

Lesson 4: Tkinter

Brought to you by the University of Maryland Balloon
Payload Program

Recap

We can think of an **'object'** as a group of variables in a labeled bin. Another classic example is a customer on a website:

A **customer** has:

- Email
- Password
- Age
- Phone Number
- Address

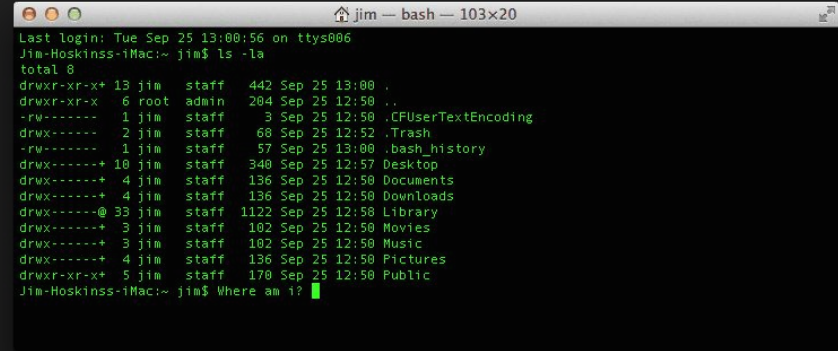
Customer 112345 has:

- Email: "customer1@gmail.com"
- Password: "password123"
- Age: 24
- Phone #: "3012469980"
- Address: "554 Normweller Road,
College Park, MD, 20872"

How People Interact with Code

Developers

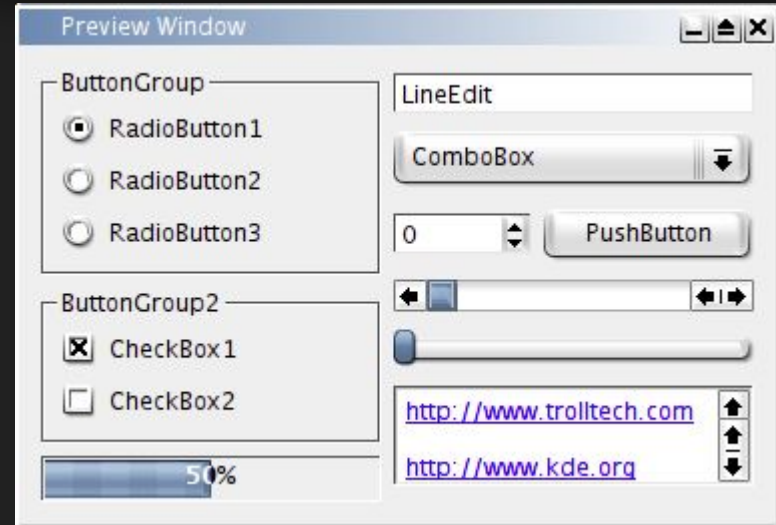
- **Command Line Interface (CLI)**
- Text only
- Fast



```
jim — bash — 103x20
Last login: Tue Sep 25 13:00:56 on ttys006
Jim-Hoskinss-iMac:~ jim$ ls -la
total 8
drwxr-xr-x+ 13 jim  staff  442 Sep 25 13:00 .
drwxr-xr-x+  6 root   admin  204 Sep 25 12:50 ..
-rw-r--r--   1 jim  staff    3 Sep 25 12:50 .CFUserTextEncoding
drwx-----  2 jim  staff   68 Sep 25 12:52 .Trash
-rw-r--r--   1 jim  staff   57 Sep 25 13:00 .bash_history
drwx-----+ 10 jim  staff  340 Sep 25 12:57 Desktop
drwx-----+  4 jim  staff  136 Sep 25 12:50 Documents
drwx-----+  4 jim  staff  136 Sep 25 12:50 Downloads
drwx-----@ 33 jim  staff 1122 Sep 25 12:50 Library
drwx-----+  3 jim  staff  102 Sep 25 12:50 Movies
drwx-----+  3 jim  staff  102 Sep 25 12:50 Music
drwx-----+  4 jim  staff  136 Sep 25 12:50 Pictures
drwxr-xr-x+  5 jim  staff  170 Sep 25 12:50 Public
Jim-Hoskinss-iMac:~ jim$ Where am i?
```

End-Users

- Graphical User Interface (GUI)
- Text, images, buttons, etc.
- Much slower



Importing Tkinter

Official Python library

- Comes with Python code
- Lets us make a GUI

```
import tkinter as tk
```

```
from tkinter import Tk, ttk
```

Anatomy of a Tkinter Code

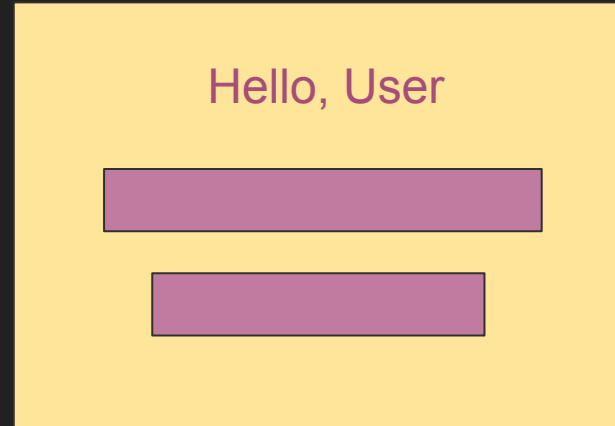
Every Tkinter GUI has a **window** and **widgets**

Windows hold all the widgets

- “Box” for everything the user will interact with
- “root” window as the base

Widgets are everything else

- Buttons, text entry, pictures, etc.



Let's code a simple window

```
import tkinter as tk          # Make sure tkinter is lowercase

root = tk.Tk()                # the *root window* (baseline)
root.title("Hello Tkinter")   # window title text
root.geometry("300x100")      # optional: width x height

root.mainloop()              # Keep code running
```

Making a basic GUI

Now we have a grey window, that's not really useful, so let's add text!

Syntax:

```
label = tk.Label(master, options)
```

Master is the window the label “belongs” to

There are a TON of options

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

Where is our widget?

Remember, the computer will only do what we say

- We need to “place” the label

```
label.pack(side="top/bottom/left/right", pady=y, padx=x)
```

Or

```
label.grid(row=y, column=x)
```

```
root.grid_columnconfigure(cell, weight=q)
```

Other useful widgets

Button

```
btn = tk.Button(master, text="string", command=function)
```

Text Box

```
user_entry = tk.Entry(master, textvariable=variable_name)
```

Let's Make a Login Page

1. Create a username box
2. Create a password box
3. Create a button to submit
4. Create a function for the button
5. Print the username and password

Entry Widgets Have Some Quirks

The variable we use has to be “initialized”

- Tells Python where to store the entry data
- Stored as an object

```
variable_name = tk.StringVar()
```

```
variable = variable_name.get()
```

Tkinter Label Options

- text: Sets the text displayed on the label.
- font: Specifies the font of the text (e.g., ("Helvetica", 12)).
- bg: Sets the background color.
- fg: Sets the foreground (text) color.
- width: Sets the width of the label in characters.
- height: Sets the height of the label in lines.
- padx: Horizontal padding.
- pady: Vertical padding.
- relief: Border style (e.g., solid, raised, sunken).
- anchor: Specifies the position of the text if the widget has more space than required for the text.

Multiple Windows

Coding on our computers

Visual Studio Code

- Where we type all our code
- code.visualstudio.com



Python

- Lets our computer run our written code
- python.org/downloads



What is Visual Studio Code (VSCode)?

A nicer way to write and edit code

- Technically, all code could be written in notepad
- VSCode has tools that make coding easier
 - Autocompletes variable names
 - Colors written code
 - User extensions for additional features
 - Highlights errors
 - More!

```
# Water (includes thermal scattering)  
water = openmc.Material(name='Water')
```

Functions

```
# Define a function for addition
```

```
def add(a, b):  
    return a + b
```

```
# Define a function for subtraction
```

```
def subtract(a, b):  
    return a - b
```

```
# Define a function for multiplication
```

```
def multiply(a, b):  
    return a * b
```

```
# Define a function for division
```

```
def divide(a, b):  
    if b == 0:  
        return "Error! Cannot divide by zero."  
    return a / b
```

```
# Using the functions
```

```
x = 10  
y = 5
```

```
# Call the functions and print the results
```

```
sum_result = add(x, y)  
print("Addition:", sum_result)
```

```
diff_result = subtract(x, y)  
print("Subtraction:", diff_result)
```

```
product_result = multiply(x, y)  
print("Multiplication:", product_result)
```

```
quotient_result = divide(x, y)  
print("Division:", quotient_result)
```