Jordan Bollinger

GUI Final Project Paper

For my final project, I decided that I wanted to try to make a GUI that could figure out the time it would take someone to run a distance given their pace. My GUI has an edit box where the user can input their desired pace, as well as a set of buttons that allows the user to select whether their input was in minutes per mile or minutes per kilometer, which was all approved by Professor Lanik. Lastly, it has a dropdown menu with options for the distance that the user can choose from, 1 mile, 5 kilometers, 15 kilometers, half marathon, or a full marathon. With each of these chosen by the user, it can take that information and tell the user the amount of time it would take to run the desired distance. To begin writing the code for my GUI, I started with the main function and getting all of the necessary elements into my GUI figure. This mostly went easily, though grouping the radio buttons for the “minutes per mile” or “minutes per kilometer” was a little harder to get right. I also had to add text elements to label the interactive elements. Once I figured out the placement and how to make the elements that I needed appear correctly, I moved on to getting the callback functions written. I started with the callback function for the radio button group, which I wrote using a conditional statement comparing the result of the user selection with different strings and saving a conversion factor to a global variable based on whichever branch ended up being true, based on user selection. However, this caused the conversion factor to always save as 1.60934 (the conversion factor for kilometers), even if miles was selected. To solve this, I had to use local variables in the branches of the if statements and then save the resulting local variable to a global variable for use in later callbacks.

The second callback function I started on is the function for selecting distance. I wrote this callback by having an if loop with a branch for each distance option, and multiplying the time by whatever the distance of that option is. The last callback function I started writing is actually the first that is used, the callback for the edit box. This was one that I struggled with for a while because I could not figure out how to get the input as a variable. Once I got that figured out, I then had to make sure that the variable is in the correct “m:ss” format, which I eventually did using a regular expression. I wanted only inputs in that format, so I added a warning that pops up if the input is not in the correct format. I then had to figure out how to split the input into separate variables for minutes and seconds, which I did using the split function, and then converted that time into purely seconds, to make the calculations easier. When I realized that converting to seconds makes the calculations easier, I also added a few lines at the end of my last callback to convert the seconds back into a “m:ss” format. I struggled with this for a while because I could not get the individual strings to concatenate correctly. Eventually, I realized that the answer was simpler than I was making it out to be and I could combine them using a plus sign. I then created a modal message box to display the user’s distance selection and result of the calculations, saying “Your ‘distance’ run will take ‘time calculated’”. Overall, after a lot of google searches and even more trial and error, I feel like I understand GUI’s a lot more than I did when I started this project.

GUI’s are incredibly useful and can be very convenient for people who are inexperienced with coding. GUI’s can be used by anyone because using them does not require knowing anything about coding; we use them daily on websites and doing anything involving a computer. Though they are easier to use, the code is also much more difficult to write. However, though the code is more difficult, as I do it more, I will become more comfortable with it and it will come much more naturally with practice. I noticed a huge difference in my level of comfort with it even between the beginning and end of working on this project. When someone is very comfortable with GUI coding, the drawbacks are not going to hold them back as much. GUI is probably the best choice if the user is going to be needing the computations often, and especially if they are going to be sharing the code, for whatever reason, with someone new to coding. Nearly anything that can be done in regular MATLAB code can be done as a GUI, so the possible applications are incredibly broad. In my future, I could use GUI’s for math classes, if I wanted to write code to solve a specific type of problem rather than do it by hand every time. GUI’s are an incredibly useful tool when the person writing them understands what they are doing and has experience with it. They take more time to write, but can save a lot of time for the user if it is something they need to do consistently. My GUI for figuring out the time it takes to run a certain distance at a given pace will be handy to have and I will probably use it in the future, as I plan to keep running for the rest of my life.