

# How to change and a new window in Tk

Multiple Windows/Frames in Tkinter GUI with Python



Now that we have our back-end to your Tkinter GUI application, we're ready to use buttons to navigate to new frames and windows.

What we do here will be your typical methodology for adding more and more pages, basically to infinity.



First, we need to just slightly modify our SeaofBTCapp class. Here's the new full class:



```
class SeaofBTCapp(tk.Tk):

    def __init__(self, *args, **kwargs):

        tk.Tk.__init__(self, *args, **kwargs)
        container = tk.Frame(self)

        container.pack(side="top", fill="both", expand = True)

        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)

        self.frames = {}

        for F in (StartPage, PageOne, PageTwo):

            frame = F(container, self)
```



```
frame.grid(row=0, column=0, sticky="nsew")
```

[Home](#)
[+=1](#)
[Support the Content](#)
[Community](#)

```
self.show_frame(StartPage)
```

[Log in](#)
[Sign up](#)

```
def show_frame(self, cont):
```

```
    frame = self.frames[cont]
```

```
    frame.tkraise()
```

Notice the major change being:

```
for F in (StartPage, PageOne, PageTwo):
```

```
    frame = F(container, self)
```

```
    self.frames[F] = frame
```

```
    frame.grid(row=0, column=0, sticky="nsew")
```

What we do here is populate this tuple with all of the possible pages to our application. This will load all of these pages for us. Within our `__init__` method, we're calling `StartPage` to show first, but later we can call upon `show_frame` to raise any other frame/window that we please.



So we've created `PageOne` and `PageTwo`. We need to have some navigation to these pages from the `StartPage`, so here's our new `StartPage` class:

```
class StartPage(tk.Frame):
```

```
    def __init__(self, parent, controller):
```

```
        tk.Frame.__init__(self, parent)
```

```
        label = tk.Label(self, text="Start Page", font=LARGE_FONT)
```

```
        label.pack(pady=10, padx=10)
```



```
button.pack()
```

[Home](#)
[+=1](#)
[Support the Content](#)
[Community](#)

```
button2 = tk.Button(self, text="Visit Page 2",
```

```
command=lambda: controller.show_frame(PageTwo))
```

[Log in](#)
[Sign up](#)

```
button2.pack()
```



Above, we've added buttons that use `controller.show_frame`, passing `PageOne` and `PageTwo` through as parameters.

Then we just need to create `PageOne` and `PageTwo` classes. Easy enough, these can be nearly identical to `StartPage`:

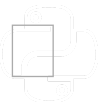
```
class PageOne(tk.Frame):

    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        label = tk.Label(self, text="Page One!!!", font=LARGE_FONT)
        label.pack(pady=10, padx=10)

        button1 = tk.Button(self, text="Back to Home",
                             command=lambda: controller.show_frame(StartPage))
        button1.pack()

        button2 = tk.Button(self, text="Page Two",
                             command=lambda: controller.show_frame(PageTwo))
        button2.pack()

class PageTwo(tk.Frame):
```



```
label = tk.Label(self, text="Page Two!!!", font=LARGE_FONT)
label.pack(pady=10, padx=10)

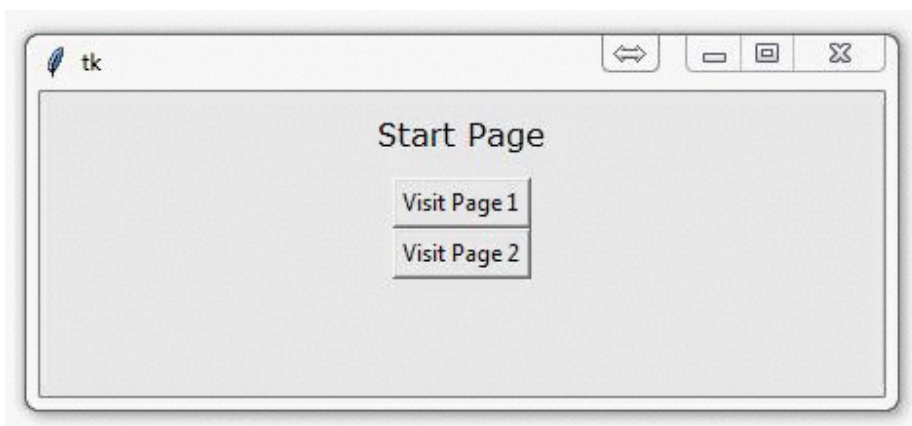
button1 = tk.Button(self, text="Back to Home",
                    command=lambda: controller.show_frame(StartPage))
button1.pack()

button2 = tk.Button(self, text="Page One",
                    command=lambda: controller.show_frame(PageOne))
button2.pack()
```



The only major changes here are the buttons and where they lead to.

Running this code should give you:



In case you got lost, here's the full code:

```
# The code for changing pages was derived from: http://stackoverflow.com/qu
# License: http://creativecommons.org/licenses/by-sa/3.0/
```



LARGE\_FONT= ("Verdana", 12)

Home

+=1

Support the Content

Community

Log in

Sign up

**class** SeaofBTCapp(tk.Tk):

**def** \_\_init\_\_(**self**, \*args, \*\*kwargs):

        tk.Tk.\_\_init\_\_(**self**, \*args, \*\*kwargs)

        container = tk.Frame(**self**)

        container.pack(side="top", fill="both", expand = **True**)

        container.grid\_rowconfigure(0, weight=1)

        container.grid\_columnconfigure(0, weight=1)

**self**.frames = {}

**for** F **in** (StartPage, PageOne, PageTwo):

            frame = F(container, **self**)

**self**.frames[F] = frame

            frame.grid(row=0, column=0, sticky="nsew")

**self**.show\_frame(StartPage)

**def** show\_frame(**self**, cont):

        frame = **self**.frames[cont]

        frame.tkraise()

**class** StartPage(tk.Frame):

**def** \_\_init\_\_(**self**, parent, controller):

        tk.Frame.\_\_init\_\_(**self**,parent)

        label = tk.Label(**self**, text="Start Page", font=LARGE\_FONT)

        label.pack(pady=10,padx=10)

        button = tk.Button(**self**, text="Visit Page 1",



```
button2 = tk.Button(self, text="Visit Page 2",
                    command=lambda: controller.show_frame(PageTwo))
button2.pack()
```

```
class PageOne(tk.Frame):
```

```
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        label = tk.Label(self, text="Page One!!!", font=LARGE_FONT)
        label.pack(pady=10, padx=10)

        button1 = tk.Button(self, text="Back to Home",
                            command=lambda: controller.show_frame(StartPage))
        button1.pack()

        button2 = tk.Button(self, text="Page Two",
                            command=lambda: controller.show_frame(PageTwo))
        button2.pack()
```

```
class PageTwo(tk.Frame):
```

```
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        label = tk.Label(self, text="Page Two!!!", font=LARGE_FONT)
        label.pack(pady=10, padx=10)

        button1 = tk.Button(self, text="Back to Home",
                            command=lambda: controller.show_frame(StartPage))
        button1.pack()

        button2 = tk.Button(self, text="Page One",
                            command=lambda: controller.show_frame(PageOne))
        button2.pack()
```

```
app = SeaofBTCapp()
app.mainloop()
```



There exists 2 quiz/question(s) for this tutorial.

video downloads, and no ads.

Home

←

Sign Up To Help Support the Content

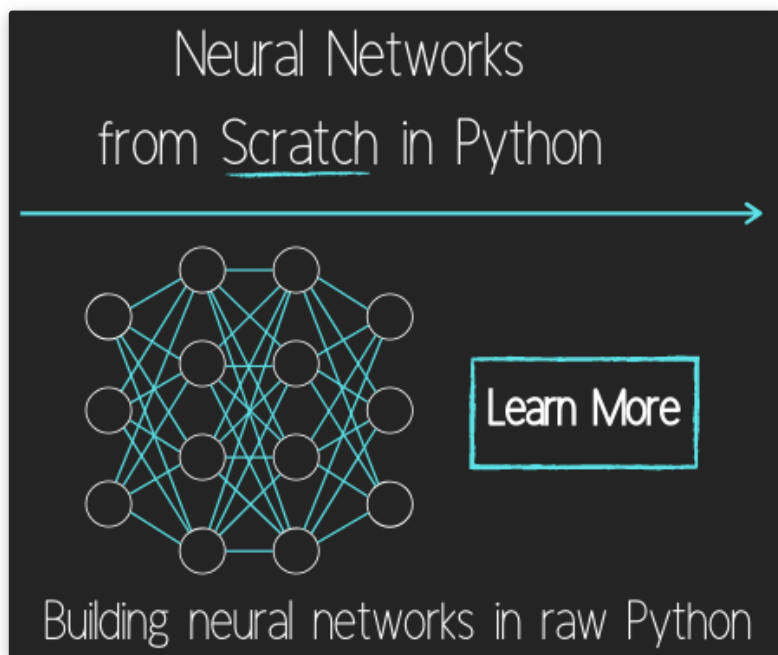
for access to these, Community

Log in

Sign up

The next tutorial:

Styling Your GUI A Bit Using TTK



Programming GUIs and windows with Tkinter and Python Introduction

Object Oriented Programming Crash Course with Tkinter

Passing functions with Parameters in Tkinter using Lambda

How to change and show a new window in Tkinter

Styling your GUI a bit using TTK

How to embed a Matplotlib graph to your Tkinter GUI

How to make the Matplotlib graph live in your application





Plotting Live Updating Data in Matplotlib and our Tkinter GUI

Home

+1

Support the Content

Community

Customizing an embedded Matplotlib Graph in Tkinter

Log in

Sign up

Creating our Main Menu in Tkinter

Building a pop-up message window

Exchange Choice Option

Time-frame and sample size option

Adding indicator Menus (3 videos)

Trading option, start/stop, and help menu options

Tutorial on adding a tutorial

Allowing the exchange choice option to affect actual shown exchange



Adding exchange choice cont'd

Adding exchange choices part 3

Indicator Support

Pulling data from the Sea of BTC API

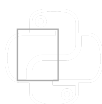
Setting up sub plots within our Tkinter GUI

Graphing an OHLC candlestick graph embedded in our Tkinter GUI

Acquiring RSI data from Sea of BTC API

Acquiring MACD data from Sea of BTC API

Converting Tkinter application to .exe and installer with cx\_Freeze



[Home](#)   [+=1](#)   [Support the Content](#)   [Community](#)

[Log in](#)   [Sign up](#)

Shop The Latest adidas®

adidas Flagship Store New York  
[New York](#) 10AM–8PM

# You've reached the end!

Contact: [Harrison@pythonprogramming.net](mailto:Harrison@pythonprogramming.net).

Support this Website!  
Consulting and Contracting  
[Facebook](#)  
[Twitter](#)  
[Instagram](#)

Legal stuff:  
[Terms and Conditions](#)  
[Privacy Policy](#)



© OVER 9000! [PythonProgramming.net](#)

Programming is a superpower.