GUI IN PYTHON

Python Tkinte-

Python provides the standard library Tkinter for creating the graphical user interface for desktop based applications.

Developing desktop based applications with python Tkinter is not a complex task. An empty Tkinter top-level window can be created by using the following steps.

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.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. #creating the application main window.
4. top = Tk()
5. #Entering the event main loop
6. top.mainloop()
```

Output:

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	The Button is used to add various kinds of buttons to the python application.
2	Canvas	The canvas widget is used to draw the canvas on the window.
3	Checkbutton	The Checkbutton is used to display the CheckButton on the window.
4	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	It can be defined as a container to which, another widget can be added and organized.
6	Label	A label is a text used to display some message or information about the other widgets.
7	ListBox	The ListBox widget is used to display a list of options to the user.
8	Menubutton	The Menubutton is used to display the menu items to the user.
9	Menu	It is used to add menu items to the user.
10	Message	The Message widget is used to display the message-box to the user.
11	Radiobutton	The Radiobutton is different from a checkbutton. Here, the user is provided and the user can select only one option among them.

12	Scale	It is used to provide the slider to the user.
13	Scrollbar	It provides the scrollbar to the user so that the user can scroll the window u
14	Text	It is different from Entry because it provides a multi-line text field to the us can write the text and edit the text inside it.
14	Toplevel	It is used to create a separate window container.
15	Spinbox	It is an entry widget used to select from options of values.
16	PanedWindow	It is like a container widget that contains horizontal or vertical panes.
17	LabelFrame	A LabelFrame is a container widget that acts as the container
18	MessageBox	This module is used to display the message-box in the desktop based applic

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

Let's discuss each one of them in detail.

Python Tkinter pack() method

The pack() widget is used to organize widget in the block. The positions widgets added to the python application using the pack() method can be controlled by using the various options specified in the method call.

However, the controls are less and widgets are generally added in the less organized manner.

The syntax to use the pack() is given below.

syntax

1. widget.pack(options)

A list of possible options that can be passed in pack() is given below.

- **expand:** If the expand is set to true, the widget expands to fill any space.
- **Fill:** By default, the fill is set to NONE. However, we can set it to X or Y to determine whether the widget contains any extra space.
- **size:** it represents the side of the parent to which the widget is to be placed on the window.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. parent = Tk()
4. redbutton = Button(parent, text = "Red", fg = "red")
5. redbutton.pack( side = LEFT)
6. greenbutton = Button(parent, text = "Black", fg = "black")
7. greenbutton.pack( side = RIGHT )
8. bluebutton = Button(parent, text = "Blue", fg = "blue")
9. bluebutton.pack( side = TOP )
10. blackbutton = Button(parent, text = "Green", fg = "red")
11. blackbutton.pack( side = BOTTOM)
12. parent.mainloop()
```

Output:

Python Tkinter grid() method

The grid() geometry manager organizes the widgets in the tabular form. We can specify the rows and columns as the options in the method call. We can also specify the column span (width) or rowspan(height) of a widget.

This is a more organized way to place the widgets to the python application. The syntax to use the grid() is given below.

Syntax

1. widget.grid(options)

   A list of possible options that can be passed inside the grid() method is given below.

   - **Column**
     The column number in which the widget is to be placed. The leftmost column is represented by 0.
   - **Columnspan**
     The width of the widget. It represents the number of columns up to which, the column is expanded.
   - **ipadx, ipady**
     It represents the number of pixels to pad the widget inside the widget's border.
   - **padx, pady**
     It represents the number of pixels to pad the widget outside the widget's border.
   - **row**
     The row number in which the widget is to be placed. The topmost row is represented by 0.
   - **rowspan**
     The height of the widget, i.e. the number of the row up to which the widget is expanded.
   - **Sticky**
     If the cell is larger than a widget, then sticky is used to specify the position of the widget inside the cell. It may be the concatenation of the sticky letters representing the position of the widget. It may be N, E, W, S, NE, NW, NS, EW, ES.

Example

```
# !/usr/bin/python3
from tkinter import *
parent = Tk()
name = Label(parent , text = "Name").grid(row = 0, column = 0)
e1 = Entry(parent).grid(row = 0, column = 1)
password = Label(parent,text = "Password").grid(row = 1, column = 0)
e2 = Entry(parent).grid(row = 1, column = 1)
submit = Button(parent, text = "Submit").grid(row = 4, column = 0)
parent.mainloop ()
```

Output:

Python Tkinter place() method

The place() geometry manager organizes the widgets to the specific x and y coordinates.

Syntax

1. widget.place(options)

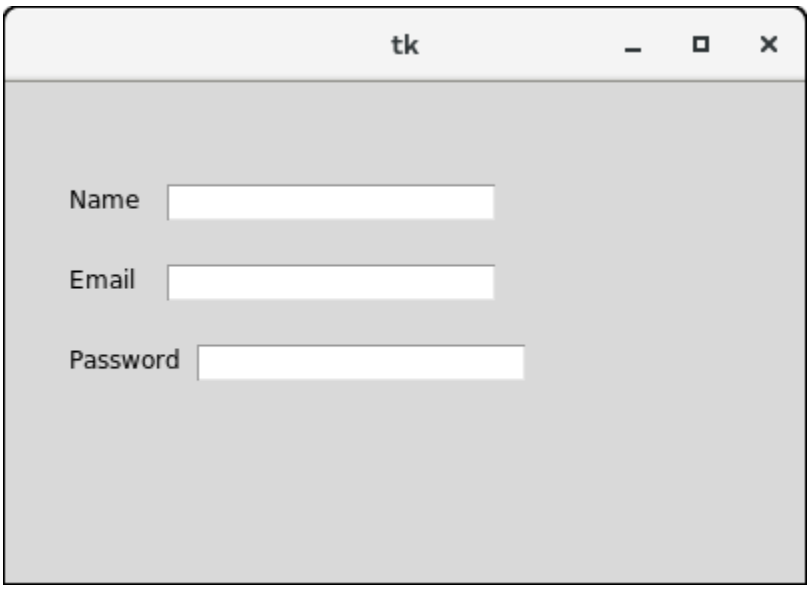
A list of possible options is given below.

- **Anchor:** It represents the exact position of the widget within the container. The default value (direction) is NW (the upper left corner)
- **bordermode:** The default value of the border type is INSIDE that refers to ignore the parent's inside the border. The other option is OUTSIDE.
- **height, width:** It refers to the height and width in pixels.
- **relheight, relwidth:** It is represented as the float between 0.0 and 1.0 indicating the fraction of the parent's height and width.
- **relx, rely:** It is represented as the float between 0.0 and 1.0 that is the offset in the horizontal and vertical direction.
- **x, y:** It refers to the horizontal and vertical offset in the pixels.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. top = Tk()
4. top.geometry("400x250")
5. name = Label(top, text = "Name").place(x = 30,y = 50)
6. email = Label(top, text = "Email").place(x = 30, y = 90)
7. password = Label(top, text = "Password").place(x = 30, y = 130)
8. e1 = Entry(top).place(x = 80, y = 50)
9. e2 = Entry(top).place(x = 80, y = 90)
10. e3 = Entry(top).place(x = 95, y = 130)
11. top.mainloop()
```

Output:



Python Tkinter Button

The button widget is used to add various types of buttons to the python application. Python allows us to configure the look of the button according to our requirements. Various options can be set or reset depending upon the requirements.

We can also associate a method or function with a button which is called when the button is pressed.

The syntax to use the button widget is given below.

Syntax

1. W = Button(parent, options)

A list of possible options is given below.

SN	Option	Description
1	Activebackground	It represents the background of the button when the mouse hover the button.

2	Activeforeground	It represents the font color of the button when the mouse hover the button.
3	Bd	It represents the border width in pixels.
4	Bg	It represents the background color of the button.
5	Command	It is set to the function call which is scheduled when the function is called.
6	Fg	Foreground color of the button.
7	Font	The font of the button text.
8	Height	The height of the button. The height is represented in the number of text lines for the textual lines or the number of pixels for the images.
10	Highlightcolor	The color of the highlight when the button has the focus.
11	Image	It is set to the image displayed on the button.
12	Justify	It illustrates the way by which the multiple text lines are represent It is set to LEFT for left justification, RIGHT for the right justification, and CENTER for the center.
13	Padx	Additional padding to the button in the horizontal direction.
14	Pady	Additional padding to the button in the vertical direction.
15	Relief	It represents the type of the border. It can be SUNKEN, RAISED,

		GROOVE, and RIDGE.
17	State	This option is set to DISABLED to make the button unresponsive. The ACTIVE represents the active state of the button.
18	Underline	Set this option to make the button text underlined.
19	Width	The width of the button. It exists as a number of letters for textual buttons or pixels for image buttons.
20	Wraplength	If the value is set to a positive number, the text lines will be wrapped to fit within this length.

Example

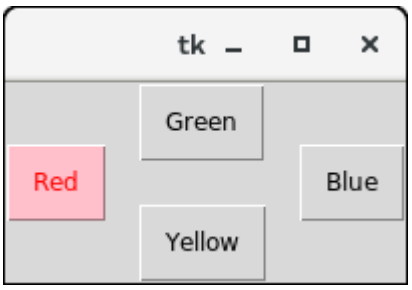
```
#python application to create a simple button

from tkinter import *


top = Tk()

top.geometry("200x100")

b = Button(top,text = "Simple")

b.pack()

