Day - 39

GUI - Tkinter

Tkinter: → python Library

→ standard python interface to the Tk GUI toolkit.

→ used to create graphical user interfaces (GUIs) for python applications.

→ simple, easy to use.

→ Tkinter allows to create window - based applications.

→ Set of tools to build interfaces, such as buttons, labels, boxes, checkboxes & more.

→ Lightweight.

creating Basic Tkinter window:

import tkinter as tk — e.g. example = tk.Tk() → object name ↓ we give name for object

→ object name. → we can give anyname.

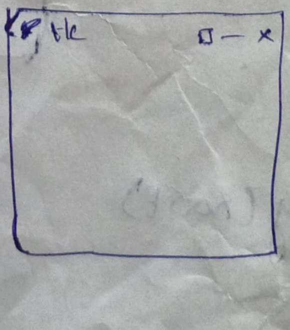
root = tk.Tk() → create main window.

capital T, small k ↳ class in the tkinter module

root.mainloop() → start Tkinter event loop.

↓ ↳ create window. (tk window).

loop call ↳



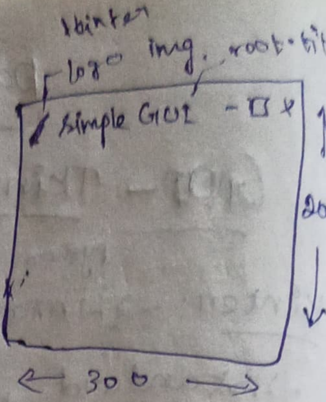
→ tk window.

import tkinter as tk.
root = tk.Tk()

root.title ("simple GUI")

→ width, height
root.geometry (" 300x200")

root.mainloop() → create a window
↳ not giving that, there is no window ~~title~~ created.



↓ root.mainloop()

widgets:

→ Label → Display text or imgs. (Like forms)

→ Button → Clickable btn that can trigger action(li
onc

→ Entry → single line text box for user i/p.
(like i/p element)

→ Text → Multi-line text box for i/p + display.
(like textarea)

Label: → inheritance with parent

create label:

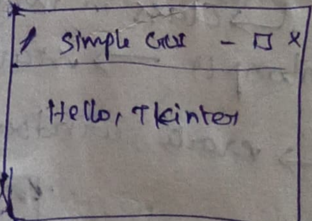
→ variable, → parent)
~~label=tk.Label(root)~~ root = tk.Tk()
~~label.pack()~~

pack → Add label to the window

→ object name, → parent
label=tk.Label(root, text = "Hello, Tkinter", ~~font =("A~~
font ("Arial", 24)

label.pack()

root.mainloop()

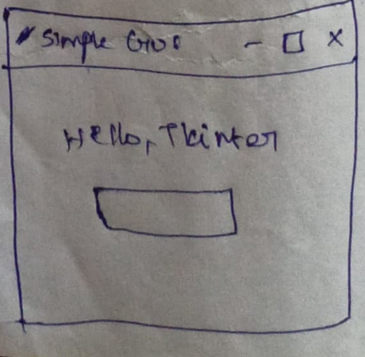


Entry:

→ object name
entry=tk.entry(root)

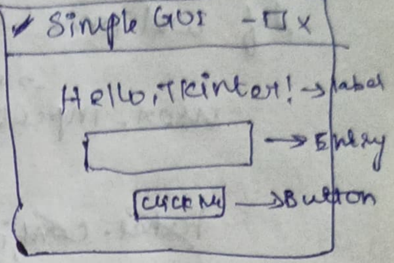
entry.pack()

root.mainloop()



Button

```
button = tk.Button(root, text = "click Me")

button.pack()
```



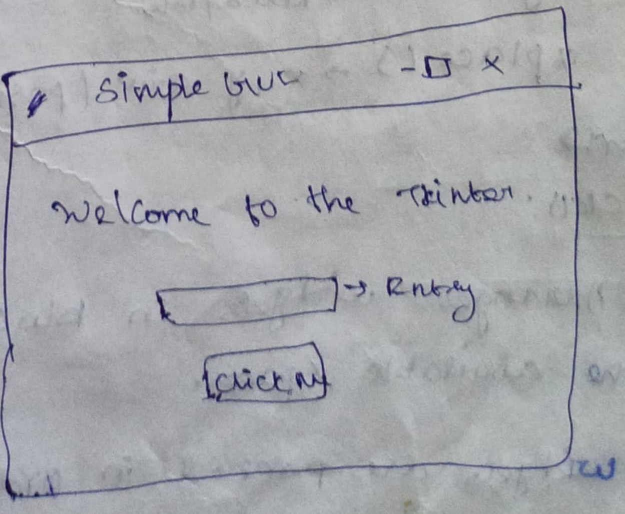
click fn in Button:

```
def welcome ():
    print ("welcome to the Tkinter GUI")

bult on= tk.Button (root, text = "click Me",
                             command = welcome)
button.pack ()
```

