Quantum GIS (QGIS) Web Client

Installation and Configuration Guide

Monday February 24, 2014

Last Updated: Monday February 24, 2014 Last Change : Monday February 24, 2014

Contents

1 For the terminally lazy

Listing

```
sudo apt-get install apache2 libapache2-mod-fcgid
cp apache-conf/qgis-web-client.conf.tmpl apache-conf/qgis-web-client.conf
```

Update the paths in the copied file then:

Listing

```
cd /etc/apache2/sites-available/
ln -s <path to apache-conf/qgis-web-client.conf> .
sudo a2enmod rewrite
sudo a2ensite qgis-web-client.conf
sudo /etc/init.d/apache2 reload
```

1. Check the symlink in cgi-bin is correct.
2. Check the QGIS libs are in your /etc/ld.so.conf path
3. Copy site/index.xml and check paths match your system OR Modify index.html and point your browser to that

2 Purpose

A WMS based webgis client that makes use of QGIS specific WMS extensions (e.g. highlighting, printing, metadata, etc.). QGIS webclient reads the configuration from the WMS GetCapabilities command and builds the layer tree accordingly. Supports legend graphic, feature info requests and printing.

The client builds on existing Web-GIS libraries OpenLayers and GeoExt, as well as ExtJS 3 for the GUI widgets.

All major browsers should be supported.

3 Installation

Requirements (Server):

- Apache2 - Webserver (Ubuntu: apache2)
- mod-fcgid (Ubuntu: libapache2-mod-fcgid)
- QGIS and QGIS Server (best installed from source)

On ubuntu you can meet these requirements by simply doing:

Listing

```
sudo apt-get install libapache2-mod-fcgid
```

The QGIS server compilation and installation will be covered in the QGIS manual.

For searching:

- python-wsgi for searching (Ubuntu: libapache2-mod-wsgi)
- psycopg2 PostgreSQL db driver (Ubuntu: python-psycopg2)
- webob - Python module providing WSGI request and response objects (Ubuntu: python-webob)

  The client part needs to be git cloned with the following command: git clone https://github.com/qgis/qgis-web-client.git

4 Configuration of Client

Global Settings for all projects (make a copy from one of the templates provided):

Listing

```
site/js/GlobalOptions.js
```

Translations (additional languages):

Listing

```
site/js/Translations.js
```

Project settings and index:

Listing

```
site/index.xml or site/index.html
```

Stylesheet of project index:

Listing

```
site/gis-project_listing.xsl
```

Thumbnails for individual projects (if you take the index.xml route):

Listing

```
thumbnails/projectname.png
```

4.1 Per Project Startup Options

These options are usually defined in the file site/index.xml. The stylesheet then generates the startup URL parameter options. On the client, the file 'site/js/GetUrlParams.js' is responsible for the proper handling of the startup options. The following options are available:

Listing

```
lang
```

An optional language parameter. Normally, this parameter is defined in the file 'site/js/GlobalOptions.js'. Optionally this can be overwritten on a per project base. Allowed values are two-character language codes that must be present in the file 'site/js/Translations.js'.

Listing

```
visibleLayers
```

A comma-separated list of layers that should be set visible on the project start.

Listing

```
format
```

This optional parameter allows a per project definition of the file format. Valid values are 'image/png', 'image/jpeg' and 'image/png;mode=8bit'. Defaults to 'image/png' if no format is given per project. For correct specification of 'image/png;mode=8bit' in a URL please encode it correctly: 'image%2fpng%3b%20mode%3d8bit'. If you specify this in site/js/GISProjectListing.js you do not need to encode it.

Listing

```
fullColorLayers
```

An optional comma-separated list of layers that need to be in full color (24bit). This parameter is only relevant if the project default image format is set to 'image/png' or 'image/png;mode=8bit'. If any of the layers in the fullColorLayers parameter list is set visible, the format changes to 'image/jpeg'.

Listing

```
maxExtent
```

The maximum extent of the project. This parameter is used if the 'Full View' navigation button is clicked. If the 'startExtent' parameter is not specified, 'maxExtent' will also be used as the 'startExtent'. The format is: left,bottom,right,top in map units.

Listing

```
startExtent
```

The initial extent on project load if the project should start with a given, but not the maximum extent (e.g. for zooming to a specific project area). Not to be confused with the 'maxExtent' parameter. The format is: left,bottom,right,top in map units.

