Building Practical Desktop apps in Python Using the Core Tkinter Library

By Doug Purcell

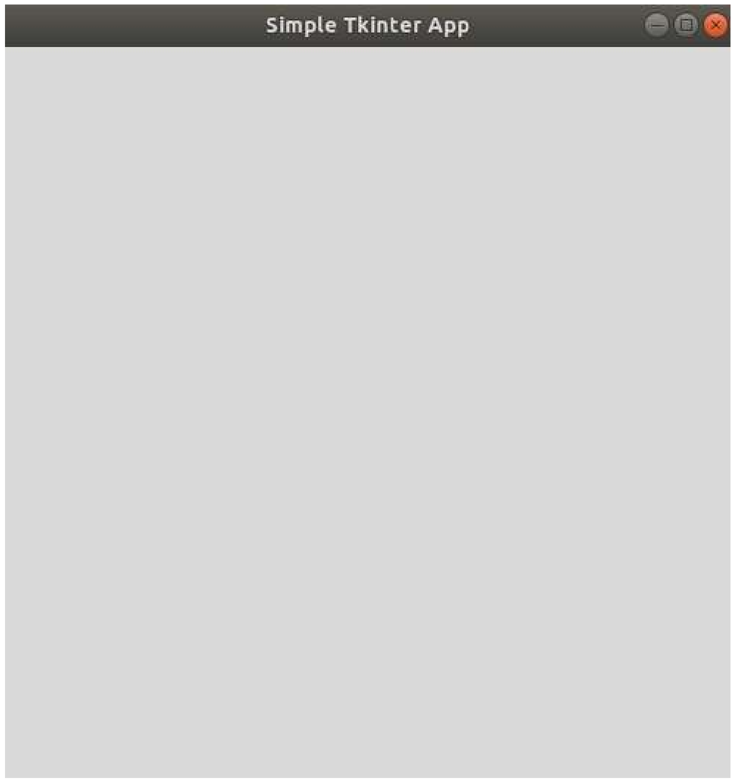
Why Bother Learning How to Code GUI?

- Makes your programs a lot more user friendly. Do you think Microsoft Windows would be as popular if users could only use the terminal?
- Allows business folks or stakeholders to easily interface with the program
- A solid way to get software engineers accustomed to quality UX principles

Why Tkinter?

- It’s wired into python’s core.
- It’s cross platform.
- It’s a wrapper to the standard Tk GUI toolkit.
- It’s easy to get started with.

A Simple Tkinter GUI



The Code

```
import tkinter as tk

root = tk.Tk()

root.title('Simple Tkinter App')

root.geometry('500x500')

tk.mainloop()
```

If You Remember Anything, Remember That!

The First Line

```
import tkinter as tk
```

What It Does?

Imports the Tkinter module. You could also use the following to import ALL of the functionality from Tkinter:

```
from tkinter import *
```

The Second Line

```
root = tk.Tk()
```

What It Does?

This starts the `tcl/tk` interpreter under the cover. Then, the tkinter commands are translated into tcl/tk commands.

The Third Line

```
root.title('Simple Tkinter App')
```

What It Does

The title method allows you to set the title of the app. Nothing tricky!

The Fourth Line

```
root.geometry('500x500')
```

What It Does?

This sets the dimensions of the window AND to position the window within the user’s Desktop in **the ‘widthxheight’ format.**

Commit That to Memory

That’s a basic skeleton for a Tkinter GUI.

The Label, Entry, and Button Widgets

Tkinter supports **15 widgets**. A widget is an object that provides various controls that the coder can use to integrate with the GUI.

A Label

A label as the name suggests allows you to display text or an image on the screen. Below is an example on how to create one:

Label Tiny Code Snippet

```
name = tk.Label(root, text='Your favorite food?').pack()
```

Label Tiny Code Snippet Analysis

- So, we call the `Label` method using `tk`, and then pass in the root which is the parent of the class.
- We can then make use of the various attributes in the label widget and text is one of them. As the name indicates it allows you to add text to the label. It’s important to note that the size of the label will automatically be set so that all of the characters in the text attribute are displayed.

