# Day7

# GUI Programming

## Introduction to GUI Programming

Graphical User Interface (GUI) programming allows users to interact with software applications through graphical elements like windows, buttons, and text fields. Python provides several libraries for creating GUIs, with Tkinter being the most commonly used.

### Tkinter Overview

- Tkinter is Python's standard GUI (Graphical User Interface) package.
- It is a thin object-oriented layer on top of the Tk GUI toolkit.
- Tk means toolkit
- Tkinter is included with standard installs of Python, So you need not to use pip install tkinter.

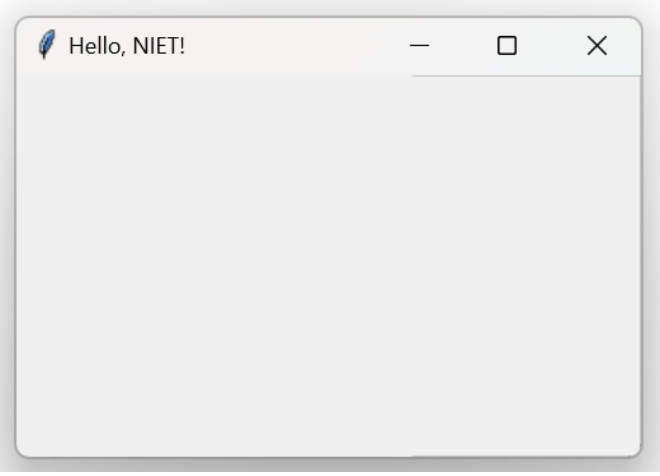
**Widgets:** Label, Button, Entry, RadioButton, text fields etc

### Creating window

Example: Create a window with title “Hello NIET”

```python
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

# Start the GUI event loop
window.mainloop()
```

It will open up a separate window like this:



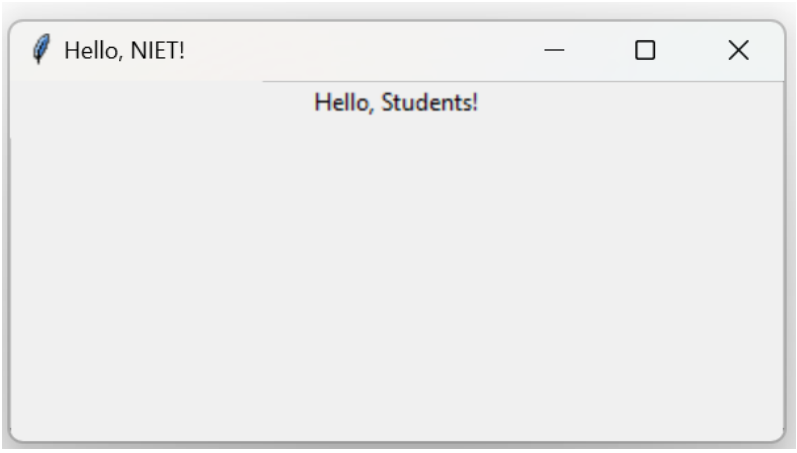

## Setting size of window

```python
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")
window.geometry('400x200')  #to set default widow size, but it can be maximized
window.mainloop()
```


## Setting Widgets in the Window's Interior

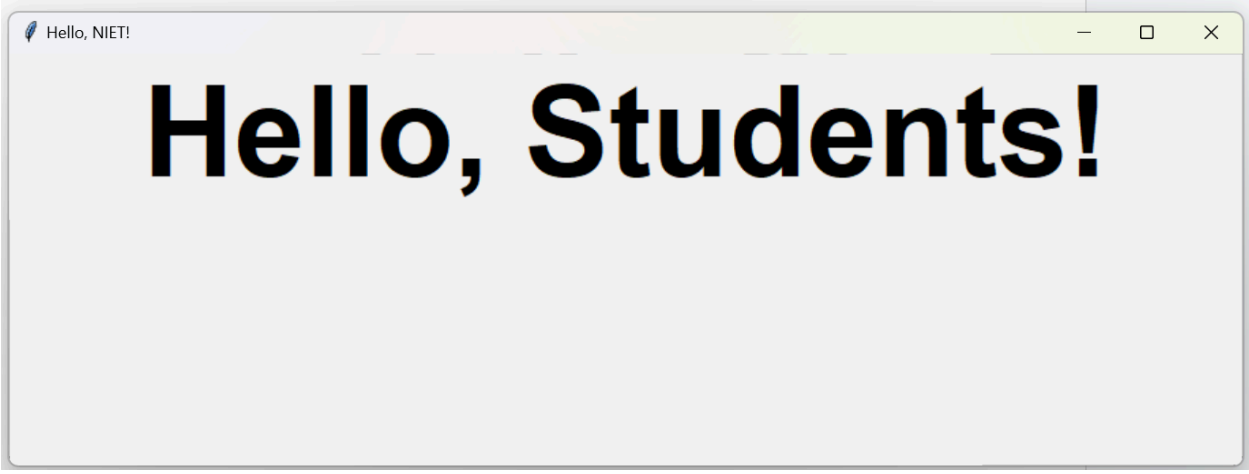
Widgets are the elements in a GUI application, like buttons, labels, and text boxes. To place widgets inside a window, we use geometry managers like pack(), grid(), and place().

