Events and Randomness

Assignment 2 Due Right Now

Assignment 3 Demo

Breakout!

- Due next Friday, February 8.
- YEAH hours (assignment review) on Monday, 7-9PM in Herrin T175.
- **Start Early!**
  - There is a nice breakdown of the required tasks suggested in the handout.
  - This program is not as hard to write as it may seem.
- **Have Fun!**
  - There are a **lot** of fun extensions you can add onto the basic functionality.
  - We love giving extra credit on this one. ^_^

Events

Events

- An **event** is some external stimulus that your program can respond to.
- Common events include:
  - Mouse motion / clicking.
  - Keyboard buttons pressed.
  - Timers expiring.
  - Network data available.

Events

An **event** is some external stimulus that your program can respond to.

Common events include:

- Mouse motion / clicking.
- Keyboard buttons pressed.
- Timers expiring.
- Network data available.

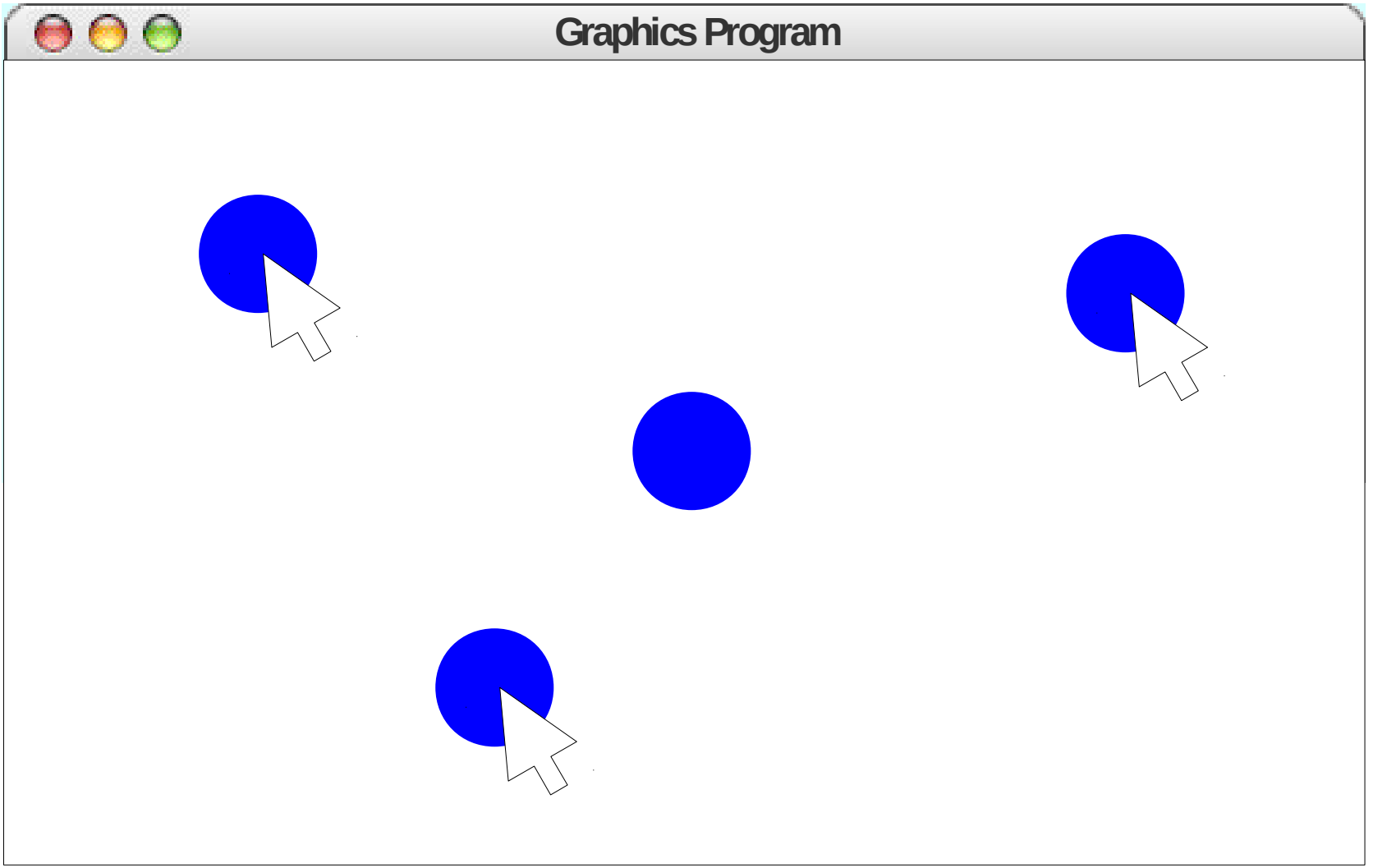
Responding to Mouse Events

- To respond to events, your program must
  - Indicate that it wants to receive events, and
  - Write methods to handle those events.
- Call the **`addMouseListeners()`** method to have your program receive mouse events.
- Write appropriate methods to process the mouse events.

Methods for Handling Events

- Define any or all of the following mouse event handlers to respond to the mouse:
  - `public void mouseMoved(MouseEvent e)`
  - `public void mouseDragged(MouseEvent e)`
  - `public void mousePressed(MouseEvent e)`
  - `public void mouseReleased(MouseEvent e)`
  - `public void mouseClicked(MouseEvent e)`
  - `public void mouseEntered(MouseEvent e)`
  - `public void mouseExited(MouseEvent e)`
- You must also `import java.awt.event.*;` for the **`MouseEvent`** class.

A Friendly Circle



Let's Code it Up!

A Problem of Scoping

- The **`mouseMoved`** handler has no way of referring to the existing circle because it is a local variable in a different method.
- How do we make it possible for the listener to know about the circle?

Instance Variables

- An **instance variable** (sometimes called a **field**) is a variable that can be read or written by any of the methods of a class.
- Syntax (defined outside of any method):

```
private type name;
```

- Instance variables are used to store information that
  - Must persist throughout the program, and
  - Cannot be stored as local variables or parameters.

The Importance of Style

- General rule of thumb:

  Don't make a variable an instance variable unless you have to.

- Use local variables for temporary information.
- Use parameters to communicate data into a method.
- Use return values to communicate data out of a method.

