Quantum GIS User Guide

Version 0.6 *'Simon'*

Gary E. Sherman
Tim Sutton
Radim Blazek (GRASS)
Lars Luthman (GPS Plugin)

December 2004

CONTENTS

QGIS Tips

CHAPTER 1: Introduction

Welcome to the wonderful world of Geographical Information Systems (GIS)! Quantum GIS (QGIS) is an Open Source Geographic Information System. The project was born in May of 2002 and was established as a project on SourceForge in June of the same year. We've worked hard to make GIS software (which is traditionaly expensive commerical software) a viable prospect for anyone with basic access to a Personal Computer. QGIS currently runs on most Unix platforms, Window, and OS X. QGIS is developed using the Qt toolkit (http://www.trolltech.com) and C++. This means that QGIS feels snappy to use and has a pleasing, easy to use graphical user interface.

QGIS aims to be an easy to use GIS, providing common functions and features. The initial goal was to provide a GIS data viewer. QGIS has reached that point in its evolution and is being used by many for their daily GIS data viewing needs. QGIS supports a number of raster and vector data formats, with new support easily added using the plugin architecture (see Appendix A for a full list of currenly supported data formats). QGIS is released under the GNU Public License (GPL). Devloping QGIS under this license means that you can (if you want to) 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 is also available as Appendix B.

Note: The latest version of this document can always be found at
http://qgis.sourceforge.net/docs/userguide.html

1.1 Major Features

QGIS has many common GIS features and functions. The major features are listed below.

1. Support for spatially enabled PostgreSQL tables using PostGIS
2. Support for ESRI shapefiles and other vector formats support by the OGR library, including MapInfo files
3. Identify features
4. Display attribute table
5. Select features
6. Label features
7. Persistent selections
8. Save and restore projects
9. Support for raster formats supported by the GDAL library
10. Change vector symbology (single, graduated, unique value, and continuous)
11. SVG markers symbology (single, unique value, and graduated)
12. Display raster data such as digital elevation models, aerial photography or landsat imagery
13. Change raster symbology (grayscale, pseudocolor and multiband RGB)
14. Export to Mapserver map file

15. Preliminary digitizing support
16. Map overview
17. Plugins

1.2 Whats New in 0.6

QGIS is still under development. In this release many new features have become available including:

1. GEOS support in the OGR provider to refine selection of features via identify. This improves over the previous method of feature selection which used a simple MBR intersection check.
2. PostGIS editing support in provider
3. Vector dialog redesign to improve usability
4. Improvement in project handling (loading and saving)
5. Scale dependent rendering
6. User option to load layers with out drawing them, thus allowing you to set scale dependency, etc without waiting for the initial draw to complete
7. Interrupt drawing of features by hitting ESC
8. Attribute actions - the ability to run an external program based on the contents of an attribute field in a layer
9. Create new vector layer (shapefile) for editing
10. Windows installer Mac OSX binary
11. New options in the graticule builder plugin
12. Enhancements to the GPS plugin
13. Man page
14. Save delimited text as shapefile
15. Improved Delimited Text plugin, including preview of text file
16. Improved SPIT handling of PostgreSQL reserved words and shapefiles with multiple geometry types
17. Display SQL query used to create a PostGIS layer
18. PostgreSQL query builder
19. Ability to redefine the query used for PostgreSQL layers from the layer properties dialog
20. North arrow, scalebar, and copyright plugins save their state in the project file
21. Datasets with UTF8, Kanjii and CJK filenames now load properly

CHAPTER 2: Getting Started

This chapter gives you a quick overview of running QGIS and examining data available on the QGIS web page.

2.1 Installation

Installation of QGIS is documented in Appendix C. The Installation Guide is distributed with the QGIS source code and is also available at http://qgis.org. Under Windows and Mac OSX, QGIS is available as a standard installer package for these platforms.

2.2 Sample Data

If you do not have any GIS data handy, you can obtain a dataset for Alaska from the QGIS web site at http://qgis.org. The Alaska data set will be used as the basis for many of the examples and screenshots provided in this document.

2.3 Starting QGIS

Assuming that QGIS is installed in the PATH, you can start QGIS by typing: **qgis** at a command prompt or by double clicking on the QGIS application link (or shortcut) on the desktop. Under MS Windows, start QGIS using the Start menu shortcut, and under Mac OSX, double click the icon in your applications folder.

2.3.1 Command Line Options

QGIS supports a number of options when started from the command line. To get a list of the options, enter `qgis --help` on the command line. The usage statement for QGIS is:

```
Usage: /home/gsherman/qgis06_rc/bin/qgis [options] [FILES]
  options:
        [--snapshot filename]   emit snapshot of loaded datasets to given file
        [--lang language]       use language for interface text
        [--project projectfile] load the given QGIS project
        [--help]                this text
  FILES:
    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
```

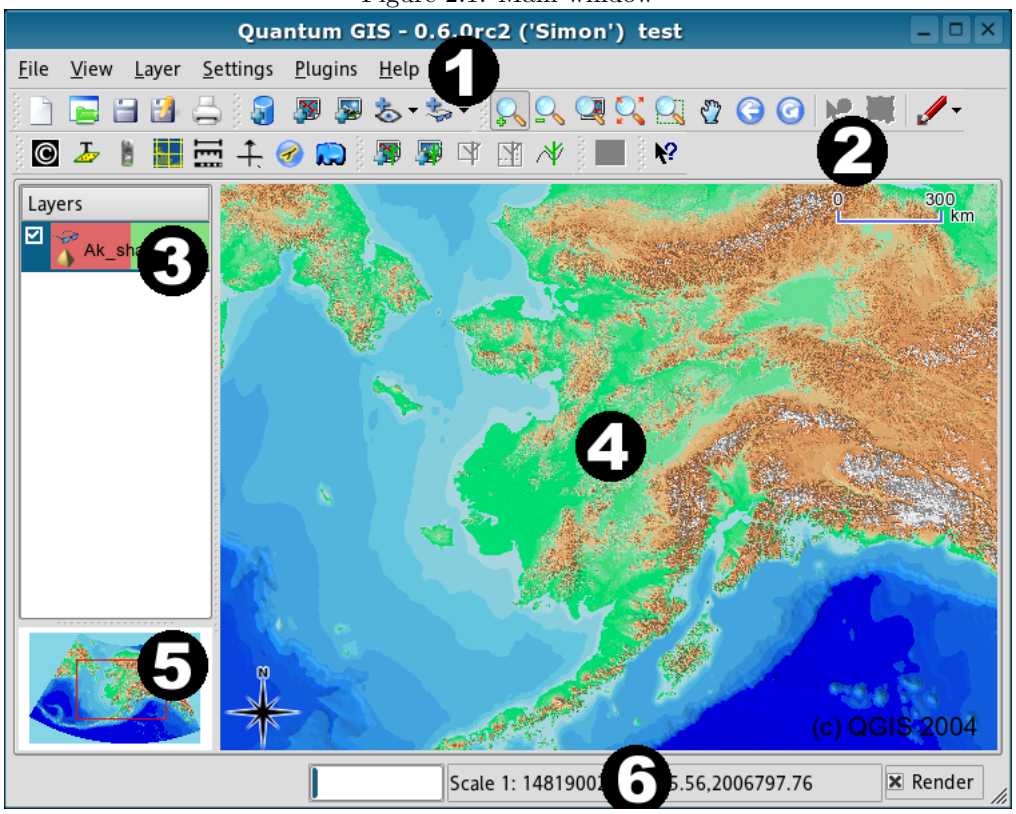
Tip 1 EXAMPLE USING COMMAND LINE ARGUMENTS

You can start QGIS by specifying one or more data files on the command line. For example, assuming you are in your data directory, you could start QGIS with two shapefiles and a raster file set to load on startup using the following command: `qgis ak_shade.tif alaska.shp majrivers.shp`

2.4 The QGIS Main Window

When QGIS starts, you are presented with the main window as shown below.

Figure 2.1: Main window



NOTE - YOUR WINDOW DECORATIONS (TITLE BAR, ETC.) MAY APPEAR DIFFERENT DEPENDING ON YOUR OPERATING SYSTEM AND WINDOW MANAGER The QGIS main window is divided into five areas:

1. The menu bar
2. The tool bar
3. The map legend

4. The map view
5. The map overview
6. The status bar

These six components of the QGIS interface are described in more detail in the following sections

2.4.1 The QGIS menu bar

The menu bar provides access to various QGIS features using a standard windows heirachical menu. The top-level menus and a summary of some of the functions provided are:

- File (project open, save, export image, properties)
- Layer (add, show, hide layers)
- View (zoom, refresh)
- Tools (plugin manager, preferences)
- Plugins (menus added by plugins as they are loaded)
- Help (documentation and web links)

2.4.2 Toolbars

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

2.4.3 The QGIS map legend

The map legend area is used to set the visibility and z-ordering of layers. Z-ordering means that layers listed nearer the top of the legend are drawn over layers listed lower down in the legend. The checkbox in each legend entry can be used to show/hide that layer.

Tip 2 VIEWING THE LAYER MENU

You can display the context menu for any layer in the legend by right-clicking on the layer name. The context menu contains items for working with the layer and viewing its properties.

Each legend entry can show the following mini icons:

 This is a raster layer that has pyramids built for it to improve rendering efficiency (see Section 4.4.4).

 This is a raster that has no pyramid layers (see Section 4.4.4).

 This layer is shown in the overview map area as well as in the main map window.

 This is a vector layer that is currently enabled for editing.

2.4.4 The QGIS map view

This is the 'business end' of QGIS - maps are displayed in this area! The map displayed in this window will depend on the vector and raster layers you have chosen to load (see sections that follow for more info on this). The map view can be panned (shifting to focus of the map display to another region), zoomed in and out, and supports various other actions as described in the toolbar description above. The map view and the legend are tightly bound to each other - the maps in view reflect changes you make in the legend area.

Tip 3 Zooming the Map with the Mouse Wheel

You can use the mouse wheel to zoom in and out on the map. 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.

2.4.5 The QGIS map overview

The map overview area provides a full extent view of layers added to it. Within the view is a rectangle showing the current map extent. This allows you to quickly determine which area of the map you are currently viewing. Note that labels are not rendered to the map overview even if the layers in the map overview have been set up for labelling.

2.4.6 The QGIS map status bar

The status bar shows you your current position in map coordinate (e.g. meters or decimal degress) as the mouse pointer is moved accross the map view. The status bar also shows the view extents of the map view as you pand and zoom in and out. A progress bar in the status bar shows progress of rendering as each layer is drawn to the map view. In some cases such as the gathering of statistics in raster layers, the progress bar will be used to show the status of lengthy operations. At the end of the status bar is a small checkbox which can be used to temporarily prevent layers being rendered to the map view)see Section 2.5 below.

