An explanation of various GUI components from Tkinter library with Examples...

Got it! I'll make sure to use a complete example with a **window** and demonstrate the component in action. This will make it clearer and more relatable for beginners. Here's an updated version, where I'll create a window and utilize the widgets with some actual data.

Note: during practicing this, create a separate Python file for each widget to clearly observe it.

1. Label

- **Priority**: Very High (Basic)
- **Purpose**: Displays text or images in the window.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Label Example")

# Create a label widget
label = tk.Label(root, text="Hello, Tkinter!", font=("Arial", 20))
label.pack(pady=20)  # Add some space around the label

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

- **Data**: In this example, the label displays a simple text, "Hello, Tkinter!".

2. Button

- **Priority**: High
- **Purpose**: Allows the user to interact with the application by clicking.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Button Example")

# Function to be called when button is clicked
def on_button_click():
    label.config(text="Button Clicked!")

# Create a label widget
```

```
label = tk.Label(root, text="Click the Button Below", font=("Arial", 14))
label.pack(pady=20)

# Create a button widget
button = tk.Button(root, text="Click Me", command=on_button_click)
button.pack()

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

- **Data**: This example uses a button that changes the label text when clicked.

3. Entry

- **Priority**: High
- **Purpose**: Provides a space where the user can enter a single line of text.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Entry Example")

# Function to show entered text
def show_entry():
    entered_text = entry.get()
    label.config(text=f"You entered: {entered_text}")

# Create a label widget
label = tk.Label(root, text="Enter some text", font=("Arial", 14))
label.pack(pady=10)

# Create an entry widget
entry = tk.Entry(root, font=("Arial", 16))
entry.pack(pady=10)

# Create a button widget
button = tk.Button(root, text="Submit", command=show_entry)
button.pack()

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

- **Data**: In this example, the user enters text into the entry box, and upon pressing the submit button, the entered text is displayed in the label.

4. Checkbutton

- **Priority**: Medium
- **Purpose**: Allows users to select or deselect an option (checkbox).

```
import tkinter as tk
```

```
# Create the root window
root = tk.Tk()
root.title("Checkbutton Example")

# Function to show checkbox state
def show_check_state():
    if checkbox_var.get():
        label.config(text="You accepted the terms!")
    else:
        label.config(text="You did not accept the terms.")

# Create a label widget
label = tk.Label(root, text="Check the box if you accept", font=("Arial", 
14))
label.pack(pady=10)

# Create a checkbutton widget
checkbox_var = tk.BooleanVar()  # Variable to store the state of the 
checkbox
checkbox = tk.Checkbutton(root, text="I accept the terms", 
variable=checkbox_var, command=show_check_state)
checkbox.pack(pady=10)

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

- **Data**: This example shows how the checkbox state (checked or unchecked) affects the label text.

5. Radiobutton

- **Priority**: Medium
- **Purpose**: Lets users select one option from a set of choices.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Radiobutton Example")

# Function to display the selected color
def show_selected_color():
    selected_color = color_choice.get()
    label.config(text=f"You selected: {selected_color}")

# Create a label widget
label = tk.Label(root, text="Choose your favorite color", font=("Arial", 
14))
label.pack(pady=10)

# Create a variable to hold the selected color
color_choice = tk.StringVar(value="Red")

# Create radiobutton widgets
```

```
radiobutton1 = tk.Radiobutton(root, text="Red", variable=color_choice,
value="Red", command=show_selected_color)
radiobutton1.pack()

radiobutton2 = tk.Radiobutton(root, text="Green", variable=color_choice,
value="Green", command=show_selected_color)
radiobutton2.pack()

radiobutton3 = tk.Radiobutton(root, text="Blue", variable=color_choice,
value="Blue", command=show_selected_color)
radiobutton3.pack()

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

- **Data**: This example shows a set of radio buttons to choose a color. When a radio button is clicked, it updates the label with the selected color.

6. Listbox

- **Priority**: Medium
- **Purpose**: Displays a list of items for the user to choose from.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Listbox Example")

# Function to display selected item from the listbox
def show_selected_item():
    selected_item = listbox.get(tk.ACTIVE)
    label.config(text=f"You selected: {selected_item}")

# Create a label widget
label = tk.Label(root, text="Select a fruit", font=("Arial", 14))
label.pack(pady=10)

# Create a listbox widget
listbox = tk.Listbox(root)
listbox.pack(pady=10)

# Add items to the listbox
fruits = ["Apple", "Banana", "Orange", "Mango"]
for fruit in fruits:
    listbox.insert(tk.END, fruit)

# Create a button to show selected item
button = tk.Button(root, text="Show Selection", command=show_selected_item)
button.pack()

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

- **Data**: In this example, the user can select a fruit from the listbox, and the label displays the selected fruit.

7. Scale

- **Priority**: Low
- **Purpose**: A slider widget that lets the user select a value from a range.

```
import tkinter as tk

# Create the root window
root = tk.Tk()
root.title("Scale Example")

# Function to show the selected value
def show_scale_value(val):
    label.config(text=f"Selected value: {val}")

# Create a label widget
label = tk.Label(root, text="Select a volume", font=("Arial", 14))
label.pack(pady=10)

# Create a scale widget
scale = tk.Scale(root, from_=0, to=100, orient=tk.HORIZONTAL,
command=show_scale_value)
scale.pack(pady=10)

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

- **Data**: This example lets the user adjust a volume level using a slider, and the selected value is shown in the label.

8. Progressbar

- **Priority**: Low
- **Purpose**: Displays a progress bar to show the status of a task.

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

# Create the root window
root = tk.Tk()
root.title("Progressbar Example")

# Function to update progress
def start_progress():
    for i in range(101):
        progress["value"] = i
        root.update_idletasks()

# Create a progressbar widget
progress = Progressbar(root, orient="horizontal", length=200,
mode="determinate")
progress.pack(pady=20)
```

```
# Create a button to start the progress
button = tk.Button(root, text="Start", command=start_progress)
button.pack()

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

- **Data**: This example shows a progress bar that fills up from 0 to 100 when the user clicks the "Start" button.

Bonus Program.... Eshia try bikanay:

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

# Create the main window
root = tk.Tk()
root.title("Progressbar with Timer")
root.geometry("300x150")

# Label to show status
status_label = tk.Label(root, text="Click to start timer",
font=("Arial", 12))
status_label.pack(pady=10)

# Progressbar
progress = Progressbar(root, orient=tk.HORIZONTAL, length=200,
mode='determinate')
progress.pack(pady=10)

# Global variable to hold progress value
progress_value = 0

# Function to update progress
def update_progress():
    global progress_value
    if progress_value <= 100:
        progress["value"] = progress_value
        status_label.config(text=f"Progress:
{progress_value}%")
        progress_value += 1
        root.after(210, update_progress)  # call this function
again after 50 ms
    else:
        status_label.config(text="Task Completed! ✅")

# Function to start the timer-based progress
def start_progress():
    global progress_value
    progress_value = 0
    update_progress()

# Button to start
start_button = tk.Button(root, text="Start Timer",
command=start_progress)
start_button.pack(pady=10)
```

```
# Run the app
root.mainloop()
```