Unit-6
GUI Programming in Python using Tkinter

Mr. Nilesh Parmar

Assistant Professor, Dept. of Computer Engg.

UVPCE, Ganpat University, Mehsana

Tkinter

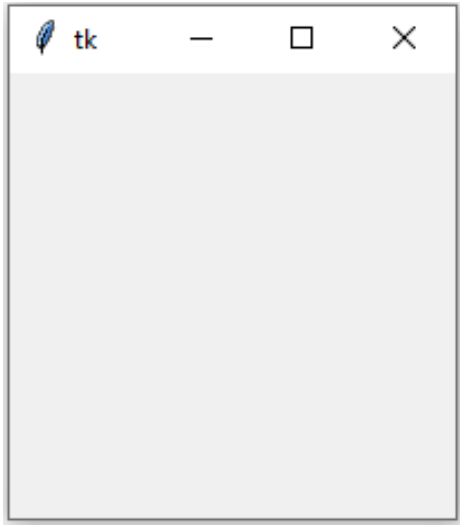
- Tkinter is an acronym for "Tk interface".
- Python provides the standard library ‘Tkinter’ for creating the graphical user interface for desktop based applications.
- ‘Tkinter’ is also get installed when you install python software.
- An empty Tkinter top-level window can be created by using the following steps.

1) import the Tkinter module.
2) Create the main application window (container).
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.

First GUI Program

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

Output:



Tkinter Widgets

- Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called as **widgets.**

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.

Cntd…

SN	Widget	Description
9	Menu	It is used to add menu items to the user.
10	Message	The Message widget is used to display the non-editable message to the user.
11	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	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 up and down.
14	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.
15	Toplevel	It is used to create a separate window container.
16	Spinbox	It is an entry widget used to select from options of values.
17	PanedWindow	It is like a container widget that contains horizontal or vertical panes.
18	LabelFrame	A LabelFrame is a container widget that acts as the container
19	MessageBox	This module is used to display the message-box in the desktop based applications.

Tkinter Geometry Managers

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

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

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.
- **Syntax of pack():** widget.pack(options)
- Options:

1) Expand: If the expand is set to true, the widget expands to fill any space.
2) 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.
3) Side: it represents the side of the parent to which the widget is to be placed on the window.

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 columnspan (width) or rowspan(height) of a widget.
- This is a more organized way to place the widgets to the python application.
- **Syntax of grid():** widget.grid(options)

Options in grid() method

1) column: The column number in which the widget is to be placed. The leftmost column is represented by 0.
2) columnspan: The width of the widget. It represents the number of columns up to which, the column is expanded.
3) ipadx, ipady: It represents the number of pixels to pad the widget inside the widget's border.
4) padx, pady :It represents the number of pixels to pad the widget outside the widget's border.
5) row: The row number in which the widget is to be placed. The topmost row is represented by 0.
6) rowspan: The height of the widget, i.e. the number of the row up to which the widget is expanded.
7) 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.

place() method

- The place() geometry manager organizes the widgets to the specific x and y coordinates.
- **Syntax of place() method:** widget.place(options)
- Options:

1) Anchor: It represents the exact position of the widget within the container. The default value (direction) is NW (the upper left corner)
2) 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.
3) height, width: It refers to the height and width in pixels.
4) relheight, relwidth: It is represented as the float between 0.0 and 1.0 indicating the fraction of the parent's height and width.
5) relx, rely: It is represented as the float between 0.0 and 1.0 that is the offset in the horizontal and vertical direction.
6) x, y: It refers to the horizontal and vertical offset in the pixels.

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 by setting or resetting various options.
- We can also associate a method or function with a button which is called when the button is pressed.
- **Syntax:** w = Button(parent_window, options)
- *Some of the Options:*

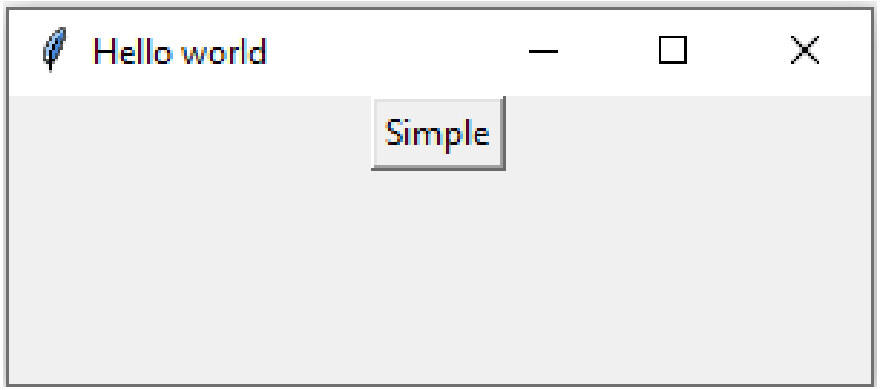
1) activebackground: to set the background color when button is under the cursor.
2) activeforeground: to set the text color when button is under the cursor.
3) fg: to set foreground color of the button.
4) bg: to set background color.
5) command: to call a function.
6) width: to set the width of the button.
7) height: to set the height of the button.
8) text: To specify the text to be written on the button

Cntd…

- Example-1:

```
from tkinter import *
top = Tk()
top.title("Hello world")
top.geometry("300x100")
b = Button(top, text = "Simple")
b.pack()
top.mainloop()
```

Output:



Cntd…

- Example-2:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
def fun():
    print("hello")
b1 = Button(top,text = "Red",bg="yellow",command=fun,activeforeground ="red", activebackground = "pink",pady=20)
b2 = Button(top, text = "Blue",fg="red",activeforeground = "blue",activebackground = "pink", pady=10)
b1.pack(side = LEFT)
b2.pack(side = TOP, fill=X)
top.mainloop()
```

Output:

Canvas

- The canvas widget is used to add the structured graphics like graphs, drawings etc. to the python application.
- Syntax: w = Canvas(parent, options)
- *Some of the Options:*

1) bd: to set the border width in pixels.
2) bg: to set the normal background color.
3) cursor: to set the cursor used in the canvas.
4) highlightcolor: to set the color shown in the focus highlight.
5) width: to set the width of the widget.
6) height: to set the height of the widget.
7) confine: It is set to make the canvas unscrollable outside the scroll region.
8) scrollregion: It represents the coordinates specified as the tuple containing the area of the canvas.

Methods associated with Canvas

1) arc − Creates an arc item, which can be a chord, a pieslice or a simple arc.

- Exmple:

```
coord = 10, 50, 240, 210
arc = canvas.create_arc(coord, start=0, extent=150, fill="blue")
```

2) image − Creates an image item, which can be an instance of either the BitmapImage or the PhotoImage classes.

- Example:

```
filename = PhotoImage(file = "sunshine.gif")
image = canvas.create_image(50, 50, anchor=NE, image=filename)
```

Cntd…

3) line − Creates a line item.

- Example:

line = canvas.create_line(x0, y0, x1, y1, ..., xn, yn, options)

4) oval − Creates a circle or an ellipse at given coordinates. It takes two pairs of coordinates; the top left and bottom right corners of bounding rectangle for oval.

- Example: oval = canvas.create_oval(x0, y0, x1, y1, options)

5) polygon − Creates a polygon item that must have at least three vertices.

- Example:

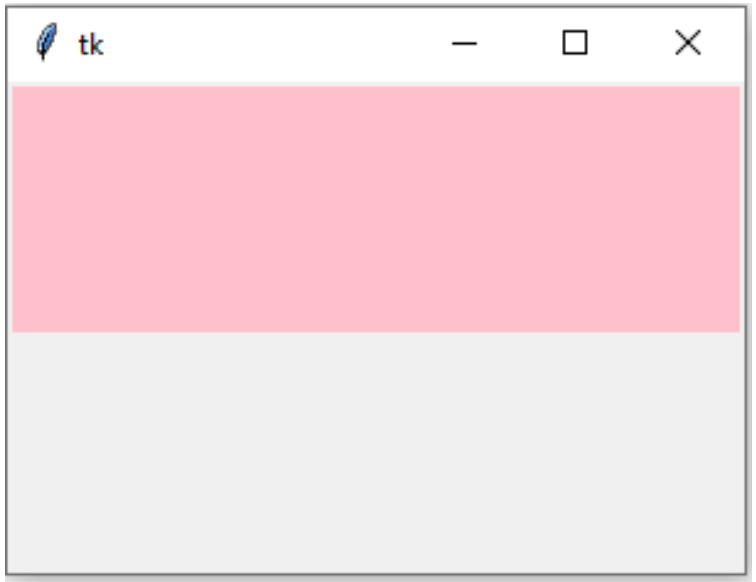
oval = canvas.create_polygon(x0, y0, x1, y1,...xn, yn, options)

Cntd…

- Example-1: creating a canvas

