Canvas Widget In Python Tkinter | Python Tkinter GUI Tutorial In Hindi #13

The Canvas() widget is a rectangular area for drawing pictures or other complex layouts. This is a highly versatile widget that can draw graphs and plots, create graphics editors, and implement various custom widgets. The canvas is a general-purpose widget, which is typically used to display and edit graphs and other drawings. Another common use for this widget is to implement various kinds of custom widgets. For example, you can use a canvas as a completion bar by drawing and updating a rectangle on the canvas.

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

- **bd:** It defines the border width in pixels. The default value is 2.
- **bg:** Normal background color.
- **height:** It defines the size of the canvas in the Y dimension.
- **width:** It defines the size of the canvas in the X-direction.
- **relief:** It specifies the type of border. Some of the values are SUNKEN, RAISED, GROOVE, and RIDGE.

Here is the list of some standard items which the Canvas widget can support:

- **arc** – It creates an arc item which can be a chord, a pie slice, or a simple arc.

```
coord = 10, 20, 300, 250
arc = canvas.create_arc(coord, start=0, extent=150, fill="red")
```

- **line** – It creates a line item.

```
#line = canvas.create_line(x0, y0, x1, y1, ..., xn, yn, options)
line = canvas.create_line(0, 0, 800, 400, 20, 100, fill="red")
```

- **oval** – It creates a circle or an ellipse at the given coordinates. It takes two pairs of coordinates; the top left and bottom right corners of the bounding rectangle for the oval.

```
#oval = canvas.create_oval(x0, y0, x1, y1, options)
oval = canvas.create_oval(344,233,244,355, fill="blue")
```

- **rectangle:** It creates a rectangle at the given coordinates. It takes two pairs of coordinates; the top left corner and the bottom right corner of the rectangle.

```
#rectangle = canvas.create_rectangle(x0, y0, x1, y1, options)
oval = canvas.create_rectangle(0, 100, 100, 0, fill="blue")
```

Code is described below:

```
from tkinter import *

root = Tk()

canvas_width = 800
canvas_height = 400

root.geometry(f"{canvas_width}x{canvas_height}")
root.title("Harry Ka GUI")
can_widget = Canvas(root, width=canvas_width, height=canvas_height)
can_widget.pack()

# The line goes from the point x1, y1 to x2, y2
can_widget.create_line(0, 0, 800, 400, fill="red")
can_widget.create_line(0, 400, 800, 0, fill="red")

# To create a rectangle specify parameters in this order - coors of top left and coors of bottom right
can_widget.create_rectangle(3, 5, 700, 300, fill="blue")

can_widget.create_text(200, 200, text="python")

can_widget.create_oval(344,233,244,355)


root.mainloop()
```

- Importing *tkinter* is the same as importing any other module in the Python code. Note that the module's name in Python 2.x is '*Tkinter*', and in Python 3.x, it is '*tkinter*'.

```
from tkinter import *
```

- To create the main window, Tkinter offers a method, 'Tk'. To change the name of the window, you can change the className to the desired one.

```
root = Tk()
```

- To set the dimensions of the Tkinter window and to set the position of the main window on the user's desktop, the geometry() function is used. As in example: the canvas_width is defined as 800 pixels and canvas_height is defined as 400 pixels so we can write the function as *geometry(f"{canvas_width}x{canvas_height}")*.

```
canvas_width = 800
```

```
canvas_height = 400

root.geometry(f"{canvas_width}x{canvas_height}")
```

- To set the title of the GUI window, we use the title() function. Here the title is taken as “Harry Ka GUI”.

```
root.title("Harry Ka GUI")
```

- For drawing or writing something in a Canvas, the Canvas() widget is used, and as attributes, the width and height are passed, which are similar to the size of the GUI window.

```
can_widget = Canvas(root, width=canvas_width, height=canvas_height)
can_widget.pack()
```

- To draw a line create_line() is used where we should pass the X and Y coordinates of the line (i.e. if x1=0, y1=0, x2=800, y2=400 we should write *create_line(0, 0, 800, 400))*

```
can_widget.create_line(0, 0, 800, 400, fill="red")
can_widget.create_line(0, 400, 800, 0, fill="red")
```

- To draw a rectangle, create_rectangle is used where we should pass the coordinates of the top left corner and the bottom right corner of the rectangle (i.e., if (x1, y1) are the coordinates of the top-left corner and (x2,y2) are the coordinates of the bottom-right corner and if x1=3, y1=5, x2=700, y2=300 we should write *create_line(3, 5, 700, 300)*).

```
can_widget.create_rectangle(3, 5, 700, 300, fill="blue")
```

- To add any text, create_text is used, and the “text” attribute is passed to write on the canvas (i.e., text=” python”)

```
can_widget.create_text(200, 200, text="python")
```

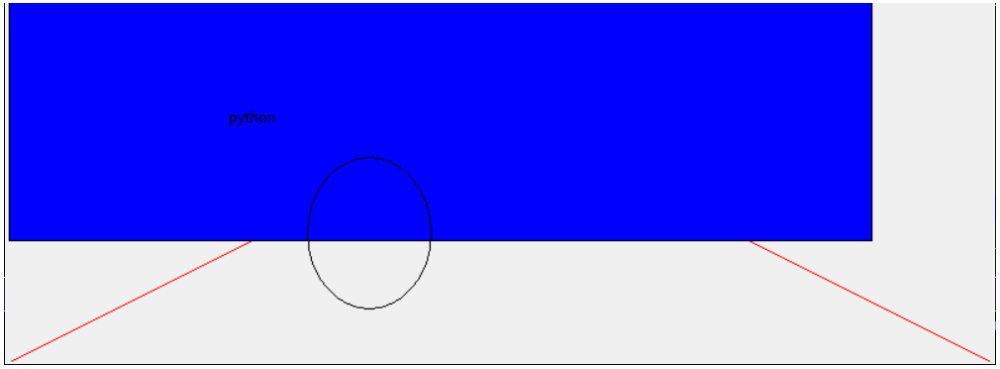
- To draw an oval create_oval is used where we should pass the two pairs of coordinates; the top left and bottom right corners of the bounding rectangle for the oval (i.e., x1=344, y1=233, x2= 244, y2=355 so we can write *create_oval(344, 233, 244, 355)*).

```
can_widget.create_oval(344,233,244,355)
```

- There is a method known by the name *mainloop(),* which is used when your application is ready to run. This is an infinite loop used to run the application, wait for an event to occur, and process the event as long as the window is not closed.

```
root.mainloop()
```

Output: The output of the code (or the GUI window) is given below:



Code as described/written in the video:

Copy

```
from tkinter import *

root = Tk()

canvas_width = 800
canvas_height = 400

root.geometry(f"{canvas_width}x{canvas_height}")
root.title("Harry Ka GUI")
can_widget = Canvas(root, width=canvas_width, height=canvas_height)
can_widget.pack()

# The line goes from the point x1, y1 to x2, y2
can_widget.create_line(0, 0, 800, 400, fill="red")
can_widget.create_line(0, 400, 800, 0, fill="red")

# To create a rectangle specify parameters in this order - coors of top left and coors of bottom right
can_widget.create_rectangle(3, 5, 700, 300, fill="blue")

can_widget.create_text(200, 200, text="python")

can_widget.create_oval(344,233,244,355)


root.mainloop()
```

Previous

Next

CodeWithHarry