Listing

```
searchtables
```

An optional list of additional search tables specific to the project. The format is 'sche-

maname.tablename'. These additional search tables will be used for the search field at the top-right corner of the Webclient-GUI. The default search tables are hard-coded in the file 'wsgi/search.wsgi', in the 'searchtables' array.

4.2 Configuration of search panels

There are two types of search panels supported, using a direct WMS GetFeatureInfo request or using URL rewriting with a much shorter search URL.

The search panels are configured in 'site/js/GlobalOptions.js'.

4.2.1 Using WMS GetFeatureInfo

Listing

```
var simpleWmsSearch = {
  title: "Search continent",
  query: 'simpleWmsSearch',
  useWmsRequest: true,
  queryLayer: "Country",
  formItems: [
    {
      xtype: 'textfield',
      name: 'name',
      fieldLabel: "Name",
      allowBlank: false,
      blankText: "Please enter a name (e.g. 'africa')"
    }
  ],
  gridColumns: [
    {header: 'Name', dataIndex: 'name', menuDisabled: 'true'}
  ],
  selectionLayer: 'Country',
  selectionZoom: 0
};
```

- `title`: title of the search tab
- `query`: identifier for this search
- `useWmsRequest`: enabled for WMS GetFeatureInfo request
- `queryLayer`: name of query layer
- `formItems`: list of Ext.form.FormPanel item configs
  - `xtype`: form field type
  - `name`: name of query layer attribute
  - `fieldLabel`: visible text for this field
  - `blankText`: popup text for blank fields
- `gridColumns`: list of Ext.grid.GridPanel column configs to show search results
- `selectionLayer`: name of layer for marking selected results (the same as `queryLayer`)
- `selectionZoom`: zoom level for jump-to when selecting results

Request URL:

When performing a search query using the above configuration, the following get request will be made.

"http://localhost/wms/helloworld?SERVICE=WMS&VERSION=1.1.1& REQUEST=GetFeatureInfo&LA
FEATURE_COUNT=10&INFO_FORMAT=text/xml&SRS=EPSG:4326& FILTER=Country:"name"+=-

4.2.2 Using URL Rewriting

For security and neatness, you may prefer to use rewritten URLs (so that your internal server file paths are not revealed. In that case your options file would contain something like this:

Listing

```
var urlRewriteSearch = {
  title: "Search letter",
  query: 'samplesearch',
  formItems: [
    {
      xtype: 'hidden',
      name: 'query',
      value: 'samplesearch'
    },
    {
      xtype: 'textfield',
      name: 'colour',
      fieldLabel: "Colour",
      allowBlank: false,
      blankText: "Please enter a colour (e.g. 'orange')"
    }
  ],
  gridColumns: [
    {header: 'PKUID', dataIndex: 'pkuid', menuDisabled: 'true'},
    {header: 'Colour', dataIndex: 'colour', menuDisabled: 'true'}
  ],
  selectionLayer: 'Hello',
  selectionZoom: 1
};
```

- `title`: title of the search tab
- `query`: identifier for this search
- `formItems`: list of Ext.form.FormPanel item configs, the `query` form field is required to match the rewrite rule (value is the same as `query`)
  - `xtype`: form field type
  - `name`: name of query layer attribute
  - `fieldLabel`: visible text for this field
  - `blankText`: popup text for blank fields

- `gridColumns`: list of Ext.grid.GridPanel column configs to show search results
- `selectionLayer`: name of layer for marking selected results
- `selectionZoom`: zoom level for jump-to when selecting results

  For every search of this type you have to add a URL rewrite rule in the Apache config. *Note:* Linebreaks added for formatting - they should be removed in your config file.

  Listing

```
RewriteCond %{QUERY_STRING} ^(?:.*)query=samplesearch&*(?:.*)$
RewriteCond %{QUERY_STRING} ^(?:(?:.*)&)?colour=([^&]*)(?:.*)$
RewriteRule ^/wms/(.+)$ /cgi-bin/qgis_mapserv.fcgi?map=/
<path-to-qgis-server-projects>/$1.qgs&SERVICE=WMS&VERSION=1.1.1&
REQUEST=GetFeatureInfo&LAYERS=Hello&QUERY_LAYERS=Hello&FEATURE_COUNT=20&
INFO_FORMAT=text/xml&SRS=EPSG:4326&FILTER=Hello:"colour"\ =\ '%1' [PT]
```

  The first RewriteCond matches the `query` id of the search panel config. The second RewriteCond extracts the values of the search request parameters.

  The RewriteRule composes the actual WMS GetFeatureInfo request to QGIS mapserver.

Request URL:

`http://localhost/wms/helloworld?query=samplesearch&colour=orange`

4.2.3 Add search panels to projects

In order for your search panel to appear in the web UI, you must enumerate them in your GlobalOptions.js for example (with url rewriting):

Listing

```
var mapSearchPanelConfigs = {
  "helloworld": [simpleWmsSearch, urlRewriteSearch]
};
```

Example (no rewriting):

Listing

```
var mapSearchPanelConfigs = {
  "../projects/helloworld.qgs": [simpleWmsSearch, urlRewriteSearch]
};
```

Search panels are added to a project by adding a new key for the map name with

a list of search panel configs to `mapSearchPanelConfigs`. If there is no search panel configuration for a project, the search will be hidden in the GUI.

The map name is whatever is passed in the get request for your .qgs file. For example if your url includes this:

Listing

```
http://localhost/cgi-bin/qgis_mapserv.fcgi?map=../projects/helloworld.qgs
```

then your mapSearchPanelConfigs should reflect ../projects/helloworld.qgs as the key for the search list.

4.3 Configuration of the theme switcher

The theme switcher allows to change to a diffent QGIS project (or map theme) without having to leave the application and using the map extent. To enable/disable the theme switcher you have to set the variable

Listing

```
var mapThemeSwitcherActive = true;
```

in the site/js/GlobalOptions.js file to true|false. In addition you should place thumbnail images of your map into the directory site/thumbnails where the file name equals the projectname. All thumbnails should be 300x200 pixels in size and in .png format. If you .qgs project is called 'helloworld.qgs' then your thumbnail should be called 'helloworld.png'.

In addition you need to make entries for topics and projects in the file 'site/js/GISProjectListing.js'. Please use the given file as a template. The file is in JSON format and starts with a few central parameters.

4.3.1 Central theme switcher parameters

Listing

```
path
```

The 'path' is the URL part used at the start of the application telling the QGIS Webclient where to find the QGIS projects (see also Apache URL rewriting). This path may be overwritten in some projects if you password-protect them in a separate Apache location.

Listing

```
mapserver
```

This is the path to the WMS server used for WMS requests (e.g. for GetCapabilities, GetFeatureInfo, etc. requests). Again, this parameter may be overwritten in some projects if you want to password-protect the WMS in a separate Apache location.

Listing

```
thumbnails
```

The URL where QGIS web client can find the project thumbnail images.

Listing

```
title
```

The overall title of your Web-GIS. This will be later appended with the name of your project, separated by a dash. It appears in the title bar of the browser window and in the title bar of the web application.

4.3.2 Per topic theme switcher parameters

You can group your projects into topics. A topic only has a single parameter with the name of the topic. In a topic element you can have several project entries in a JSON array called project.

Listing

```
name
```

The name of the topic.

4.3.3 Per project theme switcher parameters

In a topic you can have several project entries. A project can overwrite the global 'path' and 'mapserver' entries.

Listing

```
name
```


The name of the project or map. Will be displayed in the theme switcher below the thumbnail and in the title strings of the application.

Listing

```
path
```

Optional. Overrides the central settings in case you need to password-protect certain projects. The 'path' is the URL part used at the start of the application telling the QGIS Webclient where to find the QGIS projects (see also Apache URL rewriting).

Listing

```
mapserver
```

Optional. Overrides the central settings in case you need to password-protect certain projects. This is the path to the WMS server used for WMS requests (e.g. for GetCapabilities, GetFeatureInfo, etc. requests).

Listing

```
projectpath
```

The projectpath (directory) or part of the Apache rewrite expression necessary to find the project file. This parameter is mandatory.

Listing

```
projectfile
```

The QGIS project file or part of the Apache rewrite expression necessary to find the project file. This parameter is mandatory. Depending on the Apache rewrite expression you may have to omit the .qgis extension.

Listing

```
format
```

Optional. The image format that QGIS web client should request. Valid values are: 'image/jpeg', 'image/png' or 'image/png;mode=8bit'. If omitted, the value is taken from site/js/GlobalOptions.js. If it is not defined there either, the value defaults to 'image/png'.

Listing

```
visibleLayers
```


Optional. A comma separated list of layers that should be visible after loading the projects. A future QGIS Webclient version will also read the layer visibility directly from the GetProjectSettings command.

