
QGIS User Guide

Relis 2.6

QGIS Project

22 May 2015

1	Pendahuluan	3
2	Konvensi	5
2.1	Konvensi GUI	5
2.2	Konvensi Teks atau Papan Ketik	5
2.3	Petunjuk Spesifik Platform	6
3	Kata Pengantar	7
4	Fitur-fitur	9
4.1	Lihat data	9
4.2	Jelajahi data dan menyusun peta	9
4.3	Membuat, menyunting, mengelola dan ekspor data	10
4.4	Analisis data	10
4.5	Terbitkan peta di Internet	10
4.6	Memperpanjang fungsionalitas QGIS melalui plugin	10
4.7	Python Console	11
4.8	Isu yang Diketahui	12
5	Apa yang baru pada QGIS 2.6	13
5.1	Aplikasi dan Opsi Proyek	13
5.2	Penyedia Data	13
5.3	Penyusun Peta	13
5.4	Server QGIS	14
5.5	Simbologi	14
5.6	Antarmuka Pengguna	14
6	Memulai	15
6.1	Pemasangan (Instalasi)	15
6.2	Contoh data	15
6.3	Sesi Contoh	16
6.4	Memulai dan Menghentikan QGIS	17
6.5	Pilihan Baris Perintah	17
6.6	Proyek	19
6.7	Keluaran (Output)	20
7	QGIS GUI	21
7.1	Bar Menu	22
7.2	Toolbar	27
7.3	Legenda Peta	27
7.4	Tampilan Peta	30
7.5	Status Bar	30

8	Peralatan Umum	31
8.1	Shortcut Papanketik	31
8.2	Konteks Bantuan	31
8.3	Rendering	31
8.4	Mengukur	33
8.5	Fitur Identifikasi	35
8.6	Dekorasi	36
8.7	Peralatan Anotasi	38
8.8	Bookmark Spasial	39
8.9	Proyek-proyek Nesting	41
9	Konfigurasi QGIS	43
9.1	Panel dan Toolbar	43
9.2	Properti Proyek	44
9.3	Opsi	44
9.4	Penyesuaian (Customization)	53
10	Working with Projections	55
10.1	Overview of Projection Support	55
10.2	Global Projection Specification	55
10.3	Define On The Fly (OTF) Reprojection	57
10.4	Custom Coordinate Reference System	58
10.5	Default datum transformations	59
11	QGIS Peramban	61
12	Pekerjaan dengan Data Vektor	63
12.1	Supported Data Formats	63
12.2	The Symbol Library	75
12.3	The Vector Properties Dialog	79
12.4	Expressions	105
12.5	Editing	112
12.6	Query Builder	129
12.7	Field Calculator	130
13	Pekerjaan dengan Data Raster	133
13.1	Working with Raster Data	133
13.2	Raster Properties Dialog	134
13.3	Kalkulator Raster	142
14	Pekerjaan dengan Data OGC	145
14.1	QGIS sebagai OGC Klien Data	145
14.2	QGIS sebagai OGC Data Server	154
15	Pekerjaan dengan Data GPS	161
15.1	GPS Plugin	161
15.2	Live GPS tracking	165
16	GRASS GIS Integration	171
16.1	Starting the GRASS plugin	171
16.2	Loading GRASS raster and vector layers	172
16.3	GRASS LOCATION and MAPSET	172
16.4	Importing data into a GRASS LOCATION	174
16.5	The GRASS vector data model	175
16.6	Creating a new GRASS vector layer	176
16.7	Digitizing and editing a GRASS vector layer	176
16.8	The GRASS region tool	179
16.9	The GRASS Toolbox	179

17 QGIS kerangka pengolahan	189
17.1 Pengantar	189
17.2 The toolbox	190
17.3 The graphical modeler	199
17.4 Antarmuka memproses batch	205
17.5 Using processing algorithms from the console	207
17.6 Manajer riwayat	212
17.7 Writing new Processing algorithms as python scripts	213
17.8 Handling data produced by the algorithm	215
17.9 Communicating with the user	215
17.10 Documenting your scripts	216
17.11 Example scripts	216
17.12 Best practices for writing script algorithms	216
17.13 Pre- and post-execution script hooks	216
17.14 Configuring external applications	217
17.15 The QGIS Commander	223
18 Processing providers and algorithms	225
18.1 GDAL algorithm provider	225
18.2 LAStools	258
18.3 Modeler Tools	283
18.4 OrfeoToolbox algorithm provider	285
18.5 QGIS algorithm provider	360
18.6 R algorithm provider	414
18.7 SAGA algorithm provider	424
18.8 TauDEM algorithm provider	595
19 Print Composer	627
19.1 First steps	628
19.2 Rendering mode	632
19.3 Composer Items	633
19.4 Manage items	655
19.5 Revert and Restore tools	656
19.6 Atlas generation	658
19.7 Creating Output	660
19.8 Manage the Composer	661
20 Plugin	663
20.1 Plugin-plugin QGIS	663
20.2 Menggunakan QGIS Plugin Inti	667
20.3 Plugin Mengambil Koordinat	668
20.4 Plugin Pengelola DB	668
20.5 Plugin Pengonversi Dxf2Shp	669
20.6 Plugin eVis	670
20.7 Plugin fTools	680
20.8 Plugin Peralatan GDAL	683
20.9 Plugin Georeferencer	686
20.10 Plugin Interpolasi	690
20.11 Plugin Mengedit Diluar Jaringan (Offline)	691
20.12 Plugin Spasial Oracle GeoRaster	691
20.13 Plugin Raster Analisis Terrain	694
20.14 Plugin Heatmap	695
20.15 MetaSearch Catalogue Client	699
20.16 Plugin Grafik Jalan	702
20.17 Plugin Spasial Query	703
20.18 Plugin SPIT	705
20.19 Plugin SQL Anywhere	705
20.20 Plugin Pemeriksa Topologi	708
20.21 Plugin Statistik Zonal	709

21 Bantuan dan Dukungan	711
21.1 Milis	711
21.2 IRC	712
21.3 BugTracker	712
21.4 Blog	713
21.5 Plugin	713
21.6 Wiki	713
22 Appendix	715
22.1 GNU General Public License	715
22.2 GNU Free Documentation License	718
23 Literatur dan Referensi Web	725
Indeks	727

·
·

Pendahuluan

Dokumen ini merupakan buku petunjuk asli dari perangkat lunak yang dijelaskan QGIS. Perangkat lunak dan perangkat keras yang dijelaskan dalam dokumen ini dalam banyak kasus merupakan merek dagang terdaftar dan karena itu tunduk pada persyaratan hukum. QGIS menggunakan Lisensi Publik Umum GNU (GNU General Public License). Temukan informasi lebih lanjut pada QGIS alamat web <http://www.qgis.org>.

Detil, data, hasil dll. pada dokumen ini telah ditulis ulang dan diverifikasi dengan sebaik mungkin pengetahuan dan tanggungjawab dari penulis dan editor. Bagaimanapun juga, dimungkinkan adanya kesalahan pada isi dokumen ini.

Oleh karena itu, semua data tidak bertanggung jawab untuk setiap pekerjaan atau jaminan. Para penulis, editor dan penerbit tidak mengambil tanggung jawab atau kewajiban atas kegagalan dan konsekuensinya. Selalu terbuka bagi Anda untuk menunjukkan kemungkinan kesalahan.

Dokumen ini telah diatur dengan reStructuredText. Ini tersedia sebagai sumber kode reST di [github](https://github.com) dan dalam jaringan (online) dengan format HTML dan PDF di <http://www.qgis.org/en/docs/>. Versi terjemahan dari dokumen ini dapat diunduh dalam beberapa format melalui proyek dokumentasi QGIS. Informasi lebih lanjut tentang kontribusi pada dokumen ini dan tentang menerjemahkannya, silakan kunjungi: <http://www.qgis.org/wiki/>.

Tautan pada dokumen ini

Dokumen ini berisi tautan internal dan eksternal. Silakan klik pada tautan internal akan membuka dokumen, sedangkan klik tautan eksternal membuka alamat internet. Dalam bentuk PDF, tautan internal dan eksternal diperlihatkan dengan warna biru dan ditangani oleh sistem peramban web (browser). Dalam bentuk HTML, menampilkan peramban web (browser) dan menangani keduanya secara identik.

Pengguna, Penulis dan Editor Panduan Pemasangan dan Pemrograman (Coding):

Tara Athan	Radim Blazek	Godofredo Contreras	Otto Dassau	Martin Dobias
Peter Ersts	Anne Ghisla	Stephan Holl	N. Horning	Magnus Homann
Werner Macho	Carson J.Q. Farmer	Tyler Mitchell	K. Koy	Lars Luthman
Claudia A. Engel	Brendan Morely	David Willis	Jürgen E. Fischer	Marco Hugentobler
Larissa Junek	Diethard Jansen	Paolo Corti	Gavin Macaulay	Gary E. Sherman
Tim Sutton	Alex Bruy	Raymond Nijssen	Richard Duivenvoorde	Andreas Neumann
Astrid Emde	Yves Jacolin	Alexandre Neto	Andy Schmid	Hien Tran-Quang

Hak Cipta (c) 2004 - 2014 QGIS Tim Pengembang

Internet: <http://www.qgis.org>

Lisensi dokumen ini


Diizinkan untuk menyalin, mendistribusikan dan / atau memodifikasi dokumen ini di bawah syarat-syarat Lisensi Dokumentasi Bebas GNU (GNU Free Documentation License), atau versi yang lebih baru yang diterbitkan oleh Free Software Foundation; tanpa Bagian Invarian, tanpa Teks Sampul-Depan, dan tanpa Teks Sampul-Belakang. Salinan lisensi termasuk dalam Lampiran *GNU Free Documentation License*.

Konvensi

Bagian ini menjelaskan gaya seragam yang akan digunakan di seluruh manual ini.

2.1 Konvensi GUI

Gaya konvensi GUI dimaksudkan untuk meniru tampilan GUI. Secara umum, gaya akan mencerminkan penampilan non-hover, sehingga pengguna dapat memindai visual GUI untuk menemukan sesuatu yang tampak seperti petunjuk di manual.

- Menu Pilihan: *Lapisan* → *Tambah Lapisan Raster* atau :menuselection: *Pengaturan → Toolbar → Digitalisasi
- Alat:  *Tambah Lapisan Raster*
- Tombol : **[Simpan sebagai Default]**
- Kotak Judul: *Properti Lapisan*
- Tab: *Umum*
- Kotak Centang: *Render*
- Tombol Radio: *Postgis SRID* *EPSG ID*
- Pilih Nomor:
- Pilih String:
- Lihat Berkas:
- Pilih Warna: **Border**
- Slider:
- Masukkan Teks: **Display name**

Sebuah bayangan menunjukkan komponen GUI yang dapat diklik.

2.2 Konvensi Teks atau Papan Ketik

Panduan ini juga mencakup gaya yang berhubungan dengan teks, perintah papan ketik dan pemrograman (coding) untuk menunjukkan entitas yang berbeda, seperti kelas, atau metode. Gaya ini tidak sesuai dengan penampilan yang sebenarnya dari teks atau coding dalam QGIS.

- Pranala: <http://qgis.org>
- Kombinasi Keystroke: tekan `Ctrl+B`, artinya tekan dan tahan tombol `Ctrl` dan tekan tombol `B`.

Kata Pengantar

Selamat datang di dunia indah dari Sistem Informasi Geografis (GIS)!

QGIS adalah Sistem Informasi Geografis Sumber Terbuka (Open Source). Proyek ini lahir di bulan Mei 2002 dan didirikan sebagai sebuah proyek di SourceForge pada bulan Juni tahun yang sama. Kami telah bekerja keras membuat perangkat lunak GIS (merupakan perangkat lunak proprietary tradisional mahal) prospek yang layak bagi siapa saja dengan akses dasar ke Personal Komputer. QGIS saat ini berjalan pada kebanyakan platform Unix, Windows, dan OS X. QGIS dikembangkan menggunakan toolkit Qt (<http://qt.digia.com>) dan C++. Ini berarti bahwa QGIS terasa cepat dan menyenangkan, antarmuka pengguna grafis yang mudah digunakan (GUI).

QGIS bertujuan untuk menjadi GIS yang mudah digunakan, menyediakan fungsi dan fitur-fitur umum. Tujuan awalnya adalah untuk menyediakan penampil data GIS. QGIS telah mencapai titik dalam evolusi di mana ia digunakan sehari-hari oleh banyak orang untuk kebutuhan melihat data GIS mereka. QGIS mendukung sejumlah format data raster dan vektor, dengan dukungan format baru mudah ditambahkan dengan menggunakan arsitektur plugin.

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 also can find it in Appendix *GNU General Public License*.

Tip: Pembaruan Dokumentasi

Versi terbaru dari dokumen ini selalu dapat ditemukan di website dokumentasi QGIS di <http://www.qgis.org/en/docs/>

Fitur-fitur

QGIS menawarkan banyak fungsi GIS umum yang disediakan oleh fitur inti dan plugin. Sebuah ringkasan pendek dari enam kategori umum fitur dan plugin disajikan di bawah ini, diikuti oleh wawasan pertama ke konsol Python yang terintegrasi.

4.1 Lihat data

Anda dapat melihat dan overlay data vektor dan raster dalam format dan proyeksi yang berbeda tanpa konversi ke format internal maupun umum. Format yang didukung termasuk:

- Tabel spasial-enabled dan tampilan menggunakan PostGIS, SpatiaLite dan MSSQL Spasial, Oracle Spasial, format vektor yang didukung oleh perpustakaan OGR, termasuk ESRI shapefile, MapInfo, SDTS, GML dan banyak lagi, lihat bagian *Pekerjaan dengan Data Vektor*.
- Format raster dan citra yang didukung dengan terpasangnya GDAL (Geospatial Data Abstraction Library) perpustakaan, seperti GeoTiff, ERDAS IMG, ArcInfo ASCII GRID, JPEG, PNG dan banyak lagi, lihat bagian *Pekerjaan dengan Data Raster*.
- Data raster dan vektor GRASS dari basis data GRASS (lokasi/mapset). Lihat bagian *GRASS GIS Integration*.
- Data spasial dalam jaringan sebagai Layanan OGC Web, termasuk WMS, WMTS, WCS, WFS, dan WFS-T. Lihat bagian *Pekerjaan dengan Data OGC*.

4.2 Jelajahi data dan menyusun peta

Anda dapat membuat peta interaktif dan mengeksplorasi data spasial dengan GUI yang ramah. Banyak alat yang tersedia di GUI termasuk:

- QGIS peramban web
- On-the-fly proyeksi ulang
- Pengelola DB
- Penyusun Peta
- Panel Peninjau
- Bookmark spasial
- Annotation tools
- Identifikasi/pilih fitur
- Sunting/lihat/cari atribut
- Fitur pelabelan data-ditentukan

- Alat simbologi vektor dan raster data-ditentukan
- Peta komposisi atlas dengan lapisan graticule
- Bar skala panah utara dan label hak cipta untuk peta
- Dukungan menyimpan dan mengembalikan proyek-proyek

4.3 Membuat, menyunting, mengelola dan ekspor data

Anda dapat membuat, mengedit, mengelola dan ekspor lapisan vektor dan raster dalam beberapa format. QGIS menawarkan sebagai berikut:

- Alat Digitalisasi yang didukung format OGR dan lapisan vektor GRASS
- Kemampuan untuk membuat dan mengedit lapisan shapefile dan vektor GRASS
- Plugin Georeferencer ke geocode gambar
- Alat GPS untuk mengimpor dan mengekspor format GPX, dan mengkonversi format GPS lain ke GPX atau unduh/unggah langsung ke unit GPS (di Linux, usb: telah ditambahkan ke daftar perangkat GPS.)
- Dukungan memvisualisasikan dan mengedit data OpenStreetMap
- Kemampuan membuat tabel basis data spasial dari shapefile dengan Plugin Pengelola DB
- Peningkatan penanganan tabel basis data spasial
- Peralatan untuk mengelola tabel atribut vektor
- Opsi menyimpan cuplikan layar sebagai gambar ber-georeferensi.

4.4 Analisis data

Anda dapat melakukan analisis data spasial pada basis data spasial dan format OGR lain yang didukung. QGIS saat ini menawarkan analisis vektor, sampling, geoprocessing, geometri dan aplikasi manajemen basis data. Anda juga dapat menggunakan aplikasi GRASS terintegrasi, yang meliputi fungsi GRASS lengkap lebih dari 400 modul (lihat bagian *GRASS GIS Integration*). Atau, Anda bekerja dengan Plugin Processing, yang menyediakan kerangka analisis geospasial yang kuat untuk memanggil algoritma pihak asli dan ketiga dari QGIS, seperti GDAL, SAGA, GRASS, fTools dan banyak lagi (lihat bagian *Pengantar*.)

4.5 Terbitkan peta di Internet

QGIS dapat digunakan sebagai WMS, WMTS, WMS-C atau WFS dan WFS-T client, dan sebagai WMS, WCS atau WFS server (lihat bagian *Pekerjaan dengan Data OGC*.). Selain itu Anda dapat mengekspor data, mempublikasikan mereka di internet menggunakan webserver dengan UMN MapServer atau GeoServer yang terpasang.

4.6 Memperpanjang fungsionalitas QGIS melalui plugin

QGIS dapat disesuaikan dengan kebutuhan khusus Anda dengan arsitektur plugin extensible dan perpustakaan yang dapat digunakan untuk membuat plugin. Anda dapat membuat aplikasi baru dengan C++ atau Python!

4.6.1 Inti Plugin

Plugin inti termasuk:

1. Rekaman Koordinat (Tetikus merekam koordinat di CRS yang berbeda)
2. DB Manager (Pertukaran, mengedit dan melihat lapisan dan tabel; mengeksekusi query SQL)
3. Diagram Overlay (Menempatkan diagram pada lapisan vektor)
4. Pengkonversi Dxf2Shp (Mengonversi DXF ke Shape)
5. eVIS (Visualize events)
6. fTools (Analisa dan kelola data vektor)
7. GDALTools (Integrasi alat GDAL ke dalam QGIS)
8. Georeferencer GDAL (Menambahkan informasi proyeksi ke raster menggunakan GDAL)
9. Peralatan GPS (Memuat dan impor data GPS)
10. GRASS (GRASS GIS integrasi)
11. Heatmap (Menghasilkan raster heatmap dari data titik)
12. Plugin Interpolasi (interpolasi berdasarkan simpul dari lapisan vektor)
13. Penyuntingan Luar Jaringan (Memungkinkan menyunting luar jaringan dan sinkronisasi dengan basis data)
14. Oracle Spatial GeoRaster
15. Processing (formerly SEXTANTE)
16. Analisis Terrain Raster (Analisis terrain berbasis raster)
17. Plugin Grafik Jalan (Analisis jaringan terpendek)
18. Spatial Query Plugin
19. SPIT (Impor Shapefile ke PostgreSQL/PostGIS)
20. Plugin SQL Anywhere (Menyimpan lapisan vektor dengan basis data SQL Anywhere)
21. Pemeriksa Topologi (Menemukan kesalahan topologi dalam lapisan vektor)
22. Plugin Zonal Statistik (hitung, jumlah, rata-rata raster untuk setiap poligon dari lapisan vektor)

4.6.2 Plugin Eksternal Python

QGIS menawarkan semakin banyak plugin python eksternal yang diberikan oleh masyarakat. Plugin ini berada di repositori resmi plugin, dan dapat dengan mudah dipasang menggunakan Pemasang Plugin Python. Lihat bagian *load_external_plugin*.

4.7 Python Console

Untuk membuat skrip, memungkinkan untuk mengambil keuntungan dari konsol Python terintegrasi, dimana bisa dibuka dari menu: *Plugin* → *Konsol Python*. Konsol terbuka sebagai jendela utilitas non-modal. Untuk interaksi dengan lingkungan QGIS, ada variabel `qgis.utils iface`, yang merupakan contoh dari `QgsInterface`. Antarmuka ini memungkinkan akses ke kanvas peta, menu, toolbar dan bagian lain dari aplikasi QGIS.

Untuk informasi lebih lanjut tentang bekerja dengan plugin dan aplikasi Python Console dan Programming PyQGIS, silakan mengacu ke http://www.qgis.org/html/en/docs/pyqgis_developer_cookbook/index.html.

4.8 Isu yang Diketahui

4.8.1 Jumlah dari batas berkas yang dibuka

Jika Anda membuka sebuah proyek QGIS besar dan Anda yakin bahwa semua lapisan valid, tetapi beberapa lapisan ditandai sebagai lapisan buruk, Anda mungkin dihadapkan dengan masalah ini. Linux (dan OS lain juga) memiliki batas berkas yang dibuka. Batasan sumber daya per-proses dan diturunkan. Perintah `ulimit` merupakan terminasi built-in, mengubah batas hanya untuk proses terminal saat ini; batas baru akan diturunkan oleh setiap proses anak.

Anda bisa melihat semua informasi `ulimit` dengan mengetik

```
user@host:~$ ulimit -aS
```

Anda bisa melihat jumlah yang diperbolehkan saat berkas dibuka per proses dengan perintah berikut di konsol

```
user@host:~$ ulimit -Sn
```

Untuk mengubah batas **sesi yang ada**, Anda mungkin dapat menggunakan sesuatu seperti

```
user@host:~$ ulimit -Sn #number_of_allowed_open_files
user@host:~$ ulimit -Sn
user@host:~$ qgis
```

Untuk memperbaikinya selamanya

Pada kebanyakan sistem Linux, batasan sumber daya yang ditetapkan pada login dengan modul `pam_limits` sesuai dengan pengaturan yang terkandung dalam `/etc/security/limits.conf` atau `/etc/security/limits.d/*.conf`. Anda harus dapat mengedit berkas jika Anda memiliki hak istimewa root (juga via `sudo`), tetapi Anda akan perlu untuk login lagi sebelum perubahan berlaku.

Informasi tambahan:

<http://www.cyberciti.biz/faq/linux-increase-the-maximum-number-of-open-files/> <http://linuxaria.com/article/open-files-in-linux?lang=en>

.

Apa yang baru pada QGIS 2.6

This release contains new features and extends the programmatic interface over previous versions. We recommend that you use this version over previous releases.

This release includes hundreds of bug fixes and many new features and enhancements that will be described in this manual. You may also review the visual changelog at <http://changelog.linfinity.com/qgis/version/2.6.0/>.

5.1 Aplikasi dan Opsi Proyek

- **Project filename in properties:** You can now see the full path for the QGIS project file in the project properties dialog.

5.2 Penyedia Data

- **DXF Export tool improvements:**
 - Tree view and attribute selection for layer assignment in dialog
 - support fill polygons/HATCH
 - represent texts as MTEXT instead of TEXT (including font, slant and weight)
 - support for RGB colors when there's no exact color match
 - use AutoCAD 2000 DXF (R15) instead of R12

5.3 Penyusun Peta

- **Update map canvas extent from map composer extent:** **On the Item** properties of a Map element there are now two extra buttons which allow you to (1) set the Map canvas extent according with the extent of your Map element and (2) view in Map canvas the extent currently set on your Map element.
- **Multiple grid support:** It is now possible to have more than one grid in your Map element. Each grid is fully customizable and can be assigned to a different CRS. This means, for example, you can now have a map layout with both geographic and projected grids.
- **Selective export:** To every item of your map composer layout, under Rendering options, you may exclude that object from map exports.

5.4 Server QGIS

5.5 Simbologi

5.6 Antarmuka Pengguna

Memulai

Bab ini memberikan gambaran singkat cara memasang QGIS, beberapa contoh data dari QGIS halaman web dan menjalankan sesi pertama memvisualisasikan lapisan raster dan vektor sederhana.

6.1 Pemasangan (Instalasi)

Pemasangan QGIS sangat sederhana. Standar paket installer tersedia untuk MS Windows dan Mac OS X. Tersedia paket binari GNU/Linux (rpm dan deb) atau repositori perangkat lunak untuk menambah manajer instalasi. Dapatkan informasi terakhir paket binari pada website QGIS <http://download.qgis.org>.

6.1.1 Pemasangan dari sumber


Jika Anda perlu membangun QGIS dari sumber, silakan mengacu petunjuk instalasi. Mereka didistribusikan dengan kode sumber QGIS dalam sebuah berkas yang bernama 'INSTALL'. Anda juga bisa menemukannya di dalam jaringan (online) di <http://htmlpreview.github.io/?https://raw.githubusercontent.com/qgis/QGIS/master/doc/INSTALL.html>

6.1.2 Pemasangan pada media eksternal


QGIS memungkinkan Anda untuk menentukan opsi `--configpath` yang menimpa path standar untuk konfigurasi pengguna (misalnya, `~/ .qgis2` Linux) dan kemampuan **QSettings** menggunakan direktori ini juga. Hal ini memungkinkan Anda juga, membawa pemasang QGIS dalam pada flash drive bersama dengan semua plugin dan pengaturan. Lihat Bagian *Menu Sistem* untuk informasi tambahan.

6.2 Contoh data

Panduan pengguna berisi contoh-contoh berdasarkan contoh dataset QGIS.

 installer Windows memiliki pilihan untuk mengunduh contoh dataset QGIS. Jika dicentang, data akan diunduh ke folder `My Documents` Anda dan ditempatkan dalam folder bernama `GIS database`. Anda dapat menggunakan Windows Explorer untuk memindahkan folder ini ke setiap lokasi yang nyaman. Jika Anda tidak memilih kotak centang untuk memasang contoh dataset selama instalasi QGIS, Anda dapat melakukan salah satu dari berikut:

- Gunakan data GIS yang Anda miliki
- Unduh contoh data dari http://download.osgeo.org/qgis/data/qgis_sample_data.zip
- Hapus (uninstall) QGIS dan pasang ulang dengan opsi unduh data, (hanya direkomendasikan jika solusi di atas tidak berhasil)

 **X** Untuk GNU/Linux dan Mac OS X, belum ada paket instalasi dataset yang tersedia rpm, deb atau dmg. Untuk menggunakan contoh dataset unduh berkas ZIP arsip `qgis_sample_data` dari http://download.osgeo.org/qgis/data/qgis_sample_data.zip dan unzip arsip pada sistem Anda.

Dataset Alaska mencakup semua data GIS yang digunakan sebagai contoh dan cuplikan layar di buku panduan; dan termasuk basis data kecil GRASS. Proyeksi contoh dataset QGIS adalah Alaska Albers Equal Area dengan satuan kaki. Kode EPSG 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]]
```

Jika Anda berniat untuk menggunakan QGIS sebagai grafis frontend GRASS, Anda dapat menemukan pilihan contoh lokasi (misalnya Spearfish atau South Dakota) di situs resmi GRASS GIS <http://grass.osgeo.org/download/sample-data/>.



6.3 Sesi Contoh






Sekarang QGIS Anda sudah terpasang dan contoh dataset tersedia, kami ingin menunjukkan sesi contoh QGIS singkat dan sederhana. Kami akan memvisualisasikan lapisan (layer) raster dan vektor. Kami akan menggunakan lapisan (layer) raster tutupan lahan `qgis_sample_data/raster/landcover.img` dan lapisan (layer) vektor danau `qgis_sample_data/gml/lakes.gml`.

6.3.1 Mulai QGIS

-  Mulai QGIS dengan mengetik: “QGIS” di terminal, atau menggunakan bineri precompiled, menggunakan menu Aplikasi.
-  Mulai QGIS menggunakan menu Start atau shortcut desktop, atau double klik di berkas proyek QGIS.
- **X** Double klik ikon pada folder Aplikasi.

6.3.2 Muat lapisan (layer) raster dan vektor dari contoh dataset



1. Klik di ikon  Load Raster
2. Jelejahi folder `qgis_sample_data/raster/`, pilih berkas ERDAS `landcover.img` dan klik **[Buka]**.
3. If the file is not listed, check if the *Files of type*  combo box at the bottom of the dialog is set on the right type, in this case “Erdas Imagine Images (*.img, *.IMG)”.

4. Sekarang klik di ikon  Load Vector.
5.  *Berkas* harus dipilih sebagai Jenis Sumber Tipe Sumber dalam dialog baru *Tambah lapisan vektor*. Sekarang klik **[Jelajah]** untuk memilih lapisan vektor.
6. Browse to the folder `qgis_sample_data/gml/`, select 'Geography Markup Language [GML] [OGR] (.gml,.GML)' from the *Files of type*  combo box, then select the GML file `lakes.gml` and click **[Open]**. In the *Add vector layer* dialog, click **[OK]**. The *Coordinate Reference System Selector* dialog opens with *NAD27 / Alaska Albers* selected, click **[OK]**.
7. Perbesar sedikit ke daerah favorit Anda dengan beberapa danau.
8. Dobel klik lapisan (layer) `lakes` di legenda peta untuk membuka dialog *Properti*.
9. Klik pada tab *Gaya* dan pilih biru sebagai warna.
10. Klik tab *Label* dan centang kotak centang  *Label lapisan ini dengan* untuk mengaktifkan pelabelan. Pilih kolom "NAMES" sebagai kolom isian label.
11. Untuk memudahkan pembacaan label, Anda dapat menambahkan penyangga (buffer) putih di sekitar mereka, dengan klik "Buffer" dalam daftar sebelah kiri, periksa  *Draw text buffer* dan pilih 3 sebagai ukuran penyangga (buffer).
12. Klik **[Terapkan]**. Periksa apakah hasilnya baik, dan terakhir klik **[OK]**.

Anda dapat melihat betapa mudahnya untuk memvisualisasikan lapisan (layer) raster dan vektor dalam QGIS. Mari kita lanjutkan ke bagian berikut untuk mempelajari lebih lanjut tentang fungsi, fitur dan pengaturan yang tersedia dan bagaimana menggunakannya.


6.4 Memulai dan Menghentikan QGIS

Dalam bagian *Sesi Contoh* Anda sudah belajar bagaimana memulai QGIS. Kami akan mengulanginya di sini dan Anda akan melihat QGIS juga menyediakan opsi baris perintah lebih lanjut.

-  Dengan asumsi bahwa QGIS dipasang di PATH, Anda dapat memulai QGIS dengan mengetik: `qgis` pada terminal atau dengan mengklik double pada link aplikasi QGIS (atau shortcut) pada desktop atau dalam menu aplikasi.
-  Mulai QGIS menggunakan menu Start atau shortcut desktop, atau double klik di berkas proyek QGIS.
- **X** Double klik ikon di folder Aplikasi Anda. Jika Anda memulai QGIS di terminal, jalankan `/path-to-installation-executable/Contents/MacOS/Qgis`.

Untuk menghentikan QGIS, klik menu opsi  *Berkas X QGIS* → *Keluar*, atau menggunakan shortcut `Ctrl+Q`.

6.5 Pilihan Baris Perintah

 QGIS mendukung sejumlah pilihan ketika dimulai dari baris perintah. Untuk mendapatkan daftar opsi, enter `qgis --help` pada baris perintah. Pernyataan penggunaan untuk QGIS adalah:

```
qgis --help
QGIS - 2.6.0-Brighton 'Brighton' (exported)
QGIS is a user friendly Open Source Geographic Information System.
Usage: /usr/bin/qgis.bin [OPTION] [FILE]
OPTION:
  [--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
[--project projectfile] load the given QGIS project
[--extent xmin,ymin,xmax,ymax] set initial map extent
[--nologo]             hide splash screen
[--noplugins]          don't restore plugins on startup
[--nocustomization]    don't apply GUI customization
[--customizationfile] use the given ini file as GUI customization
[--optionspath path]   use the given QSettings path
[--configpath path]    use the given path for all user configuration
[--code path]          run the given python file on load
[--defaultui]          start by resetting user ui settings to default
[--help]               this text
```

FILE:

Files specified on the command line can include rasters, vectors, and QGIS project files (.qgs):

1. Rasters - supported formats include GeoTiff, DEM and others supported by GDAL
2. Vectors - supported formats include ESRI Shapefiles and others supported by OGR and PostgreSQL layers using the PostGIS extension

Tip: Contoh Menggunakan argumen baris perintah

Anda dapat memulai QGIS dengan menentukan satu atau lebih berkas data pada baris perintah. Misalnya, dengan asumsi Anda berada di direktori `qgis_sample_data`, Anda bisa memulai QGIS dengan berkas lapisan vektor dan raster yang diatur untuk dimuat pada startup dengan menggunakan perintah berikut: `qgis ./raster/landcover.img ./gml/lakes.gml`

Pilihan baris perintah `--snapshot`

Pilihan ini memungkinkan Anda untuk membuat cuplikan layar dalam format PNG dari tampilan saat ini. Hal ini sangat berguna ketika Anda memiliki banyak proyek dan ingin menghasilkan cuplikan layar dari data Anda.

Saat ini menghasilkan berkas PNG dengan piksel 800x600. Hal ini dapat diadaptasi dengan menggunakan argumen baris perintah `--width` dan `--height`. Nama berkas bisa ditambahkan setelah `--snapshot`.

Pilihan baris perintah `--lang`

Berdasarkan lokal QGIS Anda, pilih lokalisasi yang benar. Jika Anda ingin mengubah bahasa, Anda dapat menentukan kode bahasa. Sebagai contoh: `--lang=id` QGIS dimulai menggunakan lokal Indonesia. Daftar bahasa saat ini yang didukung dengan kode bahasa dan status disediakan di http://hub.qgis.org/wiki/quantum-gis/GUI_Translation_Progress.

Baris perintah `--project`

Memulai QGIS dengan berkas proyek yang sudah ada. Hanya tambahkan baris perintah `--project` ikuti nama berkas proyek yang akan dibuka dan QGIS akan membukanya dengan memuat semua lapisan (layer).

Baris perintah `--extent`

Untuk memulai dengan peta batas tertentu menggunakan opsi ini. Anda perlu menambahkan kotak bounding sejauh Anda dalam urutan dipisahkan oleh koma:

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

Baris perintah `--nologo`

Argumen baris perintah ini menyembunyikan layar splash ketika Anda mulai QGIS.

Baris perintah `--noplugins`

Jika Anda mengalami kesulitan pada saat startup dengan plugin, Anda dapat menghindari beban mereka pada saat startup dengan opsi ini. Mereka masih akan tersedia di Manajer Plugin after-wards.

Baris perintah `--nocustomization`

Menggunakan argumen baris perintah ini, Anda bisa menentukan penyesuaian berkas GUI, yang akan digunakan saat startup.

Baris perintah `--nocustomization`

Menggunakan argumen baris perintah ini, penyesuaian GUI yang ada tidak akan diterapkan pada saat startup.

Baris perintah `--optionspath`

You can have multiple configurations and decide which one to use when starting QGIS with this option. See *Opsi* to confirm where the operating system saves the settings files. Presently, there is no way to specify a file to write settings to; therefore, you can create a copy of the original settings file and rename it. The option specifies path to directory with settings. For example, to use `/path/to/config/QGIS/QGIS2.ini` settings file, use option:

```
--optionspath /path/to/config/
```

Baris perintah `--configpath`

Pilihan ini mirip dengan yang di atas, tapi selanjutnya menimpa path default untuk konfigurasi pengguna (`~/qgis2`) dan kemampuan **QSettings** untuk menggunakan direktori ini juga. Hal ini memungkinkan pengguna mis membawa instalasi QGIS pada flash drive bersama dengan semua plugin dan pengaturan.

Command line option `--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`

6.6 Proyek

Sesi QGIS Anda dianggap sebagai sebuah proyek. QGIS bekerja pada satu proyek pada satu waktu. Pengaturan yang baik dianggap sebagai per-proyek, atau sebagai default untuk proyek-proyek baru (lihat Bagian *Opsi*). QGIS bisa menyimpan kerja Anda ke dalam sebuah berkas proyek dengan menggunakan pilihan menu *Proyek* →  *Simpan* atau *Proyek* →  *Simpan Sebagai*.



Memuat proyek tersimpan ke dalam sesi QGIS menggunakan *Proyek* →  *Buka ...*, *Proyek* → *Baru dari template* atau *Proyek* → *Buka yang baru teradi* →.

Jika Anda ingin membersihkan sesi Anda dan memulai baru lagi, pilih *Proyek* →  *Baru*. Pilihan menu ini akan meminta Anda untuk menyimpan proyek yang telah ada jika perubahan yang telah dibuat sejak dibuka atau yang terakhir disimpan.

Jenis-jenis informasi yang disimpan dalam berkas proyek meliputi:

- Lapisan (layer) yang ditambahkan
- Properti lapisan (layer), termasuk simbolisasi
- Proyeksi untuk tampilan peta
- Tampilan terakhir



Berkas proyek disimpan dalam format XML, sehingga memungkinkan untuk mengedit berkas diluar QGIS jika Anda tahu apa yang Anda lakukan. Format berkas telah diperbaharui beberapa kali dibanding versi QGIS sebelumnya. Berkas proyek dari versi QGIS yang lebih tua mungkin tidak bekerja dengan baik lagi. Harus dibuat sadar akan hal ini, di tab *Umum Pengaturan* → *Opsi* Anda dapat memilih:

-  *Prompt untuk menyimpan proyek dan perubahan sumber data bila diperlukan*
-  *Peringatkan ketika membuka berkas proyek QGIS yang disimpan dengan versi lama*

Setiap kali Anda menyimpan proyek dalam QGIS 2.2 sekarang berkas cadangan dari proyek dibuat.

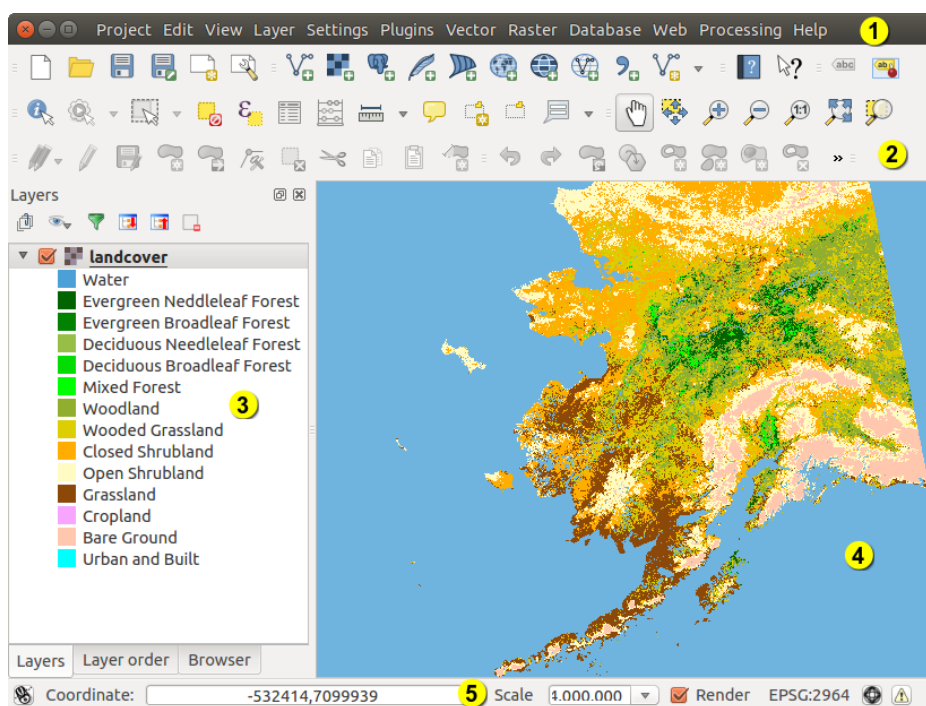
6.7 Keluaran (Output)

Ada beberapa cara untuk menghasilkan keluaran (output) dari sesi QGIS Anda. Kita telah membahasnya pada Bagian *Proyek*, menyimpan berkas proyek. Berikut ini adalah contoh cara lain untuk menghasilkan keluaran (output) berkas:

- Menu opsi *Proyek* →  *Simpan sebagai Gambar* membuka dialog berkas di mana Anda memilih nama, path dan jenis gambar (format PNG atau JPG). Sebuah berkas dengan ekstensi PNGW atau JPGW disimpan dalam folder sama dengan gambar yang mempunyai georeferensi.
- Menu opsi *Proyek* → *Ekspor DXF ...* membuka dialog di mana Anda dapat menentukan ‘Symbology mode’, ‘Symbology scale’ dan lapisan vektor yang ingin Anda ekspor ke DXF.
- Menu opsi *Proyek* →  *Penyusun Cetak Baru* membuka dialog di mana Anda dapat me-layout dan mencetak kanvas peta saat ini (lihat Bagian *Print Composer*).

QGIS GUI

Ketika QGIS dimulai, Anda akan disajikan tampilan GUI seperti gambar (nomer 1 sampai 5 dalam lingkaran kuning mengacu pada 5 area utama antarmuka yang dibahas dibawah ini).



Gambar 7.1: GUI QGIS dengan contoh data Alaska 🐧

Catatan: Jendela dekorasi Anda (judul bar, dll) dapat terlihat berbeda tergantung pada sistem operasi dan window manager Anda.

GUI QGIS dibagi dalam lima area:

1. Bar Menu
2. Bar Tool
3. Legenda Peta
4. Tampilan Peta
5. Status Bar









Kelima komponen antarmuka QGIS dijelaskan secara lebih rinci dalam bagian berikut. Lebih dari dua bagian dijelaskan di shortcut papan ketik dan bantuan.

7.1 Bar Menu

Menu bar memberikan akses ke berbagai fitur QGIS menggunakan standar hirarki menu. Menu-menu utama dan ringkasan dari beberapa menu pilihan yang tercantum di bawah ini, bersama-sama dengan ikon dari alat yang sesuai seperti yang ditampilkan pada toolbar, seperti shortcut papan ketik. Shortcut papan ketik juga dapat dikonfigurasi secara manual menggunakan dialog *Configure shortcuts*, dibuka dari *Pengaturan* → *Konfigurasi Shortcut...*


Meskipun sebagian besar pilihan menu memiliki alat yang sesuai dan sebaliknya, menu tidak terorganisir seperti toolbar. Toolbar yang berisi alat ini bisa terdaftar setelah setiap pilihan menu diisi pada kotak centang. Beberapa pilihan menu hanya muncul jika plugin yang sesuai dimuat. Untuk informasi lebih lanjut tentang alat dan toolbar, lihat Bagian *Toolbar*.

7.1.1 Proyek




Pilihan Menu	Shortcut	Referensi	Toolbar
 <i>Baru</i>	Ctrl+N	lihat <i>Proyek</i>	<i>Proyek</i>
 <i>Buka</i> <i>Baru dari template</i> → <i>Buka terakhir dikerjakan</i> →	Ctrl+O	lihat <i>Proyek</i> lihat <i>Proyek</i> lihat <i>Proyek</i>	<i>Proyek</i> <i>Proyek</i>
 <i>Simpan</i>	Ctrl+S	lihat <i>Proyek</i>	<i>Proyek</i>
 <i>Simpan Sebagai...</i>	Ctrl+Shift+S	lihat <i>Proyek</i>	<i>Proyek</i>
 <i>Simpan sebagai gambar...</i> <i>Ekspor DXF ...</i>		lihat <i>Keluaran (Output)</i> lihat <i>Keluaran (Output)</i>	
 <i>Penyusun cetak baru</i>	Ctrl+P	lihat <i>Print Composer</i>	<i>Proyek</i>
 <i>Manajer Penyusun ...</i> <i>Penyusun Cetak</i> →		lihat <i>Print Composer</i> lihat <i>Print Composer</i>	<i>Proyek</i>
 <i>Keluar QGIS</i>	Ctrl+Q		

7.1.2 Edit

Pilihan Menu	Shortcut	Referensi	Toolbar
 <i>Kembali</i>	Ctrl+Z	lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Ulangi</i>	Ctrl+Shift+Z	lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Ambil Fitur</i>	Ctrl+X	lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Salin Fitur</i>	Ctrl+C	lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Tempel Fitur</i>	Ctrl+V	lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
<i>Sisip fitur sebagai →</i>		lihat <i>Working with the Attribute Table</i>	
 <i>Tambah Fitur</i>	Ctrl+.	lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Pindah Fitur</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Hapus yand dipilih</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Rotasi Fitur</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Sederhanakan Fitur</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Tambah Ring</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Tambah Bagian</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Isi Ring</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Hapus Ring</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Hapus bagian</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Bentuk Ulang Fitur</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Offset Curves</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Pisah Fitur</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Bagian dipisah</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Gabung Fitur Terpilih</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Gabung Attr. Fitur Terpilih</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>
 <i>Node Tool</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Rotasi Simbol Titik</i>		lihat <i>Advanced digitizing</i>	<i>Digitalisasi Lanjutan</i>

Setelah mengaktifkan mode  *Toggle mengedit* untuk lapisan (layer), Anda menemukan ikon Add Feature di menu *Edit* tergantung pada jenis lapisan (titik, garis atau poligon).

7.1.3 Edit (ekstra)

Pilihan Menu	Shortcut	Referensi	Toolbar
 <i>Tambah Fitur</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Tambah Fitur</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
 <i>Tambah Fitur</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>

7.1.4 Tampilan

Pilihan Menu	Shortcut	Referensi	Toolbar
Pan Peta			<i>Navigasi Peta</i>
Geser Peta untuk Menyeleksi			<i>Navigasi Peta</i>
Perbesar	Ctrl++		<i>Navigasi Peta</i>
Perkecil	Ctrl+-		<i>Navigasi Peta</i>
Pilih →		lihat <i>Pilih dan lepas fitur</i>	<i>Atribut</i>
Identifikasi Fitur	Ctrl+Shift+I		<i>Atribut</i>
Mengukur →		lihat <i>Mengukur</i>	<i>Atribut</i>
Perbesar semua	kbd:Ctrl+Shift+F		<i>Navigasi Peta</i>
Perbesar ke lapisan			<i>Navigasi Peta</i>
Perbesar yang diseleksi	Ctrl+J		<i>Navigasi Peta</i>
Perbesaran Terakhir			<i>Navigasi Peta</i>
Perbesar Selanjutnya			<i>Navigasi Peta</i>
Perbesar Ukuran Aktual			<i>Navigasi Peta</i>
Dekorasi →		lihat <i>Dekorasi</i>	
Informasi Peta			<i>Atribut</i>
Bookmark Baru	Ctrl+B	lihat <i>Bookmark Spasial</i>	<i>Atribut</i>
Lihat Bookmarks	Ctrl+Shift+B	lihat <i>Bookmark Spasial</i>	<i>Atribut</i>
Refresh	Ctrl+R		<i>Navigasi Peta</i>

7.1.5 Lapisan

Pilihan Menu	Shortcut	Referensi	Toolbar
Baru →		lihat <i>Creating new Vector layers</i>	<i>Kelola Lapisan</i>
Lekatkan Lapisan dan Grup ...		lihat <i>Proyek-proyek Nesting</i>	
Tambah Lapisan Vektor	kbd:Ctrl+Shift+V	lihat <i>Pekerjaan dengan Data Vektor</i>	<i>Kelola Lapisan</i>
Tambahkan Lapisan Raster	Ctrl+Shift+R	lihat <i>Loading raster data in QGIS</i>	<i>Kelola Lapisan</i>
Tambah Lapisan PostGIS	Ctrl+Shift+D	lihat <i>PostGIS Layers</i>	<i>Kelola Lapisan</i>
Tambahkan Lapisan SpatiaLite	Ctrl+Shift+L	lihat <i>SpatiaLite Layers</i>	<i>Kelola Lapisan</i>
Tambahkan Lapisan MSSQL Spasial	Ctrl+Shift+M	lihat <i>MSSQL Spatial Layers</i>	<i>Kelola Lapisan</i>
Tambah Lapisan Oracle GeoRaster		lihat <i>Plugin Spasial Oracle GeoRaster</i>	<i>Kelola Lapisan</i>
Tambah Lapisan SQL Anywhere		lihat <i>Plugin SQL Anywhere</i>	<i>Kelola Lapisan</i>
Tambah Lapisan WMS/WMTS	Ctrl+Shift+W	lihat <i>Klien WMS/WMTS</i>	<i>Kelola Lapisan</i>
Tambah Lapisan WCS		lihat <i>Klien WCS</i>	<i>Kelola Lapisan</i>
Tambah Lapisan WFS		lihat <i>Klien WFS dan WFS-T</i>	<i>Kelola Lapisan</i>
Tambahkan Lapisan Delimited Teks		see <i>Delimited Text Files</i>	<i>Kelola Lapisan</i>
Gaya Salin		lihat <i>Style Menu</i>	
Gaya Tempel		lihat <i>Style Menu</i>	

Lanjut ke halaman berikutnya

Tabel 7.1 – lanjutan dari halaman sebelumnya

Pilihan Menu	Shortcut	Referensi	Toolbar
<i>Buka Tabel Atribut</i>		lihat <i>Working with the Attribute Table</i>	<i>Atribut</i>
<i>Toggle Mengedit</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
<i>Simpan Lapisan diedit</i>		lihat <i>Digitizing an existing layer</i>	<i>Digitalisasi</i>
<i>Diedit Sekarang →</i> <i>Simpan Sebagai...</i> <i>Simpan terpilih sebagai berkas vektor...</i>		lihat <i>Digitizing an existing layer</i> Lihat <i>Working with the Attribute Table</i>	<i>Digitalisasi</i>
<i>Buang Lapisan</i>	Ctrl+D		
<i>Lapisan Duplikat</i> <i>Atur CRS Lapisan</i> <i>Atur CRS proyek dari Lapisan Properti</i> <i>Query...</i>	Ctrl+Shift+C		
<i>Pelabelan</i>			
<i>Tambahkan ke Overview</i>	Ctrl+Shift+O		<i>Kelola Lapisan</i>
<i>Tambahkan Semua ke Overview</i>			
<i>Menghapus Semua dari Overview</i>			
<i>Lihat Semua Lapisan</i>	Ctrl+Shift+U		<i>Kelola Lapisan</i>
<i>Sembunyikan Semua Lapisan</i>	Ctrl+Shift+H		<i>Kelola Lapisan</i>

7.1.6 Pengaturan






Pilihan Menu	Shortcut	Referensi	Toolbar
<i>Panel →</i> <i>Toolbar →</i> <i>Toggle Mode Layar Penuh</i>	F 11	Lihat <i>Panel dan Toolbar</i> Lihat <i>Panel dan Toolbar</i>	
<i>Proyek Properti ...</i>	Ctrl+Shift+P	lihat <i>Proyek</i>	
<i>Ubah CRS ...</i> <i>Pengelola Gaya...</i>		lihat <i>Custom Coordinate Reference System</i> lihat <i>Presentation</i>	
<i>Konfigurasi shortcut ...</i> <i>Kustomisasi ...</i> <i>Opsi ...</i> <i>Opsi Snapping ...</i>		lihat <i>Penyesuaian (Customization)</i> lihat <i>Opsi</i>	

7.1.7 Plugin

Pilihan Menu	Shortcut	Referensi	Toolbar
<i>Kelola dan Pasang Plugin</i> <i>Konsol Python</i>		lihat <i>The Plugins Dialog</i>	

Saat memulai QGIS untuk pertama kali tidak semua inti plugin dimuat.

7.1.8 Vektor

Pilihan Menu	Shortcut	Referensi	Toolbar
<i>Open Street Map</i> →		lihat <i>Loading OpenStreetMap Vectors</i>	
 <i>Peralatan Analisis</i> →		lihat <i>Plugin fTools</i>	
 <i>Peralatan Riset</i> →		lihat <i>Plugin fTools</i>	
 <i>Peralatan Geoprocessing</i> →		lihat <i>Plugin fTools</i>	
 <i>Peralatan Geometri</i> →		lihat <i>Plugin fTools</i>	
 <i>Peralatan Manajemen Data</i> →		lihat <i>Plugin fTools</i>	







Saat memulai QGIS untuk pertama kali tidak semua inti plugin dimuat.

7.1.9 Raster

Pilihan Menu	Shortcut	Referensi	Toolbar
<i>Kalkulator Raster ...</i>		lihat <i>Kalkulator Raster</i>	







Saat memulai QGIS untuk pertama kali tidak semua inti plugin dimuat.


7.1.10 Pengolahan




Pilihan Menu	Shortcut	Referensi	Toolbar
 <i>Toolbox</i>		lihat <i>The toolbox</i>	
 <i>Grafis Modeler</i>		lihat <i>The graphical modeler</i>	
 <i>Sejarah dan log</i>		lihat <i>Manajer riwayat</i>	
 <i>Opsi dan Konfigurasi</i>		lihat <i>Configuring the processing framework</i>	
 <i>Penampil hasil</i>		lihat <i>Configuring external applications</i>	
 <i>Perintah</i>	Ctrl+Alt+M	lihat <i>The QGIS Commander</i>	

Saat memulai QGIS untuk pertama kali tidak semua inti plugin dimuat.

7.1.11 Bantuan

Pilihan Menu	Shortcut	Referensi	Toolbar
 <i>Konten Bantuan</i>	F1		<i>Bantuan</i>
 <i>Apakah ini?</i> <i>Dokumentasi API</i> <i>Butuh dukungan komersial?</i>	Shift+F1		<i>Bantuan</i>
 <i>Beranda QGIS</i>	Ctrl+H		
 <i>Periksa versi QGIS</i>			
 <i>Tentang</i>			
 <i>Sponsor QGIS</i>			

Harap dicatat bahwa untuk Linux  item Menu Bar yang tercantum di atas adalah yang standar di window manager KDE. Di GNOME, menu *Pengaturan* memiliki isi yang berbeda dan item bisa ditemukan di sini:

 <i>Properti Proyek</i>	<i>Proyek</i>
 <i>Opsi</i>	<i>Edit</i>
 <i>Konfigurasi Shortcuts</i>	<i>Edit</i>
<i>Pengelola Gaya</i>	<i>Edit</i>
 <i>Modifikasi CRS</i>	<i>Edit</i>
<i>Panel →</i>	<i>Tampilan</i>
<i>Toolbar →</i>	<i>Tampilan</i>
<i>Toggle Mode Layar Penuh</i>	<i>Tampilan</i>
<i>Slider Skala Tile</i>	<i>Tampilan</i>
<i>Pelacakan GPS</i>	<i>Tampilan</i>

7.2 Toolbar

Toolbar menyediakan akses ke sebagian besar fungsi yang sama seperti menu, ditambah alat tambahan untuk berinteraksi dengan peta. Setiap item toolbar memiliki popup bantuan yang tersedia. Tahan tetikus anda ke atas item dan deskripsi singkat mengenai tujuan alat itu akan ditampilkan.



Setiap menubar dipindah kesekitarnya sesuai dengan kebutuhan Anda. Selain itu setiap menubar dapat dimatikan menggunakan tombol kanan tetikus pada menu konteks Anda, arahkan tetikus ke toolbar (baca juga [Panel dan Toolbar](#)).


Tip: Mengembalikan toolbar

Jika Anda tidak sengaja telah menyembunyikan semua toolbar Anda, Anda dapat mengembalikannya dengan memilih menu opsi *Pengaturan → Toolbar →*. Jika toolbar menghilang di bawah OS Windows, tampaknya menjadi masalah di QGIS dari waktu ke waktu, Anda harus menghapus `\HKEY_CURRENT_USER\Software\QGIS\qgis\UI\state` di registry. Ketika Anda hidupkan ulang QGIS, kuncinya ditulis lagi secara standar, dan semua toolbar terlihat kembali.

7.3 Legenda Peta

The map legend area lists all the layers in the project. The checkbox in each legend entry can be used to show or hide the layer. The Legend toolbar in the map legend are list allow you to **Add group, Manage Layer Visibility** of all layers or manage preset layers combination, **Filter Legend by Map Content, Expand All** or **Collapse All**

and **Remove Layer or Group**. The button  allows you to add **Presets** views in the legend. It means that you can choose to display some layer with specific categorization and add this view to the **Presets** list. To add a preset view just click on ,


choose *Add Preset...* from the drop down menu and give a name to the preset. After that you will see a list with all the presets that you can recall pressing on the  button.

All the added presets are also present in the map composer in order to allow you to create a map layout based on your specific views (see [Main properties](#)).

Suatu lapisan (layer) dapat dipilih dan digeser ke atas atau kebawah pada legenda menjadi Z-urutan. Z-urutan berarti bahwa lapisan yang terdaftar di bagian atas legenda digambar lapisan bawahnya tercantum dalam legenda.


Catatan: Perilaku ini dapat diganti dengan panel ‘urutan lapisan’.

Lapisan di jendela legenda dapat dikelompokkan dalam grup. Ada dua cara untuk melakukannya:

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. Pilih beberapa lapisan (layer), klik kanan pada jendela legenda dan pilih *Grup Terpilih*. Lapisan-lapisan yang dipilih secara otomatis akan menjadi satu grup baru.

Untuk mengeluarkan lapisan dari grup, Anda bisa menggesernya keluar, atau klik kanan dan pilih *Ubah jadi tingkat teratas*. Grup dapat masuk kedalam grup lain.

Kotak centang grup akan memunculkan atau menyembunyikan lapisan dalam grup dengan satu klik.

Isi dari konteks menu tombol kanan tetikus tergantung pada item legenda yang dipilih lapisan (layer) raster atau vektor. Untuk lapisan vektor GRASS  *Toggle mengedit* tidak tersedia. Lihat bagian *Digitizing and editing a GRASS vector layer* untuk informasi menyunting lapisan (layer) vektor GRASS.

Tombol kanan tetikus untuk lapisan raster

- *Perbesar lapisan extent*
- *Tampilkan di overview*
- *Perbesar Skala Terbaik (100%)*
- *Peregangan Menggunakan Luas Terkini*
- *Buang*
- *Duplikat*
- *Set Layer Scale Visibility*
- *Atur CRS Lapisan*
- *Atur CRS Proyek dari Lapisan*
- *Simpan sebagai ...*
- *Save As Layer Definition Style*
- *Properti*
- *Ubah Nama*
- *Gaya Salin*

Selain itu, menurut posisi dan seleksi lapisan (layer)

- *Ubah jadi item tingkat teratas*
- *Grup dipilih*

Tombol kanan tetikus menu untuk lapisan vektor

- *Perbesar ke Lapisan Extent*
- *Tampilkan di Overview*
- *Buang*
- *Duplikat*
- *Set Layer Scale Visibility*
- *Atur CRS Lapisan*
- *Atur CRS Proyek dari Lapisan*
- *Buka Tabel Atribut*
- *Toggle Mengedit* (tidak tersedia untuk Lapisan GRASS)
- *Simpan Sebagai ...*
- *Save As Layer Definition Style*
- *Saring*
- *Tampilkan Fitur Hitung*
- *Properti*
- *Ubah Nama*

- *Gaya Salin*

Selain itu, menurut posisi dan seleksi lapisan (layer)

- *Ubah jadi item tingkat teratas*
- *Grup dipilih*

Tombol kanan tetikus menu untuk grup lapisan

- *Perbesar ke Grup*
- *Buang*
- *Atur Grup CRS*
- *Ubah Nama*
- *Add Group*

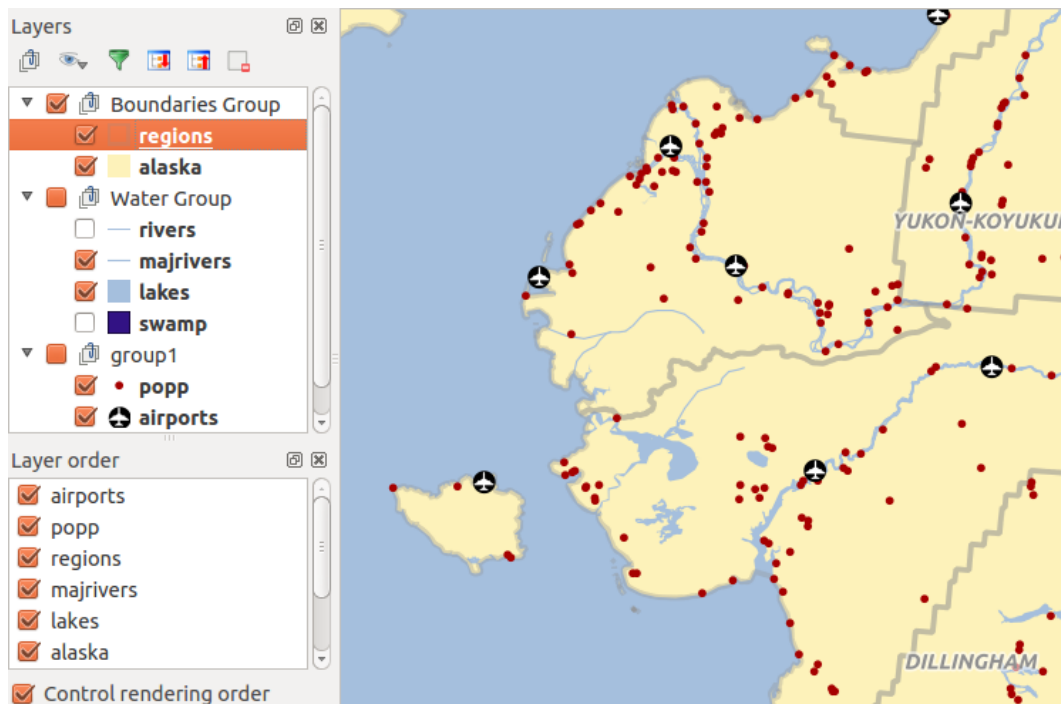
Saat ini memungkinkan memilih lebih dari satu lapisan atau grup pada waktu yang sama dengan menekan tombol `Ctrl` sambil memilih lapisan dengan tombol kiri tetikus. Kemudian Anda dapat memindah semua lapisan terpilih ke dalam grup baru pada waktu yang sama.

Anda juga dapat menghapus lebih dari satu Lapisan atau Grup sekaligus dengan memilih beberapa Lapisan (layer) dengan menekan tombol `Ctrl` dan setelah itu `Ctrl+D`. Dengan cara ini semua Lapisan atau Grup terpilih akan dibuang dari daftar lapisan.

7.3.1 Bekerja dengan Legenda urutan lapisan tersendiri

Terdapat widget yang memungkinkan untuk mendefinisikan urutan legenda gambar independen. Anda dapat mengaktifkannya dari menu *Pengaturan* → *Panel* → *Urutan Lapisan*. Di sini menentukan urutan gambar dari lapisan dalam tampilan peta. Melakukan hal ini memungkinkan untuk mengurutkan lapisan Anda dalam urutan kepentingan, sebagai contoh, tapi masih menampilkan mereka dalam urutan yang benar (lihat *figure_layer_order*).

Mengaktifkan kotak *Kontrol urutan rendering* bawah daftar lapisan akan menyebabkan kembali ke suatu perilaku standar.



Gambar 7.2: Mendefinisikan legenda urutan lapisan tersendiri 

7.4 Tampilan Peta

Ini “business end” dari QGIS - peta tampil di area ini! Peta yang ditampilkan dalam jendela ini tergantung pada lapisan vektor dan raster yang telah Anda pilih (lihat bagian informasi lebih lanjut tentang cara untuk memuat lapisan). Tampilan peta dapat digeser (mengalihkan fokus tampilan peta untuk daerah lain) dan memperbesar atau memperkecilnya. Berbagai operasi lainnya dapat dilakukan pada peta seperti yang dijelaskan dalam deskripsi toolbar di atas. Tampilan peta dan legenda terikat erat satu sama lain - peta dalam tampilan mencerminkan perubahan yang Anda buat di daerah legenda.

Tip: Perbesar peta dengan tetikus

Anda dapat menggunakan tetikus untuk memperbesar dan memperkecil peta. Tempatkan kursor tetikus di dalam area peta dan roll roda depan (jauh dari Anda) untuk memperbesar dan belakang (ke arah Anda) untuk memperkecil. Posisi kursor tetikus adalah pusat di mana perubahan terjadi. Anda dapat menyesuaikan perilaku pembesaran roda tetikus menggunakan menu *Alat Peta* dalam menu *Pengaturan* → *Opsi*.

Tip: Menggeser peta dengan tombol panah dan spasi bar


Anda dapat menggunakan tombol panah untuk menggeser peta. empatkan kursor tetikus dalam area peta dan klik panah kanan untuk menggeser ke Timur, panah kiri untuk menggeser ke Barat, panah atas untuk menggeser ke Utara dan pana bawah untuk menggeser ke Selatan. Anda juga dapat menggeser peta dengan menggunakan bar spasi atau klik pada tetikus: hanya menggerakkan tetikus bersamaan menekan bar spasi atau klik tetikus.

7.5 Status Bar

Bar status melihatkan posisi kursor Anda dalam koordinat peta (misal meter atau derajat desimal) sesuai dengan titik kursor tetikus bergerak pada tampilan peta. Di sebelah kiri tampilan koordinat di bar status adalah tombol kecil yang akan beralih antara menampilkan posisi koordinat atau menampilkan luasan peta saat Anda menggeser dan memperbesar/memperkecil tampilan peta.


Sebelah tampilan koordinat Anda menemukan tampilan skala. Itu menunjukkan skala dari tampilan peta. Jika Anda memperbesar atau memperkecil QGIS berisi skala saat ini. Ada pemilih skala yang memungkinkan Anda untuk memilih skala standar dari 1:500 sampai 1:1000000.

Suatu perkembangan (progres) bar di status bar menunjukkan kemajuan (progres) render karena setiap lapisan yang diubah di tampilan peta. Dalam beberapa kasus, seperti pengumpulan statistik di lapisan raster, progress bar akan menunjukkan status panjangnya operasi.

Jika ada plugin baru atau pembaruan plugin tersedia, Anda akan melihat pesan sebelah kiri dari bar status. Di sisi kanan status bar adalah kotak centang kecil yang dapat digunakan untuk lapisan sementara yang diberikan ke tampilan peta (lihat Bagian *Rendering*). Ikon  menghentikan proses rendering peta sekarang.

Sebelah kanan dari fungsi render, Anda akan menemukan kode EPSG dari CRS proyek sekarang dan ikon proyektor. Mengklik ini akan membuka properti proyeksi untuk proyek saat ini.

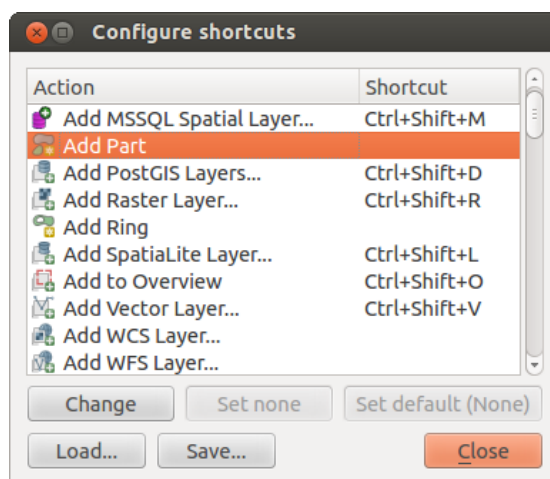
Tip: Menghitung skala koreksi dari kanvas peta Anda

Saat Anda memulai QGIS, derajat merupakan unit standar, dan QGIS memberitahu bahwa setiap koordinat pada lapisan Anda dalam derajat. Untuk mengubah nilai skala, Anda juga dapat mengubahnya ke satuan meter secara manual di tab *Umum* dalam *Pengaturan* → *Proyek Properti* atau Anda bisa memilih Coordinate Reference System (CRS) proyek dengan klik ikon  CRS status di bagian kanan bawah dari status bar. Dalam kasus terakhir, unit ditetapkan untuk menentukan proyeksi proyek, (misalnya ‘+unit=m’).

Peralatan Umum

8.1 Shortcut Papanketik

QGIS menyediakan standar shortcut papan ketik untuk banyak fitur. Anda menemukannya di Bagian *Bar Menu*. Selain itu di menu opsi *Pengaturan* → *Konfigurasi Shortcut* memungkinkan merubah standar shortcut papanketik dan menambah shortcut papanketik baru ke fitur QGIS.



Gambar 8.1: Tentukan pilihan shortcut 🐧 (Gnome)

Konfigurasi sederhana. Hanya pilih fitur dari daftar dan klik pada [**Ubah**], [**Atur none**] atau [**Atur Standar**]. Setelah Anda telah menemukan konfigurasi, Anda dapat menyimpannya sebagai berkas XML dan muat pada instalasi QGIS lain.

8.2 Konteks Bantuan

Saat Anda membutuhkan bantuan dengan topik yang spesifik, Anda dapat mengakses konteks bantuan melalui tombol [**Bantuan**] tersedia disebagian besar dialog - harap dicatat bahwa plugin pihak ketiga dapat mengarah ke halaman web khusus.

8.3 Rendering

Secara standar, QGIS membuat semua lapisan terlihat setiap kali kanvas peta di-refresh. Peristiwa yang memicu refresh kanvas peta meliputi:

- Menambahkan lapisan (layer)

- Menggeser atau memperbesar
- Mengukur jendela QGIS
- Merubah visibilitas dari lapisan (layer)

QGIS memungkinkan Anda mengontrol proses rendering dalam beberapa cara.

8.3.1 Scale Dependent Rendering

Skala render memungkinkan Anda untuk menentukan skala minimum dan skala maksimum di mana lapisan akan terlihat. Untuk mengatur skala render bergantung, buka dialog *Properti* dengan mengklik-dobel pada lapisan di legenda. Pada tab *Umum* klik pada kotak centang *Skala bergantung pada visibilitas* untuk mengaktifkan fitur, kemudian atur nilai maksimum dan minimum skala.

Anda dapat menentukan nilai skala dengan terlebih dahulu perbesar ke tingkat yang ingin Anda gunakan dan mencatat nilai skala pada status bar QGIS.

8.3.2 Mengontrol Rendering Peta

Me-render peta dapat dikontrol dalam berbagai cara, seperti yang dijelaskan di bawah ini.

Menunda Rendering

Untuk menunda rendering, klik kotak centang *Render* di sudut bawah kanan dari status bar. Ketika kotak centang *Render* tidak diaktifkan, QGIS tidak menggambar ulang kanvas dalam menanggapi setiap kejadian yang telah diuraikan dalam Bagian *Rendering*. Contoh ketika Anda mungkin ingin menunda render meliputi:

- Menambah banyak lapisan dan melambangkan mereka sebelum menggambar
- Menambahkan satu atau lebih lapisan besar dan mengatur ketergantungan skala sebelum menggambar
- Menambahkan satu atau lebih lapisan besar dan perbesar ke tampilan spesifik sebelum menggambar
- Kombinasi dari yang ada di atas

Aktifkan kotak centang *Render* mengaktifkan rendering dan menyebabkan refresh langsung dari kanvas peta.

Pengaturan Lapisan Tambah Opsi

Anda dapat mengatur pilihan untuk selalu memuat lapisan baru tanpa menggambar mereka. Ini berarti lapisan akan ditambahkan ke peta, namun visibilitas kotak centang dalam legenda akan dicentang secara default. Untuk mengatur opsi ini, pilih menu opsi *Pengaturan* → *Opsi* dan klik tab *Rendering*. Hapus centang *Secara standar lapisan baru ditambahkan ke peta selalu ditampilkan*. Setiap lapisan ditambahkan ke peta akan tidak terlihat secara standar.

Menghentikan Rendering

Untuk menghentikan penggambaran peta, tekan tombol ESC. Ini akan menghentikan refresh kanvas peta dan menghentikan proses penggambaran peta. Hal ini mungkin membutuhkan waktu selama menekan ESC dan penggambaran peta terhenti.

Catatan: Saat ini tidak memungkinkan untuk menghentikan rendering - ini dinonaktifkan di port qt4 karena masalah antarmuka pengguna (UI) dan konflik (crash).

Memperbarui tampilan peta sebelum rendering

Anda dapat mengatur pilihan untuk memperbarui tampilan peta sebagai fitur. Secara standar, QGIS tidak menampilkan semua fitur untuk lapisan sampai seluruh lapisan telah dirender. Untuk memperbarui tampilan seperti fitur dibaca dari datastore, pilih opsi menu *Pengaturan* → *Opsi* klik pada menu *Rendering*. Mengatur jumlah fitur untuk nilai yang sesuai untuk memperbarui tampilan selama rendering. Menetapkan nilai 0 menonaktifkan pembaruan selama menggambar (ini adalah default). Menetapkan nilai terlalu rendah akan menghasilkan kinerja yang buruk pada kanvas peta terus diperbarui selama pembacaan fitur. Nilai yang disarankan untuk memulai adalah 500.

Mempengaruhi Kualitas Rendering

Untuk mempengaruhi kualitas dari peta Anda memiliki 2 opsi. Pilih opsi menu *Pengaturan* → *Opsi* klik pada tab *Rendering* dan pilih atau tidak kotak centang berikut.

- *Membuat garis tampil kurang baik dengan mengorbankan beberapa kinerja menggambar*
- *Memperbaiki masalah isi poligon*

Mempercepat rendering

Ada dua pengaturan yang memungkinkan Anda untuk meningkatkan kecepatan rendering. Buka opsi dialog QGIS menggunakan *Pengaturan* → *Opsi*, ke tab *Rendering* dan pilih atau tidak kotak centang berikut.


- *Enable back buffer*. Hal ini memberikan performa grafis yang lebih baik pada kemungkinan kehilangan biaya untuk membatalkan rendering dan secara bertahap menggambar fitur. Jika dicentang, Anda dapat mengatur *Jumlah fitur untuk menggambar sebelum memperbarui tampilan*, jika pilihan ini tidak aktif.
- *Gunakan render caching yang memungkinkan mempercepat gambar ulang*


8.4 Mengukur

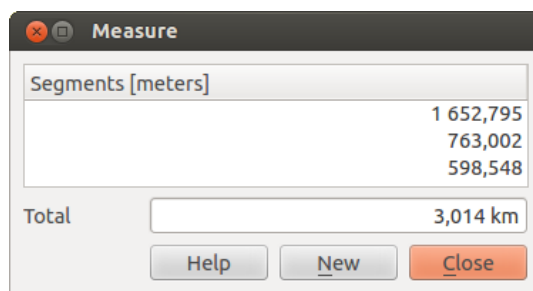
Mengukur peta dalam proyeksi sistem koordinat (misal UTM) dan data belum terproyeksi. Jika peta dimuat didefinisikan dengan sistem koordinat geografis (lintang/bujur), hasil dari garis atau daerah pengukuran akan salah. Untuk memperbaiki ini, Anda perlu mengatur peta sesuai sistem koordinat (lihat Bagian *Working with Projections*). Semua modul pengukuran juga menggunakan pengaturan snapping dari modul digitalisasi. Hal ini berguna, jika Anda ingin mengukur garis atau area di lapisan vektor.

Untuk memilih alat ukur, klik di  dan pilih alat yang ingin digunakan.

8.4.1 Mengukur panjang, area dan sudut

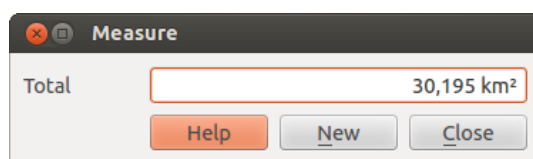
 **Mengukur Garis:** QGIS mampu mengukur jarak nyata antar poin yang diberikan sesuai dengan ellipsoid yang didefinisikan. Untuk mengonfigurasinya, pilih menu opsi *Pengaturan* → *Opsi*, klik tab *Peralatan Peta* dan pilih ellipsoid yang tepat. Di sana Anda juga dapat mendefinisikan warna karet gelang (rubberband) dan satuan pengukuran pilihan Anda (meter atau feet) dan satuan sudut (derajat, radian dan gon). Kemudian memungkinkan Anda untuk mengklik titik pada peta. Setiap panjang-segmen serta total muncul dalam jendela-ukuran. Untuk menghentikan pengukuran klik tombol tetikus sebelah kanan.

 **Mengukur Area:** Area juga dapat diukur. Pada jendela mengukur muncul ukuran daerah akumulasi. Selain itu, alat ukur akan mengambil (snap) ke lapisan yang sedang dipilih, asalkan lapisan yang memiliki toleransi yang ditetapkan. (Lihat Bagian *Setting the Snapping Tolerance and Search Radius*). Jadi jika Anda ingin mengukur persis sepanjang fitur garis, atau sekitar fitur poligon, pertama kali atur toleransi snap, kemudian pilih lapisan.




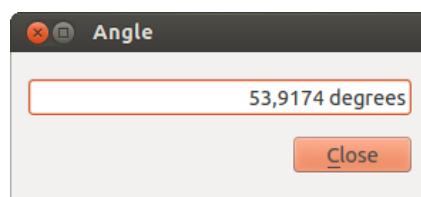
Gambar 8.2: Mengukur Panjang 🐧 (Gnome)

Sekarang, ketika menggunakan alat ukur, setiap klik tetikus (dalam pengaturan toleransi) akan merekam (snap) ke lapisan itu.




Gambar 8.3: Mengukur Area 🐧 (Gnome)






 **Mengukur Sudut**: Anda juga bisa mengukur sudut. Cursor menjadi cross-shaped. Klik untuk menggambar segmen pertama dari sudut yang ingin diukur, kemudian memindahkan cursor untuk menggambar sudut yang diinginkan. Alat ukur akan ditampilkan dalam dialog pop-up.




Gambar 8.4: Mengukur Sudut 🐧 (Gnome)

8.4.2 Pilih dan lepas fitur

Toolbar QGIS menyediakan beberapa alat untuk memilih fitur dalam kanvas peta. Untuk memilih satu atau beberapa fitur klik pada  dan pilih perangkat Anda:


-  Pilih Fitur Tunggal
-  Pilih Fitur dari Rectangle
-  Pilih Fitur dari Poligon
-  Pilih Fitur dari Freehand
-  Pilih Fitur dari Radius

Untuk melepas semua fitur yang dipilih klik di  Lepas fitur dari semua lapisan.

 **Select feature using an expression** allow user to select feature using expression dialog. See *Expressions* chapter for some example.

Users can save features selection into a **New Memory Vector Layer** or a **New Vector Layer** using *Edit* → *Paste Feature as ...* and choose the mode you want.

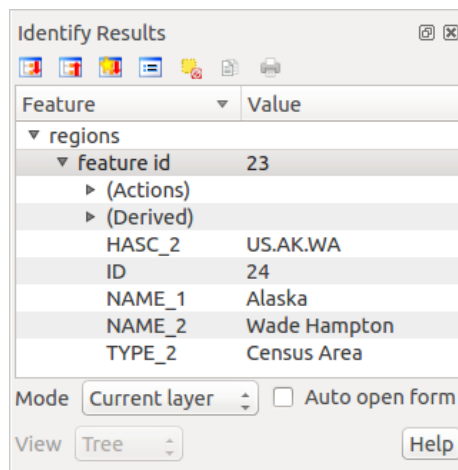
8.5 Fitur Identifikasi


Fitur identifikasi memungkinkan berinteraksi dengan kanvas peta untuk mendapatkan informasi fitur pada sebuah jendela pop-up. Identifikasi fitur menggunakan *Tampilan* → *Identifikasi fitur* atau tekan **Ctrl + Shift + I**, atau klik ikon toolbar  Identifikasi fitur.

Jika Anda klik beberapa fitur, dialog *Hasil Identifikasi* akan mendaftar semua data atribut dari semua fitur. Item pertama adalah jumlah item dalam daftar hasil diikuti dengan nama lapisan. Kemudian, anak pertama akan menjadi nama sebuah kolom dengan nilainya. Akhirnya semua informasi dari fitur tersebut akan ditampilkan.






Jendela ini dapat disesuaikan untuk menampilkan kolom kustom namun secara default akan menampilkan tiga jenis informasi:

- Aksi-aksi: Aksi-aksi ditambahkan untuk mengidentifikasi fitur windows. Ketika mengklik pada aksi label, aksi akan berjalan. Secara default hanya satu tindakan ditambahkan untuk melihat form fitur untuk mengedit.
- Derived: informasi mereka dihitung atau berasal dari informasi lainnya. Anda bisa menemukan koordinat dengan diklik, koordinat X dan Y, area dalam satuan peta dan parameter peta dalam unit peta untuk poligon, panjang unit peta untuk garis dan id fitur.
- Data atribut: Daftar kolom atribut dari data



Gambar 8.5: Dialog identifikasi fitur  (Gnome)

Di bagian bawah dari jendela, Anda memiliki lima ikon:

-  Expand tree
-  Collapse tree
-  Default behaviour
-  Salin atribut
-  Print selected HTML response

Fungsi lain dapat ditemukan dalam menu konteks dari ditentukannya item. Sebagai contoh, dari menu konteks Anda dapat:


- Lihat form fitur
- Perbesar ke fitur

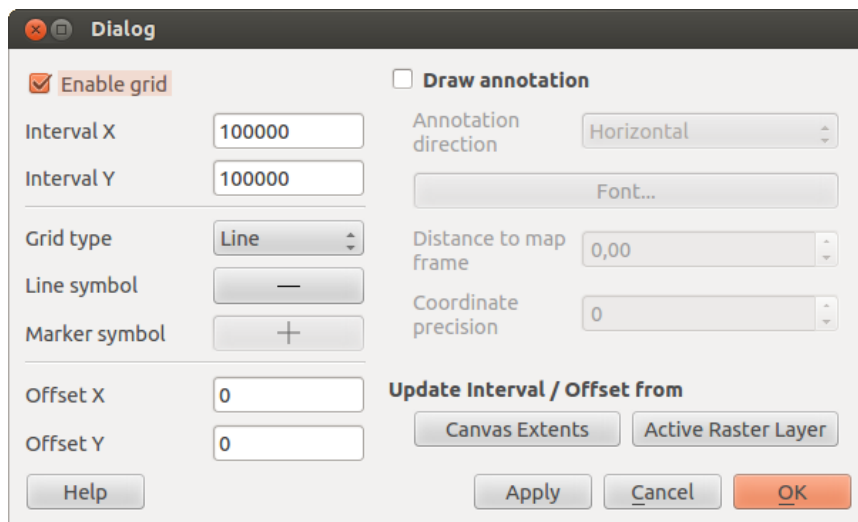
- Salin fitur: salin semua fitur geometri dan atribut
- Toggle feature selection: adds identified feature to selection
- Salin nilai atribut: hanya menyalin nilai dari atribut yang Anda klik
- Salin atribut fitur: hanya menyalin atribut
- Bersihkan hasilnya: Hapus hasil di jendela
- Bersihkan highlight: Hapus fitur highlight pada peta
- Highlight semua
- Highlight lapisan
- Aktifkan lapisan: Pilih lapisan yang akan diaktifkan
- Properti lapisa: buka jendel properti lapisan
- Perluas (expand) semua
- Runtuh (collapse) semua


8.6 Dekorasi


Dekorasi dari QGIS termasuk Grid, Label Hak Cipta, Panah Utara, dan Bar Skala. Mereka digunakan untuk 'dekorasi' peta dengan menambahkan elemen peta.

8.6.1 Kisi (Grid)

 Grid memungkinkan untuk menambahkan koordinat kisi (grid) dan koordinat anotasi untuk kanvas peta.




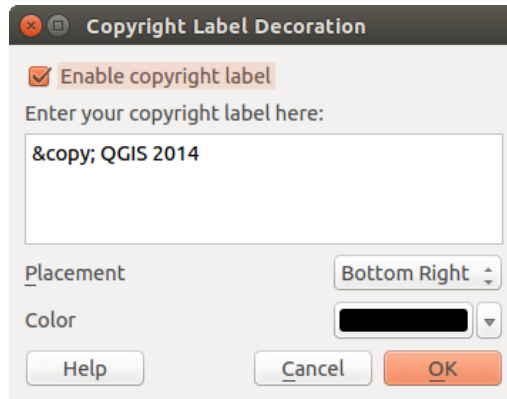
Gambar 8.6: Dialog Kisi (Grid) 

1. Pilih dari menu *Tampilan* → *Dekorasi* → *Kisi*. Mulai dialog (lihat *figure_decorations_1*).
2. Aktifkan kotak centang  *Aktifkan Kisi* dan menetapkan definisi kisi (grid) sesuai dengan lapisan yang dimuat dalam kanvas peta.
3. Aktifkan kotak centang *Gambar anotasi* dan menetapkan definisi anotasi sesuai dengan lapisan yang dimuat dalam kanvas peta.
4. Klik [**Terapkan**] untuk memverifikasi bahwa itu tampak seperti yang diharapkan.


5. Klik **[OK]** untuk menutup dialog.

8.6.2 Label Hak Cipta

 Label Hak Cipta menambahkan label hak cipta menggunakan teks untuk peta.




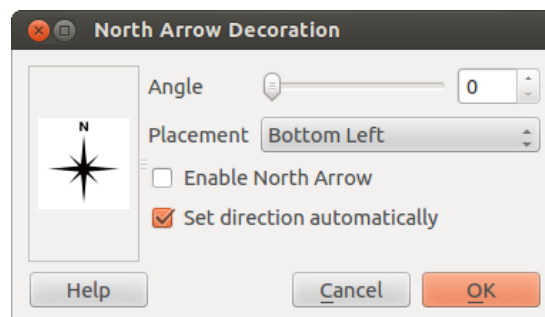
Gambar 8.7: Dialog Hak Cipta 

1. Pilih dari menu *Tampilan* → *Dekorasi* → *Label hak Cipta*. (lihat [figure_decorations_2](#)).
2. Masukkan teks yang Anda ingin tempatkan di peta. Anda bisa menggunakan HTML seperti dalam contoh
3. Pilih penempatan label dari kotak kombo *Penempatan* .
4. Pastikan kotak centang telah ditandai/aktifkan *Aktifkan Label Hak Cipta*
5. Klik **[OK]**.

Dalam contoh di atas, yang merupakan default, QGIS menempatkan simbol hak cipta diikuti dengan tanggal di bagian bawah sudut kanan dari kanvas peta.


8.6.3 Panah Utara

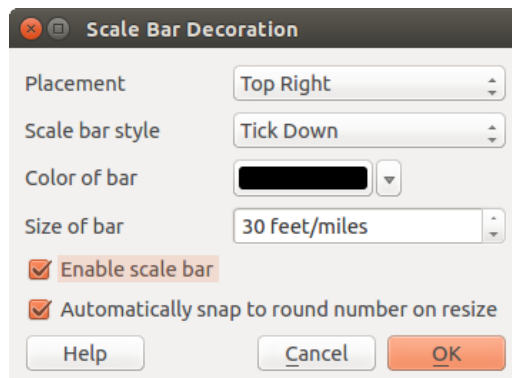
 Panah Utara menempatkan panah utara sederhana di kanvas peta. Saat ini hanya ada satu gaya yang tersedia. Anda dapat mengatur sudut panah atau membiarkan QGIS mengatur arah secara otomatis. Jika Anda memilih untuk membiarkan QGIS menentukan arah, itu membuat QGIS menebak yang terbaik bagaimana panah harus berorientasi. Untuk penempatan panah, Anda memiliki empat pilihan, sesuai dengan empat penjuru kanvas peta.



Gambar 8.8: Dialog Panah Utara 

8.6.4 Bar Skala





 Skala Bar menambahkan bar skala sederhana untuk kanvas peta. Anda bisa mengontrol gaya dan penempatan, serta pelabelan bar.



Gambar 8.9: Dialog Bar Skala 

QGIS hanya mendukung menampilkan skala dalam satuan yang sama dengan kerangka peta Anda. Jadi jika satuan lapisan Anda dalam meter, Anda tidak dapat membuat skala bar dalam kaki (feet). Demikian juga jika Anda menggunakan derajat desimal, Anda tidak dapat membuat skala bar untuk menampilkan jarak dalam meter.


Menambahkan bar skala:

1. Pilih dari menu *Tampilan* → *Dekorasi* → *Skala Bar* (lihat [figure_decorations_4](#))
2. Pilih penempatan label dari kotak kombo *Penempatan* 
3. Pilih gaya dari kotak kombo *Gaya skala bar* 
4. Pilih warna bar *Warna bar*  atau gunakan hitam warna default.
5. Atur ukuran bar dan labelnya *Ukuran bar* .
6. Pastikan kotak centang sudah aktif *Aktifkan skala bar*
7. Opsional, centang *Automatically snap to round number on resize*.
8. Klik [OK].

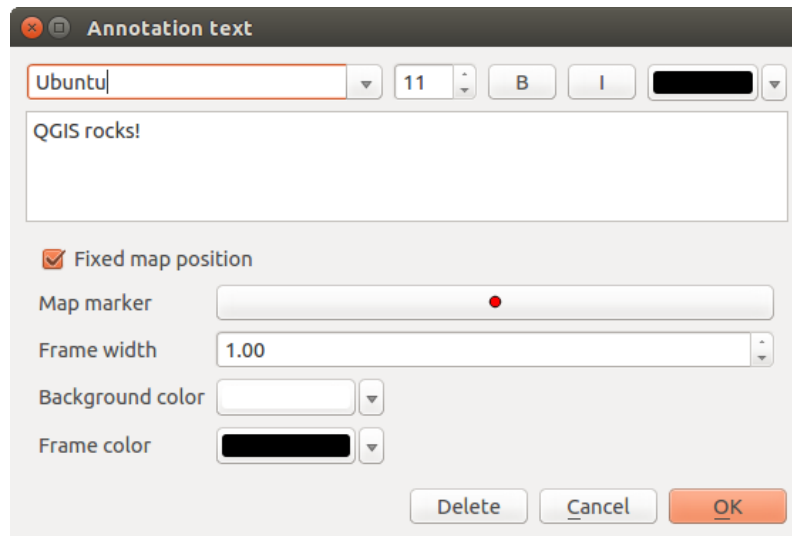
Tip: Pengaturan Dekorasi

Saat Anda menyimpan sebuah proyek .qgs, setiap perubahan yang Anda buat pada Kisi (Grid), Panah Utara, Skala Bar dan Hak Cipta akan disimpan dalam proyek dan dikembalikan pada saat Anda memuat proyek.


8.7 Peralatan Anotasi

Peralatan  Anotasi Teks dalam toolbar atribut memberikan kemungkinan untuk menempatkan teks diformat dalam balon pada kanvas Peta QGIS. Gunakan alat *Anotasi Teks* dan klik ke dalam kanvas peta.


Dobel klik pada item membuka dialog dengan berbagai pilihan. Ada editor teks untuk memasukkan teks yang diformat dan pengaturan item lain. Misalnya ada pilihan memiliki item ditempatkan pada posisi peta (ditampilkan dengan simbol penanda) atau memiliki item pada posisi layar (tidak berhubungan dengan peta). Item ini bisa dipindahkan dengan posisi peta (geser penanda peta) atau hanya dengan memindahkan balon. Ikon adalah bagian dari tema GIS, dan digunakan secara default dalam tema-tema lain juga.




Gambar 8.10: Dialog teks anotasi 

Alat  Pindah Anotasi memungkinkan memindah anotasi pada kanvas peta.


8.7.1 Anotasi HTML

Alat  Anotasi Html dalam toolbar atribut memberikan kemungkinan untuk menempatkan isi berkas HTML dalam balon pada kanvas Peta QGIS. Gunakan alat *Anotasi Html* dan klik kedalam kanvas peta dan menambahkan path ke berkas html ke dalam dialog.

8.7.2 Anotasi SVG

Alat  Anotasi SVG dalam toolbar atribut memberikan kemungkinan untuk menempatkan simbol SVG dalam balon pada kanvas Peta QGIS. Gunakan alat *Anotasi SVG* dan klik kedalam kanvas peta dan menambahkan path ke berkas SVG ke dalam dialog.

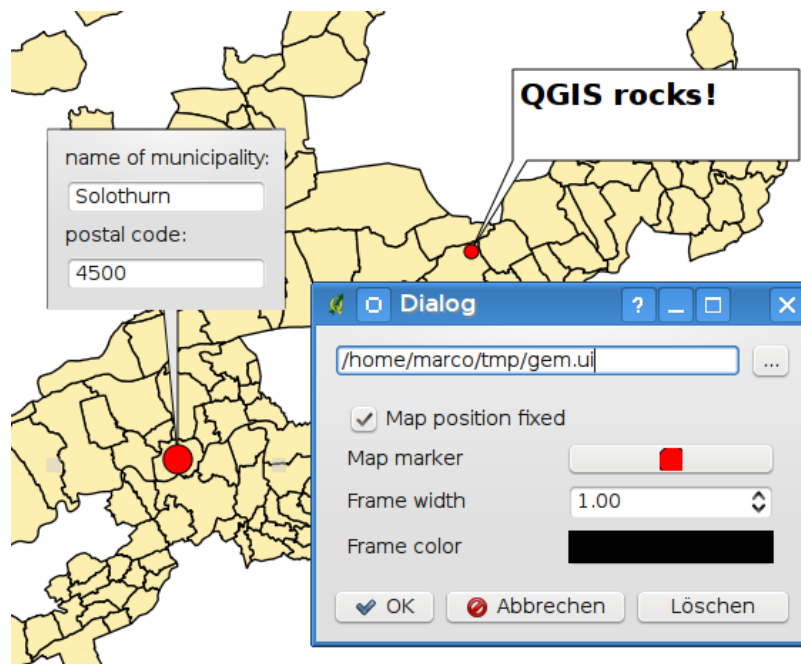
8.7.3 Form anotasi

Selain itu Anda juga dapat membuat form anotasi Anda sendiri. Alat  Form Anotasi ini berguna untuk menampilkan atribut dari lapisan vektor dalam form desainer qt disesuaikan (lihat [figure_custom_annotation](#)). Hal ini mirip dengan form desainer alat *Identifikasi fitur*, tetapi ditampilkan dalam item anotasi. Lihat juga video <https://www.youtube.com/watch?v=0pDBuSbQ02o> dari Tim Sutton untuk informasi lebih lanjut.

Catatan: Jika Anda menekan `Ctrl+T` sementara alat *Anotasi* aktif (anotasi bergerak, anotasi teks, form anotasi), visibilitas item yang terbalik.

8.8 Bookmark Spasial

Bookmark spasial memungkinkan Anda untuk “bookmark” lokasi geografis dan kembali ke spasial ini suatu saat nanti.



Gambar 8.11: Disesuaikan form anotasi bentuk desainer qt 

8.8.1 Membuat Bookmark

Membuat Bookmark

1. Perbesar atau geser ke interes area.
2. Pilih menu opsi *Tampilan* → *Bookmark Baru* atau tekan `Ctrl-B`.
3. Masukkan nama deskripsi dari bookmark (batas 255 karakter).
4. Tekan `Enter` untuk menambahkan bookmark atau **[Delete]** untuk menghapus bookmark.

Catatan Anda bisa memiliki bookmark banyak dengan satu nama.

8.8.2 Bekerja dengan Bookmark

Menggunakan atau mengelola bookmark, pilih menu opsi *Tampilan* → *Tampilkan Bookmark*. Dialog *Geospatial Bookmark* memungkinkan Anda untuk memperbesar atau menghapus bookmark. Anda tidak dapat mengedit nama bookmark atau koordinat.

8.8.3 Perbesar ke Bookmark

Dari dialog *Geospasial Bookmark*, pilih bookmark yang diinginkan dengan mengkliknya, kemudian klik **[Perbesar]**. Anda juga bisa memperbesar ke bookmark dengan double-klik padanya.

8.8.4 Menghapus Bookmark


Menghapus bookmark dari dialog *Geospasial Bookmark*, klik padanya dan kemudian klik **[Hapus]**. Konfirmasi Anda pilih dengan klik **[Ya]** atau batal menghapus dengan klik **[Tidak]**.

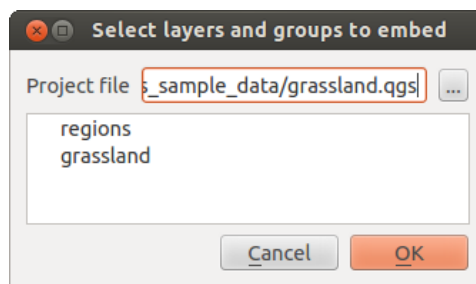
8.9 Proyek-proyek Nesting


Jika Anda ingin menanamkan (embed) isi dari berkas proyek lain kedalam proyek Anda, Anda bisa memilih *Lapisan* → *Tanam Lapisan dan Grup*.

8.9.1 Menanam (Embedding) lapisan

Dialog berikut memungkinkan Anda menanamkan lapisan dari proyek-proyek lain. Berikut ini adalah contoh kecil:


1. Tekan  untuk mencari proyek lain dari dataset Alaska.
2. Pilih berkas proyek `grassland`. Anda bisa melihat konten dari proyek (lihat `figure_embed_dialog`).
3. Tekan `Ctrl` dan klik pada lapisan `grassland` dan `regions`. Tekan **[OK]**. Lapisan akan ditanam dalam legenda peta dan tampilan peta sekarang.



Gambar 8.12: Pilih lapisan-lapisan dan grup-grup untuk ditanam (embed) 

Sementara lapisan tertanam dapat diedit, Anda tidak dapat mengubah properti mereka seperti gaya dan pelabelan.

8.9.2 Menghapus lapisan-lapisan ditanam (embedded)

Klik-kanan pada lapisan yang ditanam (embedded) dan pilih  Hapus.

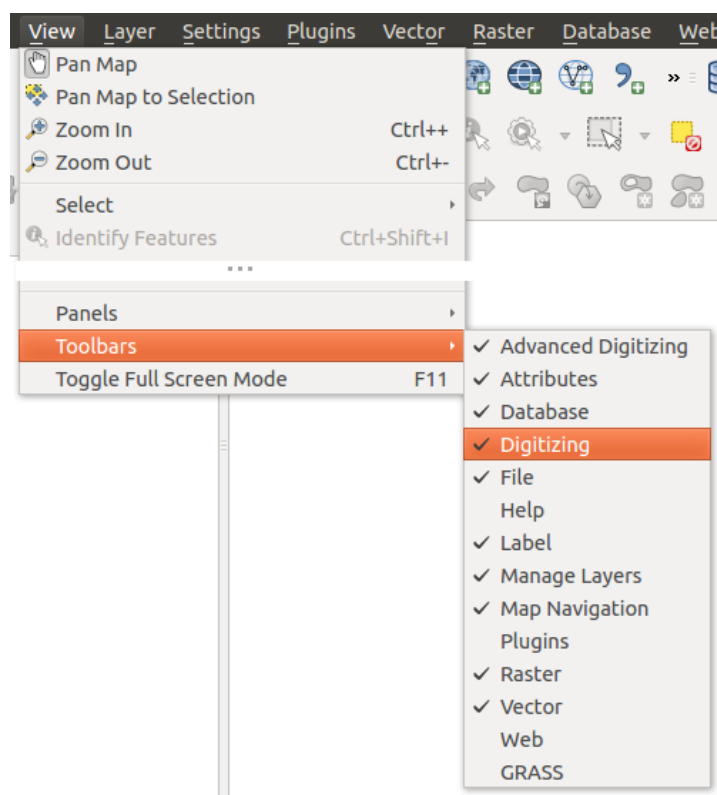
Konfigurasi QGIS

QGIS dapat dikonfigurasi melalui menu *Pengaturan*. Pilih diantara Panel, Toolbar, Properti Proyek, Opsi dan Penyesuaian.

Catatan: QGIS follows desktop guidelines for the location of options and project properties item. Consequently related to the OS you are using, location of some of items described above could be located in the :menuselection'view' menu (Panels and Toolbars) or in *Project* for Options.

9.1 Panel dan Toolbar

Dalam menu *Panel* → Anda bisa mengaktifkan dan menonaktifkan widget QGIS. Menu *Toolbar* → memberikan kemungkinan mengaktifkan dan menonaktifkan ikon grup di toolbar QGIS (lihat *figure_panels_toolbars*).





Gambar 9.1: Menu Panel dan Toolbar 




Tip: Mengaktifkan Tinjauan QGIS

Di QGIS Anda dapat menggunakan panel gambaran yang menyediakan tampilan lapisan tingkat penuh yang ditambahkan ke dalamnya. Hal ini dapat dipilih di bawah menu *Tampilan* → *Panel*. Dalam pandangan persegi panjang yang menunjukkan tingkat peta saat ini. Hal ini memungkinkan Anda dengan cepat menentukan area peta yang sedang Anda lihat. Perhatikan bahwa label tidak diberikan ke gambaran peta bahkan jika lapisan dalam gambaran peta telah diatur untuk pelabelan. Jika Anda klik dan tarik persegi panjang merah dalam gambaran yang menunjukkan gambaran Anda saat ini, tampilan utama peta akan diperbarui.

Tip: Tampilkan Pesan Log


Ini memungkinkan untuk melacak pesan QGIS. Anda bisa mengaktifkan  *Log Pesan* dalam menu  *Pengaturan* → *Panel* dan ikuti pesan yang muncul dalam tab yang berbeda selama memuat dan operasi.

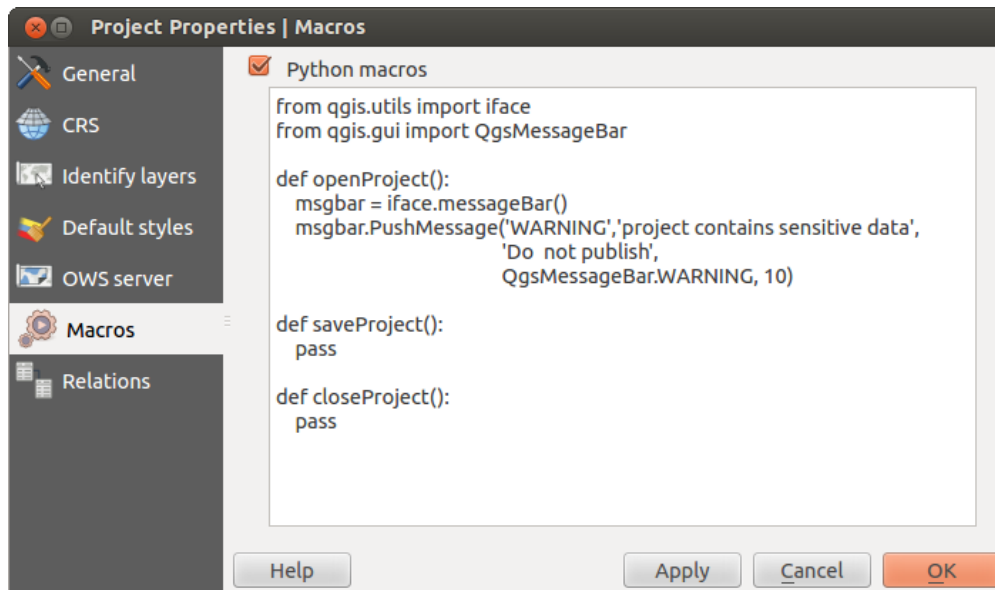
9.2 Properti Proyek

In the properties window for the project under  *Settings* → *Project Properties* (kde) or   *Project* → *Project Properties* (Gnome), you can set project-specific options. These include:

- Di dalam menu *Umum* judul proyek, seleksi dan warna latar, satuan lapisan, presisi, dan pilihan untuk menyimpan path relatif terhadap lapisan yang dapat ditentukan. Jika transformasi CRS aktif Anda dapat memilih ellipsoid untuk perhitungan jarak. Anda dapat menentukan satuan kanvas (hanya digunakan ketika transformasi CRS dinonaktifkan) dan ketepatan desimal untuk digunakan. Anda juga dapat menentukan daftar skala proyek, yang menimpa skala yang telah ditentukan global.
- Menu *CRS* memungkinkan Anda untuk memilih Koordinat Reference System untuk proyek ini, dan memungkinkan proyeksi ulang on-the-fly lapisan raster dan lapisan vektor ketika menampilkan lapisan dari CRS yang berbeda.
- Dengan menu ketiga *Identifikasi lapisan* Anda atur (atau nonaktif) lapisan akan merespon alat mengidentifikasi (lihat paragraf “peralatan Peta” dari bagian *Opsi* untuk mengaktifkan identifikasi atau lapisan multi).
- The *Default Styles* menu lets you control how new layers will be drawn when they do not have an existing `.qml` style defined. You can also set the default transparency level for new layers and whether symbols should have random colours assigned to them. There is also an additional section where you can define specific colors for the running project. You can find the added colors in the drop down menu of the color dialog window present in each renderer.
- Tab *OWS Server* memungkinkan untuk menentukan informasi tentang Server WMS dan kapabilitas WFS QGIS, Pada Tingkat dan Pembatasan CRS.
- Menu *Macros* digunakan untuk mengedit Python macros untuk proyek. Saat ini, hanya tiga macro yang tersedia: `openProject()`, `saveProject()` dan `closeProject()`.
- Menu *Relasi* digunakan untuk menentukan relasi 1:n. Relasi ditentukan dalam dialog properti proyek. Setelah ada relasi lapisan, elemen antarmuka pengguna baru form tampilan (misalnya ketika mengidentifikasi fitur dan membuka form) akan daftar entitas terkait. Ini menyediakan cara ampuh mengekspresikan misalnya sejarah inspeksi pada panjang pipa atau segmen jalan. Anda bisa menemukan lebih jauh tentang relasi 1:n di Bagian *Creating one to many relations*.

9.3 Opsi

 Beberapa pilihan dasar untuk QGIS dapat dipilih menggunakan dialog *Opsi*. Pilih menu opsi *Pengaturan* → *Opsi*. Tab di mana Anda dapat menyesuaikan pilihan Anda dijelaskan di bawah ini.



Gambar 9.2: Pengaturan Macro di QGIS

9.3.1 Menu Umum

Aplikasi

- Pilih *Gaya* (QGIS diperlukan restart) dan pilih diantara ‘Oxygen’, ‘Windows’, ‘Motif’, ‘CDE’, ‘Plastique’ dan ‘Cleanlooks’ .
- Definisikan *Ikon tema* . Sekarang hanya ‘default’.
- Definisikan *Ukuran Ikon* .
- Definisikan *Font*. Pilih antara *QT default* dan font user-defined.
- Ubah *Timeout for timed messages or dialogs* .
- *Sembunyikan layar splash saat startup*
- *Tampilkan Petunjuk saat startup*
- *Judul tebal kotak grup*
- *QGIS-styled kotak grup*
- *Gunakan dialog pemilih warna live-updating*

Berkas proyek

- *Buka proyek pada peluncuran* (memilih antara ‘Baru’, ‘Terbaru’ dan ‘Khusus’). Ketika memilih ‘khusus’ menggunakan untuk menentukan proyek.
- *Buat proyek baru dari proyek default*. Anda memiliki kemungkinan untuk menekan *Atur proyek sekarang sebagai default* atau pada *Reset default*. Anda dapat menelusuri melalui berkas-berkas dan menentukan direktori dimana Anda menemukan pengguna-ditetapkan proyek template Anda. Akan ada sebuah entri di *Project* → *Baru dari Template* jika Anda pertama kali mengaktifkan *Buat proyek baru dari proyek default* dan kemudian simpan proyek dalam folder proyek template.
- *Prompt untuk menyimpan proyek dan sumber data perubahan bila diperlukan*

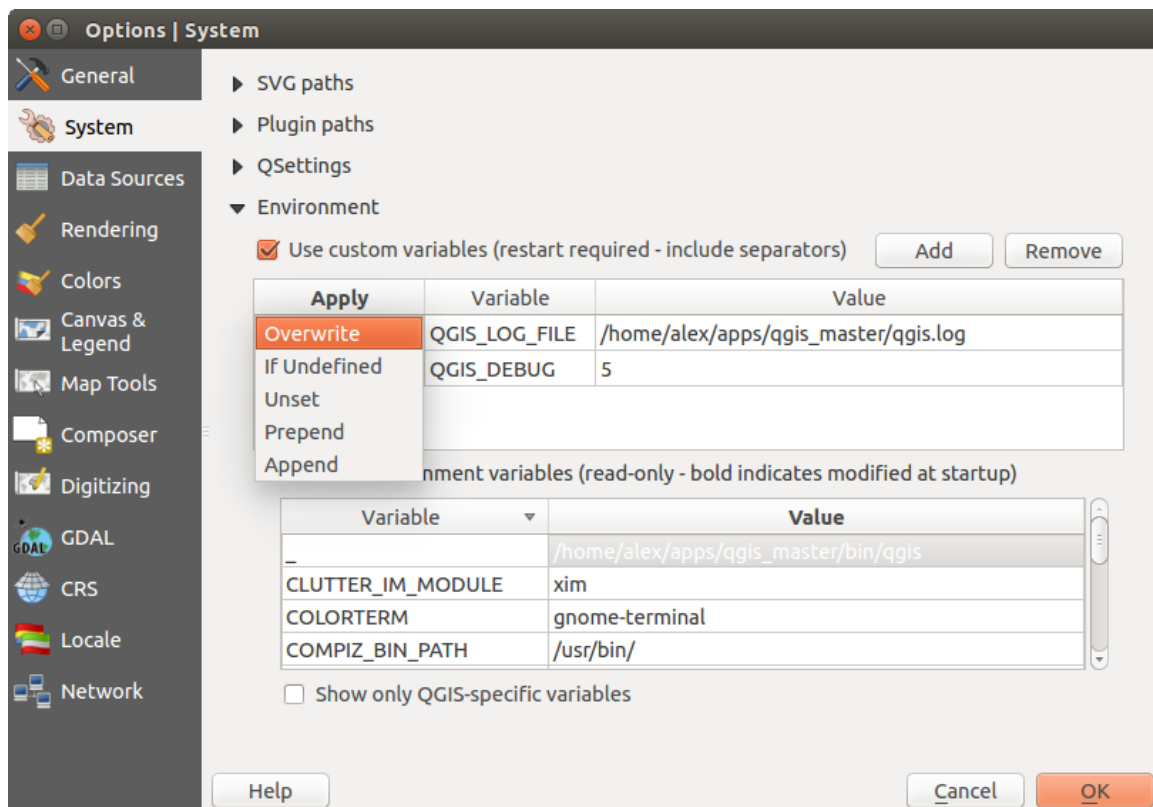
- *Peringatkan ketika membuka berkas proyek QGIS yang disimpan dengan versi lama*
- *Aktifkan macros* . Opsi ini digunakan untuk menangani macro yang ditulis untuk melakukan tindakan pada peristiwa proyek. Anda bisa memilih diantara ‘Tidak pernah’, ‘Bertanya’, ‘Hanya untuk sesi ini’ dan ‘Selalu (tidak direkomendasikan)’.

9.3.2 Menu Sistem

Lingkungan

Sistem lingkungan variabel saat ini dapat dilihat dan banyak dikonfigurasi dalam menu **Lingkungan** (lihat [figure_environment_variables](#)). Hal ini berguna untuk platform, seperti Mac, di mana aplikasi GUI tidak selalu mewarisi lingkungan shell pengguna. Ini juga berguna untuk pengaturan / melihat lingkungan variabel untuk mengatur alat eksternal yang dikendalikan oleh pengolahan toolbox (seperti SAGA, GRASS), dan untuk menyalakan keluaran debugging untuk bagian tertentu dari kode sumber.

- *Gunakan penyesuaian variabel (dibutuhkan restart - termasuk pemisah).* Anda bisa **[Tambah]** dan **[Hapus]** variabel. Variabel lingkungan yang sudah ditetapkan akan ditampilkan dalam: `gui:label:Variabel lingkungan sekarang`, dan hal itu memungkinkan untuk menyaring mereka dengan mengaktifkan *Tampilkan hanya variabel spesifik-QGIS.*








Gambar 9.3: Sistem lingkungan variabel dalam QGIS

Plugin path






[Tambah] atau **[Hapus]** *Path(s) untuk mencari tambahan C++ librari Plugin*

9.3.3 Menu sumber data

Atribut dan tabel fitur







-  *Buka tabel atribut di dock window (dibutuhkan restart QGIS)*
-  *Salin geometri di representasi WKT dari tabel atribut.* Ketika menggunakan  Salin baris yang dipilih ke papan klip dari menu *Tabel atribut* kemudian ini memiliki hasil yang juga koordinat titik atau simpul disalin ke papan klip.
- *Perilaku tabel atribut* . Ada tiga kemungkinan: ‘Tampilkan semua fitur’, ‘Tampilkan fitur dipilih’ dan ‘Tampilkan fitur terlihat pada peta’
- *Tabel atribut baris cache*  1,00. Baris cache ini memungkinkan untuk menyimpan baris x atribut terakhir dimuat sehingga proses tabel atribut akan lebih cepat. Cache akan dihapus ketika menutup tabel atribut.
- *Representasi untuk nilai NULL* Here you can define a value for data fields containing a NULL value. Di sini Anda dapat menentukan nilai untuk bidang data yang berisi nilai NULL.

Menangani sumber data

- *Pindai item yang valid di dock Browser* . Anda dapat memilih antara ‘Periksa ekstensi’ dan ‘Periksa isi berkas’.
- *Pindai isi dari berkas yang dikompresi (.zip) di browser dock* . ‘Tidak’, ‘Pindai dasar’ and ‘Pindai full’ memungkinkan.
- *Konfirmasi sublapisan raster saat membuka.* Beberapa raster mendukung sublapisan – mereka disebut subdataset di GDAL. Contohnya adalah berkas netCDF - jika ada banyak variabel netCDF, GDAL melihat setiap variabel sebagai sub dataset. Pilihan ini untuk mengontrol bagaimana menangani sub-lapisan ketika sebuah berkas dengan sub-lapisan dibuka. Anda memiliki pilihan berikut:
 - ‘Selalu’: Selalu bertanya (jika ada sub-lapisan yang ada)
 - ‘Jika diperlukan’: Menanyakan apakah lapisan tidak memiliki band, namun memiliki sub-lapisan
 - ‘Tidak Pernah’: Tidak pernah meminta, tidak akan memuat apa-apa
 - ‘Muat semua’: Tidak pernah meminta, tetapi memuat semua sub-lapisan
-  *Abaikan deklarasi encoding shapefile.* Jika shapefile punya pengkodean informasi ini akan diabaikan oleh QGIS.
-  *Tambahkan lapisan PostGIS dengan klik ganda dan pilih dalam mode diperpanjang*
-  *Tambahkan lapisan Oracle with dobel klik dan pilih dalam mode diperpanjang*

9.3.4 Menu Rendering

Tindakan rendering

-  *Secara default lapisan baru yang ditambahkan ke peta harus ditampilkan*
-  *Gunakan render caching yang memungkinkan mempercepat gambar ulang*
-  *Render layers in parallel using many CPU cores*
-  *Max cores to use*
- *Map update interval (default to 250 ms)*
-  *Aktifkan fitur simplification secara default untuk lapisan baru yang ditambahkan*
- *Simplification threshold*
-  *Sederhanakan di sisi penyedia jika mungkin*
- *Maximum scale at which the layer should be simplified*





Kualitas rendering

- Membuat garis tampil kurang bagus dengan mengorbankan beberapa kinerja menggambar

Raster-raster

- Dengan *RGB band seleksi* Anda bisa menetapkan nomor dari band Merah, Hijau dan Biru.

Contrast enhancement

- *Band abu-abu tunggal* . Sebuah band abu-abu tunggal dapat memiliki 'Tidak ada peregangan', 'Lakukan peregangan untuk MinMax', 'Peregangan dan Klip ke MinMax' dan juga 'Clip ke MinMax'
- *band multi warna (byte/band)* . Opsinya antara lain 'Tidak ada peregangan', 'Lakukan peregangan untuk MinMax', 'Peregangan dan Klip ke MinMax' dan juga 'Clip ke MinMax'.
- *band multi warna (>byte/band)* . Opsinya antara lain 'Tidak ada peregangan', 'Lakukan peregangan untuk MinMax', 'Peregangan dan Klip ke MinMax' dan juga 'Clip ke MinMax'
- *Batas (minimum/maksimum)* . Opsinya adalah 'Kumulatif jumlah potong pixel', 'Minimum/Maksimum', 'Berarti +/- standar deviasi'.
- *Kumulatif batas jumlah potong pixel*
- *Standar deviasi multiplier*

Debugging

- Refresh kanvas peta

9.3.5 Colors Menu


This menu allows you to add some custom color that you can find in each color dialog window of the renderes. You will see a set of predefined colors in the tab: you can delete or edit all of them. Moreover you can add the color you want and perform some copy and paste operation. Finally you can export the color set as a `gpl` file or import them.

9.3.6 Menu Kanvas dan Legenda

Standar penampilan peta (diganti oleh proyek properti)

- Tetapkan *Seleksi warna* dan *Warna latar*.

Legenda lapisan

- *Dobel klik pada legenda* . Anda dapat 'Buka properti lapisan' atau 'Buka atribut tabel' dengan double klik.
- Mengikuti *Gaya item legenda*:
 - *Nama lapisan kapital*
 - *Nama lapisan tebal*
 - *Nama grup tebal*
 - *Tampilkan klasifikasi nama atribut*
 - *Buat ikon raster (mungkin lambat)*
 - *Tambah lapisan baru ke grup sekarang atau yang dipilih*

9.3.7 Menu perangkat Peta

This menu offers some options regarding the behaviour of the *Identify tool*.

- *Search radius for identifying and displaying map tips* is a tolerance factor expressed as a percentage of the map width. This means the identify tool will depict results as long as you click within this tolerance.
- *Highlight color* allows you to choose with which color should features being identified are to be highlighted.
- *Buffer* expressed as a percentage of the map width, determines a buffer distance to be rendered from the outline of the identify highlight.
- *Minimum width* expressed as a percentage of the map width, determines how thick should the outline of a highlighted object be.

Alat pengukuran

- Tentukan *Warna Rubberband* untuk peralatan pengukuran
- Tentukan *Tempat desimal*
- *Perlu satuan dasar*
- *Satuan pengukuran yang dipilih* ('Meter', 'Kaki', 'Mil laut' atau 'derajat')
- *Satuan sudut yang dipilih* ('Derajat', 'Radian' atau 'Gon')

Menggeser dan memperbesar

- Tentukan *Aksi roda tetikus* ('Perbesar', 'Perbesar dan pusat', 'Perbesar pada kursor tetikus', 'Tidak ada')
- Tentukan *Faktor pembesaran* untuk roda tetikus

Skala yang ditentukan

Di sini Anda menemukan daftar skala yang telah ditentukan. Dengan tombol [+] dan [-] Anda dapat menambahkan atau menghapus skala individu Anda.

9.3.8 Menu Penyusun

Standar Komposisi

Anda bisa menentukan font *Default* disini.

Penampilan Kotak

- Tentukan the *Gaya Kotak* ('Solid', 'Dots', 'Crosses')
- Tentukan *Warna...*

Standar kotak

- Tentukan *Spacing* 1,00
- Tentukan *Ofset kotak* 1,00 untuk x dan y
- Tentukan *Toleransi Snap* 1,00

Standar panduan

- Tentukan *Toleransi Snap* 1,00

9.3.9 Menu Digitalisasi


Membuat fitur

- Menekan atribut jendela pop-up setelah setiap fitur dibuat
- Gunakan kembali nilai atribut terakhir yang dimasukkan
- *Validasi geometris.* Mengedit garis/poligon kompleks dengan banyak node menyebabkan render sangat lambat. Hal ini karena prosedur standar validasi di QGIS dapat menghabiskan banyak waktu. Untuk mempercepat rendering itu dengan memilih validasi geometri GEOS (mulai dari GEOS 3.3) atau untuk memmatikannya. Validasi geometri GEOS jauh lebih cepat, tetapi masalahnya adalah bahwa hanya geometri pertama yang akan dilaporkan.


Rubberband

- Tentukan Rubberband *Lebar garis* dan *Warna garis*


Snapping

- Buka opsi mengambil di dock window (dibutuhkan restart QGIS)
- Tentukan *Mode standar snap*  ('Simpul', 'Segmen', 'Simpul dan Segmen', 'Mati')
- Tentukan *Default toleransi snapping* dalam satuan peta atau pixel
- Tentukan *Cari radius untuk suntingan simpul* dalam satuan peta atau pixel

Penanda simpul

- Tampilkan penanda hanya utk fitur yang dipilih
- Tentukan simpul *Gaya Penanda*  ('Palang' (standar), 'Lingkaran semi transparan' atau 'Tidak ada')
- Tentukan simpul *Ukuran Penanda*

Alat ofset kurva

3 pilihan berikutnya mengacu pada alat  Kurva Ofset dalam *Advanced digitizing*. Melalui berbagai pengaturan, ini memungkinkan untuk mempengaruhi bentuk garis ofset. Opsi-opsi ini mungkin dimulai dari GEOS 3.3.

- *Gaya Join*
- *Segmen Quadrant*
- *Batas Miter*

9.3.10 Menu GDAL

GDAL adalah data pertukaran librari untuk berkas raster. Dalam tab ini Anda dapat *Edit membuat opsi* dan *Edit Opsi Pyramid* dari format raster. Menentukan driver GDAL yang akan digunakan untuk format raster seperti dalam beberapa kasus lebih dari satu driver GDAL tersedia.

9.3.11 Menu CRS

CRS standar untuk proyek baru

- Jangan diaktifkan proyeksi ulang 'on the fly'
- Aktif otomatis proyeksi ulang 'on the fly' jika lapisan memiliki CRS berbeda
- Mengaktifkan proyeksi ulang 'on the fly' secara default
- Pilih sebuah CRS dan *Selalui mulai proyek baru dengan CRS ini*

CRS untuk lapisan baru

Daerah ini memungkinkan untuk menentukan tindakan ketika sebuah lapisan baru dibuat, atau ketika lapisan tanpa CRS dimuat.

- *Konfirmasi CRS*
- *Gunakan CRS proyek*
- *Gunakan standar CRS ditampilkan dibawah ini*

Transformasi datum standar

- *Mintalah transformasi datum bila tidak ada standar yang ditentukan*
- Jika Anda telah bekerja dengan transformasi CRS ‘on-the-fly’ Anda bisa melihat hasil dari transformasi dalam jendela di bawah. Anda bisa menemukan informasi tentang ‘Sumber CRS’ dan ‘Destinasi CRS’ maupun ‘Sumber datum transform’ dan ‘Destinasi datum transform’.

9.3.12 Menu Lokal

- *Timpa sistem lokal dan Lokal gunakan sebagai pengganti*
- Informasi tentang sistem lokal aktif


9.3.13 Menu Jaringan

Umum

- Tentukan *WMS cari alamat*, default `http://geopole.org/wms/search?search=%1&type=rss`
- Tentukan *Timeout untuk permintaan jaringan (ms)* - default 60000
- Tentukan *Standar periode ekspirasi untuk WMSC/WMTS (jam)* - default 24
- Tentukan *Mencoba kembali Maksimum jika terjadi kesalahan permintaan genteng*
- Tentukan *User-Agent*

Pengaturan cache

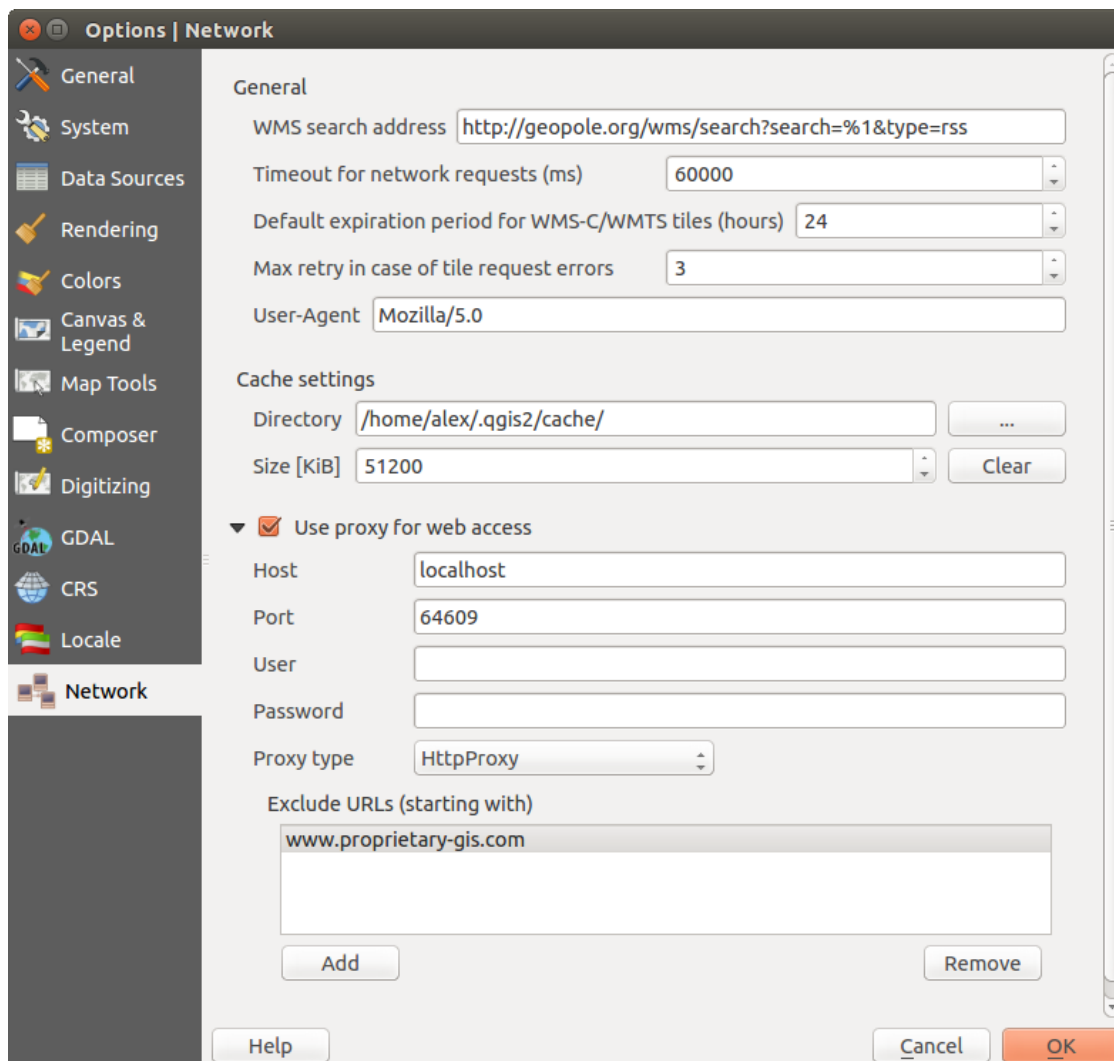
Tentukan *Direktori* dan *Ukuran* untuk cache.

- *Gunakan proxy untuk akses web* dan tentukan ‘Host’, ‘Port’, ‘Pengguna’, and ‘Kata Sandi’.
- Atur *Tipe Proxy*  sesuai dengan kebutuhan Anda.
 - *Proxy Standar*: Proxy ditentukan berdasarkan aplikasi pengaturan menggunakan proxy
 - *Socks5Proxy*: Proxy generik untuk setiap jenis koneksi. Mendukung TCP, UDP, mengikat ke port (koneksi masuk) dan otentikasi.
 - *HttpProxy*: Menggunakan perintah “CONNECT”, mendukung hanya koneksi TCP; mendukung otentikasi.
 - *HttpCachingProxy*: Menggunakan perintah normal HTTP, itu hanya berguna dalam konteks permintaan HTTP.
 - *FtpCachingProxy*: Menggunakan proxy FTP, itu hanya berguna dalam konteks permintaan FTP.

Tidak termasuk beberapa URL dapat ditambahkan ke kotak teks di bawah pengaturan-proxy (lihat [Figure_Network_Tab](#)).

Jika Anda membutuhkan informasi lebih rinci tentang pengaturan-proxy yang berbeda, silakan lihat panduan QT-librari-dokumentasi di <http://doc.trolltech.com/4.5/qnetworkproxy.html#ProxyType-enum>.



Tip: Menggunakan Proxi



Gambar 9.4: Pengaturan-proxy di QGIS

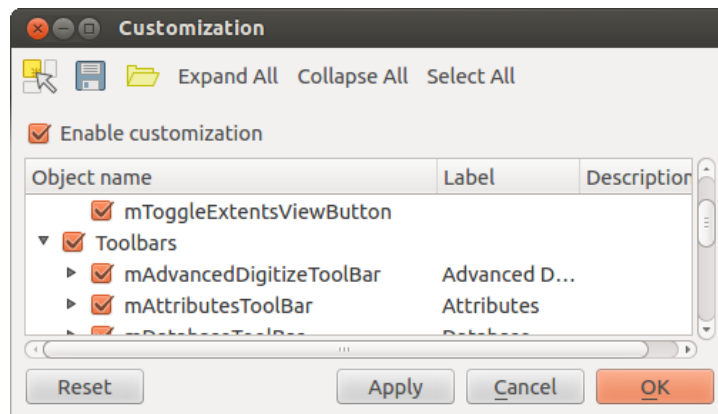
Menggunakan proxy kadang-kadang bisa rumit. Hal ini berguna ‘coba dan eror’ jenis proxy di atas, untuk memeriksa apakah mereka berhasil dalam kasus Anda.

Anda dapat mengubah pilihan sesuai dengan kebutuhan Anda. Beberapa perubahan mungkin memerlukan restart QGIS sebelum berjalan efektif.

-  Settings are saved in a text file: `$HOME/.config/QGIS/QGIS2.conf`
- **X** Anda dapat menemukan pengaturan Anda di: `$HOME/Library/Preferences/org.qgis.qgis.plist`
-  pengaturan terkirim ke registri: `HKEY\CURRENT_USER\Software\QGIS\qgis`


9.4 Penyesuaian (Customization)

Alat penyesuaian memungkinkan Anda mengaktifkan (dan nonaktif) hampir setiap elemen dalam antar muka QGIS. Hal ini bisa sangat berguna jika Anda memiliki banyak plugin yang dipasang bahwa Anda tidak pernah menggunakan dan mengisi layar Anda.



Gambar 9.5: Dialog penyesuaian 

Penyesuaian QGIS dibagi menjadi lima kelompok. Dalam *Menu* Anda dapat menyembunyikan entri dalam Menu bar. Dalam *Panel* Anda dapat menemukan Panel jendela. Jendela Panel adalah aplikasi yang dapat dimulai dan digunakan sebagai mengambang, jendela tingkat-atas atau tertanam ke jendela utama QGIS sebagai widget. (lihat juga *Panel dan Toolbar*). Dalam fitur *Status Bar* seperti informasi koordinat dapat dinonaktifkan. Dalam *Toolbars* Anda dapat mengaktifkan (non aktif) ikon toolbar QGIS dan *Widgets* Anda dapat mengaktifkan (non aktif) dialog serta tombol mereka.

Dengan  Beralih ke penangkapan widget dalam aplikasi utama Anda dapat klik elemen dalam QGIS yang Anda ingin menyembunyikan dan menemukan entri yang sesuai dalam Penyesuaian (lihat [figure_customization](#)). Anda juga dapat menyimpan berbagai setup yang berbeda untuk kasus penggunaan yang berbeda juga. Sebelum perubahan diterapkan, Anda harus me-restart QGIS.

Working with Projections


QGIS allows users to define a global and project-wide CRS (coordinate reference system) for layers without a pre-defined CRS. It also allows the user to define custom coordinate reference systems and supports on-the-fly (OTF) projection of vector and raster layers. All of these features allow the user to display layers with different CRSs and have them overlay properly.

10.1 Overview of Projection Support

QGIS has support for approximately 2,700 known CRSs. Definitions for each CRS are stored in a SQLite database that is installed with QGIS. Normally, you do not need to manipulate the database directly. In fact, doing so may cause projection support to fail. Custom CRSs are stored in a user database. See section *Custom Coordinate Reference System* for information on managing your custom coordinate reference systems.


The CRSs available in QGIS are based on those defined by the European Petroleum Search Group (EPSG) and the Institut Geographique National de France (IGNF) and are largely abstracted from the spatial reference tables used in GDAL. EPSG identifiers are present in the database and can be used to specify a CRS in QGIS.

In order to use OTF projection, either your data must contain information about its coordinate reference system or you will need to define a global, layer or project-wide CRS. For PostGIS layers, QGIS uses the spatial reference identifier that was specified when the layer was created. For data supported by OGR, QGIS relies on the presence of a recognized means of specifying the CRS. In the case of shapefiles, this means a file containing the well-known text (WKT) specification of the CRS. This projection file has the same base name as the shapefile and a `.prj` extension. For example, a shapefile named `alaska.shp` would have a corresponding projection file named `alaska.prj`.

Whenever you select a new CRS, the layer units will automatically be changed in the *General* tab of the  *Project Properties* dialog under the *Project* (Gnome, OS X) or *Settings* (KDE, Windows) menu.

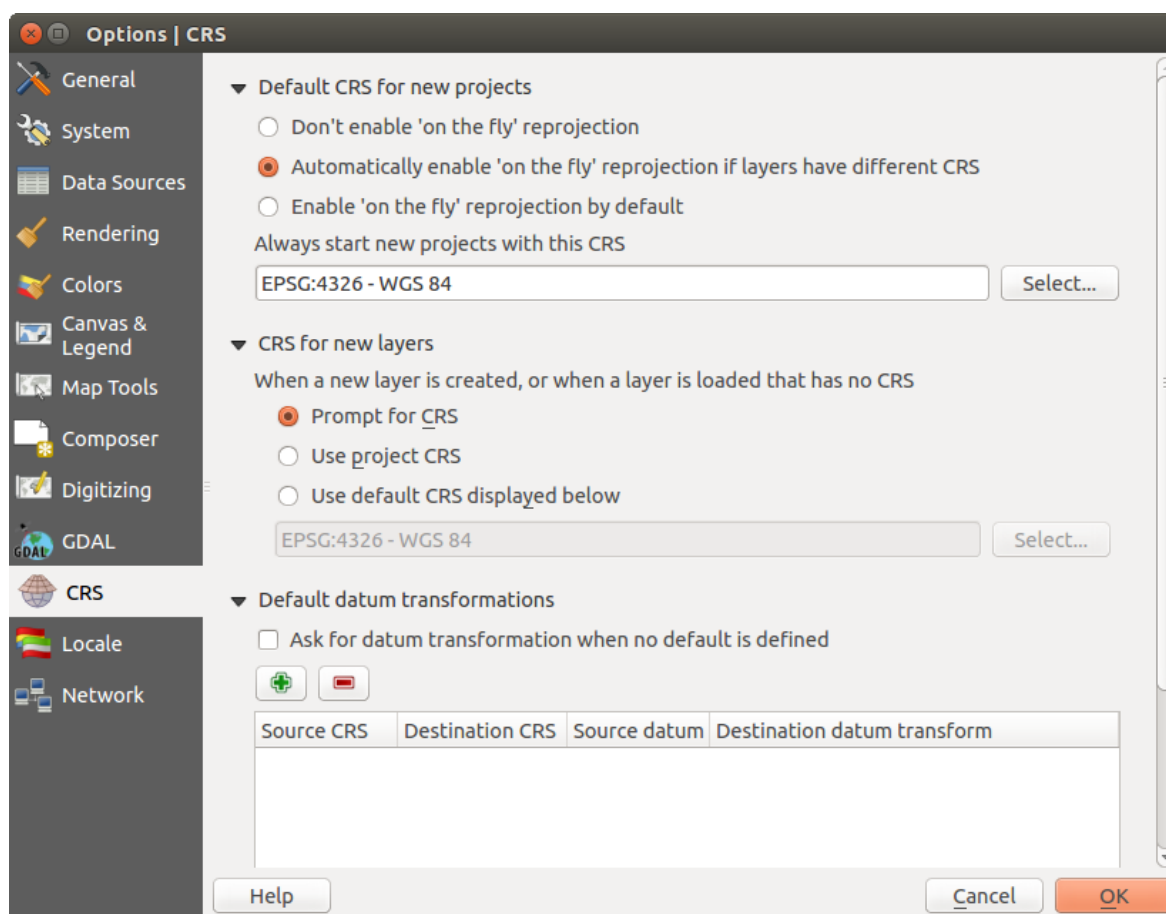
10.2 Global Projection Specification

QGIS starts each new project using the global default projection. The global default CRS is EPSG:4326 - WGS 84 (`proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs`), and it comes predefined in QGIS. This default can be changed via the **[Select...]** button in the first section, which is used to define the default coordinate reference system for new projects, as shown in [figure_projection_1](#). This choice will be saved for use in subsequent QGIS sessions.

When you use layers that do not have a CRS, you need to define how QGIS responds to these layers. This can be done globally or project-wide in the *CRS* tab under *Settings* →  *Options*.

The options shown in [figure_projection_1](#) are:

- *Prompt for CRS*
- *Use project CRS*



Gambar 10.1: CRS tab in the QGIS Options Dialog 🐧


-  Use default CRS displayed below

If you want to define the coordinate reference system for a certain layer without CRS information, you can also do that in the *General* tab of the raster and vector properties dialog (see *General Menu* for rasters and *General Menu* for vectors). If your layer already has a CRS defined, it will be displayed as shown in *Vector Layer Properties Dialog*.



Tip: CRS in the Map Legend


Right-clicking on a layer in the Map Legend (section *Legenda Peta*) provides two CRS shortcuts. *Set layer CRS* takes you directly to the Coordinate Reference System Selector dialog (see *figure_projection_2*). *Set project CRS from Layer* redefines the project CRS using the layer's CRS.

10.3 Define On The Fly (OTF) Reprojection

QGIS supports OTF reprojection for both raster and vector data. However, OTF is not activated by default. To use OTF projection, you must activate the *Enable on the fly CRS transformation* checkbox in the *CRS* tab of the  *Project Properties* dialog.

There are three ways to do this:

1. Select  *Project Properties* from the *Project* (Gnome, OSX) or *Settings* (KDE, Windows) menu.
2. Click on the  CRS status icon in the lower right-hand corner of the status bar.
3. Turn OTF on by default in the *CRS* tab of the *Options* dialog by selecting *Enable 'on the fly' reprojection by default* or *Automatically enable 'on the fly' reprojection if layers have different CRS*.

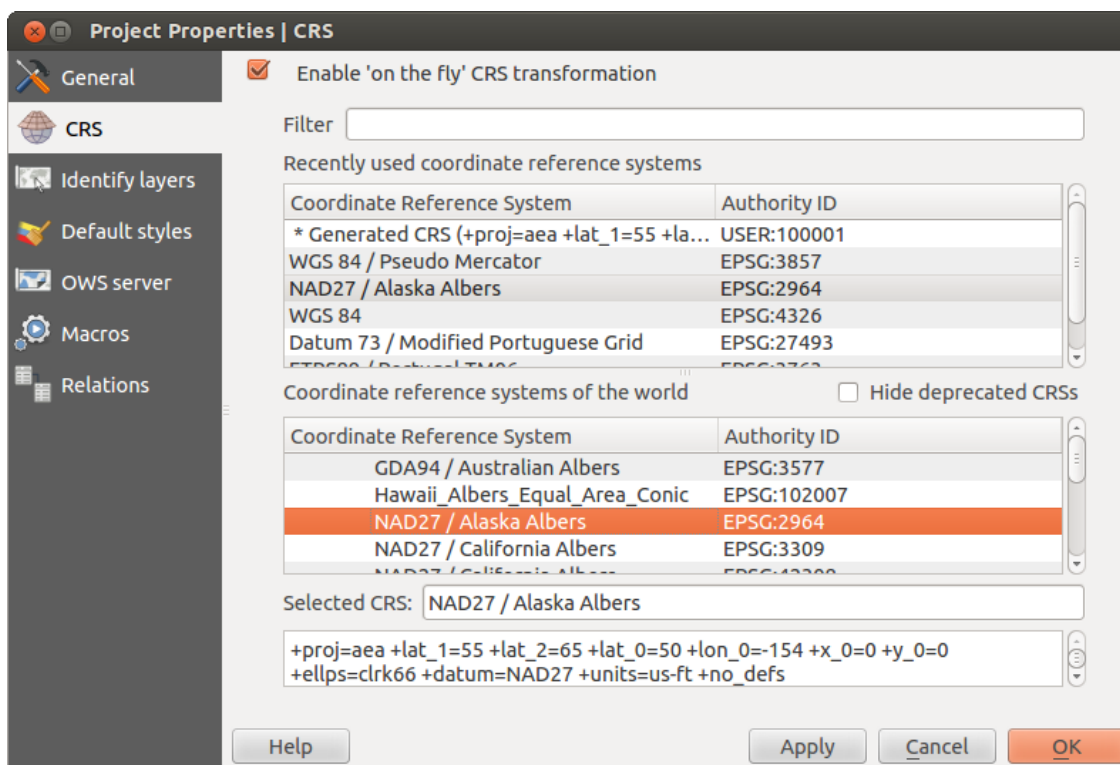
If you have already loaded a layer and you want to enable OTF projection, the best practice is to open the *CRS* tab of the *Project Properties* dialog, select a CRS, and activate the *Enable 'on the fly' CRS transformation* checkbox. The  CRS status icon will no longer be greyed out, and all layers will be OTF projected to the CRS shown next to the icon.

The *CRS* tab of the *Project Properties* dialog contains five important components, as shown in *Figure_projection_2* and described below:


1. **Enable 'on the fly' CRS transformation** — This checkbox is used to enable or disable OTF projection. When off, each layer is drawn using the coordinates as read from the data source, and the components described below are inactive. When on, the coordinates in each layer are projected to the coordinate reference system defined for the map canvas.
2. **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.
3. **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.
4. **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.
5. **PROJ.4 text** — This is the CRS string used by the PROJ.4 projection engine. This text is read-only and provided for informational purposes.

Tip: Project Properties Dialog


If you open the *Project Properties* dialog from the *Project* menu, you must click on the *CRS* tab to view the CRS settings.



Gambar 10.2: Project Properties Dialog 

Opening the dialog from the  `CRS status` icon will automatically bring the `CRS` tab to the front.

10.4 Custom Coordinate Reference System


If QGIS does not provide the coordinate reference system you need, you can define a custom CRS. To define a CRS, select  *Custom CRS...* from the *Settings* menu. Custom CRSs are stored in your QGIS user database. In addition to your custom CRSs, this database also contains your spatial bookmarks and other custom data.

Defining a custom CRS in QGIS requires a good understanding of the PROJ.4 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 <ftp://ftp.remotesensing.org/proj/OF90-284.pdf>).

This manual describes the use of the `proj.4` and related command line utilities. The cartographic parameters used with `proj.4` are described in the user manual and are the same as those used by QGIS.

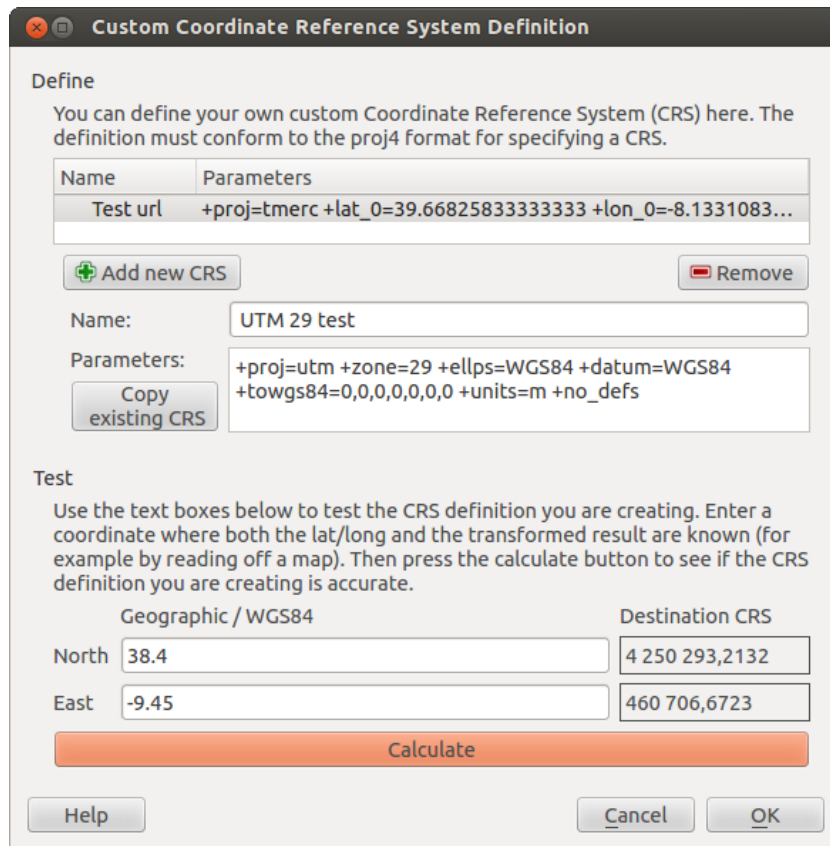
The *Custom Coordinate Reference System Definition* dialog requires only two parameters to define a user CRS:

1. A descriptive name
2. The cartographic parameters in PROJ.4 format

To create a new CRS, click the  `Add new CRS` button and enter a descriptive name and the CRS parameters.

Note that the *Parameters* must begin with a `+proj=` block, to represent the new coordinate reference system.


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.



Gambar 10.3: Custom CRS Dialog 

10.5 Default datum transformations

OTF depends on being able to transform data into a ‘default CRS’, and QGIS uses WGS84. For some CRS there are a number of transforms available. QGIS allows you to define the transformation used otherwise QGIS uses a default transformation.

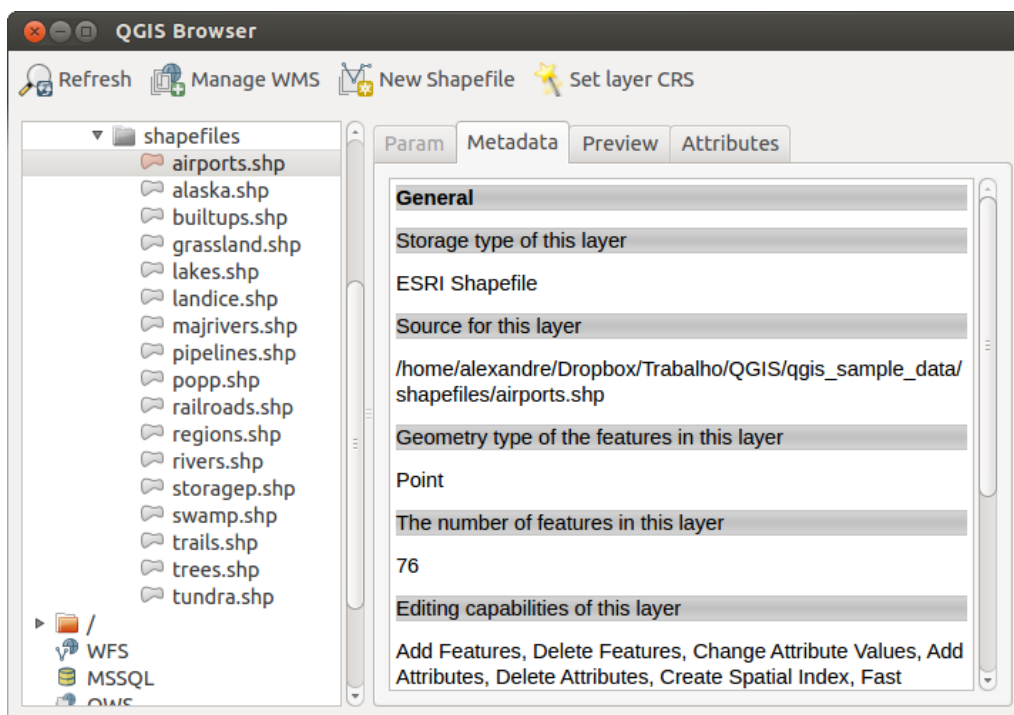
In the *CRS* tab under *Settings* →  *Options* you can:

- set QGIS to ask you when it needs define a transformation using *Ask for datum transformation when no default is defined*
- edit a list of user defaults for transformations.

QGIS asks which transformation to use by opening a dialogue box displaying PROJ.4 text describing the source and destination transforms. Further information may be found by hovering over a transform. User defaults can be saved by selecting *Remember selection*.


QGIS Peramban


Penjelajah QGIS adalah panel di QGIS yang memungkinkan Anda dengan mudah menavigasi dalam basisdata Anda. Anda memiliki akses ke berkas-berkas vektor umum (seperti berkas ESRI shapefile atau MapInfo), basisdata (seperti PostGIS, Oracle, Spatialite atau MSSQL Spatial) dan koneksi WMS/WFS. Anda juga bisa melihat data GRASS Anda (untuk mendapatkan data ke QGIS, lihat *GRASS GIS Integration*)





Gambar 11.1: Penjelajah QGIS merupakan aplikasi mandiri 🐧

Menggunakan penjelajah QGIS untuk menampilkan data Anda. Fungsi geser dan taruh membuatnya mudah untuk menempatkan data Anda ke tampilan peta dan legenda peta.


1. Mengaktifkan penjelajah QGIS. Klik-kanan pada toolbar dan centang  *Peramban* atau pilih dari *Pengaturan* → *Panel*.
2. Geser panel kedalam jendela legenda dan riliskan.
3. Klik pada tab *Peramban*
4. Jelajahi dalam basisdata Anda dan pilih folder shapefile dari direktori `qgis_sample_data`.
5. Tekan tombol Shift dan pilih berkas `airports.shp` dan `alaska.shp`.
6. Tekan tombol kiri tetikus kemudian geser dan tempatkan berkas ke dalam kanvas peta.

7. Klik-kanan pada lapisan dan pilih *Atur CRS proyek dari lapisan*. Untuk informasi lebih lanjut lihat *Working with Projections*.
8. Klik pada  Perbesar Full agar lapisan-lapisan terlihat.

Ada penjelajah kedua yang tersedia di *Pengaturan* → *Panel*. Hal ini berguna bila Anda perlu untuk memindahkan berkas atau lapisan antar lokasi.




1. Mengaktifkan penjelajah QGIS kedua: Klik-kanan pada toolbar dan centang  *Penjelajah (2)*, atau pilih dari *Pengaturan* → *Panel*.
2. Geser panel kedalam jendela legenda
3. Arahkan ke tab *Peramban (2)* dan temukan shapefile di dalam sistem berkas Anda.
4. Pilih sebuah berkas dengan tombol kiri tetikus. Sekarang Anda bisa menggunakan ikon  *Tambah Lapisan Terpilih* untuk menambahkannya kedalam proyek sekarang.

QGIS otomatis terlihat untuk Coordinate Reference System (CRS) dan perbesar ke batas lapisan jika Anda bekerja di sebuah proyek kosong QGIS. Jika sudah ada berkas dalam proyek Anda, berkas hanya akan ditambahkan dan dalam kasus itu memiliki tingkat yang sama dan CRS akan divisualisasikan. Jika berkas ini telah mendapat CRS lain dan lapisan Anda terlebih dahulu klik kanan pada lapisan dan pilih *Atur CRS Proyek dari Lapisan*. Kemudian pilih *Perbesar ke Batas Lapisan*.

Fungsi  *Saring Berkas* bekerja pada level direktori. Jelajahi folder dimana Anda ingin menyaring berkas-berkas dan memberikan kata pencarian atau wildcard. Penjelajah hanya menampilkan nama berkas yang sesuai – data lain tidak akan ditampilkan.

Ini juga mungkin untuk menjalankan penjelajah QGIS sebagai aplikasi mandiri.

Mulai Penjelajah QGIS

-  ketik di dalam “qbrowser” di perintah prompt.
-  Mulai penjelajah QGIS menggunakan menu Start atau desktop shortcut.
- Peramban  QGIS juga tersedia di folder Aplikasi Anda.

Dalam *figure_browser_standalone_metadata*, Anda bisa melihat ditingkatkannya fungsi penjelajah QGIS mandiri. Tab *Param* menyediakan detail dari koneksi dataset seperti PostGIS atau MSSQL Spatial. Tab *Metadata* berisi informasi umum tentang berkas (lihat *Metadata Menu*). Dengan tab *Preview* Anda bisa melihat-lihat di berkas Anda tanpa mengimpor mereka ke proyek QGIS Anda. Ini juga mungkin untuk melihat atribut berkas Anda dalam tab *Atribut*

Pekerjaan dengan Data Vektor

12.1 Supported Data Formats

QGIS uses the OGR library to read and write vector data formats, including ESRI shapefiles, MapInfo and MicroStation file formats, AutoCAD DXF, PostGIS, SpatiaLite, Oracle Spatial and MSSQL Spatial databases, and many more. GRASS vector and PostgreSQL support is supplied by native QGIS data provider plugins. Vector data can also be loaded in read mode from zip and gzip archives into QGIS. As of the date of this document, 69 vector formats are supported by the OGR library (see OGR-SOFTWARE-SUITE in *Literatur dan Referensi Web*). The complete list is available at http://www.gdal.org/ogr/ogr_formats.html.

Catatan: Not all of the listed formats may work in QGIS for various reasons. For example, some require external commercial libraries, or the GDAL/OGR installation of your OS may not have been built to support the format you want to use. Only those formats that have been well tested will appear in the list of file types when loading a vector into QGIS. Other untested formats can be loaded by selecting *.*.

Working with GRASS vector data is described in Section *GRASS GIS Integration*.

This section describes how to work with several common formats: ESRI shapefiles, PostGIS layers, SpatiaLite layers, OpenStreetMap vectors, and Comma Separated data (CSV). Many of the features available in QGIS work the same, regardless of the vector data source. This is by design, and it includes the identify, select, labeling and attributes functions.

12.1.1 ESRI Shapefiles


The standard vector file format used in QGIS is the ESRI shapefile. Support is provided by the OGR Simple Feature Library (<http://www.gdal.org/ogr/>).

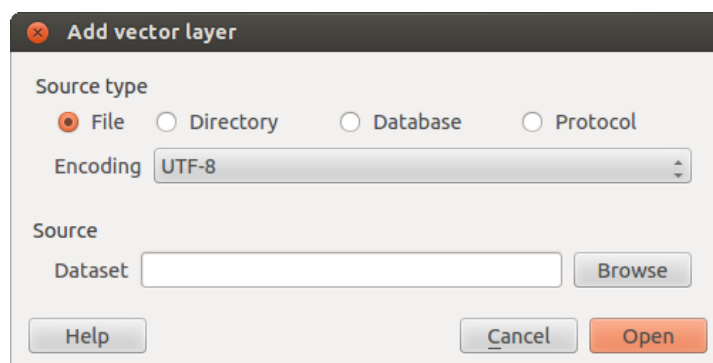
A shapefile actually consists of several files. The following three are required:

1. `.shp` file containing the feature geometries
2. `.dbf` file containing the attributes in dBase format
3. `.shx` index file


Shapefiles also can include a file with a `.prj` suffix, which contains the projection information. While it is very useful to have a projection file, it is not mandatory. A shapefile dataset can contain additional files. For further details, see the ESRI technical specification at <http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>.

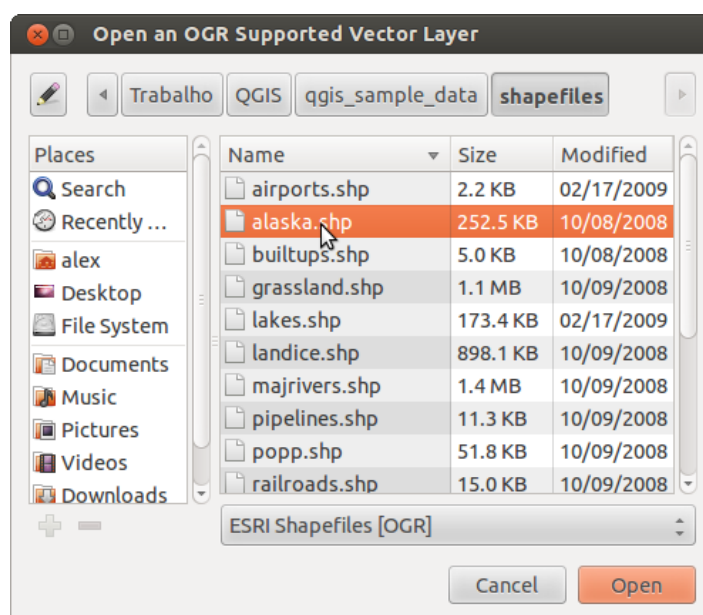
Loading a Shapefile

To load a shapefile, start QGIS and click on the  Add Vector Layer toolbar button, or simply press `Ctrl+Shift+V`. This will bring up a new window (see [figure_vector_1](#)).



Gambar 12.1: Add Vector Layer Dialog 

From the available options check **File**. Click on **[Browse]**. That will bring up a standard open file dialog (see [figure_vector_2](#)), which allows you to navigate the file system and load a shapefile or other supported data source. The selection box *Filter*  allows you to preselect some OGR-supported file formats. You can also select the encoding for the shapefile if desired.



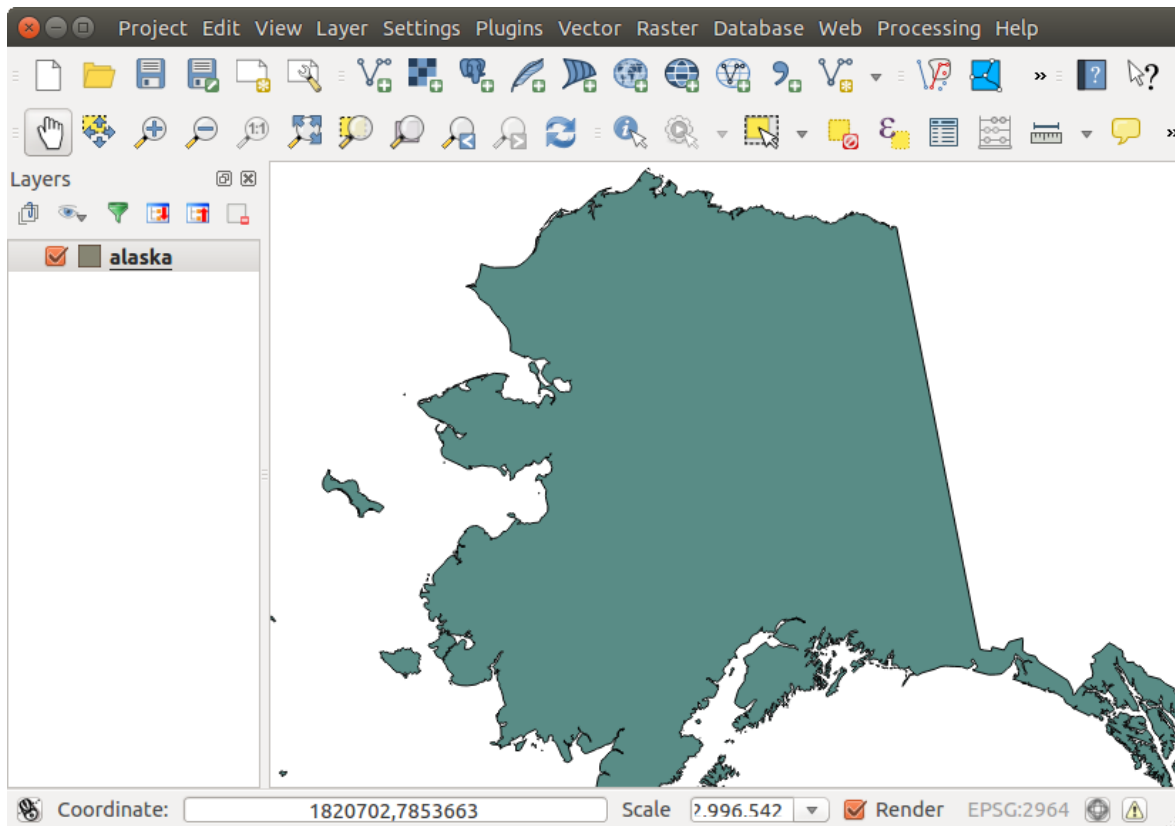
Gambar 12.2: Open an OGR Supported Vector Layer Dialog 

Selecting a shapefile from the list and clicking **[Open]** loads it into QGIS. [Figure_vector_3](#) shows QGIS after loading the `alaska.shp` file.

Tip: Layer Colors

When you add a layer to the map, it is assigned a random color. When adding more than one layer at a time, different colors are assigned to each layer.

Once a shapefile 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



Gambar 12.3: QGIS with Shapefile of Alaska loaded 🐧

legend and choosing *Properties* from the context menu. See section *Style Menu* for more information on setting symbology of vector layers.


Tip: Load layer and project from mounted external drives on OS X

On OS X, portable drives that are mounted beside the primary hard drive do not show up as expected under *File* → *Open Project*. We are working on a more OSX-native open/save dialog to fix this. As a workaround, you can type `/Volumes` in the *File name* box and press `Enter`. Then you can navigate to external drives and network mounts.

Improving Performance for Shapefiles

To improve the performance of drawing a shapefile, 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:

- Load a shapefile by clicking on the  `Add Vector Layer` toolbar button or pressing `Ctrl+Shift+V`.
- Open the *Layer Properties* dialog by double-clicking on the shapefile name in the legend or by right-clicking and choosing *Properties* from the context menu.
- In the *General* tab, click the **[Create Spatial Index]** button.




Problem loading a shape .prj file

If you load a shapefile 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 within the *General* tab of the *Layer Properties* dialog





of the layer by clicking the **[Specify...]** 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 shapefile with QGIS, two different projection files are created: a `.prj` file with limited projection parameters, compatible with ESRI software, and a `.qpj` file, providing the complete parameters of the used CRS. Whenever QGIS finds a `.qpj` file, it will be used instead of the `.prj`.

12.1.2 Loading a MapInfo Layer

 To load a MapInfo layer, click on the  **Add Vector Layer** toolbar button; or type `Ctrl+Shift+V`, change the file type filter *Files of type*  to 'Mapinfo File [OGR] (*.mif *.tab *.MIF *.TAB)' and select the MapInfo layer you want to load.

12.1.3 Loading an ArcInfo Binary Coverage

 To load an ArcInfo Binary Coverage, click on the  **Add Vector Layer** toolbar button or press `Ctrl+Shift+V` to open the *Add Vector Layer* dialog. Select  *Directory* as *Source type*. Change the file type filter *Files of type*  to 'Arc/Info Binary Coverage'. Navigate to the directory that contains the coverage file, and select it.

Similarly, you can load directory-based vector files in the UK National Transfer Format, as well as the raw TIGER Format of the US Census Bureau.

12.1.4 Delimited Text Files

Tabular data is a very common and widely used format because of its simplicity and readability – data can be viewed and edited even in a plain text editor. A delimited text file is an attribute table with each column separated by a defined character and each row separated by a line break. The first row usually contains the column names. A common type of delimited text file is a CSV (Comma Separated Values), with each column separated by a comma.

Such data files can also contain positional information in two main forms:

- As point coordinates in separate columns
- As well-known text (WKT) representation of geometry

QGIS allows you to load a delimited text file as a layer or ordinal table. But 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 in the text file.
2. The header row must contain field(s) with geometry definition. These field(s) can have any name.
3. The X and Y coordinates (if geometry is defined by coordinates) must be specified as numbers. The coordinate system is not important.

As an example of a valid text file, we import the elevation point data file `elevp.csv` that comes with the QGIS sample dataset (see section *Contoh data*):

