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.

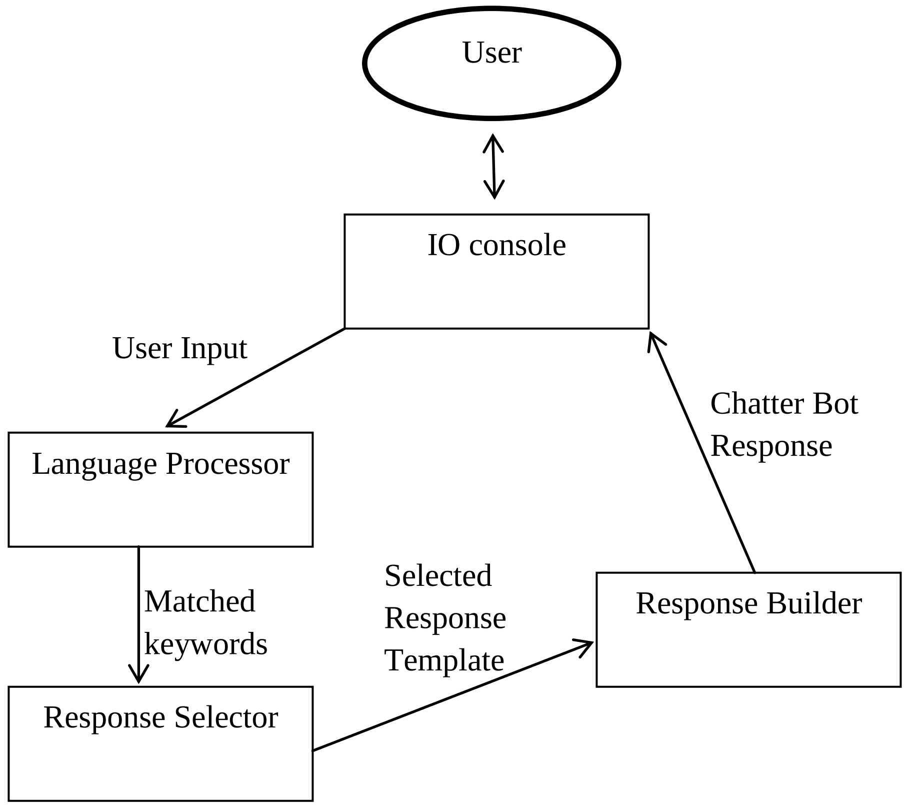
Using the code

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.



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

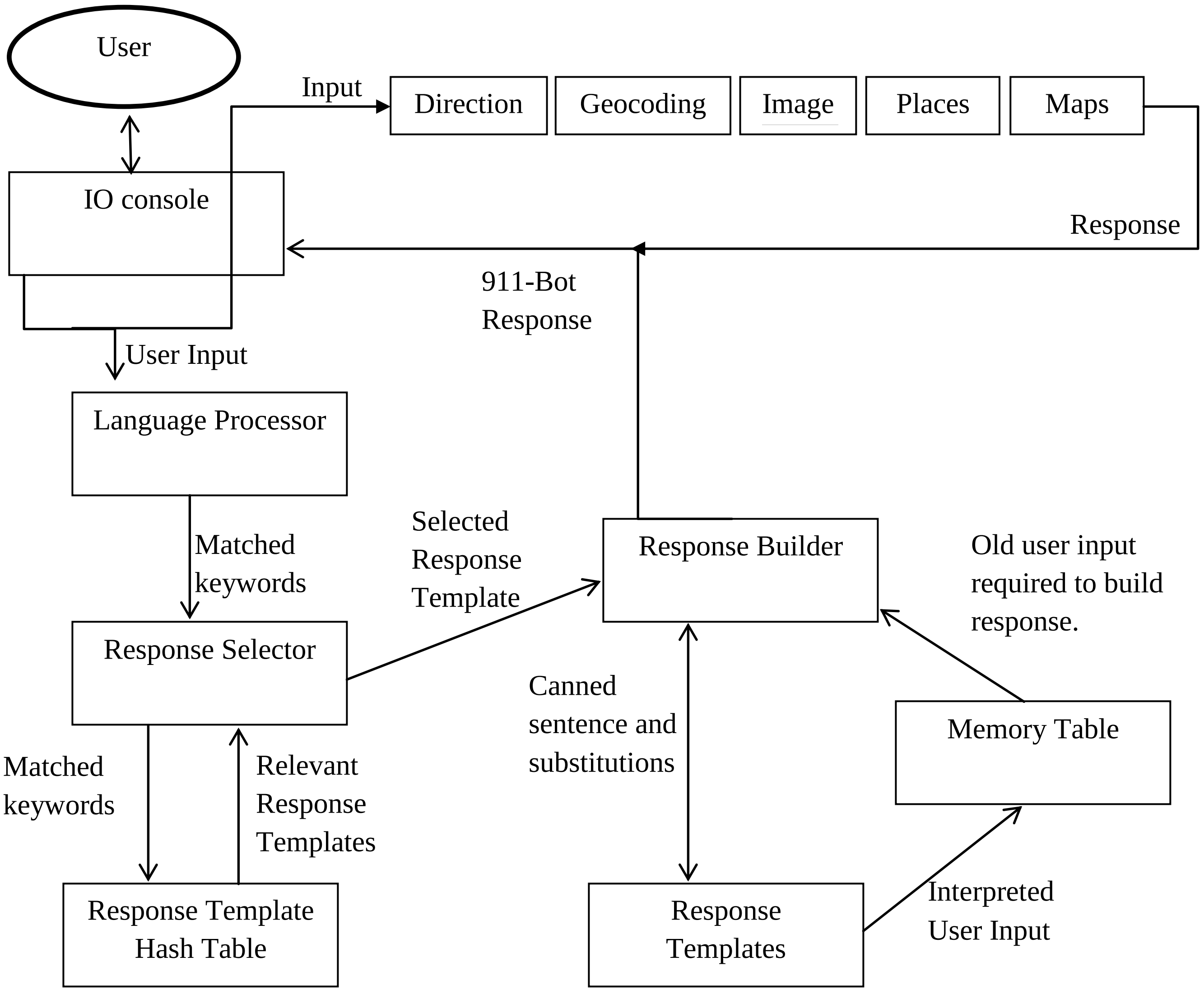
This class implements the Google Places API. An input search query is submitted, and the class prints the closest requested place name and address to the console. The class also contains a main method which makes debugging much easier.

TestClass

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

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.