top.mainaloop()
```

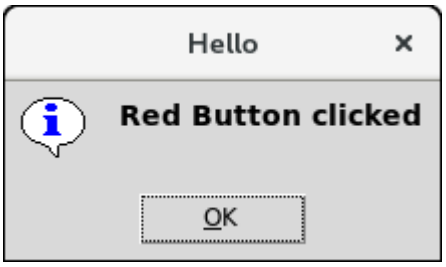
Output:

Example

```
1. from tkinter import *
2.
3. top = Tk()
4.
5. top.geometry("200x100")
6.
7. def fun():
8.     messagebox.showinfo("Hello", "Red Button clicked")
9.
10.
11. b1 = Button(top,text = "Red",command = fun,activeforeground = "red",activebackground =
     "pink",pady=10)
12.
13. b2 = Button(top, text = "Blue",activeforeground = "blue",activebackground = "pink",pady=
    10)
14.
15. b3 = Button(top, text = "Green",activeforeground = "green",activebackground = "pink",pad
    y = 10)
16.
17. b4 = Button(top, text = "Yellow",activeforeground = "yellow",activebackground = "pink",pa
    dy = 10)
18.
19. b1.pack(side = LEFT)
20.
21. b2.pack(side = RIGHT)
22.
23. b3.pack(side = TOP)
24.
25. b4.pack(side = BOTTOM)
26.
27. top.mainloop()
```

Output:





Python Tkinter Canvas

The canvas widget is used to add the structured graphics to the python application. It is used to draw the graph and plots to the python application. The syntax to use the canvas is given below.

Syntax

1. w = canvas(parent, options)

   A list of possible options is given below.

SN	Option	Description
1	bd	The represents the border width. The default width is 2.
2	bg	It represents the background color of the canvas.
3	confine	It is set to make the canvas unscrollable outside the scroll region.

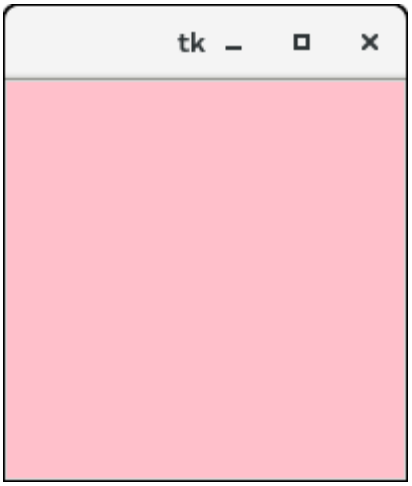
4	cursor	The cursor is used as the arrow, circle, dot, etc. on the canvas.
5	height	It represents the size of the canvas in the vertical direction.
6	highlightcolor	It represents the highlight color when the widget is focused.
7	relief	It represents the type of the border. The possible values are SUNKEN, RAISED, GROOVE, and RIDGE.
8	scrollregion	It represents the coordinates specified as the tuple containing the ar of the canvas.
9	width	It represents the width of the canvas.
10	xscrollincrement	If it is set to a positive value. The canvas is placed only to the multip of this value.
11	xscrollcommand	If the canvas is scrollable, this attribute should be the .set() method of the horizontal scrollbar.
12	yscrollincrement	Works like xscrollincrement, but governs vertical movement.
13	yscrollcommand	If the canvas is scrollable, this attribute should be the .set() method the vertical scrollbar.

Example

```
1. from tkinter import *
2.
3. top = Tk()
4.
5. top.geometry("200x200")
```

```
6.
7.  #creating a simple canvas
8.  c = Canvas(top,bg = "pink",height = "200")
9.
10.
11. c.pack()
12.
13. top.mainloop()
```

Output:



Python Tkinter Canvas

The canvas widget is used to add the structured graphics to the python application. It is used to draw th
python application. The syntax to use the canvas is given below.

Syntax

1. w = canvas(parent, options)

A list of possible options is given below.

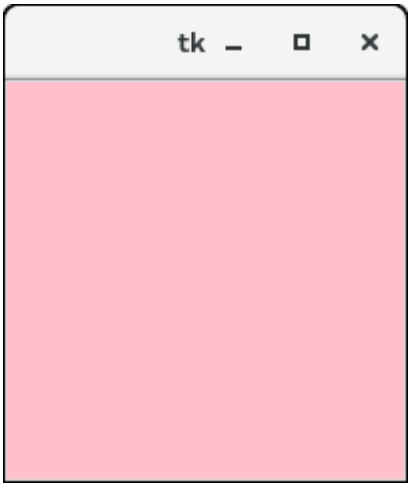
SN	Option	Description
1	bd	The represents the border width. The default width is 2.
2	bg	It represents the background color of the canvas.
3	confine	It is set to make the canvas unscrollable outside the scroll region.
4	cursor	The cursor is used as the arrow, circle, dot, etc. on the canvas.
5	height	It represents the size of the canvas in the vertical direction.
6	Highlightcolor	It represents the highlight color when the widget is focused.
7	relief	It represents the type of the border. The possible values are SUNKEN, RAISED, GROOVE, and RIDGE.
8	scrollregion	It represents the coordinates specified as the tuple containing the of the canvas.
9	width	It represents the width of the canvas.
10	xscrollincrement	If it is set to a positive value. The canvas is placed only to the mu of this value.
11	xscrollcommand	If the canvas is scrollable, this attribute should be the .set() meth of the horizontal scrollbar.
12	yscrollincrement	Works like xscrollincrement, but governs vertical movement.

13	yscrollcommand	If the canvas is scrollable, this attribute should be the .set() meth of the vertical scrollbar.

Example

```
from tkinter import *

top = Tk()

top.geometry("200x200")

#creating a simple canvas
c = Canvas(top,bg = "pink",height = "200")


c.pack()

top.mainloop()
```

Output:

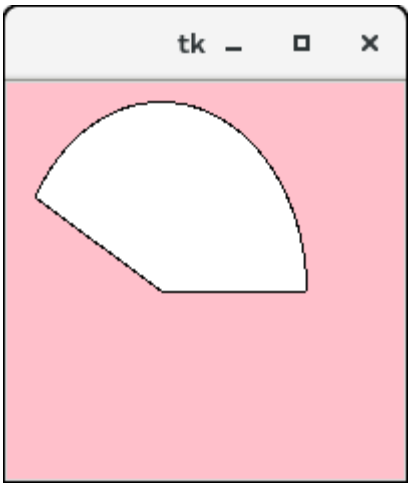


Example: Creating an arc

```
from tkinter import *

```

```
3. top = Tk()
4.
5. top.geometry("200x200")
6.
7. #creating a simple canvas
8. c = Canvas(top,bg = "pink",height = "200",width = 200)
9.
10. arc = c.create_arc((5,10,150,200),start = 0,extent = 150, fill= "white")
11.
12. c.pack()
13.
14. top.mainloop()
```

Output:



Python Tkinter Checkbutton

The Checkbutton is used to track the user's choices provided to the application. In other words, we can say that Checkbutton is used to implement the on/off selections.

The Checkbutton can contain the text or images. The Checkbutton is mostly used to provide many choices to the user among which, the user needs to choose the one. It generally implements many of many selections.

The syntax to use the checkbutton is given below.

Syntax

1. w = checkbutton(master, options)

A list of possible options is given below.

SN	Option	Description
1	activebackground	It represents the background color when the checkbutton is under the cursor.
2	activeforeground	It represents the foreground color of the checkbutton when the checkbutton is under the cursor.
3	bg	The background color of the button.
4	bitmap	It displays an image (monochrome) on the button.
5	bd	The size of the border around the corner.
6	command	It is associated with a function to be called when the state of the checkbutton is changed.
7	cursor	The mouse pointer will be changed to the cursor name when it is ove the checkbutton.
8	disableforeground	It is the color which is used to represent the text of a disabled checkbutton.
9	font	It represents the font of the checkbutton.
10	fg	The foreground color (text color) of the checkbutton.

11	height	It represents the height of the checkbutton (number of lines). The default height is 1.
12	highlightcolor	The color of the focus highlight when the checkbutton is under focus
13	image	The image used to represent the checkbutton.
14	justify	This specifies the justification of the text if the text contains multiple
15	offvalue	The associated control variable is set to 0 by default if the button is unchecked. We can change the state of an unchecked variable to some other one.
16	onvalue	The associated control variable is set to 1 by default if the button is checked. We can change the state of the checked variable to some other one.
17	padx	The horizontal padding of the checkbutton
18	pady	The vertical padding of the checkbutton.
19	relief	The type of the border of the checkbutton. By default, it is set to FLA
20	selectcolor	The color of the checkbutton when it is set. By default, it is red.
21	selectimage	The image is shown on the checkbutton when it is set.
22	state	It represents the state of the checkbutton. By default, it is set to nor We can change it to DISABLED to make the checkbutton unresponsiv

		The state of the checkbutton is ACTIVE when it is under focus.
24	underline	It represents the index of the character in the text which is to be underlined. The indexing starts with zero in the text.
25	variable	It represents the associated variable that tracks the state of the checkbutton.
26	width	It represents the width of the checkbutton. It is represented in the number of characters that are represented in the form of texts.
27	Wraplength	If this option is set to an integer number, the text will be broken into the number of pieces.

Methods

The methods that can be called with the Checkbuttons are described in the following table.

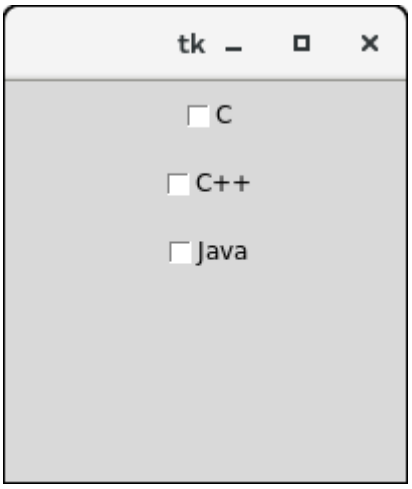
SN	Method	Description
1	deselect()	It is called to turn off the checkbutton.
2	flash()	The checkbutton is flashed between the active and normal colors.
3	invoke()	This will invoke the method associated with the checkbutton.
4	select()	It is called to turn on the checkbutton.
5	toggle()	It is used to toggle between the different Checkbuttons.

Example

