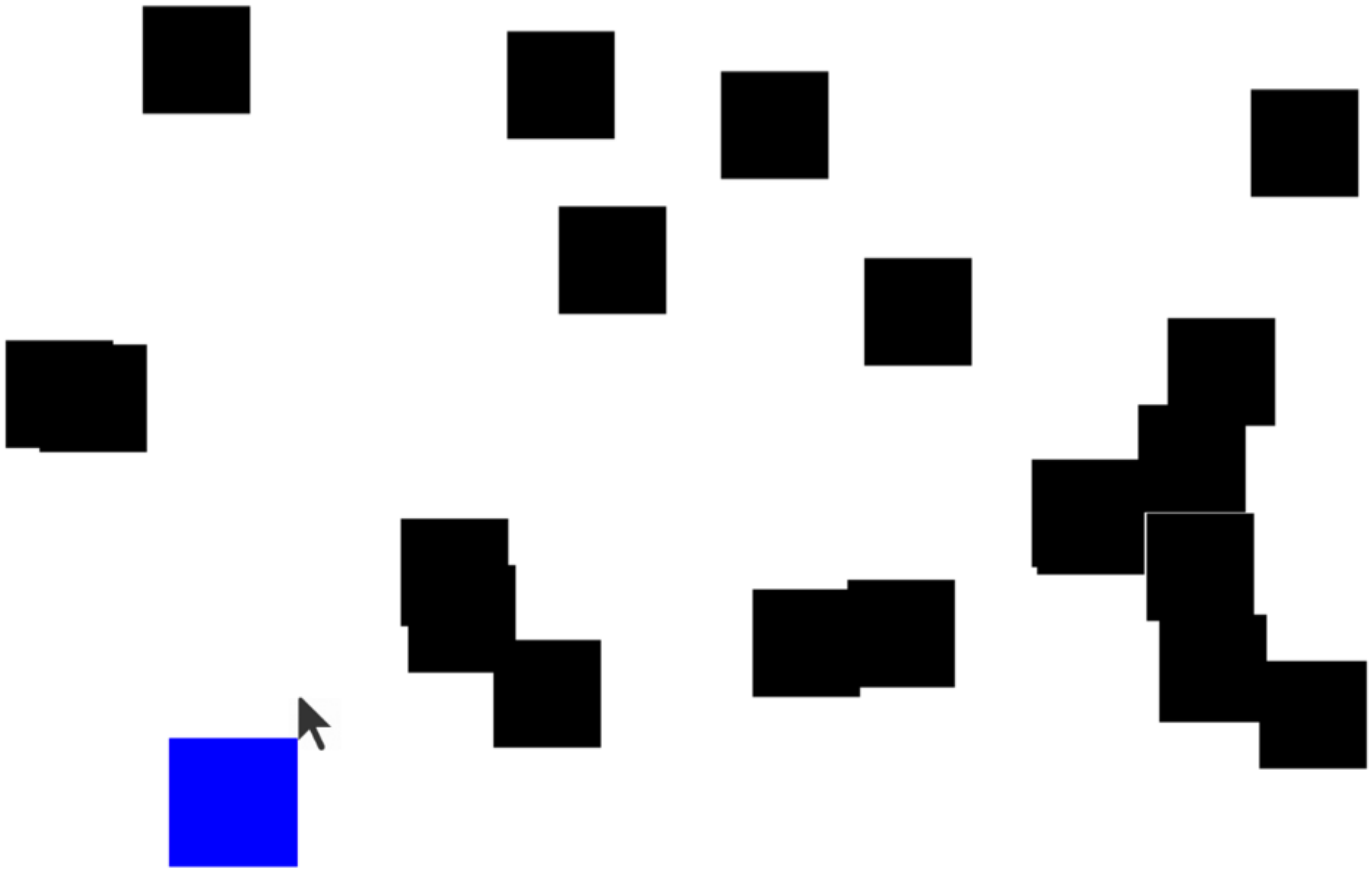
Events

Chris Piech

CS106A, Stanford University

Catch Me If You Can



We’ve Gotten Ahead of Ourselves

Source: The Hobbit

Start at the Beginning

Source: The Hobbit

Learning Goals

1. Write a program that can respond to mouse events
2. Use an instance variable in your program

Events

```
public void run() {
    // Java runs this when program launches
}

public void mouseClicked(MouseEvent event) {
    // Java runs this when mouse is clicked
}

public void mouseMoved(MouseEvent event) {
    // Java runs this when mouse is moved
}
```

