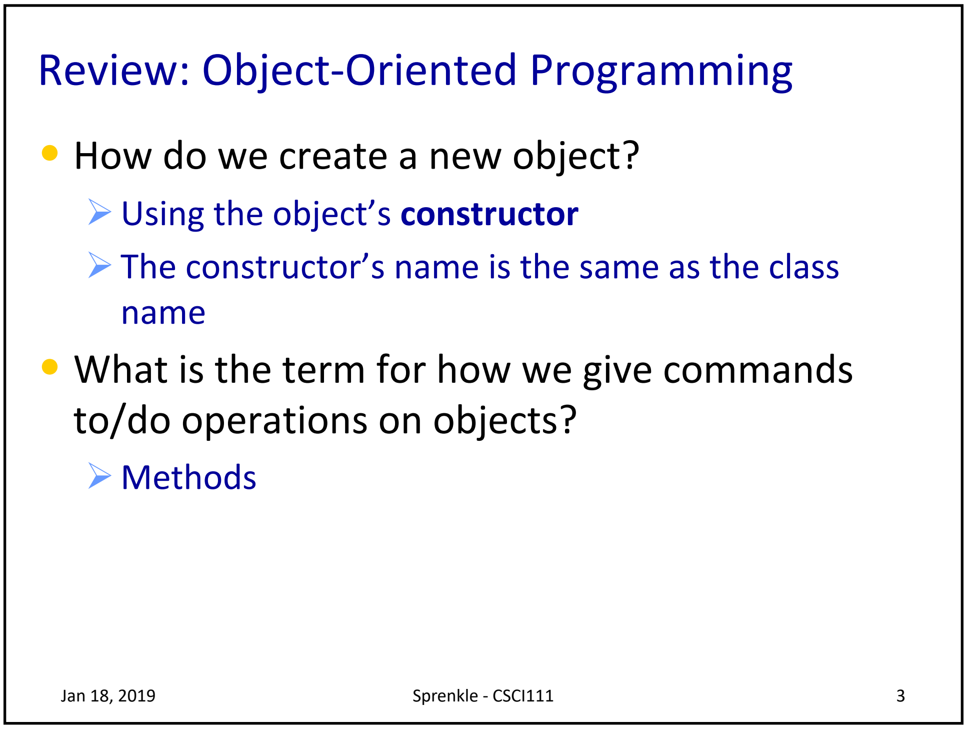
Objectives

- Continuing with OOP
- Broader Issue: Algorithm Accountability

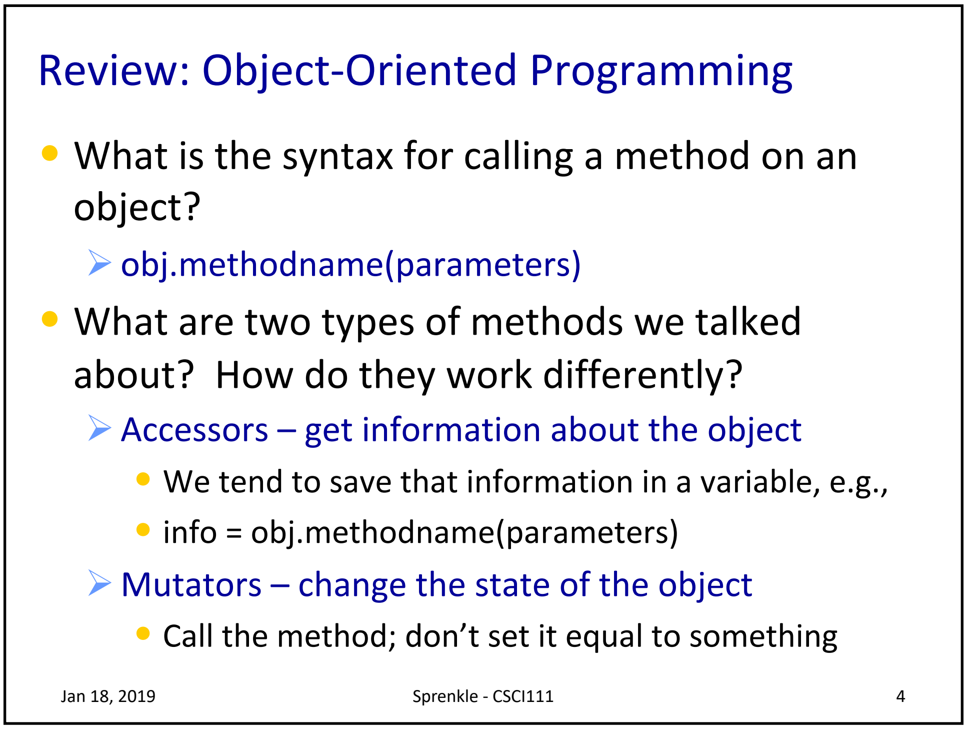
Review: Object-Oriented Programming

- How do we create a new object?
- What is the term for how we give commands to/do operations on objects?
- What is the syntax for calling a method on an object?
- What are two types of methods we talked about?
  - How do they work differently?