[*]:
```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

label = Label(window, text="Hello, Students!")
label.pack()  # Pack the label into the window

window.mainloop()
```




## Change font of the text

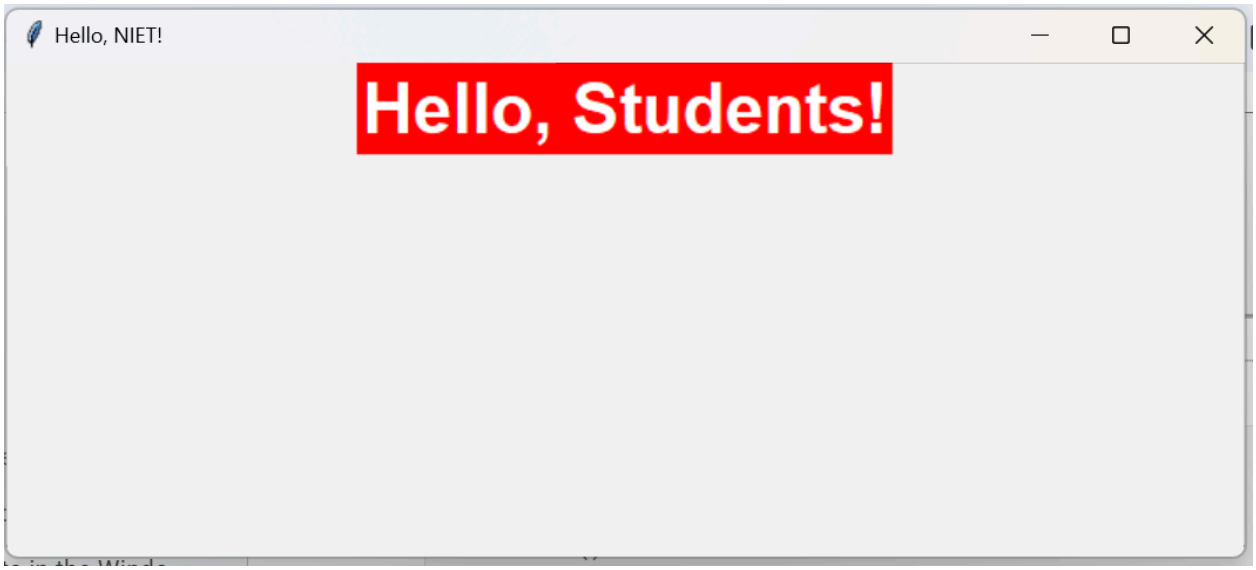
```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

label = Label(window, text="Hello, Students!", font=("Arial Bold", 70))
label.pack()  # Pack the label into the window

window.mainloop()
```




## Change label color

- Use label.config
- fg for foreground color
- bg for background color

```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

label = Label(window, text="Hello, Students!", font=("Arial Bold", 30))
label.config(bg= "red", fg= "white")
label.pack()  # Pack the label into the window

window.mainloop()
```

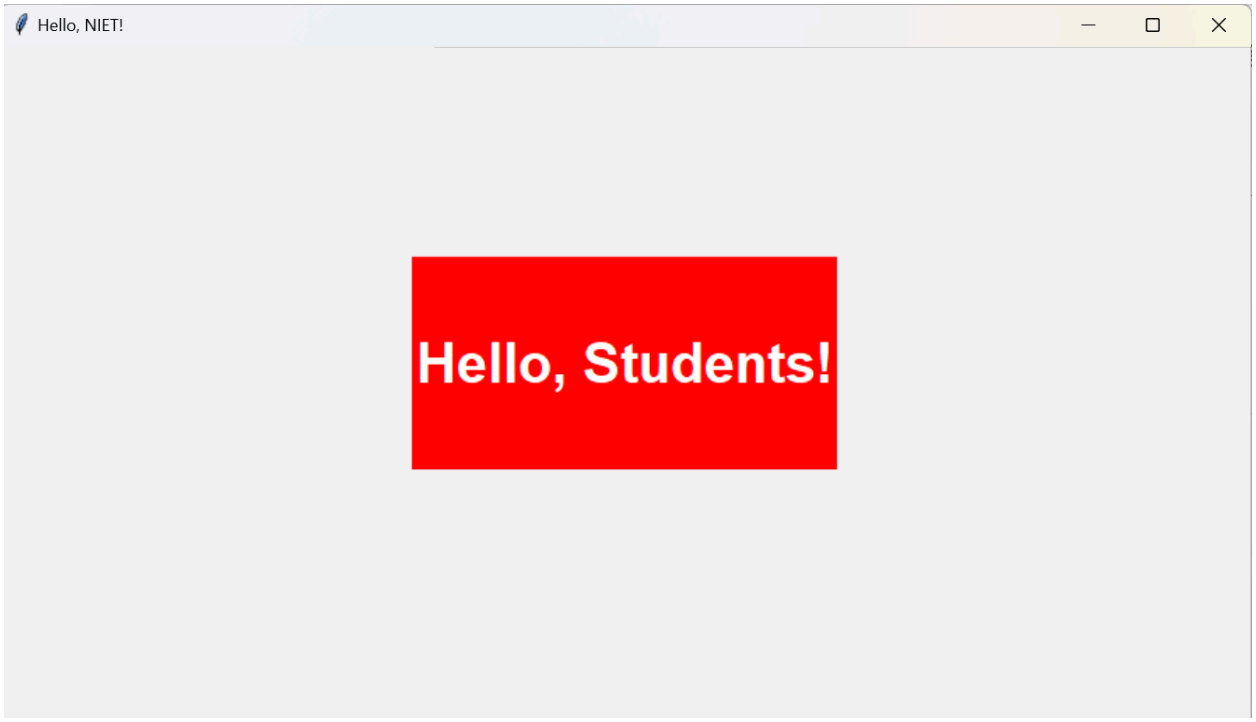



## Pack options

pack() also has padding options:

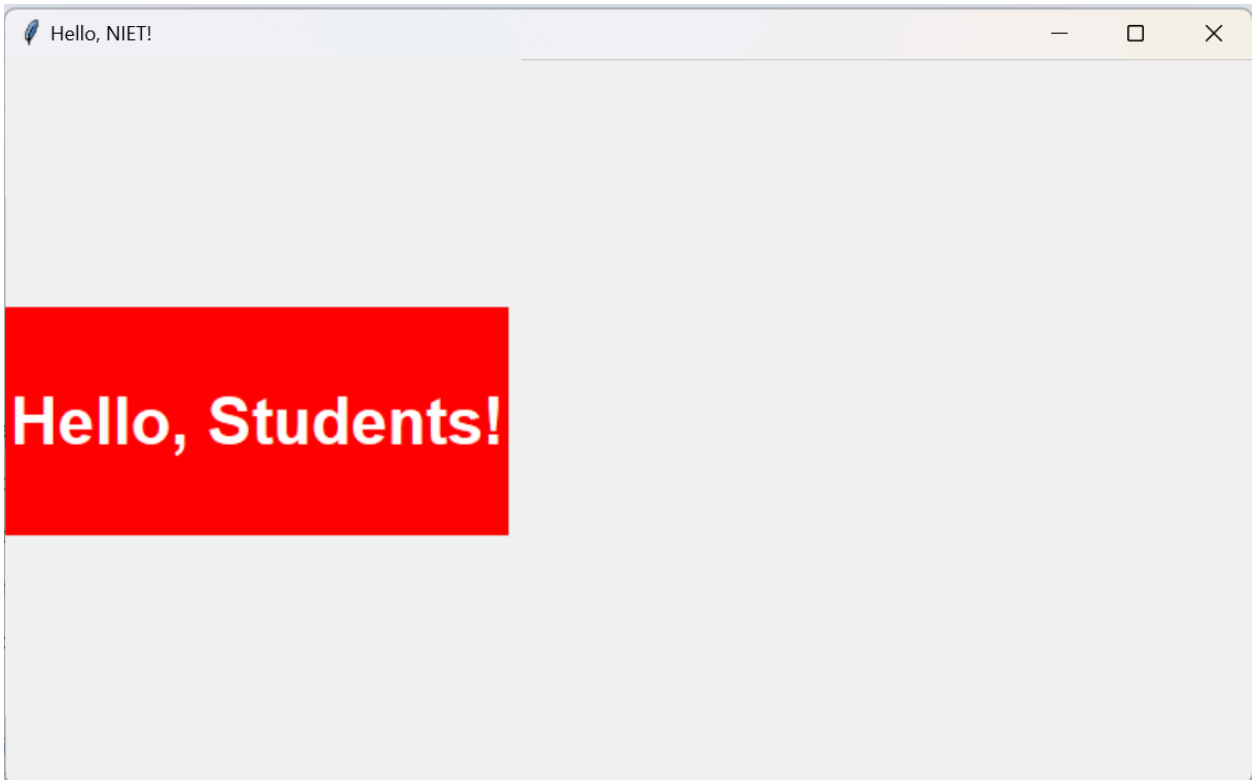
- padx, which pads externally along the x axis.
- pady, which pads externally along the y axis.
- ipadx, which pads internally(within) along the x axis.
- ipady, which pads internally(within) along the y axis.

```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

label = Label(window, text="Hello, Students!", font=("Arial Bold", 30))
label.config(bg= "red", fg= "white")
label.pack(pady=150, ipady=50)  # Pack the label into the window

window.mainloop()
```




## Side attribute in pack()

```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

label = Label(window, text="Hello, Students!", font=("Arial Bold", 30))
label.config(bg= "red", fg= "white")
label.pack(side = "left", pady=150, ipady=50)  # Pack the label into the window

window.mainloop()
```

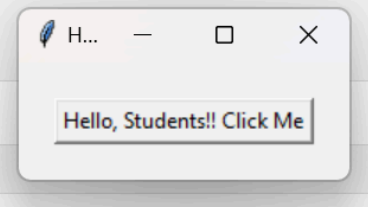



## Button Widget

- Use Button class to create button objects.

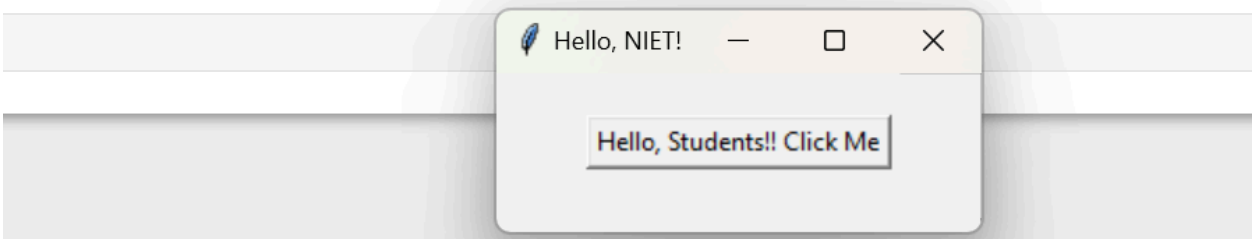
```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

def button_clicked():
    print("Button clicked!")

button = Button(window, text="Hello, Students!! Click Me", command=button_clicked)
button.pack(padx=20, pady=20)

window.mainloop()
```

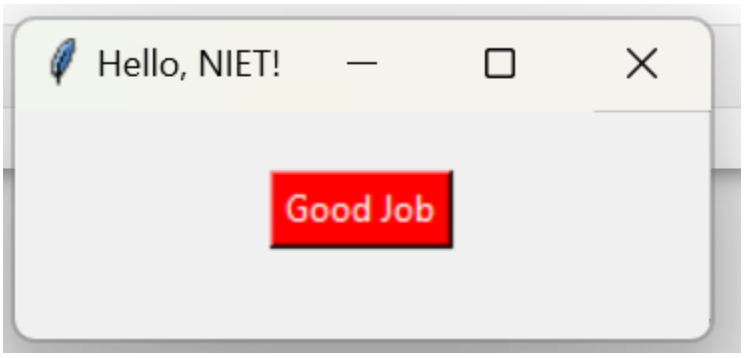
```
Button clicked!
```



## Change Button Text on click

```
from tkinter import *

# Create the main window which is an object of Tk class
window = Tk()
window.title("Hello, NIET!")

def button_clicked():
    button.config(text="Good Job", bg= "red", fg= "white")

button = Button(window, text="Hello, Students!! Click Me", command=button_clicked)
button.pack(padx=20, pady=20)

window.mainloop()
```