```
X;Y;ELEV
-300120;7689960;13
-654360;7562040;52
1640;7512840;3
[...]
```

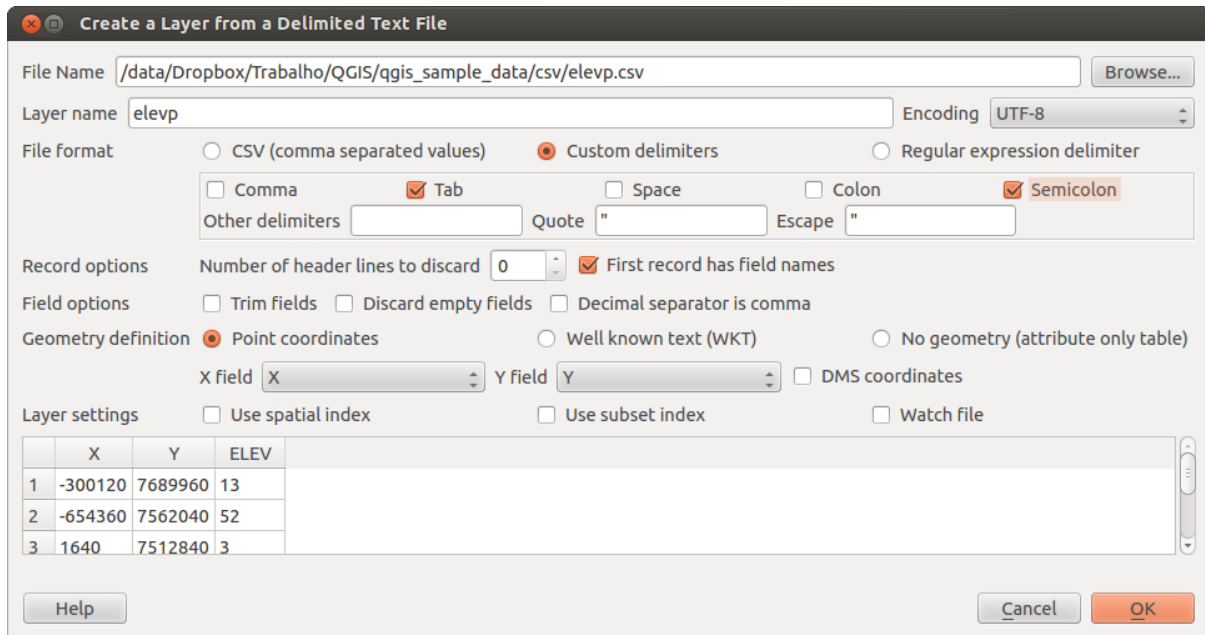
Some items to note about the text file:


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.

Loading a delimited text file

Click the toolbar icon  Add Delimited Text Layer in the *Manage layers* toolbar to open the *Create a Layer from a Delimited Text File* dialog, as shown in [figure_delimited_text_1](#).



Gambar 12.4: Delimited Text Dialog 

First, select the file to import (e.g., `qgis_sample_data/csv/elevp.csv`) by clicking on the **[Browse]** button. Once the file is selected, QGIS attempts to parse the file with the most recently used delimiter. To enable QGIS to properly parse the file, it is important to select the correct delimiter. You can specify a delimiter by activating *Custom delimiters*, or by activating *Regular expression delimiter* and entering text into the *Expression* field. For example, to change the delimiter to tab, use `\t` (this is a regular expression for the tab character).

Once the file is parsed, set *Geometry definition* to *Point coordinates* and choose the X and Y fields from the dropdown lists. If the coordinates are defined as degrees/minutes/seconds, activate the *DMS coordinates* checkbox.



Finally, enter a layer name (e.g., `elevp`), as shown in [figure_delimited_text_1](#). To add the layer to the map, click **[OK]**. The delimited text file now behaves as any other map layer in QGIS.

There is also a helper option that allows you to trim leading and trailing spaces from fields — *Trim fields*. Also, it is possible to *Discard empty fields*. If necessary, you can force a comma to be the decimal separator by activating *Decimal separator is comma*.

If spatial information is represented by WKT, activate the *Well Known Text* option and select the field with the WKT definition for point, line or polygon objects. If the file contains non-spatial data, activate *No geometry (attribute only table)* and it will be loaded as an ordinal table.

Additionally, you can enable:

- *Use spatial index* to improve the performance of displaying and spatially selecting features.



-  *Use subset index.*
-  *Watch file* to watch for changes to the file by other applications while QGIS is running.

12.1.5 OpenStreetMap data

In recent years, the OpenStreetMap project has gained popularity 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 or local knowledge. To support this objective, QGIS provides support for OSM data.

Loading OpenStreetMap Vectors





QGIS integrates OpenStreetMap import as a core functionality.

- To connect to the OSM server and download data, open the menu *Vector* → *Openstreetmap* → *Load data*. You can skip this step if you already obtained an `.osm` XML file using JOSM, Overpass API or any other source.
- The menu *Vector* → *Openstreetmap* → *Import topology from an XML file* will convert your `.osm` file into a SpatialLite database and create a corresponding database connection.
- The menu *Vector* → *Openstreetmap* → *Export topology to SpatialLite* then allows you to open the database connection, select the type of data you want (points, lines, or polygons) and choose tags to import. This creates a SpatialLite geometry layer that you can add to your project by clicking on the  *Add SpatialLite Layer* toolbar button or by selecting the  *Add SpatialLite Layer...* option from the *Layer* menu (see section *SpatialLite Layers*).

12.1.6 PostGIS Layers

PostGIS layers are stored in a PostgreSQL database. The advantages of PostGIS are the spatial indexing, filtering and query capabilities it provides. Using PostGIS, vector functions such as select and identify work more accurately than they do with OGR layers in QGIS.

Creating a stored Connection

 The first time you use a PostGIS data source, you must create a connection to the PostgreSQL database that contains the data. Begin by clicking on the  *Add PostGIS Layer* toolbar button, selecting the  *Add PostGIS Layer...* option from the *Layer* menu, or typing `Ctrl+Shift+D`. You can also open the *Add Vector Layer* dialog and select  *Database*. The *Add PostGIS Table(s)* dialog will be displayed. To access the connection manager, click on the **[New]** button to display the *Create a New PostGIS Connection* dialog. The parameters required for a connection are:

- **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`.
- **Host:** Name of the database host. This must be a resolvable host name such as would be used to open a telnet 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 is 5432.
- **Database:** Name of the database.

- **SSL mode:** How the SSL connection will be negotiated with the server. Note that massive speedups in PostGIS layer rendering can be achieved by disabling SSL in the connection editor. The following options are available:
 - Disable: Only try an unencrypted SSL connection.
 - Allow: Try a non-SSL connection. If that fails, try an SSL connection.
 - Prefer (the default): Try an SSL connection. If that fails, try a non-SSL connection.
 - Require: Only try an SSL connection.
- **Username:** User name used to log in to the database.
- **Password:** Password used with *Username* to connect to the database.

Optionally, you can activate the following checkboxes:

- *Save Username*
- *Save Password*
- *Only look in the geometry_columns table*
- *Don't resolve type of unrestricted columns (GEOMETRY)*
- *Only look in the 'public' schema*
- *Also list tables with no geometry*
- *Use estimated table metadata*

Once all parameters and options are set, you can test the connection by clicking on the **[Test Connect]** button.

Loading a PostGIS Layer



Once you have one or more connections defined, you can load layers from the PostgreSQL database. Of course, this requires having data in PostgreSQL. See section [Importing Data into PostgreSQL](#) for a discussion on importing data into the database.

To load a layer from PostGIS, perform the following steps:

- If the *Add PostGIS layers* dialog is not already open, selecting the **Add PostGIS Layer...** option from the *Layer* menu or typing `Ctrl+Shift+D` opens the dialog.
- Choose the connection from the drop-down list and click **[Connect]**.
- Select or unselect *Also list tables with no geometry*.
- Optionally, use some *Search Options* to define which features to load from the layer, or use the **[Build query]** button to start the *Query builder* dialog.
- Find the layer(s) you wish to add in the list of available layers.
- Select it by clicking on it. You can select multiple layers by holding down the `Shift` key while clicking. See section [Query Builder](#) for information on using the PostgreSQL Query Builder to further define the layer.
- Click on the **[Add]** button to add the layer to the map.

Tip: PostGIS Layers

Normally, a PostGIS layer is defined by an entry in the `geometry_columns` table. From version 0.9.0 on, QGIS can load layers that do not have an entry in the `geometry_columns` table. This includes both tables and views. Defining a spatial view provides a powerful means to visualize your data. Refer to your PostgreSQL manual for information on creating views.

Some details about PostgreSQL layers

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 any QGIS messages and give you direction on changing the PostgreSQL table or view definition to allow QGIS to load it.

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).


If the PostgreSQL layer is a view, the same requirement exists, but views do not 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).

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. It can make sense to disable this option when you use expensive views.

12.1.7 Importing Data into PostgreSQL

Data can be imported into PostgreSQL/PostGIS using several tools, including the SPIT 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 shapefiles and other data formats, and it includes support for schemas. See section *Plugin Pengelola DB* for more information.

shp2pgsql

PostGIS includes an utility called **shp2pgsql** that can be used to import shapefiles into a PostGIS-enabled database. For example, to import a shapefile 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 *Working with Projections* for more information on spatial reference systems and projections.

Tip: Exporting datasets from PostGIS

Like the import tool **shp2pgsql**, there is also a tool to export PostGIS datasets as shapefiles: **pgsql2shp**. This is shipped within your PostGIS distribution.


ogr2ogr

Besides **shp2pgsql** and **DB Manager**, there is another tool for feeding geodata in PostGIS: **ogr2ogr**. This is part of your GDAL installation.


To import a shapefile 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 `alaska.shp` into the PostGIS database `postgis` using the user `postgres` with the password `topsecret` on 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 PostgreSQL's **COPY** command instead of the default **INSERT INTO** method, you can export the following environment variable (at least available on  and **X**):

```
export PG_USE_COPY=YES
```

ogr2ogr does not create spatial indexes like **shp2pgsql** does. You need to create them manually, using the normal SQL command **CREATE INDEX** afterwards as an extra step (as described in the next section *Improving Performance*).

Improving Performance

Retrieving features from a PostgreSQL database can be time-consuming, especially over a network. You can improve the drawing performance of PostgreSQL layers by ensuring that a PostGIS spatial index exists on each layer in the database. PostGIS supports creation of a GiST (Generalized Search Tree) index to speed up spatial searches of the data (GiST index information is taken from the PostGIS documentation available at <http://postgis.refrains.net>).

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 *Literatur dan Referensi Web*) for more information.

The following is an example of creating 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$
```

12.1.8 Vector layers crossing 180° longitude

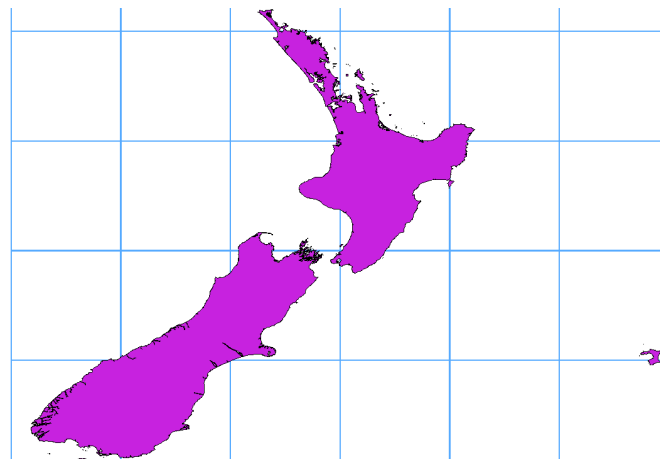
Many GIS packages don't wrap vector maps with a geographic reference system (lat/lon) crossing the 180 degrees longitude line (http://postgis.refrains.net/documentation/manual-2.0/ST_Shift_Longitude.html). As result, if

we open such a map in QGIS, we will see two far, distinct locations, that should appear near each other. In [Figure_vector_4](#), 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.



Gambar 12.5: 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 $< 0^\circ$, it adds 360° to it. The result is a $0^\circ - 360^\circ$ version of the data to be plotted in a 180° -centric map.






Gambar 12.6: Crossing 180° longitude applying the **ST_Shift_Longitude** function

Usage

- 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_5](#)).

12.1.9 Spatialite Layers

 The first time you load data from a Spatialite database, begin by clicking on the  Add Spatialite Layer toolbar button, or by selecting the  Add Spatialite Layer... option from the *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 can 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.

If you want to save a vector layer to SpatiaLite format, you can do this by right clicking the layer in the legend. Then, click on *Save as...*, define the name of the output file, and select 'SpatiaLite' as format and the CRS. Also, you can select 'SQLite' as format and then add `SPATIALITE=YES` in the OGR data source creation option field. This tells OGR to create a SpatiaLite database. See also http://www.gdal.org/ogr/drv_sqlite.html.

QGIS also supports editable views in SpatiaLite.




Creating a new SpatiaLite layer

If you want to create a new SpatiaLite layer, please refer to section *Creating a new SpatiaLite layer*.

Tip: SpatiaLite data management Plugins

For SpatiaLite data management, you can also use several Python plugins: QSpatiaLite, SpatiaLite Manager or DB Manager (core plugin, recommended). If necessary, they can be downloaded and installed with the Plugin Installer.




12.1.10 MSSQL Spatial Layers

 QGIS also provides native MS SQL 2008 support. The first time you load MSSQL Spatial data, begin by clicking on the  Add MSSQL Spatial Layer toolbar button or by selecting the  Add MSSQL Spatial Layer... option from the *Layer* menu, or by typing `Ctrl+Shift+M`.

12.1.11 Oracle Spatial Layers

The spatial features in Oracle Spatial aid users in managing geographic and location data in a native type within an Oracle database. QGIS now has support for such layers.






Creating a stored Connection

 The first time you use an Oracle Spatial data source, you must create a connection to the database that contains the data. Begin by clicking on the  Add Oracle Spatial Layer toolbar button, selecting the  Add Oracle Spatial Layer... option from the *Layer* menu, or typing `Ctrl+Shift+O`. To access the connection manager, click on the **[New]** button to display the *Create a New Oracle Spatial Connection* dialog. The parameters required for a connection are:

- **Name:** A name for this connection. It can be the same as *Database*
- **Database:** SID or SERVICE_NAME of the Oracle instance.
- **Host:** Name of the database host. This must be a resolvable host name such as would be used to open a telnet 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 is 1521.
- **Username:** Username used to login to the database.
- **Password:** Password used with *Username* to connect to the database.

Optionally, you can activate following checkboxes:



- *Save Username* Indicates whether to save the database username in the connection configuration.
- *Save Password* Indicates whether to save the database password in the connection settings.

-  *Only look in meta data 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, restrict 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 list the existing geometry types and don't offer to add others.


Once all parameters and options are set, you can test the connection by clicking on the **[Test Connect]** button.

Tip: QGIS User Settings and Security




Depending on your computing environment, storing passwords in your QGIS settings may be a security risk. Passwords are saved in clear text in the system configuration and in the project files! Your customized settings for QGIS are stored based on the operating system:

-  The settings are stored in your home directory in `~/.qgis2`.
-  The settings are stored in the registry.

Loading an Oracle Spatial Layer

 Once you have one or more connections defined, you can load layers from the Oracle database. Of course, this requires having data in Oracle.

To load a layer from Oracle Spatial, perform the following steps:

- If the *Add Oracle Spatial layers* dialog is not already open, click on the  **Add Oracle Spatial Layer** toolbar button.
- Choose the connection from the drop-down list and click **[Connect]**.
- Select or unselect  *Also list tables with no geometry*.
- Optionally, use some  *Search Options* to define which features to load from the layer or use the **[Build query]** button to start the *Query builder* dialog.
- Find the layer(s) you wish to add in the list of available layers.
- Select it by clicking on it. You can select multiple layers by holding down the `Shift` key while clicking. See section *Query Builder* for information on using the Oracle Query Builder to further define the layer.
- Click on the **[Add]** button to add the layer to the map.

Tip: Oracle Spatial Layers

Normally, an Oracle Spatial layer is defined by an entry in the `USER_SDO_METADATA` table.


12.2 The Symbol Library

12.2.1 Presentation

The Symbol Library is the place where users can create generic symbols to be used in several QGIS projects. It allows users to export and import symbols, groups symbols and add, edit and remove symbols. You can open it with the *Settings* → *Style Library* or from the **Style** tab in the vector layer's *Properties*.


Share and import symbols


Users can export and import symbols in two main formats: qml (QGIS format) and SLD (OGC standard). Note that SLD format is not fully supported by QGIS.

 share item displays a drop down list to let the user import or export symbols.

Groups and smart groups






Groups are categories of Symbols and smart groups are dynamic groups.

To create a group, right-click on an existing group or on the main **Groups** directory in the left of the library. You can also select a group and click on the  add item button.

To add a symbol into a group, you can either right click on a symbol then choose *Apply group* and then the group name added before. There is a second way to add several symbols into group: just select a group and click  and choose **Group Symbols**. All symbols display a checkbox that allow you to add the symbol into the selected groups. When finished, you can click on the same button, and choose **Finish Grouping**.

Create **Smart Symbols** is similar to creating group, but instead select **Smart Groups**. The dialog box allow user to choose the expression to select symbols in order to appear in the smart group (contains some tags, member of a group, have a string in its name, etc.)

Add, edit, remove symbol

With the *Style manager* from the [**Symbol**]  menu you can manage your symbols. You can  add item,  edit item,  remove item and  share item. 'Marker' symbols, 'Line' symbols, 'Fill' patterns and 'colour ramps' can be used to create the symbols. The symbols are then assigned to 'All Symbols', 'Groups' or 'Smart groups'.

For each kind of symbols, you will find always the same dialog structure:

- at the top left side a symbol representation
- under the symbol representation the symbol tree show the symbol layers
- at the right you can setup some parameter (unit,transparency, color, size and rotation)
- under these parameteres you find some symbol from the symbol library

The symbol tree allow adding, removing or protect new simple symbol. You can move up or down the symbol layer.

More detailed settings can be made when clicking on the second level in the *Symbol layers* dialog. You can define *Symbol layers* that are combined afterwards. A symbol can consist of several *Symbol layers*. Settings will be shown later in this chapter.

Tip: Note that once you have set the size in the lower levels of the *Symbol layers* dialog, the size of the whole symbol can be changed with the *Size* menu in the first level again. The size of the lower levels changes accordingly, while the size ratio is maintained.

12.2.2 Marker Symbols

Marker symbols have several symbol layer types:

- Ellipse marker
- Font marker
- Simple marker (default)
- SVG marker
- Vector Field marker

The following settings are possible:

- *Symbol layer type*: You have the option to use Ellipse markers, Font markers, Simple markers, SVG markers and Vector Field markers.
- *colors*
- *Size*
- *Outline style*
- *Outline width*
- *Angle*
- *Offset X,Y*: You can shift the symbol in the x- or y-direction.
- *Anchor point*
- *Data defined properties ...*

12.2.3 Line Symbols

Line marker symbols have only two symbol layer types:

- Marker line
- Simple line (default)

The default symbol layer type draws a simple line whereas the other display a marker point regularly on the line. You can choose different location vertex, interval or central point. Marker line can have offset along the line or offset line. Finally, *rotation* allows you to change the orientation of the symbol.

The following settings are possible:

- *colour*
- *Pen width*
- *Offset*
- *Pen style*
- *Join style*
- *Cap style*
- *Use custom dash pattern*
- *Dash pattern unit*
- *Data defined properties ...*

12.2.4 Polygon Symbols

Polygon marker symbols have also several symbol layer types:

- Centroid fill
- Gradient fill
- Line pattern fill
- Point pattern fill
- SVG fill
- Shapeburst fill
- Simple fill (default)
- Outline: Marker line (same as line marker)
- Outline: simple line (same as line marker)

The following settings are possible:

- *Colors* for the border and the fill.
- *Fill style*
- *Border style*
- *Border width*
- *Offset X,Y*
- *Data defined properties ...*

Using the color combo box, you can drag and drop color for one color button to another button, copy-paste color, pick color from somewhere, choose a color from the palette or from recent or standard color. The combo box allow you to fill in the feature with transparency. You can also just clic on the button to open the palette dialog. Note that you can import color from some external software like GIMP.

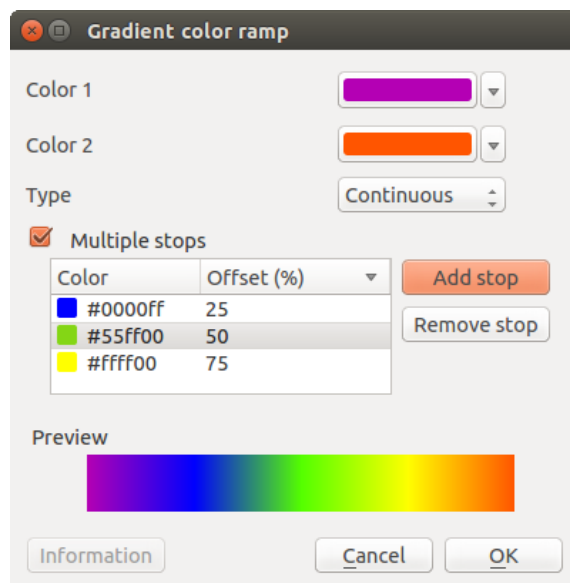
'Gradient Fill' *Symbol layer type* allows you to select between a *Two color* and *Color ramp* setting. You can use the *Feature centroid as Referencepoint*. All fills 'Gradient Fill' *Symbol layer type* is also available through the *Symbol* menu of the Categorized and Graduated Renderer and through the *Rule properties* menu of the Rule-based renderer. Other possibility is to choose a 'shapeburst fill' which is a buffered 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.

It is possible to only draw polygon borders inside the polygon. Using 'Outline: Simple line' select *Draw line only inside polygon*.

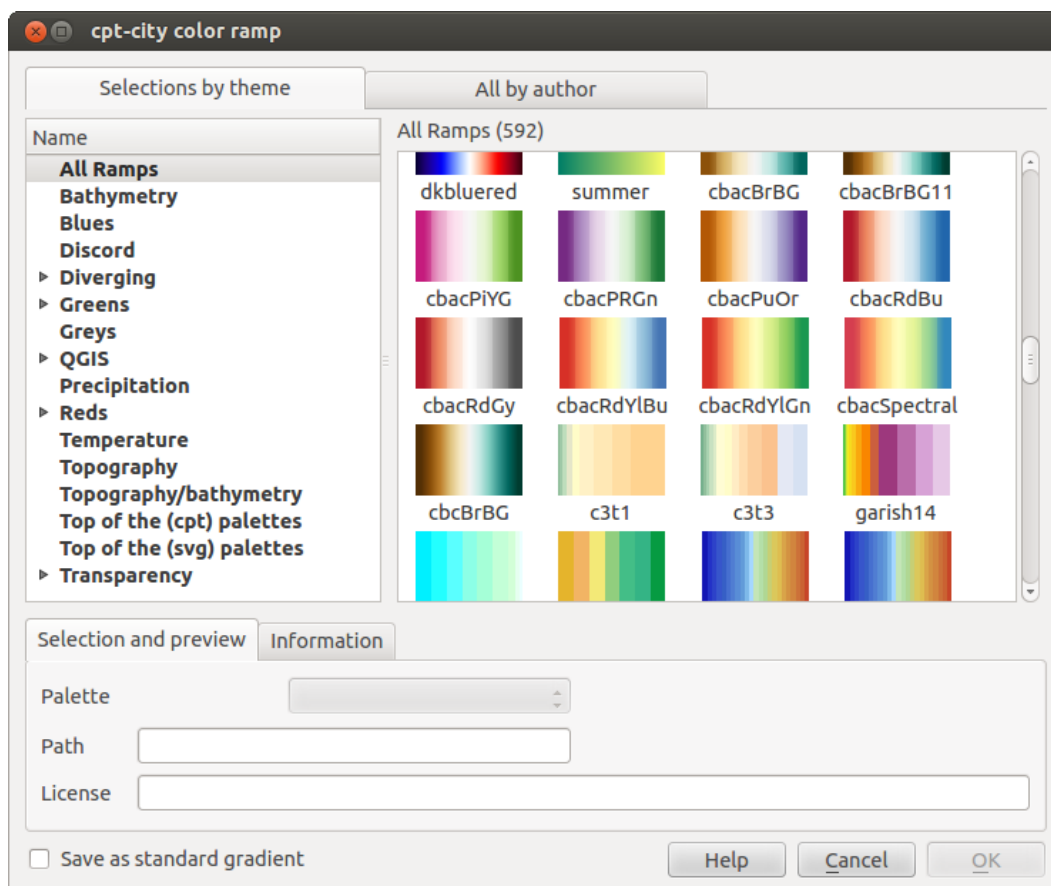
12.2.5 Color ramp

You can create a custom color ramp choosing *New color ramp...* from the *color ramp* drop-down menu. A dialog will prompt for the ramp type: Gradient, Random, colorBrewer, or cpt-city. The first three have options for number of steps and/or multiple stops in the color ramp. You can use the *Invert* option while classifying the data with a color ramp. See [figure_symbology_3](#) for an example of custom color ramp and [figure_symbology_3a](#) for the cpt-city dialog.

The cpt-city option opens a new dialog with hundreds of themes included 'out of the box'.



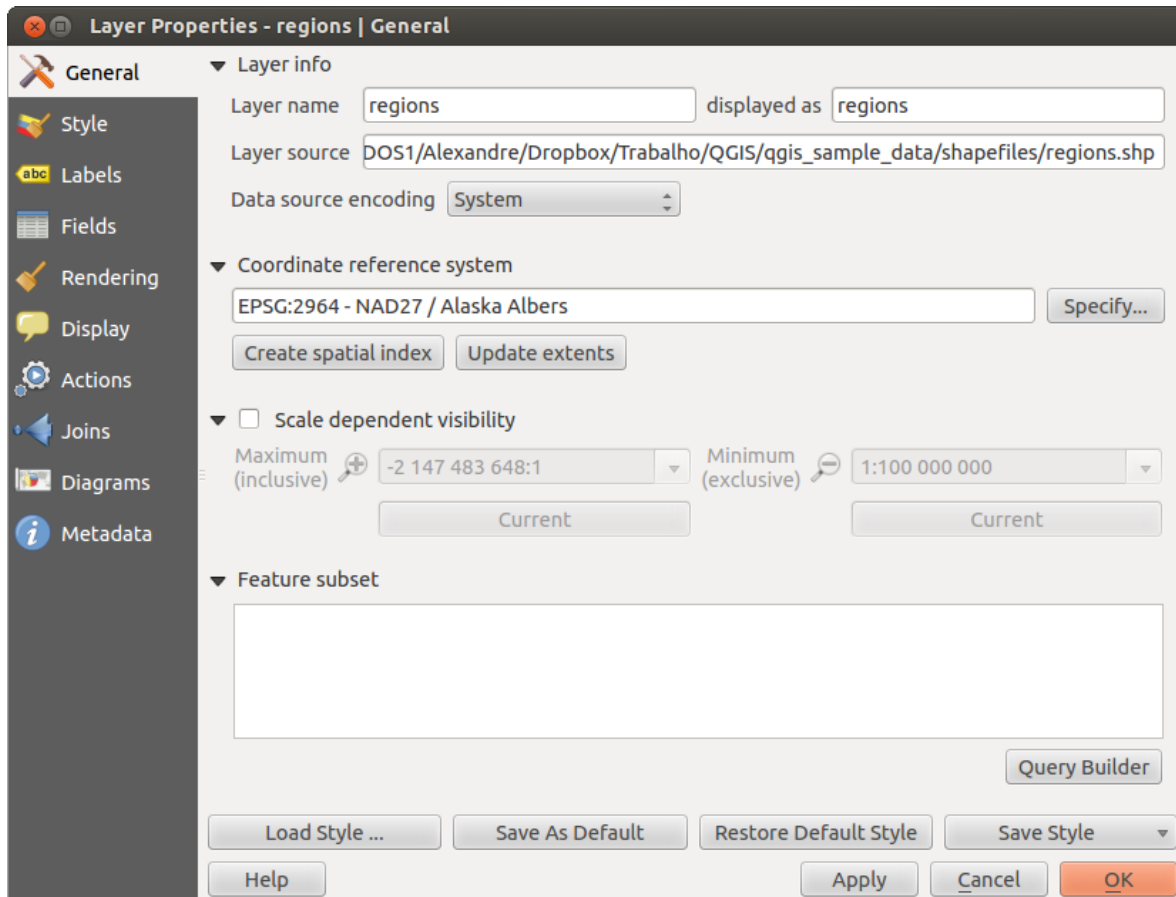
Gambar 12.7: Example of custom gradient color ramp with multiple stops 🐧



Gambar 12.8: cpt-city dialog with hundreds of color ramps 🐧

12.3 The Vector Properties Dialog

The *Layer Properties* dialog for a vector layer provides information about the layer, symbology settings and labeling options. If your vector layer has been loaded from a PostgreSQL/PostGIS datastore, you can also alter the underlying SQL for the layer by invoking the *Query Builder* dialog on the *General* tab. To access the *Layer Properties* dialog, double-click on a layer in the legend or right-click on the layer and select *Properties* from the pop-up menu.





Gambar 12.9: Vector Layer Properties Dialog 

12.3.1 Style Menu

The Style menu provides you with a comprehensive tool for rendering and symbolizing your vector data. You can use *Layer rendering* → tools that are common to all vector data, as well as special symbolizing tools that were designed for the different kinds of vector data.

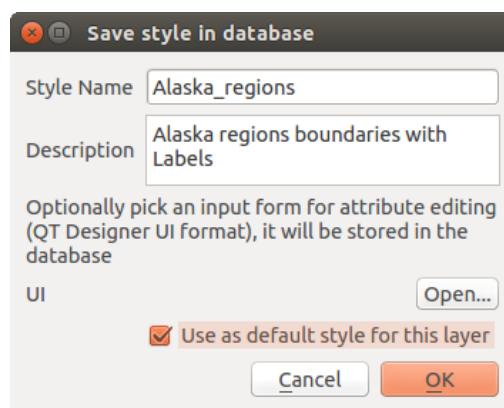
Renderers

The renderer is responsible for drawing a feature together with the correct symbol. There are four types of renderers: single symbol, categorized, graduated and rule-based. There is no continuous color renderer, because it is in fact only a special case of the graduated renderer. The categorized and graduated renderers can be created by specifying a symbol and a color ramp - they will set the colors for symbols appropriately. For point layers, there is a point displacement renderer available. For each data type (points, lines and polygons), vector symbol layer types are available. Depending on the chosen renderer, the *Style* menu provides different additional sections. On the bottom right of the symbology dialog, there is a **[Symbol]** button, which gives access to the Style Manager (see *Presentation*). The Style Manager allows you to edit and remove existing symbols and add new ones.

After having made any needed changes, the symbol can be added to the list of current style symbols (using [Symbol]  *Save in symbol library*), and then it can easily be used in the future. Furthermore, you can use the [Save Style]  button to save the symbol as a QGIS layer style file (.qml) or SLD file (.sld). 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. That means that categorized or graduated styles are converted to rule-based. If you want to preserve those renderers, you have to stick to the QML format. On the other hand, it can be very handy sometimes to have this easy way of converting styles to rule-based.

If you change the renderer type when setting the style of a vector layer the settings you made for the symbol will be maintained. Be aware that this procedure only works for one change. If you repeat changing the renderer type the settings for the symbol will get lost.

If the datasource of the layer is a database (PostGIS or Spatialite for example), you can save your layer style inside a table of the database. Just clic on :guilabel:‘ Save Style‘ comboxbox and choose **Save in database** item then fill in the dialog to define a style name, add a description, an ui file and if the style is a default style. When loading a layer from the database, if a style already exists for this layer, QGIS will load the layer and its style. You can add several style in the database. Only one will be the default style anyway.



Gambar 12.10: Save Style in database Dialog 

Tip: Select and change multiple symbols


The Symbology allows you to select multiple symbols and right click to change color, transparency, size, or width of selected entries.

Single Symbol Renderer

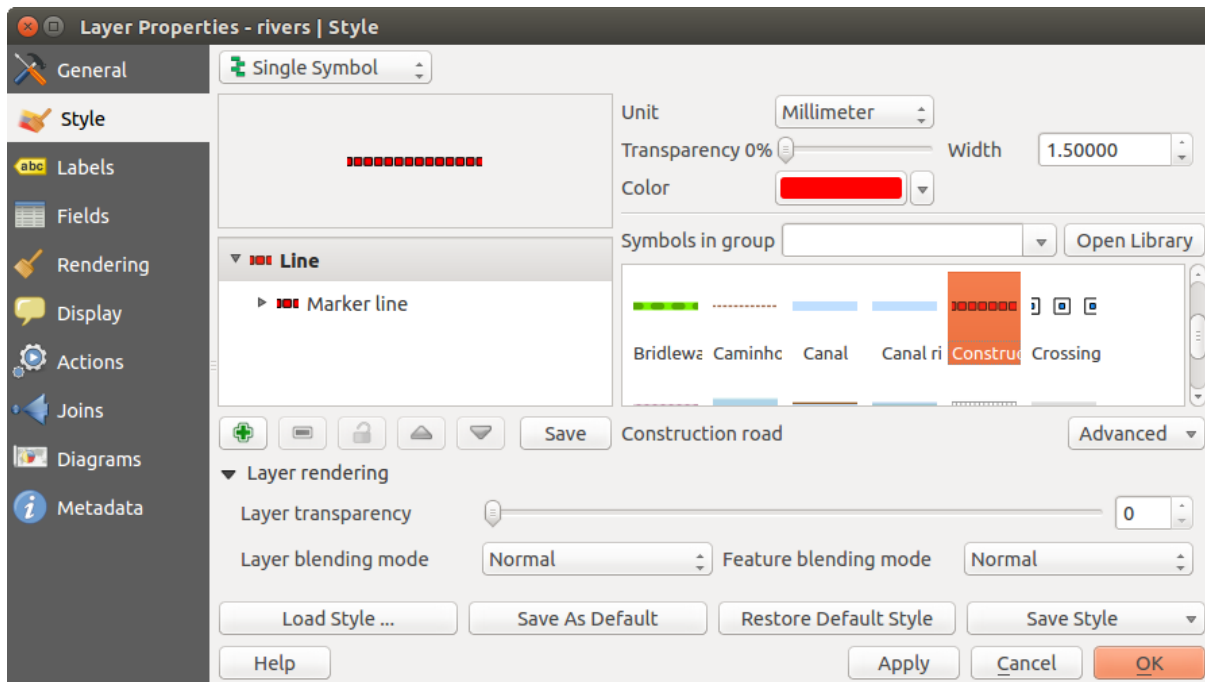
The Single Symbol Renderer is used to render all features of the layer using a single user-defined symbol. The properties, which can be adjusted in the *Style* menu, depend partially on the type of layer, but all types share the following dialog structure. In the top-left part of the menu, there is a preview of the current symbol to be rendered. On the right part of the menu, there is a list of symbols already defined for the current style, prepared to be used by selecting them from the list. The current symbol can be modified using the menu on the right side. If you click on the first level in the *Symbol layers* dialog on the left side, it's possible to define basic parameters like *Size*, *Transparency*, *color* and *Rotation*. Here, the layers are joined together.


Categorized Renderer

The Categorized Renderer is used to render all features from a layer, using a single user-defined symbol whose color reflects the value of a selected feature's attribute. The *Style* menu allows you to select:

- The attribute (using the Column listbox or the  *Set column expression* function, see [Expressions](#))
- The symbol (using the Symbol dialog)
- The colors (using the color Ramp listbox)

Then click on **Classify** button to create classes from the distinct value of the attribute column. Each classes can be disabled unchecking the checkbox at the left of the class name.



Gambar 12.11: Single symbol line properties 

You can change symbol, value and/or label of the clic, just double clicking on the item you want to change.

Right-clic shows a contextual menu to **Copy/Paste**, **Change color**, **Change transparency**, **Change output unit**, **Change symbol width**.

The [Advanced] button in the lower-right corner of the dialog allows you to set the fields containing rotation and size scale information. For convenience, the center of the menu lists the values of all currently selected attributes together, including the symbols that will be rendered.

The example in [figure_symbology_2](#) shows the category rendering dialog used for the rivers layer of the QGIS sample dataset.

Graduated Renderer

The Graduated Renderer is used to render all the features from a layer, using a single user-defined symbol whose color reflects the assignment of a selected feature's attribute to a class.

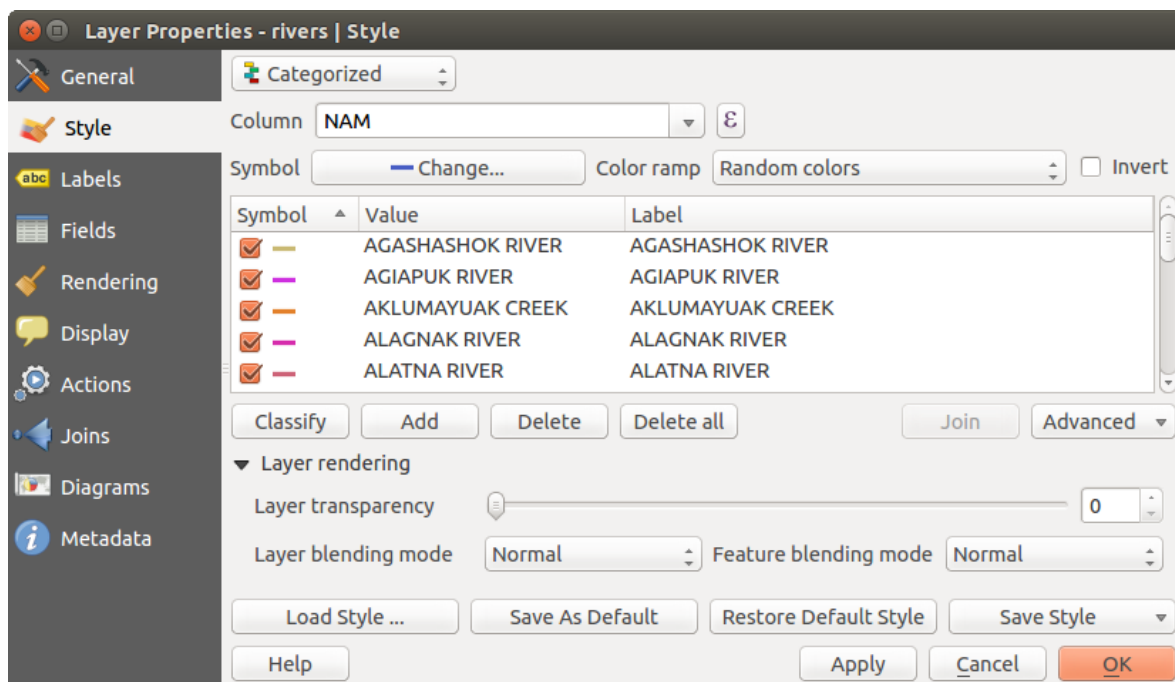
Like the Categorized Renderer, the Graduated Renderer allows you to define rotation and size scale from specified columns.

Also, analogous to the Categorized Renderer, the *Style* tab allows you to select:

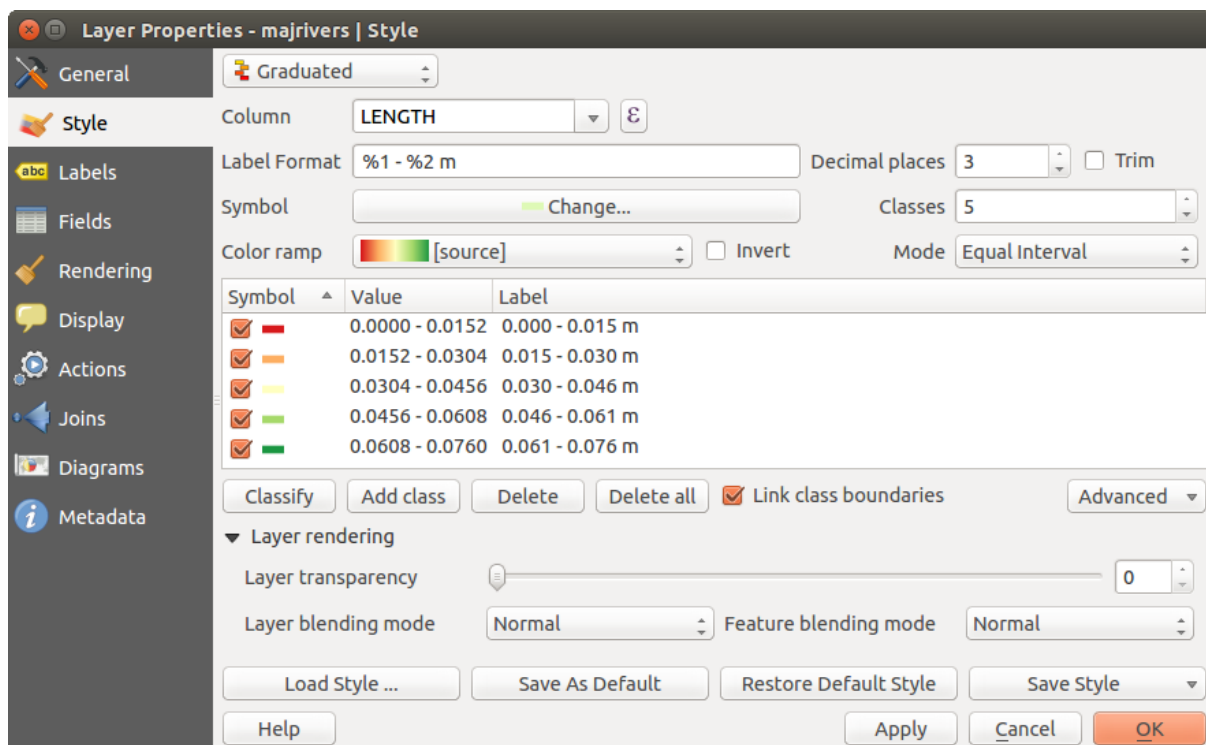
- The attribute (using the Column listbox or the ϵ ... *Set column expression* function, see [Expressions](#) chapter)
- The symbol (using the Symbol Properties button)
- The colors (using the color Ramp list)

Additionally, you can specify the number of classes and also the mode for classifying features within the classes (using the Mode list). The available modes are:

- 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;



Gambar 12.12: Categorized Symbolizing options 🐧



Gambar 12.13: Graduated Symbolizing options 🐧

- Pretty Breaks: the same of natural breaks but the extremes number of each class are integers.

The listbox in the center part of the *Style* menu lists the classes together with their ranges, labels and symbols that will be rendered.

Click on **Classify** button to create classes using the choosen mode. Each classes can be disabled unchecking the checkbox at the left of the class name.

You can change symbol, value and/or label of the clic, just double clicking on the item you want to change.

Right-clic shows a contextual menu to **Copy/Paste**, **Change color**, **Change transparency**, **Change output unit**, **Change symbol width**.

The example in [figure_symbology_4](#) shows the graduated rendering dialog for the rivers layer of the QGIS sample dataset.


Tip: Thematic maps using an expression

Categorized and graduated thematic maps can now be created using the result of an expression. In the properties dialog for vector layers, the attribute chooser has been augmented with a \mathcal{E} ... *Set column expression* function. So now you no longer 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.

Rule-based rendering

The Rule-based Renderer is used to render all the features from a layer, using rule based symbols whose color reflects the assignment of a selected feature’s attribute to a class. The rules are based on SQL statements. 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.

The example in [figure_symbology_5](#) shows the rule-based rendering dialog for the rivers layer of the QGIS sample dataset.

To create a rule, activate an existing row by double-clicking on it, or click on ‘+’ and click on the new rule. In the *Rule properties* dialog, you can define a label for the rule. Press the  button to open the expression string builder. In the **Function List**, click on *Fields and Values* to view all attributes of the attribute table to be searched. To add an attribute to the field calculator **Expression** field, double click its name in the *Fields and Values* list. Generally, you can use the various fields, values and functions to construct the calculation expression, or you can just type it into the box (see *Expressions*). You can create a new rule by copying and pasting an existing rule with the right mouse button. You can also use the ‘ELSE’ rule that will be run if none of the other rules on that level match. Since QGIS 2.6 the label for the rules appears in a pseudotree in the map legend. Just double-klick the rules in the map legend and the Style menu of the layer properties appears showing the rule that is the background for the symbol in the pseudotree.

Point displacement

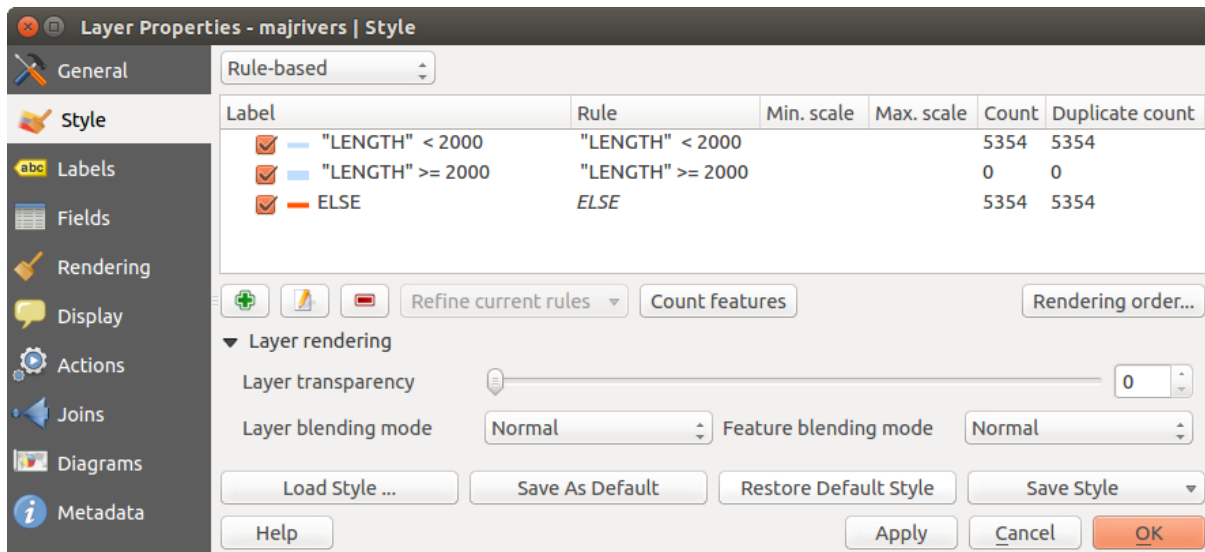
The Point Displacement Renderer works to visualize all features of a point layer, even if they have the same location. To do this, the symbols of the points are placed on a displacement circle around a center symbol.

Tip: Export vector symbology

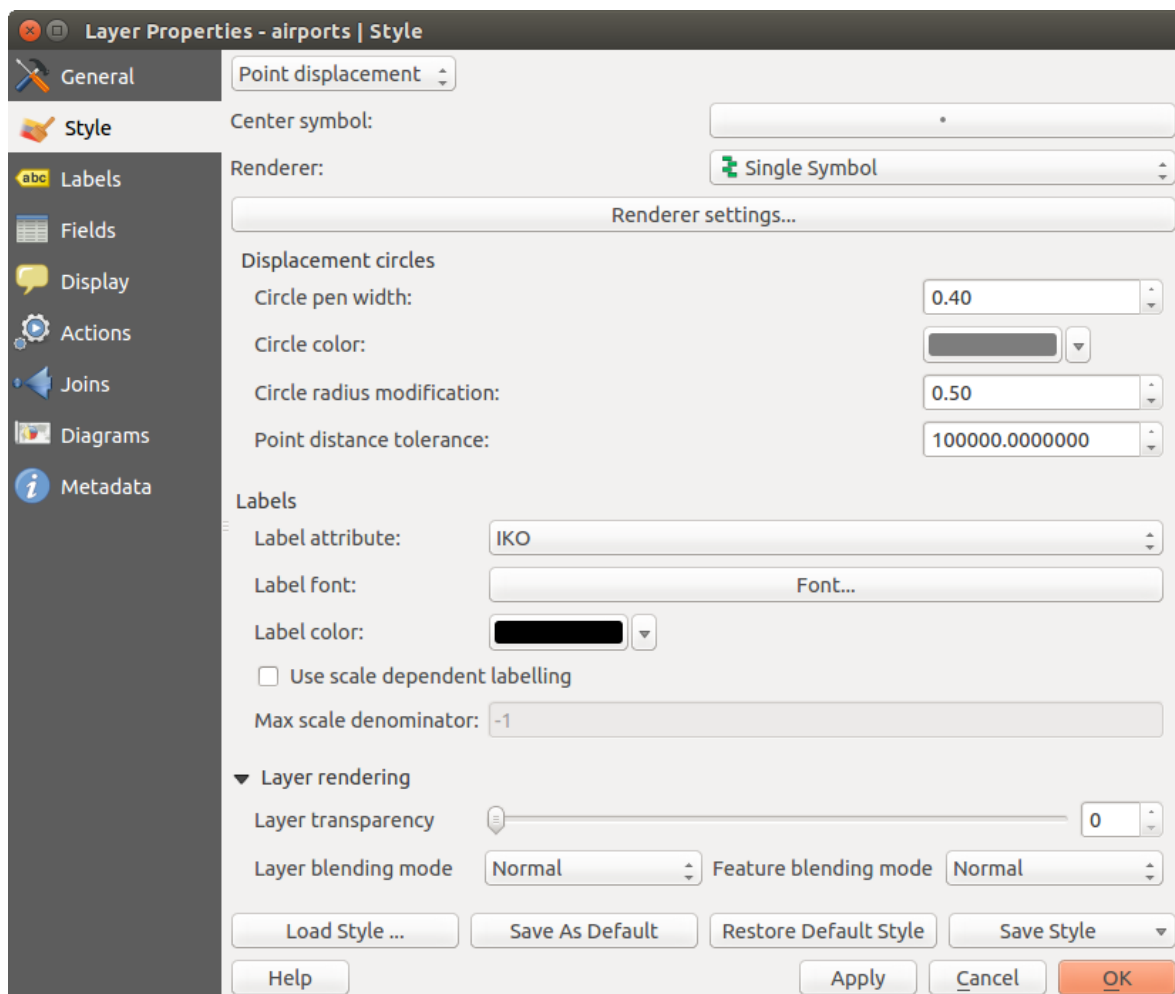
You have the option to export vector symbology from QGIS into Google *.kml, *.dxf and MapInfo *.tab files. Just open the right mouse menu of the layer and click on *Save selection as* → to specify the name of the output file and its format. In the dialog, use the *Symbology export* menu to save the symbology either as *Feature symbology* → or as *Symbol layer symbology* →. If you have used symbol layers, it is recommended to use the second setting.

Inverted Polygon

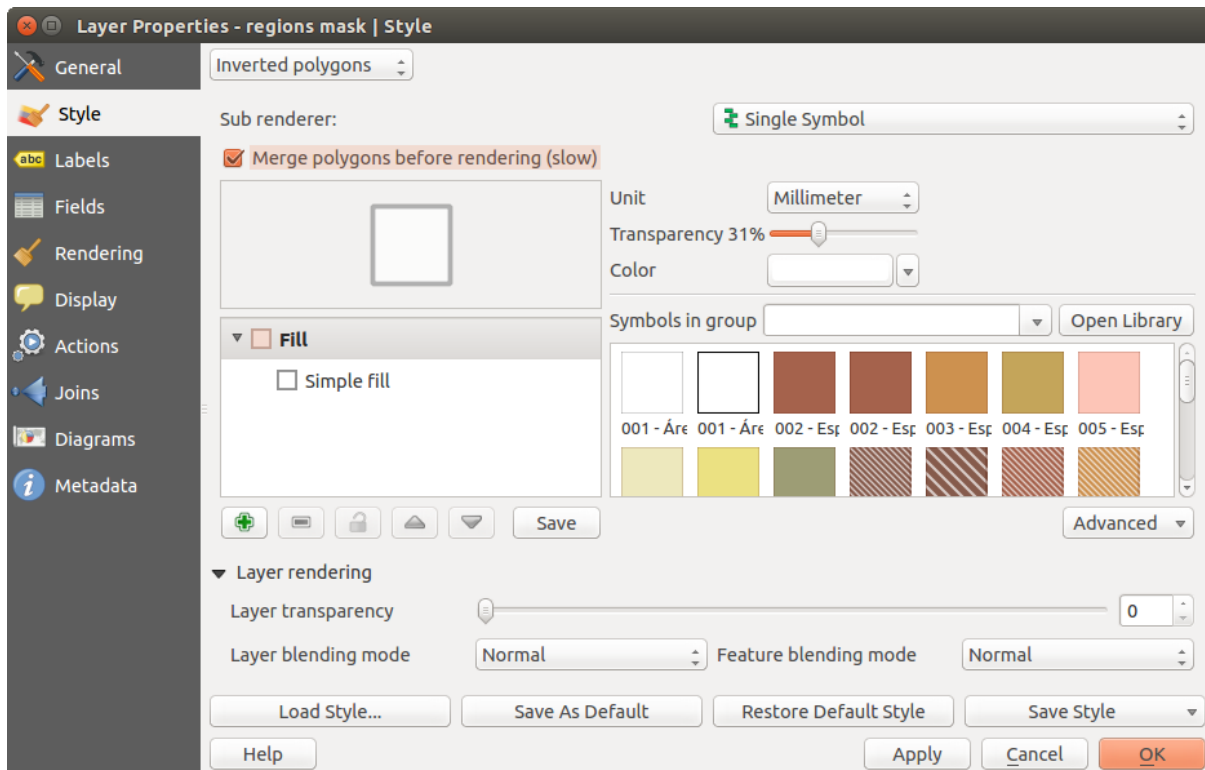
Inverted polygon renderer allows user to define a symbol to fill in outside of the layer’s polygons. As before you can select a subrenderers. These subrenderers are the same as for the main renderers.



Gambar 12.14: Rule-based Symbolizing options 🐧







Gambar 12.15: Point displacement dialog 🐧







Gambar 12.16: Inverted Polygon dialog 

Color Picker

Regardless the type of style to be used, the *select color* dialog will show when you click to choose a color - either border or fill color. This dialog has four different tabs which allow you to select colors by  color ramp,  color wheel,  color swatches or  color 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. There is also an *opacity* slider to set transparency level. On the lower left part of the dialog you can see a comparison between the *current* and the *new* color you are presently selecting and on the lower right part you have the option to add the color you just tweaked into a color slot button.


With  color ramp or with  color wheel, you can browse to all possible color combinations. There are other possibilities though. By using *color swatches*  you can choose from a preselected list. This selected list is populated with one of three methods: *Recent colors*, *Standard colors* or *Project colors*

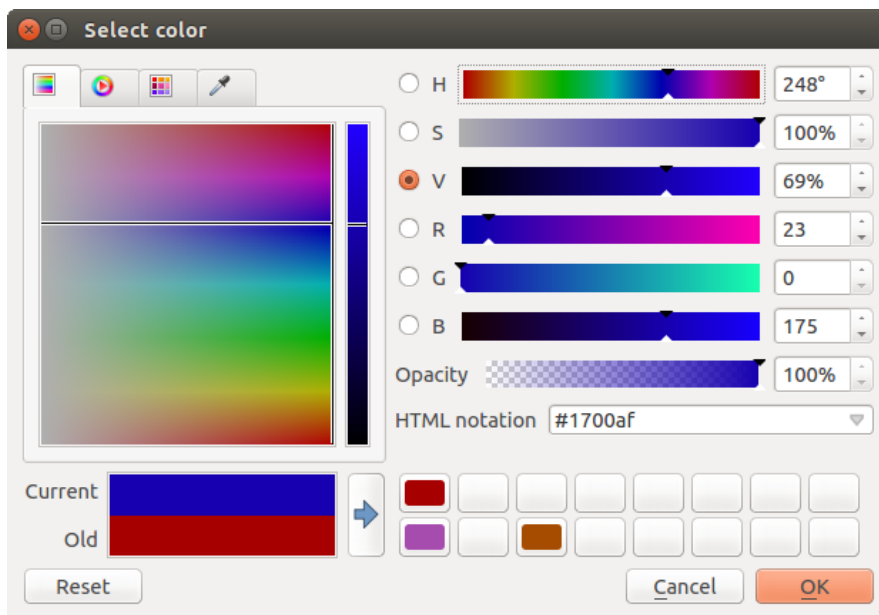
Another option is to use the  color picker which allows you to sample a color from under your mouse pointer at any part of QGIS or even from another application by pressing the space bar. Please note that the color picker is OS dependent and is currently not supported by OSX.

Tip: quick color picker + copy/paste colors

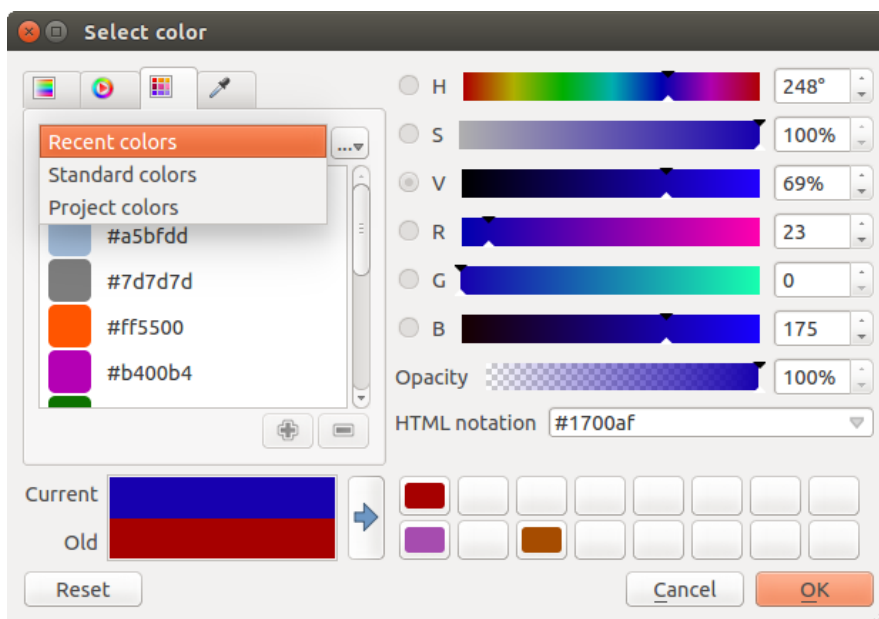
You can quickly choose from *Recent colors*, from *Standard colors* or simply *copy* or *paste* a color by clicking the drop-down arrow that follows a current color box.

Layer rendering

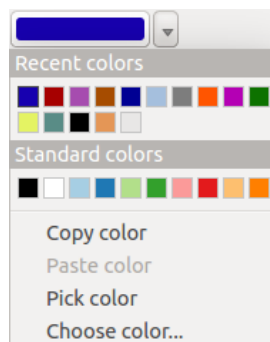
- *Layer transparency* : You can make the underlying layer in the map canvas visible with this tool. Use the slider to adapt the visibility of your vector layer to your needs. You can also make a



Gambar 12.17: Color picker ramp tab 🐧



Gambar 12.18: Color picker swatcher tab 🐧




Gambar 12.19: Quick color picker menu 🐧

precise definition of the percentage of visibility in the the menu beside the slider.

- *Layer blending mode* and *Feature blending mode*: You can achieve special rendering effects with these tools that you may previously only know from graphics programs. The pixels of your overlaying and underlaying layers are mixed through the settings described below.
 - Normal: This is the standard blend mode, which uses the alpha channel of the top pixel to blend with the pixel beneath it. The colors aren't mixed.
 - Lighten: This selects the maximum of each component from the foreground and background pixels. Be aware that the results tend to be jagged and harsh.
 - Screen: Light pixels from the source are painted over the destination, while dark pixels are not. This mode is most useful for mixing the texture of one layer with another layer (e.g., you can use a hillshade to texture another layer).
 - Dodge: Dodge will brighten and saturate underlying pixels based on the lightness of the top pixel. So, 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: This blend mode simply adds pixel values of one layer with the other. In case of values above one (in the case of RGB), white is displayed. This mode is suitable for highlighting features.
 - Darken: This creates a resultant pixel that retains the smallest components of the foreground and background pixels. Like lighten, the results tend to be jagged and harsh.
 - Multiply: Here, the numbers for each pixel of the top layer are multiplied with the corresponding pixels for the bottom layer. The results are darker pictures.
 - Burn: Darker colors in the top layer cause the underlying layers to darken. Burn can be used to tweak and colorise underlying layers.
 - Overlay: This mode combines the multiply and screen blending modes. In the resulting picture, light parts become lighter and dark parts become darker.
 - Soft light: This is very similar to overlay, but instead of using multiply/screen it uses color burn/dodge. This is supposed to emulate shining a soft light onto an image.
 - Hard light: Hard light is also very similar to the overlay mode. It's supposed to emulate projecting a very intense light onto an image.
 - Difference: Difference subtracts the top pixel from the bottom pixel, or the other way around, to always get a positive value. Blending with black produces no change, as the difference with all colors is zero.
 - Subtract: This blend mode simply subtracts pixel values of one layer from the other. In case of negative values, black is displayed.


12.3.2 Labels Menu


The  Labels core application provides smart labeling for vector point, line and polygon layers, and it only requires a few parameters. This new application also supports on-the-fly transformed layers. The core functions of the application have been redesigned. In QGIS, there are a number of other features that improve the labeling. The following menus have been created for labeling the vector layers:

- Text
- Formatting
- Buffer
- Background
- Shadow
- Placement

- Rendering

Let us see how the new menus can be used for various vector layers. **Labeling point layers**

Start QGIS and load a vector point layer. Activate the layer in the legend and click on the  Layer Labeling Options icon in the QGIS toolbar menu.

The first step is to activate the *Label this layer with* checkbox and select an attribute column to use for labeling. Click  if you want to define labels based on expressions - See [labeling_with_expressions](#).

The following steps describe a simple labeling without using the *Data defined override* functions, which are situated next to the drop-down menus.

You can define the text style in the *Text* menu (see [Figure_labels_1](#)). Use the *Type case* option to influence the text rendering. You have the possibility to render the text 'All uppercase', 'All lowercase' or 'Capitalize first letter'. Use the blend modes to create effects known from graphics programs (see [blend_modes](#)).

In the *Formatting* menu, you can define a character for a line break in the labels with the 'Wrap on character' function. Use the *Formatted numbers* option to format the numbers in an attribute table. Here, decimal places may be inserted. If you enable this option, three decimal places are initially set by default.

To create a buffer, just activate the *Draw text buffer* checkbox in the *Buffer* menu. The buffer color is variable. Here, you can also use blend modes (see [blend_modes](#)).

If the *color buffer's fill* checkbox is activated, it will interact with partially transparent text and give mixed color transparency results. Turning off the buffer fill fixes that issue (except where the interior aspect of the buffer's stroke intersects with the text's fill) and also allows you to make outlined text.


In the *Background* menu, you can define with *Size X* and *Size Y* the shape of your background. Use *Size type* to insert an additional 'Buffer' into your background. The buffer size is set by default here. The background then consists of the buffer plus the background in *Size X* and *Size Y*. You can set a *Rotation* where you can choose between 'Sync with label', 'Offset of label' and 'Fixed'. Using 'Offset of label' and 'Fixed', you can rotate the background. Define an *Offset X,Y* with X and Y values, and the background will be shifted. When applying *Radius X,Y*, the background gets rounded corners. Again, it is possible to mix the background with the underlying layers in the map canvas using the *Blend mode* (see [blend_modes](#)).

Use the *Shadow* menu for a user-defined *Drop shadow*. The drawing of the background is very variable. Choose between 'Lowest label component', 'Text', 'Buffer' and 'Background'. The *Offset* angle depends on the orientation of the label. If you choose 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. You can influence the appearance of the shadow with the *Blur radius*. The higher the number, the softer the shadows. The appearance of the drop shadow can also be altered by choosing a blend mode (see [blend_modes](#)).

Choose the *Placement* menu for the label placement and the labeling priority. Using the *Offset from point* setting, you now have the option to use *Quadrants* to place your label. Additionally, you can alter the angle of the label placement with the *Rotation* setting. Thus, a placement in a certain quadrant with a certain rotation is possible.

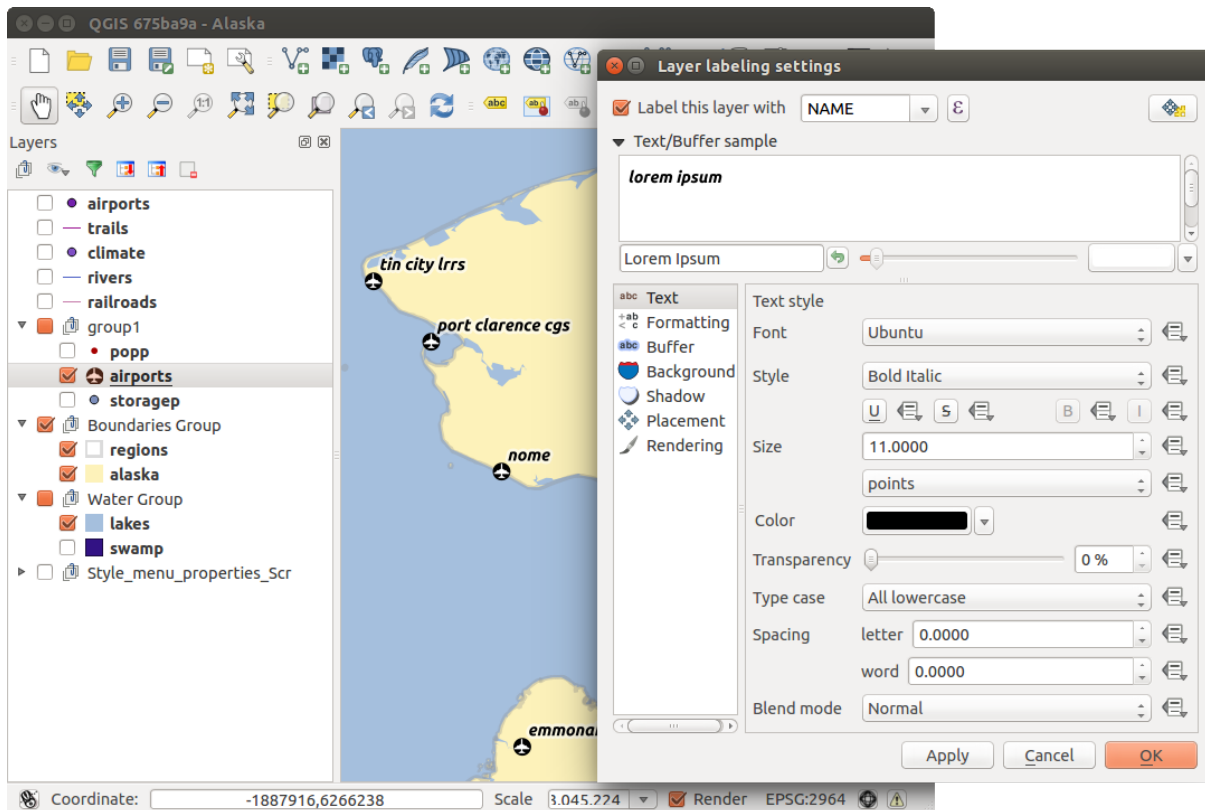
In the *Rendering* menu, you can define label and feature options. Under *Label options*, you find the scale-based visibility setting now. You can prevent QGIS from rendering only selected labels with the *Show all labels for this layer (including colliding labels)* checkbox. Under *Feature options*, you can define whether every part of a multipart feature is to be labeled. It's possible to define whether the number of features to be labeled is limited and to *Discourage labels from covering features*.

Labeling line layers

The first step is to activate the *Label this layer* checkbox in the *Label settings* tab and select an attribute column to use for labeling. Click  if you want to define labels based on expressions - See [labeling_with_expressions](#).

After that, you can define the text style in the *Text* menu. Here, you can use the same settings as for point layers.

Also, in the *Formatting* menu, the same settings as for point layers are possible.



Gambar 12.20: Smart labeling of vector point layers 🐧

The *Buffer* menu has the same functions as described in section [labeling_point_layers](#).

The *Background* menu has the same entries as described in section [labeling_point_layers](#).

Also, the *Shadow* menu has the same entries as described in section [labeling_point_layers](#).

In the *Placement* menu, you find special settings for line layers. The label can be placed *Parallel*, *Curved* or *Horizontal*. With the *Parallel* and *Curved* option, you can define the position *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 position of the label. Remember that here 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_2](#)).

You can set up a minimum distance for repeating labels. Distance can be in mm or in map units.

Some Placement setup will display more options, for example, *Curved* and *Parallel* Placements will allow the user to set up the position of the label (above, below or on the line), *distance* from the line and for *Curved*, the user can also setup inside/outside max angle between curved label.

The *Rendering* menu has nearly the same entries as for point layers. In the *Feature options*, you can now *Suppress labeling of features smaller than*.

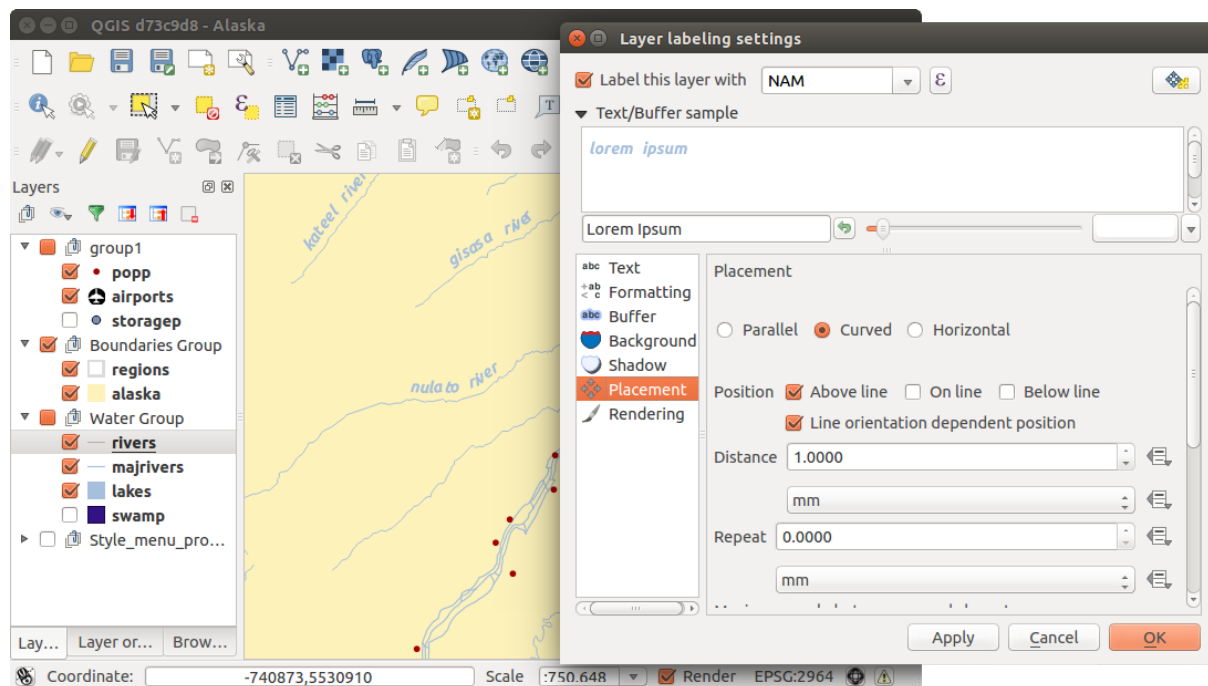
Labeling polygon layers


The first step is to activate the *Label this layer* checkbox and select an attribute column to use for labeling. Click **E...** if you want to define labels based on expressions - See [labeling_with_expressions](#).

In the *Text* menu, define the text style. The entries are the same as for point and line layers.

The *Formatting* menu allows you to format multiple lines, also similar to the cases of point and line layers.

As with point and line layers, you can create a text buffer in the *Buffer* menu.



Gambar 12.21: Smart labeling of vector line layers 

Use the *Background* menu to create a complex user-defined background for the polygon layer. You can use the menu also as with the point and line layers.



The entries in the *Shadow* menu are the same as for point and line layers.

In the *Placement* menu, you find special settings for polygon layers (see Figure_labels_3). *Offset from centroid*, *Horizontal (slow)*, *Around centroid*, *Free* and *Using perimeter* are possible.

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 with the quadrants here, and define offset and rotation. The *Around centroid* setting makes it possible to place the label around the centroid with a certain distance. Again, you can define *visible polygon* or *whole polygon* for the centroid. With the *Using perimeter* settings, 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.

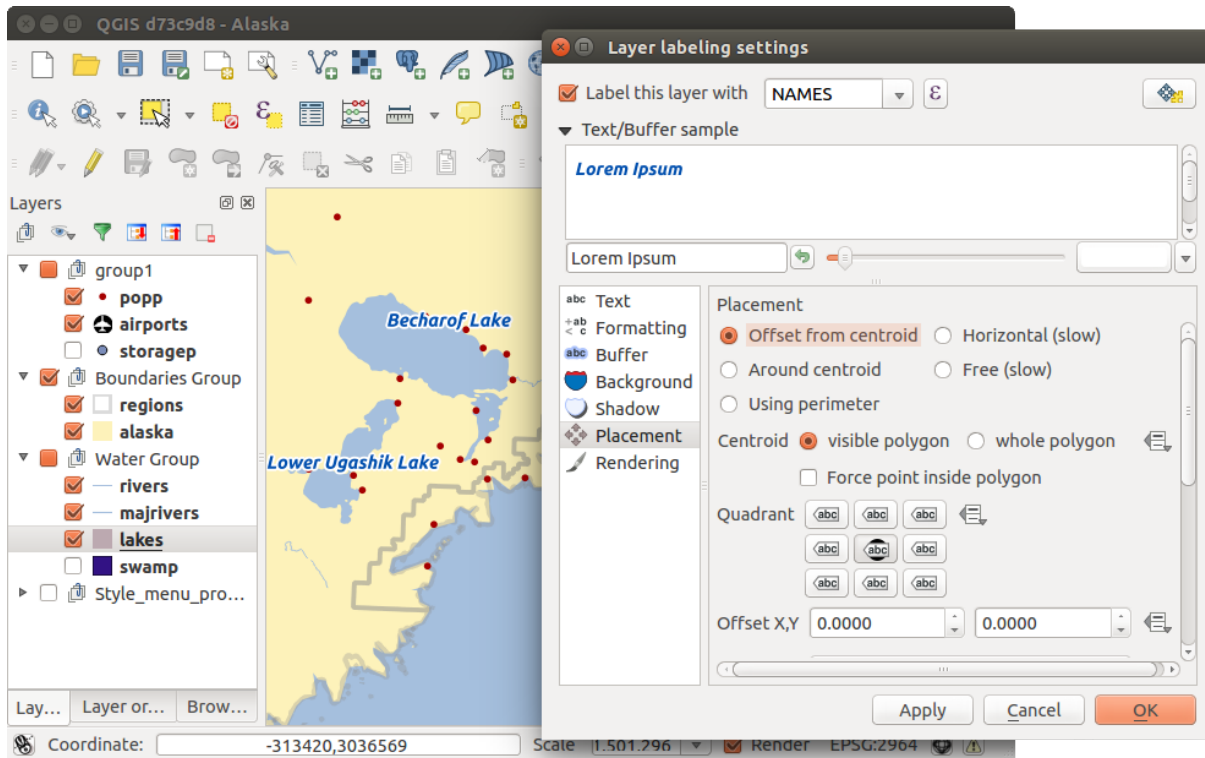
Related to the choose of Label Placement, several options will appear. As for Point Placement you can choose the distance for the polygon outline, repeat the label around the polygon perimeter.

The entries in the *Rendering* menu are the same as for line layers. You can also use *Suppress labeling of features smaller than* in the *Feature options*. **Define labels based on expressions**

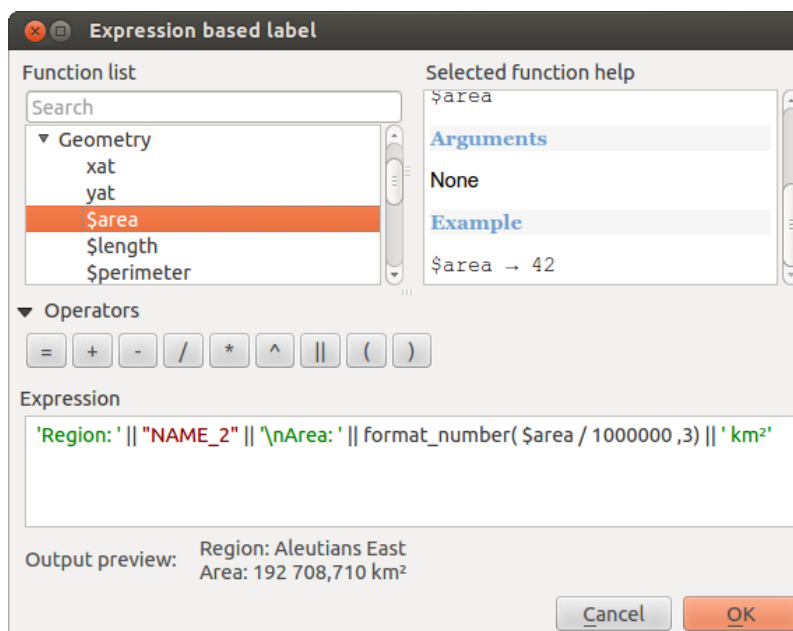
QGIS allows to use expressions to label features. Just click the  icon in the  Labels menu of the properties dialog. In figure_labels_4 you see a sample expression to label the alaska regions with name and area size, based on the field 'NAME_2', some descriptive text and the function '\$area()' in combination 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 need to combine all elements (strings, fields and functions) with a string concatenation sign '||' and that fields a written in "double quotes" and strings in 'single quotes'. Let's have a look at some examples:

```
# label based on two fields 'name' and 'place' with a comma as separator
"name" || ', ' || "place"
```

Gambar 12.22: Smart labeling of vector polygon layers 🐧



Gambar 12.23: Using expressions for labeling 🐧

```
-> John Smith, Paris

# label based on two fields 'name' and 'place' separated by comma
'My name is ' || "name" || 'and I live in ' || "place"

-> My name is John Smith and I live in Paris

# label based on two fields 'name' and 'place' with a descriptive text
# and a line break (\n)
'My name is ' || "name" || '\nI live in ' || "place"

-> My name is John Smith
    I live in Paris

# create a multi-line label based on a field and the $area function
# to show the place name and its area size based on unit meter.
'The area of ' || "place" || 'has a size of ' || $area || 'm²'


-> The area of Paris has a size of 105000000 m²



# create a CASE ELSE condition. If the population value in field
# population is <= 50000 it is a town, otherwise a city.
'This place is a ' || CASE WHEN "population <= 50000" THEN 'town' ELSE 'city' END

-> This place is a town
```





As you can see in the expression builder, you have hundreds of functions available to create simple and very complex expressions to label your data in QGIS. See [Expressions](#) chapter for more information and example on expressions.

Using data-defined override for labeling

With the data-defined override functions, the settings for the labeling are overridden by entries in the attribute table. You can activate and deactivate the function with the right-mouse button. Hover over the symbol and you see the information about the data-defined override, including the current definition field. We now describe an example using the data-defined override function for the  Move label function (see [figure_labels_5](#)).

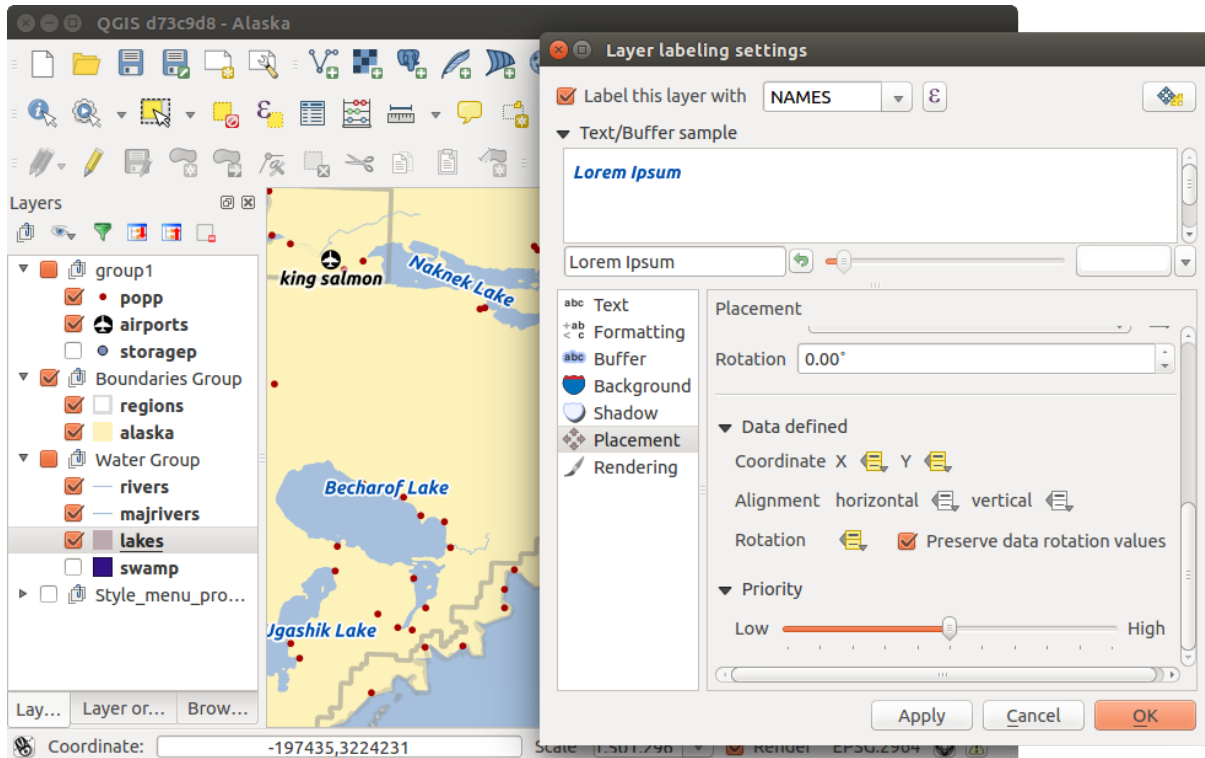
1. Import `lakes.shp` from the QGIS sample dataset.
2. Double-click the layer to open the Layer Properties. Click on *Labels* and *Placement*. Select *Offset from centroid*.
3. Look for the *Data defined* entries. Click the  icon to define the field type for the *Coordinate*. Choose 'xlabel' for X and 'ylabel' for Y. The icons are now highlighted in yellow.
4. Zoom into a lake.
5. Go to the Label toolbar and click the  icon. Now you can shift the label manually to another position (see [figure_labels_6](#)). The new position of the label is saved in the 'xlabel' and 'ylabel' columns of the attribute table.


12.3.3 Fields Menu

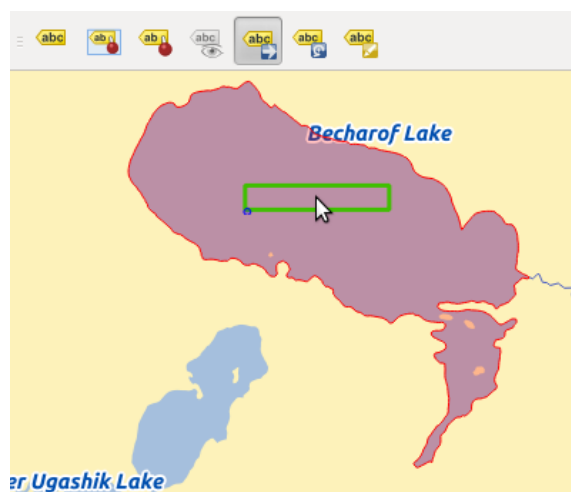
 Within the *Fields* menu, the field attributes of the selected dataset can be manipulated. The buttons  New Column and  Delete Column can be used when the dataset is in  Editing mode.


Edit Widget

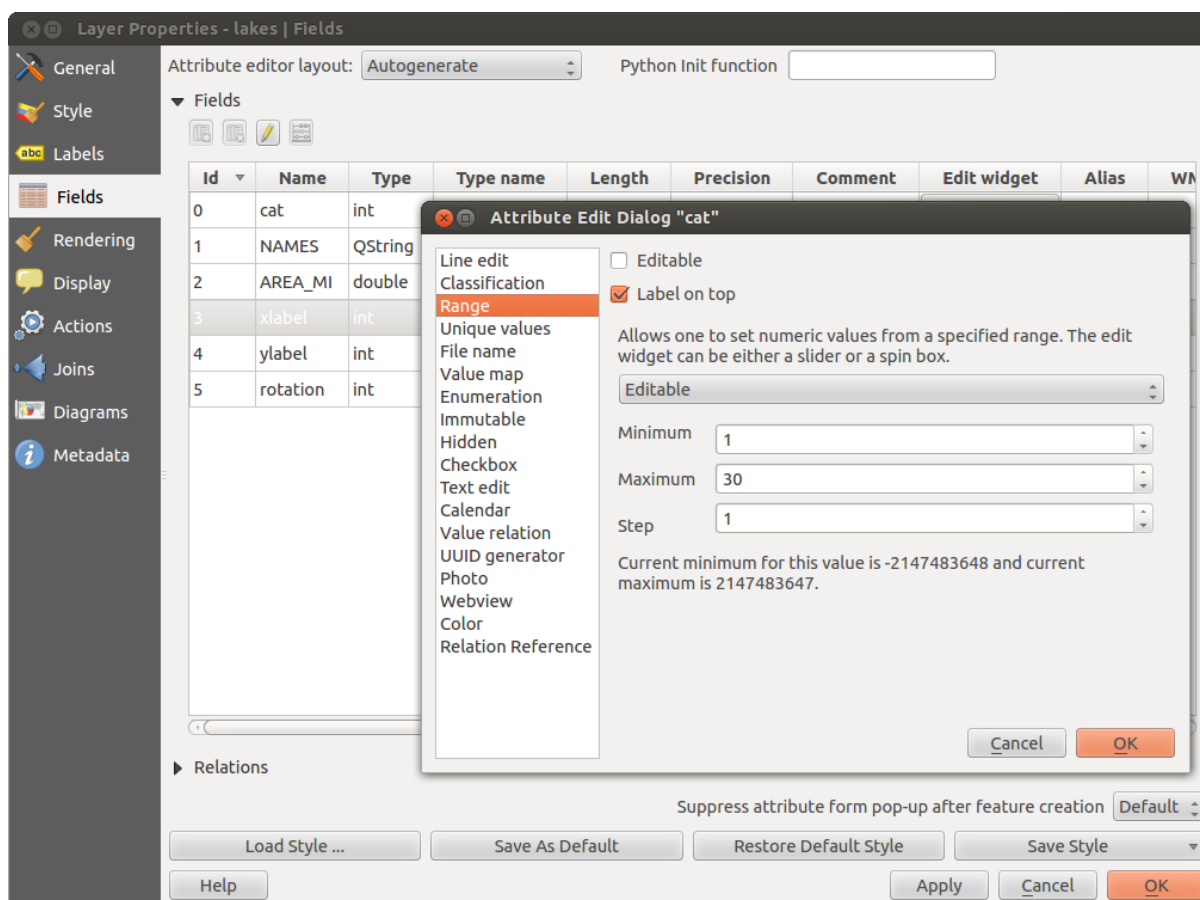
Within the *Fields* menu, you also find an **edit widget** column. This column can be used to define values or a range of values that are allowed to be added to the specific attribute table column. If you click on the **[edit widget]** button, a dialog opens, where you can define different widgets. These widgets are:



Gambar 12.24: Labeling of vector polygon layers with data-defined override 





Gambar 12.25: Move labels 



Gambar 12.26: Dialog to select an edit widget for an attribute column 🐧

- **Checkbox:** Displays a checkbox, and you can define what attribute is added to the column when the checkbox is activated or not.
- **Classification:** Displays a combo box with the values used for classification, if you have chosen 'unique value' as legend type in the *Style* menu of the properties dialog.
- **Color:** Displays a color button allowing user to choose a color from the color dialog window.
- **Date/Time:** Displays a line fields which can opens a calendar widget to enter a date, a time or both. Column type must be text. You can select a custom format, pop-up a calendar, etc.
- **Enumeration:** Opens a combo box with values that can be used within the columns type. This is currently only supported by the PostgreSQL provider.
- **File name:** Simplifies the selection by adding a file chooser dialog.
- **Hidden:** A hidden attribute column is invisible. The user is not able to see its contents.
- **Photo:** Field contains a filename for a picture. The width and height of the field can be defined.
- **Range:** Allows you to set numeric values from a specific range. The edit widget can be either a slider or a spin box.
- **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 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.
- **Webview:** Field contains a URL. The width and height of the field is variable.

With the **Attribute editor layout**, you can now define built-in forms for data entry jobs (see [figure_fields_2](#)).

Choose 'Drag and drop designer' and an attribute column. Use the  icon to create a category that will then be shown during the digitizing session (see [figure_fields_3](#)). The next step will be to assign the relevant fields to the category with the  icon. You can create more categories and use the same fields again. When creating a new category, QGIS will insert a new tab for the category in the built-in form.

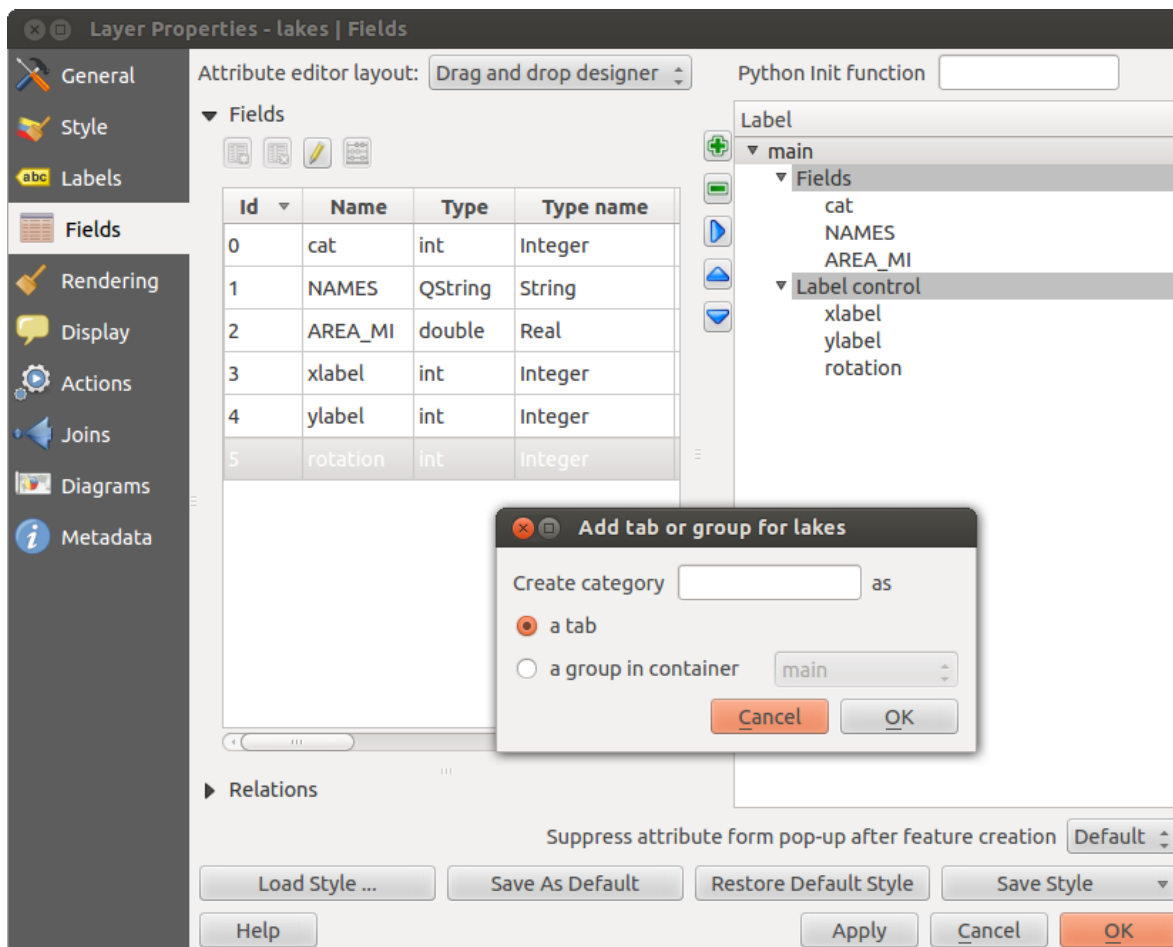
Other options in the dialog are 'Autogenerate' and 'Provide ui-file'. 'Autogenerate' just creates editors for all fields and tabulates them. The 'Provide ui-file' option allows you to use complex dialogs made with the Qt-Designer. Using a UI-file allows a great deal of freedom in creating a dialog. For detailed information, see <http://nathanw.net/2011/09/05/qgis-tips-custom-feature-forms-with-python-logic/>.

QGIS dialogs can have a Python function that is called when the dialog is opened. Use this function to add extra logic to your dialogs. An example is (in module MyForms.py):

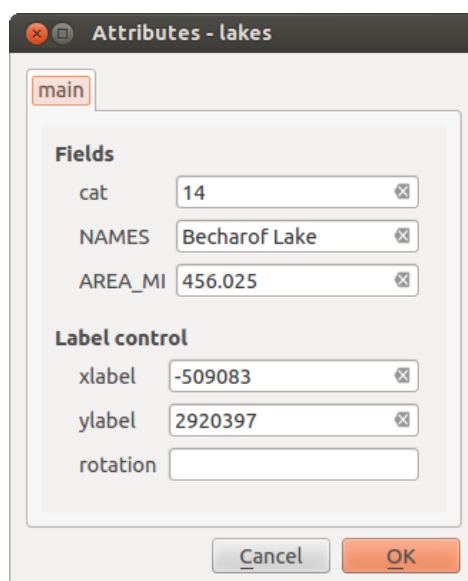
```
def open(dialog, layer, feature):
    geom = feature.geometry()
    control = dialog.findChild(QWidget, "My line edit")
```

Reference in Python Init Function like so: MyForms.open

MyForms.py must live on PYTHONPATH, in .qgis2/python, or inside the project folder.



Gambar 12.27: Dialog to create categories with the **Attribute editor layout**



Gambar 12.28: Resulting built-in form in a data entry session

12.3.4 General Menu



Use this menu to make general settings for the vector layer. There are several options available:

Layer Info

- Change the display name of the layer in *displayed as*
- Define the *Layer source* of the vector layer
- Define the *Data source encoding* to define provider-specific options and to be able to read the file

Coordinate Reference System

- *Specify* the coordinate reference system. Here, you can view or change the projection of the specific vector layer.
- Create a *Spatial Index* (only for OGR-supported formats)
- *Update Extents* information for a layer
- View or change the projection of the specific vector layer, clicking on *Specify ...*

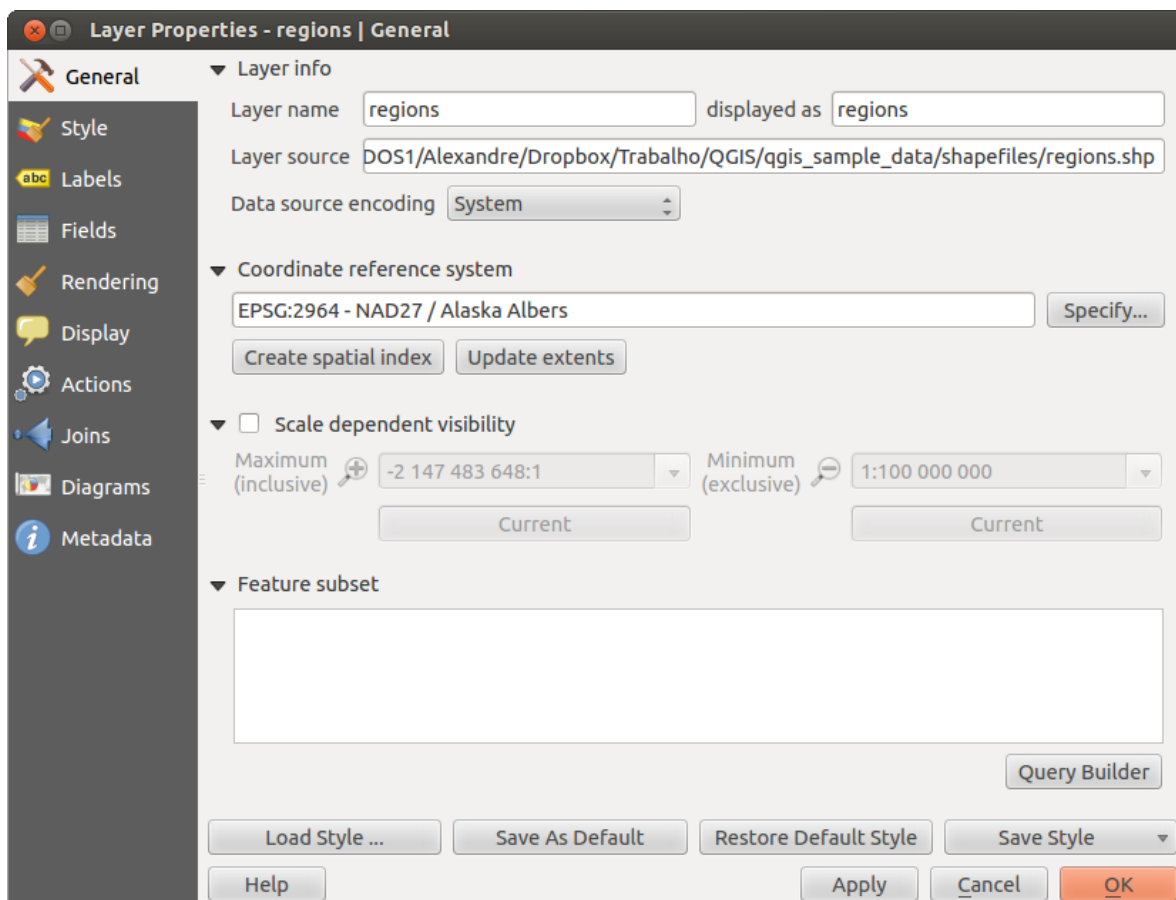



Scale dependent visibility

- You can set the *Maximum (inclusive)* and *Minimum (exclusive)* scale. The scale can also be set by the **[Current]** buttons.


Feature subset

- With the **[Query Builder]** button, you can create a subset of the features in the layer that will be visualized (also refer to section *Query Builder*).




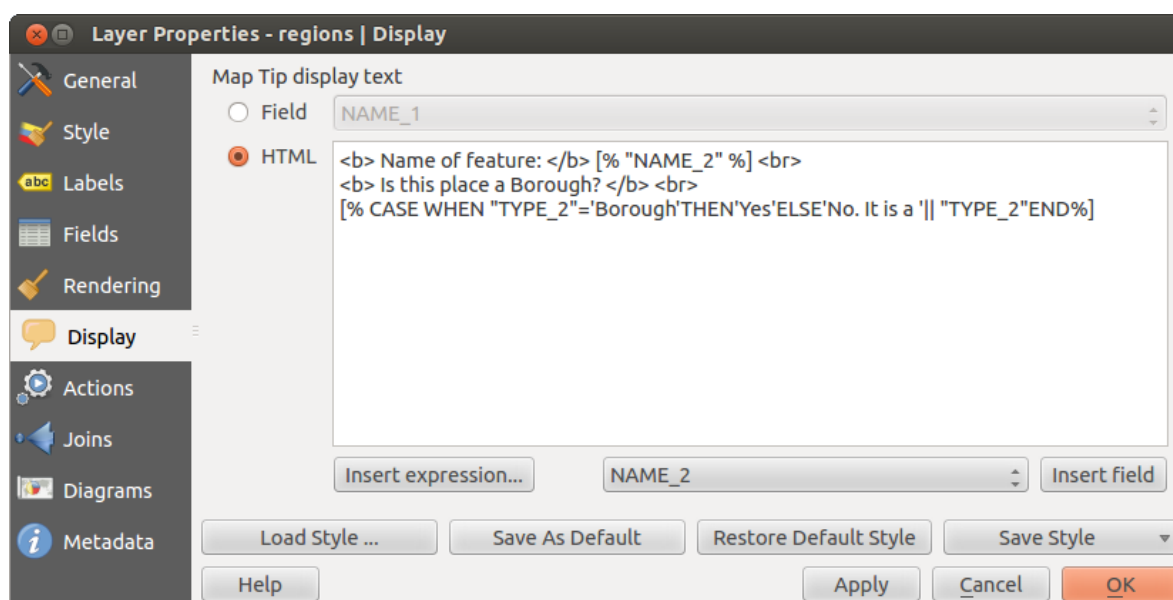
Gambar 12.29: General menu in vector layers properties dialog 

12.3.5 Rendering Menu

QGIS 2.2 introduces 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 new global setting that enables generalisation by default for newly added layers (see section *Opsi*). **Note:** Feature generalisation may introduce artefacts into your rendered output in some cases. These may include slivers between polygons and inaccurate rendering when using offset-based symbol layers.


12.3.6 Display Menu

 This menu is specifically created for Map Tips. It includes a new feature: Map Tip display text in HTML. While you can still choose a *Field* to be displayed when hovering over a feature on the map, it is now possible to insert HTML code that creates a complex display when hovering over a feature. To activate Map Tips, select the menu option *View* → *MapTips*. Figure Display 1 shows an example of HTML code.



Gambar 12.30: HTML code for map tip 

12.3.7 Actions Menu

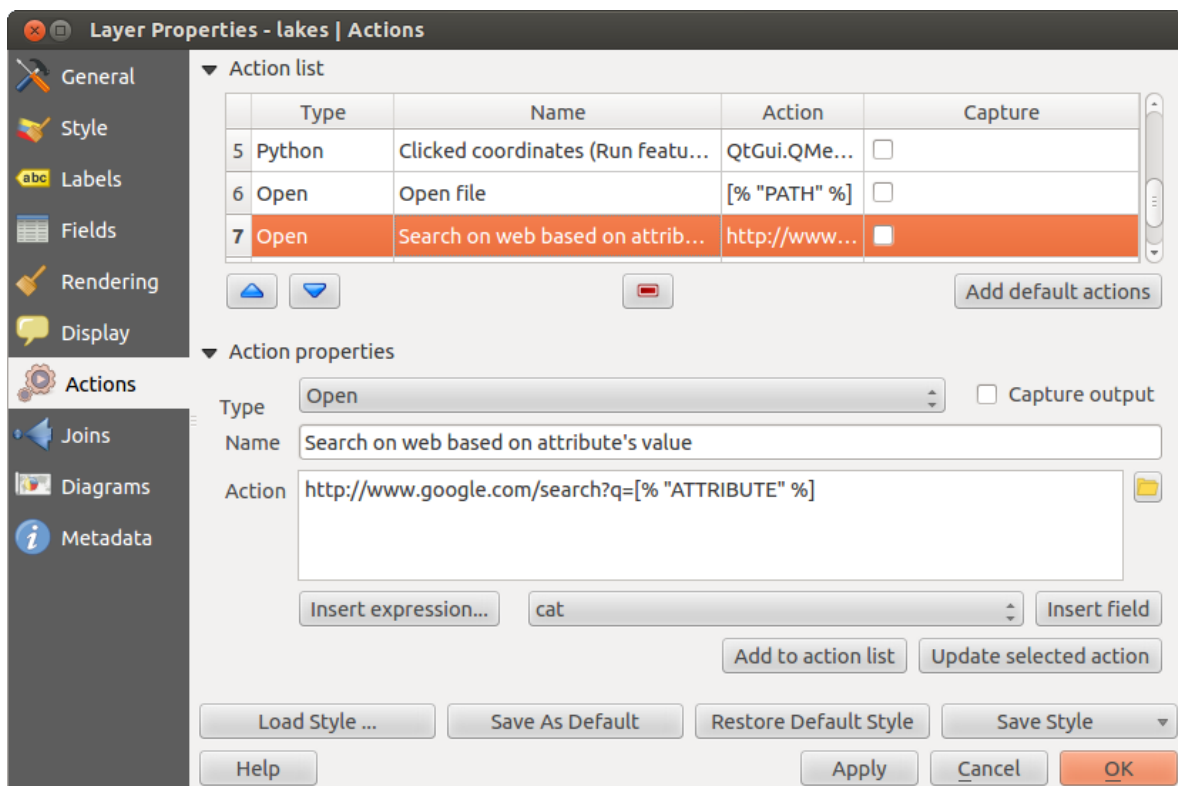
 QGIS provides the ability to perform an action based on the attributes of a feature. This can be used to perform any number of actions, for example, running a program with arguments built from the attributes of a feature or passing parameters to a web reporting tool.

Actions are useful when you frequently want to run an external application or view a web page based on one or more values in your vector layer. They are divided into six types and can be used like this:

- Generic, Mac, Windows and Unix actions start an external process.
- Python actions execute a Python expression.
- Generic and Python actions are visible everywhere.
- Mac, Windows and Unix actions are visible only on the respective platform (i.e., you can define three 'Edit' actions to open an editor and the users can only see and execute the one 'Edit' action for their platform to run the editor).



Gambar 12.31: Map tip made with HTML code 🐧



Gambar 12.32: Overview action dialog with some sample actions 🐧

There are several examples included in the dialog. You can load them by clicking on **[Add default actions]**. One example is performing a search based on an attribute value. This concept is used in the following discussion.

Defining Actions

Attribute actions are defined from the vector *Layer Properties* dialog. To define an action, open the vector *Layer Properties* dialog and click on the *Actions* menu. Go to the *Action properties*. Select 'Generic' as 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.

If you have field names that are substrings of other field names (e.g., coll and coll0), you should indicate that by surrounding the field name (and the % character) with square brackets (e.g., [%coll]). This will prevent the %coll0 field name from being mistaken for the %coll field name with a 0 on the end. The brackets will be removed by QGIS when it substitutes in the value of the field. If you want the substituted field to be surrounded by square brackets, use a second set like this: [[%coll]].

Using the *Identify Features* tool, you can open the *Identify Results* dialog. It includes a (*Derived*) item that contains information relevant to the layer type. The values in this item can be accessed in a similar way to the other fields by preceding the derived field name with (Derived) .. For example, a point layer has an X and Y field, and the values of these fields can be used in the action with %(Derived).X and %(Derived).Y. The derived attributes are only available from the *Identify Results* dialog box, not the *Attribute Table* dialog box.




Two example actions are shown below:

- konqueror http://www.google.com/search?q=%nam
- konqueror http://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 http://www.google.com/search?q=%nam. This will ensure that the konqueror application will be executed when the action is invoked.

The second example uses the %% notation, which does not rely on a particular field for its value. When the action is invoked, the %% will be replaced by the value of the selected field in the identify results or attribute table.

Using Actions

Actions can be invoked from either the *Identify Results* dialog, an *Attribute Table* dialog or from *Run Feature Action* (recall that these dialogs can be opened by clicking  Identify Features or  Open Attribute Table or  Run Feature Action). To invoke an action, right click on the record and choose the action from the pop-up menu. Actions are listed in the popup menu by the name you assigned when defining the action. Click on the action you wish to invoke.

If you are invoking an action that uses the %% notation, right-click on the field value in the *Identify Results* dialog or the *Attribute Table* dialog that you wish to pass to the application or script.

Here is another example that pulls data out of a vector layer and inserts it into a file using bash and the echo command (so it will only work on  or perhaps ). The layer in question has fields for a species name taxon_name, latitude lat and longitude long. We would like to be able to make a spatial selection of localities and export these field values to a text file for the selected record (shown in yellow in the QGIS map area). Here is the action to achieve this:

```
bash -c "echo \"%taxon_name %lat %long\" >> /tmp/species_localities.txt"
```

After selecting a few localities and running the action on each one, opening the output file will show something like this:

```
Acacia mearnsii -34.0800000000 150.0800000000
Acacia mearnsii -34.9000000000 150.1200000000
```

```
Acacia mearnsii -35.2200000000 149.9300000000
Acacia mearnsii -32.2700000000 150.4100000000
```

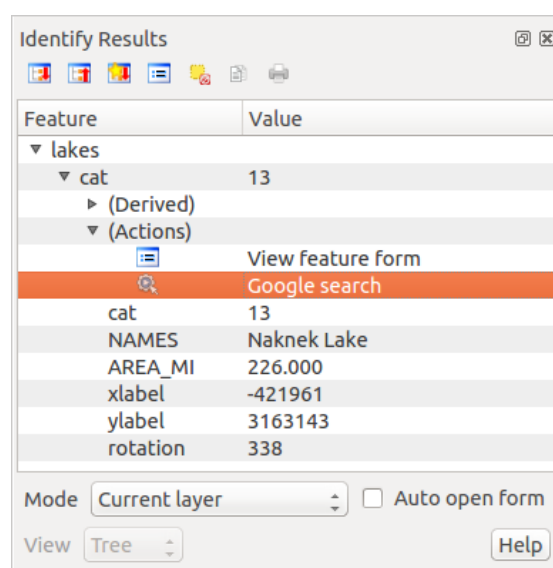
As an exercise, we can create an action that does a Google search on the `lakes` layer. First, we need to determine the URL required to perform a search on a keyword. This is easily done by just going to Google and doing a simple search, then grabbing the URL from the address bar in your browser. From this little effort, we see that the format is `http://google.com/search?q=qgis`, where `QGIS` is the search term. Armed with this information, we can proceed:

1. Make sure the `lakes` layer is loaded.
2. Open the *Layer Properties* dialog by double-clicking on the layer in the legend, or right-click and choose *Properties* from the pop-up menu.
3. Click on the *Actions* menu.
4. Enter a name for the action, for example `Google Search`.
5. For the action, we need to provide the name of the external program to run. In this case, we can use `Firefox`. If the program is not in your path, you need to provide the full path.
6. Following the name of the external application, add the URL used for doing a Google search, up to but not including the search term: `http://google.com/search?q=`
7. The text in the *Action* field should now look like this: `firefox http://google.com/search?q=`
8. Click on the drop-down box containing the field names for the `lakes` layer. It's located just to the left of the **[Insert Field]** button.
9. From the drop-down box, select 'NAMES' and click **[Insert Field]**.
10. Your action text now looks like this:
`firefox http://google.com/search?q=%NAMES`
11. To finalize the action, click the **[Add to action list]** button.

This completes the action, and it is ready to use. The final text of the action should look like this:

```
firefox http://google.com/search?q=%NAMES
```

We can now use the action. Close the *Layer Properties* dialog and zoom in to an area of interest. Make sure the `lakes` layer is active and identify a lake. In the result box you'll now see that our action is visible:



Gambar 12.33: Select feature and choose action 

When we click on the action, it brings up Firefox and navigates to the URL <http://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.

You can define multiple actions for a layer, and each will show up in the *Identify Results* dialog.

There are all kinds of uses for actions. For example, if you have a point layer containing locations of images or photos along with a file name, you could create an action to launch a viewer to display the image. You could also use actions to launch web-based reports for an attribute field or combination of fields, specifying them in the same way we did in our Google search example.

We can also make more complex examples, for instance, using **Python** actions.

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 a shapefile or SpatiaLite)? The following code will do the trick:

```
command = "firefox";
imagerelpath = "images_test/test_image.jpg";
layer = qgis.utils.iface.activeLayer();
import os.path;
layerpath = layer.source() if layer.providerType() == 'ogr'
    else (qgis.core.QgsDataSourceURI(layer.source()).database()
    if layer.providerType() == 'spatialite' else None);
path = os.path.dirname(str(layerpath));
image = os.path.join(path, imagerelpath);
import subprocess;
subprocess.Popen( [command, image ] );
```

We just have to remember that the action is one of type *Python* and the *command* and *imagerelpath* variables must be changed to fit our needs.

But what about if the relative path needs to be relative to the (saved) project file? The code of the Python action would be:

```
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 ] );
```

Another Python action example is the one that allows us to add new layers to the project. For instance, the following examples will add to the project respectively a vector and a raster. The names of the files to be added to the project and the names to be given to the layers are data driven (*filename* and *layername* are column names of the table of attributes of the vector where the action was created):


```
qgis.utils.iface.addVectorLayer('/yourpath/[% "filename" %].shp', '[% "layername" %]',
    'ogr')
```

To add a raster (a TIF image in this example), it becomes:

```
qgis.utils.iface.addRasterLayer('/yourpath/[% "filename" %].tif', '[% "layername" %]
')
```

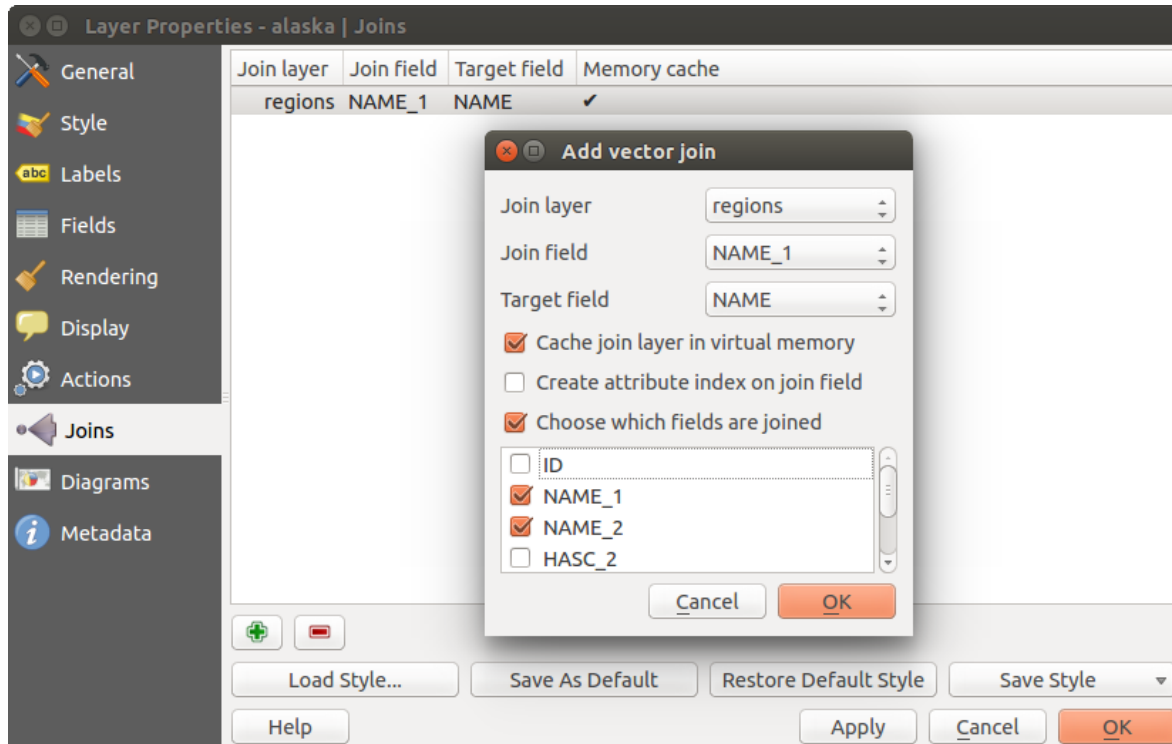
12.3.8 Joins Menu



The *Joins* menu allows you to join a loaded attribute table to a loaded vector layer. After clicking , the *Add vector join* dialog appears. As key columns, you have to define a join layer you want to connect with the

target vector layer. Then, you have to specify the join field that is common to both the join layer and the target layer. Now you can also specify a subset of fields from the joined layer based on the checkbox *Choose which fields are joined*. As a result of the join, all information from the join layer and the target layer are displayed in the attribute table of the target layer as joined information. If you specified a subset of fields only these fields are displayed in the attribute table of the target layer.

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 (see [figure_joins_1](#)).



Gambar 12.34: Join an attribute table to an existing vector layer 🐧

Additionally, the add vector join dialog allows you to:

- *Cache join layer in virtual memory*
- *Create attribute index on the join field*

12.3.9 Diagrams Menu



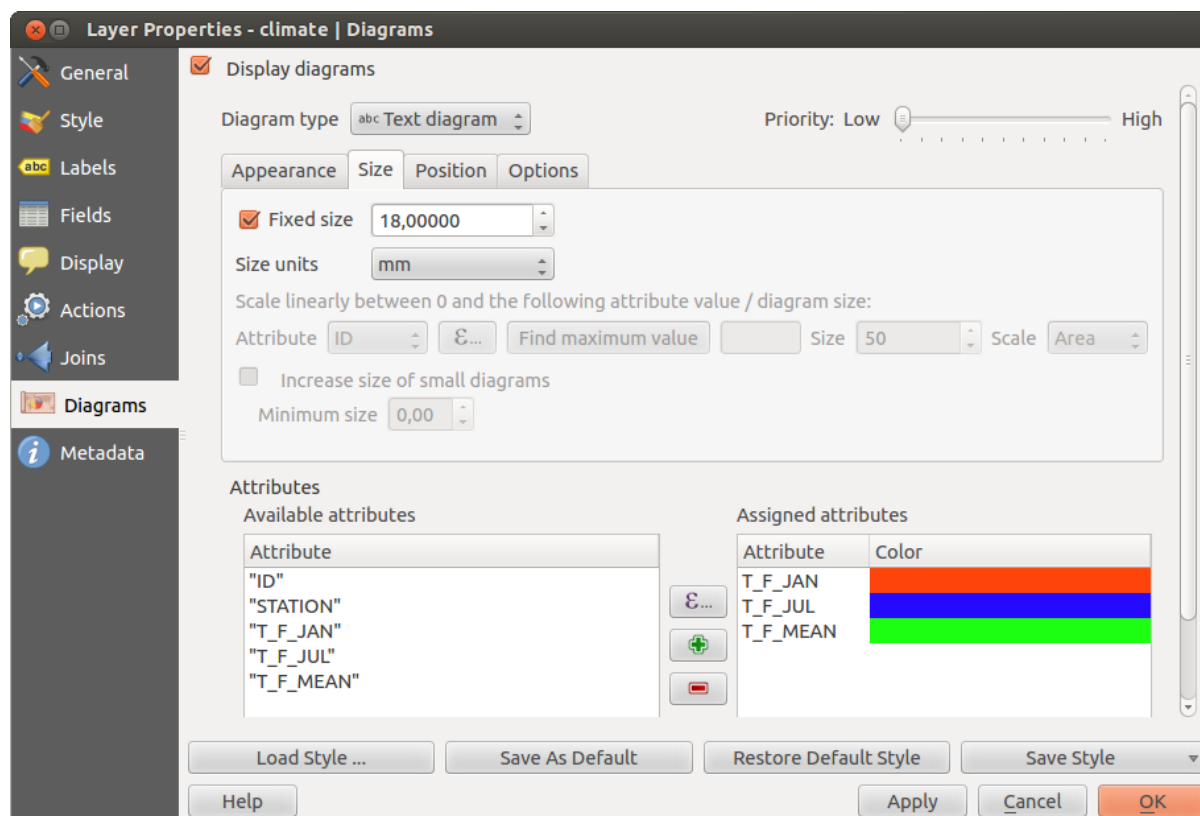
The *Diagrams* menu allows you to add a graphic overlay to a vector layer (see [figure_diagrams_1](#)).

The current core implementation of diagrams provides support for pie charts, text diagrams and histograms.




The menu is divided into four tabs: *Appearance*, *Size*, *Position* and *Options*.

In the cases of the text diagram and pie chart, text values of different data columns are displayed one below the other with a circle or a box and dividers. In the *Size* tab, diagram size is based on a fixed size or on linear scaling according to a classification attribute. The placement of the diagrams, which is done in the *Position* tab, interacts with the new labeling, so position conflicts between diagrams and labels are detected and solved. In addition, chart positions can be fixed manually.

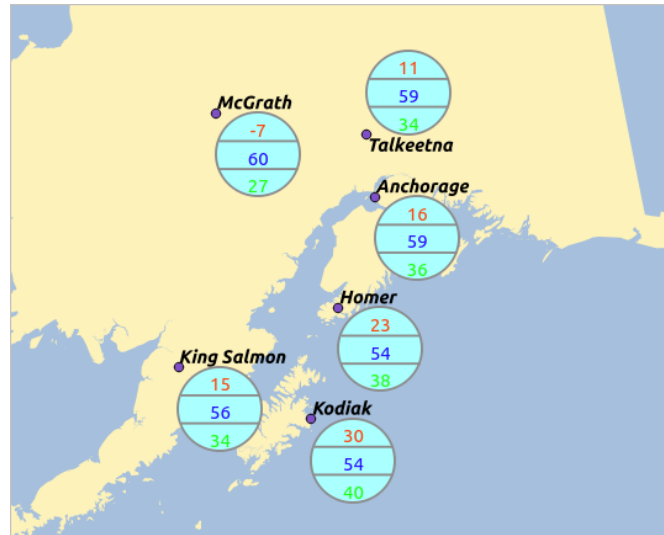
We will demonstrate an example and overlay on the Alaska boundary layer a text diagram showing temperature data from a climate vector layer. Both vector layers are part of the QGIS sample dataset (see section [Contoh data](#)).




Gambar 12.35: Vector properties dialog with diagram menu 

1. First, click on the  Load Vector icon, browse to the QGIS sample dataset folder, and load the two vector shape layers `alaska.shp` and `climate.shp`.
2. Double click the `climate` layer in the map legend to open the *Layer Properties* dialog.
3. Click on the *Diagrams* menu, activate *Display diagrams*, and from the *Diagram type*  combo box, select 'Text diagram'.
4. In the *Appearance* tab, we choose a light blue as background color, and in the *Size* tab, we set a fixed size to 18 mm.
5. In the *Position* tab, placement could be set to 'Around Point'.
6. In the diagram, we want to display the values of the three columns `T_F_JAN`, `T_F_JUL` and `T_F_MEAN`. First select `T_F_JAN` as *Attributes* and click the  button, then `T_F_JUL`, and finally `T_F_MEAN`.
7. Now click [**Apply**] to display the diagram in the QGIS main window.
8. You can adapt the chart size in the *Size* tab. Deactivate the *Fixed size* and set the size of the diagrams on the basis of an attribute with the [**Find maximum value**] button and the *Size* menu. If the diagrams appear too small on the screen, you can activate the *Increase size of small diagrams* checkbox and define the minimum size of the diagrams.
9. Change the attribute colors by double clicking on the color values in the *Assigned attributes* field. [Figure_diagrams_2](#) gives an idea of the result.
10. Finally, click [**Ok**].

Remember that in the *Position* tab, a *Data defined position* of the diagrams is possible. Here, you can use attributes to define the position of the diagram. You can also set a scale-dependent visibility in the *Appearance* tab.



Gambar 12.36: Diagram from temperature data overlaid on a map 

The size and the attributes can also be an expression. Use the  button to add an expression. See [Expressions](#) chapter for more information and example.

12.3.10 Metadata Menu





The *Metadata* menu consists of *Description*, *Attribution*, *MetadataURL* and *Properties* sections.

In the *Properties* section, you get general information about the layer, including specifics about the type and location, number of features, feature type, and editing capabilities. The *Extents* table provides you with layer extent information and the *Layer Spatial Reference System*, which is information about the CRS of the layer. This is a quick way to get information about the layer.

Additionally, 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 catalogue. 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 catalogue. In *MetadataUrl*, you can define the general path to the XML metadata catalogue. This information will be saved in the QGIS project file for subsequent sessions and will be used for QGIS server.

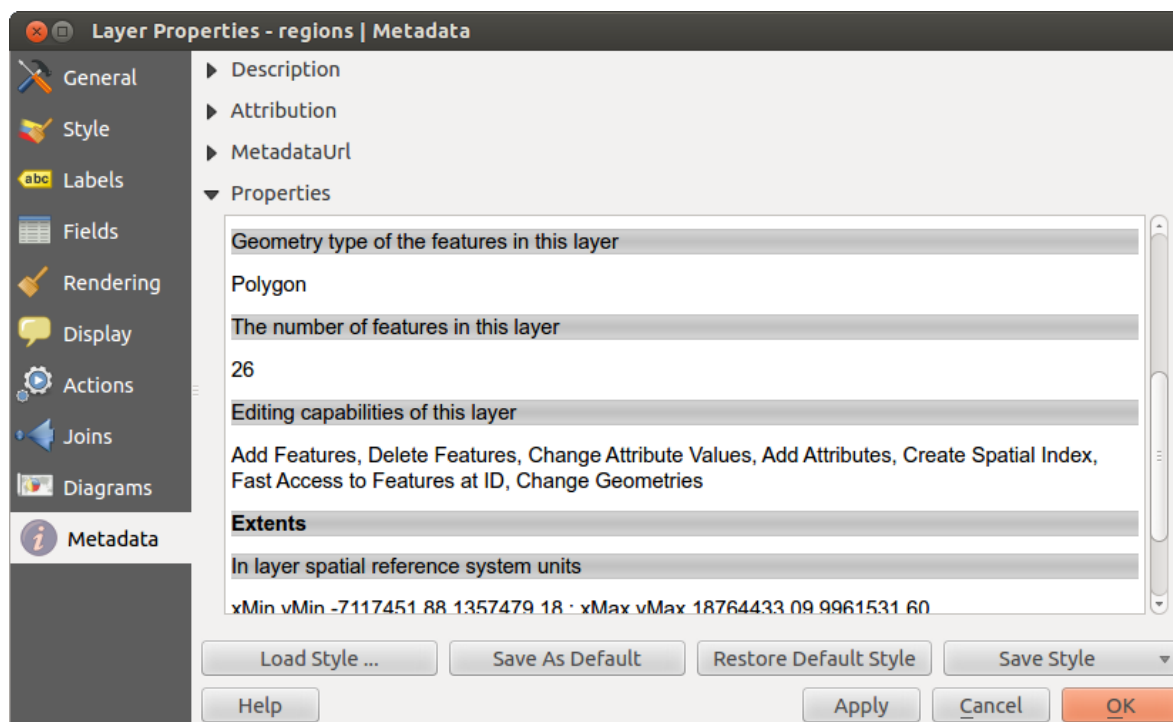
12.4 Expressions


The **Expressions** feature are available through the field calculator or the add a new column button in the attribute table or the Field tab in the Layer properties ; through the graduated, categorized and rule-based rendering in the Style tab of the Layer properties ; through the expression-based labeling  in the  Labeling core application ; through the feature selection and through the diagram tab of the Layer properties.

There are powerful way to manipulate attribute value in order to dynamically change the final value in order to change the geometry style, the content of the label, the value for diagram, select some feature or create virtual column.

12.4.1 Functions List

The **Function List** contains functions as well as fields and values. View the help function in the **Selected Function Help**. In **Expression** you see the calculation expressions you create with the **Function List**. For the most



Gambar 12.37: Metadata menu in vector layers properties dialog 

commonly used operators, see **Operators**.

In the **Function List**, click on *Fields and Values* to view all attributes of the attribute table to be searched. To add an attribute to the Field calculator **Expression** field, double click its name in the *Fields and Values* list. Generally, you can use the various fields, values and functions to construct the calculation expression, or you can just type it into the box. To display the values of a field, you just right click on the appropriate field. You can choose between *Load top 10 unique values* and *Load all unique values*. On the right side, the **Field Values** list opens with the unique values. To add a value to the Field calculator **Expression** box, double click its name in the **Field Values** list.

The *Operators*, *Math*, *Conversions*, *String*, *Geometry* and *Record* groups provide several functions. In *Operators*, you find mathematical operators. Look in *Math* for mathematical functions. The *Conversions* group contains functions that convert one data type to another. The *String* group provides functions for data strings. In the *Geometry* group, you find functions for geometry objects. With *Record* group functions, you can add a numeration to your data set. To add a function to the Field calculator **Expression** box, click on the > and then double click the function.

Operators

This group contains operators (e.g., +, -, *).

a + b	a plus b
a - b	a minus b
a * b	a multiplied by b
a / b	a divided by b
a % b	a modulo b (for example, 7 % 2 = 1, or 2 fits into 7 three times with remainder 1)
a ^ b	a power b (for example, 2^2=4 or 2^3=8)
a = b	a and b are equal
a > b	a is larger than b
a < b	a is smaller than b
a <> b	a and b are not equal
a != b	a and b are not equal

a <= b	a is less than or equal to b
a >= b	a is larger than or equal to b
a ~ b	a matches the regular expression b
+ a	positive sign
- a	negative value of a
	joins two values together into a string 'Hello' ' world'
LIKE	returns 1 if the string matches the supplied pattern
ILIKE	returns 1 if the string matches case-insensitive the supplied pattern (ILIKE can be used instead of LIKE to make the match case-insensitive)
IS	returns 1 if a is the same as b
OR	returns 1 when condition a or b is true
AND	returns 1 when condition a and b are true
NOT	returns 1 if a is not the same as b
column name "column name"	value of the field column name, take care to not be confused with simple quote, see below
'string'	a string value, take care to not be confused with double quote, see above
NULL	null value
a IS NULL	a has no value
a IS NOT NULL	a has a value
a IN (value[,value])	a is below the values listed
a NOT IN (value[,value])	a is not below the values listed

Some example:

- Joins a string and a value from a column name:

```
'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%'
```

Conditionals

This group contains functions to handle conditional checks in expressions.

CASE	evaluates multiple expressions and returns a result
CASE ELSE	evaluates multiple expressions and returns a result
coalesce	returns the first non-NULL value from the expression list
regexp_match	returns true if any part of a string matches the supplied regular expression

Some example:

- Send back a value if the first condition is true, else another value:

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

Mathematical Functions

This group contains math functions (e.g., square root, sin and cos).

sqrt(a)	square root of a
abs	returns the absolute value of a number
sin(a)	sine of a

<code>cos(a)</code>	cosine of a
<code>tan(a)</code>	tangent of a
<code>asin(a)</code>	arcsin of a
<code>acos(a)</code>	arccos of a
<code>atan(a)</code>	arctan of a
<code>atan2(y,x)</code>	arctan of y/x using the signs of the two arguments to determine the quadrant of the result
<code>exp</code>	exponential of a value
<code>ln</code>	value of the natural logarithm of the passed expression
<code>log10</code>	value of the base 10 logarithm of the passed expression
<code>log</code>	value of the logarithm of the passed value and base
<code>round</code>	round to number of decimal places
<code>rand</code>	random integer within the range specified by the minimum and maximum argument (inclusive)
<code>randf</code>	random float within the range specified by the minimum and maximum argument (inclusive)
<code>max</code>	largest value in a set of values
<code>min</code>	smallest value in a set of values
<code>clamp</code>	restricts an input value to a specified range
<code>scale_linear</code>	transforms a given value from an input domain to an output range using linear interpolation
<code>scale_exp</code>	transforms a given value from an input domain to an output range using an exponential curve
<code>floor</code>	rounds a number downwards
<code>ceil</code>	rounds a number upwards
<code>\$pi</code>	pi as value for calculations

Conversions

This group contains functions to convert one data type to another (e.g., string to integer, integer to string).

<code>toint</code>	converts a string to integer number
<code>toreal</code>	converts a string to real number
<code>tostring</code>	converts number to string
<code>todatetime</code>	converts a string into Qt data time type
<code>todate</code>	converts a string into Qt data type
<code>totime</code>	converts a string into Qt time type
<code>tointerval</code>	converts a string to an interval type (can be used to take days, hours, months, etc. off a date)

Date and Time Functions

This group contains functions for handling date and time data.

<code>\$now</code>	current date and time
<code>age</code>	difference between two dates
<code>year</code>	extract the year part from a date, or the number of years from an interval
<code>month</code>	extract the month part from a date, or the number of months from an interval

week	extract the week number from a date, or the number of weeks from an interval
day	extract the day from a date, or the number of days from an interval
hour	extract the hour from a datetime or time, or the number of hours from an interval
minute	extract the minute from a datetime or time, or the number of minutes from an interval
second	extract the second from a datetime or time, or the number of minutes from an interval

Some example:

- Get the month and the year of today in the format “10/2014”

```
month($now) || '/' || year($now)
```

String Functions

This group contains functions that operate on strings (e.g., that replace, convert to upper case).

lower	convert string a to lower case
upper	convert string a to upper case
title	converts all words of a string to title case (all words lower case with leading capital letter)
trim	removes all leading and trailing white space (spaces, tabs, etc.) from a string
wordwrap	returns a string wrapped to a maximum/minimum number of characters
length	length of string a
replace	returns a string with the supplied string replaced
regexp_replace(a,this,that)	returns a string with the supplied regular expression replaced
regexp_substr	returns the portion of a string which matches a supplied regular expression
substr(*a*,from,len)	returns a part of a string
concat	concatenates several strings to one
strpos	returns the index of a regular expression in a string
left	returns a substring that contains the n leftmost characters of the string
right	returns a substring that contains the n rightmost characters of the string
rpad	returns a string with supplied width padded using the fill character
lpad	returns a string with supplied width padded using the fill character
format	formats a string using supplied arguments
format_number	returns a number formatted with the locale separator for thousands (also truncates the number to the number of supplied places)
format_date	formats a date type or string into a custom string format

Color Functions

This group contains functions for manipulating colors.

<code>color_rgb</code>	returns a string representation of a color based on its red, green, and blue components
<code>color_rgba</code>	returns a string representation of a color based on its red, green, blue, and alpha (transparency) components
<code>ramp_color</code>	returns a string representing a color from a color ramp
<code>color_hsl</code>	returns a string representation of a color based on its hue, saturation, and lightness attributes
<code>color_hsla</code>	returns a string representation of a color based on its hue, saturation, lightness and alpha (transparency) attributes
<code>color_hsv</code>	returns a string representation of a color based on its hue, saturation, and value attributes
<code>color_hsva</code>	returns a string representation of a color based on its hue, saturation, value and alpha (transparency) attributes
<code>color_cmyk</code>	returns a string representation of a color based on its cyan, magenta, yellow and black components
<code>color_cmyka</code>	returns a string representation of a color based on its cyan, magenta, yellow, black and alpha (transparency) components

Geometry Functions

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

<code>\$geometry</code>	returns the geometry of the current feature (can be used for processing with other functions)
<code>\$area</code>	returns the area size of the current feature
<code>\$length</code>	returns the length size of the current feature
<code>\$perimeter</code>	returns the perimeter length of the current feature
<code>\$x</code>	returns the x coordinate of the current feature
<code>\$y</code>	returns the y coordinate of the current feature
<code>xat</code>	retrieves the nth x coordinate of the current feature. n given as a parameter of the function
<code>yat</code>	retrieves the nth y coordinate of the current feature. n given as a parameter of the function
<code>xmin</code>	returns the minimum x coordinate of a geometry. Calculations are in the Spatial Reference System of this Geometry
<code>xmax</code>	returns the maximum x coordinate of a geometry. Calculations are in the Spatial Reference System of this Geometry
<code>ymin</code>	returns the minimum y coordinate of a geometry. Calculations are in the Spatial Reference System of this Geometry
<code>ymax</code>	returns the maximum y coordinate of a geometry. Calculations are in the Spatial Reference System of this Geometry
<code>geomFromWKT</code>	returns a geometry created from a well-known text (WKT) representation
<code>geomFromGML</code>	returns a geometry from a GML representation of geometry
<code>bbox</code>	
<code>disjoint</code>	returns 1 if the geometries do not share any space together
<code>intersects</code>	returns 1 if the geometries spatially intersect (share any portion of space) and 0 if they don't
<code>touches</code>	returns 1 if the geometries have at least one point in common, but their interiors do not intersect
<code>crosses</code>	returns 1 if the supplied geometries have some, but not all, interior points in common
<code>contains</code>	returns 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

overlaps	returns 1 if the geometries share space, are of the same dimension, but are not completely contained by each other
within	returns 1 if geometry a is completely inside geometry b
buffer	returns a geometry that represents all points whose distance from this geometry is less than or equal to distance
centroid	returns the geometric center of a geometry
bounds	returns a geometry which represents the bounding box of an input geometry. Calculations are in the Spatial Reference System of this Geometry.
bounds_width	returns the width of the bounding box of a geometry. Calculations are in the Spatial Reference System of this Geometry.
bounds_height	returns the height of the bounding box of a geometry. Calculations are in the Spatial Reference System of this Geometry.
convexHull	returns the convex hull of a geometry (this represents the minimum convex geometry that encloses all geometries within the set)
difference	returns a geometry that represents that part of geometry a that does not intersect with geometry b
distance	returns the minimum distance (based on spatial ref) between two geometries in projected units
intersection	returns a geometry that represents the shared portion of geometry a and geometry b
symDifference	returns a geometry that represents the portions of a and b that do not intersect
combine	returns the combination of geometry a and geometry b
union	returns a geometry that represents the point set union of the geometries
geomToWKT	returns the well-known text (WKT) representation of the geometry without SRID metadata

Record Functions

This group contains functions that operate on record identifiers.

\$rownum	returns the number of the current row
\$id	returns the feature id of the current row
\$currentfeature	returns the current feature being evaluated. This can be used with the 'attribute' function to evaluate attribute values from the current feature.
\$scale	returns the current scale of the map canvas
\$uuid	generates a Universally Unique Identifier (UUID) for each row. Each UUID is 38 characters long.
getFeature	returns the first feature of a layer matching a given attribute value.
attribute	returns the value of a specified attribute from a feature.
\$map	returns the id of the current map item if the map is being drawn in a composition, or "canvas" if the map is being drawn within the main QGIS window.

Fields and Values

Contains a list of fields from the layer. Sample values can also be accessed via right-click.

Select the field name from the list, then right-click to access a context menu with options to load sample values from the selected field.

Fields name should be double-quoted. Values or string should be simple-quoted.

12.5 Editing

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

Catatan: The procedure for editing GRASS layers is different - see section *Digitizing and editing a GRASS vector layer* for details.

Tip: Concurrent Edits




This version of QGIS does not track if somebody else is editing a feature at the same time as you are. The last person to save their edits wins.

12.5.1 Setting the Snapping Tolerance and Search Radius

Before we can edit vertices, we must set the snapping tolerance and search radius to a value that allows us an optimal editing of the vector layer geometries.

Snapping tolerance

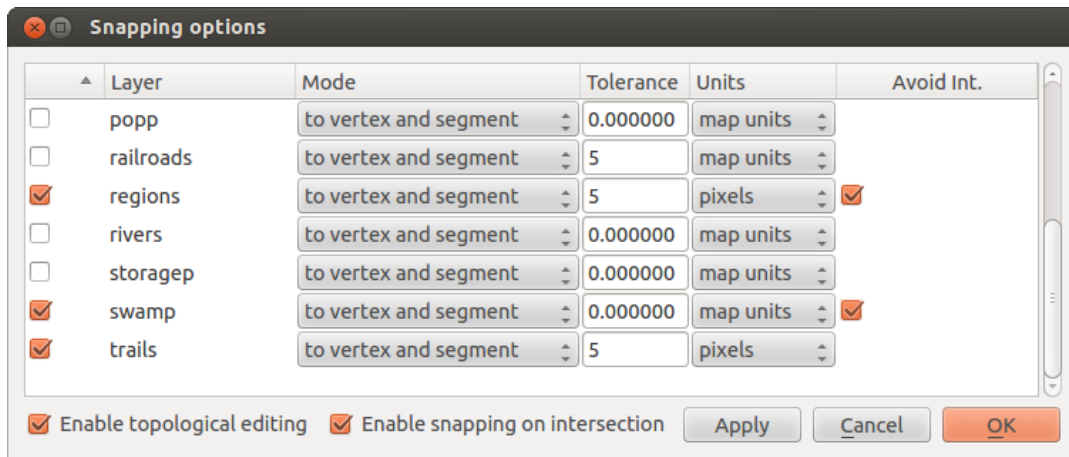
Snapping tolerance is the distance QGIS uses to search for the closest vertex and/or segment you are trying to connect to when you set a new vertex or move an existing vertex. If you aren't within the snapping tolerance, QGIS will leave the vertex where you release the mouse button, instead of snapping it to an existing vertex and/or segment. The snapping tolerance setting affects all tools that work with tolerance.

1. A general, project-wide snapping tolerance can be defined by choosing *Settings* →  *Options*. On Mac, go to *QGIS* →  *Preferences...*. On Linux: *Edit* →  *Options*. In the *Digitizing* tab, you can select between 'to vertex', 'to segment' or 'to vertex and segment' as default snap mode. You can also define a default snapping tolerance and a search radius for vertex edits. The tolerance can be set either in map units or in pixels. The advantage of choosing pixels is that the snapping tolerance doesn't have to be changed after zoom operations. In our small digitizing project (working with the Alaska dataset), we define the snapping units in feet. Your results may vary, but something on the order of 300 ft at a scale of 1:10000 should be a reasonable setting.
2. A layer-based snapping tolerance can be defined by choosing *Settings* → (or *File* →) *Snapping options...* to enable and adjust snapping mode and tolerance on a layer basis (see [figure_edit_1](#)).

Note that this layer-based snapping overrides the global snapping option set in the *Digitizing* tab. So, if you need to edit one layer and snap its vertices to another layer, then enable snapping only on the `snap to` layer, then decrease the global snapping tolerance to a smaller value. Furthermore, snapping will never occur to a layer that is not checked in the snapping options dialog, regardless of the global snapping tolerance. So be sure to mark the checkbox for those layers that you need to snap to.

Search radius

Search radius is the distance QGIS uses to search for the closest vertex you are trying to move when you click on the map. If you aren't within the search radius, QGIS won't find and select any vertex for editing, and it will pop up an annoying warning to that effect. Snap tolerance and search radius are set in map units or pixels, so you may find you need to experiment to get them set right. If you specify too big of a tolerance, QGIS may snap to the






Gambar 12.38: Edit snapping options on a layer basis 🐧

wrong vertex, especially if you are dealing with a large number of vertices in close proximity. Set search radius too small, and it won't find anything to move.

The search radius for vertex edits in layer units can be defined in the *Digitizing* tab under *Settings* → *Options*. This is the same place where you define the general, project- wide snapping tolerance.

12.5.2 Zooming and Panning

Before editing a layer, you should zoom in to your area of interest. This avoids waiting while all the vertex markers are rendered across the entire layer.

Apart from using the  pan and  zoom-in /  zoom-out icons on the toolbar with the mouse, navigating can also be done with the mouse wheel, spacebar and the arrow keys.

Zooming and panning with the mouse wheel

While digitizing, you can press the mouse wheel to pan inside of the main window, and you can roll the mouse wheel to zoom in and out on the map. For zooming, place the mouse cursor inside the map area and roll it forward (away from you) to zoom in and backwards (towards you) to zoom out. The mouse cursor position will be the center of the zoomed area of interest. You can customize the behavior of the mouse wheel zoom using the *Map tools* tab under the *Settings* → *Options* menu.

Panning with the arrow keys


Panning the map during digitizing is possible with the arrow keys. Place the mouse cursor inside the map area, and click on the right arrow key to pan east, left arrow key to pan west, up arrow key to pan north, and down arrow key to pan south.

You can also use the space bar to temporarily cause mouse movements to pan the map. The PgUp and PgDown keys on your keyboard will cause the map display to zoom in or out without interrupting your digitizing session.


12.5.3 Topological editing

Besides layer-based snapping options, you can also define topological functionalities in the *Snapping options...* dialog in the *Settings* (or *File*) menu. Here, you can define *Enable topological editing*, and/or for polygon layers, you can activate the column *Avoid Int.*, which avoids intersection of new polygons.


Enable topological editing

The option  *Enable topological editing* is for editing and maintaining common boundaries in polygon mosaics. QGIS ‘detects’ a shared boundary in a polygon mosaic, so you only have to move the vertex once, and QGIS will take care of updating the other boundary.

Avoid intersections of new polygons

The second topological option in the  *Avoid Int.* column, called *Avoid intersections of new polygons*, avoids overlaps in polygon mosaics. It is for quicker digitizing of adjacent polygons. If you already have one polygon, it is possible with this option to digitize the second one such that both intersect, and QGIS then cuts the second polygon to the common boundary. The advantage is that you don’t have to digitize all vertices of the common boundary.

Enable snapping on intersections

Another option is to use  *Enable snapping on intersection*. It allows you to snap on an intersection of background layers, even if there’s no vertex on the intersection.

12.5.4 Digitizing an existing layer

By default, QGIS loads layers read-only. This is a safeguard to avoid accidentally editing a layer if there is a slip of the mouse. However, you can choose to edit any layer as long as the data provider supports it, and the underlying data source is writable (i.e., its files are not read-only).

In general, tools for editing vector layers are divided into a digitizing and an advanced digitizing toolbar, described in section *Advanced digitizing*. You can select and unselect both under *View* → *Toolbars* →. Using the basic digitizing tools, you can perform the following functions:
















Icon	Purpose	Icon	Purpose
	Current edits		Toggle editing
	Adding Features: Capture Point		Adding Features: Capture Line
	Adding Features: Capture Polygon		Move Feature
	Node Tool		Delete Selected
	Cut Features		Copy Features
	Paste Features		Save layer edits

Table Editing: Vector layer basic editing toolbar

All editing sessions start by choosing the  *Toggle editing* option. This can be found in the context menu after right clicking on the legend entry for a given layer.

Alternatively, you can use the *Toggle Editing*  *Toggle editing* button from the digitizing toolbar to start or stop the editing mode. Once the layer is in edit mode, markers will appear at the vertices, and additional tool buttons on the editing toolbar will become available.

Tip: Save Regularly

Remember to  *Save Layer Edits* regularly. This will also check that your data source can accept all the changes.

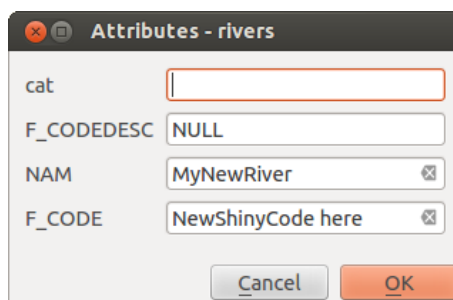
Adding Features

You can use the  Add Feature,  Add Feature or  Add Feature icons on the toolbar to put the QGIS cursor into digitizing mode.


For each feature, you first digitize the geometry, then enter its attributes. To digitize the geometry, left-click on the map area to create the first point of your new feature.

For lines and polygons, keep on left-clicking for each additional point you wish to capture. When you have finished adding points, right-click anywhere on the map area to confirm you have finished entering the geometry of that feature.

The attribute window will appear, allowing you to enter the information for the new feature. [Figure_edit_2](#) shows setting attributes for a fictitious new river in Alaska. In the *Digitizing* menu under the *Settings* → *Options* menu, you can also activate *Suppress attributes pop-up windows after each created feature* and *Reuse last entered attribute values*.






Gambar 12.39: Enter Attribute Values Dialog after digitizing a new vector feature 

With the  Move Feature(s) icon on the toolbar, you can move existing features.

Tip: Attribute Value Types


For editing, the attribute types are validated during entry. Because of this, it is not possible to enter a number into a text column in the dialog *Enter Attribute Values* or vice versa. If you need to do so, you should edit the attributes in a second step within the *Attribute table* dialog.


Current Edits

This feature allows the digitization of multiple layers. Choose  *Save for Selected Layers* to save all changes you made in multiple layers. You also have the opportunity to  *Rollback for Selected Layers*, so that the digitization may be withdrawn for all selected layers. If you want to stop editing the selected layers,  *Cancel for Selected Layer(s)* is an easy way.


The same functions are available for editing all layers of the project.

Node Tool


For shapefile-based layers as well as SpatialLite, PostgreSQL/PostGIS, MSSQL Spatial, and Oracle Spatial tables, the  Node Tool 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 node tool also works with 'on the fly' projection turned on, and it supports the topological editing feature. This tool is, unlike other tools in QGIS, persistent, so when some operation is done, selection stays active for this feature and tool. If the node tool is unable to find any features, a warning will be displayed.



It is important to set the property *Settings* →  *Options* → *Digitizing* → *Search Radius*: to a number greater than zero (i.e., 10). Otherwise, QGIS will not be able to tell which vertex is being edited.

Tip: Vertex Markers

The current version of QGIS supports three kinds of vertex markers: ‘Semi-transparent circle’, ‘Cross’ and ‘None’. To change the marker style, choose  *Options* from the *Settings* menu, click on the *Digitizing* tab and select the appropriate entry.


Basic operations

Start by activating the  *Node Tool* and selecting a feature by clicking on it. Red boxes will appear at each vertex of this feature.

- **Selecting vertices:** You can select vertices by clicking on them one at a time, by clicking on an edge to select the vertices at both ends, or by clicking and dragging a rectangle around some vertices. When a vertex is selected, its color changes to blue. To add more vertices to the current selection, hold down the `Ctrl` key while clicking. Hold down `Ctrl` or `Shift` when clicking to toggle the selection state of vertices (vertices that are currently unselected will be selected as usual, but also vertices that are already selected will become unselected).
- **Adding vertices:** To add a vertex, simply double click near an edge and a new vertex will appear on the edge near to the cursor. Note that the vertex will appear on the edge, not at the cursor position; therefore, it should be moved if necessary.
- **Deleting vertices:** After selecting vertices for deletion, click the `Delete` key. Note that you cannot use the  *Node Tool* to delete a complete feature; QGIS will ensure it retains the minimum number of vertices for the feature type you are working on. To delete a complete feature use the  *Delete Selected* tool.
- **Moving vertices:** Select all the vertices you want to move. Click on a selected vertex or edge and drag in the direction you wish to move. All the selected vertices will move together. If snapping is enabled, the whole selection can jump to the nearest vertex or line.

Each change made with the node tool 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 node tool provides tooltips to identify a vertex by hovering the pointer over it.

Cutting, Copying and Pasting Features




Selected features can be cut, copied and pasted between layers in the same QGIS project, as long as destination layers are set to  *Toggle editing* beforehand.

Features can also be pasted to external applications as text. That is, the features are represented in CSV format, with the geometry data appearing in the OGC Well-Known Text (WKT) format.

However, in this version of QGIS, text features from outside QGIS cannot be pasted to a layer within QGIS. When would the copy and paste function come in handy? Well, it turns out that you can edit more than one layer at a time and copy/paste features between layers. Why would we want to do this? Say we need to do some work on a new layer but only need one or two lakes, not the 5,000 on our `big_lakes` layer. We can create a new layer and use copy/paste to plop the needed lakes into it.

As an example, we will copy some lakes to a new layer:

1. Load the layer you want to copy from (source layer)
2. Load or create the layer you want to copy to (target layer)
3. Start editing for target layer
4. Make the source layer active by clicking on it in the legend



5. Use the  Select Single Feature tool to select the feature(s) on the source layer
6. Click on the  Copy Features tool
7. Make the destination layer active by clicking on it in the legend
8. Click on the  Paste Features tool
9. Stop editing and save the changes



What happens if the source and target layers have different schemas (field names and types are not the same)? QGIS populates what matches and ignores the rest. If you don't care about the attributes being copied to the target layer, it doesn't matter how you design the fields and data types. If you want to make sure everything - the feature and its attributes - gets copied, make sure the schemas match.

Tip: Congruency of Pasted Features



If your source and destination layers use the same projection, then the pasted features will have geometry identical to the source layer. However, if the destination layer is a different projection, then QGIS cannot guarantee the geometry is identical. This is simply because there are small rounding-off errors involved when converting between projections.

Deleting Selected Features

If we want to delete an entire polygon, we can do that by first selecting the polygon using the regular  Select Single Feature tool. You can select multiple features for deletion. Once you have the selection set, use the  Delete Selected tool to delete the features.

The  Cut Features tool on the digitizing toolbar can also be used to delete features. This effectively deletes the feature but also places it on a “spatial clipboard”. So, we cut the feature to delete. We could then use the  Paste Features tool to put it back, giving us a one-level undo capability. Cut, copy, and paste work on the currently selected features, meaning we can operate on more than one at a time.

Saving Edited Layers

When a layer is in editing mode, any changes remain in the memory of QGIS. Therefore, they are not committed/saved immediately to the data source or disk. If you want to save edits to the current layer but want to continue editing without leaving the editing mode, you can click the  Save Layer Edits button. When you turn editing mode off with  Toggle editing (or quit QGIS for that matter), you are also asked if you want to save your changes or discard them.

If the changes cannot be saved (e.g., disk full, or the attributes have values that are out of range), the QGIS in-memory state is preserved. This allows you to adjust your edits and try again.

Tip: Data Integrity

It is always a good idea to back up your data source before you start editing. While the authors of QGIS have made every effort to preserve the integrity of your data, we offer no warranty in this regard.

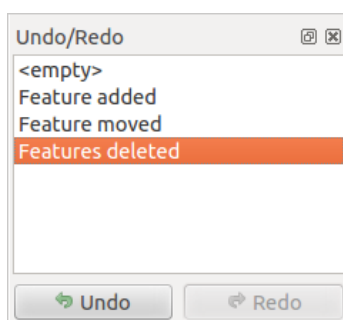
12.5.5 Advanced digitizing

Icon	Purpose	Icon	Purpose
	Undo		Redo
	Rotate Feature(s)		Simplify Feature
	Add Ring		Add Part
	Fill Ring		Delete Ring
	Delete Part		Reshape Features
	Offset Curve		Split Features
	Split Parts		Merge Selected Features
	Merge Attributes of Selected Features		Rotate Point Symbols

Table Advanced Editing: Vector layer advanced editing toolbar

Undo and Redo

The Undo and Redo tools allows you to undo or redo vector editing operations. There is also a dockable widget, which shows all operations in the undo/redo history (see [Figure_edit_3](#)). This widget is not displayed by default; it can be displayed by right clicking on the toolbar and activating the Undo/Redo checkbox. Undo/Redo is however active, even if the widget is not displayed.



Gambar 12.40: Redo and Undo digitizing steps

When Undo is hit, the state of all features and attributes are reverted to the state before the reverted operation happened. Changes other than normal vector editing operations (for example, changes done by a plugin), may or may not be reverted, depending on how the changes were performed.

To use the undo/redo history widget, simply click to select an operation in the history list. All features will be reverted to the state they were in after the selected operation.


Rotate Feature(s)

Use Rotate Feature(s) to rotate one or multiple selected features in the map canvas. You first need to select the features and then press the Rotate Feature(s) icon. The centroid of the feature(s) appears and will be the rotation anchor point. If you selected multiple features, the rotation anchor point will be the common center of the features. Press and drag the left mouse button in the desired direction to rotate the selected features.


It's also possible to create a user-defined rotation anchor point around which the selected feature will rotate. Select the features to rotate and activate the Rotate Feature(s) tool. Press and hold the `Ctrl` button and move the mouse

pointer (without pressing the mouse button) to the place where you want the rotation anchor to be moved. Release the `Ctrl` button when the desired rotation anchor point is reached. Now, press and drag the left mouse button in the desired direction to rotate the selected feature(s).


Simplify Feature

The  Simplify Feature tool allows you to reduce the number of vertices of a feature, as long as the geometry doesn't change and geometry type is not a multi geometry. First, select a feature. It will be highlighted by a red rubber band and a slider will appear. Moving the slider, the red rubber band will change its shape to show how the feature is being simplified. Click **[OK]** to store the new, simplified geometry. If a feature cannot be simplified (e.g. multi-polygons), a message will appear.

Add Ring



You can create ring polygons using the  Add Ring icon in the toolbar. This means that inside an existing area, it is possible to digitize further polygons that will occur as a 'hole', so only the area between the boundaries of the outer and inner polygons remains as a ring polygon.

Add Part


You can  add part polygons to a selected multipolygon. The new part polygon must be digitized outside the selected multi-polygon.

Fill Ring


You can use the  Fill Ring function to add a ring to a polygon and add a new feature to the layer at the same time.

Thus you need not first use the  Add Ring icon and then the  Add feature function anymore.


Delete Ring

The  Delete Ring tool allows you to delete ring polygons inside an existing area. This tool only works with polygon layers. It doesn't change anything when it is used on the outer ring of the polygon. This tool can be used on polygon and multi-polygon features. Before you select the vertices of a ring, adjust the vertex edit tolerance.

Delete Part

The  Delete Part tool allows you to delete parts from multifeatures (e.g., to delete polygons from a multi-polygon feature). It won't delete the last part of the feature; this last part will stay untouched. This tool works with all multi-part geometries: point, line and polygon. Before you select the vertices of a part, adjust the vertex edit tolerance.

Reshape Features


You can reshape line and polygon features using the  Reshape Features icon on the toolbar. It replaces the line or polygon part from the first to the last intersection with the original line. With polygons, this can sometimes lead



to unintended results. It is mainly useful to replace smaller parts of a polygon, not for major overhauls, and the reshape line is not allowed to cross several polygon rings, as this would generate an invalid polygon.

For example, you can edit the boundary of a polygon with this tool. First, click in the inner area of the polygon next to the point where you want to add a new vertex. Then, cross the boundary and add the vertices outside the polygon. To finish, right-click in the inner area of the polygon. The tool will automatically add a node where the new line crosses the border. It is also possible to remove part of the area from the polygon, starting the new line outside the polygon, adding vertices inside, and ending the line outside the polygon with a right click.

Catatan: The reshape tool may alter the starting position of a polygon ring or a closed line. So, the point that is represented 'twice' will not be the same any more. This may not be a problem for most applications, but it is something to consider.


Offset Curves

The  Offset Curve tool creates parallel shifts of line layers. The tool can be applied to the edited layer (the geometries are modified) or also to background layers (in which case it creates copies of the lines / rings and adds them to the the edited layer). It is thus ideally suited for the creation of distance line layers. The displacement is shown at the bottom left of the taskbar.

To create a shift of a line layer, you must first go into editing mode and then select the feature. You can make the  Offset Curve tool active and drag the cross to the desired distance. Your changes may then be saved with the  Save Layer Edits tool.

QGIS options dialog (Digitizing tab then **Curve offset tools** section) allows you to configure some parameters like **Join style**, **Quadrant segments**, **Miter limit**.


Split Features

You can split features using the  Split Features icon on the toolbar. Just draw a line across the feature you want to split.



Split parts

In QGIS 2.0 it is now possible to split the parts of a multi part feature so that the number of parts is increased. Just draw a line across the part you want to split using the  Split Parts icon.


Merge selected features

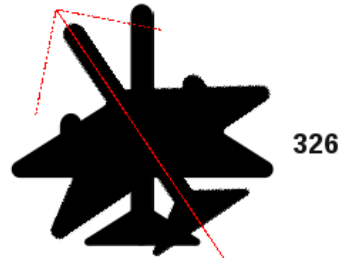
The  Merge Selected Features tool allows you to merge features that have common boundaries. A new dialog will allow you to choose which value to choose between each selected features or select a fonction (Minimum, Maximum, Median, Sum, Skip Attribute) to use for each column.


Merge attributes of selected features

The  Merge Attributes of Selected Features tool allows you to merge attributes of features with common boundaries and attributes without merging their boundaries. First, select several features at once. Then press the  Merge Attributes of Selected Features button. Now QGIS asks you which attributes are to be applied to all selected objects. As a result, all selected objects have the same attribute entries.

Rotate Point Symbols

 Rotate Point Symbols allows you to change the rotation of point symbols in the map canvas. You must first define a rotation column from the attribute table of the point layer in the *Advanced* menu of the *Style* menu of the *Layer Properties*. Also, you will need to go into the ‘SVG marker’ and choose *Data defined properties ...*. Activate *Angle* and choose ‘rotation’ as field. Without these settings, the tool is inactive.



Gambar 12.41: Rotate Point Symbols 


To change the rotation, select a point feature in the map canvas and rotate it, holding the left mouse button pressed. A red arrow with the rotation value will be visualized (see [Figure_edit_4](#)). When you release the left mouse button again, the value will be updated in the attribute table.

Catatan: If you hold the `Ctrl` key pressed, the rotation will be done in 15 degree steps.





12.5.6 Creating new Vector layers

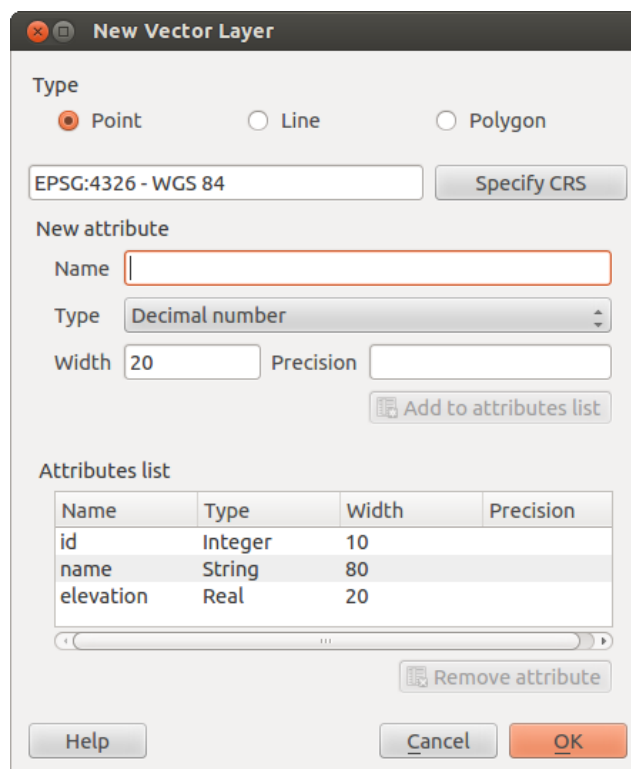
QGIS allows you to create new shapefile layers, new SpatialLite layers, and new GPX layers. Creation of a new GRASS layer is supported within the GRASS plugin. Please refer to section [Creating a new GRASS vector layer](#) for more information on creating GRASS vector layers.

Creating a new Shapefile layer

To create a new shape layer for editing, choose *New* →  *New Shapefile Layer...* from the *Layer* menu. The *New Vector Layer* dialog will be displayed as shown in [Figure_edit_5](#). Choose the type of layer (point, line or polygon) and the CRS (coordinate reference system).


Note that QGIS does not yet support creation of 2.5D features (i.e., features with X,Y,Z coordinates).


To complete the creation of the new shapefile layer, add the desired attributes by clicking on the [**Add to attributes list**] button and specifying a name and type for the attribute. A first ‘id’ column is added as default but can be removed, if not wanted. Only *Type: real* , *Type: integer* , *Type: string*  and *Type: date*  attributes are supported. Additionally and according to the attribute type, you can also define the width and precision of the new attribute column. Once you are happy with the attributes, click [**OK**] and provide a name for the shapefile. QGIS will automatically add a `.shp` extension to the name you specify. Once the layer has been created, it will be added to the map, and you can edit it in the same way as described in section [Digitizing an existing layer](#) above.



Gambar 12.42: Creating a new Shapefile layer Dialog 🐧

Creating a new SpatiaLite layer


To create a new SpatiaLite layer for editing, choose *New* →  *New SpatiaLite Layer...* from the *Layer* menu. The *New SpatiaLite Layer* dialog will be displayed as shown in [Figure_edit_6](#).


The first step is to select an existing SpatiaLite database or to create a new SpatiaLite database. This can be done with the browse button  to the right of the database field. Then, add a name for the new layer, define the layer type, and specify the coordinate reference system with [**Specify CRS**]. If desired, you can select *Create an autoincrementing primary key*.

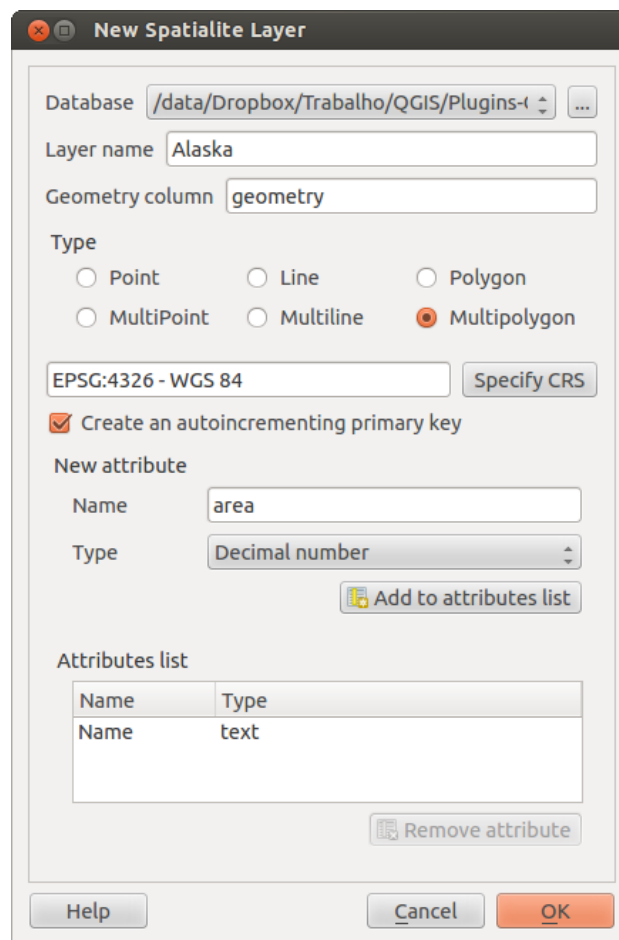
To define an attribute table for the new SpatiaLite layer, add the names of the attribute columns you want to create with the corresponding column type, and click on the [**Add to attribute list**] button. Once you are happy with the attributes, click [**OK**]. QGIS will automatically add the new layer to the legend, and you can edit it in the same way as described in section *Digitizing an existing layer* above.

Further management of SpatiaLite layers can be done with the DB Manager. See [Plugin Pengelola DB](#).

Creating a new GPX layer

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




When this plugin is loaded, choose *New* →  *Create new GPX Layer...* from the *Layer* menu. In the *Save new GPX file as* dialog, you can choose where to save the new GPX layer.



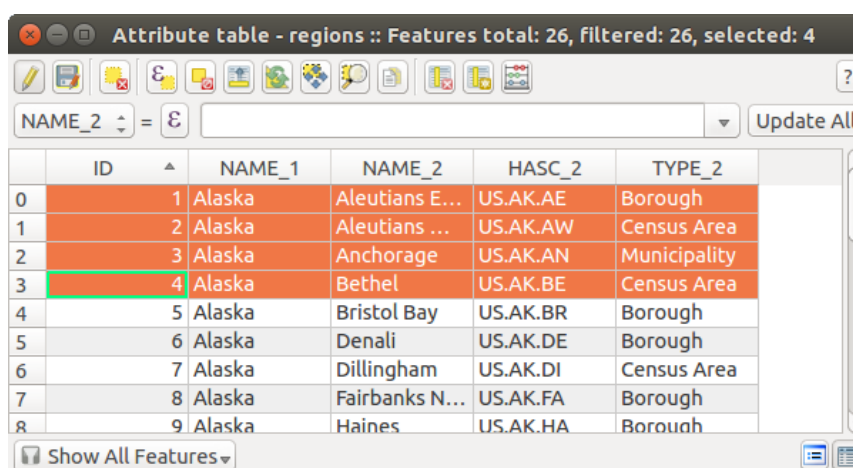
Gambar 12.43: Creating a New Spatialite layer Dialog 

12.5.7 Working with the Attribute Table

The attribute table displays features of a selected layer. Each row in the table represents one map feature, and each column contains a particular piece of information about the feature. Features in the table can be searched, selected, moved or even edited.

To open the attribute table for a vector layer, make the layer active by clicking on it in the map legend area. 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, and to click on the  *Open Attribute Table* button in the Attributes toolbar.

This will open a new window that displays the feature attributes for the layer (*figure_attributes_1*). The number of features and the number of selected features are shown in the attribute table title.



	ID	NAME_1	NAME_2	HASC_2	TYPE_2
0	1	Alaska	Aleutians E...	US.AK.AE	Borough
1	2	Alaska	Aleutians ...	US.AK.AW	Census Area
2	3	Alaska	Anchorage	US.AK.AN	Municipality
3	4	Alaska	Bethel	US.AK.BE	Census Area
4	5	Alaska	Bristol Bay	US.AK.BR	Borough
5	6	Alaska	Denali	US.AK.DE	Borough
6	7	Alaska	Dillingham	US.AK.DI	Census Area
7	8	Alaska	Fairbanks N...	US.AK.FA	Borough
8	9	Alaska	Haines	US.AK.HA	Borough


Gambar 12.44: Attribute Table for regions layer 



Selecting features in an attribute table


Each selected row in the attribute table displays the attributes of a selected feature in the layer. If the set of features selected in the main window is changed, the selection is also updated in the attribute table. Likewise, if the set of rows selected in the attribute table is changed, the set of features selected in the main window will be updated.

Rows can be selected by clicking on the row number on the left side of the row. **Multiple rows** can be marked by holding the **Ctrl** key. A **continuous selection** can be made by holding the **Shift** key and clicking on several row headers on the left side of the rows. All rows between the current cursor position and the clicked row are selected. Moving the cursor position in the attribute table, by clicking a cell in the table, does not change the row selection. Changing the selection in the main canvas does not move the cursor position in the attribute table.

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).

For a **simple search by attributes** on only one column, choose the *Column filter*  from the menu in the bottom left corner. Select the field (column) on which the search should be performed from the drop-down menu, and hit the **[Apply]** button. Then, only the matching features are shown in the attribute table.

To make a selection, you have to use the  *Select features using an Expression* icon on top of the attribute table. 

Select features using an Expression allows you to define a subset of a table using a *Function List* like in the  *Field Calculator* (see *Field Calculator*). The query result can then be saved as a new vector layer. For example, if you want to find regions that are boroughs from *regions.shp* of the QGIS sample data, you have to open the *Fields and Values*

menu and choose the field that you want to query. Double-click the field 'TYPE_2' and also **[Load all unique values]**. From the list, choose and double-click 'Borough'. In the *Expression* field, the following query appears:













```
"TYPE_2" = 'Borough'
```


Here you can also use the *Function list* → *Recent (Selection)* to make a selection that you used before. The expression builder remembers the last 20 used expressions.

The matching rows will be selected, and the total number of matching rows will appear in the title bar of the attribute table, as well as in the status bar of the main window. For searches that display only selected features on the map, use the Query Builder described in section *Query Builder*.


To show selected records only, use *Show Selected Features* from the menu at the bottom left.

The other buttons at the top of the attribute table window provide the following functionality:

-  Toggle editing mode to edit single values and to enable functionalities described below (also with Ctrl+E)
-  Save Edits (also with Ctrl+S)
-  Unselect all (also with Ctrl+U)
-  Move selected to top (also with Ctrl+T)
-  Invert selection (also with Ctrl+R)
-  Copy selected rows to clipboard (also with Ctrl+C)
-  Zoom map to the selected rows (also with Ctrl+J)
-  Pan map to the selected rows (also with Ctrl+P)
-  Delete selected features (also with Ctrl+D)
-  New Column for PostGIS layers and for OGR layers with GDAL version >= 1.6 (also with Ctrl+W)
-  Delete Column for PostGIS layers and for OGR layers with GDAL version >= 1.9 (also with Ctrl+L)
-  Open field calculator (also with Ctrl+I)

Below these buttons is the Field Calculator bar, which allows calculations to be quickly applied attributes visible in the table. This bar uses the same expressions as the  Field Calculator (see *Field Calculator*).

Tip: Skip WKT geometry

If you want to use attribute data in external programs (such as Excel), use the  Copy selected rows to clipboard button. You can copy the information without vector geometries if you deactivate *Settings* → *Options* → *Data sources* menu *Copy geometry in WKT representation from attribute table*.

Save 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). Just open the right mouse menu of the layer and click on *Save as* to define the name of the output file, its format and CRS (see section *Legenda Peta*). To save the selection ensure that the *Save only selected features* is selected. It is also possible to specify OGR creation options within the dialog.

Paste into new layer

Features that are on the clipboard may be pasted into a new layer. To do this, first make a layer editable. 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* or *New memory layer*.

This applies to features selected and copied within QGIS and also to features from another source defined using well-known text (WKT).

Working with non spatial attribute tables

QGIS allows you also to load non-spatial tables. This currently includes tables supported by OGR and delimited text, as well as the PostgreSQL, MSSQL and Oracle provider. The tables can be used for field lookups or just generally browsed and edited using the table view. When you load the table, you will see it in the legend field. It can be opened with the  Open Attribute Table tool and is then editable like any other layer attribute table.

As an example, you can use columns of the 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 *Fields Menu* to find out more.

12.5.8 Creating one 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.

As an example you have a layer with all regions of alaska (polygon) which provides some attributes about its name and region type and a unique id (which acts as primary key).

Foreign keys

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 region layer, you need to create a one to many relation using foreign keys, because there are several airports in most regions.



Gambar 12.45: Alaska region with airports 

In addition to the already existing attributes in the airports attribute table 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 the editing and QGIS takes care about 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.

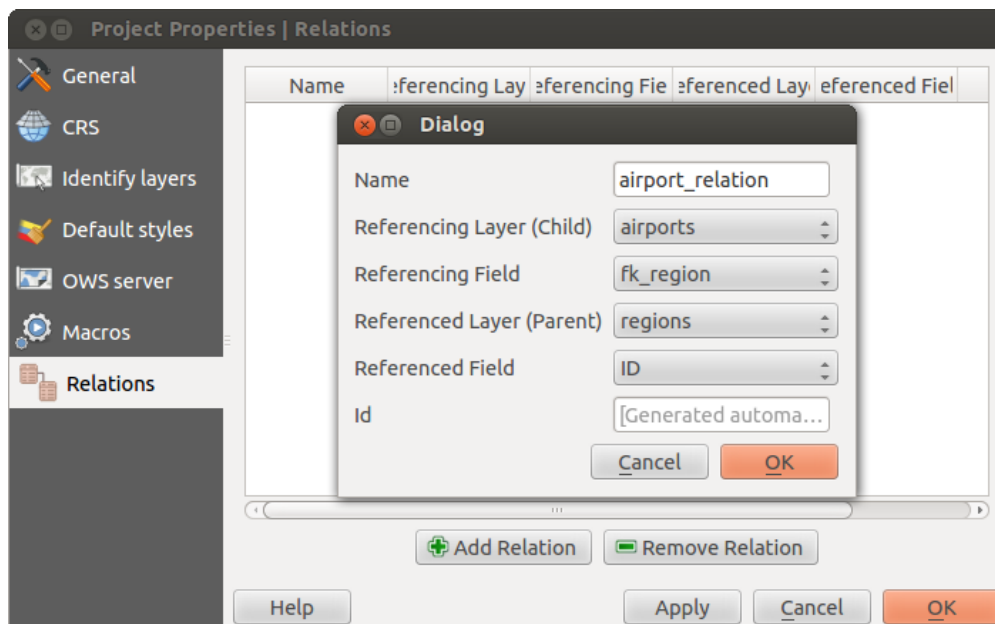
Layers

QGIS makes no difference between a table and a vector layer. Basically, a vector layer is a table with a geometry. So can add your table as a vector layer. To demonstrate you can load the 'region' shapefile (with geometries) and the 'airport' csv table (without geometries) and a foreign key (`fk_region`) to the layer region. 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).

Definition (Relation Manager)

The first thing we are going to do is to let QGIS know about the relations between the layer. This is done in *Settings* → *Project Properties*. Open the *Relations* menu and click on *Add*.

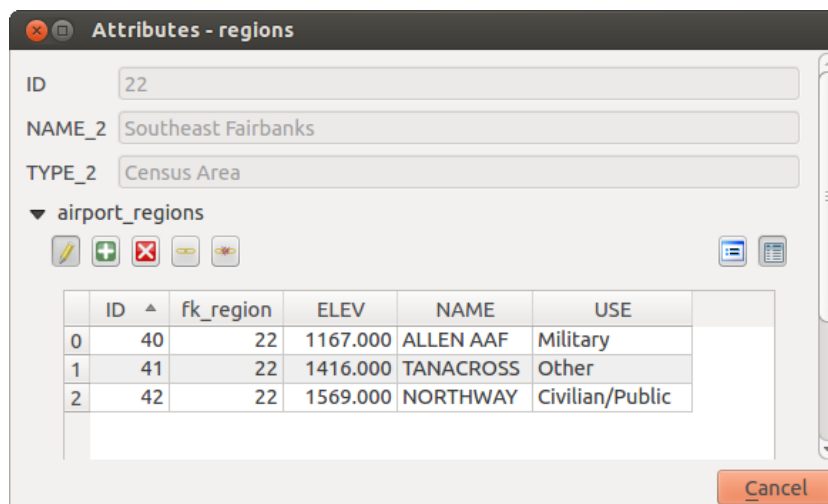
- **name** is going to be used as a title. It should be a human readable string, describing, what the relation is used for. We will just call say "Airports" in this case.
- **referencing layer** is the one with the foreign key field on it. In our case this is the airports layer
- **referencing field** will say, which field points to the other layer so this is `fk_region` in this case
- **referenced layer** is the one with the primary key, pointed to, so here it is the regions layer
- **referenced field** is the primary key of the referenced layer so it is `ID`
- **id** will be used for internal purposes and has to be unique. You may need it to build custom forms once this is supported. 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.



Gambar 12.46: Relation Manager 






Forms

Now that QGIS knows about the relation, it will be used to improve the forms it generates. As we did not change the default form method (autogenerated) it will just add a new widget in our form. So let's select the layer region in the legend and use the identify tool. Depending on your settings, the form might open directly or you will have to choose to open it in the identification dialog under actions.



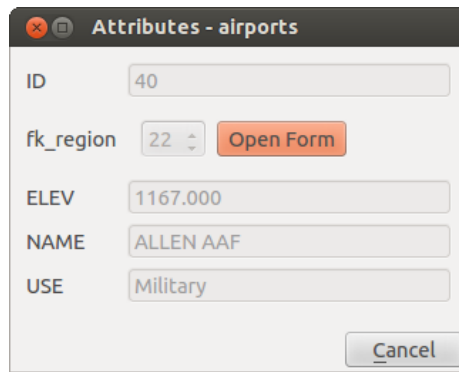
Gambar 12.47: Identification dialog regions with relation to airports 🐧


As you can see, the airports assigned to this particular region are all shown in a table. And there are also some buttons available. Let's review them shortly

- The  button is for toggling the edit mode. Be aware that it toggles the edit mode of the airport layer, although we are in the feature form of a feature from the region layer. But the table is representing features of the airport layer.
- The  button will add a new feature to the airport layer. And it will assign the new airport to the current region by default.
- The  button will delete the selected airport permanently.
- The  symbol will open a new dialog where you can select any existing airport which will then be assigned to the current region. This may be handy if you created the airport on the wrong region by accident.
- The  symbol will unlink the selected airport from the current region, leaving them unassigned (the foreign key is set to NULL) effectively.
- The two buttons to the right switch between table view and form view where the later let's you view all the airports in their respective form.

If you work on the airport table, a new widget type is available which lets you embed the feature form of the referenced region on the feature form of the airports. It can be used when you open the layer properties of the airports table, switch to the *Fields* menu and change the widget type of the foreign key field 'fk_region' to Relation Reference.

If you look at the feature dialog now, you will see, that the form of the region is embedded inside the airports form and will even have a combobox, which allows you to assign the current airport to another region.



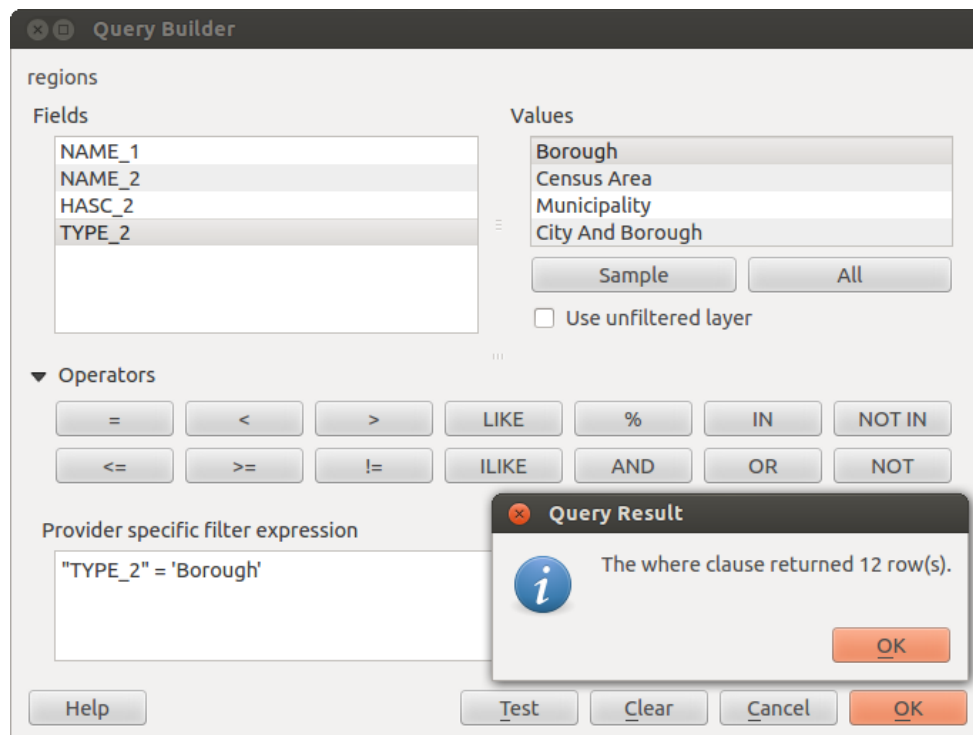
Gambar 12.48: Identification dialog airport with relation to regions 


12.6 Query Builder

The Query Builder allows you to define a subset of a table using a SQL-like WHERE clause and to display the result in the main window. The query result can then be saved as a new vector layer.

12.6.1 Query

Open the **Query Builder** by opening the Layer Properties and going to the *General* menu. Under *Feature subset*, click on the **[Query Builder]** button to open the *Query builder*. For example, if you have a *regions* layer with a *TYPE_2* field, you could select only regions that are *borough* in the *Provider specific filter expression* box of the Query Builder. [Figure_attributes_2](#) shows an example of the Query Builder populated with the *regions.shp* layer from the QGIS sample data. The Fields, Values and Operators sections help you to construct the SQL-like query.



Gambar 12.49: Query Builder 

The **Fields list** contains all attribute columns of the attribute table to be searched. To add an attribute column to

the SQL WHERE clause field, double click its name in the Fields list. Generally, you can use the various fields, values and operators to construct the query, or you can just type it into the SQL box.

The **Values list** lists the values of an attribute table. To list all possible values of an attribute, select the attribute in the Fields list and click the **[all]** button. To list the first 25 unique values of an attribute column, select the attribute column in the Fields list and click the **[Sample]** button. To add a value to the SQL WHERE clause field, double click its name in the Values 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 shows a message box with the number of features satisfying the current query, which is useful in the process of query construction. The **[Clear]** button clears the text in the SQL WHERE clause text field. The **[OK]** button closes the window and selects the features satisfying the query. The **[Cancel]** button closes the window without changing the current selection.

QGIS treats the resulting subset acts as if it were the entire layer. For example if you applied the filter above for 'Borough', you can not display, query, save or edit Ankorage, because that is a 'Manicpality' and therefore not part of the subset.

The only exception is that unless your layer is part of a database, using a subset will prevent you from editing the layer.

12.7 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 written to a new attribute field, a virtual field, or they can be used to update values in an existing field.

Tip: Virtual Fields

- Virtual fields are not permanent and are not saved.
- To make a field virtual it must be done when the field is made.

The field calculator is now available on any layer that supports edit. When you click on the field calculator icon the dialog opens (see [figure_attributes_3](#)). 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.





The quick field calculation bar in top of the attribute table is only visible if the layer is editable.

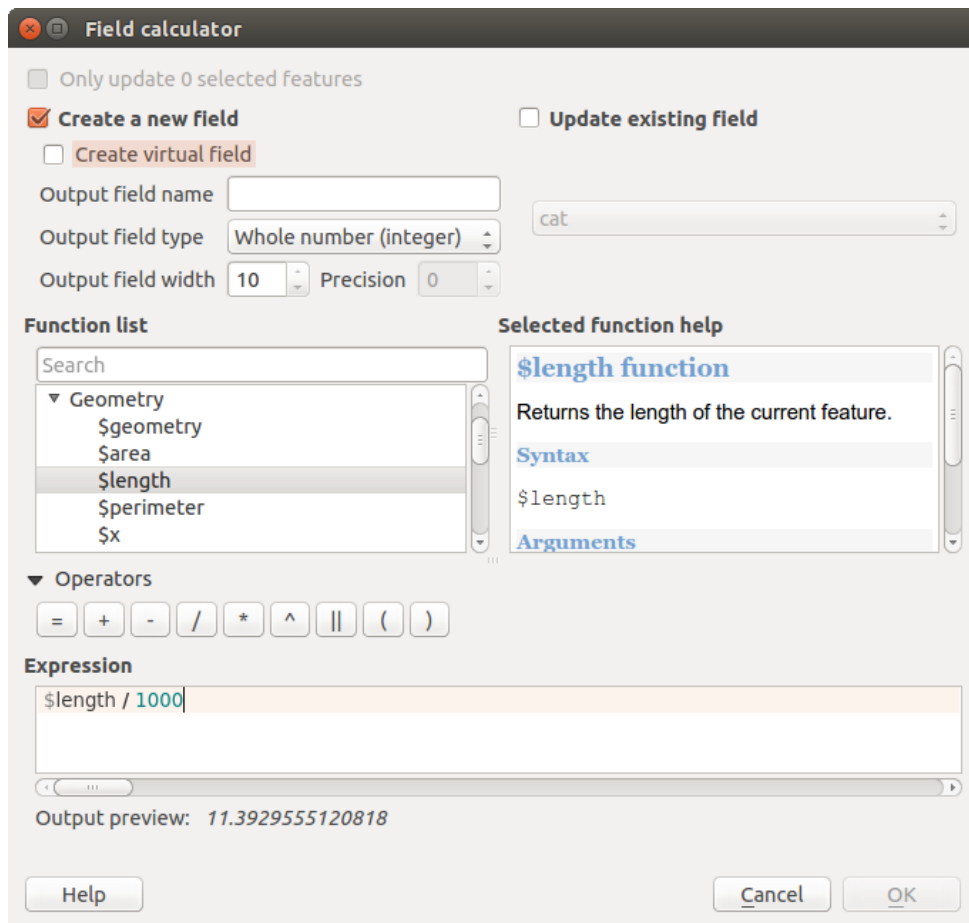
In quick field calculation bar, you first select the existing field name then open the expression dialog to create your expression or write it directly in the field then click on **Update All** button.

In the field calculator dialog, you first must select whether you want to only update selected features, create a new attribute field where the results of the calculation will be added or update an existing field.

If you choose to add a new field, you need to enter a field name, a field type (integer, real or string), the total field width, and the field precision (see [figure_attributes_3](#)). For example, if you choose a field width of 10 and a field precision of 3, it means you have 6 digits before the dot, then the dot and another 3 digits for the precision.

A short example illustrates how the field calculator works. We want to calculate the length in km of the railroads layer from the QGIS sample dataset:

1. Load the shapefile `railroads.shp` in QGIS and press  Open Attribute Table.
2. Click on  Toggle editing mode and open the  Field Calculator dialog.
3. Select the  *Create a new field* checkbox to save the calculations into a new field.



Gambar 12.50: Field Calculator 

4. Add `length` as Output field name and `real` as Output field type, and define Output field width to be 10 and Precision, 3.
5. Now double click on function `$length` in the *Geometry* group to add it into the Field calculator expression box.
6. Complete the expression by typing `"/ 1000"` in the Field calculator expression box and click **[Ok]**.
7. You can now find a new field `length` in the attribute table.

The available functions are listed in *Expressions* chapter.

Pekerjaan dengan Data Raster

13.1 Working with Raster Data

This section describes how to visualize and set raster layer properties. QGIS uses the GDAL library to read and write raster data formats, including ArcInfo Binary Grid, ArcInfo ASCII Grid, GeoTIFF, ERDAS IMAGINE, and many more. GRASS raster support is supplied by a native QGIS data provider plugin. The raster data can also be loaded in read mode from zip and gzip archives into QGIS.

As of the date of this document, more than 100 raster formats are supported by the GDAL library (see GDAL-SOFTWARE-SUITE in *Literatur dan Referensi Web*). A complete list is available at http://www.gdal.org/formats_list.html.

Catatan: Not all of the listed formats may work in QGIS for various reasons. For example, some require external commercial libraries, or the GDAL installation of your OS may not have been built to support the format you want to use. Only those formats that have been well tested will appear in the list of file types when loading a raster into QGIS. Other untested formats can be loaded by selecting the [GDAL] All files (*) filter.

Working with GRASS raster data is described in section *GRASS GIS Integration*.

13.1.1 What is raster data?

Raster data in GIS are matrices of discrete cells that represent features on, above or below the earth's surface. Each cell in the raster grid is 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 an elevation matrix.

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 in the map canvas.

QGIS makes use of georeference information inside the raster layer (e.g., GeoTiff) or in an appropriate world file to properly display the data.

13.1.2 Loading raster data in QGIS

Raster layers are loaded either by clicking on the  Add Raster Layer icon or by selecting the *Layer* →  Add Raster Layer menu option. More than one layer can be loaded at the same time by holding down the `Ctrl` or `Shift` key and clicking on multiple items in the *Open a GDAL Supported Raster Data Source* dialog.

Once a raster layer is loaded in the map legend, you can click on the layer name with the right mouse button to select and activate layer-specific features or to open a dialog to set raster properties for the layer.

Right mouse button menu for raster layers

- *Zoom to Layer Extent*
- *Zoom to Best Scale (100%)*
- *Stretch Using Current Extent*
- *Show in Overview*
- *Remove*
- *Duplicate*
- *Set Layer CRS*
- *Set Project CRS from Layer*
- *Save as ...*
- *Properties*
- *Rename*
- *Copy Style*
- *Add New Group*
- *Expand all*
- *Collapse all*
- *Update Drawing Order*

13.2 Raster Properties Dialog

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 (see [figure_raster_1](#)).

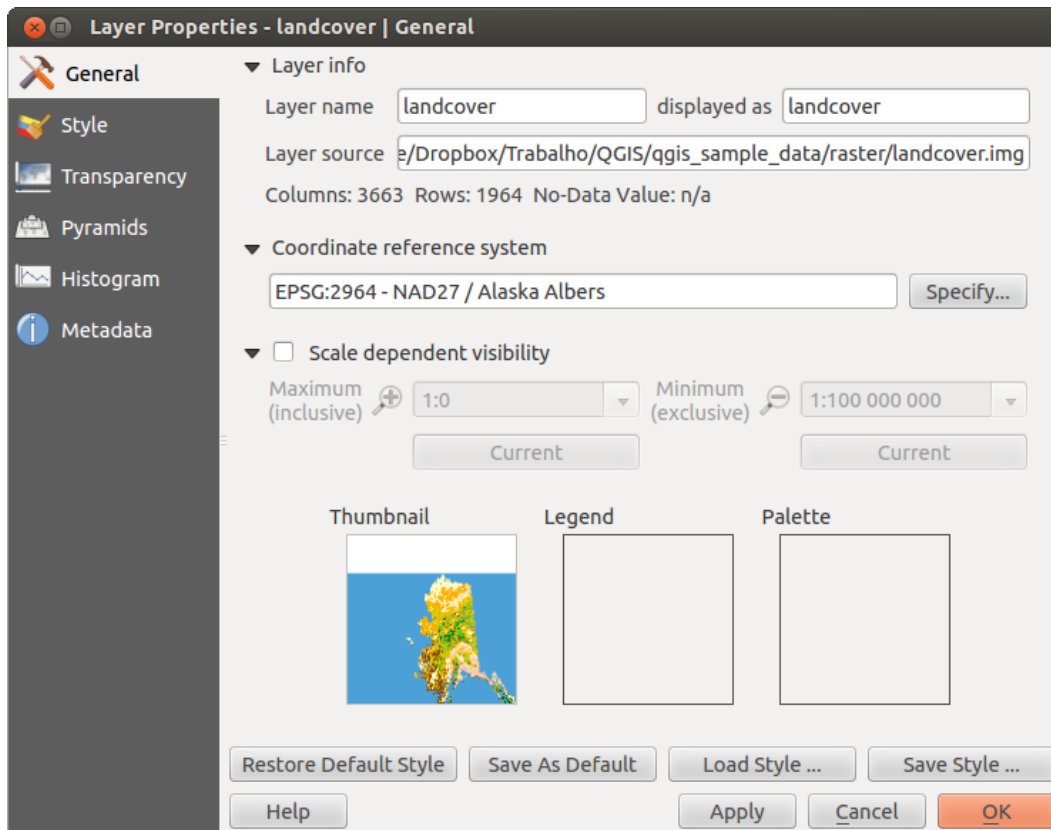
There are several menus in the dialog:

- *General*
- *Style*
- *Transparency*
- *Pyramids*
- *Histogram*
- *Metadata*

13.2.1 General Menu

Layer Info

The *General* menu displays basic information about the selected raster, including the layer source path, the display name in the legend (which can be modified), and the number of columns, rows and no-data values of the raster.

Gambar 13.1: Raster Layers Properties Dialog 

Coordinate reference system

Here, you find the coordinate reference system (CRS) information printed as a PROJ.4 string. If this setting is not correct, it can be modified by clicking the **[Specify]** button.

Scale Dependent visibility

Additionally scale-dependent visibility can be set in this tab. You will need to check the checkbox and set an appropriate scale where your data will be displayed in the map canvas.

At the bottom, you can see a thumbnail of the layer, its legend symbol, and the palette.

13.2.2 Style Menu

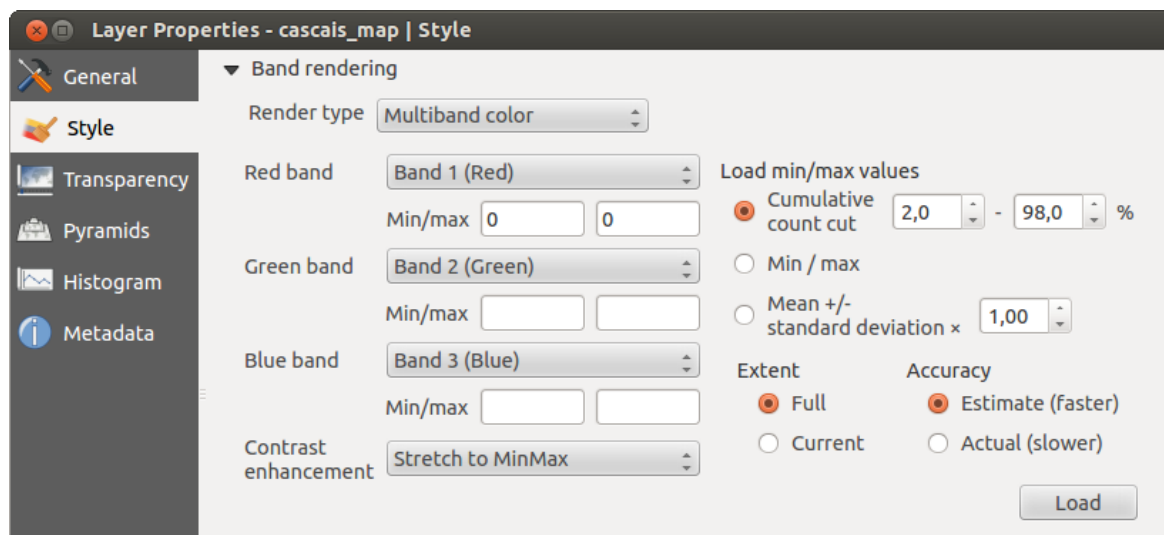
Band rendering

QGIS offers four different *Render types*. The renderer chosen is dependent on the data type.

1. Multiband color - if the file comes as a multiband with several bands (e.g., used with a satellite image with several bands)
2. Paletted - if a single band file comes with an indexed palette (e.g., used with a digital topographic map)
3. Singleband gray - (one band of) the image will be rendered as gray; QGIS will choose this renderer if the file has neither multibands nor an indexed palette nor a continuous palette (e.g., used with a shaded relief map)
4. Singleband pseudocolor - this renderer is possible for files with a continuous palette, or color map (e.g., used with an elevation map)

Multiband color

With the multiband color renderer, three selected bands from the image will be rendered, each band representing the red, green or blue component that will be used to create a color image. You can choose several *Contrast enhancement* methods: 'No enhancement', 'Stretch to MinMax', 'Stretch and clip to MinMax' and 'Clip to min max'.



Gambar 13.2: Raster Renderer - Multiband color 🐧

This selection offers you a wide range of options to modify the appearance of your raster layer. First of all, you have to get the data range from your image. This can be done by choosing the *Extent* and pressing [Load]. QGIS can *Estimate (faster)* the *Min* and *Max* values of the bands or use the *Actual (slower) Accuracy*.

Now you can scale the colors with the help of the *Load min/max values* section. A lot of images have a few very low and high data. These outliers can be eliminated using the *Cumulative count cut* setting. The standard data range is set from 2% to 98% of the data values and can be adapted manually. With this setting, the gray character of the image can disappear. With the scaling option *Min/max*, QGIS creates a color table with all of the data included in the original image (e.g., QGIS creates a color table with 256 values, given the fact that you have 8 bit bands). You can also calculate your color table using the *Mean +/- standard deviation x* . Then, only the values within the standard deviation or within multiple standard deviations are considered for the color table. This is useful when you have one or two cells with abnormally high values in a raster grid that are having a negative impact on the rendering of the raster.

All calculations can also be made for the *Current* extent.

Tip: Viewing a Single Band of a Multiband Raster

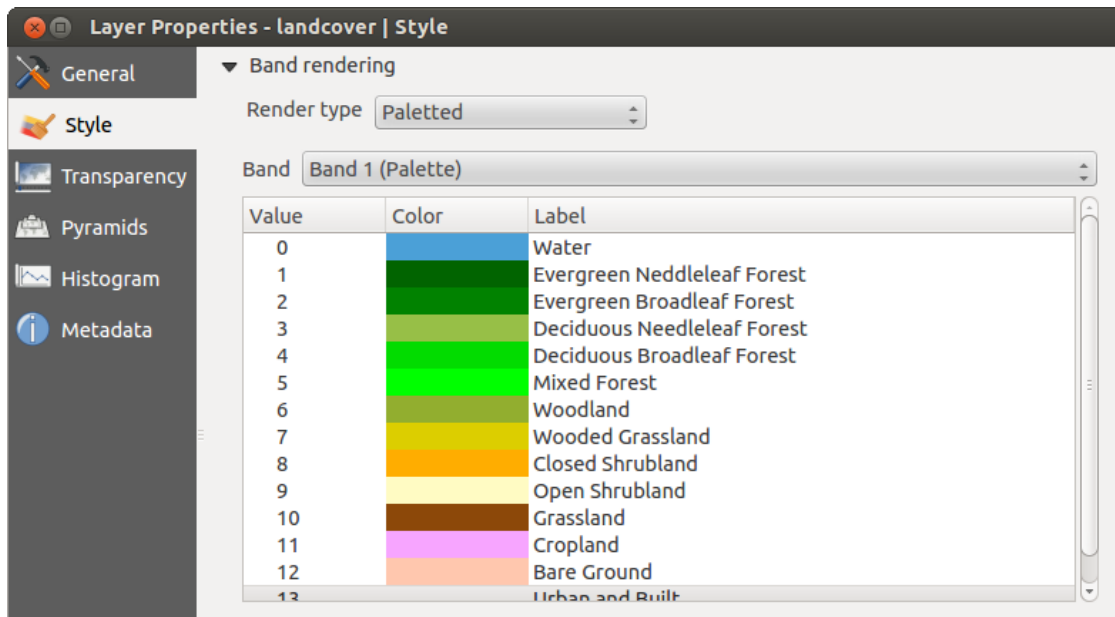
If you want to view a single band of a multiband image (for example, Red), you might think you would set the Green and Blue bands to "Not Set". But this is not the correct way. To display the Red band, set the image type to 'Singleband gray', then select Red as the band to use for Gray.

Paletted

This is the standard render option for singleband files that already include a color table, where each pixel value is assigned to a certain color. In that case, the palette is rendered automatically. If you want to change colors assigned to certain values, just double-click on the color and the *Select color* dialog appears. Also, in QGIS 2.2, it's now possible to assign a label to the color values. The label appears in the legend of the raster layer then.

Contrast enhancement

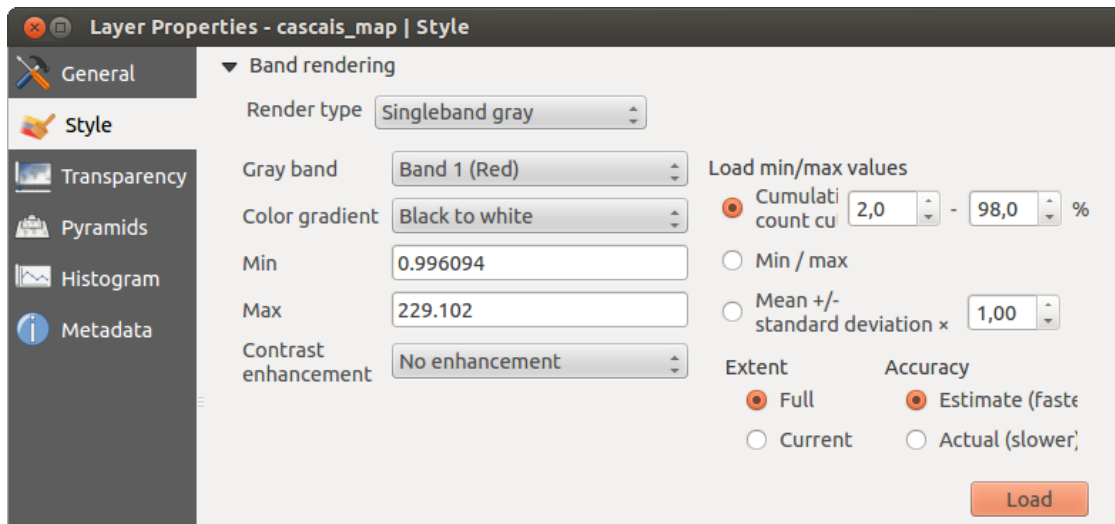
Catatan: When adding GRASS rasters, the option *Contrast enhancement* will always be set automatically to *stretch to min max*, regardless of if this is set to another value in the QGIS general options.



Gambar 13.3: Raster Renderer - Paletted 🐧

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 define a *Min* and a *Max* value by choosing the *Extent* first and then pressing [Load]. QGIS can *Estimate (faster)* the *Min* and *Max* values of the bands or use the *Actual (slower)* Accuracy.

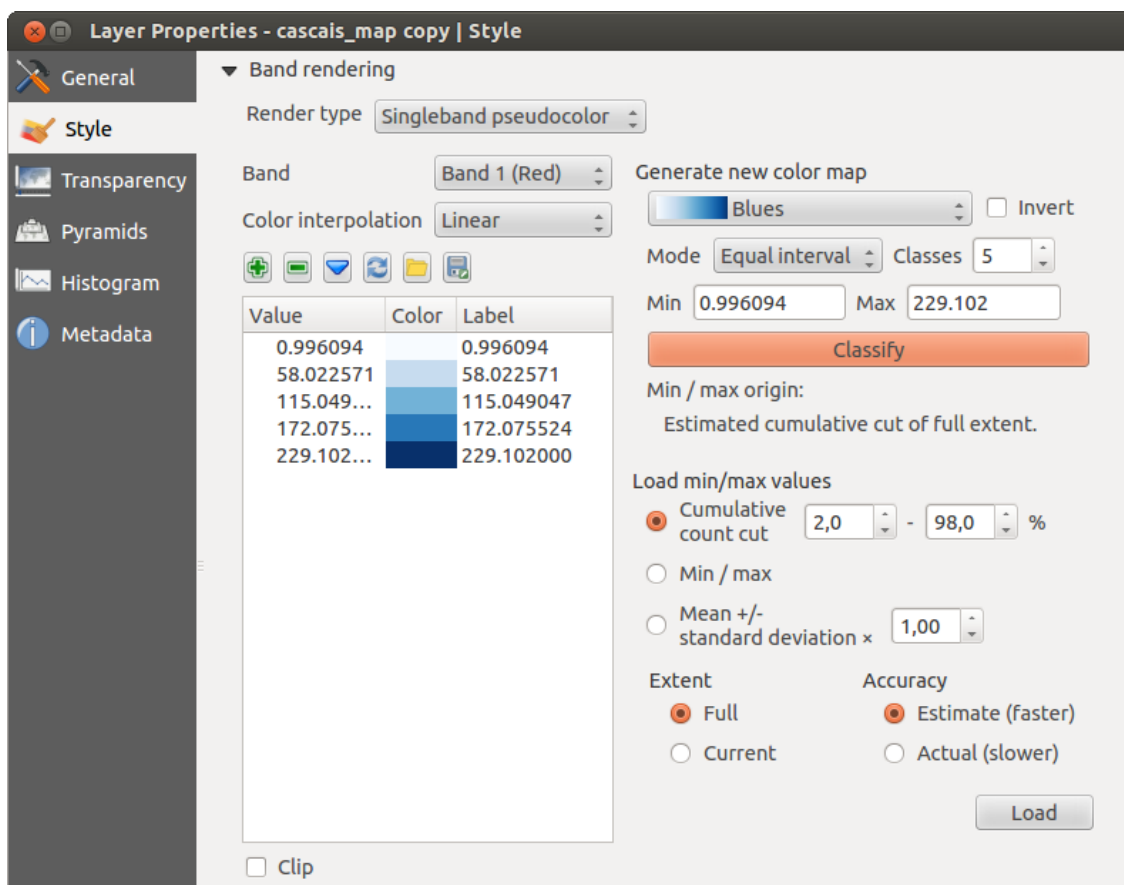


Gambar 13.4: Raster Renderer - Singleband gray 🐧

With the *Load min/max values* section, scaling of the color table is possible. Outliers can be eliminated using the *Cumulative count cut* setting. The standard data range is set from 2% to 98% of the data values and can be adapted manually. With this setting, the gray character of the image can disappear. Further settings can be made with *Min/max* and *Mean +/- standard deviation x* . While the first one creates a color table with all of the data included in the original image, the second 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 grid that are having a negative impact on the rendering of the raster.







Singleband pseudocolor




This is a render option for single-band files, including a continuous palette. You can also create individual color maps for the single bands here. Three types of color interpolation are available:



Gambar 13.5: Raster Renderer - Singleband pseudocolor

1. Discrete
2. Linear
3. Exact

In the left block, 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, and the  Sort colormap items button sorts the color table according to the pixel values in the value column. 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. Further, you can also add labels for each color, but this value won't be displayed when you use the identify feature tool. You can also click on the button  Load color map from band, which tries to load the table from the band (if it has any). And 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 defined color table for other sessions.

In the right block, *Generate new color map* allows you to create newly categorized color maps. For the *Classification mode*  'Equal interval', you only need to select the *number of classes*  and press the button *Classify*. You can invert the colors of the color map by clicking the *Invert* checkbox. In the case of the *Mode*  'Continuous', QGIS creates classes automatically depending on the *Min* and *Max*. Defining *Min/Max* values can be done with the help of the *Load min/max values* section. A lot of images have a few very low and high data. These outliers can be eliminated using the *Cumulative count cut* setting. The standard data range is set from 2% to 98% of the data values and can be adapted manually. With this setting, the gray character of the image can disappear. With the scaling option *Min/max*, QGIS creates a color table with all of the data included in the

original image (e.g., QGIS creates a color table with 256 values, given the fact that you have 8 bit bands). You can also calculate your color table using the *Mean +/- standard deviation x 1,00*. Then, only the values within the standard deviation or within multiple standard deviations are considered for the color table.

Color rendering

For every *Band rendering*, a *Color rendering* is possible.

You can also achieve special rendering effects for your raster file(s) using one of the blending modes (see *The Vector Properties Dialog*).

Further settings can be made in modifying the *Brightness*, the *Saturation* and the *Contrast*. You can also use a *Grayscale* option, where you can choose between ‘By lightness’, ‘By luminosity’ and ‘By average’. For one hue in the color table, you can modify the ‘Strength’.

Resampling


The *Resampling* option makes its appearance 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.



Gambar 13.6: Raster Rendering - Resampling 

When applying the ‘Nearest neighbour’ method, the map can have a pixelated structure when zooming in. This appearance can be improved by using the ‘Bilinear’ or ‘Cubic’ method, which cause sharp features to be blurred. The effect is a smoother image. This method can be applied, for instance, to digital topographic raster maps.


13.2.3 Transparency Menu

QGIS has the ability to display each raster layer at a different transparency level. Use the transparency slider  to indicate to what extent the underlying layers (if any) should be visible through the current raster layer. This is very useful if you like to overlay more than one raster layer (e.g., a shaded relief map overlaid by a classified raster map). This will make the look of the map more three dimensional.



Additionally, you can enter a raster value that should be treated as *NODATA* in the *Additional no data value* menu.

An even more flexible way to customize the transparency can be done in the *Custom transparency options* section. The transparency of every pixel can be set here.

As an example, we want to set the water of our example raster file `landcover.tif` to a transparency of 20%. The following steps are necessary:

1. Load the raster file `landcover.tif`.
2. Open the *Properties* dialog by double-clicking on the raster name in the legend, or by right-clicking and choosing *Properties* from the pop-up menu.
3. Select the *Transparency* menu.
4. From the *Transparency band* menu, choose 'None'.
5. Click the  Add values manually button. A new row will appear in the pixel list.
6. Enter the raster value in the 'From' and 'To' column (we use 0 here), and adjust the transparency to 20%.
7. Press the **[Apply]** button and have a look at the map.

You can repeat steps 5 and 6 to adjust more values with custom transparency.

As you can see, it is quite easy to set custom transparency, but it can be quite a lot of work. Therefore, you can use the button  Export to file to save your transparency list to a file. The button  Import from file loads your transparency settings and applies them to the current raster layer.

13.2.4 Pyramids Menu

Large 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 level of zoom.

You must have write access in the directory where the original data is stored to build pyramids.




Several resampling methods can be used to calculate the pyramids:

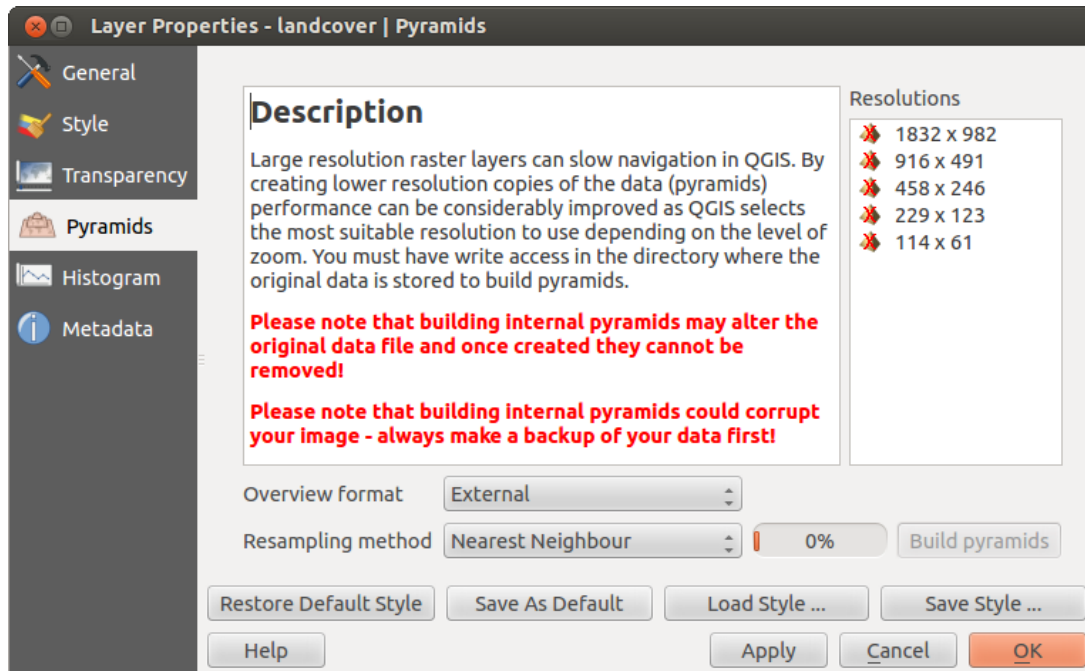
- Nearest Neighbour
- Average
- Gauss
- Cubic
- Mode
- None

If you choose 'Internal (if possible)' from the *Overview format* menu, QGIS tries to build pyramids internally. You can also choose 'External' and 'External (Erdas Imagine)'.

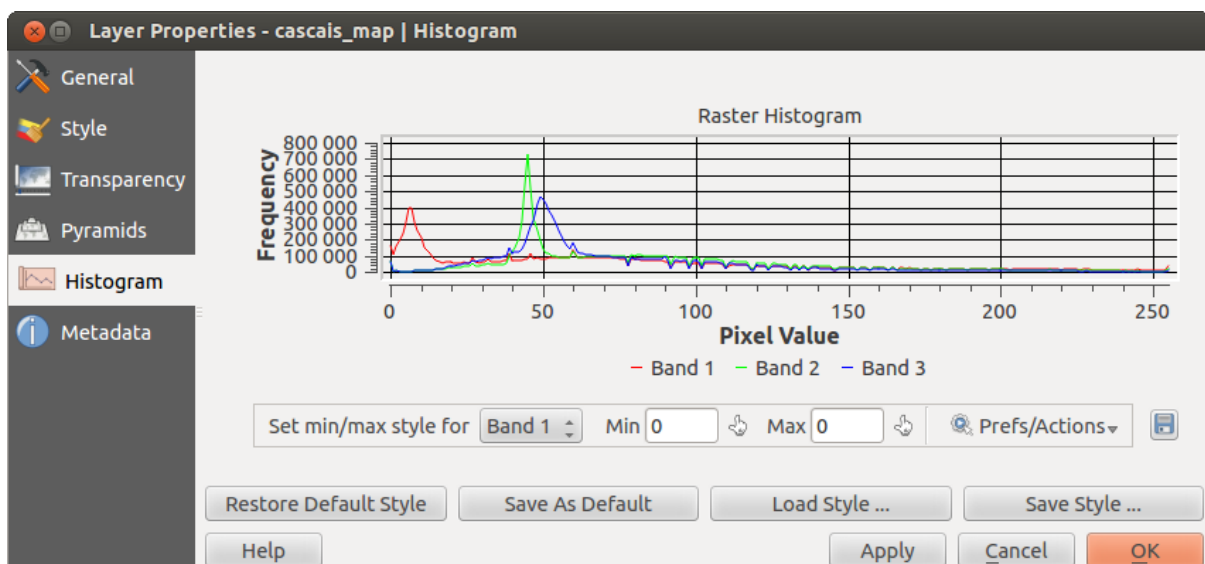
Please note that building pyramids may alter the original data file, and once created they cannot be removed. If you wish to preserve a 'non-pyramided' version of your raster, make a backup copy prior to building pyramids.

13.2.5 Histogram Menu

The *Histogram* menu allows you to view the distribution of the bands or colors in your raster. The histogram is generated automatically when you open the *Histogram* menu. All existing bands will be displayed together. You can save the histogram as an image with the  button. With the *Visibility* option in the  *Prefs/Actions* menu, you can display histograms of the 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'. With the *Actions* option, you can 'Reset' and 'Recompute histogram' after you have chosen the *Min/max options*.



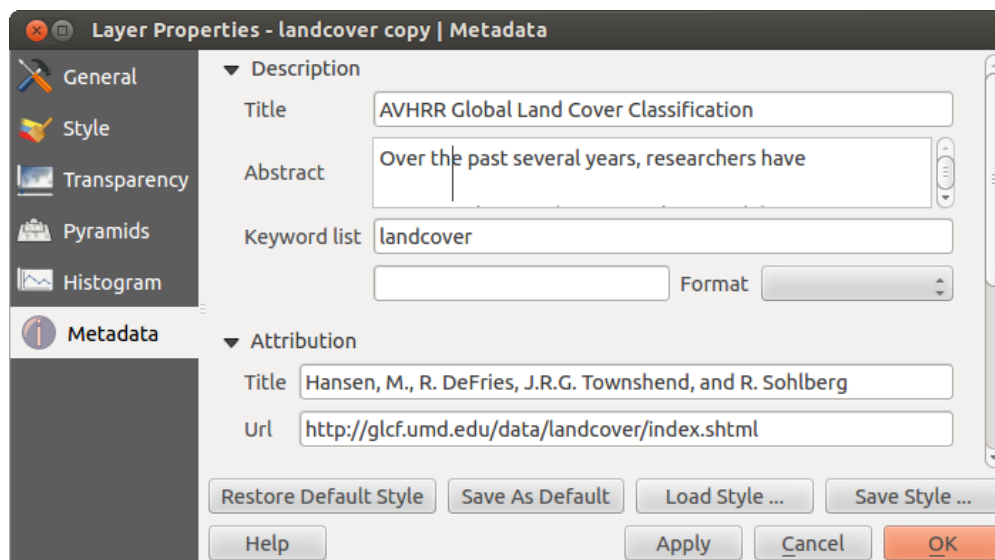
Gambar 13.7: The Pyramids Menu 



Gambar 13.8: Raster Histogram 

13.2.6 Metadata Menu

The *Metadata* menu displays a wealth of information about the raster layer, including statistics about each band in the current raster layer. From this menu, entries may be made for the *Description*, *Attribution*, *MetadataUrl* and *Properties*. In *Properties*, statistics are gathered on a ‘need to know’ basis, so it may well be that a given layer’s statistics have not yet been collected.



Gambar 13.9: Raster Metadata 

13.3 Kalkulator Raster

The *Raster Calculator* in the *Raster* menu allows you to perform calculations on the basis of existing raster pixel values (see [figure_raster_10](#)). The results are written to a new raster layer with a GDAL-supported format.

Daftar **Pita Raster** berisi semua lapisan raster yang dapat digunakan. Untuk menambahkan raster ke kalkulator raster kolom ekspresi, klik dua kali namanya dalam daftar kolom. Anda kemudian dapat menggunakan operator untuk membangun perhitungan ekspresi, atau Anda hanya dapat mengetik mereka ke dalam kotak.

Dalam bagian **Hasil lapisan**, Anda akan perlu mendefinisikan keluaran lapisan. Anda kemudian dapat menentukan luasnya daerah perhitungan berdasarkan masukan lapisan raster, atau berdasarkan koordinat X, Y dan pada kolom dan baris, untuk mengatur resolusi dari keluaran lapisan. Jika lapisan masukan memiliki resolusi yang berbeda, nilai-nilai akan resampled dengan algoritma tetangga terdekat.

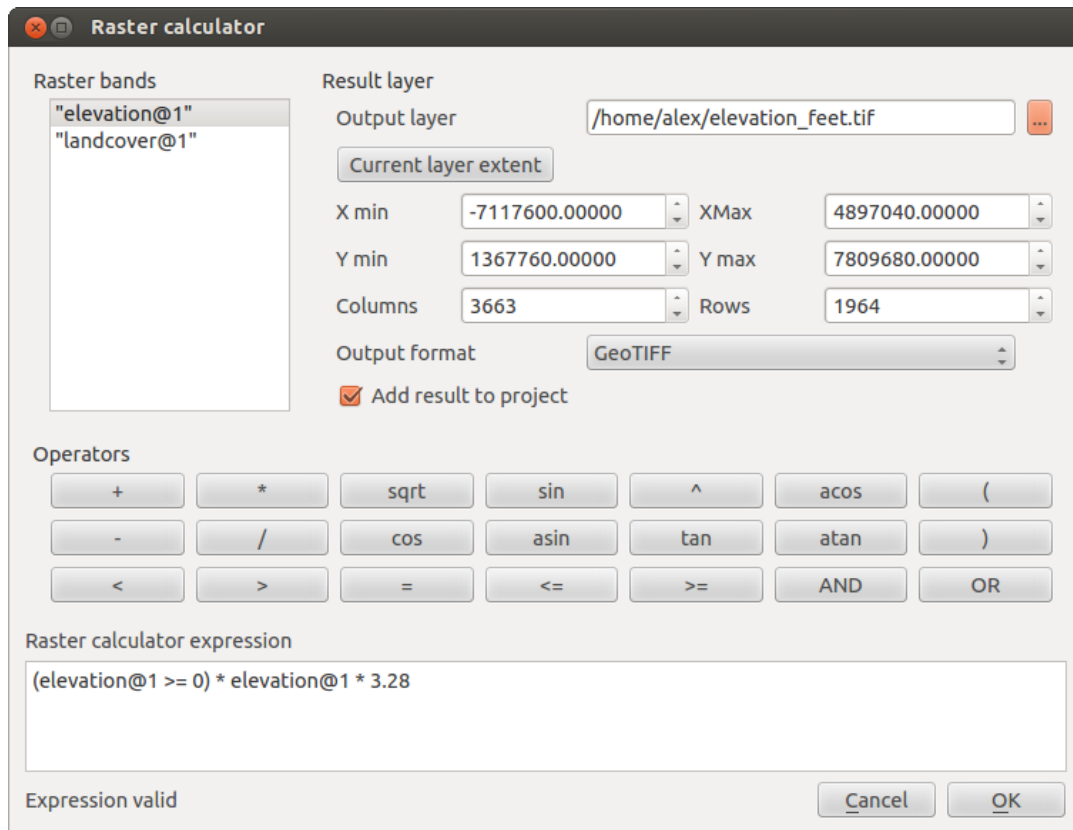
Bagian **Operator** berisi semua operator yang tersedia. Untuk menambahkan operator ke kotak ekspresi kalkulator raster, klik tombol yang sesuai. Perhitungan matematika tersedia (+, -, *, ...) dan fungsi trigonometri (*sin*, *cos*, *tan*, ...). Nantikan operator lainnya yang akan datang!

Dengan kotak centang *Tambahkan hasil ke proyek*, lapisan hasil secara otomatis akan ditambahkan ke area legenda dan dapat divisualisasikan.

13.3.1 Contoh-contoh

Konversi nilai elevasi dari meter ke kaki

Membuat elevasi raster dalam kaki dari raster dalam meter, Anda perlu menggunakan faktor konversi meter ke kaki: 3.28. Ekspresinya adalah:



Gambar 13.10: Kalkulator Raster 

```
"elevation@1" * 3.28
```

Menggunakan masker

Jika Anda ingin untuk menutupi bagian-bagian dari suatu raster - kata, misalnya, karena Anda hanya tertarik pada ketinggian di atas 0 meter – Anda dapat menggunakan ekspresi berikut untuk membuat masker dan menerapkan hasil untuk raster dalam satu langkah.

```
("elevation@1" >= 0) * "elevation@1"
```

Dengan kata lain, untuk setiap sel yang lebih besar dari atau sama dengan 0, atur nilainya ke 1. Jika nilai ke 0. Ini menciptakan masker on the fly.

If you want to classify a raster – say, for instance into two elevation classes, you can use the following expression to create a raster with two values 1 and 2 in one step.

```
("elevation@1" < 50) * 1 + ("elevation@1" >= 50) * 2
```

In other words, for every cell less than 50 set its value to 1. For every cell greater than or equal 50 set its value to 2.

.

Pekerjaan dengan Data OGC

14.1 QGIS sebagai OGC Klien Data

The Open Geospatial Consortium (OGC) is an international organization with membership of more than 300 commercial, governmental, nonprofit and research organizations worldwide. Its members develop and implement standards for geospatial content and services, GIS data processing and exchange.

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 <http://www.opengeospatial.org/>.

Spesifikasi penting OGC yang didukung oleh QGIS adalah:

- **WMS** — Web Map Service (*Klien WMS/WMTS*)
- **WMTS** — Web Map Tile Service (*Klien WMS/WMTS*)
- **WFS** — Web Feature Service (*Klien WFS dan WFS-T*)
- **WFS-T** — Web Feature Service - Transactional (*Klien WFS dan WFS-T*)
- **WCS** — Web Coverage Service (*Klien WCS*)
- **SFS** — Simple Features for SQL (*PostGIS Layers*)
- **GML** — Geography Markup Language

OGC services are increasingly being used to exchange geospatial data between different GIS implementations and data stores. QGIS can deal with the above specifications as a client, being **SFS** (through support of the PostgreSQL / PostGIS data provider, see section *PostGIS Layers*).

14.1.1 Klien WMS/WMTS

Sekilas Dukungan WMS

QGIS currently can act as a WMS client that understands WMS 1.1, 1.1.1 and 1.3 servers. In particular, it has been tested against publicly accessible servers such as DEMIS.

A WMS server acts upon requests by the client (e.g., QGIS) for a raster map with a given extent, set of layers, symbolization style, and transparency. The WMS server then consults its local data sources, rasterizes the map, and sends it back to the client in a raster format. For QGIS, this format would typically be JPEG or PNG.

WMS is generically a REST (Representational State Transfer) service rather than a full-blown Web service. As such, you can actually take the URLs generated by QGIS and use them in a web browser to retrieve the same images that QGIS uses internally. This can be useful for troubleshooting, as there are several brands of WMS server on the market and they all have their own interpretation of the WMS standard.

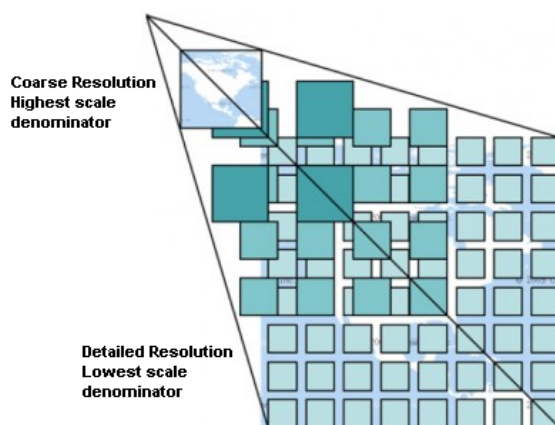
Lapis WMS dapat ditambahkan dengan sederhana, asalkan Anda tahu mengakses URL server WMS, Anda memiliki sambungan ke server, dan server memahami HTTP sebagai mekanisme transportasi data.

Sekilas Dukungan WMTS

QGIS can also act as a WMTS client. WMTS is an OGC standard for distributing tile sets of geospatial data. This is a faster and more efficient way of distributing data than WMS because with WMTS, the tile sets are pre-generated, and the client only requests the transmission of the tiles, not their production. A WMS request typically involves both the generation and transmission of the data. A well-known example of a non-OGC standard for viewing tiled geospatial data is Google Maps.

Dalam rangka untuk menampilkan data pada berbagai skala dengan apa yang pengguna inginkan, WMTS tile sets diproduksi pada beberapa tingkat skala yang berbeda dan dibuat tersedia untuk klien GIS untuk meminta mereka.

Diagram ini menggambarkan konsep tile sets:



Gambar 14.1: Konsep WMTS tile sets

The two types of WMTS interfaces that QGIS supports are via Key-Value-Pairs (KVP) and RESTful. These two interfaces are different, and you need to specify them to QGIS differently.

1) In order to access a **WMTS KVP** service, a QGIS user must open the WMS/WMTS interface and add the following string to the URL of the WMTS tile service:

```
"?SERVICE=WMTS&REQUEST=GetCapabilities"
```

An example of this type of address is

```
http://opencache.statkart.no/gatekeeper/gk/gk.open_wmts?\  
service=WMTS&request=GetCapabilities
```

For testing the topo2 layer in this WMTS works nicely. Adding this string indicates that a WMTS web service is to be used instead of a WMS service.

2. The **RESTful WMTS** service takes a different form, a straightforward URL. The format recommended by the OGC is:

```
{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 <http://maps.wien.gv.at/basemap/1.0.0/WMTSCapabilities.xml>.

Catatan: 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 http://wiki.osgeo.org/wiki/Tile_Map_Service_Specification for more information about this specification.

When you read WMTS, you can often think WMS-C also.

Memilih Server WMS/WMTS


The first time you use the WMS feature in QGIS, there are no servers defined.

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_1](#):

Nama	A name for this connection. This name will be used in the Server Connections drop-down box so that you can distinguish it from other WMS servers.
URL	URL of the server providing the data. This must be a resolvable host name – the same format as you would use to open a telnet connection or ping a host.
Nama pengguna	Username to access a secured WMS server. This parameter is optional.
Kata Sandi	Password for a basic authenticated WMS server. This parameter is optional.
Abaikan GetMap URI	<input checked="" type="checkbox"/> <i>Ignore GetMap URI reported in capabilities.</i> Use given URI from URL field above.
Abaikan GetFeatureInfo URI	<input checked="" type="checkbox"/> <i>Ignore GetFeatureInfo URI reported in capabilities.</i> Use given URI from URL field above.

Tabel OGC 1: Parameter Sambungan WMS

If you need to set up a proxy server to be able to receive WMS services from the internet, you can add your proxy server in the options. Choose *Settings* → *Options* and click on the *Network & Proxy* tab. There, you can add your proxy settings and enable them by setting *Use proxy for web access*. Make sure that you select the correct proxy type from the *Proxy type*  drop-down menu.

Once the new WMS server connection has been created, it will be preserved for future QGIS sessions.

Tip: Di URL Server WMS

Be sure, when entering the WMS server URL, that you have the base URL only. For example, you shouldn't have fragments such as `request=GetCapabilities` or `version=1.0.0` in your URL.

Memuat Lapis WMS/WMTS

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.

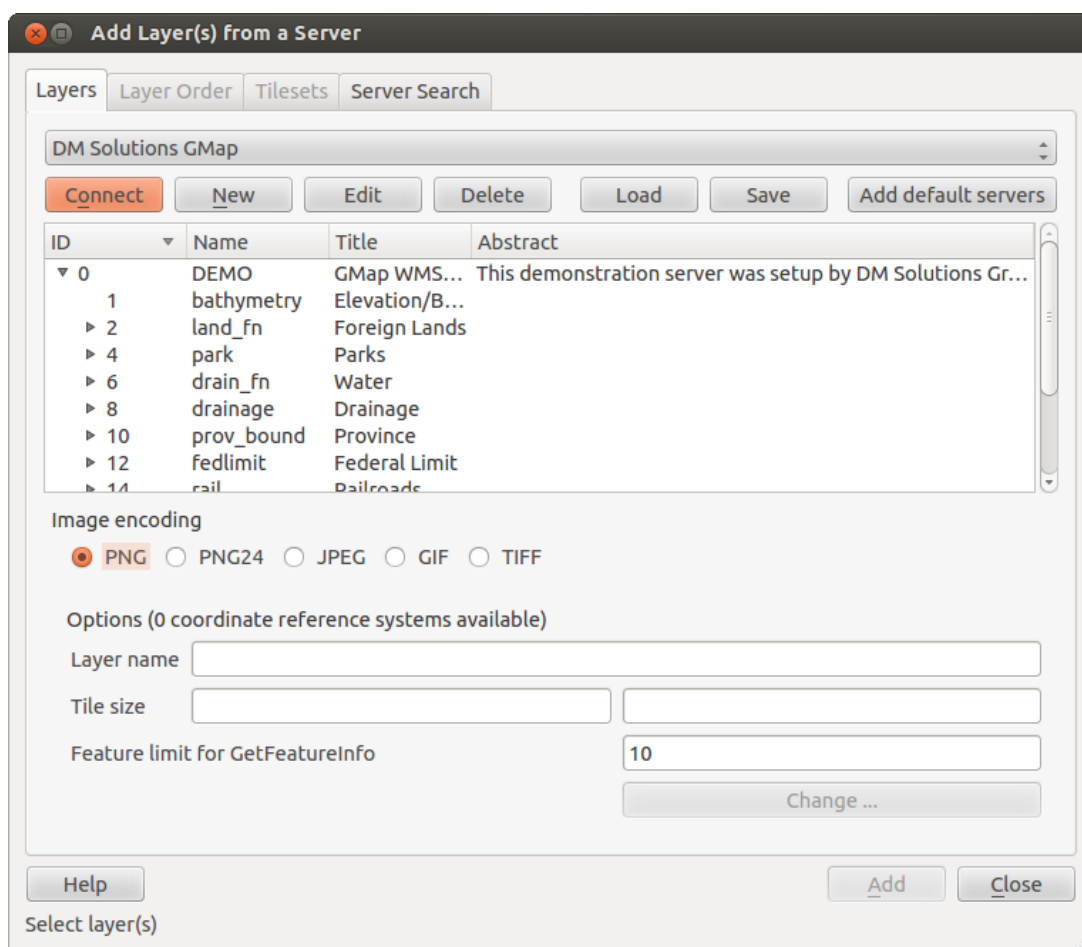
Layar Anda sekarang terlihat seperti [figure_OGR_1](#), yang menunjukkan respon yang diberikan oleh server DM Solutions Group WMS.

Pengkodean Gambar

The *Image encoding* section lists the formats that are supported by both the client and server. Choose one depending on your image accuracy requirements.

Tip: Pengkodean Gambar

Anda biasanya akan menemukan bahwa server WMS menawarkan pilihan pengkodean JPEG atau PNG. JPEG adalah format kompresi lossy, sedangkan PNG mereproduksi data raster mentah.



Gambar 14.2: Dialog menambahkan server WMS, menunjukkan lapis yang tersedia 🐧

Use JPEG if you expect the WMS data to be photographic in nature and/or you don't mind some loss in picture quality. This trade-off typically reduces by five times the data transfer requirement compared with PNG.

Use PNG if you want precise representations of the original data and you don't mind the increased data transfer requirements.

Opsi-Opsi

The Options area of the dialog provides a text field where you can add a *Layer name* for the WMS layer. This name will appear in the legend after loading the layer.

Below the layer name, you can define *Tile size* if you want to set tile sizes (e.g., 256x256) to split up the WMS request into multiple requests.

Feature limit for GetFeatureInfo mendefinisikan fitur apa dari server untuk query.

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.

Urutan Lapis

The *Layer Order* tab lists the selected layers available from the current connected WMS server. You may notice that some layers are expandable; this means that the layer can be displayed in a choice of image styles.

You can select several layers at once, but only one image style per layer. When several layers are selected, they will be combined at the WMS server and transmitted to QGIS in one go.

Tip: Mengurutkan Lapis WMS

WMS layers rendered by a server are overlaid in the order listed in the Layers section, from top to bottom of the list. If you want to change the overlay order, you can use the *Layer Order* tab.

Transparansi

In this version of QGIS, the *Global transparency* setting from the *Layer Properties* is hard coded to be always on, where available.

Tip: Transparansi Lapis WMS

Ketersediaan gambar WMS transparansi tergantung pada pengkodean gambar yang digunakan: PNG dan GIF didukung transparansi, JPEG sementara tidak didukung.

Sistem Referensi Koordinat

A coordinate reference system (CRS) is the OGC terminology for a QGIS projection.

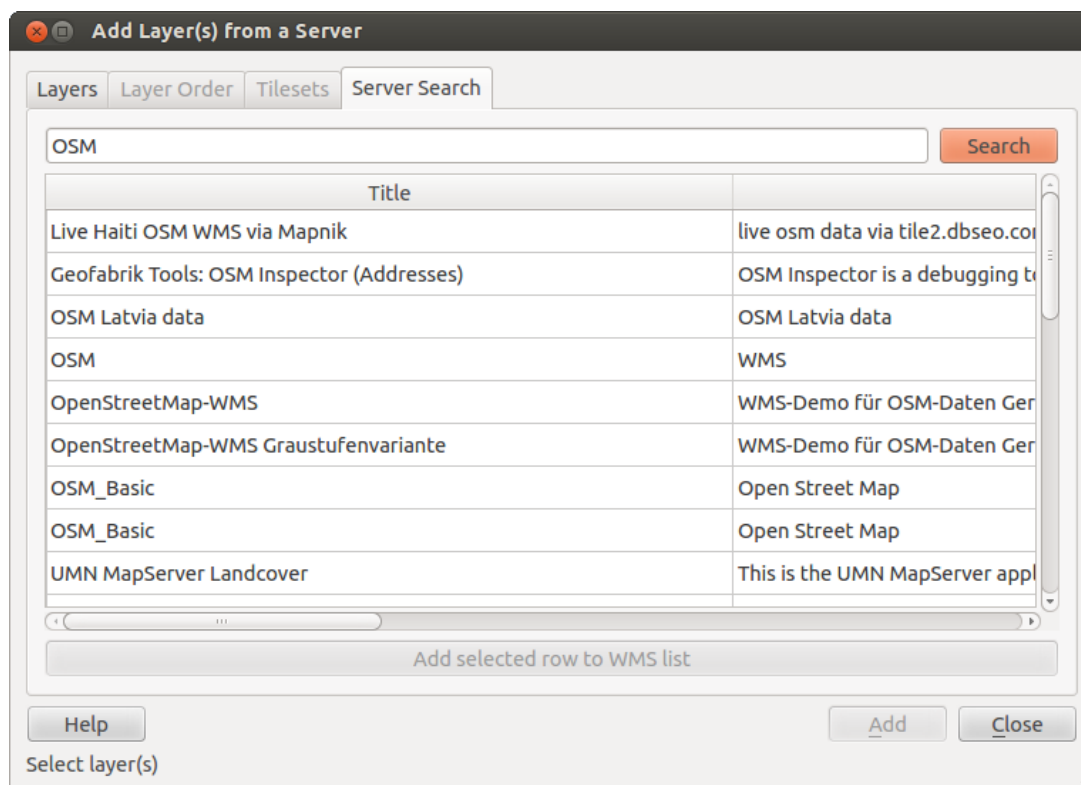
Each WMS layer can be presented in multiple CRSs, depending on the capability of the WMS server.

To choose a CRS, select **[Change...]** and a dialog similar to Figure Projection 3 in *Working with Projections* 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.

Mencari server

Within QGIS, you can search for WMS servers. [Figure_OGC_2](#) 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



Gambar 14.3: Dialog untuk mencari server WMS dengan beberapa kata kunci 🐧

of layers by clicking the **[Connect]** button. This option is quite handy when you want to search maps by specific keywords.

Basically, this option is a front end to the API of <http://geopole.org>.


Tileset

When using WMTS (Cached WMS) services like

```
http://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 *Settings* → *Panels* (KDE and Windows) or *View* → *Panels* (Gnome and MacOSX), then choosing *Tile scale*. This gives you the available scales from the tile server with a nice slider docked in.

Menggunakan Alat Identifikasi

Setelah Anda telah menambahkan server WMS, dan jika ada lapis dari server WMS adalah queryable, Anda kemudian dapat menggunakan alat  *Identify* untuk memilih pixel di peta kanvas. Sebuah query dibuat untuk server WMS untuk setiap pilihan yang dibuat. Hasil dari query dikembalikan dalam teks biasa. Format teks ini tergantung pada server WMS tertentu yang digunakan. **Pemilihan Format**

If multiple output formats are supported by the server, a combo box with supported formats is automatically added to the identify results dialog and the selected format may be stored in the project for the layer. **Dukungan format GML**

The  *Identify* tool supports WMS server response (GetFeatureInfo) in GML format (it is called Feature in the QGIS GUI in this context). If “Feature” format is supported by the server and selected, results of the Identify tool

are vector features, as from a regular vector layer. When a single feature is selected in the tree, it is highlighted in the map and it can be copied to the clipboard and pasted to another vector layer. See the example setup of the UMN Mapserver below to support GetFeatureInfo in GML format.

```
# 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"
```

Menampilkan Properti

Once you have added a WMS server, you can view its properties by right-clicking on it in the legend and selecting *Properties*. **Tab Metadata**

The tab *Metadata* displays a wealth of information about the WMS server, generally collected from the capabilities statement returned from that server. Many definitions can be gleaned by reading the WMS standards (see OPEN-GEOSPATIAL-CONSORTIUM in *Literatur dan Referensi Web*), but here are a few handy definitions:

- **Properti Server**

- **Versi WMS** — Versi WMS yang didukung oleh server.
- **Image Formats** — The list of MIME-types the server can respond with when drawing the map. QGIS supports whatever formats the underlying Qt libraries were built with, which is typically at least image/png and image/jpeg.
- **Identity Formats** — The list of MIME-types the server can respond with when you use the Identify tool. Currently, QGIS supports the `text-plain` type.

- **Properti Lapis**

- **Dipilih** — Apakah ada atau tidak lapis ini dipilih ketika server telah ditambahkan ke dalam proyek ini.
- **Visible** — Whether or not this layer is selected as visible in the legend (not yet used in this version of QGIS).
- **Bisa Diidentifikasi** — Apakah ada atau tidak lapis ini akan menghasilkan apa-apa ketika Mengidentifikasi alat yang digunakan di atasnya.
- **Can be Transparent** — Whether or not this layer can be rendered with transparency. This version of QGIS will always use transparency if this is `Yes` and the image encoding supports transparency.
- **Can Zoom In** — Whether or not this layer can be zoomed in by the server. This version of QGIS assumes all WMS layers have this set to `Yes`. Deficient layers may be rendered strangely.
- **Cascade Count** — WMS servers can act as a proxy to other WMS servers to get the raster data for a layer. This entry shows how many times the request for this layer is forwarded to peer WMS servers

for a result.

- **Fixed Width, Fixed Height** — Whether or not this layer has fixed source pixel dimensions. This version of QGIS assumes all WMS layers have this set to nothing. Deficient layers may be rendered strangely.
- **WGS 84 Bounding Box** — The bounding box of the layer, in WGS 84 coordinates. Some WMS servers do not set this correctly (e.g., UTM coordinates are used instead). If this is the case, then the initial view of this layer may be rendered with a very ‘zoomed-out’ appearance by QGIS. The WMS webmaster should be informed of this error, which they may know as the WMS XML elements `LatLonBoundingBox`, `EX_GeographicBoundingBox` or the `CRS:84 BoundingBox`.
- **Available in CRS** — The projections that this layer can be rendered in by the WMS server. These are listed in the WMS-native format.
- **Available in style** — The image styles that this layer can be rendered in by the WMS server.

Show WMS legend graphic in table of contents and composer

The QGIS WMS data provider is able to display a legend graphic in the table of contents’ layer list and in the map composer. 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.

If a legendGraphic is available, it is shown below the layer. It is little and you have to click on it to open it in real dimension (due to `QgsLegendInterface` architectural limitation). Clicking on the layer’s legend will open a frame with the legend at full resolution.


In the print composer, 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

The legend will display contextual information based on your current scale. The WMS legend will be shown only if the WMS server has `GetLegendGraphic` capability and the layer has `getCapability` url specified, so you have to select a styling.

Batasan Klien WMS

Not all possible WMS client functionality had been included in this version of QGIS. Some of the more noteworthy exceptions follow.

Pengaturan Mengedit Lapis WMS

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.

Server WMS Membutuhkan Otentikasi

Currently, publicly accessible and secured WMS services are supported. The secured WMS servers can be accessed by public authentication. You can add the (optional) credentials when you add a WMS server. See section *Memilih Server WMS/WMTS* for details.

Tip: Mengakses OGC-lapis dengan aman

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 <http://inteproxy.wald.intevation.org>.

Tip: QGIS WMS Mapserver

Since Version 1.7.0, QGIS has its own implementation of a WMS 1.3.0 Mapserver. Read more about this in chapter *QGIS sebagai OGC Data Server*.

14.1.2 Klien WCS



A Web Coverage Service (WCS) provides access to raster data in forms that are useful for client-side rendering, as input into scientific models, and for other clients. The WCS may be compared to the WFS and the WMS. As WMS and WFS service instances, a WCS allows clients to choose portions of a server's information holdings based on spatial constraints and other query criteria.

QGIS has a native WCS provider and supports both version 1.0 and 1.1 (which are significantly different), but currently it prefers 1.0, because 1.1 has many issues (i.e., each server implements it in a different way with various particularities).

The native WCS provider handles all network requests and uses all standard QGIS network settings (especially proxy). It is also possible to select cache mode ('always cache', 'prefer cache', 'prefer network', 'always network'), and the provider also supports selection of time position, if temporal domain is offered by the server.



14.1.3 Klien WFS dan WFS-T

In QGIS, a WFS layer behaves pretty much like any other vector layer. You can identify and select features, and view the attribute table. Since QGIS 1.6, editing WFS-T is also supported.

In general, adding a WFS layer is very similar to the procedure used with WMS. The difference is that there are no default servers defined, so we have to add our own.

Memuat Lapis WFS

As an example, we use the DM Solutions WFS server and display a layer. The URL is: http://www2.dmsolutions.ca/cgi-bin/mswfs_gmap

1. Click on the  Add WFS Layer tool on the Layers toolbar. The *Add WFS Layer from a Server* dialog appears.
2. Klik [**Baru**].
3. Masukkan 'DM Solutions' sebagai nama.
4. Masukkan URL (lihat di atas).
5. Klik [**OK**].
6. Pilih 'DM Solutions' dari daftar drop-down *Sambungan Server* .
7. Klik [**Sambung**].
8. Wait for the list of layers to be populated.
9. Select the *Parks* layer in the list.
10. Klik [**Terapkan**] untuk menambahkan lapis ke peta.

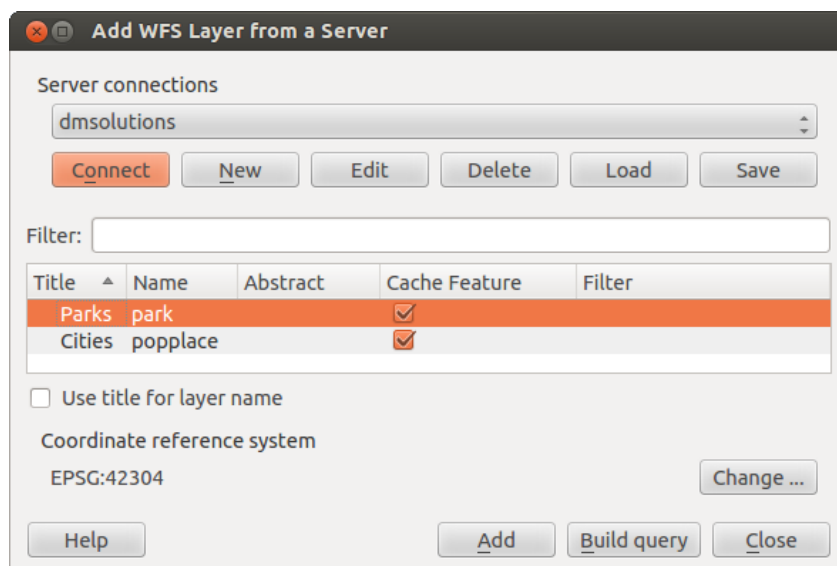
Note that any proxy settings you may have set in your preferences are also recognized.


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 province or two and view the attribute table.

Only WFS 1.0.0 is supported. At this time, there have not been many tests against WFS versions implemented in other WFS servers. If you encounter problems with any other WFS server, please do not hesitate to contact the development team. Please refer to section *Bantuan dan Dukungan* for further information about the mailing lists.

Tip: Mencari Server WFS

You can find additional WFS servers by using Google or your favorite search engine. There are a number of lists with public URLs, some of them maintained and some not.



Gambar 14.4: Menambahkan lapis WFS 

14.2 QGIS sebagai OGC Data Server

Server QGIS merupakan open source WMS 1.3, WFS 1.0.0 dan WCS 1 1.1.1 implementasi yang, di samping itu, mengimplementasikan fitur kartografi canggih untuk pemetaan tematik. Server QGIS adalah aplikasi FastCGI/CGI (Common Gateway Interface) ditulis dalam C++ yang bekerja sama dengan web server (misalnya Apache, Lighttpd). Hal ini didanai oleh proyek Uni Eropa Orchestra, Sany dan kota Uster di Swiss.

Server QGIS digunakan QGIS sebagai back end untuk logika GIS dan peta rendering. Selain itu, perpustakaan Qt digunakan untuk grafis dan pemrograman C++ platform-independen. Berbeda dengan perangkat lunak WMS lain, Server QGIS menggunakan aturan kartografi sebagai bahasa konfigurasi, baik untuk konfigurasi server dan aturan kartografi yang ditetapkan pengguna.

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.

Dalam salah satu buku pedoman berikut, kami akan memberikan contoh konfigurasi untuk membangun sebuah Server QGIS. Untuk saat ini, kami sarankan untuk membaca salah satu URL berikut ini untuk mendapatkan informasi lebih lanjut:

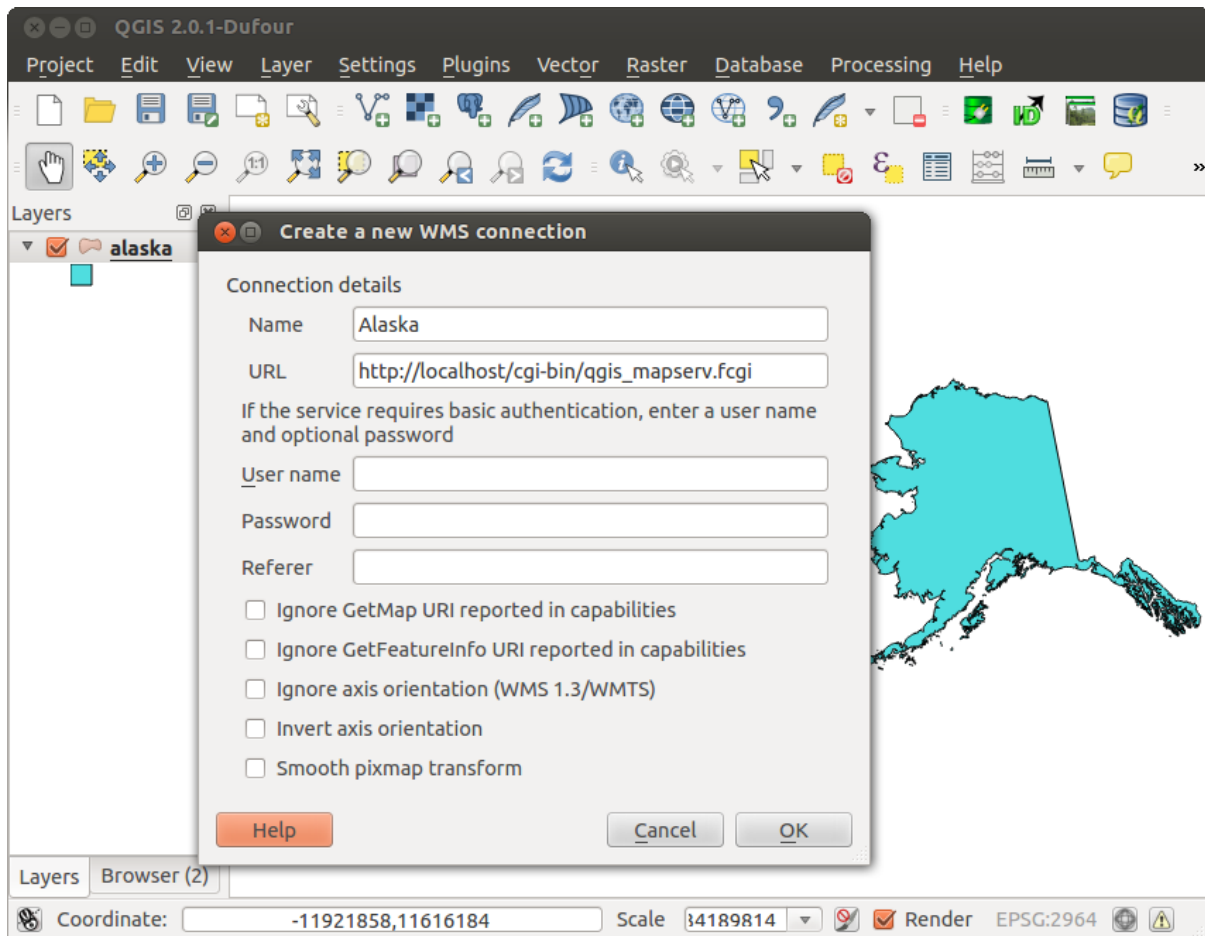
- http://karlinapp.ethz.ch/qgis_wms/
- http://hub.qgis.org/projects/quantum-gis/wiki/QGIS_Server_Tutorial
- <http://infiniti.com/2010/08/qgis-mapserver-a-wms-server-for-the-masses/>

14.2.1 Contoh Pemasangan pada Debian Squeeze

Pada titik ini, kami akan memberikan contoh instalasi singkat dan sederhana bagaimana caranya untuk Debian Squeeze. Banyak OS lain menyediakan paket untuk Server QGIS, juga. Jika Anda harus membangun semuanya dari sumber, silakan lihat URL di atas.

Terlepas dari QGIS dan Server QGIS, Anda memerlukan server web, dalam kasus apache2 kami. Anda dapat menginstal semua paket dengan `aptitude` atau `apt-get install` bersama-sama dengan paket ketergantungan yang diperlukan lainnya. Setelah instalasi, Anda harus menguji mengkonfirmasi bahwa server web dan Server QGIS bekerja seperti yang diharapkan. Pastikan server apache berjalan dengan `/etc/init.d/apache2 start`. Buka peramban web dan ketik URL: `http://localhost`. Jika apache sudah habis, Anda akan melihat pesan 'It works!'.

Sekarang coba pemasangan Server QGIS. `qgis_mapserv.fcgi` tersedia di `/usr/lib/cgi-bin/qgis_mapserv.fcgi` dan mendukung standar WMS yang menunjukkan batas-batas negara bagian Alaska. Tambahkan WMS dengan URL `http://localhost/cgi-bin/qgis_mapserv.fcgi` seperti yang dijelaskan dalam *Memilih Server WMS/WMTS*.




Gambar 14.5: WMS standar dengan batas-batas Amerika Serikat termasuk dalam QGIS Server (KDE) 

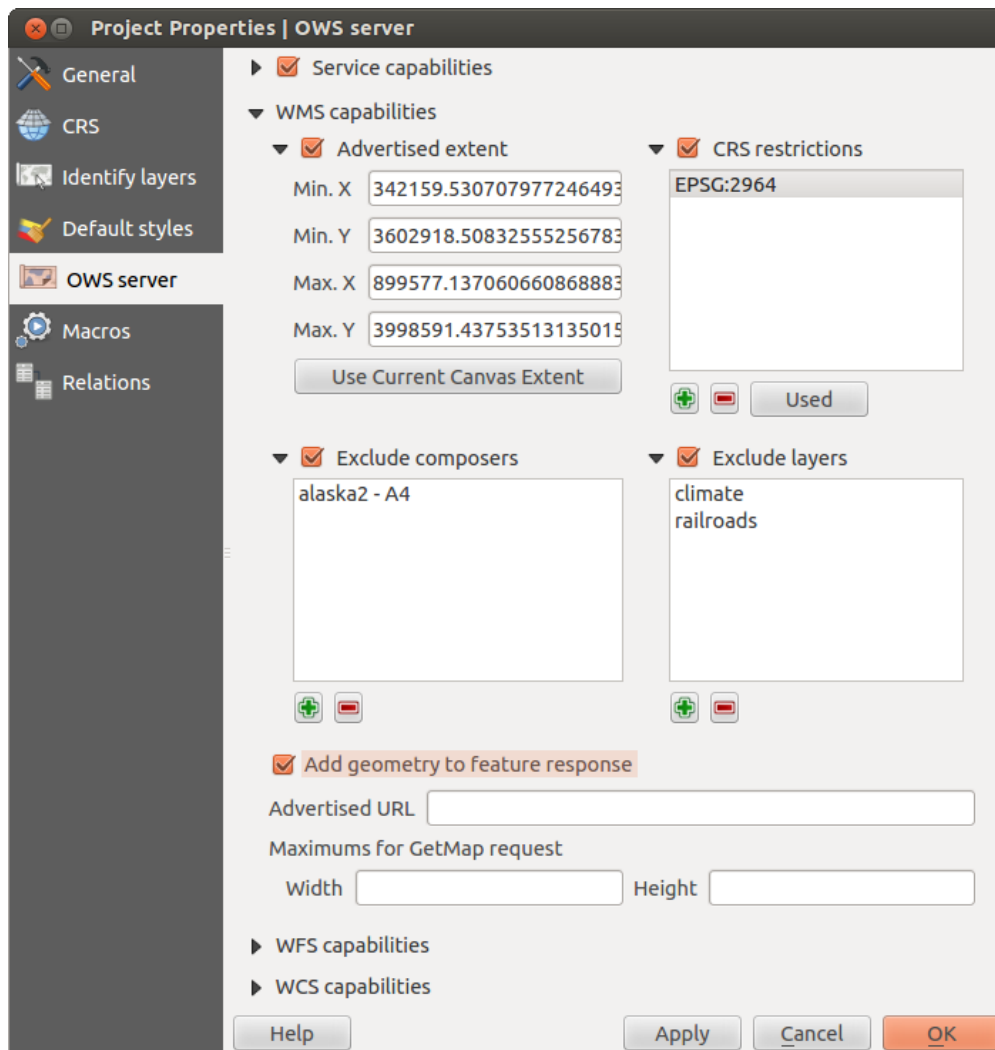
14.2.2 Membuat WMS/WFS/WCS dari proyek QGIS

Untuk memberikan Server QGIS baru WMS, WFS atau WCS, kita harus membuat berkas proyek QGIS dengan beberapa data. Di sini, kita menggunakan 'Alaska' shapefile dari contoh dataset QGIS. Tentukan warna dan gaya dari lapis dalam QGIS dan proyek CRS, jika belum ditetapkan.

Kemudian, ke menu *Server OWS* dialog *Proyek* → *Proyek Properti* dan memberikan beberapa informasi tentang OWS di kolom di bawah *Kemampuan Layanan*. Ini akan muncul di *GetCapabilities* respon dari WMS, WFS atau WCS. Jika Anda tidak mencentang *Kemampuan Layanan*, Server QGIS akan menggunakan informasi yang diberikan dalam berkas `wms_metadata.xml` yang terletak di folder `cgi-bin`.


Kemampuan WMS


Di bagian *Kemampuan WMS*, Anda dapat menentukan tingkat diiklankan di respon WMS *GetCapabilities* dengan memasukkan nilai minimum dan maksimum X dan Y dalam kolom di bawah *Batas diiklankan*. Klik *Gunakan Batas Kanvas Sekarang* menetapkan nilai-nilai ini sejauh yang sedang ditampilkan dalam kanvas peta QGIS. Dengan mencentang *Pembatasan CRS*, Anda dapat membatasi di mana sistem koordinat referensi (CRS) Server QGIS akan menawarkan untuk membuat peta. Gunakan tombol  di bawah untuk memilih CRS dari



Gambar 14.6: Definisi QGIS Server WMS/WFS/WCS proyek (KDE)

Coordinate Reference System Selector, atau klik *Digunakan* untuk menambahkan CRS digunakan proyek QGIS ke dalam daftar.

Jika Anda memiliki penyusun cetak didefinisikan dalam proyek Anda, mereka akan tercantum dalam GetCapabilities respon, dan mereka dapat digunakan oleh permintaan GetPrint untuk membuat cetakan, menggunakan salah satu tata letak penyusun cetak sebagai template. Ini adalah ekstensi-khusus QGIS untuk spesifikasi WMS 1.3.0. Jika Anda ingin mengecualikan penyusun cetak dari yang diterbitkan oleh WMS, centang *Kecuali Penyusun* dan klik tombol dibawah . Kemudian, pilih penyusun cetak dari dialog *Pilih penyusun cetak* untuk menambahkannya ke daftar penyusun yang dikecualikan.

Jika Anda ingin mengecualikan setiap lapis atau grup lapis dari yang diterbitkan oleh WMS, centang *Kecuali Lapis* dan klik menu di bawah . Akan membuka dialog *Pilih lapis dan grup dibatasi*, yang memungkinkan Anda untuk memilih lapis dan grup yang tidak ingin dipublikasikan. Gunakan tombol Shift atau Ctrl jika Anda ingin memilih beberapa entri sekaligus.

Anda dapat menerima diminta GetFeatureInfo sebagai teks biasa, XML dan GML. Default adalah XML, teks atau format GML tergantung format keluaran dipilih untuk permintaan GetFeatureInfo.

Jika Anda ingin, Anda dapat mencentang *Tambahkan geometri untuk fitur respon*. Ini akan mencakup dalam respon GetFeatureInfo fitur geometri dalam format teks. Jika Anda ingin QGIS Server QGIS mengiklankan URL permintaan khusus dalam respon WMS GetCapabilities, masukkan URL yang sesuai dalam kolom *Iklankan URL*. Selanjutnya, Anda dapat membatasi ukuran maksimum peta dikembalikan oleh permintaan GetMap dengan memasukkan lebar, tinggi dalam kolom masing-masing di bawah *Maksimal untuk permintaan GetMap*.

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 his attributes, also this information will be shown in the GetFeatureInfo output.

Kemampuan WFS

dalam area *Kemampuan WFS*, Anda dapat memilih lapis yang ingin Anda publikasikan sebagai WFS, dan menentukan apakah mereka akan memungkinkan memperbarui, memasukkan dan menghapus operasi. Jika Anda memasukkan URL di kolom *Iklankan URL* dari bagian *Kemampuan WFS*, Server QGIS akan mengiklankan URL tertentu dalam respon WFS GetCapabilities.

Kemampuan WCS

dalam area *Kemampuan WCS*, Anda dapat memilih lapis yang ingin Anda publikasikan sebagai WCS. Jika Anda memasukkan URL di kolom *Iklankan URL* dari bagian *Kemampuan WCS*, Server QGIS akan mengiklankan URL tertentu dalam respon WCS GetCapabilities.

Sekarang, simpan sesi ke dalam berkas proyek `alaska.qgs`. Untuk memberikan proyek sebagai WMS / WFS, kita membuat folder baru `/usr/lib/cgi-bin/project` dengan hak istimewa admin dan menambahkan berkas proyek `alaska.qgs` dan salinan berkas `qgis_mapserv.fcgi` - itu saja.

Sekarang coba proyek kita WMS, WFS dan WCS. Tambahkan WMS, WFS dan WCS seperti yang dijelaskan dalam *Memuat Lapis WMS/WMTS, Klien WFS dan WFS-T* dan *Klien WCS* ke QGIS dan muat data. URLnya adalah:

```
http://localhost/cgi-bin/project/qgis_mapserv.fcgi
```

Menyetel baik OWS Anda

Untuk lapis vektor, menu *Kolom* dari dialog *Lapis* → *Properti* memungkinkan Anda untuk menentukan setiap atribut jika akan diterbitkan atau tidak. Secara default, semua atribut yang diterbitkan oleh WMS dan WFS. Jika Anda ingin atribut tertentu tidak akan diterbitkan, hapus centang pada kotak centang yang sesuai dalam kolom *WMS* atau *WFS*.

Anda dapat melapisi tanda air pada peta yang diproduksi oleh WMS Anda dengan menambahkan penjelasan teks atau anotasi SVG ke berkas proyek. Lihat bagian Alat Anotasi di *Peralatan Umum* untuk petunjuk membuat anotasi. Untuk anotasi yang akan ditampilkan sebagai tanda air pada keluaran WMS, kotak centang *Posisi peta*

ditetapkan dalam dialog *Anotasi teks* harus dilepas centangnya. Hal ini dapat diakses dengan mengklik ganda anotasi sementara salah satu alat anotasi aktif. Untuk anotasi SVG, Anda akan membutuhkan mengatur proyek menyimpan path absolut (dalam menu *Umum* dari dialog *Proyek* → *Proyek Properti*) atau secara manual mengubah path ke gambar SVG dengan cara itu, path relatif valid.

Parameter tambahan yang didukung oleh permintaan WMS GetMap

Dalam permintaan WMS GetMap, Server QGIS menerima beberapa parameter tambahan di samping parameter standar sesuai dengan spesifikasi OGC WMS 1.3.0:

- Parameter **MAP**: Mirip dengan MapServer, *parameter* 'MAP' dapat digunakan untuk menentukan path ke berkas proyek QGIS. Anda dapat menentukan path absolut atau path relatif ke lokasi server executable (`qgis_mapserv.fcgi`). Jika tidak ditentukan, pencarian server QGIS untuk berkas `.qgs` dalam direktori di mana server executable berada.

Contoh:

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?\
REQUEST=GetMap&MAP=/home/qgis/mymap.qgs&...
```

- Parameter **DPI**: Parameter DPI dapat digunakan untuk menentukan resolusi keluaran yang diminta.

Contoh:

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?REQUEST=GetMap&DPI=300&...
```

- Parameter **Kekeruhan**: Opacity dapat diatur pada lapis atau tingkat grup. Nilai diperbolehkan berkisar dari 0 (sepenuhnya transparan) sampai 255 (sepenuhnya keruh).

Contoh:

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?\
REQUEST=GetMap&LAYERS=mylayer1,mylayer2&OPACITIES=125,200&...
```

QGIS Server logging

To log requests send to server, set the following environment variables:

- **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_LEVEL**: Specify desired log level. Available values are:
 - 0 INFO (log all requests),
 - 1 WARNING,
 - 2 CRITICAL (log just critical errors, suitable for production purposes).

Contoh:

```
SetEnv QGIS_SERVER_LOG_FILE /var/tmp/qgislog.txt
SetEnv QGIS_SERVER_LOG_LEVEL 0
```

Note

- When using Fcgid module use `FcgidInitialEnv` instead of `SetEnv`!
- Server logging is enabled also if executable is compiled in release mode.

Environment variables

- **QGIS_OPTIONS_PATH**: The variable specifies path to directory with settings. It works the same ways as QGIS application `-optionspath` option. It is looking for settings file in `<QGIS_OPTIONS_PATH>/QGIS/QGIS2.ini`. For exaple, to set QGIS server on Apache to use `/path/to/config/QGIS/QGIS2.ini` settings file, add to Apache config:

```
SetEnv QGIS_OPTIONS_PATH "/path/to/config/"
```

Pekerjaan dengan Data GPS


15.1 GPS Plugin



15.1.1 What is GPS?

GPS, the Global Positioning System, is a satellite-based system that allows anyone with a GPS receiver to find their exact position anywhere in the world. GPS is used as an aid in navigation, for example in airplanes, in boats and by hikers. The GPS receiver uses the signals from the satellites to calculate its latitude, longitude and (sometimes) elevation. Most receivers also have the capability to store locations (known as **waypoints**), sequences of locations that make up a planned **route** and a tracklog or **track** of the receiver's movement over time. Waypoints, routes and tracks are the three basic feature types in GPS data. QGIS displays waypoints in point layers, while routes and tracks are displayed in linestring layers.


15.1.2 Loading GPS data from a file

There are dozens of different file formats for storing GPS data. The format that QGIS uses is called GPX (GPS eXchange format), which is a standard interchange format that can contain any number of waypoints, routes and tracks in the same file.

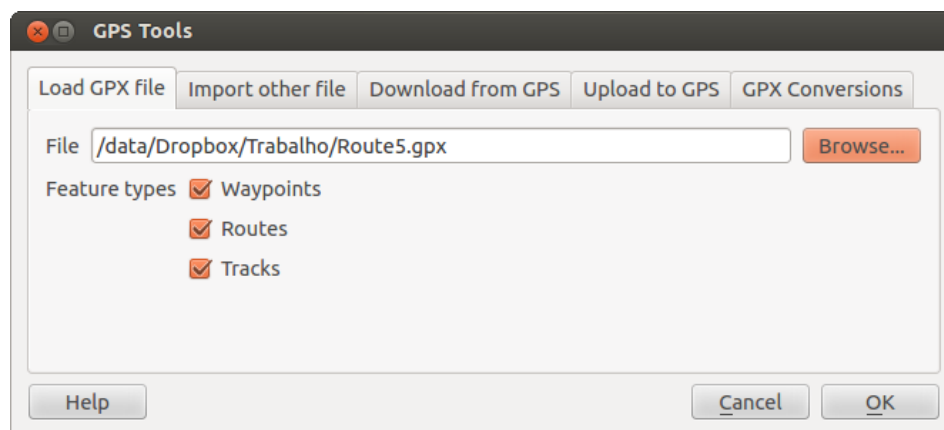
To load a GPX file, you first need to load the plugin. *Plugins* →  *Plugin Manager...* opens the Plugin Manager Dialog. Activate the *GPS Tools* checkbox. When this plugin is loaded, two buttons with a small handheld GPS device will show up in the toolbar:

-  Create new GPX Layer
-  GPS Tools

For working with GPS data, we provide an example GPX file available in the QGIS sample dataset: `qgis_sample_data/gps/national_monuments.gpx`. See section *Contoh data* for more information about the sample data.

1. Select *Vector* → *GPS* → *GPS Tools* or click the  *GPS Tools* icon in the toolbar and open the *Load GPX file* tab (see *figure_GPS_1*).
2. Browse to the folder `qgis_sample_data/gps/`, select the GPX file `national_monuments.gpx` and click **[Open]**.

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.



Gambar 15.1: The *GPS Tools* dialog window 

Catatan: 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 <http://www.topografix.com/GPX/1/1/>.

15.1.3 GPSTabel

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 GPSTabel, which is available at <http://www.gpsbabel.org>. This program can also transfer GPS data between your computer and a GPS device. QGIS uses GPSTabel 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 GPSTabel is known to work with QGIS, but you should be able to use later versions without any problems.

15.1.4 Importing GPS data



To import GPS data from a file that is not a GPX file, you use the tool *Import other file* in the GPS Tools dialog. Here, you select the file that you want to import (and the file type), which feature type you want to import from it, where you want to store the converted GPX file and what the name of the new layer should be. Note that not all GPS data formats will support all three feature types, so for many formats you will only be able to choose between one or two types.

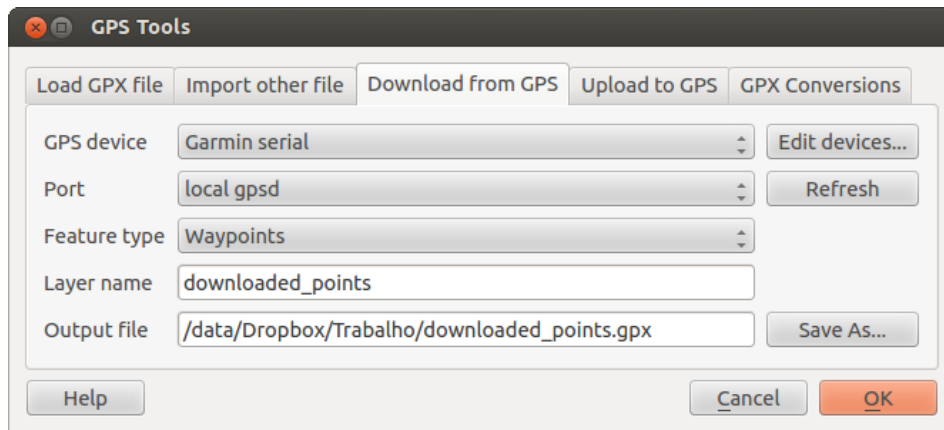
15.1.5 Downloading GPS data from a device

QGIS can use GPSTabel 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_2](#)). 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.

The device type you select in the GPS device menu determines how GPSTabel tries to communicate with your GPS device. If none of the available types work with your GPS device, you can create a new type (see section [Defining new device types](#)).

The port may be a file name or some other name that your operating system uses as a reference to the physical port in your computer that the GPS device is connected to. It may also be simply USB, for USB-enabled GPS units.

-  On Linux, this is something like `/dev/ttyS0` or `/dev/ttyS1`.
-  On Windows, it is COM1 or COM2.



Gambar 15.2: The download tool

When you click [OK], the data will be downloaded from the device and appear as a layer in QGIS.

15.1.6 Uploading GPS data to a device

You can also upload data directly from a vector layer in QGIS to a GPS device using the *Upload to GPS* tab of the GPS Tools dialog. To do this, you simply select the layer that you want to upload (which must be a GPX layer), your GPS device type, and the port (or USB) that it is connected to. Just as with the download tool, you can specify new device types if your device isn't in the list.

This tool is very useful in combination with the vector-editing capabilities of QGIS. It allows you to load a map, create waypoints and routes, and then upload them and use them on your GPS device.

15.1.7 Defining new device types

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 will be replaced by -w if you are downloading waypoints, -r if you are downloading routes and -t if you are downloading tracks. These are command-line options that tell GPSBabel which feature type to download.

%in will be replaced by the port name that you choose in the download window and %out will be replaced by the name you choose for the GPX file that the downloaded data should be stored in. So, if you create a device type with the download command `gpsbabel %type -i garmin -o gpx %in %out` (this is actually the download command for the predefined device type 'Garmin serial') and then use it to download waypoints from port /dev/ttyS0 to the file output.gpx, QGIS will replace the keywords and run the command `gpsbabel -w -i garmin -o gpx /dev/ttyS0 output.gpx`.

The upload command is the command that is used to upload data to the device. The same keywords are used, but %in is now replaced by the name of the GPX file for the layer that is being uploaded, and %out is replaced by the port name.

You can learn more about GPSBabel and its available command line options at <http://www.gpsbabel.org>.

Once you have created a new device type, it will appear in the device lists for the download and upload tools.

15.1.8 Download of points/tracks from GPS units

As described in previous sections QGIS uses GPSTools 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 http://www8.garmin.com/support/download_details.jsp?id=591

Connect the unit. Open GPS Tools and use `type=garmin serial` and `port=usb:`. Fill the fields *Layer name* and *Output file*. Sometimes it seems to have problems saving in a certain folder, using something like `c:\temp` usually works.

Ubuntu/Mint GNU/Linux

It is first needed an issue about the permissions of the device, as described at https://wiki.openstreetmap.org/wiki/USB_Garmin_on_GNU/Linux. You can try to create a file `/etc/udev/rules.d/51-garmin.rules` containing this rule

```
ATTRS{idVendor}=="091e", ATTRS{idProduct}=="0003", MODE="666"
```

After that is necessary to be sure that the `garmin_gps` kernel module is not loaded

```
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 datalogger (only Bluetooth)

MS Windows

The already referred bug does not allow to download the data from within QGIS, so it is needed to use GPSTools from the command line or using its interface. The working command is

```
gpsbabel -t -i skytraq,baud=9600,initbaud=9600 -f COM9 -o gpx -F C:/GPX/aaa.gpx
```

Ubuntu/Mint GNU/Linux

Use same command (or settings if you use GPSTools GUI) as in Windows. On Linux it maybe somehow common to get a message like

```
skytraq: Too many read errors on serial port
```

it is just a matter to turn off and on the datalogger and try again.

BlueMax GPS-4044 datalogger (both BT and USB)

MS Windows

Catatan: It needs to install its drivers before using it on Windows 7. See in the manufacturer site for the proper download.

Downloading with GPSTools, both with USB and BT returns always an error like

```
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

With USB

After having connected the cable use the `dmesg` command to understand what port is being used, for example `/dev/ttyACM3`. Then as usual use `GPSTabel` from the CLI or GUI


```
gpsbabel -t -i mtk -f /dev/ttyACM3 -o gpx -F /home/user/bluemax.gpx
```

With Bluetooth





Use `Blueman Device Manager` to pair the device and make it available through a system port, then run `GPSTabel`

```
gpsbabel -t -i mtk -f /dev/rfcomm0 -o gpx -F /home/user/bluemax_bt.gpx
```

15.2 Live GPS tracking

To activate live GPS tracking in QGIS, you need to select *Settings* → *Panels*  *GPS information*. You will get a new docked window on the left side of the canvas.


There are four possible screens in this GPS tracking window:

-  GPS position coordinates and an interface for manually entering vertices and features
-  GPS signal strength of satellite connections
-  GPS polar screen showing number and polar position of satellites
-  GPS options screen (see [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.

Peringatan: If you want to record your position to the canvas, you have to create a new vector layer first and switch it to editable status to be able to record your track.


15.2.1 Position and additional attributes

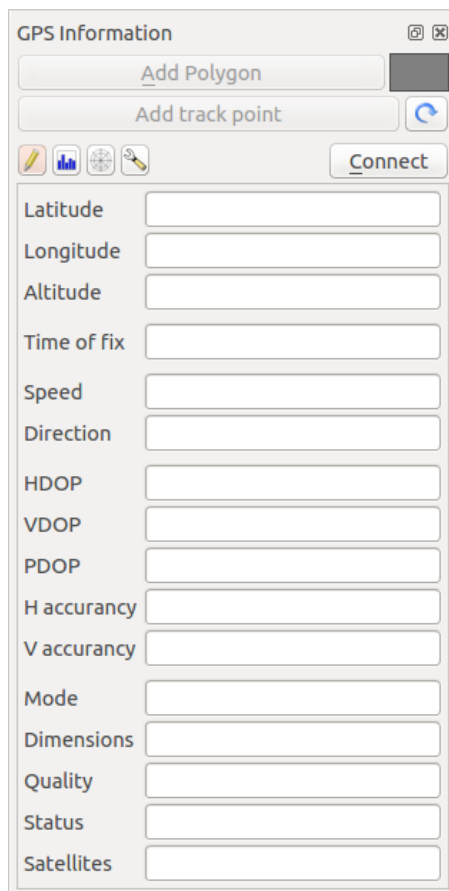
 If the GPS is receiving signals from satellites, you will see your position in latitude, longitude and altitude together with additional attributes.

15.2.2 GPS signal strength

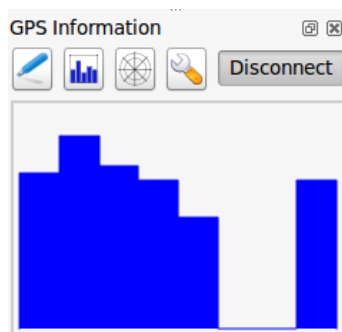
 Here, you can see the signal strength of the satellites you are receiving signals from.

15.2.3 GPS polar window

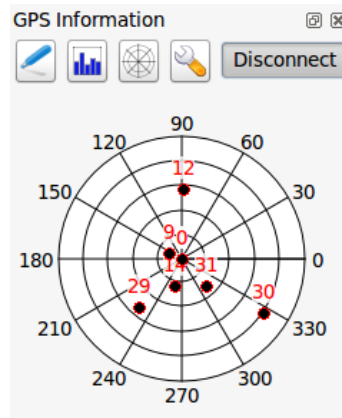
 If you want to know where in the sky all the connected satellites are, you have to switch to the polar screen. You can also see the ID numbers of the satellites you are receiving signals from.




Gambar 15.3: GPS tracking position and additional attributes 🐧




Gambar 15.4: GPS tracking signal strength 🐧



Gambar 15.5: GPS tracking polar window 

15.2.4 GPS options

 In case of connection problems, you can switch between:

- *Autodetect*
- *Internal*
- *Serial device*
- *gpsd* (selecting the Host, Port and Device your GPS is connected to)


A click on [**Connect**] again initiates the connection to the GPS receiver.

You can activate *Automatically save added features* when you are in editing mode. Or you can activate *Automatically add points* to the map canvas with a certain width and color.

Activating *Cursor*, you can use a slider  to shrink and grow the position cursor on the canvas.

Activating *Map centering* allows you to decide in which way the canvas will be updated. This includes 'always', 'when leaving', if your recorded coordinates start to move out of the canvas, or 'never', to keep map extent.

Finally, you can activate *Log file* and define a path and a file where log messages about the GPS tracking are logged.

If you want to set a feature manually, you have to go back to  *Position* and click on [**Add Point**] or [**Add track point**].

15.2.5 Connect to a Bluetooth GPS for live tracking

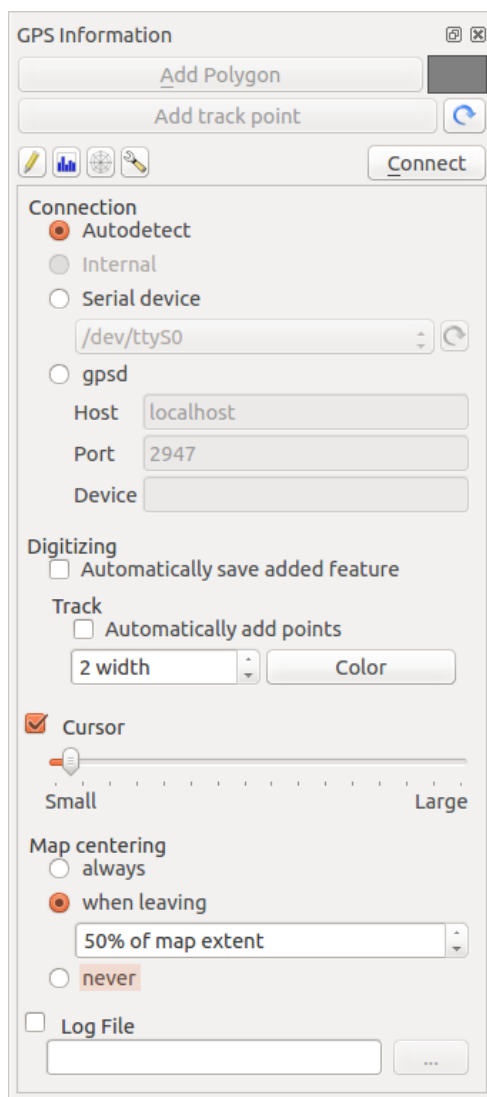
With QGIS you can connect a Bluetooth GPS for field data collection. To perform this task you need a GPS Bluetooth device and a Bluetooth receiver on your computer.

At first you must let your GPS device be recognized and paired to the computer. Turn on the GPS, go to the Bluetooth icon on your notification area and search for a New Device.


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.

Remember the number of the COM port assigned to the GPS connection as resulting by the Bluetooth properties.

After the GPS has been recognized, make the pairing for the connection. Usually the authorization code is 0000.



Gambar 15.6: GPS tracking options window 🐧


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.

If QGIS can't receive GPS data, then you should restart your GPS device, wait 5-10 seconds then try to connect again. Usually this solution work. If you receive again a connection error make sure you don't have another Bluetooth receiver near you, paired with the same GPS unit.

15.2.6 Using GPSMAP 60cs

MS Windows

Easiest way to make it work is to use a middleware (freeware, not open) called **GPSSGate**.

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

As for Windows the easiest way is to use a server in the middle, in this case **GPSD**, so

```
sudo apt-get install gpsd
```

Then load the `garmin_gps` kernel module

```
sudo modprobe garmin_gps
```

And then connect the unit. Then check with `dmesg` the actual device being used by the unit, for example `/dev/ttyUSB0`. Now you can launch `gpsd`

```
gpsd /dev/ttyUSB0
```


And finally connect with the QGIS live tracking tool.

15.2.7 Using BTGP-38KM datalogger (only Bluetooth)

Using **GPSD** (under Linux) or **GPSSGate** (under Windows) is effortless.

15.2.8 Using BlueMax GPS-4044 datalogger (both BT and USB)

MS Windows

The live tracking works for both USB and BT modes, by using **GPSSGate** or even without it, just use the  *Autodetect* mode, or point the tool the right port.

Ubuntu/Mint GNU/Linux

For USB

The live tracking works both with **GPSD**

```
gpsd /dev/ttyACM3
```

or without it, by connecting the QGIS live tracking tool directly to the device (for example `/dev/ttyACM3`).

For Bluetooth

The live tracking works both with **GPSD**

`gpsd /dev/rfcomm0`











or without it, by connecting the QGIS live tracking tool directly to the device (for example `/dev/rfcomm0`).

.


GRASS GIS Integration

The GRASS plugin provides access to GRASS GIS databases and functionalities (see GRASS-PROJECT in *Literatur dan Referensi Web*). This includes visualizing GRASS raster and vector layers, digitizing vector layers, editing vector attributes, creating new vector layers and analysing GRASS 2-D and 3-D data with more than 400 GRASS modules.

In this section, we'll introduce the plugin functionalities and give some examples of managing and working with GRASS data. The following main features are provided with the toolbar menu when you start the GRASS plugin, as described in section [sec_starting_grass](#):

-  Open mapset
-  New mapset
-  Close mapset
-  Add GRASS vector layer
-  Add GRASS raster layer
-  Create new GRASS vector
-  Edit GRASS vector layer
-  Open GRASS tools
-  Display current GRASS region
-  Edit current GRASS region







16.1 Starting the GRASS plugin

To use GRASS functionalities and/or visualize GRASS vector and raster layers in QGIS, you must select and load the GRASS plugin with the Plugin Manager. Therefore, go to the menu *Plugins* →  *Manage Plugins*, select *GRASS* and click [OK].

You can now start loading raster and vector layers from an existing GRASS LOCATION (see section [sec_load_grassdata](#)). Or, you can create a new GRASS LOCATION with QGIS (see section [Creating a new GRASS LOCATION](#)) and import some raster and vector data (see section [Importing data into a GRASS LOCATION](#)) for further analysis with the GRASS Toolbox (see section [The GRASS Toolbox](#)).

16.2 Loading GRASS raster and vector layers

With the GRASS plugin, you can load vector or raster layers using the appropriate button on the toolbar menu. As an example, we will use the QGIS Alaska dataset (see section *Contoh data*). It includes a small sample GRASS LOCATION with three vector layers and one raster elevation map.

1. Create a new folder called `grassdata`, download the QGIS 'Alaska' dataset `qgis_sample_data.zip` from <http://download.osgeo.org/qgis/data/> and unzip the file into `grassdata`.
2. Start QGIS.
3. If not already done in a previous QGIS session, load the GRASS plugin clicking on *Plugins* →  *Manage Plugins* and activate *GRASS*. The GRASS toolbar appears in the QGIS main window.
4. In the GRASS toolbar, click the  *Open mapset* icon to bring up the *MAPSET* wizard.
5. For `Gisdbase`, browse and select or enter the path to the newly created folder `grassdata`.
6. You should now be able to select the *LOCATION*  `alaska` and the *MAPSET*  `demo`.
7. Click **[OK]**. Notice that some previously disabled tools in the GRASS toolbar are now enabled.
8. Click on  *Add GRASS raster layer*, choose the map name `gtopo30` and click **[OK]**. The elevation layer will be visualized.
9. Click on  *Add GRASS vector layer*, choose the map name `alaska` and click **[OK]**. The Alaska boundary vector layer will be overlaid on top of the `gtopo30` map. You can now adapt the layer properties as described in chapter *The Vector Properties Dialog* (e.g., change opacity, fill and outline color).
10. Also load the other two vector layers, `rivers` and `airports`, and adapt their properties.

As you see, it is very simple to load GRASS raster and vector layers in QGIS. See the following sections for editing GRASS data and creating a new LOCATION. More sample GRASS LOCATIONS are available at the GRASS website at <http://grass.osgeo.org/download/sample-data/>.

Tip: GRASS Data Loading

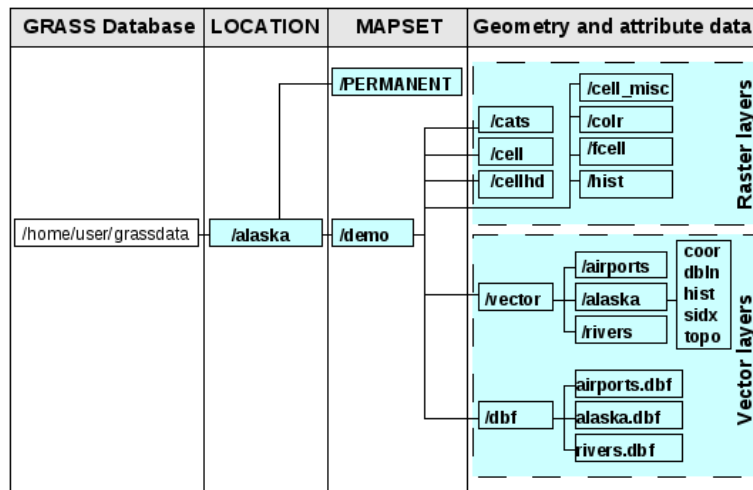
If you have problems loading data or QGIS terminates abnormally, check to make sure you have loaded the GRASS plugin properly as described in section *Starting the GRASS plugin*.

16.3 GRASS LOCATION and MAPSET

GRASS data are stored in a directory referred to as GISDBASE. This directory, often called `grassdata`, must be created before you start working with the GRASS plugin in QGIS. Within this directory, the GRASS GIS data are organized by projects stored in subdirectories called *LOCATIONS*. Each *LOCATION* is defined by its coordinate system, map projection and geographical boundaries. Each *LOCATION* can have several *MAPSETS* (subdirectories of the *LOCATION*) that are used to subdivide the project into different topics or subregions, or as workspaces for individual team members (see Neteler & Mitasova 2008 in *Literatur dan Referensi Web*). In order to analyze vector and raster layers with GRASS modules, you must import them into a GRASS *LOCATION*. (This is not strictly true – with the GRASS modules `r.external` and `v.external` you can create read-only links to external GDAL/OGR-supported datasets without importing them. But because this is not the usual way for beginners to work with GRASS, this functionality will not be described here.)



16.3.1 Creating a new GRASS LOCATION

As an example, here is how the sample GRASS *LOCATION* `alaska`, which is projected in Albers Equal Area projection with unit feet was created for the QGIS sample dataset. This sample GRASS *LOCATION* `alaska`

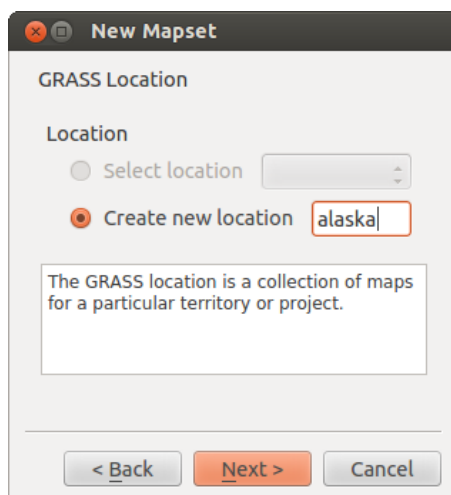


Gambar 16.1: GRASS data in the alaska LOCATION

will be used for all examples and exercises in the following GRASS-related sections. It is useful to download and install the dataset on your computer (see *Contoh data*).

1. Start QGIS and make sure the GRASS plugin is loaded.
2. Visualize the `alaska.shp` shapefile (see section *Loading a Shapefile*) from the QGIS Alaska dataset (see *Contoh data*).
3. In the GRASS toolbar, click on the  `New mapset` icon to bring up the *MAPSET* wizard.
4. Select an existing GRASS database (GISDBASE) folder `grassdata`, or create one for the new *LOCATION* using a file manager on your computer. Then click **[Next]**.
5. We can use this wizard to create a new *MAPSET* within an existing *LOCATION* (see section *Adding a new MAPSET*) or to create a new *LOCATION* altogether. Select *Create new location* (see figure `grass_location_2`).
6. Enter a name for the *LOCATION* – we used ‘alaska’ – and click **[Next]**.
7. Define the projection by clicking on the radio button *Projection* to enable the projection list.
8. We are using Albers Equal Area Alaska (feet) projection. Since we happen to know that it is represented by the EPSG ID 2964, we enter it in the search box. (Note: If you want to repeat this process for another *LOCATION* and projection and haven’t memorized the EPSG ID, click on the  `CRS Status` icon in the lower right-hand corner of the status bar (see section *Working with Projections*)).
9. In *Filter*, insert 2964 to select the projection.
10. Click **[Next]**.
11. To define the default region, we have to enter the *LOCATION* bounds in the north, south, east, and west directions. Here, we simply click on the button **[Set current lqgl extent]**, to apply the extent of the loaded layer `alaska.shp` as the GRASS default region extent.
12. Click **[Next]**.
13. We also need to define a *MAPSET* within our new *LOCATION* (this is necessary when creating a new *LOCATION*). You can name it whatever you like - we used ‘demo’. GRASS automatically creates a special *MAPSET* called `PERMANENT`, designed to store the core data for the project, its default spatial extent and coordinate system definitions (see Neteler & Mitasova 2008 in *Literatur dan Referensi Web*).
14. Check out the summary to make sure it’s correct and click **[Finish]**.
15. The new *LOCATION*, ‘alaska’, and two *MAPSET*s, ‘demo’ and ‘PERMANENT’, are created. The currently opened working set is ‘demo’, as you defined.

16. Notice that some of the tools in the GRASS toolbar that were disabled are now enabled.




Gambar 16.2: Creating a new GRASS LOCATION or a new MAPSET in QGIS

If that seemed like a lot of steps, it's really not all that bad and a very quick way to create a LOCATION. The LOCATION 'alaska' is now ready for data import (see section *Importing data into a GRASS LOCATION*). You can also use the already-existing vector and raster data in the sample GRASS LOCATION 'alaska', included in the QGIS 'Alaska' dataset *Contoh data*, and move on to section *The GRASS vector data model*.

16.3.2 Adding a new MAPSET



A user has write access only to a GRASS MAPSET he or she created. This means that besides access to your own MAPSET, you can read maps in other users' MAPSETs (and they can read yours), but you can modify or remove only the maps in your own MAPSET.

All MAPSETs include a WIND file that stores the current boundary coordinate values and the currently selected raster resolution (see Neteler & Mitasova 2008 in *Literatur dan Referensi Web*, and section *The GRASS region tool*).

1. Start QGIS and make sure the GRASS plugin is loaded.
2. In the GRASS toolbar, click on the  New mapset icon to bring up the MAPSET wizard.
3. Select the GRASS database (GISDBASE) folder `grassdata` with the LOCATION 'alaska', where we want to add a further MAPSET called 'test'.
4. Click [Next].
5. We can use this wizard to create a new MAPSET within an existing LOCATION or to create a new LOCATION altogether. Click on the radio button *Select location* (see [figure_grass_location_2](#)) and click [Next].
6. Enter the name `test` for the new MAPSET. Below in the wizard, you see a list of existing MAPSETs and corresponding owners.
7. Click [Next], check out the summary to make sure it's all correct and click [Finish].

16.4 Importing data into a GRASS LOCATION

This section gives an example of how to import raster and vector data into the 'alaska' GRASS LOCATION provided by the QGIS 'Alaska' dataset. Therefore, we use the landcover raster map `landcover.img` and the vector GML file `lakes.gml` from the QGIS 'Alaska' dataset (see *Contoh data*).

1. Start QGIS and make sure the GRASS plugin is loaded.
2. In the GRASS toolbar, click the  Open MAPSET icon to bring up the *MAPSET* wizard.
3. Select as GRASS database the folder `grassdata` in the QGIS Alaska dataset, as LOCATION 'alaska', as MAPSET 'demo' and click **[OK]**.
4. Now click the  Open GRASS tools icon. The GRASS Toolbox (see section *The GRASS Toolbox*) dialog appears.
5. To import the raster map `landcover.img`, click the module `r.in.gdal` in the *Modules Tree* tab. This GRASS module allows you to import GDAL-supported raster files into a GRASS LOCATION. The module dialog for `r.in.gdal` appears.
6. Browse to the folder `raster` in the QGIS 'Alaska' dataset and select the file `landcover.img`.
7. As raster output name, define `landcover_grass` and click **[Run]**. In the *Output* tab, you see the currently running GRASS command `r.in.gdal -o input=/path/to/landcover.img output=landcover_grass`.
8. When it says **Successfully finished**, click **[View output]**. The `landcover_grass` raster layer is now imported into GRASS and will be visualized in the QGIS canvas.
9. To import the vector GML file `lakes.gml`, click the module `v.in.ogr` in the *Modules Tree* tab. This GRASS module allows you to import OGR-supported vector files into a GRASS LOCATION. The module dialog for `v.in.ogr` appears.
10. Browse to the folder `gml` in the QGIS 'Alaska' dataset and select the file `lakes.gml` as OGR file.
11. As vector output name, define `lakes_grass` and click **[Run]**. You don't have to care about the other options in this example. In the *Output* tab you see the currently running GRASS command `v.in.ogr -o dsname=/path/to/lakes.gml output=lakes_grass`.
12. When it says **Successfully finished**, click **[View output]**. The `lakes_grass` vector layer is now imported into GRASS and will be visualized in the QGIS canvas.

16.5 The GRASS vector data model

It is important to understand the GRASS vector data model prior to digitizing.

In general, GRASS uses a topological vector model.

This means that areas are not represented as closed polygons, but by one or more boundaries. A boundary between two adjacent areas is digitized only once, and it is shared by both areas. Boundaries must be connected and closed without gaps. An area is identified (and labeled) by the **centroid** of the area.

Besides boundaries and centroids, a vector map can also contain points and lines. All these geometry elements can be mixed in one vector and will be represented in different so-called 'layers' inside one GRASS vector map. So in GRASS, a layer is not a vector or raster map but a level inside a vector layer. This is important to distinguish carefully. (Although it is possible to mix geometry elements, it is unusual and, even in GRASS, only used in special cases such as vector network analysis. Normally, you should prefer to store different geometry elements in different layers.)

It is possible to store several 'layers' in one vector dataset. For example, fields, forests and lakes can be stored in one vector. An adjacent forest and lake can share the same boundary, but they have separate attribute tables. It is also possible to attach attributes to boundaries. An example might be the case where the boundary between a lake and a forest is a road, so it can have a different attribute table.

The 'layer' of the feature is defined by the 'layer' inside GRASS. 'Layer' is the number which defines if there is more than one layer inside the dataset (e.g., if the geometry is forest or lake). For now, it can be only a number. In the future, GRASS will also support names as fields in the user interface.

Attributes can be stored inside the GRASS LOCATION as dBase or SQLite3 or in external database tables, for example, PostgreSQL, MySQL, Oracle, etc.


Attributes in database tables are linked to geometry elements using a ‘category’ value.

‘Category’ (key, ID) is an integer attached to geometry primitives, and it is used as the link to one key column in the database table.

Tip: Learning the GRASS Vector Model

The best way to learn the GRASS vector model and its capabilities is to download one of the many GRASS tutorials where the vector model is described more deeply. See <http://grass.osgeo.org/documentation/manuals/> for more information, books and tutorials in several languages.

16.6 Creating a new GRASS vector layer


To create a new GRASS vector layer with the GRASS plugin, click the  Create new GRASS vector toolbar icon. Enter a name in the text box, and you can start digitizing point, line or polygon geometries following the procedure described in section *Digitizing and editing a GRASS vector layer*.

In GRASS, it is possible to organize all sorts of geometry types (point, line and area) in one layer, because GRASS uses a topological vector model, so you don’t need to select the geometry type when creating a new GRASS vector. This is different from shapefile creation with QGIS, because shapefiles use the Simple Feature vector model (see section *Creating new Vector layers*).

Tip: Creating an attribute table for a new GRASS vector layer

If you want to assign attributes to your digitized geometry features, make sure to create an attribute table with columns before you start digitizing (see [figure_grass_digitizing_5](#)).

16.7 Digitizing and editing a GRASS vector layer

The digitizing tools for GRASS vector layers are accessed using the  Edit GRASS vector layer icon on the toolbar. Make sure you have loaded a GRASS vector and it is the selected layer in the legend before clicking on the edit tool. Figure [figure_grass_digitizing_2](#) shows the GRASS edit dialog that is displayed when you click on the edit tool. The tools and settings are discussed in the following sections.

Tip: Digitizing polygons in GRASS

If you want to create a polygon in GRASS, you first digitize the boundary of the polygon, setting the mode to ‘No category’. Then you add a centroid (label point) into the closed boundary, setting the mode to ‘Next not used’. The reason for this is that a topological vector model links the attribute information of a polygon always to the centroid and not to the boundary.

Toolbar

In [figure_grass_digitizing_1](#), you see the GRASS digitizing toolbar icons provided by the GRASS plugin. Table [table_grass_digitizing_1](#) explains the available functionalities.



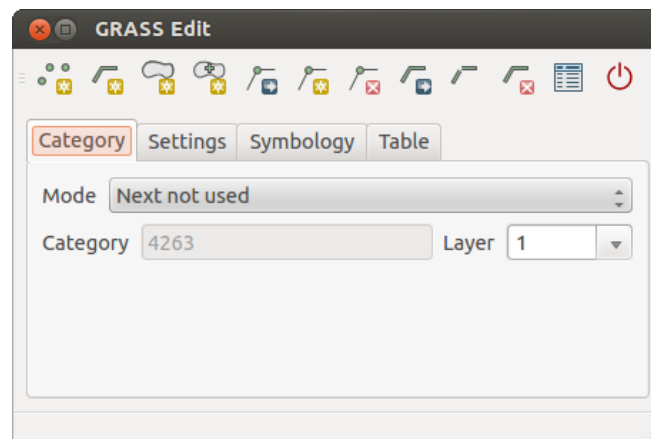
Gambar 16.3: GRASS Digitizing Toolbar

Icon	Tool	Purpose
	New Point	Digitize new point
	New Line	Digitize new line
	New Boundary	Digitize new boundary (finish by selecting new tool)
	New Centroid	Digitize new centroid (label existing area)
	Move vertex	Move one vertex of existing line or boundary and identify new position
	Add vertex	Add a new vertex to existing line
	Delete vertex	Delete vertex from existing line (confirm selected vertex by another click)
	Move element	Move selected boundary, line, point or centroid and click on new position
	Split line	Split an existing line into two parts
	Delete element	Delete existing boundary, line, point or centroid (confirm selected element by another click)
	Edit attributes	Edit attributes of selected element (note that one element can represent more features, see above)
	Close	Close session and save current status (rebuilds topology afterwards)

Table GRASS Digitizing 1: GRASS Digitizing Tools

Category Tab

The *Category* tab allows you to define the way in which the category values will be assigned to a new geometry element.



Gambar 16.4: GRASS Digitizing Category Tab

- **Mode:** The category value that will be applied to new geometry elements.
 - Next not used - Apply next not yet used category value to geometry element.
 - Manual entry - Manually define the category value for the geometry element in the 'Category' entry field.
 - No category - Do not apply a category value to the geometry element. This is used, for instance, for area boundaries, because the category values are connected via the centroid.
- **Category** - The number (ID) that is attached to each digitized geometry element. It is used to connect each geometry element with its attributes.

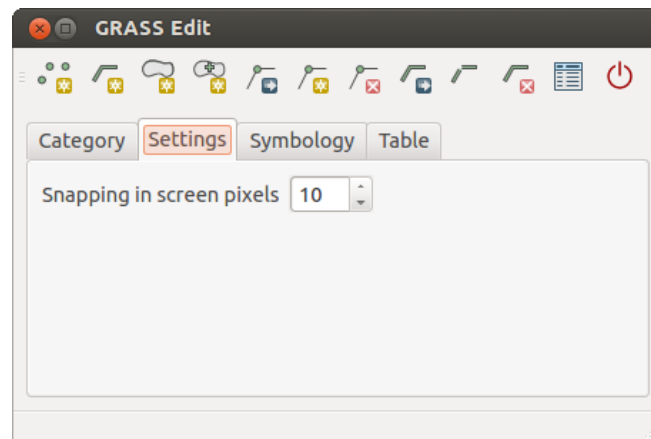
- **Field (layer)** - Each geometry element can be connected with several attribute tables using different GRASS geometry layers. The default layer number is 1.

Tip: Creating an additional GRASS ‘layer’ with lqgl

If you would like to add more layers to your dataset, just add a new number in the ‘Field (layer)’ entry box and press return. In the Table tab, you can create your new table connected to your new layer.

Settings Tab

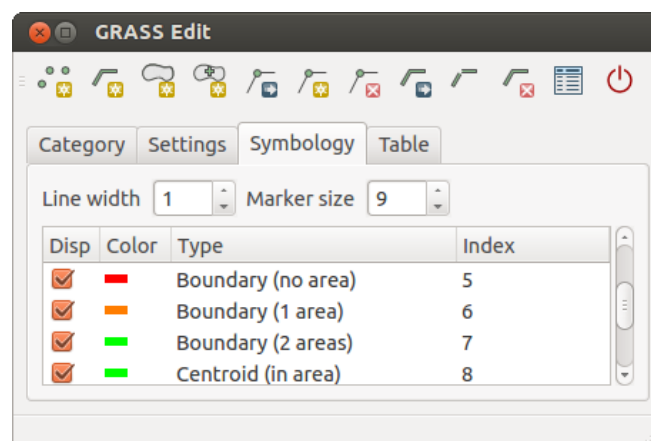
The *Settings* tab allows you to set the snapping in screen pixels. The threshold defines at what distance new points or line ends are snapped to existing nodes. This helps to prevent gaps or dangles between boundaries. The default is set to 10 pixels.



Gambar 16.5: GRASS Digitizing Settings Tab

Symbology Tab

The *Symbology* tab allows you to view and set symbology and color settings for various geometry types and their topological status (e.g., closed / opened boundary).

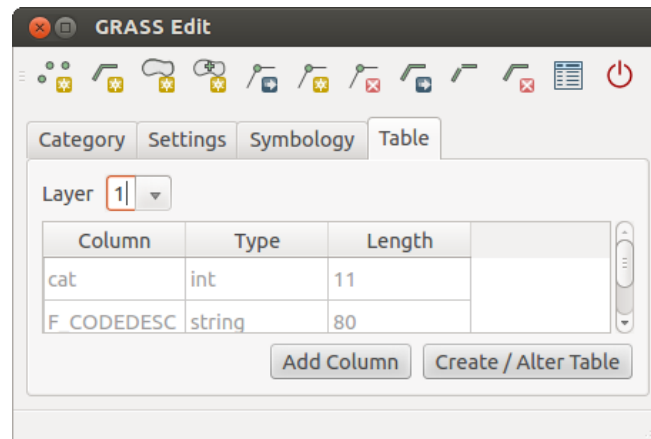


Gambar 16.6: GRASS Digitizing Symbology Tab

Table Tab

The *Table* tab provides information about the database table for a given ‘layer’. Here, you can add new columns to an existing attribute table, or create a new database table for a new GRASS vector layer (see section *Creating a new GRASS vector layer*).

Tip: GRASS Edit Permissions





Gambar 16.7: GRASS Digitizing Table Tab

You must be the owner of the GRASS MAPSET you want to edit. It is impossible to edit data layers in a MAPSET that is not yours, even if you have write permission.

16.8 The GRASS region tool


The region definition (setting a spatial working window) in GRASS is important for working with raster layers. Vector analysis is by default not limited to any defined region definitions. But all newly created rasters will have the spatial extension and resolution of the currently defined GRASS region, regardless of their original extension and resolution. The current GRASS region is stored in the `$LOCATION/$MAPSET/WIND` file, and it defines north, south, east and west bounds, number of columns and rows, horizontal and vertical spatial resolution.

It is possible to switch on and off the visualization of the GRASS region in the QGIS canvas using the  Display current GRASS region button.

With the  Edit current GRASS region icon, you can open a dialog to change the current region and the symbology of the GRASS region rectangle in the QGIS canvas. Type in the new region bounds and resolution, and click [OK]. The dialog also allows you to select a new region interactively with your mouse on the QGIS canvas. Therefore, click with the left mouse button in the QGIS canvas, open a rectangle, close it using the left mouse button again and click [OK].

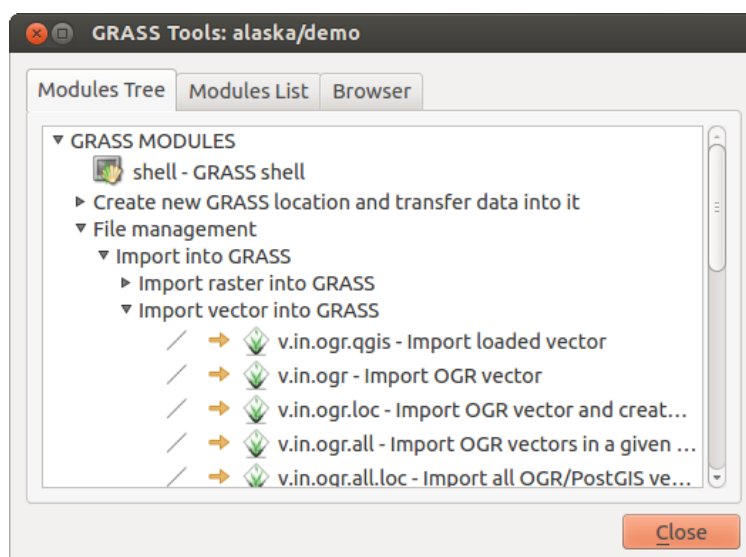
The GRASS module `g.region` provides a lot more parameters to define an appropriate region extent and resolution for your raster analysis. You can use these parameters with the GRASS Toolbox, described in section [The GRASS Toolbox](#).

16.9 The GRASS Toolbox

The  Open GRASS Tools box provides GRASS module functionalities to work with data inside a selected GRASS LOCATION and MAPSET. To use the GRASS Toolbox you need to open a LOCATION and MAPSET that you have write permission for (usually granted, if you created the MAPSET). This is necessary, because new raster or vector layers created during analysis need to be written to the currently selected LOCATION and MAPSET.

16.9.1 Working with GRASS modules

The GRASS shell inside the GRASS Toolbox provides access to almost all (more than 300) GRASS modules in a command line interface. To offer a more user-friendly working environment, about 200 of the available GRASS modules and functionalities are also provided by graphical dialogs within the GRASS plugin Toolbox.



Gambar 16.8: GRASS Toolbox and Module Tree 

A complete list of GRASS modules available in the graphical Toolbox in QGIS version 2.6 is available in the GRASS wiki at http://grass.osgeo.org/wiki/GRASS-QGIS_relevant_module_list.

It is also possible to customize the GRASS Toolbox content. This procedure is described in section *Customizing the GRASS Toolbox*.

As shown in [figure_grass_toolbox_1](#), you can look for the appropriate GRASS module using the thematically grouped *Modules Tree* or the searchable *Modules List* tab.

By clicking on a graphical module icon, a new tab will be added to the Toolbox dialog, providing three new sub-tabs: *Options*, *Output* and *Manual*.

Options

The *Options* tab provides a simplified module dialog where you can usually select a raster or vector layer visualized in the QGIS canvas and enter further module-specific parameters to run the module.

The provided module parameters are often not complete to keep the dialog clear. If you want to use further module parameters and flags, you need to start the GRASS shell and run the module in the command line.

A new feature since QGIS 1.8 is the support for a *Show Advanced Options* button below the simplified module dialog in the *Options* tab. At the moment, it is only added to the module `v.in.ascii` as an example of use, but it will probably be part of more or all modules in the GRASS Toolbox in future versions of QGIS. This allows you to use the complete GRASS module options without the need to switch to the GRASS shell.

Output

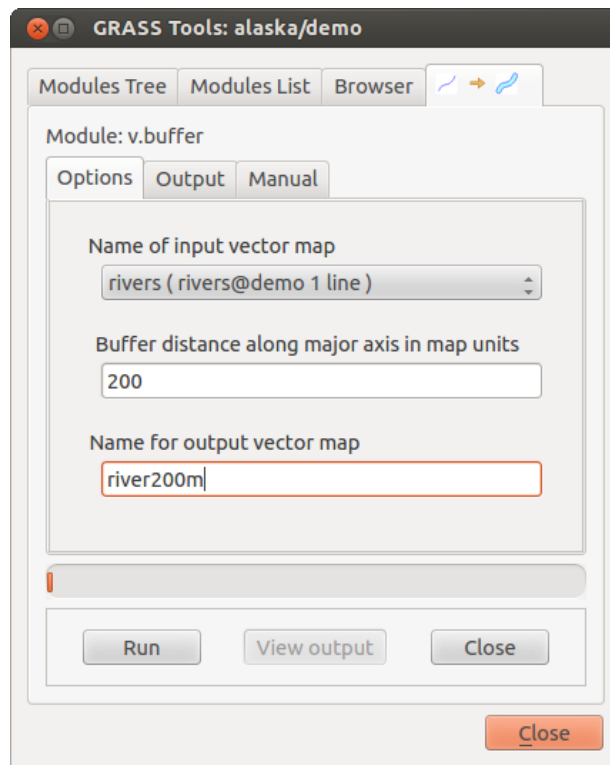
The *Output* tab provides information about the output status of the module. When you click the **[Run]** button, the module switches to the *Output* tab and you see information about the analysis process. If all works well, you will finally see a `Successfully finished` message.

Manual

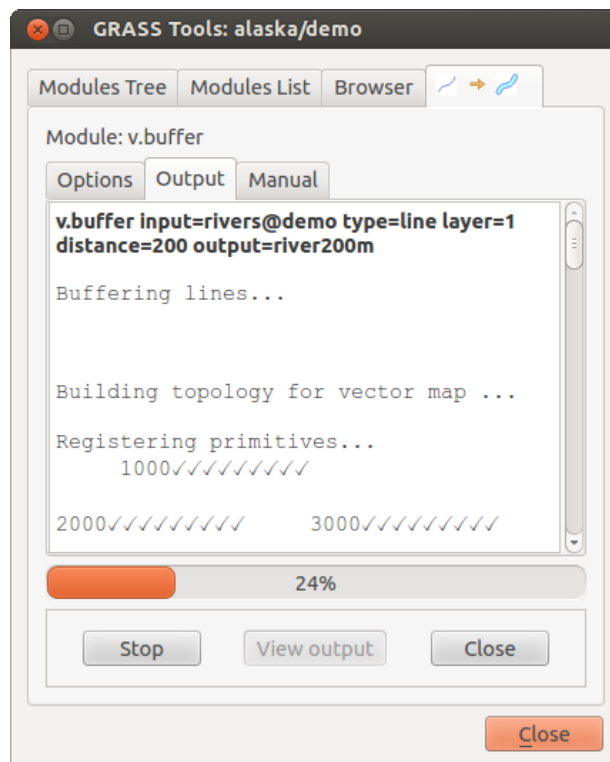
The *Manual* tab shows the HTML help page of the GRASS module. You can use it to check further module parameters and flags or to get a deeper knowledge about the purpose of the module. At the end of each module manual page, you see further links to the `Main Help index`, the `Thematic index` and the `Full index`. These links provide the same information as the module `g.manual`.

Tip: Display results immediately

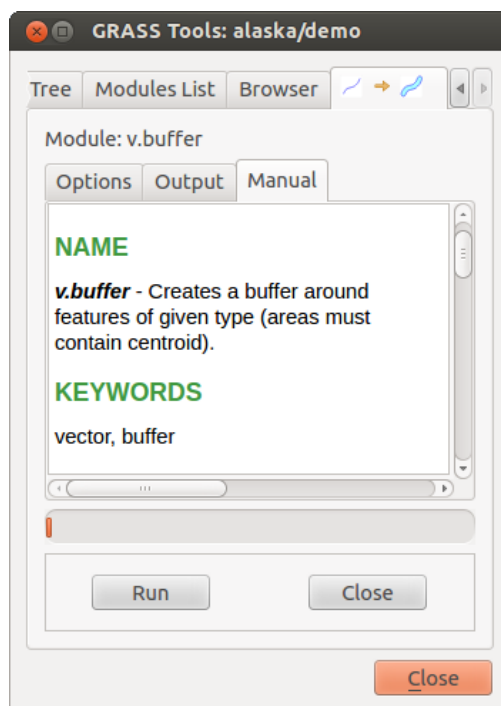
If you want to display your calculation results immediately in your map canvas, you can use the ‘View Output’ button at the bottom of the module tab.



Gambar 16.9: GRASS Toolbox Module Options 



Gambar 16.10: GRASS Toolbox Module Output 






Gambar 16.11: GRASS Toolbox Module Manual 

16.9.2 GRASS module examples

The following examples will demonstrate the power of some of the GRASS modules.

Creating contour lines

The first example creates a vector contour map from an elevation raster (DEM). Here, it is assumed that you have the Alaska LOCATION set up as explained in section *Importing data into a GRASS LOCATION*.

- First, open the location by clicking the  Open mapset button and choosing the Alaska location.
- Now load the `gtopo30` elevation raster by clicking  Add GRASS raster layer and selecting the `gtopo30` raster from the demo location.
- Now open the Toolbox with the  Open GRASS tools button.
- In the list of tool categories, double-click *Raster* → *Surface Management* → *Generate vector contour lines*.
- Now a single click on the tool **r.contour** will open the tool dialog as explained above (see *Working with GRASS modules*). The `gtopo30` raster should appear as the *Name of input raster*.
- Type into the *Increment between Contour levels* the value 100. (This will create contour lines at intervals of 100 meters.)
- Type into the *Name for output vector map* the name `ctour_100`.
- Click **[Run]** to start the process. Wait for several moments until the message `Successfully finished` appears in the output window. Then click **[View Output]** and **[Close]**.

Since this is a large region, it will take a while to display. After it finishes rendering, you can open the layer properties window to change the line color so that the contours appear clearly over the elevation raster, as in *The Vector Properties Dialog*.

Next, zoom in to a small, mountainous area in the center of Alaska. Zooming in close, you will notice that the contours have sharp corners. GRASS offers the **v.generalize** tool to slightly alter vector maps while keeping

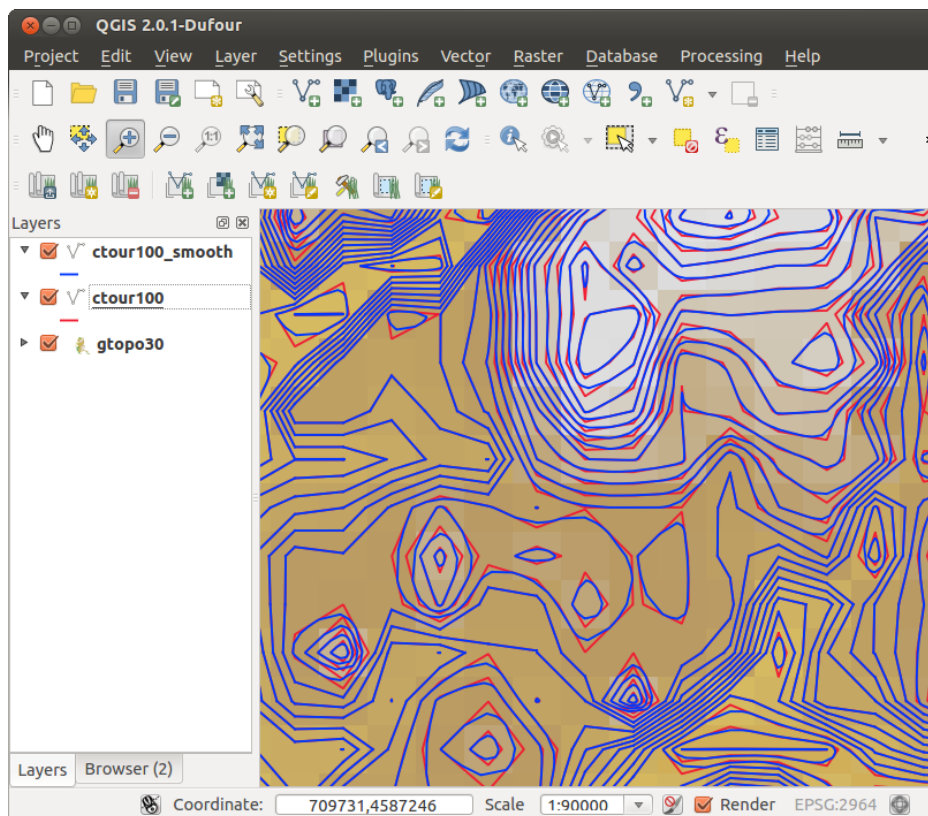
their overall shape. The tool uses several different algorithms with different purposes. Some of the algorithms (i.e., Douglas Peucker and Vertex Reduction) simplify the line by removing some of the vertices. The resulting vector will load faster. This process is useful when you have a highly detailed vector, but you are creating a very small-scale map, so the detail is unnecessary.

Tip: The simplify tool

Note that the QGIS fTools plugin has a *Simplify geometries* → tool that works just like the GRASS **v.generalize** Douglas-Peucker algorithm.

However, the purpose of this example is different. The contour lines created by `r.contour` have sharp angles that should be smoothed. Among the **v.generalize** algorithms, there is Chaiken's, which does just that (also Hermite splines). Be aware that these algorithms can **add** additional vertices to the vector, causing it to load even more slowly.

- Open the GRASS Toolbox and double-click the categories *Vector* → *Develop map* → *Generalization*, then click on the **v.generalize** module to open its options window.
- Check that the 'ctour_100' vector appears as the *Name of input vector*.
- From the list of algorithms, choose Chaiken's. Leave all other options at their default, and scroll down to the last row to enter in the field *Name for output vector map* 'ctour_100_smooth', and click **[Run]**.
- The process takes several moments. Once *Successfully finished* appears in the output windows, click **[View output]** and then **[Close]**.
- You may change the color of the vector to display it clearly on the raster background and to contrast with the original contour lines. You will notice that the new contour lines have smoother corners than the original while staying faithful to the original overall shape.



Gambar 16.12: GRASS module v.generalize to smooth a vector map 🐧

Tip: Other uses for r.contour

The procedure described above can be used in other equivalent situations. If you have a raster map of precipitation data, for example, then the same method will be used to create a vector map of isohyetal (constant rainfall) lines.

Creating a Hillshade 3-D effect

Several methods are used to display elevation layers and give a 3-D effect to maps. The use of contour lines, as shown above, is one popular method often chosen to produce topographic maps. Another way to display a 3-D effect is by hillshading. The hillshade effect is created from a DEM (elevation) raster by first calculating the slope and aspect of each cell, then simulating the sun's position in the sky and giving a reflectance value to each cell. Thus, you get sun-facing slopes lighted; the slopes facing away from the sun (in shadow) are darkened.

- Begin this example by loading the `gtopo30` elevation raster. Start the GRASS Toolbox, and under the Raster category, double-click to open *Spatial analysis* → *Terrain analysis*.
- Then click **r.shaded.relief** to open the module.
- Change the *azimuth angle* 270 to 315.
- Enter `gtopo30_shade` for the new hillshade raster, and click **[Run]**.
- When the process completes, add the hillshade raster to the map. You should see it displayed in grayscale.
- To view both the hillshading and the colors of the `gtopo30` together, move the hillshade map below the `gtopo30` map in the table of contents, then open the *Properties* window of `gtopo30`, switch to the *Transparency* tab and set its transparency level to about 25%.

You should now have the `gtopo30` elevation with its colormap and transparency setting displayed **above** the grayscale hillshade map. In order to see the visual effects of the hillshading, turn off the `gtopo30_shade` map, then turn it back on.

Using the GRASS shell

The GRASS plugin in QGIS is designed for users who are new to GRASS and not familiar with all the modules and options. As such, some modules in the Toolbox do not show all the options available, and some modules do not appear at all. The GRASS shell (or console) gives the user access to those additional GRASS modules that do not appear in the Toolbox tree, and also to some additional options to the modules that are in the Toolbox with the simplest default parameters. This example demonstrates the use of an additional option in the **r.shaded.relief** module that was shown above.

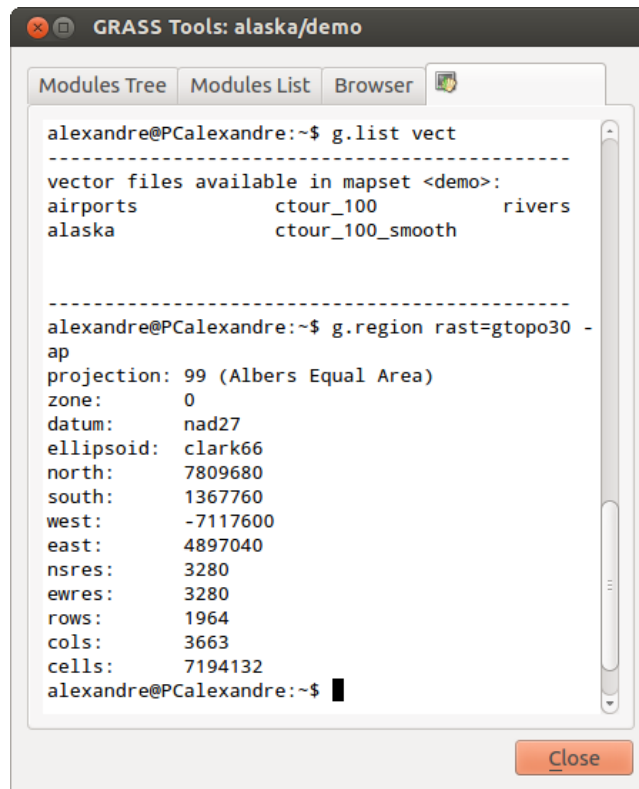
The module **r.shaded.relief** can take a parameter `zmult`, which multiplies the elevation values relative to the X-Y coordinate units so that the hillshade effect is even more pronounced.

- Load the `gtopo30` elevation raster as above, then start the GRASS Toolbox and click on the GRASS shell. In the shell window, type the command `r.shaded.relief map=gtopo30 shade=gtopo30_shade2 azimuth=315 zmult=3` and press **[Enter]**.
- After the process finishes, shift to the *Browse* tab and double-click on the new `gtopo30_shade2` raster to display it in QGIS.
- As explained above, move the shaded relief raster below the `gtopo30` raster in the table of contents, then check the transparency of the colored `gtopo30` layer. You should see that the 3-D effect stands out more strongly compared with the first shaded relief map.

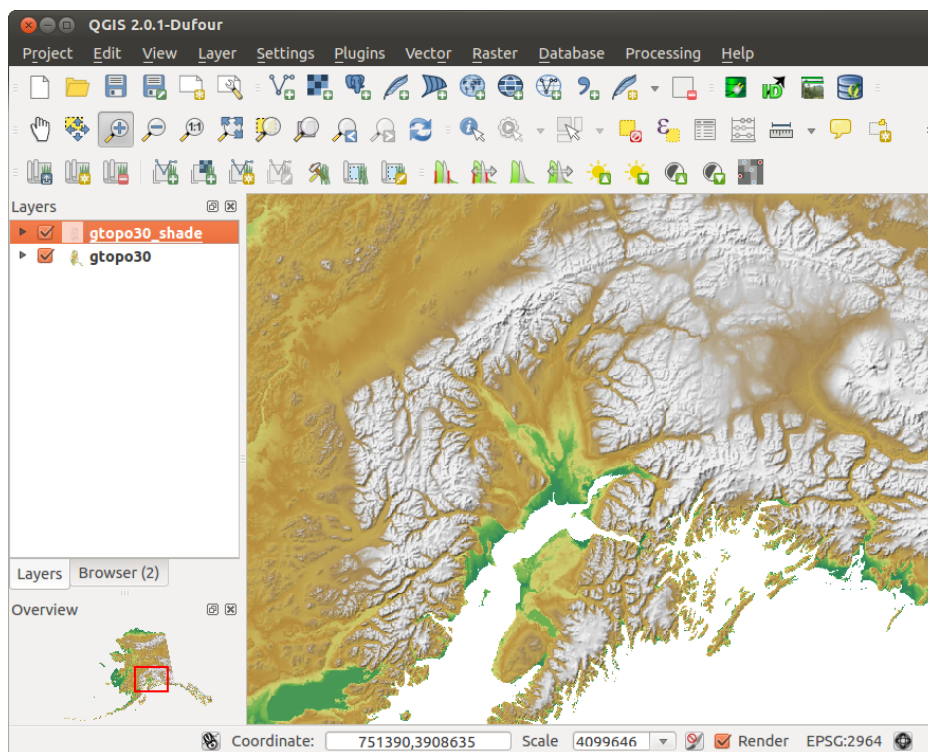
Raster statistics in a vector map

The next example shows how a GRASS module can aggregate raster data and add columns of statistics for each polygon in a vector map.


- Again using the Alaska data, refer to *Importing data into a GRASS LOCATION* to import the trees shapefile from the `shapefiles` directory into GRASS.
- Now an intermediate step is required: centroids must be added to the imported trees map to make it a complete GRASS area vector (including both boundaries and centroids).



Gambar 16.13: The GRASS shell, r.shaded.relief module 🐧



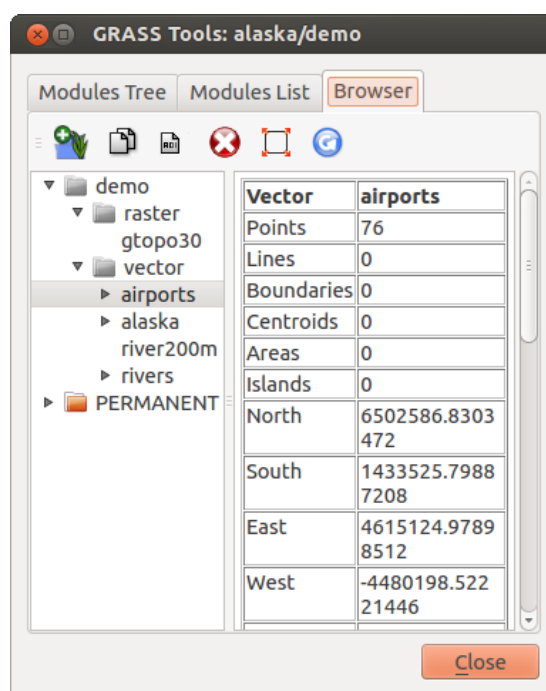
Gambar 16.14: Displaying shaded relief created with the GRASS module r.shaded.relief 🐧

- From the Toolbox, choose *Vector* → *Manage features*, and open the module **v.centroids**.
- Enter as the *output vector map* 'forest_areas' and run the module.
- Now load the forest_areas vector and display the types of forests - deciduous, evergreen, mixed - in different colors: In the layer *Properties* window, *Symbology* tab, choose from *Legend type*  'Unique value' and set the *Classification field* to 'VEGDESC'. (Refer to the explanation of the symbology tab in *Style Menu* of the vector section.)
- Next, reopen the GRASS Toolbox and open *Vector* → *Vector update* by other maps.
- Click on the **v.rast.stats** module. Enter gtopo30 and forest_areas.
- Only one additional parameter is needed: Enter *column prefix* elev, and click **[Run]**. This is a computationally heavy operation, which will run for a long time (probably up to two hours).
- Finally, open the forest_areas attribute table, and verify that several new columns have been added, including elev_min, elev_max, elev_mean, etc., for each forest polygon.

16.9.3 Working with the GRASS LOCATION browser




Another useful feature inside the GRASS Toolbox is the GRASS LOCATION browser. In [figure_grass_module_7](#), you can see the current working LOCATION with its MAPSETS.




In the left browser windows, you can browse through all MAPSETS inside the current LOCATION. The right browser window shows some meta-information for selected raster or vector layers (e.g., resolution, bounding box, data source, connected attribute table for vector data, and a command history).





Gambar 16.15: GRASS LOCATION browser 

The toolbar inside the *Browser* tab offers the following tools to manage the selected LOCATION:

-  *Add selected map to canvas*
-  *Copy selected map*
-  *Rename selected map*

-  *Delete selected map*
-  *Set current region to selected map*
-  *Refresh browser window*

The  *Rename selected map* and  *Delete selected map* only work with maps inside your currently selected MAPSET. All other tools also work with raster and vector layers in another MAPSET.

16.9.4 Customizing the GRASS Toolbox

Nearly all GRASS modules can be added to the GRASS Toolbox. An XML interface is provided to parse the pretty simple XML files that configure the modules' appearance and parameters inside the Toolbox.

A sample XML file for generating the module `v.buffer` (`v.buffer.qgm`) looks like this:

```
<?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>
```

The parser reads this definition and creates a new tab inside the Toolbox when you select the module. A more detailed description for adding new modules, changing a module's group, etc., can be found on the QGIS wiki at http://hub.qgis.org/projects/quantum-gis/wiki/Adding_New_Tools_to_the_GRASS_Toolbox.

QGIS kerangka pengolahan

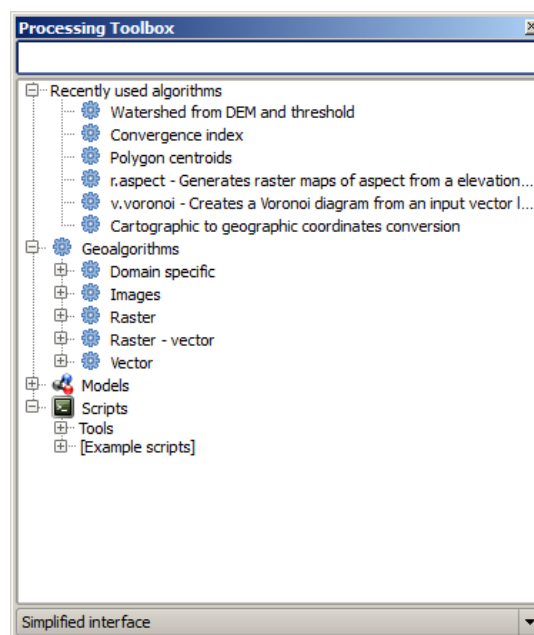
17.1 Pengantar

Bab ini memperkenalkan kerangka pengolahan QGIS, lingkungan geoprocessing yang dapat digunakan untuk memanggil algoritma asli dan pihak ketiga dari QGIS, membuat tugas analisis spasial Anda lebih produktif dan mudah untuk melakukannya.

Pada bagian berikut kita akan meninjau bagaimana menggunakan unsur-unsur grafis dari kerangka kerja ini dan mengambil yang terbaik masing-masing dari mereka.

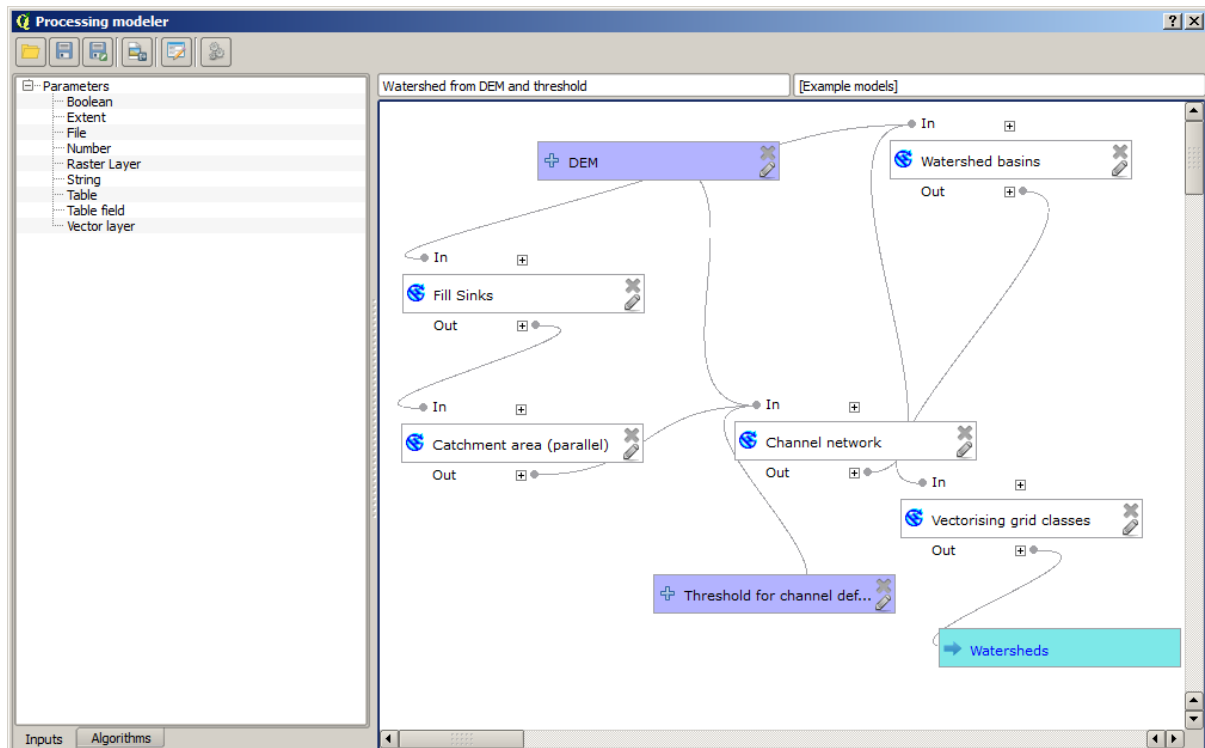
Ada empat elemen dasar dalam kerangka GUI, yang digunakan untuk menjalankan algoritma untuk tujuan yang berbeda. Memilih salah satu alat atau lain akan tergantung pada jenis analisis yang akan dilakukan dan karakteristik tertentu dari masing-masing pengguna dan proyek. Semuanya (kecuali untuk antarmuka batch processing, yang disebut dari toolbox, seperti akan kita lihat) bisa diakses dari menu *Pengolahan* (Anda akan melihat lebih dari empat entri. Yang tersisa tidak digunakan untuk mengeksekusi algoritma dan akan dijelaskan nanti dalam bab ini).

- Toolbox. Unsur utama dari GUI, digunakan untuk menjalankan algoritma tunggal atau menjalankan proses batch berdasarkan algoritma tersebut.



Gambar 17.1: Toolbox Pengolahan 

- Modeler grafis. Beberapa algoritma dapat dikombinasikan secara grafis dengan menggunakan modeler untuk mendefinisikan alur kerja, menciptakan satu proses yang melibatkan beberapa sub-proses



Gambar 17.2: Modeler Pengolahan

- Sejarah pengelola. Semua tindakan dilakukan dengan menggunakan salah satu elemen tersebut disimpan dalam berkas sejarah dan dapat kemudian dengan mudah direproduksi menggunakan manajer sejarah
- Antarmuka pengolahan batch. Antarmuka ini memungkinkan Anda untuk menjalankan proses batch dan mengotomatisasi eksekusi algoritma tunggal pada beberapa dataset.

Dalam bagian berikut ini, kita akan meninjau masing-masing elemen ini secara rinci.

17.2 The toolbox

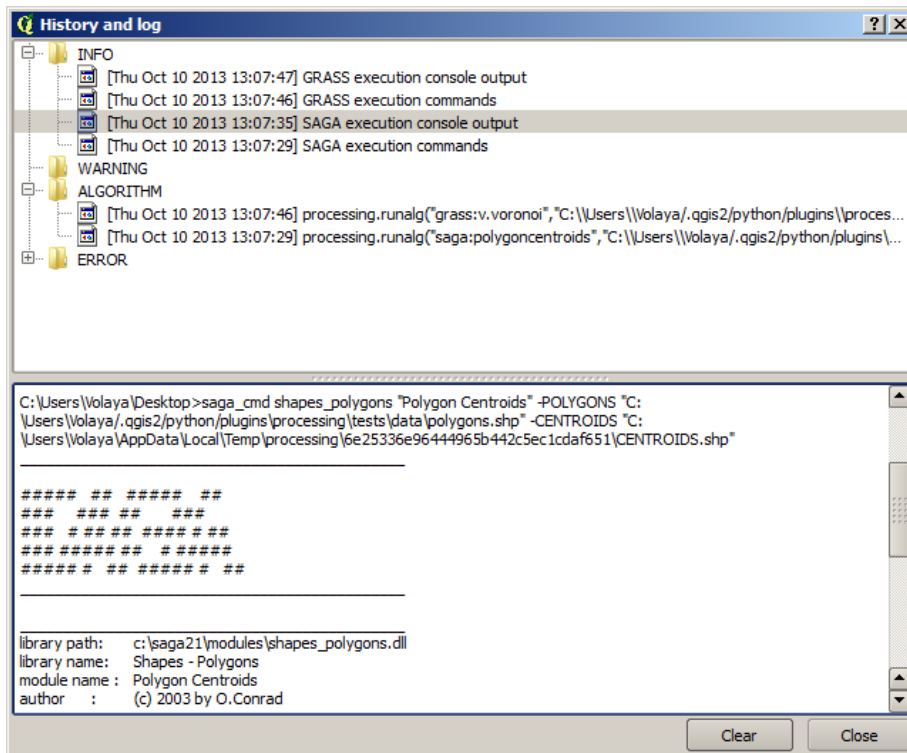
The *Toolbox* is the main element of the processing GUI, and the one that you are more likely to use in your daily work. It shows the list of all available algorithms grouped in different blocks, and it is the access point to run them, whether as a single process or as a batch process involving several executions of the same algorithm on different sets of inputs.

The toolbox contains all the available algorithms, divided into predefined groups. All these groups are found under a single tree entry named *Geoalgorithms*.

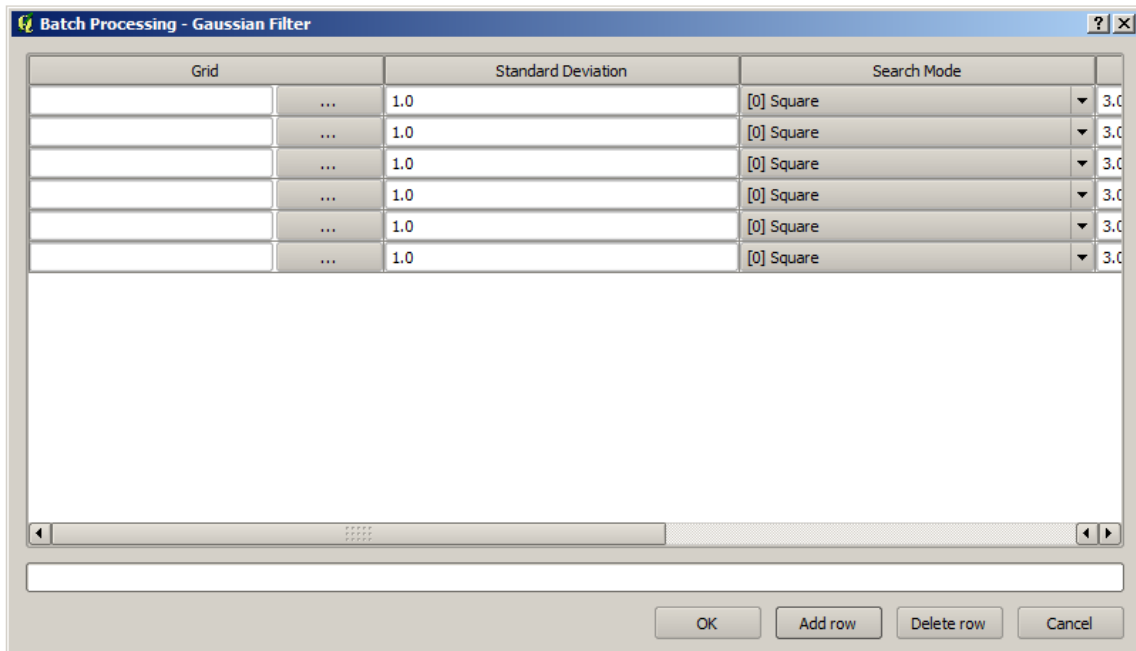
Additionally, two more entries are found, namely *Models* and *Scripts*. These include user-created algorithms, and they allow you to define your own workflows and processing tasks. We will devote a full section to them a bit later.

In the upper part of the toolbox, you will find a text box. To reduce the number of algorithms shown in the toolbox and make it easier to find the one you need, you can enter any word or phrase on the text box. Notice that, as you type, the number of algorithms in the toolbox is reduced to just those that contain the text you have entered in their names.

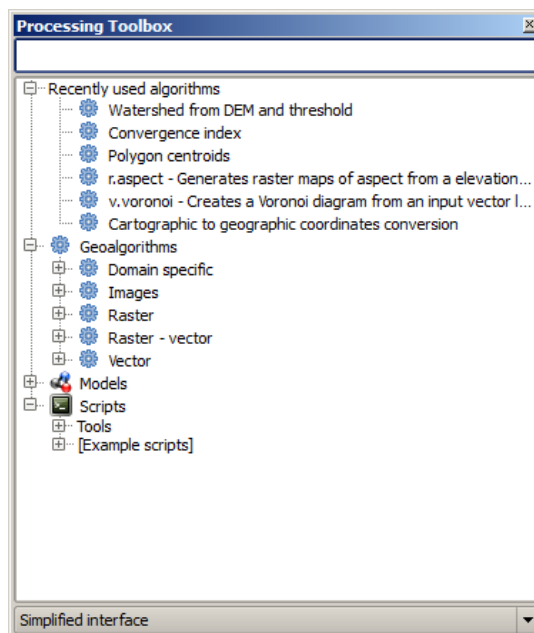
In the lower part, you will find a box that allows you to switch between the simplified algorithm list (the one explained above) and the advanced list. If you change to the advanced mode, the toolbox will look like this:



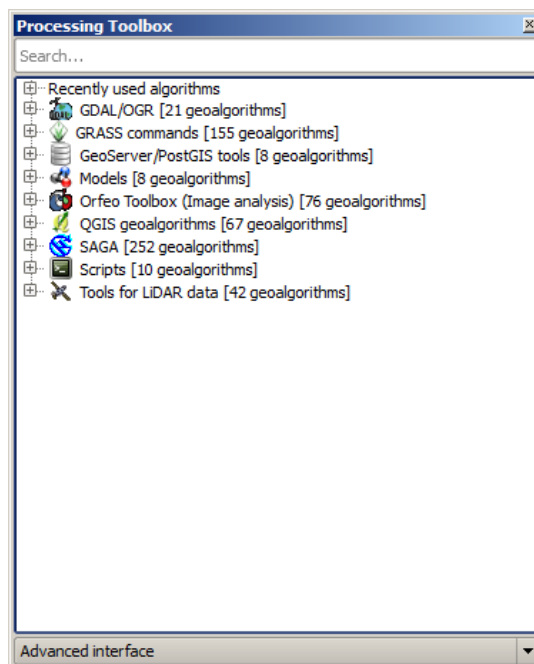
Gambar 17.3: Pengolahan Sejarah



Gambar 17.4: Antarmuka proses batch



Gambar 17.5: Processing Toolbox



Gambar 17.6: Processing Toolbox (advanced mode)

In the advanced view, each group represents a so-called ‘algorithm provider’, which is a set of algorithms coming from the same source, for instance, from a third-party application with geoprocessing capabilities. Some of these groups represent algorithms from third-party applications like SAGA, GRASS or R, while others contain algorithms directly coded as part of the processing plugin, not relying on any additional software.

This view is recommended to those users who have a certain knowledge of the applications that are backing the algorithms, since they will be shown with their original names and groups.

Also, some additional algorithms are available only in the advanced view, such as LiDAR tools and scripts based on the R statistical computing software, among others. Independent QGIS plugins that add new algorithms to the toolbox will only be shown in the advanced view.

In particular, the simplified view contains algorithms from the following providers:

- GRASS
- SAGA
- OTB
- Native QGIS algorithms

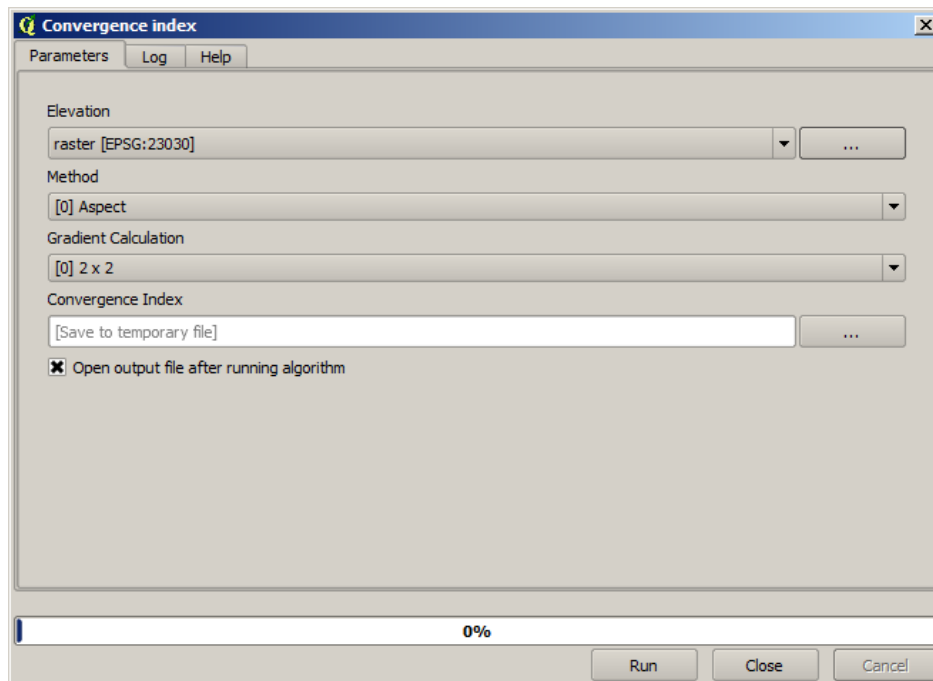
In the case of running QGIS under Windows, these algorithms are fully-functional in a fresh installation of QGIS, and they can be run without requiring any additional installation. Also, running them requires no prior knowledge of the external applications they use, making them more accessible for first-time users.

If you want to use an algorithm not provided by any of the above providers, switch to the advanced mode by selecting the corresponding option at the bottom of the toolbox.

To execute an algorithm, just double-click on its name in the toolbox.

17.2.1 The algorithm 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 SAGA ‘Convergence index’ algorithm).



Gambar 17.7: Parameters Dialog 

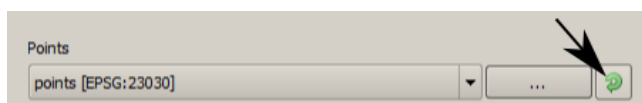
This dialog is used to set the input values that the algorithm needs to be executed. It shows a table where input values and configuration parameters are to be set. It of course has a different content, depending on the require-

ments of the algorithm to be executed, and is created automatically based on those requirements. On the left side, the name of the parameter is shown. On the right side, the value of the parameter can be set.

Although the number and type of parameters depend on the characteristics of the algorithm, the structure is similar for all of them. The parameters found in the table can be of one of the following types.

- 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.

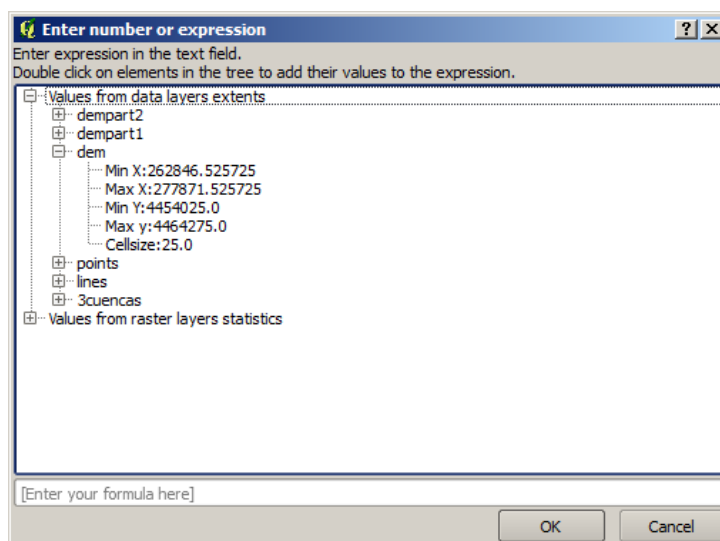
You will see a button by each vector layer selector, as shown in the figure below.



Gambar 17.8: Vector iterator button

If the algorithm contains several of them, you will be able to toggle just one of them. If the button corresponding to a vector input is toggled, the algorithm will be executed iteratively on each one of its features, instead of just once for the whole layer, producing as many outputs as times the algorithm is executed. This allows for automating the process when all features in a layer have to be processed separately.

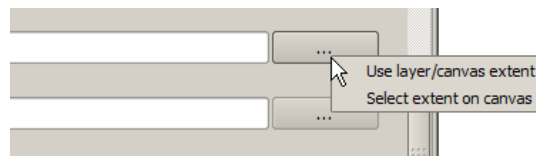
- A table, to select from a list of all available in QGIS. Non-spatial tables are loaded into QGIS like vector layers, and in fact they are treated as such by the program. Currently, the list of available tables that you will see when executing an algorithm that needs one of them is restricted to tables 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 text box. You will find a button by its side. Clicking on it, you will see a dialog that allows you to enter a mathematical expression, so you can use it as a handy calculator. 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.



Gambar 17.9: Number Selector

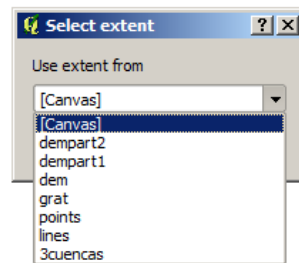
- A range, with min and max values to be introduced in two text boxes.

- 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 type the EPSG code directly in the text box, or select it 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 x_{min} , x_{max} , y_{min} , y_{max} limits. Clicking on the button on the right-hand side of the value selector, a pop-up menu will appear, giving you two options: to select the value from a layer or the current canvas extent, or to define it by dragging directly onto the map canvas.



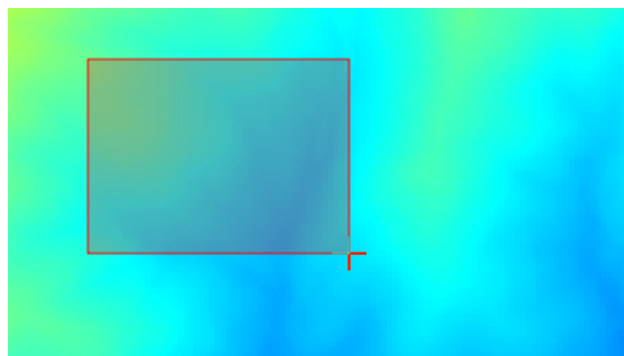
Gambar 17.10: Extent selector

If you select the first option, you will see a window like the next one.



Gambar 17.11: Extent List

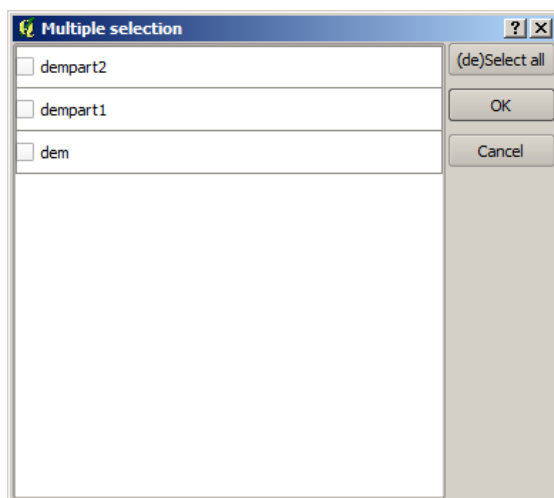
If you select the second one, the parameters window will hide itself, so you can click and drag onto the canvas. Once you have defined the selected rectangle, the dialog will reappear, containing the values in the extent text box.



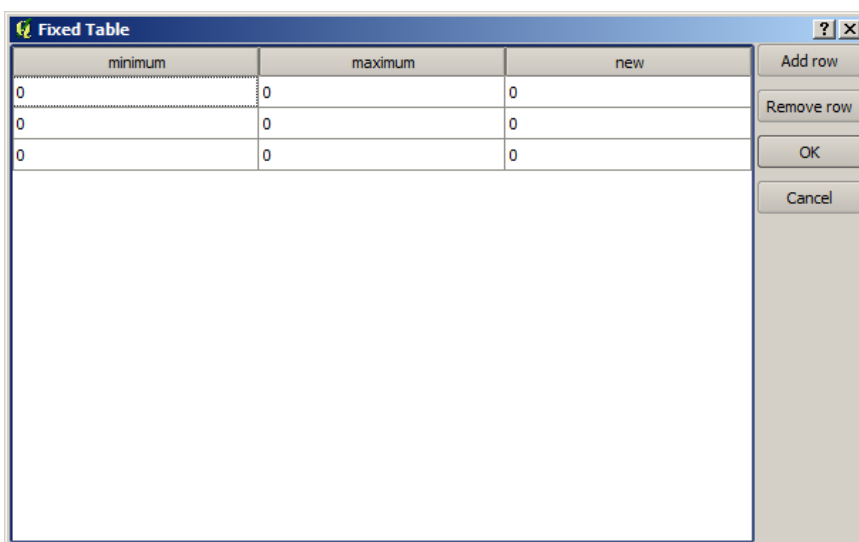
Gambar 17.12: Extent Drag

- A list of elements (whether raster layers, vector layers or tables), to select from the list of such layers available in QGIS. To make the selection, click on the small button on the left side of the corresponding row to see a dialog like the following one.
- A small table to be edited by the user. These are used to define parameters like lookup tables or convolution kernels, among others.

Click on the button on the right side to see the table and edit its values.



Gambar 17.13: Multiple Selection



Gambar 17.14: Fixed Table

Depending on the algorithm, the number of rows can be modified or not by using the buttons on the right side of the window.

You will find a **[Help]** tab in the the parameters dialog. If a help file is available, it will be shown, giving you more information about the algorithm and detailed descriptions of what each parameter does. Unfortunately, most algorithms lack good documentation, but if you feel like contributing to the project, this would be a good place to start.

A note on projections

Algorithms run from the processing framework — this is also true of most of the external applications whose algorithms are exposed through it. Do not perform any reprojection on input layers and assume that all of them are already in a common coordinate system and ready to be analyzed. Whenever you use more than one layer as input to an algorithm, whether vector or raster, it is up to you to make sure that they are all in the same coordinate system.

Note that, 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. That reprojection should be done manually, and then the resulting files should be used as input to the algorithm. Also, note that the reprojection process can be performed with the algorithms that are available in the processing framework itself.

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 configuration dialog, unchecking the *Show CRS* option.

If you try to execute an algorithm using as input two or more layers with unmatching CRSs, a warning dialog will be shown.

You still can execute the algorithm, but be aware that in most cases that will produce wrong results, such as empty layers due to input layers not overlapping.

17.2.2 Data objects generated by algorithms

Data objects generated by an algorithm can be of any of the following types:

- A raster layer
- A vector layer
- A table
- An HTML file (used for text and graphical outputs)

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 the architecture allows for any other way of storing it. For instance, a vector layer can be stored in a database or even uploaded to a remote server using a WFS-T service. Although solutions like these are not yet implemented, the processing framework is prepared to handle them, and we expect to add new kinds of output channels in a near future.

To select an output channel, just click on the button on the right side of the text box. That will open a save file dialog, where you can select the desired file path. Supported file extensions are shown in the file format selector of the dialog, depending on the kind of output and the algorithm.

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 (usually `.dbf` for tables, `.tif` for raster layers and `.shp` for vector layers) will be appended to the file path, and the file format corresponding to that extension will be used to save the layer or table.

If you do not enter any filename, 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 configuration dialog (you can open it from the *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.

When running an algorithm that uses a vector layer in iterative mode, the entered file path is used as the base path for all generated files, which are named using the base name and appending a number representing the index of the iteration. The file extension (and format) is used for all such generated files.

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.

Some external applications might have files (with no particular extension restrictions) as output, but they do not belong to any of the categories above. Those output files will not be processed by QGIS (opened or included into the current QGIS project), since most of the time they correspond to file formats or elements not supported by QGIS. This is, for instance, the case with LAS files used for LiDAR data. The files get created, but you won't see anything new in your QGIS working session.

For all the other types of output, you will find a checkbox that you can use to tell the algorithm whether to load the file once it is generated by the algorithm or not. By default, all files are opened.

Optional outputs are not supported. That is, all outputs are created. However, you can uncheck the corresponding checkbox if you are not interested in a given output, which essentially makes it behave like an optional output (in other words, the layer is created anyway, but if you leave the text box empty, it will be saved to a temporary file and deleted once you exit QGIS).

17.2.3 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.

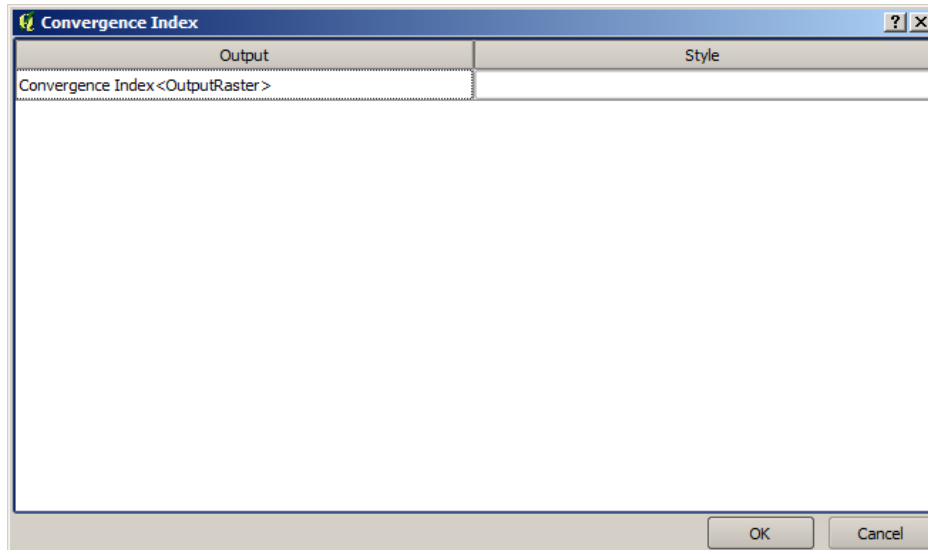
Along with the aforementioned *Output folder* entry, the *General* block contains parameters for setting the default rendering style for output layers (that is, layers generated by using algorithms from any of the framework GUI components). Just create the style you want using QGIS, save it to a file, and then enter the path to that file in the settings so the algorithms can use it. Whenever a layer is loaded by SEXTANTE 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*. You will see a dialog like the one shown next.

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.
- *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.



Gambar 17.15: Rendering Styles 

- *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.

17.3 The graphical modeler

The *graphical modeler* allows you to create complex models using a simple and easy-to-use interface. When working with a GIS, most analysis operations are not isolated, but rather part of a chain of operations instead. Using the graphical modeler, that chain of processes can be wrapped into a single process, so it is as easy and convenient to execute as a single process later on a different set of inputs. No matter how many steps and different algorithms it involves, a model is executed as a single algorithm, thus saving time and effort, especially for larger models.

The modeler can be opened from the processing menu.

The modeler has a working canvas where the structure of the model and the workflow it represents are shown. On the left part of the window, a panel with two tabs can be used to add new elements to the model.

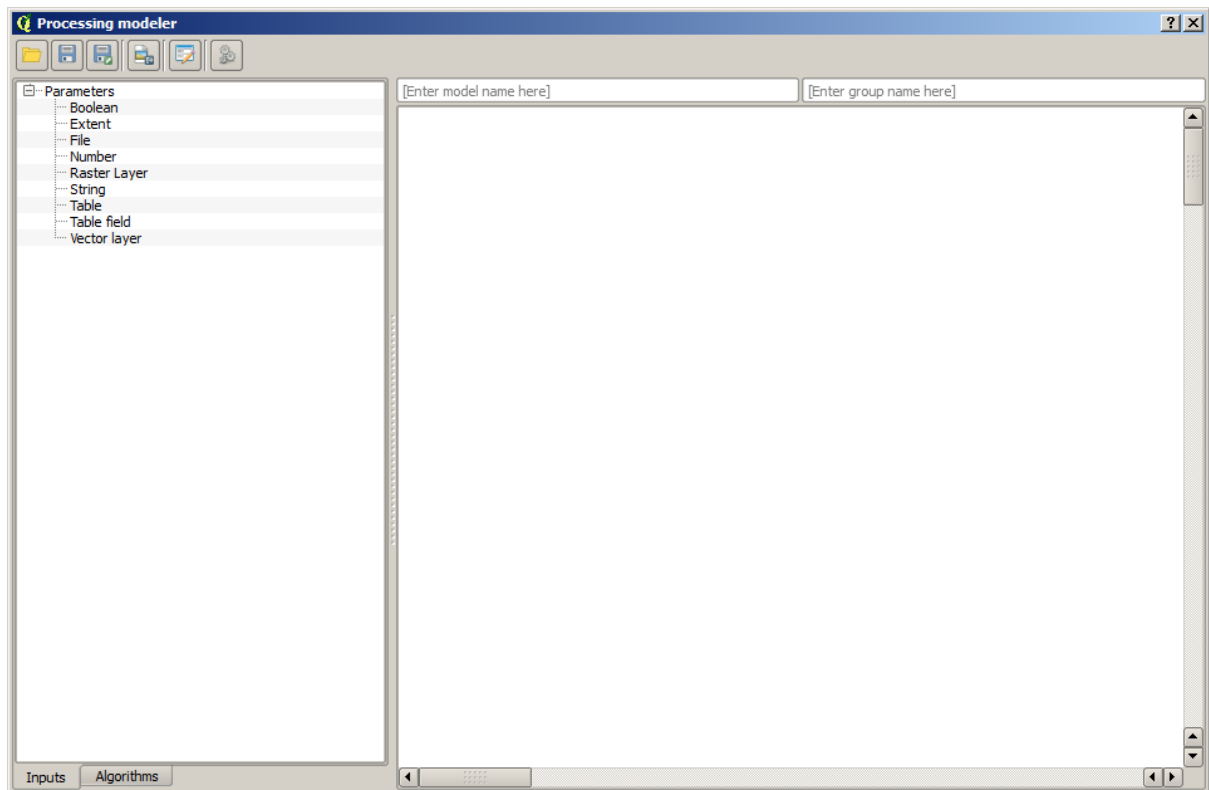
Creating a model involves two steps:


1. *Definition of necessary inputs*. These inputs will be added to the parameters window, so the user can set their values when executing the model. The model itself is an algorithm, so the parameters window is generated automatically as it happens with all the algorithms available in the processing framework.
2. *Definition of the workflow*. Using the input data of the model, the workflow is defined by adding algorithms and selecting how they use those inputs or the outputs generated by other algorithms already in the model.

17.3.1 Definition of inputs

The first step to create a model is to define the inputs it needs. The following elements are found in the *Inputs* tab on the left side of the modeler window:

- Raster layer



Gambar 17.16: Modeler 

- Vector layer
- String
- Table field
- Table
- Extent
- Number
- Boolean
- File

Double-clicking on any of these elements, a dialog is shown to define its characteristics. Depending on the parameter itself, the dialog may contain just one basic element (the description, which is what the user will see when executing the model) or more of them. For instance, when adding a numerical value, as can be seen in the next figure, apart from the description of the parameter, you have to set a default value and a range of valid values.

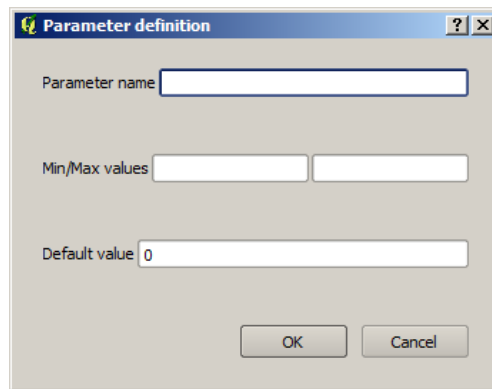
For each added input, a new element is added to the modeler canvas.

You can also add inputs by dragging the input type from the list and dropping it in the modeler canvas, in the position where you want to place it.

17.3.2 Definition of the workflow

Once the inputs have been defined, it is time to define the algorithms to apply on them. Algorithms can be found in the *Algorithms* tab, grouped much in the same way as they are in the toolbox.

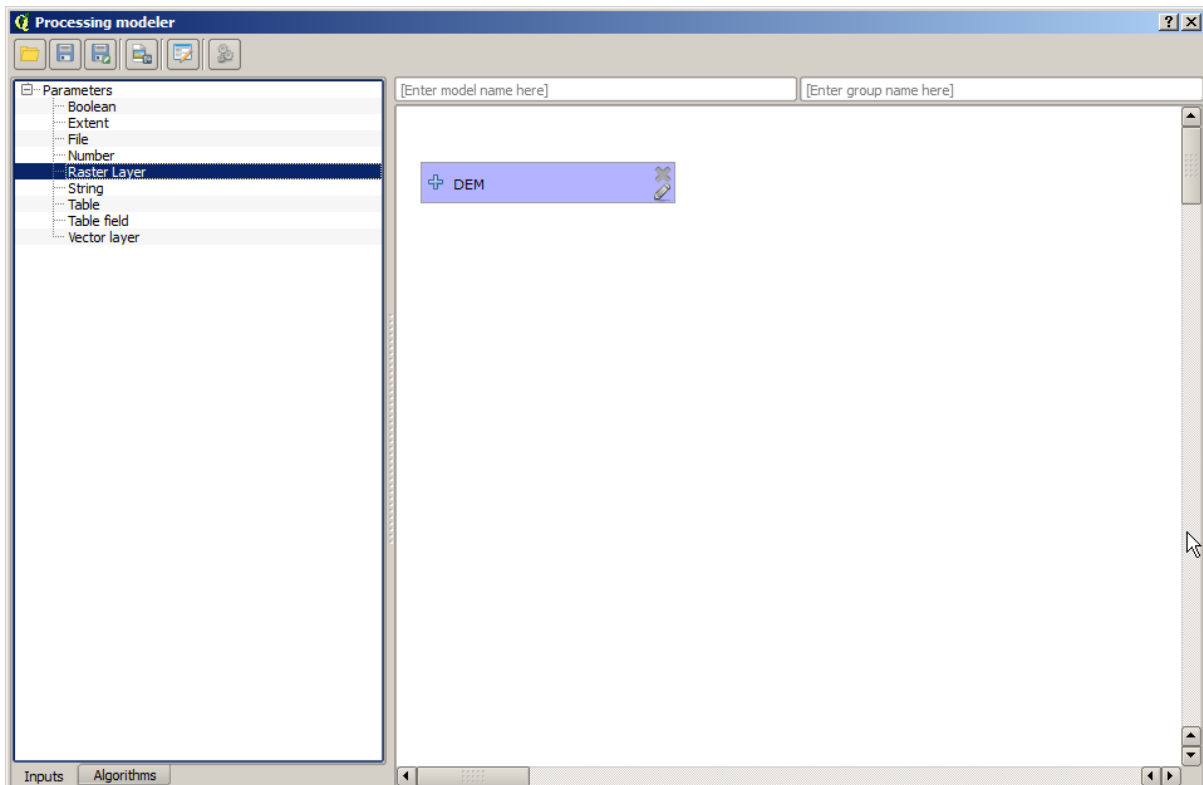
The appearance of the toolbox has two modes here as well: simplified and advanced. However, there is no element to switch between views in the modeler, so you have to do it in the toolbox. The mode that is selected in the toolbox is the one that will be used for the list of algorithms in the modeler.



Gambar 17.17: Model Parameters

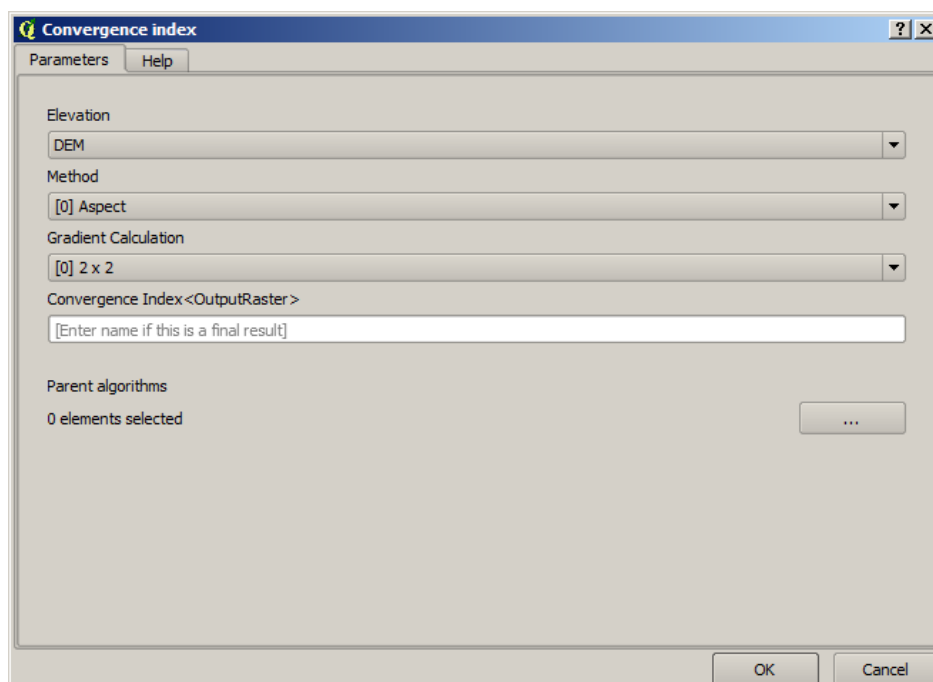


Gambar 17.18: Model Parameters



Gambar 17.19: Model Parameters

To add an algorithm to a model, double-click on its name or drag and drop it, just like it was done when adding inputs. An execution dialog will appear, with a content similar to the one found in the execution panel that is shown when executing the algorithm from the toolbox. The one shown next corresponds to the SAGA 'Convergence index' algorithm, the same example we saw in the section dedicated to the toolbox.



Gambar 17.20: Model Parameters 

As you can see, some differences exist. Instead of the file output box that was used to set the file path for output layers and tables, a simple text box is used here. If the layer generated by the algorithm is just a temporary result that will be used as the input of another algorithm and should not be kept as a final result, just do not edit that text box. Typing anything in it means that the result is final and the text that you supply will be the description for the output, which will be the output the user will see when executing the model.

Selecting the value of each parameter is also a bit different, since there are important differences between the context of the modeler and that of the toolbox. Let's see how to introduce the values for each type of parameter.

- Layers (raster and vector) and tables. These are selected from a list, but in this case, the possible values are not the layers or tables currently loaded in QGIS, but the list of model inputs of the corresponding type, or other layers or tables generated by algorithms already added to the model.
- Numerical values. Literal values can be introduced directly in the text box. But this text box is also a list that can be used to select any of the numerical value inputs of the model. In this case, the parameter will take the value introduced by the user when executing the model.
- String. As in the case of numerical values, literal strings can be typed, or an input string can be selected.
- Table field. The fields of the parent table or layer cannot be known at design time, since they depend on the selection of the user each time the model is executed. To set the value for this parameter, type the name of a field directly in the text box, or use the list to select a table field input already added to the model. The validity of the selected field will be checked at run time.

In all cases, you will find an additional parameter named *Parent algorithms* that is not available when calling the algorithm from the toolbox. This parameter allows you to define the order in which algorithms are executed by explicitly defining one algorithm as a parent of the current one, which will force the parent algorithm to be executed before the current one.

When you use the output of a previous algorithm as the input of your algorithm, that implicitly sets the previous algorithm as parent of the current one (and places the corresponding arrow in the modeler canvas). However, in some cases an algorithm might depend on another one even if it does not use any output object from it (for

instance, an algorithm that executes an SQL sentence on a PostGIS database and another one that imports a layer into that same database). In that case, just select the previous algorithm in the *Parent algorithms* parameter and the two steps will be executed in the correct order.

Once all the parameters have been assigned valid values, click on **[OK]** and the algorithm will be added to the canvas. It will be linked to all the other elements in the canvas, whether algorithms or inputs, that provide objects that are used as inputs for that algorithm.

Elements can be dragged to a different position within the canvas, to change the way the module structure is displayed and make it more clear and intuitive. Links between elements are updated automatically. You can zoom in and out by using the mouse wheel.

You can run your algorithm anytime by clicking on the **[Run]** button. However, in order to use the algorithm from the toolbox, it has to be saved and the modeler dialog closed, to allow the toolbox to refresh its contents.

17.3.3 Saving and loading models

Use the **[Save]** button to save the current model and the **[Open]** button to open any model previously saved. Models are saved with the `.model` extension. If the model has been previously saved from the modeler window, you will not be prompted for a filename. Since there is already a file associated with that model, the same file will be used for any subsequent saves.

Before saving a model, you have to enter a name and a group for it, using the text boxes in the upper part of the window.

Models saved on the `models` folder (the default folder when you are prompted for a filename to save the model) will appear in the toolbox in the corresponding branch. When the toolbox is invoked, it searches the `models` folder for files with the `.model` extension and loads the models they contain. Since a model is itself an algorithm, it can be added to the toolbox just like any other algorithm.

The models folder can be set from the processing configuration dialog, under the *Modeler* group.

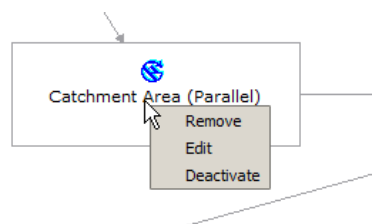
Models loaded from the `models` folder appear not only in the toolbox, but also in the algorithms tree in the *Algorithms* tab of the modeler window. That means that you can incorporate a model as a part of a bigger model, just as you add any other algorithm.

In some cases, a model might not be loaded because not all the algorithms included in its workflow are available. If you have used a given algorithm as part of your model, it should be available (that is, it should appear in the toolbox) in order to load that model. Deactivating an algorithm provider in the processing configuration window renders all the algorithms in that provider unusable by the modeler, which might cause problems when loading models. Keep that in mind when you have trouble loading or executing models.

17.3.4 Editing a model

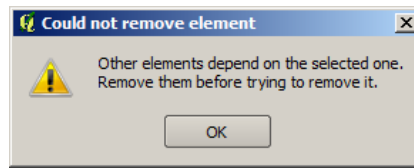
You can edit the model you are currently creating, redefining the workflow and the relationships between the algorithms and inputs that define the model itself.

If you right-click on an algorithm in the canvas representing the model, you will see a context menu like the one shown next:



Gambar 17.21: Modeler Right Click 

Selecting the *Remove* option will cause the selected algorithm to be removed. An algorithm can be removed only if there are no other algorithms depending on it. That is, if no output from the algorithm is used in a different one as input. If you try to remove an algorithm that has others depending on it, a warning message like the one you can see below will be shown:



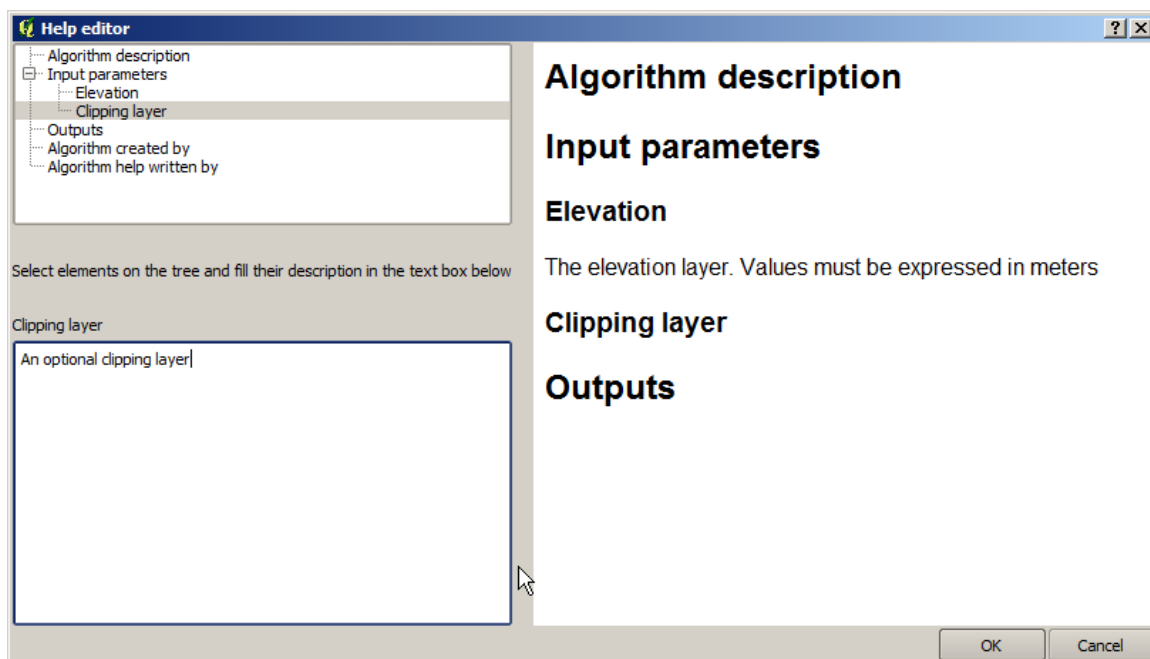
Gambar 17.22: Cannot Delete Algorithm

Selecting the *Edit* option or simply double-clicking on the algorithm icon will show the parameters dialog of the algorithm, so you can change the inputs and parameter values. Not all input elements available in the model will appear in this case as available inputs. Layers or values generated at a more advanced step in the workflow defined by the model will not be available if they cause circular dependencies.

Select the new values and then click on the [OK] button as usual. The connections between the model elements will change accordingly in the modeler canvas.

17.3.5 Editing model help files and meta-information

You can document your models from the modeler itself. Just click on the [Edit model help] button and a dialog like the one shown next will appear.



Gambar 17.23: Help Edition

On the right-hand side, you will see a simple HTML page, created using the description of the input parameters and outputs of the algorithm, along with some additional items like a general description of the model or its author. The first time you open the help editor, all these descriptions are empty, but you can edit them using the elements on the left-hand side of the dialog. Select an element on the upper part and then write its description in the text box below.

Model help is saved in a file in the same folder as the model itself. You do not have to worry about saving it, since it is done automatically.

17.3.6 About available algorithms

You might notice that some algorithms that can be executed from the toolbox do not appear in the list of available algorithms when you are designing a model. To be included in a model, an algorithm must have a correct semantic, so as to be properly linked to others in the workflow. If an algorithm does not have such a well-defined semantic (for instance, if the number of output layers cannot be known in advance), then it is not possible to use it within a model, and thus, it does not appear in the list of algorithms that you can find in the modeler dialog.

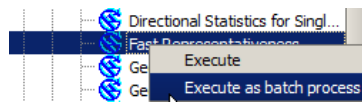
Additionally, you will see some algorithms in the modeler that are not found in the toolbox. These algorithms are meant to be used exclusively as part of a model, and they are of no interest in a different context. The 'Calculator' algorithm is an example of that. It is just a simple arithmetic calculator that you can use to modify numerical values (entered by the user or generated by some other algorithm). This tool is really useful within a model, but outside of that context, it doesn't make too much sense.

17.4 Antarmuka memproses batch

17.4.1 Pengantar

Semua algoritma (termasuk model) dapat dijalankan sebagai proses batch. Artinya, mereka dapat dijalankan menggunakan tidak hanya satu set masukan, namun beberapa dari mereka, melaksanakan algoritma sebanyak yang diperlukan. Hal ini berguna saat memproses data dalam jumlah besar, karena tidak perlu meluncurkan algoritma berkali-kali dari toolbox.

Untuk menjalankan algoritma sebagai proses batch, klik kanan pada namanya dalam kotak alat dan pilih opsi *Eksekusi sebagai proses batch* di menu pop-up yang akan muncul.



Gambar 17.24: Klik Kanan Memproses Batch 

17.4.2 Tabel parameter

Pelaksana proses batch mirip dengan melakukan eksekusi tunggal dari suatu algoritma. Nilai parameter harus didefinisikan, tetapi dalam kasus ini kita tidak perlu hanya nilai tunggal untuk masing-masing parameter, tapi satu set mereka sebagai gantinya, satu untuk setiap kali algoritma harus dieksekusi. Nilai akan diperkenalkan menggunakan tabel seperti yang ditunjukkan berikutnya.

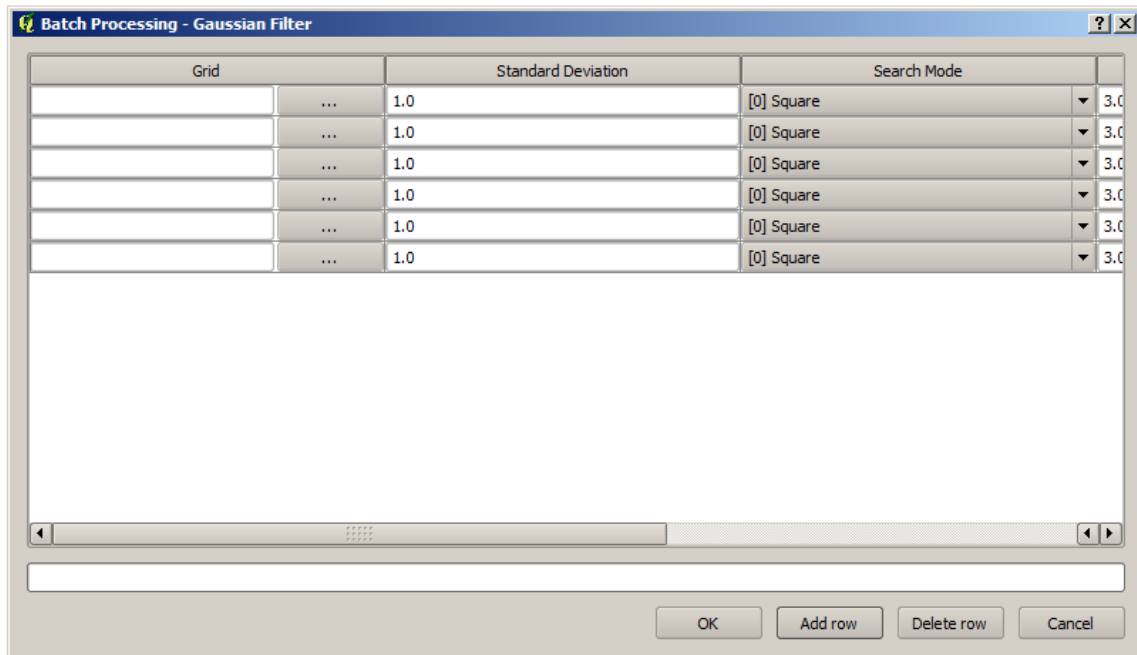
Setiap baris dari tabel ini merupakan eksekusi tunggal algoritma, dan setiap sel berisi nilai salah satu parameter. Hal ini mirip dengan dialog parameter yang Anda lihat ketika menjalankan sebuah algoritma dari toolbox, tapi dengan pengaturan yang berbeda.

Secara default, tabel berisi hanya dua baris. Anda dapat menambahkan atau menghapus baris menggunakan tombol pada bagian bawah jendela.

Setelah ukuran tabel telah ditetapkan, harus diisi dengan nilai-nilai yang diinginkan.

17.4.3 Mengarsipkan tabel parameter

Bagi kebanyakan parameter, menetapkan nilai sepele. Cukup ketik nilai atau pilih dari daftar pilihan yang tersedia, tergantung pada jenis parameter.



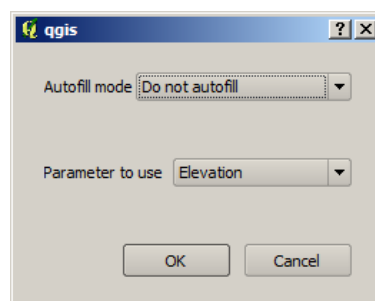
Gambar 17.25: Memproses Batch

Perbedaan utama yang ditemukan untuk parameter yang mewakili lapis atau tabel, dan untuk path berkas keluaran. Mengenai masukan lapis dan tabel, ketika sebuah algoritma dijalankan sebagai bagian dari proses batch, obyek masukan data yang diambil langsung dari berkas, dan bukan dari aturan mereka yang sudah dibuka di QGIS. Untuk alasan ini, algoritma dapat dijalankan sebagai proses batch, bahkan jika tidak ada obyek data sama sekali yang dibuka dan algoritma tidak dapat dijalankan dari toolbox.

Nama berkas untuk obyek masukan data diperkenalkan langsung dengan mengetik atau, lebih nyaman, mengklik tombol (...) di sebelah kanan dari sel, yang menunjukkan dialog pemilih berkas. Beberapa berkas dapat dipilih sekaligus. Jika parameter masukan mewakili obyek data tunggal dan beberapa berkas yang dipilih, masing-masing dari mereka akan dimasukkan ke dalam baris yang terpisah, menambahkan yang baru jika diperlukan. Jika parameter mewakili banyak masukan, semua berkas yang dipilih akan ditambahkan ke satu sel, dipisahkan oleh titik koma (;).

Obyek data keluaran akan selalu disimpan ke berkas dan, tidak seperti ketika menjalankan algoritma dari toolbox, menyimpan ke berkas sementara tidak diizinkan. Anda dapat mengetik nama secara langsung atau menggunakan dialog pemilih berkas yang muncul saat mengklik tombol yang menyertainya.

Setelah Anda memilih berkas, dialog baru ditampilkan untuk memungkinkan autocompletion dari sel-sel lain dalam kolom yang sama (parameter yang sama).

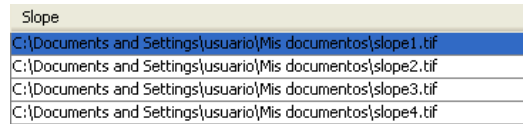


Gambar 17.26: Simpan Memproses Batch

Jika nilai default ('Jangan AutoComplete') yang dipilih, itu hanya akan menempatkan nama berkas yang dipilih dalam sel yang dipilih dari tabel parameter. Jika salah satu opsi lain yang dipilih, semua sel di bawah satu yang

dipilih otomatis akan diisi berdasarkan kriteria yang ditetapkan. Dengan cara ini, jauh lebih mudah untuk mengisi tabel, dan proses batch dapat didefinisikan dengan sedikit usaha.

Mengisi otomatis dapat dilakukan dengan hanya menambahkan angka korelatif untuk path berkas yang dipilih, atau dengan menambahkan nilai dari bidang lain pada baris yang sama. Hal ini sangat berguna untuk penamaan obyek data keluaran sesuai dengan yang masukan.



Gambar 17.27: Path Berkas Memproses Batch

17.4.4 Mengeksekusi proses batch

Untuk melaksanakan proses batch setelah Anda memperkenalkan semua nilai yang diperlukan, klik **[OK]**. Kemajuan tugas batch global akan ditampilkan dalam bar kemajuan di bagian bawah dialog.

17.5 Using processing algorithms from the console

The console allows advanced users to increase their productivity and perform complex operations that cannot be performed using any of the other GUI elements of the processing framework. Models involving several algorithms can be defined using the command-line interface, and additional operations such as loops and conditional sentences can be added to create more flexible and powerful workflows.

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.

The code that you can execute from the Python console, even if it does not call any specific processing method, can be converted into a new algorithm that you can later call from the toolbox, the graphical modeler or any other component, just like you do with any other algorithm. In fact, some algorithms that you can find in the toolbox are simple scripts.

In this section, we will see how to use processing algorithms from the QGIS Python console, and also how to write algorithms using Python.

17.5.1 Calling algorithms from the Python console

The first thing you have to do is to import the processing functions with the following line:

```
>>> 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 `runalg()` 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 `algslist()` method. Type the following line in your console:

```
>>> processing.algslist()
```

You will see something like this.

```
Accumulated Cost (Anisotropic)----->saga:accumulatedcost (anisotropic)
Accumulated Cost (Isotropic)----->saga:accumulatedcost (isotropic)
Add Coordinates to points----->saga:addcoordinatestopoints
Add Grid Values to Points----->saga:addgridvaluestopoints
Add Grid Values to Shapes----->saga:addgridvaluestoshapes
Add Polygon Attributes to Points----->saga:addpolygonattributestopoints
Aggregate----->saga:aggregate
Aggregate Point Observations----->saga:aggregatepointobservations
Aggregation Index----->saga:aggregationindex
Analytical Hierarchy Process----->saga:analyticalhierarchyprocess
Analytical Hillshading----->saga:analyticalhillshading
Average With Mask 1----->saga:averagewithmask1
Average With Mask 2----->saga:averagewithmask2
Average With Threshold 1----->saga:averagewiththeshold1
Average With Threshold 2----->saga:averagewiththeshold2
Average With Threshold 3----->saga:averagewiththeshold3
B-Spline Approximation----->saga:b-splineapproximation
...
```

That's a list of all the available algorithms, alphabetically ordered, along with their corresponding command-line names.

You can use a string as a parameter for this method. Instead of returning the full list of algorithms, it will only display those that include that string. If, for instance, you are looking for an algorithm to calculate slope from a DEM, type `alglist("slope")` to get the following result:

```
DTM Filter (slope-based)----->saga:dtmfilter (slope-based)
Downslope Distance Gradient----->saga:downslopedistancegradient
Relative Heights and Slope Positions----->saga:relativeheightsandslopepositions
Slope Length----->saga:slopelength
Slope, Aspect, Curvature----->saga:slopeaspectcurvature
Upslope Area----->saga:upslopearea
Vegetation Index[slope based]----->saga:vegetationindex[slopebased]
```

This result might change depending on the algorithms you have available.

It is easier now to find the algorithm you are looking for and its command-line name, in this case `saga:slopeaspectcurvature`.

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 and the order in which they have to be passed when calling the `runalg()` 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 `alghelp(name_of_the_algorithm)` method. Use the command-line name of the algorithm, not the full descriptive name.

Calling the method with `saga:slopeaspectcurvature` as parameter, you get the following description:

```
>>> processing.alghelp("saga:slopeaspectcurvature")
ALGORITHM: Slope, Aspect, Curvature
  ELEVATION <ParameterRaster>
  METHOD <ParameterSelection>
  SLOPE <OutputRaster>
  ASPECT <OutputRaster>
  CURV <OutputRaster>
  HCURV <OutputRaster>
  VCURV <OutputRaster>
```

Now you have everything you need to run any algorithm. As we have already mentioned, there is only one single command to execute algorithms: `runalg()`. Its syntax is as follows:

```
>>> processing.runalg(name_of_the_algorithm, param1, param2, ..., paramN,
  Output1, Output2, ..., OutputN)
```

The list of parameters and outputs to add depends on the algorithm you want to run, and is exactly the list that the `alghelp()` method gives you, in the same order as shown.

Depending on the type of parameter, values are introduced differently. The next list gives a quick review of how to introduce values for each type of input parameter:

- 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. If the input is optional and you do not want to use any data object, use `None`.
- Selection. If an algorithm has a selection parameter, the value of that parameter should be entered using an integer value. To know the available options, you can use the `algorithms()` command, as shown in the following example:

```
>>> processing.algorithms("saga:slopeaspectcurvature")
METHOD (Method)
0 - [0] Maximum Slope (Travis et al. 1975)
1 - [1] Maximum Triangle Slope (Tarboton 1997)
2 - [2] Least Squares Fitted Plane (Horn 1981, Costa-Cabral & Burgess 1996)
3 - [3] Fit 2.Degree Polynom (Bauer, Rohdenburg, Bork 1985)
4 - [4] Fit 2.Degree Polynom (Heerdegen & Beran 1982)
5 - [5] Fit 2.Degree Polynom (Zevenbergen & Thorne 1987)
6 - [6] Fit 3.Degree Polynom (Haralick 1983)
```

In this case, the algorithm has one such parameter, with seven options. Notice that ordering is zero-based.

- Multiple input. The value is a string with input descriptors separated by semicolons (;). As in the case of single layers or tables, each input descriptor can be the data object name, or its file path.
- Table Field from XXX. Use a string with the name of the field to use. This parameter is case-sensitive.
- Fixed Table. Type the list of all table values separated by commas (,) and enclosed between quotes ("). Values start on the upper row and go from left to right. You can also use a 2-D array of values representing the table.
- CRS. Enter the EPSG code number of the desired CRS.
- Extent. You must use a string with `xmin`, `xmax`, `ymin` and `ymax` values separated by commas (,).

Boolean, file, string and numerical parameters do not need any additional explanations.

Input parameters such as strings, booleans, or numerical values have default values. To use them, specify `None` in the corresponding parameter entry.

For output data objects, type the file path to be used to save it, just as it is done from the toolbox. If you want to save the result to a temporary file, use `None`. 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. If you want to add an output to the map canvas, you have to do it yourself after running the algorithm. To do so, you can use QGIS API commands, or, even easier, use one of the handy methods provided for such tasks.

The `runalg` method returns a dictionary with the output names (the ones shown in the algorithm description) as keys and the file paths of those outputs as values. You can load those layers by passing the corresponding file paths to the `load()` method.

17.5.2 Additional functions for handling data

Apart from the functions used to call algorithms, importing the `processing` package will also import some additional functions that make it easier to work with data, particularly vector data. They are just convenience

functions that wrap some functionality from the QGIS API, usually with a less complex syntax. These functions should be used when developing new algorithms, as they make it easier to operate with input data.

Below is a list of some of these commands. More information can be found in the classes under the `processing/tools` package, and also in the example scripts provided with QGIS.

- `getObject(obj)`: Returns a QGIS object (a layer or table) from the passed object, which can be a filename or the name of the object in the QGIS Table of Contents.
- `values(layer, fields)`: Returns the values in the attributes table of a vector layer, for the passed fields. Fields can be passed as field names or as zero-based field indices. Returns a dict of lists, with the passed field identifiers as keys. It considers the existing selection.
- `features(layer)`: Returns an iterator over the features of a vector layer, considering the existing selection.
- `uniqueValues(layer, field)`: Returns a list of unique values for a given attribute. Attributes can be passed as a field name or a zero-based field index. It considers the existing selection.

17.5.3 Creating scripts and running them from the toolbox

You can create your own algorithms by writing the corresponding Python code and adding a few extra lines to supply additional information needed to define the semantics of the algorithm. You can find a *Create new script* menu under the *Tools* group in the *Script* algorithms block of the toolbox. Double-click on it to open the script editing dialog. 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 `.py` extension will automatically create the corresponding algorithm.

The name of the algorithm (the one you will see in the toolbox) is created from the filename, removing its extension and replacing low hyphens with blank spaces.

Let's have a look at the following code, which calculates the Topographic Wetness Index (TWI) directly from a DEM.

```
##dem=raster
##twi=output
ret_slope = processing.runalg("saga:slopeaspectcurvature", dem, 0, None,
                             None, None, None, None)
ret_area = processing.runalg("saga:catchmentarea(mass-fluxmethod)", dem,
                             0, False, False, False, None, None, None)
processing.runalg("saga:topographicwetnessindex(twi)", ret_slope['SLOPE'],
                 ret_area['AREA'], None, 1, 0, twi)
```

As you can see, the calculation involves three algorithms, all of them coming from SAGA. The last one calculates the TWI, but it needs a slope layer and a flow accumulation layer. We do not have these layers, but since we have the DEM, we can calculate them by calling the corresponding SAGA algorithms.

The part of the code where this processing takes place is not difficult to understand if you have read the previous sections in this 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 start with a double Python comment symbol (`##`) and have the following structure:

```
[parameter_name]=[parameter_type] [optional_values]
```

Here is a list of all the parameter types that are supported in processing scripts, their syntax and some examples.

- `raster`. A raster layer.
- `vector`. A vector layer.
- `table`. A table.
- `number`. A numerical value. A default value must be provided. For instance, `depth=number 2.4`.

