Dynamic objects Adding/removal with Qt Quick Map API

With QML Map element you can add following items to the Map element:

- MapPolygon: for general polygon objects
- MapPolyline: for general polyline objects
- MapCircle: for Circle objects (polygon)
- MapRectangle: for Rectangle objects (polygon)
- MapText: for textual map objects
- MapImage: for Icon (marker/pushpin) objects.

Basically if you know the elements to be shown on coding time, you can easily define the objects inside the Map element and use the visible variable to set them hidden/visible. This would mean that no actual dynamic handling is required. Here's simple code example on toggling the visibility of a polyline objec

```
if(polylines.visible)
    polylines.visible = false;
else
    polylines.visible = true;
```

Full examples on this kind of content handling can be found from QML SimpleMapContent.zip and QML MoreMapContent.zip examples.

You can also manage map objects in groups by using the MapGroup element. Note though that with some versions there is a bug that prevents the visible variable working for MapGroup. The workaround is to use the objects array, and to loop the items inside the MapGroup one by one and set their visibility individually:

```
for (var i = 0; i < icons.objects.length; ++i) {
     icons.objects[i].visible = icons.visible;
}
```

Full example for the workaround is also shown in QML SimpleMapContent.zip example with MapImages inside a MapGroup.

If the objects need to be specified on run time, or if real dynamic handling of the content objects is required, you can use the addMapObject/removeMapObject functions defined for the Map element. Already added objects should be visible in objects, though note a bug effecting some versions, where the objects is not defined, and thus can not be used.

One possible workaround for the missing objects array, is to define own javascript array and use it to track the objects added to the Map. For example adding a new circle object into the Map could be handled as follows:

```
var newCircle = Qt.createQmlObject('import QtQuick 1.1;import QtMobility.location 1.2; MapCircle {color: "#40FF0000"; radius: 3000;
newCircle.center = map.toCoordinate(Qt.point(mouse.x,(mouse.y - titleBar.height)));

Script.myCirclesArray.push(newCircle);
map.addMapObject(newCircle);
```

Removing of the object could then be handled for example like this

```
map.removeMapObject(Script.myCirclesArray[__SelectedCircle]);
Script.myCirclesArray.splice(__SelectedCircle,1)
```

With the snippeds the "Script.myCirclesArray" is javascript array. Full example for the workaround can be found from QML DynamicPolyline.zip example.