Mathematics for Sustainability (part 2)/ Graphics/ Graphical User Interface (GUI)

Graphical User Interface (GUI)

About GUIs

When you use a computer, you are probably not often inputting via a text prompt looking at output on the Python console - you are probably used to programs that use a *Graphical User Interface* to display windows with buttons, text, images and other elements. This is usually referred to as a GUI, which is pronounced "gooey".

GUIs are a bit fiddly, as you might imagine, which is why we have stuck to the Python shell for input and output so far.

Python can do GUIs in several ways. Here we will use a built-in module called Tkinter. We start by importing everything from `tkinter`.

```
import tkinter as tk
```

tkinter 'Hello World'

First, let's define a small window into which we can put things. The code below creates a window called `gui` and then displays it using a function called `mainloop()`. If you run this code, you should see a boring, empty little box.

```
import tkinter as tk

gui = tk.Tk()

gui.mainloop()
```

`mainloop()` keeps the window visible on screen, otherwise it would just exist for a fraction of a second and then disappear. It would typically be the last line of your program.

Customising your new window

There are various settings you can set for your new window.

Since this is your first GUI, you could give the window a traditional title.

```
import tkinter as tk

gui = tk.Tk()
gui.title("Hello, World!")

gui.mainloop()
```

Perhaps the window wasn't wide enough to see the title text you added? You can give it a size using `geometry()`. The dimensions are measured in pixels. Try out some different pixel sizes to get a feel for how big different windows are.

```
import tkinter as tk

gui = tk.Tk()
gui.title("Hello, World!")
gui.geometry("300x150")

gui.mainloop()
```

You can customise the window a lot more. For example, you can give the background a colour.

```
import tkinter as tk

gui = tk.Tk()
gui.title("Hello, World!")
gui.geometry("300x150")
gui.configure(background="sky blue")

gui.mainloop()
```

Adding GUI widgets

The elements we might add within our new window - text, buttons, etc. - are called widgets. We find the widgets we need under `ttk`, so we will import this as well as `tk` - please be wary, as they are similar names it is easy to get them confused.

```
import tkinter as tk
from tkinter import ttk
```

Displaying text

We can display some text by adding a `Label` widget to our window. Run this code and make sure you are looking at a tiny window displaying the `message`.

```
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
gui.title("Trying a label")
gui.geometry("300x100")

message = ttk.Label(gui, text="Yer a Wizard, Harry!")
message.pack()

gui.mainloop()
```

The function `pack()` tells the program to put this widget into the container we have asked for, in this can `gui`.

Buttons

A button is a clickable widget, typically causing something to happen. You tell a `Button` which window to display in and what text to display on the button.

```
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
gui.title("Trying a button")
gui.geometry("300x100")

button = ttk.Button(gui, text="Click me")
button.pack()

gui.mainloop()
```

What happens when you click your button? Well, what did you expect to happen? We haven't told the button to do anything yet.

To tell the button to do something, we give it the name of a function that will run. To show you how this works, the following example just displays a message. This gets pretty complicated, though, because as well as the `Button`, we need a `Label` to display the text on and a function to do the displaying (here called `do_when_clicked()`, but could be called anything).

```
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
gui.title("A working button")
gui.geometry("300x100")

joke = ttk.Label(gui, text="Why was 6 afraid of 7?")
joke.pack()

def do_when_clicked(): # run when button is pressed
    joke.config(text="Because 7 8 9")

button = ttk.Button(gui, text="I don't know", command=do_when_clicked)
button.pack()

gui.mainloop()
```

Text input

We can use an `Entry` widget to allow the user to enter some text.

```
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
```

```
gui.title("Trying a text box")
gui.geometry("300x100")

txt = tk.Entry(gui)
txt.pack()

gui.mainloop()
```

To get the text that has been entered into the box, for example when you click a button, you use `get()` like:

```
current_value = txt.get()
```

If we want to do something with the value that was entered, we need a way to trigger something to happen. In this more complicated example, we prompt the user to enter their name then use the GUI window to display a message.

```
"""
Simple user interface. Asks the user for their name and writes a message inclu
"""
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
gui.title("Trying a text box")
gui.geometry("300x100")

message = ttk.Label(gui, text="What's your name?")
message.pack()

txt = tk.Entry(gui)
txt.pack()

def do_when_clicked(): # run when button is pressed
    """
    Replaces the prompt with a message and hides the input box and button.
    """
    name = txt.get()
    message.config(text="Yer a Wizard, {}!".format(name))
    txt.destroy()
    button.destroy()

button = ttk.Button(gui, text="Enter", command=do_when_clicked)
button.pack()

gui.mainloop()
```

Drop-down list (Combobox)

If we want the user to select from a number of options, we can use a `Combobox`. Here we define a `Combobox` called `combo`, then fill it with `values`. If you run this and click the box, you'll see the values are available to select.

```
import tkinter as tk
from tkinter import ttk

gui = tk.Tk()
gui.title("Trying a Combobox")
gui.geometry("300x100")

combo = ttk.Combobox(gui)
combo['values'] = ("Doc", "Happy", "Sneezy", "Sleepy", "Bashful", "Grumpy", "D
combo.pack()

gui.mainloop()
```

To get the currently selected item, for example when you click a button, you use `get()` like:

```
current_value = combo.get()
```

More

There is a lot more to GUIs, but hopefully this has given you some idea of how it works. You could have a play around and search online if you want to do something but don't know how.

Exercise

Write a program that will display a window that includes a dropdown list of colours (see below), a message prompting the user to choose a colour and a button. When the button is clicked, the background colour of the window should change to the colour which is selected.

Colours:

- `DarkSlateGray1`
- `plum1`
- `wheat1`
- `goldenrod`
- `CadetBlue1`
- `firebrick1`