Google Map Markers and QML

This page is about using QML Map to rapidly create an application that displays markers on the Map.

The QML Map element is very basic. Instead, I wrote an extension that is described here : QML Map extension.

Using this Map, it is easy to use an existing Google Map Marker XML service. For this example, I use the Paris bicycle rental service, Velib.

Model

The first XML service is found here. As you can see, it is simply a list of markers containing an id and coordinates. It references the 1000+ stations.

This is a typical example for using the XMLListModel. It goes like this:

```
    XmlListModel {
        id: stations
        source: "http://www.velib.paris.fr/service/carto"
        XmlRole {name: "number"; query: "@number/string()"}
        XmlRole {name: "name"; query: "@name/string()"}
        XmlRole {name: "lat"; query: "@lat/string()"}
        XmlRole {name: "lng"; query: "@lng/string()"}
    }
```

One thing to notice: there is no query yet. The query is to be added dynamically later, to include a filter on the lat / lng values, in order to avoid displaying to much elements.

The second service is http://www.velib.paris.fr/service/stationdetails/ + number, the number being given by the first service. A second model uses the second service:

```
    XmlListModel {
        id: station
        query: "/station"
        XmlRole {name: "freeVal"; query: "free/string()"}
        XmlRole {name: "availableVal"; query: "available/string()"}
    }
```

This one does not have source, as the source depends on the number of the marker selected.

Map

This model can directly be used in the MyMap object, like this

```
    MyMap {
        anchors.fill: parent
        id: stationsMap
        opacity: 0
        zoomLevel: 14

        itemMapDelegate: // To be expained later
        itemsModel: stations
        onViewportChanged: {
            stations.query = "/carto/markers/marker[@lat < " + from.latitude + " and
@lat > " + to.latitude
                    + " and @lng < " + to.longitude + " and @lng > " + from.longitude +
```


(C) Copyright Nokia 2014. All rights reserved.

```
"]"
        }
    }
```

Here again, pretty simple. As said earlier, the query is set when the viewport on the map changes. "from" and "to" are coordinates defining the corners of the viewport. The query filters the stations that are to be displayed.

Marker

The itemMapDelegate is a little more complex. It uses another QML object, ThumbItem, that basically draws an icon on the map, and call a signal when clicked.

```
Item {
    signal selected(int index)
    property variant coord: Coordinate {
        latitude: lat
        longitude: lng
    }
    x: map.toScreenPosition(coord).x + map.anchors.topMargin
    y: map.toScreenPosition(coord).y + map.anchors.leftMargin
    Image {
        anchors.left: icon.left
        anchors.bottom: parent.top
        source: "thumb_velib_shadow.png"
    }
    Image {
        id: icon
        anchors.horizontalCenter: parent.left
        anchors.bottom: parent.top
        source: (name != stationName) ? "thumb_velib.png" : "thumb_velib_select.png"
        MouseArea {
            anchors.fill: parent
            onClicked: {
                selected(index)
            }
        }
    }
}
```

"stationName" is a string defined in the main file.

The use of this object mostly need a function to receive the signal, so the definition of the itemMapDelegate is as follows:

```
        itemMapDelegate: ThumbItem {
            onSelected: {
                stationName = stations.get(index).name
                station.source = "http://www.velib.paris.fr/service/stationdetails/" +
stations.get(index).number
            }
        }
```

The "station" model now hold data about the station (free parking spots and available bikes).

Details of a station

(C) Copyright Nokia 2014. All rights reserved.

The station model is used via a repeater. The point of using a repeater is that it disappears is the model is empty.

```
    Repeater {
        model: station
        delegate: Label {
            text: stationName + "\nFree: " + freeVal + "\nAvailable: " + availableVal
        }
    }
```

Label being simply a QML component that displays a text.

And voilà. Less than 50 lines for the main files.

(C) Copyright Nokia 2014. All rights reserved.