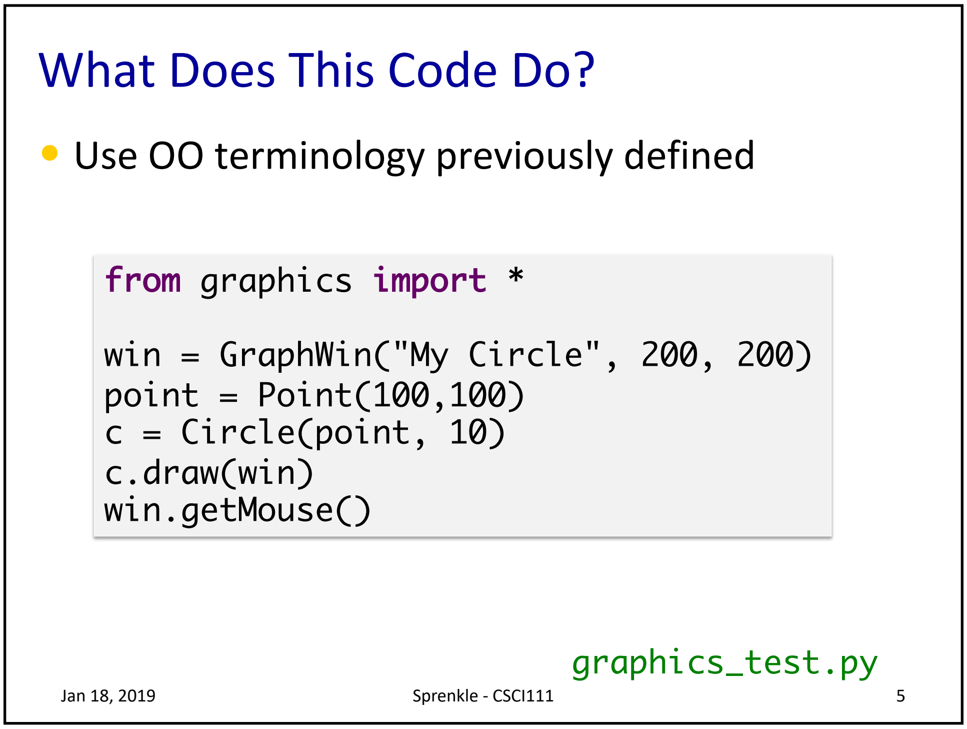
Review: Object-Oriented Programming

- How do we create a new object?
  - Using the object’s **constructor**
  - The constructor’s name is the same as the class name
- What is the term for how we give commands to/do operations on objects?
  - Methods



Review: Object-Oriented Programming

- What is the syntax for calling a method on an object?
  - obj.methodname(parameters)
- What are two types of methods we talked about? How do they work differently?
  - Accessors – get information about the object
    - We tend to save that information in a variable, e.g.,
    - info = obj.methodname(parameters)
  - Mutators – change the state of the object
    - Call the method; don’t set it equal to something



What Does This Code Do?

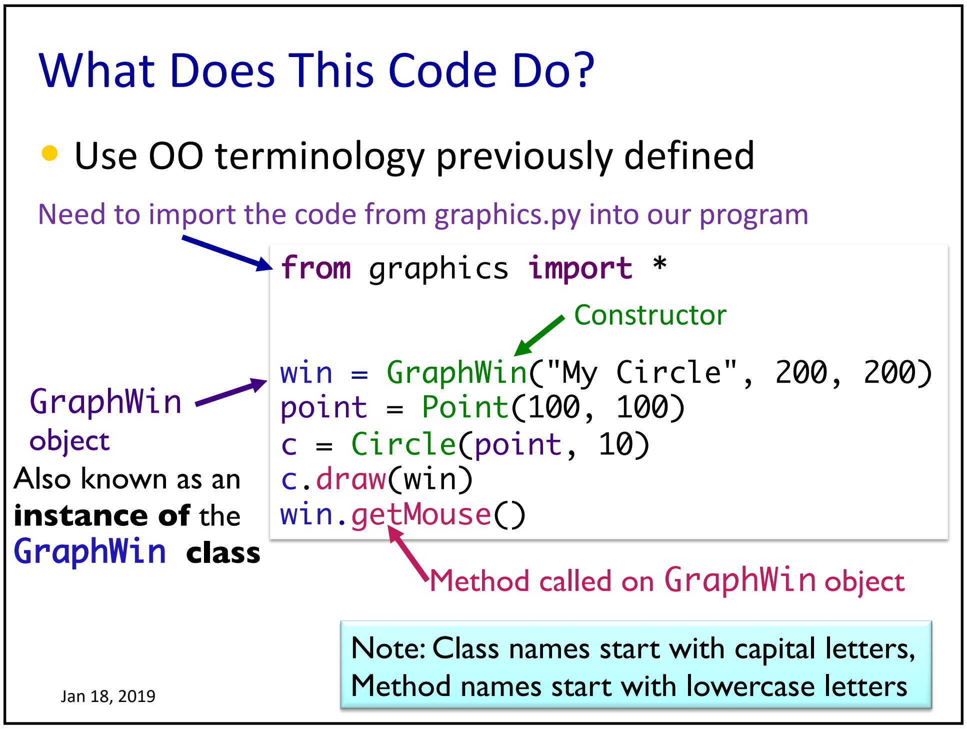
- Use OO terminology previously defined

```
from graphics import *

win = GraphWin("My Circle", 200, 200)
point = Point(100,100)
c = Circle(point, 10)
c.draw(win)
win.getMouse()
```

graphics_test.py

Jan 18, 2019 Sprenkle - CSCI111 5



What Does This Code Do?

- Use OO terminology previously defined

Need to import the code from graphics.py into our program

```
from graphics import *

win = GraphWin("My Circle", 200, 200)
point = Point(100, 100)
c = Circle(point, 10)
c.draw(win)
win.getMouse()
```

Constructor

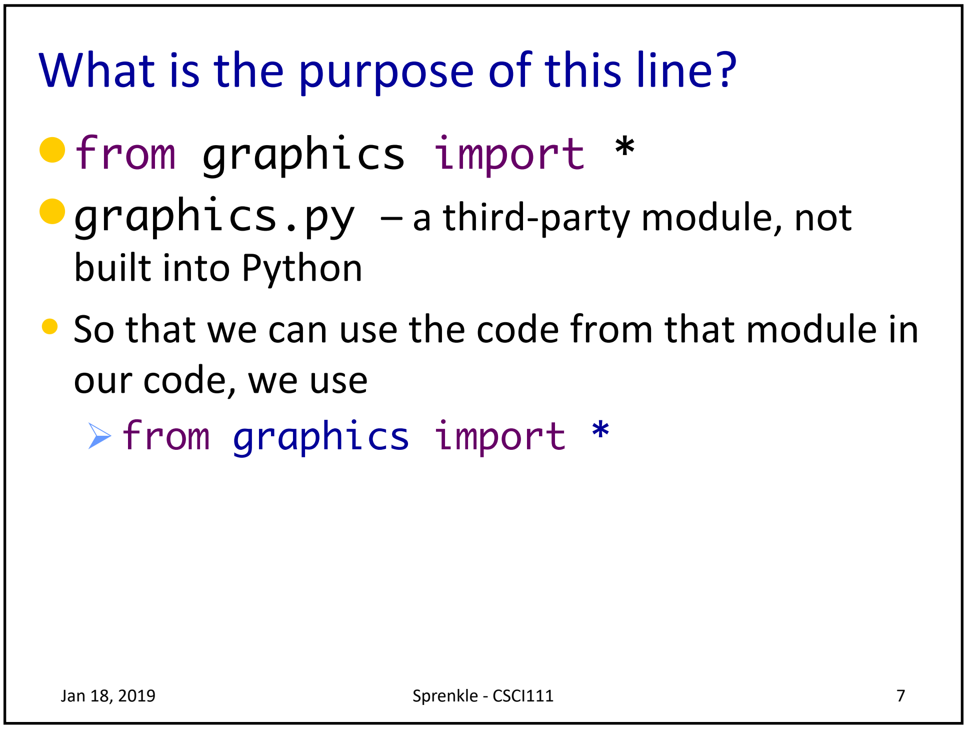
GraphWin object

Also known as an **instance of** the GraphWin **class**

Method called on GraphWin object

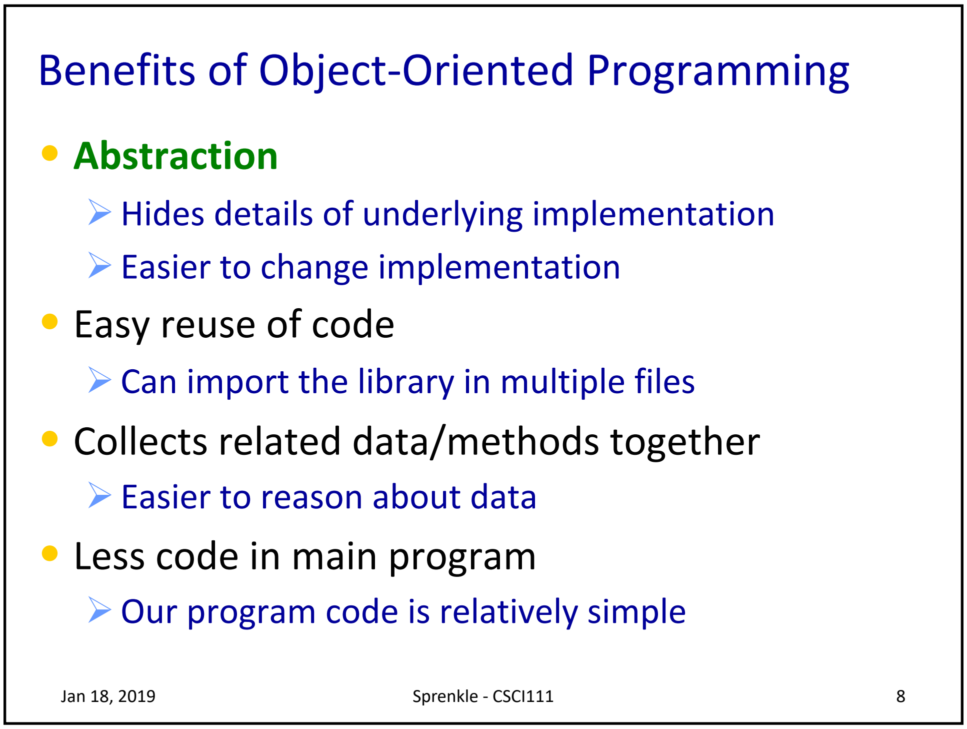
Note: Class names start with capital letters, Method names start with lowercase letters

Jan 18, 2019



What is the purpose of this line?

- `from graphics import *`
- `graphics.py` – a third-party module, not built into Python
- So that we can use the code from that module in our code, we use
  - `from graphics import *`



Benefits of Object-Oriented Programming

- **Abstraction**
  - Hides details of underlying implementation
  - Easier to change implementation
- Easy reuse of code
  - Can import the library in multiple files
- Collects related data/methods together
  - Easier to reason about data
- Less code in main program
  - Our program code is relatively simple

Colors

- Strings, such as "blue4"
- Can also create colors using the *function* `color_rgb(<red>,<green>,<blue>)`
  - Parameters in the range [0,255]
  - Example use:

    ```
    darkBlueGreen = color_rgb(10, 100, 100)
    win.setBackground(darkBlueGreen)
    ```

    - Background is a dark blue/green color
  - Example color codes:
    - `http://en.wikipedia.org/wiki/List_of_colors`

Using the Graphics Library

- How do we create an instance of a Rectangle?
- Draw the rectangle?
- Shift the instance of the Rectangle class to the **right** 10 pixels
- What are the x- and y- coordinates of the upper-left corner of the Rectangle now?

`rectangle.py`

Problem: Draw a Full-Canvas Tic-Tac-Toe Board

- Using the Graphics API
- Make lines purple with line width 3
- The width and height of the canvas is 200

`tictactoe.py`

Modification to Tic-Tac-Toe

- **clone** a vertical line and horizontal line and shift appropriately
- Why clone?
  - Maintain the same properties (color, line-width, length)
  - Simplifies code

`tictactoe_clone.py`

Moving a Circle According to the User

- Draw a circle in the upper left-hand corner of the screen
- Tell the user to click somewhere
- Move the circle to where the user clicked

Looking Ahead

- Lab 2 Prep – due before class
  - Basically, Chapter 4
- Lab 2 Tuesday
  - What are you going to draw?

(postponed broader issue)