A Sample of Attributes of the Label Widget

- width: Change the width of the label widget. If you set this too small then all of the text won't show.
- height: Adjust the height of the label.
- bg: The background color.
- fg: The foreground color.
- bd/borderwidth: The width of the label border.
- padx: Extra horizontal padding to add around the text.
- pady: Extra vertical padding to add around text.
- textvariable: Associates a Tkinter variable, typically a StringVar with the label.

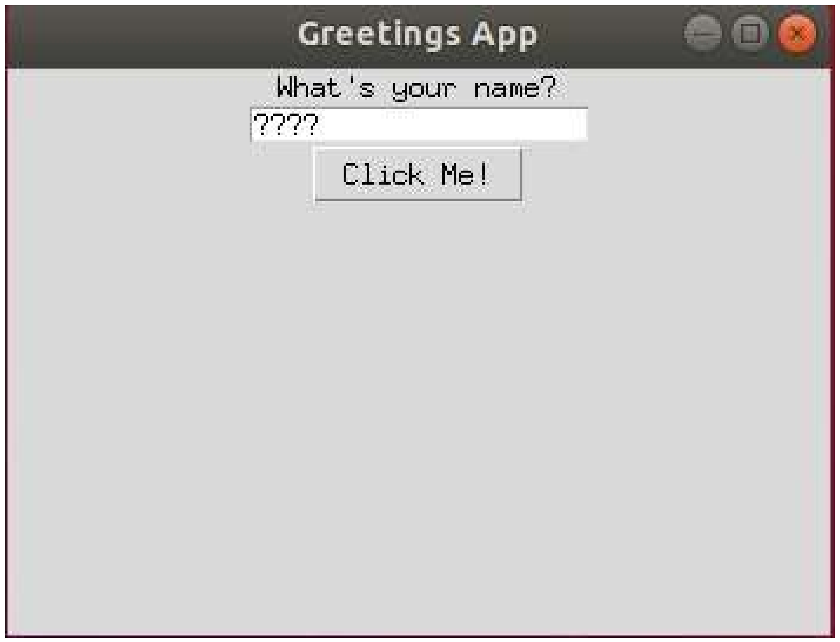
Here’s a Solid Resource

http://effbot.org/tkinterbook/label.htm

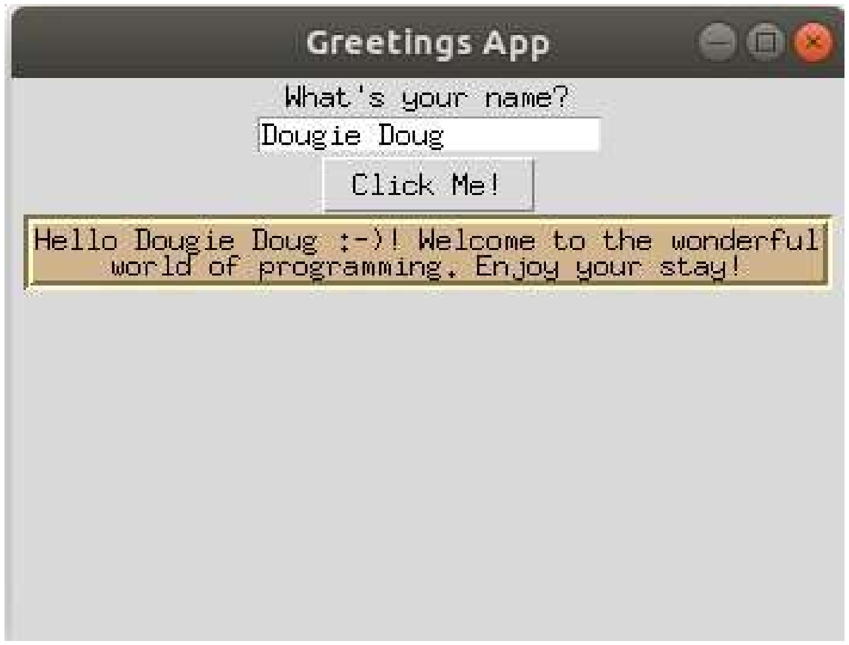
Let’s Build On Top of the Simple Example

