**Developing GUI project using Tkinter**

**Introduction**

Python provides various options for developing graphical user interfaces. Most important are [Tkinter](https://realpython.com/python-gui-tkinter/), [wxPython](https://www.wxpython.org/), JPython and [PyQt](https://riverbankcomputing.com/software/pyqt/intro), to name a few. GUI’s are used when you want to build front-end applications. As mentioned before, Python is mainly used as a back-end application and this explains the reason why most of our lectures were run on a console.

* Tkinter is the Python interface to GUI toolkits shipped with python.
* wxPython is an open source interface for wxWindows.
* JPython is a Python port for Java which gives Python scripts seamless access to java class libraries.
* PyQt −This is also a Python interface for a popular cross-platform Qt GUI library.

**Tkinter Programming**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps: Example: Import the Tkinter module by adding it through project Interpreter.

Create the GUI application main window. Add one or more of the above mentioned widgets to the GUI application. Enter the main event loop to take action against each event triggered by the user.

**import tkinter**

**root = tkinter.Tk()**

# Code to add widgets will go here...

**root.mainloop()**

**=================================================================================================================A very basic GUI sample=================================**

**from** tkinter **import** \*  
root = Tk() *# created a blank window*myLabel= Label(root, text=**"Welcome to GUI design"**) *#create an label object and position it within the root (main window)*myLabel.pack() *#put the widget in the window and displays it*root.mainloop() *#to continuously display the GUI otherwise it flashes and disappears*

To have a bigger window, root allow geometry method where you can specify the size as shown below

**root.geometry**(**"300x300"**)

There are a number of widgets in Tkinter as described below (not limited to the ones discussed).

1. **Button** - The Button widget is used to display the buttons in your application.
2. **Label-** The Label widget is used to provide a single-line caption for other widgets. It can also contain images.
3. **Message**- The Message widget is used to display multiline text fields for accepting values from a user. Try the following code:

**from** tkinter **import** \*  
root = Tk()  
root.geometry(**"200x200"**)  
var = StringVar()  
mylabel = Message( root, textvariable = var, relief = RAISED )  
var.set(**"How is GUI programming with tkinter?"**)  
mylabel.pack()  
root.mainloop()

1. **Python - Tkinter Button**  
   The Button widget is used to add buttons in a Python application. These buttons can display text or images that convey the purpose of the buttons. You can attach a function or a method to a button, which is called automatically when you click the button.

**Syntax:**

w = Button ( master, option=value, ... )Parameters:

master: This represents the parent window.

options: Here is the list of most commonly used options for this widget. These options can be used as key-value pairs separated by commas.

See code sample below for adding button widgets to the window.

Try this out

**from** tkinter **import** \*  
root = Tk() *# created a blank window  
#myLabel= Label(root, text="Welcome to GUI design") #create an label object and position it within the root (main window)  
#myLabel.pack() #put the widget in the window and displays it*mytopFrame=Frame(root)*#have added a top frame which is invisible*mytopFrame.pack()  
mybottomframe=Frame(root)  
mybottomframe.pack(side=BOTTOM) *# just packing another invisible frame at the bottom*myaddButton3=Button(mytopFrame, text=**"Add"**, fg=**"red"**)*# Positioning my button. takes 3 parameters (position, text and colour)*myaddButton3.pack(side=LEFT)  
mysubButton=Button(mytopFrame, text=**"Subtract"**, fg=**"red"**)*# Positioning my button. takes 3 parameters (position, text and colour)*mysubButton.pack(side=LEFT)  
mymultiButton2=Button(mytopFrame, text=**"Multiply"**, fg=**"red"**)*# Positioning my button. takes 3 parameters (position, text and colour)*mymultiButton2.pack(side=LEFT)  
  
myexitButton4=Button(mybottomframe, text=**"Exit"**, fg=**"red"**)*# Positioning my button. takes 3 parameters (position, text and colour)*myexitButton4.pack(side=BOTTOM)  
root.mainloop() *#to continuously display the GUI otherwise it flshes and disappears*

*====================================================================================*

1. **tkMessageBox**-This module is used to display message boxes in your applications.