2.5 Rendering

By default, QGIS renders all visible layers whenever the map canvas must be refreshed. The events that trigger a refresh of the map canvas include:

- Adding a layer
- Panning or zooming
- Resizing the QGIS window
- Changing the visibility of a layer or layers

QGIS allows you to control the rendering process in a number of ways.

2.5.1 Scale Dependent Rendering

Scale dependent rendering allows you to specify the minimum and maximum scales at which a layer will be visible. To set scale dependency rendering, open the properties dialog by double-clicking on the layer in the legend. On the *General* tab, set the minimum and maximum scale values and then click on the *Use scale dependent rendering* checkbox.

You can determine the scale values by first zooming to the level you want to use an noting the scale value in the QGIS status bar.

2.5.2 Controlling Map Rendering

Map rendering can be controlled in the following ways:

1. Stopping rendering during drawing of the map canvas
2. Temporarily suspending rendering
3. Setting an option to control the visibility of layers when they are added

2.5.3 Stopping Rendering

To stop the map drawing, press the ESC key. This will halt the refresh of the map canvas and leave the map partially drawn. It may take a bit of time between pressing ESC and the time the map drawing is halted.

2.5.4 Suspending Rendering

To suspend rendering, click the *Render* checkbox in the lower right corner of the statusbar. When the *Render* box is not checked, QGIS does not redraw the canvas in response to any of the events described in Section **??**. Examples of when you might want to suspend rendering include:

- Add many layers and symbolize them prior to drawing
- Add one or more large layers and set scale dependency before drawing
- Add one or more large layers and zoom to a specific view before drawing
- Any combination of the above

Checking the *Render* box enables rendering and causes and immediate refresh of the map canvas.

2.5.5 Setting Layer Add Option

You can set an option to always load new layers without drawing them. This means the layer will be added to the map, but its visibility checkbox in the legend will be unchecked by default. To set this option, choose *Preferences* from the *Settings* menu and click on the *Rendering* tab. Check the *New layers added to the map are not displayed* checkbox. Any layer added to the map will be off (invisible) by default.

2.5.6 Updating the Map Display During Rendering

You can set an option to update the map display as features are drawn. By default, QGIS does not display any features for a layer until the entire layer has been rendered. To update the display as features are read from the datastore, choose *Preferences* from the *Settings* menu and click on the *Rendering* tab. Set the feature count to an appropriate value to update the display during rendering. Setting a value of 0 disables update during drawing (this is the default). Setting a value too low will result in poor performance as the map canvas is continually updated during the reading of the features. A suggested value to start with is 500.

CHAPTER 3: Working with Vector Data

QGIS supports vector data in a number of formats, including shapefiles, MapInfo mif, and PostGIS layers in a PostgreSQL database. Support for additional data types is provided by plugins, for example delimited text.

This section describes how to work with two common formats: shapefiles and PostGIS layers. Many of the features available in QGIS work the same regardless of the vector data source. This is by design and includes the identify, select, labeling, and attributes functions.

3.1 Shapefiles

Shapefile support is provided by a library of functions (OGR http://www.remotesensing.org/gdal/ogr). See Appendix A.1 for a list of supported formats.

A shapefile actually consists of a minimum of three files:

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

The technical specification for the shapefile format can be found at
http://www.esri.com/software/opengis/openpdf.html.

3.1.1 Loading a Shapefile

To load a shapefile, start QGIS and click on the *Add a vector layer* toolbar bar button. This same tool can be used to load any of the formats supported by the OGR library.

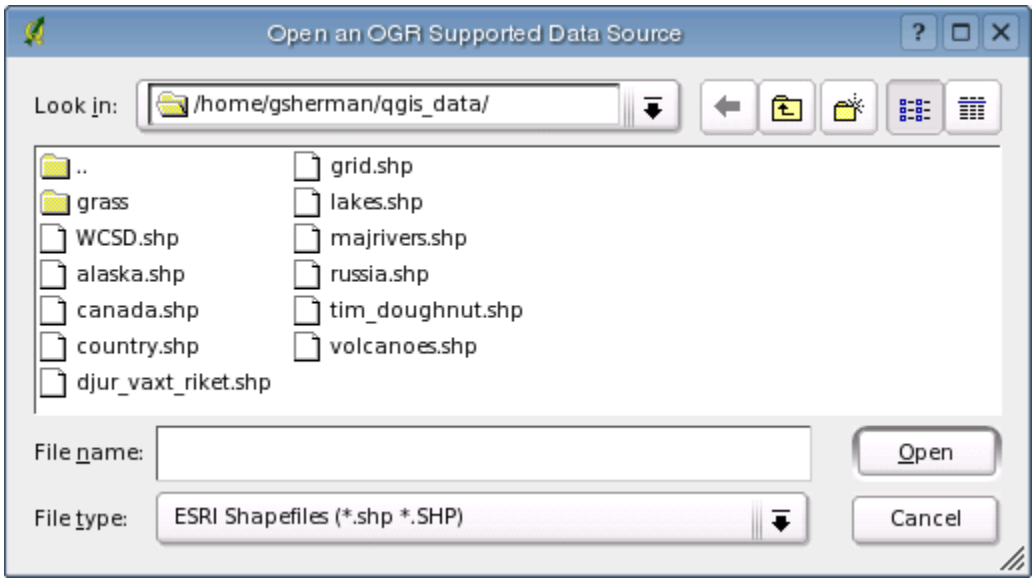
Clicking on the tool brings up a standard open file dialog (Figure 3.1) which allows you to navigate the file system and load a shapefile (or other supported data source). Selecting a shapefile from the list and clicking Ok loads it into QGIS. Figure 3.2 shows QGIS after loading the country.shp file.

Tip 4 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.

Once loaded, you can zoom around the shapefile using the map navigation tools. To change the symbology of a layer, open the layer properties dialog by double clicking on the layer name or by right-clicking on the name in the legend and choosing *Properties* from the popup menu. See Section 3.3.1 for more information on setting symbology of vector layers.

Figure 3.1: Open OGR Data Source Dialog



3.1.2 Loading a MapInfo Layer

To load a MapInfo layer, click on the *Add a vector layer* toolbar bar button and change the file type filter to *MapInfo (*.mif *.tab *.MIF *.TAB)* and select the layer you want to load.

3.1.3 Loading an ArcInfo Coverage

Loading an an ArcInfo coverage is done using the same method as with a shapefiles and MapInfo layers. Click on the *Add a vector layer* toolbar button to open the layer dialog. Navigate to the coverage directory and select one of the following files (if present in your coverage)

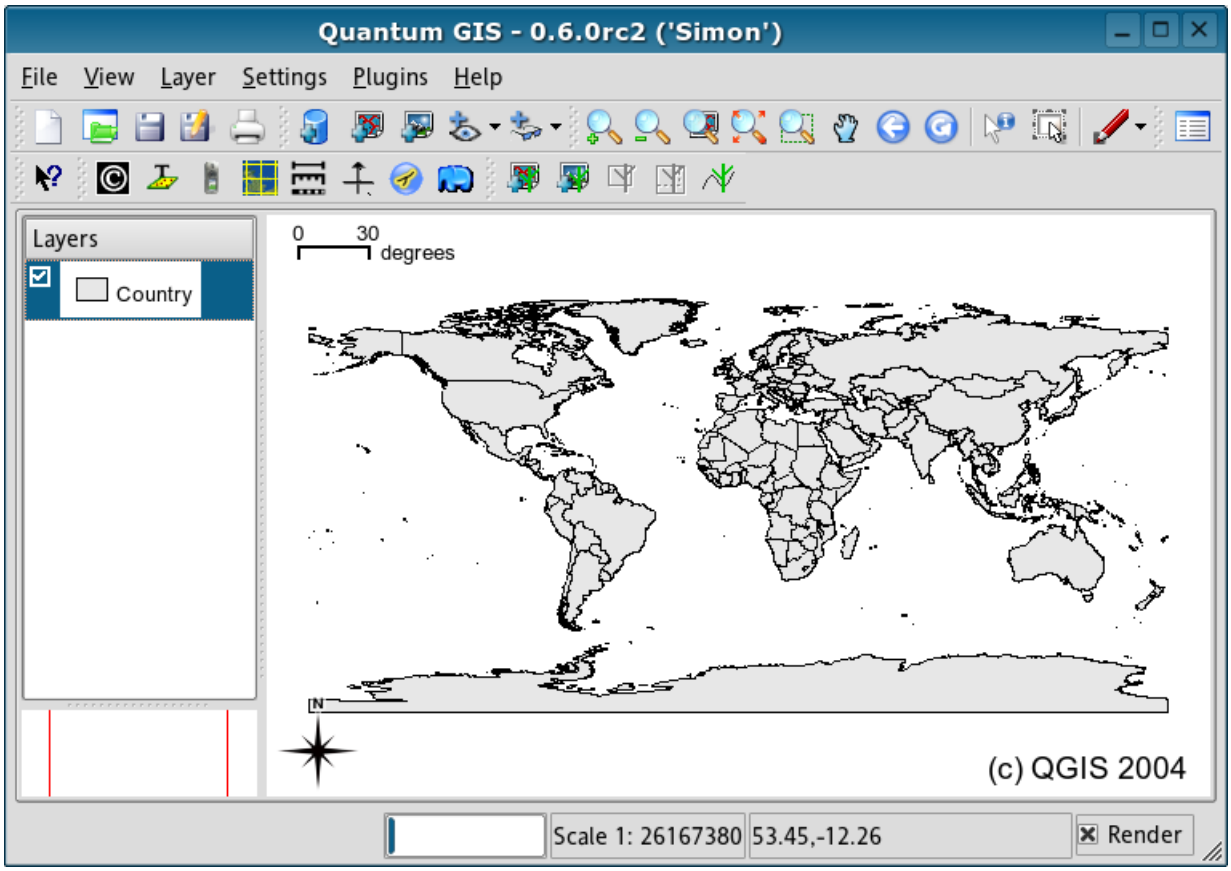
1. .lab - to load a label layer (polygon labels, or standing points
2. .cnt - to load a polygon centroid layer
3. .arc - to load an arc (line) layer
4. .pal - to load a polygon layer

3.2 PostGIS Layers

PostGIS layers are stored in a PostgreSQL database. The advantage of PostGIS is the spatial indexing, filtering, and query capability. Using PostGIS, vector functions such as select and identify work more accurately than with OGR layers in QGIS. To use PostGIS layers you must:

1. Create a stored connection in QGIS to the PostgreSQL database (if one is not already defined)
2. Connect to the database
3. Select the layer to add to the map

Figure 3.2: QGIS with the countries Shapefile Loaded



4. Optionally provide a SQL *where* clause to define which features to load from the layer
5. Load the layer

3.2.1 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 a PostGIS Layer* toolbar button. 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 shown in Table 3.1. Once the parameters have been filled in, you can test the connection by clicking on the *Test Connection* button. To save the password with the connection information, check the *Save Password* option.

