NOKIA Developer

Display Facebook Friend position with Nokia Maps API for Qt

This article demonstrates how to display Facebook friends on Nokia Map using Nokia Map API for Qt.

Introduction

The example combines getting information about Facebook friends and displays their location on Nokia Maps using pushpin. We create dynamic pushpins with friend's image using Dynamic Object Management in QML. This article also talks about Facebook login and authentication using OAuth. To start with the application first create a Facebook app from here and get the **Key** and **Secret** for future use.

Facebook Login Page

Facebook Friend list

Facebook Friends on Nokia Map

Implementation

We create two pages in `MainPage.qml`, one is for displaying Facebook friend list and the other is to display friends on Nokia Map. When we start the application we load the Facebook login dialog in the first screen, which calls the Facebook login *URL*.

```
https://www.facebook.com/dialog/oauth?client_id=<Your
Key>&redirect_uri=https://www.facebook.com/connect/login_success.html
```

By default the application will ask for the authentication of *basic information* about the user in Facebook. As we use user location and other information we need to request for more permissions from the user, which can be done with the *scope* parameter. So our *URL* should look like below.

```
https://www.facebook.com/dialog/oauth?client_id=<Your
Key>&redirect_uri=https://www.facebook.com/connect/login_success.html&scope=user_hometown,friends_hometown,user_location,friends_location,email,publish_stream,offline_access,read_stream,user_status,user_photos,friends_photos,friends_status,user_checkins,friends_checkins,user_events
```

The login page is an HTML page, so we display it on QML WebView.

```
WebView {
            id: webView
            anchors.fill: parent
            preferredHeight: height
            preferredWidth: Math.max(parent.width,640)
            url: "https://www.facebook.com/dialog/oauth?client_id=<Your
Key>&redirect_uri=https://www.facebook.com/connect/login_success.html&scope=user_hometown,friends_hometown,user_location,friends_location,email,publish_stream,offline_access,read_stream,user_status,user_photos,friends_photos,friends_status,user_checkins,friends_checkins,user_events";//&display=touch";
            onLoadFinished: {
                loginDialog.finished( webView.url );
            }
            onLoadFailed: {
                loginDialog.loadFailed();
            }
        }
```

Once the user is authenticated and authorizes the application, the browser will get an alpha-numeric number in the *code* parameter of its response. We parse the *code* for further use.

```
onFinished: {
                 if(url.indexOf("code=")>0) {
                     login.visible = false;
                     friendsList.visible= true;
                     var codeStart = url.indexOf("code=");
                     //CODE from facebook
                     var code = url.substring(codeStart + 5);
                     //get the access token
                     Facebook.getAccessToken(code);
                     console.log("Code: "+code);
                 }
             }
```

`getAccessToken()` function pass the *code* as its parameter and calls the *URL* to get the access token.

```
function getAccessToken(code) {
    //access token URL request
    var url = "https://graph.facebook.com/oauth/access_token?client_id=<Your
Key>&redirect_uri=https://www.facebook.com/connect/login_success.html&client_secret=<Your
Secret>&code=" +code;
    doWebRequest("GET", url, "", parseAccessToken);
}
```

And then we parse the access token from the response

```
//parse the access token
function parseAccessToken(response) {
    accessToken = "access_token=" + parseParameter(response, "access_token");
    console.log("parseAccessToken:"+accessToken);
    //load friends on list
    loadFriends();
}
```

`loadFriends()` gets friends information from Facebook and load the information on ListModel

```
function parseFriends(response) {
    var data = eval("[" + response + "]")[0].data;
    data = data.sort(sortByName);
    friendsModel.clear();
    for(var i in data) {
        var user = data[i];
        if(user.location!==undefined)
        {
            if(user.location.id !== "" || user.location.name !==  null )
            {
                //save friends data in model
        friendsModel.append({
                        "id": user.id,
                        "name": user.name,
                       // "profileImageUrl":"https://graph.facebook.com/" + user.id +
"/picture",
                         "profileImageUrl":user.picture ,
                        "lid":  (user.location!==undefined ? user.location.id : ""),
                        "lname": (user.location!==undefined ? user.location.name : ""),
    });
            }
        }
    }
    //count number of content in the list model
   friendsCount= friendsModel.count;
    console.log("parseFriends : "+friendsCount);
}
```