Being Random


pictureisunrelated.com

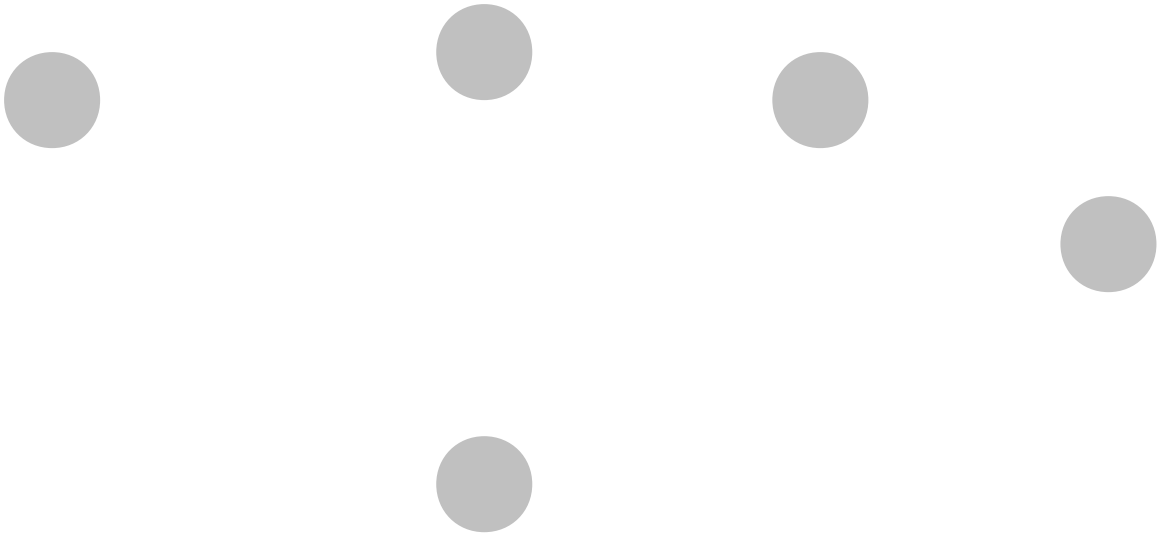
Random Number Generators

RandomGenerator

- The class **RandomGenerator** acts as a random number generator.
  - Need to `import acm.util.*;`
- An instance of **RandomGenerator** can be used to generate random numbers.

Putting it All Together

A Snowfall Simulation



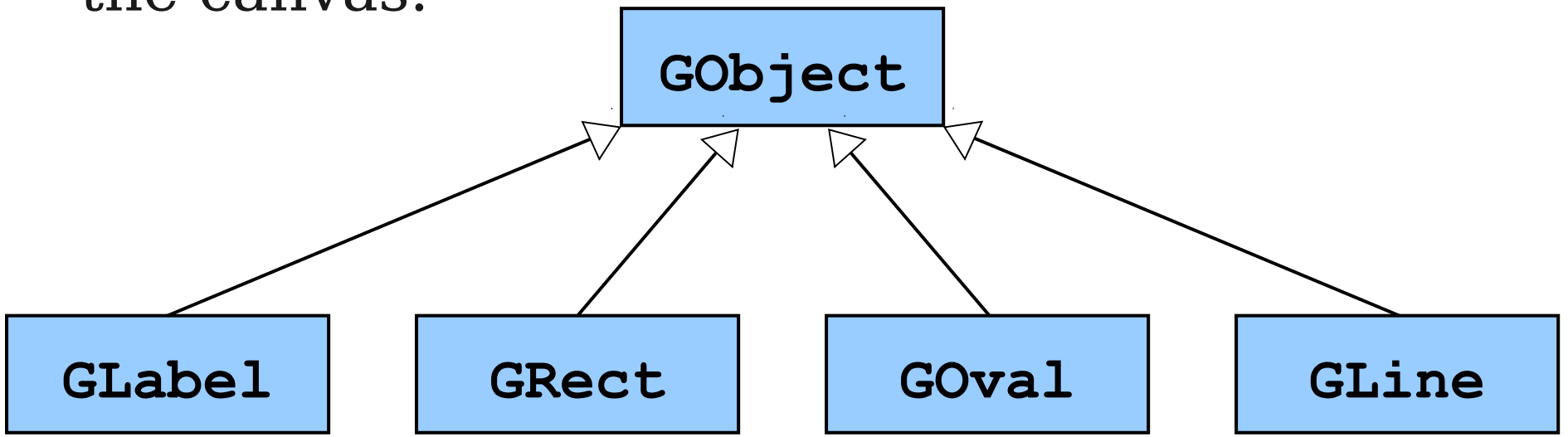
Let it Snow!

Accessing the Canvas

- It is possible to determine what, if anything, is at the canvas at a particular point.
- The method

```
GObject getElementAt(double x, double y);
```

returns which object is at the given location on the canvas.



