MMQGIS

Michael Minn (http://michaelminn.com)

9 August 2018

Describes use of MMQGIS, a set of Python vector map layer plugins for Quantum GIS

Introduction

MMQGIS is a set of Python plugins for manipulating vector map layers in Quantum GIS: CSV input/output/join, geocoding, geometry conversion, buffering, hub analysis, simplification, column modification, and simple animation. MMQGIS provides an alternative to the Processing toolbox, with verbose progress reporting, an intuitive user interface, direct shapefile/CSV-file access, and some additional capabilities missing from other plugin sets.

MMQGIS is included in the Quantum GIS Plugin Repository and should be readily available in the QGIS Python Plugin Installer (Plugins -> Fetch Python Plugins). A zip file is also available here for manual installation.

MMQGIS assumes that input and output shapefiles and CSV files are encoded in the **UTF-8** character set. MMQGIS uses the standard Python CSV file interface functions, which do not handle Unicode or other multi-byte encodings. While files that use the lower 7-bits of the 8-bit Windoze character sets (ISO-8859-x) will generally be fine, unpredictable results and errors may occur with non-ASCII characters in non-UTF-8 character sets. See this and this for two ways to save a UTF-8 CSV from Excel.

Animation Tools

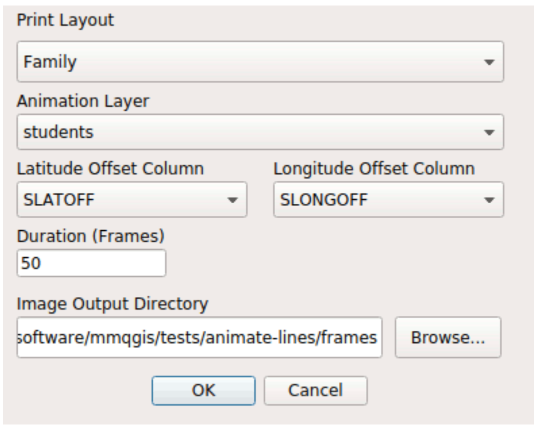
The animation tools permit creation of simple map animations as sequences of map image PNG files. Images may be combined into a single animated GIF using Gimp.

Images may be combined into video files using the **mencoder** program that comes with mplayer. For example, the following creates a 15 FPS silent MPEG4 file in an AVI container:

```
mencoder mf://*.png -mf w=640:h=480:fps=15:type=png -ovc lavc -lavcopts vcodec=mpeg4:mbd=2:trell -oac copy -o output
```

Images are exported from print layouts so at least one print layout must be defined. Print layouts permit adjustment of layout size and DPI, as well as the ability to add additional cartographic elements to the output images.

Animate Columns


Print Layout
Family
Animation Layer
students
Latitude Offset Column
SLATOFF
Longitude Offset Column
SLONGOFF
Duration (Frames)
50
Image Output Directory
software/mmqgis/tests/animate-lines/frames

The Animate Columns tool permits animation of map features in a single layer. The objects are moved over the specified *Duration* number of frames in a straight line based on the offsets specified in the *Latitude Offset Column* and *Longitude Offset Column*. Individual PNG images for each frame are written to the specified *Image Output Directory*

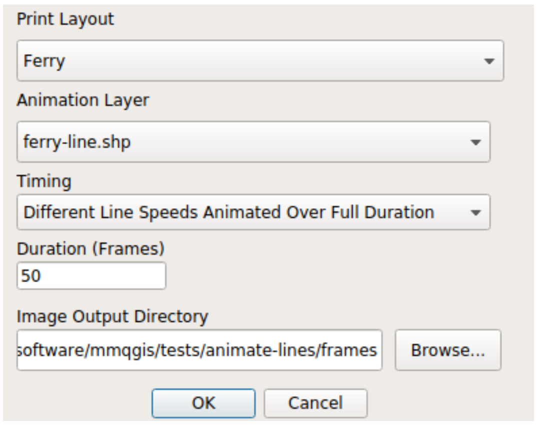
The motion is specified with offsets rather than absolute X/Y values to permit animation of lines and polygons. The offsets must be specified in the Coordinate Reference System of the layer being animated. Offsets for points may be calculated by:

```
        offset_long = (end_long - start_long)
        offset_lat = (end_lat - start_lat)
```

If you need more control over paths, you should use the Animate Rows tool, or Anita Graser's TimeManager plugin.

This tool can be slow and processing-intensive if animating a large number of features.

Animate Lines


Print Layout: Ferry

Animation Layer: ferry-line.shp

Timing: Different Line Speeds Animated Over Full Duration

Duration (Frames): 50

Image Output Directory: software/mmqgis/tests/animate-lines/frames

The Animate Lines tool facilitates animation of lines that grow to their full length over the specified *Duration* of the animation. Individual PNG images for each frame are written to the specified *Image Output Directory*

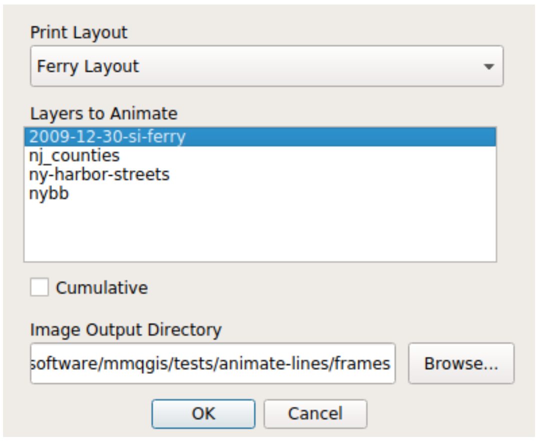
The timing of line growth can be specified in two ways. *Different Line Speeds Animated Over Ful Duration* will grow all lines at different speeds so they all complete their full path simultaneously at the end of the animation. *One Line Speed Determined By Longest Line* will grow all lines at the same linear speed, with the speed set by the longest line, which will complete at the end of the animation.

The direction of growth is determined by the order of the linestring points in the source file. Depending on how the lines were digitized, this direction may be opposite of what is desired. To control the direction of growth, line vertices in the desired order can be created in a CSV file and imported with the Geometry Import From CSV File.

If attempting to animate a layer of MultiLineStrings, this tool will combine the points from all line segments into a single line of points. This will add spurious line segments to the animation if the points in the line segments were not ordered in a contiguous manner when digitized.

Because this tool uses an edit session to modify geometry during animation, red vertex markers will be visible in the animation unless turned off in: Settings > Options > Digitizing > Vertex markers. Choosing "Show markers only for selected features" will prevent display of vertex markers during animation.

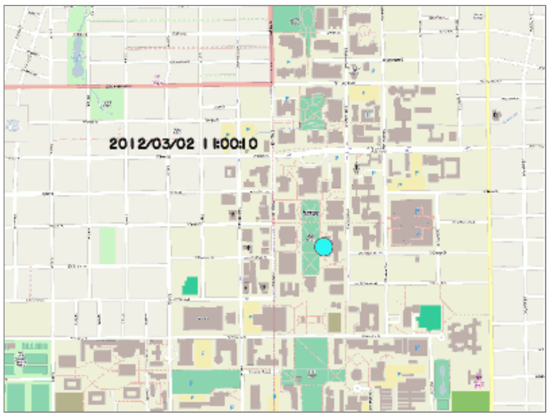
Animate Rows


Print Layout: Ferry Layout

Layers to Animate:
- 2009-12-30-si-ferry
- nj_counties
- ny-harbor-streets
- nybb

Cumulative: [ ]

Image Output Directory: software/mmqgis/tests/animate-lines/frames



The Animate Rows tool permits animation of map features in one or more layers. The features are moved by plotting successive rows in each layer selected in the *Layers to Animate* list box. The *Cumulative* checkbox causes features from previous rows to remain on the map. Individual PNG images for each frame are written to the specified *Image Output Directory*

If multiple layers are being animated simultaneously, the timing of rows in each layer will be the same. There is no capability for interpolating, frame skipping or frame duplication between layers of dissimilar length. If there are fewer rows in one animated layer than in another animated layer, no features from the shorter layer will be displayed after all rows in the shorter layer have been displayed.

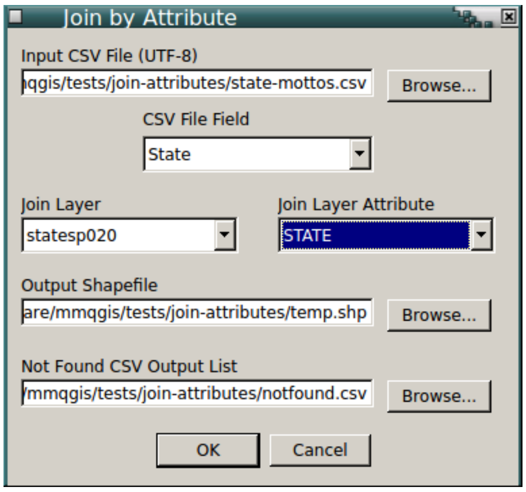
Animate Rows uses an edit buffer to sequentially display individual feature geometries. Therefore animated layers must be editable. If you wish to animate a non-editable layer (such as GPS tracking from a GPX file), you should save the layer to an editable source, such as a shapefile, and animate the editable layer.

