Final Project Documentation

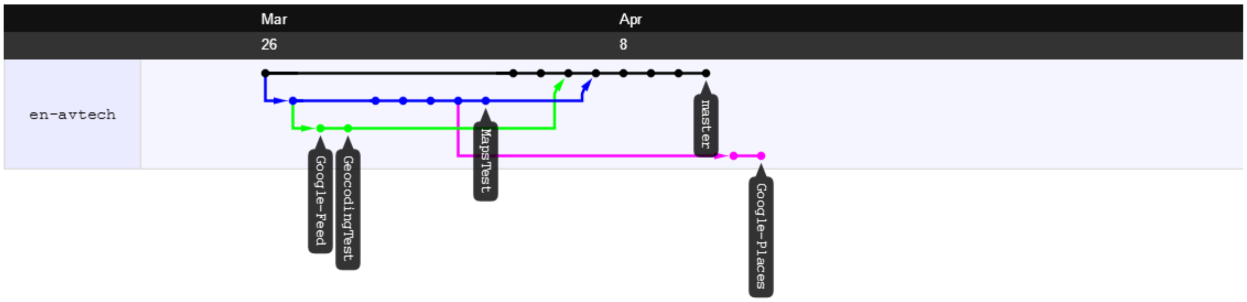
Final Project Report

This is the project report for 911-Bot, the final project for COSC 310 by Ephraim Nowak. It is a continuation of the BatterBot project, with the addition of 5 Google APIs to enhance the capabilities and feature set of the existing project.

GitHub Repository:

The link to the GitHub project for 911-Bot is located here: https://github.com/en-avtech/911-Bot

The graph below displays how different features were developed on separate branches and then merged into the master branch. Unfortunately my initially created GitHub repository became corrupted, so I had to migrate to the one linked above. The graph below is from taken from the initial repository, as it shows the most activity.



Sample Output:

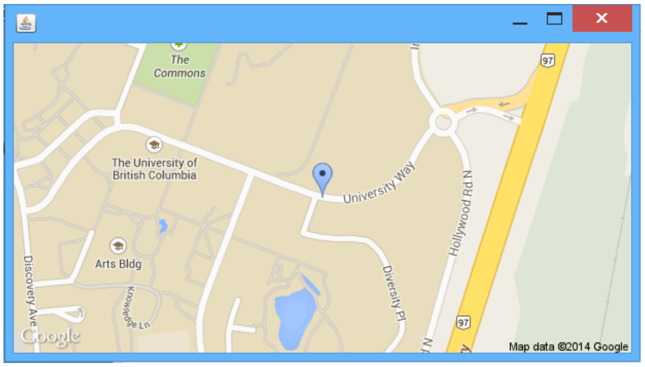
```
Connect to a server? (y/n):

n

<Dispatcher> Hello

Hi

<Dispatcher> Hi, this is Batman. Ask me about crime fighting...

But, I'm witnessing a hostage situation

<Dispatcher> Where are the hostages?

I don't know

<Dispatcher> Why are you bothering Batman with questions about that?

But this is an emergency

<Dispatcher> Do you need Fire, Ambulance, or Police?
```

