Graphical User Interface

GUI in Python

Tkinter (Tk Interface)

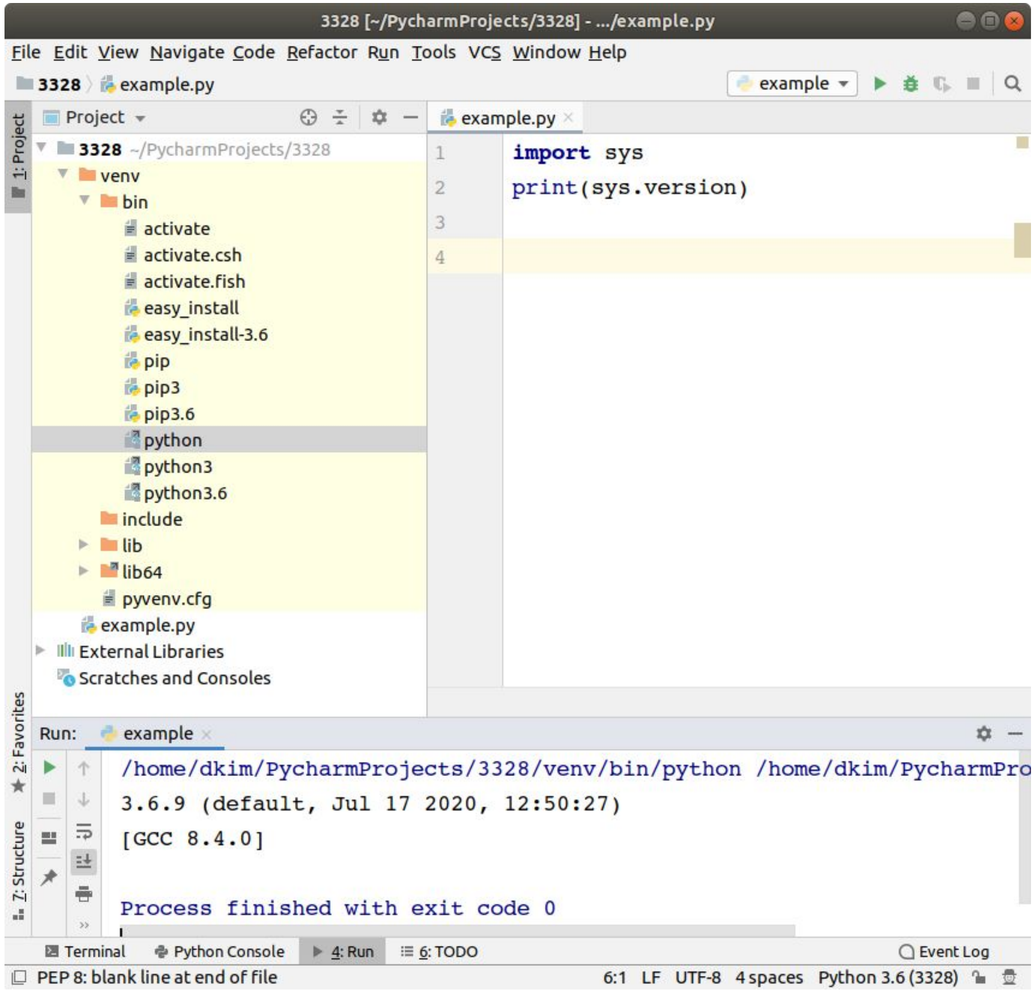
Python has a lot of GUI frameworks, but Tkinter is the only framework that’s built into the Python standard library.

It’s cross-platform, so the same code works on Windows, macOS, and Linux.

Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they’re run.

Use Python 3.6

I am going to use Python **3.6**.
You can use any Python version if you want.

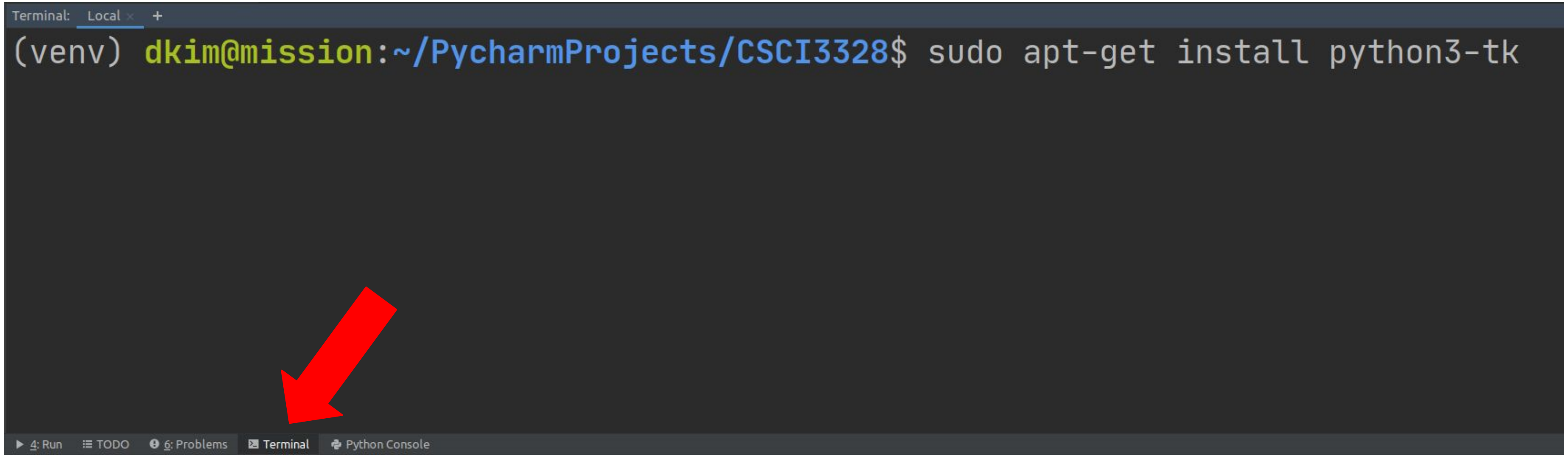


```python
import sys
print(sys.version)
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python /home/dkim/PycharmPro
3.6.9 (default, Jul 17 2020, 12:50:27)
[GCC 8.4.0]

Process finished with exit code 0
```

PEP 8: blank line at end of file 6:1 LF UTF-8 4 spaces Python 3.6 (3328)

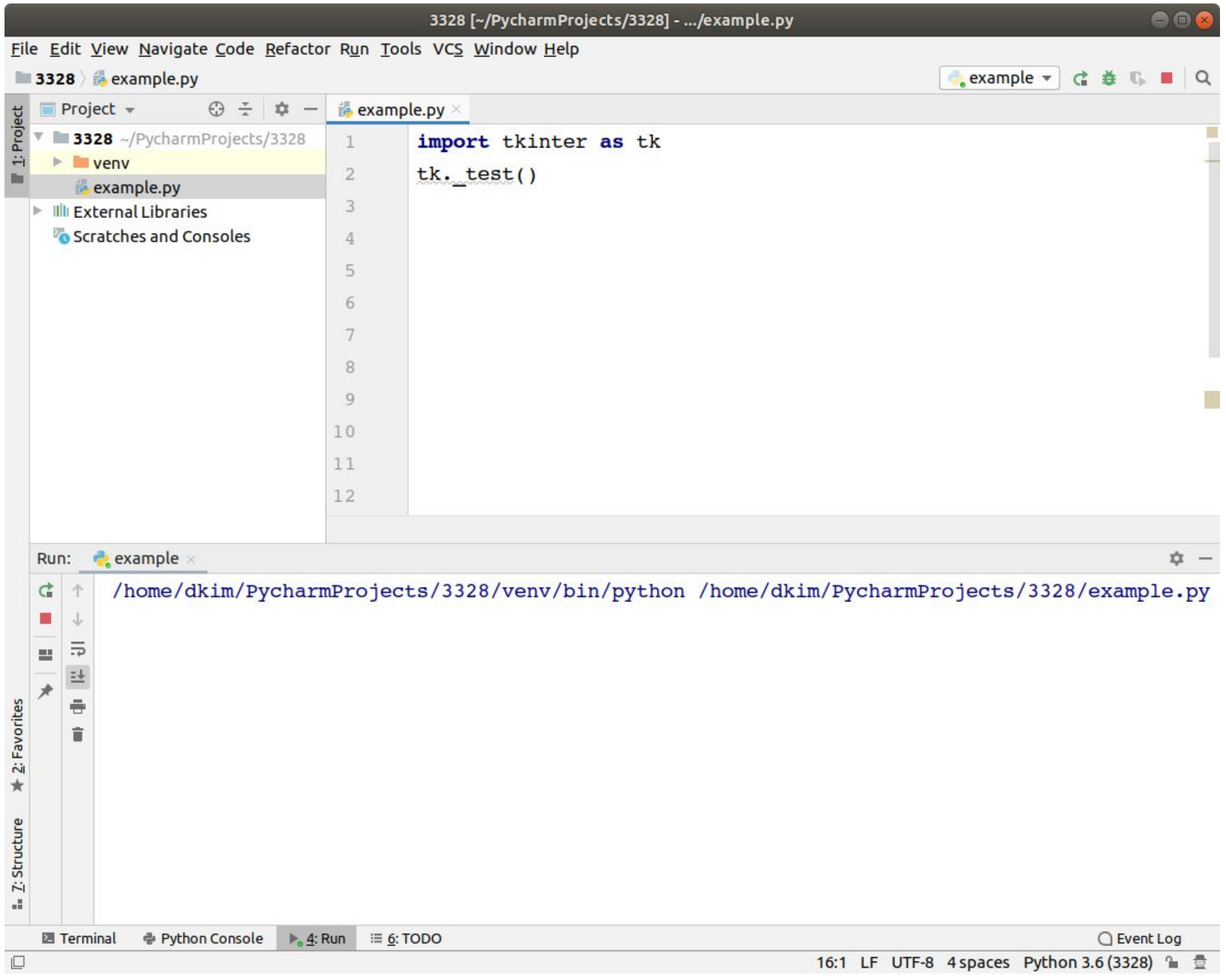
Install tkinter in Pycharm (venv)



```
(venv) dkim@mission:~/PycharmProjects/CSCI3328$ sudo apt-get install python3-tk
```

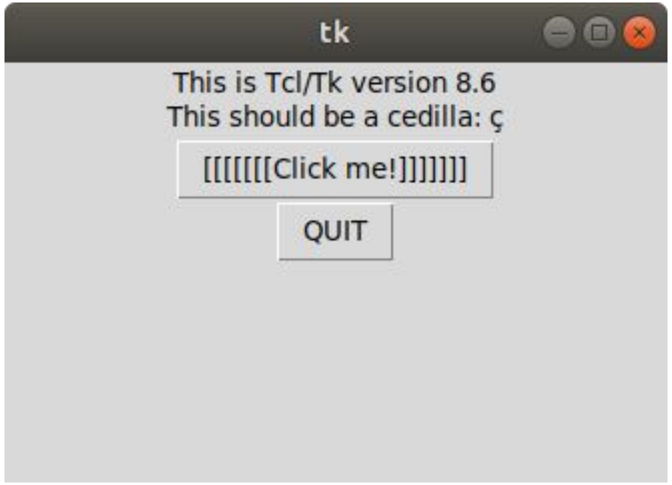
Test

Test `tkinter` module


3328 [~/PycharmProjects/3328] - .../example.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
3328 example.py
example
Project
3328 ~/PycharmProjects/3328
venv
example.py
External Libraries
Scratches and Consoles
example.py
Run: example
Terminal Python Console 4: Run 6: TODO Event Log
16:1 LF UTF-8 4 spaces Python 3.6 (3328)

```python
import tkinter as tk
tk._test()
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python /home/dkim/PycharmProjects/3328/example.py
```



This is Tcl/Tk version 8.6
This should be a cedilla: ç

Tkinter (TK interface)

- Window
    - The foundational element of a Tkinter GUI is the window. Windows are the containers in which all other GUI elements live.
- Widget
    - These other GUI elements, such as text boxes, labels, and buttons, are known as widgets. Widgets are contained inside of windows.

First, let’s create a window!

Tk class and its instance

```
import tkinter as tk
w = tk.Tk()
```

A window we want to create is an **instance of `Tkinter's` `Tk` class**. Go ahead and create a new object and assign it to the variable `w`.

When you execute the above a line, a new window pops up on your screen.

The window interface would look different depending on your operating system.

