Waterside Code Club - Tkinter Reference Sheet

Basic Setup

python

```
import tkinter as tk

# Create main window
window = tk.Tk()
window.title("My App")
window.geometry("400x300")  # width x height

# Your widgets go here

window.mainloop()  # Keep window open
```

Common Widgets

Label

Displays text or images

python

```
label = tk.Label(window, text="Hello!")
label.pack()
```

Common Attributes:

- `text` - Text to display
- `font` - Font style: `("Arial", 12)` or `("Arial", 12, "bold")`
- `fg` - Text color (foreground)
- `bg` - Background color
- `width` - Width in characters
- `height` - Height in lines
- `image` - Display an image

- `anchor` - Text alignment: `tk.W` (west/left), `tk.E` (east/right), `tk.CENTER`

Button

Clickable button

python

```
button = tk.Button(window, text="Click Me!", command=my_function)
button.pack()
```

Common Attributes:

- `text` - Button text
- `command` - Function to call when clicked (no parentheses!)
- `font` - Font style
- `fg` - Text color
- `bg` - Background color
- `width` - Width in characters
- `height` - Height in lines
- `state` - `tk.NORMAL`, `tk.DISABLED`, `tk.ACTIVE`

Entry

Single-line text input

python

```
entry = tk.Entry(window)
entry.pack()

# Get the text
text = entry.get()

# Set the text
entry.insert(0, "Default text")

# Clear the text
entry.delete(0, tk.END)
```

Common Attributes:

- `width` - Width in characters
- `font` - Font style
- `fg` - Text color
- `bg` - Background color
- `show` - Character to show instead (e.g., `"*"` for passwords)
- `justify` - Text alignment: `tk.LEFT`, `tk.CENTER`, `tk.RIGHT` *

Text

Multi-line text input

python

```
text = tk.Text(window, width=40, height=10)
text.pack()

# Get all text
content = text.get("1.0", tk.END)

# Insert text
text.insert(tk.END, "Some text")

# Clear text
text.delete("1.0", tk.END)
```

Common Attributes:

- `width` - Width in characters
- `height` - Height in lines
- `font` - Font style
- `fg` - Text color
- `bg` - Background color
- `wrap` - `tk.WORD`, `tk.CHAR`, or `tk.NONE`

Checkbutton

Checkbox for yes/no choices

python

```
var = tk.BooleanVar()
checkbox = tk.Checkbutton(window, text="I agree", variable=var)
checkbox.pack()

# Check if checked
if var.get():
    print("Checked!")
```

Common Attributes:

- `text` - Label text
- `variable` - BooleanVar to store state
- `command` - Function to call when toggled
- `font` - Font style
- `fg` - Text color
- `bg` - Background color
- `onvalue` - Value when checked (default: 1)
- `offvalue` - Value when unchecked (default: 0)

Radiobutton

Multiple choice (select one)

python

```
choice = tk.StringVar(value="Option 1")

radio1 = tk.Radiobutton(window, text="Option 1", variable=choice, value="Option 1")
radio2 = tk.Radiobutton(window, text="Option 2", variable=choice, value="Option 2")
radio1.pack()
radio2.pack()

# Get selected value
selected = choice.get()
```

Common Attributes:

- `text` - Label text
- `variable` - StringVar/IntVar to store selection
- `value` - Value when this button is selected
- `command` - Function to call when selected

- `font` - Font style
- `fg` - Text color
- `bg` - Background color

Listbox

Display a list of items

python

```
listbox = tk.Listbox(window, height=5)
listbox.pack()

# Add items
listbox.insert(tk.END, "Item 1")
listbox.insert(tk.END, "Item 2")

# Get selected item
selection = listbox.get(listbox.curselection())

# Delete item
listbox.delete(0)  # Delete first item
listbox.delete(tk.END)  # Delete last item
```

Common Attributes:

- `width` - Width in characters
- `height` - Number of visible items
- `font` - Font style
- `fg` - Text color
- `bg` - Background color
- `selectmode` - `tk.SINGLE`, `tk.MULTIPLE`, `tk.EXTENDED`

