GUI (Graphical User Interface) Module-IV[T302]MCA

Most of the programs we have done till now are text-based programming. But many applications need GUI (Graphical User Interface).
Python provides several different options for writing GUI based programs. These are listed below:

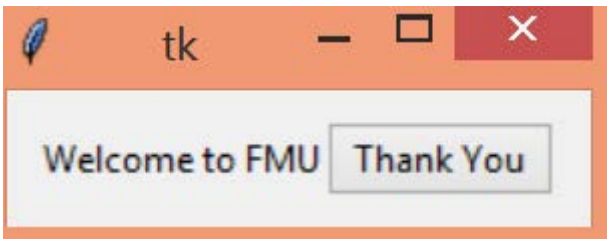
- **Tkinter**: It is easiest to start with. Tkinter is Python's standard GUI (graphical user interface) package. It is the most commonly used toolkit for GUI programming in Python.
- **JPython**: It is the Python platform for Java that is providing Python scripts seamless access o Java class Libraries for the local machine.
- **wxPython**: It is an open-source, cross-platform GUI toolkit written in C++. It is one of the alternatives to Tkinter, which is bundled with Python.

Steps to create GUI using tkinter

- Import the module Tkinter
- Build a GUI application (as a window)
- Add widgets (text box, radio button etc)
- Enter the primary, i.e., the main event's loop for taking action when the user triggered the event.

First Python GUI program

```
from tkinter import *
from tkinter import ttk
root = Tk()
frm = ttk.Frame(root, padding=10)
frm.grid()
ttk.Label(frm, text="Welcome to FMU").grid(column=0, row=0)
ttk.Button(frm, text="Thank You", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

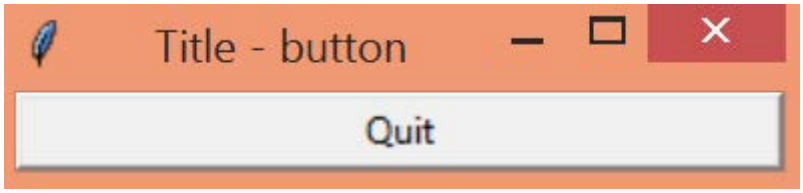


- After the imports, the next line creates an instance of the Tk class, which initializes Tk and creates its associated Tcl interpreter. It also creates a top level window, known as the root window, which serves as the main window of the application.
- The next line creates a frame widget, which in this case will contain a label and a button we'll create next. The frame is fit inside the root window.
- The next line creates a label widget holding a static text string. The grid() method is used to specify the relative layout (position) of the label within its containing frame widget, similar to how tables in HTML work.

- A button widget is then created, and placed to the right of the label. When pressed, it will call the destroy() method of the root window.
- Finally, the mainloop() method puts everything on the display, and responds to user input until the program terminates.

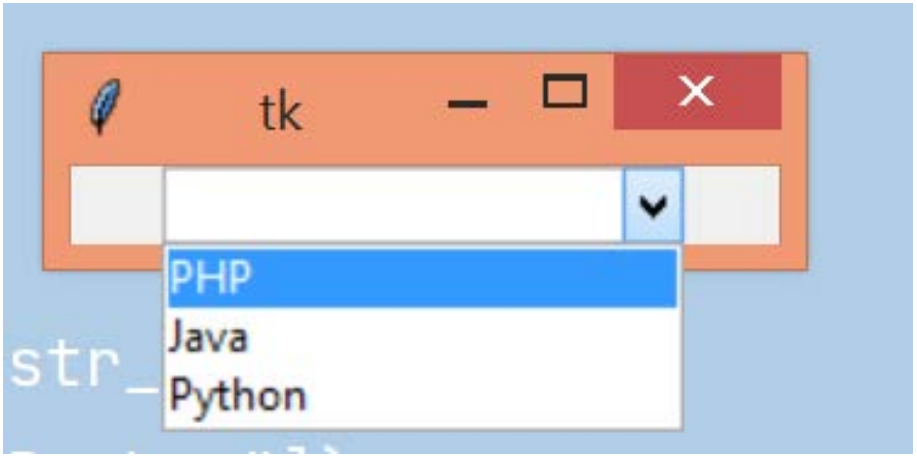
Python GUI program to add a button in your application using tkinter module

