LECTURE 10

DOHYUNG KIM

WHAT IS DISCUSSED IN THE LAST CLASS

- Multi-dimensional List

TODAY, WE WILL LEARN ABOUT

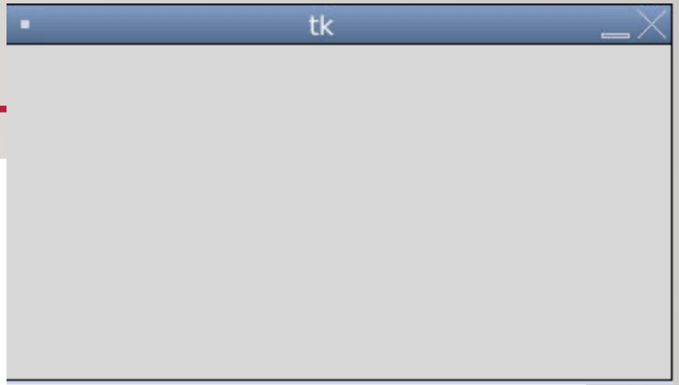
- Graphics using TkInter
- https://repl.it/languages/tkinter

CREATE A TKINTER APP

- Importing the module named Tkinter
- Create the main window (container)
- Add widgets including canvas, frame, button, etc. to the main window
- Apply the event Trigger on the widgets

```
import tkinter
w = tkinter.Tk()
'''
widgets are added here
'''
w.mainloop()
```

CREATE AN EMPTY CANVAS



```
#import module
from tkinter import *

def runCanvas(canvas_width, canvas_height):
  #create a window
  master = Tk()
  #window cannot be resized
  master.resizable(width=False, height=False)
  #create a canvas widget
  canvas = Canvas(master, width=canvas_width, height=canvas_height)
  #pack the canvas widget
  canvas.pack()
  #run the application, wait for an event to occur and process the event
  as long as the window is not closed.
  master.mainloop()

runCanvas(400, 200)
```

DRAW A LINE ON THE CANVAS

```python
from tkinter import *

def draw(canvas, width, height):
  y = height/2
  canvas.create_line(0, y, width, y)

def runCanvas(canvas_width, canvas_height):
  master = Tk()
  master.resizable(width=False, height=False)
  canvas = Canvas(master, width=canvas_width, height=canvas_height)
  canvas.pack()

  draw(canvas, canvas_width, canvas_height)

  master.mainloop()

runCanvas(400, 200)
```

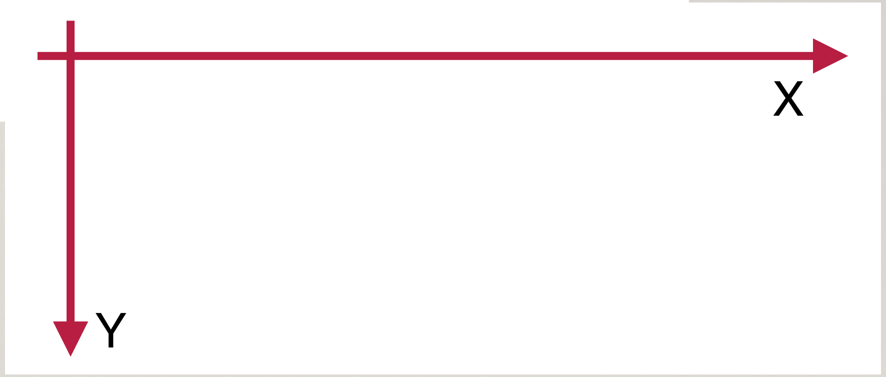
DRAW A RECTANGLE

```
from tkinter import *

def draw(canvas, width, height):
  canvas.create_rectangle(0, 0, width/4, height/4, fill="red")

def runCanvas(canvas_width, canvas_height):
  master = Tk()
  master.resizable(width=False, height=False)
  canvas = Canvas(master, width=canvas_width, height=canvas_height)
  canvas.pack()
  draw(canvas, canvas_width, canvas_height)
  master.mainloop()

runCanvas(400, 200)
```



