Carnegie Mellon University
Computer Science Academy

CS1: Unit 1 Review

Position and Size

- The canvas is 400 x 400
- `(x, y)` coordinates refer to points on the canvas
- x values increase as you head right
- y values increase as you head down
- `Rect(left, top, width, height)` draws a rectangle

Fills and Borders

- Fills
    - `Rect(left, top, width, height, fill='green')` draws a green rectangle
    - The fill argument is optional
    - The default fill is `'black'`
    - To not have a fill, set `fill=None`
- Drawing order
    - Shapes drawn first are behind shapes drawn later
- Borders
    - Set the border property.
        - Ex. To draw a rectangle with a green border:
          `Rect(left, top, width, height, border='green')`
    - Now with a border width of 4:
        - `Rect(left, top, width, height, border='green', borderWidth=4)`
    - The default border is `None`
    - The default borderWidth is 2
- Dashes
    - Dashes can be defined as `True` or `False`
        - Ex. To draw a rectangle with a dashed green border:
          `Rect(left, top, width, height, border=‘green’, dashes=True)`
    - Dashes can also be written as a pair of values `(dashWidth, gapWidth)`
        - Ex. To draw a rectangle with dashes of length 1 pixel and gaps of length 4 pixels:
          `Rect(left, top, width, height, border='green', dashes=(1, 4))`
    - The default dashes is `False`

Colors and Gradients

- Colors: find a list of color names here
- To use rgb colors, use `fill=rgb(red, green, blue)`
    - This lets you make custom colors
        - Ex. `rgb(152, 255, 152)` is mint green!
    - Values for red, green, and blue must be between 0 and 255 (inclusive)

- Gradients
  - The default gradient is a radial gradient
    - This fades from the center of the shape outwards
    - Gradients need to use at least 2 colors
      - Ex. `gradient('red', 'green', 'blue')`
  - linear gradients
    - This fades starting from one corner or edge of the shape to the other
    - Use the start to pick what direction the fade starts from
      - Ex. `gradient('red', 'green', 'blue', start='left-top')`

Opacity

- opacity is the opposite of transparency
- Values for opacity are between 0 and 100
- The default opacity is 100 (completely opaque, completely hides shapes behind it)
- A shape with an opacity of 0 is completely transparent

Rectangle, Ovals, Circles, and Lines

- Rectangles and Ovals
  - `Rect(left, top, width, height)` draws a rectangle
  - `Oval(centerX, centerY, width, height)` draws an oval
- Circles
  - `Circle( X, Y, radius)` draws a circle centered at (x, y)
  - To the autograder, this is the same as an oval with equal width and height
- Lines
  - `Line(x1, y1, x2, y2)` draws a line from (x1, y1) to (x2, y2)
  - `Line(x1, y1, x2, y2, lineWidth=5)` draws a thicker line
  - The default `lineWidth` is 2

Labels

- `Label('hello', x, y)` draws a label centered at (x, y)
- `Label('hello', x, y, size=20)` uses size to draw a smaller or larger label
- `Label('hello', x, y, bold=True, italic=True)` draw a bold and italic label
- `Label('hello', x, y, font='monospace')` uses font to change the label's font
  - The currently available fonts are `‘arial’` and `‘monospace’`
- `Label(‘hello’, x, y, fill=None, border=‘red’)` uses fill and border as with other shapes

Regular Polygons and Stars

- Regular Polygons
  - Regular polygons include: triangles, diamonds, pentagons, hexagons, and so on
  - `RegularPolygon(x, y, radius, points)`
    - Ex. Draws a pentagon centered at (100, 100) with a radius of 50: `RegularPolygon(100, 100, 50, 5)`

- Stars
  - Star(x, y, radius, points) # draws a star
  - Ex. Star(100, 100, 50, 5) draws a 5-sided star centered at (100, 100) with a radius of 50
  - Change the roundness to make the star more or less round
  - Values for roundness are between 0 (“spiky") and 100 (almost a circle)

Polygons

- Create polygons by connecting points in order
- Polygon(10, 60, 50, 20, 30, 90) connects (10, 60) to (50, 20) to (30, 90)