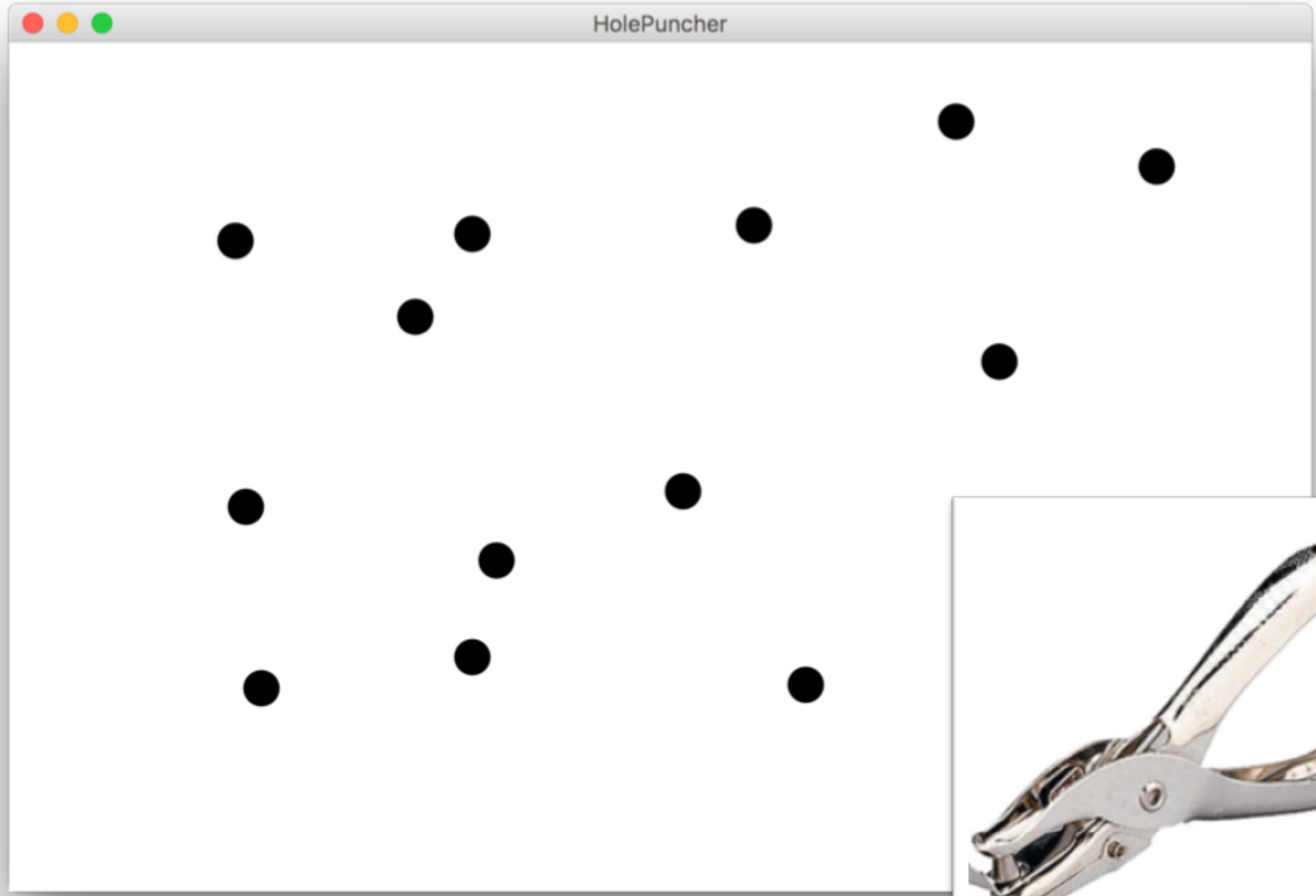
The Listener Model

```
public void run() {
    // 1. add mouse listeners
    addMouseListeners();
}

public void mouseClicked(MouseEvent event) {
    // Java runs this when mouse is clicked
}

public void mouseMoved(MouseEvent event) {
    // Java runs this when mouse is moved
}
```