If text display of timing is desired, a layer must be created with a column that has the desired display text, and that layer must be mapped with features labeled.

If you need fine-grained control by specific times or a less cumbersome way of displaying current time, you may want to consider using Anita Graser's TimeManager plugin.

Combine Tools

Attribute Join from CSV File


Join by Attribute

Input CSV File (UTF-8): qgis/tests/join-attributes/state-mottos.csv

CSV File Field: State

Join Layer: statesp020

Join Layer Attribute: STATE

Output Shapefile: are/mmqgis/tests/join-attributes/temp.shp

Not Found CSV Output List: /mmqgis/tests/join-attributes/notfound.csv

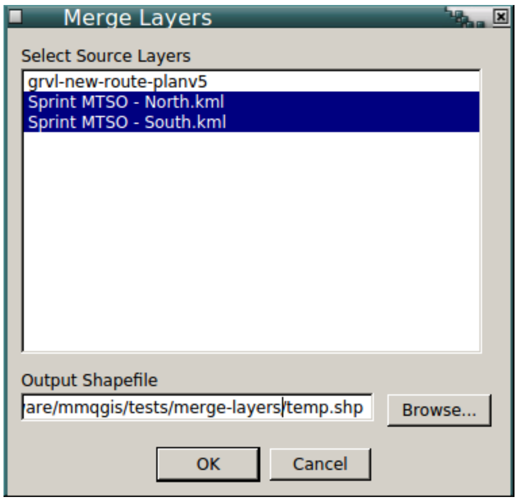
The join attributes tool permits import of attributes from a CSV file based on a join using a "key" field that is present both in the CSV file and in the attribute table of the map layer to which the data is being joined. The key is specified by selection boxes for CSV File Field and Join Layer Attribute and do not have to have the same name. The join is similar to an SQL join and if there are multiple occurances of a key in the CSV or map layer, there will be multiple combinations of the data in the output shapefile.

A box is also provided to specify the file where unmatched records from the CSV file are saved for further analysis.

Note that because CSV files contain no reliable type data in the header, data is always imported from CSV files as text. Text columns can be converted to numeric (floating point) using the "Text to Float" tool.

CSV files must be encoded in the UTF-8 character set. Although other 8-bit encodings (like Windoze ISO-8859-x) will work if only ASCII characters are present, non-ASCII characters may cause unpredictable behavior.

Merge Layers


Merge Layers

Select Source Layers:
grvl-new-route-planv5
Sprint MTSO - North.kml
Sprint MTSO - South.kml

Output Shapefile: are/mmqgis/tests/merge-layers/temp.shp

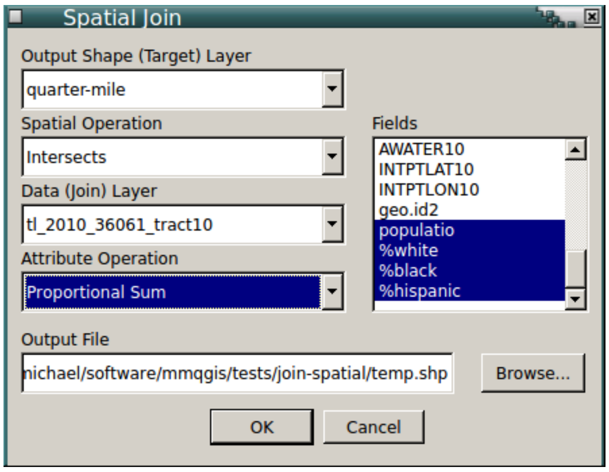
The merge layers tool merges features from multiple layers into a single shapefile and adds the merged shapefile to the project. One or more layers are selected from the "Select Source Layers" dialog list box and an output shapefile name is specified in the "Output Shapefile" dialog field.

Merged layers must all be the same geometry type (point, polygon, etc.).

If the source layers have different attribute fields (distinguished by name and type), the merged file will contain a set of all different fields from the source layers with NULL values inserted when a source layer does not have a specific output field.

Where fields with the same name in different layers have different types, the output field type will be string.

Spatial Join



A spatial join permits combination of information from different layers based on spatial relationship. The spatial joins are very common operations in standard GIS analysis.

The Output Shape (target) Layer indicates the layer that defines the shapes that will be the output from the operation.

The Data (join) Layer indicates the layer that will provide the data output from the operation.

Spatial Operation box allows specification of the relationship that should be used in the operation: Within, Intersects or Contains. Some relationships are not available for some combinations of shapes (ex: there is no within relationship when joining a polygon target layer with a point join layer).

Attribute Operation indicates what should be done when multiple features from the join layer have a relationship with the target layer: First, Sum, Average, and Proportional Sum. Proportional Sum indicates that the data from each related feature should be summed in proportion to the amount of join feature area (in map units) covered by the target feature. Some operations are not available with some combinations of shapes (ex: proportional sum is not available for point target layers).

The Fields box permits selection of the combined attributes from both source layers that will be included in the output. Multiple attributes can be selected. Selection capability is provided so that the output does not have to be cluttered with multiple unnecessary attributes.

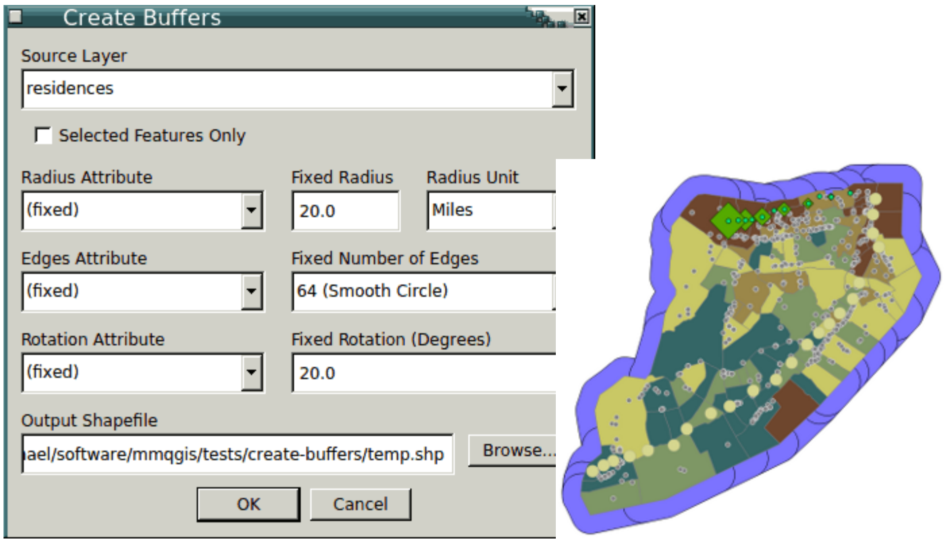
A COUNT attribute is automatically appended to the output that indicates the number of related join layer features contributing data to each feature from the target layer.

Features from the target layer that have no spatial relationship to the join layer are not included in the output.

The ranges given in the legend are rounded to four significant digits to avoid the extraneous precision associated with mathematical range calculation.

Create Tools

Create Buffers


Create Buffers

Source Layer: residences

☐ Selected Features Only

Radius Attribute: (fixed) | Fixed Radius: 20.0 | Radius Unit: Miles

Edges Attribute: (fixed) | Fixed Number of Edges: 64 (Smooth Circle)

Rotation Attribute: (fixed) | Fixed Rotation (Degrees): 20.0

Output Shapefile: ael/software/mmqgis/tests/create-buffers/temp.shp

This tool creates polygon buffers around points, lines and polygons.

This tool expresses buffer sizes as absolute linear distance units (miles, feet, meters, kilometers defined by WGS 84 great circle distance) rather than map units. The haversine formula is used to approximate meaningful radius distances independent of the original projection. While this may introduce some deviation from the original CRS, buffering is assumed in practice to be an inexact operation that can usually tolerate such discrepancy.

Buffer radius can be fixed for all buffers as specified in the dialog, or buffer sizes can be taken from an attribute in the source shape layer.

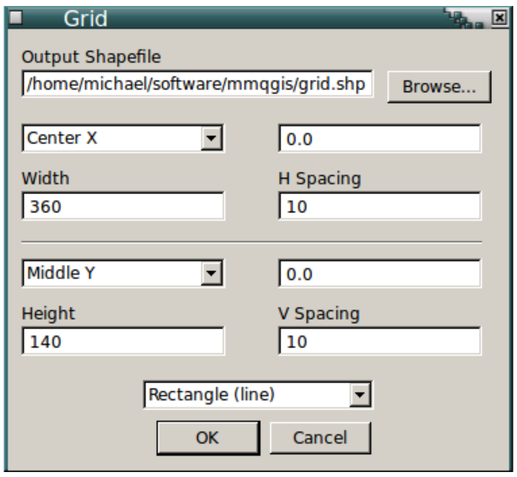
Point buffers can be created with different numbers of edges: triangles, diamonds, pentagons, hexagons, 32-edge circles or 64-edge circles. If other numbers of edges, or different numbers of edges for different points are needed, the number of edges can be taken from a numeric attribute in the source shape layer.

