### Popular Python GUI Libraries

Python offers several libraries for creating Graphical User Interfaces (GUIs). Some of the most popular ones are:

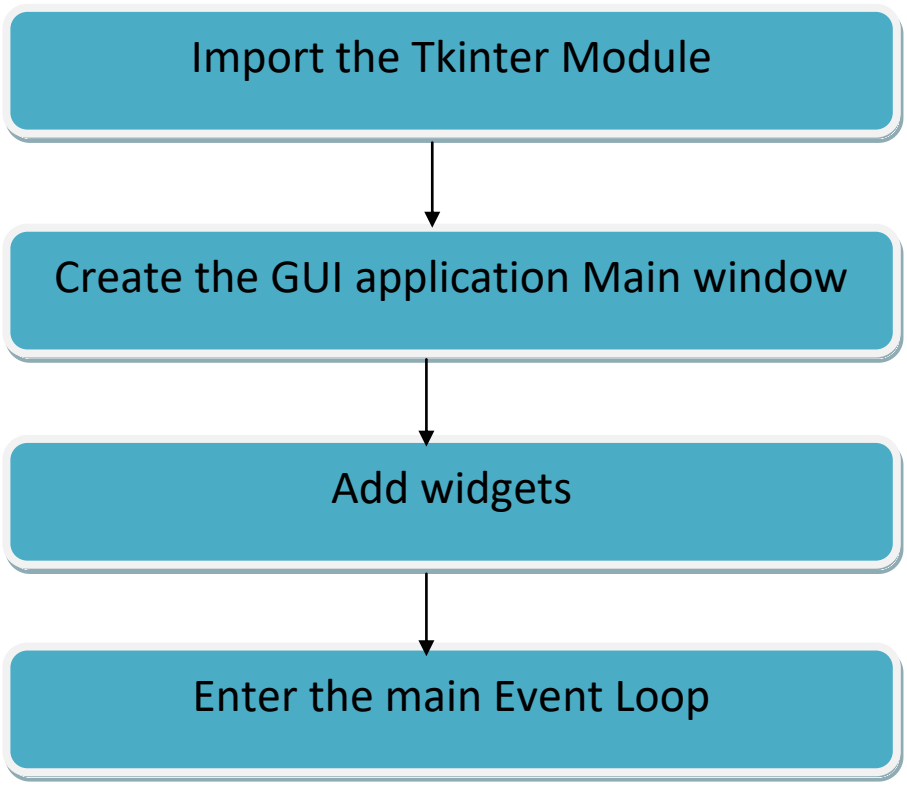
1. **Tkinter** - Built-in library for Python, lightweight and simple to use.
2. **PyQt** - Based on the Qt framework, offers advanced UI elements.
3. **PySide** - Similar to PyQt, also based on Qt but with LGPL licensing.
4. **Kivy** - Used for developing multi-touch applications, supports Android and iOS.
5. **wxPython** - A wrapper around wxWidgets, provides a native look and feel.
6. **PyGTK** - Used for creating GTK-based applications.
7. **PyGUI** - A simple and lightweight cross-platform GUI library.
8. **PyForms** - Designed for rapid development of GUI applications.
9. **Flexx** - Uses Python and JavaScript for web-based applications.
10. **PySimpleGUI** - Simplifies working with other GUI libraries like Tkinter, Qt, and WxPython.
11. **Dear PyGui** - A high-performance, GPU-accelerated GUI library.

# Tkinter

**Tkinter** is the default GUI toolkit that comes with Python. It is built on top of the Tk GUI toolkit and provides an easy way to create simple GUI applications.

### Features of Tkinter

- Comes pre-installed with Python.
- Provides widgets like buttons, labels, text boxes, etc.
- Lightweight and easy to use.
- Cross-platform (works on Windows, macOS, and Linux)



```python
import tkinter as tk


# Create the main window
root = tk.Tk()
root.title("My First Tkinter App")
root.geometry("300x200")  # Set window size

# Add a label
label = tk.Label(root, text="Hello, Tkinter!", font=("Arial", 14))
label.pack(pady=20)  # Add some padding

# Function to change label text
def change_text():
    label.config(text="Button Clicked!")

# Add a button
button = tk.Button(root, text="Click Me", command=change_text)
button.pack()

# Run the Tkinter event loop
root.mainloop()
```

**Output:**


1. `tk.Tk()` creates the main application window.
2. `root.title("Title")` sets the window title.
3. `root.geometry("widthxheight")` sets the window size.
4. `tk.Label()` creates a text label.
5. `tk.Button()` creates a button and links it to a function.
6. `root.mainloop()` keeps the window open.

**Widgets:** A widget is an element of GUI that displays information or provides a specific way for a user to interact with the operating system or application.

# Methods

Each category of Tkinter methods is demonstrated with a working example.

1. ***Window Methods*** *(title, size, background)*
2. ***Widget Configurations*** *(text, colors, fonts)*
3. ***Geometry Management*** *(pack, grid, place)*
4. ***Event Handling*** *(mouse clicks, keypress)*
5. ***Input Handling*** *(entry fields, buttons)*
6. ***Visibility & Control*** *(hiding, showing widgets)*
7. ***Canvas Drawing*** *(lines, rectangles, circles)*
8. ***Menu Creation*** *(dropdown menus)*

### 1. Window Methods :

| Method | Description |
|---|---|
| window.title("Title") | Sets the title of the window. |
| window.geometry("widthxheight") | Sets the window size (e.g., window.geometry("300x200")). |
| window.configure(bg="color") | Changes the background color. |
| window.resizable(width, height) | Enables/disables resizing (True/False). |
| window.attributes('-fullscreen', True) | Opens the window in full-screen mode. |
| window.iconbitmap("icon.ico") | Sets the window icon. |

```
import tkinter as tk

window = tk.Tk()
window.title("Window Methods Example")
window.geometry("400x300")
window.configure(bg="lightblue")
window.resizable(False, False)
window.attributes('-topmost', True)  # Opens Always on top

window.mainloop()
```

