GUI using Tkinter

Table Of Contents

1. What Is A Graphical User Interface(GUI)

2. What Is Tkinter

3. Introduction To Tkinter

4. Tkinter Widgets

5. Geometry Management

6. Organizing Layout And Widgets

 6.1 Grid

7. Binding Functions

8. Mouse Clicking Events

9. Classes

10. Drop-Down Menus

11. Alert Boxes

12. Simple Shapes

13. Images And Icons

14. Creating Calculator

**What Is A Graphical User Interface(GUI)**

**GUI** is a desktop app which helps you to interact with the computers. They are used to perform different tasks in the desktops, laptops, other electronic devices, etc.., Here, we mainly talking about the laptops and desktops.

* **GUI** apps like **Text-Editors** are used to create, read, update and delete different types of files.
* **GUI** apps like **Sudoku, Chess, Solitaire, etc..,** are games which you can play.
* **GUI** apps like **Chrome, Firefox, Microsoft Edge, etc..,** are used to surf the **Internet**.

## What Is Tkinter

**Tkinter** is an inbuilt **Python** module used to create simple **GUI** apps. It is the most commonly used module for **GUI** apps in the **Python**.

You don't need to worry about installation of the **Tkinter** module as it comes with **Python** default.

## Introduction To Tkinter

Run the following code to create a simple window with the text **Hello World!**.

**Steps:-**

* import the module **tkinter**.
* Initialize the window manager with the **tkinter.Tk()** method and assign it to a variable **window**. This method creates a blank window with close, maximize and minimize buttons.
* Rename the title of the window as you like with the **window.title(title\_of\_the\_window)**.
* **Label** is used to insert some objects into the **window**. Here, we are adding a **Label** with some text.
* **pack()** attribute of the widget is used to display the **widget** in a size it requires.
* Finally, the **mainloop()** method to display the **window** until you manually close it.

import tkinter

window = tkinter.Tk()

# to rename the title of the window

window.title("GUI")

# pack is used to show the object in the window

label = tkinter.Label(window, text = "Hello World!").pack()

window.mainloop()

## Tkinter Widgets

**Widgets** are something like elements in the **HTML**. You will find different types of **widgets** to the different types of elements in the **Tkinter**.

Let's see the brief introduction to all of these widgets in the **Tkinter**.

* **Button**:- **Button** widget is used to place the buttons in the **tkinter**.
* **Canvas**:- **Canvas** is used to draw shapes in your **GUI**.
* **Checkbutton**:- **Checkbutton** is used to create the check buttons in your application. You can select more than one option at a time.
* **Entry**:- **Entry** widget is used to create input fields in the **GUI**.
* **Frame**:- **Frame** is used as containers in the **tkinter**.
* **Label**:- **Label** is used to create a single line widgets like **text**, **images**, etc..,
* **Menu**:- **Menu** is used to create menus in the **GUI**.

## Geometry Management

All widgets in the **tkinter** will have some geometry measurements. These measurements give you to organize the widgets and their parent frames, windows, etc..,

**Tkinter** has the following three Geometry Manager classes.

* **pack()**:- It organizes the widgets in the block, which mean it occupies the entire available width. It's a standard method to show the widgets in the window
* **grid()**:- It organizes the widgets in table-like structure. You will see details about **grid** later in this tutorial.
* **place()**:- It's used to place the widgets at a specific position you want.

## Organizing Layout And Widgets

To arrange the layout in the **window**, we will use **Frame**, class. Let's create a simple program to see how the **Frame** works.