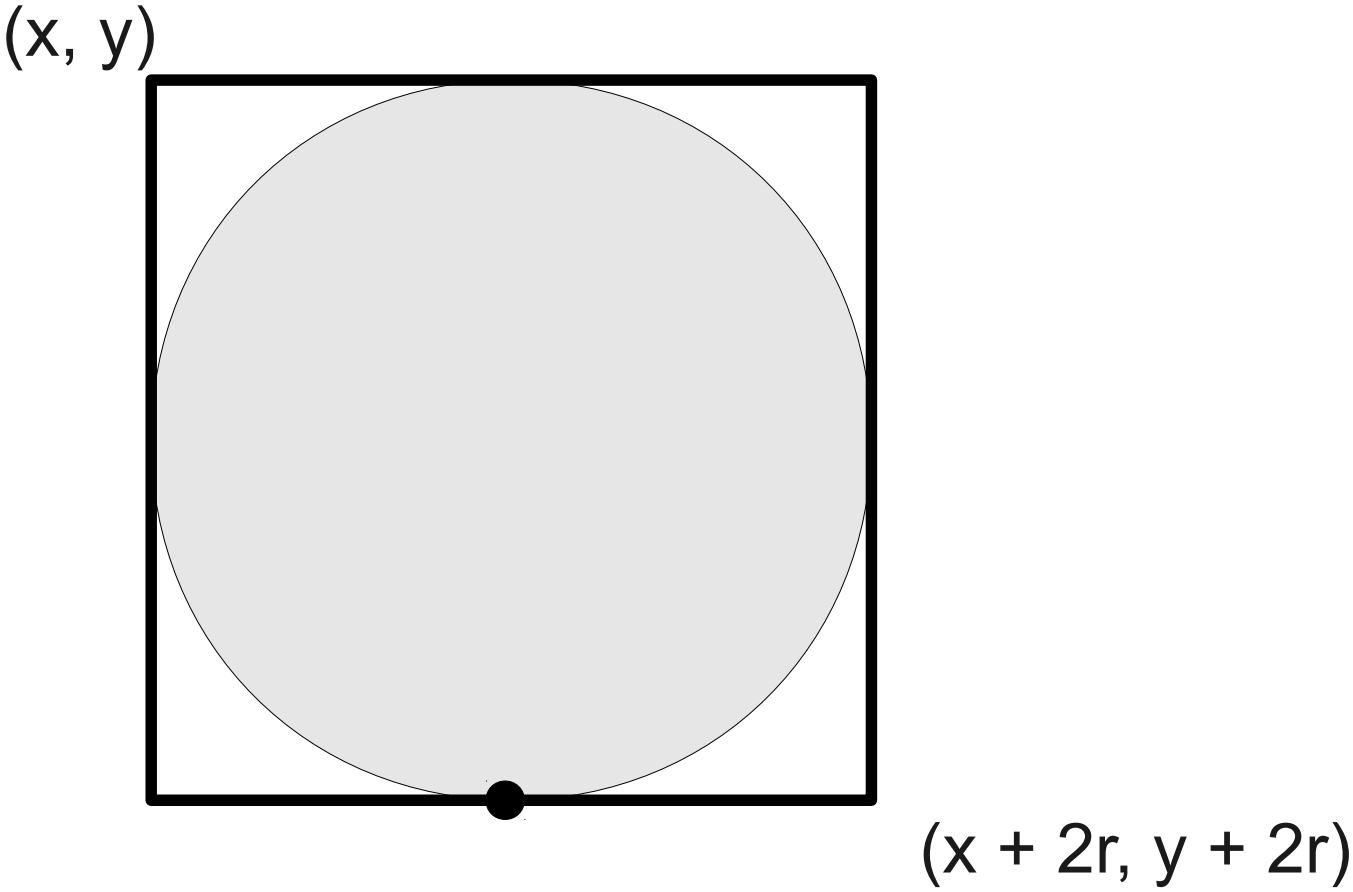
Accessing the Canvas

- It is possible to determine what, if anything, is at the canvas at a particular point.
- The method

