Python Tkinter

1. import the Tkinter module.
2. Create the main application window.
3. Add the widgets like labels, buttons, frames, etc. to the window.
4. Call the main event loop so that the actions can take place on the user's computer screen.

**from** tkinter **import** \*   
*#creating the application main window.*top = Tk()   
*#Entering the event main loop*top.mainloop()

## **Tkinter widgets**

There are various widgets like button, canvas, checkbutton, entry, etc. that are used to build the python GUI applications.

|  |  |  |
| --- | --- | --- |
| **SN** | **Widget** | **Description** |
| 1 | [Button](https://www.javatpoint.com/python-tkinter-button) | The Button is used to add various kinds of buttons to the python application. |
| 2 | [Canvas](https://www.javatpoint.com/python-tkinter-canvas) | The canvas widget is used to draw the canvas on the window. |
| 3 | [Checkbutton](https://www.javatpoint.com/python-tkinter-checkbutton) | The Checkbutton is used to display the CheckButton on the window. |
| 4 | [Entry](https://www.javatpoint.com/python-tkinter-entry) | The entry widget is used to display the single-line text field to the user. It is commonly used to accept user values. |
| 5 | [Frame](https://www.javatpoint.com/python-tkinter-frame) | It can be defined as a container to which, another widget can be added and organized. |
| 6 | [Label](https://www.javatpoint.com/python-tkinter-label) | A label is a text used to display some message or information about the other widgets. |
| 7 | [ListBox](https://www.javatpoint.com/python-tkinter-listbox) | The ListBox widget is used to display a list of options to the user. |
| 8 | [Menubutton](https://www.javatpoint.com/python-tkinter-menubutton) | The Menubutton is used to display the menu items to the user. |
| 9 | [Menu](https://www.javatpoint.com/python-tkinter-menu) | It is used to add menu items to the user. |
| 10 | [Message](https://www.javatpoint.com/python-tkinter-message) | The Message widget is used to display the message-box to the user. |
| 11 | [Radiobutton](https://www.javatpoint.com/python-tkinter-radiobutton) | The Radiobutton is different from a checkbutton. Here, the user is provided with various options and the user can select only one option among them. |
| 12 | [Scale](https://www.javatpoint.com/python-tkinter-scale) | It is used to provide the slider to the user. |
| 13 | [Scrollbar](https://www.javatpoint.com/python-tkinter-scrollbar) | It provides the scrollbar to the user so that the user can scroll the window up and down. |
| 14 | [Text](https://www.javatpoint.com/python-tkinter-text) | It is different from Entry because it provides a multi-line text field to the user so that the user can write the text and edit the text inside it. |
| 14 | [Toplevel](https://www.javatpoint.com/python-tkinter-toplevel) | It is used to create a separate window container. |
| 15 | [Spinbox](https://www.javatpoint.com/python-tkinter-spinbox) | It is an entry widget used to select from options of values. |
| 16 | [PanedWindow](https://www.javatpoint.com/python-tkinter-panedwindow) | It is like a container widget that contains horizontal or vertical panes. |
| 17 | [LabelFrame](https://www.javatpoint.com/python-tkinter-labelframe) | A LabelFrame is a container widget that acts as the container |
| 18 | [MessageBox](https://www.javatpoint.com/python-tkinter-messagebox) | This module is used to display the message-box in the desktop based applications. |

**from** tkinter **import** \*   
*#creating the application main window.*top = Tk()   
Button(top,text=**"Click me"**)  
top.mainloop()

## **Python Tkinter Geometry**

The Tkinter geometry specifies the method by using which, the widgets are represented on display. The python Tkinter provides the following geometry methods.

1. The pack() method
2. The grid() method
3. The place() method

**from** tkinter **import** \*   
parent = Tk()   
redbutton = Button(parent, text = **"Red"**, fg = **"red"**)   
redbutton.pack( side = LEFT)   
greenbutton = Button(parent, text = **"Black"**, fg = **"black"**)   
greenbutton.pack( side = RIGHT )   
bluebutton = Button(parent, text = **"Blue"**, fg = **"blue"**)   
bluebutton.pack( side = TOP )   
blackbutton = Button(parent, text = **"Green"**, fg = **"red"**)   
blackbutton.pack( side = BOTTOM)   
parent.mainloop()

……………….

**from** tkinter **import** \*   
parent = Tk()   
redbutton = Button(parent, text = **"Red"**, fg = **"red"**)   
redbutton.grid(row=0,column=0)  
greenbutton = Button(parent, text = **"Black"**, fg = **"black"**)   
greenbutton.grid(row=0,column=1)  
bluebutton = Button(parent, text = **"Blue"**, fg = **"blue"**)   
bluebutton.grid(row=1,column=0)  
blackbutton = Button(parent, text = **"Green"**, fg = **"red"**)   
blackbutton.grid(row=1,column=1)  
parent.mainloop()

…………………..

**from** tkinter **import** \*   
parent = Tk()   
redbutton = Button(parent, text = **"Red"**, fg = **"red"**)   
redbutton.place(x=10,y=0)  
greenbutton = Button(parent, text = **"Black"**, fg = **"black"**)   
greenbutton.place(x=10,y=50)  
bluebutton = Button(parent, text = **"Blue"**, fg = **"blue"**)   
bluebutton.place(x=10,y=100)  
blackbutton = Button(parent, text = **"Green"**, fg = **"red"**)   
blackbutton.place(x=100,y=100)  
parent.mainloop()

…………………………..

## When to use the Pack Manager

Compared to the **grid** manager, the pack manager is somewhat limited, but it’s much easier to use in a few, but quite common situations:

1. Put a widget inside a frame (or any other container widget), and have it fill the entire frame
2. Place a number of widgets **on top of each other**
3. Place a number of widgets **side by side**

### Placing a number of widgets on top of each other

To put a number of widgets in a column, you can use the **pack** method without any options:

from Tkinter import \*

root = Tk()

w = Label(root, text="Red", bg="red", fg="white")

w.pack()

w = Label(root, text="Green", bg="green", fg="black")

w.pack()

w = Label(root, text="Blue", bg="blue", fg="white")

w.pack()

mainloop()

To make the widget fill the entire parent, also if the user resizes the window, add **fill** and **expand** options:

from Tkinter import \*

root = Tk()

listbox = Listbox(root)

listbox.pack(fill=BOTH, expand=1)

for i in range(20):

listbox.insert(END, str(i))

mainloop()

You can use the **fill=X** option to make all widgets as wide as the parent widget:

from Tkinter import \*

root = Tk()

w = Label(root, text="Red", bg="red", fg="white")

w.pack(fill=X)

w = Label(root, text="Green", bg="green", fg="black")

w.pack(fill=X)

w = Label(root, text="Blue", bg="blue", fg="white")

w.pack(fill=X)

mainloop()