LEBANESE AMERICAN UNIVERSITY
Electrical and Computer Engineering Dept

COE 312
Data Structures

Fall 2012

W. FAWAZ

Projects I

Project I (Due date: Tuesday November 06, 2012)

I. Objective

In this project, you are tasked with developing a software tool that uses a USGS (US Geolgical Survey) feed to display a map showing the locations of the latest earthquakes in the world. More specifically, your tool will target those earthquakes that happened over the course of the past seven days with a magnitude of 2.5 or higher. This somewhat complicated task will be decomposed into three smaller manageable subtasks to make the process of developing this "earthquake viewer tool" simpler.

The proposed implementation strategy is built upon two main pillars, namely the use of a "divide and conquer"-like approach to solve the problem and the gradual integration of features into the earthquake viewer. The details pertaining to the first of the three tasks are provided next.

II. First task

The basic information about each earthquake is made available by USGS in the form of an XML feed. This XML-formatted feed can be obtained from: http://earthquake.usgs.gov/earthquakes/catalogs/eqs7day-M2.5.xml. So, your first task is to connect to that earthquake feed and then parse it by extracting information about the location of each earthquake.

Obviously, to accomplish this first task, you are required to use the DOM parser to fetch the desired information from the retrieved 7-day earthquake feed. It goes without saying that this can be achieved only after you study carefully the structure of the XML feed with the purpose of identifying the tags enclosing the location-related information. Once you have successfully extracted the information, you would be ready to tackle the second task whose guidelines are delineated in the following section.

III. Second task

Given that the information produced by the first task corresponds to an address, your second task would be to use the Google Geocoding API to convert that address into geographic coordinates (like latitude 37.423021 and longitude-122.083739). The resulting coordinates will serve as an input for the third task where you will use the coordinates to add a marker to a map.

Translating an address to a pair of geographic coordinates involves using the Google Geocoding API that provides a means for accessing a geocoder as explained at:

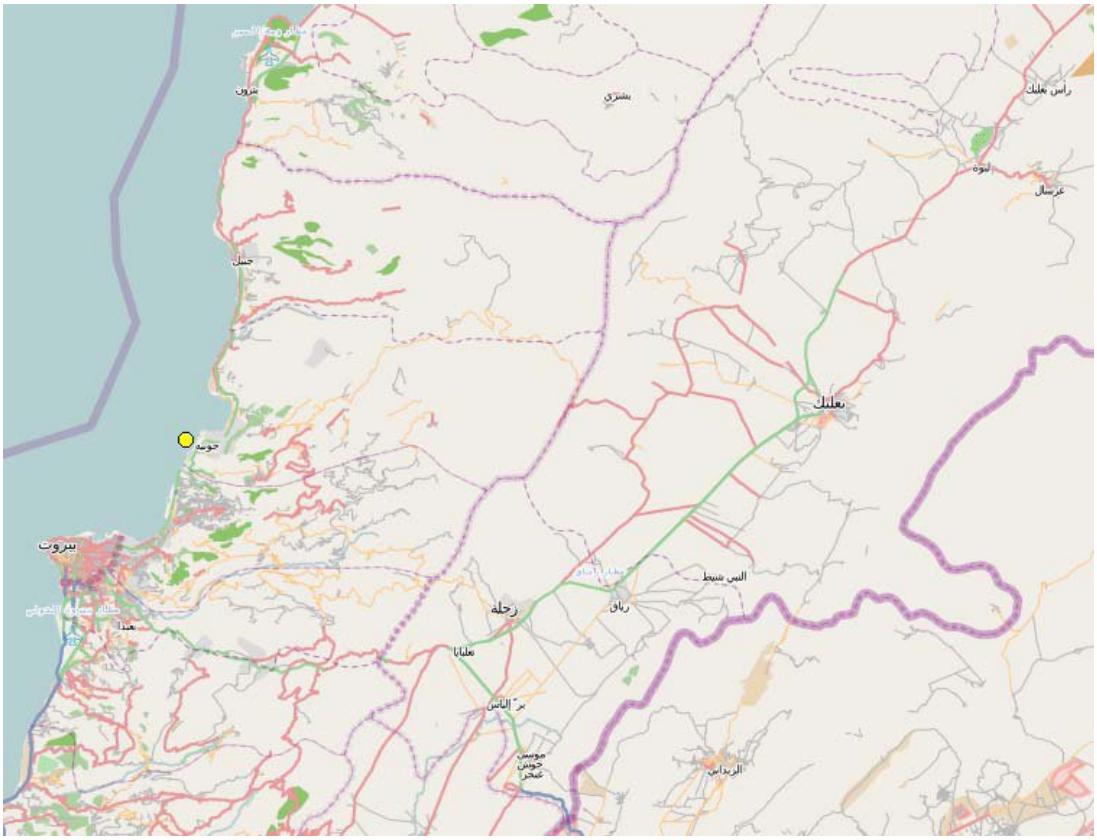
https://developers.google.com/maps/documentation/geocoding/

As illustrated in this webpage, a JSON object containing the geographic coordinates can be requested from the geocoding API. Although it is possible to get an XML version of the coordinates; for this task, a JSON version of the coordinates should be requested from the API. Once the JSON-formatted coordinates has been received your next step would be to extract the latitude and longitude coordinates wrapped inside the JSON object. Armed with these two pieces of information, you will be able to proceed to the third task of this project.

Given that the first task will result in multiple address values, the process detailed above should be repeated for each of these addresses.

IV. Third task

At this stage, you are assumed to have completed the second task correctly. This is particularly true since the third task will make heavy use of the latitude and longitude information generated by the second task. Your task at this point is to integrate into your Java application a map with a set of markers showing the locations associated with the latitude and longitude coordinates resulting from the second task. For illustration purposes, I am enclosing below a map view displaying a yellow marker that points to the city of Jounieh, Lebanon.



You have to produce a similar view but with several markers pinpointing the positions of the earthquakes on the map.

To add a map to your java application, you can use the JMapViewer java component. The following URL: http://wiki.openstreetmap.org/wiki/JMapViewer is a good source of information in this regard. The external library (.jar file) that contains the JMapViewer java class can be downloaded from:

http://svn.openstreetmap.org/applications/viewer/jmapviewer/releases/2011-02-19/JMapViewer.zip

Moreover, the HTML documentation for the JMapViewer class and other relevant classes can be found at: http://josm.openstreetmap.de/doc/. This URL is very important as it will allow you to gain a deeper understanding of both the JMapViewer class and the other classes that might be needed to accomplish the third task.

Finally, to complete the third task, add an event handler to your application to allow for a periodic refreshing of the earthquake view. In this way, the map can be updated at regular time intervals with new markers pointing to the locations of the recently occurring earthquakes, that is, the ones that took place after the last update of the map view. As you might expect, this final step requires that a Timer object be created and incorporated into your application.

What to turn in?

This project is due at the beginning of class on the due date. You have to turn in the following material in both hard and soft copies.

Criteria	Percentage
Documentation of your solution including explanations and illustrations in one or two pages along with short write-up of questions and/or problems that you encountered while doing this assignment.	2 pts (10%)
Source code that contains an appropriate amount of comments. Well-organized and correct code receives 15 pts, messy yet working code receives 8 pts, code with bugs receives 2 pts, and incomplete code receives 1 pt.	16 pts (80 %)
Execution output such as a snapshot of the contents of standard output. A correct output receives 3 pts, the one with minor errors receives 2 pts, and an incomplete output receives 1 pt.	2 pts (10%)
Total	20pts (100%)

Good Luck!