OBJECT ORIENTED PROGRAMMING

PROJECT BASED

2nd semester (Lect2)

Dr. Ann Alkazzaz

Creating Log in and Registration GUI window

Outlines

- Introduction to GUI Programming
- Introduction to Tkinter
- Code Structure - OOP Approach
- Create GUI Window in Python

Introduction to GUI Programming

- GUI stands for "Graphical User Interface." It is a type of interface that allows users to interact with electronic devices, software, or applications through graphical elements such as icons, buttons, windows, and menus.
- **Importance:** GUIs are designed to enhance user interaction by offering a visual interface for controlling and engaging with a system.

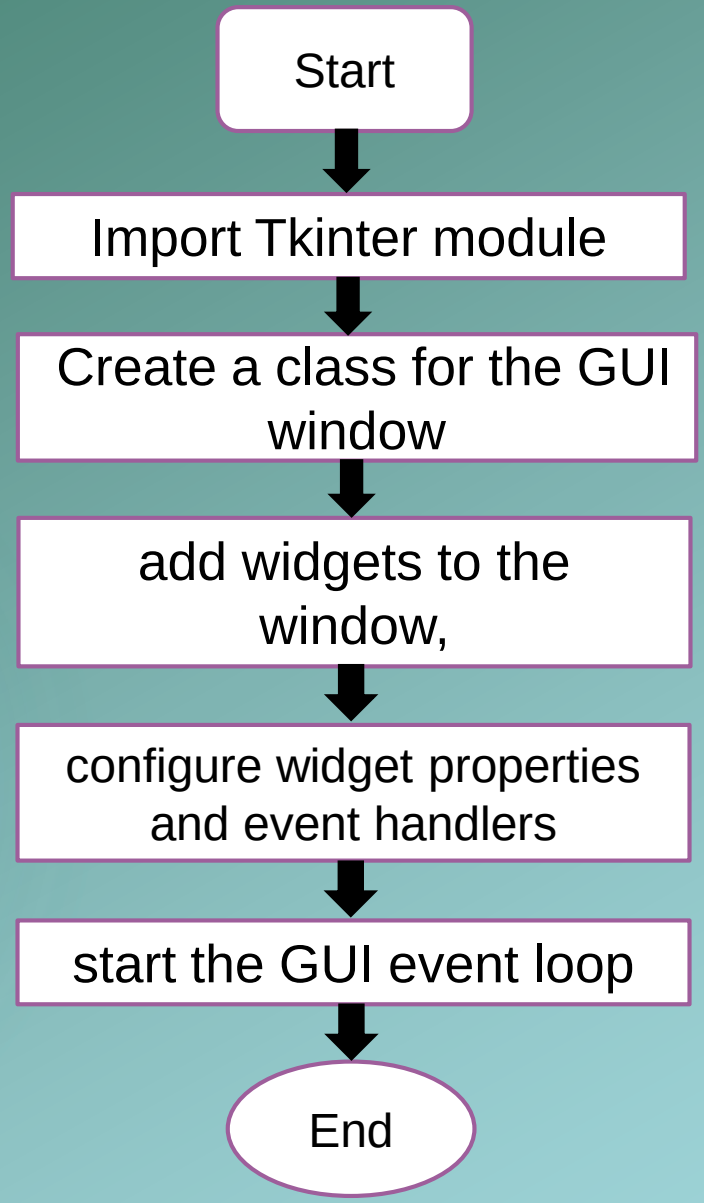
Introduction to Tkinter

- **Tkinter** is a Python library used for creating graphical user interfaces (GUIs). It provides a set of built-in widgets for creating desktop .
- Tkinter is a standard Python library, that comes installed with Python when you download and install it on your computer.

- How to use **tkinter** to create GUI window:

  There are five main steps that have to be followed to create a GUI window.

Create GIU window



Create GIU window

1. Import Tkinter module

```
import tkinter as tk
```

2. Create a class for the GUI window:

```
class GUIWindow:
    def __init__(self):
        self.window = tk.Tk()  # Create the main window
        self.window.title("GUI Window")
        # Set the size of the window
        self.window.geometry("400x300")  # Width x Height
```

- **tk**: refers to the Tkinter module, which is the standard Python interface to the Tk GUI toolkit.
- **Tk()**: This is a constructor function that creates an instance of the **Tk** class, representing the main window of a Tkinter application.
- In essence, **tk.Tk()** initializes and returns a main window object, which serves as the root or top-level window for the GUI application. This window is where you can add other widgets and elements to create the user interface.

Create GIU window

3. add widgets (buttons, labels, etc.) to the window (define methods):

```
def add_label(self, text):
    label = tk.Label(self.window, text=text, width=20, height=3)
    label.pack()
def add_button(self, text, command=None):
    button = tk.Button(self.window, text=text, command=command, width=10, height=2)
    button.pack()
```

- These two methods are designed to add a **Label** and a **Button** widget to the Tkinter window.
- The **add_label** method takes one argument text, which is the text to be displayed on the label.
  - It creates a Label widget using tk.Label.
  - The width and height parameters set the dimensions of the label widget.
  - Finally, it uses the pack() method to pack the label into the window. pack() is a geometry manager method in Tkinter used to place widgets in the window according to a packing algorithm.
- The **add_button** mehod adds a Button widget to the Tkinter window.
  - it takes two arguments: **text**, which is the text displayed on the button, and **command**, which is the function to be called when the button is clicked.
  - It creates a Button widget using **tk.Button**.
  - it uses the **pack()** method to pack the button into the window.

Create GIU window

4. define a method to start the GUI loop:

```
def start(self):
    self.window.mainloop()  # Start the GUI event loop
```

- This method typically starts the event loop which allows the GUI to respond to user interactions.
- **window.mainloop() :** means that execution of your Python commands halts there function in the Tkinter library in Python. It is called to run the main event loop of a Tkinter application. This loop waits for and registers events such as mouse clicks and button presses, then directs these events to the appropriate handlers in the application for processing.

Create GIU window

5. Instantiate the GUIWindow class and use its methods to add widgets:

```
gui = GUIWindow()
gui.add_label("Hello, Tkinter!")
gui.add_button("Click me")
gui.start()
```

- creates an instance of the GUIWindow class. This line calls the constructor (__init__ method) of the GUIWindow class, which initializes the main window (self.window) with a title and dimensions.
- calls the **add_label method** of the GUIWindow instance (gui). This method adds a Label widget to the main window (self.window) with the specified text ("Hello, Tkinter!").
- calls the **add_button method** of the GUIWindow instance (gui). This method adds a Button widget to the main window (self.window) with the specified text ("Click me").
- This method starts the GUI event loop using self.window.mainloop().