- `string`. A text string. As in the case of numerical values, a default value must be added. For instance, `name=string Victor`.
- `boolean`. A boolean value. Add `True` or `False` after it to set the default value. For example, `verbose=boolean True`.
- `multiple raster`. A set of input raster layers.
- `multiple vector`. A set of input vector layers.
- `field`. A field in the attributes table of a vector layer. The name of the layer has to be added after the `field` tag. For instance, if you have declared a vector input with `mylayer=vector`, you could use `myfield=field mylayer` to add a field from that layer as parameter.
- `folder`. A folder.
- `file`. A filename.

The parameter name is the name that will be shown to the user when executing the algorithm, and also the variable name to use in the script code. The value entered by the user for that parameter will be assigned to a variable with that name.

When showing the name of the parameter to the user, the name will be edited to improve its appearance, replacing low hyphens with spaces. So, for instance, if you want the user to see a parameter named `A numerical value`, you can use the variable name `A_numerical_value`.

Layers and table values are strings containing the file path of the corresponding object. To turn them into a QGIS object, you can use the `processing.getObjectFromUri()` function. Multiple inputs also have a string value, which contains the file paths to all selected object, separated by semicolons (;).

Outputs are defined in a similar manner, using the following tags:

- `output raster`
- `output vector`
- `output table`
- `output html`
- `output file`
- `output number`
- `output string`

The value assigned to the output variables is always a string with a file path. It will correspond to a temporary file path in case the user has not entered any output filename.

When you declare an output, the algorithm will try to add it to QGIS once it is finished. That is why, although the `runalg()` method does not load the layers it produces, the final TWI layer will be loaded (using the case of our previous example), since it is saved to the file entered by the user, which is the value of the corresponding output.

Do not use the `load()` method in your script algorithms, just when working with the console line. 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 its syntax (as defined by the tags explained above) will not match what the algorithm really creates.

Hidden outputs (numbers and strings) do not have a value. Instead, you have to assign a value to them. To do so, just set the value of a variable with the name you used to declare that output. For instance, if you have used this declaration,

```
##average=output number
```

the following line will set the value of the output to 5:

```
average = 5
```

In addition to the tags for parameters and outputs, you can also define the group under which the algorithm will be shown, using the `group` tag.

If your algorithm takes a long time to process, it is a good idea to inform the user. You have a global named `progress` available, with two possible methods: `setText(text)` and `setPercentage(percent)` to modify the progress text and the progress bar.

Several examples are provided. Please check them to see real examples of how to create algorithms using the processing framework classes. You can right-click on any script algorithm and select *Edit script* to edit its code or just to see it.

17.5.4 Documenting your scripts

As in the case of models, you can create additional documentation for your scripts, to explain what they do and how to use them. In the script editing dialog, you will find an **[Edit script help]** button. Click on it and it will take you to the help editing dialog. Check the section about the graphical modeler to know more about this dialog and how to use it.

Help files are saved in the same folder as the script itself, adding the `.help` extension to the filename. Notice that you can edit your script's help before saving the script for the first time. If you later close the script editing dialog without saving the script (i.e., you discard it), the help content you wrote will be lost. If your script was already saved and is associated to a filename, saving the help content is done automatically.

17.5.5 Pre- and post-execution script hooks

Scripts can also be used to set pre- and post-execution hooks that are run before and after an algorithm is run. This can be used to automate tasks that should be performed whenever an algorithm is executed.

The syntax is identical to the syntax explained above, but an additional global variable named `alg` is available, representing the algorithm that has just been (or is about to be) executed.

In the *General* group of the processing configuration dialog, you will find two entries named *Pre-execution script file* and *Post-execution script file* where the filename of the scripts to be run in each case can be entered.

.

17.6 Manajer riwayat

17.6.1 Pemrosesan riwayat

Setiap kali Anda mengeksekusi sebuah algoritma, informasi tentang proses disimpan dalam manajer riwayat. Seiring dengan parameter yang digunakan, tanggal dan waktu eksekusi juga disimpan.

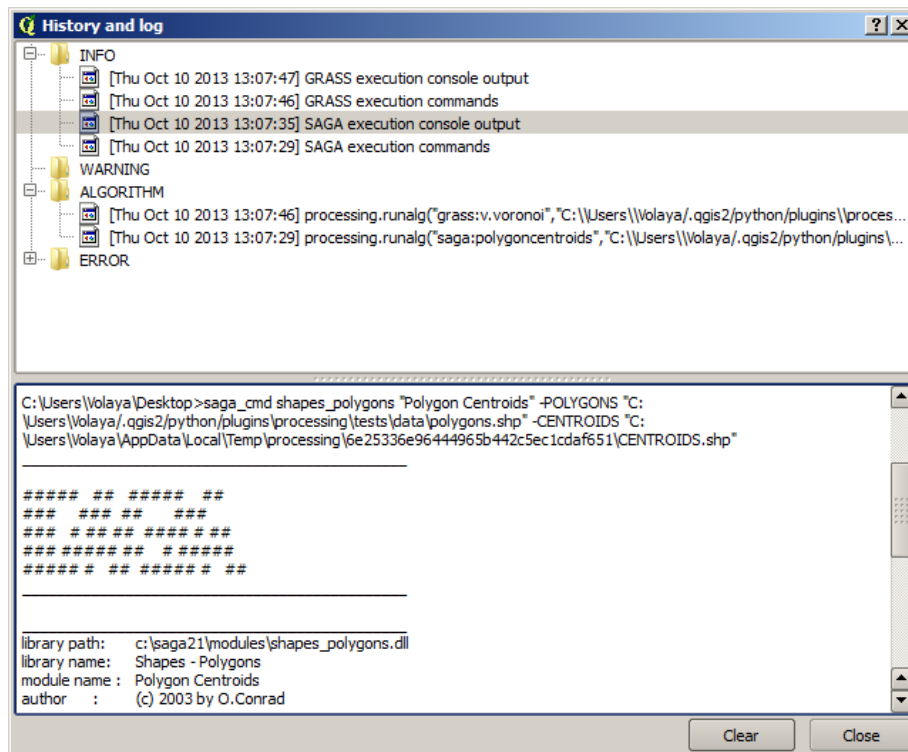
Dengan cara ini, mudah untuk melacak dan mengendalikan semua pekerjaan yang telah dikembangkan dengan menggunakan kerangka pengolahan, dan mudah mereproduksi itu.


Manajer riwayat adalah satu set entri registri dikelompokkan berdasarkan tanggal eksekusi mereka, sehingga lebih mudah untuk menemukan informasi tentang algoritma yang dijalankan pada saat tertentu.

Informasi proses disimpan sebagai ekspresi baris-perintah, bahkan jika algoritma diluncurkan dari kotakalat. Hal ini juga berguna bagi mereka belajar bagaimana menggunakan antarmuka baris perintah, karena mereka dapat memanggil algoritma menggunakan kotak alat dan kemudian memeriksa manajer riwayat melihat bagaimana algoritma yang sama bisa dipanggil dari baris perintah.

Selain menjelajah entri dalam registri, Anda juga dapat jalankan kembali proses dengan hanya mengklik dua kali pada entri yang sesuai.

Seiring dengan rekaman algoritma eksekusi, kerangka pengolahan berkomunikasi dengan pengguna melalui kelompok lain dari registri, yaitu *Error*, *Peringatan* dan *Informasi*. Dalam hal sesuatu yang tidak bekerja dengan baik, akan melihat *Error* dapat membantu Anda untuk melihat apa yang terjadi. Jika Anda mendapatkan kontak



Gambar 17.28: Riwayat 

dengan pengembang melaporkan bug atau kesalahan, informasi dalam kelompok akan sangat berguna baginya atau untuk mencari tahu apa yang salah.

Algoritma pihak ketiga biasanya dilakukan dengan memanggil antarmuka baris perintah mereka, yang berkomunikasi dengan pengguna melalui konsol. Meskipun konsol yang tidak ditampilkan, dump penuh disimpan dalam grup *Informasi* setiap kali Anda menjalankan salah satu algoritma. Jika, misalnya, Anda mengalami masalah mengeksekusi algoritma SAGA, mencari entri bernama ‘keluaran konsol eksekusi SAGA’ untuk memeriksa semua pesan yang dihasilkan oleh SAGA dan coba mencari tahu di mana masalahnya.

Beberapa algoritma, bahkan jika mereka dapat menghasilkan hasil dengan masukan data yang diberikan, bisa menambahkan komentar atau informasi tambahan ke blok *Peringatan* jika mereka mendeteksi potensi masalah dengan data, dalam rangka untuk memperingatkan Anda. Pastikan Anda memeriksa pesan tersebut jika Anda mengalami hasil yang tidak diharapkan.

17.7 Writing new Processing algorithms as python scripts

You can create your own algorithms by writing the corresponding Python code and adding a few extra lines to supply additional information needed to define the semantics of the algorithm. You can find a *Create new script* menu under the *Tools* group in the *Script* algorithms block of the toolbox. Double-click on it to open the script edition dialog. That’s where you should type your code. Saving the script from there in the *scripts* folder (the default one when you open the save file dialog), with *.py* extension, will automatically create the corresponding algorithm.

The name of the algorithm (the one you will see in the toolbox) is created from the filename, removing its extension and replacing low hyphens with blank spaces.

Let’s have the following code, which calculates the Topographic Wetness Index (TWI) directly from a DEM

```
##dem=raster
##twi=output raster
ret_slope = processing.runalg("saga:slopeaspectcurvature", dem, 0, None,
                             None, None, None, None)
```

```
ret_area = processing.runalg("saga:catchmentarea", dem,
    0, False, False, False, False, None, None, None, None)
processing.runalg("saga:topographicwetnessindextwi", ret_slope['SLOPE'],
    ret_area['AREA'], None, 1, 0, twi)
```

As you can see, it involves 3 algorithms, all of them coming from SAGA. The last one of them calculates the TWI, but it needs a slope layer and a flow accumulation layer. We do not have these ones, but since we have the DEM, we can calculate them calling the corresponding SAGA algorithms.

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 start with a double Python comment symbol (##) and have the following structure

```
[parameter_name]=[parameter_type] [optional_values]
```

Here is a list of all the parameter types that are supported in processign scripts, their syntax and some examples.

- `raster`. A raster layer
- `vector`. A vector layer
- `table`. A table
- `number`. A numerical value. A default value must be provided. For instance, `depth=number 2.4`
- `string`. A text string. As in the case of numerical values, a default value must be added. For instance, `name=string Victor`
- `longstring`. Same as `string`, but a larger text box will be shown, so it is better suited for long strings, such as for a script expecting a small code snippet.
- `boolean`. A boolean value. Add `True` or `False` after it to set the default value. For example, `verbose=boolean True`.
- `multiple raster`. A set of input raster layers.
- `multiple vector`. A set of input vector layers.
- `field`. A field in the attributes table of a vector layer. The name of the layer has to be added after the `field` tag. For instance, if you have declared a vector input with `mylayer=vector`, you could use `myfield=field mylayer` to add a field from that layer as parameter.
- `folder`. A folder
- `file`. A filename
- `crs`. A Coordinate Reference System

The parameter name is the name that will be shown to the user when executing the algorithm, and also the variable name to use in the script code. The value entered by the user for that parameter will be assigned to a variable with that name.

When showing the name of the parameter to the user, the name will be edited it to improve its appearance, replacing low hyphens with spaces. So, for instance, if you want the user to see a parameter named `A numerical value`, you can use the variable name `A_numerical_value`.

Layers and tables values are strings containing the filepath of the corresponding object. To turn them into a QGIS object, you can use the `processing.getObjectFromUri()` function. Multiple inputs also have a string value, which contains the filepaths to all selected objects, separated by semicolons (`;`).

Outputs are defined in a similar manner, using the following tags:

- `output raster`
- `output vector`
- `output table`

- `output html`
- `output file`
- `output number`
- `output string`
- `output extent`

The value assigned to the output variables is always a string with a filepath. It will correspond to a temporary filepath in case the user has not entered any output filename.

In addition to the tags for parameters and outputs, you can also define the group under which the algorithm will be shown, using the `group` tag.

The last tag that you can use in your script header is `##nomodeler`. Use that when you do not want your algorithm to be shown in the modeler window. This should be used for algorithms that do not have a clear syntax (for instance, if the number of layers to be created is not known in advance, at design time), which make them unsuitable for the graphical modeler

17.8 Handing data produced by the algorithm

When you declare an output representing a layer (raster, vector or table), the algorithm will try to add it to QGIS once it is finished. That is the reason why, although the `runalg()` method does not load the layers it produces, the final *TWI* layer will be loaded, since it is saved to the file entered by the user, which is the value of the corresponding output.

Do not use the `load()` method in your script algorithms, but just when working with the console line. 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 its syntax (as defined by the tags explained above) will not match what the algorithm really creates.

Hidden outputs (numbers and strings) do not have a value. Instead, it is you who has to assign a value to them. To do so, just set the value of a variable with the name you used to declare that output. For instance, if you have used this declaration,

```
##average=output number
```

the following line will set the value of the output to 5:

```
average = 5
```

17.9 Communicating with the user

If your algorithm takes a long time to process, it is a good idea to inform the user. You have a global named `progress` available, with two available methods: `setText(text)` and `setPercentage(percent)` to modify the progress text and the progress bar.

If you have to provide some information to the user, not related to the progress of the algorithm, you can use the `setInfo(text)` method, also from the `progress` object.

If your script has some problem, the correct way of propagating it is to raise an exception of type `GeoAlgorithmExecutionException()`. You can pass a message as 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...)

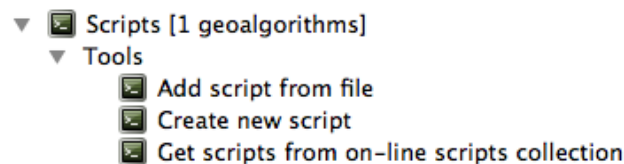
17.10 Documenting your scripts

As in the case of models, you can create additional documentation for your script, to explain what they do and how to use them. In the script editing dialog you will find a **[Edit script help]** button. Click on it and it will take you to the help editing dialog. Check the chapter about the graphical modeler to know more about this dialog and how to use it.

Help files are saved in the same folder as the script itself, adding the `.help` extension to the filename. Notice that you can edit your script's help before saving it for the first time. If you later close the script editing dialog without saving the script (i.e. you discard it), the help content you wrote will be lost. If your script was already saved and is associated to a filename, saving is done automatically.

17.11 Example scripts

Several examples are available in the on-line collection of scripts, which you can access by selecting the *Get script from on-line script collection* tool under the *Scripts/tools* entry in the toolbox.



Please, check them to see real examples of how to create algorithms using the processing framework classes. You can right-click on any script algorithm and select *Edit script* to edit its code or just to see it.

17.12 Best practices for writing script algorithms

Here's a quick summary of ideas to consider when creating your script algorithms and, especially, if you want to share 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.

- Do not load resulting layers. Let Processing handle your results and load your layers if needed.
- Always declare the outputs your algorithm creates. Avoid things such as declaring one output and then using the destination filename set for that output to create a collection of them. That will break the correct semantics of the algorithm and make it impossible to use it safely in the modeler. If you have to write an algorithm like that, make sure you add the `##nomodeler` tag.
- Do not show message boxes or use any GUI element from the script. If you want to communicate with the user, use the `setInfo()` method or throw an `GeoAlgorithmExecutionException`
- As a rule of thumb, do not forget that your algorithm might be executed in a context other than the Processing toolbox.

17.13 Pre- and post-execution script hooks

Scripts can also be used to set pre- and post-execution hooks that are run before and after an algorithm is run. This can be used to automate tasks that should be performed whenever an algorithm is executed.

The syntax is identical to the syntax explained above, but an additional global variable named `alg` is available, representing the algorithm that has just been (or is about to be) executed.

In the *General* group of the processing config dialog you will find two entries named *Pre-execution script file* and *Post-execution script file* where the filename of the scripts to be run in each case can be entered.

17.14 Configuring external applications

The processing framework can be extended using additional applications. Currently, SAGA, GRASS, OTB (Orfeo Toolbox) and R are supported, along with some other command-line applications that provide spatial data analysis functionalities. Algorithms relying on an external application are managed by their own algorithm provider.

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 geospatial algorithm.

By default, all algorithms that rely on an external application not shipped with QGIS are not enabled. You can enable them in the configuration dialog. Make sure that the corresponding application is already installed in your system. Enabling an algorithm provider without installing the application it needs will cause the algorithms to appear in the toolbox, but an error will be thrown when you try to execute them.

This is because the algorithm descriptions (needed to create the parameters dialog and provide the information needed about the algorithm) are not included with each application, but with QGIS instead. That is, they are part of QGIS, so you have them in your installation even if you have not installed any other software. Running the algorithm, however, needs the application binaries to be installed in your system.

17.14.1 A note for Windows users

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, GRASS and OTB in your system and configure them so they can be run from QGIS. All the algorithms in the simplified view of the toolbox will be ready to be run without needing any further configuration. If installing through OSGeo4W application, make sure you select for installation SAGA and OTB as well.

If you want to know more about how these providers work, or if you want to use some algorithms not included in the simplified toolbox (such as R scripts), keep on reading.

17.14.2 A note on file formats

When using an external software, opening a file in QGIS does not mean that it can be opened and processed as well 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 history and log dialog) to know more about what is going wrong.

Using GRASS raster layers is, for instance, one case in which you might have trouble and not be able to complete your work if you call an external algorithm using such a layer as input. For this reason, these layers will not appear as available to algorithms.

You should, however, find no problems at all 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 if the layer has a large size, so do not be surprised if it takes more time to process a layer from a DB connection than it does to process one of a similar size stored in a shapefile.

Providers not using external applications can process any layer that you can open in QGIS, since they open it for analysis through QGIS.

Regarding output formats, all formats supported by QGIS as output can be used, both for raster and vector layers. Some providers do not support certain formats, but all can export to common raster layer formats that can later be transformed by QGIS automatically. As in the case of input layers, if this conversion is needed, that might increase the processing time.

If the extension of the filename specified when calling an algorithm does not match the extension of any of the formats supported by QGIS, then a suffix will be added to set a default format. In the case of raster layers, the `.tif` extension is used, while `.shp` is used for vector layers.

17.14.3 A note on vector layer selections

External applications may also be made aware of the selections that exist in vector layers within QGIS. However, that requires rewriting all input vector layers, just as if they were originally in a format not supported by the external application. Only when no selection exists, or the *Use only selected features* option is not enabled in the processing general configuration, can a layer be directly passed to an external application.

In other cases, exporting only selected features is needed, which causes execution times to be longer.

SAGA

SAGA algorithms can be run from QGIS if you have SAGA installed in your system and you configure the processing framework properly so it can find SAGA executables. In particular, the SAGA command-line executable is needed to run SAGA algorithms.

If you are running Windows, both the stand-alone installer and the OSGeo4W installer include SAGA along with QGIS, and the path is automatically configured, so there is no need to do anything else.

If you have installed SAGA yourself (remember, you need version 2.1), the path to the SAGA executable must be configured. To do this, open the configuration dialog. In the *SAGA* block, you will find a setting named *SAGA Folder*. Enter the path to the folder where SAGA is installed. Close the configuration dialog, and now you are ready to run SAGA algorithms from QGIS.

If you are running Linux, SAGA binaries are not included with SEXTANTE, so you have to download and install the software yourself. Please check the SAGA website for more information. SAGA 2.1 is needed.

In this case, there is no need to configure the path to the SAGA executable, and you will not see those folders. Instead, you must make sure that SAGA is properly installed and its folder is added to the `PATH` environment variable. Just open a console and type `saga_cmd` to check that the system can find where the SAGA binaries are located.

17.14.4 About SAGA grid system limitations

Most SAGA algorithms that require several input raster layers require them to have the same grid system. That is, they must cover the same geographic area and have the same cell size, so their corresponding grids match. When calling SAGA algorithms from QGIS, you can use any layer, regardless of its cell size and extent. When multiple raster layers are used as input for a SAGA algorithm, QGIS resamples them to a common grid system and then passes them to SAGA (unless the SAGA algorithm can operate with layers from different grid systems).

The definition of that common grid system is controlled by the user, and you will find several parameters in the SAGA group of the settings window to do so. There are two ways of setting the target grid system:

- Setting it manually. You define the extent by setting the values of the following parameters:
 - *Resampling min X*
 - *Resampling max X*
 - *Resampling min Y*
 - *Resampling max Y*
 - *Resampling cellsize*

Notice that QGIS will resample input layers to that extent, even if they do not overlap with it.

- Setting it automatically from input layers. To select this option, just check the *Use min covering grid system for resampling* option. All the other settings will be ignored and the minimum extent that covers all the input layers will be used. The cell size of the target layer is the maximum of all cell sizes of the input layers.

For algorithms that do not use multiple raster layers, or for those that do not need a unique input grid system, no resampling is performed before calling SAGA, and those parameters are not used.

17.14.5 Limitations for multi-band layers

Unlike QGIS, SAGA has no support for multi-band layers. If you want to use a multiband layer (such as an RGB or multispectral image), you first have to split it into single-banded images. To do so, you can use the ‘SAGA/Grid - Tools/Split RGB image’ algorithm (which creates three images from an RGB image) or the ‘SAGA/Grid - Tools/Extract band’ algorithm (to extract a single band).

17.14.6 Limitations in cell size

SAGA assumes that raster layers have the same cell size in the X and Y axis. If you are working with a layer with different values for horizontal and vertical cell size, you might get unexpected results. In this case, a warning will be added to the processing log, indicating that an input layer might not be suitable to be processed by SAGA.

17.14.7 Logging

When QGIS calls SAGA, it does so using its command-line interface, thus passing a set of commands to perform all the required operations. SAGA shows its progress by writing information to the console, which includes the percentage of processing already done, along with additional content. This output is filtered and used to update the progress bar while the algorithm is running.

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 in detail 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 an external application and call it through the command-line have similar options, so you will find them as well in other places in the processing settings list.

R. Creating R scripts

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 a few examples). Instead, you should write your scripts and call R commands, much like you would do from R, and in a very similar manner to what we saw in the section dedicated to processing scripts. This section shows you the syntax to use to call those R commands from QGIS and how to use QGIS objects (layers, tables) in them.

The first thing you have to do, as we saw in the case of SAGA, is to tell QGIS where your R binaries are located. You can do this using the *R folder* entry in the processing configuration dialog. Once you have set that parameter, you can start creating and executing your own R scripts.

Once again, this is different in Linux, and you just have to make sure that the R folder is included in the PATH environment variable. If you can start R just typing R in a console, then you are ready to go.

To add a new 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 tells the processing framework how to perform that operation and the corresponding R commands to do so.

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 set this folder in the *R settings group* (available from the processing settings dialog), just like you do with the folder for regular processing scripts.

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, especially, `sp`, is mandatory.

```
##polyg=vector
##numpoints=number 10
##output=output vector
##sp=group
pts=spsample(polyg,numpoints,type="random")
output=SpatialPointsDataFrame(pts, as.data.frame(pts))
```

The first lines, which start with a double Python comment sign (`##`), tell QGIS the inputs of the algorithm described in the file and the outputs that it will generate. They work with exactly the same syntax as the SEXTANTE scripts that we have already seen, so they will not be described here again.

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 later be used as input for R commands.

In the above example, we are declaring an input of type `vector` named `polyg`. When executing the algorithm, QGIS will open in R the layer selected by the user and store it in a variable also named `polyg`. So, the name of a parameter is also the name of the variable that we can use in R for accessing the value of that parameter (thus, you should avoid using reserved R words as parameter names).

Spatial elements such as vector and raster layers are read using the `readOGR()` and `brick()` commands (you do not have to worry about adding those commands to your description file – QGIS will do it), and they are stored as `Spatial*DataFrame` objects. Table fields are stored as strings containing the name of the selected field.

Tables are opened using the `read.csv()` command. If a table entered by the user is not in CSV format, it will be converted prior to importing it into R.

Additionally, raster files can be read using the `readGDAL()` command instead of `brick()` by using the `##userreadgdal`.

If you are an advanced user and do not want QGIS to create the object representing the layer, you can use the `##passfilename` tag to indicate that you prefer a string with the filename instead. 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, we can now understand the first line of our first example script (the first line not starting with a Python comment).

```
pts=spsample(polyg,numpoints,type="random")
```

The variable `polyg` already contains a `SpatialPolygonsDataFrame` object, so it can be used to call the `spsample` method, just like the `numpoints` one, which indicates the number of points to add to the created sample grid.

Since we have declared an output of type `vector` named `out`, we have to create a variable named `out` and store a `Spatial*DataFrame` object in it (in this case, a `SpatialPointsDataFrame`). You can use any name for your intermediate variables. Just make sure that the variable storing your final result has the same name that you used to declare it, and that it contains a suitable value.

In this case, the result obtained from the `spsample` method has to be converted explicitly into a `SpatialPointsDataFrame` object, since it is itself an object of class `ppp`, which is not a suitable class to 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. In 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 `#passfilename` option, outputs are generated using the `raster` package (with `writeRaster()`), even though it is not used for the inputs.

If your algorithm does not generate any layer, but rather 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') sign. The output of all other lines will not be 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]])
```

The output of the last line is printed, but the output of the first is not (and neither are the outputs from other command lines added automatically by QGIS).

If your algorithm creates any kind of graphics (using the `plot()` method), add the following line:

```
##showplots
```

This will cause QGIS to redirect all R graphical outputs to a temporary file, which will be opened once R execution has finished.

Both graphics and console results will be shown in the processing results manager.

For more information, please check the script files provided with SEXTANTE. Most of them are rather simple and will greatly help you understand how to create your own scripts.

Catatan: `rgdal` and `maptools` libraries are loaded by default, so you do not have to add the corresponding `library()` commands (you just have to make sure that those two packages are installed in your R distribution). However, other additional libraries that you might need have to be explicitly loaded. Just add the necessary commands at the beginning of your script. You also have to make sure that the corresponding packages are installed in the R distribution used by QGIS. The processing framework will not take care of any package installation. If you run a script that requires a package that is not installed, the execution will fail, and SEXTANTE will try to detect which packages are missing. You must install those missing libraries manually before you can run the algorithm.

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. Additionally, a shell interpreter (usually `msys.exe`, which can be found in most GRASS for Windows distributions) has to be defined and its path set up as well.

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 in 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 that setting and point to the folder where the other version is installed. GRASS 6.4 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 console.

GRASS algorithms use a region for calculations. This region can be defined manually using values similar to the ones found in the SAGA configuration, or automatically, taking the minimum extent that covers all the input layers used to execute the algorithm each time. If the latter approach is the behaviour you prefer, just check the *Use min covering region* option in the GRASS configuration parameters.

The last parameter that has to be configured is related to the mapset. A mapset is needed to run GRASS, and the processing framework creates a temporary one for each execution. You have to specify if the data you are working with uses geographical (lat/lon) coordinates or projected ones.

GDAL

No additional configuration is needed to run GDAL algorithms. Since they are already incorporated into QGIS, the algorithms can infer their configuration from it.

Orfeo Toolbox

Orfeo Toolbox (OTB) algorithms can be run from QGIS if you have OTB installed in your system and you have configured QGIS properly, so it can find all necessary files (command-line tools and libraries).

As in the case of SAGA, OTB binaries are included in the stand-alone installer for Windows, but they are not included if you are running Linux, so you have to download and install the software yourself. Please check the OTB website for more information.

Once OTB is installed, start QGIS, open the processing configuration dialog and configure the OTB algorithm provider. In the *Orfeo Toolbox (image analysis)* block, you will find all settings related to OTB. First, ensure that algorithms are enabled.

Then, configure the path to the folder where OTB command-line tools and libraries are installed:

- 🐧 Usually *OTB applications folder* points to `/usr/lib/otb/applications` and *OTB command line tools folder* is `/usr/bin`.
- 🇺🇸 If you use the OSGeo4W installer, then install `otb-bin` package and enter `C:\OSGeo4W\apps\orfeotoolbox\applications` as *OTB applications folder* and `C:\OSGeo4W\bin` as *OTB command line tools folder*. These values should be configured by default, but if you have a different OTB installation, configure them to the corresponding values in your system.

TauDEM

To use this provider, you need to install TauDEM command line tools.

17.14.8 Windows

Please visit the [TauDEM homepage](#) for installation instructions and precompiled binaries for 32-bit and 64-bit systems. **IMPORTANT:** You need TauDEM 5.0.6 executables. Version 5.2 is currently not supported.

17.14.9 Linux

There are no packages for most Linux distributions, so you should compile TauDEM by yourself. As TauDEM uses MPICH2, first install it using your favorite package manager. Alternatively, TauDEM works fine with Open MPI, so you can use it instead of MPICH2.

Download TauDEM 5.0.6 [source code](#) and extract the files in some folder.

Open the `linearpart.h` file, and after line

```
#include "mpi.h"
```

add a new line with

```
#include <stdint.h>
```

so you'll get

```
#include "mpi.h"
#include <stdint.h>
```

Save the changes and close the file. Now open `tiffIO.h`, find line `#include "stdint.h"` and replace quotes (" ") with `<>`, so you'll get

```
#include <stdint.h>
```

Save the changes and close the file. Create a build directory and `cd` into it

```
mkdir build
cd build
```

Configure your build with the command

```
CXX=mpicxx cmake -DCMAKE_INSTALL_PREFIX=/usr/local ..
```

and then compile

```
make
```

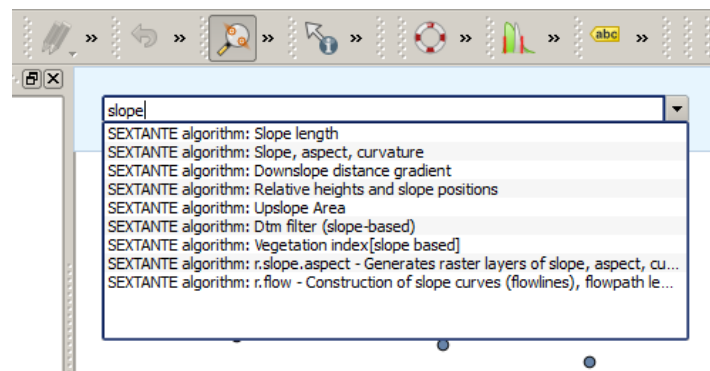
Finally, to install TauDEM into /usr/local/bin, run

```
sudo make install
```

17.15 The QGIS Commander

Processing includes a practical tool that allows you to run algorithms without having to use the toolbox, but just by typing the name of the algorithm you want to run.

This tool is known as the *QGIS commander*, and it is just a simple text box with autocompletion where you type the command you want to run.



Gambar 17.29: The QGIS Commander

The Commander is started from the *Analysis* menu or, more practically, by pressing `Shift + Ctrl + M` (you can change that default keyboard shortcut in the QGIS configuration if you prefer a different one). Apart from executing Processing algorithms, the Commander gives you access to most of the functionality in QGIS, which means that it gives you a practical and efficient way of running QGIS tasks and allows you to control QGIS with reduced usage of buttons and menus.

Moreover, the Commander is configurable, so you can add your custom commands and have them just a few keystrokes away, making it a powerful tool to help you become more productive in your daily work with QGIS.

17.15.1 Available commands

The commands available in the Commander fall in the following categories:

- Processing algorithms. These are shown as `Processing algorithm: <name of the algorithm>`.
- Menu items. These are shown as `Menu item: <menu entry text>`. All menus items available from the QGIS interface are available, even if they are included in a submenu.
- Python functions. You can create short Python functions that will be then included in the list of available commands. They are shown as `Function: <function name>`.

To run any of the above, just start typing and then select the corresponding element from the list of available commands that appears after filtering the whole list of commands with the text you have entered.

In the case of calling a Python function, you can select the entry in the list, which is prefixed by `Function:` (for instance, `Function: removeall`), or just directly type the function name (`removeall` in the previous example). There is no need to add brackets after the function name.

17.15.2 Creating custom functions

Custom functions are added by entering the corresponding Python code in the `commands.py` file that is found in the `.qgis/sexante/commander` directory in your user folder. It is just a simple Python file where you can add the functions that you need.

The file is created with a few example functions the first time you open the Commander. If you haven't launched the Commander yet, you can create the file yourself. To edit the `commands` file, use your favorite text editor. You can also use a built-in editor by calling the `edit` command from the Commander. It will open the editor with the `commands` file, and you can edit it directly and then save your changes.

For instance, you can add the following function, which removes all layers:

```
from qgis.gui import *

def removeall():
    mapreg = QgsMapLayerRegistry.instance()
    mapreg.removeAllMapLayers()
```

Once you have added the function, it will be available in the Commander, and you can invoke it by typing `removeall`. There is no need to do anything apart from writing the function itself.

Functions can receive parameters. Add `*args` to your function definition to receive arguments. When calling the function from the Commander, parameters have to be passed separated by spaces.

Here is an example of a function that loads a layer and takes a parameter with the filename of the layer to load.

```
import processing

def load(*args):
    processing.load(args[0])
```

If you want to load the layer in `/home/myuser/points.shp`, type `load /home/myuser/points.shp` in the Commander text box.

Processing providers and algorithms

18.1 GDAL algorithm provider

GDAL (Geospatial Data Abstraction Library) is a translator library for raster and vector geospatial data formats.

18.1.1 GDAL analysis

Aspect

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Compute edges [boolean] <put parameter description here>

Default: *False*

Use Zevenbergen&Thorne formula (instead of the Horn's one) [boolean] <put parameter description here>

Default: *False*

Return trigonometric angle (instead of azimuth) [boolean] <put parameter description here>

Default: *False*

Return 0 for flat (instead of -9999) [boolean] <put parameter description here>

Default: *False*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:aspect', input, band, compute_edges, zevenbergen, trig_angle, zero_flat)
```

See also

Color relief

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Compute edges [boolean] <put parameter description here>

Default: *False*

Color configuration file [file] <put parameter description here>

Matching mode [selection] <put parameter description here>

Options:

- 0 — “0,0,0,0” RGBA
- 1 — Exact color
- 2 — Nearest color

Default: *0*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:colorrelief', input, band, compute_edges, color_table, match_mode, outp
```

See also

Fill nodata

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Search distance [number] <put parameter description here>

Default: *100*

Smooth iterations [number] <put parameter description here>

Default: *0*

Band to operate on [number] <put parameter description here>

Default: *1*

Validity mask [raster] Optional.

<put parameter description here>

Do not use default validity mask [boolean] <put parameter description here>

Default: *False*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:fillnodata', input, distance, iterations, band, mask, no_default_mask,
```

See also

Grid (Moving average)

Description

<put algorithm description here>

Parameters

Input layer [vector: point] <put parameter description here>

Z field [tablefield: numeric] Optional.

<put parameter description here>

Radius 1 [number] <put parameter description here>

Default: *0.0*

Radius 2 [number] <put parameter description here>

Default: *0.0*

Min points [number] <put parameter description here>

Default: *0.0*

Angle [number] <put parameter description here>

Default: *0.0*

Nodata [number] <put parameter description here>

Default: 0.0

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:gridaverage', input, z_field, radius_1, radius_2, min_points, angle, n
```

See also

Grid (Data metrics)

Description

<put algorithm description here>

Parameters

Input layer [vector: point] <put parameter description here>

Z field [tablefield: numeric] Optional.

<put parameter description here>

Metrics [selection] <put parameter description here>

Options:

- 0 — Minimum
- 1 — Maximum
- 2 — Range

- 3 — Count
- 4 — Average distance
- 5 — Average distance between points

Default: 0

Radius 1 [number] <put parameter description here>

Default: 0.0

Radius 2 [number] <put parameter description here>

Default: 0.0

Min points [number] <put parameter description here>

Default: 0.0

Angle [number] <put parameter description here>

Default: 0.0

Nodata [number] <put parameter description here>

Default: 0.0

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:griddatametrics', input, z_field, metric, radius_1, radius_2, min_point
```

See also

Grid (Inverse distance to a power)

Description

<put algorithm description here>

Parameters

Input layer [**vector: point**] <put parameter description here>

Z field [**tablefield: numeric**] Optional.

<put parameter description here>

Power [**number**] <put parameter description here>

Default: 2.0

Smoothing [**number**] <put parameter description here>

Default: 0.0

Radius 1 [**number**] <put parameter description here>

Default: 0.0

Radius 2 [**number**] <put parameter description here>

Default: 0.0

Max points [**number**] <put parameter description here>

Default: 0.0

Min points [**number**] <put parameter description here>

Default: 0.0

Angle [**number**] <put parameter description here>

Default: 0.0

Nodata [**number**] <put parameter description here>

Default: 0.0

Output raster type [**selection**] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32

- 10 — CFloat64

Default: 5

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:gridinvdist', input, z_field, power, smothing, radius_1, radius_2, max.
```

See also

Grid (Nearest neighbor)

Description

<put algorithm description here>

Parameters

Input layer [vector: point] <put parameter description here>

Z field [tablefield: numeric] Optional.

<put parameter description here>

Radius 1 [number] <put parameter description here>

Default: 0.0

Radius 2 [number] <put parameter description here>

Default: 0.0

Angle [number] <put parameter description here>

Default: 0.0

Nodata [number] <put parameter description here>

Default: 0.0

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32

- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:gridnearestneighbor', input, z_field, radius_1, radius_2, angle, nodata)
```

See also

Hillshade

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: 1

Compute edges [boolean] <put parameter description here>

Default: *False*

Use Zevenbergen&Thorne formula (instead of the Horn's one) [boolean] <put parameter description here>

Default: *False*

Z factor (vertical exaggeration) [number] <put parameter description here>

Default: 1.0

Scale (ratio of vert. units to horiz.) [number] <put parameter description here>

Default: 1.0

Azimuth of the light [number] <put parameter description here>

Default: 315.0

Altitude of the light [number] <put parameter description here>

Default: 45.0

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:hillshade', input, band, compute_edges, zevenbergen, z_factor, scale, a
```

See also

Near black

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

How far from black (white) [number] <put parameter description here>

Default: *15*

Search for nearly white pixels instead of nearly black [boolean] <put parameter description here>

Default: *False*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:nearblack', input, near, white, output)
```

See also

Proximity (raster distance)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Values [string] <put parameter description here>

Default: *(not set)*

Dist units [selection] <put parameter description here>

Options:

- 0 — GEO
- 1 — PIXEL

Default: 0

Max dist (negative value to ignore) [number] <put parameter description here>

Default: -1

No data (negative value to ignore) [number] <put parameter description here>

Default: -1

Fixed buf val (negative value to ignore) [number] <put parameter description here>

Default: -1

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:proximity', input, values, units, max_dist, nodata, buf_val, rtype, out)
```

See also

Roughness

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: 1

Compute edges [boolean] <put parameter description here>

Default: *False*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:roughness', input, band, compute_edges, output)
```

See also

Sieve

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Threshold [number] <put parameter description here>

Default: 2

Pixel connection [selection] <put parameter description here>

Options:

- 0 — 4
- 1 — 8

Default: 0

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:sieve', input, threshold, connections, output)
```

See also

Slope

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Compute edges [boolean] <put parameter description here>

Default: *False*

Use Zevenbergen&Thorne formula (instead of the Horn's one) [boolean] <put parameter description here>

Default: *False*

Slope expressed as percent (instead of degrees) [boolean] <put parameter description here>

Default: *False*

Scale (ratio of vert. units to horiz.) [number] <put parameter description here>

Default: *1.0*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:slope', input, band, compute_edges, zevenbergen, as_percent, scale, out)
```

See also

TPI (Topographic Position Index)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Compute edges [boolean] <put parameter description here>

Default: *False*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:tpitopographicpositionindex', input, band, compute_edges, output)
```

See also

TRI (Terrain Ruggedness Index)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Compute edges [boolean] <put parameter description here>

Default: *False*

Outputs

Output file [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:triterrainruggednessindex', input, band, compute_edges, output)
```

See also

.

18.1.2 GDAL conversion

gdal2xyz

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band number [number] <put parameter description here>

Default: *1*

Outputs

Output file [table] <put output description here>

Console usage

```
processing.runalg('gdalogr:gdal2xyz', input, band, output)
```

See also

PCT to RGB

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Band to convert [selection] <put parameter description here>

Options:

- 0 — 1
- 1 — 2
- 2 — 3
- 3 — 4
- 4 — 5
- 5 — 6
- 6 — 7
- 7 — 8
- 8 — 9
- 9 — 10
- 10 — 11
- 11 — 12
- 12 — 13
- 13 — 14
- 14 — 15
- 15 — 16
- 16 — 17
- 17 — 18
- 18 — 19
- 19 — 20
- 20 — 21

- 21 — 22
- 22 — 23
- 23 — 24
- 24 — 25

Default: 0

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:pcttorgb', input, nband, output)
```

See also

Polygonize (raster to vector)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Output field name [string] <put parameter description here>

Default: *DN*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:polygonize', input, field, output)
```

See also

Rasterize (vector to raster)

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Attribute field [tablefield: any] <put parameter description here>

Write values inside an existing raster layer (*) [boolean] <put parameter description here>

Default: *False*

Set output raster size (ignored if above option is checked) [selection] <put parameter description here>

Options:

- 0 — Output size in pixels
- 1 — Output resolution in map units per pixel

Default: *1*

Horizontal [number] <put parameter description here>

Default: *100.0*

Vertical [number] <put parameter description here>

Default: *100.0*

Raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: *0*

Outputs

Output layer: mandatory to choose an existing raster layer if the (*) option is selected
<put output description here>

Console usage

```
processing.runalg('gdalogr:rasterize', input, field, writeover, dimensions, width, height, rtype,
```

See also

RGB to PCT

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Number of colors [number] <put parameter description here>

Default: 2

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:rgbttopct', input, ncolors, output)
```

See also

Translate (convert format)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Set the size of the output file (In pixels or %) [number] <put parameter description here>

Default: 100

Output size is a percentage of input size [boolean] <put parameter description here>

Default: *True*

Nodata value, leave as none to take the nodata value from input [string] <put parameter description here>

Default: *none*

Expand [selection] <put parameter description here>

Options:

- 0 — none
- 1 — gray
- 2 — rgb

- 3 — rgba

Default: 0

Output projection for output file [leave blank to use input projection] [crs]
<put parameter description here>

Default: *None*

Subset based on georeferenced coordinates [extent] <put parameter description here>

Default: *0,1,0,1*

Copy all subdatasets of this file to individual output files [boolean] <put parameter description here>

Default: *False*

Additional creation parameters [string] Optional.

<put parameter description here>

Default: *(not set)*

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:translate', input, outsize, outsize_perc, no_data, expand, srs, projwin)
```

See also

.

18.1.3 GDAL extraction

Clip raster by extent

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Nodata value, leave as none to take the nodata value from input [string] <put parameter description here>

Default: *none*

Clipping extent [extent] <put parameter description here>

Default: *0,1,0,1*

Additional creation parameters [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:cliprasterbyextent', input, no_data, projwin, extra, output)
```

See also

Clip raster by mask layer

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Mask layer [vector: polygon] <put parameter description here>

Nodata value, leave as none to take the nodata value from input [string] <put parameter description here>

Default: *none*

Create and output alpha band [boolean] <put parameter description here>

Default: *False*

Keep resolution of output raster [boolean] <put parameter description here>

Default: *False*

Additional creation parameters [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:cliprasterbymasklayer', input, mask, no_data, alpha_band, keep_resolut
```

See also

Contour

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Interval between contour lines [number] <put parameter description here>

Default: *10.0*

Attribute name (if not set, no elevation attribute is attached) [string]

Optional.

<put parameter description here>

Default: *ELEV*

Additional creation parameters [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output file for contour lines (vector) [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:contour', input_raster, interval, field_name, extra, output_vector)
```

See also

.

18.1.4 GDAL miscellaneous

Build Virtual Raster

Description

<put algorithm description here>

Parameters

Input layers [**multipleinput: rasters**] <put parameter description here>

Resolution [**selection**] <put parameter description here>

Options:

- 0 — average
- 1 — highest
- 2 — lowest

Default: *0*

Layer stack [**boolean**] <put parameter description here>

Default: *True*

Allow projection difference [**boolean**] <put parameter description here>

Default: *False*

Outputs

Output layer [**raster**] <put output description here>

Console usage

```
processing.runalg('gdalogr:buildvirtualraster', input, resolution, separate, proj_difference, outp
```

See also

Merge

Description

<put algorithm description here>

Parameters

Input layers [**multipleinput: rasters**] <put parameter description here>

Grab pseudocolor table from first layer [**boolean**] <put parameter description here>

Default: *False*

Layer stack [**boolean**] <put parameter description here>

Default: *False*

Output raster type [**selection**] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: 5

Outputs

Output layer [**raster**] <put output description here>

Console usage

```
processing.runalg('gdalogr:merge', input, pct, separate, rtype, output)
```

See also

Build overviews (pyramids)

Description

<put algorithm description here>

Parameters

Input layer [**raster**] <put parameter description here>

Overview levels [**string**] <put parameter description here>

Default: 2 4 8 16

Remove all existing overviews [boolean] <put parameter description here>

Default: *False*

Resampling method [selection] <put parameter description here>

Options:

- 0 — nearest
- 1 — average
- 2 — gauss
- 3 — cubic
- 4 — average_mp
- 5 — average_magphase
- 6 — mode

Default: *0*

Overview format [selection] <put parameter description here>

Options:

- 0 — Internal (if possible)
- 1 — External (GTiff .ovr)
- 2 — External (ERDAS Imagine .aux)

Default: *0*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:overviews', input, levels, clean, resampling_method, format)
```

See also

Information

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Suppress GCP info [boolean] <put parameter description here>

Default: *False*

Suppress metadata info [boolean] <put parameter description here>

Default: *False*

Outputs

Layer information [html] <put output description here>

Console usage

```
processing.runalg('gdalorg:rasterinfo', input, nogcp, nometadata, output)
```

See also

.

18.1.5 GDAL projections

Extract projection

Description

<put algorithm description here>

Parameters

Input file [raster] <put parameter description here>

Create also .prj file [boolean] <put parameter description here>

Default: *False*

Outputs

Console usage

```
processing.runalg('gdalorg:extractprojection', input, prj_file)
```

See also

Warp (reproject)

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Source SRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Destination SRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Output file resolution in target georeferenced units (leave 0 for no change) [number]
 <put parameter description here>

Default: *0.0*

Resampling method [selection] <put parameter description here>

Options:

- 0 — near
- 1 — bilinear
- 2 — cubic
- 3 — cubicspline
- 4 — lanczos

Default: *0*

Additional creation parameters [string] Optional.

<put parameter description here>

Default: *(not set)*

Output raster type [selection] <put parameter description here>

Options:

- 0 — Byte
- 1 — Int16
- 2 — UInt16
- 3 — UInt32
- 4 — Int32
- 5 — Float32
- 6 — Float64
- 7 — CInt16
- 8 — CInt32
- 9 — CFloat32
- 10 — CFloat64

Default: *5*

Outputs

Output layer [raster] <put output description here>

Console usage

```
processing.runalg('gdalogr:warpproject', input, source_srs, dest_srs, tr, method, extra, rtype,
```

See also

.

18.1.6 OGR conversion

Convert format

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Destination Format [selection] <put parameter description here>

Options:

- 0 — ESRI Shapefile
- 1 — GeoJSON
- 2 — GeoRSS
- 3 — SQLite
- 4 — GMT
- 5 — MapInfo File
- 6 — INTERLIS 1
- 7 — INTERLIS 2
- 8 — GML
- 9 — Geoconcept
- 10 — DXF
- 11 — DGN
- 12 — CSV
- 13 — BNA
- 14 — S57
- 15 — KML
- 16 — GPX
- 17 — PGDump
- 18 — GPSTrackMaker
- 19 — ODS
- 20 — XLSX
- 21 — PDF

Default: 0

Creation Options [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:convertformat', input_layer, format, options, output_layer)
```

See also

.

18.1.7 OGR geoprocessing

Clip vectors by extent

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Clip extent [extent] <put parameter description here>

Default: *0,1,0,1*

Additional creation Options [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:clipvectorsbyextent', input_layer, clip_extent, options, output_layer)
```

See also

Clip vectors by polygon

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Clip layer [vector: polygon] <put parameter description here>

Additional creation Options [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:clipvectorsbypolygon', input_layer, clip_layer, options, output_layer)
```

See also

.

18.1.8 OGR miscellaneous

Execute SQL

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

SQL [string] <put parameter description here>

Default: *(not set)*

Outputs

SQL result [vector] <put output description here>

Console usage

```
processing.runalg('gdalogr:executesql', input, sql, output)
```

See also**Import Vector into PostGIS database (available connections)****Description**

<put algorithm description here>

Parameters

Database (connection name) [selection] <put parameter description here>

Options:

- 0 — local

Default: 0

Input layer [vector: any] <put parameter description here>

Output geometry type [selection] <put parameter description here>

Options:

- 0 —
- 1 — NONE
- 2 — GEOMETRY
- 3 — POINT
- 4 — LINESTRING
- 5 — POLYGON
- 6 — GEOMETRYCOLLECTION
- 7 — MULTIPOINT
- 8 — MULTIPOLYGON
- 9 — MULTILINESTRING

Default: 5

Input CRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Output CRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Schema name [string] Optional.

<put parameter description here>

Default: *public*

Table name, leave blank to use input name [string] Optional.

<put parameter description here>

Default: *(not set)*

Primary Key [string] Optional.

<put parameter description here>

Default: *id*

Geometry column name [string] Optional.

<put parameter description here>

Default: *geom*

Vector dimensions [selection] <put parameter description here>

Options:

- 0 — 2
- 1 — 3

Default: *0*

Distance tolerance for simplification [string] Optional.

<put parameter description here>

Default: *(not set)*

Maximum distance between 2 nodes (densification) [string] Optional.

<put parameter description here>

Default: *(not set)*

Select features by extent (defined in input layer CRS) [extent] <put parameter description here>

Default: *0,1,0,1*

Clip the input layer using the above (rectangle) extent [boolean] <put parameter description here>

Default: *False*

Select features using a SQL "WHERE" statement (Ex: column="value") [string] Optional.

<put parameter description here>

Default: *(not set)*

Group "n" features per transaction (Default: 20000) [string] Optional.

<put parameter description here>

Default: *(not set)*

Overwrite existing table? [boolean] <put parameter description here>

Default: *True*

Append to existing table? [boolean] <put parameter description here>

Default: *False*

Append and add new fields to existing table? [boolean] <put parameter description here>

Default: *False*

Do not launder columns/table name/s? [boolean] <put parameter description here>

Default: *False*

Do not create Spatial Index? [boolean] <put parameter description here>

Default: *False*

Continue after a failure, skipping the failed feature [boolean] <put parameter description here>

Default: *False*

Additional creation options [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Console usage

```
processing.runalg('gdalogr:importvectorintopostgisdatabaseavailableconnections', database, input_
```

See also

Import Vector into PostGIS database (new connection)

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Output geometry type [selection] <put parameter description here>

Options:

- 0 —
- 1 — NONE
- 2 — GEOMETRY
- 3 — POINT
- 4 — LINESTRING
- 5 — POLYGON
- 6 — GEOMETRYCOLLECTION
- 7 — MULTIPOINT
- 8 — MULTIPOLYGON
- 9 — MULTILINESTRING

Default: 5

Input CRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Output CRS (EPSG Code) [crs] <put parameter description here>

Default: *EPSG:4326*

Host [string] <put parameter description here>

Default: *localhost*

Port [string] <put parameter description here>

Default: *5432*

Username [string] <put parameter description here>

Default: *(not set)*

Database Name [string] <put parameter description here>

Default: *(not set)*

Password [string] <put parameter description here>

Default: *(not set)*

Schema name [string] Optional.

<put parameter description here>

Default: *public*

Table name, leave blank to use input name [string] Optional.

<put parameter description here>

Default: *(not set)*

Primary Key [string] Optional.

<put parameter description here>

Default: *id*

Geometry column name [string] Optional.

<put parameter description here>

Default: *geom*

Vector dimensions [selection] <put parameter description here>

Options:

- 0 — 2
- 1 — 3

Default: *0*

Distance tolerance for simplification [string] Optional.

<put parameter description here>

Default: *(not set)*

Maximum distance between 2 nodes (densification) [string] Optional.

<put parameter description here>

Default: *(not set)*

Select features by extent (defined in input layer CRS) [extent] <put parameter description here>

Default: *0,1,0,1*

Clip the input layer using the above (rectangle) extent [boolean] <put parameter description here>

Default: *False*

Select features using a SQL "WHERE" statement (Ex: column="value") [string] Optional.

<put parameter description here>

Default: *(not set)*

Group "n" features per transaction (Default: 20000) [string] Optional.

<put parameter description here>

Default: *(not set)*

Overwrite existing table? [boolean] <put parameter description here>

Default: *True*

Append to existing table? [boolean] <put parameter description here>

Default: *False*

Append and add new fields to existing table? [boolean] <put parameter description here>

Default: *False*

Do not launder columns/table name/s? [boolean] <put parameter description here>

Default: *False*

Do not create Spatial Index? [boolean] <put parameter description here>

Default: *False*

Continue after a failure, skipping the failed feature [boolean] <put parameter description here>

Default: *False*

Additional creation options [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Console usage

```
processing.runalg('gdalogr:importvectorintopostgisdatabasewconnection', input_layer, gtype, s_s
```

See also

Information

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Outputs

Layer information [html] <put output description here>

Console usage

```
processing.runalg('gdalogr:information', input, output)
```

See also

.

18.2 LAStools

LAStools is a collection of highly efficient, multicore command line tools for LiDAR data processing.

18.2.1 las2las_filter

Description

<put algorithm description here>

Parameters

verbose [**boolean**] <put parameter description here>

Default: *False*

input LAS/LAZ file [**file**] Optional.

<put parameter description here>

filter (by return, classification, flags) [**selection**] <put parameter description here>

Options:

- 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

Default: 0

second filter (by return, classification, flags) [selection] <put parameter description here>

Options:

- 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

Default: 0

filter (by coordinate, intensity, GPS time, ...) [selection] <put parameter description here>

Options:

- 0 — —
- 1 — clip_x_above
- 2 — clip_x_below
- 3 — clip_y_above
- 4 — clip_y_below
- 5 — clip_z_above
- 6 — clip_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

Default: 0

value for filter (by coordinate, intensity, GPS time, ...) [string] <put parameter description here>

Default: *(not set)*

second filter (by coordinate, intensity, GPS time, ...) [selection] <put parameter description here>

Options:

- 0 — —
- 1 — clip_x_above
- 2 — clip_x_below
- 3 — clip_y_above
- 4 — clip_y_below
- 5 — clip_z_above
- 6 — clip_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

Default: 0

value for second filter (by coordinate, intensity, GPS time, ...) [string] <put parameter description here>

Default: *(not set)*

Outputs

output LAS/LAZ file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:las2lasfilter', verbose, input_laslaz, filter_return_class_flags1, ...)
```

See also

18.2.2 las2las_project

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

source projection [selection] <put parameter description here>

Options:

- 0 — —
- 1 — utm
- 2 — sp83
- 3 — sp27
- 4 — longlat
- 5 — latlong

Default: *0*

source utm zone [selection] <put parameter description here>

Options:

- 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)
- 62 — 2 (south)
- 63 — 3 (south)
- 64 — 4 (south)
- 65 — 5 (south)
- 66 — 6 (south)
- 67 — 7 (south)
- 68 — 8 (south)
- 69 — 9 (south)
- 70 — 10 (south)
- 71 — 11 (south)
- 72 — 12 (south)
- 73 — 13 (south)
- 74 — 14 (south)
- 75 — 15 (south)
- 76 — 16 (south)
- 77 — 17 (south)
- 78 — 18 (south)
- 79 — 19 (south)
- 80 — 20 (south)
- 81 — 21 (south)
- 82 — 22 (south)
- 83 — 23 (south)
- 84 — 24 (south)

- 85 — 25 (south)
- 86 — 26 (south)
- 87 — 27 (south)
- 88 — 28 (south)
- 89 — 29 (south)
- 90 — 30 (south)
- 91 — 31 (south)
- 92 — 32 (south)
- 93 — 33 (south)
- 94 — 34 (south)
- 95 — 35 (south)
- 96 — 36 (south)
- 97 — 37 (south)
- 98 — 38 (south)
- 99 — 39 (south)
- 100 — 40 (south)
- 101 — 41 (south)
- 102 — 42 (south)
- 103 — 43 (south)
- 104 — 44 (south)
- 105 — 45 (south)
- 106 — 46 (south)
- 107 — 47 (south)
- 108 — 48 (south)
- 109 — 49 (south)
- 110 — 50 (south)
- 111 — 51 (south)
- 112 — 52 (south)
- 113 — 53 (south)
- 114 — 54 (south)
- 115 — 55 (south)
- 116 — 56 (south)
- 117 — 57 (south)
- 118 — 58 (south)
- 119 — 59 (south)
- 120 — 60 (south)

Default: 0

source state plane code [selection] <put parameter description here>

Options:

- 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
- 62 — MN_C
- 63 — MN_N
- 64 — MN_S
- 65 — MO_C
- 66 — MO_E
- 67 — MO_W
- 68 — MS_E
- 69 — MS_W
- 70 — MT_C
- 71 — MT_N
- 72 — MT_S
- 73 — NC
- 74 — ND_N
- 75 — ND_S
- 76 — NE_N
- 77 — NE_S

- 78 — NH
- 79 — NJ
- 80 — NM_C
- 81 — NM_E
- 82 — NM_W
- 83 — NV_C
- 84 — NV_E
- 85 — NV_W
- 86 — NY_C
- 87 — NY_E
- 88 — NY_LI
- 89 — NY_W
- 90 — OH_N
- 91 — OH_S
- 92 — OK_N
- 93 — OK_S
- 94 — OR_N
- 95 — OR_S
- 96 — PA_N
- 97 — PA_S
- 98 — PR
- 99 — RI
- 100 — SC_N
- 101 — SC_S
- 102 — SD_N
- 103 — SD_S
- 104 — St.Croix
- 105 — TN
- 106 — TX_C
- 107 — TX_N
- 108 — TX_NC
- 109 — TX_S
- 110 — TX_SC
- 111 — UT_C
- 112 — UT_N
- 113 — UT_S
- 114 — VA_N
- 115 — VA_S
- 116 — VT

- 117 — WA_N
- 118 — WA_S
- 119 — WI_C
- 120 — WI_N
- 121 — WI_S
- 122 — WV_N
- 123 — WV_S
- 124 — WY_E
- 125 — WY_EC
- 126 — WY_W
- 127 — WY_WC

Default: 0

target projection [selection] <put parameter description here>

Options:

- 0 — —
- 1 — utm
- 2 — sp83
- 3 — sp27
- 4 — longlat
- 5 — latlong

Default: 0

target utm zone [selection] <put parameter description here>

Options:

- 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)
- 62 — 2 (south)
- 63 — 3 (south)
- 64 — 4 (south)
- 65 — 5 (south)
- 66 — 6 (south)
- 67 — 7 (south)
- 68 — 8 (south)
- 69 — 9 (south)
- 70 — 10 (south)
- 71 — 11 (south)
- 72 — 12 (south)
- 73 — 13 (south)
- 74 — 14 (south)
- 75 — 15 (south)
- 76 — 16 (south)
- 77 — 17 (south)
- 78 — 18 (south)
- 79 — 19 (south)
- 80 — 20 (south)
- 81 — 21 (south)
- 82 — 22 (south)
- 83 — 23 (south)
- 84 — 24 (south)
- 85 — 25 (south)
- 86 — 26 (south)
- 87 — 27 (south)
- 88 — 28 (south)
- 89 — 29 (south)
- 90 — 30 (south)
- 91 — 31 (south)
- 92 — 32 (south)
- 93 — 33 (south)

- 94 — 34 (south)
- 95 — 35 (south)
- 96 — 36 (south)
- 97 — 37 (south)
- 98 — 38 (south)
- 99 — 39 (south)
- 100 — 40 (south)
- 101 — 41 (south)
- 102 — 42 (south)
- 103 — 43 (south)
- 104 — 44 (south)
- 105 — 45 (south)
- 106 — 46 (south)
- 107 — 47 (south)
- 108 — 48 (south)
- 109 — 49 (south)
- 110 — 50 (south)
- 111 — 51 (south)
- 112 — 52 (south)
- 113 — 53 (south)
- 114 — 54 (south)
- 115 — 55 (south)
- 116 — 56 (south)
- 117 — 57 (south)
- 118 — 58 (south)
- 119 — 59 (south)
- 120 — 60 (south)

Default: 0

target state plane code [selection] <put parameter description here>

Options:

- 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
- 62 — MN_C
- 63 — MN_N
- 64 — MN_S
- 65 — MO_C
- 66 — MO_E
- 67 — MO_W
- 68 — MS_E
- 69 — MS_W
- 70 — MT_C
- 71 — MT_N
- 72 — MT_S
- 73 — NC
- 74 — ND_N
- 75 — ND_S
- 76 — NE_N
- 77 — NE_S
- 78 — NH
- 79 — NJ
- 80 — NM_C
- 81 — NM_E
- 82 — NM_W
- 83 — NV_C
- 84 — NV_E
- 85 — NV_W
- 86 — NY_C

- 87 — NY_E
- 88 — NY_LI
- 89 — NY_W
- 90 — OH_N
- 91 — OH_S
- 92 — OK_N
- 93 — OK_S
- 94 — OR_N
- 95 — OR_S
- 96 — PA_N
- 97 — PA_S
- 98 — PR
- 99 — RI
- 100 — SC_N
- 101 — SC_S
- 102 — SD_N
- 103 — SD_S
- 104 — St.Croix
- 105 — TN
- 106 — TX_C
- 107 — TX_N
- 108 — TX_NC
- 109 — TX_S
- 110 — TX_SC
- 111 — UT_C
- 112 — UT_N
- 113 — UT_S
- 114 — VA_N
- 115 — VA_S
- 116 — VT
- 117 — WA_N
- 118 — WA_S
- 119 — WI_C
- 120 — WI_N
- 121 — WI_S
- 122 — WV_N
- 123 — WV_S
- 124 — WY_E
- 125 — WY_EC

- 126 — WY_W
- 127 — WY_WC

Default: 0

Outputs

output LAS/LAZ file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:las2lasproject', verbose, input_laslaz, source_projection, source_u
```

See also

18.2.3 las2las_transform

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

transform (coordinates) [selection] <put parameter description here>

Options:

- 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

Default: 0

value for transform (coordinates) [string] <put parameter description here>

Default: *(not set)*

second transform (coordinates) [selection] <put parameter description here>

Options:

- 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

Default: 0

value for second transform (coordinates) [string] <put parameter description here>

Default: *(not set)*

transform (intensities, scan angles, GPS times, ...) [selection] <put parameter description here>

Options:

- 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

Default: 0

value for transform (intensities, scan angles, GPS times, ...) [string] <put parameter description here>

Default: *(not set)*

second transform (intensities, scan angles, GPS times, ...) [selection] <put parameter description here>

Options:

- 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

Default: 0

value for second transform (intensities, scan angles, GPS times, ...) [string]
 <put parameter description here>

Default: *(not set)*

operations (first 7 need an argument) [selection] <put parameter description here>

Options:

- 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 — scale_rgb_up
- 12 — scale_rgb_down
- 13 — remove_all_vlrs
- 14 — remove_extra
- 15 — clip_to_bounding_box

Default: 0

argument for operation [string] <put parameter description here>

Default: *(not set)*

Outputs

output LAS/LAZ file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:las2lastransform', verbose, input_laslaz, transform_coordinate1, tr
```

See also

18.2.4 las2txt

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

parse_string [string] <put parameter description here>

Default: *xyz*

Outputs

Output ASCII file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:las2txt', verbose, input_laslaz, parse_string, output)
```

See also

18.2.5 lasindex

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

is mobile or terrestrial LiDAR (not airborne) [boolean] <put parameter description here>

Default: *False*

Outputs

Console usage

```
processing.runalg('lidartools:lasindex', verbose, input_laslaz, mobile_or_terrestrial)
```

See also

18.2.6 lasinfo

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

Outputs

Output ASCII file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:lasinfo', verbose, input_laslaz, output)
```

See also

18.2.7 lasmerge

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

files are flightlines [boolean] <put parameter description here>

Default: *True*

input LAS/LAZ file [file] Optional.

<put parameter description here>

2nd file [file] Optional.

<put parameter description here>

3rd file [file] Optional.

<put parameter description here>

4th file [file] Optional.

<put parameter description here>

5th file [file] Optional.

<put parameter description here>

6th file [file] Optional.

<put parameter description here>

7th file [file] Optional.

<put parameter description here>

Outputs

output LAS/LAZ file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:lasmerge', verbose, files_are_flightlines, input_laslaz, file2, file3)
```

See also

18.2.8 lasprecision

Description

<put algorithm description here>

Parameters

verbose [boolean] <put parameter description here>

Default: *False*

input LAS/LAZ file [file] Optional.

<put parameter description here>

Outputs

Output ASCII file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:lasprecision', verbose, input_laslaz, output)
```

See also

18.2.9 lasquery

Description

<put algorithm description here>

Parameters

verbose [**boolean**] <put parameter description here>

Default: *False*

area of interest [**extent**] <put parameter description here>

Default: *0,1,0,1*

Outputs

Console usage

```
processing.runalg('lidartools:lasquery', verbose, aoi)
```

See also

18.2.10 lasvalidate

Description

<put algorithm description here>

Parameters

verbose [**boolean**] <put parameter description here>

Default: *False*

input LAS/LAZ file [**file**] Optional.

<put parameter description here>

Outputs

Output XML file [**file**] <put output description here>

Console usage

```
processing.runalg('lidartools:lasvalidate', verbose, input_laslaz, output)
```

See also

18.2.11 laszip

Description

<put algorithm description here>

Parameters

verbose [**boolean**] <put parameter description here>

Default: *False*

input LAS/LAZ file [**file**] Optional.

<put parameter description here>

only report size [**boolean**] <put parameter description here>

Default: *False*

Outputs

output LAS/LAZ file [**file**] <put output description here>

Console usage

```
processing.runalg('lidartools:laszip', verbose, input_laslaz, report_size, output_laslaz)
```

See also

18.2.12 txt2las

Description

<put algorithm description here>

Parameters

verbose [**boolean**] <put parameter description here>

Default: *False*

Input ASCII file [**file**] Optional.

<put parameter description here>

parse lines as [**string**] <put parameter description here>

Default: *xyz*

skip the first n lines [**number**] <put parameter description here>

Default: *0*

resolution of x and y coordinate [**number**] <put parameter description here>

Default: *0.01*

resolution of z coordinate [number] <put parameter description here>

Default: *0.01*

Outputs

output LAS/LAZ file [file] <put output description here>

Console usage

```
processing.runalg('lidartools:txt2las', verbose, input, parse_string, skip, scale_factor_xy, scale_factor_z)
```

See also

.

18.3 Modeler Tools

18.3.1 Calculator

Description

<put algorithm description here>

Parameters

Formula [string] <put parameter description here>

Default: *(not set)*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

dummy [number] <put parameter description here>

Default: *0.0*

Outputs

Result [number] <put output description here>

Console usage

```
processing.runalg('modelertools:calculator', formula, number0, number1, number2, number3, number4,
```

See also

18.3.2 Raster layer bounds

Description

<put algorithm description here>

Parameters

Layer [raster] <put parameter description here>

Outputs

min X [number] <put output description here>

max X [number] <put output description here>

min Y [number] <put output description here>

max Y [number] <put output description here>

Extent [extent] <put output description here>

Console usage

```
processing.runalg('modelertools:rasterlayerbounds', layer)
```

See also

18.3.3 Vector layer bounds

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Outputs

min X [number] <put output description here>

max X [number] <put output description here>

min Y [number] <put output description here>

max Y [number] <put output description here>

Extent [extent] <put output description here>

Console usage

```
processing.runalg('modelertools:vectorlayerbounds', layer)
```

See also

.

18.4 OrfeoToolbox algorithm provider

Orfeo ToolBox (OTB) is an open source library of image processing algorithms. OTB is based on the medical image processing library ITK and offers particular functionalities for remote sensing image processing in general and for high spatial resolution images in particular. Targeted algorithms for high resolution optical images (Pleiades, SPOT, QuickBird, WorldView, Landsat, Ikonos), hyperspectral sensors (Hyperion) or SAR (TerraSarX, ERS, Palsar) are available.

Catatan: Please remember that Processing contains only the interface description, so you need to install OTB by yourself and configure Processing properly.

.

18.4.1 Calibration

Optical calibration

Description

<put algorithm description here>

Parameters

Input [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Calibration Level [selection] <put parameter description here>

Options:

- 0 — toa

Default: *0*

Convert to milli reflectance [boolean] <put parameter description here>

Default: *True*

Clamp of reflectivity values between [0, 100] [boolean] <put parameter description here>

Default: *True*

Relative Spectral Response File [file] Optional.

<put parameter description here>

Outputs

Output [raster] <put output description here>

Console usage

```
processing.runalg('otb:opticalcalibration', -in, -ram, -level, -milli, -clamp, -rsr, -out)
```

See also

.

18.4.2 Feature extrcation

BinaryMorphologicalOperation (closing)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — closing

Default: *0*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:binarymorphologicaloperationclosing', -in, -channel, -ram, -structype, -st
```

See also

BinaryMorphologicalOperation (dilate)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — dilate

Default: *0*

Foreground Value [number] <put parameter description here>

Default: *1*

Background Value [number] <put parameter description here>

Default: *0*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:binarymorphologicaloperationdilate', -in, -channel, -ram, -structype, -stru
```

See also

BinaryMorphologicalOperation (erode)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — erode

Default: *0*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:binarymorphologicaloperationerode', -in, -channel, -ram, -structype, -stru
```

See also

BinaryMorphologicalOperation (opening)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — opening

Default: *0*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:binarymorphologicaloperationopening', -in, -channel, -ram, -structype, -st
```

See also

EdgeExtraction (gradient)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Edge feature [selection] <put parameter description here>

Options:

- 0 — gradient

Default: 0

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:edgeextractiongradient', -in, -channel, -ram, -filter, -out)
```

See also

EdgeExtraction (sobel)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: 1

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Edge feature [selection] <put parameter description here>

Options:

- 0 — sobel

Default: 0

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:edgeextractionsobel', -in, -channel, -ram, -filter, -out)
```

See also

EdgeExtraction (touzi)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Edge feature [selection] <put parameter description here>

Options:

- 0 — touzi

Default: *0*

The Radius [number] <put parameter description here>

Default: *1*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:edgeextractiontouzi', -in, -channel, -ram, -filter, -filter.touzi.xradius,
```

See also

GrayScaleMorphologicalOperation (closing)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — closing

Default: 0

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:grayscalemorphologicaloperationclosing', -in, -channel, -ram, -structype, ...)
```

See also

GrayScaleMorphologicalOperation (dilate)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: 1

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: 0

The Structuring Element Radius [number] <put parameter description here>

Default: 5

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — dilate

Default: 0

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:grayscalemorphologicaloperationdilate', -in, -channel, -ram, -structype, -s
```

See also

GrayScaleMorphologicalOperation (erode)

Description

<put algortithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: 1

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: 0

The Structuring Element Radius [number] <put parameter description here>

Default: 5

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — erode

Default: 0

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:grayscalemorphologicaloperationerode', -in, -channel, -ram, -structype, -s
```

See also

GrayScaleMorphologicalOperation (opening)

Description

<put algortithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Structuring Element Type [selection] <put parameter description here>

Options:

- 0 — ball

Default: *0*

The Structuring Element Radius [number] <put parameter description here>

Default: *5*

Morphological Operation [selection] <put parameter description here>

Options:

- 0 — opening

Default: *0*

Outputs

Feature Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:grayscalemorphologicaloperationopening', -in, -channel, -ram, -structype, .
```

See also

Haralick Texture Extraction

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Selected Channel [number] <put parameter description here>

Default: *1*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

X Radius [number] <put parameter description here>

Default: *2*

Y Radius [number] <put parameter description here>

Default: 2

X Offset [number] <put parameter description here>

Default: 1

Y Offset [number] <put parameter description here>

Default: 1

Image Minimum [number] <put parameter description here>

Default: 0

Image Maximum [number] <put parameter description here>

Default: 255

Histogram number of bin [number] <put parameter description here>

Default: 8

Texture Set Selection [selection] <put parameter description here>

Options:

- 0 — simple
- 1 — advanced
- 2 — higher

Default: 0

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:haralicktextureextraction', -in, -channel, -ram, -parameters.xrad, -parameter
```

See also

Line segment detection

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

No rescaling in [0, 255] [boolean] <put parameter description here>

Default: *True*

Outputs

Output **Detected lines** [**vector**] <put output description here>

Console usage

```
processing.runalg('otb:linesegmentdetection', -in, -norescale, -out)
```

See also

Local Statistic Extraction

Description

<put algorithm description here>

Parameters

Input Image [**raster**] <put parameter description here>

Selected Channel [**number**] <put parameter description here>

Default: *1*

Available RAM (Mb) [**number**] <put parameter description here>

Default: *128*

Neighborhood radius [**number**] <put parameter description here>

Default: *3*

Outputs

Feature Output Image [**raster**] <put output description here>

Console usage

```
processing.runalg('otb:localstatisticextraction', -in, -channel, -ram, -radius, -out)
```

See also

Multivariate alteration detector

Description

<put algorithm description here>

Parameters

Input Image 1 [raster] <put parameter description here>

Input Image 2 [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Outputs

Change Map [raster] <put output description here>

Console usage

```
processing.runalg('otb:multivariatealterationdetector', -in1, -in2, -ram, -out)
```

See also

Radiometric Indices

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Blue Channel [number] <put parameter description here>

Default: *1*

Green Channel [number] <put parameter description here>

Default: *1*

Red Channel [number] <put parameter description here>

Default: *1*

NIR Channel [number] <put parameter description here>

Default: *1*

Mir Channel [number] <put parameter description here>

Default: *1*

Available Radiometric Indices [selection] <put parameter description here>

Options:

- 0 — ndvi
- 1 — tndvi
- 2 — rvi

- 3 — savi
- 4 — tsavi
- 5 — msavi
- 6 — msavi2
- 7 — gemi
- 8 — ipvi
- 9 — ndwi
- 10 — ndwi2
- 11 — mndwi
- 12 — ndpi
- 13 — ndti
- 14 — ri
- 15 — ci
- 16 — bi
- 17 — bi2

Default: 0

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:radiometricindices', -in, -ram, -channels.blue, -channels.green, -channels
```

See also

.

18.4.3 Geometry

Image Envelope

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Sampling Rate [number] <put parameter description here>

Default: 0

Projection [string] Optional.

<put parameter description here>

Default: *None*

Outputs

Output Vector Data [vector] <put output description here>

Console usage

```
processing.runalg('otb:imageenvelope', -in, -sr, -proj, -out)
```

See also

OrthoRectification (epsg)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Output Cartographic Map Projection [selection] <put parameter description here>

Options:

- 0 — epsg

Default: *0*

EPSG Code [number] <put parameter description here>

Default: *4326*

Parameters estimation modes [selection] <put parameter description here>

Options:

- 0 — autosize
- 1 — autospacing

Default: *0*

Default pixel value [number] <put parameter description here>

Default: *0*

Default elevation [number] <put parameter description here>

Default: *0*

Interpolation [selection] <put parameter description here>

Options:

- 0 — bco
- 1 — nn
- 2 — linear

Default: 0

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Resampling grid spacing [number] <put parameter description here>

Default: 4

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:orthorectificationeps', -io.in, -map, -map.epsg.code, -outputs.mode, -outp
```

See also

OrthoRectification (fit-to-ortho)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Parameters estimation modes [selection] <put parameter description here>

Options:

- 0 — orthofit

Default: 0

Model ortho-image [raster] Optional.

<put parameter description here>

Default pixel value [number] <put parameter description here>

Default: 0

Default elevation [number] <put parameter description here>

Default: 0

Interpolation [selection] <put parameter description here>

Options:

- 0 — bco
- 1 — nn
- 2 — linear

Default: 0

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Resampling grid spacing [number] <put parameter description here>

Default: 4

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:orthorectificationfittoortho', -io.in, -outputs.mode, -outputs.ortho, -outp
```

See also

OrthoRectification (lambert-WGS84)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Output Cartographic Map Projection [selection] <put parameter description here>

Options:

- 0 — lambert2
- 1 — lambert93
- 2 — wgs

Default: 0

Parameters estimation modes [selection] <put parameter description here>

Options:

- 0 — autosize
- 1 — autospacing

Default: 0

Default pixel value [number] <put parameter description here>

Default: 0

Default elevation [number] <put parameter description here>

Default: 0

Interpolation [selection] <put parameter description here>

Options:

- 0 — bco
- 1 — nn
- 2 — linear

Default: 0

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Resampling grid spacing [number] <put parameter description here>

Default: 4

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:orthorectificationlambertwgs84', -io.in, -map, -outputs.mode, -outputs.def
```

See also

OrthoRectification (utm)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Output Cartographic Map Projection [selection] <put parameter description here>

Options:

- 0 — utm

Default: 0

Zone number [number] <put parameter description here>

Default: 31

Northern Hemisphere [boolean] <put parameter description here>

Default: *True*

Parameters estimation modes [selection] <put parameter description here>

Options:

- 0 — autosize
- 1 — autospacing

Default: 0

Default pixel value [number] <put parameter description here>

Default: 0

Default elevation [number] <put parameter description here>

Default: 0

Interpolation [selection] <put parameter description here>

Options:

- 0 — bco
- 1 — nn
- 2 — linear

Default: 0

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Resampling grid spacing [number] <put parameter description here>

Default: 4

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:orthorectificationutm', -io.in, -map, -map.utm.zone, -map.utm.northhem, -o
```

See also

Pansharpening (bayes)

Description

<put algorithm description here>

Parameters

Input PAN Image [raster] <put parameter description here>

Input XS Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — bayes

Default: 0

Weight [number] <put parameter description here>

Default: 0.9999

S coefficient [number] <put parameter description here>

Default: 1

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:pansharpeningbayes', -inp, -inxs, -method, -method.bayes.lambda, -method.b
```

See also

Pansharpening (lmvm)

Description

<put algorithm description here>

Parameters

Input PAN Image [raster] <put parameter description here>

Input XS Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — lmvm

Default: 0

X radius [number] <put parameter description here>

Default: 3

Y radius [number] <put parameter description here>

Default: 3

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:pansharpeninglmvm', -inp, -inxs, -method, -method.lmvm.radiusx, -method.lm
```

See also

Pansharpening (rcs)

Description

<put algorithm description here>

Parameters

Input PAN Image [raster] <put parameter description here>

Input XS Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — rcs

Default: 0

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:pansharpeningrcs', -inp, -inxs, -method, -ram, -out)
```

See also

RigidTransformResample (id)

Description

<put algorithm description here>

Parameters

Input image [raster] <put parameter description here>

Type of transformation [selection] <put parameter description here>

Options:

- 0 — id

Default: 0

X scaling [number] <put parameter description here>

Default: 1

Y scaling [number] <put parameter description here>

Default: 1

Interpolation [selection] <put parameter description here>

Options:

- 0 — nn
- 1 — linear
- 2 — bco

Default: 2

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:rigidtransformresampleid', -in, -transform.type, -transform.type.id.scalex
```

See also

RigidTransformResample (rotation)

Description

<put algorithm description here>

Parameters

Input image [raster] <put parameter description here>

Type of transformation [selection] <put parameter description here>

Options:

- 0 — rotation

Default: 0

Rotation angle [number] <put parameter description here>

Default: 0

X scaling [number] <put parameter description here>

Default: 1

Y scaling [number] <put parameter description here>

Default: 1

Interpolation [selection] <put parameter description here>

Options:

- 0 — nn
- 1 — linear
- 2 — bco

Default: 2

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:rigidtransformresamplerotation', -in, -transform.type, -transform.type.rot
```

See also

RigidTransformResample (translation)

Description

<put algorithm description here>

Parameters

Input image [raster] <put parameter description here>

Type of transformation [selection] <put parameter description here>

Options:

- 0 — translation

Default: 0

The X translation (in physical units) [number] <put parameter description here>

Default: 0

The Y translation (in physical units) [number] <put parameter description here>

Default: 0

X scaling [number] <put parameter description here>

Default: 1

Y scaling [number] <put parameter description here>

Default: 1

Interpolation [selection] <put parameter description here>

Options:

- 0 — nn
- 1 — linear
- 2 — bco

Default: 2

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:rigidtransformresampletranslation', -in, -transform.type, -transform.type.
```

See also

Superimpose sensor

Description

<put algorithm description here>

Parameters

Reference input [raster] <put parameter description here>

The image to reproject [raster] <put parameter description here>

Default elevation [number] <put parameter description here>

Default: 0

Spacing of the deformation field [number] <put parameter description here>

Default: 4

Interpolation [selection] <put parameter description here>

Options:

- 0 — bco
- 1 — nn
- 2 — linear

Default: 0

Radius for bicubic interpolation [number] <put parameter description here>

Default: 2

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Outputs

Output image [raster] <put output description here>

Console usage

```
processing.runalg('otb:superimposesensor', -inr, -inm, -elev.default, -lms, -interpolator, -inter
```

See also

.

18.4.4 Image filtering

DimensionalityReduction (ica)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — ica

Default: *0*

number of iterations [number] <put parameter description here>

Default: *20*

Give the increment weight of W in [0, 1] [number] <put parameter description here>

Default: *1*

Number of Components [number] <put parameter description here>

Default: *0*

Normalize [boolean] <put parameter description here>

Default: *True*

Outputs

Output Image [raster] <put output description here>

“ **Inverse Output Image** “ [raster] <put output description here>

Transformation matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:dimensionalityreductionica', -in, -method, -method.ica.iter, -method.ica.m
```

See also

DimensionalityReduction (maf)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — maf

Default: 0

Number of Components. [number] <put parameter description here>

Default: 0

Normalize. [boolean] <put parameter description here>

Default: *True*

Outputs

Output Image [raster] <put output description here>

Transformation matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:dimensionalityreductionmaf', -in, -method, -nbcomp, -normalize, -out, -out
```

See also

DimensionalityReduction (napca)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — napca

Default: 0

Set the x radius of the sliding window. [number] <put parameter description here>

Default: 1

Set the y radius of the sliding window. [number] <put parameter description here>

Default: 1

Number of Components. [number] <put parameter description here>

Default: 0

Normalize. [boolean] <put parameter description here>

Default: *True*

Outputs

Output Image [raster] <put output description here>

“**Inverse Output Image**“ [raster] <put output description here>

Transformation matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:dimensionalityreductionnapca', -in, -method, -method.napca.radiusx, -method
```

See also

DimensionalityReduction (pca)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Algorithm [selection] <put parameter description here>

Options:

- 0 — pca

Default: 0

Number of Components. [number] <put parameter description here>

Default: 0

Normalize. [boolean] <put parameter description here>

Default: *True*

Outputs

Output Image [raster] <put output description here>

“ **Inverse Output Image**“ [raster] <put output description here>

Transformation matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:dimensionalityreductionpca', -in, -method, -nbcomp, -normalize, -out, -out.
```

See also

Mean Shift filtering (can be used as Exact Large-Scale Mean-Shift segmentation, step 1)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Spatial radius [number] <put parameter description here>

Default: *5*

Range radius [number] <put parameter description here>

Default: *15*

Mode convergence threshold [number] <put parameter description here>

Default: *0.1*

Maximum number of iterations [number] <put parameter description here>

Default: *100*

Range radius coefficient [number] <put parameter description here>

Default: *0*

Mode search. [boolean] <put parameter description here>

Default: *True*

Outputs

Filtered output [raster] <put output description here>

Spatial image [raster] <put output description here>

Console usage

```
processing.runalg('otb:meanshiftfilteringcanbeusedasexactlargescalemeanshiftsegmentationstep1', -
```

See also

Smoothing (anidif)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Smoothing Type [selection] <put parameter description here>

Options:

- 0 — anidif

Default: *2*

Time Step [number] <put parameter description here>

Default: *0.125*

Nb Iterations [number] <put parameter description here>

Default: *10*

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:smoothinganidif', -in, -ram, -type, -type.anidif.timestep, -type.anidif.nb
```

See also

Smoothing (gaussian)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Smoothing Type [selection] <put parameter description here>

Options:

- 0 — gaussian

Default: 2

Radius [number] <put parameter description here>

Default: 2

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:smoothinggaussian', -in, -ram, -type, -type.gaussian.radius, -out)
```

See also

Smoothing (mean)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Smoothing Type [selection] <put parameter description here>

Options:

- 0 — mean

Default: 2

Radius [number] <put parameter description here>

Default: 2

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:smoothingmean', -in, -ram, -type, -type.mean.radius, -out)
```

See also

.

18.4.5 Image manipulation

ColorMapping (continuous)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Operation [selection] <put parameter description here>

Options:

- 0 — labeltocolour

Default: *0*

Color mapping method [selection] <put parameter description here>

Options:

- 0 — continuous

Default: *0*

Look-up tables [selection] <put parameter description here>

Options:

- 0 — red
- 1 — green
- 2 — blue
- 3 — grey
- 4 — hot
- 5 — cool
- 6 — spring
- 7 — summer
- 8 — autumn
- 9 — winter
- 10 — copper

- 11 — jet
- 12 — hsv
- 13 — overunder
- 14 — relief

Default: 0

Mapping range lower value [number] <put parameter description here>

Default: 0

Mapping range higher value [number] <put parameter description here>

Default: 255

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:colormappingcontinuous', -in, -ram, -op, -method, -method.continuous.lut, ...
```

See also

ColorMapping (custom)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Operation [selection] <put parameter description here>

Options:

- 0 — labeltcolor

Default: 0

Color mapping method [selection] <put parameter description here>

Options:

- 0 — custom

Default: 0

Look-up table file [file] <put parameter description here>

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:colormappingcustom', -in, -ram, -op, -method, -method.custom.lut, -out)
```

See also

ColorMapping (image)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Operation [selection] <put parameter description here>

Options:

- 0 — labeltocolor

Default: *0*

Color mapping method [selection] <put parameter description here>

Options:

- 0 — image

Default: *0*

Support Image [raster] <put parameter description here>

NoData value [number] <put parameter description here>

Default: *0*

lower quantile [number] <put parameter description here>

Default: *2*

upper quantile [number] <put parameter description here>

Default: *2*

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:colormappingimage', -in, -ram, -op, -method, -method.image.in, -method.ima
```

See also

ColorMapping (optimal)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Operation [selection] <put parameter description here>

Options:

- 0 — labeltocolor

Default: *0*

Color mapping method [selection] <put parameter description here>

Options:

- 0 — optimal

Default: *0*

Background label [number] <put parameter description here>

Default: *0*

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:colormappingoptimal', -in, -ram, -op, -method, -method.optimal.background,
```

See also

ExtractROI (fit)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Extraction mode [selection] <put parameter description here>

Options:

- 0 — fit

Default: 0

Reference image [raster] <put parameter description here>

Default elevation [number] <put parameter description here>

Default: 0

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:extractroi', -in, -ram, -mode, -mode.fit.ref, -mode.fit.elev.default, -
```

See also

ExtractROI (standard)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Extraction mode [selection] <put parameter description here>

Options:

- 0 — standard

Default: 0

Start X [number] <put parameter description here>

Default: 0

Start Y [number] <put parameter description here>

Default: 0

Size X [number] <put parameter description here>

Default: 0

Size Y [number] <put parameter description here>

Default: 0

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:extractroistandard', -in, -ram, -mode, -startx, -starty, -sizex, -sizey, -o
```

See also

Images Concatenation

Description

<put algorithm description here>

Parameters

Input images list [multipleinput: rasters] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:imagesconcatenation', -il, -ram, -out)
```

See also

Image Tile Fusion

Description

<put algorithm description here>

Parameters

Input Tile Images [**multipleinput: rasters**] <put parameter description here>

Number of tile columns [**number**] <put parameter description here>

Default: *0*

Number of tile rows [**number**] <put parameter description here>

Default: *0*

Outputs

Output Image [**raster**] <put output description here>

Console usage

```
processing.runalg('otb:imagetilefusion', -il, -cols, -rows, -out)
```

See also

Read image information

Description

<put algorithm description here>

Parameters

Input Image [**raster**] <put parameter description here>

Display the OSSIM keywordlist [**boolean**] <put parameter description here>

Default: *True*

GCPs Id [**string**] <put parameter description here>

Default: *None*

GCPs Info [**string**] <put parameter description here>

Default: *None*

GCPs Image Coordinates [**string**] <put parameter description here>

Default: *None*

GCPs Geographic Coordinates [**string**] <put parameter description here>

Default: *None*

Outputs

Console usage

```
processing.runalg('otb:readimageinformation', -in, -keywordlist, -gcp.ids, -gcp.info, -gcp.imcoord
```

See also

Rescale Image

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Output min value [number] <put parameter description here>

Default: 0

Output max value [number] <put parameter description here>

Default: 255

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:rescaleimage', -in, -ram, -outmin, -outmax, -out)
```

See also

Split Image

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output Image [file] <put output description here>

Console usage

```
processing.runalg('otb:splitimage', -in, -ram, -out)
```

See also

.

18.4.6 Learning

Classification Map Regularization

Description

<put algorithm description here>

Parameters

Input classification image [raster] <put parameter description here>

Structuring element radius (in pixels) [number] <put parameter description here>

Default: *1*

Multiple majority: Undecided(X)/Original [boolean] <put parameter description here>

Default: *True*

Label for the NoData class [number] <put parameter description here>

Default: *0*

Label for the Undecided class [number] <put parameter description here>

Default: *0*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Outputs

Output regularized image [raster] <put output description here>

Console usage

```
processing.runalg('otb:classificationmapregularization', -io.in, -ip.radius, -ip.suvbool, -ip.nod
```

See also

ComputeConfusionMatrix (raster)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Ground truth [selection] <put parameter description here>

Options:

- 0 — raster

Default: 0

Input reference image [raster] <put parameter description here>

Value for nodata pixels [number] <put parameter description here>

Default: 0

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:computeconfusionmatrixraster', -in, -ref, -ref.raster.in, -nodatalabel, -r
```

See also

ComputeConfusionMatrix (vector)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Ground truth [selection] <put parameter description here>

Options:

- 0 — vector

Default: 0

Input reference vector data [file] <put parameter description here>

Field name [string] Optional.

<put parameter description here>

Default: *Class*

Value for nodata pixels [number] <put parameter description here>

Default: 0

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Outputs

Matrix output [file] <put output description here>

Console usage

```
processing.runalg('otb:computeconfusionmatrixvector', -in, -ref, -ref.vector.in, -ref.vector.field)
```

See also

Compute Images second order statistics

Description

<put algorithm description here>

Parameters

Input images [multipleinput: rasters] <put parameter description here>

Background Value [number] <put parameter description here>

Default: *0.0*

Outputs

Output XML file [file] <put output description here>

Console usage

```
processing.runalg('otb:computeimagessecondorderstatistics', -il, -bv, -out)
```

See also

FusionOfClassifications (dempstershafer)

Description

<put algorithm description here>

Parameters

Input classifications [multipleinput: rasters] <put parameter description here>

Fusion method [selection] <put parameter description here>

Options:

- 0 — dempstershafer

Default: 0

Confusion Matrices [**multipleinput: files**] <put parameter description here>

Mass of belief measurement [**selection**] <put parameter description here>

Options:

- 0 — precision
- 1 — recall
- 2 — accuracy
- 3 — kappa

Default: 0

Label for the NoData class [**number**] <put parameter description here>

Default: 0

Label for the Undecided class [**number**] <put parameter description here>

Default: 0

Outputs

The output classification image [**raster**] <put output description here>

Console usage

```
processing.runalg('otb:fusionofclassificationsdempstershafer', -il, -method, -method.dempstershafer)
```

See also

FusionOfClassifications (majorityvoting)

Description

<put algorithm description here>

Parameters

Input classifications [**multipleinput: rasters**] <put parameter description here>

Fusion method [**selection**] <put parameter description here>

Options:

- 0 — majorityvoting

Default: 0

Label for the NoData class [**number**] <put parameter description here>

Default: 0

Label for the Undecided class [**number**] <put parameter description here>

Default: 0

Outputs

The output classification image [raster] <put output description here>

Console usage

```
processing.runalg('otb:fusionofclassificationsmajorityvoting', -il, -method, -nodatalabel, -undec
```

See also

Image Classification

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Input Mask [raster] Optional.

<put parameter description here>

Model file [file] <put parameter description here>

Statistics file [file] Optional.

<put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:imageclassification', -in, -mask, -model, -imstat, -ram, -out)
```

See also

SOM Classification

Description

<put algorithm description here>

Parameters

InputImage [raster] <put parameter description here>

ValidityMask [raster] Optional.

<put parameter description here>

TrainingProbability [number] <put parameter description here>

Default: 1

TrainingSetSize [number] <put parameter description here>

Default: 0

StreamingLines [number] <put parameter description here>

Default: 0

SizeX [number] <put parameter description here>

Default: 32

SizeY [number] <put parameter description here>

Default: 32

NeighborhoodX [number] <put parameter description here>

Default: 10

NeighborhoodY [number] <put parameter description here>

Default: 10

NumberIteration [number] <put parameter description here>

Default: 5

BetaInit [number] <put parameter description here>

Default: 1

BetaFinal [number] <put parameter description here>

Default: 0.1

InitialValue [number] <put parameter description here>

Default: 0

Available RAM (Mb) [number] <put parameter description here>

Default: 128

set user defined seed [number] <put parameter description here>

Default: 0

Outputs

OutputImage [raster] <put output description here>

SOM Map [raster] <put output description here>

Console usage

```
processing.runalg('otb:somclassification', -in, -vm, -tp, -ts, -sl, -sx, -sy, -nx, -ny, -ni, -bi,
```

See also

TrainImagesClassifier (ann)

Description

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: *0*

Maximum training sample size per class [**number**] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [**number**] <put parameter description here>

Default: *1000*

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: *True*

Training and validation sample ratio [**number**] <put parameter description here>

Default: *0.5*

Name of the discrimination field [**string**] <put parameter description here>

Default: *Class*

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — ann

Default: *0*

Train Method Type [**selection**] <put parameter description here>

Options:

- 0 — reg
- 1 — back

Default: *0*

Number of neurons in each intermediate layer [**string**] <put parameter description here>

Default: *None*

Neuron activation function type [**selection**] <put parameter description here>

Options:

- 0 — ident
- 1 — sig

- 2 — gau

Default: 1

Alpha parameter of the activation function [number] <put parameter description here>

Default: 1

Beta parameter of the activation function [number] <put parameter description here>

Default: 1

Strength of the weight gradient term in the BACKPROP method [number] <put parameter description here>

Default: 0.1

Strength of the momentum term (the difference between weights on the 2 previous iterations) [number] <put parameter description here>

Default: 0.1

Initial value Delta_0 of update-values Delta_{ij} in RPROP method [number] <put parameter description here>

Default: 0.1

Update-values lower limit Delta_{min} in RPROP method [number] <put parameter description here>

Default: 1e-07

Termination criteria [selection] <put parameter description here>

Options:

- 0 — iter
- 1 — eps
- 2 — all

Default: 2

Epsilon value used in the Termination criteria [number] <put parameter description here>

Default: 0.01

Maximum number of iterations used in the Termination criteria [number] <put parameter description here>

Default: 1000

set user defined seed [number] <put parameter description here>

Default: 0

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierann', -io.il, -io.vd, -io.imstat, -elev.default, -sam
```


See also**TrainImagesClassifier (bayes)****Description**

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: *0*

Maximum training sample size per class [**number**] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [**number**] <put parameter description here>

Default: *1000*

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: *True*

Training and validation sample ratio [**number**] <put parameter description here>

Default: *0.5*

Name of the discrimination field [**string**] <put parameter description here>

Default: *Class*

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — bayes

Default: *0*

set user defined seed [**number**] <put parameter description here>

Default: *0*

Outputs

Output confusion matrix [**file**] <put output description here>

Output model [**file**] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierbayes', -io.il, -io.vd, -io.imstat, -elev.default, -s
```

See also

TrainImagesClassifier (boost)

Description

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: 0

Maximum training sample size per class [**number**] <put parameter description here>

Default: 1000

Maximum validation sample size per class [**number**] <put parameter description here>

Default: 1000

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: True

Training and validation sample ratio [**number**] <put parameter description here>

Default: 0.5

Name of the discrimination field [**string**] <put parameter description here>

Default: Class

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — boost

Default: 0

Boost Type [**selection**] <put parameter description here>

Options:

- 0 — discrete
- 1 — real
- 2 — logit
- 3 — gentle

Default: 1

Weak count [**number**] <put parameter description here>

Default: 100

Weight Trim Rate [**number**] <put parameter description here>

Default: 0.95

Maximum depth of the tree [number] <put parameter description here>

Default: *1*

set user defined seed [number] <put parameter description here>

Default: *0*

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierboost', -io.il, -io.vd, -io.imstat, -elev.default, -s
```

See also

TrainImagesClassifier (dt)

Description

<put algorithm description here>

Parameters

Input Image List [multipleinput: rasters] <put parameter description here>

Input Vector Data List [multipleinput: any vectors] <put parameter description here>

Input XML image statistics file [file] Optional.

<put parameter description here>

Default elevation [number] <put parameter description here>

Default: *0*

Maximum training sample size per class [number] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [number] <put parameter description here>

Default: *1000*

On edge pixel inclusion [boolean] <put parameter description here>

Default: *True*

Training and validation sample ratio [number] <put parameter description here>

Default: *0.5*

Name of the discrimination field [string] <put parameter description here>

Default: *Class*

Classifier to use for the training [selection] <put parameter description here>

Options:

- 0 — dt

Default: 0

Maximum depth of the tree [number] <put parameter description here>

Default: 65535

Minimum number of samples in each node [number] <put parameter description here>

Default: 10

Termination criteria for regression tree [number] <put parameter description here>

Default: 0.01

Cluster possible values of a categorical variable into K <= cat clusters to find a suboptimal
<put parameter description here>

Default: 10

K-fold cross-validations [number] <put parameter description here>

Default: 10

Set UseSelfRule flag to false [boolean] <put parameter description here>

Default: True

Set TruncatePrunedTree flag to false [boolean] <put parameter description here>

Default: True

set user defined seed [number] <put parameter description here>

Default: 0

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierdt', -io.il, -io.vd, -io.imstat, -elev.default, -samp
```

See also

TrainImagesClassifier (gbt)

Description

<put algorithm description here>

Parameters

Input Image List [multipleinput: rasters] <put parameter description here>

Input Vector Data List [multipleinput: any vectors] <put parameter description here>

Input XML image statistics file [file] Optional.

<put parameter description here>

Default elevation [number] <put parameter description here>

Default: *0*

Maximum training sample size per class [number] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [number] <put parameter description here>

Default: *1000*

On edge pixel inclusion [boolean] <put parameter description here>

Default: *True*

Training and validation sample ratio [number] <put parameter description here>

Default: *0.5*

Name of the discrimination field [string] <put parameter description here>

Default: *Class*

Classifier to use for the training [selection] <put parameter description here>

Options:

- 0 — gbt

Default: *0*

Number of boosting algorithm iterations [number] <put parameter description here>

Default: *200*

Regularization parameter [number] <put parameter description here>

Default: *0.01*

Portion of the whole training set used for each algorithm iteration [number]

<put parameter description here>

Default: *0.8*

Maximum depth of the tree [number] <put parameter description here>

Default: *3*

set user defined seed [number] <put parameter description here>

Default: *0*

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifiergbt', -io.il, -io.vd, -io.imstat, -elev.default, -sam
```

See also

TrainImagesClassifier (knn)

Description

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: 0

Maximum training sample size per class [**number**] <put parameter description here>

Default: 1000

Maximum validation sample size per class [**number**] <put parameter description here>

Default: 1000

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: *True*

Training and validation sample ratio [**number**] <put parameter description here>

Default: 0.5

Name of the discrimination field [**string**] <put parameter description here>

Default: *Class*

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — knn

Default: 0

Number of Neighbors [**number**] <put parameter description here>

Default: 32

set user defined seed [**number**] <put parameter description here>

Default: 0

Outputs

Output confusion matrix [**file**] <put output description here>

Output model [**file**] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierknn', -io.il, -io.vd, -io.imstat, -elev.default, -sam
```

See also

TrainImagesClassifier (libsvm)

Description

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: *0*

Maximum training sample size per class [**number**] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [**number**] <put parameter description here>

Default: *1000*

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: *True*

Training and validation sample ratio [**number**] <put parameter description here>

Default: *0.5*

Name of the discrimination field [**string**] <put parameter description here>

Default: *Class*

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — libsvm

Default: *0*

SVM Kernel Type [**selection**] <put parameter description here>

Options:

- 0 — linear
- 1 — rbf
- 2 — poly
- 3 — sigmoid

Default: *0*

Cost parameter C [number] <put parameter description here>

Default: *1*

Parameters optimization [boolean] <put parameter description here>

Default: *True*

set user defined seed [number] <put parameter description here>

Default: *0*

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierlibsvm', -io.il, -io.vd, -io.imstat, -elev.default, -
```

See also

TrainImagesClassifier (rf)

Description

<put algorithm description here>

Parameters

Input Image List [multipleinput: rasters] <put parameter description here>

Input Vector Data List [multipleinput: any vectors] <put parameter description here>

Input XML image statistics file [file] Optional.

<put parameter description here>

Default elevation [number] <put parameter description here>

Default: *0*

Maximum training sample size per class [number] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [number] <put parameter description here>

Default: *1000*

On edge pixel inclusion [boolean] <put parameter description here>

Default: *True*

Training and validation sample ratio [number] <put parameter description here>

Default: *0.5*

Name of the discrimination field [string] <put parameter description here>

Default: *Class*

Classifier to use for the training [selection] <put parameter description here>

Options:

- 0 — rf

Default: 0

Maximum depth of the tree [number] <put parameter description here>

Default: 5

Minimum number of samples in each node [number] <put parameter description here>

Default: 10

Termination Criteria for regression tree [number] <put parameter description here>

Default: 0

Cluster possible values of a categorical variable into K <= cat clusters to find a suboptimal <put parameter description here>

Default: 10

Size of the randomly selected subset of features at each tree node [number] <put parameter description here>

Default: 0

Maximum number of trees in the forest [number] <put parameter description here>

Default: 100

Sufficient accuracy (OOB error) [number] <put parameter description here>

Default: 0.01

set user defined seed [number] <put parameter description here>

Default: 0

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifierrf', -io.il, -io.vd, -io.imstat, -elev.default, -samp
```

See also

TrainImagesClassifier (svm)

Description

<put algorithm description here>

Parameters

Input Image List [**multipleinput: rasters**] <put parameter description here>

Input Vector Data List [**multipleinput: any vectors**] <put parameter description here>

Input XML image statistics file [**file**] Optional.

<put parameter description here>

Default elevation [**number**] <put parameter description here>

Default: *0*

Maximum training sample size per class [**number**] <put parameter description here>

Default: *1000*

Maximum validation sample size per class [**number**] <put parameter description here>

Default: *1000*

On edge pixel inclusion [**boolean**] <put parameter description here>

Default: *True*

Training and validation sample ratio [**number**] <put parameter description here>

Default: *0.5*

Name of the discrimination field [**string**] <put parameter description here>

Default: *Class*

Classifier to use for the training [**selection**] <put parameter description here>

Options:

- 0 — svm

Default: *0*

SVM Model Type [**selection**] <put parameter description here>

Options:

- 0 — csvc
- 1 — nusvc
- 2 — oneclass

Default: *0*

SVM Kernel Type [**selection**] <put parameter description here>

Options:

- 0 — linear
- 1 — rbf
- 2 — poly
- 3 — sigmoid

Default: *0*

Cost parameter C [**number**] <put parameter description here>

Default: *1*

Parameter nu of a SVM optimization problem (NU_SVC / ONE_CLASS) [**number**] <put parameter description here>

Default: *0*

Parameter coef0 of a kernel function (POLY / SIGMOID) [number] <put parameter description here>

Default: 0

Parameter gamma of a kernel function (POLY / RBF / SIGMOID) [number] <put parameter description here>

Default: 1

Parameter degree of a kernel function (POLY) [number] <put parameter description here>

Default: 1

Parameters optimization [boolean] <put parameter description here>

Default: *True*

set user defined seed [number] <put parameter description here>

Default: 0

Outputs

Output confusion matrix [file] <put output description here>

Output model [file] <put output description here>

Console usage

```
processing.runalg('otb:trainimagesclassifiersvm', -io.il, -io.vd, -io.imstat, -elev.default, -sam
```

See also

Unsupervised KMeans image classification

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Validity Mask [raster] Optional.

<put parameter description here>

Training set size [number] <put parameter description here>

Default: 100

Number of classes [number] <put parameter description here>

Default: 5

Maximum number of iterations [number] <put parameter description here>

Default: 1000

Convergence threshold [number] <put parameter description here>

Default: *0.0001*

Outputs

Output Image [raster] <put output description here>

Centroid filename [file] <put output description here>

Console usage

```
processing.runalg('otb:unsupervisedkmeansimageclassification', -in, -ram, -vm, -ts, -nc, -maxit, ...
```

See also

.

18.4.7 Miscellaneous

Band Math

Description

<put algorithm description here>

Parameters

Input image list [multipleinput: rasters] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Expression [string] <put parameter description here>

Default: *None*

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:bandmath', -il, -ram, -exp, -out)
```

See also

ComputeModulusAndPhase-one (OneEntry)

Description

<put algorithm description here>

Parameters

Number Of inputs [selection] <put parameter description here>

Options:

- 0 — one

Default: 0

Input image [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Modulus [raster] <put output description here>

Phase [raster] <put output description here>

Console usage

```
processing.runalg('otb:computemodulusandphaseoneoneentry', -nbininput, -nbininput.one.in, -ram, -mod,
```

See also

ComputeModulusAndPhase-two (TwoEntries)

Description

<put algorithm description here>

Parameters

Number Of inputs [selection] <put parameter description here>

Options:

- 0 — two

Default: 0

Real part input [raster] <put parameter description here>

Imaginary part input [raster] <put parameter description here>

Available RAM (Mb) [number] <put parameter description here>

Default: 128

Outputs

Modulus [raster] <put output description here>

Phase [raster] <put output description here>

Console usage

```
processing.runalg('otb:computemodulusandphasetwoentries', -nbinput, -nbinput.two.re, -nbinput.t
```

See also

Images comparaison

Description

<put algortithm description here>

Parameters

Reference image [raster] <put parameter description here>

Reference image channel [number] <put parameter description here>

Default: *1*

Measured image [raster] <put parameter description here>

Measured image channel [number] <put parameter description here>

Default: *1*

Start X [number] <put parameter description here>

Default: *0*

Start Y [number] <put parameter description here>

Default: *0*

Size X [number] <put parameter description here>

Default: *0*

Size Y [number] <put parameter description here>

Default: *0*

Outputs

Console usage

```
processing.runalg('otb:imagescomparaison', -ref.in, -ref.channel, -meas.in, -meas.channel, -roi.st
```

See also

Image to KMZ Export

Description

<put algortithm description here>

Parameters

Input image [raster] <put parameter description here>

Tile Size [number] <put parameter description here>

Default: *512*

Image logo [raster] Optional.

<put parameter description here>

Image legend [raster] Optional.

<put parameter description here>

Default elevation [number] <put parameter description here>

Default: *0*

Outputs

Output .kmz product [file] <put output description here>

Console usage

```
processing.runalg('otb:imagetokmzexport', -in, -tilesize, -logo, -legend, -elev.default, -out)
```

See also

.

18.4.8 Segmentation

Connected Component Segmentation

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Mask expression [string] Optional.

<put parameter description here>

Default: *None*

Connected Component Expression [string] <put parameter description here>

Default: *None*

Minimum Object Size [number] <put parameter description here>

Default: *2*

OBIA Expression [string] Optional.

<put parameter description here>

Default: *None*

Default elevation [number] <put parameter description here>

Default: *0*

Outputs

Output Shape [vector] <put output description here>

Console usage

```
processing.runalg('otb:connectedcomponentsegmentation', -in, -mask, -expr, -minsize, -obia, -elev
```

See also

Exact Large-Scale Mean-Shift segmentation, step 2

Description

<put algorithm description here>

Parameters

Filtered image [raster] <put parameter description here>

Spatial image [raster] Optional.

<put parameter description here>

Range radius [number] <put parameter description here>

Default: *15*

Spatial radius [number] <put parameter description here>

Default: *5*

Minimum Region Size [number] <put parameter description here>

Default: *0*

Size of tiles in pixel (X-axis) [number] <put parameter description here>

Default: *500*

Size of tiles in pixel (Y-axis) [number] <put parameter description here>

Default: *500*

Directory where to write temporary files [file] Optional.

<put parameter description here>

Temporary files cleaning [boolean] <put parameter description here>

Default: *True*

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:exactlargescalemeanshiftsegmentationstep2', -in, -inpos, -ranger, -spatial,
```

See also

Exact Large-Scale Mean-Shift segmentation, step 3 (optional)

Description

<put algorithm description here>

Parameters

Input image [raster] <put parameter description here>

Segmented image [raster] <put parameter description here>

Minimum Region Size [number] <put parameter description here>

Default: 50

Size of tiles in pixel (X-axis) [number] <put parameter description here>

Default: 500

Size of tiles in pixel (Y-axis) [number] <put parameter description here>

Default: 500

Outputs

Output Image [raster] <put output description here>

Console usage

```
processing.runalg('otb:exactlargescalemeanshiftsegmentationstep3optional', -in, -inseg, -minsize,
```

See also

Exact Large-Scale Mean-Shift segmentation, step 4

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Segmented image [raster] <put parameter description here>

Size of tiles in pixel (X-axis) [number] <put parameter description here>

Default: *500*

Size of tiles in pixel (Y-axis) [number] <put parameter description here>

Default: *500*

Outputs

Output GIS vector file [vector] <put output description here>

Console usage

```
processing.runalg('otb:exactlargescalemeanshiftsegmentationstep4', -in, -inseg, -tilesizex, -tile
```

See also

Hoover compare segmentation

Description

<put algorithm description here>

Parameters

Input ground truth [raster] <put parameter description here>

Input machine segmentation [raster] <put parameter description here>

Background label [number] <put parameter description here>

Default: *0*

Overlapping threshold [number] <put parameter description here>

Default: *0.75*

Correct detection score [number] <put parameter description here>

Default: *0.0*

Over-segmentation score [number] <put parameter description here>

Default: *0.0*

Under-segmentation score [number] <put parameter description here>

Default: *0.0*

Missed detection score [number] <put parameter description here>

Default: *0.0*

Outputs

Colored ground truth output [raster] <put output description here>

Colored machine segmentation output [raster] <put output description here>

Console usage

```
processing.runalg('otb:hoovercomparesegmentation', -ingt, -inms, -bg, -th, -rc, -rf, -ra, -rm, -o
```

See also

Segmentation (cc)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Segmentation algorithm [selection] <put parameter description here>

Options:

- 0 — cc

Default: 0

Condition [string] <put parameter description here>

Default: *None*

Processing mode [selection] <put parameter description here>

Options:

- 0 — vector

Default: 0

Writing mode for the output vector file [selection] <put parameter description here>

Options:

- 0 — ulco
- 1 — ovw
- 2 — ulovw
- 3 — ulu

Default: 0

Mask Image [raster] Optional.

<put parameter description here>

8-neighbor connectivity [boolean] <put parameter description here>

Default: *True*

Stitch polygons [**boolean**] <put parameter description here>

Default: *True*

Minimum object size [**number**] <put parameter description here>

Default: *1*

Simplify polygons [**number**] <put parameter description here>

Default: *0.1*

Layer name [**string**] <put parameter description here>

Default: *layer*

Geometry index field name [**string**] <put parameter description here>

Default: *DN*

Tiles size [**number**] <put parameter description here>

Default: *1024*

Starting geometry index [**number**] <put parameter description here>

Default: *1*

OGR options for layer creation [**string**] Optional.

<put parameter description here>

Default: *None*

Outputs

Output vector file [**vector**] <put output description here>

Console usage

```
processing.runalg('otb:segmentationcc', -in, -filter, -filter.cc.expr, -mode, -mode.vector.outmode)
```

See also

Segmentation (edison)

Description

<put algorithm description here>

Parameters

Input Image [**raster**] <put parameter description here>

Segmentation algorithm [**selection**] <put parameter description here>

Options:

- 0 — edison

Default: *0*

Spatial radius [**number**] <put parameter description here>

Default: *5*

Range radius [number] <put parameter description here>

Default: *15*

Minimum region size [number] <put parameter description here>

Default: *100*

Scale factor [number] <put parameter description here>

Default: *1*

Processing mode [selection] <put parameter description here>

Options:

- 0 — vector

Default: *0*

Writing mode for the output vector file [selection] <put parameter description here>

Options:

- 0 — ulco
- 1 — ovw
- 2 — ulovw
- 3 — ulu

Default: *0*

Mask Image [raster] Optional.

<put parameter description here>

8-neighbor connectivity [boolean] <put parameter description here>

Default: *True*

Stitch polygons [boolean] <put parameter description here>

Default: *True*

Minimum object size [number] <put parameter description here>

Default: *1*

Simplify polygons [number] <put parameter description here>

Default: *0.1*

Layer name [string] <put parameter description here>

Default: *layer*

Geometry index field name [string] <put parameter description here>

Default: *DN*

Tiles size [number] <put parameter description here>

Default: *1024*

Starting geometry index [number] <put parameter description here>

Default: *1*

OGR options for layer creation [string] Optional.

<put parameter description here>

Default: *None*

Outputs

Output vector file [**vector**] <put output description here>

Console usage

```
processing.runalg('otb:segmentationedison', -in, -filter, -filter.edison.spatialr, -filter.edison
```

See also

Segmentation (meanshift)

Description

<put algorithm description here>

Parameters

Input Image [**raster**] <put parameter description here>

Segmentation algorithm [**selection**] <put parameter description here>

Options:

- 0 — meanshift

Default: 0

Spatial radius [**number**] <put parameter description here>

Default: 5

Range radius [**number**] <put parameter description here>

Default: 15

Mode convergence threshold [**number**] <put parameter description here>

Default: 0.1

Maximum number of iterations [**number**] <put parameter description here>

Default: 100

Minimum region size [**number**] <put parameter description here>

Default: 100

Processing mode [**selection**] <put parameter description here>

Options:

- 0 — vector

Default: 0

Writing mode for the output vector file [**selection**] <put parameter description here>

Options:

- 0 — ulco
- 1 — ovw
- 2 — ulovw

- 3 — ulu

Default: *0*

Mask Image [raster] Optional.

<put parameter description here>

8-neighbor connectivity [boolean] <put parameter description here>

Default: *True*

Stitch polygons [boolean] <put parameter description here>

Default: *True*

Minimum object size [number] <put parameter description here>

Default: *1*

Simplify polygons [number] <put parameter description here>

Default: *0.1*

Layer name [string] <put parameter description here>

Default: *layer*

Geometry index field name [string] <put parameter description here>

Default: *DN*

Tiles size [number] <put parameter description here>

Default: *1024*

Starting geometry index [number] <put parameter description here>

Default: *1*

OGR options for layer creation [string] Optional.

<put parameter description here>

Default: *None*

Outputs

Output vector file [vector] <put output description here>

Console usage

```
processing.runalg('otb:segmentationmeanshift', -in, -filter, -filter.meanshift.spatialr, -filter.r
```

See also

Segmentation (mprofiles)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Segmentation algorithm [selection] <put parameter description here>

Options:

- 0 — mprofiles

Default: 0

Profile Size [number] <put parameter description here>

Default: 5

Initial radius [number] <put parameter description here>

Default: 1

Radius step. [number] <put parameter description here>

Default: 1

Threshold of the final decision rule [number] <put parameter description here>

Default: 1

Processing mode [selection] <put parameter description here>

Options:

- 0 — vector

Default: 0

Writing mode for the output vector file [selection] <put parameter description here>

Options:

- 0 — ulco
- 1 — ovw
- 2 — ulovw
- 3 — ulu

Default: 0

Mask Image [raster] Optional.

<put parameter description here>

8-neighbor connectivity [boolean] <put parameter description here>

Default: *True*

Stitch polygons [boolean] <put parameter description here>

Default: *True*

Minimum object size [number] <put parameter description here>

Default: 1

Simplify polygons [number] <put parameter description here>

Default: 0.1

Layer name [string] <put parameter description here>

Default: *layer*

Geometry index field name [string] <put parameter description here>

Default: *DN*

Tiles size [number] <put parameter description here>

Default: *1024*

Starting geometry index [number] <put parameter description here>

Default: *1*

OGR options for layer creation [string] Optional.

<put parameter description here>

Default: *None*

Outputs

Output vector file [vector] <put output description here>

Console usage

```
processing.runalg('otb:segmentationmprofiles', -in, -filter, -filter.mprofiles.size, -filter.mpro
```

See also

Segmentation (watershed)

Description

<put algorithm description here>

Parameters

Input Image [raster] <put parameter description here>

Segmentation algorithm [selection] <put parameter description here>

Options:

- 0 — watershed

Default: *0*

Depth Threshold [number] <put parameter description here>

Default: *0.01*

Flood Level [number] <put parameter description here>

Default: *0.1*

Processing mode [selection] <put parameter description here>

Options:

- 0 — vector

Default: *0*

Writing mode for the output vector file [selection] <put parameter description here>

Options:

- 0 — ulco
- 1 — ovw
- 2 — ulovw
- 3 — ulu

Default: *0*

Mask Image [raster] Optional.

<put parameter description here>

8-neighbor connectivity [boolean] <put parameter description here>

Default: *True*

Stitch polygons [boolean] <put parameter description here>

Default: *True*

Minimum object size [number] <put parameter description here>

Default: *1*

Simplify polygons [number] <put parameter description here>

Default: *0.1*

Layer name [string] <put parameter description here>

Default: *layer*

Geometry index field name [string] <put parameter description here>

Default: *DN*

Tiles size [number] <put parameter description here>

Default: *1024*

Starting geometry index [number] <put parameter description here>

Default: *1*

OGR options for layer creation [string] Optional.

<put parameter description here>

Default: *None*

Outputs

Output vector file [vector] <put output description here>

Console usage

```
processing.runalg('otb:segmentationwatershed', -in, -filter, -filter.watershed.threshold, -filter
```

See also

.

18.4.9 Stereo

Stereo Framework

Description

<put algorithm description here>

Parameters

Input images list [**multipleinput: rasters**] <put parameter description here>

Couples list [**string**] Optional.

<put parameter description here>

Default: *None*

Image channel used for the block matching [**number**] <put parameter description here>

Default: *1*

Default elevation [**number**] <put parameter description here>

Default: *0*

Output resolution [**number**] <put parameter description here>

Default: *1*

NoData value [**number**] <put parameter description here>

Default: *-32768*

Method to fuse measures in each DSM cell [**selection**] <put parameter description here>

Options:

- 0 — max
- 1 — min
- 2 — mean
- 3 — acc

Default: *0*

Parameters estimation modes [**selection**] <put parameter description here>

Options:

- 0 — fit
- 1 — user

Default: *0*

Upper Left X [**number**] <put parameter description here>

Default: *0.0*

Upper Left Y [**number**] <put parameter description here>

Default: *0.0*

Size X [**number**] <put parameter description here>

Default: *0*

Size Y [number] <put parameter description here>

Default: 0

Pixel Size X [number] <put parameter description here>

Default: 0.0

Pixel Size Y [number] <put parameter description here>

Default: 0.0

Output Cartographic Map Projection [selection] <put parameter description here>

Options:

- 0 — utm
- 1 — lambert2
- 2 — lambert93
- 3 — wgs
- 4 — epsg

Default: 3

Zone number [number] <put parameter description here>

Default: 31

Northern Hemisphere [boolean] <put parameter description here>

Default: *True*

EPSG Code [number] <put parameter description here>

Default: 4326

Step of the deformation grid (in pixels) [number] <put parameter description here>

Default: 16

Sub-sampling rate for epipolar grid inversion [number] <put parameter description here>

Default: 10

Block-matching metric [selection] <put parameter description here>

Options:

- 0 — ssdmean
- 1 — ssd
- 2 — ncc
- 3 — lp

Default: 0

p value [number] <put parameter description here>

Default: 1

Radius of blocks for matching filter (in pixels) [number] <put parameter description here>

Default: 2

Minimum altitude offset (in meters) [number] <put parameter description here>

Default: -20

Maximum altitude offset (in meters) [number] <put parameter description here>

Default: *20*

Use bijection consistency in block matching strategy [boolean] <put parameter description here>

Default: *True*

Use median disparities filtering [boolean] <put parameter description here>

Default: *True*

Correlation metric threshold [number] <put parameter description here>

Default: *0.6*

Input left mask [raster] Optional.

<put parameter description here>

Input right mask [raster] Optional.

<put parameter description here>

Discard pixels with low local variance [number] <put parameter description here>

Default: *50*

Available RAM (Mb) [number] <put parameter description here>

Default: *128*

Outputs

Output DSM [raster] <put output description here>

Console usage

```
processing.runalg('otb:stereoframework', -input.il, -input.co, -input.channel, -elev.default, -out
```

See also

.

18.4.10 Vector

Concatenate

Description

<put algorithm description here>

Parameters

Input VectorDatas to concatenate [multipleinput: any vectors] <put parameter description here>

Outputs

Concatenated VectorData [**vector**] <put output description here>

Console usage

```
processing.runalg('otb:concatenate', -vd, -out)
```

See also

.

18.5 QGIS algorithm provider

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 fTools functionality, some algorithms from mmQGIS plugin and also adds its own algorithms.

.

18.5.1 Database

Import into PostGIS

Description

<put algorithm description here>

Parameters

Layer to import [**vector: any**] <put parameter description here>

Database (connection name) [**selection**] <put parameter description here>

Options:

- 0 — local

Default: 0

Schema (schema name) [**string**] <put parameter description here>

Default: *public*

Table to import to (leave blank to use layer name) [**string**] <put parameter description here>

Default: *(not set)*

Primary key field [**tablefield: any**] Optional.

<put parameter description here>

Geometry column [**string**] <put parameter description here>

Default: *geom*

Overwrite [boolean] <put parameter description here>

Default: *True*

Create spatial index [boolean] <put parameter description here>

Default: *True*

Convert field names to lowercase [boolean] <put parameter description here>

Default: *True*

Drop length constraints on character fields [boolean] <put parameter description here>

Default: *False*

Outputs

Console usage

```
processing.runalg('qgis:importintopostgis', input, database, schema, tablename, primary_key, geom)
```

See also

PostGIS execute SQL

Description

<put algorithm description here>

Parameters

Database [string] <put parameter description here>

Default: *(not set)*

SQL query [string] <put parameter description here>

Default: *(not set)*

Outputs

Console usage

```
processing.runalg('qgis:postgisexecutesql', database, sql)
```

See also

.

18.5.2 Raster general

Set style for raster layer

Description

<put algorithm description here>

Parameters

Raster layer [**raster**] <put parameter description here>

Style file [**file**] <put parameter description here>

Outputs

Styled layer [**raster**] <put output description here>

Console usage

```
processing.runalg('qgis:setstyleforrasterlayer', input, style)
```

See also

.

18.5.3 Raster

Hypsometric curves

Description

Calculate hypsometric curves for features of polygon layer and save them as CSV file for further processing.

Parameters

DEM to analyze [**raster**] DEM to use for calculating altitudes.

Boundary layer [**vector: polygon**] Polygonal vector layer with boundaries of areas used to calculate hypsometric curves.

Step [**number**] Distanse between curves.

Default: *100.0*

Use % of area instead of absolute value [**boolean**] Write area percentage to “Area” field of the CSV file instead of absolute area value.

Default: *False*

Outputs

Output directory [directory] Directory where output will be saved. For each feature from input vector layer CSV file with area and altitude values will be created.

File name consists of prefix `hystogram_` followed by layer name and feature ID.

Console usage

```
processing.runalg('qgis:hypsometriccurves', input_dem, boundary_layer, step, use_percentage, output_directory)
```

See also

Raster layer statistics

Description

Calculates basic statistics of the raster layer.

Parameters

Input layer [raster] Raster to analyze.

Outputs

Statistics [html] Analysis results in HTML format.

Minimum value [number] Minimum cell value.

Maximum value [number] Maximum cell value.

Sum [number] Sum of all cells values.

Mean value [number] Mean cell value.

valid cells count [number] Number of cell with data.

No-data cells count [number] Number of NODATA cells.

Standard deviation [number] Standard deviation of cells values.

Console usage

```
processing.runalg('qgis:rasterlayerstatistics', input_raster, output_html_file)
```

See also

Zonal Statistics

Description

Calculates some statistics values for pixels of input raster inside certain zones, defined as polygon layer.

Following values calculated for each zone:

- minimum

- maximum
- sum
- count
- mean
- standard deviation
- number of unique values
- range
- variance

Parameters

Raster layer [**raster**] Raster to analyze.

Raster band [**number**] Number of raster band to analyze.

Default: *1*

Vector layer containing zones [**vector: polygon**] Layer with zones boundaries.

Output column prefix [**string**] Prefix for output fields.

Default: *_*

Load whole raster in memory [**boolean**] Determines if raster band will be loaded in memory (**True**) or readed by chunks (**False**). Useful only when disk IO or raster scanning inefficiencies are your limiting factor.

Default: *True*

Outputs

Output layer [**vector**] The resulting layer. Basically this is same layer as zones layer with new columns containing statistics added.

Console usage

```
processing.runalg('qgis:zonalstatistics', input_raster, raster_band, input_vector, column_prefix,
```

See also

.

18.5.4 Table

Frequency analysis

Description

<put algorithm description here>

Parameters

input [vector: any] <put parameter description here>

fields [string] <put parameter description here>

Default: *(not set)*

Outputs

output [table] <put output description here>

Console usage

```
processing.runalg('qgis:frequencyanalysis', input, fields, output)
```

See also

.

18.5.5 Vector analysis

Count points in polygon

Description

Counts the number of points present in each feature of a polygon layer.

Parameters

Polygons [vector: polygon] Polygons layer.

Points [vector: point] Points layer.

Count field name [string] The name of the attribute table column containing the points number.

Default: *NUMPOINTS*

Outputs

Result [vector] Resulting layer with the attribute table containing the new column of the points count.

Console usage

```
processing.runalg('qgis:countpointsinpolygon', polygons, points, field, output)
```

See also

Count points in polygon (weighted)

Description

Counts the number of points in each feature of a polygon layer and calculates the mean of the selected field for each feature of the polygon layer. These values will be added to the attribute table of the resulting polygon layer.

Parameters

Polygons [vector: polygon] Polygons layer.

Points [vector: point] Points layer.

Weight field [tablefield: any] Weight field of the points attribute table.

Count field name [string] Name of the column for the new weighted field.

Default: *NUMPOINTS*

Outputs

Result [vector] The resulting polygons layer.

Console usage

```
processing.runalg('qgis:countpointsinpolygonweighted', polygons, points, weight, field, output)
```

See also

Count unique points in polygon

Description

Counts the number of unique values of a points in a polygons layer. Creates a new polygons layer with an extra column in the attribute table containing the count of unique values for each feature.

Parameters

Polygons [vector: polygon] Polygons layer.

Points [vector: point] Points layer.

Class field [tablefield: any] Points layer column name of the unique value chosen.

Count field name [string] Column name containing the count of unique values in the resulting polygons layer.

Default: *NUMPOINTS*

Outputs

Result [vector] The resulting polygons layer.

Console usage

```
processing.runalg('qgis:countuniquepointsinpolygon', polygons, points, classfield, field, output)
```

See also

Distance matrix

Description

<put algorithm description here>

Parameters

Input point layer [**vector: point**] <put parameter description here>

Input unique ID field [**tablefield: any**] <put parameter description here>

Target point layer [**vector: point**] <put parameter description here>

Target unique ID field [**tablefield: any**] <put parameter description here>

Output matrix type [**selection**] <put parameter description here>

Options:

- 0 — Linear ($N \times k \times 3$) distance matrix
- 1 — Standard ($N \times T$) distance matrix
- 2 — Summary distance matrix (mean, std. dev., min, max)

Default: 0

Use only the nearest (k) target points [**number**] <put parameter description here>

Default: 0

Outputs

Distance matrix [**table**] <put output description here>

Console usage

```
processing.runalg('qgis:distancematrix', input_layer, input_field, target_layer, target_field, ma
```

See also

Distance to nearest hub

Description

<put algorithm description here>

Parameters

Source points layer [vector: any] <put parameter description here>

Destination hubs layer [vector: any] <put parameter description here>

Hub layer name attribute [tablefield: any] <put parameter description here>

Output shape type [selection] <put parameter description here>

Options:

- 0 — Point
- 1 — Line to hub

Default: 0

Measurement unit [selection] <put parameter description here>

Options:

- 0 — Meters
- 1 — Feet
- 2 — Miles
- 3 — Kilometers
- 4 — Layer units

Default: 0

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:distancetonearesthub', points, hubs, field, geometry, unit, output)
```

See also

Generate points (pixel centroids) along line

Description

<put algorithm description here>

Parameters

Raster layer [raster] <put parameter description here>

Vector layer [vector: line] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:generatepointspixelcentroidsalongline', input_raster, input_vector, output)
```

See also

Generate points (pixel centroids) inside polygons

Description

<put algorithm description here>

Parameters

Raster layer [raster] <put parameter description here>

Vector layer [vector: polygon] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:generatepointspixelcentroidsinsidepolygons', input_raster, input_vector, output)
```

See also

Hub lines

Description

Creates hub and spoke diagrams with lines drawn from points on the `Spoke Point` layer to matching points in the `Hub Point` 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.

Parameters

Hub point layer [vector: any] <put parameter description here>

Hub ID field [tablefield: any] <put parameter description here>

Spoke point layer [vector: any] <put parameter description here>

Spoke ID field [tablefield: any] <put parameter description here>

Outputs

Output [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:hublines', hubs, hub_field, spokes, spoke_field, output)
```

See also

Mean coordinate(s)

Description

Calculates the mean of the coordinates of a layer starting from a field of the attribute table.

Parameters

Input layer [**vector: any**] <put parameter description here>

Weight field [**tablefield: numeric**] Optional.

Field to use if you want to perform a weighted mean.

Unique ID field [**tablefield: numeric**] Optional.

Unique field on which the calculation of the mean will be made.

Outputs

Result [**vector**] The resulting points layer.

Console usage

```
processing.runalg('qgis:meancoordinates', points, weight, uid, output)
```

See also

Nearest neighbour analysis

Description

<put algorithm description here>

Parameters

Points [**vector: point**] <put parameter description here>

Outputs

Result [**html**] <put output description here>

Observed mean distance [**number**] <put output description here>

Expected mean distance [**number**] <put output description here>

Nearest neighbour index [**number**] <put output description here>

Number of points [number] <put output description here>

Z-Score [number] <put output description here>

Console usage

```
processing.runalg('qgis:nearestneighbouranalysis', points, output)
```

See also

Sum line lengths

Description

<put algorithm description here>

Parameters

Lines [vector: line] <put parameter description here>

Polygons [vector: polygon] <put parameter description here>

Lines length field name [string] <put parameter description here>

Default: *LENGTH*

Lines count field name [string] <put parameter description here>

Default: *COUNT*

Outputs

Result [vector] <put output description here>

Console usage

```
processing.runalg('qgis:sumlinelengths', lines, polygons, len_field, count_field, output)
```

See also

.

18.5.6 Vector creation

Create grid

Description

Creates a grid.

Parameters

Grid type [selection] Grid type.

Options:

- 0 — Rectangle (line)
- 1 — Rectangle (polygon)
- 2 — Diamond (polygon)
- 3 — Hexagon (polygon)

Default: 0

Width [number] Horizontal extent of the grid.

Default: 360.0

Height [number] Vertical extent of the grid.

Default: 180.0

Horizontal spacing [number] X-axes spacing between the lines.

Default: 10.0

Vertical spacing [number] Y-axes spacing between the lines.

Default: 10.0

Center X [number] X-coordinate of the grid center.

Default: 0.0

Center Y [number] Y-coordinate of the grid center.

Default: 0.0

Output CRS [crs] Coordinate reference system for grid.

Default: *EPSG:4326*

Outputs

Output [vector] The resulting grid layer (lines or polygons).

Console usage

```
processing.runalg('qgis:creategrid', type, width, height, hspacing, vspacing, centerx, centery, c
```

See also

Points layer from table

Description

Creates points layer from geometryless table with columns that contain point coordinates.

Parameters

Input layer [table] Input table

X field [tablefield: any] Table column containing the X coordinate.

Y field [tablefield: any] Table column containing the Y coordinate.

Target CRS [crs] Coordinate reference system to use for layer.

Default: *EPSG:4326*

Outputs

Output layer [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:pointslayerfromtable', input, xfield, yfield, target_crs, output)
```

See also

Points to path

Description

<put algorithm description here>

Parameters

Input point layer [vector: point] <put parameter description here>

Group field [tablefield: any] <put parameter description here>

Order field [tablefield: any] <put parameter description here>

Date format (if order field is DateTime) [string] Optional.

<put parameter description here>

Default: *(not set)*

Outputs

Paths [vector] <put output description here>

Directory [directory] <put output description here>

Console usage

```
processing.runalg('qgis:pointstopath', vector, group_field, order_field, date_format, output_line)
```

See also

Random points along line

Description

<put algorithm description here>

Parameters

Input layer [vector: line] <put parameter description here>

Number of points [number] <put parameter description here>

Default: 1

Minimum distance [number] <put parameter description here>

Default: 0.0

Outputs

Random points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randompointsalongline', vector, point_number, min_distance, output)
```

See also

Random points in extent

Description

<put algorithm description here>

Parameters

Input extent [extent] <put parameter description here>

Default: 0,1,0,1

Points number [number] <put parameter description here>

Default: 1

Minimum distance [number] <put parameter description here>

Default: 0.0

Outputs

Random points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randompointsinextent', extent, point_number, min_distance, output)
```

See also

Random points in layer bounds

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon] <put parameter description here>

Points number [number] <put parameter description here>

Default: 1

Minimum distance [number] <put parameter description here>

Default: 0.0

Outputs

Random points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randompointsinlayerbounds', vector, point_number, min_distance, output)
```

See also

Random points inside polygons (fixed)

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon] <put parameter description here>

Sampling strategy [selection] <put parameter description here>

Options:

- 0 — Points count
- 1 — Points density

Default: 0

Number or density of points [number] <put parameter description here>

Default: *1.0*

Minimum distance [number] <put parameter description here>

Default: *0.0*

Outputs

Random points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randompointsinsidepolygonsfixed', vector, strategy, value, min_distance, ...)
```

See also

Random points inside polygons (variable)

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon] <put parameter description here>

Sampling strategy [selection] <put parameter description here>

Options:

- 0 — Points count
- 1 — Points density

Default: *0*

Number field [tablefield: numeric] <put parameter description here>

Minimum distance [number] <put parameter description here>

Default: *0.0*

Outputs

Random points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randompointsinsidepolygonsvariable', vector, strategy, field, min_distance, ...)
```

See also

Regular points

Description

<put algorithm description here>

Parameters

Input extent [extent] <put parameter description here>

Default: *0,1,0,1*

Point spacing/count [number] <put parameter description here>

Default: *0.0001*

Initial inset from corner (LH side) [number] <put parameter description here>

Default: *0.0*

Apply random offset to point spacing [boolean] <put parameter description here>

Default: *False*

Use point spacing [boolean] <put parameter description here>

Default: *True*

Outputs

Regular points [vector] <put output description here>

Console usage

```
processing.runalg('qgis:regularpoints', extent, spacing, inset, randomize, is_spacing, output)
```

See also

Vector grid

Description

<put algorithm description here>

Parameters

Grid extent [extent] <put parameter description here>

Default: *0,1,0,1*

X spacing [number] <put parameter description here>

Default: *0.0001*

Y spacing [number] <put parameter description here>

Default: *0.0001*

Grid type [selection] <put parameter description here>

Options:

- 0 — Output grid as polygons
- 1 — Output grid as lines

Default: 0

Outputs

Grid [vector] <put output description here>

Console usage

```
processing.runalg('qgis:vectorgrid', extent, step_x, step_y, type, output)
```

See also

.

18.5.7 Vector general

Delete duplicate geometries

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:deleteduplicategeometries', input, output)
```

See also

Join atributes by location

Description

<put algorithm description here>

Parameters

Target vector layer [**vector: any**] <put parameter description here>

Join vector layer [**vector: any**] <put parameter description here>

Attribute summary [**selection**] <put parameter description here>

Options:

- 0 — Take attributes of the first located feature
- 1 — Take summary of intersecting features

Default: 0

Statistics for summary (comma separated) [**string**] <put parameter description here>

Default: *sum,mean,min,max,median*

Output table [**selection**] <put parameter description here>

Options:

- 0 — Only keep matching records
- 1 — Keep all records (including non-matching target records)

Default: 0

Outputs

Output layer [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:joinattributesbylocation', target, join, summary, stats, keep, output)
```

See also

Join attributes table

Description

<put algorithm description here>

Parameters

Input layer [**vector: any**] <put parameter description here>

Input layer 2 [**table**] <put parameter description here>

Table field [**tablefield: any**] <put parameter description here>

Table field 2 [**tablefield: any**] <put parameter description here>

Outputs

Output layer [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:joinattributetable', input_layer, input_layer_2, table_field, table_fiel
```

See also

Merge vector layers

Description

<put algoritithm description here>

Parameters

Input layer 1 [vector: any] <put parameter description here>

Input layer 2 [vector: any] <put parameter description here>

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:mergevectorlayers', layer1, layer2, output)
```

See also

Polygon from layer extent

Description

<put algoritithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Calculate extent for each feature separately [boolean] <put parameter description here>

Default: *False*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:polygonfromlayerextent', input_layer, by_feature, output)
```

See also**Reproject layer****Description**

Reprojects a vector layer in a different CRS.

Parameters

Input layer [vector: any] Layer to reproject.

Target CRS [crs] Destination coordinate reference system.

Default: *EPSG:4326*

Outputs

Reprojected layer [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:reprojectlayer', input, target_crs, output)
```

See also**Save selected features****Description**

Saves the selected features as a new layer.

Parameters

Input layer [vector: any] Layer to process.

Outputs

Output layer with selected features [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:savesselectedfeatures', input_layer, output_layer)
```

See also

Set style for vector layer

Description

<put algorithm description here>

Parameters

Vector layer [vector: any] <put parameter description here>

Style file [file] <put parameter description here>

Outputs

Styled layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:setstyleforvectorlayer', input, style)
```

See also

Snap points to grid

Description

<put algorithm description here>

Parameters

Input Layer [vector: any] <put parameter description here>

Horizontal spacing [number] <put parameter description here>

Default: 0.1

Vertical spacing [number] <put parameter description here>

Default: 0.1

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:snappointstogrid', input, hspacing, vspacing, output)
```

See also

Split vector layer

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Unique ID field [tablefield: any] <put parameter description here>

Outputs

Output directory [directory] <put output description here>

Console usage

```
processing.runalg('qgis:splitvectorlayer', input, field, output)
```

See also

.

18.5.8 Vector geometry

Concave hull

Description

<put algorithm description here>

Parameters

Input point layer [vector: point] <put parameter description here>

Threshold (0-1, where 1 is equivalent with Convex Hull) [number] <put parameter description here>

Default: *0.3*

Allow holes [boolean] <put parameter description here>

Default: *True*

Split multipart geometry into singleparts geometries [boolean] <put parameter description here>

Default: *False*

Outputs

Concave hull [vector] <put output description here>

Console usage

```
processing.runalg('qgis:concavehull', input, alpha, holes, no_multigeometry, output)
```

See also

Convert geometry type

Description

Converts a geometry type to another one.

Parameters

Input layer [vector: any] Layer in input.

New geometry type [selection] Type of conversion to perform.

Options:

- 0 — Centroids
- 1 — Nodes
- 2 — Linestrings
- 3 — Multilinestrings
- 4 — Polygons

Default: 0

Outputs

Output [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:convertgeometrytype', input, type, output)
```

See also

Convex hull

Description

<put algorithm description here>

Parameters

Input layer [**vector: any**] <put parameter description here>

Field (optional, only used if creating convex hulls by classes) [**tablefield: any**]

Optional.

<put parameter description here>

Method [**selection**] <put parameter description here>

Options:

- 0 — Create single minimum convex hull
- 1 — Create convex hulls based on field

Default: 0

Outputs

Convex hull [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:convexhull', input, field, method, output)
```

See also

Create points along lines

Description

<put algorithm description here>

Parameters

lines [**vector: any**] <put parameter description here>

distance [**number**] <put parameter description here>

Default: 1

startpoint [**number**] <put parameter description here>

Default: 0

endpoint [**number**] <put parameter description here>

Default: 0

Outputs

output [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:createpointsalonglines', lines, distance, startpoint, endpoint, output)
```

See also

Delaunay triangulation

Description

<put algorithm description here>

Parameters

Input layer [vector: point] <put parameter description here>

Outputs

Delaunay triangulation [vector] <put output description here>

Console usage

```
processing.runalg('qgis:delaunaytriangulation', input, output)
```

See also

Densify geometries given an interval

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon, line] <put parameter description here>

Interval between Vertices to add [number] <put parameter description here>

Default: 1.0

Outputs

Densified layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:densifygeometriesgivenaninterval', input, interval, output)
```


See also**Densify geometries****Description**

<put algorithm description here>

Parameters

Input layer [vector: polygon, line] <put parameter description here>

Vertices to add [number] <put parameter description here>

Default: *1*

Outputs

Densified layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:densifygeometries', input, vertices, output)
```

See also**Dissolve****Description**

<put algorithm description here>

Parameters

Input layer [vector: polygon, line] <put parameter description here>

Dissolve all (do not use field) [boolean] <put parameter description here>

Default: *True*

Unique ID field [tablefield: any] Optional.

<put parameter description here>

Outputs

Dissolved [vector] <put output description here>

Console usage

```
processing.runalg('qgis:dissolve', input, dissolve_all, field, output)
```

See also

Eliminate sliver polygons

Description

<put algorithm description here>

Parameters

Input layer [**vector: polygon**] <put parameter description here>

Use current selection in input layer (works only if called from toolbox) [**boolean**]
<put parameter description here>

Default: *False*

Selection attribute [**tablefield: any**] <put parameter description here>

Comparison [**selection**] <put parameter description here>

Options:

- 0 — ==
- 1 — !=
- 2 — >
- 3 — >=
- 4 — <
- 5 — <=
- 6 — begins with
- 7 — contains

Default: *0*

Value [**string**] <put parameter description here>

Default: *0*

Merge selection with the neighbouring polygon with the [**selection**] <put parameter description here>

Options:

- 0 — Largest area
- 1 — Smallest Area
- 2 — Largest common boundary

Default: *0*

Outputs

Cleaned layer [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:eliminatesliverpolygons', input, keepselection, attribute, comparison, co
```

See also

Explode lines

Description

<put algorithm description here>

Parameters

Input layer [vector: line] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:explodelines', input, output)
```

See also

Extract nodes

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon, line] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:extractnodes', input, output)
```

See also

Fill holes

Description

<put algorithm description here>

Parameters

Polygons [vector: any] <put parameter description here>

Max area [number] <put parameter description here>

Default: *100000*

Outputs

Results [vector] <put output description here>

Console usage

```
processing.runalg('qgis:fillholes', polygons, max_area, results)
```

See also

Fixed distance buffer

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Distance [number] <put parameter description here>

Default: *10.0*

Segments [number] <put parameter description here>

Default: *5*

Dissolve result [boolean] <put parameter description here>

Default: *False*

Outputs

Buffer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:fixeddistancebuffer', input, distance, segments, dissolve, output)
```

See also

Keep n biggest parts

Description

<put algorithm description here>

Parameters

Polygons [vector: polygon] <put parameter description here>

To keep [number] <put parameter description here>

Default: 1

Outputs

Results [vector] <put output description here>

Console usage

```
processing.runalg('qgis:keepnbiggestparts', polygons, to_keep, results)
```

See also

Lines to polygons

Description

<put algorithm description here>

Parameters

Input layer [vector: line] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:linestopolygons', input, output)
```

See also

Multipart to singleparts

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:multiparttosingleparts', input, output)
```

See also

Points displacement

Description

Moves overlapped points at small distance, that they all become visible. The result is very similar to the output of the “Point displacement” renderer but it is permanent.

Parameters

Input layer [vector: point] Layer with overlapped points.

Displacement distance [number] Desired displacement distance **NOTE:** displacement distance should be in same units as layer.

Default: *0.00015*

Horizontal distribution for two point case [boolean] Controls distrobution direction in case of two overlapped points. If *True* points wwill be distributed horizontally, otherwise they will be distributed vertically.

Default: *True*

Outputs

Output layer [vector] The resulting layer with shifted overlapped points.

Console usage

```
processing.runalg('qgis:pointdisplacement', input_layer, distance, horizontal, output_layer)
```

See also

Polygon centroids

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:polygoncentroids', input_layer, output_layer)
```

See also

Polygonize

Description

<put algorithm description here>

Parameters

Input layer [vector: line] <put parameter description here>

Keep table structure of line layer [boolean] <put parameter description here>

Default: *False*

Create geometry columns [boolean] <put parameter description here>

Default: *True*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:polygonize', input, fields, geometry, output)
```

See also

Polygons to lines

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:polygonstolines', input, output)
```

See also

Simplify geometries

Description

<put algorithm description here>

Parameters

Input layer [vector: polygon, line] <put parameter description here>

Tolerance [number] <put parameter description here>

Default: *1.0*

Outputs

Simplified layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:simplifygeometries', input, tolerance, output)
```


See also**Singleparts to multipart****Description**

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Unique ID field [tablefield: any] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:singlepartstomultipart', input, field, output)
```

See also**Variable distance buffer****Description**

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Distance field [tablefield: any] <put parameter description here>

Segments [number] <put parameter description here>

Default: 5

Dissolve result [boolean] <put parameter description here>

Default: *False*

Outputs

Buffer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:variabledistancebuffer', input, field, segments, dissolve, output)
```

See also

Voronoi polygons

Description

<put algorithm description here>

Parameters

Input layer [vector: point] <put parameter description here>

Buffer region [number] <put parameter description here>

Default: *0.0*

Outputs

Voronoi polygons [vector] <put output description here>

Console usage

```
processing.runalg('qgis:voronoipolygons', input, buffer, output)
```

See also

.

18.5.9 Vector overlay

Clip

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Clip layer [vector: any] <put parameter description here>

Outputs

Clipped [vector] <put output description here>

Console usage

```
processing.runalg('qgis:clip', input, overlay, output)
```

See also

Difference

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Difference layer [vector: any] <put parameter description here>

Outputs

Difference [vector] <put output description here>

Console usage

```
processing.runalg('qgis:difference', input, overlay, output)
```

See also

Intersection

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Intersect layer [vector: any] <put parameter description here>

Outputs

Intersection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:intersection', input, input2, output)
```

See also

Line intersections

Description

<put algorithm description here>

Parameters

Input layer [vector: line] <put parameter description here>

Intersect layer [vector: line] <put parameter description here>

Input unique ID field [tablefield: any] <put parameter description here>

Intersect unique ID field [tablefield: any] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:lineintersections', input_a, input_b, field_a, field_b, output)
```

See also

Symetrical difference

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Difference layer [vector: any] <put parameter description here>

Outputs

Symetrical difference [vector] <put output description here>

Console usage

```
processing.runalg('qgis:symetricaldifference', input, overlay, output)
```

See also

Union

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Input layer 2 [vector: any] <put parameter description here>

Outputs

Union [vector] <put output description here>

Console usage

```
processing.runalg('qgis:union', input, input2, output)
```

See also

.

18.5.10 Vector selection

Extract by attribute

Description

<put algorithm description here>

Parameters

Input Layer [vector: any] <put parameter description here>

Selection attribute [tablefield: any] <put parameter description here>

Operator [selection] <put parameter description here>

Options:

- 0 — =
- 1 — !=
- 2 — >
- 3 — >=
- 4 — <
- 5 — <=
- 6 — begins with

- 7 — contains

Default: 0

Value [string] <put parameter description here>

Default: *(not set)*

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:extractbyattribute', input, field, operator, value, output)
```

See also

Extract by location

Description

<put algorithm description here>

Parameters

Layer to select from [vector: any] <put parameter description here>

Additional layer (intersection layer) [vector: any] <put parameter description here>

Include input features that touch the selection features [boolean] <put parameter description here>

Default: *False*

Include input features that overlap/cross the selection features [boolean] <put parameter description here>

Default: *False*

Include input features completely within the selection features [boolean] <put parameter description here>

Default: *False*

Outputs

Selection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:extractbylocation', input, intersect, touches, overlaps, within, output)
```

See also

Random extract

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — Number of selected features
- 1 — Percentage of selected features

Default: 0

Number/percentage of selected features [number] <put parameter description here>

Default: 10

Outputs

Selection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randomextract', input, method, number, output)
```

See also

Random extract within subsets

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

ID Field [tablefield: any] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — Number of selected features
- 1 — Percentage of selected features

Default: 0

Number/percentage of selected features [number] <put parameter description here>

Default: *10*

Outputs

Selection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randomextractwithinsubsets', input, field, method, number, output)
```

See also

Random selection

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — Number of selected features
- 1 — Percentage of selected features

Default: *0*

Number/percentage of selected features [number] <put parameter description here>

Default: *10*

Outputs

Selection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:randomselection', input, method, number)
```

See also

Random selection within subsets

Description

<put algorithm description here>

Parameters

Input layer [**vector: any**] <put parameter description here>

ID Field [**tablefield: any**] <put parameter description here>

Method [**selection**] <put parameter description here>

Options:

- 0 — Number of selected features
- 1 — Percentage of selected features

Default: 0

Number/percentage of selected features [**number**] <put parameter description here>

Default: 10

Outputs

Selection [**vector**] <put output description here>

Console usage

```
processing.runalg('qgis:randomselectionwithinsubsets', input, field, method, number)
```

See also

Select by attribute

Description

Selects and saves as new layer all features from input layer that satisfy condition.

NOTE: algorithm is case-sensitive (“qgis” is different from “Qgis” and “QGIS”)

Parameters

Input Layer [**vector: any**] Layer to process.

Selection attribute [**tablefield: any**] Field on which perform the selection.

Operator [**selection**] Comparison operator.

Options:

- 0 — =
- 1 — !=
- 2 — >
- 3 — >=
- 4 — <
- 5 — <=
- 6 — begins with
- 7 — contains

Default: *0*

Value [string] Value to compare.

Default: *(not set)*

Outputs

Output [vector] The resulting layer.

Console usage

```
processing.runalg('qgis:selectbyattribute', input, field, operator, value, output)
```

See also

Select by expression

Description

<put algorithm description here>

Parameters

Input Layer [vector: any] <put parameter description here>

Expression [string] <put parameter description here>

Default: *(not set)*

Modifys current selection by [selection] <put parameter description here>

Options:

- 0 — creating new selection
- 1 — adding to current selection
- 2 — removing from current selection

Default: *0*

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:selectbyexpression', layername, expression, method)
```

See also

Select by location

Description

<put algorithm description here>

Parameters

Layer to select from [vector: any] <put parameter description here>

Additional layer (intersection layer) [vector: any] <put parameter description here>

Include input features that touch the selection features [boolean] <put parameter description here>

Default: *False*

Include input features that overlap/cross the selection features [boolean] <put parameter description here>

Default: *False*

Include input features completely within the selection features [boolean] <put parameter description here>

Default: *False*

Modify current selection by [selection] <put parameter description here>

Options:

- 0 — creating new selection
- 1 — adding to current selection
- 2 — removing from current selection

Default: *0*

Outputs

Selection [vector] <put output description here>

Console usage

```
processing.runalg('qgis:selectbylocation', input, intersect, touches, overlaps, within, method)
```

See also

.

18.5.11 Vector table

Add autoincremental field

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:addautoincrementalfield', input, output)
```

See also

Add field to attributes table

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Field name [string] <put parameter description here>

Default: *(not set)*

Field type [selection] <put parameter description here>

Options:

- 0 — Integer
- 1 — Float
- 2 — String

Default: *0*

Field length [number] <put parameter description here>

Default: *10*

Field precision [number] <put parameter description here>

Default: *0*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:addfieldtoattributetable', input_layer, field_name, field_type, field_length)
```

See also

Advanced Python field calculator

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Result field name [string] <put parameter description here>

Default: *NewField*

Field type [selection] <put parameter description here>

Options:

- 0 — Integer
- 1 — Float
- 2 — String

Default: *0*

Field length [number] <put parameter description here>

Default: *10*

Field precision [number] <put parameter description here>

Default: *0*

Global expression [string] Optional.

<put parameter description here>

Default: *(not set)*

Formula [string] <put parameter description here>

Default: *value =*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:advancedpythonfieldcalculator', input_layer, field_name, field_type, field_name)
```

See also

Basic statistics for numeric fields

Description

<put algorithm description here>

Parameters

Input vector layer [**vector: any**] <put parameter description here>

Field to calculate statistics on [**tablefield: numeric**] <put parameter description here>

Outputs

Statistics for numeric field [**html**] <put output description here>

Coefficient of Variation [**number**] <put output description here>

Minimum value [**number**] <put output description here>

Maximum value [**number**] <put output description here>

Sum [**number**] <put output description here>

Mean value [**number**] <put output description here>

Count [**number**] <put output description here>

Range [**number**] <put output description here>

Median [**number**] <put output description here>

Number of unique values [**number**] <put output description here>

Standard deviation [**number**] <put output description here>

Console usage

```
processing.runalg('qgis:basicstatisticsfornumericfields', input_layer, field_name, output_html_filename)
```

See also

Basic statistics for text fields

Description

<put algorithm description here>

Parameters

Input vector layer [vector: any] <put parameter description here>

Field to calculate statistics on [tablefield: string] <put parameter description here>

Outputs

Statistics for text field [html] <put output description here>

Minimum length [number] <put output description here>

Maximum length [number] <put output description here>

Mean length [number] <put output description here>

Count [number] <put output description here>

Number of empty values [number] <put output description here>

Number of non-empty values [number] <put output description here>

Number of unique values [number] <put output description here>

Console usage

```
processing.runalg('qgis:basicstatisticsfortextfields', input_layer, field_name, output_html_file)
```

See also

Create equivalent numerical field

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Class field [tablefield: any] <put parameter description here>

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:createequivalentnumericalfield', input, field, output)
```

See also

Delete column

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Field to delete [tablefield: any] <put parameter description here>

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:deletecolumn', input, column, output)
```

See also

Export/Add geometry columns

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Calculate using [selection] <put parameter description here>

Options:

- 0 — Layer CRS
- 1 — Project CRS
- 2 — Ellipsoidal

Default: 0

Outputs

Output layer [vector] <put output description here>

Console usage


```
processing.runalg('qgis:exportaddgeometrycolumns', input, calc_method, output)
```

See also

Field calculator

Description

<put algorithm description here>

Parameters

Input layer [vector: any] <put parameter description here>

Result field name [string] <put parameter description here>

Default: *(not set)*

Field type [selection] <put parameter description here>

Options:

- 0 — Float
- 1 — Integer
- 2 — String
- 3 — Date

Default: *0*

Field length [number] <put parameter description here>

Default: *10*

Field precision [number] <put parameter description here>

Default: *3*

Create new field [boolean] <put parameter description here>

Default: *True*

Formula [string] <put parameter description here>

Default: *(not set)*

Outputs

Output layer [vector] <put output description here>

Console usage

```
processing.runalg('qgis:fieldcalculator', input_layer, field_name, field_type, field_length, field...
```

See also

List unique values

Description

Lists unique values of an attribute table field and counts their number.

Parameters

Input layer [vector: any] Layer to analyze.

Target field [tablefield: any] Field to analyze.

Outputs

Unique values [html] Analysis results in HTML format.

Total unique values [number] Total number of unique values in given field.

Unique values [string] List of all unique values in given field.

Console usage

```
processing.runalg('qgis:listuniquevalues', input_layer, field_name, output)
```

See also

Number of unique values in classes

Description

<put algorithm description here>

Parameters

input [vector: any] <put parameter description here>

class field [tablefield: any] <put parameter description here>

value field [tablefield: any] <put parameter description here>

Outputs

output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:numberofuniquevaluesinclasses', input, class_field, value_field, output)
```

See also

Statistics by categories

Description

<put algorithm description here>

Parameters

Input vector layer [vector: any] <put parameter description here>

Field to calculate statistics on [tablefield: numeric] <put parameter description here>

Field with categories [tablefield: any] <put parameter description here>

Outputs

Statistics [table] <put output description here>

Console usage

```
processing.runalg('qgis:statisticsbycategories', input_layer, values_field_name, categories_field)
```

See also

Text to float

Description

<put algorithm description here>

Parameters

Input Layer [vector: any] <put parameter description here>

Text attribute to convert to float [tablefield: string] <put parameter description here>

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('qgis:texttofloat', input, field, output)
```

See also

.

18.6 R algorithm provider

R also called GNU S, is a strongly functional language and environment to statistically explore data sets, make many graphical displays of data from custom data sets

Catatan: Please remember that Processing contains only R scripts, so you need to install R by yourself and configure Processing properly.

18.6.1 Basic statistics

Frequency table

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Outputs

R Console Output [html] <put output description here>

Console usage

```
processing.runalg('r:frequencytable', layer, field, r_console_output)
```

See also

Kolmogrov-Smirnov test

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Outputs

R Console Output [html] <put output description here>

Console usage

```
processing.runalg('r:kolmogrovsmirnovtest', layer, field, r_console_output)
```

See also

Summary statistics

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Outputs

R Console Output [html] <put output description here>

Console usage

```
processing.runalg('r:summarystatistics', layer, field, r_console_output)
```

See also

.

18.6.2 Home range

Characteristic hull method

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Outputs

Home_ranges [vector] <put output description here>

Console usage

```
processing.runalg('r:characteristichullmethod', layer, field, home_ranges)
```

See also

Kernel h ref

Description

<put algorithm description here>

Parameters

Layer [**vector: any**] <put parameter description here>

Field [**tablefield: any**] <put parameter description here>

Grid [**number**] <put parameter description here>

Default: *10.0*

Percentage [**number**] <put parameter description here>

Default: *10.0*

Folder [**directory**] Optional.

<put parameter description here>

Outputs

Home_ranges [**vector**] <put output description here>

Console usage

```
processing.runalg('r:kernelhref', layer, field, grid, percentage, folder, home_ranges)
```

See also

Minimum convex polygon

Description

<put algorithm description here>

Parameters

Layer [**vector: any**] <put parameter description here>

Percentage [**number**] <put parameter description here>

Default: *10.0*

Field [**tablefield: any**] <put parameter description here>

Outputs

Home_ranges [vector] <put output description here>

Console usage

```
processing.runalg('r:minimumconvexpolygon', layer, percentage, field, home_ranges)
```

See also

Single-linkage cluster analysis

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Percentage [number] <put parameter description here>

Default: *10.0*

Outputs

R Plots [html] <put output description here>

Home_ranges [vector] <put output description here>

Console usage

```
processing.runalg('r:singlelinkageclusteranalysis', layer, field, percentage, rplots, home_ranges)
```

See also

.

18.6.3 Point pattern

F function

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Nsim [number] <put parameter description here>

Default: *10.0*

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:ffunction', layer, nsim, rplots)
```

See also

G function

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Nsim [number] <put parameter description here>

Default: *10.0*

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:gfunction', layer, nsim, rplots)
```

See also

Monte-Carlo spatial randomness

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Simulations [number] <put parameter description here>

Default: *100.0*

Optional plot name [string] <put parameter description here>

Default: *(not set)*

Outputs

R Plots [html] <put output description here>

R Console Output [html] <put output description here>

Console usage

```
processing.runalg('r:montecarlospatialrandomness', layer, simulations, optional_plot_name, rplots)
```

See also

Quadrat analysis

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Outputs

R Plots [html] <put output description here>

R Console Output [html] <put output description here>

Console usage

```
processing.runalg('r:quadratanalysis', layer, rplots, r_console_output)
```

See also

Random sampling grid

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Size [number] <put parameter description here>

Default: *10.0*

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('r:randomsamplinggrid', layer, size, output)
```

See also

Regular sampling grid

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Size [number] <put parameter description here>

Default: *10.0*

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('r:regularsamplinggrid', layer, size, output)
```

See also

Relative distribution (distance covariate)

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Covariate [vector: any] <put parameter description here>

Covariate name [string] <put parameter description here>

Default: *mandatory_covariate_name_(no_spaces)*

x label [string] <put parameter description here>

Default: *(not set)*

Plot name [string] <put parameter description here>

Default: *(not set)*

Legend position [string] <put parameter description here>

Default: *float*

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:relativedistributiondistancecovariate', layer, covariate, covariate_name, x_
```

See also

Relative distribution (raster covariate)

Description

<put algorithm description here>

Parameters

points [vector: any] <put parameter description here>

covariate [raster] <put parameter description here>

covariate name [string] <put parameter description here>

Default: *mandatory_covariate_name_(no_spaces)*

x label [string] <put parameter description here>

Default: *(not set)*

plot name [string] <put parameter description here>

Default: *(not set)*

legend position [string] <put parameter description here>

Default: *float*

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:relativedistributionrastercovariate', points, covariate, covariate_name, x_l
```

See also

Ripley - Rasson spatial domain

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('r:ripleyrassonspatialdomain', layer, output)
```

See also

.

18.6.4 Raster processing

Advanced raster histogram

Description

<put algorithm description here>

Parameters

Layer [raster] <put parameter description here>

Dens or Hist [string] <put parameter description here>

Default: *Hist*

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:advancedrasterhistogram', layer, dens_or_hist, rplots)
```

See also

Raster histogram

Description

<put algorithm description here>

Parameters

Layer [raster] <put parameter description here>

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:rasterhistogram', layer, rplots)
```

See also

.

18.6.5 Vector processing

Histogram

Description

<put algorithm description here>

Parameters

Layer [vector: any] <put parameter description here>

Field [tablefield: any] <put parameter description here>

Outputs

R Plots [html] <put output description here>

Console usage

```
processing.runalg('r:histogram', layer, field, rplots)
```

See also

.

18.7 SAGA algorithm provider

SAGA (System for Automated Geoscientific Analyses) is a free, hybrid, cross-platform GIS software. SAGA provides many geoscientific methods which are bundled in so-called module libraries.

Catatan: Please remember that Processing contains only the interface description, so you need to install SAGA by yourself and configure Processing properly.

18.7.1 Geostatistics

Directional statistics for single grid

Description

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Points [**vector: any**] Optional.

<put parameter description here>

Direction [**Degree**] [**number**] <put parameter description here>

Default: *0.0*

Tolerance [**Degree**] [**number**] <put parameter description here>

Default: *0.0*

Maximum Distance [**Cells**] [**number**] <put parameter description here>

Default: *0*

Distance Weighting [**selection**] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *1.0*

Outputs

Arithmetic Mean [raster] <put output description here>

Difference from Arithmetic Mean [raster] <put output description here>

Minimum [raster] <put output description here>

Maximum [raster] <put output description here>

Range [raster] <put output description here>

Variance [raster] <put output description here>

Standard Deviation [raster] <put output description here>

Mean less Standard Deviation [raster] <put output description here>

Mean plus Standard Deviation [raster] <put output description here>

Deviation from Arithmetic Mean [raster] <put output description here>

Percentile [raster] <put output description here>

Directional Statistics for Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:directionalstatisticsforsinglegrid', grid, points, direction, tolerance, n)
```

See also

Fast representativeness

Description

<put algorithm description here>

Parameters

Input [raster] <put parameter description here>

Level of Generalisation [number] <put parameter description here>

Default: *16*

Outputs

Output [raster] <put output description here>

Output Lod [raster] <put output description here>

Output Seeds [raster] <put output description here>

Console usage

```
processing.runalg('saga:fastrepresentativeness', input, lod, result, result_lod, seeds)
```

See also

Geographically weighted multiple regression (points/grids)

Description

<put algorithm description here>

Parameters

Predictors [multipleinput: rasters] <put parameter description here>

Output of Regression Parameters [boolean] <put parameter description here>

Default: *True*

Points [vector: point] <put parameter description here>

Dependent Variable [tablefield: any] <put parameter description here>

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *1.0*

Search Range [selection] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [number] <put parameter description here>

Default: 100

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of observations
- 1 — [1] all points

Default: 0

Maximum Number of Observations [number] <put parameter description here>

Default: 10

Minimum Number of Observations [number] <put parameter description here>

Default: 4

Outputs

Regression [raster] <put output description here>

Coefficient of Determination [raster] <put output description here>

Regression Parameters [raster] <put output description here>

Residuals [vector] <put output description here>

Console usage

```
processing.runalg('saga:geographicallyweightedmultipleregressionpointsgrids', predictors, parameter)
```

See also

Geographically weighted multiple regression (points)

Description

<put algorithm description here>

Parameters

Points [vector: any] <put parameter description here>

Dependent Variable [tablefield: any] <put parameter description here>

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [number] <put parameter description here>

Default: 1

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: 1.0

Search Range [selection] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [number] <put parameter description here>

Default: 100

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of observations
- 1 — [1] all points

Default: 0

Maximum Number of Observations [number] <put parameter description here>

Default: 10

Minimum Number of Observations [number] <put parameter description here>

Default: 4

Outputs

Regression [vector] <put output description here>

Console usage

```
processing.runalg('saga:geographicallyweightedmultipleregressionpoints', points, dependent, distan
```

See also

Geographically weighted multiple regression

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Dependent Variable [tablefield: any] <put parameter description here>

Target Grids [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [number] <put parameter description here>

Default: 1

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: 1

Search Range [selection] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [number] <put parameter description here>

Default: 100

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of observations
- 1 — [1] all points

Default: 0

Maximum Number of Observations [number] <put parameter description here>

Default: 10

Minimum Number of Observations [number] <put parameter description here>

Default: 4

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Quality [raster] <put output description here>

Intercept [raster] <put output description here>

Quality [raster] <put output description here>

Intercept [raster] <put output description here>

Console usage

```
processing.runalg('saga:geographicallyweightedmultipleregression', points, dependent, target, dist
```

See also

Geographically weighted regression (points/grid)

Description

<put algorithm description here>

Parameters

Predictor [raster] <put parameter description here>

Points [vector: point] <put parameter description here>

Dependent Variable [tablefield: any] <put parameter description here>

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [number] <put parameter description here>

Default: 1

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: 1.0

Search Range [selection] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [number] <put parameter description here>

Default: 0

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of observations
- 1 — [1] all points

Default: 0

Maximum Number of Observations [number] <put parameter description here>

Default: 10

Minimum Number of Observations [number] <put parameter description here>

Default: 4

Outputs

Regression [raster] <put output description here>

Coefficient of Determination [raster] <put output description here>

Intercept [raster] <put output description here>

Slope [raster] <put output description here>

Residuals [vector] <put output description here>

Console usage

```
processing.runalg('saga:geographicallyweightedregressionpointsgrid', predictor, points, dependent,
```

See also

Geographically weighted regression

Description

<put algorithm description here>

Parameters

Points [**vector: point**] <put parameter description here>

Dependent Variable [**tablefield: any**] <put parameter description here>

Predictor [**tablefield: any**] <put parameter description here>

Target Grids [**selection**] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Distance Weighting [**selection**] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [**number**] <put parameter description here>

Default: 0

Inverse Distance Offset [**boolean**] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [**number**] <put parameter description here>

Default: 0.0

Search Range [**selection**] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [**number**] <put parameter description here>

Default: 100

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of observations
- 1 — [1] all points

Default: 0

Maximum Number of Observations [number] <put parameter description here>

Default: 10

Minimum Number of Observations [number] <put parameter description here>

Default: 4

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Quality [raster] <put output description here>

Intercept [raster] <put output description here>

Slope [raster] <put output description here>

Console usage

```
processing.runalg('saga:geographicallyweightedregression', points, dependent, predictor, target, 0)
```

See also

Global moran's i for grids

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Case of contiguity [selection] <put parameter description here>

Options:

- 0 — [0] Rook
 - 1 — [1] Queen
- Default: 0

Outputs

Result [table] <put output description here>

Console usage

```
processing.runalg('saga:globalmoransiforgrids', grid, contiguity, result)
```

See also

Minimum distance analysis

Description

Performs a complete distance analysis of a point layer:

- minimum distance of points
- maximum distance of points
- average distance of all the points
- standard deviation of the distance
- duplicated points

Parameters

Points [vector: point] Layer to analyze.

Outputs

Minimum Distance Analysis [table] The resulting table.

Console usage

```
processing.runalg('saga:minimumdistanceanalysis', points, table)
```

See also

Multi-band variation

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Radius [**Cells**] [**number**] <put parameter description here>

Default: *1*

Distance Weighting [**selection**] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [**number**] <put parameter description here>

Default: *1*

Inverse Distance Offset [**boolean**] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [**number**] <put parameter description here>

Default: *1.0*

Outputs

Mean Distance [**raster**] <put output description here>

Standard Deviation [**raster**] <put output description here>

Distance [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:multibandvariation', bands, radius, distance_weighting_weighting, distance_offset)
```

See also

Multiple regression analysis (grid/grids)

Description

<put algorithm description here>

Parameters

Dependent [**raster**] <put parameter description here>

Grids [**multipleinput: rasters**] <put parameter description here>

Grid Interpolation [**selection**] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Include X Coordinate [boolean] <put parameter description here>

Default: *True*

Include Y Coordinate [boolean] <put parameter description here>

Default: *True*

Method [selection] <put parameter description here>

Options:

- 0 — [0] include all
- 1 — [1] forward
- 2 — [2] backward
- 3 — [3] stepwise

Default: 0

P in [number] <put parameter description here>

Default: 5

P out [number] <put parameter description here>

Default: 5

Outputs

Regression [raster] <put output description here>

Residuals [raster] <put output description here>

Details: Coefficients [table] <put output description here>

Details: Model [table] <put output description here>

Details: Steps [table] <put output description here>

Console usage

```
processing.runalg('saga:multipleregressionanalysisgridgrids', dependent, grids, interpol, coord_x,
```

See also

Multiple regression analysis (points/grids)

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Shapes [**vector: any**] <put parameter description here>

Attribute [**tablefield: any**] <put parameter description here>

Grid Interpolation [**selection**] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Include X Coordinate [**boolean**] <put parameter description here>

Default: *True*

Include Y Coordinate [**boolean**] <put parameter description here>

Default: *True*

Method [**selection**] <put parameter description here>

Options:

- 0 — [0] include all
- 1 — [1] forward
- 2 — [2] backward
- 3 — [3] stepwise

Default: 0

P in [**number**] <put parameter description here>

Default: 5

P out [**number**] <put parameter description here>

Default: 5

Outputs

Details: **Coefficients** [**table**] <put output description here>

Details: **Model** [**table**] <put output description here>

Details: **Steps** [**table**] <put output description here>

Residuals [**vector**] <put output description here>

Regression [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:multipleregressionanalysispointsgrids', grids, shapes, attribute, interpo
```

See also

Polynomial regression

Description

<put algorithm description here>

Parameters

Points [vector: any] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Polynom [selection] <put parameter description here>

Options:

- 0 — [0] simple planar surface
- 1 — [1] bi-linear saddle
- 2 — [2] quadratic surface
- 3 — [3] cubic surface
- 4 — [4] user defined

Default: 0

Maximum X Order [number] <put parameter description here>

Default: 4

Maximum Y Order [number] <put parameter description here>

Default: 4

Maximum Total Order [number] <put parameter description here>

Default: 4

Trend Surface [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Residuals [vector] <put output description here>

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:polynomialregression', points, attribute, polynom, xorder, yorder, torder)
```

See also

Radius of variance (grid)

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Standard Deviation [number] <put parameter description here>

Default: 1.0

Maximum Search Radius (cells) [number] <put parameter description here>

Default: 20

Type of Output [selection] <put parameter description here>

Options:

- 0 — [0] Cells
- 1 — [1] Map Units

Default: 0

Outputs

Variance Radius [raster] <put output description here>

Console usage

```
processing.runalg('saga:radiusofvariancegrid', input, variance, radius, output, result)
```

See also

Regression analysis

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Shapes [vector: any] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Grid Interpolation [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Regression Function [selection] <put parameter description here>

Options:

- 0 — [0] $Y = a + b * X$ (linear)
- 1 — [1] $Y = a + b / X$
- 2 — [2] $Y = a / (b - X)$
- 3 — [3] $Y = a * X^b$ (power)
- 4 — [4] $Y = a e^{(b * X)}$ (exponential)
- 5 — [5] $Y = a + b * \ln(X)$ (logarithmic)

Default: 0

Outputs

Regression [raster] <put output description here>

Residuals [vector] <put output description here>

Console usage

```
processing.runalg('saga:regressionanalysis', grid, shapes, attribute, interpol, method, regression)
```

See also

Representativeness

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Radius (Cells) [number] <put parameter description here>

Default: *10*

Exponent [number] <put parameter description here>

Default: *1*

Outputs

Representativeness [raster] <put output description here>

Console usage

```
processing.runalg('saga:representativeness', input, radius, exponent, result)
```

See also

Residual analysis

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Radius (Cells) [number] <put parameter description here>

Default: *7*

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *1.0*

Outputs

Mean Value [raster] <put output description here>
Difference from Mean Value [raster] <put output description here>
Standard Deviation [raster] <put output description here>
Value Range [raster] <put output description here>
Minimum Value [raster] <put output description here>
Maximum Value [raster] <put output description here>
Deviation from Mean Value [raster] <put output description here>
Percentile [raster] <put output description here>

Console usage

```
processing.runalg('saga:residualanalysis', grid, radius, distance_weighting_weighting, distance_w
```

See also

Spatial point pattern analysis

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>
Vertex Distance [Degree] [number] <put parameter description here>
Default: 5

Outputs

Mean Centre [vector] <put output description here>
Standard Distance [vector] <put output description here>
Bounding Box [vector] <put output description here>

Console usage

```
processing.runalg('saga:spatialpointpatternanalysis', points, step, centre, stddist, bbox)
```

See also

Statistics for grids

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Outputs

Arithmetic Mean [**raster**] <put output description here>

Minimum [**raster**] <put output description here>

Maximum [**raster**] <put output description here>

Variance [**raster**] <put output description here>

Standard Deviation [**raster**] <put output description here>

Mean less Standard Deviation [**raster**] <put output description here>

Mean plus Standard Deviation [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:statisticsforgrids', grids, mean, min, max, var, stddev, stddevlo, stddevhi)
```

See also

Variogram cloud

Description

<put algorithm description here>

Parameters

Points [**vector: point**] <put parameter description here>

Attribute [**tablefield: any**] <put parameter description here>

Maximum Distance [**number**] <put parameter description here>

Default: *0.0*

Skip Number [**number**] <put parameter description here>

Default: *1*

Outputs

Variogram Cloud [**table**] <put output description here>

Console usage

```
processing.runalg('saga:variogramcloud', points, field, distmax, nskip, result)
```

See also

Variogram surface

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Number of Distance Classes [number] <put parameter description here>

Default: 10

Skip Number [number] <put parameter description here>

Default: 1

Outputs

Number of Pairs [raster] <put output description here>

Variogram Surface [raster] <put output description here>

Covariance Surface [raster] <put output description here>

Console usage

```
processing.runalg('saga:variogramsurface', points, field, distcount, nskip, count, variance, covar
```

See also

Zonal grid statistics

Description

<put algorithm description here>

Parameters

Zone Grid [raster] <put parameter description here>

Categorical Grids [multipleinput: rasters] Optional.

<put parameter description here>

Grids to analyse [multipleinput: rasters] Optional.

<put parameter description here>

Aspect [raster] Optional.

<put parameter description here>

Short Field Names [boolean] <put parameter description here>

Default: *True*

Outputs

Zonal Statistics [table] <put output description here>

Console usage

```
processing.runalg('saga:zonalgridstatistics', zones, catlist, statlist, aspect, shortnames, outta
```

See also

.

18.7.2 Grid analysis

Accumulated cost (anisotropic)

Description

<put algorithm description here>

Parameters

Cost Grid [raster] <put parameter description here>

Direction of max cost [raster] <put parameter description here>

Destination Points [raster] <put parameter description here>

k factor [number] <put parameter description here>

Default: *1*

Threshold for different route [number] <put parameter description here>

Default: *0*

Outputs

Accumulated Cost [raster] <put output description here>

Console usage

```
processing.runalg('saga:accumulatedcostanisotropic', cost, direction, points, k, threshold, accco
```

See also

Accumulated cost (isotropic)

Description

<put algorithm description here>

Parameters

Cost Grid [raster] <put parameter description here>

Destination Points [raster] <put parameter description here>

Threshold for different route [number] <put parameter description here>

Default: 0.0

Outputs

Accumulated Cost [raster] <put output description here>

Closest Point [raster] <put output description here>

Console usage

```
processing.runalg('saga:accumulatedcostisotropic', cost, points, threshold, acccost, closestpt)
```

See also

Aggregation index

Description

<put algorithm description here>

Parameters

Input Grid [raster] <put parameter description here>

Max. Number of Classes [number] <put parameter description here>

Default: 5

Outputs

Result [table] <put output description here>

Console usage

```
processing.runalg('saga:aggregationindex', input, maxnumclass, result)
```

See also**Analytical hierarchy process****Description**

<put algorithm description here>

Parameters

Input Grids [**multipleinput: rasters**] <put parameter description here>

Pairwise Comparisons Table [**table**] <put parameter description here>

Outputs

Output Grid [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:analyticalhierarchyprocess', grids, table, output)
```

See also**Cross-classification and tabulation****Description**

<put algorithm description here>

Parameters

Input Grid 1 [**raster**] <put parameter description here>

Input Grid 2 [**raster**] <put parameter description here>

Max. Number of Classes [**number**] <put parameter description here>

Default: 5

Outputs

Cross-Classification Grid [**raster**] <put output description here>

Cross-Tabulation Table [**table**] <put output description here>

Console usage

```
processing.runalg('saga:crossclassificationandtabulation', input, input2, maxnumclass, resultgrid)
```

See also

Fragmentation (alternative)

Description

<put algorithm description here>

Parameters

Classification [raster] <put parameter description here>

Class Identifier [number] <put parameter description here>

Default: 1

Neighborhood Min [number] <put parameter description here>

Default: 1

Neighborhood Max [number] <put parameter description here>

Default: 1

Level Aggregation [selection] <put parameter description here>

Options:

- 0 — [0] average
- 1 — [1] multiplicative

Default: 0

Add Border [boolean] <put parameter description here>

Default: *True*

Connectivity Weighting [number] <put parameter description here>

Default: 1.1

Minimum Density [Percent] [number] <put parameter description here>

Default: 10

Minimum Density for Interior Forest [Percent] [number] <put parameter description here>

Default: 99

Search Distance Increment [number] <put parameter description here>

Default: 0.0

Density from Neighbourhood [boolean] <put parameter description here>

Default: *True*

Outputs

Density [Percent] [raster] <put output description here>

Connectivity [Percent] [raster] <put output description here>

Fragmentation [raster] <put output description here>

Summary [table] <put output description here>

Console usage

```
processing.runalg('saga:fragmentationalternative', classes, class, neighborhood_min, neighborhood_max)
```

See also

Fragmentation classes from density and connectivity

Description

<put algorithm description here>

Parameters

Density [Percent] [raster] <put parameter description here>

Connectivity [Percent] [raster] <put parameter description here>

Add Border [boolean] <put parameter description here>

Default: *True*

Connectivity Weighting [number] <put parameter description here>

Default: *0*

Minimum Density [Percent] [number] <put parameter description here>

Default: *10*

Minimum Density for Interior Forest [Percent] [number] <put parameter description here>

Default: *99*

Outputs

Fragmentation [raster] <put output description here>

Console usage

```
processing.runalg('saga:fragmentationclassesfromdensityandconnectivity', density, connectivity, border)
```

See also

Fragmentation (standard)

Description

<put algorithm description here>

Parameters

Classification [raster] <put parameter description here>

Class Identifier [number] <put parameter description here>

Default: 1

Neighborhood Min [number] <put parameter description here>

Default: 1

Neighborhood Max [number] <put parameter description here>

Default: 3

Level Aggregation [selection] <put parameter description here>

Options:

- 0 — [0] average
- 1 — [1] multiplicative

Default: 0

Add Border [boolean] <put parameter description here>

Default: *True*

Connectivity Weighting [number] <put parameter description here>

Default: 1.1

Minimum Density [Percent] [number] <put parameter description here>

Default: 10

Minimum Density for Interior Forest [Percent] [number] <put parameter description here>

Default: 99

Neighborhood Type [selection] <put parameter description here>

Options:

- 0 — [0] square
- 1 — [1] circle

Default: 0

Include diagonal neighbour relations [boolean] <put parameter description here>

Default: *True*

Outputs

Density [Percent] [raster] <put output description here>

Connectivity [Percent] [raster] <put output description here>

Fragmentation [raster] <put output description here>

Summary [table] <put output description here>

Console usage

```
processing.runalg('saga:fragmentationstandard', classes, class, neighborhood_min, neighborhood_max)
```


See also

Layer of extreme value

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Method [**selection**] <put parameter description here>

Options:

- 0 — [0] Maximum
- 1 — [1] Minimum

Default: 0

Outputs

Result [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:layerofextremevalue', grids, criteria, result)
```

See also

Least cost paths

Description

<put algorithm description here>

Parameters

Source Point (s) [**vector: point**] <put parameter description here>

Accumulated cost [**raster**] <put parameter description here>

Values [**multipleinput: rasters**] Optional.

<put parameter description here>

Outputs

Profile (points) [**vector**] <put output description here>

Profile (lines) [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:leastcostpaths', source, dem, values, points, line)
```

See also

Ordered Weighted Averaging

Description

<put algorithm description here>

Parameters

Input Grids [**multipleinput: rasters**] <put parameter description here>

Weights [**fixedtable**] <put parameter description here>

Outputs

Output Grid [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:orderedweightedaveraging', grids, weights, output)
```

See also

Pattern analysis

Description

<put algorithm description here>

Parameters

Input Grid [**raster**] <put parameter description here>

Size of Analysis Window [**selection**] <put parameter description here>

Options:

- 0 — [0] 3 X 3
- 1 — [1] 5 X 5
- 2 — [2] 7 X 7

Default: 0

Max. Number of Classes [**number**] <put parameter description here>

Default: 0

Outputs

Relative Richness [raster] <put output description here>

Diversity [raster] <put output description here>

Dominance [raster] <put output description here>

Fragmentation [raster] <put output description here>

Number of Different Classes [raster] <put output description here>

Center Versus Neighbours [raster] <put output description here>

Console usage

```
processing.runalg('saga:patternanalysis', input, winsize, maxnumclass, relative, diversity, domin
```

See also

Soil texture classification

Description

<put algorithm description here>

Parameters

Sand [raster] Optional.

<put parameter description here>

Silt [raster] Optional.

<put parameter description here>

Clay [raster] Optional.

<put parameter description here>

Outputs

Soil Texture [raster] <put output description here>

Sum [raster] <put output description here>

Console usage

```
processing.runalg('saga:soiltextureclassification', sand, silt, clay, texture, sum)
```

See also

.

18.7.3 Grid calculus

Function

Description

<put algorithm description here>

Parameters

xmin [number] <put parameter description here>

Default: 0.0

xmax [number] <put parameter description here>

Default: 0.0

ymin [number] <put parameter description here>

Default: 0.0

ymax [number] <put parameter description here>

Default: 0.0

Formula [string] <put parameter description here>

Default: (not set)

Outputs

Function [raster] <put output description here>

Console usage

```
processing.runalg('saga:function', xmin, xmax, ymin, ymax, formul, result)
```

See also

Fuzzify

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

A [number] <put parameter description here>

Default: 0.0

B [number] <put parameter description here>

Default: 0.0

C [number] <put parameter description here>

Default: *0.0*

D [number] <put parameter description here>

Default: *0.0*

Membership Function Type [selection] <put parameter description here>

Options:

- 0 — [0] linear
- 1 — [1] sigmoidal
- 2 — [2] j-shaped

Default: *0*

Adjust to Grid [boolean] <put parameter description here>

Default: *True*

Outputs

Fuzzified Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:fuzzify', input, a, b, c, d, type, autofit, output)
```

See also

Fuzzy intersection (and)

Description

<put algorithm description here>

Parameters

Grids [multipleinput: rasters] <put parameter description here>

Operator Type [selection] <put parameter description here>

Options:

- 0 — [0] min(a, b) (non-interactive)
- 1 — [1] a * b
- 2 — [2] max(0, a + b - 1)

Default: *0*

Outputs

Intersection [raster] <put output description here>

Console usage

```
processing.runalg('saga:fuzzyintersectionand', grids, type, and)
```

See also

Fuzzy union (or)

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Operator Type [**selection**] <put parameter description here>

Options:

- 0 — [0] $\max(a, b)$ (non-interactive)
- 1 — [1] $a + b - a * b$
- 2 — [2] $\min(1, a + b)$

Default: 0

Outputs

Union [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:fuzzyunionor', grids, type, or)
```

See also

Geometric figures

Description

Draws simple geometric figures.

Parameters

Cell Count [**number**] Number of cells to use.

Default: 0

Cell Size [**number**] Size of the single cell.

Default: 0

Figure [selection] Type of the figure.

Options:

- 0 — [0] Cone (up)
- 1 — [1] Cone (down)
- 2 — [2] Plane

Default: 0

Direction of Plane [Degree] [number] Rotation factor in degrees.

Default: 0

Outputs

Result [raster] The resulting layer.

Console usage

```
processing.runalg('saga:geometricfigures', cell_count, cell_size, figure, plane, result)
```

See also

Gradient vector from cartesian to polar coordinates

Description

<put algorithm description here>

Parameters

X Component [raster] <put parameter description here>

Y Component [raster] <put parameter description here>

Polar Angle Units [selection] <put parameter description here>

Options:

- 0 — [0] radians
- 1 — [1] degree

Default: 0

Polar Coordinate System [selection] <put parameter description here>

Options:

- 0 — [0] mathematical
- 1 — [1] geographical
- 2 — [2] user defined

Default: 0

User defined Zero Direction [number] <put parameter description here>

Default: 0.0

User defined Orientation [selection] <put parameter description here>

Options:

- 0 — [0] clockwise
- 1 — [1] counterclockwise

Default: 0

Outputs

Direction [raster] <put output description here>

Length [raster] <put output description here>

Console usage

```
processing.runalg('saga:gradientvectorfromcartesiantopolarcoordinates', dx, dy, units, system, sy
```

See also

Gradient vector from polar to cartesian coordinates

Description

<put algorithm description here>

Parameters

Direction [raster] <put parameter description here>

Length [raster] <put parameter description here>

Polar Angle Units [selection] <put parameter description here>

Options:

- 0 — [0] radians
- 1 — [1] degree

Default: 0

Polar Coordinate System [selection] <put parameter description here>

Options:

- 0 — [0] mathematical
- 1 — [1] geographical
- 2 — [2] user defined

Default: 0

User defined Zero Direction [number] <put parameter description here>

Default: 0.0

User defined Orientation [selection] <put parameter description here>

Options:

- 0 — [0] clockwise

- 1 — [1] counterclockwise

Default: 0

Outputs

X Component [raster] <put output description here>

Y Component [raster] <put output description here>

Console usage

```
processing.runalg('saga:gradientvectorfrompolarcartesiancoordinates', dir, len, units, system, :
```

See also

Grid difference

Description

Creates a new grid layer as the result of the difference between two other grid layers.

Parameters

A [raster] First layer.

B [raster] Second layer.

Outputs

Difference (A - B) [raster] The resulting layer.

Console usage

```
processing.runalg('saga:griddifference', a, b, c)
```

See also

Grid division

Description

Creates a new grid layer as the result of the division between two other grid layers.

Parameters

Dividend [raster] First layer.

Divisor [raster] Second layer.

Outputs

Quotient [raster] The resulting layer.

Console usage

```
processing.runalg('saga:griddivision', a, b, c)
```

See also

Grid normalisation

Description

Normalises the grid values according to minimum and maximum values chosen.

Parameters

Grid [raster] Grid to normalize.

Target Range (min) [number] Minimum value.

Default: *0*

Target Range (max) [number] Maximum value.

Default: *1*

Outputs

Normalised Grid [raster] The resulting layer.

Console usage

```
processing.runalg('saga:gridnormalisation', input, range_min, range_max, output)
```

See also

Grids product

Description

<put algorithm description here>

Parameters

Grids [multipleinput: rasters] <put parameter description here>

Outputs

Product [raster] <put output description here>

Console usage

```
processing.runalg('saga:gridsproduct', grids, result)
```

See also

Grids sum

Description

Creates a new grid layer as the result of the sum of two or more grid layers.

Parameters

Grids [multipleinput: rasters] Grid layers to sum

Outputs

Sum [raster] The resulting layer.

Console usage

```
processing.runalg('saga:gridssum', grids, result)
```

See also

Grid standardisation

Description

Standardises the grid layer values.

Parameters

Grid [raster] Grid to process.

Stretch Factor [number] stretching factor.

Default: *1.0*

Outputs

Standardised Grid [raster] The resulting layer.

Console usage

```
processing.runalg('saga:gridstandardisation', input, stretch, output)
```

See also

Grid volume

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Count Only Above Base Level
- 1 — [1] Count Only Below Base Level
- 2 — [2] Subtract Volumes Below Base Level
- 3 — [3] Add Volumes Below Base Level

Default: 0

Base Level [number] <put parameter description here>

Default: 0.0

Outputs

Console usage

```
processing.runalg('saga:gridvolume', grid, method, level)
```

See also

Metric conversions

Description

Performs numerical conversions of the grid values.

Parameters

Grid [raster] Grid to process.

Conversion [selection] Conversion type.

Options:

- 0 — [0] radians to degree
- 1 — [1] degree to radians
- 2 — [2] Celsius to Fahrenheit
- 3 — [3] Fahrenheit to Celsius

Default: 0

Outputs

Converted Grid [raster] The resulting layer.

Console usage

```
processing.runalg('saga:metricconversions', grid, conversion, conv)
```

See also

Polynomial trend from grids

Description

<put algorithm description here>

Parameters

Dependent Variables [multipleinput: rasters] <put parameter description here>

Independent Variable (per Grid and Cell) [multipleinput: rasters] Optional.

<put parameter description here>

Independent Variable (per Grid) [fixedtable] <put parameter description here>

Type of Approximated Function [selection] <put parameter description here>

Options:

- 0 — [0] first order polynom (linear regression)
- 1 — [1] second order polynom
- 2 — [2] third order polynom
- 3 — [3] fourth order polynom
- 4 — [4] fifth order polynom

Default: 0

Outputs

Polynomial Coefficients [raster] <put output description here>

Coefficient of Determination [raster] <put output description here>

Console usage

```
processing.runalg('saga:polynomialtrendfromgrids', grids, y_grids, y_table, polynom, parms, quality)
```

See also

Random field

Description

Generates a random grid layer.

Parameters

Width (Cells) [number] Width of the layer in cells.

Default: *100*

Height (Cells) [number] Height of the layer in cells.

Default: *100*

Cellsize [number] Cell size to use.

Default: *100.0*

West [number] West coordinate of the bottom-left corner of the grid.

Default: *0.0*

South [number] South coordinate of the bottom-left corner of the grid.

Default: *0.0*

Method [selection] Statistical method used for the calculation.

Options:

- 0 — [0] Uniform
- 1 — [1] Gaussian

Default: *0*

Range Min [number] Minimum cell value to use.

Default: *0.0*

Range Max [number] Maximum cell value to use.

Default: *1.0*

Arithmetic Mean [number] Mean of all the cell values to use.

Default: *0.0*

Standard Deviation [number] Standard deviation of all the cell values to use.

Default: *1.0*

Outputs

Random Field [raster] The resulting layer.

Console usage

```
processing.runalg('saga:randomfield', nx, ny, cellsize, xmin, ymin, method, range_min, range_max,
```

See also

Random terrain generation

Description

<put algorithm description here>

Parameters

Radius (cells) [number] <put parameter description here>

Default: *10*

Iterations [number] <put parameter description here>

Default: *10*

Target Dimensions [selection] <put parameter description here>

Options:

- 0 — [0] User defined

Default: *0*

Grid Size [number] <put parameter description here>

Default: *1.0*

Cols [number] <put parameter description here>

Default: *100*

Rows [number] <put parameter description here>

Default: *100*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:randomterraingeneration', radius, iterations, target_type, user_cell_size)
```

See also

Raster calculator

Description

<put algorithm description here>

Parameters

Main input layer [raster] <put parameter description here>

Additional layers [multipleinput: rasters] Optional.

<put parameter description here>

Formula [string] <put parameter description here>

Default: *(not set)*

Outputs

Result [raster] <put output description here>

Console usage

```
processing.runalg('saga:rastercalculator', grids, xgrids, formula, result)
```

See also

.

18.7.4 Grid filter

Dtm filter (slope-based)

Description

<put algorithm description here>

Parameters

Grid to filter [raster] <put parameter description here>

Search Radius [number] <put parameter description here>

Default: 2

Approx. Terrain Slope [number] <put parameter description here>

Default: 30.0

Use Confidence Interval [boolean] <put parameter description here>

Default: *True*

Outputs

Bare Earth [raster] <put output description here>

Removed Objects [raster] <put output description here>

Console usage

```
processing.runalg('saga:dtmfilterslopebased', input, radius, terrainslope, stddev, ground, nongro
```

See also

Filter clumps

Description

<put algorithm description here>

Parameters

Input Grid [raster] <put parameter description here>

Min. Size [number] <put parameter description here>

Default: 10

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:filterclumps', grid, threshold, output)
```

See also

Gaussian filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Standard Deviation [number] <put parameter description here>

Default: 1

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Search Radius [number] <put parameter description here>

Default: 3

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:gaussianfilter', input, sigma, mode, radius, result)
```

See also

Laplacian filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] standard kernel 1
- 1 — [1] standard kernel 2
- 2 — [2] Standard kernel 3
- 3 — [3] user defined kernel

Default: 0

Standard Deviation (Percent of Radius) [number] <put parameter description here>

Default: 0

Radius [number] <put parameter description here>

Default: 1

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] square
- 1 — [1] circle

Default: 0

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:laplacianfilter', input, method, sigma, radius, mode, result)
```

See also

Majority filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Radius [number] <put parameter description here>

Default: 1

Threshold [Percent] [number] <put parameter description here>

Default: 0

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:majorityfilter', input, mode, radius, threshold, result)
```

See also

Morphological filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Radius [number] <put parameter description here>

Default: *1*

Method [selection] <put parameter description here>

Options:

- 0 — [0] Dilation
- 1 — [1] Erosion
- 2 — [2] Opening
- 3 — [3] Closing

Default: *0*

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:morphologicalfilter', input, mode, radius, method, result)
```

See also

Multi direction lee filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Estimated Noise (absolute) [number] <put parameter description here>

Default: *1.0*

Estimated Noise (relative) [number] <put parameter description here>

Default: *1.0*

Weighted [boolean] <put parameter description here>

Default: *True*

Method [selection] <put parameter description here>

Options:

- 0 — [0] noise variance given as absolute value
- 1 — [1] noise variance given relative to mean standard deviation
- 2 — [2] original calculation (Ringeler)

Default: *0*

Outputs

Filtered Grid [raster] <put output description here>

Minimum Standard Deviation [raster] <put output description here>

Direction of Minimum Standard Deviation [raster] <put output description here>

Console usage

```
processing.runalg('saga:multidirectionleefilter', input, noise_abs, noise_rel, weighted, method, ...)
```

See also

Rank filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Radius [number] <put parameter description here>

Default: 1

Rank [Percent] [number] <put parameter description here>

Default: 50

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:rankfilter', input, mode, radius, rank, result)
```

See also

Simple filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Filter [selection] <put parameter description here>

Options:

- 0 — [0] Smooth
- 1 — [1] Sharpen
- 2 — [2] Edge

Default: 0

Radius [number] <put parameter description here>

Default: 2

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:simplefilter', input, mode, method, radius, result)
```

See also

User defined filter

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Filter Matrix [table] Optional.

<put parameter description here>

Default Filter Matrix (3x3) [fixedtable] <put parameter description here>

Outputs

Filtered Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:userdefinedfilter', input, filter, filter_3x3, result)
```

See also

.

18.7.5 Grid gridding**Inverse distance weighted****Description**

Inverse distance grid interpolation from irregular distributed points.

Parameters

Points [**vector: point**] <put parameter description here>

Attribute [**tablefield: any**] <put parameter description here>

Target Grid [**selection**] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Distance Weighting [**selection**] <put parameter description here>

Options:

- 0 — [0] inverse distance to a power
- 1 — [1] linearly decreasing within search radius
- 2 — [2] exponential weighting scheme
- 3 — [3] gaussian weighting scheme

Default: 0

Inverse Distance Power [**number**] <put parameter description here>

Default: 2

Exponential and Gaussian Weighting Bandwidth [**number**] <put parameter description here>

Default: 1

Search Range [**selection**] <put parameter description here>

Options:

- 0 — [0] search radius (local)
- 1 — [1] no search radius (global)

Default: 0

Search Radius [**number**] <put parameter description here>

Default: 100.0

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Number of Points [selection] <put parameter description here>

Options:

- 0 — [0] maximum number of points
- 1 — [1] all points

Default: 0

Maximum Number of Points [number] <put parameter description here>

Default: 10

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:inversedistanceweighted', shapes, field, target, weighting, power, bandwi
```

See also

Kernel density estimation

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Weight [tablefield: any] <put parameter description here>

Radius [number] <put parameter description here>

Default: 10

Kernel [selection] <put parameter description here>

Options:

- 0 — [0] quartic kernel
- 1 — [1] gaussian kernel

Default: 0

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:kerneldensityestimation', points, population, radius, kernel, target, outp
```

See also

Modifed quadratic shepard

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Quadratic Neighbors [number] <put parameter description here>

Default: 13

Weighting Neighbors [number] <put parameter description here>

Default: 19

Left [number] <put parameter description here>

Default: 0.0

Right [number] <put parameter description here>

Default: 0.0

Bottom [number] <put parameter description here>

Default: *0.0*

Top [number] <put parameter description here>

Default: *0.0*

Cellsize [number] <put parameter description here>

Default: *100.0*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:modifiedquadraticshepard', shapes, field, target, quadratic_neighbors, wei
```

See also

Natural neighbour

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: *0*

Sibson [boolean] <put parameter description here>

Default: *True*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Cellsize [number] <put parameter description here>

Default: *100.0*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:naturalneighbour', shapes, field, target, sibson, output_extent, user_size)
```

See also

Nearest neighbour

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:nearestneighbour', shapes, field, target, output_extent, user_size, user_size)
```

See also

Shapes to grid

Description

<put algorithm description here>

Parameters

Shapes [vector: any] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Method for Multiple Values [selection] <put parameter description here>

Options:

- 0 — [0] first
- 1 — [1] last
- 2 — [2] minimum
- 3 — [3] maximum
- 4 — [4] mean

Default: 0

Method for Lines [selection] <put parameter description here>

Options:

- 0 — [0] thin
- 1 — [1] thick

Default: 0

Preferred Target Grid Type [selection] <put parameter description here>

Options:

- 0 — [0] Integer (1 byte)
- 1 — [1] Integer (2 byte)
- 2 — [2] Integer (4 byte)
- 3 — [3] Floating Point (4 byte)
- 4 — [4] Floating Point (8 byte)

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:shapestogrid', input, field, multiple, line_type, grid_type, output_extent)
```

See also

Triangulation

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:triangulation', shapes, field, target, output_extent, user_size, user_grid)
```

See also

18.7.6 Grid spline

B-spline approximation

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Resolution [number] <put parameter description here>

Default: 1.0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:bsplineapproximation', shapes, field, target, level, output_extent, user_
```

See also

Cubic spline approximation

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Minimal Number of Points [number] <put parameter description here>

Default: 3

Maximal Number of Points [number] <put parameter description here>

Default: 20

Points per Square [number] <put parameter description here>

Default: 5

Tolerance [number] <put parameter description here>

Default: 140.0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:cubicsplineapproximation', shapes, field, target, npmin, npmax, nppc, k, c
```

See also

Multilevel b-spline interpolation (from grid)

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Method [selection] <put parameter description here>

Options:

- 0 — [0] without B-spline refinement
- 1 — [1] with B-spline refinement

Default: 0

Threshold Error [number] <put parameter description here>

Default: 0.0001

Maximum Level [number] <put parameter description here>

Default: 11.0

Data Type [selection] <put parameter description here>

Options:

- 0 — [0] same as input grid
- 1 — [1] floating point

Default: 0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:multilevelbsplineinterpolationfromgrid', gridpoints, target, method, epsi
```

See also

Multilevel b-spline interpolation

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Method [selection] <put parameter description here>

Options:

- 0 — [0] without B-spline refinement
- 1 — [1] with B-spline refinement

Default: 0

Threshold Error [number] <put parameter description here>

Default: 0.0001

Maximum Level [number] <put parameter description here>

Default: 11.0

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: *100.0*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:multilevelbsplineinterpolation', shapes, field, target, method, epsilon, .
```

See also

Thin plate spline (global)

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: *0*

Regularisation [number] <put parameter description here>

Default: *0.0*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Cellsize [number] <put parameter description here>

Default: *100.0*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:thinplatesplineglobal', shapes, field, target, regul, output_extent, user.
```

See also

Thin plate spline (local)

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Regularisation [number] <put parameter description here>

Default: 0.0001

Search Radius [number] <put parameter description here>

Default: 100.0

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] all directions
- 1 — [1] quadrants

Default: 0

Points Selection [selection] <put parameter description here>

Options:

- 0 — [0] all points in search radius
- 1 — [1] maximum number of points

Default: 0

Maximum Number of Points [number] <put parameter description here>

Default: 10

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:thinplatesplinelocal', shapes, field, target, regul, radius, mode, select,
```

See also

Thin plate spline (tin)

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Regularisation [number] <put parameter description here>

Default: 0.0

Neighbourhood [selection] <put parameter description here>

Options:

- 0 — [0] immediate
- 1 — [1] level 1
- 2 — [2] level 2

Default: 0

Add Frame [boolean] <put parameter description here>

Default: True

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Cellsize [number] <put parameter description here>

Default: 100.0

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:thinplatesplinetin', shapes, field, target, regul, level, frame, output_e
```

See also

.

18.7.7 Grid tools

Aggregate

Description

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Aggregation Size [**number**] <put parameter description here>

Default: 3

Method [**selection**] <put parameter description here>

Options:

- 0 — [0] Sum
- 1 — [1] Min
- 2 — [2] Max

Default: 0

Outputs

Console usage

```
processing.runalg('saga:aggregate', input, size, method)
```

See also

Change grid values

Description

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Replace Condition [**selection**] <put parameter description here>

Options:

- 0 — [0] Grid value equals low value
- 1 — [1] Low value < grid value < high value

- 2 — [2] Low value \leq grid value $<$ high value

Default: 0

Lookup Table [fixedtable] <put parameter description here>

Outputs

Changed Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:changegridvalues', grid_in, method, lookup, grid_out)
```

See also

Close gaps

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Mask [raster] Optional.

<put parameter description here>

Tension Threshold [number] <put parameter description here>

Default: 0.1

Outputs

Changed Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:closegaps', input, mask, threshold, result)
```

See also

Close gaps with spline

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Mask [raster] Optional.

<put parameter description here>

Only Process Gaps with Less Cells [number] <put parameter description here>

Default: *0*

Maximum Points [number] <put parameter description here>

Default: *1000*

Number of Points for Local Interpolation [number] <put parameter description here>

Default: *10*

Extended Neighbourhood [boolean] <put parameter description here>

Default: *True*

Neighbourhood [selection] <put parameter description here>

Options:

- 0 — [0] Neumann
- 1 — [1] Moore

Default: *0*

Radius (Cells) [number] <put parameter description here>

Default: *0*

Relaxation [number] <put parameter description here>

Default: *0.0*

Outputs

Closed Gaps Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:closegapswithspline', grid, mask, maxgapcells, maxpoints, localpoints, ex
```

See also

Close one cell gaps

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Outputs

Changed Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:closeonecellgaps', input, result)
```

See also

Convert data storage type

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Data storage type [selection] <put parameter description here>

Options:

- 0 — [0] bit
- 1 — [1] unsigned 1 byte integer
- 2 — [2] signed 1 byte integer
- 3 — [3] unsigned 2 byte integer
- 4 — [4] signed 2 byte integer
- 5 — [5] unsigned 4 byte integer
- 6 — [6] signed 4 byte integer
- 7 — [7] 4 byte floating point number
- 8 — [8] 8 byte floating point number

Default: 0

Outputs

Converted Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:convertdatastoragetype', input, type, output)
```

See also

Crop to data

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Outputs

Cropped layer [raster] <put output description here>

Console usage

```
processing.runalg('saga:croptodata', input, output)
```

See also

Grid buffer

Description

<put algorithm description here>

Parameters

Features Grid [raster] <put parameter description here>

Distance [number] <put parameter description here>

Default: *1000*

Buffer Distance [selection] <put parameter description here>

Options:

- 0 — [0] Fixed
- 1 — [1] Cell value

Default: *0*

Outputs

Buffer Grid [raster] <put output description here>

Console usage


```
processing.runalg('saga:gridbuffer', features, dist, buffertype, buffer)
```

See also

Grid masking

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Mask [raster] <put parameter description here>

Outputs

Masked Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:gridmasking', grid, mask, masked)
```

See also

Grid orientation

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Copy
- 1 — [1] Flip
- 2 — [2] Mirror
- 3 — [3] Invert

Default: 0

Outputs

Changed Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:gridorientation', input, method, result)
```

See also

Grid proximity buffer

Description

<put algorithm description here>

Parameters

Source Grid [**raster**] <put parameter description here>

Buffer distance [**number**] <put parameter description here>

Default: *500.0*

Equidistance [**number**] <put parameter description here>

Default: *100.0*

Outputs

Distance Grid [**raster**] <put output description here>

Allocation Grid [**raster**] <put output description here>

Buffer Grid [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:gridproximitybuffer', source, dist, ival, distance, alloc, buffer)
```

See also

Grid shrink/expand

Description

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Operation [**selection**] <put parameter description here>

Options:

- 0 — [0] Shrink
- 1 — [1] Expand

Default: 0

Search Mode [selection] <put parameter description here>

Options:

- 0 — [0] Square
- 1 — [1] Circle

Default: 0

Radius [number] <put parameter description here>

Default: 1

Method [selection] <put parameter description here>

Options:

- 0 — [0] min
- 1 — [1] max
- 2 — [2] mean
- 3 — [3] majority

Default: 0

Outputs

Result Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:gridshrinkexpand', input, operation, mode, radius, method_expand, result)
```

See also

Invert data/no-data

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Outputs

Result [raster] <put output description here>

Console usage

```
processing.runalg('saga:invertdatanodata', input, output)
```

See also

Merge raster layers

Description

<put algorithm description here>

Parameters

Grids to Merge [**multipleinput: rasters**] <put parameter description here>

Preferred data storage type [**selection**] <put parameter description here>

Options:

- 0 — [0] 1 bit
- 1 — [1] 1 byte unsigned integer
- 2 — [2] 1 byte signed integer
- 3 — [3] 2 byte unsigned integer
- 4 — [4] 2 byte signed integer
- 5 — [5] 4 byte unsigned integer
- 6 — [6] 4 byte signed integer
- 7 — [7] 4 byte floating point
- 8 — [8] 8 byte floating point

Default: 0

Interpolation [**selection**] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Overlapping Cells [**selection**] <put parameter description here>

Options:

- 0 — [0] mean value
- 1 — [1] first value in order of grid list

Default: 0

Outputs

Merged Grid [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:mergerasterlayers', grids, type, interpol, overlap, merged)
```

See also

Patching

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Patch Grid [raster] <put parameter description here>

Interpolation Method [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Outputs

Completed Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:patching', original, additional, interpolation, completed)
```

See also

Proximity grid

Description

<put algorithm description here>

Parameters

Features [raster] <put parameter description here>

Outputs

Distance [raster] <put output description here>

Direction [raster] <put output description here>

Allocation [raster] <put output description here>

Console usage

```
processing.runalg('saga:proximitygrid', features, distance, direction, allocation)
```

See also

Reclassify grid values

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] single
- 1 — [1] range
- 2 — [2] simple table

Default: 0

old value (for single value change) [number] <put parameter description here>

Default: 0.0

new value (for single value change) [number] <put parameter description here>

Default: 1.0

operator (for single value change) [selection] <put parameter description here>

Options:

- 0 — [0] =
- 1 — [1] <
- 2 — [2] <=
- 3 — [3] >=
- 4 — [4] >

Default: 0

minimum value (for range) [number] <put parameter description here>

Default: 0.0

maximum value (for range) [number] <put parameter description here>

Default: *1.0*

new value (for range) [number] <put parameter description here>

Default: *2.0*

operator (for range) [selection] <put parameter description here>

Options:

- 0 — [0] <=
- 1 — [1] <

Default: *0*

Lookup Table [fixedtable] <put parameter description here>

operator (for table) [selection] <put parameter description here>

Options:

- 0 — [0] min <= value < max
- 1 — [1] min <= value <= max
- 2 — [2] min < value <= max
- 3 — [3] min < value < max

Default: *0*

replace no data values [boolean] <put parameter description here>

Default: *True*

new value for no data values [number] <put parameter description here>

Default: *0.0*

replace other values [boolean] <put parameter description here>

Default: *True*

new value for other values [number] <put parameter description here>

Default: *0.0*

Outputs

Reclassified Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:reclassifygridvalues', input, method, old, new, soperator, min, max, rnew,
```

See also

Resampling

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Preserve Data Type [boolean] <put parameter description here>

Default: *True*

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: *0*

Interpolation Method (Scale Up) [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation
- 5 — [5] Mean Value
- 6 — [6] Mean Value (cell area weighted)
- 7 — [7] Minimum Value
- 8 — [8] Maximum Value
- 9 — [9] Majority

Default: *0*

Interpolation Method (Scale Down) [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: *0*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Cellsize [number] <put parameter description here>

Default: *100.0*

Outputs

Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:resampling', input, keep_type, target, scale_up_method, scale_down_method)
```

See also

Sort grid

Description

<put algorithm description here>

Parameters

Input Grid [raster] <put parameter description here>

Down sort [boolean] <put parameter description here>

Default: *True*

Outputs

Sorted Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:sortgrid', grid, down, output)
```

See also

Split RGB bands

Description

<put algorithm description here>

Parameters

Input layer [raster] <put parameter description here>

Outputs

Output R band layer [raster] <put output description here>

Output G band layer [raster] <put output description here>

Output B band layer [raster] <put output description here>

Console usage

```
processing.runalg('saga:splitrgbbands', input, r, g, b)
```

See also

Threshold buffer

Description

<put algorithm description here>

Parameters

Features Grid [raster] <put parameter description here>

Value Grid [raster] <put parameter description here>

Threshold Grid [raster] Optional.

<put parameter description here>

Threshold [number] <put parameter description here>

Default: *0.0*

Threshold Type [selection] <put parameter description here>

Options:

- 0 — [0] Absolute
- 1 — [1] Relative from cell value

Default: *0*

Outputs

Buffer Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:thresholdbuffer', features, value, thresholdgrid, threshold, thresholdtype)
```

See also

.

18.7.8 Grid visualization

Histogram surface

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] rows
- 1 — [1] columns
- 2 — [2] circle

Default: 0

Outputs

Histogram [raster] <put output description here>

Console usage

```
processing.runalg('saga:histogramsurface', grid, method, hist)
```

See also

Rgb composite

Description

<put algorithm description here>

Parameters

R [raster] <put parameter description here>

G [raster] <put parameter description here>

B [raster] <put parameter description here>

Method for R value [selection] <put parameter description here>

Options:

- 0 — 0 - 255
- 1 — Rescale to 0 - 255
- 2 — User defined rescale
- 3 — Percentiles
- 4 — Percentage of standard deviation

Default: 0

Method for G value [selection] <put parameter description here>

Options:

- 0 — 0 - 255
- 1 — Rescale to 0 - 255

- 2 — User defined rescale
- 3 — Percentiles
- 4 — Percentage of standard deviation

Default: 0

Method for B value [selection] <put parameter description here>

Options:

- 0 — 0 - 255
- 1 — Rescale to 0 - 255
- 2 — User defined rescale
- 3 — Percentiles
- 4 — Percentage of standard deviation

Default: 0

Rescale Range for RED min [number] <put parameter description here>

Default: 0

Rescale Range for RED max [number] <put parameter description here>

Default: 255

Percentiles Range for RED max [number] <put parameter description here>

Default: 1

Percentiles Range for RED max [number] <put parameter description here>

Default: 99

Percentage of standard deviation for RED [number] <put parameter description here>

Default: 150.0

Rescale Range for GREEN min [number] <put parameter description here>

Default: 0

Rescale Range for GREEN max [number] <put parameter description here>

Default: 255

Percentiles Range for GREEN max [number] <put parameter description here>

Default: 1

Percentiles Range for GREEN max [number] <put parameter description here>

Default: 99

Percentage of standard deviation for GREEN [number] <put parameter description here>

Default: 150.0

Rescale Range for BLUE min [number] <put parameter description here>

Default: 0

Rescale Range for BLUE max [number] <put parameter description here>

Default: 255

Percentiles Range for BLUE max [number] <put parameter description here>

Default: 1

Percentiles Range for BLUE max [number] <put parameter description here>

Default: *99*

Percentage of standard deviation for BLUE [number] <put parameter description here>

Default: *150.0*

Outputs

Output RGB [raster] <put output description here>

Console usage

```
processing.runalg('saga:rgbcomposite', grid_r, grid_g, grid_b, r_method, g_method, b_method, r_ra
```

See also

.

18.7.9 Imagery classification

Change detection

Description

<put algorithm description here>

Parameters

Initial State [raster] <put parameter description here>

Look-up Table [table] Optional.

<put parameter description here>

Value [tablefield: any] <put parameter description here>

Value (Maximum) [tablefield: any] <put parameter description here>

Name [tablefield: any] <put parameter description here>

Final State [raster] <put parameter description here>

Look-up Table [table] Optional.

<put parameter description here>

Value [tablefield: any] <put parameter description here>

Value (Maximum) [tablefield: any] <put parameter description here>

Name [tablefield: any] <put parameter description here>

Report Unchanged Classes [boolean] <put parameter description here>

Default: *True*

Output as... [selection] <put parameter description here>

Options:

- 0 — [0] cells
- 1 — [1] percent
- 2 — [2] area

Default: 0

Outputs

Changes [raster] <put output description here>

Changes [table] <put output description here>

Console usage

```
processing.runalg('saga:changedetection', initial, ini_lut, ini_lut_min, ini_lut_max, ini_lut_name,
```

See also

Cluster analysis for grids

Description

<put algorithm description here>

Parameters

Grids [multipleinput: rasters] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Iterative Minimum Distance (Forgy 1965)
- 1 — [1] Hill-Climbing (Rubin 1967)
- 2 — [2] Combined Minimum Distance / Hillclimbing

Default: 0

Clusters [number] <put parameter description here>

Default: 5

Normalise [boolean] <put parameter description here>

Default: *True*

Old Version [boolean] <put parameter description here>

Default: *True*

Outputs

Clusters [raster] <put output description here>

Statistics [table] <put output description here>

Console usage

```
processing.runalg('saga:clusteranalysisforgrids', grids, method, ncluster, normalise, oldversion,
```

See also

Supervised classification

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Training Areas [**vector: polygon**] <put parameter description here>

Class Identifier [**tablefield: any**] <put parameter description here>

Method [**selection**] <put parameter description here>

Options:

- 0 — [0] Binary Encoding
- 1 — [1] Parallelepiped
- 2 — [2] Minimum Distance
- 3 — [3] Mahalanobis Distance
- 4 — [4] Maximum Likelihood
- 5 — [5] Spectral Angle Mapping
- 6 — [6] Winner Takes All

Default: *0*

Normalise [**boolean**] <put parameter description here>

Default: *True*

Distance Threshold [**number**] <put parameter description here>

Default: *0.0*

Probability Threshold (Percent) [**number**] <put parameter description here>

Default: *0.0*

Probability Reference [**selection**] <put parameter description here>

Options:

- 0 — [0] absolute
- 1 — [1] relative

Default: *0*

Spectral Angle Threshold (Degree) [**number**] <put parameter description here>

Default: *0.0*

Outputs

Class Information [table] <put output description here>

Classification [raster] <put output description here>

Quality [raster] <put output description here>

Console usage

```
processing.runalg('saga:supervisedclassification', grids, roi, roi_id, method, normalise, thresho
```

See also

.

18.7.10 Imagery RGA

Fast region growing algorithm

Description

<put algorithm description here>

Parameters

Input Grids [multipleinput: rasters] <put parameter description here>

Seeds Grid [raster] <put parameter description here>

Smooth Rep [raster] Optional.

<put parameter description here>

Outputs

Segmente [raster] <put output description here>

Mean [raster] <put output description here>

Console usage

```
processing.runalg('saga:fastregiongrowingalgorithm', input, start, rep, result, mean)
```

See also

.

18.7.11 Imagery segmentation

Grid skeletonization

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Standard
- 1 — [1] Hilditch's Algorithm
- 2 — [2] Channel Skeleton

Default: 0

Initialisation [selection] <put parameter description here>

Options:

- 0 — [0] Less than
- 1 — [1] Greater than

Default: 0

Threshold (Init.) [number] <put parameter description here>

Default: 0.0

Convergence [number] <put parameter description here>

Default: 3.0

Outputs

Skeleton [raster] <put output description here>

Skeleton [vector] <put output description here>

Console usage

```
processing.runalg('saga:gridskeletonization', input, method, init_method, init_threshold, converg
```

See also

Seed generation

Description

<put algorithm description here>

Parameters

Features [**multipleinput: rasters**] <put parameter description here>

Bandwidth (Cells) [**number**] <put parameter description here>

Default: 2

Type of Surface [**selection**] <put parameter description here>

Options:

- 0 — [0] smoothed surface
- 1 — [1] variance (a)
- 2 — [2] variance (b)

Default: 0

Extraction of... [**selection**] <put parameter description here>

Options:

- 0 — [0] minima
- 1 — [1] maxima
- 2 — [2] minima and maxima

Default: 0

Feature Aggregation [**selection**] <put parameter description here>

Options:

- 0 — [0] additive
- 1 — [1] multiplicative

Default: 0

Normalized [**boolean**] <put parameter description here>

Default: *True*

Outputs

Surface [**raster**] <put output description here>

Seeds Grid [**raster**] <put output description here>

Seeds [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:seedgeneration', grids, factor, type_surface, type_seeds, type_merge, norm
```

See also

Simple region growing

Description

<put algorithm description here>

Parameters

Seeds [raster] <put parameter description here>

Features [multipleinput: rasters] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] feature space and position
- 1 — [1] feature space

Default: 0

Neighbourhood [selection] <put parameter description here>

Options:

- 0 — [0] 4 (von Neumann)
- 1 — [1] 8 (Moore)

Default: 0

Variance in Feature Space [number] <put parameter description here>

Default: 1.0

Variance in Position Space [number] <put parameter description here>

Default: 1.0

Threshold - Similarity [number] <put parameter description here>

Default: 0.0

Refresh [boolean] <put parameter description here>

Default: *True*

Leaf Size (for Speed Optimisation) [number] <put parameter description here>

Default: 256

Outputs

Segments [raster] <put output description here>

Similarity [raster] <put output description here>

Seeds [table] <put output description here>

Console usage

```
processing.runalg('saga:simpleregiongrowing', seeds, features, method, neighbour, sig_1, sig_2, t
```

See also

Watershed segmentation

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Output [selection] <put parameter description here>

Options:

- 0 — [0] Seed Value
- 1 — [1] Segment ID

Default: 0

Method [selection] <put parameter description here>

Options:

- 0 — [0] Minima
- 1 — [1] Maxima

Default: 0

Join Segments based on Threshold Value [selection] <put parameter description here>

Options:

- 0 — [0] do not join
- 1 — [1] seed to saddle difference
- 2 — [2] seeds difference

Default: 0

Threshold [number] <put parameter description here>

Default: 0

Allow Edge Pixels to be Seeds [boolean] <put parameter description here>

Default: *True*

Borders [boolean] <put parameter description here>

Default: *True*

Outputs

Segments [raster] <put output description here>

Seed Points [vector] <put output description here>

Borders [raster] <put output description here>

Console usage

```
processing.runalg('saga:watershedsegmentation', grid, output, down, join, threshold, edge, bborder)
```

See also

.

18.7.12 Imagery tools

Vegetation index[distance based]

Description

<put algorithm description here>

Parameters

Near Infrared Band [raster] <put parameter description here>

Red Band [raster] <put parameter description here>

Slope of the soil line [number] <put parameter description here>

Default: *0.0*

Intercept of the soil line [number] <put parameter description here>

Default: *0.0*

Outputs

PVI (Richardson and Wiegand) [raster] <put output description here>

PVI (Perry & Lautenschlager) [raster] <put output description here>

PVI (Walther & Shabaani) [raster] <put output description here>

PVI (Qi, et al) [raster] <put output description here>

Console usage

```
processing.runalg('saga:vegetationindexdistancebased', nir, red, slope, intercept, pvi, pvi1, pvi2)
```

See also

Vegetation index[slope based]

Description

<put algorithm description here>

Parameters

Near Infrared Band [raster] <put parameter description here>

Red Band [raster] <put parameter description here>

Outputs

Normalized Difference Vegetation Index [raster] <put output description here>

Ratio Vegetation Index [raster] <put output description here>

Transformed Vegetation Index [raster] <put output description here>

Corrected Transformed Vegetation Index [raster] <put output description here>

Thiam's Transformed Vegetation Index [raster] <put output description here>

Normalized Ratio Vegetation Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:vegetationindexslopebased', nir, red, ndvi, ratio, tvi, ctvi, ttvi, nratio)
```

See also

.

18.7.13 Kriging

Ordinary kriging (global)

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Create Variance Grid [boolean] <put parameter description here>

Default: *True*

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: *0*

Variogram Model [selection] <put parameter description here>

Options:

- 0 — [0] Spherical Model
- 1 — [1] Exponential Model
- 2 — [2] Gaussian Model
- 3 — [3] Linear Regression
- 4 — [4] Exponential Regression
- 5 — [5] Power Function Regression

Default: *0*

Block Kriging [boolean] <put parameter description here>

Default: *True*

Block Size [number] <put parameter description here>

Default: *100*

Logarithmic Transformation [boolean] <put parameter description here>

Default: *True*

Nugget [number] <put parameter description here>

Default: *0.0*

Sill [number] <put parameter description here>

Default: *0.0*

Range [number] <put parameter description here>

Default: *0.0*

Linear Regression [number] <put parameter description here>

Default: *1.0*

Exponential Regression [number] <put parameter description here>

Default: *0.1*

Power Function - A [number] <put parameter description here>

Default: *1.0*

Power Function - B [number] <put parameter description here>

Default: *0.5*

Grid Size [number] <put parameter description here>

Default: *1.0*

Fit Extent [boolean] <put parameter description here>

Default: *True*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Outputs

Grid [raster] <put output description here>

Variance [raster] <put output description here>

Console usage

```
processing.runalg('saga:ordinarykrigingglobal', shapes, field, bvariance, target, model, block, d
```

See also

Ordinary kriging

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Create Variance Grid [boolean] <put parameter description here>

Default: *True*

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: *0*

Variogram Model [selection] <put parameter description here>

Options:

- 0 — [0] Spherical Model
- 1 — [1] Exponential Model
- 2 — [2] Gaussian Model
- 3 — [3] Linear Regression
- 4 — [4] Exponential Regression
- 5 — [5] Power Function Regression

Default: *0*

Block Kriging [boolean] <put parameter description here>

Default: *True*

Block Size [number] <put parameter description here>

Default: *100*

Logarithmic Transformation [boolean] <put parameter description here>

Default: *True*

Nugget [number] <put parameter description here>

Default: *0.0*

Sill [number] <put parameter description here>

Default: *10.0*

Range [number] <put parameter description here>

Default: *100.0*

Linear Regression [number] <put parameter description here>

Default: *1.0*

Exponential Regression [number] <put parameter description here>

Default: *0.1*

Power Function - A [number] <put parameter description here>

Default: *1*

Power Function - B [number] <put parameter description here>

Default: *0.5*

Maximum Search Radius (map units) [number] <put parameter description here>

Default: *1000.0*

Min.Number of m_Points [number] <put parameter description here>

Default: *4*

Max. Number of m_Points [number] <put parameter description here>

Default: *20*

Grid Size [number] <put parameter description here>

Default: *1.0*

Fit Extent [boolean] <put parameter description here>

Default: *True*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Outputs

Grid [raster] <put output description here>

Variance [raster] <put output description here>

Console usage

```
processing.runalg('saga:ordinarykriging', shapes, field, bvariance, target, model, block, dblock,
```

See also

Universal kriging (global)

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Create Variance Grid [boolean] <put parameter description here>

Default: *True*

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Variogram Model [selection] <put parameter description here>

Options:

- 0 — [0] Spherical Model
- 1 — [1] Exponential Model
- 2 — [2] Gaussian Model
- 3 — [3] Linear Regression
- 4 — [4] Exponential Regression
- 5 — [5] Power Function Regression

Default: 0

Block Kriging [boolean] <put parameter description here>

Default: *True*

Block Size [number] <put parameter description here>

Default: 100

Logarithmic Transformation [boolean] <put parameter description here>

Default: *True*

Nugget [number] <put parameter description here>

Default: 0.0

Sill [number] <put parameter description here>

Default: 0.0

Range [number] <put parameter description here>

Default: 0.0

Linear Regression [number] <put parameter description here>

Default: 1

Exponential Regression [number] <put parameter description here>

Default: 0.5

Power Function - A [number] <put parameter description here>

Default: 1.0

Power Function - B [number] <put parameter description here>

Default: 0.1

Grids [multipleinput: rasters] <put parameter description here>

Grid Interpolation [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation

- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Grid Size [number] <put parameter description here>

Default: 1.0

Fit Extent [boolean] <put parameter description here>

Default: True

Output extent [extent] <put parameter description here>

Default: 0,1,0,1

Outputs

Grid [raster] <put output description here>

Variance [raster] <put output description here>

Console usage

```
processing.runalg('saga:universalkrigingglobal', shapes, field, bvariance, target, model, block, ...)
```

See also

Universal kriging

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Create Variance Grid [boolean] <put parameter description here>

Default: True

Target Grid [selection] <put parameter description here>

Options:

- 0 — [0] user defined

Default: 0

Variogram Model [selection] <put parameter description here>

Options:

- 0 — [0] Spherical Model
- 1 — [1] Exponential Model
- 2 — [2] Gaussian Model
- 3 — [3] Linear Regression

- 4 — [4] Exponential Regression
- 5 — [5] Power Function Regression

Default: 0

Block Kriging [boolean] <put parameter description here>

Default: *True*

Block Size [number] <put parameter description here>

Default: 100

Logarithmic Transformation [boolean] <put parameter description here>

Default: *True*

Nugget [number] <put parameter description here>

Default: 0.0

Sill [number] <put parameter description here>

Default: 0.0

Range [number] <put parameter description here>

Default: 0.0

Linear Regression [number] <put parameter description here>

Default: 1.0

Exponential Regression [number] <put parameter description here>

Default: 0.1

Power Function - A [number] <put parameter description here>

Default: 1

Power Function - B [number] <put parameter description here>

Default: 0.5

Grids [multipleinput: rasters] <put parameter description here>

Grid Interpolation [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Min.Number of m_Points [number] <put parameter description here>

Default: 4

Max. Number of m_Points [number] <put parameter description here>

Default: 20

Maximum Search Radius (map units) [number] <put parameter description here>

Default: 1000.0

Grid Size [number] <put parameter description here>

Default: *1.0*

Fit Extent [boolean] <put parameter description here>

Default: *True*

Output extent [extent] <put parameter description here>

Default: *0,1,0,1*

Outputs

Grid [raster] <put output description here>

Variance [raster] <put output description here>

Console usage

```
processing.runalg('saga:universalkriging', shapes, field, bvariance, target, model, block, dblock,
```

See also

.

18.7.14 Shapes grid

Add grid values to points

Description

Creates a new vector layer as a result of the union of a points layer with the interpolated value of one or more base background grid layer(s). This way, the new layer created will have a new column in the attribute table that reflects the interpolated value of the background grid.

Parameters

Points [vector: point] Input layer.

Grids [multipleinput: rasters] Background grid layer(s)

Interpolation [selection] interpolation method to use.

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: *0*

Outputs

Result [vector] The resulting layer.

Console usage

```
processing.runalg('saga:addgridvaluestopoints', shapes, grids, interpol, result)
```

See also

Add grid values to shapes

Description

<put algorithm description here>

Parameters

Shapes [vector: any] <put parameter description here>

Grids [multipleinput: rasters] <put parameter description here>

Interpolation [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbor
- 1 — [1] Bilinear Interpolation
- 2 — [2] Inverse Distance Interpolation
- 3 — [3] Bicubic Spline Interpolation
- 4 — [4] B-Spline Interpolation

Default: 0

Outputs

Result [vector] <put output description here>

Console usage

```
processing.runalg('saga:addgridvaluestoshapes', shapes, grids, interpol, result)
```

See also

Clip grid with polygon

Description

<put algorithm description here>

Parameters

Input [raster] <put parameter description here>

Polygons [vector: polygon] <put parameter description here>

Outputs

Output [raster] <put output description here>

Console usage

```
processing.runalg('saga:clipgridwithpolygon', input, polygons, output)
```

See also

Contour lines from grid

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Minimum Contour Value [number] <put parameter description here>

Default: *0.0*

Maximum Contour Value [number] <put parameter description here>

Default: *10000.0*

Equidistance [number] <put parameter description here>

Default: *100.0*

Outputs

Contour Lines [vector] <put output description here>

Console usage

```
processing.runalg('saga:contourlinesfromgrid', input, zmin, zmax, zstep, contour)
```

See also

Gradient vectors from directional components

Description

<put algorithm description here>

Parameters

X Component [raster] <put parameter description here>

Y Component [raster] <put parameter description here>

Step [number] <put parameter description here>

Default: *1*

Size Range Min [number] <put parameter description here>

Default: *25.0*

Size Range Max [number] <put parameter description here>

Default: *100.0*

Aggregation [selection] <put parameter description here>

Options:

- 0 — [0] nearest neighbour
- 1 — [1] mean value

Default: *0*

Style [selection] <put parameter description here>

Options:

- 0 — [0] simple line
- 1 — [1] arrow
- 2 — [2] arrow (centered to cell)

Default: *0*

Outputs

Gradient Vectors [vector] <put output description here>

Console usage

```
processing.runalg('saga:gradientvectorsfromdirectionalcomponents', x, y, step, size_min, size_max)
```

See also

Gradient vectors from direction and length

Description

<put algorithm description here>

Parameters

Direction [raster] <put parameter description here>

Length [raster] <put parameter description here>

Step [number] <put parameter description here>

Default: 1

Size Range Min [number] <put parameter description here>

Default: 25.0

Size Range Max [number] <put parameter description here>

Default: 100.0

Aggregation [selection] <put parameter description here>

Options:

- 0 — [0] nearest neighbour
- 1 — [1] mean value

Default: 0

Style [selection] <put parameter description here>

Options:

- 0 — [0] simple line
- 1 — [1] arrow
- 2 — [2] arrow (centered to cell)

Default: 0

Outputs

Gradient Vectors [vector] <put output description here>

Console usage

```
processing.runalg('saga:gradientvectorsfromdirectionandlength', dir, len, step, size_min, size_max)
```

See also

Gradient vectors from surface

Description

<put algorithm description here>

Parameters

Surface [raster] <put parameter description here>

Step [number] <put parameter description here>

Default: 1

Size Range Min [number] <put parameter description here>

Default: 25.0

Size Range Max [number] <put parameter description here>

Default: 100.0

Aggregation [selection] <put parameter description here>

Options:

- 0 — [0] nearest neighbour
- 1 — [1] mean value

Default: 0

Style [selection] <put parameter description here>

Options:

- 0 — [0] simple line
- 1 — [1] arrow
- 2 — [2] arrow (centered to cell)

Default: 0

Outputs

Gradient Vectors [vector] <put output description here>

Console usage

```
processing.runalg('saga:gradientvectorsfromsurface', surface, step, size_min, size_max, aggr, sty
```

See also

Grid statistics for polygons

Description

<put algorithm description here>

Parameters

Grids [multipleinput: rasters] <put parameter description here>

Polygons [vector: polygon] <put parameter description here>

Number of Cells [boolean] <put parameter description here>

Default: *True*

Minimum [boolean] <put parameter description here>

Default: *True*

Maximum [boolean] <put parameter description here>

Default: *True*

Range [boolean] <put parameter description here>

Default: *True*

Sum [boolean] <put parameter description here>

Default: *True*

Mean [boolean] <put parameter description here>

Default: *True*

Variance [boolean] <put parameter description here>

Default: *True*

Standard Deviation [boolean] <put parameter description here>

Default: *True*

Quantiles [number] <put parameter description here>

Default: *0*

Outputs

Statistics [vector] <put output description here>

Console usage

```
processing.runalg('saga:gridstatisticsforpolygons', grids, polygons, count, min, max, range, sum,
```

See also

Grid values to points (randomly)

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Frequency [number] <put parameter description here>

Default: *100*

Outputs

Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:gridvaluestopointsrandomly', grid, freq, points)
```

See also

Grid values to points

Description

<put algorithm description here>

Parameters

Grids [**multipleinput: rasters**] <put parameter description here>

Polygons [**vector: any**] Optional.

<put parameter description here>

Exclude NoData Cells [**boolean**] <put parameter description here>

Default: *True*

Type [**selection**] <put parameter description here>

Options:

- 0 — [0] nodes
- 1 — [1] cells

Default: *0*

Outputs

Shapes [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:gridvaluestopoints', grids, polygons, nodata, type, shapes)
```

See also

Local minima and maxima

Description

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Outputs

Minima [**vector**] <put output description here>

Maxima [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:localminimaandmaxima', grid, minima, maxima)
```

See also**Vectorising grid classes****Description**

<put algorithm description here>

Parameters

Grid [**raster**] <put parameter description here>

Class Selection [**selection**] <put parameter description here>

Options:

- 0 — [0] one single class specified by class identifier
- 1 — [1] all classes

Default: 0

Class Identifier [**number**] <put parameter description here>

Default: 0

Vectorised class as... [**selection**] <put parameter description here>

Options:

- 0 — [0] one single (multi-)polygon object
- 1 — [1] each island as separated polygon

Default: 0

Outputs

Polygons [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:vectorisinggridclasses', grid, class_all, class_id, split, polygons)
```

See also

.

18.7.15 Shapes lines**Convert points to line(s)****Description**

Converts points to lines.

Parameters

Points [vector: point] Points to convert.

Order by... [tablefield: any] Lines will be ordered following this field.

Separate by... [tablefield: any] Lines will be grouped according to this field.

Outputs

Lines [vector] The resulting layer.

Console usage

```
processing.runalg('saga:convertpointstolines', points, order, separate, lines)
```

See also

Convert polygons to lines

Description

Creates lines from polygons.

Parameters

Polygons [vector: polygon] Layer to process.

Outputs

Lines [vector] The resulting layer.

Console usage

```
processing.runalg('saga:convertpolygonstolines', polygons, lines)
```

See also

Line dissolve

Description

<put algorithm description here>

Parameters

Lines [vector: any] <put parameter description here>

1. **Attribute** [tablefield: any] <put parameter description here>
2. **Attribute** [tablefield: any] <put parameter description here>

3. Attribute [tablefield: any] <put parameter description here>

Dissolve... [selection] <put parameter description here>

Options:

- 0 — [0] lines with same attribute value(s)
- 1 — [1] all lines

Default: 0

Outputs

Dissolved Lines [vector] <put output description here>

Console usage

```
processing.runalg('saga:linedissolve', lines, field_1, field_2, field_3, all, dissolved)
```

See also

Line-polygon intersection

Description

<put algorithm description here>

Parameters

Lines [vector: line] <put parameter description here>

Polygons [vector: polygon] <put parameter description here>

Output [selection] <put parameter description here>

Options:

- 0 — [0] one multi-line per polygon
- 1 — [1] keep original line attributes

Default: 0

Outputs

Intersection [vector] <put output description here>

Console usage

```
processing.runalg('saga:linepolygonintersection', lines, polygons, method, intersect)
```

See also

Line properties

Description

Calculates some information on each line of the layer.

Parameters

Lines [**vector: line**] Layer to analyze.

Number of Parts [**boolean**] Determines whether to calculate number of segments in line.

Default: *True*

Number of Vertices [**boolean**] Determines whether to calculate number of vertices in line.

Default: *True*

Length [**boolean**] Determines whether to calculate total line length.

Default: *True*

Outputs

Lines with Property Attributes [**vector**] The resulting layer.

Console usage

```
processing.runalg('saga:lineproperties', lines, bparts, bpoints, blength, output)
```

See also

Line simplification

Description

Simplifies the geometry of a lines layer.

Parameters

Lines [**vector: line**] Layer to process.

Tolerance [**number**] Simplification tolerance.

Default: *1.0*

Outputs

Simplified Lines [**vector**] The resulting layer.

Console usage

```
processing.runalg('saga:linesimplification', lines, tolerance, output)
```

See also

.

18.7.16 Shapes points

Add coordinates to points

Description

Adds the X and Y coordinates of feature in the attribute table of input layer.

Parameters

Points [vector: point] Input layer.

Outputs

Output [vector] Resulting layer with the updated attribute table.

Console usage

```
processing.runalg('saga:addcoordinatestopoints', input, output)
```

See also

Add polygon attributes to points

Description

Adds the specified field of the polygons layer to the attribute table of the points layer. The new attributes added for each point depend on the value of the background polygon layer.

Parameters

Points [vector: point] Points layer.

Polygons [vector: polygon] Background polygons layer.

Attribute [tablefield: any] Attribute of the polygons layer that will be added to the points layer.

Outputs

Result [vector] The resulting layer.

Console usage

```
processing.runalg('saga:addpolygonattributestopoints', input, polygons, field, output)
```

See also

Aggregate point observations

Description

<put algorithm description here>

Parameters

Reference Points [vector: any] <put parameter description here>

ID [tablefield: any] <put parameter description here>

Observations [table] <put parameter description here>

X [tablefield: any] <put parameter description here>

Y [tablefield: any] <put parameter description here>

Track [tablefield: any] <put parameter description here>

Date [tablefield: any] <put parameter description here>

Time [tablefield: any] <put parameter description here>

Parameter [tablefield: any] <put parameter description here>

Maximum Time Span (Seconds) [number] <put parameter description here>

Default: *60.0*

Maximum Distance [number] <put parameter description here>

Default: *0.002*

Outputs

Aggregated [table] <put output description here>

Console usage

```
processing.runalg('saga:aggregatepointobservations', reference, reference_id, observations, x, y,
```

See also

Clip points with polygons

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Polygons [vector: polygon] <put parameter description here>

Add Attribute to Clipped Points [tablefield: any] <put parameter description here>

Clipping Options [selection] <put parameter description here>

Options:

- 0 — [0] one layer for all points
- 1 — [1] separate layer for each polygon

Default: 0

Outputs

Clipped Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:clippointswithpolygons', points, polygons, field, method, clips)
```

See also

Convert lines to points

Description

Converts lines layer into a points.

Parameters

Lines [vector: line] Lines layer to convert.

Insert Additional Points [boolean] Determines whether to add additional nodes or not.

Default: *True*

Insert Distance [number] Distance between the additional points.

Default: *1.0*

Outputs

Points [vector] The resulting layer.

Console usage

```
processing.runalg('saga:convertlinestopoints', lines, add, dist, points)
```

See also

Convert multipoints to points

Description

<put algorithm description here>

Parameters

Multipoints [**vector: point**] <put parameter description here>

Outputs

Points [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:convertmultipointstopoints', multipoints, points)
```

See also

Convex hull

Description

<put algorithm description here>

Parameters

Points [**vector: point**] <put parameter description here>

Hull Construction [**selection**] <put parameter description here>

Options:

- 0 — [0] one hull for all shapes
- 1 — [1] one hull per shape
- 2 — [2] one hull per shape part

Default: 0

Outputs

Convex Hull [**vector**] <put output description here>

Minimum Bounding Box [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:convexhull', shapes, polypoints, hulls, boxes)
```

See also

Distance matrix

Description

Generates a distance matrix between each point of the input layer. A unique ID will be created in the first row of the resulting matrix (symmetric matrix), while every other cell reflects the distance between the points.

Parameters

Points [vector: point] Input layer.

Outputs

Distance Matrix Table [table] The resulting table.

Console usage

```
processing.runalg('saga:distancematrix', points, table)
```

See also

Fit n points to shape

Description

<put algorithm description here>

Parameters

Shapes [vector: polygon] <put parameter description here>

Number of points [number] <put parameter description here>

Default: 10

Outputs

Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:fitnpointstoshape', shapes, numpoints, points)
```

See also

Points filter

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Radius [number] <put parameter description here>

Default: *1*

Minimum Number of Points [number] <put parameter description here>

Default: *0*

Maximum Number of Points [number] <put parameter description here>

Default: *0*

Quadrants [boolean] <put parameter description here>

Default: *True*

Filter Criterion [selection] <put parameter description here>

Options:

- 0 — [0] keep maxima (with tolerance)
- 1 — [1] keep minima (with tolerance)
- 2 — [2] remove maxima (with tolerance)
- 3 — [3] remove minima (with tolerance)
- 4 — [4] remove below percentile
- 5 — [5] remove above percentile

Default: *0*

Tolerance [number] <put parameter description here>

Default: *0.0*

Percentile [number] <put parameter description here>

Default: *50*

Outputs

Filtered Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:pointsfilter', points, field, radius, minnum, maxnum, quadrants, method, t
```

See also

Points thinning

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Resolution [number] <put parameter description here>

Default: 1.0

Outputs

Thinned Points [vector] <put output description here>

Console usage

```
processing.runalg('saga:pointsthinning', points, field, resolution, thinned)
```

See also

Remove duplicate points

Description

<put algorithm description here>

Parameters

Points [vector: any] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Point to Keep [selection] <put parameter description here>

Options:

- 0 — [0] first point
- 1 — [1] last point
- 2 — [2] point with minimum attribute value
- 3 — [3] point with maximum attribute value

Default: 0

Numeric Attribute Values [selection] <put parameter description here>

Options:

- 0 — [0] take value from the point to be kept

- 1 — [1] minimum value of all duplicates
- 2 — [2] maximum value of all duplicates
- 3 — [3] mean value of all duplicates

Default: 0

Outputs

Result [vector] <put output description here>

Console usage

```
processing.runalg('saga:removeduplicatepoints', points, field, method, numeric, result)
```

See also

Separate points by direction

Description

<put algorithm description here>

Parameters

Points [vector: point] <put parameter description here>

Number of Directions [number] <put parameter description here>

Default: 4

Tolerance (Degree) [number] <put parameter description here>

Default: 5

Outputs

Output [vector] <put output description here>

Console usage

```
processing.runalg('saga:separatepointsbydirection', points, directions, tolerance, output)
```

See also

.

18.7.17 Shapes polygons

Convert lines to polygons

Description

Converts lines to polygons.

Parameters

Lines [vector: line] Lines to convert.

Outputs

Polygons [vector] The resulting layer.

Console usage

```
processing.runalg('saga:convertlinestopolygons', lines, polygons)
```

See also

Convert polygon/line vertices to points

Description

Converts the line or polygon vertices into points.

Parameters

Shapes [vector: any] Layer to process.

Outputs

Points [vector] The resulting layer.

Console usage

```
processing.runalg('saga:convertpolygonlineverticestopoints', shapes, points)
```

See also

Polygon centroids

Description

Calculates the centroids of polygons.

Parameters

Polygons [vector: polygon] Input layer.

Centroids for each part [boolean] Determines whether centroids should be calculated for each part of multipart polygon or not.

Default: *True*

Outputs

Centroids [vector] The resulting layer.

Console usage

```
processing.runalg('saga:polygoncentroids', polygons, method, centroids)
```

See also

Polygon dissolve

Description

<put algorithm description here>

Parameters

Polygons [vector: polygon] <put parameter description here>

1. **Attribute** [tablefield: any] Optional.

<put parameter description here>

2. **Attribute** [tablefield: any] Optional.

<put parameter description here>

3. **Attribute** [tablefield: any] Optional.

<put parameter description here>

Dissolve... [selection] <put parameter description here>

Options:

- 0 — [0] polygons with same attribute value
- 1 — [1] all polygons
- 2 — [2] polygons with same attribute value (keep inner boundaries)
- 3 — [3] all polygons (keep inner boundaries)

Default: *0*

Outputs

Dissolved Polygons [vector] <put output description here>

Console usage

```
processing.runalg('saga:polygondissolve', polygons, field_1, field_2, field_3, dissolve, dissolve)
```

See also

Polygon-line intersection

Description

<put algorithm description here>

Parameters

Polygons [vector: **polygon**] <put parameter description here>

Lines [vector: **line**] <put parameter description here>

Outputs

Intersection [vector] <put output description here>

Console usage

```
processing.runalg('saga:polygonlineintersection', polygons, lines, intersect)
```

See also

Polygon parts to separate polygons

Description

<put algorithm description here>

Parameters

Polygons [vector: **polygon**] <put parameter description here>

Ignore Lakes [boolean] <put parameter description here>

Default: *True*

Outputs

Polygon Parts [vector] <put output description here>

Console usage

```
processing.runalg('saga:polygonpartstoseparatepolygons', polygons, lakes, parts)
```

See also

Polygon properties

Description

<put algorithm description here>

Parameters

Polygons [**vector: polygon**] <put parameter description here>

Number of Parts [**boolean**] <put parameter description here>

Default: *True*

Number of Vertices [**boolean**] <put parameter description here>

Default: *True*

Perimeter [**boolean**] <put parameter description here>

Default: *True*

Area [**boolean**] <put parameter description here>

Default: *True*

Outputs

Polygons with Property Attributes [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:polygonproperties', polygons, bparts, bpoints, blength, barea, output)
```

See also

Polygon shape indices

Description

Calculates spatial statistics for polygons. This includes:

- area
- perimeter
- perimeter / area
- perimeter / square root of the area
- maximum distance
- maximum distance / area
- maximum distance / square root of the area
- shape index

Parameters

Shapes [vector: polygon] Layer to analyze.

Outputs

Shape Index [vector] The resulting layer.

Console usage

```
processing.runalg('saga:polygonshapeindices', shapes, index)
```

See also

Polygons to edges and nodes

Description

Extracts boundaries and nodes of polygons in separate files.

Parameters

Polygons [vector: polygon] Input layer.

Outputs

Edges [vector] Resulting line layer with polygons boundaries.

Nodes [vector] Resulting line layer with polygons nodes.

Console usage

```
processing.runalg('saga:polygonstoedgesandnodes', polygons, edges, nodes)
```

See also

.

18.7.18 Shapes tools

Create graticule

Description

Creates a grid.

Parameters

Extent [vector: any] Optional.

Grid will be created according to the selected layer.

Output extent [extent] Extent of the grid.

Default: *0,1,0,1*

Division Width [number] X-axes spacing between the lines.

Default: *1.0*

Division Height [number] Y-axes spacing between the lines.

Default: *1.0*

Type [selection] Geometry type of the resulting grid.

Options:

- 0 — [0] Lines
- 1 — [1] Rectangles

Default: *0*

Outputs

Graticule [vector] The resulting layer.

Console usage

```
processing.runalg('saga:creategraticule', extent, output_extent, distx, disty, type, graticule)
```

See also

Cut shapes layer

Description

<put algorithm description here>

Parameters

Vector layer to cut [vector: any] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] completely contained
- 1 — [1] intersects
- 2 — [2] center

Default: *0*

Cutting polygons [vector: any] <put parameter description here>

Outputs

Result [vector] <put output description here>

Extent [vector] <put output description here>

Console usage

```
processing.runalg('saga:cutshapeslayer', shapes, method, polygons_polygons, cut, extent)
```

See also

Get shapes extents

Description

Creates polygons according to the extent of the input layer features.

Parameters

Shapes [vector: any] Input layer.

Parts [boolean] Determines whether create polygon for each feature (`True`) or just create single polygon for whole layer (`False`).

Default: *True*

Outputs

Extents [vector] The resulting layer.

Console usage

```
processing.runalg('saga:getshapesextents', shapes, parts, extents)
```

See also

Merge shapes layers

Description

Merges two or more input layer into a unique resulting layer. You can merge together only layer of the same type (polygons with polygons, lines with lines, points with points).

The attribute table of the resulting layer will include only the attributes of the first input layer. Two additional columns will be added: one corresponding to the ID of every merged layer and the other one corresponding to the original name of the merged layer.

Parameters

Main Layer [vector: any] Initial layer.

Additional Layers [multipleinput: any vectors] Optional.

Layer(s) to merge with.

Outputs

Merged Layer [vector] The resulting layer.

Console usage

```
processing.runalg('saga:mergeshapelayers', main, layers, out)
```

See also

Polar to cartesian coordinates

Description

<put algorithm description here>

Parameters

Polar Coordinates [vector: any] <put parameter description here>

Exaggeration [tablefield: any] <put parameter description here>

Exaggeration Factor [number] <put parameter description here>

Default: *1*

Radius [number] <put parameter description here>

Default: *6371000.0*

Degree [boolean] <put parameter description here>

Default: *True*

Outputs

Cartesian Coordinates [vector] <put output description here>

Console usage

```
processing.runalg('saga:polartocartesiancoordinates', polar, f_exagg, d_exagg, radius, degree, ca
```


See also**Quadtree structure to shapes****Description**

<put algorithm description here>

Parameters

Shapes [**vector: any**] <put parameter description here>

Attribute [**tablefield: any**] <put parameter description here>

Outputs

Polygons [**vector**] <put output description here>

Lines [**vector**] <put output description here>

Duplicated Points [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:quadtreestructuretoshapes', shapes, attribute, polygons, lines, points)
```

See also**Shapes buffer****Description**

Creates buffer around features based on fixed distance or distance field.

Parameters

Shapes [**vector: any**] Input layer.

Buffer Distance [**selection**] Buffering method.

Options:

- 0 — [0] fixed value
- 1 — [1] attribute field

Default: 0

Buffer Distance (Fixed) [**number**] Buffer distance for “fixed value” method.

Default: 100.0

Buffer Distance (Attribute) [**tablefield: any**] Name of the distance field for “attribute field” method.

Scaling Factor for Attribute Value [**number**] <put parameter description here>

Default: 1.0

Number of Buffer Zones [number] Number of buffer(s) to generate.

Default: *1.0*

Circle Point Distance [Degree] [number] Smoothness of the buffer borders: great numbers means rough borders.

Default: *5.0*

Dissolve Buffers [boolean] Determines whether to dissolve results or not.

Default: *True*

Outputs

Buffer [vector] The resulting layer.

Console usage

```
processing.runalg('saga:shapesbuffer', shapes, buf_type, buf_dist, buf_field, buf_scale, buf_zones)
```

See also

Split shapes layer randomly

Description

Splits the input layer randomly in two parts.

Parameters

Shapes [vector: any] Layer to split.

Split ratio (%) [number] Split ratio between the resulting layers.

Default: *50*

Outputs

Group A [vector] First resulting layer.

Group B [vector] Second resulting layer.

Console usage

```
processing.runalg('saga:splitshapeslayerrandomly', shapes, percent, a, b)
```

See also

Transform shapes

Description

<put algorithm description here>

Parameters

Shapes [**vector: any**] <put parameter description here>

dX [**number**] <put parameter description here>

Default: *0.0*

dY [**number**] <put parameter description here>

Default: *0.0*

Angle [**number**] <put parameter description here>

Default: *0.0*

Scale Factor X [**number**] <put parameter description here>

Default: *1.0*

Scale Factor Y [**number**] <put parameter description here>

Default: *1.0*

X [**number**] <put parameter description here>

Default: *0.0*

Y [**number**] <put parameter description here>

Default: *0.0*

Outputs

Output [**vector**] <put output description here>

Console usage

```
processing.runalg('saga:transformshapes', in, dx, dy, angle, scalex, scaley, anchorx, anchory, out)
```

See also

.

18.7.19 Shapes transect

Transect through polygon shapefile

Description

<put algorithm description here>

Parameters

Line Transect (s) [**vector: line**] <put parameter description here>

Theme [**vector: any**] <put parameter description here>

Theme Field [**tablefield: any**] <put parameter description here>

Outputs

Result table [table] <put output description here>

Console usage

```
processing.runalg('saga:transectthroughpolygonshapefile', transect, theme, theme_field, transect_
```

See also

.

18.7.20 Simulation fire

Fire risk analysis

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Fuel Model [raster] <put parameter description here>

Wind Speed [raster] <put parameter description here>

Wind Direction [raster] <put parameter description here>

Dead Fuel Moisture 1H [raster] <put parameter description here>

Dead Fuel Moisture 10H [raster] <put parameter description here>

Dead Fuel Moisture 100H [raster] <put parameter description here>

Herbaceous Fuel Moisture [raster] <put parameter description here>

Wood Fuel Moisture [raster] <put parameter description here>

Value [raster] Optional.

<put parameter description here>

Base Probability [raster] Optional.

<put parameter description here>

Number of Events [number] <put parameter description here>

Default: *1000*

Fire Length [number] <put parameter description here>

Default: *100*

Outputs

Danger [raster] <put output description here>

Compound Probability [raster] <put output description here>

Priority Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:fireriskanalysis', dem, fuel, windspd, winddir, m1h, m10h, m100h, mherb, m
```

See also

Simulation

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Fuel Model [raster] <put parameter description here>

Wind Speed [raster] <put parameter description here>

Wind Direction [raster] <put parameter description here>

Dead Fuel Moisture 1H [raster] <put parameter description here>

Dead Fuel Moisture 10H [raster] <put parameter description here>

Dead Fuel Moisture 100H [raster] <put parameter description here>

Herbaceous Fuel Moisture [raster] <put parameter description here>

Wood Fuel Moisture [raster] <put parameter description here>

Ignition Points [raster] <put parameter description here>

Update View [boolean] <put parameter description here>

Default: *True*

Outputs

Time [raster] <put output description here>

Flame Length [raster] <put output description here>

Intensity [raster] <put output description here>

Console usage

```
processing.runalg('saga:simulation', dem, fuel, windspd, winddir, m1h, m10h, m100h, mherb, mwood,
```

See also

.

18.7.21 Simulation hydrology

Overland flow - kinematic wave d8

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Gauges [vector: any] Optional.

<put parameter description here>

Simulation Time [h] [number] <put parameter description here>

Default: 24

Simulation Time Step [h] [number] <put parameter description here>

Default: 0.1

Manning's Roughness [number] <put parameter description here>

Default: 0.03

Max. Iterations [number] <put parameter description here>

Default: 100

Epsilon [number] <put parameter description here>

Default: 0.0001

Precipitation [selection] <put parameter description here>

Options:

- 0 — [0] Homogenous
- 1 — [1] Above Elevation
- 2 — [2] Left Half

Default: 0

Threshold Elevation [number] <put parameter description here>

Default: 0.0

Outputs

Runoff [raster] <put output description here>

Flow at Gauges [table] <put output description here>

Console usage

```
processing.runalg('saga:overlandflowkinematicwaved8', dem, gauges, time_span, time_step, roughnes
```

See also

Water retention capacity

Description

<put algorithm description here>

Parameters

Plot Holes [vector: any] <put parameter description here>

DEM [raster] <put parameter description here>

Outputs

Final Parameters [vector] <put output description here>

Water Retention Capacity [raster] <put output description here>

Console usage

```
processing.runalg('saga:waterretentioncapacity', shapes, dem, output, retention)
```

See also

.

18.7.22 Table calculus

Fill gaps in records

Description

<put algorithm description here>

Parameters

Table [table] <put parameter description here>

Order [tablefield: any] <put parameter description here>

Interpolation [selection] <put parameter description here>

Options:

- 0 — [0] Nearest Neighbour
- 1 — [1] Linear

- 2 — [2] Spline

Default: 0

Outputs

Table without Gaps [table] <put output description here>

Console usage

```
processing.runalg('saga:fillgapsinrecords', table, order, method, nogaps)
```

See also

Principle components analysis

Description

<put algorithm description here>

Parameters

Table [table] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] correlation matrix
- 1 — [1] variance-covariance matrix
- 2 — [2] sums-of-squares-and-cross-products matrix

Default: 0

Number of Components [number] <put parameter description here>

Default: 3

Outputs

Principle Components [table] <put output description here>

Console usage

```
processing.runalg('saga:principlecomponentsanalysis', table, method, nfirst, pca)
```

See also

Running average

Description

<put algorithm description here>

Parameters

Input [table] <put parameter description here>

Attribute [tablefield: any] <put parameter description here>

Number of Records [number] <put parameter description here>

Default: 10

Outputs

Output [table] <put output description here>

Console usage

```
processing.runalg('saga:runningaverage', input, field, count, output)
```

See also

.

18.7.23 Table tools

Change date format

Description

Converts the date format of the input layer.

Parameters

Table [table] Input table.

Date Field [tablefield: any] Attribute the date.

Input Format [selection] Input date format.

Options:

- 0 — [0] dd.mm.yy
- 1 — [1] yy.mm.dd
- 2 — [2] dd:mm:yy
- 3 — [3] yy:mm:dd
- 4 — [4] ddmmyyyy, fix size
- 5 — [5] yyyymmdd, fix size
- 6 — [6] ddmmyy, fix size
- 7 — [7] yymmdd, fix size
- 8 — [8] Julian Day

Default: 0

Output Format [selection] Output date format.

Options:

- 0 — [0] dd.mm.yy
- 1 — [1] yy.mm.dd
- 2 — [2] dd:mm:yy
- 3 — [3] yy:mm:dd
- 4 — [4] ddmmyyyy, fix size
- 5 — [5] yyyyymmdd, fix size
- 6 — [6] ddmmyy, fix size
- 7 — [7] yymmdd, fix size
- 8 — [8] Julian Day

Default: 0

Outputs

Output [table] The resulting table.

Console usage

```
processing.runalg('saga:changedateformat', table, field, fmt_in, fmt_out, output)
```

See also

Change time format

Description

Converts the time format of the input layer.

Parameters

Table [table] Input table.

Time Field [tablefield: any] Attribute with time.

Input Format [selection] Input time format.

Options:

- 0 — [0] hh.mm.ss
- 1 — [1] hh:mm:ss
- 2 — [2] hhmmss, fix size
- 3 — [3] hours
- 4 — [4] minutes
- 5 — [5] seconds

Default: 0

Output Format [selection] Output time format.

Options:

- 0 — [0] hh.mm.ss
- 1 — [1] hh:mm:ss
- 2 — [2] hhmmss, fix size
- 3 — [3] hours
- 4 — [4] minutes
- 5 — [5] seconds

Default: 0

Outputs

Output [table] The resulting table.

Console usage

```
processing.runalg('saga:changetimeformat', table, field, fmt_in, fmt_out, output)
```

See also

.

18.7.24 Terrain channels

Channel network and drainage basins

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Threshold [number] <put parameter description here>

Default: 5.0

Outputs

Flow Direction [raster] <put output description here>

Flow Connectivity [raster] <put output description here>

Strahler Order [raster] <put output description here>

Drainage Basins [raster] <put output description here>

Channels [vector] <put output description here>

Drainage Basins [vector] <put output description here>

Junctions [vector] <put output description here>

Console usage

```
processing.runalg('saga:channelnetworkanddrainagebasins', dem, threshold, direction, connection, c
```

See also

Channel network

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Flow Direction [raster] Optional.

<put parameter description here>

Initiation Grid [raster] <put parameter description here>

Initiation Type [selection] <put parameter description here>

Options:

- 0 — [0] Less than
- 1 — [1] Equals
- 2 — [2] Greater than

Default: 0

Initiation Threshold [number] <put parameter description here>

Default: 0.0

Divergence [raster] Optional.

<put parameter description here>

Tracing: Max. Divergence [number] <put parameter description here>

Default: 10

Tracing: Weight [raster] Optional.

<put parameter description here>

Min. Segment Length [number] <put parameter description here>

Default: 10

Outputs

Channel Network [raster] <put output description here>

Channel Direction [raster] <put output description here>

Channel Network [vector] <put output description here>

Console usage

```
processing.runalg('saga:channelnetwork', elevation, sinkroute, init_grid, init_method, init_value)
```

See also

Overland flow distance to channel network

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Channel Network [raster] <put parameter description here>

Flow Algorithm [selection] <put parameter description here>

Options:

- 0 — [0] D8
- 1 — [1] MFD

Default: 0

Outputs

Overland Flow Distance [raster] <put output description here>

Vertical Overland Flow Distance [raster] <put output description here>

Horizontal Overland Flow Distance [raster] <put output description here>

Console usage

```
processing.runalg('saga:overlandflowdistancetochannelnetwork', elevation, channels, method, distanceto)
```

See also

Strahler order

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Outputs

Strahler Order [raster] <put output description here>

Console usage

```
processing.runalg('saga:strahlerorder', dem, strahler)
```

See also

Vertical distance to channel network

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Channel Network [raster] <put parameter description here>

Tension Threshold [Percentage of Cell Size] [number] <put parameter description here>

Default: *1*

Keep Base Level below Surface [boolean] <put parameter description here>

Default: *True*

Outputs

Vertical Distance to Channel Network [raster] <put output description here>

Channel Network Base Level [raster] <put output description here>

Console usage

```
processing.runalg('saga:verticaldistancetochannelnetwork', elevation, channels, threshold, nounde
```

See also

Watershed basins

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Channel Network [raster] <put parameter description here>

Sink Route [raster] Optional.
<put parameter description here>

Min. Size [number] <put parameter description here>
Default: 0

Outputs

Watershed Basins [raster] <put output description here>

Console usage

```
processing.runalg('saga:watershedbasins', elevation, channels, sinkroute, minsize, basins)
```

See also

.

18.7.25 Terrain hydrology

Burn stream network into dem

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Streams [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] simply decrease cell's value by epsilon
- 1 — [1] lower cell's value to neighbours minimum value minus epsilon

Default: 0

Epsilon [number] <put parameter description here>

Default: 1.0

Outputs

Processed DEM [raster] <put output description here>

Console usage

```
processing.runalg('saga:burnstreamnetworkintodem', dem, stream, method, epsilon, burn)
```

See also

Catchment area (flow tracing)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Sink Routes [raster] Optional.

<put parameter description here>

Weight [raster] Optional.

<put parameter description here>

Material [raster] Optional.

<put parameter description here>

Target [raster] Optional.

<put parameter description here>

Step [number] <put parameter description here>

Default: *1*

Method [selection] <put parameter description here>

Options:

- 0 — [0] Rho 8
- 1 — [1] Kinematic Routing Algorithm
- 2 — [2] DEMON

Default: *0*

DEMON - Min. DQV [number] <put parameter description here>

Default: *0.0*

Flow Correction [boolean] <put parameter description here>

Default: *True*

Outputs

Catchment Area [raster] <put output description here>

Catchment Height [raster] <put output description here>

Catchment Slope [raster] <put output description here>

Total accumulated Material [raster] <put output description here>

Accumulated Material from _left_ side [raster] <put output description here>

Accumulated Material from _right_ side [raster] <put output description here>

Console usage

```
processing.runalg('saga:catchmentareaflowtracing', elevation, sinkroute, weight, material, target,
```

See also

Catchment area (recursive)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Sink Routes [raster] Optional.

<put parameter description here>

Weight [raster] Optional.

<put parameter description here>

Material [raster] Optional.

<put parameter description here>

Target [raster] Optional.

<put parameter description here>

Step [number] <put parameter description here>

Default: *1*

Target Areas [raster] Optional.

<put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8
- 1 — [1] Rho 8
- 2 — [2] Deterministic Infinity
- 3 — [3] Multiple Flow Direction

Default: *0*

Convergence [number] <put parameter description here>

Default: *1.1*

Outputs

Catchment Area [raster] <put output description here>

Catchment Height [raster] <put output description here>

Catchment Slope [raster] <put output description here>

Total accumulated Material [raster] <put output description here>

Accumulated Material from _left_ side [raster] <put output description here>

Accumulated Material from _right_ side [raster] <put output description here>

Flow Path Length [raster] <put output description here>

Console usage

```
processing.runalg('saga:catchmentarearecursive', elevation, sinkroute, weight, material, target, ...)
```

See also

Catchment Area

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8
- 1 — [1] Rho 8
- 2 — [2] Braunschweiger Reliefmodell
- 3 — [3] Deterministic Infinity
- 4 — [4] Multiple Flow Direction
- 5 — [5] Multiple Triangular Flow Directon

Default: 0

Outputs

Catchment Area [raster] <put output description here>

Console usage

```
processing.runalg('saga:catchmentarea', elevation, method, carea)
```

See also

Cell balance

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Parameter [raster] Optional.

<put parameter description here>

Default Weight [number] <put parameter description here>

Default: *1.0*

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8
- 1 — [1] Multiple Flow Direction

Default: *0*

Outputs

Cell Balance [raster] <put output description here>

Console usage

```
processing.runalg('saga:cellbalance', dem, weights, weight, method, balance)
```

See also

Edge contamination

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Outputs

Edge Contamination [raster] <put output description here>

Console usage

```
processing.runalg('saga:edgecontamination', dem, contamination)
```

See also

Fill Sinks

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Minimum Slope [Degree] [number] <put parameter description here>

Default: *0.01*

Outputs

Filled DEM [raster] <put output description here>

Console usage

```
processing.runalg('saga:fillsinks', dem, minslope, result)
```

See also

Fill sinks (wang & liu)

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Minimum Slope [Degree] [number] <put parameter description here>

Default: *0.01*

Outputs

Filled DEM [raster] <put output description here>

Flow Directions [raster] <put output description here>

Watershed Basins [raster] <put output description here>

Console usage

```
processing.runalg('saga:fillsinkswangliu', elev, minslope, filled, fdir, wshed)
```

See also

Fill sinks xxi (wang & liu)

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Minimum Slope [Degree] [number] <put parameter description here>

Default: *0.01*

Outputs

Filled DEM [raster] <put output description here>

Console usage

```
processing.runalg('saga:fillsinksxxlwangliu', elev, minslope, filled)
```

See also

Flat detection

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Flat Area Values [selection] <put parameter description here>

Options:

- 0 — [0] elevation
- 1 — [1] enumeration

Default: *0*

Outputs

No Flats [raster] <put output description here>

Flat Areas [raster] <put output description here>

Console usage

```
processing.runalg('saga:flatdetection', dem, flat_output, noflats, flats)
```

See also

Flow path length

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Seeds [raster] Optional.

<put parameter description here>

Seeds Only [boolean] <put parameter description here>

Default: *True*

Flow Routing Algorithm [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8 (D8)
- 1 — [1] Multiple Flow Direction (FD8)

Default: *0*

Convergence (FD8) [number] <put parameter description here>

Default: *1.1*

Outputs

Flow Path Length [raster] <put output description here>

Console usage

```
processing.runalg('saga:flowpathlength', elevation, seed, seeds_only, method, convergence, length)
```

See also**Flow width and specific catchment area****Description**

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Total Catchment Area (TCA) [raster] Optional.

<put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8
- 1 — [1] Multiple Flow Direction (Quinn et al. 1991)
- 2 — [2] Aspect

Default: 0

Outputs

Flow Width [raster] <put output description here>

Specific Catchment Area (SCA) [raster] <put output description here>

Console usage

```
processing.runalg('saga:flowwidthandspecificcatchmentarea', dem, tca, method, width, sca)
```

See also**Lake flood****Description**

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Seeds [raster] <put parameter description here>

Absolute Water Levels [boolean] <put parameter description here>

Default: *True*

Outputs

Lake [raster] <put output description here>

Surface [raster] <put output description here>

Console usage

```
processing.runalg('saga:lakeflood', elev, seeds, level, outdepth, outlevel)
```

See also

Ls factor

Description

<put algorithm description here>

Parameters

Slope [raster] <put parameter description here>

Catchment Area [raster] <put parameter description here>

Area to Length Conversion [selection] <put parameter description here>

Options:

- 0 — [0] no conversion (areas already given as specific catchment area)
- 1 — [1] 1 / cell size (specific catchment area)
- 2 — [2] square root (catchment length)

Default: 0

Method (LS) [selection] <put parameter description here>

Options:

- 0 — [0] Moore et al. 1991
- 1 — [1] Desmet & Govers 1996
- 2 — [2] Bochner & Selige 2006

Default: 0

Rill/Interrill Erosivity [number] <put parameter description here>

Default: 0.0

Stability [selection] <put parameter description here>

Options:

- 0 — [0] stable
- 1 — [1] instable (thawing)

Default: 0

Outputs

LS Factor [raster] <put output description here>

Console usage

```
processing.runalg('saga:lsfactor', slope, area, conv, method, erosivity, stability, ls)
```

See also

Saga wetness index

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

t [number] <put parameter description here>

Default: *10*

Outputs

Catchment area [raster] <put output description here>

Catchment slope [raster] <put output description here>

Modified catchment area [raster] <put output description here>

Wetness index [raster] <put output description here>

Console usage

```
processing.runalg('saga:sagawetnessindex', dem, t, c, gn, cs, sb)
```

See also

Sink drainage route detection

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Threshold [boolean] <put parameter description here>

Default: *True*

Threshold Height [number] <put parameter description here>

Default: *100.0*

Outputs

Sink Route [raster] <put output description here>

Console usage

```
processing.runalg('saga:sinkdrainageroutedetection', elevation, threshold, thrsheight, sinkroute)
```

See also

Sink removal

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Sink Route [raster] Optional.

<put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deepen Drainage Routes
- 1 — [1] Fill Sinks

Default: *0*

Threshold [boolean] <put parameter description here>

Default: *True*

Threshold Height [number] <put parameter description here>

Default: *100.0*

Outputs

Preprocessed DEM [raster] <put output description here>

Console usage

```
processing.runalg('saga:sinkremoval', dem, sinkroute, method, threshold, thrsheight, dem_preproc)
```

See also

Slope length

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Outputs

Slope Length [raster] <put output description here>

Console usage

```
processing.runalg('saga:slopelength', dem, length)
```

See also

Stream power index

Description

<put algorithm description here>

Parameters

Slope [raster] <put parameter description here>

Catchment Area [raster] <put parameter description here>

Area Conversion [selection] <put parameter description here>

Options:

- 0 — [0] no conversion (areas already given as specific catchment area)
- 1 — [1] 1 / cell size (pseudo specific catchment area)

Default: 0

Outputs

Stream Power Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:streampowerindex', slope, area, conv, spi)
```

See also

Topographic wetness index (twi)

Description

<put algorithm description here>

Parameters

Slope [raster] <put parameter description here>

Catchment Area [raster] <put parameter description here>

Transmissivity [raster] Optional.

<put parameter description here>

Area Conversion [selection] <put parameter description here>

Options:

- 0 — [0] no conversion (areas already given as specific catchment area)
- 1 — [1] 1 / cell size (pseudo specific catchment area)

Default: 0

Method (TWI) [selection] <put parameter description here>

Options:

- 0 — [0] Standard
- 1 — [1] TOPMODEL

Default: 0

Outputs

Topographic Wetness Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:topographicwetnessindextwi', slope, area, trans, conv, method, twi)
```

See also

Upslope Area

Description

<put algorithm description here>

Parameters

Target Area [raster] Optional.

<put parameter description here>

Target X coordinate [number] <put parameter description here>

Default: *0.0*

Target Y coordinate [number] <put parameter description here>

Default: *0.0*

Elevation [raster] <put parameter description here>

Sink Routes [raster] Optional.

<put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Deterministic 8
- 1 — [1] Deterministic Infinity
- 2 — [2] Multiple Flow Direction

Default: *0*

Convergence [number] <put parameter description here>

Default: *1.1*

Outputs

Upslope Area [raster] <put output description here>

Console usage

```
processing.runalg('saga:upslopearea', target, target_pt_x, target_pt_y, elevation, sinkroute, met)
```

See also

.

18.7.26 Terrain lighting

Analytical hillshading

Description

<put algorithm description here>

Parameters

Elevation [**raster**] <put parameter description here>

Shading Method [**selection**] <put parameter description here>

Options:

- 0 — [0] Standard
- 1 — [1] Standard (max. 90Degree)
- 2 — [2] Combined Shading
- 3 — [3] Ray Tracing

Default: 0

Azimuth [**Degree**] [**number**] <put parameter description here>

Default: 315.0

Declination [**Degree**] [**number**] <put parameter description here>

Default: 45.0

Exaggeration [**number**] <put parameter description here>

Default: 4.0

Outputs

Analytical Hillshading [**raster**] <put output description here>

Console usage

```
processing.runalg('saga:analyticalhillshading', elevation, method, azimuth, declination, exaggeration)
```

See also

Sky view factor

Description

<put algorithm description here>

Parameters

Elevation [**raster**] <put parameter description here>

Maximum Search Radius [**number**] <put parameter description here>

Default: 10000

Method [**selection**] <put parameter description here>

Options:

- 0 — [0] multi scale
- 1 — [1] sectors

Default: 0

Multi Scale Factor [number] <put parameter description here>

Default: 3

Number of Sectors [number] <put parameter description here>

Default: 8

Outputs

Visible Sky [raster] <put output description here>

Sky View Factor [raster] <put output description here>

Sky View Factor (Simplified) [raster] <put output description here>

Terrain View Factor [raster] <put output description here>

Console usage

```
processing.runalg('saga:skyviewfactor', dem, maxradius, method, level_inc, ndirs, visible, svf, s
```

See also

Topographic correction

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Original Image [raster] <put parameter description here>

Azimuth [number] <put parameter description here>

Default: 180.0

Height [number] <put parameter description here>

Default: 45.0

Method [selection] <put parameter description here>

Options:

- 0 — [0] Cosine Correction (Teillet et al. 1982)
- 1 — [1] Cosine Correction (Civco 1989)
- 2 — [2] Minnaert Correction
- 3 — [3] Minnaert Correction with Slope (Riano et al. 2003)
- 4 — [4] Minnaert Correction with Slope (Law & Nichol 2004)
- 5 — [5] C Correction
- 6 — [6] Normalization (after Civco, modified by Law & Nichol)

Default: 0

Minnaert Correction [number] <put parameter description here>

Default: *0.5*

Maximum Cells (C Correction Analysis) [number] <put parameter description here>

Default: *1000*

Value Range [selection] <put parameter description here>

Options:

- 0 — [0] 1 byte (0-255)
- 1 — [1] 2 byte (0-65535)

Default: *0*

Outputs

Corrected Image [raster] <put output description here>

Console usage

```
processing.runalg('saga:topographiccorrection', dem, original, azi, hgt, method, minnaert, maxcel.
```

See also

.

18.7.27 Terrain morphometry

Convergence index

Description

Calculates an index of convergence/divergence regarding to overland flow. By its meaning it is similar to plan or horizontal curvature, but gives much smoother results. The calculation uses the aspects of surrounding cells, i.e. it looks to which degree surrounding cells point to the center cell. The result is given as percentages, negative values correspond to convergent, positive to divergent flow conditions. Minus 100 would be like a peak of a cone, plus 100 a pit, and 0 an even slope.

Parameters

Elevation [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Aspect
- 1 — [1] Gradient

Default: *0*

Gradient Calculation [selection] <put parameter description here>

Options:

- 0 — [0] 2 x 2

- 1 — [1] 3 x 3

Default: 0

Outputs

Convergence Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:convergenceindex', elevation, method, neighbours, result)
```

See also

- Koethe, R. / Lehmeier, F. (1996): 'SARA, System zur Automatischen Relief-Analyse', Benutzerhandbuch, 2. Auflage [Geogr. Inst. Univ. Goettingen, unpublished]

Convergence index (search radius)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Radius [Cells] [number] <put parameter description here>

Default: 10

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [number] <put parameter description here>

Default: 1

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: 1

Gradient [boolean] <put parameter description here>

Default: *True*

Difference [selection] <put parameter description here>

Options:

- 0 — [0] direction to the center cell
- 1 — [1] center cell's aspect direction

Default: 0

Outputs

Convergence Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:convergenceindexsearchradius', elevation, radius, distance_weighting_weight)
```

See also

Curvature classification

Description

<put algorithm description here>

Parameters

Plan Curvature [raster] <put parameter description here>

Profile Curvature [raster] <put parameter description here>

Threshold for plane [number] <put parameter description here>

Default: 0.001

Outputs

Curvature Classification [raster] <put output description here>

Console usage

```
processing.runalg('saga:curvatureclassification', cplan, cprof, threshold, class)
```

See also

Diurnal anisotropic heating

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Alpha Max (Degree) [number] <put parameter description here>

Default: *202.5*

Outputs

Diurnal Anisotropic Heating [raster] <put output description here>

Console usage

```
processing.runalg('saga:diurnalanisotropicheating', dem, alpha_max, dah)
```

See also

Downslope distance gradient

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Vertical Distance [number] <put parameter description here>

Default: *10*

Output [selection] <put parameter description here>

Options:

- 0 — [0] distance
- 1 — [1] gradient (tangens)
- 2 — [2] gradient (degree)

Default: *0*

Outputs

Gradient [raster] <put output description here>

Gradient Difference [raster] <put output description here>

Console usage

```
processing.runalg('saga:downslopedistancegradient', dem, distance, output, gradient, difference)
```

See also

Effective air flow heights

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Wind Direction [raster] Optional.

<put parameter description here>

Wind Speed [raster] Optional.

<put parameter description here>

Constant Wind Direction [Degree] [number] <put parameter description here>

Default: *135*

Old Version [boolean] <put parameter description here>

Default: *True*

Search Distance [km] [number] <put parameter description here>

Default: *300*

Acceleration [number] <put parameter description here>

Default: *1.5*

Use Pyramids with New Version [boolean] <put parameter description here>

Default: *True*

Lee Factor [number] <put parameter description here>

Default: *0.5*

Luv Factor [number] <put parameter description here>

Default: *1.0*

Wind Direction Units [selection] <put parameter description here>

Options:

- 0 — [0] radians
- 1 — [1] degree

Default: *0*

Wind Speed Scale Factor [number] <put parameter description here>

Default: *1.0*

Outputs

Effective Air Flow Heights [raster] <put output description here>

Console usage

```
processing.runalg('saga:effectiveairflowheights', dem, dir, len, dir_const, oldver, maxdist, acce
```

See also

Hypsometry

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Number of Classes [number] <put parameter description here>

Default: *100.0*

Sort [selection] <put parameter description here>

Options:

- 0 — [0] up
- 1 — [1] down

Default: *0*

Classification Constant [selection] <put parameter description here>

Options:

- 0 — [0] height
- 1 — [1] area

Default: *0*

Use Z-Range [boolean] <put parameter description here>

Default: *True*

Z-Range Min [number] <put parameter description here>

Default: *0.0*

Z-Range Max [number] <put parameter description here>

Default: *1000.0*

Outputs

Hypsometry [table] <put output description here>

Console usage

```
processing.runalg('saga:hypsometry', elevation, count, sorting, method, bzrange, zrange_min, zran
```

See also

Land surface temperature

Description

<put algorithm description here>

Parameters

Elevation [m] [raster] <put parameter description here>

Short Wave Radiation [kW/m²] [raster] <put parameter description here>

Leaf Area Index [raster] <put parameter description here>

Elevation at Reference Station [m] [number] <put parameter description here>

Default: 0.0

Temperature at Reference Station [Deg.Celsius] [number] <put parameter description here>

Default: 0.0

Temperature Gradient [Deg.Celsius/km] [number] <put parameter description here>

Default: 6.5

C Factor [number] <put parameter description here>

Default: 1.0

Outputs

Land Surface Temperature [Deg.Celsius] [raster] <put output description here>

Console usage

```
processing.runalg('saga:landsurfacetemperature', dem, swr, lai, z_reference, t_reference, t_gradient)
```

See also

Mass balance index

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Vertical Distance to Channel Network [raster] Optional.

<put parameter description here>

T Slope [number] <put parameter description here>

Default: *15.0*

T Curvature [number] <put parameter description here>

Default: *0.01*

T Vertical Distance to Channel Network [number] <put parameter description here>

Default: *15.0*

Outputs

Mass Balance Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:massbalanceindex', dem, hrel, tslope, tcurve, threl, mbi)
```

See also

Morphometric protection index

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Radius [number] <put parameter description here>

Default: *2000.0*

Outputs

Protection Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:morphometricprotectionindex', dem, radius, protection)
```

See also

Multiresolution index of valley bottom flatness (mrvbf)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Initial Threshold for Slope [number] <put parameter description here>

Default: *16*

Threshold for Elevation Percentile (Lowness) [number] <put parameter description here>

Default: *0.4*

Threshold for Elevation Percentile (Upness) [number] <put parameter description here>

Default: *0.35*

Shape Parameter for Slope [number] <put parameter description here>

Default: *4.0*

Shape Parameter for Elevation Percentile [number] <put parameter description here>

Default: *3.0*

Update Views [boolean] <put parameter description here>

Default: *True*

Classify [boolean] <put parameter description here>

Default: *True*

Maximum Resolution (Percentage) [number] <put parameter description here>

Default: *100*

Outputs

MRVBF [raster] <put output description here>

MRRTF [raster] <put output description here>

Console usage

```
processing.runalg('saga:multiresolutionindexofvalleybottomflatnessmrvbf', dem, t_slope, t_pctl_v,
```

See also

Real area calculation

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Outputs

Real Area Grid [raster] <put output description here>

Console usage

```
processing.runalg('saga:realareacalculation', dem, area)
```

See also

Relative heights and slope positions

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

w [number] <put parameter description here>

Default: *0.5*

t [number] <put parameter description here>

Default: *10.0*

e [number] <put parameter description here>

Default: *2.0*

Outputs

Slope Height [raster] <put output description here>

Valley Depth [raster] <put output description here>

Normalized Height [raster] <put output description here>

Standardized Height [raster] <put output description here>

Mid-Slope Positon [raster] <put output description here>

Console usage

```
processing.runalg('saga:relativeheightsandslopepositions', dem, w, t, e, ho, hu, nh, sh, ms)
```

See also

Slope, aspect, curvature

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Maximum Slope (Travis et al. 1975)
- 1 — [1] Maximum Triangle Slope (Tarboton 1997)
- 2 — [2] Least Squares Fitted Plane (Horn 1981, Costa-Cabral & Burgess 1996)
- 3 — [3] Fit 2.Degree Polynom (Bauer, Rohdenburg, Bork 1985)
- 4 — [4] Fit 2.Degree Polynom (Heerdegen & Beran 1982)
- 5 — [5] Fit 2.Degree Polynom (Zevenbergen & Thorne 1987)
- 6 — [6] Fit 3.Degree Polynom (Haralick 1983)

Default: 5

Outputs

Slope [raster] <put output description here>

Aspect [raster] <put output description here>

Curvature [raster] <put output description here>

Plan Curvature [raster] <put output description here>

Profile Curvature [raster] <put output description here>

Console usage

```
processing.runalg('saga:slopeaspectcurvature', elevation, method, slope, aspect, curv, hcurv, vcurv)
```

See also

Surface specific points

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Method [selection] <put parameter description here>

Options:

- 0 — [0] Mark Highest Neighbour
- 1 — [1] Opposite Neighbours
- 2 — [2] Flow Direction
- 3 — [3] Flow Direction (up and down)

- 4 — [4] Peucker & Douglas

Default: 0

Threshold [number] <put parameter description here>

Default: 2.0

Outputs

Result [raster] <put output description here>

Console usage

```
processing.runalg('saga:surfacespecificpoints', elevation, method, threshold, result)
```

See also

Terrain ruggedness index (tri)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Radius (Cells) [number] <put parameter description here>

Default: 1

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: 0

Inverse Distance Weighting Power [number] <put parameter description here>

Default: 1

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: 1.0

Outputs

Terrain Ruggedness Index (TRI) [raster] <put output description here>

Console usage

```
processing.runalg('saga:terrainruggednessindextri', dem, radius, distance_weighting_weighting, di
```

See also

Topographic position index (tpi)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Standardize [boolean] <put parameter description here>

Default: *True*

Min Radius [number] <put parameter description here>

Default: *0.0*

Max Radius [number] <put parameter description here>

Default: *100.0*

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *75.0*

Outputs

Topographic Position Index [raster] <put output description here>

Console usage

```
processing.runalg('saga:topographicpositionindextpi', dem, standard, radius_min, radius_max, dist
```

See also

Tpi based landform classification

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Min Radius A [number] <put parameter description here>

Default: *0*

Max Radius A [number] <put parameter description here>

Default: *100*

Min Radius B [number] <put parameter description here>

Default: *0*

Max Radius B [number] <put parameter description here>

Default: *1000*

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *75.0*

Outputs

Landforms [raster] <put output description here>

Console usage

```
processing.runalg('saga:tpibasedlandformclassification', dem, radius_a_min, radius_a_max, radius_b_min, radius_b_max, distance_weighting, inverse_distance_weighting_power, inverse_distance_offset, gaussian_and_exponential_weighting_bandwidth)
```

See also

Vector ruggedness measure (vrm)

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Radius (Cells) [number] <put parameter description here>

Default: *1*

Distance Weighting [selection] <put parameter description here>

Options:

- 0 — [0] no distance weighting
- 1 — [1] inverse distance to a power
- 2 — [2] exponential
- 3 — [3] gaussian weighting

Default: *0*

Inverse Distance Weighting Power [number] <put parameter description here>

Default: *1*

Inverse Distance Offset [boolean] <put parameter description here>

Default: *True*

Gaussian and Exponential Weighting Bandwidth [number] <put parameter description here>

Default: *1*

Outputs

Vector Terrain Ruggedness (VRM) [raster] <put output description here>

Console usage

```
processing.runalg('saga:vectorruggednessmeasurevrm', dem, radius, distance_weighting_weighting, d
```

See also

Wind effect

Description

<put algorithm description here>

Parameters

Elevation [raster] <put parameter description here>

Wind Direction [raster] Optional.

<put parameter description here>

Wind Speed [raster] Optional.

<put parameter description here>

Constant Wind Direction [Degree] [number] <put parameter description here>

Default: *135*

Old Version [boolean] <put parameter description here>

Default: *True*

Search Distance [km] [number] <put parameter description here>

Default: *300.0*

Acceleration [number] <put parameter description here>

Default: *1.5*

Use Pyramids [boolean] <put parameter description here>

Default: *True*

Wind Direction Units [selection] <put parameter description here>

Options:

- 0 — [0] radians
- 1 — [1] degree

Default: *0*

Wind Speed Scale Factor [number] <put parameter description here>

Default: *1.0*

Outputs

Wind Effect [raster] <put output description here>

Windward Effect [raster] <put output description here>

Leeward Effect [raster] <put output description here>

Console usage

```
processing.runalg('saga:windeffect', dem, dir, len, dir_const, oldver, maxdist, accel, pyramids, o
```

See also

.

18.7.28 Terrain profiles

Cross profiles

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Lines [vector: line] <put parameter description here>

Profile Distance [number] <put parameter description here>

Default: *10.0*

Profile Length [number] <put parameter description here>

Default: *10.0*

Profile Samples [number] <put parameter description here>

Default: *10.0*

Outputs

Cross Profiles [vector] <put output description here>

Console usage

```
processing.runalg('saga:crossprofiles', dem, lines, dist_line, dist_profile, num_profile, profile)
```

See also

Profile from points table

Description

<put algorithm description here>

Parameters

Grid [raster] <put parameter description here>

Input [table] <put parameter description here>

X [tablefield: any] <put parameter description here>

Y [tablefield: any] <put parameter description here>

Outputs

Result [table] <put output description here>

Console usage

```
processing.runalg('saga:profilefrompointstable', grid, table, x, y, result)
```

See also

Profiles from lines

Description

<put algorithm description here>

Parameters

DEM [raster] <put parameter description here>

Values [multipleinput: rasters] Optional.

<put parameter description here>

Lines [vector: line] <put parameter description here>

Name [tablefield: any] <put parameter description here>

Each Line as new Profile [boolean] <put parameter description here>

Default: *True*

Outputs

Profiles [vector] <put output description here>

Profiles [vector] <put output description here>

Console usage

```
processing.runalg('saga:profilesfromlines', dem, values, lines, name, split, profile, profiles)
```

See also

.

18.8 TauDEM algorithm provider

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.

Catatan: 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](#)

18.8.1 Basic Grid Analysis

D8 Contributing Area

Description

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.

Parameters

D8 Flow Direction Grid [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 Shapefile [vector: point] Optional.

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 [raster] Optional.

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 [boolean] 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.

Default: *True*

Outputs

D8 Contributing Area Grid [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.

Console usage

```
processing.runalg('taudem:d8contributingarea', -p, -o, -wg, -nc, -ad8)
```

See also

D8 Flow Directions

Description

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 Flow Direction Coding:

- 1 — East
- 2 — Northeast
- 3 — North
- 4 — Northwest
- 5 — West
- 6 — Southwest
- 7 — South
- 8 — Southeast

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.

Parameters

Pit Filled Elevation Grid [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.

Outputs

D8 Flow Direction Grid [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 Grid [raster] A grid giving slope in the D8 flow direction. This is measured as drop/distance.

Console usage

```
processing.runalg('taudem:d8flowdirections', -fel, -p, -sd8)
```

See also

D-Infinity Contributing Area

Description

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 ($0, \pi/2, \pi, 3\pi/2$) 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.

Parameters

D-Infinity Flow Direction Grid [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 Shapefile [vector: point] Optional.

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 [raster] Optional.

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).

Check for edge contamination [boolean] 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 imputed 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.

Default: *True*

Outputs

D-Infinity Specific Catchment Area Grid [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.

Console usage

```
processing.runalg('taudem:dinfinitecontributingarea', -ang, -o, -wg, -nc, -sca)
```

See also

D-Infinity Flow Directions

Description

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 ($\pi/4$ 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.

Parameters

Pit Filled Elevation Grid [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.

Outputs

D-Infinity Flow Directions Grid [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 Grid [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.

Console usage

```
processing.runalg('taudem:dinfinityflowdirections', -fel, -ang, -slp)
```

See also

Grid Network

Description

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.

Parameters

D8 Flow Direction Grid [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 Shapefile [vector: point] Optional.

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.

Mask Grid [raster] Optional.

A grid that is used to determine the domain to be analyzed. If the mask grid value \geq 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 [number] This input parameter is used in the calculation mask grid value \geq mask threshold to determine if the grid cell is in the domain to be analyzed.

Default: *100*

Outputs

Longest Upslope Length Grid [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 Grid [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 Grid [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.

Console usage

```
processing.runalg('taudem:gridnetwork', d8_flow_dir_grid, outlets_shape, mask_grid, threshold, lo
```

See also

Pit Remove

Description

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.

Parameters

Elevation Grid [raster] A digital elevation model (DEM) grid to serve as the base input for the terrain analysis and stream delineation.

Outputs

Pit Removed Elevation Grid [raster] A grid of elevation values with pits removed so that flow is routed off of the domain.

Console usage

```
processing.runalg('taudem:pitremove', -z, -fel)
```

See also

.

18.8.2 Specialized Grid Analysis

D8 Distance To Streams

Description

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.

Parameters

D8 Flow Direction Grid [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.

Stream Raster Grid [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.

Threshold [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.

Default: 50

Outputs

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.

Console usage

```
processing.runalg('taudem:d8distancetostreams', -p, -src, -thresh, -dist)
```

See also

D-Infinity Avalanche Runout

Description

Identifies an avalanche’s affected area and the flow path length to each cell in that affected 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.

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 (P_{ik}) 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. $P_{ik} > T$ ($=0.2$ 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

Parameters

D-Infinity Flow Direction Grid [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**”.

Pit Filled Elevation Grid [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.

Default: *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.

Default: *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.

Default: *True*

Outputs

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.

Console usage

```
processing.runalg('taudem:dinfiniteavalancherunout', -ang, -fel, -ass, -thresh, -alpha, -direct, ...)
```

See also

D-Infinity Concentration Limited Accumulation

Description

This function applies to the situation where an unlimited supply of a substance is loaded into flow at a concentration or solubility threshold C_{sol} 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

$$C(x) = C_{sol}, \text{ and } L(x) = C_{sol} Q(x),$$

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:

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.

Parameters

D-Infinity Flow Direction Grid [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.

Outlets shapefile [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.

Default: *1.0*

Check for edge contamination [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.

Default: *True*

Outputs

Concentration Grid [raster] A grid giving the resulting concentration of the compound of interest in the flow.

Console usage

```
processing.runalg('taudem:dinfiniteconcentrationlimitedaccumulation', -ang, -dg, -dm, -q, -o, -cs
```

See also

D-Infinity Decaying Accumulation

Description

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.

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.

Parameters

D-Infinity Flow Direction Grid [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.

Outlets Shapefile [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.

Check for edge contamination [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.

Default: *True*

Outputs

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.

Console usage

```
processing.runalg('taudem:dinfiniteaccumulation', -ang, -dm, -wg, -o, -nc, -dsca)
```

See also

D-Infinity Distance Down

Description

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.

Parameters

D-Infinity Flow Direction Grid [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**”.

Pit Filled Elevation Grid [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.

Stream Raster Grid [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.

Weight Path Grid [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 [selection] 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.

Options:

- 0 — Minimum
- 1 — Maximum
- 2 — Average

Default: 2

Distance Method [selection] 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).

Options:

- 0 — Pythagoras
- 1 — Horizontal
- 2 — Vertical
- 3 — Surface

Default: 1

Check for edge contamination [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.

Default: *True*

Outputs

D-Infinity Drop to Stream Grid [raster] Grid containing the distance to stream calculated using the D-infinity flow model and the statistical and path methods chosen.

Console usage

```
processing.runalg('taudem:dinfinitydistancedown', dinf_flow_dir_grid, pit_filled_grid, stream_grid)
```

See also

D-Infinity Distance Up

Description

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.

Parameters

D-Infinity Flow Direction Grid [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**”.

Pit Filled Elevation Grid [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.

Slope Grid [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 [selection] 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.

Options:

- 0 — Minimum
- 1 — Maximum
- 2 — Average

Default: 2

Distance Method [selection] 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).

Options:

- 0 — Pythagoras

- 1 — Horizontal
- 2 — Vertical
- 3 — Surface

Default: *1*

Proportion Threshold [number] The proportion threshold parameter where only grid cells that contribute flow with a proportion greater than this user specified threshold (τ) is considered to be upslope of any given grid cell. Setting $\tau=0.5$ 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.

Default: *0.5*

Check for edge contamination [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.

Default: *True*

Outputs

D-Infinity Distance Up [raster] Grid containing the distances up to the ridge calculated using the D-Infinity flow model and the statistical and path methods chosen.

Console usage

```
processing.runalg('taudem:dinfinitydistanceup', dinf_flow_dir_grid, pit_filled_grid, slope_grid, ...)
```

See also

D-Infinity Reverse Accumulation

Description

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.

Parameters

D-Infinity Flow Direction Grid [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.

Outputs

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.

Console usage

```
processing.runalg('taudem:dinfinityreverseaccumulation', -ang, -wg, -racc, -dmax)
```

See also

D-Infinity Transport Limited Accumulation - 2

Description

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, T_{cap} . The total supply at a grid cell is calculated as the sum of the transport in from upslope grid cells, T_{in} , plus the local supply contribution, E (e.g. erosion). This function also outputs deposition, D , calculated as total supply minus actual transport.

Here E is the supply. T_{out} at each grid cell becomes T_{in} for downslope grid cells and is reported as Transport limited accumulation (t_{la}). D is deposition (t_{dep}). The function provides the option to evaluate concentration of a compound (contaminant) adhered to the transported substance. This is evaluated as follows:

Where L_{in} is the total incoming compound loading and C_{in} and T_{in} refer to the Concentration and Transport entering from each upslope grid cell.

If

else

where C_s 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,

C_{out} 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.

Parameters

D-Infinity Flow Direction Grid [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.

Transport Capacity Grid [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.

Input Concentration Grid [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.

Outlets Shapefile [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.

Check for edge contamination [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.

Default: *True*

Outputs

Transport Limited Accumulation Grid [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.

Deposition Grid [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.

Output Concentration Grid [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.

Console usage

```
processing.runalg('taudem:dinfiniteytransportlimitedaccumulation2', dinf_flow_dir_grid, supply_grid)
```

See also

D-Infinity Transport Limited Accumulation

Description

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, T_{cap} . The total supply at a grid cell is calculated as the sum of the transport in from upslope grid cells, T_{in} , plus the local supply contribution, E (e.g. erosion). This function also outputs deposition, D , calculated as total supply minus actual transport.

Here E is the supply. T_{out} at each grid cell becomes T_{in} for downslope grid cells and is reported as Transport limited accumulation (t_{la}). D is deposition (t_{dep}). The function provides the option to evaluate concentration of a compound (contaminant) adhered to the transported substance. This is evaluated as follows:

Where L_{in} is the total incoming compound loading and C_{in} and T_{in} refer to the Concentration and Transport entering from each upslope grid cell.

If

else

where C_s 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,

C_{out} 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.

Parameters

D-Infinity Flow Direction Grid [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.

Transport Capacity Grid [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.

Outlets Shapefile [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.

Check for edge contamination [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.

Default: *True*

Outputs

Transport Limited Accumulation Grid [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.

Deposition Grid [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.

Console usage

```
processing.runalg('taudem:dinfiniteynitytransportlimitedaccumulation', dinf_flow_dir_grid, supply_grid,
```

See also

D-Infinity Upslope Dependence

Description

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.

Parameters

D-Infinity Flow Direction Grid [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.

Destination Grid [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.

Outputs

Output Upslope Dependence Grid [raster] A grid quantifying the amount each source point in the domain contributes to the zone defined by the destination grid.

Console usage

```
processing.runalg('taudem:dinfinityupslopedependence', -ang, -dg, -dep)
```

See also

Slope Average Down

Description

This tool computes slope in a D8 downslope direction averaged over a user selected distance. Distance should be specified in horizontal map units.

Parameters

D8 Flow Direction Grid [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.

Pit Filled Elevation Grid [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).

Default: 50

Outputs

Slope Average Down Grid [raster] This output is a grid of slopes calculated in the D8 downslope direction, averaged over the selected distance.

Console usage

```
processing.runalg('taudem:slopeaveragedown', -p, -fel, -dn, -slpd)
```

See also

Slope Over Area Ratio

Description

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.

Parameters

Slope Grid [raster] A grid of slope. This grid can be generated using either 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.

Outputs

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.

Console usage

```
processing.runalg('taudem:slopeoverarearatio', -slp, -sca, -sar)
```

See also

18.8.3 Stream Network Analysis

D8 Extreme Upslope Value

Description

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.

Parameters

D8 Flow Directions Grid [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.

Outlets Shapefile [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.

Check for edge contamination [boolean] A flag that indicates whether the tool should check for edge contamination.

Default: *True*

Use max upslope value [boolean] A flag to indicate whether the maximum or minimum upslope value is to be calculated.

Default: *True*

Outputs

Extreme Upslope Values Grid [raster] A grid of the maximum/minimum upslope values.

Console usage

```
processing.runalg('taudem:d8extremeupslopevalue', -p, -sa, -o, -nc, -min, -ssa)
```

See also

Length Area Stream Source

Description

Creates an indicator grid (1, 0) that evaluates $A \geq (M) (L^y)$ 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 $L \sim A^{0.6}$. However for hillslopes with parallel flow $L \sim A$. So a transition from hillslopes to streams may be represented by $L \sim A^{0.8}$ suggesting identifying grid cells as stream cells if $A > M (L^{(1/0.8)})$.

Parameters

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, $A > (M) (L^y)$, 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 $A > (M) (L^y)$ to determine the transition to a stream.

Threshold [number] The multiplier threshold (M) parameter which is used in the formula: $A > (M) (L^y)$, to identify the beginning of streams.

Default: *0.03*

Exponent [number] The exponent (y) parameter which is used in the formula: $A > (M) (L^y)$, 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).

Default: *1.3*

Outputs

Stream Source Grid [raster] An indicator grid (1,0) that evaluates $A \geq (M)(L^y)$, 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.

Console usage

```
processing.runalg('taudem:lengthareastreamsource', length_grid, contrib_area_grid, threshold, exp
```

See also

Move Outlets To Streams

Description

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.

Parameters

D8 Flow Direction Grid [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.

Stream Raster Grid [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.

Outlets Shapefile [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.

Default: *50*

Outputs

Output Outlet Shapefile [vector] 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.

Console usage

```
processing.runalg('taudem:moveoutletstostreams', -p, -src, -o, -md, -om)
```

See also

Peuker Douglas

Description

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).

Parameters

Elevation Grid [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.

Center Smoothing Weight [number] The center weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Default: *0.4*

Side Smoothing Weight [number] The side weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Default: *0.1*

Diagonal Smoothing Weight [number] The diagonal weight parameter used by a kernel to smooth the DEM before the tool identifies upwardly curved grid cells.

Default: *0.05*

Outputs

Stream Source Grid [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).

Console usage

```
processing.runalg('taudem:peukerdouglas', elevation_grid, center_weight, side_weight, diagonal_we
```

See also

- 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.

Slope Area Combination

Description

Creates a grid of slope-area values = $(S_m)^m (A_n)^n$ 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.

Parameters

Slope Grid [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.

Slope Exponent [number] The slope exponent (m) parameter which will be used in the formula: $(S_m)^m (A_n)^n$, that is used to create the slope-area grid.

Default: *2*

Area Exponent [number] The area exponent (n) parameter which will be used in the formula: $(S_m)^m (A_n)^n$, that is used to create the slope-area grid.

Default: *1*

Outputs

Slope Area Grid [raster] A grid of slope-area values = $(S_m)^m (A_n)^n$ calculated from the slope grid, specific catchment area grid, m slope exponent parameter, and n area exponent parameter.

Console usage

```
processing.runalg('taudem:slopeareacombination', slope_grid, area_grid, slope_exponent, area_exponent)
```

See also

Stream Definition By Threshold

Description

Operates on any grid and outputs an indicator (1, 0) grid identifying cells with input values \geq 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 \geq 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
```

Parameters

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.

Threshold [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 \geq this threshold.

Default: *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 \geq 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.

Outputs

Stream Raster Grid [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.

Console usage

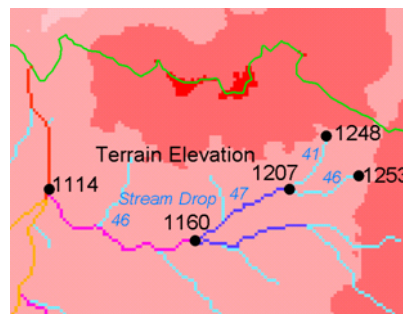
```
processing.runalg('taudem:streamdefinitionbythreshold', -ssa, -thresh, -mask, -src)
```

See also

Stream Drop Analysis

Description

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 Broscoc (1959). For the science behind using this to determine a stream delineation threshold, see Tarboton et al. (1991, 1992), Tarboton and Ames (2001).

Parameters

D8 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 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 Flow Direction Grid [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.

Pit Filled Elevation Grid [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 is 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.

Outlets Shapefile [vector: point] A point shapefile defining the outlets upstream of which drop analysis is performed.

Minimum Threshold [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).

Default: 5

Maximum Threshold [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).

Default: 500

Number of Threshold Values [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).

Default: 10

Spacing for Threshold Values [selection] This parameter indicates whether logarithmic or linear spacing should be used when looking for possible threshold values using drop analysis.

Options:

- 0 — Logarithmic
- 1 — Linear

Default: 0

Outputs

D-Infinity Drop to Stream Grid [file] This is a comma delimited text file with the following header line:

```
:: 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).

Console usage

```
processing.runalg('taudem:streamdropanalysis', d8_contrib_area_grid, d8_flow_dir_grid, pit_filled,
```

See also

- Broscoc, 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, <http://www.engineering.usu.edu/dtarb/asce2001.pdf>.

Stream Reach and Watershed

Description

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.

Parameters

Pit Filled Elevation Grid [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 Flow Direction Grid [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.

Stream Raster Grid [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 delineate 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 separate watershed is delineated for each stream reach. Default is *False* (separate watershed).

Default: *False*

Outputs

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] 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
- Order — Strahler Stream Order
- 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)

Network Coordinates [file] This output is a text file that contains the coordinates and attributes of points along the stream network. Columns are as follows:

- X coordinate
- Y Coordinate
- Distance along channels to the downstream end of a terminal link
- Elevation
- Contributing area

Console usage

```
processing.runalg('taudem:streamreachandwatershed', -fel, -p, -ad8, -src, -o, -sw, -ord, -w, -net,
```

See also

.

Print Composer

With the Print Composer you can create nice maps and atlases that can be printed or saved as PDF-file, an image or an SVG-file. This is a powerfull way to share geographical information produced with QGIS that can be included in reports or published.

The Print Composer 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 and position each element and adjust the properties to create your layout. The layout can be printed or exported to image formats, PostScript, PDF or to SVG (export to SVG is not working properly with some recent Qt4 versions; you should try and check individually on your system). 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. See a list of tools in [table_composer_1](#):






















































Icon	Purpose	Icon	Purpose
	Save Project		New Composer
	Duplicate Composer		Composer Manager
	Load from template		Save as template
	Print or export as PostScript		Export to an image format
	Export print composition to SVG		Export as PDF
	Revert last change		Restore last change
	Zoom to full extent		Zoom to 100%
	Zoom in		Zoom out
	Refresh View		Zoom to specific region
	Pan		Move content within an item
	Select/Move item in print composition		Add image to print composition
	Add new map from QGIS map canvas		Add new legend to print composition
	Add label to print composition		Add basic shape to print composition
	Add scale bar to print composition		Add attribute table to print composition
	Add arrow to print composition		Ungroup items of print composition
	Add an HTML frame		Unlock All items
	Group items of print composition		Lower selected items
	Lock Selected Items		Move selected items to bottom
	Raise selected items		Align selected items right
	Move selected items to top		Align selected items center vertical
	Align selected items left		Align selected items bottom
	Align selected items center		First Feature
	Align selected items top		Next Feature
	Preview Atlas		Print Atlas
	Previous Feature		Atlas Settings
	Last feature		
	Export Atlas as Image		

Table Composer 1: Print Composer Tools

All Print Composer tools are available in menus and as icons in a toolbar. The toolbar can be switched off and on using the right mouse button over the toolbar.

19.1 First steps

19.1.1 Open a new Print Composer Template

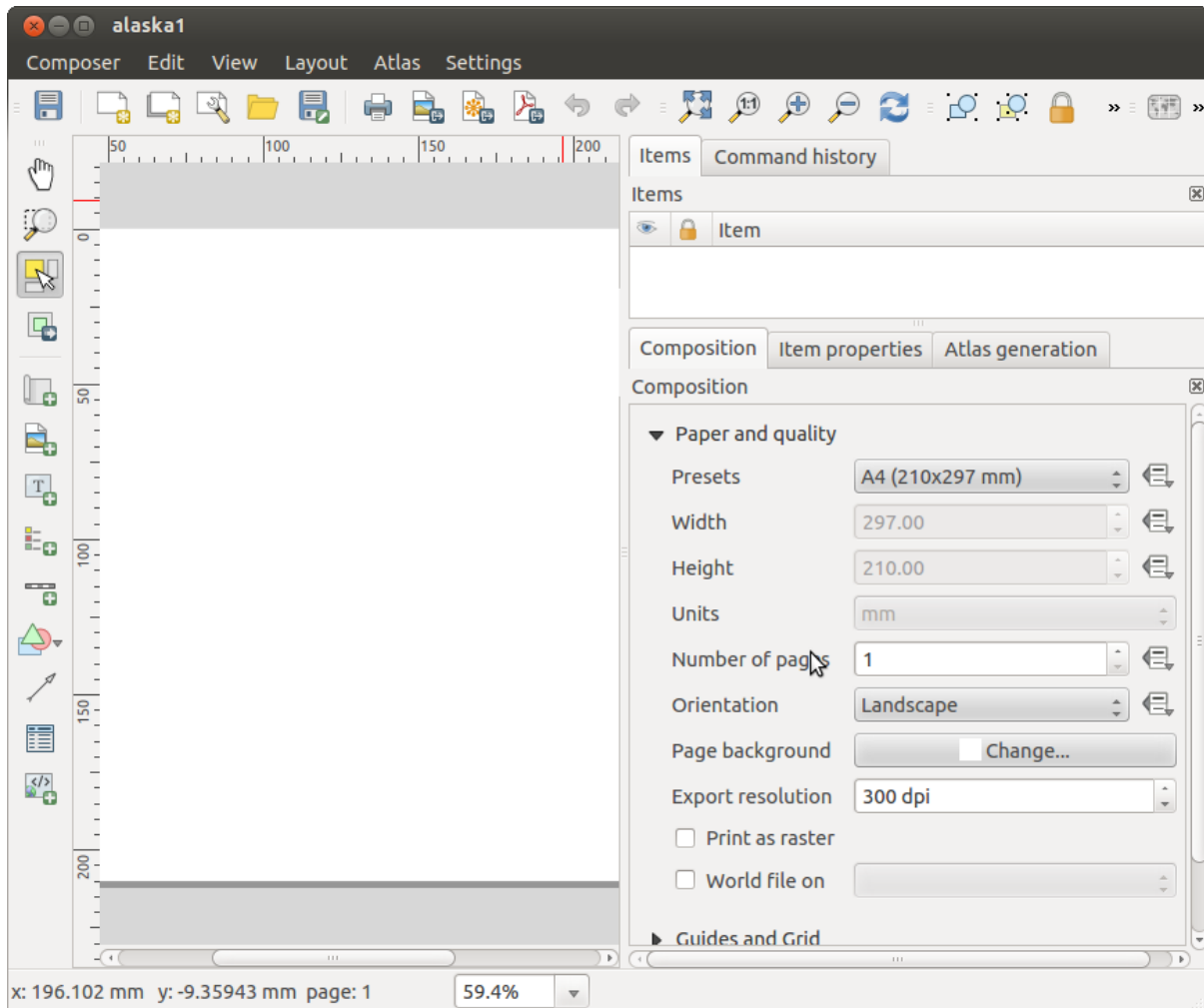
Before you start to work with the Print Composer, you need to load some raster and vector layers in the QGIS map canvas and adapt their properties to suit your own convenience. After everything is rendered and symbolized to your liking, click the  New Print Composer icon in the toolbar or choose *File* → *New Print Composer*. You will

be prompted to choose a title for the new Composer.

19.1.2 Overview of the Print Composer

Opening the Print Composer provides you with a blank canvas that represents the paper surface when using the print option. Initially you find buttons on the left beside the canvas to add map composer items; the current QGIS map canvas, text labels, images, legends, scale bars, basic shapes, arrows, attribute tables and HTML frames. In this toolbar you also find toolbar buttons to navigate, zoom in on an area and pan the view on the composer and toolbar buttons to select a map composer item and to move the contents of the map item.



Figure_composer_overview shows the initial view of the Print Composer before any elements are added.



Gambar 19.1: Print Composer 

On the right beside the canvas you find two panels. The upper panel holds the tabs *Items* and *Command History* and the lower panel holds the tabs *Composition*, *Item properties* and *Atlas generation*.

- The *Items* tab provides a list of all map composer items added to the canvas.
- The *Command history* tab displays a history of all changes applied to the Print Composer layout. With a mouse click, it is possible to undo and redo layout steps back and forth to a certain status.
- The *Composition* tab allows you to set paper size, orientation, the page background, number of pages and print quality for the output file in dpi. Furthermore, you can also activate the *Print as raster* checkbox. This means all items will be converted to raster before printing or saving as PostScript or PDF. In this tab, you can also customize settings for grid and smart guides.








- The *Item Properties* tab displays the properties for the selected item. Click the  Select/Move item icon to select an item (e.g., legend, scale bar or label) on the canvas. Then click the *Item Properties* tab and customize the settings for the selected item.
- The *Atlas generation* tab allows you to enable the generation of an atlas for the current Composer and gives access to its parameters.
- Finally, you can save your print composition with the  Save Project button.

In the bottom part of the Print Composer window, you can find a status bar with mouse position, current page number and a combo box to set the zoom level.

You can add multiple elements to the Composer. It is also possible to have more than one map view or legend or scale bar in the Print Composer canvas, on one or several pages. Each element has its own properties and, in the case of the map, its own extent. If you want to remove any elements from the Composer canvas you can do that with the *Delete* or the *Backspace* key.

Navigation tools





To navigate in the canvas layout, the Print Composer provides some general tools:


-  Zoom in
-  Zoom out
-  Zoom to full extent
-  Zoom to 100%
-  Refresh the view (if you find the view in an inconsistent state)
-  Pan composer
-  Marquee zoom mode (zoom to a specific region of the Composer)

You can change the zoom level also using the mouse wheel or the combo box in the status bar. If you need to switch to pan mode while working in the Composer area, you can hold the *Spacebar* or the mouse wheel. With *Ctrl+Spacebar*, you can temporarily switch to marquee zoom mode, and with *Ctrl+Shift+Spacebar*, to zoom out mode.

19.1.3 Sample Session

To demonstrate how to create a map please follow the next instructions.

1. On the left site, select the  Add new map toolbar button and draw a rectangle on the canvas holding down the left mouse button. Inside the drawn rectangle the QGIS map view to the canvas.
2. Select the  Add new scalebar toolbar button and place the map item with the left mouse button on the Print Composer canvas. A scalebar will be added to the canvas.
3. Select the  Add new legend toolbar button and draw a rectangle on the canvas holding down the left mouse button. Inside the drawn rectangle the legend will be drawn.
4. Select the  Select/Move item icon to select the map on the canvas and move it a bit.
5. While the map item is still selected you can also change the size of the map item. Click while holding down the left mouse button, in a white little rectangle in one of the corners of the map item and draw it to a new location to change its size.

6. Click the *Item Properties* tab on the left lower panel and find the setting for the orientation. Change it the value of the setting *Map orientation* to '15.00° '. You should see the orientation of the map item change.
7. Finally, you can save your print composition with the  Save Project button.

19.1.4 Print Composer Options

From *Settings* → *Composer Options* you can set some options that will be used as default during your work.

- *Compositions defaults* let you specify the default font to use.
- With *Grid appearance*, you can set the grid style and its color.
- *Grid defaults* defines spacing, offset and tolerance of the grid. There are three types of grid: **Dots**, **Solid lines** and **Crosses**.
- *Guide defaults* defines the tolerance for the guides.

19.1.5 Composition tab — General composition setup

In the *Composition* tab, you can define the global settings of your composition.

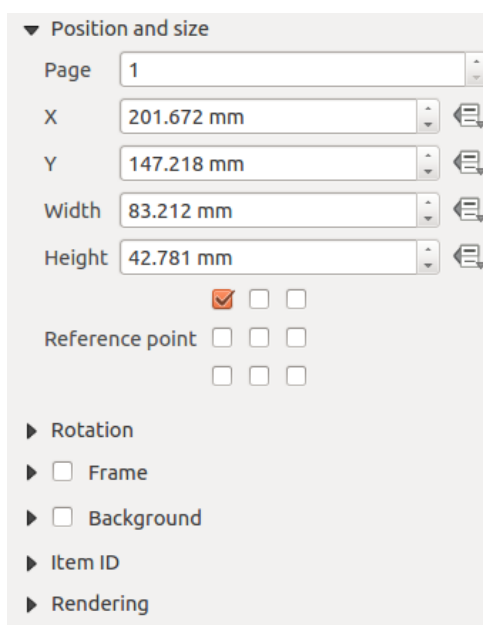
- You can choose one of the *Presets* for your paper sheet, or enter your custom *width* and *height*.
- Composition can now be divided into several pages. For instance, a first page can show a map canvas, and a second page can show the attribute table associated with a layer, while a third one shows an HTML frame linking to your organization website. Set the *Number of pages* to the desired value. You can choose the page *Orientation* and its *Exported resolution*. When checked, *print as raster* means all elements will be rasterized before printing or saving as PostScript or PDF.
- *Grid* lets you customize grid settings like *spacings*, *offsets* and *tolerance* to your need.
- In *Snap to alignments*, you can change the *Tolerance*, which is the maximum distance below which an item is snapped to smart guides.

Snap to grid and/or to smart guides can be enabled from the *View* menu. In this menu, you can also hide or show the grid and smart guides.

19.1.6 Composer items common options


Composer items have a set of common properties you will find on the bottom of the *Item Properties* tab: Position and size, Rotation, Frame, Background, Item ID and Rendering (See [figure_composer_common_1](#)).

- The *Position and size* dialog lets you define size and position of the frame that contains the item. You can also choose which *Reference point* will be set at the **X** and **Y** coordinates previously defined.
- The *Rotation* sets the rotation of the element (in degrees).
- The *Frame* shows or hides the frame around the label. Click on the **[Color]** and **[Thickness]** buttons to adjust those properties.
- The *Background* enables or disables 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 also be adjusted through the **alpha** field.
- Use the *Item ID* to create a relationship to other Print Composer items. This is used with QGIS server and any potential web client. You can set an ID on an item (e.g., a map and 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 what items and which IDs are available in a layout.
- *Rendering* mode can be selected in the option field. See [Rendering_Mode](#).



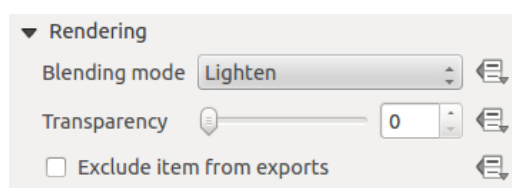
Gambar 19.2: Common Item properties Dialogs 

Catatan:


- If you checked *Use live-updating color chooser dialogs* in the QGIS general options, the color button will update as soon as you choose a new color from **Color Dialog** windows. If not, you need to close the **Color Dialog**.
- The  *Data defined override* icon next to a field means that you can associate the field with data in the map item or use expressions. These are particularly helpful with atlas generation (See [atlas_data_defined_overrides](#)).

19.2 Rendering mode

QGIS now allows advanced rendering for Composer items just like vector and raster layers.




Gambar 19.3: Rendering mode 

- *Transparency* : You can make the underlying item in the Composer 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 the menu beside the slider.
- *Exclude item from exports*: You can decide to make an item not visible in all exports. After activating this checkbox, the item will not be included in PDF's, prints etc..
- *Blending mode*: You can achieve special rendering effects with these tools that you previously only may know from graphics programs. The pixels of your overlaying and underlaying items are mixed through the settings described below.

- Normal: This is the standard blend mode, which uses the alpha channel of the top pixel to blend with the pixel beneath it; the colors aren't mixed.
- Lighten: This selects the maximum of each component from the foreground and background pixels. Be aware that the results tend to be jagged and harsh.
- Screen: Light pixels from the source are painted over the destination, while dark pixels are not. This mode is most useful for mixing the texture of one layer with another layer (e.g., you can use a hillshade to texture another layer).
- Dodge: Dodge will brighten and saturate underlying pixels based on the lightness of the top pixel. So, 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: This blend mode simply adds pixel values of one layer with pixel values of the other. In case of values above 1 (as in the case of RGB), white is displayed. This mode is suitable for highlighting features.
- Darken: This creates a resultant pixel that retains the smallest components of the foreground and background pixels. Like lighten, the results tend to be jagged and harsh.
- Multiply: Here, the numbers for each pixel of the top layer are multiplied with the numbers for the corresponding pixel of the bottom layer. The results are darker pictures.
- Burn: Darker colors in the top layer cause the underlying layers to darken. Burn can be used to tweak and colorise underlying layers.
- Overlay: This mode combines the multiply and screen blending modes. In the resulting picture, light parts become lighter and dark parts become darker.
- Soft light: This is very similar to overlay, but instead of using multiply/screen it uses color burn/dodge. This mode is supposed to emulate shining a soft light onto an image.
- Hard light: Hard light is very similar to the overlay mode. It's supposed to emulate projecting a very intense light onto an image.
- Difference: Difference subtracts the top pixel from the bottom pixel, or the other way around, to always get a positive value. Blending with black produces no change, as the difference with all colors is zero.
- Subtract: This blend mode simply subtracts pixel values of one layer with pixel values of the other. In case of negative values, black is displayed.


19.3 Composer Items




19.3.1 The Map item

Click on the  toolbar button in the Print Composer toolbar to add the QGIS map canvas. Now, drag a rectangle onto the Composer canvas with the left mouse button to add the map. To display the current map, you can choose between three different modes in the map *Item Properties* tab:

- **Rectangle** is the default setting. It only displays an empty box with a message 'Map will be printed here'.
- **Cache** renders the map in the current screen resolution. If you zoom the Composer window in or out, the map is not rendered again but the image will be scaled.
- **Render** means that if you zoom the Composer window in or out, the map will be rendered again, but for space reasons, only up to a maximum resolution.

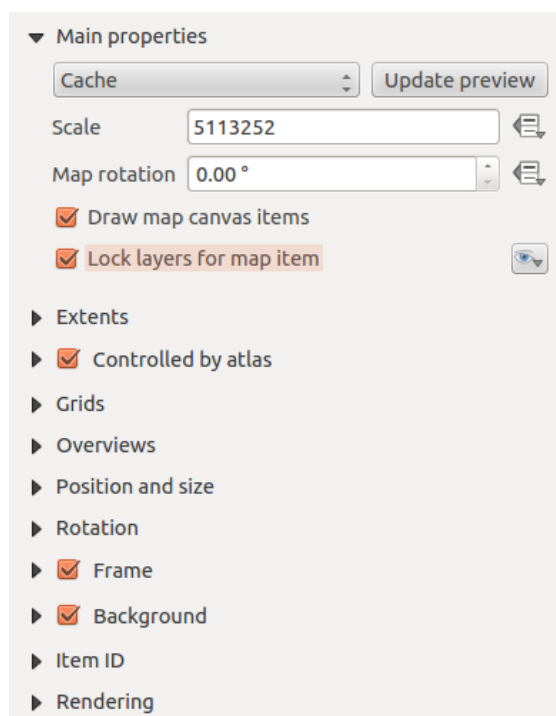
Cache is the default preview mode for newly added Print Composer maps.

You can resize the map element by clicking on the  button, selecting the element, and dragging one of the blue handles in the corner of the map. With the map selected, you can now adapt more properties in the map *Item Properties* tab.

To move layers within the map element, select the map element, click the  icon and move the layers within the map item frame with the left mouse button. After you have found the right place for an item, you can lock the item position within the Print Composer canvas. Select the map item and use the toolbar  or the *Items* tab to Lock the item. A locked item can only be selected using the *Items* tab. Once selected you can use the *Items* tab to unlock individual items. The  icon will unlock all locked composer items.

Main properties




The *Main properties* dialog of the map *Item Properties* tab provides the following functionalities (see [figure_composer_map_1](#)):



Gambar 19.4: Map Item properties Tab 

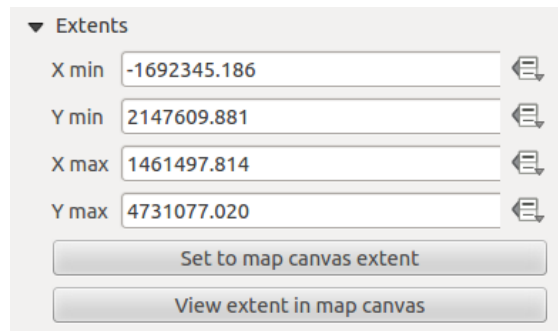
- The **Preview** area allows you to define the preview modes 'Rectangle', 'Cache' and 'Render', as described above. If you change the view on the QGIS map canvas by changing vector or raster properties, you can update the Print Composer view by selecting the map element in the Print Composer and clicking the **[Update preview]** button.
- The field *Scale* sets a manual scale.
- The field *Rotation* allows you to rotate the map element content clockwise in degrees. Note that a coordinate frame can only be added with the default value 0.
- *Draw map canvas items* lets you show annotations that may be placed on the map canvas in the main QGIS window.
- You can choose to lock the layers shown on a map item. Check *Lock layers for map item*. After this is checked, any layer that would be displayed or hidden in the main QGIS window will not appear or be

hidden in the map item of the Composer. But style and labels of a locked layer are still refreshed according to the main QGIS interface.

- The  button allows you to add quickly all the presets views you have prepared in QGIS. Clicking on the  button you will see the list of all the preset views: just select the preset you want to display. The map canvas will automatically lock the preset layers by enabling the *Lock layers for map item*: if you want to unselect the preset, just uncheck the and press on the  button. See *Legenda Peta* to find out how to create presets views.

Extents

The *Extents* dialog of the map item tab provides the following functionalities (see [figure_composer_map_2](#)):



Gambar 19.5: Map Extents Dialog 

- The **Map extents** area allows you to specify the map extent using X and Y min/max values and by clicking the **[Set to map canvas extent]** button. This button sets the map extent of the composer map item to the extent of the current map view in the main QGIS application. The button **[View extent in map canvas]** does exactly the opposite, it updates the extent of the map view in the QGIS application to the extent of the composer map item.

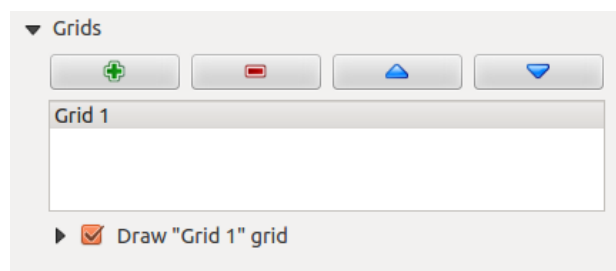
If you change the view on the QGIS map canvas by changing vector or raster properties, you can update the Print Composer view by selecting the map element in the Print Composer and clicking the **[Update preview]** button in the map *Item Properties* tab (see [figure_composer_map_1](#)).

Grids

The *Grids* dialog of the map *Item Properties* tab provides the the possibility to add several grids to a map item.

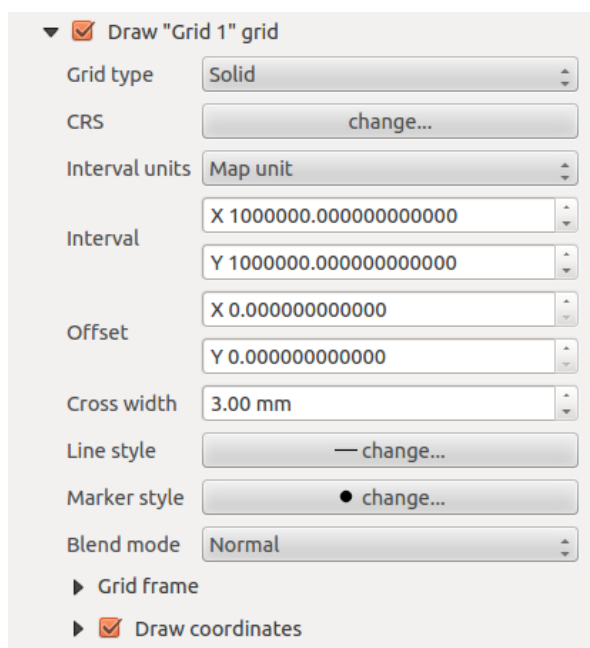
- With the plus and minus button you can add or remove a selected grid.
- With the up and down button you can move a grid in the list and set the drawing priority.

When you double click on the added grid you can give it another name.



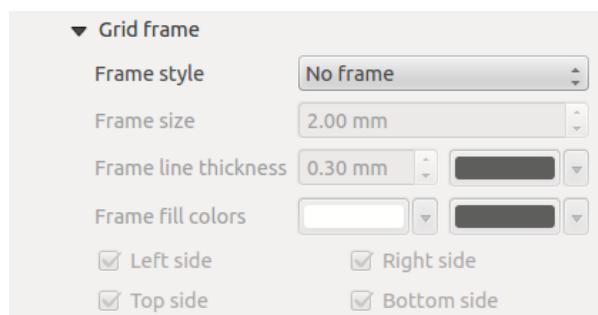
Gambar 19.6: Map Grids Dialog 

After you have added a grid, you can activate the checkbox *Show grid* to overlay a grid onto the map element. Expand this option to provides a lot of configuration options, see [Figure_composer_map_4](#).



Gambar 19.7: Draw Grid Dialog 

As grid type, you can specify to use a solid line or cross. Symbology of the grid can be chosen. See section [Rendering_Mode](#). Furthermore, you can define an interval in the X and Y directions, an X and Y offset, and the width used for the cross or line grid type.

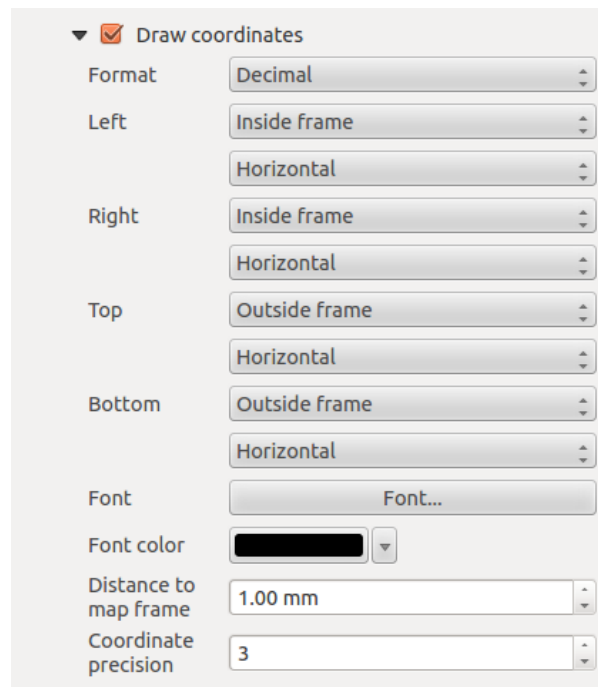



Gambar 19.8: Grid Frame Dialog 

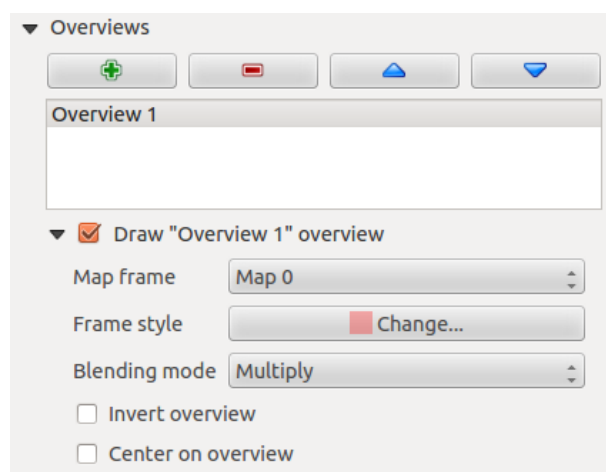
- There are different options to style the frame that holds the map. Following options are available: No Frame, Zebra, Interior ticks, Exterior ticks, Interior and Exterior ticks and Lineborder.
- Advanced rendering mode is also available for grids (see section [Rendering_mode](#)).
- The *Draw coordinates* checkbox allows you to add coordinates to the map frame. The annotation can be drawn inside or outside the map frame. The annotation direction can be defined as horizontal, vertical, horizontal and vertical, or boundary direction, for each border individually. Units can be in meters or in degrees. Finally, you can define the grid color, the annotation font, the annotation distance from the map frame and the precision of the drawn coordinates.

Overviews

The *Overviews* dialog of the map *Item Properties* tab provides the following functionalities:



Gambar 19.9: Grid Draw Coordinates dialog 



Gambar 19.10: Map Overviews Dialog 

You can choose to create an overview map, which shows the extents of the other map(s) that are available in the composer. First you need to create the map(s) you want to include in the overview map. Next you create the map you want to use as the overview map, just like a normal map.


- With the plus and minus button you can add or remove an overview.
- With the up and down button you can move an overview in the list and set the drawing priority.

Open *Overviews* and press the green plus icon-button to add an overview. Initially this overview is named 'Overview 1' (see [Figure_composer_map_7](#)). You can change the name when you double-click on the overview item in the list named 'Overview 1' and change it to another name.

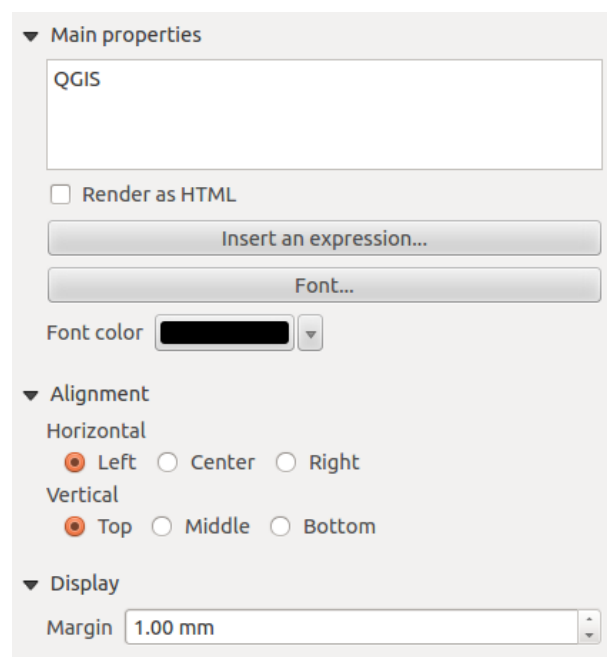
When you select the overview item in the list you can customize it.

- The *Draw "<name_overview>" overview* needs to be activated to draw the extent of selected map frame.
- The *Map frame* combo list can be used to select the map item whose extents will be drawn on the present map item.
- The *Frame Style* allows you to change the style of the overview frame.
- The *Blending mode* allows you to set different transparency blend modes. See [Rendering_Mode](#).
- The *Invert overview* creates a mask around the extents when activated: the referenced map extents are shown clearly, whereas everything else is blended with the frame color.
- The *Center on overview* puts the extent of the overview frame in the center of the overview map. You can only activate one overview item to center, when you have added several overviews.

19.3.2 The Label item

To add a label, click the  *Add label* icon, place the element with the left mouse button on the Print Composer canvas and position and customize its appearance in the label *Item Properties* tab.

The *Item Properties* tab of a label item provides the following functionality for the label item (see [Figure_composer_label](#)):



Gambar 19.11: Label Item properties Tab 


Main properties

- The main properties dialog is where the text (HTML or not) or the expression needed to fill the label is added to the Composer canvas.
- Labels can be interpreted as HTML code: check *Render as HTML*. You can now insert a URL, a clickable image that links to a web page or something more complex.
- You can also insert an expression. Click on **[Insert an expression]** to open a new dialog. Build an expression by clicking the functions available in the left side of the panel. Two special categories can be useful, particularly associated with the atlas functionality: geometry functions and records functions. At the bottom, a preview of the expression is shown.
- Define *Font* by clicking on the **[Font...]** button or a *Font color* selecting a color using the color selection tool.

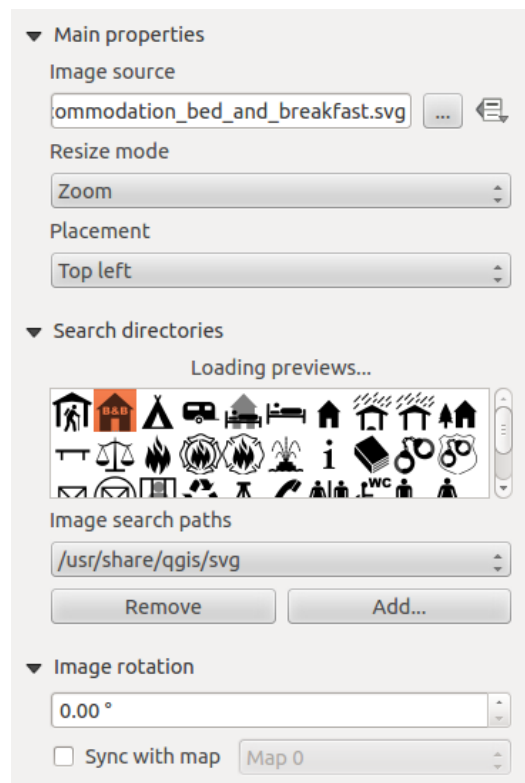
Alignment and Display


- You can define the horizontal and vertical alignment in the *Alignment* zone.
- In the **Display** tag, you can define a margin in mm. This is the margin from the edge of the composer item.

19.3.3 The Image item



To add an image, click the  Add image icon, place the element with the left mouse button on the Print Composer canvas and position and customize its appearance in the image *Item Properties* tab.

The image *Item Properties* tab provides the following functionalities (see [figure_composer_image_1](#)):



Gambar 19.12: Image Item properties Tab 

You first have to select the image you want to display. There are several ways to set the *image source* in the **Main properties** area.

1. Use the browse button  of *image source* to select a file on your computer using the browse dialog. The browser will start in the SVG-libraries provided with QGIS. Besides SVG, you can also select other image formats like .png or .jpg.
2. You can enter the source directly in the *image source* text field. You can even provide a remote URL-address to an image.
3. From the **Search directories** area you can also select an image from *loading preview..* to set the image source.
4. Use the data defined button  to set the image source from a record or using a regular expression.


With the *Resize mode* option, you can set how the image is displayed when the frame is changed, or choose to resize the frame of the image item so it matches the original size of the image.


You can select one of the following modes:

- Zoom: Enlarges the image to the frame while maintaining aspect ratio of picture.
- Stretch: Stretches image to fit inside the frame, ignores aspect ratio.
- Clip: Use this mode for raster images only, it sets the size of the image to original image size without scaling and the frame is used to clip the image, so only the part of the image inside the frame is visible.
- Zoom and resize frame: Enlarges image to fit frame, then resizes frame to fit resultant image.
- Resize frame to image size: Sets size of frame to match original size of image without scaling.

Selected resize mode can disable the item options 'Placement' and 'Image rotation'. The *Image rotation* is active for the resize mode 'Zoom' and 'Clip'.


With *Placement* you can select the position of the image inside it's frame. The **Search directories** area allows you to add and remove directories with images in SVG format to the picture database. A preview of the pictures found in the selected directories is shown in a pane and can be used to select and set the image source.

Images can be rotated with the *Image rotation* field. Activating the  *Sync with map* checkbox synchronizes the rotation of a picture in the QGIS map canvas (i.e., a rotated north arrow) with the appropriate Print Composer image.

It is also possible to select a north arrow directly. If you first select a north arrow image from **Search directories** and then use the browse button  of the field *Image source*, you can now select one of the north arrow from the list as displayed in [figure_composer_image_2](#).

Catatan: Many of the north arrows do not have an 'N' added in the north arrow, this is done on purpose for languages that do not use an 'N' for North, so they can use another letter.

19.3.4 The Legend item

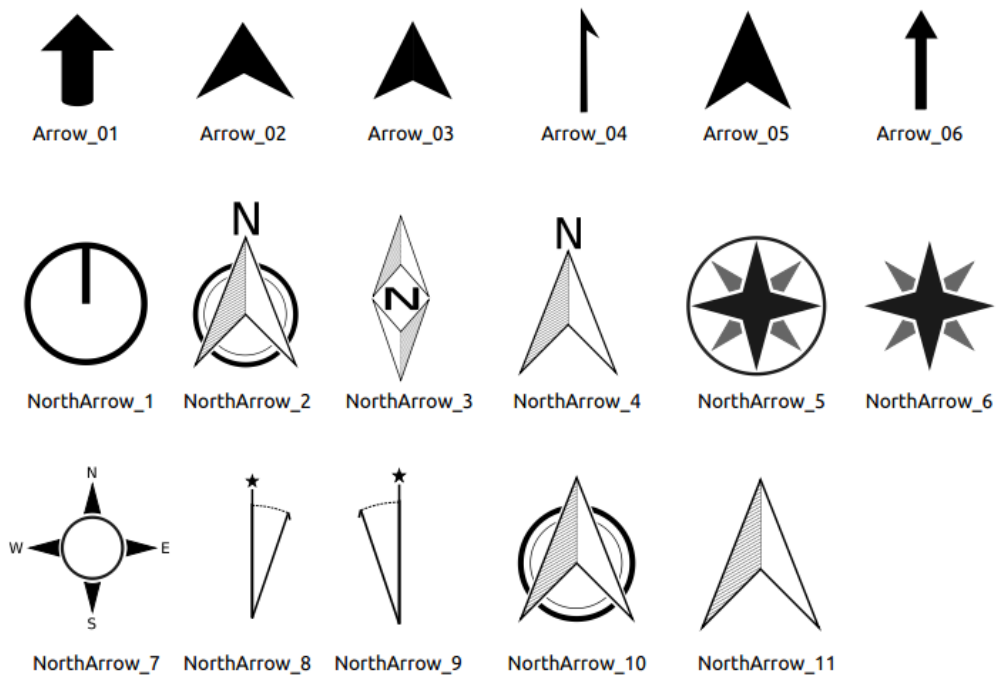
To add a map legend, click the  Add new legend icon, place the element with the left mouse button on the Print Composer canvas and position and customize the appearance in the legend *Item Properties* tab.

The *Item properties* of a legend item tab provides the following functionalities (see [figure_composer_legend_1](#)):

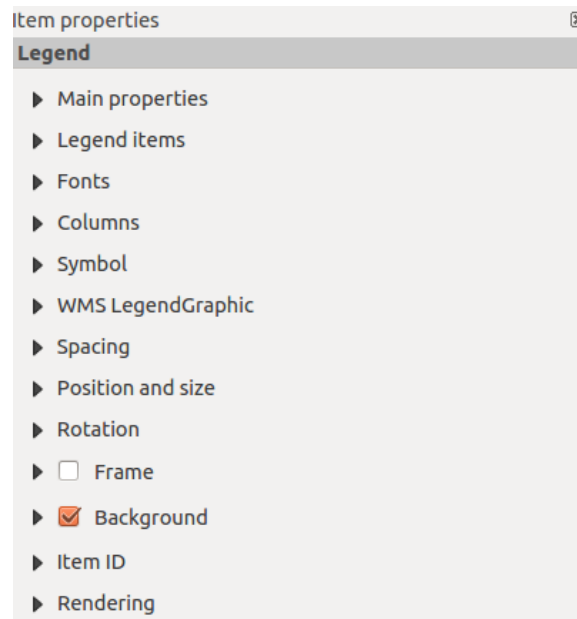
Main properties

The *Main properties* dialog of the legend *Item Properties* tab provides the following functionalities (see [figure_composer_legend_2](#)):

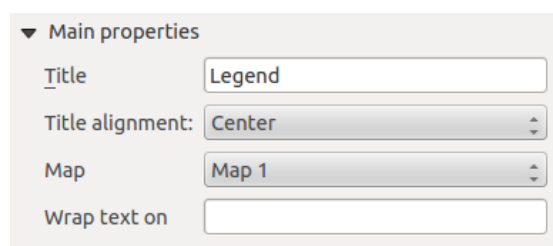
In Main properties you can:



Gambar 19.13: North arrows available for selection in provided SVG library



Gambar 19.14: Legend Item properties Tab 

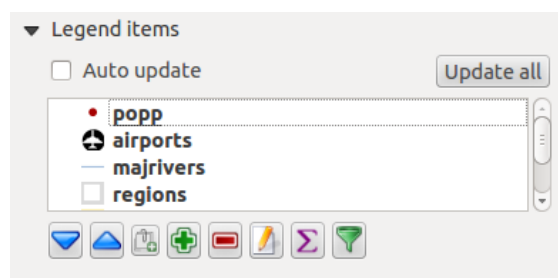


Gambar 19.15: Legend Main properties Dialog 

- Change the title of the legend.
- Set the title alignment to Left, Center or Right.
- You can choose which *Map* item the current legend will refer to in the select list.
- You can wrap the text of the legend title on a given character.

Legend items

The *Legend items* dialog of the legend *Item Properties* tab provides the following functionalities (see [figure_composer_legend_3](#)):



Gambar 19.16: Legend Legend Items Dialog 

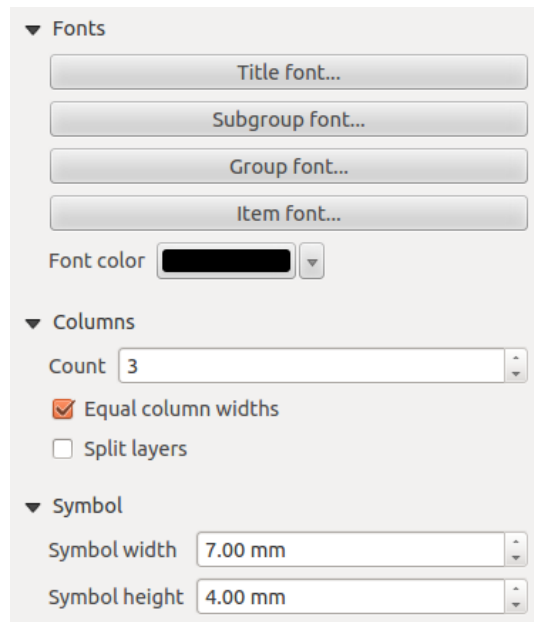
- The legend will be updated automatically if *Auto-update* is checked. When *Auto-update* is unchecked this will give you more control over the legend items. The icons below the legend items list will be activated.
- The legend items window lists all legend items and allows you to change item order, group layers, remove and restore items in the list, edit layer names and add a filter.
 - The item order can be changed using the [**Up**] and [**Down**] buttons or with ‘drag-and-drop’ functionality. The order can not be changed for WMS legend graphics.
 - Use the [**Add group**] button to add a legend group.
 - Use the [**plus**] and [**minus**] button to add or remove layers.
 - The [**Edit**] button is used to edit the layer-, groupname or title, first you need to select the legend item.
 - The [**Sigma**] button adds a feature count for each vector layer.
 - Use the [**filter**] button the filter the legend by map content, only the legend items visible in the map will be listed in the legend.


After changing the symbology in the QGIS main window, you can click on [**Update**] to adapt the changes in the legend element of the Print Composer.

Fonts, Columns, Symbol

The *Fonts*, *Columns* and *Symbol* dialogs of the legend *Item Properties* tab provide the following functionalities (see [figure_composer_legend_4](#)):

- You can change the font of the legend title, group, subgroup and item (layer) in the legend item. Click on a category button to open a **Select font** dialog.
- You provide the labels with a **Color** using the advanced color picker, however the selected color will be given to all font items in the legen..
- Legend items can be arranged over several columns. Set the number of columns in the *Count* field.
 - *Equal column widths* sets how legend columns should be adjusted.

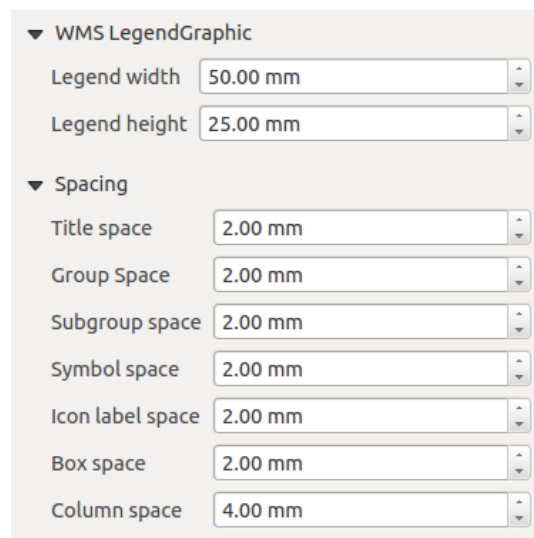



Gambar 19.17: Legend Fonts, Columns, Symbol and Spacing Dialogs 

- The *Split layers* option allows a categorized or a graduated layer legend to be divided between columns.
- You can change the width and height of the legend symbol in this dialog.

WMS legendGraphic and Spacing

The *WMS legendGraphic* and *Spacing* dialogs of the legend *Item Properties* tab provide the following functionalities (see [figure_composer_legend_5](#)):




Gambar 19.18: WMS legendGraphic Dialogs 

When you have added a WMS layer and you insert a legend composer item, a request will be send 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.

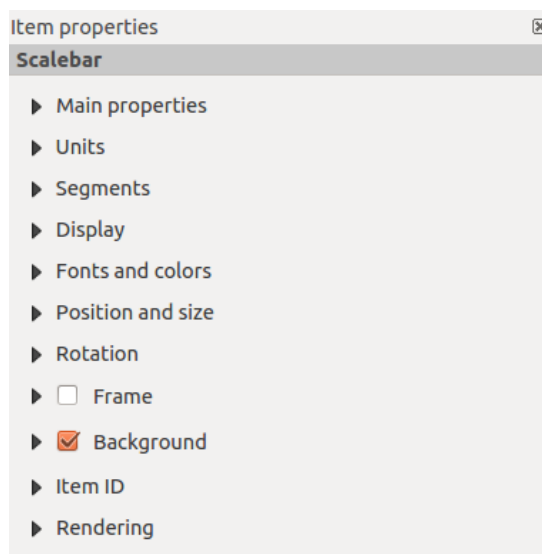
WMS legendGraphic is used to be able to adjust the *Legend width* and the *legend height* of the WMS legend raster image.

Spacing around title, group, subgroup, symbol, icon label, box space or column space can be customized through this dialog.

19.3.5 The Scale Bar item

To add a scale bar, click the  Add new scalebar icon, place the element with the left mouse button on the Print Composer canvas and position and customize the appearance in the scale bar *Item Properties* tab.

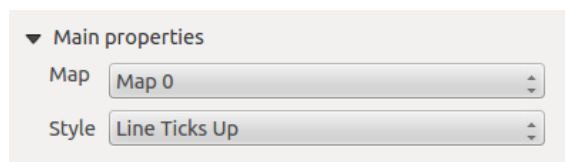
The *Item properties* of a scale bar item tab provides the following functionalities (see [figure_composer_scalebar_1](#)):



Gambar 19.19: Scale Bar Item properties Tab 

Main properties

The *Main properties* dialog of the scale bar *Item Properties* tab provides the following functionalities (see [figure_composer_scalebar_2](#)):

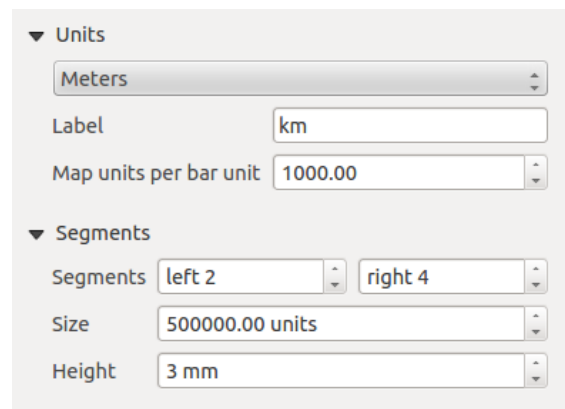



Gambar 19.20: Scale Bar Main properties Dialog 

- First, choose the map the scale bar will be attached to.
- Then, choose the style of the scale bar. Six styles are available:
 - **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 (i.e., 1:50000).

Units and Segments

The *Units* and *Segments* dialogs of the scale bar *Item Properties* tab provide the following functionalities (see [figure_composer_scalebar_3](#)):



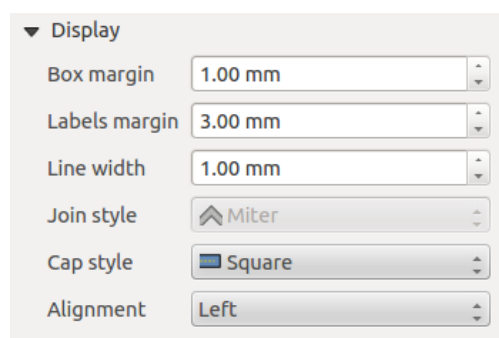
Gambar 19.21: Scale Bar Units and Segments Dialogs 

In these two dialogs, you can set how the scale bar will be represented.

- Select the map units used. There are four possible choices: **Map Units** is the automated unit selection; **Meters**, **Feet** or **Nautical Miles** force unit conversions.
- The *Label* field defines the text used to describe the units of the scale bar.
- The *Map units per bar unit* allows you to fix the ratio between a map unit and its representation in the scale bar.
- You can define how many *Segments* will be drawn on the left and on the right side of the scale bar, and how long each segment will be (*Size* field). *Height* can also be defined.

Display

The *Display* dialog of the scale bar *Item Properties* tab provide the following functionalities (see [figure_composer_scalebar_4](#)):



Gambar 19.22: Scale Bar Display 

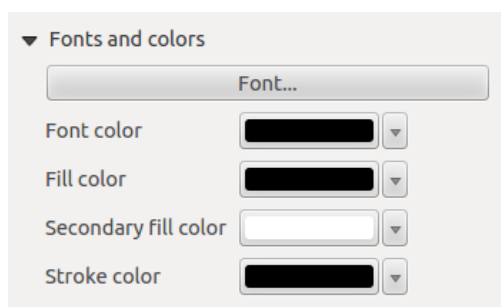
You can define how the scale bar will be displayed in its frame.

- *Box margin* : space between text and frame borders
- *Labels margin* : space between text and scale bar drawing
- *Line width* : line width of the scale bar drawing

- *Join style* : Corners at the end of scalebar in style Bevel, Rounded or Square (only available for Scale bar style Single Box & Double Box)
- *Cap style* : End of all lines in style Square, Round or Flat (only available for Scale bar style Line Ticks Up, Down and Middle)
- *Alignment* : Puts text on the left, middle or right side of the frame (works only for Scale bar style Numeric)

Fonts and colors

The *Fonts and colors* dialog of the scale bar *Item Properties* tab provide the following functionalities (see [figure_composer_scalebar_5](#)):





Gambar 19.23: Scale Bar Fonts and colors Dialogs 

You can define the fonts and colors used for the scale bar.

- Use the [**F**ont] button to set the font
- *Font color*: set the font color
- *Fill color*: set the first fill color
- *Secondary fill color*: set the second fill color
- *Stroke color*: set the color of the lines of the Scale Bare

Fill colors are only used for scale box styles Single Box and Double Box. To select a color you can use the list option using the dropdown arrow to open a simple color selection option or the more advanced color selection option, that is started when you click in the colored box in the dialog.

19.3.6 The Basic Shape Items

To add a basic shape (ellipse, rectangle, triangle), click the  Add basic shape icon or the  Add Arrow icon, place the element holding down the left mouse. Customize the appearance in the *Item Properties* tab.

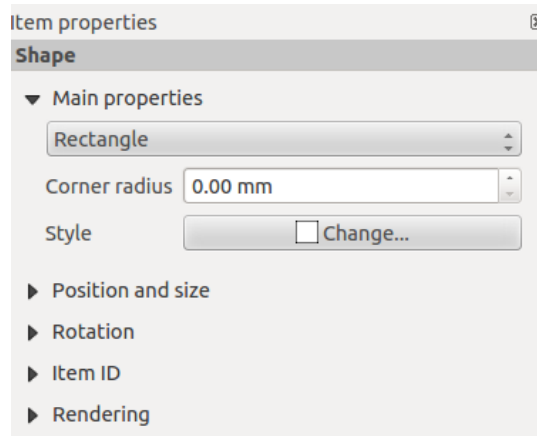
When you also hold down the `Shift` key while placing the basic shape you can create a perfect square, circle or triangle.

The *Shape* item properties tab allows you to select if you want to draw an ellipse, rectangle or triangle inside the given frame.

You can set the style of the shape using the advanced symbol style dialog with which you can define its outline and fill color, fill pattern, use markers etcetera.


For the rectangle shape, you can set the value of the corner radius to round of the corners.

Catatan: Unlike other items, you can not style the frame or the background color of the frame.



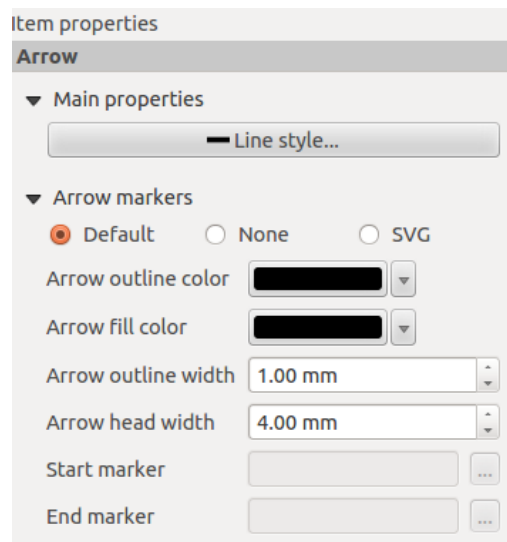
Gambar 19.24: Shape Item properties Tab 


19.3.7 The Arrow item

To add an arrow, click the  *Add Arrow* icon, place the element holding down the left mouse button and drag a line to draw the arrow on the Print Composer canvas and position and customize the appearance in the scale bar *Item Properties* tab.

When you also hold down the *Shift* key while placing the arrow, it is placed in an angle of exactly 45°.

The arrow item can be used to add a line or a simple arrow that can be used, for example, to show the relation between other print composer items. To create a north arrow, the image item should be considered first. QGIS has a set of North arrows in SVG format. Furthermore you can connect an image item with a map so it can rotate automatically with the map (see [the_image_item](#)).



Gambar 19.25: Arrow Item properties Tab 

Item Properties

The *Arrow* item properties tab allows you to configure an arrow item.

The [**Line style ...**] button can be used to set the line style using the line style symbol editor.

In *Arrows markers* you can select one of three radio buttons.

- *Default* : To draw a regular arrow, gives you options to style the arrow head
- *None* : To draw a line without arrow head
- *SVG Marker* : To draw a line with an *SVG Start marker* and/or *End marker*

For *Default* Arrow marker you can use following options to style the arrow head.


- *Arrow outline color* : Set the outline color of the arrow head
- *Arrow fill color* : Set the fill color of the arrow head
- *Arrow outline width* : Set the outline width of the arrow head
- *Arrow head width*: Set the size of the arrow head

For *SVG Marker* you can use following options.

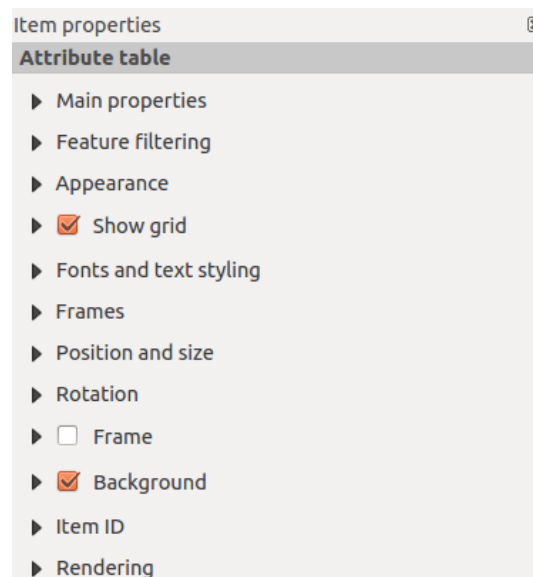
- *Start marker* : Choose an SVG image to draw at the beginning of the line
- *End marker* : Choose an SVG image to draw at the end of the line
- *Arrow head width*: Sets the size of Start and/or headmarker


SVG images are automatically rotated with the line. The color of the SVG image can not be changed.

19.3.8 The Attribute Table item

It is possible to add parts of a vector attribute table to the Print Composer canvas: Click the  Add attribute table icon, place the element with the left mouse button on the Print Composer canvas, and position and customize the appearance in the *Item Properties* tab.

The *Item properties* of an attribute table item tab provides the following functionalities (see [figure_composer_table_1](#)):

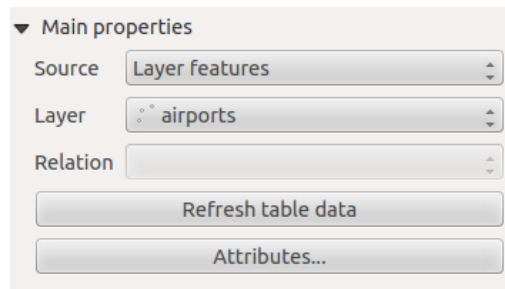



Gambar 19.26: Attribute table Item properties Tab 

Main properties

The *Main properties* dialogs of the attribute table *Item Properties* tab provide the following functionalities (see [figure_composer_table_2](#)):

- For *Source* you can normally select only 'Layer features'.



Gambar 19.27: Attribute table Main properties Dialog 

- With *Layer* you can choose from the vector layers loaded in the project.
- The button [**Refresh table data**] can be used to refresh the table when the actual contents of the table has changed.
- The button [**Attributes...**] starts the *Select attributes* menu, see [figure_composer_table_3](#), that can be used to change the visible contents of the table. After making changes use the [**OK**] button to apply changes to the table.

In the *Columns* section you can:

- Remove an attribute, just select an attribute row by clicking anywhere in a row and press the minus button to remove the selected attribute.
- Add a new attribute use the plus button. At the end a new empty row appears and you can select empty cell of the column *Attribute*. You can select a field attribute from the list or you can select to build a new attribute using a regular expression.
- Use the up and down arrows to change the order of the attributes in the table.
- Select a cel in the Headings column to change the Heading, just type a new name.
- Select a cel in the Alignment column and you can choose between Left, Center or Right alignment.
- Select a cel in the Width column and you can change it from Automatic to a width in mm, just type a number. When you want to change it back to Automatic, use the cross.
- The [**Reset**] button can allways be used to restore it to the original attribute settings.

In the *Sorting* section you can:

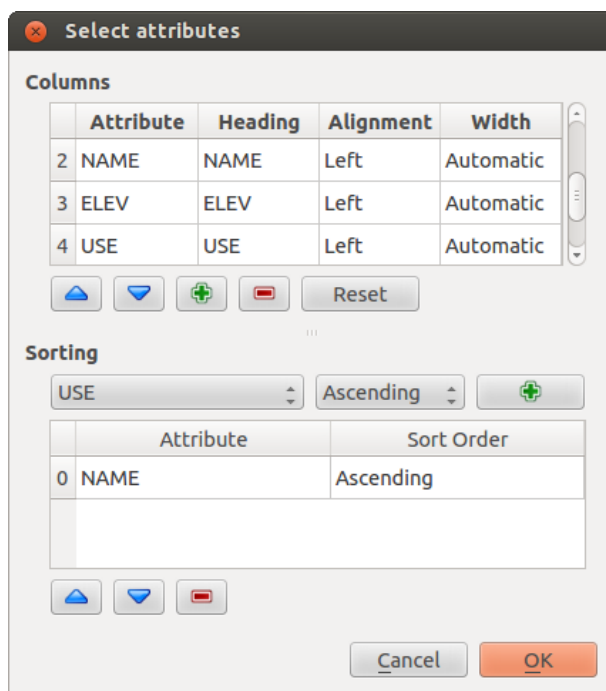
- Add an attribute to sort the table with. Select an attribute and set the sorting order to ‘Ascending’ or ‘Descending’ and press the plus button. A new line is added to the sort order list.
- select a row in the list and use the up and down button to change the sort priority on attribute level.
- use the minus button to remove an attribute from the sort order list.

Feature filtering

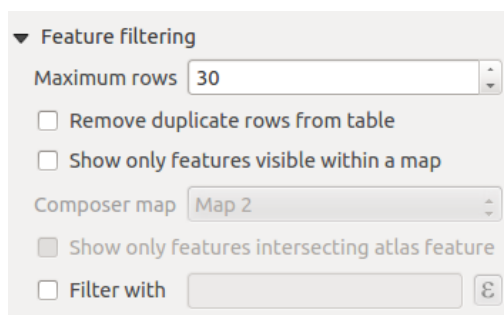
The *Feature filtering* dialogs of the attribute table *Item Properties* tab provide the following functionalities (see [figure_composer_table_4](#)):


You can:

- Define the *Maximum rows* to be displayed.
- Activate *Remove duplicate rows from table* to show unique records only.
- Activate *Show only visible features within a map* and select the corresponding *Composer map* to display the attributes of features only visible on selected map.



Gambar 19.28: Attribute table Select attributes Dialog 



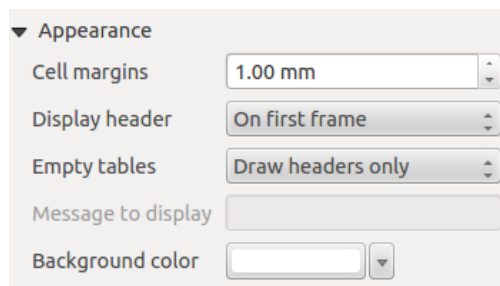
Gambar 19.29: Attribute table Feature filtering Dialog 


- 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 shown on the map of that particular page of the atlas.
- Activate *Filter with* and provide a filter by typing in the input line or insert a regular expressing use the given expression button. A few examples of filtering statements you can use when you have loaded the airports layer from the Sample dataset:
 - ELEV > 500
 - NAME = ' ANIAK'
 - NAME NOT LIKE 'AN%
 - regexp_match(attribute(\$currentfeature, 'USE') , '[i]')

The last regular expression will include only the arpoirts that have a letter 'i' in the attribute field 'USE'.

Appearance

The *Appearance* dialogs of the attribute table *Item Properties* tab provide the following functionalities (see figure_composer_table_5):

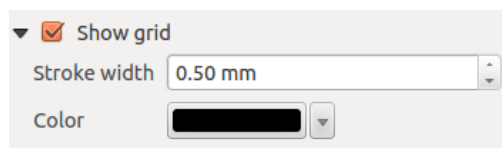



Gambar 19.30: Attribute table appearance Dialog 

- With *Cell margins* you can define the margin around text in each cell of the table.
- With *Display header* you can select from a list one of 'On first frame', 'On all frames' default option, or 'No header'.
- The option *Empty table* controls what will be displayed when the result selection is empty.
 - **Draw headers only**, will only draw the header except if you have choosen 'No header' for *Display header*.
 - **Hide entire table**, will only draw the background of the table. You can activate *Don't draw background if frame is empty* in *Frames* to completely hide the table.
 - **Draw empty cells**, will fill the attribute table with empty cells, this option can also be used to provide additional empty cells when you have a result to show!
 - **Show set message**, will draw the header and adds a cell spanning all columns and display a message like 'No result' that can be provided in the option *Message to display*
- The option *Message to display* is only activated when you have selected **Show set message** for *Empty table*. The message provided will be shown in the table in the first row, when the result is an empty table.
- With *Background color* you can set the background color of the table.

Show grid

The *Show grid* dialog of the attribute table *Item Properties* tab provide the following functionalities (see figure_composer_table_6):

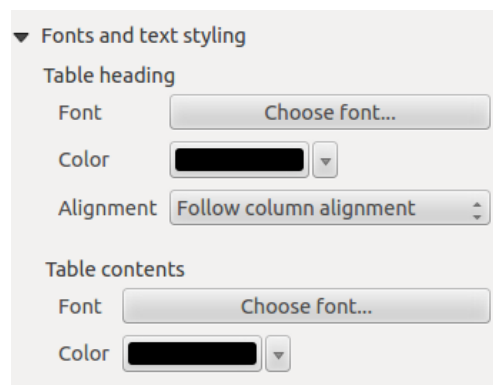


Gambar 19.31: Attribute table Show grid Dialog 

- Activate *Show grid* when you want to display the grid, the outlines of the table cells.
- With *Stroke width* you can set the thickness of the lines used in the grid.
- The *Color* of the grid can be set using the color selection dialog.

Fonts and text styling

The *Fonts and text styling* dialog of the attribute table *Item Properties* tab provide the following functionalities (see [figure_composer_table_7](#)):

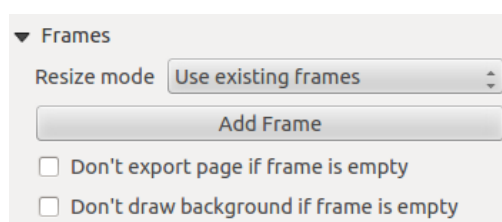



Gambar 19.32: Attribute table Fonts and text styling Dialog 

- You can define *Font* and *Color* for *Table heading* and *Table contents*.
- For *Table heading* you can additionally set the *Alignment* and choose from *Follow column alignment*, *Left*, *Center* or *Right*. The column alignment is set using the *Select Attributes* dialog (see [Figure_composer_table_3](#)).

Frames

The *Frames* dialog of the attribute table *Item Properties* tab provide the following functionalities (see [figure_composer_table_8](#)):




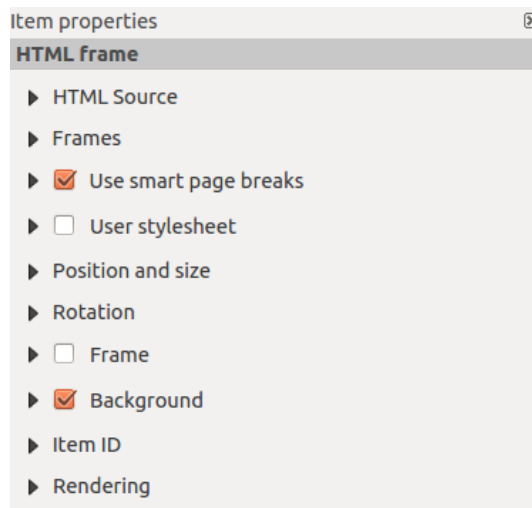
Gambar 19.33: Attribute table Frames Dialog 


- With *Resize mode* you can select how to render the attribute table contents:
 - *Use existing frames* displays the result in the first frame and added frames only.
 - *Extent 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 composer items, maps, scalebars, legends etc. will not be visible in the result.
- Activate *Don't draw background if frame is empty* prevents the background to be drawn when the table frame has no contents.

19.3.9 The HTML frame item

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!

Click the  **Add HTML frame** icon, place the element by dragging a rectangle holding down the left mouse button on the Print Composer canvas and position and customize the appearance in the *Item Properties* tab (see [figure_composer_html_1](#)).

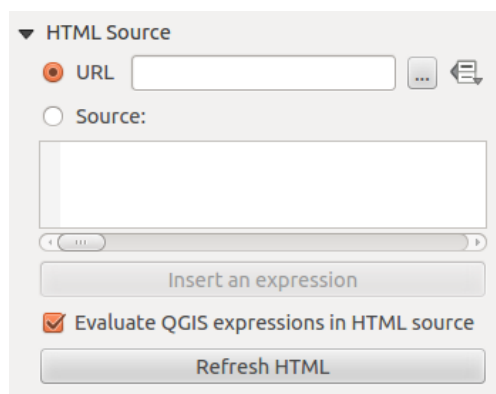


Gambar 19.34: HTML frame, the item properties Tab 


HTML Source

As an HTML source, you can either set a URL and activate the URL radiobutton or enter the HTML source directly in the textbox provided and activate the Source radiobutton.

The *HTML Source* dialog of the HTML frame *Item Properties* tab provides the following functionalities (see [figure_composer_html_2](#)):

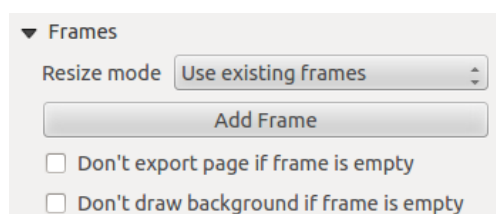


Gambar 19.35: HTML frame, the HTML Source properties 🐧

- 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 an URL from the contents of an attribute field of a table or using a regular expression.
- In *Source* you can enter text in the textbox with some HTML tags or provide a full HTML page.
- 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.
- Activate *Evaluate QGIS expressions in HTML code* to see the result of the expression you have included, otherwise you will see the expression instead.
- Use the **[Refresh HTML]** button to refresh the HTML frame(s) to see the result of changes.

Frames

The *Frames* dialog of the HTML frame *Item Properties* tab provides the following functionalities (see [figure_composer_html_3](#)):



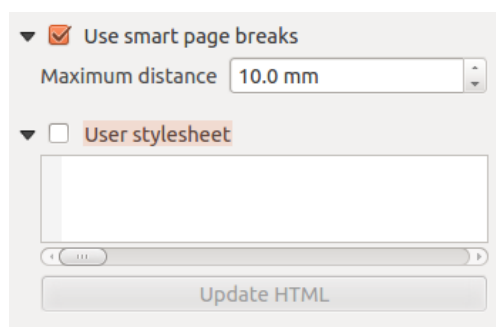
Gambar 19.36: HTML frame, the Frames properties 🐧


- With *Resize mode* you can select how to render the HTML contents:
 - *Use existing frames* displays the result in the first frame and added frames only.
 - *Extent 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 that will 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 map layout from being exported when the frame has no HTML contents. This means all other composer items, maps, scalebars, legends etc. will not be visible in the result.
- Activate *Don't draw background if frame is empty* prevents the HTML frame being drawn if the frame is empty.

Use smart page breaks and User style sheet

The *Use smart page breaks* dialog and *Use style sheet* dialog of the HTML frame *Item Properties* tab provides the following functionalities (see [figure_composer_html_4](#)):



Gambar 19.37: HTML frame, Use smart page breaks and User stylesheet properties 


- Activate *Use smart page breaks* to prevent the html frame contents from breaking mid-way a line of text so it continues nice and smooth in the next frame.
- Set the *Maximum distance* allowed when calculating where to place page breaks in the html. This distance is the maximum amount of empty space allowed at the bottom of a frame after calculating the optimum break location. Setting a larger value will result in better choice of page break location, but more wasted space at the bottom of frames. This is only used when *Use smart page breaks* is activated.
- Activate *User stylesheet* to apply HTML styles that often is provided in cascading style sheets. An example of style code is provide below to set the color of `<h1>` header tag to green and set the font and fontsize 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 stylesheet settings.

19.4 Manage items


19.4.1 Size and position

Each item inside the Composer can be moved/resized to create a perfect layout. For both operations the first step is to activate the  *Select/Move item* tool and to click on the item; you can then move it using the mouse while holding the left button. If you need to constrain the movements to the horizontal or the vertical axis, just hold the *Shift*

while moving the mouse. If you need a better precision, you can move a selected item using the `Arrow` keys on the keyboard; if the movement is too slow, you can speed up it by holding `Shift`.

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.

The correct position for an item can be obtained using snapping to grid or smart guides. Guides are set by clicking and dragging in the rulers. Guide are moved by clicking in the ruler, level with the guide and dragging to a new place. To delete a guide move it off the canvas. If you need to disable the snap on the fly just hold `Ctrl` while moving the mouse.

You can choose multiple items with the  `Select/Move item` button. Just hold the `Shift` button and click on all the items you need. You can then resize/move this group just like a single item.


Once you have found the correct position for an item, you can lock it by using the items on the toolbar or ticking the box next to the item in the *Items* tab. Locked items are **not** selectable on the canvas.


Locked items can be unlocked by selecting the item in the *Items* tab and unchecking the tickbox or you can use the icons on the toolbar.

To unselect an item, just click on it holding the `Shift` button.

Inside the *Edit* menu, you can find actions to select all the items, to clear all selections or to invert the current selection.

19.4.2 Alignment

Raising or lowering functionalities for elements are inside the  `Raise selected items` pull-down menu. Choose an element on the Print Composer canvas and select the matching functionality to raise or lower the selected element compared to the other elements (see [table_composer_1](#)). This order is shown in the *Items* tab. You can also raise or lower objects in the *Items* tab by clicking and dragging an object's label in this list.

There are several alignment functionalities available within the  `Align selected items` pull-down menu (see [table_composer_1](#)). To use an alignment functionality, you first select some elements and then click on the matching alignment icon. All selected elements will then be aligned within to their common bounding box. When moving items on the Composer canvas, alignment helper lines appear when borders, centers or corners are aligned.



19.4.3 Copy/Cut and Paste items

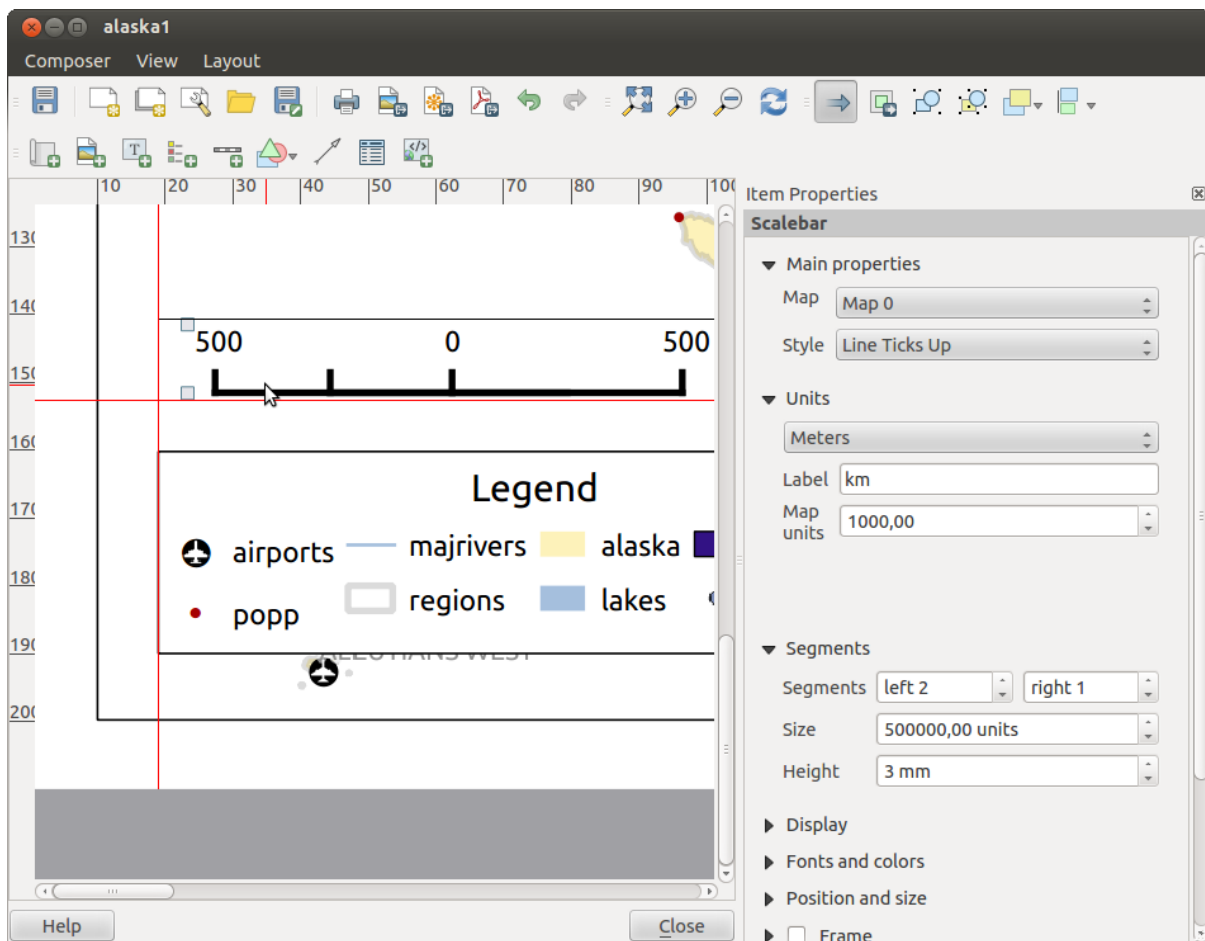
The print composer includes actions to use the common Copy/Cut/Paste functionality for the items in the layout. As usual first you need to select the items using one of the options seen above; at this point the actions can be found in the *Edit* menu. When using the Paste action, the elements will be pasted according to the current mouse position.

Catatan: HTML items can not be copied in this way. As a workaround, use the **[Add Frame]** button in the *Item Properties* tab.

19.5 Revert and Restore tools

During the layout process, it is possible to revert and restore changes. This can be done with the revert and restore tools:

-  Revert last changes
-  Restore last changes



Gambar 19.38: Alignment helper lines in the Print Composer 🐧

This can also be done by mouse click within the *Command history* tab (see [figure_composer_29](#)).

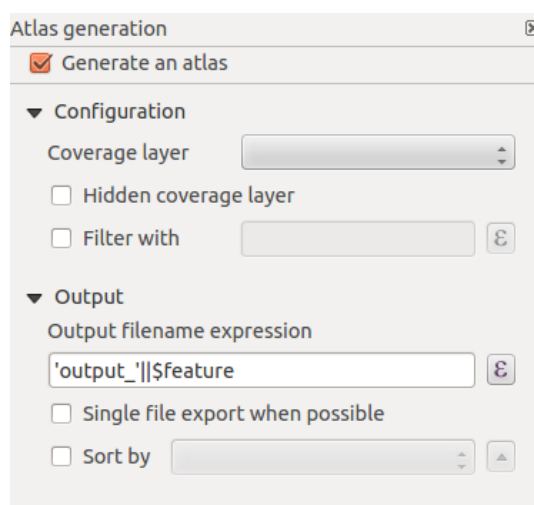


Gambar 19.39: Command history in the Print Composer 


19.6 Atlas generation



The Print Composer includes generation functions that allow you to create map books in an automated way. The concept is to use a coverage layer, which contains geometries and fields. For each geometry in the coverage layer, a new output will be generated where the content of some canvas maps will be moved to highlight the current geometry. Fields associated with this geometry can be used within text labels.


Every page will be generated with each feature. To enable the generation of an atlas and access generation parameters, refer to the *Atlas generation* tab. This tab contains the following widgets (see [Figure_composer_atlas](#)):




Gambar 19.40: Atlas generation tab 

- *Generate an atlas*, which enables or disables the atlas generation.
- A *Coverage layer*  combo box that allows you to choose the (vector) layer containing the geometries on which to iterate over.
- An optional *Hidden coverage layer* that, if checked, will hide the coverage layer (but not the other ones) during the generation.
- 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 selected. The button on the right allows you to display the expression builder.
- An *Output filename expression* textbox that is used to generate a filename for each geometry if needed. It is based on expressions. This field 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 optional  *Sort by* that, if checked, allows you to sort features of the coverage layer. The associated combo box allows you to choose which column will be used as the sorting key. Sort order (either ascending or descending) is set by a two-state button that displays an up or a down arrow.

You can use multiple map items with the atlas generation; each map will be rendered according to the coverage features. To enable atlas generation for a specific map item, you need to check  *Controlled by Atlas* under the item properties of the map item. Once checked, you can set:

- An input box *Margin around feature* that allows you to select the amount of space added around each geometry within the allocated map. Its value is meaningful only when using the auto-scaling mode.
- A  *Fixed scale* that allows you to toggle between auto-scale and fixed-scale mode. In fixed-scale mode, the map will only be translated for each geometry to be centered. In auto-scale mode, the map's extents are computed in such a way that each geometry will appear in its entirety.

19.6.1 Labels


In order to adapt labels to the feature the atlas plugin iterates over, you can include expressions. For example, for a city layer with fields CITY_NAME and ZIPCODE, you could insert this:

The area of [% upper(CITY_NAME) || ', ' || ZIPCODE || ' is ' format_number(\$area/1000000,2) %] km2

The information [% upper(CITY_NAME) || ', ' || ZIPCODE || ' is ' format_number(\$area/1000000,2) %] is an expression used inside the label. That would result in the generated atlas as:


The area of PARIS,75001 is 1.94 km2

19.6.2 Data Defined Override Buttons


There are several places where you can use a  Data Defined Override button to override the selected setting. These options are particularly usefull with Atlas Generation.

For the following examples the *Regions* layer of the QGIS sample dataset is used and selected for Atlas Generation. We also assume the paper format *A4 (210X297)* is selected in the *Composite* tab for field *Presets*.


With a *Data Defined Override* button you can dynamically set the paper orientation. When the height (north-south) of the extents of a region is greater than it's width (east-west), you rather want to use *portrait* instead of *landscape* orientation to optimize the use of paper.

In the *Composition* you can set the field *Orientation* and select *Landscape* or *Portrait*. We want to set the orientation dynamically using an expression depending on the region geometry. press the  button of field *Orientation*, select *Edit ...* so the *Expression string builder* dialog opens. Give following expression:


CASE WHEN bounds_width(\$atlasgeometry) > bounds_height(\$atlasgeometry) THEN 'Landscape' ELSE 'Portrait'

Now the paper orients itself automatically for each Region you need to reposition the location of the composer item as well. For the map item you can use the  button of field *Width* to set it dynamically using following expression:

(CASE WHEN bounds_width(\$atlasgeometry) > bounds_height(\$atlasgeometry) THEN 297 ELSE 210 END) - 10

Use the  button of field *Height* to provide following expression:

(CASE WHEN bounds_width(\$atlasgeometry) > bounds_height(\$atlasgeometry) THEN 210 ELSE 297 END) - 10

When you want to give a title above map in the center of the page, insert a label item above the map. First use the item properties of the label item to set the horizontal alignment to  *Center*. Next activate from *Reference point* the upper middle checkbox. You can provide following expression for field *X* :

```
(CASE WHEN bounds_width($atlasgeometry) > bounds_height($atlasgeometry) THEN 297 ELSE 210 END) /
```

For all other composer items you can set the position in a similar way so they are correctly positioned when page is automatically rotated in portrait or landscape.


Information provided is derived from the excellent blog (in english and portugese) on the Data Defined Override options [Multiple_format_map_series_using_QGIS_2.6](#) .

This is just one example of how you can use Data Defined Overrides.

19.6.3 Preview

Once the atlas settings have been configured and map items selected, you can create a preview of all the pages by clicking on *Atlas* → *Preview Atlas* and using the arrows, in the same menu, to navigate through all the features.





19.6.4 Generation


The atlas generation can be done in different ways. For example, with *Atlas* → *Print Atlas*, you can directly print it. You can also create a PDF using *Atlas* → *Export Atlas as PDF*: The user will be asked for a directory for saving all the generated PDF files (except if the  *Single file export when possible* has been selected). If you need to print just a page of the atlas, simply start the preview function, select the page you need and click on *Composer* → *Print* (or create a PDF).

19.7 Creating Output

[Figure_composer_output](#) shows the Print Composer with an example print layout, including each type of map item described in the sections above.

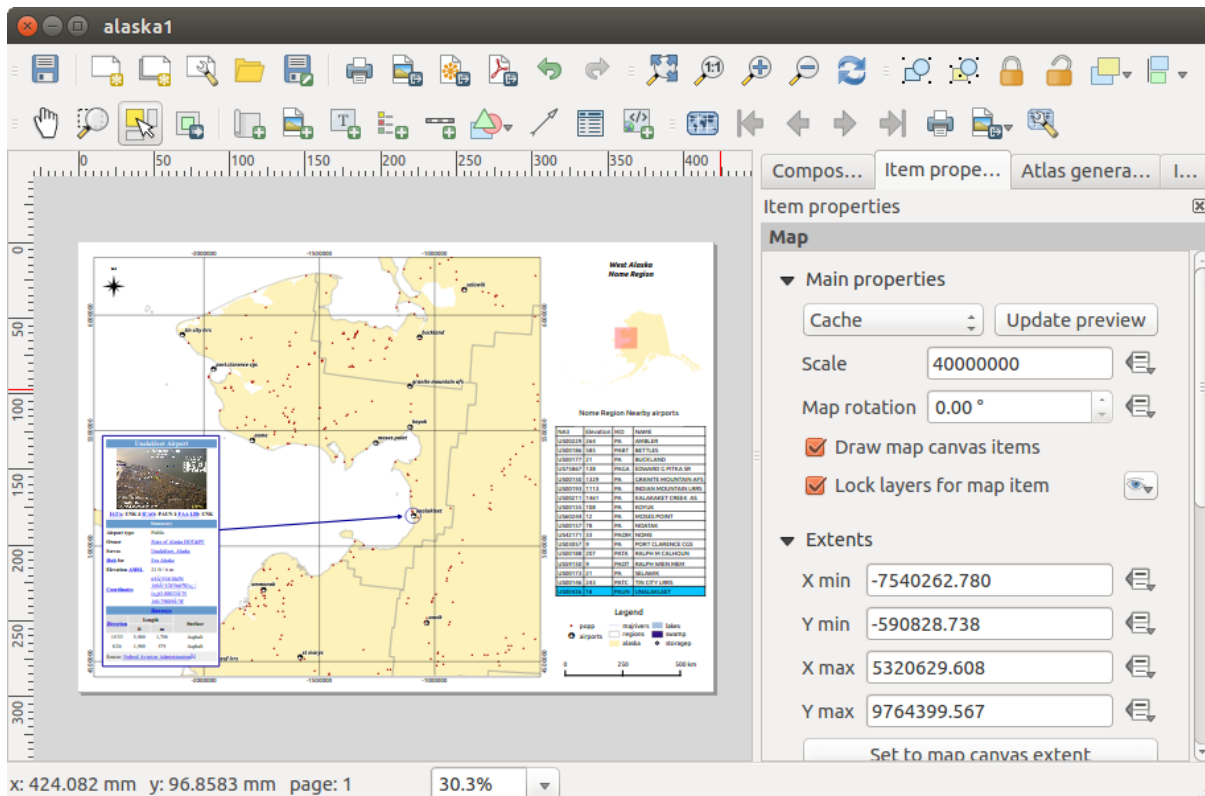
The Print Composer allows you to create several output formats, and it is possible to define the resolution (print quality) and paper size:

- The  **Print** icon allows you to print the layout to a connected printer or a PostScript file, depending on installed printer drivers.
- The  **Export as image** icon exports the Composer canvas in several image formats, such as PNG, BPM, TIF, JPG,...
-  **Export as PDF** saves the defined Print Composer canvas directly as a PDF.
- The  **Export as SVG** icon saves the Print Composer canvas as an SVG (Scalable Vector Graphic).

If you need to export your layout as a **georeferenced image** (i.e., to load back inside QGIS), you need to enable this feature under the Composition tab. Check  *World file on* and choose the map item to use. With this option, the 'Export as image' action will also create a world file.

Catatan:

- Currently, the SVG output is very basic. This is not a QGIS problem, but a problem with the underlying Qt library. This will hopefully be sorted out in future versions.
 - Exporting big rasters can sometimes fail, even if there seems to be enough memory. This is also a problem with the underlying Qt management of rasters.
-



Gambar 19.41: Print Composer with map view, legend, image, scale bar, coordinates, text and HTML frame added

19.8 Manage the Composer

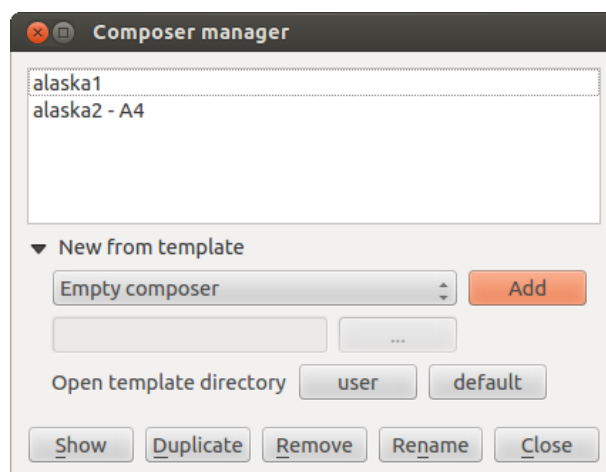
With the Save as template and Add items from template icons, you can save the current state of a Print Composer session as a .qpt template and load the template again in another session.

The Composer Manager button in the QGIS toolbar and in *Composer* → *Composer Manager* allows you to add a new Composer template, create a new composition based on a previously saved template or to manage already existing templates.

By default, the Composer manager searches for user templates in `~/qgis2/composer_template`.

The New Composer and Duplicate Composer buttons in the QGIS toolbar and in *Composer* → *New Composer* and *Composer* → *Duplicate Composer* allow you to open a new Composer dialog, or to duplicate an existing composition from a previously created one.

Finally, you can save your print composition with the Save Project button. This is the same feature as in the QGIS main window. All changes will be saved in a QGIS project file.



Gambar 19.42: The Print Composer Manager 

20.1 Plugin-plugin QGIS

QGIS telah dirancang dengan arsitektur plugin. Hal ini memungkinkan banyak fitur baru / fungsi dapat dengan mudah ditambahkan ke aplikasi. Banyak fitur di QGIS benar-benar diterapkan sebagai plugin.

You can manage your plugins in the plugin dialog which can be opened with *Plugins > Manage and install plugins*

When a plugin needs to be updated, and if plugins settings have been set up accordingly, QGIS main interface could display a blue link in the status bar to tell you that there are some plugins updating waiting to be applied.

20.1.1 The Plugins Dialog

The menus in the Plugins dialog allow the user to install, uninstall and upgrade plugins in different ways. Each plugin have some metadatas displayed in the right panel:

- information if the plugin is experimental
- description
- rating vote(s) (you can vote for your preferred plugin!)
- tags
- some useful links as the home page, tracker and code repository
- author(s)
- version available

You can use the filter to find a specific plugin.



Semua

Di sini, semua plugin yang tersedia terdaftar, termasuk plugin inti dan eksternal. Gunakan [**Upgrade semua**] untuk mencari versi baru dari plugin. Selanjutnya, Anda dapat menggunakan [**pasang plugin**], jika plugin tercantum dalam daftar namun tidak dipasang, dan [**Uninstall plugin**] maupun [**Reinstall plugin**], jika plugin telah terpasang. Jika sebuah plugin terpasang, dapat diaktifkan/dinonaktifkan menggunakan kotak centang.

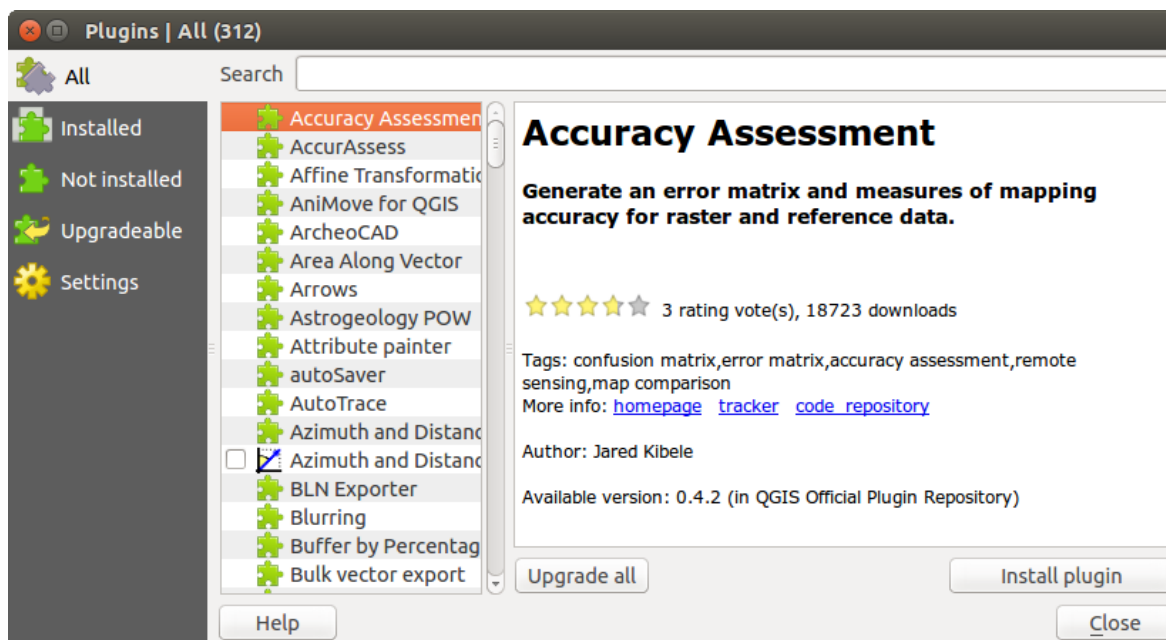


Terpasang

Dalam menu ini, Anda hanya dapat menemukan plugin terpasang. Plugin eksternal dapat dihapus dan diinstal ulang dengan menggunakan tombol [**Uninstall plugin**] and [**Reinstall plugin**]. Anda bisa [**Upgrade semua**].



Tidak terpasang



Gambar 20.1: Menu  Semua 




Gambar 20.2: Menu  Terpasang 

Menu ini berisi semua plugin yang tidak dipasang. Anda dapat menggunakan tombol **[Pasang plugin]** untuk mengimplementasikan plugin dalam QGIS.



Gambar 20.3: Menu  Tidak terpasang 

Upgradeable

Jika Anda mengaktifkan *Tampilkan juga plugin eksperimental* dalam menu  *Pengaturan*, Anda bisa menggunakan menu ini untuk mencari versi Plugin yang lebih baru. Bisa dilakukan dengan tombol **[Upgrade plugin]** atau **[Upgrade semua]**.

Pengaturan

Dalam menu ini, Anda dapat menggunakan pilihan berikut:

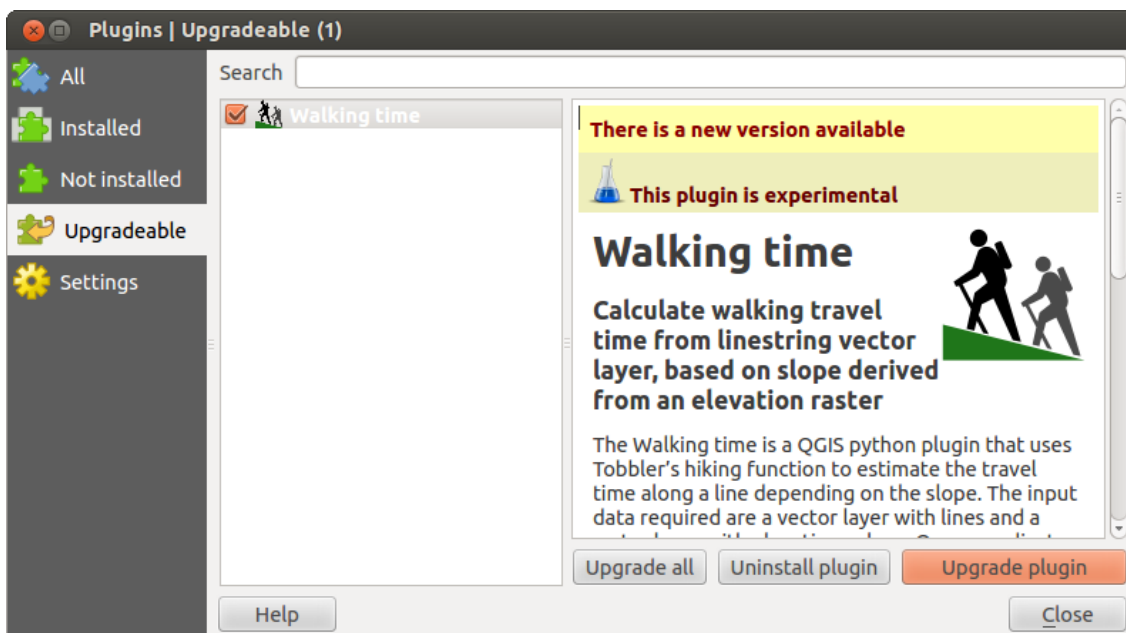
- *Periksa pembaruan saat startup*. Setiap kali sebuah plugin baru atau update plugin tersedia, QGIS akan memberitahu Anda ‘every time QGIS starts’, ‘once a day’, ‘every 3 days’, ‘every week’, ‘every 2 weeks’ or ‘every month’.
- *Tampilkan juga plugin eksperimental*. QGIS akan menunjukkan plugin di tahap awal pengembangan, yang umumnya tidak cocok untuk penggunaan produksi.
- *Tampilkan juga plugin usang*. Plugin ini usang dan umumnya tidak cocok untuk penggunaan produksi.

Untuk menambah repositori penulis eksternal, klik **[Tambah...]** dalam bagian *Plugin repository*. Jika Anda tidak ingin salah satu atau lebih dari repositori ditambahkan, mereka dapat dinonaktifkan melalui tombol **[Edit...]**, atau benar-benar dihapus dengan tombol **[Hapus]**.

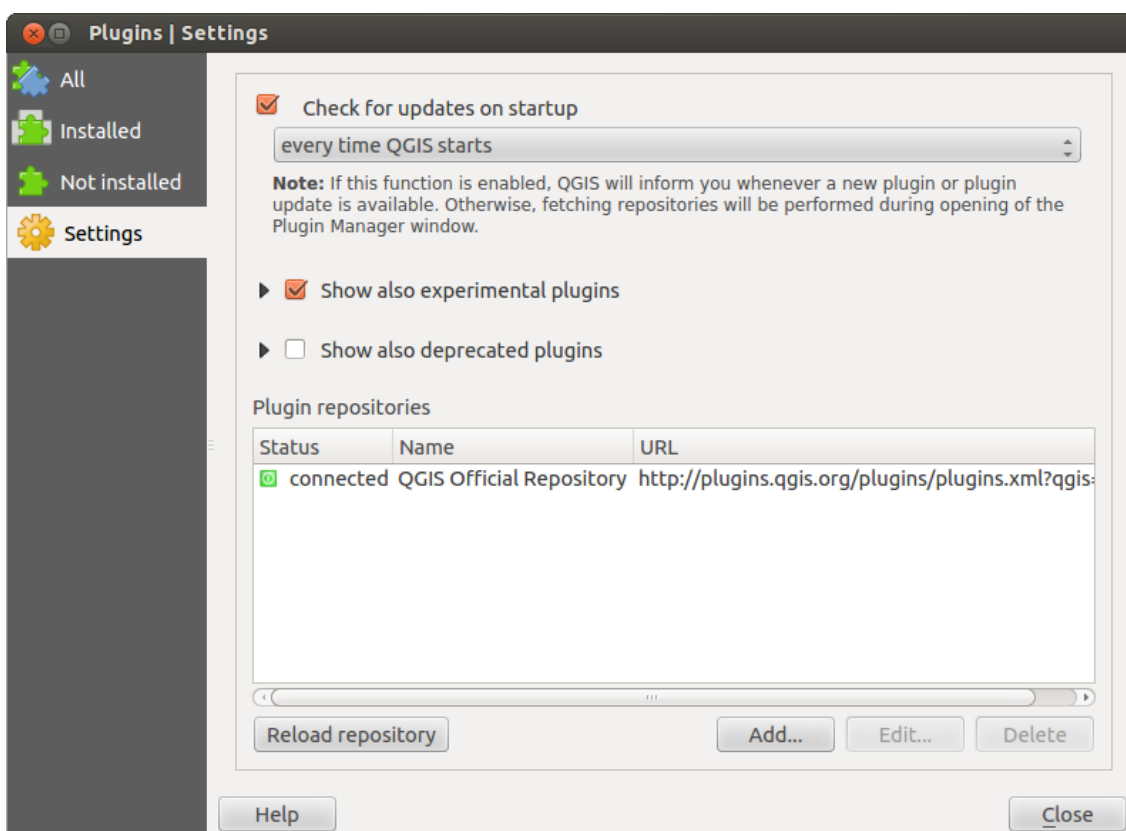
Fungsi *Cari* tersedia di hampir setiap menu (kecuali  *Pengaturan*). Di sini, Anda dapat mencari plugin tertentu.

Tip: Plugin inti dan eksternal

Plugin QGIS diimplementasikan dengan baik sebagai **Plugin Inti** atau **Plugin Eksternal**. **Plugin Inti** dipelihara oleh Tim Pengembangan QGIS dan secara otomatis bagian dari setiap distribusi QGIS. Mereka ditulis dalam salah satu dari dua bahasa: C++ atau Python. **Plugin Eksternal** saat ini ditulis menggunakan Python. Mereka disimpan dalam repositori eksternal dan dipelihara oleh penulis individu.



Gambar 20.4: Menu  Upgradeable 



Gambar 20.5: Menu  Settings 

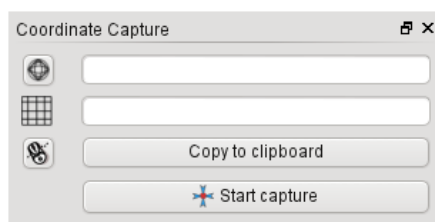
Dokumentasi rinci tentang penggunaan, minimum versi QGIS, homepage, penulis, dan informasi penting lainnya yang disediakan untuk Repositori ‘Resmi’ QGIS di <http://plugins.qgis.org/plugins/>. Untuk repositori eksternal lainnya, mungkin tersedia dengan plugin eksternal sendiri. Secara umum tidak termasuk dalam panduan ini.

20.2 Menggunakan QGIS Plugin Inti

Ikon	Plugin	Deskripsi	Referensi Panduan
	Tangkapan Koordinat	Tetikus menangkap koordinat di CRS yang berbeda	:ref:coordcapt
	Pengelola DB	Mengelola basis data Anda dengan QGIS	<i>Plugin Pengelola DB</i>
	Pengubah DXF2Shape	Mengubah berkas format dari DXF ke SHP	<i>Plugin Pengonversi Dxf2Shp</i>
	eVis	Alat Visualisasi	<i>Plugin eVis</i>
	fTools	Perangkat alat vektor	<i>Plugin fTools</i>
	Perangkat GPS	Alat untuk bongkar impor data GPS	<i>GPS Plugin</i>
	GRASS	Fungsionalitas GRASS	<i>GRASS GIS Integration</i>
	Alat GDAL	Fungsionalitas raster GDAL	<i>Plugin Peralatan GDAL</i>
	Pemberi Georeferensi GDAL	Georeferensi Raster dengan GDAL	<i>Plugin Georeferencer</i>
	Heatmap	Membuat heatmap raster dari memasukkan titik vektor	<i>Plugin Heatmap</i>
	Plugin Interpolasi	Interpolasi pada dasar simpul dari lapisan vektor	<i>Plugin Interpolasi</i>
	Menyunting di luar jaringan	Penyuntingan di luar jaringan dan sinkronisasi dengan basis data	<i>Plugin Mengedit Diluar Jaringan (Offline)</i>
	Oracle Spasial Georaster	Akses Oracle Spasial GeoRaster	<i>Plugin Spasial Oracle GeoRaster</i>
	Pengelola Plugin	Mengelola inti dan plugin eksternal	<i>The Plugins Dialog</i>
	Analisis Raster Terrain	Fitur menghitung geomorfologi dari DEM	<i>Plugin Raster Analisis Terrain</i>
	Plugin Grafik Jalan	Analisis jalur terpendek	<i>Plugin Grafik Jalan</i>
	Plugin SQL Anywhere	Akses SQL dimanapun DB	<i>Plugin SQL Anywhere</i>
	Query Spasial	Query spasial pada vektor	<i>Plugin Spasial Query</i>
	SPIT	Alat Impor Shapefile ke PostgreSQL/PostGIS	<i>Plugin SPIT</i>
	Statistik Zonal	Hitung statistik raster untuk poligon vektor	<i>Plugin Statistik Zonal</i>
	MetaSearch	Interact with metadata catalogue services (CSW)	<i>MetaSearch Catalogue Client</i>

20.3 Plugin Mengambil Koordinat


Plugin Mengambil Koordinat mudah digunakan dan memberikan kemampuan menampilkan koordinat di kanvas peta untuk dua Sistem Koordinat Referensi (CRS) dipilih.



Gambar 20.6: Plugin Mengambil Koordinat 🐧

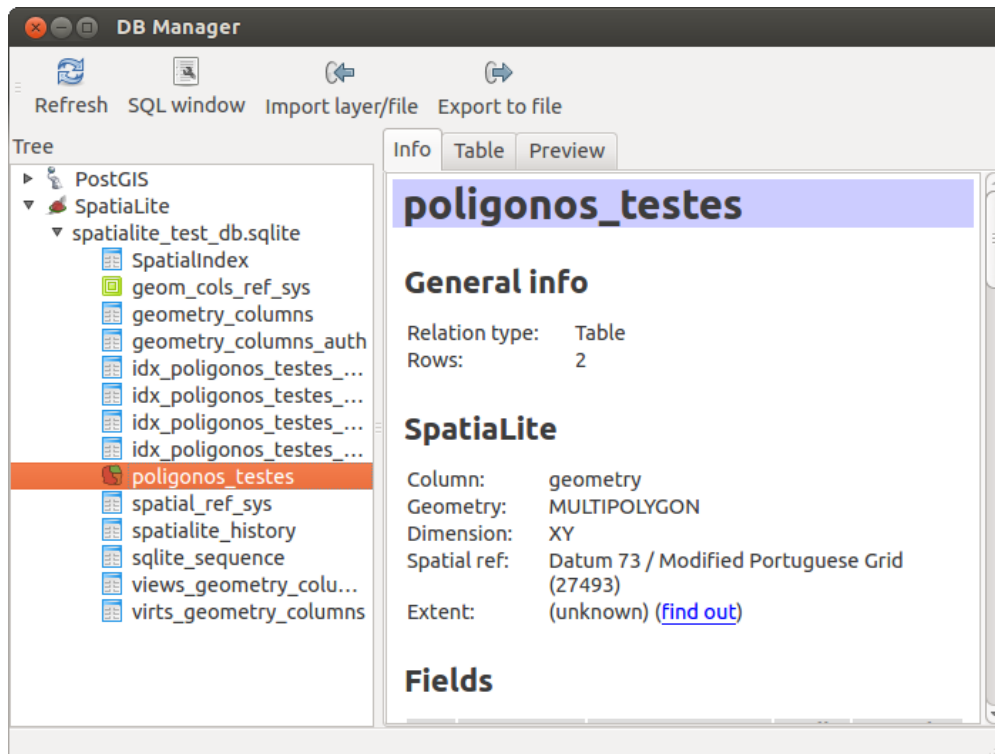
1. Mulai QGIS, pilih 🗝️ *Proyek Properti* dari menu *Pengaturan* (KDE, Windows) atau *Berkas* (Gnome, OSX) dan klik tab *Proyeksi*. Sebagai alternatif, Anda juga bisa klik ikon 🌐 CRS status di sudut kanan bawah status bar.
2. Klik pada kotak centang *Aktifkan proyeksi on the fly* dan pilih proyeksi sistem koordinat dari yang Anda pilih (lihat juga *Working with Projections*).
3. Activate the coordinate capture plugin in the Plugin Manager (see *The Plugins 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_1](#). Alternatively, you can also go to *Vector* → *Coordinate Capture* and see if *Coordinate Capture* is enabled.
4. Klik pada ikon 🌐 Klik pilih CRS untuk menampilkan koordinat dan memilih CRS berbeda dari yang Anda pilih di atas.
5. Untuk mulai mengambil koordinat, klik pada [**Mulai ambil**]. Sekarang Anda dapat mengklik di mana saja di kanvas peta dan plugin akan menampilkan koordinat untuk kedua CRS yang Anda pilih.
6. Untuk mengaktifkan tetikus pelacak koordinat, klik ikon 🖱️ pelacak tetikus.
7. Anda juga bisa menyalin koordinat yang dipilih ke papan klip.

20.4 Plugin Pengelola DB

Plugin Pengelola DB resmi bagian dari inti QGIS dan bermaksud untuk menggantikan Plugin SPIT dan tambahan untuk mengintegrasikan semua format basisdata lain yang didukung oleh QGIS dalam satu antarmuka pengguna. Plugin  *Manajer DB* menyediakan beberapa fitur. Anda dapat menggeser lapisan dari Penjelajah QGIS ke Pengelola DB dan akan mengimpor lapisan Anda ke dalam basisdata spasial Anda. Anda dapat menggeser dan menempatkan tabel antar basisdata spasial dan mereka akan diimpor. Anda juga dapat menggunakan Pengelola DB untuk mengeksekusi query SQL terhadap basisdata spasial Anda dan kemudian melihat keluaran spasial untuk queries dengan menambahkan hasilnya ke QGIS sebagai lapisan query.

Menu *Basisdata* memungkinkan untuk terhubung ke basisdata yang sudah ada, untuk memulai SQL-Window dan untuk keluar dari Plugin Pengelola DB. Satu Anda terhubung ke basisdata yang sudah ada, menu *Skema* dan *Tabel* tambahan muncul.

Menu *Skema* termasuk alat untuk membuat dan menghapus (kosong) skema dan, jika topologi tersedia (seperti PostGIS 2) untuk memulai *TopoViewer*.



Gambar 20.7: Dialog Pengelola DB 

Menu *Tabel* memungkinkan untuk membuat dan mengedit tabel dan menghapus tabel dan tampilan. Hal ini juga memungkinkan untuk mengosongkan tabel dan untuk memindahkan tabel dari skema satu ke skema lain. Seperti fungsionalitas lebih lanjut Anda dapat melakukan VACUUM dan kemudian ANALYZE untuk setiap tabel yang dipilih. VACUUM sederhana mengambil kembali ruang dan membuatnya tersedia untuk digunakan kembali dan ANALYZE update statistik untuk menentukan cara yang paling efisien untuk mengeksekusi query. Akhirnya Anda dapat mengimpor lapisan/berkas, jika mereka dimuat dalam QGIS atau ada dalam sistem berkas. Dan Anda dapat mengekspor tabel basisdata untuk Shape dengan fitur Ekspor berkas.

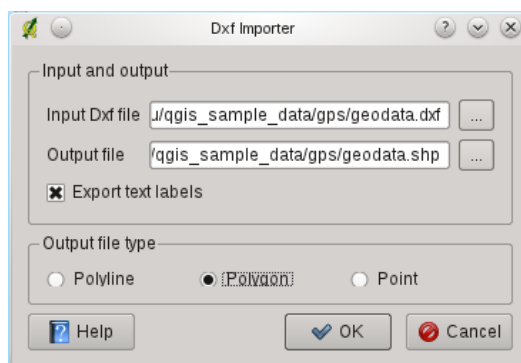
Jendela *Tree* daftar semua database yang ada didukung oleh QGIS. Dengan double-klik Anda bisa koneksi ke basisdata. Dengan tombol tetikus-kanan Anda bisa mengubah nama dan menghapus skema dan tabel yang ada. Tabel juga dapat ditambahkan ke kanvas QGIS dengan menu konteks.

Jika terkoneksi ke sebuah basisdata, jendela **utama** dari Pengelola DB menyediakan tiga tab. Tab *Info* memberikan informasi tentang tabel dan geometri serta tentang kolom yang ada, Kendala dan Indeks. Hal ini juga memungkinkan untuk menjalankan Vacuum Analisis dan untuk membuat indeks spasial pada tabel yang dipilih, jika belum melakukannya. Tab *Tabel* menampilkan semua atribut dan tab *Pratinjau* render geometri sebagai pratinjau.

20.5 Plugin Pengonversi Dxf2Shp

Plugin pengonversi dxf2shape dapat digunakan untuk mengkonversi data vektor dari DXF ke format Shapefile. Hal ini membutuhkan parameter berikut harus ditentukan sebelum menjalankan:



- **Masukan berkas DXF:** Masukkan path ke berkas DXF yang akan dikonversi.
- **Keluaran berkas SHP:** Masukkan nama yang diinginkan dari Shapefile yang akan dibuat.
- **Keluaran tipe berkas:** Tentukan jenis geometri keluaran Shapefile. Saat ini jenis yang didukung adalah polyline, polygon, dan titik.



Gambar 20.8: Plugin Pengonversi Dxf2Shp

- **Ekspor label teks:** Ketika kotak centang ini diaktifkan, lapisan titik Shapefile tambahan akan dibuat, dan tabel DBF terkait akan berisi informasi tentang kolom “TEXT” yang ditemukan dalam berkas DXF, dan teks string mereka sendiri.

20.5.1 Menggunakan Plugin

1. Mulai QGIS, muat plugin Dxf2Shape di Manajer Plugin (lihat Bagian *The Plugins Dialog*) dan klik pada ikon  Pengonversi Dxf2Shape yang muncul di menu toolbar QGIS. Dialog plugin Dxf2Shape akan muncul seperti *Figure_dxf2shape_1*.
2. Masukkan berkas DXF, nama untuk keluaran Shapefile dan jenis Shapefile.
3. Aktifkan kotak centang  *Ekspor teks label* jika Anda ingin membuat lapisan titi ekstra dengan label.
4. Klik [OK].

20.6 Plugin eVis

(Bagian ini berasal dari Horning, N., K. Koy, P. Ersts. 2009. eVis (v1.1.0) User’s Guide. American Museum of Natural History, Center for Biodiversity and Conservation. Tersedia di <http://biodiversityinformatics.amnh.org/>, dan dirilis di bawah 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.

Sekarang eVis secara otomatis terpasang dan diaktifkan pada QGIS versi baru, dan seperti semua plugin, dapat dinonaktifkan dan diaktifkan menggunakan Manajer Plugin (lihat *The Plugins Dialog*).

Plugin Evis terdiri dari tiga modul: ‘Alat sambungan basisdata’, ‘Alat ID Event’, dan ‘Navigasi Event’. Bekerja sama untuk melihat foto geocoded dan dokumen lain yang terkait dengan fitur disimpan dalam berkas vektor, database, atau spreadsheet.

20.6.1 Navigasi Event

Modul ‘Event Browser’ menyediakan fungsi tampilan foto geo kode yang memiliki tautan pada fitur vektor yang tampil dalam jendela peta lqgl. Data titik, sebagai contoh, dapat terbentuk dari berkas vektor yang dapat diinput menggunakan lqgl atau dapat juga dengan menggunakan hasil ‘query’ basis data. Fitur vektor wajib memiliki

atribut informasi yang berhubungan dengannya sehubungan dengan lokasi dan nama berkas yang berisi foto dan, opsional, arah mata angin sudut pengambilan gambar. Lapisan vektor wajib dimuat dalam lqgl sebelum menjalankan 'Event Browser'.

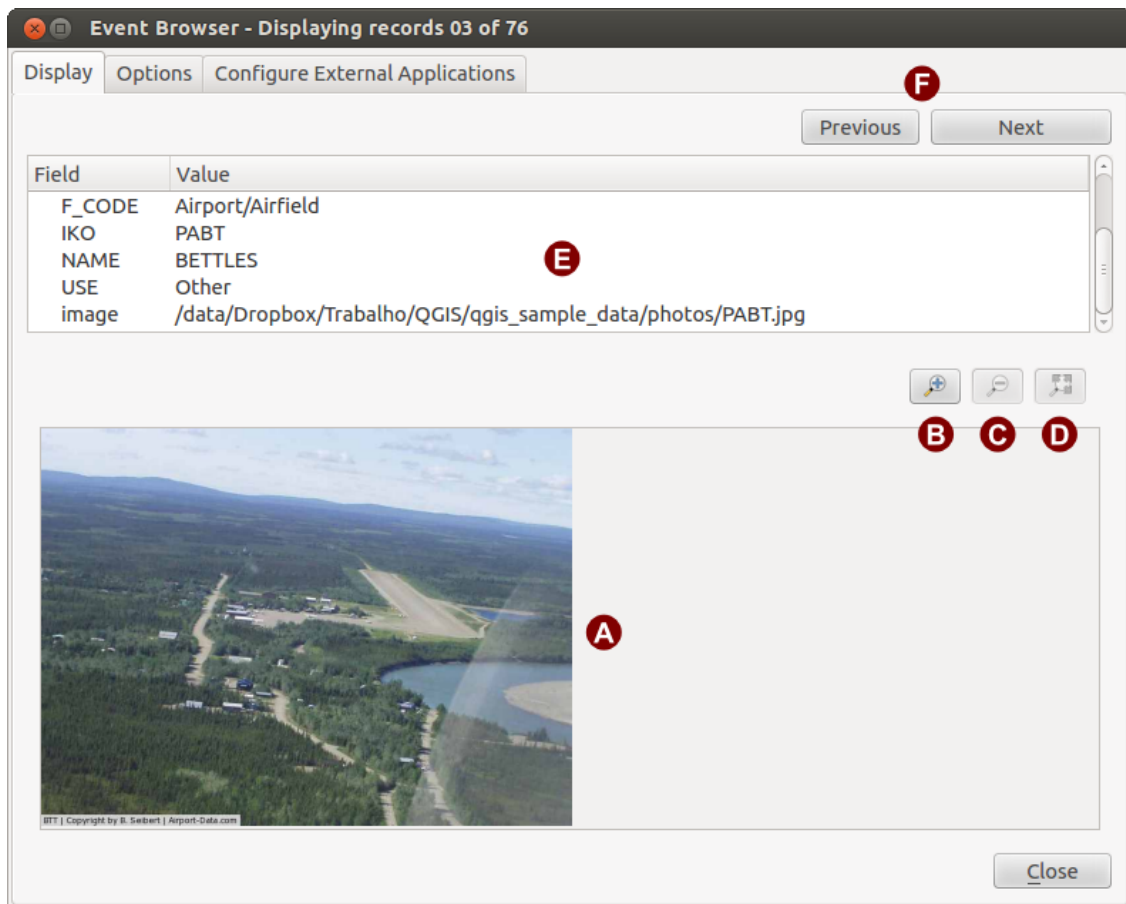
Peluncuran modul Navigasi Event

Meluncurkan Modul Navigasi Kegiatan, klik *Basisdata* → *eVis* → *eVis Navigasi Kegiatan*. Akan membuka jendela *Navigasi Kegiatan Umum*.

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.

Memahami jendela Tampilan

Melihat jendela *Tampilan*, klik tab *Tampilan* di jendela *Navigasi Kegiatan*. Jendela *Tampilan* digunakan untuk melihat foto-foto geocoded dan data atribut yang terkait.

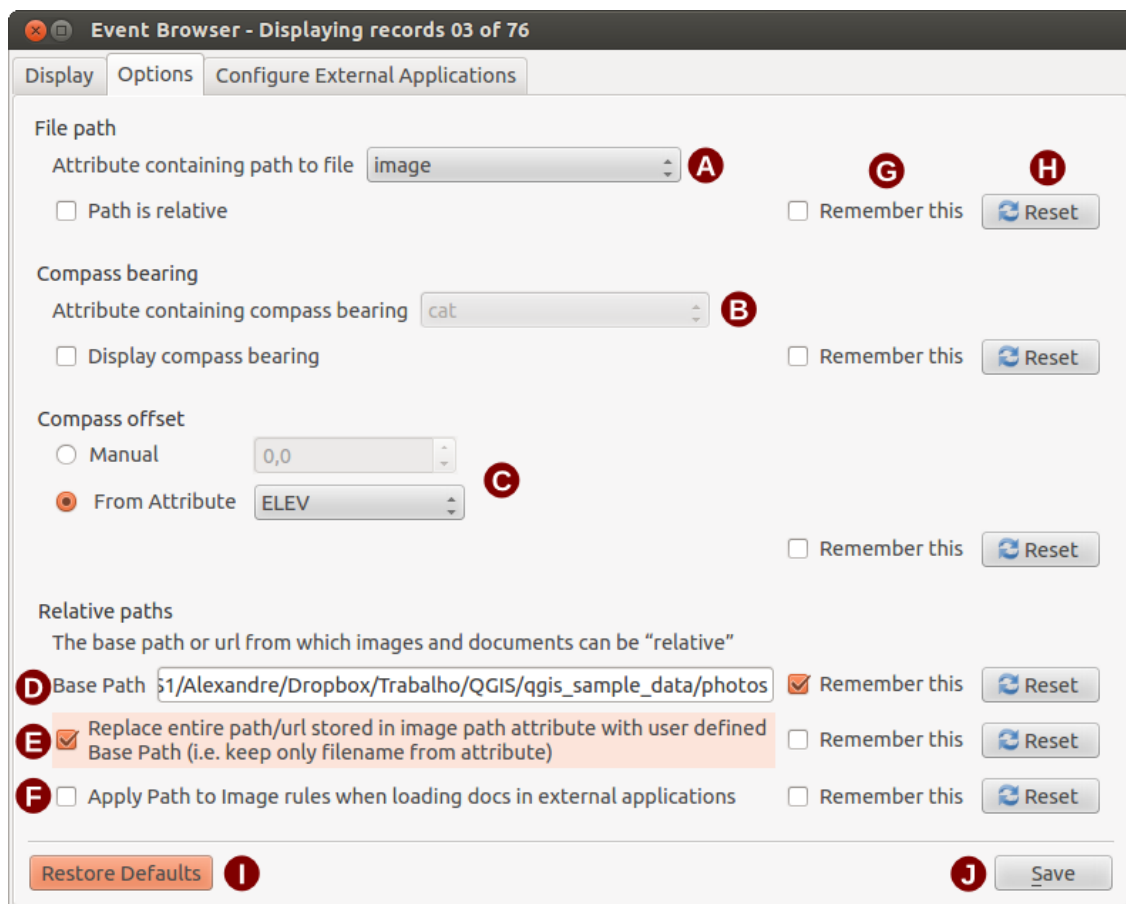


Gambar 20.9: Tampilan jendela *eVis*

1. **Jendela Tampilan:** Sebuah jendela di mana foto itu akan muncul.
2. **Tombol memperbesar:** Memperbesar untuk melihat lebih detail. Jika seluruh gambar tidak dapat ditampilkan di jendela layar, scroll bar akan muncul di sisi kiri dan bawah jendela untuk memungkinkan Anda untuk menjelajah gambar.
3. **Tombol memperkecil:** Memperkecil untuk melihat lebih banyak daerah.

4. **Perbesar ke seluruh area:** Menampilkan secara penuh dari foto.
5. **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.
6. **Tombol navigasi:** Gunakan tombol Sebelumnya dan Berikutnya untuk memuat fitur sebelumnya atau berikutnya ketika lebih dari satu fitur yang dipilih.

Memahami jendela Opsi



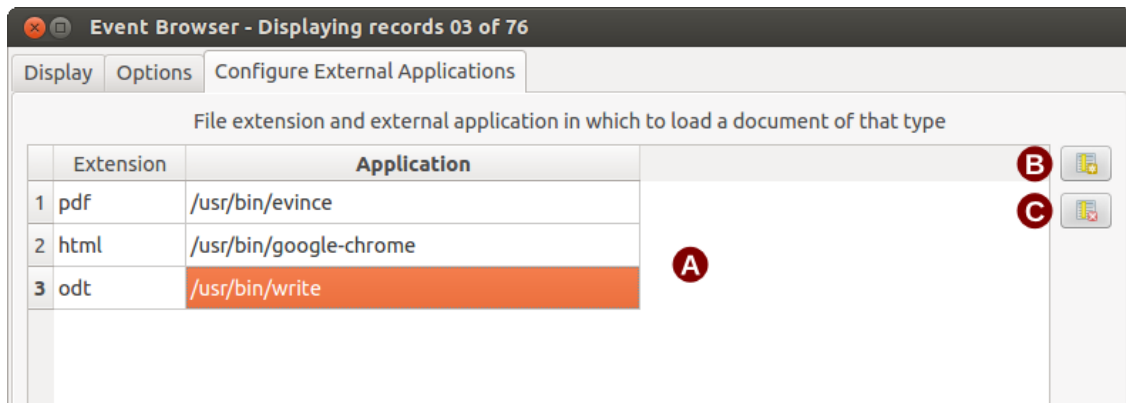
Gambar 20.10: Jendela Opsi *eVis*

1. **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 *Menentukan lokasi dan nama dari sebuah foto* below.
2. **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.
3. **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.

4. **Path direktori dasar:** Path dasar ke mana path relatif didefinisikan dalam [Figure_eVis_2](#) (A) akan ditambahkan.
5. **Ganti path:** Jika kotak centang ini diperiksa, hanya nama berkas dari A akan ditambahkan ke basis path.
6. **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.
7. **Ingat pengaturan:** Jika kotak centang dicentang, nilai-nilai untuk parameter terkait akan disimpan untuk sesi berikutnya ketika jendela ditutup atau bila tombol **** [Simpan] **** di bawah ditekan.
8. **Atur ulang nilai:** Mengatur ulang nilai pada baris ini ke pengaturan default.
9. **Kembali ke default:** Ini akan mengatur ulang semua kolom ke pengaturan default. Memiliki efek yang sama seperti mengklik tombol **[Atur Ulang]**.
10. **Simpan:** Akan menyimpan pengaturan tanpa menutup panel *Opsi*.

Memahami jendela Konfigurasi Aplikasi Eksternal



Gambar 20.11: Jendela Aplikasi Eksternal *eVis*

1. **Tabel referensi berkas:** Sebuah tabel yang berisi jenis berkas yang dapat dibuka dengan menggunakan *eVis*. Setiap jenis berkas membutuhkan ekstensi berkas dan path ke aplikasi yang dapat membuka jenis berkas. Hal ini memberikan kemampuan membuka berbagai berkas seperti film, rekaman suara, dan dokumen teks, bukan hanya gambar.
2. **Tambah jenis berkas baru:** Tambahkan jenis berkas baru dengan ekstensi yang unik dan path untuk aplikasi yang dapat membuka berkas.
3. **Hapus baris saat ini:** Menghapus jenis berkas yang disorot dalam tabel dan didefinisikan oleh ekstensi berkas dan path ke sebuah aplikasi terkait.

20.6.2 Menentukan lokasi dan nama dari sebuah foto

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	http://biodiversityinformatics.amnh.org/\evis_testdata/DSC_0170.JPG	10

20.6.3 Menentukan lokasi dan nama dokumen pendukung lainnya

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*).

20.6.4 Menggunakan Navigasi Event

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.

Menutup jendela *Navigasi kegiatan*, klik tombol **[Tutup]** dari jendela *Tampilan*.

20.6.5 Alat ID Event

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.

Peluncuran modul ID Event

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 ref:*evis_browser* section of this guide.

20.6.6 Sambungan Basisdata


Modul ‘Sambungan Basisdata’ menyediakan alat untuk menghubungkan dan query basisdata atau sumber daya ODBC lainnya, seperti 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.

Peluncuran modul Sambungan Basisdata

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.

Menyambung ke basisdata

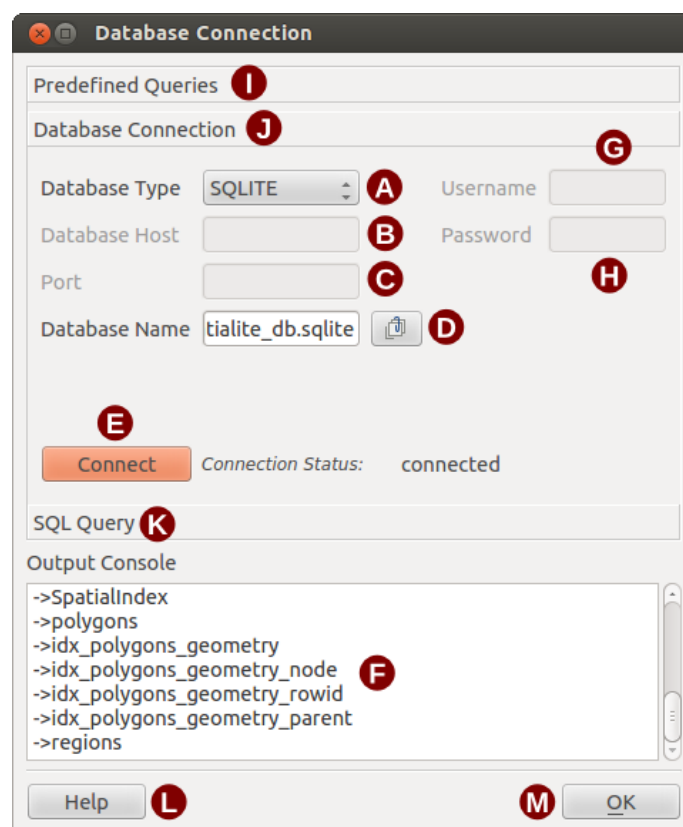
Click on the *Database Connection* tab to open the database connection interface. Next, use the *Database Type*  combo box to select the type of database that you want to connect to. If a password or username is required, that information can be entered in the *Username* and *Password* textboxes.

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”.

Masukkan nama basisdata di kotak teks *Nama Basisdata*. Jika Anda memilih ‘ODBC’ sebagai tipe basisdata, Anda harus memasukkan nama sumber data.

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.

1. **Type database:** Sebuah daftar drop-down untuk menentukan jenis basisdata yang akan digunakan.
2. **Host Basisdata:** Nama host basisdata.
3. **Port:** Nomor port jika MySQL atau PostgreSQL tipe database yang dipilih.
4. **Nama Basisdata:** Nama dari basisdata.
5. **Sambung:** Sebuah tombol untuk menghubungkan ke basisdata menggunakan parameter yang didefinisikan di atas.
6. **Keluaran Konsol:** Jendela konsol di mana pesan yang terkait dengan pengolahan akan ditampilkan.
7. **Namapengguna:** Namapengguna/Username untuk digunakan ketika basisdata dilindungi kata sandi/password.
8. **Kata sandi:** Kata sandi/Password digunakan ketika basisdata dilindungi kata sandi.
9. **Query yang ditentukan:** Tab untuk membuka jendela “Query yang ditentukan”.
10. **Sambungan Basisdata:** Tab membuka jendela “Sambungan Basisdata”.
11. **Query SQL:** Tab membuka jendela “Query SQL”.
12. **Bantuan:** Tampilan bantuan dalam jaringan.
13. **OK:** Tutup jendela utama “Sambungan Basisdata”.



Gambar 20.12: Jendela Sambungan Basisdata eVis

Menjalankan query SQL

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 <http://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.

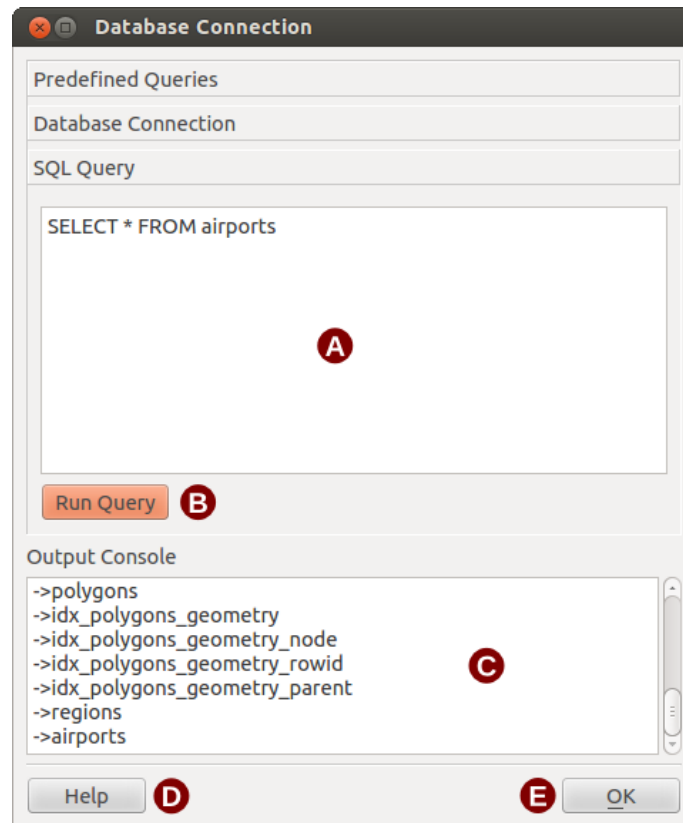
Di jendela *Pilih Berkas Basisdata*, masukkan nama lapisan yang akan dibuat dari hasil query dalam kotak teks *Nama Lapisan Baru*.

1. **Jendela Teks Query SQL:** Sebuah layar untuk mengetik query-query SQL.
2. **Jalankan Query:** Tombol untuk mengeksekusi query dimasukkan dalam *Jendela Query SQL*.
3. **Jendela Konsol:** Jendela konsol di mana pesan yang terkait dengan pengolahan akan ditampilkan.
4. **Bantuan:** Tampilan bantuan dalam jaringan.
5. **OK:** Menutup jendela *Sambungan Basisdata*.

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...'

Tip: Membuat lapisan vektor dari Microsoft Excel Worksheet





Gambar 20.13: Tab query SQL eVis

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.

Menjalankan query-query yang ditentukan

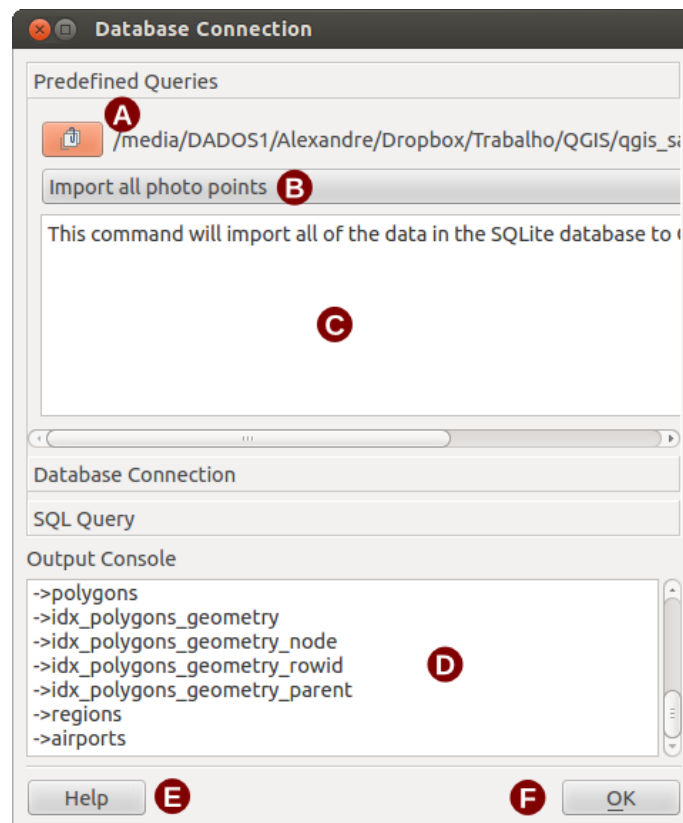
With predefined queries, you can select previously written queries stored in XML format in a file. This is particularly helpful if you are not familiar with SQL commands. Click on the *Predefined Queries* tab to display the predefined query interface.

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.

1. **Buka Berkas:** Meluncurkan “Buka Berkas” navigasi berkas untuk mencari berkas XML memegang query yang telah ditetapkan.



Gambar 20.14: Tab *eVis* Query Ditentukan

2. **Query Ditentukan:** Sebuah daftar drop-down dengan semua query yang didefinisikan oleh query berkas XML yang telah ditetapkan.
3. **Deskripsi Query:** Sebuah deskripsi singkat dari query. Deskripsi ini adalah dari query yang telah ditentukan berkas XML.
4. **Jendela Konsol:** Jendela konsol di mana pesan yang terkait dengan pengolahan akan ditampilkan.
5. **Bantuan:** Tampilan bantuan dalam jaringan.
6. **OK:** Tutup jendela utama “Sambungan Basisdata”.

Format XML untuk *eVis* query yang ditentukan

Tag XML dibaca oleh *eVis*

Tag	Deskripsi
query	Mendefinisikan awal dan akhir dari pernyataan query.
deskripsi singkat	Sebuah deskripsi singkat dari query yang muncul dalam menu drop-down eVis.
deskripsi tipebasisdata	Penjelasan lebih rinci dari query ditampilkan dalam jendela teks Query yang Ditentukan. Tipe basisdata, yang didefinisikan dalam menu drop-down Tipe Basisdata dalam tab Sambungan Basisdata.
portbasisdata	Port sebagaimana didefinisikan dalam kotak teks Port di tab Sambungan Basisdata.
namabasisdata	Nama basisdata sebagaimana dimaksud dalam Nama Basisdata kotak teks dalam tab Sambungan Basisdata.
usernamebasisdata	Nama pengguna basisdata sebagai didefinisikan dalam kotak teks Nama pengguna dalam tab Sambungan Basisdata.
katasandibasisdata	Kata sandi basisdata seperti yang didefinisikan dalam kotak teks Kata sandi di tab Sambungan Basisdata.
sqlstatement	Perintah SQL.
sambungotomatis	Sebuah bendera ("true" "atau" false ") untuk menentukan apakah tag di atas harus digunakan secara otomatis terhubung ke basisdata tanpa menjalankan sambungan basisdata rutin dalam tab Sambungan Basisdata.

Contoh berkas XML lengkap dengan tiga query ditampilkan di bawah ini:

```
<?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>
    <databasetype>SQLITE</databasetype>
    <databasehost />
    <databaseport />
```

```

<databasename>C:\Workshop\Vis_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>









```

20.7 Plugin fTools

Tujuan dari plugin fTools Python adalah menyediakan one-stop sumber daya untuk banyak tugas GIS umum berbasis vektor, tanpa perlu software tambahan, perpustakaan, atau kompleks solusinya. Menyediakan paket pengelolaan data spasial dan fungsi analisis yang baik cepat dan fungsional.








fTools sekarang otomatis terpasang dan aktif dalam QGIS versi terbaru, dan seperti semua plugin, bisa dinonaktifkan dan diaktifkan menggunakan Pengelola Plugin (Lihat Bagian *load_core_plugin*). Ketika diaktifkan, plugin fTools menambahkan menu *Vektor* ke QGIS, menyediakan fungsi mulai dari Peralatan Analisis dan Riset ke Peralatan Geometri dan Geoprocessing, serta beberapa Peralatan Tata Kelola Data yang berguna.

20.7.1 Alat Analisis

Ikon	Alat	Kegunaan
	Jarak Matriks	Ukur jarak antara dua lapisan titik, dan hasil keluarannya a) Matriks jarak Square, b) Matriks jarak Linear, atau c) Ringkasan dari jarak. Dapat membatasi jarak ke k fitur terdekat.
	Sum panjang garis	Hitung jumlah total garis panjang untuk setiap poligon dari lapisan vektor poligon.
	Poin di poligon	Hitung jumlah poin yang terjadi pada setiap poligon dari masukan lapisan vektor poligon.
	Daftar nilai unik	Daftar semua nilai unik di dalam masukan kolom lapisan vektor
	Dasar statistik	Hitung statistik dasar (rata-rata, std dev, N, sum, CV) pada koom masukan.
	Analisis tetangga terdekat	Menghitung Statistik tetangga terdekat, menilai tingkat pengelompokan dalam lapisan titik vektor
	Koordinat Mean	Hitung pusat rata-rata normal atau tertimbang dari seluruh lapisan vektor, atau beberapa fitur berdasarkan kolom ID unik.
	Persimpangan garis	Cari persimpangan antara garis, dan hasil keluaran sebagai titik shapefile. Berguna untuk mencari persimpangan jalan atau sungai, mengabaikan persimpangan garis dengan panjang > 0.










Tabel Ftools 1: fTools Analysis tools

20.7.2 Peralatan Riset

Ikon	Alat	Kegunaan
	Seleksi Random	Secara acak memilih nomor n dari fitur, atau persentase n dari fitur.
	Seleksi acak dalam subset	Secara acak memilih fitur dalam subset berdasarkan kolom ID unik.
	Poin Acak	Menghasilkan poin pseudo-acak atas masukan lapisan yang diberikan.
	Poin reguler	Menghasilkan kotak biasa dari poin atas wilayah tertentu dan ekspor mereka sebagai titik shapefile.
	Kotak vektor	Menghasilkan kotak garis atau poligon yang didasarkan pada pengguna jaringan jarak tertentu.
	Pilih dari lokasi	Pilih fitur berdasarkan lokasi relatif mereka terhadap lapisan lain untuk membentuk pilihan baru, atau menambah atau mengurangi dari pilihan saat ini.
	Poligon dari lapisan batas	Buat layer poligon persegi panjang tunggal dari luasnya sebuah masukan lapisan raster atau lapisan vektor.













Tabel Ftools 2: fTools Research tools

20.7.3 Peralatan Geoprocessing

Ikon	Alat	Kegunaan
	Convex hull(s)	Buat convex hull minimum untuk masukan lapisan, atau berdasarkan kolom ID.
	Penyangga	Buat penyangga sekitar fitur berdasarkan jarak, atau kolom jarak.
	Memotong	Tampilan lapisan-lapisan seperti keluaran berisi daerah di mana kedua lapisan berpotongan.
	Penggabungan	Tampilan lapisan-lapisan seperti keluaran berisi daerah berpotongan dan non-berpotongan.
	Perbedaan simetrik	Tampilan lapisan seperti keluaran berisi daerah-daerah dari masukan dan perbedaan lapisan yang tidak berpotongan.
	Klip	Tampilan lapisan seperti keluaran berisi daerah yang bersinggungan lapisan klip.
	Perbedaan	Tampilan lapisan seperti keluaran yang berisi daerah yang tidak memotong lapisan klip.
	Larut	Menggabungkan fitur berdasarkan masukan bidang. Semua fitur dengan nilai masukan yang identik digabungkan untuk membentuk satu fitur tunggal.
	Menghilangkan poligon sepotong	Menggabungkan fitur yang dipilih dengan poligon tetangga dengan daerah terbesar atau batas umum terbesar.

Tabel Ftools 3: fTools Peralatan Geoprocessing






20.7.4 Peralatan Geometri

Ikon	Alat	Kegunaan
	Periksa validitas geometri	Periksa poligon untuk persimpangan, ditutup-lubang, dan memperbaiki pengurutan node.
	Ekspor/Tambah kolom geometri	Tambah lapisan vektor geometri ke lapisan poin (XCOORD, YCOORD), garis (LENGTH), atau poligon (AREA, PERIMETER).
	Polygon centroids	Menghitung centroid berlaku untuk setiap poligon dalam masukan lapisan poligon.
	Triangulasi Delaunay	Menghitung dan keluaran (sebagai poligon) triangulasi Delaunay dari masukan lapisan vektor titik.
	Poligon Voronoi	Menghitung poligon voronoi dari masukan lapisan vektor.
	Sederhanakan geometri	Generalisasi garis atau poligon dengan algoritma Douglas-Peucker dimodifikasi.
	Geometri densify	Densify garis atau poligon dengan menambahkan simpul.
	Multipart ke singgelpart	Mengkonversi fitur multipart ke beberapa fitur singgelpart. Membuat poligon dan garis sederhana.
	Singgelpart ke multipart	Merge beberapa fitur ke fitur tunggal berdasarkan kolom ID unik.
	Poligon ke garis	Mengkonversi poligon ke garis, poligon multipart untuk beberapa garis singgelpart.
	Garis ke poligon	Mengkonversi garis ke poligon, garis multi ke beberapa bagian poligon tunggal.
	Ekstrak simpul	Ekstrak simpul dari lapisan garis dan poligon dan keluaran mereka sebagai poin.

Tabel Ftools 4: fTools Peralatan Geometri

Catatan: Alat *Menyederhanakan geometri* dapat digunakan untuk menghapus simpul duplikat di garis dan poligon geometri, hanya mengatur parameter *Menyederhanakan toleransi* ke 0 dan ini akan melakukan trik.

20.7.5 Peralatan Tata Kelola Data

Ikon	Alat	Kegunaan
	Tentukan proyeksi sekarang	Tentukan CRS untuk shapefile yang CRS belum ditetapkan.
	Menggabung atribut sesuai lokasi	Menggabung dengan atribut tambahan ke lapisan vektor didasarkan pada hubungan spasial. Atribut dari satu lapisan vektor yang ditambahkan ke tabel atribut dari lapisan lain dan diekspor sebagai shapefile.
	Membagi lapisan vektor	Membagi lapisan masukan ke beberapa lapisan terpisah berdasarkan masukan kolom.
	Menggabung shapefile menjadi satu	Menggabungkan beberapa shapefile dalam folder ke dalam shapefile baru berdasarkan jenis lapisan (titik, garis, area).
	Membuat indeks spasial	Buat indeks spasial untuk yang didukung format OGR.

Tabel Ftools 5: fTools Peralatan Tata Kelola Data

20.8 Plugin Peralatan GDAL

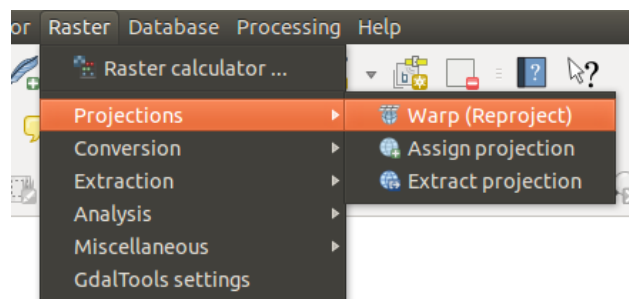
20.8.1 Apakah Peralatan GDAL itu?

Plugin Alat GDAL menawarkan GUI untuk koleksi alat dalam Geospatial Data Abstraction Library, <http://gdal.osgeo.org>. Ini adalah alat manajemen raster untuk query, re-proyeksi, warp dan menggabungkan berbagai format raster. Juga termasuk adalah alat untuk membuat lapisan kontur (vektor), atau relief shaded dari DEM raster, dan untuk membuat vrt (Virtual Raster Tile dalam format XML) dari koleksi satu atau lebih berkas raster. Alat ini tersedia bila plugin terpasang dan diaktifkan.

Perpustakaan GDAL




Perpustakaan GDAL terdiri dari satu set program baris perintah, masing-masing dengan daftar besar pilihan. Pengguna nyaman dengan menjalankan perintah dari terminal dapat memilih baris perintah, dengan akses pilihan set lengkap. Plugin alat GDAL menawarkan antarmuka yang mudah ke alat-alat, memperlihatkan hanya pilihan yang paling populer.

20.8.2 Daftar dari alat-alat GDAL








Gambar 20.15: Daftar menu *Alat-alat GDAL*



Proyeksi

 <p><i>Warp (Reproject)</i></p>	<p>Utilitas ini merupakan mosaicing gambar, proyeksi ulang dan warping utilitas. Program ini dapat proek ulang untuk setiap proyeksi yang didukung, dan juga dapat menerapkan GCPs disimpan dengan gambar jika gambar “raw” dengan informasi kontrol. Untuk informasi lebih lanjut Anda dapat membaca di situs GDAL http://www.gdal.org/gdalwarp.html</p>
 <p><i>Menetapkan proyeksi</i></p>	<p>Alat ini memungkinkan menetapkan proyeksi untuk raster yang sudah bergeoreferensi tetapi informasi proyeksi keliru. Juga dengan membantu mengubah definisi proyeksi yang ada. Kedua berkas tunggal dan mode batch didukung. Untuk informasi lebih lanjut silakan kunjungi halaman utilitas di situs GDAL http://www.gdal.org/gdalwarp.html</p>
 <p><i>Ekstrak proyeksi</i></p>	<p>Utilitas ini membantu Anda untuk mengekstrak informasi proyeksi dari berkas masukan. Jika Anda ingin mengambil proyeksi dari seluruh direktori Anda dapat menggunakan mode Batch. Ini menciptakan kedua berkas <code>.prj</code> dan <code>.wld</code>.</p>







Konversi

 <i>Rasterize</i>	<p>Program ini membakar vektor geometri (titik, garis dan poligon) ke pita raster dari gambar raster. Vektor dibaca dari OGR yang didukung format vektor. Perhatikan bahwa data vektor harus sama sistem koordinat dengan data raster; pada proyeksi ulang fly tidak disediakan. Untuk informasi lebih lanjut lihat http://www.gdal.org/gdal_rasterize.html</p>
 <i>Polygonize</i>	<p>Utilitas ini menciptakan vektor poligon untuk semua daerah piksel terhubung di raster berbagi nilai umum pixel. Setiap poligon dibuat dengan atribut yang menunjukkan nilai pixel poligon itu. Utilitas akan menciptakan keluaran sumber data vektor jika tidak sudah ada, default ke format shapefile ESRI. Lihat juga http://www.gdal.org/gdal_polygonize.html</p>
 <i>Terjemahan</i>	<p>Utilitas ini dapat digunakan untuk mengkonversi data raster antar format yang berbeda, berpotensi melakukan beberapa operasi seperti subsettings, resampling, dan rescaling piksel dalam proses. Untuk informasi Anda bisa membaca di http://www.gdal.org/gdal_translate.html</p>
 <i>RGB ke PCT</i>	<p>Utilitas ini akan menghitung tabel warna-pseudo yang optimal untuk citra RGB yang diberikan dengan menggunakan algoritma cut median pada RGB histogram downsampled. Kemudian mengubah gambar menjadi gambar berwarna-pseudo menggunakan tabel warna. Konversi ini menggunakan Floyd-Steinberg dithering (difusi error) untuk memaksimalkan keluaran visual gambar kualitas. Utilitas ini juga dijelaskan pada http://www.gdal.org/rgb2pct.html</p>
 <i>PCT ke RGB</i>	<p>Utilitas ini akan mengkonversi sebuah pita pseudocolor pada berkas masukan ke berkas keluaran RGB dari format yang diinginkan. Untuk informasi lebih lanjut lihat http://www.gdal.org/pct2rgb.html</p>






Ekstraksi

 <i>Kontur</i>	<p>Program ini menghasilkan berkas kontur vektor dari masukan data elevasi raster (DEM). Pada http://www.gdal.org/gdal_contour.html Anda bisa menemukan banyak informasi.</p>
 <i>Clipper</i>	<p>Utilitas ini memungkinkan untuk klip (ekstrak bagian) raster menggunakan batas yang dipilih atau berdasarkan masker lapisan batas. Informasi lebih lanjut dapat ditemukan di http://www.gdal.org/gdal_translate.html.</p>

Analisis

 <i>Sieve</i>	<p>Utilitas ini menghilangkan raster poligon lebih kecil dari ukuran ambang batas yang disediakan (dalam piksel) dan menggantikan mereka dengan nilai piksel dari tetangga poligon terbesar. Hasilnya dapat ditulis kembali ke pita raster yang sudah ada, atau disalin ke dalam sebuah berkas baru. Untuk informasi lebih lanjut, lihat http://www.gdal.org/gdal_sieve.html .</p>
 <i>Hampir Hitam</i>	<p>Utilitas ini akan memindai gambar dan mencoba untuk mengatur semua piksel yang hampir hitam (atau hampir putih) di sekitar tepi persis hitam (atau putih). Hal ini sering digunakan untuk “fix up” berkurangnya kompresi foto udara sehingga warna piksel dapat diperlakukan sebagai transparan ketika mosaicing. Lihat juga http://www.gdal.org/nearblack.html .</p>
 <i>Isi tidak ada data</i>	<p>Utilitas ini mengisi seleksi raster area (biasanya tidak ada data area) dengan interpolasi dari piksel yang valid di sekitar tepi area. Lihat juga http://www.gdal.org/gdal_fillnodata.html .</p>
 <i>Kedekatan</i>	<p>Utilitas ini menghasilkan peta raster kedekatan menunjukkan jarak dari pusat setiap piksel ke pusat piksel terdekat diidentifikasi sebagai piksel sasaran. Target piksel adalah pada raster sumber yang nilai raster piksel di set dari nilai target piksel. Untuk informasi lebih lanjut, lihat http://www.gdal.org/gdal_proximity.html .</p>
 <i>Kisi (Interpolasi)</i>	<p>Utilitas ini menciptakan kisi biasa (raster) dari data yang tersebar dibaca dari sumber data OGR. Masukan data akan diinterpolasi untuk mengisi node kisi dengan nilai-nilai, Anda dapat memilih dari berbagai metode interpolasi. Utilitas ini juga dijelaskan di situs GDAL http://www.gdal.org/gdal_grid.html .</p>
 <i>DEM (Terrain model)</i>	<p>Alat untuk menganalisis dan memvisualisasikan DEM. Hal ini dapat membuat relief shaded, lereng, aspek, warna relief, Indeks Terrain Ketidakrataaan, Indeks Posisi Topografi dan kekasaran peta dari setiap GDAL didukung elevasi raster. Untuk informasi lebih lanjut Anda dapat membaca pada http://www.gdal.org/gdaldem.html .</p>

Bermacam-macam

 <i>Build Virtual Raster (Katalog)</i>	<p>Program ini membangun VRT (Virtual Dataset) yang merupakan mosaik dari daftar masukan dataset GDAL. Lihat juga http://www.gdal.org/gdalbuildvrt.html .</p>
 <i>Gabung</i>	<p>Utilitas ini secara otomatis akan mosaik serangkaian gambar. Semua gambar harus dalam sistem koordinat yang sama dan memiliki sejumlah pencocokan pita, tetapi mereka mungkin tumpang tindih, dan pada resolusi yang berbeda. Di daerah-daerah tumpang tindih, gambar terakhir akan disalin lebih dari yang sebelumnya. Utilitas ini juga dijelaskan pada http://www.gdal.org/gdal_merge.html .</p>
 <i>Informasi</i>	<p>Utilitas ini berisi berbagai informasi tentang GDAL didukung dataset raster. Lihat juga http://www.gdal.org/gdalinfo.html .</p>
 <i>Gambaran Build</i>	<p>Utilitas gdaladdo dapat digunakan untuk membangun atau membangun kembali gambar ikhtisar untuk format berkas yang paling didukung dengan salah satu dari beberapa algoritma downsampling. Untuk informasi lebih lanjut, lihat http://www.gdal.org/gdaladdo.html .</p>
 <i>Indeks Tile</i>	<p>Utilitas ini membangun shapefile dengan catatan untuk setiap berkas masukan raster, atribut yang berisi nama berkas, dan geometri poligon menguraikan raster. Lihat juga http://www.gdal.org/gdaltindex.html .</p>

Pengaturan Peralatan GDAL

Gunakan dialog ini untuk menempelkan variabel GDAL Anda.

20.9 Plugin Georeferencer

Plugin Georeferencer adalah alat untuk menghasilkan berkas-berkas dunia untuk raster-raster. Hal ini memungkinkan Anda untuk membuat referensi raster dengan sistem koordinat geografis atau diproyeksikan dengan menciptakan GeoTiff baru atau dengan menambahkan sebuah berkas dunia dengan gambar yang ada. Pendekatan dasar Georeferensi raster adalah untuk menemukan titik-titik pada raster Anda secara akurat dan dapat menentukan koordinat mereka.

Fitur-fitur

Ikona	Tujuan	Ikona	Tujuan
	Buka raster		Memulai georeferensi
	Menghasilkan Skrip GDAL		Muat Poin-Poin GCP
	Simpan Poin GCP Sebagai		Pengaturan Transformasi
	Tambah Poin		Hapus Poin
	Pindahkan Poin GCP		Pan
	Perbesar		Perkecil
	Perbesar Ke Lapisan		Perbesaran Terakhir
	Perbesar Selanjutnya		Tautan Georeferencer ke QGIS
	Tautan QGIS ke Georeferencer		Peregangan histogram penuh
	Peregangan histogram lokal		

Tabel Georeferensi 1: Alat-Alat Georeferensi

20.9.1 Prosedur biasa

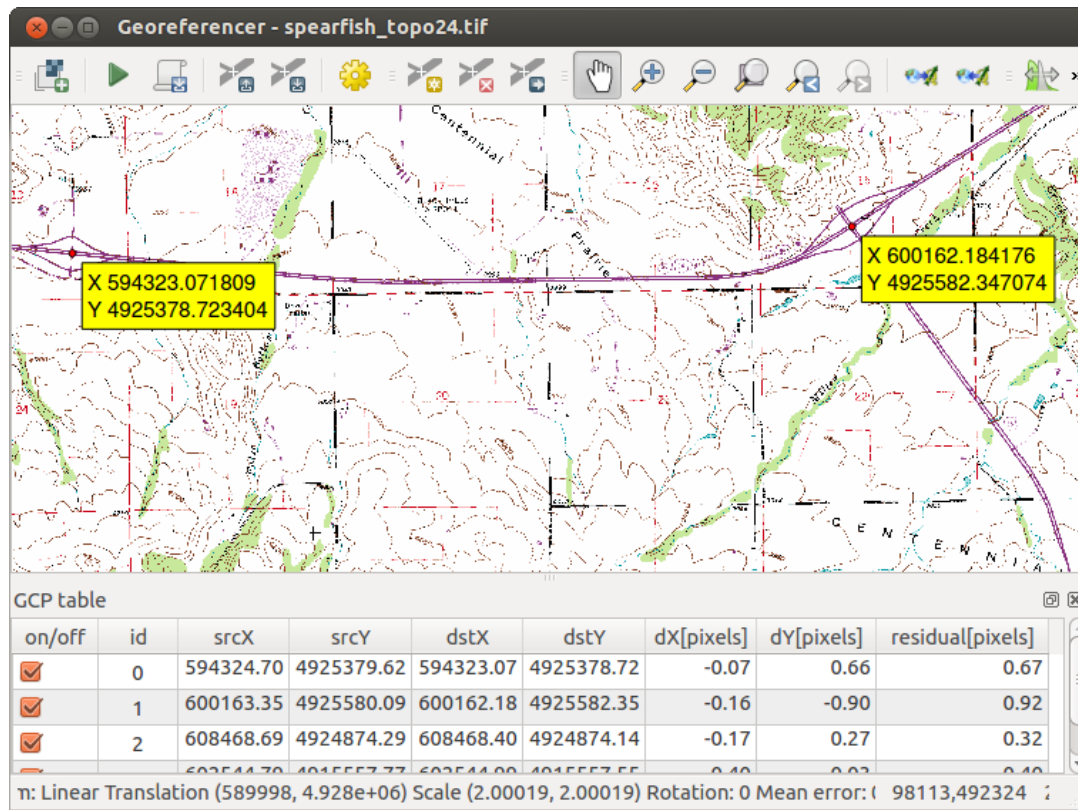
Sebagai koordinat X dan Y (DMS (dd mm ss.ss), DD (dd.dd) atau koordinat diproyeksikan (mmmm.mm)), yang sesuai dengan titik yang dipilih pada gambar, dua alternatif prosedur dapat digunakan:

- Raster itu sendiri kadang-kadang memberikan salib dengan koordinat “written” pada gambar. Dalam hal ini Anda dapat memasukkan koordinat secara manual.
- Menggunakan lapisan sudah bergeoreferensi, hal ini dapat berupa data vektor atau raster yang berisi benda-benda yang sama/fitur yang Anda miliki pada gambar yang ingin Anda Georeferensi dan proyeksi. Dalam hal ini Anda dapat memasukkan koordinat dengan mengklik pada dataset referensi dimuat dalam kanvas peta QGIS.

Prosedur yang biasa untuk Georeferensi gambar memilih beberapa poin pada raster, menentukan koordinat mereka, dan memilih jenis transformasi yang relevan. Berdasarkan parameter masukan dan data, plugin akan menghitung parameter berkas dunia. Semakin banyak titik koordinat Anda berikan, akan semakin baik hasilnya.

Langkah pertama jalankan QGIS, muat Plugin Georeferencer (lihat *The Plugins Dialog*) dan klik *Raster* → *Georeferencer* , yang muncul di menu bar QGIS. Dialog Plugin Georeferencer akan tampil seperti [figure_georeferencer_1](#).

Untuk contoh ini, kita menggunakan toposheet South Dakota dari SDGS. Hal ini nantinya dapat divisualisasikan bersama dengan data dari lokasi GRASS `spearfish60`. Anda dapat mengunduh toposheet di sini: http://grass.osgeo.org/sampled/spearfish_toposheet.tar.gz.

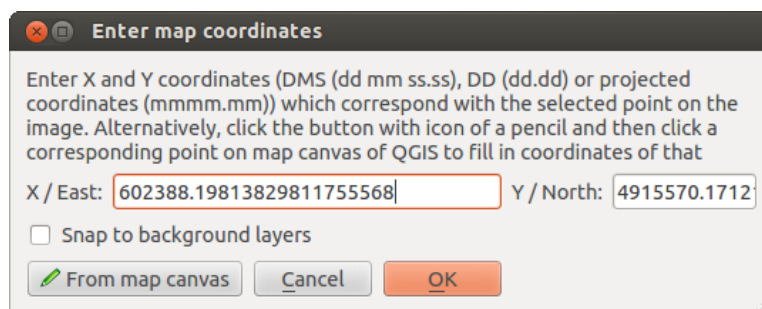



Gambar 20.16: Dialog Plugin Georeferencer



Memasukkan titik-titik kontrol lapangan (GCPs)

1. Untuk memulai georeferensi sebuah raster rintisan, kita harus memuat dengan menggunakan tombol . Raster akan muncul di wilayah kerja utama dialog. Setelah raster dimuat, kita bisa mulai untuk memasukkan titik-titik acuan.
2. Menggunakan tombol **Tambah Poin**, menambahkan poin-poin untuk wilayah kerja utama dan memasukkan koordinat mereka (lihat Gambar [figure_georeferencer_2](#)). Untuk prosedur ini, Anda memiliki tiga pilihan:
 - Klik pada poin di gambar raster dan masukkan koordinat X dan Y secara manual.
 - Klik pada poin gambar raster dan pilih tombol **Dari kanvas peta** untuk menambahkan koordinat X dan Y coordinates dengan bantuan peta georeferensi sudah dimuat dalam kanvas peta QGIS.
 - Dengan tombol , Anda bisa menggeser GCP di dalam jendela, jika mereka berada di tempat yang salah.
3. Lanjutkan memasukkan titik-titik. Anda harus memiliki setidaknya 4 titik, dan semakin banyak koordinat yang dapat Anda berikan, akan semakin baik hasilnya. Ada alat tambahan pada dialog plugin untuk memperbesar dan menggeser wilayah kerja dalam rangka untuk mencari satu set titik GCP relevan.

Poin-poin yang ditambahkan ke peta akan disimpan dalam sebuah berkas teks yang terpisah (`[filename].points`) biasanya bersama-sama dengan gambar raster. Hal ini memungkinkan kita untuk membuka kembali plugin Georeferencer di kemudian hari dan menambah poin baru atau menghapus yang sudah ada untuk mengoptimalkan hasilnya. Berkas poin berisi nilai-nilai dalam bentuk: `mapX, mapY,`

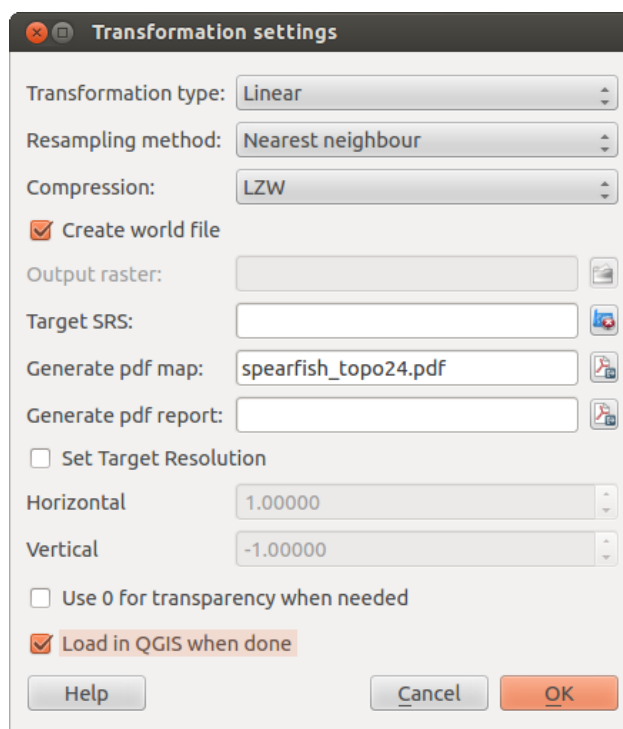



Gambar 20.17: Tambahkan titik-titik ke gambar raster 

pixelX, pixelY. Anda bisa menggunakan tombol-tombol  Muat poin-poin GCP dan  Simpan poin GCP sebagai untuk mengelola berkas.

Pengaturan mendefinisikan transformasi

Setelah Anda telah menambahkan GCPs Anda ke gambar raster, Anda harus menentukan pengaturan transformasi untuk proses georeferensi.



Gambar 20.18: Pengaturan mendefinisikan transformasi 

Algoritma Transformasi Tersedia

Tergantung pada berapa banyak titik kontrol lapangan yang telah direkam, Anda mungkin ingin menggunakan algoritma transformasi yang berbeda. Pilihan algoritma transformasi juga tergantung pada jenis dan kualitas masukan data dan jumlah distorsi geometris Anda bersedia untuk memasukkan kepada hasil akhir.

Saat ini, tersedia *Tipe Transformasi* :

- Algoritma **Linear** adalah digunakan untuk membuat berkas-dunia, dan berbeda dari algoritma yang lain, karena tidak benar-benar mengubah raster. Algoritma ini kemungkinan besar tidak akan cukup jika Anda berurusan dengan bahan yang dipindai.
- Transformasi **Helmert** melakukan scaling dan rotasi transformasi sederhana.
- Algoritma 1-3 **Polynomial** yang paling banyak digunakan algoritma yang diperkenalkan untuk mencocokkan sumber dan tujuan poin kontrol lapangan. Paling banyak digunakan algoritma polinomial adalah urutan kedua transformasi polinomial, yang memungkinkan beberapa kelengkungan. Urutan pertama transformasi polinomial (affine) mempertahankan collinearity dan memungkinkan scaling, terjemahan dan rotasi saja.
- Algoritma **Thin Plate Spline** (TPS) adalah metode georeferencing yang lebih modern, yang mampu memperkenalkan deformasi lokal dalam data. Algoritma ini berguna ketika kualitas asli sangat rendah sebagai rujukan geografis.
- Transformasi **Proyektif** rotasi linear dan terjemahan koordinat.

Tentukan metode Sampel ulang

Jenis resampling Anda memilih kemungkinan akan tergantung pada data masukan Anda dan tujuan akhir dari latihan. Jika Anda tidak ingin mengubah statistik dari gambar, Anda mungkin ingin memilih 'Nearest neighbour', sedangkan 'resampling Cubic' kemungkinan akan memberikan hasil yang lebih rapi.

Hal ini dimungkinkan untuk memilih lima metode resampling yang berbeda:

1. Nearest neighbour
2. Linear
3. Cubic
4. Cubic Spline
5. Lanczos

Tentukan pengaturan transformasi

Ada beberapa opsi yang perlu didefinisikan untuk keluaran raster georeferensi.

- Kotak centang *Buat berkas dunia* hanya tersedia jika Anda memutuskan untuk menggunakan jenis transformasi linear, karena ini berarti bahwa gambar raster sebenarnya tidak akan berubah. Dalam hal ini, kolom *Keluaran raster* tidak aktif, karena hanya berkas dunia baru yang akan dibuat.
- Untuk semua jenis transformasi lain Anda harus menentukan *Keluaran raster*. Sebagai default berkas baru ([filename]_modified) akan dibuat dalam folder yang sama bersama-sama dengan gambar raster asli.
- Sebagai langkah berikutnya Anda harus menentukan *Target SRS* (Spatial Reference System) untuk georeferensi raster (lihat bagian *label_projections*).
- Jika Anda ingin, Anda bisa **menghasilkan peta pdf** dan juga **laporan pdf**. Laporan ini mencakup informasi tentang parameter transformasi yang digunakan. Sebuah gambar dari residual dan daftar dengan semua GCP dan eror RMS.
- Selanjutnya Anda dapat mengaktifkan kotak centang *Set Target Resolusi* dan tentukan pixel resolusi dari keluaran raster. Resolusi horisontal dan vertikal default adalah 1.
- *Gunakan 0 untuk transparan jika dibutuhkan* bisa diaktifkan, jika pixel dengan nilai 0 akan divisualisasikan transparan. Dalam contoh toposheet kami semua bidang putih akan jadi transparan.
- Akhirnya *muat di QGIS saat selesai* muat keluaran raster secara otomatis ke dalam kanvas Peta QGIS ketika transformasi sudah dilakukan.


Tampilkan dan menyesuaikan properti raster

Mengklik ada dialog *Raster properti* didalam menu *Pengaturan* membuka properti raster dari lapisan yang ingin Anda Georeferensi.

Konfigurasi georeferensi


- Anda dapat menentukan apakah Anda ingin menunjukkan koordinat GCP dan/atau ID.
- Sebagai unit sisa piksel dan unit peta dapat dipilih.
- Untuk laporan PDF margin kiri dan kanan dapat ditentukan dan Anda juga dapat mengatur ukuran kertas untuk peta PDF.
- Akhirnya Anda bisa mengaktifkan *show georeferencer window docked*.

Menjalankan transformasi



Setelah semua GCPs telah dikumpulkan dan semua pengaturan transformasi ditentukan, tekan tombol  Mulai georeferencing untuk membuat raster baru bergeoreferensi.

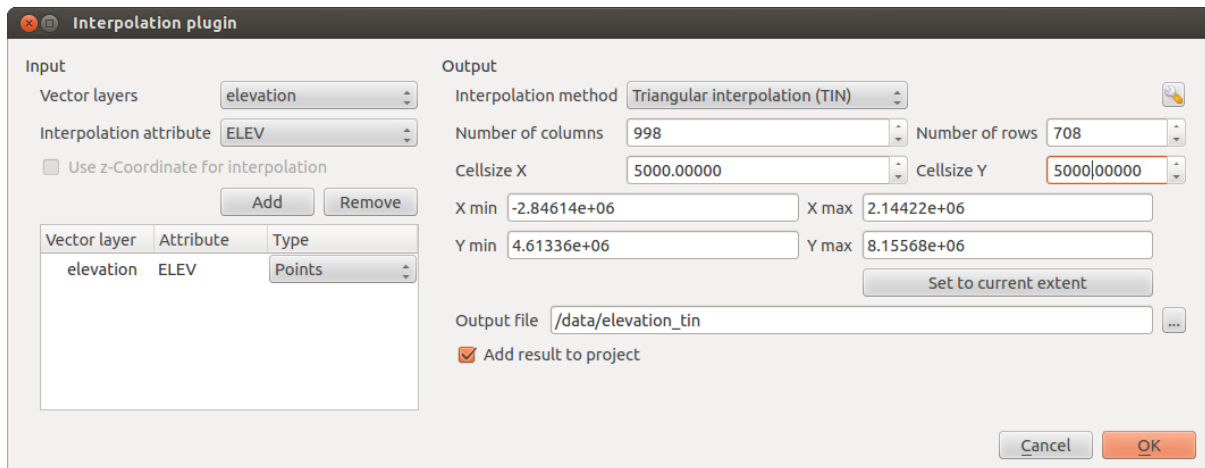
20.10 Plugin Interpolasi

Plugin Interpolasi dapat digunakan untuk membuat interpolasi TIN atau IDW dari suatu layer vektor titik. Plugin ini sangat sederhana serta menyediakan tampilan antarmuka yang intuitif untuk membuat layer raster terinterpolasi (lihat [Figure_interpolation_1](#)). Plugin ini memerlukan parameter-parameter berikut sebelum dapat dijalankan:

- Masukan **Lapisan-lapisan Vektor**: Tentukan masukan lapisan vektor titik dari daftar lapisan titik yang dimuat. Jika beberapa lapisan ditentukan, maka data dari semua lapisan yang digunakan untuk interpolasi. Catatan: Hal ini dimungkinkan untuk menyisipkan baris atau poligon untuk triangulasi, dengan menentukan “titik”, “garis struktur” atau “baris break” dalam combobox *Tipe* .
- **Atribut interpolasi**: Pilih kolom atribut untuk digunakan dalam interpolasi atau aktifkan checkbox *Gunakan koordinat-Z* untuk menggunakan nilai Z yang tersimpan pada layer.
- **Metode Interpolasi**: Pilih metode interpolasi. Pilihannya antara lain ‘Triangulated Irregular Network (TIN)’ atau ‘Inverse Distance Weighted (IDW)’.
- **Jumlah kolom/baris**: Tentukan jumlah baris dan kolom untuk keluaran berkas raster.
- **File keluaran**: Masukkan nama untuk file raster yang dihasilkan.
- *Tambah hasil ke proyek* untuk memuat hasil kedalam kanvas peta.

20.10.1 Menggunakan plugin interpolasi

1. Mulai QGIS dan muat lapisan vektor titik (seperti `elevp.csv`).
2. Muat plugin interpolasi di Pengelola Plugin (lihat Bagian *managing_plugins*) dan klik *:menuselection: ‘Raster → Interpolasi →  Interpolasi*, yang muncul dalam menu toolbar QGIS. Dialog plugin Interpolasi ditunjukkan dalam [Figure_interpolation_1](#).
3. Pilih layer masukan (seperti `elevp` ) dan kolom (seperti `ELEV`) untuk interpolasi.
4. Pilih metode interpolasi (seperti ‘Triangulated Irregular Network (TIN)’), dan masukkan ukuran sel 5000 sebagaimana juga nama file keluaran (seperti, `elevation_tin`).




Gambar 20.19: Plugin Interpolasi 



5. Klik [OK].

20.11 Plugin Mengedit Diluar Jaringan (Offline)


Untuk pengumpulan data merupakan situasi umum bekerja dengan laptop atau ponsel di luar jaringan (offline) di lapangan. Setelah kembali ke jaringan, perubahan perlu disinkronkan dengan sumber data master, misalnya basisdata PostGIS. Jika beberapa orang bekerja secara bersamaan pada dataset yang sama, sulit untuk menggabungkan hasil edit dengan tangan, bahkan jika orang tidak mengubah fitur yang sama.

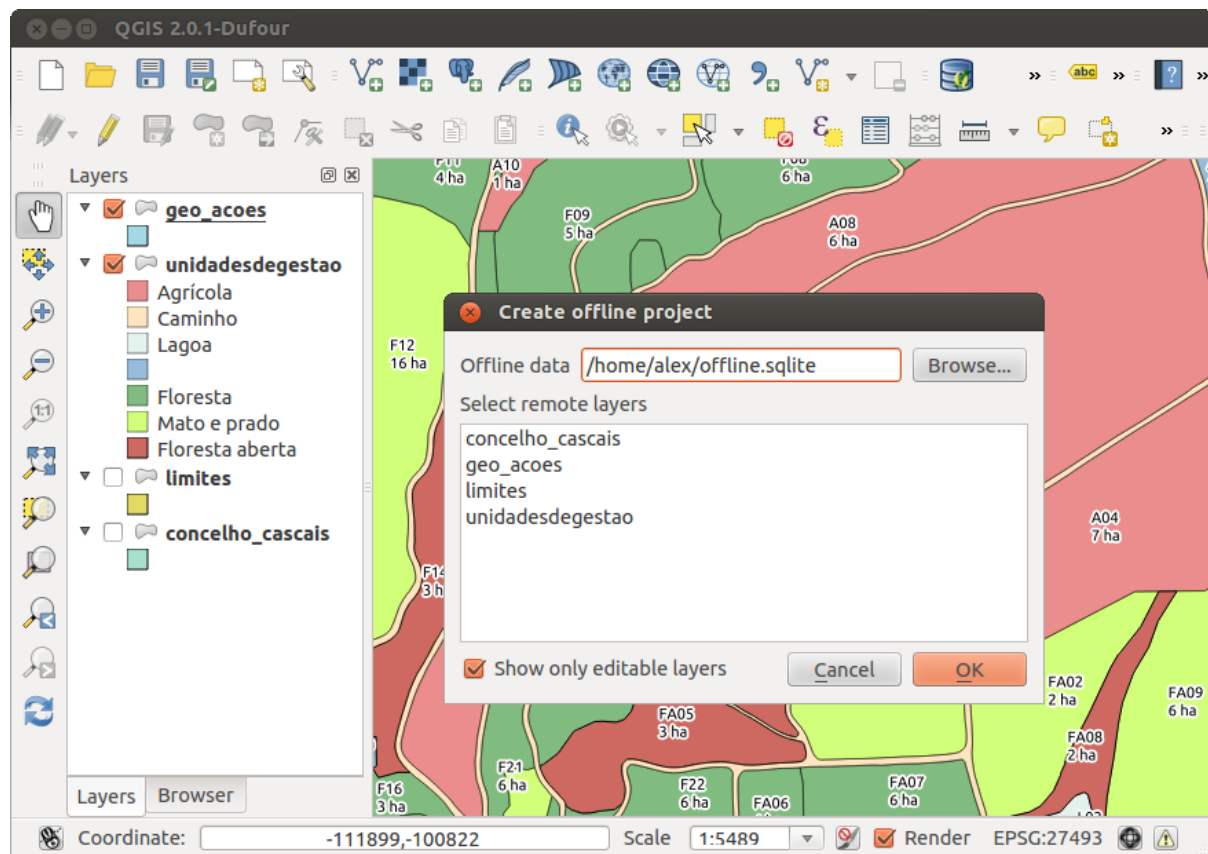
Plugin  Mengedit Offline Plugin secara otomatis melakukan sinkronisasi dengan menyalin isi dari sumber data (biasanya PostGIS atau WFS-T) ke basisdata SpatiaLite dan menyimpan suntingan luar jaringan (offline) ke tabel khusus. Setelah terhubung ke jaringan lagi, memungkinkan untuk menerapkan secara suntingan luar jaringan (offline) ke dataset induk.

20.11.1 Menggunakan plugin

- Buka beberapa lapisan vektor, (misal dari sumber data PostGIS atau WFS-T).
- Menyimpannya sebagai proyek.
- Pergi ke *Basisdata* → *Mengedit Luring* →  *Konversikan ke proyek luring* dan pilih lapisan untuk menyimpan. Isi dari lapisan disimpan ke tabel SpatiaLite.
- Edit lapisan luar jaringan.
- Setelah terhubung lagi, mengunggah perubahan menggunakan *Basisdata* → *Mengedit luring* →  *Sinkronisasi*.

20.12 Plugin Spasial Oracle GeoRaster

Dalam basisdata Oracle, data raster dapat disimpan dalam obyek SDO_GEORASTER tersedia ekstensi Oracle Spasial. Dalam QGIS, plugin  Spasial Oracle GeoRaster ini didukung oleh GDAL dan tergantung pada produk basisdata Oracle yang sedang dipasang dan bekerja pada mesin Anda. Sementara Oracle adalah perangkat lunak




Gambar 20.20: Membuat proyek luar jaringan dari lapisan PostGIS atau WFS

proprietary, mereka menyediakan perangkat lunak bebas untuk tujuan pengembangan dan pengujian. Berikut adalah salah satu contoh sederhana bagaimana untuk memuat gambar raster ke GeoRaster:

```
$ gdal_translate -of georaster input_file.tif geor:scott/tiger@orcl
```

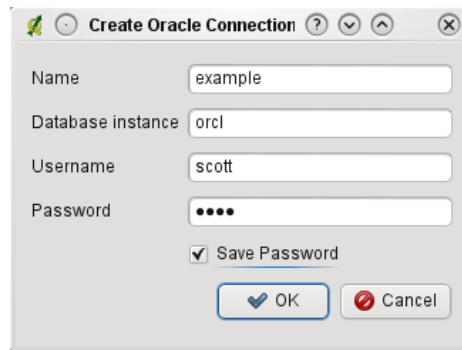
Hal ini akan memuat raster ke default tabel GDAL_IMPORT, sebagai kolom bernama RASTER.

20.12.1 Mengelola koneksi-koneksi

Pertama, Plugin Oracle GeoRaster harus diaktifkan menggunakan Pengelola Plugin (lihat *The Plugins Dialog*). Pertama kali Anda memuat GeoRaster dalam QGIS, Anda harus membuat koneksi/sambungan ke basisdata Oracle yang berisi data. Untuk melakukan hal ini, mulai dengan klik tombol toolbar  Tambah lapisan Oracle GeoRaster – ini akan membuka jendela dialog *Pilih Spasial Oracle GeoRaster*. Klik pada **[Baru]** membuka jendela dialog, dan menentukan parameter sambungan (Lihat *Figure_oracle_raster_1*):

- **Nama:** Masukkan nama dari sambungan basisdata.
- **Contoh basisdata:** Masukkan nama dari basisdata yang ingin Anda sambungkan.
- **Namapengguna:** Tentukan nama pengguna/username Anda sendiri yang akan Anda gunakan untuk mengakses basisdata.
- **Kata sandi:** Memasukkan kata sandi/password yang terkait dengan nama pengguna yang diperlukan untuk mengakses basisdata.

Sekarang, kembali jendela utama dialog *Spasial Oracle GeoRaster* (lihat *Figure_oracle_raster_2*), gunakan daftar drop-down untuk memilih satu sambungan, dan gunakan tombol **[Sambung]** untuk sambungan tersebut. Anda mungkin juga **[Edit]** koneksi dengan membuka dialog sebelumnya dan membuat perubahan pada informasi koneksi, atau gunakan tombol **[Hapus]** untuk menghapus sambungan dari daftar drop-down.



Gambar 20.21: Buat dialog koneksi Oracle

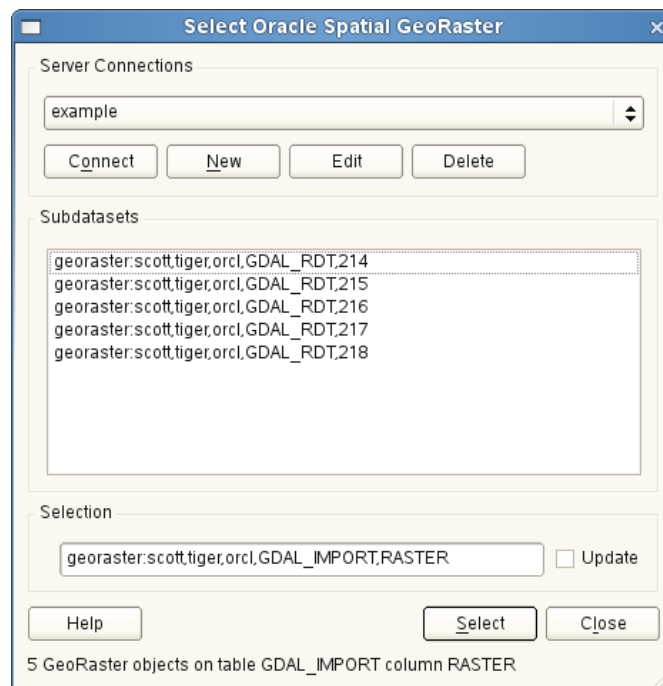
20.12.2 Memilih sebuah GeoRaster

Setelah sambungan telah ditetapkan, jendela subdatasets akan menampilkan nama-nama semua tabel yang berisi kolom GeoRaster dalam basisdata dalam format nama GDAL subdataset.

Klik pada salah satu subdatasets terdaftar dan kemudian klik pada **[Pilih]** memilih nama tabel. Sekarang daftar subdatasets lain akan menunjukkan nama-nama kolom GeoRaster di tabel itu. Biasanya sebuah daftar singkat, karena sebagian besar pengguna tidak akan memiliki lebih dari satu atau dua kolom GeoRaster di tabel yang sama.

Klik pada salah satu subdatasets terdaftar dan kemudian klik pada **[Pilih]** memilih salah satu kombinasi tabel/kolom. Dialog sekarang akan menampilkan semua baris yang berisi obyek-obyek GeoRaster. Perhatikan bahwa daftar subdataset sekarang akan menampilkan Tabel Data Raster dan pasangan Id Raster.

Setiap saat, masukan seleksi dapat diedit untuk bisa langsung ke GeoRaster yang diketahui atau kembali ke awal dan pilih nama tabel lain.



Gambar 20.22: Pilih dialog Oracle GeoRaster

Masukan data seleksi juga dapat digunakan untuk memasukkan WHERE klausul pada akhir identifikasi string (misal, `geor:scott/tiger@orcl,gdal_import,raster,geoid=`). Lihat http://www.gdal.org/frmt_georaster.html untuk informasi lebih lanjut.

20.12.3 Menampilkan GeoRaster

Akhirnya, dengan memilih GeoRaster dari daftar Tabel Data Raster dan Id Raster, gambar raster akan dimuat ke QGIS.

Dialog *Pilih Spasial Oracle GeoRaster* dapat ditutup sekarang dan waktu berikutnya terbuka, akan menjaga sambungan yang sama dan akan menampilkan daftar sama subdatasets sebelumnya, sehingga sangat mudah untuk membuka gambar lain dari konteks yang sama.

Catatan: GeoRasters yang berisi piramida akan ditampilkan lebih cepat, tapi piramida harus dihasilkan di luar QGIS menggunakan Oracle PL/SQL atau gdaladdo.

Berikut ini adalah contoh menggunakan gdaladdo:

```
gdaladdo georaster:scott/tiger@orcl,georaster\_table,georaster,georid=6 -r
nearest 2 4 6 8 16 32
```

Ini contoh menggunakan PL/SQL:

```
$ sqlplus scott/tiger
SQL> DECLARE
  gr sdo_georaster;
BEGIN
  SELECT image INTO gr FROM cities WHERE id = 1 FOR UPDATE;
  sdo_geor.generatePyramid(gr, 'rLevel=5, resampling=NN');
  UPDATE cities SET image = gr WHERE id = 1;
  COMMIT;
END;
```

20.13 Plugin Raster Analisis Terrain



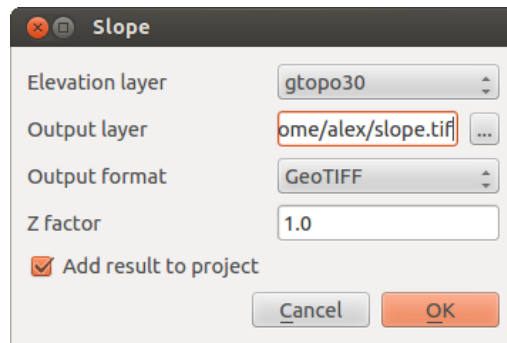
Plugin Raster Analisis Terrain dapat digunakan untuk menghitung kemiringan, aspek, bayangan bukit, indeks kekasaran dan relief untuk model elevasi digital (DEM). Sangat sederhana menangani dan menyediakan grafis antarmuka pengguna intuitif untuk membuat lapisan raster baru (Lihat [Figure_raster_terrain_1](#)).

Deskripsi dari analisis:

- **Lereng:** Menghitung sudut kemiringan untuk setiap sel dalam derajat (berdasarkan urutan pertama estimasi derivatif).
- **Aspek:** Eksposisi (dimulai dengan 0 untuk arah utara, dalam derajat berlawanan jarum jam).
- **Bukitbayangan:** Buat peta berbayang menggunakan cahaya dan bayangan untuk memberikan penampilan lebih tiga dimensi untuk peta relief berbayang.
- **Indeks Ketidakrataan:** Sebuah pengukuran kuantitatif dari medan heterogenitas seperti yang dijelaskan oleh Riley et al. (1999). Hal ini dihitung untuk setiap lokasi, dengan meringkas perubahan elevasi dalam kisi 3x3 piksel.
- **Relief:** Membuat peta relief berbayang dari data elevasi digital. Diimplementasikan metode memilih warna elevasi dari menganalisis frekuensi distribusi.

20.13.1 Menggunakan plugin

1. Mulai QGIS dan muat layer raster `gtopo30` dari contoh lokasi GRASS.
2. Muat plugin Raster Analisis Terrain plugin di Pengelola Plugin (lihat Bagian *The Plugins Dialog*).




Gambar 20.23: Plugin Raster Pemodelan Terrain (perhitungan kemiringan)

3. Pilih metode analisis dari menu (misalnya *Raster* → *Analisis Terrain* → *Kemiringan*). Dialog *Kemiringan* seperti dalam gambar [Figure_raster_terrain_1](#).
4. Spesifikasi keluaran berkas path dan keluaran tipe berkas:
5. Klik [OK].

20.14 Plugin Heatmap


Plugin *Heatmap* menggunakan Kernel Density Estimation untuk menciptakan kepadatan (heatmap) raster dari titik masukan lapisan vektor. Kepadatan tersebut dihitung berdasarkan jumlah poin di lokasi, dengan jumlah poin berkerumun yang lebih besar menghasilkan nilai yang lebih besar. Heatmaps memudahkan identifikasi “hotspot” dan pengelompokan poin.

20.14.1 Mengaktifkan plugin Heatmap


First this core plugin needs to be activated using the Plugin Manager (see *The Plugins Dialog*). After activation, the heatmap icon  can be found in the Raster Toolbar, and under the *Raster* → *Heatmap* menu.

Pilih menu *Tampilan* → *Toolbar* → *Raster* untuk memunculkan Toolbar Raster jika tidak terlihat.

20.14.2 Menggunakan plugin Heatmap

Klik alat tombol  *Heatmap* membuka dialog Plugin Heatmap (lihat [figure_heatmap_2](#)).

Dialog memiliki opsi-opsi berikut:

- **Masukan poin lapisan:** Daftar semua titik lapisan vektor dalam proyek saat ini dan digunakan untuk memilih lapisan yang akan dianalisis.
- **Raster keluaran:** Memungkinkan Anda menggunakan tombol  untuk memilih nama folder dan berkas keluaran raster plugin Heatmap yang dihasilkan. Sebuah ekstensi berkas tidak diperlukan.
- **Format keluaran:** Memilih format keluaran. Meskipun semua format didukung oleh GDAL dapat dipilih, dalam banyak kasus GeoTIFF adalah format terbaik.
- **Radius:** Digunakan untuk menentukan radius pencarian heatmap (atau lebar pita kernel) dalam meter atau unit peta. Jari-jari menentukan jarak sekitar titik di mana pengaruh titik akan terasa. Nilai yang lebih besar menghasilkan kehalusan yang lebih besar, tetapi nilai-nilai yang lebih kecil mungkin menunjukkan rincian halus dan variasi dalam kepadatan titik.

Saat kotak centang  *Lanjutan* dicentang, opsi tambahan akan tersedia:

- **Baris dan Kolom:** Digunakan untuk mengubah dimensi keluaran raster. Nilai-nilai ini juga terkait dengan **Ukuran sel X** dan **Ukuran sel Y**. Peningkatan jumlah baris atau kolom akan mengurangi ukuran sel dan meningkatkan ukuran berkas keluaran. Nilai-nilai dalam Baris dan Kolom juga terkait, sehingga menggandakan jumlah baris secara otomatis akan menggandakan jumlah kolom dan ukuran sel juga akan dibagi dua. Wilayah geografis raster keluaran akan tetap sama!
- **Ukuran sel X dan Ukuran sel Y:** Mengontrol ukuran geografis masing-masing piksel dalam keluaran raster. Mengubah nilai-nilai ini juga akan mengubah jumlah Baris dan Kolom dalam keluaran raster.
- **Bentuk Kernel:** Bentuk kernel mengontrol tingkat di mana pengaruh titik menurun sebagai jarak dari titik meningkat. Kernel yang berbeda peluruhan pada tingkat yang berbeda, sehingga kernel triweight memberikan fitur bobot yang lebih besar untuk jarak yang lebih dekat ke titik maka kernel Epanechnikov tidak. Akibatnya, triweight hasil dalam hotspot “tajam”, dan hasil Epanechnikov dalam hotspot “halus”. Sejumlah fungsi kernel standar yang tersedia di QGIS, dijelaskan dan diilustrasikan pada [Wikipedia](#).
- **Rasio Decay:** Dapat digunakan dengan kernel Segitiga untuk lebih mengontrol bagaimana panas/heat dari fitur menurun dengan jarak dari fitur tersebut.
 - Nilai 0 (= minimum) menunjukkan bahwa panas akan terkonsentrasi di pusat radius tertentu dan benar-benar padam di tepi.
 - Nilai 0,5 mengindikasikan bahwa piksel di tepi jari-jari akan diberikan setengah panas/heat sebagai piksel di pusat radius pencarian.
 - Nilai 1 berarti panas/heat tersebar secara merata di seluruh radius pencarian lingkaran. (Ini sama dengan ‘Uniform’ kernel.)
 - Sebuah nilai yang lebih besar dari 1 menunjukkan bahwa panas lebih tinggi menuju tepi radius pencarian daripada di pusat.

Masukan poin lapisan juga dapat memiliki bidang atribut yang dapat mempengaruhi bagaimana mereka mempengaruhi heatmap ini:




- **Gunakan radius dari bidang:** Mengatur radius pencarian untuk setiap fitur dari kolom atribut dalam masukan lapisan.
- **Gunakan bobot dari bidang:** Memungkinkan fitur masukan untuk dilakukan pembobotan dengan bidang atribut. Hal ini dapat digunakan untuk meningkatkan pengaruh fitur tertentu pada heatmap yang dihasilkan.

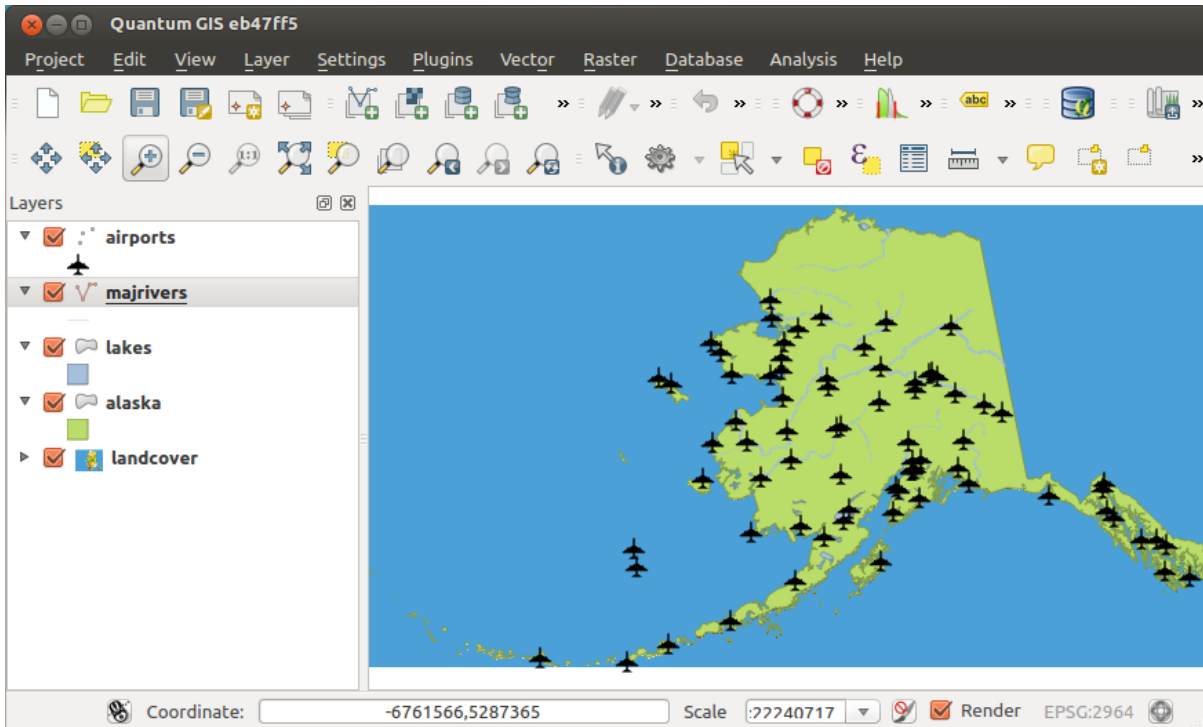
Ketika sebuah nama berkas keluaran raster yang ditentukan, tombol **[OK]** dapat digunakan untuk membuat heatmap tersebut.

20.14.3 Tutorial: Membuat Heatmap

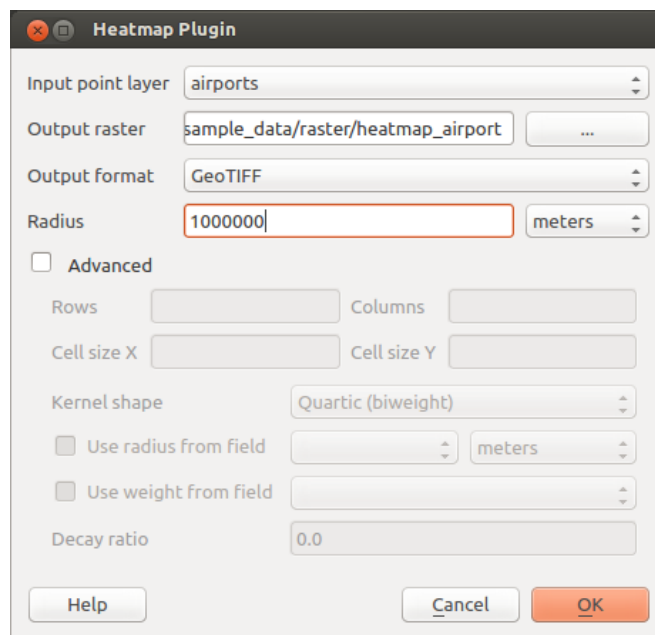
Sebagai contoh berikut, kita akan menggunakan lapisan vektor poin `airports` dari contoh dataset QGIS (lihat *Contoh data*). Tutorial QGIS sangat baik yang lain untuk membuat peta panas/heat dapat ditemukan di <http://qgis.spatialthoughts.com>.

Dalam `Figure_Heatmap_1`, bandara Alaska ditunjukkan.

1. Pilih tombol  *Heatmap* membuka dialog Heatmap (lihat `Figure_Heatmap_2`).
2. Dalam kolom *Masukan lapisan poin* , pilih `airports` dari daftar lapisan titik yang dimuat dalam proyek saat ini.
3. Tentukan nama berkas keluaran dengan mengklik tombol  selanjutnya ke *Raster keluaran*. Masukkan nama berkas `heatmap_airports` (tidak ada ekstensi berkas yang diperlukan).
4. Tinggalkan *Format keluaran* sebagai format default, `GeoTIFF`.
5. Ubah *Radius* ke 1000000 meter.
6. Klik **[OK]** membuat dan memuat bandara heatmap (lihat `Figure_Heatmap_3`).

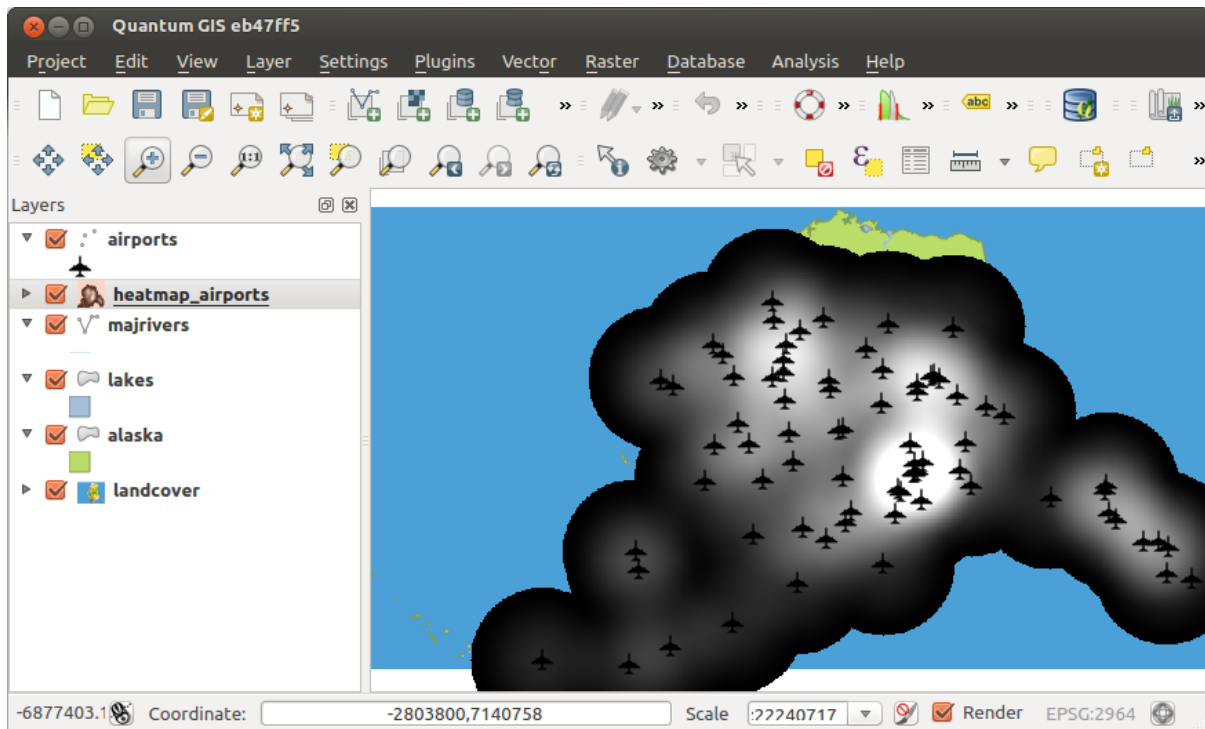


Gambar 20.24: Bandara Alaska 🐧





Gambar 20.25: Dialog Heatmap 🐧

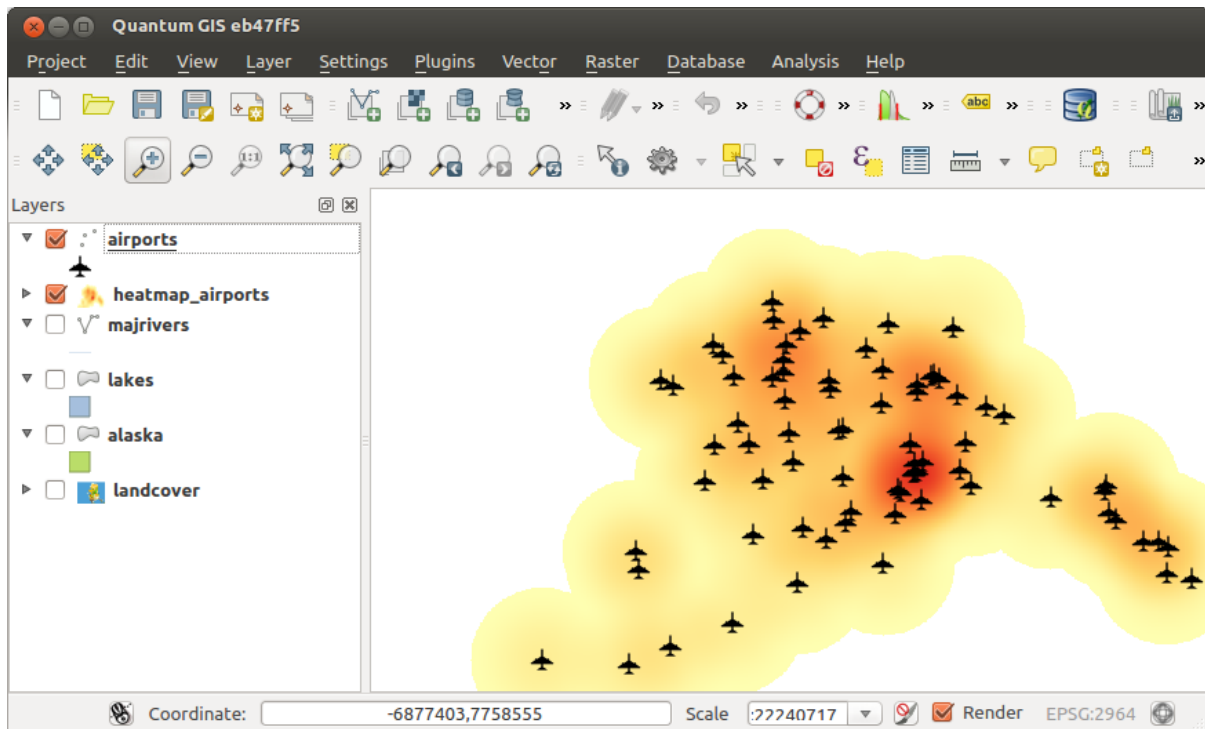
QGIS akan menghasilkan heatmap dan menambahkan hasilnya ke jendela peta Anda. Secara default, heatmap yang diarsir pada Skala abu-abu, dengan area yang lebih terang menunjukkan konsentrasi yang lebih tinggi dari bandara. Heatmap sekarang dapat ditata dalam QGIS untuk memperbaiki penampilan.



Gambar 20.26: Heatmap setelah pemuatan tampak seperti permukaan abu-abu 🐧

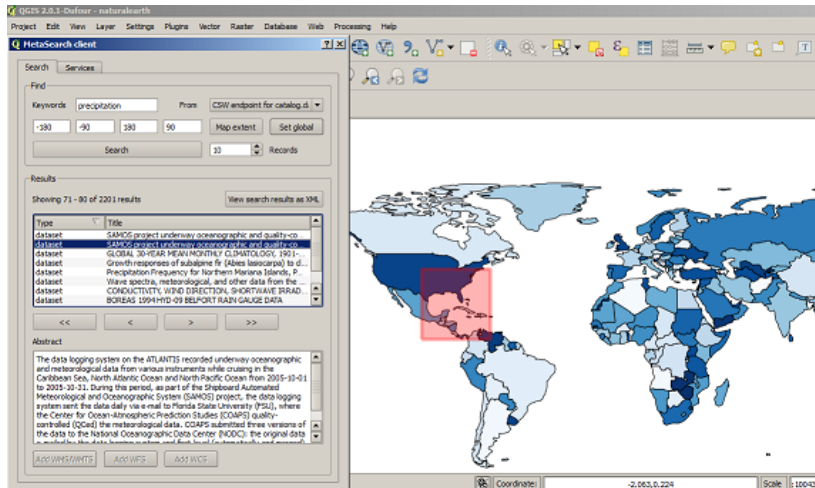
1. Buka dialog properti dialog dari lapisan `heatmap_airports` (pilih lapisan `heatmap_airports`, buka menu konteks dengan tombol kanan tetikus dan pilih *Properti*).
2. Pilih tab *Style*.
3. Ubah *Tipe Render*  ke 'Singleband pseudocolor'.
4. Pilih yang sesuai *Warna peta* , misalnya `YlOrRed`.
5. Klik tombol **[Muat]** untuk mengambil nilai minimum dan maksimum dari raster, kemudian klik tombol **[Klasifikasi]**.
6. Klik **[OK]** untuk memperbarui lapisan.

Hasil akhir akan ditampilkan dalam `Figure_Heatmap_4`.



Gambar 20.27: Gaya heatmap dari bandara Alaska 🐧

20.15 MetaSearch Catalogue Client



20.15.1 Introduction

MetaSearch is a QGIS plugin to interact with metadata catalogue services, supporting the OGC Catalogue Service for the Web (CSW) standard.

MetaSearch provides an easy and intuitive approach and user-friendly interface to searching metadata catalogues within QGIS.

20.15.2 Installation

MetaSearch is included by default with QGIS 2.0 and higher. All dependencies are included within MetaSearch.

Install MetaSearch from the QGIS plugin manager, or manually from <http://plugins.qgis.org/plugins/MetaSearch>.

20.15.3 Working with Metadata Catalogues in QGIS

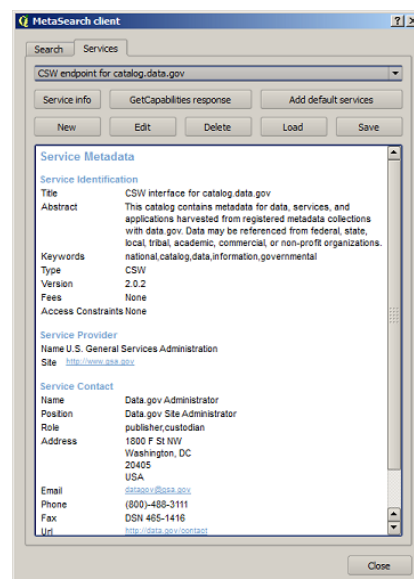
CSW (Catalogue Service for the Web)

CSW (Catalogue 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 MetaSearch icon or select Web / MetaSearch / MetaSearch via the QGIS main menu. The MetaSearch dialog will appear. The main GUI consists of two tabs: 'Services' and 'Search'.

Managing Catalogue Services



The 'Services' tab allows the user to manage all available catalogue services. MetaSearch provides a default list of Catalogue Services, which can be added by pressing 'Add default services' button.

To all listed Catalogue Service entries, click the dropdown select box.

To add a Catalogue Service entry, click the 'New' button, and 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). Clicking ok will add the service to the list of entries.

To edit an existing Catalogue Service entry, select the entry you would like to edit and click the 'Edit' button, and modify the Name or URL values, then click ok.

To delete a Catalogue 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.

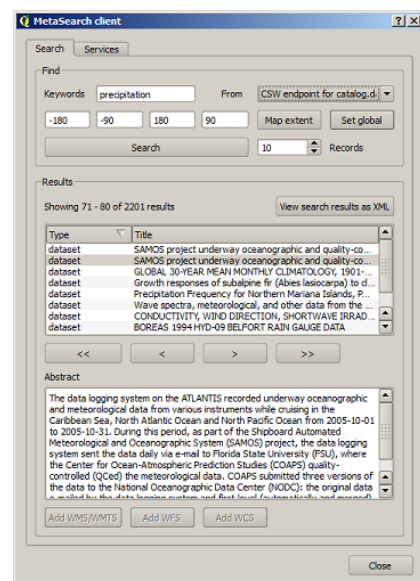
```
<?xml version="1.0" encoding="UTF-8"?>
<qgsCSWConnections version="1.0">
  <csw name="Data.gov CSW" url="http://catalog.data.gov/csw-all"/>
  <csw name="Geonorge - National CSW service for Norway" url="http://www.geonorge.no/geonetwork">
  <csw name="Geoportale Nazionale - Servizio di ricerca Italiano" url="http://www.pcn.minambiente.it">
  <csw name="LINZ Data Service" url="http://data.linz.govt.nz/feeds/csw"/>
```

```
<CSW name="Nationaal Georegister (Nederland)" url="http://www.nationaalgeoregister.nl/geonetw
<CSW name="RNDT - Repertorio Nazionale dei Dati Territoriali - Servizio di ricerca" url="http
<CSW name="UK Location Catalogue Publishing Service" url="http://csw.data.gov.uk/geonetwork/s
<CSW name="UNEP/GRID-Geneva Metadata Catalog" url="http://metadata.grid.unep.ch:8080/geonetwo
</qgsCSWConnections>
```

To load a list of entries, click the ‘Load’ button. A new window will appear; click the ‘Browse’ button and navigate to the XML file of entries you wish to load and click ‘Open’. The list of entries will be displayed. Select the entries you wish to add from the list and click ‘Load’.

The ‘Service info’ button displays information about the selected Catalogue 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 Catalogue Services



The ‘Search’ tab allows the user to query Catalogue Services for data and services, set various search parameters and view results.

The following search parameters are available:

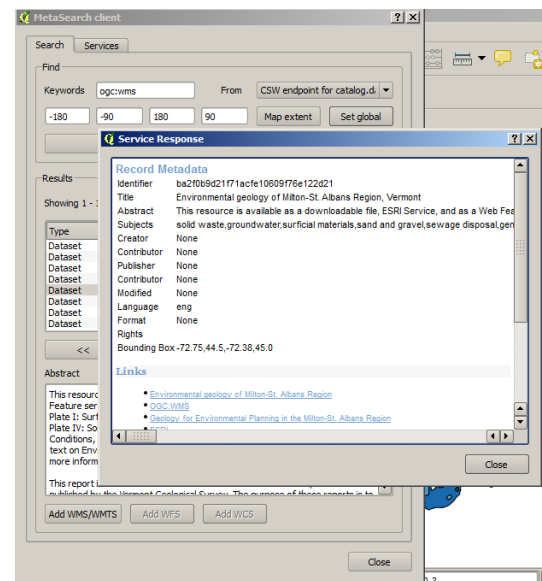
- **Keywords:** free text search keywords
- **From:** the Catalogue Service to perform the query against
- **Bounding box:** the spatial area of interest to filter on. The default bounding box is the map view / canvas. Click ‘Set global’ to do a global search, or enter custom values as desired
- **Records:** the number of records to return when searching. Default is 10 records

Clicking the ‘Search’ button will search the selected Metadata Catalogue. Search results are displayed in a list and are sortable by clicking on the column title. You can navigate through search results with the directional buttons below the search results. Clicking the ‘View search results as XML’ button opens a window with the service response in raw XML format.

Clicking a result will show the record’s abstract in the ‘Abstract’ window and provides the following options:

- 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 an OGC web service (WMS/WMTS, WFS, WCS), the appropriate ‘Add to WMS/WMTS|WFS|WCS’ buttons will be enabled for the user to add to QGIS. When clicking this button, MetaSearch will verify if this is a valid OWS. The OWS will then be added to the appropriate QGIS connection list, and the appropriate WMS/WMTS|WFS|WCS connection dialogue will then appear



Settings

You can fine tune MetaSearch with the following settings:

- **Results paging:** when searching metadata catalogues, the number of results to show per page
- **Timeout:** when searching metadata catalogues, the number of seconds for blocking connection attempt. Default value is 10

20.16 Plugin Grafik Jalan

Plugin Grafik Jalan merupakan plugin C++ untuk QGIS, yang menghitung jalur terpendek antara dua titik pada setiap lapisan polyline dan plot jalan ini melalui jaringan jalan.

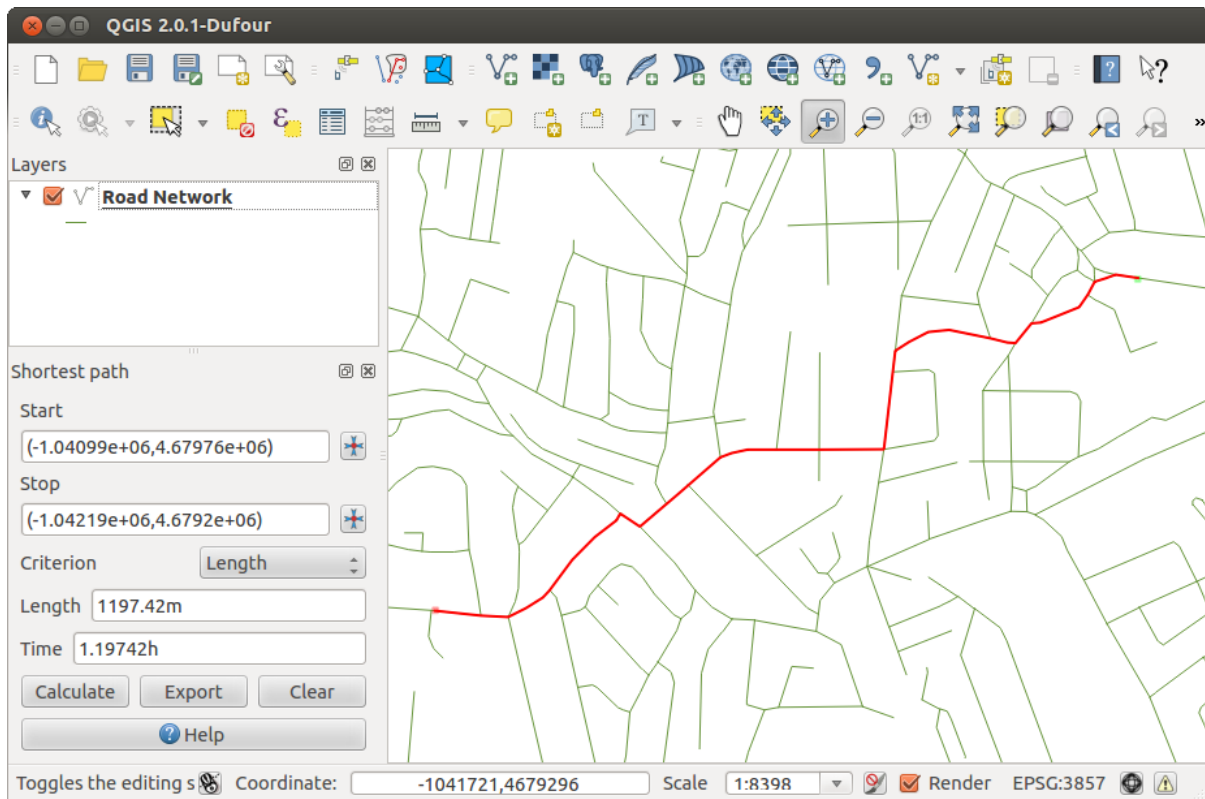
Fitur Utama:

- Menghitung path, maupun panjang dan waktu travel.
- Mengoptimalkan dari panjang atau dari waktu travel.
- Ekspor path ke lapisan vektor.
- Arah jalan highlight (ini lambat dan digunakan terutama untuk tujuan debug dan untuk pengujian pengaturan).

Seperti lapisan jalan Anda bisa menggunakan setiap lapisan vektor polyline dalam format yang didukung QGIS. Dua garis dengan titik umum dianggap terhubung. Harap dicatat, ini diperlukan untuk menggunakan CRS lapisan sebagai CRS proyek ketika mengedit lapisan jalan. Hal ini disebabkan fakta bahwa perhitungan koordinat antara CRS yang berbeda memperkenalkan beberapa kesalahan yang dapat mengakibatkan diskontinuitas, bahkan ketika ‘snapping’ digunakan.

Dalam tabel lapisan atribut kolom berikut dapat digunakan:

- Kecepatan pada bagian jalan (kolom numerik).



Gambar 20.28: Plugin Grafik Jalan 🐧

- Arah (jenis, yang dapat dicor ke string). Arah maju dan mundur yang sesuai dengan jalan satu arah, dua arah mengindikasikan dua arah jalan.

Jika beberapa bidang tidak memiliki nilai apapun atau tidak ada — nilai standar yang digunakan. Anda dapat mengubah default dan beberapa pengaturan plugin di dialog pengaturan plugin.


20.16.1 Menggunakan plugin

Setelah Plugin aktif Anda akan melihat sebuah panel tambahan di sisi kiri dari jendela utama QGIS. Sekarang membuat beberapa definisi dialog *Pengaturan plugin grafik jalan* di menu *Vektor* → *Grafik Jalan* (lihat [figure_road_graph_2](#)).

Setelah mengatur *Satuan waktu*, *Satuan Jarak* dan *Toleransi topologi* Anda bisa memilih lapisan vektor di tab *Lapisan transportasi*. Disini Anda juga bisa memilih *Kolom Arah* dan *Kolom Kecepatan*. Dalam tab *Pengaturan standar* Anda bisa mengatur *Arah* untuk menghitung.

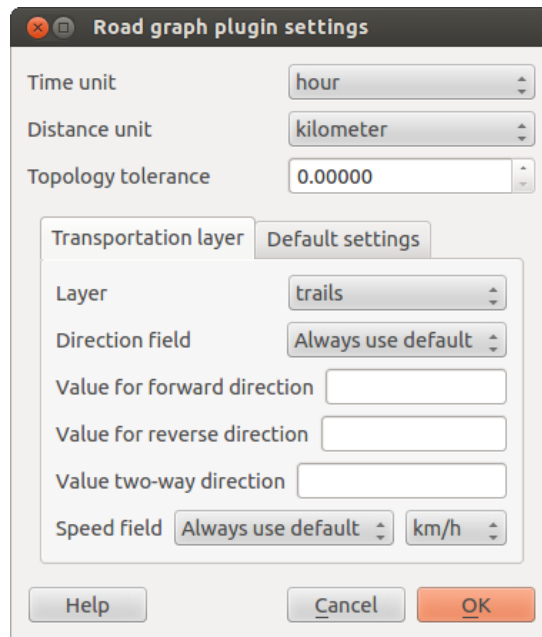
Akhirnya pada panel *Shortest Path* pilih mulai dan titik Berhenti di lapisan jaringan jalan dan klik **[Hitung]**.

20.17 Plugin Spasial Query

Plugin  Spasial Query memungkinkan untuk membuat query spasial (pilih fitur) di lapisan sasaran dengan mengacu pada lapisan lain. Fungsi ini didasarkan pada perpustakaan GEOS dan tergantung pada sumber fitur lapisan yang dipilih.

Operator yang mungkin adalah:

- Isi




Gambar 20.29: Pengaturan plugin grafik jalan 🐧






- Sama
- Tumpang tindih
- Persilangan
- Berpotongan
- Memisah
- Bersinggungan
- Di dalam

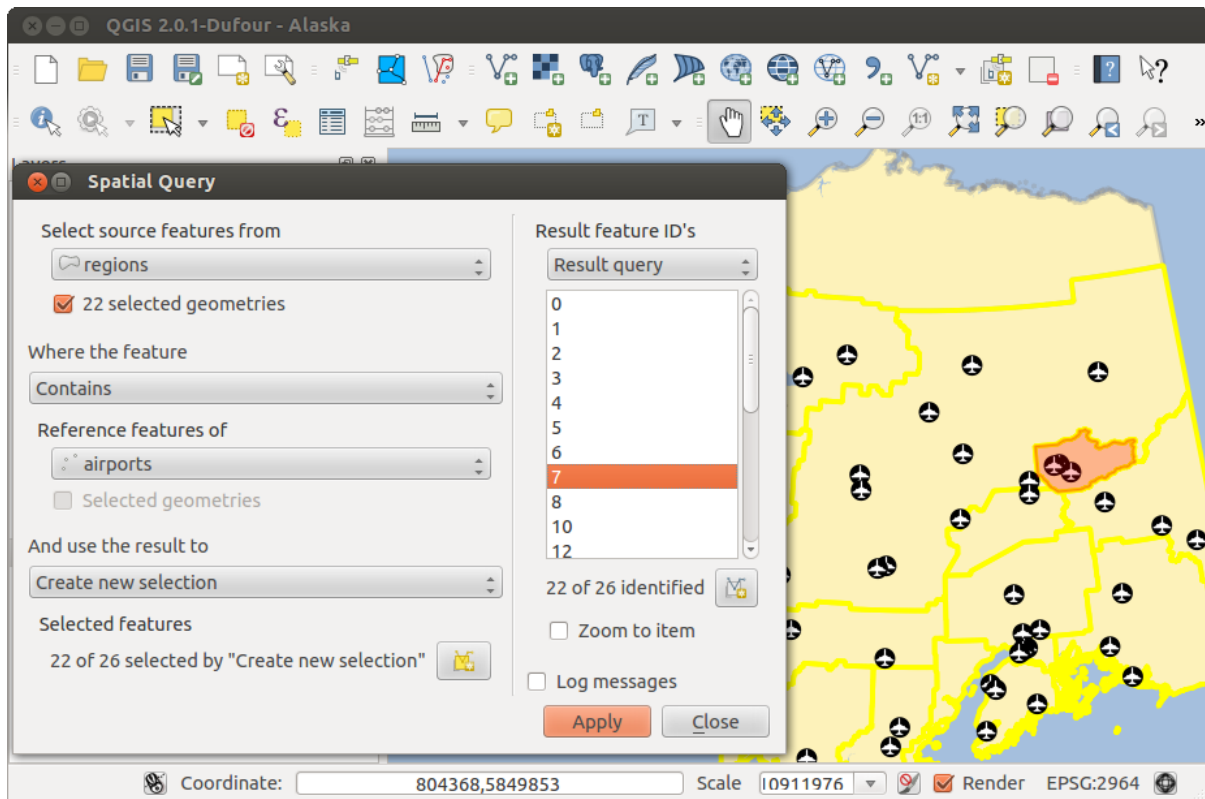
20.17.1 Menggunakan plugin

Sebagai contoh, kita ingin mencari daerah dalam dataset Alaska yang mengandung bandara. Langkah-langkah berikut diperlukan:

1. Mulai QGIS dan muat lapisan vektor `regions.shp` dan `airports.shp`.
2. Muat Plugin Spasial Query dari Pengelola Plugin (Lihat Bagian *The Plugins Dialog*) dan klik ikon  yang muncul di menu toolbar QGIS. Dialog plugin muncul.
3. Pilih lapisan `regions` sebagai sumber lapisan dan `airports` sebagai lapisan fitur referensi.
4. Pilih 'Isi' sebagai operator dan klik **[Terapkan]**.

Sekarang Anda mendapatkan daftar ID fitur dari query dan Anda memiliki beberapa pilihan seperti yang ditunjukkan pada `figure_spatial_query_1`.

- Klik pada  Buat lapisan dengan daftar item.
- Pilih ID dari daftar dan klik  Buat lapisan yang dipilih
- Pilih 'Hapus dari seleksi sekarang' dalam kolom *Dan gunakan hasilnya* .
- Selain itu ANda juga bisa  *Perbesar ke item* atau tampilkan  *Pesan Log*.



Gambar 20.30: Analisis Spasial Query - regions contain airports 🐧

20.18 Plugin SPIT

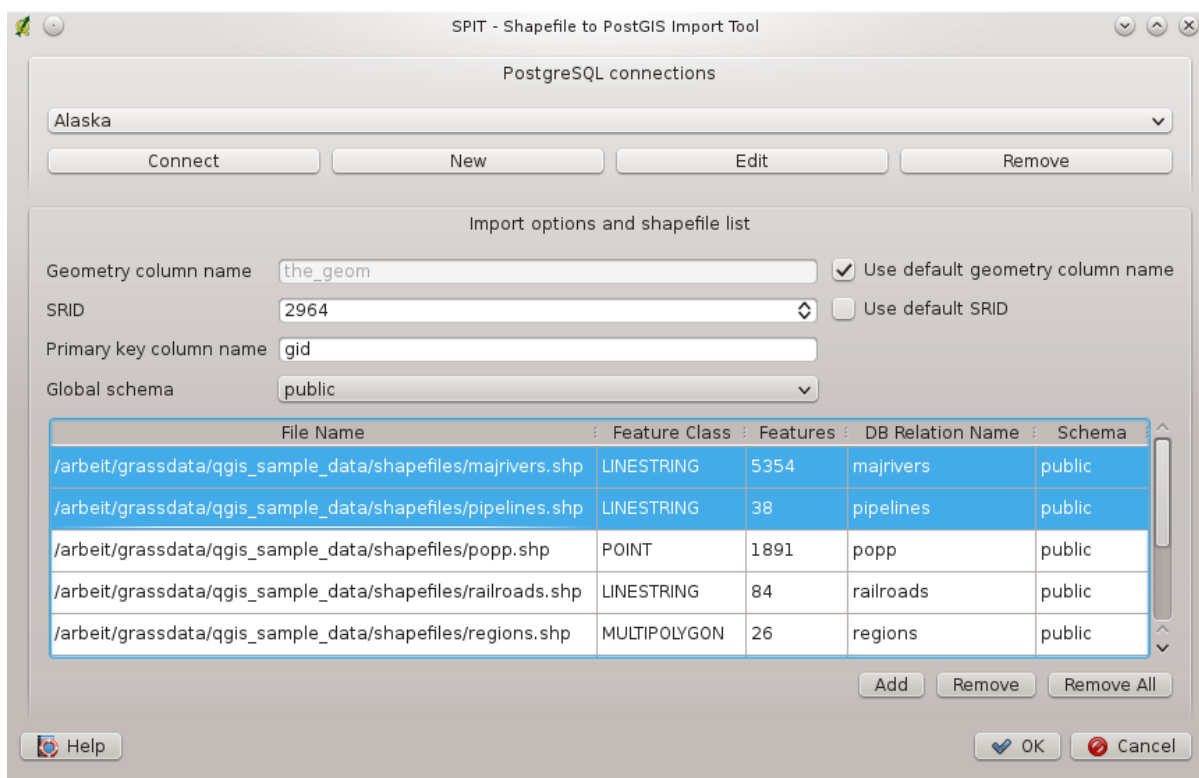
QGIS datang dengan nama plugin SPIT (Shapefile to PostGIS Import Tool). SPIT dapat digunakan untuk memuat beberapa shapefile pada satu waktu dan termasuk dukungan untuk skema. Menggunakan SPIT, buka Manajer Plugin dari menu *Plugin*, di menu *Installed* centang kotaknya *SPIT* dan klik **[OK]**.

Mengimpor shapefile, gunakan *Basisdata* → *Spit* → *Impor Shapefiles ke PostgreSQL* dari menu bar membuka dialog *SPIT - Alat Impor Shapefile ke PostGIS*. Pilih basisdata PostGIS yang ingin Anda koneksi dan klik **[Koneksi]**. Jika Anda ingin, Anda dapat menentukan atau mengubah beberapa opsi impor. Sekarang Anda dapat menambahkan satu atau lebih berkas ke antrian dengan mengklik tombol **[Tambah]**. Untuk memroses berkas, klik tombol **[OK]**. Proses impor serta kesalahan / peringatan akan ditampilkan setiap shapefile yang diproses.

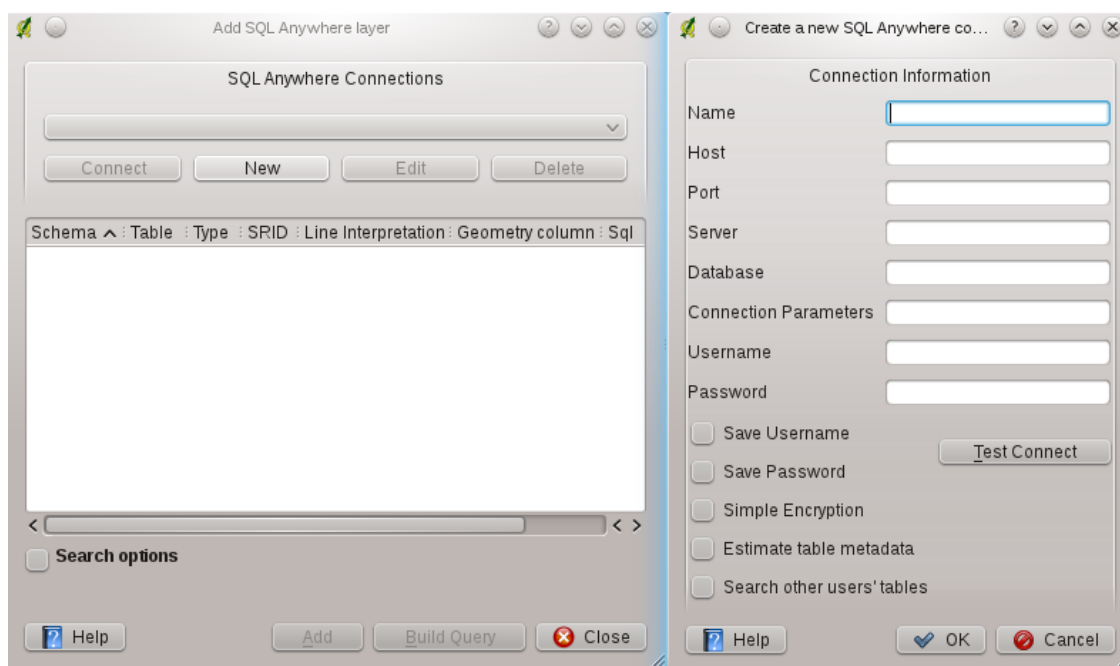
20.19 Plugin SQL Anywhere

SQL Anywhere adalah relational database management system (RDBMS) proprietary dari Sybase. SQL Anywhere termasuk dukungan spasial antara lain OGC, shapefile dan dibangun dalam fungsi untuk ekspor ke KML, GML dan format SVG.

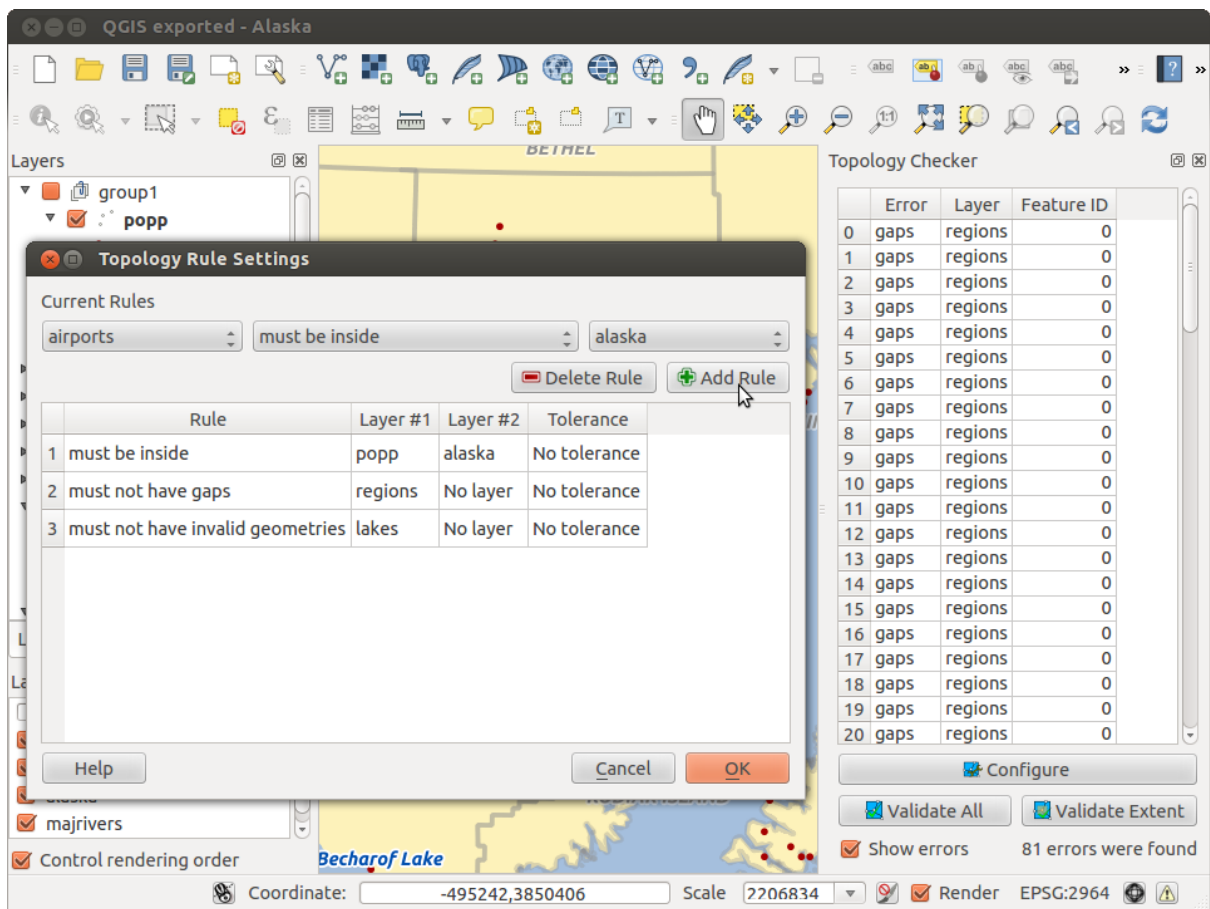
SQL Anywhere memungkinkan untuk terhubung ke spasial basisdata SQL Anywhere aktif. Dialog *Add SQL Anywhere layer* mirip dalam dialog dengan fungsi untuk PostGIS dan SpatiaLite.



Gambar 20.31: Menggunakan Plugin SPIT untuk mengimpor berkas Shape ke PostGIS 🐧



Gambar 20.32: Dialog SQL Anywhere (KDE) 🐧



Gambar 20.33: Plugin Pemeriksa Topologi

20.20 Plugin Pemeriksa Topologi

Topologi menggambarkan hubungan antara titik, garis dan poligon yang mewakili fitur dari wilayah geografis. Dengan plugin Pemeriksa Topologi Anda dapat melihat lebih berkas vektor dan memeriksa topologi dengan beberapa aturan topologi. Aturan-aturan ini memeriksa hubungan spasial apakah fitur Anda 'Equal', 'Contain', 'Cover', 'CoveredBy', 'Cross', 'Disjoint', 'Intersect', 'Overlap', 'Touches' dan 'Within' satu sama lain. Hal ini tergantung pada pertanyaan individu aturan topologi yang Anda terapkan pada data vektor. Misalnya biasanya Anda tidak akan menerima overshoot dalam lapisan baris tetapi jika mereka menggambarkan jalan buntu Anda tidak akan menghapusnya dari lapisan vektor Anda.

QGIS telah terintegrasi dengan fitur editing topologi yang besar untuk menciptakan fitur baru tanpa kesalahan. Tapi kesalahan data yang ada dan diinduksi kesalahan pengguna sulit untuk mencari tahu. Plugin ini membantu Anda mengetahui kesalahan tersebut melalui daftar aturan.

Hal ini sangat sederhana untuk membuat aturan topologi dengan plugin pemeriksa topologi

Pada **lapisan poin** aturan berikut ini tersedia:

- **Harus ditutupi oleh:** Disini Anda bisa memilih lapisan vektor dari proyek Anda. Poin yang tidak tercakup oleh lapisan vektor yang diberikan, terjadi 'Galat' di bidang.
- **Harus ditutupi oleh titik akhir dari:** Disini Anda bisa memilih lapisan garis dari proyek Anda.
- **Harus berada di dalam:** Disini Anda bisa memilih lapisan poligon dari proyek Anda. Poin harus berada di dalam poligon. Jika tidak, QGIS menulis sebuah 'Galat' untuk titik.
- **Harus tidak duplikat:** Setiap poin diwakili dua kali atau lebih, akan terjadi 'Galat' di bidang.
- **Harus tidak memiliki geometri yang tidak valid:** Periksa apakah geometri berlaku.
- **Harus tidak banyak-bagian-geometri:** Semua bagian poin yang ditulis kedalam bidang 'Galat'.

Pada **lapisan garis** aturan berikut ini tersedia:


- **Titik akhir harus ditutupi oleh:** Disini Anda bisa memilih lapisan poin dari proyek Anda.
- **Harus tidak teruntai:** Ini akan menunjukkan dalam lapisan garis.
- **Harus tidak duplikat:** Setiap kali fitur garis diwakili dua kali atau lebih, ini akan terjadi 'Galat' di bidang.
- **Harus tidak memiliki geometri yang tidak valid:** Periksa apakah geometri berlaku.
- **Harus tidak multi-bagian geometri:** Kadang-kadang, geometri sebenarnya merupakan kumpulan sederhana (singgel-bagian) geometri. Geometri seperti ini disebut multi-bagian geometri. Jika berisi hanya satu jenis geometri sederhana, kita menyebutnya multi-point, multi-linestring atau multi-poligon. Semua multi-bagian garis ditulis kedalam bidang 'Galat'.
- **Harus tidak memiliki pseudos:** Poin akhir sebuah garis geometri yang harus dihubungkan ke titik akhir dari dua geometri lainnya. Jika poin akhir terhubung ke satu poin akhir geometri lain, poin akhir disebut titik pseudo.

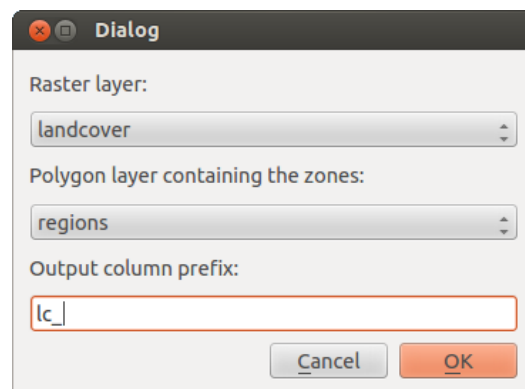
Pada **lapisan poligon** aturan berikut ini tersedia:

- **Harus berisi:** Lapisan poligon harus mengandung setidaknya satu poin/titik geometri dari lapisan kedua.
- **Harus tidak duplikat:** Poligon dari lapisan yang sama tidak harus memiliki geometri identik. Setiap kali fitur poligon diwakili dua kali atau lebih akan terjadi 'Galat' di bidang.
- **Harus tidak memiliki gap:** Poligon yang berdekatan seharusnya tidak membentuk gap/celah di antara mereka. Batas administratif bisa disebutkan sebagai contoh (poligon negara bagian AS tidak memiliki gap/celah antara mereka ...).
- **Harus tidak memiliki geometri yang tidak valid:** Periksa apakah geometri valid. Beberapa aturan yang mendefinisikan geometri yang valid:
 - Ring poligon harus tertutup.
 - Ring yang mendefinisikan lubang harus berada di dalam ring yang menentukan batas-batas eksterior.

- Cincin tidak mungkin berpotongan (mereka mungkin tidak menyentuh atau silang satu sama lain).
- Cincin mungkin tidak menyentuh cincin lain, kecuali pada suatu titik.
- **Harus tidak multi-bagian geometri:** Kadang-kadang, geometri sebenarnya merupakan kumpulan sederhana (singgel-bagian) geometri. Geometri seperti ini disebut multi-bagian geometri. Jika berisi hanya satu jenis geometri sederhana, kita menyebutnya multi-point, multi-linestring atau multi-poligon. Sebagai contoh, sebuah negara yang terdiri dari beberapa pulau dapat direpresentasikan sebagai multi-poligon.
- **Tidak tumpang tindih:** Poligon yang berdekatan harus tidak berbagi area umum.
- **Tidak boleh tumpang tindih dengan:** Poligon yang berdekatan dari satu lapisan tidak harus berbagi area umum dengan poligon dari lapisan lain.

20.21 Plugin Statistik Zonal

Dengan  Plugin Statistik Zonal Anda dapat menganalisis hasil klasifikasi tematik. Hal ini memungkinkan untuk menghitung beberapa nilai-nilai piksel dari lapisan raster dengan bantuan lapisan vektor poligonal (lihat [figure_zonal_statistics](#)). Anda dapat menghitung jumlah, nilai rata-rata dan jumlah total piksel yang berada di dalam poligon. Plugin menghasilkan kolom keluaran dalam lapisan vektor dengan awalan user-defined.



Gambar 20.34: Dialog Statistik Zonal (KDE) 

Bantuan dan Dukungan

21.1 Milis

QGIS masih aktif dikembangkan dan karena itu tidak akan selalu bekerja seperti yang Anda harapkan. Cara yang lebih disukai untuk mendapatkan bantuan adalah dengan bergabung di milis qgis-pengguna. Pertanyaan Anda akan menjangkau khalayak yang lebih luas dan jawabannya juga akan menguntungkan orang lain.

21.1.1 qgis-pengguna

Milis ini digunakan untuk membahas QGIS secara umum, serta pertanyaan-pertanyaan spesifik mengenai instalasi dan penggunaan. Anda dapat berlangganan milis qgis-pengguna dengan mengunjungi tautan URL: <http://lists.osgeo.org/mailman/listinfo/qgis-user>

21.1.2 fossGIS-talk-liste

Untuk audien berbahasa Jerman, FOSSGIS e.V. menyediakan milis fossGIS-talk-liste. Milis ini digunakan untuk diskusi GIS open source secara umum termasuk QGIS. Anda dapat berlangganan dengan mengunjungi URL: <https://lists.fossGIS.de/mailman/listinfo/fossGIS-talk-liste>

21.1.3 qgis-pengembang

Jika Anda seorang pengembang menghadapi masalah yang bersifat teknis, Anda mungkin ingin bergabung dengan milis qgis-pengembang di sini: <http://lists.osgeo.org/mailman/listinfo/qgis-developer>

21.1.4 qgis-commit

Setiap kali komit dibuat untuk kode repositori QGIS surel dikirim ke daftar ini. Jika Anda ingin mendapatkan informasi terbaru setiap perubahan basis kode saat ini, Anda bisa berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-commit>

21.1.5 qgis-trac

Daftar ini memberikan notifikasi surel yang terkait dengan manajemen proyek, termasuk laporan bug, tugas, dan permintaan fitur. Anda dapat berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-trac>

21.1.6 qgis-komunitas-tim

Daftar ini berkaitan dengan topik seperti dokumentasi, konteks bantuan, panduan-pengguna, pengalaman online termasuk situs web, blog, milis, forum, dan upaya penerjemahan. Jika Anda ingin bekerja pada panduan-pengguna juga, dalam daftar ini adalah titik awal yang baik untuk mengajukan pertanyaan Anda. Anda dapat berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-community-team>

21.1.7 qgis-rilis-tim

Daftar ini berkaitan dengan topik seperti proses rilis, pemaketan binari untuk berbagai OS dan mengumumkan rilis baru ke dunia pada umumnya. Anda dapat berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-release-team>

21.1.8 qgis-tr

Daftar ini berkaitan dengan upaya penerjemahan. Jika Anda ingin bekerja pada penerjemahan panduan atau grafis antarmuka pengguna (GUI), dalam daftar ini adalah titik awal yang baik untuk mengajukan pertanyaan Anda. Anda bisa berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-tr>

21.1.9 qgis-edu

Daftar ini berkaitan dengan upaya pendidikan QGIS. Jika Anda ingin bekerja pada bahan pendidikan QGIS, daftar ini adalah awal yang baik untuk mengajukan pertanyaan Anda. Anda bisa berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-edu>

21.1.10 qgis-psc

Daftar ini digunakan untuk membahas isu-isu Komite Pengarah yang berkaitan dengan arah dan manajemen secara keseluruhan QGIS. Anda dapat berlangganan di: <http://lists.osgeo.org/mailman/listinfo/qgis-psc>

Anda dipersilakan untuk berlangganan ke salah satu daftar. Harap ingat untuk berkontribusi ke dalam daftar dengan menjawab pertanyaan dan berbagi pengalaman Anda. Perhatikan bahwa qgis-commit dan qgis-trac dirancang hanya untuk pemberitahuan saja dan tidak dimaksudkan untuk mengirim surel.

21.2 IRC

Kami juga menggunakan IRC - kunjungi kami dengan bergabung dalam kanal #qgis di irc.freenode.net. Butuh waktu untuk menanggapi pertanyaan Anda karena banyak orang-orang di saluran ini melakukan hal-hal lain dan mungkin diperlukan waktu beberapa saat bagi mereka untuk melihat pertanyaan Anda. Jika Anda melewatkan sebuah diskusi di IRC, tidak masalah! Kami log semua diskusi, sehingga Anda dapat dengan mudah mengejar ketinggalan. Hanya pergi ke <http://qgis.org/irclogs> dan baca IRC-logs.

Dukungan komersial untuk QGIS juga tersedia. Periksa website <http://qgis.org/en/commercial-support.html> untuk informasi selengkapnya.

21.3 BugTracker

Sementara milis qgis-pengguna berguna untuk jenis pertanyaan umum 'bagaimana cara melakukan XYZ di QGIS', Anda mungkin ingin memberitahu kami tentang bug di QGIS. Anda bisa mengirimkan laporan bug menggunakan pelacak bug QGIS di <http://hub.qgis.org/projects/quantum-gis/issues>. Saat membuat tiket bug baru, mohon diinformasikan alamat surel dimana kami bisa meminta informasi tambahan.

Harap diingat bahwa bug Anda mungkin tidak selalu sesuai dengan prioritas Anda seperti harapan Anda (tergantung pada tingkat kesulitannya). Beberapa bug mungkin memerlukan upaya pengembang yang signifikan untuk memperbaiki dan tenaga kerja tidak selalu tersedia.

Permintaan fitur dapat disampaikan juga menggunakan sistem tiket yang sama untuk bug. Pastikan untuk memilih jenis *Feature*.

Jika Anda telah menemukan bug dan memperbaikinya sendiri Anda bisa mengirimkan patchnya juga. Sekali lagi, kirim ke sistem tiket redmine di <http://hub.qgis.org/wiki/quantum-gis/issues>. Periksa kotak centang *Patch supplied* dan melampirkan patch Anda sebelum mengirimkan bug. Seseorang pengembang akan meninjau dan menerapkannya ke QGIS. Harap jangan khawatir jika patch Anda tidak diterapkan langsung — pengembang terikat dengan komitmen-komitmen lainnya.

21.4 Blog

Komunitas juga menjalankan weblog di <http://planet.qgis.org/planet/> yang memiliki beberapa artikel menarik bagi pengguna dan pengembang juga disediakan oleh masyarakat dalam blog lain. Anda diundang berkontribusi pada blog QGIS Anda sendiri!

21.5 Plugin

Situs web <http://plugins.qgis.org> menyediakan web resmi Portal plugin QGIS. Di sini Anda menemukan daftar semua plugin QGIS stabil dan eksperimental tersedia melalui 'Repositori Plugin QGIS Resmi'.

21.6 Wiki

Akhirnya, kami memelihara situs web WIKI di <http://hub.qgis.org/projects/quantum-gis/wiki> di mana Anda dapat menemukan berbagai informasi berguna yang berhubungan dengan pengembangan QGIS, rencana rilis, tautan untuk mengunduh, pesan petunjuk-terjemahan dan sebagainya. Silakan dikunjungi, ada banyak informasi bagus didalamnya!

.

Appendix

22.1 GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow. **TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION**

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to

any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
 - (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
 - (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public 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.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

QGIS Qt exception for GPL

In addition, as a special exception, the QGIS Development Team gives permission to link the code of this program with the Qt library, including but not limited to the following versions (both free and commercial): Qt/Non-commercial Windows, Qt/Windows, Qt/X11, Qt/Mac, and Qt/Embedded (or with modified versions of Qt that use the same license as Qt), and distribute linked combinations including the two. You must obey the GNU General Public License in all respects for all of the code used other than Qt. If you modify this file, you may extend this exception to your version of the file, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

22.2 GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc

<<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

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. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **Document**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “**you**”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “**Modified Version**” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “**Secondary Section**” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “**Invariant Sections**” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “**Cover Texts**” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “**Transparent**” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called **Opaque**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “**Title Page**” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “**publisher**” means any person or entity that distributes copies of the Document to the public.

A section “**Entitled XYZ**” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “**Acknowledgements**”, “**Dedications**”, “**Endorsements**”, or “**History**”.)

To “**Preserve the Title**” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

1. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
2. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
3. State on the Title page the name of the publisher of the Modified Version, as the publisher.
4. Preserve all the copyright notices of the Document.
5. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
6. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

7. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
8. Include an unaltered copy of this License.
9. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
10. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
11. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
12. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
13. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
14. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
15. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

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/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a

proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

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”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with ... Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Literatur dan Referensi Web

GDAL-SOFTWARE-SUITE. Geospatial data abstraction library. <http://www.gdal.org>, 2013.

GRASS-PROJECT. Geographic resource analysis support system. <http://grass.osgeo.org> , 2013.

NETELER, M., AND MITASOVA, H. Open source gis: A grass gis approach, 2008.

OGR-SOFTWARE-SUITE. Geospatial data abstraction library. <http://www.gdal.org/ogr> , 2013.

OPEN-GEOSPATIAL-CONSORTIUM. Web map service (1.1.1) implementation specification. <http://portal.opengeospatial.org>, 2002.

OPEN-GEOSPATIAL-CONSORTIUM. Web map service (1.3.0) implementation specification. <http://portal.opengeospatial.org>, 2004.

POSTGIS-PROJECT. Spatial support for postgresql. <http://postgis.refrations.net/> , 2013.

-
-
- %%, 100
 - Actions, 100
 - Alat Analisis, 680
 - Alat-Alat Georeferensi, 686
 - anotasi, 38
 - apache, 154
 - apache2, 154
 - Arc/Info_ASCII_Grid, 133
 - Arc/Info_Binary_Grid, 133
 - ArcInfo_Binary_Coverage, 66
 - Atlas_Generation, 658
 - attribute table, 124
 - Attribute_Actions, 100
 - Attribute_Table, 648
 - Attribute_Table_Selection, 124
 - Avoid_Intersections_Of_Polygons, 114

 - baris perintah, 17
 - bookmark, 39
 - bookmark spasial
 - lihat bookmark, 39
 - Browse_Maps, 61

 - Calculator_Field, 130
 - CAT, 145
 - Categorized_Renderer, 80
 - CGI, 154
 - Colliding_labels, 88
 - Color_interpolation, 138
 - color_Ramp, 77
 - colorBrewer, 77
 - Colormap, 138
 - Comma Separated Values, 66
 - Common_Gateway_Interface, 154
 - Compose_Maps, 627
 - Composer_Manager, 660
 - Composer_Template, 628
 - Contrast_enhancement, 136
 - Coordinate_Reference_System, 55, 149
 - Create_Maps, 627
 - Create_New_Layers, 121
 - crossing the 180 degrees longitude line, 71
 - CRS, 55, 149
 - CSV, 66, 116

 - Current_Edits, 115
 - Custom_color_Ramp, 77
 - Custom_CRS, 58

 - Datum_transformation, 59
 - DB_Manager, 73
 - Debian_Squeeze, 154
 - default_CRS, 55
 - define an action, 100
 - Derived_Fields, 130
 - Digitizing, 114
 - Discrete, 138
 - Displacement_plugin, 83
 - dokumentasi, 7

 - editing, 112
 - Elements_Alignment, 655
 - EPSG, 55
 - Equal_Interval, 81
 - Erdas Imagine, 133
 - ESRI, 63
 - European_Petroleum_Search_Group, 55
 - example actions, 100
 - Export_as_image, 660
 - Export_as_PDF, 660
 - Export_as_SVG, 660
 - Expressions, 105

 - FastCGI, 154
 - Field_Calculator, 130
 - Field_Calculator_Functions, 106
 - Fitur Identifikasi, 35

 - Gambaran Peta, 43
 - GDAL, 133
 - GeoTIFF, 133
 - GeoTiff, 133
 - GiST (Generalized Search Tree) index, 71
 - GML, 145
 - GNU General Public License, 715
 - Gradient_color_Ramp, 77
 - Graduated_Renderer, 81
 - GRASS, 171, *lihat* Creating new vectors;editing;creating a new layer
 - attribute linkage, 176
 - attribute storage, 175
-

- category settings, 177
- digitizing tools, 176
- display results, 180, 183
- region, 179
- region display, 179
- region editing, 179
- snapping tolerance, 178
- symbolology settings, 178
- table editing, 178
- toolbox, 183
- GRASS toolbox, 179
 - Browser, 186
 - customize, 187
- GRASS vector data model, 175
- Grid
 - Grids
 - Map_Grid, 635
- Histogram, 140
- HTML_Frame, 653
- IGNF, 55
- Import_Maps, 61
- Institut_Geographique_National_de_France, 55
- InteProxy, 152
- Inverted_Polygon_Renderer, 83
- jendela utama, 21
- join, 102
- join layer, 102
- keluaran simpan sebagai gambar, 20
- Konteks Bantuan, 31
- kualitas rendering, 33
- Layout_Maps, 627
- legenda, 27
- license document, 715
- load a shapefile, 64
- loading_raster, 133
- Map_Legend, 640
- Map_Navigation, 113
- Map_Template, 628
- MapInfo, 66
- Memperbarui rendering sebelum menggambar, 33
- Menghentikan rendering, 32
- menghitung skala, 30
- mengukur, 33
 - area, 33
 - panjang garis, 33
 - sudut, 33
- menu, 22
- merge attributes of features, 120
- Merge_Attributes_of_Selected_Features, 120
- Merge_Selected_Features, 120
- Metadata, 140
- MSSQL Spatial, 73
- Multi_Band_Raster, 135
- multipolygon, 119
- Natural_Breaks_(Jenks), 81
- New_GPX_Layer, 121, 122
- New_Shapefile_Layer, 121
- New_SpatialLite_Layer, 121
- New_Spatialite_Layer, 121
- Node_Tool, 115
- Nodes, 116
- Non_Spatial_Attribute_Tables, 126
- OGC, 145
- OGR, 63
- OGR Simple Feature Library, 63
- ogr2ogr, 70
- Open_Geospatial_Consortium, 145
- OpenStreetMap, 68
- Oracle Spatial, 73
- OSM, 68
- Pan, 113
- penyusun cetak cepat, 20
- Peralatan Riset, 680
- pgsql2shp, 70
- Picture_database, 639
- plugin-plugin, 663
- Point_Displacement_Renderer, 83
- PostGIS, 68
- PostGIS spatial index, 71
- PostgreSQL, 68
- Pretty_Breaks, 81
- print_composer
 - tools, 627
- Printing
 - Export_Map, 660
- Proj.4, 58
- Proj4, 57
- Proj4_text, 57
- Projections, 55
- Proxy, 147
- proxy-server, 147
- proyek nesting, 41
- Pyramids, 140
- QGIS_mapserver, 152
- QGIS_Server, 154
- QSpatialite, 73
- Quantile, 81
- Query_Builder, 128
- Raster, 133
- Raster_Calculator, 142
- Relations, 126
- Renderer_Categorized, 80
- Renderer_Graduated, 81
- Renderer_Point_Displacement, 83
- Renderer_Single_Symbol, 80
- Rendering, 31
- Rendering tergantung skala, 32

Rendering_Mode, 632
 Rendering_Rule-based, 83
 Revert_Layout_Actions, 656
 ring polygons, 119
 Rotate_Point_symbols, 120
 Rotated_North_Arrow, 639
 Rule-based_Rendering, 83

 Scalebar
 Map_Scalebar, 644
 Search_Radius, 112
 Secured_OGC_Authentication, 152
 Select_using_Query, 130
 SFS, 145
 Shapefile, 63
 Shapefile_to_Postgis_Import_Tool, 704
 Shared_Polygon_Boundaries, 113
 Shortcut Papanketik, 31
 shp2pgsql, 70
 Single_Band_Raster, 135
 Single_Symbol_Renderer, 80
 Skala, 32
 SLD, 154
 SLD/SE, 154
 Snapping, 112
 Snapping_On_Intersections, 114
 Snapping_Tolerance, 112
 Spatialite, 72
 Spatialite_Manager, 73
 SPIT, 704
 Split_Features, 120
 SQLite, 72
 SRS, 149
 ST_Shift_Longitude, 71
 Symbology, 87, 135

 tata letk toolbar, 27
 tetikus perbesar, 30
 Three_Band_Color_Raster, 135
 Tiger_Format, 66
 Toggle Editing, 114
 tombol mengarahkan, 30
 toolbar, 27
 Topological_Editing, 113
 Transparency, 139

 UK_National_Transfer_Format, 66
 US_Census_Bureau, 66

 Vertex, 116
 Vertices, 116
 visibilitas lapisan, 27

 WCS, 145, 153
 Web Coverage Service, 153
 WFS, 145, 153
 WFS-T, 153
 WFS_Transactional, 153
 WKT, 55, 116

 WMS, 145
 WMS-C, 150
 WMS_1.3.0, 152
 WMS_client, 145
 WMS_identify, 150
 WMS_layer_transparency, 149
 WMS_metadata, 151
 WMS_properties, 151
 WMS_tiles, 150
 WMTS, 150
 WMTS_client, 145
 Work_with_Attribute_Table, 122

 Zoom_In Zoom_Out, 113