# **NLU.py**

# **Overview**

This script includes two main functions: **convert\_hebrew\_number** and **parse**. The **convert\_hebrew\_number** function converts Hebrew numbers to their corresponding integers. The **parse** function uses the requests library to send a POST request to YAP API with a JSON body containing the input text, then process the response and extract information from it.

## **Requirements**

Python 3

requests library

player.py script

Additional sound files

## **Usage**

Make sure you have the required libraries installed by running pip install requests in your command line.

## Number Conversion function

This function is called "**convert\_hebrew\_number**" and it takes in one parameter, "number\_string". The function first defines a dictionary "number\_dict" that maps Hebrew number words to their corresponding integer values. The function then uses the get() method of the dictionary to look up the "number\_string" passed in as a parameter and assigns the value returned to a variable "num". The function then checks if the value of "num" is not None, if so it returns the value of "num" as an integer. If "num" is None, the function returns the input "number\_string" as it is.

## Parsing function

The **‘parse’** function takes in a string of text in Hebrew as an input and uses the requests library to send a POST request to the YAP Hebrew parsing API. The API returns a JSON object containing a dependency tree (among other things) of the input text. The function then initializes several variables including teeth, tooth, diagnosis, num\_of\_teeth, known\_symptoms (which can be modified or added easily), and free\_text.

The function then checks if the key "dep\_tree" is present in the JSON object returned by the API (aka the request went well). If it is present, the function iterates through the dependency tree using a for loop.

Within the for loop, the function checks if the part of speech (pos) of a word is "CD" (cardinal number), which would indicate a tooth number.

If a tooth number is found, the function calls the “**convert\_hebrew\_number**” function to convert the Hebrew word to its numerical equivalent in case it was not already in its numerical form.

The function then checks if more than one tooth number has been found and returns an error message if this is the case.

The function also checks for symptoms within the dependency tree and appends them to the diagnosis variable if they are found. Additionally, the function checks for free text (symptoms or diagnoses that are not common) within the dependency tree and appends it to the free\_text variable if it is found.

Finally, the function appends the tooth number, diagnosis, and free text to the teeth list and plays a success sound cue if a tooth number and at least one of the other variables have been found. If a tooth number and symptoms have not been found, the function returns an error message and plays a failure sound cue. The function then returns the list of teeth to the ‘main’ file in order to update the GUI.