Displaying Images Using Label | Python Tkinter GUI Tutorial In Hindi #5

We perhaps add one or more images to make our GUI more creative and presentable. The **PhotoImage()** class displays images in labels, buttons, canvases, and text widgets. You can use the PhotoImage() class whenever you need to display an icon or an image in a Tkinter application. This being said, let's see how to see an image using Python in the shortest way.

Code is described below:

```
from tkinter import *
mahmudul_root = Tk()
mahmudul_root.geometry("1255x944")
```

```
photo = PhotoImage(file="1.png")
varun_label = Label(image=photo)
varun_label.pack()
mahmudul_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.

```
mahmudul_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 the example: the width is 1255 pixels, and the height is 944 pixels, so that we can write the function as ***geometry(1255x944)***.

```
mahmudul_root.geometry("1255x944")
```

- To display an image, i.e., "1.png" in a Tkinter application, the PhotoImage() class is used.

```
photo = PhotoImage(file="1.png")
```

- To call the Label widget, which is a child of the root widget. The keyword parameter "**image**" specifies the

photo "1.png" to be shown:

```
varun_label = Label(image=photo)
```

- The pack method tells Tk to fit the size of the window to the given image.

```
varun_label.pack()
```

- 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.\

```
mahmudul_root.mainloop()
```

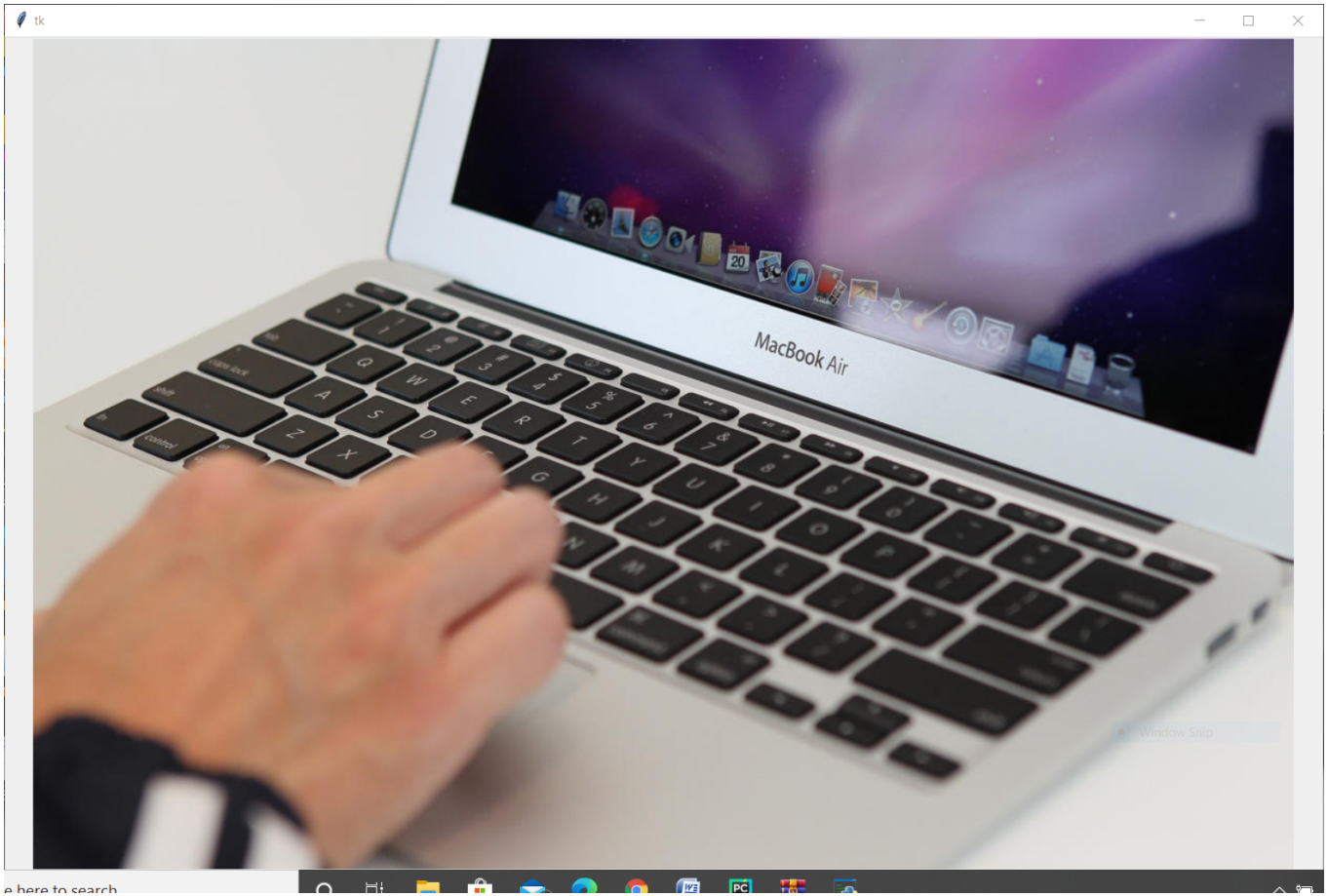
Note: If we need to work with other file formats (example: .jpg), the **Python Imaging Library (PIL)** contains classes that let you load images in over 30 formats and convert them to Tkinter-compatible image objects. So, in this case, we have to install the *pillow* library (if it is not installed in the system) by writing pip install pillow in the terminal and then write the code as follows:

```
from tkinter import *
from PIL import Image, ImageTk
mahmudul_root = Tk()
mahmudul_root.geometry("1255x944")
image = Image.open("1.jpg")
photo = ImageTk.PhotoImage(image)
varun_label = Label(image=photo)
```

```
varun_label.pack()
mahmudul_root.mainloop()
```

- Import **Image** and **ImageTk** from PIL library.
- Take *image* as a variable when the image "**1.jpg**" will be opened.
- Take *photo* as another variable to store the image using the *PhotoImage()* class.
- Call the Label() widget and pack it.
- Write the mainloop() method to run the application.

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



Code as described/written in the video

Copy

```
from tkinter import *
from PIL import Image, ImageTk

mahmudul_root = Tk()

mahmudul_root.geometry("1255x944")
# photo = PhotoImage(file="1.png")

# For Jpg Images

image = Image.open("photo.jpg")
photo = ImageTk.PhotoImage(image)

varun_label = Label(image=photo)
varun_label.pack()



mahmudul_root.mainloop()
```

Previous Next

CodeWithHarry