Erick Galvez RUID: 189005643

Weather App GUI

Proposal

The goal of this project was to get a fully functional weather GUI running that can be executed on any computer using an executable file and give the user a real time response of the weather entered within the search bar. In order to achieve this we used python tkinter to build the frame and design of our GUI. With multiple functionality such as giving the current weather of whichever location entered, giving the user the option to change the temperature display from default, fahrenheit, to celsius, having the option to compare the weather with another api source to check for accuracy, and finally having the option to display the five day forecast of the location entered.

Description and obstacles faced

In the building process of making a shell of the application we ran into the following problems. With displaying a background image python does not always recognize a .jpeg, .gif, or .png and would result in an error unless it was pasted in paint and re-saved as whichever image file specified since that fixed formatting issues of the image file. However, when tested running on the ilabs instead of our own text editor, sublime, it would still give background image errors so that had to be taken out if needed to be tested on the ilabs and in the final executable file. When it came to formatting the output and having it all display within the frame and in the right location it took time to learn about the different ways of placing them with options like .grid(), .pack() and .place() ultimately we went with place since it was the easiest to understand how to move the frames, sizing and output.

To continue,there were errors with the display in the five day forecast in which the text will reach the bottom of the frame and would not display the rest. So we tried to make a scrollbar available for the frame that displayed the results. However, it proved more difficult than expected especially since we were using place() most solutions we tried looking online used grid() or pack() and to try to change the very foundation of the layout at this point was too much to do. We were able to just increase the size of the application display and shrink the text font a bit in order to display everything and it all came out okay.

Then, converting from fahrenheit to celsius had to be done in two different ways. The main api we use is called openweathermap and when it comes to receiving data since it returns the information in a dictionary we can had to find the key words to get the temperature in celsius or fahrenheit. However, since we use another api, Yahoo weather, from what I saw online when it comes to getting information it has certain function calls and when looking in depth into what exactly it returns it only returns the temperature in celsius so we had to use the conversion formula in that case to get the temperature in fahrenheit. To wrap up and lead to how it ties into what we learned the functionality of the button was tricky to grasp at first especially when we were only able to get it to output the first given input and not any other input that followed. After a bit of confusion we realized that we needed to use a lambda expression in order to constantly get new input.

Relating to material learned in class

When it came to things we learned about in class practicing with the dynamic imperative language python in previous homeworks made it a lot easier to jump in and not have to look up the basic such as, creating other methods and working with lists/dictionaries. Especially since the information that we were receiving from the weather api was given back as a dictionary and being able to understand the concept of how a list functions made it easy to get the information weather it be for the current weather or the five day weather forecast. Another major thing we learned from class that was vital to the program was lambda calculus since we constantly had to be getting different inputs from the user and then sending it over to the function to display a different result each time. So in our program you can see that the function that is being called to get the weather is the lambda abstraction and is bound but the parameters received by the user is free which is used in the function application. It is just as explained in the alligator egg example many lectures ago. With the variable, user input, being the egg, the alligator is the function and the alligator eating is the application, getweather(user Input) with the color changing being the result displayed.

README

The GUI will be given in an executable file to run along with a WeatherApp.py and get_icon.py file, which is used to display an icon image relating to the current weather in daily weather search, a folder with the image icons and finally this pdf. The requirements to run the code on the ilabs are listed below. To use the app properly the user can either enter a zip code and search, a city name and search and if the results are of a different than expected location after entering either the zip code or city name include a comma and the country initials. Ex- entering bergen will result in a not found display, however, entering bergen,US and bergen,NO will display the current weather in the city bergen for both United States and in Norway. If the user wants to then change the result weather all they have to do is check the checkbox then click the weather button and same goes for the comparison.

<u>Requirements to use on ilab</u>

Python3 -m pip install yahoo-weather in order for the yahoo api to work on the ilabs