Interactive Metro Map for Moscow and St.-Petersburg

Nikita Karpinsky, Evgeny Linsky and Alexander Malakhov
State University of Aerospace Instrumentation (SUAI lab)
Problem statement

- MeeGo and Symbian^1 support
- Map dragging and scaling implementation
- Pinch-to-zoom for MeeGo support
- UI is designed in QML, program logic is in C++
- Shortest route calculation
- Several maps supported

Main problem: fast map rendering
Rendering map as a single image

Map is an image loaded in QML.

```qml
Flickable {
    function recount() {
        map.height = default_height * zoom_value;
        map.width = default_width * zoom_value;
    }

    Image {
        id: map
        source: "spb_map.jpg"
    }
}
```

Dragging is very slow.
Rendering map as multitude of images

Images replace each other without being zoomed.

```javascript
Flickable {
  function recount() {
    map.source = "spb_map_" + zoom_value + "_.jpg";
  }
  Image {
    id: map
    source: "spb_map_1.jpg"
  }
}

Scaling is discrete.
```
Rendering map as visible part of vector graphic

Map is a QDeclarativeItem, only visible part is rendered.

```cpp
void paint(QPainter *painter, /*other arguments*/) {
    QRectF rect = boundingRect();
    for (int i = 0; i < edges.count(); ++i) {
        if (rect.contains(edges[i].x1, edges[i].y1) ||
            rect.contains(edges[i].x2, edges[i].y2))
            painter->drawLine(edges[i].x1, edges[i].y1,
                              edges[i].x2, edges[i].y2);
    }
}
```

Dragging is very slow.
Creating QPixmap once for each map scale

Pixmap is created for each scale. While dragging the pixmap is drawn on the painter.

```cpp
void createPixmap()
{
    pixmap = new QPixmap(width, height);
    QPainter* painter = new QPainter(pixmap);
    //draw map
}

void paint(QPainter *painter, /*other arguments*/) {
    QRectF rect = boundingRect();
    painter->drawPixmap(rect, *pixmap, rect);
}
```

Dragging is fast.
Creating pixmap using Qt SVG

Instead of using drawing functions an SVG image is used.

```cpp
void createPixmap()
{
    pixmap = new QPixmap(width, height);
    QPainter* painter = new QPainter(pixmap);
    QGraphicsSvgItem* svg = new QGraphicsSvgItem("path");
    svg->paint(painter, option);
}
```

It works more slowly than previous method.
Rendering times

(X + Y) means rendering times in QML + rendering times in C++. Times for QML are very inaccurate.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dragging</th>
<th>Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single image</td>
<td>(10+0)ms</td>
<td>(10+0)ms</td>
</tr>
<tr>
<td>Multitude of images</td>
<td>(4+0)ms</td>
<td>(6+0)ms</td>
</tr>
<tr>
<td>Visible part is rendered in C++</td>
<td>(3+20)ms</td>
<td>(3+20)ms</td>
</tr>
<tr>
<td>Creating pixmap for each scale</td>
<td>(3+0.5)ms</td>
<td>(3+60)ms</td>
</tr>
<tr>
<td>Using Qt SVG</td>
<td>(3+0.5)ms</td>
<td>(3+230)ms</td>
</tr>
</tbody>
</table>