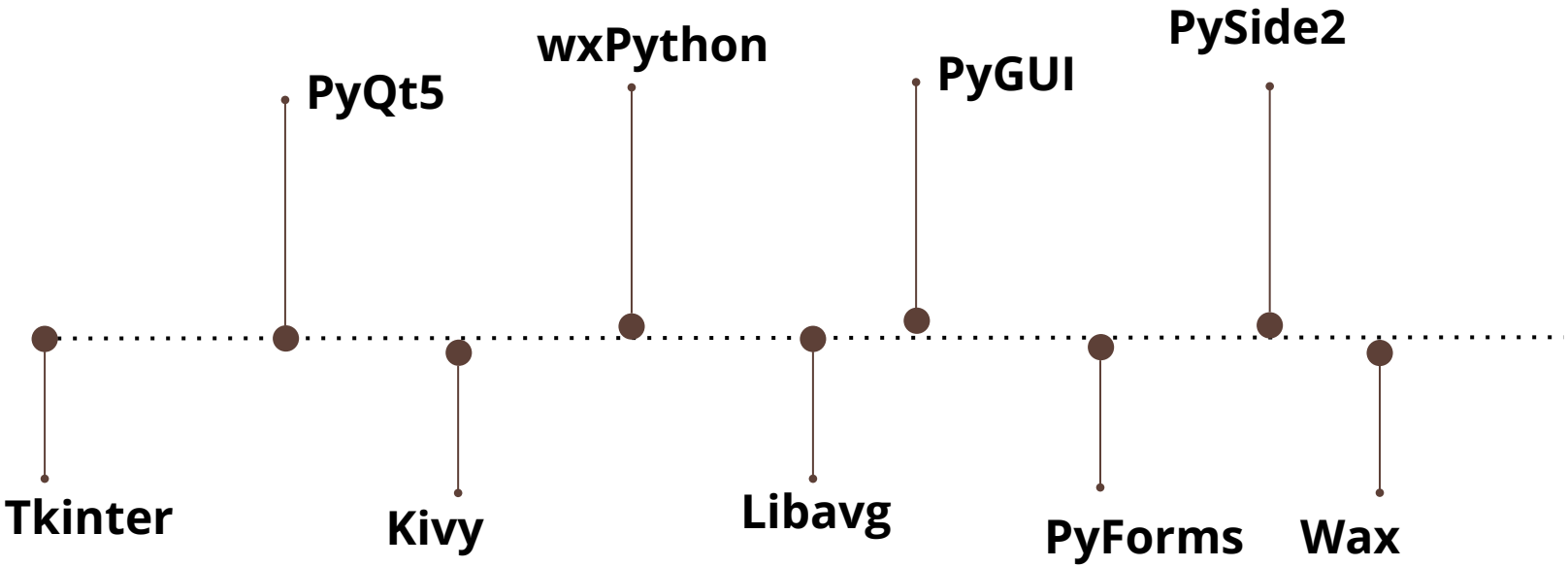
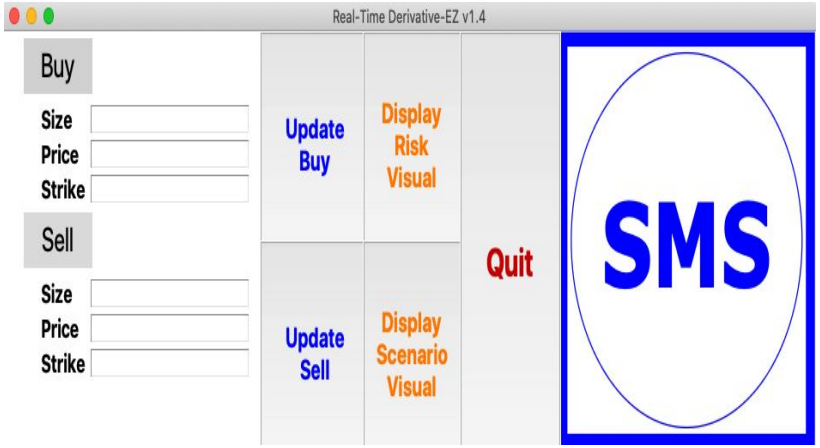
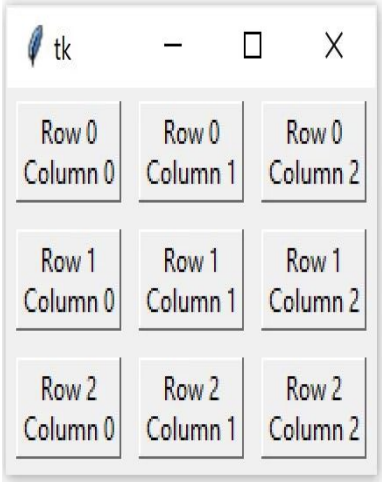
TKINTER

