Welcome!

We'll get started shortly. Please take the Zoom poll in the meanwhile!

CS 49 Week 6

Surajit A. Bose

Agenda

- Graphics:
  - Drawing shapes
  - Centering shapes
  - Worked example: Centered Square
  - Worked example: THIS big!
- Section problem: Random Circles

How to get hold of me / get help from other resources

- Surajit's office hours
  - Fridays 12 noon–1p, directly after section
  - By appointment on Zoom
- Pronto DM for a quick response (usually within a couple hours)
- Email bosesurajit@fhda.edu or Canvas inbox, 24 hr turnaround
- The github repo has section materials, starter code, and solutions

---

- Pronto DM or Canvas inbox for Lane
- Lane's office hours
- Online or in-person tutoring at the STEM center (Room 4213)

The **`graphics`** library and the canvas

The **graphics** library

- The standard Python graphics library is **Tkinter**
- For this class, we use a purpose-built subset called **graphics**
- **graphics** enables drawing lines, rectangles, and ovals
- These are drawn on a background called the **Canvas**
- Need to include this **import** statement at the top of the code:

```
from graphics import Canvas
```

Canvas (Slide 1 of 2)

- The **Canvas** is the background on which to draw shapes
- The canvas must be created by specifying a width and height
- Typically these are specified as constants:

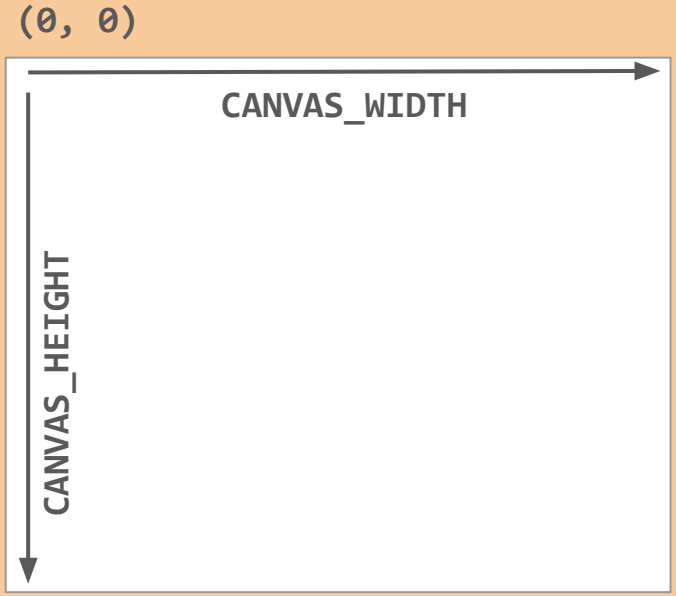
```
from graphics import Canvas

CANVAS_WIDTH = 400
CANVAS_HEIGHT = 300

def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
```

Canvas (Slide 2 of 2)

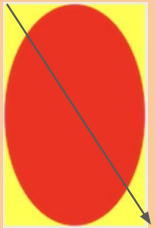
- Positions on the canvas are determined by a coordinate system
- **`(0, 0)`** is the top left of the canvas
- The x-axis goes from **`0`** to **`CANVAS_WIDTH`**
- The y-axis goes from **`0`** to **`CANVAS_HEIGHT`**
- Note that the y-axis values are positive!
- The bottom left of the canvas is **`(CANVAS_WIDTH, CANVAS_HEIGHT)`**



Drawing and centering shapes

Drawing Shapes (Slide 1 of 2)

- Drawing a line requires specifying two sets of coordinates:
  - The **`(x, y)`** of the top left point
  - The **`(x, y)`** of the bottom right point
  - **`canvas.create_line(x1, y1, x2, y2)`**
- Drawing a rectangle requires specifying coordinates for its diagonal:
  - **`canvas.create_rectangle(left_x, top_y, right_x, bottom_y)`**
- Drawing an oval requires specifying coordinates for the diagonal of the bounding box, an imaginary rectangle just surrounding the oval:
  - **`canvas.create_oval(left_x, top_y, right_x, bottom_y)`**



Drawing Shapes (Slide 2 of 2)

- To draw a square or a circle, the difference between **`left_x`** and **`right_x`** should equal the difference between **`top_y`** and **`bottom_y`**
- For any shape, there can be an optional fifth argument that specifies the color: **`'red'`**, **`'blue'`**, **`'green'`**, **`'purple'`**, etc.
- If no color is specified, the default is **`'black'`**
- Sample code for a colored square and circle:

```
canvas.create_rectangle(25, 85, 45, 105, 'fuchsia')
canvas.create_oval(94, 119, 124, 149, 'yellow')
```

Centering shapes on the canvas

- The midpoint of the canvas is **`(CANVAS_WIDTH / 2, CANVAS_HEIGHT / 2)`**
- To center a shape on the canvas:
  - To set left_x, subtract half the desired width of the shape from the midpoint
  - To set top_y, subtract half the desired height of the shape from the midpoint
  - Example: draw a square of size **`SIZE`** that is centered on the canvas:

```
midpoint_width = CANVAS_WIDTH / 2
midpoint_height = CANVAS_HEIGHT / 2
left_x = midpoint_width - SIZE / 2,
top_y = midpoint_height - SIZE / 2
canvas.create_rectangle(left_x, top_y,
    left_x + SIZE, left_y + SIZE)
```

Worked example: THIS big!

THIS big!

- Draw a square where each side is **`THIS_BIG`** pixels
- The square should be centered on a point **`CENTER_X, CENTER_Y`**
- **`CENTER_X, CENTER_Y`** is not the midpoint of the canvas! It is an arbitrary point somewhere on the campus
- **`CANVAS_WIDTH`**, **`CANVAS_HEIGHT`**, **`THIS_BIG`**, **`CENTER_X`**, and **`CENTER_Y`** are all defined constants
- Your code should work whatever the values of those constants

Section problem: Random Circles

Random Circles

- Draw **`N_CIRCLES`** circles of size **`CIRCLE_SIZE`** on the canvas
- The circles should be at random positions on the canvas and of random colors
- To get a random color, use **`random_color()`** as the fifth argument to **`canvas.create_oval()`**
- Stepwise refinement:
  - Draw one circle at a fixed position
  - Randomize the position
  - Repeat as many times as specified (what kind of loop will we need?)
  - Optional: Make the circles fit in the canvas
  - Optional: Put the code to draw the random circles in a separate function

That's all, folks!

Next up: Python Functions!