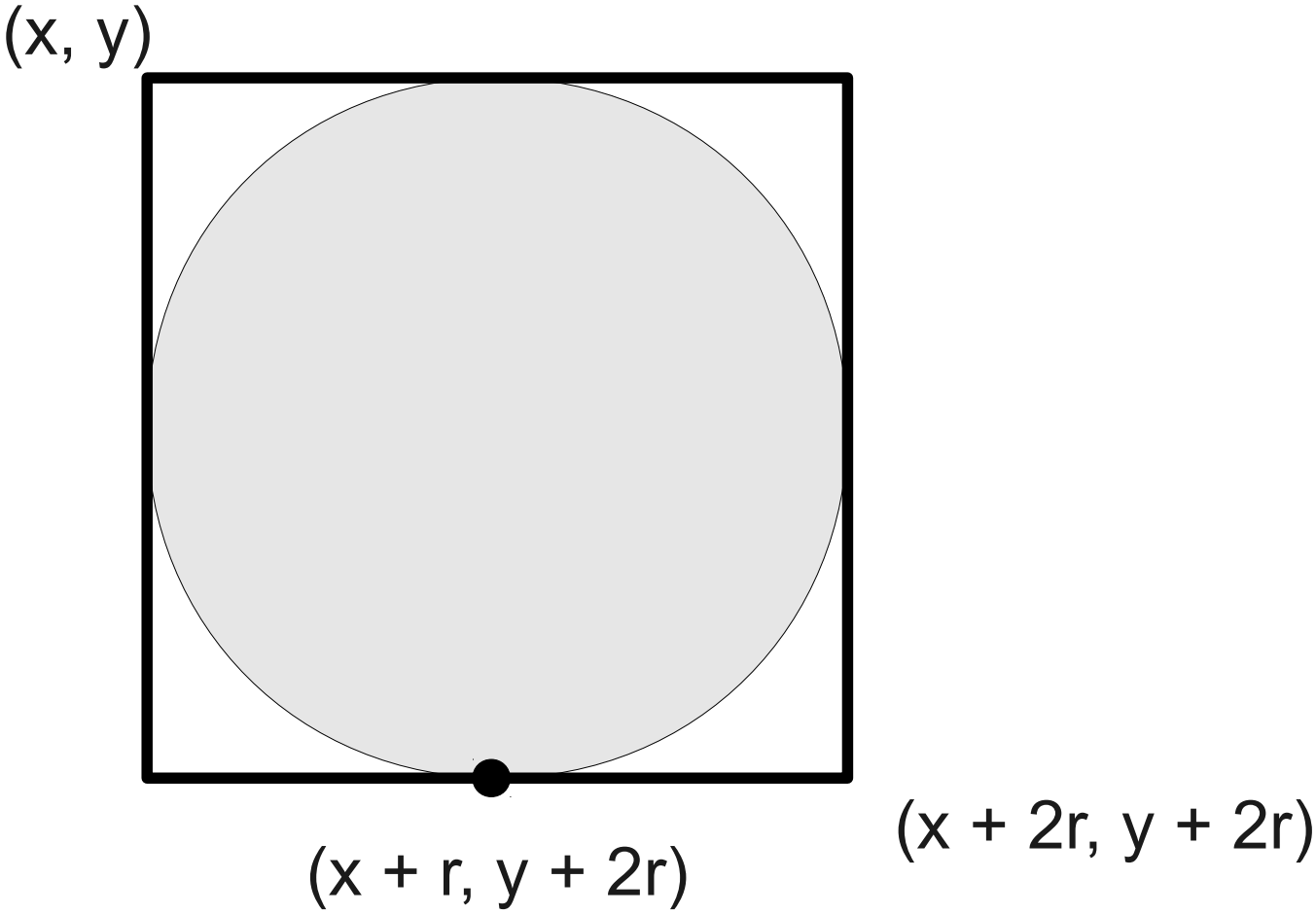
  ```
  GObject getElementAt(double x, double y);
  ```

  returns which object is at the given location on the canvas.
- The return type is **`GObject`**, since we don't know what specific type (**`GRect`**, **`GOval`**, etc.) is really there.
- If no object is present, the special value **`null`** is returned.