```
from tkinter import *
top = Tk()
top.geometry("300x200")
c= Canvas (top,bg="pink", height="100")
c.pack()
top.mainloop()
```

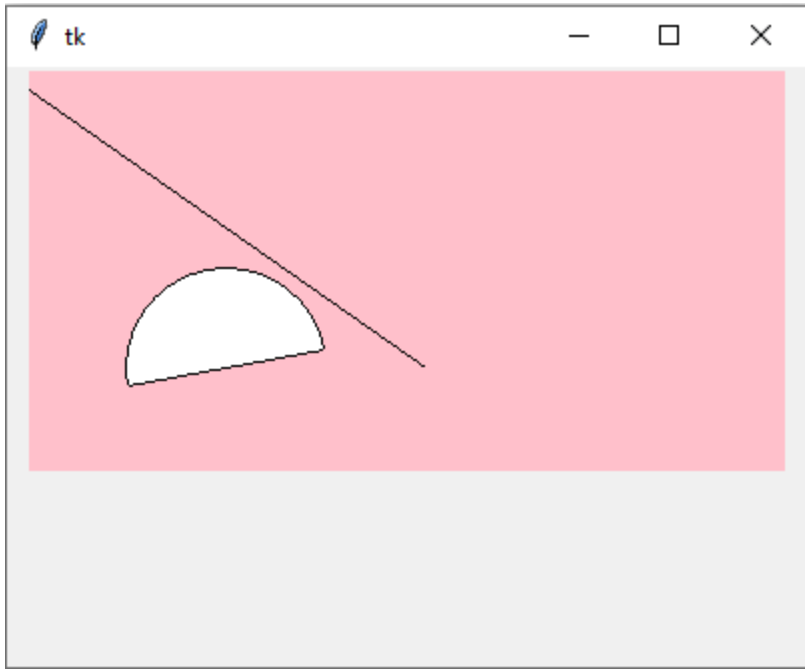
Output:



Cntd…

- Example-2:

Output:




```
from tkinter import *
top = Tk()
top.geometry("400x300")
c= Canvas (top,bg="pink",height="200")
c.create_arc((50,100,150,200),start=10,extent=180,fill="white")
c.create_line(0,10,200,150 )
c.pack()
top.mainloop()
```

Checkbutton

- 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.
- Checkbutton can contain the text or images.
- Syntax: w = Checkbutton(parent, options)
- *Some of Options:*

1) command: It is associated with a function to be called when the state of the checkbutton is changed.
2) cursor: The mouse pointer will be changed to the cursor name when it is over the checkbutton.
3) 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.
4) 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.
5) variable: It represents the associated variable that tracks the state of the checkbutton.

Methods of Checkbutton

Sr No	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.

Cntd…

- Example:

```
from tkinter import *
top = Tk()
top.geometry("400x300")
val1=IntVar()
val2=IntVar()
val3=IntVar()
chk1= Checkbutton(top,text="c",variable=val1,onvalue=1,offvalue=0)
chk2= Checkbutton(top,text="c++",variable=val2,onvalue=1,offvalue=0)
chk3= Checkbutton(top,text="java",variable=val3,onvalue=1,offvalue=0)
chk1.pack()
chk2.pack()
chk3.pack()
top.mainloop()
```

Output:

Entry

- The Entry widget is used to provide the single line text-box to the user to accept a value from the user.
- For multiple lines of text, we must use the **text** widget.
- **Syntax:** w = Entry (parent, options)
- *Some of the Options:*

1) bd: To set the border width in pixels.
2) bg: To set the normal background color.
3) cursor: The mouse pointer will be changed to the cursor type set to the arrow, dot, etc.
4) command: To call a function.
5) highlightcolor: To set the border color when widget is in focus. It works with highlightthickness parameter.
6) width: To set the width of the displayed text or image inside Entry widget.

Some of the Methods of Entry

Sr No	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 specify the index of the 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 at the specified index.
5	insert(index,s)	It is used to insert the specified string before the character placed at the specified index.
6	select_adjust(index)	It includes the selection of the character present at the specified index.
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 the index.

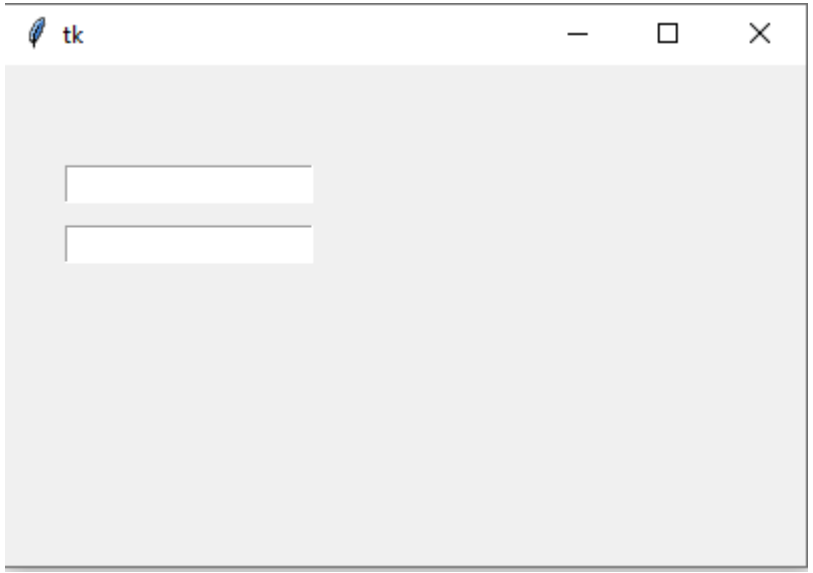
Note: We will see example about how to use these methods, after learning Label widget.

Cntd…

- Example:

```
from tkinter import *
top = Tk()
top.geometry("400x250")
e1=Entry(top)
e2=Entry(top)
e1.place(x=30,y=50)
e2.place(x=30,y=80)
top.mainloop()
```

Output:



Label

- The Label is used to specify the container box where we can place the text or images. Here, text may be the message to the user about other widgets or some information required to display.
- **Syntax:** w = Label (parent, options)
- *Some of the options:*

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 within the specified space.
2) bg: The background color displayed behind the widget.
3) fg: The foreground color of the text written inside the widget.
4) width:The width of the widget. It is specified as the number of characters.
5) height: The height of the widget.
6) image: The image that is to be shown as the label.

Cntd…

- Example:

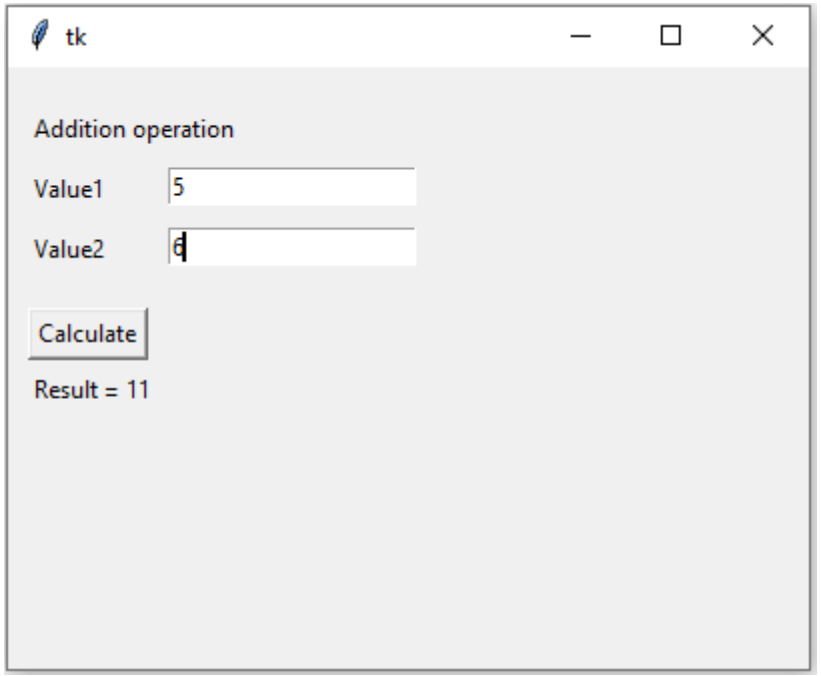
```
from tkinter import *
top = Tk()
top.geometry("400x250")
l1=Label(top,bg="red",text="Hello World")
l1.place(x=30,y=50)
top.mainloop()
```

Output:

- Example: Addition operation

```
from tkinter import *
top = Tk()
top.geometry("400x300")
l1=Label(top,text="Addition operation")
l2=Label(top,text="Value1")
l3=Label(top,text="Value2")
l4=Label(top,text='m')
n1=StringVar()
n2=StringVar()
e1=Entry(top,textvariable=n1)
e2=Entry(top,textvariable=n2)
l1.place(x=10,y=20)
l2.place(x=10,y=50)
l3.place(x=10,y=80)
l4.place(x=10,y=150)
e1.place(x=80,y=50)
e2.place(x=80,y=80)
def disp():
    print('hi')
    num1 = (n1.get())
    num2 = (n2.get())
    result = int(num1)+int(num2)
    l4.config(text="Result = %d" % result)
b=Button(top, text="Calculate",command=disp).place(x=10,y=120)
top.mainloop()
```

Output:



Frame

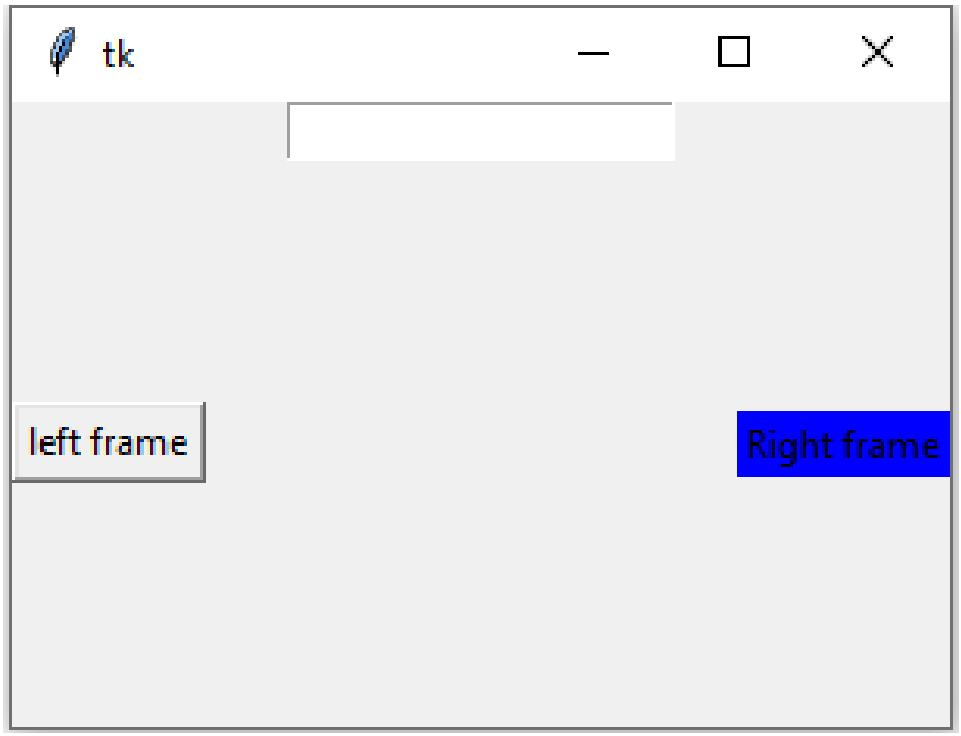
- Python Tkinter Frame widget is used to organize the group of widgets. It acts like a container which can be used to hold the other widgets.
- **Syntax:** w = Frame(parent, options)
- *Some of the options:*

1) bd: It represents the border width of widget contained by frame.
2) bg: The background color of the widget contained by frame.
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) width: It represents the width of the widget.

- Example:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
frame=Frame(top)
frame.pack()
t= Entry(frame)
t.pack()
lframe=Frame(top)
lframe.pack(side=LEFT)
b=Button(lframe,text="left frame")
b.pack()
rframe=Frame(top)
rframe.pack(side=RIGHT)
l=Label(rframe,text="Right frame",bg="blue")
l.pack()
top.mainloop()
```

Output:



Listbox

- The Listbox widget is used to display the list of items. 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.
- **Syntax:** w = Listbox(parent, options)
- *Some of the options:*
- bg: The background color of the widget.
- bd: It represents the size of the border. Default value is 2 pixel.
- cursor: The mouse pointer will look like the cursor type like dot, arrow, etc.
- font: The font type of the Listbox items.
- fg: The color of the text.
- height : It represents count of the lines shown in the Listbox. Default value is 10.

Some of the methods of 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.

Cntd…

- Example: Inserting Items in ListBox

```
from tkinter import *
top = Tk()
top.geometry("200x250")
lbl=Label(top,text="List of countries:")
lbl.grid(row=0,column=0)
listbox=Listbox(top)
listbox.insert(1,"India")
listbox.insert(2,"USA")
listbox.insert(3,"Russia")
listbox.insert(4,"Japan")
listbox.grid(row=1,column=0)
top.mainloop()
```

Output:

• Example: Continuing previous example, now we will delete selected data items

```
from tkinter import *
top = Tk()
top.geometry("200x250")
lbl=Label(top,text="List of countries:")
lbl.grid(row=0,column=0)
lstbox=Listbox(top)
lstbox.insert(1,"India")
lstbox.insert(2,"USA")
lstbox.insert(3,"Russia")
lstbox.insert(4,"Japan")
lstbox.grid(row=1,column=0)
b= Button(top, text="click", command= lambda t=lstbox: lstbox.delete(ANCHOR))
b.grid(row=2,column=0)
top.mainloop()
```

Output:

MenuButton

- The Menubutton widget is drop-down menu that is shown to the user all the time. It is used to provide the user an option to select the appropriate choice.
- The Menubutton is used to implement various types of menus. A Menu is associated with the Menubutton that can display the choices of the Menubutton when clicked by the user.
- **Syntax:** w = Menubutton(top, options)
- *Some of the options:*

1) anchor: It specifies the exact position of the widget content when the widget is assigned more space than needed.

2) bg: It specifies the background color of the widget.

3) bitmap: It is set to the graphical content which is to be displayed to the widget.

4) bd: It represents the size of the border. The default value is 2 pixels.

5) fg: The normal foreground color of the widget.