Let’s create a simple Greetings app

Greetings App Screenshot 1



Greetings App Screenshot 2



Dougie Doug

```
Hello Dougie Doug :-)! Welcome to the wonderful
world of programming. Enjoy your stay!
```

Run The Demo!

```
$ python tkinter_greeting_app.py
```

The Code Part I

```
import tkinter as tk

root = tk.Tk()   # starts a tcl/tk interpreter under the cover

root.title('Greetings App')

root.geometry('350x250')

def greeting():

           greetings = tk.Label(root, bg='tan', borderwidth=3.5, relief='groove', text="Hello {} :-
)! Welcome to the wonderful\n"

                                "world of programming. Enjoy your
stay!".format(default_value.get()))

           greetings.pack()
```

The Code Part II

```
default_value = tk.StringVar()

default_value.set('????')

first_name_label = tk.Label(root, text='What\'s your name?').pack()

first_name = tk.Entry(root, textvariable=default_value).pack()

button = tk.Button(text='Click Me!', command=greeting).pack()

tk.mainloop()
```

Greetings App Analysis

- If we want text to appear in the Entry widget itself then there's no attribute that we can place within the Entry widget. Instead, we need to create what's called a StringVar variable, and then pass it in as the text variable value of the Entry widget.
- A StringVar is in essence part of the variable classes in tkinter; there's also the BooleanVar, DoubleVar, and IntVar classes. These serve as wrappers for their respective datatype.
- default_value = tk.StringVar()
- default_value.set(**'pizza?'**)
- first_name = tk.Entry(root, textvariable=default_value).pack()

Ordering Layouts In Tkinter

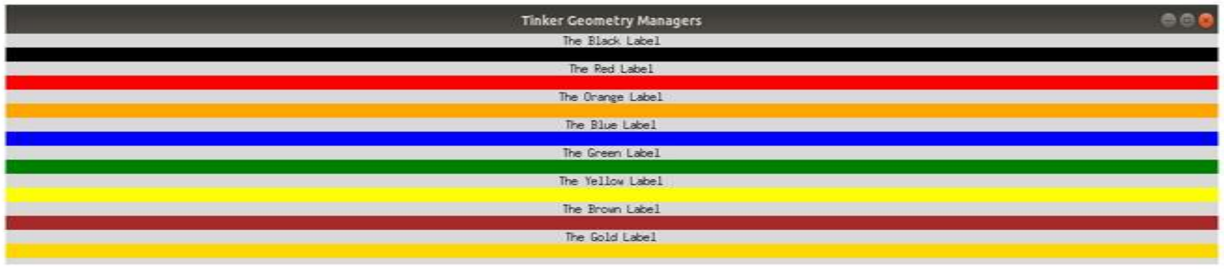
There are three ways in which you can control the layout of items in your GUI, they are:

- pack
- grid
- place

Pack

This is the easiest one to use as once you use it the method takes care of the ordering itself.

Pack GUI Example



Show The Pack Demo

```
$ python tkinter_pack.py
```

Pack Code Snippet Part I

```
import tkinter as tk

root = tk.Tk()

root.title('Tinker Geometry Managers')

colors = ['black', 'red', 'orange', 'blue', 'green',

          'yellow', 'brown', 'gold']

# The label geometry layout manager in Tkinter

label_one = tk.Label(text='The Black Label').pack()

label_one_black = tk.Label(root, bg=colors[0]).pack(fill=tk.X)
```