## Numeric Widgets

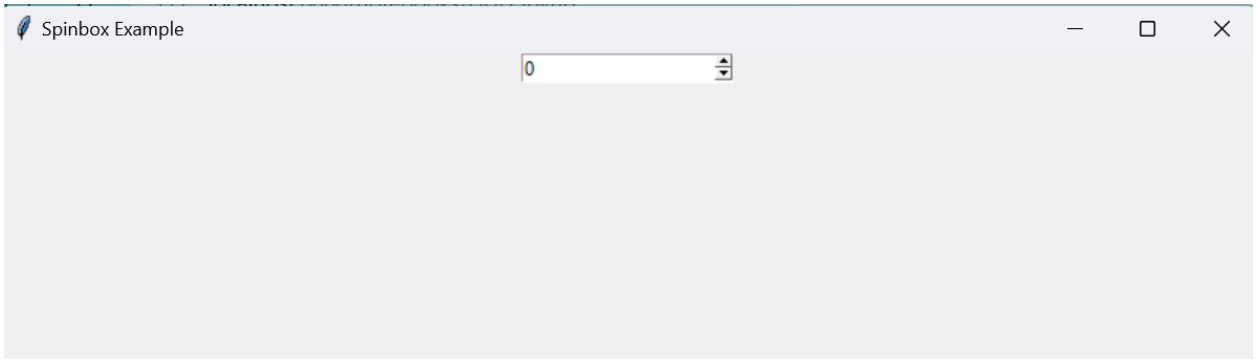
- Numeric widgets allow users to input numbers.
- Common numeric widgets include **Scale** and **Spinbox**.

**Example: Scale Widget**

```
import tkinter as tk

window = tk.Tk()
window.title("Scale Example")
window.geometry('800x200')

# Create a scale widget
scale = tk.Scale(window, from_=0, to=100, orient=tk.HORIZONTAL)
scale.pack()

window.mainloop()
```

**Example: Spinbox Widget**

```python
import tkinter as tk

window = tk.Tk()
window.title("Spinbox Example")
window.geometry('800x200')

# Create a spinbox widget
spinbox = tk.Spinbox(window, from_=0, to=10)
spinbox.pack()

window.mainloop()
```



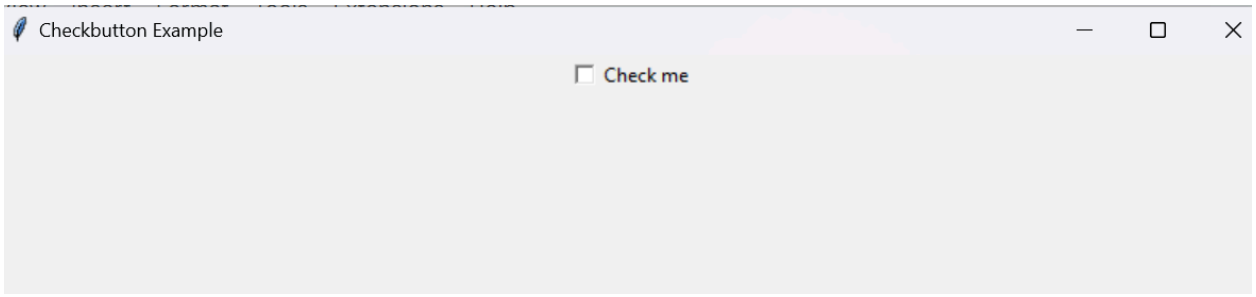

## Boolean Widgets

Boolean widgets represent boolean values (True/False).

The most common boolean widgets are **Checkbutton** and **Radiobutton**.

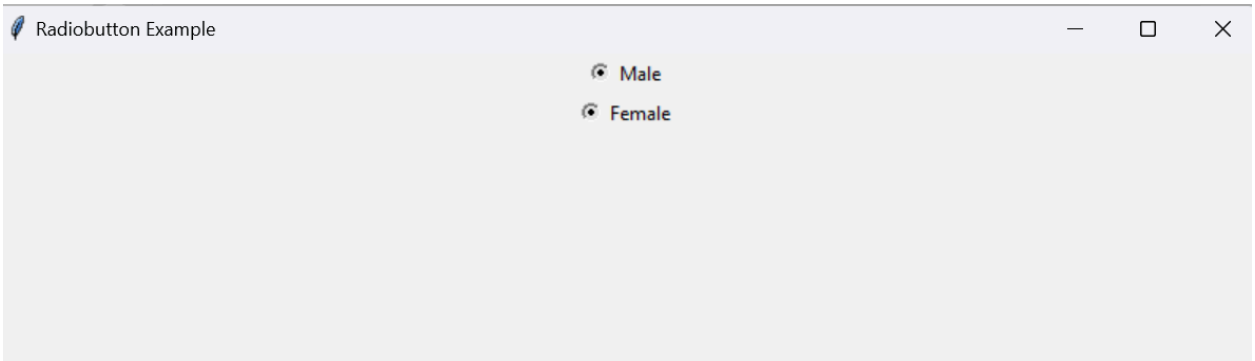
## Example: Checkbutton Widget

```
import tkinter as tk

window = tk.Tk()
window.title("Checkbutton Example")
window.geometry('800x200')

# Create a Checkbutton widget
check_var = tk.BooleanVar()
checkbutton = tk.Checkbutton(window, text="Check me", variable=check_var)
checkbutton.pack()

window.mainloop()
```