Listing

```
fullColorLayers
```

Optional. A comma separated list of layers that would trigger a format change from 'image/png' to 'image/jpeg'. Per defaul, the project would use 'image/png' or 'image/png;mode=8bit' but if the user toggles the visibility of a layer with orthophoto data or satellit images, the format will change to 'image/jpeg'.

Listing

```
updateInterval
```

Optional. A prosa text indicating how often the project will get data update. E.g. daily, weekly, monthly, weekly or occasional.

Listing

```
lastUpdate
```

Optional. The date of the last data update, e.g. '2012-10-23'.

Listing

```
responsible
```

Optional. The organization and/or person responsible for the project and the data involved.

Listing

```
startExtent
```

Optional. The bounding box (left,bottom,right,top in map units) used when starting the project. If not specified, maxExtent or the extent from the GetProjectSettings is used.

Listing

```
maxExtent
```

Optional. The maximum bounding box (left,bottom,right,top in map units) of the project. If not specified the extent from the GetProjectSettings is used.

Listing

```
showFeatureInfoLayerTitle
```

Optional. Boolean (true|false). Defines whether the layer title is displayed or not at the top of the popup bubble displaying the feature info results. Influences both the hover and the click popups.

Listing

```
tags
```

Optional. Tags or keywords displayed in the tooltips in the theme switcher. The tags are also used in the search filter used in the theme switcher.

4.4 Extending the interface

You can add buttons to implements additional functions (editing, advanced identify, etc.). See the example in site/js/Customizations.js.

5 URL Rewriting

Using a standard installation of QGIS server, GlobalOptions.js will have a WMS server configuration like

Listing

```
var serverAndCGI = "/cgi-bin/qgis_mapserv.fcgi";
```

A sample URL for QGIS Web Client installed in /var/www/qgis-web-client:

Listing

```
http://localhost/qgis-web-client/qgiswebclient.html?map=/opt/geodata/maps/NaturalEarth.qgs&visibleLayers=HYP_50M_SR_W
```

With the following rules for Apache mod_rewrite you can shorten the URLs to

Listing

```
var serverAndCGI = "/wms";
```

and

Listing

```
http://localhost/maps/NaturalEarth?visibleLayers=HYP_50M_SR_W
```

Rules in VirtualHost configuration:

Listing

```
# Forbid direct access
RewriteRule ^/cgi-bin/.*$ - [F]

# Search with SearchPanel (e.g. Address)
RewriteCond %{QUERY_STRING} ^(?:.*)query=address&*(?:.*)$
RewriteCond %{QUERY_STRING} ^(?:(?:.*)&)?street=([^&]*)(?:(?:.*)&)+number=([^&]*)(?:.*)$
RewriteRule ^/wms/(.+)$ /cgi-bin/qgis_mapserv.fcgi?map=/opt/geodata/maps/$1.qgs&SERVICE=WMS&VERSION=1.1.1&REQUEST=Get

# Rewrite /wms/mapname to qgis_mapserv.fcgi?map=mappath/mapname.qgs
RewriteRule ^/wms/(.+)$ /cgi-bin/qgis_mapserv.fcgi?map=/opt/geodata/maps/$1.qgs [QSA,PT]
# Rewrite /maps/mapname to qgis-web-client main page. mapname will be extracted for wms calls in Javascript code.
RewriteRule ^/maps/([^\.]+)$ /qgis-web-client/site/qgiswebclient.html [PT]
# Rewrite /maps/* to qgis-web-client/site (e.g. /maps/gis_icons/mActionZoomNext.png -> /qgis-web-client/site/gis_icon
RewriteRule ^/maps/(.*) /qgis-web-client/site/$1 [PT]
```

For supporting qgs files in subdirectories (e.g. /maps/subdir/mapnampe) replace last rule with:

Listing

```
RewriteRule ^/maps/[^/]+/(.*) /qgis-web-client/site/$1 [PT]
```

For adding zones in different subdirecories (e.g. maps and maps-protected) add the following rules:

Listing

```
RewriteRule ^/wms-protected/(.+)$ /cgi-bin/qgis_mapserv.fcgi?map=/opt/geodata/maps-protected/$1.qgs [QSA,PT]
RewriteRule ^/maps-protected/([^\.]+)$ /qgis-web-client/site/qgiswebclient.html [PT]
RewriteRule ^/maps-protected/(.*) /qgis-web-client/site/$1 [PT]
```

6 Configuration of search python script

Searching is handled by two separate python-wsgi scripts: "search.wsgi" lists back a hit list while the user is typing in the searchbox. It groups the results and returns a bounding box of the result. "getSearchGeom.wsgi" returns the actual wkt geometry for a selected search result. It is recommended to install the wsgi scripts in a separate directory, e.g. /home/www/wsgi, a place that is not reachable by regular web traffic.

6.1 Configuration of mod_wsgi

You need to enable mod_wsgi as root. (Ubuntu: a2enmod mod_wsgi).

You need to configure apache with the following lines (e.g. in file /etc/apache2/sites-available/default):

Listing

```
#mod_wsgi
WSGIDaemonProcess gis processes=5 threads=15 display-name=%{GROUP}
WSGIScriptAlias /wsgi/ /home/www/wsgi/
WSGIScriptAliasMatch ^/wsgi/([^/]+) /home/www/wsgi/$1.wsgi
```

6.2 Adaption of the wsgi scripts to your settings and needs

6.2.1 DB connection

In the files "search.wsgi" and "getSearchGeom.wsgi" please edit the line containing the db connection strings. Search for the line

Listing

```
conn = psycopg2.connect("host='yourhost' dbname='yourdb' port='5432' user='yourusername' password='yourpassword'")
```

and adapt the parameters according to your server/db. It is highly recommended to **connect with a database user having limited rights only** (e.g. select rights on relevant tables only).

6.2.2 Search type to be used

The search can use PostgreSQL's tsvector data type. "A tsvector value is a sorted list of distinct lexemes, which are words that have been normalized to merge different variants of the same word." from the PostgreSQL doc. Thus tsvector skips all the fill words and

reduces nouns to their single form, a behaviour useful for searching texts. However as we are normally dealing with **place names** here we want them to stay as they are. If you use a language where the single form is a lot different from the plural form but your name contains a plural you will not get a suitable result. If you want to use the tsvector search option you should activate the lines

Listing

```
sql += "searchstring_tsvector @@ to_tsquery(\'not_your_language\', %s)"
data += (querystrings[j]+":*",)
```

not_your_language is to be replaced with an entry e.g. *finnish* if you have German place names. Thus plural forms and fillwords are kept as they are. Be aware of side effects! Be sure to fill the field *searchstring_tsvector* with `'to_tsvector('not_your_language', 'yourstring')'`.

The use of

Listing

```
sql += "searchstring::tsvector @@ lower(%s)::tsquery"
data += (querystrings[j]+":*",)
```

is **discouraged** as it does not find a place name like *Stoke-sub-Hamden* when you enter *Stoke*.

If you do not want to use tsvector at all you can enable the full string comparison on the field *searchstring* (activated by default).

Listing

```
sql += "searchstring ILIKE %s"
data += ("%" + querystrings[j] + "%",)
```

This method however is slower than tsvector but not relevantly at least if you only have a couple 1000 datasets.

7 PostgreSQL table setup for searching

Listing

```
CREATE TABLE cadastre.searchtable
(
  searchstring text, --the search string (all lower case), e.g. "zürichstrasse 46, 8610 uster"
  displaytext text NOT NULL, --the display text for the search combobox, e.g. "Zürichstrasse 46, 8610 Uster (address)
  search_category text, --should have a leading two digit number:, e.g.
                        --"03_parcels", where 03 is the order of the search categories, the number
                        --should be unique across all search tables
  the_geom geometry,    --the actual geometry
  geometry_type text,   --the geometry type as returned by ST_GeometryType(the_geom)
  searchstring_tsvector tsvector, -- be sure to fill this with to_tsvector()
  CONSTRAINT searchtable_pkey PRIMARY KEY (displaytext)
)
WITH (
  OIDS=FALSE
);
GRANT SELECT ON TABLE cadastre.searchtable TO qwc_user;

-- Index: cadastre.in_cadastre_searchstring_tsvector_gin

CREATE INDEX in_cadastre_searchstring_tsvector_gin
  ON cadastre.searchtable
  USING gin
  (searchstring_tsvector);
```

The above search table can also be a view or materialized view. One can combine several search tables by specifying the ‘`searchtables=searchtable1,searchtable_n`‘ parameter when requesting the *search.wsgi* script. Any searchtable passed to *search.wsgi* may only contain the letters *A to Z, a to z* and the underscore. Double quoting the search table throws an error, thus searchtables’ names must contain lower characters only.

Using views is generally slower than properly indexed tables, check for yourself what works best.

8 Configuration of search PHP script

As an alternative to python-wsgi scripts, a set of PHP scripts is available under `php` folder. PHP scripts are similar in scope to python-wsgi scripts but should require less configuration and should work out of the box with just a few lines of configuration. The DB table structure must not be altered in order to use PHP search scripts.

8.1 Differences with python-wsi scripts

The main difference is that there are no hardcoded values and much less configuration is needed because all needed informations are read from the project file.

Another notable difference is that layer names are used instead of table names, this in order to not disclose internal DB details.

8.2 Available scripts

8.2.1 Search

The "search.php" scripts works in the same way as the "search.wsgi" (see the paragraph above). Accepted parameters:

- map (map name or path)
- query (search text)
- searchtables (optional: layer names to search in)

The companion "search_geom.php" has the same role of the wsgi script "getSearchGeom.wsgi".

- map (map name or path)
- searchtable (layer name)
- displaytext (the matched string)

8.2.2 Unique list

This simple script returns the unique values of a given column of a given PostgreSQL layer. Accepted parameters:

- map (map name or path)
- layer (layer name)
- field (column name)

The script returns a json array of unique values and can be useful to implement select combo boxes for the search panels.

8.2.3 Get legend

This script has no wsgi counterpart, it works with recent QGIS Server versions (2.0.1 and newer) and can be used to build a template-based HTML legend instead of the image provided by GetLegendGraphic calls.

To use this feature you must activate it in `GlobalOptions.js`, search for the commented line below:

Listing

```
var interactiveLegendGetLegendURL = '../php/get_legend.php?map=' + project_map + '&';
```

Legends generated by this script can be cached for speed, see the paragraph on configuration below.

Accepted parameters:

- map (map name or path)
- layer (layer name)

8.3 The configuration file

Configuration for the services is stored in ‘config.php‘.

Example:

Listing

```
/****************************
 * Map rewrite configuration
 */
// Prefix map name with path
#define('MAP_PATH_REWRITE', '/home/xxx/public_html/QGIS-Web-Client/projects/');
// Append .qgs to the map name
#define('MAP_PATH_APPEND_QGS', true);


/************************************
 * search configuration
 */
// Configuration for searchable layers
$searchlayers_config = array(
    // Key is layer name
    'Country' => array(
        // SQL for text search: where to search
        'search_column' => 'name'
```

```
    )
);

// Default search tables
define('DEFAULT_SEARCH_LAYERS', 'Country');
// Limit search results
define('SEARCH_LIMIT', 100);


/*************************************
 *  Get legend configuration
 */
// Cache expiry time in seconds 0=never cache
define('GET_LEGEND_CACHE_EXPIRY', 60*60);
// Cache directory, defaults to dirname(__FILE__) . '/legend_cache'
define('GET_LEGEND_CACHE_DIRECTORY', null);
// Defaults to current URL + '../cgi-bin/qgis_mapserv.fcgi?'
define('WMS_ONLINE_RESOURCE', null);

/* End configuration */
```

QGIS Web Client needs to know where to find the scripts, since most configuration is read from the project file, this must be passed in the query string, the file where this parameters are set is `GlobalOptions.js` see the example below:

Listing

```
// Adds project_map, read value from query string
var project_map = Ext.urlDecode(window.location.search.substring(1)).map;

var searchBoxQueryURL = '../php/search.php?map=' + project_map;
var searchBoxGetGeomURL = '../php/search_geom.php?map=' + project_map;
```

8.4 TODO

Permalinks: the permalinks script.

9 License

The QGIS web client is released under a BSD license.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

10 Developers and Contributors

Developers:

- Jürgen Fischer
- Marco Hugentobler
- Pirmin Kalberer
- Andreas Neumann
- Alessandro Pasotti
- Niccolo Rigacci
- Denis Rouzaud
- Bernhard Ströbl
- Tim Sutton
- Mathias Walker

Translators:

- Giovanni Allegri
- Germán Carrillo
- Paolo Cavallini
- Diana Galindo
- Mayeul Kauffmann
- Samuel Mesa
- Alessandro Pasotti
- Nelson Silva
- Pavlo Taranov

11 Acknowledgements

We'd like to thank the OpenLayers, GeoExt and ExtJS teams for providing their base libraries we build upon.