Tip 5 QGIS User Settings and Security

Your customized settings for QGIS are stored based on the operating system. On Linux/Unix, the settings are stored in your home directory in .qt/qgisrc. On Windows, the settings are stored in the registry. Depending on your computing environment, storing passwords in your QGIS settings may be a security risk.

Table 3.1: PostGIS Connection Parameters

Name	A name for this connection. Can be the same as *Database*
Host	Name of the database host. This must be a resolvable host name the same as would be used to open a telnet connection or ping the host
Database	Name of the database
Port	Port number the PostgreSQL database server listens on. The default port is 5432.
Username	User name used to login to the database
Password	password used with *Username* to connect to the database

3.2.2 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 3.2.4 for a discussion on importing data into the database.

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

1. If the PostGIS layer dialog is not already open, click on the *Add a PostGIS Layer* toolbar button
2. Choose the connection from the drop-down list and click *Connect*
3. Find the layer you wish to add in the list of available layers
4. Select it by clicking on it. You can select multiple layers by holding down the shift key while clicking. See Section 3.2.3 for information on using the PostgreSQL Query Builder to further define the layer.
5. Click on the *Add* button to add the layer to the map

3.2.3 Using the Query Builder

The PostgreSQL Query Builder allows you to define a subset of a table and add it as a layer in QGIS. For example, if you have a towns layer with a population field you could select only larger towns by entering *population > 100000* in the SQL box of the query builder. Figure **??** shows an example of the query builder populated with data from a layer in PostgreSQL.

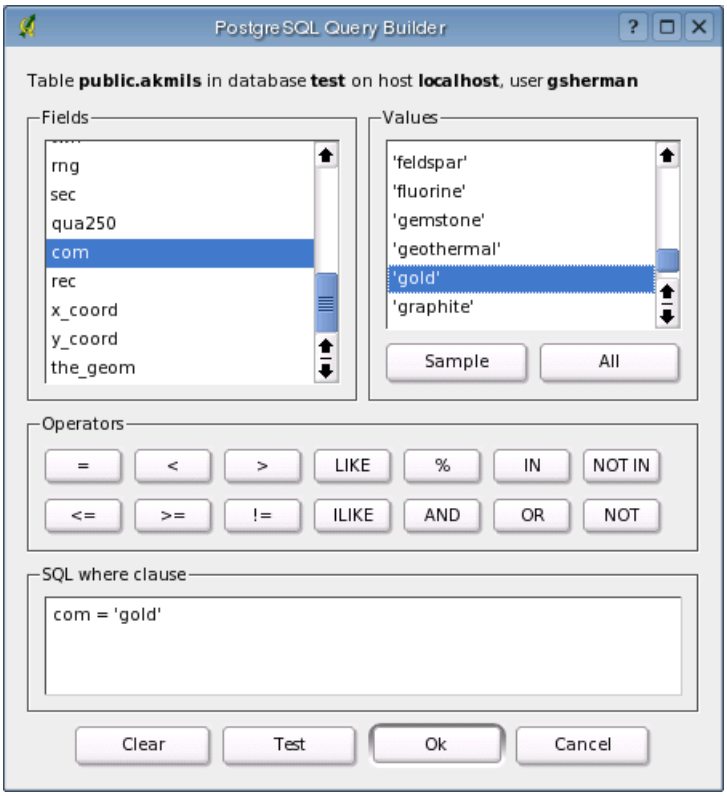
The query builder lists the layer's database fields in the list box on the left. You can get a sample of the data contained in the highlighted field by clicking on the *Sample* button. This retrieves the first 25 distinct values for the field from the database. To get a list of all possible values for a field, click on the *All* button. To add a selected field or value to the query, double-click on it. You can use the various buttons to construct the query or you can just type it into the SQL box.

To test a query, click on the *Test* button. This will return a count of the number of records that will be included in the layer. When satisfied with the query, click *Ok*. The SQL for the where clause will be shown in the SQL column of the layer list.

Tip 6 Changing the Layer Definition

You can change the layer defintion after it is loaded by altering the SQL query used to define the layer. To do this, open the vector layer properties dialog by double-clicking on the layer in the legend and click on the *Query Builder* button on the *General* tab. See Section 3.3 for more information.

Figure 3.3: PostgreSQL Query Builder



3.2.4 Importing Data into PostgreSQL

Data can be imported into PostgreSQL using a number of methods. PostGIS includes a utility called shp2pgsql that can be used to import shapefiles into a PostGIS enabled database.

QGIS comes with a plugin named SPIT (Shapefile to PostGIS Import Tool). SPIT can be used to load mutliple shapefiles at one time and includes support for schemas. To use SPIT, open the Plugin Manager from the Tools menu and load the plugin by checking the box next to the SPIT plugin and click Ok. The SPIT icon will be added to the plugin toolbar.

To import a shapefile, click on the SPIT tool in the toolbar to open the dialog. You can add one or more files to the queue by clicking on the *Add* button. To process the files, click on the Import button. The progress of the import as well as any errors/warnings will be displayed as each shapefile is processed.

Tip 7 Importing Shapefiles Containing PostgreSQL Reserved Words

If a shapefile is added to the queue containing fields that are reserved words in the PostgreSQL database a dialog will popup showing the status of each field. You can edit the field names prior to import and change any that are reserved words (or change any other field names as desired). Attempting to import a shapefile with reserved words as field names will likely fail.

3.3 The Vector Properties Dialog

The vector properties dialog provides information about a 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 - either by hand editing the SQL on the *General* tab, or by invoking the query builder dialog on the *General* tab. To access the properties dialog, double-click on a layer in the legend or right-click on the layer and select Properties from the popup menu.

3.3.1 Vector Symbology

QGIS supports a number of symbology renderers to control how vector features are displayed. Currently the following renderers are available:

Single symbol - a single style is applied to every object in the layer.
Graduated symbol - objects within the layer are displayed with different symbols classified by the values of a particular field.
Continuous colour - objects within the layer are displayed with a spread of colours classified by the numerical values within a specified field.
Unique value - objects are classified by the unique values within a specified field with each value having a different symbol.

For layers containing point features, additional renderers are available that use SVG icons:

Single marker - a single specified icon is used for every point within the layer.
Graduated marker - points within the layer are displayed with different icons classified by values within a particular field.
Unique value marker - points are classified by unique values within a specified field with each value having a different icon.

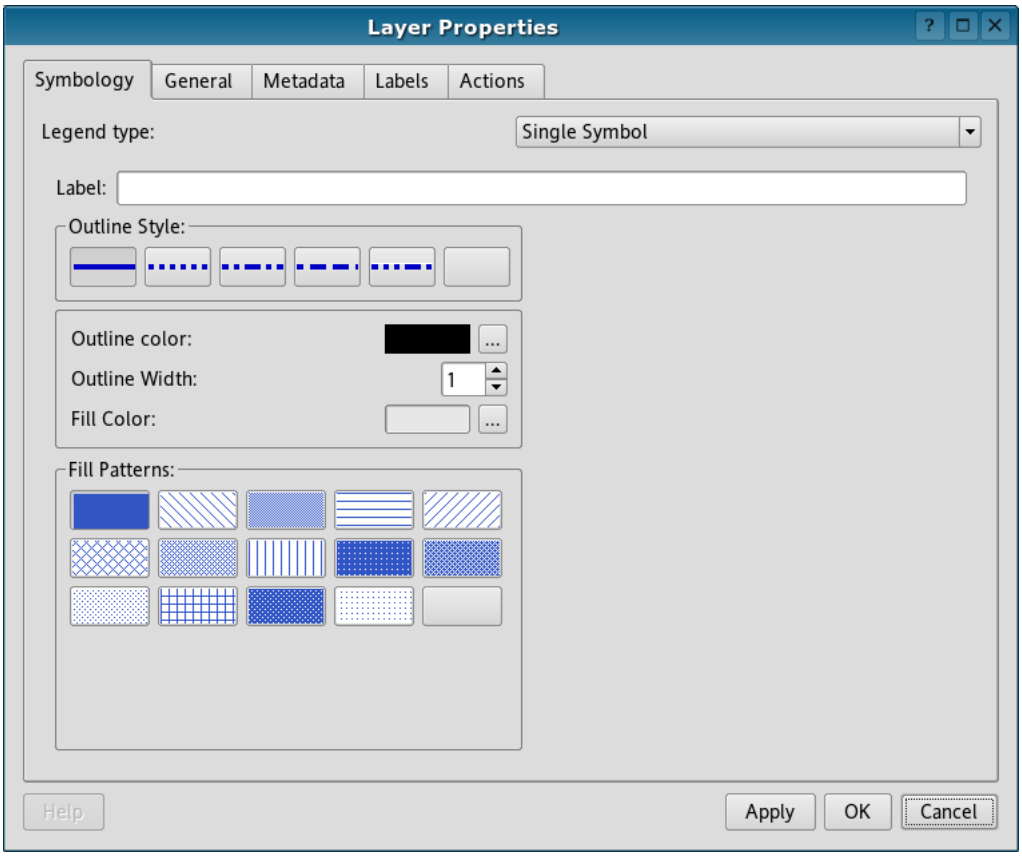
To change the symbology for a layer, simply double click on its legend entry and the vector layer properties dialog will be shown.

3.4 Attribute Actions

A new tool in QGIS 0.6 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. An example is performing a search based on an attribute value. This concept is used in the following discussion.

Figure 3.4: Vector Layer Properties Dialog



3.4.1 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* tab. 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 preceeded by a backslash.

Two example actions are shown below:

1. `konqueror http://www.google.com/search?q=%nam`
2. `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 provided the full path. To be sure, 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 envoked.

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.