```
1. from tkinter import *
```

```
2.
3.  top = Tk()
4.
5.  top.geometry("200x200")
6.
7.  checkvar1 = IntVar()
8.
9.  checkvar2 = IntVar()
10.
11. checkvar3 = IntVar()
12.
13. chkbtn1 = Checkbutton(top, text = "C", variable = checkvar1, onvalue = 1, offvalue = 0, he
    ight = 2, width = 10)
14.
15. chkbtn2 = Checkbutton(top, text = "C++", variable = checkvar2, onvalue = 1, offvalue = 0
    , height = 2, width = 10)
16.
17. chkbtn3 = Checkbutton(top, text = "Java", variable = checkvar3, onvalue = 1, offvalue = 0,
     height = 2, width = 10)
18.
19. chkbtn1.pack()
20.
21. chkbtn2.pack()
22.
23. chkbtn3.pack()
24.
25. top.mainloop()
```

Output:



Python Tkinter Entry

The Entry widget is used to provde the single line text-box to the user to accept a value from the user. We can use the Entry widget to accept the text strings from the user. It can only be used for one line of text from the user. For multiple lines of text, we must use the text widget.

The syntax to use the Entry widget is given below.

Syntax

1. w = Entry (parent, options)

A list of possible options is given below.

SN	Option	Description
1	bg	The background color of the widget.
2	bd	The border width of the widget in pixels.
3	cursor	The mouse pointer will be changed to the cursor type set to the arrow, dot, etc.
4	exportselection	The text written inside the entry box will be automatically copied to the clipboard by default. We can set the exportselection to 0 to not copy this.
5	fg	It represents the color of the text.
6	font	It represents the font type of the text.
7	highlightbackground	It represents the color to display in the traversal highlight region when the widget does not have the input focus.

8	highlightcolor	It represents the color to use for the traversal highlight rectangle that is drawn around the widget when it has the input focus.
9	highlightthickness	It represents a non-negative value indicating the width of the highlight rectangle to draw around the outside of the widget when it has the input focus.
10	insertbackground	It represents the color to use as background in the area covered by the insertion cursor. This color will normally override either the normal background for the widget.
11	insertborderwidth	It represents a non-negative value indicating the width of the 3-D border to draw around the insertion cursor. The value may have any of t to Tk_GetPixels.
12	insertofftime	It represents a non-negative integer value indicating the number of milliseconds the insertion cursor should remain "off" in each blink cycle. zero, then the cursor doesn't blink: it is on all the time.
13	insertontime	Specifies a non-negative integer value indicating the number of milliseconds the insertion cursor should remain "on" in each blink cycle.
14	insertwidth	It represents the value indicating the total width of the insertion cursor. any of the forms acceptable to Tk_GetPixels.
15	justify	It specifies how the text is organized if the text contains multiple lines.
16	relief	It specifies the type of the border. Its default value is FLAT.
17	selectbackground	The background color of the selected text.

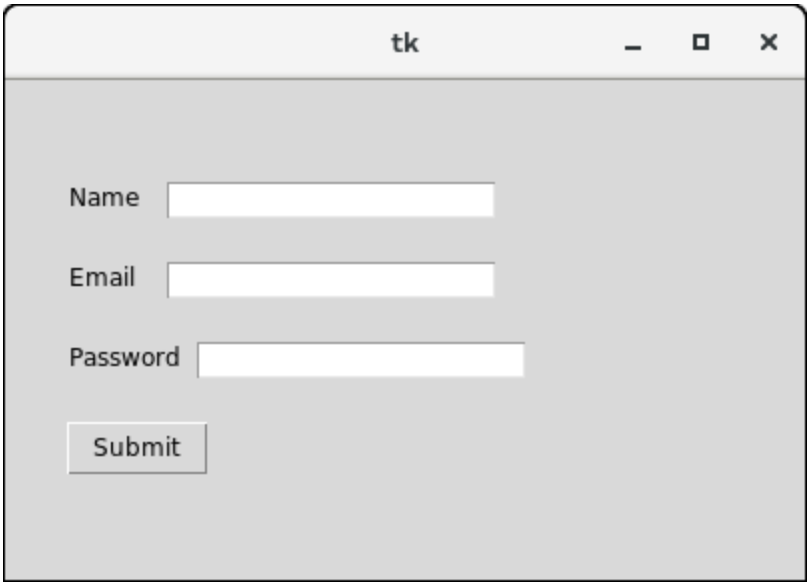
18	selectborderwidth	The width of the border to display around the selected task.
19	selectforeground	The font color of the selected task.
20	show	It is used to show the entry text of some other type instead of the string. For example, the password is typed using stars (*).
21	textvariable	It is set to the instance of the StringVar to retrieve the text from the entry.
22	width	The width of the displayed text or image.
23	xscrollcommand	The entry widget can be linked to the horizontal scrollbar if we want the user to enter more text then the actual width of the widget.

Example

```
# !/usr/bin/python3

from tkinter import *

top = Tk()

top.geometry("400x250")

name = Label(top, text = "Name").place(x = 30,y = 50)

email = Label(top, text = "Email").place(x = 30, y = 90)

password = Label(top, text = "Password").place(x = 30, y = 130)

sbmitbtn = Button(top, text = "Submit",activebackground = "pink", activeforeground = "blue").place(x = 30, y = 170)

e1 = Entry(top).place(x = 80, y = 50)


```

```
20. e2 = Entry(top).place(x = 80, y = 90)
21.
22.
23. e3 = Entry(top).place(x = 95, y = 130)
24.
25. top.mainloop()
```

Output:



Entry widget methods

Python provides various methods to configure the data written inside the widget. There are the following methods provided by the Entry widget.

SN	Method	Description
1	delete(first, last = none)	It is used to delete the specified characters inside the widget.
2	get()	It is used to get the text written inside the widget.
3	icursor(index)	It is used to change the insertion cursor position. We can spec character before which, the cursor to be placed.

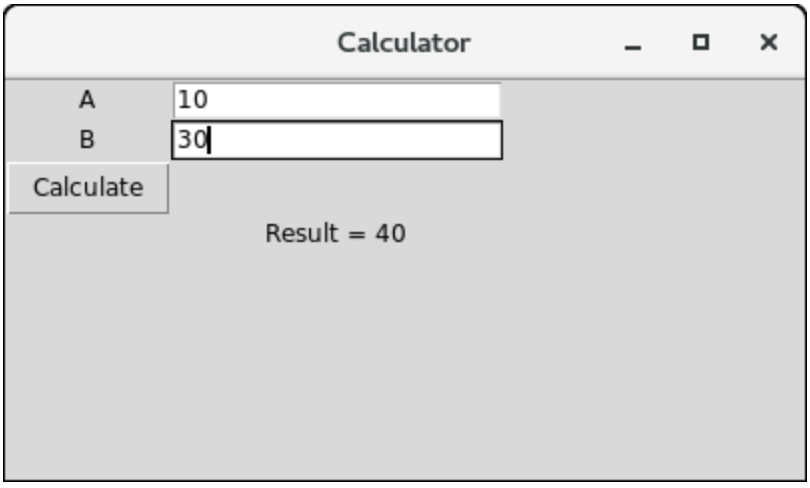
4	index(index)	It is used to place the cursor to the left of the character written index.
5	insert(index,s)	It is used to insert the specified string before the character pla index.
6	select_adjust(index)	It includes the selection of the character present at the specifi
7	select_clear()	It clears the selection if some selection has been done.
8	select_form(index)	It sets the anchor index position to the character specified by t
9	select_present()	It returns true if some text in the Entry is selected otherwise r
10	select_range(start,end)	It selects the characters to exist between the specified range.
11	select_to(index)	It selects all the characters from the beginning to the specified
12	xview(index)	It is used to link the entry widget to a horizontal scrollbar.
13	xview_scroll(number,what)	It is used to make the entry scrollable horizontally.

Example: A simple calculator

```
1. import tkinter as tk
2. from functools import partial
3.
4.
5. def call_result(label_result, n1, n2):
6.     num1 = (n1.get())
7.     num2 = (n2.get())
8.     result = int(num1)+int(num2)
9.     label_result.config(text="Result = %d" % result)
10.    return
11.
12. root = tk.Tk()
13. root.geometry('400x200+100+200')
```

```
14.
15. root.title('Calculator')
16.
17. number1 = tk.StringVar()
18. number2 = tk.StringVar()
19.
20. labelNum1 = tk.Label(root, text="A").grid(row=1, column=0)
21.
22. labelNum2 = tk.Label(root, text="B").grid(row=2, column=0)
23.
24. labelResult = tk.Label(root)
25.
26. labelResult.grid(row=7, column=2)
27.
28. entryNum1 = tk.Entry(root, textvariable=number1).grid(row=1, column=2)
29.
30. entryNum2 = tk.Entry(root, textvariable=number2).grid(row=2, column=2)
31.
32. call_result = partial(call_result, labelResult, number1, number2)
33.
34. buttonCal = tk.Button(root, text="Calculate", command=call_result).grid(row=3, column=0
    )
35.
36. root.mainloop()
```

Output:



Python Tkinter Frame

Python Tkinter Frame widget is used to organize the group of widgets. It acts like a container which can be used to hold the other widgets. The rectangular areas of the screen are used to organize the widgets to the python application.

The syntax to use the Frame widget is given below.

Syntax

1. w = Frame(parent, options)

A list of possible options is given below.

SN	Option	Description
1	bd	It represents the border width.
2	bg	The background color of the widget.
3	cursor	The mouse pointer is changed to the cursor type set to different values like an arrow, dot, etc.
4	height	The height of the frame.
5	highlightbackground	The color of the background color when it is under focus.
6	highlightcolor	The text color when the widget is under focus.
7	highlightthickness	It specifies the thickness around the border when the widget is under the focus.
8	relief	It specifies the type of the border. The default value if FLAT.
9	width	It represents the width of the widget.

Example

```
from tkinter import *

top = Tk()
top.geometry("140x100")
frame = Frame(top)
frame.pack()

leftframe = Frame(top)
leftframe.pack(side = LEFT)

rightframe = Frame(top)
rightframe.pack(side = RIGHT)

btn1 = Button(frame, text="Submit", fg="red",activebackground = "red")
btn1.pack(side = LEFT)

btn2 = Button(frame, text="Remove", fg="brown", activebackground = "brown")
btn2.pack(side = RIGHT)

btn3 = Button(rightframe, text="Add", fg="blue", activebackground = "blue")
btn3.pack(side = LEFT)

btn4 = Button(leftframe, text="Modify", fg="black", activebackground = "white")
btn4.pack(side = RIGHT)

top.mainloop()
```

Output:

Python Tkinter Label

The Label is used to specify the container box where we can place the text or images. This widget is used to provide the message to the user about other widgets used in the python application.

There are the various options which can be specified to configure the text or the part of the text shown in the Label.

The syntax to use the Label is given below.

Syntax

1. w = Label (master, options)

A list of possible options is given below.

SN	Option	Description
1	anchor	It specifies the exact position of the text within the size provided to the widget. The default value is CENTER, which is used to center the text w the specified space.
2	bg	The background color displayed behind the widget.
3	bitmap	It is used to set the bitmap to the graphical object specified so that, the label can represent the graphics instead of text.
4	bd	It represents the width of the border. The default is 2 pixels.
5	cursor	The mouse pointer will be changed to the type of the cursor specified, i arrow, dot, etc.
6	font	The font type of the text written inside the widget.
7	fg	The foreground color of the text written inside the widget.

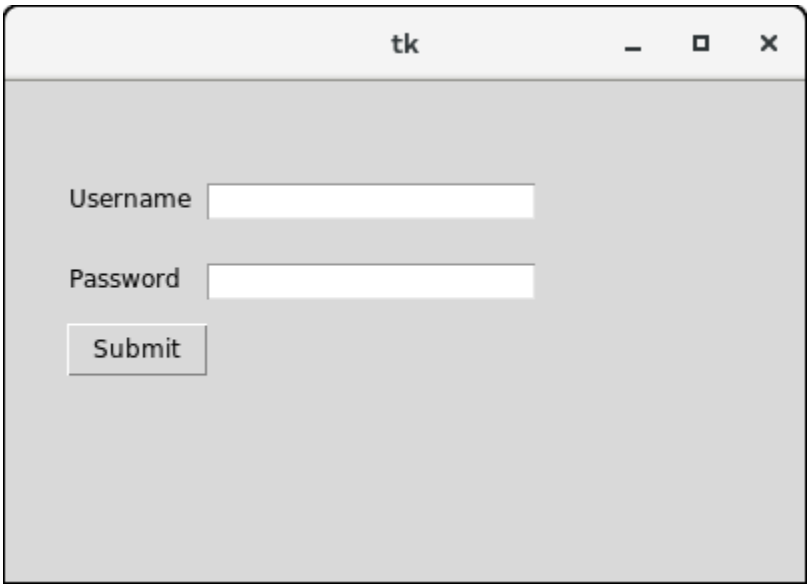
8	height	The height of the widget.
9	image	The image that is to be shown as the label.
10	justify	It is used to represent the orientation of the text if the text contains multiple lines. It can be set to LEFT for left justification, RIGHT for right justification, and CENTER for center justification.
11	padx	The horizontal padding of the text. The default value is 1.
12	pady	The vertical padding of the text. The default value is 1.
13	relief	The type of the border. The default value is FLAT.
14	text	This is set to the string variable which may contain one or more line of text.
15	textvariable	The text written inside the widget is set to the control variable StringVar so that it can be accessed and changed accordingly.
16	underline	We can display a line under the specified letter of the text. Set this opt to the number of the letter under which the line will be displayed.
17	width	The width of the widget. It is specified as the number of characters.
18	wraplength	Instead of having only one line as the label text, we can break it to the number of lines where each line has the number of characters specified to this option.

Example 1

