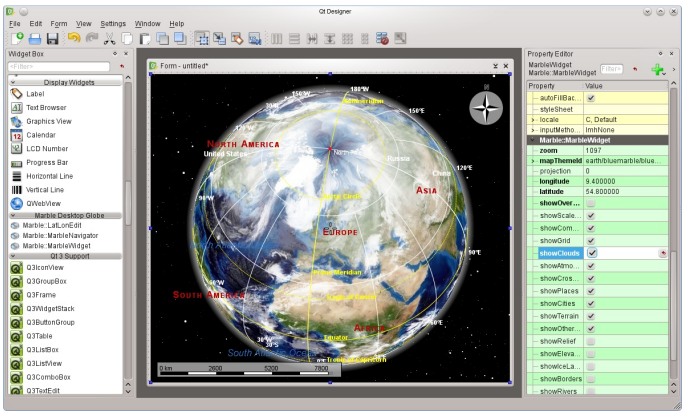
MARBLE

A Qt Geo-Mapping Library



Marble's core: libMarble

The popular Marble Virtual Globe is more than an application: Marble's whole feature set is offered by a mature software library called libMarble which is based on **Nokia's Qt framework**.
So if you're a Qt developer then you can easily add a Marble globe or other Marble capabilities into your own Qt application. Even better: since libMarble is highly modular and plugin-based you can tailor its feature set up to your needs.
And as Marble is a crossplatform library your application can run on all **desktop and mobile platforms** supported by Qt.

libMarble – Qt meets geo-mapping

With libMarble we provide a natural extension of the Qt library for creating and working with maps. The scope covers the full range of map related topics:

- a plugin-based **Location API**
- an intuitive **Maps API** that conveniently supports the most frequent use cases
- **Geodetic Storage and Tool APIs** modelled after the **OGC Standard KML**

All APIs of libMarble adhere to the well-known Qt design concepts and Qt conventions. For geographic aspects of the API we've chosen the popular open standard KML (as used in Google Earth/Maps) as a reference.
So any developer with Qt and KML knowledge will feel right at home using libMarble and can immediately get productive.

Website: http://www.marble-globe.org

License: GNU LGPL 2+ (Open Source)
Client Version: **1.0** , January 26, 2010
Library Version: **0.11.0**
OS: Linux, Windows, Mac OS X, MeeGo
Qt / C++

Visualizing your data

There are many solutions for integrating your map data into libMarble:

- **Maps can be specified using XML** in a simple text editor. So usually there is only little or no C++ code at all required to add or change map data.
- **You can develop Qt/C++ plugins** for libMarble to visualize data, webservices or your current location. In fact most Marble features are based on plugins, so you can use the existing code as an example.
- **Image tiles** are another popular way to provide map data. Marble supports all popular tile layout schemes (such as those used by OpenStreetMap, Google Maps and Ovi Maps).

In libMarble everything is handled in geodetic coordinates (latitude and longitude), so you don't need to worry about the projection used in the end – no matter whether it's the Globe, Equirectangular or Mercator.
Also **Marble supports Open Standards** which helps to import your data.

MARBLE

A Qt Geo-Mapping Library

License

libMarble is distributed under the LGPL license. Within the terms of the license the LGPL allows for **commercial and Open Source development**. See the LGPL license for details and limitations.
Marble and its library come with a selection of free map data that is covered by Terms of Use which are similar in spirit to the LGPL license.

Documentation

Some basic documentation is available on our website. Additionally there are in-depth tutorials available at:

http://techbase.kde.org/Projects/Marble

Community & Support

Marble and its library are developed by a **big growing community of volunteers**. If you need help or if you want to contribute there are several ways to get in touch with our Marble Team:

- Send an e-mail to **marble-devel@kde.org**
- Or join us on IRC (irc.freenode.org, #marble)
- Or follow us on Facebook (http://www.facebook.com/marbleglobe) or Twitter (http://www.twitter.com/marbleglobe)

In addition to our community support there is **commercial support** available (e.g. via Qt consulting companies such as basysKom or C-xx).

Tutorial – Hello Marble!

The API of the Marble library allows for a very easy integration of a map widget into your application.
Let's prove that with a tiny Hello world-like example: Qt beginners might want to have a look at the Qt Widgets Tutorial to learn more about the details of the code.
For a start we just create a QApplication object and a MarbleWidget object which serves as a window. By default the MarbleWidget uses the Atlas map theme. However for our first example we choose to display streets. So we set the maptheme id to OpenStreetMap. Then we call QWidget::show() to show the map widget and we call QApplication::exec() to start the application's event loop. That's all!

```
#include <QtGui/QApplication>
#include <marble/MarbleWidget.h>

using namespace Marble;

int main(int argc, char** argv)
{
  QApplication app(argc,argv);

  // Create a Marble QWidget without a parent
  MarbleWidget *mapWidget = new MarbleWidget();

  // Load the OpenStreetMap map
  mapWidget->setMapThemeId(
     "earth/openstreetmap/openstreetmap.dgml"
  );

  mapWidget->show();
  return app.exec();
}
```

Copy and paste the code above into a text editor (or Qt Creator). Then save it as my_marble.cpp and compile it by entering the folling command on the command line:

```
g++ -I /usr/include/qt4/ -o my_marble
my_marble.cpp -lmarblewidget -lQtGui
```

If things go fine, execute ./my_marble and you end up with a fully usable OpenStreetMap application:

http://www.marble-globe.org

part of the KDE Family