Final Project Documentation

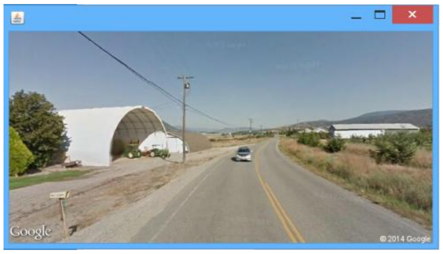
```
I need the police

<Dispatcher> OK. The police has been dispatched. What are your coordinates?

I'm not sure what my coordinates are.

<Dispatcher> At what address are you located?

3333 University Way, Kelowna

<Dispatcher> Your location in latitude/longitude coordinates is: 49.940033,-
119.39273

thanks

<Dispatcher> You're welcome.

I need help finding a place near here

<Dispatcher> What place are you looking for?

RCMP near kelowna

<Dispatcher> The nearest place is: Royal Canadian Mounted Police at 350 Doyle
Ave, Kelowna, BC, Canada

Can you give me directions to get there?

<Dispatcher> Where are you right now?

3333 University Way, Kelowna

<Dispatcher> And where would you like to go?

350 Doyle Ave, Kelowna

<Dispatcher> Head east on University Way. At the roundabout, take the  1st
exit onto Hollywood Rd N. At the roundabout, take the 2nd exit onto the BC-97
S ramp. Merge onto BC-97 S. Continue onto Hwy 97 N. Continue onto Harvey
Ave/BC-97 S. Turn  right  onto Water St. Turn right onto Doyle Ave.
Destination will be on the left.

Is the police on the way?

<Dispatcher> I'll be there as soon as I can!!

Can you show me where I am on a map?

<Dispatcher> Where are you right now?

3333 University Way, Kelowna

<Dispatcher> Getting Map from
https://maps.googleapis.com/maps/api/staticmap?center=3333+University+Way%2C+
```

Final Project Documentation

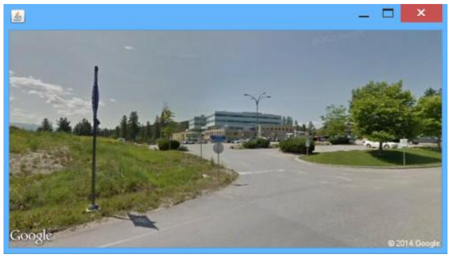
```
Kelowna&size=600x300&markers=color:blue%7C3333+University+Way%2C+Kelowna&sens
or=false&key=AIzaSyDJ8jbyWJzJQrrYGVN807xxnkfy7PlHEdk
```



```
Can you send me a Google Street View image of my location, so I can make sure
<Dispatcher> Where are you right now?

Old Vernon Road, Kelowna

<Dispatcher> Getting Image from
https://maps.googleapis.com/maps/api/streetview?size=600x300&location=Old+Ver
non+Road%2C+Kelowna&sensor=false&key=AIzaSyDJ8jbyWJzJQrrYGVN807xxnkfy7PlHEdk
```



```
<Dispatcher> Does the image match the location where you are right now
(yes/no)?

no

<Dispatcher> Where are you right now?

3333 University Way, Kelowna

<Dispatcher> Getting Image from
https://maps.googleapis.com/maps/api/streetview?size=600x300&location=3333+Un
iversity+Way%2C+Kelowna&sensor=false&key=AIzaSyDJ8jbyWJzJQrrYGVN807xxnkfy7PlH
Edk
```



Final Project Documentation

```
<Dispatcher> Does the image match the location where you are right now
(yes/no)?

<Dispatcher> OK. Please stay on the line, and we'll have help there right
away.

<Dispatcher> Why don't you ask me some questions about Batman while we wait?

Why are you a bat?

<Dispatcher> Bats are really scary!!! I fell down a well when I was a kid and
cried the bats were so scary!

Who's in the Justice League?

<Dispatcher> The core members of the Justice League are Myself, Superman,
Wonder Woman, Green Lantern, Flash, and the Martian. They are my only friends

Do you like Wonder Woman?

<Dispatcher> Yes...her and I have always had a 'thing'.  We're both in the
Justice League.

How did your parents die?

<Dispatcher> My parents died when I was  little boy. I watched them be
murdered...

Who killed them?

<Dispatcher> A bad guy! I don't want to talk about it!!! :'(

OK. Tell me about your enemies.

<Dispatcher> I have many enemies. The joker, Raz Al-gul, the Penguin, and
Two-Face are the enemies I encounter the most

Thanks! The police is here!

<Dispatcher> You're welcome.

Bye

<Dispatcher> Goodbye, Gotham Citizen
```

This conversation was a sample of the conversation which demonstrated all of the new features implemented for the final submission. This includes the Google Places, Geocoding, Directions, Street View Image, and Static Map APIs.

Feel free to try this out on your own computer. It's pretty cool!