**Steps:-**

* **Frame** is used to create the divisions in the window. You can align the frames as you like with **side** parameter of **pack()** method.
* **Button** is used to create a button in the window. It takes several parameters like **text**(Value of the Button), **fg**(Color of the text), **bg**(Background color), etc..,

**Note:-** The parameter of any **widget** method must be where to place the widget. In the below code, we use to place in the **window**, **top\_frame**, **bottom\_frame**.

import tkinter

window = tkinter.Tk()

window.title("GUI")

# creating 2 frames TOP and BOTTOM

top\_frame = tkinter.Frame(window).pack()

bottom\_frame = tkinter.Frame(window).pack(side = "bottom")

# now, create some widgets in the top\_frame and bottom\_frame

btn1 = tkinter.Button(top\_frame, text = "Button1", fg = "red").pack()# 'fg - foreground' is used to color the contents

btn2 = tkinter.Button(top\_frame, text = "Button2", fg = "green").pack()# 'text' is used to write the text on the Button

btn3 = tkinter.Button(bottom\_frame, text = "Button2", fg = "purple").pack(side = "left")# 'side' is used to align the widgets

btn4 = tkinter.Button(bottom\_frame, text = "Button2", fg = "orange").pack(side = "left")

window.mainloop()

Now, we will see how to use the **fill** parameter of **pack()**

import tkinter

window = tkinter.Tk()

window.title("GUI")

# creating 3 simple Labels containing any text

# sufficient width

tkinter.Label(window, text = "Suf. width", fg = "white", bg = "purple").pack()

# width of X

tkinter.Label(window, text = "Taking all available X width", fg = "white", bg = "green").pack(fill = "x")

# height of Y

tkinter.Label(window, text = "Taking all available Y height", fg = "white", bg = "black").pack(side = "left", fill = "y")

window.mainloop()

### Grid

**Grid** is another way to organize the **widgets**. It uses the **Matrix row column** concepts. Something like this.

2 x 2 Matrix

0 0 0 1

1 0 1 1

import tkinter

window = tkinter.Tk()

window.title("GUI")

# creating 2 text labels and input labels

tkinter.Label(window, text = "Username").grid(row = 0) # this is placed in 0 0

# 'Entry' is used to display the input-field

tkinter.Entry(window).grid(row = 0, column = 1) # this is placed in 0 1

tkinter.Label(window, text = "Password").grid(row = 1) # this is placed in 1 0

tkinter.Entry(window).grid(row = 1, column = 1) # this is placed in 1 1

# 'Checkbutton' is used to create the check buttons

tkinter.Checkbutton(window, text = "Keep Me Logged In").grid(columnspan = 2) # 'columnspan' tells to take the width of 2 columns

# you can also use 'rowspan' in the similar manner

## Binding Functions

Calling functions whenever an event occurs refers to a binding function.

* In the below example, when you click the button, it calls a function called **say\_hi**.
* Function **say\_hi** creates a new label with the text **Hi**.

import tkinter

window = tkinter.Tk()

window.title("GUI")

# creating a function called say\_hi()

def say\_hi():

tkinter.Label(window, text = "Hi").pack()

tkinter.Button(window, text = "Click Me!", command = say\_hi).pack() # 'command' is executed when you click the button

# in this above case we're calling the function 'say\_hi'.

window.mainloop()

Another way to bind functions is using **events**. Events are something like **mousemove, mouseover, clicking, scrolling, etc..,**.

The following program also produces the same output as the above one.

* '**<Button-1>**' parameter of **bind** method is the left clicking event, i.e., when you click the left button the **bind** method call the function **say\_hi**
  + <Button-1> for **left** click
  + <Button-2> for **middle** click
  + <Button-3> for **right** click
* Here, we are **binding** the **left** click event to a **button**. You can bind it to any other **widget** you want.
* You will have different parameters for different events

import tkinter

window = tkinter.Tk()

window.title("GUI")

# creating a function with an arguments 'event'

def say\_hi(event): # you can rename 'event' to anything you want

tkinter.Label(window, text = "Hi").pack()

btn = tkinter.Button(window, text = "Click Me!")

btn.bind("<Button-1>", say\_hi) # 'bind' takes 2 parameters 1st is 'event' 2nd is 'function'

btn.pack()

window.mainloop()

## Mouse Clicking Events

Clicking events are of 3 different types namely **leftClick**, **middleClick**, and **rightClick**.

Now, you will learn how to call a particular function based on the event that occurs.

