Unit 14 Exercises – Graphical User Interfaces

Aims and objectives

- To experiment with using the Tkinter module to implement GUIs
- To gain experience of using Tkinter documentation

Task 1 – Preparation

Set up the files on Moodle under Unit14, which contains copies of the example programs from the Lecture. Read through the lecture slides and try the examples. It is especially important to do this if you did not go to the lecture.

Find the *Tkinter Reference* at **http://infohost.nmt.edu/tcc/help/pubs/tkinter/tkinter.pdf** or on Moodle and keep it open in the PDF viewer. It will be referred to several times in the exercises that follow.

Task 2 – Modifying the examples

1. Make a copy of the file *example5.py* and call it *task2.py*.

2. Add a call to the *title* method of the root window, *top*, to change the window's title from "Tk" to the title of your choice. The *title* method takes a string parameter, specifying the title.

3. In the *Tkinter Reference*, read the section on **Geometry strings**. Experiment with calling the *geometry* method of the root window to change its size and position. What happens when the window is larger than is needed to accommodate its widgets? What happens when it is smaller?

4. Each widget has a large number of options, for example *bg* for background colour. There are two ways of using these options. One is as arguments when creating the widget. The other is as arguments to the *configure* method of the widget (for the root window, this is the only way). Use the *bg* option to modify the example so that the background colour is white instead of grey. To make it look right, you will have to do this for all of the widgets in the example. To make the *Quit* button white even when it is pressed, use the *activebackground* option. The list of all of the options for the *Button* widget is specified in **The Button widget** section of the *Tkinter Reference*. Experiment with some of the others.

Task 3 – Currency converter

1. Make a copy of the file *example6.py* and call it *task3.py*.

2. CHANGE THIS: IT’S MUCH EASIER TO HAVE TWO SETS OF BUTTONS, ONE FOR THE INPUT CURRENCY AND ONE FOR THE OUTPUT CURRENCY. Based on this example, implement a currency converter. The idea is to have a radio button for each of a range of currencies (as many as you want), and use the text entry widget to enter an amount of money in whichever currency is currently selected by the buttons. Then pressing the buttons converts to other currencies. You will probably need to use the function *float* to convert from a string to a floating-point number, and you might want to use the string formatting operator to display

results to two decimal places. Documentation of *Radiobutton* is in **The Radiobutton widget** section of the *Tkinter Reference*.

Task 4 – A menu (do this in a file called *task4.py*)

1. The following code, if inserted into *example6.py* after setting up the top-level window, will add a "File" menu with the option "Exit". Try it out. How can you make the menu button appear all the way to the left of the window? The example in the **The Menubutton widget** section of the *Tkinter Reference* is very similar, if you remove every `"self"`.

```
fileMenuButton = Tkinter.Menubutton(top,text="File")
fileMenuButton.grid(row=0,column=0)
fileMenu = Tkinter.Menu(fileMenuButton,tearoff=0)
fileMenuButton.configure(menu=fileMenu)
fileMenu.add_command(label="Exit",command=top.destroy)
```

2. Modify your currency converter so that instead of using radio buttons to select the currency, the currencies appear in a menu. Use radio buttons within the menu so that you still have the idea of the currently-selected currency (use *add_radiobutton* instead of *add_command*; note that you can still use the *command* option in the same way as with the *Radiobutton* widget.

Task 5 (optional) – A GUI for the interpreter from Unit 13 (do this in *task5.py*)

Implement a GUI for the interpreter for the drawing language from Unit 13. You can use a *Canvas* widget (see **The Canvas widget** section of the *Tkinter Reference*) to provide a drawing area within your root window. Implement some way of letting the user specify the file name containing the drawing commands. You could do this with an *Entry* widget. More ambitiously, you could provide a "File" menu with an "Open" option, which pops up a new window (read **Toplevel: Top-level window methods** section of the *Tkinter Reference*) allowing a file name to be entered.