technocamps

Llywodraeth Cymru
Welsh Government

Prifysgol Abertawe
Swansea University

Cardiff University
Prifysgol Caerdydd

Prifysgol Bangor University

Cardiff Metropolitan University | Prifysgol Metropolitan Caerdydd

University of South Wales
Prifysgol De Cymru

Medr
Y Comisiwn Addysg Drydyddol ac Ymchwil
Commission for Tertiary Education and Research

institute of CODING in wales technocamps

Prifysgol Aberystwyth University

Prifysgol Wrecsam
Wrexham University

Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

technocamps

Pyshop 2: Tkinter Boogaloo

Start at the Very Beginning

Download the broken login program here:

tc1.me/tkinter1

Create a new folder called ‘gui_shop’ (or something sensible) and place this file within.

When done you may open the file, read over the code and familiarise yourselves with it.

Start at the Very Beginning

Before we begin testing this code, we will need a way to run it!

We will do this by making a new Python file called 'gui_shop_main.py' in the same directory.

In this file we will need to import and call the 'create_login' function from the 'gui_shop_login' file.

```
from gui_shop_login import create_login

create_login()
```

Start at the Very Beginning

Now, by running the file ‘gui_shop_main’ the login code should also run and launch a new GUI Window.

However, this will look a little off. There are 6 issues with this code; for now these are all logic errors (i.e. the program will still run).

Your challenge is to find and fix the errors so that the program runs correctly!

Note: that we do not have the code for the shopface yet.

Now STOP. Comment Time.

Before we continue you should ensure that your code is well commented!

Commenting is an essential skill that you will be tested on.

- Don't comment every line of code, space-out code and comment chunks that do a thing.
- Comments shouldn't explain what the code is doing (that's what the code is for) but should explain the purpose.
- Functions should be commented with `"""docs"""` that explain the functions purpose and outline any arguments (`Args:`) and/or Return statements

Gotta Grow Fast

Download the other modules here:

tc1.me/tkinter2

Place them in the directory with your other files.

Gotta Grow Fast

We need a way to switch between the login window and the shopface via log-in/log-out buttons.

This is quite tricky to do when we're using Python procedurally, as we can run into errors by 'infinitely' switching windows*.

So, we're going to have to cheat the system and be a little clever!

- If you're interested – we can also avoid a "RecursionError" by building our programs in an Object Oriented structure, as opposed to Procedural. This is how tkinter is *meant* to be used – but it is beyond the scope of the Computer Science GCSE.

The Main of the Game

To avoid the error, we need to destroy the login window before switching to the shop. But if we do this within the login window the shop will never be called!

So, we'll have to implement this within 'gui_shop_main'. Begin by importing the *create window* functions into main:

```
from gui_shop_login import create_login
from gui_shop_shopface import create_shopface


~~~
```

The Main of the Game

Now we'll create two functions that achieve the *destroy* → *create* behaviour. Notice how we can pass a function as a variable by dropping the end brackets!

```
~~~

def switch_to_shop(login_root):
    login_root.destroy()
    create_shopface(switch_to_login)

~~~
```

The Main of the Game

And finally after the second function we simply call create_login once to begin the cycle!

```
~~~

def switch_to_login(shopface_root):
    shopface_root.destroy()
    create_login(switch_to_shop)

create_login(switch_to_shop)
```

Login, Or Not Login

We must also update our gui_shop_login to use this new behaviour. Then our login window should launch the shop!

```
def create_login(switch_to_shop):

~~~

if ((username_entry.get() == stored_username) and
    (password_entry.get() == stored_password)):

      switch_to_shop(login_root)
```

I code, therefore I comment

Once again, before continuing we will add some brief commenting to ‘gui_shop_main’!

Remember:

- Don’t comment every line of code, space-out code and comment chunks that do a thing.
- Comments shouldn’t explain what the code is doing (that’s what the code is for) but should explain the purpose.
- Functions should be commented with `"""docs"""` that explain the functions purpose and outline any arguments (`Args:`) and/or Return statements

Pyth’ Finds A Way