One of the only limitations remaining in the program is the lack of memory. This is actually not really a limitation, as the code already has this feature constructed. It is just a matter of incorporating it into the conversation. This was outside the scope of this submission, but could be included quite easily with the current version of 911-Bot.

Features for API:

- The language processor has been made available as a feature. It extracts key words from input.
- The response builder, which assembles logical responses based on user input, has also been made available as a feature.
- In conjunction with the response builder, the response selector is used to determine which response to display, and more importantly when to display it. It includes features for tracking how often a response has been used, and when in a conversation to use the response to guide natural conversational flow.
- Response templates can also be made available as API's as they are used to create variable responses with substitutions available for words or phrases.
- Similarly, the scale rules used by the program can be made available as an API. They encapsulate the weighting of various keywords to ensure the most relevant response is selected.

Final Project Updates

The README file has been updated to reflect changes implemented since Assignment 3.

New features implemented are:

- **Google Places API:** Allows the user to search for places (such as RCMP detachments) near them.

  ```
  RCMP near kelowna

  <Dispatcher> The nearest place is: Royal Canadian Mounted Police at 350
  Doyle Ave, Kelowna, BC, Canada
  ```

- **Google Geocoding API:** Converts a street address to latitude and longitude coordinates, which is useful when dispatching air resources.

  ```
  <Dispatcher> At what address are you located?

  3333 University Way, Kelowna

  <Dispatcher> Your location in latitude/longitude coordinates is:
  49.940033,-119.39273
  ```

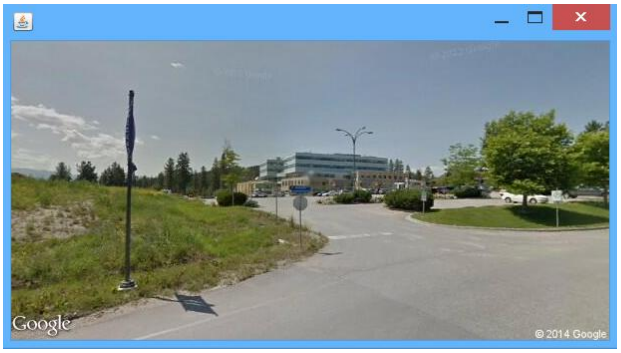
Final Project Documentation

Google Directions API: Gives the caller directions from their current location to where they are trying to go.

```
<Dispatcher> And where would you like to go?

350 Doyle Ave, Kelowna

<Dispatcher> Head east on University Way. At the roundabout, take the
1st exit onto Hollywood Rd N. At the roundabout, take the 2nd exit onto
the BC-97 S ramp. Merge onto BC-97 S. Continue onto Hwy 97 N. Continue
onto Harvey Ave/BC-97 S. Turn  right  onto Water St. Turn right onto
Doyle Ave. Destination will be on the left.
```

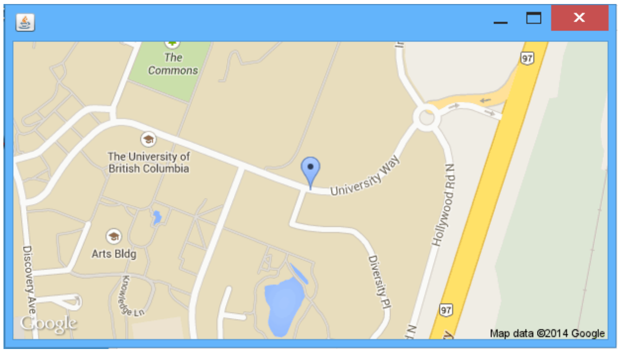
- **Google Street View Image API:** Displays a static Street View image of the caller's location in a JFrame, to allow the caller to verify if the dispatcher has the correct location.

```
<Dispatcher> Where are you right now?

3333 University Way, Kelowna

<Dispatcher> Getting Image from
https://maps.googleapis.com/maps/api/streetview?size=600x300&location=3
333+University+Way%2C+Kelowna&sensor=false&key=AIzaSyDJ8jbyWJzJQrrYGVN8
07xxnkfy7PlHEdk
```