## Example: Radiobutton Widget

```
import tkinter as tk

window = tk.Tk()
window.title("Radiobutton Example")
window.geometry('800x200')

# Create Radiobutton widgets
radio_var = tk.StringVar()
radiobutton1 = tk.Radiobutton(window, text="Male", variable=radio_var, value="male")
radiobutton2 = tk.Radiobutton(window, text="Female", variable=radio_var, value="female")

radiobutton1.pack()
radiobutton2.pack()

window.mainloop()
```



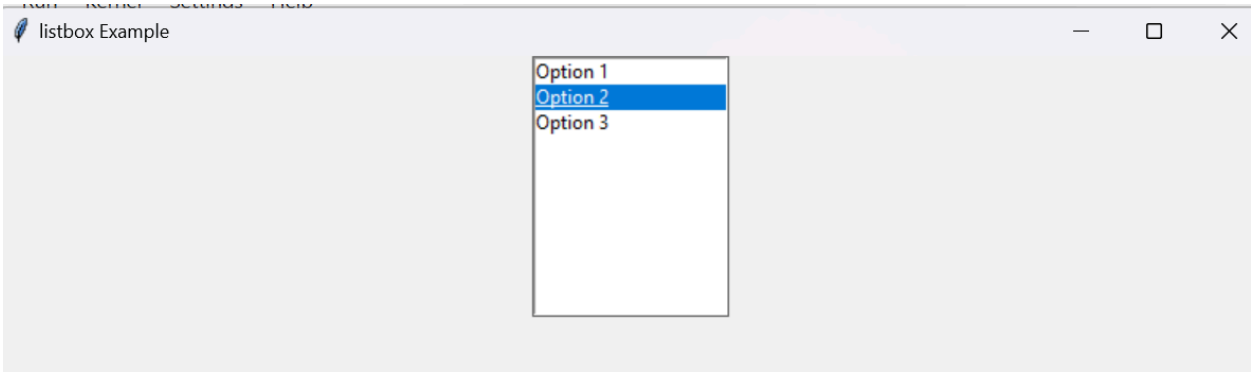

## Selection Widgets

Selection widgets allow users to select from a list of options.

Common selection widgets include **Listbox** and **Combobox**.

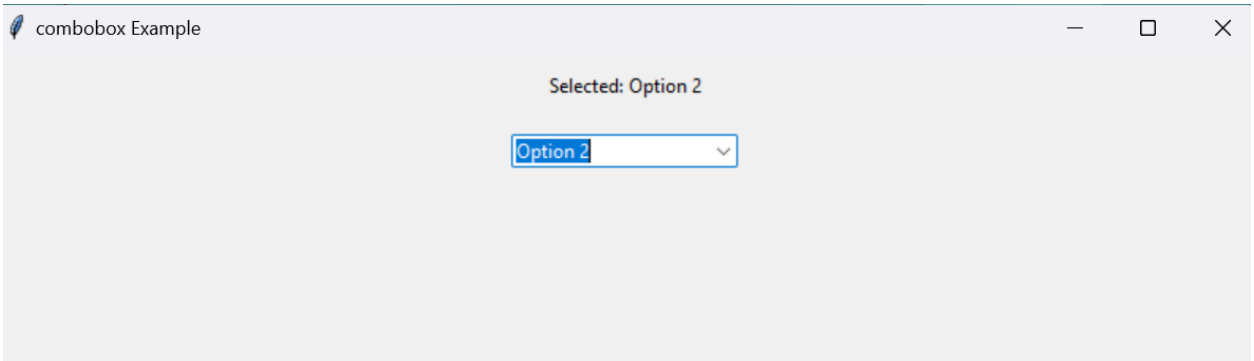
### Example: Listbox Widget

```
import tkinter as tk

window = tk.Tk()
window.title("listbox Example")
window.geometry('800x200')

# Create a Listbox widget
listbox = tk.Listbox(window)
listbox.insert(1, "Option 1")
listbox.insert(2, "Option 2")
listbox.insert(3, "Option 3")
listbox.pack()

window.mainloop()
```




**Example: combobox Widget**

```
import tkinter as tk
from tkinter.ttk import Combobox

window = tk.Tk()
window.title("combobox Example")
window.geometry('800x200')

def combobox_selected(event):
    selected_value = combobox.get()
    label.config(text=f"Selected: {selected_value}")

# Create a label to display the selected value
label = Label(window, text="Select an option:")
label.pack(padx=20, pady=10)

# Create a Combobox widget
options = ["Option 1", "Option 2", "Option 3"]
combobox = Combobox(window, values=options)
combobox.pack(padx=20, pady=10)

# Bind the combobox selection event to the handler
combobox.bind("<<ComboboxSelected>>", combobox_selected)

window.mainloop()
```




## String Widgets

String widgets allow users to input and display text.

Common string widgets include **Entry** and **Label**.

### Example: Entry Widget

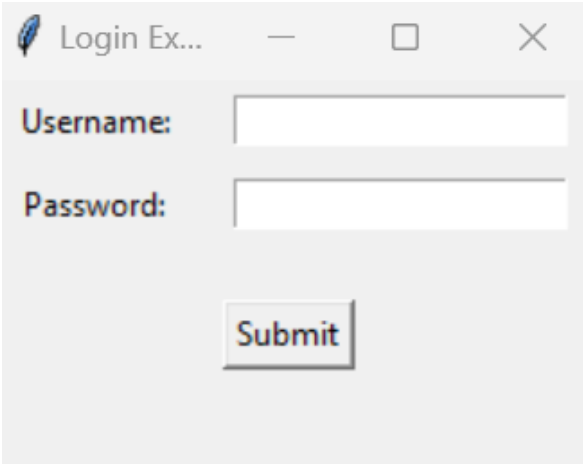
```
import tkinter as tk

window = tk.Tk()
window.title("Entry Example")
window.geometry('800x200')

# Create an Entry widget
entry = tk.Entry(window)
entry.pack()

window.mainloop()
```


## Grid

- grid is used to place widgets.