Cntd…

- Example:

```
from tkinter import *
top = Tk()
top.geometry("200x250")
mb= Menubutton(top,text="file")
mb.menus=Menu(mb)
mb["menu"]=mb.menus
mb.menus.add_checkbutton(label="Hindi",variable=IntVar())
mb.menus.add_checkbutton(label="Gujarati",variable=IntVar())
mb.menus.add_checkbutton(label="English",variable=IntVar())
mb.grid()
top.mainloop()
```

Output:

Menu

- The Menu widget is used to create various types of menus: top level, pull down, and pop up.
- 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.
- **Syntax:** w = Menu(top, options)
- *Some of the options:*
- 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 default is 1 pixel.
- 3) activeforeground: The font color of the widget when the widget has the focus.
- 4) bg: The background color of the widget.
- 5) cursor:The mouse pointer is changed to the cursor type when it hovers the widget. The cursor type can be set to arrow or dot.

Some of the methods of Menu

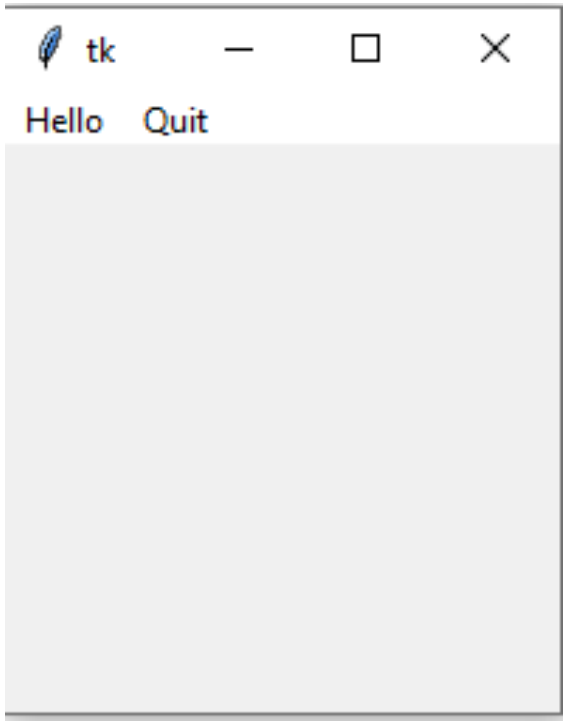
SN	Option	Description
1	add_command(options)	It is used to add the Menu items to the menu.
2	add_radiobutton(options)	This method adds the radiobutton to the menu.
3	add_checkbutton(options)	This method is used to add the checkbuttons to the menu.
4	add_cascade(options)	It is used to create a hierarchical menu to the parent menu by associating the given menu to the parent menu.
5	add_seperator()	It is used to add the separator line to the menu.

Cntd…

- Example:

```
from tkinter import *
top = Tk()
def hello():
    print("hello!")
menubar = Menu(top)
menubar.add_command(label="Hello", command=hello)
menubar.add_command(label="Quit", command=top.quit)
top.config(menu=menubar)
top.mainloop()
```

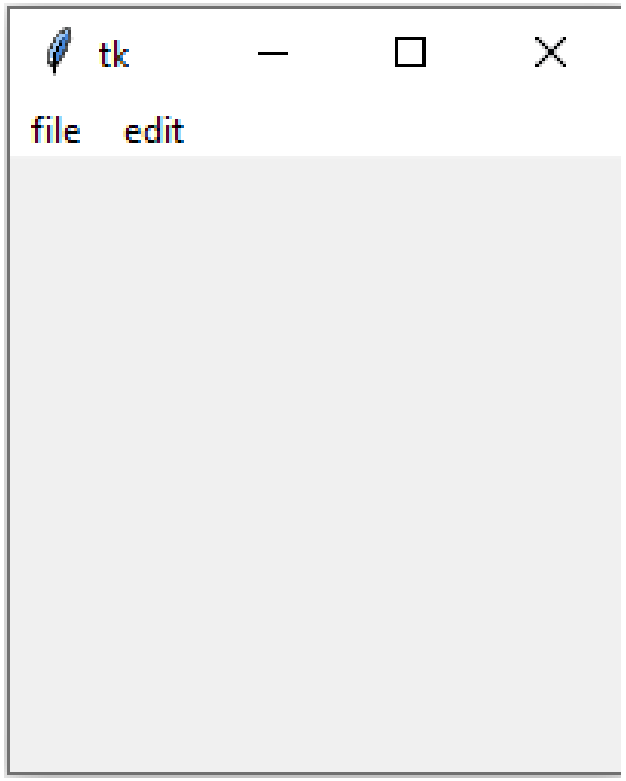
Output:



- Example: Basic Notepad

```
from tkinter import *
top = Tk()
menubar = Menu(top)
#File menu
file_menu = Menu(menubar, tearoff=0)
menubar.add_cascade(label="file",menu=file_menu)
file_menu.add_command(label="New")
file_menu.add_command(label="Open")
file_menu.add_command(label="Save")
file_menu.add_separator()
file_menu.add_command(label="Exit", command=top.quit)
#Edit menu
edit_menu=Menu(menubar,tearoff=0)
menubar.add_cascade(label="edit",menu=edit_menu)
edit_menu.add_command(label="Undo")
edit_menu.add_command(label="Redo")
edit_menu.add_command(label="Copy")
edit_menu.add_command(label="Paste")
top.config(menu=menubar)
top.mainloop()
```

Output:



Message

- 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:** w = Message(parent, options)
- *Some of the options:*

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. The default is CENTER.

2) bitmap: It is used to display the graphics on the widget. It can be set to any graphical or image object.

3) bd: It represents the size of the border in the pixel. The default size is 2 pixel.

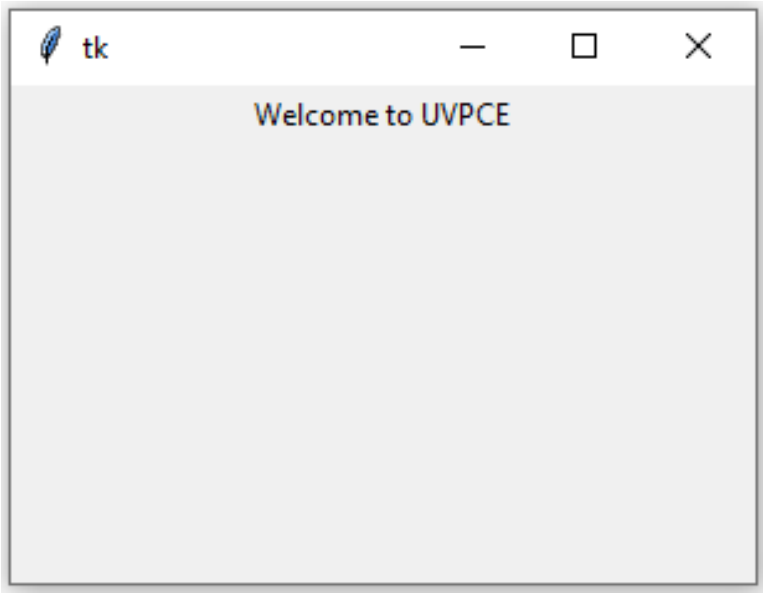
4) cursor: The mouse pointer is changed to the specified cursor type. The cursor type can be an arrow, dot, etc.

Cntd…

- Example-1:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
var = StringVar()
msg = Message( top, text = "Welcome to UVPCE",width=200)
msg.pack()
top.mainloop()
```

Output:



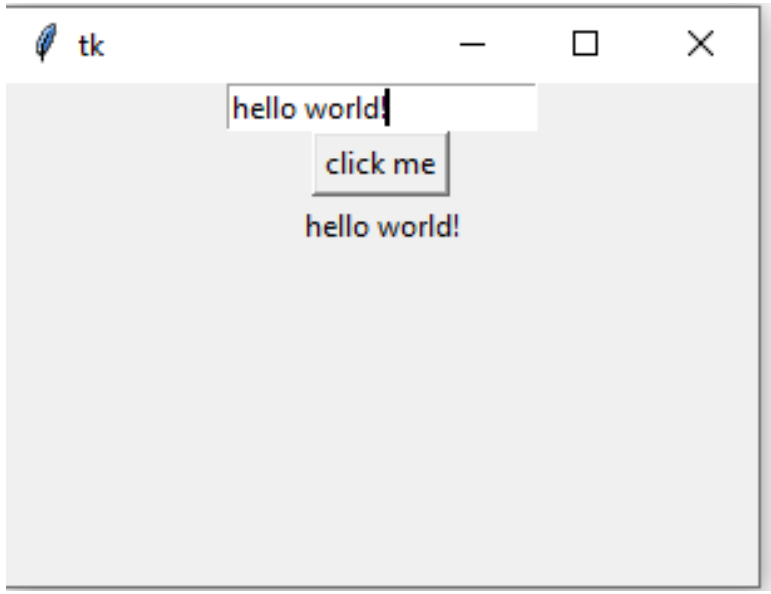
Note: The Message widget is a variant of the Label, designed to display multiline messages. The message widget can wrap text, and adjust its width to maintain a given aspect ratio.

Cntd…

- Example-2: Display textbox contents in label.

```
from tkinter import *
top = Tk()
top.geometry("300x200")
t= Entry(top)
t.pack()
def display():
    msg.config(text=t.get(),width=300)
b= Button(top,text="click me",command=display)
b.pack()
msg = Message(top)
msg.pack()
top.mainloop()
```

Output:



RadioButton

- The Radiobutton 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 radiobutton.
- To keep track of user's selection for radiobutton, it is associated with a single variable. Each button displays a single value for that particular variable.
- **Syntax:** w = Radiobutton(top, options)
- *Some of the Options:*

1) 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.

2) bitmap: It is used to display the graphics on the widget. It can be set to any graphical or image object.

3) borderwidth: It represents the size of the border.

4) command: This option is set to the procedure which must be called every-time when the state of the radiobutton is changed.

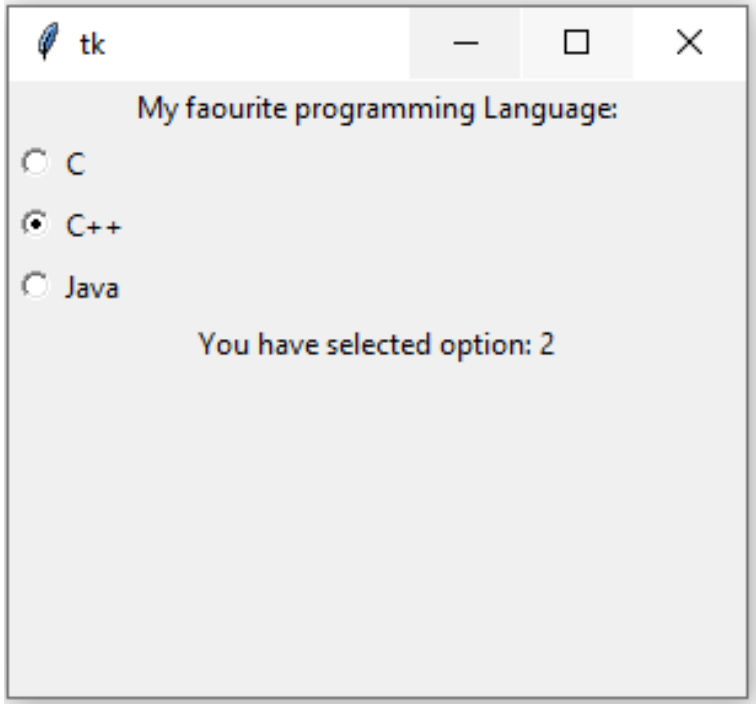
Methods of Radiobutton

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():
    mystring="You have selected option: "+str(radio.get())
    lbl2.config(text = mystring)
top = Tk()
top.geometry("300x250")
radio= IntVar()
lbl=Label(top,text="My faourite programming Language:").pack(anchor=N)
r1= Radiobutton(top,text="C",variable=radio, value=1,command=selection).pack(anchor=W)
r2= Radiobutton(top,text="C++",variable=radio, value=2,command=selection).pack(anchor=W)
r3= Radiobutton(top,text="Java",variable=radio, value=3,command=selection).pack(anchor=W)
lbl2=Label(top)
lbl2.pack()
top.mainloop()
```

Output:



Scale

- The Scale widget is used to implement the graphical slider so that the user can slide through the range of values shown on slider and select 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.
- **Syntax:** w = Scale(top, options)
- *Some of the options:*

1) cursor: The mouse pointer is changed to the cursor type assigned to this option. It can be an arrow, dot, etc.

2) 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.

3) font: The font type of the widget text.

4) from_: It is used to represent one end of the widget range.

5) to : It represents a float or integer value that specifies the other end of the range represented by the scale.

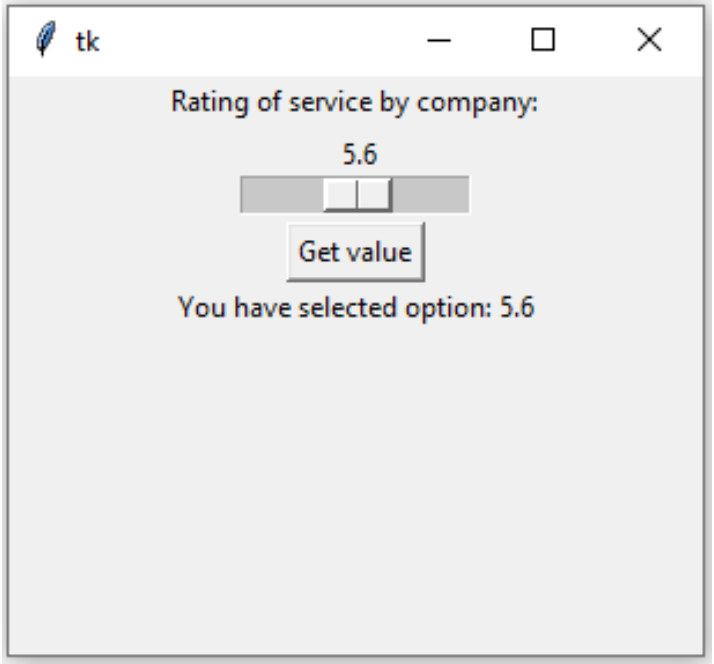
Methods of Scale

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 find_value():
    mystring="You have selected option: "+str(val.get())
    lbl2.config(text = mystring)
top = Tk()
top.geometry("300x250")
radio= IntVar()
lbl=Label(top,text="Rating of service by company:").pack(anchor=N)
val=DoubleVar()
sc= Scale(top,variable=val,from_=1, to= 10, resolution=0.1,orient=HORIZONTAL)
sc.pack()
btn= Button(top,text="Get value",command=find_value)
btn.pack()
lbl2=Label(top)
lbl2.pack()
top.mainloop()
```

Output:



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.
- **Syntax:** w = Scrollbar(top, options)
- *Some of the Options:*

1) elementborderwidth: It represents the border width around the arrow heads and slider. The default value is -1.

2) Highlightbackground: The focus highlighcolor when the widget doesn't have the focus.

3) highlighcolor: The focus highlighcolor when the widget has the focus.

4) highlightthickness: It represents the thickness of the focus highlight.

5) 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.

6) orient: It can be set to HORIZONTAL or VERTICAL depending upon the orientation of the scrollbar.

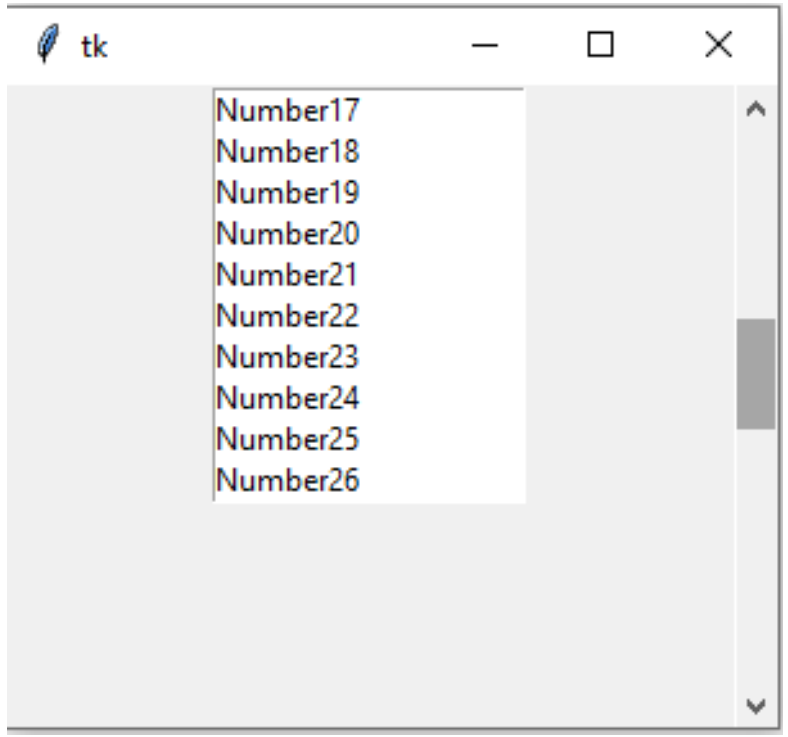
Methods of Scrollbar

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-1:

```
from tkinter import *
top = Tk()
top.geometry("300x250")
sb= Scrollbar(top)
sb.pack(side=RIGHT,fill=Y)
mylist= Listbox(top,yscrollcommand=sb.set)
for i in range(50):
    mylist.insert(END,"Number"+str(i))
mylist.pack()
sb.config( command = mylist.yview )
top.mainloop()
```

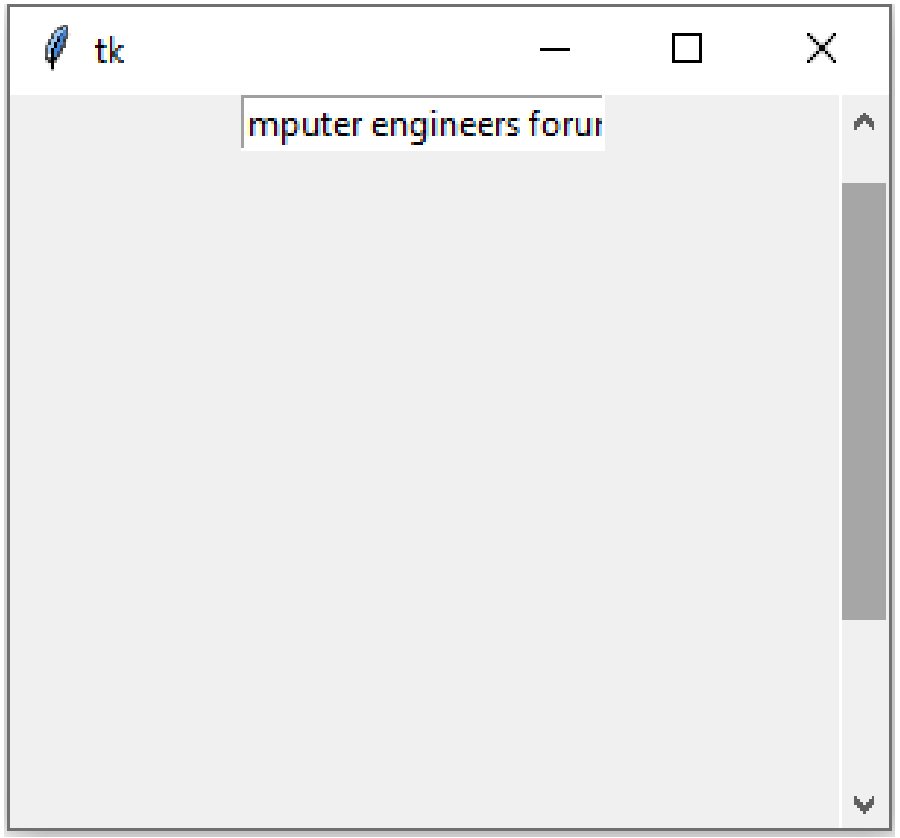
Output:



- Example-2: Using scrollbar for Entry widget

```
from tkinter import *
top = Tk()
top.geometry("300x250")
sb= Scrollbar(top)
sb.pack(side=RIGHT,fill=Y)
en=Entry(top,xscrollcommand=sb.set)
en.pack()
sb.config(command = en.xview )
top.mainloop()
```

Output:



Text

- 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 provides helper structures- **marks and tags** to modify the specific sections of the Text. We can also use the windows and images with the Text as it can also be used to display the formatted text.
- **Syntax:** w = Text(top, options)
- *Some of the Options:*

1) 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.

2) font: The font type of the text.

3) highlightbackground: The highlightcolor when the widget doesn't has the focus.

4) highlightthickness: The thickness of the focus highlight. The default value is 1.

5) highlighcolor: The color of the focus highlight when the widget has the focus.

Methods of Text

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.

Methods for Mark

- Mark is used to bookmark the specified position between two characters within a given text. Mark handling methods are:

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.

Methods for Tags

- The tag is the name given to the specific area of the text. The tags are used to configure the different areas of the text separately. The list of tag-handling methods are:

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.

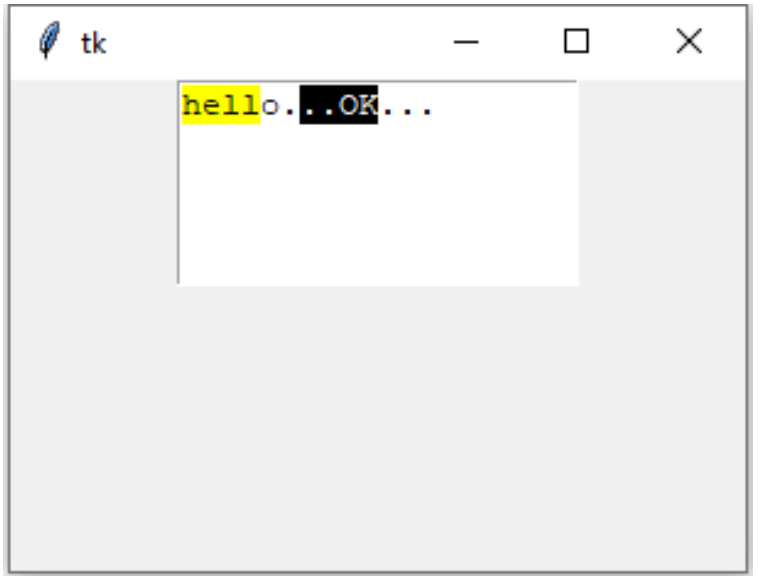
Named indices in Text

- INSERT (or “insert”) corresponds to the insertion cursor.
- CURRENT (or “current”) corresponds to the character closest to the mouse pointer. However, it is only updated if you move the mouse without holding down any buttons (if you do, it will not be updated until you release the button).
- END (or “end”) corresponds to the position just after the last character in the buffer.

- Example:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
t= Text(top,width=20,height=5)
t.insert(END,"hello...")
t.insert(END,"OK...")
t.tag_add("Write Here", "1.0", "1.4")
t.tag_add("Click Here", "1.6", "1.10")
t.tag_config("Write Here", background="yellow", foreground="black")
t.tag_config("Click Here", background="black", foreground="white")
t.mark_set("my_mark","1.1")                #mark is invisible
t.mark_gravity("my_mark",LEFT)             #default value of gravity is RIGHT
t.pack()
top.mainloop()
```

Output:



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 title bars, borders, and other window decorations.
- **Syntax:** w = Toplevel(options)
- *Some of the Options:*

1) cursor: The mouse pointer is changed to the cursor type set to the arrow, dot, etc. when the mouse is in the window.