***Some operating systems or environments (including my case) do need a call to **`mainloop()`** to run the program.

`mainloop()`

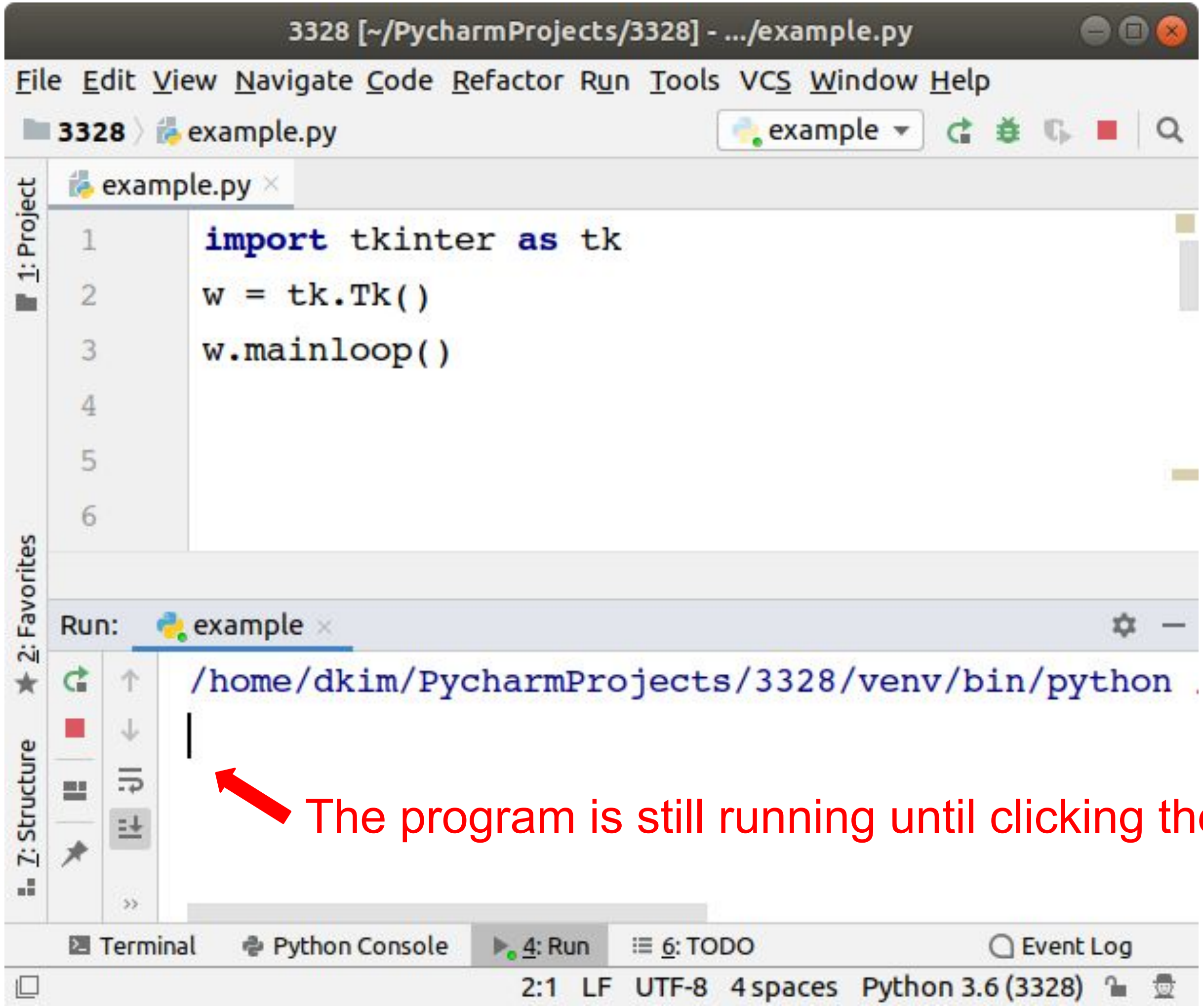
```
import tkinter as tk
w = tk.Tk()
w.mainloop()
```

`w.mainloop()` tells Python to run the Tkinter event loop.

This method listens for events, such as button clicks or keypresses, and blocks any code that comes after it from running until the window it's called on is closed.

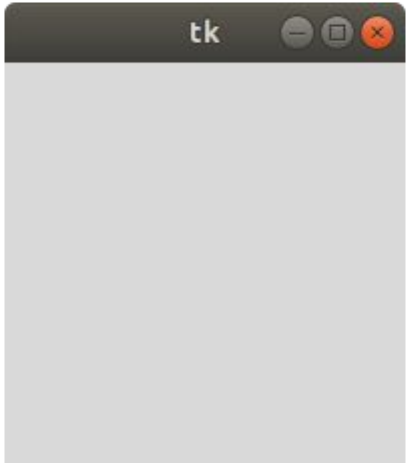
Once you run it, you'll see a blinking cursor in the console. It means the program is still running.

***Don't forget to close it by clicking the close button (x button).


3328 [~/PycharmProjects/3328] - .../example.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
3328
example.py
example
example.py
1: Project
2: Favorites
7: Structure
Run:
example
Terminal
Python Console
4: Run
6: TODO
Event Log
2:1 LF UTF-8 4 spaces Python 3.6 (3328)

```python
import tkinter as tk
w = tk.Tk()
w.mainloop()
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python
```



The program is still running until clicking the x button.

Adding a Widget

Let’s add a Widget.

You can add some text to the window using the tk.Label class.

Create a Label widget with the text "Hello, Dr. Kim" and assign it to a variable called `hello` which is a Label instance.

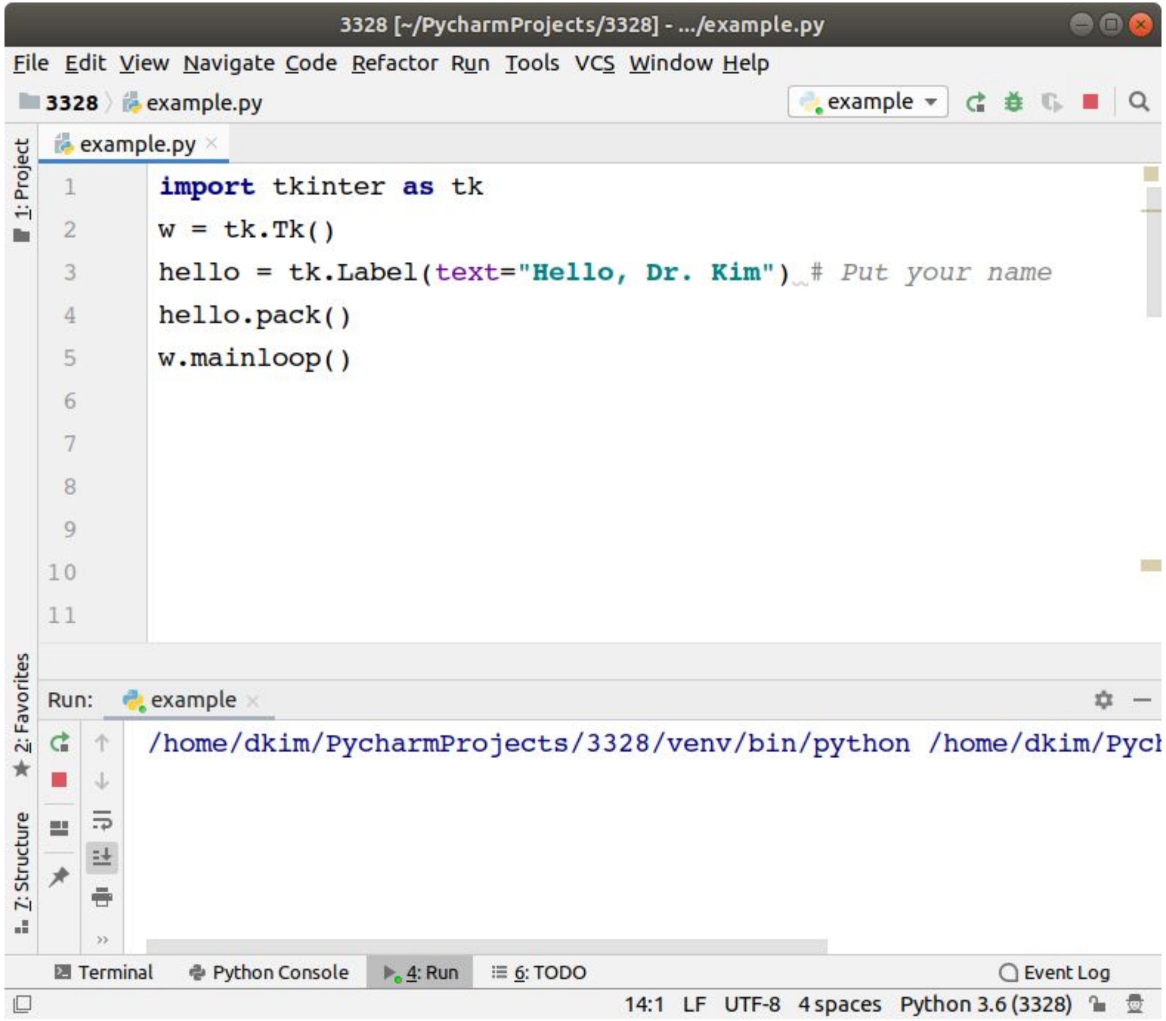
```
hello = tk.Label(text="Hello, Dr. Kim") # put your name
```

Label.pack()

We just created a Label widget, but we haven’t added it to the window yet.

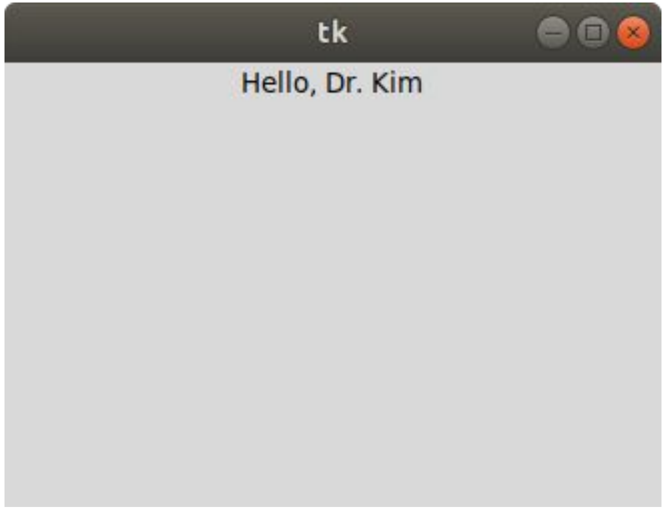
You can use the Label widget’s pack() method to add.

```
hello.pack()
```


3328 [~/PycharmProjects/3328] - .../example.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
3328 example.py
example
Run: example
Terminal Python Console 4: Run 6: TODO Event Log
14:1 LF UTF-8 4 spaces Python 3.6 (3328)

```python
import tkinter as tk
w = tk.Tk()
hello = tk.Label(text="Hello, Dr. Kim") # Put your name
hello.pack()
w.mainloop()
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python /home/dkim/Pych
```



```
Hello, Dr. Kim
```