```
import tkinter as tk
parent = tk.Tk()
parent.title('Title - button')
my_button = tk.Button(parent, text='Quit', height=1, width=35,
command=parent.destroy)
my_button.pack()
parent.mainloop()
```



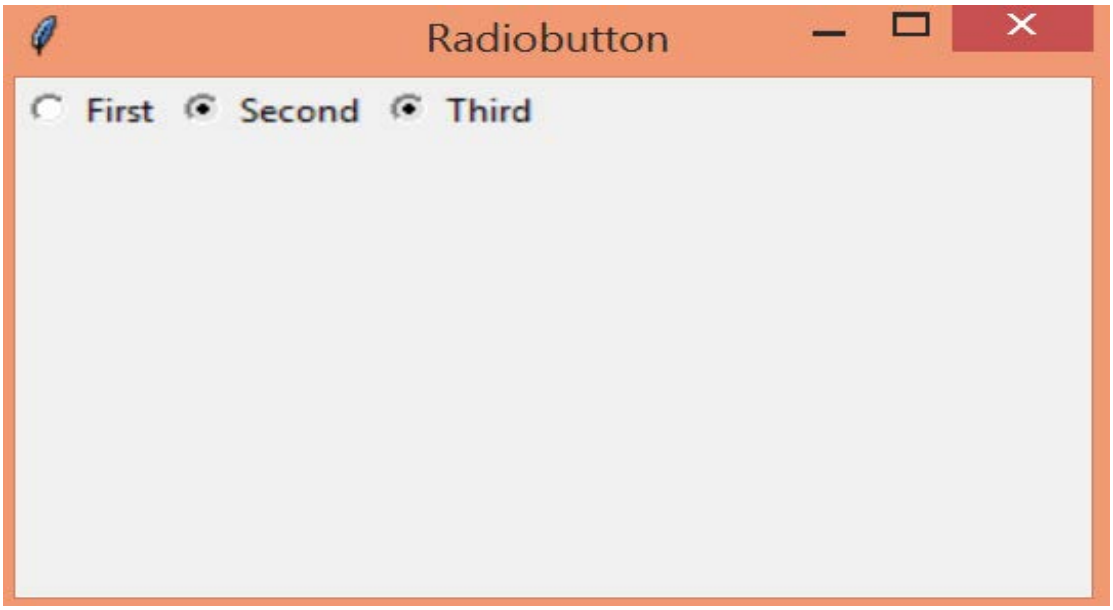
Python GUI program to create a Combobox with three options using tkinter module

```
from tkinter import ttk
root = tk.Tk()
my_str_var = tk.StringVar()
my_combobox = ttk.Combobox(
   root, textvariable = my_str_var,
   values=["PHP", "Java", "Python"])
my_combobox.pack()
root.mainloop()
```



Python GUI program to create three radio buttons widgets using tkinter module.

```
import tkinter as tk
parent = tk.Tk()
parent.title("Radiobutton ")
parent.geometry('350x200')
radio1 = tk.Radiobutton(parent, text='First', value=1)
radio2 = tk.Radiobutton(parent, text='Second', value=2)
radio3 = tk.Radiobutton(parent, text='Third', value=3)
radio1.grid(column=0, row=0)
radio2.grid(column=1, row=0)
radio3.grid(column=2,row=0)
parent.mainloop()
```



Python GUI tkinter program to add two integers from two text boxes and display their sum in another text box

```
from tkinter import *

def addNumbers():
    res = int(e1.get()) + int(e2.get())
    myText.set(res)

master = Tk()
myText = StringVar()
Label(master, text="First").grid(row=0, sticky=W)
Label(master, text="Second").grid(row=1, sticky=W)
Label(master, text="Result:").grid(row=3, sticky=W)
result = Label(master, text="", textvariable=myText).grid(row=3, column=1, sticky=W)
e1 = Entry(master)
e2 = Entry(master)
e1.grid(row=0, column=1)
e2.grid(row=1, column=1)
b = Button(master, text="Calculate", command=addNumbers)
b.grid(row=0, column=2, columnspan=2, rowspan=2, sticky=W + E + N + S, padx=5,
```

```
pady=5)

mainloop()
```