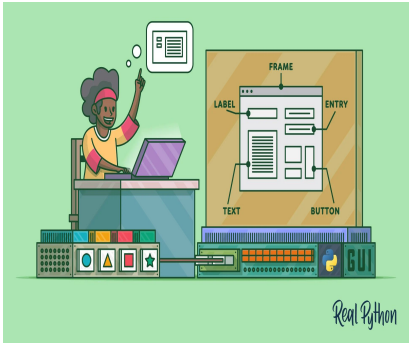
PYTHON - GUI

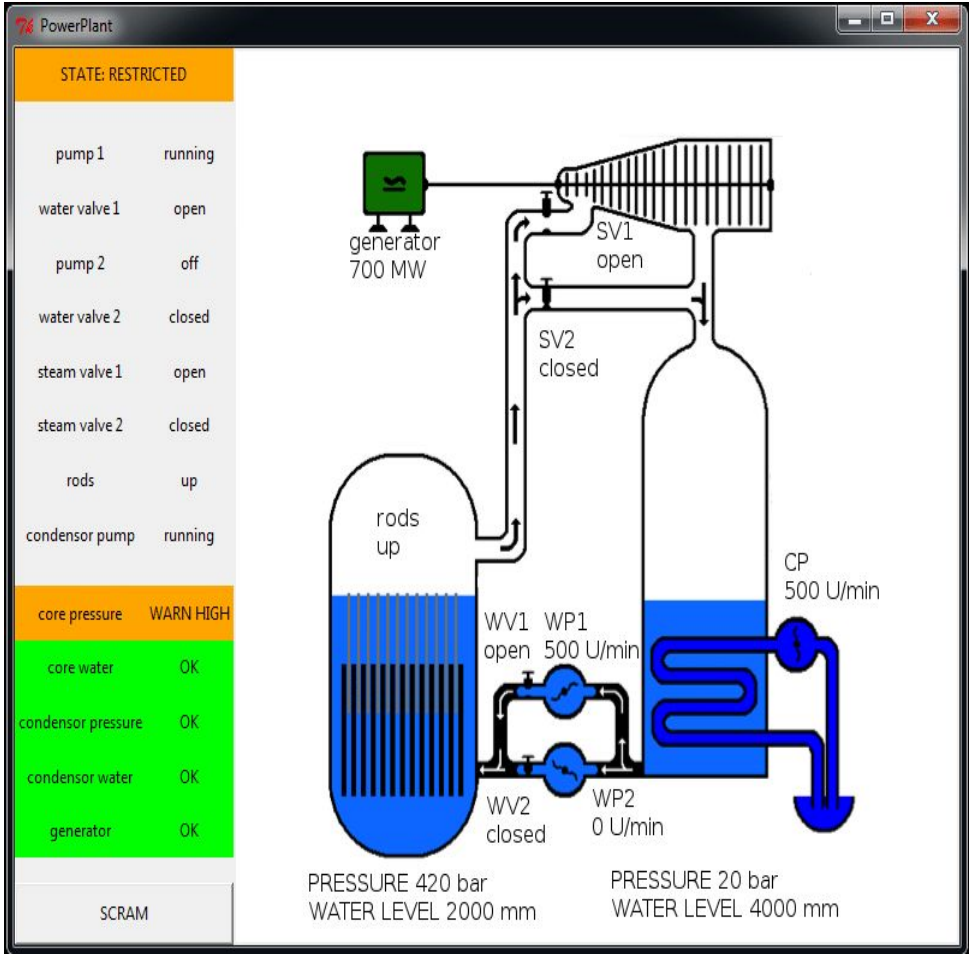
Python GUI Frameworks Example




Real-Time Derivative-EZ v1.4
Buy
Size
Price
Strike
Sell
Size
Price
Strike
Update Buy
Display Risk Visual
Update Sell
Display Scenario Visual
Quit
SMS


tk
Row 0 Column 0
Row 0 Column 1
Row 0 Column 2
Row 1 Column 0
Row 1 Column 1
Row 1 Column 2
Row 2 Column 0
Row 2 Column 1
Row 2 Column 2