```
1.  # !/usr/bin/python3
```

```
2.
3.  from tkinter import *
4.
5.  top = Tk()
6.
7.  top.geometry("400x250")
8.
9.  #creating label
10. uname = Label(top, text = "Username").place(x = 30,y = 50)
11.
12. #creating label
13. password = Label(top, text = "Password").place(x = 30, y = 90)
14.
15.
16. sbmitbtn = Button(top, text = "Submit",activebackground = "pink", activeforeground = "blue").place(x
17.
18. e1 = Entry(top,width = 20).place(x = 100, y = 50)
19.
20.
21. e2 = Entry(top, width = 20).place(x = 100, y = 90)
22.
23.
24. top.mainloop()
```

Output:



Python Tkinter Listbox

The Listbox widget is used to display the list items to the user. We can place only text items in the Listbox and all text items contain the same font and color.

The user can choose one or more items from the list depending upon the configuration.

The syntax to use the Listbox is given below.

1. w = Listbox(parent, options)

A list of possible options is given below.

SN	Option	Description
1	Bg	The background color of the widget.
2	Bd	It represents the size of the border. Default value is 2 pixel.
3	Cursor	The mouse pointer will look like the cursor type like dot, arrow, etc.
4	Font	The font type of the Listbox items.
5	Fg	The color of the text.
6	Height	It represents the count of the lines shown in the Listbox. The default value is 10.
7	Highlightcolor	The color of the Listbox items when the widget is under focus.
8	Highlightthickness	The thickness of the highlight.
9	Relief	The type of the border. The default is SUNKEN.
10	Selectbackground	The background color that is used to display the selected text.

11	Selectmode	It is used to determine the number of items that can be selected from the list. It can set to BROWSE, SINGLE, MULTIPLE, EXTENDED.
12	Width	It represents the width of the widget in characters.
13	Xscrollcommand	It is used to let the user scroll the Listbox horizontally.
14	Yscrollcommand	It is used to let the user scroll the Listbox vertically.

Methods

There are the following methods associated with the Listbox.

SN	Method	Description
1	activate(index)	It is used to select the lines at the specified index.
2	curselection()	It returns a tuple containing the line numbers of the selected element or elements, counting from 0. If nothing is selected, returns an empty tuple.
3	delete(first, last = None)	It is used to delete the lines which exist in the given range.
4	get(first, last = None)	It is used to get the list items that exist in the given range.
5	index(i)	It is used to place the line with the specified index at the top of the widget.
6	insert(index, *elements)	It is used to insert the new lines with the specified number of elements before the specified index.

7	nearest(y)	It returns the index of the nearest line to the y coordinate of the Listbox widget.
8	see(index)	It is used to adjust the position of the listbox to make the lines specified by the index visible.
9	size()	It returns the number of lines that are present in the Listbox widget.
10	xview()	This is used to make the widget horizontally scrollable.
11	xview_moveto(fraction)	It is used to make the listbox horizontally scrollable by the fraction of width of the longest line present in the listbox.
12	xview_scroll(number, what)	It is used to make the listbox horizontally scrollable by the number of characters specified.
13	yview()	It allows the Listbox to be vertically scrollable.
14	yview_moveto(fraction)	It is used to make the listbox vertically scrollable by the fraction of width of the longest line present in the listbox.
15	yview_scroll (number, what)	It is used to make the listbox vertically scrollable by the number of characters specified.

Example 1

```
1. # !/usr/bin/python3
2.
3. from tkinter import *
4.
5. top = Tk()
```

```
6.
7.  top.geometry("200x250")
8.
9.  lbl = Label(top,text = "A list of favourite countries...")
10.
11. listbox = Listbox(top)
12.
13. listbox.insert(1,"India")
14.
15. listbox.insert(2, "USA")
16.
17. listbox.insert(3, "Japan")
18.
19. listbox.insert(4, "Austrelia")
20.
21. lbl.pack()
22. listbox.pack()
23.
24. top.mainloop()
```

Output:

Example 2: Deleting the active items from the list

```
1.  # !/usr/bin/python3
2.
```

```
from tkinter import *

top = Tk()

top.geometry("200x250")

lbl = Label(top,text = "A list of favourite countries...")

listbox = Listbox(top)

listbox.insert(1,"India")

listbox.insert(2, "USA")

listbox.insert(3, "Japan")

listbox.insert(4, "Austrelia")

#this button will delete the selected item from the list

btn = Button(top, text = "delete", command = lambda listbox=listbox: listbox.delete(ANCHOR))

lbl.pack()


listbox.pack()

btn.pack()
top.mainloop()
```

Output:

After pressing the delete button.

Python Tkinter Menubutton

The Menubutton widget can be defined as the drop-down menu that is shown to the user all the time. It is used to provide the user a option to select the appropriate choice exist within the application.

The Menubutton is used to implement various types of menus in the python application. A Menu is associated with the Menubutton that can display the choices of the Menubutton when clicked by the user.

The syntax to use the python tkinter Menubutton is given below.

Syntax

1. w = Menubutton(Top, options)

A list of various options is given below.

SN	Option	Description
1	Activebackground	The background color of the widget when the widget is under focus.
2	Activeforeground	The font color of the widget text when the widget is under focus.
3	Anchor	It specifies the exact position of the widget content when the widget is assigned needed.
4	Bg	It specifies the background color of the widget.
5	Bitmap	It is set to the graphical content which is to be displayed to the widget.
6	Bd	It represents the size of the border. The default value is 2 pixels.
7	Cursor	The mouse pointer will be changed to the cursor type specified when the widget is under the focus. The possible value of the cursor type is arrow, or dot etc.
8	Direction	It direction can be specified so that menu can be displayed to the specified direction of the button. Use LEFT, RIGHT, or ABOVE to place the widget accordingly.

9	Disabledforegr ound	The text color of the widget when the widget is disabled.
10	Fg	The normal foreground color of the widget.
11	Height	The vertical dimension of the Menubutton. It is specified as the number of lines.
12	Highlightcolor	The highlight color shown to the widget under focus.
13	Image	The image displayed on the widget.
14	Justify	This specified the exact position of the text under the widget when the text is unable to fill the width of the widget. We can use the LEFT for the left justification, RIGHT for the right justification, CENTER for the centre justification.
15	Menu	It represents the menu specified with the Menubutton.
16	Padx	The horizontal padding of the widget.
17	Pady	The vertical padding of the widget.
18	Relief	This option specifies the type of the border. The default value is RAISED.
19	State	The normal state of the Mousebutton is enabled. We can set it to DISABLED to make it unresponsive.
20	Text	The text shown with the widget.
21	Textvariabl e	We can set the control variable of string type to the text variable so that we can control the text of the widget at runtime.

22	Underline	The text of the widget is not underlined by default but we can set this option to make the text of the widget underlined.
23	Width	It represents the width of the widget in characters. The default value is 20.
24	Wraplength	We can break the text of the widget in the number of lines so that the text contains the number of lines not greater than the specified value.

Example

```
# !/usr/bin/python3

from tkinter import *

top = Tk()

top.geometry("200x250")

menubutton = Menubutton(top, text = "Language", relief = FLAT)

menubutton.grid()

menubutton.menu = Menu(menubutton)

menubutton["menu"]=menubutton.menu

menubutton.menu.add_checkbutton(label = "Hindi", variable=IntVar())

menubutton.menu.add_checkbutton(label = "English", variable = IntVar())

menubutton.pack()

top.mainloop()
```

Output:

Python Tkinter Menu

The Menu widget is used to create various types of menus (top level, pull down, and pop up) in the python application.

The top-level menus are the one which is displayed just under the title bar of the parent window. We need to create a new instance of the Menu widget and add various commands to it by using the add() method.

The syntax to use the Menu widget is given below.

Syntax

1. w = Menu(top, options)

A list of possible options is given below.

SN	Option	Description
1	activebackground	The background color of the widget when the widget is under the focus
2	activeborderwidth	The width of the border of the widget when it is under the mouse. The

3	activeforeground	The font color of the widget when the widget has the focus.
4	bg	The background color of the widget.
5	bd	The border width of the widget.
6	cursor	The mouse pointer is changed to the cursor type when it hovers the w type can be set to arrow or dot.
7	disabledforeground	The font color of the widget when it is disabled.
8	font	The font type of the text of the widget.
9	fg	The foreground color of the widget.
10	postcommand	The postcommand can be set to any of the function which is called wh hovers the menu.
11	relief	The type of the border of the widget. The default type is RAISED.
12	image	It is used to display an image on the menu.
13	selectcolor	The color used to display the checkbutton or radiobutton when they ar
14	tearoff	By default, the choices in the menu start taking place from position 1. = 1, then it will start taking place from 0th position.
15	title	Set this option to the title of the window if you want to change the titl

Methods

The Menu widget contains the following methods.

SN	Option	Description
1	add_command(options)	It is used to add the Menu items to the menu.
2	add_radiobutton(options)	This method adds the radiobutton to the menu.
3	add_checkbutton(options)	This method is used to add the checkbuttons to the menu.
4	add_cascade(options)	It is used to create a hierarchical menu to the parent menu by associating the given menu to the parent menu.
5	add_seperator()	It is used to add the seperator line to the menu.
6	add(type, options)	It is used to add the specific menu item to the menu.
7	delete(startindex, endindex)	It is used to delete the menu items exist in the specified range.
8	entryconfig(index, options)	It is used to configure a menu item identified by the given index.
9	index(item)	It is used to get the index of the specified menu item.
10	insert_seperator(index)	It is used to insert a seperator at the specified index.
11	invoke(index)	It is used to invoke the associated with the choice given at the specified index.
12	type(index)	It is used to get the type of the choice specified by the index.

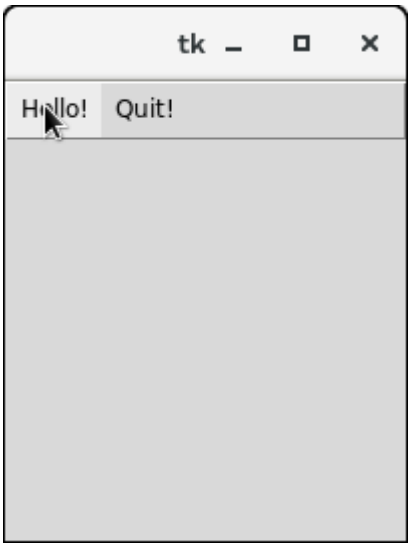
Creating a top level menu

A top-level menu can be created by instantiating the Menu widget and adding the menu items to the menu.

Example 1

```
# !/usr/bin/python3

from tkinter import *

top = Tk()

def hello():
    print("hello!")

