Randomness & Events

An Interesting Website

www.boxcar2d.com

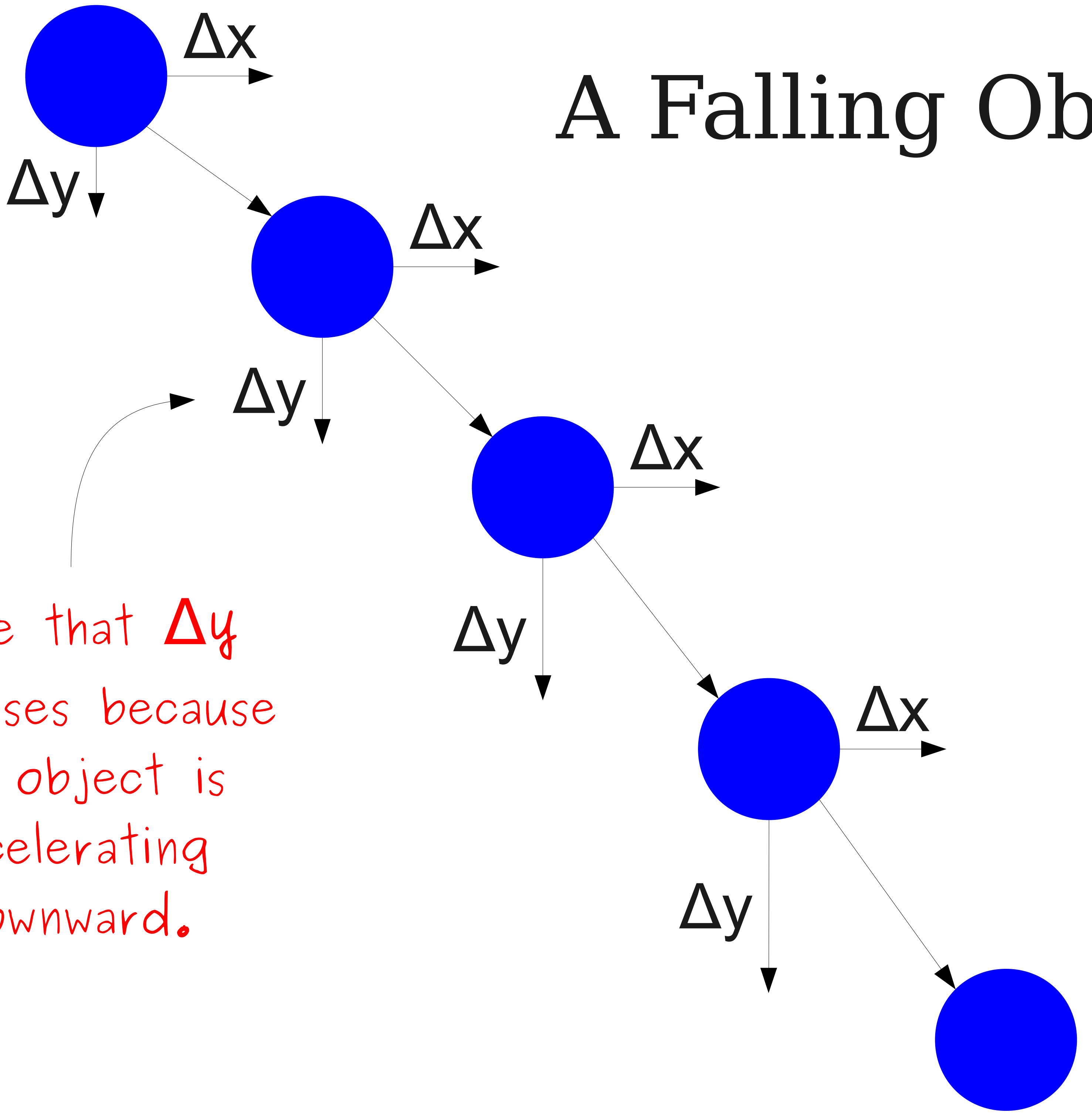
Animation

- By repositioning objects after they have been added to the canvas, we can create animations.
- General pattern for animation:

```
while (not-done-condition) {
    update graphics
    pause(pause-time);
}
```