FRAME
LABEL
ENTRY
TEXT
BUTTON
GUI
Real Python


PowerPlant
STATE: RESTRICTED
pump 1 running
water valve 1 open
pump 2 off
water valve 2 closed
steam valve 1 open
steam valve 2 closed
rods up
condensor pump running
core pressure WARN HIGH
core water OK
condensor pressure OK
condensor water OK
generator OK
SCRAM
generator 700 MW
SV1 open
SV2 closed
rods up
CP 500 U/min
WV1 open
WP1 500 U/min
WV2 closed
WP2 0 U/min
PRESSURE 420 bar
WATER LEVEL 2000 mm
PRESSURE 20 bar
WATER LEVEL 4000 mm

Tkinter

Python GUI

Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library. Tkinter has several strengths. It's cross-platform, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run.

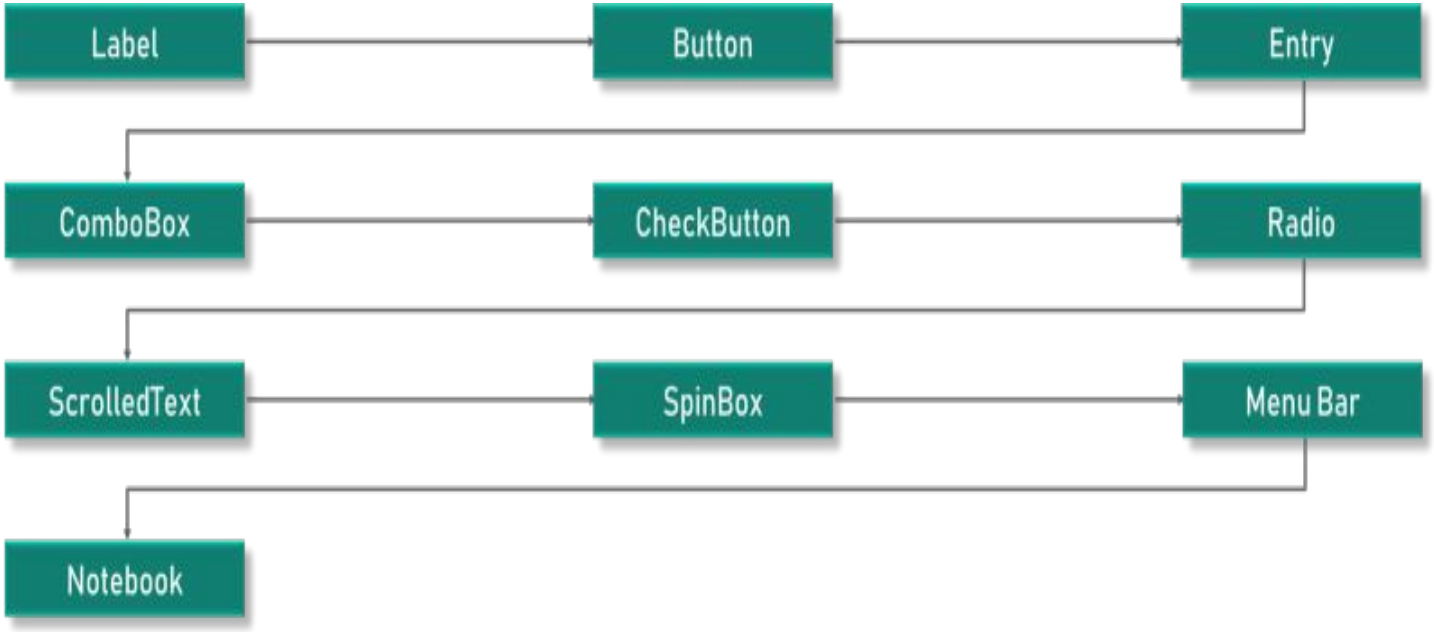
Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

TO CREATE TKINTER APP

1. Import Tkinter Module
2. Create Main Window
3. Add Widgets
4. Enter in event mainloop

Tkinter Widgets

So Widgets are basically like html elements.