Pack Code Snippet Part II

```
label_two = tk.Label(text='The Red Label').pack()

label_two_red = tk.Label(root, bg=colors[1]).pack(fill=tk.X)

label_three = tk.Label(text='The Orange Label').pack()

label_three_orange = tk.Label(root, bg=colors[2]).pack(fill=tk.X)

label_four = tk.Label(text='The Blue Label').pack()

label_four_blue = tk.Label(root, bg=colors[3]).pack(fill=tk.X)

label_five = tk.Label(text='The Green Label').pack()

label_five_blue = tk.Label(root, bg=colors[4]).pack(fill=tk.X)

label_six = tk.Label(text='The Yellow Label').pack()

label_six_yellow = tk.Label(root, bg=colors[5]).pack(fill=tk.X)
```

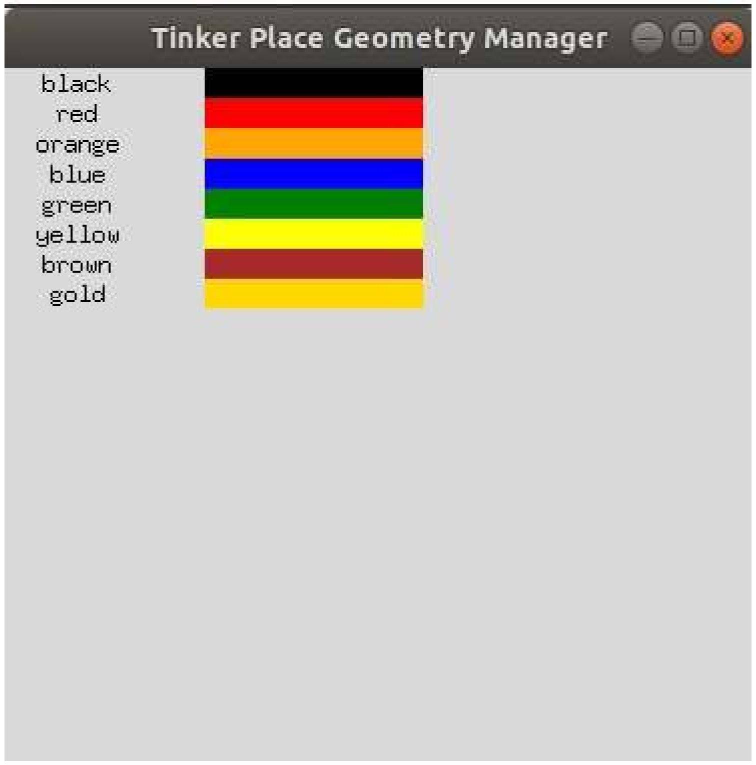
Pack Code Snippet Part III

```
label_seven = tk.Label(text='The Brown Label').pack()

label_six_brown = tk.Label(root, bg=colors[6]).pack(fill=tk.X)

label_eight = tk.Label(text='The Gold Label').pack()

label_six_gold = tk.Label(root, bg=colors[7]).pack(fill=tk.X)

root.mainloop()
```

Place

The place layout manager allows you to do absolute and relative positioning with tkinter. So, you can specify exactly where you want a widget to appear in the GUI.

Place Geometry Manager



Run Place Demo

```
$ python tkinter_place_example.py
```

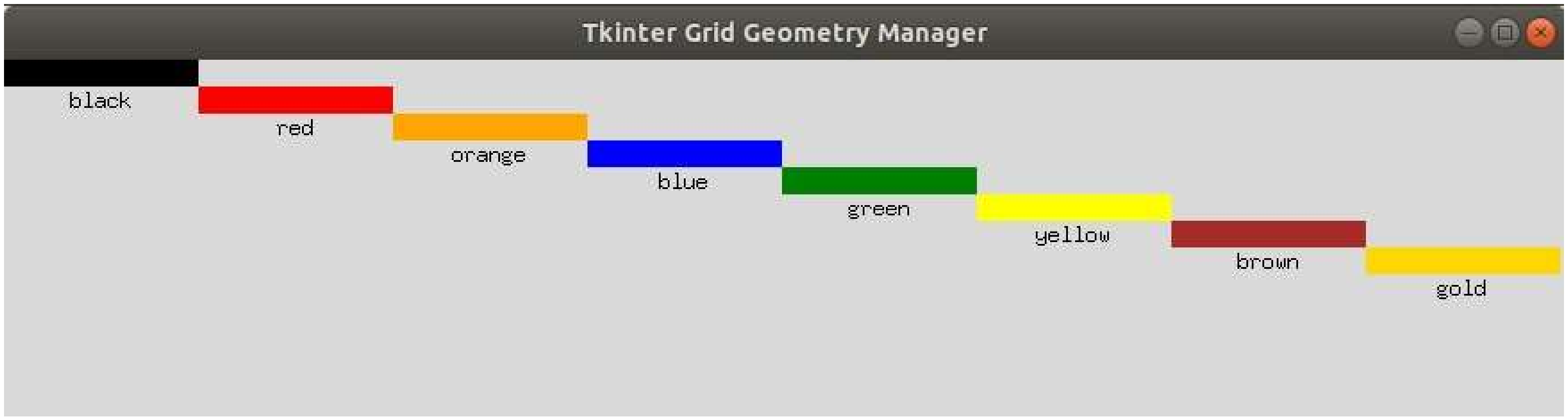
Place Code Snippet

```
import tkinter as tk

root = tk.Tk()

root.title('Tinker Place Geometry Manager')

root.geometry('375x350')

colors = ['black', 'red', 'orange', 'blue', 'green',

            'yellow', 'brown', 'gold']

width, height = 0, 0

for x in range(len(colors)):

            tk.Label(text=colors[x], width=10).place(x=0, y=height)

            tk.Label(bg=colors[x], width=15).place(x=100, y=height)

            height += 15

tk.mainloop()
```

Grid

The last layout manager to explain is the grid. It organizes components of the GUI by placing them in a 2-dimensional table which consists of rows and columns.

Grid Example



Run Grid!

```
$ python tkinter_grid_example.py
```

Grid Code Snippet

```
import tkinter as tk

root = tk.Tk()

root.title('Tkinter Grid Geometry Manager')

root.geometry('875x200')

colors = ['black', 'red', 'orange', 'blue', 'green',

            'yellow', 'brown', 'gold']

i = 0

for x in colors:

            tk.Label(text=x, width=15).grid(row=i+1, column=i)

            tk.Label(bg=x, width=15).grid(row=i, column=i)

            i += 1

tk.mainloop()
```

Converting to OOP

```
import tkinter as tk

class HelloWorld:

              def __init__(self):

              self.root = tk.Tk()

    self.root.title('Simple Tkinter Class Example')

              self.root.geometry('500x500')

if __name__ == '__main__':

              app = HelloWorld()

              tk.mainloop()
```

Temperature Converter

Run program:

```
$ python fahrenheit_to_celsius.py
```

Read project and solution here: https://github.com/purcellconsult/Code-Cool-Stuff-With-Python/blob/master/book/chapter_05.md#project-temperature-converter-gui

BMI Calculator

```
$ python bmi_calculator
```

Analyze project and here: https://github.com/purcellconsult/Code-Cool-Stuff-With-Python/blob/master/book/chapter_05.md#project-bmi-calculator-app

Guessing Game

$ Run program: `python secret_number_game.py`

Analyze project and code here:
https://github.com/purcellconsult/Code-Cool-Stuff-With-Python/blob/master/book/chapter_05.md#project-the-secret-number-game

Hope You Didn’t Forget This :)!

```
import tkinter as tk

root = tk.Tk()

root.title('Simple Tkinter App')

root.geometry('500x500')

tk.mainloop()
```

Thanks! Connect With Me on LinkedIn!

- I’m active on LinkedIn plus my messages there are way less cluttered than email. Simply search Doug Purcell on LinkedIn, I should appear numeral uno
- Need help with understanding python’s basics? Complete my free course on GitHub: **https://github.com/purcellconsult/Cracking-Python-Bootcamp**
- Email me if you need any python help. My email is: **purcellconsult@gmail.com**. If I meet you in person make sure to jot my memory of the meetup or conference.