# create a toplevel menu
menubar = Menu(root)
menubar.add_command(label="Hello!", command=hello)
menubar.add_command(label="Quit!", command=top.quit)

# display the menu
top.config(menu=menubar)

top.mainloop()
```

Output:

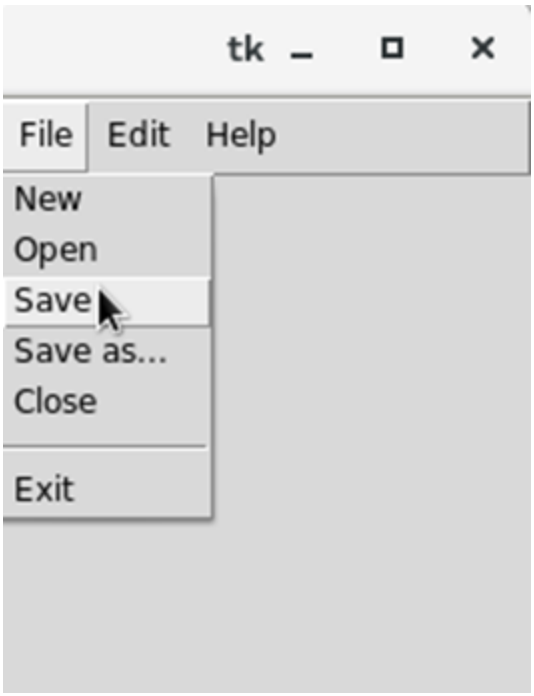


Clicking the hello Menubutton will print the hello on the console while clicking the Quit Menubutton will make an exit from the python application.

Example 2

```
1. from tkinter import Toplevel, Button, Tk, Menu
2. 
3. top = Tk()
4. menubar = Menu(top)
5. file = Menu(menubar, tearoff=0)
6. file.add_command(label="New")
7. file.add_command(label="Open")
8. file.add_command(label="Save")
9. file.add_command(label="Save as...")
10. file.add_command(label="Close")
11. 
12. file.add_separator()
13. 
14. file.add_command(label="Exit", command=top.quit)
15. 
16. menubar.add_cascade(label="File", menu=file)
17. edit = Menu(menubar, tearoff=0)
18. edit.add_command(label="Undo")
19. 
20. edit.add_separator()
21. 
22. edit.add_command(label="Cut")
23. edit.add_command(label="Copy")
24. edit.add_command(label="Paste")
25. edit.add_command(label="Delete")
26. edit.add_command(label="Select All")
27. 
28. menubar.add_cascade(label="Edit", menu=edit)
29. help = Menu(menubar, tearoff=0)
30. help.add_command(label="About")
31. menubar.add_cascade(label="Help", menu=help)
32. 
33. top.config(menu=menubar)
34. top.mainloop()
```

Output:



Python Tkinter Message

The Message widget is used to show the message to the user regarding the behaviour of the python application. The message widget shows the text messages to the user which can not be edited.

The message text contains more than one line. However, the message can only be shown in the single font.

The syntax to use the Message widget is given below.

Syntax

1. w = Message(parent, options)

A list of possible options is given below.

SN	Option	Description
1	anchor	It is used to decide the exact position of the text within the space

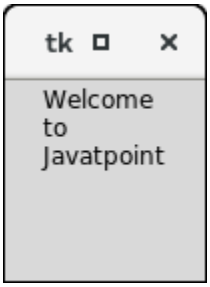
		provided to the widget if the widget contains more space than the need of the text. T CENTER.
2	Bg	The background color of the widget.
3	bitmap	It is used to display the graphics on the widget. It can be set to any graphical or image object.
4	Bd	It represents the size of the border in the pixel. The default size is 2 pixel.
5	cursor	The mouse pointer is changed to the specified cursor type. The cursor type can be an arrow, dot, etc.
6	font	The font type of the widget text.
7	Fg	The font color of the widget text.
8	height	The vertical dimension of the message.
9	image	We can set this option to a static image to show that onto the widget.
10	justify	This option is used to specify the alignment of multiple line of code with respect to each other. The possible values can be LEFT (left alignment), CENTER (default), and RIGHT (right alignment).
11	padx	The horizontal padding of the widget.
12	pady	The vertical padding of the widget.
13	relief	It represents the type of the border. The default type is FLAT.

14	text	We can set this option to the string so that the widget can represent the specified text.
15	textvariable	This is used to control the text represented by the widget. The textvariable can be set to the text that is shown in the widget.
16	underline	The default value of this option is -1 that represents no underline. We can set this option to an existing number to specify that nth letter of the string will be underlined.
17	width	It specifies the horizontal dimension of the widget in the number of characters (not pixel).
18	wraplength	We can wrap the text to the number of lines by setting this option to the desired number so that each line contains only that number of characters.

Example

```
1. from tkinter import *
2.
3. top = Tk()
4. top.geometry("100x100")
5. var = StringVar()
6. msg = Message( top, text = "Welcome to Javatpoint")
7.
8. msg.pack()
9. top.mainloop()
```

Output:



Python Tkinter Radiobutton

The Radiobutton widget is used to implement one-of-many selection in the python application. It shows multiple choices to the user out of which, the user can select only one out of them. We can associate different methods with each of the radiobutton.

We can display the multiple line text or images on the radiobuttons. To keep track the user's selection the radiobutton, it is associated with a single variable. Each button displays a single value for that particular variable.

The syntax to use the Radiobutton is given below.

Syntax

1. w = Radiobutton(top, options)

SN	Option	Description
1	activebackground	The background color of the widget when it has the focus.
2	activeforeground	The font color of the widget text when it has the focus.
3	anchor	It represents the exact position of the text within the widget if the widget contains more space than the requirement of the text. The default value is CENTER.
4	bg	The background color of the widget.

5	bitmap	It is used to display the graphics on the widget. It can be set to any graphical or image object.
6	borderwidth	It represents the size of the border.
7	command	This option is set to the procedure which must be called every-time when the state of the radiobutton is changed.
8	cursor	The mouse pointer is changed to the specified cursor type. It can be set to the arrow, dot, etc.
9	font	It represents the font type of the widget text.
10	fg	The normal foreground color of the widget text.
11	height	The vertical dimension of the widget. It is specified as the number of lines (not pixel).
12	highlightcolor	It represents the color of the focus highlight when the widget has the focus.
13	highlightbackground	The color of the focus highlight when the widget is not having the focus.
14	image	It can be set to an image object if we want to display an image on the radiobutton instead the text.
15	justify	It represents the justification of the multi-line text. It can be set to CENTER(default), LEFT, or RIGHT.
16	padx	The horizontal padding of the widget.

17	pady	The vertical padding of the widget.
18	relief	The type of the border. The default value is FLAT.
19	selectcolor	The color of the radio button when it is selected.
20	selectimage	The image to be displayed on the radiobutton when it is selected.
21	state	It represents the state of the radio button. The default state of the Radiobutton is NORMAL. However, we can set this to DISABLED to make the radiobutton unresponsive.
22	text	The text to be displayed on the radiobutton.
23	textvariable	It is of String type that represents the text displayed by the widget.
24	underline	The default value of this option is -1, however, we can set this option to the number of character which is to be underlined.
25	value	The value of each radiobutton is assigned to the control variable when it is turned on by the user.
26	variable	It is the control variable which is used to keep track of the user's choices. It is shared among all the radiobuttons.
27	width	The horizontal dimension of the widget. It is represented as the number of characters.
28	wraplength	We can wrap the text to the number of lines by setting this option to the desired number so that each line contains only that number of characters.

Methods

The radiobutton widget provides the following methods.

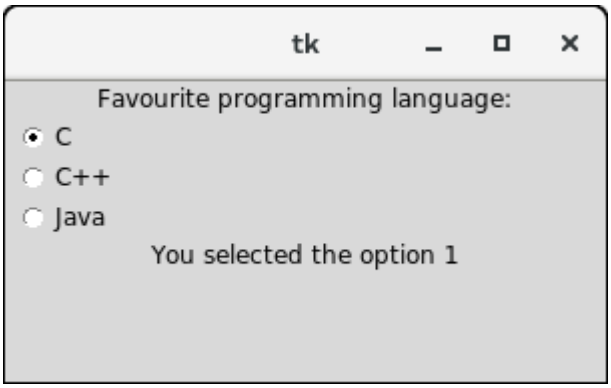
SN	Method	Description
1	deselect()	It is used to turn of the radiobutton.
2	flash()	It is used to flash the radiobutton between its active and normal colors few times.
3	invoke()	It is used to call any procedure associated when the state of a Radiobutton is changed.
4	select()	It is used to select the radiobutton.

Example

```
from tkinter import *

def selection():
   selection = "You selected the option " + str(radio.get())
   label.config(text = selection)

top = Tk()
top.geometry("300x150")
radio = IntVar()
lbl = Label(text = "Favourite programming language:")
lbl.pack()
R1 = Radiobutton(top, text="C", variable=radio, value=1,
                 command=selection)
R1.pack( anchor = W )

R2 = Radiobutton(top, text="C++", variable=radio, value=2,
                 command=selection)
R2.pack( anchor = W )

R3 = Radiobutton(top, text="Java", variable=radio, value=3,
                 command=selection)
```

```
22. R3.pack( anchor = W)
23.
24. label = Label(top)
25. label.pack()
26. top.mainloop()
```

Output:



Python Tkinter Scale

The Scale widget is used to implement the graphical slider to the python application so that the user can slide through the range of values shown on the slider and select the one among them.

We can control the minimum and maximum values along with the resolution of the scale. It provides an alternative to the Entry widget when the user is forced to select only one value from the given range of values.

The syntax to use the Scale widget is given below.

Syntax