Window size

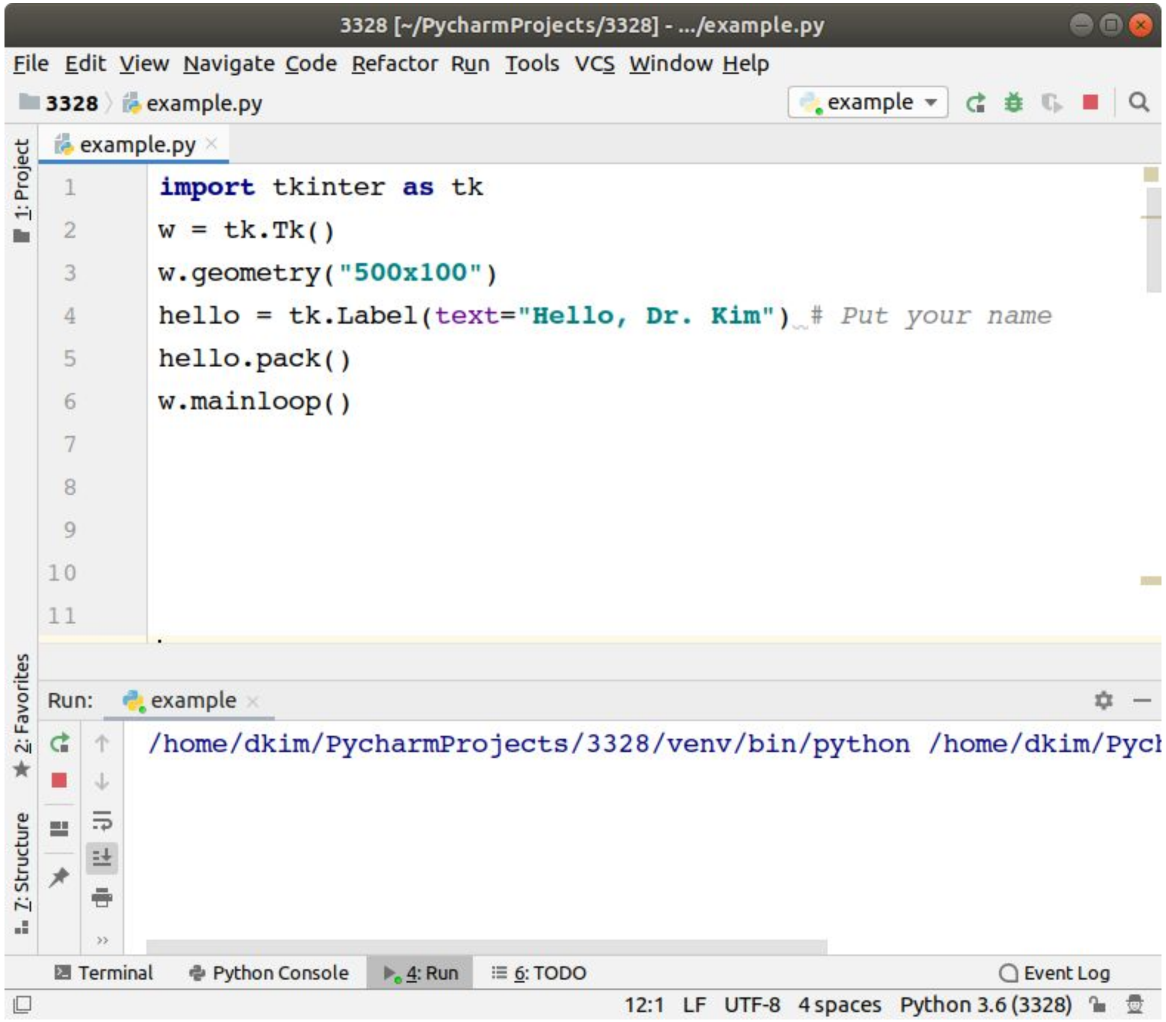
We can use the geometry method of the window object to set a size of the window.

We set the Width to 500 pixels and the Height to 100 pixels as its arguments.

Note that we are using a lowercase “x” here instead of a “*” to essentially say: I want the window to be 500 pixels by (x) 100 pixels.

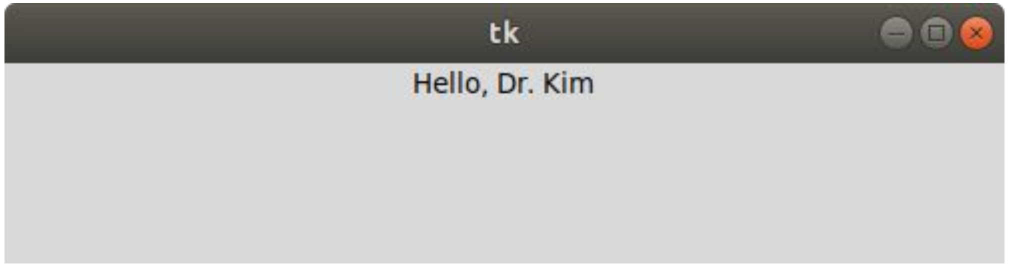
For example,

```
w.geometry("500x100")
```


3328 [~/PycharmProjects/3328] - .../example.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
3328 example.py
example
example.py
1: Project
2: Favorites
7: Structure
Run: example
Terminal Python Console 4: Run 6: TODO Event Log
12:1 LF UTF-8 4 spaces Python 3.6 (3328)

```python
import tkinter as tk
w = tk.Tk()
w.geometry("500x100")
hello = tk.Label(text="Hello, Dr. Kim") # Put your name
hello.pack()
w.mainloop()
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python /home/dkim/Pych
```



```
Hello, Dr. Kim
```

Initial Window Position

When the first Tkinter window is run, it will usually appear in the top left-hand corner by default.

To change this we can add the height and width position to the geometry method.

When Tkinter positions a window it references the top left corner of the window.

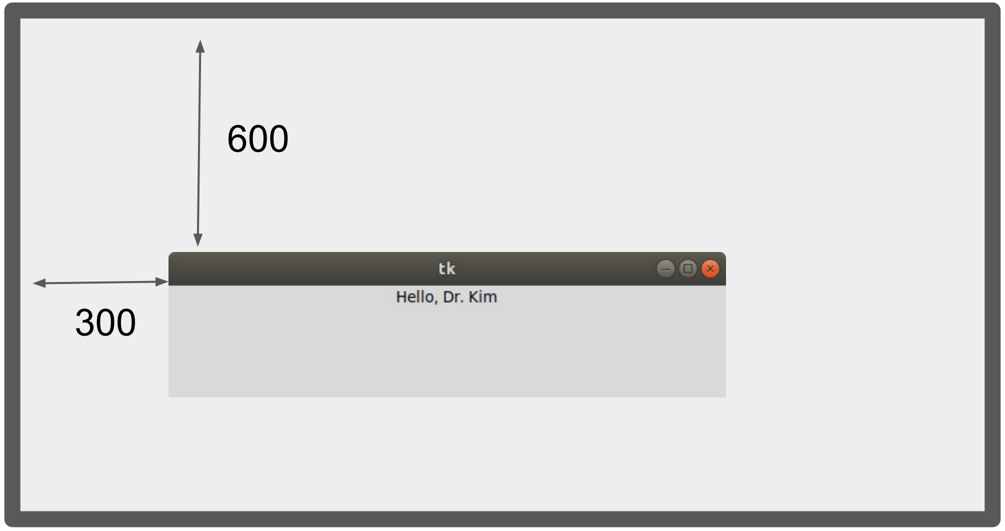
```
.geometry("window width x window height + position right + position down")
```

Note the "+" symbol before each position.

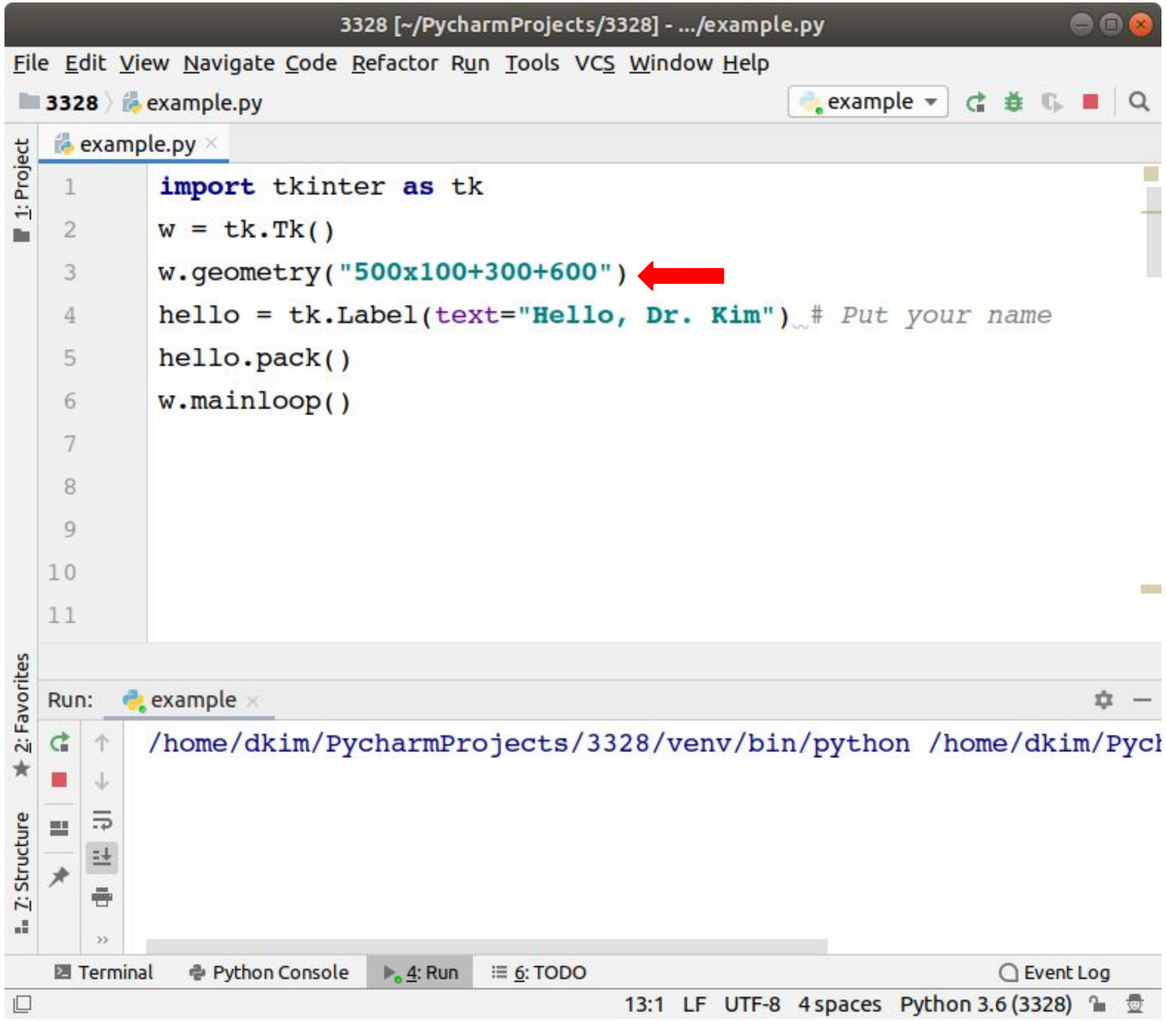
Initial Window Position

```
w.geometry("500x100+300+600")
```

Here, we position the top left corner of the window right 300 pixels and down 600 pixels.



Screen


3328 [~/PycharmProjects/3328] - .../example.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
3328
example.py
example
1: Project
2: Favorites
7: Structure
Run:
Terminal
Python Console
4: Run
6: TODO
Event Log
13:1 LF UTF-8 4 spaces Python 3.6 (3328)

```python
import tkinter as tk
w = tk.Tk()
w.geometry("500x100+300+600")
hello = tk.Label(text="Hello, Dr. Kim") # Put your name
hello.pack()
w.mainloop()
```

```
/home/dkim/PycharmProjects/3328/venv/bin/python /home/dkim/Pyc
```

Font style and size

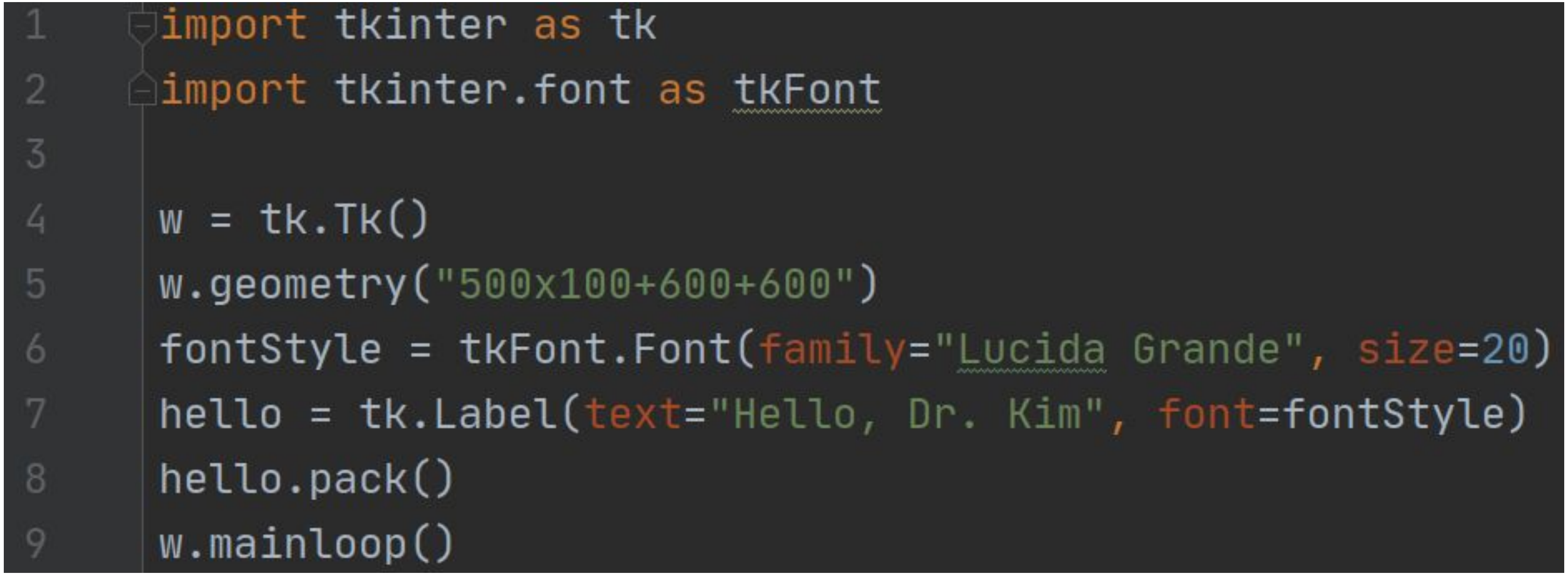
To change the font style and size, we use `tkinter.font` module and its `Font` class.

```
import tkinter.font as tkFont

fontStyle = tkFont.Font(family="Lucida Grande", size=20)
```

Then, we put the `Font` instance, `fontStyle` into its initializer as arguments.