**Example: Create a login sample page using grid.**

```
from tkinter import *
# Create the main window
window = Tk()
window.title("Login Example")

def submit_clicked():
    result_label.config(text="Good Job")

# Username label and entry
username_label = Label(window, text="Username:")
username_label.grid(row=0, column=0, padx=10, pady=5)

username_entry = Entry(window)
username_entry.grid(row=0, column=1, padx=10, pady=5)

# Password label and entry
password_label = Label(window, text="Password:")
password_label.grid(row=1, column=0, padx=10, pady=5)

password_entry = Entry(window, show="*")
password_entry.grid(row=1, column=1, padx=10, pady=5)

# Submit button
submit_button = Button(window, text="Submit", command=submit_clicked)
submit_button.grid(row=2, column=0, columnspan=2, pady=20)

# Label to show the result
result_label = Label(window, text="")
result_label.grid(row=3, column=0, columnspan=2, pady=5)

window.mainloop()
```



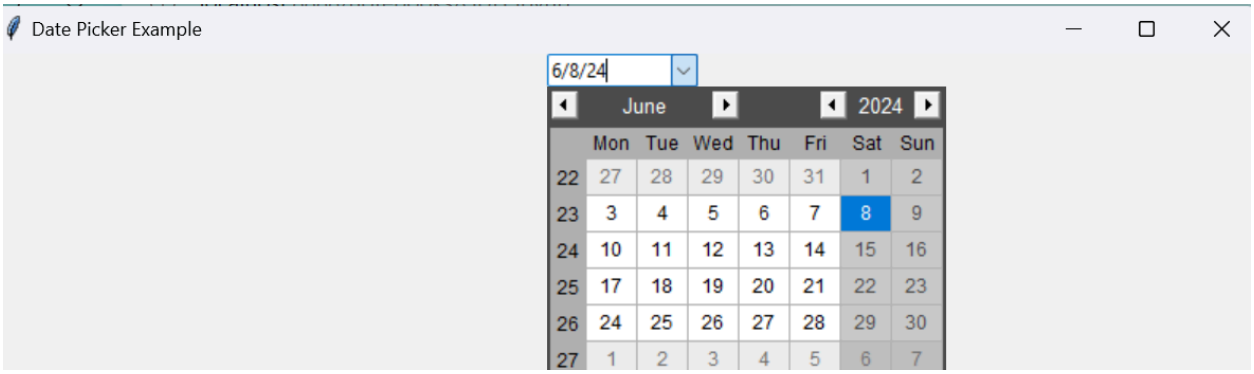

## Date Picker

Tkinter does not have a built-in date picker, but you can use the tkcalendar module to add one.

### Example: Date Picker

First, install the tkcalendar module:

```
from tkinter import *
from tkcalendar import DateEntry

root = Tk()
root.title("Date Picker Example")
root.geometry('800x200')

# Create a DateEntry widget
date_entry = DateEntry(root)
date_entry.pack()

root.mainloop()
```

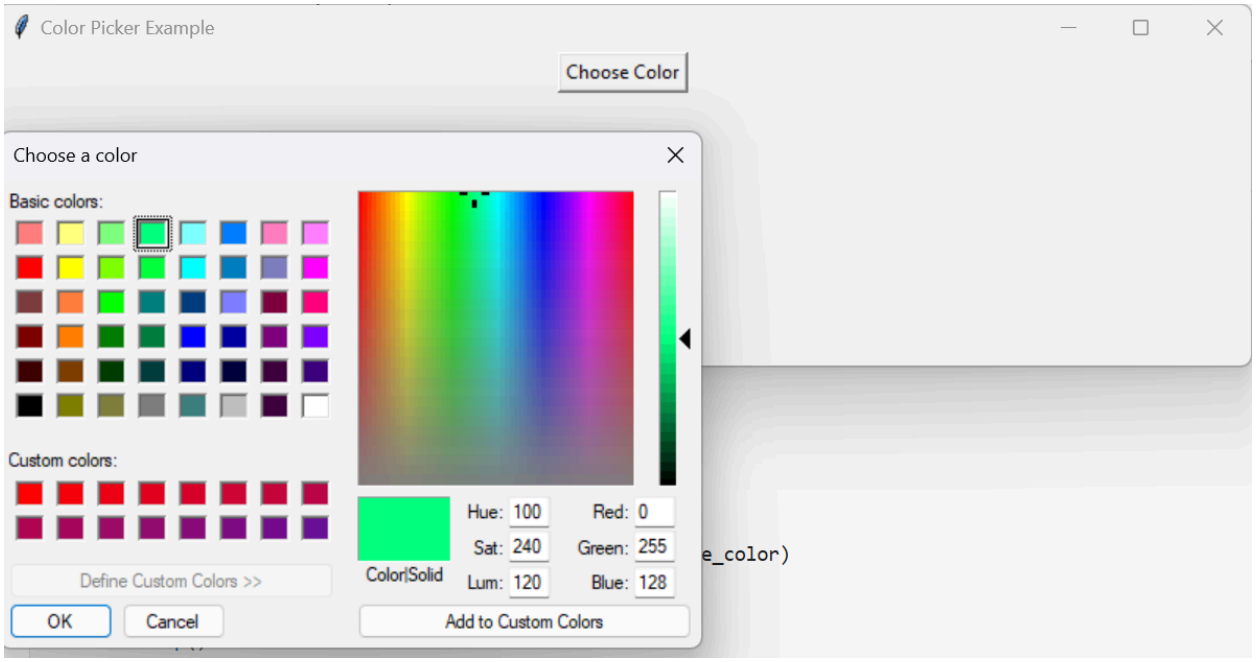



# Color Picker

- Tkinter provides a built-in color picker dialog through the colorchooser module.

