Introduction to Programming

Spring 2022

Objects and Graphics

- Overview
- The Object of Objects
- Simple Graphics Programming
- **Using Graphical Objects**
- Interactive Graphics
- **Graphics Module Reference**

Using Graphical Objects

- Computation is performed by asking an object to carry out one of its operations (by sending messages)
- In the previous example we manipulated `GraphWin`, `Point`, `Circle`, `Oval`, `Line`, `Text` and `Rectangle`.
  - These are examples of classes.

Using Graphical Objects

- Each object is an *instance* of some class, and the class describes the properties of the instance.
- Examples

Class	**Objects (Instances)**
Car	my_red_car, my_ford_mustang
Student	student_1, student_2
Point	p1, p2

Using Graphical Objects

- To create a new instance of a class, we use a special operation called a constructor.
- `class-name(<param1>, <param2>, …)`
- `class-name` is the name of the class we want to create a new instance of, e.g. `Circle` or `Point`.
- The parameters are required to initialize the object. For example, `Point` requires two numeric values.
- A class can have multiple constructors
- Example

Using Graphical Objects

- Example
  - `p = Point (50, 60)`
  - The constructor for the point requires two parameters, the x and y coordinates for the point
  - These values are stored as *instance variable*s inside of the object

Using Graphical Objects

- Example:
  - x = 45
  - y = 12.56
  - p = Point (50, 60)

Using Graphical Objects

- To perform an operation on an object, we send the object a message. The set of messages an object responds to are called the *methods* of the object.
- Methods are like functions that live inside the object.
- Methods are invoked using dot-notation:
- `object_name.method_name (<param1>, <param2>, …)`
- Parameter are the data that you want to pass to the method

Using Graphical Objects

- `p.getX()` and `p.getY()` returns the x and y values of the point.
- Provides information about an instance of the class
- Example
- Methods like these are referred to as accessors because they allow us to access information from the instance variables of the object.

Using Graphical Objects

- Other methods change the state of the object by changing the values of the object's instance variables.
- `move(dx, dy)` moves the object dx units in the x direction and dy in the y direction.
- Move erases the old image and draws it in its new position. Methods that change the state of an object are called mutators.
- Example

Using Graphical Objects

- It's possible for two different variables to refer to the same object – changes made to the object through one variable will be visible to the other.

```
leftEye = Circle(Point(80,50), 5)
leftEye.setFill('yellow')
leftEye.setOutline('red')
rightEye = leftEye
rightEye.move(20,0)
```

- The idea is to create the left eye and copy that to the right eye which gets moved over 20 units.

Using Graphical Objects

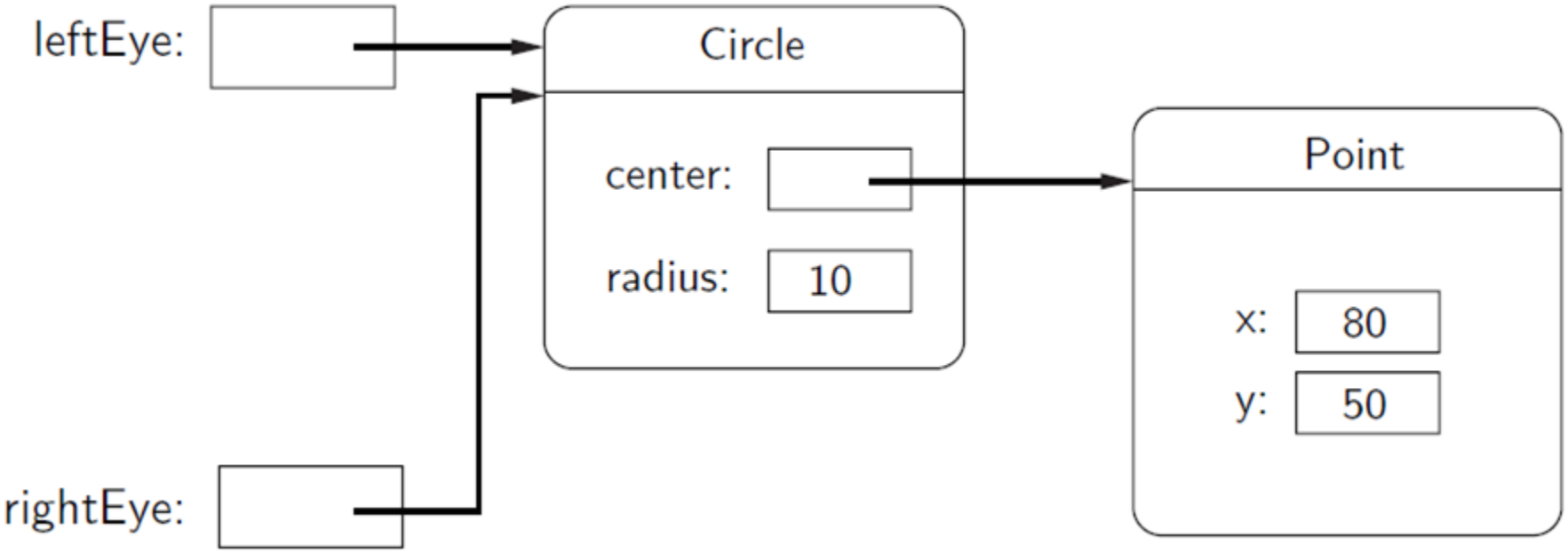
- The assignment

```
rightEye = leftEye
```

makes rightEye and leftEye refer to the same circle!

- The situation where two variables refer to the same object is called aliasing.

Using Graphical Objects



Using Graphical Objects

- There are two ways to get around this.
- We could make two separate circles, one for each eye:

```
leftEye = Circle(Point(80, 50), 5)
leftEye.setFill('yellow')
leftEye.setOutline('red')
rightEye = Circle(Point(100, 50), 5)
rightEye.setFill('yellow')
rightEye.setOutline('red')
```

Using Graphical Objects

- The graphics library has a better solution. Graphical objects have a clone method that will make a copy of the object!

```
leftEye = Circle(Point(80, 50), 5)
leftEye.setFill('yellow')
leftEye.setOutline('red')
rightEye = leftEye.clone()

rightEye.move(20, 0)
```

Handling Textual Input

- The `GraphWin` object provides a `getKey()` method that works like the `getMouse` method.

Handling Textual Input

- There's also an `Entry` object that can get keyboard input.
- The `Entry` object draws a box on the screen that can contain text. It understands `setText` and `getText`, with one difference that the input can be edited.

Handling Textual Input

Class Work

- Describe in your own words the objects produced by each of the following operations:

```
1) Point (130, 130)

2)c = Circle (Point (30, 40), 25)
  c.setFill ('blue')
  c.setOutline ('red')

3) r = Rectangle (Point (20, 20),
        Point (40, 40))
   r.setFill (color_rgb(0, 255, 150))
   r.setWidth (3)
```

Class Work

```
4) l = Line (Point (100, 100),
             Point (100, 200)
   l.setOutline ('red')
   l.setArrow('first')

5) Oval (Point (50, 50),
                Point (60, 100))
6) shape = Polygon (Point (5, 5), Point
(10, 10), Point (5, 10), point (10, 5))
   shape.setFill ('Orange')
```