* Run the following program and click the **left, middle, right** buttons to calls a specific **function**.
* That **function** will create a new label with the mentioned text.

import tkinter

window = tkinter.Tk()

window.title("GUI")

#creating 3 different functions for 3 events

def left\_click(event):

tkinter.Label(window, text = "Left Click!").pack()

def middle\_click(event):

tkinter.Label(window, text = "Middle Click!").pack()

def right\_click(event):

tkinter.Label(window, text = "Right Click!").pack()

window.bind("<Button-1>", left\_click)

window.bind("<Button-2>", middle\_click)

window.bind("<Button-3>", right\_click)

window.mainloop()

## Drop-Down Menus

I hope all of you know what drop-down menus are. You will create drop-down menus in **tkinter**using the class **Menu**. Follow the below steps to create drop-down menus.

**Steps:-**

* Create a **root menu** to insert different types of **menu options** using **tkinter.Menu(para)** and it takes a parameter where to place the **Menu**
* You have to tell the **tkinter** to initiate **Menu** using **window\_variable.config(menu = para)** and it takes a parameter called **menu** which is the **root menu** you previously defined.
* Now, creating **sub menus** using same method **tkinter.Menu(para)** and it takes the parameter **root menu**.
* **root menu.add\_cascade(para1, menu = para2)** creates the name of the **sub menu**, and it takes 2 parameters one is **label** which is the name of the **sub menu**, and another one is **menu** which is **sub menu**.
* **sub menu.add\_command()** adds an option to the **sub menu**.
* **sub menu.add\_separator()** adds a separator

import tkinter

window = tkinter.Tk()

window.title("GUI")

def function():

pass

# creating a root menu to insert all the sub menus

root\_menu = tkinter.Menu(window)

window.config(menu = root\_menu)

# creating sub menus in the root menu

file\_menu = tkinter.Menu(root\_menu) # it intializes a new su menu in the root menu

root\_menu.add\_cascade(label = "File", menu = file\_menu) # it creates the name of the sub menu

file\_menu.add\_command(label = "New file.....", command = function) # it adds a option to the sub menu 'command' parameter is used to do some action

file\_menu.add\_command(label = "Open files", command = function)

file\_menu.add\_separator() # it adds a line after the 'Open files' option

file\_menu.add\_command(label = "Exit", command = window.quit)

# creting another sub menu

edit\_menu = tkinter.Menu(root\_menu)

root\_menu.add\_cascade(label = "Edit", menu = edit\_menu)

edit\_menu.add\_command(label = "Undo", command = function)

edit\_menu.add\_command(label = "Redo", command = function)

window.mainloop()

## Alert Box

You can create alert boxes in the **tkinter** using **messagebox** method. You can also create **questions**using the **messasgebox** method.

import tkinter

import tkinter.messagebox

window = tkinter.Tk()

window.title("GUI")

# creating a simple alert box

tkinter.messagebox.showinfo("Alert Message", "This is just a alert message!")

# creating a question to get the response from the user [Yes or No Question]

response = tkinter.messagebox.askquestion("Simple Question", "Do you love Python?")

# If user clicks 'Yes' then it returns 1 else it returns 0

if response == 1:

tkinter.Label(window, text = "You love Python!").pack()

else:

tkinter.Label(window, text = "You don't love Python!").pack()

window.mainloop()

## Images And Icons

You can add **Images** and **Icons** using **PhotoImage** method.

import tkinter

window = tkinter.Tk()

window.title("GUI")

# taking image from the directory and storing the source in a variable

icon = tkinter.PhotoImage(file = "images/haha.png")

# displaying the picture using a 'Label' by passing the 'picture' variriable to 'image' parameter

label = tkinter.Label(window, image = icon)

label.pack()

window.mainloop()

## Creating Calculator

Every **GUI** apps include two steps.