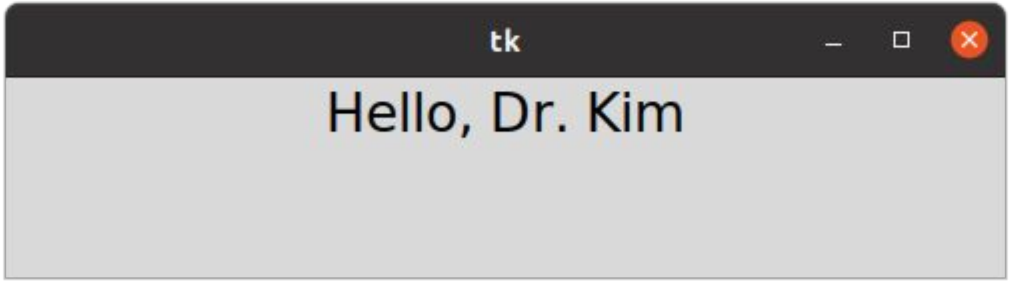
```
label = tk.Label(text="Hello, Dr. Kim!", font=fontStyle)
```

Font style and size



```python
import tkinter as tk
import tkinter.font as tkFont

w = tk.Tk()
w.geometry("500x100+600+600")
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
hello = tk.Label(text="Hello, Dr. Kim", font=fontStyle)
hello.pack()
w.mainloop()
```

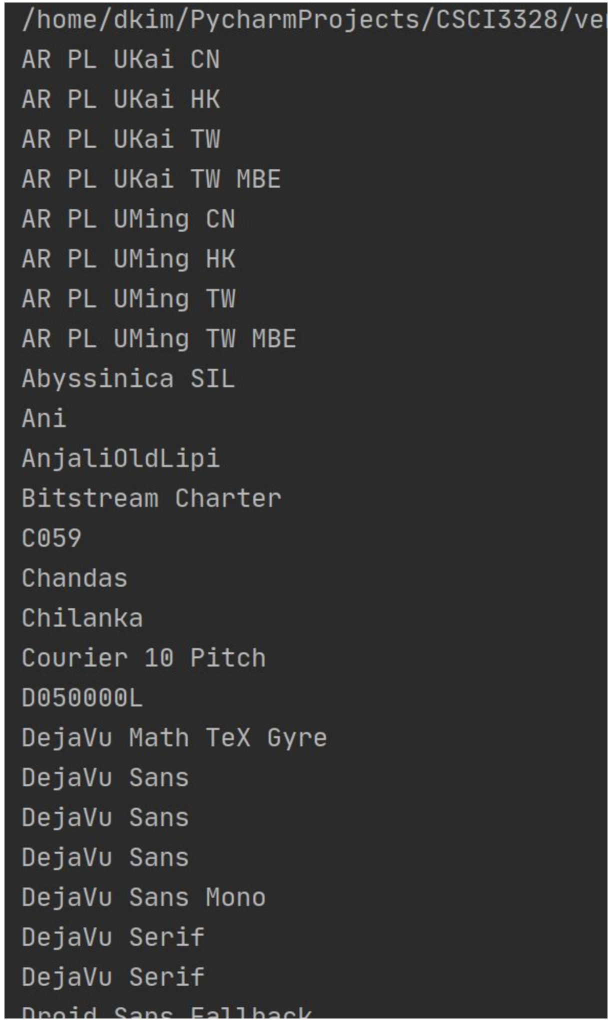
Font style and size



Hello, Dr. Kim

Available Font Family

```
import tkinter as tk

from tkinter import font

root = tk.Tk()

f = list(font.families())

f.sort()

for i in f:

    print(i)
```



```
/home/dkim/PycharmProjects/CSCI3328/ve
AR PL UKai CN
AR PL UKai HK
AR PL UKai TW
AR PL UKai TW MBE
AR PL UMing CN
AR PL UMing HK
AR PL UMing TW
AR PL UMing TW MBE
Abyssinica SIL
Ani
AnjaliOldLipi
Bitstream Charter
C059
Chandas
Chilanka
Courier 10 Pitch
D050000L
DejaVu Math TeX Gyre
DejaVu Sans
DejaVu Sans
DejaVu Sans
DejaVu Sans Mono
DejaVu Serif
DejaVu Serif
Droid Sans Fallback
```

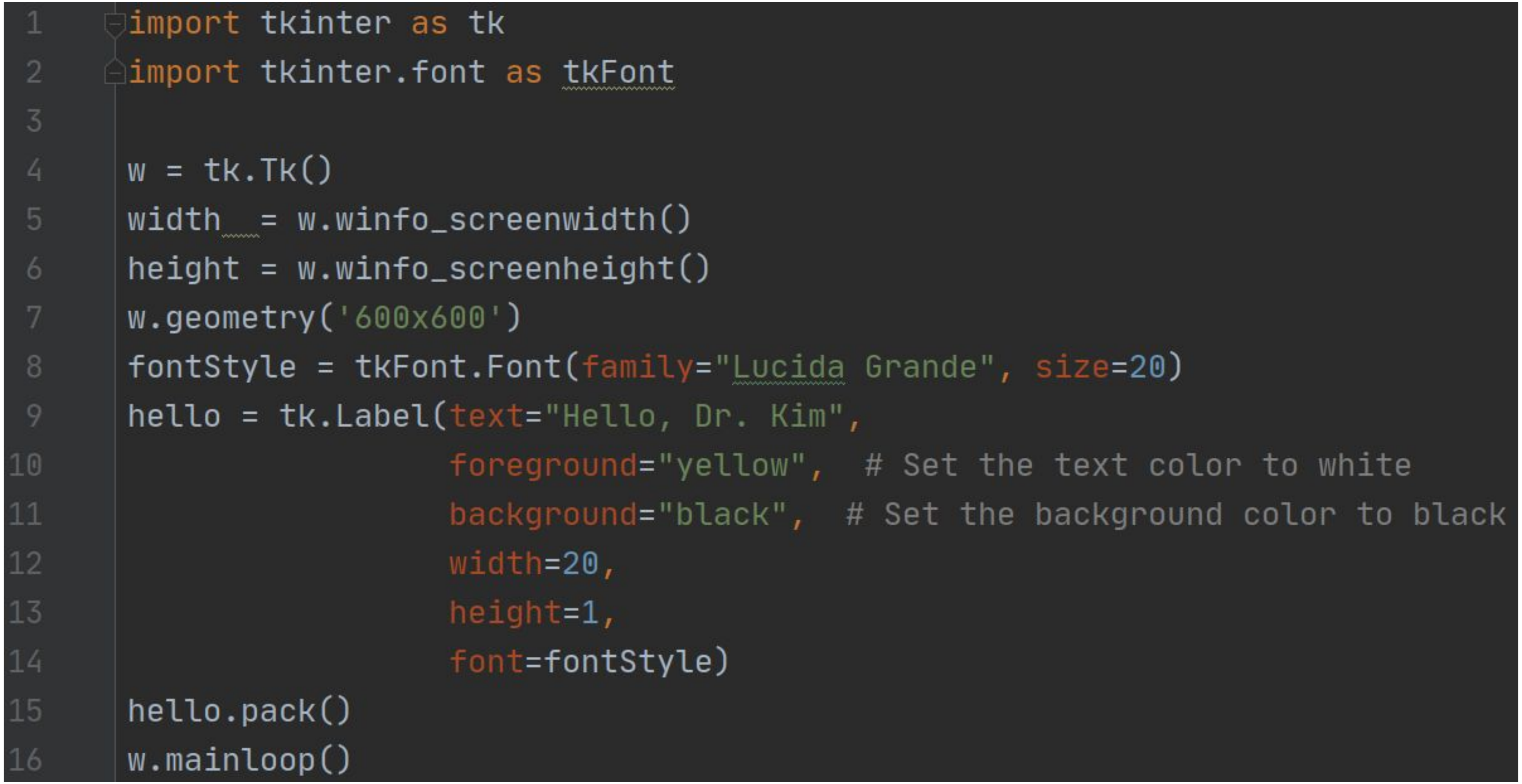
Label()

Tk class (window) has the Label method to display a text.

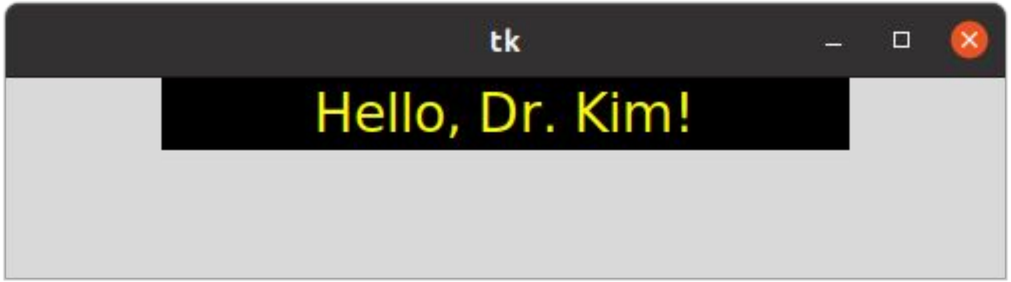
You can set the text color, background, and size of the label as arguments.

For example,

```
label = tk.Label(
    text="Hello, Dr. Kim!",
    foreground="yellow",  # Set the text color to white
    background="black",  # Set the background color to black
    width=20,
    height=1
)
```



```python
import tkinter as tk
import tkinter.font as tkFont

w = tk.Tk()
width  = w.winfo_screenwidth()
height = w.winfo_screenheight()
w.geometry('600x600')
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
hello = tk.Label(text="Hello, Dr. Kim",
                 foreground="yellow",  # Set the text color to white
                 background="black",  # Set the background color to black
                 width=20,
                 height=1,
                 font=fontStyle)
hello.pack()
w.mainloop()
```



```
Hello, Dr. Kim!
```

Colors

Here are numerous valid color names, including:

"red"

"orange"

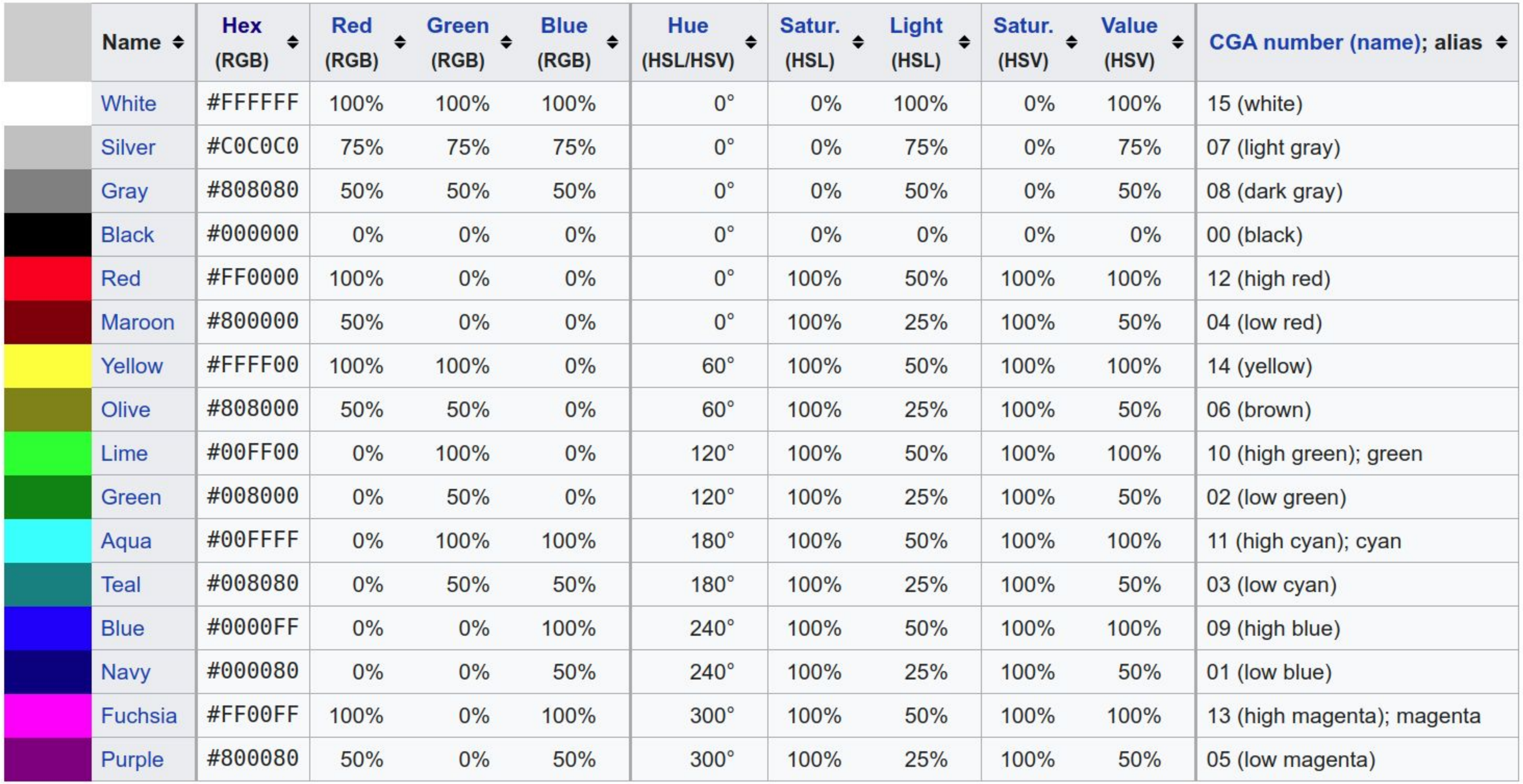
"yellow"

"green"

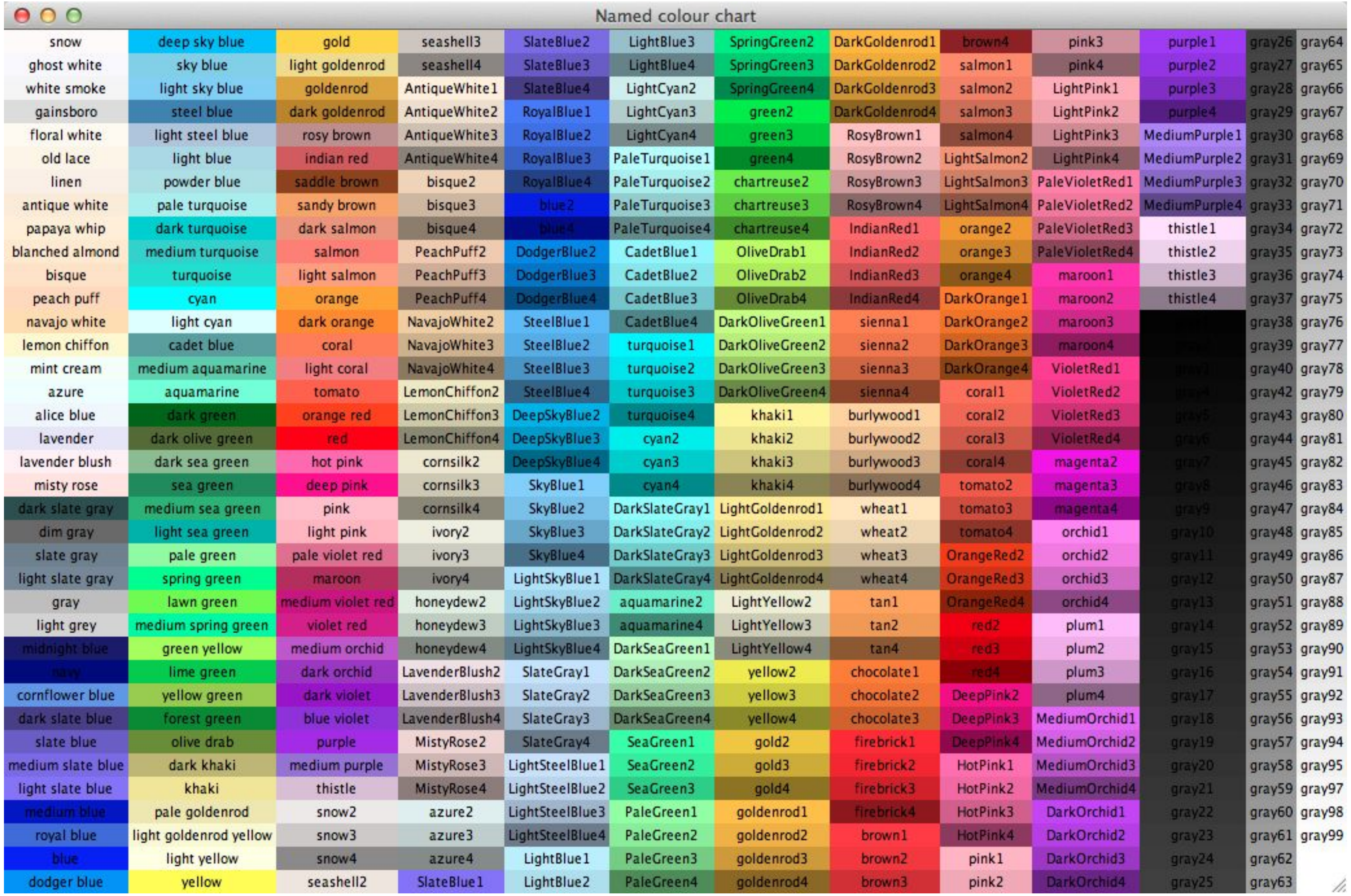
"blue"

"purple"

CSS 1–2.0, HTML 3.2–4, and VGA color names



	Name	Hex (RGB)	Red (RGB)	Green (RGB)	Blue (RGB)	Hue (HSL/HSV)	Satur. (HSL)	Light (HSL)	Satur. (HSV)	Value (HSV)	CGA number (name); alias
⬜	White	#FFFFFF	100%	100%	100%	0°	0%	100%	0%	100%	15 (white)
⬜	Silver	#C0C0C0	75%	75%	75%	0°	0%	75%	0%	75%	07 (light gray)
⬛	Gray	#808080	50%	50%	50%	0°	0%	50%	0%	50%	08 (dark gray)
⬛	Black	#000000	0%	0%	0%	0°	0%	0%	0%	0%	00 (black)
🟥	Red	#FF0000	100%	0%	0%	0°	100%	50%	100%	100%	12 (high red)
🟫	Maroon	#800000	50%	0%	0%	0°	100%	25%	100%	50%	04 (low red)
🟨	Yellow	#FFFF00	100%	100%	0%	60°	100%	50%	100%	100%	14 (yellow)
🟫	Olive	#808000	50%	50%	0%	60°	100%	25%	100%	50%	06 (brown)
🟩	Lime	#00FF00	0%	100%	0%	120°	100%	50%	100%	100%	10 (high green); green
🟩	Green	#008000	0%	50%	0%	120°	100%	25%	100%	50%	02 (low green)
🟦	Aqua	#00FFFF	0%	100%	100%	180°	100%	50%	100%	100%	11 (high cyan); cyan
🟦	Teal	#008080	0%	50%	50%	180°	100%	25%	100%	50%	03 (low cyan)
🟦	Blue	#0000FF	0%	0%	100%	240°	100%	50%	100%	100%	09 (high blue)
🟦	Navy	#000080	0%	0%	50%	240°	100%	25%	100%	50%	01 (low blue)
🟪	Fuchsia	#FF00FF	100%	0%	100%	300°	100%	50%	100%	100%	13 (high magenta); magenta
🟪	Purple	#800080	50%	0%	50%	300°	100%	25%	100%	50%	05 (low magenta)



Named colour chart

snow	deep sky blue	gold	seashell3	SlateBlue2	LightBlue3	SpringGreen2	DarkGoldenrod1	brown4	pink3	purple1	gray26	gray64
ghost white	sky blue	light goldenrod	seashell4	SlateBlue3	LightBlue4	SpringGreen3	DarkGoldenrod2	salmon1	pink4	purple2	gray27	gray65
white smoke	light sky blue	goldenrod	AntiqueWhite1	SlateBlue4	LightCyan2	SpringGreen4	DarkGoldenrod3	salmon2	LightPink1	purple3	gray28	gray66
gainsboro	steel blue	dark goldenrod	AntiqueWhite2	RoyalBlue1	LightCyan3	green2	DarkGoldenrod4	salmon3	LightPink2	purple4	gray29	gray67
floral white	light steel blue	rosy brown	AntiqueWhite3	RoyalBlue2	LightCyan4	green3	RosyBrown1	salmon4	LightPink3	MediumPurple1	gray30	gray68
old lace	light blue	indian red	AntiqueWhite4	RoyalBlue3	PaleTurquoise1	green4	RosyBrown2	LightSalmon2	LightPink4	MediumPurple2	gray31	gray69
linen	powder blue	saddle brown	bisque2	RoyalBlue4	PaleTurquoise2	chartreuse2	RosyBrown3	LightSalmon3	PaleVioletRed1	MediumPurple3	gray32	gray70
antique white	pale turquoise	sandy brown	bisque3	blue2	PaleTurquoise3	chartreuse3	RosyBrown4	LightSalmon4	PaleVioletRed2	MediumPurple4	gray33	gray71
papaya whip	dark turquoise	dark salmon	bisque4	blue4	PaleTurquoise4	chartreuse4	IndianRed1	orange2	PaleVioletRed3	thistle1	gray34	gray72
blanched almond	medium turquoise	salmon	PeachPuff2	DodgerBlue2	CadetBlue1	OliveDrab1	IndianRed2	orange3	PaleVioletRed4	thistle2	gray35	gray73
bisque	turquoise	light salmon	PeachPuff3	DodgerBlue3	CadetBlue2	OliveDrab2	IndianRed3	orange4	maroon1	thistle3	gray36	gray74
peach puff	cyan	orange	PeachPuff4	DodgerBlue4	CadetBlue3	OliveDrab4	IndianRed4	DarkOrange1	maroon2	thistle4	gray37	gray75
navajo white	light cyan	dark orange	NavajoWhite2	SteelBlue1	CadetBlue4	DarkOliveGreen1	sienna1	DarkOrange2	maroon3	[illegible]	gray38	gray76
lemon chiffon	cadet blue	coral	NavajoWhite3	SteelBlue2	turquoise1	DarkOliveGreen2	sienna2	DarkOrange3	maroon4	gray2	gray39	gray77
mint cream	medium aquamarine	light coral	NavajoWhite4	SteelBlue3	turquoise2	DarkOliveGreen3	sienna3	DarkOrange4	VioletRed1	gray3	gray40	gray78
azure	aquamarine	tomato	LemonChiffon2	SteelBlue4	turquoise3	DarkOliveGreen4	sienna4	coral1	VioletRed2	gray4	gray42	gray79
alice blue	dark green	orange red	LemonChiffon3	DeepSkyBlue2	turquoise4	khaki1	burlywood1	coral2	VioletRed3	gray5	gray43	gray80
lavender	dark olive green	red	LemonChiffon4	DeepSkyBlue3	cyan2	khaki2	burlywood2	coral3	VioletRed4	gray6	gray44	gray81
lavender blush	dark sea green	hot pink	cornsilk2	DeepSkyBlue4	cyan3	khaki3	burlywood3	coral4	magenta2	gray7	gray45	gray82
misty rose	sea green	deep pink	cornsilk3	SkyBlue1	cyan4	khaki4	burlywood4	tomato2	magenta3	gray8	gray46	gray83
dark slate gray	medium sea green	pink	cornsilk4	SkyBlue2	DarkSlateGray1	LightGoldenrod1	wheat1	tomato3	magenta4	gray9	gray47	gray84
dim gray	light sea green	light pink	ivory2	SkyBlue3	DarkSlateGray2	LightGoldenrod2	wheat2	tomato4	orchid1	gray10	gray48	gray85
slate gray	pale green	pale violet red	ivory3	SkyBlue4	DarkSlateGray3	LightGoldenrod3	wheat3	OrangeRed2	orchid2	gray11	gray49	gray86
light slate gray	spring green	maroon	ivory4	LightSkyBlue1	DarkSlateGray4	LightGoldenrod4	wheat4	OrangeRed3	orchid3	gray12	gray50	gray87
gray	lawn green	medium violet red	honeydew2	LightSkyBlue2	aquamarine2	LightYellow2	tan1	OrangeRed4	orchid4	gray13	gray51	gray88
light grey	medium spring green	violet red	honeydew3	LightSkyBlue3	aquamarine4	LightYellow3	tan2	red2	plum1	gray14	gray52	gray89
midnight blue	green yellow	medium orchid	honeydew4	LightSkyBlue4	DarkSeaGreen1	LightYellow4	tan4	red3	plum2	gray15	gray53	gray90
navy	lime green	dark orchid	LavenderBlush2	SlateGray1	DarkSeaGreen2	yellow2	chocolate1	red4	plum3	gray16	gray54	gray91
cornflower blue	yellow green	dark violet	LavenderBlush3	SlateGray2	DarkSeaGreen3	yellow3	chocolate2	DeepPink2	plum4	gray17	gray55	gray92
dark slate blue	forest green	blue violet	LavenderBlush4	SlateGray3	DarkSeaGreen4	yellow4	chocolate3	DeepPink3	MediumOrchid1	gray18	gray56	gray93
slate blue	olive drab	purple	MistyRose2	SlateGray4	SeaGreen1	gold2	firebrick1	DeepPink4	MediumOrchid2	gray19	gray57	gray94
medium slate blue	dark khaki	medium purple	MistyRose3	LightSteelBlue1	SeaGreen2	gold3	firebrick2	HotPink1	MediumOrchid3	gray20	gray58	gray95
light slate blue	khaki	thistle	MistyRose4	LightSteelBlue2	SeaGreen3	gold4	firebrick3	HotPink2	MediumOrchid4	gray21	gray59	gray97
medium blue	pale goldenrod	snow2	azure2	LightSteelBlue3	PaleGreen1	goldenrod1	firebrick4	HotPink3	DarkOrchid1	gray22	gray60	gray98
royal blue	light goldenrod yellow	snow3	azure3	LightSteelBlue4	PaleGreen2	goldenrod2	brown1	HotPink4	DarkOrchid2	gray23	gray61	gray99
blue	light yellow	snow4	azure4	LightBlue1	PaleGreen3	goldenrod3	brown2	pink1	DarkOrchid3	gray24	gray62	
dodger blue	yellow	seashell2	SlateBlue1	LightBlue2	PaleGreen4	goldenrod4	brown3	pink2	DarkOrchid4	gray25	gray63	

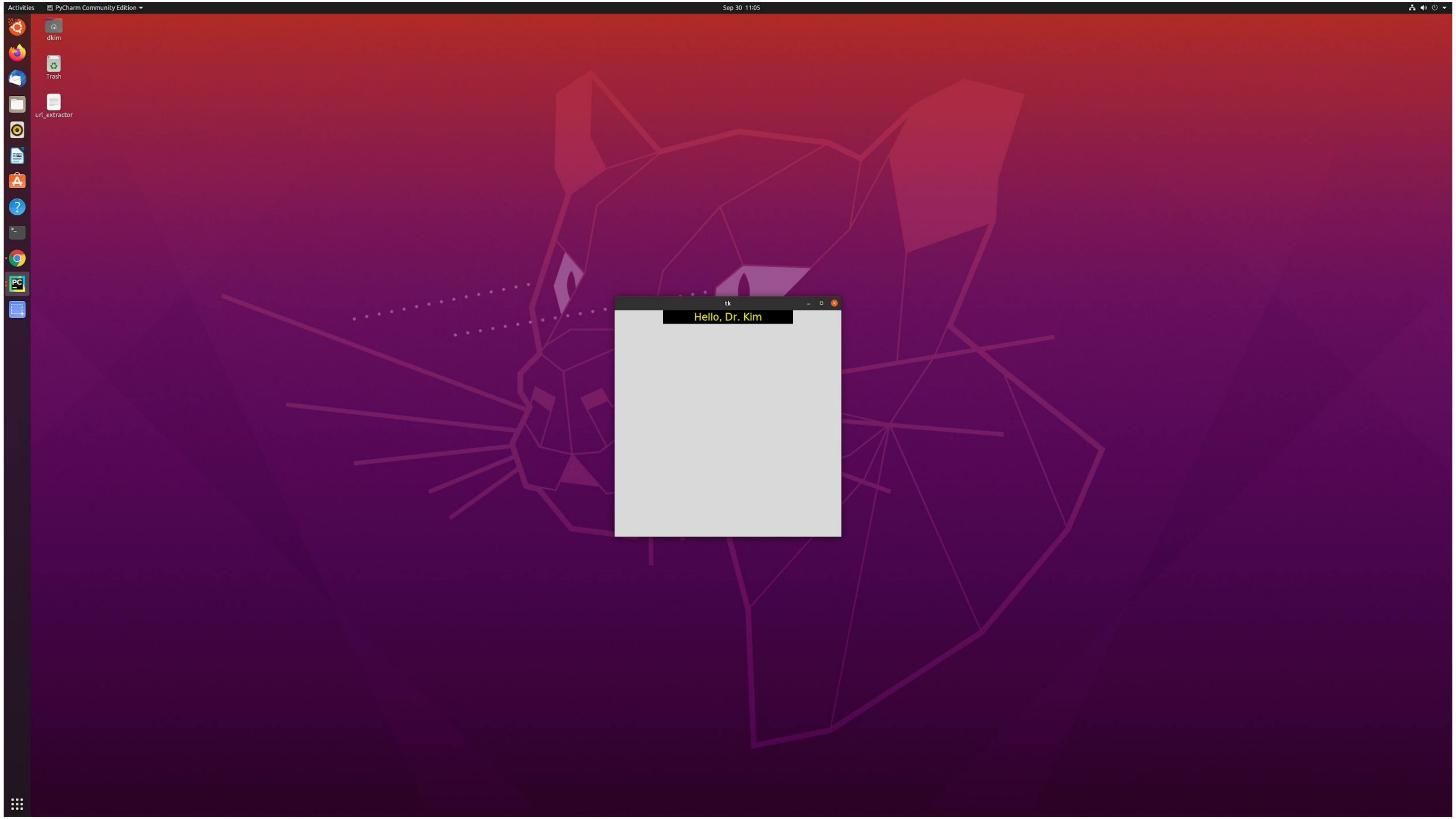
Lab 22-1

Make a Python GUI program that displays a window (size: 600 by 600) **on the center of the screen**. Using a Label Widget, display your name with different colors (any colors) for text and background.

(Hint)

```
width  = w.winfo_screenwidth()
```

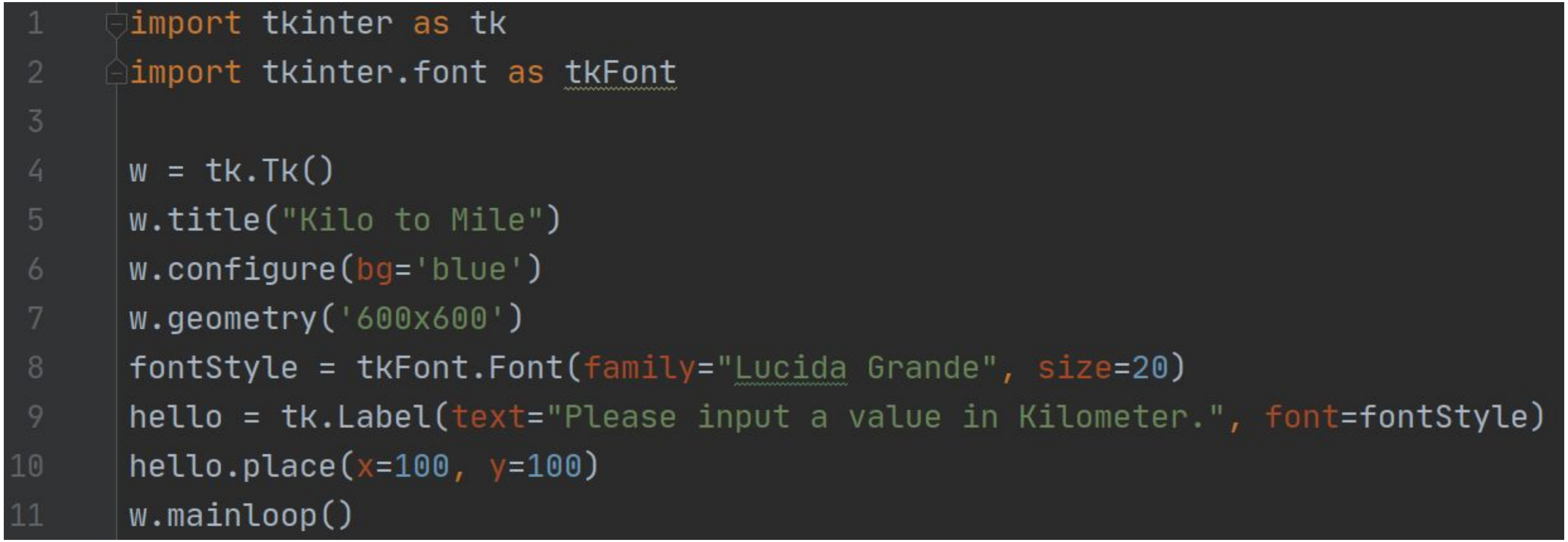
```
height = w.winfo_screenheight()
```

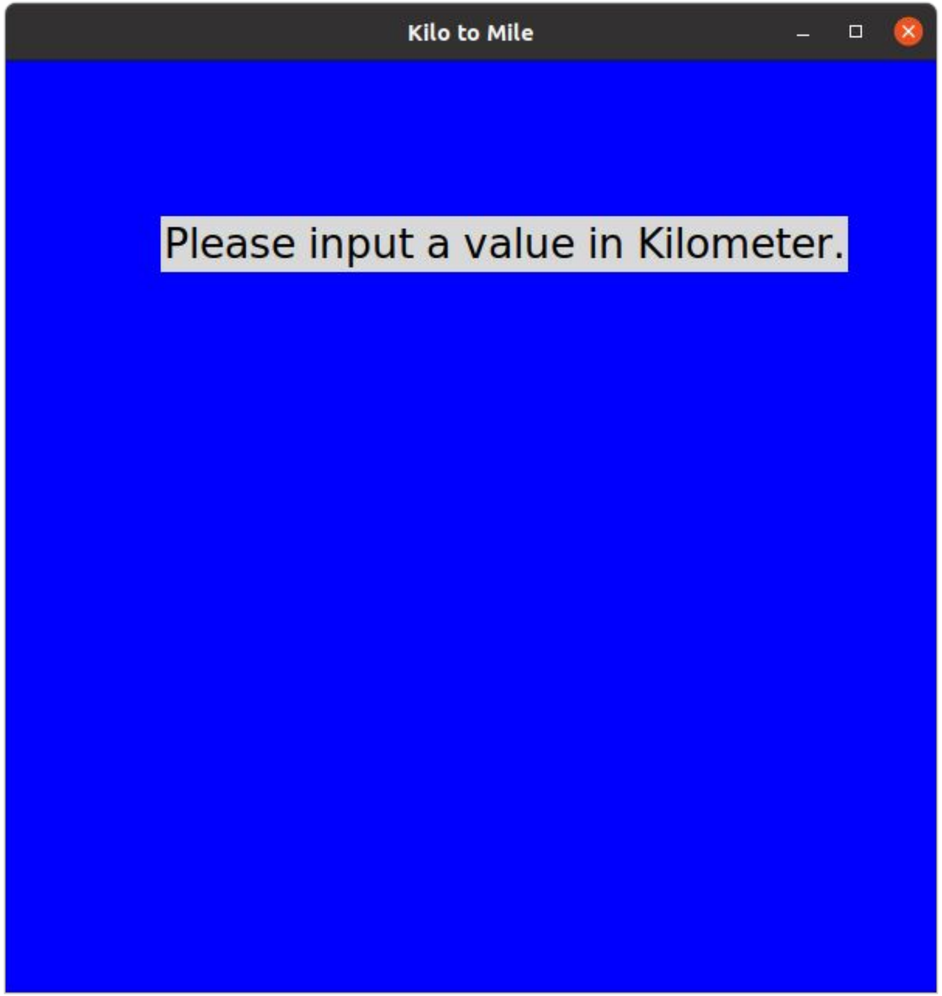

Activities
PyCharm Community Edition
Sep 30 11:05
dkim
Trash
url_extractor
tk

Hello, Dr. Kim

place()

You can use .place() to control the precise location that a widget should occupy in a window. You must provide two keyword arguments, x and y, which specify the x- and y-coordinates for the top-left corner of the widget. Both x and y are measured in pixels, not text units.


```python
import tkinter as tk
import tkinter.font as tkFont

w = tk.Tk()
w.title("Kilo to Mile")
w.configure(bg='blue')
w.geometry('600x600')
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
hello = tk.Label(text="Please input a value in Kilometer.", font=fontStyle)
hello.place(x=100, y=100)
w.mainloop()
```



Please input a value in Kilometer.

Entry

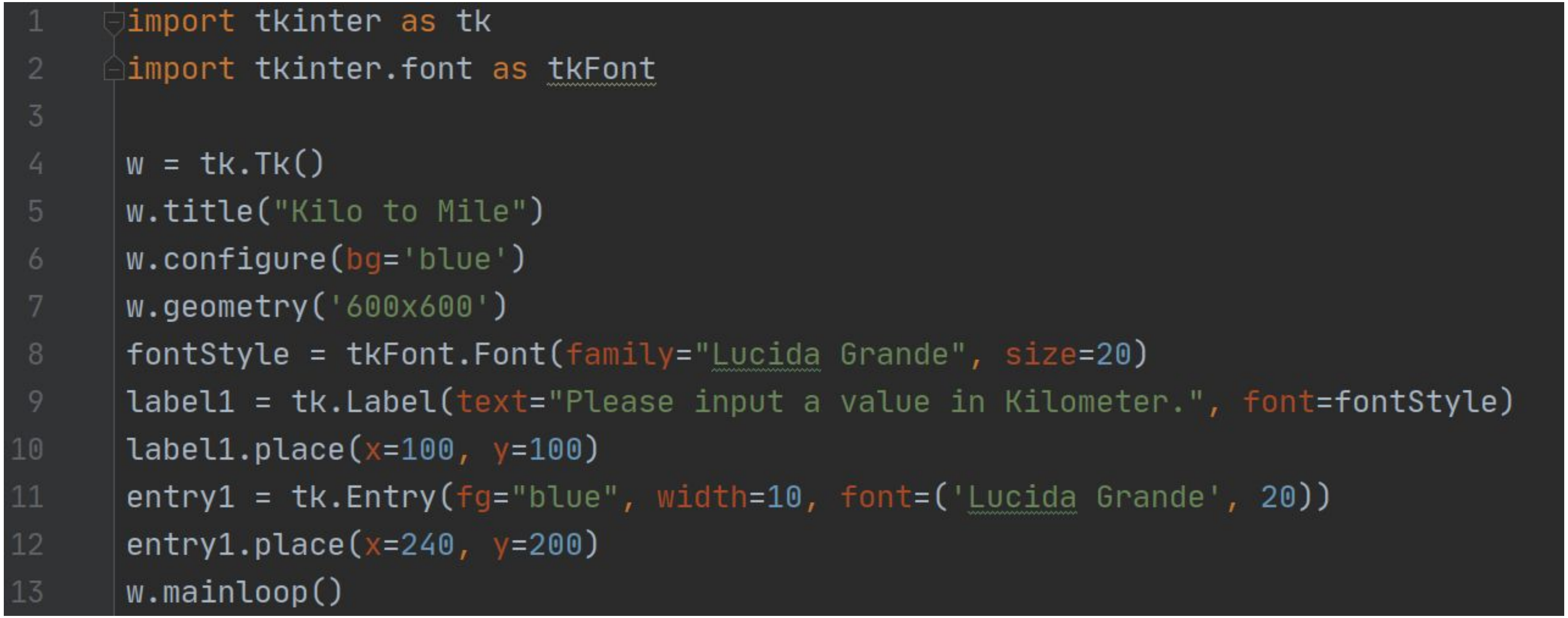
When you need to get a little bit of text from a user, like a name or an email address, use an Entry widget. They display a small text box that the user can type some text into. Creating and styling an Entry widget works pretty much exactly like Label and Button widgets. For example,

```
entry = tk.Entry(fg="yellow", bg="blue", width=50)
```

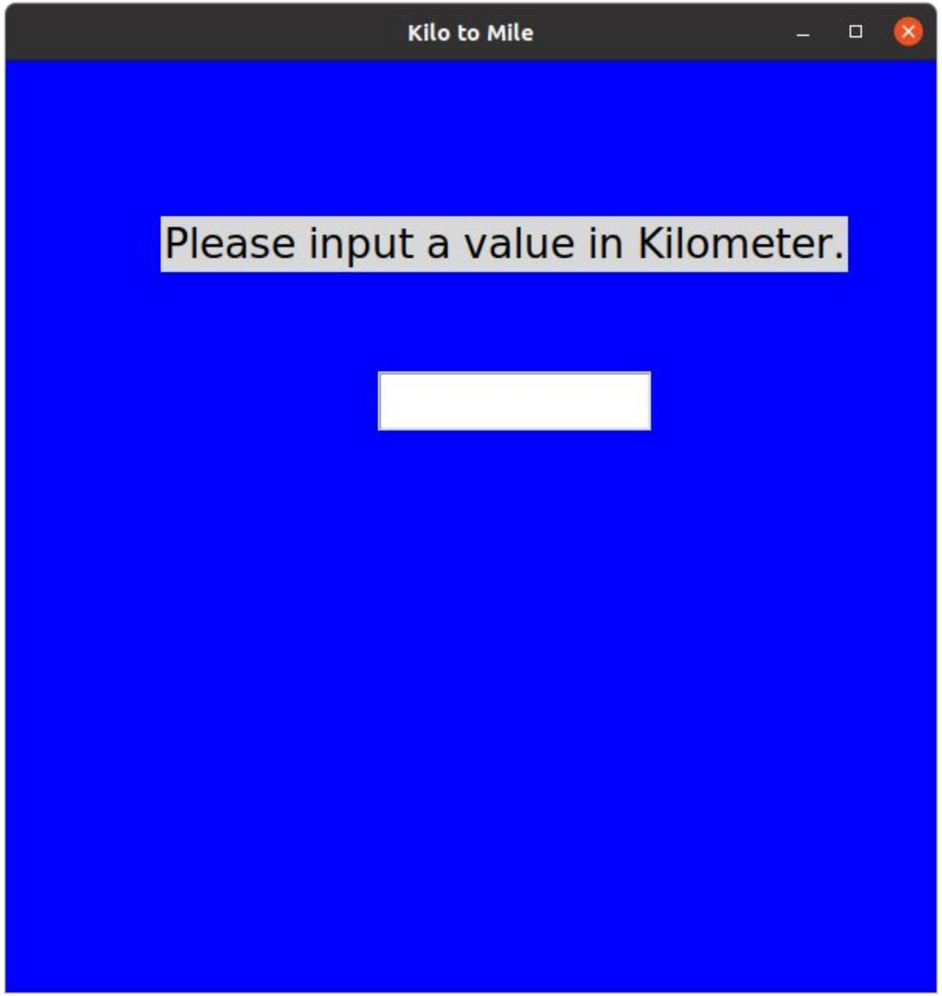
You can use .get() to retrieve the text and assign it to a variable.

```
str = entry.get()
```

Entry



```python
import tkinter as tk
import tkinter.font as tkFont

w = tk.Tk()
w.title("Kilo to Mile")
w.configure(bg='blue')
w.geometry('600x600')
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
label1 = tk.Label(text="Please input a value in Kilometer.", font=fontStyle)
label1.place(x=100, y=100)
entry1 = tk.Entry(fg="blue", width=10, font=('Lucida Grande', 20))
entry1.place(x=240, y=200)
w.mainloop()
```

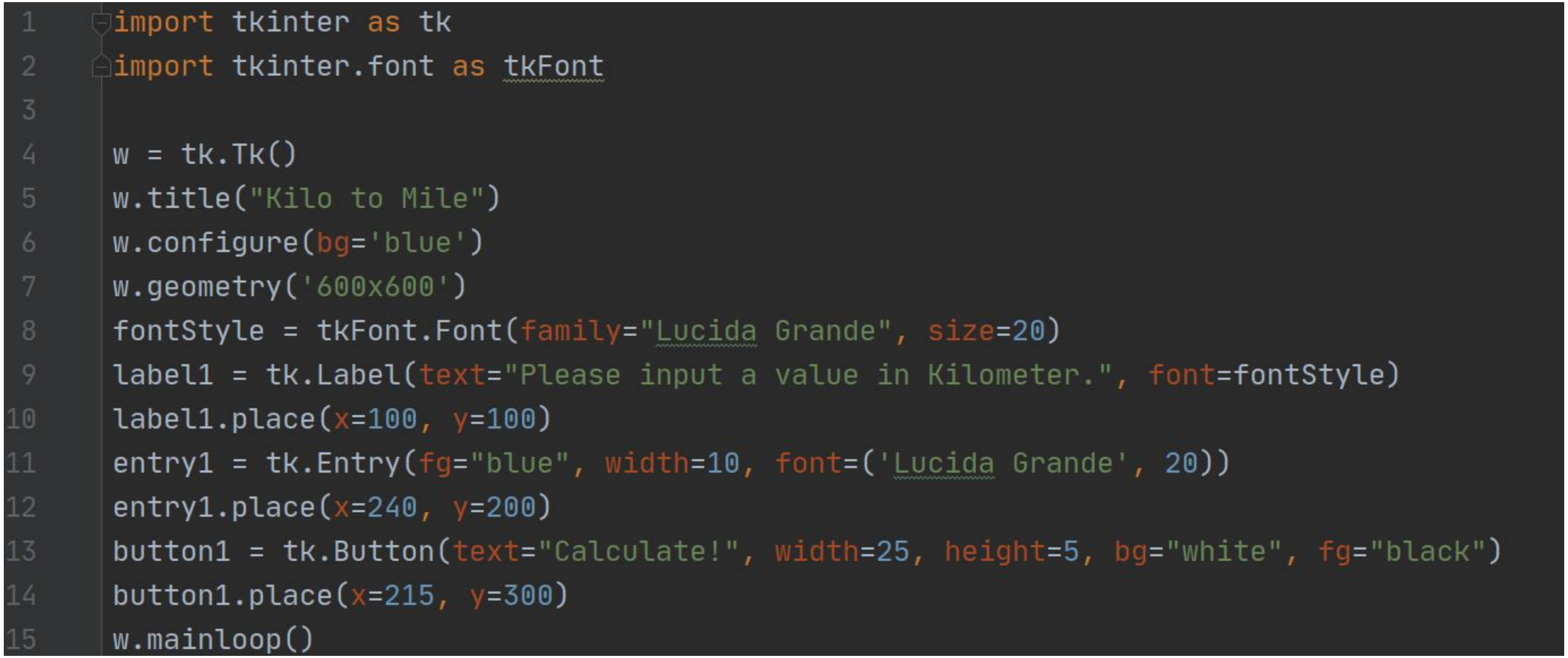


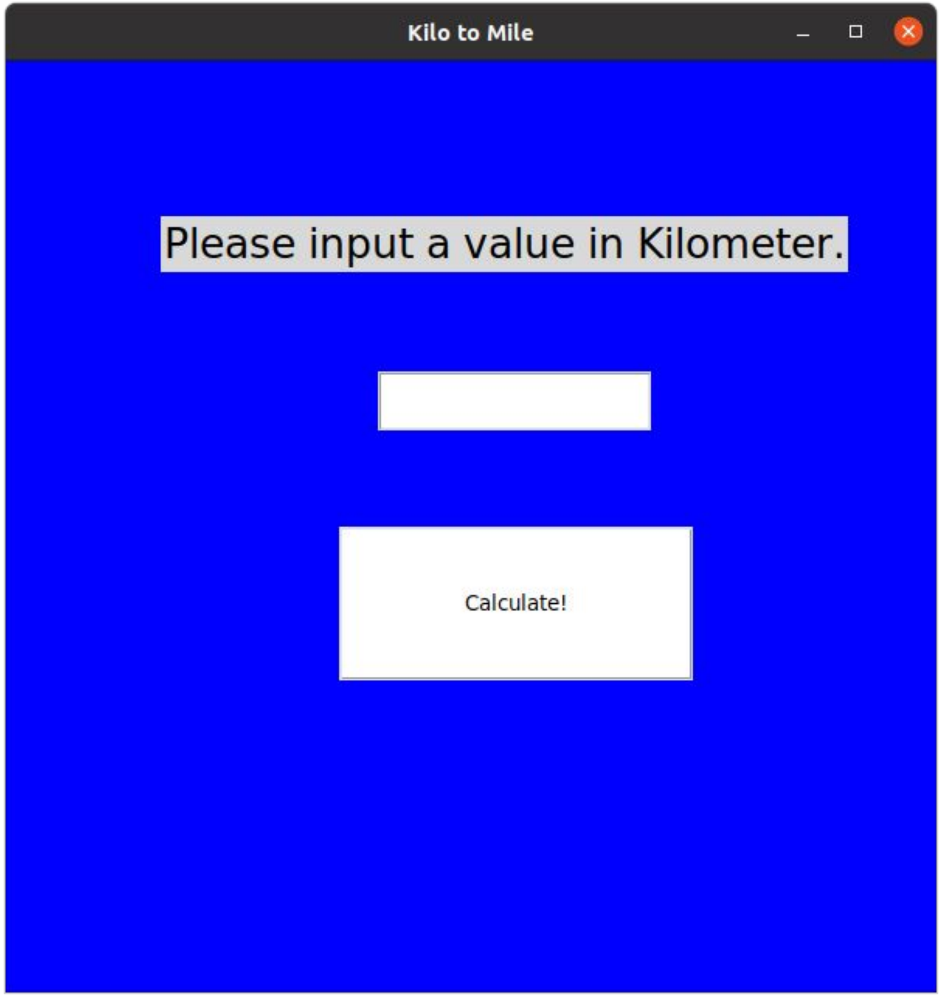
Please input a value in Kilometer.