### 2. Widget Configuration Methods :

| “Method | Description |
|---|---|
| widget.config(option=value) | Changes a widget's property dynamically. |
| widget.configure(option=value) | Alternative to config(), modifies properties. |

| | |
|---|---|
| widget['text'] = "New Text" | Updates the text of a widget. |
| widget['bg'] = "red" | Changes the background color. |
| widget['fg'] = "white" | Changes the foreground (text) color. |
| widget['font'] = ("Arial", 14, "bold") | Changes the font style. |

```
window = tk.Tk()
window.geometry("300x200")

label = tk.Label(window, text="Original Text", fg="black", bg="white", font=("Arial", 12))
label.pack(pady=10)

def update_label():
    label.config(text="Updated Text!", fg="red", bg="yellow", font=("Arial", 16, "bold"))

button = tk.Button(root, text="Change Label", command=update_label)
button.pack()

window.mainloop()
```

### 3.Geometry Management Methods

| Method | Description |
|---|---|
| widget.pack() | Places widget in the next available space. |
| widget.pack(side="left", padx=10, pady=10) | Aligns widgets (top, bottom, left, right). |
| widget.grid(row=0, column=0) | Uses grid layout for precise positioning. |
| widget.grid(columnspan=2, padx=5, pady=5) | Merges columns or adds spacing. |
| widget.place(x=50, y=100) | Positions widget at an exact location. |

```
windows = tk.Tk()
windows.geometry("300x200")

label1 = tk.Label(windows, text="Pack Example")
label1.pack(pady=5)

label2 = tk.Label(windows, text="Grid Example")
label2.grid(row=0, column=0, padx=10, pady=10)

label3 = tk.Label(windows, text="Place Example")
label3.place(x=100, y=100)

windows.mainloop()
```

### 4.Event Handling Methods

| Method | Description |
|---|---|
| widget.bind("<Event>", function) | Binds an event to a function. |
| widget.bind("<Button-1>", function) | Binds left mouse click. |
| widget.bind("<KeyPress>", function) | Binds a key press event. |

```
import tkinter as tk

root = tk.Tk()
root.geometry("300x200")

def on_click(event):
    label.config(text="Mouse Clicked!")

label = tk.Label(root, text="Click Anywhere", font=("Arial", 14))
label.pack(pady=20)

root.bind("<Button-1>", on_click)  # Binds left mouse click

root.mainloop()
```

**5. Input Handling Methods** (Working with Entry Fields)

| Method | Description |
|---|---|
| entry.get() | Retrieves the text inside an entry box. |
| entry.delete(0, END) | Clears the entry field. |
| entry.insert(0, "Default Text") | Adds text to an entry field. |

```
import tkinter as tk

root = tk.Tk()
root.geometry("300x200")

entry = tk.Entry(root)
entry.pack(pady=10)

def show_text():
    print(entry.get())

button = tk.Button(root, text="Get Text", command=show_text)
button.pack()

root.mainloop()
```

**6.Visibility and Control Methods** (Hiding/Destroying Widgets)

| Method | Description |
|---|---|
| widget.destroy() | Deletes the widget. |
| widget.pack_forget() | Hides the widget (can be restored). |
| widget.grid_forget() | Removes a widget from grid layout. |
| widget.lower() | Moves widget behind others. |
| widget.lift() | Moves widget to the front. |

```
 import tkinter as tk


root = tk.Tk()
root.geometry("300x200")

label = tk.Label(root, text="I will disappear", font=("Arial", 14))
label.pack(pady=10)

def hide_label():
    label.pack_forget()

def show_label():
    label.pack()

button_hide = tk.Button(root, text="Hide", command=hide_label)
button_hide.pack()

button_show = tk.Button(root, text="Show", command=show_label)
button_show.pack()

root.mainloop()
```

7. **Canvas & Drawing Methods** (Creating Shapes)

| Method | Description |
|---|---|
| canvas.create_line(x1, y1, x2, y2, fill="color") | Draws a line. |
| canvas.create_rectangle(x1, y1, x2, y2, fill="color") | Draws a rectangle. |
| canvas.create_oval(x1, y1, x2, y2, fill="color") | Draws an oval/circle. |
| canvas.create_text(x, y, text="Hello") | Adds text inside the canvas. |

```
import tkinter as tk

root = tk.Tk()
root.geometry("300x300")

canvas = tk.Canvas(root, width=300, height=300)
canvas.pack()

canvas.create_line(50, 50, 200, 50, fill="red", width=5)
```

```
canvas.create_rectangle(50, 100, 200, 200, fill="blue")
canvas.create_oval(100, 50, 150, 100, fill="green")

root.mainloop()
```

8. **Menu Methods** (Creating Dropdown Menus)

| Method | Description |
|---|---|
| menu.add_command(label="Item", command=function) | Adds a menu item. |
| menu.add_separator() | Adds a separator line. |

```
import tkinter as tk

root = tk.Tk()
root.geometry("300x200")

menu_bar = tk.Menu(root)

file_menu = tk.Menu(menu_bar, tearoff=0)
file_menu.add_command(label="Open", command=lambda: print("Open Clicked"))
file_menu.add_command(label="Exit", command=root.quit)

menu_bar.add_cascade(label="File", menu=file_menu)

root.config(menu=menu_bar)
root.mainloop()
```