Point buffers can be rotated based on a fixed number of degrees, or based on a numeric attribute in the source shape layer that indicates the rotation for each individual feature. The first node of a point buffer at zero degrees rotation is directly north from the point being buffered.

Line buffers can be created as normal rounded-end, single-sided, or flat-ended. Flat ended buffers are created by dissolving a north-sided and south-sided buffer for each line. This may cause odd shapes when line ends point in directions that deviate significantly from the general direction of the line. With multiple-line features, each line segment is treated as a separate shape, so angular deviations between line segments may result in slivers.

Line buffers can also be north-, south-, east- or west-side only. The determination of which side of the line is buffered is determined by the angular direction of line as defined by the start and end points of the line. This may result in undesirable results if buffering layers that mix vertically- and horizontally-oriented features. This tool does not currently have the capability to buffer based on directional characteristics like traffic or water flow through the features that lines are used to represent. For lines that are exactly perpendicular, the west side is considered the north side for north side buffering. For lines that are exactly horizontal, the south side is considered the west side.

Create Grid Layer


Grid

Output Shapefile: /home/michael/software/mmqgis/grid.shp

Center X: 0.0

Width: 360 | H Spacing: 10

Middle Y: 0.0

Height: 140 | V Spacing: 10

Rectangle (line)

The MMQGIS grid tool creates a shapefile containing a grid of shapes.

The *Shape Type* combo box permits selection of six types of grids:

- Lines: Creates separate horizontal and vertical lines for each grid cell

- Rectangles: Creates separate rectangles for each grid cell
- Points: Creates points at each grid cell corner
- Random Points: Creates points for each grid cell, with each point randomly located in each grid cell. This is useful for creating distributions of points that are random but have a known average spatial regularity
- Diamonds: Creates interlocking diamonds (rectangles rotated 45 degrees)
- Hexagon: Creates a grid of interlocking hexagons. To maintain equilaterality, the ratio between the horizontal and vertical spacing is fixed (and different). Changing horizontal spacing will automatically change vertical spacing, and vice versa.

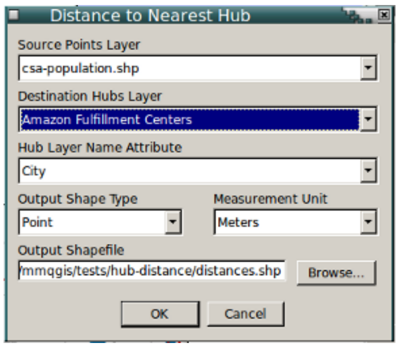
The grid is specified with an extent (left x, bottom y, right x and top y of the grid boundary), an *X Spacing* width (longitude), and a *Y Spacing* width (latitude).

The grid extent can be automatically calculated from the current map window, the extent of a map layer, or the entire world, or it can be specified with user-defined values.

Distance *Units* can be in degrees (WGS 84), the units of the coordinate reference system used by the project (Project Units), or the units for one of the map layers (Layer Units). If you wish to have a grid based on ground distance units (such as meters or feet), you should either define the project with a projected coordinate system or include a layer that has a projected coordinate system in the desired units.

The grid is saved to a shapefile specified in the *Output Shapefile* form field.

Hub Distance


Distance to Nearest Hub

Source Points Layer: csa-population.shp

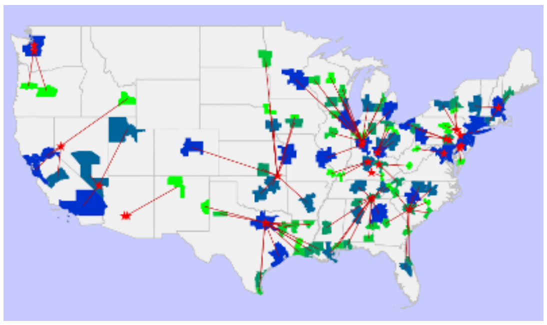
Destination Hubs Layer: Amazon Fulfillment Centers

Hub Layer Name Attribute: City

Output Shape Type: Point

Measurement Unit: Meters

Output Shapefile: /mmqgis/tests/hub-distance/distances.shp



The hub distance tool iterates through each feature on the source points layer and finds the closest "hub" from the destination hubs layer based on Ellipsoidal distance. The output is a shapefile containing all the attributes from the source layer along with a distance field and the name of the hub based on an attribute selected in the Hub Layer Name Attribute box.

The output shapefile can be either points (one for each source point) or lines from the source points to the closest hubs.

Distances can be specified in the units selected in the Measurement Unit combo box. Unless "Layer Units" is selected, points are converted to WGS84 and great-circle distances are calculated using the Haversine formula with Lambert's (1942) formula to correct for ellipsoidal flattening. Distance is then added as an attribute to the output file.

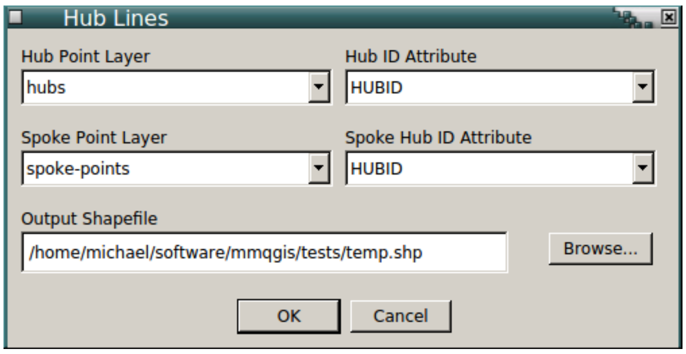
If your data is not WGS84 lat/long, it may be preferable to use "Layer Units" so the calculated distances are compatible with the distortions in your layer projection.

If "Layer Units" is selected, Euclidean distance is calculated in the coordinate system used for the two layers. No attempt is made to transform between coordinate systems, and using source and hub layers with different coordinate systems may yield odd and undesirable results.

If "Equally Distribute Points Across Hubs" is checked, the tool uses a different algorithm that sequentially assigns points to hubs, and then iterates through pairs of points while exchanging hubs when the combined point/hub distance of the pair would be smaller. This yields a graph with evenly distributed points and a minimized total point/hub distance. The minimization algorithm requires multiple passes and may be slow on large data sets. The minimized results should be mathematically valid, but may yield non-intuitive results depending on the spatial distribution of your data.

This tool does not incorporate any kind of network analysis, so if paths to hubs are non-linear (e.g. when dealing with city blocks), the closest great-circle or Euclidean distance may not be the closed in terms of traveling distance.

Hub Lines


Hub Lines

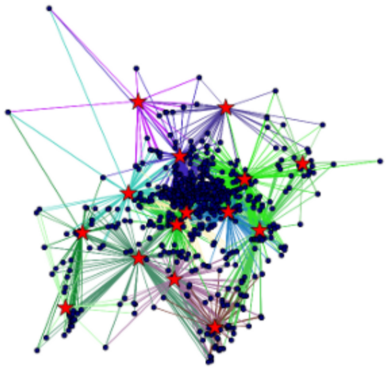
Hub Point Layer: hubs

Hub ID Attribute: HUBID

Spoke Point Layer: spoke-points

Spoke Hub ID Attribute: HUBID

Output Shapefile: /home/michael/software/mmqgis/tests/temp.shp

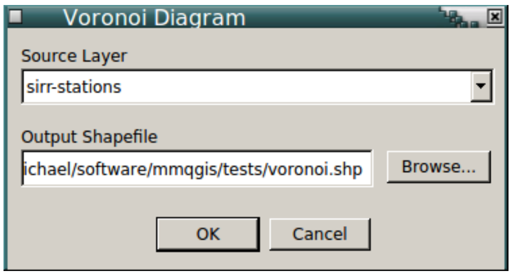


The hub lines tool creates hub and spoke diagrams with lines drawn from points on the "Spoke Point" layer to matching points in the "Hub Point" layer. Determination of which hub goes with each point is based on a match between the "Hub ID Attribute" on the hub points and the "Spoke Hub ID Attribute" on the spoke points.

The lines are output to a shapefile of lines and each line inherits all attributes from the matching spoke points.

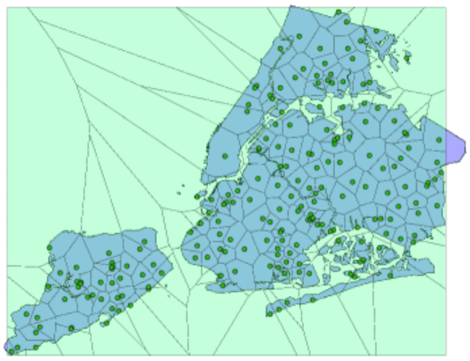
No attempt is made to transform between coordinate systems, so using source and hub layers with different coordinate systems may yield odd and undesirable results. This tool also does not incorporate any kind of network awareness (e.g. as when dealing with streets between city blocks), and all lines are straight lines from spoke to hub.

Voronoi Diagram


Voronoi Diagram

Source Layer: sirr-stations

Output Shapefile: ichael/software/mmqgis/tests/voronoi.shp



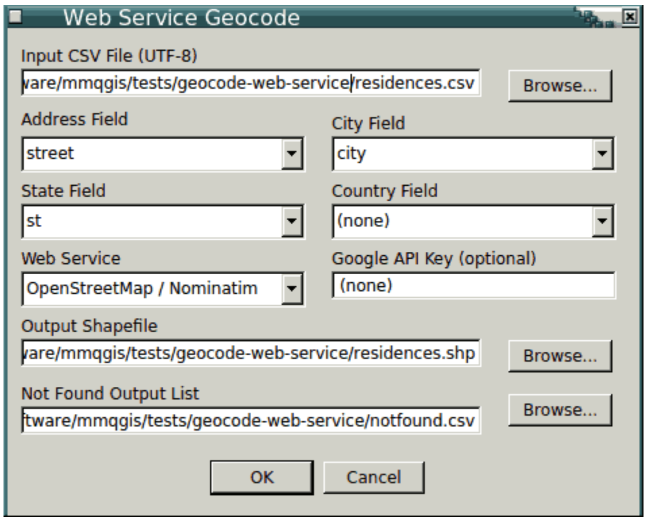
A Voronoi diagram is a collection of polygons, each surrounding individual points and representing the area around each point that is closer to that point than any other. The shapes are named after Russian mathematician Georgy Fedoseevich Voronoi, who published a formal definition in 1908, but the concept of this type of polygon extends back to Descartes in the 17th century. Similar polygons were famously used by physician John Snow to trace the source of a London cholera epidemic in 1854 and Voronoi diagrams remain useful in GIS for analyzing areas of influence associated with individual points within a collection.

The Voronoi diagram tool requires a point layer and outputs a polygon shapefile with the option to add it to the map. The boundary of the Voronoi diagram is the min/max extent of the points in the source layer.

The algorithm used to calculate the edges and nodes starts with tangents at the midpoints of lines between each point. The closest tangent is assumed to be a border and intersections to the border are calculated to circle each border until back to the beginning. The algorithm is computationally intensive, but still seems to run fairly quickly on a reasonably small set of points.

Geocode Tools

Geocode CSV With Web Service



Input CSV File (UTF-8): vare/mmqgis/tests/geocode-web-service/residences.csv
Address Field: street
City Field: city
State Field: st
Country Field: (none)
Web Service: OpenStreetMap / Nominatim
Google API Key (optional): (none)
Output Shapefile: vare/mmqgis/tests/geocode-web-service/residences.shp
Not Found Output List: tware/mmqgis/tests/geocode-web-service/notfound.csv

This tool imports addresses from a CSV file and uses either the Google Maps ™ API or the OpenStreetMap Nominatim web service to geocode those addresses. The result is a point shapefile that is added to the current map, along with a Not Found CSV file containing all rows that could not be geocoded (for whatever reason).

To use the Google geocoder, you will need to get an API key and include it in the *API Key* dialog box.

All columns from the input CSV file are added as attributes in the output shapefile. Addresses may be spread across as many as four different columns (street, city, state, country - which are concatenated for the query), although only one meaningful column is absolutely required (such as for a city/state combination).

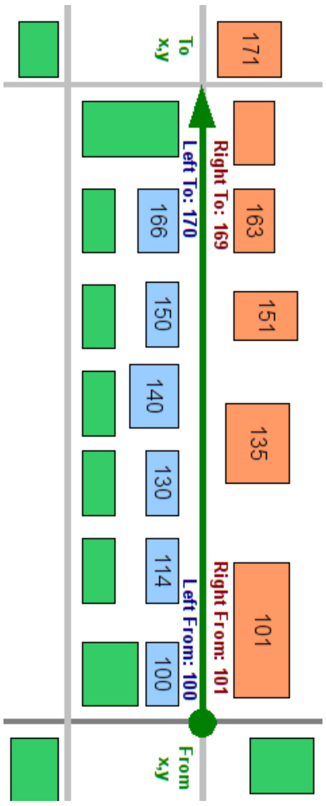
Two additional attributes are added to each shape to preserve exactly what the web service geocoded so that accuracy can be assessed. *addrtype* is the Google <type> element or OSM class attribute and indicates what kind of address type this is (street, route, etc). *addrlocat* is the Google <location_type> element or OSM type attribute and indicates the relationship of the coordinates to the addressed feature (rooftop, geometric center, interpolation, etc).

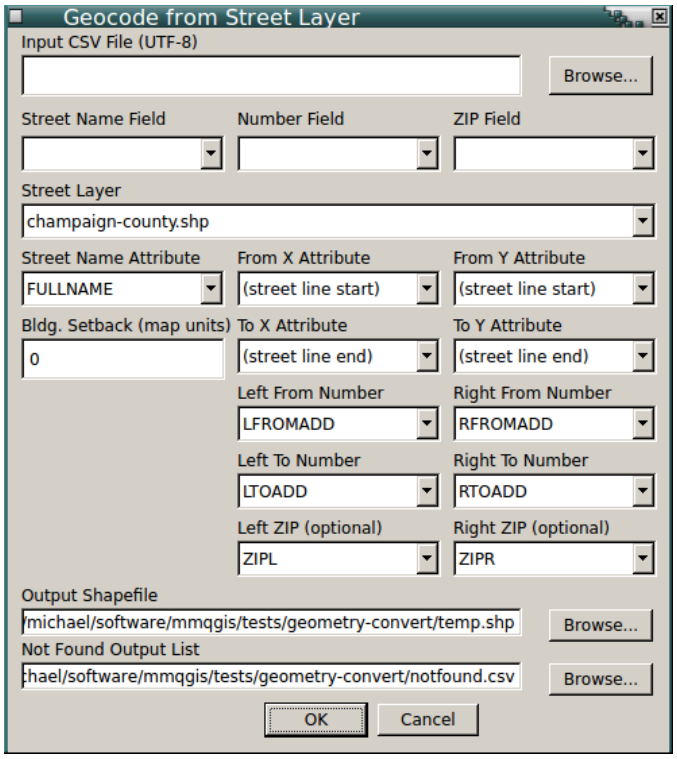
The input CSV file should be encoded in the UTF-8 character set. Although other 8-bit encodings (like Windoze ISO-8859-x) will work if only ASCII characters are present, non-ASCII characters may cause unpredictable behavior.

If the service returns more than one location for the address, only the first of the locations will be used for an output feature.

Use of this plugin requires an active Internet connection. Google places both rate and volume restrictions on the number of addresses that can be geocoded within various time limits. You should visit Google's Google Geocoding API page for more details, current information and Google's terms of service.

Geocode from Street Layer





Input CSV File (UTF-8):
Street Name Field:
Number Field:
ZIP Field:
Street Layer: champaign-county.shp
Street Name Attribute: FULLNAME
From X Attribute: (street line start)
From Y Attribute: (street line start)
Bldg. Setback (map units): 0
To X Attribute: (street line end)
To Y Attribute: (street line end)
Left From Number: LFROMADD
Right From Number: RFROMADD
Left To Number: LTOADD
Right To Number: RTOADD
Left ZIP (optional): ZIPL
Right ZIP (optional): ZIPR
Output Shapefile: /michael/software/mmqgis/tests/geometry-convert/temp.shp
Not Found Output List: :hael/software/mmqgis/tests/geometry-convert/notfound.csv

The street address geocoding tool requires a layer with street centerline features and attributes indicating the range of addresses associated with each feature.

Examples of street centerline shapefiles include the US Census Bureau's TIGER/Line Shapefiles and the New York City Department of City Planning's DCPLION files. Shapefiles created as ESRI "address locators" can be used with this geocoder, but newer version file geodatabases and older Access geodatabases cannot be used because these proprietary formats are not supported by QGIS. MapInfo tables are supported and can be used if these are provided as an alternative.

The street centerline layer should have attributes for FROM x/y and TO x/y (in map coordinates), a range of addresses on the left side, and a range of addresses on the right side.

Optionally, an attribute selector is given to permit the FROM/TO x/y from the shape lines themselves can be used ("street line start" and "street line end"). However, this assumes that the order of line vertices in the shapefile consistently starts with FROM or TO points, which may not be true and which may result in inconsistently geocoded locations.

Addresses (along with other attributes) are read in from a CSV file. Addresses should have separate attribute fields for number and street name. Optional ZIP Code fields are provided for additional accuracy when using TIGER/Line files. The attributes columns from the street centerline layer can also be selected, although when a layer is selected, the dialog will attempt to find columns with appropriate names.

The input address CSV file and the street centerline shapefile should be encoded in the UTF-8 character set. Although other 8-bit encodings (like Windoze ISO-8859-x) will work if only ASCII characters are present, non-ASCII characters may cause unpredictable behavior.

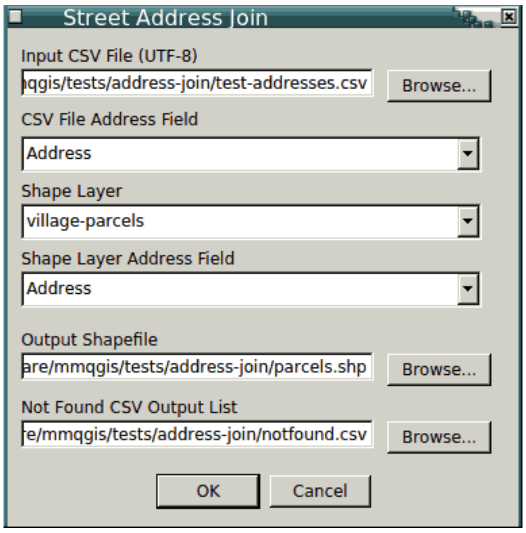
The "Building Setback" indicates how far the geocoded points should be set away from the street centerline (i.e. how far buildings are from the middle of the street, in map units).

To improve searching, street names are internally modified with common abbreviations such as "st" for "street" and "w" for "west". This add fuzziness to the search process that may result in unexpected results. This address handling may be augmented or refined in future releases.

As with many tools of this type, the matching and interpolation is quite fragile. Street names in the street layer must match the names in the CSV file exactly - "First Street" will not match "1st street". The tool will also only handle address street numbers that are entirely numeric - "172-10 Seventh Ave." or "1872a Main" will throw fatal errors.

Output is a new point shapefile and a CSV file listing which addresses were not matched.

Street Address Join



Input CSV File (UTF-8): qgis/tests/address-join/test-addresses.csv
CSV File Address Field: Address
Shape Layer: village-parcels
Shape Layer Address Field: Address
Output Shapefile: are/mmqgis/tests/address-join/parcels.shp
Not Found CSV Output List: e/mmqgis/tests/address-join/notfound.csv

The *Street Address Join* tool permits joining table data from a CSV file to shapes from a map layer based on a fuzzy match between street addresses.

The CSV file of data is selected with the *Input CSV File* box, and the column in the CSV file containing the address is selected in the *CSV Address File Field* combo box.

The map layer of shapes is selected from the *Shape Layer* combo box, and the attribute field in the layer containing the address to match to the CSV addresses is selected in the *Shape Layer Address Field* combo box.

The joined data and shapes will be written to the file given in the *Output Shapefile* box, and rows from the CSV file with no matching address in the map layer will be written to the file specified in the *Not Found CSV Output List*.

Addresses are complex and highly ambiguous, and the address parsing algorithm used in MMQGI is not as robust as libpostal and relies on many of the falsehoods programmers believe about addresses.

This tool works by normalizing addresses from both the CSV file and the map layer to a common format, and then performing direct string comparisons between the two lists of addresses to find matches. Some normalization transformations include:

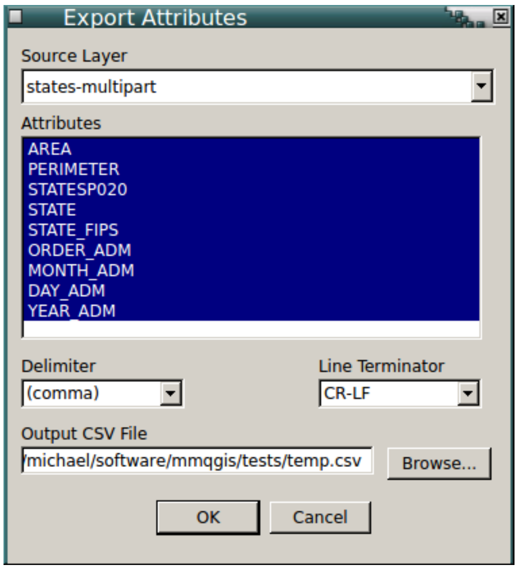
- All lower case
- Removal of periods and commas
- Removal of trailing numbers that appear to be unit numbers
- Removal of st, nd, rd, and th from after numbers
- Replacement of trailing street types (like boulevard, parkway, street, plaza, etc) with abbreviations (blvd, pkwy, st, pl)
- Separation into three parts: lot/house number, street name, and (optional) unit number

While many of these transformations increase rather than reduce ambiguity, the hope is that they reduce false positives and negatives within limited geographic areas.

This tool also does not currently provide capabilities for distinguishing duplicate addresses by zip code or city name. However, for simple tasks like matching lists of street addresses within limited geographic areas (like cities or counties) in the USA, this tool may be more precise than using a web geocoder or street layer.

Import / Export Tools

Attributes Export to CSV File


Export Attributes

Source Layer: states-multipart

Attributes:
AREA
PERIMETER
STATESP020
STATE
STATE_FIPS
ORDER_ADM
MONTH_ADM
DAY_ADM
YEAR_ADM

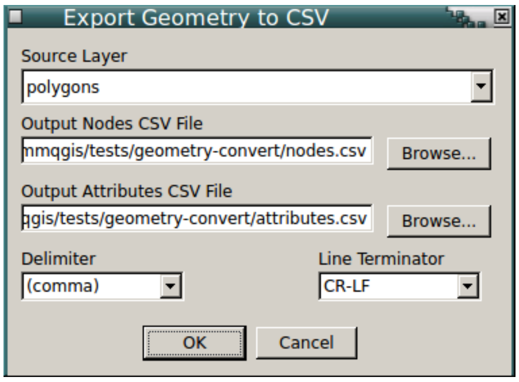
Delimiter: (comma)
Line Terminator: CR-LF

Output CSV File: /michael/software/mmqgis/tests/temp.csv

The export attributes tool saves attributes from a map layer to a CSV file, which can then be viewed or edited. This can be helpful when you have some use for the attribute data without the associated geographic data. When dealing with data sets that have large numbers of rows or columns, viewing or searching exported data can be simpler or faster than the QGIS open attribute table, which can be unacceptably slow with large data sets.

A multiple-selection list box on the dialog permits selection of the attribute columns to export to a CSV file specified at the bottom of the form.

Geometry Export To CSV


Export Geometry to CSV

Source Layer: polygons

Output Nodes CSV File: mmqgis/tests/geometry-convert/nodes.csv

Output Attributes CSV File: qgis/tests/geometry-convert/attributes.csv

Delimiter: (comma)
Line Terminator: CR-LF

Geometry Export to CSV exports points, polylines or polygons to comma-separated variable (CSV) files.

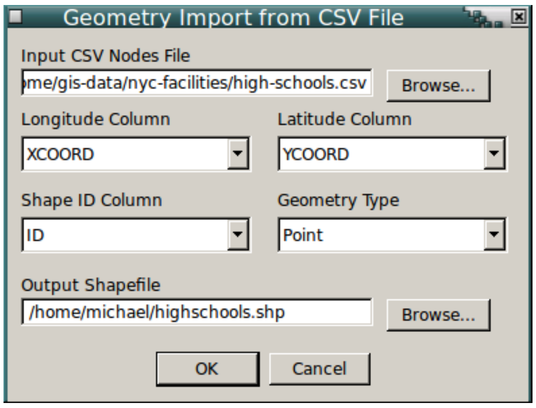
For point layers, the output CSV file includes an "x" column, a "y" column, and a "shapeid" column in addition to all attributes associated with each point.

For polyline and polygon layers, individual nodes of each shape are given a separate row in the output file. Nodes for each shape have the same "shapeid" value. Node lines are written in the same sequential order that nodes occur in the shape. Attributes for each shape are exported to a separate CSV file, with a "shapeid" column referencing the associated nodes in the node CSV file.

Elevation (z) export is not currently supported because QGIS geometries do not currently support z-axis / 2.5D

Polygons with holes (inner rings) have hole polygons saved under separate shapeids that contains "ring" in the name.

Geometry Import From CSV


Geometry Import from CSV File

Input CSV Nodes File: me/gis-data/nyc-facilities/high-schools.csv

Longitude Column: XCOORD

Latitude Column: YCOORD

Shape ID Column: ID

Geometry Type: Point

Output Shapefile: /home/michael/highschools.shp

Geometry Import From CSV imports points, polylines exports points, polylines or polygons to comma-separated variable (CSV) files. A combo box is provided to indicate whether the nodes from the CSV file should be imported as discrete points, as lines or as polygons.

For point layers, only a single input CSV file is needed that includes latitude and longitude columns. All other columns are imported as attributes. This is essentially the same as the "Add Delimited Text Layer" native QGIS command. Attributes are always imported as text strings and if columns need to be converted to integer or floating-point types, the MMQGIS "Text to Float" command should be used.

For polyline and polygon layers, individual nodes of each shape should be provided on separate rows in an input CSV node file. The file should have three columns: Latitude, Longitude, and Shape ID. Any additional columns will be ignored. Nodes for each shape should have the same unique Shape ID value and should occur in the file in the same sequence that they will be added to the shape.

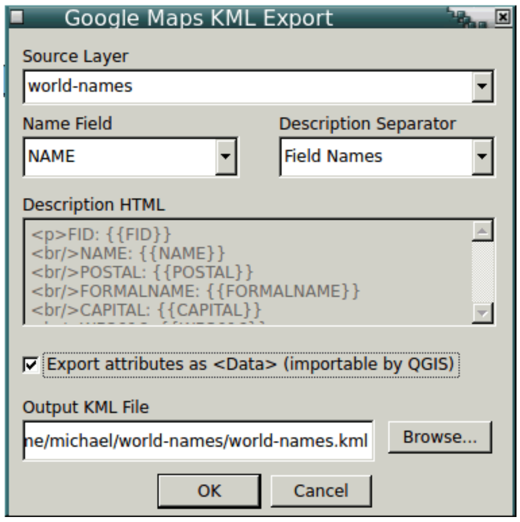
If the imported polylines or polygons have additional attributes to be imported, they should be placed in a separate CSV file with a Shape ID column to join to the imported nodes using the MMQGIS "Attributes Join From CSV File" command.

Other than providing a compliment to the "Geometry Export To CSV" command, this command can be used to import transit "shapes.txt" data from General Transit Feed Specification (GTFS) releases by transit agencies.

The input CSV file(s) should be encoded in the UTF-8 character set. Although other 8-bit encodings (like Windoze ISO-8859-x) will work if only ASCII characters are present, non-ASCII characters may cause unpredictable behavior.

Type errors such as non-numeric latitude/longitude may throw a cryptic Python error.

Google Maps (tm) KML Export


Google Maps KML Export

Source Layer: world-names

Name Field: NAME

Description Separator: Field Names

Description HTML:

```
<p>FID: {{FID}}
<br/>NAME: {{NAME}}
<br/>POSTAL: {{POSTAL}}
<br/>FORMALNAME: {{FORMALNAME}}
<br/>CAPITAL: {{CAPITAL}}
```

☑ Export attributes as <Data> (importable by QGIS)

Output KML File: ne/michael/world-names/world-names.kml

Although QGIS permits seamless import and export of formally correct KML files, the KML is often not optimal for import into Google Maps (tm).

This tool exports features to KML with the capability to explicitly specify fields for the Name and Description that are always displayed in the current (as of this writing) default Google Maps (tm) interface.

Multiple fields can be combined in the description. The Description Separator indicates how multiple fields should be separated: as separate paragraphs, preceded by a field name on separate lines in a single paragraph, simply separated by commas in a single paragraph, or using custom HTML.

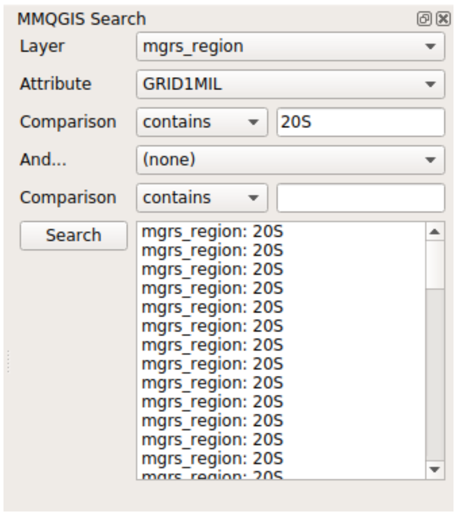
With custom HTML, field values can be included enclosed in double braces. For example, the following would create a description including the NAME, POP and YEARINC fields for each placemark written to the KML file.

```
<p>City: {{NAME}}
<br/>Population: {{POP}}
<br/>Incorporated: {{YEARINC}}</p>
```

The tool can also include <ExtendedData> with all of the field values by selecting **Export attributes as <Data>.** This permits the KML file to be used to store geometries and attributes that can then be imported back into QGIS or other GIS software. However, when simply creating a KML file for display on a web map, you may want to switch this off to minimize file size (load time) and avoid cluttering the popup with unnecessary information, especially when you have a large number of attributes.

This tool exports points using the default Google Maps (tm) PNG icons. Since a full palette of icon colors is not available, icon colors are approximated based on the RGB values of point layer symbol colors.

Search


MMQGIS Search
Layer: mgrs_region
Attribute: GRID1MIL
Comparison: contains 20S
And...: (none)
Comparison: contains

mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S
mgrs_region: 20S

The search tool permits interactive browsing of features that match the specified search criteria. Results are displayed in a list box at the bottom of the search dialog.

Clicking one of the features on the results box moves the map canvas to center that feature in the window.

This tool also provides options in the layer selection combo box for searching addresses using Google Maps or Nominatim / Open Street Map. This option only searches for addresses since generalized serches (such as for restaurants or landmarks) require use of the Google Places API, which requires a website-specific API key that cannot be included in a generalized plugin. Selection of an address only pans the map display to place the location in the center of the display, and no markings are added to the map.

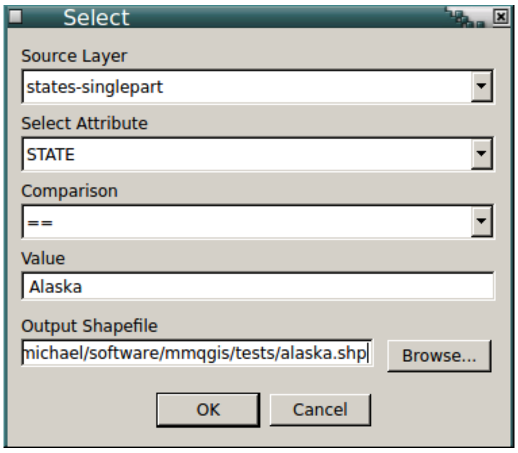
Multiple selections in the results list are possible using the CTRL and SHIFT keys. If multiple features are selected, the map canvas centers on a central area between the centroids of all selected features.

Layer feature searches can be performed on one or two specific attribute fields. If a value is specified for a second attribute, a feature must match BOTH conditions to be included in the results list.

Layer feature searches can also be made across all layers by choosing the *[All Layers]* in the layers combo box, and/or all fields in the specified layers by choosing the *[all]* attributes option in the attribute combo box. Choosing to search all layers or attributes can be time-consuming, and/or yield an unmanageable set of search result features.

Possible comparisons on numeric or string attributes are the common permutations of equal, greater than, or less than. String attributes can be searched with "contains" or "begins with." Searches are case insensitive.

Select



Source Layer: states-singlepart
Select Attribute: STATE
Comparison: ==
Value: Alaska
Output Shapefile: michael/software/mmqgis/tests/alaska.shp

The select tool permits export of features from a layer based on comparison to attribute values. The selection is written to a new shapefile.

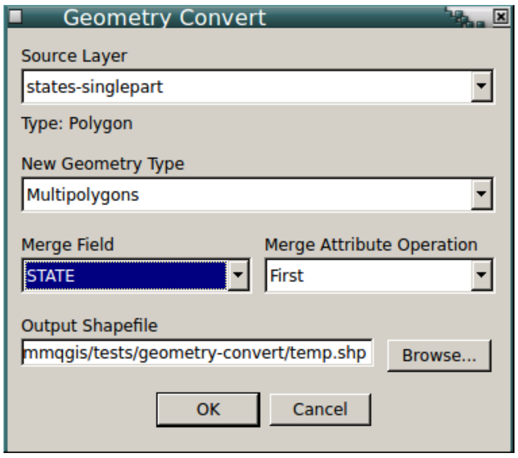
While the native QGIS export selection capability is suitable for most select operations, this tool may save time when selecting from extremely large data sets. The frequent and explicit status updates also provide awareness and assurance that the software is not frozen.

The select tool can select based on a single attribute (Select Attribute) a comparison (equal, not equal, greater, greater equal, less, less equal, begins with and contains), and a user-specified comparison value. Values can be numeric or text and type conversion will be made based on the native type of the specified selection attribute. The "begins with" and "contains" comparisons are meant for strings and may give unpredictable results when applied to integer or floating-point attributes.

No capability is provided for complex selections or selection based on multiple attributes. However, is is generally possible, albeit cumbersome, to execute multiple selection operations and/or merge operations from separate output files to perform complex selections.

Modify Tools

Convert Geometry

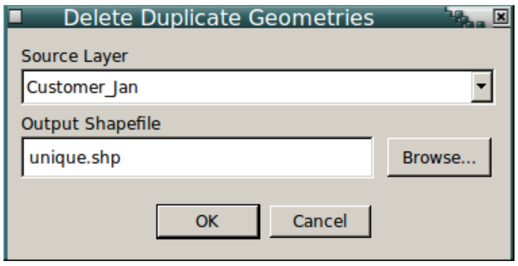


Source Layer: states-singlepart
Type: Polygon
New Geometry Type: Multipolygons
Merge Field: STATE
Merge Attribute Operation: First
Output Shapefile: mmqgis/tests/geometry-convert/temp.shp

The Geometry Convert tool changes the geometry types of shapes.

- All shapes other than points can be converted to centroids or individual node points
- Multi-part shapes (multi-point, multi-line, multi-polygon) can be decomposed to individual parts (point, line, polygon, respectively)
- Single-part shapes can be merged into multi-part shapes based on a common attribute (Merge Field). Numeric attributes from from the multiple source features will be handled using the Merge Attribute Operation (First or Sum). String attributes from multiple source features are always merged using the value from the first encountered feature.
- Polygons and multi-polygons can be converted to lines
- Shapes with elevation (2.5-D) are converted to X/Y (2-D). This can be helpful for converting GPS or KML layers to 2-D for merging with other 2-D layers (elevation is discarded)

Delete Duplicate Geometries



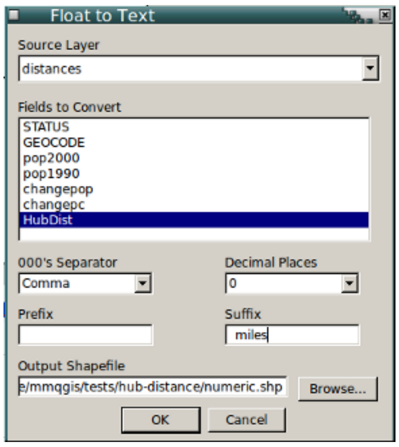
Source Layer: Customer_Jan

Output Shapefile: unique.shp

The Delete Duplicate Geometries tool removes duplicates shapes from a layer as it saves that layer to a shapefile. Two shapes are considered duplicates if their geometries are exactly identical, as determined by the QgsGeometry equals() function. Attributes are not considered, only geometry.

To improve speed, this tool loads all geometries into memory - which may cause memory issues with large datasets or large numbers of complex polygons.

Float to Text



Source Layer: distances

Fields to Convert: STATUS, GEOCODE, pop2000, pop1990, changepop, changepc, HubDist (selected)

000's Separator: Comma

Decimal Places: 0

Prefix:

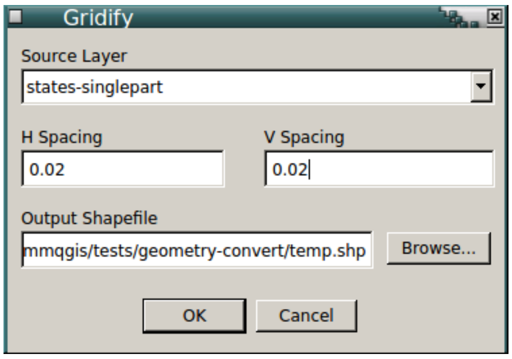
Suffix: miles

Output Shapefile: e/mmqgis/tests/hub-distance/numeric.shp

As of this writing, QGIS does not have a feature for number formatting when numeric values are used to label features. The Text to Float converts integer and floating point values to strings, with the ability to specify thousands separators, the number of decimal digits, prefixes and suffixes (eg. dollar sign prefix or "miles" suffix). This is the opposite of the "Text to Float" tool.

The Source Layer box is used to select the layer containing the attributes for conversion. The Fields to Convert selection box is used to select the attributes to be converted. Multiple fields may be selected. The Output Shapefile box selects the name of the file where the converted records will be written.

Gridify

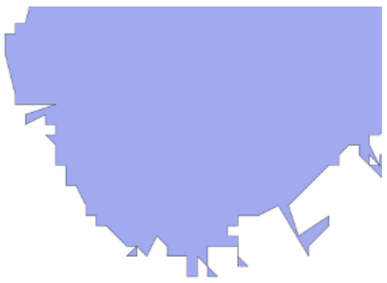


Source Layer: states-singlepart

H Spacing: 0.02

V Spacing: 0.02

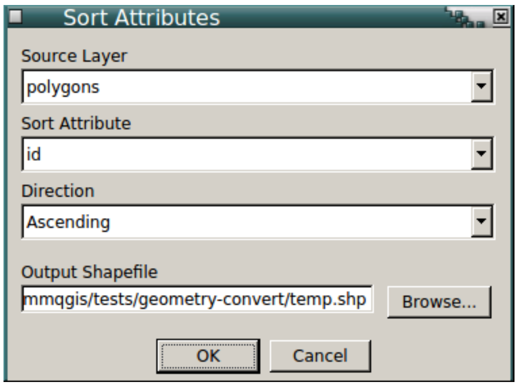
Output Shapefile: mmqgis/tests/geometry-convert/temp.shp



The gridify tool permits simplification of points, lines and polygons in a shapefile by aligning all vertices to a specified grid and then removing redundant points. This makes it possible to significantly improve display refresh time when working with shapefiles that are more detailed than is necessary for the final map. However, the gridification process can result in some odd artifacts when viewed at higher resolutions, particularly in dealing with coves and other appendages along the edges of larger polygons.

The alignment grid is specified with horizontal and vertical spacing. Defaults are based on 0.5% of the extent of the target layer. The gridified shapes are saved to a shapefile specified in the "Output Shapefile" form field.

Sort



Sort Attributes

Source Layer: polygons

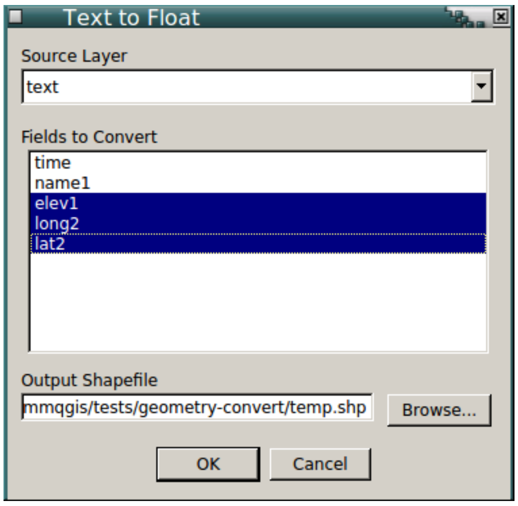
Sort Attribute: id

Direction: Ascending

Output Shapefile: mmqgis/tests/geometry-convert/temp.shp

The sort tool permits sorting of attribute data and associated shapefile data to maintain alignment. Sorting is based on a single attribute column and the result of the sort is saved to a shapefile specified in the Output Shapefile box. Sorting by attribute can be helpful for ordered CSV export or when viewing the Attribute Table

Text to Float



Text to Float

Source Layer: text

Fields to Convert: time, name1, elev1, long2, lat2

Output Shapefile: mmqgis/tests/geometry-convert/temp.shp

Numeric values may be stored as text strings in shapefiles, making it impossible to use them for symbology values. The Text to Float tool converts string attributes to floating point.

Source Layer selects the layer containing the attributes for conversion. The Fields to Convert selection box is used to select the attributes to be converted. Multiple fields may be selected. The Output Shapefile box selects the name of the file where the converted records will be written.

Care should be exercised in selecting fields since fields containing strings that cannot be converted to floating point (e.g. strings with non-numeric characters) will be assumed to have a value of zero.

Integer fields are converted to floating point, which should result in no loss of precision but may increase shapefile size. However, integer strings exceeding MAXINT will get unpredictable values and there may be some loss of precision when converting real values with large numbers of digits of significance.

Version History

v2018.08.09
Fix KML export bug truncating custom description HTML after final parameter substitution (Polczynski)
Fix KML styling when symbols have multiple styles (QgsFeatureRenderer::Capability == MoreSymbolsPerFeature) (Wren)
KML export ignores auxiliary storage parameters to avoid failure when they are included by default from the dialog (Wren)

v2018.07.10
Finish migrating search tool to 3.x (Tenzin, Yaremchuk)
Change search tool from dialog to dockable widget (Yaremchuk)
Grid tool dialog bug fix so specific offsets can be specified (Pampel)
KML opacity export: handle symbol opacity, color alpha, and layer opacity (Wrenn)
Increase CSV format sniff from 4096 to 8192 bytes to handle CSV files with large numbers of columns (Cleto)

v2018.02.27
Beta release for QGIS 3, Python 3, and QT 5

v2018.01.02
Fix crash in hex grid dialog when x or y spacing is blank or invalid (Gem)

Restore cumulative option on Animate Rows (Sloan)
Hub lines and hub distance use centroid() rather than boundingBox().center() (DeJesus)

v2017.05.14
Fix mmqgis_load_combo_box_with_vector_layers() so it can fill both QListWidget and QComboBox and doesn't throw error in merge_layers (Lavure)
Add all layers and all attributes options to search (Hamilton)
Fix space replacement with '+' so Google search works

v2017.02.28
Merge layers dialog uses order of layers from layer TOC (Stautz)
Handle both symbol and layer transparency in KML export (Wrenn)
Graceful failure when 2.5D line passed to animate_lines (Brandsma)
Animate Rows uses editor buffer rather than subset string, which failed to animate if the first field was not sequential (Brandsma)

v2017.02.12
Upgraded mmqgis_read_csv_header() to gracefully fail with non-UTF-8 characters (Barta)
Upgraded mmqgis_geocode_web_service() to gracefully fail with non-UTF-8 characters (Wright)
Fixed bug in mmqgis_animate_lines() that was causing line features to be considered invalid (Lawhead)
Fixed KML export crash due to inconsistent API spec for derived class QgsFeatureRendererV2:symbols() (Wrenn)

v2016.12.29
Get KML export working again under 2.18 (Asch)
KML export multipolygons as MultiGeometry
KML attribute export as ExtendedData
Customizable KML description HTML
Fix geometry_export_to_csv non-ASCII character errors (Hollander)
Fix attribute join mangling large integers
Dialog layer selection boxes use layer order from TOC (Petersen)
Invalid rightzipcode assignment in street geocode (Janko)
Revert street geocoder to simpler street name normalizer (Torres)
Accept LongLong and ULongLong as spatial join numeric types under Windoze 64-bit (Petersen)
Spatial join (and most tools) include joined fields (Petersen)
layer.fields() allows removal of quirky layer.dataProvider.fields()
Change merge attribute handling to convert different attribute types with same name to string

v2016.10.26
Remove forgotten "math." prefixes in upgraded distance and bearing calculation code

v2016.10.19
Remove hub_lines debug lines that cause crash on Windows

v2016.10.18
Put "from math import *" back in so math functions do not need "math." prefix
Hub lines ignore feature.geometry() == None

v2016.10.12
Add meaninful text conversion of Date and DateTime in geometry export
Change http to https for Google geocoding with API key
KML export and gridify ignore feature.geometry() == None

v2016.08.18
Fix bug in hexagon grid with incorrect calculation of fixed aspect ratio
Make minimum QGIS version 2.10

v2016.08.17
Address join added faster string search and handing of text versions of numbered streets (Fifth vs 5th)
Street Geocode add handling for 2-D and 2.5-D lines
Street Geocode add geometry type test at start for more-graceful handling of wrong geometry type
Fixed incorrectly reversed aspect ratio on hexagonal grid

v2016.08.04
Bug fix missing header names when exporting all attributes
Consolidate grid and point tools (Watke)
API key field for Google geocode (Shaw)
Add retries and system error messages for web geocoding (Waelti, Shaw)
Street address join tool (Rotondaro)
Add line centroids (Kolajo)
Add rotation and number of edges for point buffers (Thompson)
Remove Color Ramp, Label Layer and Delete Column tools, which all now have decent native equivalents
Replace all layer.dataProvider() with direct QgsLayer calls (except fields)

v2016.05.15
Add notfound handling of non-integer street numbers in street geocoding (Barowicz)
Add username/password to proxy handling (Ghensi)
Add handling of joined attributes in attribute_export (Krticka)

v2016.01.31
Add decimal mark for attribute export
Use xml.etree.ElementTree for Nominatim XML because it inconsistently uses double quotes or single quotes for attributes
Non-string selection attribute for animate rows
Change web geocode to use urllib2 and add proxy handling
Add graceful geocoding stop when exceeding Google daily quota
Bug fix so buffers radius from attribute works

v2015.11.03
Add handling of polygons with inner rings to export KML
Added semicolon as separator option in export attributes

v2015.10.12
Fix crash looking for dataProvider.fields() in spatial join with raster layers
Remove errant debug line 4122 in mmqgis_merge()

v2015.08.08
Update tab order in all dialogs to be top-down, left-to-right
Kludge dialect.escapechar in mmqgis_geocode_web_service to prevent CSV crash with non-ASCII characters in Windoze

v2015.05.27
Fix crash in join with OpenLayers layers
Fix crash in mmqgis_geocode_web_service_dialog() if non-CSV file read
Add equal distribution option to hub distance

v2015.02.11
QDate conversion to yyyy-MM-dd string in export attributes to CSV
Create Grid Points Layer tool
Animate Lines tool

v2015.01.29
Correct/swap top/bottom attribute name on create grid
Spatial join converts integer attributes to real for sum/average/proportional-sum
Spatial join error check duplicate and ambiguous field names
Spatial join bug fix attribute sum/average/proportional-sum

v2014.09.19
Allow maximum 10-character field names from Join CSV rather than erroneous 9-character limit
Bug fix Join CSV handling of multiply-duplicated truncated field names so fields are not lost

v2014.08.25
Work around KML export style and symbolForFeature() API crash bug using start/stopRender()

v2014.08.24
CSV header conversion from UTF-8 so field names can be non-ASCII
KML export using io.open(encoding=utf-8) to support non-ASCII characters
Bug fix to save hub distance shapefile as WGS84 rather than source CRS

v2014.08.03
CSV format sniff read 4096 rather than 2048 bytes so error is not thrown on files with long lines

v2014.07.25
Add pixmap parameter to saveAsImage() so animation works with v2.4 asynchronous map refresh

v2014.07.10
mmqgis_merge() add case insensitivity to layer names
mmqgis_merge() add test for type mismatch for attributes with the same name
layer.dataProvider().crs() is now just layer.crs()
Add graceful failure when join by attribute or convert geometry type with no layers
Add "Only selected features" option for buffers

v2014.06.08
Add "Mixed" symbol types to color ramp
Add Red-Yellow and Yellow-Red to presets
Add escapechar to dialect for attribute join notfound CSV write
Fix mmqgis_round() so created grid centers correctly in WGS 84
Add addrtype and addrlocat fields for web geocoded addresses
Upgrade of animate columns to use rolled-back editing sessions rather than temporary layers

Upgrade of animate rows to use setSubsetString instead of temporary layers
Upgrade search dialog with Google and OSM search options

v2014.02.28
Add duplicate node check to Voronoi to avoid creating invalid geometries

v2014.02.24
Remove leading zeros in metadata.txt version numbers to avoid version number update loop since http://plugins.qgis.org strips leading zeros.

v2014.01.06
Remove debug lines from from mmqgis_geocode_street_layer() that were causing IOError on Windoze.

v2014.01.01
Add OpenStreetMap/Nomatim as web service geocoding option along with the Google
Add absolute measurement units to Distance to Nearest Hub
Add flat-end and single-sided buffers for lines

v2013.07.16
Upgrade mmqgis_search() to use QgsExpression for improved speed

v2013.07.15
Add Search tool for interactively browsing features
Add setDragDropMode() to merge layers list so layer merging can be ordered.

v2013.06.14
Inner rings (holes) are saved as separate polygons in Geometry Export
metadata.txt minimum version now has to be 2.0.0 to be fetched from repository, even though Master is 1.9
Fix code broken by more API changes
QgsFeature::attributes() no longer QVariant (API change in QVariant typecasting?)
QVariant toString(), toDouble() and toInt() replaced by unicode(), float() and int()
Coded substitute for removed QgsVectorLayer::featureAtId()
Conditional call to isUsingRendererV2() - old renderer V1 has been removed

v2013.03.23
Fix overlap/holes topology problem in create grid layer

v2013.03.14
Extensive upgrade to work with new QGIS vector API v1.9
Menu moved to top-level menu bar and reorganized
Continued fixes for UTF-8 support in all tools
Addition of create buffers, spatial join, and Google (tm) maps KML export
Enhance convert geometry tool to handle conversion from singlepart to multipart
Enhancement of color ramp tool (although the QGIS new symbology interface makes this tool less useful)
Addition of ZIP field to street geocode for TIGER line files

v2012.12.07
Replace QgsRenderer with QgsRendererV2 in animation tools (legacy renderer unsupported in Windoze, V2 unsupported in Linux)

v2012.10.25
__init__.py syntax error

v2012.10.19
Add urllib.quote() to Google geocode to handle non-ASCII address characters

v2012.08.28
mmqgis_select() uses unicode() instead of str() for string conversions
Upgrade street geocode to be able to use start/end x/y from layer geometries

v2012.05.10
Minimum qgis version now 1.4
Add exception handling in grid dialog for crs functions new to qgis 1.7 and 2.0
New tool: Geometry Convert

v2012.05.06
New tools: Animate Rows, Animate Columns, Delete Columns, Float to Text
text_to_float handles numbers with comma separators
Hub distance now calculated in meters, Km, feet or miles using QgsDistanceArea
mmqgis_library functions now return error message rather than creating error message box
mmqgis_library functions now all have addlayer parameter

v2012.04.06
Catch CSV sniffer errors in attribute join, Google geocode, and geocode streets

Change qt.4 legacy menu activated() to triggered()
Add submenus to facilitate future additions

v2012.01.11
Add Hub Lines tool
Add Delete Duplicate Geometries

v2011.11.12
Sort by attribute sorts int and float columns numerically

v2011.10.08
Commented out debug lines - causing "Bad file descriptor" error on Windoze

v2011.10.07
Fixed bug in raster color map - slightly better interpolation algorithm

v2011.10.06
Added is_float() tester for import geometry
Add explicit qgis parameter for mmqgis_read_csv_header()
Add notes for lack of UTF-8 and Unicode
Add direction for sort
Remove google geocode print debug line 494
Replace color_ramp /tmp directory with mkstemp()

v2011.09.14
Replace all str() with unicode() in mmqgis_dialogs.py to handle non-ASCII characters.

v2011.08.27
Add delimiter and line terminator options to export attributes and export geometry
Change attribute join text encoding from utf-8 to iso-8859-1 (M$-Windoze charset)
CSV attribute and geometry export character coding iso-8859-1
Fix case sensitivity in attribute join duplicate field name correction

v2011.08.22
Bug fix - unicode text handling in join attributes and export geometry

v2011.08.16
Bug fix - disable plugin removes menu entries - mmqgis_menu.unload()

v2011.07.30
Bug fix - bad shape ID on import geometries of points
Add error handling for CSV import delimiter sniffer exceptions
New grid types: rectangular polygon, diamond and hexagon

v2011.03.14
Add export / import geometries to / from CSV

v2011.01.21
Modify mmqgis_label_point() to use unicode() rather than str() for point labeling

v2011.01.08
Add to Official QGIS repository

v2010.09.29
- Added color map
- Added Google (tm) geocode
- Extensive internal reorganization of code
- Created mmqgis_library.py

v2010.01.02 - Added hub distance and select tools

v2009.09.04 - Added text to float tool

v2009.09.01 - Added merge layers tool

v2009.08.28 - Initial public release

I would say there is a certain bias toward Manhattan when you make a book about New York...Whether you are conscious of it or not, you are contributing to this record of a place as it exists in people's imaginations. And New York as it exists in people's imaginations is mostly Manhattan. (Eric Himmel)
