How to show city locations in a map using Qt and Google Maps for JavaScript

This article shows how to display city location on a map using QWebKit and Google Maps API.

Designing

20 Sep 2009

Before starting the real implementation, we need to build the application GUI (Graphical User Interface). Qt offers a strong tool called Qt Designer to build GUIs. In this example, we used Qt Designer to create the application form. Such a form contains five buttons (QPushButton) and a web view (QWebView). You can add new Qt components inside a form by dragging and dropping elements from Widget Box. The properties like name and label of a Qt component can be changed through the Property Editor.

Implementation

NOTE: Usage of this code with the free Google Maps API Key breaks Google's Terms and Conditions (section 10.8). You should purchase an Enterprise License if you wish to use the Google Maps API as shown in this example.

First, we need to implement a custom QWebView component. In this example called Map, such a component has some additional services that allow us to show the city locations in a map using the Google Maps API. The main service of the Map component is implemented by geoCode method. The geoCode method requests the coordinates of the given local, by using the Google Maps API.

```
void Map::geoCode(QString local)
{
   QUrl geoCodeUrl("http://maps.google.com/maps/geo");
   geoCodeUrl.addQueryItem("q", local);
   geoCodeUrl.addQueryItem("output", "csv");
   geoCodeUrl.addQueryItem("key", "GOOGLE_MAPS_KEY");
   manager->get( QNetworkRequest(geoCodeUrl) );
}
```

The geoCode response is received by replyFinished method. Such a method parses and stores the coordinate (latitude and longitude) in array of coordinates. After that, the reloadMap signal is emitted.

```
void Map::replyFinished(QNetworkReply *reply)
{
   QString replyStr( reply->readAll() );
   QStringList coordinateStrList = replyStr.split(",");

   if( coordinateStrList.size() == 4)
   {
      QPointF coordinate( coordinateStrList[2].toFloat(),
                          coordinateStrList[3].toFloat() );
      coordinates << coordinate;
   }
   emit reloadMap();
}
```

The reloadMap signal is connected to loadCoordinates slot. Thus, when the signal reloadMap is emitted, the loadCoordinates slot is invoked. The loadCoordinates method uses the QtWebKit's capability to call the JavaScript method Open that is defined in HTML loaded in Map component.

```
void Map::loadCoordinates()
{
   foreach(QPointF point ,coordinates)
   {
      this->page()->mainFrame()->evaluateJavaScript(
                     QString("Open(%1,%2)").arg(point.x()).arg(point.y()) );
   }
}
```

Such a HTML has the JavaScript code to show the city location using the Google Maps API. The initialize function creates a GMap2 with the center point in (0, 0) and the zoom level 1. Thus, the globe is shown completely on the map. The Open function updates the GMap2 center point to the arguments passed as parameter and the zoom level to 13.

```
var map;

function initialize()
{
```


(C) Copyright Nokia 2013. All rights reserved.

```
   if (GBrowserIsCompatible())
   {
      map = new GMap2(document.getElementById("map"));
      map.setCenter( new GLatLng(0,0),1 );
   }
}

function Open(x,y)
{
   map.setCenter( new GLatLng(x,y),13 );
}
```

The MainScreen uses the Map component to show the map with the city locations. For example: when the button1 is clicked, the geoCode method is invoked passing Campina Grande as the city to be shown.

```
void MainScreen::on_button1_clicked()
{
   map->clearCoordinates();
   map->geoCode("Campina Grande");
}
```

Compiling

In order to compile our application, we needed to include the following lines in our Qt project file. Such lines enable the Qt modules needed by our application

```
QT += network \
         xml\
         webkit
```

After that, we can compile our application using the Scratchbox cross-compiler for ARM processor by executing the following commands:

```
qmake –project
//then add the lines described above using a text editor(vi or gedit)
qmake showmapARM.pro
make
```

Executing

The screenshots below shows our application execution:

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.