↳ O/P: when I click, click me btn in tk window → print ~~welcome~~ welcome fn in terminal.

—— x —— x —— x ——

```
def welome():          ⇒ when I click, btn, change the label text.
    label.config(text = "welome to the Tkinter")
button = tk.Button (root, text = "click Me",
                              command = welcome)

button.pack()
```



don't repeat the label name, btn name, entry x text name again x again so we put name for different.

E.g. label1, label 2, or name, age, ...
same as for btn, entry & text

Change label dynamically using entry:

def welcome():

→variable

user_input = entry.get()   → get string in entry and store it to user_input

label.config(text="welcom

label.config(text=f"Welcome to {user_input}")

button=tk.Button(root, text="Click Me", command=welcom

button. pack() ~~pack(text)~~

→ initially click btn → shows

→ when open window → shows → Hello, Tkinter.

↓

click btn → Welcome to

↓ Give some i/p in entry (user i/p)

again click btn → Welcome to {user i/p}

Basic Layouts in tkinter:-

- * pack() → padding, margin [Left to Right & Top to botto
- * grid() → Rows, Cols.
- * place() → x, y axis (position based)

~~1. pack()~~

1. Pack():

→ Arranges widgets in blocks & places them in the available spaces.

→ widgets are packed in order (top to bottom, left to right)

2. Grid():

→ places widgets in grid sim. similar to
↳ window.
table.

→ like rows, cols.
↳ we can put any name instead of label

E.g. label = tk.Label (root, text = "Name:")

label.grid (row = 0, column = 0) ⇒ Row 0, Col 0

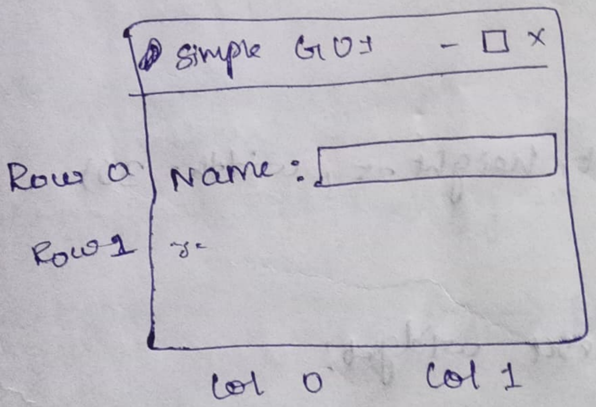
entry = tk.Entry (root)

entry.grid (row = 0, column = 1) → Row 0, Col-1

if want to add age.
↓

age = tk.Label (root, text = 'age')

age.grid (row = 1, column = 0)



3. place():- → using x and y axis.

→ Allows for more precise control over the placement of widgets by specifying coordinates.

label = tk.Label (root, text = "Hello")

label.place (x = 50, y = 50) → like graph.

Working with Frames:- → all widgets places inside fram (container)

→ useful for organizing + grouping widgets. in a window.

→ frame is a container for widgets, allow better org + layout control.

Create frame:

```
frame=tk.Frame(root)
frame.pack()
```

Add widgets inside frame. → add widget inside the fra we can use frame instea root.

```
label=tk.Label(frame, text="This is inside fra
label.pack()
root.mainloop()
```

Text widget:

~~text=tk.roo~~t

```
text=tk.Text(root, height=25, width=30)
text.pack()
```

get content from text widget:

```
def get_content():
                      line-1
                      index position
                                   E
        content=text.get("1.0", tk.End)
        Print(~~f~~" Content:', Content)
```

Align widgets:

For Grid:

sticky => Aligns widgets within grid Cell.

sticky="e" → ~~left~~ right side

sticky="n" → top

sticky="s" → bottom

sticky="w" → Left