3.4.2 Using Actions

Actions can be invoked from either the *Identify Results* dialog or the *Attribute table* dialog. To invoke an action, right click on the record and choose the action from the popup menu. Actions are listed in the popup menu by the name you assigned when defining the actions. 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* 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 Gnu/Linux and perhaps Mac OSX). The layer in question has fields for a species name (taxon_name), latitude (lat) and longitude (long). I would like to be able to make a spatial selection of a localites 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 localties 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
```

3.5 Editing

As of version 0.6 QGIS supports basic capabilities for editing spatial data. Before reading any further you should note that at this stage editing is still experimental. Before performing any edits, always make a backup of the dataset you are about to edit.

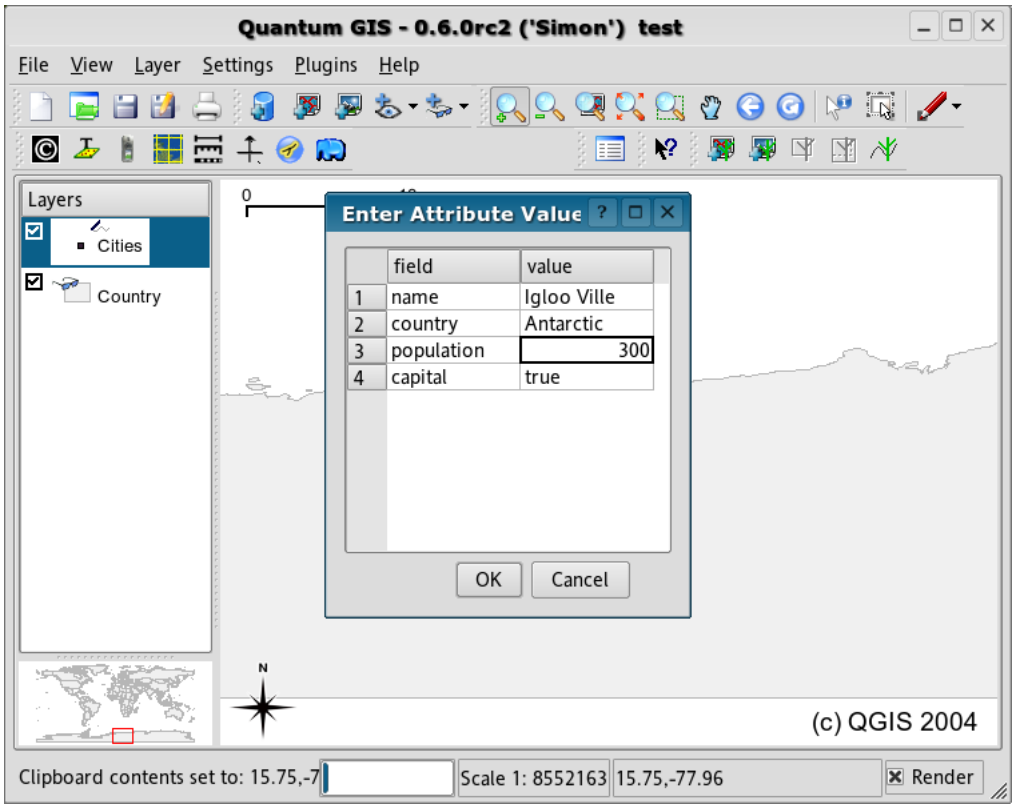
Note - the procedure for editing GRASS layers is different - see Section 5.4 for details.

3.5.1 Editing an Existing Layer

If you wish to edit an existing layer, choose *Start Editing* from the context menu after right clicking on the legend entry for that layer. Remember to backup your data before starting! Once the layer is in edit mode you will see a small  icon to remind you.

Now that the layer is editable, you can use the *Capture Points* icon (or similiar icon for line and polygon layers) on the toolbar to put the qgis cursor into digitising mode. If you are capturing a point feature simply use the pan and zoom tools to navigate to the area of interest, then click the *Capture Points* tool and click on the map area to create the new point feature. A window will appear allowing you to set the attributes. Figure 3.5 shows setting attributes for a fictitious new city in the Antarctic.

Figure 3.5: Vector Digitizing Atributes Capture Dialog



In its current implementation the attributes dialog box does not check that the data matches the type expected, so make sure of this before pressing *Ok*.

To delete a feature, select it using the selection tool and choose *Delete selection* from the editing tools.

Once you have finished adding features, choose *Stop Editing* from the layer's context menu. Choosing *Yes* will save the changes to disk, while choosing *No* at this point will cause them to be discarded.

3.5.2 Creating a New Layer

In its current incarnation, the editing of attributes associated with spatial features is extremely basic and subsequently some of the procedures explained below involve some extra steps which will later no longer be necessary as the editing tools mature.

To create a new layer, choose *New Vector Layer* from the *Layer* menu. You will be prompted for the layer type - either point, line or polygon. Note QGIS that does not yet support creation of 2.5D features (i.e.

features with X,Y,Z coordinates) or measure features. At this time, only shapefiles can be created. In a future version of QGIS, creation of any OGR or PostgreSQL layer type will be supported.

Having selected the layer type, you will be prompted for the filename for your new layer. QGIS will automatically add a .shp extension to the name you specify. Once the layer has been created, you edit it in the same way as described in Section 3.5.1 above. When the shapefile is created, only one attribute field called "dummy" is added. In order to add additional fields you need to take a few extra steps as referred to earlier [1].

Immediately after creating the new shape file, remove it from the legend. Then in a bash or MSDOS shell, use the shapelib[2] 'dbfcreate' tool to create a new copy of the associated .dbf file. Here is an example:

```
# dbfcreate timtest.dbf -s taxon_name 255, -n lat 3 5, -n long 3 5, -s notes 255
# dbfdump -h timtest.dbf
Field 0: Type=String, Title='taxon_name', Width=255, Decimals=0
Field 1: Type=Double, Title='lat', Width=3, Decimals=5
Field 2: Type=Double, Title='long', Width=3, Decimals=5
Field 3: Type=String, Title='notes', Width=255, Decimals=0
```

Having modified the .dbf file to your needs you can now go ahead and digitise features using the same procedure as described in Section 3.5.1.

[1] These limitations will be removed in the next version of QGIS.
[2] shapelib is available from http://shapelib.maptools.org/

CHAPTER 4: Working with Raster Data

QGIS supports a number of raster data formats. This section describes how to work with raster data in QGIS.

4.1 What is raster data?

indexrasters!definition 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.

In GIS, a raster layer would have georeferencing data associated with it which will allow it to be positioned correctly in the map display to allow other vector and raster data to be overlayed with it. QGIS makes use of georeferenced rasters to properly display the data.

4.2 Raster formats supported in QGIS

QGIS supports a number of different raster formats. Currently tested formats include:

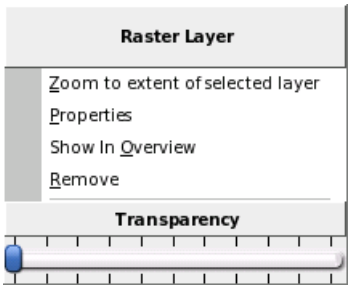
- Arc/Info Binary Grid
- Arc/Info ASCII Grid
- Grass Raster
- GeoTIFF
- Spatial Data Transfer Standard Grids (with some limitations)
- USGS ASCII DEM
- Erdas Imagine

Because the raster implmentation in QGIS is based on the GDAL library, other raster formats implemented in GDAL are also likely to work, but have not yet been tested. See Appendix A.2 for more details.

4.3 Loading raster data in QGIS

Raster layers are loaded either by clicking on the Load Raster icon or by selecting the View->Add Raster Layer menu option. More than one layer can be loaded at the same time by holding down the Control key and clicking on multiple items in the file dialog.

4.4 Raster Properties



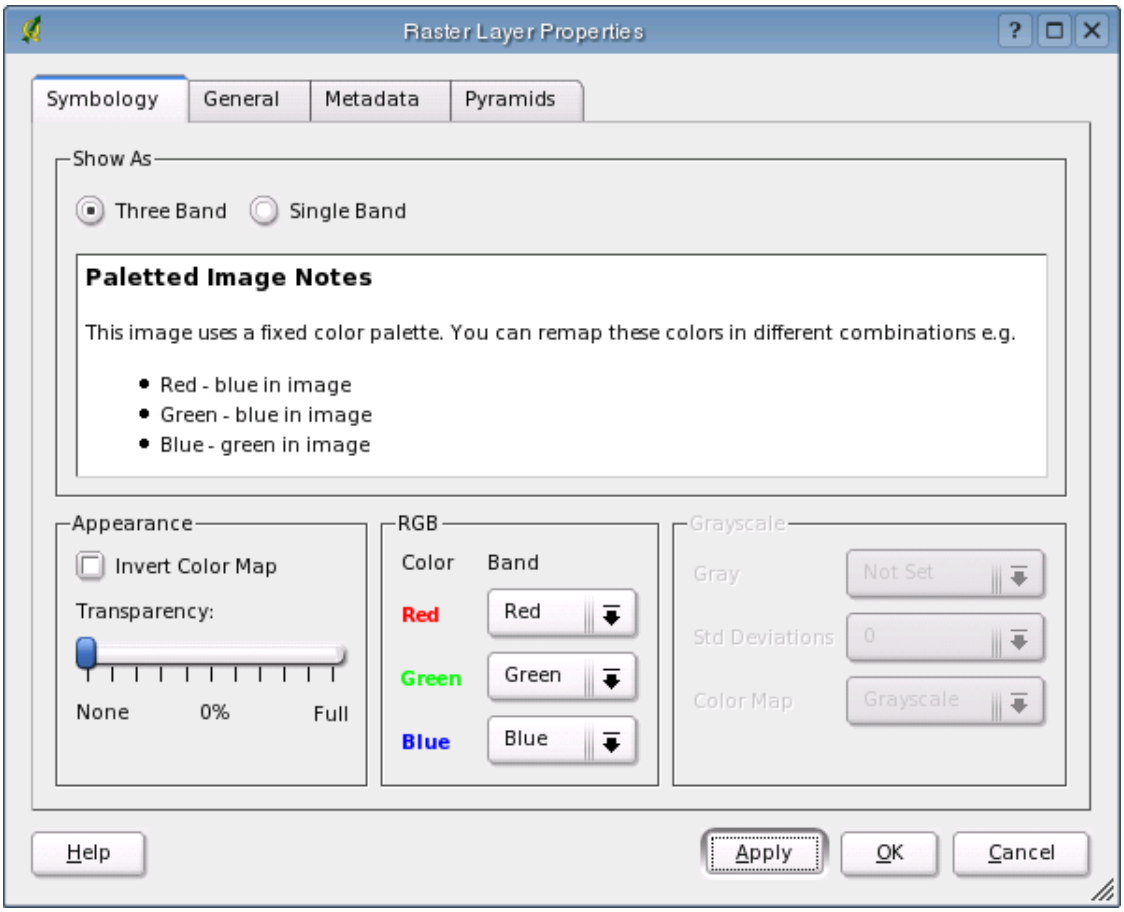
To view and set the properties for a raster layer, right click on the layer name. This displays the raster layer context menu that includes a number of items that allow you to:

- Zoom to the full extent of the raster
- Show the raster in the map overview window
- Open the properties dialog (of course)
- Remove the layer from the map
- Set the transparency using a slider control

Choose *Properties* from the context menu to open the raster properties dialog for the layer.

Figure 4.1 shows the properties dialog. There are four tabs on the dialog: *Symbology*, *General*, *Metadata*, and *Pyramids*.

Figure 4.1: Raster Layers Properties Dialog



4.4.1 Symbology Tab

QGIS supports three forms of raster layers:

- Single Band Grayscale Rasters
- Palette Based RGB Rasters
- Multiband RGB Rasters

From these three basic layer types, eight forms of symbolised raster display can be used:

1. Single Band Grayscale
2. Single Band Pseudocolor
3. Paletted Grayscale (where only the red, green or blue component of the image is displayed)
4. Paletted Pseudocolor (where only the red, green or blue component of the image is displayed, but using a pseudocolor algorithm)
5. Paletted RGB
6. Multiband Grayscale (using only one of the bands to display the image)
7. Mulitiband Pseudocolor (using only one of the bands shown in pseudocolor)
8. Multiband RGB (using any combination of three bands)

QGIS can invert the colors in a given layer so that light colors become dark (and dark colors become light). Use the *Invert Color Map* checkbox to enable / disable this behavior.

QGIS has the ability to display each raster layer at varying transparency levels. Use the transparency slider to indicate to what extent the underlying layers (if any) should be visible though the current raster layer. The transparency can also be set using the transparency slider in the layer context menu which is accessible by right-clicking on the layer in the legend.

QGIS can restrict the data displayed to only show cells whose values are within a given number of standard deviations of the mean for the layer. 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. This option is only available for pseudocolor images.

4.4.2 General Tab

The General tab displays basic information about the selected raster, including the layer source and display name in the legend (which can be modified). This tab also shows a thumbnail of the layer, its legend symbol, and the palette.

4.4.3 Metadata Tab

The Metadata tab displays a wealth of information about the raster layer, including statistics about each band in the current raster layer. Statistics are gathered on a 'need to know' basis, so it may well be that a given layers statistics have not yet been collected.

Tip 8 Gathering Raster Statistics

To gather statistics for a layer, select pseudocolor rendering and click the *Apply* button. Gathering statistics for a layer can be time consuming. Please be patient while QGIS examines your data!

4.4.4 Pyramids Tab

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.

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.

CHAPTER 5: GRASS

The GRASS plugin adds the following features to QGIS:

- Add GRASS vector layers
- Add GRASS raster layers
- Vector layers digitizing
- Changing of the GRASS region

5.1 Starting QGIS with GRASS

When using the GRASS plugin, QGIS can be started in two ways: from the GRASS shell or from a regular shell.

5.1.1 From GRASS shell

If QGIS is started from the GRASS shell (GRASS started by grass57 command), no additional settings are required.

5.1.2 Outside GRASS shell

If QGIS is not started from the GRASS shell, the environment variables must be properly set before starting QGIS.

The path to GRASS libraries must be added to LD_LIBRARY_PATH environment variable. For example (in bash):