Examples

Mouse Location

Hole Puncher



Now With Dancing Children

Normal Program

Run Method

Normal Program

Run Method

```
public void run() {
    while(true) {
        update();
        pause(DELAY);
    }
}
```

Normal Program

Run Method

```
public void run() {
    while(true) {
        update();
        pause(DELAY);
    }
}
```

Normal Program

Run Method

```
public void run() {
    while(true) {
        update();
        pause(DELAY);
    }
}
```

Normal Program

Run Method

```
public void run() {
    while(true) {
        update();
        pause(DELAY);
    }
}
```

Normal Program

Run Method

```
public void run() {
    while(true) {
        update();
        pause(DELAY);
    }
}
```

Normal Program

Run Method

New Listener Characters

Mouse Listener

Mouse Moved Method

Program with a Mouse Method

Run Method

Mouse Moved Method

Program Starts Running

Run Method

Mouse Moved Method

Add Mouse Listener

Run Method

Mouse Moved Method

Mouse Listener

```
addMouseListeners();
```

Program Runs as Usual

Run Method

Mouse Moved Method

Mouse Listener

Mouse Moved!

Run Method

Mouse Moved Method

Mouse Listener

Calls Mouse Moved Method

Run Method

Mouse Moved Method

Mouse Listener

When done, Run continues.

Run Method

Mouse Moved Method

Mouse Listener

Keeps Doing Its Thing...

Run Method

Mouse Moved Method

Mouse Listener

Mouse Moved!

Run Method

Mouse Moved Method

Mouse Listener

Calls Mouse Moved Method

Run Method

Mouse Moved Method

Mouse Listener

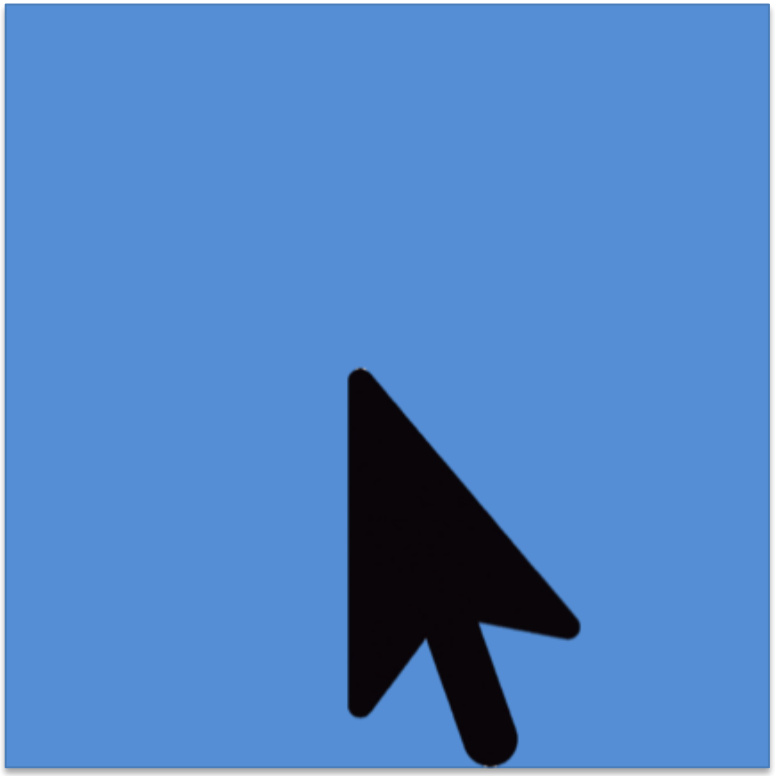
When done, Run continues.

Run Method

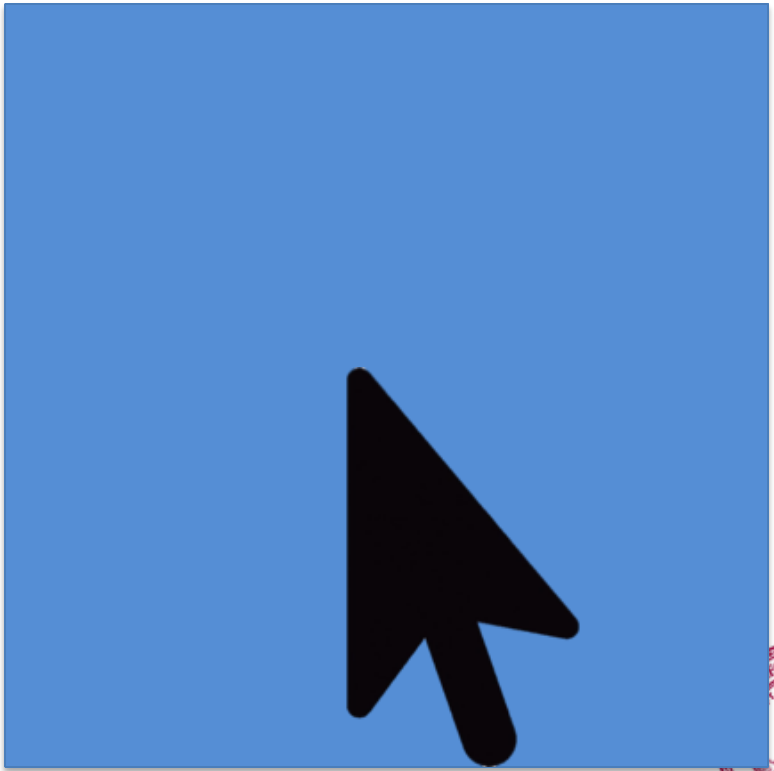
Mouse Moved Method

Mouse Listener

Mouse Tracker

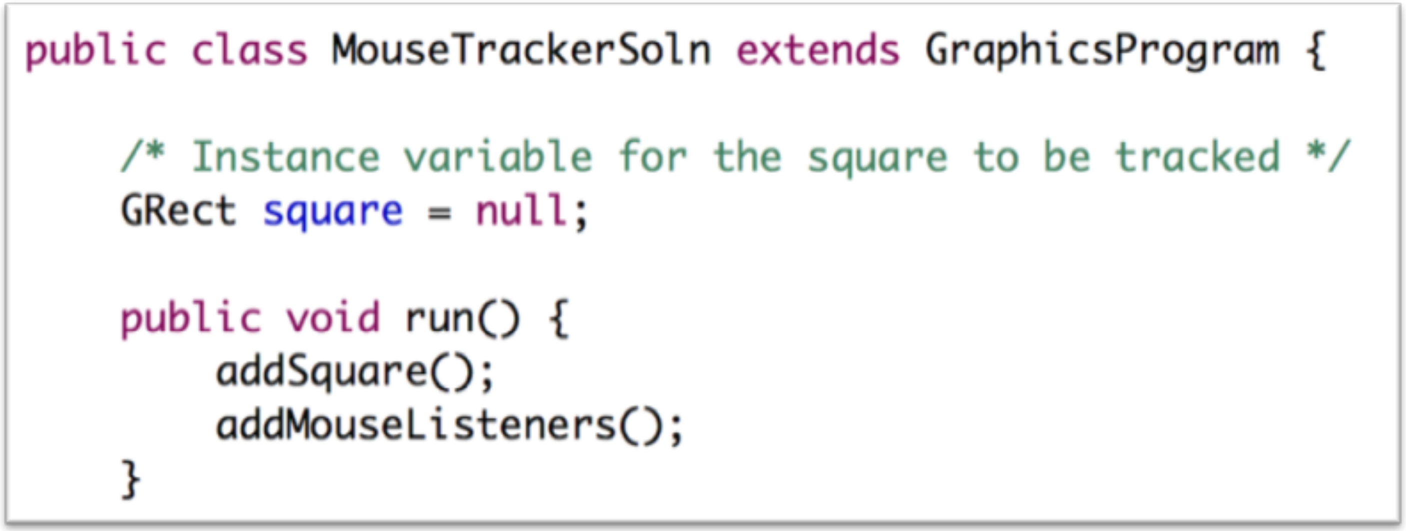


Mouse Tracker



Instance Variables

1. Variables exist until their inner-most control block ends.
2. If a variable is defined outside all methods, its inner-most control block is the entire program!
3. We call these variables **instance variables**

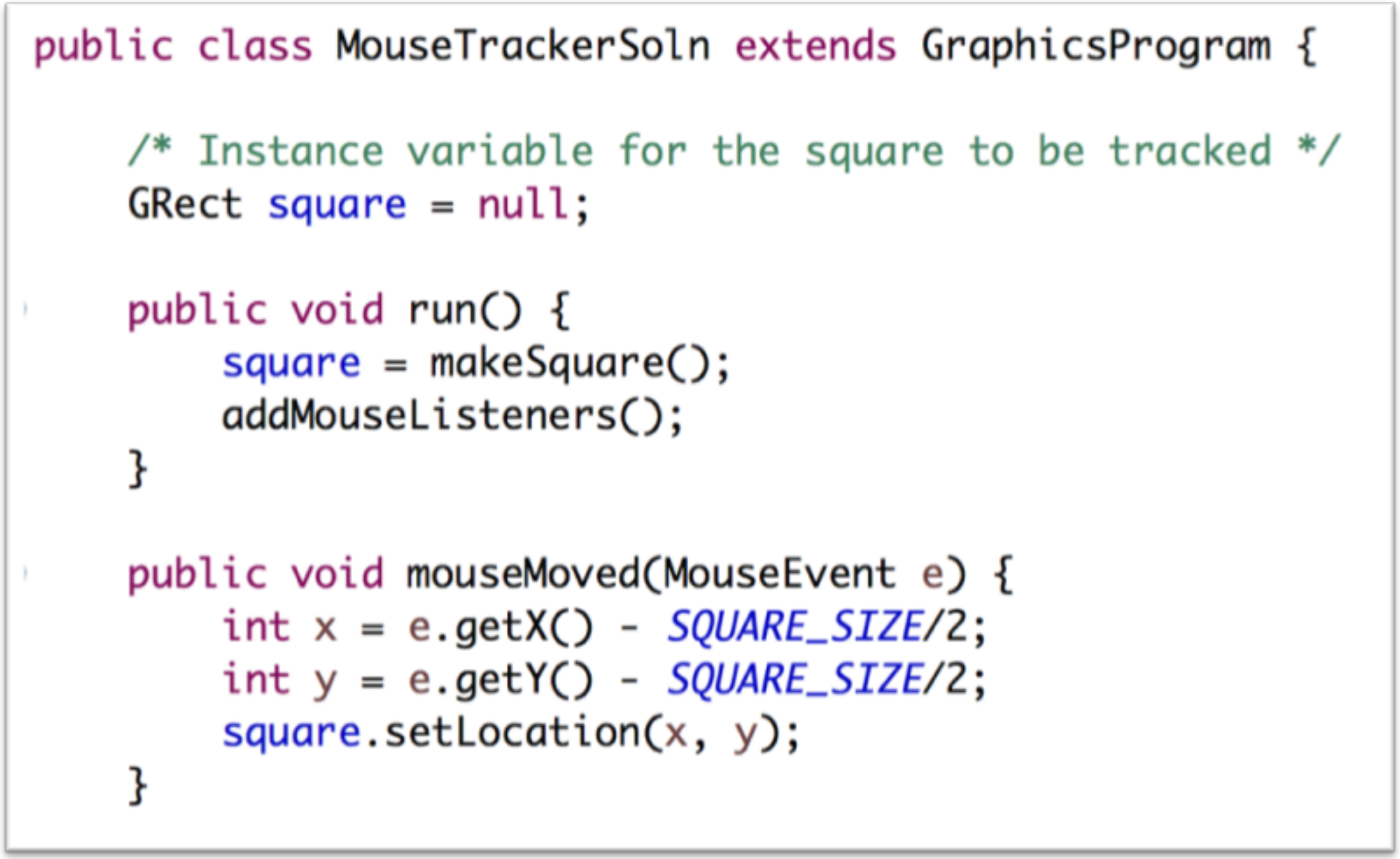


```
public class MouseTrackerSoln extends GraphicsProgram {

    /* Instance variable for the square to be tracked */
    GRect square = null;

    public void run() {
        addSquare();
        addMouseListeners();
    }
```

