# Provided in src is the java and json files. If you would rather import the files directly from Github the link is here:

https://github.com/ty2610/Dynamic\_Messenger\_Project.git

This contains .iml and .idea files for Intellij users. If you are using a different IDE these files can be deleted.

There is a Jar file, both jars files within must be added to the dependencies of the project. The files are as follows:

1: commons-land3-3.6.jar - this is so I can use a function that counts the amount of a certain character in a string (StringUtil).

2: json-simple-1.1.1.jar - this is what I use to handle my intaking of Json files.

I decided against a whole spring framework, thus I went down the route of jars. I also developed this on Java 9, but I didn't use any java 9 exclusive content, thus I think one can get away with using a Java 8 sdk. Once the workspace is set up, run the file main, and hopefully the in program instructions makes sense from there.

# I decided to separately load each Json, and make an object exclusively for each object. I separated many functions away from main so that additional work wouldn’t have to reuse code. I allow the user to enter the attribute they are searching on and then enter an attribute value to search for. This could cause a problem if two individuals have the same name, the program will just pick the first Guest object. I make sure each input is put to lowercase to ease the user experience. If a user wants to input a dynamic message, I check that input to make sure it follows the guidelines I set. When I want to output a message, I look for the dynamic content, grab the value it is looking for, and place it where the dynamic request is in the string. I do this mostly with string manipulation using indexOf and substrings. I use many different technique throughout, some include: Lists, stream manipulation, and Json parsing.

# Java was my code of choice because I am very comfortable with it. I really enjoy the many tools at my disposal when solving a problem. I have been working with C/C++ in school a lot in the past year, so I thought it would be good to do a solo project in Java to keep the skills fresh.

# The process I took was to try to go down every user case possible to make sure that the program will run smoothly. I used every combination of selectors throughout the whole testing. I then went through every incorrect path, entering bad data, and seeing how I handle it. Finally I let my technologically challenged girlfriend use it to see if it is user friendly, and shaped some features around that.

# If I had more time, I would have loved to make it so users could edit the data in the Jsons, and add their own Message template to the Json to be used later. It would have been cool to have more attributes in the Jsons and even more Jsons.