Lab 11

Reminder: This lab is one of the graded labs worth more points.

1. **SUMMARY**
   1. You will be calling a weather web service that provides current weather info for a given city and comparing the weather between two cities the user provides.
   2. This lab will involve the following new features:
      1. Consuming a REST web service.
      2. Utilizing Google’s Gson library to convert json to java objects.
2. **DETAILS**
   1. Create a normal java application.
   2. Download the gson jar file provided with this lab.
   3. First create the java classes that the gson library will fill with incoming json data.
      1. These java classes will model the json that is returned from the weather web service.
      2. Here is what the returned json looks like:

{

"coord":{"lon":-116.96,"lat":32.79},

"weather":[

{"id":800,"main":"Clear","description":"clear sky","icon":"02d"}

],

"base":"stations",

"main":{

"temp":71.6,

"pressure":1019,

"humidity":14,

"temp\_min":68,

"temp\_max":75.2

},

"visibility":16093,

"wind":{"speed":2.95,"deg":234},

"clouds":{"all":5},

"dt":1479581760,

"sys":{

"type":1,

"id":476,

"message":0.1939,

"country":"US",

"sunrise":1479565376,

"sunset":1479602637

},

"id":5345529,

"name":"El Cajon",

"cod":200

}

* + 1. All we care about is getting the value “name” and values from “main” like “temp” “temp\_min” and “temp\_max”.
    2. So we will 2 create java classes to mirror this relationship:

**IMPORTANT**: Remember that you can add a package from the File > New menu. All these files go in “com.grossmont.ws”. When you create a new class, just right click on the package folder you want to add the class to.

NOTE: You’ll notice that the instance variable names below match the json fields passed back. That matching is necessary for gson to be able to map the json to your fields.

* + 1. **Weather\_main.java**
       1. This is just the “main” object portion of the json that holds the below fields.
       2. Package: **com.grossmont.ws**
       3. Create five public instance variables:
          1. All are **float** type -- **temp, pressure, humidity, temp\_min, temp\_max**

You’ll notice that in the above json sample, there is a mapping of main and inside main are these five variable names.

* + 1. **Weather.java**
       1. This is the root object that holds the top most fields in the returned json.
       2. Package: **com.grossmont.ws**
       3. Create two public instance variables:
          1. **main** – this is Weather\_main type that you built above.
          2. **name** – this is String type.
  1. **WeatherServiceManager.java** (this class was provided to you with this lab doc).
     1. You need to put this file in the src/com/grossmont/ws folder in your project.
     2. You will fill in code where you see the ######## markings:
        1. Simply follow the guidelines in the java file.
     3. You are essentially wiring in the classes you just created to methods that retrieve data from them.
  2. **SCREENSHOT**: Please take a screenshot of the console print out at the bottom of your IntelliJ after you run it ~ OR ~ If you’re taking on the Extra Credit version below, then take a screenshot of both JSP pages running in the browser.

**IMPORTANT: You can only get a perfect score if you provide a screenshot**.