button

Button widgets are used to display clickable buttons. They can be configured to call a function whenever they’re clicked.

```
button = tk.Button(

    text="Calculate!",

    width=25,

    height=5,

    bg="blue",

    fg="yellow")
```


```python
import tkinter as tk
import tkinter.font as tkFont

w = tk.Tk()
w.title("Kilo to Mile")
w.configure(bg='blue')
w.geometry('600x600')
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
label1 = tk.Label(text="Please input a value in Kilometer.", font=fontStyle)
label1.place(x=100, y=100)
entry1 = tk.Entry(fg="blue", width=10, font=('Lucida Grande', 20))
entry1.place(x=240, y=200)
button1 = tk.Button(text="Calculate!", width=25, height=5, bg="white", fg="black")
button1.place(x=215, y=300)
w.mainloop()
```


Kilo to Mile
Please input a value in Kilometer.
Calculate!

bind

To call an event handler whenever an event occurs on a widget, use **.bind()**. The event handler is said to be bound to the event because it’s called every time the event occurs.

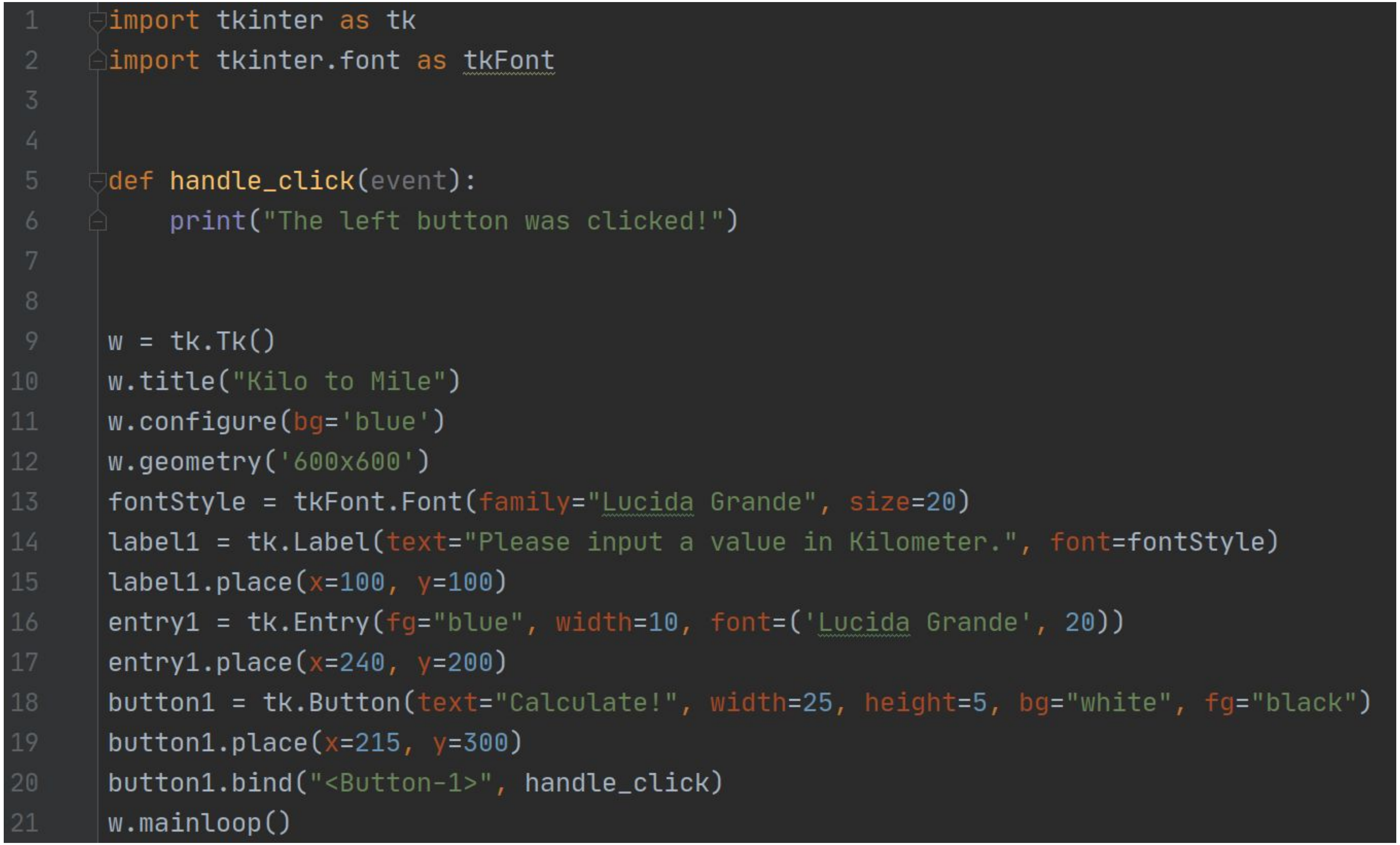
.bind() always takes at least two arguments:

1. An **event** that’s represented by a string of the form "<event_name>", where event_name can be any of Tkinter’s events
2. An **event handler** that’s the name of the function to be called whenever the event occurs

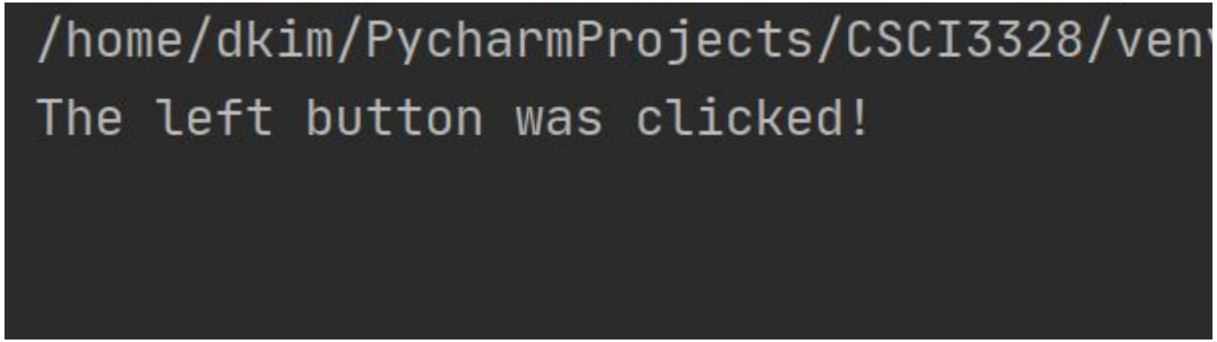
bind

```
def handle_click(event):

    print("The button was clicked!")

button = tk.Button(text="Click me!")

button.bind("<Button-1>", handle_click)
```

In this example, the "<Button-1>" event on the button widget is bound to the **handle_click** event handler. The "**<Button-1>**" event occurs whenever the **left mouse button** is pressed while the mouse is over the widget. There are other events for mouse button clicks, including "<Button-2>" for the middle mouse button and "<Button-3>" for the right mouse button.



```python
import tkinter as tk
import tkinter.font as tkFont


def handle_click(event):
    print("The left button was clicked!")


w = tk.Tk()
w.title("Kilo to Mile")
w.configure(bg='blue')
w.geometry('600x600')
fontStyle = tkFont.Font(family="Lucida Grande", size=20)
label1 = tk.Label(text="Please input a value in Kilometer.", font=fontStyle)
label1.place(x=100, y=100)
entry1 = tk.Entry(fg="blue", width=10, font=('Lucida Grande', 20))
entry1.place(x=240, y=200)
button1 = tk.Button(text="Calculate!", width=25, height=5, bg="white", fg="black")
button1.place(x=215, y=300)
button1.bind("<Button-1>", handle_click)
w.mainloop()
```

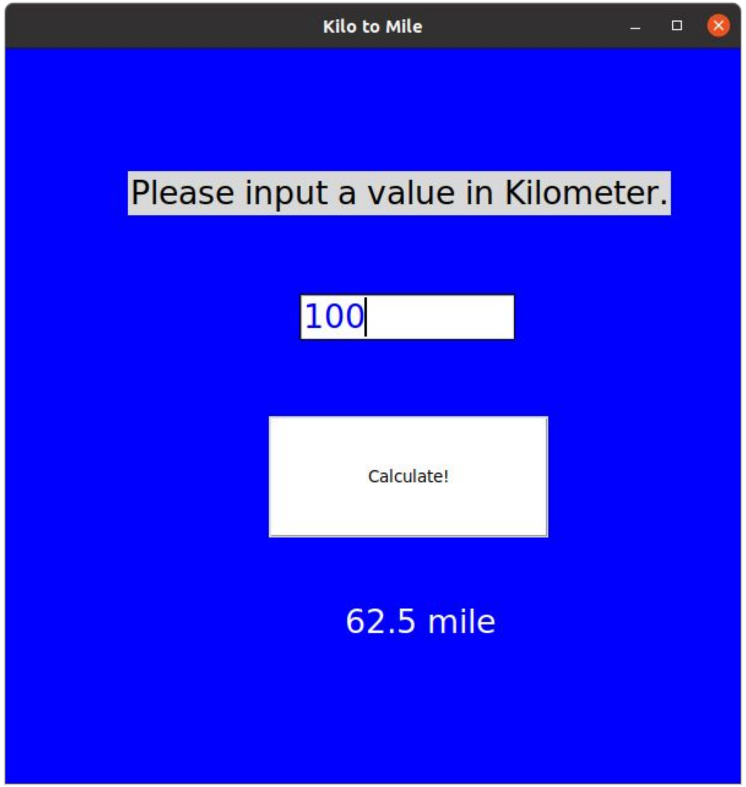


```
/home/dkim/PycharmProjects/CSCI3328/ven
The left button was clicked!
```

Lab 22-2

Make a Python GUI program that converts kilometer to mile.

Example

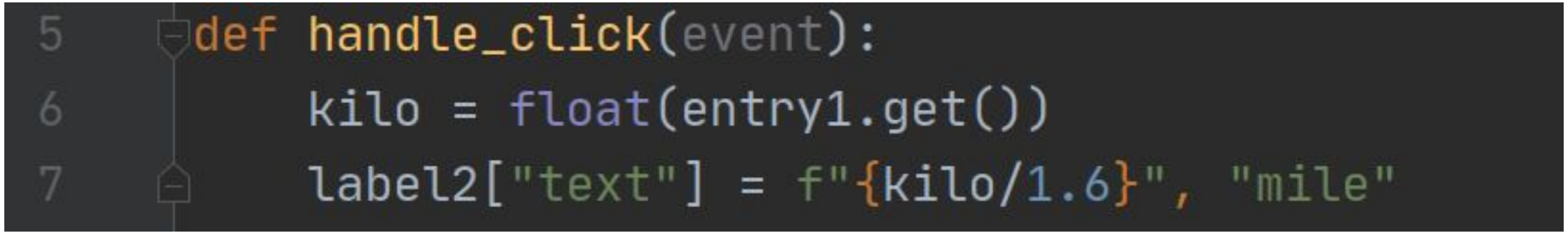


Please input a value in Kilometer.

100

62.5 mile

Hint



```python
def handle_click(event):
    kilo = float(entry1.get())
    label2["text"] = f"{kilo/1.6}", "mile"
```