Physics Simulation

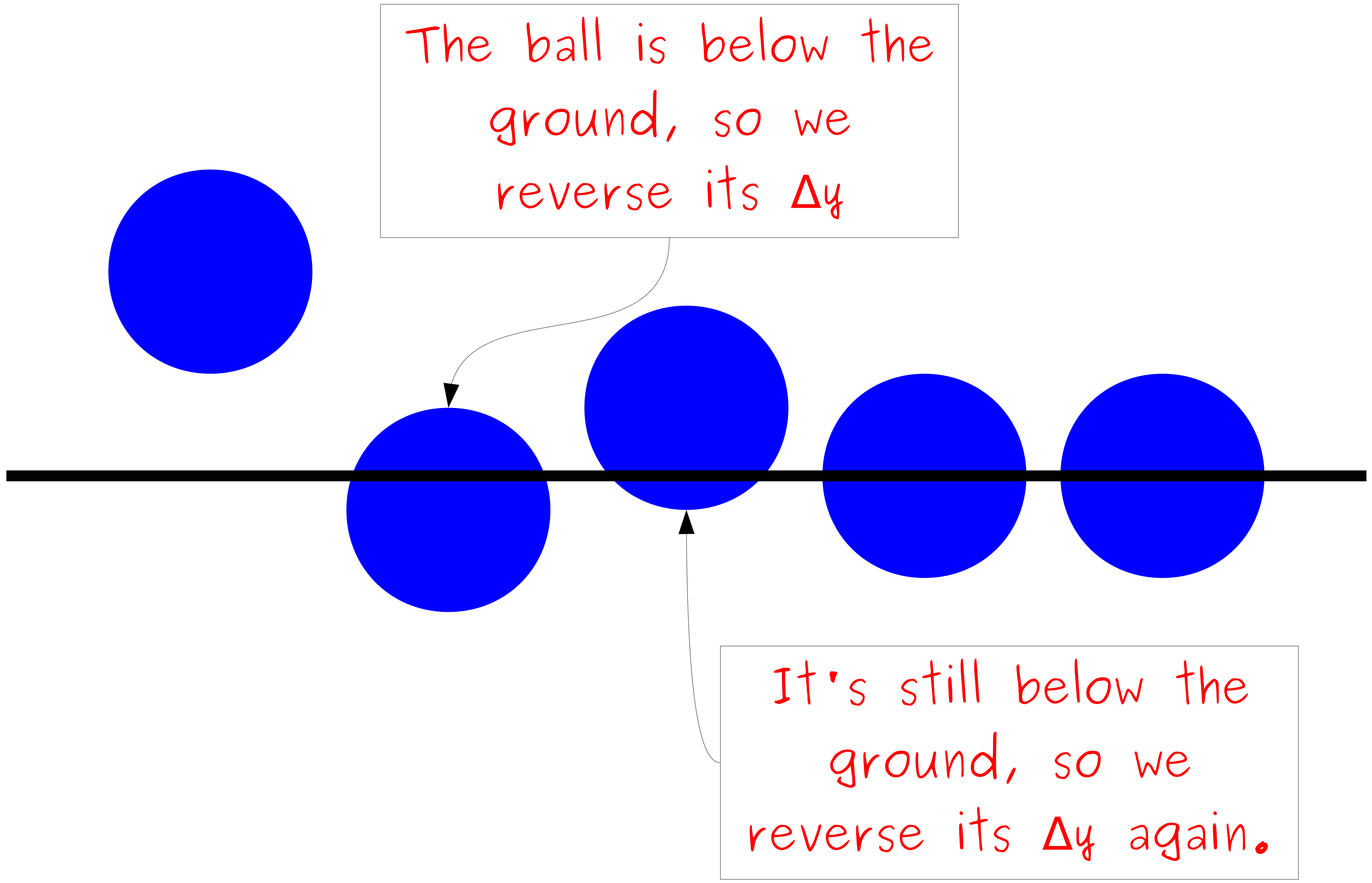
A Falling Object



