Reverse Geocoding in QML

Introduction

Geocoding is the process of finding associated geographic coordinates (often expressed as latitude and longitude) from other geographic data, such as street addresses, or zip codes (postal codes). With geographic coordinates the features can be mapped and entered into Geographic Information Systems, or the coordinates can be embedded into media such as digital photographs via geotagging. Reverse geocoding is the opposite: finding an associated textual location such as a street address, from geographic coordinates. In this article and code snippet, we explain how we can get address of a place by reverse geocoding with Nokia map.

How to get address from coordinate

We call `reverseGeocode()` method of `QGeoSearchManager` and pass the coordinate of the place that we are looking for. If we are interested to a place near by then we can use our device's positioning system to get the current location and that can be passed to `reverseGeocode()` method as shown in the following snippet. `geocodeFinished` slot will be called when it has found the address.

```
QGeoSearchReply *reply = m_searchManager->reverseGeocode(m_geoCoordinate);
QObject::connect(reply, SIGNAL(finished()), this,SLOT(geocodeFinished()));
QObject::connect(reply,SIGNAL(error(QGeoSearchReply::Error, QString)), this,SLOT(resultsError(QGeoSearchReply::Error, QString)));
```

`geocodeFinished` has been implemented in the following code.

```
void CLocationSource::geocodeFinished()
{
    QGeoSearchReply* reply = static_cast<QGeoSearchReply *>(sender());

    if (reply->error() != QGeoSearchReply::NoError) {
        // Errors are handled in a different slot (resultsError)
        return;
    }

    QList<QGeoPlace> places = reply->places();
    if (places.length() == 0)
    {

    // do something here
    }
    else
    {
        QGeoAddress address = places[0].address();
        QString message =  address.city() + ", " + address.country().toLower();
        setAddress(message);
    }

    disconnect(reply, SIGNAL(finished()), this,SLOT(geocodeFinished()));
    disconnect(reply,SIGNAL(error(QGeoSearchReply::Error, QString)), this,SLOT(resultsError(QGeoSearchReply::Error, QString)));
    reply->deleteLater();
}
```

We need very thin QML code as most stuffs are done in Qt C++ side and that is brought to QML side by using `qmlRegisterType()` method.

```
        PositionSource {
            id: apositionSource
            updateInterval: 1000
            active: false
            // nmeaSource: "nmealog.txt"
        }
        CLocationSource
        {
            id: positionSource
            onLonChanged:
            {
                console.log(" onLonChanged in QML")
            }
            onLatChanged:
            {
                console.log(" onLatChanged in QML")
            }
            onAddressChanged:
            {
                console.log(" Address has been changed")
            }
        }

    Coordinate
    {
        id: mycordinate
        latitude: positionSource.lat
        longitude: positionSource.lon
    }

    Map
    {
        id: map
        plugin : Plugin { name : "nokia" }
        anchors.fill: parent; size.width: parent.width; size.height: parent.height; zoomLevel: 12
        center: mycordinate

    }
        Column
```


(C) Copyright Nokia 2013. All rights reserved.

```
        {
            Text {text: "Click on screen to swap positioningMethod: " }
            Text {text: "<==== Position ====>"}
            Text {text: "latitude: "   + mycordinate.latitude}
            Text {text: "longitude: "   + mycordinate.longitude}
            Text {text: "Address: "   + positionSource.address}
        }
```

Examples

Download the example for Nokia Belle: File:Mapqmlreversegeocode.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.