2) 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.

3) relief: It represents the type of the window.

4) width: It represents the width of the window.

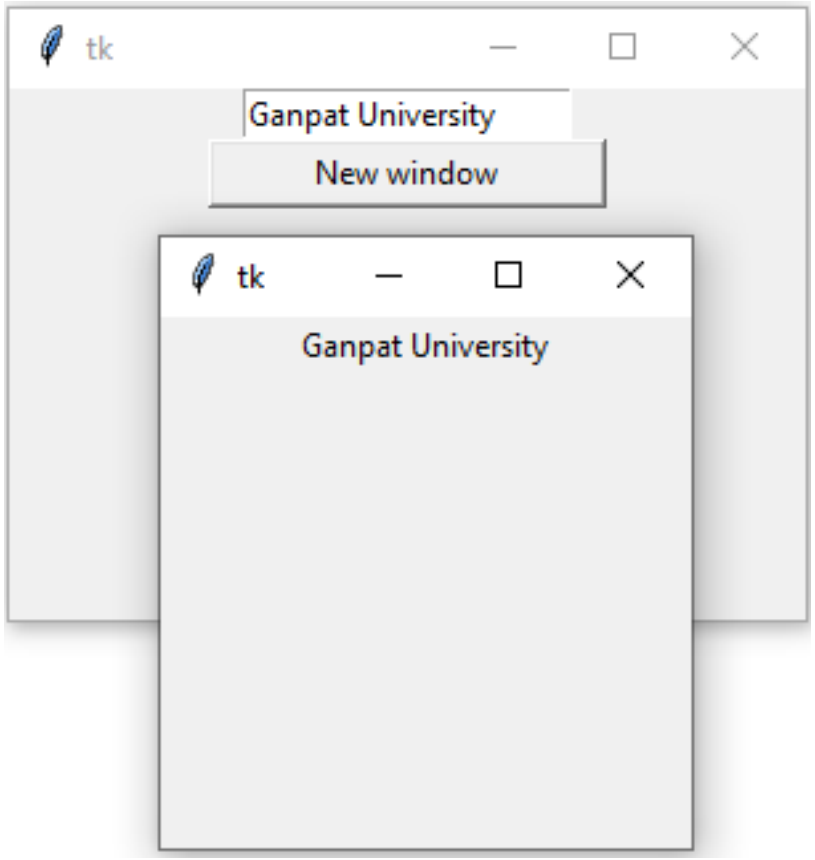
Methods of Toplevel

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: Displaying content of Entry widget into another window

```
from tkinter import *
top = Tk()
top.geometry("300x200")
txt=Entry(top)
txt.pack()
def open_window():
    t=Toplevel(top)
    t.geometry("200x200")
    lbl=Label(t,text=txt.get())
    lbl.pack()
b= Button(top,text="New window",width="20",command=open_window)
b.pack()
top.mainloop()
```

Output:



Spinbox

- The Spinbox widget looks similar to 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.
- **Syntax:** w = Spinbox(top, options)
- *Some of the Options:*

1) activebackground: The background color of the widget when it has the focus.

2) command: The associated callback with the widget which is called each time the state of the widget is called.

3) cursor: The mouse pointer is changed to the cursor type assigned to this option.

4) from_: It is used to show the starting range of the widget.

5) to: It specify the maximum limit of the widget value.

Methods of Spinbox

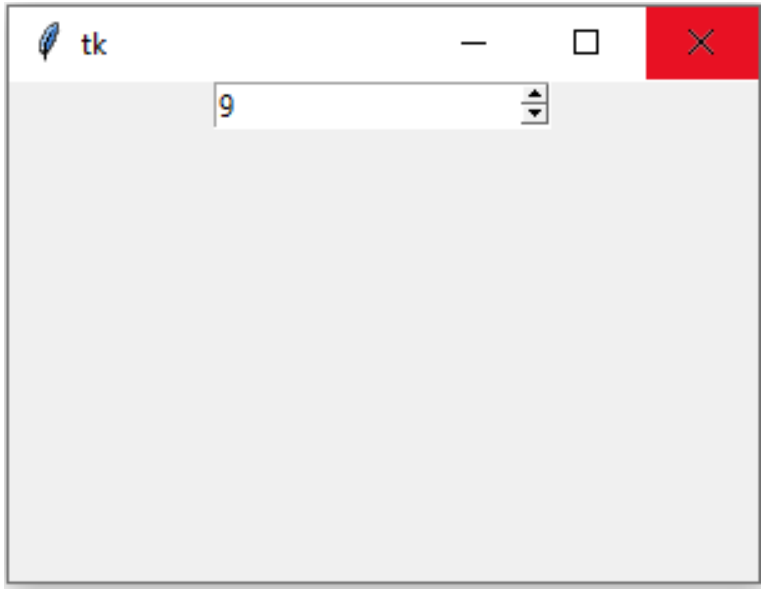
SN	Option	Description
1	delete(startindex, endindex)	This method is used to delete the characters present at the specified range.
2	get(startindex, endindex)	It is used to get the characters present in the specified range.
3	identify(x, y)	It is used to identify the widget's element within the specified 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.

Cntd…

- Example:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
spn=Spinbox(top,from_= 5, to= 25)
spn.pack()
top.mainloop()
```

Output:



PanedWindow

- The PanedWindow widget acts like a Container widget which contains one or more child 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 different layouts in the python applications.
- **Syntax:** w= PanedWindow(master, options)
- *Some of the Options:*

1) cursor: The mouse pointer is changed to specified cursor type when it is over the window.

2) 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.

3) handlesize: It represents the size of the handle. The default size is 8 pixels. However, the handle will always be a square.

4) orient: It will be set to HORIZONTAL if we want to place the child windows side by side. It can be set to VERTICAL if we want to place the child windows from top to bottom.

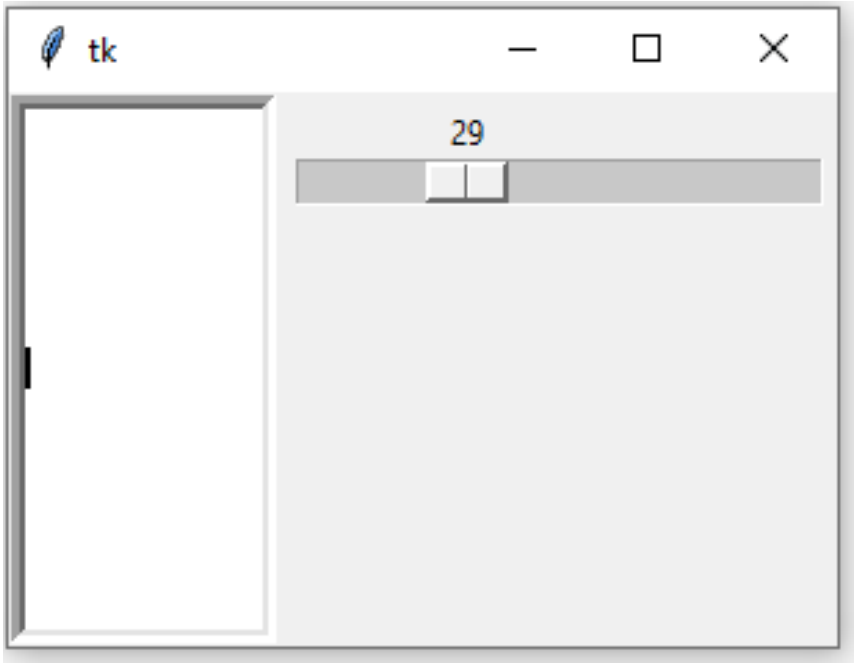
Methods of PanedWindow

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:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
pw= PanedWindow(top)
pw.pack(fill=BOTH,expand=1)
en= Entry(pw,bd=5)
pw.add(en)
pw2= PanedWindow(pw)
pw.add(pw2)
sc= Scale(pw2,orient=HORIZONTAL)
pw2.add(sc)
top.mainloop()
```

Output:



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. Additionally, It also can display a label.
- **Syntax:** w = LabelFrame(top, options)
- *Some of the options:*

1) Class: The default value of the class is LabelFrame.

2) colormap: This option is used to specify which colormap 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.

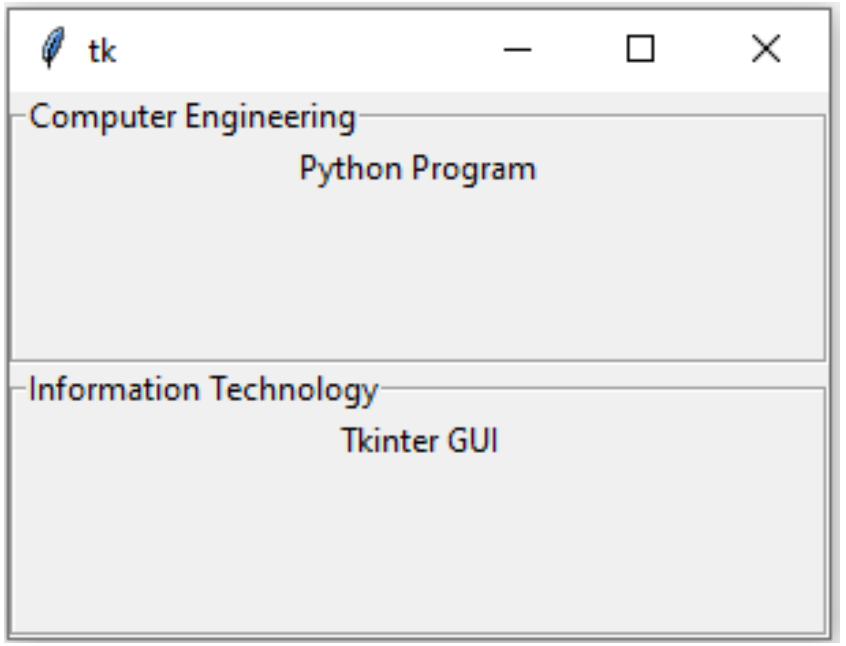
3) container: If this is set to true, the LabelFrame becomes the container widget. The default value is false.

4) 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.

- Example:

```
from tkinter import *
top = Tk()
top.geometry("300x200")
lf1= LabelFrame(text="Computer Engineering")
lf1.pack(fill=BOTH,expand="yes")
lbl=Label(lf1,text="Python Program")
lbl.pack()
lf2=LabelFrame(text="Information Technology")
lf2.pack(fill=BOTH,expand="yes")
lbl2=Label(lf2,text="Tkinter GUI")
lbl2.pack()
top.mainloop()
```

Output:



Messagebox

- The messagebox module is used to display the pop-up dialogbox showing some message or information.
- Message box is a property of windows, it does not work in Linux, hence, it’s a different package in tkinter and needs to called separately.
- There are the various functions which are used to display the relevant messages depending upon the application requirements.
- **Syntax:** messagebox.function_name(title, message [, options])
- *Parameters:*

➢function_name: It represents an appropriate functionality provided by message box.

➢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 in the message box.

Cntd…

- options: There are two options which can be used to configure the message dialog box.

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.

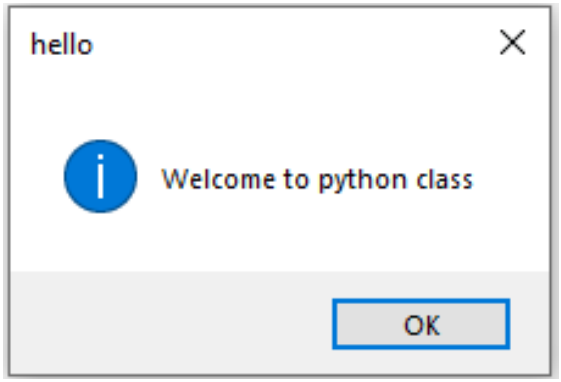
- List of functions used to show the appropriate messageboxes is given below:

1) showinfo()
2) showwarning()
3) showerror()
4) askquestion()
5) askokcancel()
6) askyesno()
7) askretrycancel()

Note: All the functions have same syntax but specific functionalities

1) showinfo(): It is used where we need to show some relevant information to user.

Output:




```
from tkinter import *
from tkinter import messagebox
top = Tk()
messagebox.showinfo("hello","Welcome to python class")
top.mainloop()
```

2) showwarning(): It is used to display the warning to user.

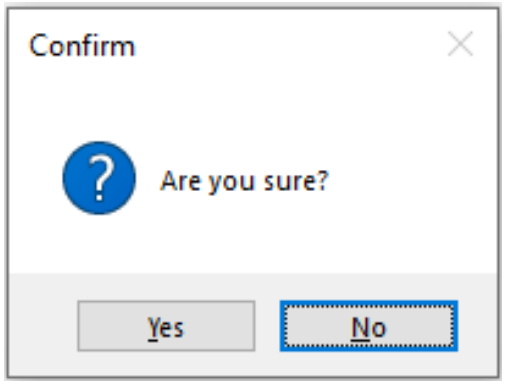
Output:


```
from tkinter import *
from tkinter import messagebox
top = Tk()
messagebox.showwarning("Caution","Battery is low")
top.mainloop()
```

Note: showerror() works in same way

4) askquestion(): It is used to ask some question to the user which can be answered in yes or no.

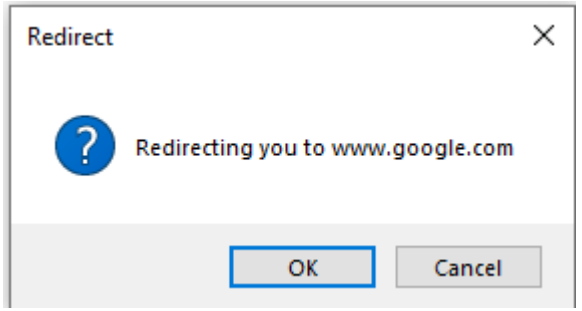
Output:




```
from tkinter import *
from tkinter import messagebox
top = Tk()
messagebox.askquestion("Confirm","Are you sure?",default="no")
top.mainloop()
```

5) askokcancel(): It is used to confirm the user's action regarding some application activity.

Output:

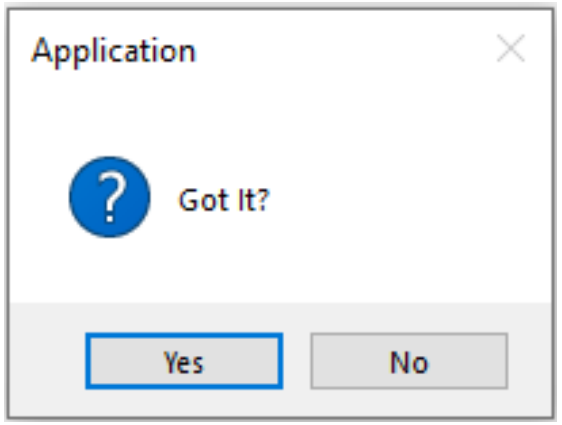



```
from tkinter import *
from tkinter import messagebox
top=Tk()
messagebox.askokcancel("Redirect","Redirecting you to www.google.com")
mainloop()
```

6) askyesno(): It is used to ask the user about some action to which, the user can answer in yes or no.

```
from tkinter import *
from tkinter import messagebox
top = Tk()
messagebox.askyesno("Application","Got It?")
top.mainloop()
```

Output:



7) askretrycancel(): It is used to ask the user about doing a particular task again or not.

```
from tkinter import *
from tkinter import messagebox
top = Tk()
messagebox.askretrycancel("Password Mismatch","Wrong password.\nWanna try again?")
top.mainloop()
```

Output: