Eric Feldman

CSS 436 – Cloud Computing

Assignment 2 – API Calls

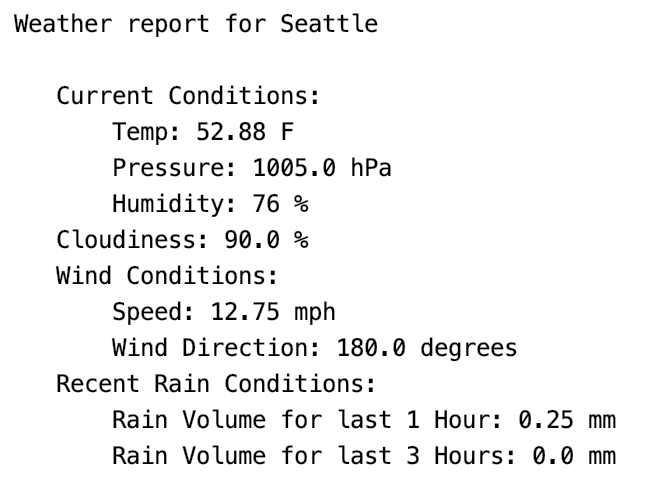
October 20, 2019

Application, Design, and Usage

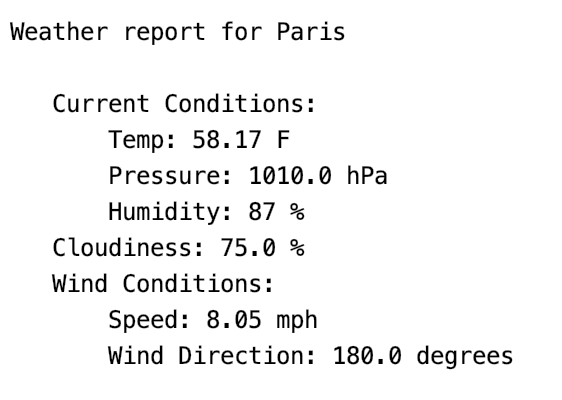
For this REST API project, I have utilized the API service called OpenWeather. This company provides various API tiers for their customers to access. For this assignment I have taken advantage of their free tier which provides up to 60 calls per minute to a small selection of subscriptions. The application I have designed allows for the user to pass a single command line argument to the user specifying the city name for which they would like to retrieve some weather data. If the name has more than one word in it, such as Hong Kong make sure to include quotes around the entire name so the program knows that all words are part of the same name. For example, Hong Kong should be entered as “Hong Kong”. The design of this program is mostly straight forward. Building off of assignment 1, I used Java’s HTTP Client to great a HTTP request to the API and GET the body (JSON) from the call. The call is made to OpenWeather’s endpoint *currentWeatherAPI* using my private key and specified city name from the user. The getData() method uses Gson (Google’s JSON parser) to construct a weather object representing the JSON object that was retrieved from the API. This weather object was defined by me and all the sub objects within it for which I wanted to grab data from. The data is then outputted to the user to read. Sometimes the JSON object is missing parameters for some cities if the weather is not happening. For example, if I ask for the weather in Tucson, it may not return a “rain” parameter if there is no rain there currently. So in main() I check if the objects are null (meaning they did not get created by Gson because they were missing) and only print the parameters where valid data was returned.

This app may be useful for someone who wants to retrieve some quick weather parameters for a city and implement this on their website or program. Examples are some examples of output I have generated:

**Input: Seattle**

****

**Input: Paris**

****