**Example:**

**Try this out**

**from** tkinter **import** \*  
**from** tkinter **import** messagebox  
root = Tk()  
root.geometry(**"400x200"**) *#just defining the size of your window***def** hello(): *#declaring function for the messagebox* messagebox.showinfo(**"Say Hello"**, **"Hello World"**)  
  
B1 = Button(root, text=**"Good morning "**, command=hello()) *#Calling your function after clicking button*B1.place(x=35, y=50)  
  
root.mainloop()

1. **Listbox**- The Listbox widget is used to provide a list of options to a user.
2. **Menu**-The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.
3. **Entry**- The Entry widget is used to display a single-line text field for accepting values from a user.

**Try this out (code to create three widgets - button, text-entry and label)**

*#import the 'tkinter module*

**import tkinter**

window = tkinter.Tk()

window.geometry(**"200x200")**

*#create the title for the window*

window.title**("Creating 3 Widgets - label, textentry and button")**

***#creating a label widget (label1)***

label1=tkinter.Label(window**, text="Label")**

*#creating a text entry widget (txtentry)*

txtentry=tkinter.Entry(window)

*#creating a button widget (called btn)*

btn=tkinter.Button(window**, text ="Button")**

*#Adding the widgets, in order to the created window. Keyword =pack*

label1.pack()

txtentry.pack()

btn.pack()

*#Finally, draw the window + start the application*

window.mainloop()

1. **Listbox**- The Listbox widget is used to provide a list of options to a user. You will just need an object which inherit from Listbox class as shown below: myListbox1 = Listbox(window)

**Try this out (code to create a listbox and display 9 provincial capitals in SA)**

**from** tkinter **import** \*  
window = Tk()  
window.geometry(**"300x300"**)  
window.title(**"Using a listbox"**)  
myLabel= Label(window, text=**"Provincial capitals of SA"**) *#create an label object and position it within the root (main window)*myLabel.pack() *#put the widget in the window and displays it*myListbox1 = Listbox(window)  
myListbox1.insert(1, **"Cape Town"**)  
myListbox1.insert(2, **"Bloemfontein"**)  
myListbox1.insert(3, **"Durban"**)  
myListbox1.insert(4, **"Pretoria"**)  
myListbox1.insert(5, **"Joburg"**)  
myListbox1.insert(6, **"Kimberley"**)  
myListbox1.insert(7, **"East London"**)  
myListbox1.insert(8, **"Polokwane"**)  
myListbox1.insert(9, **"Witbank"**)  
myListbox1.pack()  
window.mainloop()

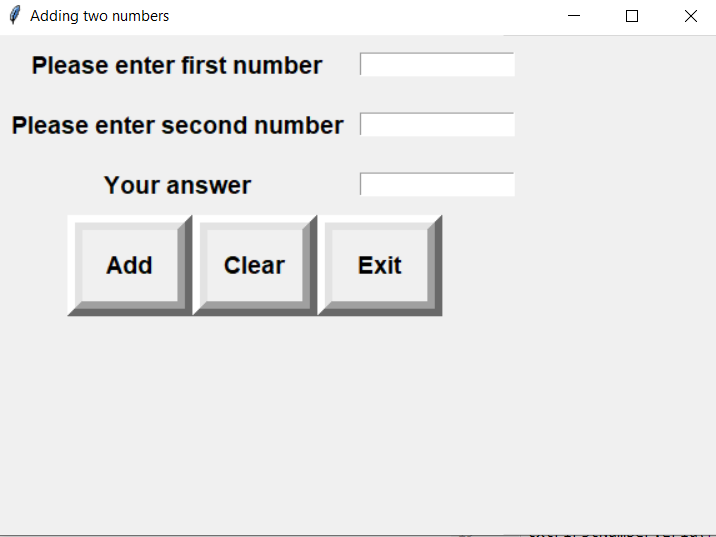
10. Text- The Text widget is used to display text in multiple lines.

Other widgets include: Scale, Scrollbar, SpinBox, Panelwindow etc. These you can research on your own.

So let us try to do a complete exercise. You should find this easy.

**Exercise 1**

Design a program in Python which accepts two numbers using textboxes and displays the answer in the third textbox. First Design the interface like the one below.



When btnAdd button is clicked, the program should show the answer in the third textbox. When btnClear button is clicked, the program must clear the three textboxes. When btnExit is clicked, the program should terminate. Add a try catch to trap an error if user clicks without entering values for number1 and number 2.

**Project 2: temperature convertor (Degrees Celsius to Fahrenheit and the other way round)**

Design and interface as one below and write a program which converts Fahrenheit to equivalent degrees Celsius and the other way round. You have to look for the appropriate formula for conversion.

See sample of the interface. You are free to tweak the interface to your own design.

