5 Labels the Unsung Heroes of UI Forms

# What is a Label in the Tk World?

The label is like one of the most overlooked things when creating a form, using TK and Python. If it were not for the label, you wouldn’t understand what to do with any of the other elements on the form. The label will tell you stuff, like write your name here, or check this box there. Not only can you use it for guidance, but you can also use it for communication. You can give a message, show the user what the computer just calculated, and screen readers rely on the label to give directions for accessibility reasons too. Yes, the label is certainly a very important part, and you really could not get by in creating a form without it.

# How should you go about using a label?

First of all, a label needs to be as descriptive as possible, do not use vague names or terms, or you will just confuse your readers all the more.

# How do you create a label on the form?

Start off with your basic code.

#=======Tkinter Library Imports =======

import tkinter as tk

from tkinter import ttk

# Creating the visiable application window

root = tk.Tk()

# Optional: Set window properties

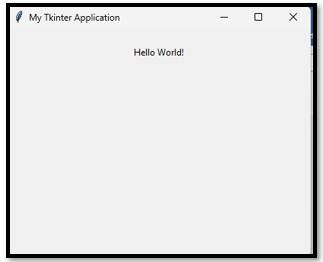
root.title("My Tkinter Application")  # Title of the window

root.geometry("400x300")  # Set the window size (width x height)

# This will show you the window

root.mainloop()

Next just add a label. With intellisense this is a whole lot easier than it has been in the past. Basically all you have to do is to type out something like ‘myLabel’ and intellisense will take off and write the rest of the code for you.



myLabel = tk.Label(root, text="Hello World!")

myLabel.pack(pady=10) # Add some padding

# Add some styling through config



Attach your label to the config method, which is a built in method of TK, use the dot operator to attach it.

myLabel = tk.Label(root, text="Hello World!")

myLabel.config(font=("Arial", 20), fg="blue", bg="lightgray", border=2, relief="ridge", padx=10, pady=10)

myLabel.pack(pady=20) # Add some padding

# How to bind the Label to a click event

Here we are going to write a very simple function. Look how event is used as the parameter. So, it knows everything it needs to know to be an event, just by placing that word in there. You do not need to know how an event works, all you need to do is snag it, throw it into the little parenthesis in the function definition, and now it knows how to be an event.



def on\_label\_click(event):

In your definition, you want to do sort of what you did above, using the config method, but this time it is being done inside of a function so it can be dynamic. All we are doing with the config method is changing the text on the label. So,, when you click on the label, it will change what it is saying, to instead say what is passed to it through its little parenthesis.

def on\_label\_click(event):

    myLabel.config(text="You clicked the label!")

Now you told it what you wanted it to do in the function, but it still isn’t actually doing anything. That is because we need to bind the label to the function that we just wrote. If you do not bind it – it will not work. So, we had better bind it.

def on\_label\_click(event):

    myLabel.config(text="You clicked the label!")

myLabel.bind("<Button-1>", on\_label\_click)

And now when you click on the label, this is what it is telling you.