```
export LD_LIBRARY_PATH=/usr1/grass57/dist.i686-pc-linux-gnu/lib:$LD_LIBRARY_PATH
```

The GISBASE environment variable must be set to the full path of the directory where GRASS is installed (the same as used for –with-grass= option). For example (in bash):

```
export GISBASE=/usr1/grass57/dist.i686-pc-linux-gnu
```

5.2 Loading GRASS Data

With the GRASS plugin loaded, you can load a vector or raster layer using the appropriate button on the toolbar.

Tip 9 GRASS Data Loading

If you have problems loading data or QGIS terminates abnormally, check to make sure you have started GRASS properly as described in Section 5.1.

5.3 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 without gaps. An area is identified (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.

It is possible to store more 'layers' in one vector dataset. For example, fields, forests and lakes can be stored in one vector. Adjacent forest and lake can share the same boundary, but they have separate attribute tables. It is also possible to attach attributes to boundaries. For example, the boundary between lake and forest is a road with different attribute table.

The 'layer' of the feature is defined by 'field' (sorry for this name). 'Field' is the number which defines 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 are stored in external database tables, for example DBF, PostgreSQL, etc.

Attributes in database tables are linked to geometry elements using 'category'. 'Category' (key, ID) is an integer attached to geometry primitives, and it is used as the link to one column in the database table.

Tip 10 Learning the GRASS Vector Model

The best way to learn the GRASS vector model and its capabilities is to download the demo mapset from `http://mpa.itc.it/radim/g51/g51test-12-multi.tar.gz`. Extract the mapset, add all layers from vector 'multi' to QGIS, and query attributes. Finaly start editing of vector 'multi', to see how those layers are stored.

5.4 Digitizing and Editing Tools

The digitizing tools for GRASS vector layers are accessed using the *Edit GRASS Vector Layer* tool 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. In this release, the vector must exist prior to beginning to edit. The ability to create a new "empty" layer will be added in a future version. Figure 5.1 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.

5.4.1 Toolbar

Table 5.1 lists the digitizing tools provided by the GRASS plugin. These correspond to the tool buttons in the toolbar(s) across the top of the dialog.

Figure 5.1: GRASS Edit Dialog

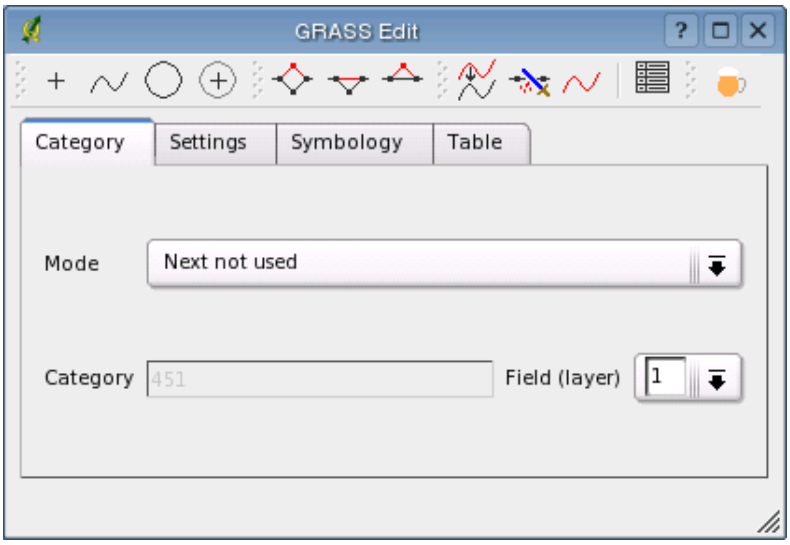


Table 5.1: GRASS Digitizing Tools

Tool	**Purpose**
New Point	digitize new point
New Line	digitize new line (finish by selecting new tool)
New Boundary	digitize new boundary (finish by selecting new tool)
New Centroid	digitize new centroid (label existing area)
Move vertex	select one vertex of existing line or boundary and identify new position
Add vertex	add a new vertex to existing line
Delete vertex	delete one vertex from existing line (confirm selected vertex by another click)
Move line	select existing line and click on new position
Split line	split an existing line to 2 parts
Delete line	delete existing line (confirm selected line by another click)
Edit attributes	edit attributes of existing element (note that one element can represent more features, see above)
Mug	close digitizing session

5.4.2 Category Tab

This tab allows you to set the way in which the category will be assigned to each new feature and/or assign a category to a feature.

- Mode: what category should be attached to geometry
  - Next not used - next category not yet used in vector
  - Manual entry - define the category in 'Category entry'
  - No category - digitize geometry without category
- Category - a number (ID) attached to digitized feature
- Field - feature (attribute table) identification

5.4.3 Settings Tab

This tab allow you to set the snapping in screen pixels. This is the threshold in pixels in which new points or line ends are snapped to existing nodes. This helps prevent gaps or dangles between boundaries

5.4.4 Symbology Tab

This tab allows you to view and set symbology for various geometry types and their topological status (e.g. closed / opened boundary).

5.4.5 Table

This tab provides the means to view, create, or modify the database table for a given field.

Tip 11 GRASS Edit Permissions

You must be the owner of the GRASS mapset you want to edit. It is impossible to edit vectors in mapsets which are not yours, even if you have write permissions.

5.4.6 Region Tool

The current region (window) in GRASS is very important for all raster modules. All new created rasters have the extension and resolution of the current region, regardless their original region. The region is stored in $LOCATION/$MAPSET/WIND file, and it defines north, south, east, west, number of columns, number of rows, horizontal and vertical resolution.

It is possible to switch on/off the grass region in QGIS canvas using the *Display Current GRASS Region* button.

With the *Edit Current GRASS Region* you can open a tool in which you can change the current region and symbology of the rectangle on the QGIS Canvas. When the tool is running, it is also possible to select a new region interactively on the QGIS canvas.

Both tools are available only if QGIS was started from a GRASS shell or if the GISRC enviroment variable pointing to a valid GISRC file was set (i.e. only if you are running GRASS within your mapset).

CHAPTER 6: Using Plugins

6.1 An Introduction to Using Plugins

QGIS has been designed with a plugin architecture. This allows new features/functions to be added to the application. Many of the features in QGIS are actually implemented as plugins.

There are two types of plugins in QGIS: core and user-contributed. A core plugin is maintained by the QGIS development team and is part of every QGIS distribtution. A user-contributed plugin is an external plugin that is maintained by the individual author. The QGIS Community site (http://community.qgis.org) serves as the repository for user contributed plugins.

6.1.1 Finding and Installing a Plugin

When you install QGIS, all of the core plugins are included (these are described below). Additional user-contributed plugins may be available on the QGIS Community site. To see what user-contributed plugins are available, see the plugins page on the Community site (http://community.qgis.org/plugins).

Typically user-contributed plugins are distributed in source form and require compiling. For instructions on building and installing a user-contributed plugin, see the documentation included with the plugin.

6.1.2 Managing Plugins

Managing plugins consists of loading or unloading them from QGIS. Loaded plugins are "remembered" when you exit the application and restored the next time you run QGIS.

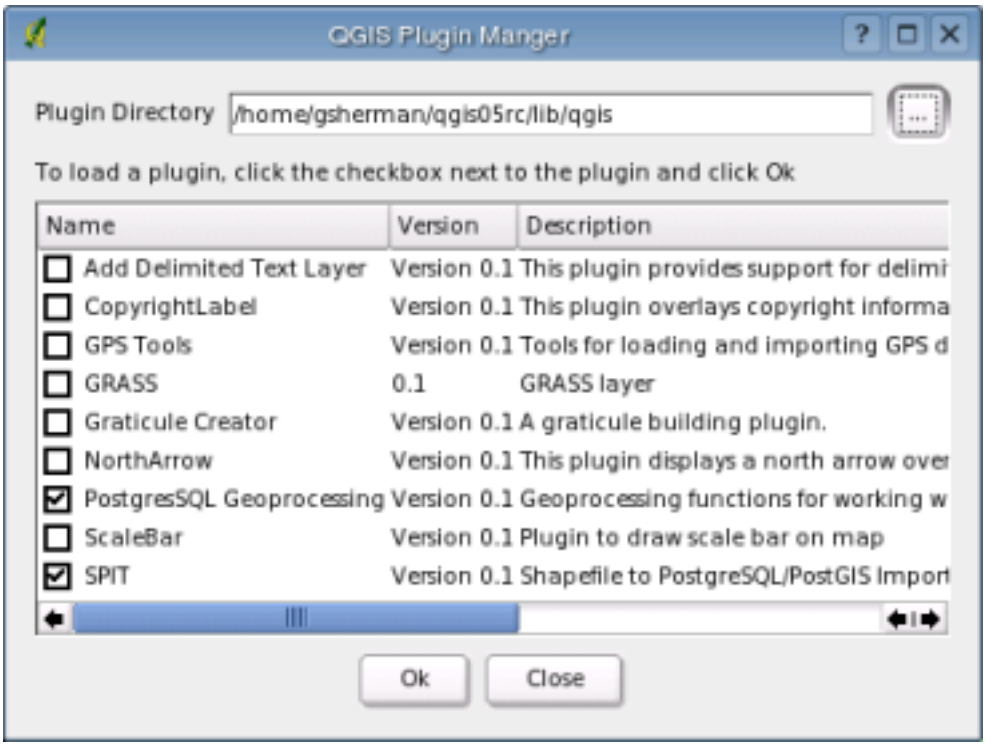
To manage plugins, open the *Plugin Manager* from the *Tools* menu. The Plugin Manager displays all the available plugins and their status (loaded or unloaded). Figure 6.1 shows the Plugin Manager dialog.

