Jolla Code Camp 2014 Report

Group 7:

Chandra Satriana (0424786)
Md. Mohaimenul Hossain (0424689)
Alifia Fithritama (0424663)

1. Introduction

The theme for Jolla Code camp 2014 is about Open Data, so we decided to make an location-based application using Open Street Map. We named our application **"Rate My Place!"**. The idea behind this application came up from our very own experience as Erasmus Mundus Students who are moving from one country to another every 6 months or so. Every time we are moving to a new place, we need to prepare many things, and one of them is about finding an apartment. Without any prior knowledge about the new places, finding a good apartment can turn into an overwhelming experience. Therefore, we would like to develop a user-friendly application that provides an integrated solution so the user can find all the information they need in all in one place. Our targeted user is anyone who wants to search for a living place. We hope that our application will be useful for people who are moving a lot and have limited time to search about apartment information, for example: exchange students, travelers, or businessmen.

Having the idea in mind, we started to transform our idea into more concrete form. We made a list of all contained features to begin with. First, we want the application to be global, so it can be used by anyone anywhere in the world. To do so, user should be able to input any address that he/she wants and the application should be able to find this address, therefore we need to include map of the entire globe. Next, we determined the 7 location parameters (bus stops, shops, atm, university, city center, restaurant, and hospitals) used to define a good apartment. The lower the distance of an apartment to these location parameters, the better it is. After that, we decided how to calculate the rating of an apartment. We came up with a 5-stars based rating to make it easier for the user to understand the result.

To help the user determine the quality of the apartment, we calculate a rating (1 star is lowest, 5 stars is highest) based on the distance criteria. The nearest the distance from the 7 places, the higher the rating will be. If distance is within 300 meters, score will be 5 stars. Within 500 meters, score will be 4 stars. Within 700 meters, score will be 3 stars. Within 1 km, score will be 2 stars. And lastly, within 2 km, score will be 1 star. After getting 7 individual scores, finally we calculate the total score using simple average of those individual scores. For example, an apartment which has bus stops, shops, atm, university, city center, restaurant, and hospitals in 300 meters distance will got highest rate of 5 stars because each of individual scores are also 5 stars. However, we put some exception that if the individual score of bus stop is 5 stars (meaning: there is a bus stop within 300 meter distance from apartment), then the total score will get at least 3 stars. We made an assumption that as long as there is a bus top nearby, it is basically easy to go almost anywhere, therefore the bus stop has the highest priority among all.

2. Implementation

In this section we are going to explain about some codes that we are using in the application and also the User Interface of the application. The implementation steps are as follows: (1) Extract user latitude-longitude information of from Nominatim API, (2) Find the nearest distance of 7 parameters from location, (3) Calculate the rating based on the found distance, (4) Design the User Interface to navigate the user from one page to another.

2.1. Retrieving Coordinate of an address

The address which is inputted by the user on the home screen will be translated into a coordinate (latitude and longitude). To get this information, we use the API of Open Street Map, which is called Nominatim. The query to the API can be returned in an XML format. An example of XML data returned from a query will be such as:

```
<searchresults timestamp="Mon, 03 Mar 14 17:44:12 +0000" attribution="Data ©
OpenStreetMap contributors, ODbL 1.0.
http://www.openstreetmap.org/copyright"querystring="karankokatu
4,lappeenranta" viewbox="28.13,61.05,28.14,61.04" polygon="false" exclude_pla
ce_ids="59553626"more_url="http://nominatim.openstreetmap.org/search?format=x
ml&exclude_place_ids=59553626&accept-language=id,en-
US;q=0.8,en;q=0.6,fr;q=0.4,ms;q=0.2,fi;q=0.2&viewbox=28.13%2C61.05%2C28.14%2C
61.04&q=karankokatu+4%2Clappeenranta">

<place place_id="59553626" osm_type="way" osm_id="82497158" place_rank="30" b
oundingbox="61.0489463806152,61.0496063232422,28.1347599029541,28.13638114929
2"lat="61.04927585" lon="28.1357250142573" display_name="LOAS Karankokatu, 4,
Karankokatu, Kourula, Lappeenranta, South Karelia, Southern Finland, 53810,
Finlandia"class="place" type="house" importance="0.501"/>

</searchresults>
```

So firstly, a HTTP request is made using the QNetworkRequest, and parsing the returned XML using the QDomDocument. Below is the code written in C++:

```
QString urlString =
"http://nominatim.openstreetmap.org/search.php?q="+address+"&viewbox=28.13$2C
61.05$2C28.14$2C61.04&format=xml";

    //Getting the XML for the user's address
    manager = new QNetworkAccessManager(this);
    reply = manager->get(QNetworkRequest(urlString));
    QEventLoop loop;
    connect(reply, SIGNAL(finished()), &loop, SLOT(quit()));
    loop.exec();


    //Parsing the XML to get the coordinate of the user's address
    QDomDocument *doc = new QDomDocument();
    doc->setContent(reply);
```

```
    //Get the root element
      QDomElement docElem = doc->documentElement();
      QDomNodeList nodeList = docElem.elementsByTagName("place");

      //Check each node one by one.
          //for(int ii = 0;ii < nodeList.count(); ii++)
      for(int ii = 0;ii < 1; ii++)
          {
              QDomElement el = nodeList.at(ii).toElement();
              latit = el.attribute("lat");
              longit = el.attribute("lon");
          }

      setLatLong(latit+"|"+longit);
```

2.2. Finding the parameters (ATMs, Hospitals, etc) closest to the address

To get the nearest ATMs for example, we use another Open Street Map API, which is called Overpass, located at overpass.osm.rambler.ru. As there is no possibility of getting the distance from the user's location to the ATM, we need to make the query in several iterations. First, we do it for within 300m, then if not found we'll have to query for within 500m, and so on until 2000m. Below is the code:

```
//within 500m
            //1. Getting ATM list
            QString atmString =
"(node(around:500,"+latit+","+longit+")[\"amenity\"=\"atm\"];>;);out;";
            atmString =
"http://overpass.osm.rambler.ru/cgi/interpreter?data="+atmString;
            //Getting the XML
            //manager = new QNetworkAccessManager(this);
            reply = manager->get(QNetworkRequest(atmString));
            connect(reply, SIGNAL(finished()), &loop, SLOT(quit()));
            loop.exec();

              doc->setContent(reply);
            //Get the root element
              docElem = doc->documentElement();
              nodeList = docElem.elementsByTagName("node");
              if (nodeList.count()>0){
                  //setAddressToRate("ada di <500m");
                  theATM.score = "4";
                  theATM.distance="500";

                //  setAddressToRate("Score"+theATM.score);
                  for(int ii=0;ii<nodeList.count();ii++){
                      // get the current one as QDomElement
                      QDomElement el = nodeList.at(ii).toElement();

                      atmResult =
atmResult+"$"+"lat:"+el.attribute("lat")+"|lon:"+el.attribute("lon");
                  }
                  atmResult=atmResult+"|score:4|distance:500";
                  atmResulti=4;
```

2.3. Rating the place

To rate the place, we use a simple algorithm by weighing the same factor for every parameter, but when the ATM is very close (given a score of 5), we make sure that the final overall score will not be less than 3 stars. Below is the code:

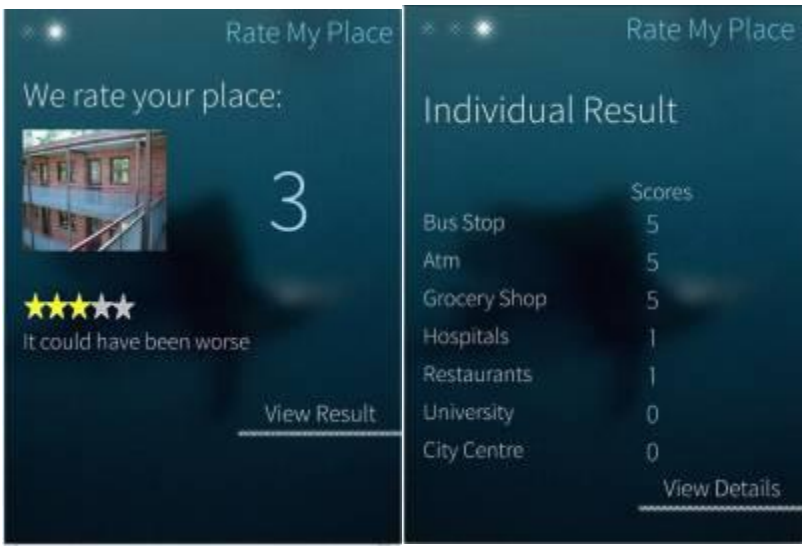
```
totalz = totalz+atmResulti + municipalityResulti+restaurantResulti +
busStopResulti + universityResulti+ smarketResulti + hospitalResulti;

totalz = qRound (totalz/7.0);

if(busStopResulti==5)
{if(totalz<3){totalz=3;}}
```

2.4. User Interface

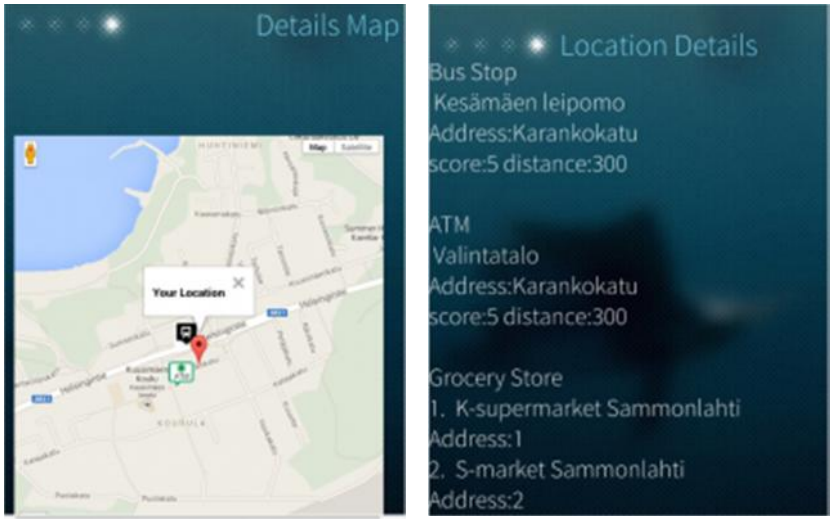
When we are planning to build **"Rate My Place!"** we wanted to make this application to be as user-friendly as possible so that anyone can use it without difficulty. Therefore, while we are designing the GUI our main goal was to build the interface simple and easy to operate.

When user clicks the cover page, it will direct to the first page of the app where user can enter name of their city and name of the location they are looking for. After that, user clicks the **"Rate My Place!"** button. As we have used different parameters to rate the place, it will take some time to get the result. During this time there will be a progress indicator showing that the system is still busy.

Next, there will be a new page where user can see the image of the searched place. They can also see the marking that we gave and how many stars that we gave to that place. There is also a predefined verdict according to the score, which will help user to understand the result even better and in a funny way. And in the bottom there is "view result" button which will lead to a new page. In this new page Individual result of every parameter like bus stops, shops, atm, university, city center, restaurant, and hospitals will be shown.



In the bottom there are two separate buttons: "View Map" and "View Details". "View Map" will lead to a new page that will show the user his position and all the nearby results in the map. It is a static map. But it uses some colorful user friendly markers which will be easy to understand for the user. Another button is "View Details". If users press this button it will lead to a new page where they can see all the detailed results of nearby bus stops, shops, atm, university, city center, restaurant, and hospitals with their names and addresses.



3. Conclusion

Build a Jolla application is a new learning experience for us. Being a newbie in developing Jolla application, we encountered some difficulties here and there. The main difficulty is to get the XML data from the Open Street Map. Later we found that it is not possible to use Javascript to do it, so we have to use C++ code. Also, we found that the running SDK gives a lot of burden to our computers and often make them hung, as a matter of fact one of our team member must uninstalled it and then installed it again because his computer crashed. However on top of those hard times, thanks to supportive mentor and the code camp spirit that we share, everybody is willing to help each other to solve the problem. We are grateful to have an opportunity to take a part in Jolla Code Camp 2014. We met new friends, we learned new things, we had a lot of fun, and of course we treasured priceless memories. We hope people will find our **"Rate My Place!"** application useful and really use it in their daily life. For us, there is no bigger appreciation than an honest acknowledgement from sincere users. Last but not least, we thank Lappeenranta University of Technology (LUT) and Jolla for organizing the Code camp 2014 and we wish a very good luck for the upcoming code camp events in the future.

4. References

Sailfish OS: https://sailfishos.org/

Oracle Virtual Box: https://www.virtualbox.org

QT Creator IDE: https://qt-project.org/wiki/Category:Tools::QtCreator

JavaScript: https://developer.mozilla.org/en/docs/Web/JavaScript

C++: https://www.cplusplus.com

Open Street Map : http://nominatim.openstreetmap.org/

Open Street Map Overpass API: http://wiki.openstreetmap.org/wiki/Overpass_API

Overpass Turbo: http://overpass-turbo.eu/

Jolla Code camp 2014: http://codecamp.fi/doku.php/jolla2014/start/