- **Google Static Maps API:** Displays a static map of the caller's current location in a JFrame.

```
<Dispatcher> Where are you right now?

3333 University Way, Kelowna

<Dispatcher> Getting Map from
https://maps.googleapis.com/maps/api/staticmap?center=3333+University+W
ay%2C+Kelowna&size=600x300&markers=color:blue%7C3333+University+Way%2C+
Kelowna&sensor=false&key=AIzaSyDJ8jbyWJzJQrrYGVN807xxnkfy7PlHEdk
```



Final Project Documentation

README File

README - Ephraim Nowak Last Updated April 8, 2014

Purpose

The purpose of this code is to provide an advanced chat bot engine, and an example of 911-Bot.

Using the code

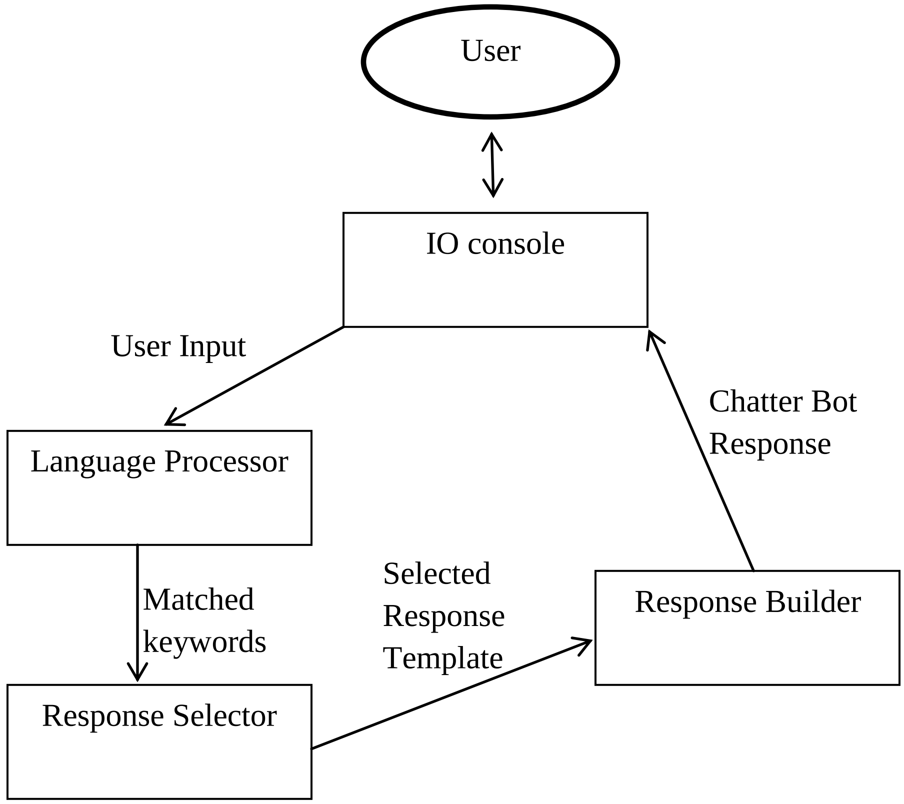
The current version of this code is only designed to be used through the eclipse console. The code can then be used by creating a project in eclipse, bringing in the relevant files, and running the BatterBotDriver class, which contains a main method which will initiate the conversation cycle. You can create your own chat bot by creating a method that will create new Response Templates, and calling the cycle() method.

Simple System Overview:

We begin by examining the flow of data by defining five main components in our system. We have the Language Processor, the Response Selector, the Response builder, and the IO controller. Another critical component of the system is the Response Template class, which we discuss more in the next section. Using these components, we can construct the basic cycle observed when the user enters an input string.

The program starts with the IO console reading the user input. This input String is the given to the Language Processor, which extracts from the input all of the recognized keywords. These matched keywords are then passed to the Response Selector. The Response Selector ranks all of the relevant response templates, selecting the one with the most matched keywords. This Response Template is then passed to the Response Builder. Each Response Template contains a String with some blanks and possible variable substitutions. The Response Builder then fills in these blanks with the variable substitutions, and hands the complete string to the IO console to be printed. The cycle then repeats.

Final Project Documentation



BatterBotDriver

We use a driver class BatterBotDriver which contains all of the components needed for the Chatter Bot. This class also stores the main function cycle() which iterates through the user input and chatter bot response cycle. Additionally, there is a method setup(), which can be used to load our sample Batter Bot Response Template files.

IO Console

The current IO class is very simple, and is simply used to read user input from the eclipse console, and then print the chatter bot responses.

IO Socket

This IO class establishes a client socket connection to YashaBot, given the host IP address and port number. It then reads the user input from the eclipse console and sends it to the server, and then prints the reply to the eclipse console.

Language Processor

Our implementation of the Language Processor is relatively simple. The Language Processor takes in the user input as a String. The Language Processor then loops through a list of all of the known keywords, and searches through the user input looking for each one. Every keyword that is found is then stored in a list, which is passed to the Response Selector.

Response Selector

The Response Selector contains a reference to a Hash Table storing pointers to all of the Response Templates. The keys for the hash table are the keywords, and the values are then the pointers to the Response Templates. This allows for quickly finding all of the relevant Response Tables based on a set of found keywords. The response selector now also uses the Java API for WordNet Synonyms (JAWS), which allows it to select responses based on synonyms of keywords.

Response Builder

Given a Response Template as input, the Response Builder fills in the canned sentence with its variable substitutions and returns the response as a string. A variable substitution may optionally come from the memTable, which contains previous user input.

MemTable

This is a dictionary data structure used to store previous user input. For example, the key may be “Name” and the value “John Doe”. When the question “What is your name?” Is asked, the response would then be parsed and stored in the memTable.

Response Template

The Response Template is a class used to store the ‘canned’ sentences and their variable substitutions. Each Response Template will contain:

- An output sentence with possible blanks to be filled in
- The Array of buckets, which contain the variable substitutions
- A list of key words that will trigger that response to be selected

Additionally, a Response Template may contain a scaleRules function definition, which allows for more complicated selection rules to be defined. This function should return a value that will multiply the normal ranking determined by the number of relevant keywords. For example, supposing that the given response should only be said once, the function scaleRules could return 1 if the response has not yet been used and a value of 0 if the response already has been used. Alternatively, this function could be used to increase the weight of the responses ranking by using a number larger than 1. This should be used with caution.

If a Response Template is asking a question of the user, for example, “What is your phone number?” then the response Template must also define and additional step for interpreting the following user input. In our example, this would mean extracting the phone number from the response. The Response Template then also needs to have a memTable key, called memEntry, which is the key that will be used to store that user response in the memTable.

KeyWordList

We have created our own class, the KeyWordList, which is used to store a list of keywords. This class is very simple, and was designed to facilitate the integration of new features in the future. Currently it simply wraps an array of Strings.

Directions

This class implements the Google Directions API. The origin and destination can be set, and the directions from the origin to the destination are printed to the console. The class also contains a main method which makes debugging much easier.

Feed

This class was used to test the Google Feeds API. THIS CLASS IS NOT USED BY 911-Bot!

Geocoding

This class implements the Google Geocoding API. The address can be set, and the latitude and longitude coordinates of the address are printed to the console. The class also contains a main method which makes debugging much easier.

Image

This class implements the Google Static Street View Image API. The address can be set, and Street View Image of the address appears in a JFrame. The class also contains a main method which makes debugging much easier.

Maps

This class implements the Google Static Maps API. The address can be set, and Google Maps image of the address appears in a JFrame. The class also contains a main method which makes debugging much easier.

Places

This class implements the Google Places API. An input search query is submitted, and the class prints the closest requested place name and address to the console. The class also contains a main method which makes debugging much easier.

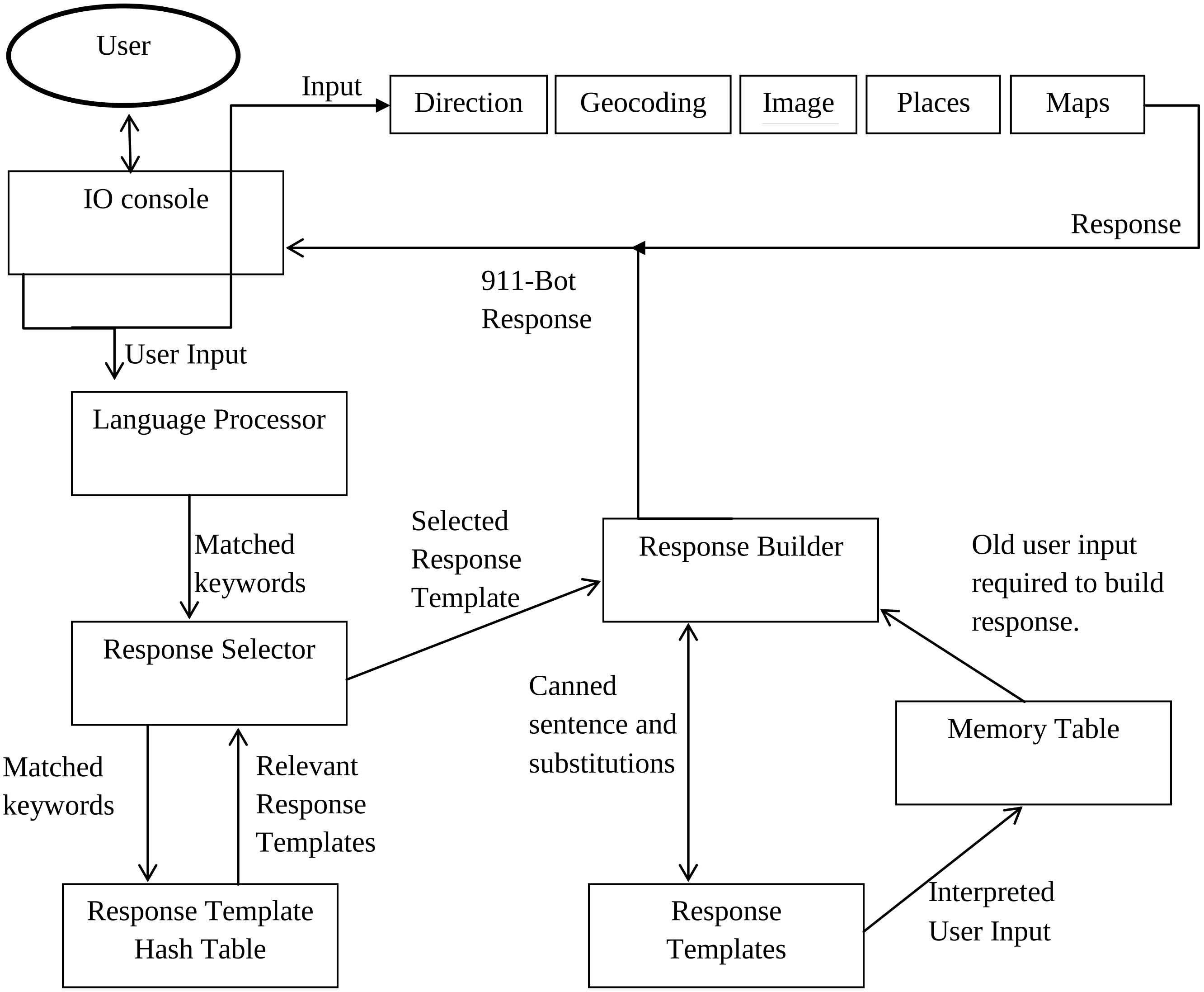
TestClass

This was a test class for the Google Feeds API. THIS CLASS IS NOT USED BY 911-Bot!

Final Project Documentation

Detailed System Diagram

A more complete diagram of the relationship between the different parts of the system can be made using the more detailed information about each class.



We note that this diagram is incomplete in that it does not show the flow of data from the IO console to the Response Template class. The previously used Response Template is also responsible for parsing the user input in the event that the user's response needs to be stored in the Memory Table.