```
1. w = Scale(top, options)
```

A list of possible options is given below.

SN	Option	Description
1	activebackground	The background color of the widget when it has the focus.

2	Bg	The background color of the widget.
3	Bd	The border size of the widget. The default is 2 pixel.
4	Command	It is set to the procedure which is called each time when we move the slider. If the slider is moved rapidly, the callback is done when it settles.
5	Cursor	The mouse pointer is changed to the cursor type assigned to this option. It can be an arrow, dot, etc.
6	Digits	If the control variable used to control the scale data is of string type, this option is used to specify the number of digits when the numeric scale is converted to a string.
7	Font	The font type of the widget text.
8	Fg	The foreground color of the text.
9	from_	It is used to represent one end of the widget range.
10	highlightbackground	The highlight color when the widget doesn't have the focus.
11	Highlighcolor	The highlight color when the widget has the focus.
12	Label	This can be set to some text which can be shown as a label with the scale. It is shown in the top left corner if the scale is horizontal or the top right corner if the scale is vertical.

13	Length	It represents the length of the widget. It represents the X dimension if the scale is horizontal or y dimension if the scale is vertical.
14	Orient	It can be set to horizontal or vertical depending upon the type of the scale.
15	Relief	It represents the type of the border. The default is FLAT.
16	Repeatdelay	This option tells the duration up to which the button is to be pressed before the slider starts moving in that direction repeatedly. The default is 300 ms.
17	Resolution	It is set to the smallest change which is to be made to the scale value.
18	Showvalue	The value of the scale is shown in the text form by default. We can set this option to 0 to suppress the label.
19	Sliderlength	It represents the length of the slider window along the length of the scale. The default is 30 pixels. However, we can change it to the appropriate value.
20	State	The scale widget is active by default. We can set this to DISABLED to make it unresponsive.
21	Takefocus	The focus cycles through the scale widgets by default. We can set this option to 0 if we don't want this to happen.
22	Tickinterval	The scale values are displayed on the multiple of the specified tick interval. The default value of the tickinterval is 0.
23	To	It represents a float or integer value that specifies the other end of the range represented by the scale.

24	Troughcolor	It represents the color of the through.
25	Variable	It represents the control variable for the scale.
26	Width	It represents the width of the through part of the widget.

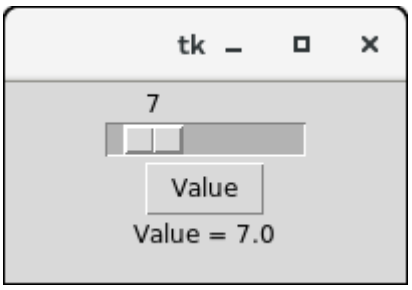
Methods

SN	Method	Description
1	get()	It is used to get the current value of the scale.
2	set(value)	It is used to set the value of the scale.

Example

```
from tkinter import *

def select():
   sel = "Value = " + str(v.get())
   label.config(text = sel)

top = Tk()
top.geometry("200x100")
v = DoubleVar()
scale = Scale( top, variable = v, from_ = 1, to = 50, orient = HORIZONTAL)
scale.pack(anchor=CENTER)

btn = Button(top, text="Value", command=select)
btn.pack(anchor=CENTER)

label = Label(top)
label.pack()

top.mainloop()
```

Output:



Python Tkinter Scrollbar

The scrollbar widget is used to scroll down the content of the other widgets like listbox, text, and canvas. However, we can also create the horizontal scrollbars to the Entry widget.

The syntax to use the Scrollbar widget is given below.

Syntax

1. w = Scrollbar(top, options)

A list of possible options is given below.

SN	Option	Description
1	activebackground	The background color of the widget when it has the focus.
2	bg	The background color of the widget.
3	bd	The border width of the widget.
4	command	It can be set to the procedure associated with the list which can be called each time when the scrollbar is moved.
5	cursor	The mouse pointer is changed to the cursor type set to this option

		which can be an arrow, dot, etc.
6	elementborderwidth	It represents the border width around the arrow heads and slider. The default value is -1.
7	Highlightbackground	The focus highlighcolor when the widget doesn't have the focus.
8	highlighcolor	The focus highlighcolor when the widget has the focus.
9	highlightthickness	It represents the thickness of the focus highlight.
10	jump	It is used to control the behavior of the scroll jump. If it set to 1, then the callback is called when the user releases the mouse button.
11	orient	It can be set to HORIZONTAL or VERTICAL depending upon the orientati
12	repeatdelay	This option tells the duration up to which the button is to be pressed before the slider starts moving in that direction repeatedly. The default is 300 ms.
13	repeatinterval	The default value of the repeat interval is 100.
14	takefocus	We can tab the focus through this widget by default. We can set this option to 0 if we don't want this behavior.
15	troughcolor	It represents the color of the trough.
16	width	It represents the width of the scrollbar.

Methods

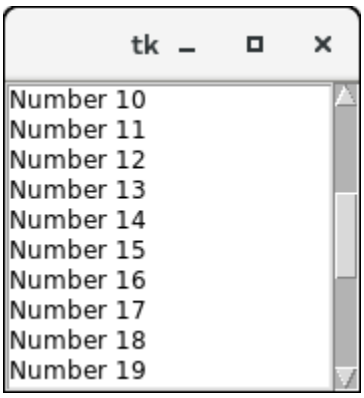
The widget provides the following methods.

SN	Method	Description
1	get()	It returns the two numbers a and b which represents the current position of the scrollbar.
2	set(first, last)	It is used to connect the scrollbar to the other widget w. The yscrollcommand or xscrollcommand of the other widget to this method.

Example

```
from tkinter import *

top = Tk()
sb = Scrollbar(top)
sb.pack(side = RIGHT, fill = Y)

mylist = Listbox(top, yscrollcommand = sb.set )

for line in range(30):
    mylist.insert(END, "Number " + str(line))

mylist.pack( side = LEFT )
sb.config( command = mylist.yview )

mainloop()
```

Output:



Python Tkinter Text

The Text widget is used to show the text data on the Python application. However, Tkinter provides us the Entry widget which is used to implement the single line text box.

The Text widget is used to display the multi-line formatted text with various styles and attributes.

The Text widget is mostly used to provide the text editor to the user.

The Text widget also facilitates us to use the marks and tabs to locate the specific sections of the Text. windows and images with the Text as it can also be used to display the formatted text.

The syntax to use the Text widget is given below.

Syntax

1. w = Text(top, options)

A list of possible options that can be used with the Text widget is given below.

SN	Option	Description
1	Bg	The background color of the widget.

2	Bd	It represents the border width of the widget.
3	cursor	The mouse pointer is changed to the specified cursor type, i.e. arrow, dot, etc.
4	exportselection	The selected text is exported to the selection in the window manager. We can set this to 0 if we don't want the text to be exported.
5	font	The font type of the text.
6	Fg	The text color of the widget.
7	height	The vertical dimension of the widget in lines.
8	highlightbackground	The highlightcolor when the widget doesn't has the focus.
9	highlightthickness	The thickness of the focus highlight. The default value is 1.
10	highlighcolor	The color of the focus highlight when the widget has the focus.
11	insertbackground	It represents the color of the insertion cursor.
12	insertborderwidth	It represents the width of the border around the cursor. The default is 0.
13	insertofftime	The time amount in Milliseconds during which the insertion cursor is off in the blink cycle.
14	insertontime	The time amount in Milliseconds during which the insertion

		cursor is on in the blink cycle.
15	insertwidth	It represents the width of the insertion cursor.
16	padx	The horizontal padding of the widget.
17	pady	The vertical padding of the widget.
18	relief	The type of the border. The default is SUNKEN.
19	selectbackground	The background color of the selected text.
20	selectborderwidth	The width of the border around the selected text.
21	spacing1	It specifies the amount of vertical space given above each line of the text. The default is 0.
22	spacing2	This option specifies how much extra vertical space to add between displayed lines of text when a logical line wraps. The default is 0.
23	spacing3	It specifies the amount of vertical space to insert below each line of the text.
24	state	It the state is set to DISABLED, the widget becomes unresponsive to the mouse and keyboard unresponsive.
25	tabs	This option controls how the tab character is used to position the text.

26	width	It represents the width of the widget in characters.
27	wrap	This option is used to wrap the wider lines into multiple lines. Set this option to the WORD to wrap the lines after the word that fit into the available space. The default value is CHAR which breaks the line which gets too wider at any character.
28	xscrollcommand	To make the Text widget horizontally scrollable, we can set this option to the set() method of Scrollbar widget.
29	yscrollcommand	To make the Text widget vertically scrollable, we can set this option t Scrollbar widget.

Methods

We can use the following methods with the Text widget.

SN	Method	Description
1	delete(startindex, endindex)	This method is used to delete the characters of the specified range.
2	get(startindex, endindex)	It returns the characters present in the specified range.
3	index(index)	It is used to get the absolute index of the specified index.
4	insert(index, string)	It is used to insert the specified string at the given index.
5	see(index)	It returns a boolean value true or false depending upon whether the text at the specified index is visible or not.

Mark handling methods

Marks are used to bookmark the specified position between the characters of the associated text.

SN	Method	Description
1	index(mark)	It is used to get the index of the specified mark.
2	mark_gravity(mark, gravity)	It is used to get the gravity of the given mark.
3	mark_names()	It is used to get all the marks present in the Text widget.
4	mark_set(mark, index)	It is used to inform a new position of the given mark.
5	mark_unset(mark)	It is used to remove the given mark from the text.

Tag handling methods

The tags are the names given to the separate areas of the text. The tags are used to configure the different areas of the text separately. The list of tag-handling methods along with the description is given below.

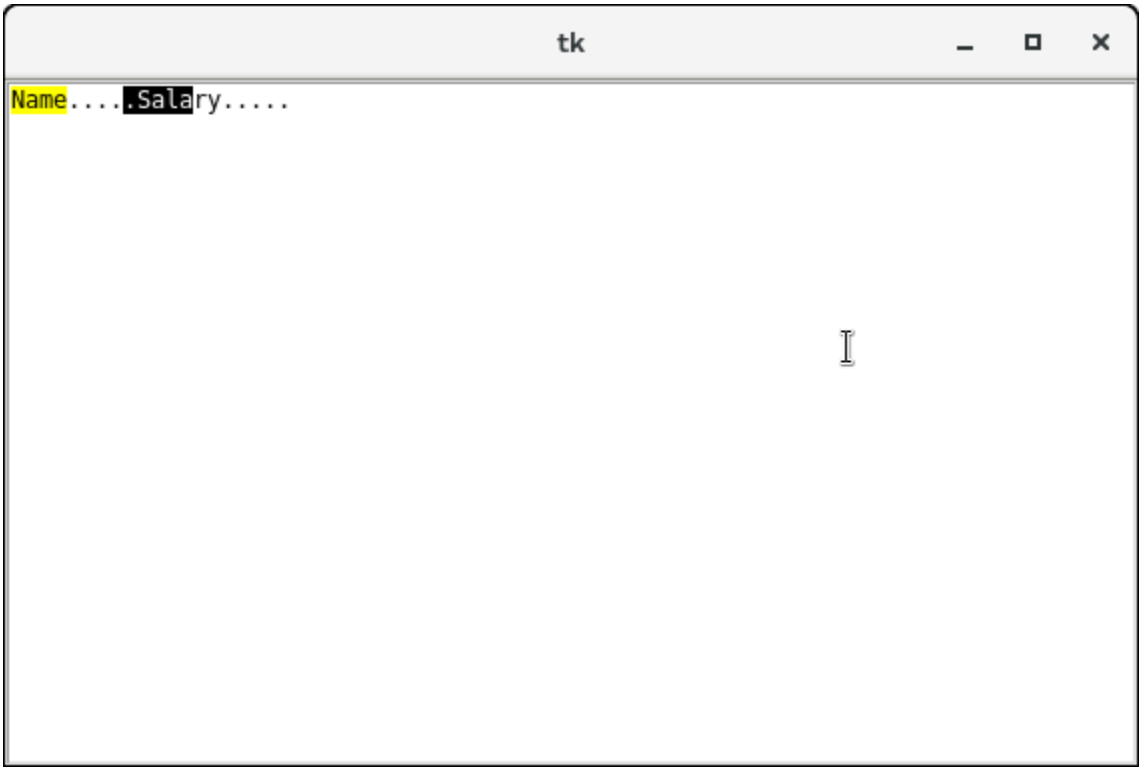
SN	Method	Description
1	tag_add(tagname, startindex, endindex)	This method is used to tag the string present in the specified range.
2	tag_config	This method is used to configure the tag properties.
3	tag_delete(tagname)	This method is used to delete a given tag.

4	tag_remove(tagname, startindex, endindex)	This method is used to remove a tag from the specified range.

Example