A Simple Collision Detector

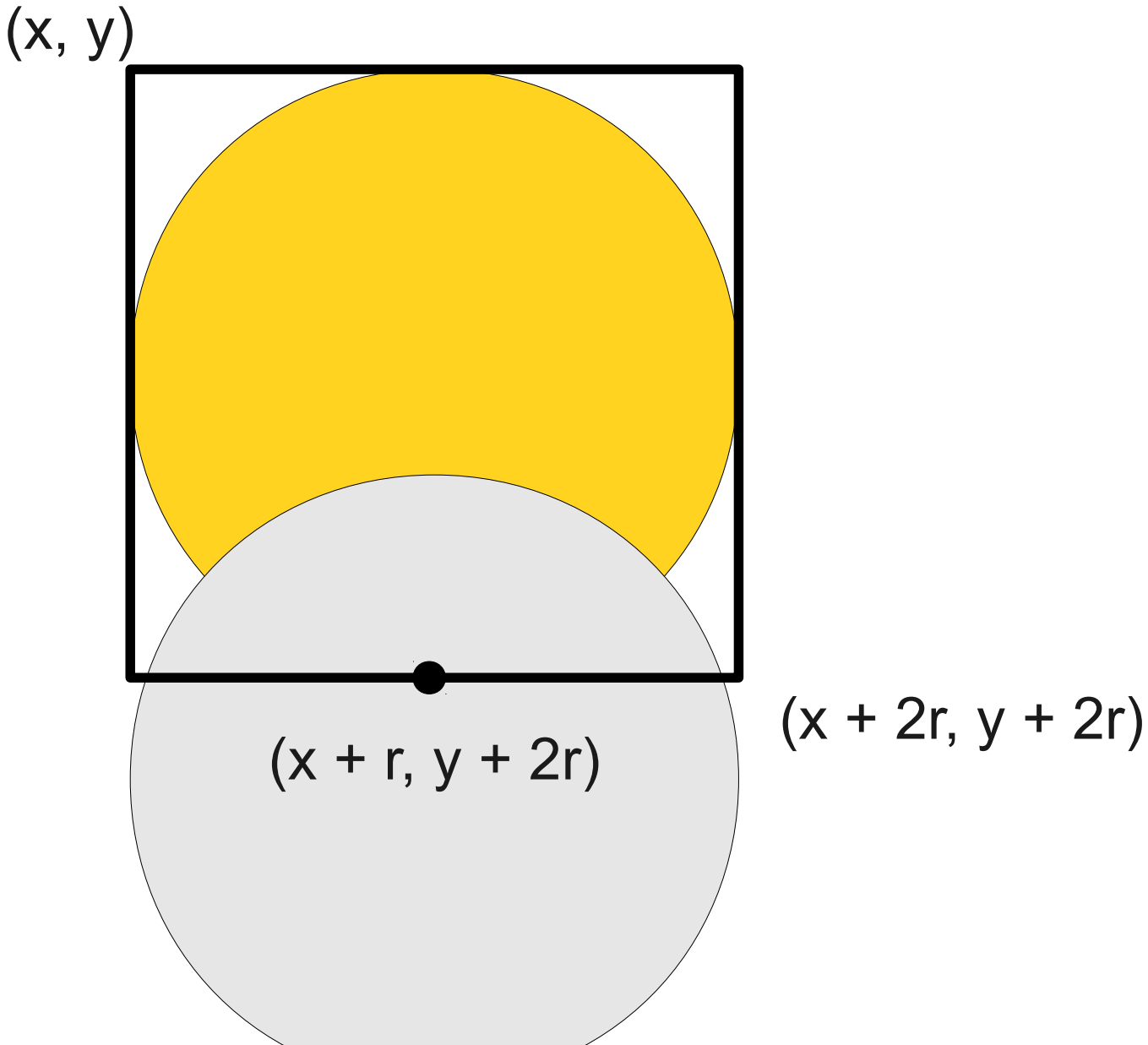


A Simple Collision Detector



What Went Wrong?

A Simple Collision Detector



Reordering Objects

- Each GObject can have its **z-order** adjusted.
- The method

  ***object*`.sendToBack();`**

  moves the object to the back of the z-order.
- **`getElementAt`** will return the topmost object where it hits.