Once the information are loaded in the ListModel we display the list of Facebook friend List on the page

```
//display friends in list view
         FriendsList {
             id: friendsList
             friendsModel: friendsModel
             y: commonTools.height
             height: parent.height - commonTools.height
         }
```

On clicking the menu *Friends on Map* we call the function `startcalculatingfrineds()`, which first calculates the number of friends loaded in the model and then gets the location name for each friends.

```
function parseFriendsLocation()
    {
           //when friend list is more then zero
        // get information about each friend from the model
        if( i <friendsNumber)
        {
         fid =  friendsModel.get(i).id;
         fname = friendsModel.get(i).name;
         fprofileImageUrl = friendsModel.get(i).profileImageUrl;
            flid = (friendsModel.get(i).lid!==undefined ? friendsModel.get(i).lid : ""
);
            flname = (friendsModel.get(i).lname!==undefined ? friendsModel.get(i).lname
:"");
         i++;
            if(flname !== "" || flid!== "" )
            {
             //pass friend location to get the latitude and longitude
            feedModel.locationName = flname;
            feedModel.reload();
            }
         else
         {
                //no user
             console.log("no location for the user");
        }
        }
        else
        {
            //count else
            console.log("Stopped : "+friendsNumber);
        }
    }
```

The Facebook API we used here doesn't provide the *latitude* and *longitude* information of friends in the friend list.

```
https://graph.facebook.com/me/friends?fields=name,id,birthday,location,picture&" +
accessToken
```


(C) Copyright Nokia 2014. All rights reserved.

Though the below Facebook Query Language gives the *latitude* and *longitude* information of the user's location, but not for all location id.

```
https://api.facebook.com/method/fql.query?query=SELECT location,name FROM page WHERE
page_id = <user location id>&access_token=<Your Access Token>
```

So we though of using Google API to get the *latitude* and *longitude* of each location individually.

```
http://maps.googleapis.com/maps/api/geocode/xml?address="+locationName+"&sensor=false
```

```
XmlListModel {
    id: feedModel
    property string access_Token: ""
    property string xmlData:"";
    property string locationID:"";
    property string locationName:""
// using Google API to get latitude and longitude of a location
    source:"http://maps.googleapis.com/maps/api/geocode/xml?
address="+locationName+"&sensor=false"
    query:"/GeocodeResponse/result/geometry/location" //google query
    namespaceDeclarations:""
    //get latitude and longitude
    //google
     XmlRole { name: "lat"; query: "lat/string()" }
     XmlRole { name: "lng"; query: "lng/string()" }
}
```

And then calls the `addPushPin()` function which first creates instances of `PushPin.qml` component dynamically using Dynamic Object Management in QML. Each instance is a MapImage with properties *latitude,longitude,name* and *image*. When the instances are ready we call the method `addMapObject()` to add the instance on the Map.

```
function addPushPin( latitude, longitude,name,friendImage)
{
            count = map.markers.length
            counter++
     component = Qt.createComponent("PushPin.qml");
    console.log(" Call "+component.status + " " + Component.Null);
    //create pushpin object
     object = component.createObject(container,
                                    { "latid": latitude,  "longit":
longitude,"name":name, "friendImage":friendImage}
                                    );
    if (object.status == Component.Ready)
    {
        console.log("Component ready");
        map.addMapObject(object);
         myArray = new Array()
                for (var i = 0; i<count; i++){
                    myArray.push(map.markers[i])
                }
                myArray.push(object)
                map.markers = myArray
    }
    else // when status is ready add the pushpin on the map
        object.statusChanged.connect(addRemoteImages);
}
//add pushpn on the map
function addRemoteImages()
{
   //     console.log("addRemoteImages");
        map.addMapObject(object);
            myArray = new Array()
                for (var i = 0; i<count; i++){
                    myArray.push(map.markers[i])
                }
                myArray.push(object)
                map.markers = myArray
}
```

We also put the entire pushpins object into an *Array*, so that we can have a track of all the pushpin plotted on the Map. So when ever user switches from Facebook friends List View to Map View it will display the friends location on the Nokia Map.

Summary

This article shows the use of Nokia Map API along with Facebook API. It also talks about how to create a Map object dynamically using Dynamic Object Management in QML.

Source Code

The full source code of the example is available here: File:FacebookFriends NokiaMap.zip

Related Article

- Google Landmarks on a Nokia Map using Qt Quick

Note: We will also come up with the same article on Windows Phone 7 . Please share your feedback to improvise the same.

(C) Copyright Nokia 2014. All rights reserved.