Basic Example

```
import tkinter

m = tkinter.Tk()

#widgets are added here

m.mainloop()
```

Tkinter methods

1. **Tk(screenName=None, baseName=None, className='Tk', useTk=1):** To create a main window

   m=tkinter.Tk() where m is the name of the main window object

2. **mainloop():** There is a method used when your application is ready to run. mainloop() 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.

   m.mainloop()

Geometry Management

All widgets in tkinter will have some geometry measurements.This help us to organize widgets with parent frames or windows.

Tkinter has three built-in layout managers that use geometric methods to position widgets in an application frame:

- pack() organizes widgets in horizontal and vertical boxes that are limited to left, right, top, bottom positions. Each box is offset and relative to each other.
- place() places widgets in a two dimensional grid using x and y absolute coordinates.
- grid() locates widgets in a two dimensional grid using row and column absolute coordinates.

Button Widget

Button widget is used to add various types of button in python application.For looks and other configuration we have different properties.

SYNTAX:

w=Button(parent,options)

OPTIONS :

1. activebackground : Background color when mouse hover
2. activeforeground : Font color when mouse hover
3. Bd : border width in pixels

4. Bg : background color
5. Command : function call on click
6. Fg : foreground color
7. Font
8. Height
9. Image : Image on button
10. Padx : padding in horizontal direction
11. Pady : padding in vertical direction
12. Relief : Border type : SUNKEN,RAISED,GROOVE,RIDGE
13. State : ACTIVE by default and can be changed to DISABLED
14. Underline : to underline button text
15. Width

Example

Button

```
from tkinter import *
top = Tk()
top. geometry ("200x100")

def fun():
        messagebox.showinfo("Hello", "Red Button clicked")

b1 = Button(top,text = "Red",command = fun,activeforeground = "red" activebackground = "pink"
,pady=10)
b2 = Button(top, text = "Blue",activeforeground = "blue" ,activebackground = "pink" ,pady=10)
b3 = Button(top, text = "Green",activeforeground = "green" ,activebackground = "pink", pady=10)
b4 = Button(top, text = "Yellow" ,activeforeground = "yellow" ,activebackground = "pink" ,pady=10)

b1.pack(side = LEFT)
b2.pack(side = RIGHT)
b3.pack(side = TOP)
b4. pack(side = BOTTOM)
top.mainloop()
```

Pack method

SYNTAX:

w.pack(options)

OPTIONS :

1. Expand : set true to fill space
2. Fill : None by default , X - horizontally , Y - vertically , BOTH
3. Side : BOTTOM,LEFT,RIGHT,TOP(Default)
4. Padx,pady : for padding

```
from tkinter import *
# creating Tk window
pane = Tk()

b1 = Button(pane, text  = “Click me !")
b1.pack(fill=Y, expand = True)

# Button 2
b2 = Button(pane, text = “Click me too")
b2.pack(fill = X, expand = True)

# Execute Tkinter
pane.mainloop()
```

Place method

SYNTAX:

w.place(options)

OPTIONS :

1. Anchor : Compass direction like N,S ,W,E,NS,NE,SW,SE
2. height,width
3. Relheight,relwidth - height width between - 0.0 and 1.0
4. x,y,relx,rely - offsets from parent

```
# creating Tk window
master = Tk()

# setting geometry of tk window
master . geometry ("200x200")

# button widget
b1 = Button(master, text = “Click me !")
b1.place(relx = 1, x =-2, y = 2, anchor = NE)

# button widget
b2 = Button(master, text = “GFG")
b2.place(relx = 0.5, rely = 0.5, anchor = CENTER)

mainloop()
```

Grid method

SYNTAX:

w.grid(options)

OPTIONS :

1. column , columnspan
2. row , rowspan
3. ipadx , ipady : padding inside widget border
4. padx , pady : padding outside widget border
5. sticky : by default CENTER , values are N,S,W,etc.

```
from tkinter import *

# creating main tkinter window/topLevel
master = Tk()

# this will create Label widget
11 = Label(master, text First:")
12 = Label(master, text = “Second:")

# grid method to arrange Labels in respective
# rows and columns as specified
l1.grid(row = 0, column = 0, sticky = W, pady = 2)
l2.grid(row = 1, column = 0, sticky = W, pady = 2)

mainloop()
```