Typically all QGIS plugins are installed in the same location. This location is shown in the Plugin Directory text field. You can tell QGIS to load plugins from another location by specifying a different directory.

Tip 12 CRASHING PLUGINS

If you find that QGIS crashes on startup, a plugin may be at fault. You can stop all plugins from loading by editing your .qt/qgisrc file in your home directory on Linux/Unix (Windows users will have to edit the registry). On Linux/Unix, open the qgisrc file in a text editor and find the [Plugins] section. Set all the plugin values to false to prevent them from loading. For example, to prevent the Delimited text plugin from loading, the entry in qgisrc should look like this: `Add Delimited Text Layer=false`.Do this for each plugin in the [Plugins] section. You can then start QGIS and add the plugins one at a time from the Plugin Manger to determine which is causing the problem.

Figure 6.1: Plugin Manager



6.1.3 Data Providers

Data Providers are "special" plugins that provides access to a data store. By default, QGIS supports PostGIS layers and disk-based data stores supported by the OGR library (Appendix A.1). A Data Provider plugin extends the ability of QGIS to use other data sources.

Data Provider plugins are registered automatically by QGIS at startup. They are not managed by the Plugin Manager but are used behind the scenes when a corresponding data type is added as a layer in QGIS.

6.1.4 Core Plugins

QGIS currently contains 9 core plugins that can be loaded using the Plugin Manager. Table 6.1 lists each of the core plugins along with a description of their purpose. Figure 6.2 shows the icon for each plugin in the Plugin toolbar (the number corresponds to the Item in Table 6.1. Note the GRASS plugin is not included below because it installs its own toolbar (see Section 5 for a discussion of available features in GRASS plugin).

Tip 13 PLUGINS SETTINGS SAVED TO PROJECT

When you save a .qgs project, any changes you have made to NorthArrow, ScaleBar and Copyright plugins will be saved in the project and restored next time you load it.

Table 6.1: QGIS Core Plugins

Item	Plugin	Description
1	Copyright Label	Display a copyright label on the map canvas
2	Delimited Text	Load a delimited text file containing x,y coordinates as a point layer
3	GPS Tools	Load and display GPS data
4	Graticule Creator	Create a latitude/longitude grid and save as a shapefile
5	Scalebar	Add a scalebar to the map canvas
6	North Arrow	Add a north arrow to the map canvas
7	PostgreSQL Geoprocessing	Buffer a PostGIS layer
8	SPIT	Shapefile to PostGIS Import Tool - import shapefiles into PostgreSQL

Figure 6.2: Plugin Toolbar and Icons

6.2 Using the GPS Plugin

6.2.1 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. It 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 receivers 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.

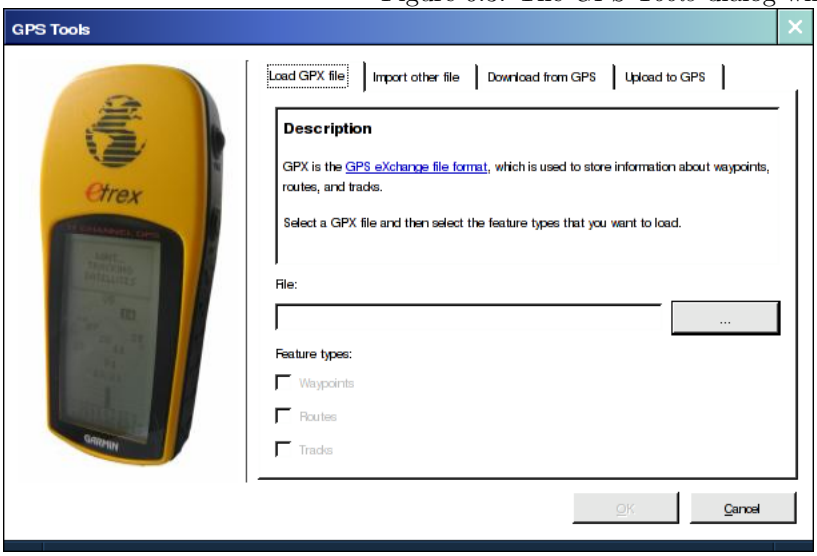
6.2.2 Loading GPS data from a file

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 need to use the *GPS Tools* plugin. When this plugin is loaded a button with a small handheld GPS device will show up in the toolbar (the device looks a bit like a mobile phone). Clicking on this button will open the *GPS Tools* dialog (see figure 6.3).

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.

Figure 6.3: The *GPS Tools* dialog window



6.2.3 GPSBabel

Since QGIS uses GPX files you need a way to convert other GPS file formats to GPX. This can be done for many formats using the free program GPSBabel, which is available at http://www.gpsbabel.org. This program can also transfer GPS data between your computer and a GPS device. QGIS uses GPSBabel to do these things, so it is recommended that you install it. However, if you just want to load GPS data from GPX files you will not need it. Version 1.2.3 of GPSBabel is known to work with QGIS, but you should be able to use later versions without any problems.

6.2.4 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, which featuretype 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.

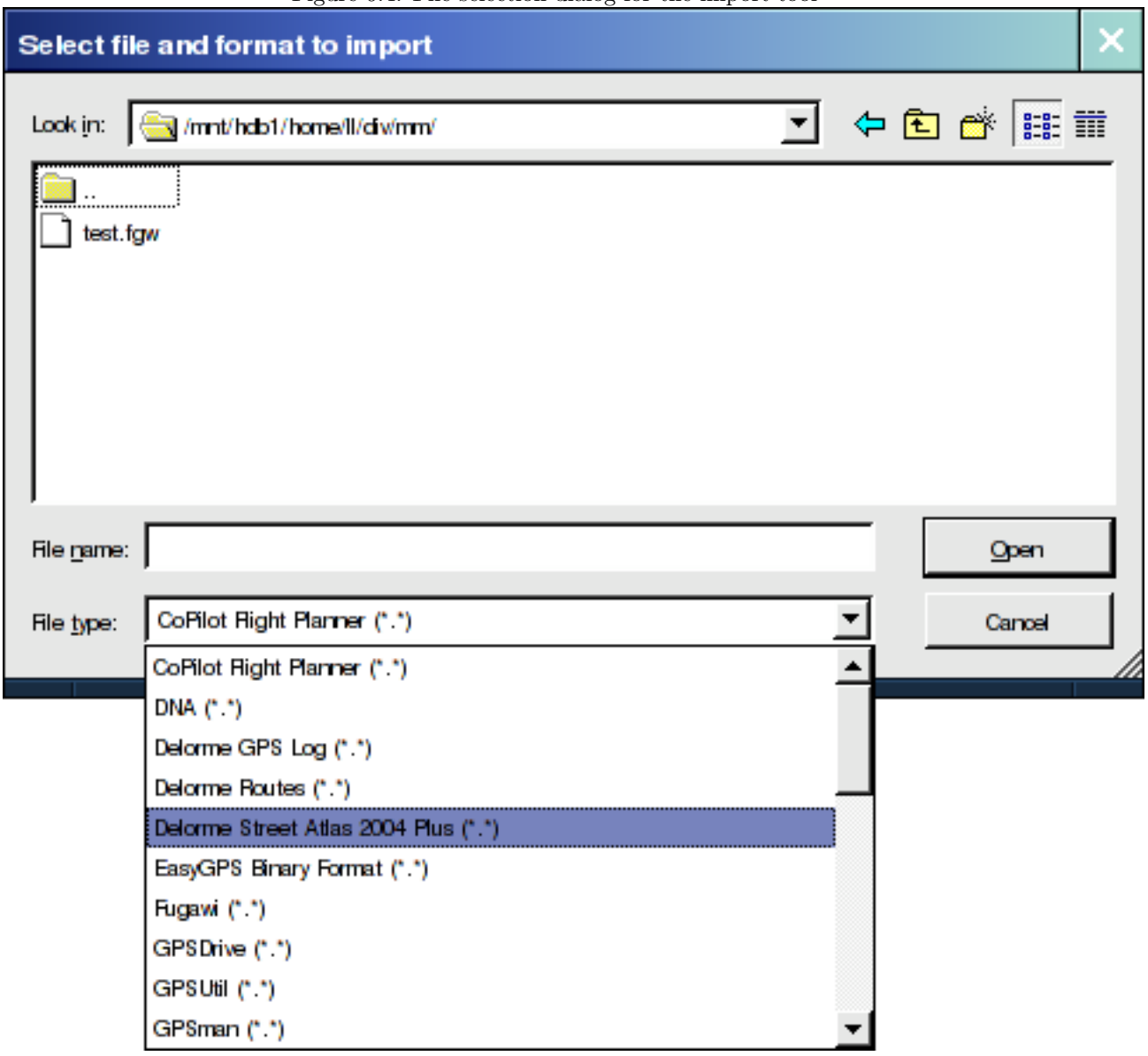
When you select the file to import you must also select the format of that file by using the menu in the file selection dialog (see figure 6.4). All formats do not support all three feature types, so for many formats you will only be able to choose between one or two types.

6.2.5 Downloading GPS data from a device

QGIS can use GPSBabel to download data from a GPS device directly into vector layers. For this you use the tool *Download from GPS* (see figure 6.5), where you select your type of GPS device, the port that it is connected to, 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 GPSBabel tries to communicate with

Figure 6.4: File selection dialog for the import tool

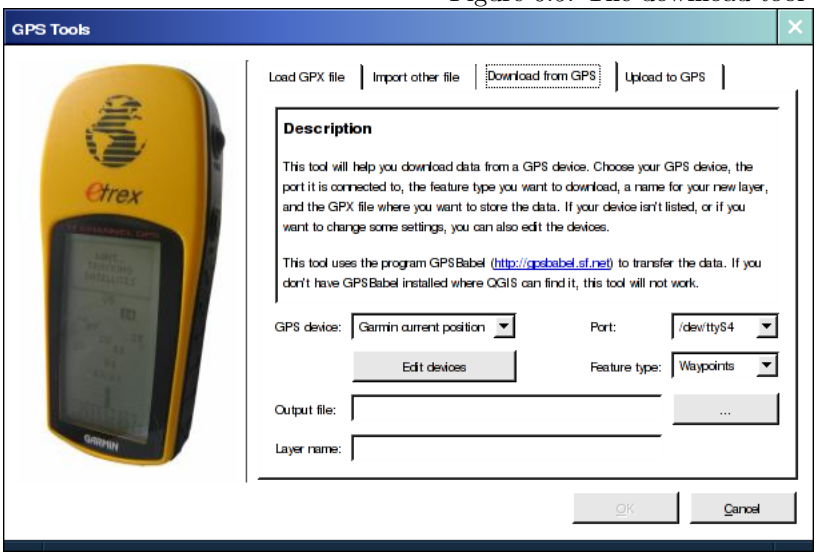


the device. If none of the device types works with your GPS device you can create a new type (see section 6.2.7).

The port is a filename 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. On Linux this is something like /dev/ttyS0 or /dev/ttyS1 and on Windows it's COM1 or COM2.

When you click OK the data will be downloaded from the device and appear as a layer in QGIS.

Figure 6.5: The download tool



6.2.6 Uploading GPS data to a device

You can also upload data directly from a vector layer in QGIS to a GPS device, using the tool *Upload to GPS*. The layer must be a GPX layer. To do this you simply select the layer that you want to upload, the type of your GPS device, and the port that it's 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 together with the vector editing capabilities of QGIS. You can load a map, create some waypoints and routes, and then upload them and use them in your GPS device.

6.2.7 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 and upload 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 window.

To define a new device you simply click the *New device* button, enter a name, a download command, and an upload command for your device, and click the *Update device* button.

The name will be listed in the device menus in the upload and download windows, and 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 it's 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.

CHAPTER 7: Help and Support

QGIS is still under active development and as such it won't always work like you expect it to. The preferred way to get help is by joining the qgis-users mailing list Your questions will reach abroader audience and answers will benefit others. You can subscribe to the qgis-users mailing list by visiting here: http://lists.sourceforge.net/lists/listinfo/qgis-user

If you are a developer facing problems of a more technical nature, you may want to join the qgis-developer mailing list here: http://lists.sourceforge.net/lists/listinfo/qgis-developer

We also maintain a presence on IRC - visit us by joining the #qgis channel on irc.freenode.net. Please wait around for a response to your question as many folks on the channel are doing other things andit may take a while for them to notice your question. Commercial support for QGIS is available from Micro Resources

While the qgis-users mailing list is useful for general 'how do I do xyz in QGIS' type questions, you may wish to notify us about bugs in QGIS. You can submit bug reports using the QGIS bug tracker. When reporting a bug, either login to SourceForge or, if you don't have a SourceForge id, provide an email address where we can request additional information. Feature requests can be submitted using the feature tracker. Please bear in mind that your bug may not always enjoy the priority you might hope for (depending on its severity). Some bugs may require may require significant developer effort to remedy and the manpower is not always available for this.

If you have found a bug and fixed it yourself you can submit it to the QGIS Sourceforge patch queue where someone will review it and apply it to QGIS. Please dont be alarmed if your patch is not applied straight away - developers may be tied up with other committments.

There is also a community site for QGIS where we encourage QGIS users to share their experiences and provide case studies about how they are using QGIS. The community site is available at: http://community.qgis.org

Lastly, we maintain a WIKI web site at http://wiki.qgis.org where you can find a variety of useful information relating to QGIS development, release plans, links to download sites and so on.

APPENDIX A: Supported Data Formats

A.1 Supported OGR Formats

At the date of this document, the following formats are supported by the OGR library. Formats known to work in QGIS are indicated in **bold**.

- **Arc/Info Binary Coverage**
- Comma Separated Value (.csv)
- DODS/OPeNDAP
- **ESRI Shapefile**
- FMEObjects Gateway
- GML
- IHO S-57 (ENC)
- **Mapinfo File**
- Microstation DGN
- OGDI Vectors
- ODBC
- Oracle Spatial
- PostgreSQL[1]
- **SDTS**
- SQLite
- UK .NTF
- U.S. Census TIGER/Line
- VRT - Virtual Datasource

A.2 GDAL Raster Formats

At the date of this document, the following formats are supported by the GDAL library. Note that not all of these format may work in QGIS for various reasons. For example, some require external commercial libraries. 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 *All other files (*)* filter. Formats known to work in QGIS are indicated in **bold**.

- **Arc/Info ASCII Grid**
- **Arc/Info Binary Grid (.adf)**
- Microsoft Windows Device Independent Bitmap (.bmp)
- BSB Nautical Chart Format (.kap)
- VTP Binary Terrain Format (.bt)

[1] QGIS implements its own PostgreSQL functions. OGR should be built without PostgreSQL support

- CEOS (Spot for instance)
- First Generation USGS DOQ (.doq)
- New Labelled USGS DOQ (.doq)
- Military Elevation Data (.dt0, .dt1)
- ERMapper Compressed Wavelets (.ecw)
- ESRI .hdr Labelled
- ENVI .hdr Labelled Raster
- Envisat Image Product (.n1)
- EOSAT FAST Format
- FITS (.fits)
- Graphics Interchange Format (.gif)
- **GRASS Rasters**[2]
- **TIFF / GeoTIFF (.tif)**
- Hierarchical Data Format Release 4 (HDF4)
- **Erdas Imagine (.img)**
- Atlantis MFF2e
- Japanese DEM (.mem)
- **JPEG JFIF (.jpg)**
- JPEG2000 (.jp2, .j2k)
- JPEG2000 (.jp2, .j2k)
- NOAA Polar Orbiter Level 1b Data Set (AVHRR)
- Erdas 7.x .LAN and .GIS
- In Memory Raster
- Atlantis MFF
- Multi-resolution Seamless Image Database MrSID
- NITF
- NetCDF
- OGDI Bridge
- PCI .aux Labelled
- PCI Geomatics Database File
- Portable Network Graphics (.png)
- Netpbm (.ppm,.pgm)
- **USGS SDTS DEM (*CATD.DDF)**
- SAR CEOS
- **USGS ASCII DEM (.dem)**
- X11 Pixmap (.xpm)

[2]GRASS raster support is supplied by the QGIS GRASS data provider plugin

APPENDIX B: Gnu Public License

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

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.

B.1 Quantum GIS 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-commerical 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.

APPENDIX C: QGIS Installation Guide

C.1 Introduction

This document briefly describes how to build QGIS 0.6 (*'Simon'*) from the source distribution. These instructions are for Linux/Unix and other POSIX systems which have the required build environment.

The latest version of this document can always be found at http://qgis.org/docs/install.html.

C.1.1 Installing Windows Version

Installing the Windows version of QGIS is simply a matter of running the user friendly setup wizard. See the README.WIN32 file for additional information regarding the Windows version of QGIS. At version 0.6, the GRASS plugin is not available in Windows.

C.1.2 Installing Mac OSX Version

To install the compressed disk image containing the OSX version of QGIS, double-click to expand and mount the image, then drag QGIS application to your hard drive. The OSX version does not include GRASS support.

C.1.3 Building from Source

The remainder of this document deals with compiling and installing QGIS from the source code. Specifically this applies to Linux/Unix systems.

QGIS can be installed with three levels of support for data stores:

1. Basic raster and vector support (GDAL and OGR formats)
2. PostreSQL/GEOS/PostGIS
3. GRASS raster and vector support

Basic support uses the GDAL/OGR libraries and supports many raster and vector formats. For more information on the available formats, see http://www.remotesensing.org:16080/gdal/formats_list.html and http://www.remotesensing.org:16080/gdal/ogr/ogr_formats.html.

PostgreSQL/PostGIS support allows you to store spatial data in a PostgreSQL database. GRASS support provides access to GRASS mapsets.

Note: - If you plan to build QGIS with GRASS support, version 1.2.3 or higher of GDAL must be used.

Each of the requirements are discussed below. Note that the information given below is abstracted from the installation documentation for each of the libraries. See the install information for each library to get detailed instructions. In the documentation below, the file names and versions used are examples.

If you are building QGIS without PostgreSQL or GRASS support, skip to the section on Installing GDAL/OGR.

C.2 Getting QGIS

QGIS is available in both source and package format from http://qgis.org.

In addition, packages for many Linux distributions are independently maintained in various locations. See the *Download* section on http://qgis.org for the latest information on package locations.

Packages for most of the software/libraries discussed below can be found for almost all Linux distributions. While it is possible to mix compiling from source and installing packages to meet the requirements for QGIS, sometimes this becomes tricky. Following the steps below will generally ensure a successful installation. If you are using SuSE 9.1, the LinGIS distribution ftp://ftp.lingis.org is a good choice for installing QGIS and its dependencies.

C.3 PostgreSQL

QGIS uses the latest features of PostgreSQL. For this reason, version 7.4.x or higher is recommended with QGIS version 0.5. If you choose to add PostgreSQL, you must also install PostGIS and the GEOS library (see below).

1. Download PostgreSQL source from www.postgresql.org
2. Extract the source

   ```
   tar -xzf postgresql-7.4.1.tar.gz
   ```

3. Change to the source directory

   ```
   cd postgresql-7.4.1
   ```

4. Configure PostgreSQL:

   ```
   ./configure --prefix=/usr/local/pgsql
   ```

5. Build

```
make
```

6. Install

```
make install
```

7. As root, create the postgres user and setup the database (following taken from PostgreSQL INSTALL file with modification)

   - Create the postgres user

     ```
     adduser postgres
     ```

   - Create the directory for the PostgreSQL database

     ```
     mkdir /usr/local/pgsql/data
     ```

   - Change ownership of the data directory to the postgres user

     ```
     chown postgres /usr/local/pgsql/data
     ```

   - su to the postgres user (or login as postgres)

     ```
     su - postgres
     ```

   - Change to the PostgreSQL install directory

     ```
     cd /usr/local/pgsql
     ```

   - Initialize the database

     ```
     ./bin/initdb -D /usr/local/pgsql/data
     ```

   - Start the PostgreSQL daemon

     ```
     ./bin/pg_ctl start  -o "-i" -D /usr/local/pgsql/data -l /home/postgres/serverlog
     ```

   - Create the test database

     ```
     ./bin/createdb test
     ```

8. PostgreSQL should now be running. Logon as the postgres user (or use su - postgres). You should be able to connect to the test database and execute a test query with the following commands:

```
psql test
select version();
version
------------------------------------------------------------------------------------------
PostgreSQL 7.4.1 on i686-pc-linux-gnu, compiled by GCC gcc (GCC) 3.3.1 (SuSE Linux)
(1 row)

\q
```

9. PostgreSQL install is done

C.4 GEOS

Note: As of version 0.6, GEOS is a requirement in order to build QGIS.

QGIS uses GEOS to properly fetch features from the the underlying datastore when doing an identify or select operation.

To install GEOS:

1. Download GEOS source from http://geos.refractions.net
2. Untar GEOS

   ```
   tar -xzf geos-2.0.0.tar.gz
   ```

3. Change to the GEOS source dir

   ```
   cd geos-2.0-.0
   ```

4. Follow the instructions in the GEOS README file to complete the installation. Typically the install goes like this:

   ```
   ./configure
   make
   make install
   ```

C.5 PostGIS

NOTE - You must edit the PostGIS Makefile and make sure that USE_GEOS=1 is set. Also adjust GEOS_DIR to point to your GEOS installation directory.

1. Download PostGIS source from http://postgis.refractions.net
2. Untar PostGIS into the contrib subdirectory of the postgresql build directory. The contrib subdirectory is located in the directory created in step 3 of the PostgreSQL installation process.
3. Change to the postgis subdirectory
4. Edit the Makefile to enable GEOS support (see the note above)
5. PostGIS provides a manual in the doc/html subdirectory that explains the build process (see the Installation section)
6. The quick and dirty steps to install PostGIS are:

```
cd contrib
gunzip postgis-0.8.0.tar.gz
tar xvf postgis-0.8.0.tar
cd postgis-0.8.0
make
make install
createlang plpgsql yourtestdatabase
psql -d yourtestdatabase -f postgis.sql
psql -d yourtestdatabase -f spatial_ref_sys.sql
```

The **better way** to install PostGIS is to carefully follow the instructions in the PostGIS manual in the doc/html subdirectory or the online manual at http://postgis.refractions.net/docs

C.6 GRASS

If you want QGIS to support GRASS vector and raster layers, you must build GRASS prior to proceeding. Follow the directions on the GRASS website carefully to build version 5.7. Additional information and the build instructions can be found at http://grass.itc.it.

The GRASS software is available for download at http://grass.itc.it/download.html.

C.7 GDAL/OGR

The GDAL and OGR libraries provide support for raster and vector data formats. QGIS makes use of both of these libraries (which come bundled in one distribution).

Note: A Linux binary of GDAL is available at http://www.remotesensing.org/gdal. If you choose to install the binary you will also need to download and unpack the source tree since QGIS needs the header files in order to compile.

To install GDAL/OGR from source:

1. Download the GDAL distribution from http://www.remotesensing.org/gdal. You should use version 1.1.9 or higher. Versions prior to 1.1.9 contained a bug that caused problems when a null feature was encountered. If you want to build vector support for GRASS, you must use GDAL 1.2.3.
2. Untar the distribution

   ```
   tar xfvz /../path/../gdal-x.x.x.tar.gz
   ```

3. Change to the gdal-x.x.x subdirectory that was created by step 2

   ```
   cd gdal-x.x.x
   ```

4. Configure GDAL

   ```
   ./configure
   ```

   or if you want GRASS support

   ```
   ./configure --with-grass=<full path to grass install>
   ```

   Depending on the GDAL version you are building, it may be necessary to specify –without-ogdi when running configure if you don't have the OGDI libary avaiable on your system.

5. Build and install GDAL:

   ```
   make
   su
   make install
   ```

6. In order to run GDAL after installing it is necessary for the shared library to be findable. This can often be accomplished by setting LD_LIBRARY_PATH to include /usr/local/lib. On Linux, you can add /usr/local/lib (or whatever path you used for installing GDAL) to /etc/ld.so.conf and run ldconfig as root.
7. Make sure that gdal-config (found in the bin subdirectory where GDAL was installed) is included in the PATH. If necessary, add the path to gdal-config to the PATH environment variable.

   ```
   export PATH=/../path/../gdal-config:$PATH
   ```

8. Check the install by running:

   ```
   gdal-config --prefix
   ```

If you've had problems during the installation, refer to this manual, where the whole process is described with some more detail: http://www.remotesensing.org/gdal/gdal_building.html

C.8 Qt

Qt 3.1.2 or higher is required in order to compile QGIS. You may already have Qt on your system. If so, check to see if you have version 3.1.2 or later. You can check the Qt version using the find command:

```
find ./ -name qglobal.h 2>/dev/null | xargs grep QT_VERSION_STR
```

If you have the locate utility installed you can do the same more quickly using:

```
locate qglobal.h | xargs grep QT_VERSION_STR
```

In either case the result should look something like this:

```
#define QT_VERSION_STR   "3.3.1"
```

In the example above, Qt 3.3.1 is installed.

If Qt is not installed, you will have to install the Qt development package for your distribution. If you are not able to install the required Qt packages, you will have to build from source.

To install Qt from source:

1. Download Qt from http://www.trolltech.com/developer (choose the Qt/X11 Free Edition)
2. Unpack the distribution
3. Follow directions provided in the distribution directory (doc/html/install-x11.html)
4. Use whatever configure options you like but make sure you include -thread for use with QGIS. You can configure Qt with minimal options:

   ```
   ./configure -thread
   ```

5. Complete the installation per the instructions provided in the Qt documentation (see step 3)

C.9 Building QGIS

After you have installed the required libraries, you are ready to build QGIS. Download and untar the QGIS distribution and change to the QGIS source directory. You have two options for building and installing QGIS: **Quick and Dirty** and the **right way**.

C.9.1 Quick and Dirty

If you don't need PostgreSQL support and have installed GDAL , you can configure and build QGIS by changing to the distribution directory and typing:

```
./configure
    make
    make install
```

The above assumes that the gdal-config program is in your PATH See the next section for the full configuration instructions.

C.9.2 Configuring QGIS the Right Way

To see the configure options available, change the the QGIS directory and enter:

```
./configure --help
```

Among other options, there are three that are important to the success of the build:

```
--with-qtdir=DIR              Qt installation directory default=$QTDIR
--with-gdal=path/gdal-config  Full path to 'gdal-config' script,
                              e.g. '--with-gdal=/usr/local/bin/gdal-config'
--with-pg=path/pg_config      PostgreSQL (PostGIS) Support
                             (full path to pg_config)
 --with-grass=DIR             GRASS Support (full path to GRASS binary package)
```

Qt

The configure script will detect Qt, unless it is installed in a non-standard location. Setting the QTDIR environment variable will make ensure that the detection succeeds. You can also specify the path using the –with-qtdir option.

GDAL

If the gdal-config script is in the PATH, configure will automatically detect and configure GDAL support. If not in the path, you can specify the full path to gdal-config using the –with-gdal option. For example:

```
/configure --with-gdal=/usr/mystuff/bin/gdal-config
```

PostgreSQL

If the pg_config script is in the PATH, configure will automatically detect and configure PostgreSQL support. If not, you can use the –with-pg option to specify the full path to pg_config. For example:

```
./configure --with-pg=/usr/local/psql/bin/pg_config
```

GRASS

To build QGIS with GRASS support you must specify the full path to the installed GRASS binary package:

```
./configure --with-grass=/usr/local/grass-5.7.0
```

This assumes that GRASS is installed in the default location. Change the path to match the location of your GRASS installation.

Example Use of Configure

An example of use of configure for building QGIS with all options:

```
./configure --prefix=/usr/local/qgis \
--with-gdal=/usr/local/gdal/bin/gdal-config \
--with-pg=/usr/local/psql/bin/pg_config \
--with-grass=/usr/local/grass-5.7.0
```

This will configure QGIS to use GDAL, GRASS, and PostgreSQL. QGIS will be installed in /usr/local/qgis.

If QTDIR is set and gdal-config and pg_config are both in the PATH, there is no need to use the –with-gdal and –with-pg options. The configure script will properly detect and configure GDAL and PostgreSQL. You must still use the –with-grass option if building with GRASS support.

Compiling and Installing QGIS

Once properly configured simply issue the following commands:

```
make
make install
```

NOTE - As of version 0.1, you can no longer run QGIS from the src directory. You must do a make install and start QGIS from the installed location. In the case of the example above, the QGIS binary resides in the bin subdirectory of the directory specified with the prefix option (/usr/local/qgis/bin).

For information on using QGIS see the QGIS User Guide.

C.10 Building Plugins

The QGIS source distribution contains a number of "core" plugins. These are built along with QGIS using the instructions above. Additional external plugins are available from the QGIS community website at http://community.qgis.org. Instructions for building an external plugin can be found at http://wiki.qgis.org/qgiswiki/StepByStepBuildInstructions. Some external plugins may include instructions on building. If so, follow the instructions provided with the plugin rather than those provided in the wiki.

INDEX