Frame

Container for grouping widgets

python

```
frame = tk.Frame(window, bg="lightgray")
frame.pack()

# Put widgets inside the frame
tk.Label(frame, text="Inside frame").pack()
```

Common Attributes:

- `width` - Width in pixels
- `height` - Height in pixels
- `bg` - Background color
- `borderwidth` - Border thickness
- `relief` - Border style: `tk.FLAT`, `tk.RAISED`, `tk.SUNKEN`, `tk.GROOVE`, `tk.RIDGE`

Layout Managers

pack()

Simple stacking layout

python

```
widget.pack()                      # Stack vertically (default)
widget.pack(side=tk.LEFT)          # Place on left side
widget.pack(side=tk.RIGHT)         # Place on right side
widget.pack(side=tk.TOP)           # Place on top
widget.pack(side=tk.BOTTOM)        # Place on bottom
widget.pack(padx=10, pady=10)      # Add spacing
widget.pack(fill=tk.BOTH)          # Fill available space
widget.pack(expand=True)           # Expand to fill space
```

grid()

Table-like layout (rows and columns)

python

```
widget.grid(row=0, column=0)           # Position in grid
widget.grid(row=0, column=0, padx=5)   # Add horizontal spacing
widget.grid(row=0, column=0, pady=5)   # Add vertical spacing
widget.grid(row=0, column=0, sticky=tk.W)  # Align west (left)
widget.grid(row=0, column=0, columnspan=2) # Span 2 columns
widget.grid(row=0, column=0, rowspan=2)    # Span 2 rows
```

place()

Absolute positioning (x, y coordinates)

python

```
widget.place(x=100, y=50)        # Exact pixel position
widget.place(relx=0.5, rely=0.5) # Relative position (center)
```

Tip: Don't mix pack() and grid() in the same parent container!

Colors

Named Colors

python

```
"red", "blue", "green", "yellow", "orange", "purple", "pink",
"black", "white", "gray", "brown", "cyan", "magenta",
"lightblue", "lightgreen", "darkblue", "darkgreen"
```

Hex Colors

python

```
"#FF0000"  # Red
"#00FF00"  # Green
"#0000FF"  # Blue
"#FFFFFF"  # White
"#000000"  # Black
```

Variables

BooleanVar

For checkboxes (True/False)

python

```
var = tk.BooleanVar()
var.set(True)      # Set value
value = var.get()  # Get value
```

StringVar

For text

python

```
var = tk.StringVar()
var.set("Hello")   # Set value
value = var.get()  # Get value
```

IntVar

For numbers

python

```
var = tk.IntVar()
var.set(42)        # Set value
value = var.get()  # Get value
```

Window Methods

python

```
window.title("My App")            # Set window title
window.geometry("400x300")        # Set window size
window.resizable(False, False)    # Disable resizing
window.config(bg="lightblue")     # Set background color
window.minsize(200, 100)          # Set minimum size
window.maxsize(800, 600)          # Set maximum size
window.destroy()                  # Close window
```

Message Boxes

python

```
from tkinter import messagebox

messagebox.showinfo("Title", "This is info")       # Info message
messagebox.showwarning("Title", "This is warning") # Warning
messagebox.showerror("Title", "This is error")     # Error
result = messagebox.askyesno("Title", "Yes or No?") # Yes/No question
result = messagebox.askokcancel("Title", "OK?")    # OK/Cancel
```

Widget Methods

python

```
widget.config(text="New text")    # Change widget properties
widget.destroy()                  # Remove widget
widget.pack_forget()              # Hide widget (can show again)
widget.grid_forget()              # Hide widget (grid)
```

Common Patterns

Get button to update label

python

```
label = tk.Label(window, text="Original")
label.pack()

def change_text():
    label.config(text="Changed!")

button = tk.Button(window, text="Change", command=change_text)
button.pack()
```

Get input from entry

python

```
entry = tk.Entry(window)
entry.pack()

def submit():
    text = entry.get()
    print(text)

button = tk.Button(window, text="Submit", command=submit)
button.pack()
```