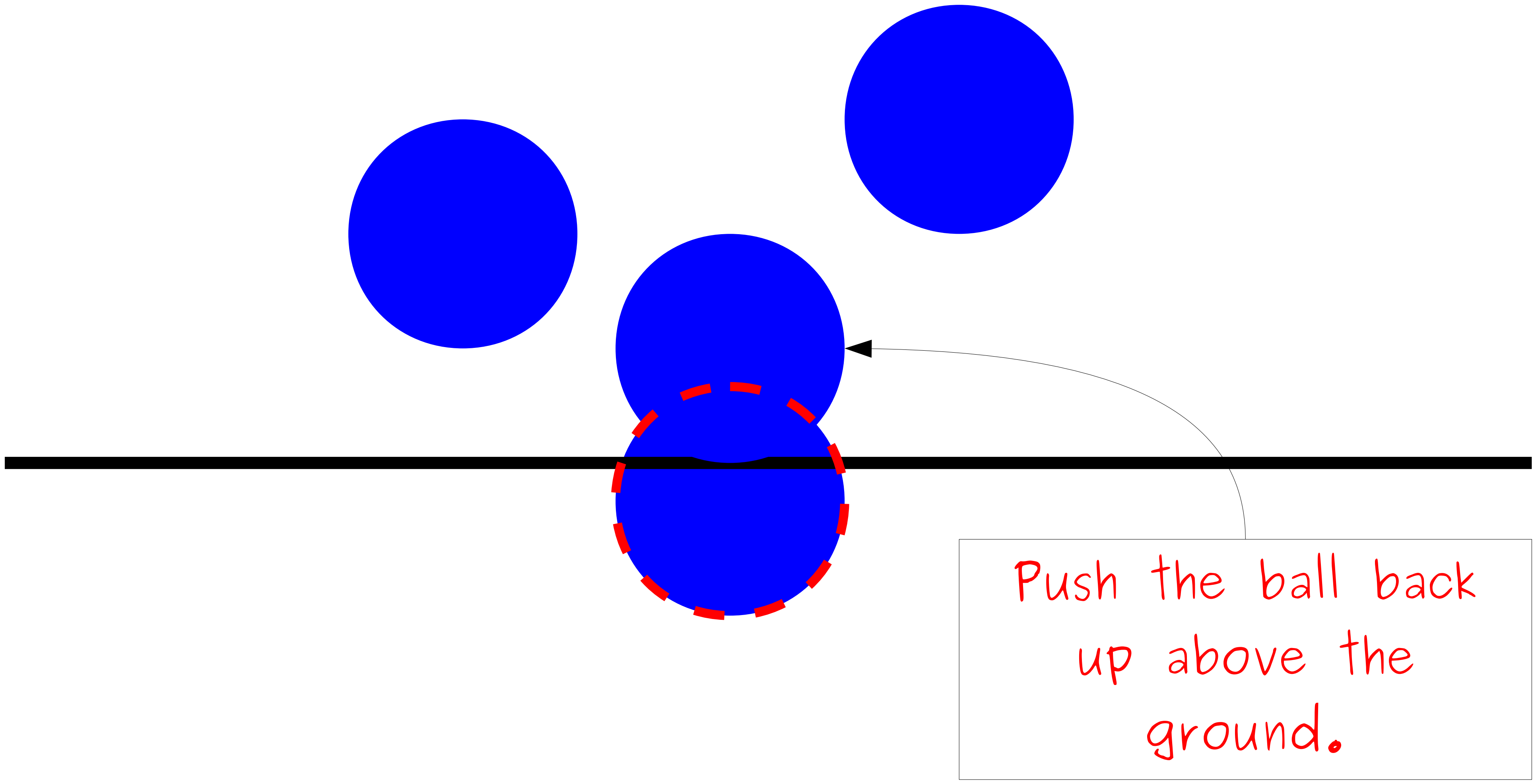
Note that Δy increases because the object is accelerating downward.