* Creating User Interface
* Adding functionalities to the **GUI**

from tkinter import \*

# creating basic window

window = Tk()

window.geometry("312x324") # size of the window width:- 500, height:- 375

window.resizable(0, 0) # this prevents from resizing the window

window.title("Calcualtor")

################################### functions ######################################

# 'btn\_click' function continuously updates the input field whenever you enters a number

def btn\_click(item):

global expression

expression = expression + str(item)

input\_text.set(expression)

# 'btn\_clear' function clears the input field

def btn\_clear():

global expression

expression = ""

input\_text.set("")

# 'btn\_equal' calculates the expression present in input field

def btn\_equal():

global expression

result = str(eval(expression)) # 'eval' function evalutes the string expression directly

# you can also implement your own function to evalute the expression istead of 'eval' function

input\_text.set(result)

expression = ""

expression = ""

# 'StringVar()' is used to get the instance of input field

input\_text = StringVar()

# creating a frame for the input field

input\_frame = Frame(window, width = 312, height = 50, bd = 0, highlightbackground = "black", highlightcolor = "black", highlightthickness = 1)

input\_frame.pack(side = TOP)

# creating a input field inside the 'Frame'

input\_field = Entry(input\_frame, font = ('arial', 18, 'bold'), textvariable = input\_text, width = 50, bg = "#eee", bd = 0, justify = RIGHT)

input\_field.grid(row = 0, column = 0)

input\_field.pack(ipady = 10) # 'ipady' is internal padding to increase the height of input field

# creating another 'Frame' for the button below the 'input\_frame'

btns\_frame = Frame(window, width = 312, height = 272.5, bg = "grey")

btns\_frame.pack()

# first row

clear = Button(btns\_frame, text = "C", fg = "black", width = 32, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_clear()).grid(row = 0, column = 0, columnspan = 3, padx = 1, pady = 1)

divide = Button(btns\_frame, text = "/", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_click("/")).grid(row = 0, column = 3, padx = 1, pady = 1)

# second row

seven = Button(btns\_frame, text = "7", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(7)).grid(row = 1, column = 0, padx = 1, pady = 1)

eight = Button(btns\_frame, text = "8", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(8)).grid(row = 1, column = 1, padx = 1, pady = 1)

nine = Button(btns\_frame, text = "9", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(9)).grid(row = 1, column = 2, padx = 1, pady = 1)

multiply = Button(btns\_frame, text = "\*", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_click("\*")).grid(row = 1, column = 3, padx = 1, pady = 1)

# third row

four = Button(btns\_frame, text = "4", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(4)).grid(row = 2, column = 0, padx = 1, pady = 1)

five = Button(btns\_frame, text = "5", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(5)).grid(row = 2, column = 1, padx = 1, pady = 1)

six = Button(btns\_frame, text = "6", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(6)).grid(row = 2, column = 2, padx = 1, pady = 1)

minus = Button(btns\_frame, text = "-", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_click("-")).grid(row = 2, column = 3, padx = 1, pady = 1)

# fourth row

one = Button(btns\_frame, text = "1", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(1)).grid(row = 3, column = 0, padx = 1, pady = 1)

two = Button(btns\_frame, text = "2", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(2)).grid(row = 3, column = 1, padx = 1, pady = 1)

three = Button(btns\_frame, text = "3", fg = "black", width = 10, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(3)).grid(row = 3, column = 2, padx = 1, pady = 1)

plus = Button(btns\_frame, text = "+", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_click("+")).grid(row = 3, column = 3, padx = 1, pady = 1)

# fourth row

zero = Button(btns\_frame, text = "0", fg = "black", width = 21, height = 3, bd = 0, bg = "#fff", cursor = "hand2", command = lambda: btn\_click(0)).grid(row = 4, column = 0, columnspan = 2, padx = 1, pady = 1)

point = Button(btns\_frame, text = ".", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_click(".")).grid(row = 4, column = 2, padx = 1, pady = 1)

equals = Button(btns\_frame, text = "=", fg = "black", width = 10, height = 3, bd = 0, bg = "#eee", cursor = "hand2", command = lambda: btn\_equal()).grid(row = 4, column = 3, padx = 1, pady = 1)

window.mainloop()