Using variables

python

```
name_var = tk.StringVar()

entry = tk.Entry(window, textvariable=name_var)
entry.pack()

def show():
    print(name_var.get())

button = tk.Button(window, text="Show", command=show)
button.pack()
```

Quick Start Template

python

```
import tkinter as tk

def main():
    # Create window
    window = tk.Tk()
    window.title("Waterside Code Club")
    window.geometry("400x300")

    # Add your widgets here
    label = tk.Label(window, text="Hello, Code Club!", font=("Arial", 16))
    label.pack(pady=20)

    def button_click():
        label.config(text="Button Clicked!")

    button = tk.Button(window, text="Click Me!", command=button_click)
    button.pack()

    # Start the app
    window.mainloop()

if __name__ == "__main__":
    main()
```

Tips for Code Club

1. Always call `window.mainloop()` at the end - this keeps your window open!

2. Functions for buttons - Don't use parentheses:

   - ✅ `command=my_function`
   - ❌ `command=my_function()`

3. Get text from Entry - Use `.get()`:

   python

   ```
   text = entry.get()
   ```

4. Change widget properties - Use `.config()`:

   python

```
label.config(text="New text", fg="red")
```

5. Layout managers - Don't mix `pack()` and `grid()` in the same container!
6. Colors - Use names (`"red"`) or hex codes (`"#FF0000"`)
7. Debugging - Use `print()` statements to see what's happening:

   python

   ```
   def button_click():
      print("Button was clicked!")
      label.config(text="Changed!")
   ```

Project Ideas

- Calculator - Make a working calculator
- To-Do List - Add and remove tasks
- Quiz App - Multiple choice questions
- Drawing App - Use Canvas widget
- Password Generator - Generate random passwords
- Timer/Stopwatch - Count up or down
- Unit Converter - Convert between units
- Tic-Tac-Toe - Two player game
- Mad Libs - Fill in the blanks story
- Color Mixer - Mix RGB values to see colors

Tkinter Widget Attributes - Quick Reference

Label Attributes

Attribute	Description	Example
`text`	Text to display	`text="Hello"`
`font`	Font style	`font=("Arial", 12)` or `font=("Arial", 12, "bold")`
`fg`	Text color	`fg="blue"` or `fg="#0000FF"`
`bg`	Background color	`bg="yellow"`
`width`	Width in characters	`width=20`
`height`	Height in lines	`height=3`
`anchor`	Text position	`anchor=tk.W` (left), `tk.E` (right), `tk.CENTER`
`justify`	Multi-line alignment	`justify=tk.LEFT`, `tk.CENTER`, `tk.RIGHT`
`padx`	Horizontal padding	`padx=10`
`pady`	Vertical padding	`pady=10`
`relief`	Border style	`relief=tk.RAISED`, `tk.SUNKEN`, `tk.FLAT`
`borderwidth`	Border thickness	`borderwidth=2`
`wraplength`	Wrap text width	`wraplength=200`

Button Attributes

Attribute	Description	Example
`text`	Button text	`text="Click Me"`
`command`	Function to call	`command=my_function` (no parentheses!)
`font`	Font style	`font=("Arial", 12, "bold")`
`fg`	Text color	`fg="white"`
`bg`	Background color	`bg="green"`
`width`	Width in characters	`width=15`
`height`	Height in lines	`height=2`
`state`	Button state	`state=tk.NORMAL`, `tk.DISABLED`, `tk.ACTIVE`
`activebackground`	Color when clicked	`activebackground="lightgreen"`
`activeforeground`	Text color when clicked	`activeforeground="black"`
`relief`	Button style	`relief=tk.RAISED`, `tk.FLAT`, `tk.GROOVE`
`borderwidth`	Border thickness	`borderwidth=3`
`padx`	Horizontal padding	`padx=10`
`pady`	Vertical padding	`pady=5`

Entry Attributes

Attribute	Description	Example
`width`	Width in characters	`width=20`
`font`	Font style	`font=("Arial", 12)`
`fg`	Text color	`fg="black"`
`bg`	Background color	`bg="white"`
`show`	Hide characters (password)	`show="*"`
`textvariable`	Link to StringVar	`textvariable=my_var`
`justify`	Text alignment	`justify=tk.LEFT`, `tk.CENTER`, `tk.RIGHT`
`state`	Entry state	`state=tk.NORMAL`, `tk.DISABLED`, `tk.READONLY`
`relief`	Border style	`relief=tk.SUNKEN`
`borderwidth`	Border thickness	`borderwidth=2`

Entry Methods:

- `entry.get()` - Get text
- `entry.insert(0, "text")` - Insert text at position
- `entry.delete(0, tk.END)` - Clear text

Text Attributes

Attribute	Description	Example
`width`	Width in characters	`width=40`
`height`	Height in lines	`height=10`
`font`	Font style	`font=("Arial", 12)`
`fg`	Text color	`fg="black"`
`bg`	Background color	`bg="white"`
`wrap`	Text wrapping	`wrap=tk.WORD`, `tk.CHAR`, `tk.NONE`
`state`	Text state	`state=tk.NORMAL`, `tk.DISABLED`
`relief`	Border style	`relief=tk.SUNKEN`
`borderwidth`	Border thickness	`borderwidth=2`

Text Methods:

- `text.get("1.0", tk.END)` - Get all text
- `text.insert(tk.END, "text")` - Append text
- `text.delete("1.0", tk.END)` - Clear text

Checkbutton Attributes

Attribute	Description	Example
`text`	Label text	`text="I agree"`
`variable`	BooleanVar to store state	`variable=my_bool_var`
`command`	Function when toggled	`command=my_function`
`font`	Font style	`font=("Arial", 12)`
`fg`	Text color	`fg="black"`
`bg`	Background color	`bg="white"`
`onvalue`	Value when checked	`onvalue=1`
`offvalue`	Value when unchecked	`offvalue=0`
`activebackground`	Color when active	`activebackground="lightblue"`
`selectcolor`	Checkbox color	`selectcolor="green"`

Radiobutton Attributes

Attribute	Description	Example
`text`	Label text	`text="Option 1"`
`variable`	Shared variable	`variable=choice_var`
`value`	This button's value	`value="option1"`
`command`	Function when selected	`command=my_function`
`font`	Font style	`font=("Arial", 12)`
`fg`	Text color	`fg="black"`
`bg`	Background color	`bg="white"`
`activebackground`	Color when active	`activebackground="lightblue"`
`selectcolor`	Button color	`selectcolor="blue"`

Listbox Attributes

Attribute	Description	Example
`width`	Width in characters	`width=30`
`height`	Visible items	`height=10`
`font`	Font style	`font=("Arial", 12)`
`fg`	Text color	`fg="black"`
`bg`	Background color	`bg="white"`
`selectmode`	Selection mode	`selectmode=tk.SINGLE` , `tk.MULTIPLE`
`selectbackground`	Selection color	`selectbackground="blue"`
`selectforeground`	Selected text color	`selectforeground="white"`
`relief`	Border style	`relief=tk.SUNKEN`
`borderwidth`	Border thickness	`borderwidth=2`

Listbox Methods:

- `listbox.insert(tk.END, "item")` - Add item
- `listbox.delete(0)` - Delete item at index
- `listbox.get(0)` - Get item at index
- `listbox.curselection()` - Get selected index

Frame Attributes

Attribute	Description	Example
`width`	Width in pixels	`width=300`
`height`	Height in pixels	`height=200`
`bg`	Background color	`bg="lightgray"`
`relief`	Border style	`relief=tk.RAISED` , `tk.SUNKEN` , `tk.GROOVE`
`borderwidth`	Border thickness	`borderwidth=2`
`padx`	Horizontal padding	`padx=10`
`pady`	Vertical padding	`pady=10`

Canvas Attributes

Attribute	Description	Example
`width`	Width in pixels	`width=400`
`height`	Height in pixels	`height=300`
`bg`	Background color	`bg="white"`
`relief`	Border style	`relief=tk.SUNKEN`
`borderwidth`	Border thickness	`borderwidth=2`

Canvas Methods:

- `canvas.create_line(x1, y1, x2, y2)` - Draw line
- `canvas.create_rectangle(x1, y1, x2, y2)` - Draw rectangle
- `canvas.create_oval(x1, y1, x2, y2)` - Draw oval/circle
- `canvas.create_text(x, y, text="text")` - Draw text

Scale (Slider) Attributes

Attribute	Description	Example
`from_`	Minimum value	`from_=0`
`to`	Maximum value	`to=100`
`orient`	Orientation	`orient=tk.HORIZONTAL`, `tk.VERTICAL`
`length`	Length in pixels	`length=200`
`width`	Width in pixels	`width=20`
`label`	Label text	`label="Volume"`
`variable`	IntVar to store value	`variable=my_int_var`
`command`	Function when changed	`command=my_function`
`resolution`	Step size	`resolution=5`

Spinbox Attributes

Attribute	Description	Example
`from_`	Minimum value	`from_=0`
`to`	Maximum value	`to=100`
`width`	Width in characters	`width=10`
`font`	Font style	`font=("Arial", 12)`
`textvariable`	IntVar to store value	`textvariable=my_var`
`wrap`	Wrap around	`wrap=True`
`values`	List of values	`values=("Small", "Medium", "Large")`

Menu (Simple Example)

python

```
menubar = tk.Menu(window)
window.config(menu=menubar)

file_menu = tk.Menu(menubar, tearoff=0)
menubar.add_cascade(label="File", menu=file_menu)
file_menu.add_command(label="New", command=new_file)
file_menu.add_command(label="Open", command=open_file)
file_menu.add_separator()
file_menu.add_command(label="Exit", command=window.quit)
```

Common Layout Options

pack() Options

Option	Description	Example
`side`	Which side	`side=tk.TOP`, `tk.BOTTOM`, `tk.LEFT`, `tk.RIGHT`
`fill`	Fill space	`fill=tk.X`, `tk.Y`, `tk.BOTH`, `tk.NONE`
`expand`	Expand to fill	`expand=True` or `expand=False`
`padx`	Horizontal padding	`padx=10`
`pady`	Vertical padding	`pady=10`
`anchor`	Anchor position	`anchor=tk.N`, `tk.S`, `tk.E`, `tk.W`, `tk.CENTER`

grid() Options

Option	Description	Example
`row`	Row number	`row=0`
`column`	Column number	`column=0`
`rowspan`	Span rows	`rowspan=2`
`columnspan`	Span columns	`columnspan=2`
`padx`	Horizontal padding	`padx=5`
`pady`	Vertical padding	`pady=5`
`sticky`	Alignment	`sticky=tk.W`, `tk.E`, `tk.N`, `tk.S`, or `"NSEW"`

place() Options

Option	Description	Example
`x`	X coordinate	`x=100`
`y`	Y coordinate	`y=50`
`relx`	Relative X (0-1)	`relx=0.5` (center)
`rely`	Relative Y (0-1)	`rely=0.5` (center)
`width`	Width in pixels	`width=200`
`height`	Height in pixels	`height=100`
`anchor`	Anchor point	`anchor=tk.CENTER`

Color Reference

Named Colors

`"red"`, `"blue"`, `"green"`, `"yellow"`, `"orange"`, `"purple"`, `"pink"`, `"brown"`, `"black"`, `"white"`, `"gray"`, `"lightblue"`, `"lightgreen"`, `"darkblue"`, `"darkgreen"`, `"lightgray"`, `"darkgray"`, `"cyan"`, `"magenta"`, `"gold"`, `"silver"`, `"navy"`, `"maroon"`, `"olive"`, `"teal"`, `"lime"`, `"aqua"`, `"coral"`, `"salmon"`, `"khaki"`, `"violet"`, `"indigo"`, `"beige"`, `"ivory"`, `"azure"`, `"lavender"`, `"mint"`, `"peach"`, `"cream"`, `"tan"`, `"wheat"`, `"plum"`, `"orchid"`, `"thistle"`, `"turquoise"`, `"skyblue"`, `"steelblue"`, `"royalblue"`, `"dodgerblue"`, `"deepskyblue"`, `"slateblue"`, `"seagreen"`, `"springgreen"`, `"forestgreen"`, `"limegreen"`, `"palegreen"`, `"yellowgreen"`, `"goldenrod"`, `"chocolate"`, `"sienna"`, `"peru"`, `"sandybrown"`, `"tomato"`, `"orangered"`, `"firebrick"`, `"crimson"`, `"hotpink"`, `"deeppink"`

Hex Colors (RGB format)

- `"#RRGGBB"` where RR, GG, BB are hexadecimal values (00-FF)
- Examples: `"#FF0000"` (red), `"#00FF00"` (green), `"#0000FF"` (blue)
- `"#FFFFFF"` (white), `"#000000"` (black), `"#808080"` (gray)

Font Reference

Font Syntax

python

```
font=("Family", size)
font=("Family", size, "style")
font=("Family", size, "style1 style2")
```

Common Font Families

- `"Arial"`, `"Helvetica"`, `"Times New Roman"`, `"Times"`, `"Courier New"`, `"Courier"`, `"Verdana"`, `"Georgia"`, `"Comic Sans MS"`, `"Trebuchet MS"`, `"Impact"`, `"Tahoma"`, `"Calibri"`, `"Consolas"`, `"Monaco"`, `"Lucida Console"`

Font Styles

- `"bold"` - Bold text
- `"italic"` - Italic text
- `"underline"` - Underlined text
- `"overstrike"` - Strikethrough text
- `"bold italic"` - Combined styles

Examples

python

```
font=("Arial", 12)
font=("Times New Roman", 16, "bold")
font=("Courier", 10, "italic")
font=("Helvetica", 14, "bold italic underline")
```

Relief Styles (Border Styles)

- `tk.FLAT` - No border (default)
- `tk.RAISED` - Raised border
- `tk.SUNKEN` - Sunken border
- `tk.GROOVE` - Grooved border
- `tk.RIDGE` - Ridged border
- `tk.SOLID` - Solid border

Constants Reference

Anchor/Alignment

- `tk.N` - North (top)
- `tk.S` - South (bottom)
- `tk.E` - East (right)
- `tk.W` - West (left)
- `tk.NE` - Northeast (top-right)
- `tk.NW` - Northwest (top-left)
- `tk.SE` - Southeast (bottom-right)
- `tk.SW` - Southwest (bottom-left)
- `tk.CENTER` - Center

Special Values

- `tk.END` - End of text/list
- `tk.INSERT` - Current cursor position
- `tk.ACTIVE` - Active item
- `tk.ALL` - All items

States

- `tk.NORMAL` - Normal/enabled
- `tk.DISABLED` - Disabled/grayed out
- `tk.ACTIVE` - Currently active
- `tk.READONLY` - Read-only (Entry)

Boolean Values

- `True` or `False`
- `1` or `0`
- `"yes"` or `"no"`
- `"on"` or `"off"`

```python
"""
WATERSIDE CODE CLUB
Tkinter GUI Programming Guide

Tkinter is Python's built-in library for creating graphical user interfaces (GUIs)
with windows, buttons, text boxes, and more!
"""

import tkinter as tk
from tkinter import messagebox

# ============================================
# EXAMPLE 1: BASIC WINDOW
# ============================================

def basic_window():
    """Create a simple window"""
    window = tk.Tk()
    window.title("My First Window")
    window.geometry("400x300")  # Width x Height
    window.mainloop()

# ============================================
# EXAMPLE 2: LABELS AND BUTTONS
# ============================================

def labels_and_buttons():
    """Labels display text, buttons perform actions"""
    window = tk.Tk()
    window.title("Labels and Buttons")
    window.geometry("400x300")

    # Create a label
    label = tk.Label(window, text="Welcome to Waterside Code Club!",
                     font=("Arial", 16), fg="blue")
    label.pack(pady=20)

    # Create a button
    def button_clicked():
        label.config(text="Button was clicked!")

    button = tk.Button(window, text="Click Me!", command=button_clicked,
                       bg="green", fg="white", font=("Arial", 12))
    button.pack(pady=10)

    window.mainloop()

# ============================================
# EXAMPLE 3: TEXT ENTRY
# ============================================
def text_entry():
    """Get input from the user"""
    window = tk.Tk()
    window.title("Text Entry")
    window.geometry("400x300")

    # Label
    tk.Label(window, text="Enter your name:", font=("Arial", 12)).pack(pady=10)

    # Entry widget (text input)
    name_entry = tk.Entry(window, font=("Arial", 12), width=20)
    name_entry.pack(pady=10)

    # Result label
    result_label = tk.Label(window, text="", font=("Arial", 14), fg="green")
    result_label.pack(pady=20)

    # Button to submit
    def submit():
        name = name_entry.get()
        result_label.config(text=f"Hello, {name}!")

    tk.Button(window, text="Submit", command=submit,
              bg="blue", fg="white").pack(pady=10)

    window.mainloop()

# ============================================
# EXAMPLE 4: COUNTER APP
# ============================================

def counter_app():
    """Simple counter with increment and decrement buttons"""
    window = tk.Tk()
    window.title("Counter App")
    window.geometry("300x250")

    count = 0

    # Display label
    count_label = tk.Label(window, text="0", font=("Arial", 48), fg="blue")
    count_label.pack(pady=30)

    def increment():
        nonlocal count
        count += 1
        count_label.config(text=str(count))

    def decrement():
        nonlocal count
        count -= 1
        count_label.config(text=str(count))

    def reset():
        nonlocal count
        count = 0
        count_label.config(text="0")

    # Button frame
    button_frame = tk.Frame(window)
    button_frame.pack(pady=10)

    tk.Button(button_frame, text="-", command=decrement,
              width=5, font=("Arial", 16)).pack(side=tk.LEFT, padx=5)
    tk.Button(button_frame, text="Reset", command=reset,
              width=8, font=("Arial", 16)).pack(side=tk.LEFT, padx=5)
    tk.Button(button_frame, text="+", command=increment,
              width=5, font=("Arial", 16)).pack(side=tk.LEFT, padx=5)

    window.mainloop()

# ============================================
# EXAMPLE 5: CALCULATOR
# ============================================

def simple_calculator():
    """Basic calculator app"""
    window = tk.Tk()
    window.title("Simple Calculator")
    window.geometry("300x350")

    # Display
    display = tk.Entry(window, font=("Arial", 20), justify="right")
    display.pack(pady=20, padx=20, fill=tk.BOTH)

    def button_click(value):
        current = display.get()
        display.delete(0, tk.END)
        display.insert(0, current + str(value))

    def clear():
        display.delete(0, tk.END)

    def calculate():
        try:
            result = eval(display.get())
            display.delete(0, tk.END)
            display.insert(0, str(result))
        except:
            display.delete(0, tk.END)
            display.insert(0, "Error")
```

```python
    # Button frame
    button_frame = tk.Frame(window)
    button_frame.pack()

    # Create buttons
    buttons = [
        ['7', '8', '9', '/'],
        ['4', '5', '6', '*'],
        ['1', '2', '3', '-'],
        ['0', '.', '=', '+']
    ]

    for i, row in enumerate(buttons):
        for j, btn_text in enumerate(row):
            if btn_text == '=':
                btn = tk.Button(button_frame, text=btn_text, width=5, height=2,
                               font=("Arial", 14), command=calculate)
            else:
                btn = tk.Button(button_frame, text=btn_text, width=5, height=2,
                               font=("Arial", 14),
                               command=lambda x=btn_text: button_click(x))
            btn.grid(row=i, column=j, padx=2, pady=2)

    # Clear button
    tk.Button(button_frame, text="Clear", width=22, height=2,
              font=("Arial", 14), command=clear, bg="red",
              fg="white").grid(row=4, column=0, columnspan=4, pady=5)

    window.mainloop()

# ============================================
# EXAMPLE 6: CHECKBOXES AND RADIO BUTTONS
# ============================================

def checkboxes_and_radio():
    """Multiple choice widgets"""
    window = tk.Tk()
    window.title("Checkboxes and Radio Buttons")
    window.geometry("400x400")

    tk.Label(window, text="Choose your toppings:",
             font=("Arial", 14, "bold")).pack(pady=10)

    # Checkboxes (multiple selection)
    cheese = tk.BooleanVar()
    pepperoni = tk.BooleanVar()
    mushrooms = tk.BooleanVar()

    tk.Checkbutton(window, text="Cheese", variable=cheese,
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)
    tk.Checkbutton(window, text="Pepperoni", variable=pepperoni,
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)
    tk.Checkbutton(window, text="Mushrooms", variable=mushrooms,
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)

    tk.Label(window, text="\nChoose size:",
             font=("Arial", 14, "bold")).pack(pady=10)

    # Radio buttons (single selection)
    size = tk.StringVar(value="Medium")

    tk.Radiobutton(window, text="Small", variable=size, value="Small",
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)
    tk.Radiobutton(window, text="Medium", variable=size, value="Medium",
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)
    tk.Radiobutton(window, text="Large", variable=size, value="Large",
                   font=("Arial", 12)).pack(anchor=tk.W, padx=50)

    # Result label
    result = tk.Label(window, text="", font=("Arial", 12), fg="green")
    result.pack(pady=20)

    def order():
        toppings = []
        if cheese.get():
            toppings.append("Cheese")
        if pepperoni.get():
            toppings.append("Pepperoni")
        if mushrooms.get():
            toppings.append("Mushrooms")

        order_text = f"{size.get()} pizza with: {', '.join(toppings)}"
        result.config(text=order_text)

    tk.Button(window, text="Order!", command=order, bg="orange",
              fg="white", font=("Arial", 12)).pack(pady=10)

    window.mainloop()

# ============================================
# EXAMPLE 7: LISTBOX
# ============================================

def listbox_example():
    """Display a list of items"""
    window = tk.Tk()
    window.title("To-Do List")
    window.geometry("400x400")

    tk.Label(window, text="My To-Do List",
             font=("Arial", 16, "bold")).pack(pady=10)
    # Entry for new tasks
    task_entry = tk.Entry(window, font=("Arial", 12), width=30)
    task_entry.pack(pady=10)

    # Listbox to display tasks
    listbox = tk.Listbox(window, font=("Arial", 12), width=30, height=10)
    listbox.pack(pady=10)

    def add_task():
        task = task_entry.get()
        if task:
            listbox.insert(tk.END, task)
            task_entry.delete(0, tk.END)

    def delete_task():
        try:
            listbox.delete(listbox.curselection())
        except:
            pass

    # Buttons
    button_frame = tk.Frame(window)
    button_frame.pack(pady=10)

    tk.Button(button_frame, text="Add Task", command=add_task,
              bg="green", fg="white", width=12).pack(side=tk.LEFT, padx=5)
    tk.Button(button_frame, text="Delete Task", command=delete_task,
              bg="red", fg="white", width=12).pack(side=tk.LEFT, padx=5)

    window.mainloop()

# ============================================
# EXAMPLE 8: COLOR CHANGER
# ============================================

def color_changer():
    """Change background color with buttons"""
    window = tk.Tk()
    window.title("Color Changer")
    window.geometry("400x300")

    tk.Label(window, text="Click a button to change the background color!",
             font=("Arial", 12)).pack(pady=20)

    colors = ["red", "blue", "green", "yellow", "orange", "purple", "pink"]

    button_frame = tk.Frame(window)
    button_frame.pack(pady=20)

    for i, color in enumerate(colors):
```

```python
          tk.Button(button_frame, text=color.upper(), bg=color, fg="white",
                    width=10, command=lambda c=color: window.config(bg=c)).grid(
                    row=i//3, column=i%3, padx=5, pady=5)

    window.mainloop()

# ============================================
# RUN EXAMPLES - Uncomment one to run!
# ============================================

if __name__ == "__main__":
    # Uncomment the example you want to run:

    # basic_window()
    # labels_and_buttons()
    # text_entry()
    counter_app()
    # simple_calculator()
    # checkboxes_and_radio()
    # listbox_example()
    # color_changer()
```