PRACTICE

- Draw multiple rectangles on the canvas

```
def draw(canvas, width, height):
  canvas.create_rectangle(0, 0, width/4, height/4, fill="red")
  canvas.create_rectangle(100, 50, 150, 200, fill="green", width=3)
  canvas.create_rectangle(50, 10, width, 70, fill="yellow", width=5, outline="blue")
```

- Draw a rectangle in the center of the canvas

COLOR MANIPULATION

- Each color has its r,g,b value (0 ~ 255), e.g.,
  - red: (255, 0, 0), yellow : (255, 255, 0), blue: (0, 0, 255), …
- You can change r,g,b value into rgbString

```
def rgbString(rgbValue):
  return "#%02x%02x%02x" % (rgbValue[0], rgbValue[1], rgbValue[2])
```

- rgbString is used when we draw widget

```
yellowRGB = (255, 255, 0)
color1 = rgbString(yellowRGB)
canvas.create_rectangle(0, 0, width/2, height/2, fill=color1)
```

COLOR MANIPULATION

- If you want to mix two colors

```
def interpolate(t, c1, c2):
  return int((1-t)*c1[0]+t*c2[0]), int((1-t)*c1[1]+t*c2[1]), int((1-t)*c1[2]+t*c2[2])

yellowRGB = (255, 255, 0)
redRGB = 255, 0, 0
mixedRGB = interpolate(0.5, yellowRGB, redRGB)
```

```python
from tkinter import *

def rgbString(rgb):
  return "#%02x%02x%02x" % (rgb[0], rgb[1], rgb[2])

def interpolate(t, c1, c2):
  return int((1-t)*c1[0]+t*c2[0]), int((1-t)*c1[1]+t*c2[1]), int((1-t)*c1[2]+t*c2[2])

def draw(canvas, width, height):
  yellowRGB = (255, 255, 0)
  redRGB = 255, 0, 0
  mixedRGB = interpolate(0.5, yellowRGB, redRGB)

  color1 = rgbString(yellowRGB)
  color2 = rgbString(redRGB)
  mixed = rgbString(mixedRGB)

  yellowRGB = (255, 255, 0)
  color1 = rgbString(yellowRGB)
  canvas.create_rectangle(0, 0, width/2, height/2, fill=color1)
  canvas.create_rectangle(width/2, height/2, width, height, fill=color2)
  canvas.create_rectangle(width/4, height/4, width*3/4, height*3/4, fill=mixed)
```

DRAW OTHER SHAPES

- You can draw oval, polygon, text, …

```python
def draw(canvas, width, height):
  canvas.create_oval(100, 50, 300, 150, fill="yellow")
  canvas.create_polygon(100,30,200,50,300,30,200,10, fill="green")
  canvas.create_text(200, 100, text="Python",
                          fill="red", font="Helvetica 26 bold underline")
  canvas.create_text(200, 100, text="Programming", anchor=N,
                          fill="darkBlue", font="Times 28 bold italic")
  canvas.create_text(200, 100, text="Fun", anchor=S,
                          fill="purple", font="Calibri 30 bold italic")
```

TUPLE AND LIST PARAMETER

- Point values can be provided in tuple or list format

```
def draw(canvas, width, height):
  canvas.create_oval(50, 50, 100, 150, fill="red")

  p1 = (100, 100)
  p2 = (200, 150)
  canvas.create_oval(p1, p2, fill="yellow")

  canvas.create_polygon(50, 30, 150, 50, 250, 30, 150, 10,\
  fill="green")

  points = [(250,70),(300,50),(350,70),(300,90)]
  canvas.create_polygon(points, fill="blue")
```

PRACTICE

- You can draw a clock using the code below. Draw hour/minute hand on the clock for 4 o’clock

```
def drawClock(canvas, width, height):
  c = (width/2, height/2)
  r = min(width, height)/3

  canvas.create_oval(c[0]-r-15, c[1]-r-15, c[0]+r+15, c[1]+r+15,\
  fill="yellow")

  for hour in range(12):
    angle = math.pi/2 - 2*math.pi*hour/12
    x = c[0] + r * math.cos(angle)
    y = c[1] - r * math.sin(angle)
    label = str(hour if (hour > 0) else 12)
    canvas.create_text(x, y, text=label, font="Calibri 14 bold")
```

QUESTION?