### Example: Color Picker

```
import tkinter as tk
from tkinter import colorchooser

def choose_color():
    color_code = colorchooser.askcolor(title="Choose a color")
    print(color_code)

root = tk.Tk()
root.title("Color Picker Example")
root.geometry('800x200')

# Create a button to open the color picker
button = tk.Button(root, text="Choose Color", command=choose_color)
button.pack()

root.mainloop()
```




## Container Widgets

- Container widgets are used to hold other widgets.
- Common container widgets include Frame and PanedWindow.

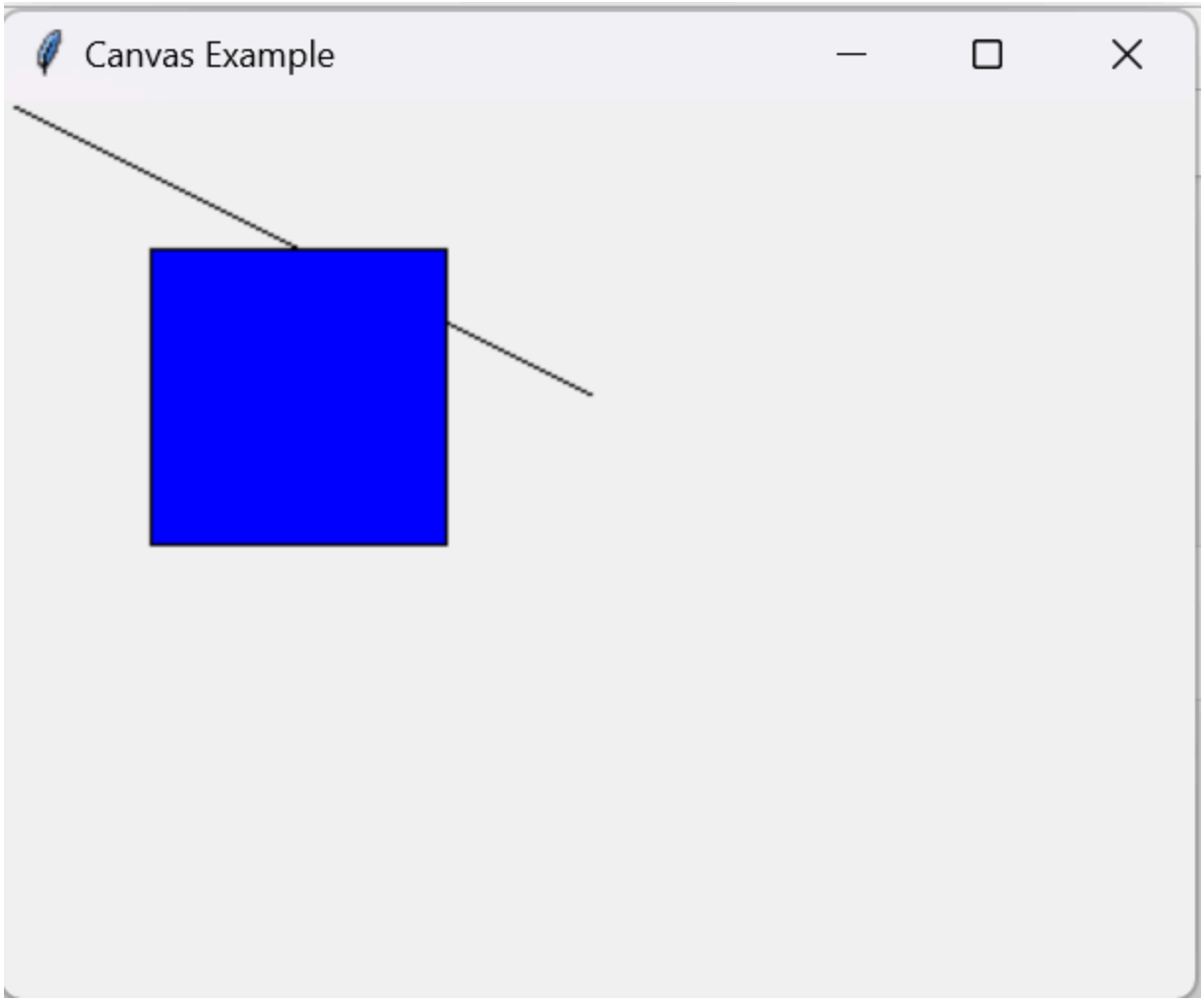
### Example: Frame Widget

```
import tkinter as tk

root = tk.Tk()
root.title("Frame Example")
root.geometry('800x200')

# Create a Frame widget
frame = tk.Frame(root, bg="lightblue", width=200, height=100)
frame.pack()

root.mainloop()
```


## Canvas Widget

The Canvas widget is used for drawing shapes, such as lines, circles, and rectangles.

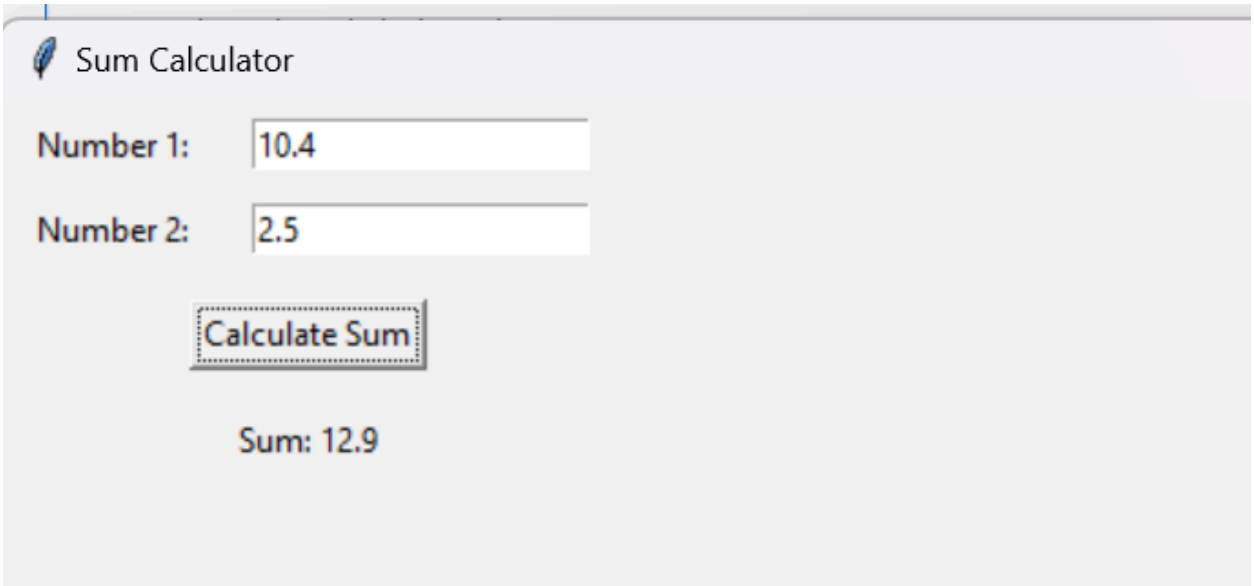
```
import tkinter as tk

root = tk.Tk()
root.title("Canvas Example")

# Create a Canvas widget
canvas = tk.Canvas(root, width=400, height=300)
canvas.pack()

# Draw shapes on the canvas
canvas.create_line(0, 0, 200, 100)
canvas.create_rectangle(50, 50, 150, 150, fill="blue")

root.mainloop()
```


Canvas Example

### Example to add two numbers

```
from tkinter import *

# Create the main window
window = Tk()
window.title("Sum Calculator")
window.geometry('800x200')

def calculate_sum():
    # Get the numbers from the entries
    num1 = float(entry1.get())
    num2 = float(entry2.get())

    # Calculate the sum
    total = num1 + num2

    # Update the result label
    result_label.config(text=f"Sum: {total}")

# First number label and entry
label1 = Label(window, text="Number 1:")
label1.grid(row=0, column=0, padx=10, pady=5)

entry1 = Entry(window)
entry1.grid(row=0, column=1, padx=10, pady=5)
```

```python
# Second number label and entry
label2 = Label(window, text="Number 2:")
label2.grid(row=1, column=0, padx=10, pady=5)

entry2 = Entry(window)
entry2.grid(row=1, column=1, padx=10, pady=5)

# Submit button
submit_button = Button(window, text="Calculate Sum", command=calculate_sum)
submit_button.grid(row=2, column=0, columnspan=2, pady=10)

# Label to show the result
result_label = Label(window, text="Sum: ")
result_label.grid(row=3, column=0, columnspan=2, pady=5)

window.mainloop()
```




## Project: Arithmetic calculator

```
from tkinter import *

# Create the main window
window = Tk()
window.title("Arithmetic Calculator")
window.geometry('800x200')

def calculate():
    # Get the numbers from the entries
    try:
        num1 = float(entry1.get())
        num2 = float(entry2.get())
    except ValueError:
        result_label.config(text="Please enter valid numbers.")
        return

    # Get the selected operation
    operation = operation_var.get()
```

```
    # Perform the selected operation
    if operation == "+":
        result = num1 + num2
    elif operation == "-":
        result = num1 - num2
    elif operation == "*":
        result = num1 * num2
    elif operation == "/":
        if num2 == 0:
            result_label.config(text="Cannot divide by zero.")
            return
        result = num1 / num2
    else:
        result_label.config(text="Please select an operation.")
        return

    # Update the result label
    result_label.config(text=f"Result: {result}")
```

```
# First number label and entry
label1 = Label(window, text="Number 1:")
label1.grid(row=0, column=0, padx=10, pady=5)

entry1 = Entry(window)
entry1.grid(row=0, column=1, padx=10, pady=5)

# Second number label and entry
label2 = Label(window, text="Number 2:")
label2.grid(row=1, column=0, padx=10, pady=5)

entry2 = Entry(window)
entry2.grid(row=1, column=1, padx=10, pady=5)

# Operation label and radio buttons
operation_var = StringVar()
operation_var.set("+")

label3 = Label(window, text="Operation:")
label3.grid(row=2, column=0, padx=10, pady=5)

operations_frame = Frame(window)
operations_frame.grid(row=2, column=1, padx=10, pady=5)

radio_add = Radiobutton(operations_frame, text="+", variable=operation_var, value="+")
radio_add.pack(side=LEFT)

radio_subtract = Radiobutton(operations_frame, text="-", variable=operation_var, value="-")
radio_subtract.pack(side=LEFT)
```

```
radio_multiply = Radiobutton(operations_frame, text="*", variable=operation_var, value="*")
radio_multiply.pack(side=LEFT)

radio_divide = Radiobutton(operations_frame, text="/", variable=operation_var, value="/")
radio_divide.pack(side=LEFT)

# Submit button
submit_button = Button(window, text="Calculate", command=calculate)
submit_button.grid(row=3, column=0, columnspan=2, pady=10)

# Label to show the result
result_label = Label(window, text="Result: ")
result_label.grid(row=4, column=0, columnspan=2, pady=5)

window.mainloop()
```


Arithmetic Calculator
Number 1:
Number 2:
Operation:
+
-
*
/
Calculate
Result: