Visual Programming

College Of Computer Sciences and Mathematics

Dr. Firas Aswad

Dept. of Networks

Jan. 29, 2025

Class Policy

1-No one enters after I enter the classroom. If you are late, get a permission letter from the dept. office.

2- Put your phone on silent, and leave it on my desk.

3- You are allowed to bring your water with you to drink and a snack to eat, with two conditions: you make no noises or bother your classmate.

4- I don't allow students to use gums except for specific conditions.

5- I'll take attendance.

6- You have weekly quizzes for about 2 to 5 minutes about the materials we covered the previous week. We will collect these quizzes and give a couple of forgiveness excuses. Finally, we take the highest scores (for example, the top 13 quizzes).

7- I'll never forgive the student who cheated in my course.

8- Phones are not allowed to take pictures or record audio; otherwise, I'll zero him/her in that week.

9- Don't try to send someone to make me change your score or treat you differently (it will not work, and it will just make me dislike you).

10- Have fun and enjoy programming (Hard workers will be rewarded, and slackers will not be).

What is GUI ?

- GUI stands for "Graphical User Interface".
- What is Interface?
- The interface is a point where two things meet, they do interact. In general, an interface is a device or a system that uses unrelated entities to interact.
- By this definition, remote control is an interface between a person and a television, the English language is an interface between two people, and the behavioral protocol applied in government institutions is the interface between people of different ranks and positions.

What is UI?

- What is the **User Interface?** In short form, we call it a "**UI**".
  UI is an interface created for the user of a particular machine or computer.

- In general, the user interface (UI) is the point at which human users interact with a computer, website, or application.

- The **goal of effective UI** is to make the user's experience **easy** and intuitive, requiring minimum effort on the user's part to receive the maximum desired outcome.

- An ideal example for UI is a normal calculator, where the user can type any numbers, multiplication, or subtraction. The calculator is made for user interaction where the user can give inputs and see output.

- **UI** is created in layers of interaction that appeal to the human senses (sight, touch, auditory, and more).
- They include input devices like a keyboard, mouse, trackpad, microphone, touch screen, fingerprint scanner, e-pen, and camera and output devices like monitors, speakers, and printers.

Graphical user interface:

- **Form-based user interface:** Used to enter data into a program or application by offering a limited selection of choices. For example, a settings menu on a device is form-based.

- **Menu-driven user interface:** A UI that uses a list of choices to navigate within a program or website. For example, ATMs use menu-driven UIs and are easy for anyone to use.

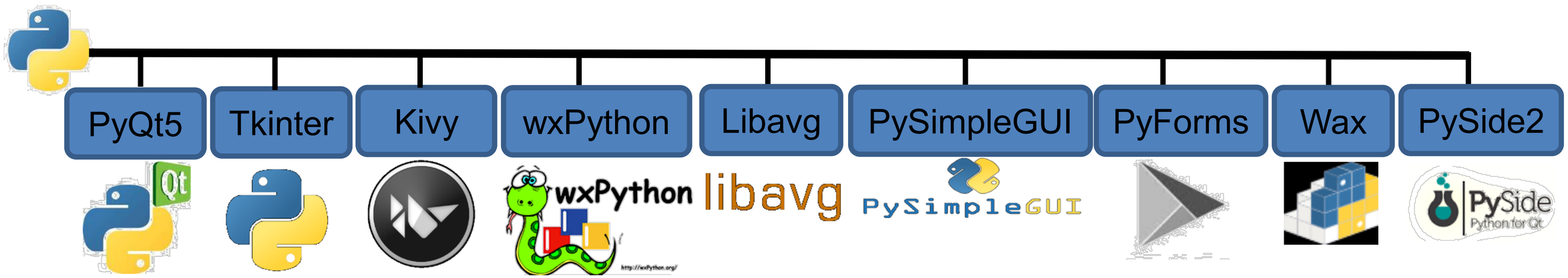
- **Touch user interface:** User interface through haptics or touch. Most smartphones, tablets, and devices that operate using a touch screen use haptic input.

