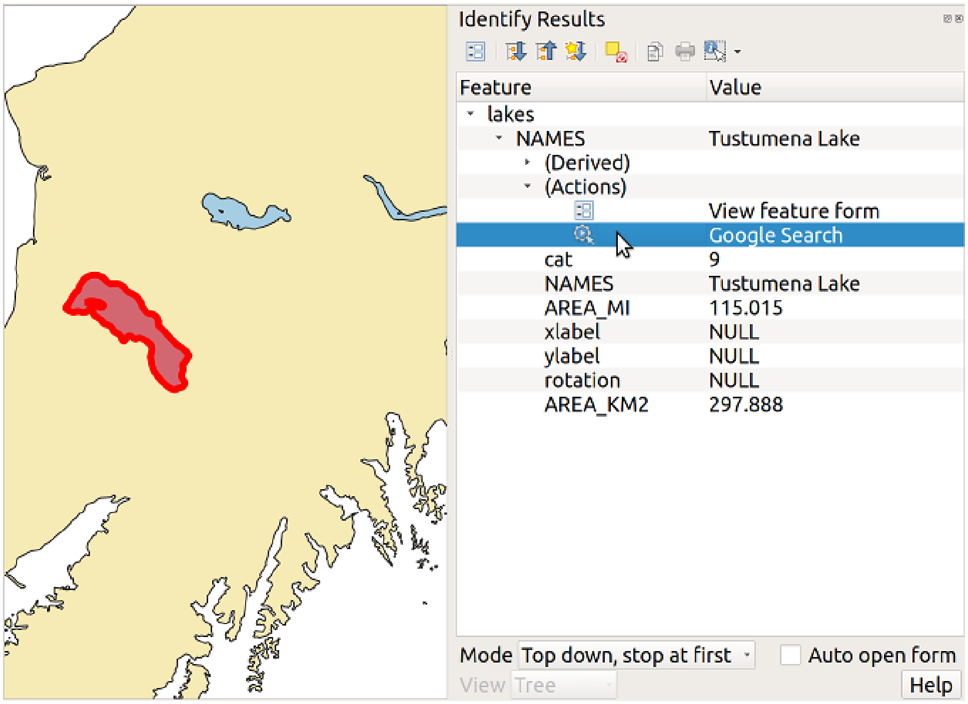


Abb. 14.58: Wählen Sie ein Objekt und eine Aktion aus

```
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_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" %]')
```

14.1.12 Display Properties

The *Display* tab helps you configure fields to use for feature identification:

- The *Display name*: based on a field or an *expression*. This is:
  - 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 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 no *HTML Map Tip* is set.
- The *HTML Map Tip* is specifically created for the map tips: it's a more complex and full HTML text mixing fields, expressions and html tags (multiline, fonts, images, hyperlink…).

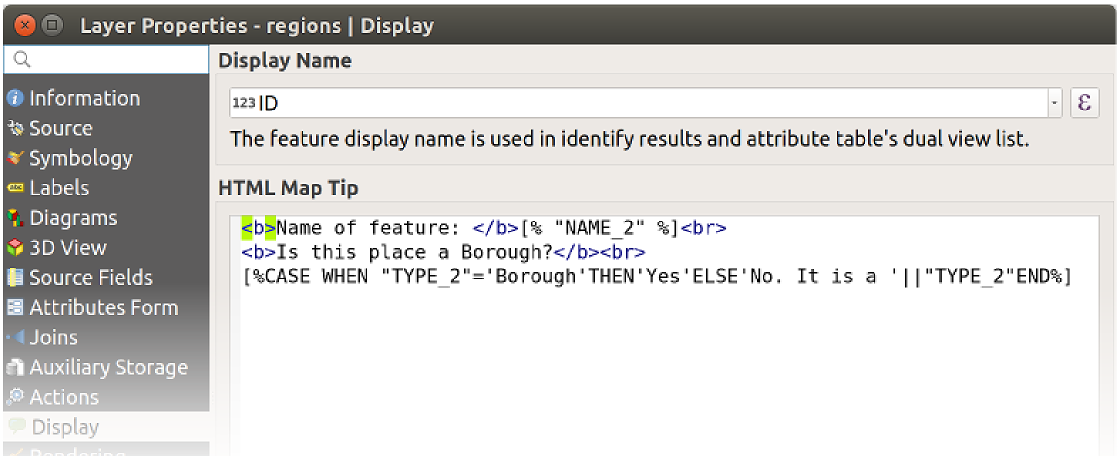


Abb. 14.59: HTML-Code für Kartenhinweis

To activate map tips, select the menu option *View* ⍰ *Show Map Tips* or click on the Show Map Tips icon of the *Attributes Toolbar*. Map tip is a cross-session feature meaning that once activated, it stays on and apply to any layer in any project, even in future QGIS sessions until it's toggled off.

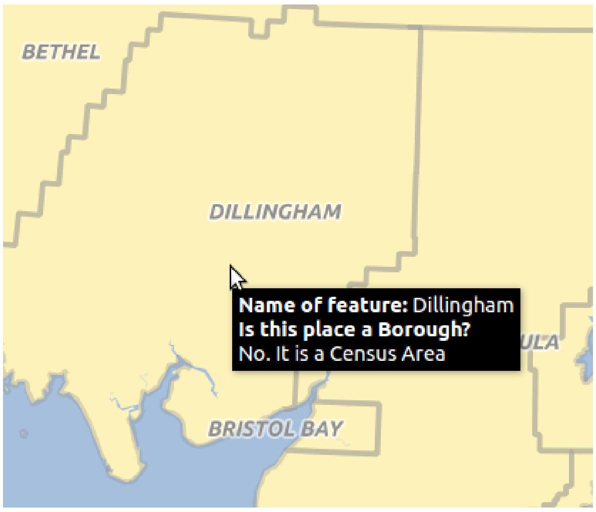


Abb. 14.60: Kartenhinweis erstellt mit HTML-Code

14.1.13 Rendering Properties

Maßstabsabhängige Sichtbarkeit

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 *Maßstabsabhängige Layeranzeige* for more information.

Simplify geometry

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).

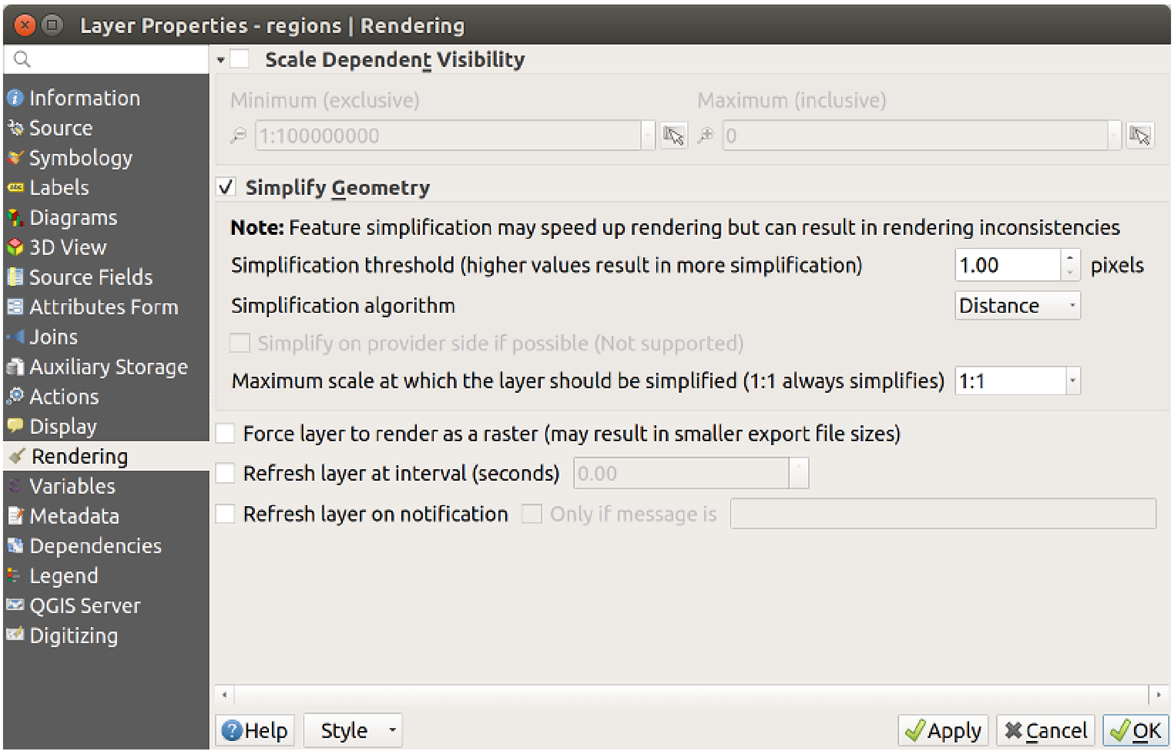


Abb. 14.61: Dialog Layergeometrien vereinfachen

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 Symbolayern sein.

While rendering extremely detailed layers (e.g. polygon layers with a huge number of nodes), this 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.

Refresh layer at interval (seconds): set a timer to automatically refresh individual layers at a matching interval. Canvas updates are deferred in order to avoid refreshing multiple times if more than one layer has an auto update interval set.

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.

14.1.14 Variables Properties

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.

14.1.15 Metadata Properties

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 currently saved in the project file. They can also be saved in a `.qmd` file alongside file based layers or in a local `.sqlite` database for remote layers (e.g. PostGIS).

14.1.16 Dependencies Properties

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.

14.1.17 Legend Properties

The *Legend* properties tab provides you with advanced settings for the *Layers panel* and/or the *print layout legend*. These options include:

- *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.

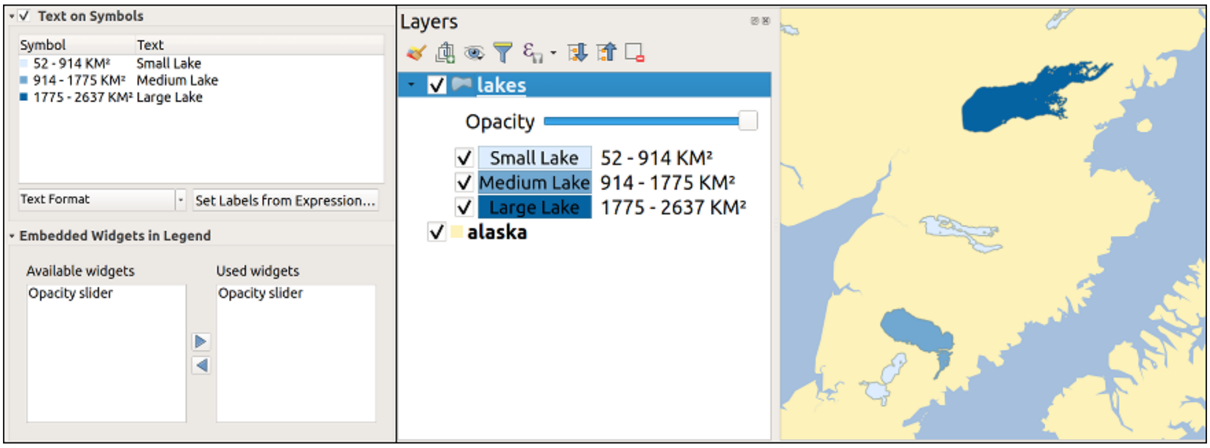


Abb. 14.62: Setting text on symbols (left) and its rendering in the *Layers* panel (right)

- 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.

14.1.18 QGIS Server Properties

The *QGIS Server* tab consists of *Description*, *Attribution*, *MetadataURL*, and *LegendUrl* sections.

You can add or edit a title and abstract for the layer in the *Description* section. It's also possible to 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 *DataUrl* field.

Use *Attribution* to get attribute data from an XML metadata catalog.

In *MetadataUrl*, you can define the general path 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.

Im Abschnitt *LegendenUrl*, können Sie der URL eines Legendenbilds in dem URL Feld anbieten. Sie können das Format Drop-Down-Menü nutzen, um das entsprechende Format des Bilds anzuwenden. Zur Zeit werden png, jpg und jpeg Formate unterstützt.

14.1.19 Digitizing Properties

The *Digitizing* tab gives access to options that help to ensure the quality of digitized geometries.

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.

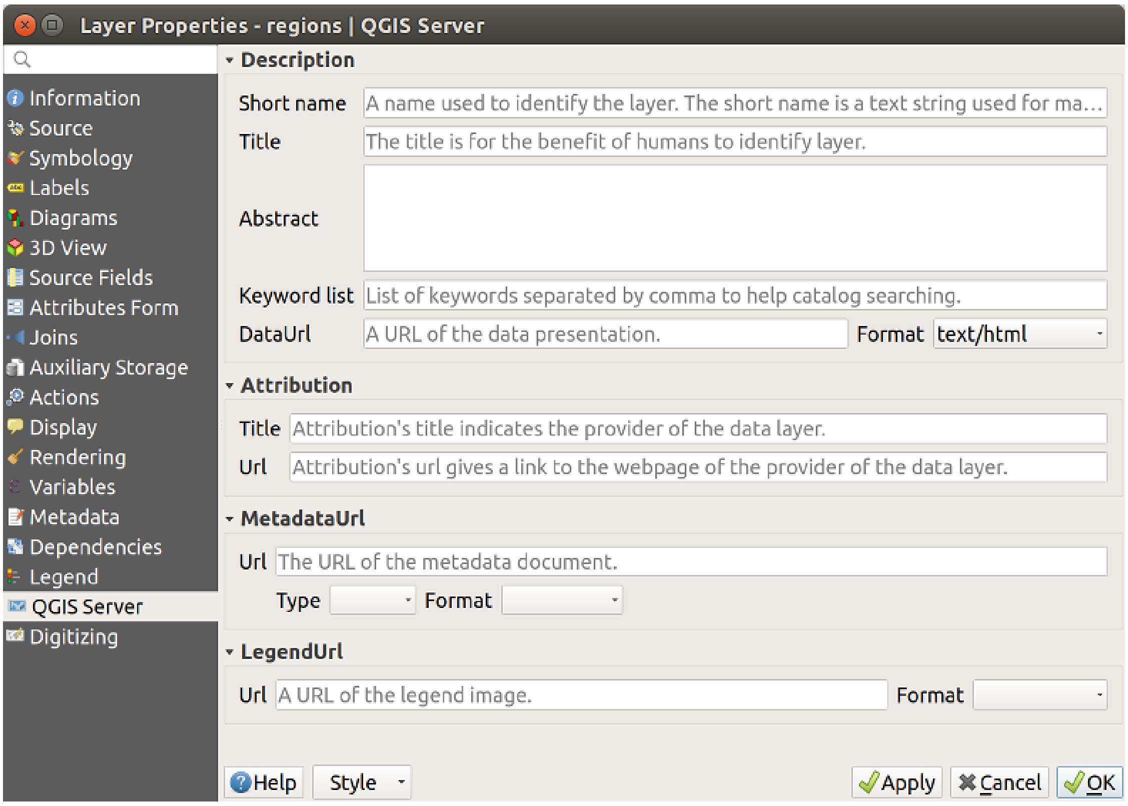


Abb. 14.63: QGIS Server tab in vector layers properties dialog

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.

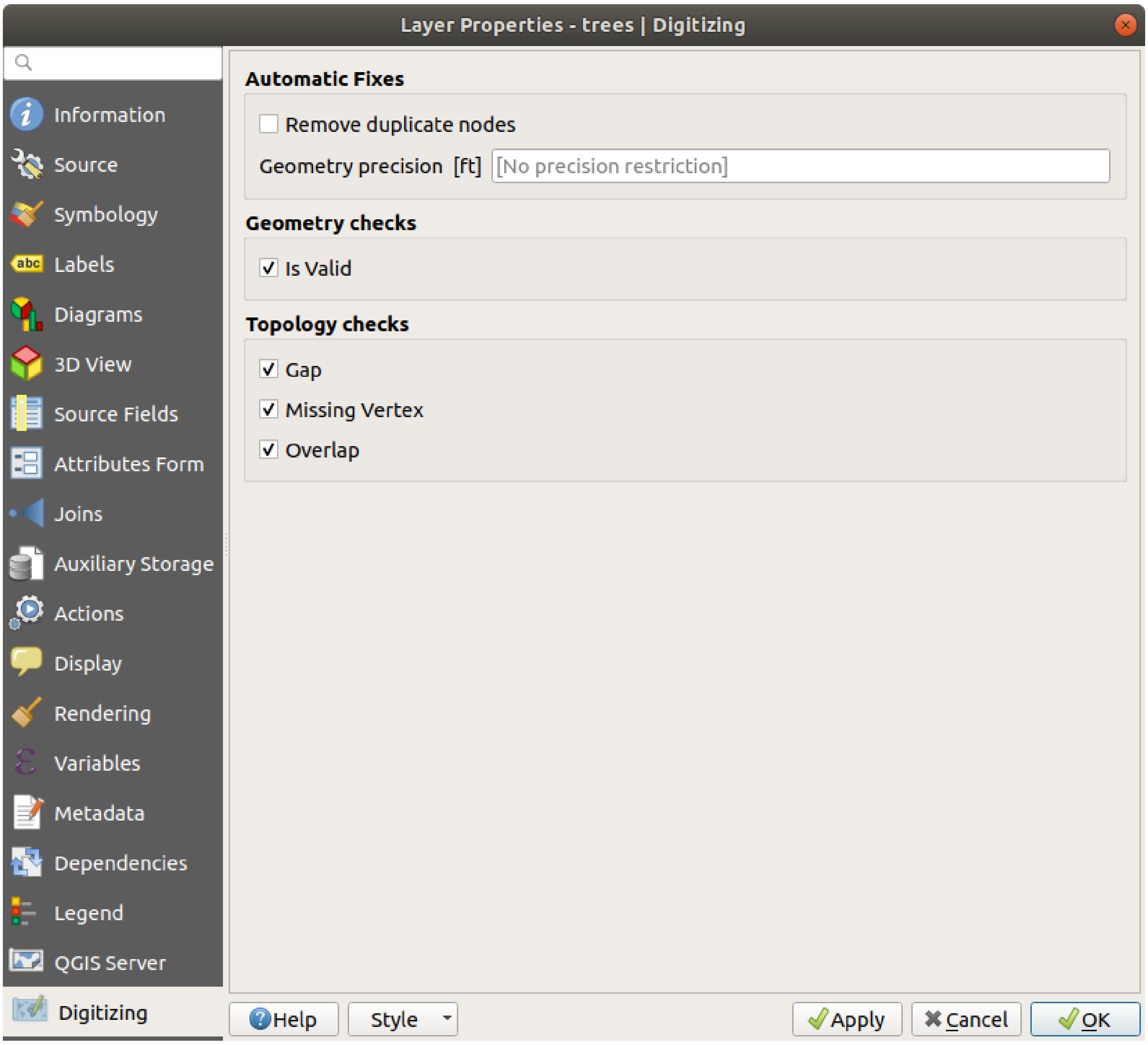
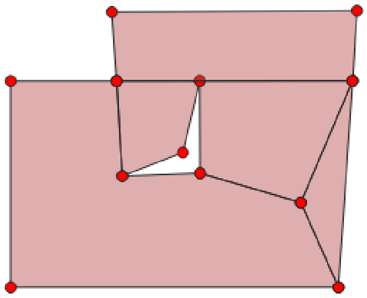
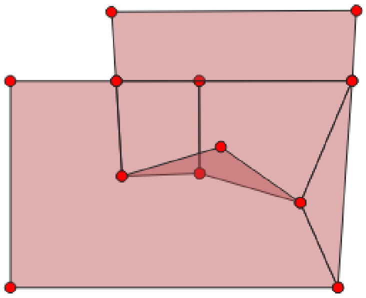
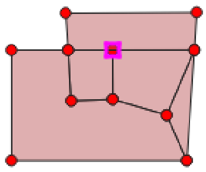


Abb. 14.64: The QGIS Digitizing tab in the vector layers properties dialog

Topology check option	Illustration
The Gap check will check for gaps between neighbouring polygons.	
The Overlap check will check for overlaps between neighbouring polygons.	
The Missing vertex check will check for shared boundaries of neighbouring polygons where one border misses a vertex which is present on the other one.	

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.

14.2 Ausdrücke

Auf Basis von Layerdaten und vorhanden oder benutzerdefinierten Funktionen bieten **Ausdrücke** ein mächtiges Werkzeug, um Attributwerte, Geometrien und Variablen zu verarbeiten. Dies kann genutzt werden, um dynamisch die Symbolisierung, die Lage von Beschriftungen, die Werte in Diagrammen, die Höhe von Layoutelementen anzupassen oder auch um Features zu selektieren oder virtuelle Felder anzulegen.

14.2.1 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 ε 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:

- den *Ausdrucksbereich* der die Erstellung und Prüfung von Ausdrücken mit Hilfe einer Liste von vordefinierten Funktionen erleichtert;
- den *Funktionseditor* der bei der Erstellung angepasster Funktionen und damit der Erweiterung der Liste der Funktionen unterstützt.

Einige Anwendungsfälle für Ausdrücke:

- Aus dem Feldrechner berechnen Sie ein „pop_density" Feld, indem Sie die existierenden Felder „total_pop" und „area_km2" nutzen:

```
"total_pop" / "area_km2"
```

- 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␣
↪density'
     WHEN "pop_density" >= 150 THEN 'High population density'
END
```

- Update a region layer field with the names (comma separated) of contained airports:

```
aggregate('airport_layer', 'concatenate', "name", within($geometry,␣
↪geometry(@parent)), ', ')
```

- 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
```

Der vorhergehende Ausdruck kann ebenso genutzt werden, um die in der Karte anzuzeigenden bzw. zu beschriftenden Objekte festzulegen.

Die Nutzung von Ausdrücken eröffnet vielfältige Möglichkeiten.

Tipp: Nutzen Sie Parameternamen, um die Lesbarkeit von Ausdrücken zu verbessern

Einige Funktionen erfordern viele zu erstellende Parameter. Die Ausdrucksverarbeitung erlaubt die Verwendung von Namen für Parameter. Das heißt an Stelle der kryptischen Ausdrucksschreibweise `clamp( 1, 2, 9)` kann die Schreibweise `clamp( min:=1, value:=2, max:=9)` verwendet werden. Das erlaubt es auch, die Reihenfolge der Argumente des Ausdrucks zu tauschen, z.B. `clamp( value:=2, max:=9, min:=1)`. Die Verwendung von Parameternamen hilft, zu verdeutlichen, worauf sich die Argumente eines Ausdrucks beziehen. Das ist hilfreich, um Ausdrücke zu einem späteren Zeitpunkt noch zu verstehen!

14.2.2 Liste der Funktionen

Der Reiter *Ausdruck* stellt die Hauptschnittstelle zur Eingabe von Ausdrücken unter Nutzung von Funktionen, Layerattributen und Werten dar. Er enthält die folgenden Widgets:

- Einen Editorbereich zur Eingabe oder Einfügen von Ausdrücken. Autovervollständigung steht in diesem Bereich zur Beschleunigung der Eingabe zur Verfügung:
  - Die zum Eingabetext passenden Variablen, Funktionsnamen und Feldnamen werden unterhalb der Eingabe angezeigt. Nutzen Sie die `Up` und `Down` Tasten, um die Einträge durchzublättern 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:

  - *Unterstreichung*: 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 ) )
  ),
    aggregate( -- finds airports at the same altitude in the region
      'airports',
      'concatenate',
      "NAME",
      filter := within( $geometry, geometry( @parent ) )
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        and "ELEV" = @airport_alti
    )
    || ' : ' || @airport_alti || 'm'
    -- using || allows regions without airports to be skipped
)
```

- In der *Ausgabevoransicht* unterhalb des Ausdruckseditors wird das Resultat des Ausdrucks angewendet auf das erste Objekt des Layers dargestellt. Im Falle eines Fehlers wird eine Fehlermeldung angezeigt und die Details zur Fehlermeldung können über den bereit gestellten link angesehen werden.
- 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.

Sobald in der Funktionsauswahl ein Feld selektiert ist, wird ein Widget zu den Werten dieses Attributes angezeigt. Mit Doppelklick auf einen Wert, wird der Wert in den Ausdruckseditor übernommen.

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.

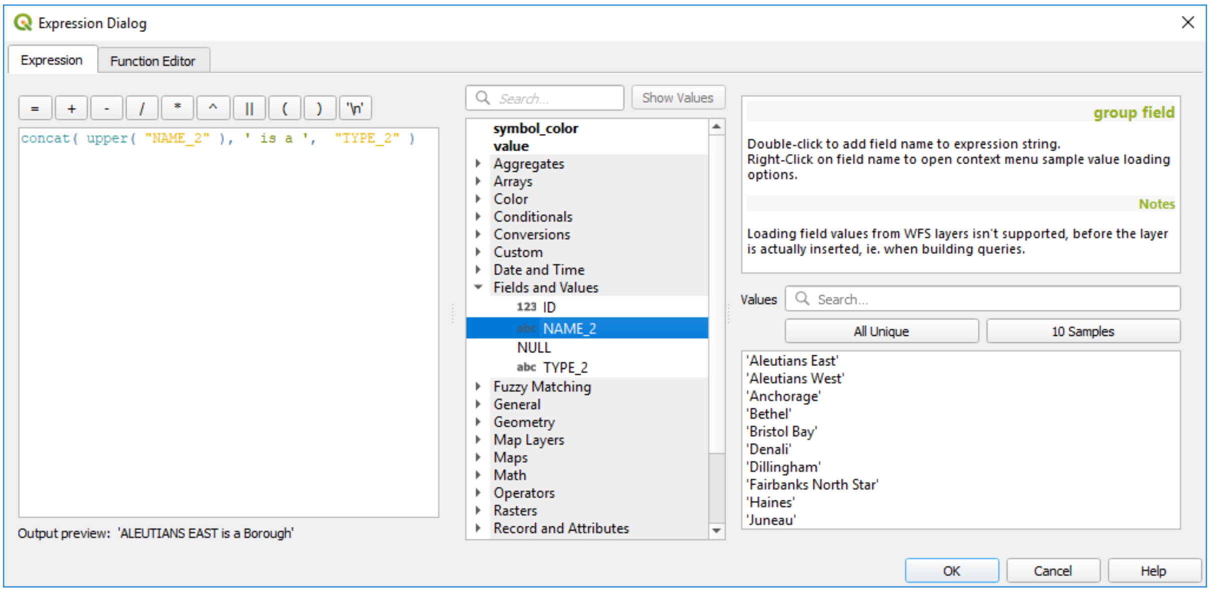


Abb. 14.65: Der Ausdruck Reiter

Aggregatsfunktionen

Diese Gruppe enthält Funktionen, die Werte über Layern und Feldern aggregieren.

Funktion	Beschreibung
aggregate	Liefert einen mit Objekten eines anderen Layers berechneten Aggregatwert
array_agg	Liefert ein Array der aggregierten Werte eines Feldes oder Ausdrucks
collect	Liefert eine mehrteilige Geometrie der aggregierten Geometrien des Ausdrucks
concatenate	Returns all aggregated strings from a field or expression joined by a delimiter
concatenate_unique	Returns all unique aggregated strings from a field or expression joined by a delimiter
count	Liefert die Anzahl der passenden Objekte
count_distinct	Liefert die Anzahl von eindeutigen Werten
count_missing	Liefert die Anzahl von fehlenden Werten (NULL)
iqr	Liefert den berechneten Interquartilbereich eines Feldes oder Ausdrucks
majority	Liefert den aggregierten Mehrheitswert (am häufigsten auftretender Wert) eines Feldes oder Ausdrucks
max_length	Liefert die maximale Zeichenkettenlänge eines Feldes oder Ausdrucks
maximum	Liefert den aggregierten Maximalwert eines Feldes oder Ausdrucks
mean	Liefert den aggregierten Mittelwert eines Feldes oder Ausdrucks
median	Liefert den aggregierten Medianwert aus einem Feld oder Ausdruck
min_length	Liefert die minimale Zeichenkettenlänge eines Feldes oder Ausdrucks
minimum	Liefert den aggregierten minimalen Wert eines Feldes oder Ausdrucks
minority	Liefert den aggregierten Minderheitswert (am wenigsten auftretender Wert) eines Feldes oder Ausdrucks
q1	Liefert das berechnete erste Quartil eines Feldes oder Ausdrucks
q3	Liefert das berechnete dritte Quartil eines Feldes oder Ausdrucks
range	Liefert den aggregierten Wertebereich (maximum - minimum) eines Feldes oder Ausdrucks
relation_aggregate	Liefert einen aus allen passenden Kindobjekten aggregierten Wert einer Layerbeziehung
stdev	Liefert den aggregierten Standardabweichungswert eines Feldes oder Ausdrucks
sum	Liefert den aggegierten summierten Wert eines Feldes oder Ausdrucks

Beispiele:

- Gebe den Maximalwert für das Feld „passengers“ und der Gruppierung der Objekte nach dem Feld „station_class“ aus:

  ```
  maximum("passengers", group_by:="station_class")
  ```

- Berechne die Anzahl der Passagiere für Bahnhöhe innerhalb des aktuellen Atlas-Objektes:

  ```
  aggregate('rail_stations','sum',"passengers",
    intersects(@atlas_geometry, $geometry))
  ```

- Gebe dern Mittelwert des Feldes „field_from_related_table“ für alle passenden Kindobjekte bezüglich der Relation ‚my_relation‘ des Layers aus:

  ```
  relation_aggregate('my_relation', 'mean', "field_from_related_table")
  ```

  oder:

  ```
  relation_aggregate(relation:='my_relation', aggregate := 'mean',
    expression := "field_from_related_table")
  ```

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.

Funktion	Beschreibung
array	Liefert ein Array mit allen zum Parameter passenden Werten
array_all	Returns true if an array contains all the values of a given array
array_append	Liefert einen Array an den der gegebene Wert am Ende angehängt ist
array_cat	Liefert einen Array der alle gegeben Arrays aneinanderhängt
array_contains	Gibt wahr aus, wenn das Array den gegebenen Wert enthält
array_distinct	Gibt ein Array mit den eindeutigen Werten des gegebenen Arrays aus
array_filter	Liefert einen Array der nur die Werte enthält, für die ein Ausdruck wahr ist
array_find	Gibt den Index eines Wertes im Array (0 für den ersten Wert). Gibt -1 zurück, wenn der Wert nicht im Array enthalten ist
array_first	Gibt den ersten Wert eines Arrays zurück
array_foreach	Gibt ein Array mit dem Ergebnis der Auswertung des gegebenen Ausdrucks für jedes Element zurück
array_get	Liefert den n-ten Wert eines Arrays (0 für den ersten Wert)
array_insert	Liefert ein Array in das der gegebene Wert an der gegebenen Position eingefügt wurde
array_intersect	Gibt wahr zurück, wenn ein Element aus array_1 in array_2 enthalten ist
array_last	Gibt den letzten Wert eines Arrays zurück
array_length	Gibt die Anzahl der Elemente eines Arrays zurück
array_prepend	Liefert ein Array bei dem der gegebene Wert am Anfang eingefügt ist
array_remove_all	Gibt ein Array zurück, aus dem die gegebenen Wert entfernt sind
array_remove_at	Liefert ein Array aus dem das Element mit dem gegebenen Index entfernt wurde
array_reverse	Liefert das gegebene Array in umgekehrter Reihenfolge zurück
array_slice	Gibt einen Teil eines Arrays zurück. Der Teil wird durch die Argumente start_pos und end_pos bestimmt.
array_sort	Returns the provided array with its elements sorted
array_to_string	Verbindet die Elemente eines Arrays zu einer durch Trennzeichen getrennten Zeichenkette. Für leere Werte kann optional ein Wert vorgegeben werden.
generate_series	Erzeugt ein Array aus einer Folge von Zahlen
regexp_matches	Gibt einen Array aller Zeichenketten zurück, die durch gruppierte Erfassung im gegebenen regulären Ausdruck aus einer Zeichenkette erfasst werden
string_to_array	Zerteilt eine Zeichenkette an der Stelle des vorgegeben Trennzeichens in einen Array. Für leere Werte kann optional eine Zeichenkette vorgegeben werden.

Farbfunktionen

Diese Gruppe enthält Funktionen zur Farbmanipulation.

Funktion	Beschreibung
color_cmyk	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Cyan-, Magenta-, Gelb- und Schwarzkomponenten
color_cmyka	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Cyan-, Magenta-, Gelb-, Schwarz- und Alpha-(Transparenz-)komponenten
color_grayscale_average	Applies a grayscale filter and returns a string representation from a provided color
color_hsl	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Farb-, Sättigungs- und Helligkeitsattributen
color_hsla	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Farb-, Sättigungs-, Helligkeits- und Alpha-(Transparenz-)Attributen
color_hsv	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Farb-, Sättigungs- und Helligkeitsattributen
color_hsva	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Farb-, Sättigungs-, Helligkeits- und Alpha-(Transparenz-)Attributen
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
color_part	Gibt eine bestimmte Komponente einer Farbzeichenkette zurück, z.B. rote oder Alpha-Komponente
color_rgb	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Rot-, Grün- und Blau-Komponenten
color_rgba	Liefert die Zeichenkettendarstellung einer Farbe nach ihrem Rot-, Grün-, Blau- und Alpha-(Transparenz-)Komponenten
create_ramp	Returns a gradient ramp from a map of color strings and steps
darker	Liefert eine dunklere (oder hellere) Farbzeichenkette
lighter	Liefert eine hellere (oder dunklere) Farbzeichenkette
project_color	Liefert eine Farbe aus dem Farbschema des Projekts
ramp_color	Liefert eine Farbe aus einem Verlaufs als Zeichenkette
set_color_part	Setzt eine bestimmte Komponente einer Farbzeichenkette, z.B. rote oder Alpha-Komponente

Funktionen für Bedingungen

Diese Gruppe enthält Funktionen um bedingte Prüfungen in Ausdrücken zu handhaben.

Funktion	Beschreibung
CASE WHEN … THEN … END	Wertet einen Ausdruck aus und liefert ein Ergebnis, wenn es wahr ist. Sie können mehrere Bedingungen testen
CASE WHEN … THEN … ELSE … END	Wertet einen Ausdruck aus und gibt ein anderes Ergebnis zurück, je nachdem es wahr oder falsch ist. Sie können mehrere Bedingungen testen
coalesce	Liefert den ersten Nicht-NULL-Wert aus der Ausdrucksliste
if	Prüft eine Bedingung und liefert unterschiedliche Ergebnisse je nach deren Ausgang
nullif(value1, value2)	Returns a null value if value1 equals value2 otherwise it returns value1. This can be used to conditionally substitute values with NULL.
try	Tries an expression and returns its value if error-free, an alternative value (if provided) or Null if an error occurs

Einige Beispiele:

- Sende einen Wert zurück wenn die erste Bedingung wahr ist, sonst einen anderen Wert:

```
CASE WHEN "software" LIKE '%QGIS%' THEN 'QGIS' ELSE 'Other' END
```

Funktionen zum Umwandlung

Diese Gruppe enthält Funktionen, um einen Datentypen in einen anderen umzuwandeln (z.B. Zeichenketten zu Ganzzahlen oder umgekehrt).

Funktion	Beschreibung
to_date	In eine Datumsobjekt umzuwandelnde Zeichenkette
to_datetime	In eine Zeitpunktobjekt umzuwandelnde Zeichenkette
to_dm	Wandelt Koordinaten in Grad, Minute um
to_dms	Wandelt Koordinaten in Grad, Minute, Sekunde um
to_int	Wandelt eine Zeichenkette in Fließkommazahlen
to_interval	Wandelt eine Zeichenkette in einen Intervalltyp (kann für Tage, Stunden, Monate usw. eines Datum verwendet werden)
to_real	Wandelt eine Zeichenkette in Fließkommazahlen
to_string	Wandelte eine Zahl in eine Zeichenkette
to_time	Wandelt eine Zeichenkette in ein Zeitobjekt

Benutzerdefinierte Funktionen

Diese Gruppe enthält vom Nutzer erstellte Funktionen. Unter *Funktions Editor* findet man weitere Details.

Datum und Zeit Funktionen

Diese Gruppe enthält Funktionen die auf Datums- und Zeitdaten angewendet werden können.

Funktion	Beschreibung
age	Liefert die Differenz zwischen zwei Terminen oder Terminzeiten als Intervall
day	Bestimmt den Tag aus einem Datum oder die Anzahl der Tage aus einem Intervall.
day_of_week	Liefert eine Zahl die dem Tag der Woche entspricht, für ein bestimmtes Datum oder eine Datumzeit
epoch	Gibt den Zeitabstand in Millisekunden zwischen der Unixepoche und einem gegebenen Datumswert an.
hour	Bestimmt den Stundenteil aus einem Datum oder die Stundenzahl aus einem Intervall
minute	Bestimmt den Minutenteil aus einem Datum oder die Minutenzahl aus einem Intervall
month	Bestimmt den Monat aus einem Datum oder die Anzahl der Monate aus einem Intervall
now	Bestimmt das aktuelle Datum und die Zeit
second	Bestimmt den Sekundenteil aus einem Datum oder die Sekundenzahl aus einem Intervall
week	Bestimmt die Wochennummer aus einem Datum oder die Anzahl der Wochen aus einem Intervall
year	Extracts the year part from a date or datetime, or the number of years from an interval

In dieser Gruppe sind einige Funktionen der Gruppen *Funktionen zum Umwandlung* (to_date, to_time, to_datetime, to_interval) und *Zeichenkettenfunktionen* (format_date) enthalten.

Einige Beispiele:

- Ausgabe des aktuellen Monats und Jahres in der Form „Monat_Nummer/Jahr“:

```
format_date(now(),'MM/yyyy')
-- Returns '03/2017'
```

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') - to_datetime(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>
```

Bemerkung: Speichern von Datums- und Zeitangaben und Zeitintervallen in Feldern

Die Fähigkeit, Werte mit *Datum*, *Zeit* und *Datum mit Zeit* direkt in Feldern zu speichern, hängt vom Datenformat ab (z.B. unterstützt das shape-Format Werte für *Datum* aber nicht *Datum mit Zeit* oder *Zeit*). Es folgen einige Vorschläge, um diese Limitation zu umgehen:

- *Datum*, *Datum mit Zeit* und *Zeit* können in Text-Feldern nach Umwandlung mit der `to_format()` Funktion gespeichert werden.
- *Zeitintervalle* können in Felder mit ganzzahligem oder dezimalem Datentyp gespeichert werden. Davor muss eine entsprechende Umwandlung erfolgen (z.B. `day()` um ein Zeitintervall in Tagen zu erhalten).

Felder und Werte

Enthält eine Liste von Feldern des Layers.

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 duch 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.

Files and Paths Functions

This group contains functions which manipulate file and path names.

Funktion	Beschreibung
base_file_name	Returns the base name of the file without the directory or file suffix.
file_exists	Returns true if a file path exists.
file_name	Returns the name of a file (including the file extension), excluding the directory.
file_path	Returns the directory component of a file path, without the file name
file_size	Returns the size (in bytes) of a file.
file_suffix	Returns the file extension from a file path.
is_directory	Returns true if a path corresponds to a directory.
is_file	Returns true if a path corresponds to a file.

Unscharfer Vergleich Funktion

Diese Gruppe enthält unscharfe Vergleichsfunktionen zwischen Werten.

Funktion	Beschreibung
hamming_distance	Liefert der Anzahl von Zeichen an korrespondieren Positionen innerhalb der Eingabezeichenketten an dem die Zeichen unterschiedliche sind.
levensheim	Liefert die minimale Anzahl von Zeichenbearbeitungen (Einfügen, Löschen oder Substitutionen) die erforderlich, sind um eine Zeichenkette zu einer anderen zu ändern. Misst die Ähnlichkeit zwischen zwei Zeichenketten
longest_common_substring	Liefert die längste gemeinsame Teilzeichenkette zwischen zwei Zeichenketten
soundex	Liefert die Soundex Darstellung einer Zeichenkette

Allgemeine Funktionen

This group contains general assorted functions.

Funktion	Beschreibung
env	Erhält eine Umgebungsvariable und gibt ihren Inhalt als Zeichenkette zurück. Wenn die Variable nicht gefunden wurde, wird `NULL` zurückgegeben.
eval	Evaluates an expression which is passed in a string. Useful to expand dynamic parameters passed as context variables or fields.
is_layer_visible	Gibt wahr zurück, wenn der angegebene Layer sichtbar ist
layer_property	Liefert eine Eigenschaft des Layers oder einen Wert seiner Metadaten. Es kann ein Layername sein, KBS, Geometrietyp, Objektanzahl…
var	Returns the value stored within a specified variable. See *Variables* below
with_variable	Erstellt und belegt eine Variable für den im dritten Argument verwendeten Ausdruck. Nützlich bei Ausdrücken, die denselben Wert mehrfach wiederholen.

Geometriefunktionen

This group contains functions that operate on geometry objects (e.g. buffer, transform, $area).

Funktion	Beschreibung
$area	Liefert die Fläche des aktuellen Objekts
$geometry	Liefert die Geometrie des aktuellen Objekts (kann zur Verarbeitung mit anderen Funktionen verwendet werden)
$length	Liefert die Länge des aktuellen Objekts
$perimeter	Liefert die Umfanglänge des aktuellen Objekts.
$x	Liefert die X-Koordinate des aktuellen Objektes
$x_at(n)	Liefert die X-Koordinate des n-ten Knotens der aktuellen Objektgeometrie
$y	Liefert die Y-Koordinate des aktuellen Objektes
$y_at(n)	Liefert die Y-Koordinate des n-ten Knotens der aktuellen Objektgeometrie
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
area	Liefert die Fläche eines Polygonobjekts. Berechnung erfolgen im Bezugssystem der Geometrie
azimuth	Returns the north-based azimuth as the angle in radians measured clockwise from the vertical on point_a to point_b
boundary	Returns the closure of the combinatorial boundary of the geometry (ie the topological boundary of the geometry - see also *Boundary*).

Fortsetzung auf der nächsten Seite

Tab. 14.3 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
bounds	Gibt die Geometrie des Begrenzungsrahmens um die Eingabegeometrie zurück. Berechnungen erfolgen im räumlichen Bezugssystem der Eingabegeometrie (siehe auch *Bounding boxes*)
bounds_height	Gibt die Höhe des Begrenzungsrahmens einer Geometrie zurück. Berechungen erfolgen im räumlichen Bezugssystem der Eingabegeometrie
bounds_width	Gibt die Breite des Begrenzungsrahmens einer Geometrie zurück. Berechnungen erfolgen im räumlichen Bezugssystem der Eingabegeometrie
buffer	Gibt eine Geometrie aller Punkte zurück, deren Abstand zur Eingabegeometrie kleiner oder gleich des vorgegebenen Abstandswertes ist. Die Berechnung erfolgen im räumlichen Bezugssystem der Eingabegeometrie (see also *Buffer*)
buffer_by_m	Erstellt einen Puffer entlang einer Liniengeomtrie bei dem der Pufferdurchmesser den m-Werten folgend wechselt (see also *Variable width buffer (by M value)*)
centroid	Liefert das geometrische Zentrum einer Geometrie (see also *Centroids*)
closest_point	Liefert einen Punkt in einer Geometrie, der der zweiten Geometrie am nächsten liegt.
collect_geometries	Collects a set of geometries into a multi-part geometry object (see also *Collect geometries*)
combine	Liefert die Kombination von zwei Geometrien
contains(a,b)	Liefert true dann und nur dann, wenn kein Punkt von b außerhalb von a liegt und mindestens ein Punkt von b in a liegt.
convex_hull	Liefert die konvexe Hülle einer Geometrie (dies stellt die minimale konvexe Geometrie dar, die alle in der Menge enthaltenen Geometrien enthält) (see also *Convex hull*)
crosses	Liefert 1 (wahr), wenn die gelieferte Geometrien einige, aber nicht alle inneren Punkte gemeinsam haben
difference(a,b)	Gibt eine Geomtrie für den Teil von Geometrie a zurück, der sich nicht mit Geometrie b überschneidet (see also *Difference*)
disjoint	Liefert 1 (true), wenn die Geometrie keinen Raum teilen
distance	Liefert den kleinsten Abstand (basierend auf dem räumlichen Bezugssystem) zwischen zwei Geometrien in einer projizierten Einheit
distance_to_vertex	Liefert den Abstand entlang der Geomtrie zu einem gegebenen Stützpunkt
end_point	Liefert den letzten Knoten einer Geometrie (see also *Extract specific vertices*)
extend	Verlängert den Anfang und das Ende einer Liniengeometrie um einen gegebenen Betrag (see also *Extend lines*)
exterior_ring	Liefert den äußeren Ring eines Polygons als Linestring. Wenn die Geometrie kein Polygon ist, ist das Ergebnis NULL
extrude(geom,x,y)	Liefert eine extrudierte Version der Eingabe- (Multi-) Kurven- oder (Multi-) Liniengeomtrie mit einer durch x und y vorgegebenen Erweiterung
flip_coordinates	Gibt eine Kopie der Eingabegeometrie mit getauschten X- und Y-Koordinaten zurück (see also *Swap X and Y coordinates*)
force_rhr	Forces a geometry to respect the Right-Hand-Rule (see also *Force right-hand-rule*)
geom_from_gml	Liefert eine Geometrie erstellt aus einer GML-Darstellung der Geometrie
geom_from_wkt	Liefert eine Geometrie aus einer Well-Known-Text (WKT)-Darstellung
geom_to_wkt	Liefert eine Well-Known-Text (WKT)-Darstellung einer Geometrie ohne SRID-Metadaten
geometry	Liefert die Geometrie eines Objekts
geometry_n	Liefert die nth Geometrie aus einer Geometriecollection oder NULL wenn die Eingabe keine Collection ist
hausdorff_distance	Returns basically a measure of how similar or dissimilar two geometries are, with a lower distance indicating more similar geometries
inclination	Liefert die Neigung gemessen vom Zenit (0) zum Fußpunkt (180) von point_a zu point_b
interior_ring_n	Liefert die Geometrie des n-ten inneren Rings aus einer Polygongeometrie oder null, wenn die Geometrie kein Polygon ist

Fortsetzung auf der nächsten Seite

Tab. 14.3 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
intersection	Liefert eine Geomtrie, die die gemeinsamen Teile von zwei Geometrien repräsentiert (siehe auch *Intersection*)
intersects	Prüft, ob sich zwei Geometrien schneiden. Liefert 1 (wahr,) wenn sich die Geometrien räumlich schneiden (einen beliebigen gemeinsamen Raum haben) und 0, wenn nicht.
intersects_bbox	Prüft ob sich der Begrenzungsrahmen einer Geometrie mit dem Begrenzungsrahmen einer anderen Geometrie überlappt. Gibt 1 (wahr) zurück, wenn sich die Begrenzungsrahmen der Geometrien räumlich überschneiden (einen beliebigen gemeinsamen Teilraum haben) und 0 wenn nicht
is_closed	Gibt wahr zurück wenn ein Linestring geschlossen ist (Start- und Endpunkt stimmen überein), falsch wenn ein Linestring nicht geschlossen ist und NULL wenn die Geometrie kein Linestring ist
length	Liefert die Länge eines Liniengeometrie Objekts (oder Länge einer Zeichenkette)
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.
line_interpolate_point	Gibt einen Punkt entsprechend der gegebenen Distanz entlang der Liniengeomtrie zurück. (siehe auch *Interpolate point on line*)
line_locate_point	Liefert den Abstand des nächsten Punktes auf einer Linie zu einem gegebenen Punkt.
line_merge	Returns a (Multi-)LineString geometry, where any connected LineStrings from the input geometry have been merged into a single linestring.
line_substring	Gibt den Teil einer Linie zwischen den gegebenen Start- und Enddistanzwerten (gemessen vom Beginn der Linie) (siehe auch *Line substring*)
m	Gibt den M-Wert einer Punktgeometrie zurück
make_circle	Erzeugt eine Kreisgeomtrie basierend auf Mittelpunkt und Radius
make_ellipse	Erzeugt eine Ellipsengeometrie basierend auf Mittelpunkt, Vorgabe der Axen und des Azimuts
make_line	Creates a line geometry from a series or an array of point geometries
make_point(x,y,z,m)	Erzeugt eine Punktgeometrie aus X- und Y- (und optional Z- oder M-) Werten
make_point_m(x,y,m)	Erzeugt eine Punktgeometrie aus X- und Y-Koordinaten und M-Werten
make_polygon	Liefert eine Polygongeometrie aus einem äußeren Ring und einer optionalen Liste von inneren Ringen
make_rectangle_3points	Creates a rectangle from 3 points
make_regular_polygon	Erzeugt ein regelmäßiges Polygon
make_square	Creates a square from a diagonal
make_triangle	Erzeugt ein Dreieckspolygon
minimal_circle	Gibt den minimalen umschließenden Kreis um die Eingabegeometrie zurück (siehe auch *Minimum enclosing circles*)
nodes_to_points	Gibt eine Multipunktgeometrie zurück, die alle Knoten der Eingabegeometrie enthält (siehe auch *Extract vertices*)
num_geometries	Liefert die Anzahl von Geometrien in Geometrycollections, oder null, wenn die Eingabegeometrie keine Collection ist
num_interior_rings	Liefert die Anzahl innerer Ringe eines Polygons oder einer Geometrycollection oder null, falls die Geometrie kein Polygon und keine Collection ist
num_points	Liefert die Anzahl von Stützpunkten einer Geometrie
num_rings	Liefert die Ringanzahl (ohne äußere Ringe) in einem Polygon oder einer Geometrycollection oder null, wenn die Eingabegeometrie weder Polygon noch Collection ist
offset_curve	Gibt eine seitlich zur Eingabeliniengeomtrie versetzte Linie zurück. Abstände werden im räumlichen Bezugssystem der Eingabegeomtrie angegeben. (siehe auch *Offset lines*)
order_parts	Sortiert die Teile eine Multigeometrie nach gegebenem Kriterium

Fortsetzung auf der nächsten Seite

Tab. 14.3 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
oriented_bbox	Gibt den minimalen gedrehten Begrenzungsrahmen der Eingabegeometrie zurück (siehe auch *Oriented minimum bounding box*)
overlaps	Prüft, ob sich zwei Geometrien überlappen. Liefert wahr, wenn sich die Geometrien Raum teilen, die gleiche Dimension haben, aber nicht komplett ineinander enthalten sind
perimeter	Liefert den Perimeter eines Polygonobjekts. Berechnung erfolgen im Bezugssystem der Geometrie
point_n	Liefert einen bestimmten Knoten einer Geometrie (siehe auch *Extract specific vertices*)
point_on_surface	Liefert einen Punkt, der innerhalb der Fläche einer Geometrie liegt (siehe auch *Point on Surface*)
pole_of_inaccessibility	Berechnet den Pol der Unzugänglichkeit für eine Fläche. Es handelt sich um den Punkt innerhalb der einer Fläche, der den größten Abstand zur Flächenumrandung hat (siehe auch *Pole of inaccessibility*)
project	Returns a point projected from a start point using a distance, a bearing (azimuth) and an elevation in radians (see also *Project points (Cartesian)*)
relate	Liefert die Dimensional Extended 9 Intersection Model (DE-9IM) Darstellung der Beziehungen zwischen zwei Geometrien
reverse	Kehrt die Richtung einer Linie durch Umkehrung der Stützpunktreihenfolge um (siehe auch *Reverse line direction*)
segments_to_lines	Erzeugt aus jedem Liniensegment einer Linie der Eingabegeometrie eine Multiliniengeometrie (siehe auch *Explode lines*)
shortest_line	Liefert die kürzeste Linie, die zwei Geometrien verbindet. Die resultierende Linie beginnt bei Geometrie 1 und endet bei Geometrie 2
simplify	Vereinfacht eine Geometrie durch Entfernen von Stützpunkten unter Nutzung einers entfernungsbasierten Schwellenwertes (siehe auch *Simplify*)
simplify_vw	Vereinfacht eine Geometrie durch Entfernen von Stützpunkten unter Nutzung eines flächenbasierten Schwellenwertes (siehe auch *Simplify*)
single_sided_buffer	Gibt eine Geomtrie zurück, die durch Pufferung einer Linie auf nur einer Seite entsteht. Abstände werden im räumlichen Bezugssystem der Geometrie angegeben (siehe auch *Single sided buffer*)
smooth	Glättet eine Geometrie indem Stützpunkte hinzugefügt werden, die die Ecken der Geometrie abrunden (siehe auch *Smooth*)
start_point	Liefert den ersten Knoten einer Geometrie (siehe auch *Extract specific vertices*)
sym_difference	Liefert eine Geometrie, die die Teile von zwei Geometrien enthält, die sich nicht überschneiden (siehe auch *Symmetrical difference*)
tapered_buffer	Erzeugt einen Puffer entlang einer Liniengeometrie wobei sich der Pufferdurchmesser gleichmäßig über die Länge der Linie ändert (see also *Tapered buffers*)
touches	Prüft, ob sich zwei Geometrien berühren. Liefert 1 (wahr), wenn sich die Geometrien mindestens einen Punkt gemeinsam haben, aber deren Inneres sich nicht überschneidet
transform	Liefert die vom Quell-KBS in das Ziel-KBS transformierte Geometrie (see also *Reproject layer*)
translate	Gibt eine räumlich versetzte Version der Geometrie zurück. Berechnungen erfolgen im räumlichen Bezugssystem der Geometrie (siehe auch *Translate*)
union	Liefert eine Geometrie die alle Punkte der Geometrien umfasst
wedge_buffer	Erzeugt einen von einer Punktgeometrie ausgehenden keilförmigen Puffer mit gegebenen Winkel und Radien (siehe auch also *Create wedge buffers*)
within (a,b)	Prüft, ob eine Geometrie in einer anderen enthalten ist. Liefert 1 (true), wenn die Geometrie komplett innerhalb der Geometrie b liegt
x	Liefert die X-Koordinate einer Punktgeometrie oder die X-Koordinate des Schwerpunktes bei anderen Geometrien
x_max	Liefert die maximale X-Koordinate einer Geometrie. Die Berechnung erfolgt im räumlichen Bezugssystem der Geometrie

Fortsetzung auf der nächsten Seite

Tab. 14.3 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
x_min	Liefert die minimale X-Koordinate einer Geometrie. Die Berechnung erfolgt im räumlichen Bezugssystem der Geometrie
y	Liefert die Y-Koordinate einer Punktgeometrie oder die Y-Koordinate des Schwerpunktes bei anderen Geometrien
y_max	Liefert die maximale Y-Koordinate einer Geometrie. Die Berechnung erfolgt im räumlichen Bezugssystem der Geometrie
y_min	Liefert die minimale Y-Koodinate einer Geometrie. die Berechnung erfolgt im räumlichen Bezugssystem der Geometrie
z	Liefert die Z-Koordinate einer Punktgeometrie

Einige Beispiele:

- Rückgabe der X-Koordinate des Schwerpunktes des aktuellen Objektes:

```
x( $geometry )
```

- Sendet einen Wert zur zugehörigen Objektfläche zurück:

```
CASE WHEN $area > 10 000 THEN 'Larger' ELSE 'Smaller' END
```

- You can manipulate the current geometry using the variable `$geometry` to create a buffer or get a point on the geometry's surface:

```
buffer( $geometry, 10 )
point_on_surface( $geometry )
```

- Given a point feature, generate a closed line (using `make_line`) around the point's 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 )␣
↪)
  )
)
```

Layout Funktionen

Diese Gruppe enthält Funktionen zur Beeinflussung der Eigenschaften von Drucklayoutelementen

Funktion	Beschreibung
item_variables	Returns a map of variables from a layout item inside this print layout

An example:

- Rückgabe des Maßstabes von ‚Map 0' im aktuellen Drucklayout:

```
map_get( item_variables('Map 0'), 'map_scale')
```

Kartenlayer

Diese Gruppe enthält eine Liste der im aktuellen Projekt verfügbaren Layer. Dies eröffnet einen komfortablen Weg, um Ausdrücke mit Verweis auf verschiedene Layer zu erstellen, z.B. beim Erstellen von *Aggregaten*, *Attribut-* oder *räumlichen* Abfragen.

It also provides some convenient functions to manipulate layers.

Funktion	Beschreibung
decode_uri	Takes a layer and decodes the uri of the underlying data provider. Available information depends on the data provider type.

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.

Funktion	Beschreibung
from_json	Loads a json-formatted string
hstore_to_map	Creates a map from a hstore-formatted string
json_to_map	Creates a map from a json-formatted string
map	Returns a map containing all the keys and values passed as pair of parameters
map_akeys	Returns all the keys of a map as an array
map_avals	Returns all the values of a map as an array
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.
map_delete	Returns a map with the given key and its corresponding value deleted
map_exist	Returns true if the given key exists in the map
map_get	Returns the value of a map, given it's key
map_insert	Returns a map with an added key/value
map_to_hstore	Merges map elements into a hstore-formatted string
map_to_json	Merges map elements into a json-formatted string
to_json	Creates a json-formatted string from a map, an array or other value

Mathematische Funktionen

Diese Gruppe enthält mathematische Funktionen (z.B. sqrt, sin und cos).

Funktion	Beschreibung
abs	Liefert den Betrag einer Zahl
acos	Liefert den inversen Kosinus eines Wert im Bogenmaß
asin	Liefert den inversen Sinus eines Wert im Bogenmaß
atan	Liefert den inversen Tangent eines Wertes im Bogenmaß
atan2(y,x)	Liefert den inversen Tangent von dy/dx durch Verwendung des Vorzeichens der beiden Argumente um den Quadranten des Ergebnisses festzulegen
azimuth(a,b)	Liefert auf Norden bezogenen Azimuth als Winkel im Bogenmaß im Uhrzeigersinn gemessen von der Vertikalen von punkt a zu punkt b
ceil	Rundet eine Zahl auf
clamp	Beschränke eine Eingabenwert auf einen gegebenen Bereichen
cos	Returns the cosine of an angle in radians

Fortsetzung auf der nächsten Seite

Tab. 14.4 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
degrees	Wandelt Bogenmaß in Grad um
exp	Liefert den Exponential eines Wert
floor	Rundet eine Zahl ab
inclination	Liefert die Neigung gemessen vom Zenit (0) zum Fußpunkt (180) von point_a zu point_b.
ln	Liefert den natürlichen Logarithmus des übergebenen Ausdruck
log	Liefert den Wert des Logarithmus des gegebenen Werts und der Basis
log10	Liefert den 10er-Logarithmus des gegebenen Ausdrucks
max	Liefert den größten Wert ungleich NULL aus einer Wertemenge
min	Liefert den kleinsten Wert ungleich NULL aus einer Wertemenge
pi	Liefert den Wert von pi für Berechnungen
radians	Wandelt Grad in Bogenmaß um
rand	Liefert eine zufällige Zahl aus dem durch Minimum und Maximum festgegelegten Bereichs (inklusiv)
randf	Liefert eine zufällige Fließkommazahl aus einem durch Minimum und Maximum gegebenen Bereichs (inklusiv)
round	Rundet auf eine bestimmte Stellenzahl
scale_exp	Wandelt einen bestimmten Wert aus einer Eingangsdomäne zu einem Leistungsbereich einer exponentielle Kurve
scale_linear	Wandelt einen gegebenen Wert des Eingabe- mit linearer Interpolation in einen Ausgabebereich
sin	Returns the sine of an angle in radians
sqrt	Liefert die Quadratwurzel eines Werts
tan	Returns the tangent of an angle in radians

Operatoren

Diese Gruppe enthält Operatoren (z. B. ., +, -, *). Beachten Sie, dass für die meisten mathematischen Fubktionen unten gilt, ist eine Eingabe NULL, dann ist das Ergebnis auch NULL.

Funktion	Beschreibung
a + b	Addition von zwei Werten (a plus b)
a - b	Subtraktion von zwei Werten (a minus b)
a * b	Multiplikation von zwei Werten (a multipliziert mit b)
a / b	Division von zwei Werten (a dividiert durch b)
a % b	Rest der Division von a durch b (beispielsweise 7 % 2 = 1, oder 2 passt in 7 dreimal mit dem Rest 1)
a ^ b	Potenz von zwei Werten (zum Beispiel, 2^2=4 oder 2^3=8)
a < b	Vergleicht zwei Werte und ergibt 1, wenn der linke kleiner dem rechten Wert ist (a ist kleiner als b)
a <= b	Compares two values and evaluates to 1 if the left value is less than or equal to the right value
a <> b	Vergleicht zwei Werte und ergibt 1 wenn sie nicht gleich sind
a = b	Vergleicht zwei Werte und ergibt 1 wenn sie gleich sind
a != b	a und b sind nicht gleich
a > b	Vergleicht zwei Werte und ergibt 1, wenn der linke größer dem rechten Wert ist (a ist größer als b)
a >= b	Vergleicht zwei Werte und ergibt 1, wenn der linke größer oder gleich dem rechten Wert ist
a ~ b	a entspricht dem regulären Ausdruck b
||	Verknüpft zwei Werte zusammen zu einer Zeichenkette. Wenn einer der beiden Werte NULL ist, ist das Ergebnis auch NULL.
‚\n‘	Setzt eine neue Linie in einer Zeichenkette ein
LIKE	Liefert 1, wenn der erste Parameter zum gegebenen Muster passt

Fortsetzung auf der nächsten Seite

Tab. 14.5 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
ILIKE	Liefert 1, wenn der erste Parameter ohne Berücksichtung der Groß-/Kleinschreibung zum gegebenen Muster passt (ILIKE kann statt LIKE benutzt werden ‚damit die Groß-/Kleinschreibung berücksichtigt wird)
a IS b	Testet, ob zwei Werte identisch sind. Gibt 1 zurück, wenn a gleich b ist
a OR b	Gibt 1 zurück, wenn Bedingung a oder Bedingung b wahr ist
a AND b	Gibt 1 zurück, wenn Bedingung a und Bedingung b wahr sind
NOT	Negiert eine Bedingung
„Column_name"	Value of the field *Column_name*, take care to not be confused with simple quote, see below
‚Zeichenkette'	einen Zeichenkettenwert, achten Sie darauf, es nicht mit doppelten Anführungszeichen zu verwechseln, siehe oben
NULL	ein Nullwert
a IS NULL	a hat keinen Wert
a IS NOT NULL	a hat einen Wert
a IN (value[,value])	a ist unter den gelistet Werten
a NOT IN (value[,value])	a ist nicht unter den gelisteten Werten

Einige Beispiele:

- Verbindet eine Zeichenkette und einen Wert von einem Spaltennamen:

```
'My feature''s id is: ' || "gid"
```

- Test if the „description" attribute field starts with the ‚Hello' string in the value (note the position of the `%` character):

```
"description" LIKE 'Hello%'
```

Processing Functions

This group contains functions that operate on processing algorithms.

Funktion	Beschreibung
parameter	Returns the value of a processing algorithm input parameter

Rasterfunktionen

Diese Gruppe enthält Funktionen zur Verarbeitung von Rasterlayern.

Funktion	Beschreibung
raster_statistic	Gibt eine Rasterlayer-Statistik von Rasterlayern zurück
raster_value	Gibt den Wert des Rasterbandes am angegebenen Punkt zurück

Funktionen zu Datensätzen und Attributen

Diese Gruppe enthält Funktionen die sich auf datensatzbezeichner beziehen.

Funktion	Beschreibung
$currentfeature	Liefert das aktuellen Objekt. Dies kann mit der Funktion ‚attribute' verwendet werde um Attributwerte des aktuellen Objekts zu bestimmen.
$id	Liefert die Objektkennung der aktuellen Zeile
attribute	Gibt den Attributwert eines Objektes zurück
attributes	Returns a *map* of all attributes from a feature, with field names as map keys
get_feature	Gibt das erste Objekt eines Layers mit dem gegebenen Attributwert zurück
get_feature_by_id	Gibt das Objekt eines Layers mit der gegebenen Objekt-ID zurück
is_selected	Gibt zurück, ob ein Objekt ausgewählt ist
num_selected	Liefert die Anzahl der ausgewählten Objekte auf einem gegebenen Layer
represent_value	Returns the configured representation value for a field value (convenient with some *widget types*)
sql_fetch_and_increment	Manage autoincrementing values in SQLite databases
uuid	Generiert einen Universally Unique Identifier (UUID) für jede Zeile. Jede UUID hat eine Länge von 38 Zeichen.

Einige Beispiele:

- Liefert das erste Objekt in Layer „LayerA" dessen Feld „id" die selben Werte wie das Feld „name" des aktuellen Objekts, hat (eine Art Jointure):

```
get_feature( 'layerA', 'id', attribute( $currentfeature, 'name') )
```

- Berechnen Sie die Fläche des verbundenen Objekts aus dem vorherigen Beispiel:

```
area( geometry( get_feature( 'layerA', 'id', attribute( $currentfeature, 'name
↪') ) ) )
```

Zeichenkettenfunktionen

Diese Gruppe enthält Funktionen für Zeichenketten (z.B. Ersetzen und in Großbuchstaben umwandeln).

Funktion	Beschreibung
char	Gibt das Zeichen mit dem zugehörigen Unicode-Kode aus
concat	Verketten mehrerer Zeichenkette zu einem
format	Formatiert eine Zeichenkette mit den gegebenen Argumenten
format_date	Formatiert einen Dateityp oder eine Zeichenkette in einen benutzerdefinierten Zeichenketten Formatstyp.
format_number	Liefert eine Zahl formatiert mit den lokalen Tausendertrennern (schneidet die Zahl auch auf die angegebene Stellenanzahl)
left(string, n)	Liefert einen Teilstring mit den ersten n Zeichen der Zeichenkette
length	Liefert die Länge einer Zeichenkette (oder Länge eines Liniengeometrie Objekts)
lower	wandelt eine Zeichenkette in Kleinbuchstaben um
lpad	Liefert eine links bis auf die gegebene Länge mit dem Füllzeichen aufgefüllte Zeichenkette
regexp_match	Liefert die Position der ersten Fundstelle eines regulären Ausdrucks in einer Zeichenkette oder 0 wenn die Teilzeichenkette nicht gefunden wurde
regexp_replace	Liefert eine Zeichenkette in der ein regulärer Ausdruck ersetzt wurde
regexp_substr	Liefert den Teil der Zeichenkette, die dem gegebenen regulären Ausdrucks entspricht
replace	Returns a string with the supplied string, array, or map of strings replaced by a string, an array of strings or paired values

Fortsetzung auf der nächsten Seite

Tab. 14.7 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
right(string, n)	Liefert einen Teilstring mit den ersten n Zeichen der Zeichenkette
rpad	Liefert eine rechts bis auf die gegebene Länge mit dem Füllzeichen aufgefüllte Zeichenkette
strpos	Liefert die Position der ersten Fundstelle einer Teilzeichenkette in einer anderen Zeichenkette oder 0 wenn die Teilzeichenkette nicht gefunden wurde
substr	Liefert eine Teilzeichenkette
title	Wandelt alle Wörter einer Zeilenkette ins Titelformat (alle Wörter in Kleinbuchstaben bis auf die Anfangsbuchstaben in Großbuchstaben)
trim	Entfernt Leerzeichen am Anfang und Ende einer Zeichenkette (Leerzeichen, Tabulatoren usw)
upper	Wandelt eine Zeichenkette in Großbuchstaben.
wordwrap	Returns a string wrapped to a maximum/minimum number of characters

Verkettung von Feldern

You can concatenate strings or field values using either `||` or `+` operators or the `concat` function, 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''s id is: ' + "gid" => triggers an error as gid is 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:

```
"country_name" || NULL => NULL
concat('My feature''s id is: ', NULL) => My feature's id is
concat("firstname", "nickname", "lastname") => Chuck Norris (if empty nickname)
"firstname" + "nickname" + "lastname" => NULL (if one field is empty)
```

- For other cases, do at your convenience:

```
'My country is ' + "country_name" + ' (' + "country_code" + ')'
'My country is ' || "country_name" || ' (' || "country_code" || ')'
concat('My country is ', "country_name", ' (', "country_code", ')')
# All the above return: My country is France (FR)
```

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`).

Funktion	Beschreibung
algorithm_id	The unique ID of an algorithm
atlas_feature	The current atlas feature (as feature object)
atlas_featureid	The current atlas feature ID

Fortsetzung auf der nächsten Seite

Tab. 14.8 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
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
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_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
grid_axis	The current grid annotation axis (eg, ‚x' for longitude, ‚y' for latitude)
grid_number	The current grid annotation value
item_id	The layout item user ID (not necessarily unique)
item_uuid	The layout item unique ID
layer	The current layer
layer_id	The ID of current layer
layer_name	The name of current layer
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)
layout_pagewidth	The active page width in the layout (in mm)
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_wkt	The WKT definition of the Coordinate reference system of the current map
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_layer_ids	The list of map layer IDs visible in the map

Fortsetzung auf der nächsten Seite

Tab. 14.8 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
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_units	The units of map measurements
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 *aggregate* 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
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_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)
user_account_name	The current user's operating system account name
user_full_name	The current user's operating system user name
Wert	The current value

Fortsetzung auf der nächsten Seite

Tab. 14.8 – Fortsetzung der vorherigen Seite

Funktion	Beschreibung
with_variable	Erlaubt das Setzen einer Variable für die Verwendung in einem Ausdruck und vermeidet so die wiederholte Neuberechnung desselben Wertes

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'
  )
)
```

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.

14.2.3 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.

Um eine neue Funtkion 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 Kode und die Hilfeinträge 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.

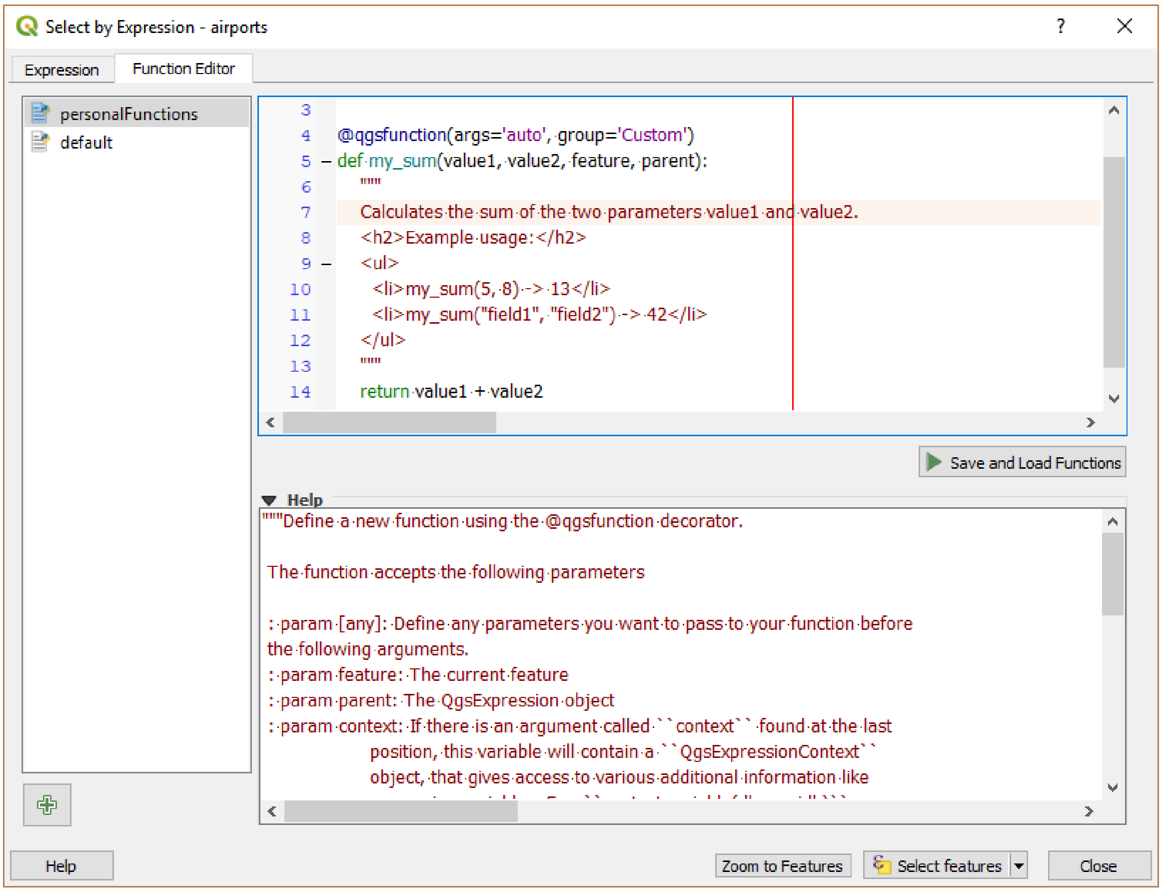


Abb. 14.66: Der Funktionseditor Reiter

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.

Hier ist ein kurzes Beispiel, wie Sie eine eigene Funktion erzeugen:

```
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
```

Diese kurze Beispiel erzeugt eine Funktion `my_sum` mit zwei Werten. Bei Nutzung der Funktionsargumentes `args='auto'` wird die Anzahl der erforderlichen Argumente aus der Anzahl der in Python definierten Argumente berechnet (minus 2 - `feature`, und `parent`).

Die Funktion kann in Ausdrücken verwendet werden:

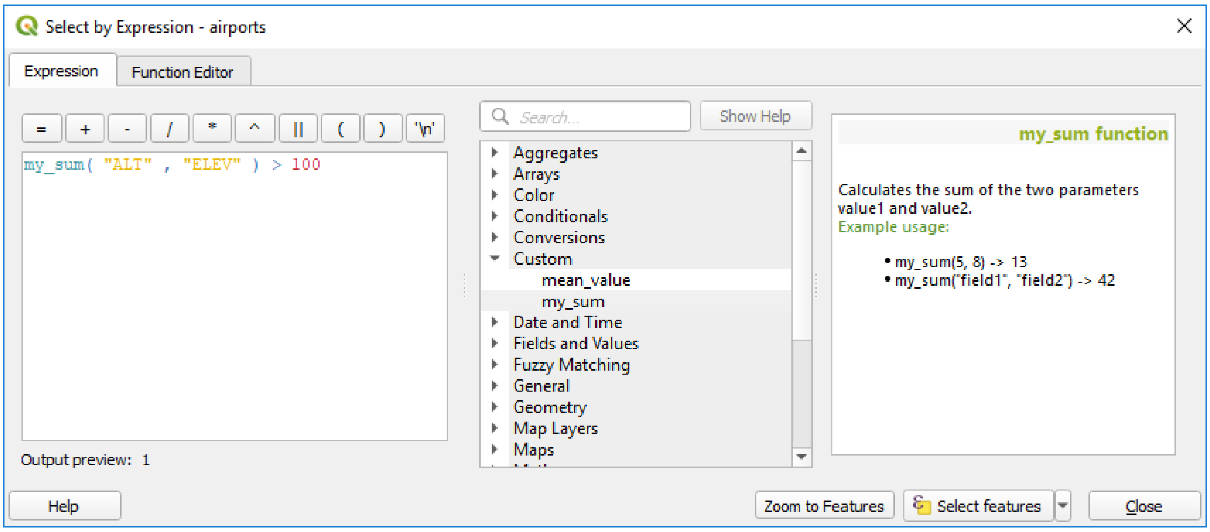


Abb. 14.67: Benutzerdefinierte Funktion im Reiter Ausdruck

Weitere Informationen über das Erstellen von Python-Kodes können in PyQGIS-Developer-Cookbook gefunden werden.

14.3 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.

14.3.1 Foreword: Spatial and non-spatial tables

QGIS allows you to load spatial and non-spatial layers. This currently includes tables supported by OGR and delimited text, as well as the PostgreSQL, MSSQL, SpatiaLite, DB2 and Oracle provider. 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 *Attributes Form Properties* to find out more.

14.3.2 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.

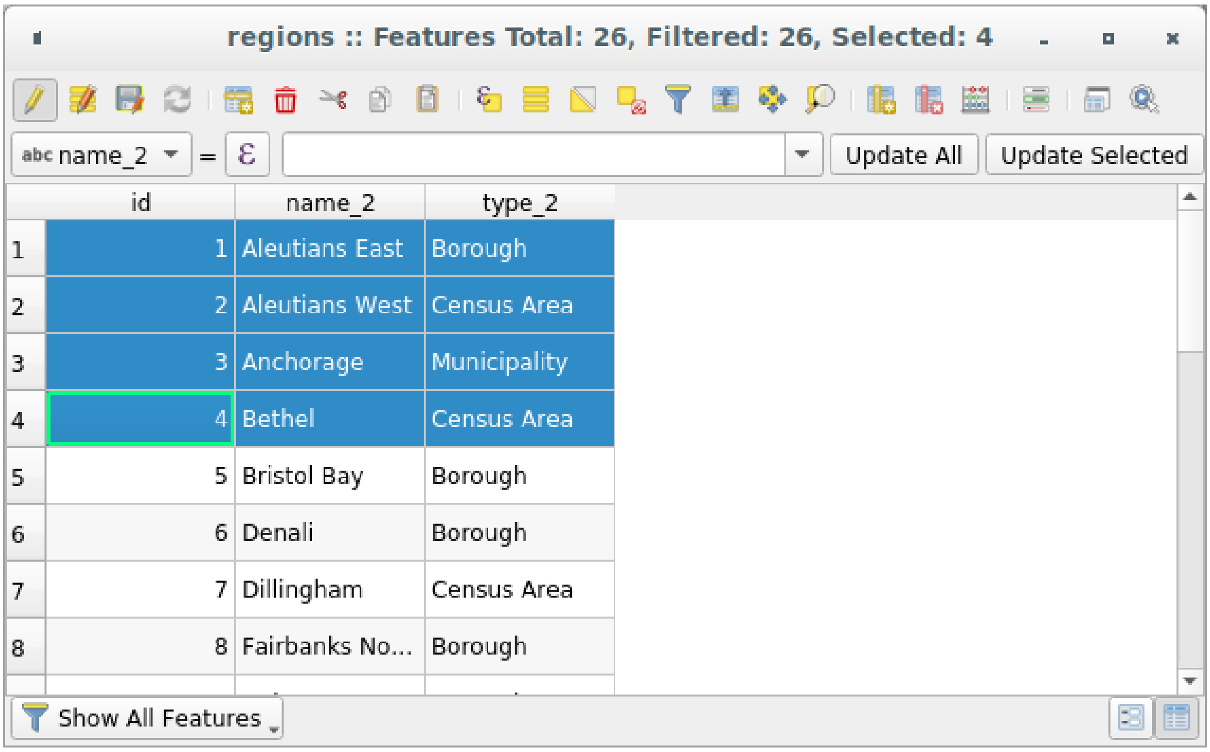


Abb. 14.68: Attributtabelle des regions Layer

Die anderen Knöpfe im Kopf der Attributtabelle stellen die folgenden Funktionalitäten zur Verfügung:

Tab. 14.9: Available Tools

Icon	Label	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`
	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		`Ctrl+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	
	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 OGR 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 *Editiere 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.
- 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 *Attributes Form Properties*). You can browse through the feature identifiers with the arrows on the bottom of the first panel. Once you markered the feature in yellow in the list it is selected in yellow on the canvas. Use the on top of the attribute table to zoom to the feature. Clicking on an entry in the list (without using the rectangles) makes a feature flash in red color once so you can see where it is situated.

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‘.

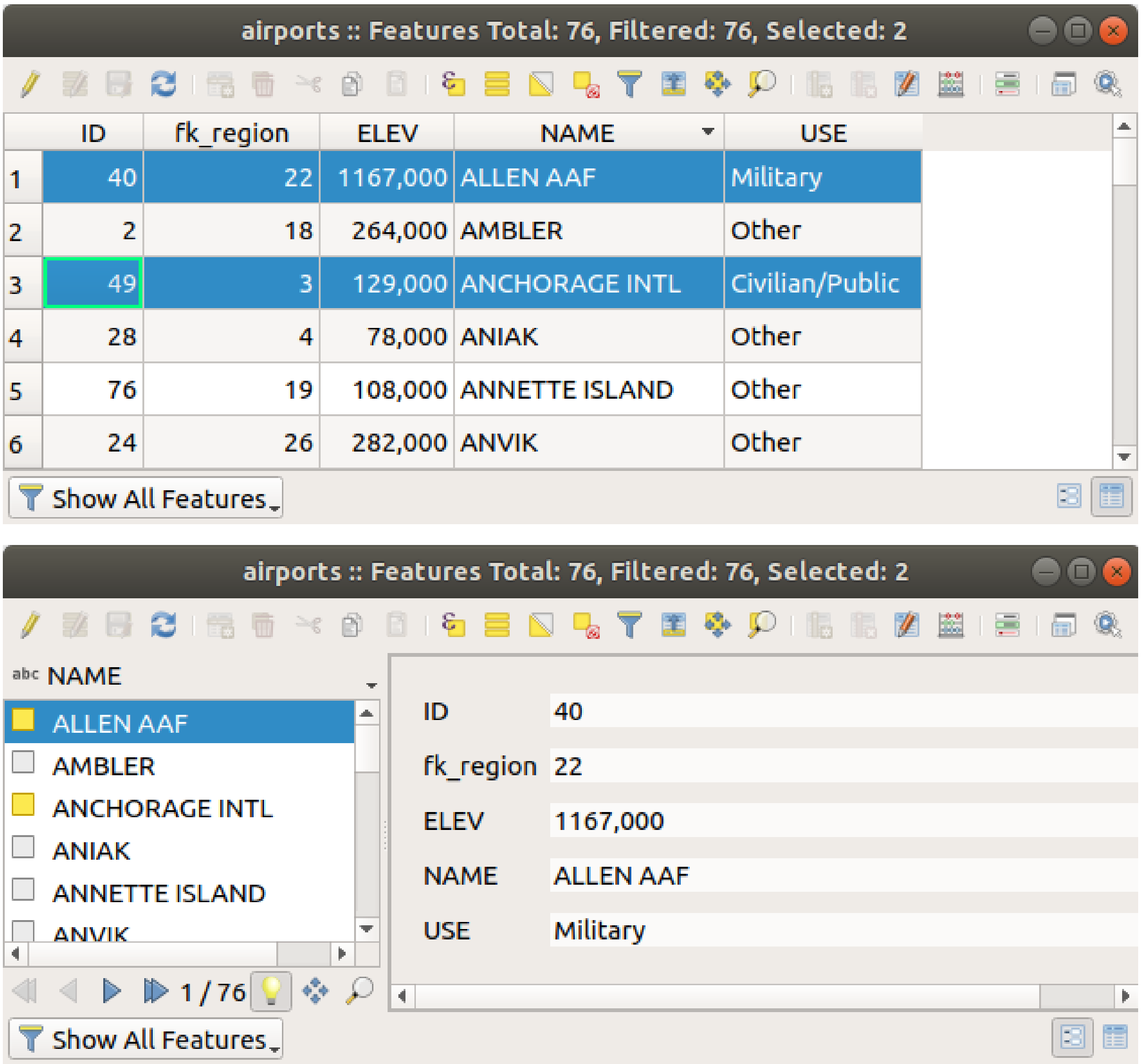


Abb. 14.69: Attribute table in table view (top) vs form view (bottom)

Configuring the columns

Right-click in a column header when in table view to have access to tools that help you configure what can be displayed in the attribute table and how.

Hiding and organizing columns and enabling actions

By right-clicking in a column header, you can choose to hide it from the attribute table. To change several columns behavior at once, unhide a column or change the order of the columns, choose *Organize columns ...*. In the new dialog, you can:

- check/uncheck columns you want to show or hide
- 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
- enable a new virtual *Actions* column that displays in each row a drop-down box or button list of actions for each row, see *Actions Properties* for more information about actions.

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
- *Autosize* to resize at the best fit the column.

It can also be changed by dragging the boundary on the right of the column heading. The new size of the column is maintained for the layer, and restored at the next opening of the attribute table.

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 row 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…).

You can enable the conditional formatting panel clicking on at the top right of the attributes window in table view (not available 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 choosen 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.

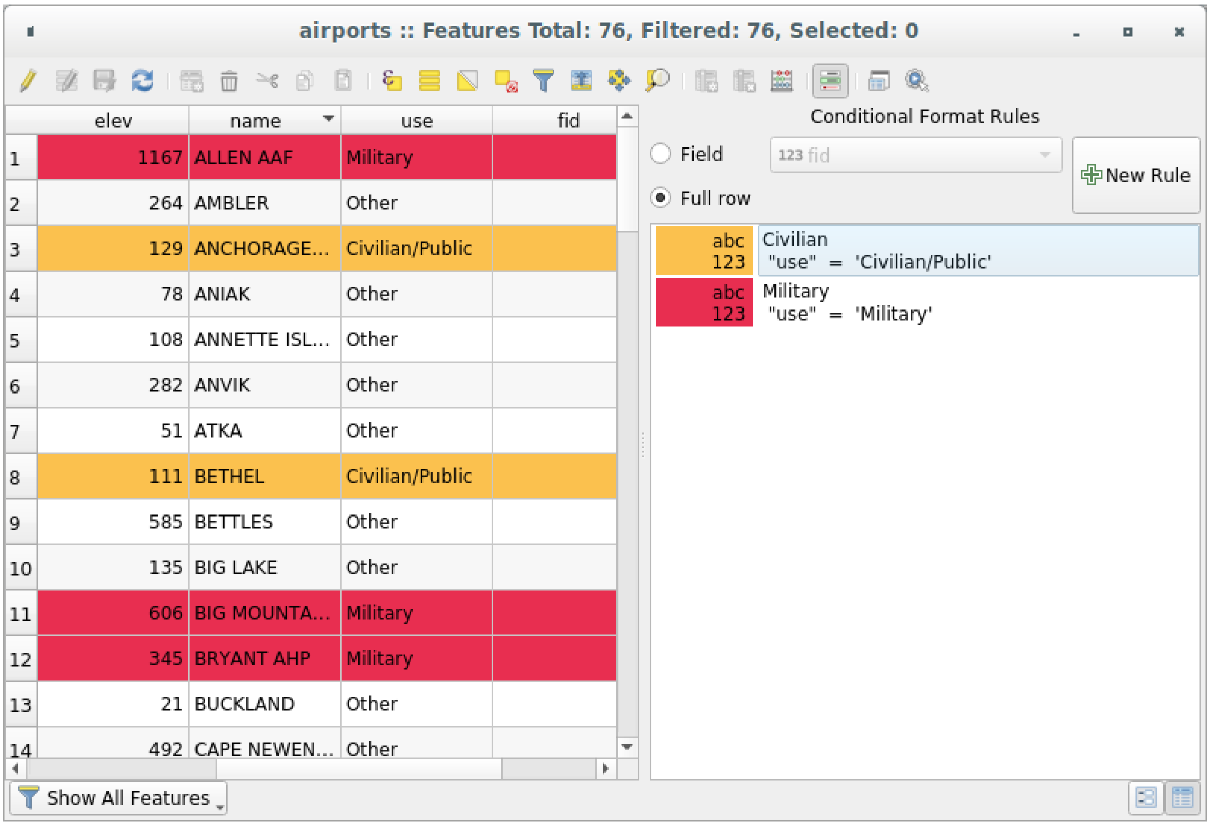


Abb. 14.70: Bedingte Formatierungsregeln für Attributtabellen

14.3.3 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 Mauspfeilposition 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 *Display Properties*). 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 Features from All Layers*
- *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*
- *Show Features visible on map*
- *Show Edited and New Features*
- *Field Filter* - allows the user to filter based on value of a field: choose a column from a list, type a value and press `Enter` to filter. Then, only the matching features are shown in the attribute table.
- *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.

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 momentaneously 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.

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.

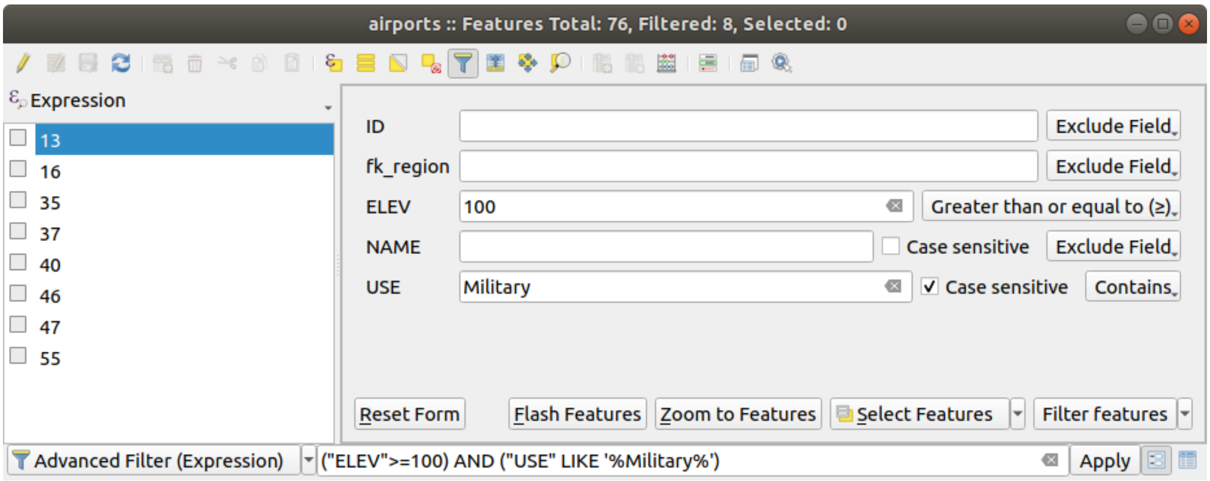


Abb. 14.71: Attribute table filtered by the filter form

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`.

14.3.4 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.

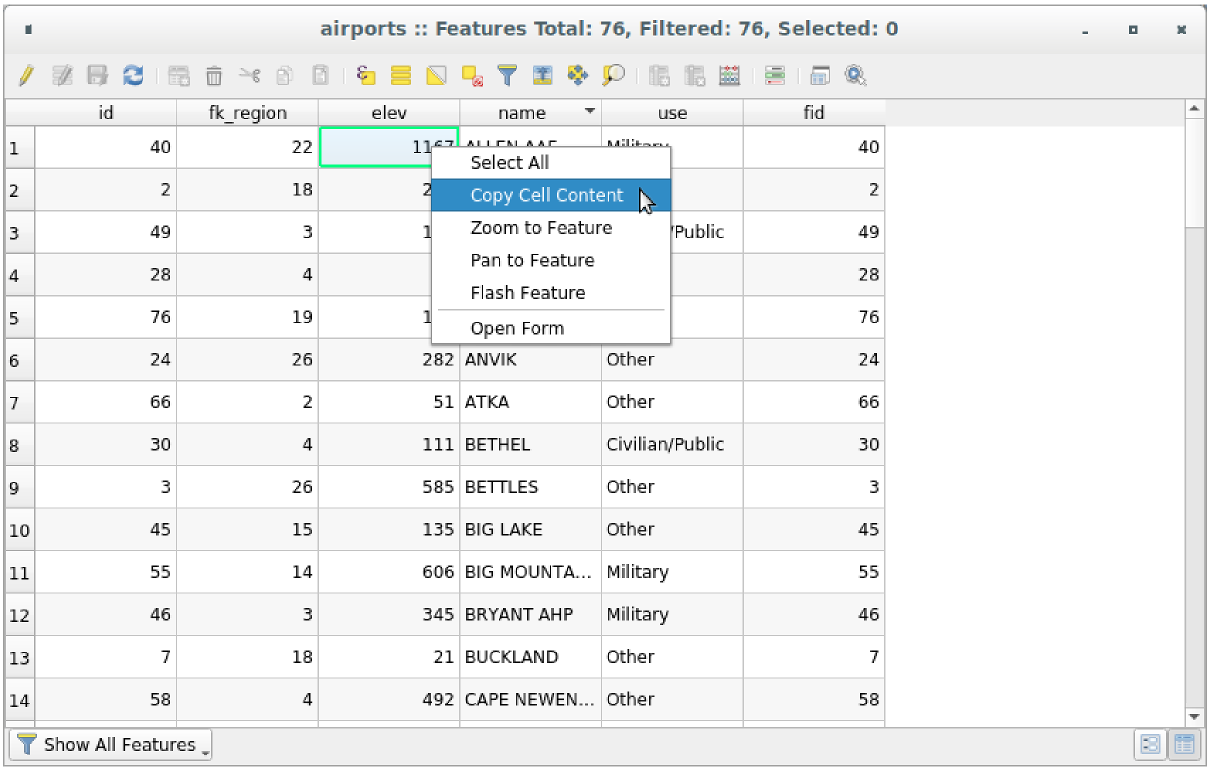


Abb. 14.72: Copy cell content button

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 *Actions Properties* 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 OGR creation options within the dialog.

14.3.5 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 *figure_field_calculator*). 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.

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.
8. Complete the expression by typing `/ 1000` in the Field calculator expression box and click *OK*.

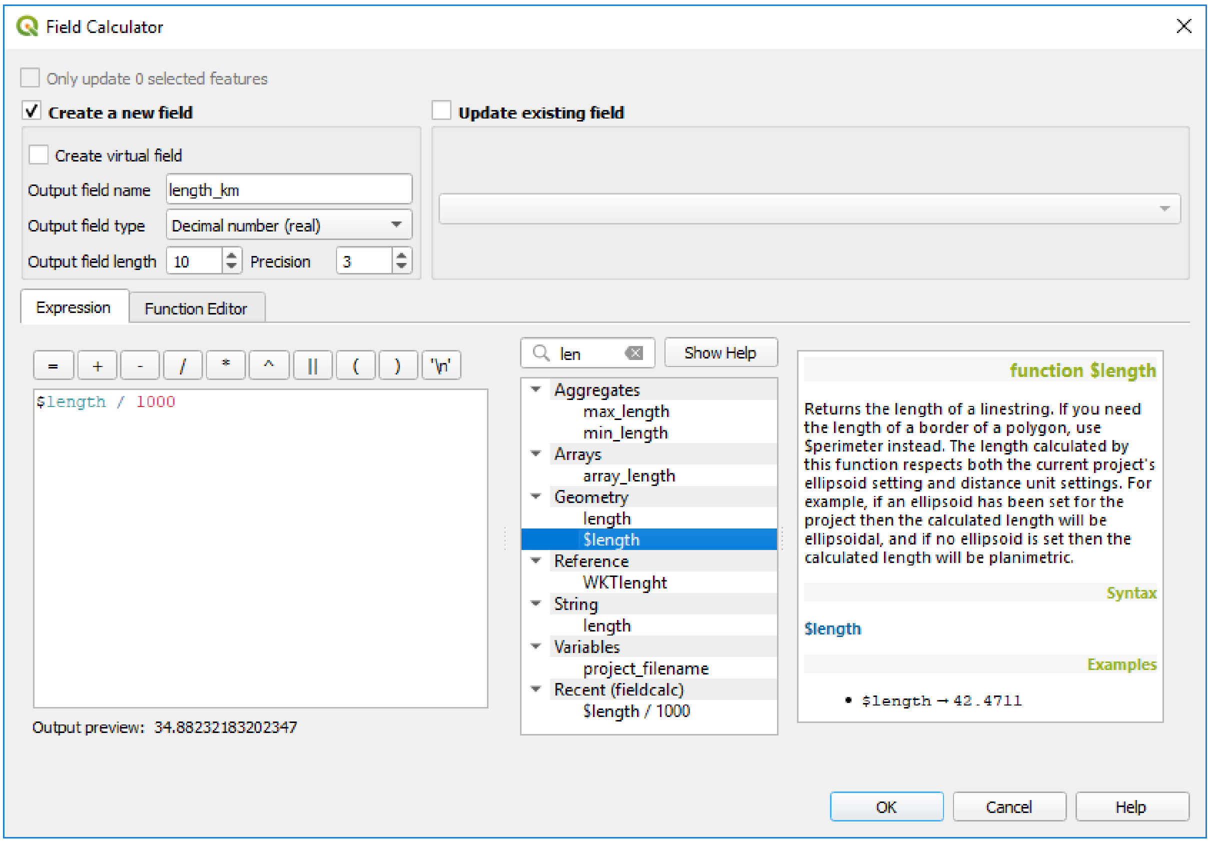


Abb. 14.73: Feldrechner

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 verfügabr 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 ε expression button.
3. Click on *Update All*, *Update Selected* or *Update Filtered* button according to your need.

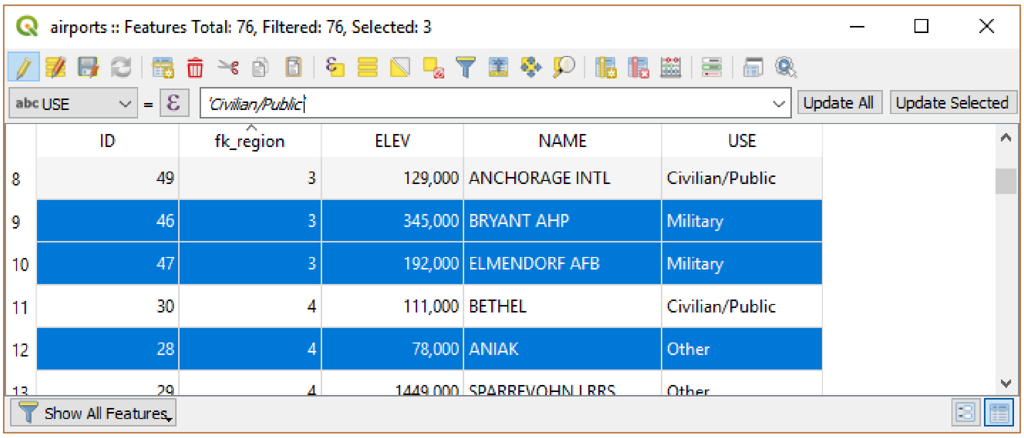


Abb. 14.74: Quick Field Calculation Bar

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.

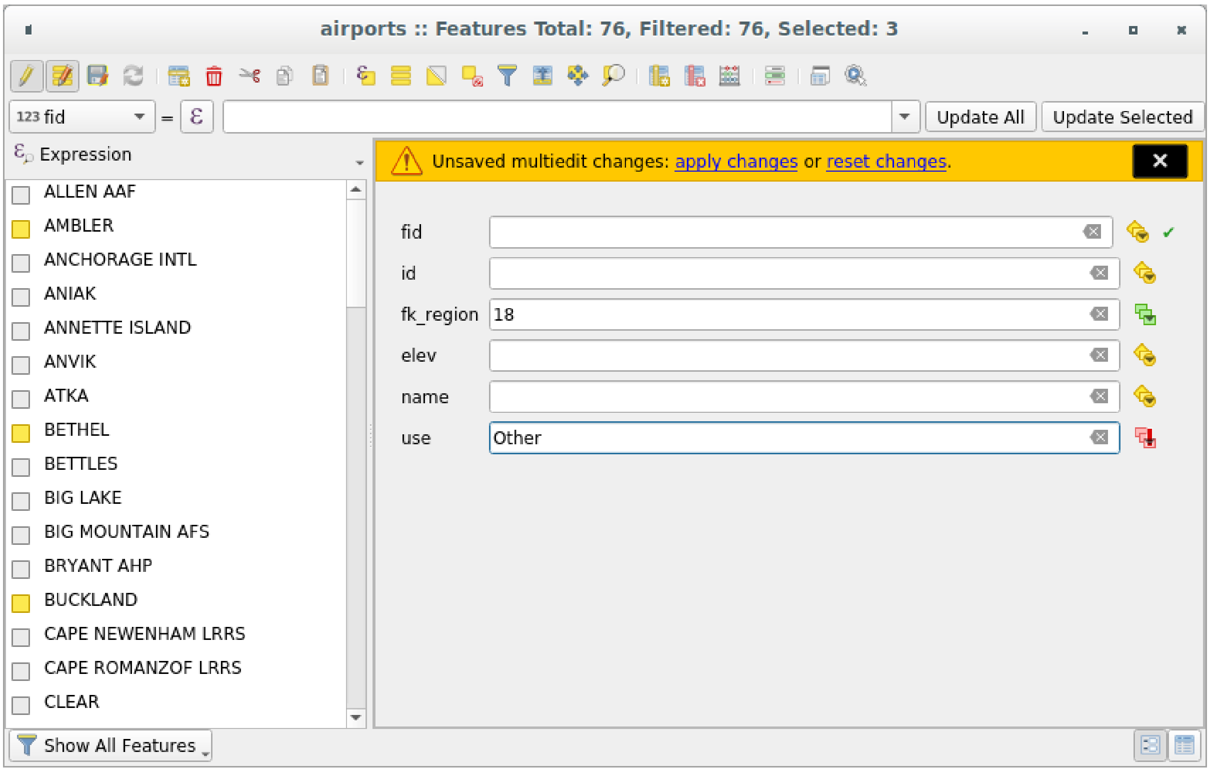


Abb. 14.75: Editing fields of multiple features

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.

14.3.6 Creating one or many to many relations

Relations are a technique often used in databases. The concept is that features (rows) of different layers (tables) can belong to each other.

Introducing 1-N relations

Als Beispiel nehmen wir einen Layer mit allen Regionen von Alaska (Polygon) in dem einige Attribute über den Namen und den Regionstyp sowie eine eindeutige ID (die als Primärschlüssel eingesetzt wird) vorhanden sind.

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.

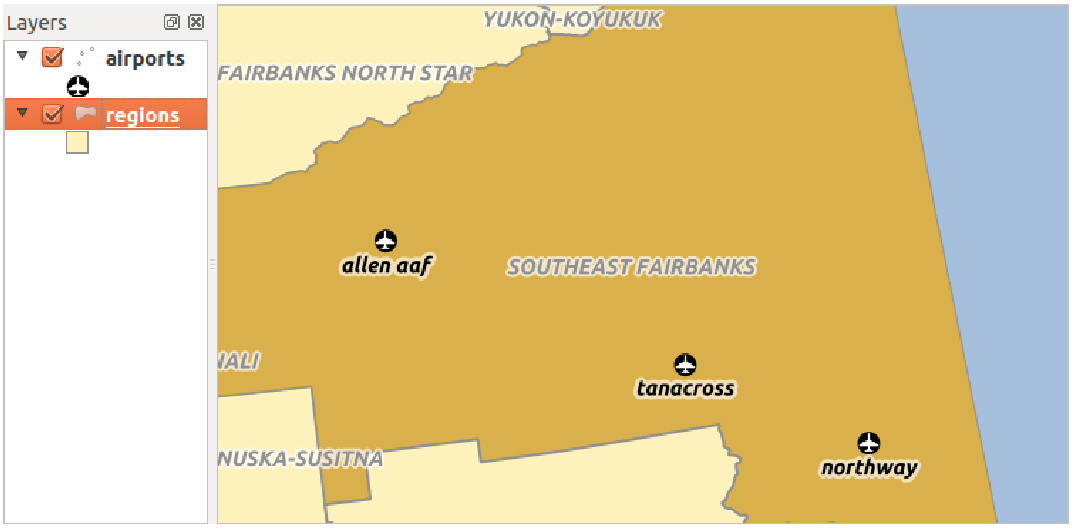


Abb. 14.76: Die Region Alaska mit Flughäfen

Layers in 1-N relations

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` shapefile and the `airports` shapefile which has a foreign key field (`fk_region`) to the layer regions. This means, that each airport belongs to exactly one region while each region can have any number of airports (a typical one to many relation).

Foreign keys in 1-N relations

In addition to the already existing attributes in the airports attribute table, you'll need 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 field fk_region will always contain an id of a region. It can be seen like a pointer to the region it belongs to. And 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) and all you have to do is to tell QGIS the relations between your tables.

Defining 1-N relations (Relation Manager)

The first thing we are going to do is to let QGIS know about the relations between the layers. This is done in *Project ⍰ 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 can define the primary key of the referenced layer, so it is `ID`. For this layer you can define multiple referenced fields by using the button.
- **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`. When using multiple field relations you can add another referencing field by using the button.
- **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.

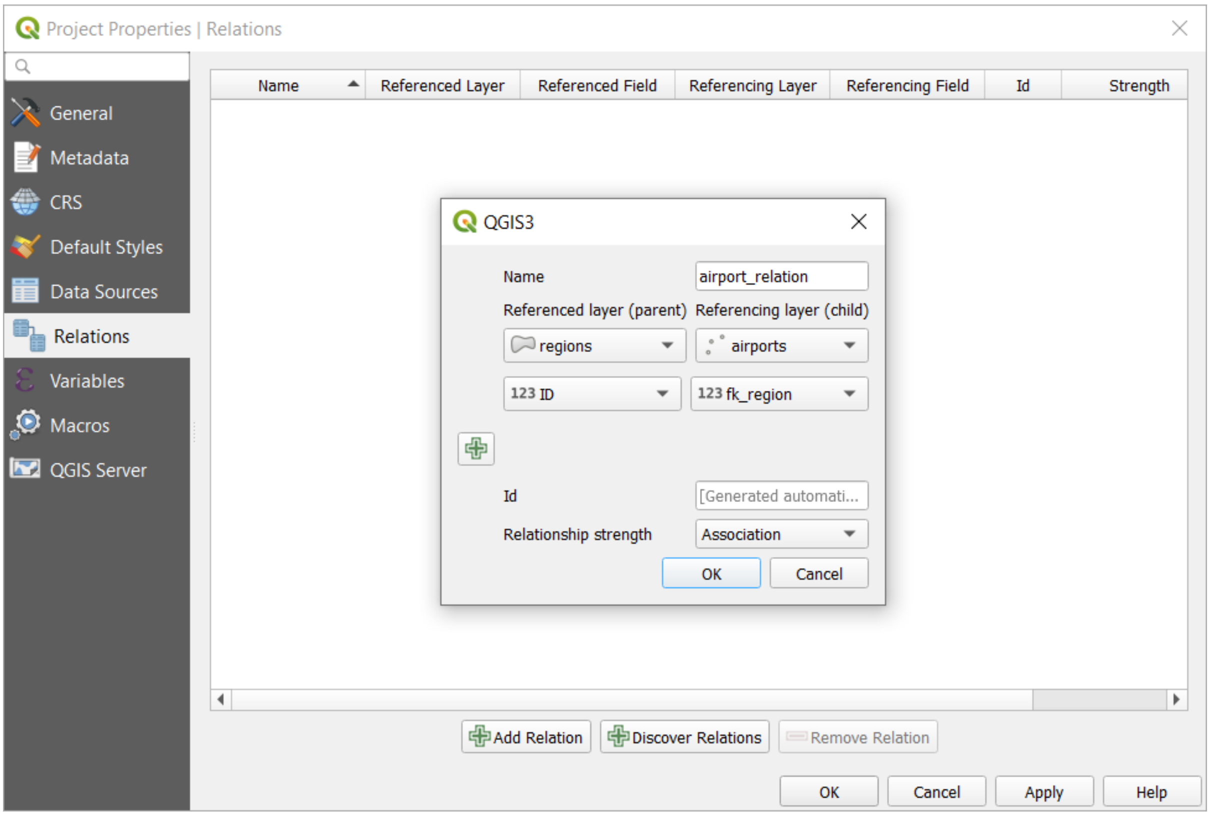


Abb. 14.77: Relation Manager

Forms for 1-N relations

Jetzt wo QGIS von der Beziehung weiss wird es dazu benutzt das Formular das es erstellt zu verbessern. Da wir die Standard Formular Methode (autogenerated) nicht verändert haben, wird es Ihrem Formular einfach ein neues Bearbeitungselement hinzufügen. Also lassen Sie uns einen Layer ‚region' in der Legende auswählen und das Objekte abfragen Werkzeug benutzen. Abhängig von Ihren Einstellungen öffnet sich das Formular entweder direkt oder Sie müssen den Identifizieren Dialog unter Aktionen öffnen.

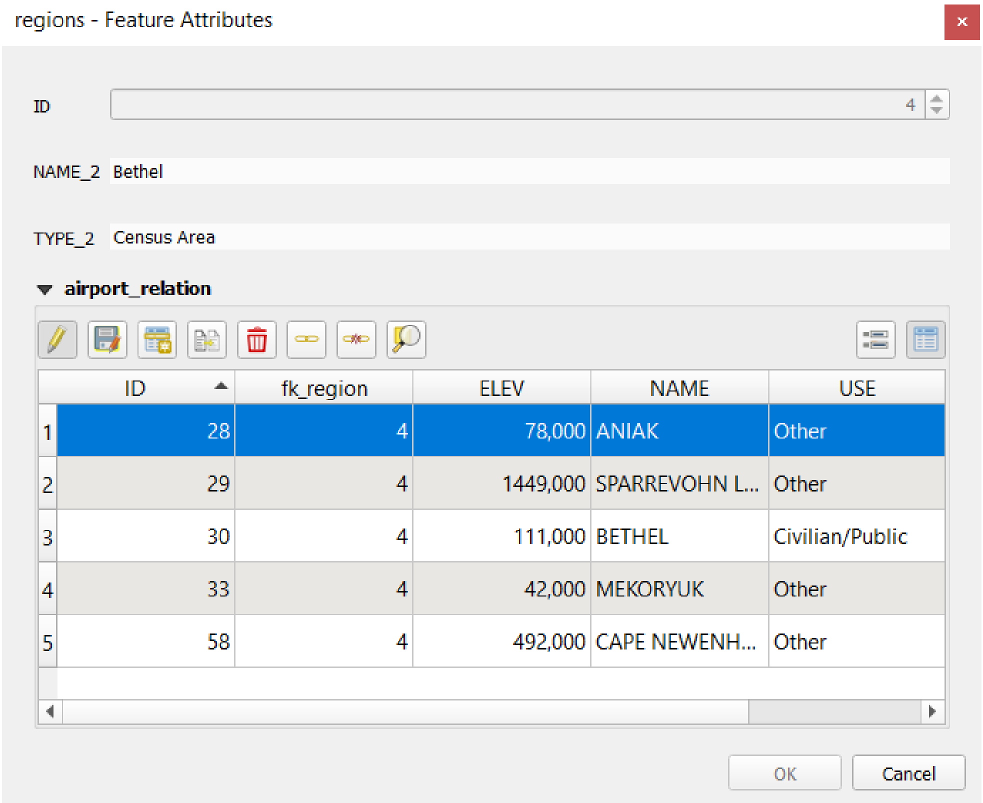


Abb. 14.78: identifikationsergebnis Dialog von regions mit Beziehung zu airports

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:

- Der Knopf ist zum Umschalten des Bearbeitungsmodus da. Seien Sie sich dessen bewusst dass es den Bearbeitungsmodus des ‚airport' Layers umschaltet, trotzdessen wir uns im Objektformular eines Objektes aus dem ‚region' Layer befinden. Die Tabelle jedoch stellt Objekte des ‚aiport' Layer dar.
- The button is for saving all the edits.
- The button will add a new record to the airport layer attribute table. And it will assign the new airport to the current region by default.
- The button allows you to copy one or more child features.
- The button will delete the selected airport permanently.
- Das Symbol öffnet einen neuen Dialog in dem Sie jeden vorhandenen ‚airport' auswählen können was dann der aktuellen ‚region' zugewiesen wird. Dies kann nützlich sein wenn Sie den ‚airport' aus Versehen in der falschen ‚region' erstellt haben.
- Das Symbol hebt die Verbindung zwischen dem ausgewählten ‚airport' und der aktuellen ‚region' wieder

auf so dass dieser auf wirksame Art und Weise nicht mehr zugewiesen ist (der Fremdschlüssel wird auf NULL gesetzt).

- With the button you can zoom the map to the selected child features.
- The two buttons and to the right switch between table view and form view where the later let's you view all the airports in their respective form.

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.

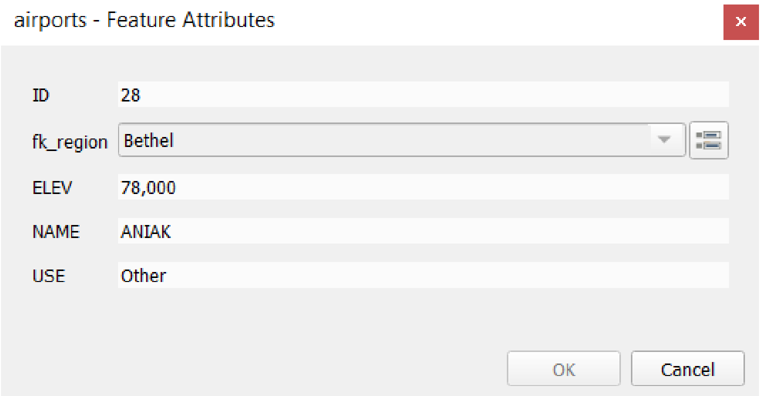


Abb. 14.79: Objekte abfragen Dialog von ‚airport' mit Beziehung zu ‚regions'

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.

Wenn Sie jetzt einen Blick auf den Objektdialog werfen werden Sie sehen dass das Formular der ‚region' in das ‚airports' Formular eingebettet ist und sogar eine Kombobox vorhanden ist mit dem der aktuelle ‚airport' zu einer anderen ‚region' zugeordnet werden kann.

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. 14.80, 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.

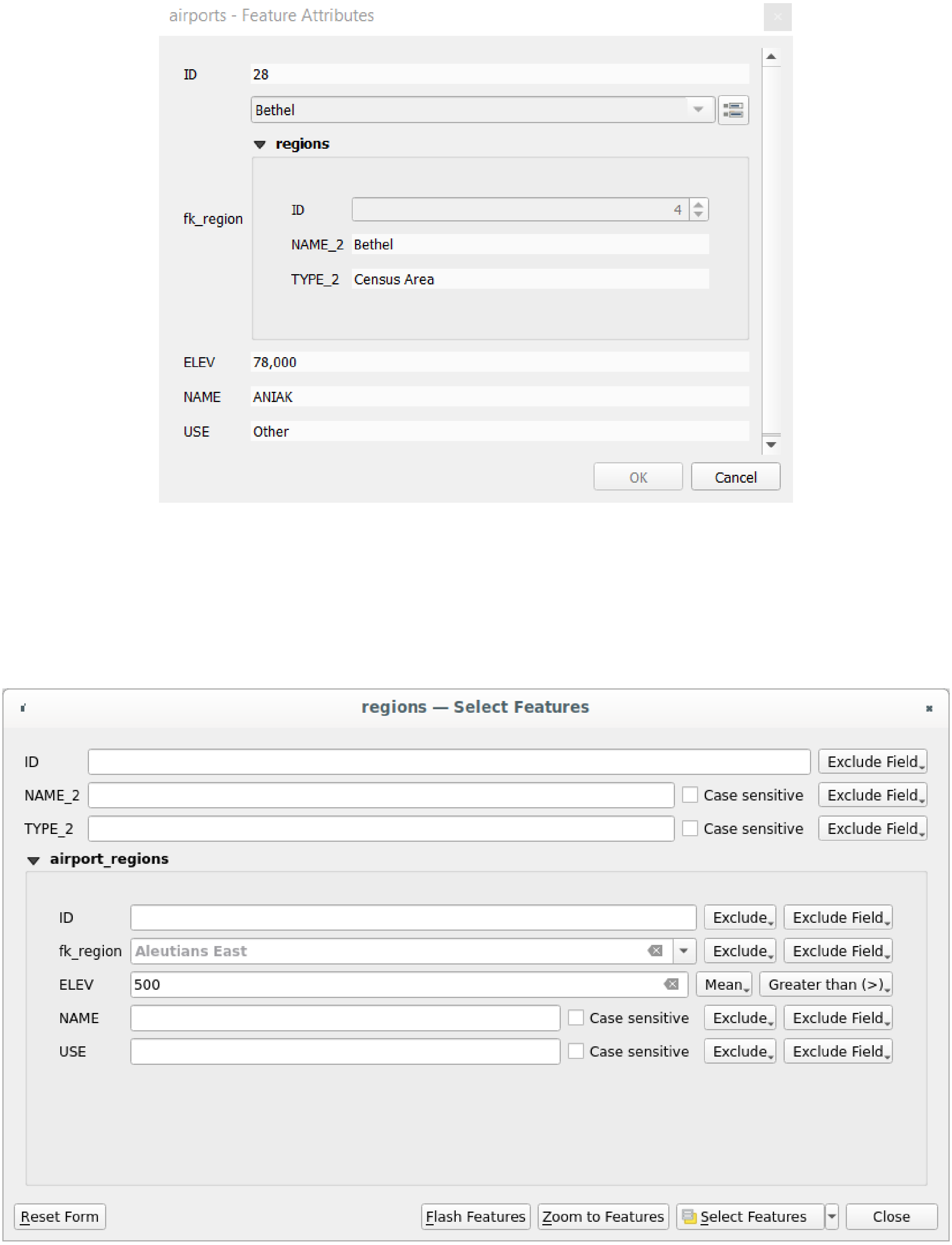


Abb. 14.80: Select parent features with child values

Introducing 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);

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.

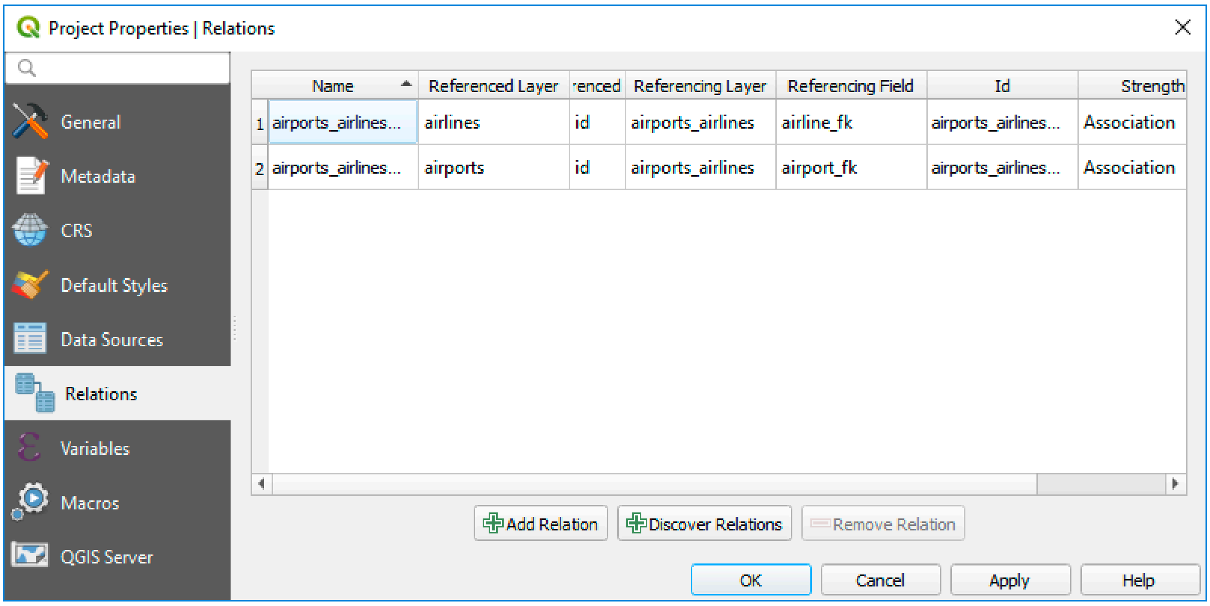


Abb. 14.81: Relations and autodiscover

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 cardinalilty 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.

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.

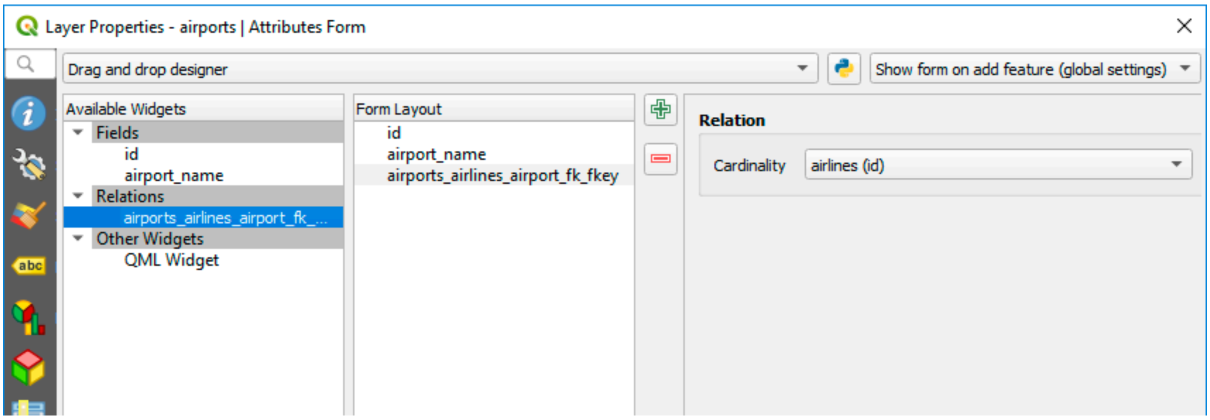


Abb. 14.82: Set relationship cardinality

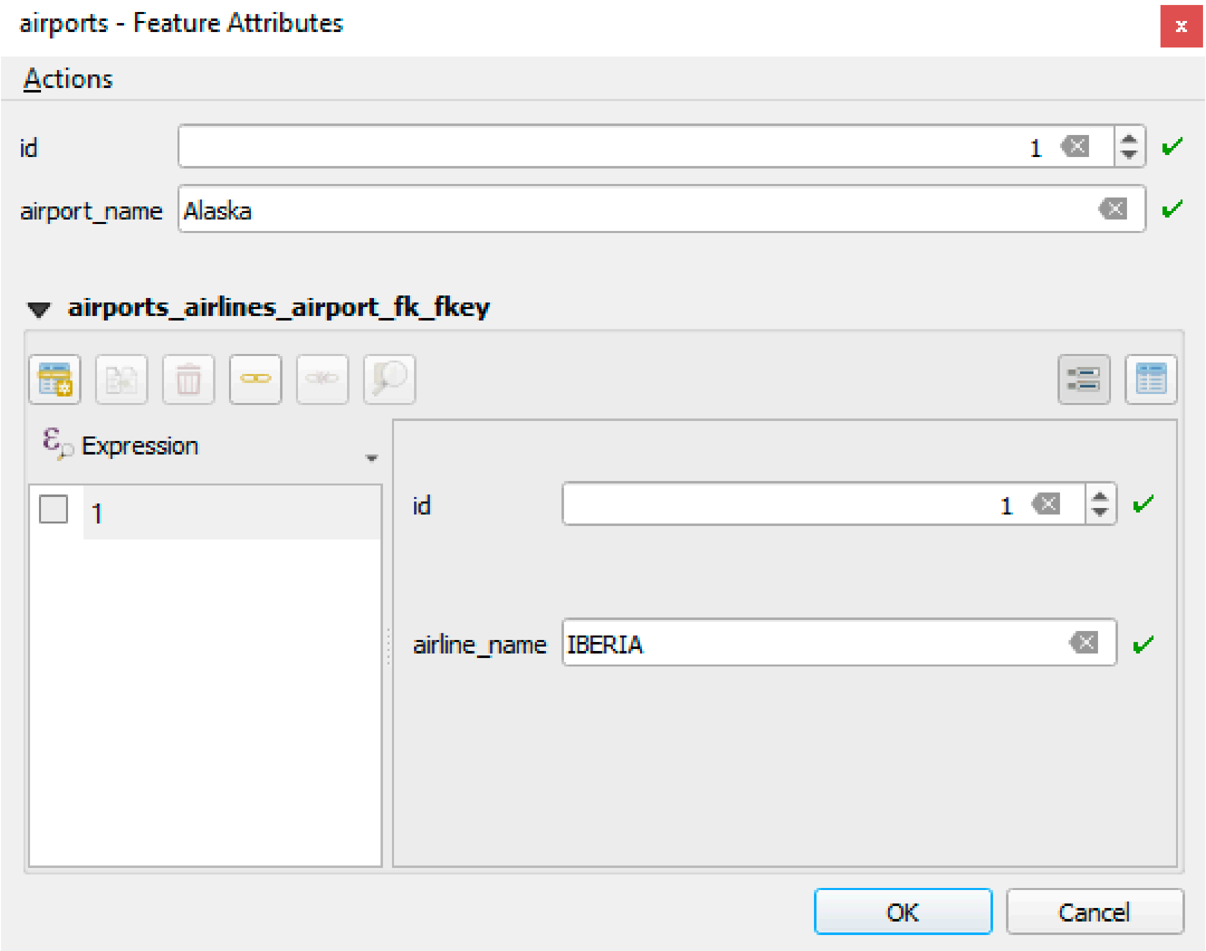


Abb. 14.83: N-M relationship between airports and airlines

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 are layers (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.

14.4 Editierfunktionen

QGIS has various capabilities for editing OGR, SpatiaLite, PostGIS, MSSQL Spatial and Oracle Spatial vector layers and tables.

Bemerkung: Die Vorgehensweise GRASS Layer zu bearbeiten ist anders - siehe Abschnitt *Digitalisieren und Editieren eines GRASS Vektorlayers* für Details.

Tipp: **Zeitgleiches Editieren**

This version of 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.

14.4.1 Einstellen der Fangtoleranz und des Suchradius

For optimal and accurate editing of vector layer geometries, we need to set an appropriate value of snapping tolerance and search radius for features vertices.

Fangtoleranz

When you add a new vertex or move an existing one, the snapping tolerance is the distance QGIS uses to search for the closest vertex or segment you are trying to connect to. If you are not within the snapping tolerance, QGIS will leave the vertex where you release the mouse button, instead of snapping it to an existing vertex or segment.

Die Fangtoleranz Einstellungen betreffen alle Werkzeuge die mit Toleranz arbeiten.

You can enable / disable snapping by using the Enable snapping button on the *Snapping Toolbar* or pressing `s`. The snapping mode, tolerance value, and units can also be configured in this toolbar.

The snapping configuration can also be set in *Project* ► *Snapping Options…*.

Sie haben drei Möglichkeiten, um die Layer, auf die eingrastet werden soll, auszuwählen:

- *All layers*: quick setting for all visible layers in the project so that the pointer snaps to all vertices and/or segments. In most cases, it is sufficient to use this snapping mode, but beware when using it for projects with many vector layers, as it may affect performance.
- *Current layer*: only the active layer is used, a convenient way to ensure topological consistency within the layer being edited.

- *Advanced Configuration*: allows you to enable and adjust snapping mode and tolerance on a layer basis (see *figure_edit_snapping*). If you need to edit a layer and snap its vertices to another, make sure that the target layer is checked and increase the snapping tolerance to a higher value. Snapping will not occur to a layer that is not checked in the snapping options dialog.

As for snapping mode, you can choose between `To vertex`, `To segment`, and `To vertex and segment`.

The tolerance values can be set either in the project's `map units` or in `pixels`. The advantage of choosing `pixels` is that it keeps the snapping constant at different map scales. 10 to 12 pixels is normally a good value, but it depends on the DPI of your screen. Using map units allows the tolerance to be related to real ground distances. For example, if you have a minimum distance between elements, this option can be useful to ensure that you don't add vertices too close to each other.

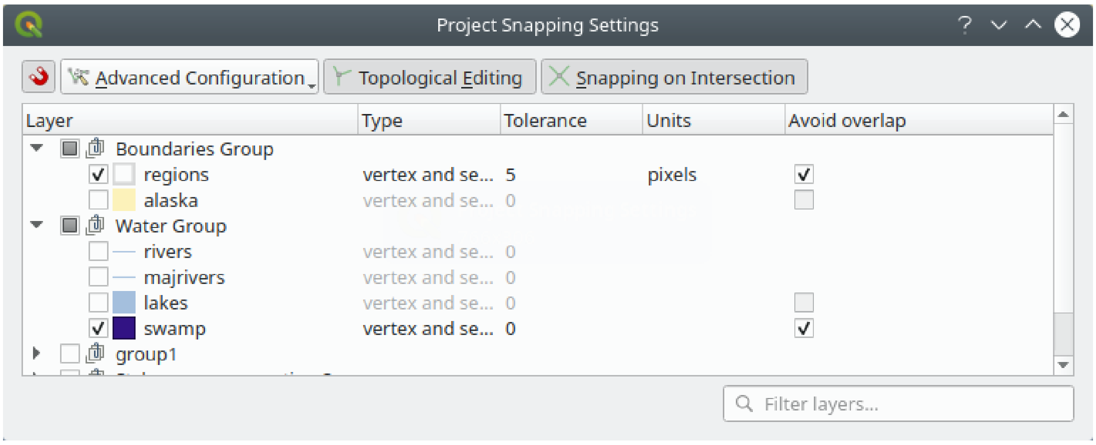


Abb. 14.84: Einrasteinstellungen (Erweiterte Konfiguration)

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 Anhaken von *Einrasten auf verborgene Objekte aktivieren* unter *Einstellungen* ► *Optionen* ► *Digitalisierung* 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 Einrasteinheiten voreinstellen, die dann im Dialog *Einrasteinstellungen* vorausgewählt sind.

Fang auf Schnittpunkte aktivieren

Ein 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.

Suchradius

Search radius for vertex edits is the distance QGIS uses to `search` for the vertex to select when you click on the map. If you are not within the search radius, QGIS will not find and select any vertex for editing. The search radius for vertex edits can be defined under the *Settings* ► *Options* ► *Digitizing* tab (this is where you define the snapping default values).

Snap tolerance and search radius are set in `map units` or `pixels`. You may need to experiment to get them right. If you specify a too big tolerance, QGIS may snap to the wrong vertex, especially if you are dealing with a large number of vertices in close proximity. The smaller the search radius, the more difficult it will be to hit what you want to move.

14.4.2 Topologisches Editieren

In addition to these snapping options, the *Snapping options…* dialog (*Project* ► *Snapping options*) and the *Snapping* toolbar allow you to enable / disable some other topological functionalities.

Topologisches Editieren ermöglichen

The Topological editing button helps when editing and maintaining features with common boundaries. With this option enabled, QGIS ‚detects' shared boundaries. When you move common vertices/segments, QGIS will also move them in the geometries of the neighboring features.

Das topologische Editieren funktioniert mit Objekten aus unterschiedlichen Layern, solange die Layer sichtbar sind und sich im Bearbeitungsmodus befinden.

Avoid overlap of new polygons

When the snapping mode is set to `Advanced configuration`, for polygon layers, there's an option called *Avoid overlap*. This option prevents you from drawing new features that overlap existing ones in the selected layer, speeding up digitizing of adjacent polygons.

With avoid overlap enabled, if you already have one polygon, you can digitize a second one such that they intersect. QGIS will then cut the second polygon to the boundary of the existing one. The advantage is that you don't have to digitize all vertices of the common boundary.

Bemerkung: Wenn die neue Geometrie vollständig durch vorhandene abgedeckt ist, wird sie gelöscht und 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.

Geometrieprüfung

Ein Kern-Plugin, dass dem Anwender hilft, Geometrieungültigkeiten zu finden. Weitere Informationen über diese Plugin erhalten Sie unter *Geometry Checker Plugin*.

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 Digitalisierprozess beschleunigen, weil sie nicht mehr jeden Stützpunkt manuell setzen müssen:

1. Enable the Tracing tool (in the *Snapping* toolbar) by pushing the icon or pressing `T` key.
2. *Snap to* 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 überlicherweise 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, plaziert QGIS alle Zwischenstützpunkte entlang des dargestellten Weges.

Unfold the Enable Tracing icon and set the *Offset* option to digitize a path parallel to the features instead of tracing along them. A positive value shifts the new drawing to the left side of the tracing direction and a negative value does the opposite.

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

By pressing the `T` key, tracing can be enabled/disabled anytime (even while digitizing a feature), so it is possible to digitize parts of the feature with tracing enabled and other parts with tracing disabled. Tools behave as usual when tracing is disabled.

14.4.3 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

From the *Project ▸ Properties… ▸ Data Sources ▸ Layers Capabilities* table, you can choose to set any layer read-only regardless the provider permission. This can be a handy way, in a multi-users environment to avoid unauthorized users to mistakenly edit layers (e.g., Shapefile), hence potentially corrupt data. Note that this setting only applies inside the current project.

In general, tools for editing vector layers are divided into a digitizing and an advanced digitizing toolbar, described in section *Erweiterte Digitalisierung*. You can select and unselect both under *View ▸ Toolbars ▸*.

Using the basic digitizing tools, you can perform the following functions:

Icon	Funktion	Icon	Funktion
	Aktuelle Änderungen		Bearbeitungsstatus umschalten
	Layeränderungen speichern		
	Datensatz hinzufügen		Objekt hinzufügen: Punkt hinzufügen
	Objekt hinzufügen: Linie hinzufügen		Objekt hinzufügen: Polygon hinzufügen
	Knotenwerkzeug (alle Layer)		Knotenwerkzeug (aktueller Layer)
	Attribute von allen gewählten Objekten gleichzeitig ändern		
	Ausgewähltes Löschen		Ausgewählte Objekte ausschneiden
	Objekte kopieren		Objekte einfügen
	Rückgängig		Wiederholen

Tabelle Bearbeiten: Funktionen der Werkzeugleiste Digitalisierung

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.

Alle Bearbeitungssitzungen beginnen mit dem Aktivieren der Funktion Bearbeitungsstatus umschalten, die Sie im Kontextmenü eines gegebenen Layers, im Dialog der Attributtabelle, der Digitalisierungswerkzeugleiste oder dem Menü *Bearbeiten* finden.

Sobald der Layer im Bearbeitungsmodus ist, werden zusätzliche Werkzeuge verfügbar und Markierungen an den Eckpunkten alles 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.

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.

Um ein geometrieloses Objekt hinzuzufügen, klicken Sie auf die Schaltfläche Objekt hinzufügen und tragen Sie die Attribute in das Objektformular ein, das aufgeht. Um Objekte mit den räumlichen Werkzeugen hinzuzufügen, digitalisieren Sie zunächst die Geometrie und tragen dann die ihre Attribute ein. Das Digitalisieren einer Geometrie geschieht folgendermassen:

1. Klicken Sie mit links in den Kartenbereich, um einen ersten Punkt Ihres neuen Objekts zu erzeugen. Bei Punktobjekten genügt das schon und sollte, falls erforderlich, das Objektformular öffnen, in das Sie die Attribute eintragen. Falls Sie die *Geometriegenauigkeit* in den Layereigenschaften gesetzt haben, können Sie *am Gitter einrasten* benutzen, um Objekte in regelmässigem Abstand zu erzeugen.
2. Für Linien und Polygone klicken Sie für jeden weiteren Knotenpunkt wieder die linke Maustaste oder benutzen Sie *automatic tracing*, um die Digitlisierung zu beschleunigen. Damit erzeugen Sie aufeinanderfolgende gerade Linien zwischen den Knotenpunkten, die Sie platzieren.

Bemerkung: Wenn Sie die `Entfernen` oder `Backspace` Taste drücken, entfernt das den letzthinzugefügten Knoten.

3. Wenn Sie alle Punkte digitalisiert haben, klicken Sie mit der rechten Maustaste auf eine beliebige Stelle, um die Digitalisierung der Geometrie dieses Objektes abzuschließen.

Bemerkung: Während Sie eine Linie oder ein Polygon digitalisieren, können Sie zwischen den Werkzeugen *Linienobjekt hinzufügen* und *circular string tools* hin- und herschalten, um zusammengesetzte Geometrien mit Bögen zu erzeugen.

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*.

4. Das Attributfenster wird erscheinen in dem Sie Informationen für das neue Objekt eingeben können. *Figure_edit_2* zeigt die Eingabe von Attributen für einen fiktiven neuen Fluss in Alaska. In dem Menü *Digitalisierung* unter *Einstellungen* ▸ *Optionen* können Sie auch folgende aktivieren:
   - *Attributeingabeformular bei der Objekterstellung unterdrücken* um das Öffnen des Formulares zu vermeiden
   - oder *Letzte Attributwerteingaben wiederverwenden* um Felder beim Öffnen automatisch gefüllt zu haben und nur ändernde Werte zu korrigieren.

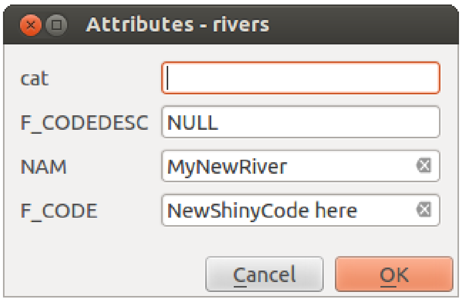


Abb. 14.85: Der Attributwertedialog nach dem Digitalisieren eines neuen Vektorobjekts

Knotenwerkzeug

Bemerkung: Tiefgreifende Änderungen in QGIS 3

In QGIS 3, the node tool has been fully redesigned and renamed to *vertex tool*. It was previously working with „click and drag" ergonomy, and now uses a „click - click" workflow. This allows major improvements like taking profit of the advanced digitizing panel with the vertex tool while digitizing or editing objects of multiple layers at the same time.

For any editable vector layer, the Vertex tool (Current Layer) provides manipulation capabilities of feature vertices similar to CAD programs. It is possible to simply select multiple vertices at once and to move, add or delete them altogether. The vertex tool also supports the topological editing feature. This tool is selection persistent, so when some operation is done, selection stays active for this feature and tool.

Es ist wichtig die Eigenschaften unter menuselection:*Einstellungen* –> *Optionen* ⍰ *Digitalisierung* ⍰ *Suchradius:* 1,00 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: Stützpunktmarken

Die aktuelle Version von QGIS unterstützt drei Arten von Stützpunktmarkern:'Teiltransparenter Kreis', ‚Kreuz' und ‚Keine'. Um den Markierungsstil zu ändern wählen Sie *Optionen* aus dem *Einstellungen* Menü, klicken Sie auf das *Digitalisierung* Menü und wählen Sie den entsprechenden Eintrag.

Eine einfache Übung

Aktivieren Sie das Knotenwerkzeug (aktueller Layer). Wenn Sie die Maus über ein Objekt bewegen, werden rote Kreise sichtbar.

- **Stützpunkte auswählen**: Sie können Stützpunkte auswählen indem Sie nacheinander darauf klicken während Sie die `Umschalt`-Taste gedrückt halten oder indem Sie ein Rechteck um einige Stützpunkte druch klicken und ziehen erzeugen. Wenn ein Stützpunkt ausgewählt ist, ändert sich seine Farbe zu blau. Um der aktuellen Auswahl weitere Stützpunkte hinzuzufügen halten Sie die `Umschalt`-Taste beim Klicken gedrückt. Um Stützpunkte aus der Auswahl zu entfernen, halten Sie `Strg` gedrückt.
- **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.

  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.

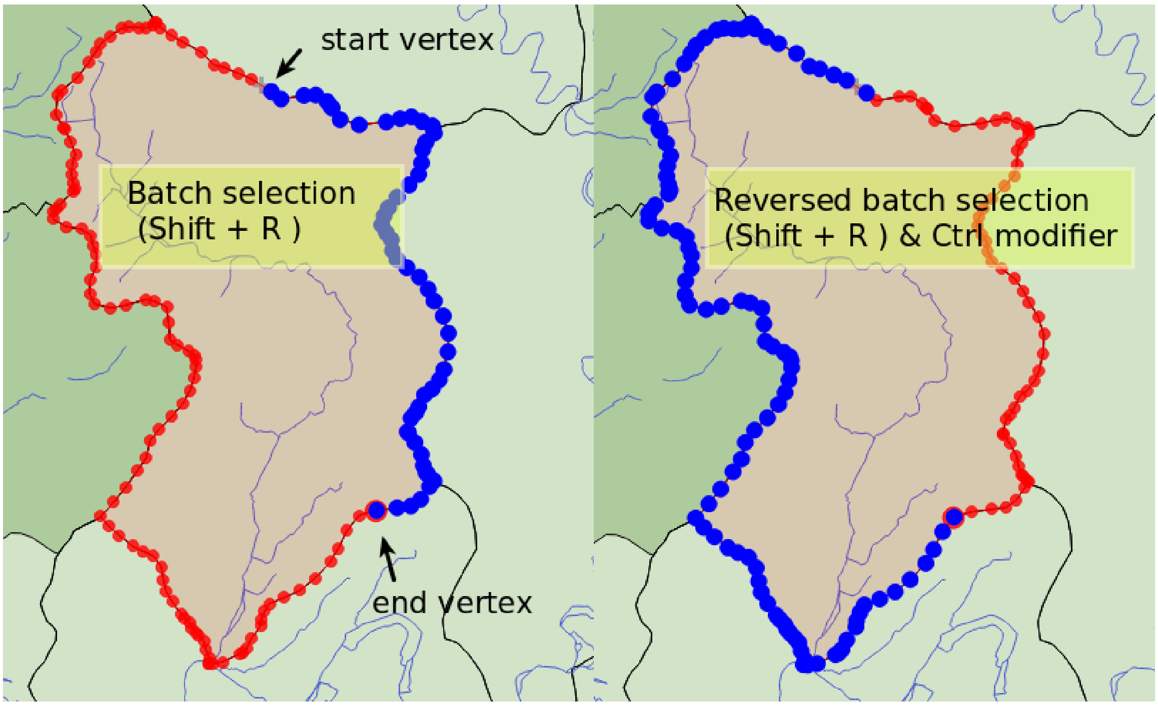


Abb. 14.86: Stützpunkte im Stapelbetrieb mit `Umschalt+R` auswählen

- **Adding vertices**: To add a vertex, a virtual new node appears on the segment center. Simply grab it to add a new vertex. A double-click on any location of the boundary also creates a new node. For lines, a virtual node is also proposed at both extremities of a line to extend it.

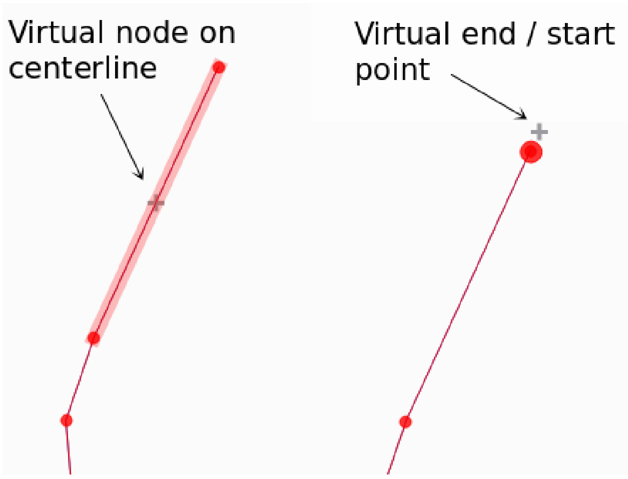


Abb. 14.87: Virtuelle Knoten um Stützpunkte hinzuzufügen

- **Deleting vertices**: Select the vertices and click the `Delete` key. Deleting all the vertices of a feature generates, if compatible with the datasource, a geometryless feature. Note that this doesn't delete the complete feature, just the geometry part. To delete a complete feature use the Delete Selected tool.
- **Stützpunkte verschieben**: Wählen Sie alle Stützpunkte aus, die verschoben werden sollen und klicken Sie auf einen Stützpunkt oder eine Kante; klicken Sie danach auf die gewünschte Zielposition. Alle ausgewählten Stützpunkte werden zusammen verschoben. Falls Einrasten eingeschaltet ist, kann die ganze Auswahl entsprechend springen. Sie können das Bedienfeld Erweiterte Digitalisierung benutzen, um Abstand, Winkel oder exakte Koordinaten einzugeben, bevor Sie zum zweiten mal klicken.

  Hier können Sie die Funktion am Gitter einrasten benutzen. Wenn Sie einen Wert für die *Geometriegenauigkeit* in den Layereigenschaften gesetzt haben, erscheint ein Gitter ab einem der Geometriegenauigkeit entsprechenden Massstab.

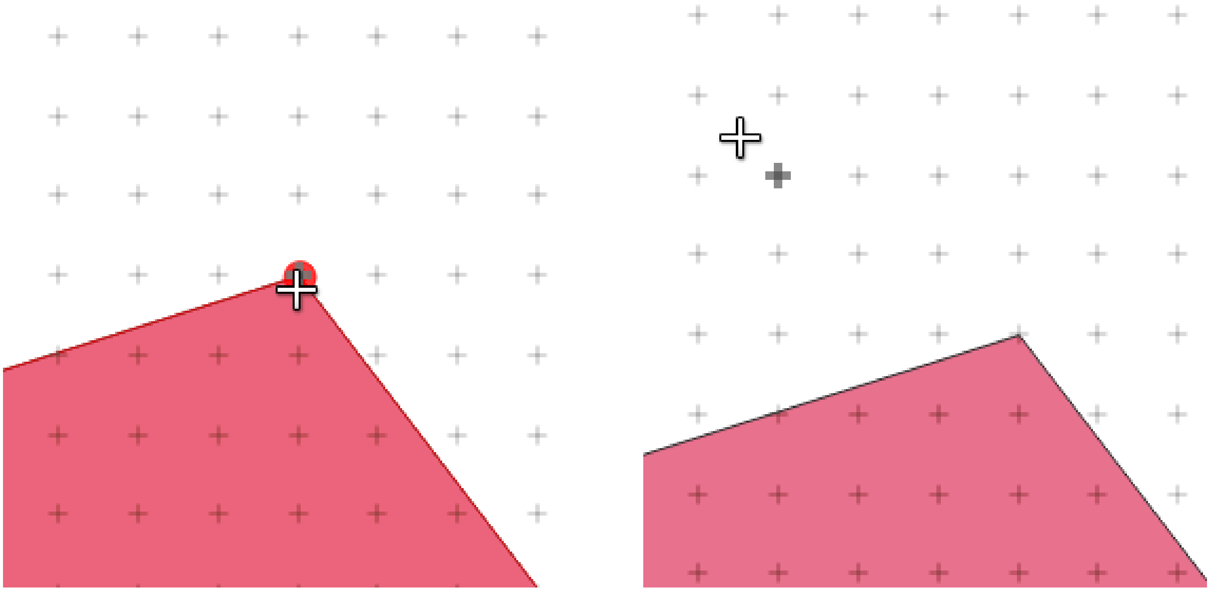


Abb. 14.88: Einen Stützpunkt auswählen und auf das Gitter verschieben

Each change made with the vertex is stored as a separate entry in the *Undo* dialog. Remember that all operations support topological editing when this is turned on. On-the-fly projection is also supported, and the vertex tool provides tooltips to identify a vertex by hovering the pointer over it.

Das Bedienfeld Stützpunktbearbeitung

Während Sie das *Knotenwerkzeug* für ein Objekt benutzen, können Sie mit einem Rechtsklick das Bedienfeld *Stützpunktbearbeitung* öffnen, das alle Stützpunkte des Objektes mit ihren *x*, *y* (*z*, *m* falls zutreffend) Koordinanten und *r* (für den Radius bei einer bogenförmigen Geometrie) auflistet. Durch Anwählen einer Zeile in der Tabelle wird der entsprechende Stützpunkt im Kartenfenster ausgewählt und umgekehrt. Verändern Sie einfach eine Koordinate in der Tabelle, um die Lage des Stützpunkts zu verändern. Sie können auch mehrere Zeilen auswählen und sie zusammen löschen.

Bemerkung: In QGIS 3.4 geändertes Verhalten

Wenn Sie mit der rechten Maustaste auf ein Objekt klicken, wird sofort das Bedienfeld Stützpunktbearbeitung angezeigt und das Objekt selbst gesperrt, wodurch Sie andere Objekte nicht mehr bearbeiten können. Während es gesperrt ist, kann ausschließlich das gesperrte Objekt bearbeitet werden: auswählen und verschieben von Stützpunkten und Segmenten durch klicken und ziehen ist nur für dieses Objekt möglich. Neue Stützpunkte können Sie nur für das gesperrte Objekt hinzufügen. Darüberhinaus öffnet sich das Bedienfeld Stützpunktbearbeitung jetzt automatisch selbst, wenn Sie das Knotenwerkzeug aktivieren und seine Position bzw. sein Andockstatus bleiben erhalten.

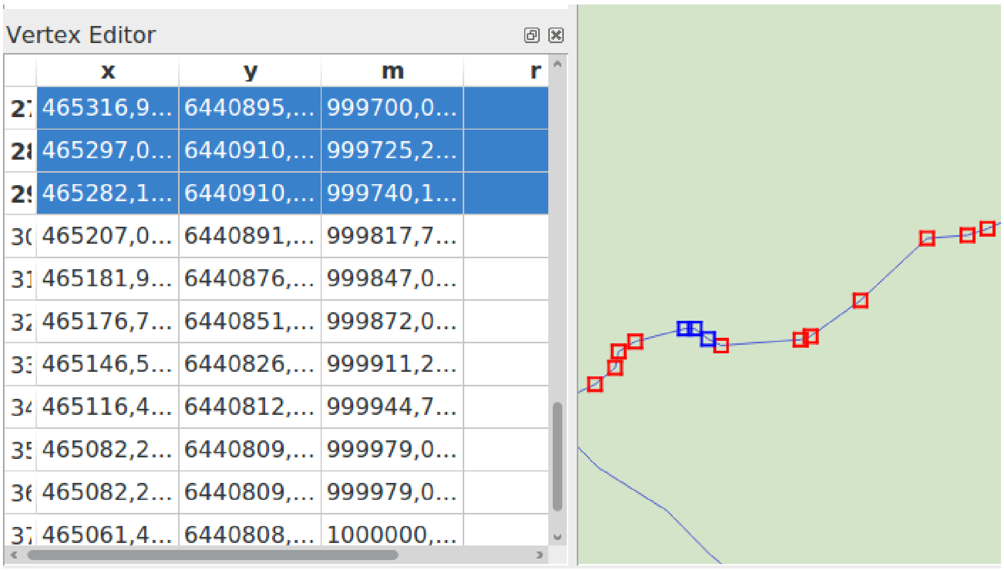


Abb. 14.89: Das Bedienfeld Stützpunktbearbeitung mit ausgewählten Knoten

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 Quellayer. 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

Wenn Sie eine neue Spalte in die Attributtabelle mit dem Typ ‚string' erstellt haben und die Werte von einer anderen Attributspalte einfügen, die eine größere Länge als die Länge der Spaltengröße hat, wird diese auf den gleichen Betrag verlängert werden. Dies ist so, weil der GDAL Shapedatei Driver beginnend ab GDAL/OGR 1.10 String und Integer Fehler auto-erkennt und die Felder auf die benötigte Länge der eingefügten Daten anpasst.

Ausgewählte Objekte löschen

If we want to delete an entire feature (attribute and geometry), we can do that by first selecting the geometry using the regular Select Features by area or single click tool. Selection can also be done from the attribute table. Once you have the selection set, press `Delete` or `Backspace` key or use the Delete Selected tool to delete the features. Multiple selected features can be deleted at once.

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 der Rücknahme-/Wiederholungsverlauf zeigt (siehe *Figure_edit_undo*). Dieses Bedienfeld wird nicht standardmäßig angezeigt; es kann angezeigt werden, indem man mit rechts auf die Werkzeugleiste klickt und das Rücknahme/Wiederholungs-Kontrollkästchen aktiviert. Rücknahme/Wiederholung ist auch dann aktiv wenn das Bedienfeld nicht angezeigt wird.

Wenn Rückgängig oder `Strg+z` (oder:kbd:*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.

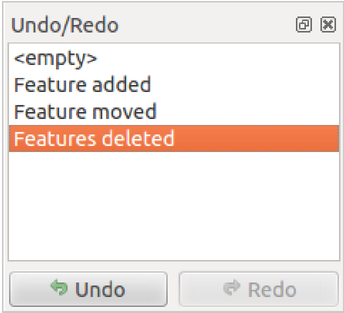


Abb. 14.90: Rückgängig und Wiederholen von Digitalisierschritten

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 aussschalten wollen werden Sie ebenfalls gefragrt 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 zugängig.

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).

14.4.4 Erweiterte Digitalisierung

Icon	Funktion	Icon	Funktion
	Erweiterte Digitalsierungswerkzeuge einschalten		Spurverfolgung einschalten
	Objekt(e) verschieben		Objekt(e) kopieren und verschieben
	Objekt(e) drehen		Objekt vereinfachen
	Ring hinzufügen		Teil hinzufügen
	Ring füllen		Swap direction
	Ring löschen		Teil löschen
	Linie versetzen		Objekte überarbeiten
	Teile zerlegen		Objekte trennen
	Attribute gewählter Objekte vereinen		Gewählte Objekte verschmelzen
	Punktsymbole drehen		Punktsymbolversatz
	Trim or Extend Feature		

Tabelle Erweiterte Digitalisierung: Werkzeugleiste Erweiterte Digitalisierung für Vektorlayer

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, bennutzen 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* ° 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 `Shift` vor dem Klick auf die Karte halten, wird die Drehung in 45 Grad Schritten erfolgen, was anschließend in dem Benutzerinterface-Widget bearbeitet werden kann.

Zum Abbrechen der Objektdrehung, drücken Sie die Taste `ESC` oder klicken Sie auf die Schaltfläche Objekt(e) drehen.

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 geometrielosen Objekt hinzuzufügen. Zuerst wählen Sie das Objekt in der Attributtabelle, dann digitalisieren Sie eine neue Geometrie mit dem Werkzeug Teil hinzufügen.

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 Ring füllen.
2. Zeichnen Sie ein Polygon über das vorhandene Objekt: QGIS fügt der Geometrie ein neus Loch hinzu (wie wenn Sie das Werkzeug Ring hinzufügen benutzt hätten) und erzeugt ein neues Objekt, dessen Geometrie dem Loch entspricht (wie wenn Sie *traced* über die innere Grenze mit dem Werkzeug Polygonobjekt hinzufügen angewendet hätten).

3. Falls das Loch im Objekt bereits exisitert, 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 letzen Schnittpunkt mit der ursprünglichen Linie.

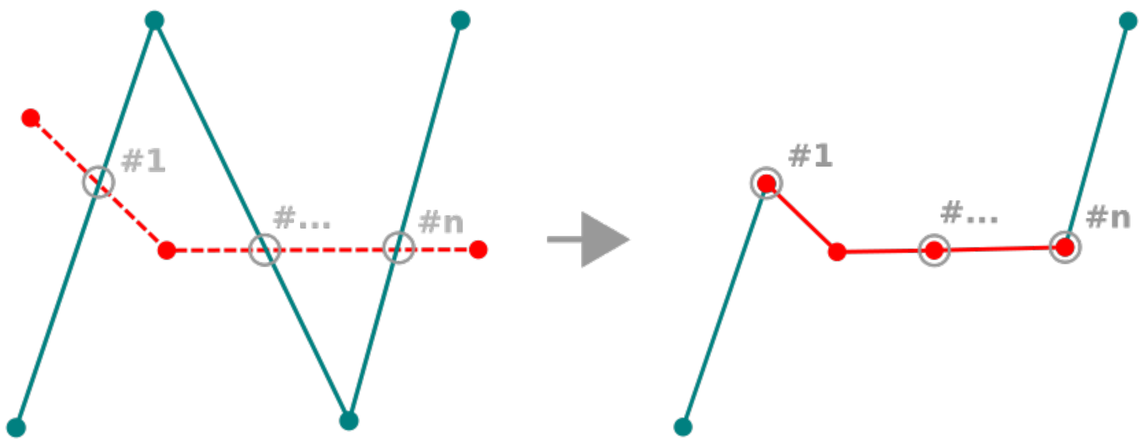


Abb. 14.91: Eine Linie umformen

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 letzen 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.

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 Umformline mehrere Ringe eines Polygons kreuzt, weil das zu einem ungültigen Polygon führen würde.

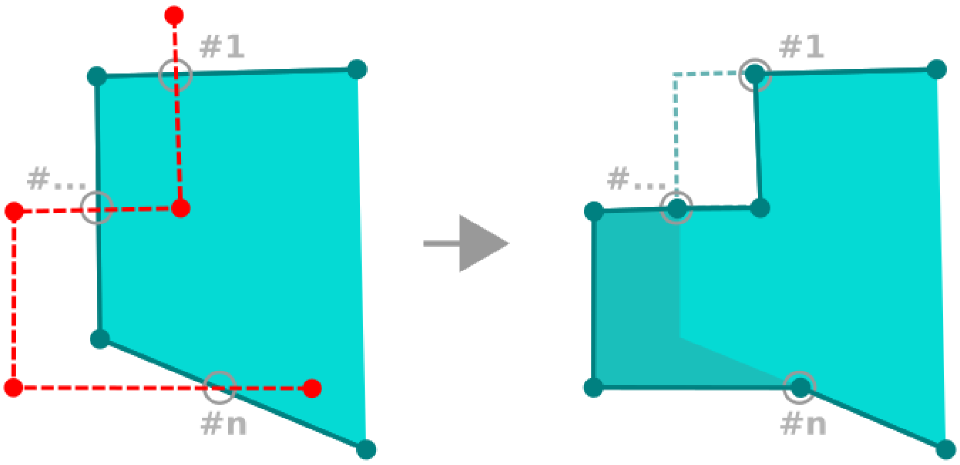


Abb. 14.92: Ein Polygon umformen

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 einen Versatz eines Linienlayers zu erstellen müssen Sie erst in den Bearbeitungsmodus gehen und das Werkzeug Linie versetzen aktivieren. Klicken Sie dann auf ein Objekt, um es zu verschieben. Bewegen Sie die Maus und klicken Sie, wo Sie es haben wollen oder geben Sie die gewünschte Distanz in der Benutzereingabe ein. Ihre Änderungen können dann mit dem Werkzeug Layeränderungen speichern gespeichert werden.

Der QGIS Optionen Dialog (Digitalisierung Reiter dann **Werkzeug zum Linien versetzen** Abschnitt) ermöglicht es Ihnen einige Parameter wie **Verbindungsstil**,**Quadrantsegmente**, **Eckengrenze** zu konfigurieren.

Reverse Line

Changing the direction of a line geometry can be useful for cartographical purposes or when preparing for network analysis.

To change a line direction:

1. Activate the reverse line tool by clicking Reverse line.
2. Click on the line. The direction of the line is reversed.

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. Zeichnen Sie eine Linie über das Objekt oder die Objekte, die Sie zerteilen wollen. Gibt es eine aktive Auswahl, werden nur die ausgewählten Objekte zerteilt. Falls, *default values or clauses* definiert wurden, werden sie auf die entsprechenden Felder angewendet, andere Attribute des Elternobjekts werden automatisch in die neuen Objekte kopiert.
3. Danach können Sie üblicherweise beliebige Attribute der neuen Objekte verändern.

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 zelegen, 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;
   - auf die Schaltfläche *Alle Felder überspringen* klicken, um leere Attribute zu benutzen;
   - oder durch öffnen des Aufklappmenüs oben in der Tabelle haben Sie die Möglichkeit jede der genannten Optionen nur auf das jeweilige Feld anzuwenden. Sie können auch eine Aggregatfunktion auf die ursprünglichen Objektattribute anwenden (Minimum, Maximum, Median, Summe, Anzahl, Verknüpfung… abhängig vom Feldtyp, siehe ref:*statistical_summary* für eine Auflistung aller Funktionen).

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. Zunächst müssen Sie das Symbol eine a *data-defined* Drehung einstellen:
   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.

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.

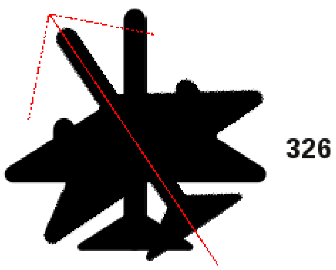


Abb. 14.93: Ein Punktsymbol drehen

2. Klicken Sie mit dem Werkzeug Punktsymbole drehen auf ein Punktobjekt im Kartenfenster und bewegen Sie den Mauszeiger, während Sie die linke Maustaste gedrückt halten. Ein roter Pfeil mit dem Drehwinkel wird angezeigt (siehe *Figure_rotate_point*).

3. Sobald Sie die linke Maustaste loslassen, wird das Symbol mit diesem Drehwinkel angezeigt und das bei der Drehung eingestellte Feld in der Attributtabelle wird entsprechend aktualisiert.

Tipp: Wenn Sie zusätzlich die `Strg`-Taste gedrückt halten, findet die Drehung in 15 Grad Schritten statt.

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 mit den Versatzkoordinaten gefüllt, während Sie das Symbol im Kartenfenster hin- und herschieben.

Bemerkung: Das Werkzeug Punktsymbolversatz verschiebt nicht das Punktobjekt selbst; dafür sollten Sie das Werkzeug Knotenwerkzeug (aktueller Layer) oder das Werkzeug Objekt verschieben benutzen.

Trim/Extend Feature

When a digitized line is too short or too long to snap to another line (missing or crossing the line), it is necessary to be able to extend or shorten the segment.

The Trim/Extend tool allows you to also modify (multi)lines AND (multi)polygons. Moreover, it is not necessarily the end of the lines that is concerned; any segment of a geometry can be modified.

Bemerkung: This can lead to invalid geometries.

Bemerkung: You must activate segment snapping for this tool to work.

The tool asks you to select a limit (a segment) with respect to which another segment will be extended or trimmed. Unlike the vertex tool, a check is performed to modify only the layer being edited.

When both segments are in 3D, the tool performs an interpolation on the limit segment to get the Z value.

In the case of a trim, you must select the part that will be shortened by clicking on it.

14.4.5 Formen digitalisieren

Die *Werzeugleiste für Formen* bietet Ihnen Werkzeuge, um regelmäßige Formen und Geometrien mit Bögen zu zeichnen.

Kreisbogen hinzufügen

Mit den Werkzeugen Kreisbogen hinzufügen oder Kreisbogen über Radius hinzufügen können Sie Linien- oder Polygonobjekte mit bogenförmiger Geometrie hinzufügen.

Das Zeichnen von Objekten mit diesen Werkzeugen folgt der selben Regel wie bei anderen Digitalisierwerkzeugen: klicken Sie mit links, um Stützpunkte hinzuzufügen und mit rechts, um die Geometrie abzuschließen. Während Sie die Geometrie zeichnen, können Sie die Werkzeuge wechseln und auch zu den *linear geometry tools* wechseln, um zusammengesetzte Geometrien zu erzeugen.

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.

Draw Circles

There is a set of tools for drawing circles. The tools are described below.

Circles are converted into circular strings. Therefore, as explained in *Kreisbogen hinzufügen*, if allowed by the data provider, it will be saved as a curved geometry, if not, QGIS will segmentize the circular arcs.

- Add circle from 2 points: The two points define the diameter and the orientation of the circle. (Left-click, right-click)
- Add circle from 3 points: Draws a circle from three known points on the circle. (Left-click, left-click, right-click)
- Add circle from center and a 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.
- Add circle from 3 tangents: Draws a circle that is tangential to three segments. **Note that you must activate snapping to segments** (See *Einstellen der Fangtoleranz und des Suchradius*). Click on a segment to add a tangent. If two tangents are parallel, an error message appears and the input is cleared. (Left-click, left-click, right-click)
- Add 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.

Draw Ellipses

There is a set of tools for drawing ellipses. The tools are described below.

Ellipses cannot be converted as circular strings, so they will always be segmented.

- Add Ellipse from center and two points: Draws an ellipse with a given center, major axis and minor axis. (Left-click, left-click, right-click)
- Add Ellipse from center and a point: Draws an ellipse into a bounding box with the center and a corner. (Left-click, right-click)
- Add Ellipse from extent: Draws an ellipse into a bounding box with two opposite corners. (Left-click, right-click)
- Add Ellipse from foci: Draws an ellipse by 2 points for foci and a point on the ellipse. (Left-click, left-click, right-click)

Draw Rectangles

There is a set of tools for drawing rectangles. The tools are described below.

- Rectangle from center and a point: Draws a rectangle from the center and a corner. (Left-click, right-click)
- Rectangle from extent: Draws a rectangle from two opposite corners. (Left-click, right-click)
- Rectangle from 3 points (distance): Draws an oriented rectangle from three points. The first and second points determine the length and angle of the first edge. The third point determines the length of the other edge. (Left-click, left-click, right-click)
- Rectangle from 3 points (projected): Same as the preceding tool, but the length of the second edge is computed from the projection of the third point on the first edge. (Left-click, left-click, right-click)

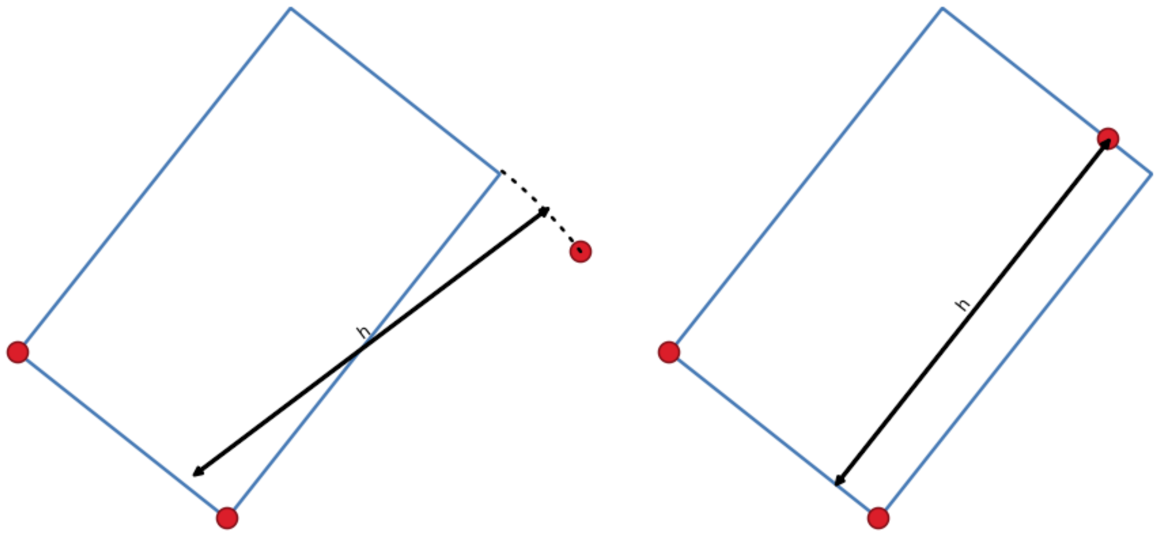


Abb. 14.94: Draw rectangle from 3 points using distance (right) and projected (left)

Draw Regular Polygons

There is a set of tools for drawing regular polygons. The tools are described below. Left-click to place the first point. A dialog appears, where you can set the number of polygon edges. Right-click to finish the regular polygon.

- Regular polygon from two points: Draws a regular polygon where the two points determine the length and angle of the first edge.
- Regular polygon from center and a point: Draws a regular polygon from the provided center point. The second point determines the angle and distance to the middle of an edge.
- Regular polygon from center and a corner: Same as the preceding tool, but the second point determines the angle and distante to a vertex.

14.4.6 Das Bedienfeld Erweiterte Digitalisierung

Wenn Sie neue oder vorhandene Geometrien erfassen, umformen, oder zerteilen, haben Sie auch die Möglichkeit, das Bedienfeld Erweiterte Digitalisierung zu verwenden. Sie können Linien exakt parallel oder rechtwinklig oder in einem bestimmten Winkel digitalisieren oder Linien auf bestimmten Winkeln sperren. Darüber hinaus können Sie Koordinaten direkt eingeben, so dass Sie eine neue Geometrie genau festlegen können.

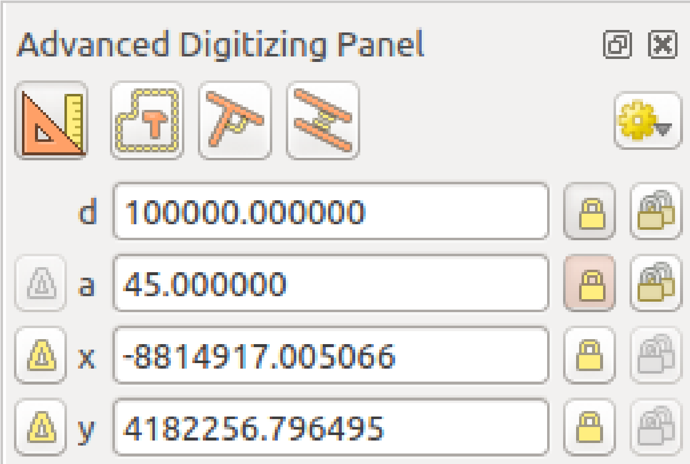


Abb. 14.95: Das Bedienfeld Erweiterte Digitalisierung

Das *Bedienfeld Erweiterete Digitalisierung* können Sie mit einem Rechtsklick in die Werkzeugleiste, über das Menü *Ansicht ▯ Bedienfelder▯* oder die Tastenkombination `Strg+4` öffnen. Sobald das Bedienfeld sichtbar ist, können Sie die Werkzeuge selbst über die Schaltfläche Erweiterte Digitalisierungswerkzeuge einschalten aktivieren.

Bemerkung: Die Werkzeuge sind nicht aktiviert, wenn die Kartenansicht geographische Koordinaten zeigt.

Die Idee dahinter

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.

Einrasteinstellungen

Klicken Sie auf die Schaltfläche , um die Einrasteinstellungen der Werkzeuge der Erweiterten Digitalisierung festzulegen. Sie können festlegen, dass das Werkzeug nur auf gebräuchliche Winkel einrastet. Es gibt folgende Optionen:

- *Nicht auf gemeinsame Winkel einrasten* muss heißen: „gebräuchliche" Winkel
- *Auf 30º Winkel einrasten*
- *Auf 45º Winkel einrasten*
- *Auf 90º Winkel einrasten*

Sie können auch das Einrasten auf Objekte steuern. Hier gibt es folgende Möglichkeiten:

- *Kein Einrasten auf Stützpunkten oder Segementen*
- *Einrasten nach Projektkonfiguration*

- *Auf allen Layern einrasten*

Tastenkürzel

Um die Arbeit mit dem Bedienfeld Erweiterte Digitalisierung zu beschleunigen, gibt es eine Reihe von Tastenkürzeln:

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
`C`	Konstruktionsmodus an-/ausschalten		
`P`	Zwischen senkrechtem und parallelem Modus umschalten		

Mit absoluter Referenz digitalsieren

Sobald Sie eine neue Geometrie von Grund auf neu zeichnen, kann es hilfreich sein, den ersten Stützpunkt an festgelegten Koordinaten zu setzen.

Um beispielsweise ein neues Objekt in einem Polygonlayer zu erzeugen, klicken Sie auf die Schaltfläche . Sie können sich für eine die X- und Y-Koordinate entscheiden, an der Sie mit dem Zeichnen des Objektes beginnen wollen, dann:

- klicken Sie auf das Eingabefeld *x* (oder benutzen das Tastenkürzel `X`),
- 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,
- klicken Sie auf das Eingabefeld *y* (oder benutzen das Tastenkürzel `Y`),
- 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.

Zwei blaugepunktete Linien und ein grünes Kreuz zeigen genau die Koordinaten an, die Sie eingegeben haben. Beginnen Sie den Digitalisiervorgang, indem Sie in das Kartenfenster klicken; die Position des Mauszeigers wird am grünen Kreuz beibehalten.

Sie können nun entweder freihand weiter digitalisieren oder ein neues Koordinatenpaar eingeben oder Sie können die **Länge** (den Abstand) und den **Winkel** des Segments eingeben.

Falls Sie ein Segment mit einer bestimmten Länge zeichnen wollen, klicken Sie das Eingabefeld *d (Abstand)* an (Tastenkürzel `D`), geben Sie den Abstandswert (in Karteneinheiten) ein und drücken Sie `Eingabe` oder klicken Sie auf die Schaltfläche rechts davon, um die Mausposition im Kartenfenster mit diesem Abstand beizubehalten. Im Kartenfenster wird um den angeklickten Punkt ein Kreis angezeigt, dessen Radius dem Wert entspricht, den Sie als Abstand eingetragen haben.

Sie können auch den Winkel des Segments festlegen. Klicken Sie, wie oben beschrieben, auf die Eingabefläche *a (Winkel)* (Tastenkürzel `A`), geben Sie den Winkel (in Grad) ein und drücken Sie `Eingabe` oder klicken Sie auf die Schaltfläche rechts davon. Dadurch folgt das Segment dem gewünschten Winkel:

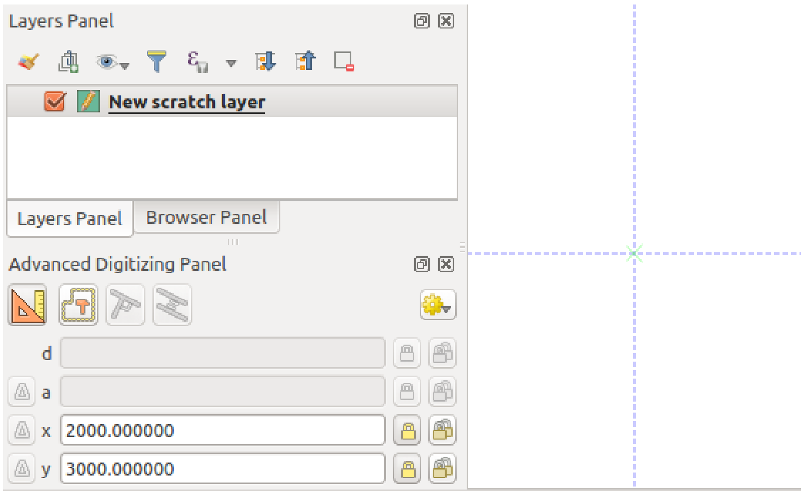


Abb. 14.96: Eine Zeichnung mit vorgegebenen Koordinaten beginnen

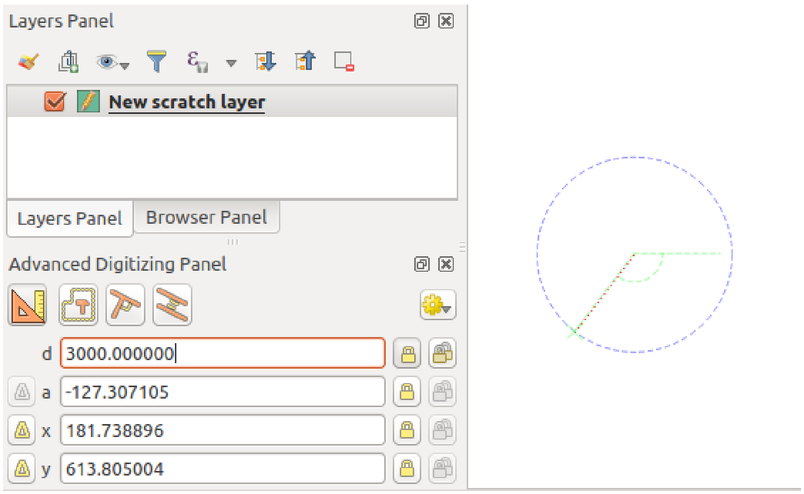


Abb. 14.97: Segment mit einer vorgegebenen Länge

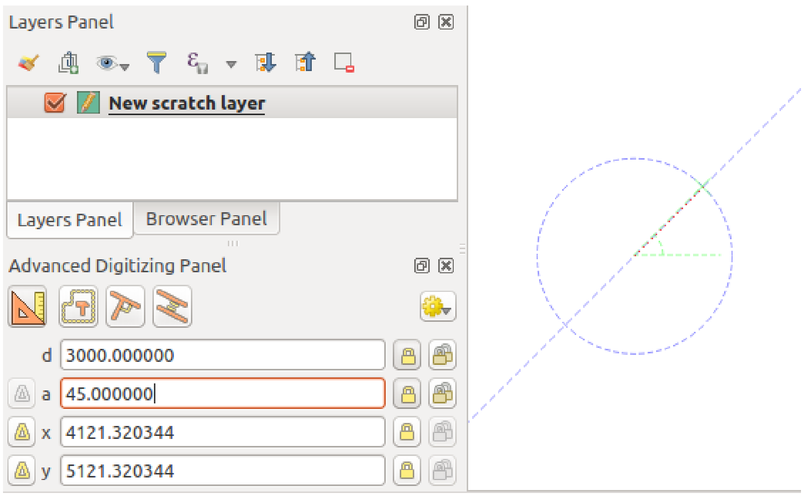


Abb. 14.98: Segment mit vorgegebenem Winkel

Mit relativer Referenz digitalsieren

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 `Shift+A`) um Winkeleingabe 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.

Zur Koordinateneingabe klicken Sie auf die Schaltfläche links der Eingabezeilen *x* oder *y* (oder drücken Sie `Shift+X` oder `Shift+Y``) um die Koordinateneingabe relativ zum zuletzt eingegebenen Stützpunkt ein- bzw. auszuschalten. Sind diese Optionen eingeschaltet, werden Koordinatenberechnungen so durchgeführt, als wäre der zuletzt eingegebenene Stützpunkt der Koordinatenurspung.

Laufendes Beibehalten

Sie können sowohl beim Digitalisieren mit absoluter als auch mit relativer Referenz Winkel, Abstand sowie die X- und Y-Festlegungen laufend beibehalten, indem Sie auf die Schaltflächen *Laufend beibehalten* klicken. Laufendes Beibehalten ermöglicht es Ihnen mehrere Punkte oder Stützpunkte mit den selben Festlegungen zu digitalisieren.

Parallele und senkrechte Linien

Alle oben beschriebenen Werkzeuge können mit den Werkzeugen Senkrecht und Parallel kombiniert werden. Diese beiden Werkzeuge ermöglichen es Ihnen, Segmente exakt senkrecht oder parallel zu einem andern Segment zu zeichnen.

Um ein *senkrechtes* Segment zu zeichnen, klicken Sie während des Editierens auf die Schaltfläche senkrecht (Tastenkürzel `P`), um das Werkzeug zu aktivieren. Bevor Sie die senkrechte Linie zeichnen, klicken Sie auf das Segment eines vorhandenen Objekts, zu dem die neue Linie senkrecht sein soll (die Linie des vorhandenen Objekts wird hellorange eingefärbt); Sie sollten eine blau gepunktete Linie an der Stelle sehen, an der Ihr Objekt gefangen wird:

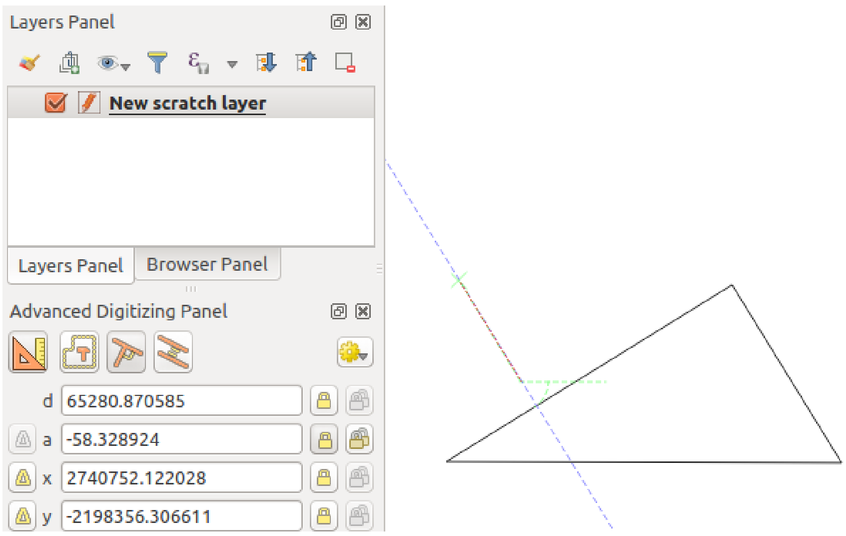


Abb. 14.99: Senkrechtes Digitalisieren

Die Schritte, um ein *paralleles* Objekt zu zeichnen sind die selben: klicken Sie auf die Schaltfläche Parallel (zweimal Tastenkürzel `P`), klicken Sie auf das Segment, das Sie als Referenz benutzen wollen und beginnen Sie damit, das Objekt zu zeichen:

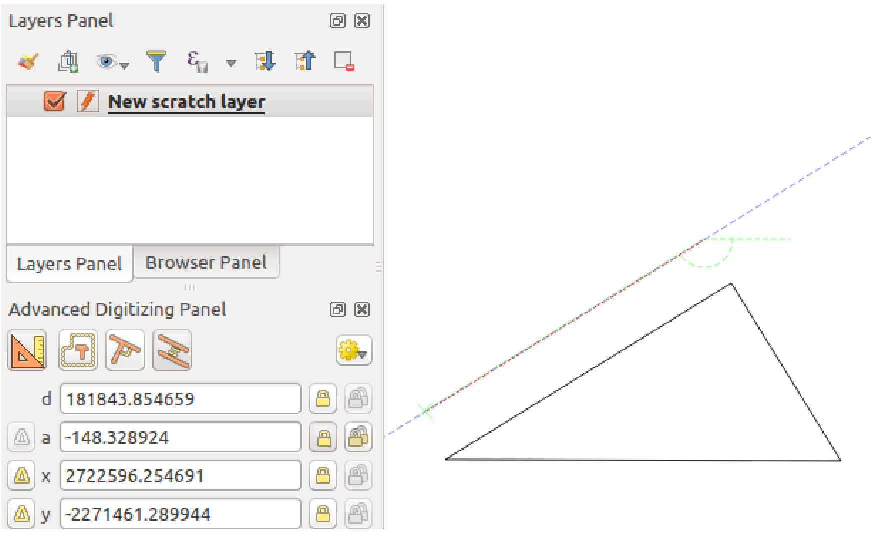


Abb. 14.100: Paralleles Digitalisieren

Diese beiden Werkzeuge finden genau den richtigen Winkel für senkrechtes und paralleles Zeichnen und behalten ihn während des Editierens bei.

Konstruktionsmodus

Sie können den Konstruktionsmodus ein- und ausschalten, indem Sie auf die Schaltfläche Konstructionsmodus klicken oder das Tastenkürzel `C` benutzen. Während der Konstruktionsmodus eingeschaltet ist, fügen Sie mit Klicks im Kartenfenster keine neuen Stützpunkte hinzu, sondern die Klickpositionen werden erfasst, so dass Sie sie als Referenzpunkte zur Festlegung von relativem Abstand, Winkel, X- und Y-Kordinaten benutzen können.

Der Konstruktionsmodus kann z.B. benutzt werden, um einen Punkt in einer genau festgelegten Entfernung von einem bestehenden Punkt zu zeichnen.

Ist ein Punkt im Kartenfenster vorhanden und der Einrastmodus richtig eingestellt, können Sie ganz einfach weitere Punkte in vorgegebenen Abständen oder Winkeln von ihm zeichnen. Zusätzlich zur Schaltfläche müssen Sie den *Konstruktionsmodus* durch Klicken auf die Schaltfläche Konstruktionsmodus oder mit dem Tastenkürzel `C` einschalten .

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:

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 die Winkeleinstellung z.B. auch dafür benutzen, einen anderen Punkt mit dem selben Abstand vom bestehenden aber in einem bestimmten Winkel zum gerade neu erzeugten Punkt zu erzeugen. Klicken Sie auf die Schaltfläche Konstruktionsmodus oder Tastenkürzel `C` um in den Konstruktionsmodus zu wechseln. Klicken Sie auf den eben erzeugten Punkt und dann auf den ursprünglich vorhandenen, um eine Richtung zu erhalten. Klicken Sie danach in das Eingabefeld *d* (Tastenkürzel `D`), geben Sie den gewünschten Abstand ein und drücken Sie `Eingabe`. Klicken Sie in das Eingabefeld *a* (Tastenkürzel `A`), geben Sie den gewünschten Winkel ein und drücken Sie `Eingabe`. Die Mausposition im Kartenfenster wird nun sowohl im Abstand als auch im Winkel beibehalten.

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.

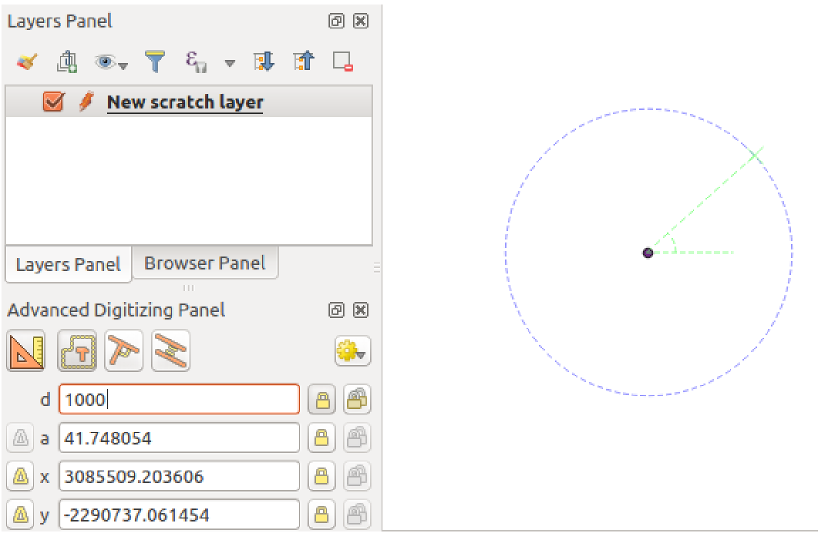


Abb. 14.101: Abstand von einem Punkt

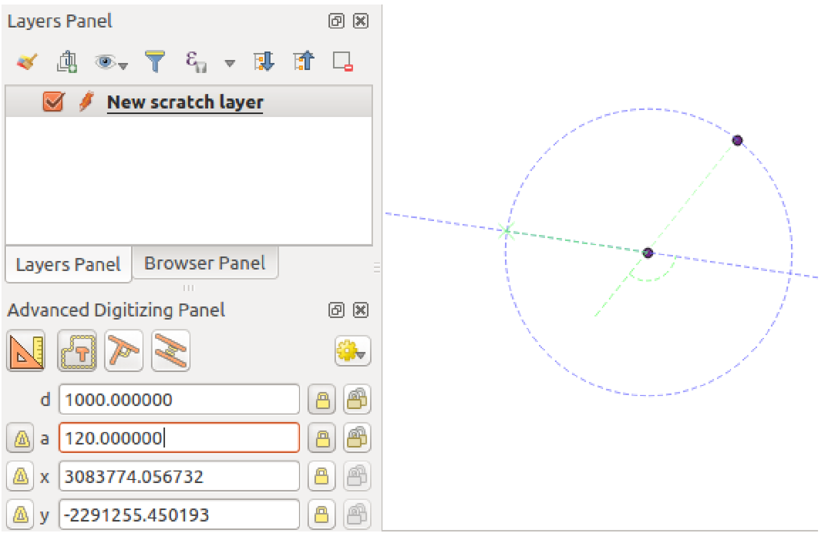


Abb. 14.102: Abstand von und Winkel zu einem Punkt

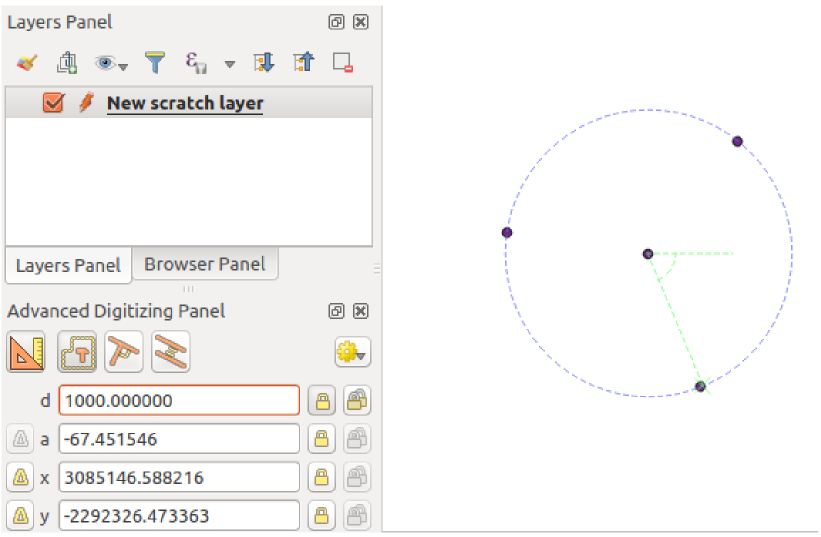


Abb. 14.103: Punkte in festgelegtem Abstand und Winkel

14.4.7 Mit Verarbeitung Objekte innerhalb eines Layers verändern

Das Menü *Processing menu* biete eine große Zahl von Werkzeugen zur Analyse und Erzeugung neuer Objekte auf der Basis der Eigenschaften vorhandener Objekte oder ihrer Bezeihung 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.

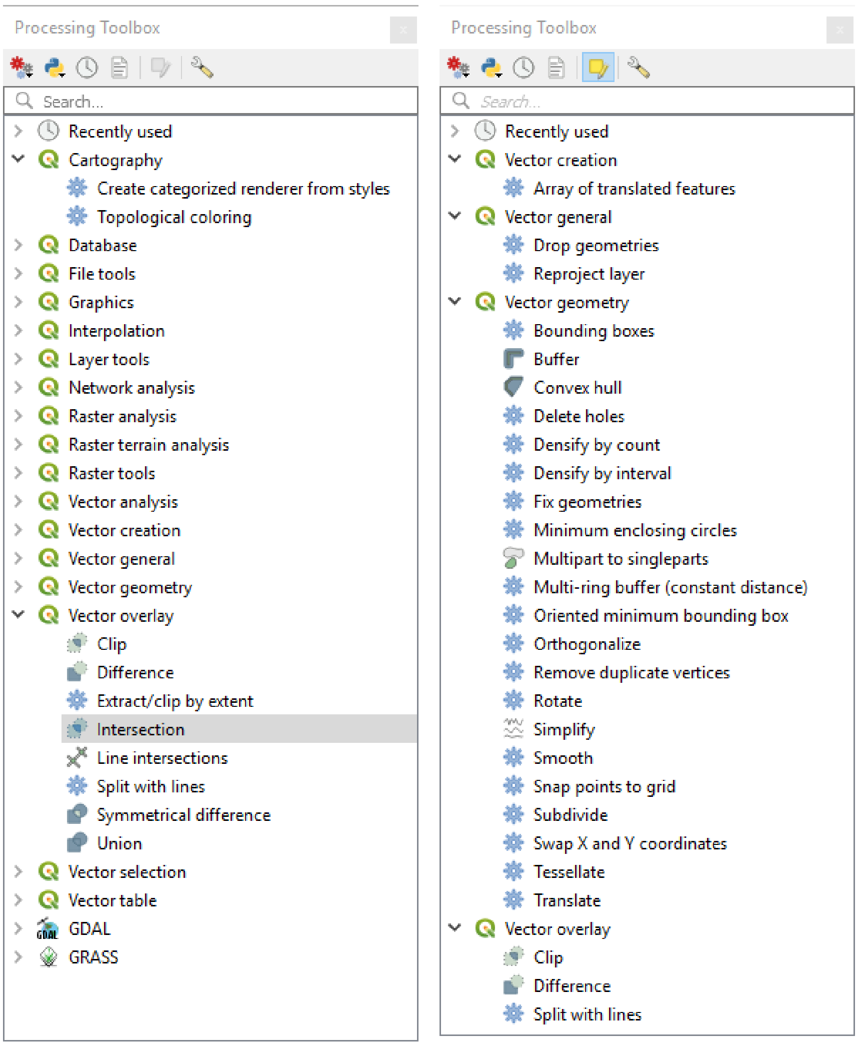


Abb. 14.104: Verarbeitungswerkzeuge: alle (links) und insitu-Änderung (rechts)

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 des 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.

KAPITEL 15

Arbeiten mit Rasterdaten

15.1 Dialogfenster Rasterlayereigenschaften

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:

- *Information*
- *Source*
- *Symbology*
- *Transparency*
- *Histogram*
- *Rendering*
- *Pyramids*
- *Metadata*
- *Legend*
- *QGIS Server*

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 (symbology, label, actions, default values, forms…) of embedded layers (see *Layer/Gruppen einbinden*) 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.

15.1.1 Information Properties

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:

- based on the provider of the layer (format of storage, path, 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 *provided metadata*: access, links, contacts, history… as well as dataset information (CRS, Extent, bands…).

15.1.2 Source Properties

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*.

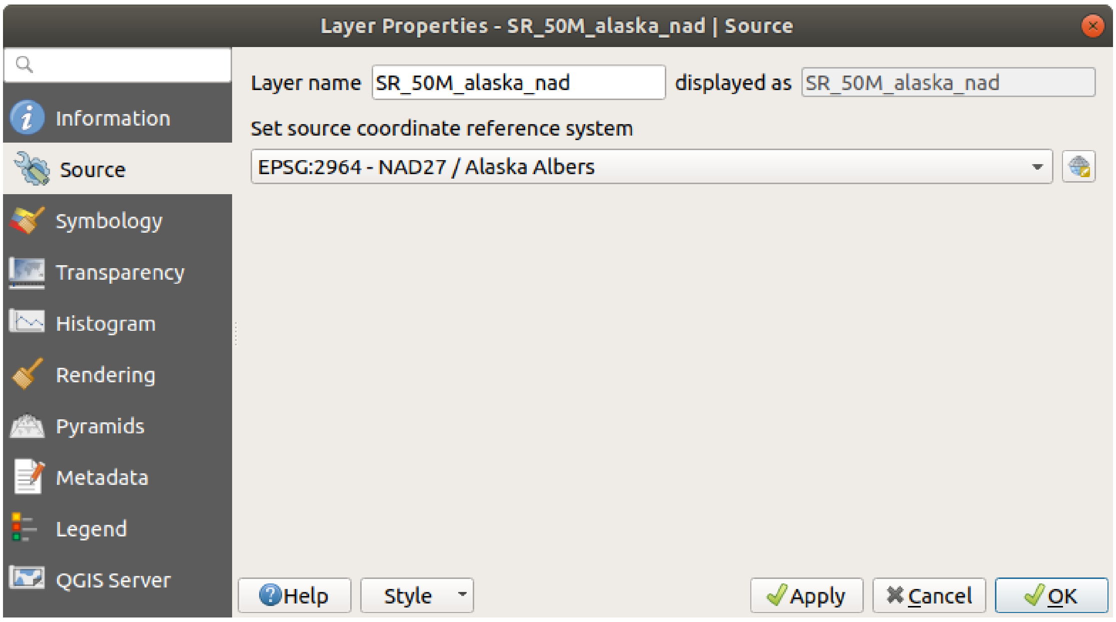


Abb. 15.1: Raster Layer Properties - Source Dialog

15.1.3 Symbology Properties

Kanaldarstellung

QGIS offers four different *Render types*. The choice of renderer depends on the data type.

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.

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.

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

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*).

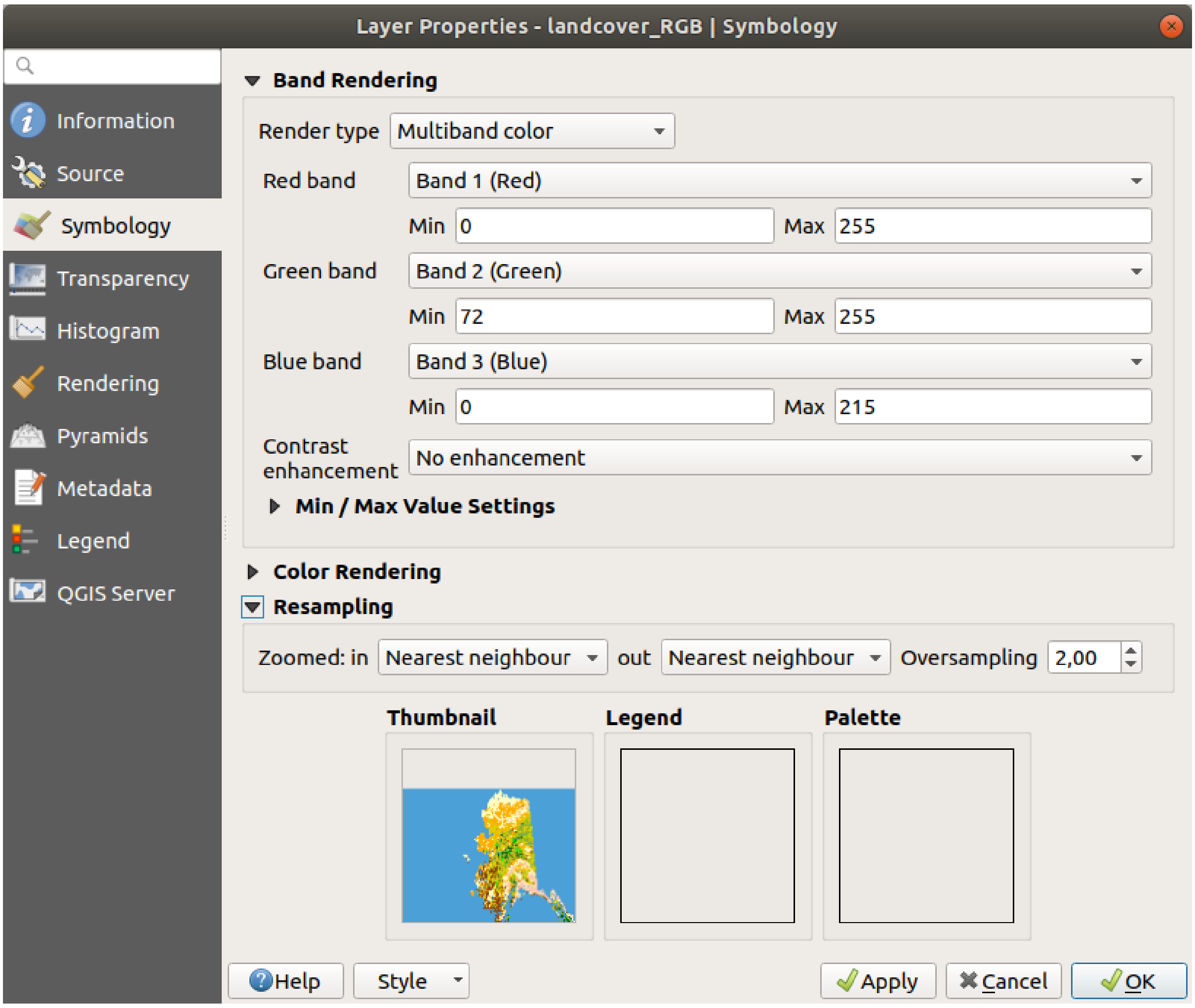


Abb. 15.2: Raster Symbology - Multiband color rendering

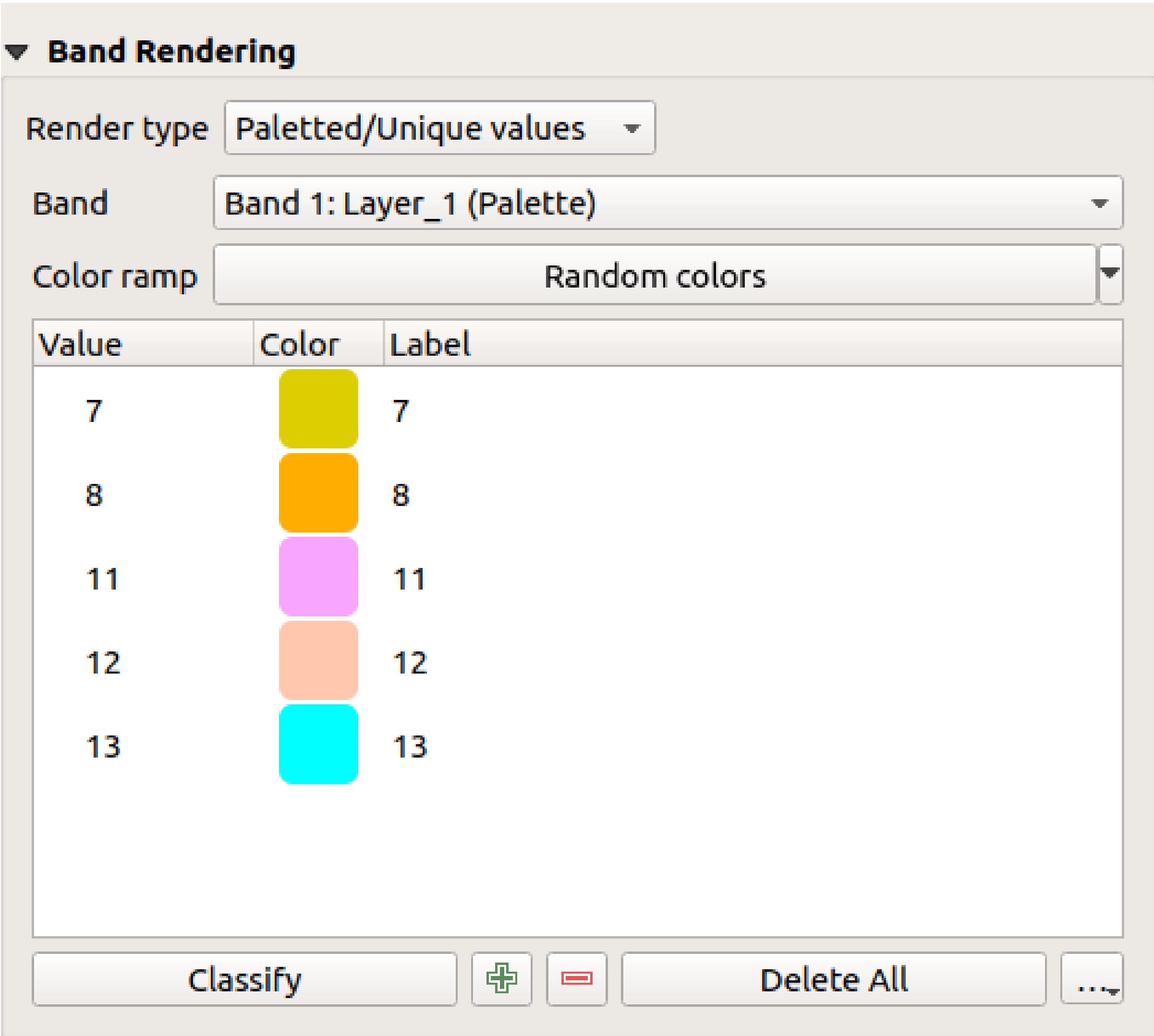


Abb. 15.3: Raster Symbology - Paletted unique value rendering

Singleband gray

This renderer allows you to render a single band layer 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‘.

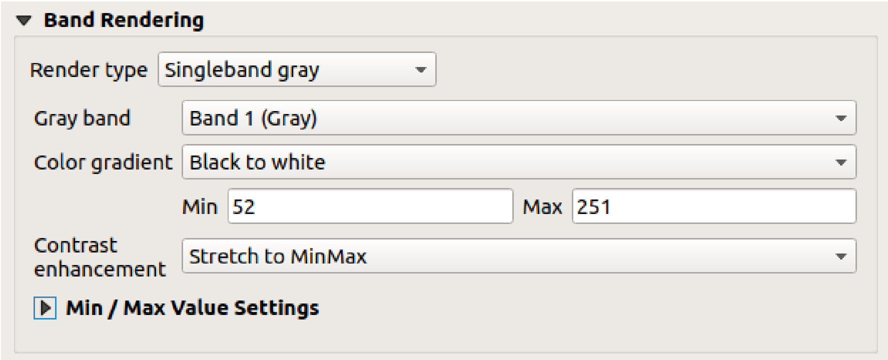


Abb. 15.4: Raster Symbology - Singleband gray rendering

Singleband pseudocolor

This is a render option for single-band files that include a continuous palette. You can also create color maps for a bands of a multiband raster.

Using a *Band* of the layer and a *values range*, three types of color *Interpolation* are available:

- Discrete (a <= symbol appears in the header of the *Value* column)
- Linear
- Exact (an = symbol appears in the header of the *Value* column)

The *Color ramp* drop down lists the available color ramps. 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 and in the QML file.

The *Label unit suffix* is a label added after the value in the legend.

For classification *Mode* ‚Equal interval‘, you only need to select the *number of classes* 1,00 and press the button *Classify*. For *Mode* ‚Continuous‘, QGIS creates classes automatically depending on *Min* and *Max*.

The button Add values manually adds a value to the table. The button Remove selected row deletes a value from the table. Double clicking in the *Value* column lets you insert a specific value. Double clicking in the *Color* column opens the dialog *Change color*, where you can select a color to apply for that value. Further, you can also add labels for each color, 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…* for the selection
- *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.

The *Clip out of range values* allows QGIS to not render pixel greater than the *Max* value.

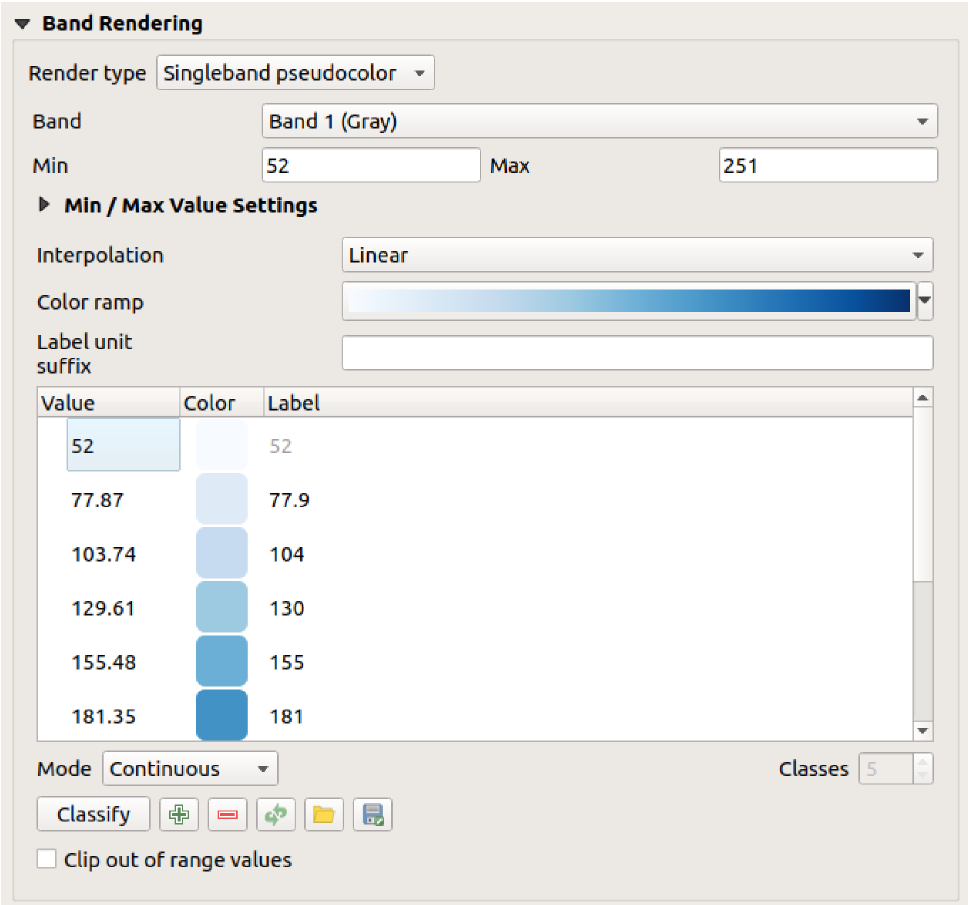


Abb. 15.5: Raster Symbology - Singleband pseudocolor rendering

Hillshade

Render a band of the raster layer using hillshading.

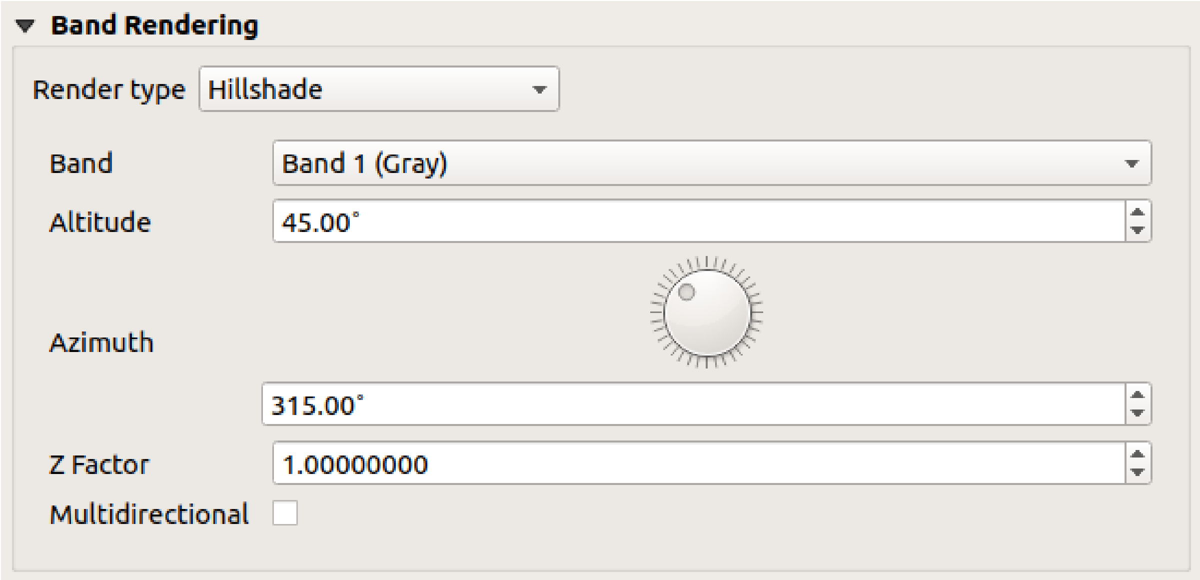


Abb. 15.6: Raster Symbology - Hillshade rendering

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`).

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.

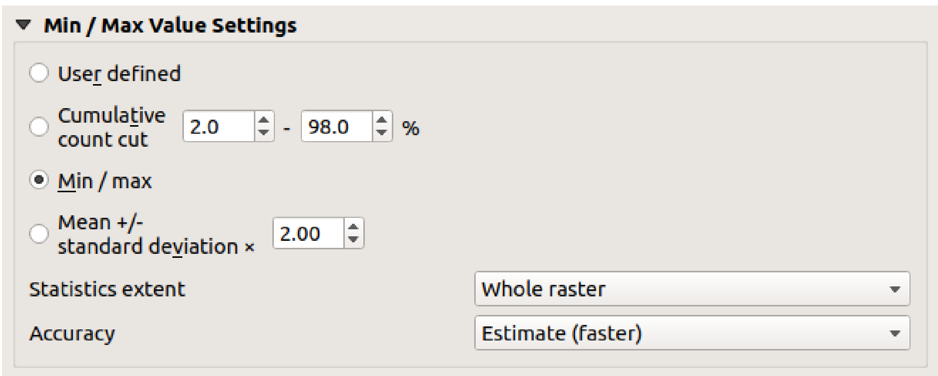


Abb. 15.7: Raster Symbology - Min and Max Value Settings

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.

Farbdarstellung

For all kinds of *Band rendering*, the *Color rendering* set.

You can achieve special rendering effects for your raster file(s) by using one of the blending modes (see *Mischmodi*).

Further settings can be made by modifying the *Brightness*, *Saturation* and *Contrast*. You can also use a *Grayscale* option, where you can choose between ‚Off', ‚By lightness', ‚By luminosity' and ‚By average'. For one *Hue* in the color table, you can modify the ‚Strength'.

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.

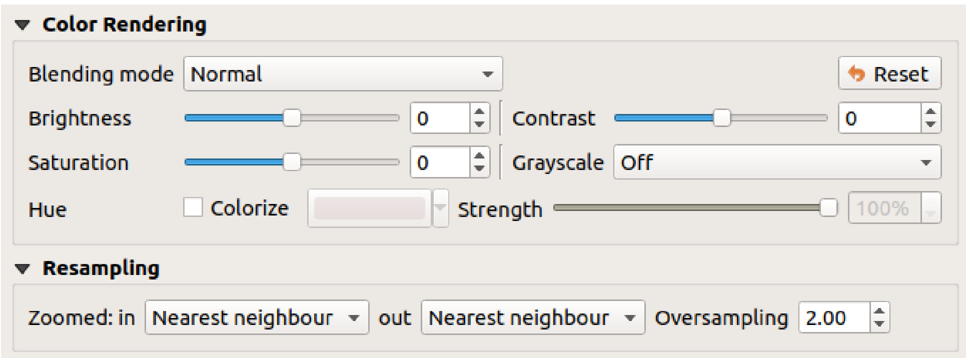


Abb. 15.8: Raster Symbology - Color rendering and Resampling settings

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' or ‚Cubic' 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.

At the bottom of the *Symbology* tab, you can see a thumbnail of the layer, its legend symbol, and the palette.

15.1.4 Transparency Properties

QGIS has the ability to set the transparency level of a raster layer. Use the transparency 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.

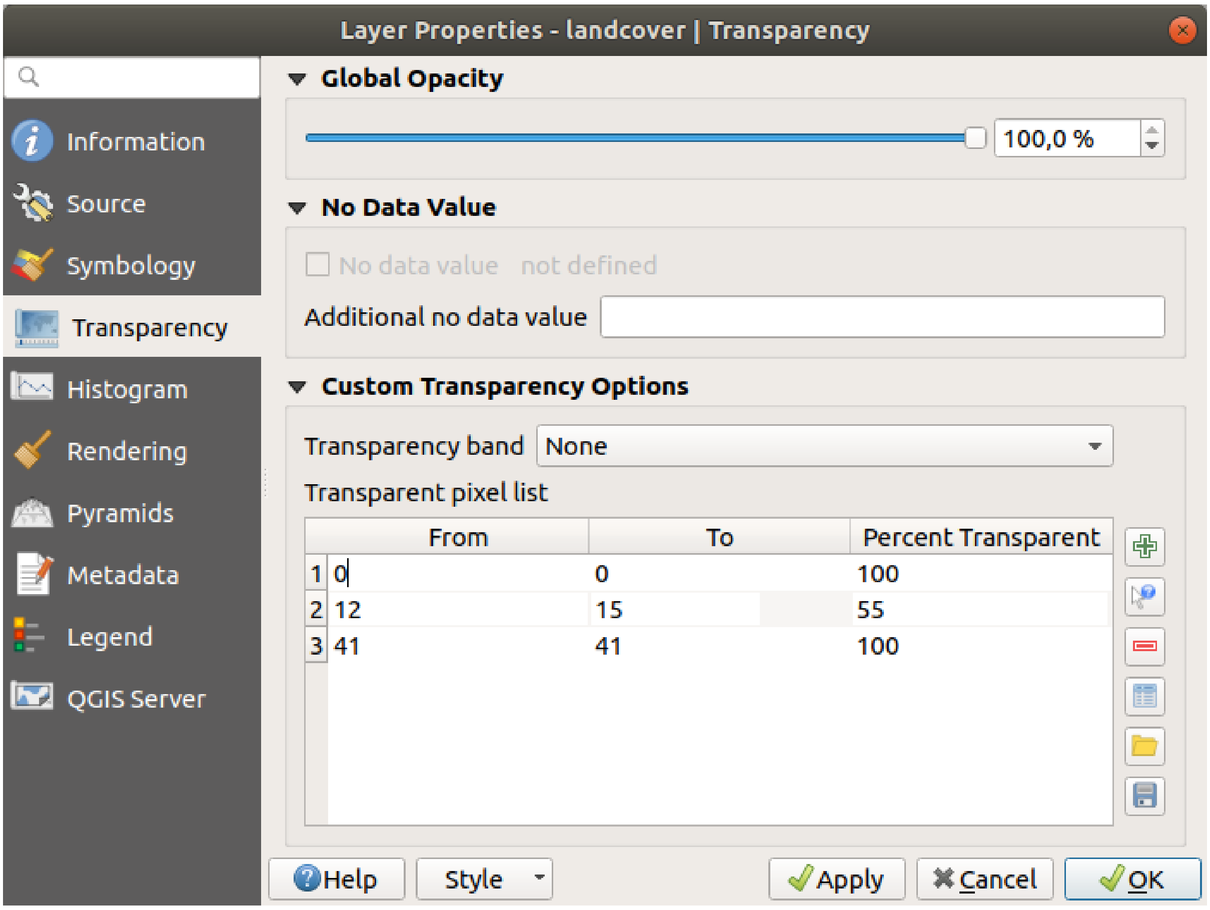


Abb. 15.9: Raster Transparency

Additionally, you can enter a raster value that should be treated as an *Additional no data value*.

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.

15.1.5 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).

15.1.6 Rendering Properties

In the *Rendering* tab, it's possible to:

- set *Scale dependent visibility* for the layer: You can set the *Maximum (inclusive)* and *Minimum (exclusive)* scale, 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 *Maßstabsabhängige Layeranzeige* for more information.
- *Refresh layer at interval (seconds)*: set a timer to automatically refresh individual layers. Canvas updates are deferred in order to avoid refreshing multiple times if more than one layer has an auto update interval set.

15.1.7 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

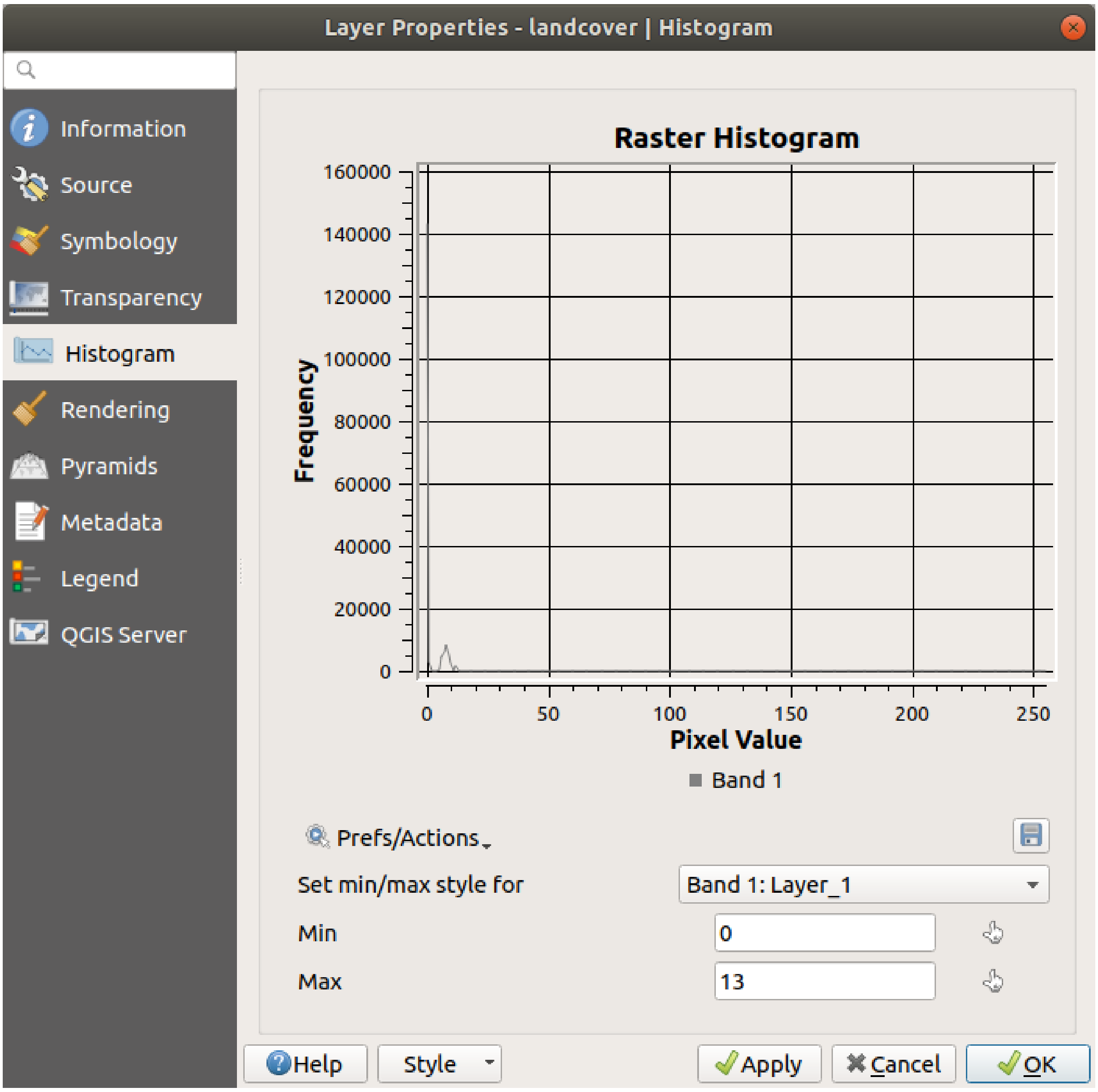


Abb. 15.10: Rasterhistogramm

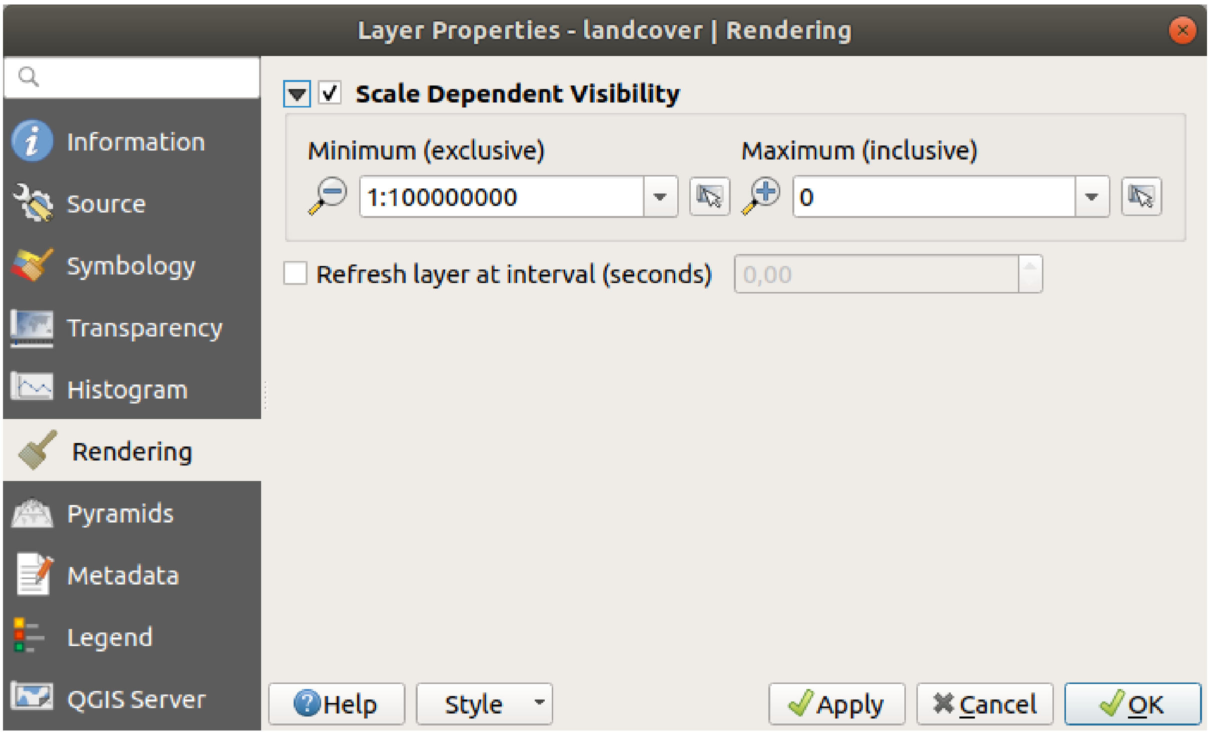


Abb. 15.11: Raster Rendering

- Durchschnitt
- Gauß
- Kubisch
- Cubic Spline
- Laczos
- Modus
- Keine

Finally, click *Build Pyramids* to start the process.

15.1.8 Metadata Properties

The *Metadata* tab provides you with options to create and edit a metadata report on your layer. See *vector layer metadata properties* for more information.

15.1.9 Legend Properties

The *Legend* tab 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.

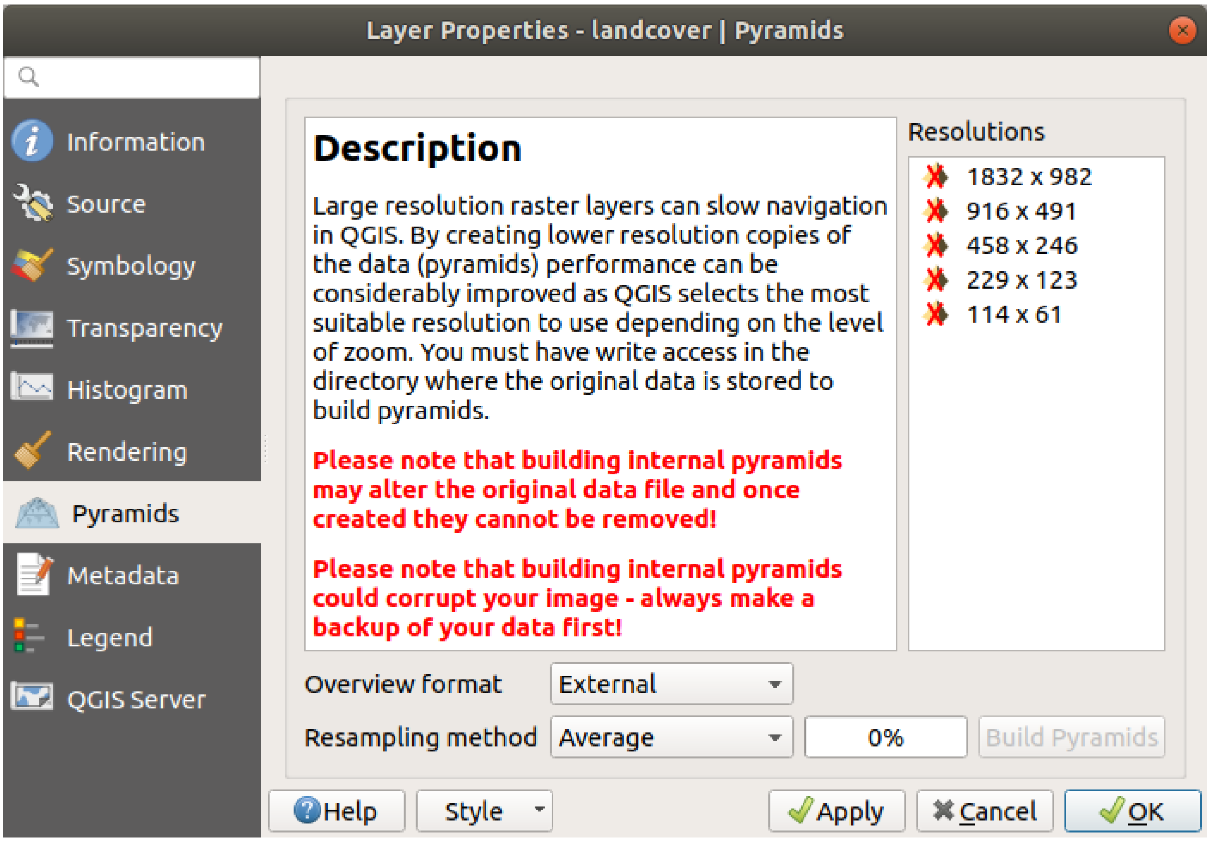


Abb. 15.12: Raster Pyramids

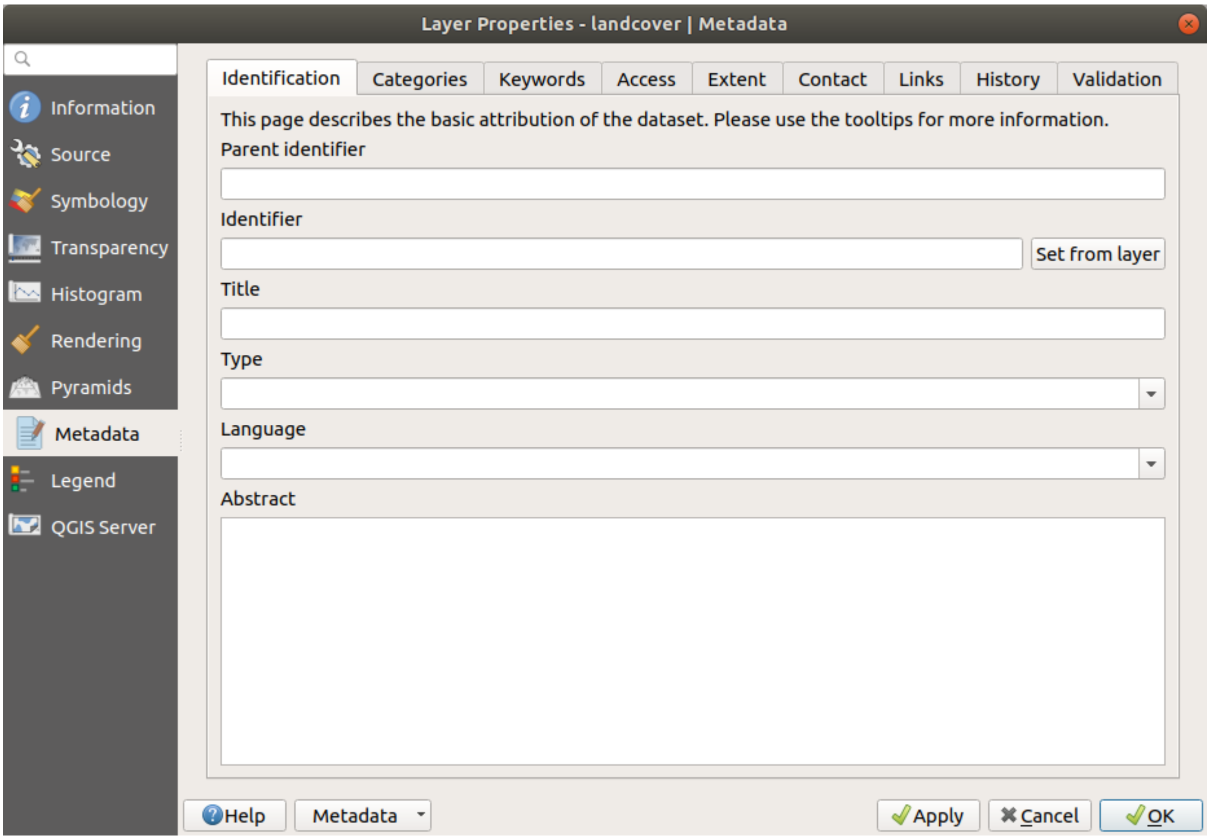


Abb. 15.13: Raster Metadata

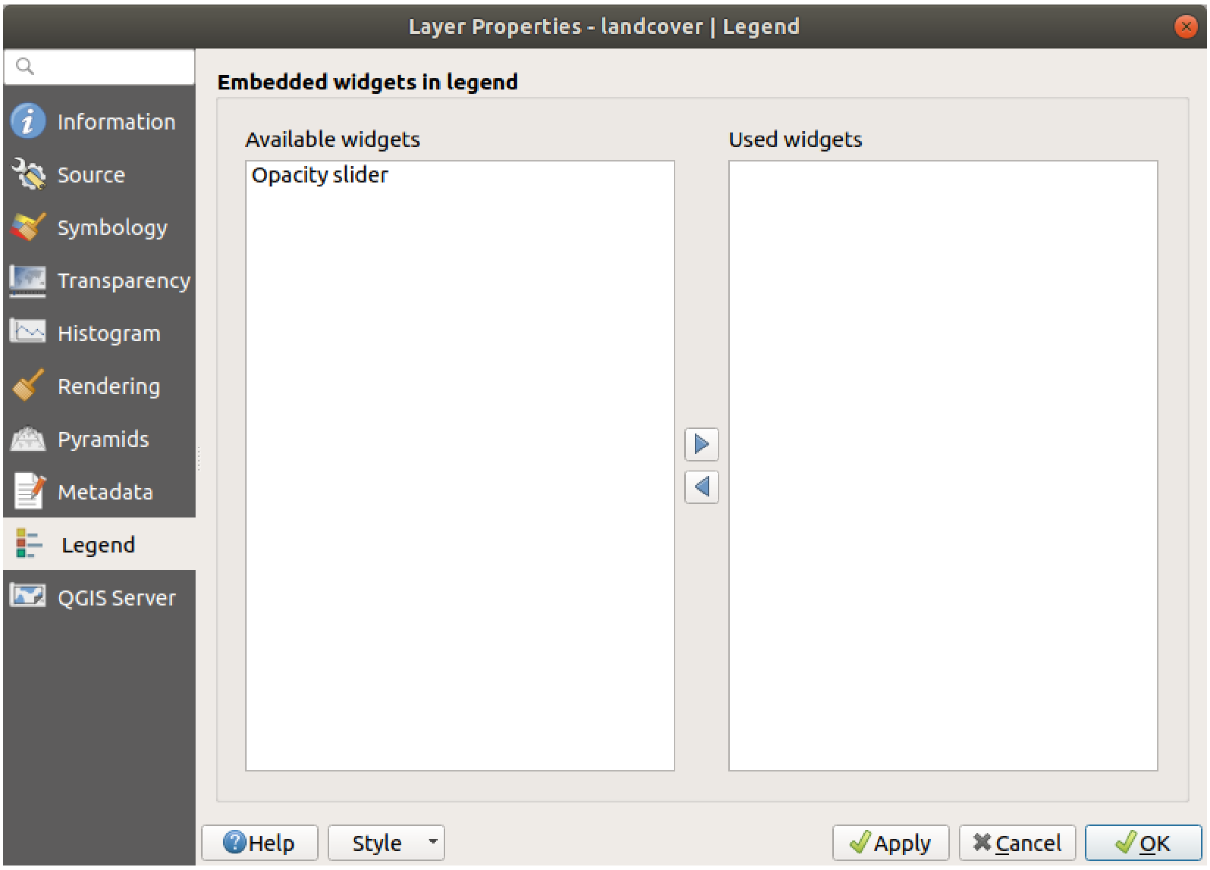


Abb. 15.14: Raster Legend

15.1.10 QGIS Server Properties

From the *QGIS Server* tab, information can be provided for *Description*, *Attribution*, *MetadataUrl* and *LegendUrl*.

15.2 Rasteranalyse

15.2.1 Rasterrechner

Der *Rasterrechner* im *Raster* Menü ermöglicht es Ihnen Berechnungen auf der Basis von bestehenden Pixelwerten durchzuführen (siehe *figure_raster_calculator*). Die Ergebnisse werden in einen neuen Rasterlayer in einem GDAL-unterstützten Format geschrieben.

Die **Rasterkanäle** Liste enthält alle geladenen Rasterlayer die benutzt werden können. Um einen Rasterlayer dem Rasterrechnerausdruck Feld hinzuzufügen, machen Sie einen Doppelklick auf seinen Namen in der Felder Liste. Sie können dann die Operatoren zum konstruieren von Berechnungsausdrücken benutzen oder Sie geben Sie einfach in die Box ein.

Im Abschnitt **Ergebnislayer** müssen Sie einen Ausgabelayer definieren. Sie können dann den Analysebereich auf Grundlage eines Eingaberasters oder basierend auf Min/Max X und Y-Koordinaten bzw. mittels Spalten und Zeilen angeben, um die Auflösung des Ausgabelayer festzulegen. Wenn die Eingabelayer eine abweichende Auflösung besitzen, werden die Werte auf Basis des nearest neighbor Algorithmus resampelt.

Der Bereich **Operatoren** stellt Operatoren für die Berechnungen bereit. Um einen Operator auszuwählen, klicken Sie auf die entsprechende Schaltfläche. Es stehen mathematische Berechnungen (+, -, *, …) und trigonometrische Funktionen (`sin`, `cos`, `tan`, …) zur Verfügung. Bedingte Ausdrücke (=` , ! =, <` <` , >=, ….) geben entweder 0 für falsch oder 1 für wahr zurück und können daher mit anderen Operatoren und Funktionen verwendet werden. Die Anzahl wird mit den nächsten Versionen sicherlich noch wachsen!

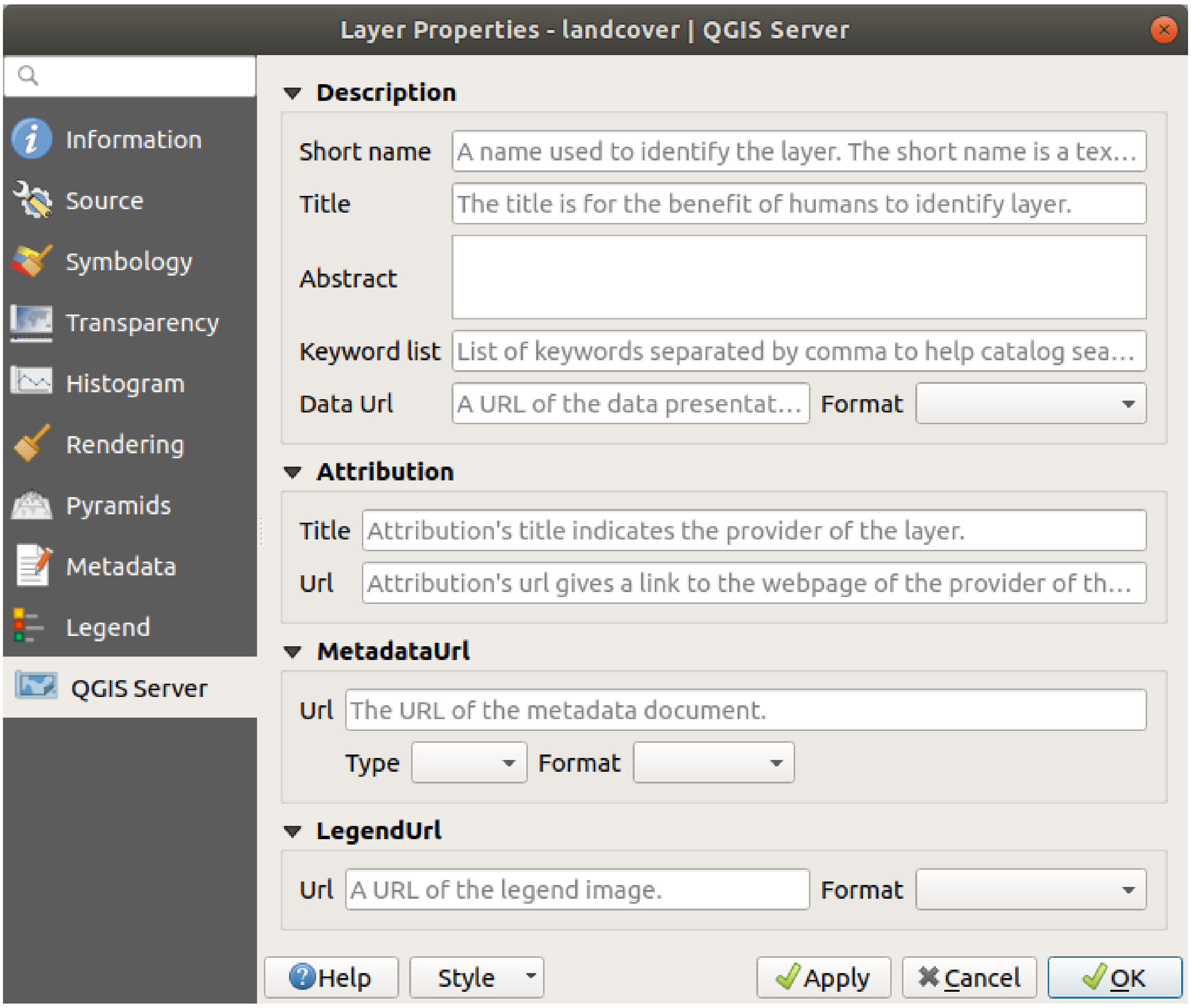


Abb. 15.15: QGIS Server in Raster Properties

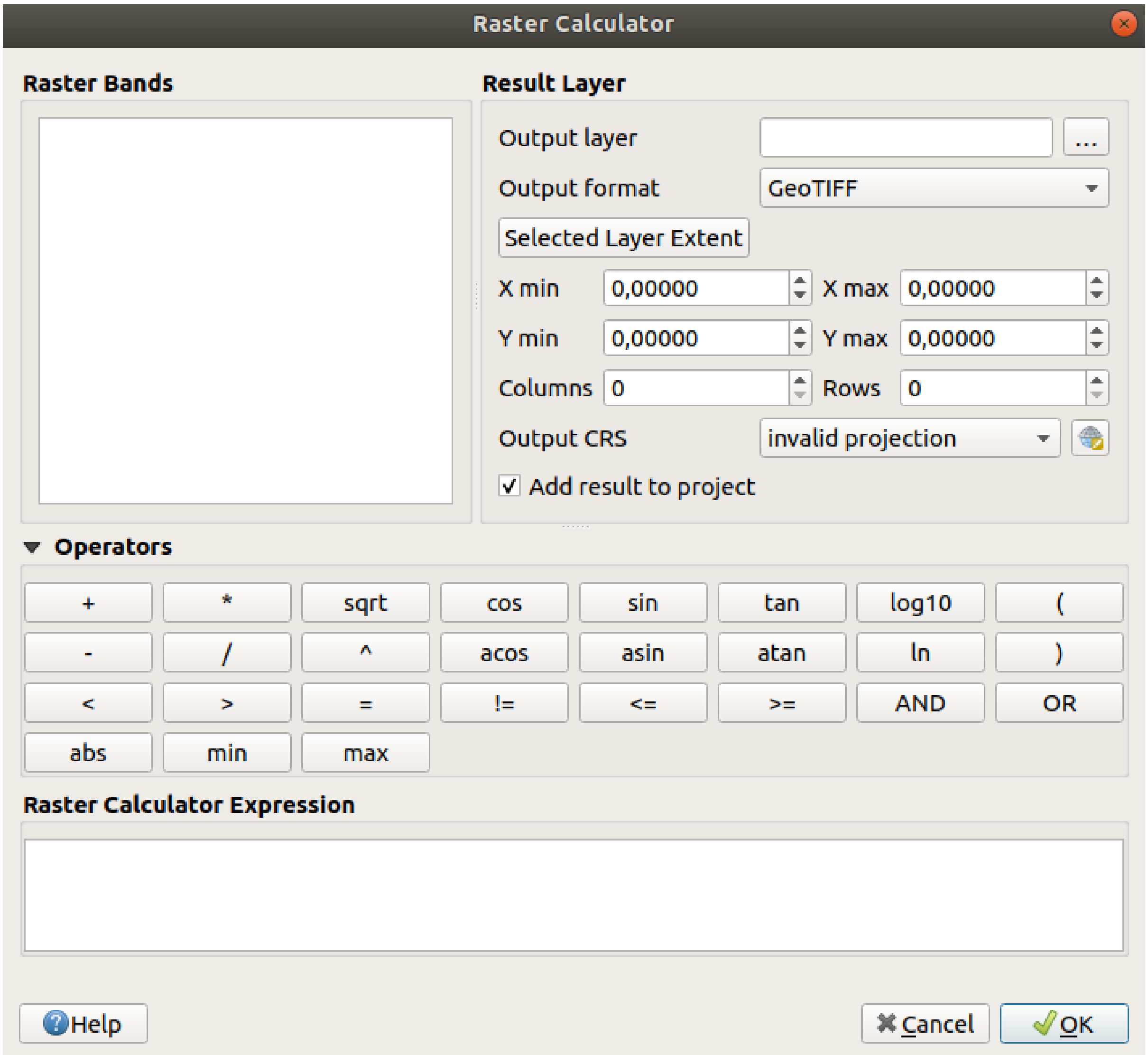


Abb. 15.16: Raster Calculator (`abs`, `min` and `max` added in 3.10)

Mit dem Aktivieren des Kontrollkästchens ☑ *Ergebnis zum Projekt hinzufügen* wird der Ausgabelayer automatische der Legende hinzugefügt und kann somit visualisiert werden.

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.

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.

15.2.2 Raster Ausrichtung

Dieses Tool ist in der Lage mehrere Raster als Eingabe zu nehmen und sie perfekt auszurichten, das heißt:

- auf gleichem KBS reprojizieren,
- zu derselben Zellengröße resamplen und in dem Gitter versetzen,
- auf ein Interessengebiet schneiden,
- wenn erforderlich, Werte neu skalieren.

Alle Raster werden in anderen Dateien gespeichert.

Öffnen Sie zunächst die Werkzeuge aus *Raster ▸ Raster ausrichten…* und klicken Sie auf die Schaltfläche Neues Raster hinzufügen, um ein bestehendes Raster in QGIS auszuwählen. Wählen Sie eine Ausgabedatei aus, um das Raster zu speichern (nach der Ausrichtung, der Resampling-Methode und, wenn die Werkzeuge *Werte entsprechend der Zellengröße neu skalieren müssen*). Die Resampling-Methode kann sein (siehe *figure_raster_align_edit*):

- **Nächster Nachbar**
- **Bilinear (2x2 Kern)**
- **Kubisch (4x4 kernel)**: Cubic Convolution Approximation
- **Kubisch B-Spline (4x4 kernel)**: Kubische B-Spline Annäherung
- **Lanczos (6x6 kernel)**: Lanczos windowed sinc interpolation
- **Average**: computes the average of all non-NODATA contributing pixels
- **Mode**: selects the value which appears most often of all the sampled points

- **Maximum**, **Minimum**, **Mediane**, **First Quartile (Q1)** or **Third Quartile (Q3)** of all non-NODATA contributing pixels

Bemerkung: Methods like maximum, minimum, mediane, first and third quartiles are available only if QGIS is built with GDAL >= 2.0.

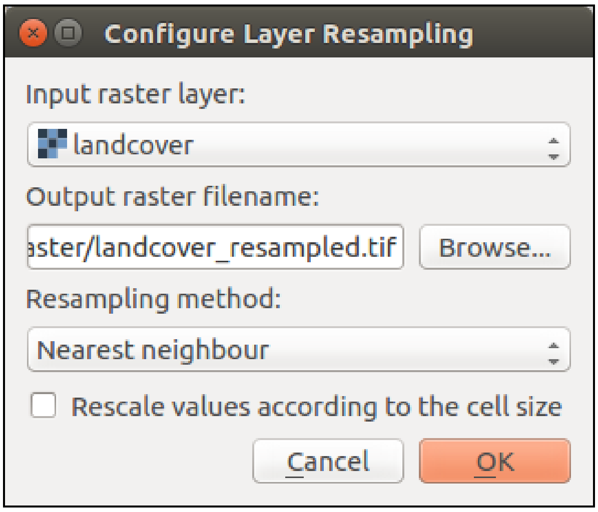


Abb. 15.17: Select Raster Resampling Options

In the main *Align raster* dialog, you can still Edit file settings or Remove an existing file from the list of raster layers. You can also choose one or more other options (see *figure_raster_align*):

- Select the *Reference Layer*,
- Transform into a new *CRS*,
- Setup a different *Cell size*,
- Setup a different *Grid Offset*,
- *Clip to Extent*: it can be user-defined or based on a layer or the map view
- *Output Size*,
- *Add aligned raster to the map canvas*.

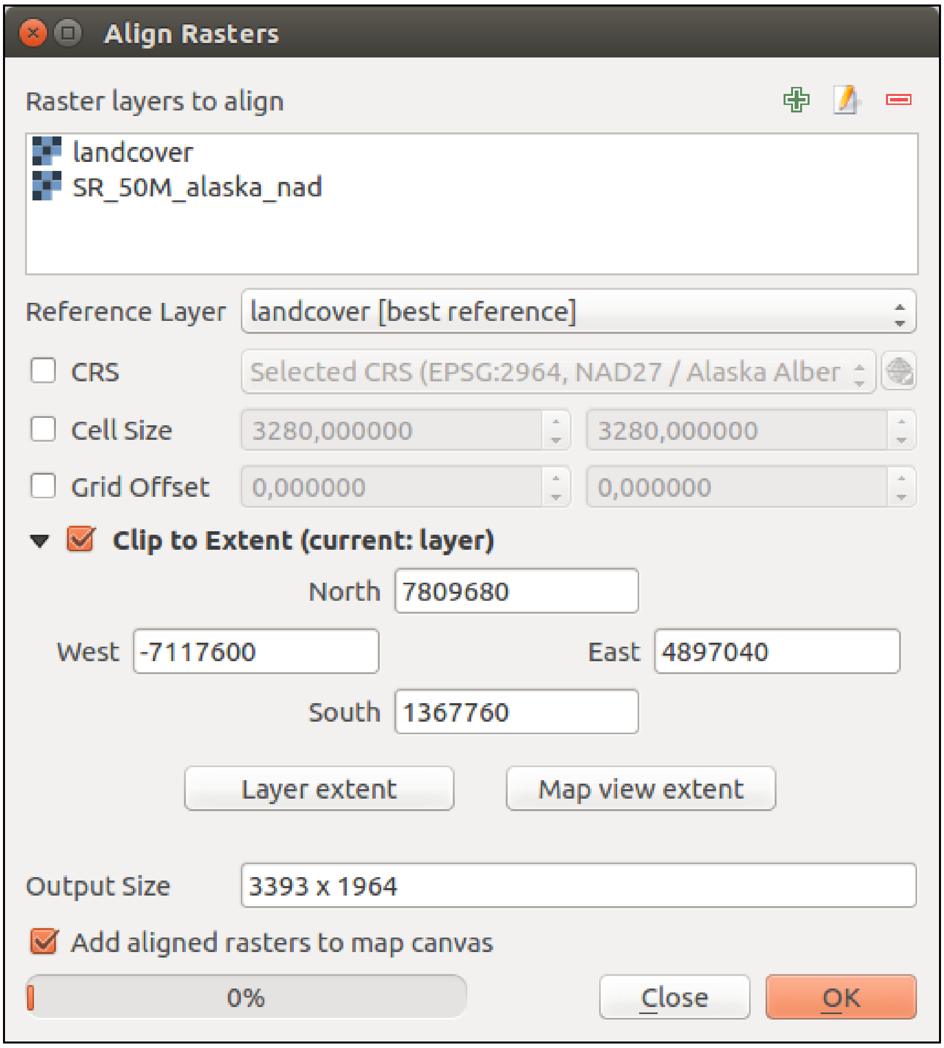


Abb. 15.18: Raster Ausrichtung

KAPITEL 16

Working with Mesh Data

16.1 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 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

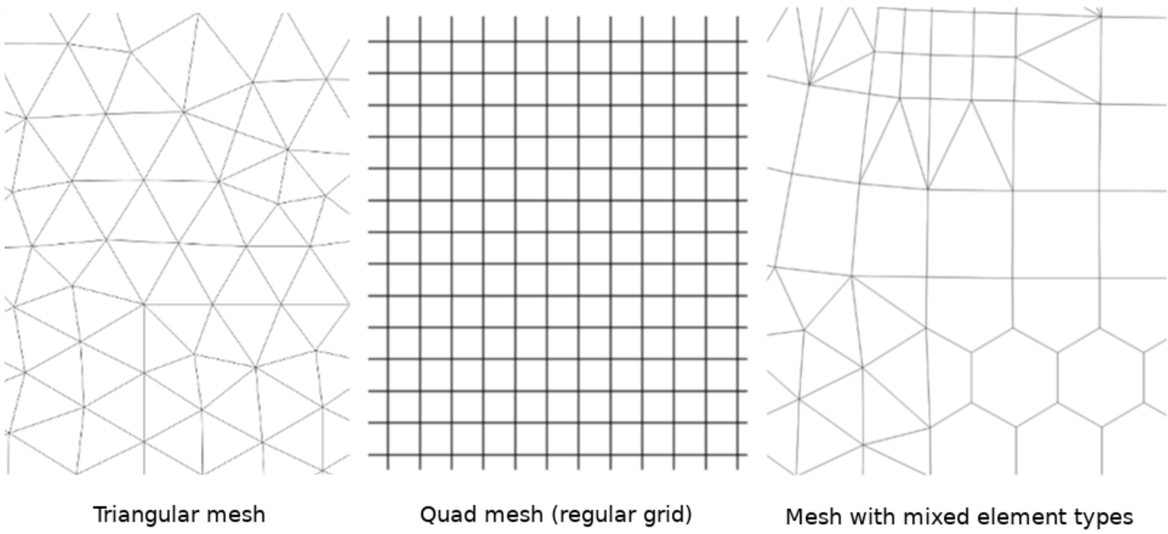


Abb. 16.1: Different mesh types

QGIS can currently render mesh data using triangles or regular quads.

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:

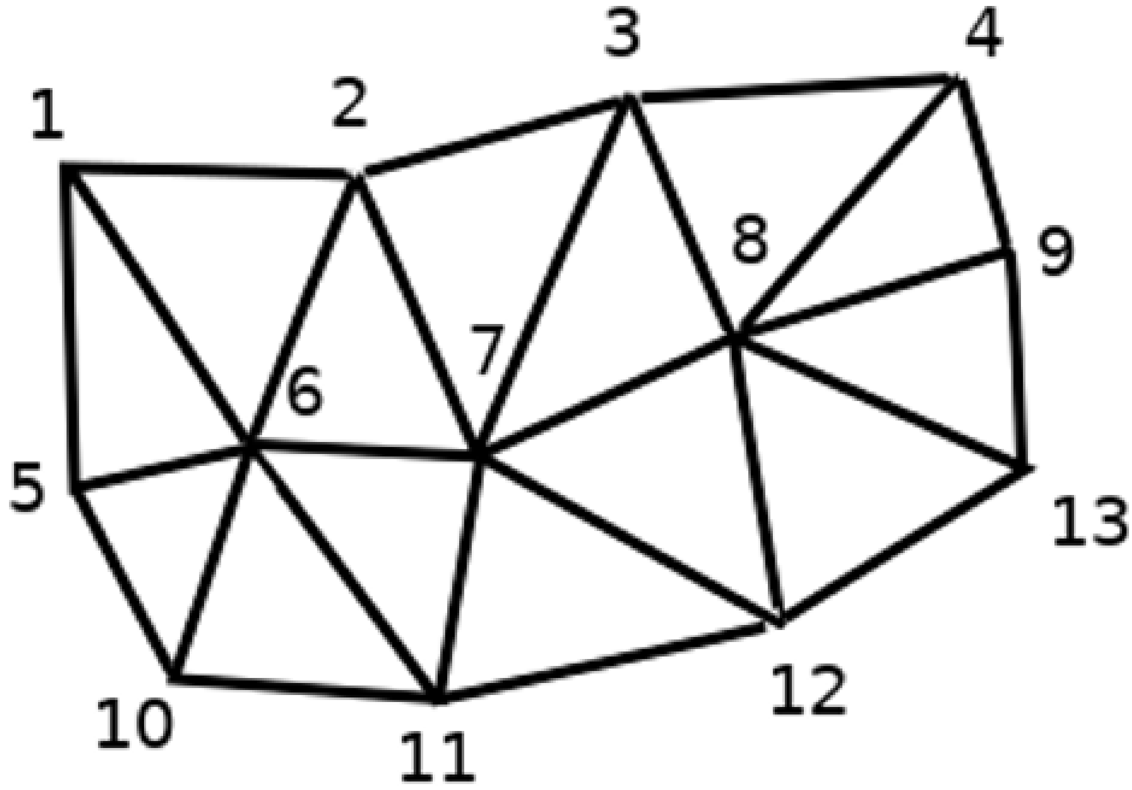


Abb. 16.2: Triangular grid with numbered vertices

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:

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.

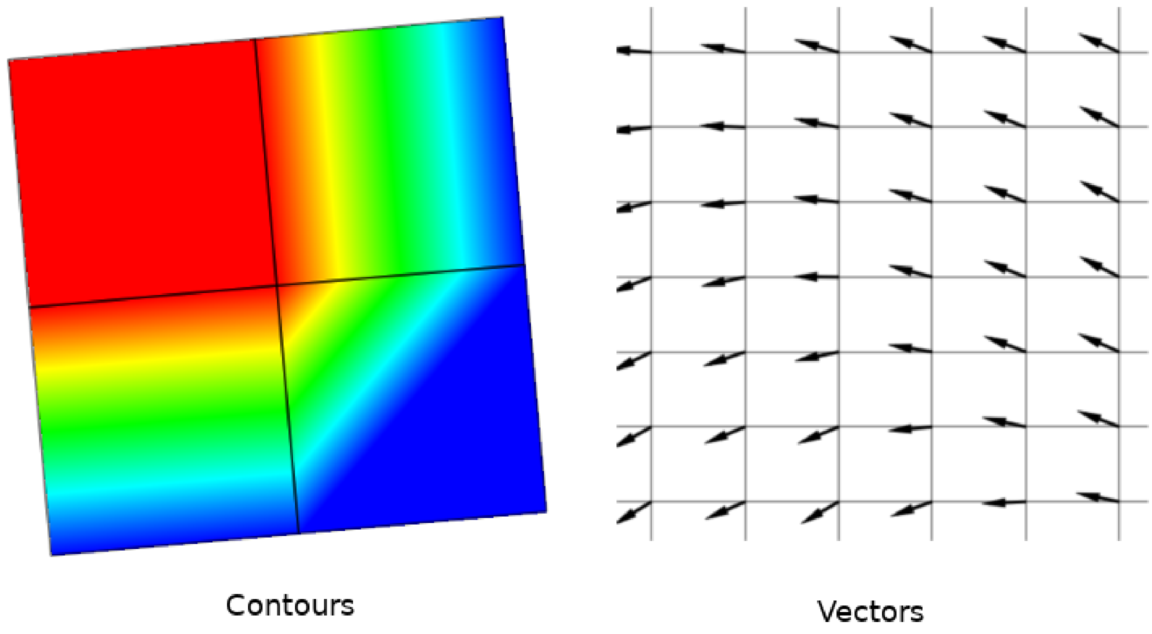


Abb. 16.3: Possible visualisation of mesh data

16.2 Supported formats

QGIS accesses mesh data using the MDAL drivers. Hence, the natively supported formats are:

- `NetCDF`: Generic format for scientific data
- `GRIB`: Format commonly used in meteorology
- `XMDF`: As an example, hydraulic outputs from TUFLOW modelling package
- `DAT`: Outputs of various hydrodynamic modelling packages (e.g. BASEMENT, HYDRO_AS-2D, TUFLOW)
- `3Di`: 3Di modelling package format based on Climate and Forecast Conventions (http://cfconventions.org/)
- Some examples of mesh datasets can be found at https://apps.ecmwf.int/datasets/data/interim-full-daily/levtype=sfc/

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.

16.3 Mesh Dataset Properties

16.3.1 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) uri, vertex count, face count and dataset groups count.

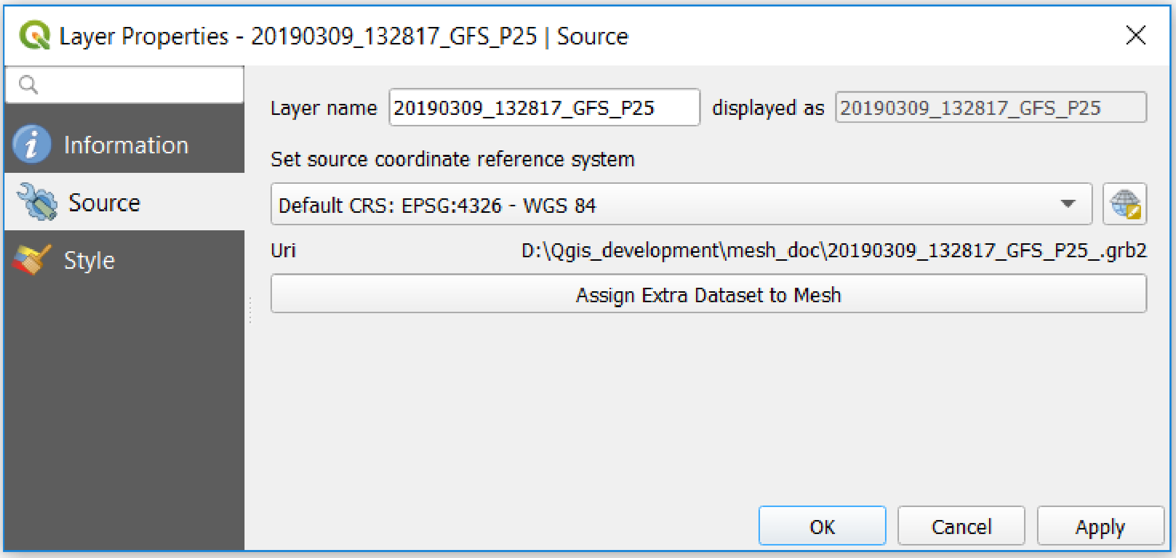


Abb. 16.4: Mesh Layer Properties

16.3.2 Source Properties

The *Source* tab displays basic information about the selected mesh, including:

- the Layer name to display in the *Layers* panel
- setting 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 *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.

Use the *Assign Extra Dataset to Mesh* button to add more groups to the current mesh layer.

16.3.3 Symbology Properties

Click the *Symbology* button to activate the dialog as shown in the following image:

Symbology properties are divided in several tabs:

- *General*
- *Contours Symbology*
- *Vectors Symbology*
- *Rendering*

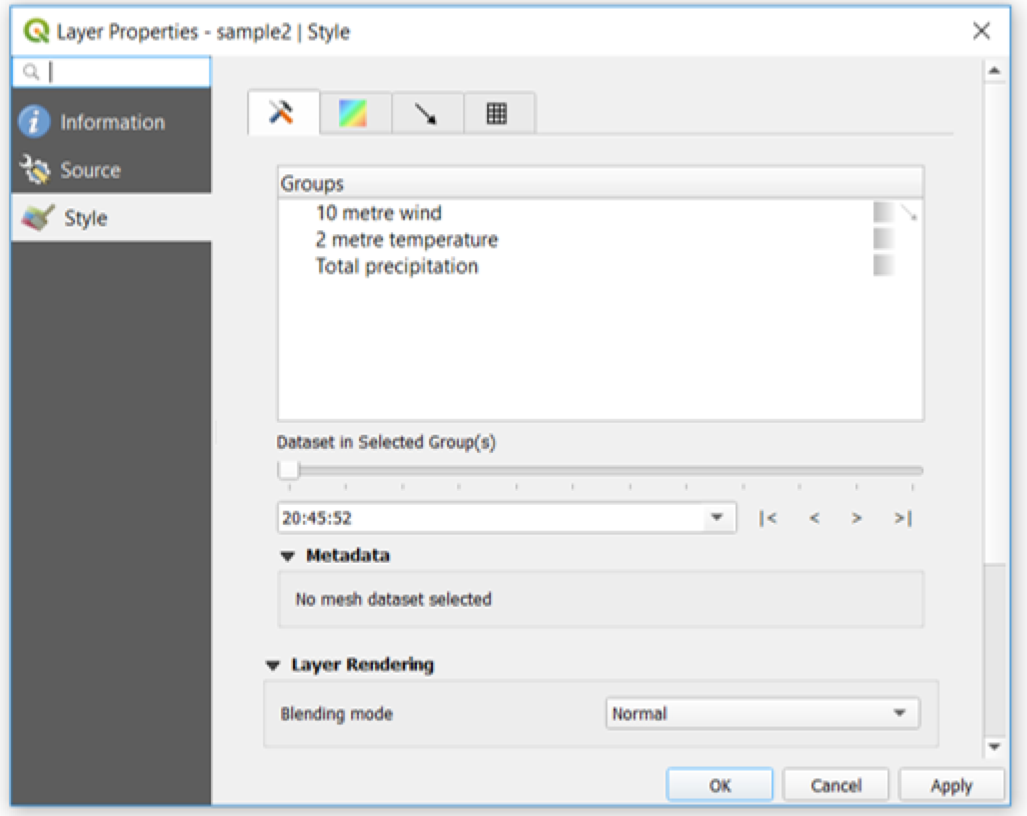


Abb. 16.5: Mesh Layer Symbology

General

The tab presents the following items:

- groups available in the mesh dataset
- dataset in the selected group(s), for example, if the layer has a temporal dimension
- metadata if available
- *blending mode* available for the selected dataset.

The slider , the combo box and the |<, <, >, >| buttons allow to explore another dimension of the data, if available. As the slider moves, the metadata is presented accordingly. See the figure *Mesh groups* below as an example. The map canvas will display the selected dataset group as well.

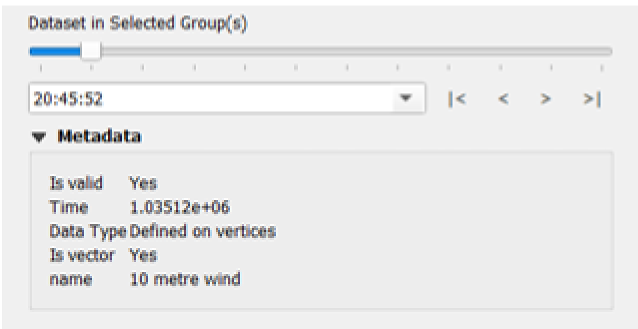


Abb. 16.6: Dataset in Selected Group(s)

You can apply symbology to each group using the tabs.

Contours Symbology

Under *Groups*, click on to show contours with default visualization parameters.

In the tab you can see and change the current visualization options of contours for the selected group, as shown in Abb. 16.7 below:

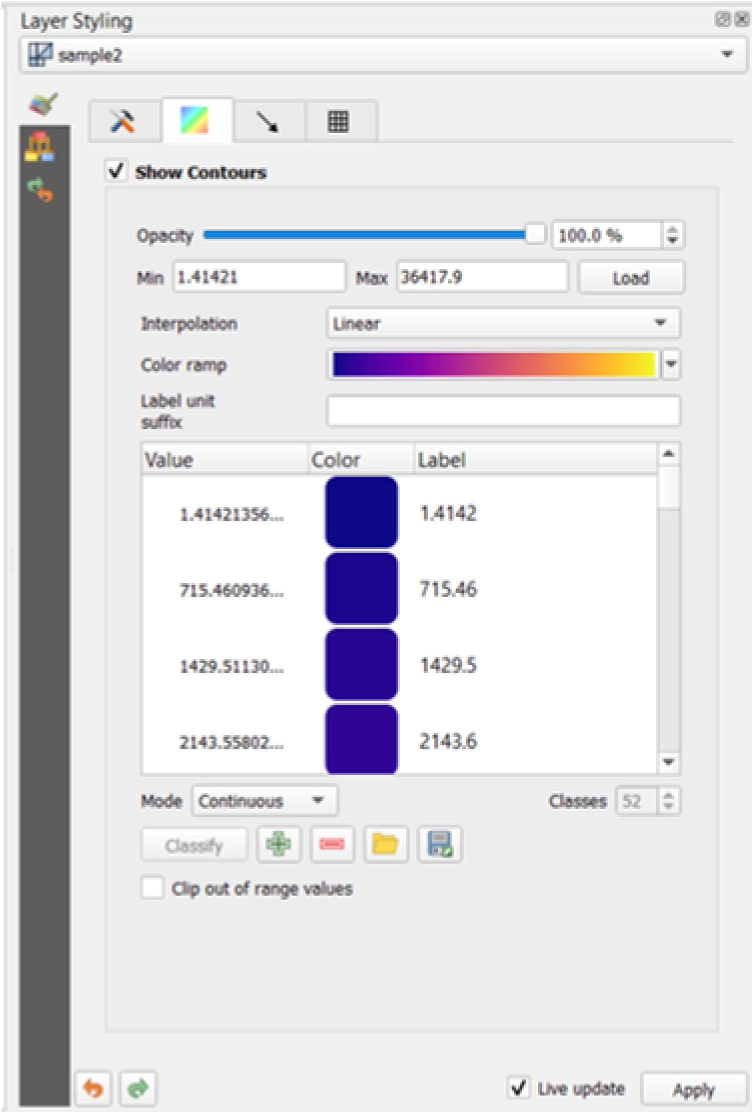


Abb. 16.7: Styling Contours in a Mesh Layer

Use the slide bar or combo box to set the opacity of the current group.

Use *Load* to adjust the min and max values of the current group.

The *Interpolation* list contains three options to render contours: *Linear*, *Discrete* and *Exact*.

The *Color ramp* widget opens the *color ramp drop-down shortcut*.

The *Label unit suffix* is a label added after the value in the legend.

By selecting *Continuous* in the classification *Mode*, QGIS creates classes automatically considering the *Min* and *Max* values. With 'Equal interval', you only need to select the number of classes using the combo box *Classes* and press the button *Classify*.

The button Add values manually adds a value to the individual color table. The button Remove selected row deletes a value from the individual color table. Double clicking on the value column lets you insert a specific value. Double clicking on the color column opens the dialog *Change color*, where you can select a color to apply on that value.

Vectors Symbology

In the tab , click on to display vectors if available. The map canvas will display the vectors in the selected group with default parameters. Click on the tab to change the visualization parameters for vectors as shown in the image below:

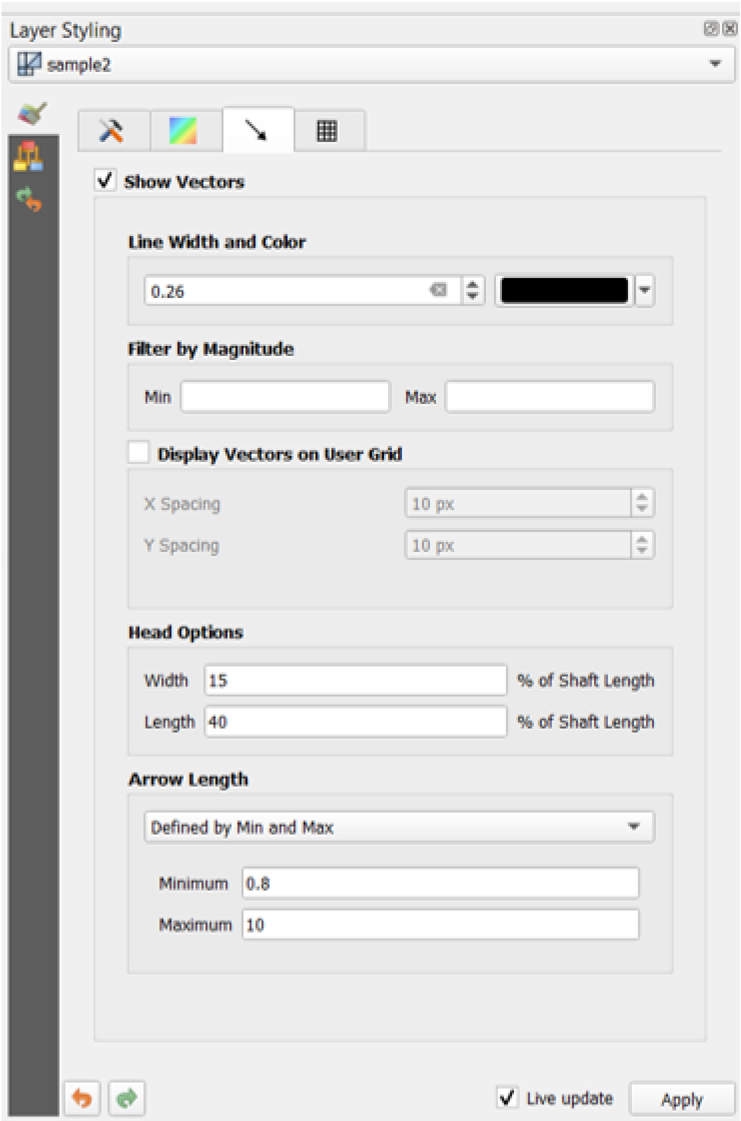


Abb. 16.8: Styling Vectors in a Mesh Layer

The line width can be set using the combo box or typing the value. The color widget opens the dialog *Change color*, where you can select a color to apply to vectors.

Enter values for *Min* and *Max* to filter vectors according to their magnitude.

Check on the box *Display Vectors on User Grid* and specify the *X spacing* and the *Y spacing*, QGIS will render the vector considering the given spacing.

With the Head Options *Head Options*, QGIS allows the shape of the arrow head to be set by specifying width and length (in percentage).

Vector's *Arrow length* can be rendered in QGIS in three different ways:

- Defined by Min and Max: You specify the minimum and maximum length for the vectors, QGIS will adjust their visualization accordingly
- Scale to magnitude: You specify the (multiplying) factor to use
- Fixed: all the vectors are shown with the same length

Layeranzeige kontrollieren

In the tab , QGIS offers two possibilities to display the grid, as shown in Abb. 16.9:

- `Native Mesh Rendering` that shows quadrants
- `Triangular Mesh Rendering` that display triangles

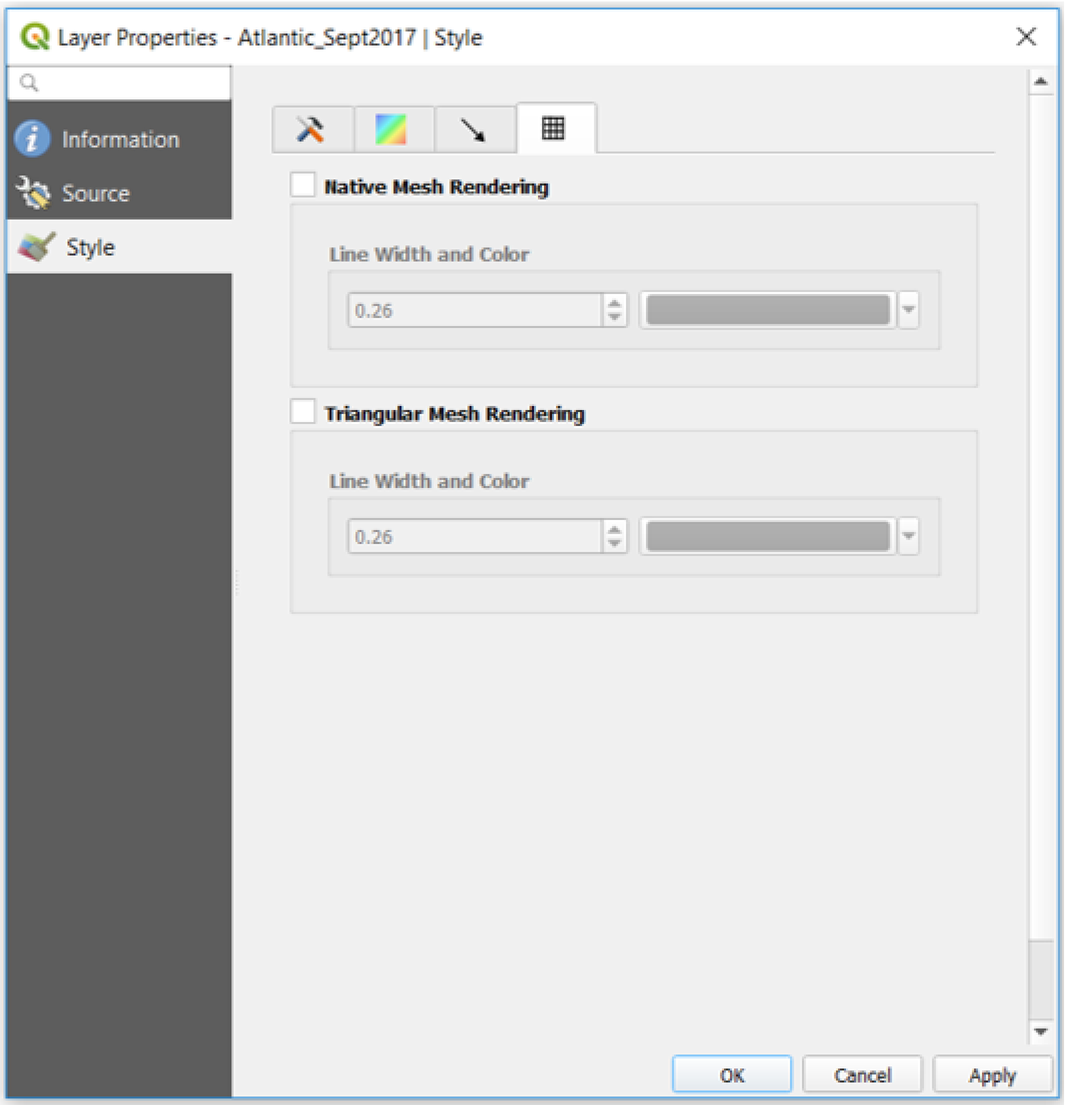


Abb. 16.9: Mesh Rendering

The line width and color can be changed in this dialog, and both the grid renderings can be turned off.

KAPITEL 17

Kartenlayout

Mit Drucklayouts und Berichten können Sie Karten und Kartenserien (Atlas) erzeugen und sie entweder ausdrucken oder als Bild, PDF- oder SVG-Datei speichern.

17.1 Überblick über das Drucklayout

The print layout provides growing layout and printing capabilities. It allows you to add elements such as the QGIS map canvas, text labels, images, legends, scale bars, basic shapes, arrows, attribute tables 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.

17.1.1 Sample Session for beginners

Bevor Sie mit dem Drucklayout anfangen zu arbeiten, müssen Sie einige Raster- oder Vektorlayer in das QGIS Kartenfenster laden und deren Eigenschaften Ihren Wünschen anpassen. Wenn alles nach Ihren Wünschen dargestellt und symbolisiert ist, klicken Sie das auf die Schaltfläche **|newComposer|** Neues Drucklayout in der Werkzeugleiste oder wählen Sie *Projekt ⍰ Neues Drucklayout*. Sie werden aufgefordert einen Titel für das neue Layout auszuwählen.

Um zu zeigen wie man eine Karte erstellt folgen Sie den nächsten Anweisungen.

1. Wählen Sie den Neue Karte hinzufügen Werkzugleistenknopf aus und zeichnen Sie ein Rechteck auf die Seite indem Sie die linke Maustaste gedrückt halten. Innerhalb des gezeichneten Rechteckes erscheint die QGIS Kartenansicht auf der Seite.
2. Wählen Sie die Schaltfläche Neuen Maßstab hinzufügen in der Werkzeugleiste und klicken Sie mit links in die Zeichenfläche der Druckzusammenstellung. Der Zeichenfläche wird ein Maßstab hinzugefügt.
3. Wähllen Sie den Neue Legende hinzufügen Werkzeugleistenknopf und zeichnen Sie ein Rechteck auf die Seite indem sie die linke Maustaste gedrückt halten. Innerhalb des Rechtecks wird die Legende gezeichnet.
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. Klicken Sie den Reiter *Elementeigenschaften* im linken unteren Bedienfeld und suchen Sie die Einstellung für die Kartendrehung. Ändern Sie den Wert der Einstellung *Kartendrehung* auf ‚15.00°'. Sie sollten sehen, wie der Drehwinkel des Kartenelements sich ändert.
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.

Sie können dem Drucklayout mehrere Elemente hinzufügen. Es ist auch möglich, dass es mehr als eine Karten-, Legenden- oder Maßstabselement enthält, und das auf einer oder mehreren Seiten. Jedes Element hat seine eigenen Eigenschaften und, im Fall der Karte, seine eigenen Ausmaße. Wenn Sie ein Element aus der Zusammenstellungsansicht entfernen wollen, können Sie das mit den Tasten `Entf` oder `Rücktaste` tun.

17.1.2 Layoutverwaltung

Die *Layoutverwaltung* ist das Hauptfenster, um Drucklayouts innerhalb des Prohjekts zu verwalten. Sie gibt Ihnen eine Übersicht über die vorhandenen Layouts und Berichte im Projekt und bietet folgende Werkzeuge:

- 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;
- 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 Layouverwaltung in der *Layout-Werkzeugleiste*.

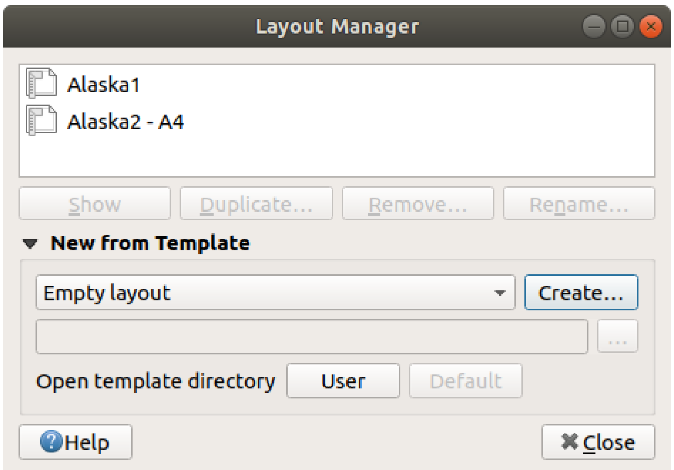


Abb. 17.1: Die Layout-Verwaltung

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.

Im unteren Teil können Sie neue Drucklayouts oder Berichte erzeugen und dabei eine Vorlage benutzen. In der Voreinstellung wird QGIS die Vorlagen im Nutzerprofil und den Vorlagenverzeichnissen der Anwendung suchen (sie sind mit den beiden Schaltflächen unten im Fenster zugänglich) aber auch in jedem Verzeichnis, das in *Einstellungen Optionen Layouts* als *Pfad(e) um nach zusätzlichen Druckvorlagen zu suchen* eingetragen wurde. Alle gefundenen Vorlagen werden im Auswahlfeld angezeigt. Wählen Sie eine aus und drücken Sie auf die Schaltfläche *Erzeugen…*, um einen neuen Bericht oder ein neues Drucklayout zu erzeugen.

Sie können auch Vorlagen aus einem eigenen Verzeichnus 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.

17.1.3 Menüs, Werkzeuge und Bedienfelder des Drucklayouts

Das Öffnen des Drucklayouts bietet Ihnen eine leere Zeichenfläche, die den ausgedruckten Papierbogen darstellt. In der Starteinstellungen finden Sie links neben der Zeichenfläche Schaltflächen, um Elemente des Drucklayouts hinzuzufügen; die aktuelle QGIS Kartenansicht, Beschriftungen, Bilder, Legenden, Maßstäbe, einfache Formen, Pfeile, Attributtabellen und HTML-Rahmen. In dieser Werkzeugleiste finden Sie auch Schaltflächen zum Navigieren, Zoomen in einen Bereich und zum Verschieben der Ansicht im Layout sowie Schaltflächen, um ein Element auszuwählen und den Inhalt des Kartenelements zu verschieben.

figure_layout_overview zeigt die Starteinstellung des Drucklayouts bevor Elemente hinzugefügt wurden.

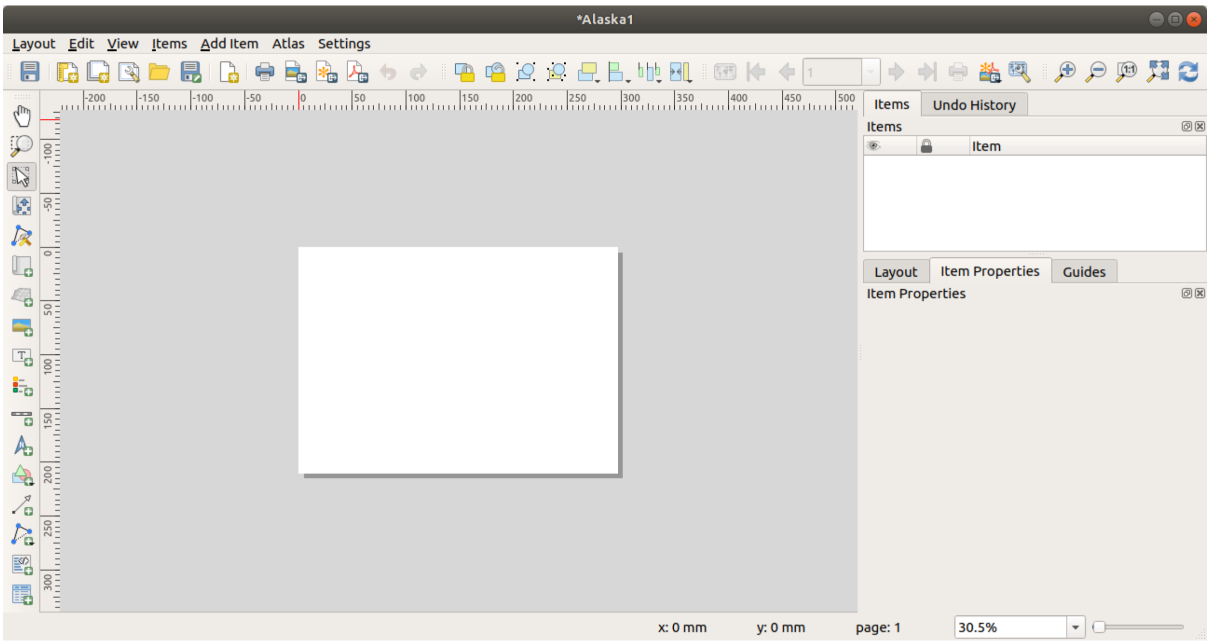


Abb. 17.2: Drucklayout

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).
- Der Reiter *Rücknahmeprotokoll* Reiter stellt den Verlauf aller Änderungen, die am Drucklayout durchgeführt wurden, dar. Mit einem Mausklick ist es möglich, Layoutschritte vorwärts oder rückwärts bis zu einem bestimmten Status rückgängig zu machen oder wiederherzustellen.
- 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.

Die Werkzeugleisten und Reiter können an- und ausgeschaltet werden, indem Sie mit der rechten Maustaste in eine Werkzeugleiste klicken oder über das Menü *Ansicht ▸ Werkzeugleiste* oder *Ansicht▸ Bedienfelder*.

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.
- Lööschen 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 Indormationen, die mit QGIS erzeugt wurden, verfügbar zu machen, so dass sie in Berichte eingebunden oder publiziert werden können. Diese Werkzeug sind *Als Bild exportieren…*, *Als PDF exportieren…*, *Als SVGexportieren…* und *Drucken…*.

Unten ist eine Liste aller verfügbaren Werkzeuge in diesem Menü mit einigen hilfreichen Informationen.

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
Projekt speichern	`Strg+S`	*Layout*	*Vorstellung von QGIS-Projekten*
Neues Layout	`Strg+N`	*Layout*	*Layoutverwaltung*
Layout duplizieren		*Layout*	*Layoutverwaltung*
Layout löschen			
Layout-Verwaltung…		*Layout*	*Layoutverwaltung*
Layouts ⍰			
Layouteigenschaften…			*The Layout Panel*
Layout umbenennen…			
Seiten hinzufügen…		*Layout*	*Working with the page properties*
Elemente aus Vorlage hinzufügen		*Layout*	*Creating a layout item*
Als Vorlage speichern…		*Layout*	*Layoutverwaltung*
Als Bild exportieren…		*Layout*	*Speichern als Rasterbild*
Als SVG exportieren…		*Layout*	*Als SVG exportieren*
Als PDF exportieren…		*Layout*	*Als PDF exportieren*
Seiteneinstellungen…	`Strg+Umschalt+P`		
Drucken…	`Strg+P`	*Layout*	*Eine Ausgabe erzeugen*
Schließen	`Strg+Q`		

Menü Bearbeiten

Das Menü *Bearbeiten* bietet Ihnen Werkzeuge, um Drucklayouts zu bearbeiten. Das beinhaltet gägnige 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. 17.1: Available Tools

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
Rückgängig (letze Änderung)	Strg+Z	*Layout*	*The Undo History Panel: Revert and Restore actions*
Wiederherstellen (der letzten rückgängig gemachten Änderung)	Strg+Y	*Layout*	*The Undo History Panel: Revert and Restore actions*
Löschen	Entf		
Ausschneiden	Strg+X		
Kopieren	Strg+C		
Einfügen	Strg+V		
Durch Einfügen ersetzen	Strg+Shift+V		
Alles wählen	Strg+A		
Alles abwählen	Strg+Umschalt+A		
Auswahl umkehren			
Nächstes Elemente darunter wählen	Strg+Alt+[		
Nächstes Elemente darüber wählen	Strg+Alt+]		
Layout verschieben	P	*Werkzeugkiste*	
Zoom	Z	*Werkzeugkiste*	
Eintrag wählen/verschieben	V	*Werkzeugkiste*	*Interacting with layout items*
Inhalt verschieben	C	*Werkzeugkiste*	*Das Kartenelement*
Knotenelement bearbeiten		*Werkzeugkiste*	*The Node-Based Shape Items*

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
Refresh	`F5`	*Navigation*	
Preview ▸			
Hineinzoomen	`Ctrl++`	*Navigation*	
Herauszoomen	`Ctrl+-`	*Navigation*	
Zoom to 100%	`Ctrl+1`	*Navigation*	
Volle Ausdehnung	`Ctrl+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	`Ctrl+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
Group	`Ctrl+G`	*Actions*	*Grouping items*
Ungroup	`Ctrl+Shift+G`	*Actions*	*Grouping items*
Raise	`Ctrl+]`	*Actions*	*Ausrichtung*
Lower	`Ctrl+[`	*Actions*	*Ausrichtung*
Bring to Front	`Ctrl+Shift+]`	*Actions*	*Ausrichtung*
Send to Back	`Ctrl+Shift+[`	*Actions*	*Ausrichtung*
Lock Selected Items	`Ctrl+L`	*Actions*	*Locking items*
Unlock All	`Ctrl+Shift+L`	*Actions*	*Locking items*
Align Items ⍰		*Actions*	*Ausrichtung*
Distribute Items ⍰		*Actions*	*Moving and resizing items*
Resize ⍰		*Actions*	*Moving and resizing items*

Add Item menu

These are tools to create layout items. Each of them is deeply described in *Layout-Elemente* chapter.

Werkzeug	Werkzeugleiste	Referenz
Add Map	*Werkzeugkiste*	*Das Kartenelement*
Add Picture	*Werkzeugkiste*	*Das Bildelement*
Add Label	*Werkzeugkiste*	*Das Beschriftungselement*
Add Legend	*Werkzeugkiste*	*Das Legendenelement*
Add Scale Bar	*Werkzeugkiste*	*Das Maßstabselement*
Add North Arrow	*Werkzeugkiste*	*The North Arrow Item*
Add Shape ⍰	*Werkzeugkiste*	*The Regular Shape Item*
Add Arrow	*Werkzeugkiste*	*The Arrow Item*
Add Node Item ⍰	*Werkzeugkiste*	*The Node-Based Shape Items*
Add HTML	*Werkzeugkiste*	*Das HTML-Rahmen Element*
Add Attribute Table	*Werkzeugkiste*	*Das Attributtabellenelement*

Atlas menu

Werkzeug	Tastenkürzel	Werkzeugleiste	Referenz
Preview Atlas	`Ctrl+ALt+/`	*Atlas*	*Preview and generate an atlas*
First Feature	`Ctrl+<`	*Atlas*	*Preview and generate an atlas*
Previous Feature	`Ctrl+,`	*Atlas*	*Preview and generate an atlas*
Next Feature	`Strg+.`	*Atlas*	*Preview and generate an atlas*
Last feature	`Ctrl+>`	*Atlas*	*Preview and generate an atlas*
Print Atlas…		*Atlas*	*Preview and generate an atlas*
Export Atlas as Images…		*Atlas*	*Preview and generate an atlas*
Export Atlas as SVG…		*Atlas*	*Preview and generate an atlas*
Export Atlas as PDF…		*Atlas*	*Preview and generate an atlas*
Atlas Settings		*Atlas*	*Einen Atlas erzeugen*

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.

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.

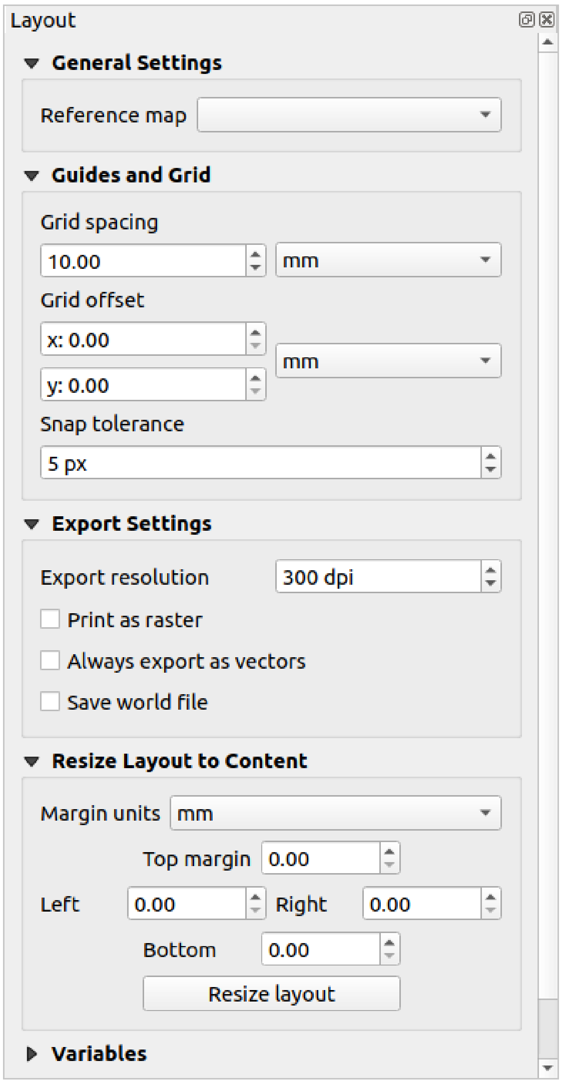


Abb. 17.3: Layout Settings in the Print Layout

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 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.

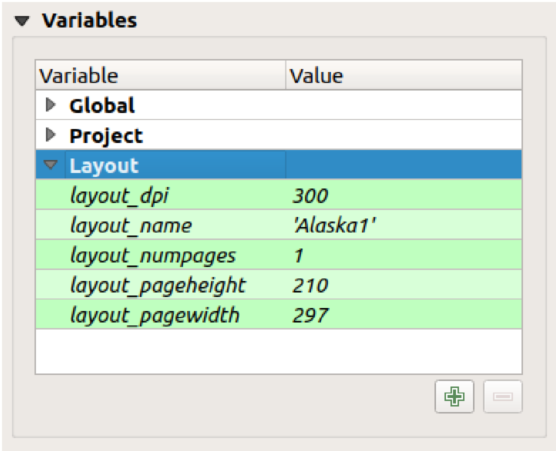


Abb. 17.4: Variables Editor in the Print Layout

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.

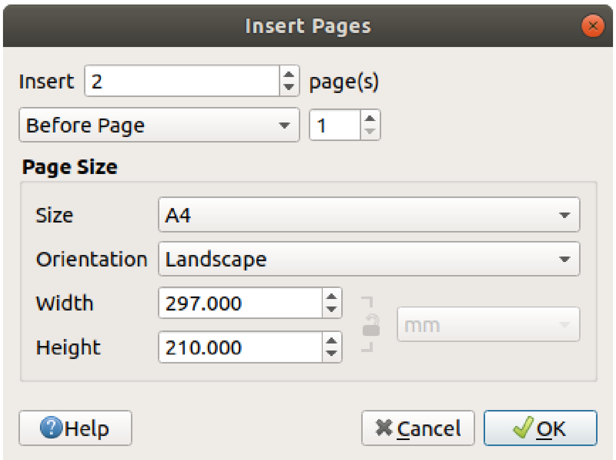


Abb. 17.5: Creating a new page in the Print Layout

Updating page properties

Any page can be later customized through the Page *Item Properties* panel. Right-click on a page and select *Page Properties…*. The *Item Properties* panel opens with settings such as:

- 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.

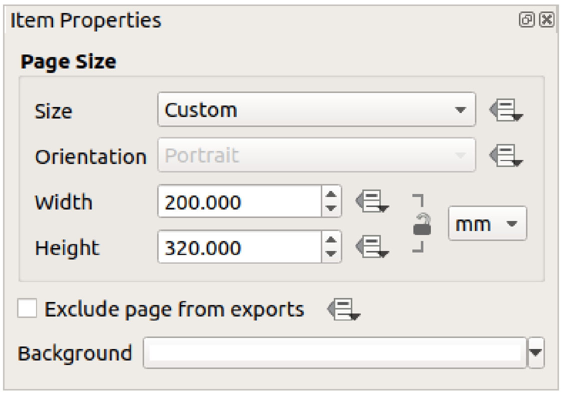


Abb. 17.6: Page properties dialog

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.

The *Guides* panel allows creation of snap lines at specific locations:

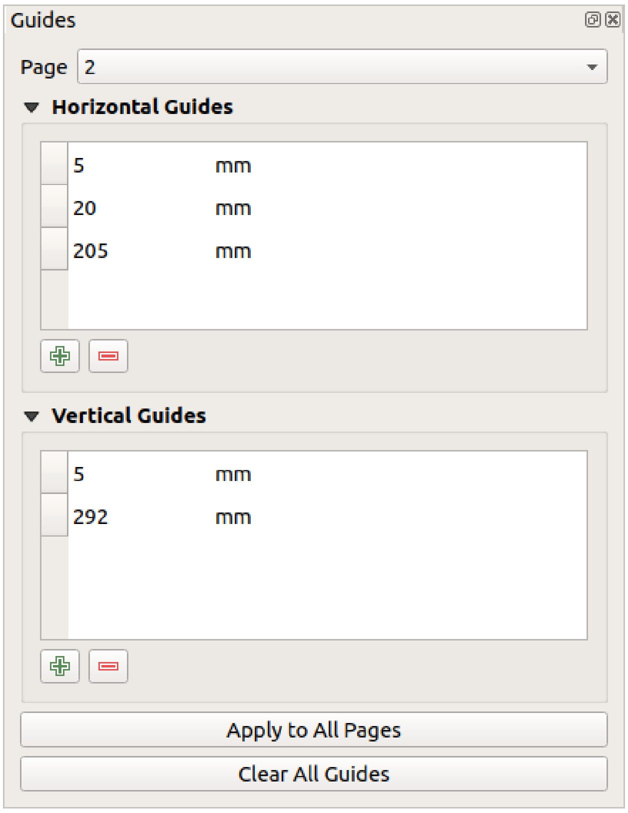


Abb. 17.7: The Guides panel

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.

Für irgendein ausgewähltes Objekt, können Sie:

- 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 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 *figure_layout*). The History panel lists the last actions done within the print layout. Just select the point you want to revert to and once you do new action all the actions done after the selected one will be removed.

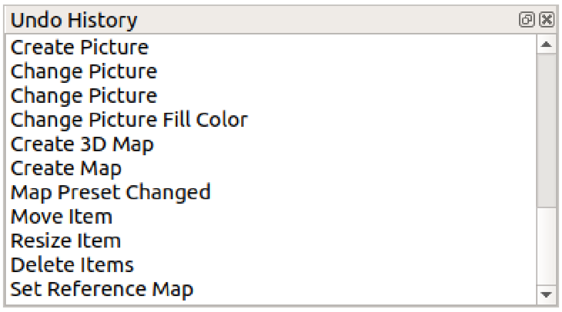


Abb. 17.8: Undo History in the Print Layout

17.2 Layout-Elemente

17.2.1 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*);

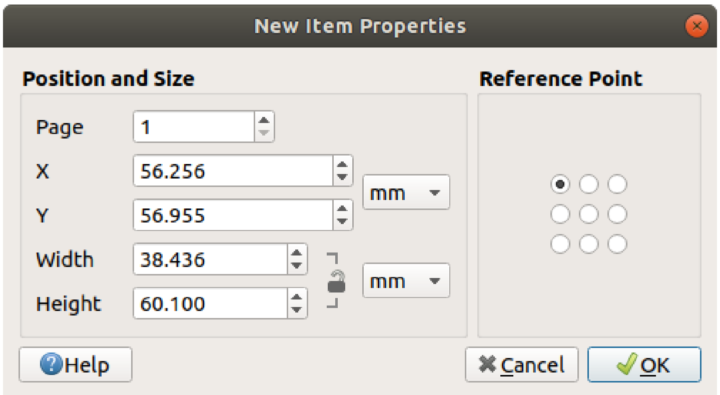


Abb. 17.9: New Item properties dialog

   - 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 *The Node-Based Shape Items* 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.

Tipp: 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, just 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 form a square.

Likewise, automated tools are available to organize multiple items position by distributing equidistantly:

- edges (left, right, top or bottom) of items;
- centers of items either 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 *View* 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.

Ausrichtung

Raising or lowering the visual hierarchy for elements 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 compared to 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.

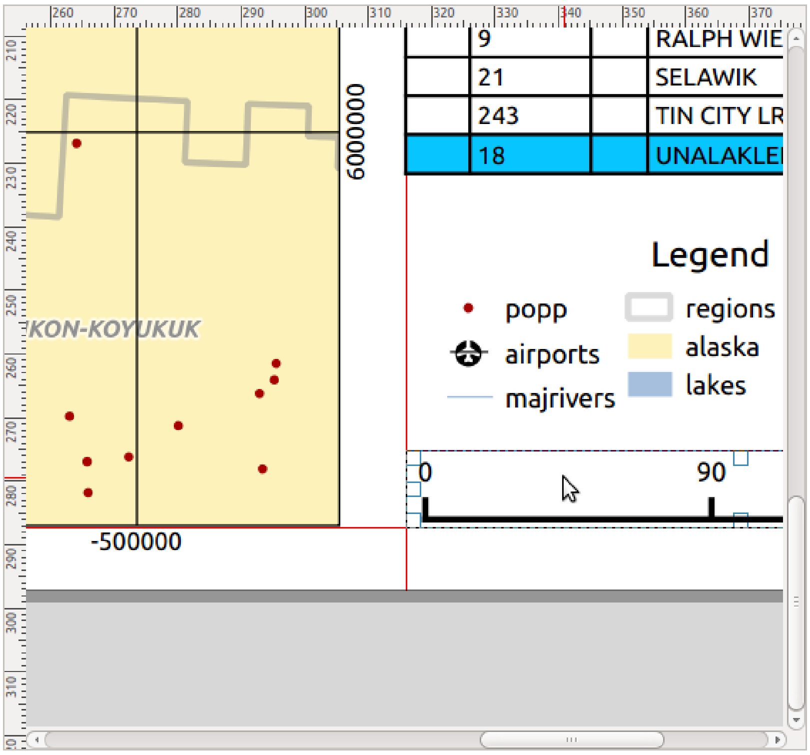


Abb. 17.10: Alignment helper lines in the print layout

There are several alignment options available within the Align selected items pull-down menu (see *figure_layout_common_align*). 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.

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 *figure_layout_common*).

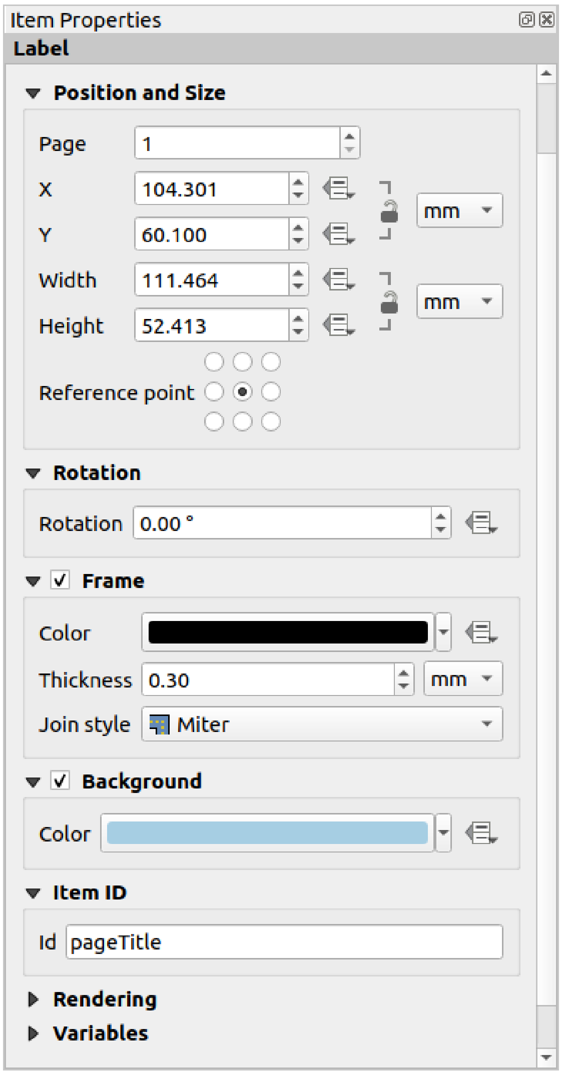


Abb. 17.11: Common Item Properties groups

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.
- Use the *Background color* menu for setting a background color. Click on the [Color…] button to display a dialog where you can pick a color or choose from a custom setting. 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.

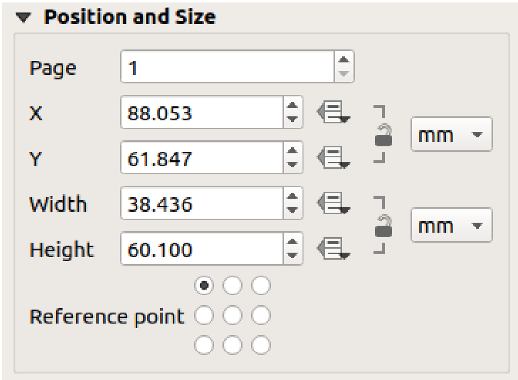


Abb. 17.12: Position and size

- 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.

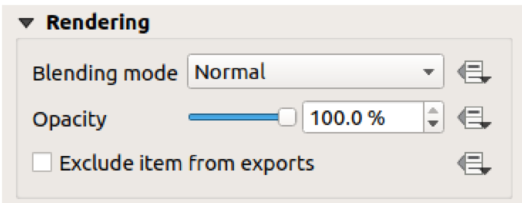


Abb. 17.13: Darstellung

- *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).

- *Transparency* : 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.
- *Element aus Export ausnehmen*: Sie können sich entschließen ein Element in allen Exporten nicht sichtbar zu machen. Nach dem Aktivieren dieses Kontrollkästchens wird das Element nicht in PDF's, Drucken etc. enthalten sein

Variables

The *Variables* lists all the variables available at the layout item's level (which includes all global, project and composition's variables). Map items also include Map settings variables that provide easy access to values like the map's scale, extent, and so on.

In *Variables*, it's also possible to manage 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.

17.2.2 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:

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.

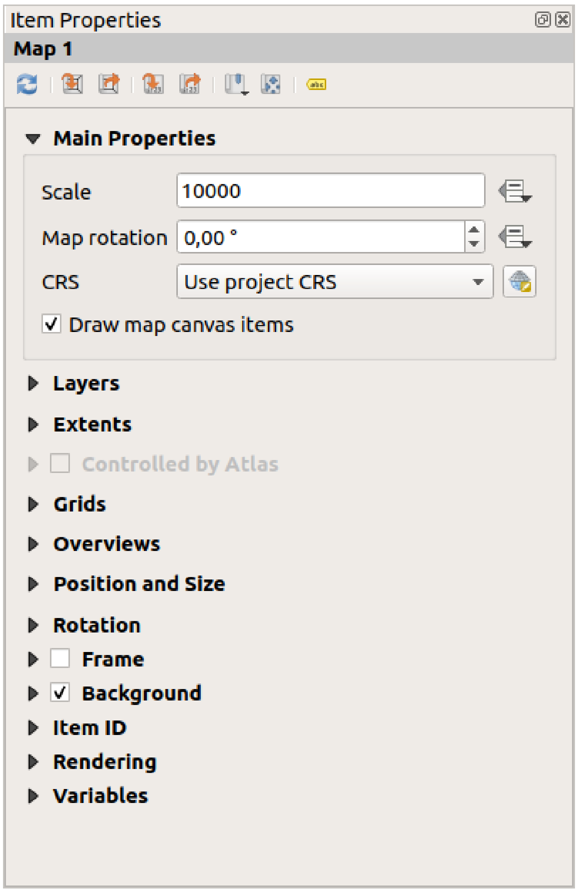


Abb. 17.14: Map Item Properties Panel

- *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*.

Haupteigenschaften

In the *Main properties* group (see *figure_layout_map*) 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 *figure_layout_map_layers*) helps you do that.

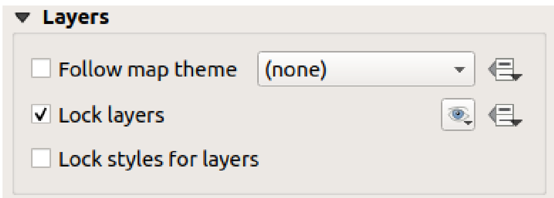


Abb. 17.15: Map Layers group

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 (symbology, 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 *figure_layout_map_extents*):

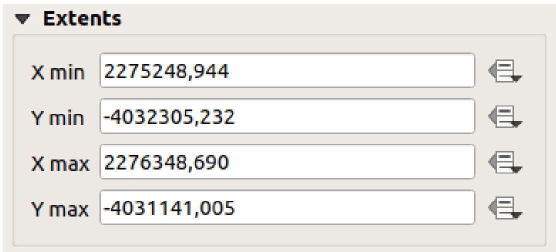


Abb. 17.16: Map Extents group

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. Clicking the *Set to Map Canvas Extent* button sets the extent of the layout map item to the extent of the main map canvas. The button *View Extent in Map Canvas* does exactly the opposite; it updates the extent of the main map canvas to the extent of the layout map item.

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.

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.

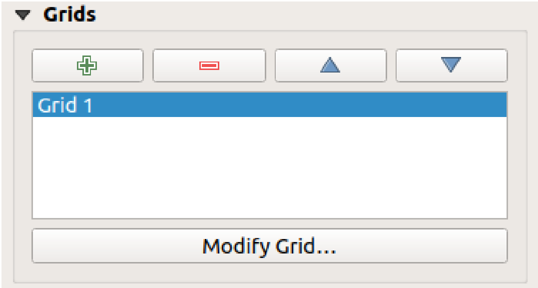


Abb. 17.17: Kartengitterdialog

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. If not changed, it will follow the Map CRS. The *Change* button lets you set it to a different CRS. Once set, it can be changed back to default by selecting any group heading (e.g **Geographic Coordinate System**) under *Predefined Coordinate Reference Systems* in the CRS selection dialog.
- the *Interval* type to use for the grid references. Available options are `Map Unit`, `Fit Segment Width`, `Millimeter` or `Centimeter`:
  - choosing `Fit Segment Width` will dynamically select the grid interval based on the map extent to a „pretty" interval. When selected, the `Minimum` and `Maximum` intervals can be set.
  - the other options allow you to set the distance between two consecutive grid references in the `X` and `Y` directions.
- 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.

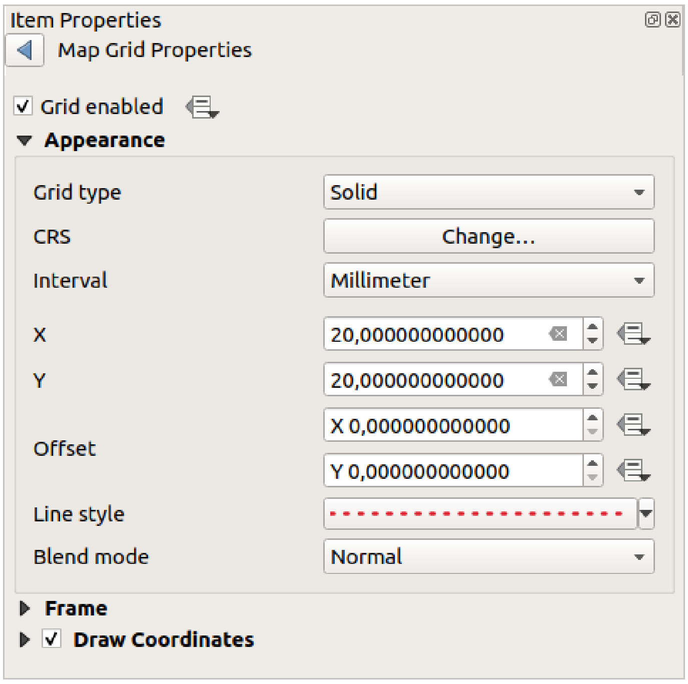


Abb. 17.18: Grid Appearance Dialog

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.

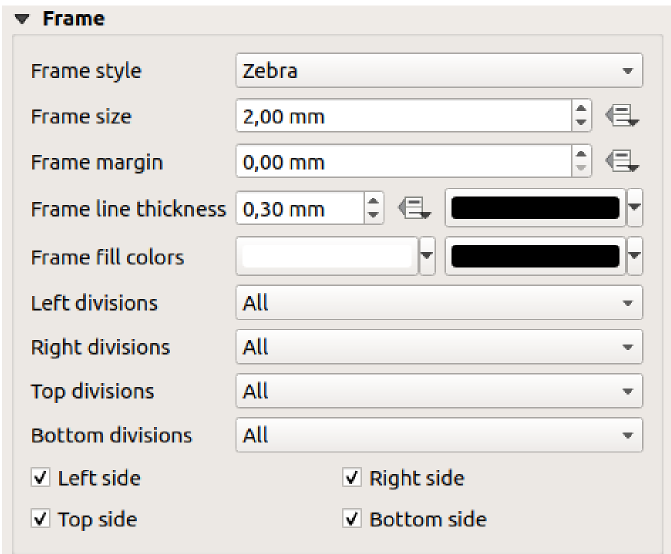


Abb. 17.19: Gitterrahmendialog

Coordinates

The *Draw coordinates* checkbox allows you to add coordinates to the map frame. 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.

You can choose which annotation to show. The options are: show all, latitude only, longitude only, or disable(none). This is useful when the map is rotated. The annotation can be drawn inside or outside the map frame. The annotation direction can be defined as horizontal, vertical ascending or vertical descending.

Finally, you can define the annotation font, font color, distance from the map frame and the precision of the drawn coordinates.

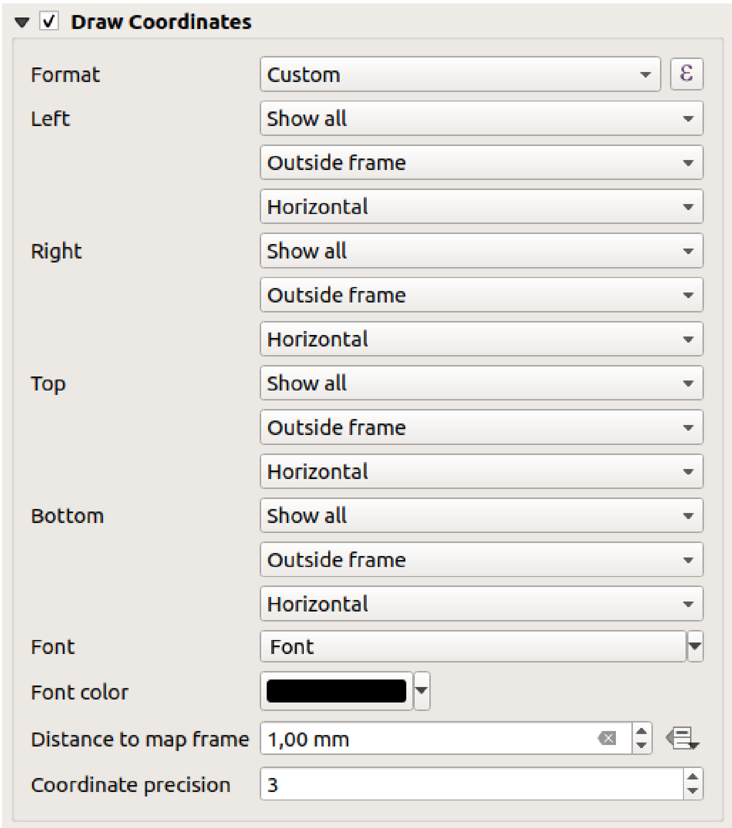


Abb. 17.20: Gitter-Koordinaten-Zeichnen-Dialog

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:

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 *Figure_layout_map_overview*). You can:

- Rename it with a double-click
- With the and buttons, add or remove overviews

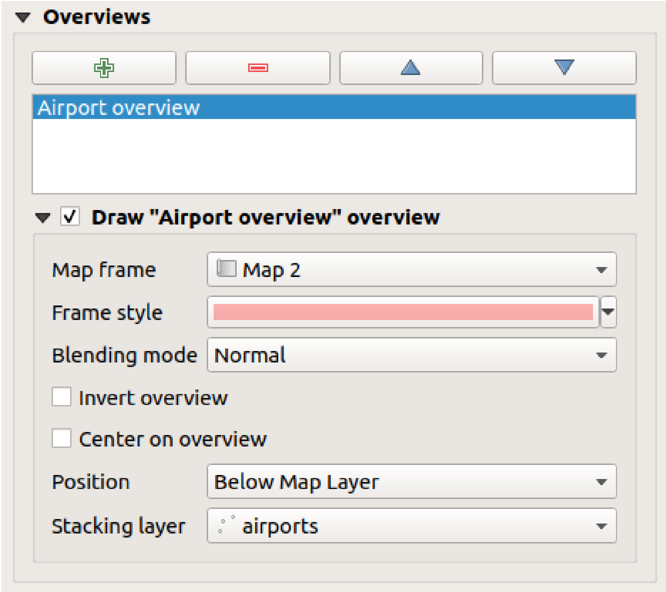


Abb. 17.21: Map Overviews group

- 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.

17.2.3 The 3D Map Item

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. 17.22):

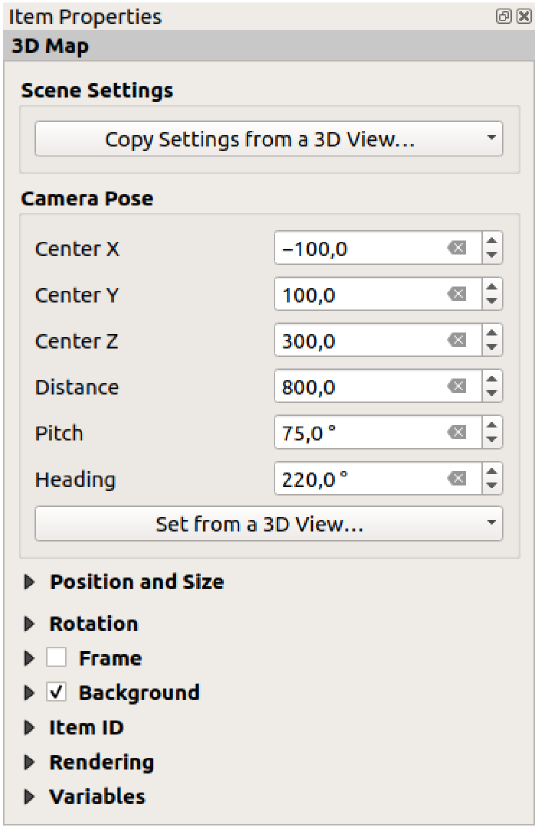


Abb. 17.22: 3D Map Item Properties

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.

17.2.4 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 *figure_layout_label*):

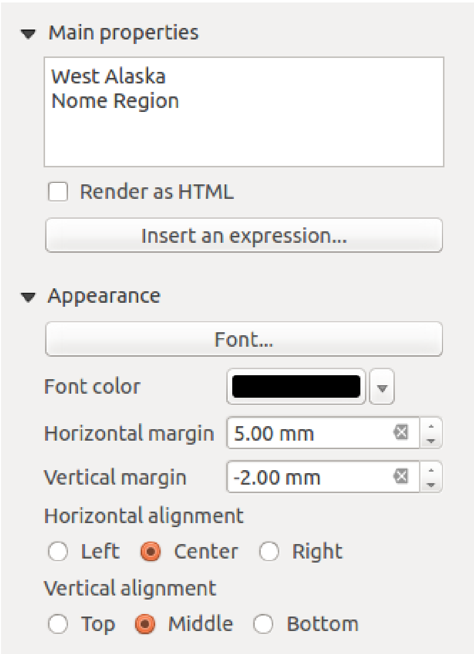


Abb. 17.23: Label Item Properties Panel

Haupteigenschaften

The *Main properties* group is the place to provide the text (it can be in HTML) or the expression to build the label. Expressions need to be surrounded by `[%` and `%]` in order to be interpreted as such.

- Beschriftungen können als HTML-Code interpretiert werden: aktivieren Sie *Als HTML darstellen*. Sie können jetzt eine URL, ein klickbares Bild, das zu einer Webseite führt oder etwas komplexeres eingeben.
- You can also use *expressions*: click on *Insert an expression* button, write your formula as usual and when the dialog is applied, QGIS automatically adds the surrounding characters.

Bemerkung: Clicking the *Insert an Expression* button when no selection is made in the textbox will append the new expression to the existing text. If you want to update an existing text, you need to select it the part of interest beforehand.

You can combine HTML rendering and expressions, leading to e.g. a text like:

```
[% '<b>Check out the new logo for ' || '<a href="https://www.qgis.org" title="Nice␣
↪logo" target="_blank">QGIS ' ||@qgis_short_version || '</a>' || ' : <img src=
↪"https://qgis.org/en/_downloads/qgis-icon128.png" alt="QGIS icon"/>' %]
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

which will render: **Check out the new logo for** QGIS 3.0 **:**

Darstellung

- Define *Font* by clicking on the *Font…* button or a *Font color* by pushing the *color widget*.
- 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 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
  )
  ```

17.2.5 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 *figure_layout_legend*):

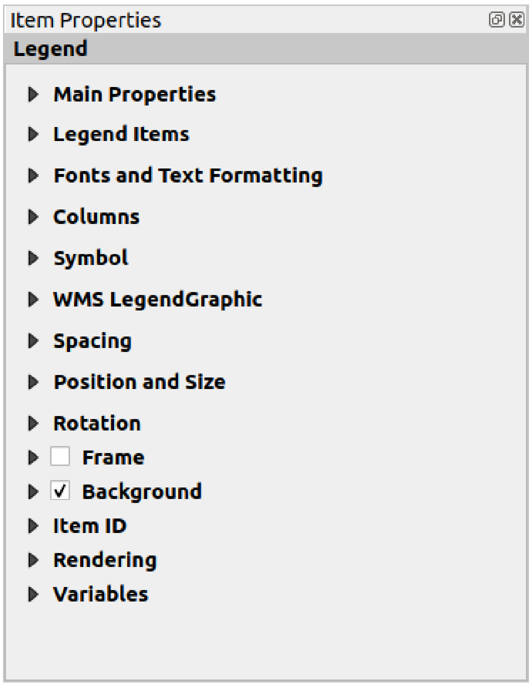


Abb. 17.24: Legend Item Properties Panel

Haupteigenschaften

The *Main properties* group of the legend *Item Properties* panel provides the following functionalities (see *figure_layout_legend_ppt*):

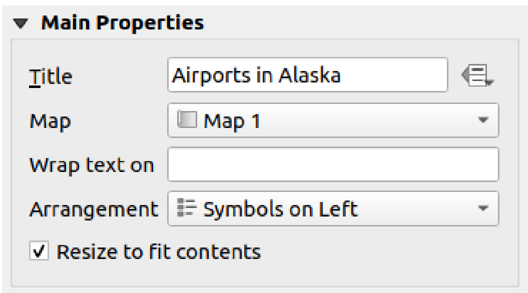


Abb. 17.25: Legend Main properties group

In den Haupteigenschaften können Sie:

- Change the *Title* of the legend. It can be made dynamic using the *data-defined override* setting, useful for example when generating an atlas;
- 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 *figure_layout_legend_items*):

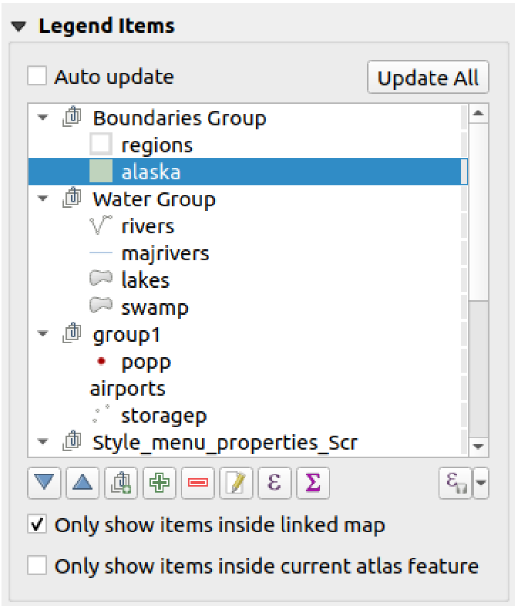


Abb. 17.26: Legend Items group

- 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.
- Das Legendenelementfenster führt alle Legendenelement auf und Sie können die Elementreihenfolge ändern, Layer gruppieren und Elemente in der Liste wiederherstellen, Namen bearbeiten und einen Filter hinzufügen.
  - 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.

– 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 categorized layer, you can append to each class in the legend their number of features, ie *class (number)*:

  1. Select the layer entry in the legend tree
  2. Press the button, opening the *Expression String Builder* dialog
  3. Enter the following expression:

     ```
     concat( @symbol_label, ' (', @symbol_count, ')' )
     ```

  4. Press *OK*

– 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 map*, only the legend items visible in the linked map will be listed in the legend. 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.

Fonts

The *Fonts* group of the legend *Item Properties* panel provides the following functionalities:

- You can change the font of the legend title, group, subgroup and item (feature) in the legend item using the *font selector* widget
- 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).
- You set the *Color* of the labels using the *color selector* widget. The selected color will apply to all the font items in the legend.

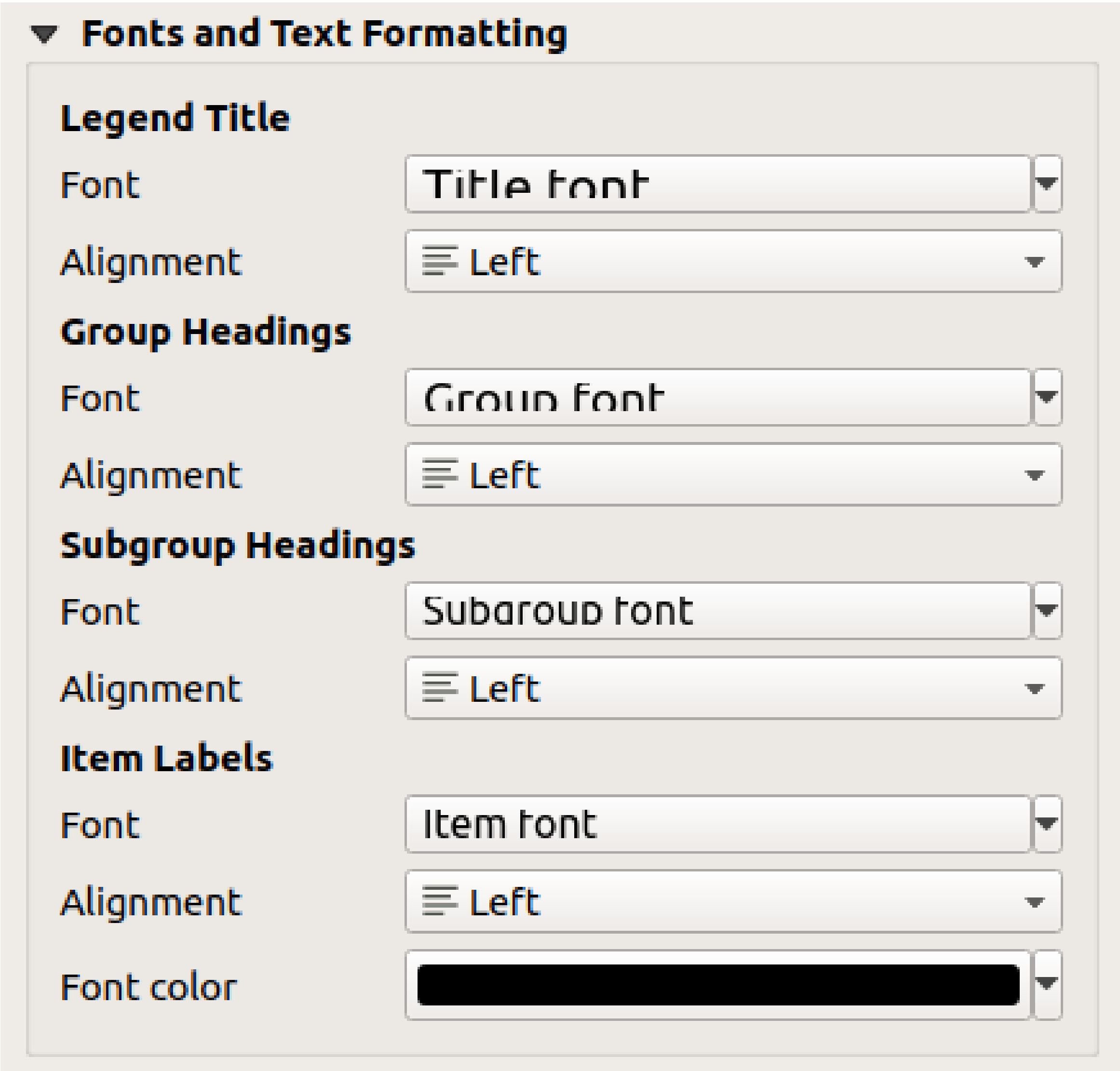


Abb. 17.27: Legend Fonts properties

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* 1,00 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.

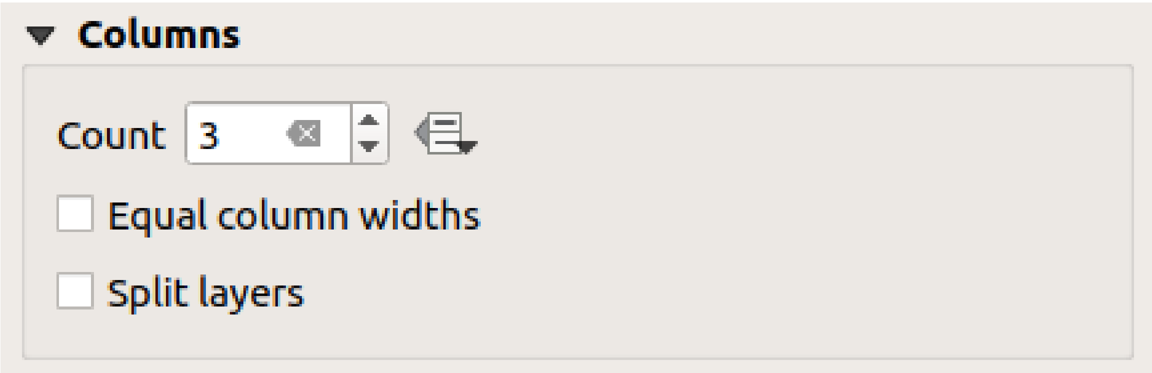


Abb. 17.28: Legend Columns settings

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*
- *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 *Tickness*.

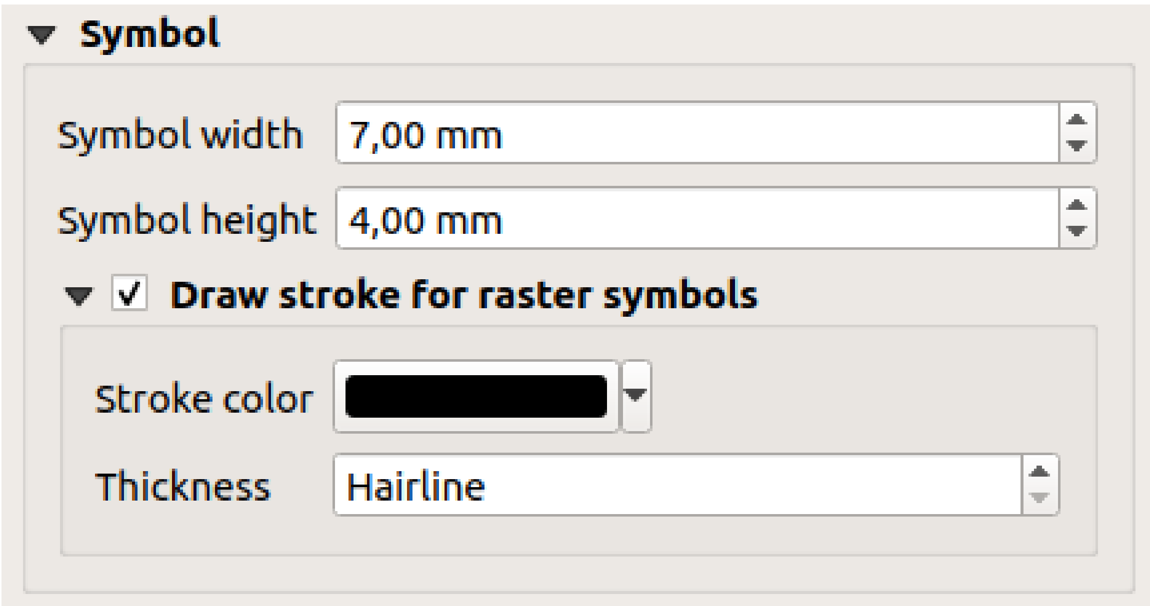


Abb. 17.29: Legend Symbol configuration

WMS-LegendGraphic und Zwischenräume

The *WMS LegendGraphic* and *Spacing* groups of the legend *Item Properties* panel provide the following functionalities (see *figure_layout_legend_wms*):

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 Lengendenrasterbildes anzupassen.

Spacing around title, groups, subgroups, symbols, labels, boxes, columns and lines can be customized through this dialog.

17.2.6 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 *figure_layout_scalebar*):

Haupteigenschaften

The *Main properties* group of the scale bar *Item Properties* panel provides the following functionalities (see *figure_layout_scalebar_ppt*):

1. First, choose the map the scale bar will be attached to
2. Wählen Sie dann den Stil des Maßstabs. Sechs Stile stehen zur Verfügung:
   - **Single box** and **Double box** styles, which contain one or two lines of boxes alternating colors;
   - **Middle**, **Up** or **Down** line ticks;
   - **Numeric**, where the scale ratio is printed (e.g., `1:50000`).
3. Set properties as appropriate

Einheiten und Segmente

The *Units* and *Segments* groups of the scale bar *Item Properties* panel (not available for the **Numeric** style) provide the following functionalities (see *figure_layout_scalebar_units*):

In these two groups, you can set how the scale bar will be represented.

- 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**… which may force unit conversions.
- 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.
- Sie können definieren wie viele *Segmente* links und rechts vom Maßstab gezeichnet werden.
- You can set how long each segment will be (*Fixed width*), or limit the 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 the range set.

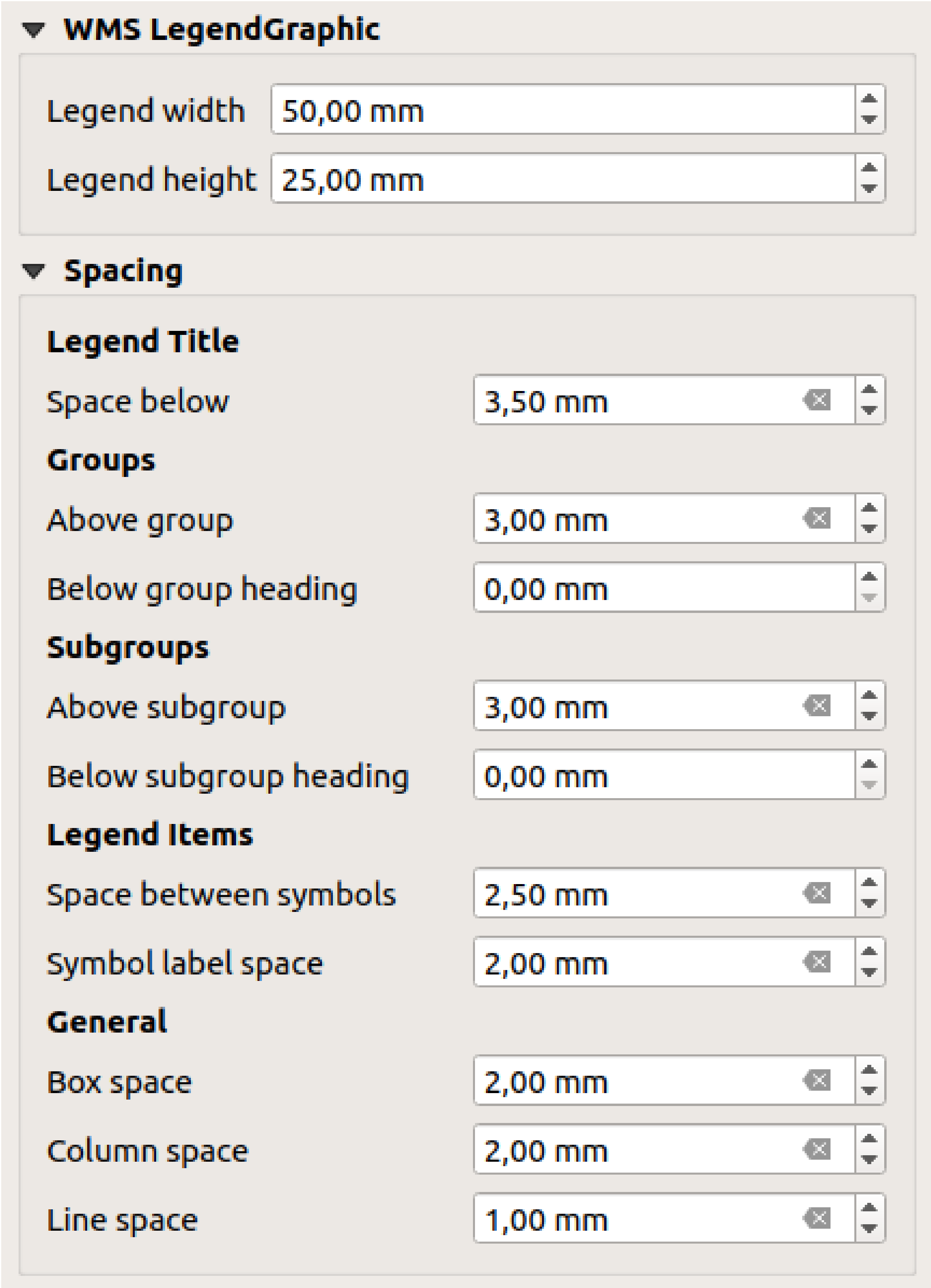


Abb. 17.30: WMS LegendGraphic and Spacing groups

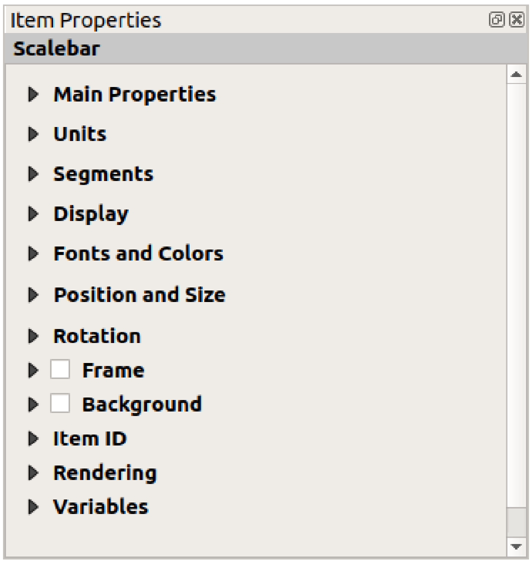


Abb. 17.31: Scale Bar Item Properties Panel

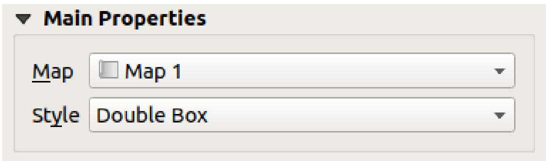


Abb. 17.32: Scale Bar Main properties group

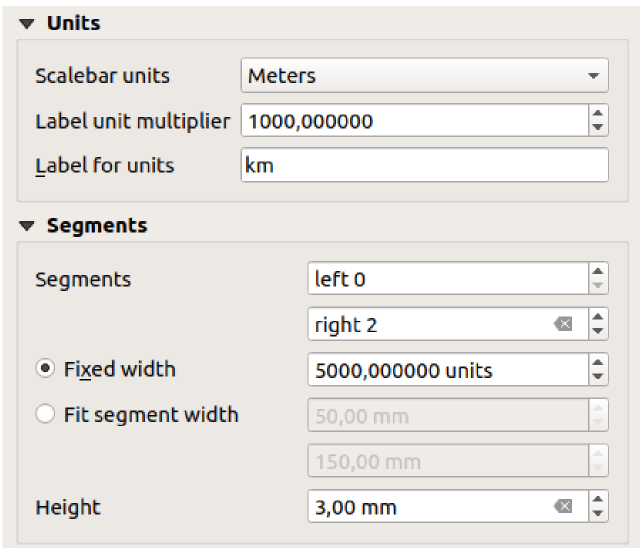


Abb. 17.33: Scale Bar Units and Segments groups

- *Höhe* wird benutzt, um die Höhe der Leiste einzustellen.

Anzeigen

The *Display* group of the scale bar *Item Properties* panel provides the following functionalities (see *figure_layout_scalebar_display*):

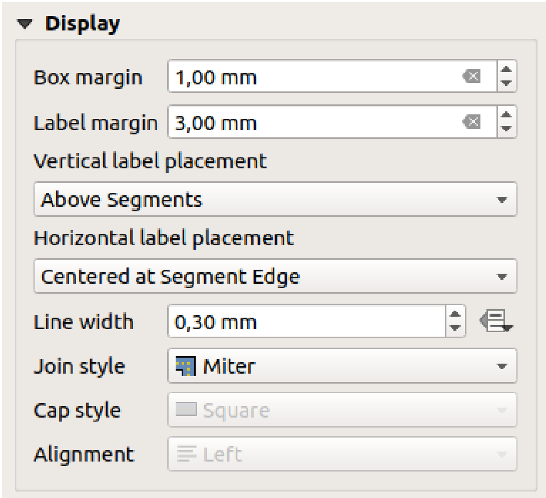


Abb. 17.34: Scale Bar Display group

Sie können festlegen wie der Maßstab in seinem Rahmen darsgestell 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
- *Linienbreite*: Linienbreite der Maßstabsdarstellung
- *Join style* : Corners at the end of scale bar in Bevel, Miter or Round style (only available for Scale bar style Single Box & Double Box)
- *Endenstil*: Das Ende aller Linien im Stil Quadratisch, Rund Flach (nur erhältlich für die Maßstabsstile Linieneinteilung Oben, Unten und Mittig)
- *Alignment* : Puts text on the left, center or right side of the frame (works only for Scale bar style Numeric)

Schriftarten und Farben

The *Fonts and colors* group of the scale bar *Item Properties* panel provides the following functionalities (see *figure_layout_scalebar_fonts*):

You can define the *fonts* and *colors* used for the scale bar. These properties are data-definable.

- Use the *Font* button to set the *properties* (size, font, color, letter spacing, shadow, background…) of the scale bar label.

  Example: The following code applied to the bold property of the scale labels will display texts 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
```

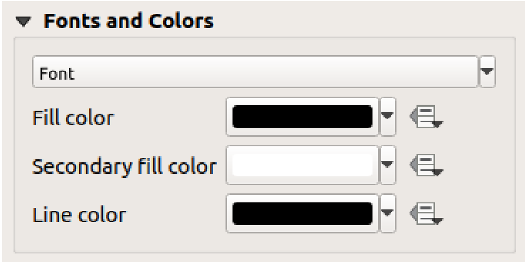


Abb. 17.35: Scale Bar Fonts and colors groups

- *Füllfarbe*: setzen Sie die erste Füllfarbe
- *Zweite Füllfarbe*: setzen Sie die zweite Füllfarbe
- *Line color*: set the color of the lines of the Scale Bar

Fill colors are only used for *Single Box* and *Double Box* styles.

17.2.7 Das Attributtabellenelement

Any layer in the project can have its attributes shown in the print layout. You can use this to decorate and explain your map with information about underlying data. Use the *Add Attribute Table* 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 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 *figure_layout_table*):

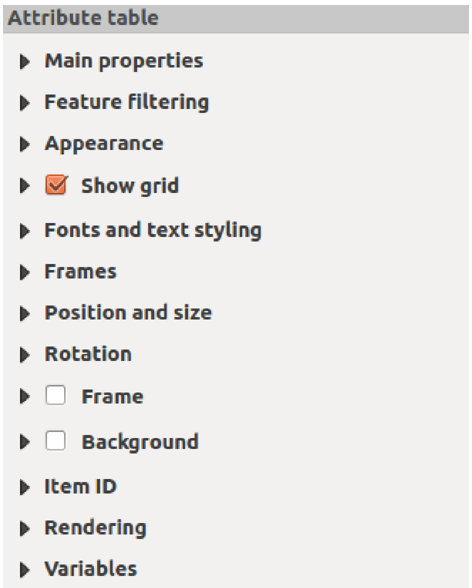


Abb. 17.36: Attribute table Item Properties Panel

Haupteigenschaften

The *Main properties* group of the attribute table provides the following functionalities (see *figure_layout_table_ppt*):

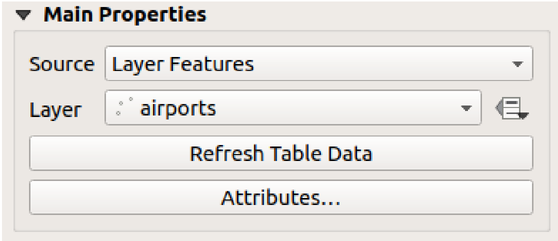


Abb. 17.37: Attribute table Main properties Group

- 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. 17.40) 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 *figure_layout_table_atlas*): 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 *figure_layout_table_relation*): 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.

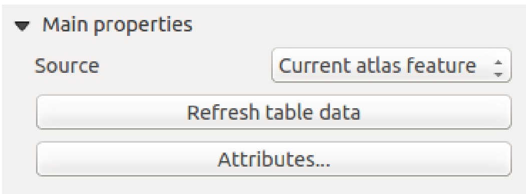


Abb. 17.38: Attribute table Main properties for ‚Current atlas feature'

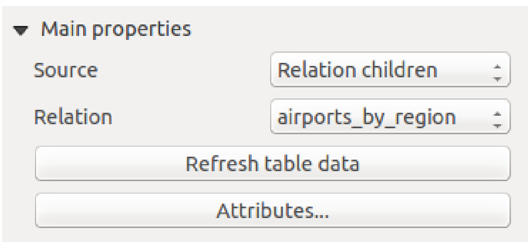


Abb. 17.39: Attribute table Main properties for ‚Relation children'

- The button *Attributes…* starts the *Select Attributes* dialog, (see *figure_layout_table_select*) 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.

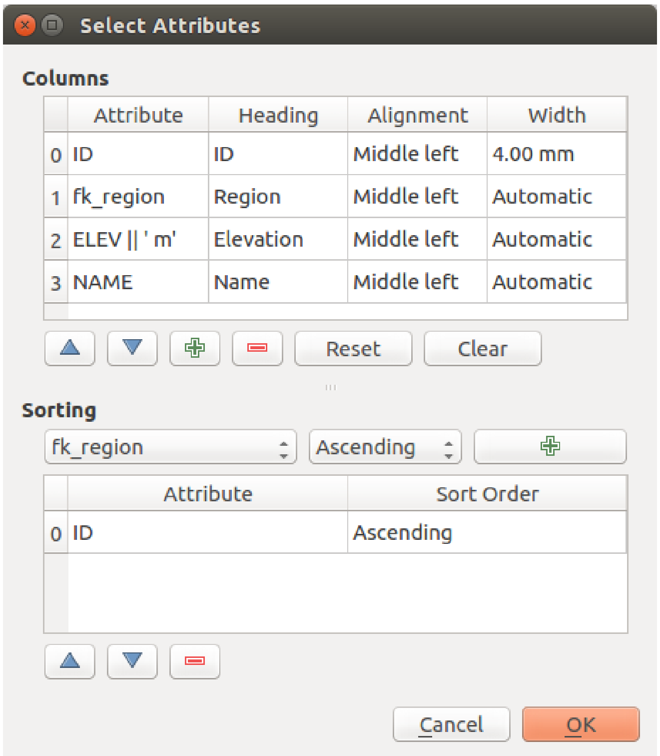


Abb. 17.40: Attribute table Select attributes Dialog

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. Select an attribute and set the sorting order to **Ascending** or **Descending** and press the button. A new line is added to the sort order list.
- 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 sort order list.

Objektfiltrierung

The *Feature filtering* group of the attribute table provides the following functionalities (see *figure_layout_table_filter*):

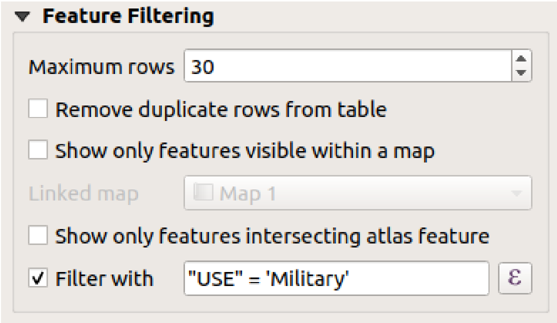


Abb. 17.41: Attribute table Feature filtering Group

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 *figure_layout_table_appearance*):

- 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.

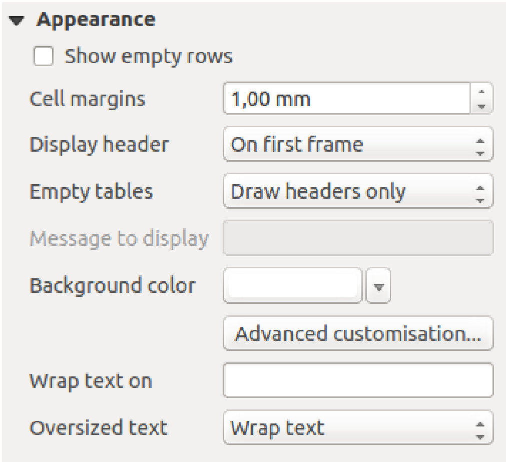


Abb. 17.42: Attribute table appearance Group

- **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 *figure_layout_table_background*)

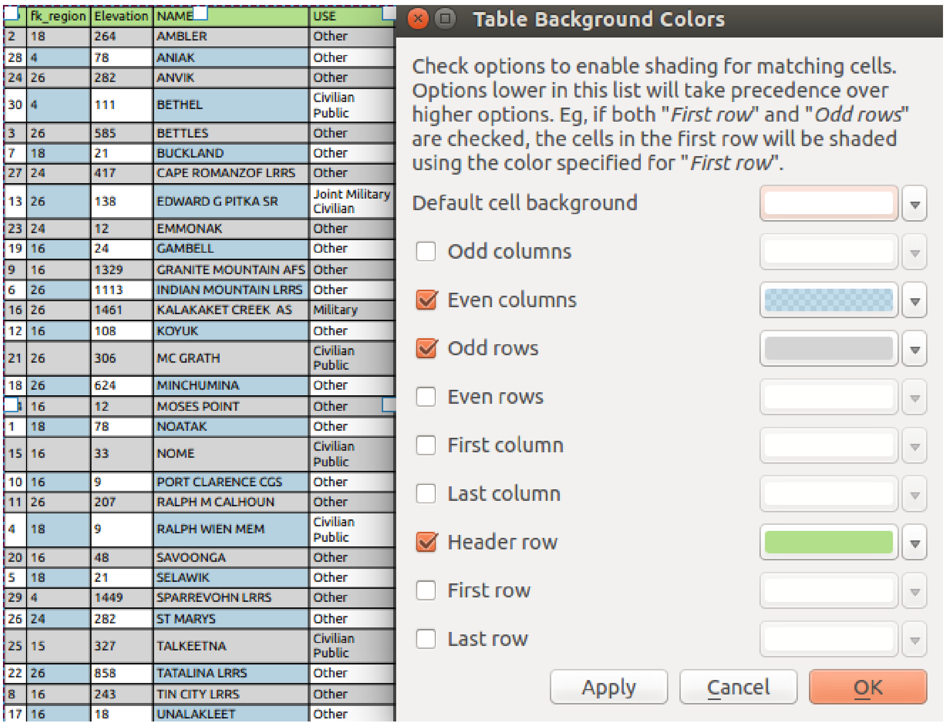


Abb. 17.43: Attribute table Advanced Background Dialog

- 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**.

Gitter anzeigen

The *Show grid* group of the attribute table provides the following functionalities (see *figure_layout_table_grid*):

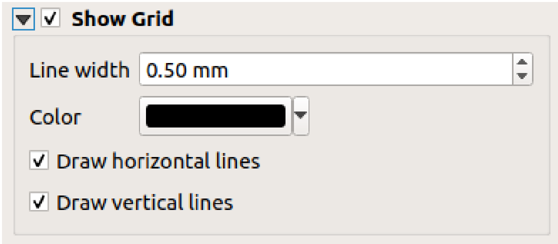


Abb. 17.44: Attribute table Show grid Group

- 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 attribute table provides the following functionalities (see *figure_layout_table_fonts*):

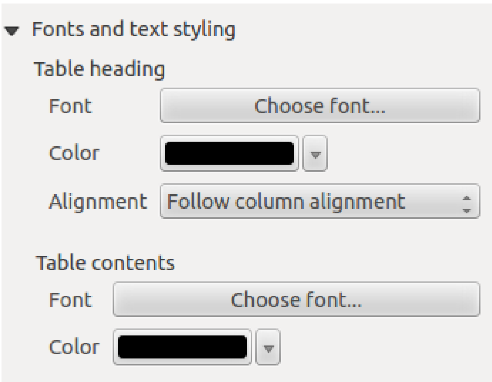


Abb. 17.45: Attribute table Fonts and text styling Group

- You can define *Font* and *Color* for *Table heading* and *Table contents*, using font and color selector widgets.
- 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 *figure_layout_table_select*).

Rahmen

The *Frames* group of the attribute table properties provides the following functionalities (see *figure_layout_table_frames*):

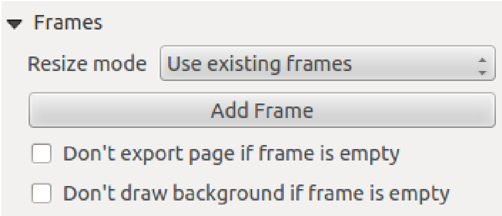


Abb. 17.46: Attribute table Frames Group

- 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.

17.2.8 The Picture and the North Arrow Items

The *Picture* item is a tool that helps decorate your map with pictures, logos… It can also be used to add north arrows, despite the dedicated *North arrow* tool.

Das Bildelement

You can add a picture with the Add Picture following *items creation instructions* and manipulate it the same way as exposed in *Interacting with layout items*.

Zunächst ist das Bildelement ein leerer Rahmen, den Sie über die *Elementeigenschaften* an Ihre Vorstellungen anpassen können. Abweichend von den *allgemeinen Elementeigenschaften*, gibt es folgende Funktionen (see *figure_layout_image*):

Zunächst müssen Sie das Bild, das Sie darstellen wollen, auswählen. Es gibt mehere Möglichkeiten die *Bildquelle* festzulegen:

1. In der Gruppe *Haupteigenschaften* benutzen Sie die Schaltfläche … durchsuchen der *Bildquelle*, um eine Datei auf Ihrem Computer auszuwählen. Die Suche startet in den SVG-Verzeichnissen, die mit QGIS zusammen installiert sind. Außer `SVG` können Sie auch andere Bildformate wie `.png` oder `.jpg` auswählen.
2. Sie können die Qelle direkt in das Eingabefeld *Bildquelle* eingeben. Sie können sogar die URL-Adresse eines Bildes eingeben.

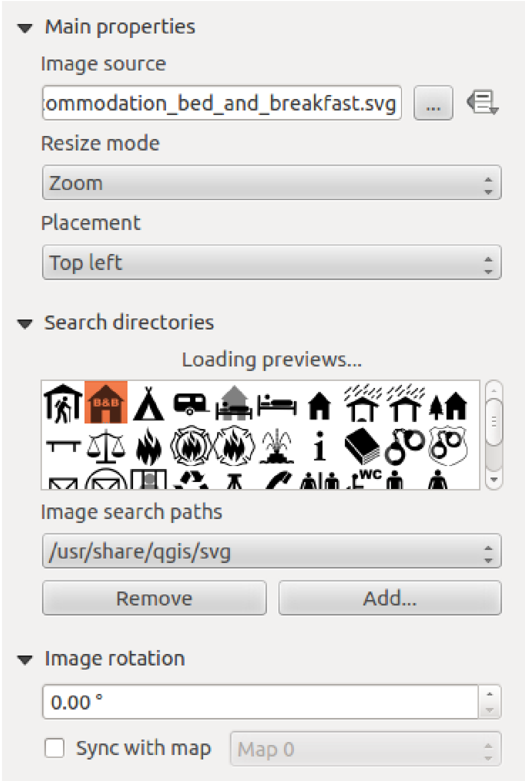


Abb. 17.47: Elementeigenschaften für Bilder

3. Aus dem Bereich *Verzeichnisse durchsuchen* können Sie eebnfalls ein Bild als Bildquelle auswählen. Diese Bilder werden per Voreinstellung aus den Verzeichnissen gelesen, die Sie im Menü :menuselection:`Einstellungen –> Optionen –> System –> SVG Pfade ` einstellen.
4. Benutzen Sie die Schaltfläche Datendefinierte Übersteuerung, um die Bildquelle aus einem Objektattribut zu lesen oder über einen Ausdruck einzustellen.

Bemerkung: In der Gruppe *Verzeichnisse durchsuchen* können Sie die Schaltflächen *Hinzufügen* und *Entfernen* benutzen, um die Liste der Verzeichnisse anzupassen, aus dernen Bilder geholt und in der Vorschau angezeigt werden.

Mit der Einstellung *Größenmodus* können Sie festlegen, wie das Bld dargestellt wird, wenn sich die Größe des Rahmens ändert:

- `Zoom`: vergröert/verkleinert das Bild passend zum Rahmen, während das Seitenverhältnis des Bildes beibehalten wird;
- `Strecken`: dehnt das Bild, so dass es in den Rahmen passt und ignoriert das Seitenverhältnis;
- `Zuschneiden`: benutzen Sie diese Einstellung ausschließlich für Rasterbilder. Es setzt die Bildgröße auf die Originalgröße und skaliert das Bild nicht, wobei der Rahmen dazu benutzt wird, das Bild abzuschneiden, so dass nur der Teil des Bildes, der sich im Rahmen befindet, sichtbar ist;
- `Rahmen zoomen und Größe anpassen`: vergrößert das Bild, so dass es in den Rahmen passt und verändert dann die Größe des Rahmens so, dass er zum sich ergebenden Bild passt;
- `Rahmen auf Bildgröße setzen`: stellt die Größe des Rahmens so ein, dass sie zur Originalgröße des Bildes passt.

Abhängig vom ausgewählten *Größenmodus*, sind die Einstellungen für *Platzierung* und *Bilddrehung* inaktiv oder nicht. Mit *Platzierung* können Sie die Lage des Bildes innerhalb seines Rahmens auswählen.

Von QGIS zur Verfügung gestellte `.SVG`-Dateien sind anpassbar, was bedeutet, dass Sie einfach eine andere *Füllfar-*

be, *Strichfarbe* (auch mit Deckkraft) und *Strichbreite* als im Original auswählen können, indem Sie die entsprechende Einstellung in der Gruppe *SVG Parameter* benutzen. Diese Eigenschaften können auch *datengetrieben übersteuert* sein.

Sollten Sie eine .SVG-Datei hinzufügen, die diese Einstellugnen nicht ermöglicht, kann es nötig sein, die folgenden Tags für z.B. die Deckkraft in die Datei einzufügen:

- *fill-opacity="param(fill-opacity)"*
- *stroke-opacity="param(outline-opacity)"*

In diesem Blogeintrag finden Sie ein Beispiel.

Bilder können mit dem Bedienfeld *Bilddrehung* gedreht werden. Wenn Sie das Kontrollkästchen *Mit Karte abgleichen* anhaken, wird der Drehwinkel des Bildes mit dem auf das ausgewählte Kartenelement angewendeten Drehwinkel synchronisiert; das macht es bequemer für Nordpfeile, die Sie folgendermaßen ausrichten können:

- **Gitternorden**: Richtung einer Gitterlinie, die parallel zum Zentralmeridian eines nationalen oder lokalen Gitternetzes ist;
- oder **Geographisch-Nord**: Richtung eines Längengrads, der am Nordpol konvergiert.

Sie können der Bilddrehung auch einen *Versatz* für die Missweisung hinzufügen.

The North Arrow Item

You can add a north arrow with the Add North Arrow 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 filled with the map over which the north arrow item is drawn. If none, then it falls back to the *reference map*.

Bemerkung: Vielen Nordpfeilen enthalten kein ‚N'. Das ist mit Absicht so, für Sprachen die kein ‚N' für Norden verwenden, so dass Sie einen anderen Buchstaben verwenden können.

17.2.9 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*.

The HTML item can be customized using its *Item Properties* panel. Other than the *items common properties*, this feature has the following functionalities (see *figure_layout_html*):

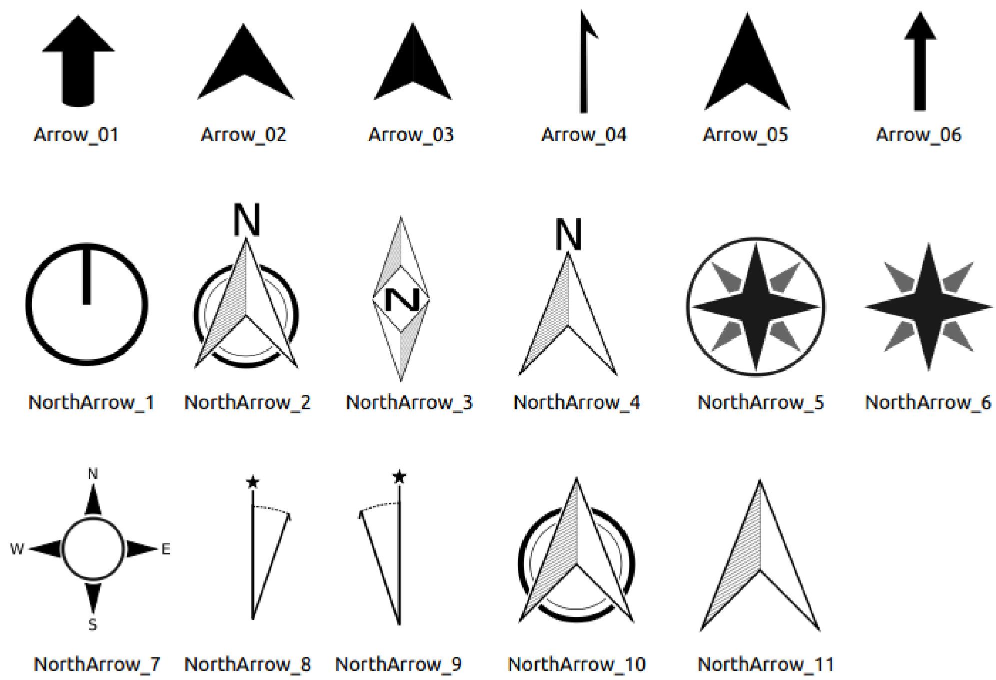


Abb. 17.48: Nordpfeile, die von der SVG-Bibliothek zur Verfügung gestellt werden.

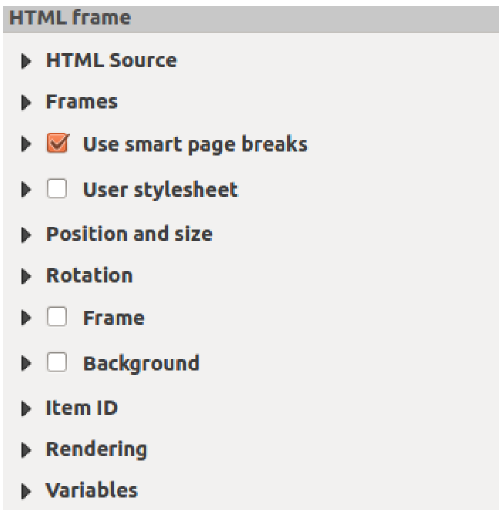


Abb. 17.49: HTML Frame, the Item Properties Panel

HTML-Quelle

The *HTML Source* group of the HTML frame *Item Properties* panel provides the following functionalities (see *figure_layout_html_ppt*):

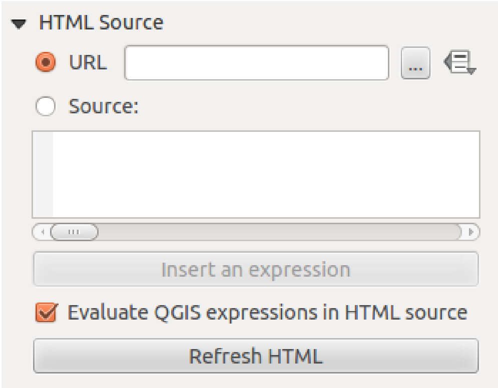


Abb. 17.50: HTML frame, the HTML Source properties

- In *URL* you can enter the URL of a webpage you copied from your Internet browser or select an HTML file using the … Browse 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 an Expression* button can be used to insert 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 *figure_layout_html_frames*):

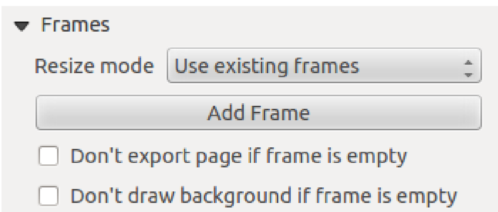


Abb. 17.51: HTML frame, the Frames properties

- 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 *figure_layout_html_breaks*):

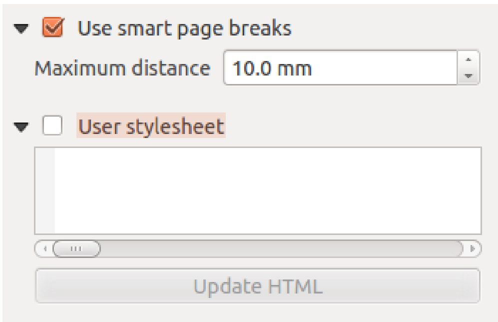


Abb. 17.52: HTML frame, Use smart page breaks and User style sheet properties

- 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.
- Activate *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.

17.2.10 The Shape Items

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).

The Regular Shape Item

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 figure_layout_label):

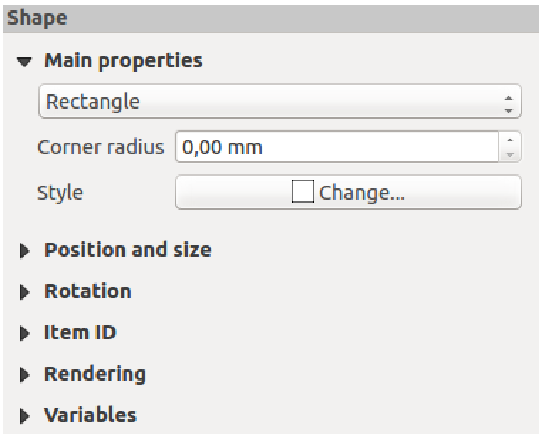


Abb. 17.53: Shape Item Properties Panel

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.

The Node-Based Shape Items

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°.
4. When you're done, right-click to terminate the shape.

You can customize the appearance of the shape in the *Item Properties* panel.

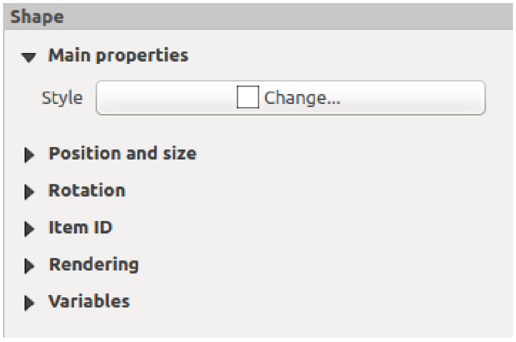


Abb. 17.54: Polygon Node Shape Item Properties Panel

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*.

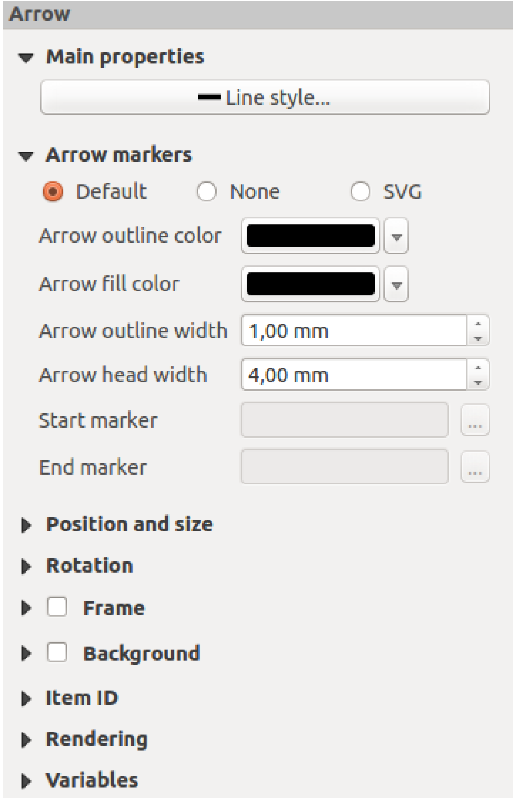


Abb. 17.55: Polyline Node Shape Item Properties Panel

The Arrow Item

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.

17.3 Eine Ausgabe erzeugen

Abb. 17.56 shows an example print layout including all the types of layout items described in the previous section.

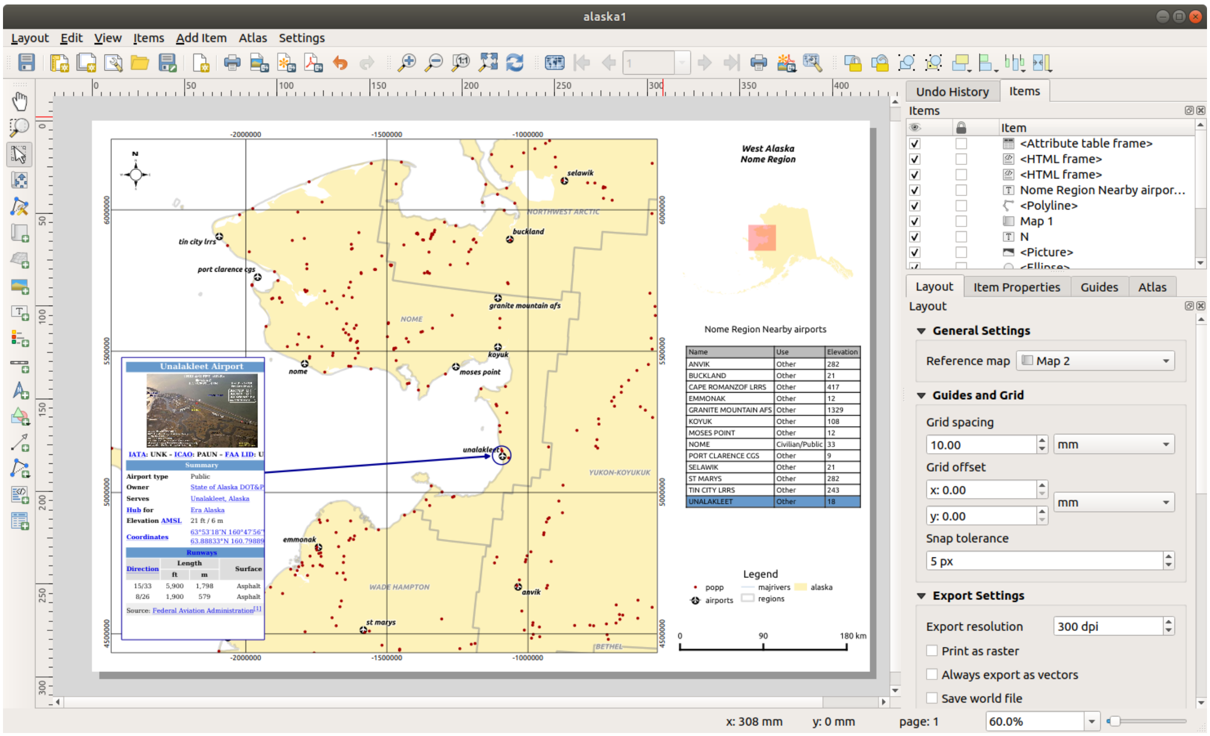


Abb. 17.56: Print Layout with map view, legend, image, scale bar, coordinates, text and HTML frame added

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.

17.3.1 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

17.3.2 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 (`.tfw` 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.

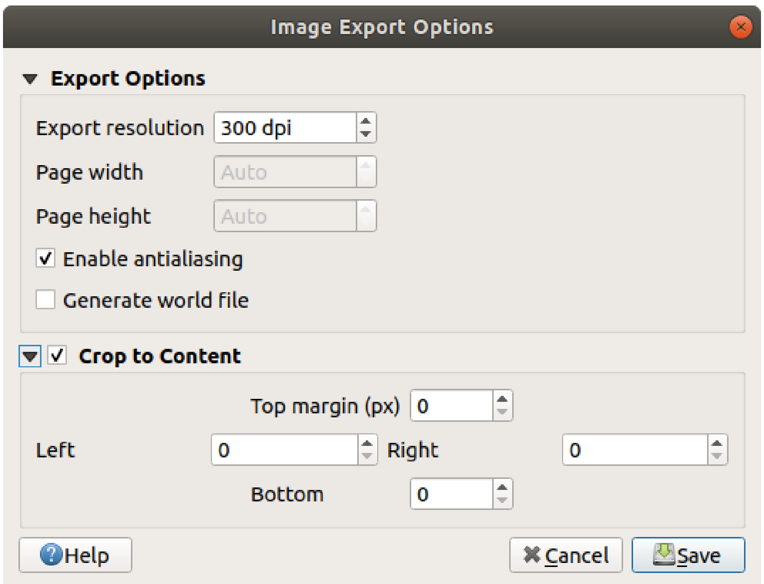


Abb. 17.57: Image Export Options, output is resized to items extent

---

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…)

17.3.3 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.

Bemerkung: Zur Zeit ist die SVG Ausgabe sehr einfach gehalten. Dies ist kein QGIS Probelm sondern ein Problem mit der darunterliegenden Qt Bibliothek. Dies wird hoffentlich in zukünftigen Versionen behoben.

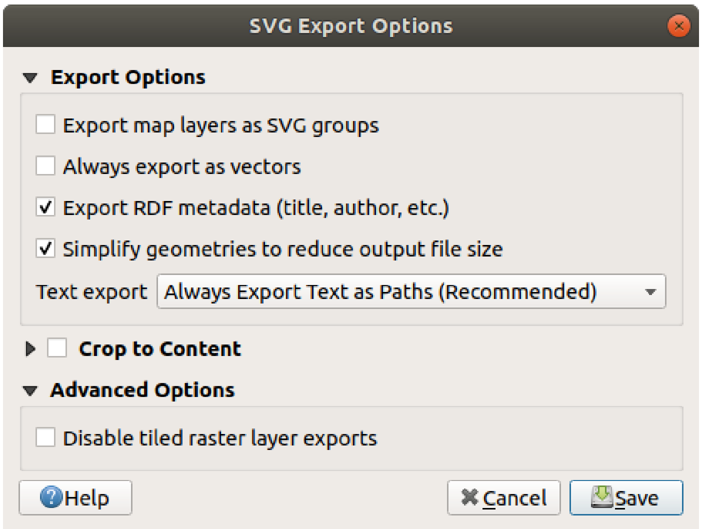


Abb. 17.58: SVG Exportoptionen

17.3.4 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*
   - *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.
   - *Create Geospatial PDF (GeoPDF)*: Generate a georeferenced PDF file (requires GDAL version 3 or later).
   - *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).

Bemerkung: Since QGIS 3.10, with GDAL 3, GeoPDF export is supported, and a number of GeoPDF specific options are available:

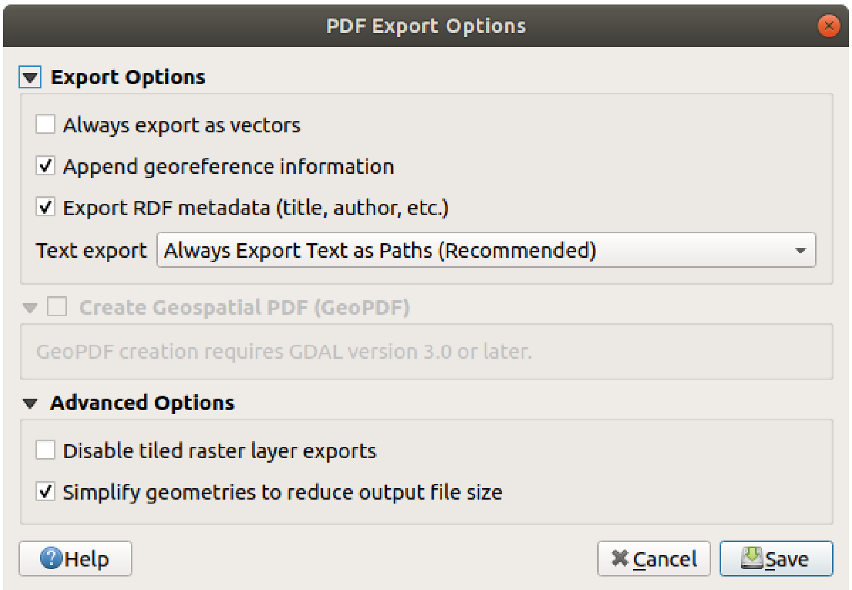


Abb. 17.59: PDF Export Options

- *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.

17.3.5 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 *figure_layout_atlas*):

- *Generate an atlas* enables or disables atlas generation.
- *Configuration*

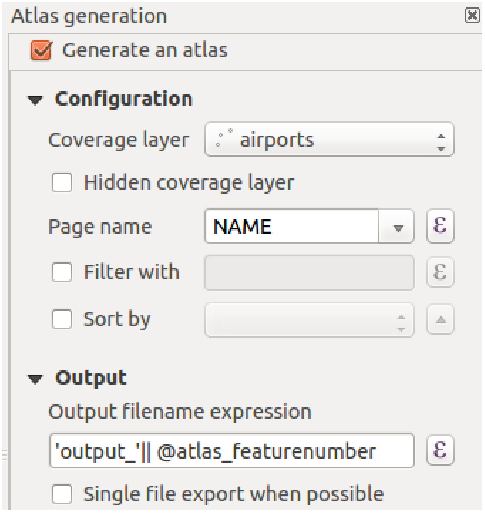


Abb. 17.60: Atlas Panel

  - 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 .

This is just one example of how you can use some advanced settings with atlas.

Preview and generate an atlas

Abb. 17.61: 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.

17.4 Creating a Report

This section will help you set up a report in QGIS.

17.4.1 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.

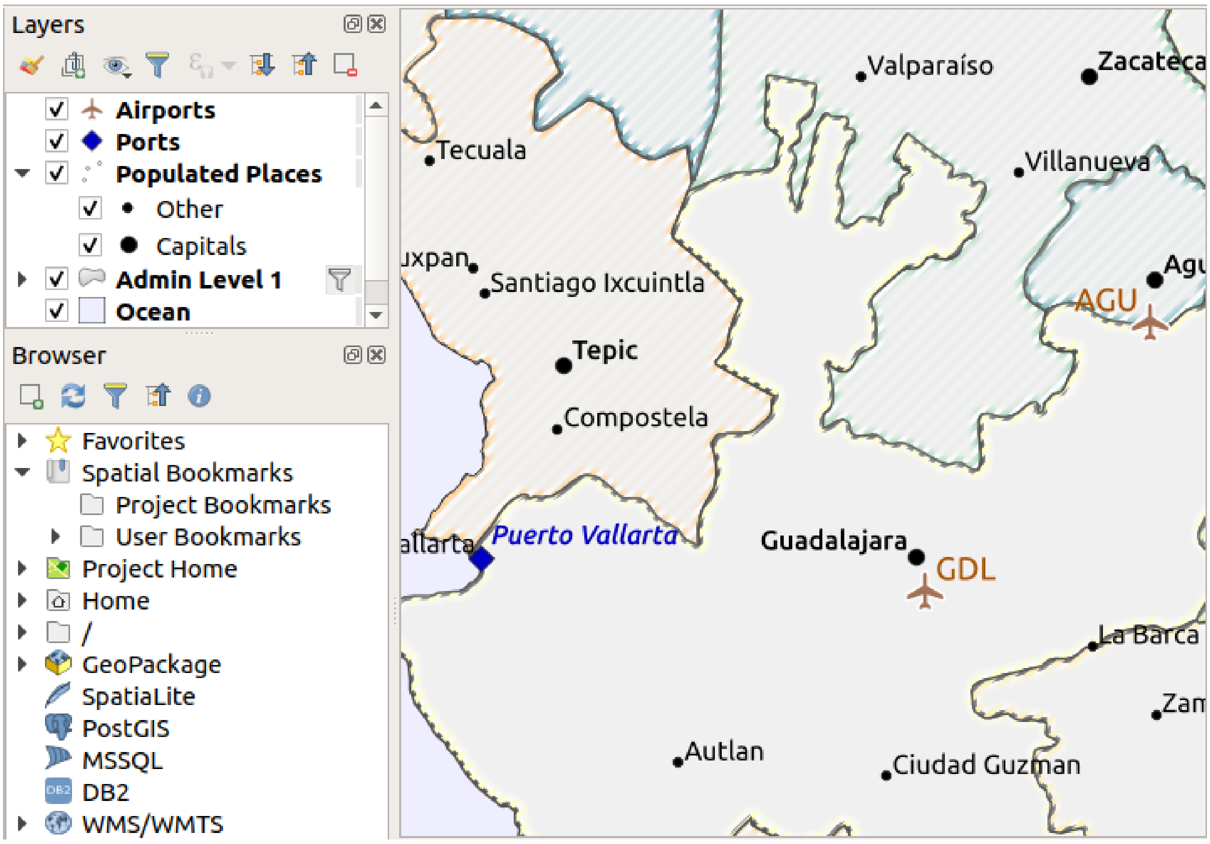
A report can be created with *Project New Report* or inside the *Project Layout Manager*.

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*.

17.4.2 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:

17.4.3 Layout Report Workspace

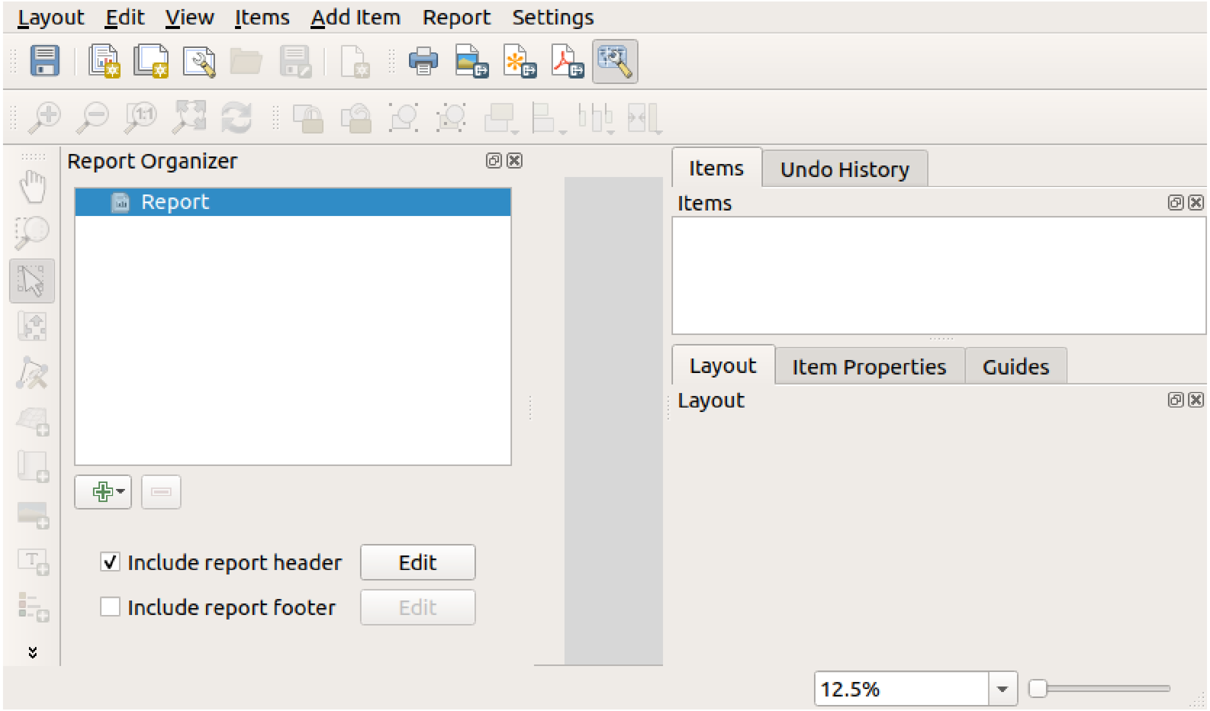
QGIS reports can consist of multiple, nested sections. In our new blank report we initially only have the main report section. The only options for this report section is *Include report header* and *Include report footer*. If we enable these options, a header will be included as the first page(s) (individual 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:

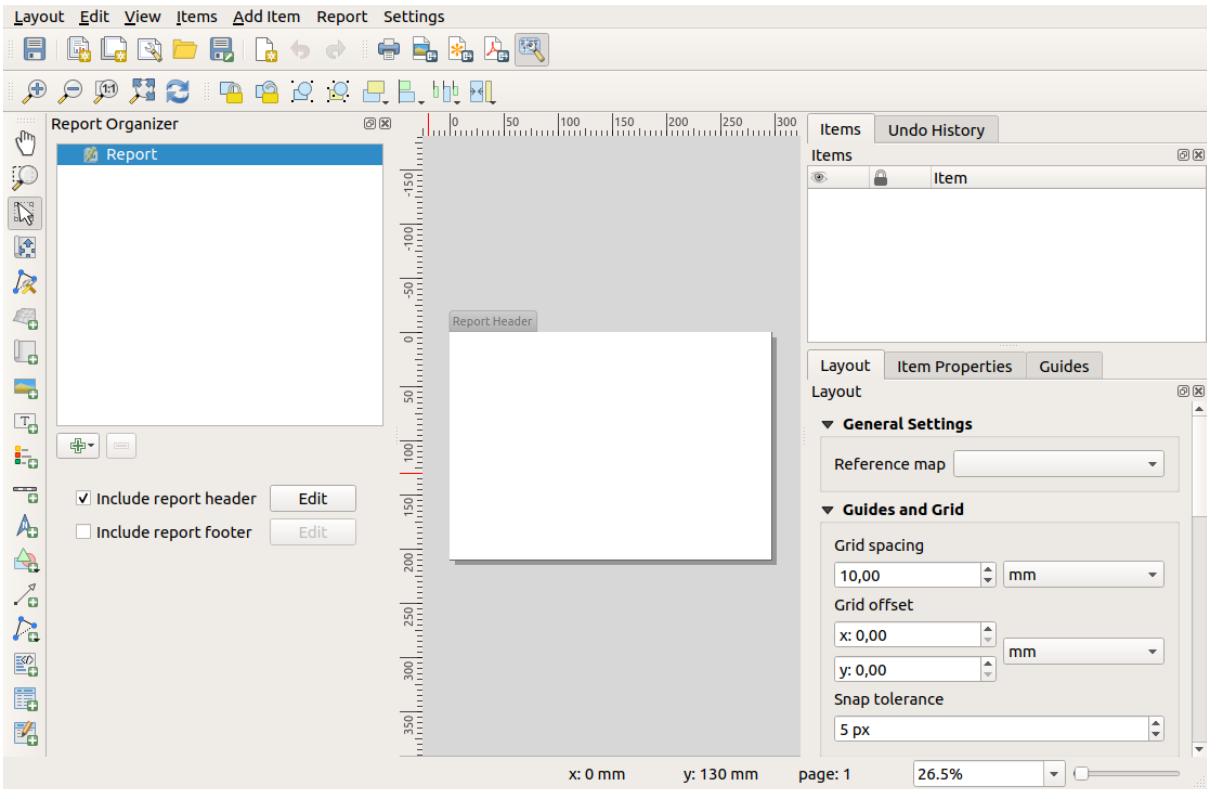
A few things happen as a result. Firstly, an edit pencil is shown next to *Report* in the *Report Organizer*, indicating that the report section is currently being edited in the designer. We also see a new page with a small *Report Header* title. The page has *landscape* orientation by default, but this (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.

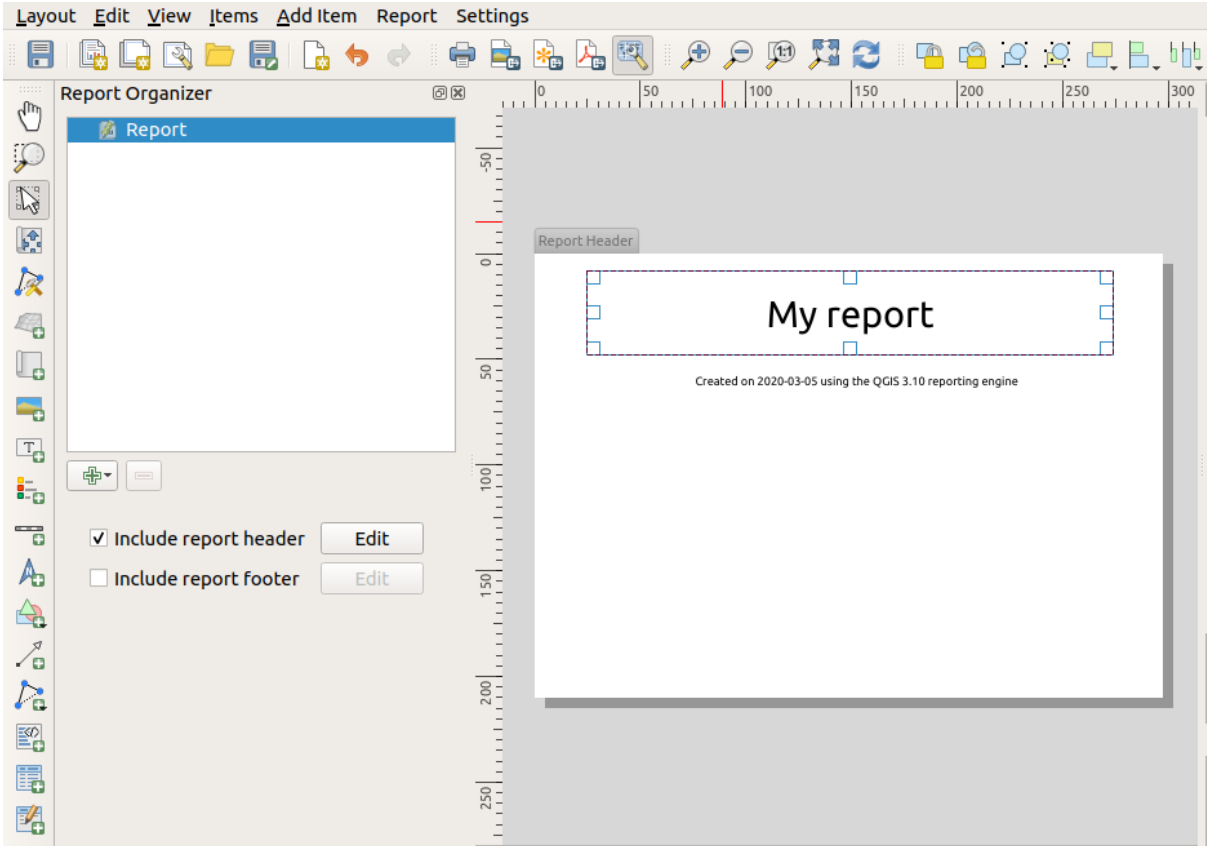
In QGIS reports, every component of the report is made up of individual layouts. They can be created and modified using the same tools as for standard print layouts – so you can use any desired combination of labels, pictures, maps, tables, etc. Let us add some items to our report header to demonstrate:

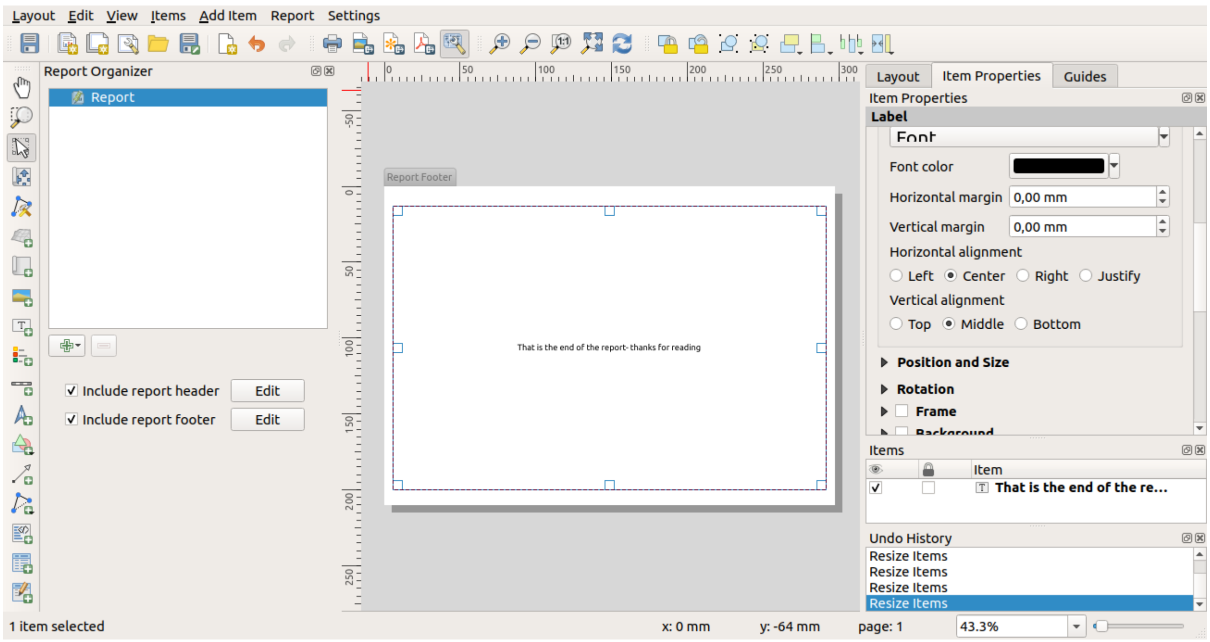
We will also create a simple footer for the report by checking the *Include report footer* option and hitting *Edit*.

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

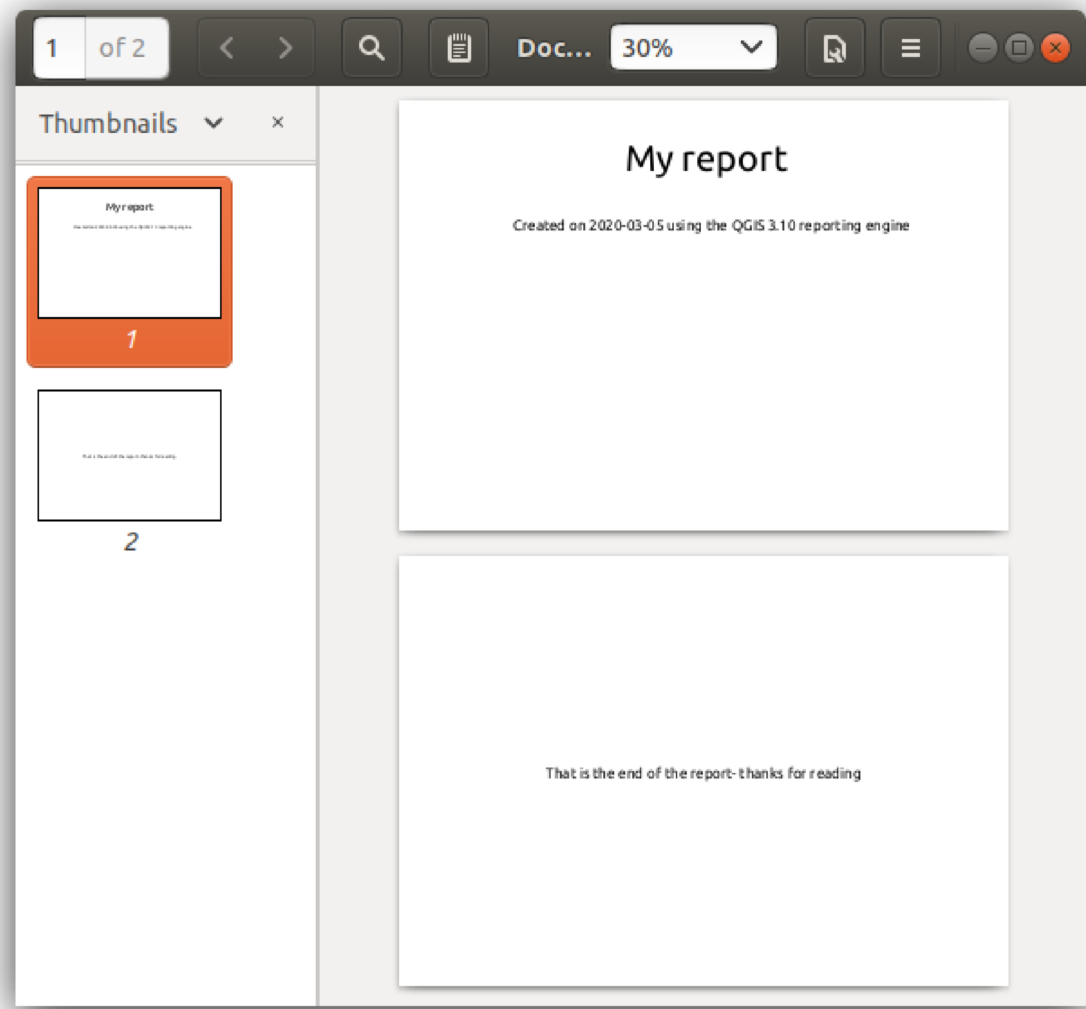

Layout Edit View Items Add Item Report Settings
Report Organizer
Report
Include report header
Edit
Include report footer
Edit
Items
Undo History
Items
Layout
Item Properties
Guides
Layout
12.5%


Layout Edit View Items Add Item Report Settings
Report Organizer
Report
Include report header
Edit
Include report footer
Edit
Report Header
Items
Undo History
Items
Item
Layout
Item Properties
Guides
Layout
General Settings
Reference map
Guides and Grid
Grid spacing
10,00
mm
Grid offset
x: 0,00
y: 0,00
mm
Snap tolerance
5 px
x: 0 mm
y: 130 mm
page: 1
26.5%


Layout Edit View Items Add Item Report Settings
Report Organizer
Report
Include report header
Edit
Include report footer
Edit
Report Header
My report
Created on 2020-03-05 using the QGIS 3.10 reporting engine


Layout Edit View Items Add Item Report Settings
Report Organizer
Report
Include report header
Edit
Include report footer
Edit
Report Footer
That is the end of the report- thanks for reading
Layout
Item Properties
Guides
Label
Font
Font color
Horizontal margin
0,00 mm
Vertical margin
0,00 mm
Horizontal alignment
Left
Center
Right
Justify
Vertical alignment
Top
Middle
Bottom
Position and Size
Rotation
Frame
Items
Item
That is the end of the re...
Undo History
Resize Items
Resize Items
Resize Items
Resize Items
1 item selected
x: 0 mm
y: -64 mm
page: 1
43.3%

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*.

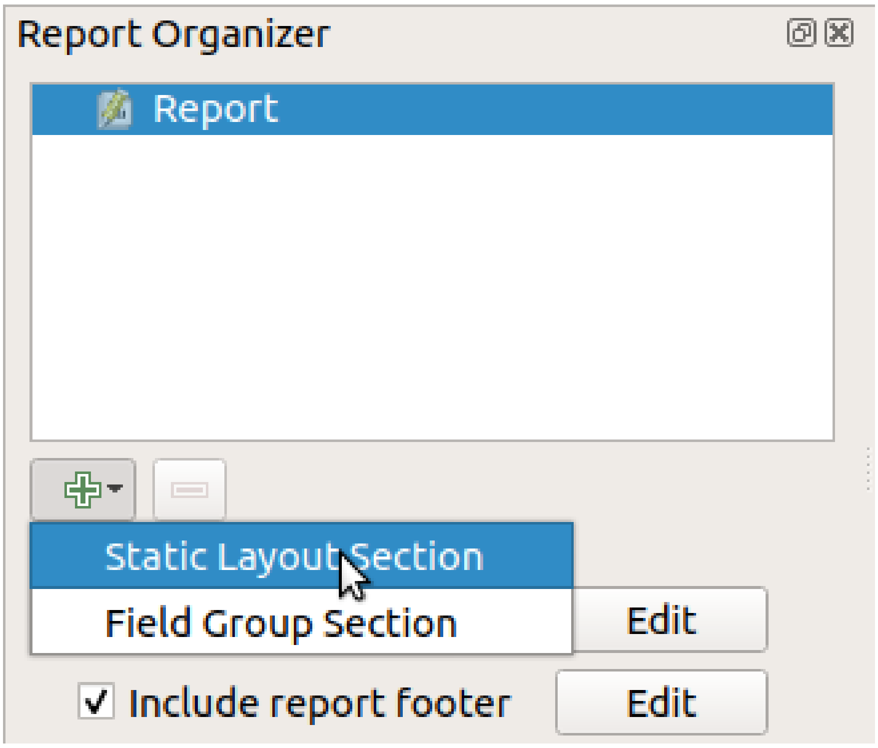
The *Add Static Layout Section* is a single, static body layout. This can be used to embed static layouts mid-way through a report.

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 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:

So more or less an atlas, but with a header and footer page.

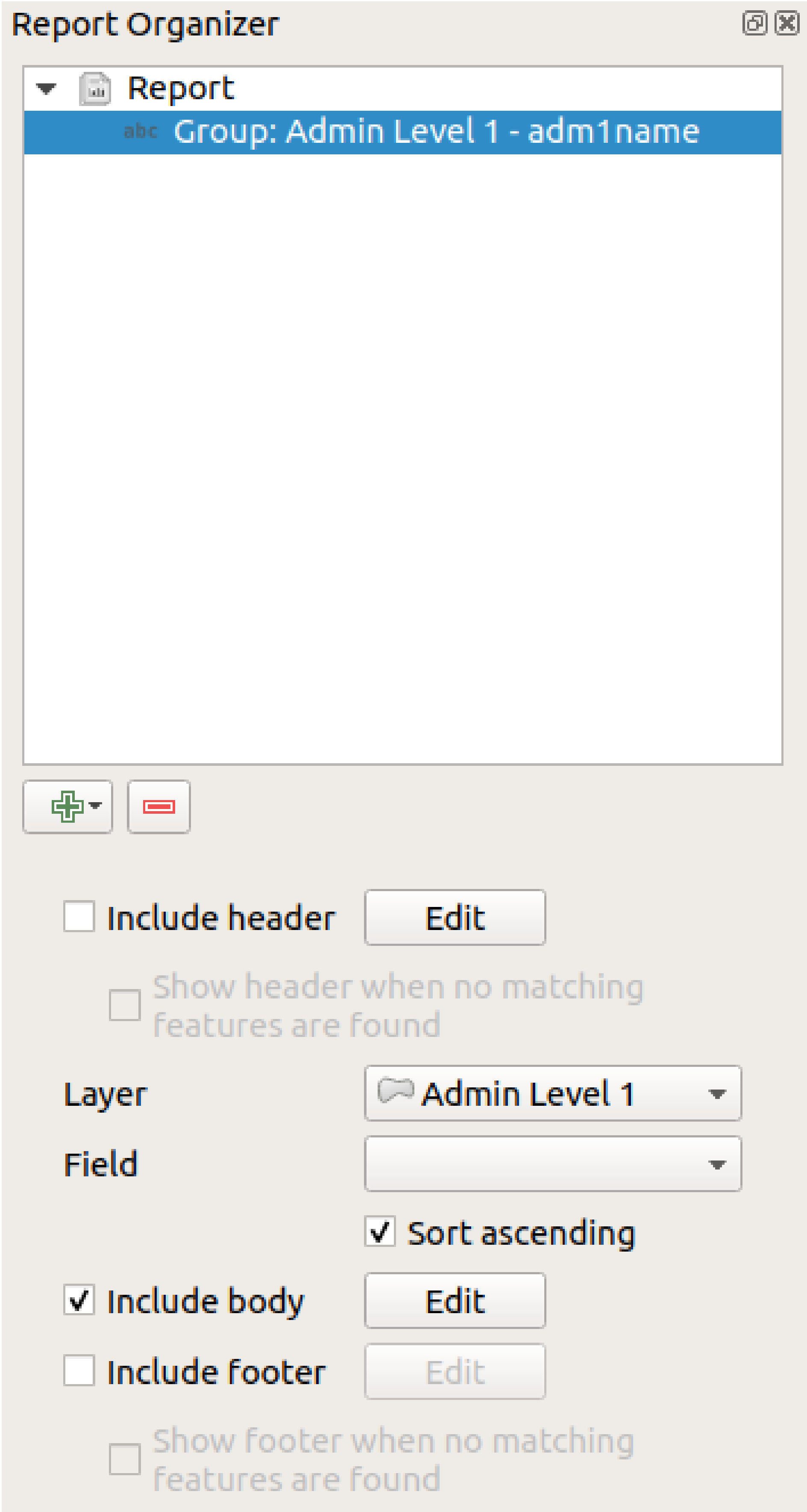
Let us make things more interesting by adding a subsection to our state group. We do this by first selecting the *Admin Level 1* field group in the organizer, then hitting the Add Field button and adding a new *Field Group Section*:

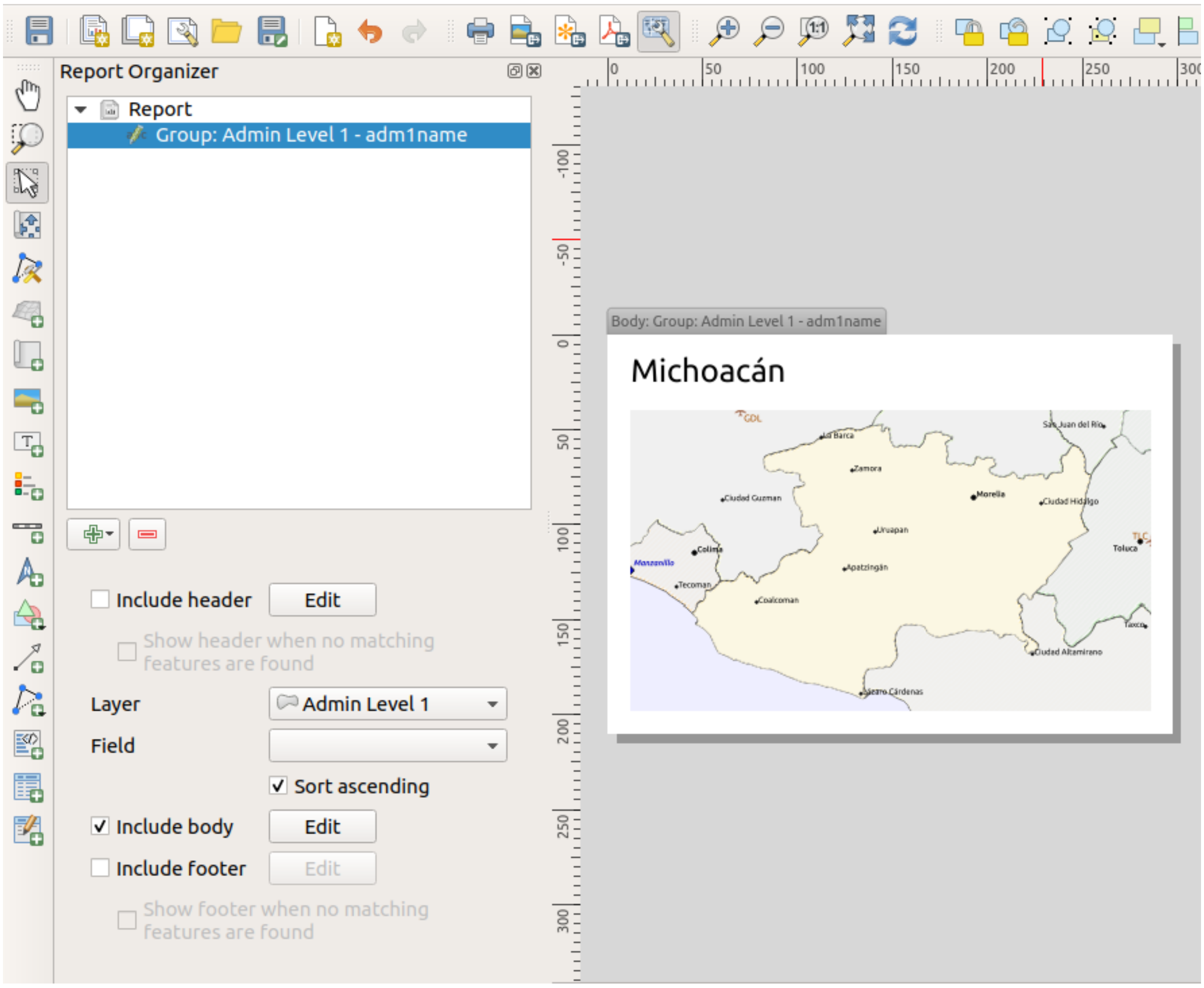
When iterating over the features of a *Field Group Section*, the features will be filtered to match the defining field of its parent group (`adm1name` in this case). Here, the subsection we added will iterate over a *Populated Places* layer, including a body section for each place encountered. The magic 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:

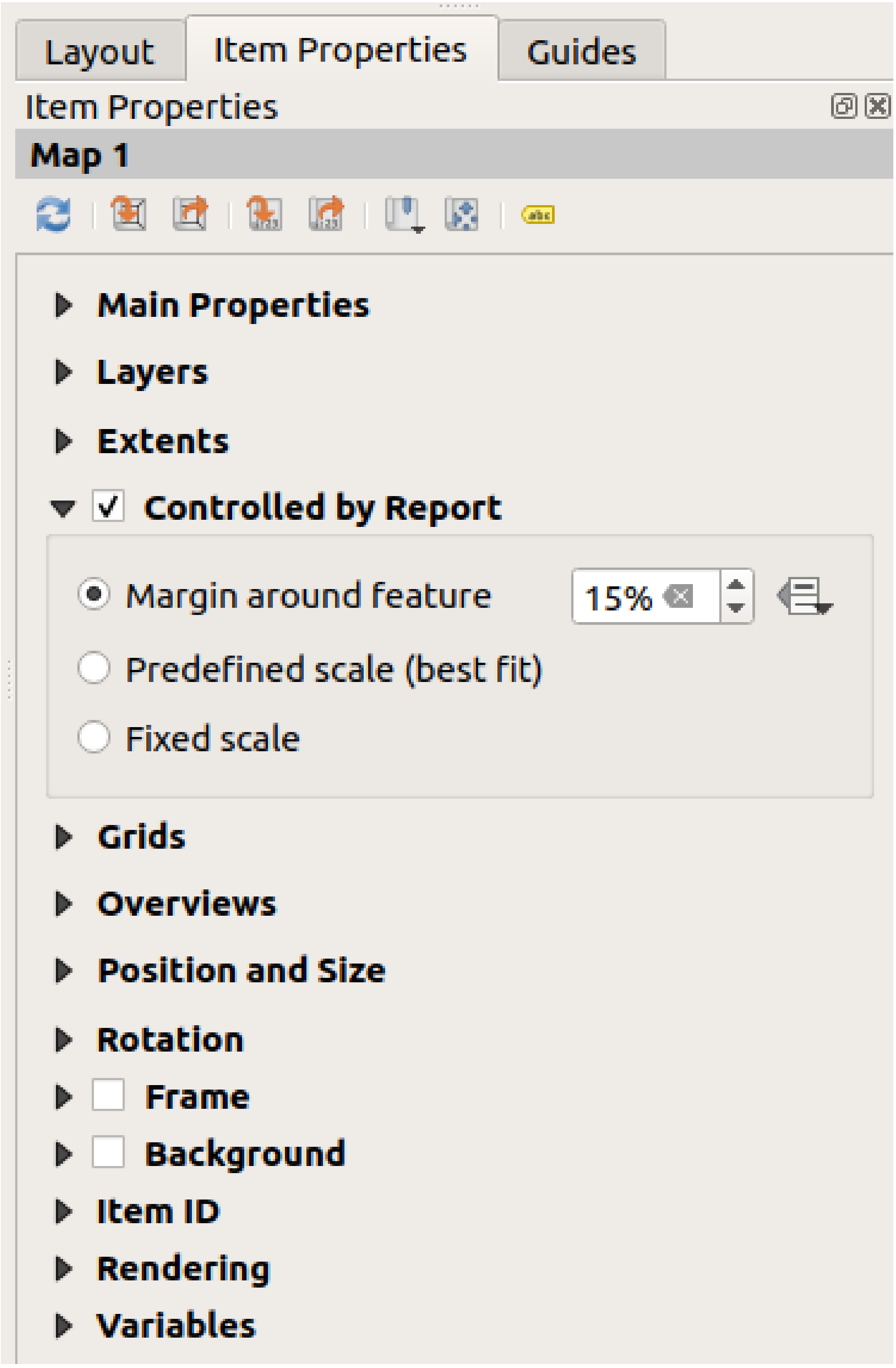
Here we created a basic body for the Populated Places group, including a map of the place and a table of some place attributes. So our report is now a report header, a page for the first state, followed by a page for every populated place within that state, then the rest of the states with their populated 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


Report Organizer
Report
Group: Admin Level 1 - adm1name
Include header
Edit
Show header when no matching features are found
Layer
Admin Level 1
Field
Sort ascending
Include body
Edit
Include footer
Edit
Show footer when no matching features are found


Report Organizer
Report
Group: Admin Level 1 - adm1name
Include header
Edit
Show header when no matching features are found
Layer
Admin Level 1
Field
Sort ascending
Include body
Edit
Include footer
Edit
Show footer when no matching features are found
Body: Group: Admin Level 1 - adm1name
Michoacán


Layout
Item Properties
Guides
Item Properties
Map 1
Main Properties
Layers
Extents
Controlled by Report
Margin around feature
15%
Predefined scale (best fit)
Fixed scale
Grids
Overviews
Position and Size
Rotation
Frame
Background
Item ID
Rendering
Variables

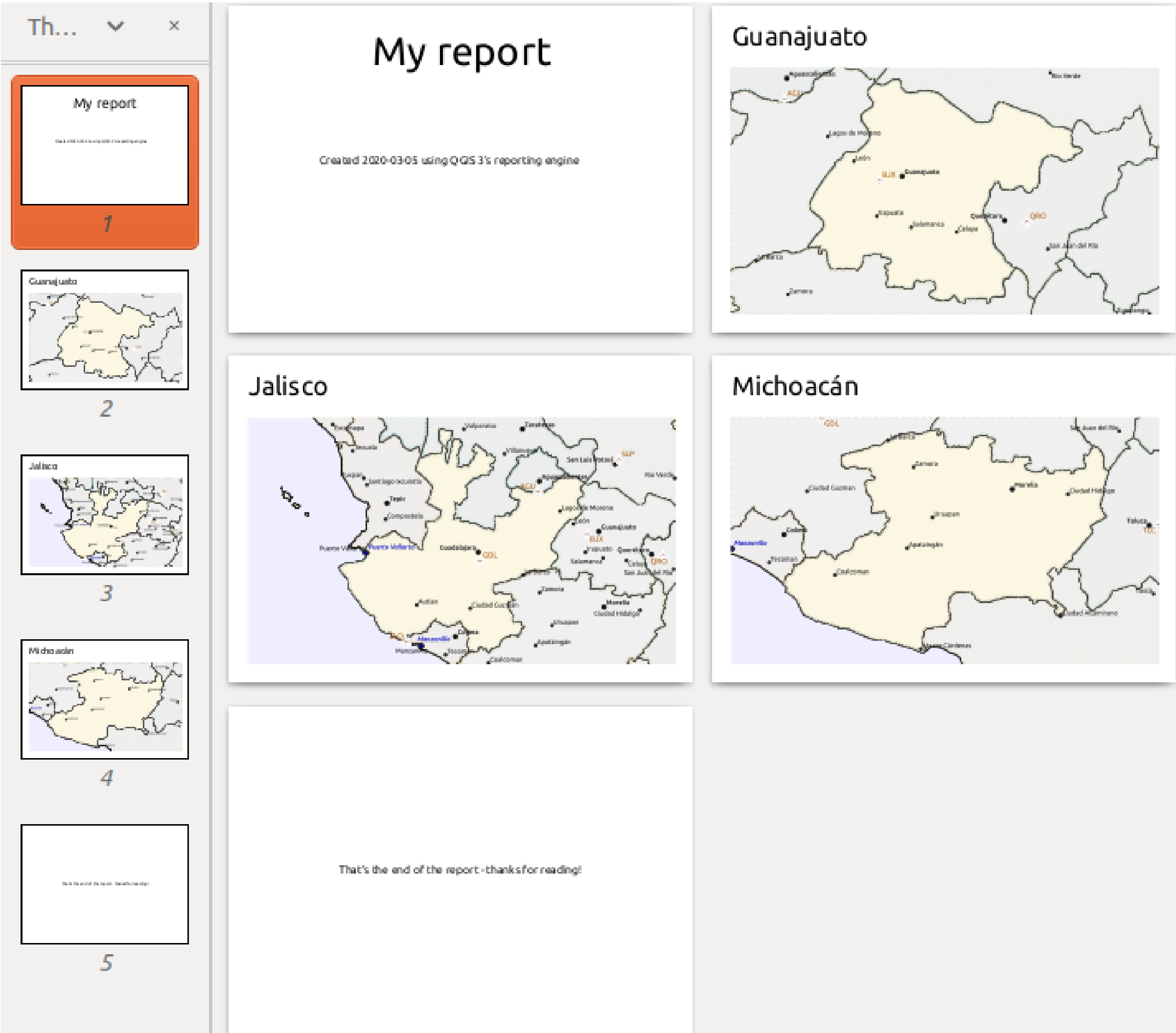
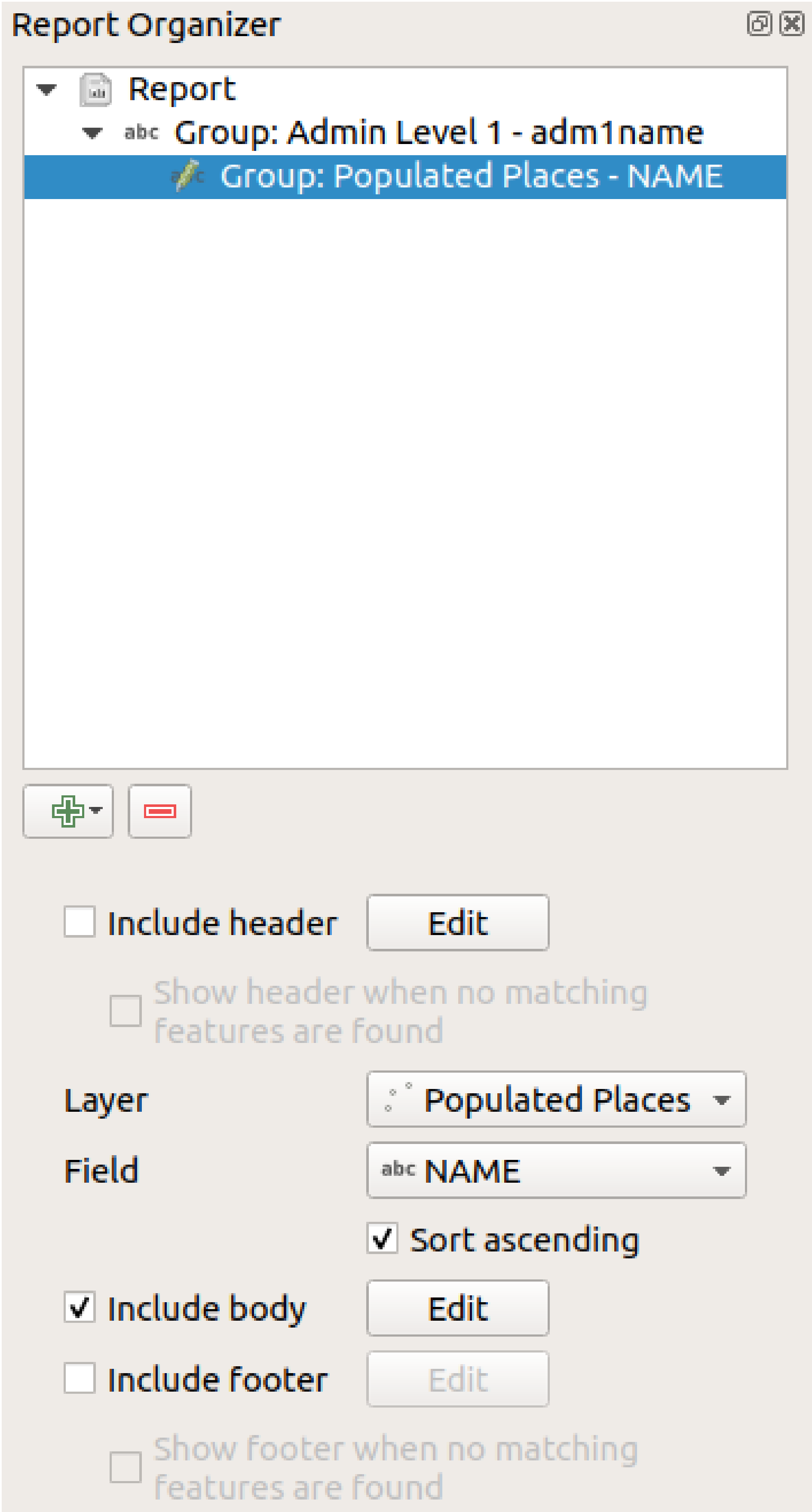
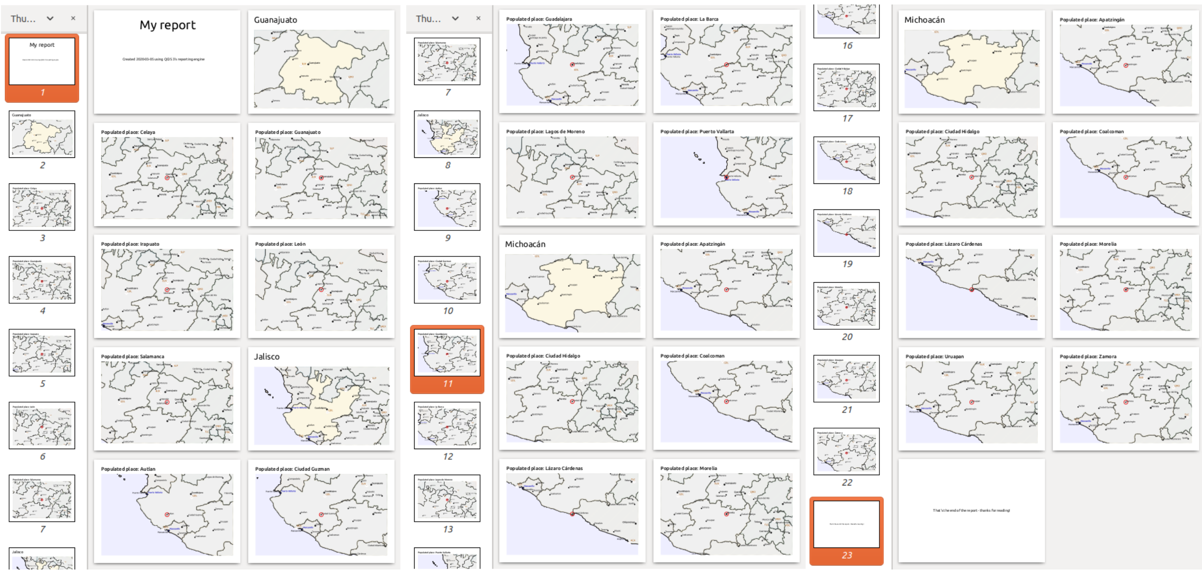


Abb. 17.62: The report header, a page for each state, and the report footer.


Report Organizer
Report
Group: Admin Level 1 - adm1name
Group: Populated Places - NAME
Include header
Edit
Show header when no matching features are found
Layer
Populated Places
Field
NAME
Sort ascending
Include body
Edit
Include footer
Edit
Show footer when no matching features are found



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.

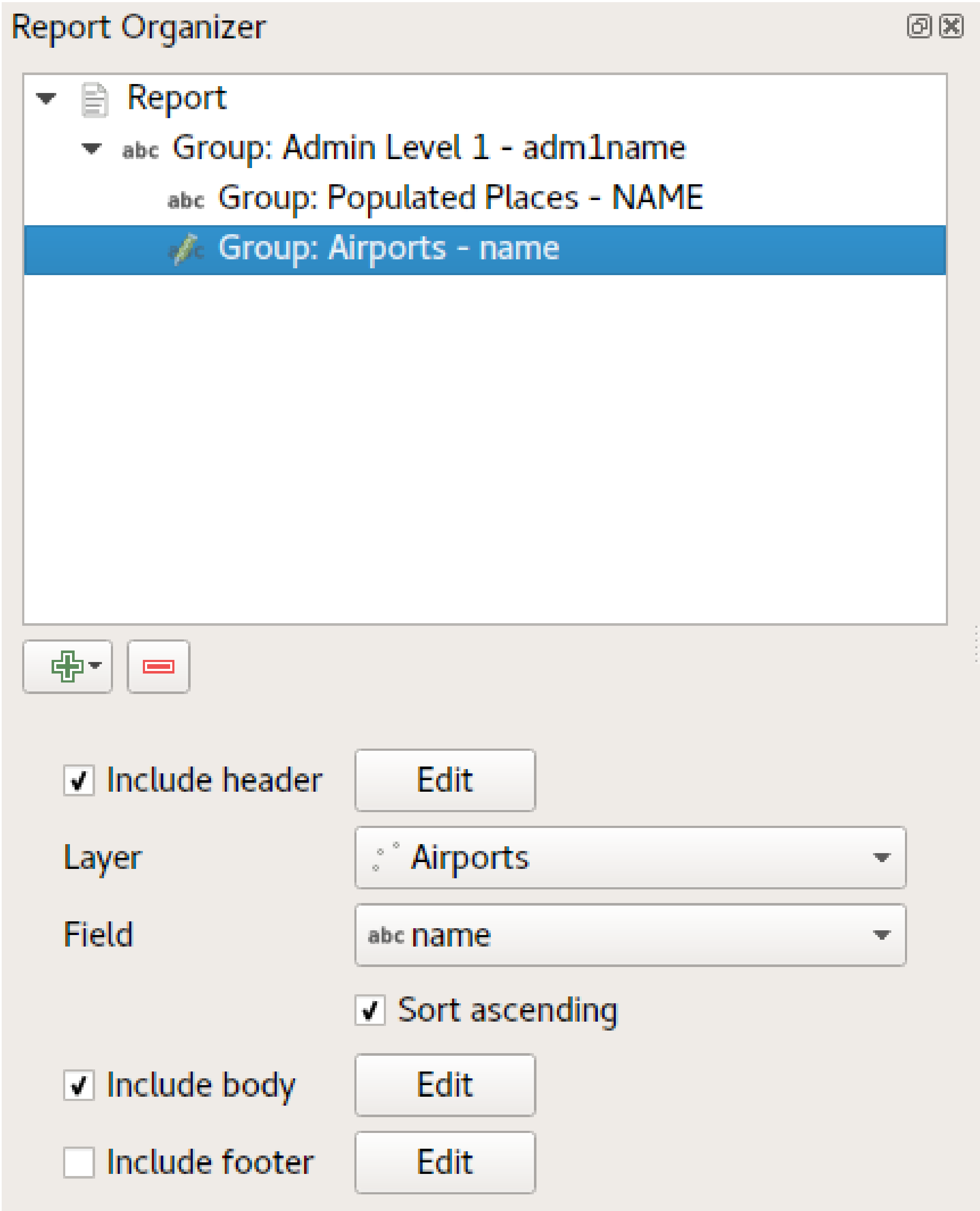
QGIS depends on absolute file names for images in reports.

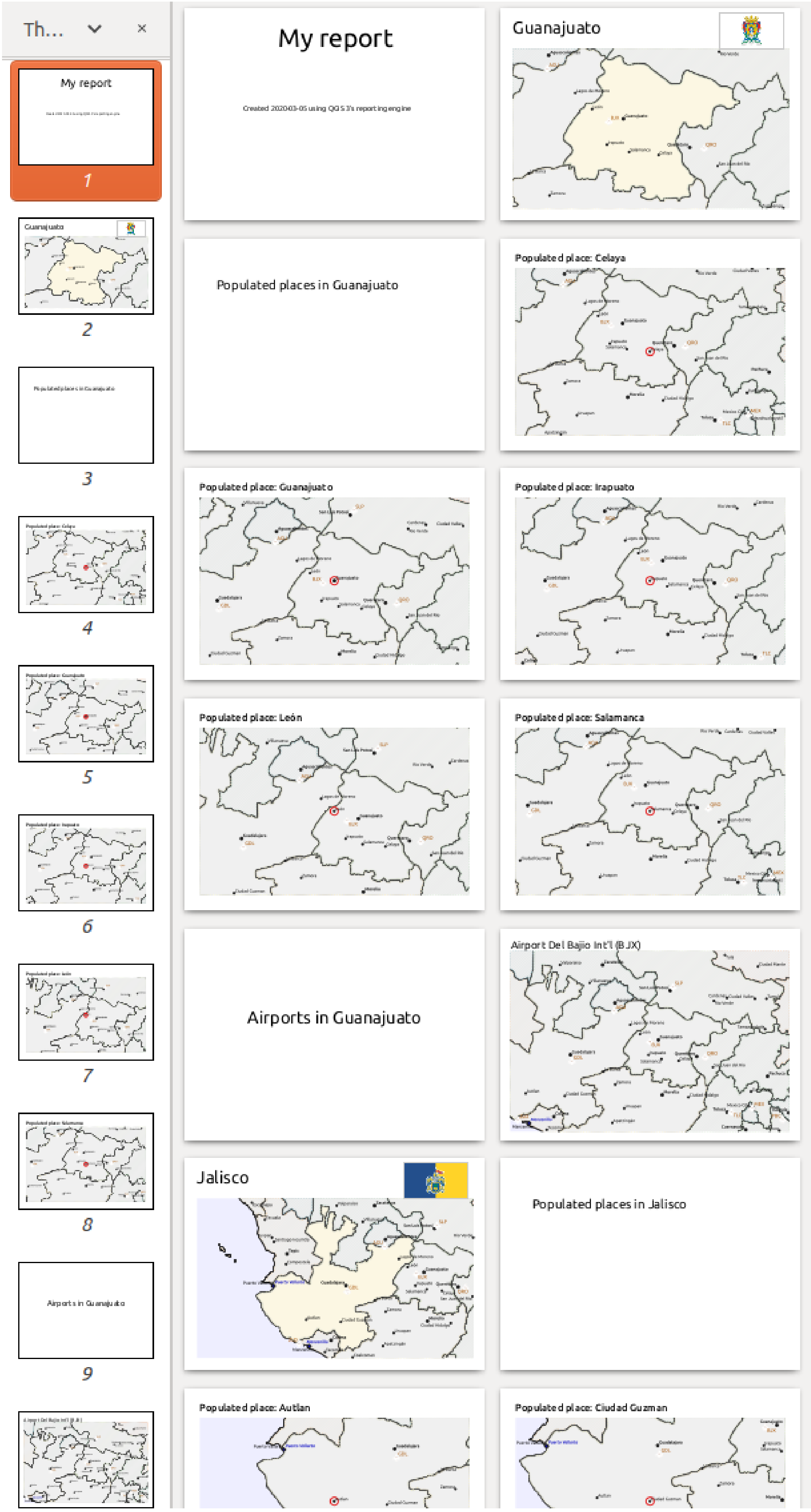
For dynamic pictures, you first add a picture to the body part of the group, as usual. In the *Item properties* of the picture, you set the *Image Source* using the Data defined override button, and either select an attribute that contains the absolute path of the images or *Edit…* (to enter an expression that generates the absolute image path).

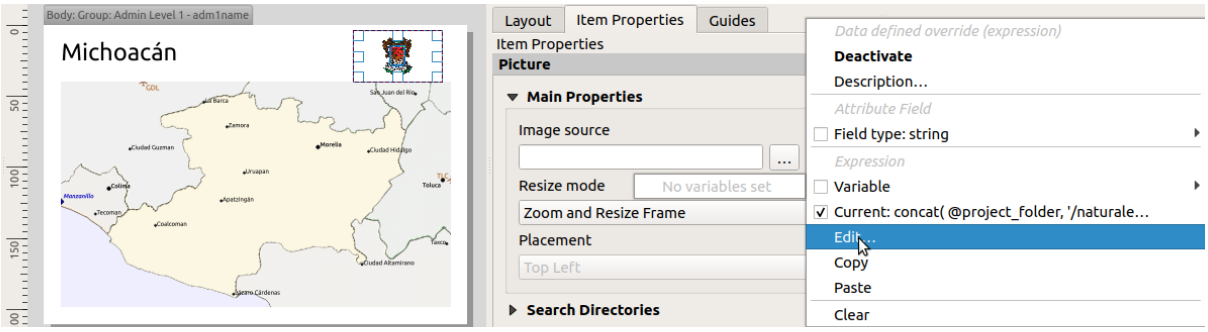
Below is an example expression that uses string concatenation to specify the absolute path to the pictures, using the directory where the project file is located `@project_path`) and an attribute (`adm1name`) from which the file name is generated (in this case by transforming the string in the `adm1name` attribute to uppercase, and appending ‚_flag.png‘):

```
concat(@project_folder, '/naturalearth/pictures/' ,
        upper("adm1name"), '_flag.png')
```

This means that the pictures are located in the `naturalearth/pictures` subdirectory of the project file directory.


Report Organizer
Report
Group: Admin Level 1 - adm1name
Group: Populated Places - NAME
Group: Airports - name
Include header
Edit
Layer
Airports
Field
name
Sort ascending
Include body
Edit
Include footer
Edit


My report
Guanajuato
Populated places in Guanajuato
Populated place: Celaya
Populated place: Guanajuato
Populated place: Irapuato
Populated place: León
Populated place: Salamanca
Airports in Guanajuato
Airport Del Bajío Int'l (BJX)
Jalisco
Populated places in Jalisco
Populated place: Autlan
Populated place: Ciudad Guzman



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', '#88CCCCCC')
```

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 :guilabel`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:

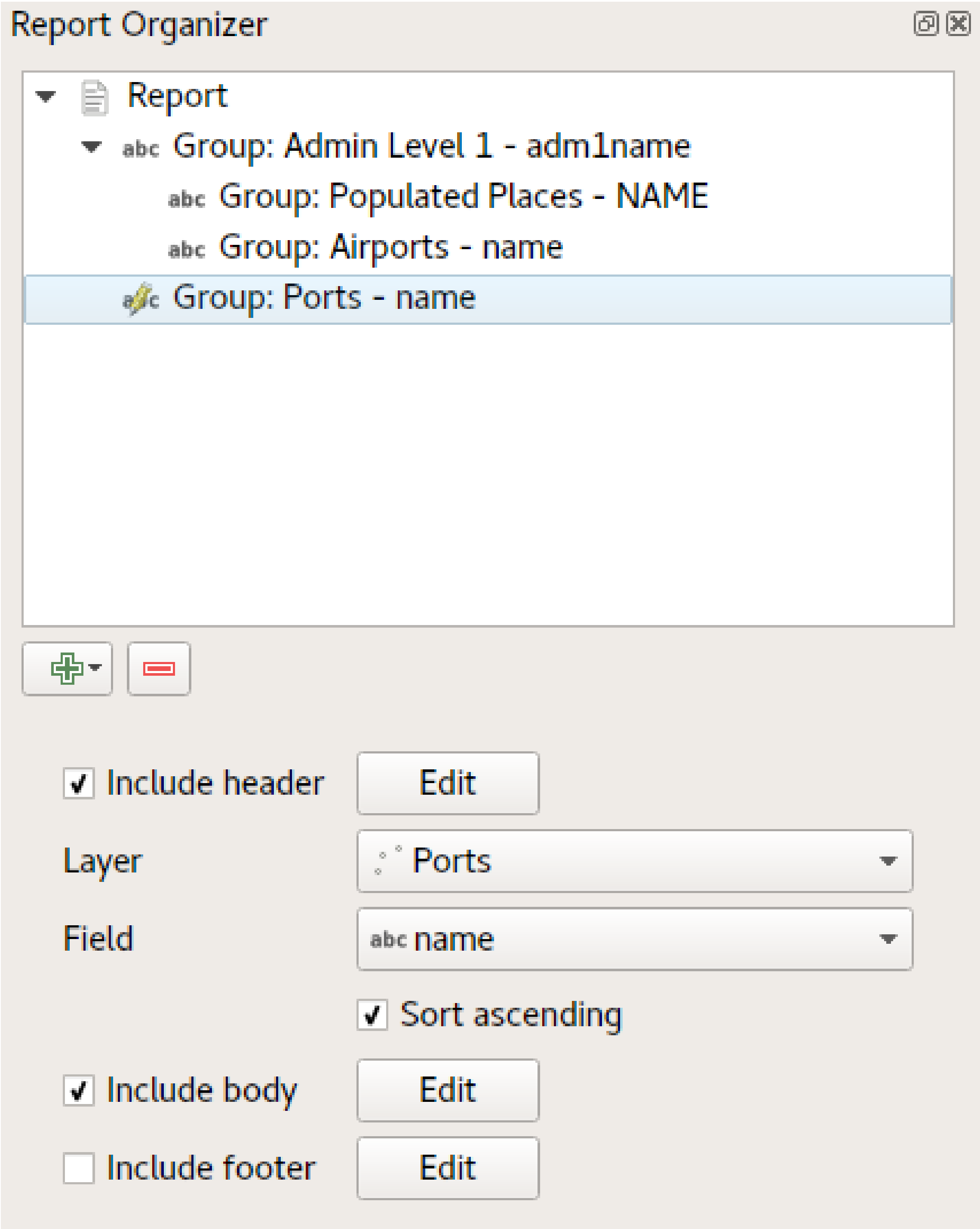
This results in the last part of our report exporting as:

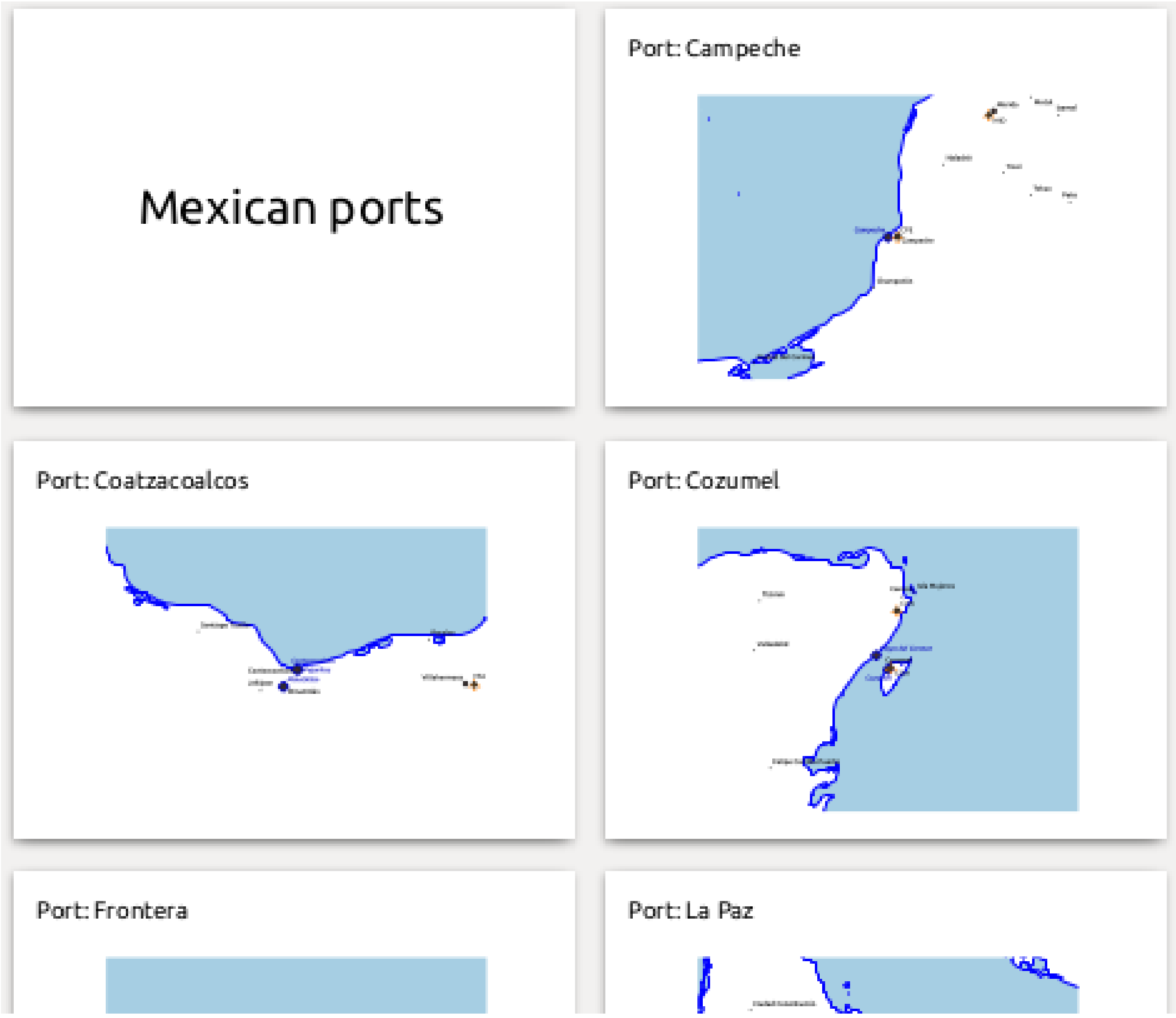
17.4.4 Exporteinstellungen

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!


Report Organizer
Report
Group: Admin Level 1 - adm1name
Group: Populated Places - NAME
Group: Airports - name
Group: Ports - name
Include header
Edit
Layer
Ports
Field
abc name
Sort ascending
Include body
Edit
Include footer
Edit


Mexican ports
Port: Campeche
Port: Coatzacoalcos
Port: Cozumel
Port: Frontera
Port: La Paz

KAPITEL 18

Arbeiten mit OGC Daten

18.1 QGIS als OGC Datenclient

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.opengeospatial.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*).

18.1.1 WMS/WMTS Client

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.

Darüber hinaus speichert QGIS Ihre WMS-Antworten (d. h. Bilder) 24 Stunden lang im Cache, solange die GetCapabilities-Anforderung nicht ausgelöst wird. Die GetCapabilities-Anforderung wird jedes Mal ausgelöst, wenn die Schaltfläche: guilabel: *Connect* im Dialogfeld: guilabel:` Layer von WMS (T) Server hinzufügen` verwendet wird, um die WMS-Capabilities abzurufen. Dies ist eine automatische Funktion zur Optimierung der Projektladezeit. Wenn ein Projekt mit einer WMS-Ebene gespeichert wird, werden die entsprechenden WMS-Kacheln beim nächsten Öffnen des Projekts aus dem Cache geladen, sofern sie nicht älter als 24 Stunden sind.

Überblick über die WMTS Unterstützung

|qgl| 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:

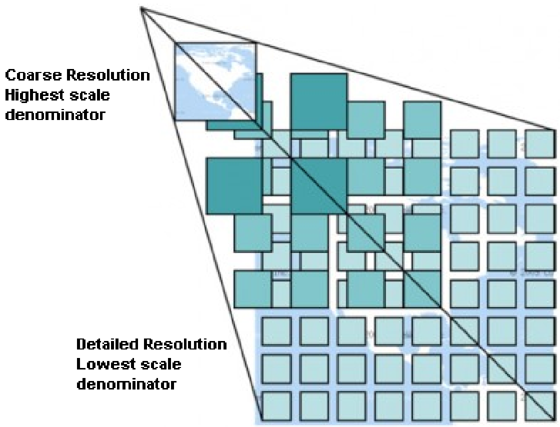


Abb. 18.1: Konzept der WMTS Tile Sets

Dei 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"
   ```

   Ein Beispiel für diesen Typ von Adresse ist

   ```
   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: You can still find some old services called WMS-C. These services are quite similar to WMTS (i.e., same purpose but working a little bit differently). You can manage them the same as you do WMTS services. Just add `?tiled=true` at the end of the url. See https://wiki.osgeo.org/wiki/Tile_Map_Service_Specification for more information about this specification.

Wenn Sie WMTS lesen können Sie auch an WMS-C denken.

WMS/WMTS Server auswählen

Wenn Sie das WMS Feature das erste Mal in QGIS verwenden sind keine Server definiert.

Begin by clicking the Add WMS layer button on the toolbar, or selecting *Layer ▯ Add WMS Layer…*.

The dialog *Add Layer(s) from a Server* for adding layers from the WMS server appears. You can add some servers to play with by clicking the *Add Default Servers* button. This will add two WMS demo servers for you to use: the WMS servers of the DM Solutions Group and Lizardtech. To define a new WMS server in the *Layers* tab, select the *New* button. Then enter the parameters to connect to your desired WMS server, as listed in *table_OGC_wms*:

Name	Ein Name für diese Verbindung. Dieser Name wird in der Serververbindungs Drop-down Box verwendet und kann diese dann von anderen WMS Servern unterscheiden.
URL	URL des Servers der die Daten bereitstellt. Dies muss ein auflösbarer Hostname sein-das gleiche Format das Sie auch benutzen würden um eine Telnetverbindung zu öffnen oder einen Ping an einen Host zu senden.
Benutzername	Benutzername um einen abgesicherten WMS Server anzubinden. Dieser Parameter ist optional.
Passwort	Passwort für einen durch Authentifizierung abgesicherten WMS Server. Dieser Parameter ist optional.
Gemeldete GetMap-URI aus Diensteigenschaften ignorieren	*Gemeldete GetMap-URI aus Diensteigenschaften ignorieren* und benutze stattdessen die angegebene URL aus dem URL-Feld oben.
Gemeldete GetFeatureInfo-URI ignorieren	*Gemeldete GetFeatureInfo-URI ignorieren*. Verwenden Sie die vorgegebene URI aus dem URL Feld oben.

Tabelle OGC 1: WMS Verbindungs-Parameter

Wenn Sie einen Poxyserver, um WMS Services aus dem Internet empfangen zu können, aufsetzen müssen können Sie Ihren Proxyserver in den Optionen hinzufügen. Wählen Sie *Einstellungen ▸ Optionen* und klicken Sie auf das Menü *Netzwerk*. Dort können sie Ihre Proxyeinstellungen hinzufügen und diese aktivieren indem Sie das Kontrollkästchen *Proxy für Webzugriff benutzen* aktivieren. Vergewissern Sie sich dass Sie den richtigen Proxytyp aus dem *Proxytyp* Dropdownmenü ausgewählt haben

Wenn Sie die WMS-Verbindung einmal gesetzt haben, ist sie für zukünftige QGIS Sitzungen gespeichert.

Tipp: WMS-Server-URLs

Vergewissern Sie sich beim Eingeben der WMS Server URL dass Sie nur die einfache URL eingeben. Zum Beispiel sollten keine Fragmente wie `request=GetCapabilities` oder `version=1.0.0` enthalten sein.

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 a *Authentication configuration* instead (*configurations* tab). See *Authentifizierungssystem* for more details.

WMS/WMTS Layer laden

Once you have successfully filled in your parameters, you can use the *Connect* button to retrieve the capabilities of the selected server. This includes the image encoding, layers, layer styles and projections. Since this is a network operation, the speed of the response depends on the quality of your network connection to the WMS server. While downloading data from the WMS server, the download progress is visualized in the lower left of the WMS dialog.

Your screen should now look a bit like *figure_OGC_add_wms*, which shows the response provided by the European Soil Portal WMS server.

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 Optionen Bereich des Dialogs stellt ein Textfeld zur Verfügung in das Sie einen *Layernamen* für den WMS Layer hinzufügen können. Dieser Name wird nach dem Laden in der Legende erscheinen.

Unter dem Layernamen können Sie wenn Sie den WMS Request in mehrere Requests aufsplitten wollen die *Kachelgröße* (z.B. 256x256) definieren.

Die *Objektbegrenzung für GetFeatureInfo* legt fest welche Attributspalten vom Server abgefragt werden.

If you select a WMS from the list, a field with the default projection provided by the mapserver appears. If the *Change…* button is active, you can click on it and change the default projection of the WMS to another CRS provided by the WMS server.

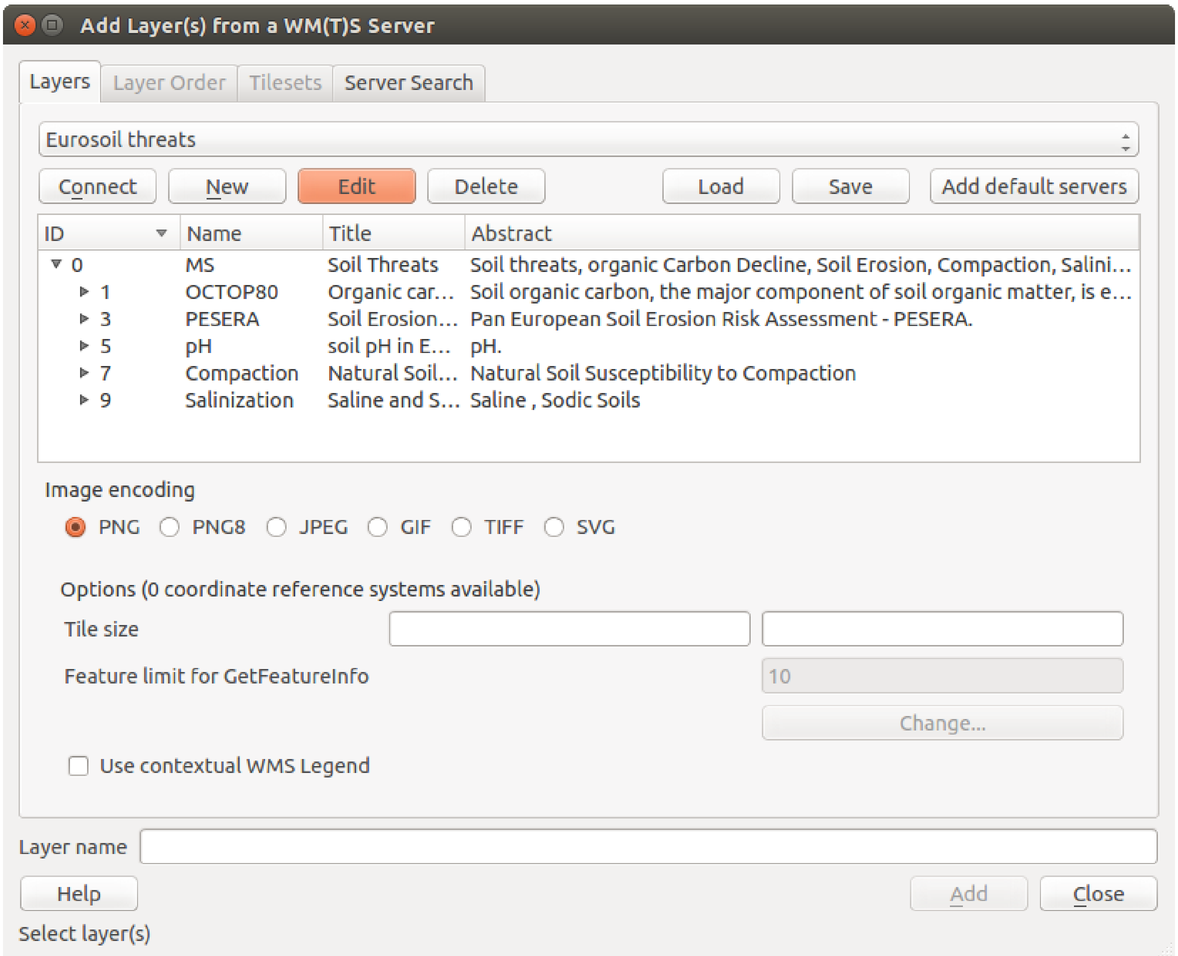


Abb. 18.2: Dialog zum Hinzufügen eines WMS Servers bei dem die erhältlichen Layer gezeigt werden

Schließlich können Sie ☑ *Use contextual WMS-Legend* aktivieren, wenn der WMS Server diese Funktion unterstützt. Dann wird nur die entsprechende Legende für Ihre aktuellen Ausmaße der Kartenansicht angezeigt und somit keine Legendenelemente die Sie nicht in der aktuellen Karte sehen können.

Layerreihenfolge

Der Reiter *Layerreihenfolge* listet die vom gerade verbundenen WMS Server ausgewählten Layer auf. Sie stellen vielleicht fest dass einige Layer ausklappbar sind. Das bedeutet dass der Layer in einer Auswahl von Bildstilen dargestellt werden kann.

Sie können mehrere Layer auf einmal auswählen aber nur einen Bildstil pro Layer. Wenn mehrere Layer ausgewählt sind werden Sie am WMS Server kombiniert und in einem Rutsch an QGIS weitergegeben.

Tipp: WMS Layer anordnen

Von einem Server dargestellte WMS Layer werden in der Reihenfolge aus dem Abschnitt Layers von oben bis unten überlagert. Wenn Sie die Layerreihenfolge ändern wollen können Sie den Reiter *Layerreihenfolge* benutzen.

Transparenz

In dieser Version von QGIS ist die Einstellung *Globale Transparenz* aus dem Menü *Layereigenschaften* immer gemäß den Layereigenschaften eingestellt.

Tipp: Transparenz von WMS-Layern

Die WMS Bildtransparenz steht Ihnen abhänig von der Bildkodierung zur Verfügung: PNG und GIF unterstützen Transparenz währenddessen JPEG keine Untersützung bietet.

Koordinatenbezugsystem

Koordinatenbezugssystem (KBS) ist die Bezeichnung des OGC für eine Projektion in QGIS.

Jeder WMS Layer kann abhängig von den Fähigkeiten des WMS in mehreren KBS dargestellt werden.

To choose a CRS, select *Change…* and a dialog similar to the one shown in Abb. 10.3 will appear. The main difference with the WMS version of the dialog is that only those CRSs supported by the WMS server will be shown.

Serversuche

Within QGIS, you can search for WMS servers. *Figure_OGC_search* shows the *Server Search* tab with the *Add Layer(s) from a Server* dialog.

As you can see, it is possible to enter a search string in the text field and hit the *Search* button. After a short while, the search result will be populated into the list below the text field. Browse the result list and inspect your search results within the table. To visualize the results, select a table entry, press the *Add Selected Row to WMS List* button and change back to the *Layers* tab. QGIS has automatically updated your server list, and the selected search result is already enabled in the list of saved WMS servers in the *Layers* tab. You only need to request the list of layers by clicking the *Connect* button. This option is quite handy when you want to search maps by specific keywords.

Diese Suchfunktion ist ein Frontend zur API von http://geopole.org.

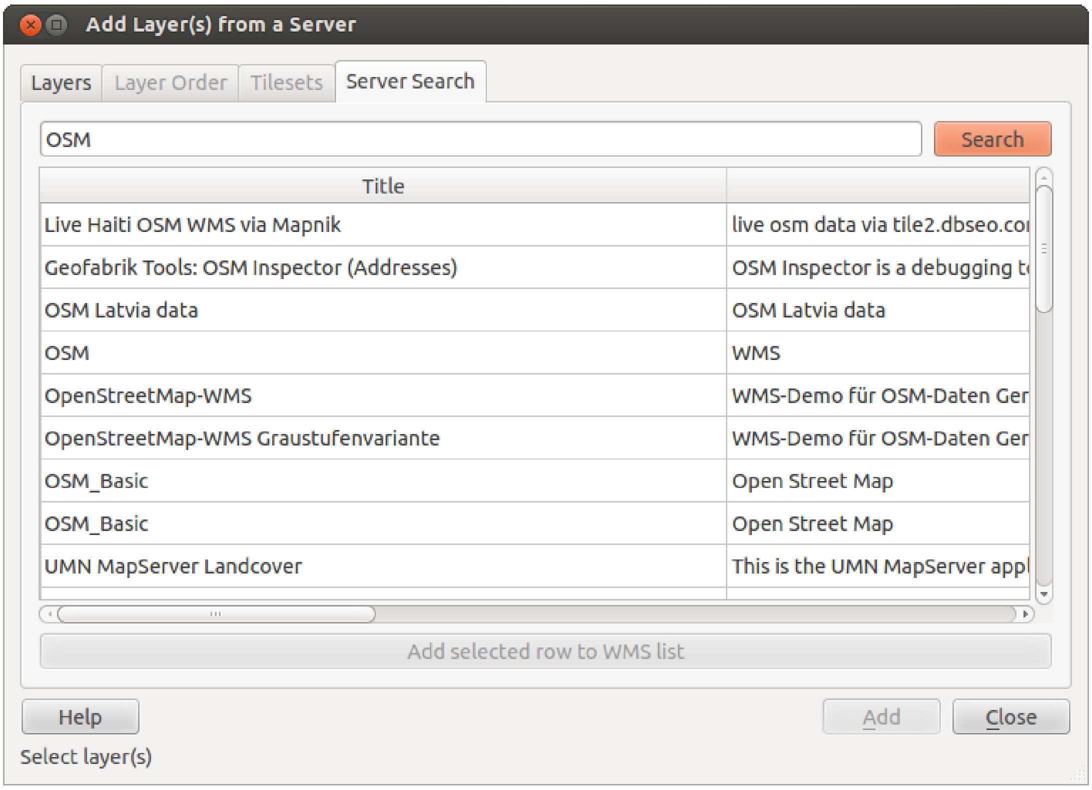


Abb. 18.3: Dialog zum Suchen von WMS Servern nach einigen Stichwörtern

Tilesets

Wenn Sie WMTS (gecachte WMS) Dienste verwenden wie

```
https://opencache.statkart.no/gatekeeper/gk/gk.open_wmts?\
  service=WMTS&request=GetCapabilities
```

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. 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.

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 Unterstüzung 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 send 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"
```

Eigenschaften

Nachdem Sie einen WMS Server hinzugefügt haben können Sie sich seine Eigenschaften mit einem Rechtsklick in der Legende und dem Auswählen von *Eigenschaften* ansehen.

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.

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.

Einschränkungen des WMS-Klienten

Nicht alle mögliche WMS Client Funktionalitäten sind in diese Version von QGIS| integriert worden. Einige der bemerkenswerteren Ausnahmen folgen noch.

WMS-Layereigenschaften ändern

Once you've completed the Add WMS layer procedure, there is no way to change the settings. A work-around is to delete the layer completely and start again.

WMS-Server, die eine Authentifizierung benötigen

Derzeit werden öffentlich zugängliche und gesicherte WMS Services unterstützt. Die gesicherten WMS Server können mit öffentlicher Authentifizierung angebunden werden. Sie können die (optionalen) Anmeldeinformationen hinzufügen wenn Sie einen WMS Server hinzufügen. Schlagen Sie unter Abschnitt *WMS/WMTS Server auswählen* Details nach.

Tipp: Zugriff auf abgesicherte OGC-Layer

If you need to access secured layers with secured methods other than basic authentication, you can use InteProxy as a transparent proxy, which does support several authentication methods. More information can be found in the InteProxy manual at https://inteproxy.wald.intevation.org.

Tipp: QGIS WMS Mapserver

Seit Version 1.7.0 besitzt QGIS seine eigene Implementierung eines WMS 1.3.0 Mapservers. Lesen mehr darüber in Kapitel *QGIS as OGC Data Server*.

18.1.2 WCS Client

Ein Web Coverage Service (WCS) stellt eine Anbindung zu Rasterdaten in Formaten die nützlich für die clientseitige 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 cachen', ‚Cache vorziehen', ‚Netzwerk vorziehen', ‚Immer Netzwerk') und der Provider unterstützt ebenfalls das Auswählen der Zeitposition wenn Temporal Domain vom Server angeboten wird.

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 a *Authentication configuration* instead (*configurations* tab). See *Authentifizierungssystem* for more details.

18.1.3 WFS und WFS-T Klient

In QGIS verhält sich ein WFS-Layer weitestgehend wie ein anderer Vektorlayer. Sie können Objekte abfragen, auswählen und sich die Attributtabelle anschauen. Seit QGIS 1.6 wird das Bearbeiten (WFS-T) unterstützt.

Im Allgemeinen verhält sich das Hinzufügen eines WFS Layers sehr ähnlich wie die Vorgehensweise die beim WMS verwendet wird. Der Unterschied besteht darin dass keine voreingestellten Server definiert sind, also müssen wir eigene hinzufügen.

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 we create a connection to the WFS server first.

1. Open the *Data Source Manager* dialog by pressing the Open Data Source Manager button
2. Enable the *WFS* 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)
6. In the WFS settings dialog, you can:

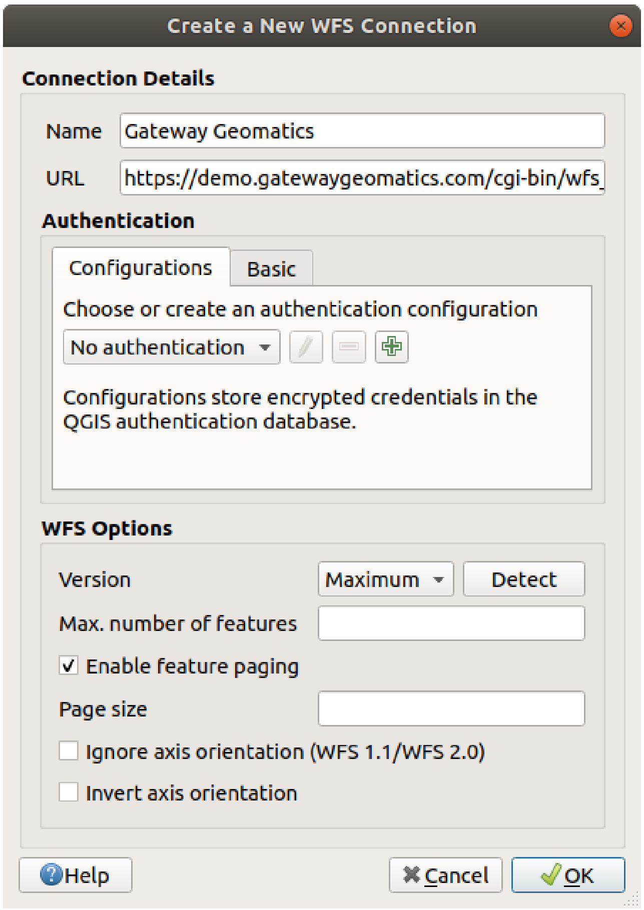


Abb. 18.4: Creating a connection to a WFS server

- 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 GetFetFeature request. If empty, no limit is set.
- *Invert axis orientation*.
- And depending on the WFS version:
  - Force to *Ignore axis orientation (WFS 1.1/WFS 2.0)*
  - *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.

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.

Beachten Sie dass auch jede Proxyeinstellung die Sie in Ihren eingestellt haben berücksichtigt wird.

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
   - or *Build query* to specify particular features to retrieve, by either using the corresponding button or double-clicking the target layer.
5. Click *Add* to add the layer to the map.

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.

Bemerkung: QGIS supports different versions of the WFS protocol, with background download and progressive rendering, on-disk caching of downloaded features and version autodetection.

Tipp: WFS-Server finden

Sie können weitere WFS Server mit Hilfe von Google oder ihrer bevorzugten Suchmaschine finden. Es gibt eine Vielzahl von Listen im Internet, die Links zu öffentlichen Servern bereitstellen.

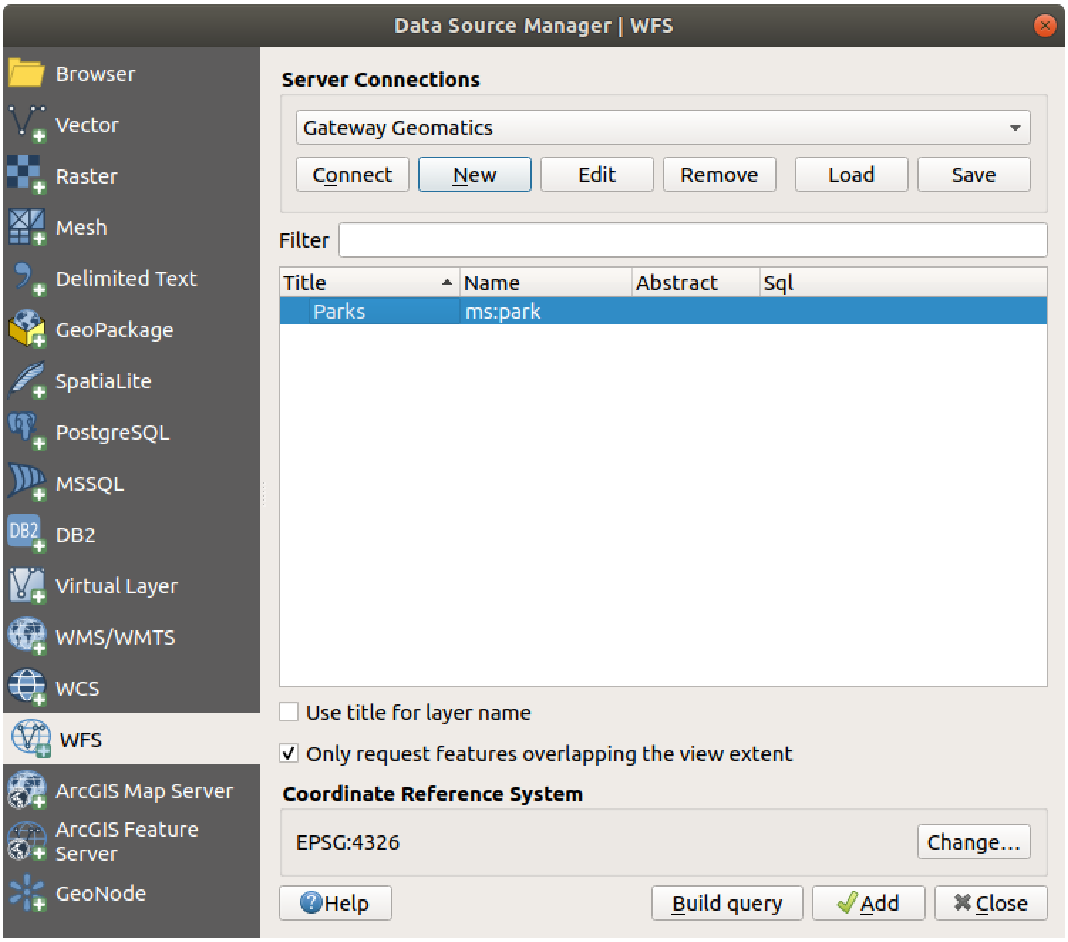


Abb. 18.5: Einen WFS Layer hinzufügen

18.2 QGIS as OGC Data Server

QGIS Server is an open source WMS 1.3, WFS 1.0.0, WFS 1.1.0 and WCS 1.1.1 implementation that, in addition, implements advanced cartographic features for thematic mapping. QGIS Server is a FastCGI/CGI (Common Gateway Interface) application written in C++ that works together with a web server (e.g., Apache, Nginx). It has Python plugin support allowing for fast and efficient development and deployment of new features.

QGIS Server uses QGIS as back end for the GIS logic and for map rendering. Furthermore, the Qt library is used for graphics and for platform-independent C++ programming. In contrast to other WMS software, the QGIS Server uses cartographic rules as a configuration language, both for the server configuration and for the user-defined cartographic rules.

As QGIS desktop and QGIS Server use the same visualization libraries, the maps that are published on the web look the same as in desktop GIS.

In the following sections, we will provide a sample configuration to set up a QGIS Server on Linux (Debian, Ubuntu and derivatives) and on Windows. For more information about server plugin development, please read server_plugins.

18.2.1 Der erste Einstieg

Installation on Debian-based systems

We will give a short and simple installation how-to for a minimal working configuration on Debian based systems (including Ubuntu and derivatives). However, many other distributions and OSs provide packages for QGIS Server.

Requirements and steps to add official QGIS repositories to install current QGIS Server on a Debian based system are provided in QGIS installers page.

Bemerkung: In Ubuntu you can use your regular user, prepending `sudo` to commands requiring admin permissions. In Debian you can work as admin (`root`), without using `sudo`.

We strongly suggest installing the LTR version.

Once the chosen repository is configured, installation is simply done with:

```
apt install qgis-server
# if you want to install server plugins, also:
apt install python-qgis
```

You can test the installation by running:

```
/usr/lib/cgi-bin/qgis_mapserv.fcgi
```

If you get the following output, the server is correctly installed:

```
QFSFileEngine::open: No file name specified
Warning 1: Unable to find driver ECW to unload from GDAL_SKIP environment variable.
Warning 1: Unable to find driver ECW to unload from GDAL_SKIP environment variable.
Warning 1: Unable to find driver JP2ECW to unload from GDAL_SKIP environment␣
↪variable.
Warning 1: Unable to find driver ECW to unload from GDAL_SKIP environment variable.
Warning 1: Unable to find driver JP2ECW to unload from GDAL_SKIP environment␣
↪variable.
Content-Length: 206
Content-Type: text/xml; charset=utf-8

<ServiceExceptionReport version="1.3.0" xmlns="https://www.opengis.net/ogc">
 <ServiceException code="Service configuration error">Service unknown or␣
↪unsupported</ServiceException>
</ServiceExceptionReport>
```

Let's add a sample project. You can use your own, or one from Training demo data:

```
mkdir /home/qgis/projects/
cd /home/qgis/projects/
wget https://github.com/qgis/QGIS-Training-Data/archive/v2.0.zip
unzip v2.0.zip
mv QGIS-Training-Data-2.0/exercise_data/qgis-server-tutorial-data/world.qgs .
mv QGIS-Training-Data-2.0/exercise_data/qgis-server-tutorial-data/naturalearth.
↪sqlite .
```

Of course, you can use your favorite GIS software to open this file and take a look on the configuration and available layers.

HTTP Server configuration

To run QGIS server you need a web server. Recommended choices are **Apache** or **Nginx**.

Bemerkung: In the following, please replace `localhost` with the name or IP address of your server.

Apache

Install Apache and mod_fcgid:

```
apt install apache2 libapache2-mod-fcgid
a2enmod cgi
```

QGIS Server is now available at http://localhost/. To check, type in a browser:

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?SERVICE=WMS&VERSION=1.3.0&
↪REQUEST=GetCapabilities
```

If you get something like:

```
<WMS_Capabilities version="1.3.0" xsi:schemaLocation="http://www.opengis.net/wms␣
↪http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd http://www.opengis.
↪net/sld http://schemas.opengis.net/sld/1.1.0/sld_capabilities.xsd http://www.
↪qgis.org/wms http://localhost/cgi-bin/qgis_mapserv.fcgi?SERVICE=WMS&
↪REQUEST=GetSchemaExtension">
...
```

the server is correctly installed and responds through Apache.

Let's now add `mod_fcgid` configuration directives for QGIS Server:

```
# Tell QGIS Server instances to use a specific display number for xvfb
# necessary for printing, see below
FcgidInitialEnv DISPLAY ":99"
# Activate QGIS log (different from apache logs)
FcgidInitialEnv QGIS_SERVER_LOG_FILE /var/log/qgis/qgisserver.log
FcgidInitialEnv QGIS_SERVER_LOG_LEVEL "0"
FcgidInitialEnv QGIS_SERVER_LOG_STDERR "1"
FcgidInitialEnv QGIS_DEBUG 1
# Add a default QGIS project
SetEnv QGIS_PROJECT_FILE /home/qgis/projects/world.qgs
# QGIS_AUTH_DB_DIR_PATH must lead to a directory writeable by www-data
FcgidInitialEnv QGIS_AUTH_DB_DIR_PATH "/var/www/qgis-server/qgisserverdb/"
FcgidInitialEnv QGIS_AUTH_PASSWORD_FILE "/var/www/qgis-server/qgisserverdb/qgis-
↪auth.db"
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
<IfModule mod_fcgid.c>
# Longer timeout for WPS... default = 40
FcgidIOTimeout 120
FcgidMaxRequestLen 26214400
FcgidConnectTimeout 60
</IfModule>
```

Bemerkung: See the `mod_fcgid` documentation for more information on the `Fcgid` parameters used. And see below (`xvfb`) to understand when and why the `DISPLAY` environment variable needs to be set.

These directives can be added either to `/etc/apache2/mods-enabled/fcgid.conf` for a system-wide configuration, or to a specific Apache `VirtualHost`, if you want QGIS server to be available only for that address; the default one is available at `/etc/apache2/sites-available/000-default.conf`.

```
<Location /qgisserver>
 SetHandler fcgid-script
 FcgidWrapper /usr/lib/cgi-bin/qgis_mapserv.fcgi virtual
 Options +ExecCGI -MultiViews +FollowSymLinks
 Require all granted
</Location>
```

Then create all the needed directories with appropriate permissions:

```
mkdir -p /var/log/qgis/
chown www-data:www-data /var/log/qgis
mkdir -p /var/www/qgis-server/qgisserverdb/
chown www-data:www-data /var/www/qgis-server/qgisserverdb/
```

Now restart Apache for the new configuration to be taken into account:

```
systemctl restart apache2
```

QGIS Server is now available at http://localhost/qgisserver. To check, type in a browser, as in the simple case:

```
http://localhost/qgisserver/cgi-bin/qgis_mapserv.fcgi?SERVICE=WMS&VERSION=1.3.0&
↪REQUEST=GetCapabilities
```

NGINX

You can also use QGIS Server with NGINX. Unlike Apache, NGINX does not automatically spawn FastCGI processes. The FastCGI processes are to be started by something else.

On Debian-based systems, you can use **spawn-fcgi** or **fcgiwrap** to start and manage the QGIS Server processes. Official Debian packages exist for both.

Warnung: **fcgiwrap** is easier to set up than **spawn-fcgi**, because it's already wrapped in a Systemd service. But it also leads to a solution that is much slower than using spawn-fcgi. With fcgiwrap a new QGIS Server process is created on each request, meaning that the QGIS Server initialization process, which includes reading and parsing the QGIS project file, is done on each request. With spawn-fcgi, the QGIS Server process remains alive between requests, resulting in much better performance. For that reason, spawn-fcgi is recommended for production use.

Another option is to rely on **Systemd**, the init system for GNU/Linux that most Linux distributions use today. One of the advantages of this method is that it requires no other components or processes. It's meant to be simple, yet robust and efficient for production deployments.

Install NGINX:

```
apt install nginx
```

spawn-fcgi

If you want to use spawn-fcgi, the first step is to install the package:

```
apt install spawn-fcgi
```

Then, introduce the following block in your NGINX server configuration:

```
location /qgisserver {
    gzip           off;
    include        fastcgi_params;
    fastcgi_pass   unix:/var/run/qgisserver.socket;
}
```

And restart NGINX to take into account the new configuration:

```
service nginx restart
```

Finally, considering that there is no default service file for spawn-fcgi, you have to manually start QGIS Server in your terminal:

```
spawn-fcgi -s /var/run/qgisserver.socket \
                -U www-data -G www-data -n \
                /usr/lib/cgi-bin/qgis_mapserv.fcgi
```

todo: Add instructions to add a `spawn-fcgi.service`

QGIS Server is now available at http://localhost/qgisserver.

Bemerkung: With the above command spawn-fcgi spawns only one QGIS Server process. To use more than one QGIS Server process you can combine spawn-fcgi with the multiwatch tool, which is also packaged in Debian.

Of course, you can add an init script (like a `qgis-server.service` file with systemd) to start QGIS Server at boot time or whenever you want.

todo: Add instructions to add a `qgis-server.service`

fcgiwrap

Using fcgiwrap is much easier to setup than **spawn-fcgi** but it's much slower. You first have to install the corresponding package:

```
apt install fcgiwrap
```

Then, introduce the following block in your NGINX server configuration:

```
location /qgisserver {
    gzip           off;
    include        fastcgi_params;
    fastcgi_pass   unix:/var/run/fcgiwrap.socket;
    fastcgi_param  SCRIPT_FILENAME /usr/lib/cgi-bin/qgis_mapserv.fcgi;
}
```

Finally, restart NGINX and **fcgiwrap** to take into account the new configuration:

```
service nginx restart
service fcgiwrap restart
```

QGIS Server is now available at http://localhost/qgisserver.

Configuration

The **include fastcgi_params;** used in previous configuration is important as it adds the parameters from `/etc/nginx/fastcgi_params`:

```
fastcgi_param  QUERY_STRING       $query_string;
fastcgi_param  REQUEST_METHOD     $request_method;
fastcgi_param  CONTENT_TYPE       $content_type;
fastcgi_param  CONTENT_LENGTH     $content_length;

fastcgi_param  SCRIPT_NAME        $fastcgi_script_name;
fastcgi_param  REQUEST_URI        $request_uri;
fastcgi_param  DOCUMENT_URI       $document_uri;
fastcgi_param  DOCUMENT_ROOT      $document_root;
fastcgi_param  SERVER_PROTOCOL    $server_protocol;
fastcgi_param  REQUEST_SCHEME     $scheme;
fastcgi_param  HTTPS              $https if_not_empty;

fastcgi_param  GATEWAY_INTERFACE  CGI/1.1;
fastcgi_param  SERVER_SOFTWARE    nginx/$nginx_version;

fastcgi_param  REMOTE_ADDR        $remote_addr;
fastcgi_param  REMOTE_PORT        $remote_port;
fastcgi_param  SERVER_ADDR        $server_addr;
fastcgi_param  SERVER_PORT        $server_port;
fastcgi_param  SERVER_NAME        $server_name;

# PHP only, required if PHP was built with --enable-force-cgi-redirect
fastcgi_param  REDIRECT_STATUS    200;
```

Of course, you may override these variables in your own configuration. For example:

```
include fastcgi_params;
fastcgi_param SERVER_NAME domain.name.eu;
```

Moreover, you can use some *Environment variables* to configure QGIS Server. With NGINX as HTTP Server, you have to use `fastcgi_param` to define these variables as shown below:

```
fastcgi_param  QGIS_DEBUG              1;
fastcgi_param  QGIS_SERVER_LOG_FILE    /var/log/qgis/qgisserver.log;
fastcgi_param  QGIS_SERVER_LOG_LEVEL   0;
```

Bemerkung: When using spawn-fcgi, you may directly define environment variables before running the server. For example: `export QGIS_SERVER_LOG_FILE=/var/log/qgis/qgisserver.log`

Systemd

This method to deploy QGIS Server relies on two Systemd units:

- a Socket unit
- and a Service unit.

The **QGIS Server Socket unit** defines and creates a file system socket, used by NGINX to start and communicate with QGIS Server. The Socket unit has to be configured with `Accept=false`, meaning that the calls to the `accept()` system call are delegated to the process created by the Service unit. It is located in `/etc/systemd/system/qgis-server@.socket`, which is actually a template:

```
[Unit]
Description=QGIS Server Listen Socket (instance %i)

[Socket]
Accept=false
ListenStream=/var/run/qgis-server-%i.sock
SocketUser=www-data
SocketGroup=www-data
SocketMode=0600

[Install]
WantedBy=sockets.target
```

Now enable and start sockets:

```
systemctl enable qgis-server@1.socket
systemctl start qgis-server@1.socket
systemctl enable qgis-server@2.socket
systemctl start qgis-server@2.socket
systemctl enable qgis-server@3.socket
systemctl start qgis-server@3.socket
systemctl enable qgis-server@4.socket
systemctl start qgis-server@4.socket
```

The **QGIS Server Service unit** defines and starts the QGIS Server process. The important part is that the Service process' standard input is connected to the socket defined by the Socket unit. This has to be configured using `StandardInput=socket` in the Service unit configuration located in `/etc/systemd/system/qgis-server@.service`:

```
[Unit]
Description=QGIS Server Service (instance %i)

[Service]
User=www-data
Group=www-data
StandardOutput=null
StandardError=journal
StandardInput=socket
ExecStart=/usr/lib/cgi-bin/qgis_mapserv.fcgi
EnvironmentFile=/etc/qgis-server/env

[Install]
WantedBy=multi-user.target
```

Now start socket service:

```
sudo systemctl start qgis-server@sockets.service
```

Note that the QGIS Server *environment variables* are defined in a separate file, `/etc/qgis-server/env`. It could look like this:

```
QGIS_PROJECT_FILE=/etc/qgis/myproject.qgs
QGIS_SERVER_LOG_STDERR=1
QGIS_SERVER_LOG_LEVEL=3
```

Finally, introduce the NGINX configuration for this setup:

```
upstream qgis-server_backend {
   server unix:/var/run/qgis-server-1.sock;
   server unix:/var/run/qgis-server-2.sock;
   server unix:/var/run/qgis-server-3.sock;
   server unix:/var/run/qgis-server-4.sock;
}

server {
   …

   location /qgis {
       gzip off;
       include fastcgi_params;
       fastcgi_pass qgis-server_backend;
   }
}
```

Now restart NGINX for the new configuration to be taken into account:

```
service nginx restart
```

Thanks to Oslandia for sharing their tutorial.

Xvfb

QGIS Server needs a running X Server to be fully usable, in particular for printing. On servers it is usually recommended not to install it, so you may use `xvfb` to have a virtual X environment.

To install the package:

```
apt install xvfb
```

Then, according to your HTTP server, you should configure the **DISPLAY** parameter or directly use **xvfb-run**.

With Apache you just add to your `Fcgi` configuration (see above):

```
FcgidInitialEnv DISPLAY       ":99"
```

Create the service file:

```
sh -c \
"echo \
'[Unit]
Description=X Virtual Frame Buffer Service
After=network.target

[Service]
ExecStart=/usr/bin/Xvfb :99 -screen 0 1024x768x24 -ac +extension GLX +render -
↪noreset

[Install]
WantedBy=multi-user.target' \
> /etc/systemd/system/xvfb.service"
```

Enable, start and check the status of the `xvfb.service`:

```
systemctl enable xvfb.service
systemctl start xvfb.service
systemctl status xvfb.service
```

Now restart Apache for the new configuration to be taken into account:

```
systemctl restart apache2
```

With NGINX and spawn-fcgi using `xvfb-run`:

```
xvfb-run /usr/bin/spawn-fcgi -f /usr/lib/cgi-bin/qgis_mapserv.fcgi \
                             -s /tmp/qgisserver.socket \
                             -G www-data -U www-data -n
```

The other option is to start a virtual X server environment with a specific display number thanks to **Xvfb**:

```
/usr/bin/Xvfb :99 -screen 0 1024x768x24 -ac +extension GLX +render -noreset
```

Then we just have to set the **DISPLAY** environment variable in the HTTP server configuration. For example with NGINX:

```
fastcgi_param  DISPLAY       ":99";
```

Installation on Windows

QGIS Server can also be installed on Windows systems. While the QGIS Server package is available in the 64 bit version of the OSGeo4W network installer (https://qgis.org/en/site/forusers/download.html) there is no Apache (or other web server) package available, so this must be installed by other means.

A simple procedure is the following:

- Download the XAMPP installer (https://www.apachefriends.org/download.html) for Windows and install Apache
- Download the OSGeo4W installer, follow the „Advanced Install“ and install both the QGIS Desktop and QGIS Server packages
- Edit the httpd.conf file (`C:\xampp\apache\httpd.conf` if the default installation paths have been used) and make the following changes:

From:

```
ScriptAlias /cgi-bin/ "C:/xampp/cgi-bin/"
```

To:

```
ScriptAlias /cgi-bin/ "c:/OSGeo4W64/apps/qgis/bin/"
```

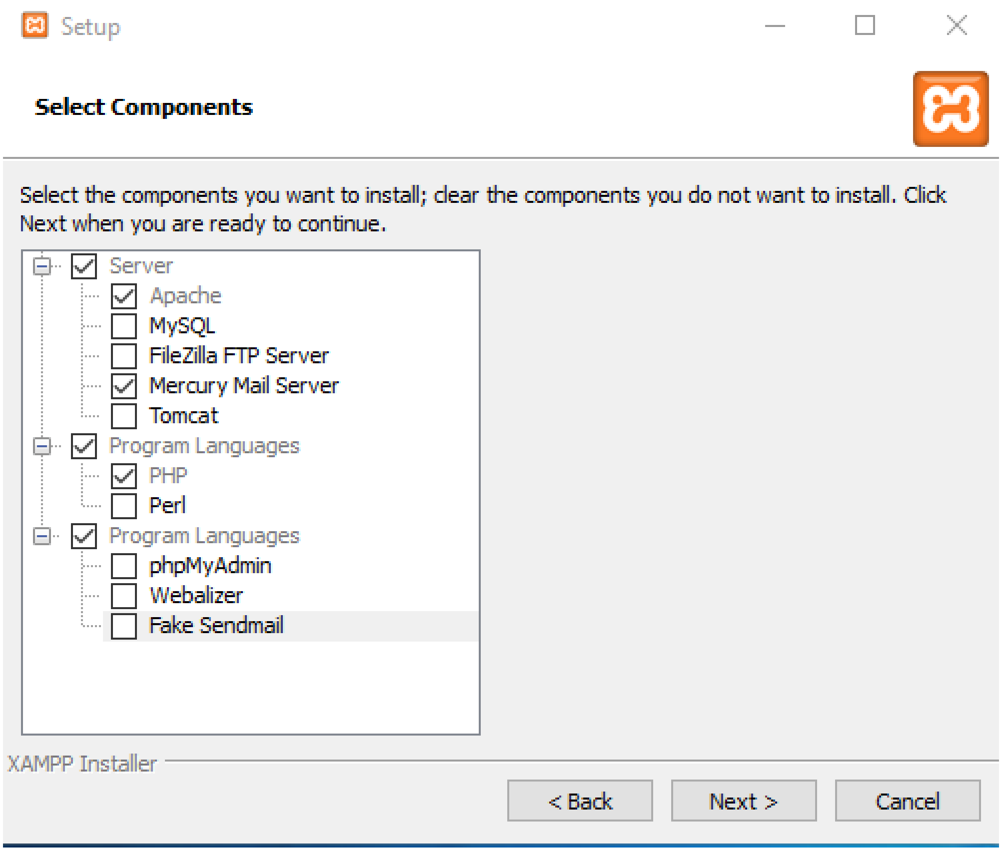
From:

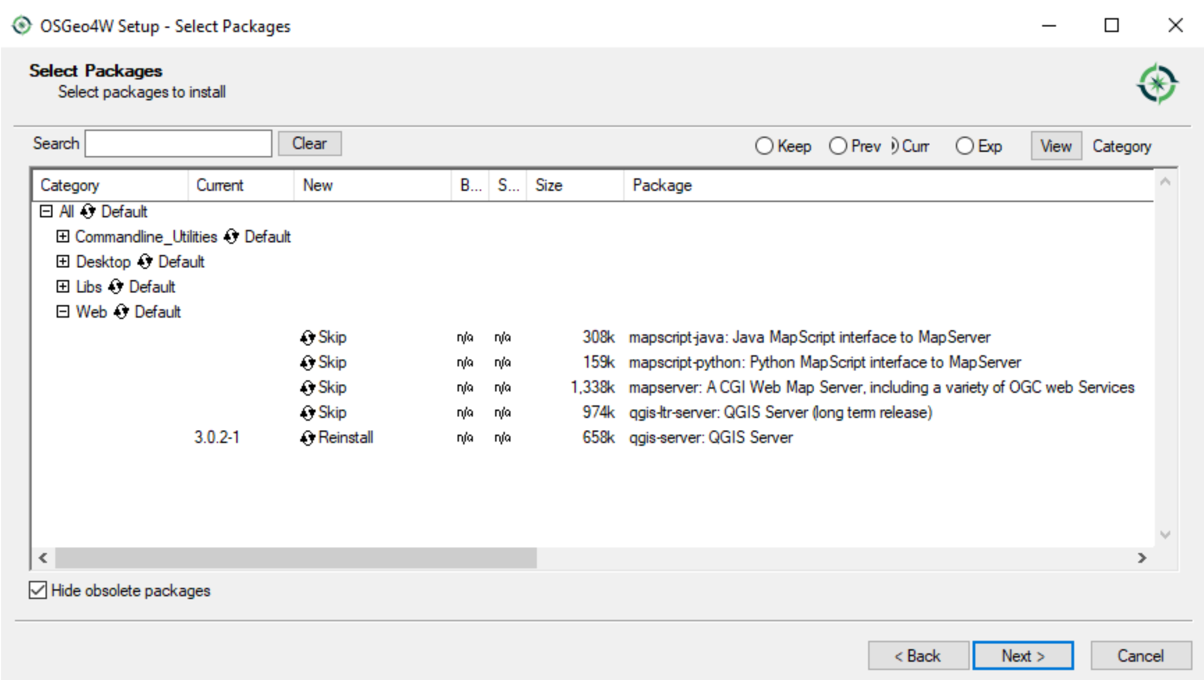
```
<Directory "C:/xampp/cgi-bin">
AllowOverride None
Options None
Require all granted
</Directory>
```

To:

```
<Directory "c:/OSGeo4W64/apps/qgis/bin">
SetHandler cgi-script
AllowOverride None
```

(Fortsetzung auf der nächsten Seite)


Setup
Select Components
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Server
Apache
MySQL
FileZilla FTP Server
Mercury Mail Server
Tomcat
Program Languages
PHP
Perl
Program Languages
phpMyAdmin
Webalizer
Fake Sendmail
XAMPP Installer
< Back
Next >
Cancel


OSGeo4W Setup - Select Packages
Select Packages
Select packages to install
Search
Clear
Keep
Prev
Curr
Exp
View
Category
Category
Current
New
B...
S...
Size
Package
All Default
Commandline_Utilities Default
Desktop Default
Libs Default
Web Default
Skip n/a n/a 308k mapscript-java: Java MapScript interface to MapServer
Skip n/a n/a 159k mapscript-python: Python MapScript interface to MapServer
Skip n/a n/a 1,338k mapserver: A CGI Web Map Server, including a variety of OGC web Services
Skip n/a n/a 974k qgis-ltr-server: QGIS Server (long term release)
3.0.2-1 Reinstall n/a n/a 658k qgis-server: QGIS Server
Hide obsolete packages
< Back
Next >
Cancel

(Fortsetzung der vorherigen Seite)

```
Options ExecCGI
Order allow,deny
Allow from all
Require all granted
</Directory>
```

From:

```
AddHandler cgi-script .cgi .pl .asp
```

To:

```
AddHandler cgi-script .cgi .pl .asp .exe
```

Then at the bottom of httpd.conf add:

```
SetEnv GDAL_DATA "C:\OSGeo4W64\share\gdal"
SetEnv QGIS_AUTH_DB_DIR_PATH "C:\OSGeo4W64\apps\qgis\resources"
SetEnv PYTHONHOME "C:\OSGeo4W64\apps\Python37"
SetEnv PATH "C:\OSGeo4W64\bin;C:\OSGeo4W64\apps\qgis\bin;C:\OSGeo4W64\apps\Qt5\bin;
↪C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem"
SetEnv QGIS_PREFIX_PATH "C:\OSGeo4W64\apps\qgis"
SetEnv QT_PLUGIN_PATH "C:\OSGeo4W64\apps\qgis\qtplugins;C:\OSGeo4W64\apps\Qt5\
↪plugins"
```

Restart the Apache web server from the XAMPP Control Panel and open browser window to testing a GetCapabilities request to QGIS Server

```
http://localhost/cgi-bin/qgis_mapserv.fcgi.exe?SERVICE=WMS&VERSION=1.3.0&
↪REQUEST=GetCapabilities
```

Serve a project

Now that QGIS Server is installed and running, we just have to use it.

Obviously, we need a QGIS project to work on. Of course, you can fully customize your project by defining contact information, precise some restrictions on CRS or even exclude some layers. Everything you need to know about that is described later in *Configure your project*.

But for now, we are going to use a simple project already configured and previously downloaded in `/home/qgis/projects/world.qgs`, as described above.

By opening the project and taking a quick look on layers, we know that 4 layers are currently available:

- airports
- places
- countries
- countries_shapeburst

You don't have to understand the full request for now but you may retrieve a map with some of the previous layers thanks to QGIS Server by doing something like this in your web browser to retrieve the *countries* layer:

```
http://localhost/qgisserver?
  MAP=/home/qgis/projects/world.qgs&
  LAYERS=countries&
  SERVICE=WMS&
  REQUEST=GetMap&
  CRS=EPSG:4326&
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
WIDTH=400&
HEIGHT=200
```

If you obtain the next image, then QGIS Server is running correctly:

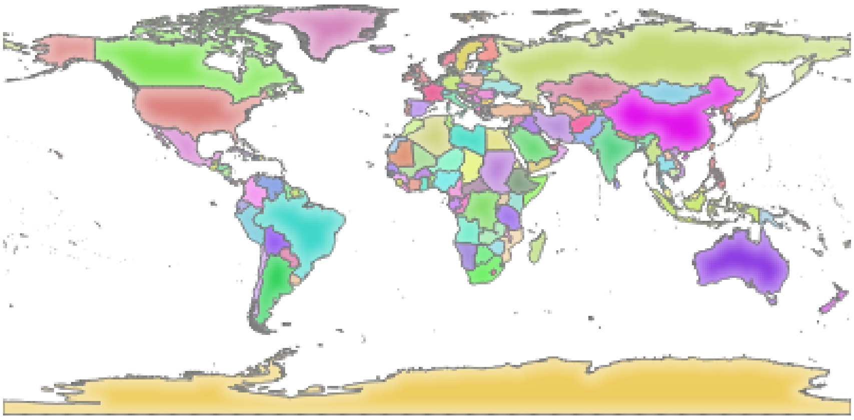


Abb. 18.6: Server response to a basic GetMap request

Note that you may define **PROJECT_FILE** environment variable to use a project by default instead of giving a **MAP** parameter (see *Environment variables*).

For example with spawn-fcgi:

```
export PROJECT_FILE=/home/qgis/projects/world.qgs
spawn-fcgi -f /usr/lib/bin/cgi-bin/qgis_mapserv.fcgi \
           -s /var/run/qgisserver.socket \
           -U www-data -G www-data -n
```

Configure your project

To provide a new QGIS Server WMS, WFS or WCS, you have to create a QGIS project file with some data or use one of your current project. Define the colors and styles of the layers in QGIS and the project CRS, if not already defined.

Then, go to the *QGIS Server* menu of the *Project ⁇ Properties…* dialog and provide some information about the OWS in the fields under *Service Capabilities*. This will appear in the GetCapabilities response of the WMS, WFS or WCS. If you don't check *Service capabilities*, QGIS Server will use the information given in the `wms_metadata.xml` file located in the `cgi-bin` folder.

WMS capabilities

In the *WMS capabilities* section, you can define the extent advertised in the WMS GetCapabilities response by entering the minimum and maximum X and Y values in the fields under *Advertised extent*. Clicking *Use Current Canvas Extent* sets these values to the extent currently displayed in the QGIS map canvas. By checking *CRS restrictions*, you can restrict in which coordinate reference systems (CRS) QGIS Server will offer to render maps. It is recommended that you restrict the offered CRS as this reduces the size of the WMS GetCapabilities response. Use the button below to select those CRSs from the Coordinate Reference System Selector, or click *Used* to add the CRSs used in the QGIS project to the list.

If you have print layouts defined in your project, they will be listed in the *GetProjectSettings* response, and they can be used by the GetPrint request to create prints, using one of the print layout layouts as a template. This is a QGIS-specific extension to the WMS 1.3.0 specification. If you want to exclude any print layout from being published by

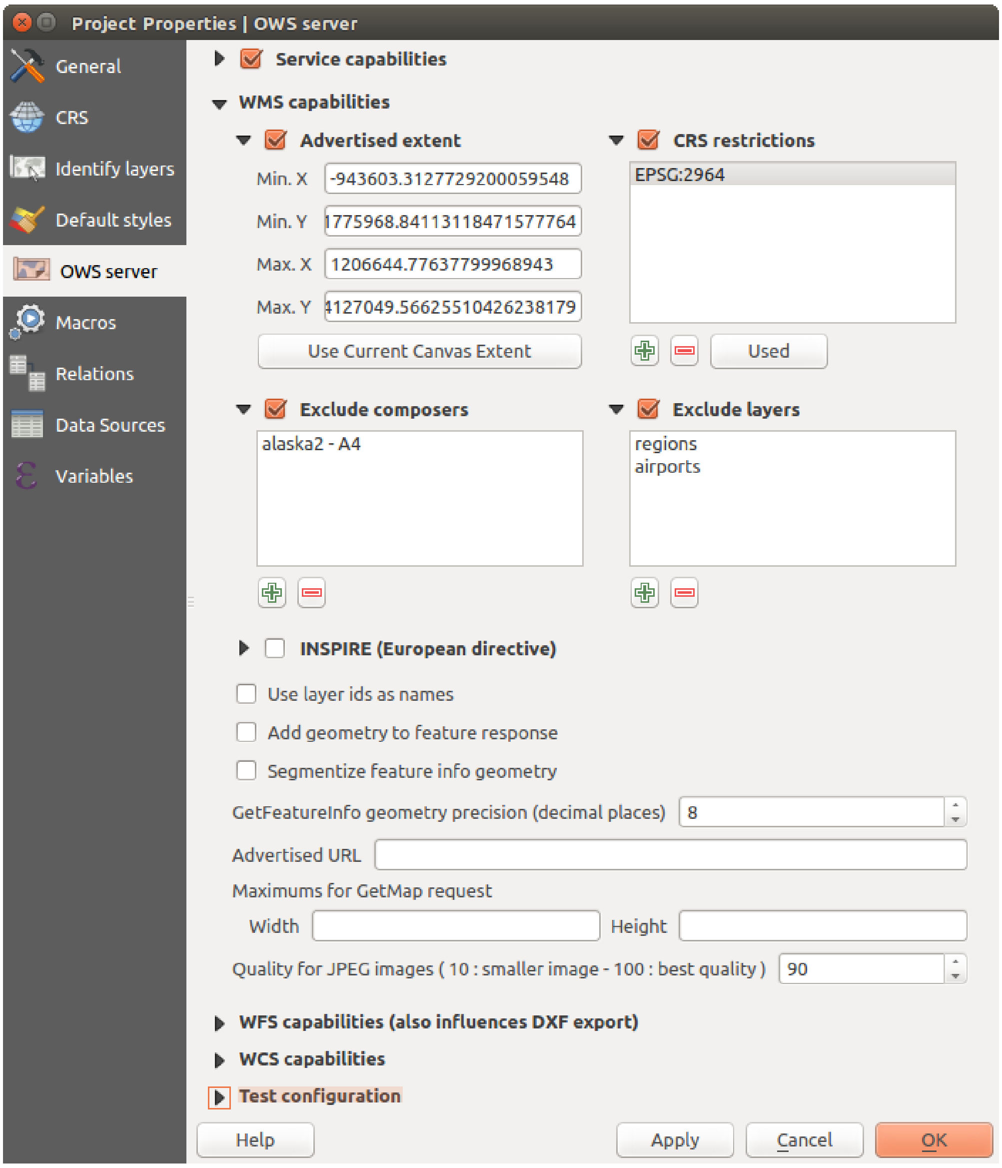


Abb. 18.7: Definitions for a QGIS Server WMS/WFS/WCS project

the WMS, check *Exclude layouts* and click the button below. Then, select a print layout from the *Select print layout* dialog in order to add it to the excluded layouts list.

If you want to exclude any layer or layer group from being published by the WMS, check *Exclude Layers* and click the button below. This opens the *Select restricted layers and groups* dialog, which allows you to choose the layers and groups that you don't want to be published. Use the `Shift` or `Ctrl` key if you want to select multiple entries. It is recommended that you exclude from publishing the layers that you don't need as this reduces the size of the WMS GetCapabilities response which leads to faster loading times on the client side.

You can receive requested GetFeatureInfo as plain text, XML and GML. Default is XML, text or GML format depends the output format chosen for the GetFeatureInfo request.

If you wish, you can check *Add geometry to feature response*. This will include the bounding box for each feature in the GetFeatureInfo response. See also the *WITH_GEOMETRY* parameter.

As many web clients can't display circular arcs in geometries you have the option to segmentize the geometry before sending it to the client in a GetFeatureInfo response. This allows such clients to still display a feature's geometry (e.g. for highlighting the feature). You need to check the *Segmentize feature info geometry* to activate the option.

You can also use the *GetFeatureInfo geometry precision* option to set the precision of the GetFeatureInfo geometry. This enables you to save bandwidth when you don't need the full precision.

If you want QGIS Server to advertise specific request URLs in the WMS GetCapabilities response, enter the corresponding URL in the *Advertised URL* field.

Furthermore, you can restrict the maximum size of the maps returned by the GetMap request by entering the maximum width and height into the respective fields under *Maximums for GetMap request*.

If one of your layers uses the *Map Tip display* (i.e. to show text using expressions) this will be listed inside the GetFeatureInfo output. If the layer uses a Value Map for one of its attributes, this information will also be shown in the GetFeatureInfo output.

WFS capabilities

In the *WFS capabilities* area you can select the layers you want to publish as WFS, and specify if they will allow update, insert and delete operations. If you enter a URL in the *Advertised URL* field of the *WFS capabilities* section, QGIS Server will advertise this specific URL in the WFS GetCapabilities response.

WCS capabilities

In the *WCS capabilities* area, you can select the layers that you want to publish as WCS. If you enter a URL in the *Advertised URL* field of the *WCS capabilities* section, QGIS Server will advertise this specific URL in the WCS GetCapabilities response.

Fine tuning your OWS

For vector layers, the *Fields* menu of the *Layer ▸ Layer Properties* dialog allows you to define for each attribute if it will be published or not. By default, all the attributes are published by your WMS and WFS. If you don't want a specific attribute to be published, uncheck the corresponding checkbox in the *WMS* or *WFS* column.

You can overlay watermarks over the maps produced by your WMS by adding text annotations or SVG annotations to the project file. See the *Beschriftungstools* section for instructions on creating annotations. For annotations to be displayed as watermarks on the WMS output, the *Fixed map position* checkbox in the *Annotation text* dialog must be unchecked. This can be accessed by double clicking the annotation while one of the annotation tools is active. For SVG annotations, you will need either to set the project to save absolute paths (in the *General* menu of the *Project ▸ Properties…* dialog) or to manually modify the path to the SVG image so that it represents a valid relative path.

18.2.2 Services

QGIS Server is able to serve data according to standard protocols as described by the **Open Geospatial Consortium (OGC)**:

- WMS 1.1.0 and 1.3.0
- WFS 1.0.0 and 1.1.0
- WFS3 (OGC API - Features)
- WCS 1.1.1
- WMTS 1.0.0

Extra vendor parameters and requests are supported in addition to the original standard that greatly enhance the possibilities of customizing its behavior thanks to the QGIS rendering engine.

Web Map Service (WMS)

The **1.1.0** and **1.3.0** WMS standards implemented in QGIS Server provide a HTTP interface to request map or legend images generated from a QGIS project. A typical WMS request defines the QGIS project to use, the layers to render as well as the image format to generate. Basic support is also available for **Styled Layer Descriptor (SLD)**.

Specifications:

- WMS 1.1.0
- WMS 1.3.0
- SLD 1.1.0 WMS profile

Standard requests provided by QGIS Server:

Request	Beschreibung
GetCapabilities	Returns XML metadata with information about the server
GetMap	Returns a map
GetFeatureInfo	Retrieves data (geometry and values) for a pixel location
GetLegendGraphics	Returns legend symbols

Vendor requests provided by QGIS Server:

Request	Beschreibung
GetPrint	Returns a QGIS composition
GetProjectSettings	Returns specific information about QGIS Server

GetMap

Standard parameters for the **GetMap** request according to the OGC WMS 1.1.0 and 1.3.0 specifications:

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service (WMS)
VERSION	No	Version of the service
REQUEST	Yes	Name of the request (GetMap)
LAYERS	No	Layers to display
STYLES	No	Layers' style
SRS / CRS	Yes	Coordinate reference system
BBOX	No	Map extent
WIDTH	Yes	Width of the image in pixels
HEIGHT	Yes	Height of the image in pixels
FORMAT	No	Image format
TRANSPARENT	No	Transparent background
SLD	No	URL of an SLD to be used for styling
SLD_BODY	No	In-line SLD (XML) to be used for styling

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file
BGCOLOR	No	Specify the background color
DPI	No	Specify the output resolution
IMAGE_QUALITY	No	JPEG compression
OPACITIES	No	Opacity for layer or group
FILTER	No	Subset of features
SELECTION	No	Highlight features
FILE_NAME	No	Only for `FORMAT=application/dxf` File name of the downloaded file
FORMAT_OPTIONS	No	Only for `FORMAT=application/dxf` key:value pairs separated by semicolon. • SCALE: to be used for symbology rules, filters and styles (not actual scaling of the data - data remains in the original scale). • MODE: corresponds to the export options offered in the QGIS Desktop DXF export dialog. Possible values are `NOSYMBOLOGY`, `FEATURESYMBOLOGY` and `SYMBOLLAYERSYMBOLOGY`. • LAYERSATTRIBUTES: specify a field that contains values for DXF layer names - if not specified, the original QGIS layer names are used. • USE_TITLE_AS_LAYERNAME: if enabled, the title of the layer will be used as layer name. • CODEC: specify a codec to be used for encoding. Default is `ISO-8859-1` check the QGIS desktop DXF export dialog for valid values.
TILED	No	Working in *tiled mode*

URL example:

```
http://localhost/qgis_server?
SERVICE=WMS
&VERSION=1.3.0
&REQUEST=GetMap
&MAP=/home/qgis/projects/world.qgs
&LAYERS=mylayer1,mylayer2,mylayer3
&STYLES=style1,default,style3
&OPACITIES=125,200,125
&CRS=EPSG:4326
&WIDTH=400
&HEIGHT=400
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
&FORMAT=image/png
&TRANSPARENT=TRUE
&DPI=300
&TILED=TRUE
```

SERVICE

This parameter has to be `WMS` in case of the **GetMap** request.

VERSION

This parameter allows to specify the version of the service to use. Available values for the `VERSION` parameter are:

- `1.1.0`
- `1.3.0`

If no version is indicated in the request, then `1.3.0` is used by default.

According to the version number, slight differences have to be expected as explained later for the next parameters:

- `CRS` / `SRS`
- `BBOX`

REQUEST

This parameter is `GetMap` in case of the **GetMap** request.

LAYERS

This parameter allows to specify the layers to display on the map. Names have to be separated by a comma.

In addition, QGIS Server introduced some options to select layers by:

- a short name
- the layer id

The short name of a layer may be configured through *Properties ▸ Metadata* in layer menu. If the short name is defined, then it's used by default instead of the layer's name:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&LAYERS=mynickname1,mynickname2
&...
```

Moreover, there's a project option allowing to select layers by their id in *OWS Server ▸ WMS capabilities* menu of the *Project ▸ Properties…* dialog. To activate this option, the checkbox *Use layer ids as names* has to be selected.

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&LAYERS=mylayerid1,mylayerid2
&...
```

STYLES

This parameter can be used to specify a layer's style for the rendering step. Styles have to be separated by a comma. The name of the default style is `default`.

SRS / CRS

This parameter allows to indicate the map output Spatial Reference System in WMS **1.1.0** and has to be formed like `EPSG:XXXX`. Note that `CRS` is also supported if current version is **1.1.0**.

For WMS **1.3.0**, `CRS` parameter is preferable but `SRS` is also supported.

Note that if both `CRS` and `SRS` parameters are indicated in the request, then it's the current version indicated in `VERSION` parameter which is decisive.

In the next case, the `SRS` parameter is kept whatever the `VERSION` parameter because `CRS` is not indicated:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.3.0
&SRS=EPSG:2854
&...
```

In the next case, the `SRS` parameter is kept instead of `CRS` because of the `VERSION` parameter:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.1.0
&CRS=EPSG:4326
&SRS=EPSG:2854
&...
```

In the next case, the `CRS` parameter is kept instead of `SRS` because of the `VERSION` parameter:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.3.0
&CRS=EPSG:4326
&SRS=EPSG:2854
&...
```

BBOX

This parameter allows to specify the map extent with units according to the current CRS. Coordinates have to be separated by a comma.

However, a slight difference has to be noticed according to the current `VERSION` parameter. In WMS **1.1.0**, coordinates are formed like `minx,miny,maxx,maxy` or `minlong,minlat,maxlong,maxlat`. For example:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.1.0
&SRS=epsg:4326
&BBOX=-180,-90,180,90
&...
```

But the axis is reversed in WMS **1.3.0**, so coordinates are formed like: `miny,minx,maxy,maxx` or `minlat,minlong,maxlat,maxlong`. For example:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.3.0
&CRS=epsg:4326
&BBOX=-90,-180,90,180
&...
```

WIDTH

This parameter allows to specify the width in pixels of the output image.

HEIGHT

This parameter allows to specify the height in pixels of the output image.

FORMAT

This parameter may be used to specify the format of map image. Available values are:

- `jpg`
- `jpeg`
- `image/jpeg`
- `image/png`
- `image/png; mode=1bit`
- `image/png; mode=8bit`
- `image/png; mode=16bit`
- `application/dxf` Only layers that have read access in the WFS service are exported in the DXF format.

  URL example:

  ```
  http://localhost/qgisserver?
  SERVICE=WMS&VERSION=1.3.0
  &REQUEST=GetMap
  &FORMAT=application/dxf
  &LAYERS=Haltungen,Normschacht,Spezialbauwerke
  &STYLES=
  &CRS=EPSG%3A21781&BBOX=696136.28844801,245797.12108743,696318.91114315,245939.
  ↪25832905
  &WIDTH=1042
  &HEIGHT=811
  &FORMAT_OPTIONS=MODE:SYMBOLLAYERSYMBOLOGY;SCALE:250&FILE_NAME=plan.dxf
  ```

TRANSPARENT

This boolean parameter can be used to specify the background transparency. Available values are (not case sensitive):

- `TRUE`
- `FALSE`

However, this parameter is ignored if the format of the map image indicated with `FORMAT` is different from PNG.

MAP

This parameter allows to define the QGIS project file to use.

As mentioned in *GetMap parameters table*, `MAP` is mandatory because a request needs a QGIS project to actually work. However, the `QGIS_PROJECT_FILE` environment variable may be used to define a default QGIS project. In this specific case, `MAP` is not longer a required parameter. For further information you may refer to *Advanced configuration*.

BGCOLOR

This parameter allows to indicate a background color for the map image. However it cannot be combined with `TRANSPARENT` parameter in case of PNG images (transparency takes priority). The colour may be literal or in hexadecimal notation.

URL example with the literal notation:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.3.0
&BGCOLOR=green
&...
```

URL example with the hexadecimal notation:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&VERSION=1.3.0
&BGCOLOR=0x00FF00
&...
```

DPI

This parameter can be used to specify the requested output resolution.

IMAGE_QUALITY

This parameter is only used for JPEG images. By default, the JPEG compression is `-1`.

You can change the default per QGIS project in the *OWS Server ▸ WMS capabilities* menu of the *Project ▸ Properties…* dialog. If you want to override it in a `GetMap` request you can do it using the `IMAGE_QUALITY` parameter.

OPACITIES

Opacity can be set on layer or group level. Allowed values range from 0 (fully transparent) to 255 (fully opaque).

FILTER

A subset of layers can be selected with the `FILTER` parameter. The syntax is basically the same as for the QGIS subset string. However, there are some restrictions to avoid SQL injections into databases via QGIS Server. If a dangerous string is found in the parameter, QGIS Server will return the next error:

```
Indeed, text strings need to be enclosed with quotes (single quotes
for strings, double quotes for attributes). A space between each
word / special character is mandatory. Allowed Keywords and special
characters are 'AND','OR','IN','=','<','>=','>','>=','!=*,'(',')'.
Semicolons in string expressions are not allowed.
```

URL example:

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&LAYERS=mylayer1,mylayer2,mylayer3
&FILTER=mylayer1:"col1";mylayer1,mylayer2:"col2" = 'blabla'
&...
```

In this example, the same filter (field `col2` equals the string `blabla`) is applied to layers `mylayer1` and `mylayer2`, while the filter on `col1` is only applied to `mylayer1`.

Bemerkung: It is possible to make attribute searches via GetFeatureInfo and omit the X/Y parameter if a FILTER is there. QGIS Server then returns info about the matching features and generates a combined bounding box in the XML output.

SELECTION

The `SELECTION` parameter can highlight features from one or more layers. Vector features can be selected by passing comma separated lists with feature ids.

```
http://localhost/qgisserver?
SERVICE=WMS
&REQUEST=GetMap
&LAYERS=mylayer1,mylayer2
&SELECTION=mylayer1:3,6,9;mylayer2:1,5,6
&...
```

The following image presents the response from a GetMap request using the `SELECTION` option e.g. `http://myserver.com/...&SELECTION=countries:171,65`.

As those features id's correspond in the source dataset to **France** and **Romania** they're highlighted in yellow.

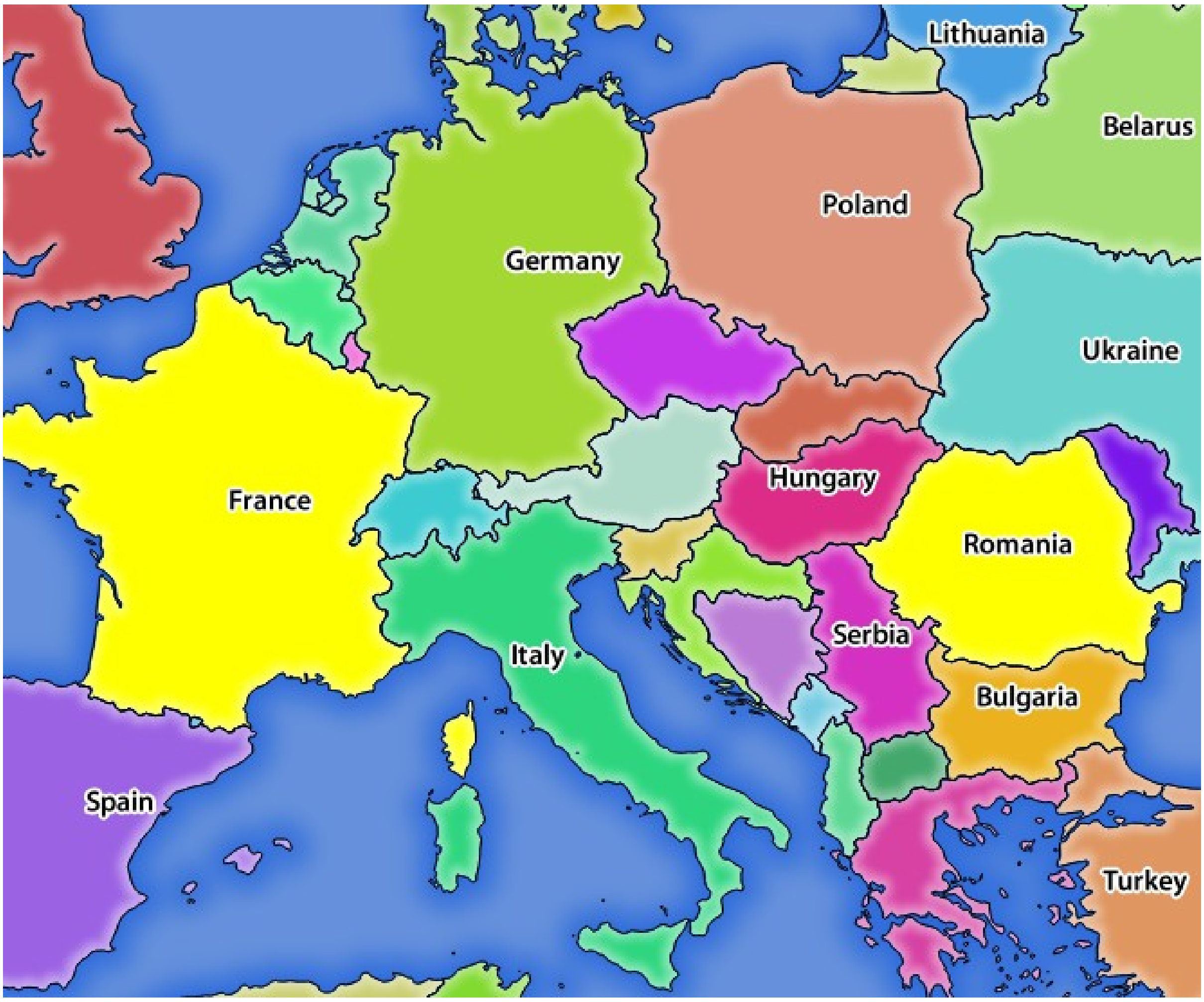


Abb. 18.8: Server response to a GetMap request with SELECTION parameter

TILED

Set the `TILED` parameter to `TRUE` to tell QGIS Server to work in *tiled* mode, and to apply the *Tile buffer* configured in the QGIS project.

When `TILED` is `TRUE` and when a non-zero Tile buffer is configured in the QGIS project, features outside the tile extent are drawn to avoid cut symbols at tile boundaries.

`TILED` defaults to `FALSE`.

GetFeatureInfo

Standard parameters for the **GetFeatureInfo** request according to the OGC WMS 1.1.0 and 1.3.0 specifications:

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service (WMS)
VERSION	No	*See GetMap*
REQUEST	Yes	*See GetMap*
LAYERS	No	*See GetMap*
STYLES	No	*See GetMap*
SRS / CRS	Yes	*See GetMap*
BBOX	No	*See GetMap*
WIDTH	Yes	*See GetMap*
HEIGHT	Yes	*See GetMap*
TRANSPARENT	No	*See GetMap*
INFO_FORMAT	No	Output format
QUERY_LAYERS	Yes	Layers to query
FEATURE_COUNT	No	Maximum number of features to return
I	No	Pixel column of the point to query
X	No	Same as *I* parameter, but in WMS 1.1.0
J	No	Pixel row of the point to query
Y	No	Same as *J* parameter, but in WMS 1.1.0

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	*See GetMap*
FILTER	No	*See GetMap*
FI_POINT_TOLERANCE	No	Tolerance in pixels for point layers
FI_LINE_TOLERANCE	No	Tolerance in pixels for line layers
FI_POLYGON_TOLERANCE	No	Tolerance in pixels for polygon layers
FILTER_GEOM	No	Geometry filtering
WITH_MAPTIP	No	Add map tips to the output
WITH_GEOMETRY	No	Add geometry to the output

URL example:

```
http://localhost/qgisserver?
SERVICE=WMS
&VERSION=1.3.0
&REQUEST=GetMap
&MAP=/home/qgis/projects/world.qgs
&LAYERS=mylayer1,mylayer2,mylayer3
&CRS=EPSG:4326
&WIDTH=400
&HEIGHT=400
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
&INFO_FORMAT=text/xml
&TRANSPARENT=TRUE
&QUERY_LAYERS=mylayer1
&FEATURE_COUNT=3
&I=250
&J=250
```

INFO_FORMAT

This parameter may be used to specify the format of the result. Available values are:

- `text/xml`
- `text/html`
- `text/plain`
- `application/vnd.ogc.gml`
- `application/json`

QUERY_LAYERS

This parameter specifies the layers to display on the map. Names are separated by a comma.

In addition, QGIS Server introduces options to select layers by:

- short name
- layer id

See the `LAYERS` parameter defined in *See GetMap* for more information.

FEATURE_COUNT

This parameter specifies the maximum number of features per layer to return. For example if `QUERY_LAYERS` is set to `layer1,layer2` and `FEATURE_COUNT` is set to `3` then a maximum of 3 features from layer1 will be returned. Likewise a maximun of 3 features from layer2 will be returned.

By default, only 1 feature per layer is returned.

I

This parameter, defined in WMS 1.3.0, allows you to specify the pixel column of the query point.

X

Same parameter as `I`, but defined in WMS 1.1.0.

J

This parameter, defined in WMS 1.3.0, allows you to specify the pixel row of the query point.

Y

Same parameter as `J`, but defined in WMS 1.1.0.

FI_POINT_TOLERANCE

This parameter specifies the tolerance in pixels for point layers.

FI_LINE_TOLERANCE

This parameter specifies the tolerance in pixels for line layers.

FI_POLYGON_TOLERANCE

This parameter specifies the tolerance in pixels for polygon layers.

FILTER_GEOM

This parameter specifies a WKT geometry with which features have to intersect.

WITH_MAPTIP

This parameter specifies whether to add map tips to the output.

Available values are (not case sensitive):

- `TRUE`
- `FALSE`

WITH_GEOMETRY

This parameter specifies whether to add geometries to the output. To use this feature you must first enable the *Add geometry to feature response* option in the QGIS project. See *Configure your project*.

Available values are (not case sensitive):

- `TRUE`
- `FALSE`

GetPrint

QGIS Server has the capability to create print layout output in pdf or pixel format. Print layout windows in the published project are used as templates. In the **GetPrint** request, the client has the possibility to specify parameters of the contained layout maps and labels.

Parameters for the **GetPrint** request:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file
SERVICE	Yes	Name of the service (WMS)
VERSION	No	*See GetMap*
REQUEST	Yes	Name of the request (GetPrint)
LAYERS	No	*See GetMap*
TEMPLATE	Yes	Layout template to use
SRS / CRS	Yes	*See GetMap*
FORMAT	Yes	Output format
ATLAS_PK	No	Atlas features
STYLES	No	*See GetMap*
TRANSPARENT	No	*See GetMap*
OPACITIES	No	*See GetMap*
SELECTION	No	*See GetMap*
mapX:EXTENT	No	Extent of the map ‚X‘
mapX:LAYERS	No	Layers of the map ‚X‘
mapX:STYLES	No	Layers‘ style of the map ‚X‘
mapX:SCALE	No	Layers‘ scale of the map ‚X‘
mapX:ROTATION	No	Rotation of the map ‚X‘
mapX:GRID_INTERVAL_X	No	Grid interval on x axis of the map ‚X‘
mapX:GRID_INTERVAL_Y	No	Grid interval on y axis of the map ‚X‘

URL example:

```
http://localhost/qgisserver?
SERVICE=WMS
&VERSION=1.3.0
&REQUEST=GetPrint
&MAP=/home/qgis/projects/world.qgs
&CRS=EPSG:4326
&FORMAT=png
&map0:EXTENT=-180,-90,180,90
&map0:LAYERS=mylayer1,mylayer2,mylayer3
&map0:OPACITIES=125,200,125
&map0:ROTATION=45
```

Note that the layout template may contain more than one map. In this way, if you want to configure a specific map, you have to use `mapX:` parameters where `X` is a positive number that you can retrieve thanks to the **GetProjectSettings** request.

For example:

```
<WMS_Capabilities>
...
<ComposerTemplates xsi:type="wms:_ExtendedCapabilities">
<ComposerTemplate width="297" height="210" name="Druckzusammenstellung 1">
<ComposerMap width="171" height="133" name="map0"/>
<ComposerMap width="49" height="46" name="map1"/></ComposerTemplate>
</ComposerTemplates>
...
</WMS_Capabilities>
```

SERVICE

This parameter has to be `WMS`.

REQUEST

This parameter has to be `GetPrint` for the **GetPrint** request.

TEMPLATE

This parameter can be used to specify the name of a layout template to use for printing.

FORMAT

This parameter specifies the format of map image. Available values are:

- `jpg`
- `jpeg`
- `image/jpeg`
- `png`
- `image/png`
- `svg`
- `image/svg`
- `image/svg+xml`
- `pdf`
- `application/pdf`

If the `FORMAT` parameter is different from one of these values, then an exception is returned.

ATLAS_PK

This parameter allows activation of Atlas rendering by indicating which features we want to print. In order to retrieve an atlas with all features, the `*` symbol may be used (according to the maximum number of features allowed in the project configuration).

When `FORMAT` is `pdf`, a single PDF document combining the feature pages is returned. For all other formats, a single page is returned.

mapX:EXTENT

This parameter specifies the extent for a layout map item as xmin,ymin,xmax,ymax.

mapX:ROTATION

This parameter specifies the map rotation in degrees.

mapX:GRID_INTERVAL_X

This parameter specifies the grid line density in the X direction.

mapX:GRID_INTERVAL_Y

This parameter specifies the grid line density in the Y direction.

mapX:SCALE

This parameter specifies the map scale for a layout map item. This is useful to ensure scale based visibility of layers and labels even if client and server may have different algorithms to calculate the scale denominator.

mapX:LAYERS

This parameter specifies the layers for a layout map item. See *See GetMap* for more information on this parameter.

mapX:STYLES

This parameter specifies the layers' styles defined in a specific layout map item. See *See GetMap* for more information on this parameter.

GetLegendGraphics

Several additional parameters are available to change the size of the legend elements:

- **BOXSPACE** space between legend frame and content (mm)
- **LAYERSPACE** vertical space between layers (mm)
- **LAYERTITLESPACE** vertical space between layer title and items following (mm)
- **SYMBOLSPACE** vertical space between symbol and item following (mm)
- **ICONLABELSPACE** horizontal space between symbol and label text (mm)
- **SYMBOLWIDTH** width of the symbol preview (mm)
- **SYMBOLHEIGHT** height of the symbol preview (mm)

These parameters change the font properties for layer titles and item labels:

- **LAYERFONTFAMILY / ITEMFONTFAMILY** font family for layer title / item text
- **LAYERFONTBOLD / ITEMFONTBOLD** `TRUE` to use a bold font
- **LAYERFONTSIZE / ITEMFONTSIZE** Font size in point
- **LAYERFONTITALIC / ITEMFONTITALIC** `TRUE` to use italic font
- **LAYERFONTCOLOR / ITEMFONTCOLOR** Hex color code (e.g. `#FF0000` for red)
- **LAYERTITLE** `FALSE` to get only the legend graphics without the layer title
- **RULELABEL**:

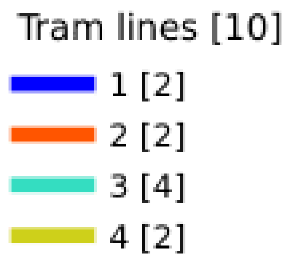
- `FALSE` legend graphics without item labels
- `AUTO` hide item label for layers with *Single symbol* rendering

Content based legend. These parameters let the client request a legend showing only the symbols for the features falling into the requested area:

- **BBOX** the geographical area for which the legend should be built
- **CRS / SRS** the coordinate reference system adopted to define the BBOX coordinates
- **WIDTH / HEIGHT** if set these should match those defined for the GetMap request, to let QGIS Server scale symbols according to the map view image size.

Content based legend features are based on the UMN MapServer implementation:

- **SHOWFEATURECOUNT** if set to `TRUE` adds in the legend the feature count of the features like in the following image:



GetProjectSettings

This request type works similar to **GetCapabilities**, but it is more specific to QGIS Server and allows a client to read additional information which is not available in the GetCapabilities output:

- initial visibility of layers
- information about vector attributes and their edit types
- information about layer order and drawing order
- list of layers published in WFS

Web Feature Service (WFS)

The **1.0.0** and **1.1.0** WFS standards implemented in QGIS Server provide a HTTP interface to query geographic features from a QGIS project. A typical WFS request defines the QGIS project to use and the layer to query.

Specifications document according to the version number of the service:

- WFS 1.0.0
- WFS 1.1.0

Standard requests provided by QGIS Server:

Request	Beschreibung
GetCapabilities	Returns XML metadata with information about the server
GetFeature	Returns a selection of features
DescribeFeatureType	Returns a description of feature types and properties
Transaction	Allows features to be inserted, updated or deleted

GetFeature

Standard parameters for the **GetFeature** request according to the OGC WFS 1.0.0 and 1.1.0 specifications:

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service
VERSION	No	Version of the service
REQUEST	Yes	Name of the request
TYPENAME	No	Name of layers
OUTPUTFORMAT	No	Output Format
RESULTTYPE	No	Type of the result
PROPERTYNAME	No	Name of properties to return
MAXFEATURES	No	Maximum number of features to return
SRSNAME	No	Coordinate reference system
FEATUREID	No	Filter the features by ids
FILTER	No	OGC Filter Encoding
BBOX	No	Map Extent
SORTBY	No	Sort the results

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file
STARTINDEX	No	Paging
GEOMETRYNAME	No	Type of geometry to return
EXP_FILTER	No	Expression filtering

SERVICE

This parameter has to be `WFS` in case of the **GetFeature** request.

For example:

```
http://localhost/qgisserver?
SERVICE=WFS
&...
```

VERSION

This parameter allows to specify the version of the service to use. Available values for the `VERSION` parameter are:

- `1.0.0`
- `1.1.0`

If no version is indicated in the request, then `1.1.0` is used by default.

URL example:

```
http://localhost/qgisserver?
SERVICE=WFS
&VERSION=1.1.0
&...
```

REQUEST

This parameter is `GetFeature` in case of the **GetFeature** request.

URL example:

```
http://localhost/qgisserver?
SERVICE=WFS
&VERSION=1.1.0
&REQUEST=GetFeature
&...
```

RESULTTYPE

This parameter may be used to specify the kind of result to return. Available values are:

- `results`: the default behavior
- `hits`: returns only a feature count

URL example:

```
http://localhost/qgisserver?
SERVICE=WFS
&VERSION=1.1.0
&REQUEST=GetFeature
&RESULTTYPE=hits
&...
```

GEOMETRYNAME

This parameter can be used to specify the kind of geometry to return for features. Available values are:

- `extent`
- `centroid`
- `none`

URL example:

```
http://localhost/qgisserver?
SERVICE=WFS
&VERSION=1.1.0
&REQUEST=GetFeature
&GEOMETRYNAME=centroid
&...
```

STARTINDEX

This parameter is standard in WFS 2.0, but it's an extension for WFS 1.0.0. Actually, it can be used to skip some features in the result set and in combination with `MAXFEATURES`, it provides the ability to page through results.

URL example:

```
http://localhost/qgisserver?
SERVICE=WFS
&VERSION=1.1.0
&REQUEST=GetFeature
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
&STARTINDEX=2
&...
```

Web Map Tile Service (WMTS)

The **1.0.0** WMTS standard implemented in QGIS Server provides a HTTP interface to request tiled map images generated from a QGIS project. A typical WMTS request defined the QGIS project to use, some WMS parameters like layers to render, as well as tile parameters.

Specifications document of the service:

- WMTS 1.0.0

Standard requests provided by QGIS Server:

Request	Beschreibung
GetCapabilities	Returns XML metadata with information about the server
GetTile	Returns a tile
GetFeatureInfo	Retrieves data (geometry and values) for a pixel location

GetCapabilities

Standard parameters for the **GetCapabilities** request according to the OGC WMTS 1.0.0 specifications:

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service (WMTS)
REQUEST	Yes	Name of the request (GetCapabilities)

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file

URL example:

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetCapabilities
&MAP=/home/qgis/projects/world.qgs
```

SERVICE

This parameter has to be `WMTS` in case of the **GetCapabilities** request.

REQUEST

This parameter is `GetCapabilities` in case of the **GetCapabilities** request.

MAP

This parameter allows to define the QGIS project file to use.

GetTile

Standard parameters for the **GetTile** request according to the OGC WMTS 1.0.0 specifications:

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service (WMTS)
REQUEST	Yes	Name of the request (GetTile)
LAYER	Yes	Layer identifier
FORMAT	Yes	Output format of the tile
TILEMATRIXSET	Yes	Name of the pyramid
TILEMATRIX	Yes	Meshing
TILEROW	Yes	Row coordinate in the mesh
TILECOL	Yes	Column coordinate in the mesh

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file

URL example:

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetTile
&MAP=/home/qgis/projects/world.qgs
&LAYER=mylayer
&FORMAT=image/png
&TILEMATRIXSET=EPSG:4326
&TILEROW=0
&TILECOL=0
```

SERVICE

This parameter has to be `WMTS` in case of the **GetTile** request.

REQUEST

This parameter is `GetTile` in case of the **GetTile** request.

LAYER

This parameter allows to specify the layer to display on the tile.

In addition, QGIS Server introduced some options to select a layer by:

- a short name
- the layer id

The short name of a layer may be configured through *Properties ▸ Metadata* in layer menu. If the short name is defined, then it's used by default instead of the layer's name:

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetTile
&LAYER=mynickname
&...
```

Moreover, there's a project option allowing to select layers by their id in *OWS Server ▸ WMS capabilities* menu of the *Project ▸ Project Properties* dialog. To activate this option, the checkbox *Use layer ids as names* has to be selected.

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetTile
&LAYER=mylayerid1
&...
```

FORMAT

This parameter may be used to specify the format of tile image. Available values are:

- `jpg`
- `jpeg`
- `image/jpeg`
- `image/png`

If the `FORMAT` parameter is different from one of these values, then the default format PNG is used instead.

TILEMATRIXSET

This parameter defines the CRS to use when computing the underlying pyramid. Format: `EPSG:XXXX`.

TILEMATRIX

This parameter allows to define the matrix to use for the output tile.

TILEROW

This parameter allows to select the row of the tile to get within the matrix.

TILECOL

This parameter allows to select the column of the tile to get within the matrix.

MAP

This parameter allows to define the QGIS project file to use.

As mentioned in *GetMap parameters table*, `MAP` is mandatory because a request needs a QGIS project to actually work. However, the `QGIS_PROJECT_FILE` environment variable may be used to define a default QGIS project. In this specific case, `MAP` is not longer a required parameter. For further information you may refer to *Advanced configuration*.

GetFeatureInfo

Standard parameters for the **GetFeatureInfo** request according to the OGC WMTS 1.0.0 specifications:

- WMS 1.1.0

Parameter	Required	Beschreibung
SERVICE	Yes	Name of the service (WMTS)
REQUEST	Yes	Name of the request (GetFeatureInfo)
LAYER	Yes	Layer identifier
INFOFORMAT	No	Output format
I	No	X coordinate of a pixel
J	No	Y coordinate of a pixel
TILEMATRIXSET	Yes	*See GetTile*
TILEMATRIX	Yes	*See GetTile*
TILEROW	Yes	*See GetTile*
TILECOL	Yes	*See GetTile*

In addition to the standard ones, QGIS Server supports the following extra parameters:

Parameter	Required	Beschreibung
MAP	Yes	Specify the QGIS project file

URL example:

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetFeatureInfo
&MAP=/home/qgis/projects/world.qgs
&LAYER=mylayer
&INFOFORMAT=image/html
&I=10
&J=5
```

SERVICE

This parameter has to be `WMTS` in case of the **GetFeatureInfo** request.

REQUEST

This parameter is `GetFeatureInfo` in case of the **GetFeatureInfo** request.

MAP

This parameter allows to define the QGIS project file to use.

As mentioned in *GetMap parameters table*, `MAP` is mandatory because a request needs a QGIS project to actually work. However, the `QGIS_PROJECT_FILE` environment variable may be used to define a default QGIS project. In this specific case, `MAP` is not longer a required parameter. For further information you may refer to *Advanced configuration*.

LAYER

This parameter allows to specify the layer to display on the tile.

In addition, QGIS Server introduced some options to select a layer by:

- a short name
- the layer id

The short name of a layer may be configured through *Properties ▶ Metadata* in layer menu. If the short name is defined, then it's used by default instead of the layer's name:

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetFeatureInfo
&LAYER=mynickname
&...
```

Moreover, there's a project option allowing to select layers by their id in *OWS Server ▶ WMS capabilities* menu of the *Project ▶ Project Properties* dialog. To activate this option, the checkbox *Use layer ids as names* has to be selected.

```
http://localhost/qgisserver?
SERVICE=WMTS
&REQUEST=GetFeatureInfo
&LAYER=mylayerid1
&...
```

INFOFORMAT

This parameter allows to define the output format of the result. Available values are:

- `text/xml`
- `text/html`
- `text/plain`
- `application/vnd.ogc.gml`

The default value is `text/plain`.

I

This parameter allows to define the X coordinate of the pixel for which we want to retrieve underlying information.

J

This parameter allows to define the Y coordinate of the pixel for which we want to retrieve underlying information.

WFS3 (OGC API Features)

WFS3 is the first implementation of the new generation of OGC protocols. It is described by the OGC API - Features - Part 1: Core document.

Here is a quick informal summary of the most important differences between the well known WFS protocol and WFS3:

- WFS3 is based on a REST API
- WFS3 API must follow the OPENAPI specifications
- WFS3 supports multiple output formats but it does not dictate any (only GeoJSON and HTML are currently available in QGIS WFS3) and it uses content negotiation to determine which format is to be served to the client
- JSON and HTML are first class citizens in WFS3
- WFS3 is self-documenting (through the `/api` endpoint)
- WFS3 is fully navigable (through links) and browsable

Wichtig: While the WFS3 implementation in QGIS can make use of the `MAP` parameter to specify the project file, no extra query parameters are allowed by the OPENAPI specification. For this reason it is strongly recommended that `MAP` is not exposed in the URL and the project file is specified in the environment by other means (i.e. setting `QGIS_PROJECT_FILE` in the environment through a web server rewrite rule).

Bemerkung: The **API** endpoint provides comprehensive documentation of all supported parameters and output formats of your service. The following paragraphs will only describe the most important ones.

Resource representation

The QGIS Server WFS3 implementation currently supports the following resource representation (output) formats:

- HTML
- JSON

The format that is actually served will depend on content negotiation, but a specific format can be explicitly requested by appending a format specifier to the endpoints.

Supported format specifier extensions are:

- `.json`
- `.html`

Additional format specifier aliases may be defined by specific endpoints:

- `.openapi`: alias for `.json` supported by the **API** endpoint
- `.geojson`: alias for `.json` supported by the **Features** and **Feature** endpoints

Endpoints

The API provides a list of endpoints that the clients can retrieve. The system is designed in such a way that every response provides a set of links to navigate through all the provided resources.

Endpoints points provided by the QGIS implementation are:

Name	Path	Beschreibung
Landing Page	`/`	General information about the service and provides links to all available endpoints
Conformance	`/conformance`	Information about the conformance of the service to the standards
API	`/api`	Full description of the endpoints provided by the service and the returned documents structure
Collections	`/collections`	List of all collections (i.e. ‚vector layers') provided by the service
Collection	`/collections/{collectionId}`	Information about a collection (name, metadata, extent etc.)
Funktionalitäten	`/collections/{collectionId}/items`	List of the features provided by the collection
Feature	`/collections/{collectionId}/items/{featureId}`	Information about a single feature

Landing Page

The main endpoint is the **Landing Page**. From that page it is possible to navigate to all the available service endpoints. The **Landing Page** must provide links to

- the API definition (path `/api` link relations `service-desc` and `service-doc`),
- the Conformance declaration (path `/conformance`, link relation `conformance`), and
- the Collections (path `/collections`, link relation `data`).

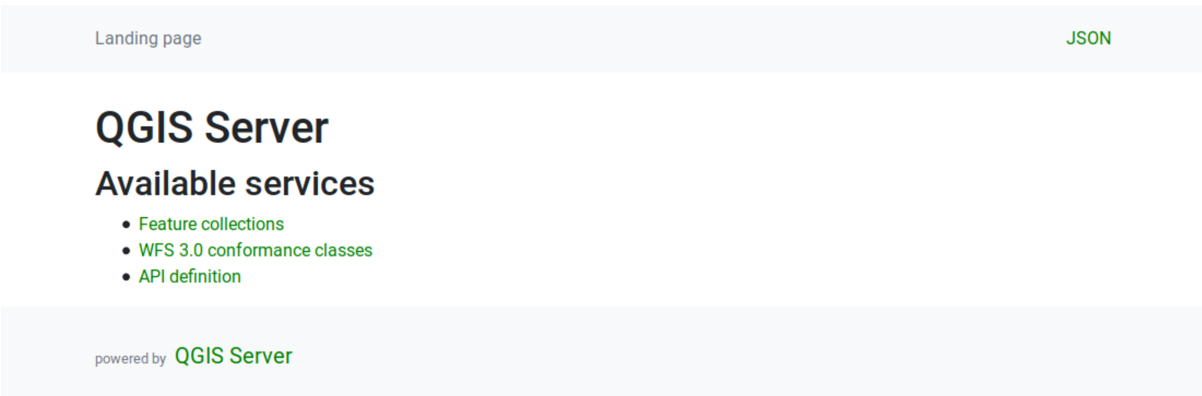


Abb. 18.9: Server WFS3 landing page

API Definition

The **API Definition** is an OPENAPI-compliant description of the API provided by the service. In its HTML representation it is a browsable page where all the endpoints and their response formats are accurately listed and documented. The path of this endpoint is `/api`.

The API definition provides a comprehensive and authoritative documentation of the service, including all supported parameters and returned formats.

Bemerkung: This endpoint is analogue to WFS's `GetCapabilities`

Collections list

The collections endpoint provides a list of all the collections available in the service. Since the service „serves" a single QGIS project the collections are the vector layers from the current project (if they were published as WFS in the project properties). The path of this endpoint is `/collections/`.

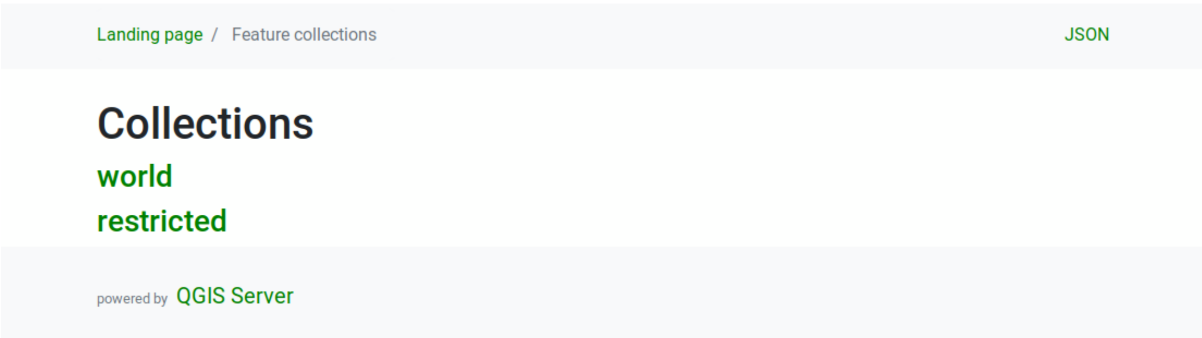


Abb. 18.10: Server WFS3 collections list page

Collection detail

While the collections endpoint does not provide detailed information about each available collection, that information is available in the `/collections/{collectionId}` endpoints. Typical information includes the extent, a description, CRSs and other metadata.

The HTML representation also provides a browsable map with the available features.

Features list

This endpoint provides a list of all features in a collection knowing the collection ID. The path of this endpoint is `/collections/{collectionId}/items`.

The HTML representation also provides a browsable map with the available features.

Bemerkung: This endpoint is analogue to `GetFeature` in WFS 1 and WFS 2.

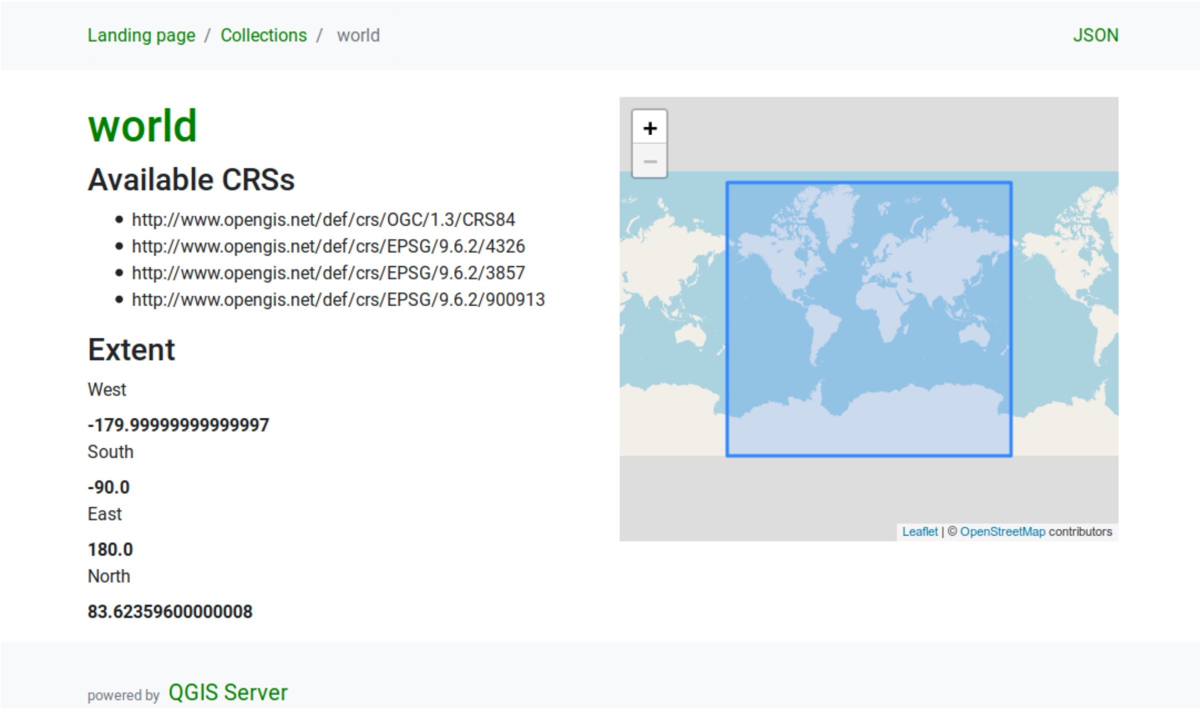


Abb. 18.11: Server WFS3 collection detail page

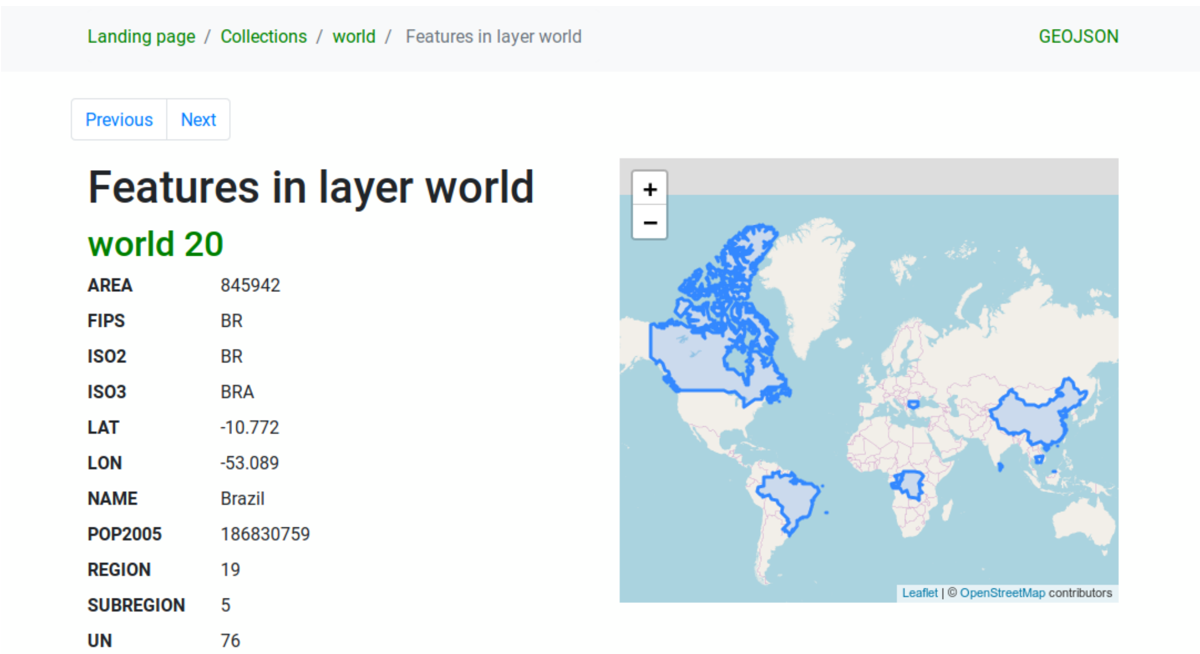


Abb. 18.12: Server WFS3 features list page

Feature detail

This endpoint provides all the available information about a single feature, including the feature attributes and its geometry. The path of this endpoint is `/collections/{collectionId}/items/{itemId}`.

The HTML representation also provides a browsable map with the feature geometry.

Abb. 18.13: Server WFS3 feature detail page

Pagination

Pagination of a long list of features is implemented in the OGC API through `next` and `prev` links, QGIS server constructs these links by appending `limit` and `offset` as query string parameters.

URL example:

```
http://localhost/qgisserver/wfs3/collection_one/items.json?offset=10&limit=10
```

Bemerkung: The maximum acceptable value for `limit` can be configured with the `QGIS_SERVER_API_WFS3_MAX_LIMIT` server configuration setting (see: *Environment variables*).

Objektfiltrierung

The features available in a collection can be filtered/searched by specifying one or more filters.

Date and time filter

Collections with date and/or datetime attributes can be filtered by specifying a `datetime` argument in the query string. By default the first date/datetime field is used for filtering. This behavior can be configured by setting a „Date“ or „Time“ dimension in the *QGIS Server ▯ Dimension* section of the layer properties dialog.

The date and time filtering syntax is fully described in the *API Definition* and also supports ranges (begin and end values are included) in addition to single values.

URL examples:

Returns only the features with date dimension matching `2019-01-01`

```
http://localhost/qgisserver/wfs3/collection_one/items.json?datetime=2019-01-01
```

Returns only the features with datetime dimension matching `2019-01-01T01:01:01`

```
http://localhost/qgisserver/wfs3/collection_one/items.json?datetime=2019-01-
↪01T01:01:01
```

Returns only the features with datetime dimension in the range `2019-01-01T01:01:01 - 2019-01-01T12:00:00`

```
http://localhost/qgisserver/wfs3/collection_one/items.json?datetime=2019-01-
↪01T01:01:01/2019-01-01T12:00:00
```

Bounding box filter

A bounding box spatial filter can be specified with the `bbox` parameter:

The order of the comma separated elements is:

- Lower left corner, WGS 84 longitude
- Lower left corner, WGS 84 latitude
- Upper right corner, WGS 84 longitude
- Upper right corner, WGS 84 latitude

Bemerkung: The OGC specifications also allow a 6 item bbox specifier where the third and sixth items are the Z components, this is not yet supported by QGIS server.

URL example:

```
http://localhost/qgisserver/wfs3/collection_one/items.json?bbox=-180,-90,180,90
```

If the *CRS* of the bounding box is not WGS84 (*http://www.opengis.net/def/crs/OGC/1.3/CRS84*), a different CRS can be specified by using the optional parameter `bbox-crs`. The CRS format identifier must be in the OGC URI format:

URL example:

```
http://localhost/qgisserver/wfs3/collection_one/items.json?bbox=913191,5606014,
↪913234,5606029&bbox-crs=http://www.opengis.net/def/crs/EPSG/9.6.2/3857
```

Attribute filters

Attribute filters can be combined with the bounding box filter and they are in the general form: `<attribute name>=<attribute value>`. Multiple filters can be combined using the `AND` operator.

URL example:

filters all features where attribute `name` equals „my value"

```
http://localhost/qgisserver/wfs3/collection_one/items.json?attribute_one=my%20value
```

Partial matches are also supported by using a `*` („star") operator:

URL example:

filters all features where attribute `name` ends with „value"

```
http://localhost/qgisserver/wfs3/collection_one/items.json?attribute_one=*value
```

Attribute selection

The feature attributes returned by a *Features list* call can be limited by adding a comma separated list of attribute names in the optional `properties` query string argument.

URL example:

returns only the `name` attribute

```
http://localhost/qgisserver/wfs3/collection_one/items.json?properties=name
```

The HTML template language

The HTML representation uses a set of HTML templates to generate the response. The template is parsed by a template engine called inja. The templates can be customized by overriding them (see: *Template overrides*). The template has access to the same data that are available to the `JSON` representation and a few additional functions are available to the template:

Custom template functions

- `path_append( path )`: appends a directory path to the current url
- `path_chomp( n )`: removes the specified number „n" of directory components from the current url path
- `json_dump( )`: prints the JSON data passed to the template
- `static( path )`: returns the full URL to the specified static path. For example: „static(„/style/black.css")" with a root path „http://localhost/qgisserver/wfs3" will return „http://localhost/qgisserver/wfs3/static/style/black.css".
- `links_filter( links, key, value )`: Returns filtered links from a link list
- `content_type_name( content_type )`: Returns a short name from a content type, for example „text/html" will return „HTML"

Template overrides

Templates and static assets are stored in subdirectories of the QGIS server default API resource directory (`/usr/share/qgis/resources/server/api/` on a Linux system), the base directory can be customized by changing the environment variable `QGIS_SERVER_API_RESOURCES_DIRECTORY`.

A typical Linux installation will have the following directory tree:

```
/usr/share/qgis/resources/server/api/
└── ogc
    ├── schema.json
    ├── static
    │   ├── jsonFormatter.min.css
    │   ├── jsonFormatter.min.js
    │   └── style.css
    └── templates
        └── wfs3
            ├── describeCollection.html
            ├── describeCollections.html
            ├── footer.html
            ├── getApiDescription.html
            ├── getFeature.html
            ├── getFeatures.html
            ├── getLandingPage.html
            ├── getRequirementClasses.html
            ├── header.html
            ├── leaflet_map.html
            └── links.html
```

To override the templates you can copy the whole tree to another location and point `QGIS_SERVER_API_RESOURCES_DIRECTORY` to the new location.

Extra parameters supported by all request types

The following extra parameters are supported by all protocols.

- **FILE_NAME**: if set, the server response will be sent to the client as a file attachment with the specified file name.

Bemerkung: Not available for WFS3.

- **MAP**: Similar to MapServer, the `MAP` parameter can be used to specify the path to the QGIS project file. You can specify an absolute path or a path relative to the location of the server executable (`qgis_mapserv.fcgi`). If not specified, QGIS Server searches for .qgs files in the directory where the server executable is located.

  Example:

  ```
  http://localhost/cgi-bin/qgis_mapserv.fcgi?\
    REQUEST=GetMap&MAP=/home/qgis/projects/world.qgs&...
  ```

Bemerkung: You can define a **QGIS_PROJECT_FILE** as an environment variable to tell the server executable where to find the QGIS project file. This variable will be the location where QGIS will look for the project file. If not defined it will use the MAP parameter in the request and finally look at the server executable directory.

REDLINING

This feature is available and can be used with `GetMap` and `GetPrint` requests.

The redlining feature can be used to pass geometries and labels in the request which are overlapped by the server over the standard returned image (map). This permits the user to put emphasis or maybe add some comments (labels) to some areas, locations etc. that are not in the standard map.

The request is in the format:

```
http://qgisplatform.demo/cgi-bin/qgis_mapserv.fcgi?map=/world.qgs&SERVICE=WMS&
↪VERSION=1.3.0&
REQUEST=GetMap
...
&HIGHLIGHT_GEOM=POLYGON((590000 5647000, 590000 6110620, 2500000 6110620, 2500000␣
↪5647000, 590000 5647000))
&HIGHLIGHT_SYMBOL=<StyledLayerDescriptor><UserStyle><Name>Highlight</Name>
↪<FeatureTypeStyle><Rule><Name>Symbol</Name><LineSymbolizer><Stroke><SvgParameter␣
↪name="stroke">%23ea1173</SvgParameter><SvgParameter name="stroke-opacity">1</
↪SvgParameter><SvgParameter name="stroke-width">1.6</SvgParameter></Stroke></
↪LineSymbolizer></Rule></FeatureTypeStyle></UserStyle></StyledLayerDescriptor>
&HIGHLIGHT_LABELSTRING=Write label here
&HIGHLIGHT_LABELSIZE=16
&HIGHLIGHT_LABELCOLOR=%23000000
&HIGHLIGHT_LABELBUFFERCOLOR=%23FFFFFF
&HIGHLIGHT_LABELBUFFERSIZE=1.5
```

Here is the image outputed by the above request in which a polygon and a label are drawn on top of the normal map:

You can see there are several parameters in this request:

- **HIGHLIGHT_GEOM**: You can add POINT, MULTILINESTRING, POLYGON etc. It supports multipart geometries. Here is an example: `HIGHLIGHT_GEOM=MULTILINESTRING((0 0, 0 1, 1 1))`. The coordinates should be in the CRS of the GetMap/GetPrint request.
- **HIGHLIGHT_SYMBOL**: This controls how the geometry is outlined and you can change the stroke width, color and opacity.
- **HIGHLIGHT_LABELSTRING**: You can pass your labeling text to this parameter.
- **HIGHLIGHT_LABELSIZE**: This parameter controls the size of the label.
- **HIGHLIGHT_LABELCOLOR**: This parameter controls the label color.
- **HIGHLIGHT_LABELBUFFERCOLOR**: This parameter controls the label buffer color.
- **HIGHLIGHT_LABELBUFFERSIZE**: This parameter controls the label buffer size.

External WMS layers

QGIS Server allows including layers from external WMS servers in WMS GetMap and WMS GetPrint requests. This is especially useful if a web client uses an external background layer in the web map. For performance reasons, such layers should be directly requested by the web client (not cascaded via QGIS server). For printing however, these layers should be cascaded via QGIS server in order to appear in the printed map.

External layers can be added to the LAYERS parameter as EXTERNAL_WMS:<layername>. The parameters for the external WMS layers (e.g. url, format, dpiMode, crs, layers, styles) can later be given as service parameters <layername>:<parameter>. In a GetMap request, this might look like this:

```
http://localhost/qgisserver?
SERVICE=WMS&REQUEST=GetMap
...
&LAYERS=EXTERNAL_WMS:basemap,layer1,layer2
&STYLES=,,
```

(Fortsetzung auf der nächsten Seite)

Abb. 18.14: Server response to a GetMap request with redlining parameters

(Fortsetzung der vorherigen Seite)

```
&basemap:url=http://externalserver.com/wms.fcgi
&basemap:format=image/jpeg
&basemap:dpiMode=7
&basemap:crs=EPSG:2056
&basemap:layers=orthofoto
&basemap:styles=default
```

Similarly, external layers can be used in GetPrint requests:

```
http://localhost/qgisserver?
SERVICE=WMS
...
&REQUEST=GetPrint&TEMPLATE=A4
&map0:layers=EXTERNAL_WMS:basemap,layer1,layer2
&map0:EXTENT=<minx,miny,maxx,maxy>
&basemap:url=http://externalserver.com/wms.fcgi
&basemap:format=image/jpeg
&basemap:dpiMode=7
&basemap:crs=EPSG:2056
&basemap:layers=orthofoto
&basemap:styles=default
```

18.2.3 Plugins

Installation

To install the HelloWorld example plugin for testing the servers, you firstly have to create a directory to hold server plugins. This will be specified in the virtual host configuration and passed on to the server through an environment variable:

```
mkdir -p /var/www/qgis-server/plugins
cd /var/www/qgis-server/plugins
wget https://github.com/elpaso/qgis-helloserver/archive/master.zip
unzip master.zip
mv qgis-helloserver-master HelloServer
```

HTTP Server configuration

Apache

To be able to use server plugins, FastCGI needs to know where to look. So, we have to modify the Apache configuration file to indicate the **QGIS_PLUGINPATH** environment variable to FastCGI:

```
FcgidInitialEnv QGIS_PLUGINPATH "/var/www/qgis-server/plugins"
```

Moreover, a basic HTTP authorization is necessary to play with the HelloWorld plugin previously introduced. So we have to update the Apache configuration file a last time:

```
# Needed for QGIS HelloServer plugin HTTP BASIC auth
<IfModule mod_fcgid.c>
    RewriteEngine on
    RewriteCond %{HTTP:Authorization} .
    RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
</IfModule>
```

Then, restart Apache:

```
systemctl restart apache2
```

How to use a plugin

Test the server with the HelloWorld plugin:

```
wget -q -O - "http://localhost/cgi-bin/qgis_mapserv.fcgi?SERVICE=HELLO"
 HelloServer!
```

You can have a look at the default GetCapabilities of the QGIS server at:

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?SERVICE=WMS&VERSION=1.3.0&
↪REQUEST=GetCapabilities
```

18.2.4 Advanced configuration

Logging

To log requests sent to the server, you have to set the following environment variables:

- **QGIS_SERVER_LOG_LEVEL**
- **QGIS_SERVER_LOG_FILE**
- **QGIS_SERVER_LOG_STDERR**

Take a look on *Environment variables* to understand their meanings.

Environment variables

You can configure some aspects of QGIS Server by setting **environment variables**.

According to the HTTP server and how you run QGIS Server, there are several ways to define these variables. This is fully described in *HTTP Server configuration*.

Name	Beschreibung	Default	Services
QGIS_OPTIONS_PATH	Specifies the path to the directory with settings. It works the same way as QGIS application `--optionspath` option. It is looking for settings file in `<QGIS_OPTIONS_PATH>/QGIS/QGIS3.ini`.	,‘	All

QUERY_STRING	The query string, normally passed by the web server. This variable can be useful while testing QGIS server binary from the command line.
	For example for testing a GetCapabilities request on the command line to a project that also requires a PostgreSQL connection defined in a pg_service.conf file:
	`PGSERVICEFILE=/etc/pg_service.conf \`
	`QUERY_STRING="MAP=/home/qgis/projects/world.qgs&SERVICE=WMS&REQUEST=GetCapabilities" \`
	`/usr/lib/cgi-bin/qgis_mapserv.fcgi`
	The result should be either the content of the GetCapabilities response or, if something is wrong, an error message.
		,‘	All

QGIS_PROJECT_FILE	The `.qgs` or `.qgz` project file, normally passed as a parameter in the query string (with *MAP*), you can also set it as an environment variable (for example by using `mod_rewrite` Apache module). Note that you may also indicate a project stored in PostgreSQL, e.g. `postgresql://localhost:5432?sslmode=disable&dbname=mydb&schema=myschema&project=myproject`.	,‘	All
QGIS_SERVER_LOG_FILE	Specify path and filename. Make sure that server has proper permissions for writing to file. File should be created automatically, just send some requests to server. If it's not there, check permissions. QGIS_SERVER_LOG_FILE is deprecated since QGIS 3.4. File logging support will be removed in QGIS 4.0.	,‘	All
QGIS_SERVER_LOG_STDERR	Activate logging to stderr. This variable has no effect when `QGIS_SERVER_LOG_FILE` is set. • `0` or `false` (case insensitive) • `1` or `true` (case insensitive)	false	All
MAX_CACHE_LAYERS	Specify the maximum number of cached layers (default: `100`).	100	All
QGIS_PLUGINPATH	Useful if you are using Python plugins for the server, this sets the folder that is searched for Python plugins.	,‘	All
QGIS_SERVER_LOG_LEVEL	Specify desired log level. Available values are: • `0` or `INFO` (log all requests) • `1` or `WARNING` • `2` or `CRITICAL` (log just critical errors, suitable for production purposes)	0	All
QGIS_SERVER_PARALLEL_RENDERING	Activates parallel rendering for WMS Get-	false	WMS

Settings summary

When QGIS Server is starting, you have a summary of all configurable parameters thanks to environment variables. Moreover, the value currently used and the origin is also displayed.

For example with spawn-fcgi:

```
export QGIS_OPTIONS_PATH=/home/user/.local/share/QGIS/QGIS3/profiles/default/
export QGIS_SERVER_LOG_FILE=/home/user/qserv.log
export QGIS_SERVER_LOG_LEVEL=2
spawn-fcgi -f /usr/lib/cgi-bin/qgis_mapserv.fcgi -s /tmp/qgisserver.sock -U www-
↪data -G www-data -n

 QGIS Server Settings:

   - QGIS_OPTIONS_PATH / '' (Override the default path for user configuration): '/
↪home/user/.local/share/QGIS/QGIS3/profiles/default/' (read from ENVIRONMENT_
↪VARIABLE)

   - QGIS_SERVER_PARALLEL_RENDERING / '/qgis/parallel_rendering' (Activate/
↪Deactivate parallel rendering for WMS getMap request): 'true' (read from INI_
↪FILE)

   - QGIS_SERVER_MAX_THREADS / '/qgis/max_threads' (Number of threads to use when␣
↪parallel rendering is activated): '4' (read from INI_FILE)

   - QGIS_SERVER_LOG_LEVEL / '' (Log level): '2' (read from ENVIRONMENT_VARIABLE)

   - QGIS_SERVER_LOG_FILE / '' (Log file): '/tmp/qserv.log' (read from ENVIRONMENT_
↪VARIABLE)

   - QGIS_PROJECT_FILE / '' (QGIS project file): '' (read from DEFAULT_VALUE)

   - MAX_CACHE_LAYERS / '' (Specify the maximum number of cached layers): '100'␣
↪(read from DEFAULT_VALUE)

   - QGIS_SERVER_CACHE_DIRECTORY / '/cache/directory' (Specify the cache␣
↪directory): '/root/.local/share/QGIS/QGIS3/profiles/default/cache' (read from␣
↪DEFAULT_VALUE)

   - QGIS_SERVER_CACHE_SIZE / '/cache/size' (Specify the cache size): '52428800'␣
↪(read from INI_FILE)

 Ini file used to initialize settings: /home/user/.local/share/QGIS/QGIS3/profiles/
↪default/QGIS/QGIS3.ini
```

In this particular case, we know that **QGIS_SERVER_MAX_THREADS** and **QGIS_SERVER_PARALLEL_RENDERING** values are read from the ini file found in **QGIS_OPTIONS_PATH** directory (which is defined through an environment variable). The corresponding entries in the ini file are **/qgis/max_threads** and **/qgis/parallel_rendering** and their values are **true** and **4** threads.

Short name for layers, groups and project

A number of elements have both a `<Name>` and a `<Title>`. The **Name** is a text string used for machine-to-machine communication while the **Title** is for the benefit of humans.

For example, a dataset might have the descriptive Title "Maximum Atmospheric Temperature" and be requested using the abbreviated **Name** "ATMAX". User can already set title for layers, groups and project.

OWS name is based on the name used in layer tree. This name is more a label for humans than a name for machine-to-machine communication.

QGIS Server supports:

- short name line edits to layers properties You can change this by right clicking on a layer, choose *Properties ► Metadata tab ► Description ► Short name*.
- WMS data dialog to layer tree group (short name, title, abstract)

  By right clicking on a layer group and selecting the *Set Group WMS data* option you will get:

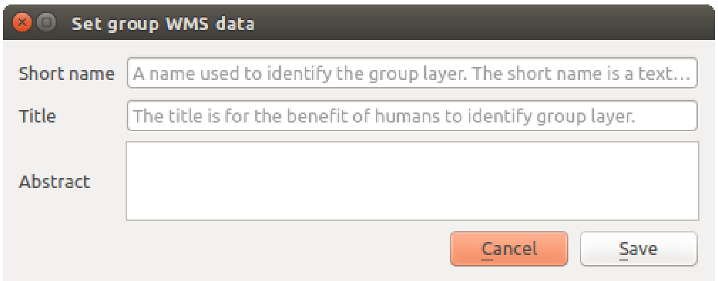


Abb. 18.15: Set group WMS data dialog

- short name line edits to project properties - add a regexp validator `"^[A-Za-z][A-Za-z0-9\._-]*"` to short name line edit accessible through a static method
- add a regexp validator `"^[A-Za-z][A-Za-z0-9\._-]*"` to short name line edit accessible through a static method

  You can choose a short name for the project root by going to *Project properties ► OWS Server ► Service capabilities ► Short name*.
- add a `TreeName` element in the `fullProjectSettings`

If a short name has been set for layers, groups or project it is used by QGIS Sever as the layer name.

Connection to service file

In order to make apache aware of the PostgreSQL service file (see the *PostgreSQL Service connection file* section) you need to make your `*.conf` file look like:

```
SetEnv PGSERVICEFILE /home/web/.pg_service.conf

<Directory "/home/web/apps2/bin/">
  AllowOverride None
.....
```

Add fonts to your linux server

Keep in mind that you may use QGIS projects that point to fonts that may not exist by default on other machines. This means that if you share the project, it may look different on other machines (if the fonts don't exist on the target machine).

In order to ensure this does not happen you just need to install the missing fonts on the target machine. Doing this on desktop systems is usually trivial (double clicking the fonts).

For linux, if you don't have a desktop environment installed (or you prefer the command line) you need to:

- On Debian based systems:

```
sudo su
mkdir -p /usr/local/share/fonts/truetype/myfonts && cd /usr/local/share/fonts/
↪truetype/myfonts

# copy the fonts from their location
cp /fonts_location/* .

chown root *
cd .. && fc-cache -f -v
```

- On Fedora based systems:

```
sudo su
mkdir /usr/share/fonts/myfonts && cd /usr/share/fonts/myfonts

# copy the fonts from their location
cp /fonts_location/* .

chown root *
cd .. && fc-cache -f -v
```

18.2.5 Containerized deployment

There are many ways to use containerized application, from the most simple (simple Docker images) to sophisticated (Kubernetes and so on).

Bemerkung: This kind of deployment needs the docker application to be installed and running. Check this tutorial.

Hinweis: Docker run pre packaged application (aka images) which can be retrieved as sources (Dockerfile and resources) to build or already built from registries (private or public).

Simple docker images

As the docker image does not exist in a public registry. you will need to build it. To do so create a directory `qgis-server` and within its directory:

- create a file `Dockerfile` with this content:

```
FROM debian:buster-slim

ENV LANG=en_EN.UTF-8
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
RUN apt-get update \
    && apt-get install --no-install-recommends --no-install-suggests --allow-
↪unauthenticated -y \
        gnupg \
        ca-certificates \
        wget \
        locales \
    && localedef -i en_US -f UTF-8 en_US.UTF-8 \
    && wget -O - https://qgis.org/downloads/qgis-2019.gpg.key | gpg --import \
    && gpg --export --armor 8D5A5B203548E5004487DD1951F523511C7028C3 | apt-key add␣
↪- \
    && echo "deb http://qgis.org/debian buster main" >> /etc/apt/sources.list.d/
↪qgis.list \
    && apt-get update \
    && apt-get install --no-install-recommends --no-install-suggests --allow-
↪unauthenticated -y \
        qgis-server \
        spawn-fcgi \
        xauth \
        xvfb \
    && apt-get remove --purge -y \
        gnupg \
        wget \
    && rm -rf /var/lib/apt/lists/*

RUN useradd -m qgis

ENV TINI_VERSION v0.17.0
ADD https://github.com/krallin/tini/releases/download/${TINI_VERSION}/tini /tini
RUN chmod +x /tini

ENV QGIS_PREFIX_PATH /usr
ENV QGIS_SERVER_LOG_STDERR 1
ENV QGIS_SERVER_LOG_LEVEL 2

COPY cmd.sh /home/qgis/cmd.sh
RUN chown qgis:qgis /home/qgis/cmd.sh

USER qgis
WORKDIR /home/qgis

ENTRYPOINT ["/tini", "--"]

CMD ["/home/qgis/cmd.sh"]
```

- create a file `cmd.sh` with this content:

```
#!/bin/bash

[[ $DEBUG == "1" ]] && env

exec /usr/bin/xvfb-run --auto-servernum --server-num=1 /usr/bin/spawn-fcgi -p 5555␣
↪-n -d /home/qgis -- /usr/lib/cgi-bin/qgis_mapserv.fcgi
```

- build the image with:

```
docker build -f Dockerfile -t qgis-server ./
```

First try

To run the server you will need a QGis project file. You can use one of yours or pick this sample.

To do so, create a directory `data` within the directory `qgis-server` and copy your file in it. To comply with the following explanations, rename it to `osm.qgs`.

Now, you can run the server with:

```
docker network create qgis
docker run -d --rm --name qgis-server --net=qgis --hostname=qgis-server \
              -v $(pwd)/data:/data:ro -p 5555:5555 \
              -e "QGIS_PROJECT_FILE=/data/osm.qgs" \
              qgis-server
```

Options used:

- **-d**: run in the background
- **–rm**: remove the container when it is stopped
- **–name**: name of the container to be created
- **–net**: (previously created) sub network
- **–hostname**: container hostname, for later referencing
- **-v**: local data directory to be mounted in the container
- **-p**: host/container port mapping
- **-e**: environment variable to be used in the container

To check, type `docker ps | grep qgis-server` and you should see a line with **qgis-server**:

```
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS          ␣
↪   PORTS                    NAMES
4de8192da76e   qgis-server    "/tini -- /home/qgis…"   3 seconds ago   Up 2 seconds␣
↪   0.0.0.0:5555->5555/tcp   qgis-server
```

Usable sample

As the server is only accepting fastcgi connections, you have to have an HTTP server that handles this protocol. To do so we have to create a simple Nginx configuration file and start a Nginx image.

Create a file `nginx.conf` in the current directory with this content:

```
server {
  listen 80;
  server_name _;
  location / {
    root  /usr/share/nginx/html;
    index index.html index.htm;
  }
  location /qgis-server {
    proxy_buffers 16 16k;
    proxy_buffer_size 16k;
    gzip off;
    include fastcgi_params;
    fastcgi_pass qgis-server:5555;
  }
}
```

And type this command:

```
docker run -d --rm --name nginx --net=qgis --hostname=nginx \
              -v $(pwd)/nginx.conf:/etc/nginx/conf.d/default.conf:ro -p 8080:80 \
              nginx:1.13
```

To check capabilities availability, type in a browser http://localhost:8080/qgis-server/?SERVICE=WMS&VERSION=1.3.0&REQUEST=GetCapabilities

Cleanup

To cleanup the running images, type:

```
docker stop qgis-server nginx
```

Docker stacks

The previous method is scriptable, but not easily packageable nor standardized or easily manageable.

To work with a docker image set you could use a docker stack managed by an orchestrator. In a stack, the images are working in the same private network, and you can start / stop the whole stack or deploy the stack to other workers. There are many orchestrators, for example Swarm, Kubernetes and Mesos.

In the following, we will present simple configurations for testing purposes. They are not suitable for production.

Swarm/docker-compose

Docker now has its own orchestrator: Swarm (compatible with docker-compose files). You have to enable it (the Mac version will also work with Linux).

Stack description

Now that you have Swarm working, create the service file (see deploy swarm) `qgis-stack.yaml`:

```yaml
version: '3.7'

services:
  qgis-server:
    # Should use version with utf-8 locale support:
    image: qgis-server:latest
    volumes:
    - REPLACE_WITH_FULL_PATH/data:/data:ro
    environment:
    - LANG=en_EN.UTF-8
    - QGIS_PROJECT_FILE=/data/osm.qgs
    - QGIS_SERVER_LOG_LEVEL=0  # INFO (log all requests)
    - DEBUG=1                  # display env before spawning QGIS Server

  nginx:
    image: nginx:1.13
    ports:
    - 8080:80
    volumes:
    - REPLACE_WITH_FULL_PATH/nginx.conf:/etc/nginx/conf.d/default.conf:ro
    depends_on:
    - qgis-server
```

To deploy (or update) the stack, type:

```
docker stack deploy -c qgis-stack.yaml qgis-stack
```

Check the stack deployment status until you obtain **1/1** in the **replicas** column:

```
docker stack services qgis-stack
```

Something like:

```
ID                NAME                 MODE          REPLICAS      IMAGE                  ␣
↪   PORTS
gmx7ewlvwsqt      qgis_nginx           replicated    1/1           nginx:1.13             ␣
↪   *:8080->80/tcp
l0v2e7cl43u3      qgis_qgis-server     replicated    1/1           qgis-server:latest
```

To check WMS capabilities, type in a web browser http://localhost:8080/qgis-server/?SERVICE=WMS&VERSION=1.3.0&REQUEST=GetCapabilities

Cleanup

To cleanup, type:

```
docker stack rm qgis-stack
```

Kubernetes

Installation

If you have a **Docker Desktop** installation, using Kubernetes (aka k8s) is pretty straight forward: enable k8s.

If not, follow the minikube tutorial or microk8s for Ubuntu.

As Kubernetes installation can be really complex, we will only focus on aspects used by this demo. For further / deeper information, check the official documentation.

microk8s

microk8s needs extra steps: you have to enable the registry and tag the qgis-server image in order to have Kubernetes to find the created images.

First, enable the registry:

```
microk8s enable dashboard dns registry
```

Then, tag and push the image to your newly created registry:

```
docker tag qgis-server 127.0.0.1:32000/qgis-server && docker push 127.0.0.1:32000/
↪qgis-server
```

Finally, add or complete the `/etc/docker/daemon.json` to have your registry **127.0.0.1:32000** listed in the **insecure-registries** field:

```
{
  "insecure-registries": ["127.0.0.1:32000"]
}
```

Creating manifests

Kubernetes describes the objects to deploy in yaml manifests. There are many different kinds, but we will only use deployments (handle pods, i.e. docker images) and services to expose the deployments to internal or external purposes.

Deployment manifests

Create a file `deployments.yaml` with this content:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: qgis-server
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      myLabel: qgis-server
  template:
    metadata:
      labels:
        myLabel: qgis-server
    spec:
      containers:
        - name: qgis-server
          image: localhost:32000/qgis-server:latest
          imagePullPolicy: IfNotPresent
          env:
            - name: LANG
              value: en_EN.UTF-8
            - name: QGIS_PROJECT_FILE
              value: /data/osm.qgs
            - name: QGIS_SERVER_LOG_LEVEL
              value: "0"
            - name: DEBUG
              value: "1"
          ports:
            - containerPort: 5555
          volumeMounts:
            - name: qgis-data
              mountPath: /data/
      volumes:
        - name: qgis-data
          hostPath:
            path: REPLACE_WITH_FULL_PATH/data

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: qgis-nginx
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      myLabel: qgis-nginx
  template:
    metadata:
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
      labels:
        myLabel: qgis-nginx
    spec:
      containers:
        - name: qgis-nginx
          image: nginx:1.13
          ports:
            - containerPort: 80
          volumeMounts:
            - name: nginx-conf
              mountPath: /etc/nginx/conf.d/default.conf
      volumes:
        - name: nginx-conf
          hostPath:
            path: REPLACE_WITH_FULL_PATH/nginx.conf
```

Service manifests

Create a file `services.yaml` with this content:

```
apiVersion: v1
kind: Service
metadata:
  name: qgis-server
  namespace: default
spec:
  type: ClusterIP
  selector:
    myLabel: qgis-server
  ports:
    - port: 5555
      targetPort: 5555
---
apiVersion: v1
kind: Service
metadata:
  name: qgis-nginx
  namespace: default
spec:
  type: NodePort
  selector:
    myLabel: qgis-nginx
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30080
```

Deploying manifests

To deploy the images and services in Kubernetes, one can use the dashboard (click on the **+** on the upper right) or the command line.

Bemerkung: When using the command line with microk8s you will have to prefix each command with *microk8s*.

To deploy or update your manifests:

```
kubectl apply -k ./
```

To check what is currently deployed:

```
kubectl get pods,services,deployment
```

You should obtain something like:

```
NAME                                  READY   STATUS    RESTARTS   AGE
pod/qgis-nginx-54845ff6f6-8skp9      1/1     Running   0          27m
pod/qgis-server-75df8ddd89-c7t7s     1/1     Running   0          27m

NAME                          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)       ␣
↪   AGE
service/Kubernetes            ClusterIP   10.152.183.1     <none>        443/TCP       ␣
↪   5h51m
service/qgis-exec-server      ClusterIP   10.152.183.218   <none>        5555/TCP      ␣
↪   35m
service/qgis-nginx            NodePort    10.152.183.234   <none>        80:30080/TCP␣
↪   27m
service/qgis-server           ClusterIP   10.152.183.132   <none>        5555/TCP      ␣
↪   27m

NAME                              READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/qgis-nginx       1/1     1            1           27m
deployment.apps/qgis-server      1/1     1            1           27m
```

To read nginx/qgis logs, type:

```
kubectl logs -f POD_NAME
```

To check WMS capabilities, type in a web browser http://localhost:30080/qgis-server/?SERVICE=WMS&VERSION=1.3.0&REQUEST=GetCapabilities

Cleanup

To clean up, type:

```
kubectl delete -n default service/qgis-server service/qgis-nginx deployment/qgis-
↪nginx deployment/qgis-server
```

Cloud deployment

Managing your own cluster of servers to handle the deployment of containerized applications, is a complex job. You have to handle multiple issues, such as hardware, bandwidth and security at different levels.

Cloud deployment solutions can be a good alternative when you do not want to focus on infrastructure management.

A cloud deployment may use proprietary mechanisms, but they are also compatible with the stages explained previously (*docker images* and *stack management*).

AWS usecase

With Amazon AWS, through ECS (Elastic Container Service) functionalities, you can use docker-compose or Kubernetes compatible wrappers to manage your stack. You will have to create an image registry for your custom images to be accessible.

To use docker-compose alike functionalities, you need to install the **ecs-cli** client and have proper permissions / roles. Then, with the help of the *ecs-cli compose* commands (see the ecs-cli compose manual and ecs-cli tutorial), you can reuse the *stack description*.

To use Kubernetes, you can use the AWS web console or the command line tool eksctl and have the proper permissions / roles. Then with a well configured kubectl environment, you can reuse the *Kubernetes manifests*.

KAPITEL 19

Arbeiten mit GPS Daten

19.1 GPS Plugin

19.1.1 Was ist GPS?

GPS, das Global Positioning System, ist ein satellitenbasiertes System das es jedem mit einem GPS Empfänger ermöglicht seine genaue Position überall auf der Welt zu finden. GPS wird als Navigationshilfe in Flugzeugen, in Schiffen und von Wanderern benutzt. Der GPS Empfänger verwendet die Signale von den Satelliten um seine Länge, Breite und (manchmal) Höhe zu berechnen. Die meisten Empfänger haben auch die Fähigkeit Orte (bekannt als **Wegpunkte**), Sequenzen von Wegpunkten, die zusammen eine geplante **Route** ergeben und Tracklogs oder **Spuren** der zeitlichen Bewegung des Empfängers zu speichern. Wegpunkte, Routen und Spuren sind die drei Grundtypen in GPS Daten. QGIS stellt Wegpunkte in Punktlayern dar, wohingegen Routen und Spuren in Vektorlinien dargestellt werden.

Bemerkung: QGIS supports also GNSS receivers. But we keep using the term GPS in this documentation.

19.1.2 GPS-Daten aus einer Datei laden

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.

Um eine GPX Datei zu laden müssen Sie zuerst das Plugin laden. *Erweiterungen ⍰ |showPluginManager| :menu-selection:`Erweiterungen verwalten und installieren …* öffnet den Pluginmanagerdialog. Aktivieren Sie das *GPS Werkzeuge* Kontrollkästchen. Wenn dieses Plugin geladen ist erscheint ein Knopf mit einem kleinen tragbaren GPS Gerät in der Werkzeugleiste und in:menuselection:*Layer –> Layer erstellen –>* :

- GPS-Werkzeuge
- *Erstelle neuen GPX Layer*

Für das Arbeiten mit GPS-Daten stellen wir eine GPX-Beispieldatei im **|qg|** Beispieldatensatz zur Verfügung: `qgis_sample_data/gps/national_monuments.gpx`. Siehe Abschnitt *Beispieldaten herunterladen* für weitere Informationen über die Beispieldaten.

1. Select *Vector* ▯ *GPS Tools* or click the GPS Tools icon in the toolbar and open the *Load GPX file* tab (see *figure_GPS*).
2. Browse to the folder `qgis_sample_data/gps/`, select the GPX file `national_monuments.gpx` and click *Open*.

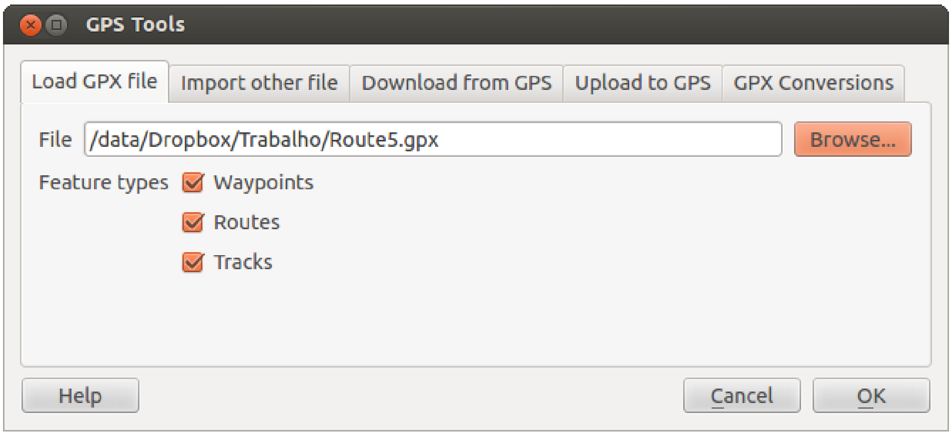


Abb. 19.1: Das *GPS Werkzeuge* Dialogfenster

Use the *Browse…* button to select the GPX file, then use the checkboxes to select the feature types you want to load from that GPX file. Each feature type will be loaded in a separate layer when you click *OK*. The file `national_monuments.gpx` only includes waypoints.

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 https://www.topografix.com/GPX/1/1/.

19.1.3 GPSBabel

Since QGIS uses GPX files, you need a way to convert other GPS file formats to GPX. This can be done for many formats using the free program GPSBabel, which is available at https://www.gpsbabel.org. This program can also transfer GPS data between your computer and a GPS device. QGIS uses GPSBabel to do these things, so it is recommended that you install it. However, if you just want to load GPS data from GPX files you will not need it. Version 1.2.3 of GPSBabel is known to work with QGIS, but you should be able to use later versions without any problems.

19.1.4 GPS-Daten importieren

Um GPS-Daten aus einer Datei, die nicht im GPX-Format vorliegt zu importieren, benutzen Sie den Reiter *Aus anderer Datei importieren*. Wählen Sie dann die Datei (und den Dateityp), die importiert werden soll aus, von welchem Datenformat Sie importieren möchten und wo die konvertierte GPX-Datei unter welchem Namen abgelegt werden soll. Beachten Sie, dass nicht für alle Datenformate die drei GPS-Datentypen Wegpunkte, Routen und Spuren unterstützt werden. Manchmal sind es nur ein oder zwei.

19.1.5 GPS-Daten von einem Empfänger herunterladen

QGIS can use GPSBabel to download data from a GPS device directly as new vector layers. For this we use the *Download from GPS* tab of the GPS Tools dialog (see *Figure_GPS_download*). Here, we select the type of GPS device, the port that it is connected to (or USB if your GPS supports this), the feature type that you want to download, the GPX file where the data should be stored, and the name of the new layer.

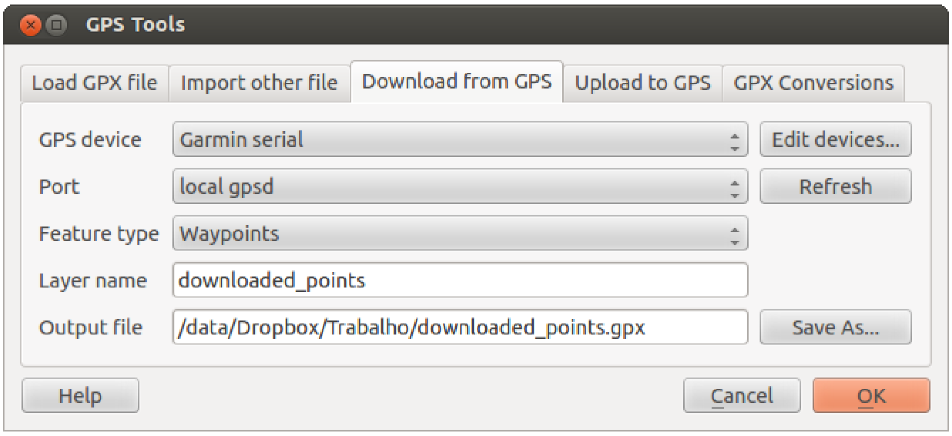


Abb. 19.2: Das Downloadwerkzeug

Durch die Angabe des Typs Ihres GPS-Empfängers legen Sie fest, wie GPSBabel mit dem Gerät kommuniziert. Wenn kein vorhandener Typ mit Ihrem Empfänger funktioniert, können Sie einen eigenen, neuen Gerätetyp erstellen (vgl. Abschnitt *Neues GPS-Gerät definieren*).

Der Verbindungsport ist ein Dateiname oder ein anderer Name, den Ihr System als Referenz für den physischen Port benutzt, über den eine Verbindung zum GPS-Empfänger hergestellt wird. Es kann auch einfach USB sein, wenn dies von dem GPS-Gerät unterstützt wird.

- Unter Linux ist dies etwas wie `/dev/ttyS0` oder `/dev/ttyS1`.
- Unter Windows ist dies `COM1` oder `COM2`.

When you click *OK*, the data will be downloaded from the device and appear as a layer in QGIS.

19.1.6 GPS-Daten auf einen Empfänger hochladen

Sie können auch einen Vektorlayer aus |qg| auf einen GPS-Empfänger hochladen, indem Sie den Reiter *Zum GPS hochladen* verwenden. Der Vektorlayer muss dazu ein GPX-Layer sein. Sie wählen dazu einen entsprechenden Layer aus, den Typ Ihres GPS-Empfängers und den Verbindungsport (oder USB). Genau wie beim Reiter *Vom GPS herunterladen* können Sie bei Bedarf auch einen eigenen, neuen Empfänger-Typ erstellen, wenn Ihr Gerät nicht in der Liste auftaucht.

Dieses Werkzeug ist sehr nützlich, besonders im Zusammenspiel mit den Vektorfunktionen von QGIS. Sie können eine Karte laden, ein paar Wegpunkte oder Routen digitalisieren und es dann auf Ihren GPS-Empfänger hochladen.

19.1.7 Neues GPS-Gerät definieren

There are lots of different types of GPS devices. The QGIS developers can't test all of them, so if you have one that does not work with any of the device types listed in the *Download from GPS* and *Upload to GPS* tools, you can define your own device type for it. You do this by using the GPS device editor, which you start by clicking the *Edit Devices* button in the download or the upload tab.

To define a new device, you simply click the *New Device* button, enter a name, enter download and upload commands for your device, and click the *Update Device* button. The name will be listed in the device menus in the upload and download windows – it can be any string. The download command is the command that is used to download data from the device to a GPX file. This will probably be a GPSBabel command, but you can 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.

`%type` wird ersetzt durch `-w`, wenn Sie Wegpunkte herunterladen, `-r` wenn es eine Route ist und `-t`, wenn es sich um Spuren handelt. GPSBabel erfährt dadurch, um welchen GPS-Datentyp es sich handelt.

`%in` wird ersetzt durch den Namen des Verbindungsports und `%out` durch den Namen, den Sie für die GPX-Datei gewählt haben. Wenn Sie also einen neuen Gerätetyp mit dem Kommando `gpsbabel %type -i garmin -o gpx %in %out` (es handelt sich hierbei um das Standard Kommando für einen ‚Garmin Serial') definieren und diesen benutzen, um Wegpunkte von Port `/dev/ttyS0` in die Datei `output.gpx` zu schreiben, dann ersetzt QGIS die Schlüsselworte und startet das Kommando `gpsbabel -w -i garmin -o gpx /dev/ttyS0 output.gpx`.

Das Kommando hinaufladen wird benutzt, um die Daten auf Ihren GPS-Empfänger zu transferieren. Es werden dazu die gleichen Schlüsselworte benutzt, nur dass `%in` durch den Namen der hochzuladenen GPX-Datei und `%out` durch den Namen des Verbindungsports ersetzt wird.

You can learn more about GPSBabel and its available command line options at https://www.gpsbabel.org.

Wenn Sie einmal einen eigenen Gerätetypen erstellt haben, wird dieser in der Liste der GPS-Geräte dauerhaft angezeigt werden.

19.1.8 Downloaden von Punkten/Spuren von GPS Einheiten

As described in previous sections QGIS uses GPSBabel to download points/tracks directly in the project. QGIS comes out of the box with a pre-defined profile to download from Garmin devices. Unfortunately there is a bug #6318 that does not allow create other profiles, so downloading directly in QGIS using the GPS Tools is at the moment limited to Garmin USB units.

Garmin GPSMAP 60cs

MS Windows

Install the Garmin USB drivers from https://www8.garmin.com/support/download_details.jsp?id=591

Verbinden Sie die Einheit. Öffnen Sie GPS Werkzeuge und verwenden Sie `type=garmin serial` und `port=usb:`. Füllen Sie die Felder *Layername* und *Ausgabedatei* aus. Manchmal scheint es Probleme beim Speichern in einen bestimmten Ordner zu geben, wenn Sie etwas wie `c:\temp` verwenden funktioniert es für gewöhnlich.

Ubuntu/Mint GNU/Linux

Es wird zuerst ein Eintrag über die Rechte des Gerätes benötigt, wie beschrieben auf https://wiki.openstreetmap.org/wiki/USB_Garmin_on_GNU/Linux. Sie können versuchen eine Datei `/etc/udev/rules.d/51-garmin.rules` zu erstellen, die diese Regel enthält.

```
ATTRS{idVendor}=="091e", ATTRS{idProduct}=="0003", MODE="666"
```

Danach ist es nötig sicher zu gehen das das `garmin_gps` Kernelmodul nicht geladen ist

```
rmmod garmin_gps
```

and then you can use the GPS Tools. Unfortunately there seems to be a bug #7182 and usually QGIS freezes several times before the operation work fine.

BTGP-38KM Datenlogger (nur Bluetooth)

MS Windows

Der bereits erwähnte Bug lässt es nicht zu, dass Daten innerhalb von |qgl heruntergeladen werden, also müssen Sie GPSBabel aus der Kommandozeile heraus oder mit Hilfe seiner Schnittstelle verwenden.

```
gpsbabel -t -i skytraq,baud=9600,initbaud=9600 -f COM9 -o gpx -F C:/GPX/aaa.gpx
```

Ubuntu/Mint GNU/Linux

Verwenden Sie den gleichen Befehl (oder Einstellungen wenn Sie die GPSBabel GUI verwenden) wie in Windows. Unter Linux ist vielleicht üblich eine Nachricht wie folgt zu erhalten

```
skytraq: Too many read errors on serial port
```

es ist einfach eine Frage des aus- und anschalten des Dateloggers und es erneut zu versuchen.

BlueMax GPS-4044 Datenlogger (sowohl BT als auch USB)

MS Windows

Bemerkung: Es muss seine Treiber installieren bevor man es unter Windows 7 verwendet. Sehen Sie in den Herstellerseiten für den richtigen Download nach.

Downloaden mit GPSBabel, mit USB und BR, gibt immer einen Fehler heraus wie

```
gpsbabel -t -i mtk -f COM12 -o gpx -F C:/temp/test.gpx
mtk_logger: Can't create temporary file data.bin
Error running gpsbabel: Process exited unsuccessfully with code 1
```

Ubuntu/Mint GNU/Linux

Mit USB

Nachdem Sie das Kabel angeschlossen haben verwenden Sie den `dmesg` Befehl um zu verstehen welcher Port benutzt wird, zum Beispiel `/dev/ttyACM3`. Benutzen Sie dann wie immer GPSBabel aus der Kommandozeile oder der GUI.

```
gpsbabel -t -i mtk -f /dev/ttyACM3 -o gpx -F /home/user/bluemax.gpx
```

Mit Bluetooth

Verwenden Sie Blueman Device Manager um das Gerät zu verbinden und machen Sie es über einen Systemport zugänglich, starten Sie dann GPSBabel.

```
gpsbabel -t -i mtk -f /dev/rfcomm0 -o gpx -F /home/user/bluemax_bt.gpx
```

19.2 Live GPS tracking

To activate live GPS tracking in QGIS, you need to select *View ⍰ Panels* *GPS Information Panel* or press `Ctrl+0`. You will get a new docked window on the left side of the canvas.

Es sind 4 Bildschirme im GPS Tracking Fenster möglich:

- GPS position coordinates and an interface for manually entering vertices and features
- GPS Signalstärke von Satellitenverbindungen
- GPS Optionsansicht (siehe *figure_gps_options*)

With a plugged-in GPS receiver (has to be supported by your operating system), a simple click on *Connect* connects the GPS to QGIS. A second click (now on *Disconnect*) disconnects the GPS receiver from your computer. For GNU/Linux, gpsd support is integrated to support connection to most GPS receivers. Therefore, you first have to configure gpsd properly to connect QGIS to it.

Warnung: Wenn Sie Ihre Position in die Oberfläche aufnehmen wollen müssen Sie erst einen neuen Vektorlayer erstellen und diesen in den Bearbeitungsmodus bringen um Ihren Track aufnehmen zu können.

19.2.1 Positionskoordinaten

If the GPS is receiving signals from satellites, you will see your position in latitude, longitude and altitude together with additional attributes.

19.2.2 GPS Signalstärke

Hier können Sie die Signalstärke der Satelliten von denen Sie Signale empfangen sehen.

19.2.3 GPS Optionen

Wenn es Probleme bei der Verbindung zum GPS-Gerät geben sollte können Sie innerhalb dieser Einstellungen wechseln:

- *Automatisch feststellen*
- *Intern*
- *Serielles Gerät*
- *gpsd* (wählt den Host, Port und das Gerät mit dem Ihr GPS verbunden ist)

A click on *Connect* again initiates the connection to the GPS receiver.

Sie können *Hinzugefügte Objekte automatisch speichern* aktivieren wenn Sie sich im Bearbeitungsmodus befinden. Oder Sie können *Punkte automatisch hinzufügen* aktivieren um Punkte mit einer bestimmten Größe und Farbe der Kartenansicht hinzuzufügen.

Indem Sie das Kontrollkästchen *Cursor* aktivieren, können Sie den Schieberegler verwenden, um den Positionscursor im Kartenfenster kleiner oder größer zu machen.

You can also set an *Acquisition interval (seconds)* and a *Distance threshold (meters)* parameters to keep the cursor still active when the receiver is in static conditions.

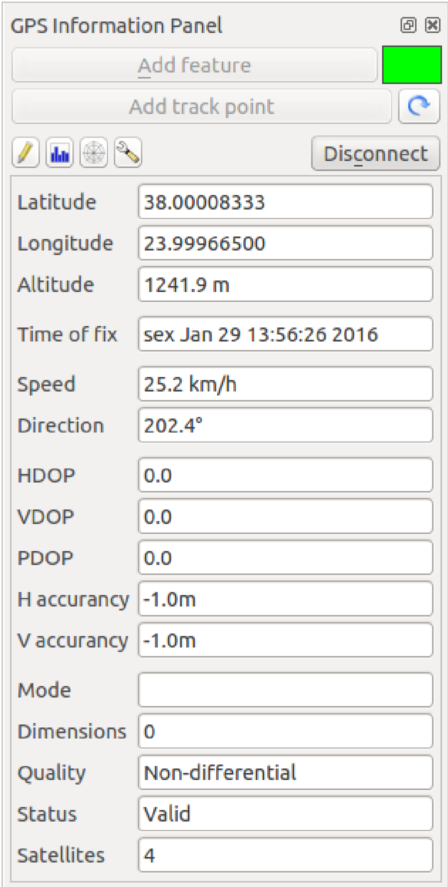


Abb. 19.3: GPS Tracking Position und zusätzliche Attribute

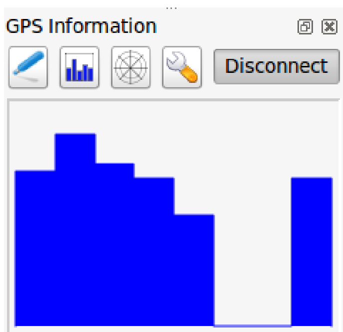


Abb. 19.4: GPS Tracking Signalstärke

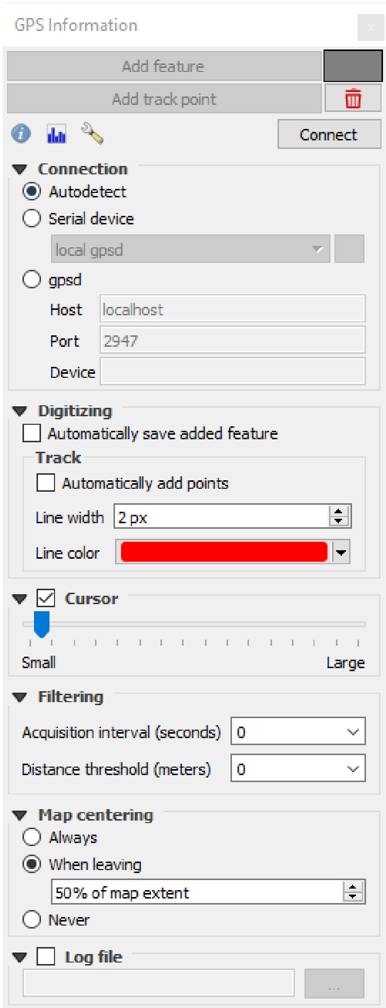


Abb. 19.5: GPS Tracking Optionsansicht

Das Aktivieren des Radioknopfes *Karte zentrieren* ermöglicht es auszuwählen, wie das Kartenfenster aktualisiert werden soll. Dies enthält ‚immer beim Verlassen', wenn die aufgenommenen Koordinaten den Bereich des Kartenfensters verlassen oder ‚niemals', um die Kartenausschnitt beizubehalten.

Schliesslich können Sie das Kontrollkästchen *Logdatei* aktivieren und einen Pfad angeben, wo die Logdateien über die GPS-Messung abgelegt werden.

If you want to set a feature manually, you have to go back to Position and click on *Add Point* or *Add Track Point*.

19.2.4 Ein Bluetooth GPS fürs Live Tracking anbinden

Mit |qg| 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 angebundenen COM Ports aus den Bluetooth Eigenschaften resultieren.

Machen Sie die Kopplung für die Verbindung nachdem das GPS erkannt wurde. Normalerweise ist der Authorisationskode `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 |qg| 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.

19.2.5 GPSMAP 60cs verwenden

MS Windows

Easiest way to make it work is to use a middleware (freeware, not open) called GPSGate.

Launch the program, make it scan for GPS devices (works for both USB and BT ones) and then in QGIS just click *Connect* in the Live tracking panel using the *Autodetect* mode.

Ubuntu/Mint GNU/Linux

Wie unter Windows ist der einfachste Weg einen Server, in diesem Fall GPSD, dazwischen zu benutzen, also

```
sudo apt install gpsd
```

Laden Sie dann das `garmin_gps` Kernelmodul

```
sudo modprobe garmin_gps
```

Und verbinden Sie dann die Einheit. Überprüfen Sie dann mit `dmesg` die aktuelle von dem Gerät verwendete Einheit, zum Beispiel `/dev/ttyUSB0`. Starten Sie jetzt gpsd

```
gpsd /dev/ttyUSB0
```

Und verbinden Sie sich zuletzt mit dem |qg| Live Tracking Tool.

19.2.6 BTGP-38KM Datenlogger verwenden (nur Bluetooth)

Sie können GPSD (unter Linux) oder GPSGate (unter Windows) mühelos verwenden.

19.2.7 BlueMax GPS-4044 Datenlogger verwenden (sowohl BT als auch USB)

MS Windows

Das Live Tracking funktioniert mit USB und BT Modus, mit oder ohne GPSGate, benutzen Sie einfach den  *Automatisch feststellen* Modus oder stellen Sie das Tool auf den richtigen Port ein.

Ubuntu/Mint GNU/Linux

Für USB

Das Live Tracking funktioniert sowohl mit GPSD

```
gpsd /dev/ttyACM3
```

oder ohne es, indem man das |qg| 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 |qg| Live Tracking Tool direkt mit dem Gerät verbindet (zum Beispiel `/dev/rfcomm0`).

KAPITEL 20

Authentifizierungssystem

20.1 Authentifizierungssystem Übersicht

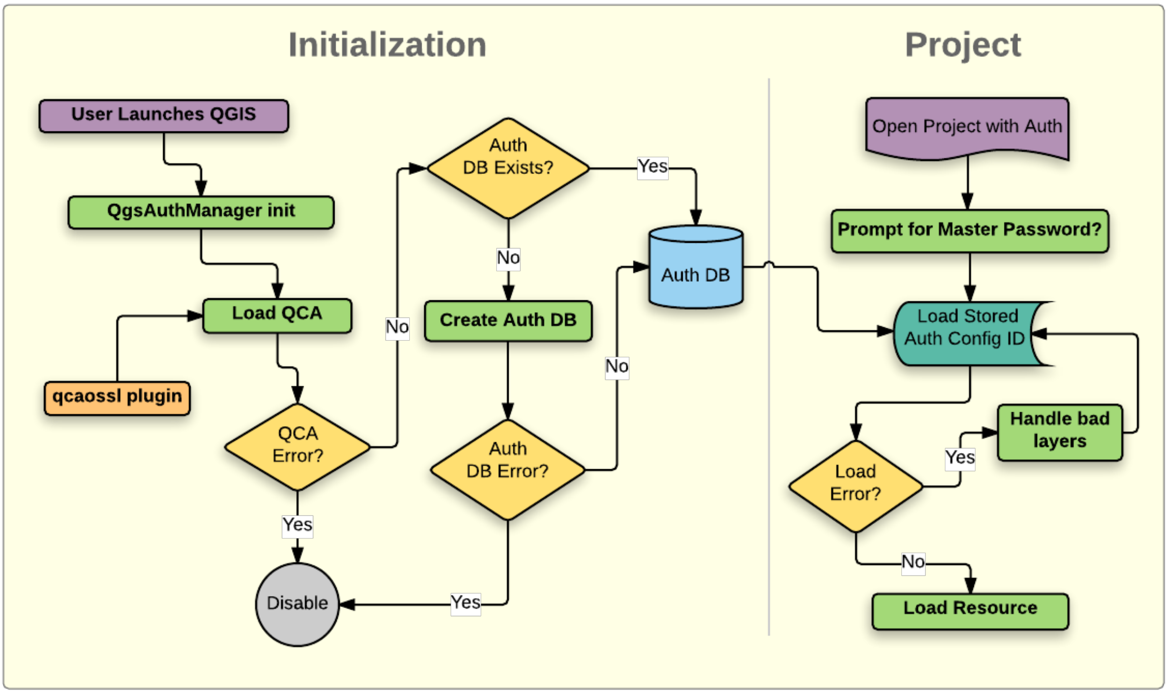


Abb. 20.1: Anatomie des Authentifizierungssystems

20.1.1 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.

20.1.2 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.

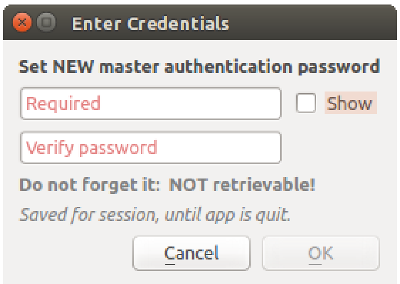


Abb. 20.2: Neues Hauptpasswort eingeben

Bemerkung: Ein Pfad zu einer Datei, die das Hauptpasswort beinhalet, 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.

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 Password drei Mal falsch eingibt, wird ein Dialog Ihnen anbieten, die Datenbank zu löschen.

Abb. 20.3: Hauptpasswort zurücksetzen

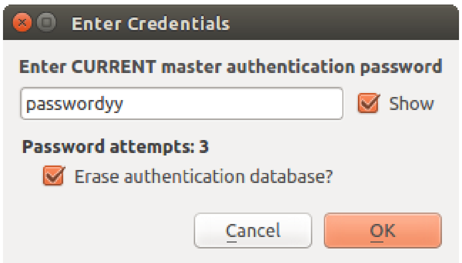


Abb. 20.4: Passwortabfrage nach drei ungültigen Versuchen

20.1.3 Authentifikationskonfiguration

Sie können Authentifizierungseinstellungen über *Konfigurationen* in dem *Authentifizierung* Reiter des QGIS Optionen Dialog (*Einstellungen▸ Optionen*) vornehmen.

Nutzen Sie den Knopf, um neue Konfigurationen hinzuzufügen, den Knopf um Sie zu entfernen und den Knopf um bestehende zu bearbeiten.

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.

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*).

20.1.4 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.

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.

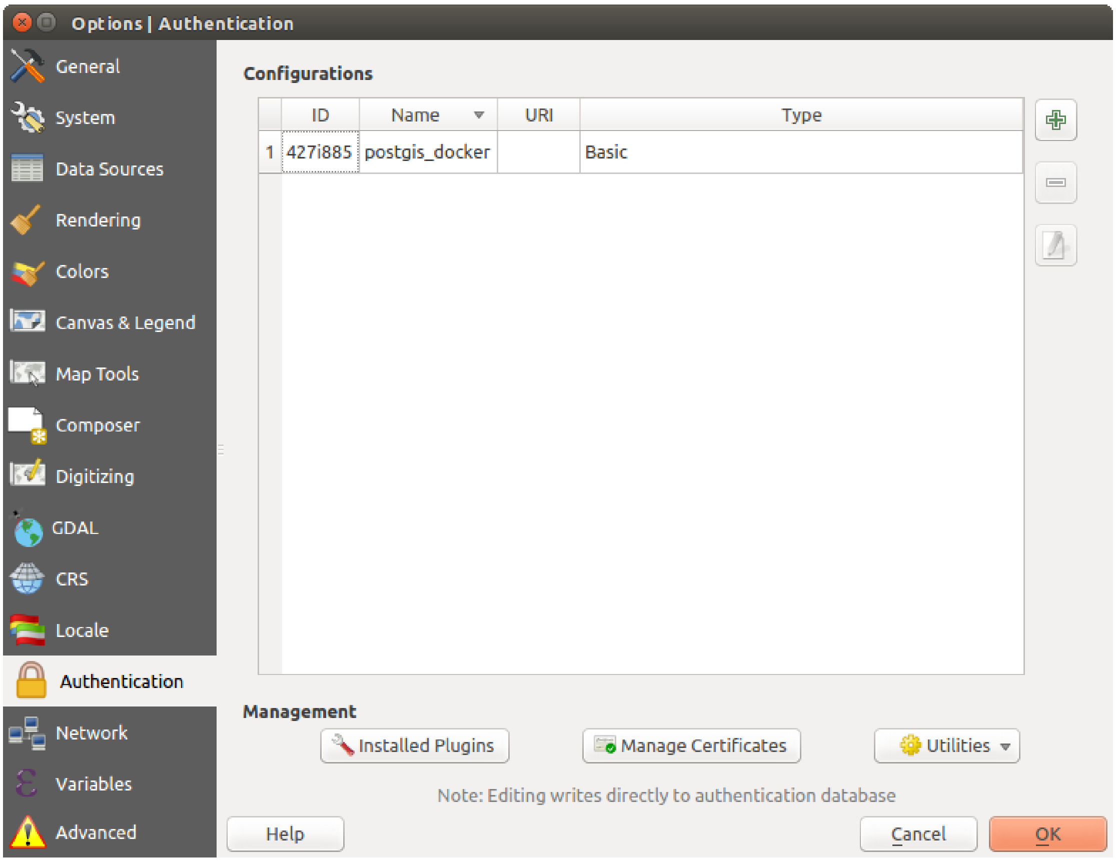


Abb. 20.5: Konfigurationen Editor

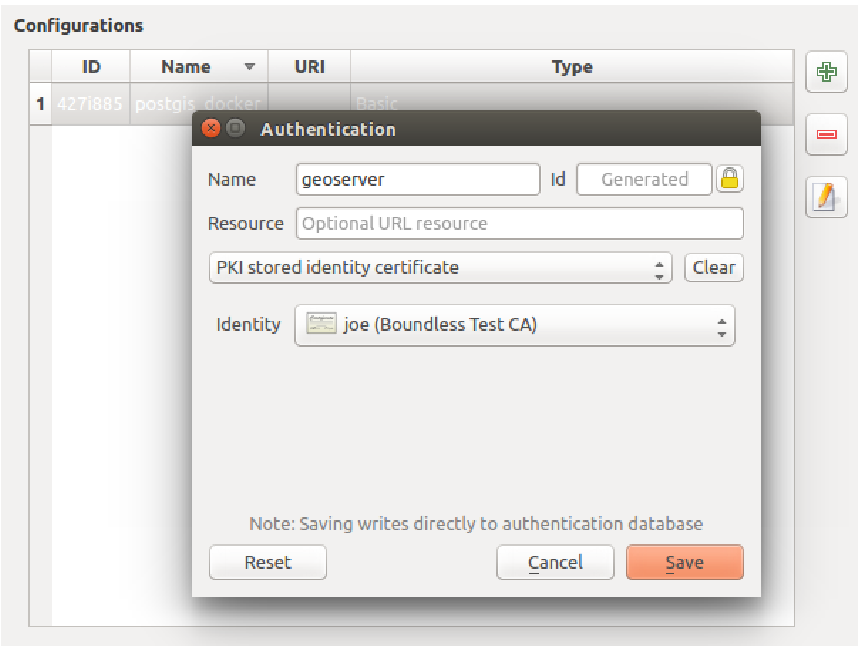


Abb. 20.6: Konfigurationen von innerhalb des Konfigurationseditors hinzufügen

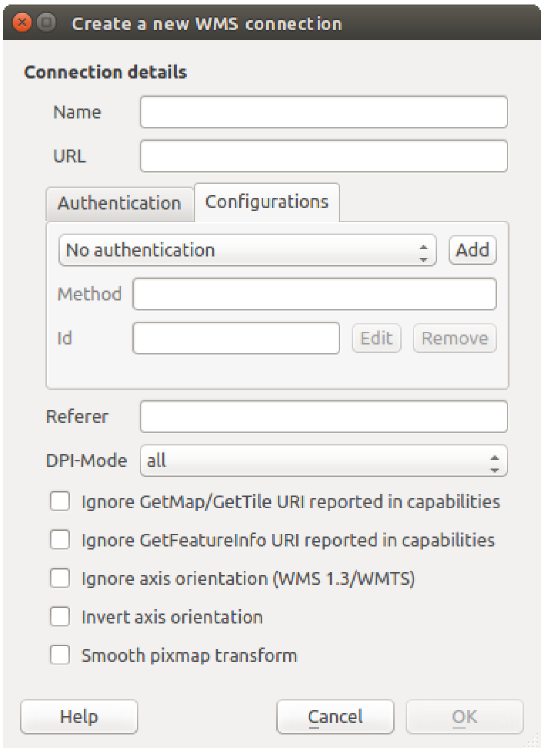
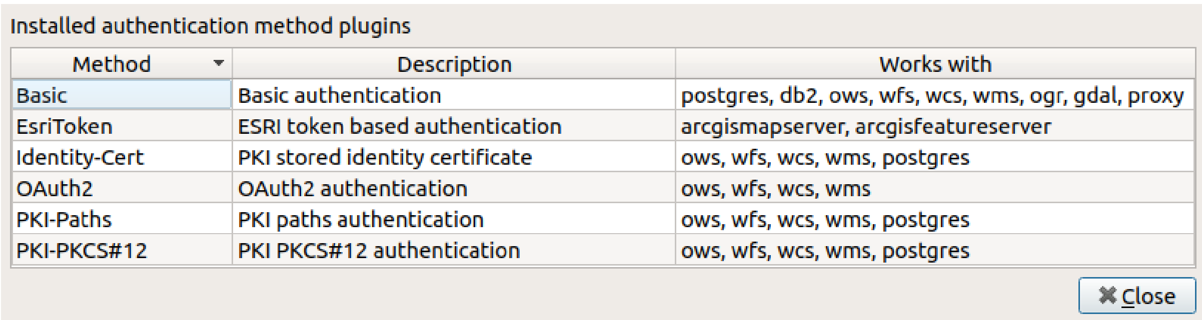


Abb. 20.7: WMS connection dialog showing *Add*, *Edit*, and *Remove* authentication configuration buttons



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. 20.8: Verfügbare Liste von Methodenerweiterungen

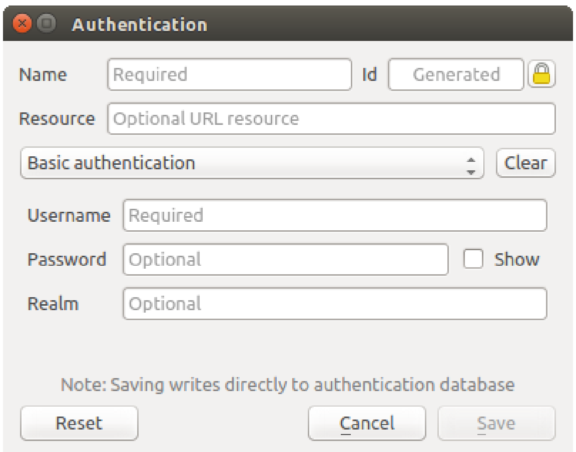


Abb. 20.9: Einfache HTTP Authentifikationskonfiguration

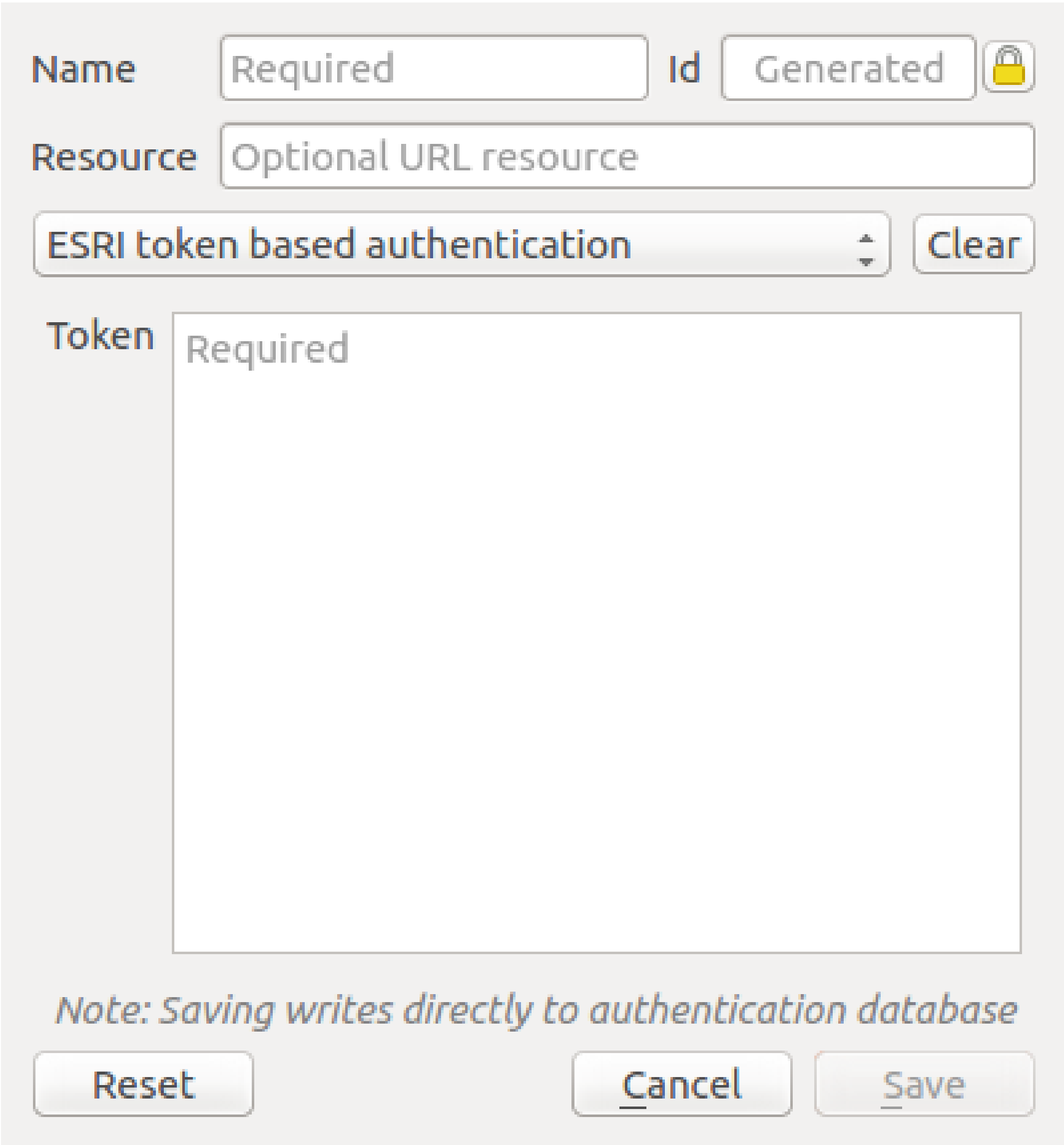


Abb. 20.10: ESRI Token authentication configs

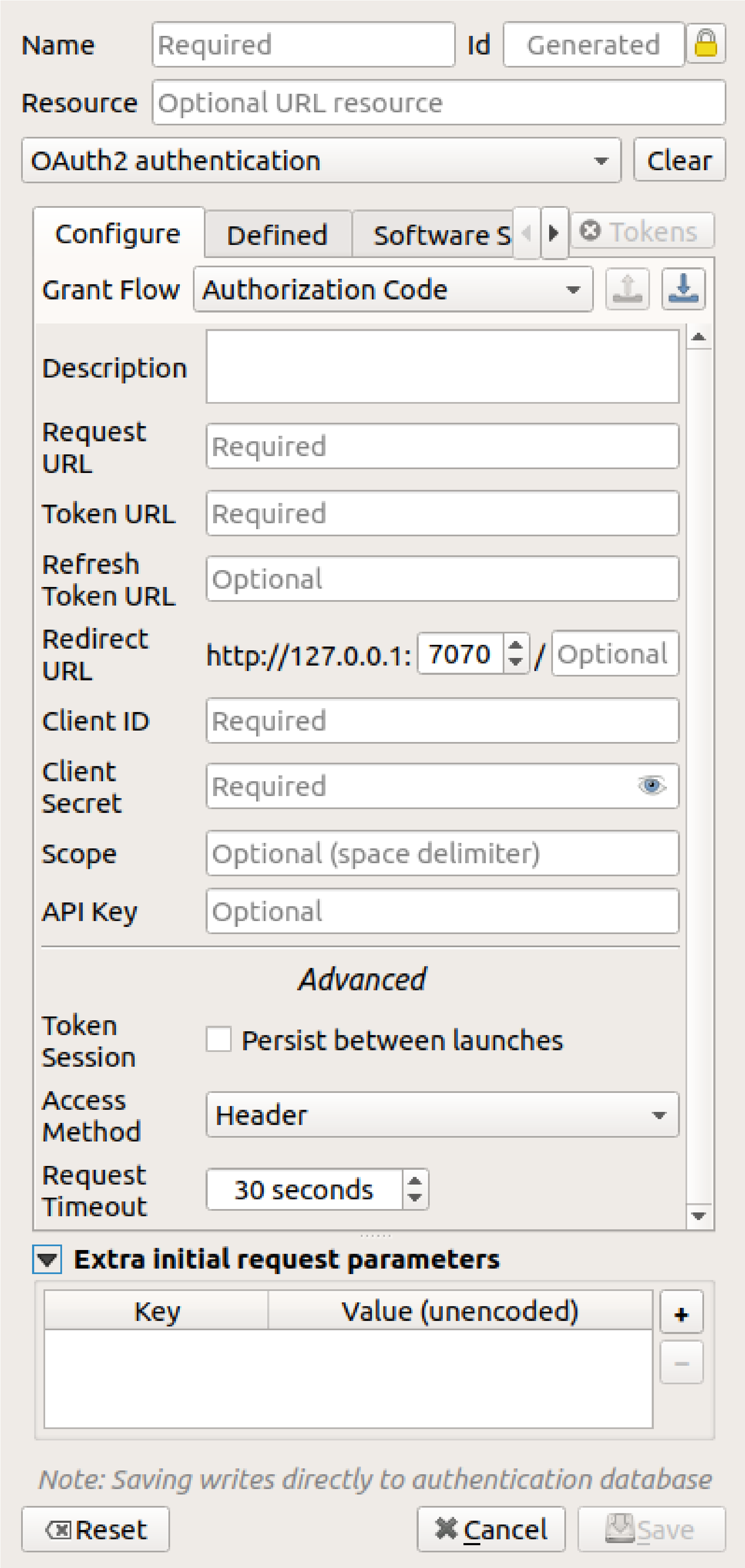


Abb. 20.11: OAuth2 authentication configs

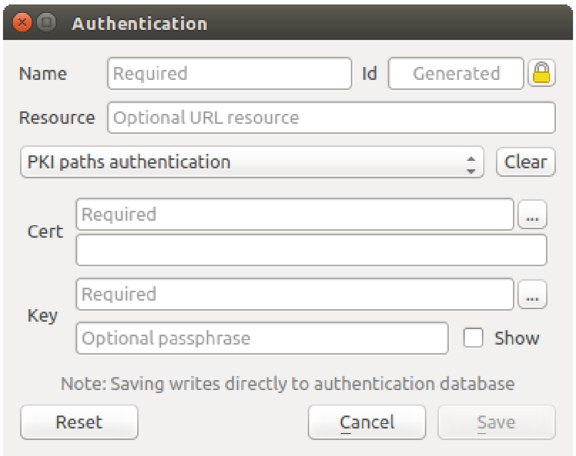


Abb. 20.12: PKI Pfad-Authentifikationskonfiguration

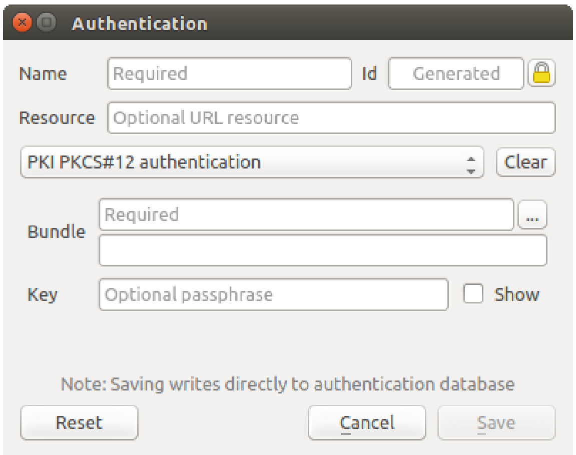


Abb. 20.13: PKI PKCS#12 Dateipfad Authentifikationskonfiguration

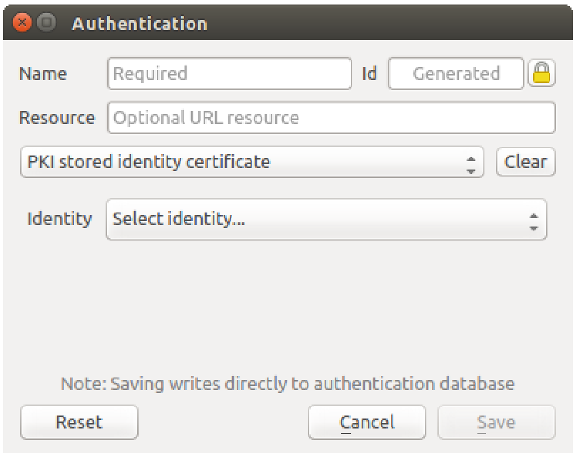


Abb. 20.14: Gespeicherte Identität Authentifikationskonfigurationen

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.

20.1.5 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:

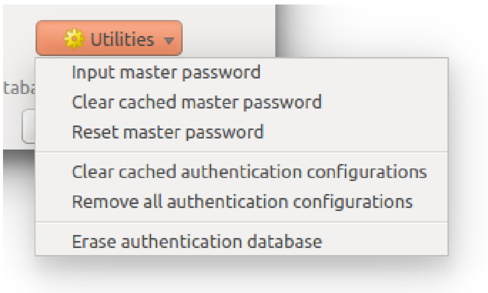


Abb. 20.15: Werkzeug-Menü

- **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.

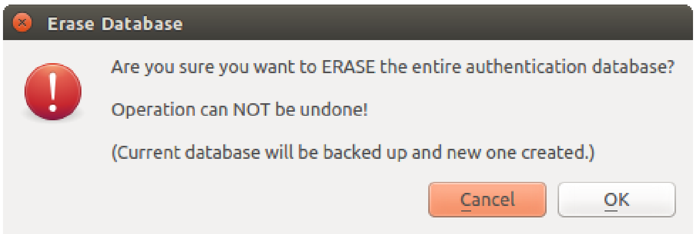


Abb. 20.16: Menü DB Löschverifizierung

20.1.6 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.

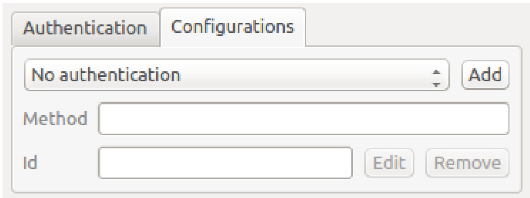


Abb. 20.17: Authentication configuration selector with no authentication

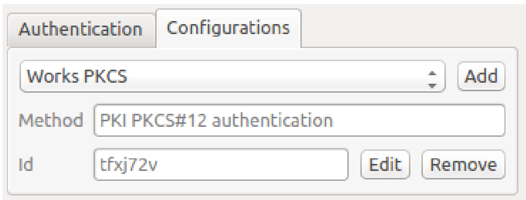


Abb. 20.18: Authentication configuration selector with selected config

20.1.7 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.

20.2 Benutzerauthentifizierung Workflows

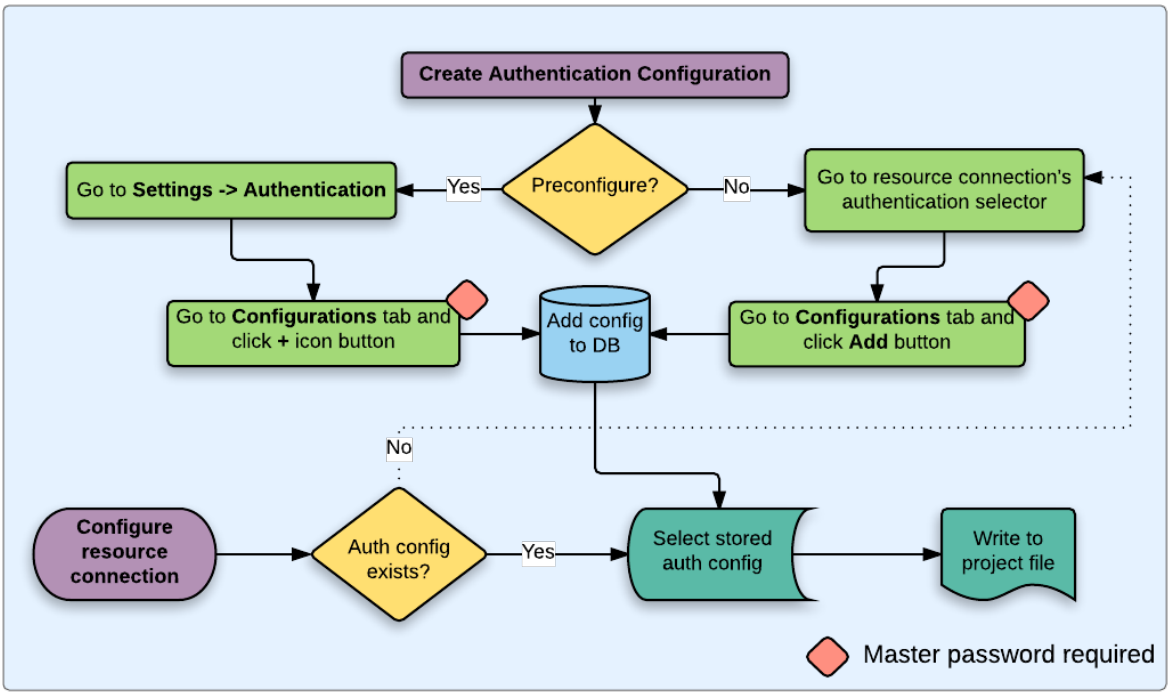


Abb. 20.19: Generischer Benutzer Workflow

20.2.1 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.

20.2.2 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 new auth system, the `key=value` will be an abstracted representation of the credentials, e.g. `authfg=81t21b9`.

20.2.3 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.

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.

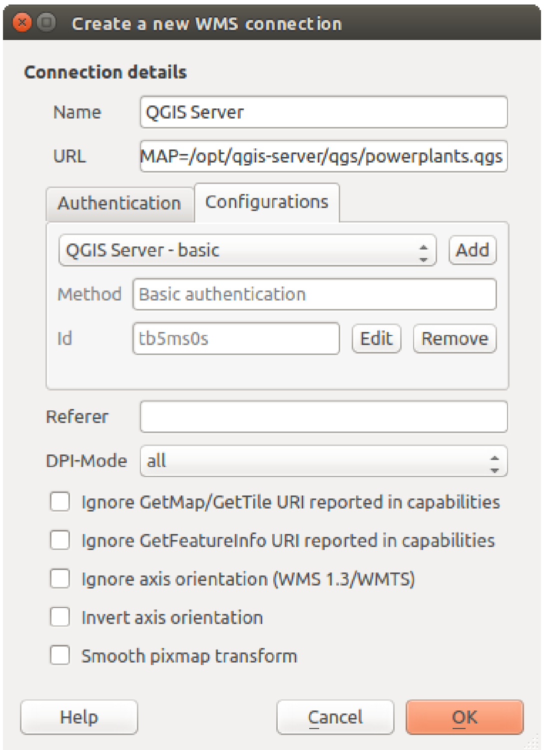


Abb. 20.20: Konfigurieren einer WMS-Verbindung für HTTP BASIC

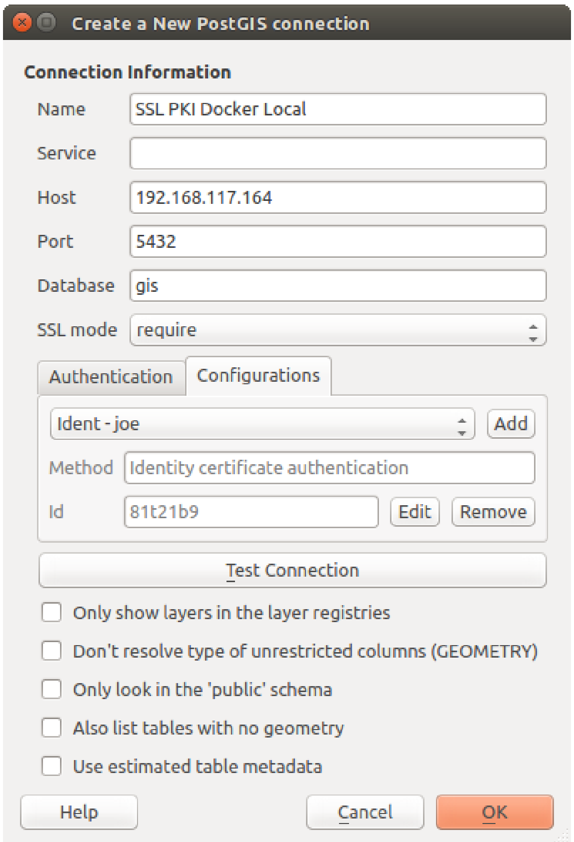


Abb. 20.21: Einstellen einer Postgress-SSL mit PKI-Verbindung

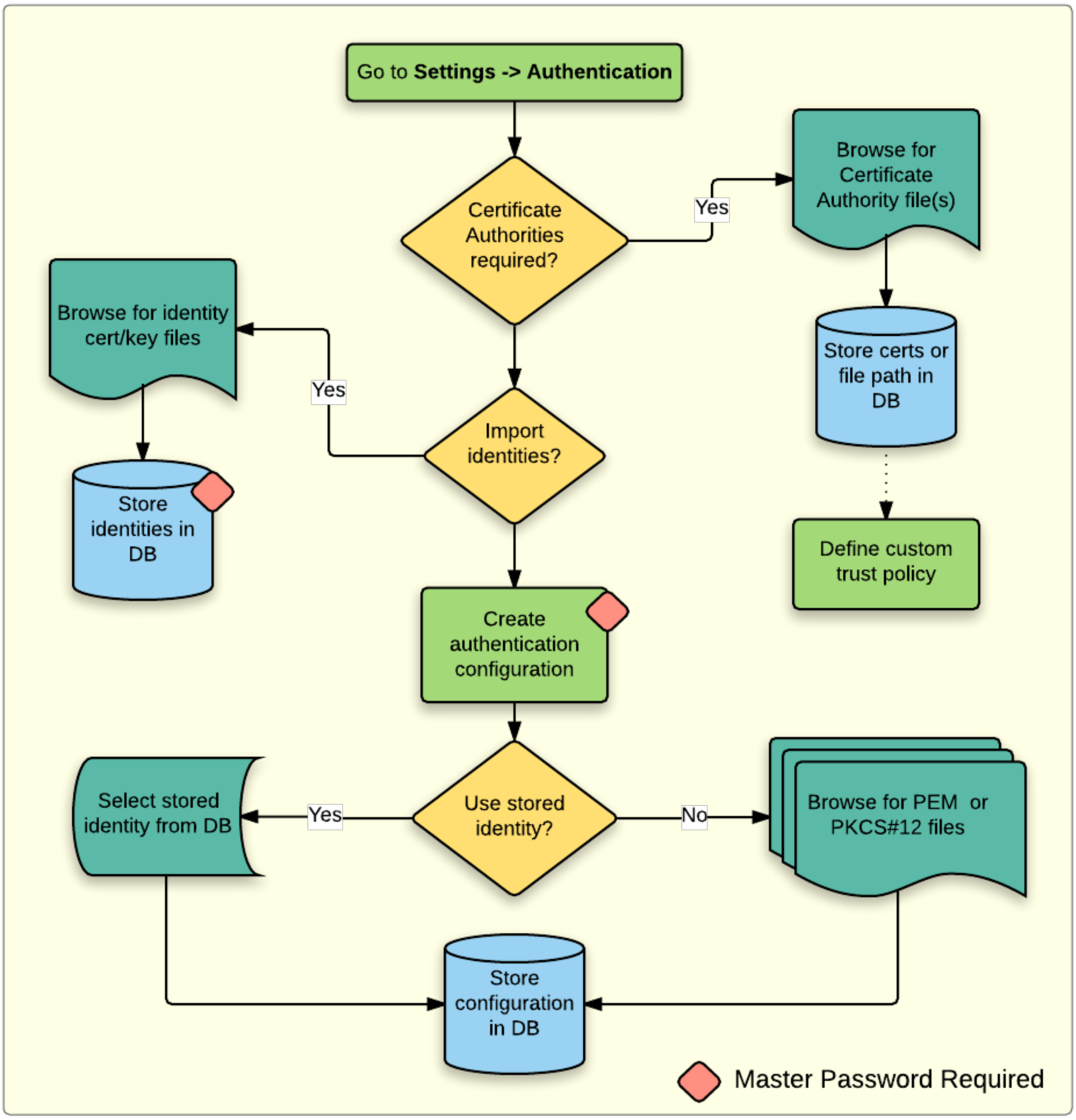


Abb. 20.22: PKI Konfiguration Workflow

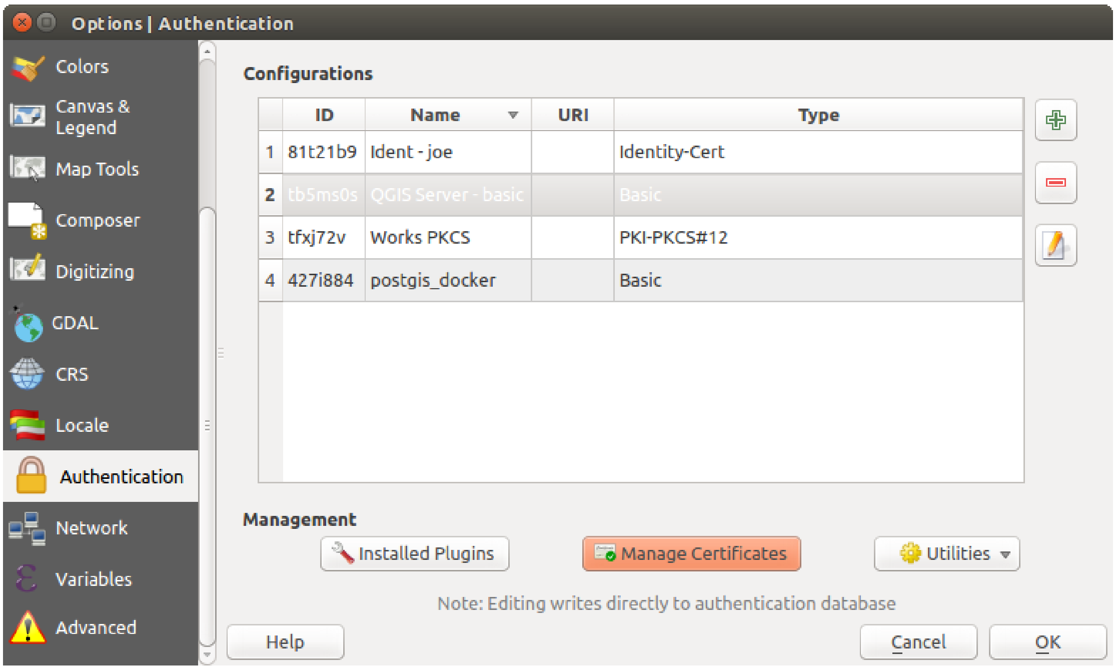


Abb. 20.23: Öffnen der Zertifikatsverwaltung

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 Vertrauenskette 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.)

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.

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.

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.

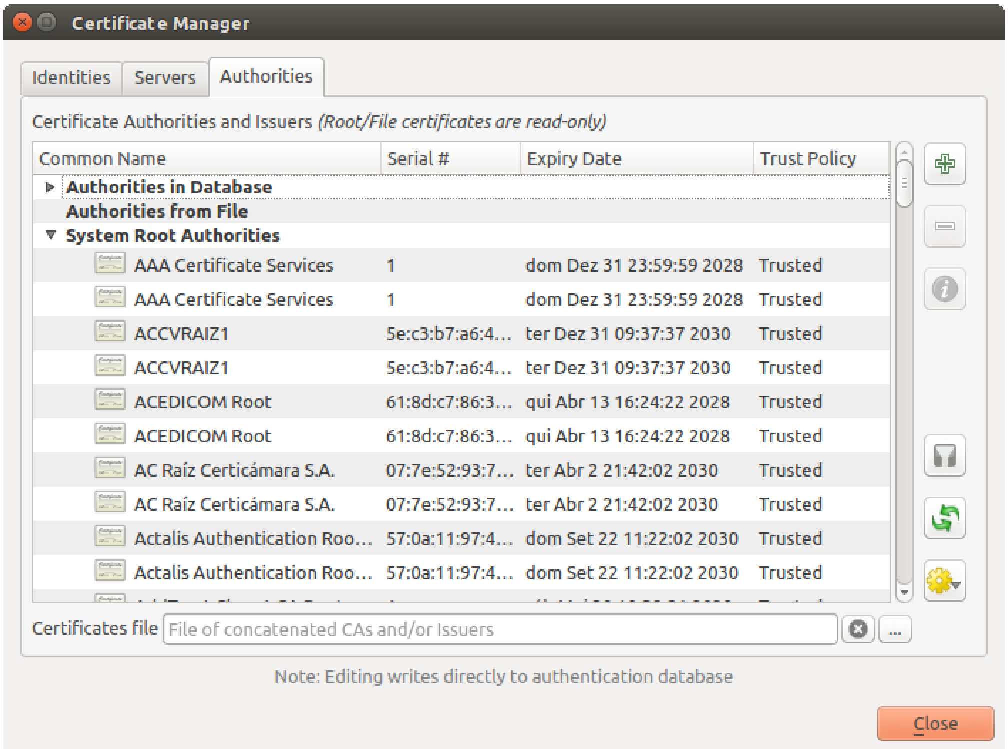


Abb. 20.24: Autoritäteneditor

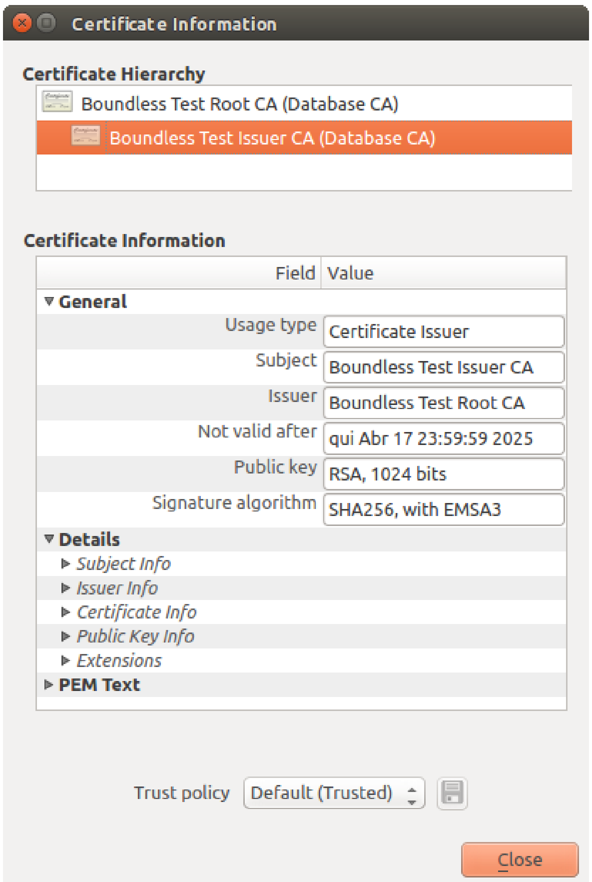


Abb. 20.25: Zertifikatsinformation Dialog

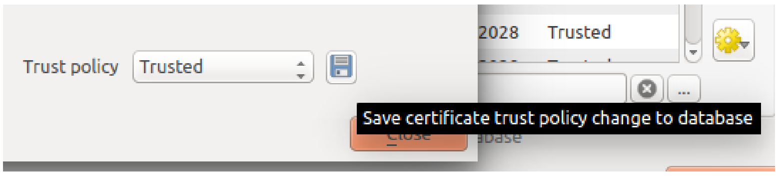


Abb. 20.26: Speichern der Vertrauensrichtlinien Änderungen

Warnung: Die Standard-Vertrauensrichtlinie ändern kann zu Problemen mit sicheren Verbindungen führen.

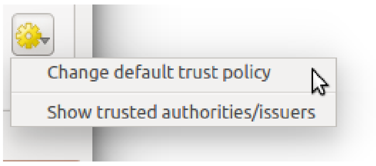


Abb. 20.27: Autoritätenoptionen-Menü

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.

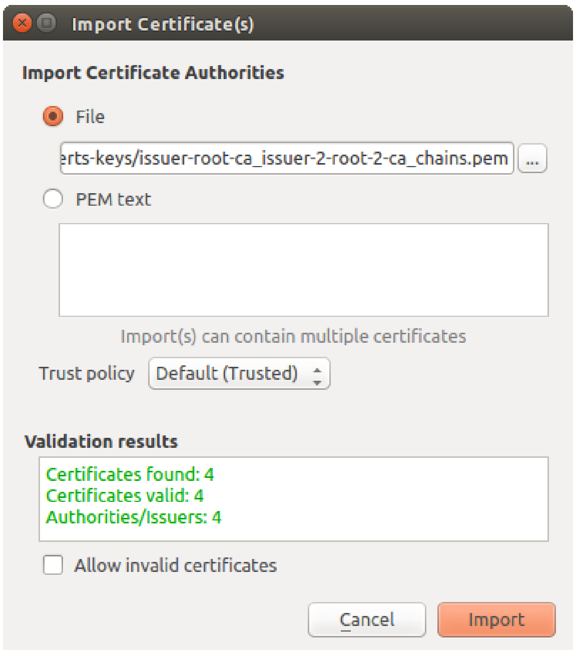


Abb. 20.28: Zertifikat importieren Dialog

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.

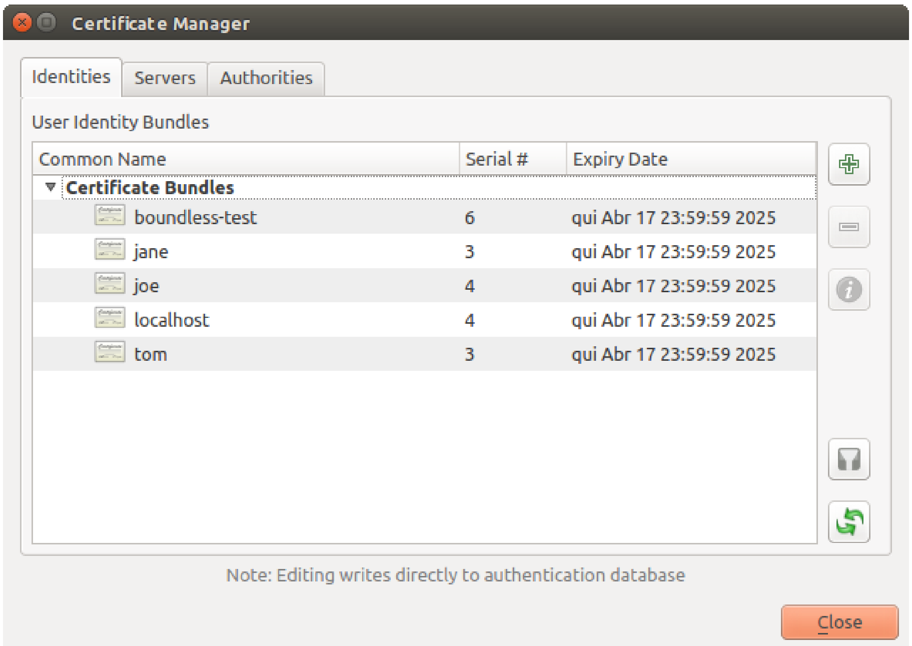


Abb. 20.29: Identitäteneditor

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:*Authoritäten* 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.

20.2.4 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.

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.

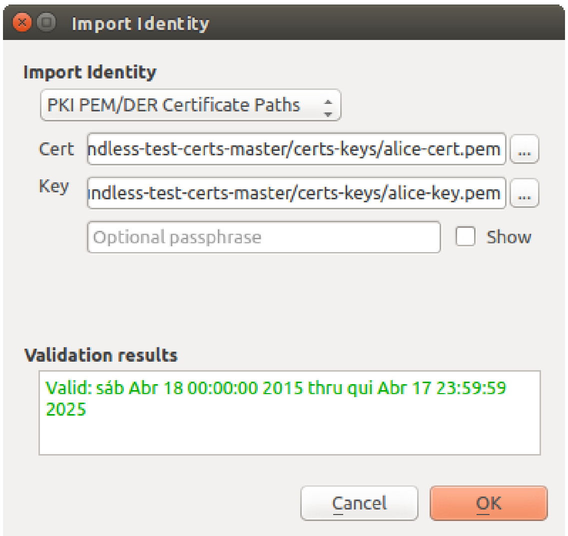


Abb. 20.30: PEM/DER Identität importieren

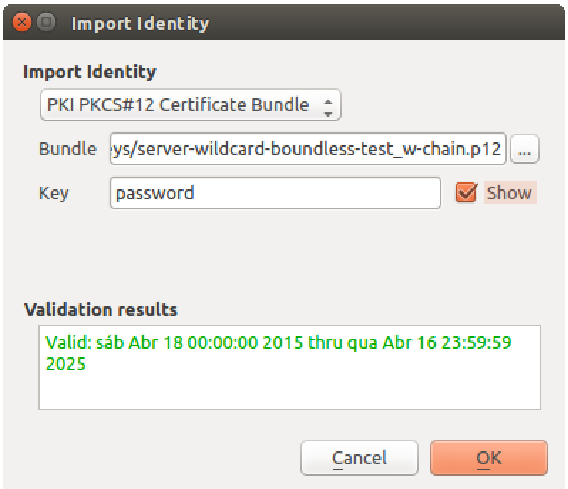


Abb. 20.31: PKCS#12 Identität importieren

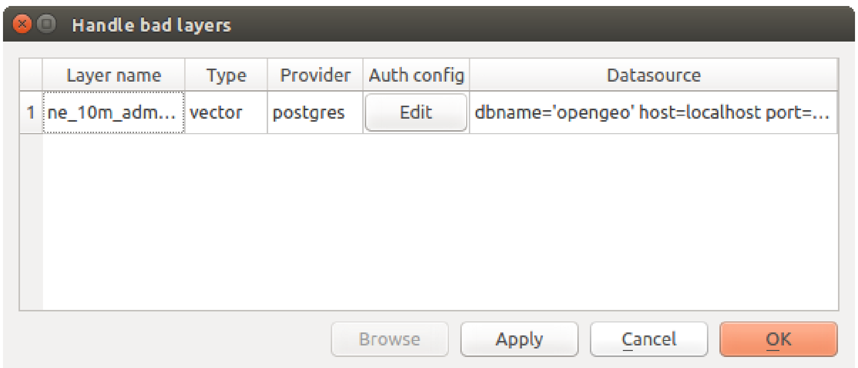


Abb. 20.32: Defekte Layer mit Authentifizierung behandeln

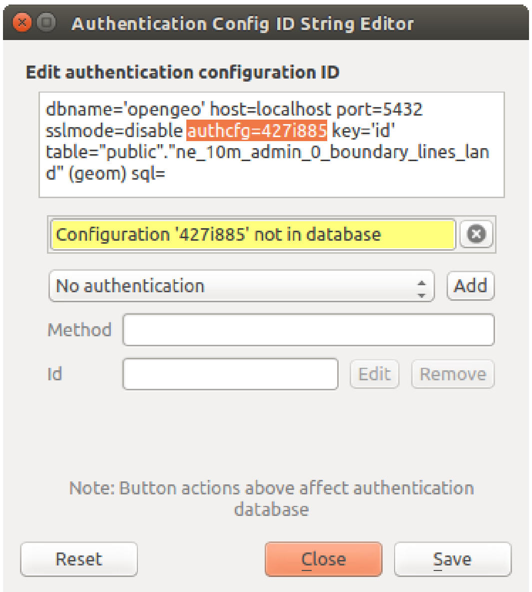


Abb. 20.33: Defekte Layer mit Authentifikationskonfiguration bearbeiten

20.2.5 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 add 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.

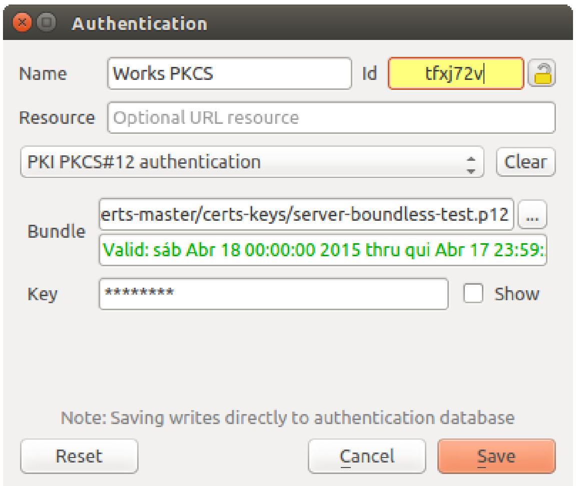


Abb. 20.34: Kennung einer Layer Authentifikationskonfiguration ändern (entsichert gelbes Textfeld)

> **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.

20.2.6 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

Wenn das Authentifizierungssystem initiiert wird, wird der Server die Datei `qgis-auth.db` im Ordner `~/.qgis2/` oder einem durch die Variable `QGIS_AUTH_DB_DIR_PATH` definierten Ordner ablegen. Es kann sein, dass der Server Benutzer kein HOME-Verzeichnis hat. In diesem Fall verwenden Sie die entsprechende Umgebungsvariable, um ein Verzeichnis zu definieren, in dem der Server Benutzer Lese-/Schreibberechtigungen hat und das nicht innerhalb des Web zugänglichen Bereichs liegt.

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.

20.2.7 SSL Serverausnahme

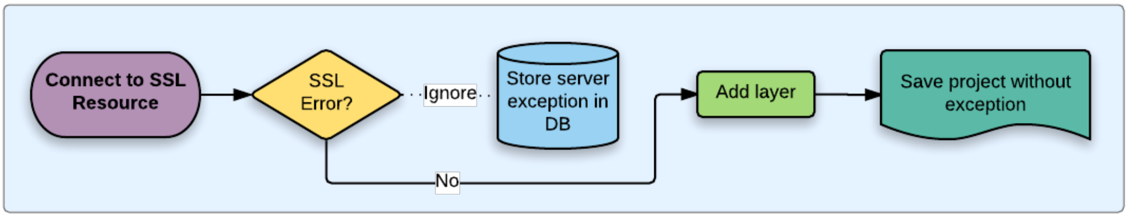


Abb. 20.35: SSL Serverausnahmen

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 sich den 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):

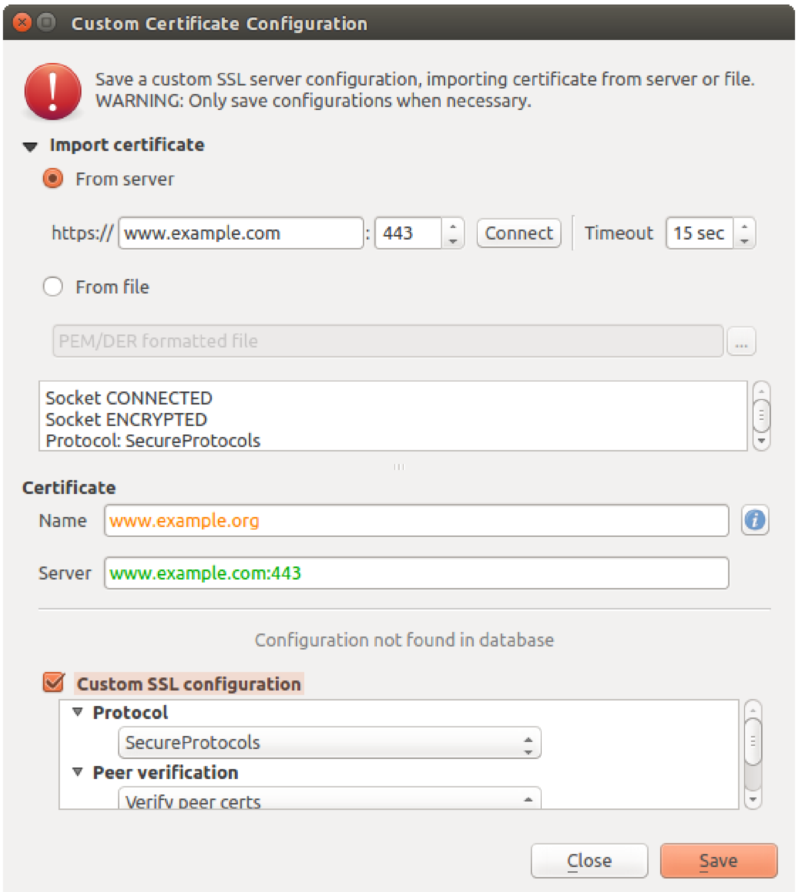


Abb. 20.36: Manuell eine Konfiguration hinzufügen

Sobald eine SSL-Konfiguration in der Datenbank gespeichert wird, kann sie bearbeitet oder gelöscht werden.

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.

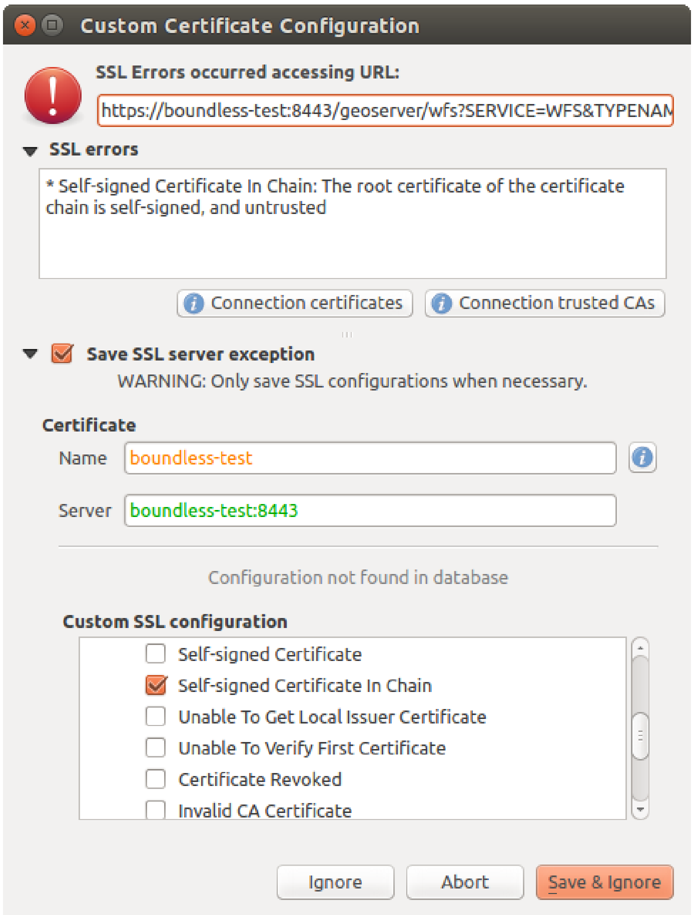


Abb. 20.37: Konfiguration während einem SSL Fehler hinzufügen

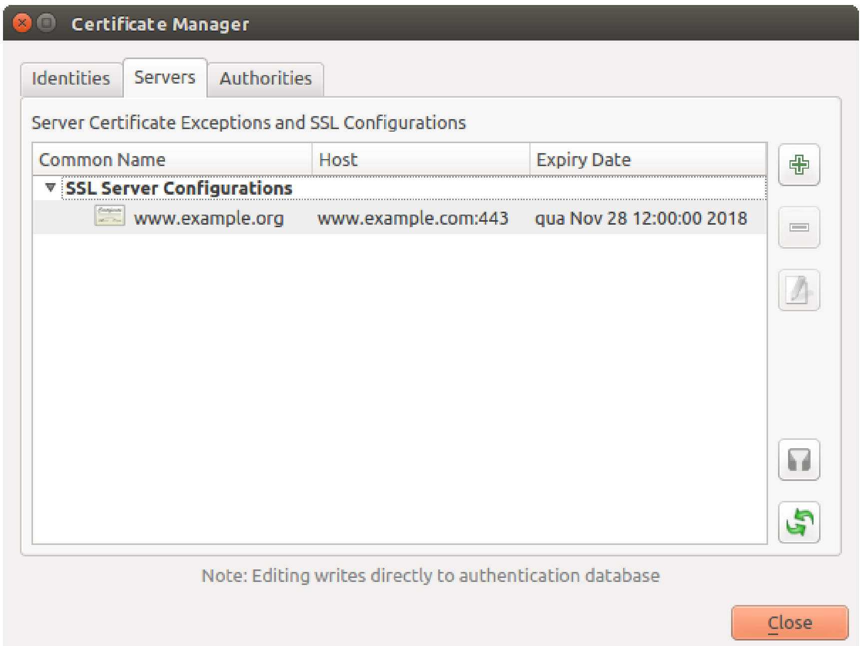


Abb. 20.38: Bestehende SSL Konfiguration

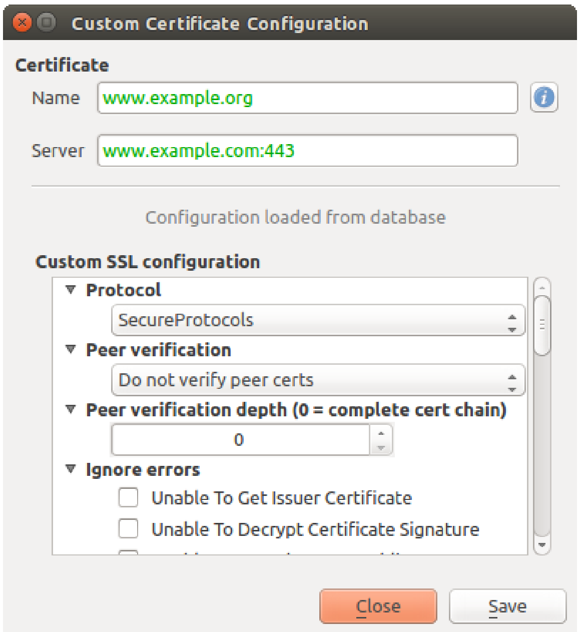


Abb. 20.39: Bestehende SSL Konfiguration bearbeiten

20.3 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 verflogen, welche Plugins der Benutzer speziell hat
- erlaubt auf Authentifizierungssystem zuzugreifen, obwohl es schwierig sein kann, abzuleiten welches Plugin tatsächlich abruft.
- 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 beudeuten, 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.

20.3.1 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-ossl (OpenSSL) Plugin zur Laufzeit geladen wird. Das QCA-ossl Plugin ist mit den OpenSSL-Libs direkt verknüpft. Packagers wäre die benötige 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-ossl` zur Laufzeit nicht gefunden wird.

KAPITEL 21

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 GRAS 7 oder für beide Versionen gleichzeitig enthalten (Binärdateien haben unterschiedliche Dateinamen). Nur eine Version des Anbieters/Plugin kann zur Laufzeit jedoch geladen werden.

21.1 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 eine 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`.

Mehr GRASS Beispiel `LOCATIONs` sind unter der GRASS Website unter https://grass.osgeo.org/download/sample-data/ verfügbar.

21.2 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.

21.3 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 .

21.4 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.

21.5 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.

21.6 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
- Schließe Mapset
- GRASS-Werkzeugkiste öffnen
- Aktuelle GRASS-Region darstelllen
- GRASS Optionen

21.7 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.

21.8 Information zur GRASS-Datenbank

GRASS Daten werden in einem Ordner gespeichert, der als GISDBASE bezeichnet wird. Standardmäßig wird der Ordner `grassdata` genannt und er muss erstellt worden sein, bevor man beginnt, mit dem GRASS Plugin in QGIS zu arbeiten. Innerhalb dieses Ordners sind die GRASS Daten als Projekte in Unterordnern genannt `LOCATIONs` organisiert. Jede `LOCATION` ist durch ein Koordinatenbezugssystem, Kartenprojektion und eine geographische Grenze definiert und kann darüberhinaus weitere Unterordner `MAPSETs` (Unterordner des Ordners `LOCATION`) besitzen, um die Layer der `LOCATION` weiter z.B. thematisch oder räumlich zu unterteilen (siehe Neteler & Mitasova 2008 in *Literatur und Internetreferenzen*). Um Raster- und Vektorlayer mit den GRASS Modulen zu analysieren, müssen diese zuerst in eine passende GRASS `LOCATION` importiert werden. (Dies ist nicht ganz korrekt. Mit den GRASS Modulen `r.external` und `v.external` können Sie eine ‚read-only' Verknüpfung zu externen durch GDAL/OGR-unterstützte Layer erstellen, ohne die Daten importieren zu müssen. Da dies aber nicht der normale Weg für GRASS Anfänger ist, wird auf diese Möglichkeit nicht näher eingegangen.).

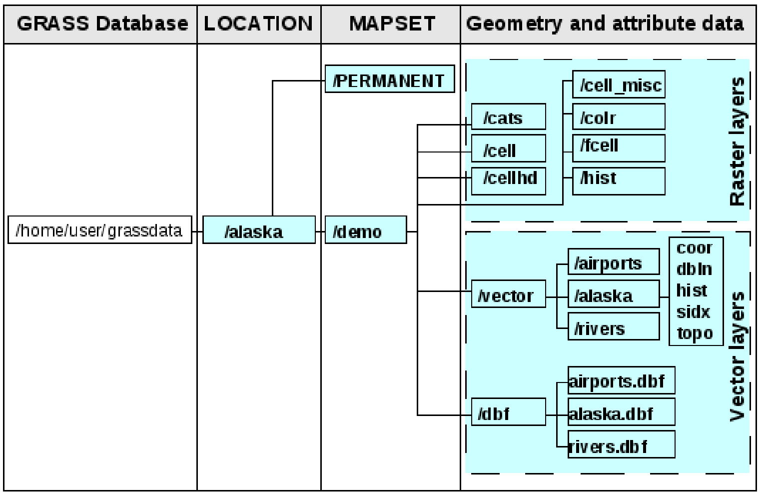


Abb. 21.1: GRASS Daten in der Alaska LOCATION

21.9 Daten in eine GRASS LOCATION importieren

Siehe Abschnitt import_data dnd um herauszufinden, wie die Daten einfach importiert per Drag & Drop in den Browser importert 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 con OGR-unterstützen 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.

21.9.1 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 Shapedatei `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. Wir können diesen Wizard verwenden, um ein neues `MAPSET` innerhalb einer vorhandenen `LOCATION` (siehe Abschnitt *Eine neue GRASS MAPSET erstellen*) oder eine komplett neue `LOCATION` zu erstellen. Wählen Sie *Erstelle neue Location* (siehe *figure_grass_new_location*).
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.

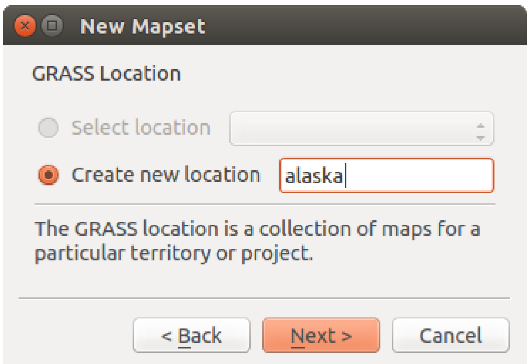


Abb. 21.2: Erstellen einer neuen GRASS LOCATION oder einer neuen MAPSET in QGIS

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' is 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 |qg| ‚Alaska' Datensatz *Beispieldaten herunterladen* enthalten ist und weiter zu Abschnitt *Das GRASS Vektormodell* gehen.

21.9.2 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 Grenzkoordinatenwerte 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 Neues Mapset 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. Wir können den Wizard zur Erstellung eines neuen `MAPSET` innerhalb einer vorhandenen `LOCATION` oder zur Erstellung einer vollständig neuen `LOCATION` verwenden. Klicken Sie auf den Knopf *Wähle Location* (siehe *figure_grass_new_location*) und klicken Sie auf *Weiter*.
6. Enter the name `test` for the new `MAPSET`. Below in the wizard, you see a list of existing `MAPSETs` and corresponding owners.
7. Klicken Sie auf *Weiter*, vergewissern Sie sich, dass alle Angaben richtig sind und klicken auf *Abschließen*.

21.10 Das GRASS Vektormodell

Es ist wichtig, das :index:`GRASS Vektordaten Modell ,vor der Digitalisierung zu verstehen. Im Allgemeinen verwendet GRAS 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 verschiendenen 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

Der beste Weg, etwas über das GRASS Vektormodell und seine Fähigkeiten zu lernen, ist eines der vielen GRASS Tutorials herunterzuladen. Dort wird das Vektormodell tiefergehend beschrieben. Siehe https://grass.osgeo.org/documentation/manuals/ für weitere Informationen, Bücher und Tutorials in verschiedenen Sprachen.

21.11 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*).

21.12 Digitalisieren und Editieren eines GRASS Vektorlayers

GRASS Vektorebenen können mit den Standard QGIS Digitalisierfunktionen 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 Attributtabellen 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 Attributtabellen.

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.

Werkzeugleiste

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.

21.13 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.

21.14 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.

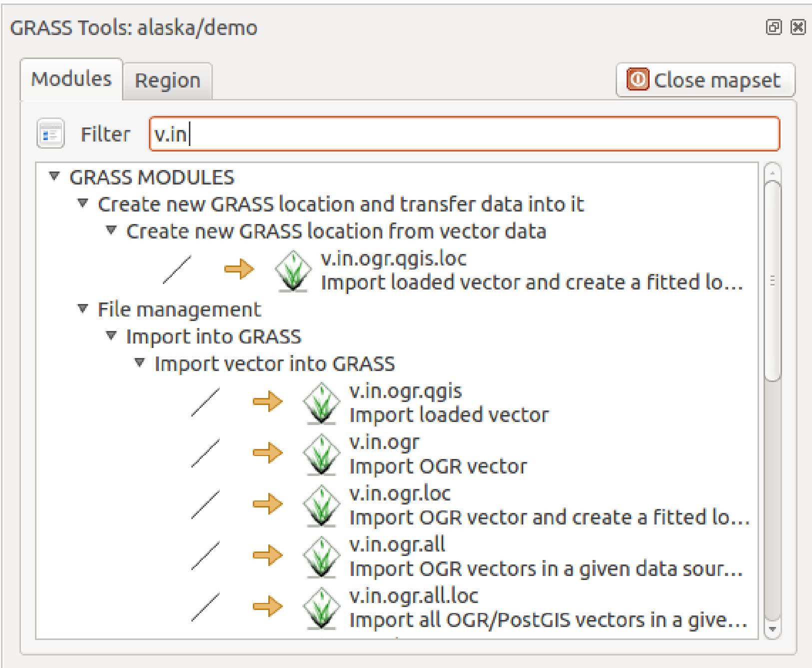


Abb. 21.3: GRASS Werkzeuge und Modulbaum

21.14.1 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.

A complete list of GRASS modules available in the graphical Toolbox in QGIS version 3.10 is available in the GRASS wiki at https://grasswiki.osgeo.org/wiki/GRASS-QGIS_relevant_module_list.

Es ist außerdem möglich, die GRASS Werkzeugkiste anzupassen und weitere Module zu integrieren. Die Herangehensweise ist in Abschnitt *Anpassen der Module* beschrieben.

Wie in der Abbildung *figure_grass_toolbox* gezeigt, kann man die passenden GRASS Module mit Hilfe des thematisch geordneten *Modulbaum* oder mit Hilfe der suchfähigen *Modulliste* finden.

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.

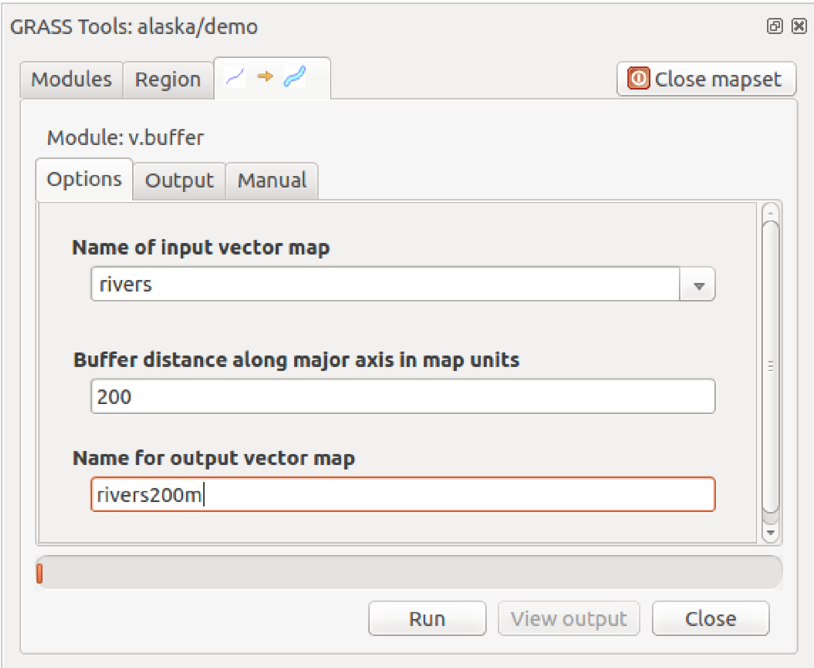


Abb. 21.4: GRASS Werkzeuge Modul Optionen

Die zur Verfügung gestellten Modulparameter sind oftmals nicht vollständig um den Dialog einfach zu halten. Wenn Sie weitergehende Modulparamter 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

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

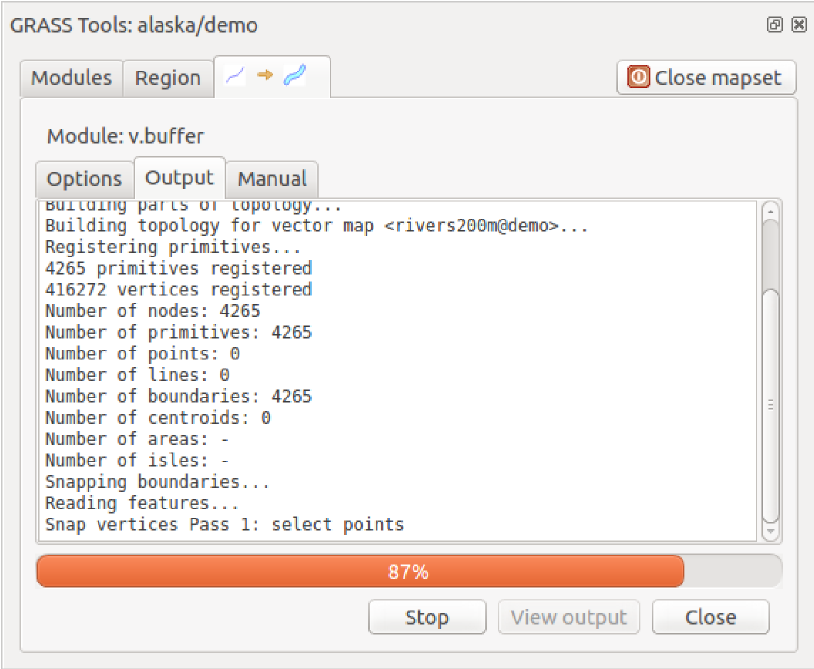


Abb. 21.5: GRASS Werkzeuge Modul Ergebnis

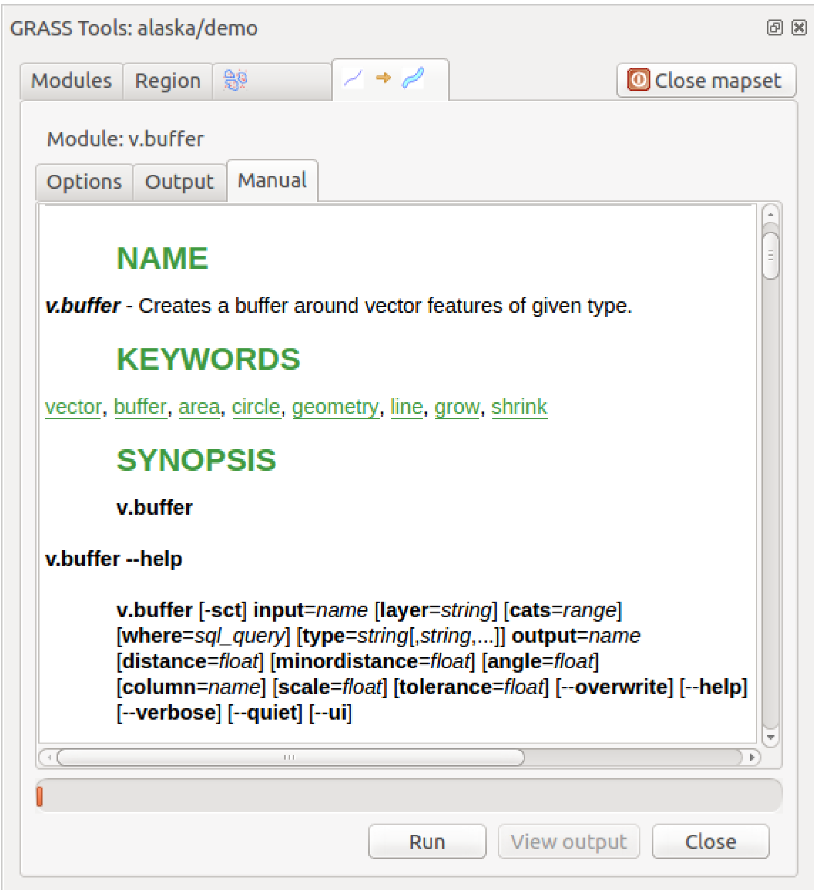


Abb. 21.6: GRASS Werkzeuge Modul Handbuch

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 Ergenisse direkt in der Kartenansicht ansehen, nutzen Sie den Knopf ‚Ergebnis visualisieren' im unteren Bereich des jeweiligen Modulreiters

21.14.2 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* 1,00 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 Ergebnisfenster 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 detailierte Daten nur in einem kleinen Maßstab anzeigen möchte.

Tipp: 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. 21.7: GRASS Moduk 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* 1,00 270 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.

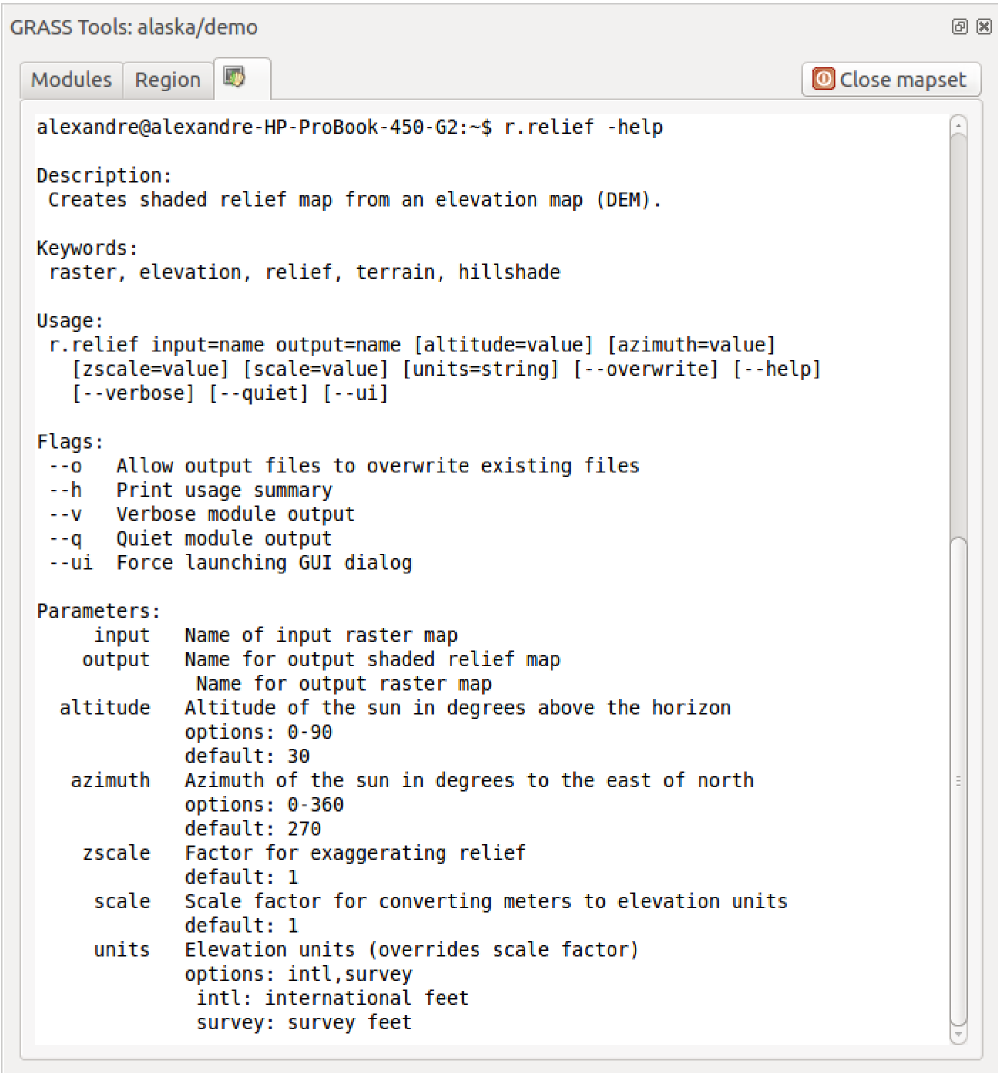


Abb. 21.8: Die GRASS Shell, Modul r.shaded.relief

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.

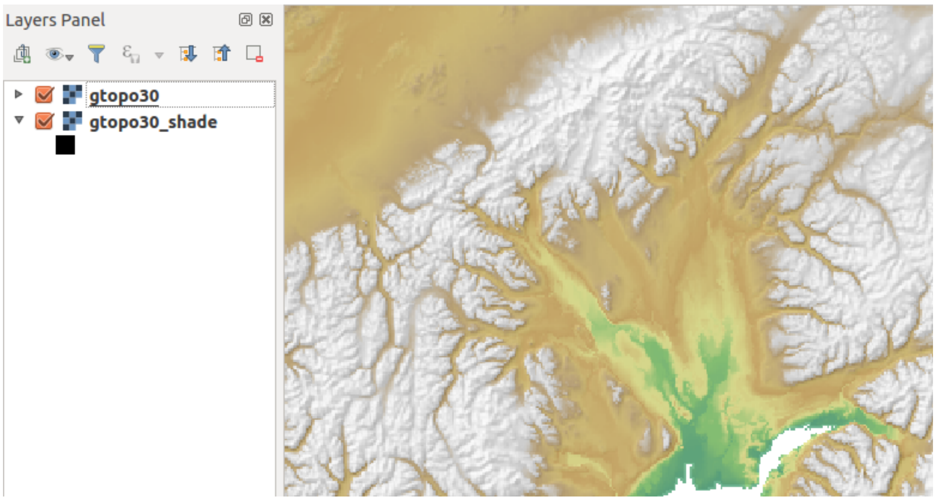


Abb. 21.9: Darstellen einer Schummering erstellt mit dem GRASS Modul r.shaded.relief

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 *Symbology Properties* 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.

21.14.3 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.

KAPITEL 22

QGIS Verarbeitung Umgebung

22.1 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.
- The *Graphical Modeler*: Several algorithms can be combined graphically using the modeler to define a workflow, creating a single process that involves several subprocesses.
- 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.
- The *Batch Processing* interface: This interface allows you to execute batch processes and automate the execution of a single algorithm on multiple datasets.

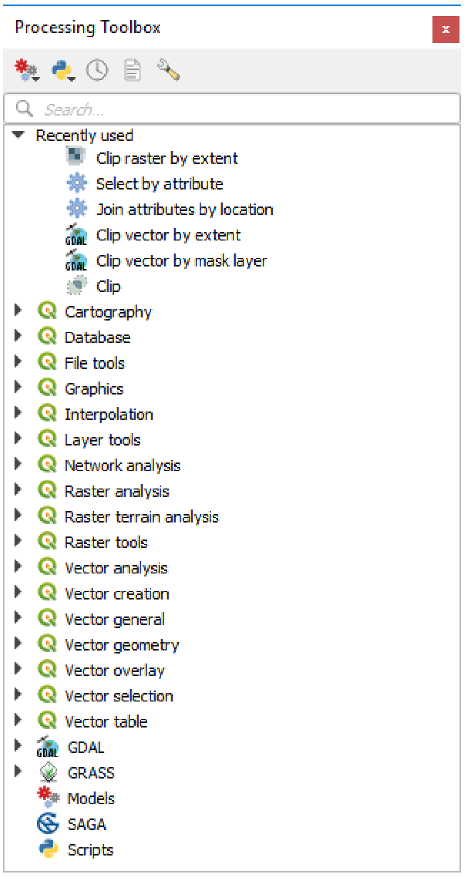


Abb. 22.1: Verarbeitungswerkzeuge

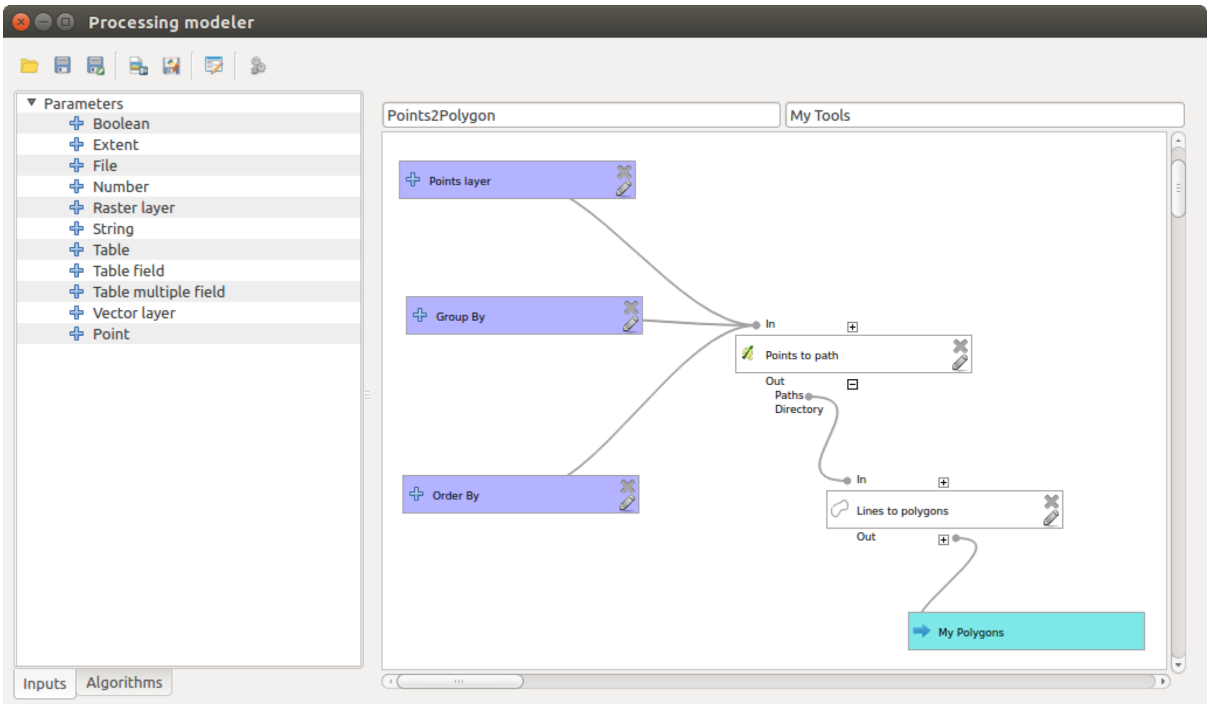


Abb. 22.2: Verarbeitungsmodellierung

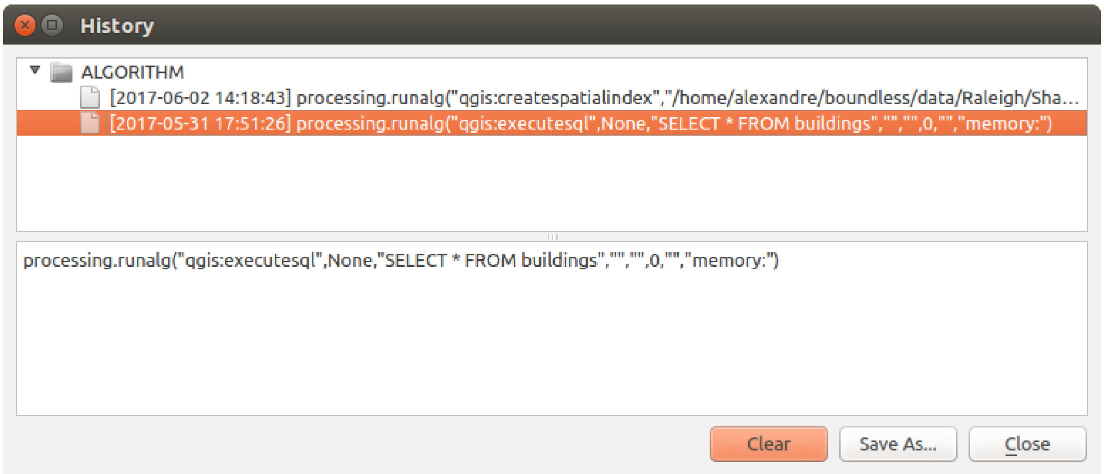


Abb. 22.3: Verarbeitung Protokoll

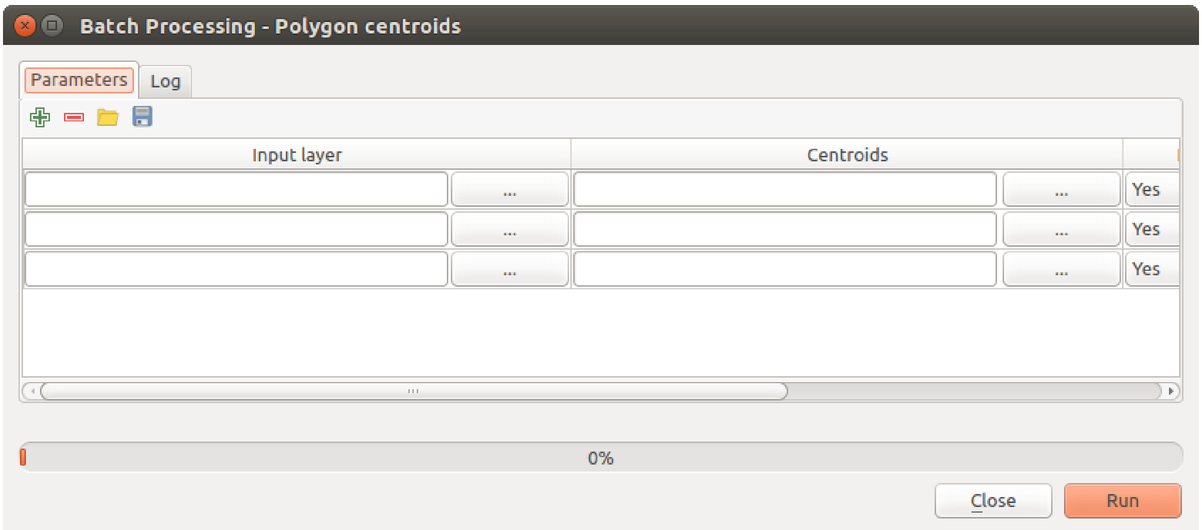


Abb. 22.4: Batch Processing Schnittstelle

In den folgenden Abschnitten werden wir auf jedes dieser Elemente im Detail eingehen.

22.2 Configuring the Processing Framework

As has been mentioned, the configuration menu gives access to a new dialog where you can configure how algorithms work. Configuration parameters are structured in separate blocks that you can select on the left-hand side of the dialog.

Along with the aforementioned *Output folder* entry, the *General* block contains parameters for setting the default rendering style for output layers (that is, layers generated by using algorithms from any of the framework GUI components). 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.

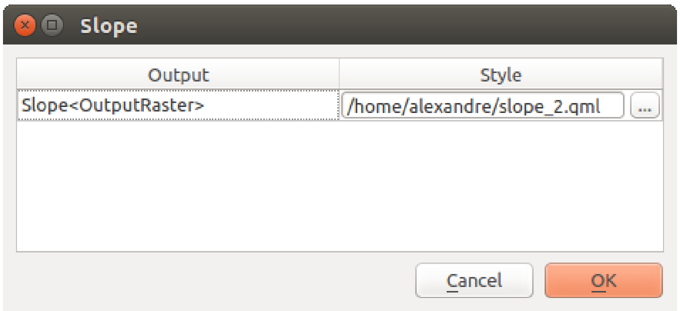


Abb. 22.5: Rendering Styles

Select the style file (`.qml`) that you want for each output and press *OK*.

Other configuration parameters in the *General* group are listed below:

- *Use filename as layer name*. 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.
- *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
- *Use only selected features*. If this option is selected, whenever a vector layer is used as input for an algorithm, only its selected features will be used. If the layer has no selected features, all features will be used.
- *Pre-execution script file* and *Post-execution script file*. These parameters refer to scripts written using the processing scripting functionality, and are explained in the section covering scripting and the console.

Apart from the *General* block in the settings dialog, you will also find a block for algorithm providers. Each entry in this block contains an *Activate* item that you can use to make algorithms appear or not in the toolbox. Also, some algorithm providers have their own configuration items, which we will explain later when covering particular algorithm providers.

22.3 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.

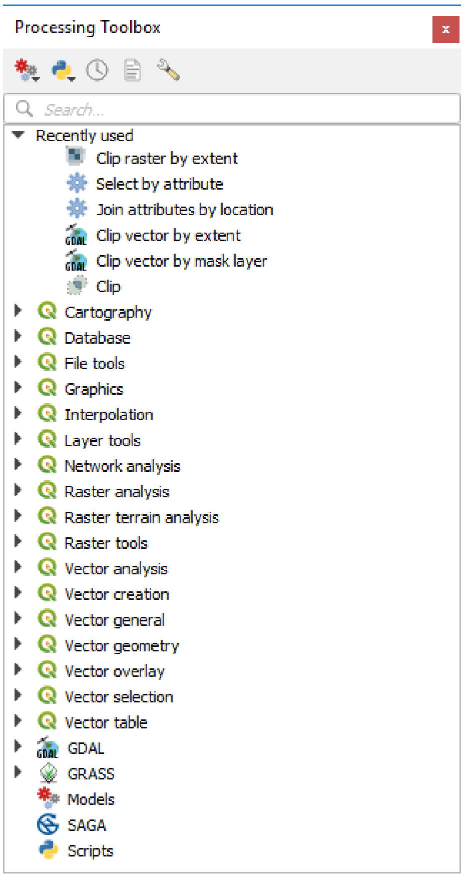


Abb. 22.6: Verarbeitungswerkzeuge

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.

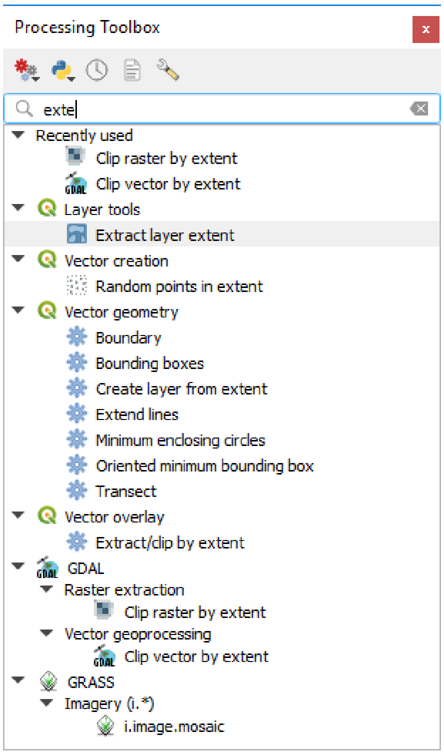


Abb. 22.7: Die Verarbeitungswerkzeuge zeigen Suchergebnisse

To execute a tool, just double-click on its name in the toolbox.

22.3.1 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 figure below is shown (in this case, the dialog corresponds to the `Centroids` algorithm).

Dieser Dialog wird verwendet, um die Eingangs-Werte anzugeben, damit der Algorithmus ausgeführt werden kann. Er zeigt eine Tabelle, in der Eingangswerte und Konfigurationsparameter zu setzen sind. Der Inhalt des Dialogs steht in Abhängigkeit vom Algorithmus, der ausgeführt werden soll, und wird automatisch auf Basis der Anforderungen erstellt.

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 **raster layer**, to select from a list of all such layers available (currently opened) 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.
- A **vector layer**, to select from a list of all vector layers available in QGIS. Layers not loaded in QGIS can be selected as well, as in the case of raster layers, but only if the algorithm does not require a table field selected from the attributes table of the layer. In that case, only opened layers can be selected, since they need to be open so as to retrieve the list of field names available.

  You will see an iterator button by each vector layer selector, as shown in the figure below.

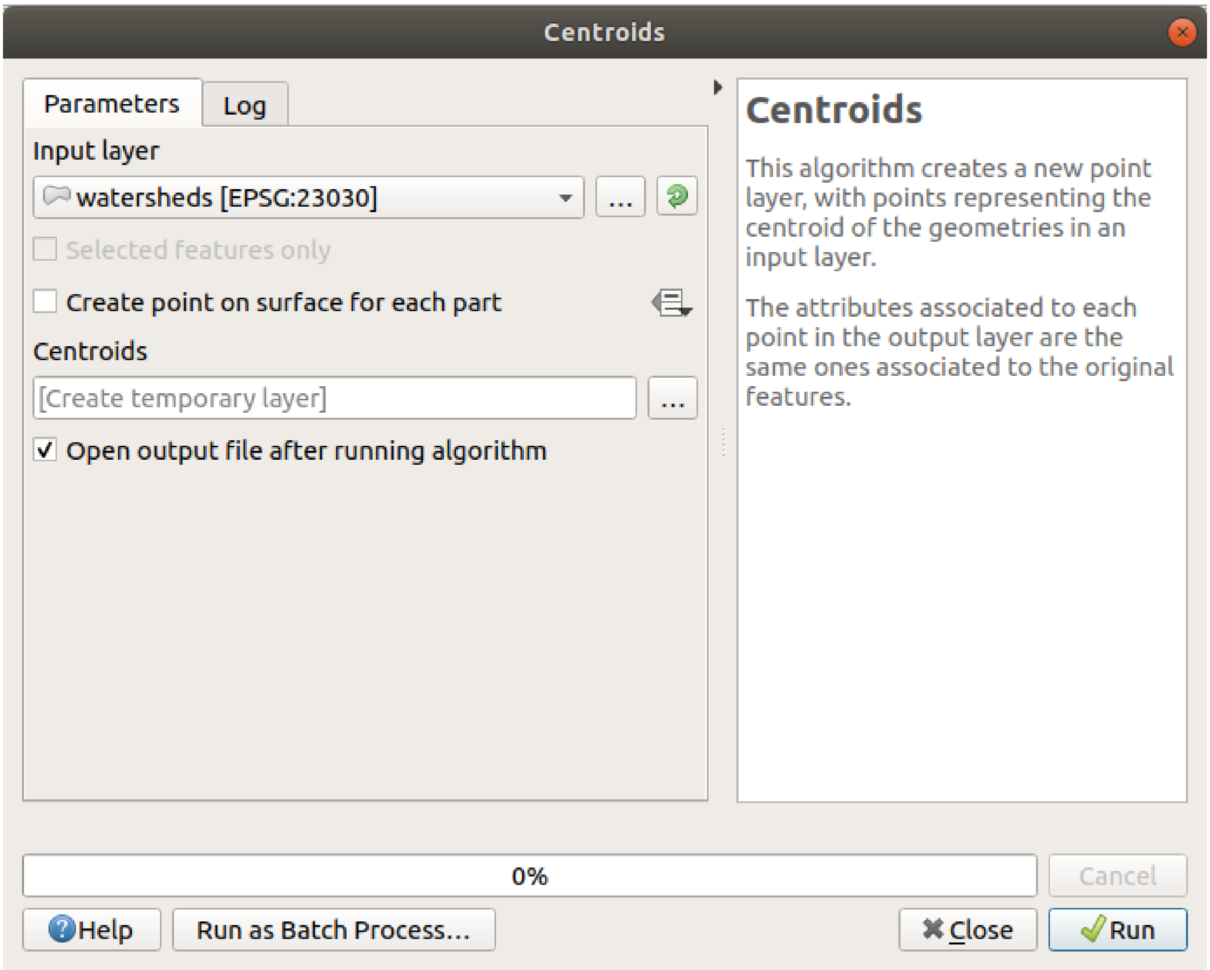


Abb. 22.8: Algorithm Dialog - Parameters

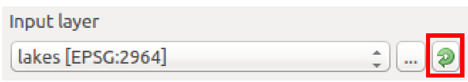


Abb. 22.9: Vektor Interator Knopf

Wenn der Algorithmus mehrere davon enthält können Sie auch nur einen von Ihnen umschalten. Wenn der zu einer Vektoreingabe gehörende Knopf umgeschaltet ist wird der Algorithmus iterativ mit jedem seiner Objekte ausgeführt anstatt nur einmal für den ganzen Layer und erstellt so viele Ausgaben wie der Algorithmus ausgeführt wurde. Dies ermöglicht es den Ablauf zu automatisieren wenn alle Objekte in einem Layer separat verarbeitet werden müssen.

Bemerkung: By default, the parameters dialog will show a description of the CRS of each 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.

- A **table**, to select from a list of all available in QGIS. Non-spatial tables are loaded into QGIS like vector layers, and in fact they are treated as such by the program. Currently, the list of available tables that you will see when executing an algorithm that needs one of them is restricted to tables coming from files in dBase (`.dbf`) or Comma-Separated Values (`.csv`) formats.
- 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.

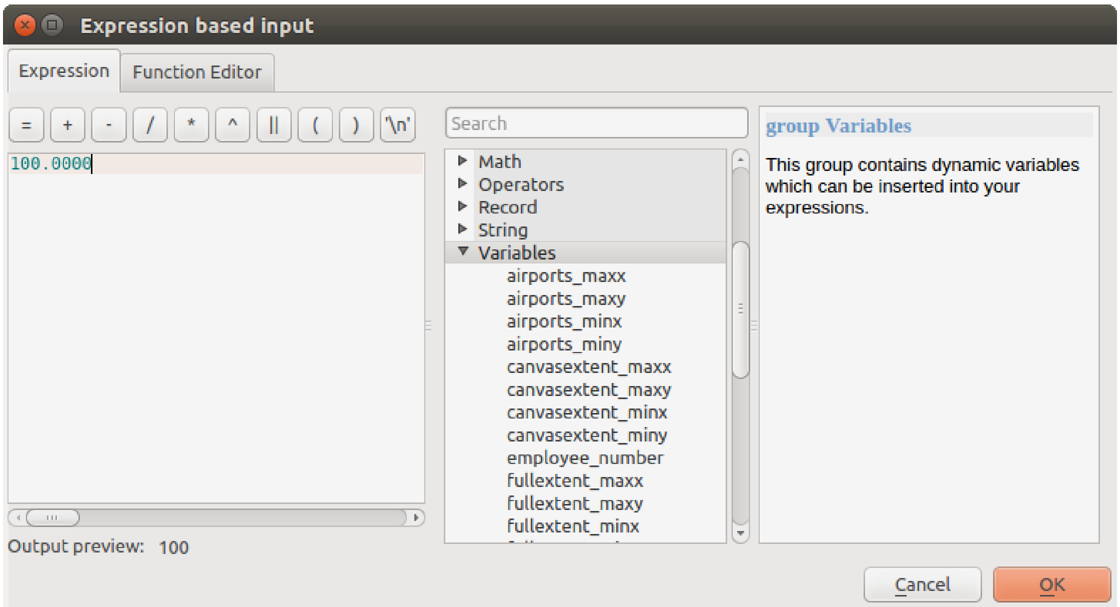


Abb. 22.10: Expression based input

- 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**, to be entered by four numbers representing its `xmin`, `xmax`, `ymin`, `ymax` limits. Clicking on the button on the right-hand side of the value selector, a pop-up menu will appear, giving you options:
  - to select the value from a layer or the current canvas extent;
  - or to define it by dragging directly onto the map canvas.

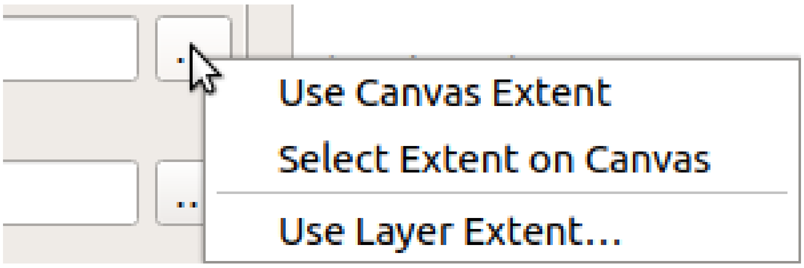


Abb. 22.11: Ausdehnung Auswahl

Wenn Sie die erste Option wählen, sehen Sie einen Dialog wie den nächsten.

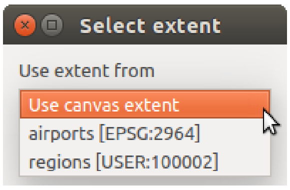


Abb. 22.12: Ausdehnung Liste

Wenn Sie den zweiten wählen, wird das Parameter-Fenster verschwinden, damit Sie durch Klicken und Ziehen im Kartenfenster einen Bereich definieren können. Wenn das Rechteck definiert ist, wird der Dialog mit den Werten wieder auftauchen.

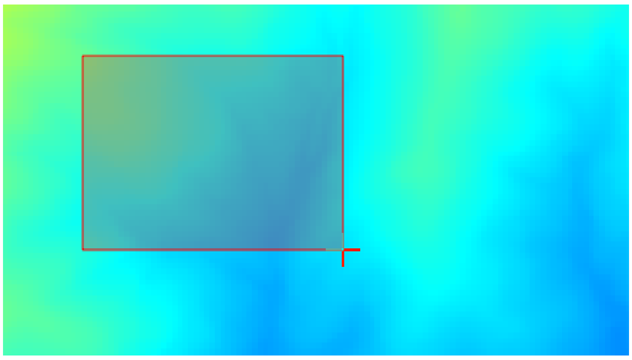


Abb. 22.13: Ausdehnung aufziehen

- 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.
- 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.

  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 parameter 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.

Along with the *Parameters* tab, there is another tab named *Log* (see figure below). Information provided by the algorithm during its execution is written in this tab, and allow you to track the execution and be aware and have more

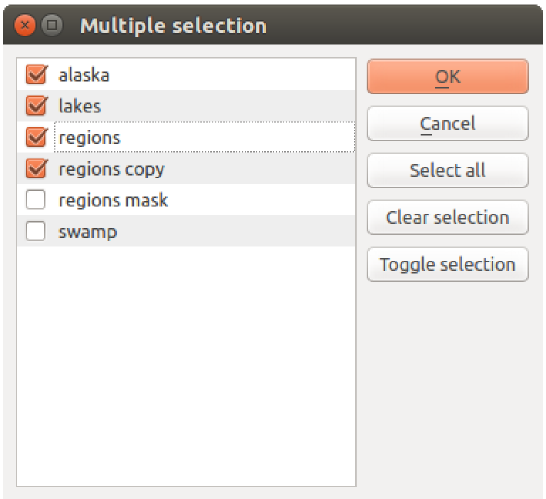


Abb. 22.14: Mehrfachauswahl

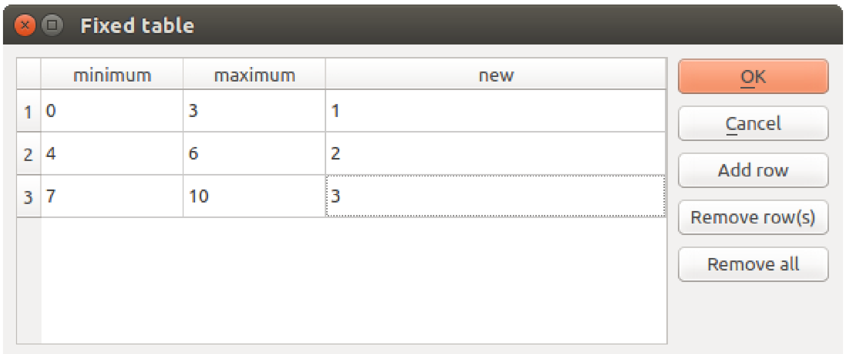


Abb. 22.15: Fixe Tabelle

details about the algorithm as it runs. Notice that not all algorithms write information to this tab, and many of them might run silently without producing any output other than the final files.

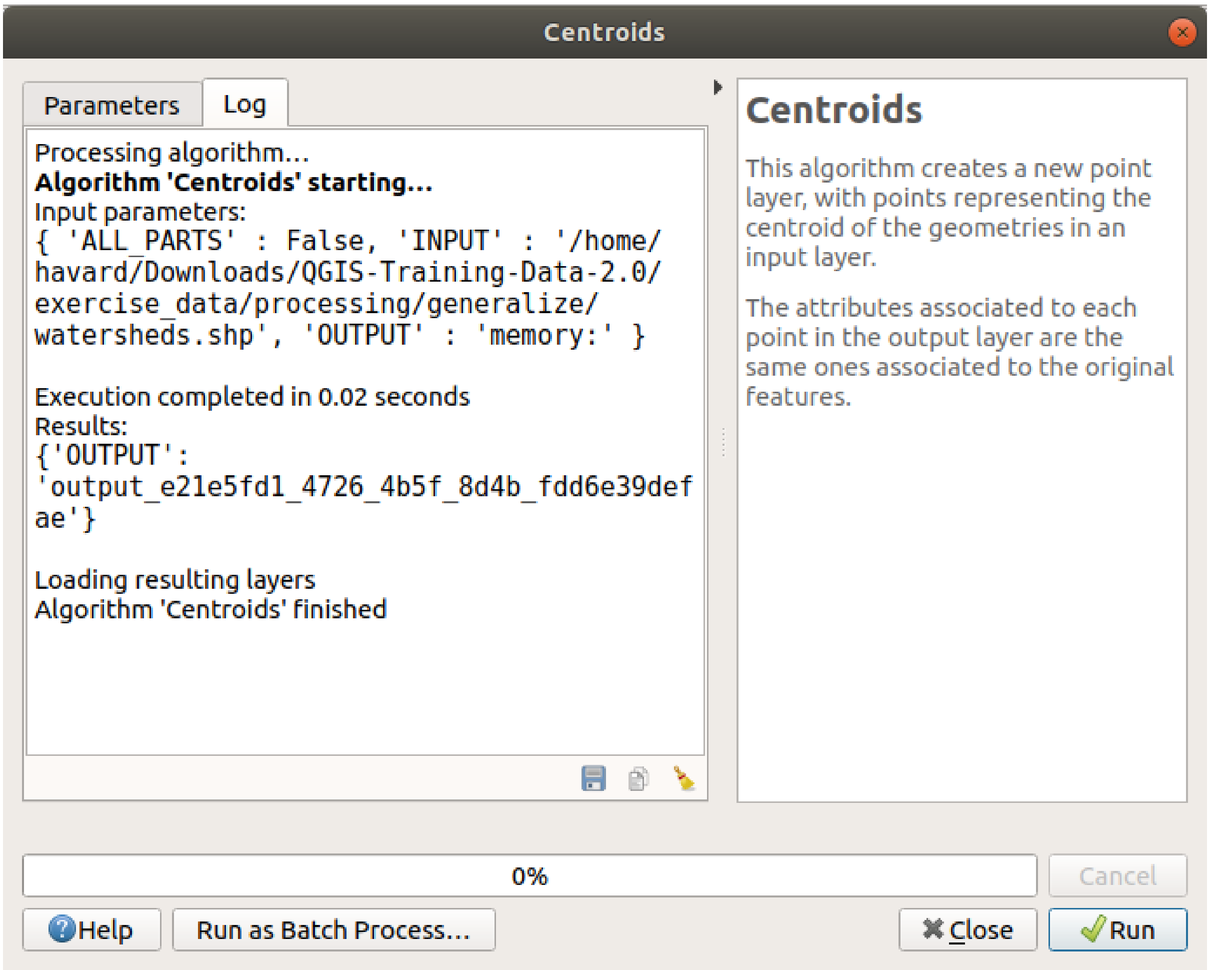


Abb. 22.16: Algorithm Dialog - Log

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.

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).

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.

22.3.2 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 |qgl 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).

22.4 Das Protokoll

22.4.1 Das Verarbeitung Protokoll

Every time you execute an algorithm, information about the process is stored in the history manager. The date and time of the execution are saved, along with the parameters used, making it is easy to track and control all the work that has been developed using the Processing framework, and to reproduce it.

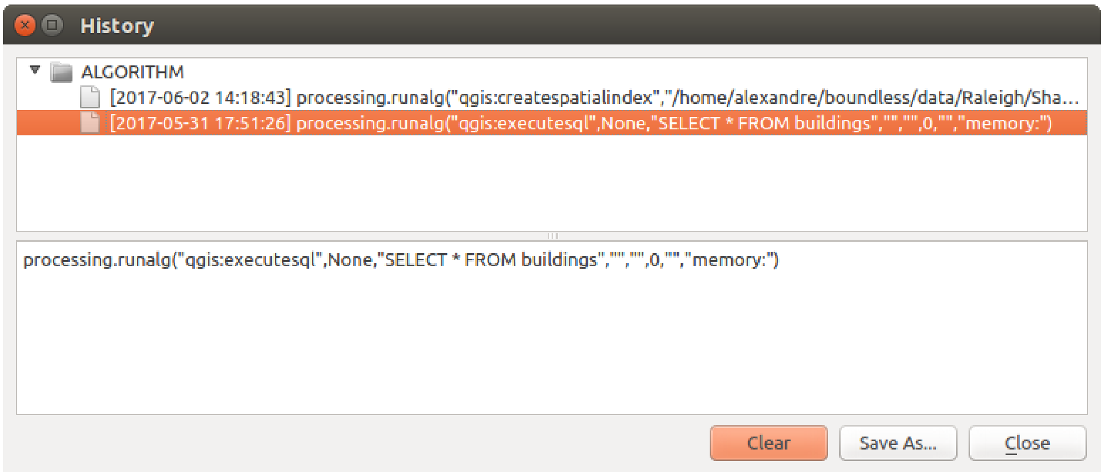


Abb. 22.17: History

Process information is kept as a command-line expression, even if the algorithm was launched from the toolbox. This makes it useful for those learning how to use the command-line interface, since they can call an algorithm using the toolbox and then check the history manager to see how it could be called from the command line.

Apart from browsing the entries in the registry, you can also re-execute processes by simply double-clicking on the entry. The algorithm dialog then opens with parameters already set, and you can change any of them to fit your needs and re-run the algorithm.

The *History* dialog also provides a convenient way to contribute to the consolidation of the testing infrastructure of QGIS Processing algorithms and scripts. When you right-click on an entry, you can *Create Test…* using the concerned algorithm and parameters, following instructions at https://github.com/qgis/QGIS/blob/release-3_10/python/plugins/processing/tests/README.md.

22.4.2 Das Verarbeitung Log

The history dialog only contains the execution calls, but not the information produced by the algorithm when executed. That information is written to the QGIS log (*View ▸ Panels ▸ Log Messages Panel*).

Third-party algorithms are usually executed by using their command-line interfaces, which communicate with the user via the console. Although that console is not shown, usually a full dump of it is written to the log each time you run one of those algorithms. To avoid cluttering the log with that information, you can disable it for each provider in the settings dialog.

Some algorithms, even if they can produce a result with the given input data, output comments or additional information to log when they detect potential problems with the data, in order to warn you. Make sure you check those messages in the log if you get unexpected results.

22.5 Die Grafische Modellierung

Der *Grafische Modellierer* ermöglicht es Ihnen, komplexe Modelle über eine einfache und leicht zu bedienende Schnittstelle zu erstellen. Wenn Sie mit einem GIS arbeiten, sind die meisten Analyseoperationen nicht isoliert, sondern Teil einer Kette von Operationen. Mit dem grafischen Modellierer kann diese Kette von Operationen in einen einzigen Prozess verpackt werden, so dass sie später bequem mit einem anderen Satz von Eingaben ausgeführt werden kann. Unabhängig davon, wie viele Schritte und verschiedene Algorithmen es umfasst, wird ein Modell als ein einziger Algorithmus ausgeführt, was Zeit und Mühe spart.

Der grafische Modellierer kann über das Menü Verarbeitung geöffnet werden (*Verarbeitung ▸ Grafische Modellierung*).

Der Modellierer hat eine Arbeitsfläche, auf der die Struktur des Modells und der Arbeitsablauf, den es repräsentiert, gezeigt werden. Der linke Teil des Fensters ist ein Bereich mit zwei Registerkarten, die zum Hinzufügen neuer Elemente zum Modell verwendet werden können.

Das Erstellen eines Modells geht in zwei Schritten:

1. *Definition von erforderlichen Eingaben*. Diese Eingaben werden dem Paramterfenster 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.

22.5.1 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:

- Authentifikationskonfiguration
- Boolean
- KBS
- Farbe
- Abstand
- Aufzählung
- Ausdruck
- Ausdehnung
- Feldabbildung

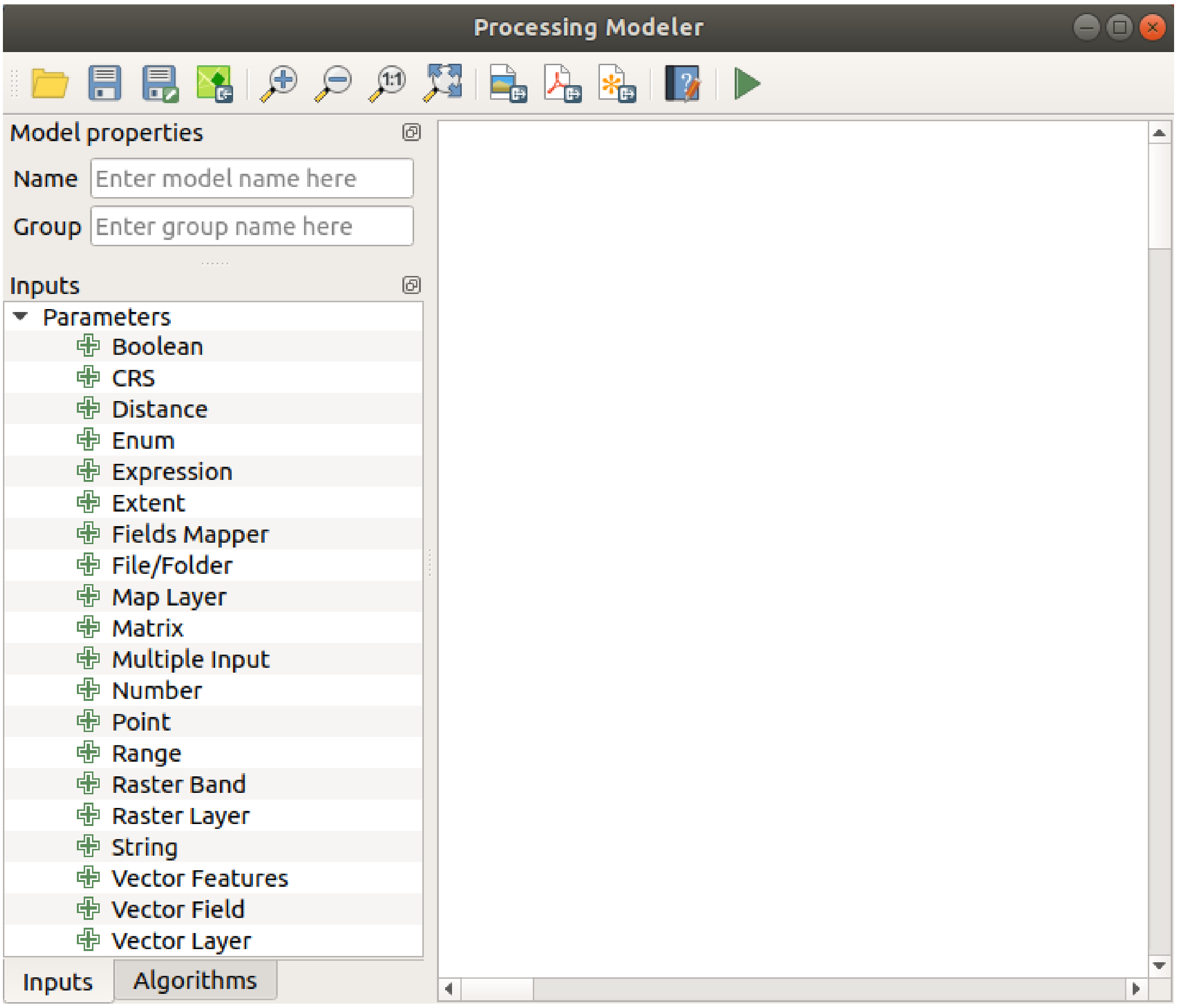


Abb. 22.18: Modellierung

- Datei/Ordner
- Kartenlayer
- Matrix
- Mehrfacheingabe
- Number
- Punkt
- Drucklayout
- Drucklayout Element
- Bereich
- Rasterkanal
- Rasterlayer
- Maßstab
- Zeichenkette
- Vektorobjekte
- Vektorfeld
- Vektorlayer

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.

Sie können Ihre Eingabe als zwingend für Ihr Modell definieren, indem Sie das Kontrollkästchen Zwingend ankreuzen. Durch Ankreuzen des Kontrollkästchens :sup: *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.

Sie können Eingaben auch hinzufügen, indem Sie den Eingabetyp aus der Liste an die gewünschten Stelle im Modellierungsfenster ziehen.

22.5.2 Definition des Arbeitsablaufs

Sobald die Eingaben definiert sind, ist es an der Zeit, die Algorithmen des Modells zu definieren. Die Algorithmen finden Sie auf der Registerkarte *Algorithmen*, ähnlich gruppiert wie in der Verarbeitungs-Werkzeugkiste.

Um einen Algorithmus zu einem Modell hinzuzufügen, doppelklicken Sie auf seinen Namen oder ziehen Sie ihn per Drag & Drop, genau wie bei Eingaben. Es erscheint ein Ausführungsdialogfeld mit einem ähnlichen Inhalt wie das Ausführungsfenster, das angezeigt wird, wenn der Algorithmus aus der Werkzeugkiste ausgeführt wird. Die nebenan gezeigten entsprechen dem QGIS-Algorithmus ‚Drapieren (Z-Wert von Raster übernehmen)' und dem QGIS-Algorithmus ‚Entlang Linie klettern'.

Wie Sie sehen können bestehen einige Unterschiede. Anstelle des Dateiausgabefensters, das für die Einstellung des Dateipfades für Ausgabelayer und -tabellen verwendet wurde, wird hier eine einfaches Textfenster verwendet. Wenn der vom Algorithmus erstellte Layer nur ein vorläufiges Ergebnis ist, das als Eingabe für einen anderen Algorithmus verwendet wird und nicht als endgültiges Ergebnis vorgehalten werden soll, bearbeiten Sie dieses Textfenster nicht. Wenn Sie etwas eingeben heißt das, dass das Ergebnis endgültig ist und dass der Text den Sie vergeben die Beschreibung für die Ausgabe, die die Ausgabe ist, die der Benutzer sieht wenn das Modell ausgeführt wird, ist.

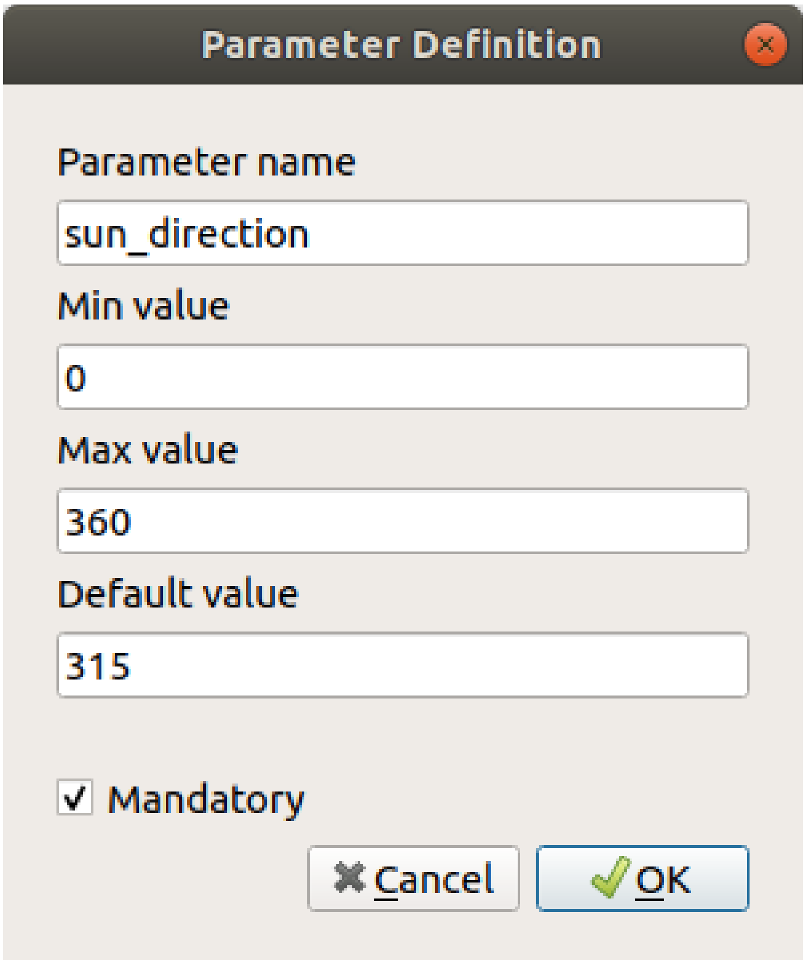


Abb. 22.19: Definition der Modellparameter

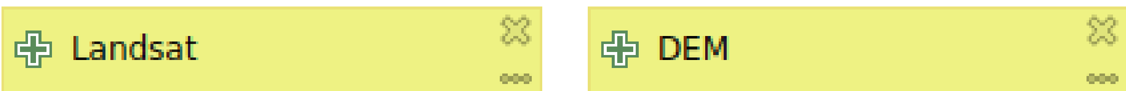


Abb. 22.20: Modellparameter

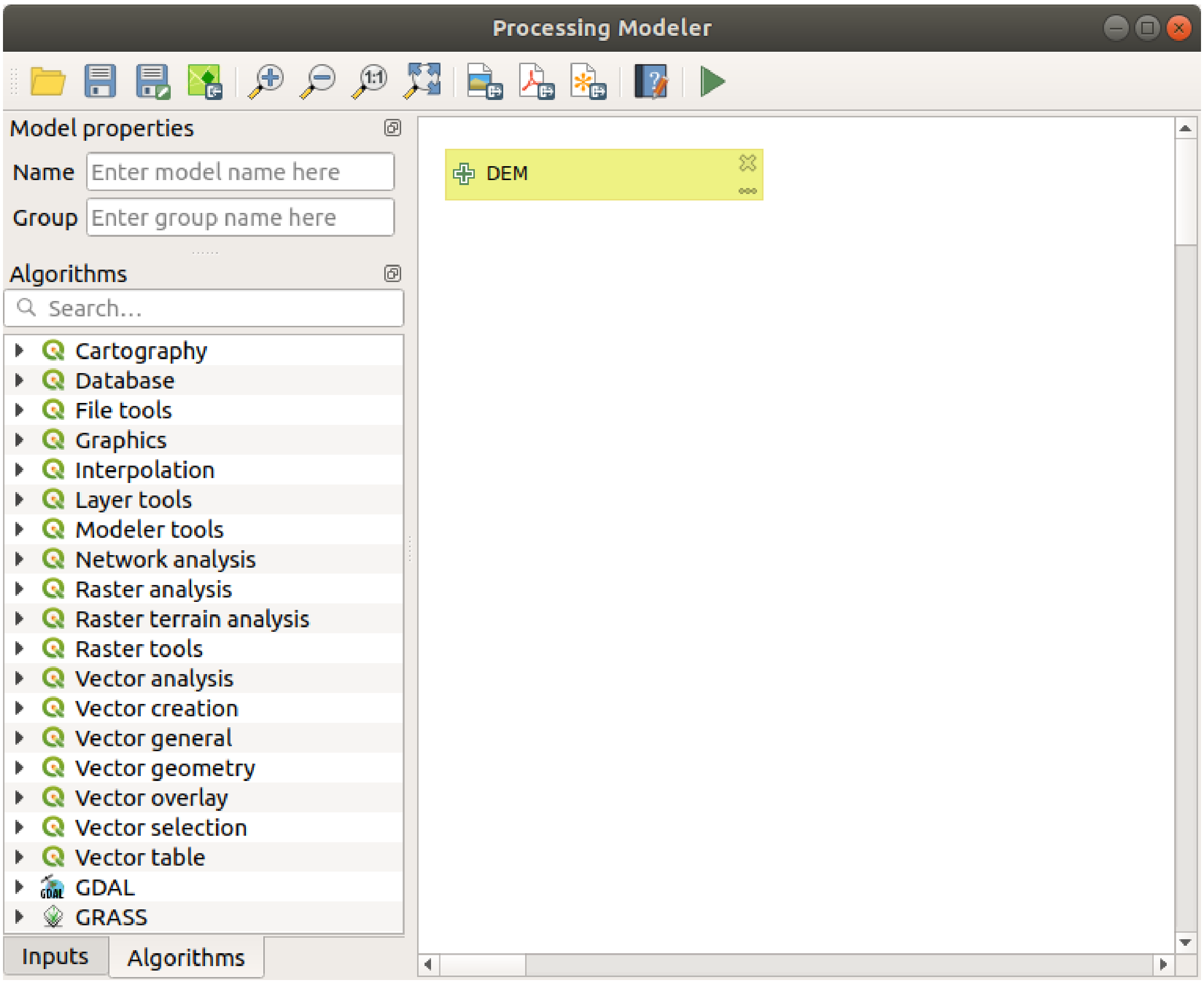


Abb. 22.21: Modell-Eingaben

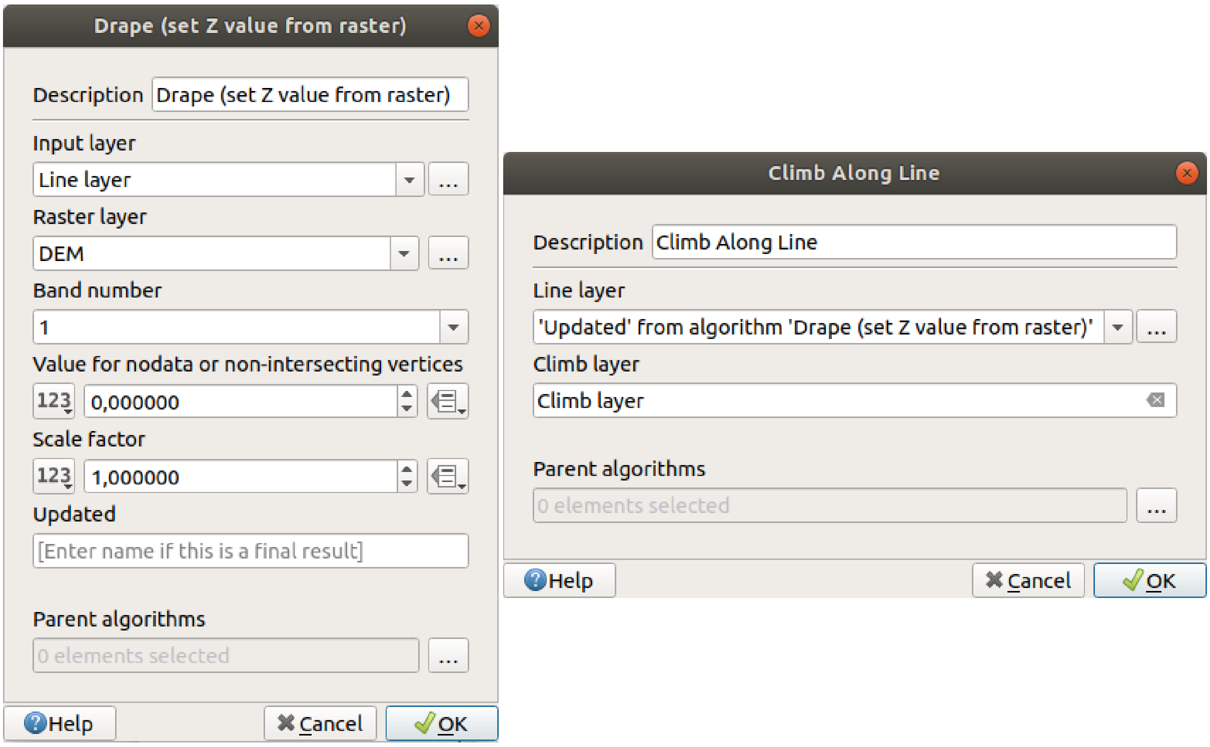


Abb. 22.22: Modell-Algorithmus-Parameter

Das Auswählen des Wertes für jeden Parameter ist ebenfalls etwas schwierig, da es wichtige Unterschiede zwischen dem Kontext der Modellierung und der der Werkzeugkiste gibt. Wollen wir sehen wie man die Werte für jeden Typ von Parameter vorstellen.

- Layer (Raster und Vektor) und Tabellen. Sie werden aus einer Liste ausgewählt, aber in diesem Fall sind die möglichen Werte nicht die Layer oder Tabellen, die momentan in QGIS geladen sind, sondern die Liste der Modelleingaben des entsprechenden Typs oder andere Layer oder Tabellen, die durch Algorithmen bereits erzeugt und zum Modell hinzugefügt wurden.
- Numerische Werte. Feste Werte können direkt in das Textfeld eingegeben werden. Durch Klicken auf die Schaltfläche neben dem Textfeld können Ausdrücke eingegeben werden. Zu den verfügbaren Variablen für Ausdrücke gehören numerische Eingaben des Modells, Ausgaben von Modellalgorithmen und auch statistische Werte aus verfügbaren Ebenen innerhalb des Modells.
- Zeichenkette. Feste Zeichenfolgen können in das entsprechende Textfeld eingegeben werden. Durch Klicken auf die Schaltfläche neben dem Textfeld können Ausdrücke wie bei numerischen Werten eingegeben werden.
- Vektorfeld. Die Feldwerte eines Vektorlayers können bei der Modellierung nicht bekannt sein, da sie bei jeder Ausführung des Modells von der Auswahl des Benutzers abhängen. Um den Wert für diesen Parameter festzulegen, geben Sie den Namen eines Feldes direkt in das Textfeld ein oder verwenden Sie die Liste, um ein Tabellenfeld auszuwählen. Die Gültigkeit des ausgewählten Feldes wird zur Laufzeit überprüft.

Sie werden in jedem Fall einen zusätzlichen Paramter genannt *Parent algorithms*, der nicht zugänglich ist wenn Sie den Algorithmus aus der Werkzeugkiste aufrufen, finden. Dieser Parameter ermöglicht es Ihnen die Reihenfolge in welcher die Algorithmen ausgeführt werden, indem ein Algorithmus ausdrücklich als Eltern des aktuellen definiert werden, festzulegen. Dies bewirkt, dass der Elternalgorithmus vor dem aktuellen ausgeführt wird.

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 *Parent algorithms* 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.

Elemente können an eine andere Position auf der Oberfläche gezogen werden. Dies ist nützlich, um die Struktur des Modells klarer und intuitiver zu gestalten. Verknüpfungen zwischen Elementen werden automatisch aktualisiert. Sie können mit dem Mausrad hinein- und herauszoomen.

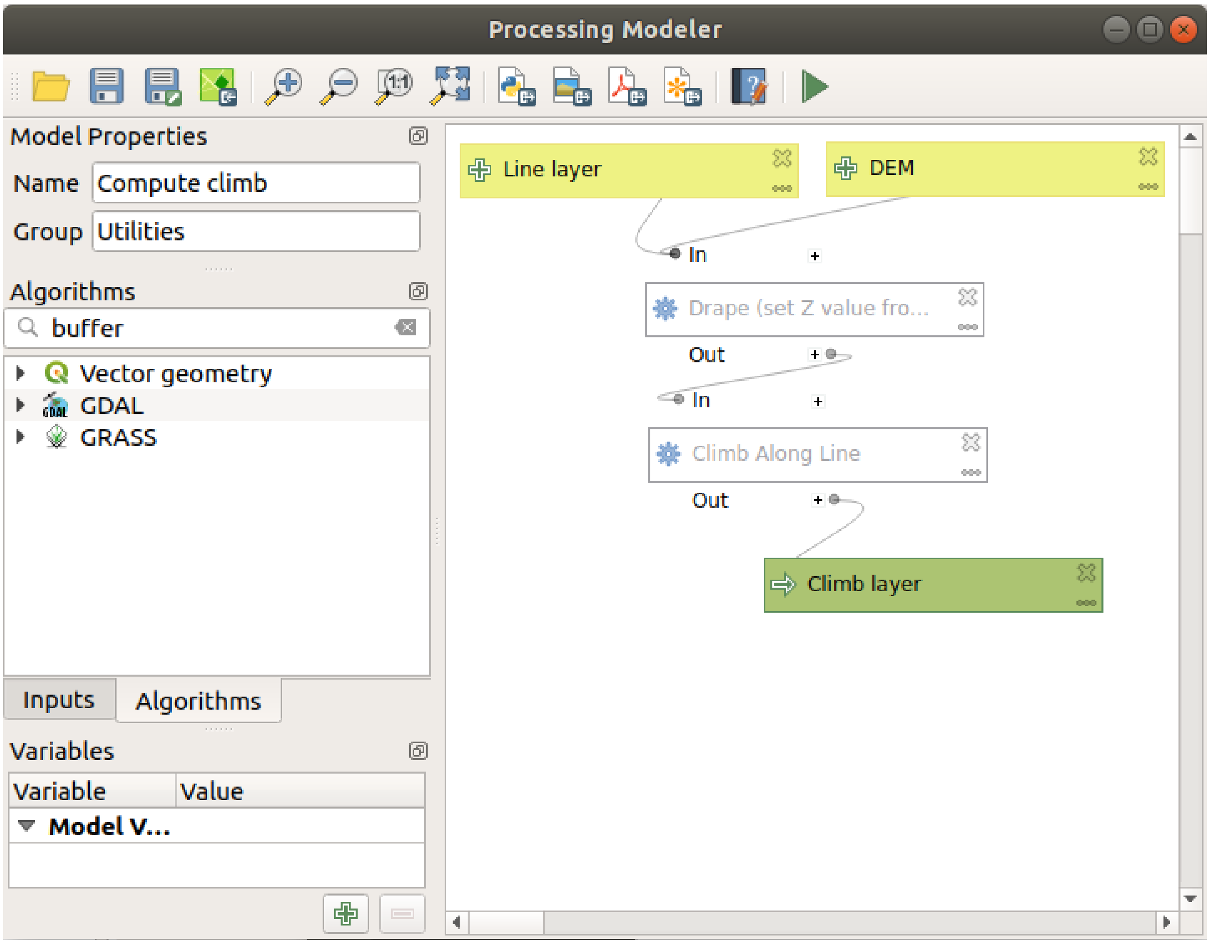


Abb. 22.23: Ein komplettes Modell

Sie können Ihren Algorithmus jederzeit ausführen, indem Sie auf die Schaltfläche *Run* klicken. Um den Algorithmus aus der Toolbox heraus zu verwenden, muss er gespeichert und der Modellierdialog geschlossen werden, damit die Toolbox ihren Inhalt aktualisieren kann.

22.5.3 Speichern und laden von Modellen

Verwenden Sie die Schaltfläche *Speichern*, um das aktuelle Modell zu speichern, und die Schaltfläche *Öffnen*, um ein zuvor gespeichertes Modell zu öffnen. Modelle werden mit der Erweiterung `.model3` gespeichert. Wenn das Modell bereits vom Modellierer-Fenster aus gespeichert wurde, werden Sie nicht zur Eingabe eines Dateinamens aufgefordert. Da mit dem Modell bereits eine Datei verknüpft ist, wird diese Datei für spätere Speicherungen verwendet.

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.

Projektmodelle sind im Menü *Projektmodelle* der Toolbox verfügbar.

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 Bild, PDF oder SVG

Ein Modell kann auch als Bild, SVG oder PDF (zur Darstellung) exportiert werden.

22.5.4 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:

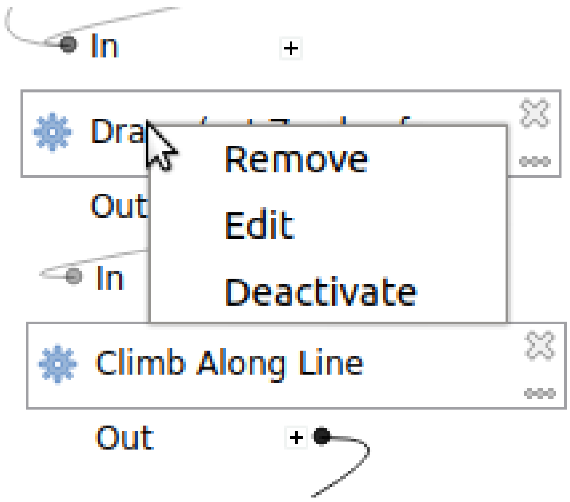


Abb. 22.24: Rechtsklick auf den Modeler

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:

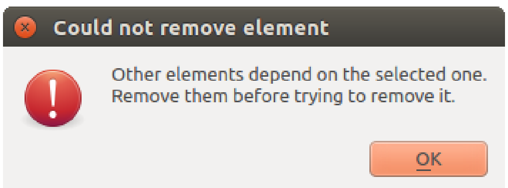


Abb. 22.25: Kann Algorithmus nicht entfernen

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.

Ein Modell kann teilweise ausgeführt werden, indem einige der Algorithmen deaktiviert werden. Um dies zu machen, rechtsklicken Sie auf ein Algorithmus Element und wählen Sie *Deaktivieren*. Der ausgewählte Algorithmus und alle diejenigen, die in dem Modell von ihm abhängen werden in grau angezeigt und nicht als Teil des Modells ausgeführt.

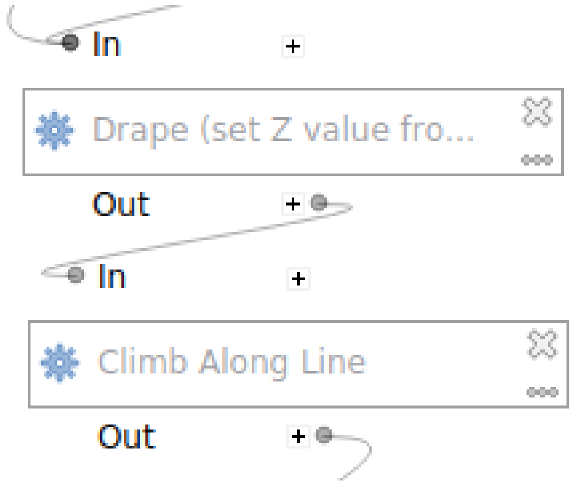


Abb. 22.26: Modell mit deaktivierten Algorithmen

Wenn Sie mit der rechten Maustaste auf einen Algorithmus klicken, der nicht aktiv ist, sehen Sie die Menüoption :guilabel: *Aktivieren*, mit der Sie den Algorithmus reaktivieren können.

22.5.5 Editieren der Modell-Hilfe Datei und der Metainformationen

Sie können Ihre Modelle im Modellierer selbst dokumentieren. Klicken Sie einfach auf die Schaltfläche *Modellhilfe bearbeiten*, und es erscheint ein Dialog wie der folgende.

Auf der rechten Seite finden Sie eine einfache HTML-Seite, die anhand der Eingabeparameter und Ausgaben des Algorithmus erstellt wurde, zusammen mit einigen zusätzlichen Einträgen wie eine allgemeine Beschreibung des Modells und seines Autors. Wenn Sie den Hilfeeditor das erste Mal öffnen, sind alle diese Beschreibungen leer, Sie können Sie aber mit Hilfe der Elemente auf der linken Seite des Dialogs bearbeiten. Wählen Sie ein Element im oberen Teil aus und schreiben Sie seine Beschreibung in das Textfeld unten.

Modell-Hilfe wird als Teil des Modells selbst gespeichert.

22.5.6 Exportieren eines Modells als Python Skript

Wie wir in einem späteren Kapitel sehen werden, können Verarbeitungsalgorithmen von der QGIS-Python-Konsole aufgerufen werden, und neue Verarbeitungsalgorithmen können mit Python erstellt werden. Ein schneller Weg, ein solches Python-Skript zu erstellen, ist die Erstellung eines Modells und der anschließende Export als Python-Datei.

Klicken Sie dazu mit der rechten Maustaste auf den Namen des Modells in der Processing Toolbox und wählen Sie *Modell als Python-Algorithmus exportieren…*.

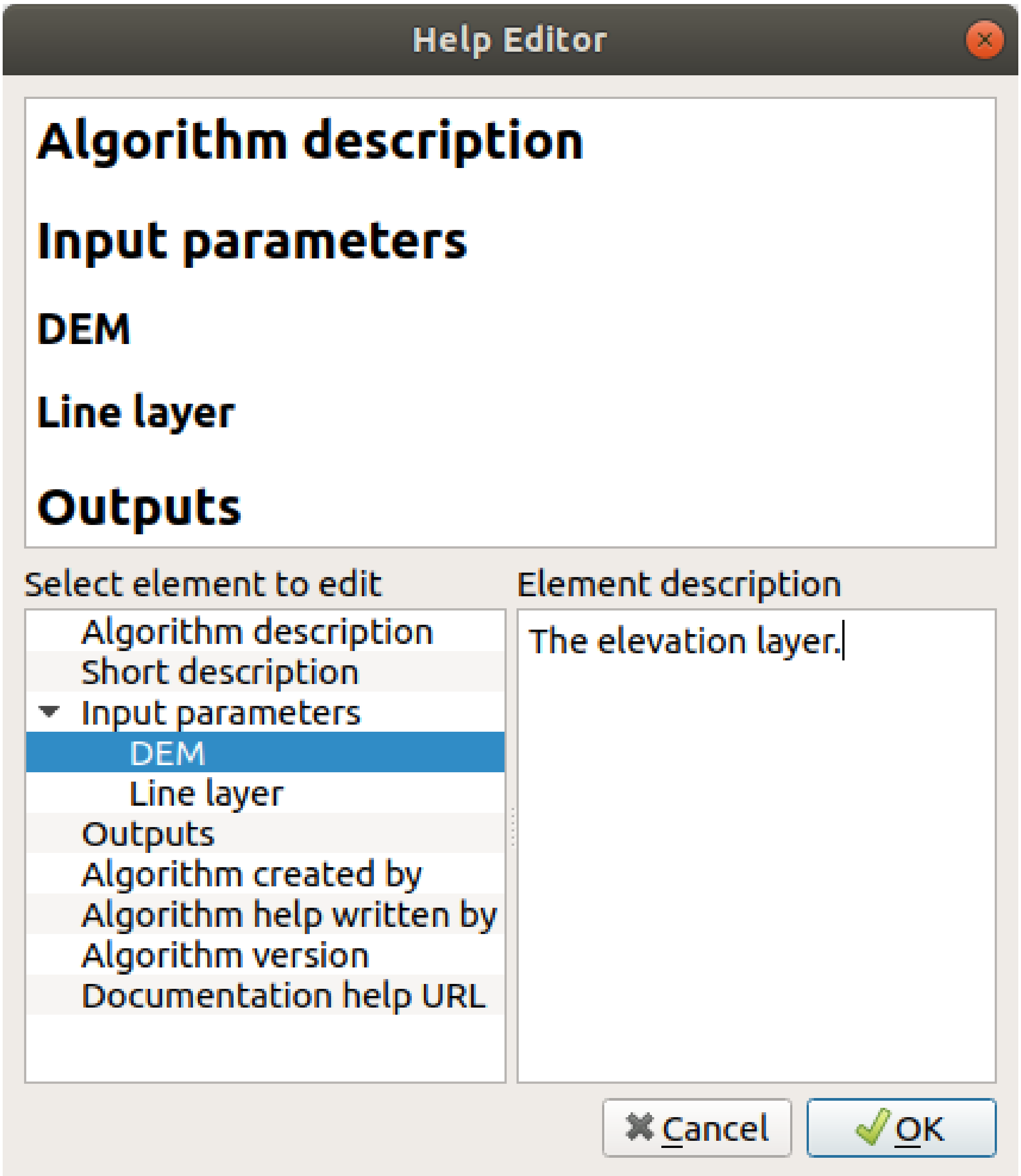


Abb. 22.27: Hilfe bearbeiten

22.5.7 Zu den verfügbaren Algorithmen

Möglicherweise stellen Sie fest, dass einige Algorithmen, die über die Toolbox ausgeführt werden können, beim Entwurf eines Modells nicht in der Liste der verfügbaren Algorithmen erscheinen. Um in ein Modell aufgenommen zu werden, muss ein Algorithmus die richtige Semantik haben. Wenn ein Algorithmus nicht über eine so gut definierte Semantik verfügt (z. B. wenn die Anzahl der Ausgabelayer nicht im Voraus bekannt ist), kann er nicht in einem Modell verwendet werden, und er erscheint nicht in der Liste der Algorithmen, die Sie im Modellierungsdialog finden.

22.6 Die Batch Processing Schnittstelle

22.6.1 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*.

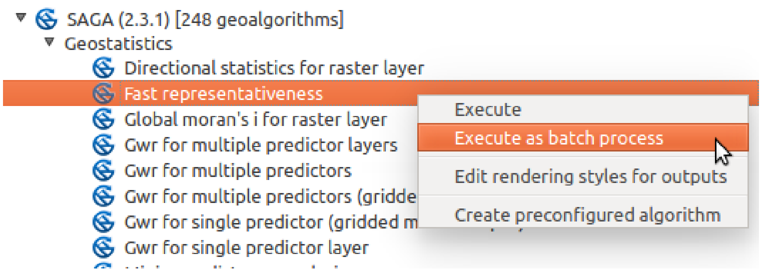


Abb. 22.28: Batch Processing from right-click

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.

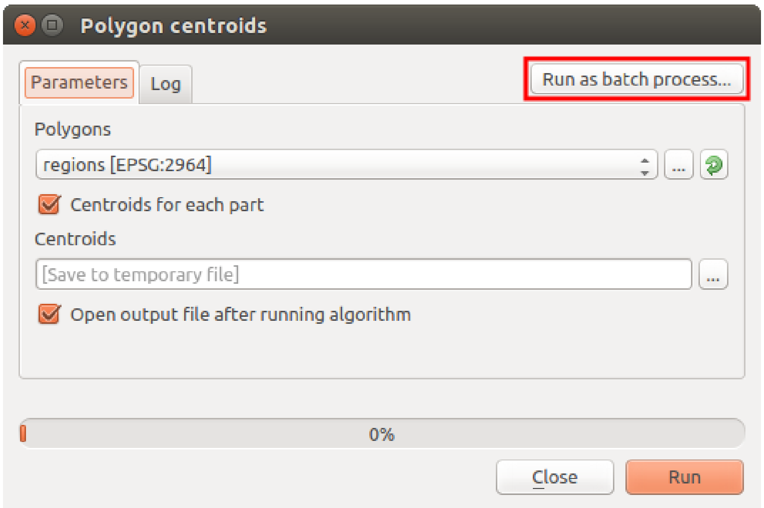


Abb. 22.29: Batch-Prozess vom Algorithmus Dialog

22.6.2 Die Parameter-Tabelle

Ausführen eines Batch-Prozesses ist vergleichbar mit der Durchführung der einmaligen Ausführung eines Algorithmus. Parameter-Werte müssen definiert werden, aber in diesem Fall müssen wir nicht nur einen einzelnen Wert für jeden Parameter angeben, sondern eine Reihe von ihnen, einen für jedes Mal, wenn der Algorithmus ausgeführt werden soll. Werte werden mit Hilfe einer Tabelle wie in der nächsten Abbildung zu sehen übergeben.

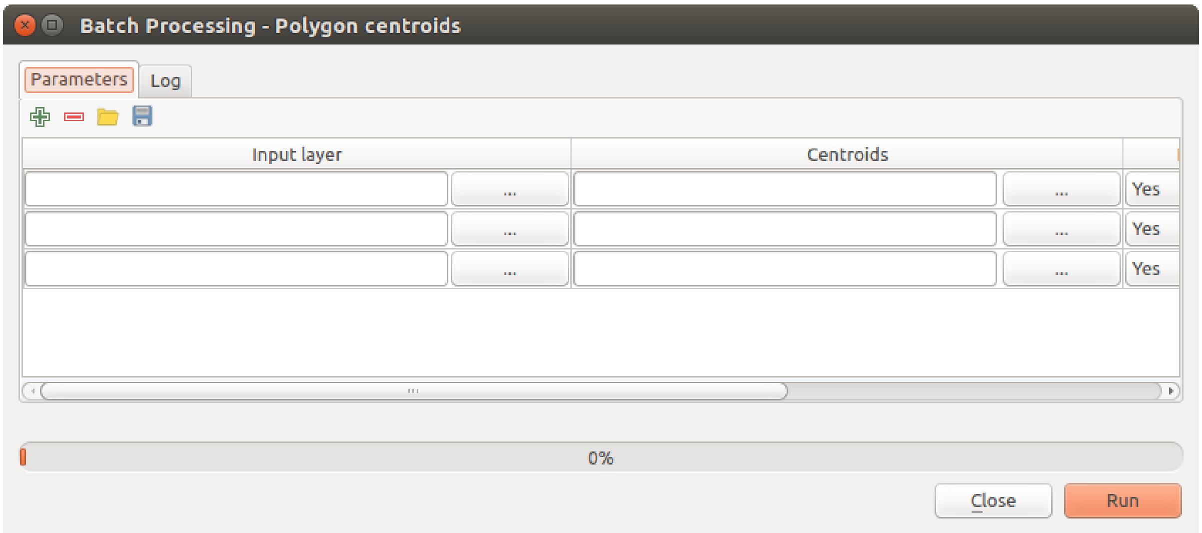


Abb. 22.30: Batch Processing

Jede Zeile dieser Tabelle stellt eine einzelne Ausführung des Algorithmus dar, und jede Zelle enthält den Wert eines der Parameter. Es ist vergleichbar mit dem Parameter-Dialog, den Sie sehen, wenn Sie die Ausführung eines Algorithmus aus der Toolbox starten, aber mit einer anderen Anordnung.

Standardmäßig enthält die Tabelle nur zwei Zeilen. Sie können weitere hinzufügen oder entfernen mit den Tasten am unteren Teil des Fensters.

Sobald die Größe der Tabelle gesetzt ist, muss sie mit den gewünschten Werten gefüllt werden.

22.6.3 Füllen der Parameter-Tabelle

Bei den meisten Parametern ist das Setzen des Wertes trivial. Geben Sie einfach den Wer ein oder wählen Sie diesen aus der Liste zur Verfügung stehender Optionen, abhängig vom Parametertyp, aus.

Filenames for input data objects are introduced directly typing or, more conveniently, clicking on the … button on the right hand of the cell, which will show a context menu with two option: one for selecting from the layers currently opened and another to select from the filesystem. This second option, when selected, shows a typical file chooser dialog. Multiple files can be selected at once. If the input parameter represents a single data object and several files are selected, each one of them will be put in a separate row, adding new ones if needed. If the parameter represents a multiple input, all the selected files will be added to a single cell, separated by semicolons (`;`).

Layer-Kennungen können direkt im Parametertextfeld eingegeben werden. Sie können den vollständigen Pfad zu einer Datei oder den Namen einer Ebene geben, die derzeit in der aktuellen QGIS Projekt geladen wird. Der Name des Layers wird automatisch in seinem Quell-Pfad aufgelöst werden. Beachten Sie, dass, wenn mehrere Layer den gleichen Namen haben, könnte dies zu unerwarteten Ergebnissen führen aufgrund von Mehrdeutigkeit.

Ausgabedaten Objekte werden immer in einer Datei gespeichert und im Gegensatz zur Ausführung eines Algorithmus aus der Toolbox, ist das vorübergehende Speichern als temporäre Datei oder Datenbank nicht gestattet. Sie können den Namen direkt eingeben oder im Dateiauswahldialog, wenn Sie auf den zugehörigen Button klicken.

Sobald Sie die Datei auswählen, erscheint ein neuer Dialog, der Autovervollständigung von anderen Zellen in derselben Spalte (gleiche Parameter) ermöglicht.

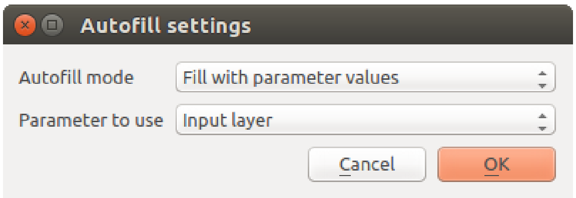


Abb. 22.31: Batch Processing Save

Wenn der Standardwert (‚Do not autocomplete') gewählt ist, wird einfach der gewählte Dateiname in die ausgewählte Zelle der Parameters Tabelle eingegeben. Wenn eine der anderen Optionen gewählt ist, werden alle Zellen unter der ausgewählten automatisch anhand von definierten Kriterien ausgefüllt. So ist es erheblich einfacher die Tabelle auszufüllen und der Stapelprozeß kann mit weniger Aufwand definiert werden.

Automatische Füllung kann durch einfaches Hinzufügen von korrelativen Zahlen zum ausgewählten Dateipfad oder Anhängen des Wertes eines anderen Feldes zu derselben Zeile erreicht werden. Dies ist besonders nützlich für die Benennung von Ausgabedaten-Objekte entsprechend der Eingänge.

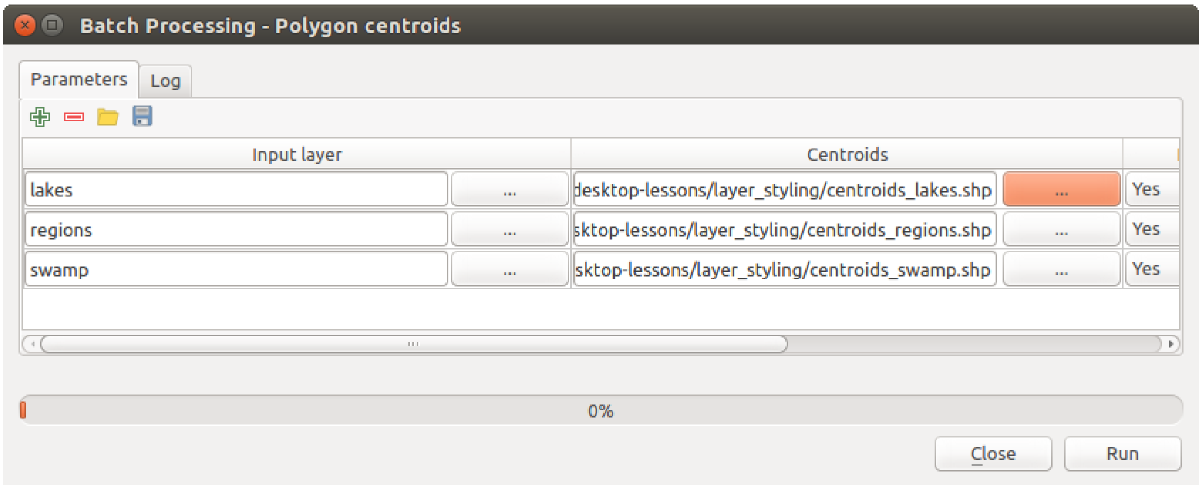


Abb. 22.32: Batch Processing Dateipfad

22.6.4 Ausführen eines Batch-Prozesses

To execute the batch process once you have introduced all the necessary values, just click on *OK*. Progress of the global batch task will be shown in the progress bar in the lower part of the dialog.

22.7 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 Kommandozeilenssschnittstelle 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 Scripte.

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.

22.7.1 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:buffervectors ---------> 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.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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.

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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
   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

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.

```
>>> processing.run("native:buffer", {'INPUT': '/data/lines.shp',
              'DISTANCE': 100.0,
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
              'SEGMENTS': 10,
              'DISSOLVE': True,
              'END_CAP_STYLE': 0,
              'JOIN_STYLE': 0,
              'MITER_LIMIT': 10,
              '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:

```
>>> myresult = processing.run("native:buffer", {'INPUT': '/data/lines.shp',
              'DISTANCE': 100.0,
              'SEGMENTS': 10,
              'DISSOLVE': True,
              'END_CAP_STYLE': 0,
              'JOIN_STYLE': 0,
              'MITER_LIMIT': 10,
              'OUTPUT': '/data/buffers.shp'})
>>> myresult['OUTPUT']
/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:

```
>>> my_dialog = processing.createAlgorithmDialog("native:buffer", {
              'INPUT': '/data/lines.shp',
              'DISTANCE': 100.0,
              'SEGMENTS': 10,
              'DISSOLVE': True,
              'END_CAP_STYLE': 0,
              'JOIN_STYLE': 0,
              'MITER_LIMIT': 10,
              'OUTPUT': '/data/buffers.shp'})
>>> my_dialog.show()
```

The `execAlgorithmDialog` method opens the dialog immediately:

```
>>> processing.execAlgorithmDialog("native:buffer", {
              'INPUT': '/data/lines.shp',
              'DISTANCE': 100.0,
              'SEGMENTS': 10,
              'DISSOLVE': True,
              'END_CAP_STYLE': 0,
              'JOIN_STYLE': 0,
              'MITER_LIMIT': 10,
              'OUTPUT': '/data/buffers.shp'})
```

22.7.2 Scripte 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.

```
from qgis.core import (QgsProcessingAlgorithm,
       QgsProcessingParameterNumber,
       QgsProcessingParameterFeatureSource,
       QgsProcessingParameterFeatureSink)

from qgis import processing

class algTest(QgsProcessingAlgorithm):
    INPUT_BUFFERDIST = 'BUFFERDIST'
    OUTPUT_BUFFER = 'OUTPUT_BUFFER'
    INPUT_VECTOR = 'INPUT_VECTOR'

    def __init__(self):
        super().__init__()

    def name(self):
        return "algTest"

    def displayName(self):
        return "algTest script"

    def createInstance(self):
        return type(self)()

    def initAlgorithm(self, config=None):
        self.addParameter(QgsProcessingParameterFeatureSource(
            self.INPUT_VECTOR, "Input vector"))
        self.addParameter(QgsProcessingParameterNumber(
            self.INPUT_BUFFERDIST, "Buffer distance",
            QgsProcessingParameterNumber.Double,
            100.0))
        self.addParameter(QgsProcessingParameterFeatureSink(
            self.OUTPUT_BUFFER, "Output buffer"))

    def processAlgorithm(self, parameters, context, feedback):
        #DO SOMETHING
        algresult = processing.run("native:smoothgeometry",
            {'INPUT': parameters[self.INPUT_VECTOR],
             'ITERATIONS':2,
             'OFFSET':0.25,
             'MAX_ANGLE':180,
             'OUTPUT': 'memory:'},
            context=context, feedback=feedback, is_child_algorithm=True)
        smoothed = algresult['OUTPUT']
        algresult = processing.run('native:buffer',
            {'INPUT': smoothed,
            'DISTANCE': parameters[self.INPUT_BUFFERDIST],
            'SEGMENTS': 5,
            'END_CAP_STYLE': 0,
            'JOIN_STYLE': 0,
            'MITER_LIMIT': 10,
            'DISSOLVE': True,
            'OUTPUT': parameters[self.OUTPUT_BUFFER]},
            context=context, feedback=feedback, is_child_algorithm=True)
        buffered = algresult['OUTPUT']
        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 define a dummy function for the `onFinish` parameter for `run`. This is the `no_post_process` function. 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:

- `QgsProcessingParameterAuthConfig`
- `QgsProcessingParameterBand`
- `QgsProcessingParameterBoolean`
- `QgsProcessingParameterColor`
- `QgsProcessingParameterCrs`
- `QgsProcessingParameterDistance`
- `QgsProcessingParameterEnum`
- `QgsProcessingParameterExpression`
- `QgsProcessingParameterExtent`
- `QgsProcessingParameterFeatureSink`
- `QgsProcessingParameterFeatureSource`
- `QgsProcessingParameterField`
- `QgsProcessingParameterFile`
- `QgsProcessingParameterFileDestination`
- `QgsProcessingParameterFolderDestination`
- `QgsProcessingParameterLayout`
- `QgsProcessingParameterLayoutItem`
- `QgsProcessingParameterMapLayer`
- `QgsProcessingParameterMatrix`
- `QgsProcessingParameterMeshLayer`
- `QgsProcessingParameterMultipleLayers`
- `QgsProcessingParameterNumber`
- `QgsProcessingParameterPoint`
- `QgsProcessingParameterRange`
- `QgsProcessingParameterRasterDestination`
- `QgsProcessingParameterRasterLayer`
- `QgsProcessingParameterScale`
- `QgsProcessingParameterString`
- `QgsProcessingParameterVectorDestination`

- QgsProcessingParameterVectorLayer

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 Scripte 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.

22.7.3 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.

22.8 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*.

22.8.1 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

```
from qgis.PyQt.QtCore import QCoreApplication
from qgis.core import (QgsProcessing,
                       QgsProcessingAlgorithm,
                       QgsProcessingException,
                       QgsProcessingOutputNumber,
                       QgsProcessingParameterDistance,
                       QgsProcessingParameterFeatureSource,
                       QgsProcessingParameterVectorDestination,
                       QgsProcessingParameterRasterDestination)
from qgis import processing


class ExampleProcessingAlgorithm(QgsProcessingAlgorithm):
    """
    This is an example algorithm that takes a vector layer,
    creates some new layers and returns some results.
    """

    def tr(self, string):
        """
        Returns a translatable string with the self.tr() function.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        """
        return QCoreApplication.translate('Processing', string)

    def createInstance(self):
        # Must return a new copy of your algorithm.
        return ExampleProcessingAlgorithm()

    def name(self):
        """
        Returns the unique algorithm name.
        """
        return 'bufferrasterextend'

    def displayName(self):
        """
        Returns the translated algorithm name.
        """
        return self.tr('Buffer and export to raster (extend)')

    def group(self):
        """
        Returns the name of the group this algorithm belongs to.
        """
        return self.tr('Example scripts')

    def groupId(self):
        """
        Returns the unique ID of the group this algorithm belongs
        to.
        """
        return 'examplescripts'

    def shortHelpString(self):
        """
        Returns a localised short help string for the algorithm.
        """
        return self.tr('Example algorithm short description')

    def initAlgorithm(self, config=None):
        """
        Here we define the inputs and outputs of the algorithm.
        """
        # 'INPUT' is the recommended name for the main input
        # parameter.
        self.addParameter(
            QgsProcessingParameterFeatureSource(
                'INPUT',
                self.tr('Input vector layer'),
                types=[QgsProcessing.TypeVectorAnyGeometry]
            )
        )
        self.addParameter(
            QgsProcessingParameterVectorDestination(
                'BUFFER_OUTPUT',
                self.tr('Buffer output'),
            )
        )
        # 'OUTPUT' is the recommended name for the main output
        # parameter.
        self.addParameter(
            QgsProcessingParameterRasterDestination(
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
                'OUTPUT',
                self.tr('Raster output')
            )
        )
        self.addParameter(
            QgsProcessingParameterDistance(
                'BUFFERDIST',
                self.tr('BUFFERDIST'),
                defaultValue = 1.0,
                # Make distance units match the INPUT layer units:
                parentParameterName='INPUT'
            )
        )
        self.addParameter(
            QgsProcessingParameterDistance(
                'CELLSIZE',
                self.tr('CELLSIZE'),
                defaultValue = 10.0,
                parentParameterName='INPUT'
            )
        )
        self.addOutput(
            QgsProcessingOutputNumber(
                'NUMBEROFFEATURES',
                self.tr('Number of features processed')
            )
        )

    def processAlgorithm(self, parameters, context, feedback):
        """
        Here is where the processing itself takes place.
        """
        # First, we get the count of features from the INPUT layer.
        # This layer is defined as a QgsProcessingParameterFeatureSource
        # parameter, so it is retrieved by calling
        # self.parameterAsSource.
        input_featuresource = self.parameterAsSource(parameters,
                                                     'INPUT',
                                                     context)
        numfeatures = input_featuresource.featureCount()

        # Retrieve the buffer distance and raster cell size numeric
        # values. Since these are numeric values, they are retrieved
        # using self.parameterAsDouble.
        bufferdist = self.parameterAsDouble(parameters, 'BUFFERDIST',
                                            context)
        rastercellsize = self.parameterAsDouble(parameters, 'CELLSIZE',
                                                context)
        if feedback.isCanceled():
            return {}
        buffer_result = processing.run(
            'native:buffer',
            {
                # Here we pass on the original parameter values of INPUT
                # and BUFFER_OUTPUT to the buffer algorithm.
                'INPUT': parameters['INPUT'],
                'OUTPUT': parameters['BUFFER_OUTPUT'],
                'DISTANCE': bufferdist,
                'SEGMENTS': 10,
                'DISSOLVE': True,
                'END_CAP_STYLE': 0,
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
                'JOIN_STYLE': 0,
                'MITER_LIMIT': 10
            },
            # Because the buffer algorithm is being run as a step in
            # another larger algorithm, the is_child_algorithm option
            # should be set to True
            is_child_algorithm=True,
            #
            # It's important to pass on the context and feedback objects to
            # child algorithms, so that they can properly give feedback to
            # users and handle cancelation requests.
            context=context,
            feedback=feedback)

        # Check for cancelation
        if feedback.isCanceled():
            return {}

        # Run the separate rasterization algorithm using the buffer result
        # as an input.
        rasterized_result = processing.run(
            'qgis:rasterize',
            {
                # Here we pass the 'OUTPUT' value from the buffer's result
                # dictionary off to the rasterize child algorithm.
                'LAYER': buffer_result['OUTPUT'],
                'EXTENT': buffer_result['OUTPUT'],
                'MAP_UNITS_PER_PIXEL': rastercellsize,
                # Use the original parameter value.
                'OUTPUT': parameters['OUTPUT']
            },
            is_child_algorithm=True,
            context=context,
            feedback=feedback)

        if feedback.isCanceled():
            return {}

        # Return the results
        return {'OUTPUT': rasterized_result['OUTPUT'],
                'BUFFER_OUTPUT': buffer_result['OUTPUT'],
                '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.

22.8.2 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

```
from qgis import processing
from qgis.processing import alg
from qgis.core import QgsProject

@alg(name='bufferrasteralg', label='Buffer and export to raster (alg)',
     group='examplescripts', group_label='Example scripts')
# 'INPUT' is the recommended name for the main input parameter
@alg.input(type=alg.SOURCE, name='INPUT', label='Input vector layer')
# 'OUTPUT' is the recommended name for the main output parameter
@alg.input(type=alg.RASTER_LAYER_DEST, name='OUTPUT',
           label='Raster output')
@alg.input(type=alg.VECTOR_LAYER_DEST, name='BUFFER_OUTPUT',
           label='Buffer output')
@alg.input(type=alg.DISTANCE, name='BUFFERDIST', label='BUFFER DISTANCE',
           default=1.0)
@alg.input(type=alg.DISTANCE, name='CELLSIZE', label='RASTER CELL SIZE',
           default=10.0)
@alg.output(type=alg.NUMBER, name='NUMBEROFFEATURES',
            label='Number of features processed')

def bufferrasteralg(instance, parameters, context, feedback, inputs):
    """
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    Description of the algorithm.
    (If there is no comment here, you will get an error)
    """
    input_featuresource = instance.parameterAsSource(parameters,
                                                     'INPUT', context)
    numfeatures = input_featuresource.featureCount()
    bufferdist = instance.parameterAsDouble(parameters, 'BUFFERDIST',
                                            context)
    rastercellsize = instance.parameterAsDouble(parameters, 'CELLSIZE',
                                                context)
    if feedback.isCanceled():
        return {}
    buffer_result = processing.run('native:buffer',
                               {'INPUT': parameters['INPUT'],
                                'OUTPUT': parameters['BUFFER_OUTPUT'],
                                'DISTANCE': bufferdist,
                                'SEGMENTS': 10,
                                'DISSOLVE': True,
                                'END_CAP_STYLE': 0,
                                'JOIN_STYLE': 0,
                                'MITER_LIMIT': 10
                                },
                               is_child_algorithm=True,
                               context=context,
                               feedback=feedback)
    if feedback.isCanceled():
        return {}
    rasterized_result = processing.run('qgis:rasterize',
                               {'LAYER': buffer_result['OUTPUT'],
                                'EXTENT': buffer_result['OUTPUT'],
                                'MAP_UNITS_PER_PIXEL': rastercellsize,
                                'OUTPUT': parameters['OUTPUT']
                               },
                               is_child_algorithm=True, context=context,
                               feedback=feedback)
    if feedback.isCanceled():
        return {}
    return {'OUTPUT': rasterized_result['OUTPUT'],
            'BUFFER_OUTPUT': buffer_result['OUTPUT'],
            '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 graphical modeler.

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.

22.8.3 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 (`algfactory.py` contains the complete list of alg constants). Sorted on class name.

Input types

Class	Alg constant	Beschreibung
`QgsProcessingParameterAuthConfig`	`alg.AUTH_CFG`	Allows users to select from available authentication configurations or create new authentication configurations
`QgsProcessingParameterBand`	`alg.BAND`	A band of a raster layer
`QgsProcessingParameterBoolean`	`alg.BOOL`	A boolean value
`QgsProcessingParameterColor`	`alg.COLOR`	A color
`QgsProcessingParameterCrs`	`alg.CRS`	A Coordinate Reference System
`QgsProcessingParameterDistance`	`alg.DISTANCE`	A double numeric parameter for distance values
`QgsProcessingParameterEnum`	`alg.ENUM`	An enumeration, allowing for selection from a set of predefined values
`QgsProcessingParameterExpression`	`alg.EXPRESSION`	An expression
`QgsProcessingParameterExtent`	`alg.EXTENT`	A spatial extent defined by xmin, xmax, ymin, ymax
`QgsProcessingParameterField`	`alg.FIELD`	A field in the attribute table of a vector layer
`QgsProcessingParameterFile`	`alg.FILE`	A filename of an existing file
`QgsProcessingParameterFileDestination`	`alg.FILE_DEST`	A filename for a newly created output file
`QgsProcessingParameterFolderDestination`	`alg.FOLDER_DEST`	A folder
`QgsProcessingParameterNumber`	`alg.INT`	An integer
`QgsProcessingParameterLayout`	`alg.LAYOUT`	A layout
`QgsProcessingParameterLayoutItem`	`alg.LAYOUT_ITEM`	A layout item
`QgsProcessingParameterMapLayer`	`alg.MAPLAYER`	A map layer
`QgsProcessingParameterMatrix`	`alg.MATRIX`	A matrix
`QgsProcessingParameterMeshLayer`	`alg.MESH_LAYER`	A mesh layer
`QgsProcessingParameterMultipleLayers`	`alg.MULTILAYER`	A set of layers
`QgsProcessingParameterNumber`	`alg.NUMBER`	A numerical value
`QgsProcessingParameterPoint`	`alg.POINT`	A point
`QgsProcessingParameterRange`	`alg.RANGE`	A number range
`QgsProcessingParameterRasterLayer`	`alg.RASTER_LAYER`	Rasterlayer
`QgsProcessingParameterRasterDestination`	`alg.RASTER_LAYER_DEST`	Rasterlayer
`QgsProcessingParameterScale`	`alg.SCALE`	A map scale
`QgsProcessingParameterFeatureSink`	`alg.SINK`	A feature sink
`QgsProcessingParameterFeatureSource`	`alg.SOURCE`	A feature source
`QgsProcessingParameterScale`		A map scale
`QgsProcessingParameterString`	`alg.STRING`	A text string
`QgsProcessingParameterVectorLayer`	`alg.VECTOR_LAYER`	Vektorlayer
`QgsProcessingParameterVectorDestination`	`alg.VECTOR_LAYER_DEST`	Vektorlayer

Output types

Class	Alg constant	Beschreibung
QgsProcessingOutputBoolean	alg.BOOL	A boolean value
QgsProcessingOutputNumber	alg.DISTANCE	A double numeric parameter for distance values
QgsProcessingOutputFile	alg.FILE	A filename of an existing file
QgsProcessingOutputFolder	alg.FOLDER	A folder
QgsProcessingOutputHtml	alg.HTML	HTML
QgsProcessingOutputNumber	alg.INT	A integer
QgsProcessingOutputLayerDefinition	alg.LAYERDEF	A layer definition
QgsProcessingOutputMapLayer	alg.MAPLAYER	A map layer
QgsProcessingOutputMultipleLayers	alg.MULTILAYER	A set of layers
QgsProcessingOutputNumber	alg.NUMBER	A numerical value
QgsProcessingOutputRasterLayer	alg.RASTER_LAYER	Rasterlayer
QgsProcessingOutputString	alg.STRING	A text string
QgsProcessingOutputVectorLayer	alg.VECTOR_LAYER	Vektorlayer

22.8.4 Handing 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.

22.8.5 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, …)

22.8.6 Ihre Scripte dokumentieren

You can document your scripts by overloading the `helpString()` and `helpUrl()` methods of `QgsProcessingAlgorithm`.

22.8.7 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.

22.8.8 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 https://github.com/qgis/QGIS/blob/release-3_10/python/plugins/processing/algs/qgis.

22.9 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 graphical modeler, 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.

22.9.1 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.

22.9.2 Eine Bemwerkung 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.

22.9.3 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.

22.9.4 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 Mahrfachrasterlayer 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 bauchen, 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 Pixelgöße 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.

22.9.5 R scripts

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 *http://cran.at.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.

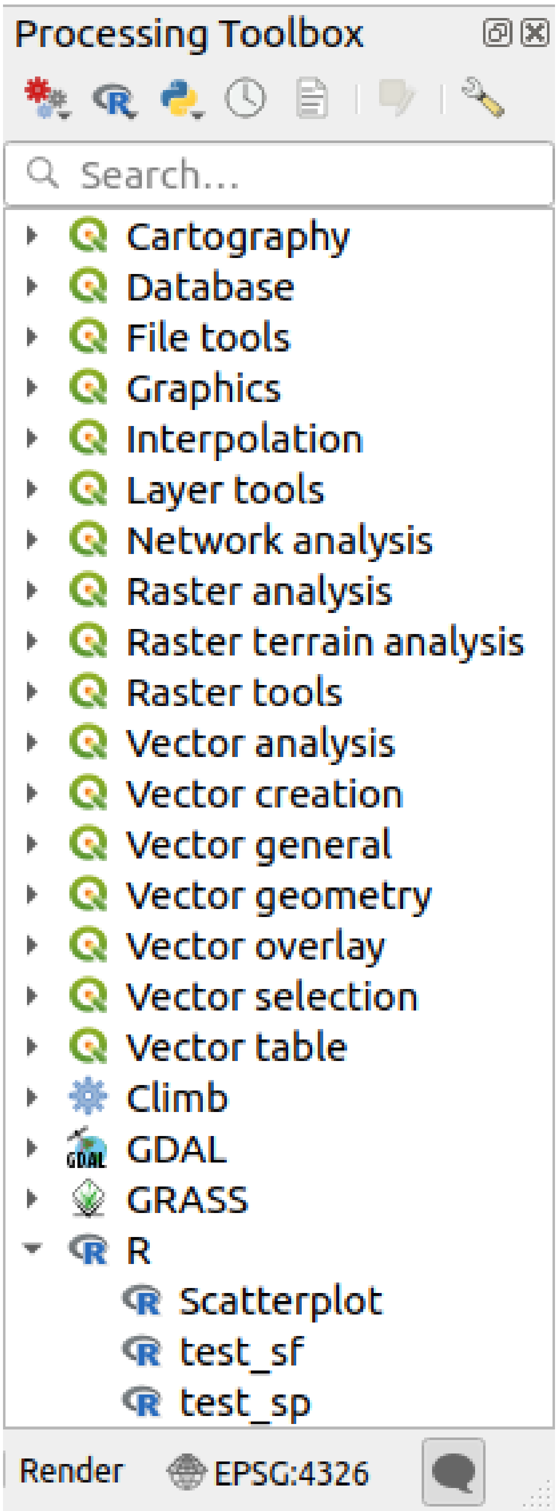
If you have R configured correctly for QGIS, you should be able to run these scripts.

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


Processing Toolbox
Search...
Cartography
Database
File tools
Graphics
Interpolation
Layer tools
Network analysis
Raster analysis
Raster terrain analysis
Raster tools
Vector analysis
Vector creation
Vector general
Vector geometry
Vector overlay
Vector selection
Vector table
Climb
GDAL
GRASS
R
Scatterplot
test_sf
test_sp
Render
EPSG:4326

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).

   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 scrips 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.

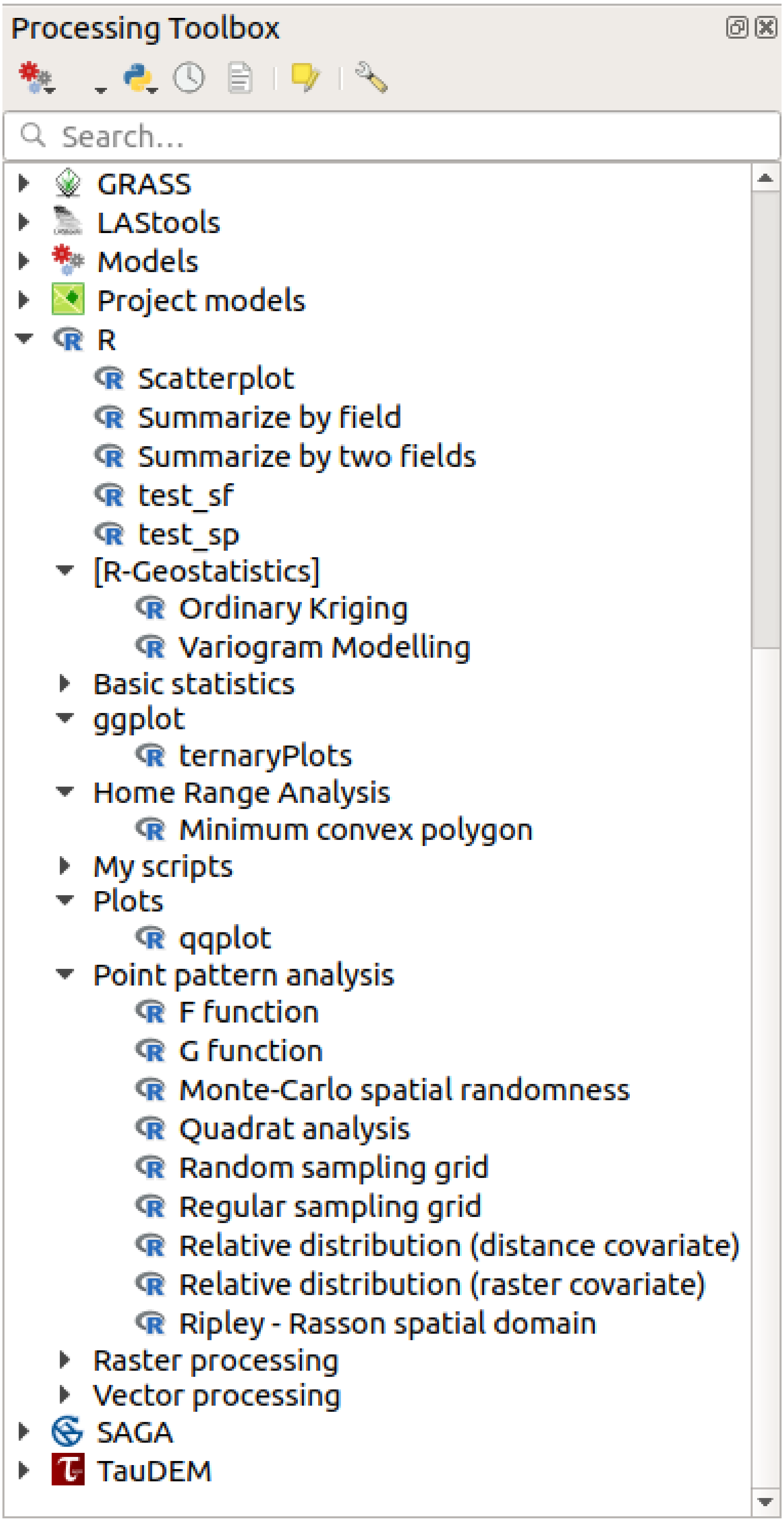
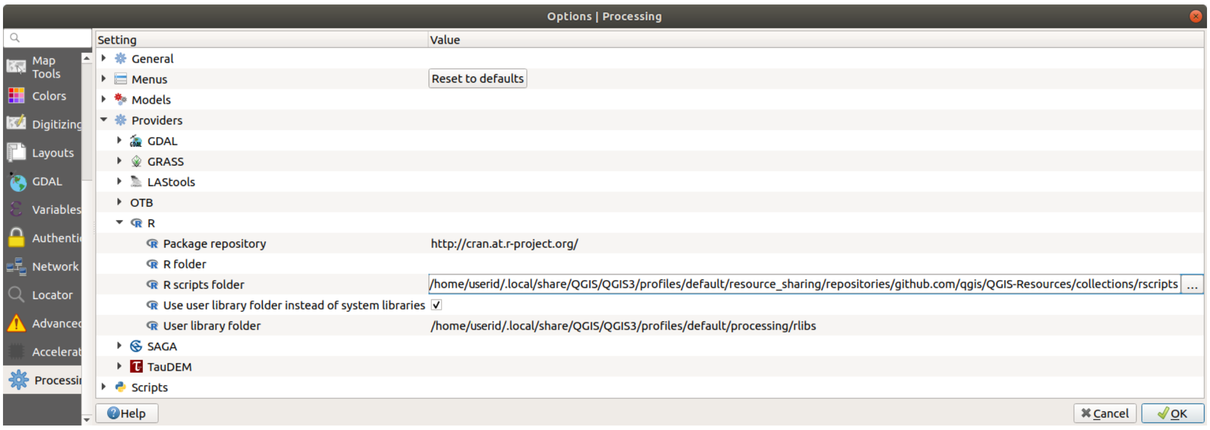
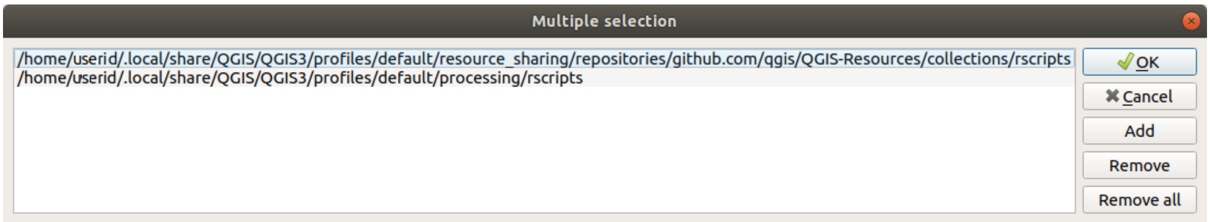


Abb. 22.33: The *Processing Toolbox* with some R scripts shown





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 elsewhere). 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.

22.9.6 R libraries

The R script `sp_test` tries to load the R packages `sp` and `raster`.

R libraries installed when running sf_test

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`

22.9.7 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.

22.9.8 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.

22.9.9 OTB Applications

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.

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* (left panel)*
2. You can see OTB under „Providers“:
   1. Expand the *OTB* tab
   2. Tick the *Activate* option
   3. Set the *OTB folder*. This is the location of your OTB installation.
   4. Set the *OTB application folder*. This is the location of your OTB applications (<PATH_TO_OTB_INSTALLATION>/lib/otb/applications)
   5. 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

- **Activate**: This is a checkbox to activate or deactivate the OTB provider. An invalid OTB setting will uncheck this when saved.
- **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 between QGIS and OTB versions

OTB compiled with GDAL 3.X is not compatible with QGIS 3.10. This is the case for the binary packages of OTB 7.1 and above. Therefore QGIS 3.10 is only compatible with OTB official binary packages 6.6.1 and 7.0.0.

Troubleshoot

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 here

KAPITEL 23

Datenanbieter und Algorithmen in Verarbeitung

Im Folgenden werden die Algorithmen und ihre Parameter beschrieben (wie sie in der Benutzeroberfläche dargestellt werden).

23.1 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.

23.1.1 Cartography

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

Label	Name	Type	Beschreibung
Input databases	INPUT	[file] [list]	Files containing QGIS style items
Objects to combine	OBJECTS	[enumeration] [list]	Types of style items in the input databases you would like to put in the new database. These can be: • 0 — *Symbols* • 1 — *Color ramps* • 2 — *Text formats* • 3 — *Label settings*
Output style database	OUTPUT	[file] Default: `[Save to temporary file]`	Output `.XML` file combining the selected style items. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Color ramp count	COLORRAMPS	[number]	
Label settings count	LABELSETTINGS	[number]	
Output style database	OUTPUT	[file]	Output `.XML` file combining the selected style items
Symbol count	SYMBOLS	[number]	
Text format count	TEXTFORMATS	[number]	

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Vector layer to apply a categorized style to
Categorize using expression	FIELD	[expression]	Field or expression to categorize the features
Style database (leave blank to use saved symbols)	STYLE	[file]	File (.XML) containing the symbols to apply to the input layer categories. The file can be obtained from the Style Manager *Share symbols* tool. If no file is specified, QGIS local symbols library is used.
Use case-sensitive match to symbol names	CASE_SENSITIVE	[boolean] Default: False	If True (checked), applies a case sensitive comparison between the categories and symbols names
Ignore non-alphanumeric characters while matching	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.
Non-matching categories Optional	NON_MATCHING_CATEGORIES	[table] Default: [Skip output]	Output table for categories which do not match any symbol in the database. One of: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Non-matching symbol names Optional	NON_MATCHING_SYMBOLS	[table] Default: [Skip output]	Output table for symbols from the provided style database which do not match any category. One of: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Non-matching categories	NON_MATCHING_CATEGORIES	[table]	Lists categories which could not be matched to any symbol in the provided style database
Non-matching symbol names	NON_MATCHING_SYMBOLS	[table]	Lists symbols from the provided style database which could not match any category
Categorized layer	OUTPUT	[same as input]	The input vector layer with the categorized style applied. No new layer is output.

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input project (leave blank to use current) Optional	INPUT	[file]	A QGIS project file to extract the style items from
Objects to extract	OBJECTS	[enumeration] [list]	Types of style items in the input project you would like to put in the new database. These can be: • 0 — *Symbols* • 1 — *Color ramps* • 2 — *Text formats* • 3 — *Label settings*
Output style database	OUTPUT	[file] Default: [Save to temporary file]	Specify the output .XML file for the selected style items. One of: • Save to a Temporary Layer (TEMPORARY_OUTPUT) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Color ramp count	COLORRAMPS	[number]	Number of color ramps
Label settings count	LABELSETTINGS	[number]	Number of label settings
Output style database	OUTPUT	[file]	Output .XML file for the selected style items
Symbol count	SYMBOLS	[number]	Number of symbols
Text format count	TEXTFORMATS	[number]	Number of text formats

Python code

Algorithm ID: `qgis: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.

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.

Label	Name	Type	Beschreibung
Print layout	`LAYOUT`	[enumeration]	A print layout in the current project
Map item Optional	`MAP`	[enumeration] Default: *All the map items*	The map item(s) whose information you want to extract. If none is provided then all the map items are processed.
Overrride CRS Optional	`CRS`	[crs] Default: *The layout CRS*	Select the CRS for the layer in which the information will be reported.
Extent	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer for the extent(s). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Map height	`HEIGHT`	[number]	
Extent	`OUTPUT`	[vector: polygon]	Output polygon vector layer containing extents of all the input layout map item(s)
Map rotation	`ROTATION`	[number]	
Map scale	`SCALE`	[number]	
Map width	`WIDTH`	[number]	

Python code

Algorithm ID: `qgis: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.

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.

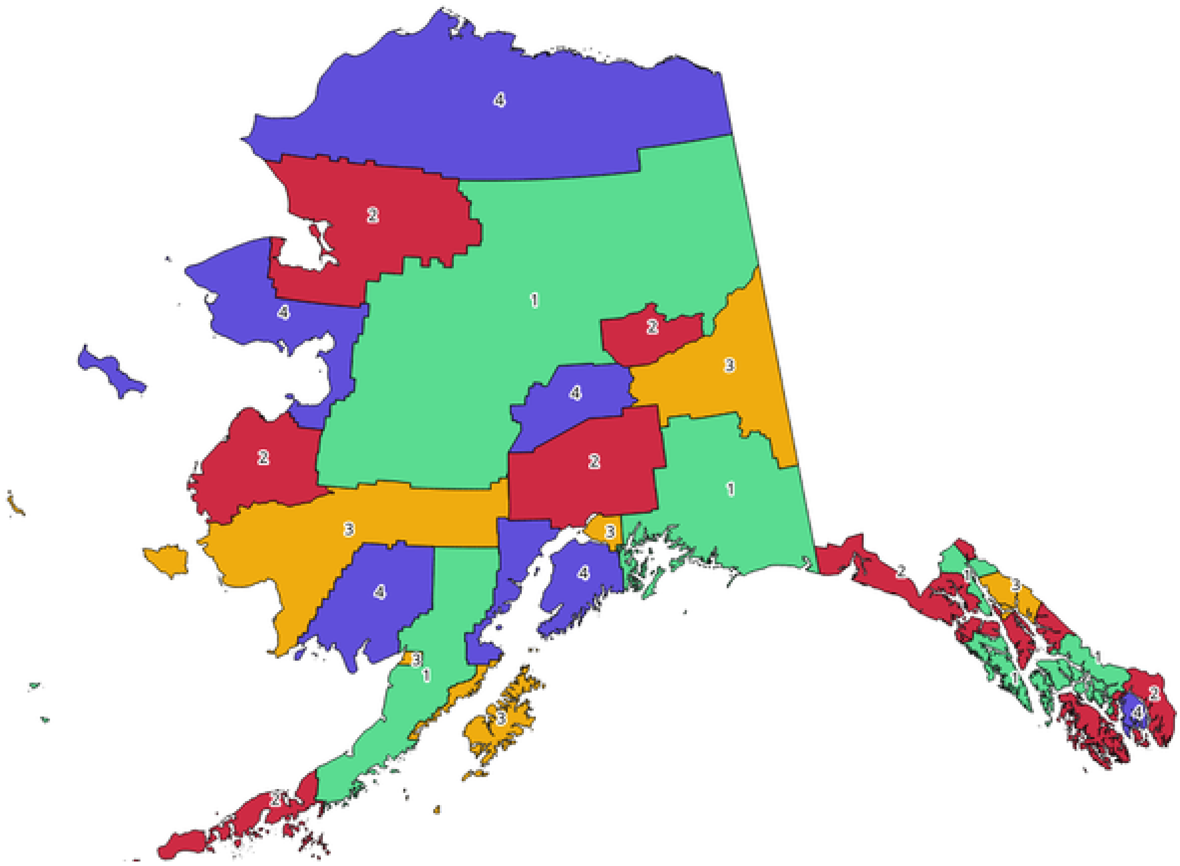


Abb. 23.1: Topological colors example

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	The input polygon layer
Minimum number of colors	`MIN_COLORS`	[number] Default: 4	The minimum number of colors to assign. Minimum 1, maximum 1000.
Minimum distance between features	`MIN_DISTANCE`	[number] Default: 0.0	Prevent nearby (but non-touching) features from being assigned equal colors. Minimum 0.0.
Balance color assignment	`BALANCE`	[enumeration] Default: 0	Options are: • 0 — By feature count Attempts to assign colors so that the count of features assigned to each individual color index is balanced. • 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. • 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.
Colored	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Colored	`OUTPUT`	[vector: polygon]	Polygon vector layer with an added `color_id` column

Python code

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.

23.1.2 Datenbank

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*).

Parameter

Label	Name	Type	Beschreibung
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.
Schema (schema name) Optional	`SCHEMA`	[string] Default: ‚public‘	Name of the schema to store the data. It can be a new one or already exist.
Table to import to (leave blank to use layer name) Optional	`TABLENAME`	[string] Default: ‚‘	Defines a table name for the imported vector file. If nothing is added, the layer name will be used.
Primary key field Optional	`PRIMARY_KEY`	[tablefield: any]	Sets the primary key field from an existing field in the vector layer. A column with **unique** values can be used as Primary key for the database.
Geometry column	`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.
Encoding Optional	`ENCODING`	[string] Default: ‚UTF-8‘	Defines the encoding of the output layer
Overwrite	`OVERWRITE`	[boolean] Default: True	If the specified table exists, setting this option to `True` will make sure that it is deleted and a new table will be created before the features are added. If this option is `False` and the table exists, the algorithm will throw an exception („relation already exists“).
Create spatial index	`CREATEINDEX`	[boolean] Default: True	Specifies whether to create a spatial index or not
Convert field names to lowercase	`LOWERCASE_NAMES`	[boolean] Default: True	Converts the field names of the input vector layer to lowercase
Drop length constraint on character fields	`DROP_STRING_LENGTH`	[boolean] Default: False	Should length constraints on character fields be dropped or not
Create single-part geometries instead of multi-part	`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.

Python code

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*).

Parameter

Label	Name	Type	Beschreibung
Layer to import	`INPUT`	[vector: any]	Vector layer to add to the database
File database	`DATABASE`	[vector: any]	The SQLite/SpatiaLite database file to connect to
Table to import to (leave blank to use layer name) Optional	`TABLENAME`	[string] Default: ‚‘	Defines the table name for the imported vector file. If nothing is specified, the layer name will be used.
Primary key field Optional	`PRIMARY_KEY`	[tablefield: any]	Use a field in the input vector layer as the primary key
Geometry column	`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.
Encoding Optional	`ENCODING`	[string] Default: ‚UTF-8‘	Defines the encoding of the output layer
Overwrite	`OVERWRITE`	[boolean] Default: True	If the specified table exists, setting this option to `True` 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 `False` and the table exists, the algorithm will throw an exception („table already exists“).
Create spatial index	`CREATEINDEX`	[boolean] Default: True	Specifies whether to create a spatial index or not
Convert field names to lowercase	`LOWERCASE_NAMES`	[boolean] Default: True	Convert the field names of the input vector layer to lowercase
Drop length constraint on character fields	`DROP_STRING_LENGTH`	[boolean] Default: False	Should length constraints on character fields be dropped or not

Fortsetzung auf der nächsten Seite

Tab. 23.4 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Create single-part geometries instead of multi-part	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.

Python code

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.

Parameter

Label	Name	Type	Beschreibung
Input layers	LAYERS	[vector: any] [list]	The (vector) layers to import into the GeoPackage. Raster layers are not supported. If a raster layer is added, a `QgsProcessingException` will be thrown.
Overwrite existing GeoPackage	OVERWRITE	[boolean] Default: False	If the specified GeoPackage exists, setting this option to `True` will make sure that it is deleted and a new one will be created before the layers are added. If set to `False`, layers will be appended.
Save layer styles into GeoPackage	SAVE_STYLES	[boolean] Default: True	Save the layer styles
Destination GeoPackage	OUTPUT	[file]	If not specified the GeoPackage will be saved in the temporary folder.

Ausgaben

Label	Name	Type	Beschreibung
Layers within new package	OUTPUT_LAYERS	[string] [list]	The list of layers added to the GeoPackage.

Python code

Algorithm ID: qgis: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*

Parameter

Label	Name	Type	Beschreibung
Database (connection name)	DATABASE	[string]	The database connection (not the database name). Existing connections will be shown in the combobox.
SQL query	SQL	[string]	Defines the SQL query, for example `'UPDATE my_table SET field=10'`.
Unique ID field name	ID_FIELD	[string] Default: id	Sets the primary key field (a column in the result table)
Geometry field name Optional	GEOMETRY_FIELD	[string] Default: ‚geom‘	Name of the geometry column (a column in the result table)

Ausgaben

Label	Name	Type	Beschreibung
SQL layer	`OUTPUT`	[vector: any]	The resulting vector layer to be loaded into QGIS.

Python code

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*

Parameter

Label	Name	Type	Beschreibung
Database (connection name)	`DATABASE`	[string]	The database connection (not the database name). Existing connections will be shown in the combobox.
SQL query	`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.

Python code

Algorithm ID: `qgis: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 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*.

Parameter

Label	Name	Type	Beschreibung
File Database	`DATABASE`	[vector] Default: not set	The SQLite/SpatiaLite database file to connect to
SQL query	`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.

Python code

Algorithm ID: `qgis: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.

23.1.3 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

Label	Name	Type	Beschreibung
URL	`URL`	[string]	The URL of the file to download.
File destination	`OUTPUT`	[string] Default: `[Save to temporary file]`	Specification of the file destination. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
File destination	`OUTPUT`	[string]	The location of the downloaded file

Python code

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.

23.1.4 Graphics

Bar plot

Creates a bar plot from a category and a layer field.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Category field name	`NAME_FIELD`	[tablefield: any]	Categorical field to use for grouping the bars (X axis)
Value field	`VALUE_FIELD`	[tablefield: any]	Value to use for the plot (Y axis).
Bar plot	`OUTPUT`	[html] Default: `[Save to temporary file]`	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Bar plot	OUTPUT	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Category name field	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the boxes (X axis)
Value field	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).
Additional statistic lines	MSD	[enumeration] Default: 0	Additional statistics information to add to the plot. One of: • 0 — Show Mean • 1 — Show Standard Deviation • 2 — Don't show mean and standard deviation
Box plot	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Box plot	`OUTPUT`	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input table	`INPUT`	[vector: any]	Input vector layer
Category name field	`NAME_FIELD`	[tablefield: any]	Categorical field to use for grouping the boxes (X axis)
Value field	`VALUE_FIELD`	[tablefield: any]	Value to use for the plot (Y axis).
Plot	`OUTPUT`	[html] Default: `[Save to temporary file]`	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Plot	`OUTPUT`	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Category name field	NAME_FIELD	[tablefield: any]	Categorical field to use for grouping the features (X axis)
Value field	VALUE_FIELD	[tablefield: any]	Value to use for the plot (Y axis).
Polar plot	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Polar plot	OUTPUT	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Band number	BAND	[raster band]	Raster band to use for the histogram
number of bins	BINS	[number] Default: 10	The number of bins to use in the histogram (X axis). Minimum 2.
Histogram	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Histogram	OUTPUT	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Attribute	FIELD	[tablefield: any]	Value to use for the plot (Y axis).
number of bins	BINS	[number] Default: 10	The number of bins to use in the histogram (X axis). Minimum 2.
Histogram	OUTPUT	[html] Default: `[Save to temporary file]`	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Histogram	OUTPUT	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
X attribute	`XFIELD`	[tablefield: any]	Field to use for the X axis
Y attribute	`YFIELD`	[tablefield: any]	Field to use for the Y axis
Scatterplot	`OUTPUT`	[html] Default: `[Save to temporary file]`	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Scatterplot	`OUTPUT`	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

Algorithm ID: `qgis:vectorlayerscatterplot`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
X attribute	XFIELD	[tablefield: any]	Field to use for the X axis
Y attribute	YFIELD	[tablefield: any]	Field to use for the Y axis
Z attribute	ZFIELD	[tablefield: any]	Field to use for the Z axis
Histogram	OUTPUT	[html] Default: [Save to temporary file]	Specify the HTML file for the plot. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Histogram	OUTPUT	[html]	HTML file with the plot. Available in the *Processing ▸ Result Viewer*.

Python code

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.

23.1.5 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

Label	Name	Type	Beschreibung
Point layer	INPUT	[vector: point]	Point vector layer to use for the heatmap
Radius	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.
Output raster size	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`) **or** 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).
Radius from field Optional	RADIUS_FIELD	[tablefield: numeric]	Sets the search radius for each feature from an attribute field in the input layer.
Weight from field 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.
Kernel shape	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 then the Epanechnikov kernel does. Consequently, triweight results in “sharper” hotspots and Epanechnikov results in “smoother” hotspots. There are many shapes available (please see the Wikipedia page for further information): • 0 — Quartic • 1 — Triangular • 2 — Uniform • 3 — Triweight • 4 — Epanechnikov

Fortsetzung auf der nächsten Seite

Tab. 23.6 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Decay ratio (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. • 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. • 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. • A value of 1 means the heat is spread evenly over the whole search radius circle. (This is equivalent to the 'Uniform' kernel.) • A value greater than 1 indicates that the heat is higher towards the edge of the search radius than at the center.
Output value scaling	`OUTPUT_VALUE`	[enumeration] Default: *Raw*	Allow to change the values of the output heatmap raster. One of: • 0 — Raw • 1 — Scaled
Heatmap	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer with kernel density values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Heatmap	`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 *Figure_Heatmap_data_processing*, the airports of Alaska are shown.

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 *Figure_Heatmap_created_processing*).

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.

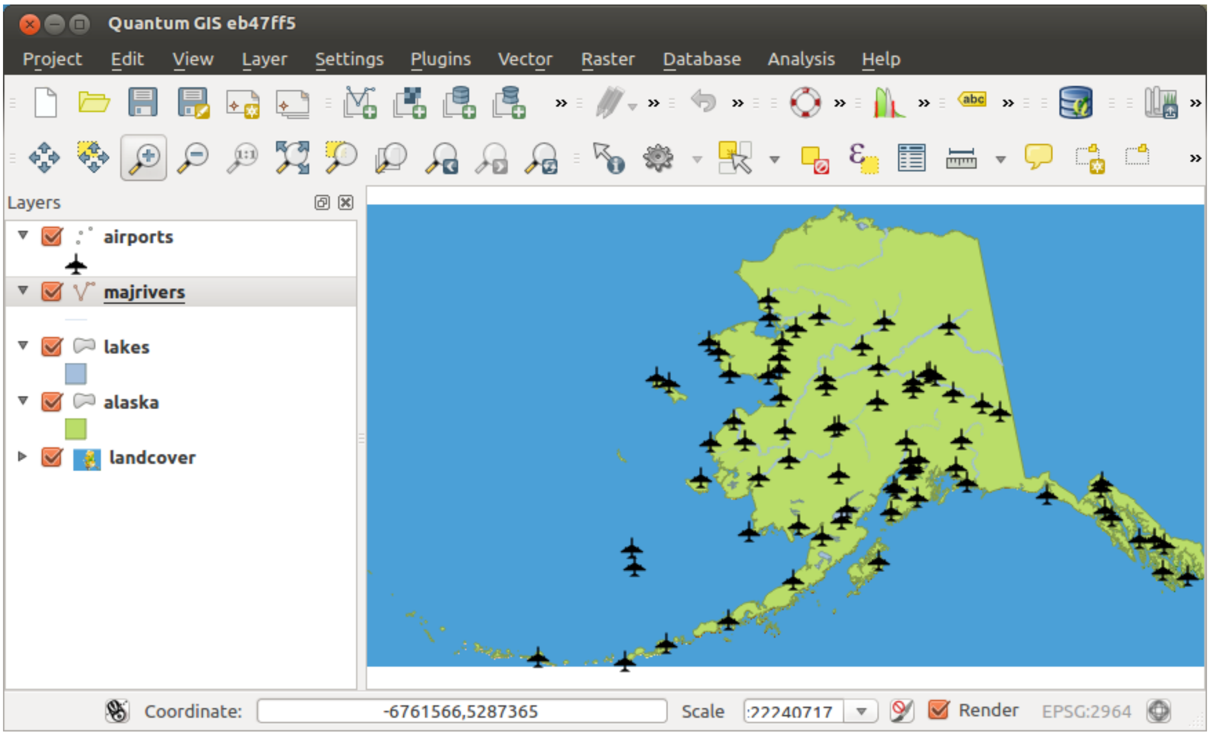


Abb. 23.2: Airports of Alaska

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 *Symbology* 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 *Figure_Heatmap_styled_processing*.

Python code

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.

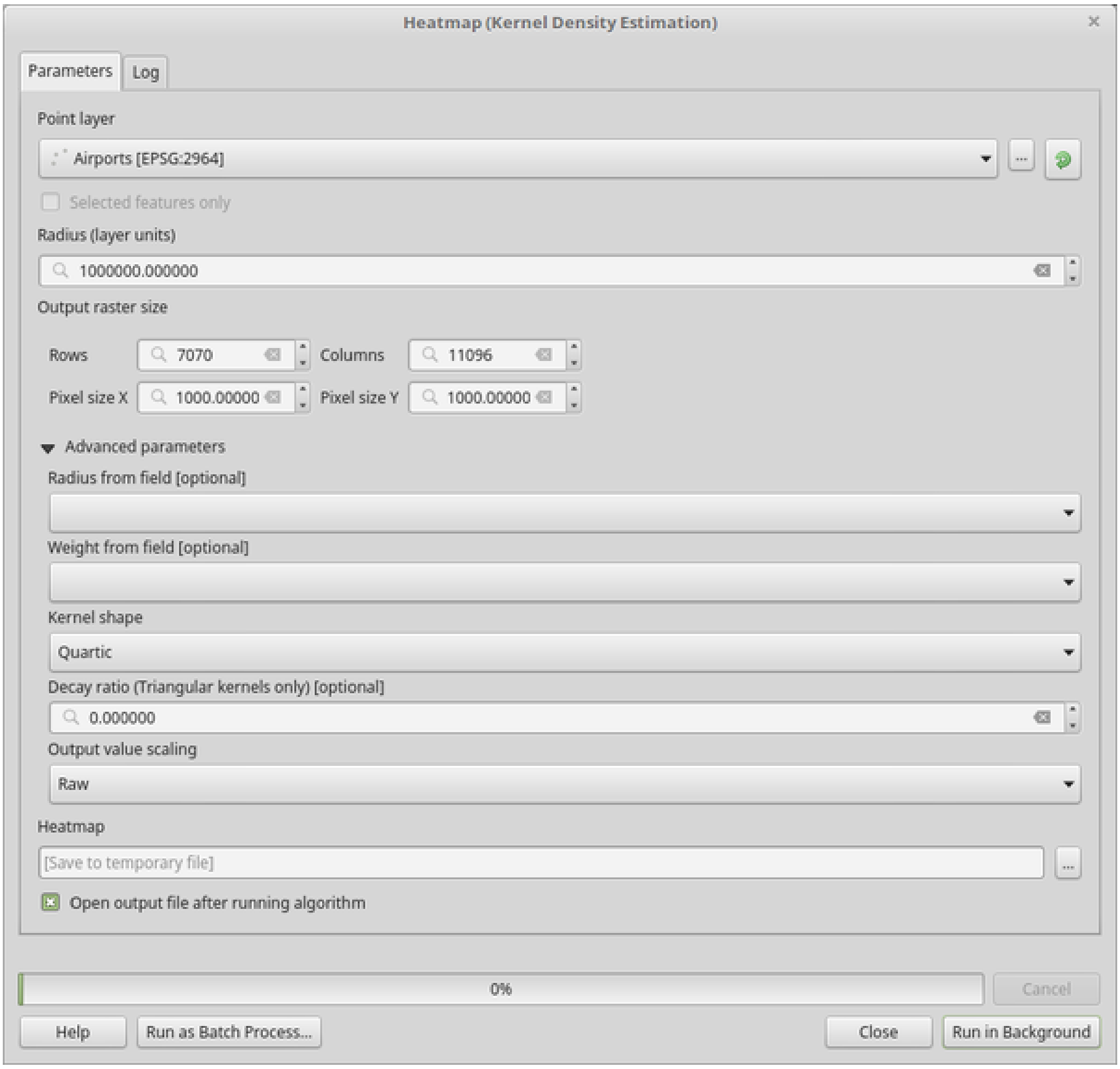


Abb. 23.3: The Heatmap Dialog

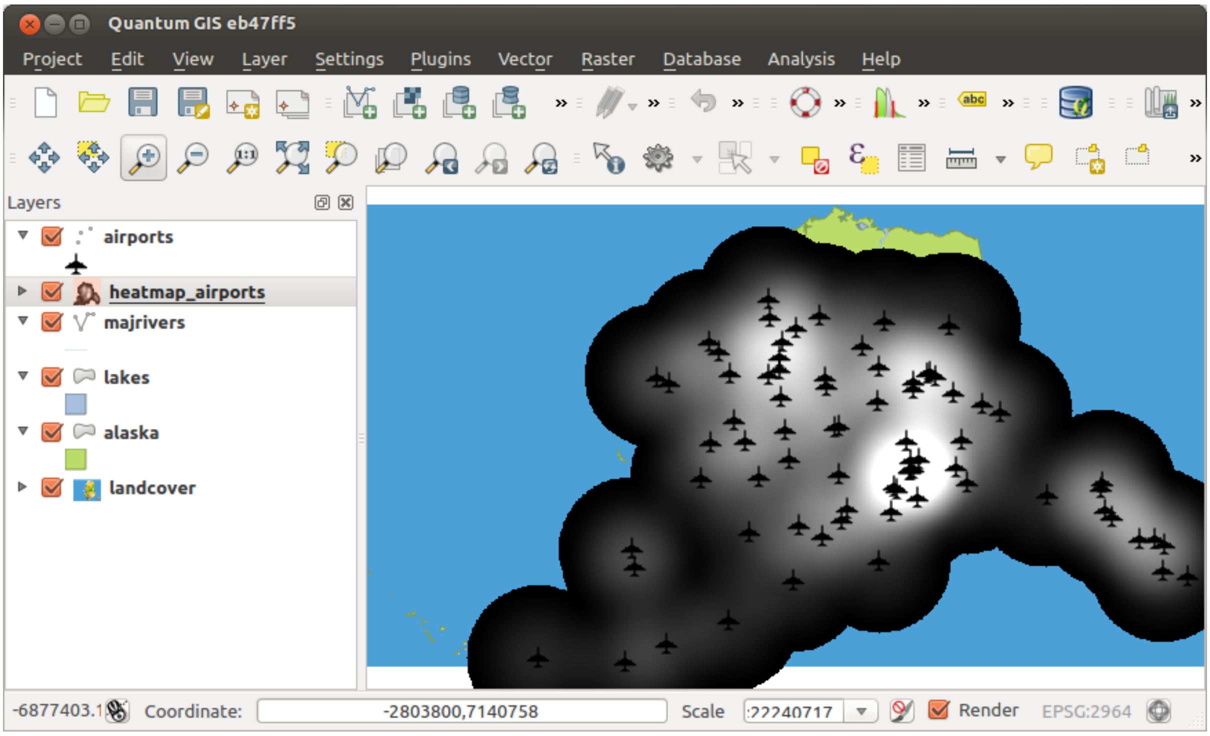


Abb. 23.4: The heatmap after loading looks like a grey surface

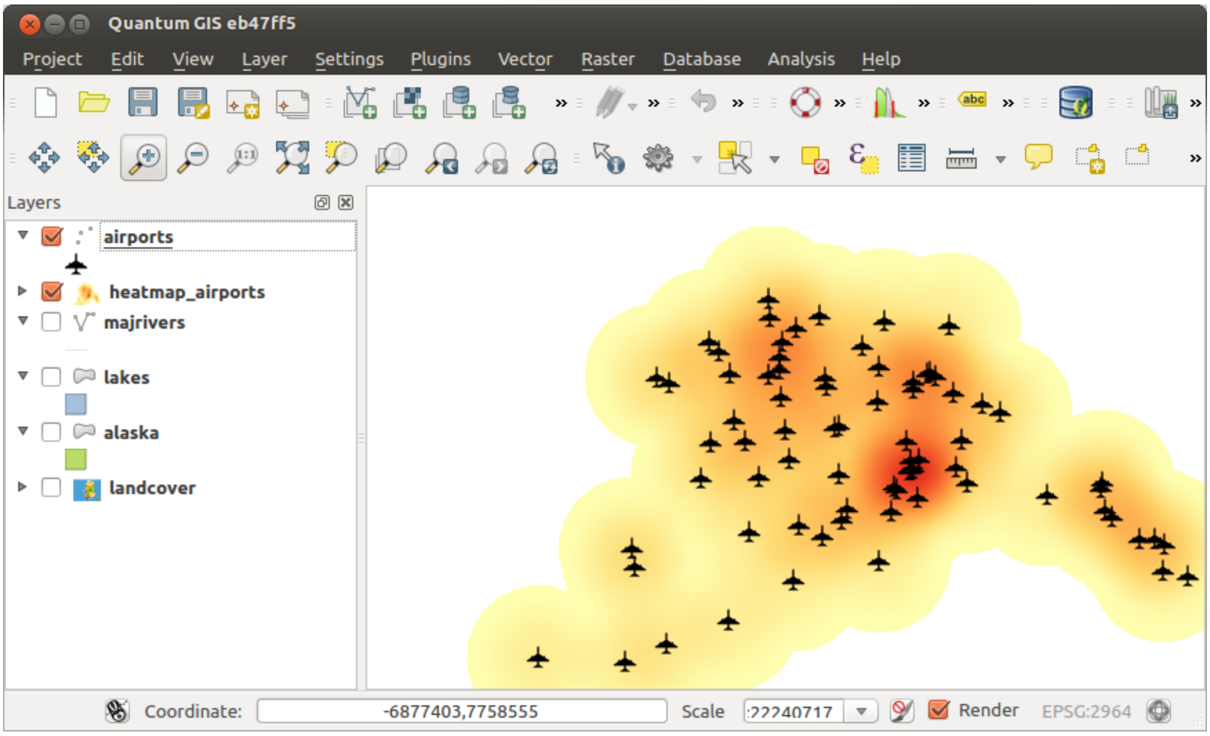


Abb. 23.5: Styled heatmap of airports of Alaska

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

Label	Name	Type	Beschreibung
Input layer(s)	`INTERPOLATION_DATA`	[string]	Vector layer(s) and field(s) to use for the interpolation, coded in a string (see the `ParameterInterpolationData` class in InterpolationWidgets for more details). The following GUI elements are provided to compose the interpolation data string: • **Vector layer** [vector: any] • **Interpolation attribute** [tablefield: numeric]: Attribute to use in the interpolation • **Use Z-coordinate for interpolation** [boolean]: Uses the layer's stored Z values (Default: False) For each of the added layer-field combinations, a type can be chosen: • *Points* • *Structured lines* • *Break lines* In the string, the layer-field elements are separated by `'::|::'`. The sub-elements of the layer-field elements are separated by `'::~::'`.
Distance coefficient P	`DISTANCE_COEFFICIENT`	[number] Default: 2.0	Sets the distance coefficient for the interpolation. Minimum: 0.0, maximum: 100.0.
Extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	Extent of the output raster layer. You have to declare the output extent by either choosing it from the map canvas, selecting it from another layer or type it manually.

Fortsetzung auf der nächsten Seite

Tab. 23.8 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output raster size	`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`) **or** 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).
Interpolated	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Raster layer of interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated	`OUTPUT`	[raster]	Raster layer of interpolated values

Python code

Algorithm ID: `qgis:idwinterpolation`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer(s)	INTERPOLATION_DATA	[string]	Vector layer(s) and field(s) to use for the interpolation, coded in a string (see the ParameterInterpolationData class in InterpolationWidgets for more details). The following GUI elements are provided to compose the interpolation data string: • **Vector layer** [vector: any] • **Interpolation attribute** [tablefield: numeric]: Attribute to use in the interpolation • **Use Z-coordinate for interpolation** [boolean]: Uses the layer's stored Z values (Default: False) For each of the added layer-field combinations, a type can be chosen: • *Points* • *Structured lines* • *Break lines* In the string, the layer-field elements are separated by '::|::'. The sub-elements of the layer-field elements are separated by '::~::'.
Interpolation method	METHOD	[enumeration] Default: 0	Set the interpolation method to be used. One of: • *Linear* • *Clough-Toucher (cubic)*
Extent (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Extent of the output raster layer. You have to declare the output extent by either choosing it from the map canvas, selecting it from another layer or type it manually.
Output raster size	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) **or** 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).
Interpolated	OUTPUT	[raster] Default: [Save to temporary file]	The output TIN interpolation as a raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 23.10 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Triangulation	`TRIANGULATION`	[vector: line] Default: `[Skip output]`	The output TIN as a vector layer. One of: • Skip Output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table…

Ausgaben

Label	Name	Type	Beschreibung
Interpolated	`OUTPUT`	[raster]	The output TIN interpolation as a raster layer
Triangulation	`TRIANGULATION`	[vector: line]	The output TIN as a vector layer.

Python code

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.

23.1.6 Layer tools

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.

Default menu: *Vector* ► *Research Tools*

Parameter

Label	Name	Type	Beschreibung
Layer	`INPUT`	[layer]	Input layer
Extent	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the polygon vector layer for the output extent. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

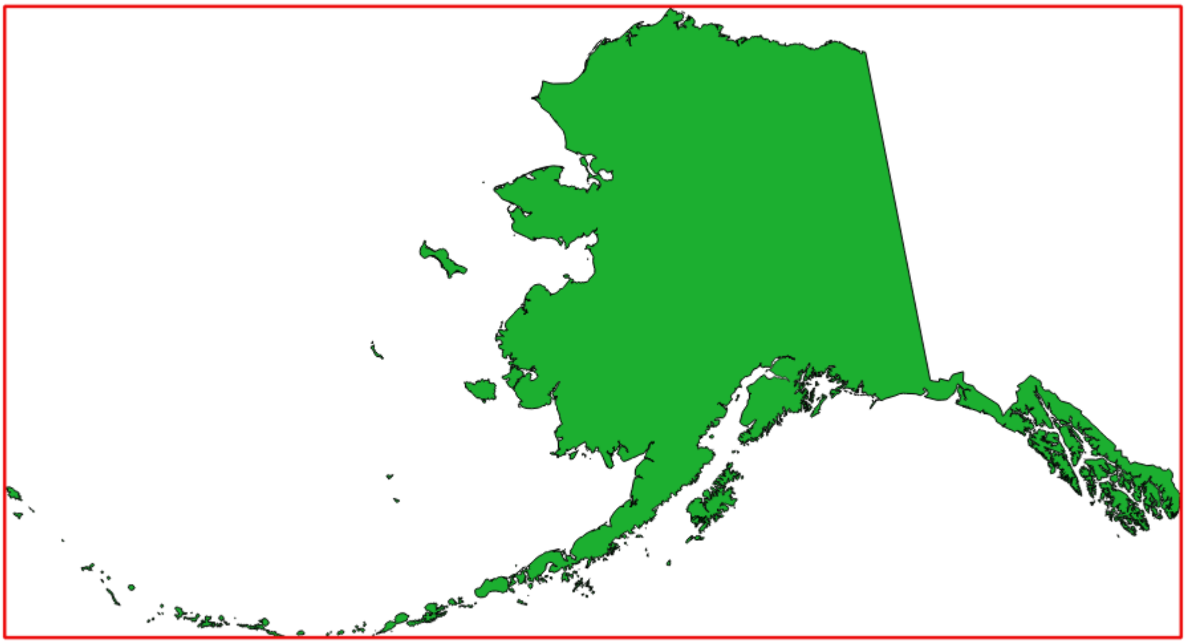


Abb. 23.6: In red the bounding box of the source layer

Ausgaben

Label	Name	Type	Beschreibung
Extent	`OUTPUT`	[vector: polygon]	Output (polygon) vector layer with the extent (minimum bounding box)

Python code

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.

23.1.7 Modeler tools

These tools are only available in the Graphical Modeler. They are not available in the Processing Toolbox.

Load layer into project

Loads a layer to the current project.

Parameter

Label	Name	Type	Beschreibung
Layer	`INPUT`	[layer]	Layer to load in the legend
Loaded layer name	`NAME`	[string]	Name of the loaded layer

Ausgaben

Label	Name	Type	Beschreibung
Layer	`OUTPUT`	[same as input]	The (renamed) loaded layer

Python code

Algorithm ID: qgis: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.

Rename layer

Renames a layer.

Parameter

Label	Name	Type	Beschreibung
Layer	INPUT	[layer]	Layer to rename
New name	NAME	[string]	The new name of the layer

Ausgaben

Label	Name	Type	Beschreibung
Layer	OUTPUT	[same as input]	The (renamed) output layer

Python code

Algorithm ID: qgis: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.

String concatenation

Concatenates two strings into a single one in the Processing Modeler.

Parameter

Label	Name	Type	Beschreibung
Input 1	`INPUT_1`	[string]	First string
Input 2	`INPUT_2`	[string]	Second string

Ausgaben

Label	Name	Type	Beschreibung
Concatenation	`CONCATENATION`	[string]	The concatenated string

Python code

Algorithm ID: `qgis: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.

23.1.8 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

Label	Name	Type	Beschreibung
Vector layer representing network	`INPUT`	[vector: line]	Line vector layer representing the network to be covered
Vector layer with start points	`START_POINTS`	[vector: point]	Point vector layer whose features are used as start points to generate the service areas
Path type to calculate	`STRATEGY`	[enumeration] Default: 0	The type of path to calculate. One of: • 0 — Shortest • 1 — Fastest
Travel cost (distance for „Shortest“, time for „Fastest“	`TRAVEL_COST`	[number] Default: 0	The value is estimated as a distance (in the network layer units) when looking for the *Shortest* path and as time (in seconds) for the *Fastest* path.

Direction field Optional	DIRECTION_FIELD	[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.
Value for forward direction Optional	VALUE_FORWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a forward direction
Value for backward direction Optional	VALUE_BACKWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a backward direction
Value for both directions Optional	VALUE_BOTH	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify bidirectional edges
Default direction Optional	DEFAULT_DIRECTION	[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: • 0 — Forward direction • 1 — Backward direction • 2 — Both directions
Speed field 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.
Default speed (km/h) Optional	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
Topology tolerance Optional	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected

Include upper/lower bound points	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.

Fortsetzung auf der nächsten Seite

Tab. 23.14 – Fortsetzung der vorherigen Seite

Service area (lines)	`OUTPUT_LINES`	[vector: line] Default: `[Create temporary layer]`	Specify the output line layer for the service area. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Service area (boundary nodes)	`OUTPUT`	[vector: point] Default: `[Skip output]`	Specify the output point layer for the service area boundary nodes. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Service area (boundary nodes)	`OUTPUT`	[vector: point]	The output point layer with the service area boundary nodes.
Service area (lines)	`OUTPUT_LINES`	[vector: line]	Line layer representing the parts of the network that can be serviced by the start points, for the given cost.

Python code

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

Label	Name	Type	Beschreibung
Vector layer representing the network	INPUT	[vector: line]	Line vector layer representing the network to be covered
Start point (x, y)	START_POINT	[coordinates]	Coordinate of the point to calculate the service area around.
Path type to calculate	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: • 0 — Shortest • 1 — Fastest
Travel cost	TRAVEL_COST	[number] Default: 0	The value is estimated as a distance (in the network layer units) when looking for the *Shortest* path and as time (in seconds) for the *Fastest* path.
Advanced parameters	GUI only		Group of advanced network analysis parameters - see below.
Service area (lines)	OUTPUT_LINES	[vector: line] Default: [Create temporary layer]	Specify the output line layer for the service area. One of: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Service area (boundary nodes)	OUTPUT	[vector: point] Default: [Skip output]	Specify the output point layer for the service area boundary nodes. One of: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Advanced parameters

Label	Name	Type	Beschreibung
Direction field Optional	DIRECTION_FIELD	[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.

Fortsetzung auf der nächsten Seite

Tab. 23.16 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Value for forward direction Optional	VALUE_FORWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a forward direction
Value for backward direction Optional	VALUE_BACKWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a backward direction
Value for both directions Optional	VALUE_BOTH	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify bidirectional edges
Default direction Optional	DEFAULT_DIRECTION	[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: • 0 — Forward direction • 1 — Backward direction • 2 — Both directions
Speed field 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.
Default speed (km/h) Optional	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
Topology tolerance Optional	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected
Include upper/lower bound points	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

Label	Name	Type	Beschreibung
Service area (boundary nodes)	OUTPUT	[vector: point]	The output point layer with the service area boundary nodes.
Service area (lines)	OUTPUT_LINES	[vector: line]	Line layer representing the parts of the network that can be serviced by the start point, for the given cost.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Vector layer representing network	`INPUT`	[vector: line]	Line vector layer representing the network to be covered
Path type to calculate	`STRATEGY`	[enumeration] Default: 0	The type of path to calculate. One of: • 0 — Shortest • 1 — Fastest
Vector layer with start points	`START_POINTS`	[vector: point]	Point vector layer whose features are used as start points of the routes
End point (x, y)	`END_POINT`	[coordinates]	Point feature representing the end point of the routes
Advanced parameters	GUI only		The **Advanced parameters** group:
Direction field Optional	`DIRECTION_FIELD`	[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.
Value for forward direction Optional	`VALUE_FORWARD`	[string] Default: ‚' (empty string)	Value set in the direction field to identify edges with a forward direction
Value for backward direction Optional	`VALUE_BACKWARD`	[string] Default: ‚' (empty string)	Value set in the direction field to identify edges with a backward direction
Value for both directions Optional	`VALUE_BOTH`	[string] Default: ‚' (empty string)	Value set in the direction field to identify bidirectional edges

Fortsetzung auf der nächsten Seite

Tab. 23.17 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Default direction Optional	`DEFAULT_DIRECTION`	[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: • 0 — Forward direction • 1 — Backward direction • 2 — Both directions
Speed field 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.
Default speed (km/h) Optional	`DEFAULT_SPEED`	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
Topology tolerance Optional	`TOLERANCE`	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected
			End of the **Advanced parameters** group
Shortest path	`OUTPUT`	[vector: line]	Specify the output line layer for the shortest paths. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Shortest path	`OUTPUT`	[vector: line]	Line layer of the shortest or fastest path from each of the start points to the end point

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Vector layer representing network	INPUT	[vector: line]	Line vector layer representing the network to be covered
Path type to calculate	STRATEGY	[enumeration] Default: 0	The type of path to calculate. One of: • 0 — Shortest • 1 — Fastest
Start point (x, y)	START_POINT	[coordinates]	Point feature representing the start point of the routes
Vector layer with end points	END_POINTS	[vector: point]	Point vector layer whose features are used as end points of the routes
Direction field Optional *Advanced*	DIRECTION_FIELD	[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.
Value for forward direction Optional *Advanced*	VALUE_FORWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a forward direction
Value for backward direction Optional *Advanced*	VALUE_BACKWARD	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify edges with a backward direction
Value for both directions Optional *Advanced*	VALUE_BOTH	[string] Default: ‚‘ (empty string)	Value set in the direction field to identify bidirectional edges
Default direction Optional *Advanced*	DEFAULT_DIRECTION	[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: • 0 — Forward direction • 1 — Backward direction • 2 — Both directions
Speed field Optional *Advanced*	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.

Fortsetzung auf der nächsten Seite

Tab. 23.18 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Default speed (km/h) Optional *Advanced*	DEFAULT_SPEED	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
Topology tolerance Optional *Advanced*	TOLERANCE	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected
Shortest path	OUTPUT	[vector: line]	Specify the output line layer for the shortest paths. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Shortest path	OUTPUT	[vector: line]	Line layer of the shortest or fastest path from each of the start points to the end point

Python code

Algorithm ID: qgis: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

Label	Name	Advanced	Type	Beschreibung
Vector layer representing network	INPUT		[vector: line]	Line vector layer representing the network to be covered
Path type to calculate	STRATEGY		[enumeration] Default: 0	The type of path to calculate. One of: • 0 — Shortest • 1 — Fastest
Start point (x, y)	START_POINT		[coordinates]	Point feature representing the start point of the routes

Fortsetzung auf der nächsten Seite

Tab. 23.19 – Fortsetzung der vorherigen Seite

Label	Name	Advanced	Type	Beschreibung
End point (x, y)	`END_POINT`		[coordinates]	Point feature representing the end point of the routes
Direction field Optional	`DIRECTION_FIELD`	X	[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.
Value for forward direction Optional	`VALUE_FORWARD`	X	[string] Default: ‚' (empty string)	Value set in the direction field to identify edges with a forward direction
Value for backward direction Optional	`VALUE_BACKWARD`	X	[string] Default: ‚' (empty string)	Value set in the direction field to identify edges with a backward direction
Value for both directions Optional	`VALUE_BOTH`	X	[string] Default: ‚' (empty string)	Value set in the direction field to identify bidirectional edges
Default direction Optional	`DEFAULT_DIRECTION`	X	[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: • 0 — Forward direction • 1 — Backward direction • 2 — Both directions
Speed field Optional	`SPEED_FIELD`	X	[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.
Default speed (km/h) Optional	`DEFAULT_SPEED`	X	[number] Default: 50.0	Value to use to calculate the travel time if no speed field is provided for an edge
Topology tolerance Optional	`TOLERANCE`	X	[number] Default: 0.0	Two lines with nodes closer than the specified tolerance are considered connected

Fortsetzung auf der nächsten Seite

Tab. 23.19 – Fortsetzung der vorherigen Seite

Label	Name	Advanced	Type	Beschreibung
Shortest path	`OUTPUT`		[vector: line]	Specify the output line layer for the shortest paths. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Shortest path	`OUTPUT`	[vector: line]	Line layer of the shortest or fastest path from each of the start point to the end point

Python code

Algorithm ID: `qgis: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.

23.1.9 Raster analysis

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

Label	Name	Type	Beschreibung
Input layers	INPUT	[raster] [list]	List of input raster layers
Reference layer	REF_LAYER	[raster]	The reference layer to create the output layer from (extent, CRS, pixel dimensions)
Treat nodata values as false	NODATA_AS_FALSE	[boolean] Default: False	Treat nodata values in the input files as 0 when performing the operation
Output no data value	NO_DATA	[number] Default: -9999.0	Value to use for nodata in the output layer
Output data type	DATA_TYPE	[enumeration] Default: 5	Output raster data type. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Output layer	OUTPUT	[raster]	Output raster layer

Ausgaben

Label	Name	Type	Beschreibung
Extent	EXTENT	[extent]	The extent of the output raster layer
CRS authority identifier	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
Width in pixels	WIDTH_IN_PIXELS	[integer]	The width in pixels of the output raster layer
Height in pixels	HEIGHT_IN_PIXELS	[integer]	The height in pixels of the output raster layer
Total pixel count	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer
NODATA pixel count	NODATA_PIXEL_COUNT	[integer]	The count of nodata pixels in the output raster layer
True pixel count	TRUE_PIXEL_COUNT	[integer]	The count of True pixels (value = 1) in the output raster layer
False pixel count	FALSE_PIXEL_COUNT	[integer]	The count of False pixels (value = 0) in the output raster layer
Output layer	OUTPUT	[raster]	Output raster layer containing the result

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layers	`INPUT`	[raster] [list]	List of input raster layers
Reference layer	`REF_LAYER`	[raster]	The reference layer to create the output layer from (extent, CRS, pixel dimensions)
Treat nodata values as false	`NODATA_AS_FALSE`	[boolean] Default: False	Treat nodata values in the input files as 0 when performing the operation
Output no data value	`NO_DATA`	[number] Default: -9999.0	Value to use for nodata in the output layer
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Output raster data type. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Output layer	`OUTPUT`	[raster]	Output raster layer

Ausgaben

Label	Name	Type	Beschreibung
Extent	EXTENT	[extent]	The extent of the output raster layer
CRS authority identifier	CRS_AUTHID	[crs]	The coordinate reference system of the output raster layer
Width in pixels	WIDTH_IN_PIXELS	[integer]	The width in pixels of the output raster layer
Height in pixels	HEIGHT_IN_PIXELS	[integer]	The height in pixels of the output raster layer
Total pixel count	TOTAL_PIXEL_COUNT	[integer]	The count of pixels in the output raster layer
NODATA pixel count	NODATA_PIXEL_COUNT	[integer]	The count of nodata pixels in the output raster layer
True pixel count	TRUE_PIXEL_COUNT	[integer]	The count of True pixels (value = 1) in the output raster layer
False pixel count	FALSE_PIXEL_COUNT	[integer]	The count of False pixels (value = 0) in the output raster layer
Output layer	OUTPUT	[raster]	Output raster layer containing the result

Python code

Algorithm ID: qgis: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.

Bemerkung: When using the calculator in *Die Batch Processing Schnittstelle* or from the *QGIS Python Konsole* the files to use have to be specified. The corresponding layers are referred using the base name of the file (without the full path). For instance, if using a layer at `path/to/my/rasterfile.tif`, the first band of that layer will be referred as `rasterfile.tif@1`.

Parameter

Label	Name	Type	Beschreibung
Layers	GUI only		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	GUI only		Contains some calculator like buttons that can be used to fill the expression box.
Expression	`EXPRESSION`	[string]	Expression that will be used to calculate the output raster layer. You can use the operator buttons provided to type directly the expression in this box.
Predefined expressions	GUI only		You can use the predefined `NDVI` expression or you can define new expressions for calculations. The *Add…* button loads a defined expression (and lets you set the parameters). The *Save…* button lets you define a new expression.
Reference layer(s) (used for automated extent, cellsize, and CRS) Optional	`LAYERS`	[raster] [list]	Layer(s) that will be used to fetch extent, cell size and CRS. By choosing the layer in this box you avoid filling in all the other parameters by hand. 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`.
Cell size (use 0 or empty to set it 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.
Output extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	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.
Output CRS 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.
Ergebnis	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis	OUTPUT	[raster]	Output raster file with the calculated values.

Python code

Algorithm ID: qgis: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 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Band number	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.
Ergebnis	OUTPUT_HTML_FILE	[html] Default: [Save to temporary file]	Specification of the output file: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Maximum value	MAX	[number]	
Mean value	MEAN	[number]	
Minimum value	MIN	[number]	

Fortsetzung auf der nächsten Seite

Tab. 23.25 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Ergebnis	`OUTPUT_HTML_FILE`	[html]	The output file contains the following information: • Analyzed file: path of the raster layer • Minimum value: minimum value of the raster • Maximum value: maximum value of the raster • Range: difference between the maximum and minimum values • Sum: total sum of the values • Mean value: mean of the values • Standard deviation: standard deviation of the values • Sum of the squares: sum of the squared differences of each observation from the overall mean
Range	`RANGE`	[number]	
Standard deviation	`STD_DEV`	[number]	
Sum	`SUM`	[number]	
Sum of the squares	`SUM_OF_SQUARES`	[number]	

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input raster layer
Band number	`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.
Unique values report	`OUTPUT_HTML_FILE`	[file] Default: `[Save to temporary file]`	Specification of the output file: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 23.26 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Unique values table	`OUTPUT_TABLE`	[table] Default: `[Skip output]`	Specification of the table for unique values: • Skip Output • Create Temporary Layer • Save to File… • Save to GeoPackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
CRS authority identifier	`CRS_AUTHID`	[crs]	
Extent	`EXTENT`	[extent]	
Height in pixels	`HEIGHT_IN_PIXELS`	[number]	
NODATA pixel count	`NODATA_PIXEL_COUNT`	[number]	
Total pixel count	`TOTAL_PIXEL_COUNT`	[number]	
Unique values report	`OUTPUT_HTML_FILE`	[html]	The output HTML file contains the following information: • Analyzed file: the path of the raster layer • Extent: xmin, ymin, xmax, ymax coordinates of the extent • Projection: projection of the layer • Width in pixels: number of columns and pixel width size • Height in pixels: number of rows and pixel width size • Total pixel count: count of all the pixels • NODATA pixel count: count of pixels with NODATA value
Unique values table	`OUTPUT_TABLE`	[table]	A table with three columns: • *value*: pixel value • *count*: count of pixels with this value • m^2: total area in square meters of pixels with this value.
Width in pixels	`WIDTH_IN_PIXELS`	[number]	

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input Layer	INPUT	[raster]	Input raster layer
Band number	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.
Zones layer	ZONES	[raster]	Raster layer defining zones. Zones are given by contiguous pixels having the same pixel value.
Zones band number	ZONES_BAND	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that defines the zones
Reference layer 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: • 0 — Input layer • 1 — Zones layer
Statistics	OUTPUT_TABLE	[table]	Table with the calculated statistics

Ausgaben

Label	Name	Type	Beschreibung
CRS authority identifier	CRS_AUTHID	[crs]	
Extent	EXTENT	[extent]	
Height in pixels	HEIGHT_IN_PIXELS	[number]	
NODATA pixel count	NODATA_PIXEL_COUNT	[number]	

Fortsetzung auf der nächsten Seite

Tab. 23.29 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Statistics	`OUTPUT_TABLE`	[table]	The output layer contains the following information **for each zone**: • Area: the area in square raster units in the zone; • Sum: the total sum of the pixel values in the zone; • Count: the number of pixels in the zone; • Min: the minimum pixel value in the zone; • Max: the maximum pixel value in the zone; • Mean: the mean of the pixel values in the zone;
Total pixel count	`TOTAL_PIXEL_COUNT`	[number]	
Width in pixels	`WIDTH_IN_PIXELS`	[number]	

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
INPUT layer	`INPUT`	[raster]	Input raster, representing a surface
Band number	`BAND`	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band that shall define the surface.
Base level	`LEVEL`	[number] Default: 0.0	Define a base or reference value. This base is used in the volume calculation according to the `Method` parameter (see below).

Fortsetzung auf der nächsten Seite

Tab. 23.30 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Method	`METHOD`	[enumeration] Default: 0	Define the method for the volume calculation given by the difference between the raster pixel value and the `Base level`. Options: • 0 — Count Only Above Base Level: only pixels above the base level will add to the volume. • 1 — Count Only Below Base Level: only pixels below the base level will add to the volume. • 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. • 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.
Surface volume report	`OUTPUT_HTML_FILE`	[html] Default: `[Save to temporary file]`	Specification of the output HTML report. One of: • Skip output • Save to Temporary File • Save to File… The file encoding can also be changed here.
Surface volume table	`OUTPUT_TABLE`	[table] Default: `[Skip output]`	Specification of the output table. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Volume	`VOLUME`	[number]	The calculated volume
Area	`AREA`	[number]	The area in square map units
Pixel_count	`PIXEL_COUNT`	[number]	The total number of pixels that have been analyzed
Surface volume report	`OUTPUT_HTML_FILE`	[html]	The output report (containing volume, area and pixel count) in HTML format
Surface volume table	`OUTPUT_TABLE`	[table]	The output table (containing volume, area and pixel count)

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Raster layer to reclassify
Band number	`RASTER_BAND`	[raster band] Default: The first band of the raster layer	If the raster is multiband, choose the band you want to reclassify.
Layer containing class breaks	`INPUT_TABLE`	[vector: any]	Vector layer containing the values to use for classification.
Minimum class value field	`MIN_FIELD`	[tablefield: numeric]	Field with the minimum value of the range for the class.
Maximum class value field	`MAX_FIELD`	[tablefield: numeric]	Field with the maximum value of the range for the class.
Output value field	`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).
Output no data value	`NO_DATA`	[number] Default: -9999.0	Value to apply to no data values.
Range boundaries	`RANGE_BOUNDARIES`	[enumeration] Default: 0	Defines comparison rules for the classification. Options: • 0 — min < value <= max • 1 — min <= value < max • 2 — min <= value <= max • 3 — min < value < max
Use no data when no range matches value	`NODATA_FOR_MISSING`	[boolean] Default: False	Values that do not belong to a class will result in the no data value. If False, the original value is kept.

Fortsetzung auf der nächsten Seite

Tab. 23.31 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Reclassified raster	`OUTPUT`	[raster]	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Reclassified raster	`OUTPUT`	[raster]	Output raster layer with reclassified band values

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Raster layer to reclassify
Band number	`RASTER_BAND`	[raster band] Default: 1	Raster band for which you want to recalculate values.
Reclassification table	`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.

Fortsetzung auf der nächsten Seite

Tab. 23.32 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output no data value	`NO_DATA`	[number] Default: -9999.0	Value to apply to no data values.
Range boundaries	`RANGE_BOUNDARIES`	[enumeration] Default: 0	Defines comparison rules for the classification. Options: • 0 — min < value <= max • 1 — min <= value < max • 2 — min <= value <= max • 3 — min < value < max
Use no data when no range matches value	`NODATA_FOR_MISSING`	[boolean] Default: False	Applies the no data value to band values that do not fall in any class. If False, the original value is kept.
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Reclassified raster	`OUTPUT`	[raster] Default: ‚[Save to temporary file]'	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here

Ausgaben

Label	Name	Type	Beschreibung
Reclassified raster	`OUTPUT`	[raster] Default: ‚[Save to temporary file]'	The output raster layer.

Python code

Algorithm ID: `qgis: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.

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

Label	Name	Type	Beschreibung
Input Point Layer	`INPUT`	[vector: point]	Point vector layer to use for sampling
Raster Layer to sample	`RASTERCOPY`	[raster]	Raster layer to sample at the given point locations.
Output column prefix	`COLUMN_PREFIX`	[string] Default: ‚rvalue'	Prefix for the names of the added columns.
Sampled Points (Optional)	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output layer containing the sampled values. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to GeoPackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Sampled Points (Optional)	`OUTPUT`	[vector: point]	The output layer containing the sampled values.

Python code

Algorithm ID: `qgis: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).

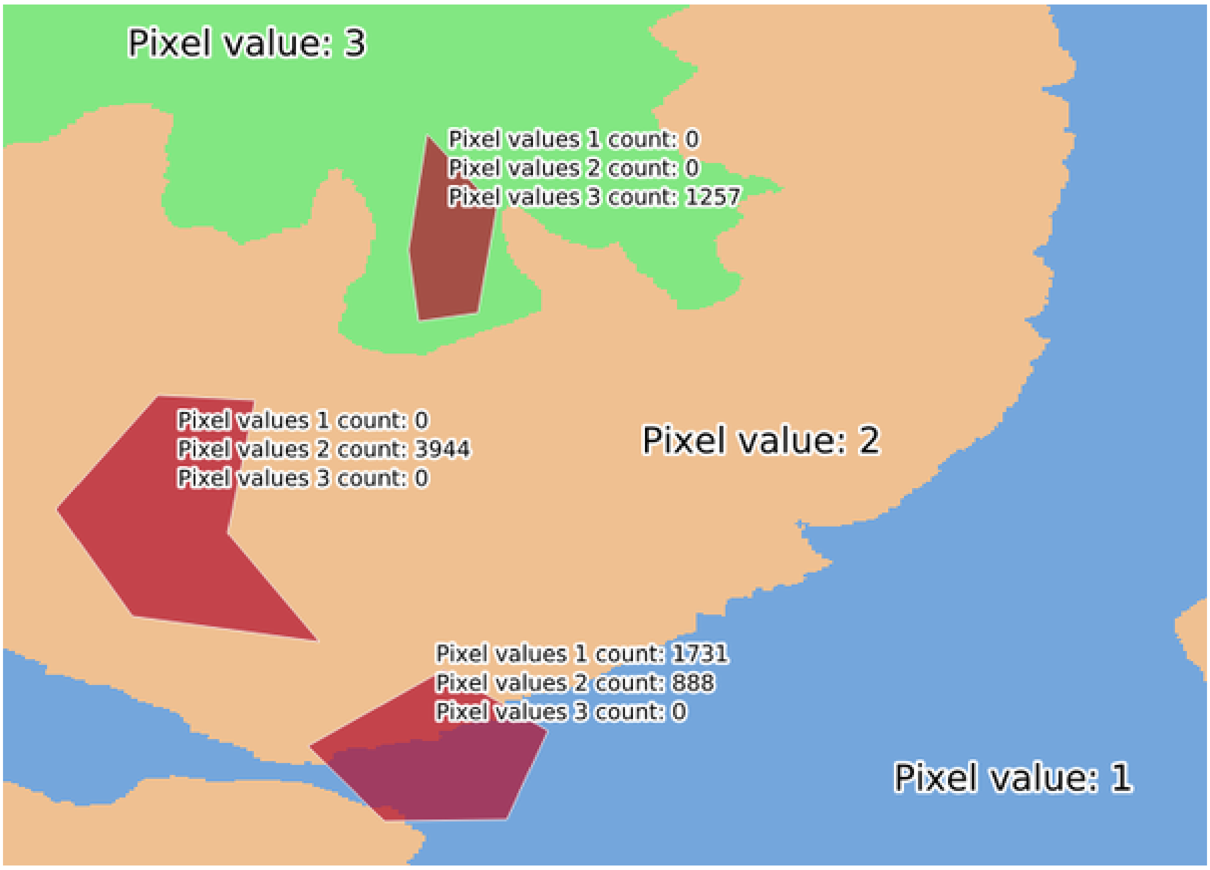


Abb. 23.7: Raster layer histogram example

Parameter

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Input raster layer.
Band number	`RASTER_BAND`	[raster band] Default: The first band of the input layer	If the raster is multiband, choose a band.
Vector layer containing zones	`INPUT_VECTOR`	[vector: polygon]	Vector polygon layer that defines the zones.
Output column prefix	`COLUMN_PREFIX` Optional	[string] Default: ‚HISTO_'	Prefix for the output columns names.
Output zones	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector polygon layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to GeoPackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output zones (Optional)	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	The output vector polygon layer.

Python code

Algorithm ID: `qgis: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.

> **Warnung:** No new output file will be created. The algorithm adds new columns to the source vector layer.

Parameter

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Input raster layer.
Raster band	`RASTER_BAND`	[raster band] Default: The first band of the input layer	If the raster is multiband, choose a band for the statistics.
Vector layer containing zones	`INPUT_VECTOR`	[vector: polygon]	Vector polygon layer that defines the zones.
Output column prefix	`COLUMN_PREFIX`	[string] Default: ‚_‘	Prefix for the output columns names.

Fortsetzung auf der nächsten Seite

Tab. 23.33 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Statistics to calculate	`STATISTICS`	[enumeration] [list] Default: [0,1,2]	List of statistical operator for the output. Options: • 0 — Count • 1 — Sum • 2 — Mean • 3 — Median • 4 — St. dev. • 5 — Minimum • 6 — Maximum • 7 — Range • 8 — Minority • 9 — Majority • 10 — Variety • 11 — Variance

Ausgaben

Label	Name	Type	Beschreibung
Vector layer containing zones	`INPUT_VECTOR`	[vector: polygon]	The input zone vector layer with added statistics.

Python code

Algorithm ID: `qgis:zonalstatistics`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.1.10 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.

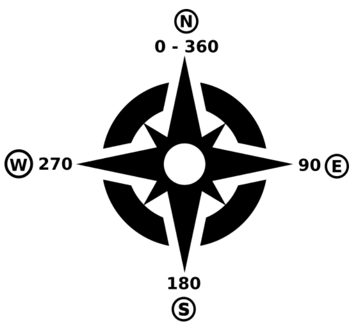


Abb. 23.8: Aspect values

The following picture shows the aspect layer reclassified with a color ramp:

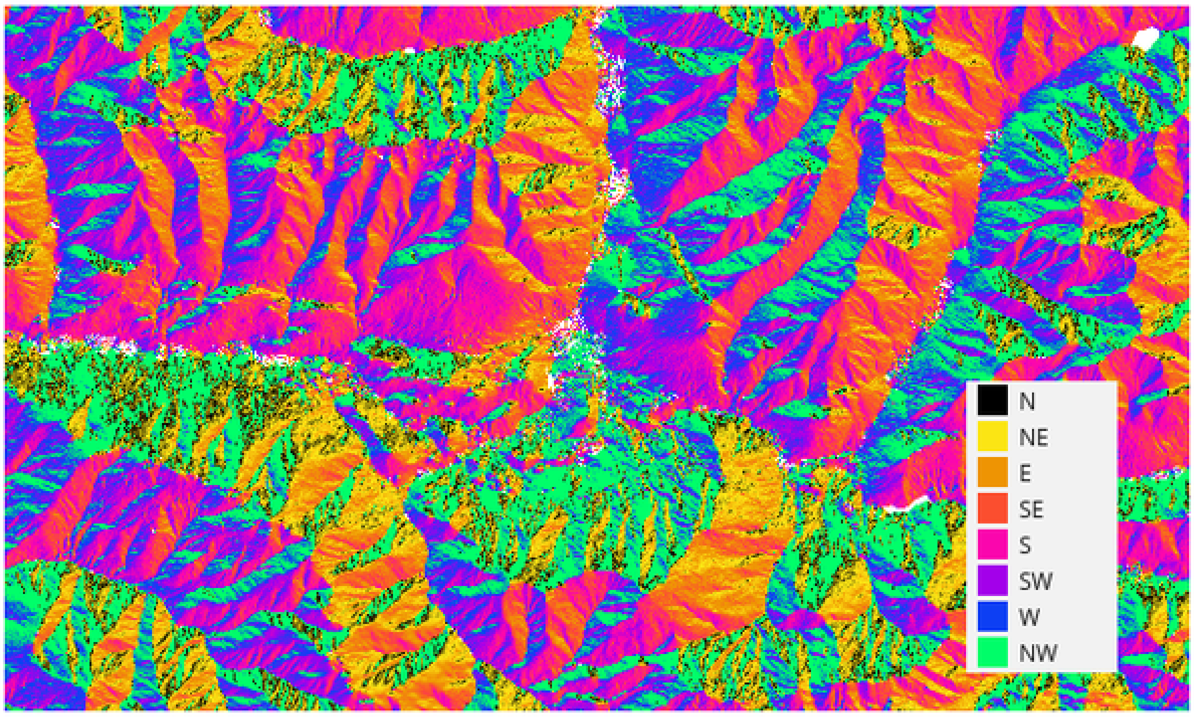


Abb. 23.9: Aspect layer reclassified

Parameter

Label	Name	Type	Beschreibung
Elevation layer	INPUT	[raster]	Digital Terrain Model raster layer
Z factor	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).
Aspect	OUTPUT	[raster]	Specify the output aspect raster layer. One of: • Save to a Temporary Layer (TEMPORARY_OUTPUT) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Aspect	OUTPUT	[raster]	The output aspect raster layer

Python code

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.

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.

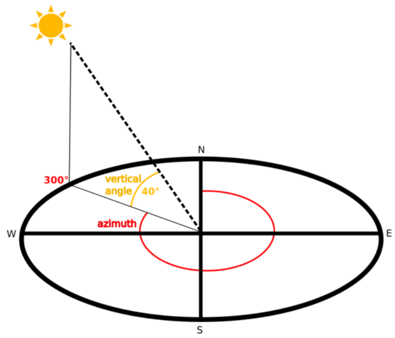


Abb. 23.10: Azimuth and vertical angle

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.

Particularly interesting is to give the hillshade layer a transparency value and overlap it with the elevation raster:

Parameter

Label	Name	Type	Beschreibung
Elevation layer	`INPUT`	[raster]	Digital Terrain Model raster layer
Z factor	`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).
Azimuth (horizontal angle)	`AZIMUTH`	[number] Default: 300.0	Set the horizontal angle (in degrees) of the sun (clockwise direction). Range: 0 to 360. 0 is north.
Vertical angle	`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).

Fortsetzung auf der nächsten Seite

Tab. 23.34 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Hillshade	`OUTPUT`	[raster]	Specify the output hillshade raster layer. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Hillshade	`OUTPUT`	[raster]	The output hillshade raster layer

Python code

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

Label	Name	Type	Beschreibung
DEM to analyze	`INPUT_DEM`	[raster]	Digital Terrain Model raster layer to use for calculating altitudes
Boundary layer	`BOUNDARY_LAYER`	[vector: polygon]	Polygon vector layer with boundaries of areas used to calculate hypsometric curves
Step	`STEP`	[number] Default: 100.0	Vertical distance between curves
Use % of area instead of absolute value	`USE_PERCENTAGE`	[boolean] Default: False	Write area percentage to "Area" field of the CSV file instead of the absolute area
Hypsometric curves	`OUTPUT_DIRECTORY`	[folder]	Specify the output folder for the hypsometric curves. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

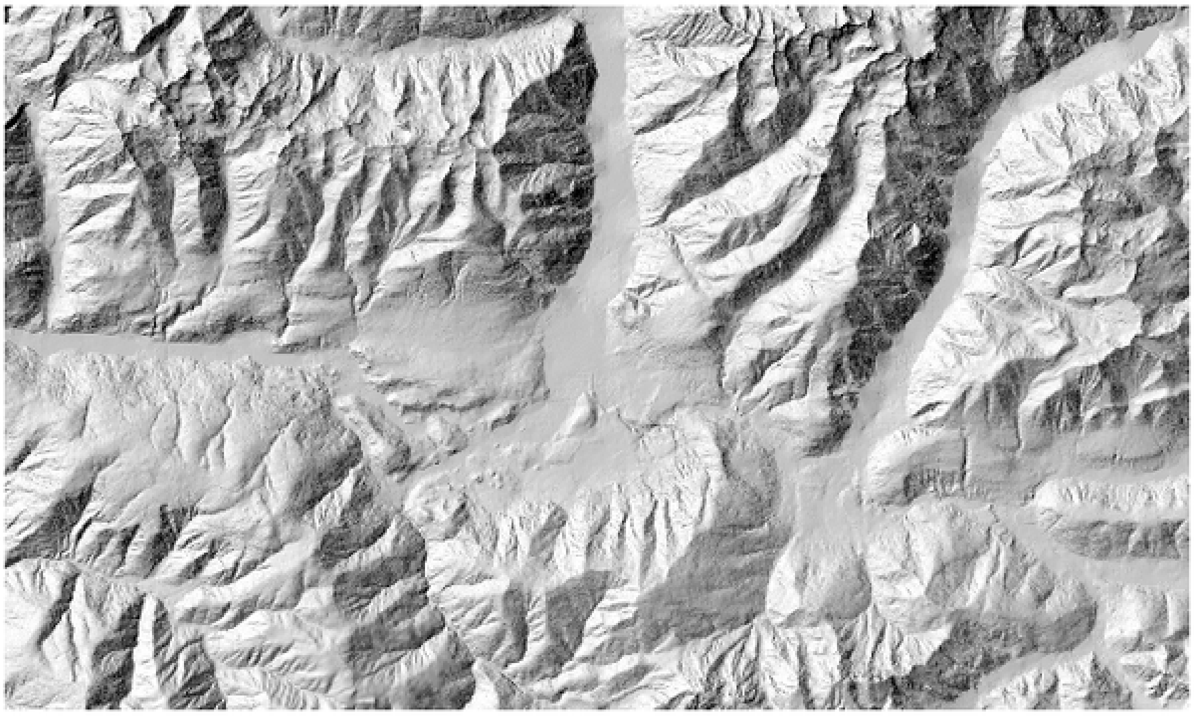


Abb. 23.11: Hillshade layer with azimuth 300 and vertical angle 45

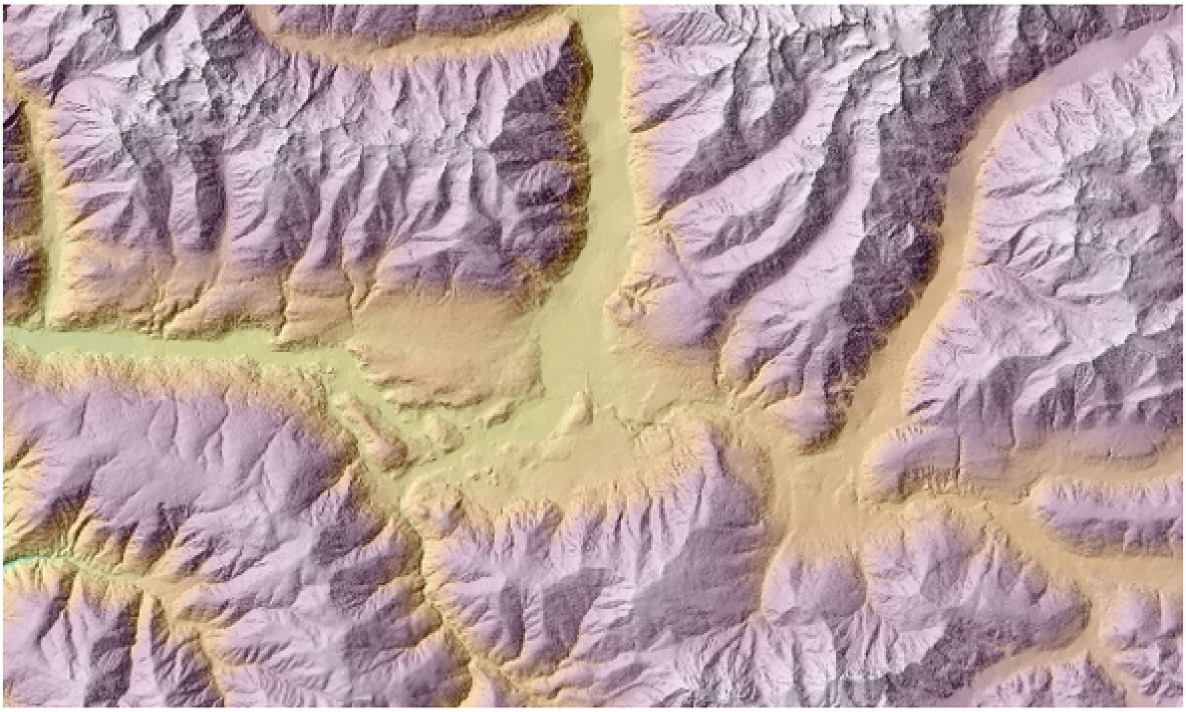
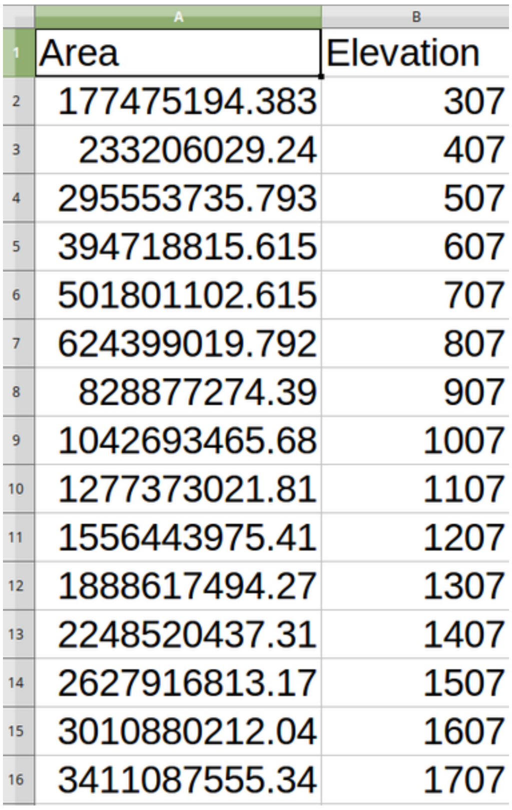


Abb. 23.12: Overlapping the hillshade with the elevation layer

Ausgaben

Label	Name	Type	Beschreibung
Hypsometric curves	`OUTPUT_DIRECTORY`	[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 `histogram_`, 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

Python code

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.

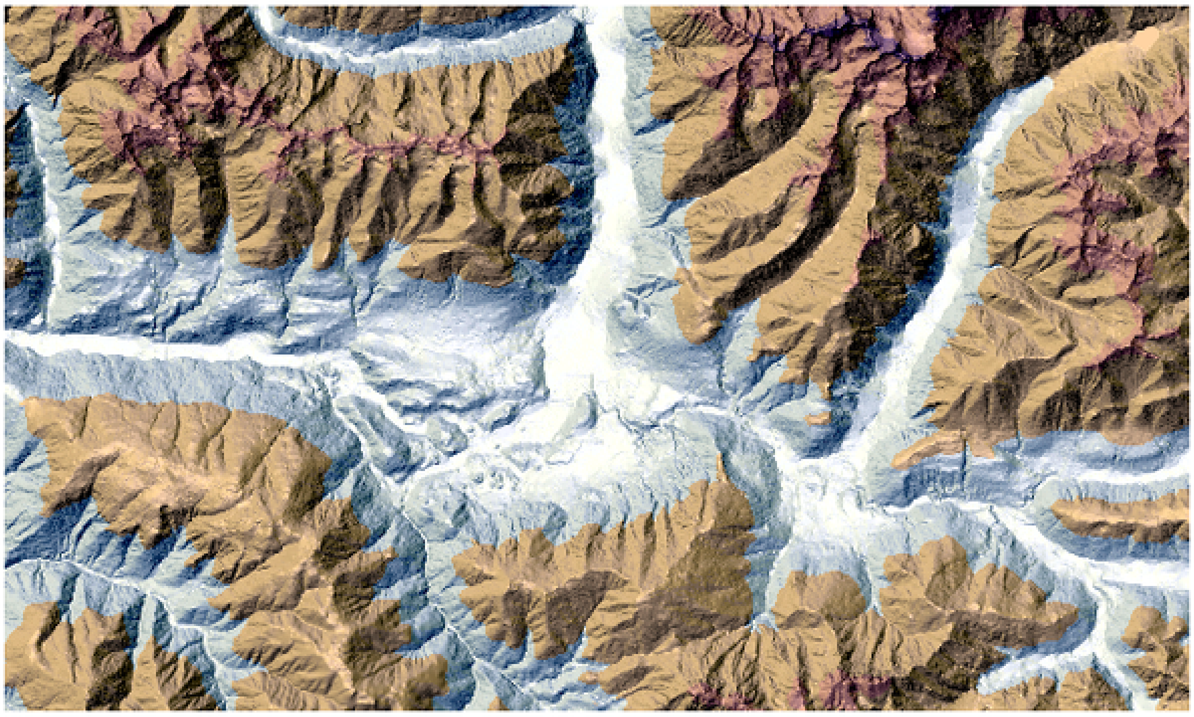


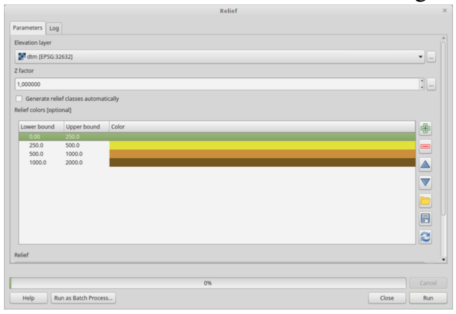
Abb. 23.13: Relief layer

Parameter

Label	Name	Type	Beschreibung
Elevation layer	INPUT	[raster]	Digital Terrain Model raster layer
Z factor	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).
Generate relief classes automatically	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. 23.36 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Relief colors 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.  Abb. 23.14: Manually setting of relief color classes The buttons in the right side panel give you the chance to: add or remove color classes, change the order of the color classes already defined, open an existing file with color classes and save the current classes as file.
Relief	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output relief raster layer. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.
Frequency distribution	`FREQUENCY_DISTRIBUTION`	[table] Default: `[Skip output]`	Specify the CSV table for the output frequency distribution. One of: • Skip Output • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Relief	`OUTPUT`	[raster]	The output relief raster layer
Frequency distribution	`OUTPUT`	[table]	The output frequency distribution

Python code

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.

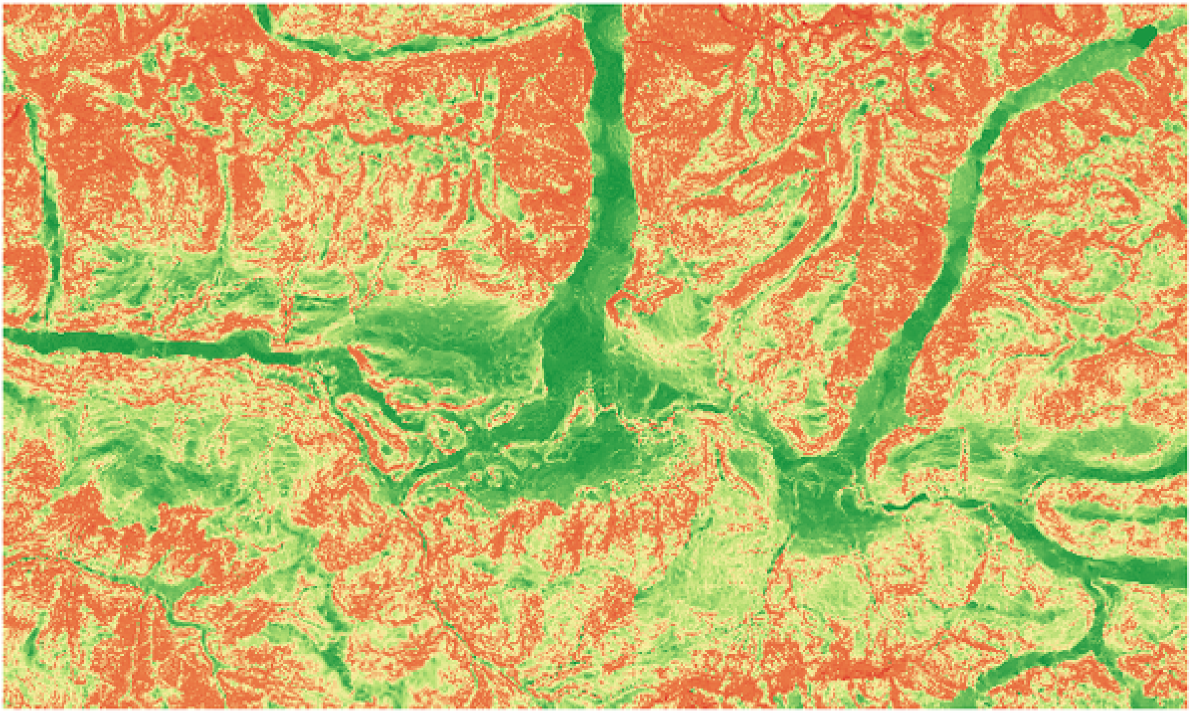


Abb. 23.15: Ruggedness layer from low (red) to high values (green)

Parameter

Label	Name	Type	Beschreibung
Elevation layer	`INPUT`	[raster]	Digital Terrain Model raster layer
Z factor	`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).
Ruggedness	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output ruggedness raster layer. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ruggedness	`OUTPUT`	[raster]	The output ruggedness raster layer

Python code

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**.

In the following picture you can see to the left the DTM layer with the elevation of the terrain while to the right the calculated slope:

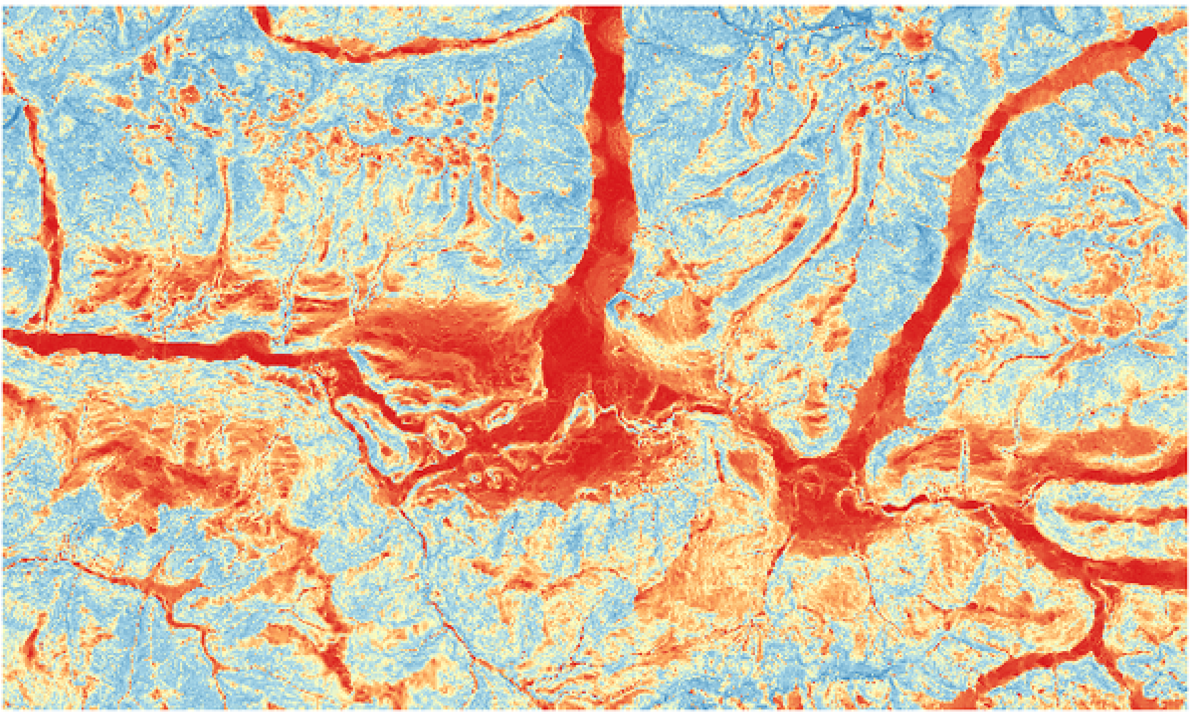


Abb. 23.16: Flat areas in red, steep areas in blue

Parameter

Label	Name	Type	Beschreibung
Elevation layer	`INPUT`	[raster]	Digital Terrain Model raster layer
Z factor	`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).
Slope	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output slope raster layer. One of: • Save to a Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Slope	OUTPUT	[raster]	The output slope raster layer

Python code

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.

23.1.11 Raster tools

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

Label	Name	Type	Beschreibung
Minimum extent to render (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Specify the extent of the output raster layer. One of: • Use Canvas Extent • Select Extent on Canvas • Use Layer Extent… It will internally be extended to a multiple of the tile size.
Tile size	TILE_SIZE	[number] Default: 1024	Size of the tile of the output raster layer. Minimum value: 64.
Map units per pixel	MAP_UNITS_PER_PIXEL	[number] Default: 100.0	Pixel size (in map units). Minimum value: 0.0
Make background transparent	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`.
Map theme to render Optional	MAP_THEME	[enumeration]	Use an existing *map theme* for the rendering.
Single layer to render Optional	LAYER	[enumeration]	Choose a single layer for the rendering
Output layer	OUTPUT	[raster] Default: Save to temporary file	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output layer	OUTPUT	[raster]	Output raster layer

Python code

Algorithm ID: `qgis: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.

Create constant raster layer

Generates a raster layer where all pixels have the same value.

Parameter

Label	Name	Type	Beschreibung
Desired extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	Specify the extent of the output raster layer. One of: • Use Canvas Extent • Select Extent on Canvas • Use Layer Extent… It will internally be extended to a multiple of the tile size.
Target CRS	`TARGET_CRS`	[crs] Default: Project CRS	CRS for the output raster layer
Pixel size	`PIXEL_SIZE`	[number] Default: 0.1	Pixel size (X=Y) in map units. Minimum value: 0.01
Constant value	`NUMBER`	[number] Default: 1	Constant pixel value for the output raster layer.
Constant	`OUTPUT`	[raster]	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Constant	`OUTPUT`	[raster]	Raster covering the desired extent with the specified pixel size and value.

Python code

Algorithm ID: `qgis: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.

Generate XYZ tiles (Directory)

Generates raster "XYZ" tiles using the current QGIS project as individual images to a directory structure.

Parameter

Label	Name	Type	Beschreibung
Extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	Specify the extent of the tiles. One of: • Use Canvas Extent • Select Extent on Canvas • Use Layer Extent… It will internally be extended to a multiple of the tile size.
Minimum zoom	`ZOOM_MIN`	[number] Default: 12	Minimum 0, maximum 25.
Maximum zoom	`ZOOM_MAX`	[number] Default: 12	Minimum 0, maximum 25.
DPI	`DPI`	[number] Default: 96	Minimum 48, maximum 600.
Background color Optional	`BACKGROUND_COLOR`	[color] Default: QColor(0, 0, 0, 0)	Choose the background color for the tiles
Tile format	`TILE_FORMAT`	[enumeration] Default: 0	One of: • 0 — PNG • 1 — JPG
Quality (JPG only) Optional	`QUALITY`	[number] Default: 75	Minimum 1, maximum 100.
Metatile size 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.
Tile width Optional	`TILE_WIDTH`	[number] Default: 256	Minimum 1, maximum 4096.
Tile height Optional	`TILE_HEIGHT`	[number] Default: 256	Minimum 1, maximum 4096.
Use inverted tile Y axis (TMS conventions) Optional	`TMS_CONVENTION`	[boolean] Default: False	
Output directory	`OUTPUT_DIRECTORY`	[folder] Default: `[Save to temporary folder]`	Specification of the output raster. One of: • Skip Output • Save to a Temporary Directory • Save to Directory… The file encoding can also be changed here.
Output html (Leaflet)	`OUTPUT_HTML`	[html] Default: `[Save to temporary file]`	Specification of the output HTML file. One of: • Skip Output • Save to a Temporary File • Save to File…

Ausgaben

Label	Name	Type	Beschreibung
Output directory	OUTPUT_DIRECTORY	[folder]	Output directory (for the tiles)
Output html (Leaflet)	OUTPUT_HTML	[html]	The output HTML (Leaflet) file

Python code

Algorithm ID: `qgis:tilesxyzdirectory`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Extent (xmin, xmax, ymin, ymax)	EXTENT	[extent]	Specify the extent of the tiles. One of: • Use Canvas Extent • Select Extent on Canvas • Use Layer Extent… It will internally be extended to a multiple of the tile size.
Minimum zoom	ZOOM_MIN	[number] Default: 12	Minimum 0, maximum 25.
Maximum zoom	ZOOM_MAX	[number] Default: 12	Minimum 0, maximum 25.
DPI	DPI	[number] Default: 96	Minimum 48, maximum 600.
Background color Optional	BACKGROUND_COLOR	[color] Default: QColor(0, 0, 0, 0)	Choose the background color for the tiles
Tile format	TILE_FORMAT	[enumeration] Default: 0	One of: • 0 — PNG • 1 — JPG
Quality (JPG only) Optional	QUALITY	[number] Default: 75	Minimum 1, maximum 100.

Fortsetzung auf der nächsten Seite

Tab. 23.39 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Metatile size 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.
Output file (for MBTiles)	`OUTPUT_FILE`	[file] Default: `[Save to temporary file]`	Specification of the output file. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output file (for MBTiles)	`OUTPUT_FILE`	[file]	The output file.

Python code

Algorithm ID: `qgis:tilesxyzmbtiles`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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 style for raster layer

Sets the style of a raster layer. The style must be defined as a `QML` file.

No new output are created: the `QML` style is assigned to the raster layer chosen.

Siehe auch:

Set style for vector layer

Parameter

Label	Name	Type	Beschreibung
Raster layer	`INPUT`	[raster]	The raster layer
Style file	`STYLE`	[file]	Path to the `QML` style file.

Ausgaben

Label	Name	Type	Beschreibung
Raster layer	`INPUT`	[raster]	The raster layer with the chosen style

Python code

Algorithm ID: `qgis:setstyleforrasterlayer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.1.12 Vektoranalyse

Grundstatistik für Felder

Erzeugt Grunstatistiken 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

Label	Name	Type	Beschreibung
Input vector	`INPUT_LAYER`	[vector: any]	Vector layer to calculate the statistics on
Field to calculate statistics on	`FIELD_NAME`	[tablefield: any]	Any supported table field to calculate the statistics
Statistics	`OUTPUT_HTML_FILE`	[html]	HTML file for the calculated statistics

Ausgaben

Label	Name	Type	Beschreibung
Statistics	`OUTPUT_HTML_FILE`	[html]	HTML file with the calculated statistics
Count	`COUNT`	[number]	
Number of unique values	`UNIQUE`	[number]	
Number of empty (null) values	`EMPTY`	[number]	
Number of non-empty values	`FILLED`	[number]	
Minimum value	`MIN`	[same as input]	
Maximum value	`MAX`	[same as input]	
Minimum length	`MIN_LENGTH`	[number]	
Maximum length	`MAX_LENGTH`	[number]	
Mean length	`MEAN_LENGTH`	[number]	

Fortsetzung auf der nächsten Seite

Tab. 23.40 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Coefficient of Variation	`CV`	[number]	
Sum	`SUM`	[number]	
Mean value	`MEAN`	[number]	
Standard deviation	`STD_DEV`	[number]	
Range	`RANGE`	[number]	
Median	`MEDIAN`	[number]	
Minority (rarest occurring value)	`MINORITY`	[same as input]	
Majority (most frequently occurring value)	`MAJORITY`	[same as input]	
First quartile	`FIRSTQUARTILE`	[number]	
Third quartile	`THIRDQUARTILE`	[number]	
Interquartile Range (IQR)	`IQR`	[number]	

Python code

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

Label	Name	Type	Beschreibung
Line layer	`INPUT`	[vector: line]	Line layer to calculate the climb for. Must have Z values
Climb layer	`OUTPUT`	[vector: line]	The output (line) layer

Ausgaben

Label	Name	Type	Beschreibung
Climb layer	OUTPUT	[vector: line]	Line layer containing new attributes with the results from climb calculations.
Total climb	TOTALCLIMB	[number]	The sum of the climb for all the line geometries in the input layer
Total descent	TOTALDESCENT	[number]	The sum of the descent for all the line geometries in the input layer
Minimum elevation	MINELEVATION	[number]	The minimum elevation for the geometries in the layer
Maximum elevation	MAXELEVATION	[number]	The maximum elevation for the geometries in the layer

Python code

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.

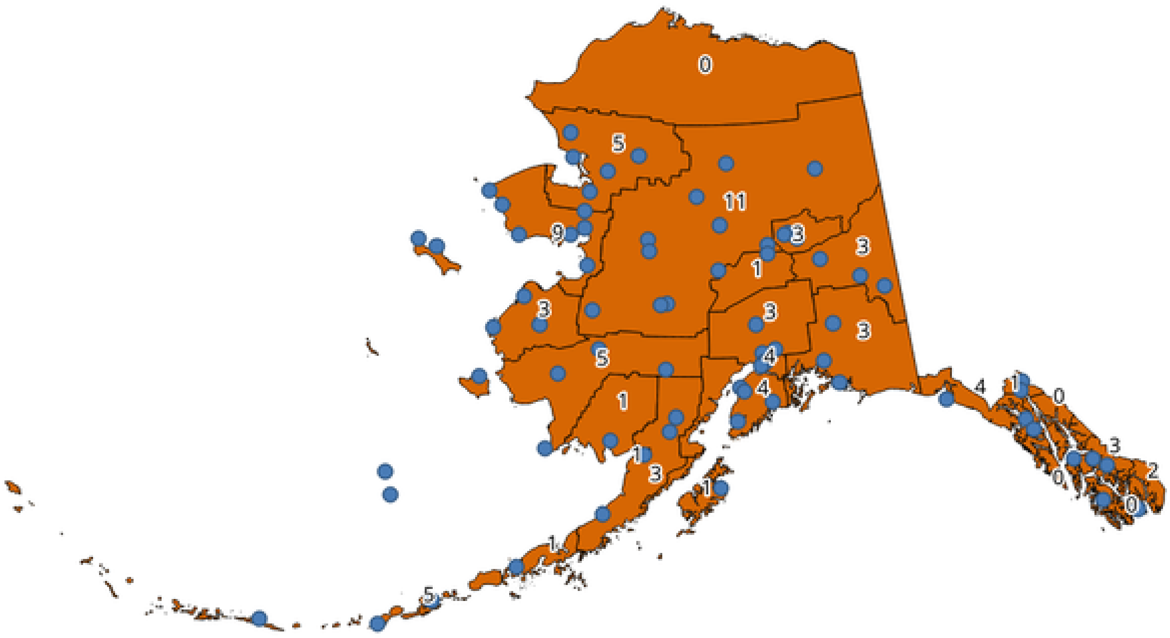


Abb. 23.17: The labels in the polygons show the point count

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.

`Default menu`: *Vector ▸ Analysis Tools*

Parameter

Label	Name	Type	Beschreibung
Polygons	`POLYGONS`	[vector: polygon]	Polygon layer whose features are associated with the count of points they contain
Points	`POINTS`	[vector: point]	Point layer with features to count
Weight field 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`.
Class field 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.
Count field name	`FIELD`	[string] Default: ‚NUMPOINTS'	The name of the field to store the count of points
Count	`OUTPUT`	[vector: polygon]	Specification of the output layer

Ausgaben

Label	Name	Type	Beschreibung
Count	`OUTPUT`	[vector: polygon]	Resulting layer with the attribute table containing the new column with the points count

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:

K-means clustering

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: point]	Layer to analyze
Minimum cluster size	MIN_SIZE	[number] Default: 5	Minimum number of features to generate a cluster
Maximum distance between clustered points	EPS	[number] Default: 1.0	Distance beyond which two features can not belong to the same cluster (eps)
Cluster field name	FIELD_NAME	[string] Default: ‚CLUSTER_ID‘	Name of the field where the associated cluster number shall be stored
Treat border points as noise (DBSCAN*) 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.
Clusters	OUTPUT	[vector: point]	Vector layer for the result of the clustering

Ausgaben

Label	Name	Type	Beschreibung
Clusters	OUTPUT	[vector: point]	Vector layer containing the original features with a field setting the cluster they belong to
Number of clusters	NUM_CLUSTERS	[number]	The number of clusters discovered

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input point layer	INPUT	[vector: point]	Point layer for which the distance matrix is calculated (**from** points)
Input unique ID field	INPUT_FIELD	[tablefield: any]	Field to use to uniquely identify features of the input layer. Used in the output attribute table.
Target point layer	TARGET	[vector: point]	Point layer containing the nearest point(s) to search (**to** points)
Target unique ID field	TARGET_FIELD	[tablefield: any]	Field to use to uniquely identify features of the target layer. Used in the output attribute table.
Output matrix type	MATRIX_TYPE	[enumeration] Default: 0	Different types of calculation are available: • 0 — Linear (N * *k* x 3) distance matrix: for each input point, reports the distance to each of the *k* nearest target points. The output matrix consists of up to *k* rows per input point, and each row has three columns: *InputID*, *TargetID* and *Distance*. • 1 — Standard (N x T) distance matrix • 2 — Summary distance matrix (mean, std. dev., min, max): for each input point, reports statistics on the distances to its target points.
Use only the nearest (k) target points	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.
Distance matrix	OUTPUT	[vector: point]	

Ausgaben

Label	Name	Type	Beschreibung
Distance matrix	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.

Python code

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.

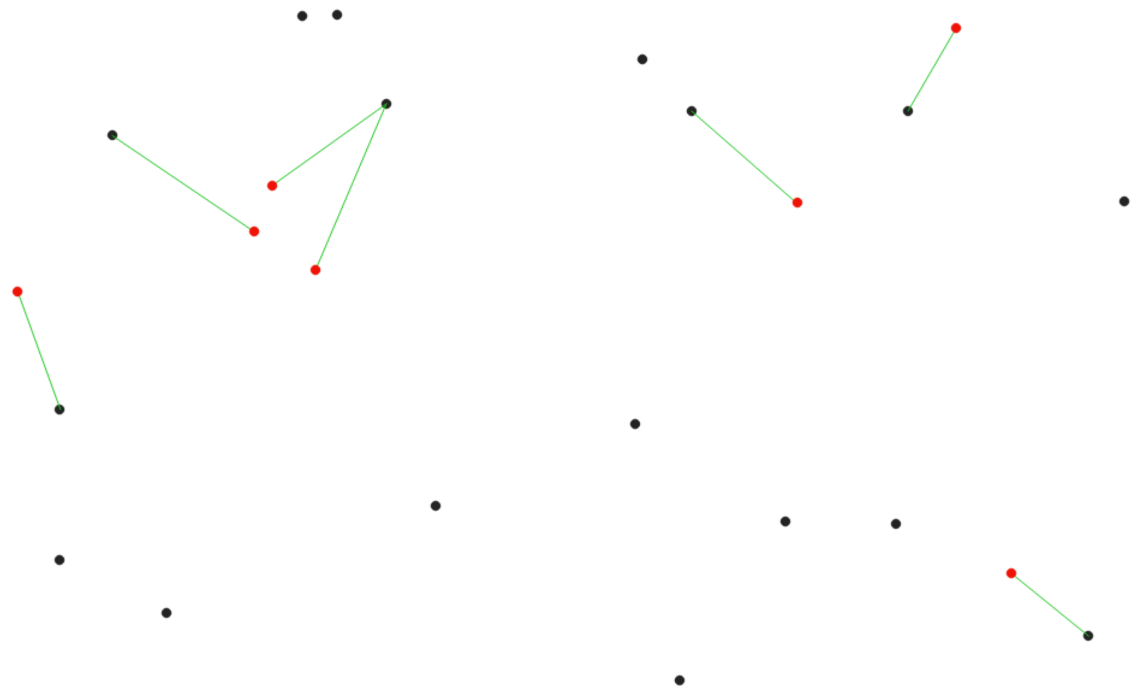


Abb. 23.18: Display the nearest hub for the red input features

Siehe auch:

Distance to nearest hub (points), *Join attributes by nearest*

Parameter

Label	Name	Type	Beschreibung
Source points layer	`INPUT`	[vector: any]	Vector layer for which the nearest feature is searched
Destination hubs layer	`HUBS`	[vector: any]	Vector layer containing the features to search for
Hub layer name attribute	`FIELD`	[tablefield: any]	Field to use to uniquely identify features of the destination layer. Used in the output attribute table
Measurement unit	`UNIT`	[enumeration] Default: 0	Units in which to report the distance to the closest feature: • 0 — Meters • 1 — Feet • 2 — Miles • 3 — Kilometers • 4 — Layer units
Hub distance	`OUTPUT`	[vector: line]	Line vector layer for the distance matrix output

Ausgaben

Label	Name	Type	Beschreibung
Hub distance	OUTPUT	[vector: line]	Line vector layer with the attributes of the input features, the identifier of their closest feature and the calculated distance.

Python code

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

Label	Name	Type	Beschreibung
Source points layer	INPUT	[vector: any]	Vector layer for which the nearest feature is searched
Destination hubs layer	HUBS	[vector: any]	Vector layer containing the features to search for
Hub layer name attribute	FIELD	[tablefield: any]	Field to use to uniquely identify features of the destination layer. Used in the output attribute table
Measurement unit	UNIT	[enumeration] Default: 0	Units in which to report the distance to the closest feature: • 0 — Meters • 1 — Feet • 2 — Miles • 3 — Kilometers • 4 — Layer units
Hub distance	OUTPUT	[vector: point]	Point vector layer for the distance matrix output.

Ausgaben

Label	Name	Type	Beschreibung
Hub distance	OUTPUT	[vector: point]	Point vector layer with the attributes of the input features, the identifier of their closest feature and the calculated distance.

Python code

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 (±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.

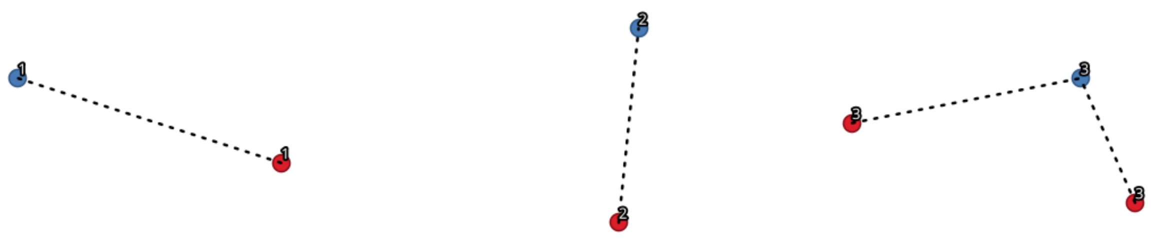


Abb. 23.19: Join points based on a common field / attribute

Parameter

Label	Name	Type	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) Optional	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.
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) Optional	SPOKE_FIELDS	[tablefield: any] [list]	Field(s) of the spoke layer to be copied. If no fields are chosen all fields are taken.
Create geodesic lines	GEODESIC	[boolean] Default: False	Create geodesic lines (the shortest path on the surface of an ellipsoid)
Distance between vertices (geodesic lines only)	GEODESIC_DISTANCE	[number] Default: 1000.0 (kilometers)	Distance between consecutive vertices (in kilometers). A smaller distance results in a denser, more accurate line
Split lines at antimeridian (±180 degrees longitude)	ANTIMERIDIAN_SPLIT	[boolean] Default: False	Split lines at ±180 degrees longitude (to improve rendering of the lines)
Hub lines	OUTPUT	[vector: line]	The resulting line layer

Ausgaben

Label	Name	Type	Beschreibung
Hub lines	OUTPUT	[vector: line]	The resulting line layer

Python code

Algorithm ID: qgis: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.

Siehe auch:

DBSCAN clustering

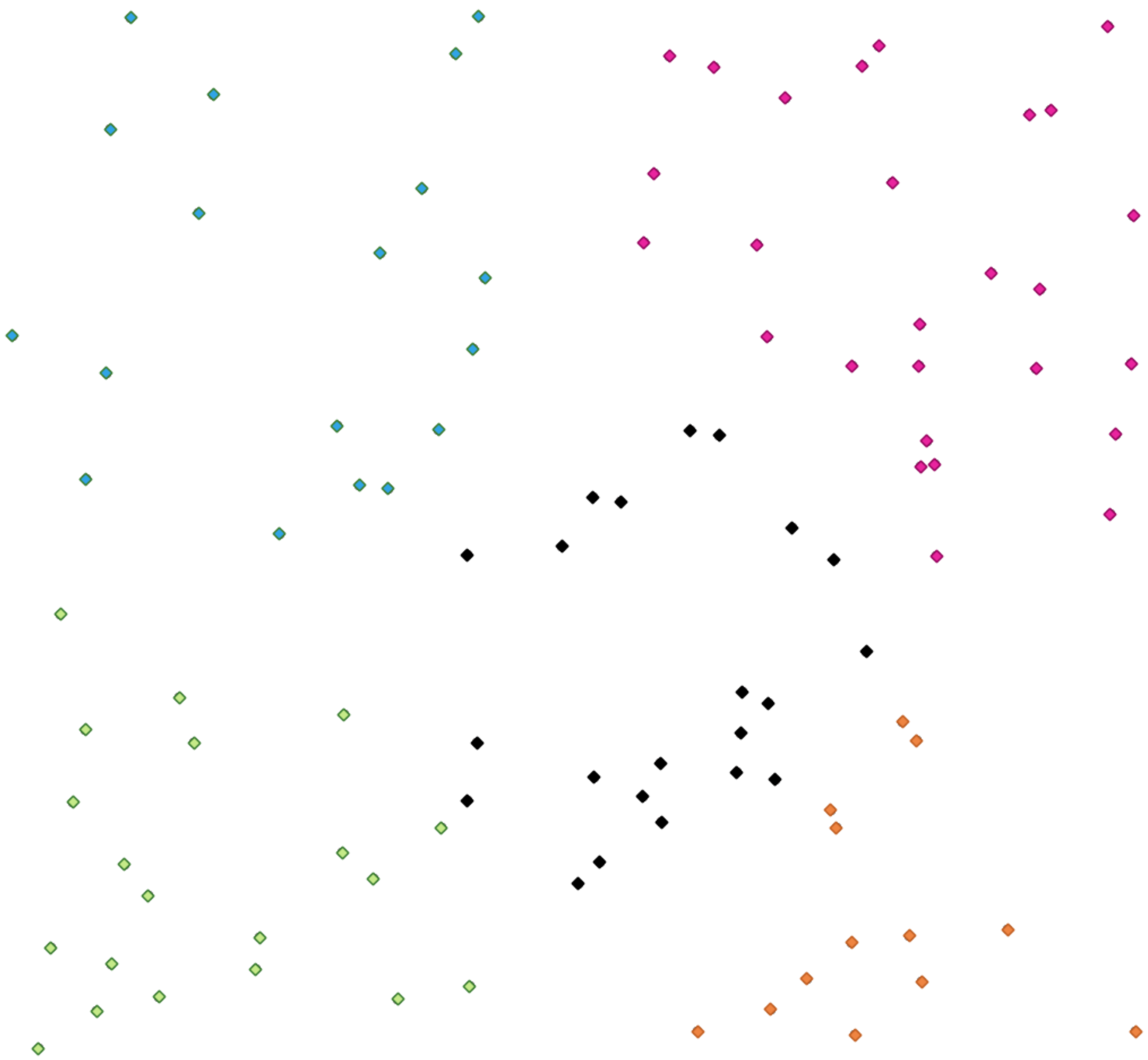


Abb. 23.20: A five class point clusters

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Layer to analyze
Number of clusters	CLUSTERS	[number] Default: 5	Number of clusters to create with the features
Cluster field name	FIELD_NAME	[string] Default: ‚CLUSTER_ID'	Name of the cluster number field
Clusters	OUTPUT	[vector: any]	Vector layer for generated the clusters

Ausgaben

Label	Name	Type	Beschreibung
Clusters	OUTPUT	[vector: any]	Vector layer containing the original features with a field specifying the cluster they belong to

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Layer to analyze
Target field(s)	FIELDS	[tablefield: any]	Field to analyze
Unique values	OUTPUT	[table]	Summary table layer with unique values
HTML report	OUTPUT_HTML_FILE	[html]	HTML report of unique values in the *Processing ► Results viewer*

Ausgaben

Label	Name	Type	Beschreibung
Unique values	`OUTPUT`	[table]	Summary table layer with unique values
HTML report	`OUTPUT_HTML_FILE`	[html]	HTML report of unique values. Can be opened from the *Processing ► Results viewer*
Total unique values	`TOTAL_VALUES`	[number]	The number of uniqe values in the input field
UNIQUE_VALUES	`Unique values`	[string]	A string with the comma separated list of unique values found in the input field

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Weight field Optional	`WEIGHT`	[tablefield: numeric]	Field to use if you want to perform a weighted mean
Unique ID field	`UID`	[tablefield: numeric]	Unique field on which the calculation of the mean will be made
Mean coordinates	`OUTPUT`	[vector: point]	The (point vector) layer for the result

Ausgaben

Label	Name	Type	Beschreibung
Mean coordinates	OUTPUT	[vector: point]	Resulting point(s) layer

Python code

Algorithm ID: qgis: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.

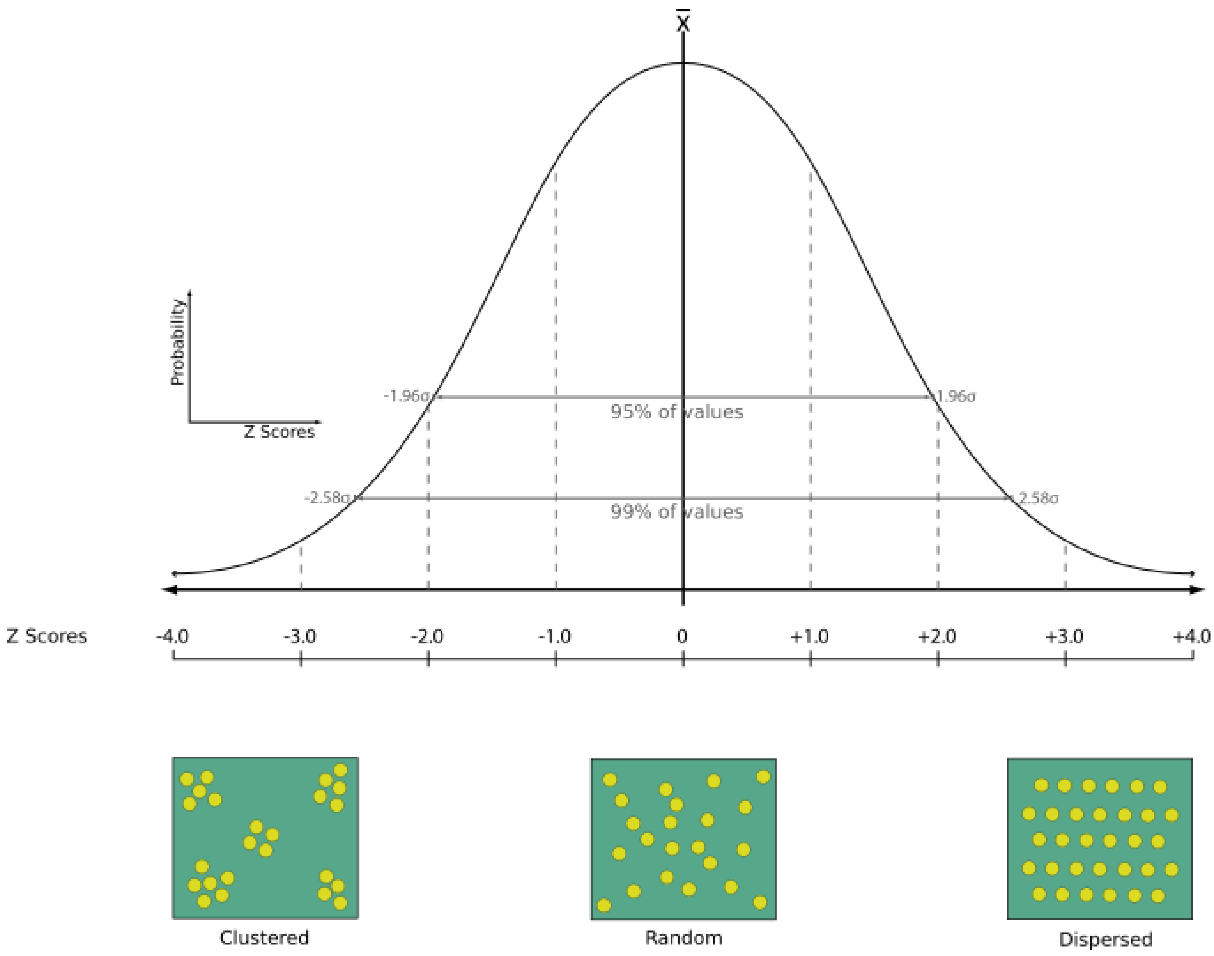
Default menu: *Vector* ▸ *Analysis Tools*

Siehe auch:

Join attributes by nearest

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: point]	Point vector layer to calculate the statistics on
Nearest neighbour	OUTPUT_HTML_FILE	[html]	HTML file for the computed statistics


X̄
Probability
Z Scores
-1.96σ
1.96σ
95% of values
-2.58σ
2.58σ
99% of values
Z Scores
-4.0
-3.0
-2.0
-1.0
0
+1.0
+2.0
+3.0
+4.0
Clustered
Random
Dispersed

Ausgaben

Label	Name	Type	Beschreibung
Nearest neighbour	OUTPUT_HTML_FILE	[html]	HTML file with the computed statistics
Observed mean distance	OBSERVED_MD	[number]	Observed mean distance
Expected mean distance	EXPECTED_MD	[number]	Expected mean distance
Nearest neighbour index	NN_INDEX	[number]	Nearest neighbour index
Number of points	POINT_COUNT	[number]	Number of points
Z-Score	Z_SCORE	[number]	Z-Score

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input layer.
Overlap layers	LAYERS	[vector: any] [list]	The overlay layers.
Output layer	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output layer	`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.

Python code

Algorithm ID: `qgis:overlapanalysis`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input vector layer	`INPUT`	[vector: any]	Input vector layer with unique classes and values
Field to calculate statistics on (if empty, only count is calculated) Optional	`VALUES_FIELD_NAME`	[tablefield: any]	If empty only the count will be calculated
Field(s) with categories	`CATEGORIES_FIELD_NAME`	[vector: any] [list]	The fields that (combined) define the categories
Statistics by category	`OUTPUT`	[table]	Table for the generated statistics

Ausgaben

Label	Name	Type	Beschreibung
Statistics by category	`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`)	✔	✔	✔

Fortsetzung auf der nächsten Seite

Tab. 23.46 – Fortsetzung der vorherigen Seite

Statistics	Text	Numeric	Datum
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`)	✓		

Python code

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.

Default menu: *Vector ⍰ Analysis Tools*

Parameter

Label	Name	Type	Beschreibung
Lines	LINES	[vector: line]	Input vector line layer
Polygons	POLYGONS	[vector: polygon]	Polygon vector layer
Lines length field name	LEN_FIELD	[string] Default: ‚LENGTH'	Name of the field for the lines length
Lines count field name	COUNT_FIELD	[string] Default: ‚COUNT'	Name of the field for the lines count
Line length	OUTPUT	[vector: polygon]	The output polygon vector layer

Ausgaben

Label	Name	Type	Beschreibung
Line length	OUTPUT	[vector: polygon]	Polygon output layer with fields of lines length and line count

Python code

Algorithm ID: `qgis: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.

23.1.13 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.

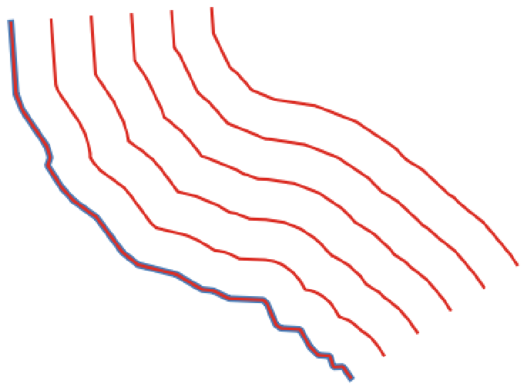


Abb. 23.21: In blue the source layer, in red the offset one

Allows *features in-place modification*

Siehe auch:

Offset lines, *Array of translated features*

Parameter

Label	Name	Type	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 step distance	`OFFSET`	[number] Default: 1.0	Distance between two consecutive offset copies
Segments	`SEGMENTS`	[number] Default: 8	Number of line segments to use to approximate a quarter circle when creating rounded offsets
Join style	`JOIN_STYLE`	[enumeration] Default: 0	Specify whether round, miter or beveled joins should be used when offsetting corners in a line. One of: • 0 — Round • 1 — Miter • 2 — Bevel
Miter limit	`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.
Offset lines	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output line layer with offset features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Offset lines	`OUTPUT`	[vector: line]	Output line layer with offset features. The original features are also copied.

Python code

Algorithm ID: `qgis: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.

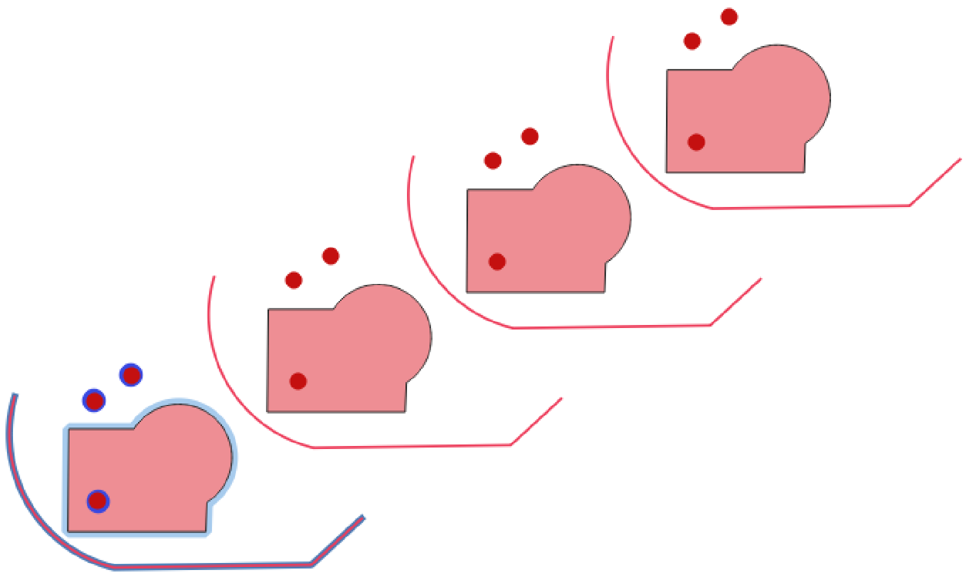


Abb. 23.22: Input layers in blue tones, output layers with translated features in red tones

Allows *features in-place modification*

Siehe auch:

Translate, *Array of offset (parallel) lines*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer to translate
Number of features to create	COUNT	[number] Default: 10	Number of copies to generate for each feature
Step distance (x-axis)	DELTA_X	[number] Default: 0.0	Displacement to apply on the X axis
Step distance (y-axis)	DELTA_Y	[number] Default: 0.0	Displacement to apply on the Y axis
Step distance (z-axis)	DELTA_Z	[number] Default: 0.0	Displacement to apply on the Z axis
Step distance (m values)	DELTA_M	[number] Default: 0.0	Displacement to apply on M

Fortsetzung auf der nächsten Seite

Tab. 23.49 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Translated	`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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Translated	`OUTPUT`	[same as input]	Output vector layer with translated (moved) copies of the features. The original features are also copied.

Python code

Algorithm ID: `qgis: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:

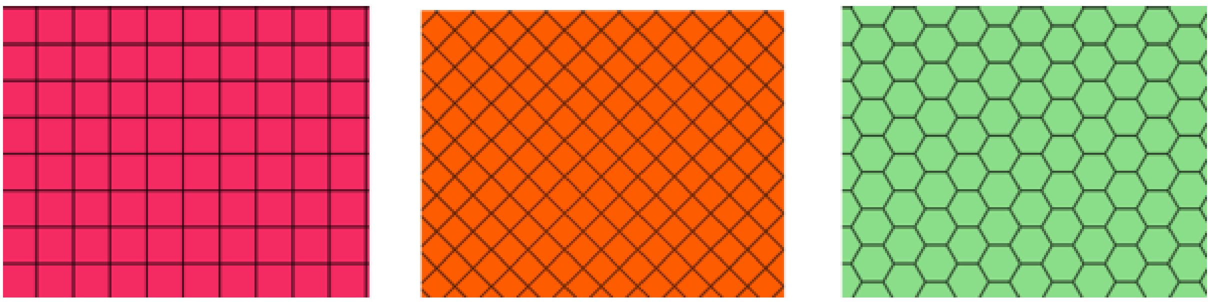


Abb. 23.23: Different grid cell shapes

The size 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

Label	Name	Type	Beschreibung
Grid type	TYPE	[enumeration] Default: 0	Shape of the grid. One of: • 0 — Point • 1 — Line • 2 — Rectangle (polygon) • 3 — Diamond (polygon) • 4 — Hexagon (polygon)
Grid extent	EXTENT	[extent]	Extent of the grid
Horizontal spacing	HSPACING	[number] Default: 1.0	Size of a grid cell on the X-axis
Vertical spacing	VSPACING	[number] Default: 1.0	Size of a grid cell on the Y-axis
Horizontal overlay	HOVERLAY	[number] Default: 0.0	Overlay distance between two consecutive grid cells on the X-axis
Vertical overlay	VOVERLAY	[number] Default: 0.0	Overlay distance between two consecutive grid cells on the Y-axis
Grid CRS	CRS	[crs] Default: *Project CRS*	Coordinate reference system to apply to the grid
Grid	OUTPUT	[vector: any] Default: `[Create temporary layer]`	Resulting vector grid layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Grid	OUTPUT	[vector: any]	Resulting vector grid layer. The output geometry type (point, line or polygon) depends on the *Grid type*.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer or a table.
X field	`XFIELD`	[tablefield: any]	Field containing the X coordinate
Y field	`YFIELD`	[tablefield: any]	Field containing the Y coordinate
Z field Optional	`ZFIELD`	[tablefield: any]	Field containing the Z coordinate
M field Optional	`MFIELD`	[tablefield: any]	Field containing the M value
Target CRS	`TARGET_CRS`	[crs] Default: `EPSG:4326`	Coordinate reference system to use for layer. The provided coordinates are assumed to be compliant.
Points from table	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the resulting point layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Points from table	`OUTPUT`	[vector: point]	The resulting point layer

Python code

Algorithm ID: `qgis: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.

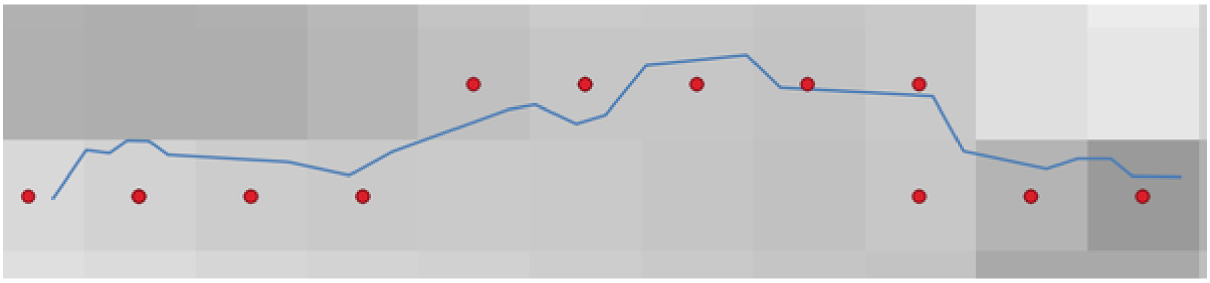


Abb. 23.24: Points of the pixel centroids

Parameter

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Input raster layer
Vector layer	`INPUT_VECTOR`	[vector: line]	Input line vector layer
Points along line	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Resulting point layer with pixel centroids. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Points along line	`OUTPUT`	[vector: point]	Resulting point layer with pixel centroids

Python code

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.

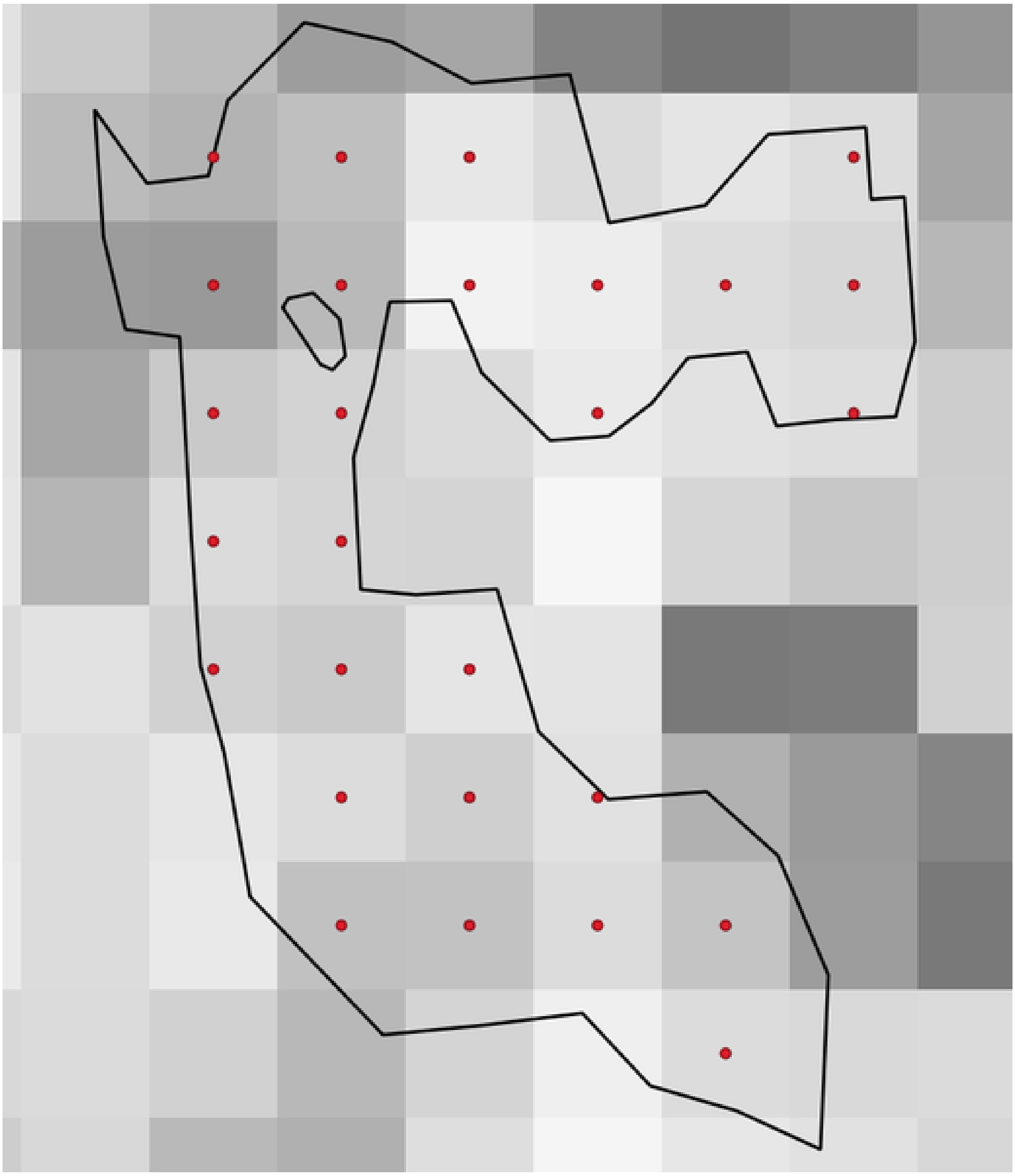


Abb. 23.25: Points of the pixel centroids

Parameter

Label	Name	Type	Beschreibung
Raster layer	INPUT_RASTER	[raster]	Input raster layer
Vector layer	INPUT_VECTOR	[vector: polygon]	Input polygon vector layer
Points inside polygons	OUTPUT	[vector: point] Default: [Create temporary layer]	Resulting point layer of pixel centroids. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Points inside polygons	OUTPUT	[vector: point]	Resulting point layer of pixel centroids

Python code

Algorithm ID: `qgis: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 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

Label	Name	Type	Beschreibung
Input folder	FOLDER	[folder]	Path to the source folder containing the geotagged photos
Scan recursively	RECURSIVE	[boolean] Default: False	If checked, the folder and its subfolders will be scanned

Fortsetzung auf der nächsten Seite

Tab. 23.52 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Photos	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the point vector layer for the geotagged photos. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Invalid photos table Optional	`INVALID`	[table] Default: `[Skip output]`	Specify the table of unreadable or non-geotagged photos. One of: • Skip Output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Photos	`OUTPUT`	[vector: point]	Point vector layer with geotagged photos. The form of the layer is automatically filled with paths and photo previews settings.
Invalid photos table Optional	`INVALID`	[table]	Table of unreadable or non-geotagged photos can also be created.

Python code

Algorithm ID: `qgis: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 a field in the input point layer (if the order field is a date/time field, the format must be specified).

Points can be grouped by a field 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

Label	Name	Type	Beschreibung
Input point layer	`INPUT`	[vector: point]	Input point vector layer
Order field	`ORDER_FIELD`	[tablefield: any]	Field containing the order to connect the points in the path
Group field Optional	`GROUP_FIELD`	[tablefield: any]	Point features of the same value in the field will be grouped in the same line. If not set, a single path is drawn with all the input points.
Date format (if order field is DateTime) Optional	`DATE_FORMAT`	[string]	The format to use for the `Order field` parameter. Specify this only if the `Order field` is of type Date/Time.
Paths	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the line vector layer of the path. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Directory for text output	`OUTPUT_TEXT_DIR`	[folder] Default: `[Skip output]`	Specify the directory that will contain the description files of points and paths. One of: • Skip Output • Save to a Temporary Directory • Save to Directory… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Paths	`OUTPUT`	[vector: line]	Line vector layer of the path
Directory for text output	`OUTPUT`	[folder]	Directory containing description files of points and paths

Python code

Algorithm ID: `qgis: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 in the lines of another layer.

For each line in the input layer, a given number of points is added to the resulting layer. A minimum distance can be specified, to avoid points being too close to each other.

Parameter

Label	Name	Type	Beschreibung
Input point layer	INPUT	[vector: line]	Input line vector layer
Number of points	POINTS_NUMBER	[number] Default: 1	Number of points to create
Minimum distance between points	MIN_DISTANCE	[number] Default: 0.0	The minimum distance between points
Random points	OUTPUT	[vector: point] Default: [Create temporary layer]	The output random points. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Random points	OUTPUT	[vector: point]	The output random points layer.

Python code

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 minimum distance can be specified, to avoid points being too close to each other.

Default menu: *Vector ▸ Research Tools*

Parameter

Label	Name	Type	Beschreibung
Input extent	`EXTENT`	[extent]	Map extent for the random points
Number of points	`POINTS_NUMBER`	[number] Default: 1	Number of point to create
Minimum distance between points	`MIN_DISTANCE`	[number] Default: 0.0	The minimum distance between points
Target CRS	`TARGET_CRS`	[crs] Default: *Project CRS*	CRS of the random points layer
Random points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	The output random points. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Random points	`OUTPUT`	[vector: point]	The output random points layer.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	Input polygon layer defining the area
Number of points	`POINTS_NUMBER`	[number] Default: 1	Number of points to create
Minimum distance between points	`MIN_DISTANCE`	[number] Default: 0.0	The minimum distance between points
Random points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	The output random points. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Random points	`OUTPUT`	[vector: point]	The output random points layer.

Python code

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 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*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	Input polygon vector layer
Sampling strategy	`STRATEGY`	[enumeration] Default: 0	Sampling strategy to use. One of: • 0 — Points count: number of points for each feature • 1 — Points density: density of points for each feature
Point count or density	`VALUE`	[number] Default: 1.0	The number or density of points, depending on the chosen *Sampling strategy*.
Minimum distance between points	`MIN_DISTANCE`	[number] Default: 0.0	The minimum distance between points
Random points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	The output random points. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Random points	`OUTPUT`	[vector: point]	The output random points layer.

Python code

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.

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

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Input raster layer
Band number	`RASTER_BAND`	[raster band]	Raster band to extract data from
Field name	`FIELD_NAME`	[string] Default: ‚VALUE'	Name of the field to store the raster band value
Vector points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the resulting point layer of pixels centroids. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Vector points	`OUTPUT`	[vector: point]	Resulting point layer with pixels centroids

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Raster layer	`INPUT_RASTER`	[raster]	Input raster layer
Band number	`RASTER_BAND`	[raster band]	Raster band to extract data from
Field name	`FIELD_NAME`	[string] Default: ‚VALUE‘	Name of the field to store the raster band value
Vector polygons	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the resulting polygon layer of pixel extents. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Vector polygons	`OUTPUT`	[vector: polygon]	Resulting polygon layer of pixel extents

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	Map extent for the random points
Point spacing/count	`SPACING`	[number] Default: 100	Spacing between the points, or the number of points, depending on whether `Use point spacing` is checked or not.
Initial inset from corner (LH side)	`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.
Apply random offset to point spacing	`RANDOMIZE`	[boolean] Default: False	If checked the points will have a random spacing
Use point spacing	`IS_SPACING`	[boolean] Default: True	If unchecked the point spacing is not taken into account
Output layer CRS	`CRS`	[crs] Default: *Project CRS*	CRS of the random points layer
Regular points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output regular point layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Regular points	`OUTPUT`	[vector: point]	The output regular point layer.

Python code

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.

23.1.14 Vector general

Projektion zuweisen

Assigns a new projection to a vector layer.

It creates a new layer with the exact same features and geometries as the input one, but assigned to a new CRS. The geometries are **not** reprojected, they are just assigned to a different CRS.

This algorithm can be used to repair layers which have been assigned an incorrect projection.

Attributes are not modified by this algorithm.

Siehe auch:

Define Shapefile projection, *Find projection*, *Reproject layer*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Vector layer with wrong or missing CRS
Assigned CRS	`CRS`	[crs] Default: `EPSG:4326 - WGS84`	Select the new CRS to assign to the vector layer
Assigned CRS (Optional)	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output layer containing only the duplicates. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Assigned CRS	`OUTPUT`	[same as input]	Vector layer with assigned projection

Python code

Algorithm ID: `qgis: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 vector

Creates a virtual vector layer that contains a set of vector layer. The output virtual vector layer will not be open 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

Label	Name	Type	Beschreibung
Input datasources	`INPUT`	[vector: any] [list]	Select the vector layers you want to use to build the virtual vector
Create „unioned“ VRT	`UNIONED`	[boolean] Default: False	Check if you want to unite all the vectors in a single `vrt` file
Virtual vector	`OUTPUT`	[same as input] Default: `[Save to temporary file]`	Specify the output layer containing only the duplicates. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Virtual vector	`OUTPUT`	[vector: any]	The output virtual vector made from the chosen sources

Convert layer to spatial bookmarks

Creates spatial bookmarks corresponding to the extent of features contained in a layer.

Parameter

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: line, polygon]	The input vector layer
Bookmark destination	`DESTINATION`	[enumeration] Default: 0	Select the destination for the bookmarks. One of: • 0 — Project bookmarks • 1 — User bookmarks
Name field	`NAME_EXPRESSION`	[expression]	Field or expression that will give names to the generated bookmarks
Group field	`GROUP_EXPRESSION`	[expression]	Field or expression that will provide groups for the generated bookmarks

Ausgaben

Label	Name	Type	Beschreibung
Count of bookmarks added	`COUNT`	[number]	

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Bookmark source	`SOURCE`	[enumeration] [list] Default: [0,1]	Select the source(s) of the bookmarks. One or more of: • 0 — Project bookmarks • 1 — User bookmarks
Output CRS	`CRS`	[crs] Default: `EPSG:4326 - WGS 84`	The CRS of the output layer
Ergebnis	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis	OUTPUT	[vector: polygon]	The output (bookmarks) vector layer

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input Layer	INPUT	[vector: any]	Select the vector layer you want to create an attribute index for
Attribute to index	FIELD	[tablefield: any]	Field of the vector layer

Ausgaben

Label	Name	Type	Beschreibung
Indexed layer	OUTPUT	[same as input]	A copy of the input vector layer with an index for the specified field

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Input vector layer

Ausgaben

Label	Name	Type	Beschreibung
Indexed layer	`OUTPUT`	[same as input]	A copy of the input vector layer with a spatial index

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Vector layer with missing projection information
CRS	CRS	[crs]	Select the CRS to assign to the vector layer

Ausgaben

Label	Name	Type	Beschreibung
	INPUT	[same as input]	The input vector layer with the defined projection

Python code

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:

Remove null geometries, *Delete duplicates by attribute*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The layer with duplicate geometries you want to clean
Cleaned	OUTPUT	[same as input] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Count of discarded duplicate records	DUPLICATE_COUNT	[number]	Count of discarded duplicate records
Cleaned	OUTPUT	[same as input]	The output layer without any duplicated geometries
Count of retained records	RETAINED_COUNT	[number]	Count of unique records

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input layer
Fields to match duplicates by	FIELDS	[tablefield: any] [list]	Fields defining duplicates. Features with identical values for all these fields are considered duplicates.
Filtered (no duplicates)	OUTPUT	[same as input] Default: `[Create temporary layer]`	Specify the output layer containing the unique features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 23.56 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Filtered (duplicates) (Optional)	`DUPLICATES`	[same as input] Default: `[Skip output]`	Specify the output layer containing only the duplicates. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Filtered (duplicates) 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]`).
Count of discarded duplicate records	`DUPLICATE_COUNT`	[number]	Count of discarded duplicate records
Filtered (no duplicates)	`OUTPUT`	[same as input]	Vector layer containing the unique features.
Count of retained records	`RETAINED_COUNT`	[number]	Count of unique records

Python code

Algorithm ID: `qgis:deleteduplicatesbyattribute`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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*

Siehe auch:

Delete duplicate geometries, *Remove null geometries*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Dropped geometries	`OUTPUT`	[table]	Specify the output geometryless layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Dropped geometries	`OUTPUT`	[table]	The output geometryless layer. A copy of the original attribute table.

Python code

Algorithm ID: `qgis: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 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 particulary 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

Label	Name	Type	Beschreibung
Additional input datasources (called input1, .., inputN in the query)	`INPUT_DATASOURCES`	[vector: any] [list]	List of layers to query. In the SQL editor you can refer these layers with their **real** name or also with **input1**, **input2**, **inputN** depending on how many layers have been chosen.
SQL query	`INPUT_QUERY`	[string]	Type the string of your SQL query, e.g. `SELECT * FROM input1.`
Unique identifier field Optional	`INPUT_UID_FIELD`	[string]	Specify the column with unique ID
Geometry field Optional	`INPUT_GEOMETRY_FIELD`	[string]	Specify the geometry field
Geometry type Optional	`INPUT_GEOMETRY_TYPE`	[enumeration] Default: 0	Choose the geometry of the result. By default the algorithm will autodetect it. One of: • 0 — Autodetect • 1 — No geometry • 2 — Point • 3 — LineString • 4 — Polygon • 5 — MultiPoint • 6 — MultiLineString • 7 — MultiPolygon
CRS Optional	`INPUT_GEOMETRY_CRS`	[crs]	The CRS to assign to the output layer
SQL Output	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Specify the output layer created by the query. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
SQL Output	`OUTPUT`	[vector: any]	Vector layer created by the query

Python code

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.

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

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Layer to save the selection from
Selected features	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the vector layer for the selected features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Selected features	`OUTPUT`	[same as input]	Vector layer with only the selected features, or no feature if none was selected.

Python code

Algorithm ID: `qgis:saveselectedfeatures`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Layer with unknown projection
Target area for layer (xmin, xmax, ymin, ymax)	`TARGET_AREA`	[extent]	The area that the layer covers. The options for specifying the extent are: • Use Canvas Extent • Select Extent on Canvas • Use Layer Extent It is also possible to provide the extent coordinates directly (xmin, xmax, ymin, ymax).
CRS candidates	`OUTPUT`	[table] Default: `[Create temporary layer]`	Specify the table (geometryless layer) for the CRS suggestions (EPSG codes). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
CRS candidates	`OUTPUT`	[table]	A table with all the CRS (EPSG codes) of the matching criteria.

Python code

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.

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

Label	Name	Type	Beschreibung
Input Layer	`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.
Table field	`FIELD`	[tablefield: any]	Field of the source layer to use for the join
Input layer 2	`INPUT_2`	[vector: any]	Layer with the attribute table to join
Table field 2	`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.
Layer 2 fields to copy Optional	`FIELDS_TO_COPY`	[tablefield: any] [list]	Select the specific fields you want to add. By default all the fields are added.
Join type	`METHOD`	[enumeration] Default: 1	The type of the final joined layer. One of: • 0 — Create separate feature for each matching feature (one-to-many) • 1 — Take attributes of the first matching feature only (one-to-one)
Discard records which could not be joined	`DISCARD_NONMATCHING`	[boolean] Default: True	Check if you don't want to keep the features that could not be joined
Joined field prefix Optional	`PREFIX`	[string]	Add a prefix to joined fields in order to easily identify them and avoid field name collision
Joined layer	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the join. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Unjoinable features from first layer	`NON_MATCHING`	[same as input] Default: `[Skip output]`	Specify the output vector layer for unjoinable features from first layer. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Number of joined features from input table	JOINED_COUNT	[number]	
Unjoinable features from first layer Optional	NON_MATCHING	[same as input]	Vector layer with the non-matched features
Joined layer	OUTPUT	[same as input]	Output vector layer with added attributes from the join
Number of unjoinable features from input table Optional	UNJOINABLE_COUNT	[number]	

Python code

Algorithm ID: `qgis: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)*

Parameter

Label	Name	Type	Beschreibung
Input Layer	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.
Join layer	JOIN	[vector: any]	The attributes of this vector layer will be **added** to the source layer attribute table.

Fortsetzung auf der nächsten Seite

Tab. 23.61 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Geometric predicate	`PREDICATE`	[enumeration] [list] Default: [0]	Select the geometric criteria. One or more of: • 0 — intersects • 1 — contains • 2 — equals • 3 — touches • 4 — overlaps • 5 — within • 6 — crosses
Fields to add (leave empty to use all fields) Optional	`JOIN_FIELDS`	[tablefield: any] [list]	Select the specific fields you want to add. By default all the fields are added.
Join type	`METHOD`	[enumeration]	The type of the final joined layer. One of: • 0 — Create separate feature for each matching feature (one-to-many) • 1 — Take attributes of the first matching feature only (one-to-one)
Discard records which could not be joined	`DISCARD_NONMATCHING`	[boolean] Default: False	Remove from the output the input layer records which could not be joined
Joined field prefix Optional	`PREFIX`	[string]	Add a prefix to joined fields in order to easily identify them and avoid field name collision
Joined layer	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the join. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Unjoinable features from first layer	`NON_MATCHING`	[same as input] Default: `[Skip output]`	Specify the output vector layer for unjoinable features from first layer. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Number of joined features from input table	JOINED_COUNT	[number]	
Unjoinable features from first layer Optional	NON_MATCHING	[same as input]	Vector layer of the non-matched features
Joined layer	OUTPUT	[same as input]	Output vector layer with added attributes from the join

Python code

Algorithm ID: `qgis: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

Parameter

Label	Name	Type	Beschreibung
Input Layer	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.
Join layer	JOIN	[vector: any]	The attributes of this vector layer will be **added** to the source layer attribute table.

Fortsetzung auf der nächsten Seite

Tab. 23.62 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Geometric predicate	`PREDICATE`	[enumeration] [list] Default: [0]	Select the geometric criteria. One or more of: • 0 — intersects • 1 — contains • 2 — equals • 3 — touches • 4 — overlaps • 5 — within • 6 — crosses
Fields to summarize (leave empty to use all fields) Optional	`JOIN_FIELDS`	[tablefield: any] [list]	Select the specific fields you want to add and summarize. By default all the fields are added.
Summaries to calculate (leave empty to use all fields) Optional	`SUMMARIES`	[enumeration] [list] Default: []	Choose which type of summary you want to add to each field and for each feature. One or more of: • 0 — count • 1 — unique • 2 — min • 3 — max • 4 — range • 5 — sum • 6 — mean • 7 — median • 8 — stddev • 9 — minority • 10 — majority • 11 — q1 • 12 — q3 • 13 — iqr • 14 — empty • 15 — filled • 16 — min_length • 17 — max_length • 18 — mean_length
Discard records which could not be joined	`DISCARD_NONMATCHING`	[boolean] Default: False	Remove from the output the input layer records which could not be joined
Joined layer	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the join. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Joined layer	OUTPUT	[same as input]	Output vector layer with summarized attributes from the join

Python code

Algorithm ID: qgis:joinbylocationsummary

```python
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input layer.
Input layer 2	INPUT_2	[vector: any]	The join layer.
Layer 2 fields to copy (leave empty to copy all fields)	FIELDS_TO_COPY	[fields]	Join layer fields to copy (if empty, all fields will be copied).
Discard records which could not be joined	DISCARD_NONMATCHING	[boolean] Default: False	Remove from the output the input layer records which could not be joined
Joined field prefix	PREFIX	[string]	Joined field prefix
Maximum nearest neighbors	NEIGHBORS	[number] Default: 1	Maximum number of nearest neighbors
Maximum distance	MAX_DISTANCE	[number]	Maximum search distance

Fortsetzung auf der nächsten Seite

Tab. 23.63 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Joined layer	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the vector layer containing the joined features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Unjoinable features from first layer	`NON_MATCHING`	[same as input] Default: `[Skip output]`	Specify the vector layer containing the features that could not be joined. One of: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Joined layer	`OUTPUT`	[same as input]	The output joined layer.
Unjoinable features from first layer	`NON_MATCHING`	[same as input]	Layer containing the features from first layer that could not be joined to any features in the join layer.
Number of joined features from input table	`JOINED_COUNT`	[number]	Number of features from the input table that have been joined.
Number of unjoinable features from input table	`UNJOINABLE_COUNT`	[number]	Number of features from the input table that could not be joined.

Python code

Algorithm ID: `qgis: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.

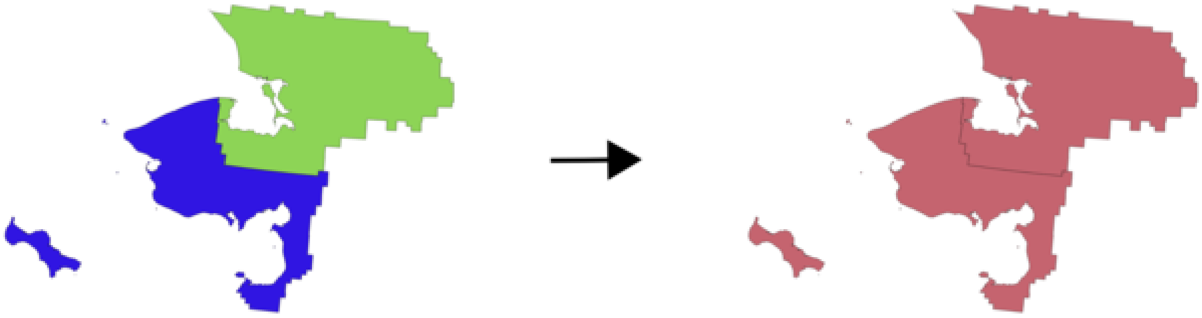
Merge vector layers

Combines multiple vector layers of the **same geometry** type into a single one.

If attributes tables are different, the attribute table of the resulting layer will contain the attributes from all input layers. Non-matching fields will be appended at the end of the attribute table.

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

Label	Name	Type	Beschreibung
Input Layers	`LAYERS`	[vector: any] [list]	The layers that are to be merged into a single layer. Layers should be of the same geometry type.
Destination CRS Optional	`CRS`	[crs]	Choose the CRS for the output layer. If not specified, the CRS of the first input layer is used.
Merged	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Merged	`OUTPUT`	[same as input]	Output vector layer containing all the features and attributes from the input layers.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Input vector layer to sort
Expression	`EXPRESSION`	[expression]	Expression to use for the sorting
Sort ascending	`ASCENDING`	[boolean] Default: True	If checked the vector layer will be sorted from small to large values.
Sort nulls first	`NULLS_FIRST`	[boolean] Default: False	If checked, Null values are placed first
Ordered	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ordered	`OUTPUT`	[same as input]	Output (sorted) vector layer

Python code

Algorithm ID: `qgis: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.

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*

Siehe auch:

Projektion zuweisen, *Define Shapefile projection*, *Find projection*

Parameter

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Input vector layer to reproject
Target CRS	`TARGET_CRS`	[crs] Default: `EPSG:4326 - WGS 84`	Destination coordinate reference system
Reprojected	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Reprojected	`OUTPUT`	[same as input]	Output (reprojected) vector layer

Python code

Algorithm ID: `qgis: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.

Set style for vector layer

Sets the style of a vector layer. The style must be defined in a QML file.

No new output are created: the style is immediately assigned to the vector layer.

Siehe auch:

Set style for raster layer

Parameter

Label	Name	Type	Beschreibung
Input Layer	INPUT	[vector: any]	Input vector layer you want to set the style for
Style file	STYLE	[file]	qml file of the style

Ausgaben

Label	Name	Type	Beschreibung
Input Layer	INPUT	[same as input]	The input vector layer with the new style

Python code

Algorithm ID: qgis:setstyleforvectorlayer

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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.

Parameter

Label	Name	Type	Beschreibung
Input Layer	INPUT	[vector: any]	Input vector layer
Split using values in the field	FIELD	[tablefield: any]	Field to use for splitting
Split value using character	CHAR	[string]	Character to use for splitting
Use regular expression separator	REGEX	[boolean] Default: False	

Fortsetzung auf der nächsten Seite

Tab. 23.65 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Split	`OUTPUT`	[same as input] Default: `Create temporary layer`	Specify output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Split	`OUTPUT`	[same as input]	The output vector layer.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input Layer	`INPUT`	[vector: any]	Input vector layer
Unique ID field	`FIELD`	[tablefield: any]	Field to use for splitting
Output directory	`OUTPUT`	[folder] Default: `[Save to temporary folder]`	Specify the directory for the output layers. One of: • Save to a Temporary Directory • Save to Directory… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output directory	OUTPUT	[folder]	The directory for the output layers
Output layers	OUTPUT_LAYERS	[same as input] [list]	The output vector layers resulting from the split.

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input Layer	INPUT	[vector: any]	Input vector layer

Ausgaben

Label	Name	Type	Beschreibung
Truncated layer	OUTPUT	[folder]	The truncated (empty) layer

Python code

Algorithm ID: qgis: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.

23.1.15 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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Calculate using	`CALC_METHOD`	[enumeration] Default: 0	Calculation parameters to use for the geometric properties. One of: • 0 — Layer CRS • 1 — Project CRS • 2 — Ellipsoidal
Added geom info	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (input copy with geometry) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Added geom info	`OUTPUT`	[same as input]	Copy of the input vector layer with the addition of the geometry fields

Python code

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.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Group by expression	`GROUP_BY`	[tablefield: any] Default: ‚NULL‘	Choose the grouping field. If *NULL* all features will be grouped.

Fortsetzung auf der nächsten Seite

Tab. 23.68 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Aggregates	`AGGREGATES`	[list]	List of output layer field definitions. Example of a field definition: *{‚aggregate‘: ‚sum‘, ‚delimiter‘: ‚,‘, ‚input‘: ‚ $area‘, ‚length‘: 10, ‚name‘: ‚totarea‘, ‚precision‘: 0, ‚type‘: 6}* 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: • Click the button to add a new field. • Click to delete the selected field. • Use and to change order of the fields. • Click to reset to the default (the fields of the input layer). For each of the fields you'd like to retrieve information from, you need to define the following: **`Input expression`** [expression] (**`input`**) Field or expression from the input layer. **`Aggregate function`** [enumeration] (**`aggregate`**) *Function* to use on the input expression to return the aggregated value. Default: *concatenate* (for string data type), *sum* (for numeric data type) **`Delimiter`** [string] (**`delimiter`**) Text string to separate aggregated values, for example in case of concatenation. Default: *,* **`Output field name`** [string] (**`name`**) Name of the aggregated field in the output layer. By default input field name is kept. **`Type`** [enumeration] (**`type`**) Data type of the output field. One of: • 1 — Boolean • 2 — Integer • 4 — Integer64 • 6 — Double • 10 — String • 14 — Date • 16 — DateTime **`Length`** [number] (**`length`**) Length of the output field. **`Precision`** [number] (**`precision`**) Precision of the output field.
Load fields from layer	GUI only	[vector: any]	You can load fields from another layer and use them for the aggregation

Fortsetzung auf der nächsten Seite

Tab. 23.68 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Aggregated	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (aggregate) layer One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Aggregated	`OUTPUT`	[same as input]	Multigeometry vector layer with the aggregated values

Python code

Algorithm ID: `qgis: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.

For **lines geometries**, the boundaries are their end points.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Boundary	`OUTPUT`	[vector: point, line] Default: `[Create temporary layer]`	Specify the output (boundary) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

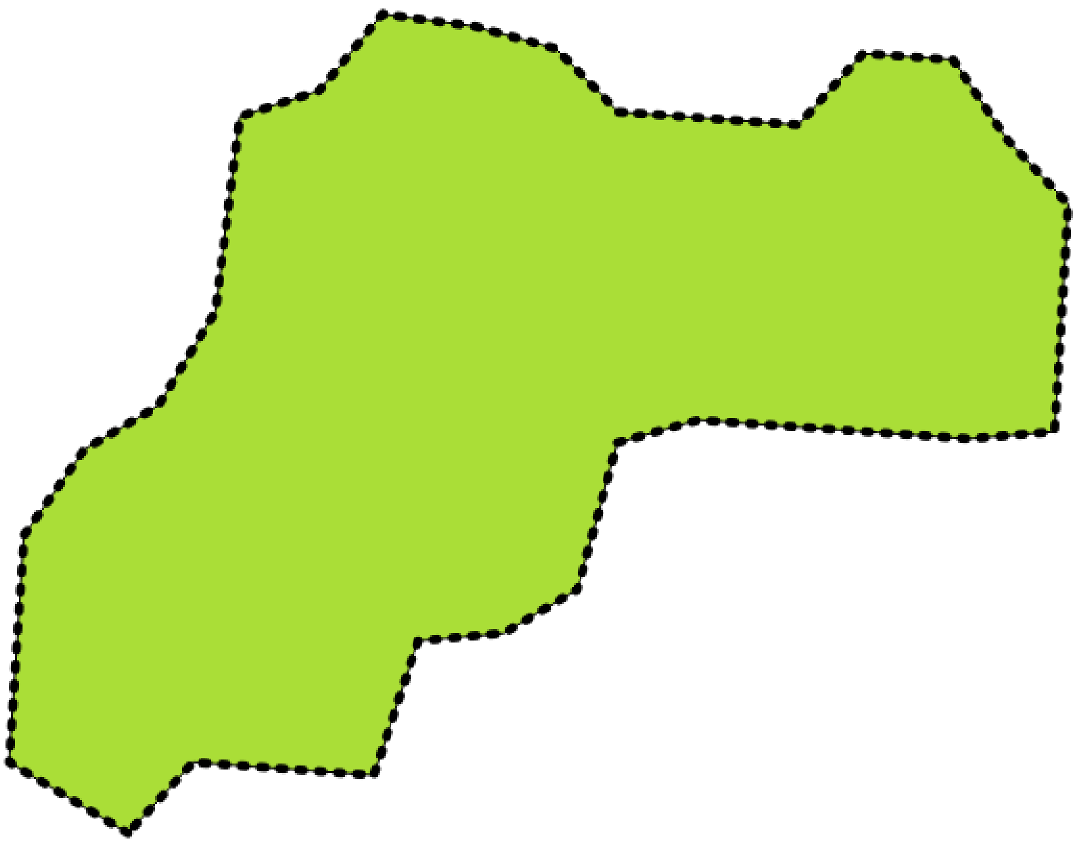


Abb. 23.26: Boundaries (black dashed line) of the source polygon layer

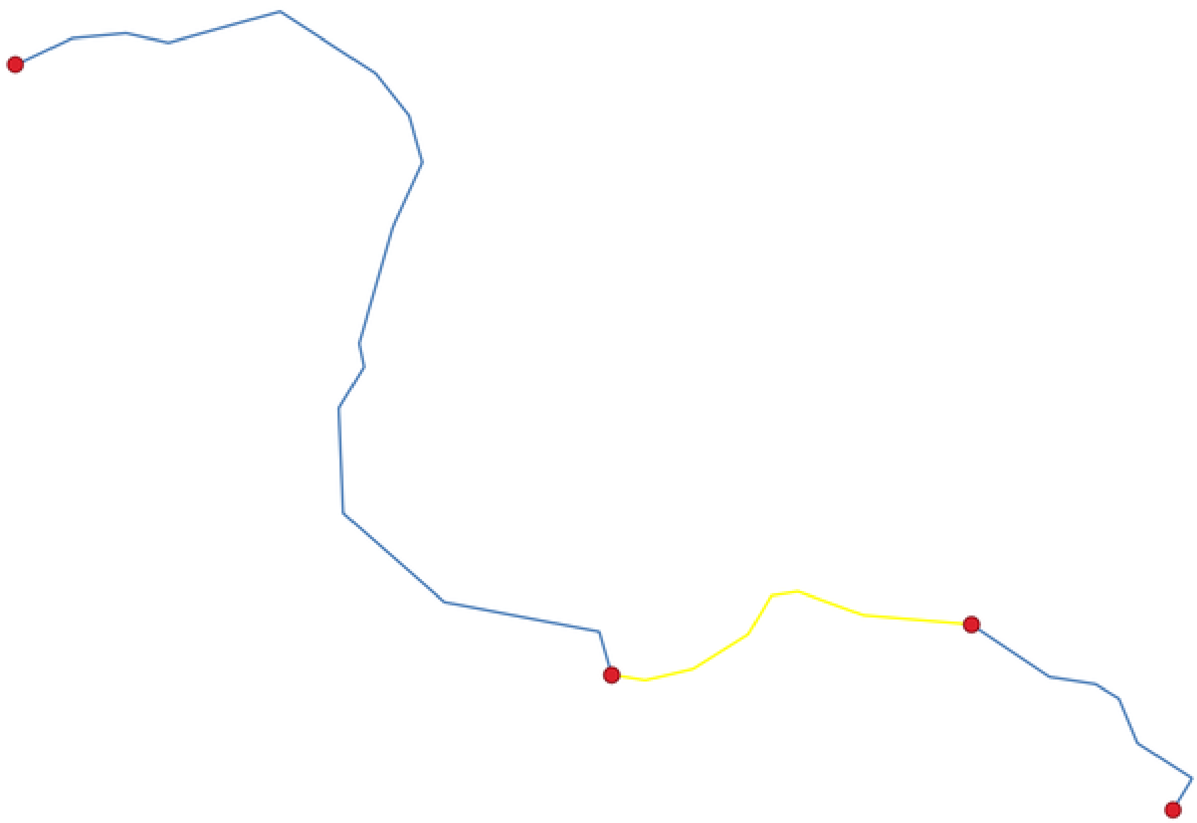


Abb. 23.27: Boundary layer (red points) for lines. In yellow a selected feature.

Ausgaben

Label	Name	Type	Beschreibung
Boundary	`OUTPUT`	[vector: point, line]	Boundaries from the input layer (point for line, and line for polygon)

Python code

Algorithm ID: `qgis: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.

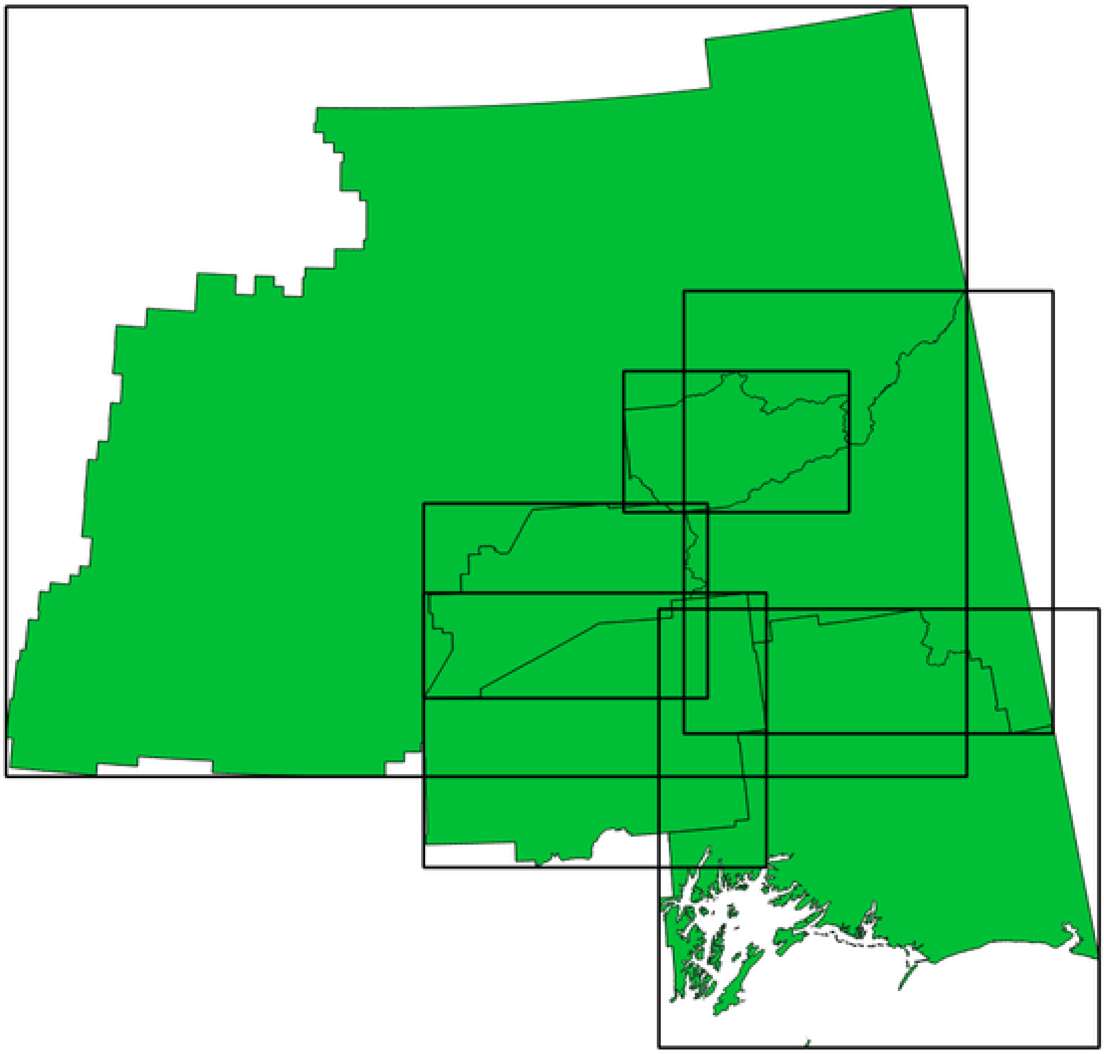


Abb. 23.28: Black lines represent the bounding boxes of each polygon feature

Allows *features in-place modification*

Siehe auch:

Minimum bounding geometry

Parameter

Label	Name	Type	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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Bounds	`OUTPUT`	[vector: polygon]	Bounding boxes of input layer

Python code

Algorithm ID: `qgis: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 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).

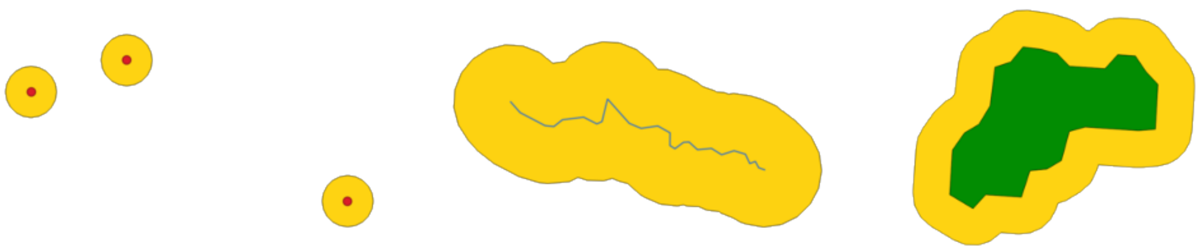


Abb. 23.29: Buffer (in yellow) of points, line and polygon

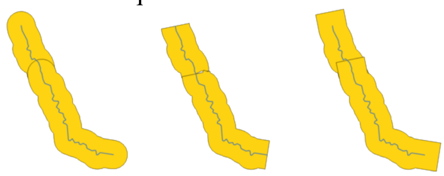
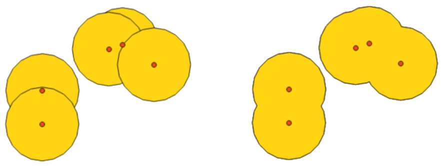
Allows *features in-place modification*

Default menu: *Vector ▸ Geoprocessing Tools*

Siehe auch:

Variable distance buffer, *Multi-ring buffer (constant distance)*, *Variable width buffer (by M value)*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Distance	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 *Variable distance buffer*).
Segments	SEGMENTS	[number] Default: 5	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
End cap style	END_CAP_STYLE	[enumeration] Default: 0	Controls how line endings are handled in the buffer. One of: • 0 — Round • 1 — Flat • 2 — Square  Abb. 23.30: Round, flat and square cap styles
Join style	JOIN_STYLE	[enumeration] Vorgabe: *0*	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: • 0 — Round • 1 — Miter • 2 — Bevel
Miter limit	MITER_LIMIT	[number] Default: 2.0	Controls the maximum distance from the offset curve to use when creating a mitered join (only applicable for miter join styles). Minimum: 1.
Dissolve result	DISSOLVE	[boolean] Default: False	Dissolve the final buffer. If `True` (checked), overlapping buffers will be dissolved (combined) into a new feature.  Abb. 23.31: Standard and dissolved buffer
Buffered	OUTPUT	[vector: polygon] Default: `[Create temporary layer]`	Specify the output (buffer) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffered	OUTPUT	[vector: polygon]	Output (buffer) polygon layer

Python code

Algorithm ID: qgis: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.

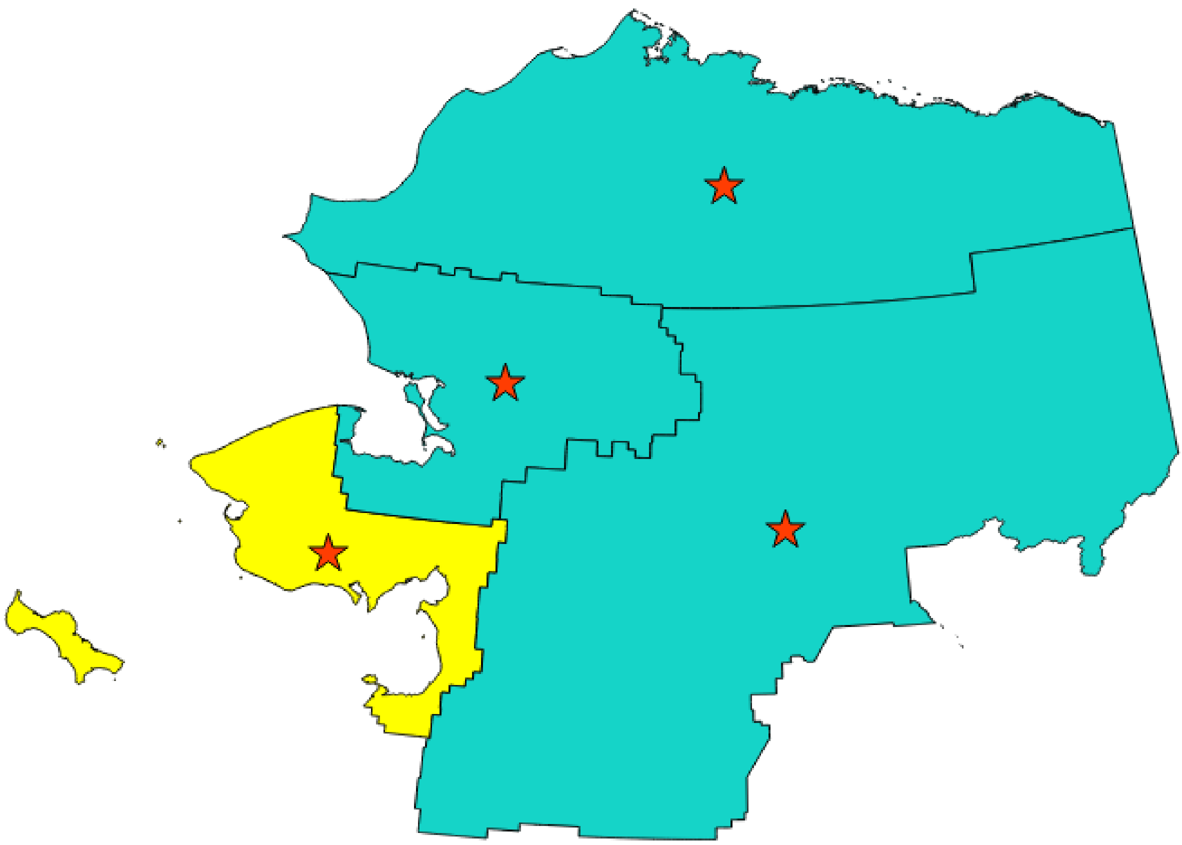


Abb. 23.32: The red stars represent the centroids of the features of the input layer.

Allows *features in-place modification*

Default menu: *Vector ⮕ Geometry Tools*

Siehe auch:

Point on Surface

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Create centroid for each part	`ALL_PARTS`	[boolean] Default: False	If True (checked), a centroid will be created for each part of the geometry
Centroids	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output (centroid) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Centroids	`OUTPUT`	[vector: point]	Output point vector layer (centroids)

Python code

Algorithm ID: `qgis: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):

Default menu: *Vector ⮕ Geometry Tools*

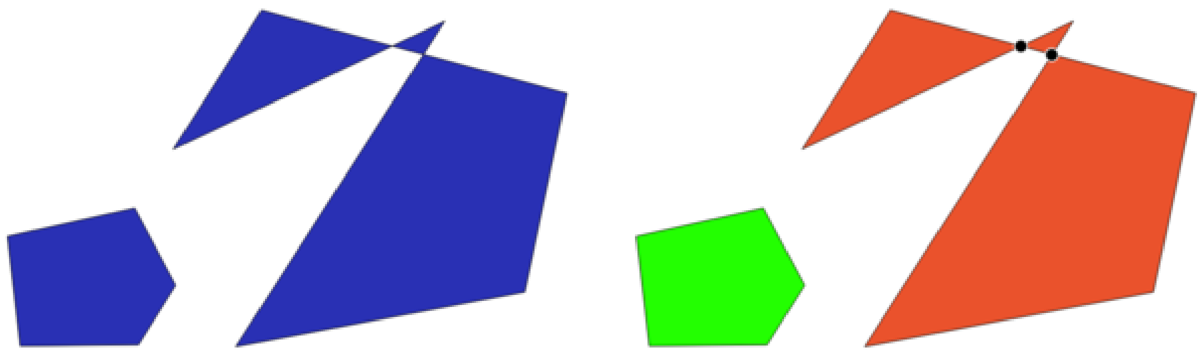


Abb. 23.33: Left: the input layer. Right: the valid layer (green), the invalid layer (orange)

Siehe auch:

Fix geometries and the core plugin *Geometry Checker Plugin*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT_LAYER	[vector: any]	Input vector layer
Method	METHOD	[enumeration] Default: 2	Method to use to check validity. Options: • 0: The one selected in digitizing settings • 1: QGIS • 2: GEOS
Ignore ring self intersection	IGNORE_RING_SELF_INTERSECTION	[boolean] Default: False	Ignore self intersecting rings when checking for validity.
Valid output	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: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Invalid output	INVALID_OUTPUT	[same as input] Default: [Create temporary layer]	Vector layer containing copy of the invalid features of the source layer with the field _errors listing the summary of the error(s) found. One of: • Skip output • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Fortsetzung auf der nächsten Seite

Tab. 23.70 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Error output	`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: • Skip output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Count of errors	`ERROR_COUNT`	[number]	The number of geometries that caused errors.
Error output	`ERROR_OUTPUT`	[vector: point]	Point layer of the exact position of the validity problems detected with the `message` field describing the error(s) found.
Count of invalid features	`INVALID_COUNT`	[number]	The number of invalid geometries.
Invalid output	`INVALID_OUTPUT`	[same as input]	Vector layer containing copy of the invalid features of the source layer with the field `_errors` listing the summary of the error(s) found.
Count of valid features	`VALID_COUNT`	[number]	The number of valid geometries.
Valid output	`VALID_OUTPUT`	[same as input]	Vector layer containing a copy of the valid features of the source layer.

Python code

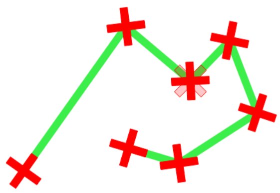
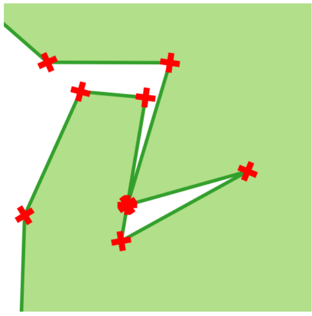
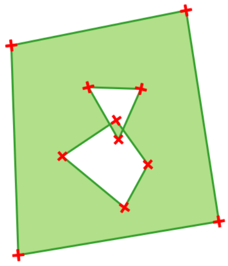
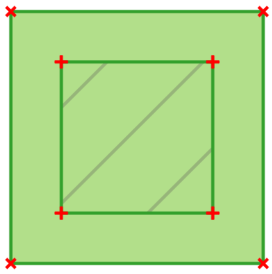
Algorithm ID: `qgis:checkvalidity`

```
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.

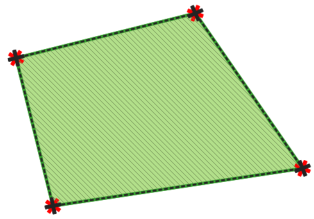
Types of error messages and their meanings

Tab. 23.72: If the GEOS method is used the following error messages can occur:

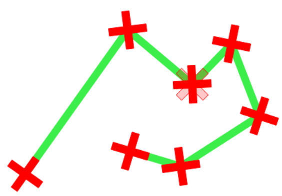
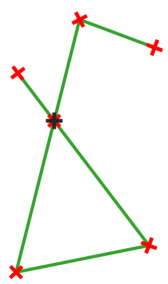
Error message	Explanation	Example
Repeated point	This error happens when a given vertex is repeated.	
Ring self-intersection	This error happens when a geometry touches itself and generates a ring.	
Self-intersection	This error happens when a geometry touches itself.	
Topology validation error		
Hole lies outside shell		
Holes are nested		
Interior is disconnected		
Nested shells	This error happens when a polygon geometry is on top of another polygon geometry.	

Fortsetzung auf der nächsten Seite

Tab. 23.72 – Fortsetzung der vorherigen Seite

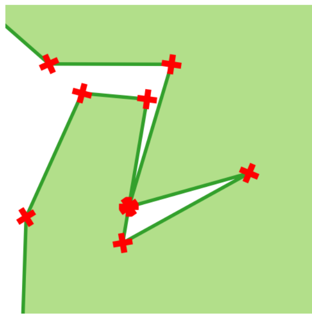
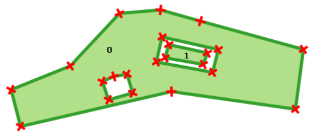
Error message	Explanation	Example
Duplicate rings	This error happens when two rings (exterior or interior) of a polygon geometry are identical	
Too few points in geometry component		
Invalid coordinate	For a point geometry, this error happens when the geometry does not have a proper coordinate pair. The coordinate pair does not contain a latitude value and a longitude value in that order.	
Ring is not closed		

Tab. 23.73: 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		
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).	

Fortsetzung auf der nächsten Seite

Tab. 23.73 – Fortsetzung der vorherigen Seite

Error message	Explanation	Example
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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Unique ID fields	`FIELD`	[tablefield: any] [list]	Choose one or more attributes to collect the geometries
Collected	`OUTPUT`	[same as input]	Vector layer with collected geometries

Ausgaben

Label	Name	Type	Beschreibung
Collected	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the collected geometries. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Python code

Algorithm ID: `qgis: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 (alpha shapes)

Computes the concave hull of the features in an input point layer.

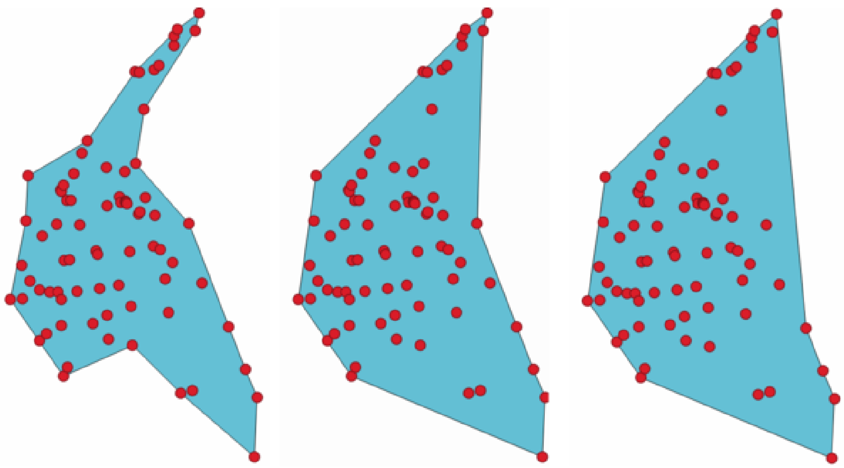


Abb. 23.34: Concave hulls with different thresholds (0.3, 0.6, 0.9)

Siehe auch:

Convex hull, *Concave hull (k-nearest neighbor)*

Parameter

Label	Name	Type	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_MULTIGEOMETRY`	[boolean] Default: True	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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Concave hull	`OUTPUT`	[vector: polygon]	The output vector layer

Python code

Algorithm ID: `qgis: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.

Concave hull (k-nearest neighbor)

Generates a concave hull polygon from a set of points. If the input layer is a line or polygon layer, it will use the vertices.

The number of neighbors to consider determines the concaveness of the output polygon. A lower number will result in a concave hull that follows the points very closely, while a higher number will have a smoother shape. The minimum number of neighbor points to consider is 3. A value equal to or greater than the number of points will result in a convex hull.

If a field is selected, the algorithm will group the features in the input layer using unique values in that field and generate individual polygons in the output layer for each group.

Siehe auch:

Concave hull (alpha shapes)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Number of neighboring points to consider (a lower number is more concave, a higher number is smoother)	`KNEIGHBORS`	[number] Default: 3	Determines the concaveness of the output polygon. A small number will result in a concave hull that follows the points very closely, while a high number will make the polygon look more like the convex hull (if the number is equal to or larger than the number of features, the result will be the convex hull). Minimum value: 3.
Field Optional	`FIELD`	[tablefield: any] Default: None	If specified, one concave hull polygon is generated for each unique value of the field (by selecting features using this value).
Concave hull	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Concave hull	`OUTPUT`	[vector: polygon]	The output vector layer

Python code

Algorithm ID: `qgis:knearestconcavehull`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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.

Not all conversions are possible. For instance, a line can be converted to a point, but a point cannot be converted to a line. A line can also be converted to a polygon.

Siehe auch:

Polygonize, *Lines to polygons*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
New geometry type	`TYPE`	[enumeration] Default: 0	Geometry type to apply to the output features. One of: • 0 — Centroids • 1 — Nodes • 2 — Linestrings • 3 — Multilinestrings • 4 — Polygons
Converted	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Converted	`OUTPUT`	[vector: any]	Output vector layer - the type depends on the parameters

Python code

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.

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.

Allows *features in-place modification*

Default menu: *Vector ⍰ Geoprocessing Tools*

Siehe auch:

Minimum bounding geometry, *Concave hull (alpha shapes)*

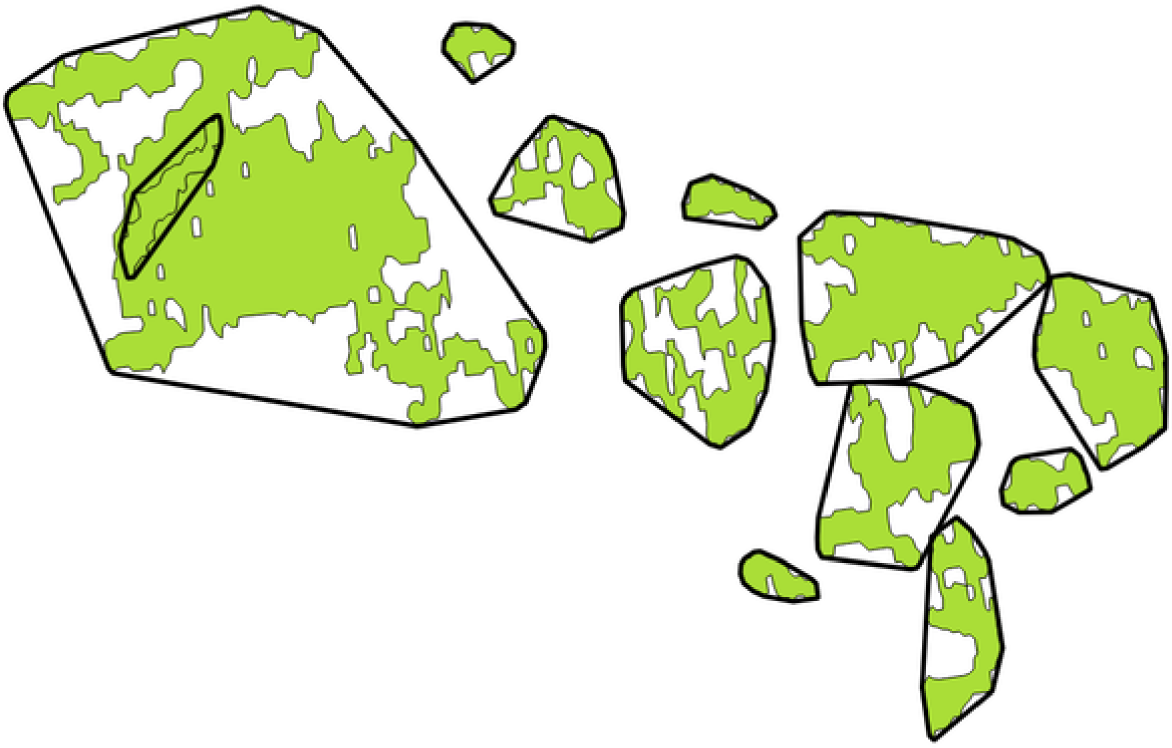


Abb. 23.35: Black lines identify the convex hull for each layer feature

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Convex hull	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Convex hull	OUTPUT	[vector: polygon]	The output (convex hull) vector layer

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Extent (xmin, xmax, ymin, ymax)	`INPUT`	[extent]	Input extent
Extent	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extent	`OUTPUT`	[vector: polygon]	The output (extent) vector layer

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Point	`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.
Point	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Point	`OUTPUT`	[vector: point]	The output point vector layer containing the input point.

Python code

Algorithm ID: `qgis: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.

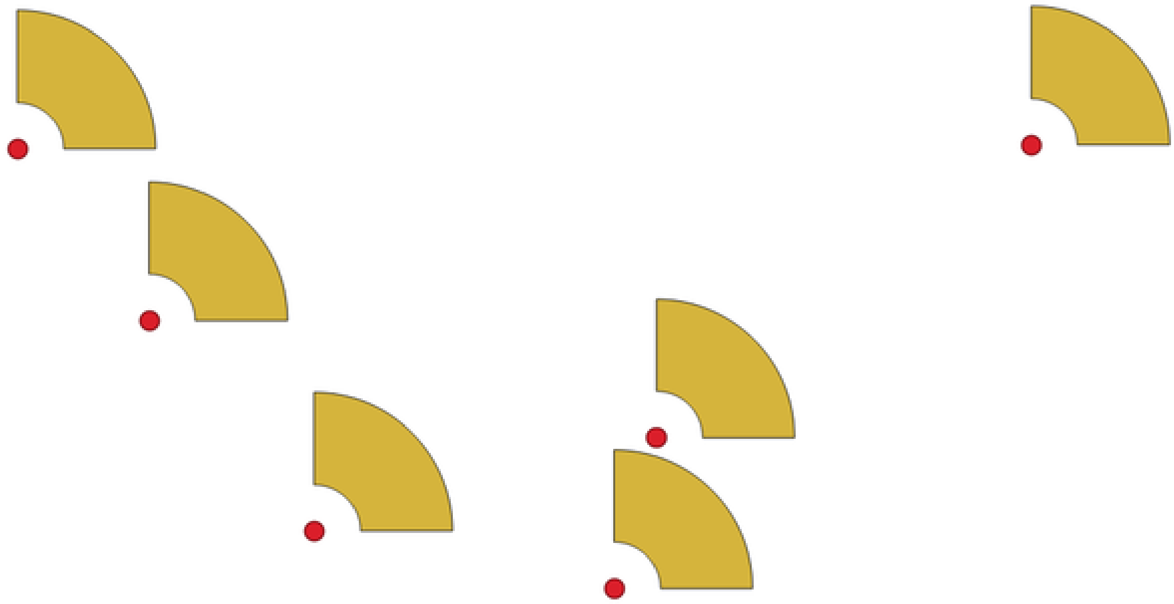


Abb. 23.36: Wedge buffers

The native output from this algorithm are CurvePolygon geometries, but these may be automatically segmentized to Polygons depending on the output format.

Siehe auch:

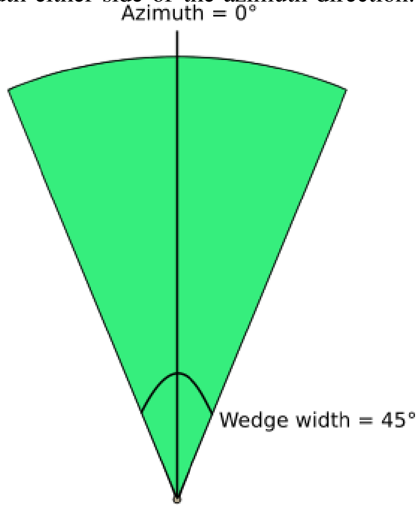
Buffer, *Variable width buffer (by M value)*, *Tapered buffers*

Parameter

Label	Name	Type	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. 23.75 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Wedge width (in degrees)	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.  Abb. 23.37: Azimuth and width values of the wedge buffer
Outer radius	OUTER_RADIUS	[number] Default: 1.0	The outer *size* (length) of the wedge: the size is meant from the source point to the edge of the wedge shape.
Inner radius Optional	INNER_RADIUS	[number] Default: 0.0	Inner radius value. If 0 the wedge will begin from the source point.
Buffers	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffers	OUTPUT	[vector: polygon]	The output (wedge buffer) vector layer

Python code

Algorithm ID: qgis: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.

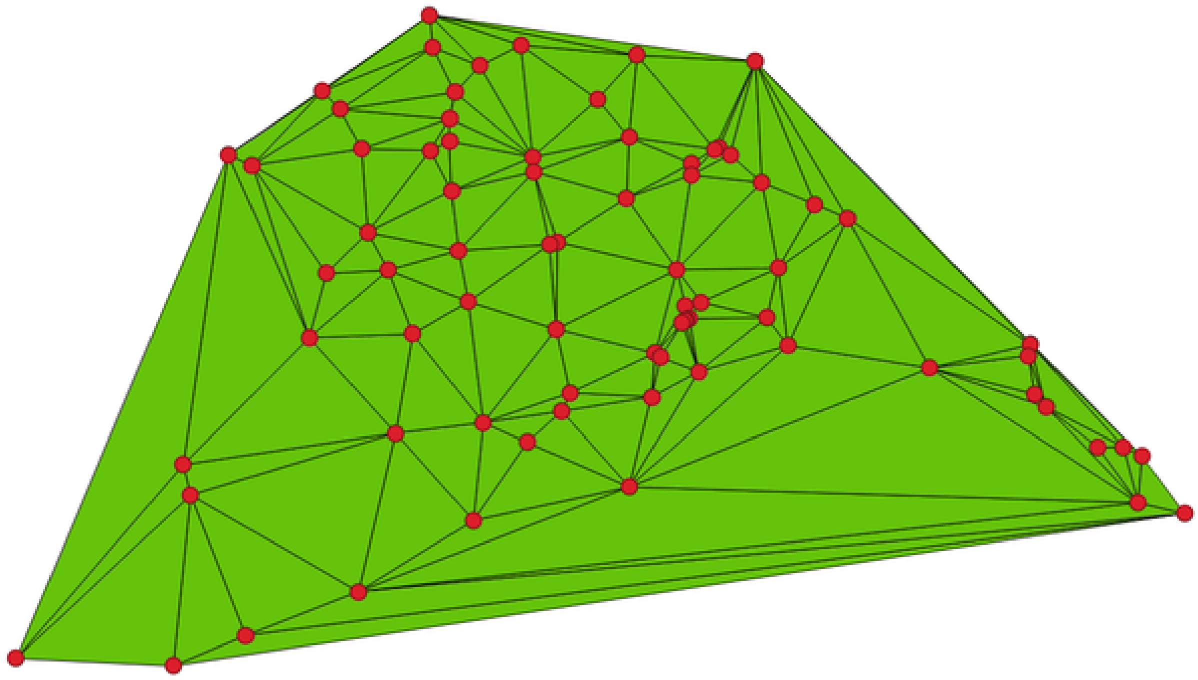


Abb. 23.38: Delaunay triangulation on points

Default menu: *Vector ￼ Geometry Tools*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: point]	Input point vector layer
Delaunay triangulation	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Delaunay triangulation	`OUTPUT`	[vector: polygon]	The output (Delaunay triangulation) vector layer

Python code

Algorithm ID: `qgis: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.

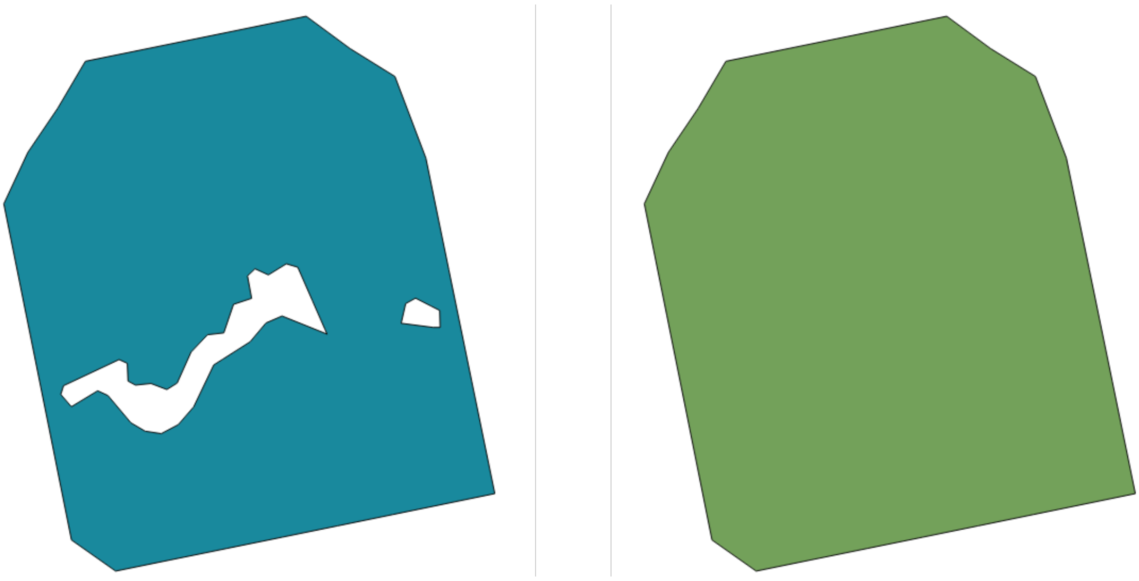


Abb. 23.39: Before and after the cleaning

Allows *features in-place modification*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	Input polygon vector layer
Remove holes with area less than Optional	`MIN_AREA`	[number] Default: 0.0	Only holes with an area less than this threshold will be deleted. If `0.0` is added, **all** the holes will be deleted.
Cleaned	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Cleaned	OUTPUT	[same as input]	The output (cleaned) vector layer

Python code

Algorithm ID: qgis: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.

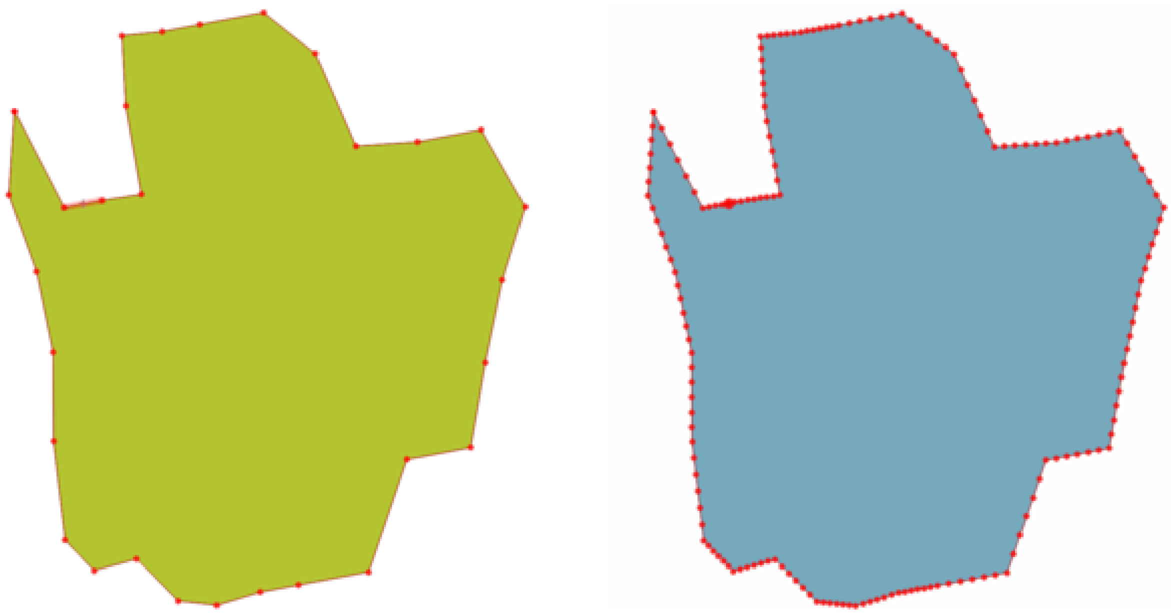


Abb. 23.40: Red points show the vertices before and after the densify

Allows *features in-place modification*

Default menu: *Vector ▸ Geometry Tools*

Siehe auch:

Densify by interval

Parameter

Label	Name	Type	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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Densified	`OUTPUT`	[same as input]	The output (densified) vector layer

Python code

Algorithm ID: `qgis: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] -> [10 0]` to be converted to `[0 0] -> [2.5 0] -> [5 0] -> [7.5 0] -> [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.

Allows *features in-place modification*

Siehe auch:

Densify by count

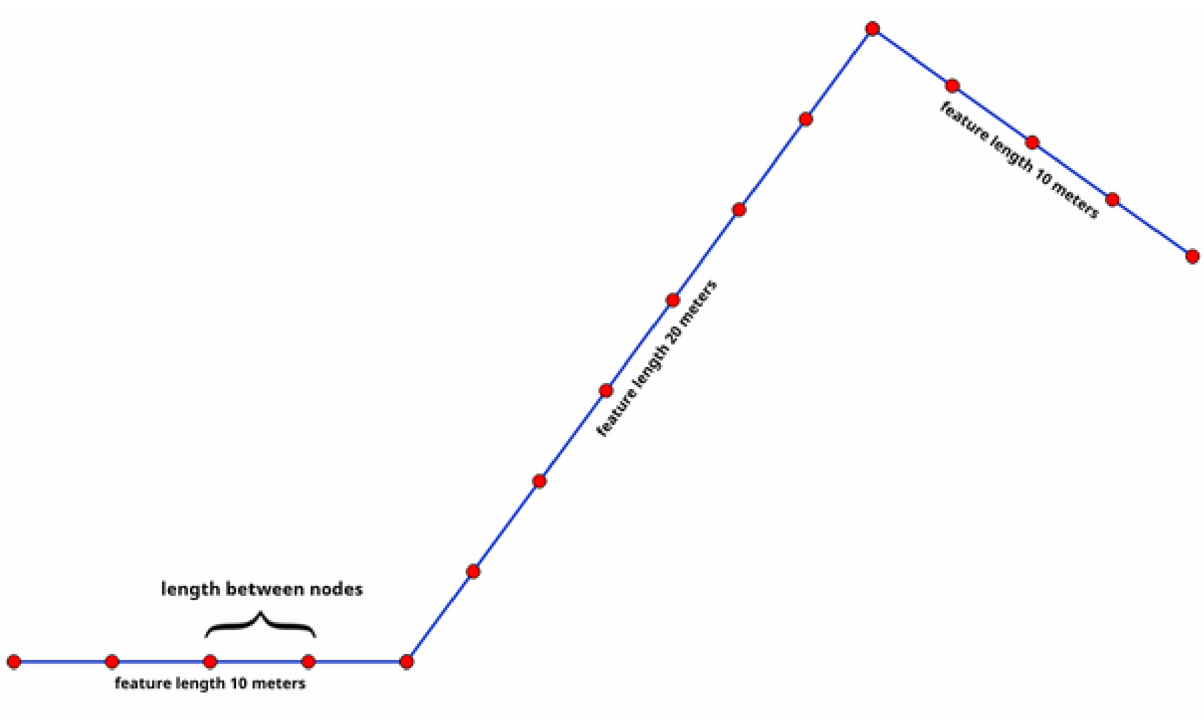


Abb. 23.41: Densify geometry at a given interval

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Interval between vertices to add	`INTERVAL`	[number] Default: 1.0	Maximum distance between two consecutive vertices
Densified	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Densified	`OUTPUT`	[same as input]	The output (densified) vector layer

Python code

Algorithm ID: qgis: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.

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.

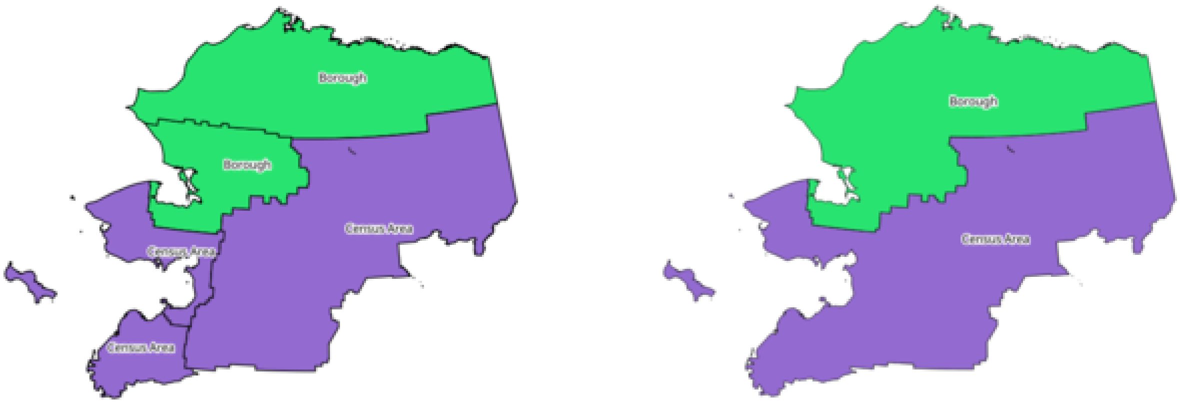


Abb. 23.42: Dissolve the polygon layer on a common attribute

Default menu: *Vector* ⮕ *Geoprocessing Tools*

Siehe auch:

Aggregate, *Collect geometries*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Dissolve field(s) 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.
Dissolved	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Dissolved	`OUTPUT`	[same as input]	The output vector layer with dissolved geometries

Python code

Algorithm ID: `qgis: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.

Siehe auch:

Set M value from raster, *Set Z value*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Raster layer	`RASTER`	[raster]	Raster layer with Z values
Band number	`BAND`	[raster band] Default: 1	The raster band to take the Z values from
Value for nodata or non-intersecting vertices	`NODATA`	[number] Default: 0	Value to use in case the vertex does not intersect (a valid pixel of) the raster
Scale factor	`SCALE`	[number] Default: 1.0	Scaling value: the band values are multiplied by this value.
Updated	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer (with Z values from the raster layer). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Updated	`OUTPUT`	[same as input]	The output vector layer with Z values from the raster layer

Python code

Algorithm ID: `qgis: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:

Set M value, *Set Z value*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer with M or Z values
Drop M Values	`DROP_M_VALUES`	[boolean] Default: False	Removes the M values from the geometries
Drop Z Values	`DROP_Z_VALUES`	[boolean] Default: False	Removes the Z values from the geometries
Z/M Dropped	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Z/M Dropped	`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).

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: polygon]	Input polygon vector layer
Merge selection with the neighboring polygon with the	MODE	[enumeration] Default: None	Choose the parameter to use in order to get rid of the selected polygons: • 0 — Largest Area • 1 — Smallest Area • 2 — Largest Common Boundary
Eliminated	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Eliminated	OUTPUT	[vector: polygon]	The output polygon vector layer.

Python code

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.

Allows *features in-place modification*

Siehe auch:

Subdivide, *Line substring*

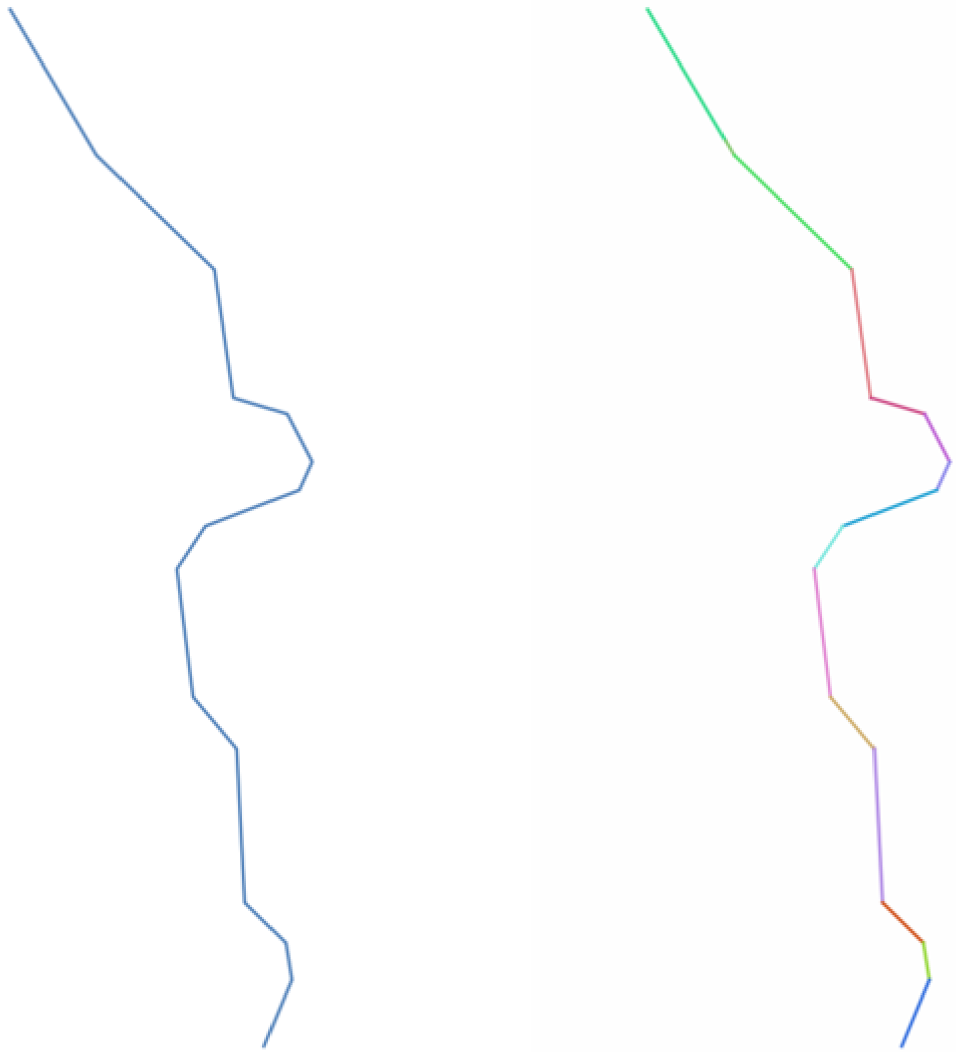


Abb. 23.43: The original line layer and the exploded one

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Exploded	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Exploded	`OUTPUT`	[vector: line]	The output line vector layer with features representing each segment of the input layer.

Python code

Algorithm ID: `qgis: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.

Allows *features in-place modification*

Siehe auch:

Line substring

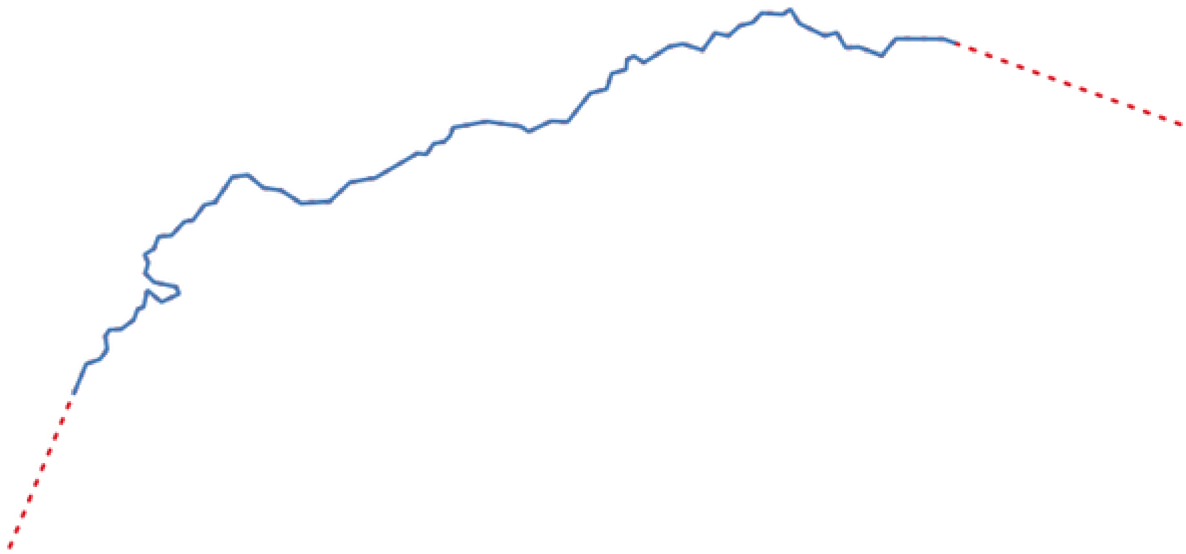


Abb. 23.44: The red dashes represent the initial and final extension of the original layer

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Start distance	`START_DISTANCE`	[number]	Distance by which to extend the first segment of the line (starting point)
End distance	`END_DISTANCE`	[number]	Distance by which to extend the last segment of the line (ending point)
Extended	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extended	`OUTPUT`	[vector: line]	The output (extended) line vector layer.

Python code

Algorithm ID: `qgis: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, *Set M value*, *Drop M/Z values*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Summaries to calculate	`SUMMARIES`	[enumeration] Default: [0]	Statistics on the M values of a geometry. One or more of: • 0 — First • 1 — Last • 2 — Count • 3 — Sum • 4 — Mean • 5 — Median • 6 — St.dev (pop) • 7 — Minimum • 8 — Maximum • 9 — Range • 10 — Minority • 11 — Majority • 12 — Variety • 13 — Q1 • 14 — Q3 • 15 — IQR
Output column prefix	`COLUMN_PREFIX`	[string] Default: ‚m_'	The prefix for the output (M) column
Extracted	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extracted	OUTPUT	[same as input]	The output vector layer (with M values)

Python code

Algorithm ID: qgis: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.

Siehe auch:

Extract vertices, *Filter vertices by M value*, *Filter vertices by Z value*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Vertex indices	VERTICES	[string] Default: ‚0'	Comma-separated string of the indices of the vertices to extract.
Vertices	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Vertices	OUTPUT	[vector: point]	The output (point) vector layer containing the specified vertices from the input layer geometries.

Python code

Algorithm ID: qgis: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.

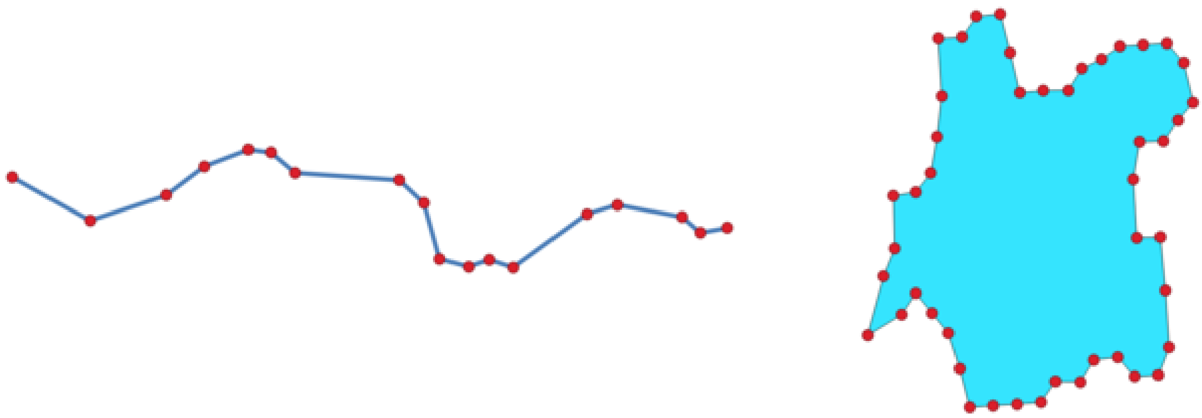


Abb. 23.45: Vertices extracted for line and polygon layer

Default menu: *Vector Geometry Tools*

Siehe auch:

Extract specific vertices, *Filter vertices by M value*, *Filter vertices by Z value*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Vertices	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Vertices	`OUTPUT`	[vector: point]	The output (point) vector layer containing the vertices from the input layer geometries.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer

Fortsetzung auf der nächsten Seite

Tab. 23.78 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Summaries to calculate	SUMMARIES	[enumeration] Default: [0]	Statistics on the Z values of a geometry. One or more of: • 0 — First • 1 — Last • 2 — Count • 3 — Sum • 4 — Mean • 5 — Median • 6 — St.dev (pop) • 7 — Minimum • 8 — Maximum • 9 — Range • 10 — Minority • 11 — Majority • 12 — Variety • 13 — Q1 • 14 — Q3 • 15 — IQR
Output column prefix	COLUMN_PREFIX	[string] Default: ‚z_'	The prefix for the output (Z) column
Extracted	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extracted	OUTPUT	[same as input]	The output vector layer (with Z values)

Python code

Algorithm ID: qgis: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.

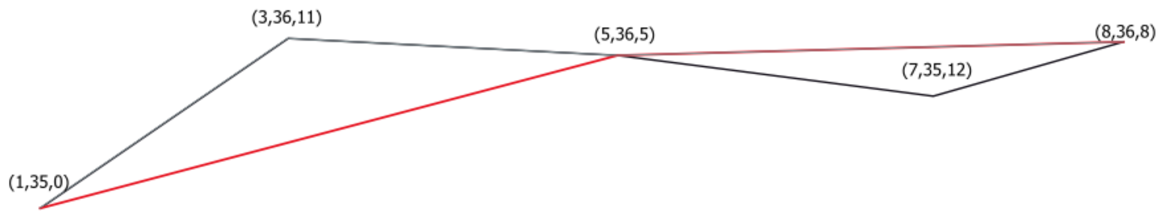


Abb. 23.46: The red line represents the black line with only vertices whose M value is <=10.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer to remove vertices from
Minimum Optional	`MIN`	[number] Default: *Not set*	Minimum of M values allowed
Maximum Optional	`MAX`	[number] Default: *Not set*	Maximum of M values allowed
Filtered	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Filtered	`OUTPUT`	[same as input]	The output vector layer of features with only the filtered vertices.

Python code

Algorithm ID: `qgis: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.

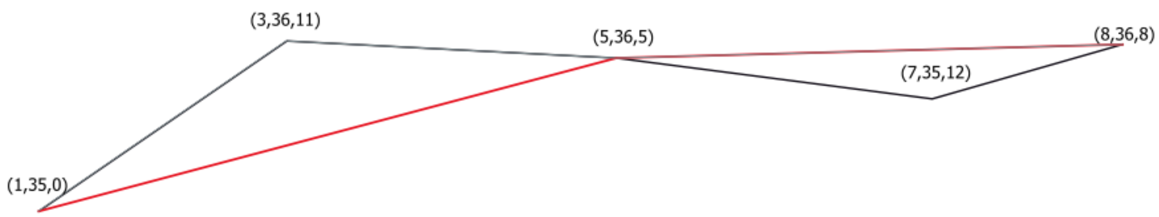


Abb. 23.47: The red line represents the black line with only vertices whose Z value is <=10.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer to remove vertices from
Minimum Optional	`MIN`	[number] Default: *Not set*	Minimum of Z values allowed
Maximum Optional	`MAX`	[number] Default: *Not set*	Maximum of Z values allowed
Filtered	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Filtered	`OUTPUT`	[same as input]	The output vector layer of features with only the filtered vertices.

Python code

Algorithm ID: `qgis: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.

Bemerkung: M values will be dropped from the output.

Allows *features in-place modification*

Siehe auch:

Check validity

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Fixed geometries	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Fixed geometries	OUTPUT	[same as input]	The output vector layer with fixed geometries.

Python code

Algorithm ID: qgis: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.

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: polygon]	Input vector layer
Reoriented	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Reoriented	OUTPUT	[vector: polygon]	The output vector layer with reoriented geometries.

Python code

Algorithm ID: `qgis: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 (±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.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Split	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output line vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Split	`OUTPUT`	[vector: line]	The output line vector layer split at the antimeridian.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Output geometry type	`OUTPUT_GEOMETRY`	[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: • 0 — Polygon • 1 — Line • 2 — Point
Output geometry has z values	`WITH_Z`	[boolean] Default: False	Choose if the output geometry should include the Z dimension
Output geometry has m values	`WITH_M`	[boolean] Default: False	Choose if the output geometry should include the M dimension
Geometry expression	`EXPRESSION`	[expression] Default: ‚$geometry'	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.
Modified geometry	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Modified geometry	`OUTPUT`	[vector: any]	The output vector layer

Python code

Algorithm ID: `qgis: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. 23.48: Interpolated point at 500m of the beginning of the line

Siehe auch:

Points along geometry

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Distance	`DISTANCE`	[number] Default: 0.0	Distance from the beginning of the line
Interpolated points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated points	`OUTPUT`	[vector: point]	The output point vector layer with features at a set distance along the line or polygon boundary

Python code

Algorithm ID: `qgis: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.

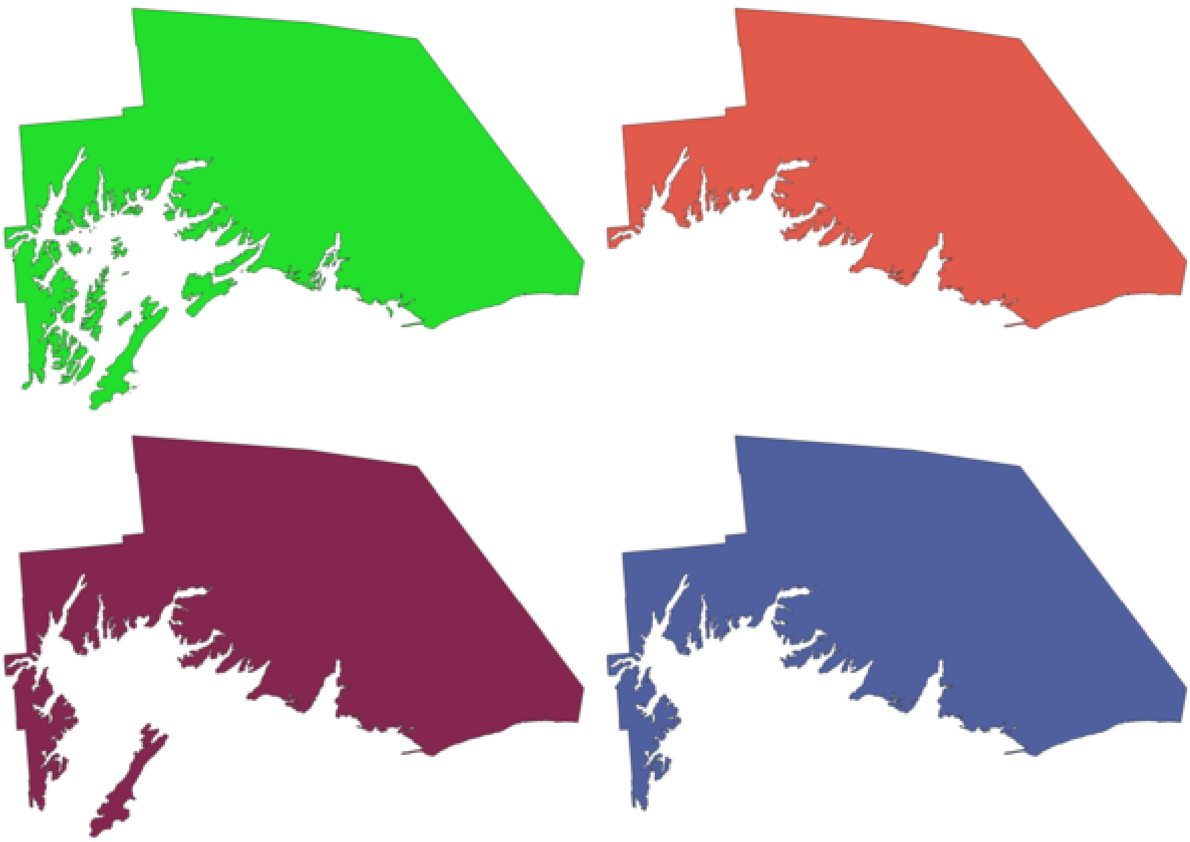


Abb. 23.49: Clockwise from top left: original multipart feature, one, two and three biggest parts kept

Parameter

Label	Name	Type	Beschreibung
Polygons	`INPUT`	[vector: polygon]	Input polygon vector layer
Parts to keep	`PARTS`	[number] Default: 1	Number of parts to keep. If 1, only the biggest part of the feature will be kept.
Parts	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Parts	`OUTPUT`	[vector: polygon]	The output polygon vector layer with the N biggest parts of each feature

Python code

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.

Allows *features in-place modification*

Siehe auch:

Extend lines

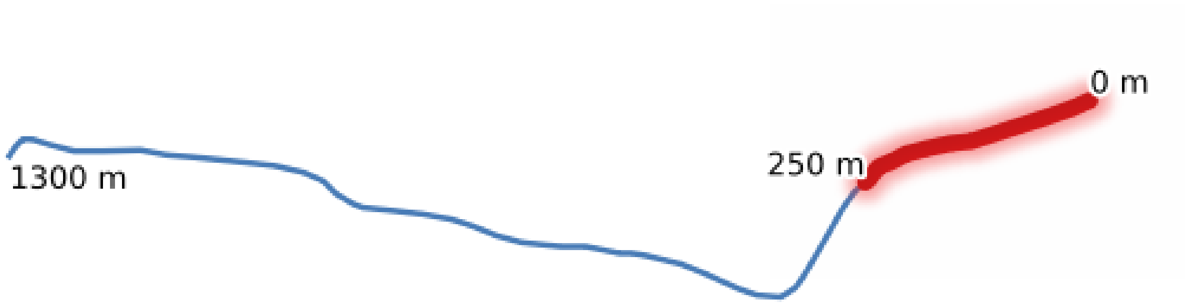


Abb. 23.50: Substring line with starting distance set at 0 meters and the ending distance at 250 meters.

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer
Start distance	START_DISTANCE	[number]	Distance along the input line to the start point of the output feature
End distance	END_DISTANCE	[number]	Distance along the input line to the end point of the output feature
Substring	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Substring	OUTPUT	[vector: line]	The output line vector layer.

Python code

Algorithm ID: qgis: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 from of the input line layer.

Default menu: *Vector ⍰ Geometry Tools*

Siehe auch:

Polygons to lines, *Polygonize*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Polygons	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Polygons	`OUTPUT`	[vector: polygon]	The output polygon vector layer.

Python code

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*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line]	Input line vector layer
Merged	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Merged	OUTPUT	[vector: line]	The output (merged) line vector layer.

Python code

Algorithm ID: qgis: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

Siehe auch:

Minimum enclosing circles

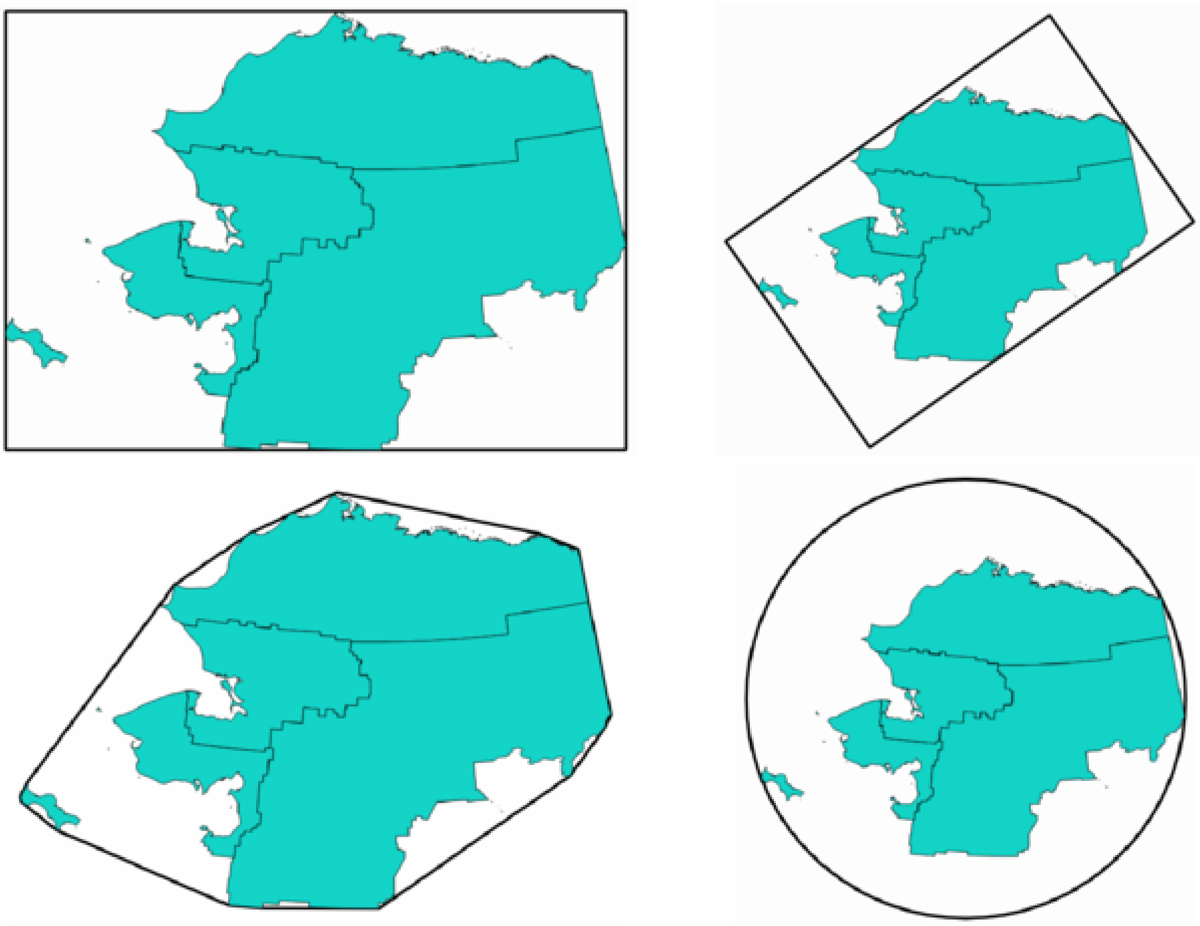


Abb. 23.51: Clockwise from top left: envelope, oriented rectangle, circle, convex hull

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Field 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.
Geometry type	`TYPE`	[enumeration] Default: 0	Enclosing geometry types. One of: • 0 — Envelope (Bounding Box) • 1 — Minimum Oriented Rectangle • 2 — Minimum Enclosing Circle • 3 — Convex Hull
Bounding geometry	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Bounding geometry	`OUTPUT`	[vector: polygon]	The output (bounding) polygon vector layer.

Python code

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.

Allows *features in-place modification*

Siehe auch:

Minimum bounding geometry

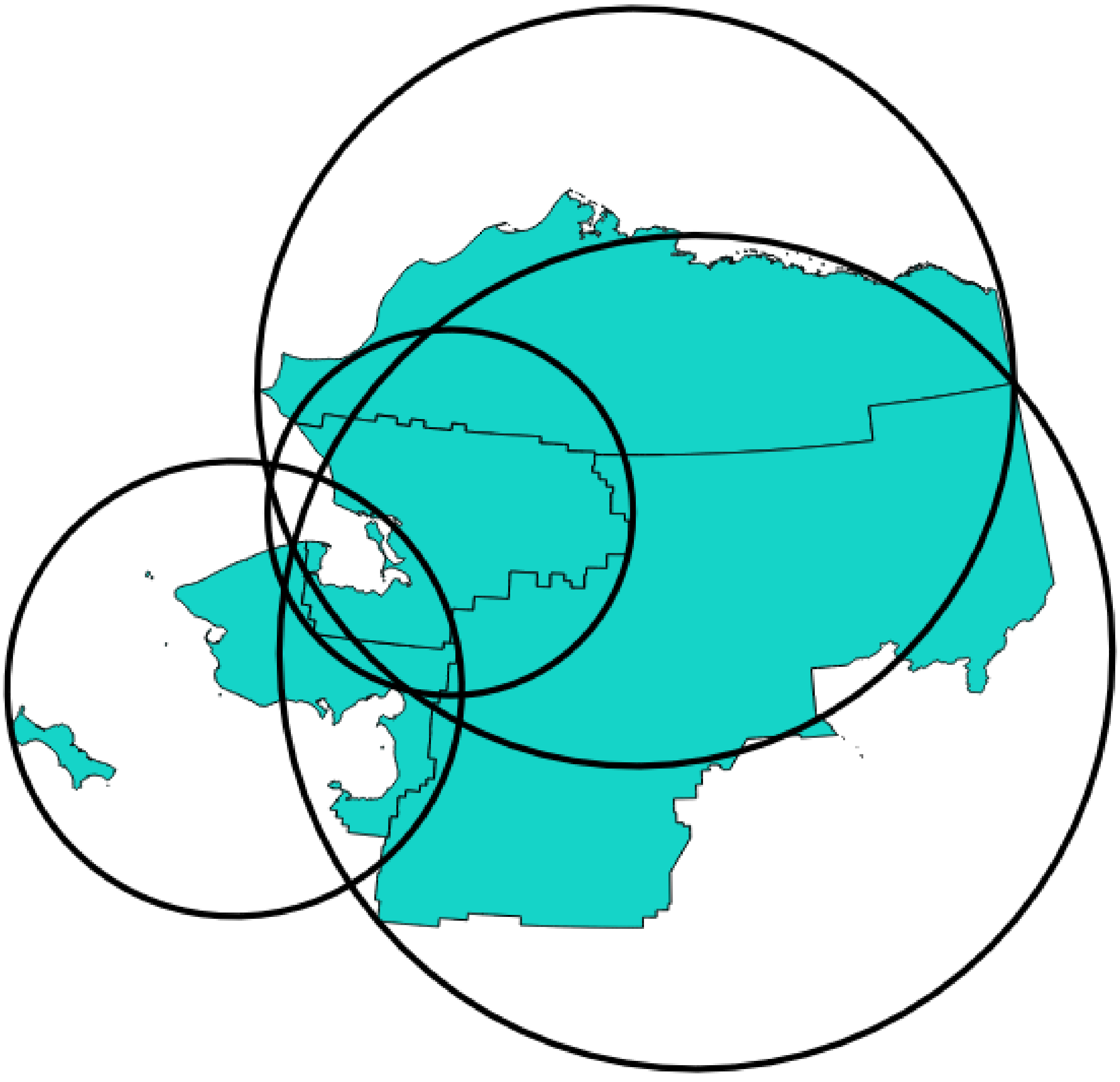


Abb. 23.52: Enclosing circles for each feature

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Number of segment in circles	`SEGMENTS`	[number] Default: 72	The number of segment used to approximate a circle. Minimum 8, maximum 100000.
Minimum enclosing circles	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Minimum enclosing circles	`OUTPUT`	[vector: polygon]	The output polygon vector layer.

Python code

Algorithm ID: `qgis:minimumenclosingcircle`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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.

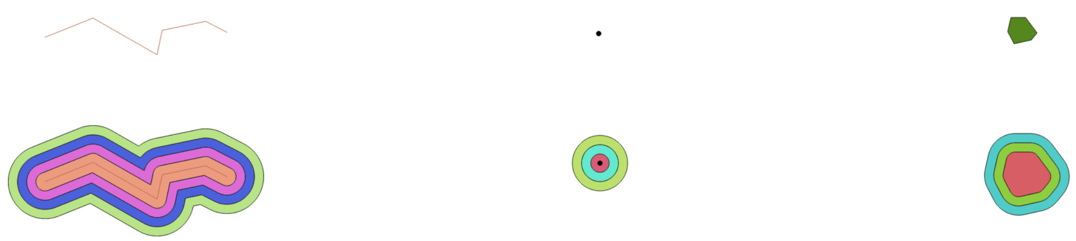


Abb. 23.53: Multi-ring buffer for a line, point and polygon layer

Allows *features in-place modification*

Siehe auch:

Buffer, *Variable distance buffer*, *Rectangles, ovals, diamonds (fixed)*, *Rectangles, ovals, diamonds (variable)*, *Single sided buffer*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Number of rings	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).
Distance between rings	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).
Multi-ring buffer (constant distance)	OUTPUT	[vector: polygon] Default: [Create temporary layer]	Specify the output polygon vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Multi-ring buffer (constant distance)	OUTPUT	[vector: polygon]	The output polygon vector layer.

Python code

Algorithm ID: qgis: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. 23.54: Left the multipart source layer and right the single part output result

Allows *features in-place modification*

Default menu: *Vector ▯ Geometry Tools*

Siehe auch:

Collect geometries, *Promote to multipart*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Single parts	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Single parts	`OUTPUT`	[same as input]	The output vector layer.

Python code

Algorithm ID: `qgis: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.

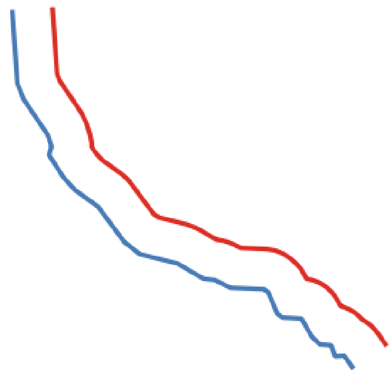


Abb. 23.55: In blue the source layer, in red the offset one

Allows *features in-place modification*

Siehe auch:

Array of offset (parallel) lines, *Translate*

Parameter

Label	Name	Type	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 *Variable distance buffer*).
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] Vorgabe: *0*	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: • 0 — Round • 1 — Miter • 2 — Bevel
Miter limit	MITER_LIMIT	[number] Default: 2.0	Controls the maximum distance from the offset curve to use when creating a mitered join (only applicable for miter join styles). Minimum: 1.

Fortsetzung auf der nächsten Seite

Tab. 23.83 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Offset	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output (offset) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Offset	`OUTPUT`	[vector: line]	Output (offset) line layer

Python code

Algorithm ID: `qgis: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.

Allows *features in-place modification*

Siehe auch:

Minimum bounding geometry

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Bounding boxes	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

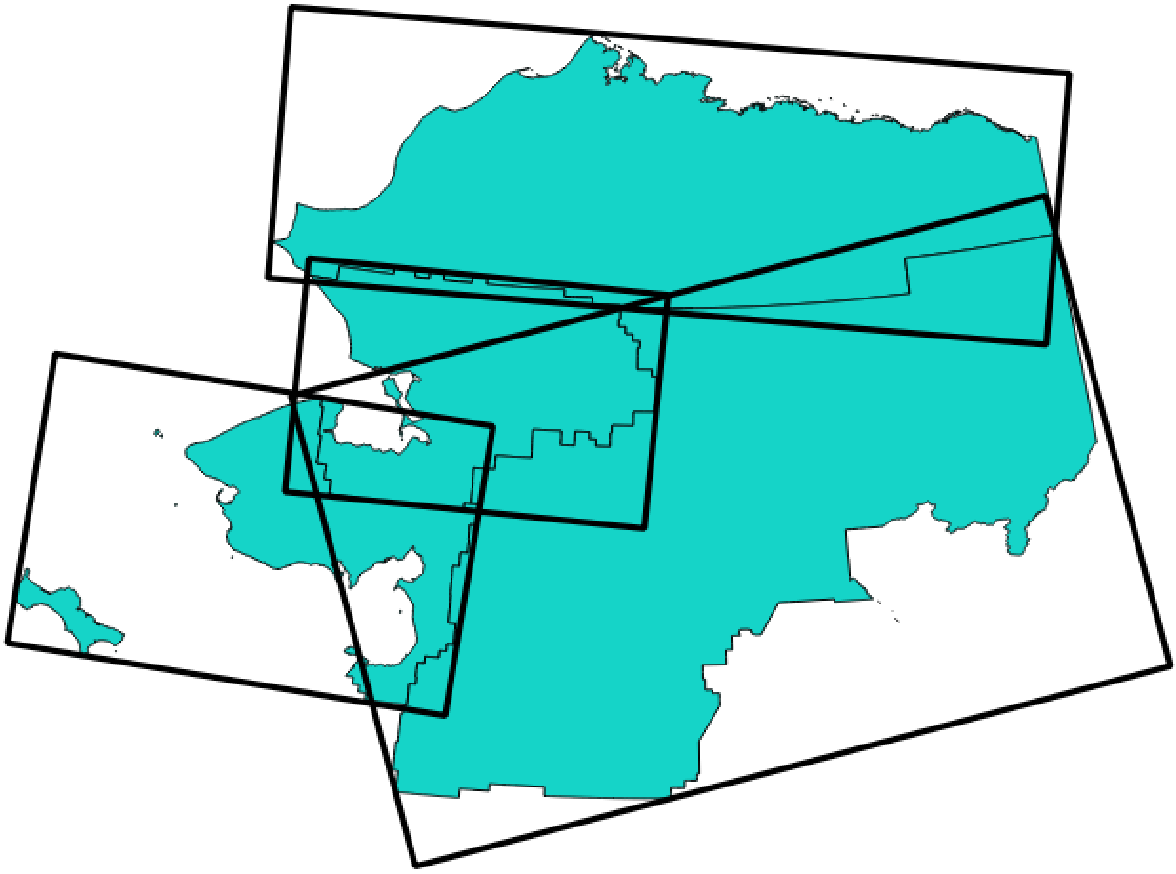


Abb. 23.56: Oriented minimum bounding box

Ausgaben

Label	Name	Type	Beschreibung
Bounding boxes	OUTPUT	[vector: polygon]	The output polygon vector layer.

Python code

Algorithm ID: qgis: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.

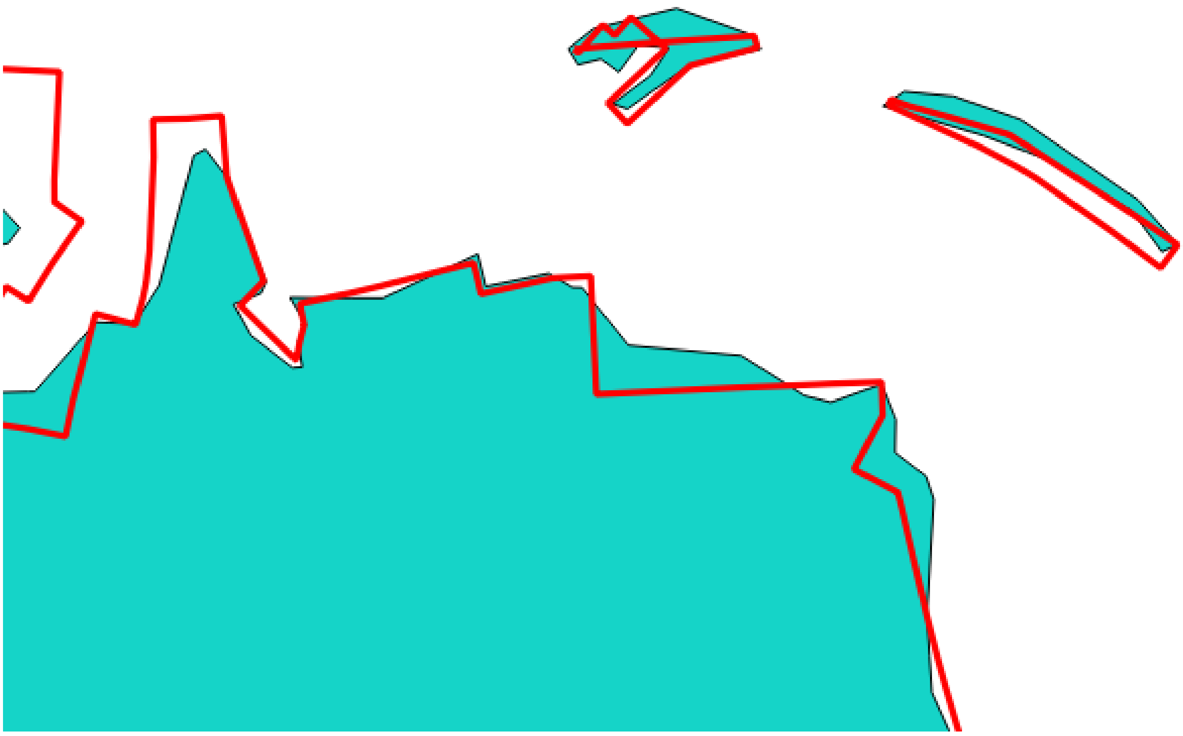


Abb. 23.57: In blue the source layer and in the red orthogonalized result

Allows *features in-place modification*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Maximum angle tolerance (degrees)	`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.
Maximum algorithm iterations	`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.
Orthogonalized	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Orthogonalized	`OUTPUT`	[same as input]	The output polygon vector layer with adjusted angles.

Python code

Algorithm ID: `qgis: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*

Siehe auch:

Centroids

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Create point on surface for each part	`ANGLE_TOLERANCE`	[boolean]	If checked, a point will be created for each part of the geometry.
Point	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output point vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Point	`OUTPUT`	[vector: point]	The output point vector layer.

Python code

Algorithm ID: `qgis: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.

Siehe auch:

Interpolate point on line

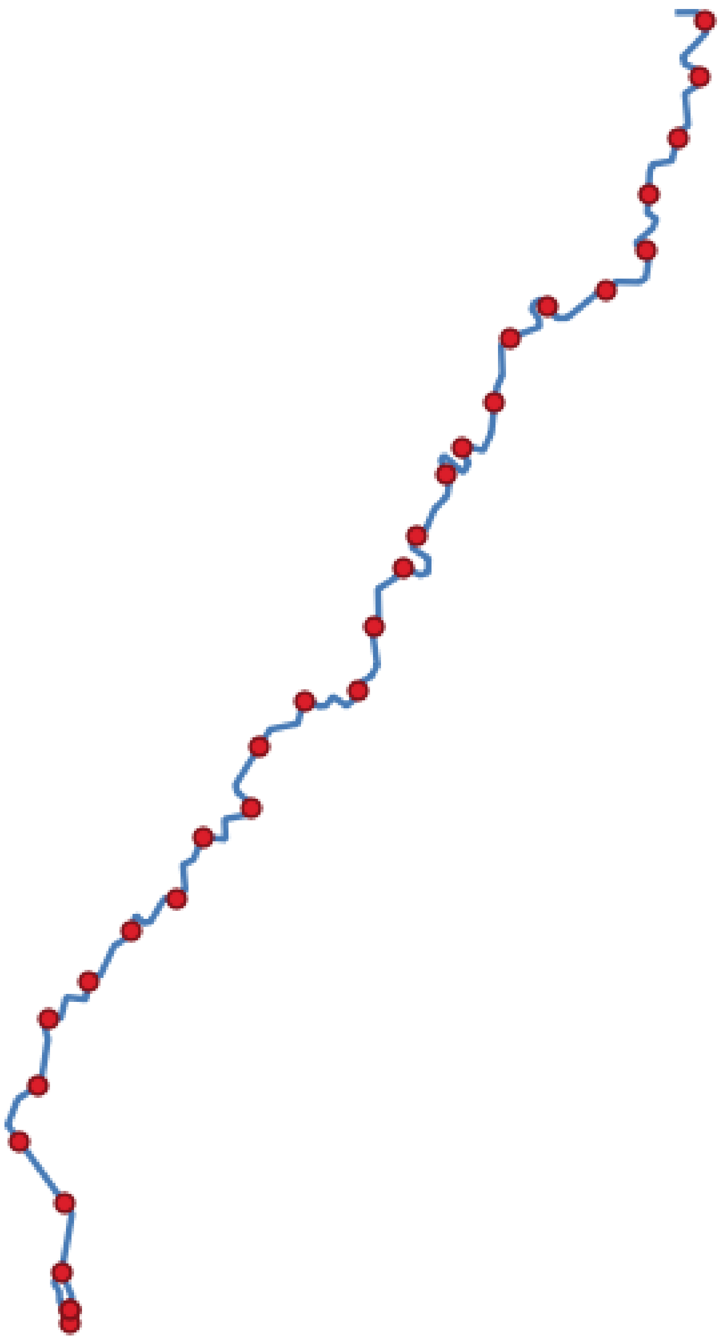


Abb. 23.58: Points created along the source line layer

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	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 shoud be created.
Interpolated points	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated points	`OUTPUT`	[vector: point]	Point vector layer with features placed along lines or polygon boundaries of the input layer.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: point]	Input point vector layer
Minimum distance to other points	`PROXIMITY`	[number] Default: 1.0	Distance below which point features are considered close. Close features are distributed altogether.
Displacement distance	`DISTANCE`	[number] Default: 1.0	Radius of the circle on which close features are placed
Horizontal distribution for two point case	`HORIZONTAL`	[boolean] Default: False	When only two points are identified as close, aligns them horizontally on the circle instead of vertically.
Displaced	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Displaced	`OUTPUT`	[vector: point]	Output point vector layer

Python code

Algorithm ID: `qgis:pointsdisplacement`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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. 23.59: Pole of inaccessibility

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	Input vector layer
Tolerance	`TOLERANCE`	[number] Default: 1.0	Set the tolerance for the calculation
Point	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Point	`OUTPUT`	[vector: point]	The output point vector layer

Python code

Algorithm ID: `qgis: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.

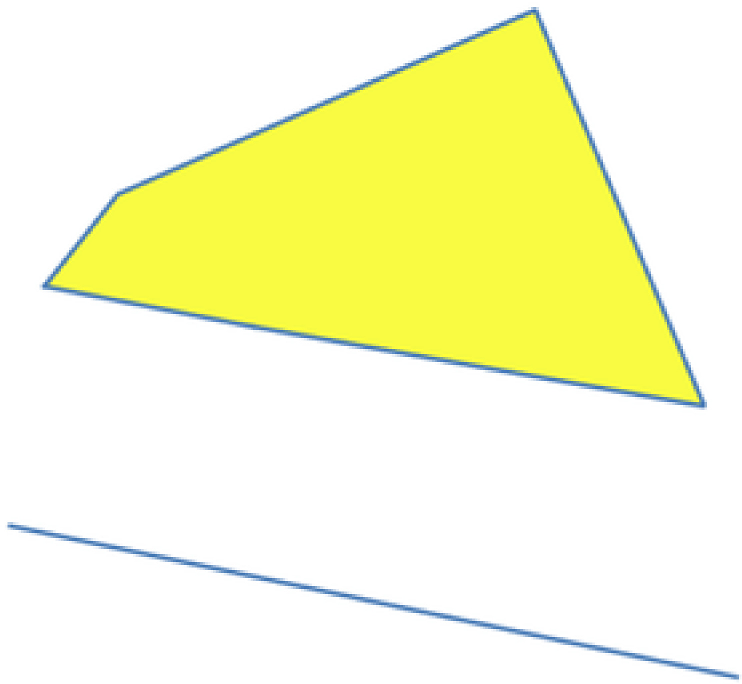


Abb. 23.60: The yellow polygons generated from the closed lines

Bemerkung: The line layer must have closed shapes in order to be transformed into a polygon.

Siehe auch:

Polygons to lines

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Keep table structure of line layer Optional	`KEEP_FIELDS`	[boolean] Default: False	Check to copy the original attributes of the input layer
Polygons from lines	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output polygon vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Polygons from lines	OUTPUT	[vector: polygon]	The output polygon vector layer from lines

Python code

Algorithm ID: qgis: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.

Abb. 23.61: Black lines as the result of the algorithm

Default menu: *Vector ► Geometry Tools*

Siehe auch:

Polygonize

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: polygon]	Input polygon vector layer
Lines	OUTPUT	[vector: line] Default: [Create temporary layer]	Specify the output line vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Lines	OUTPUT	[vector: line]	The output line vector layer from polygons

Python code

Algorithm ID: qgis:polygonstolines

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: point]	Input point vector layer
Bearing (degrees from North)	BEARING	[number] Default: 0.0	Clockwise angle starting from North, in degree (°) unit
Distance	DISTANCE	[number] Default: 1.0	Distance to offset geometries, in layer units
Projected	OUTPUT	[vector: point] Default: [Create temporary layer]	Specify the output point vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Projected	`OUTPUT`	[vector: point]	The output (projected) point vector layer

Python code

Algorithm ID: `qgis: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*

Siehe auch:

Aggregate, *Collect geometries*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Multiparts	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output multipart vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Multiparts	`OUTPUT`	[same as input]	The output multipart vector layer

Python code

Algorithm ID: `qgis: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 (fixed)

Creates a buffer area for all the features in an input layer with different shape choice.

Parameters can vary depending on the shape chosen.

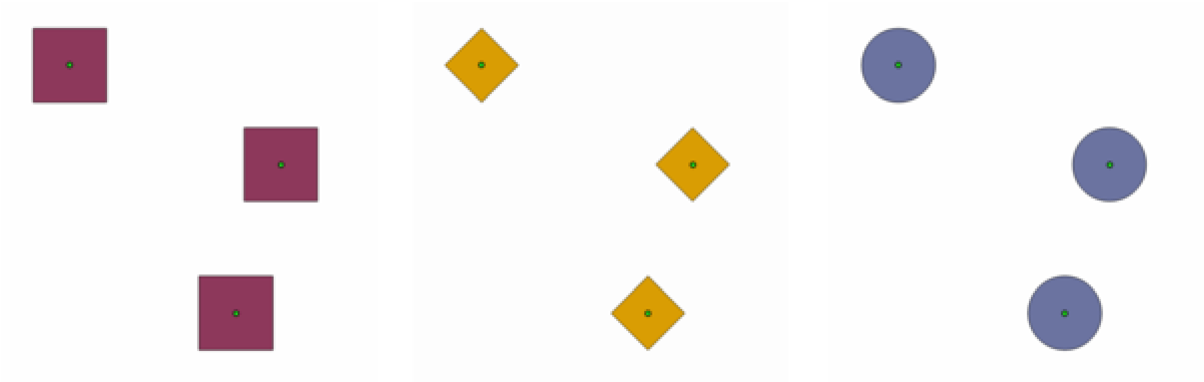


Abb. 23.62: Different buffer shapes

Siehe auch:

Rectangles, ovals, diamonds (variable)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: point]	Input point vector layer
Buffer shape	`SHAPE`	[enumeration]	The shape to use. One of: • 0 — Rectangles • 1 — Ovals • 2 — Diamonds
Width	`WIDTH`	[number] Default: 1.0	Width of the buffer shape
Height	`HEIGHT`	[number] Default: 1.0	Height of the buffer shape
Rotation Optional	`ROTATION`	[number] Default: None	Rotation of the buffer shape
Number of segment	`SEGMENTS`	[number] Default: 36	Number of segments for a full circle (*Ovals* shape)

Fortsetzung auf der nächsten Seite

Tab. 23.88 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Ergebnis	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis	`OUTPUT`	[vector: polygon]	The output vector layer (with the buffer shapes)

Python code

Algorithm ID: `qgis: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.

Rectangles, ovals, diamonds (variable)

Creates a buffer area for all the features in an input layer with different shape choice.

Buffer shape parameters are specified through attribute of the input layer.

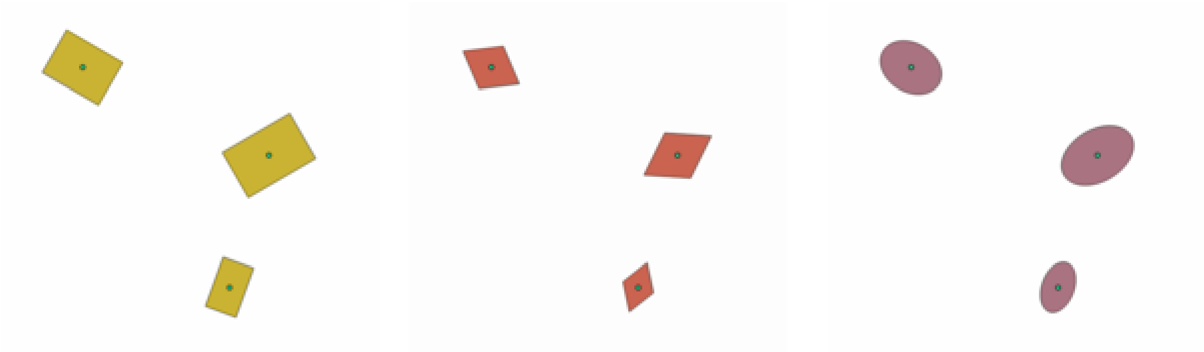


Abb. 23.63: Different buffer shapes with different parameters

Siehe auch:

Rectangles, ovals, diamonds (fixed)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: point]	Input point vector layer
Buffer shape	`SHAPE`	[enumeration] Default: 0	The shape to use. One of: • 0 — Rectangles • 1 — Ovals • 2 — Diamonds
Width field	`WIDTH`	[tablefield: numeric] Default: First	Width of the buffer shape
Height field	`HEIGHT`	[tablefield: numeric] Default: First	Height of the buffer shape
Rotation field Optional	`ROTATION`	[tablefield: numeric]	Rotation of the buffer shape
Number of segment	`SEGMENTS`	[number] Default: 36	Number of segments for a full circle (*Ovals* shape)
Ergebnis	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis	`OUTPUT`	[vector: polygon]	The output vector layer (with the buffer shapes)

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.

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.

Allows *features in-place modification*

Siehe auch:

Extract vertices, *Extract specific vertices*, *Delete duplicate geometries*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Tolerance	`TOLERANCE`	[number] Default: 0.000001	Vertices closer than the specified distance are considered duplicates
Use Z value	`USE_Z_VALUE`	[boolean] Default: False	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.
Cleaned	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Cleaned	`OUTPUT`	[same as input]	The output vector layer (without duplicate vertices)

Python code

Algorithm ID: `qgis: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.

Siehe auch:

Delete duplicate geometries

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer (with non-NULL geometries)
Non null geometries	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the non-NULL geometries. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Null geometries	`NULL_OUTPUT`	[same as input] Default: `[Skip output]`	Specify the output vector layer for the NULL geometries. One of: • Skip Output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Null geometries	`NULL_OUTPUT`	[same as input]	The output vector layer (only NULL geometries)
Non null geometries	`OUTPUT`	[same as input]	The output vector layer (without NULL geometries)

Python code

Algorithm ID: `qgis: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.

Allows *features in-place modification*

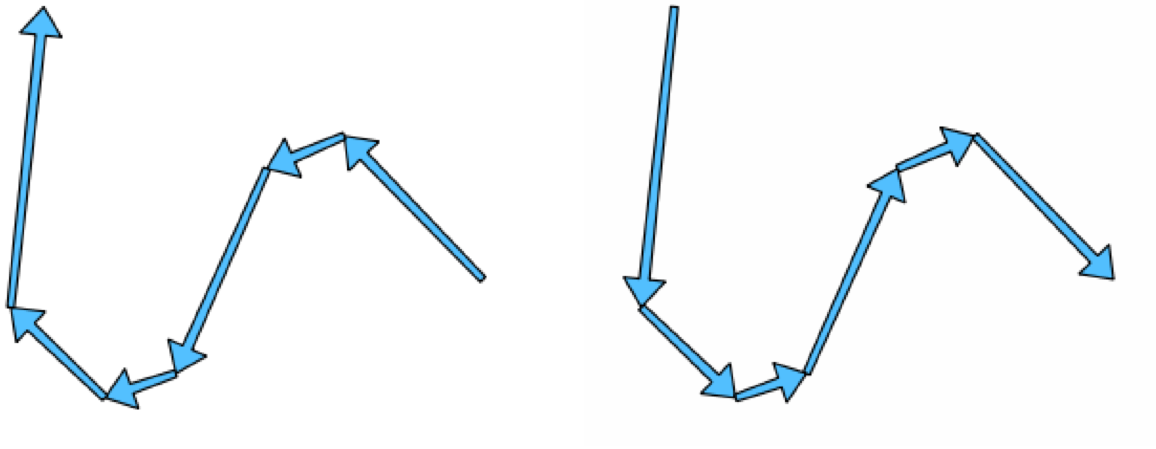


Abb. 23.64: Before and after the direction inversion

Parameter

Label	Name	Type	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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Reversed	`OUTPUT`	[vector: line]	The output line vector layer (with reversed lines)

Python code

Algorithm ID: `qgis: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*

Siehe auch:

Translate, *Swap X and Y coordinates*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Rotation (degrees clockwise)	ANGLE	[number] Default: 0.0	Angle of the rotation in degrees
Rotation anchor point (x, y) 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.
Rotated	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer (with rotated geometries). One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Rotated	OUTPUT	[same as input]	The output vector layer with rotated geometries

Python code

Algorithm ID: qgis: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.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Maximum angle between vertices (degrees)	`ANGLE`	[number] Default: 5.0	Maximum allowed radius angle between vertices on the straightened geometry
Segmentized	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer (with segmentized geometries). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Segmentized	`OUTPUT`	[same as input]	The output vector layer with segmentized geometries

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Maximum offset distance	`DISTANCE`	[number] Default: 1.0	Maximum allowed offset distance between the original curve and the segmentized representation, in the layer units.
Segmentized	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer (with segmentized geometries). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Segmentized	`OUTPUT`	[same as input]	The output vector layer with segmentized geometries

Python code

Algorithm ID: `qgis: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.

Set M value

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.

Tipp: 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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
M Value	`M_VALUE`	[number] Default: 0.0	M value to assign to the feature geometries
M Added	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
M Added	`OUTPUT`	[same as input]	The output vector layer (with M values assigned to the geometries)

Python code

Algorithm ID: `qgis: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.

Siehe auch:

Drape (set Z value from raster), *Set M value*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Raster layer	`RASTER`	[raster]	Raster layer with M values
Band number	`BAND`	[raster band] Default: 1	The raster band from which the M values are taken
Value for nodata or non-intersecting vertices	`NODATA`	[number] Default: 0.0	Value to use in case the vertex does not intersect (a valid pixel of) the raster
Scale factor	`SCALE`	[number] Default: 1.0	Scaling value: the band values are multiplied by this value.
Updated	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer (with updated M values). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Updated	`OUTPUT`	[same as input]	The output vector layer (with updated M values)

Python code

Algorithm ID: `qgis: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.

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), *Set M value*, *Drop M/Z values*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Z Value	`Z_VALUE`	[number] Default: 0.0	Z value to assign to the feature geometries
Z Added	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Z Added	`OUTPUT`	[same as input]	The output vector layer (with Z values assigned)

Python code

Algorithm ID: `qgis: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.

Allows *features in-place modification*

Default menu: *Vector ⏵ Geometry Tools*

Siehe auch:

Smooth, *Densify by count*, *Densify by interval*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line, polygon]	Input line or polygon vector layer
Simplification method	`METHOD`	[enumeration] Default: 0	Simplification method. One of: • 0 — Distance (Douglas-Peucker) • 1 — Snap to grid • 2 — Area (Visvalingam)
Tolerance	`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.
Simplified	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (simplified) vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Simplified	`OUTPUT`	[same as input]	The output (simplified) vector layer

Abb. 23.65: Clockwise from top left: source layer and increasing simplification tolerances

Python code

Algorithm ID: `qgis: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. 23.66: Left versus right side buffer on the same vector line layer

Siehe auch:

Buffer

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Distance	`DISTANCE`	[number] Default: 10.0	Buffer distance.
Side	`SIDE`	[enumeration]	Which side to create the buffer on. One of: • 0 – Left • 1 – Right
Segments	`SEGMENTS`	[number] Default: 8	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.

Fortsetzung auf der nächsten Seite

Tab. 23.93 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Join style	`JOIN_STYLE`	[enumeration]	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line. Options are: • 0 — Round • 1 — Miter • 2 — Bevel
Miter limit	`MITER_LIMIT`	[number] Default: 2.0	Controls the maximum distance from the offset curve to use when creating a mitered join (only applicable for miter join styles). Minimum: 1.0
Buffer	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output (buffer) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffer	`OUTPUT`	[vector: polygon]	Output (buffer) polygon layer

Python code

Algorithm ID: `qgis: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.

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.

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*

Siehe auch:

Simplify, *Densify by count*, *Densify by interval*

Abb. 23.67: Increasing number of iterations causes smoother geometries

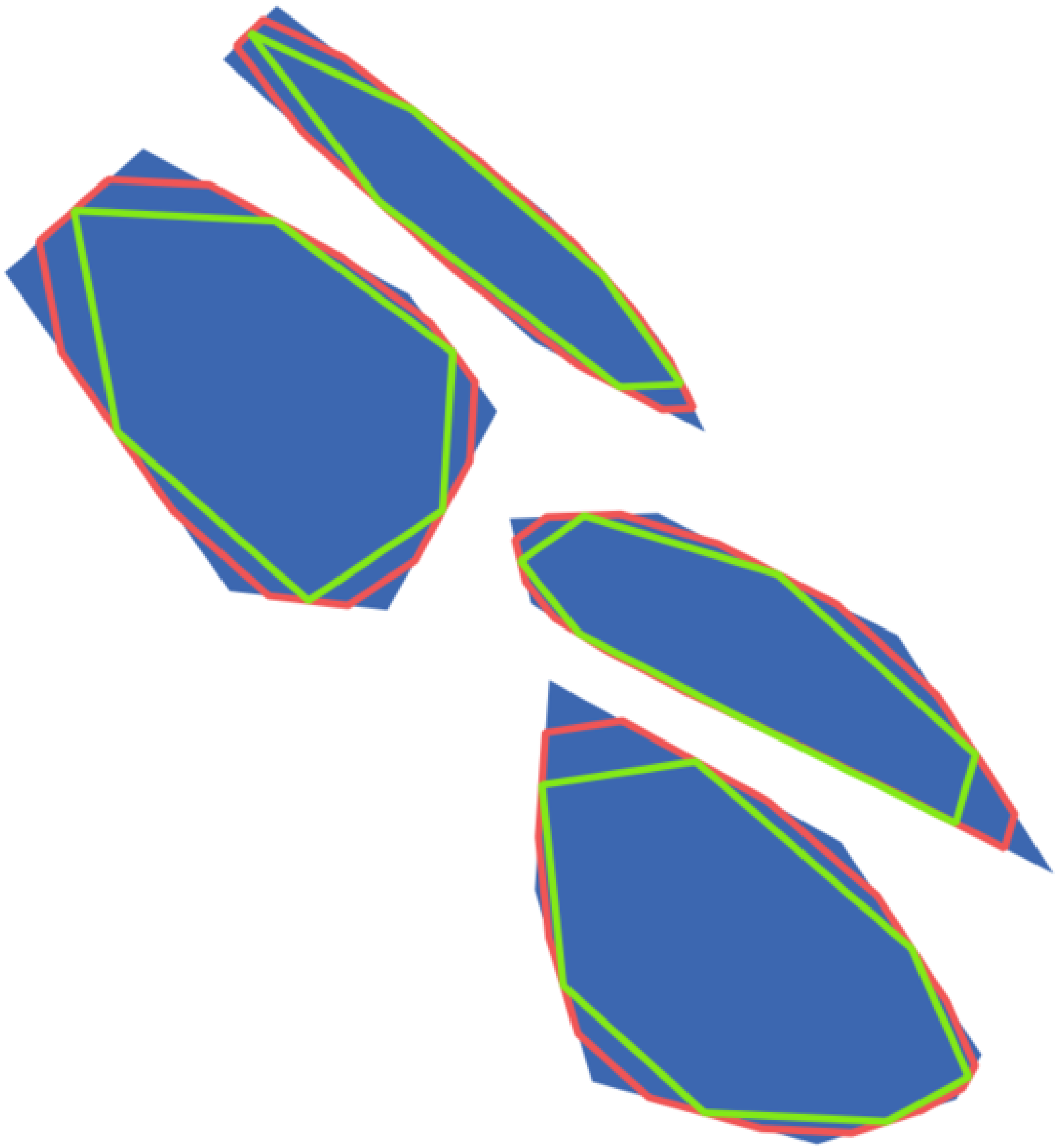


Abb. 23.68: Blue: the input layer. Offset 0.25 gives the red line, while offset 0.50 gives the green line.

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line, polygon]	Input line or polygon vector layer
Iterations	ITERATIONS	[number] Default: 1	Increasing the number of iterations will give smoother geometries (and more vertices).
Offset	OFFSET	[number] Default: 0.25	Increasing values will *move* the smoothed lines / boundaries further away from the input lines / boundaries.
Maximum node angle to smooth	MAX_ANGLE	[number] Default: 180.0	Every node below this value will be smoothed
Smoothed	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output (smoothed) layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Smoothed	OUTPUT	[same as input]	Output (smoothed) vector layer

Python code

Algorithm ID: qgis: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.

Siehe auch:

Snap points to grid

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Reference layer	`REFERENCE_LAYER`	[vector: any]	Vector layer to snap to
Tolerance	`TOLERANCE`	[number] Default: 10.0	Control how close input vertices need to be to the reference layer geometries before they are snapped.
*Behavior**	`BEHAVIOR`	[enumeration] Default: 0	Snapping can be done to an existing node or a segment (its closest point to the vertex to move). Available snapping options: • 0 — Prefer aligning nodes, insert extra vertices where required • 1 — Prefer closest point, insert extra vertices where required • 2 — Prefer aligning nodes, don't insert new vertices • 3 — Prefer closest point, don't insert new vertices • 4 — Move end points only, prefer aligning nodes • 5 — Move end points only, prefer closest point • 6 — Snap end points to end points only • 7 — Snap to anchor nodes (single layer only)
Snapped geometry	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (snapped) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Snapped geometry	`OUTPUT`	[same as input]	Output (snapped) vector layer

Python code

Algorithm ID: `qgis: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.

Bemerkung: Snapping to grid may generate an invalid geometry in some corner cases.

Allows *features in-place modification*

Siehe auch:

Snap geometries to layer

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
X Grid Spacing	`HSPACING`	[number] Default: 1.0	Grid spacing on the X axis
Y Grid Spacing	`VSPACING`	[number] Default: 1.0	Grid spacing on the Y axis
Z Grid Spacing	`ZSPACING`	[number] Default: 0.0	Grid spacing on the Z axis
M Grid Spacing	`MSPACING`	[number] Default: 0.0	Grid spacing on the M axis
Snapped	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (snapped) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Snapped	`OUTPUT`	[same as input]	Output (snapped) vector layer

Python code

Algorithm ID: `qgis: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.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	The input line vector layer
Maximum line length	`LENGTH`	[number] Default: 10.0	The maximum length of a line in the output.
Split	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output line vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Split	`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.

Python code

Algorithm ID: `qgis: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.

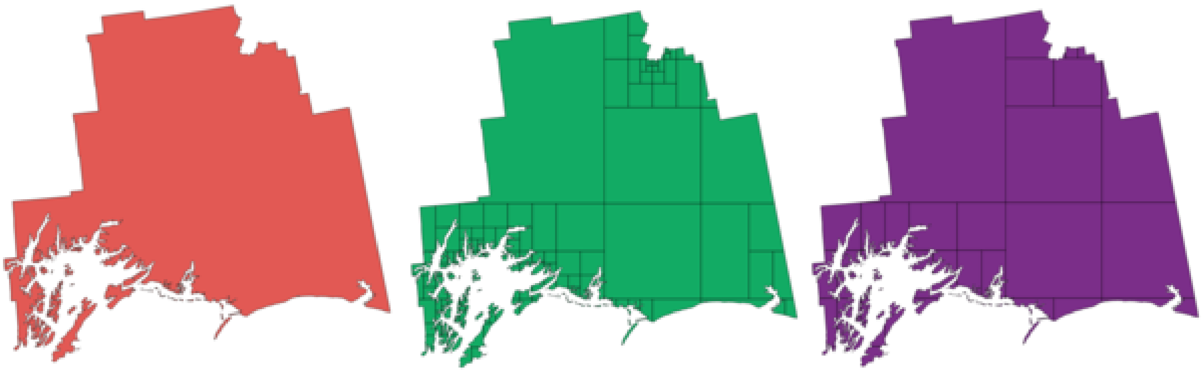


Abb. 23.69: Left the input layer, middle maximum nodes value is 100 and right maximum value is 200

Bemerkung: Subdividing a geometry can generate geometry parts that may not be valid and may contain self-intersections.

Allows *features in-place modification*

Siehe auch:

Explode lines, *Line substring*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Maximum nodes in parts	`MAX_NODES`	[number] Default: 256	Maximum number of vertices each new geometry part is allowed to have. Fewer *sub-parts* for higher values.
Subdivided	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output (subdivided) vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Subdivided	`OUTPUT`	[same as input]	Output vector layer

Python code

Algorithm ID: `qgis: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*

Siehe auch:

Translate, *Rotate*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Swapped	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Swapped	`OUTPUT`	[same as input]	Output (swapped) vector layer

Python code

Algorithm ID: `qgis: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.

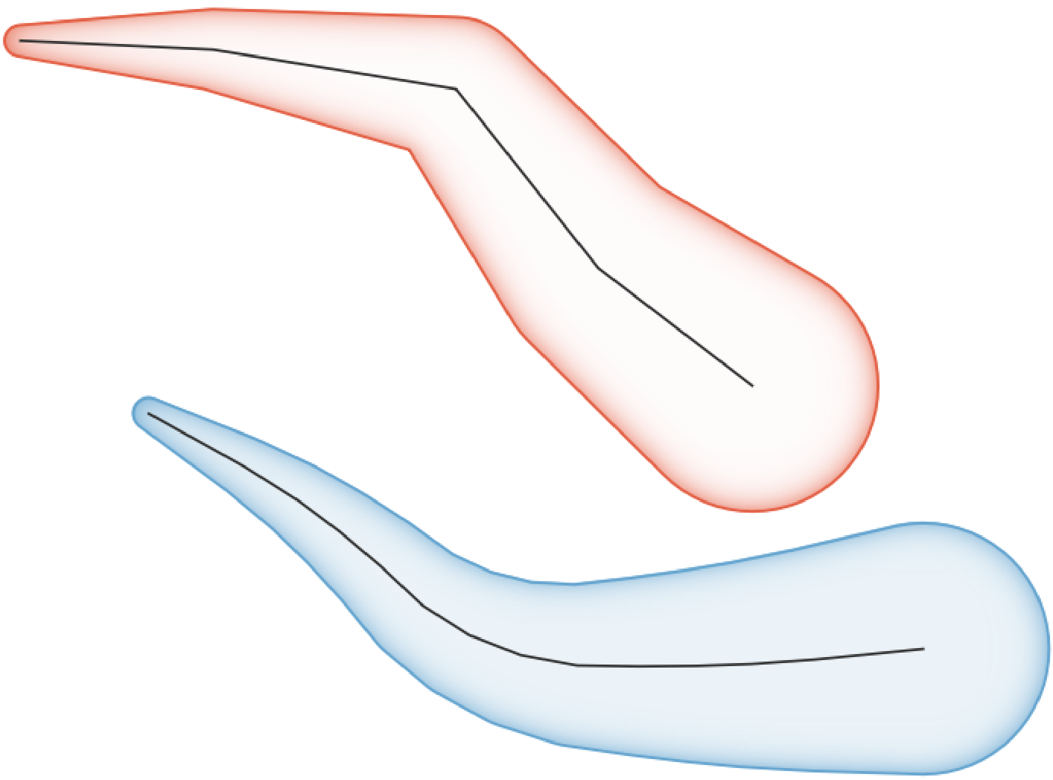


Abb. 23.70: Tapered buffer example

Siehe auch:

Variable width buffer (by M value), *Buffer*, *Create wedge buffers*

Parameter

Label	Name	Type	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.
Buffered	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output (buffer) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffered	`OUTPUT`	[vector: polygon]	Output (buffer) polygon layer

Python code

Algorithm ID: `qgis: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.

Allows *features in-place modification*

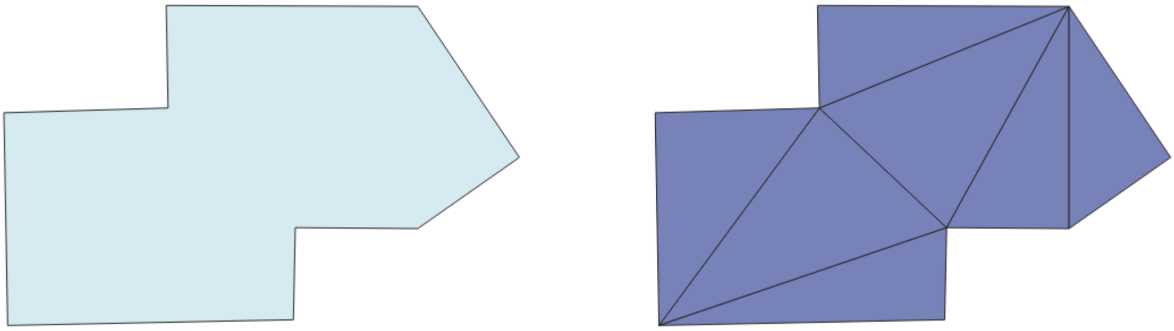


Abb. 23.71: Tessellated polygon (right)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: polygon]	Input polygon vector layer
Tesselated	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Tesselated	`OUTPUT`	[vector: polygon]	Output multipolygonZ layer

Python code

Algorithm ID: `qgis: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)

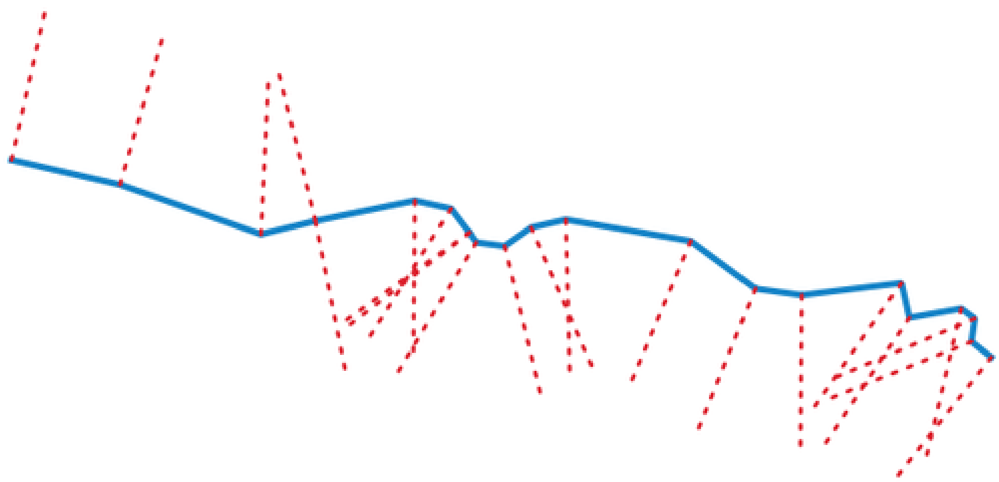


Abb. 23.72: Dashed red lines represent the transect of the input line layer

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Length of the transect	`LENGTH`	[number] Default: 5.0	Length in map unit of the transect
Angle in degrees from the original line at the vertices	`ANGLE`	[number] Default: 90.0	Change the angle of the transect
Side to create the transect	`SIDE`	[enumeration]	Choose the side of the transect. Available options are: • 0 — Left • 1 — Right • 2 — Both

Fortsetzung auf der nächsten Seite

Tab. 23.98 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Transect	`OUTPUT`	[vector: line] Default: `[Create temporary layer]`	Specify the output line layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Transect	`OUTPUT`	[vector: line]	Output line layer

Python code

Algorithm ID: `qgis: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.

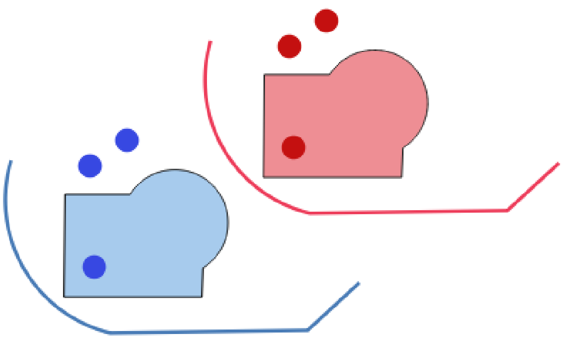


Abb. 23.73: Dashed lines represent the translated geometry of the input layer

Allows *features in-place modification*

Siehe auch:

Array of translated features, *Offset lines*, *Rotate*, *Swap X and Y coordinates*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Offset distance (x-axis)	DELTA_X	[number] Default: 0.0	Displacement to apply on the X axis
Offset distance (y-axis)	DELTA_Y	[number] Default: 0.0	Displacement to apply on the Y axis
Offset distance (z-axis)	DELTA_Z	[number] Default: 0.0	Displacement to apply on the Z axis
Offset distance (m values)	DELTA_M	[number] Default: 0.0	Displacement to apply on the M axis
Translated	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Translated	OUTPUT	[same as input]	Output vector layer

Python code

Algorithm ID: qgis: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 distance buffer

Computes a buffer area for all the features in an input layer.

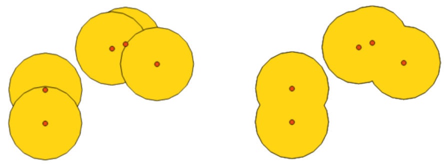
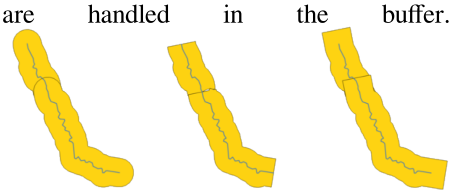
The size of the buffer for a given feature is defined by an attribute, so it allows different features to have different buffer sizes.

Bemerkung: This algorithm is only available from the *Graphical modeler*.

Siehe auch:

Buffer

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Distance field	DISTANCE	[tablefield: numeric]	Attribute for the distance radius of the buffer
Segments	SEGMENTS	[number] Vorgabe: *5*	Controls the number of line segments to use to approximate a quarter circle when creating rounded offsets.
Dissolve result	DISSOLVE	[boolean] Vorgabewert: *False*	Choose to dissolve the final buffer, resulting in a single feature covering all input features.  Abb. 23.74: Normal and dissolved buffer
End cap style	END_CAP_STYLE	[enumeration]	Controls how line endings are handled in the buffer.  Abb. 23.75: Round, flat and square cap styles
Join style	JOIN_STYLE	[enumeration]	Specifies whether round, miter or beveled joins should be used when offsetting corners in a line.
Miter limit	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.

Ausgaben

Label	Name	Type	Beschreibung
Buffer	OUTPUT	[vector: polygon]	Buffer polygon vector layer.

Python code

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.

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.

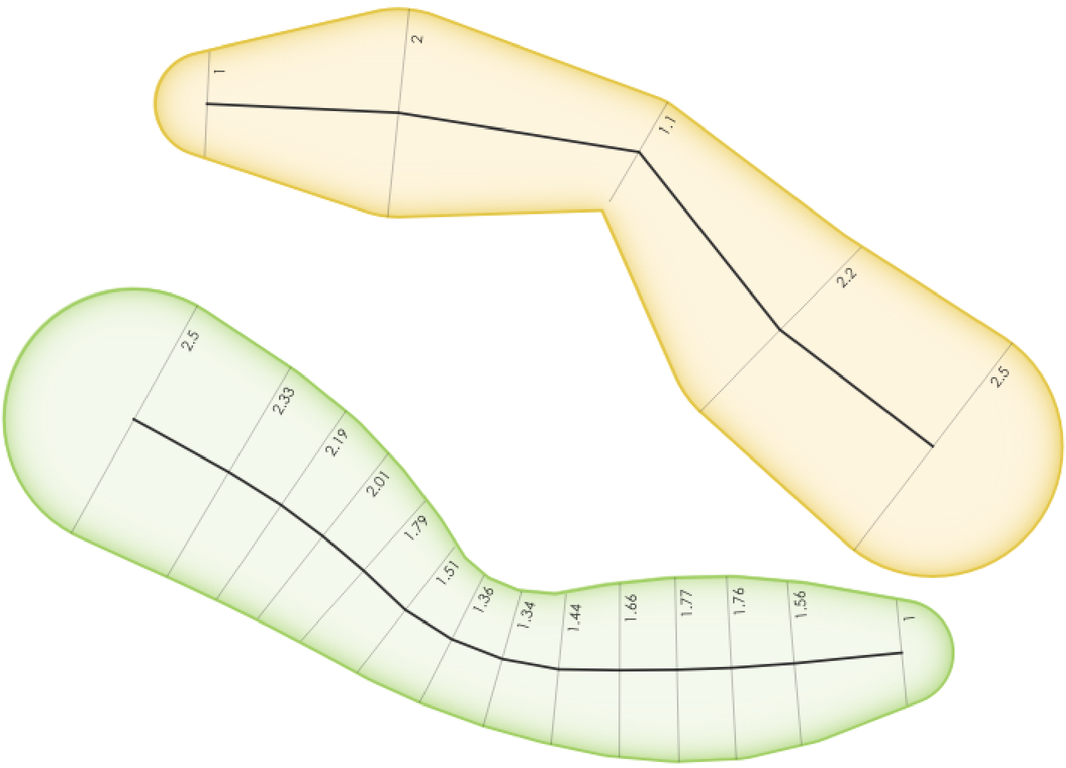


Abb. 23.76: Variable buffer example

Siehe auch:

Tapered buffers, *Buffer*, *Set M value*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	Input line vector layer
Segments	`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).
Buffered	`OUTPUT`	[vector: polygon] Default: `[Create temporary layer]`	Specify the output (buffer) layer. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffered	OUTPUT	[vector: polygon]	Variable buffer polygon layer

Python code

Algorithm ID: `qgis: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.

Default menu: *Vector ▸ Geometry Tools*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: point]	Input point vector layer
Buffer region (% of extent)	BUFFER	[number] Default: 0.0	The extent of the output layer will be this much bigger than the extent of the input layer
Voronoi polygons	OUTPUT	[vector: polygon] Default: `[Create temporary layer]`	Specify the output layer (with the Voronoi polygons). One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Voronoi polygons	OUTPUT	[vector: polygon]	Voronoi polygons of the input point vector layer

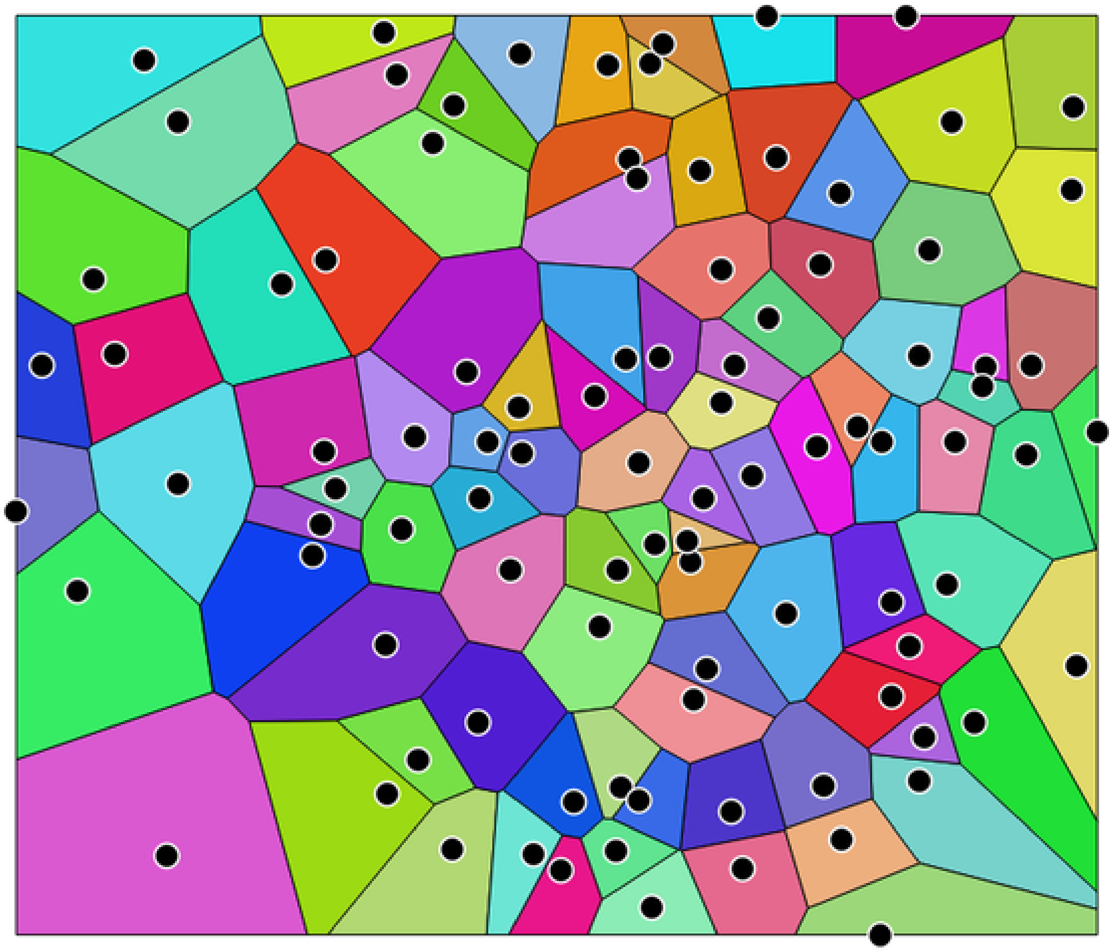


Abb. 23.77: Voronoi polygons

Python code

Algorithm ID: `qgis: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.

23.1.16 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: Feature modification**
>
> The attributes of the features are **not modified**, although properties such as area or length of the features will be modified by the clipping 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.

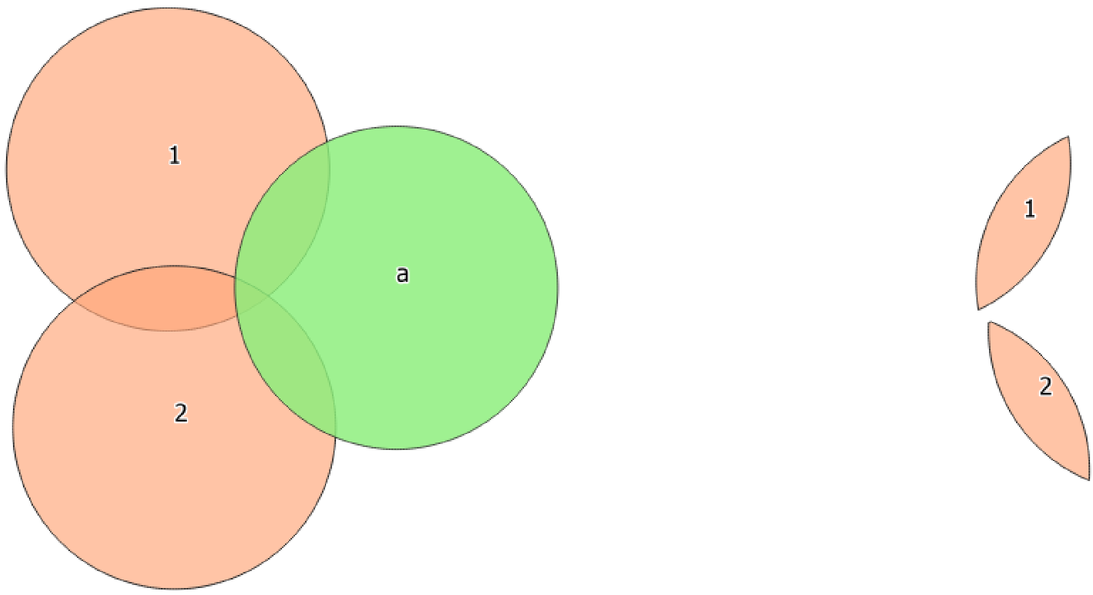


Abb. 23.78: Clipping operation between a two-features input layer and a single feature overlay layer (left) - resulting features are moved for clarity (right)

Allows *features in-place modification*

Default menu: *Vector □ Geoprocessing Tools*

Siehe auch:

Intersection, *Difference*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Layer containing the features to be clipped
Overlay layer	`OVERLAY`	[vector: polygon]	Layer containing the clipping features
Clipped	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Clipped	`OUTPUT`	[same as input]	Layer containing features from the input layer split by the overlay layer.

Python code

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.

Attributes are not modified (see *warning*).

Allows *features in-place modification*

Default menu: *Vector ⍰ Geoprocessing Tools*

Siehe auch:

Symmetrical difference, *Clip*

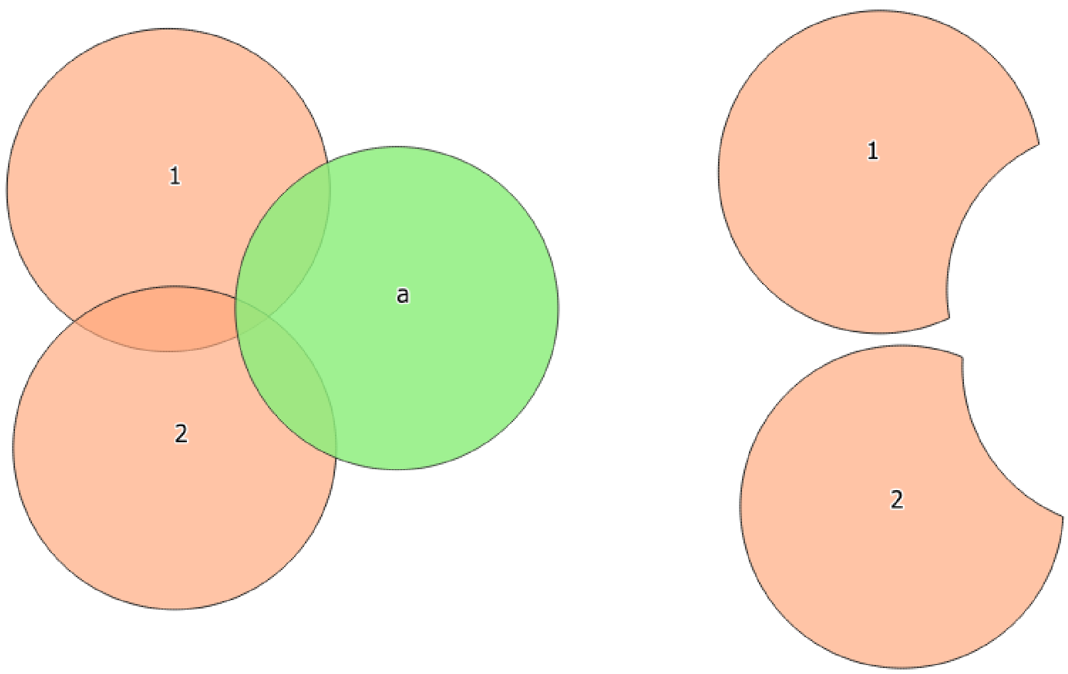


Abb. 23.79: Difference operation between a two-features input layer and a single feature overlay layer (left) - resulting features are moved for clarity (right)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Layer to extract (parts of) features from.
Overlay layer	`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.
Difference	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Difference	OUTPUT	[same as input]	Layer containing (parts of) features from the input layer not overlapping the overlay layer.

Python code

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.

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.

Siehe auch:

Clip

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Layer to extract (parts of) features from.
Extent (xmin, xmax, ymin, ymax)	`EXTENT`	[extent]	Extent for clipping.
Clip features to extent	`CLIP`	[boolean] Default: False	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.
Extracted	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the layer to contain the features from the input layer that are inside the clip extent. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extracted	`OUTPUT`	[same as input]	Layer containing the clipped features.

Python code

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.

Attributes are not modified (see *warning*).

Default menu: *Vector ▸ Geoprocessing Tools*

Siehe auch:

Clip, *Difference*

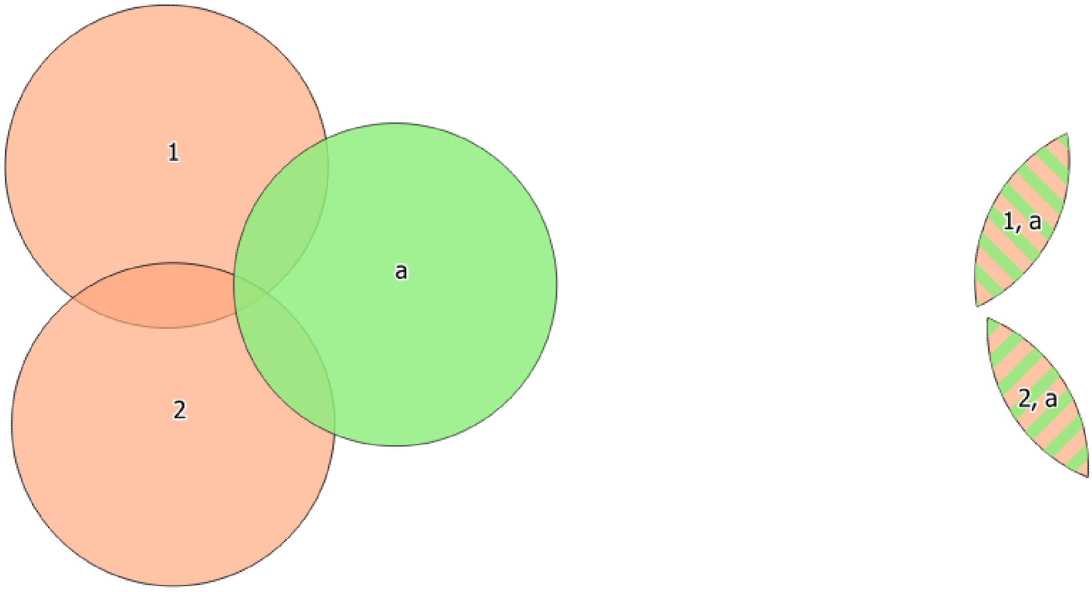


Abb. 23.80: The intersection operation: A two-features input layer and a single feature overlay layer (left) - resulting features are moved for clarity (right)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Layer to extract (parts of) features from.
Overlay layer	`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.
Input fields to keep (leave empty to keep all fields) 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.
Overlay fields to keep (leave empty to keep all fields) 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.
Overlay fields prefix Optional	`OVERLAY_FIELDS_PREFIX`	[string]	Prefix to add to the field names of the intersect layer's fields to avoid name collisions with fields in the input layer.
Intersection	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Intersection	`OUTPUT`	[same as input]	Layer containing (parts of) features from the input layer that overlap the overlay layer.

Line intersections

Creates point features where the lines from the two layers intersect.

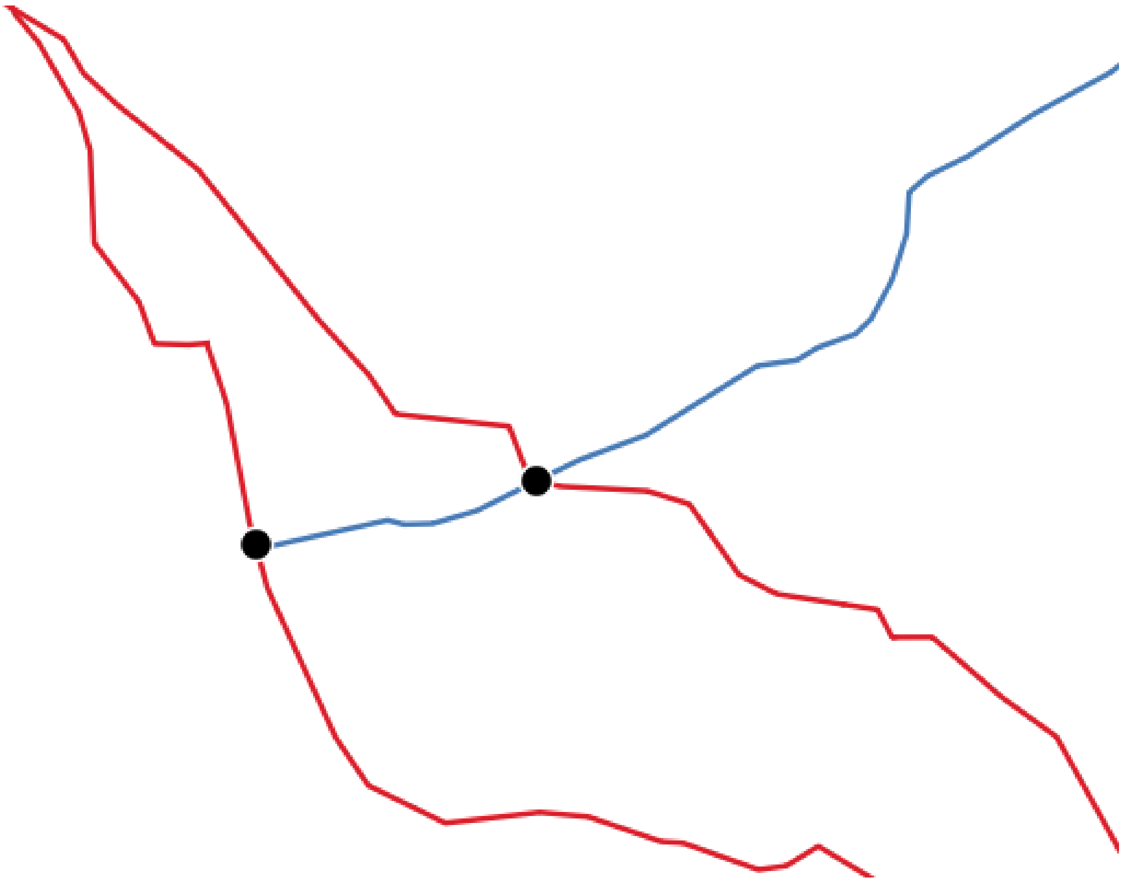


Abb. 23.81: Points of intersection

Default menu: *Vector ⊡ Analysis Tools*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line]	Input line layer.
Intersect layer	INTERSECT	[vector: line]	Layer to use to find line intersections.
Input fields to keep (leave empty to keep all fields) 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.
Intersect fields to keep (leave empty to keep all fields) Optional	INTERSECT_FIELDS	[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.
Intersect fields prefix Optional	OVERLAY_FIELDS_PREFIX	[string]	Prefix to add to the field names of the intersect layer's fields to avoid name collisions with fields in the input layer.

Fortsetzung auf der nächsten Seite

Tab. 23.103 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Intersection	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Intersections	`OUTPUT`	[vector: point]	Point vector layer with the intersections.

Python code

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 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.

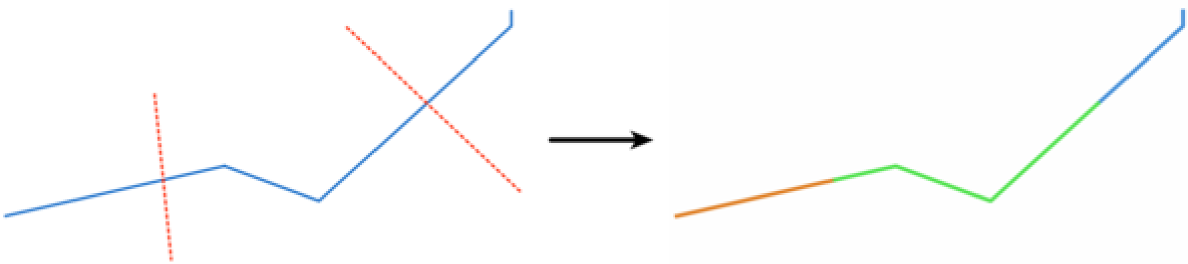


Abb. 23.82: Split lines

Allows *features in-place modification*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line, polygon]	Layer containing the lines or polygons to split.
Split layer	LINES	[vector: line]	Line layer whose lines 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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Split	OUTPUT	[same as input]	Output vector layer with split lines or polygons from input layer.

Python code

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.

Attributes are not modified (see *warning*).

Default menu: *Vector ► Geoprocessing Tools*

Siehe auch:

Difference, *Clip*, *Intersection*

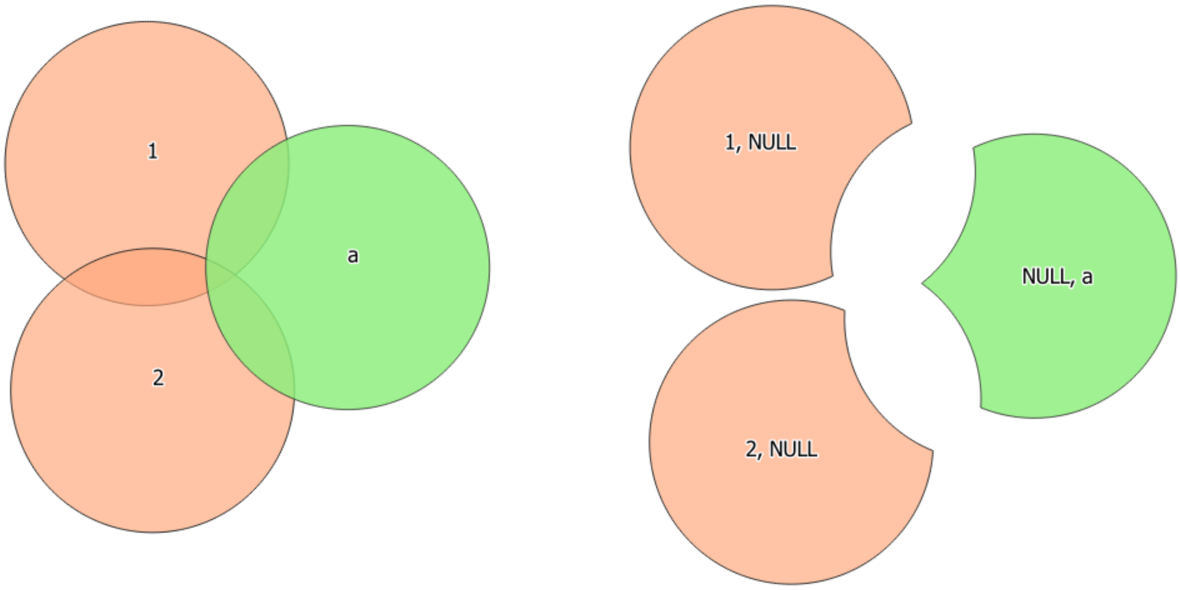


Abb. 23.83: Symmetrical difference operation between a two-features input layer and a single feature overlay layer (left) - resulting features are moved for clarity (right)

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	First layer to extract (parts of) features from.
Overlay layer	`OVERLAY`	[vector: any]	Second layer to extract (parts of) features from. Ideally the geometry type should be the same as input layer.
Overlay fields prefix Optional	`OVERLAY_FIELDS_PREFIX`	[string]	Prefix to add to the field names of the overlay layer's fields to avoid name collisions with fields in the input layer.
Symmetrical difference	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Symmetrical difference	OUTPUT	[same as input]	Layer containing (parts of) features from each layer not overlapping the other layer.

Python code

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.

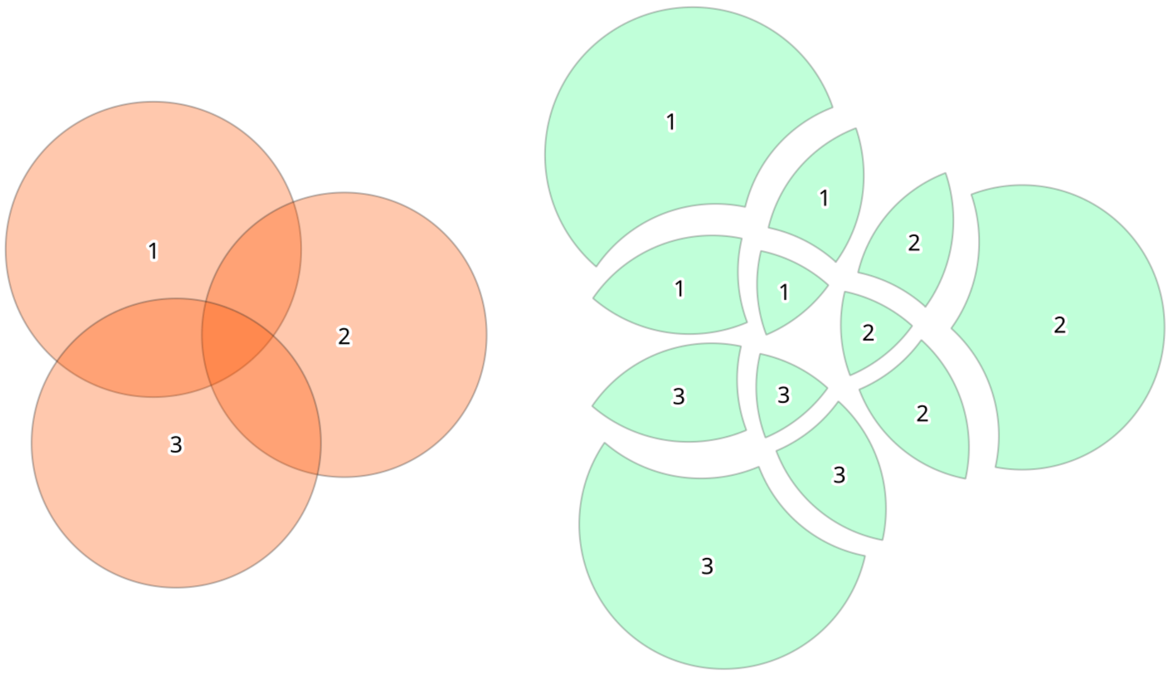


Abb. 23.84: Union operation with a single input layer of three overlapping features (left) - resulting features are moved for clarity (right)

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. 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.

Bemerkung: For `union(A,B)` algorithm, if there are overlaps among geometries of layer A or among geometries

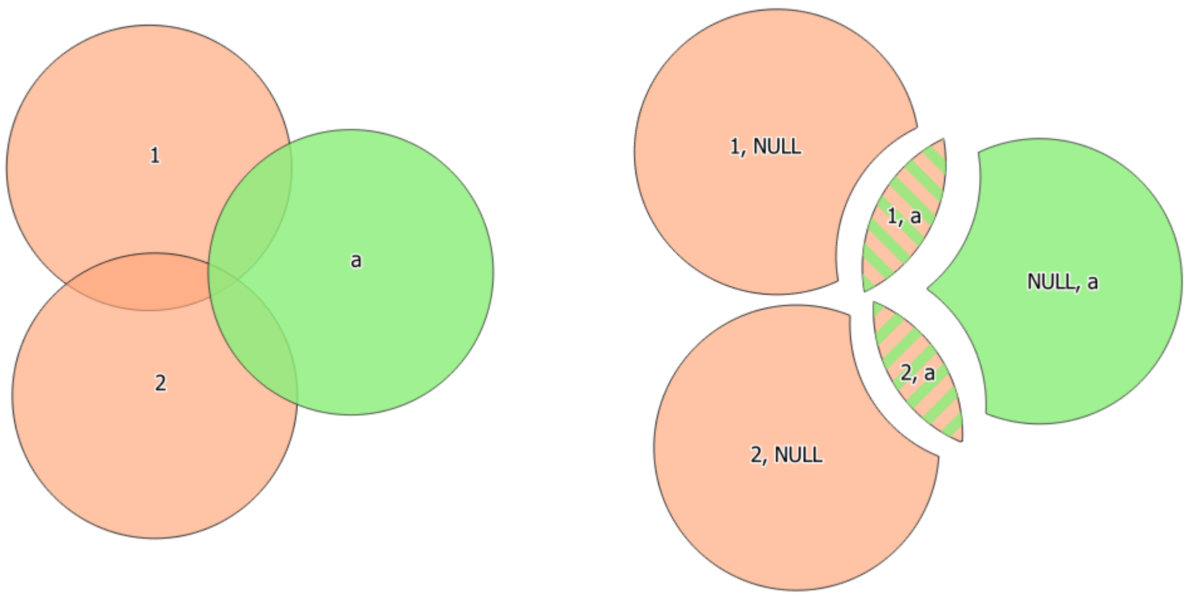


Abb. 23.85: Union operation between a two-features input layer and a single feature overlay layer (left) - resulting features are moved for clarity (right)

of layer B, these are not resolved: you need to do `union(union(A,B))` to resolve all overlaps, i.e. run single layer `union(X)` on the produced result `X=union(A,B)`.

Default menu: *Vector ▸ Geoprocessing Tools*

Siehe auch:

Clip, *Difference*, *Intersection*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer to split at any intersections.
Overlay layer Optional	`OVERLAY`	[vector: any]	Layer that will be combined to the first one. Ideally the geometry type should be the same as input layer.
Overlay fields prefix Optional	`OVERLAY_FIELDS_PREFIX`	[string]	Prefix to add to the field names of the overlay layer's fields to avoid name collisions with fields in the input layer.
Union	`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: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Union	OUTPUT	[same as input]	Layer containing all the overlapping and non-overlapping parts from the processed layer(s).

Python code

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.

23.1.17 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Layer to extract features from.
Selection attribute	FIELD	[tablefield: any]	Filtering field of the layer
Operator	OPERATOR	[enumeration] Default: 0	Many different operators are available: • 0 — = • 1 — ≠ • 2 — > • 3 — >= • 4 — < • 5 — <= • 6 — begins with • 7 — contains • 8 — is null • 9 — is not null • 10 — does not contain
Value Optional	VALUE	[string]	Value to be evaluated

Fortsetzung auf der nächsten Seite

Tab. 23.104 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Extracted (attribute)	`OUTPUT`	[same as input] Default: `[Create Temporary Layer]`	Specify the output vector layer for matching features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Extracted (non-matching)	`FAIL_OUTPUT`	[same as input] Default: `[Skip output]`	Specify the output vector layer for non-matching features. One of: • Skip Output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table…

Ausgaben

Label	Name	Type	Beschreibung
Extracted (attribute)	`OUTPUT`	[same as input]	Vector layer with matching features from the input layer
Extracted (non-matching)	`FAIL_OUTPUT`	[same as input]	Vector layer with non-matching features from the input layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Expression	`EXPRESSION`	[expression]	Expression to filter the vector layer
Matching features	`OUTPUT`	[same as input] Default: `[Create Temporary Layer]`	Specify the output vector layer for matching features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.
Non-matching	`FAIL_OUTPUT`	[same as input] Default: `[Skip output]`	Specify the output vector layer for non-matching features. One of: • Skip Output • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table…

Ausgaben

Label	Name	Type	Beschreibung
Matching features	`OUTPUT`	[same as input]	Vector layer with matching features from the input layer
Non-matching	`FAIL_OUTPUT`	[same as input]	Vector layer with non-matching features from the input layer

Python code

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.

Available geometric predicates are:

Intersect Tests whether a geometry intersects another. Returns 1 (true) if the geometries spatially intersect (share any portion of space - overlap or touch) and 0 if they don't. In the picture above, this will select circles 1, 2 and 3.

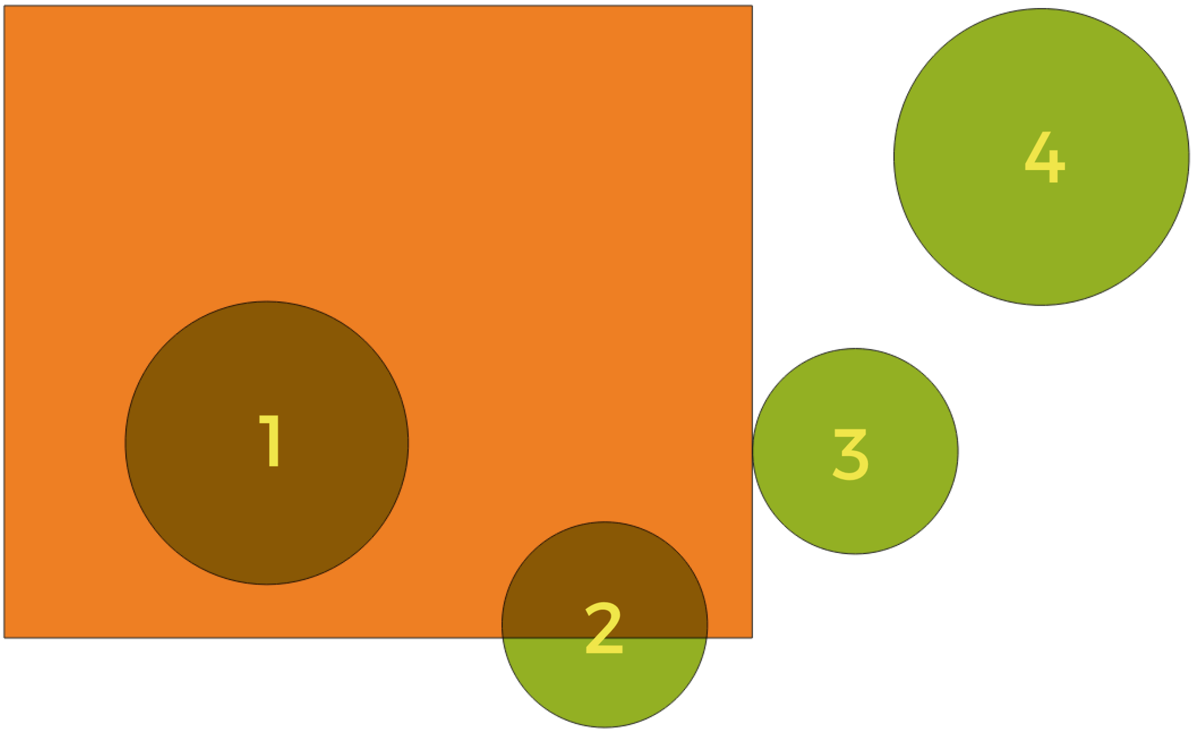


Abb. 23.86: In this example, the dataset from which we want to select (the *source vector layer*) consists of the green circles, the orange rectangle is the dataset that it is being compared to (the *intersection vector layer*).

Contain Returns 1 (true) if and only if no points of b lie in the exterior of a, and at least one point of the interior of b lies in the interior of a. In the picture, no circle is selected, but the rectangle would be if you would select it the other way around, as it contains a circle completely. This is the opposite of *are within*.

Disjoint Returns 1 (true) if the geometries do not share any portion of space (no overlap, not touching). Only circle 4 is selected.

Equal Returns 1 (true) if and only if geometries are exactly the same. No circles will be selected.

Touch Tests whether a geometry touches another. Returns 1 (true) if the geometries have at least one point in common, but their interiors do not intersect. Only circle 3 is selected.

Overlap Tests whether a geometry overlaps another. Returns 1 (true) if the geometries share space, are of the same dimension, but are not completely contained by each other. Only circle 2 is selected.

Are within Tests whether a geometry is within another. Returns 1 (true) if geometry a is completely inside geometry b. Only circle 1 is selected.

Cross Returns 1 (true) if the supplied geometries have some, but not all, interior points in common and the actual crossing is of a lower dimension than the highest supplied geometry. For example, a line crossing a polygon will cross as a line (selected). Two lines crossing will cross as a point (selected). Two polygons cross as a polygon (not selected).

Siehe auch:

Auswahl nach der Lage

Parameter

Label	Name	Type	Beschreibung
Extract features from	`INPUT`	[vector: any]	Input vector layer
Where the features (geometric predicate)	`PREDICATE`	[enumeration] [list] Default: [0]	Spatial condition for the selection. One or more of: • 0 — intersect • 1 — contain • 2 — disjoint • 3 — equal • 4 — touch • 5 — overlap • 6 — are within • 7 — cross If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.
By comparing to the features from	`INTERSECT`	[vector: any]	Intersection vector layer
Extracted (location)	`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: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table…

Ausgaben

Label	Name	Type	Beschreibung
Extracted (location)	`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.

Python code

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.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Source vector layer to select the features from
Method	`METHOD`	[enumeration] Default: 0	Random selection methods. One of: • 0 — Number of selected features • 1 — Percentage of selected features
Number/percentage of selected features	`NUMBER`	[number] Default: 10	Number or percentage of features to select
Extracted (random)	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Specify the output vector layer for the randomly selected features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… Vector layer containing randomly selected features

Ausgaben

Label	Name	Type	Beschreibung
Extracted (random)	`OUTPUT`	[same as input]	Vector layer containing randomly selected features from the input layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Vector layer to select the features from
ID field	`FIELD`	[tablefield: any]	Category of the source vector layer to select the features from
Method	`METHOD`	[enumeration] Default: 0	Random selection method. One of: • 0 — Number of selected features • 1 — Percentage of selected features
Number/percentage of selected features	`NUMBER`	[number] Default: 10	Number or percentage of features to select
Extracted (random stratified)	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer for the randomly selected features. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Extracted (random stratified)	`OUTPUT`	[same as input]	Vector layer containing randomly selected features from the input layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Vector layer for the selection
Method	METHOD	[enumeration] Default: 0	Random selection method. One of: • 0 — Number of selected features • 1 — Percentage of selected features
Number/percentage of selected features	NUMBER	[number] Default: 10	Number or percentage of features to select

Ausgaben

Label	Name	Type	Beschreibung
Input layer	INPUT	[same as input]	The input layer with features selected

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Vector layer to select features in
ID field	`FIELD`	[tablefield: any]	Category of the input layer to select the features from
Method	`METHOD`	[enumeration] Default: 0	Random selection method. One of: • 0 — Number of selected features • 1 — Percentage of selected features
Number/percentage of selected features	`NUMBER`	[number] Default: 10	Number or percentage of features to select

Ausgaben

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[same as input]	The input layer with features selected

Python code

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

Creates a selection in a vector layer.

The criteria for selecting features is based on the values of an attribute from the input layer.

Siehe auch:

Extract by attribute

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Vector layer to select features in
Selection attribute	`FIELD`	[tablefield: any]	Filtering field of the layer

Fortsetzung auf der nächsten Seite

Tab. 23.108 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Operator	`OPERATOR`	[enumeration] Default: 0	Many different operators are available: • 0 — = • 1 — ≠ • 2 — > • 3 — >= • 4 — < • 5 — <= • 6 — begins with • 7 — contains • 8 — is null • 9 — is not null • 10 — does not contain
Value Optional	`VALUE`	[string]	Value to be evaluated
Modify current selection by	`METHOD`	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: • 0 — creating new selection • 1 — adding to current selection • 2 — selecting within current selection • 3 — removing from current selection

Ausgaben

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[same as input]	The input layer with features selected

Python code

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

Creates a selection in a vector layer.

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Expression	EXPRESSION	[expression]	Expression to filter the input layer
Modify current selection by	METHOD	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: • 0 — creating new selection • 1 — adding to current selection • 2 — selecting within current selection • 3 — removing from current selection

Ausgaben

Label	Name	Type	Beschreibung
Input layer	INPUT	[same as input]	The input layer with features selected

Python code

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

Creates a selection in a vector layer.

The criteria for selecting features is based on the spatial relationship between each feature and the features in an additional layer.

Available geometric predicates are:

Intersect Tests whether a geometry intersects another. Returns 1 (true) if the geometries spatially intersect (share any portion of space - overlap or touch) and 0 if they don't. In the picture above, this will select circles 1, 2 and 3.

Contain Returns 1 (true) if and only if no points of b lie in the exterior of a, and at least one point of the interior of b lies in the interior of a. In the picture, no circle is selected, but the rectangle would be if you would select it the other way around, as it contains a circle completely. This is the opposite of *are within*.

Disjoint Returns 1 (true) if the geometries do not share any portion of space (no overlap, not touching). Only circle 4 is selected.

Equal Returns 1 (true) if and only if geometries are exactly the same. No circles will be selected.

Touch Tests whether a geometry touches another. Returns 1 (true) if the geometries have at least one point in common, but their interiors do not intersect. Only circle 3 is selected.

Overlap Tests whether a geometry overlaps another. Returns 1 (true) if the geometries share space, are of the same dimension, but are not completely contained by each other. Only circle 2 is selected.

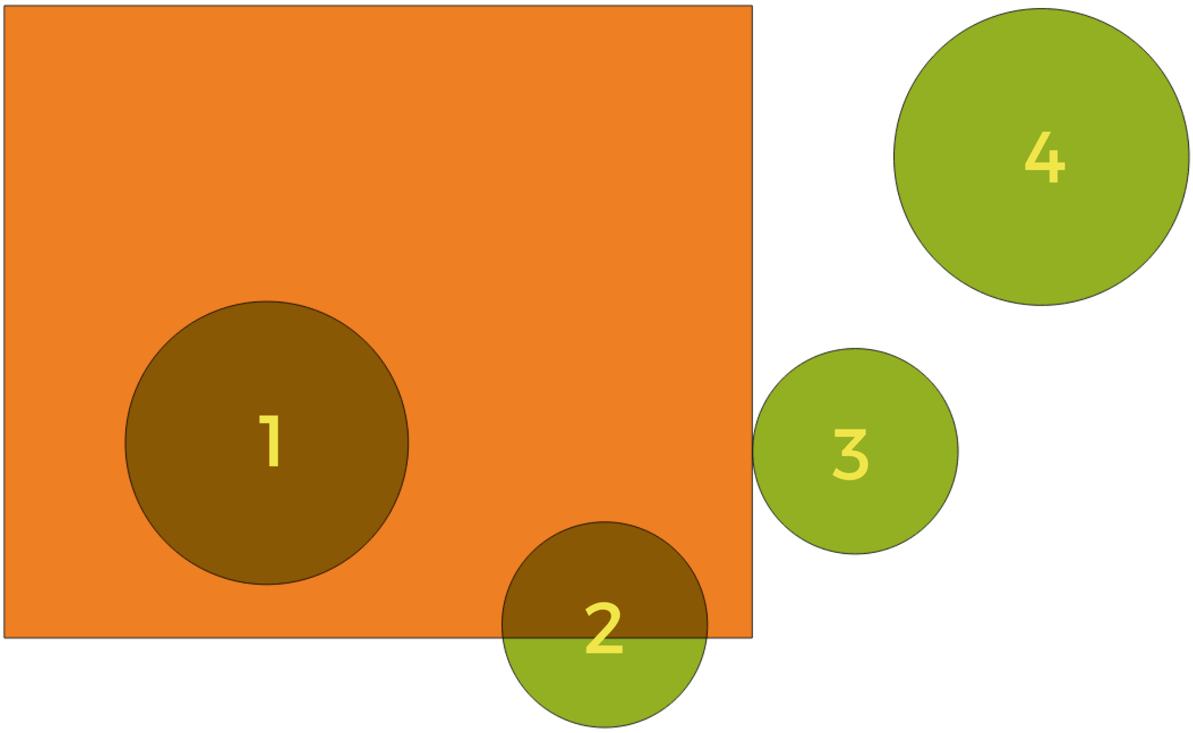


Abb. 23.87: In this example, the dataset from which we want to select (the *source vector layer*) consists of the green circles, the orange rectangle is the dataset that it is being compared to (the *intersection vector layer*).

Are within Tests whether a geometry is within another. Returns 1 (true) if geometry a is completely inside geometry b. Only circle 1 is selected.

Cross Returns 1 (true) if the supplied geometries have some, but not all, interior points in common and the actual crossing is of a lower dimension than the highest supplied geometry. For example, a line crossing a polygon will cross as a line (selected). Two lines crossing will cross as a point (selected). Two polygons cross as a polygon (not selected).

Default menu: *Vector ► Research Tools*

Siehe auch:

Extract by location

Parameter

Label	Name	Type	Beschreibung
Select features from	`INPUT`	[vector: any]	Input vector layer

Fortsetzung auf der nächsten Seite

Tab. 23.109 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Where the features (geometric predicate)	`PREDICATE`	[enumeration] [list] Default: [0]	Spatial condition for the selection. One or more of: • 0 — intersect • 1 — contain • 2 — disjoint • 3 — equal • 4 — touch • 5 — overlap • 6 — are within • 7 — cross If more than one condition is chosen, at least one of them (OR operation) has to be met for a feature to be extracted.
By comparing to the features from	`INTERSECT`	[vector: any]	Intersection vector layer
Modify current selection by	`METHOD`	[enumeration] Default: 0	How the selection of the algorithm should be managed. One of: • 0 — creating new selection • 1 — adding to current selection • 2 — selecting within current selection • 3 — removing from current selection

Ausgaben

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[same as input]	The input layer with features selected

Python code

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.

23.1.18 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input vector layer.
Field name	FIELD_NAME	[string] Default: ‚AUTO‘	Name of the field with autoincremental values
Start values at Optional	START	[number] Default: 0	Choose the initial number of the incremental count
Group values by 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.
Sort expression Optional	SORT_EXPRESSION	[expression]	Use an expression to sort the features in the layer either globally or if set, based on group fields.
Sort ascending	SORT_ASCENDING	[boolean] Default: True	When a `sort expression` is set, use this option to control the order in which features are assigned values.
Sort nulls first	SORT_NULLS_FIRST	[boolean] Default: False	When a `sort expression` is set, use this option to set whether *Null* values are counted first or last.
Incremented	OUTPUT	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer with the auto increment field. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Incremented	OUTPUT	[same as input]	Vector layer with auto incremental field

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input layer
Field name	FIELD_NAME	[string]	Name of the new field
Field type	FIELD_TYPE	[enumeration] Default: 0	Type of the new field. You can choose between: • 0 — Integer • 1 — Float • 2 — String
Field length	FIELD_LENGTH	[number] Default: 10	Length of the field
Field precision	FIELD_PRECISION	[number] Default: 0	Precision of the field. Useful with Float field type.
Added	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer (TEMPORARY_OUTPUT) • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Added	OUTPUT	[same as input]	Vector layer with new field added

Python code

Algorithm ID: qgis:addfieldtoattributestable

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input layer.
Class field	`FIELD`	[tablefield: any]	Features that have the same value for this field will get the same index.
Output field name	`FIELD_NAME`	[string] Default: ‚NUM_FIELD'	Name of the new field containing the indexes.
Layer with index field	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Vector layer with the numeric field containing indexes. One of: • Skip Output • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.
Class summary	`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: • Skip Output • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Layer with index field	`OUTPUT`	[same as input]	Vector layer with the numeric field containing indexes.
Class summary	`SUMMARY_OUTPUT`	[table] Default: `[Skip Output]`	Table with summary of the class field mapped to the corresponding unique value.

Python code

Algorithm ID: `qgis: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).

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: point]	The input layer.
Coordinate system	`CRS`	[crs] Default: „EPSG:4326“	Coordinate reference system to use for the generated x and y fields.
Field prefix Optional	`PREFIX`	[string]	Prefix to add to the new field names to avoid name collisions with fields in the input layer.
Added fields	`OUTPUT`	[vector: point] Default: `[Create temporary layer]`	Specify the output layer. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Added fields	`OUTPUT`	[vector: point]	The output layer - identical to the input layer but with two new double fields, `x` and `y`.

Python code

Algorithm ID: `qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Result field name	`FIELD_NAME`	[string] Default: ‚NewField‘	Name of the new field
Field type	`FIELD_TYPE`	[enumeration] Default: 0	Type of the new field. One of: • 0 — Integer • 1 — Float • 2 — String
Field length	`FIELD_LENGTH`	[number] Default: 10	Length of the field
Field precision	`FIELD_PRECISION`	[number] Default: 3	Precision of the field. Useful with Float field type.
Global expression 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.
Formula	`FORMULA`	[string]	The Python formula to evaluate. Example: To calculate the area of an input polygon layer you can add: `value = $geom.area()`
Calculated	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the vector layer with the new calculated field. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Calculated	`OUTPUT`	[same as input]	Vector layer with the new calculated field

Python code

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.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer to drop field(s) from
Fields to drop	`COLUMN`	[tablefield: any] [list]	The field(s) to drop
Remaining fields	`OUTPUT`	[same as input] Default: `[Create temporary layer]`	Specify the output vector layer with the remaining fields. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Remaining fields	`OUTPUT`	[same as input]	Vector layer with the remaining fields

Python code

Algorithm ID: `qgis: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 OGR (when reading an OSM file with the other_tags field.

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
HStore field	FIELD	[tablefield: any]	The field(s) to drop
Expected list of fields separated by a comma Optional	EXPECTED_FIELDS	[string] Default: ,‘	Comma-separated list of fields to extract. The HStore field will be updated by removing these keys.
Exploded	OUTPUT	[same as input] Default: [Create temporary layer]	Specify the output vector layer. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Exploded	OUTPUT	[same as input]	Output vector layer

Python code

Algorithm ID: qgis: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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer containing the binary data
Binary field	`FIELD`	[tablefield: any]	Field containing the binary data
File name	`FILENAME`	[expression]	Field or expression-based text to name each output file
Destination folder	`FOLDER`	[folder] Default: `[Save to a temporary folder]`	Folder in which to store the output files. One of: • Save to a Temporary Directory • Save to Directory… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Folder	`FOLDER`	[folder]	The folder that contains the output files.

Python code

Algorithm ID: `qgis: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.

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`.

Bemerkung: This algorithm is only available from the *Graphical modeler*.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input layer.
Outputs and filters (one or more)	`OUTPUT_<name of the filter>`	[same as input]	The output layers with filters (as many as there are filters).

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis (one or more)	`native:filter_1:<name of filter>`	[same as input]	The output layers with filtered features (as many as there are filters).

Python code

Algorithm ID: `qgis:featurefilter`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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 *Die Grafische Modellierung*.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The layer to calculate on
Output field name	`FIELD_NAME`	[string]	The name of the field for the results
Output field type	`FIELD_TYPE`	[enumeration] Default: 0	The type of the field. One of: • 0 — Float • 1 — Integer • 2 — String • 3 — Date
Output field width	`FIELD_LENGTH`	[number] Default: 10	The length of the result field (minimum 0)
Field precision	`FIELD_PRECISION`	[number] Default: 3	The precision of the result field (minimum 0, maximum 15)
Create new field	`NEW_FIELD`	[boolean] Default: True	Should the result field be a new field
Formula	`FORMULA`	[expression]	The formula to use to calculate the result
Output file	`OUTPUT`	[vector: any] Default: `[Save to temporary file]`	Specification of the output layer.

Ausgaben

Label	Name	Type	Beschreibung
Calculated	`OUTPUT`	[vector: any]	Output layer with the calculated field values

Python code

Algorithm ID: `qgis: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.

Refactor layer fields 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

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The layer to modify

Fortsetzung auf der nächsten Seite

Tab. 23.117 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Fields mapping	`FIELDS_MAPPING`	[list]	List of output fields with their definitions. The embedded table lists all the fields of the source layer and allows you to edit them: • Click to create a new field. • Click to remove a field. • Use and to change the selected field order. • Click to reset to the default view. For each of the fields you'd like to reuse, you need to fill the following options: ***Source expression*** **(`expression`) [expression]** Field or expression from the input layer. ***Field name*** **(`name`) [string]** Name of the field in the output layer. By default input field name is kept. ***Type*** **(`type`) [enumeration]** Data type of the output field. One of: • Date (14) • DateTime (16) • Double (6) • Integer (2) • Integer64 (4) • String (10) • Boolean (1) ***Length*** **(`length`) [number]** Length of the output field. ***Precision*** **(`precision`) [number]** Precision of the output field. Fields from another layer can be loaded into the field list in *Load fields from layer*.
Refactored	`OUTPUT`	[vector: any] Default: `[Create temporary layer]`	Specification of the output layer. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Refactored	`OUTPUT`	[vector: any]	Output layer with refactored fields

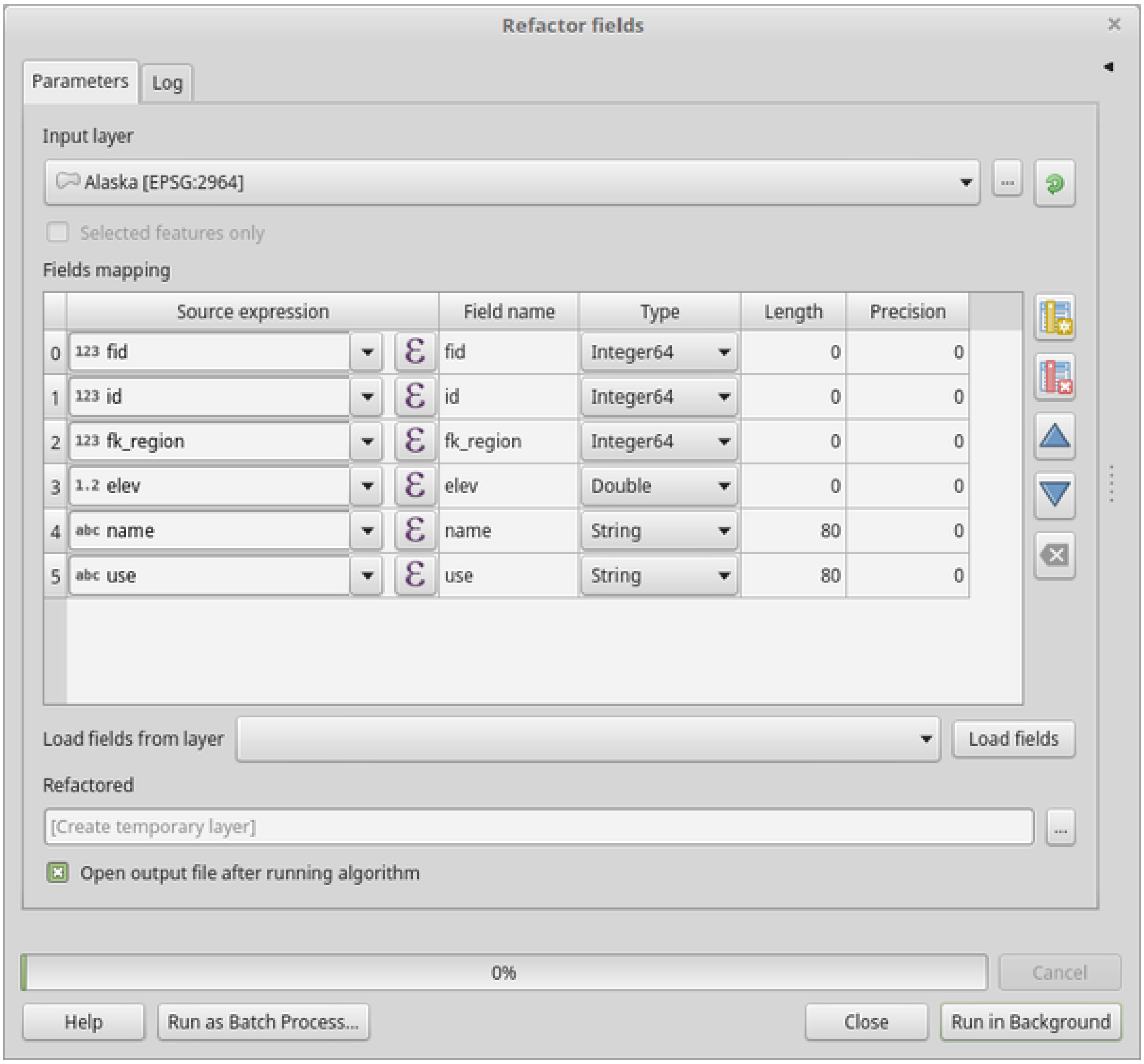


Abb. 23.88: Refactor fields dialog

Python code

Algorithm ID: `qgis: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.

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer.
Text attribute to convert to float	`FIELD`	[tablefield: string]	The string field for the input layer that is to be converted to a float field.
Float from text	`OUTPUT`	[same as input] Default: `[Create Temporary Layer]`	Specify the output layer. One of: • Create Temporary Layer • Save to File… • Save to Geopackage… • Save to PostGIS Table…… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Float from text	`OUTPUT`	[same as input]	Output vector layer with the string field converted into a float field

Python code

Algorithm ID: `qgis:texttofloat`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.2 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.

23.2.1 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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input elevation raster layer
Band number	`BAND`	[raster band] Default: 1	The number of the band to use as elevation
Return trigonometric angle instead of azimuth	`TRIG_ANGLE`	[boolean] Default: False	Activating the trigonometric angle results in different categories: 0° (East), 90° (North), 180° (West), 270° (South).
Return 0 for flat instead of -9999	`ZERO_FLAT`	[boolean] Default: False	Activating this option will insert a 0-value for the value -9999 on flat areas.
Compute edges	`COMPUTE_EDGES`	[boolean] Default: False	Generates edges from the elevation raster
Use Zevenbergen&Thorne formula instead of the Horn's one	`ZEVENBERGEN`	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Aspect	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Aspect	OUTPUT	[raster]	Output raster with angle values in degrees

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input elevation raster layer
Band number	BAND	[raster band] Default: 1	The number of the band to use as elevation
Compute edges	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
Color configuration file	COLOR_TABLE	[file]	A text-based color configuration file
Matching mode	MATCH_MODE	[enumeration] Default: 2	One of: • 0 — Use strict color matching • 1 — Use closest RGBA quadruples • 2 — Use smoothly blended colours
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Color relief	OUTPUT	[raster] Default: [Save to temporary file]	Output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Color relief	OUTPUT	[raster]	A 4-band output raster

Python code

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 no data values by interpolation from edges. The values for the no-data regions are calculated by the sourrounding pixel values using inverse distance weighting. After the interpolation a smoothing of the results takes placce. 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Band number	BAND	[raster band] Default: 1	The band to operate on. Nodata values must be represented by the value 0.
Maximum distance (in pixels) to search out for values to interpolate	DISTANCE	[number] Default: 10	The number of pixels to search in all directions to find values to interpolate from
Number of smoothing iterations to run after the interpolation	ITERATIONS	[number] Default: 0	The number of 3x3 filter passes to run (0 or more) to smoothen the results of the interpolation.
Do not use default validity mask for the input band	NO_MASK	[boolean] Default: False	Activates the user-defined validity mask
Validity mask	MASK_LAYER	[raster]	A raster layer that defines the areas to fill.
Additional creation options 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 *GDAL driver options section*).

Fortsetzung auf der nächsten Seite

Tab. 23.120 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Filled	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Filled	`OUTPUT`	[raster]	Output raster

Python code

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

Label	Name	Type	Beschreibung
Point layer	`INPUT`	[vector: point]	Input point vector layer

Fortsetzung auf der nächsten Seite

Tab. 23.121 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Data metric to use	METRIC	[enumeration] Default: 0	One of: • 0 — Minimum, minimum value found in grid node search ellipse • 1 — Maximum, maximum value found in grid node search ellipse • 2 — Range, a difference between the minimum and maximum values found in grid node search ellipse • 3 — Count, a number of data points found in grid node search ellipse • 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 • 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
The first radius of search ellipse	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
The second radius of search ellipse	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
Angle of search ellipse rotation in degrees (counter clockwise)	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
Minimum number of data points to use	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.
Nodata	NODATA	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 23.121 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (data metrics)	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (data metrics)	`OUTPUT`	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Point layer	INPUT	[vector: point]	Input point vector layer
Weighting power	POWER	[number] Default: 2.0	Weighting power
Smoothing	SMOOTHING	[number] Default: 0.0	Smoothing parameter
The radius of the search circle	RADIUS	[number] Default: 1.0	The radius of the search circle
Maximum number of data points to use	MAX_POINTS	[number] Default: 12	Do not search for more points than this number.
Minimum number of data points to use	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.
Nodata	NODATA	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Output data type	DATA_TYPE	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (IDW with NN search)	OUTPUT	[raster] Default: `[Save to temporary file]`	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (IDW with NN search)	OUTPUT	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Point layer	INPUT	[vector: point]	Input point vector layer
Weighting power	POWER	[number] Default: 2.0	Weighting power
Smothing	SMOOTHING	[number] Default: 0.0	Smoothing parameter
The first radius of search ellipse	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
The second radius of search ellipse	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
Angle of search ellipse rotation in degrees (counter clockwise)	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
Maximum number of data points to use	MAX_POINTS	[number] Default: 0	Do not search for more points than this number.

Fortsetzung auf der nächsten Seite

Tab. 23.123 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Minimum number of data points to use	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.
Nodata	NODATA	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Output data type	DATA_TYPE	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (IDW)	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (IDW)	OUTPUT	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Point layer	`INPUT`	[vector: point]	Input point vector layer
Search distance	`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`, no data value will be used.
Nodata	`NODATA`	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	`Z_FIELD`	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 23.124 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	DATA_TYPE	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (Linear)	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (Linear)	OUTPUT	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Point layer	`INPUT`	[vector: point]	Input point vector layer
The first radius of search ellipse	`RADIUS_1`	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
The second radius of search ellipse	`RADIUS_2`	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
Angle of search ellipse rotation in degrees (counter clockwise)	`ANGLE`	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
Minimum number of data points to use	`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.
Nodata	`NODATA`	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	`Z_FIELD`	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (moving average)	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (moving average)	OUTPUT	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Point layer	INPUT	[vector: point]	Input point vector layer
The first radius of search ellipse	RADIUS_1	[number] Default: 0.0	The first radius (X axis if rotation angle is 0) of the search ellipse
The second radius of search ellipse	RADIUS_2	[number] Default: 0.0	The second radius (Y axis if rotation angle is 0) of the search ellipse
Angle of search ellipse rotation in degrees (counter clockwise)	ANGLE	[number] Default: 0.0	Angle of ellipse rotation in degrees. Ellipse rotated counter clockwise.
Nodata	NODATA	[number] Default: 0.0	No data marker to fill empty points
Z value from field Optional	Z_FIELD	[tablefield: numeric]	Field for the interpolation
Additional creation options 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 *GDAL driver options section*).

Fortsetzung auf der nächsten Seite

Tab. 23.126 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Interpolated (Nearest neighbour)	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Interpolated (Nearest neighbour)	`OUTPUT`	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input Elevation raster layer
Band number	`BAND`	[raster band] Default: 1	Band containing the elevation information
Z factor (vertical exaggeration)	`Z_FACTOR`	[number] Default: 1.0	The factor exaggerates the height of the output elevation raster
Scale (ratio of vert. units to horiz.)	`SCALE`	[number] Default: 1.0	The ratio of vertical units to horizontal units
Azimuth of the light	`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.
Altitude of the light	`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.
Compute edges	`COMPUTE_EDGES`	[boolean] Default: False	Generates edges from the elevation raster
Use Zevenbergen&Thorne formula (instead of the Horn's one)	`ZEVENBERGEN`	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
Combined shading	`COMBINED`	[boolean] Default: False	
Multidirectional shading	`MULTIDIRECTIONAL`	[boolean] Default: False	
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Hillshade	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer with interpolated values. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Hillshade	`OUTPUT`	[raster]	Output raster with interpolated values

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input Elevation raster layer
How far from black (white)	`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.
Search for nearly white pixels instead of nearly black	`WHITE`	[boolean] Default: False	Search for nearly white (255) pixels instead of nearly black pixels
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Nearblack	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Nearblack	`OUTPUT`	[raster]	Output raster

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input Elevation raster layer
Band number	`BAND`	[raster band] Default: 1	Band containing the elevation information
A list of pixel values in the source image to be considered target pixels 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.
Distance units	`UNITS`	[enumeration] Default: 1	Indicate whether distances generated should be in pixel or georeferenced coordinates. One of: • 0 — Georeferenced coordinates • 1 — Pixel coordinates
The maximum distance to be generated 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 *Distance units*.

Fortsetzung auf der nächsten Seite

Tab. 23.129 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Value to be applied to all pixels that are within the maxdist of target pixels 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.
Nodata value to use for the destination proximity raster Optional	`NODATA`	[number] Default: 0.0	Specify the nodata value to use for the output raster
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the data type of the output raster file. Options: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Proximity map	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Proximity map	`OUTPUT`	[raster]	Output raster

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input elevation raster layer
Band number	`BAND`	[raster band] Default: 1	The number of the band to use as elevation
Compute edges	`COMPUTE_EDGES`	[boolean] Default: False	Generates edges from the elevation raster
Additional creation options 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 *GDAL driver options section*).
Roughness	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Roughness	`OUTPUT`	[raster]	Single-band output roughness raster. The value -9999 is used as nodata value.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input elevation raster layer
Threshold	THRESHOLD	[number] Default: 10	Only raster polygons smaller than this size will be removed
Use 8-connectedness	EIGHT_CONNECTEDNESS	[boolean] Default: False	Use eight connectedness instead of four connectedness
Do not use the default validity mask for the input band	NO_MASK	[boolean] Default: False	
Validity mask Optional	MASK_LAYER	[raster]	Validity mask to use instead of the default
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Sieved	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Sieved	OUTPUT	[raster]	Output raster layer.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input Elevation raster layer
Band number	`BAND`	[raster band] Default: 1	Band containing the elevation information
Ratio of vertical units to horizontal	`SCALE`	[number] Default: 1.0	The ratio of vertical units to horizontal units
Slope expressed as percent (instead of degrees)	`AS_PERCENT`	[boolean] Default: False	Express slope as percent instead of degrees
Compute edges	`COMPUTE_EDGES`	[boolean] Default: False	Generates edges from the elevation raster
Use Zevenbergen&Thorne formula (instead of the Horn's one)	`ZEVENBERGEN`	[boolean] Default: False	Activates Zevenbergen&Thorne formula for smooth landscapes
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Slope	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Slope	OUTPUT	[raster]	Output raster

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input elevation raster layer
Band number	BAND	[raster band] Default: 1	The number of the band to use as elevation
Compute edges	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
Additional creation options 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 *GDAL driver options section*).
Terrain Ruggedness Index	OUTPUT	[raster] Default: [Save to temporary file]	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Terrain Ruggedness Index	OUTPUT	[raster]	Output ruggedness raster. The value -9999 is used as nodata value.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input elevation raster layer
Band number	BAND	[raster band] Default: 1	The number of the band to use for elevation values
Compute edges	COMPUTE_EDGES	[boolean] Default: False	Generates edges from the elevation raster
Additional creation options 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 *GDAL driver options section*).
Terrain Ruggedness Index	OUTPUT	[raster] Default: `[Save to temporary file]`	Specify the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Terrain Ruggedness Index	OUTPUT	[raster]	Output raster.

Python code

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.

23.2.2 Raster conversion

gdal2xyz

Converts raster data to XYZ ASCII file format.

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Raster layer to convert
Band number	BAND	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to convert
Output comma-separated values	CSV	[boolean] Default: False	Sets whether the output file should be of type comma-separated values (csv).
XYZ ASCII file	OUTPUT	[file] Default: `[Save to temporary file]`	Specification of the output file. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
XYZ ASCII file	INPUT	[table]	Table file containing the values exported from the raster band.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input 8 bit raster image
Band number	`BAND`	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to convert
Generate a RGBA file	`RGBA`	[boolean] Default: False	Sets whether the output file should be of type RGBA.
PCT to RGB	`OUTPUT`	[file] Default: `[Save to temporary file]`	Specification of the output file. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
PCT to RGB	`OUTPUT`	[raster]	24 bit RGB raster image

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input raster layer
Band number	`BAND`	[raster band] Default: The first band of the input layer	If the raster is multiband, choose the band you want to use
Name of the field to create	`FIELD`	[string] Default: ‚DN'	Specify the field name for the attributes of the connected regions.
Use 8-connectedness	`EIGHT_CONNECTEDNESS`	[boolean] Default: False	If not set, raster cells must have a common border to be considered connected (*4-connected*). If set, touching raster cells are also considered connected (*8-connected*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Vectorized	`OUTPUT`	[vector: polygon] Default: `[Save to temporary file]`	Specification of the output (polygon) vector layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Vectorized	`OUTPUT`	[vector: polygon]	Output vector layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Selected band(s)	BANDS	[raster band] [list] Default: None	Ordered list of the bands to use to create the new raster
Additional creation options 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 *GDAL driver options section*).
Output data type	DATA_TYPE	[enumeration] Default: 0	Defines the data type of the output raster file. Options: • 0 — Use Input Layer Data Type • 1 — Byte • 2 — Int16 • 3 — UInt16 • 4 — UInt32 • 5 — Int32 • 6 — Float32 • 7 — Float64 • 8 — CInt16 • 9 — CInt32 • 10 — CFloat32 • 11 — CFloat64
Converted	OUTPUT	[raster] Default: Save to temporary file	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Converted	OUTPUT	[raster]	Output raster layer with rearranged bands.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input (RGB) raster layer
Number of colors	`NCOLORS`	[number] Default: 2	The number of colors the resulting image will contain. A value from 2-256 is possible.
RGB to PCT	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
RGB to PCT	`OUTPUT`	[raster]	Output raster layer.

Python code

Algorithm ID: `gdal:rgbtopct`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Override the projection of the output file Optional	TARGET_CRS	[crs]	Specify a projection for the output file
Assign a specified nodata value to output bands Optional	NODATA	[number] Default: Not set	Defines the value to use for nodata in the output raster
Copy all subdatasets of this file to individual output files	COPY_SUBDATASETS	[boolean] Default: False	Create individual files for subdatasets
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Output data type	DATA_TYPE	[enumeration] Default: 0	Defines the data type of the output raster file. Options: • 0 — Use Input Layer Data Type • 1 — Byte • 2 — Int16 • 3 — UInt16 • 4 — UInt32 • 5 — Int32 • 6 — Float32 • 7 — Float64 • 8 — CInt16 • 9 — CInt32 • 10 — CFloat32 • 11 — CFloat64
Converted	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output (translated) raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Converted	`OUTPUT`	[raster]	Output (translated) raster layer.

Python code

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.

23.2.3 Raster extraction

Clip raster by extent

Clips any GDAL-supported raster file to a given extent.

This algorithm is derived from the GDAL grid utility.

Default menu: *Raster ▸ Extraction*

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	The input raster
Clipping extent	`EXTENT`	[extent]	Extent that should be used for the output raster. Only pixels within the specified bounding box will be included in the output.
Assign a specified nodata value to output bands Optional	`NODATA`	[number] Default: None	Defines a value that should be inserted for the nodata values in the output raster
Additional creation options 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 *GDAL driver options section*).

Fortsetzung auf der nächsten Seite

Tab. 23.141 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	DATA_TYPE	[enumeration] Default: 0	Defines the format of the output raster file. Optionen: • 0 — Use Input Layer Data Type • 1 — Byte • 2 — Int16 • 3 — UInt16 • 4 — UInt32 • 5 — Int32 • 6 — Float32 • 7 — Float64 • 8 — CInt16 • 9 — CInt32 • 10 — CFloat32 • 11 — CFloat64
Additional command-line parameters Optional	EXTRA	[string] Default: None	Add extra GDAL command line options
Clipped (extent)	OUTPUT	[raster] Default: ‚[Save to temporary file]'	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here

Ausgaben

Label	Name	Type	Beschreibung
Clipped (extent)	OUTPUT	[raster]	Output raster layer clipped by the given extent

Python code

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 grid utility.

Default menu: *Raster ▸ Extraction*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	The input raster
Mask layer	EXTENT	[vector: polygon]	Vector mask for clipping the raster
Source CRS	SOURCE_CRS	[crs]	
Target CRS	TARGET_CRS	[crs]	
Assign a specified nodata value to output bands Optional	NODATA	[number] Default: None	Defines a value that should be inserted for the nodata values in the output raster
Create an output alpha band	ALPHA_BAND	[boolean] Default: False	Creates an alpha band for the result. The alpha band then includes the transparency values of the pixels.
Match the extent of the clipped raster to the extent of the mask layer	CROP_TO_CUTLINE	[boolean] Default: True	Applies the vector layer extent to the output raster if checked.
Keep resolution of input raster	KEEP_RESOLUTION	[boolean] Default: False	The resolution of the output raster will not be changed
Set output file resolution	SET_RESOLUTION	[boolean] Default: False	Shall the output resolution (cell size) be specified
X Resolution to output bands Optional	X_RESOLUTION	[number] Default: None	The width of the cells in the output raster
Y Resolution to output band Optional	Y_RESOLUTION	[number] Default: None	The height of the cells in the output raster
Use multithreaded warping implementation	MULTITHREADING	[boolean] Default: False	
Additional creation options 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 *GDAL driver options section*).

Fortsetzung auf der nächsten Seite

Tab. 23.142 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	`DATA_TYPE`	[enumeration] Default: 0	Defines the format of the output raster file. Optionen: • 0 — Use Input Layer Data Type • 1 — Byte • 2 — Int16 • 3 — UInt16 • 4 — UInt32 • 5 — Int32 • 6 — Float32 • 7 — Float64 • 8 — CInt16 • 9 — CInt32 • 10 — CFloat32 • 11 — CFloat64
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Clipped (mask)	`OUTPUT`	[raster] Default: ‚[Save to temporary file]'	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here

Ausgaben

Label	Name	Type	Beschreibung
Clipped (mask)	`OUTPUT`	[raster]	Output raster layer clipped by the vector layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input raster
Band number	`BAND`	[raster band]	Raster band to create the contours from
Interval between contour lines	`INTERVAL`	[number] Default: 10.0	Defines the interval between the contour lines in the given units of the elevation raster (minimum value 0)
Attribute name (if not set, no elevation attribute is attached) Optional	`FIELD_NAME`	[string] Default: ‚ELEV‘	Defines the attribute name for the field containing the values of the contour lines.
Produce 3D vector	`CREATE_3D`	[boolean] Default: False	Forces production of 3D vectors instead of 2D. Includes elevation at every vertex.
Treat all raster values as valid	`IGNORE_NODATA`	[boolean] Default: False	Ignores any nodata values in the dataset.
Input pixel value to treat as „nodata“ Optional	`NODATA`	[number] Default: None	Defines a value that should be inserted for the nodata values in the output raster
Offset from zero relative to which to interpret intervals Optional	`OFFSET`	[number] Default: 0.0	
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Additional creation options 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 *GDAL driver options section*).
Contours	`OUTPUT`	[vector: line] Default: ‚[Save to temporary file]‘	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Contours	`OUTPUT`	[vector: line]	Output vector layer with contour lines

Python code

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.

23.2.4 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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	Input raster layer
Overview levels	`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.
Remove all existing overviews	`CLEAN`	[boolean] Default: False	Removes existing overviews from the raster. By default these are not removed.
Resampling method Optional	`RESAMPLING`	[enumeration] Default: 0	Calculates the overviews with a defined resampling method. Possible resampling methods are: • 0 – Nearest Neighbour (`nearest`) • 1 – Average (`average`) • 2 – Gaussian (`gauss`) • 3 – Cubic Convolution (`cubic`) • 4 – B-Spline Convolution (`cubicspline`) • 5 – Lanczos Windowed Sinc (`lanczos`) • 6 – Average MP (`average_mp`) • 7 – Average in Mag/Phase Space (`average_magphase`) • 8 – Mode (`mode`)

Fortsetzung auf der nächsten Seite

Tab. 23.144 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Overviews format 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: • 0 – Internal (if possible) • 1 – External (GTiff .ovr) • 2 – External (ERDAS Imagine .aux)
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Pyramidized	`OUTPUT`	[raster]	Output raster layer

Ausgaben

Label	Name	Type	Beschreibung
Pyramidized	`OUTPUT`	[raster]	Output raster layer with overviews

Python code

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

Label	Name	Type	Beschreibung
Input layers	`INPUT`	[raster] [list]	GDAL-supported raster layers.
Resolution	`RESOLUTION`	[enumeration] Default: 0	The output resolution of the mosaic. By default the average resolution of the raster files will be chosen. Optionen: • 0 — Average (`average`) • 1 — Highest (`highest`) • 2 — Lowest (`lowest`)

Fortsetzung auf der nächsten Seite

Tab. 23.145 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Place each input file into a separate band	SEPARATE	[boolean] Default: True	With ‚True' you can define that each raster file goes into a separated stacked band in the VRT band.
Allow projection difference	PROJ_DIFFERENCE	[boolean] Default: False	Allows that the output bands have different projections derived from the projection of the input raster layers.
Add alpha mask band to VRT when source raster has none	ADD_ALPHA	[boolean] Default: False	Adds an alpha mask band to the VRT when the source raster has none.
Override projection for the output file (optional)	ASSIGN_CRS	[crs] Default: None	Overrides the projection for the output file. No reprojection is done.
Resampling algorithm	RESAMPLING	[enumeration] Default: 0	The resampling algorithm to be used Optionen: • 0 — Nearest Neighbour (nearest) • 1 — Bilinear (bilinear) • 2 — Cubic Convolution (cubic) • 3 — B-Spline Convolution (cubicspline) • 4 — Lanczos Windowed Sinc (lanczos) • 5 — Average (average) • 6 — Mode (mode)
Nodata value(s) for input bands (space separated) Optional	SRC_NODATA	[string] Default: None	Space separated Nodata value(s) for input band(s)
Additional command-line parameters	EXTRA	[string] Default: None	Add extra GDAL command line options
Virtual	OUTPUT	[raster] Default: [Save to temporary file]	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Virtual	OUTPUT	[raster]	Output raster layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[raster]	GDAL-supported raster layer.
Tile cutting profile	`PROFILE`	[enumeration] Default: 0	One of: • 0 — Mercator (`mercator`) • 1 — Geodetic (`geodetic`) • 2 — Raster (`raster`)
Zoom levels to render Optional	`ZOOM`	[string] Default: ‚‘	
Web viewer to generate	`VIEWER`	[enumerate] Default: 0	One of: • 0 — All (`all`) • 1 — GoogleMaps (`google`) • 2 — OpenLayers (`openlayers`) • 3 — Leaflet (`leaflet`) • 4 — None (`none`)
Title of the map Optional	`TITLE`	[string] Default: ‚‘	
Copyright of the map	`COPYRIGHT`	[string] Default: ‚‘	

Fortsetzung auf der nächsten Seite

Tab. 23.146 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Resampling method	`RESAMPLING`	[enumeration] Default: 0	The resampling algorithm to be used Optionen: • 0 — Average (`average`) • 1 — Nearest neighbour (`near`) • 2 — Bilinear (`bilinear`) • 3 — Cubic (`cubic`) • 4 — Cubic spline (`cubicspline`) • 5 — Lanczos Windowed sinc (`lanczos`) • 6 — Antialias (`antialias`)
The spatial reference system used for the source input data Optional	`SOURCE_CRS`	[crs] Default: None	
Transparency value to assign to the input data Optional	`NODATA`	[number] Default: 0.0	
URL address where the generated tiles are going to be published Optional	`URL`	[string] Default: ‚‘	
Google Maps API key (http://code.google.com/apis/maps/signup.html) Optional	`GOOGLE_KEY`	[string] Default: ‚‘	Your Google maps API key.
Bing Maps API key (https://www.bingmapsportal.com/) Optional	`BING_KEY`	[string] Default: ‚‘	Your Bing maps API key.
Generate only missing files	`RESUME`	[boolean] Default: False	
Generate KML for Google Earth	`KML`	[boolean] Default: False	
Avoid automatic generation of KML files for EPSG:4326	`NO_KML`	[boolean] Default: False	
Output directory	`OUTPUT`	[folder] Default: `[Save to temporary file]`	Specify the output folder for the tiles.

Ausgaben

Label	Name	Type	Beschreibung
Output directory	OUTPUT	[folder]	The output folder (for the tiles)

Python code

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

Label	Name	Type	Beschreibung
Input layers	INPUT	[raster] [list]	Input raster layers
Grab pseudocolor table from first layer	PCT	[boolean] Default: False	The pseudocolor table from the first layer will be used for the coloring
Place each input file into a separate band	SEPARATE	[boolean] Default: False	Place each input file into a separate band
Output data type	DATA_TYPE	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Input pixel value to treat as „nodata“ Optional	NODATA_INPUT	[number] Default: None	Ignores pixels from files being merged in with this pixel value

Fortsetzung auf der nächsten Seite

Tab. 23.147 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Assign specified „nodata“ value to output Optional	`NODATA_OUTPUT`	[number] Default: None	Assigns the specified nodata value to output bands.
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Merged	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Merged	`OUTPUT`	[raster]	Output raster layer

Python code

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

Label	Name	Type	Beschreibung
Spectral dataset	`SPECTRAL`	[raster]	Input (spectral) raster layer
Panchromatic dataset	`PANCHROMATIC`	[raster]	Input (panchromatic) raster layer

Fortsetzung auf der nächsten Seite

Tab. 23.148 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Resampling algorithm	`RESAMPLING`	[enumeration] Default: 2	The resampling algorithm to be used Optionen: • 0 — Nearest Neighbour (`nearest`) • 1 — Bilinear (`bilinear`) • 2 — Cubic (`cubic`) • 3 — Cubic Spline (`cubicspline`) • 4 — Lanczos Windowed Sinc (`lanczos`) • 5 — Average (`average`)
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options
Ergebnis	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output (sharpened) raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Ergebnis	`OUTPUT`	[raster]	Output (sharpened) raster layer

Python code

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

Parameter

Label	Name	Type	Beschreibung
Input layer A	INPUT_A	[raster]	First input raster layer (mandatory)
Number of raster band for A	BAND_A	[raster band]	Band for input layer A (mandatory)
Input layer B Optional	INPUT_B	[raster] Default: None	Second input raster layer
Number of raster band for B Optional	BAND_B	[raster band]	Band for input layer B
Input layer C Optional	INPUT_C	[raster] Default: None	Third input raster layer
Number of raster band for C Optional	BAND_C	[raster band]	Band for input layer C
Input layer D Optional	INPUT_D	[raster] Default: None	Fourth input raster layer
Number of raster band for D Optional	BAND_D	[raster band]	Band for input layer D
Input layer E Optional	INPUT_E	[raster] Default: None	Fifth input raster layer
Number of raster band for E Optional	BAND_E	[raster band]	Band for input layer E
Input layer F Optional	INPUT_F	[raster]	Sixth input raster layer
Number of raster band for F Optional	BAND_F	[raster band] Default: None	Band for input layer F
Calculation in gdalnumeric syntax using +-/* or any numpy array functions (i.e. logical_and())	FORMULA	[string] Default: ,‘	The calculation formula. Examples: • `A*(A>0)` — outputs the value of the raster A if the value of A is greater than 0. If not, outputs 0. • `A*(A>0 and A>B)` — outputs the value of A if that value is bigger than 0 and bigger than the value of B. If not, outputs 0. • `A*logical_or(A<=177, A>=185)` — outputs the value of A if A <= 177 or A >= 185. If not, outputs 0. • `sqrt(A*A+B*B)` — Outputs the square root of the sum of the value of A squared and the value of B squared.
Set output nodata value Optional	NO_DATA	[number] Default: None	Value to use for nodata

Fortsetzung auf der nächsten Seite

Tab. 23.149 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output raster type	`RTYPE`	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: ‚‘	Add extra GDAL command line options
Calculated	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Specify the output (calculated) raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Calculated	`OUTPUT`	[raster]	Output (calculated) raster layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer
Force computation of the actual min/max values for each band	MIN_MAX	[boolean] Default: False	Forces computation of the actual min/max values for each band in the dataset
Read and display image statistics (force computation if necessary)	STATS	[boolean] Default: False	Reads and displays image statistics. Forces computation if no statistics are stored in an image.
Suppress GCP info	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.
Suppress metadata info	NO_METADATA	[boolean] Default: False	Suppresses metadata printing. Some datasets may contain a lot of metadata strings.
Additional command-line parameters	EXTRA	[string] Default: None	Add extra GDAL command line options
Layer information	OUTPUT	[html] Default: `[Save to temporary file]`	Specify the HTML file for output. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Layer information	OUTPUT	[html]	The HTML file containing information about the input raster layer

Python code

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

Label	Name	Type	Beschreibung
Input files	`INPUT`	[raster] [list]	The input raster files
Tile width	`TILE_SIZE_X`	[number] Default: 256	Width of the tiles in pixels (minimum 0)
Tile height	`TILE_SIZE_Y`	[number] Default: 256	Height of the tiles in pixels (minimum 0)
Overlap in pixels between consecutive tiles	`OVERLAP`	[number] Default: 0	
Number of pyramid levels to build	`LEVELS`	[number] Default: 1	Minimum: 0
Source coordinate reference system	`SOURCE_CRS`	[crs] Default: None	
Resampling method	`RESAMPLING`	[enumeration] Default: 0	The resampling algorithm to be used Optionen: • 0 — Nearest Neighbour (`nearest`) • 1 — Bilinear (`bilinear`) • 2 — Cubic (`cubic`) • 3 — Cubic Spline (`cubicspline`) • 4 — Lanczos Windowed Sinc (`lanczos`)
Column delimiter used in the CSV file Optional	`DELIMITER`	[string] Default: ‚;‘	Delimiter to use in the CSV file containing the tile(s) georeferencing information
Additional creation options 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 *GDAL driver options section*).
Additional command-line parameters Optional	`EXTRA`	[string] Default: ‚‘	Add extra GDAL command line options

Fortsetzung auf der nächsten Seite

Tab. 23.151 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output data type	`DATA_TYPE`	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Build only the pyramids	`ONLY_PYRAMIDS`	[boolean] Default: False	
Use separate directory for each tile row	`DIR_FOR_ROW`	[boolean] Default: False	
Output directory	`OUTPUT`	[folder] Default: `[Save to temporary folder]`	Specify the output folder for the tiles. One of: • Save to Temporary Directory • Save to Directory… The file encoding can also be changed here.
CSV file containing the tile(s) georeferencing information	`OUTPUT_CSV`	[file] Default: `[Skip output]`	One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output directory	`OUTPUT`	[folder]	The output folder for the tiles.
CSV file containing the tile(s) georeferencing information	`OUTPUT_CSV`	[file]	The CSV file with georeferencing information for the tiles.

Python code

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

Label	Name	Type	Beschreibung
Input files	LAYERS	[raster] [list]	The input raster files. Can be multiple files.
Field name to hold the file path to the indexed rasters	PATH_FIELD_NAME Optional	[string] Default: ‚location‘	The output field name to hold the file path/location to the indexed rasters.
Store absolute path to the indexed rasters	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.
Skip files with different projection reference	PROJ_DIFFERENCE	[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.
Transform geometries to the given CRS 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.
The name of the field to store the SRS of each tile Optional	CRS_FIELD_NAME	[string]	The name of the field to store the SRS of each tile
The format in which the CRS of each tile must be written	CRS_FORMAT	[enumeration] Default: 0	Format for the CRS. One of: • 0 – Auto (AUTO) • 1 – Well-known text (WKT) • 2 – EPSG (EPSG) • 3 – Proj.4 (PROJ)
Tile index	OUTPUT	[vector: polygon] Default: [Save to temporary file]	Specify the polygon vector layer to write the index to. One of: • Save to a Temporary File • Save to File The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Tile index	OUTPUT	[vector: polygon]	The polygon vector layer with the tile index.

Python code

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.

23.2.5 Rasterprojektionen

Projektion zuweisen

Wendet ein Koordinatenbezugssystem auf ein Rasterdatensatz an.

This algorithm is derived from the GDAL edit utility.

Default menu: *Raster ▸ Projections*

Parameter

Label	Name	Type	Beschreibung
Input layer	INPUT_LAYER	[raster]	Input raster layer
Desired CRS	CRS	[crs]	The projection (CRS) of the output layer

Ausgaben

Label	Name	Type	Beschreibung
Layer with projection	OUTPUT	[raster]	The output raster layer (with the new projection information)

Python code

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

Label	Name	Type	Beschreibung
Input file	`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 `.wld` file. Using a non-file raster layer will lead to an error.
Create also .prj file	`PRJ_FILE_CREATE`	[boolean] Default: False	If this is activated a `.prj` file containing the projection information is also created.

Ausgaben

Label	Name	Type	Beschreibung
World file	`WORLD_FILE`	[file]	Text file with extension `.wld` containing transformation parameters for the raster file.
ESRI Shapefile prj file	`PRJ_FILE`	[file]	Text file with `.prj` extension that describes the CRS. Will be `None` if *Create also .prj file* is False.

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[raster]	Input raster layer to reproject
Source CRS Optional	SOURCE_CRS	[crs]	Defines the CRS of the input raster layer
Target CRS Optional	TARGET_CRS	[crs] Default: EPSG:4326	The CRS of the output layer
Resampling method to use	RESAMPLING	[enumeration] Default: 0	Pixel value resampling method to use. Options: • 0 — Nearest neighbour • 1 — Bilinear • 2 — Cubic • 3 — Cubic spline • 4 — Lanczos windowed sinc • 5 — Average • 6 — Mode • 7 — Maximum • 8 — Minimum • 9 — Median • 10 — First quartile • 11 — Third quartile
Nodata value for output bands Optional	NODATA	[number] Default: None	Sets nodata value for output bands. If not provided, then nodata values will be copied from the source dataset.
Output file resolution in target georeferenced units Optional	TARGET_RESOLUTION	[number] Default: None	Defines the output file resolution of reprojection result
Additional creation options 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 *GDAL driver options section*).
Output data type	DATA_TYPE	[enumeration] Default: 0	Defines the format of the output raster file. Optionen: • 0 — Use input layer data type • 1 — Byte • 2 — Int16 • 3 — UInt16 • 4 — UInt32 • 5 — Int32 • 6 — Float32 • 7 — Float64 • 8 — CInt16 • 9 — CInt32 • 10 — CFloat32 • 11 — CFloat64
Georeferenced extents of output file to be created Optional	TARGET_EXTENT	[extent]	Sets the georeferenced extent of the output file to be created (in the *Target CRS* by default. In the *CRS of the target raster extent*, if specified).

Fortsetzung auf der nächsten Seite

Tab. 23.153 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
CRS of the target raster extent Optional	`TARGET_EXTENT_CRS`	[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.
Use multithreaded warping implementation	`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.
Additional command-line parameters Optional	`EXTRA`	[string] Default: None	Add extra GDAL command line options.
Reprojected	`OUTPUT`	[raster] Default: ‚[Save to temporary file]'	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Reprojected	`OUTPUT`	[raster] Default: `[Save to temporary file]`	Reprojected output raster layer

Python code

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.

23.2.6 Vector conversion

Convert format

Converts any OGR-supported vector layer into another OGR-supported format.

This algorithm is derived from the ogr2ogr utility.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
Converted	`OUTPUT`	[same as input]	Specification of the output vector layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here. 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.

Ausgaben

Label	Name	Type	Beschreibung
Converted	`OUTPUT`	[same as input]	The output vector layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Input raster layer	`INPUT_RASTER`	[raster]	Input raster layer
Field to use for a burn-in value Optional	`FIELD`	[tablefield: numeric]	Defines the attribute field to use to set the pixels values

Fortsetzung auf der nächsten Seite

Tab. 23.154 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Add burn in values to existing raster values	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.
Additional command-line parameters Optional	EXTRA	[string] Default: ‚'	Add extra GDAL command line options

Ausgaben

Label	Name	Type	Beschreibung
Rasterized	OUTPUT	[raster]	The overwritten input raster layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	Input vector layer
Input raster layer	INPUT_RASTER	[raster]	Input raster layer
A fixed value to burn	BURN	[number] Default: 0.0	The value to burn
Add burn in values to existing raster values	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.
Additional command-line parameters Optional	EXTRA	[string] Default: ‚'	Add extra GDAL command line options

Ausgaben

Label	Name	Type	Beschreibung
Rasterized	`OUTPUT`	[raster]	The overwritten input raster layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Field to use for a burn-in value Optional	`FIELD`	[tablefield: numeric]	Defines the attribute field from which the attributes for the pixels should be chosen
A fixed value to burn Optional	`BURN`	[number] Default: 0.0	A fixed value to burn into a band for all features.
Output raster size units	`UNITS`	[enumeration] Default: 0	Units to use when defining the output raster size/resolution. One of: • 0 — Pixels • 1 — Georeferenced units
Width/Horizontal resolution	`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.
Height/Vertical resolution	`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.
Output extent	`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.

Fortsetzung auf der nächsten Seite

Tab. 23.156 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Assign a specified nodata value to output bands Optional	NODATA	[number] Default: 0.0	Assigns a specified nodata value to output bands
Additional creation options 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 *GDAL driver options section*).
Output data type	DATA_TYPE	[enumeration] Default: 5	Defines the format of the output raster file. Optionen: • 0 — Byte • 1 — Int16 • 2 — UInt16 • 3 — UInt32 • 4 — Int32 • 5 — Float32 • 6 — Float64 • 7 — CInt16 • 8 — CInt32 • 9 — CFloat32 • 10 — CFloat64
Pre-initialize the output image with value 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.
Invert rasterization	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.
Rasterized	OUTPUT	[raster] Default: ‚[Save to temporary file]'	Specification of the output raster layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here 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.

Ausgaben

Label	Name	Type	Beschreibung
Rasterized	OUTPUT	[raster]	Output raster layer

Python code

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.

23.2.7 Vector geoprocessing

Buffer vectors

Create buffers around the features of a vector layer.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Geometry column name	`GEOMETRY`	[string] Default: ‚geometry'	The name of the input layer geometry column to use
Buffer distance	`DISTANCE`	[number] Default: 10.0	Minimum: 0.0
Dissolve by attribute Optional	`FIELD`	[tablefield: any] Default: None	Field to use for dissolving
Dissolve results	`DISSOLVE`	[boolean] Default: False	If set, the result is dissolved. If no field is set for dissolving, all the buffers are dissolved into one feature.
Produce one feature for each geometry in any kind of geometry collection in the source file	`EXPLODE_COLLECTIONS`	[boolean] Default: False	
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
Buffer	`OUTPUT`	[vector: polygon] Default: `[Save to temporary file]`	Specify the output buffer layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Buffer	`OUTPUT`	[vector: polygon]	The output buffer layer

Python code

Algorithm ID: `gdal:buffervectors`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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 ogr2ogr utility.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Clip extent	`EXTENT`	[extent]	Defines the bounding box that should be used for the output vector file. It has to be defined in target CRS coordinates.
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚‘ (no additional options)	Additional GDAL creation options.
Clipped (extent)	`OUTPUT`	[same as input] Default: `[Save to temporary file]`	Specify the output (clipped) layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Clipped (extent)	`OUTPUT`	[same as input]	The output (clipped) layer. The default format is „ESRI Shapefile“.

Python code

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 ogr2ogr utility.

Parameter

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	The input vector layer
Mask layer	`MASK`	[vector: polygon]	Layer to be used as clipping extent for the input vector layer.
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚‘ (no additional options)	Additional GDAL creation options.
Clipped (mask)	`OUTPUT`	[same as input] Default: `[Save to temporary file]`	The output (masked) layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Clipped (mask)	`OUTPUT`	[same as input]	The output (masked) layer. The default format is „ESRI Shapefile“.

Python code

Algorithm ID: `gdal:clipvectorbymasklayer`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	The input layer to dissolve
Dissolve field (optional)	FIELD	[tablefield: any]	The field of the input layer to use for dissolving
Geometry column name	GEOMETRY	[string] Default: ‚geometry'	The name of the input layer geometry column to use for dissolving.
Produce one feature for each geometry in any kind of geometry collection in the source file	EXPLODE_COLLECTIONS	[boolean] Default: False	Produce one feature for each geometry in any kind of geometry collection in the source file
Keep input attributes	KEEP_ATTRIBUTES	[boolean] Default: False	Keep all attributes from the input layer
Count dissolved features	COUNT_FEATURES	[boolean] Default: False	Count the dissolved features and include it in the output layer.
Compute area and perimeter of dissolved features	COMPUTE_AREA	[boolean] Default: False	Compute the area and perimeter of dissolved features and include them in the output layer
Compute min/max/sum/mean for attribute	COMPUTE_STATISTICS	[boolean] Default: False	Calculate statistics (min, max, sum and mean) for the numeric attribute specified and include them in the output layer
Numeric attribute to calculate statistics on (optional)	STATISTICS_ATTRIBUTE	[tablefield: numeric]	The numeric attribute to calculate statistics on
Additional creation options (optional)	OPTIONS	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
Dissolved	OUTPUT	[same as input] Default: `[Save to temporary file]`	Specify the output layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Dissolved	OUTPUT	[same as input]	The output multipart geometry layer (with dissolved geometries)

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	The input line layer
Geometry column name	`GEOMETRY`	[string] Default: ‚geometry'	The name of the input layer geometry column to use
Offset distance (left-sided: positive, right-sided: negative)	`DISTANCE`	[number] Default: 10.0	
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
Offset curve	`OUTPUT`	[vector: line] Default: `[Save to temporary file]`	Specify the output line layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Offset curve	`OUTPUT`	[vector: line]	The output offset curve layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: line]	The input line layer
Geometry column name	`GEOMETRY`	[string] Default: ‚geometry'	The name of the input layer geometry column to use
Buffer distance	`DISTANCE`	[number] Default: 10.0	
Buffer side	`BUFFER_SIDE`	[enumeration] Default: 0	One of: • 0 — Right • 1 — Left
Dissolve by attribute Optional	`FIELD`	[tablefield: any] Default: None	Field to use for dissolving
Dissolve all results	`DISSOLVE`	[boolean] Default: False	If set, the result is dissolved. If no field is set for dissolving, all the buffers are dissolved into one feature.
Produce one feature for each geometry in any kind of geometry collection in the source file	`EXPLODE_COLLECTIONS`	[boolean] Default: False	
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
One-sided buffer	`OUTPUT`	[vector: polygon] Default: `[Save to temporary file]`	Specify the output buffer layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
One-sided buffer	`OUTPUT`	[vector: polygon]	The output buffer layer

Python code

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: line]	The input line layer
Geometry column name	GEOMETRY	[string] Default: ‚geometry'	The name of the input layer geometry column to use
Distance from line start represented as a fraction of line length	DISTANCE	[number] Default: 0.5 (middle of the line)	
Additional creation options (optional)	OPTIONS	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
Points along line	OUTPUT	[vector: point] Default: `[Save to temporary file]`	Specify the output point layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Points along line	OUTPUT	[vector: point]	The output point layer

Python code

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.

23.2.8 Vector miscellaneous

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

Label	Name	Type	Beschreibung
Input layer	INPUT	[vector: any]	OGR-supported input vector layer
SQL expression	SQL	[string]	Defines the SQL query, for example `SELECT * FROM my_table WHERE name is not null`.
SQL dialect	DIALECT	[enumeration] Default: 0	SQL dialect to use. One of: • 0 — None • 1 — OGR SQL • 2 — SQLite
Additional creation options (optional)	OPTIONS	[string] Default: ‚' (no additional options)	Additional GDAL creation options.
SQL result	OUTPUT	[vector: any]	Specification of the output layer. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here. 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.

Ausgaben

Label	Name	Type	Beschreibung
SQL result	OUTPUT	[vector: any]	Vector layer created by the query

Python code

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

Label	Name	Type	Beschreibung
Database (connection name)	DATABASE	[string]	The PostgreSQL database to connect to
Input layer	INPUT	[vector: any]	OGR-supported vector layer to export to the database
Shape encoding Optional	SHAPE_ENCODING	[string] Default: ,‘	Sets the encoding to apply to the data
Output geometry type	GTYPE	[enumeration] Default: 0	Defines the output geometry type. One of: • 0 — • 1 — NONE • 2 — GEOMETRY • 3 — POINT • 4 — LINESTRING • 5 — POLYGON • 6 — GEOMETRYCOLLECTION • 7 — MULTIPOINT • 8 — MULTIPOLYGON • 9 — MULTILINESTRING
Assign an output CRS Optional	A_SRS	[crs] Default: None	Defines the output CRS of the database table
Reproject to this CRS on output Optional	T_SRS	[crs] Default: None	Reprojects/transforms to this CRS on output
Override source CRS Optional	S_SRS	[crs] Default: None	Overrides the input layer CRS
Schema (schema name) Optional	SCHEMA	[string] Default: ,public‘	Defines the schema for the database table
Table to export to (leave blank to use layer name) 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.
Primary Key (new field) Optional	PK	[string] Default: ,id‘	Defines which attribute field will be the primary key of the database table
Primary Key (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
Geometry column name Optional	GEOCOLUMN	[string] Default: ,geom‘	Defines in which attribute field of the database there will be the geometry information
Vector dimensions Optional	DIM	[enumeration] Default: 0 (2D)	Defines if the vector file to be imported has 2D or 3D data. One of: • 0 — 2 • 1 — 3

Fortsetzung auf der nächsten Seite

Tab. 23.163 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Distance tolerance for simplification Optional	SIMPLIFY	[string] Default: ,‘	Defines a distance tolerance for the simplification of the vector geometries to be imported. By default there is no simplification.
Maximum distance between 2 nodes (densification) Optional	SEGMENTIZE	[string] Default: ,‘	The maximum distance between two nodes. Used to create intermediate points. By default there is no densification.
Select features by extent (defined in input layer CRS) Optional	SPAT	[extent] Default: None	You can select features from a given extent that will be in the output table.
Clip the input layer using the above (rectangle) extent Optional	CLIP	[boolean] Default: False	The input layer will be clipped by the extent you defined before
Select features using a SQL „WHERE“ statement (Ex: column=“value“) Optional	WHERE	[string] Default: ,‘	Defines with a SQL „WHERE“ statement which features should be selected from the input layer
Group N features per transaction (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.
Overwrite existing table 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.
Append to existing table 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.
Append and add new fields to existing table 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.
Do not launder columns/table names 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).
Do not create Spatial Index Optional	INDEX	[boolean] Default: False	Prevents a spatial index for the output table from being created. By default, a spatial index is added.
Continue after a failure, skipping the failed feature Optional	SKIPFAILURES	[boolean] Default: False	
Promote to Multipart Optional	PROMOTETOMULTI	[boolean] Default: True	Casts features geometry type to multipart in the output table
Keep width and precision of input attributes Optional	PRECISION	[boolean] Default: True	Avoids modifying column attributes to comply with input data

Fortsetzung auf der nächsten Seite

Tab. 23.163 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Additional creation options (optional)	`OPTIONS`	[string] Default: ‚' (no additional options)	Additional GDAL creation options.

Ausgaben

This algorithm has no output.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	OGR-supported vector layer to export to the database
Shape encoding Optional	`SHAPE_ENCODING`	[string] Default: ‚'	Sets the encoding to apply to the data
Output geometry type	`GTYPE`	[enumeration] Default: 0	Defines the output geometry type. One of: • 0 — • 1 — NONE • 2 — GEOMETRY • 3 — POINT • 4 — LINESTRING • 5 — POLYGON • 6 — GEOMETRYCOLLECTION • 7 — MULTIPOINT • 8 — MULTIPOLYGON • 9 — MULTILINESTRING
Assign an output CRS Optional	`A_SRS`	[crs] Default: None	Defines the output CRS of the database table
Reproject to this CRS on output Optional	`T_SRS`	[crs] Default: None	Reprojects/transforms to this CRS on output

Fortsetzung auf der nächsten Seite

Tab. 23.164 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Override source CRS Optional	S_SRS	[crs] Default: None	Overrides the input layer CRS
Host Optional	HOST	[string] Default: ‚localhost‘	Name of the database host
Port Optional	PORT	[string] Default: ‚5432‘	Port number the PostgreSQL database server listens on
Username Optional	USER	[string] Default: ‚‘	User name used to log in to the database
Database name Optional	DBNAME	[string] Default: ‚‘	Name of the database
Password Optional	PASSWORD	[string] Default: ‚‘	Password used with Username to connect to the database
Schema (schema name) Optional	SCHEMA	[string] Default: ‚public‘	Defines the schema for the database table
Table name, leave blank to use input name 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.
Primary Key (new field) Optional	PK	[string] Default: ‚id‘	Defines which attribute field will be the primary key of the database table
Primary Key (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
Geometry column name Optional	GEOCOLUMN	[string] Default: ‚geom‘	Defines in which attribute field to store the geometry information
Vector dimensions Optional	DIM	[enumeration] Default: 0 (2D)	Defines if the vector file to be imported has 2D or 3D data. One of: • 0 — 2D • 1 — 3D
Distance tolerance for simplification Optional	SIMPLIFY	[string] Default: ‚‘	Defines a distance tolerance for the simplification of the vector geometries to be imported. By default no simplification there is no simplification.
Maximum distance between 2 nodes (densification) Optional	SEGMENTIZE	[string] Default: ‚‘	The maximum distance between two nodes. Used to create intermediate points. By default there is no densification.
Select features by extent (defined in input layer CRS) Optional	SPAT	[extent] Default: None	You can select features from a given extent that will be in the output table.
Clip the input layer using the above (rectangle) extent Optional	CLIP	[boolean] Default: False	The input layer will be clipped by the extent you defined before

Fortsetzung auf der nächsten Seite

Tab. 23.164 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Fields to include (leave empty to use all fields) Optional	FIELDS	[string] [list] Default: []	Defines fields to keep from the imported vector file. If none is selected, all the fields are imported.
Select features using a SQL „WHERE" statement (Ex: column="value") Optional	WHERE	[string] Default: ,'	Defines with a SQL „WHERE" statement which features should be selected for the output table
Group N features per transaction (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.
Overwrite existing table 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.
Append to existing table 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.
Append and add new fields to existing table 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.
Do not launder columns/table names 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).
Do not create Spatial Index Optional	INDEX	[boolean] Default: False	Prevents a spatial index for the output table from being created. By default, a spatial index is added.
Continue after a failure, skipping the failed feature Optional	SKIPFAILURES	[boolean] Default: False	
Promote to Multipart Optional	PROMOTETOMULTI	[boolean] Default: True	Casts features geometry type to multipart in the output table
Keep width and precision of input attributes Optional	PRECISION	[boolean] Default: True	Avoids modifying column attributes to comply with input data
Additional creation options (optional)	OPTIONS	[string] Default: ,' (no additional options)	Additional GDAL creation options.

Ausgaben

This algorithm has no output.

Python code

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

Label	Name	Type	Beschreibung
Input layer	`INPUT`	[vector: any]	Input vector layer
Summary output only Optional	`SUMMARY_ONLY`	[boolean] Default: True	
Suppress metadata info Optional	`NO_METADATA`	[boolean] Default: False	
Layer information	`OUTPUT`	[html] Default: `[Save to temporary file]`	Specify the output HTML file that includes the file information. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here. If no HTML-file is defined the output will be written to a temporary file

Ausgaben

Label	Name	Type	Beschreibung
Layer information	`OUTPUT`	[html]	The output HTML-file that includes the file information.

Python code

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.

23.3 LAStools algorithm provider

LAStools is a collection of highly efficient, multicore command line tools for LiDAR data processing.

23.3.1 blast2dem

Beschreibung

Turns points (up to billions) via seamless Delaunay triangulation implemented using streaming into large elevation, intensity, or RGB rasters.

For more info see the blast2dem page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	INPUT_LASLAZ	[file]	The file containing the points to be rastered in LAS/LAZ format.

Fortsetzung auf der nächsten Seite

Tab. 23.166 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
filter (by return, classification, flag)	FILTER_RETURN_CLASS_FLAGS1	[enumeration] Default: 0	Specifies which points to use to construct the temporary TIN that is then rasterized. One of: • 0 — — • 1 — keep_last • 2 — keep_first • 3 — keep_middle • 4 — keep_single • 5 — drop_single • 6 — keep_double • 7 — keep_class 2 • 8 — keep_class 2 8 • 9 — keep_class 8 • 10 — keep_class 6 • 11 — keep_class 9 • 12 — keep_class 3 4 5 • 13 — keep_class 2 6 • 14 — drop_class 7 • 15 — drop_withheld • 16 — drop_synthetic • 17 — drop_overlap • 18 — keep_withheld • 19 — keep_synthetic • 20 — keep_keypoint • 21 — keep_overlap
step size / pixel size	STEP	[number] Default: 1.0	Specifies the size of the cells of the grid the TIN is rasterized onto
Attribute	ATTRIBUTE	[enumeration] Default: 0	Specifies the attribute that is to be rastered. One of: • 0 — elevation • 1 — slope • 2 — intensity • 3 — rgb
Product	PRODUCT	[enumeration] Default: 0	Specifies how the attribute is to be turned into raster values. One of: • 0 — actual values • 1 — hillshade • 2 — gray • 3 — false
Use tile bounding box (after tiling with buffer)	USE_TILE_BB	[boolean] Default: False	Specifies to limit the rastered area to the tile bounding box (only meaningful for input LAS/LAZ tiles that were created with lastile).
additional command line parameter(s) Optional	ADDITIONAL_OPTIONS	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.

Fortsetzung auf der nächsten Seite

Tab. 23.166 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output raster file	`OUTPUT_RASTER`	[raster] Default: `[Skip output]`	Specifies where the output raster is stored. Use image rasters like TIF, PNG, and JPG for false color, gray ramps, and hillshades. Use value rasters like TIF, BIL, IMG, ASC, DTM, FLT, XYZ, and CSV for actual values. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output raster file	`OUTPUT_RASTER`	[raster]	The output raster

Python code

Algorithm ID: `lastools:blast2dem`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.2 blast2iso

Beschreibung

Turns points (up to billions) via seamless Delaunay triangulation implemented using streaming into iso-contour lines.

For more info see the blast2iso page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file containing the points to be used for creating iso-contour lines.
smooth underlying TIN	`SMOOTH`	[number] Default: 0	Specifies if and with how many passes the temporary TIN should be smoothed
extract isoline with a spacing of	`ISO_EVERY`	[number] Default: 10.0	Specifies spacing at which iso-contour lines are getting extracted (contour interval)

Fortsetzung auf der nächsten Seite

Tab. 23.167 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
clean isolines shorter than (0 = do not clean)	`CLEAN`	[number] Default: 0.0	Omits iso-contour lines that are shorter than the specified length
simplify segments shorter than (0 = do not simplify)	`SIMPLIFY_LENGTH`	[number] Default: 0.0	Rudimentary simplification of iso-contour line segments that are shorter than the specified length.
simplify segment pairs with area less than (0 = do not simplify)	`SIMPLIFY_AREA`	[number] Default: 0.0	Rudimentary simplification of bumps formed by consecutive line segments whose area is smaller than the specified size.
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output vector file	`OUTPUT_VECTOR`	[vector: line] Default: `[Skip output]`	Specifies where the output vector is stored. Use SHP or WKT output files. If your input LiDAR file is in geographic coordinates (long/lat) or has geo-referencing information (but only then) you can also create a KML output file. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output vector file	`OUTPUT_VECTOR`	[vector: line]	The output line vector layer with contours

Python code

Algorithm ID: `lastools:blast2iso`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.3 las2dem

Beschreibung

Turns points (up to 20 million) via a temporary Delaunay triangulation that is rasterized with a user-defined step size into an elevation, intensity, or RGB raster.

For more info see the las2dem page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	CPU64	[boolean] Default: False	
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	INPUT_LASLAZ	[file]	The file containing the points to be rastered in LAS/LAZ format.
filter (by return, classification, flags)	FILTER_RETURN_CLASS_FLAGS1	[enumeration] Default: 0	Specifies which points to use to construct the temporary TIN that is then rasterized. One of: • 0 — — • 1 — keep_last • 2 — keep_first • 3 — keep_middle • 4 — keep_single • 5 — drop_single • 6 — keep_double • 7 — keep_class 2 • 8 — keep_class 2 8 • 9 — keep_class 8 • 10 — keep_class 6 • 11 — keep_class 9 • 12 — keep_class 3 4 5 • 13 — keep_class 3 • 14 — keep_class 4 • 15 — keep_class 5 • 16 — keep_class 2 6 • 17 — drop_class 7 • 18 — drop_withheld • 19 — drop_synthetic • 20 — drop_overlap • 21 — keep_withheld • 22 — keep_synthetic • 23 — keep_keypoint • 24 — keep_overlap
step size / pixel size	STEP	[number] Default: 1.0	Specifies the size of the cells of the grid the TIN is rasterized onto
Attribute	ATTRIBUTE	[enumeration] Default: 0	Specifies the attribute to rasterise. One of: • 0 — elevation • 1 — slope • 2 — intensity • 3 — rgb • 4 — edge_longest • 5 — edge_shortest

Fortsetzung auf der nächsten Seite

Tab. 23.168 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Product	`PRODUCT`	[enumeration] Default: 0	Specifies how the attribute is to be turned into raster values. One of: • 0 — actual values • 1 — hillshade • 2 — gray • 3 — false
Use tile bounding box (after tiling with buffer)	`USE_TILE_BB`	[boolean] Default: False	Specifies to limit the rastered area to the tile bounding box (only meaningful for input LAS/LAZ tiles that were created with lastile).
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output raster file	`OUTPUT_RASTER`	[raster] Default: `[Skip output]`	Specifies where the output raster is stored. Use image rasters like TIF, PNG, and JPG for false color, gray ramps, and hillshades. Use value rasters like TIF, BIL, IMG, ASC, DTM, FLT, XYZ, and CSV for actual values. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output raster file	`OUTPUT_RASTER`	[raster]	The output raster

Python code

Algorithm ID: `lastools:las2dem`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.4 las2iso

Beschreibung

Turns point clouds (up to 20 million per file) into iso-contour lines by creating a temporary Delaunay triangulation on which the contours are then traced.

For more info see the las2iso page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	CPU64	[boolean] Default: False	
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	INPUT_LASLAZ	[file]	The file containing the points to be used for creating iso-contour lines.
smooth underlying TIN	SMOOTH	[number] Default: 0	Specifies if and with how many passes the temporary TIN should be smoothed
extract isoline with a spacing of	ISO_EVERY	[number] Default: 10.0	Specifies spacing at which iso-contour lines are getting extracted (contour interval)
clean isolines shorter than (0 = do not clean)	CLEAN	[number] Default: 0.0	Omits iso-contour lines that are shorter than the specified length
simplify segments shorter than (0 = do not simplify)	SIMPLIFY_LENGTH	[number] Default: 0.0	Rudimentary simplification of iso-contour line segments that are shorter than the specified length.
simplify segment pairs with area less than (0 = do not simplify)	SIMPLIFY_AREA	[number] Default: 0.0	Rudimentary simplification of bumps formed by consecutive line segments whose area is smaller than the specified size.
additional command line parameter(s) Optional	ADDITIONAL_OPTIONS	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output vector file	OUTPUT_VECTOR	[vector: line] Default: [Skip output]	Specifies where the output vector is stored. Use SHP or WKT output files. If your input LiDAR file is in geographic coordinates (long/lat) or has geo-referencing information (but only then) you can also create a KML output file. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output vector file	OUTPUT_VECTOR	[vector: line]	The output line vector layer with contours

Python code

Algorithm ID: `lastools:las2iso`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.5 las2las_filter

Beschreibung

Uses las2las to filter LiDAR points based on different attributes and to write the surviving subset of points to a new LAZ or LAS file.

For more info see the las2las page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file containing the points to be used for creating iso-contour lines.

Fortsetzung auf der nächsten Seite

Tab. 23.170 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
filter (by return, classification, flags)	`FILTER_RETURN_CLASS_FLAGS1`	[enumeration] Default: 0	Filters points based on various options such as return, classification, or flags. One of: • 0 — — • 1 — keep_last • 2 — keep_first • 3 — keep_middle • 4 — keep_single • 5 — drop_single • 6 — keep_double • 7 — keep_class 2 • 8 — keep_class 2 8 • 9 — keep_class 8 • 10 — keep_class 6 • 11 — keep_class 9 • 12 — keep_class 3 4 5 • 13 — keep_class 3 • 14 — keep_class 4 • 15 — keep_class 5 • 16 — keep_class 2 6 • 17 — drop_class 7 • 18 — drop_withheld • 19 — drop_synthetic • 20 — drop_overlap • 21 — keep_withheld • 22 — keep_synthetic • 23 — keep_keypoint • 24 — keep_overlap

Fortsetzung auf der nächsten Seite

Tab. 23.170 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
second filter (by return, classification, flags)	FILTER_RETURN_CLASS_FLAGS2	[enumeration] Default: 0	Filters points based on various options such as return, classification, or flags. One of: • 0 — — • 1 — keep_last • 2 — keep_first • 3 — keep_middle • 4 — keep_single • 5 — drop_single • 6 — keep_double • 7 — keep_class 2 • 8 — keep_class 2 8 • 9 — keep_class 8 • 10 — keep_class 6 • 11 — keep_class 9 • 12 — keep_class 3 4 5 • 13 — keep_class 3 • 14 — keep_class 4 • 15 — keep_class 5 • 16 — keep_class 2 6 • 17 — drop_class 7 • 18 — drop_withheld • 19 — drop_synthetic • 20 — drop_overlap • 21 — keep_withheld • 22 — keep_synthetic • 23 — keep_keypoint • 24 — keep_overlap
filter (by coordinate, intensity, GPS time, …)	FILTER_COORDS_INTENSITY1	[enumeration] Default: 0	Filters points based on various other options (that require a value as argument). One of: • 0 — — • 1 — drop_x_above • 2 — drop_x_below • 3 — drop_y_above • 4 — drop_y_below • 5 — drop_z_above • 6 — drop_z_below • 7 — drop_intensity_above • 8 — drop_intensity_below • 9 — drop_gps_time_above • 10 — drop_gps_time_below • 11 — drop_scan_angle_above • 12 — drop_scan_angle_below • 13 — keep_point_source • 14 — drop_point_source • 15 — drop_point_source_above • 16 — drop_point_source_below • 17 — keep_user_data • 18 — drop_user_data • 19 — drop_user_data_above • 20 — drop_user_data_below • 21 — keep_every_nth • 22 — keep_random_fraction • 23 — thin_with_grid

Fortsetzung auf der nächsten Seite

Tab. 23.170 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
value for filter (by coordinate, intensity, GPS time, …)	`FILTER_COORDS_INTENSITY1_ARG`	[number] Default: None	The value to use as the argument for the filter selected above
second filter (by coordinate, intensity, GPS time, …)	`FILTER_COORDS_INTENSITY2`	[enumeration] Default: 0	Filters points based on various other options (that require a value as argument). One of: • 0 — — • 1 — drop_x_above • 2 — drop_x_below • 3 — drop_y_above • 4 — drop_y_below • 5 — drop_z_above • 6 — drop_z_below • 7 — drop_intensity_above • 8 — drop_intensity_below • 9 — drop_gps_time_above • 10 — drop_gps_time_below • 11 — drop_scan_angle_above • 12 — drop_scan_angle_below • 13 — keep_point_source • 14 — drop_point_source • 15 — drop_point_source_above • 16 — drop_point_source_below • 17 — keep_user_data • 18 — drop_user_data • 19 — drop_user_data_above • 20 — drop_user_data_below • 21 — keep_every_nth • 22 — keep_random_fraction • 23 — thin_with_grid
value for second filter (by coordinate, intensity, GPS time, …)	`FILTER_COORDS_INTENSITY2_ARG`	[number] Default: None	The value to use as the argument for the filter selected above
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file] Default: `[Skip output]`	Specifies where the output point cloud is stored. Use LAZ for compressed output, LAS for uncompressed output, and TXT for ASCII. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output LAS/LAZ file	OUTPUT_LASLAZ	[file]	The output LAS/LAZ format file

Python code

Algorithm ID: `lastools:las2las_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.

23.3.6 las2las_project

Transform LAS/LAZ files in a folder to another CRS.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	CPU64	[boolean] Default: False	
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	INPUT_LASLAZ	[file]	Input LAS/LAZ file
source projection	SOURCE_PROJECTION	[enumeration] Default: 0	One of: • 0 — — • 1 — epsg • 2 — utm • 3 — sp83 • 4 — sp27 • 5 — longlat • 6 — latlong • 7 — ecef

Fortsetzung auf der nächsten Seite

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
source utm zone	SOURCE_UTM	[enumeration] Default: 0	One of: • 0 — — • 1 — 1 (north) • 2 — 2 (north) • 3 — 3 (north) • 4 — 4 (north) • 5 — 5 (north) • 6 — 6 (north) • 7 — 7 (north) • 8 — 8 (north) • 9 — 9 (north) • 10 — 10 (north) • 11 — 11 (north) • 12 — 12 (north) • 13 — 13 (north) • 14 — 14 (north) • 15 — 15 (north) • 16 — 16 (north) • 17 — 17 (north) • 18 — 18 (north) • 19 — 19 (north) • 20 — 20 (north) • 21 — 21 (north) • 22 — 22 (north) • 23 — 23 (north) • 24 — 24 (north) • 25 — 25 (north) • 26 — 26 (north) • 27 — 27 (north) • 28 — 28 (north) • 29 — 29 (north) • 30 — 30 (north) • 31 — 31 (north) • 32 — 32 (north) • 33 — 33 (north) • 34 — 34 (north) • 35 — 35 (north) • 36 — 36 (north) • 37 — 37 (north) • 38 — 38 (north) • 39 — 39 (north) • 40 — 40 (north) • 41 — 41 (north) • 42 — 42 (north) • 43 — 43 (north) • 44 — 44 (north) • 45 — 45 (north) • 46 — 46 (north) • 47 — 47 (north) • 48 — 48 (north) • 49 — 49 (north) • 50 — 50 (north) • 51 — 51 (north) • 52 — 52 (north) • 53 — 53 (north) • 54 — 54 (north) • 55 — 55 (north) • 56 — 56 (north) • 57 — 57 (north) • 58 — 58 (north) • 59 — 59 (north) • 60 — 60 (north) • 61 — 1 (south)

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
source state plane code	`SOURCE_SP`	[enumeration] Default: 0	One of: • 0 — — • 1 — AK_10 • 2 — AK_2 • 3 — AK_3 • 4 — AK_4 • 5 — AK_5 • 6 — AK_6 • 7 — AK_7 • 8 — AK_8 • 9 — AK_9 • 10 — AL_E • 11 — AL_W • 12 — AR_N • 13 — AR_S • 14 — AZ_C • 15 — AZ_E • 16 — AZ_W • 17 — CA_I • 18 — CA_II • 19 — CA_III • 20 — CA_IV • 21 — CA_V • 22 — CA_VI • 23 — CA_VII • 24 — CO_C • 25 — CO_N • 26 — CO_S • 27 — CT • 28 — DE • 29 — FL_E • 30 — FL_N • 31 — FL_W • 32 — GA_E • 33 — GA_W • 34 — HI_1 • 35 — HI_2 • 36 — HI_3 • 37 — HI_4 • 38 — HI_5 • 39 — IA_N • 40 — IA_S • 41 — ID_C • 42 — ID_E • 43 — ID_W • 44 — IL_E • 45 — IL_W • 46 — IN_E • 47 — IN_W • 48 — KS_N • 49 — KS_S • 50 — KY_N • 51 — KY_S • 52 — LA_N • 53 — LA_S • 54 — MA_I • 55 — MA_M • 56 — MD • 57 — ME_E • 58 — ME_W • 59 — MI_C • 60 — MI_N • 61 — MI_S

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
target projection	TARGET_PROJECTION	[enumeration] Default: 0	One of: • 0 — — • 1 — epsg • 2 — utm • 3 — sp83 • 4 — sp27 • 5 — longlat • 6 — latlong • 7 — ecef

Fortsetzung auf der nächsten Seite

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
target utm zone	`TARGET_UTM`	[enumeration] Default: 0	One of: • 0 — — • 1 — 1 (north) • 2 — 2 (north) • 3 — 3 (north) • 4 — 4 (north) • 5 — 5 (north) • 6 — 6 (north) • 7 — 7 (north) • 8 — 8 (north) • 9 — 9 (north) • 10 — 10 (north) • 11 — 11 (north) • 12 — 12 (north) • 13 — 13 (north) • 14 — 14 (north) • 15 — 15 (north) • 16 — 16 (north) • 17 — 17 (north) • 18 — 18 (north) • 19 — 19 (north) • 20 — 20 (north) • 21 — 21 (north) • 22 — 22 (north) • 23 — 23 (north) • 24 — 24 (north) • 25 — 25 (north) • 26 — 26 (north) • 27 — 27 (north) • 28 — 28 (north) • 29 — 29 (north) • 30 — 30 (north) • 31 — 31 (north) • 32 — 32 (north) • 33 — 33 (north) • 34 — 34 (north) • 35 — 35 (north) • 36 — 36 (north) • 37 — 37 (north) • 38 — 38 (north) • 39 — 39 (north) • 40 — 40 (north) • 41 — 41 (north) • 42 — 42 (north) • 43 — 43 (north) • 44 — 44 (north) • 45 — 45 (north) • 46 — 46 (north) • 47 — 47 (north) • 48 — 48 (north) • 49 — 49 (north) • 50 — 50 (north) • 51 — 51 (north) • 52 — 52 (north) • 53 — 53 (north) • 54 — 54 (north) • 55 — 55 (north) • 56 — 56 (north) • 57 — 57 (north) • 58 — 58 (north) • 59 — 59 (north) • 60 — 60 (north)

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
target state plane code	`TARGET_SP`	[enumeration] Default: 0	One of: • 0 — — • 1 — AK_10 • 2 — AK_2 • 3 — AK_3 • 4 — AK_4 • 5 — AK_5 • 6 — AK_6 • 7 — AK_7 • 8 — AK_8 • 9 — AK_9 • 10 — AL_E • 11 — AL_W • 12 — AR_N • 13 — AR_S • 14 — AZ_C • 15 — AZ_E • 16 — AZ_W • 17 — CA_I • 18 — CA_II • 19 — CA_III • 20 — CA_IV • 21 — CA_V • 22 — CA_VI • 23 — CA_VII • 24 — CO_C • 25 — CO_N • 26 — CO_S • 27 — CT • 28 — DE • 29 — FL_E • 30 — FL_N • 31 — FL_W • 32 — GA_E • 33 — GA_W • 34 — HI_1 • 35 — HI_2 • 36 — HI_3 • 37 — HI_4 • 38 — HI_5 • 39 — IA_N • 40 — IA_S • 41 — ID_C • 42 — ID_E • 43 — ID_W • 44 — IL_E • 45 — IL_W • 46 — IN_E • 47 — IN_W • 48 — KS_N • 49 — KS_S • 50 — KY_N • 51 — KY_S • 52 — LA_N • 53 — LA_S • 54 — MA_I • 55 — MA_M • 56 — MD • 57 — ME_E • 58 — ME_W • 59 — MI_C • 60 — MI_N • 61 — MI_S

Tab. 23.171 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚'	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[folder] Default: `[Save to temporary folder]`	Specifies where the folder for the output point clouds. One of: • Skip Output • Save to a Temporary Directory • Save to Directory… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file]	The output LAS/LAZ format file

Python code

Algorithm ID: `lastools:las2las_project`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.7 las2las_transform

Beschreibung

Uses las2las to filter LiDAR points based on different attributes and to write the surviving subset of points to a new LAZ or LAS file.

For more info see the las2las page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The first file containing points to be merged

Fortsetzung auf der nächsten Seite

Tab. 23.172 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
transform (coordinates)	TRANSFORM_COORDINATE1	[enumeration] Default: 0	Either translate, scale, or clamp the X, Y, or Z coordinate by the value specified below. One of: • 0 — — • 1 — translate_x • 2 — translate_y • 3 — translate_z • 4 — scale_x • 5 — scale_y • 6 — scale_z • 7 — clamp_z_above • 8 — clamp_z_below
value for transform (coordinates)	TRANSFORM_COORDINATE1_ARG	[string] Default: ‚‘	The value that specifies the amount of translating, scaling, or clamping done by the transform selected above.
second transform (coordinates)	TRANSFORM_COORDINATE2	[enumeration] Default: 0	Either translate, scale, or clamp the X, Y, or Z coordinate by the value specified below. One of: • 0 — — • 1 — translate_x • 2 — translate_y • 3 — translate_z • 4 — scale_x • 5 — scale_y • 6 — scale_z • 7 — clamp_z_above • 8 — clamp_z_below
value for second transform (coordinates)	TRANSFORM_COORDINATE2_ARG	[string] Default: ‚‘	The value that specifies the amount of translating, scaling, or clamping done by the transform selected above.
transform (intensities, scan angles, GPS times, …)	TRANSFORM_OTHER	[enumeration] Default: 0	Either translate, scale, or clamp the X, Y, or Z coordinate by the value specified below. One of: • 0 — — • 1 — scale_intensity • 2 — translate_intensity • 3 — clamp_intensity_above • 4 — clamp_intensity_below • 5 — scale_scan_angle • 6 — translate_scan_angle • 7 — translate_gps_time • 8 — set_classification • 9 — set_user_data • 10 — set_point_source • 11 — scale_rgb_up • 12 — scale_rgb_down • 13 — repair_zero_returns
value for transform (intensities, scan angles, GPS times, …)	TRANSFORM_OTHER_ARG	[string] Default: ‚‘	The value that specifies the amount of scaling, translating, clamping or setting that is done by the transform selected above.

Fortsetzung auf der nächsten Seite

Tab. 23.172 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
second transform (intensities, scan angles, GPS times, …)	`TRANSFORM_OTHER2`	[enumeration] Default: 0	Either translate, scale, or clamp the X, Y, or Z coordinate by the value specified below. One of: • 0 — — • 1 — scale_intensity • 2 — translate_intensity • 3 — clamp_intensity_above • 4 — clamp_intensity_below • 5 — scale_scan_angle • 6 — translate_scan_angle • 7 — translate_gps_time • 8 — set_classification • 9 — set_user_data • 10 — set_point_source • 11 — scale_rgb_up • 12 — scale_rgb_down • 13 — repair_zero_returns
value for second transform (intensities, scan angles, GPS times, …)	`TRANSFORM_OTHER2_ARG`	[string] Default: ,‘	The value that specifies the amount of scaling, translating, clamping or setting that is done by the transform selected above.
operations (first 7 need an argument)	`OPERATION`	[enumeration] Default: 0	One of: • 0 — — • 1 — set_point_type • 2 — set_point_size • 3 — set_version_minor • 4 — set_version_major • 5 — start_at_point • 6 — stop_at_point • 7 — remove_vlr • 8 — auto_reoffset • 9 — week_to_adjusted • 10 — adjusted_to_week • 11 — auto reoffset • 12 — scale_rgb_up • 13 — scale_rgb_down • 14 — remove_all_vlrs • 15 — remove_extra • 16 — clip_to_bounding_box
argument for operation	`OPERATIONARG`	[string] Default: ,‘	The value to use as the argument for the operation selected above
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file] Default: `[Skip output]`	Specifies where the output point cloud is stored. Use LAZ for compressed output, LAS for uncompressed output, and TXT for ASCII. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output LAS/LAZ file	OUTPUT_LASLAZ	[file]	The output (merged) LAS/LAZ format file

Python code

Algorithm ID: `lastools:las2las_transform`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.8 las2txt

Beschreibung

Translates a LAS/LAZ file to a text file.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	
run new 64 bit executable	CPU64	[boolean] Default: False	
open LAStools GUI	GUI	[boolean] Default: False	
input LAS/LAZ file	INPUT_LASLAZ	[file] Default: None	
parse_string	PARSE	[string] Default: ‚xyz‘	
additional command line parameters Optional	ADDITIONAL_OPTIONS	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output ASCII file	OUTPUT_GENERIC	[file] Default: `[Create temporary layer]`	Specify the output file. One of: • Create Temporary Layer (`TEMPORARY_OUTPUT`) • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output ASCII file	`OUTPUT_GENERIC`	[file]	The output file

Python code

Algorithm ID: `lastools:las2txt`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.9 lasindex

Beschreibung

<fügen Sie hier die Übersetzung des Algorithmus ein>

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	
input LAS/LAZ file	`INPUT_LASLAZ`	[file] Default: None	
append *.lax file to *.laz file	`APPEND_LAX`	[boolean] Default: False	
is mobile or terrestrial LiDAR (not airborne)	`MOBILE_OR_TERRESTRIAL`	[boolean] Default: False	
additional command line parameters Optional	`ADDITIONAL_OPTIONS`	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.

Ausgaben

The algorithm has no output.

Python code

Algorithm ID: `lastools:lasindex`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.10 lasgrid

Grids a selected attribute (e.g. elevation, intensity, classification, scan angle, …) of a large point clouds with a user-defined step size onto raster using a particular method (e.g. min, max, average).

For more info see the lasgrid page and its online README file.

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file containing the points to be rastered in LAS/LAZ format.

Fortsetzung auf der nächsten Seite

Tab. 23.175 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
filter (by return, classification, flags)	FILTER_RETURN_CLASS_FLAGS1	[enumeration] Default: 0	Specifies the subset of points to use for the gridding. One of: • 0 — — • 1 — keep_last • 2 — keep_first • 3 — keep_middle • 4 — keep_single • 5 — drop_single • 6 — keep_double • 7 — keep_class 2 • 8 — keep_class 2 8 • 9 — keep_class 8 • 10 — keep_class 6 • 11 — keep_class 9 • 12 — keep_class 3 4 5 • 13 — keep_class 3 • 14 — keep_class 4 • 15 — keep_class 5 • 16 — keep_class 2 6 • 17 — drop_class 7 • 18 — drop_withheld • 19 — drop_synthetic • 20 — drop_overlap • 21 — keep_withheld • 22 — keep_synthetic • 23 — keep_keypoint • 24 — keep_overlap
step size / pixel size	STEP	[number] Default: 1.0	Specifies the size of the cells of the grid the TIN is rasterized onto
Attribute	ATTRIBUTE	[enumeration] Default: 0	Specifies the attribute to rasterise. One of: • 0 — elevation • 1 — intensity • 2 — rgb • 3 — classification
Method	METHOD	[enumeration] Default: 0	Specifies how the attributes falling into one cell are turned into a raster value. One of: • 0 — lowest • 1 — heighest • 2 — average • 3 — stddev
use tile bounding box (after tiling with buffer)	USE_TILE_BB	[boolean] Default: False	Specifies to limit the rastered area to the tile bounding box (only meaningful for input LAS/LAZ tiles that were created with lastile).
additional command line parameter(s) Optional	ADDITIONAL_OPTIONS	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.

Fortsetzung auf der nächsten Seite

Tab. 23.175 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Output raster file	`OUTPUT_RASTER`	[raster] Default: `[Skip output]`	Specifies where the output raster is stored. Use image rasters like TIF, PNG, and JPG for false color, gray ramps, and hillshades. Use value rasters like TIF, BIL, IMG, ASC, DTM, FLT, XYZ, and CSV for actual values. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output raster file	`OUTPUT_RASTER`	[raster]	The output raster

Python code

Algorithm ID: `lastools:lasgrid`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.11 lasinfo

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file to get information about.
compute density	`COMPUTE_DENSITY`	[boolean] Default: False	
repair bounding box	`REPAIR_BB`	[boolean] Default: False	
repair counters	`REPAIR_COUNTERS`	[boolean] Default: False	

Fortsetzung auf der nächsten Seite

Tab. 23.176 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
histogram	HISTO1	[enumeration] Default: 0	First histogram. One of: • 0 — — • 1 — x • 2 — y • 3 — z • 4 — intensity • 5 — classification • 6 — scan_angle • 7 — user_data • 8 — point_source • 9 — gps_time • 10 — X • 11 — Y • 12 — Z • 13 — attribute0 • 14 — attribute1 • 15 — attribute2
bin size	HISTO1_BIN	[number] Default: 1.0	
histogram	HISTO2	[enumeration] Default: 0	Second histogram. One of: • 0 — — • 1 — x • 2 — y • 3 — z • 4 — intensity • 5 — classification • 6 — scan_angle • 7 — user_data • 8 — point_source • 9 — gps_time • 10 — X • 11 — Y • 12 — Z • 13 — attribute0 • 14 — attribute1 • 15 — attribute2
bin size	HISTO2_BIN	[number] Default: 1.0	

Fortsetzung auf der nächsten Seite

Tab. 23.176 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
histogram	`HISTO3`	[enumeration] Default: 0	Third histogram. One of: • 0 — — • 1 — x • 2 — y • 3 — z • 4 — intensity • 5 — classification • 6 — scan_angle • 7 — user_data • 8 — point_source • 9 — gps_time • 10 — X • 11 — Y • 12 — Z • 13 — attribute0 • 14 — attribute1 • 15 — attribute2
bin size	`HISTO3_BIN`	[number] Default: 1.0	
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output ASCII file	`OUTPUT_GENERIC`	[file] Default: `[Skip output]`	Specifies where the output is stored. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output ASCII file	`OUTPUT_GENERIC`	[file]	The file with the output

Python code

Algorithm ID: `lastools:lasinfo`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.12 lasmerge

Merge up to seven LAS/LAZ files into one.

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	CPU64	[boolean] Default: False	
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
files are flightlines	FILES_ARE_FLIGHTLINES	[boolean] Default: False	
apply file source ID	APPLY_FILE_SOURCE_ID	[boolean] Default: False	
input LAS/LAZ file	INPUT_LASLAZ	[file]	The first file containing points to be merged
2nd file Optional	FILE2	[file]	The second file to merge
3rd file Optional	FILE3	[file]	The third file to merge
4th file Optional	FILE4	[file]	The fourth file to merge
5th file Optional	FILE5	[file]	The fifth file to merge
6th file Optional	FILE6	[file]	The sixth file to merge
7th file Optional	FILE7	[file]	The seventh file to merge
additional command line parameter(s) Optional	ADDITIONAL_OPTIONS	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	OUTPUT_LASLAZ	[file] Default: [Skip output]	Specifies where the output point cloud is stored. Use LAZ for compressed output, LAS for uncompressed output, and TXT for ASCII. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output LAS/LAZ file	OUTPUT_LASLAZ	[file]	The output (merged) LAS/LAZ format file

Python code

Algorithm ID: `lastools:lasmerge`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.13 lasprecision

Parameter

Label	Name	Type	Beschreibung
verbose	VERBOSE	[boolean] Default: False	Generates more textual control output to the console
open LAStools GUI	GUI	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	INPUT_LASLAZ	[file]	The file the input point cloud
additional command line parameter(s) Optional	ADDITIONAL_OPTIONS	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output ASCII file	OUTPUT_GENERIC	[file] Default: `[Skip output]`	Specifies where the output ASCII file is stored. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output ASCII file	OUTPUT_GENERIC	[file]	The output ASCII file

Python code

Algorithm ID: `lastools:lasprecision`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.14 lasquery

Beschreibung

<fügen Sie hier die Übersetzung des Algorithmus ein>

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file the input point cloud
area of interest	`AOI`	[extent]	The extent
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.

Ausgaben

Python code

Algorithm ID: `lastools:lasquery`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.15 lasvalidate

Parameter

Label	Name	Type	Beschreibung
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file the input point cloud
save report to ,*_LVS.xml‘	`ONE_REPORT_PER_FILE`	[boolean]	
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ,‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output XML file	`OUTPUT_GENERIC`	[file] Default: `[Skip output]`	Specifies where the output XML file is stored. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output XML file	`OUTPUT_GENERIC`	[file]	The output XML file

Python code

Algorithm ID: `lastools:lasvalidate`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.16 laszip

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with pre-populated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file to be zipped
only report size	`REPORT_SIZE`	[boolean] Default: False	

Fortsetzung auf der nächsten Seite

Tab. 23.181 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
create spatial indexing file (*.lax)	`CREATE_LAX`	[boolean] Default: False	
append *.lax into *.laz file	`APPEND_LAX`	[boolean] Default: False	
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚'	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file] Default: `[Skip output]`	Specifies where the output point cloud is stored. Use LAZ for compressed output, LAS for uncompressed output, and TXT for ASCII. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file]	The output file

Python code

Algorithm ID: `lastools:laszip`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.3.17 txt2las

Parameter

Label	Name	Type	Beschreibung
verbose	`VERBOSE`	[boolean] Default: False	Generates more textual control output to the console
run new 64 bit executable	`CPU64`	[boolean] Default: False	
open LAStools GUI	`GUI`	[boolean] Default: False	Starts the GUI of LAStools with prepopulated input files
input LAS/LAZ file	`INPUT_LASLAZ`	[file]	The file to be zipped
parse lines as	`PARSE`	[string] Default: ‚xyz'	

Fortsetzung auf der nächsten Seite

Tab. 23.182 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
skip the first n lines	SKIP	[number] Default: 0	
resolution of x and y coordinate	SCALE_FACTOR_XY	[number] Default: 0.01	
resolution of z coordinate	SCALE_FACTOR_Z	[number] Default: 0.01	
resolution of z coordinate	SCALE_FACTOR_Z	[number] Default: 0.01	
source projection	PROJECTION	[enumeration] Default: 0	One of: • 0 — — • 1 — epsg • 2 — utm • 3 — sp83 • 4 — sp27 • 5 — longlat • 6 — latlong • 7 — ecef
source epsg code	EPSG_CODE	[number]	

Fortsetzung auf der nächsten Seite

Tab. 23.182 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
utm zone	`UTM`	[enumeration] Default: 0	One of: • 0 — — • 1 — 1 (north) • 2 — 2 (north) • 3 — 3 (north) • 4 — 4 (north) • 5 — 5 (north) • 6 — 6 (north) • 7 — 7 (north) • 8 — 8 (north) • 9 — 9 (north) • 10 — 10 (north) • 11 — 11 (north) • 12 — 12 (north) • 13 — 13 (north) • 14 — 14 (north) • 15 — 15 (north) • 16 — 16 (north) • 17 — 17 (north) • 18 — 18 (north) • 19 — 19 (north) • 20 — 20 (north) • 21 — 21 (north) • 22 — 22 (north) • 23 — 23 (north) • 24 — 24 (north) • 25 — 25 (north) • 26 — 26 (north) • 27 — 27 (north) • 28 — 28 (north) • 29 — 29 (north) • 30 — 30 (north) • 31 — 31 (north) • 32 — 32 (north) • 33 — 33 (north) • 34 — 34 (north) • 35 — 35 (north) • 36 — 36 (north) • 37 — 37 (north) • 38 — 38 (north) • 39 — 39 (north) • 40 — 40 (north) • 41 — 41 (north) • 42 — 42 (north) • 43 — 43 (north) • 44 — 44 (north) • 45 — 45 (north) • 46 — 46 (north) • 47 — 47 (north) • 48 — 48 (north) • 49 — 49 (north) • 50 — 50 (north) • 51 — 51 (north) • 52 — 52 (north) • 53 — 53 (north) • 54 — 54 (north) • 55 — 55 (north) • 56 — 56 (north) • 57 — 57 (north) • 58 — 58 (north) • 59 — 59 (north) • 60 — 60 (north)

Tab. 23.182 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
state plane code	SP	[enumeration] Default: 0	One of: • 0 — — • 1 — AK_10 • 2 — AK_2 • 3 — AK_3 • 4 — AK_4 • 5 — AK_5 • 6 — AK_6 • 7 — AK_7 • 8 — AK_8 • 9 — AK_9 • 10 — AL_E • 11 — AL_W • 12 — AR_N • 13 — AR_S • 14 — AZ_C • 15 — AZ_E • 16 — AZ_W • 17 — CA_I • 18 — CA_II • 19 — CA_III • 20 — CA_IV • 21 — CA_V • 22 — CA_VI • 23 — CA_VII • 24 — CO_C • 25 — CO_N • 26 — CO_S • 27 — CT • 28 — DE • 29 — FL_E • 30 — FL_N • 31 — FL_W • 32 — GA_E • 33 — GA_W • 34 — HI_1 • 35 — HI_2 • 36 — HI_3 • 37 — HI_4 • 38 — HI_5 • 39 — IA_N • 40 — IA_S • 41 — ID_C • 42 — ID_E • 43 — ID_W • 44 — IL_E • 45 — IL_W • 46 — IN_E • 47 — IN_W • 48 — KS_N • 49 — KS_S • 50 — KY_N • 51 — KY_S • 52 — LA_N • 53 — LA_S • 54 — MA_I • 55 — MA_M • 56 — MD • 57 — ME_E • 58 — ME_W • 59 — MI_C • 60 — MI_N • 61 — MI_S

Tab. 23.182 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
additional command line parameter(s) Optional	`ADDITIONAL_OPTIONS`	[string] Default: ‚‘	Specifies other command-line switches not available via this menu but known to the (advanced) LAStools user.
Output LAS/LAZ file	`OUTPUT_LASLAZ`	[file] Default: `[Skip output]`	Specifies where the output point cloud is stored. Use LAZ for compressed output, LAS for uncompressed output, and TXT for ASCII. One of: • Skip Output • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
output LAS/LAZ file	`OUTPUT_LASLAZ`	[file]	The output file

Python code

Algorithm ID: `lastools:txt2las`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.4 TauDEM Algorithmus-Anbieter

TauDEM (Terrain Analysis Using Digital Elevation Models) is a set of Digital Elevation Model (DEM) tools for the extraction and analysis of hydrologic information from topography as represented by a DEM. This is software developed at Utah State University (USU) for hydrologic digital elevation model analysis and watershed delineation.

TauDEM is distributed as a set of standalone command line executable programs for a Windows and source code for compiling and use on other systems.

Bemerkung: Please remember that Processing contains only the interface description, so you need to install TauDEM 5.0.6 by yourself and configure Processing properly.

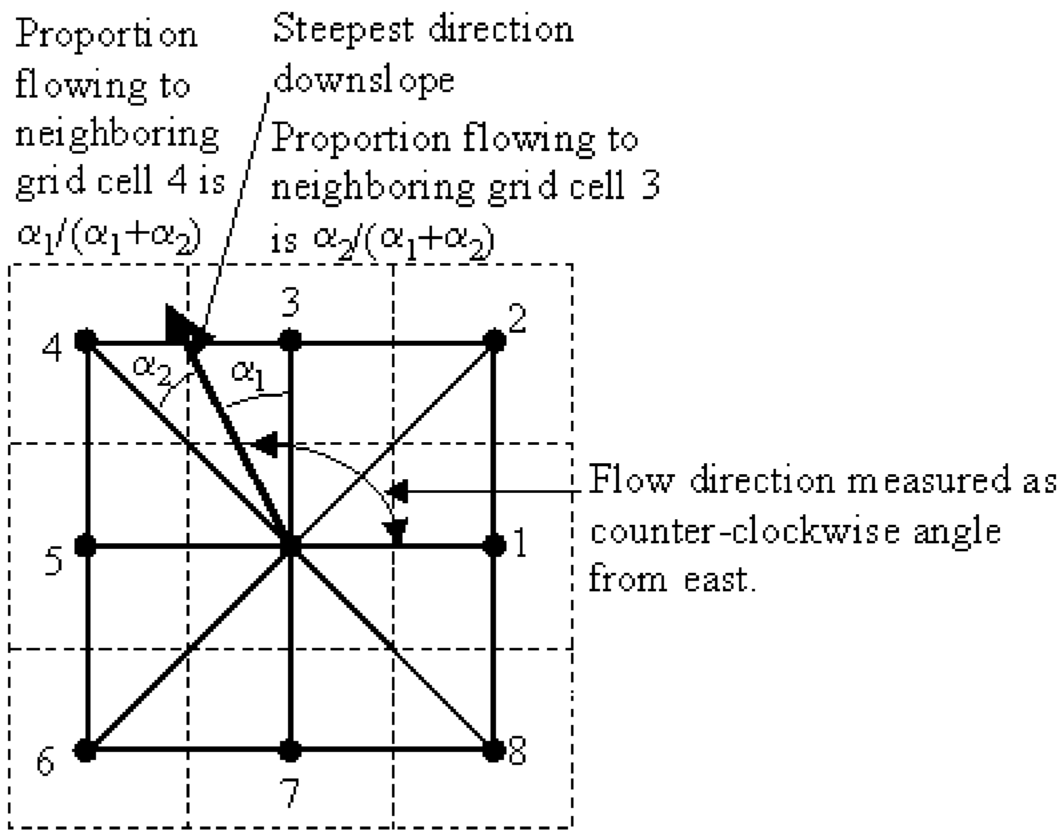
Documentation for TauDEM algorithms derived from official TauDEM documentation

23.4.1 Grundlegende Gitteranalyse

D-Unendlich beitragende Fläche

Beschreibung

Calculates a grid of specific catchment area which is the contributing area per unit contour length using the multiple flow direction D-infinity approach. D-infinity flow direction is defined as steepest downward slope on planar triangular facets on a block centered grid. The contribution at each grid cell is taken as the grid cell length (or when the optional weight grid input is used, from the weight grid). The contributing area of each grid cell is then taken as its own contribution plus the contribution from upslope neighbors that have some fraction draining to it according to the D-infinity flow model. The flow from each cell either all drains to one neighbor, if the angle falls along a cardinal (0, $\pi/2$, π, $3\pi/2$) or ordinal ($\pi/4$, $3\pi/4$, $5\pi/4$, $7\pi/4$) direction, or is on an angle falling between the direct angle to two adjacent neighbors. In the latter case the flow is proportioned between these two neighbor cells according to how close the flow direction angle is to the direct angle to those cells. The contour length used here is the grid cell size. The resulting units of the specific catchment area are length units the same as those of the grid cell size.



When the optional weight grid is not used, the result is reported in terms of specific catchment area, the upslope area per unit contour length, taken here as the number of cells times grid cell length (cell area divided by cell length). This assumes that grid cell length is the effective contour length, in the definition of specific catchment area and does not distinguish any difference in contour length dependent upon the flow direction. When the optional weight grid is used, the result is reported directly as a summation of weights, without any scaling.

If the optional outlet point shapefile is used, only the outlet cells and the cells upslope (by the D-infinity flow model) of them are in the domain to be evaluated.

By default, the tool checks for edge contamination. This is defined as the possibility that a contributing area value may be underestimated due to grid cells outside of the domain not being counted. This occurs when drainage is inwards from the boundaries or areas with „no data" values for elevation. The algorithm recognizes this and reports „no data" for the contributing area. It is common to see streaks of „no data" values extending inwards from boundaries along flow paths that enter the domain at a boundary. This is the desired effect and indicates that contributing area for these grid cells is unknown due to it being dependent on terrain outside of the domain of data available. Edge contamination checking may be turned off in cases where you know it is not an issue or want to ignore these problems, if for example, the DEM has been clipped along a watershed outline.

Parameter

Label	Name	Type	Beschreibung
D-infinity flow directions	`DINF_FLOWDIR`	[raster]	A grid of flow directions based on the D-infinity flow method using the steepest slope of a triangular facet. Flow direction is determined as the direction of the steepest downward slope on the 8 triangular facets of a 3x3 block centered grid. Flow direction is encoded as an angle in radians, counter-clockwise from east as a continuous (floating point) quantity between 0 and 2π. The resulting flow in a grid is then usually interpreted as being proportioned between the two neighboring cells that define the triangular facet with the steepest downward slope.
Outlets Optional	`OUTLETS`	[vector: point]	A point shapefile defining the outlets of interest. If this input file is used, only the cells upslope of these outlet cells are considered to be within the domain being evaluated.
Weight grid Optional	`WEIGHT_GRID`	[raster]	A grid giving contribution to flow for each cell. These contributions (also sometimes referred to as weights or loadings) are used in the contributing area accumulation. If this input file is not used, the result is reported in terms of specific catchment area (the upslope area per unit contour length) taken as the number of cells times grid cell length (cell area divided by cell length).

Fortsetzung auf der nächsten Seite

Tab. 23.183 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Check for edge contamination	`EDGE_CONTAMINATION`	[boolean] Default: True	A flag that indicates whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a contributing area value may be underestimated due to the fact that grid cells outside of the domain have not been evaluated. This occurs when drainage is inwards from the boundaries or areas with NODATA values for elevation. The algorithm recognizes this and reports NODATA for the impated cells. It is common to see streaks of NODATA values extending inwards from boundaries along flow paths that enter the domain at a boundary. This is the desired effect and indicates that contributing area for these grid cells is unknown due to it being dependent on terrain outside of the domain of available data. Edge contamination checking may be turned off in cases where you know this is not an issue, or want to ignore these problems, if for example, the DEM has been clipped along a watershed outline.
D-infinity specific catchment area	`DINF_CONTRIB_AREA`	[raster] Default: `[Save to temporary file]`	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
D-infinity specific catchment area	`DINF_CONTRIB_AREA`	[raster]	A grid of specific catchment area which is the contributing area per unit contour length using the multiple flow direction D-infinity approach. The contributing area of each grid cell is then taken as its own contribution plus the contribution from upslope neighbors that have some fraction draining to it according to the D-infinity flow model.

Algorithm ID: `taudem:areadinf`

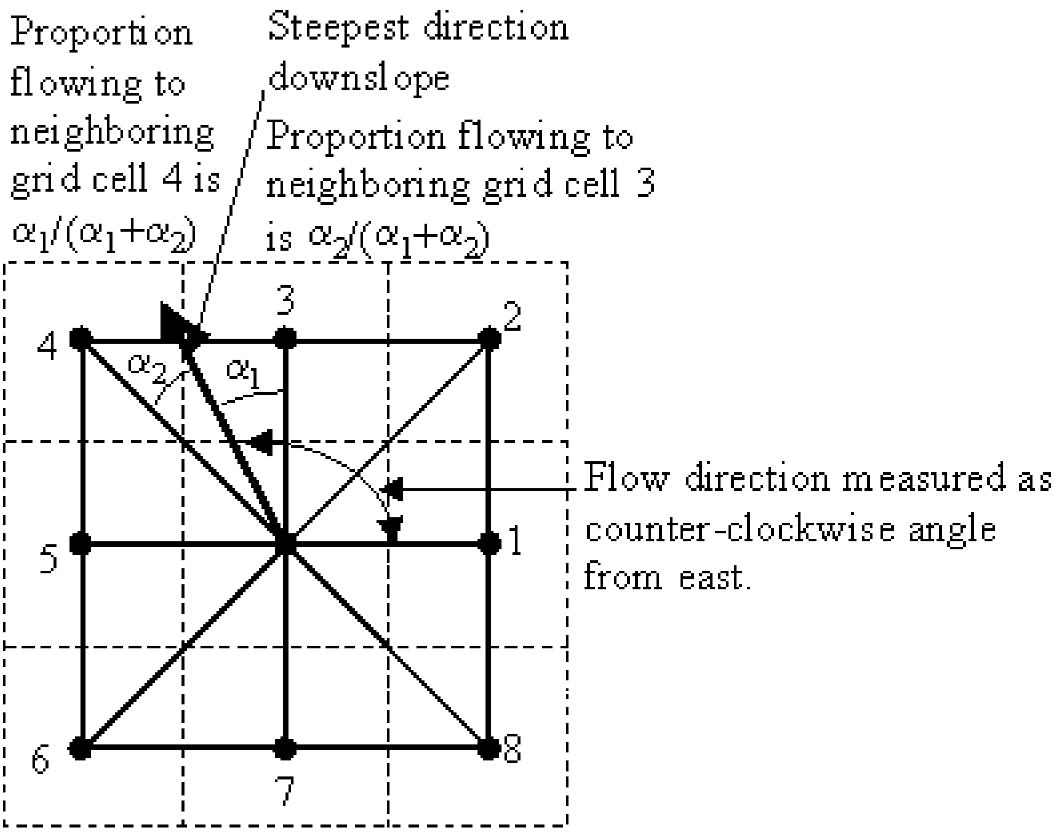
```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity Fließrichtungen

Beschreibung

Assigns a flow direction based on the D-infinity flow method using the steepest slope of a triangular facet (Tarboton, 1997, „A New Method for the Determination of Flow Directions and Contributing Areas in Grid Digital Elevation Models“, Water Resources Research, 33(2): 309-319). Flow direction is defined as steepest downward slope on planar triangular facets on a block centered grid. Flow direction is encoded as an angle in radians counter-clockwise from east as a continuous (floating point) quantity between 0 and 2π. The flow direction angle is determined as the direction of the steepest downward slope on the eight triangular facets formed in a 3 x 3 grid cell window centered on the grid cell of interest. The resulting flow in a grid is then usually interpreted as being proportioned between the two neighboring cells that define the triangular facet with the steepest downward slope.



A block-centered representation is used with each elevation value taken to represent the elevation of the center of the corresponding grid cell. Eight planar triangular facets are formed between each grid cell and its eight neighbors. Each of these has a downslope vector which when drawn outwards from the center may be at an angle that lies within or outside the 45 degree (π/4 radian) angle range of the facet at the center point. If the slope vector angle is within the facet angle, it represents the steepest flow direction on that facet. If the slope vector angle is outside a facet, the steepest flow direction associated with that facet is taken along the steepest edge. The slope and flow direction associated with the grid cell is taken as the magnitude and direction of the steepest downslope vector from all eight facets. Slope is measured as drop/distance, i.e. tan of the slope angle.

In the case where no slope vectors are positive (downslope), the flow direction is set using the method of Garbrecht and Martz (1997) for the determination of flow across flat areas. This makes flat areas drain away from high ground and towards low ground. The flow path grid to enforce drainage along existing streams is an optional input, and if used, takes precedence over elevations for the setting of flow directions.

The D-infinity flow direction algorithm may be applied to a DEM that has not had its pits filled, but it will then result in „no data“ values for the D-infinity flow direction and slope associated with the lowest point of the pit.

Parameter

Label	Name	Type	Beschreibung
Pit filled elevation	`PIT_FILLED`	[raster]	A grid of elevation values. This is usually the output of the **„Pit Remove“** tool, in which case it is elevations with pits removed. Pits are low elevation areas in digital elevation models (DEMs) that are completely surrounded by higher terrain. They are generally taken to be artifacts of the digitation process that interfere with the processing of flow across DEMs. So they are removed by raising their elevation to the point where they just drain off the domain. This step is not essential if you have reason to believe that the pits in your DEM are real. If a few pits actually exist and so should not be removed, while at the same time others are believed to be artifacts that need to be removed, the actual pits should have NODATA elevation values inserted at their lowest point. NODATA values serve to define edges of the domain in the flow field, and elevations are only raised to where flow is off an edge, so an internal NODATA value will stop a pit from being removed, if necessary.
D-infinity flow directions	`DINF_FLOWDIR`	[raster] Default: `[Save to temporary file]`	Specification of the output flow direction raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.
D-infinity slope	`DINF_SLOPE`	[raster] Default: `[Save to temporary file]`	Specification of the output slope raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
D-infinity flow directions	`DINF_FLOWDIR`	[raster]	A grid of flow directions based on the D-infinity flow method using the steepest slope of a triangular facet. Flow direction is determined as the direction of the steepest downward slope on the 8 triangular facets of a 3x3 block centered grid. Flow direction is encoded as an angle in radians, counter-clockwise from east as a continuous (floating point) quantity between 0 and 2π. The resulting flow in a grid is then usually interpreted as being proportioned between the two neighboring cells that define the triangular facet with the steepest downward slope.
D-infinity slope	`DINF_SLOPE`	[raster]	A grid of slope evaluated using the D-infinity method described in Tarboton, D. G., (1997), „A New Method for the Determination of Flow Directions and Contributing Areas in Grid Digital Elevation Models“, Water Resources Research, 33(2): 309-319. This is the steepest outwards slope on one of eight triangular facets centered at each grid cell, measured as drop/distance, i.e. tan of the slope angle.

Algorithm ID: `taudem:dinfflowdir`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D8 beitragende Fläche

Beschreibung

Calculates a grid of contributing areas using the single direction D8 flow model. The contribution of each grid cell is taken as one (or when the optional weight grid is used, the value from the weight grid). The contributing area for each grid cell is taken as its own contribution plus the contribution from upslope neighbors that drain in to it according to the D8 flow model.

If the optional outlet point shapefile is used, only the outlet cells and the cells upslope (by the D8 flow model) of them are in the domain to be evaluated.

By default, the tool checks for edge contamination. This is defined as the possibility that a contributing area value may be underestimated due to grid cells outside of the domain not being counted. This occurs when drainage is inwards from the boundaries or areas with „no data“ values for elevation. The algorithm recognizes this and reports „no data“ for the contributing area. It is common to see streaks of „no data“ values extending inwards from boundaries along flow paths that enter the domain at a boundary. This is the desired effect and indicates that contributing area for these grid cells is unknown due to it being dependent on terrain outside of the domain of data available. Edge contamination checking may be turned off in cases where you know this is not an issue or want to ignore these problems, if for example, the DEM has been clipped along a watershed outline.

Parameter

Label	Name	Type	Beschreibung
D8 flow directions	`D8_FLOWDIR`	[raster]	A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.
Outlets Optional	`OUTLETS`	[vector: point]	A point shapefile defining the outlets of interest. If this input file is used, only the cells upslope of these outlet cells are considered to be within the domain being evaluated.
Weight grid Optional	`WEIGHT_GRID`	[raster]	A grid giving contribution to flow for each cell. These contributions (also sometimes referred to as weights or loadings) are used in the contributing area accumulation. If this input file is not used, the contribution to flow will assumed to be one for each grid cell.
Check for edge contamination	`EDGE_CONTAMINATION`	[boolean] Default: True	A flag that indicates whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a contributing area value may be underestimated due to the fact that grid cells outside of the domain have not been evaluated. This occurs when drainage is inwards from the boundaries or areas with NODATA values for elevation. The algorithm recognizes this and reports NODATA for the impated cells. It is common to see streaks of NODATA values extending inwards from boundaries along flow paths that enter the domain at a boundary. This is the desired effect and indicates that contributing area for these grid cells is unknown due to it being dependent on terrain outside of the domain of available data. Edge contamination checking may be turned off in cases where you know this is not an issue, or want to ignore these problems, if for example, the DEM has been clipped along a watershed outline.
D8 specific catchment area	`D8_CONTRIB_AREA`	[raster] Default: `[Save to temporary file]`	Specification of the output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
D8 specific catchment area	`D8_CONTRIB_AREA`	[raster]	A grid of contributing area values calculated as the cells own contribution plus the contribution from upslope neighbors that drain in to it according to the D8 flow model.

Algorithm ID: `taudem:aread8`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

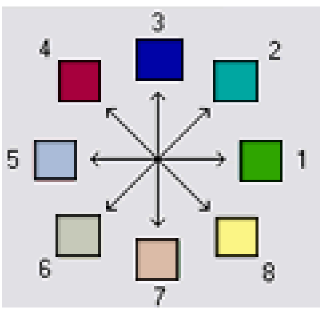
D8 Flußrichtungen

Beschreibung

Creates 2 grids. The first contains the flow direction from each grid cell to one of its adjacent or diagonal neighbors, calculated using the direction of steepest descent. The second contain the slope, as evaluated in the direction of steepest descent, and is reported as drop/distance, i.e. tan of the angle. Flow direction is reported as NODATA for any grid cell adjacent to the edge of the DEM domain, or adjacent to a NODATA value in the DEM. In flat areas, flow directions are assigned away from higher ground and towards lower ground using the method of Garbrecht and Martz (1997). The D8 flow direction algorithm may be applied to a DEM that has not had its pits filled, but it will then result in NODATA values for flow direction and slope at the lowest point of each pit.

D8 Fliessrichtung Raster

- 1 — Ost
- 2 — Nordost
- 3 — Nord
- 4 — Nordwest
- 5 — West
- 6 — Südwest
- 7 — Süd
- 8 — Südost



The flow direction routing across flat areas is performed according to the method described by Garbrecht, J. and L. W. Martz, (1997), „The Assignment of Drainage Direction Over Flat Surfaces in Raster Digital Elevation Models“, Journal of Hydrology, 193: 204-213.

Parameter

Label	Name	Type	Beschreibung
Pit filled elevation	PIT_FILLED	[raster]	A grid of elevation values. This is usually the output of the **„Pit Remove“** tool, in which case it is elevations with pits removed. Pits are low elevation areas in digital elevation models (DEMs) that are completely surrounded by higher terrain. They are generally taken to be artifacts of the digitation process that interfere with the processing of flow across DEMs. So they are removed by raising their elevation to the point where they just drain off the domain. This step is not essential if you have reason to believe that the pits in your DEM are real. If a few pits actually exist and so should not be removed, while at the same time others are believed to be artifacts that need to be removed, the actual pits should have NODATA elevation values inserted at their lowest point. NODATA values serve to define edges of the domain in the flow field, and elevations are only raised to where flow is off an edge, so an internal NODATA value will stop a pit from being removed, if necessary.
D8 flow directions	D8_FLOWDIR	[raster] Default: [Save to temporary file]	Specification of the output flow direction raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.
D8 slope	D8_SLOPE	[raster] Default: [Save to temporary file]	Specification of the output slope raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
D8 flow directions	D8_FLOWDIR	[raster]	A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope.
D8 slope	D8_SLOPE	[raster]	A grid giving slope in the D8 flow direction. This is measured as drop/distance.

Algorithm ID: `taudem:d8flowdir`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Gitternetzwerk

Beschreibung

Creates 3 grids that contain for each grid cell: 1) the longest path, 2) the total path, and 3) the Strahler order number. These values are derived from the network defined by the D8 flow model.

The longest upslope length is the length of the flow path from the furthest cell that drains to each cell. The total upslope path length is the length of the entire grid network upslope of each grid cell. Lengths are measured between cell centers taking into account cell size and whether the direction is adjacent or diagonal.

Strahler order is defined as follows: A network of flow paths is defined by the D8 Flow Direction grid. Source flow paths have a Strahler order number of one. When two flow paths of different order join the order of the downstream flow path is the order of the highest incoming flow path. When two flow paths of equal order join the downstream flow path order is increased by 1. When more than two flow paths join the downstream flow path order is calculated as the maximum of the highest incoming flow path order or the second highest incoming flow path order + 1. This generalizes the common definition to cases where more than two flow paths join at a point.

Where the optional mask grid and threshold value are input, the function is evaluated only considering grid cells that lie in the domain with mask grid value greater than or equal to the threshold value. Source (first order) grid cells are taken as those that do not have any other grid cells from inside the domain draining in to them, and only when two of these flow paths join is order propagated according to the ordering rules. Lengths are also only evaluated counting paths within the domain greater than or equal to the threshold.

If the optional outlet point shapefile is used, only the outlet cells and the cells upslope (by the D8 flow model) of them are in the domain to be evaluated.

Parameter

Label	Name	Type	Beschreibung
D8 flow directions	`D8_FLOWDIR`	[raster]	A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.
Mask Grid Optional	`MASK_GRID`	[raster]	A grid that is used to determine the domain do be analyzed. If the mask grid value >= mask threshold (see below), then the cell will be included in the domain. While this tool does not have an edge contamination flag, if edge contamination analysis is needed, then a mask grid from a function like **„D8 Contributing Area“** that does support edge contamination can be used to achieve the same result.
Mask threshold Optional	`THRESHOLD`	[number] Default: 100.0	This input parameter is used in the calculation mask grid value >= mask threshold to determine if the grid cell is in the domain to be analyzed.
Outlets Optional	`OUTLETS`	[vector: point]	A point shapefile defining the outlets of interest. If this input file is used, only the cells upslope of these outlet cells are considered to be within the domain being evaluated.

Fortsetzung auf der nächsten Seite

Tab. 23.188 – Fortsetzung der vorherigen Seite

Label	Name	Type	Beschreibung
Longest upslope length	LONGEST_PATH	[raster] Default: [Save to temporary file]	Specification of the output raster with total upslope lengths. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.
Total upslope length	TOTAL_PATH	[raster] Default: [Save to temporary file]	Specification of the output raster with upslope lengths. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.
Strahler network order	STRAHLER_ORDER	[raster] Default: [Save to temporary file]	Specification of the output raster with Strahler network order. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Longest upslope length	LONGEST_PATH	[raster]	A grid that gives the length of the longest upslope D8 flow path terminating at each grid cell. Lengths are measured between cell centers taking into account cell size and whether the direction is adjacent or diagonal.
Total upslope length	TOTAL_PATH	[raster]	The total upslope path length is the length of the entire D8 flow grid network upslope of each grid cell. Lengths are measured between cell centers taking into account cell size and whether the direction is adjacent or diagonal.
Strahler network order	STRAHLER_ORDER	[raster]	A grid giving the Strahler order number for each cell. A network of flow paths is defined by the D8 Flow Direction grid. Source flow paths have a Strahler order number of one. When two flow paths of different order join the order of the downstream flow path is the order of the highest incoming flow path. When two flow paths of equal order join the downstream flow path order is increased by 1. When more than two flow paths join the downstream flow path order is calculated as the maximum of the highest incoming flow path order or the second highest incoming flow path order + 1. This generalizes the common definition to cases where more than two flow paths join at a point.

Algorithm ID: `taudem:gridnet`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Löcher entfernen

Beschreibung

Identifies all pits in the DEM and raises their elevation to the level of the lowest pour point around their edge. Pits are low elevation areas in digital elevation models (DEMs) that are completely surrounded by higher terrain. They are generally taken to be artifacts that interfere with the routing of flow across DEMs, so are removed by raising their elevation to the point where they drain off the edge of the domain. The pour point is the lowest point on the boundary of the „watershed" draining to the pit. This step is not essential if you have reason to believe that the pits in your DEM are real. If a few pits actually exist and so should not be removed, while at the same time others are believed to be artifacts that need to be removed, the actual pits should have NODATA elevation values inserted at their lowest point. NODATA values serve to define edges in the domain, and elevations are only raised to where flow is off an edge, so an internal NODATA value will stop a pit from being removed, if necessary.

Parameter

Label	Name	Type	Beschreibung
Elevation	`ELEVATION`	[raster]	A digital elevation model (DEM) grid to serve as the base input for the terrain analysis and stream delineation.
Depression mask Optional	`DEPRESSION_MASK`	[raster]	
Consider only 4 way neighbors	`FOUR_NEIGHBOURS`	[boolean] Default: False	
Pit removed elevation	`PIT_FILLED`	[raster] Default: `[Save to temporary file]`	Specification of the (pit filled) output raster. One of: • Save to a Temporary File • Save to File… The file encoding can also be changed here.

Ausgaben

Label	Name	Type	Beschreibung
Pit removed elevation	`PIT_FILLED`	[raster]	A grid of elevation values with pits removed so that flow is routed off of the domain.

Algorithm ID: `taudem:pitremove`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.4.2 Spezialisierte Gitteranalyse

D8 Abstand zu Strömen

Beschreibung

Computes the horizontal distance to stream for each grid cell, moving downslope according to the D8 flow model, until a stream grid cell is encountered.

Parameter

`D8 Fließrichtung Raster` [raster] This input is a grid of flow directions that are encoded using the D8 method where all flow from a cells goes to a single neighboring cell in the direction of steepest descent. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.

`Stromrastergitter` [raster] A grid indicating streams. Such a grid can be created by several of the tools in the **„Stream Network Analysis“** toolset. However, the tools in the **„Stream Network Analysis“** toolset only create grids with a value of 0 for no stream, or 1 for stream cells. This tool can also accept grids with values greater than 1, which can be used in conjunction with the `Threshold` parameter to determine the location of streams. This allows Contributing Area grids to be used to define streams as well as the normal Stream Raster grids. This grid expects integer (long integer) values and any non-integer values will be truncated to an integer before being evaluated.

`Schwellen` [number] This value acts as threshold on the `Stream Raster Grid` to determine the location of streams. Cells with a `Stream Raster Grid` value greater than or equal to the `Threshold` value are interpreted as streams.

Vorgabe: *50*

Ausgaben

`Output Distance to Streams` [raster] A grid giving the horizontal distance along the flow path as defined by the D8 Flow Directions Grid to the streams in the Stream Raster Grid.

Algorithm ID: `taudem:d8hdisttostrm`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Unendlich Lawinenauslauf

Beschreibung

Identifies an avalanche's affected area and the flow path length to each cell in that affacted area. All cells downslope from each source area cell, up to the point where the slope from the source to the affected area is less than a threshold angle called the Alpha Angle can be in the affected area. This tool uses the D-infinity multiple flow direction method for determining flow direction. This will likely cause very small amounts of flow to be dispersed to some downslope cells that might overstate the affected area, so a threshold proportion can be set to avoid this excess dispersion. The flow path length is the distance from the cell in question to the source cell that has the highest angle.

All points downslope from the source area are potentially in the affected area, but not beyond a point where the slope from the source to the affected area is less than a threshold angle called the Alpha Angle.

Slope is to be measured using the straight line distance from source point to evaluation point.

Elevations

10	10	10	10	10	10
10	9	9	9	9	10
10	9	8	7	6.99	10
10	9	9	8	6.98	10
10	9	8	7	6.97	10
10	10	10	10	6.96	10

Yellow cell is the source
Green: downslope of source

Straight-line distance from highest point of source

0	1	2	3	4	5
1	1.414214	2.236068	3.162278	4.123106	5.09902
2	2.236068	2.828427	3.605551	4.472136	5.385165
3	3.162278	3.605551	4.242641	5	5.830952
4	4.123106	4.472136	5	5.656854	6.403124
5	5.09902	5.385165	5.830952	6.403124	7.071068

Yellow cell is the source
Green: downslope of source

Drop in elevation from highest point in source

0	0	0	0	0	0
0	1	1	1	1	0
0	1	2	3	3.01	0
0	1	1	2	3.02	0
0	1	2	3	3.03	0
0	0	0	0	3.04	0

Yellow cell is the source
Green: downslope of source

2 The cell size (a fiddle factor for me to make sensible values)
18 The threshold angle for being in the runout zone

The slope angle from the highest point in the source to each cell

0	0	0	0	0	0
0	19	13	9	7	0
0	13	19	23	19	0
0	9	8	13	17	0
0	7	13	17	15	0
0	0	0	0	13	0

Yellow cell is the source
Green: downslope of source
Grey cells are BOTH downslope of the source AND have a sufficiently steep angle to be in the runout zone

It makes more physical sense to me for the angle to be measured along the flow path. Nevertheless it is equally easy to code straight line angles as angles along the flow path, so an option that allows switching will be provided. The most practical way to evaluate avalanche runout is to keep track of the source point with the greatest angle to each point. Then the recursive upslope flow algebra approach will look at a grid cell and all its upslope neighbors that flow to it. Information from the upslope neighbors will be used to calculate the angle to the grid cell in question and retain it in the runout zone if the angle exceeds the alpha angle. This procedure makes the assumption that the maximum angle at a grid cell will be from the set of cells that have maximum angles to the inflowing neighbors. This will always be true of angle is calculated along a flow path, but I can conceive of cases where flow paths bend back on themselves where this would not be the case for straight line angles.

The D-infinity multiple flow direction field assigns flow from each grid cell to multiple downslope neighbors using proportions (`Pik`) that vary between 0 and 1 and sum to 1 for all flows out of a grid cell. It may be desirable to specify a threshold `T` that this proportion has to exceed before a grid cell is counted as flowing to a downslope grid cell, e.g. `Pik > T` (=0.2 say) to avoid dispersion to grid cells that get very little flow. `T` will be specified as a user input. If all upslope grid cells are to be used `T` may be input as 0.

Avalanche source sites are to be input as a short integer grid (name suffix `*ass`, e.g. `demass`) comprised of positive values where avalanches may be triggered and 0 values elsewhere.

The following grids are output:

- rz — A runout zone indicator with value 0 to indicate that this grid cell is not in the runout zone and value > 0 to indicate that this grid cell is in the runout zone. Since there may be information in the angle to the associated source site, this variable will be assigned the angle to the source site (in degrees)
- dm — Along flow distance from the source site that has the highest angle to the point in question

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions“**.

`Höhengitter mit gefüllten Löchern` [raster] This input is a grid of elevation values. As a general rule, it is recommended that you use a grid of elevation values that have had the pits removed for this input. Pits are generally taken to be artifacts that interfere with the analysis of flow across them. This grid can be obtained as the output of the **„Pit Remove“** tool, in which case it contains elevation values where the pits have been filled to the point where they just drain.

`Avalanche Source Site Grid` [raster] This is a grid of source areas for snow avalanches that are commonly identified manually using a mix of experience and visual interpretation of maps. Avalanche source sites are to be input as a short integer grid (name suffix `*ass`, e.g. `demass`) comprised of positive values where avalanches may be triggered and 0 values elsewhere.

`Proportion Threshold` [number] This value is a threshold proportion that is used to limit the dispersion of flow caused by using the D-infinity multiple flow direction method for determining flow direction. The D-infinity multiple flow direction method often causes very small amounts of flow to be dispersed to some downslope cells that might overstate the affected area, so a threshold proportion can be set to avoid this excess dispersion.

Vorgabe: *0.2*

`Alpha Angle Threshold` [number] This value is the threshold angle, called the Alpha Angle, that is used to determine which of the cells downslope from the source cells are in the affected area. Only the cells downslope from each source area cell, up to the point where the slope from the source to the affected area is less than a threshold angle are in the affected area.

Vorgabe: *18*

`Measure distance along flow path` [boolean] This option selects the method used to measure the distance used to calculate the slope angle. If option is *True* then measure it along the flow path, where the *False* option causes the slope to be measure along the straight line distance from the source cell to the evaluation cell.

Vorgabe: *True*

Ausgaben

`Runout Zone Grid` [raster] This grid Identifies the avalanche's runout zone (affected area) using a runout zone indicator with value 0 to indicate that this grid cell is not in the runout zone and value > 0 to indicate that this grid cell is in the runout zone. Since there may be information in the angle to the associated source site, this variable will be assigned the angle to the source site (in degrees).

`Path Distance Grid` [raster] This is a grid of the flow distance from the source site that has the highest angle to each cell.

Algorithm ID: `taudem:dinfavalanche`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Unendlich konzentrationsbegrenzte Akkumulation

Beschreibung

This function applies to the situation where an unlimited supply of a substance is loaded into flow at a concentration or solubility threshold Csol over a region indicated by an indicator grid (dg). It a grid of the concentration of a substance at each location in the domain, where the supply of substance from a supply area is loaded into the flow at a concentration or solubility threshold. The flow is first calculated as a D-infinity weighted contributing area of an input Effective Runoff Weight Grid (notionally excess precipitation). The concentration of substance over the supply area (indicator grid) is at the concentration threshold. As the substance moves downslope with the D-infinity flow field, it is subject to first order decay in moving from cell to cell as well as dilution due to changes in flow. The decay multiplier grid gives the fractional (first order) reduction in quantity in moving from grid cell `x` to the next downslope cell. If the outlets shapefile is used, the tool only evaluates the part of the domain that contributes flow to the locations given by the shapefile. This is useful for a tracking a contaminant or compound from an area with unlimited supply of that compound that is loaded into a flow at a concentration or solubility threshold over a zone and flow from the zone may be subject to decay or attenuation.

The indicator grid (`dg`) is used to delineate the area of the substance supply using the (0, 1) indicator function `i(x)`. `A[]` denotes the weighted accumulation operator evaluated using the D-Infinity Contributing Area function. The Effective Runoff Weight Grid gives the supply to the flow (e.g. the excess rainfall if this is overland flow) denoted as `w(x)`. The specific discharge is then given by:

```
Q(x)=A[w(x)]
```

This weighted accumulation `Q(x)` is output as the Overland Flow Specific Discharge Grid. Over the substance supply area concentration is at the threshold (the threshold is a saturation or solubility limit). If `i(x) = 1`, then

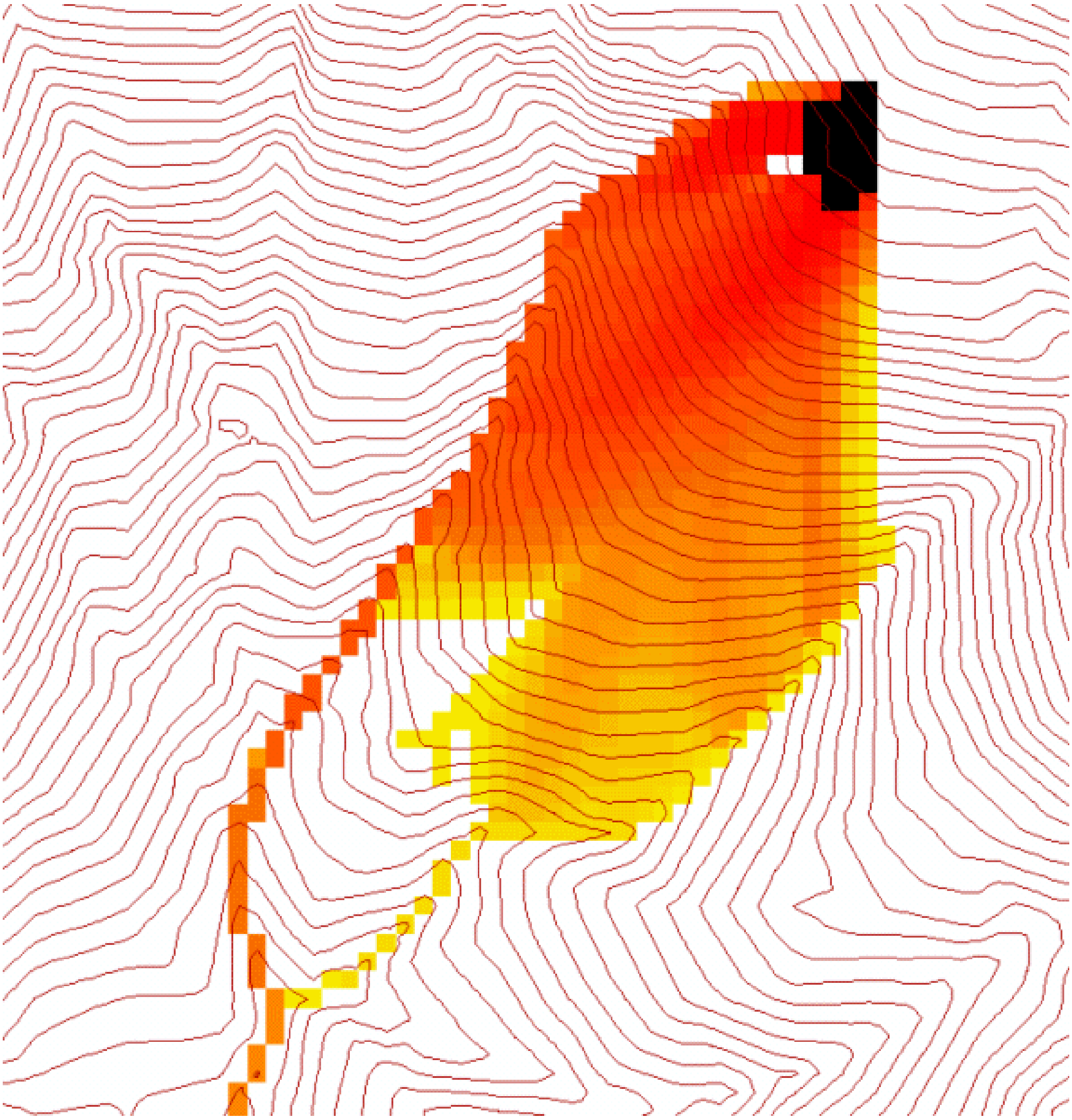
```
C(x) = Csol, and L(x) = Csol Q(x),
```

where `L(x)` denotes the load being carried by the flow. At remaining locations, the load is determined by load accumulation and the concentration by dilution:

$$L(x) = L(i, j) = \sum_{\text{k contributing neighbors}} p_k d(i_k, j_k) L(i_k, j_k)$$

$$C(x) = L(x)/Q(x)$$

Here `d(x) = d(i, j)` is a decay multiplier giving the fractional (first order) reduction in mass in moving from grid cell `x` to the next downslope cell. If travel (or residence) times `t(x)` associated with flow between cells are available `d(x)` may be evaluated as `exp(-k t(x))` where `k` is a first order decay parameter. The Concentration grid output is `C(x)`. If the outlets shapefile is used, the tool only evaluates the part of the domain that contributes flow to the locations given by the shapefile.



Useful for a tracking a contaminant released or partitioned to flow at a fixed threshold concentration.

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This grid can be created by the function **„D-Infinity Flow Directions“**.

`Disturbance Indicator Grid` [raster] A grid that indicates the source zone of the area of substance supply and must be 1 inside the zone and 0 or NODATA over the rest of the domain.

`Decay Multiplier Grid` [raster] A grid giving the factor by which flow leaving each grid cell is multiplied before accumulation on downslope grid cells. This may be used to simulate the movement of an attenuating or decaying substance. If travel (or residence) times `t(x)` associated with flow between cells are available `d(x)` may be evaluated as `exp(-k t(x))` where `k` is a first order decay parameter.

`Effective Runoff Weight Grid` [raster] A grid giving the input quantity (notionally effective runoff or excess precipitation) to be used in the D-infinity weighted contributing area evaluation of Overland Flow Specific Discharge.

`Auslaß-Shapedatei` [vector: point] Optional

This optional input is a point shapefile defining outlets of interest. If this file is used, the tool will only evaluate the area upslope of these outlets.

`Concentration Threshold` [number] The concentration or solubility threshold. Over the substance supply area, concentration is at this threshold.

Vorgabe: *1.0*

`Kantenverunreinigung prüfen` [boolean] This option determines whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being considered when determining contributing area.

Vorgabe: *True*

Ausgaben

`Concentration Grid` [raster] A grid giving the resulting concentration of the compound of interest in the flow.

Algorithm ID: `taudem:dinfconclimaccum`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Unendlich abnehmende Akkumulation

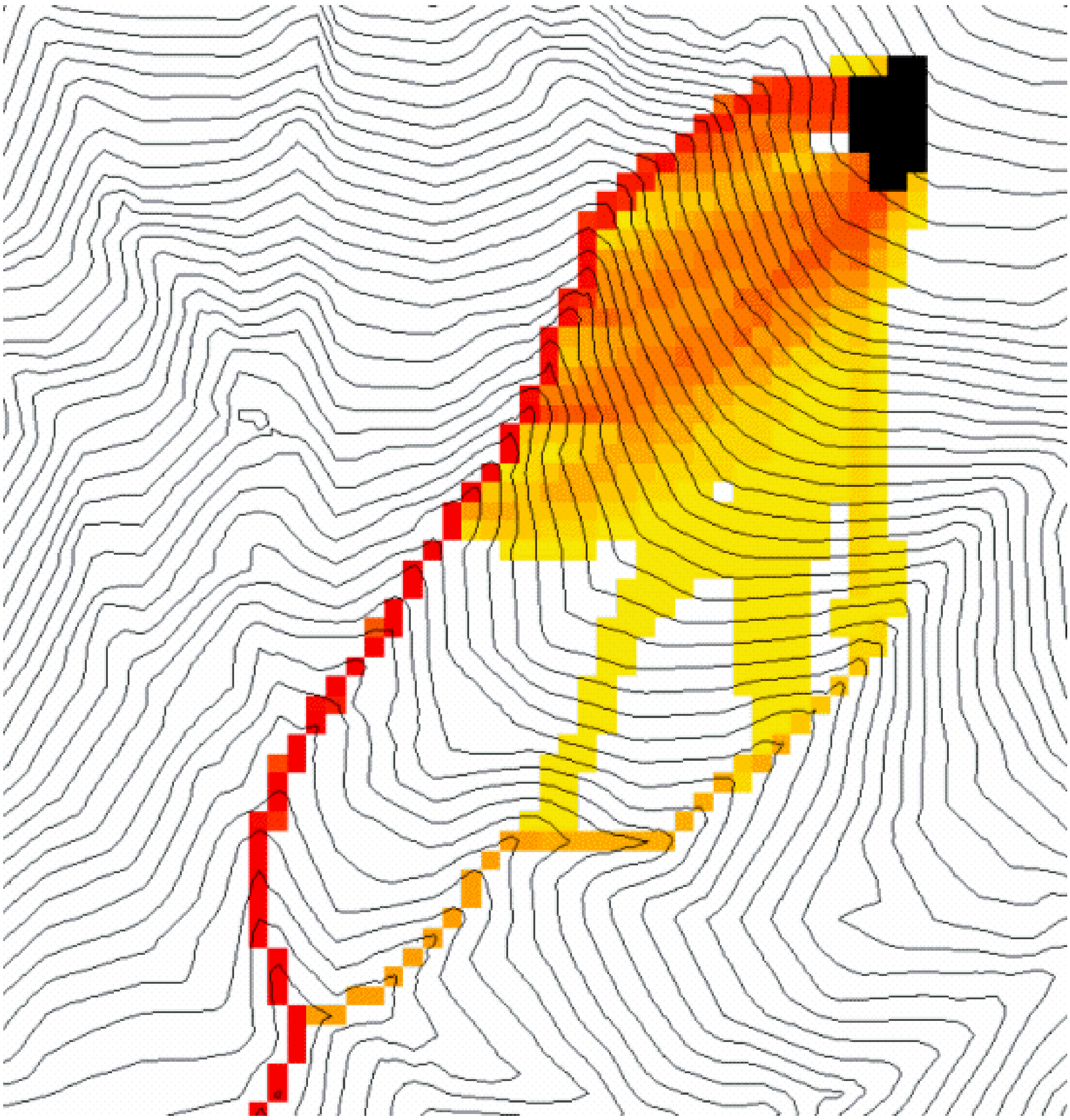
Beschreibung

The D-Infinity Decaying Accumulation tool creates a grid of the accumulated quantity at each location in the domain where the quantity accumulates with the D-infinity flow field, but is subject to first order decay in moving from cell to cell. By default, the quantity contribution of each grid cell is the cell length to give a per unit width accumulation, but can optionally be expressed with a weight grid. The decay multiplier grid gives the fractional (first order) reduction in quantity in accumulating from grid cell `x` to the next downslope cell.

A decayed accumulation operator `DA[.]` takes as input a mass loading field `m(x)` expressed at each grid location as `m(i, j)` that is assumed to move with the flow field but is subject to first order decay in moving from cell to cell. The output is the accumulated mass at each location `DA(x)`. The accumulation of `m` at each grid cell can be numerically evaluated.

$$DA[m(x)] = DA(i, j) = m(i, j)\,\Delta^2 + \sum_{k \text{ contributing neighbors}} p_k d(i_k, j_k) DA(i_k, j_k)$$

Here `d(x) = d(i ,j)` is a decay multiplier giving the fractional (first order) reduction in mass in moving from grid cell `x` to the next downslope cell. If travel (or residence) times `t(x)` associated with flow between cells are available `d(x)` may be evaluated as `exp(-k t(x))` where `k` is a first order decay parameter. The weight grid is used to represent the mass loading `m(x)`. If not specified this is taken as 1. If the outlets shapefile is used the function is only evaluated on that part of the domain that contributes flow to the locations given by the shapefile.



Useful for a tracking contaminant or compound subject to decay or attenuation.

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This grid can be created by the function **„D-Infinity Flow Directions“**.

`Decay Multiplier Grid` [raster] A grid giving the factor by which flow leaving each grid cell is multiplied before accumulation on downslope grid cells. This may be used to simulate the movement of an attenuating substance.

`Weight Grid` [raster] Optional

A grid giving weights (loadings) to be used in the accumulation. If this optional grid is not specified, weights are taken as the linear grid cell size to give a per unit width accumulation.

`Auslaß-Shapedatei` [vector: point] Optional

This optional input is a point shapefile defining outlets of interest. If this file is used, the tool will only evaluate the area upslope of these outlets.

`Kantenverunreinigung prüfen` [boolean] This option determines whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being considered when determining contributing area.

Vorgabe: *True*

Ausgaben

`Decayed Specific Catchment Area Grid` [raster] The D-Infinity Decaying Accumulation tool creates a grid of the accumulated mass at each location in the domain where mass moves with the D-infinity flow field, but is subject to first order decay in moving from cell to cell.

Algorithm ID: `taudem:dinfdecayaccum`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity Entfernung nach unten

Beschreibung

Calculates the distance downslope to a stream using the D-infinity flow model. The D-infinity flow model is a multiple flow direction model, because the outflow from each grid cell is proportioned between up to 2 downslope grid cells. As such, the distance from any grid cell to a stream is not uniquely defined. Flow that originates at a particular grid cell may enter the stream at a number of different cells. The statistical method may be selected as the longest, shortest or weighted average of the flow path distance to the stream. Also one of several ways of measuring distance may be selected: the total straight line path (Pythagoras), the horizontal component of the straight line path, the vertical component of the straight line path, or the total surface flow path.

Parameter

`D-Infinity Fliessrichtung Raster` **[raster]** A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions“**.

`Höhengitter mit gefüllten Löchern` **[raster]** This input is a grid of elevation values. As a general rule, it is recommended that you use a grid of elevation values that have had the pits removed for this input. Pits are generally taken to be artifacts that interfere with the analysis of flow across them. This grid can be obtained as the output of the **„Pit Remove“** tool, in which case it contains elevation values where the pits have been filled to the point where they just drain.

`Stromrastergitter` **[raster]** A grid indicating streams, by using a grid cell value of 1 on streams and 0 off streams. This is usually the output of one of the tools in the **„Stream Network Analysis“** toolset.

`Pfadgewichtungsgitter` **[raster]** Optional

A grid giving weights (loadings) to be used in the distance calculation. This might be used for example where only flow distance through a buffer is to be calculated. The weight is then 1 in the buffer and 0 outside it. Alternatively the weight may reflect some sort of cost function for travel over the surface, perhaps representing travel time or attenuation of a process. If this input file is not used, the loadings will assumed to be one for each grid cell.

`Statistical Method` **[enumeration]** Statistical method used to calculate the distance down to the stream. In the D-Infinity flow model, the outflow from each grid cell is proportioned between two downslope grid cells. Therefore, the distance from any grid cell to a stream is not uniquely defined. Flow that originates at a particular grid cell may enter the stream at a number of cells. The distance to the stream may be defined as the longest (maximum), shortest (minimum) or weighted average of the distance down to the stream.

Optionen:

- 0 — Minimum
- 1 — Maximum
- 2 — Durchschnitt

Vorgabe: *2*

`Distance Method` **[enumeration]** Distance method used to calculate the distance down to the stream. One of several ways of measuring distance may be selected: the total straight line path (Pythagoras), the horizontal component of the straight line path (horizontal), the vertical component of the straight line path (vertical), or the total surface flow path (surface).

Optionen:

- 0 — Pythagoras
- 1 — Horizontal
- 2 — Vertikal
- 3 — Oberfläche

Vorgabe: *1*

`Kantenverunreinigung prüfen` **[boolean]** A flag that determines whether the tool should check for edge contamination. This is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being counted. In the context of Distance Down this occurs when part of a flow path traced downslope from a grid cell leaves the domain without reaching a stream grid cell. With edge contamination checking selected, the algorithm recognizes this and reports no data for the result. This is the desired effect and indicates that values for these grid cells is unknown due to it being dependent on terrain outside of the domain of data available. Edge contamination checking may be overridden in cases where you know this is not an issue or want to evaluate the distance using only the fraction of flow paths that terminate at a stream.

Vorgabe: *True*

Ausgaben

`D-Infinity Abstand nach unten` [raster] Grid containing the distance to stream calculated using the D-infinity flow model and the statistical and path methods chosen.

Algorithm ID: `taudem:dinfdistdown`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity Entfernung hoch

Beschreibung

This tool calculates the distance from each grid cell up to the ridge cells along the reverse D-infinity flow directions. Ridge cells are defined to be grid cells that have no contribution from grid cells further upslope. Given the convergence of multiple flow paths at any grid cell, any given grid cell can have multiple upslope ridge cells. There are three statistical methods that this tool can use: maximum distance, minimum distance and waited flow average over these flow paths. A variant on the above is to consider only grid cells that contribute flow with a proportion greater than a user specified threshold (t) to be considered as upslope of any given grid cell. Setting t=0.5 would result in only one flow path from any grid cell and would give the result equivalent to a D8 flow model, rather than D-infinity flow model, where flow is proportioned between two downslope grid cells. Finally there are several different optional paths that can be measured: the total straight line path (Pythagoras), the horizontal component of the straight line path, the vertical component of the straight line path, or the total surface flow path.

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions“**.

`Höhengitter mit gefüllten Löchern` [raster] This input is a grid of elevation values. As a general rule, it is recommended that you use a grid of elevation values that have had the pits removed for this input. Pits are generally taken to be artifacts that interfere with the analysis of flow across them. This grid can be obtained as the output of the **„Pit Remove“** tool, in which case it contains elevation values where the pits have been filled to the point where they just drain.

`Neigungsgitter` [raster] This input is a grid of slope values. This is measured as drop/distance and it is most often obtained as the output of the **„D-Infinity Flow Directions“** tool.

`Statistical Method` [enumeration] Statistical method used to calculate the distance down to the stream. In the D-Infinity flow model, the outflow from each grid cell is proportioned between two downslope grid cells. Therefore, the distance from any grid cell to a stream is not uniquely defined. Flow that originates at a particular grid cell may enter the stream at a number of cells. The distance to the stream may be defined as the longest (maximum), shortest (minimum) or weighted average of the distance down to the stream.

Optionen:

- 0 — Minimum
- 1 — Maximum
- 2 — Durchschnitt

Vorgabe: *2*

`Distance Method` [enumeration] Distance method used to calculate the distance down to the stream. One of several ways of measuring distance may be selected: the total straight line path (Pythagoras), the horizontal component of the straight line path (horizontal), the vertical component of the straight line path (vertical), or the total surface flow path (surface).

Optionen:

- 0 — Pythagoras
- 1 — Horizontal
- 2 — Vertikal
- 3 — Oberfläche

Vorgabe: *1*

`Proportion Threshold` [number] The proportion threshold parameter where only grid cells that contribute flow with a proportion greater than this user specified threshold (`t`) is considered to be upslope of any given grid cell. Setting `t=0.5` would result in only one flow path from any grid cell and would give the result equivalent to a D8 flow model, rather than D-Infinity flow model, where flow is proportioned between two downslope grid cells.

Vorgabe: *0.5*

`Kantenverunreinigung prüfen` [boolean] A flag that determines whether the tool should check for edge contamination. This is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being counted.

Vorgabe: *True*

Ausgaben

`D-Infinity Entfernung hoch` [raster] Grid containing the distances up to the ridge calculated using the D-Infinity flow model and the statistical and path methods chosen.

Algorithm ID: `taudem:dinfdistup`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

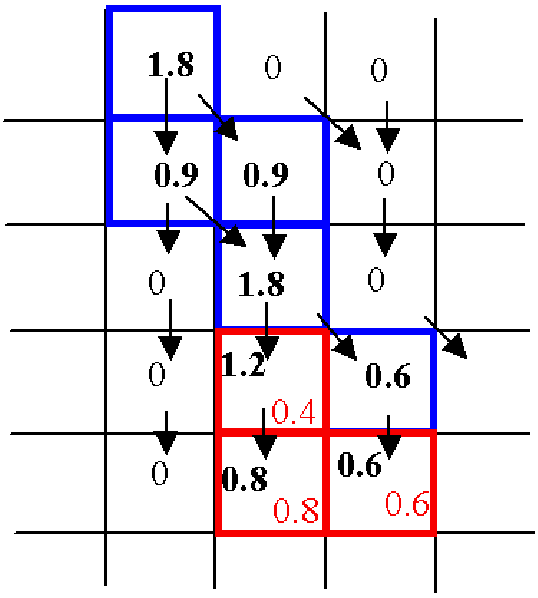
The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity umgekehrte Akkumulation

Beschreibung

This works in a similar way to evaluation of weighted Contributing area, except that the accumulation is by propagating the weight loadings upslope along the reverse of the flow directions to accumulate the quantity of weight loading downslope from each grid cell. The function also reports the maximum value of the weight loading downslope from each grid cell in the Maximum Downslope grid.

This function is designed to evaluate and map the hazard due to activities that may have an effect downslope. The example is land management activities that increase runoff. Runoff is sometimes a trigger for landslides or debris flows, so the weight grid here could be taken as a terrain stability map. Then the reverse accumulation provides a measure of the amount of unstable terrain downslope from each grid cell, as an indicator of the danger of activities that may increase runoff, even though there may be no potential for any local impact.



Reverse accumulation of field weights indicated in red

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions“**.

`Weight Grid` [raster] A grid giving weights (loadings) to be used in the accumulation.

Ausgaben

`Reverse Accumulation Grid` [raster] The grid giving the result of the **„Reverse Accumulation“** function. This works in a similar way to evaluation of weighted Contributing area, except that the accumulation is by propagating the weight loadings upslope along the reverse of the flow directions to accumulate the quantity of loading downslope from each grid cell.

`Maximum Downslope Grid` [raster] The grid giving the maximum of the weight loading grid downslope from each grid cell.

Algorithm ID: `taudem:dinfrevaccum`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity transportbegrenzte Akkumulation - 2

Beschreibung

This function is designed to calculate the transport and deposition of a substance (e.g. sediment) that may be limited by both supply and the capacity of the flow field to transport it. This function accumulates substance flux (e.g. sediment transport) subject to the rule that transport out of any grid cell is the minimum between supply and transport capacity, `Tcap`. The total supply at a grid cell is calculated as the sum of the transport in from upslope grid cells, `Tin`, plus the local supply contribution, `E` (e.g. erosion). This function also outputs deposition, `D`, calculated as total supply minus actual transport.

$$T_{out} = \min(E + \sum T_{in}, T_{cap})$$

$$D = E + \sum T_{in} - T_{out}$$

Here `E` is the supply. `Tout` at each grid cell becomes `Tin` for downslope grid cells and is reported as Transport limited accumulation (`tla`). `D` is deposition (`tdep`). The function provides the option to evaluate concentration of a compound (contaminant) adhered to the transported substance. This is evaluated as follows:

$$L_{in} = \sum T_{in} C_{in}$$

Where `Lin` is the total incoming compound loading and `Cin` and `Tin` refer to the Concentration and Transport entering from each upslope grid cell.

$$T_{out} < \sum T_{in}$$

If

else

where `Cs` is the concentration supplied locally and the difference in the second term on the right represents the additional supply from the local grid cell. Then,

`Cout` at each grid cell comprises is the concentration grid output from this function.

If the outlets shapefile is used the tool only evaluates that part of the domain that contributes flow to the locations given by the shapefile.

Transport limited accumulation is useful for modeling erosion and sediment delivery, including the spatial dependence of sediment delivery ratio and contaminant that adheres to sediment.

$$L_{out} = L_{in} \left(T_{out} / \sum T_{in} \right)$$

$$L_{out} = L_{in} + C_s \left(T_{out} - \sum T_{in} \right)$$

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions"**.

`Supply Grid` [raster] A grid giving the supply (loading) of material to a transport limited accumulation function. In the application to erosion, this grid would give the erosion detachment, or sediment supplied at each grid cell.

`Transportkapazitätsgitter` [raster] A grid giving the transport capacity at each grid cell for the transport limited accumulation function. In the application to erosion this grid would give the transport capacity of the carrying flow.

`Eingangs Konzentrationsgitter` [raster] A grid giving the concentration of a compound of interest in the supply to the transport limited accumulation function. In the application to erosion, this grid would give the concentration of say phosphorous adhered to the eroded sediment.

`Auslaß-Shapedatei` [vector: point] Optional

This optional input is a point shapefile defining outlets of interest. If this file is used, the tool will only evaluate the area upslope of these outlets.

`Kantenverunreinigung prüfen` [boolean] This option determines whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being considered when determining the result.

Vorgabe: *True*

Ausgaben

`Transportbegrenztes Akkumulations Gitter` [raster] This grid is the weighted accumulation of supply accumulated respecting the limitations in transport capacity and reports the transport rate calculated by accumulating the substance flux subject to the rule that the transport out of any grid cell is the minimum of the total supply (local supply plus transport in) to that grid cell and the transport capacity.

`Ablagerungs Raster` [raster] A grid giving the deposition resulting from the transport limited accumulation. This is the residual from the transport in to each grid cell minus the transport capacity out of the grid cell. The deposition grid is calculated as the transport in + the local supply - the transport out.

`Ausgangs Konzentrationsgitter` [raster] If an input concentration in supply grid is given, then this grid is also output and gives the concentration of a compound (contaminant) adhered or bound to the transported substance (e.g. sediment) is calculated.

$$C_{out} = L_{out} / T_{out}$$

Algorithm ID: unknown

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D-Infinity transportbegrenzte Akkumulation

Beschreibung

This function is designed to calculate the transport and deposition of a substance (e.g. sediment) that may be limited by both supply and the capacity of the flow field to transport it. This function accumulates substance flux (e.g. sediment transport) subject to the rule that transport out of any grid cell is the minimum between supply and transport capacity, `Tcap`. The total supply at a grid cell is calculated as the sum of the transport in from upslope grid cells, `Tin`, plus the local supply contribution, `E` (e.g. erosion). This function also outputs deposition, `D`, calculated as total supply minus actual transport.

$$T_{out} = \min(E + \sum T_{in}, T_{cap})$$

$$D = E + \sum T_{in} - T_{out}$$

Here `E` is the supply. `Tout` at each grid cell becomes `Tin` for downslope grid cells and is reported as Transport limited accumulation (`tla`). `D` is deposition (`tdep`). The function provides the option to evaluate concentration of a compound (contaminant) adhered to the transported substance. This is evaluated as follows:

$$L_{in} = \sum T_{in} C_{in}$$

Where `Lin` is the total incoming compound loading and `Cin` and `Tin` refer to the Concentration and Transport entering from each upslope grid cell.

$$T_{out} < \sum T_{in}$$

If

else

where `Cs` is the concentration supplied locally and the difference in the second term on the right represents the additional supply from the local grid cell. Then,

`Cout` at each grid cell comprises is the concentration grid output from this function.

$$L_{out} = L_{in}\left(T_{out} / \sum T_{in}\right)$$

$$L_{out} = L_{in} + C_s\left(T_{out} - \sum T_{in}\right)$$

If the outlets shapefile is used the tool only evaluates that part of the domain that contributes flow to the locations given by the shapefile.

Transport limited accumulation is useful for modeling erosion and sediment delivery, including the spatial dependence of sediment delivery ratio and contaminant that adheres to sediment.

Parameter

`D-Infinity Fliessrichtung Raster` [raster] A grid giving flow direction by the D-infinity method. Flow direction is measured in radians, counter clockwise from east. This can be created by the tool **„D-Infinity Flow Directions“**.

`Supply Grid` [raster] A grid giving the supply (loading) of material to a transport limited accumulation function. In the application to erosion, this grid would give the erosion detachment, or sediment supplied at each grid cell.

`Transportkapazitätsgitter` [raster] A grid giving the transport capacity at each grid cell for the transport limited accumulation function. In the application to erosion this grid would give the transport capacity of the carrying flow.

`Auslaß-Shapedatei` [vector: point] Optional

This optional input is a point shapefile defining outlets of interest. If this file is used, the tool will only evaluate the area upslope of these outlets.

`Kantenverunreinigung prüfen` [boolean] This option determines whether the tool should check for edge contamination. Edge contamination is defined as the possibility that a value may be underestimated due to grid cells outside of the domain not being considered when determining the result.

Vorgabe: *True*

Ausgaben

`Transportbegrenztes Akkumulations Gitter` [raster] This grid is the weighted accumulation of supply accumulated respecting the limitations in transport capacity and reports the transport rate calculated by accumulating the substance flux subject to the rule that the transport out of any grid cell is the minimum of the total supply (local supply plus transport in) to that grid cell and the transport capacity.

`Ablagerungs Raster` [raster] A grid giving the deposition resulting from the transport limited accumulation. This is the residual from the transport in to each grid cell minus the transport capacity out of the grid cell. The deposition grid is calculated as the transport in + the local supply - the tranport out.

Algorithm ID: `taudem:dinftranslimaccum`

$$C_{out} = L_{out} / T_{out}$$

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

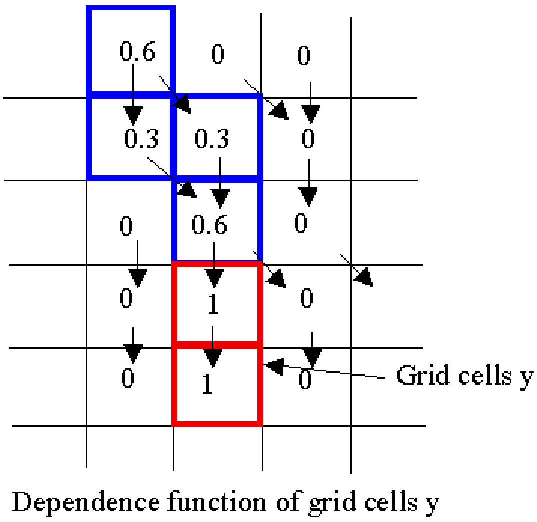
The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

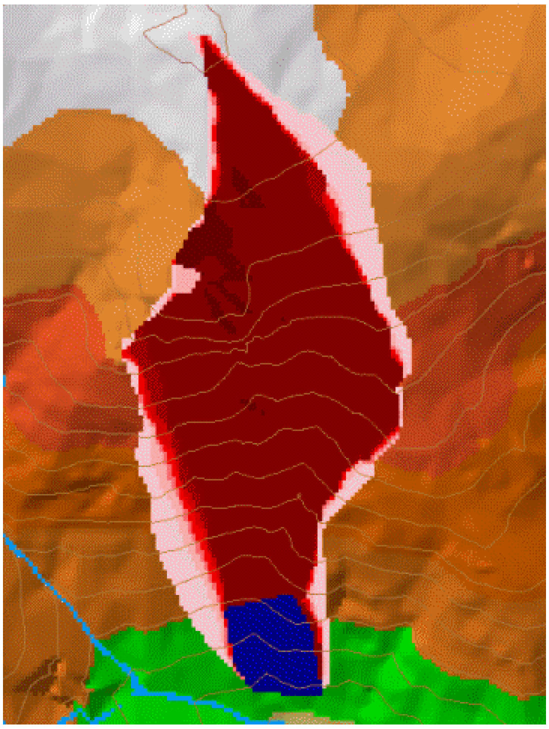
D-Unendlich Aufwärtsabhängigkeit

Beschreibung

The D-Infinity Upslope Dependence tool quantifies the amount each grid cell in the domain contributes to a destination set of grid cells. D-Infinity flow directions proportion flow from each grid cell between multiple downslope grid cells. Following this flow field downslope the amount of flow originating at each grid cell that reaches the destination zone is defined. Upslope influence is evaluated using a downslope recursion, examining grid cells downslope from each grid cell, so that the map produced identifies the area upslope where flow through the destination zone originates, or the area it depends on, for its flow.

The figures below illustrate the amount each source point in the domain `x` (blue) contributes to the destination point or zone `y` (red). If the indicator weighted contributing area function is denoted `I(y; x)` giving the weighted contribution using a unit value (1) from specific grid cells `y` to grid cells `x`, then the upslope dependence is: `D(x; y) = I(y; x)`.





This is useful for example to track where flow or a flow related substance or contaminant that enters a destination area may come from.

Parameter

D-Infinity Fliessrichtung Raster [raster] A grid giving flow direction by the D-Infinity method where the flow direction angle is determined as the direction of the steepest downward slope on the eight triangular facets formed in a 3x3 grid cell window centered on the grid cell of interest. This grid can be produced using the **„D-Infinity Flow Direction“** tool.

Zielgitter [raster] A grid that encodes the destination zone that may receive flow from upslope. This grid must be 1 inside the zone y and 0 over the rest of the domain.

Ausgaben

Output Upslope Dependence Grid [raster] A grid quantifing the amount each source point in the domain contributes to the zone defined by the destination grid.

Algorithm ID: `taudem:dinfupdependence`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Gemitteltes Gefälle

Beschreibung

This tool computes slope in a D8 downslope direction averaged over a user selected distance. Distance should be specified in horizontal map units.

Parameter

D8 Fließrichtung Raster [raster] This input is a grid of flow directions that are encoded using the D8 method where all flow from a cells goes to a single neighboring cell in the direction of steepest descent. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.

Höhengitter mit gefüllten Löchern [raster] This input is a grid of elevation values. As a general rule, it is recommended that you use a grid of elevation values that have had the pits removed for this input. Pits are generally taken to be artifacts that interfere with the analysis of flow across them. This grid can be obtained as the output of the **„Pit Remove“** tool, in which case it contains elevation values where the pits have been filled to the point where they just drain.

Downslope Distance [number] Input parameter of downslope distance over which to calculate the slope (in horizontal map units).

Vorgabe: *50*

Ausgaben

`Slope Average Down Grid` [raster] This output is a grid of slopes calculated in the D8 downslope direction, averaged over the selected distance.

Algorithm ID: `taudem:slopeavedown`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Gefälle-über-Flächenverhältnis

Beschreibung

Calculates the ratio of the slope to the specific catchment area (contributing area). This is algebraically related to the more common ln(a/tan beta) wetness index, but contributing area is in the denominator to avoid divide by 0 errors when slope is 0.

Parameter

`Neigungsgitter` [raster] A grid of slope. This grid can be generated using ether the **„D8 Flow Directions“** tool or the **„D-Infinity Flow Directions“** tool.

`Specific Catchment Area Grid` [raster] A grid giving the contributing area value for each cell taken as its own contribution plus the contribution from upslope neighbors that drain in to it. Contributing area is counted in terms of the number of grid cells (or summation of weights). This grid can be generated using either the **„D8 Contributing Area“** tool or the **„D-Infinity Contributing Area“** tool.

Ausgaben

`Slope Divided By Area Ratio Grid` [raster] A grid of the ratio of slope to specific catchment area (contributing area). This is algebraically related to the more common `ln(a/tan beta)` wetness index, but contributing area is in the denominator to avoid divide by 0 errors when slope is 0.

Algorithm ID: `taudem:slopearearatio`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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 wetness index

Beschreibung

Calculates the topographic wetness index (TWI).

Parameter

`Slope` [raster] A grid of slope. This grid can be generated using ether the **„D8 Flow Directions"** tool or the **„D-Infinity Flow Directions"** tool.

`Specific catchment area` [raster] A grid giving the contributing area value for each cell taken as its own contribution plus the contribution from upslope neighbors that drain in to it. Contributing area is counted in terms of the number of grid cells (or summation of weights). This grid can be generated using either the **„D8 Contributing Area"** tool or the **„D-Infinity Contributing Area"** tool.

Ausgaben

`Wetness index` [raster] A grid of the wetness index (TWI).

Algorithm ID: `taudem:twi`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.4.3 Stromnetzwerkanalyse

Connect down

Beschreibung

Parameter

`D8 flow directions` [raster] A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions"** tool.

`D8 contribution area` [raster]

`Watershed` [raster]

`Grid cells move to downstream` [number]

`Outlets` [vector: point] Optional

A point shape file defining outlets of interest. If this input file is used, only the area upslope of these outlets will be evaluated by the tool.

Ausgaben

`Extreme Upslope Values Grid` [raster] A grid of the maximum/minimum upslope values.

Algorithm ID: `taudem:connectdown`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

D8 Extremer Aufwärtswert

Beschreibung

Evaluates the extreme (either maximum or minimum) upslope value from an input grid based on the D8 flow model. This is intended initially for use in stream raster generation to identify a threshold of the slope times area product that results in an optimum (according to drop analysis) stream network.

If the optional outlet point shapefile is used, only the outlet cells and the cells upslope (by the D8 flow model) of them are in the domain to be evaluated.

By default, the tool checks for edge contamination. This is defined as the possibility that a result may be underestimated due to grid cells outside of the domain not being counted. This occurs when drainage is inwards from the boundaries or areas with „no data" values for elevation. The algorithm recognizes this and reports „no data" for the result for these grid cells. It is common to see streaks of „no data" values extending inwards from boundaries along flow paths that enter the domain at a boundary. This is the desired effect and indicates that the result for these grid cells is unknown due to it being dependent on terrain outside of the domain of data available. Edge contamination checking may be turned off in cases where you know this is not an issue or want to ignore these problems, if for example, the DEM has been clipped along a watershed outline.

Parameter

`D8 Fließrichtung Raster` [raster] A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions"** tool.

`Upslope Values Grid` [raster] This is the grid of values of which the maximum or minimum upslope value is selected. The values most commonly used are the slope times area product needed when generating stream rasters according to drop analysis.

`Auslaß-Shapedatei` [vector: point] Optional

A point shape file defining outlets of interest. If this input file is used, only the area upslope of these outlets will be evaluated by the tool.

`Kantenverunreinigung prüfen` [boolean] A flag that indicates whether the tool should check for edge contamination.

Vorgabe: *True*

`Use max upslope value` [boolean] A flag to indicate whether the maximum or minimum upslope value is to be calculated.

Vorgabe: *True*

Ausgaben

`Extreme Upslope Values Grid` [raster] A grid of the maximum/minimum upslope values.

Algorithm ID: `taudem:d8flowpathextremeup`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Einzugsgebiet messen

Beschreibung

Calculates Gage Watersheds Grid. Each grid cell is labeled with the identifier (from column `id`) of the gage to which it drains directly without passing through any other gages.

Parameter

`D8 Fließrichtung Raster` [raster] A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.

`Gages Shapefile` [vector: point] A point shapefile defining the gages to which watersheds will be delineated. This shapefile should have a colmun `id`. Grid cells draining directly to each point in this shapefile will be labeled with this id.

Ausgaben

`Gage Watershed Grid` [raster] A grid identifies each gage watershed. Each grid cell is labeled with the identifier (from column `id`) of the gage to which it drains directly without passing through any other gages.

`Downstream Identifiers File` [file] Text file giving watershed downslope connectivity

Algorithm ID: `taudem:gagewatershed`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Länge Fläche Strom Quelle

Beschreibung

Creates an indicator grid (1, 0) that evaluates `A >= (M)(Ly)` based on upslope path length, D8 contributing area grid inputs, and parameters `M` and `y`. This grid indicates likely stream source grid cells. This is an experimental method with theoretical basis in Hack's law which states that for streams `L ~ A 0.6`. However for hillslopes with parallel flow `L ~ A`. So a transition from hillslopes to streams may be represented by `L ~ A 0.8` suggesting identifying grid cells as stream cells if `A > M (L (1/0.8))`.

Parameter

Length Grid [raster] A grid of the maximum upslope length for each cell. This is calculated as the length of the flow path from the furthest cell that drains to each cell. Length is measured between cell centers taking into account cell size and whether the direction is adjacent or diagonal. It is this length (`L`) that is used in the formula, `A >(M)(Ly)`, to determine which cells are considered stream cells. This grid can be obtained as an output from the **„Grid Network“** tool.

Contributing Area Grid [raster] A grid of contributing area values for each cell that were calculated using the D8 algorithm. The contributing area for a cell is the sum of its own contribution plus the contribution from all upslope neighbors that drain to it, measured as a number of cells. This grid is typically obtained as the output of the **„D8 Contributing Area“** tool. In this tool, it is the contributing area (`A`) that is compared in the formula `A > (M)(Ly)` to determine the transition to a stream.

Schwellen [number] The multiplier threshold (`M`) parameter which is used in the formula: `A > (M)(Ly)`, to identify the beginning of streams.

Vorgabe: *0.03*

Exponent [number] The exponent (`y`) parameter which is used in the formula: `A > (M)(Ly)`, to identify the beginning of streams. In branching systems, Hack's law suggests that `L = 1/M A(1/y)` with `1/y = 0.6` (or 0.56) (`y` about 1.7). In parallel flow systems `L` is proportional to `A` (`y` about 1). This method tries to identify the transition between these two paradigms by using an exponent `y` somewhere in between (`y` about 1.3).

Vorgabe: *1.3*

Ausgaben

Strom Quellennetz [raster] An indicator grid (1,0) that evaluates A >= (M)(L^y), based on the maximum upslope path length, the D8 contributing area grid inputs, and parameters `M` and `y`. This grid indicates likely stream source grid cells.

Algorithm ID: `taudem:lengtharea`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Abläufe zu Strömem verschieben

Beschreibung

Moves outlet points that are not aligned with a stream cell from a stream raster grid, downslope along the D8 flow direction until a stream raster cell is encountered, the „max_dist“ number of grid cells are examined, or the flow path exits the domain (i.e. a „no data“ value is encountered for the D8 flow direction). The output file is a new outlets shapefile where each point has been moved to coincide with the stream raster grid, if possible. A field „dist_moved“ is added to the new outlets shapefile to indicate the changes made to each point. Points that are already on a stream cell are not moved and their „dist_moved“ field is assigned a value 0. Points that are initially not on a stream cell are moved by sliding them downslope along the D8 flow direction until one of the following occurs: a) A stream raster grid cell is encountered before traversing the „max_dist“ number of grid cells. In which case, the point is moved and the „dist_moved“ field is assigned a value indicating how many grid cells the point was moved. b) More than the „max_number“ of grid cells are traversed, or c) the traversal ends up going out of the domain (i.e., a „no data“ D8 flow direction value is encountered). In which case, the point is not moved and the „dist_moved“ field is assigned a value of -1.

Parameter

`D8 Fließrichtung Raster` [raster] A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.

`Stromrastergitter` [raster] This output is an indicator grid (1, 0) that indicates the location of streams, with a value of 1 for each of the stream cells and 0 for the remainder of the cells. This file is produced by several different tools in the **„Stream Network Analysis“** toolset.

`Auslaß-Shapedatei` [vector: point] A point shape file defining points of interest or outlets that should ideally be located on a stream, but may not be exactly on the stream due to the fact that the shapefile point locations may not have been accurately registered with respect to the stream raster grid.

`Maximum Number of Grid Cells to traverse` [number] This input parameter is the maximum number of grid cells that the points in the input outlet shapefile will be moved before they are saved to the output outlet shapefile.

Vorgabe: *50*

Ausgaben

`Output Outlet Shapefile` [vector: point] A point shape file defining points of interest or outlets. This file has one point in it for each point in the input outlet shapefile. If the original point was located on a stream, then the point was not moved. If the original point was not on a stream, the point was moved downslope according to the D8 flow direction until it reached a stream or the maximum distance had been reached. This file has an additional field „dist_moved“ added to it which is the number of cells that the point was moved. This field is 0 if the cell was originally on a stream, -1 if it was not moved because there was not a stream within the maximum distance, or some positive value if it was moved.

Algorithm ID: `taudem:moveoutletstostreams`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Peuker-Douglas

Beschreibung

Creates an indicator grid (1, 0) of upward curved grid cells according to the Peuker and Douglas algorithm.

With this tool, the DEM is first smoothed by a kernel with weights at the center, sides, and diagonals. The Peuker and Douglas (1975) method (also explained in Band, 1986), is then used to identify upwardly curving grid cells. This technique flags the entire grid, then examines in a single pass each quadrant of 4 grid cells, and unflags the highest. The remaining flagged cells are deemed „upwardly curved“, and when viewed, resemble a channel network. This proto-channel network generally lacks connectivity and requires thinning, issues that were discussed in detail by Band (1986).

Parameter

`Höhengitter` [raster] A grid of elevation values. This is usually the output of the **„Pit Remove“** tool, in which case it is elevations with pits removed.

`Mitteglättungsgewichtung` [number] The center weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Vorgabe: *0.4*

`Seitenglättungsgewichtung` [number] The side weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Vorgabe: *0.1*

`Diagonale Glättungsgewichtung` [number] The diagonal weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Vorgabe: *0.05*

Ausgaben

`Strom Quellennetz` [raster] An indicator grid (1, 0) of upward curved grid cells according to the Peuker and Douglas algorithm, and if viewed, resembles a channel network. This proto-channel network generally lacks connectivity and requires thinning, issues that were discussed in detail by Band (1986).

Siehe auch

- Band, L. E., (1986), „Topographic partition of watersheds with digital elevation models“, Water Resources Research, 22(1): 15-24.
- Peuker, T. K. and D. H. Douglas, (1975), „Detection of surface-specific points by local parallel processing of discrete terrain elevation data“, Comput. Graphics Image Process., 4: 375-387.

Algorithm ID: `taudem:peukerdouglas`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Peuker Douglas stream

Beschreibung

Parameter

Ausgaben

`Stream source` [raster] An indicator grid (1, 0) of upward curved grid cells according to the Peuker and Douglas algorithm, and if viewed, resembles a channel network. This proto-channel network generally lacks connectivity and requires thinning, issues that were discussed in detail by Band (1986).

Algorithm ID: `taudem:peukerdouglasstreamdef`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Neigungsflächenkombination

Beschreibung

Creates a grid of slope-area values = `(Sm)  (An)` based on slope and specific catchment area grid inputs, and parameters `m` and `n`. This tool is intended for use as part of the slope-area stream raster delineation method.

Parameter

`Neigungsgitter` [raster] This input is a grid of slope values. This grid can be obtained from the **„D-Infinity Flow Directions“** tool.

`Contributing Area Grid` [raster] A grid giving the specific catchment area for each cell taken as its own contribution (grid cell length or summation of weights) plus the proportional contribution from upslope neighbors that drain in to it. This grid is typically obtained from the **„D-Infinity Contributing Area“** tool.

`Neigungs Exponent` [number] The slope exponent (`m`) parameter which will be used in the formula: `(Sm) (An)`, that is used to create the slope-area grid.

Vorgabe: *2*

`Flächen-Exponent` [number] The area exponent (`n`) parameter which will be used in the formula: `(Sm) (An)`, that is used to create the slope-area grid.

Vorgabe: *1*

Ausgaben

`Neigungsflächen Gitter` [raster] A grid of slope-area values = `(Sm) (An)` calculated from the slope grid, specific catchment area grid, `m` slope exponent parameter, and `n` area exponent parameter.

Algorithm ID: `taudem:slopearea`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the 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 area stream definition

Beschreibung

Creates a grid of slope-area values = `(Sm)  (An)` based on slope and specific catchment area grid inputs, and parameters `m` and `n`. This tool is intended for use as part of the slope-area stream raster delineation method.

Parameter

`D8 flow directions` [raster]

`D-infinity Contributing Area` [raster] A grid giving the specific catchment area for each cell taken as its own contribution (grid cell length or summation of weights) plus the proportional contribution from upslope neighbors that drain in to it. This grid is typically obtained from the **„D-Infinity Contributing Area“** tool.

`Slope` [raster] This input is a grid of slope values. This grid can be obtained from the **„D-Infinity Flow Directions“** tool.

`Mask grid` [raster]

`Outlets` [vector: point]

`Pit-filled grid for drop analysis` [raster]

`D8 contributing area for drop analysis` [raster]

`Neigungs Exponent` [number] The slope exponent (`m`) parameter which will be used in the formula: `(Sm) (An)`, that is used to create the slope-area grid.

Vorgabe: *2*

`Flächen-Exponent` [number] The area exponent (`n`) parameter which will be used in the formula: `(Sm) (An)`, that is used to create the slope-area grid.

Vorgabe: *1*

`Accumulation threshold` [number]

`Minimum threshold` [number]

`Maximum threshold` [number]

`Number of drop thresholds` [number]

`Type of threshold step` [enumeration].

Optionen:

- 0 — Logarithmisch
- 1 — Linear

Vorgabe: *0*

`Kantenverunreinigung prüfen` [boolean]

`Select threshold by drop analysis` [boolean]

Ausgaben

`Stream raster` [raster]

`Slope area` [raster] A grid of slope-area values = `(Sm) (An)` calculated from the slope grid, specific catchment area grid, `m` slope exponent parameter, and `n` area exponent parameter.

`Maximum upslope` [raster]

`Drop analysis` [file]

Algorithm ID: `taudem:slopeareastreamdef`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Stromdefinition nach Schwelle

Beschreibung

Operates on any grid and outputs an indicator (1, 0) grid identifying cells with input values >= the threshold value. The standard use is to use an accumulated source area grid to as the input grid to generate a stream raster grid as the output. If you use the optional input mask grid, it limits the domain being evaluated to cells with mask values >= 0. When you use a D-infinity contributing area grid (`*sca`) as the mask grid, it functions as an edge contamination mask. The threshold logic is:

```
src = ((ssa >= thresh) & (mask >= s0)) ? 1:0
```

Parameter

`Accumulated Stream Source Grid` [raster] This grid nominally accumulates some characteristic or combination of characteristics of the watershed. The exact characteristic(s) varies depending on the stream network raster algorithm being used. This grid needs to have the property that grid cell values are monotonically increasing downslope along D8 flow directions, so that the resulting stream network is continuous. While this grid is often from an accumulation, other sources such as a maximum upslope function will also produce a suitable grid.

`Schwellen` [number] This parameter is compared to the value in the Accumulated Stream Source grid (`*ssa`) to determine if the cell should be considered a stream cell. Streams are identified as grid cells for which ssa value is >= this threshold.

Vorgabe: *100*

`Mask Grid` [raster] Optional

This optional input is a grid that is used to mask the domain of interest and output is only provided where this grid is >= 0. A common use of this input is to use a D-Infinity contributing area grid as the mask so that the delineated stream network is constrained to areas where D-infinity contributing area is available, replicating the functionality of an edge contamination mask.

Ausgaben

`Stromrastergitter` [raster] This is an indicator grid (1, 0) that indicates the location of streams, with a value of 1 for each of the stream cells and 0 for the remainder of the cells.

Algorithm ID: `taudem:threshold`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Stream definition with drop analysis

Beschreibung

Parameter

Ausgaben

Algorithm ID: `taudem:streamdefdropanalysis`

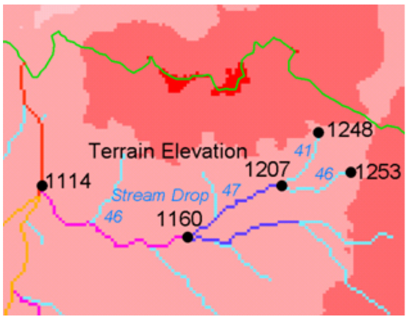
```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Gewässergefälleanalyse

Beschreibung

Applies a series of thresholds (determined from the input parameters) to the input accumulated stream source grid (`*ssa`) grid and outputs the results in the `*drp.txt` file the stream drop statistics table. This function is designed to aid in the determination of a geomorphologically objective threshold to be used to delineate streams. Drop Analysis attempts to select the right threshold automatically by evaluating a stream network for a range of thresholds and examining the constant drop property of the resulting Strahler streams. Basically it asks the question: Is the mean stream drop for first order streams statistically different from the mean stream drop for higher order streams, using a T-test. Stream drop is the difference in elevation from the beginning to the end of a stream defined as the sequence of links of the same stream order. If the T-test shows a significant difference then the stream network does not obey this „law“ so a larger threshold needs to be chosen. The smallest threshold for which the T-test does not show a significant difference gives the highest resolution stream network that obeys the constant stream drop „law“ from geomorphology, and is the threshold chosen for the „objective“ or automatic mapping of streams from the DEM. This function can be used in the development of stream network rasters, where the exact watershed characteristic(s) that were accumulated in the accumulated stream source grid vary based on the method being used to determine the stream network raster.



The constant stream drop „law“ was identified by Broscoe (1959). For the science behind using this to determine a stream delineation threshold, see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Parameter

`D8 beitragende Fläche Raster` [raster] A grid of contributing area values for each cell that were calculated using the D8 algorithm. The contributing area for a cell is the sum of its own contribution plus the contribution from all upslope neighbors that drain to it, measured as a number of cells or the sum of weight loadings. This grid can be obtained as the output of the **„D8 Contributing Area“** tool. This grid is used in the evaluation of drainage density reported in the stream drop table.

`D8 Fließrichtung Raster` [raster] A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions“** tool.

`Höhengitter mit gefüllten Löchern` [raster] A grid of elevation values. This is usually the output of the **„Pit Remove“** tool, in which case it is elevations with pits removed.

`Accumulated Stream Source Grid` [raster] This grid must be monotonically increasing along the downslope D8 flow directions. It it compared to a series of thresholds to determine the beginning of the streams. It is often generated by accumulating some characteristic or combination of characteristics of the watershed with the **„D8 Contributing Area“** tool, or using the maximum option of the **„D8 Flow Path Extreme“** tool. The exact method varies depending on the algorithm being used.

`Auslaß-Shapedatei` [vector: point] A point shapefile defining the outlets upstream of which drop analysis is performed.

`Minimum Schwellenwert` [number] This parameter is the lowest end of the range searched for possible threshold values using drop analysis. This technique looks for the smallest threshold in the range where the absolute value of the t-statistic is less than 2. For the science behind the drop analysis see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Vorgabe: *5*

`Maximum Schwellenwert` [number] This parameter is the highest end of the range searched for possible threshold values using drop analysis. This technique looks for the smallest threshold in the range where the absolute value of the t-statistic is less than 2. For the science behind the drop analysis see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Vorgabe: *500*

`Anzahl der Schwellenwerte` [number] The parameter is the number of steps to divide the search range into when looking for possible threshold values using drop analysis. This technique looks for the smallest threshold in the range where the absolute value of the t-statistic is less than 2. For the science behind the drop analysis see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Vorgabe: *10*

`Spacing for Threshold Values` [enumeration] This parameter indicates whether logarithmic or linear spacing should be used when looking for possible threshold values using drop analysis.

Optionen:

- 0 — Logarithmisch
- 1 — Linear

Vorgabe: *0*

Ausgaben

D-Infinity Abstand nach unten [file] Dies ist eine Komma getrennte Textdatei mit der folgenden Kopfzeile :

:: Threshold,DrainDen,NoFirstOrd,NoHighOrd,MeanDFirstOrd,MeanDHighOrd,StdDevFirstOrd,StdDevHighOrd,T

The file then contains one line of data for each threshold value examined, and then a summary line that indicates the optimum threshold value. This technique looks for the smallest threshold in the range where the absolute value of the t-statistic is less than 2. For the science behind the drop analysis, see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Algorithm ID: `taudem:dropanalysis`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

Siehe auch

- Broscoe, A. J., (1959), „Quantitative analysis of longitudinal stream profiles of small watersheds“, Office of Naval Research, Project NR 389-042, Technical Report No. 18, Department of Geology, Columbia University, New York.
- Tarboton, D. G., R. L. Bras and I. Rodriguez-Iturbe, (1991), „On the Extraction of Channel Networks from Digital Elevation Data“, Hydrologic Processes, 5(1): 81-100.
- Tarboton, D. G., R. L. Bras and I. Rodriguez-Iturbe, (1992), „A Physical Basis for Drainage Density“, Geomorphology, 5(1/2): 59-76.
- Tarboton, D. G. and D. P. Ames, (2001), „Advances in the mapping of flow networks from digital elevation data“, World Water and Environmental Resources Congress, Orlando, Florida, May 20-24, ASCE, https://www.researchgate.net/publication/2329568_Advances_in_the_Mapping_of_Flow_Networks_From_Digital_Elevation_Data.

Flußausdehnung und Wasserscheiden

Beschreibung

This tool produces a vector network and shapefile from the stream raster grid. The flow direction grid is used to connect flow paths along the stream raster. The Strahler order of each stream segment is computed. The subwatershed draining to each stream segment (reach) is also delineated and labeled with the value identifier that corresponds to the WSNO (watershed number) attribute in the Stream Reach Shapefile.

This tool orders the stream network according to the Strahler ordering system. Streams that don't have any other streams draining in to them are order 1. When two stream reaches of different order join the order of the downstream reach is the order of the highest incoming reach. When two reaches of equal order join the downstream reach order is increased by 1. When more than two reaches join the downstream reach order is calculated as the maximum of the highest incoming reach order or the second highest incoming reach order + 1. This generalizes the common definition to cases where more than two reaches join at a point. The network topological connectivity is stored in the Stream Network Tree file, and coordinates and attributes from each grid cell along the network are stored in the Network Coordinates file.

The stream raster grid is used as the source for the stream network, and the flow direction grid is used to trace connections within the stream network. Elevations and contributing area are used to determine the elevation and contributing area attributes in the network coordinate file. Points in the outlets shapefile are used to logically split stream reaches to facilitate representing watersheds upstream and downstream of monitoring points. The program uses the attribute field „id" in the outlets shapefile as identifiers in the Network Tree file. This tool then translates the text file vector network representation in the Network Tree and Coordinates files into a shapefile. Further attributes are also evaluated. The program has an option to delineate a single watershed by representing the entire area draining to the Stream Network as a single value in the output watershed grid.

Parameter

`Höhengitter mit gefüllten Löchern` **[raster]** A grid of elevation values. This is usually the output of the **„Pit Remove"** tool, in which case it is elevations with pits removed.

`D8 Fließrichtung Raster` **[raster]** A grid of D8 flow directions which are defined, for each cell, as the direction of the one of its eight adjacent or diagonal neighbors with the steepest downward slope. This grid can be obtained as the output of the **„D8 Flow Directions"** tool.

`D8 Drainage Area` **[raster]** A grid giving the contributing area value in terms of the number of grid cells (or the summation of weights) for each cell taken as its own contribution plus the contribution from upslope neighbors that drain in to it using the D8 algorithm. This is usually the output of the **„D8 Contributing Area"** tool and is used to determine the contributing area attribute in the Network Coordinate file.

`Stromrastergitter` **[raster]** An indicator grid indicating streams, by using a grid cell value of 1 on streams and 0 off streams. Several of the **„Stream Network Analysis"** tools produce this type of grid. The Stream Raster Grid is used as the source for the stream network.

`Outlets Shapefile as Network Nodes` **[vector: point]** Optional

A point shape file defining points of interest. If this file is used, the tool will only deliniate the stream network upstream of these outlets. Additionally, points in the Outlets Shapefile are used to logically split stream reaches to facilitate representing watersheds upstream and downstream of monitoring points. This tool REQUIRES THAT THERE BE an integer attribute field „id" in the Outlets Shapefile, because the „id" values are used as identifiers in the Network Tree file.

`Delineate Single Watershed` **[boolean]** This option causes the tool to delineate a single watershed by representing the entire area draining to the Stream Network as a single value in the output watershed grid. Otherwise a seperate watershed is delineated for each stream reach. Default is *False* (seperate watershed).

Vorgabewert: *False*

Ausgaben

`Stream Order Grid` **[raster]** The Stream Order Grid has cells values of streams ordered according to the Strahler order system. The Strahler ordering system defines order 1 streams as stream reaches that don't have any other reaches draining in to them. When two stream reaches of different order join the order of the downstream reach is the order of the highest incoming reach. When two reaches of equal order join the downstream reach order is increased by 1. When more than two reaches join the downstream reach order is calculated as the maximum of the highest incoming reach order or the second highest incoming reach order + 1. This generalizes the common definition to cases where more than two flow paths reaches join at a point.

`Watershed Grid` **[raster]** This output grid identified each reach watershed with a unique ID number, or in the case where the delineate single watershed option was checked, the entire area draining to the stream network is identified with a single ID.

`Stream Reach Shapefile` **[vector: line]** This output is a polyline shapefile giving the links in a stream network. The columns in the attribute table are:

- LINKNO — Link Number. A unique number associated with each link (segment of channel between junctions). This is arbitrary and will vary depending on number of processes used
- DSLINKNO — Link Number of the downstream link. -1 indicates that this does not exist
- USLINKNO1 — Link Number of first upstream link. (-1 indicates no link upstream, i.e. for a source link)

- USLINKNO2 — Link Number of second upstream link. (-1 indicates no second link upstream, i.e. for a source link or an internal monitoring point where the reach is logically split but the network does not bifurcate)
- DSNODEID — Node identifier for node at downstream end of stream reach. This identifier corresponds to the „id" attribute from the Outlets shapefile used to designate nodes
- Reihenfolge — Strahler-Strom-Reihenfolge
- Length — Length of the link. The units are the horizontal map units of the underlying DEM grid
- Magnitude — Shreve Magnitude of the link. This is the total number of sources upstream
- DS_Cont_Ar — Drainage area at the downstream end of the link. Generally this is one grid cell upstream of the downstream end because the drainage area at the downstream end grid cell includes the area of the stream being joined
- Drop — Drop in elevation from the start to the end of the link
- Slope — Average slope of the link (computed as drop/length)
- Straight_L — Straight line distance from the start to the end of the link
- US_Cont_Ar — Drainage area at the upstream end of the link
- WSNO — Watershed number. Cross reference to the `*w.shp` and `*w` grid files giving the identification number of the watershed draining directly to the link
- DOUT_END — Distance to the eventual outlet (i.e. the most downstream point in the stream network) from the downstream end of the link
- DOUT_START — Distance to the eventual outlet from the upstream end of the link
- DOUT_MID — Distance to the eventual outlet from the midpoint of the link

`Network Connectivity Tree` [file] This output is a text file that details the network topological connectivity is stored in the Stream Network Tree file. Columns are as follows:

- Link Number (Arbitrary — will vary depending on number of processes used)
- Start Point Number in Network coordinates (`*coord.dat)` file (Indexed from 0)
- End Point Number in Network coordinates (`*coord.dat`) file (Indexed from 0)
- Next (Downstream) Link Number. Points to Link Number. -1 indicates no links downstream, i.e. a terminal link
- First Previous (Upstream) Link Number. Points to Link Number. -1 indicates no upstream links
- Second Previous (Upstream) Link Numbers. Points to Link Number. -1 indicates no upstream links. Where only one previous link is -1, it indicates an internal monitoring point where the reach is logically split, but the network does not bifurcate
- Strahler Order of Link
- Monitoring point identifier at downstream end of link. -1 indicates downstream end is not a monitoring point
- Network magnitude of the link, calculated as the number of upstream sources (following Shreve)

`Netzwerk Koordinaten` [file] This output is a text file that contains the coordinates and attributes of points along the stream network. Columns are as follows:

- X-Koordinate
- Y-Koordinate
- Distance along channels to the downstream end of a terminal link
- Geländehöhe
- Beitragende Fläche

Algorithm ID: `taudem:streamnet`

```
import processing
processing.run("algorithm_id", {parameter_dictionary})
```

The *algorithm id* is displayed when you hover over the algorithm in the Processing Toolbox. The *parameter dictionary* provides the parameter NAMEs and values. See *Verarbeitung Algorithmen von der Konsole aus verwenden* for details on how to run processing algorithms from the Python console.

23.5 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.

KAPITEL 24

Plugins

24.1 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.

24.1.1 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* hinzugefgt werden.

Bemerkung: According to the *plugin manager settings*, QGIS main interface can display an icon on the right of the status bar to inform you that there are updates for your installed plugins or new plugins available.

24.1.2 Der Erweiterungen Dialog

The tabs in the Plugins dialog allow the user to install, uninstall and upgrade plugins in different ways. Each plugin has some metadata displayed in the right panel:

- Informationen darüber, ob das Plugin experimentell ist
- Beschreibung
- Bewertungen (Sie können eine Bewertung für Ihre bevorzugte Erweiterung abgeben!)
- Elemente
- einige nützliche Links wie die Homepage, die Fehlerverfolgung und das Quellcode-Repositum
- Autoren
- verfügbare Version

At the top of the dialog, a *Search* function helps you find any plugin using metadata information (author, name, description…). It is available in nearly every tab (except *Settings*).

The Settings tab

The *Settings* tab is the main place you can configure which plugins can be displayed in your application. You can use the following options:

- *Beim Start nach Aktualisierungen suchen*. Wann immer ein neues Plugin oder ein Pluginupdate zur Verfügung steht wird QGIS Sie ‚bei jedem QGIS-Start', ‚einmal am Tag', ‚alle drei Tage', ‚jede Woche', ‚alle zwei Wochen' oder ‚jeden Monat' informieren.
- *Auch experimentelle Erweiterungen zeigen*. QGIS wird Ihnen Plugins in frühen Entwicklungsphasen zeigen, die im Allgemeinen für den Produktiveinsatz ungeeignet sind.
- *Show also deprecated plugins*. Because they use functions that are no longer available in QGIS, these plugins are set deprecated and generally unsuitable for production use. They appear among invalid plugins list.

By default, QGIS provides you with its official plugin repository with the URL https://plugins.qgis.org/plugins/plugins.xml?qgis=3.0 (in case of QGIS 3.0) in the *Plugin repositories* section. 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.

The All tab

In the *All* tab, all the available plugins are listed, including both core and external plugins. Use *Upgrade All* to look for new versions of the plugins. Furthermore, you can use *Install Plugin* if a plugin is listed but not installed, *Uninstall Plugin* as well as *Reinstall Plugin* if a plugin is installed. An installed plugin can be temporarily de/activated using the checkbox.

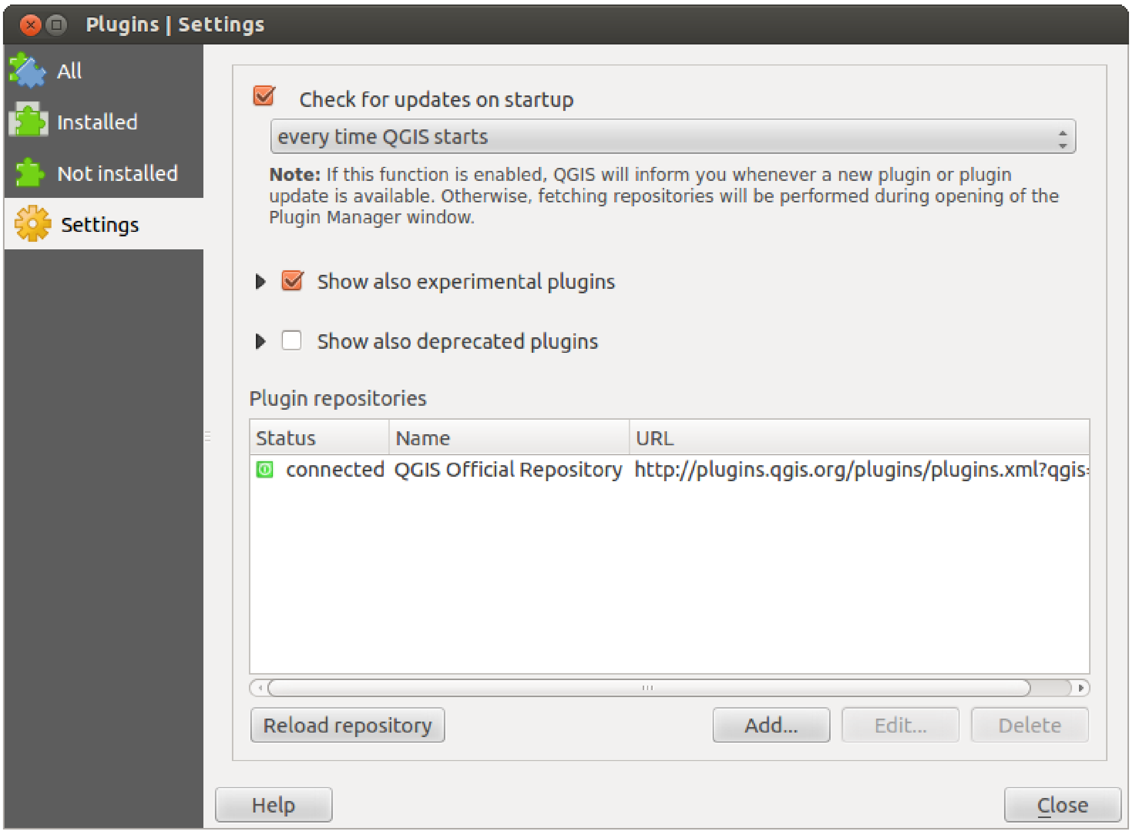


Abb. 24.1: The *Settings* tab

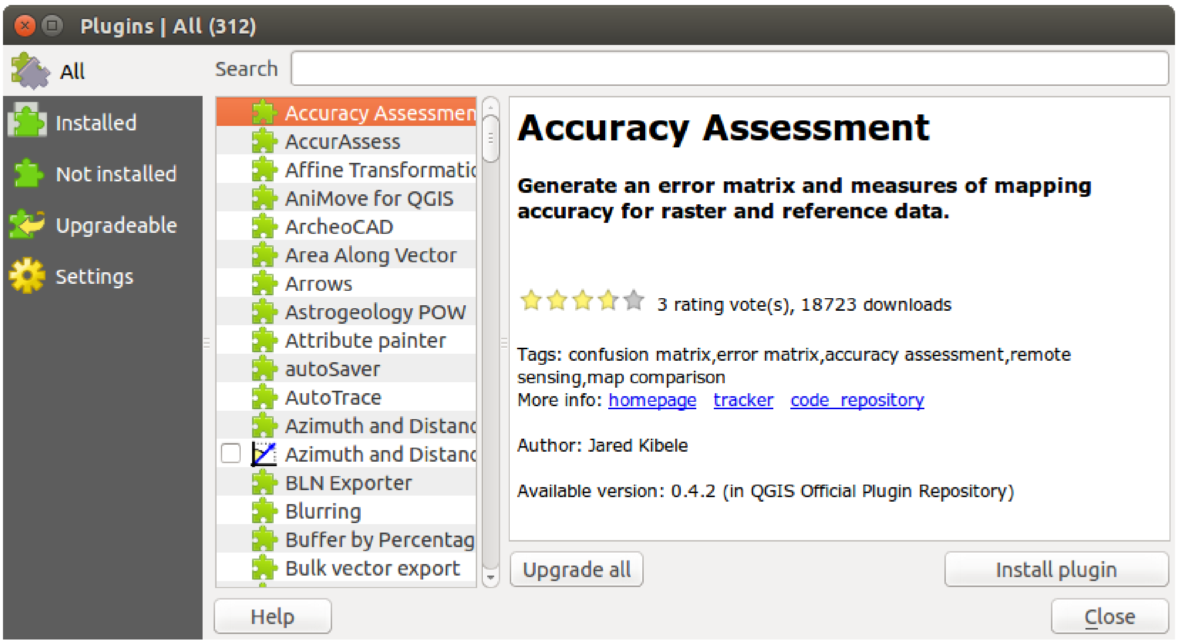


Abb. 24.2: The *All* tab

The Installed tab

In the *Installed* tab, you'll find listed the Core plugins, that you can not uninstall. You can extend this list with external plugins that can be uninstalled and reinstalled any time, using the *Uninstall Plugin* and *Reinstall Plugin* buttons. You can *Upgrade All* the plugins here as well.

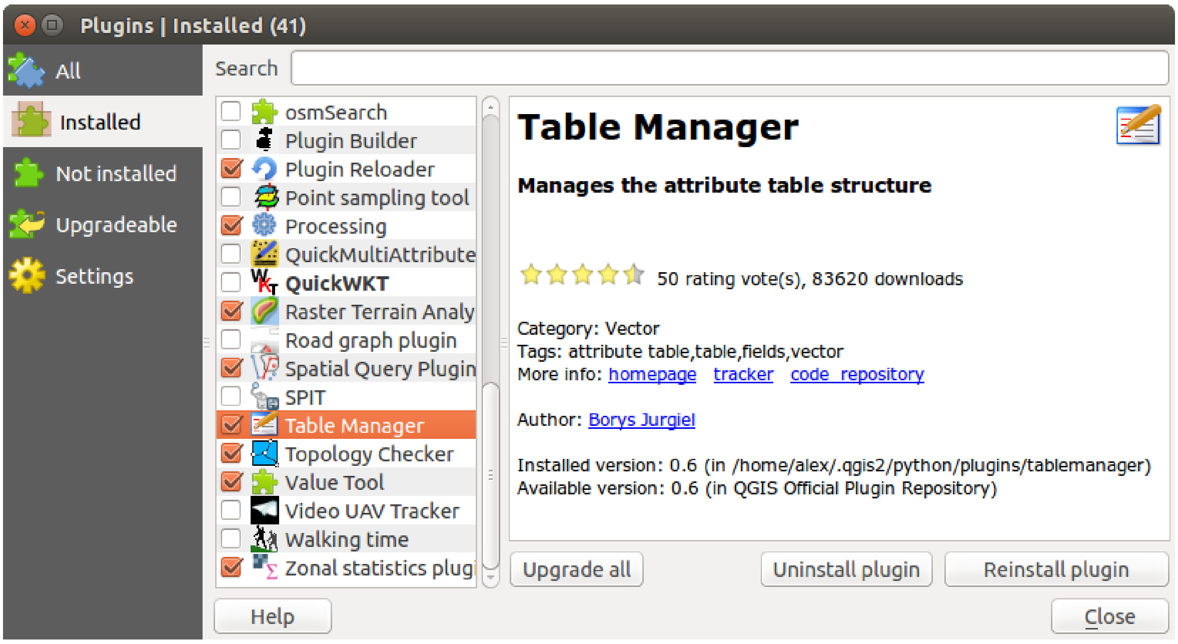


Abb. 24.3: The *Installed* tab

The Not installed tab

The *Not installed* tab lists all plugins available that are not installed. You can use the *Install Plugin* button to implement a plugin into QGIS.

The Upgradeable and New tabs

The *Upgradeable* and *New* tabs are enabled when new plugins are added to the repository or a new version of an installed plugin is released. If you activated *Show also experimental plugins* in the *Settings* menu, those also appear in the list giving you opportunity to early test upcoming tools.

Installation can be done with the *Install Plugin*, *Upgrade Plugin* or *Upgrade All* buttons.

The Invalid tab

The *Invalid* tab lists all installed plugins that are currently broken for any reason (missing dependency, errors while loading, incompatible functions with QGIS version…). You can try the *Reinstall Plugin* button to fix an invalidated plugin but most of the times the fix will be elsewhere (install some libraries, look for another compatible plugin or help to upgrade the broken one).

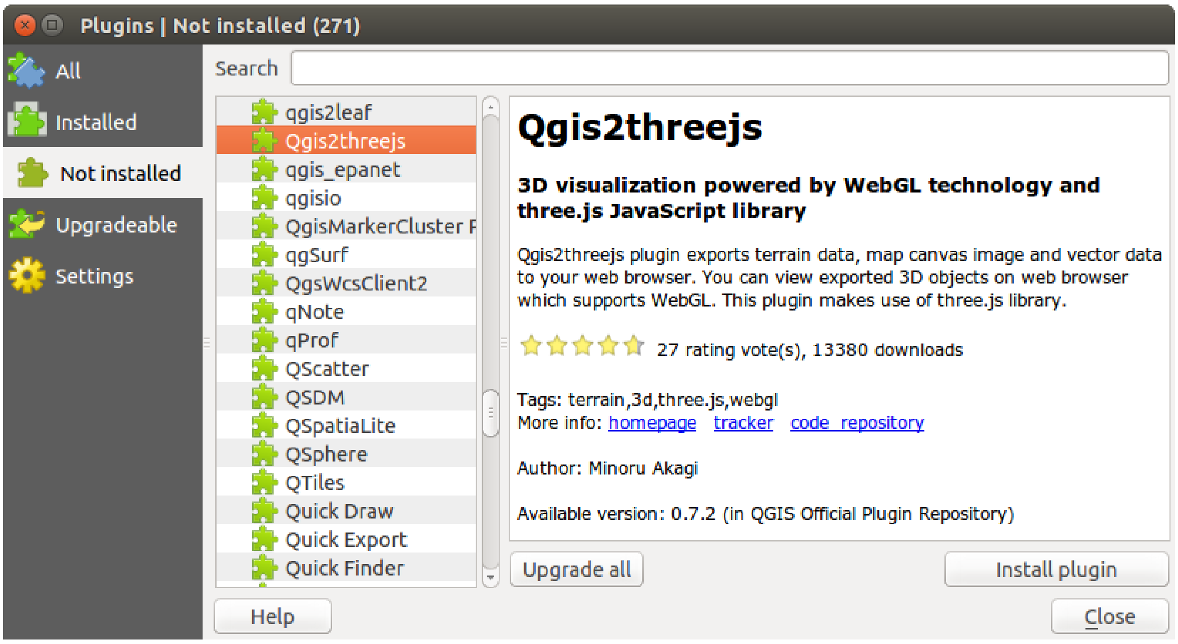


Abb. 24.4: The *Not installed* tab

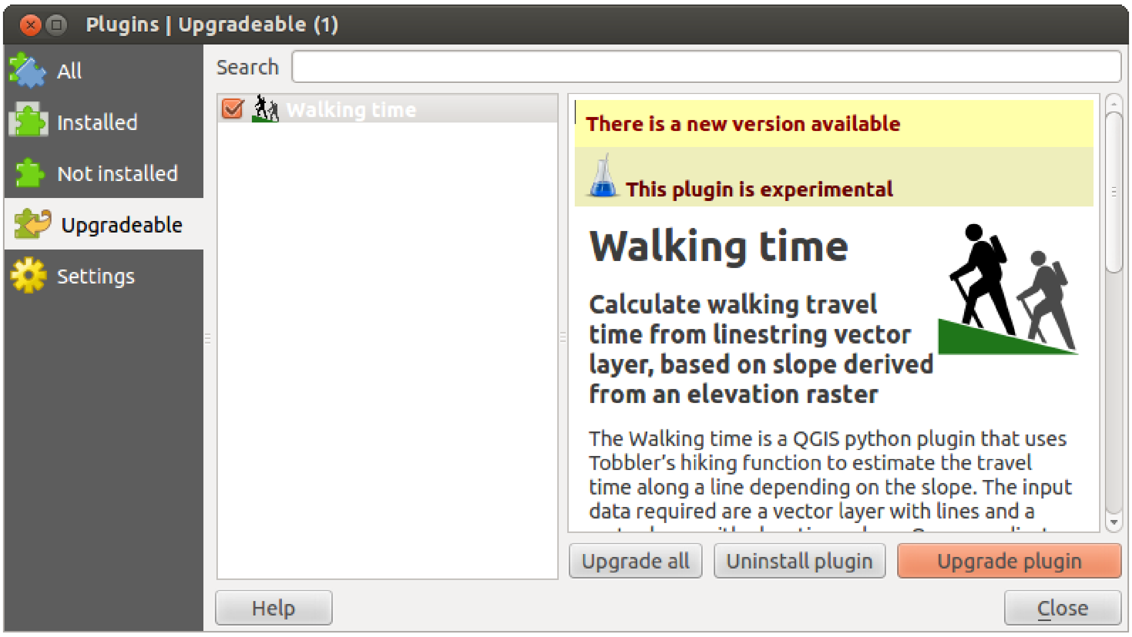


Abb. 24.5: The *Upgradeable* tab

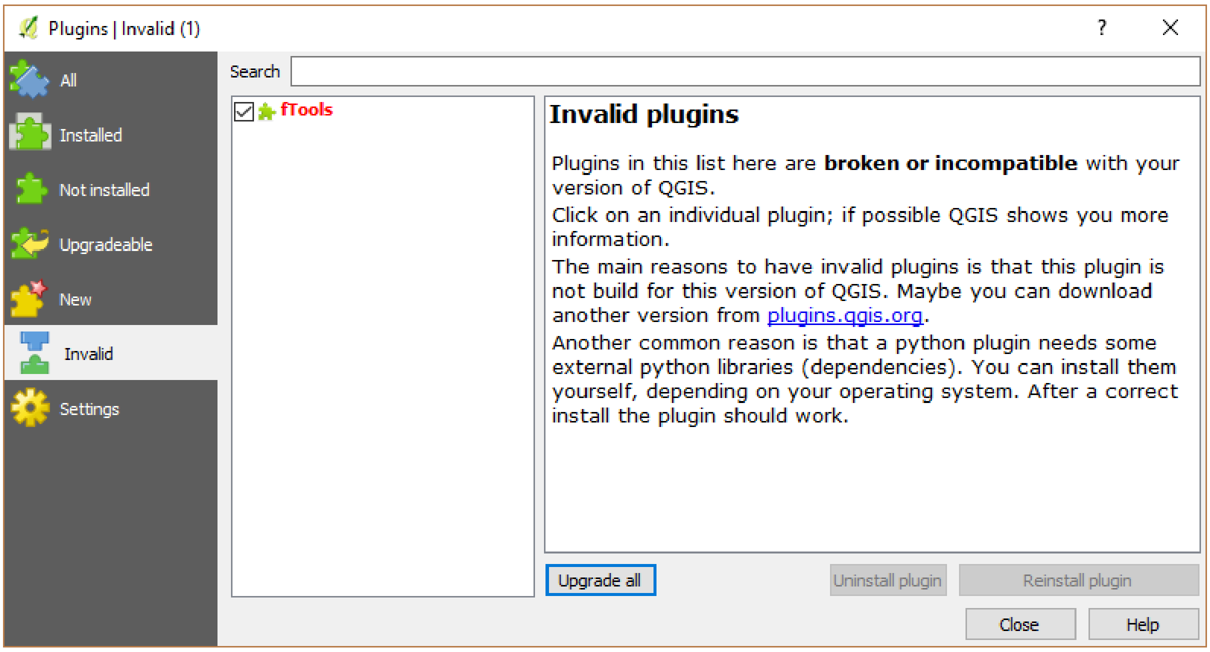


Abb. 24.6: The *Invalid* tab

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.

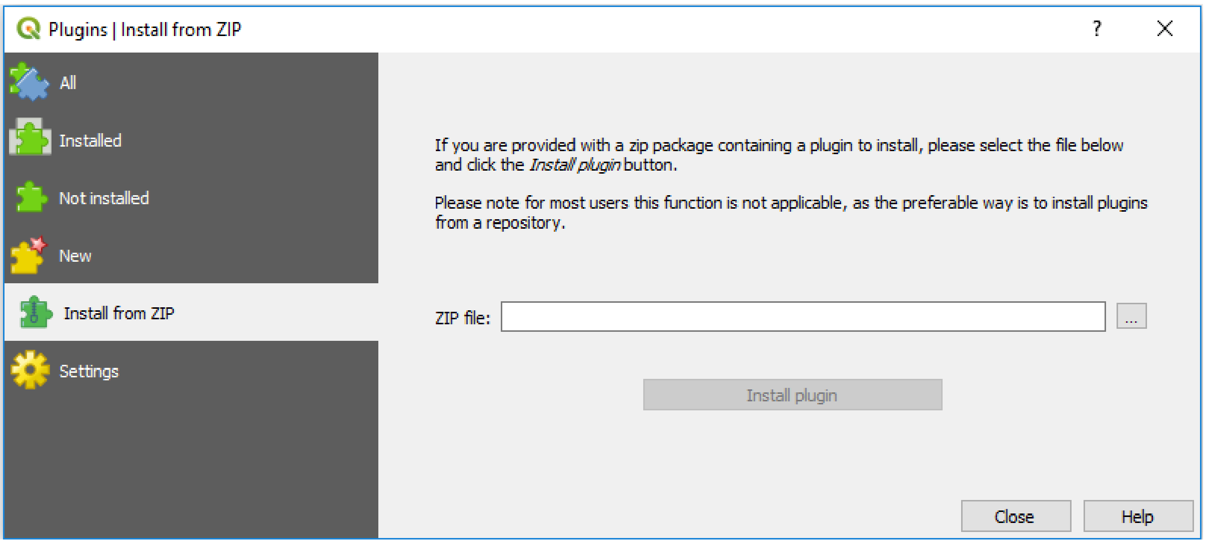


Abb. 24.7: The *Install from zip* tab

24.2 Using QGIS Core Plugins

24.2.1 Coordinate Capture Plugin

The coordinate capture plugin is easy to use and provides the ability to display coordinates on the map canvas for two selected coordinate reference systems (CRS).

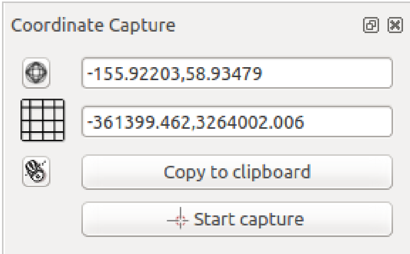


Abb. 24.8: Coordinate Capture Plugin

1. Start QGIS, select *Properties…* from the *Project* menu and click on the *CRS* tab. As an alternative, you can also click on the CRS icon in the lower right-hand corner of the status bar.
2. Select a projected coordinate system of your choice (see also *Arbeiten mit Projektionen*).
3. Activate the coordinate capture plugin in the Plugin Manager (see *Der Erweiterungen Dialog*) and ensure that the dialog is visible by going to *View ⍰ Panels* and ensuring that *Coordinate Capture* is enabled. The coordinate capture dialog appears as shown in Figure *figure_coordinate_capture*. Alternatively, you can also look for *Vector ⍰ Coordinate Capture*.
4. Click on the Click to the select the CRS to use for coordinate display icon and select a different CRS from the one you selected above.
5. To start capturing coordinates, click on *Start Capture*. You can now click anywhere on the map canvas and the plugin will show the coordinates for both of your selected CRS.
6. To enable mouse coordinate tracking, click the mouse tracking icon.
7. You can also copy selected coordinates to the clipboard.

24.2.2 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.

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…*.

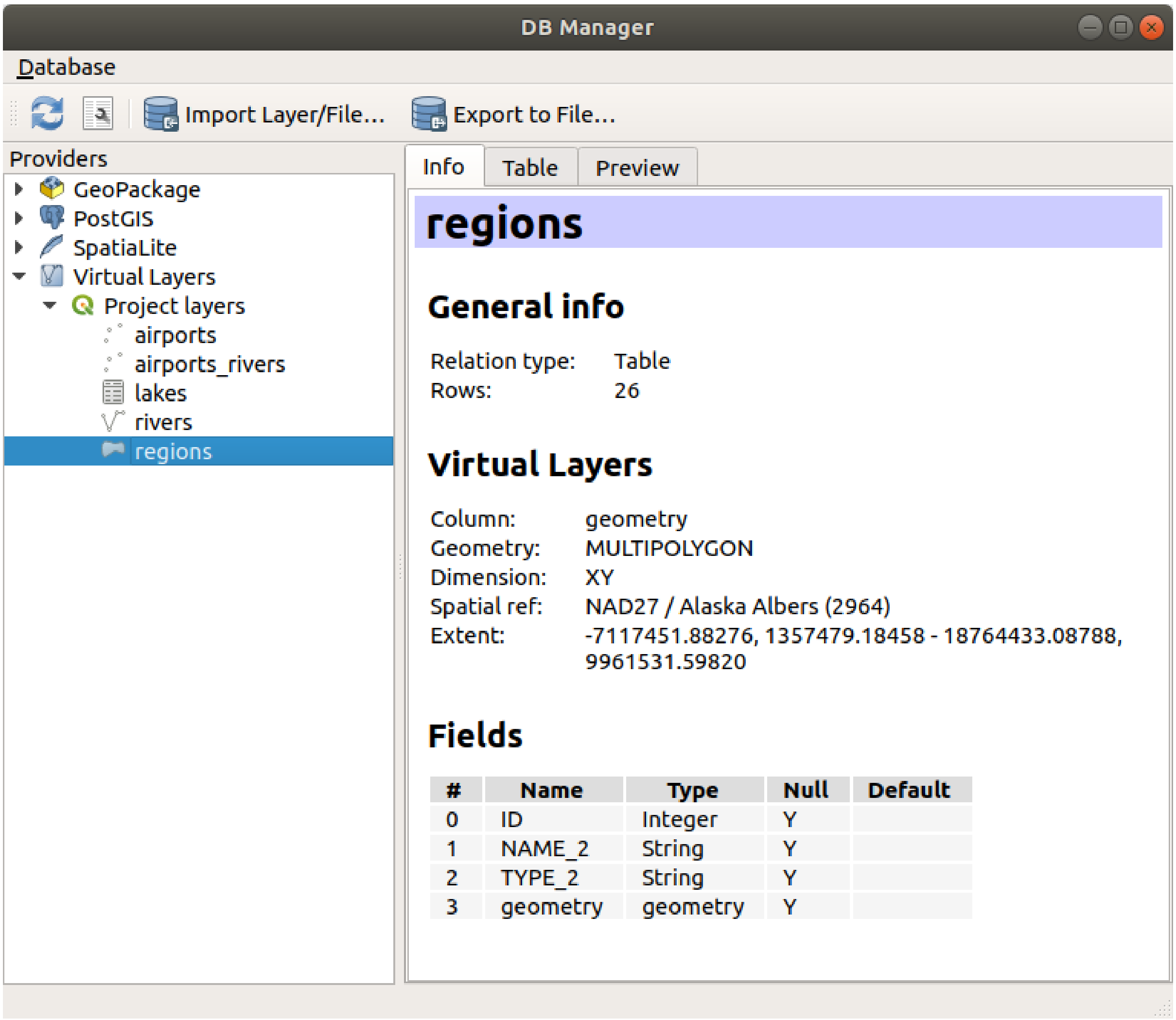


Abb. 24.9: DB Manager dialog

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.

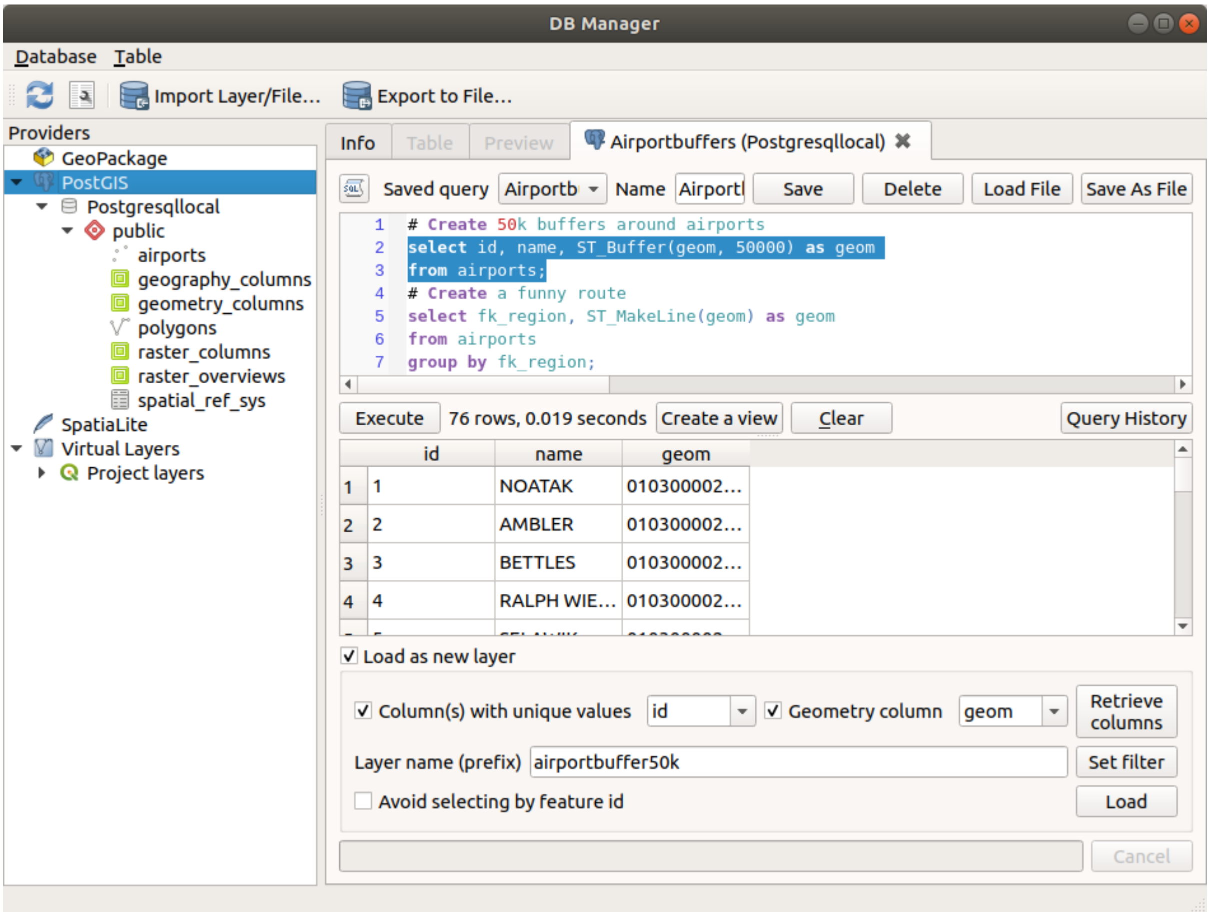


Abb. 24.10: Executing SQL queries in the DB Manager SQL window

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.

24.2.3 eVis Plugin

(This section is derived from Horning, N., K. Koy, P. Ersts. 2009. eVis (v1.1.0) User's Guide. American Museum of Natural History, Center for Biodiversity and Conservation. Available from https://www.amnh.org/research/center-for-biodiversity-conservation/capacity-development/biodiversity-informatics, and released under the GNU FDL.)

The Biodiversity Informatics Facility at the American Museum of Natural History's (AMNH) Center for Biodiversity and Conservation (CBC) has developed the Event Visualization Tool (eVis), another software tool to add to the suite of conservation monitoring and decision support tools for guiding protected area and landscape planning. This plugin enables users to easily link geocoded (i.e., referenced with latitude and longitude or X and Y coordinates) photographs, and other supporting documents, to vector data in QGIS.

eVis is now automatically installed and enabled in new versions of QGIS, and as with all plugins, it can be disabled and enabled using the Plugin Manager (see *Der Erweiterungen Dialog*).

The eVis plugin is made up of three modules: the ‚Database Connection tool', ‚Event ID tool', and the ‚Event Browser'. These work together to allow viewing of geocoded photographs and other documents that are linked to features stored in vector files, databases, or spreadsheets.

Event Browser

The Event Browser module provides the functionality to display geocoded photographs that are linked to vector features displayed in the QGIS map window. Point data, for example, can be from a vector file that can be input using QGIS or it can be from the result of a database query. The vector feature must have attribute information associated with it to describe the location and name of the file containing the photograph and, optionally, the compass direction the camera was pointed when the image was acquired. Your vector layer must be loaded into QGIS before running the Event Browser.

Launch the Event Browser module

To launch the Event Browser module, click on *Database* ▯ *eVis* ▯ *eVis Event Browser*. This will open the *Generic Event Browser* window.

The *Event Browser* window has three tabs displayed at the top of the window. The *Display* tab is used to view the photograph and its associated attribute data. The *Options* tab provides a number of settings that can be adjusted to control the behavior of the eVis plugin. Lastly, the *Configure External Applications* tab is used to maintain a table of file extensions and their associated application to allow eVis to display documents other than images.

Understanding the Display window

To see the *Display* window, click on the *Display* tab in the *Event Browser* window. The *Display* window is used to view geocoded photographs and their associated attribute data.

A. **Display window**: A window where the photograph will appear.

B. **Zoom in button**: Zoom in to see more detail. If the entire image cannot be displayed in the display window, scroll bars will appear on the left and bottom sides of the window to allow you to pan around the image.

C. **Zoom out button**: Zoom out to see more area.

D. **Zoom to full extent** button: Displays the full extent of the photograph.

E. **Attribute information window**: All of the attribute information for the point associated with the photograph being viewed is displayed here. If the file type being referenced in the displayed record is not an image but is of a file type defined in the *Configure External Applications* tab, then when you double-click on the value of the field containing the path to the file, the application to open the file will be launched to view or hear the contents of the file. If the file extension is recognized, the attribute data will be displayed in green.

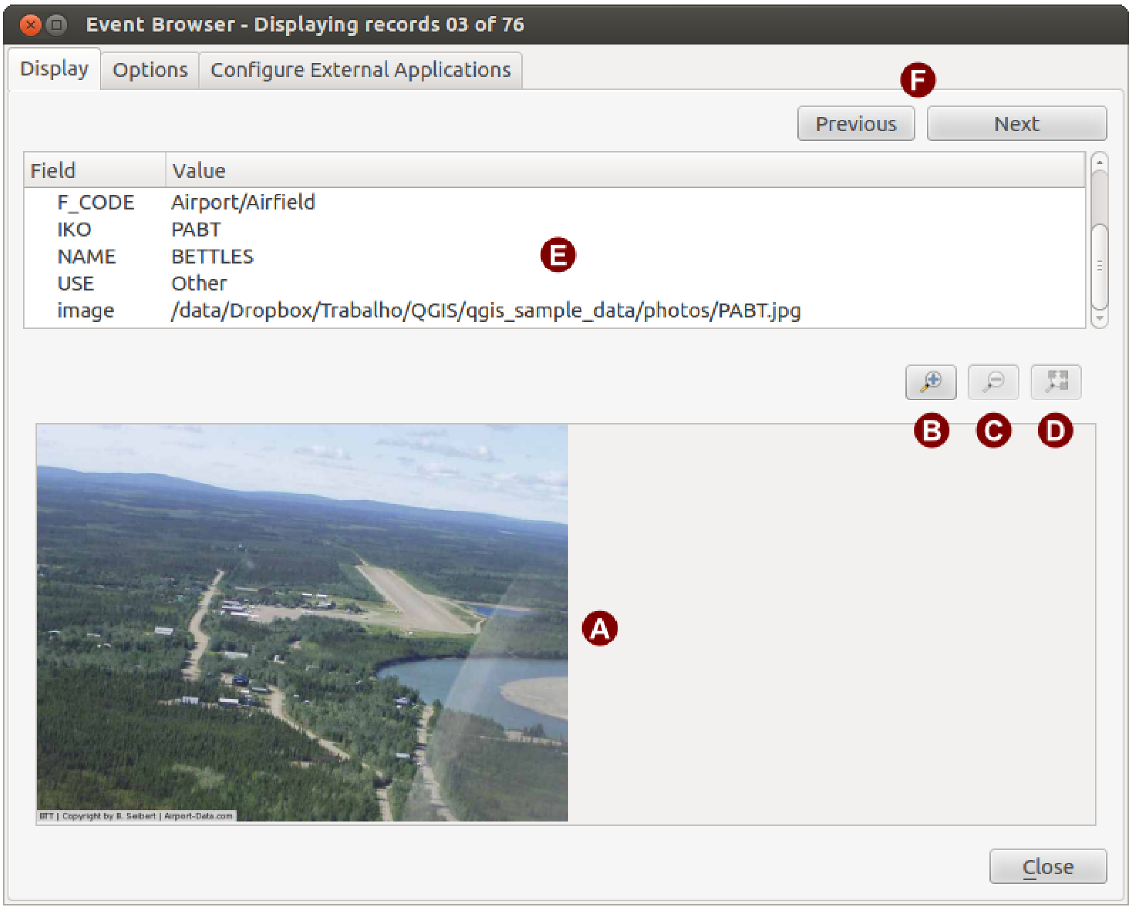


Abb. 24.11: The *eVis* display window

F. **Navigation buttons**: Use the Previous and Next buttons to load the previous or next feature when more than one feature is selected.

Understanding the Options window

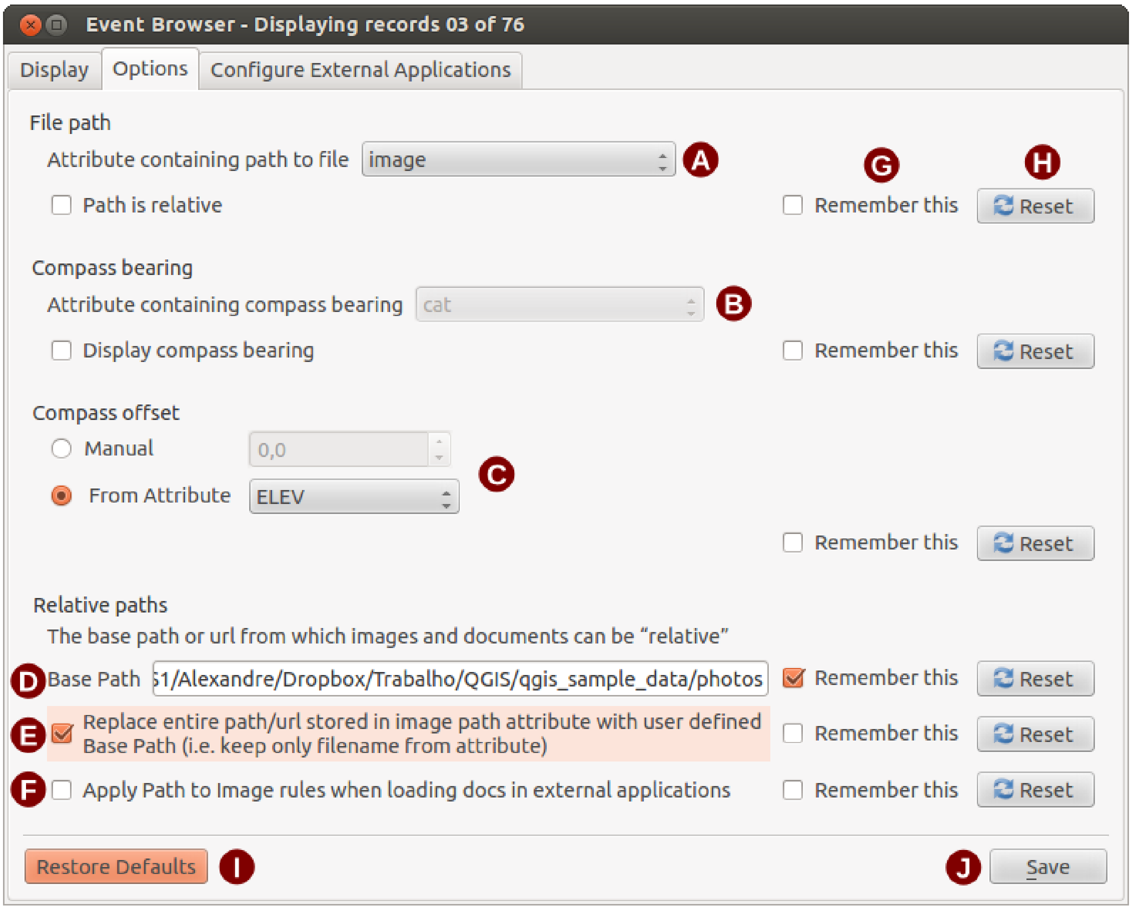


Abb. 24.12: The *eVis* Options window

A. **File path**: A drop-down list to specify the attribute field that contains the directory path or URL for the photographs or other documents being displayed. If the location is a relative path, then the checkbox must be clicked. The base path for a relative path can be entered in the *Base Path* text box below. Information about the different options for specifying the file location are noted in the section *Specifying the location and name of a photograph* below.

B. **Compass bearing**: A drop-down list to specify the attribute field that contains the compass bearing associated with the photograph being displayed. If compass bearing information is available, it is necessary to click the checkbox below the drop-down menu title.

C. **Compass offset**: Compass offsets can be used to compensate for declination (to adjust bearings collected using magnetic bearings to true north bearings). Click the *Manual* radio button to enter the offset in the text box or click the *From Attribute* radio button to select the attribute field containing the offsets. For both of these options, east declinations should be entered using positive values, and west declinations should use negative values.

D. **Directory base path**: The base path onto which the relative path defined in *Figure_eVis_options* (A) will be appended.

E. **Replace path**: If this checkbox is checked, only the file name from A will be appended to the base path.

F. **Apply rule to all documents**: If checked, the same path rules that are defined for photographs will be used for non-image documents such as movies, text documents, and sound files. If not checked, the path rules will only apply to photographs, and other documents will ignore the base path parameter.

G. **Remember settings**: If the checkbox is checked, the values for the associated parameters will be saved for the next session when the window is closed or when the *Save* button below is pressed.

H. **Reset values**: Resets the values on this line to the default setting.

I. **Restore defaults**: This will reset all of the fields to their default settings. It has the same effect as clicking all of the *Reset* buttons.

J. **Save**: This will save the settings without closing the *Options* pane.

Understanding the Configure External Applications window

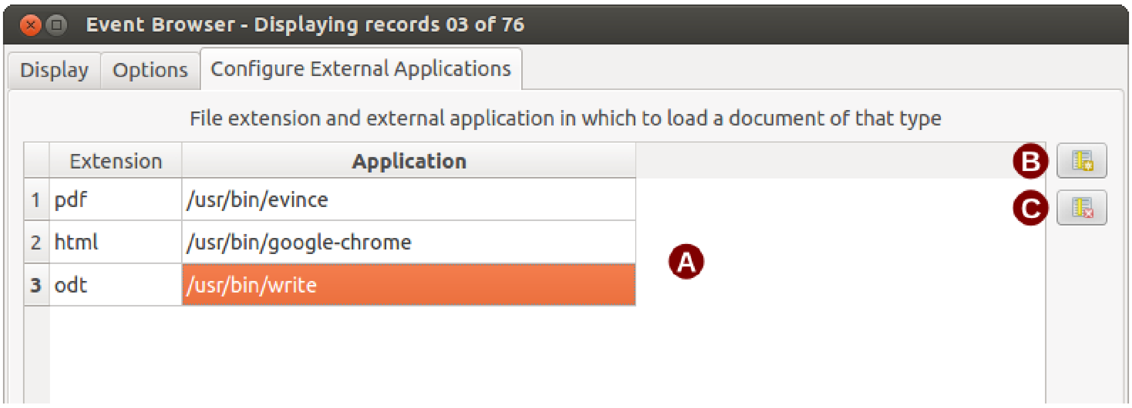


Abb. 24.13: The *eVis* External Applications window

A. **File reference table**: A table containing file types that can be opened using eVis. Each file type needs a file extension and the path to an application that can open that type of file. This provides the capability of opening a broad range of files such as movies, sound recordings, and text documents instead of only images.

B. **Add new file type**: Add a new file type with a unique extension and the path for the application that can open the file.

C. **Delete current row**: Delete the file type highlighted in the table and defined by a file extension and a path to an associated application.

Specifying the location and name of a photograph

The location and name of the photograph can be stored using an absolute or relative path, or a URL if the photograph is available on a web server. Examples of the different approaches are listed in Table *evis_examples*.

```
X         Y         FILE                                              BEARING
780596    1784017   C:\Workshop\eVis_Data\groundphotos\DSC_0168.JPG   275
780596    1784017   /groundphotos/DSC_0169.JPG                        80
780819    1784015   https://biodiversityinformatics.amnh.org/\
                    evis_testdata/DSC_0170.JPG                        10
780596    1784017   pdf:https://www.testsite.com/attachments.php?\
                    attachment_id-12                                  76
```

Specifying the location and name of other supporting documents

Supporting documents such as text documents, videos, and sound clips can also be displayed or played by eVis. To do this, it is necessary to add an entry in the file reference table that can be accessed from the *Configure External Applications* window in the *Generic Event Browser* that matches the file extension to an application that can be used to open the file. It is also necessary to have the path or URL to the file in the attribute table for the vector layer. One additional rule that can be used for URLs that don't contain a file extension for the document you want to open is to specify the file extension before the URL. The format is — `file extension:URL`. The URL is preceded by the file extension and a colon; this is particularly useful for accessing documents from wikis and other web sites that use a database to manage the web pages (see Table *evis_examples*).

Using the Event Browser

When the *Event Browser* window opens, a photograph will appear in the display window if the document referenced in the vector file attribute table is an image and if the file location information in the *Options* window is properly set. If a photograph is expected and it does not appear, it will be necessary to adjust the parameters in the *Options* window.

If a supporting document (or an image that does not have a file extension recognized by eVis) is referenced in the attribute table, the field containing the file path will be highlighted in green in the attribute information window if that file extension is defined in the file reference table located in the *Configure External Applications* window. To open the document, double-click on the green-highlighted line in the attribute information window. If a supporting document is referenced in the attribute information window and the file path is not highlighted in green, then it will be necessary to add an entry for the file's filename extension in the *Configure External Applications* window. If the file path is highlighted in green but does not open when double-clicked, it will be necessary to adjust the parameters in the *Options* window so the file can be located by eVis.

If no compass bearing is provided in the *Options* window, a red asterisk will be displayed on top of the vector feature that is associated with the photograph being displayed. If a compass bearing is provided, then an arrow will appear pointing in the direction indicated by the value in the compass bearing display field in the *Event Browser* window. The arrow will be centered over the point that is associated with the photograph or other document.

To close the *Event Browser* window, click on the *Close* button from the *Display* window.

Event ID Tool

The ‚Event ID' module allows you to display a photograph by clicking on a feature displayed in the QGIS map window. The vector feature must have attribute information associated with it to describe the location and name of the file containing the photograph and, optionally, the compass direction the camera was pointed when the image was acquired. This layer must be loaded into QGIS before running the ‚Event ID' tool.

Launch the Event ID module

To launch the ‚Event ID' module, either click on the Event ID icon or click on *Database ▯ eVis ▯ Event ID Tool*. This will cause the cursor to change to an arrow with an ‚i' on top of it signifying that the ID tool is active.

To view the photographs linked to vector features in the active vector layer displayed in the QGIS map window, move the Event ID cursor over the feature and then click the mouse. After clicking on the feature, the *Event Browser* window is opened and the photographs on or near the clicked locality are available for display in the browser. If more than one photograph is available, you can cycle through the different features using the *Previous* and *Next* buttons. The other controls are described in the *Event Browser* section of this guide.

Database connection

The ‚Database Connection' module provides tools to connect to and query a database or other ODBC resource, such as a spreadsheet.

eVis can directly connect to the following types of databases: PostgreSQL, MySQL, and SQLite; it can also read from ODBC connections (e.g., MS Access). When reading from an ODBC database (such as an Excel spreadsheet), it is necessary to configure your ODBC driver for the operating system you are using.

Launch the Database Connection module

To launch the ‚Database Connection' module, either click on the appropriate icon eVis Database Connection or click on *Database ‣ eVis ‣ Database Connection*. This will launch the *Database Connection* window. The window has three tabs: *Predefined Queries*, *Database Connection*, and *SQL Query*. The *Output Console* window at the bottom of the window displays the status of actions initiated by the different sections of this module.

Connect to a database

Click on the *Database Connection* tab to open the database connection interface. Next, use the *Database Type* combo box to select the type of database that you want to connect to. If a password or username is required, that information can be entered in the *Username* and *Password* textboxes.

Enter the database host in the *Database Host* textbox. This option is not available if you selected ‚MS Access' as the database type. If the database resides on your desktop, you should enter „localhost".

Enter the name of the database in the *Database Name* textbox. If you selected ‚ODBC' as the database type, you need to enter the data source name.

When all of the parameters are filled in, click on the *Connect* button. If the connection is successful, a message will be written in the *Output Console* window stating that the connection was established. If a connection was not established, you will need to check that the correct parameters were entered above.

A. **Database Type**: A drop-down list to specify the type of database that will be used.
B. **Database Host**: The name of the database host.
C. **Port**: The port number if a MySQL or PostgreSQL database type is selected.
D. **Database Name**: The name of the database.
E. **Connect**: A button to connect to the database using the parameters defined above.
F. **Output Console**: The console window where messages related to processing are displayed.
G. **Username**: Username for use when a database is password protected.
H. **Password**: Password for use when a database is password protected.
I. **Predefined Queries**: Tab to open the „Predefined Queries" window.
J. **Database Connection**: Tab to open the „Database Connection" window.
K. **SQL Query**: Tab to open the „SQL Query" window.
L. **Help**: Displays the online help.
M. **OK**: Closes the main „Database Connection" window.

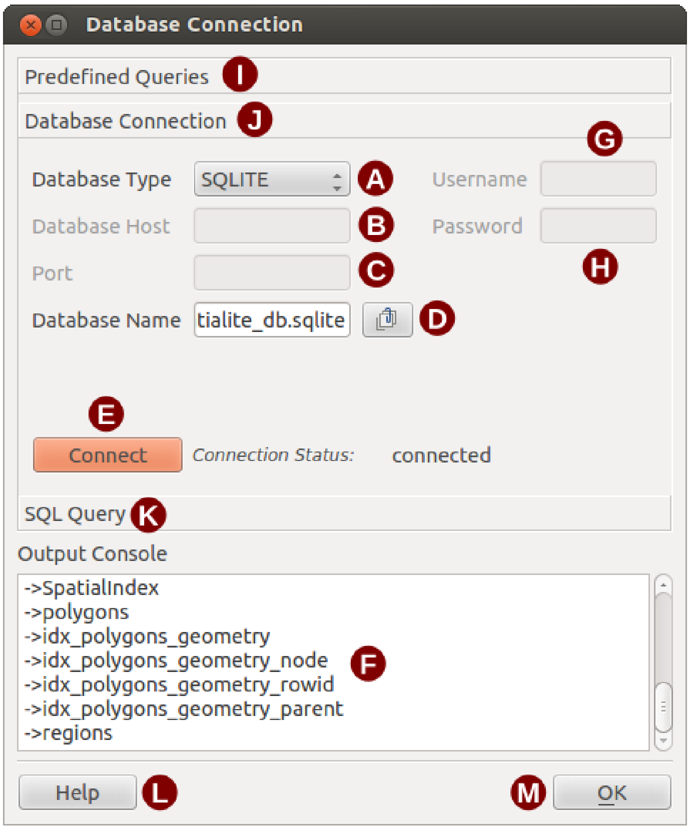


Abb. 24.14: The *eVis* Database connection window

Running SQL queries

SQL queries are used to extract information from a database or ODBC resource. In eVis, the output from these queries is a vector layer added to the QGIS map window. Click on the *SQL Query* tab to display the SQL query interface. SQL commands can be entered in this text window. A helpful tutorial on SQL commands is available at https://www.w3schools.com/sql. For example, to extract all of the data from a worksheet in an Excel file, `select * from [sheet1$]` where `sheet1` is the name of the worksheet.

Click on the *Run Query* button to execute the command. If the query is successful, a *Database File Selection* window will be displayed. If the query is not successful, an error message will appear in the *Output Console* window.

In the *Database File Selection* window, enter the name of the layer that will be created from the results of the query in the *Name of New Layer* textbox.

A. **SQL Query Text Window**: A screen to type SQL queries.

B. **Run Query**: Button to execute the query entered in the *SQL Query Window*.

C. **Console Window**: The console window where messages related to processing are displayed.

D. **Help**: Displays the online help.

E. **OK**: Closes the main *Database Connection* window.

Use the *X Coordinate* and *Y Coordinate* combo boxes to select the fields from the database that stores the `X` (or longitude) and `Y` (or latitude) coordinates. Clicking on the *OK* button causes the vector layer created from the SQL query to be displayed in the QGIS map window.

To save this vector file for future use, you can use the QGIS ‚Save as…' command that is accessed by right-clicking on the layer name in the QGIS map legend and then selecting ‚Save as…'

Tipp: Creating a vector layer from a Microsoft Excel Worksheet

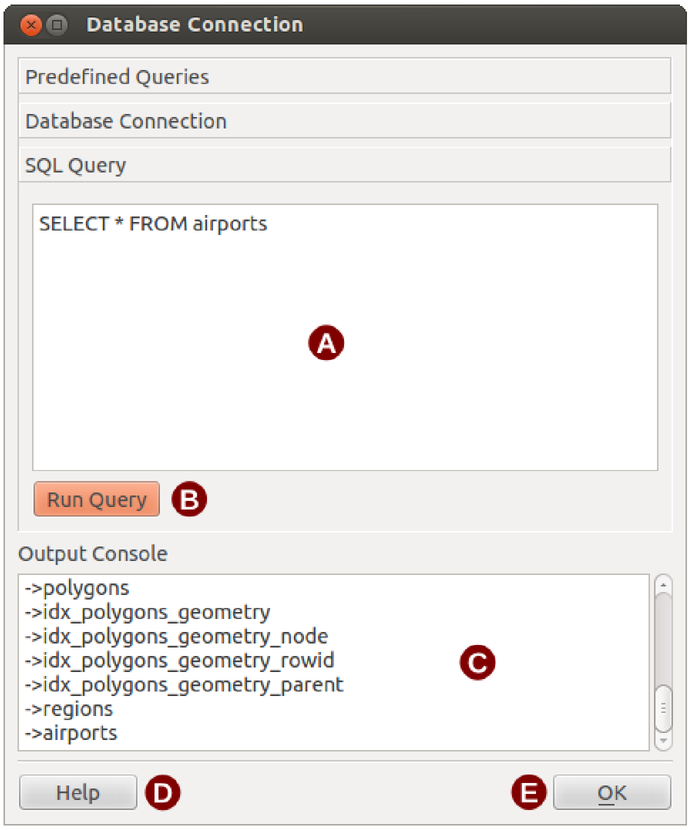


Abb. 24.15: The eVis SQL query tab

When creating a vector layer from a Microsoft Excel Worksheet, you might see that unwanted zeros („0“) have been inserted in the attribute table rows beneath valid data. This can be caused by deleting the values for these cells in Excel using the `Backspace` key. To correct this problem, you need to open the Excel file (you'll need to close QGIS if you are connected to the file, to allow you to edit the file) and then use *Edit ⍰ Delete* to remove the blank rows from the file. To avoid this problem, you can simply delete several rows in the Excel Worksheet using *Edit ⍰ Delete* before saving the file.

Running predefined queries

With predefined queries, you can select previously written queries stored in XML format in a file. This is particularly helpful if you are not familiar with SQL commands. Click on the *Predefined Queries* tab to display the predefined query interface.

To load a set of predefined queries, click on the Open File icon. This opens the *Open File* window, which is used to locate the file containing the SQL queries. When the queries are loaded, their titles as defined in the XML file will appear in the drop-down menu located just below the Open File icon. The full description of the query is displayed in the text window under the drop-down menu.

Select the query you want to run from the drop-down menu and then click on the *SQL Query* tab to see that the query has been loaded into the query window. If it is the first time you are running a predefined query or are switching databases, you need to be sure to connect to the database.

Click on the *Run Query* button in the *SQL Query* tab to execute the command. If the query is successful, a *Database File Selection* window will be displayed. If the query is not successful, an error message will appear in the *Output Console* window.

A. **Open File**: Launches the „Open File“ file browser to search for the XML file holding the predefined queries.

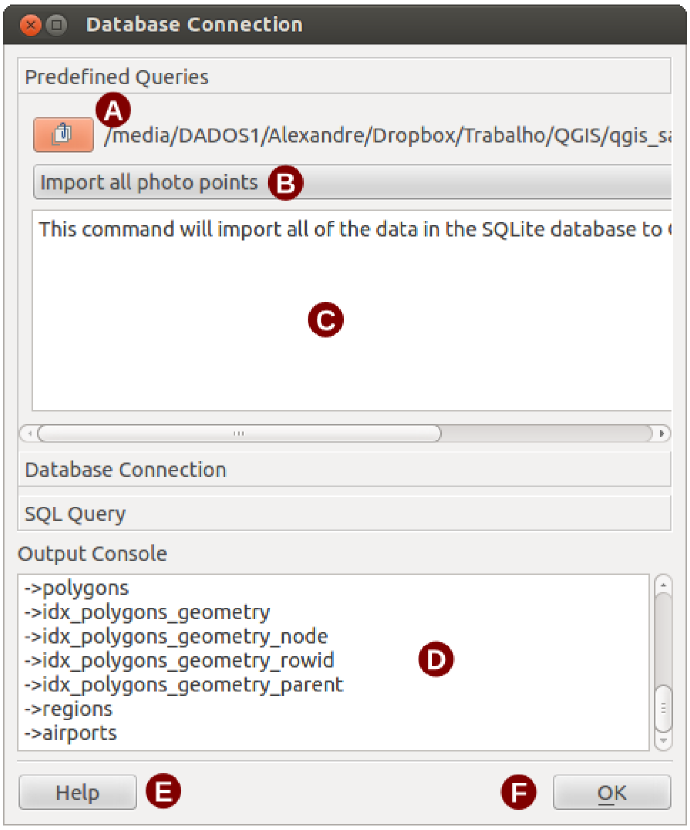


Abb. 24.16: The *eVis* Predefined Queries tab

B. **Predefined Queries**: A drop-down list with all of the queries defined by the predefined queries XML file.

C. **Query description**: A short description of the query. This description is from the predefined queries XML file.

D. **Console Window**: The console window where messages related to processing are displayed.

E. **Help**: Displays the online help.

F. **OK**: Closes the main „Database Connection“ window.

XML format for eVis predefined queries

The XML tags read by eVis

Tag	Beschreibung
query	Defines the beginning and end of a query statement.
shortdescription	A short description of the query that appears in the eVis drop-down menu.
Beschreibung	A more detailed description of the query displayed in the Predefined Query text window.
databasetype	The database type, defined in the Database Type drop-down menu in the Database Connection tab.
databaseport	The port as defined in the Port text box in the Database Connection tab.
databasename	The database name as defined in the Database Name text box in the Database Connection tab.
databaseusername	The database username as defined in the Username text box in the Database Connection tab.
databasepassword	The database password as defined in the Password text box in the Database Connection tab.
sqlstatement	The SQL command.
autoconnect	A flag („true“ or „false“) to specify if the above tags should be used to automatically connect to the database without running the database connection routine in the Database Connection tab.

A complete sample XML file with three queries is displayed below:

```
<?xml version="1.0"?>
<doc>
 <query>
   <shortdescription>Import all photograph points</shortdescription>
   <description>This command will import all of the data in the SQLite database to␣
↪QGIS
      </description>
   <databasetype>SQLITE</databasetype>
   <databasehost />
   <databaseport />
   <databasename>C:\textbackslash Workshop/textbackslash
eVis\_Data\textbackslash PhotoPoints.db</databasename>
   <databaseusername />
   <databasepassword />
   <sqlstatement>SELECT Attributes.*, Points.x, Points.y FROM Attributes LEFT JOIN
      Points ON Points.rec_id=Attributes.point_ID</sqlstatement>
   <autoconnect>false</autoconnect>
 </query>
  <query>
   <shortdescription>Import photograph points "looking across Valley"</
↪shortdescription>
   <description>This command will import only points that have photographs
↪"looking across
      a valley" to QGIS</description>
   <databasetype>SQLITE</databasetype>
   <databasehost />
   <databaseport />
   <databasename>C:\Workshop\eVis_Data\PhotoPoints.db</databasename>
   <databaseusername />
   <databasepassword />
   <sqlstatement>SELECT Attributes.*, Points.x, Points.y FROM Attributes LEFT JOIN
      Points ON Points.rec_id=Attributes.point_ID where COMMENTS='Looking across
      valley'</sqlstatement>
   <autoconnect>false</autoconnect>
 </query>
 <query>
   <shortdescription>Import photograph points that mention "limestone"</
↪shortdescription>
   <description>This command will import only points that have photographs that␣
↪mention
      "limestone" to QGIS</description>
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    <databasetype>SQLITE</databasetype>
    <databasehost />
    <databaseport />
    <databasename>C:\Workshop\eVis_Data\PhotoPoints.db</databasename>
    <databaseusername />
    <databasepassword />
    <sqlstatement>SELECT Attributes.*, Points.x, Points.y FROM Attributes LEFT JOIN
       Points ON Points.rec_id=Attributes.point_ID where COMMENTS like '%limestone%'
       </sqlstatement>
    <autoconnect>false</autoconnect>
  </query>
</doc>
```

24.2.4 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*: to allow only some geometry types like point, multipoint, line, multiline, polygon and multipolygon.
- *Geometry validity*: depending on geometry types, the user can choose *Self intersections*, *Duplicate nodes*, *Self contacts* and *Polygon with less than 3 nodes*.
- *Geometry properties*: depending on geometry types, the user can choose *Polygons and multipolygons may not contain any holes*, *Multipart objects must consist of more than one part* and *Lines must not have dangles*.
- *Geometry conditions*: user can add some condition to validate the geometries with a minimal segment length, a minimum angle between segment, a minimal polygon area and sliver polygons detection.
- *Topology checks*: depending on geometry types, the user can choose *Checks for duplicates*, *Checks for features within other features*, *Checks for overlaps smaller than* 1,00, *Checks for gaps smaller than* 1,00, *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 the user how get the result between modify the current layer and create a new layer.

When you are happy with the configuration, you can click on the *Run* button.

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;

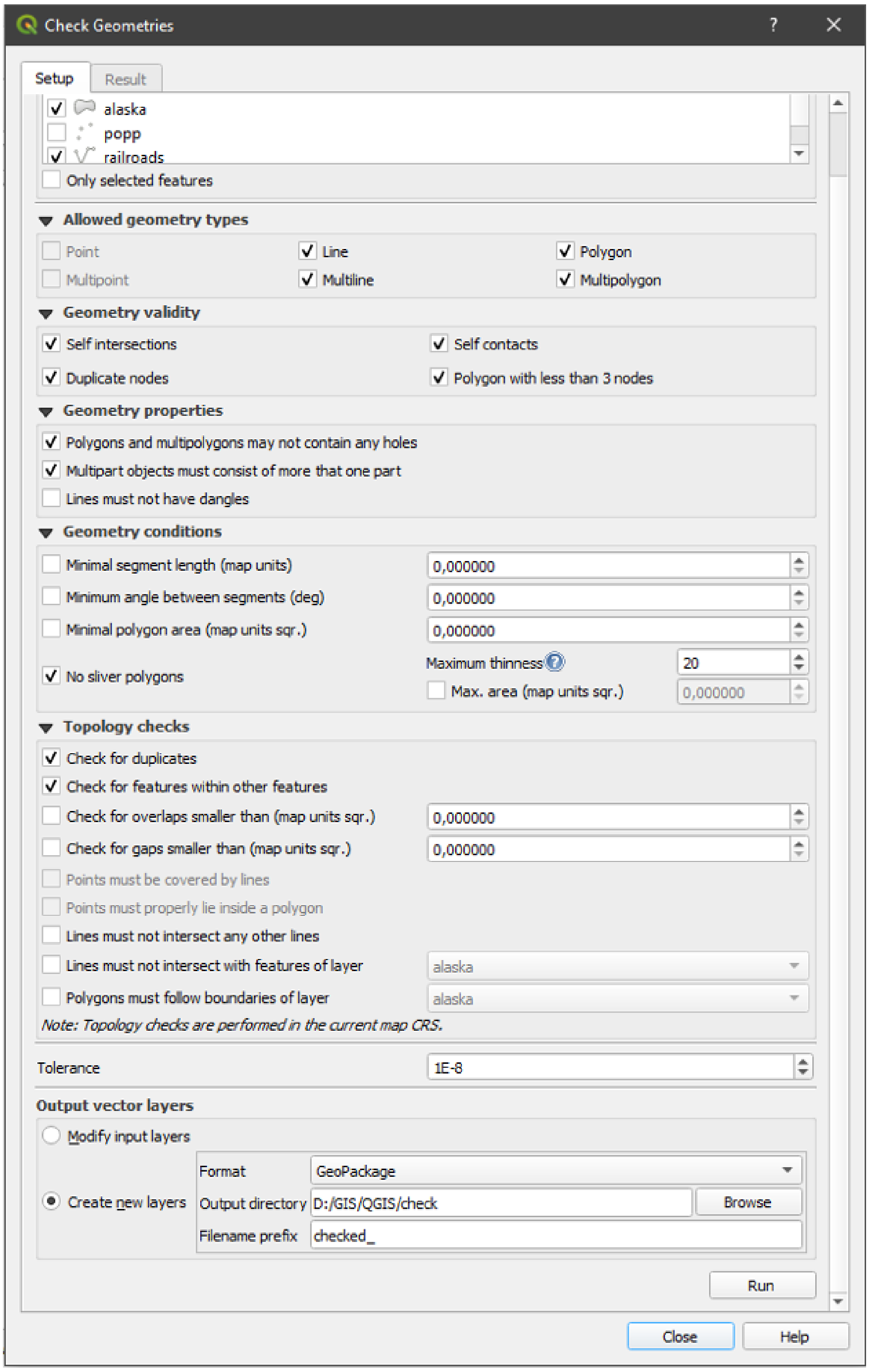


Abb. 24.17: The Geometry Checker Plugin

- 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.

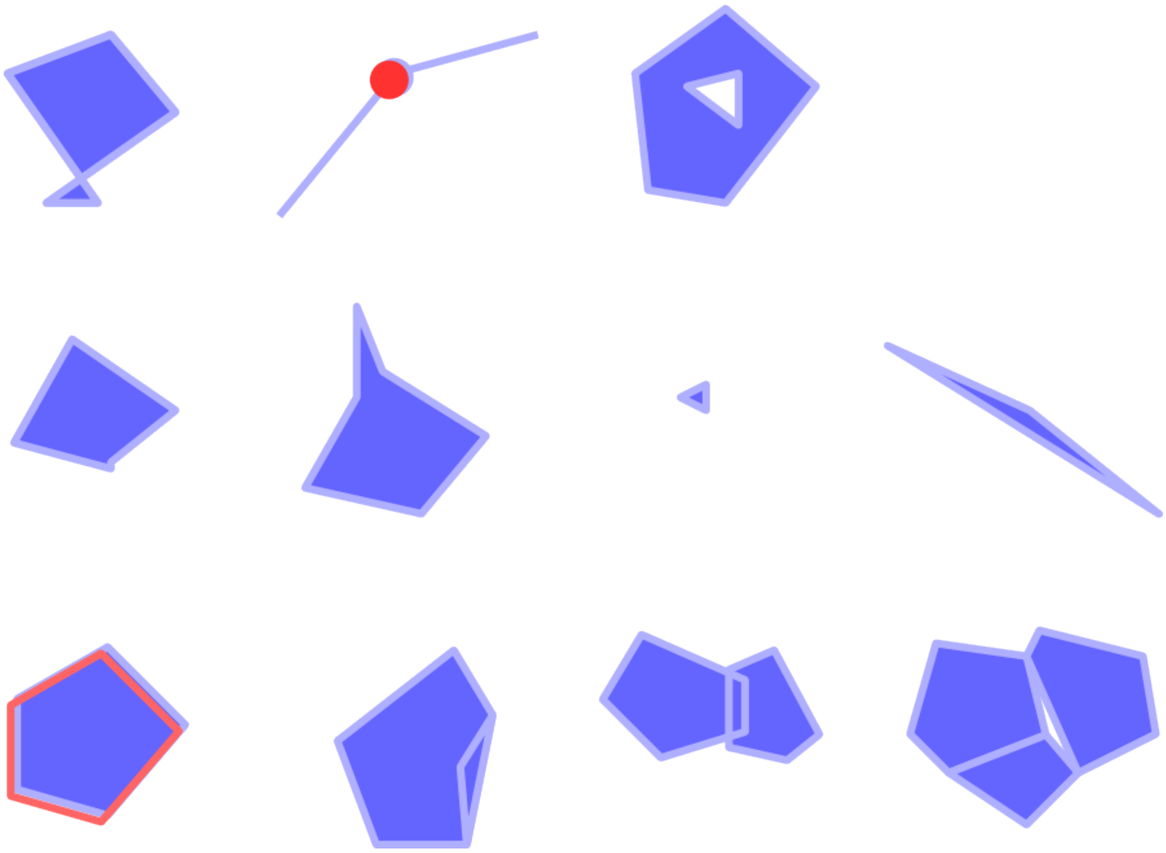


Abb. 24.18: Some checks supported by the plugin

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 contour of 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.

Tipp: Fix multiple errors

You can fix multiple errors by selecting more than one row in the table with the *CTRL + click* action.

The default action can be changed with the last icon *Error resolution settings* For some type of errors, you can change the default action between some specific action or *No action*.

Finally, you can choose which *attribute to use when merging features by attribute value*.

24.2.5 Plugin „Georeferenzierung"

Das Plugin „Georeferenzierung" erlaubt es, die Lage und Projektion von bestehende Rasterdaten anzupassen. Dazu kann zum einen ein Worldfile erstellt werden, zum anderen ist es möglich, für jede Art von Koordiantensystemen einen neuen Rasteratensatz im GeoTiff-Format zu erstellen. Die Grundlage der Georeferenzierung besteht darin, Bezugspunkte auf der Rasterkarte zu finden, denen eindeutige Koordinaten zugewiesen werden können („Passpunkte").

Funktionen

Symbol	Funktion	Symbol	Funktion
	Raster öffnen		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

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 plugin will compute the world file parameters. The more coordinates you provide, the better the result will be.

The first step is to start QGIS, load the Georeferencer Plugin (see *Der Erweiterungen Dialog*) and click on *Raster* *Georeferencer*, which appears in the QGIS menu bar. The Georeferencer Plugin dialog appears as shown in *figure_georeferencer_dialog*.

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/sampledata/spearfish_toposheet.tar.gz.

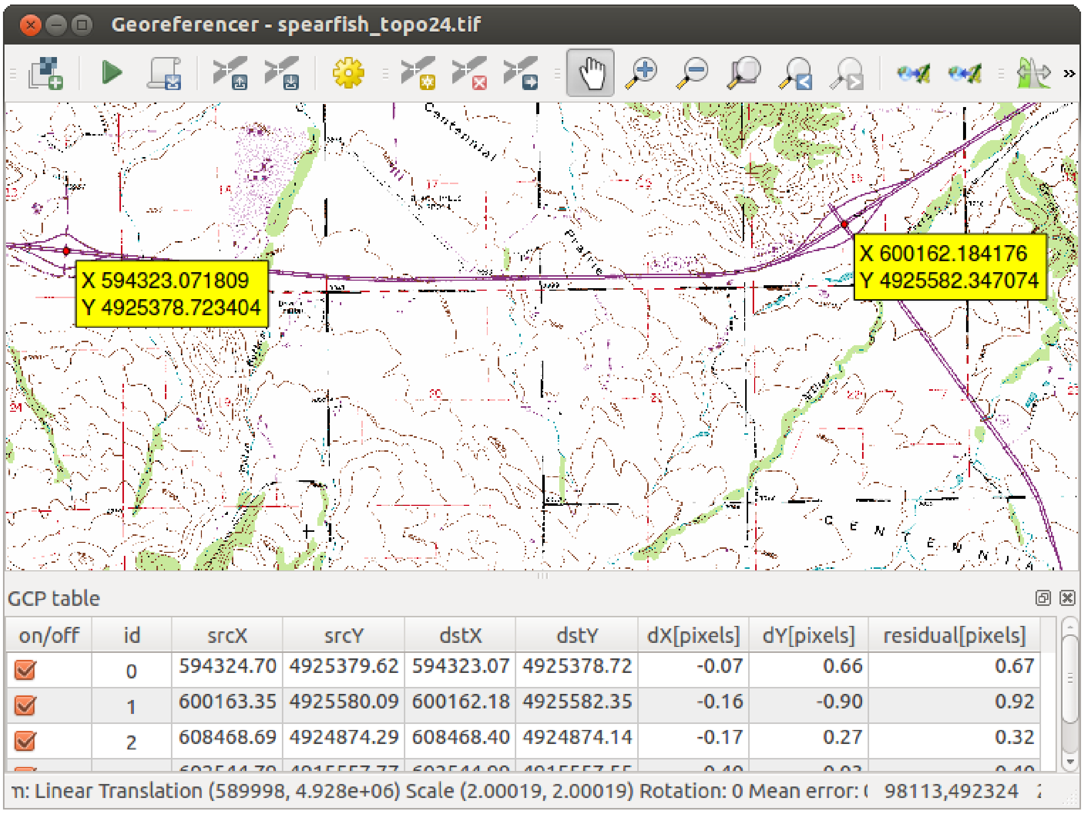


Abb. 24.19: Georeferencer Plugin Dialog

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 *figure_georeferencer_add_points*). For this procedure you have three options:
   - Click on a point in the raster image and enter the X and Y coordinates manually.
   - 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.
   - With the button, you can move the GCPs in both windows, if they are at the wrong place.
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 on the plugin dialog to zoom and pan the working area in order to locate a relevant set of GCP points.

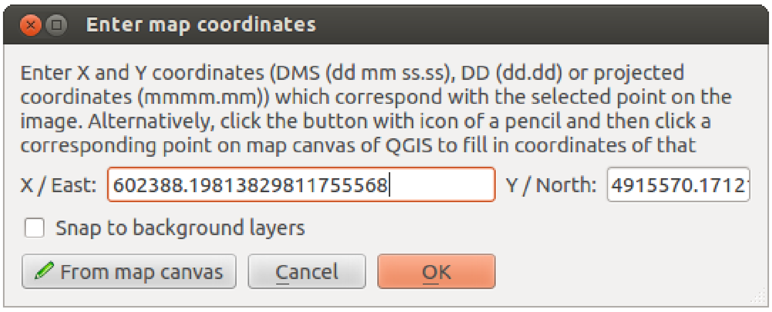


Abb. 24.20: Add points to the raster image

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 plugin 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.

Available Transformation algorithms

Depending on how many ground control points you have captured, you may want to use different transformation algorithms. Choice of transformation algorithm is also dependent on the type and quality of input data and the amount of geometric distortion that you are willing to introduce to the final result.

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. This algorithm likely won't be sufficient if you are dealing with scanned material.
- The **Helmert** transformation performs simple scaling and rotation transformations.
- The **Polynomial** algorithms 1-3 are among the most widely used algorithms introduced to match source and destination ground control points. The most widely used polynomial algorithm is the second-order polyno-

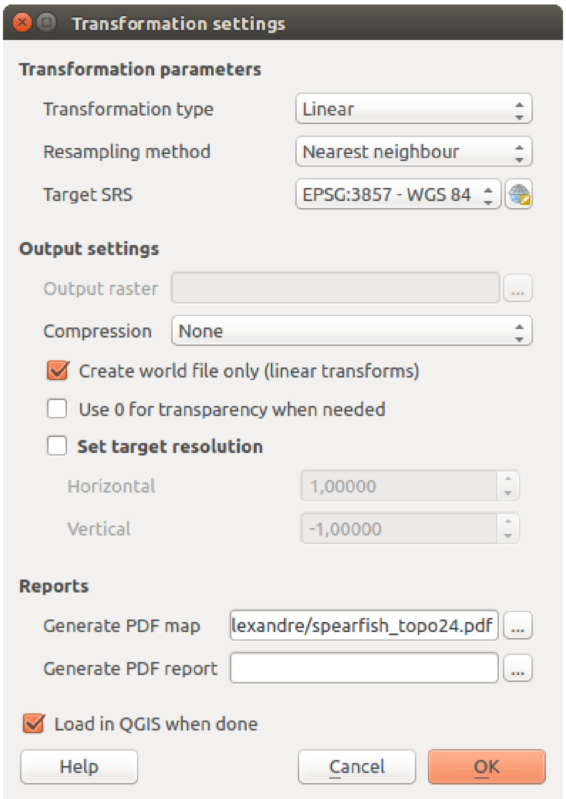


Abb. 24.21: Defining the georeferencer transformation settings

mial transformation, which allows some curvature. First-order polynomial transformation (affine) preserves collinearity and allows scaling, translation and rotation only.

- The **Thin Plate Spline** (TPS) algorithm is a more modern georeferencing method, which is able to introduce local deformations in the data. This algorithm is useful when very low quality originals are being georeferenced.
- The **Projective** transformation is a linear rotation and translation of coordinates.

Define the Resampling method

The type of resampling you choose will likely depending on your input data and the ultimate objective of the exercise. If you don't want to change statistics of the image, you might want to choose ‚Nearest neighbour', whereas a ‚Cubic resampling' will likely provide a more smoothed result.

It is possible to choose between five different resampling methods:

1. Nearest neighbour
2. Linear
3. Kubisch
4. Cubic Spline
5. Lanczos

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 SRS* (Spatial 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.
- Finally, *Load in QGIS 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 define whether you want to show GCP coordinates and/or IDs.
- As residual units, pixels and map units can be chosen.
- For the PDF report, a left and right margin can be defined and you can also set the paper size for the PDF map.
- Finally, you can activate to *Show Georeferencer window docked*.

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.

24.2.6 MetaSearch Catalog Client

Einführung

MetaSearch is a QGIS plugin to interact with metadata catalog services, supporting the OGC Catalog Service for the Web (CSW) standard.

MetaSearch provides an easy and intuitive approach and user-friendly interface to searching metadata catalogs within QGIS.

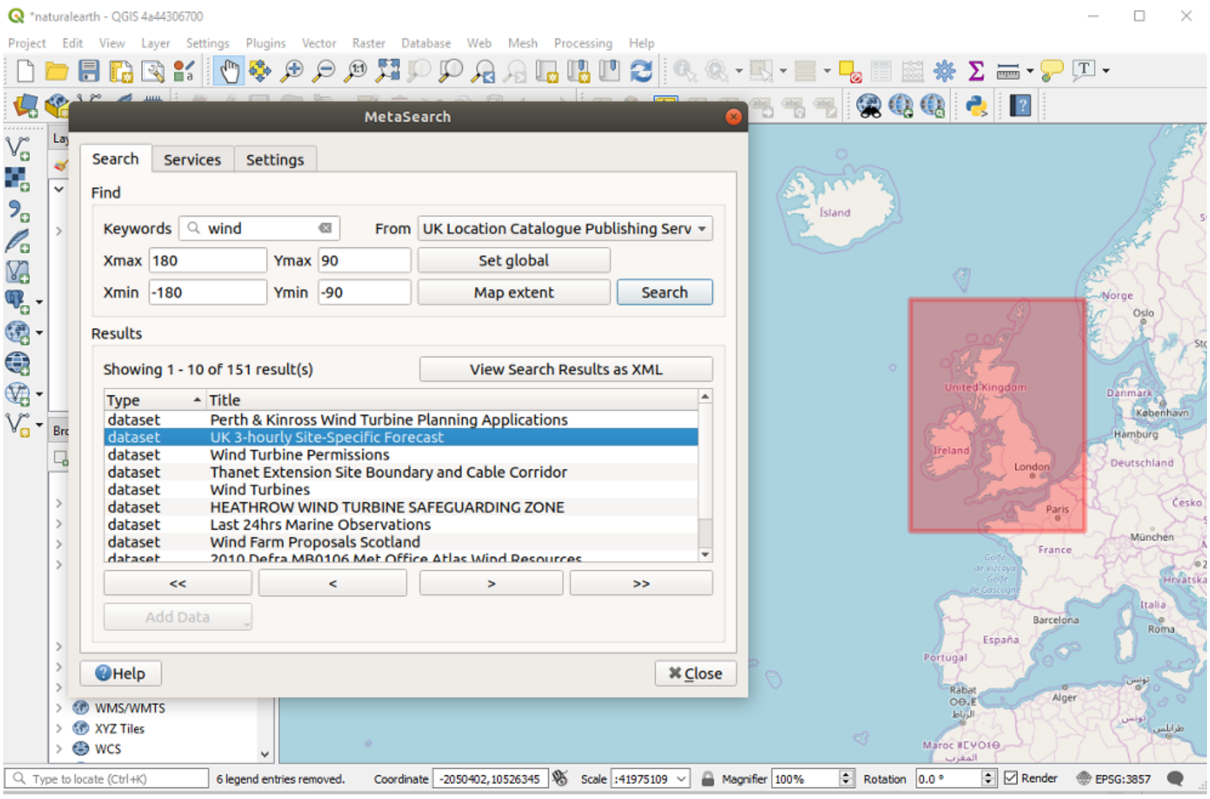


Abb. 24.22: Search and results of Services in MetaSearch

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.

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

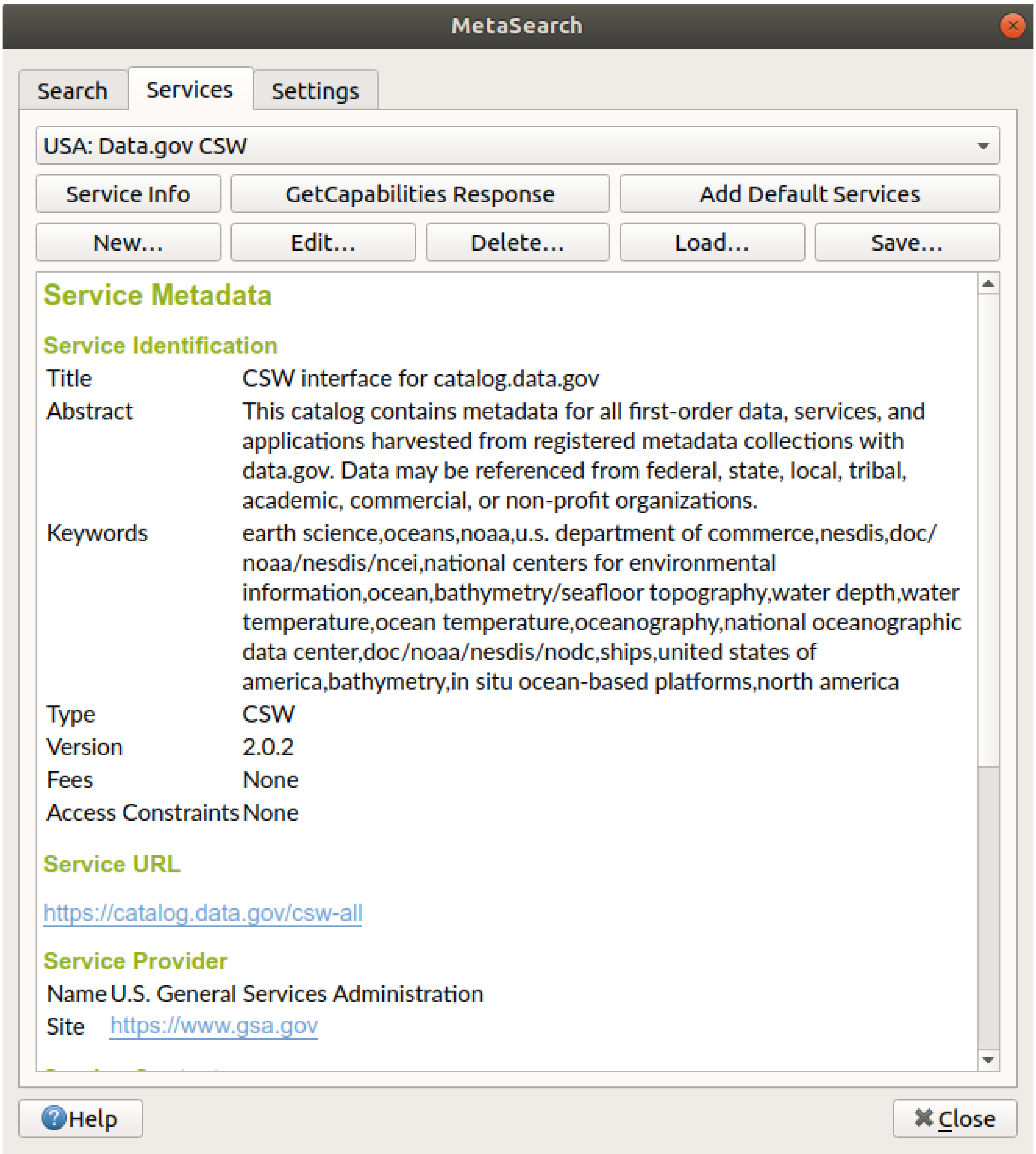


Abb. 24.23: Managing Catalog Services

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 *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 only the base URL is required (not a full GetCapabilities URL).
3. If the CSW 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 for 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 name="Data.gov CSW" url="https://catalog.data.gov/csw-all"/>
    <csw name="Geonorge - National CSW service for Norway" url="https://www.geonorge.no/geonetwork/srv/eng/csw"/>
    <csw name="Geoportale Nazionale - Servizio di ricerca Italiano" url="http://www.pcn.minambiente.it/geoportal/csw"/>
    <csw name="LINZ Data Service" url="http://data.linz.govt.nz/feeds/csw"/>
    <csw name="Nationaal Georegister (Nederland)" url="http://www.nationaalgeoregister.nl/geonetwork/srv/eng/csw"/>
    <csw name="RNDT - Repertorio Nazionale dei Dati Territoriali - Servizio di ricerca" url="http://www.rndt.gov.it/RNDT/CSW"/>
    <csw name="UK Location Catalogue Publishing Service" url="http://csw.data.gov.uk/geonetwork/srv/en/csw"/>
    <csw 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 XML response, click the *GetCapabilities Response* button. A separate window will open displaying Capabilities XML.

Searching Catalog Services

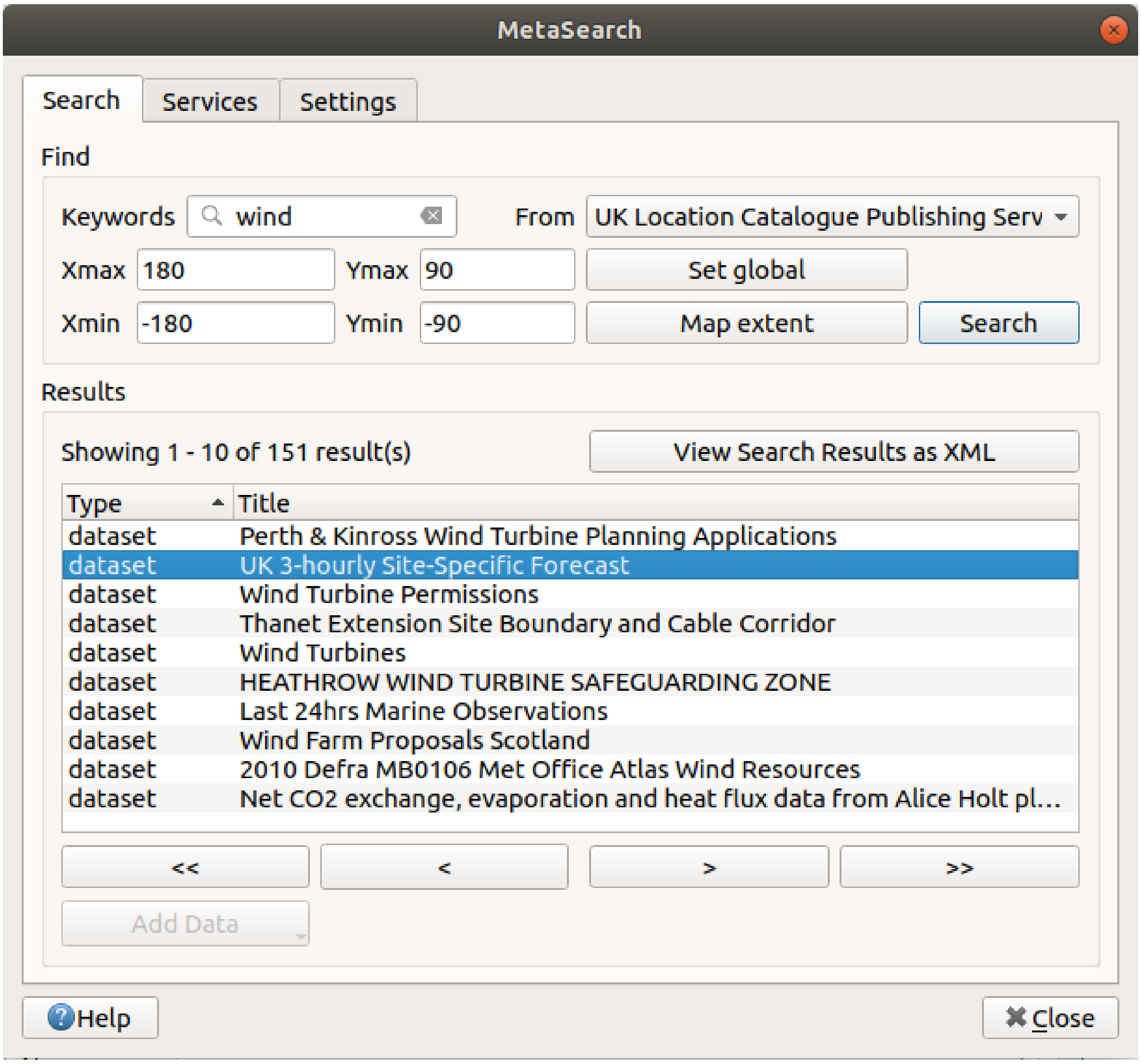


Abb. 24.24: Searching catalog services

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 on defined by *Xmax*, *Xmin*, *Ymax*, and *Ymin*. Click *Set Global* to do a global search, click *Map Extent* to do a search on the visible area only or manually enter custom values as desired.

Clicking the *Search* button will search the selected Metadata Catalog. Search results are displayed in a list and are sortable 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 Search Results as XML* button to open a window with the service response in raw XML format.

- if the metadata record has an associated bounding box, a footprint of the bounding box will be displayed on the map;
- double-clicking the record displays the record metadata with any associated access links. Clicking the links opens the link in the user's web browser;
- if the record is a supported web service (WMS/WMTS, WFS, WCS, ArcGIS MapServer, ArcGIS FeatureServer, etc.), the *Add Data* button will be enabled for the user to add to QGIS. 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 then appear.

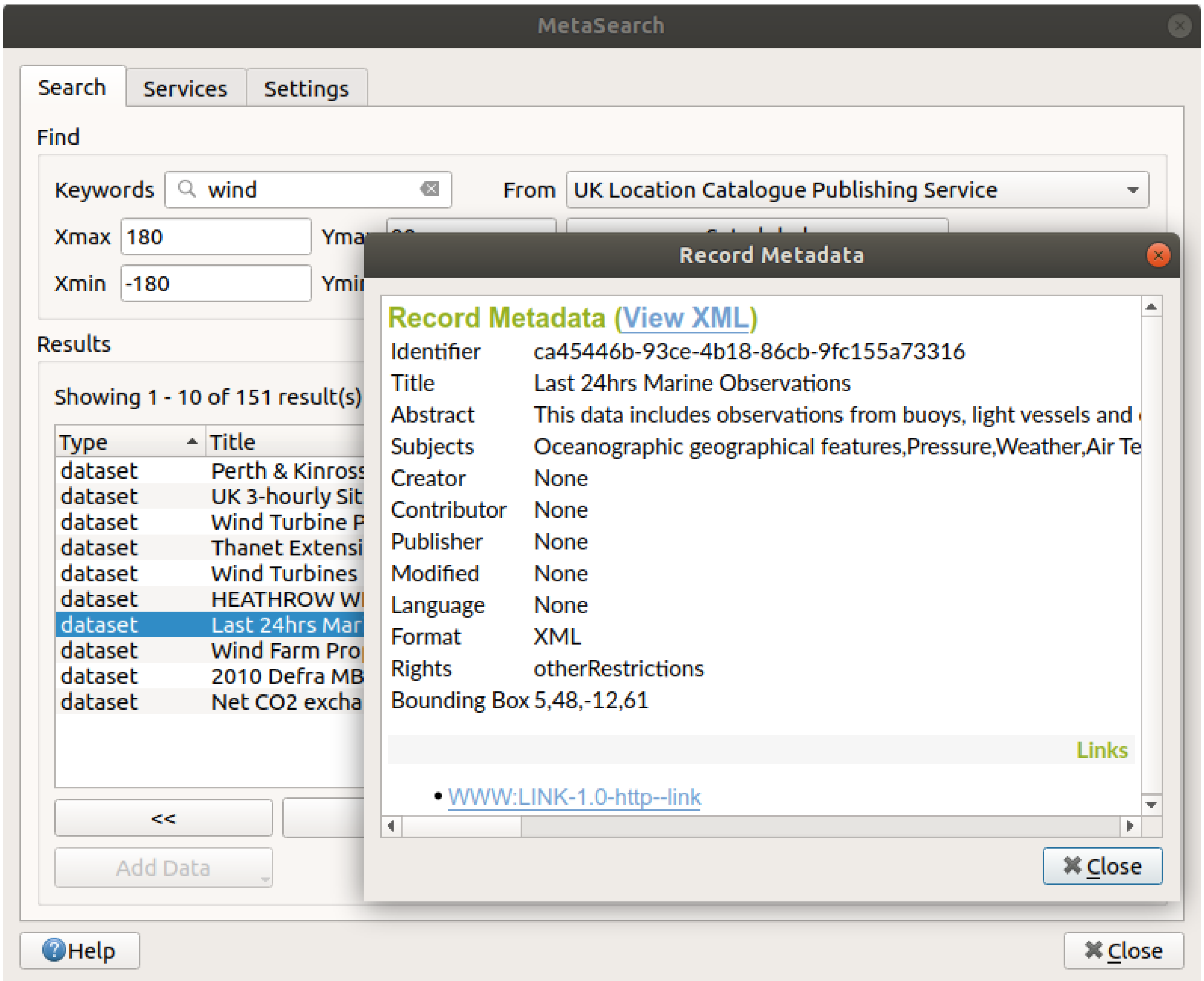


Abb. 24.25: Metadata record display

Einstellungen

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.
- *Results paging*: when searching metadata catalogs, the number of results to show per page. Default value is 10.

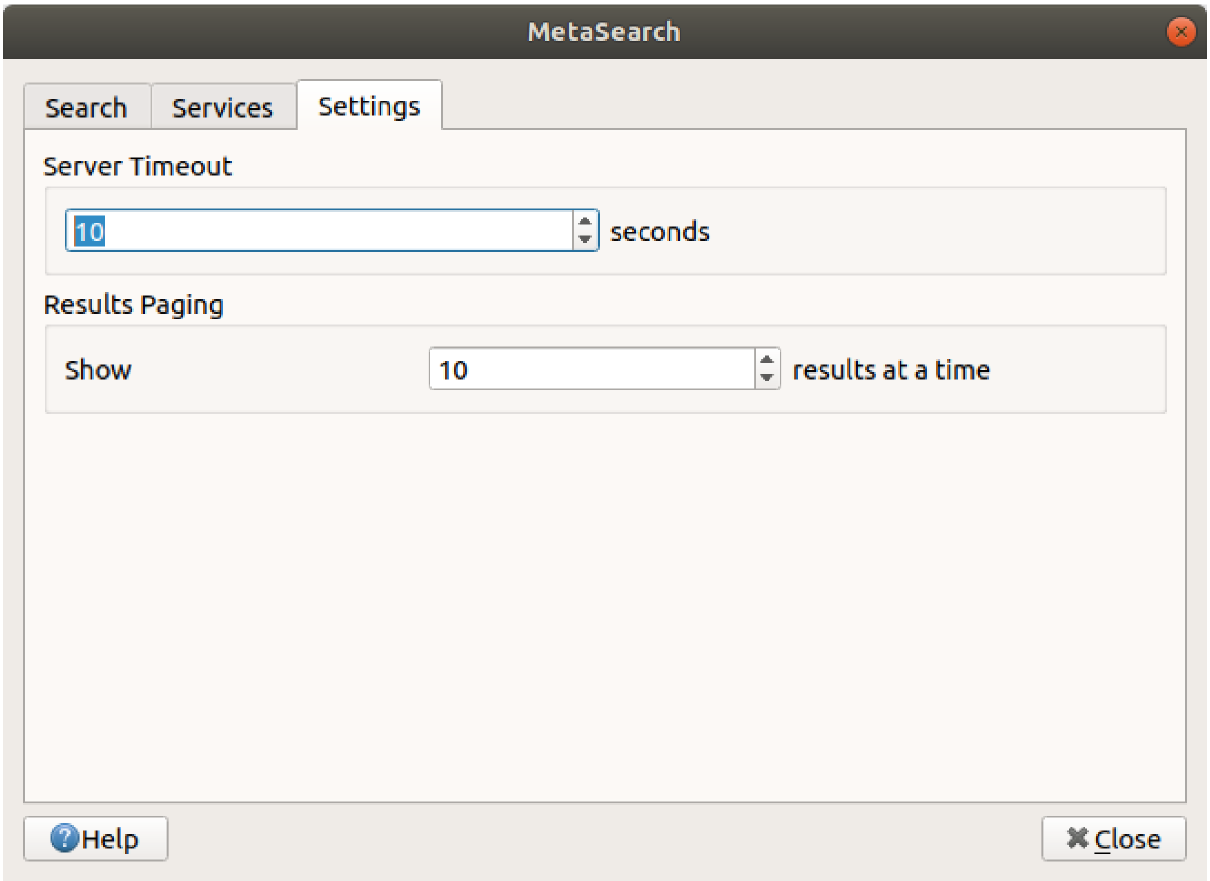


Abb. 24.26: MetaSearch settings

24.2.7 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 a datasource (usually PostGIS or WFS-T) 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 a 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.
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*.

24.2.8 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.

It is very simple to create topology rules with the Topology Checker plugin.

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.

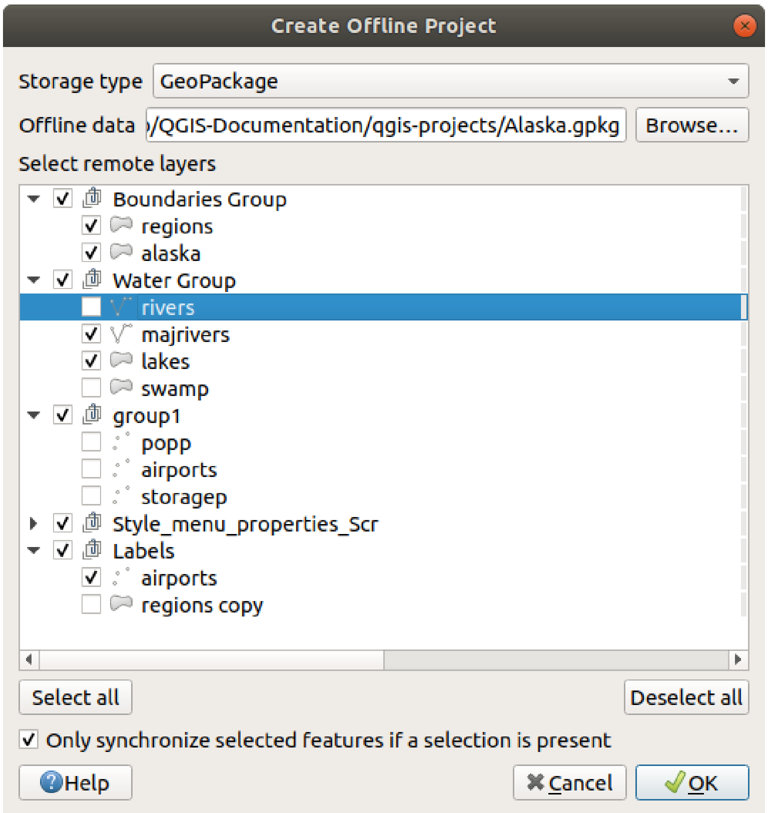


Abb. 24.27: Create an offline project

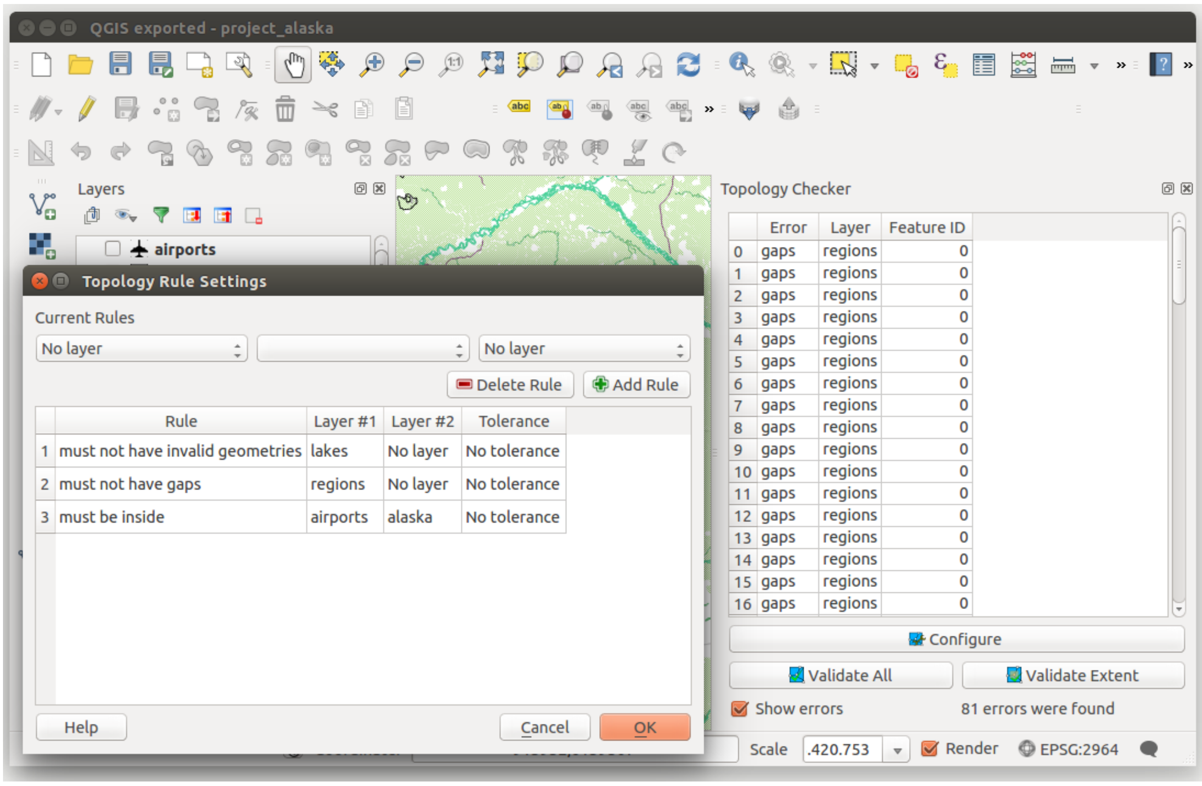


Abb. 24.28: The Topology Checker Plugin

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.

Below is the list of Core plugins provided with QGIS. They are not necessarily enabled by default.

Icon	Plugin	Beschreibung	Manual Reference
	Coordinate Capture	Capture mouse coordinate in different CRS	*Coordinate Capture Plugin*
	DB Manager	Manage your databases within QGIS	*DB Manager Plugin*
	eVis	Event Visualization Tool	*eVis Plugin*
	Geometrieprüfung	Check and repair errors in vector geometries	*Geometry Checker Plugin*
	Georeferencer GDAL	Georeference rasters with GDAL	*Plugin „Georeferenzierung“*
	GPS Tools	Tools for loading and importing GPS data	*GPS Plugin*
	GRASS	GRASS functionality	*GRASS GIS Integration*
	MetaSearch Catalog Client	Interact with metadata catalog services (CSW)	*MetaSearch Catalog Client*
	Offline Editing	Offline editing and synchronizing with database	*Offline Editing Plugin*
	Verarbeitung	Spatial data processing framework	*QGIS Verarbeitung Umgebung*
	Topology Checker	Find topological errors in vector layers	*Topology Checker Plugin*

24.3 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 the 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`).

24.3.1 The Interactive Console

The interactive console is composed of a toolbar, an input area and an output one.

Werkzeugleiste

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 (see *Optionen*);
- Help…: browses the current documentation.

Console

The console 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 *Optionen*;
- 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 file `~/.qgis2/console_history.txt`;
- Open QGIS C++ API documentation by typing `_api`;
- Open QGIS Python API documentation by typing `_pyqgis`.
- Open PyQGIS Cookbook by typing `_cookbook`.

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 (`>>>`, `...`).

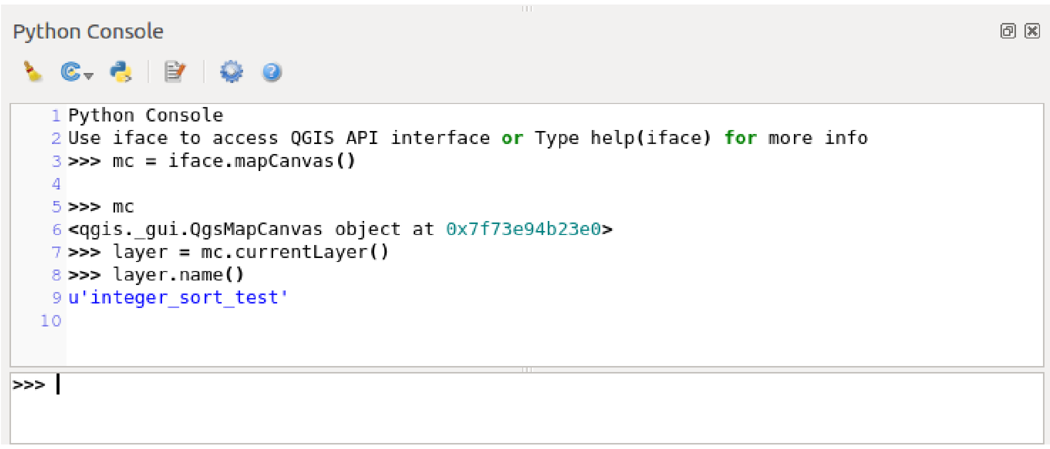


Abb. 24.29: The Python Console

24.3.2 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 codepad.org 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 codepad.org.
- `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.

24.3.3 Optionen

Accessible from the Console toolbar and the contextual menus of the Console output panel and the Code Editor, the *Python Console Settings* help manage and control the Python console behavior.

For both *Console* and *Editor* 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)
- *Typing*
  - *Automatic parentheses insertion*: Enables autoclosing for parentheses
  - *Automatic insertion of the ‚import' string on ‚from xxx'*: Enables insertion of ‚import' when specifying imports

For *Editor* you can also specify:

- *Run and Debug*

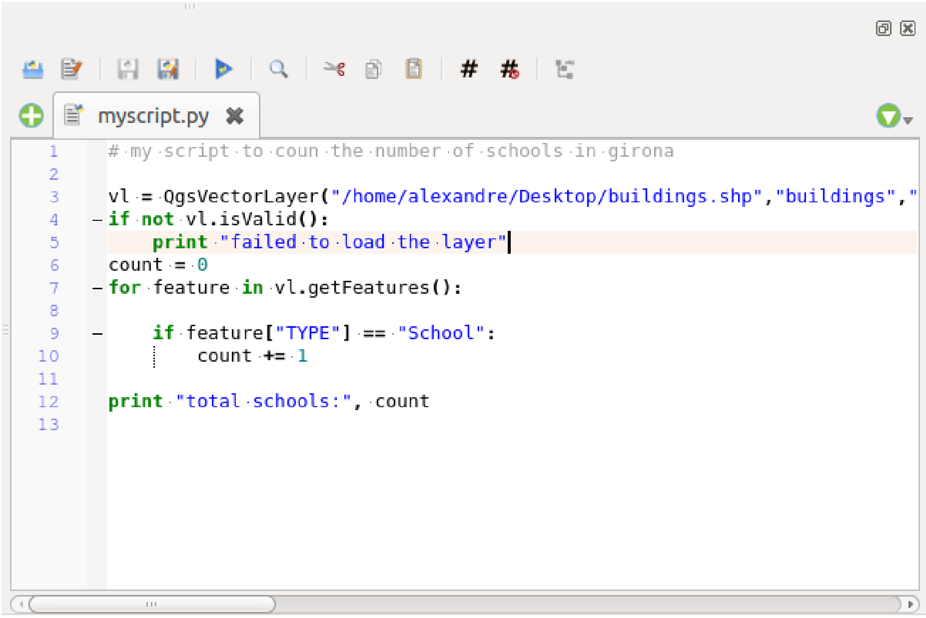


Abb. 24.30: The Python Console editor

  - *Enable Object Inspector (switching between tabs may be slow)*: Enable the object inspector.
  - *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.
- *Font and Colors*: Here you can specify the font to use in the editor and the colors to use for highlighting

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.

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.

KAPITEL 25

Hilfe und Support

25.1 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.

25.1.1 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

25.1.2 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.

25.1.3 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

25.1.4 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

25.1.5 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

25.1.6 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.

25.2 IRC

We also maintain a presence on IRC - visit us by joining the #qgis channel on irc.freenode.net. Please wait for a response to your question, as many folks on the channel are doing other things and it may take a while for them to notice your question. If you missed a discussion on IRC, not a problem! We log all discussion, so you can easily catch up. Just go to https://qgis.org/irclogs and read the IRC-logs.

25.3 Commercial support

Commercial support for QGIS is also available. Check the website https://qgis.org/en/site/forusers/commercial_support.html for more information.

25.4 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.

25.5 Blog

The QGIS community also runs a weblog at https://planet.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!

25.6 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‘.

25.7 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!

KAPITEL 26

Contributors

QGIS is an open source project developed by a team of dedicated volunteers and organisations. We strive to be a welcoming community for people of all race, creed, gender and walks of life. At any moment, you can get involved.

26.1 Autoren

Below are listed people who dedicate their time and energy to write, review, and update the whole QGIS documentation.

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	Akgar 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 Willitts
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 Koran Translator
Zoltan Siki	Håvard Tveite	Mattheo 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 Bărăscu	Maning Sambale	Claudia A. Engel	Yoichi Kayama
Otto Dassau	Denis Rouzaud	Nick Bearman	embelding	ajazepk
Ramon	Andrei	zstadler	icephale	

26.2 Translators

QGIS is a multi-language application and as is, also publishes a documentation translated into several languages. Many other languages are being translated and would be released as soon as they reach a reasonable percentage of translation. If you wish to help improving a language or request a new one, please see https://qgis.org/en/site/getinvolved/index.html.

The current translations are made possible thanks to:

Language	Contributors
Bahasa Indonesia	Emir Hartato, I Made Anombawa, Januar V. Simarmata, Muhammad Iqnaul Haq Siregar, Trias Aditya
Chinese (Traditional)	Calvin Ngei, Zhang Jun, Richard Xie
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, 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 Sodré Barros, Leônidas Descovi Filho, Marcelo Soares Souza, Narcélio de Sá Pereira Filho, Sidney Schaberle Goveia
Romanian	Alex Bădescu, Bogdan Pacurar, Georgiana Ioanovici, Lonut Losifescu-Enescu, Sorin Călinică, Tudor Bărăscu
Russian	Alexander Bruy, Artem Popov
Spanish	Carlos Dávila, Diana Galindo, Edwin Amado, Gabriela Awad, Javier César Aldariz, Mayeul Kauffmann, Fran Raga
Ukrainian	Alexander Bruy

KAPITEL 27

Anhänge

27.1 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 Verbreitungssystems 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.

27.2 Appendix 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

http://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.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. Anwendbarkeit und Definitionen

Diese Lizenz trifft auf jedes Handbuch oder sonstiges Werk zu, in beliebiger Form, das einen vom Urheberrechtsinhaber 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 Titelei-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 Bewandtnisse), 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 modifizierte 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 Titelei-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 Urheberrechtsinhabern, 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

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See 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
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.

27.3 Anhang C: QGIS-Dateiformate

27.3.1 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.

A QGIS file contains everything that is needed for storing a QGIS project, including:

- project title
- project CRS
- the layer tree
- snapping settings
- relations
- the map canvas extent
- project models
- legend
- 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.
- project properties

The figures below show the top level tags in a QGS file and the expanded `ProjectLayers` tag.

27.3.2 QLR - The QGIS Layer Definition file

A Layer Definition file (QLR) is an XML file that contains a pointer to the layer data source in addition to QGIS style information for the layer.

The use case for this file is simple: To have a single file for opening a data source and bringing in all the related style information. QLR files also allow you to mask the underlying datasource in an easy to open file.

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.

```
-<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>
   +<destinationsrs></destinationsrs>
     <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. 27.1: The top level tags in a QGS file

27.3.3 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).

27.4 Appendix D: QGIS R script syntax

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 algoritm 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
```

```
−<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" forceraster="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. 27.2: The expanded top level ProjectLayers tag of a QGS file

```
-<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/qgisbostaskdepplugin/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. 27.3: Die Tags der obersten Ebene einer QLR-Datei

```
-<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. 27.4: The top level tags of a QML file (only the renderer_v2 tag with its symbol tag is expanded)

27.4.1 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`

27.4.2 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.

27.4.3 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 ##load_vector_using_rgdal is specified)
vector point	Layer = vector point	sf object (or SpatialDataFrame object, if ##load_vector_using_rgdal is specified)
vector line	Layer = vector line	sf object (or SpatialDataFrame object, if ##load_vector_using_rgdal is specified)
vector polygon	Layer = vector polygon	sf object (or SpatialPolygonsDataFrame object, if ##load_vector_using_rgdal is used)
multiple vector	Layer = multiple vector	sf object (or SpatialDataFrame objects if ##load_vector_using_rgdal is specified)
table	Layer = table	Datenrahmenumwandlung von csv, voreingestelltes Objekt der `read.csv` Funktion
field	Field = Field Layer	Name des ausgewählten Feldes, z.B. `"Area"`
raster	Layer = raster	RasterBrick Objekt, voreingestelltes Objekt des `raster` Pakets
multiple raster	Layer = multiple raster	RasterBrick Objekt, voreingestelltes Objekt des `raster` 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 ergebenen KBS im Format: `"EPSG:4326"`
extent	E = extent	Ausdehnungsobjekt des `raster` Pakets, man kann Werte als *E@xmin*` 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)
```

27.4.4 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:",
↪"Maximum value:","Range:","Mean value:","Median value:","Standard deviation:"))
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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*.

KAPITEL 28

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.opengeospatial.org, 2002.

OPEN-GEOSPATIAL-CONSORTIUM. Web map service (1.3.0) implementation specification. https://portal.opengeospatial.org, 2004.

POSTGIS-PROJECT. Spatial support for postgresql. http://postgis.refractions.net/, 2013.