```
from tkinter import *

top = Tk()
text = Text(top)
text.insert(INSERT, "Name.....")
text.insert(END, "Salary.....")

text.pack()

text.tag_add("Write Here", "1.0", "1.4")
text.tag_add("Click Here", "1.8", "1.13")

text.tag_config("Write Here", background="yellow", foreground="black")
text.tag_config("Click Here", background="black", foreground="white")

top.mainloop()
```

Output:



Tkinter Toplevel

The Toplevel widget is used to create and display the toplevel windows which are directly managed by the window manager. The toplevel widget may or may not have the parent window on the top of them.

The toplevel widget is used when a python application needs to represent some extra information, pop-up, or the group of widgets on the new window.

The toplevel windows have the title bars, borders, and other window decorations.

The syntax to use the Toplevel widget is given below.

Syntax

1. w = Toplevel(options)

A List of possible options is given below.

SN	Options	Description

1	bg	It represents the background color of the window.
2	bd	It represents the border size of the window.
3	cursor	The mouse pointer is changed to the cursor type set to the arrow, dot, etc. when the mouse is in the window.
4	class_	The text selected in the text widget is exported to be selected to the window manager. We can set this to 0 to make this behavior false.
5	font	The font type of the text inserted into the widget.
6	fg	The foreground color of the widget.
7	height	It represents the height of the window.
8	relief	It represents the type of the window.
9	width	It represents the width of the window,

Methods

The methods associated with the Toplevel widget is given in the following list.

SN	Method	Description
1	deiconify()	This method is used to display the window.
2	frame()	It is used to show a system dependent window identifier.
3	group(window)	It is used to add this window to the specified window group.

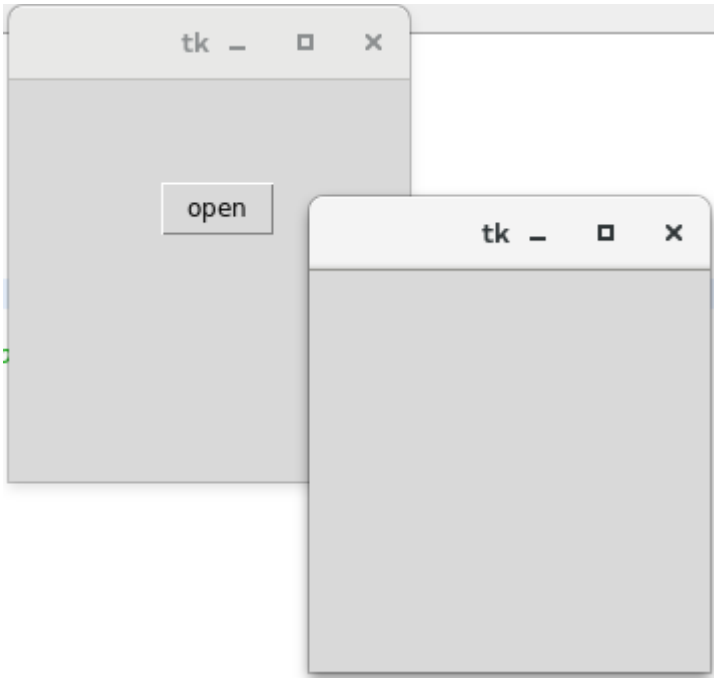
4	iconify()	It is used to convert the toplevel window into an icon.
5	protocol(name, function)	It is used to mention a function which will be called for the specific protocol.
6	state()	It is used to get the current state of the window. Possible values are normal, iconic, withdrawn, and icon.
7	transient([master])	It is used to convert this window to a transient window (temporary).
8	withdraw()	It is used to delete the window but doesn't destroy it.
9	maxsize(width, height)	It is used to declare the maximum size for the window.
10	minsize(width, height)	It is used to declare the minimum size for the window.
11	positionfrom(who)	It is used to define the position controller.
12	resizable(width, height)	It is used to control whether the window can be resizable or not.
13	sizefrom(who)	It is used to define the size controller.
14	title(string)	It is used to define the title for the window.

Example

```
1. from tkinter import *
2.
3. root = Tk()
4.
5. root.geometry("200x200")
6.
```

```
7.  def open():
8.      top = Toplevel(root)
9.      top.mainloop()
10.
11. btn = Button(root, text = "open", command = open)
12.
13. btn.place(x=75,y=50)
14.
15. root.mainloop()
```

Output:



Python Tkinter Spinbox

The Spinbox widget is an alternative to the Entry widget. It provides the range of values to the user, out of which, the user can select the one.

It is used in the case where a user is given some fixed number of values to choose from.

We can use various options with the Spinbox to decorate the widget.

The syntax to use the Spinbox is given below.

Syntax

1. w = Spinbox(top, options)

A list of possible options is given below.

SN	Option	Description
1	activebackground	The background color of the widget when it has the focus.
2	bg	The background color of the widget.
3	bd	The border width of the widget.
4	command	The associated callback with the widget which is called each time the state of the widget is called.
5	cursor	The mouse pointer is changed to the cursor type assigned to this option.
6	disabledbackground	The background color of the widget when it is disabled.
7	disabledforeground	The foreground color of the widget when it is disabled.
8	fg	The normal foreground color of the widget.
9	font	The font type of the widget content.
10	format	This option is used for the format string. It has no default value.

11	from_	It is used to show the starting range of the widget.
12	justify	It is used to specify the justification of the multi-line widget content.
13	relief	It is used to specify the type of the border. The default is SUNKEN.
14	repeatdelay	This option is used to control the button auto repeat. The value is given in milliseconds.
15	repeatinterval	It is similar to repeatdelay. The value is given in milliseconds.
16	state	It represents the state of the widget. The default is NORMAL. The possible values are NORMAL, DISABLED, or "readonly".
17	textvariable	It is like a control variable which is used to control the behaviour of the widget text.
18	to	It specify the maximum limit of the widget value. The other is specified by the from_ option.
19	validate	This option controls how the widget value is validated.
20	validatecommand	It is associated to the function callback which is used for the validation of the widget content.
21	values	It represents the tuple containing the values for this widget.
22	vcmd	It is same as validation command.
23	width	It represents the width of the widget.

24	wrap	This option wraps up the up and down button the Spinbox.
25	xscrollcommand	This options is set to the set() method of scrollbar to make this widget horizontally scrollable.

Methods

There are the following methods associated with the widget.

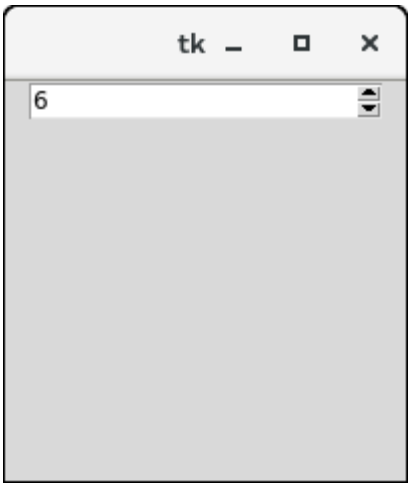
SN	Option	Description
1	delete(startindex, endindex)	This method is used to delete the characters present at the specified range.
2	get(startindex, endindex)	It is used to get the characters present in the specified range.
3	identify(x, y)	It is used to identify the widget's element within the specified range.
4	index(index)	It is used to get the absolute value of the given index.
5	insert(index, string)	This method is used to insert the string at the specified index.
6	invoke(element)	It is used to invoke the callback associated with the widget.

Example

```
from tkinter import *

top = Tk()

top.geometry("200x200")

```

```
7. spin = Spinbox(top, from_= 0, to = 25)
8.
9. spin.pack()
10.
11. top.mainloop()
```

Output:



Tkinter PanedWindow

The PanedWindow widget acts like a Container widget which contains one or more child widgets (panes) arranged horizontally or vertically. The child panes can be resized by the user, by moving the separator lines known as sashes by using the mouse.

Each pane contains only one widget. The PanedWindow is used to implement the different layouts in the python applications.

The syntax to use the PanedWindow is given below.

Syntax

```
1. w= PanedWindow(master, options)
```

A list of possible options is given below.

SN	Option	Description

1	Bg	It represents the background color of the widget when it doesn't have the focus.
2	Bd	It represents the 3D border size of the widget. The default option specifies that the trough contains no border whereas the arrowheads and slider contain the 2-pixel border size.
3	borderwidth	It represents the border width of the widget. The default is 2 pixel.
4	cursor	The mouse pointer is changed to the specified cursor type when it is over the window.
5	handlepad	This option represents the distance between the handle and the end of the sash. For the horizontal orientation, it is the distance between the top of the sash and the handle. The default is 8 pixels.
6	handlesize	It represents the size of the handle. The default size is 8 pixels. However, the handle will always be a square.
7	height	It represents the height of the widget. If we do not specify the height, it will be calculated by the height of the child window.
9	relief	It represents the type of the border. The default is FLAT.
10	sashpad	It represents the padding to be done around each sash. The default is 0.
11	sashrelief	It represents the type of the border around each of the sash. The default is FLAT.

12	sashwidth	It represents the width of the sash. The default is 2 pixels.
13	showhandle	It is set to True to display the handles. The default value is false.
14	Width	It represents the width of the widget. If we don't specify the width of the widget, it will be calculated by the size of the child widgets.

Methods

There are the following methods that are associated with the PanedWindow.

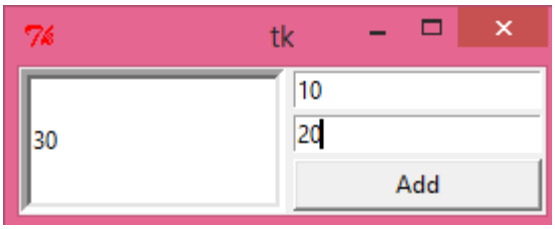
SN	Method	Description
1	add(child, options)	It is used to add a window to the parent window.
2	get(startindex, endindex)	This method is used to get the text present at the specified range.
3	config(options)	It is used to configure the widget with the specified options.

Example