- **Voice user interface**: Interactions between humans and machines using auditory commands. Examples include virtual assistant devices, talk-to-text, GPS, and much more.

- A graphical user interface (GUI) allows a user to interact visually with a program. GUIs are built from **GUI controls** (which are sometimes called **components or widgets** short for window gadgets).

- GUI controls are objects that can display information on the screen or enable users to interact with an app via the mouse, keyboard, or some other form of input (such as voice commands).

Python Libraries for GUI (Python GUI Frameworks for Developers)



Tkinter

- It is considered a favorite among developers and learners for its ease of use and simplicity.
- It is an inbuilt Python module used to create GUI apps, and it is one of the most commonly used modules for GUI apps.
- There is no need to install it as it comes by default with all Python versions.
- When combined with Tkinter with Python, provides a fast and easy way to create GUI applications.

Building a GUI with Tkinter widgets

```
# banana_survey.py
"""A banana preferences survey written in Python with Tkinter"""

import tkinter as tk
root = tk.Tk()
root = mainloop()
```

The following is a description of the preceding code:

- The first line imported the tkinter module into the namespace with tk as its alias. Now, we can access all definitions of the classes, attributes, and methods of Tkinter by appending the alias tk to the name as in tk.Tk().

- The second line created an instance of the tkinter.Tk class. This created what is called the root window, which is shown in the preceding screenshot. According to the conventions, the root window in Tkinter is usually called root, but you are free to call it by any other name.

- The third line executed the mainloop (that is, the event loop) method of the root object. The mainloop method is what keeps the root window visible. If you remove the third line, the window created in line 2 will disappear immediately as soon as the script stops running.

- Tkinter also exposed the mainloop method as tkinter.mainloop(). So, you can even call mainloop() directly instead of calling root.mainloop().

Widgets – the building blocks of GUI programs

The syntax that is used to add a widget is as follows:

```
my_widget = tk.Widget-name (its container window, ** its configuration options)
```

```
import tkinter as tk
root = tk.Tk()
label = tk.Label(root, text="I am a label widget")
button = tk.Button(root, text="I am a button")
label.pack()
button.pack()
root.mainloop()
```

The following is a description of the preceding code:

- This code added a new instance named label for the label widget. The first parameter defined root as its parent or container. The second parameter configured its text option to read I am a label widget.

- Similarly, we defined an instance of a Button widget. This is also bound to the root window as its parent.

- We used the pack() method, which is essentially required to position the label and button widgets within the window.

Ways to create widgets

```
my_label = tk.Label(root, text="I am a label widget”)
my_label.pack()
```

OR

```
tk.Label(root, text="I am a label widget").pack()
```

- You can either save a reference to the widget created (my_label, as in the first example) or create a widget without keeping any reference to it (as demonstrated in the second example).

- Ideally, You should reference the widget in case the widget has to be accessed later in the program.

Some important widget features

- All widgets are actually objects derived from their respective widget classes. So, a statement such as button = Button(its_parent) actually creates a button instance from the Button class.

- Each widget has a set of options that decides its behavior and appearance. This includes attributes such as text labels, colors, and font size. For example, the Button widget has attributes to manage its label, control its size, change its foreground and background colors, change the size of the border, and more.

- To set these attributes, you can set the values directly at the time of creating the widget, as demonstrated in the preceding example. Alternatively, you can later set or change the options of the widget by using the .config() or .configure() method. Note that the .config() or .configure() methods are interchangeable and provide the same functionality.

In fact, the .config() method is simply an alias of the .configure() method.

Tkinter includes 21 core widgets

Top-level

Canvas

Frame

Menu

OptionMenu

Scale

Text

Label

Checkbutton

LabelFrame

Menubutton

PanedWindow

Scrollbar

Bitmap

Button

Entry

Listbox

Message

Radiobutton

Spinbox

Image

Thank You!