From Last Time…

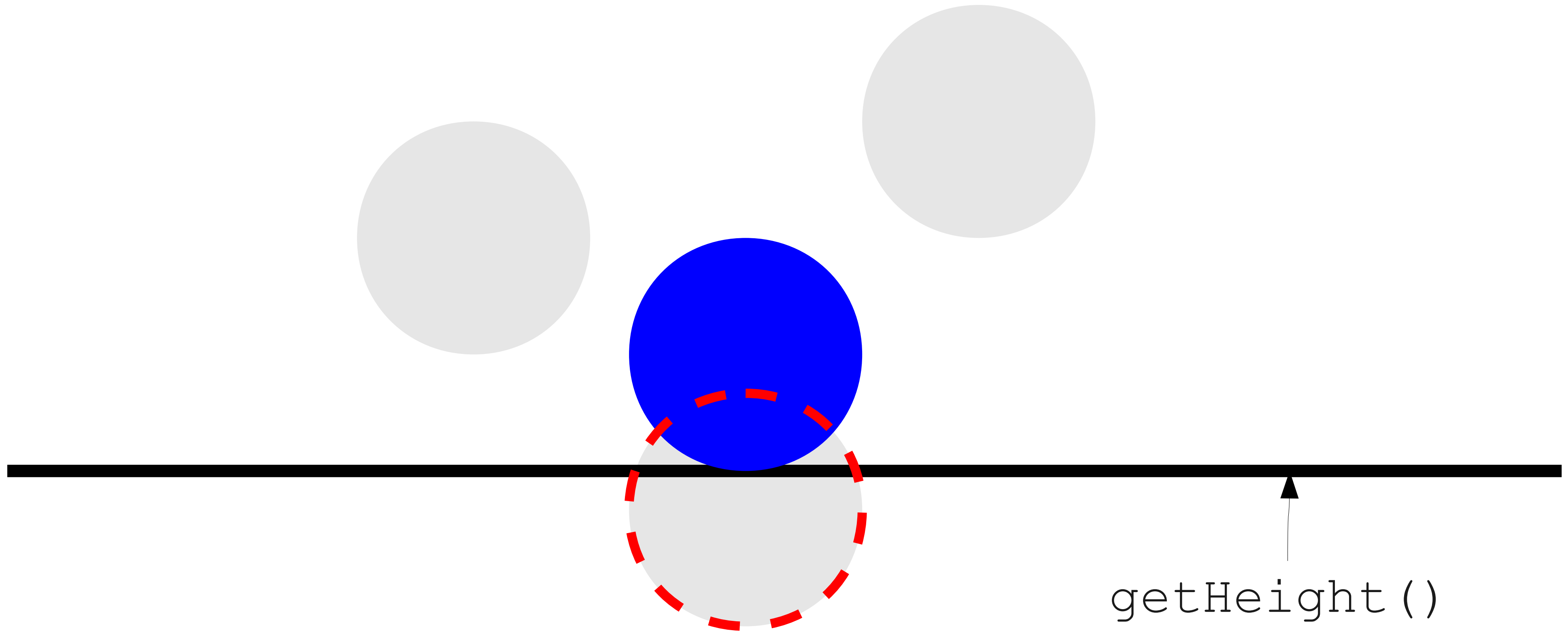
A Sticky Situation


The ball is below the ground, so we reverse its Δy
It's still below the ground, so we reverse its Δy again.