```
# !/usr/bin/python3
from tkinter import *

def add():
    a = int(e1.get())
    b = int(e2.get())
    leftdata = str(a+b)
    left.insert(1,leftdata)

w1 = PanedWindow()
w1.pack(fill = BOTH, expand = 1)

left = Entry(w1, bd = 5)
w1.add(left)

```

```
16. w2 = PanedWindow(w1, orient = VERTICAL)
17. w1.add(w2)
18.
19. e1 = Entry(w2)
20. e2 = Entry(w2)
21.
22. w2.add(e1)
23. w2.add(e2)
24.
25. bottom = Button(w2, text = "Add", command = add)
26. w2.add(bottom)
27.
28. mainloop()
```

Output:



Tkinter LabelFrame

The LabelFrame widget is used to draw a border around its child widgets. We can also display the title for the LabelFrame widget. It acts like a container which can be used to group the number of interrelated widgets such as Radiobuttons.

This widget is a variant of the Frame widget which has all the features of a frame. It also can display a label.

The syntax to use the LabelFrame widget is given below.

Syntax

```
1. w = LabelFrame(top, options)
```

A list of options is given below.

SN	Option	Description
1	bg	The background color of the widget.
2	bd	It represents the size of the border shown around the indicator. The default is 2 pixels.
3	Class	The default value of the class is LabelFrame.
4	colormap	This option is used to specify which colomap is to be used for this widget. By colormap, we mean the 256 colors that are used to form the graphics. With this option, we can reuse the colormap of another window on this widget.
5	container	If this is set to true, the LabelFrame becomes the container widget. The default value is false.
6	cursor	It can be set to a cursor type, i.e. arrow, dot, etc. the mouse pointer is changed to the cursor type when it is over the widget.
7	fg	It represents the foreground color of the widget.
8	font	It represents the font type of the widget text.
9	height	It represents the height of the widget.
10	labelAnchor	It represents the exact position of the text within the widget. The default is NW(north-west)

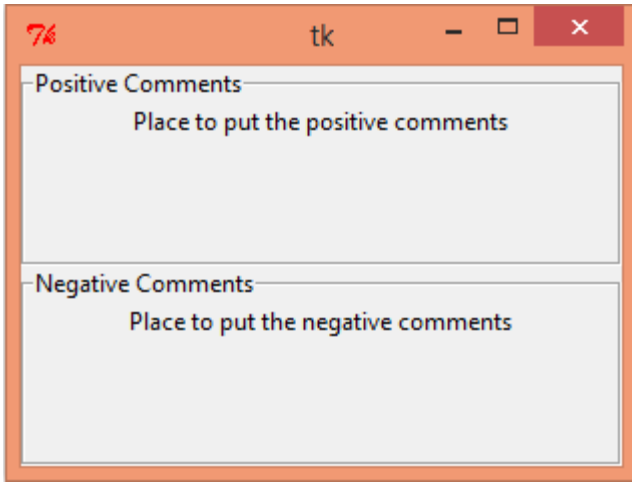
11	labelwidget	It represents the widget to be used for the label. The frame uses the text for the label if no value specified.
12	highlightbackground	The color of the focus highlight border when the widget doesn't have the focus.
13	highlightcolor	The color of the focus highlight when the widget has the focus.
14	highlightthickness	The width of the focus highlight border.
15	padx	The horizontal padding of the widget.
16	pady	The vertical padding of the widget.
17	relief	It represents the border style. The default value is GROOVE.
18	text	It represents the string containing the label text.
19	width	It represents the width of the frame.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3.
4. top = Tk()
5. top.geometry("300x200")
6.
7. labelframe1 = LabelFrame(top, text="Positive Comments")
8. labelframe1.pack(fill="both", expand="yes")
9.
10. toplabel = Label(labelframe1, text="Place to put the positive comments")
11. toplabel.pack()
12.
13. labelframe2 = LabelFrame(top, text = "Negative Comments")
```

```
14. labelframe2.pack(fill="both", expand = "yes")
15.
16. bottomlabel = Label(labelframe2,text = "Place to put the negative comments")
17. bottomlabel.pack()
18.
19. top.mainloop()
```

Output:



Tkinter messagebox

The messagebox module is used to display the message boxes in the python applications. There are the various functions which are used to display the relevant messages depending upon the application requirements.

The syntax to use the messagebox is given below.

Syntax

```
1. messagebox.function_name(title, message [, options])
```

Parameters

- **function_name:** It represents an appropriate message box function.
- **title:** It is a string which is shown as a title of a message box.
- **message:** It is the string to be displayed as a message on the message box.
- **options:** There are various options which can be used to configure the message dialog box.

The two options that can be used are default and parent.

1. default

The default option is used to mention the types of the default button, i.e. ABORT, RETRY, or IGNORE in the message box.

2. parent

The parent option specifies the parent window on top of which, the message box is to be displayed.

There is one of the following functions used to show the appropriate message boxes. All the functions are used with the same syntax but have the specific functionalities.

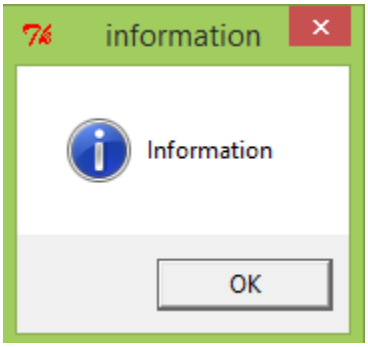
1. showinfo()

The showinfo() messagebox is used where we need to show some relevant information to the user.

Example

```
1. # !/usr/bin/python3
2.
3. from tkinter import *
4.
5. from tkinter import messagebox
6.
7. top = Tk()
8.
9. top.geometry("100x100")
10.
11. messagebox.showinfo("information","Information")
12.
13. top.mainloop()
```

Output:



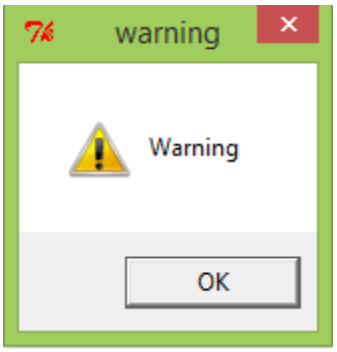
2. showwarning()

This method is used to display the warning to the user. Consider the following example.

Example

```
1.  # !/usr/bin/python3
2.  from tkinter import *
3.
4.  from tkinter import messagebox
5.
6.  top = Tk()
7.  top.geometry("100x100")
8.  messagebox.showwarning("warning","Warning")
9.
10. top.mainloop()
```

Output:



3. showerror()

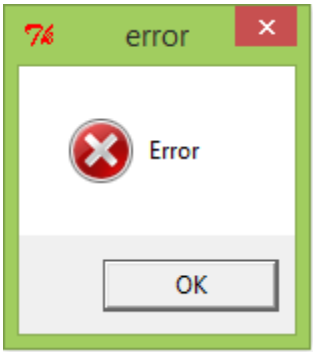
This method is used to display the error message to the user. Consider the following example.

Example

```
1.  # !/usr/bin/python3
2.  from tkinter import *
3.  from tkinter import messagebox
4.
5.  top = Tk()
```

```
6. top.geometry("100x100")
7. messagebox.showerror("error","Error")
8. top.mainloop()
```

Output:



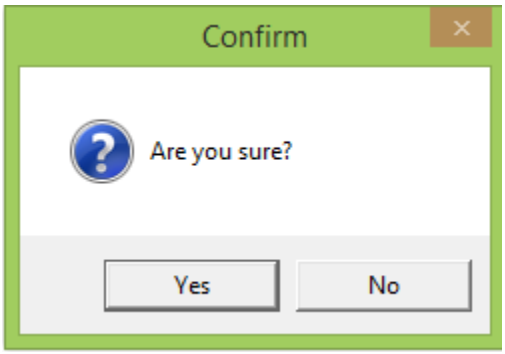
4. askquestion()

This method is used to ask some question to the user which can be answered in yes or no. Consider the following example.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. from tkinter import messagebox
4.
5. top = Tk()
6. top.geometry("100x100")
7. messagebox.askquestion("Confirm","Are you sure?")
8. top.mainloop()
```

Output:



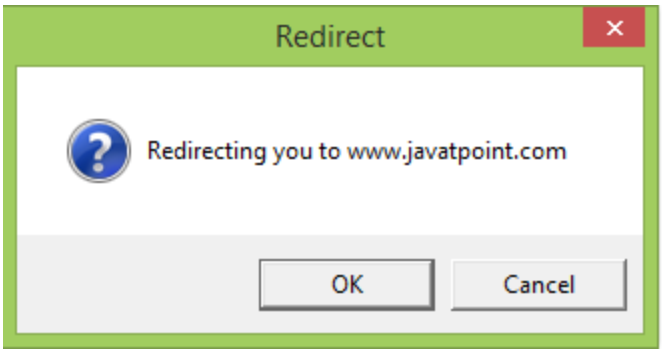
5. askokcancel()

This method is used to confirm the user's action regarding some application activity. Consider the following example.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. from tkinter import messagebox
4.
5. top = Tk()
6. top.geometry("100x100")
7. messagebox.askokcancel("Redirect","Redirecting you to www.javatpoint.com")
8. top.mainloop()
```

Output:



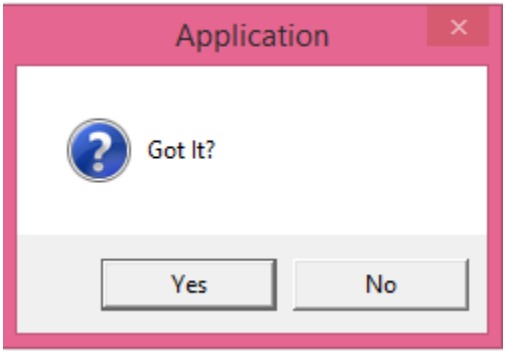
6. askyesno()

This method is used to ask the user about some action to which, the user can answer in yes or no. Consider the following example.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. from tkinter import messagebox
4.
5. top = Tk()
6. top.geometry("100x100")
7. messagebox.askyesno("Application","Got It?")
8. top.mainloop()
```

Output:



7. askretrycancel()

This method is used to ask the user about doing a particular task again or not. Consider the following example.

Example

```
1. # !/usr/bin/python3
2. from tkinter import *
3. from tkinter import messagebox
4.
5. top = Tk()
6. top.geometry("100x100")
7. messagebox.askretrycancel("Application","try again?")
8.
9. top.mainloop()
```

Output: