Levi Russell

Dr. Tucker

Java CS-372

24 January 2019

Project Experience

When Stephen and I were talking about project ideas we decided we both wanted our project to use an API, mainly because we learned about HTTP access that day. The only problem was I had no idea how an API worked or how to use one. So I began researching how to use an API and how I would implement it into our project. We found a Wikipedia page with a list of almost all the API’s available for the public to use. We settled on using the Spotify API, but didn’t know what else to do. So we went to the Spotify development page and found that almost all songs on Spotify have a danceability, energy, and valence (happiness) values. We then set out to create a project that gives you music based on your mood.

Getting the API to send back a token was in itself a project. I searched for hours on Stack Overflow and Google trying to find how to set up the Spotify API to send back a token. One of the main problems was most of the documentation for using the Spotify API was either in Python or JavaScript which didn’t really help us. After a couple of days we got the API to send back a token and we thought we were home free and almost done, but we were very wrong.

We then spent the next couple of days creating the basic layout of the project and got to where Spotify would send back a Json file with the list of songs. We were able to parse it using regex but it didn’t work very well and we weren’t able to get all the data we needed. So we then started to try and use an actual Json parser. When we finally got it set up we found that the documentation on how to use it was scarce. Se we tried our best to get it to send back the data we needed and were successful except for getting data that was nested in Json objects so we asked Pete, you, how and he just pointed at the screen that what we were trying to get was not a Json object but a Json array and then it worked perfectly.

The next roadblock we ran into was getting the audio features data for thousands of songs from Spotify. We first started by having each song asked Spotify for a Json file for the audio features, but this lead to many http 429 errors which means we were asking for data too fast and too much. So we then found that you can ask for multiple song audio data by separating the song ID’s by commas and we could then get 100 song’s features in one Json file. But I ended up creating a for loop that was more complicated than it had to be and we were then not setting the right track data to the right track. I worked on this for loop for a couple of hours before I stepped back for awhile and thought it would be better to use a do while loop, and it was indeed simpler and actually worked.

We ended up being able to follow our project specification almost perfectly, except for the UML. When we created the project specification UML we had no idea how to use an API or how the data was sent back so we only had one class for the API and the data it sent back. We ended up creating two more classes one for getting the songs from Spotify using playlist ID’s we got from the most popular categories, and one to create a song object that holds the track’s features. At the end of the project we looked at the Final Project requirements and realized we never used inheritance, but then we remembered we learned in class that all java classes inherit from Object, so we were okay. Looking back at the project we used a lot of things we learned in class like Java Swing for the UI, HTTP access to communicate with Spotify, and multi-Threading for simplicity and efficiency. And after watching the class presentations it made me realize modern programming languages are very powerful and the only thing holding you back from creating what you want is yourself.