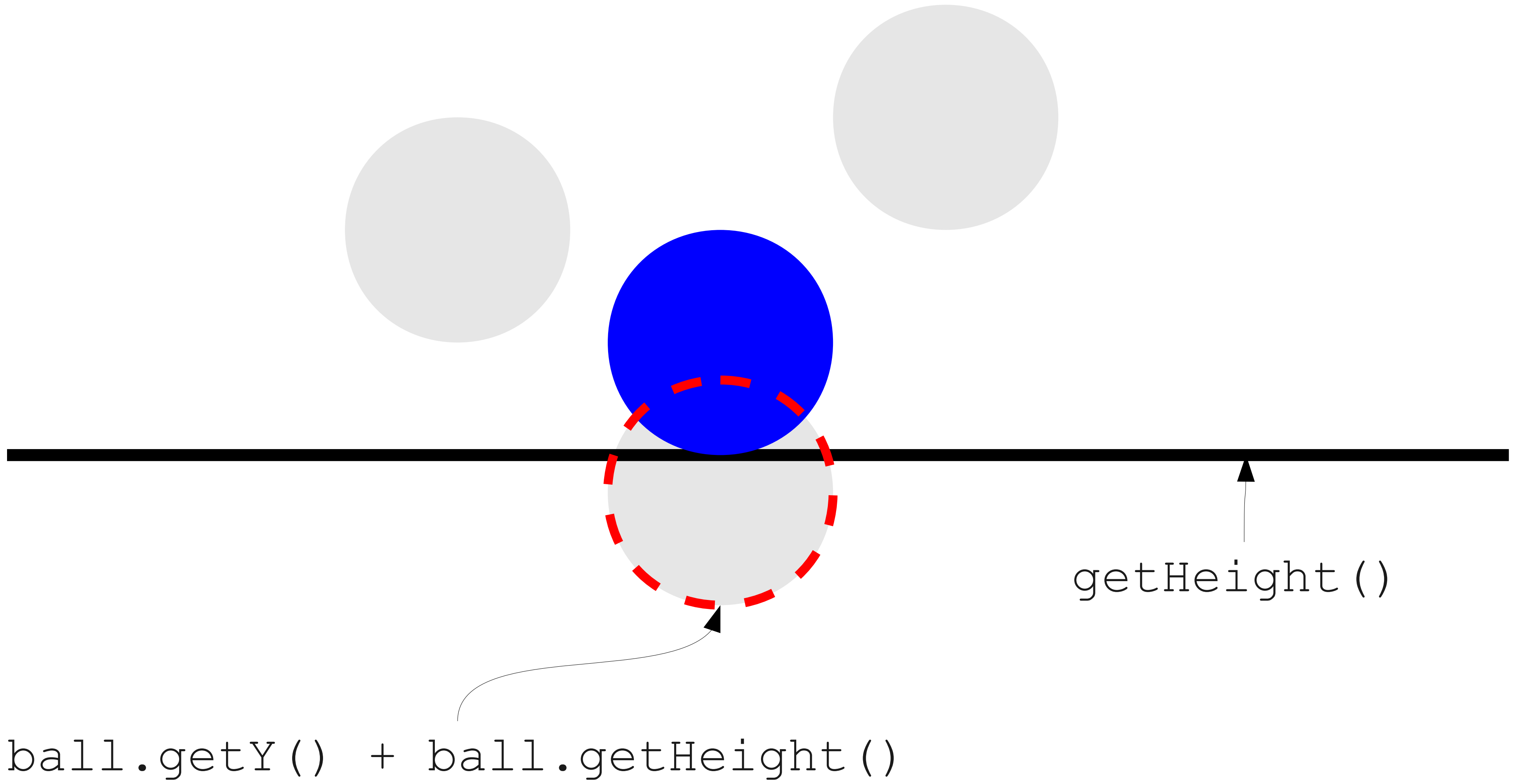
Unsticking the Situation



Unsticking the Situation



Unsticking the Situation

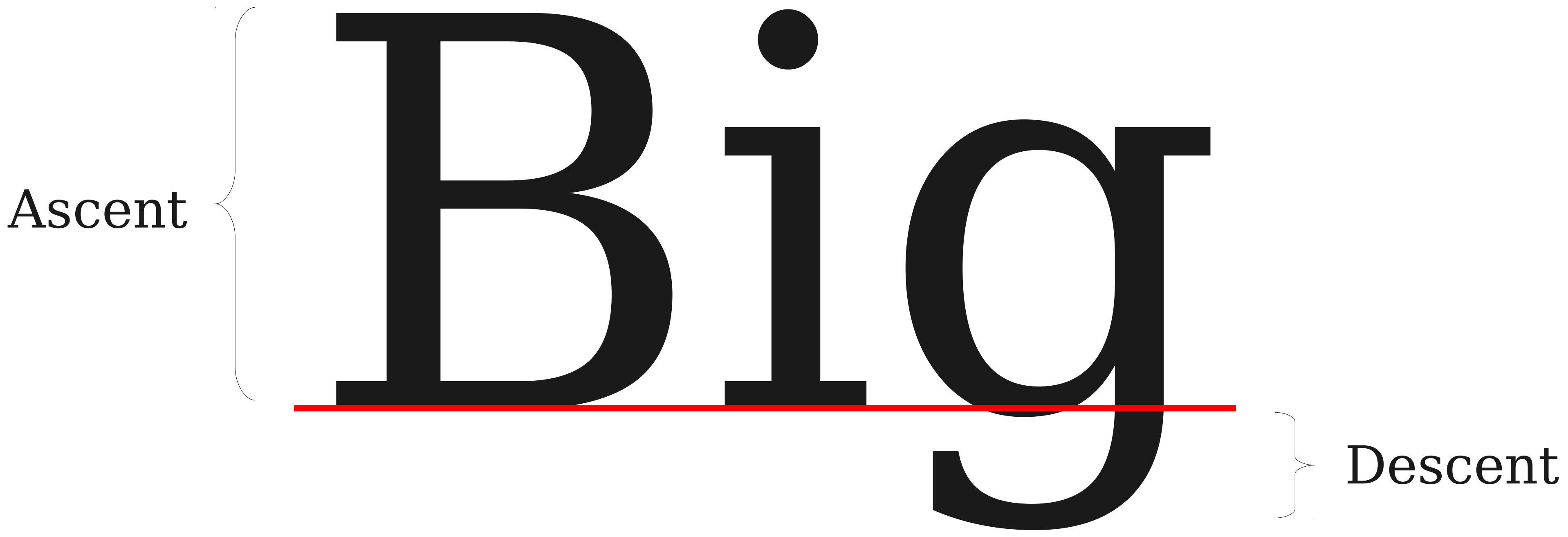


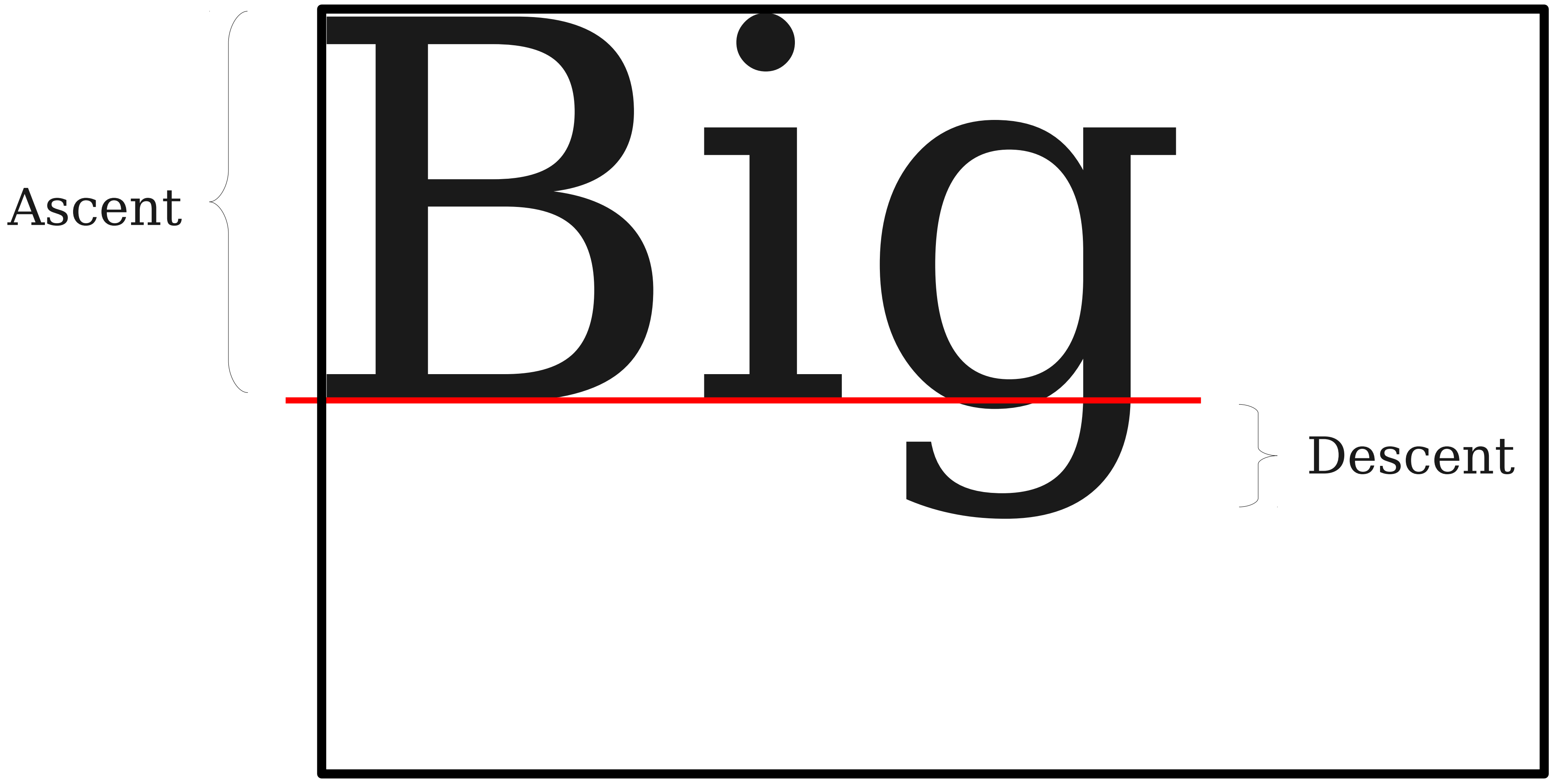
Being Random

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.


Ascent
Big
Descent


Ascent
Big
Descent

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.

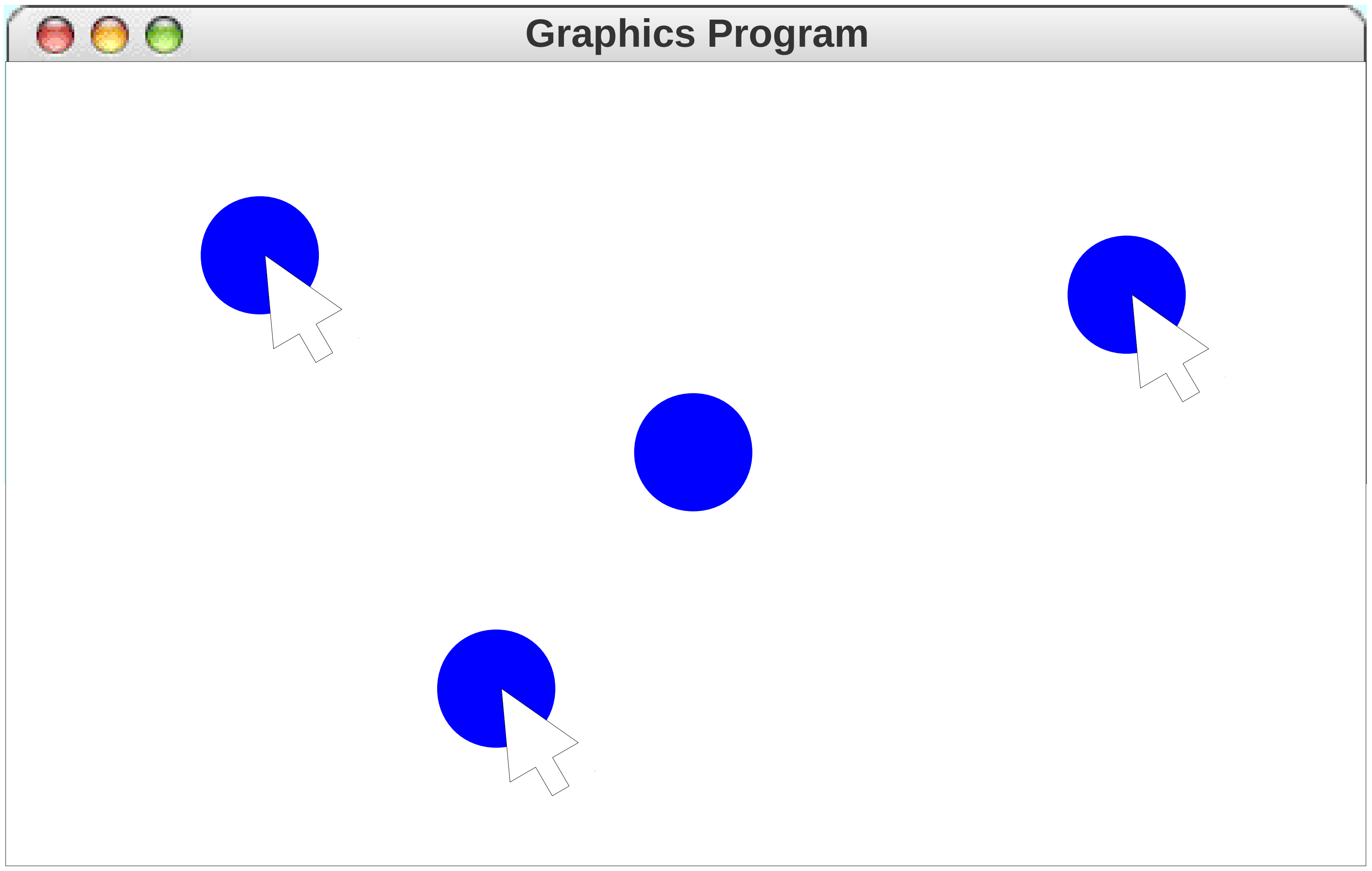
Responding to Mouse Events

- 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.