* Instance variables have special meanings in programs with multiple files. For now you need to know that all methods can see them and that their initialization line is executed before run.

Instance Variables + Events

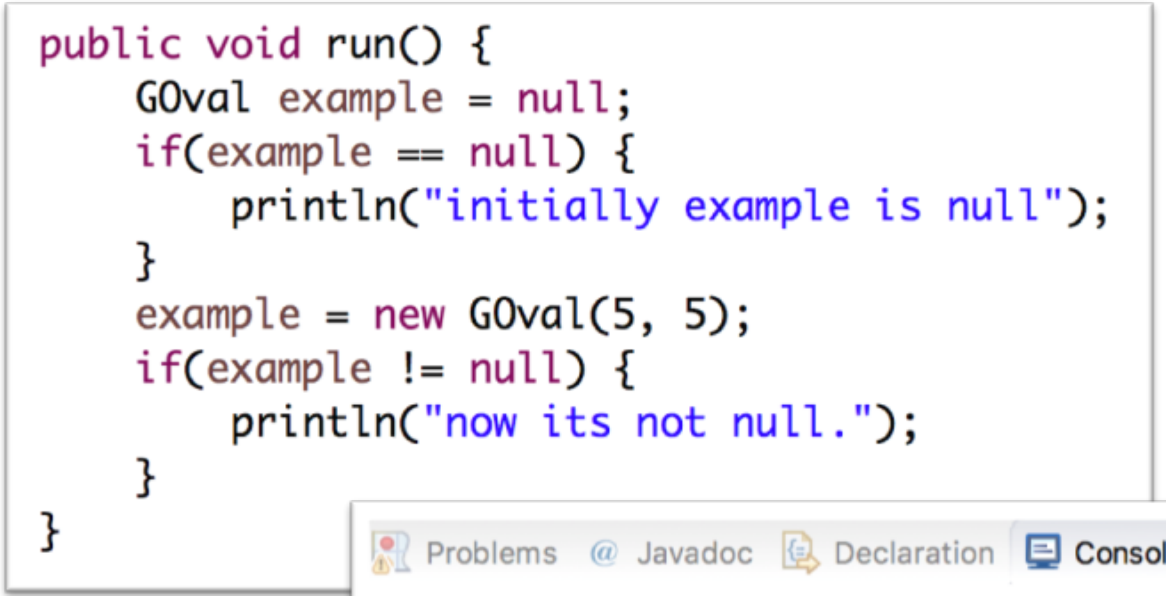
Often you need instance variables to pass information between the run method and the mouse event methods!



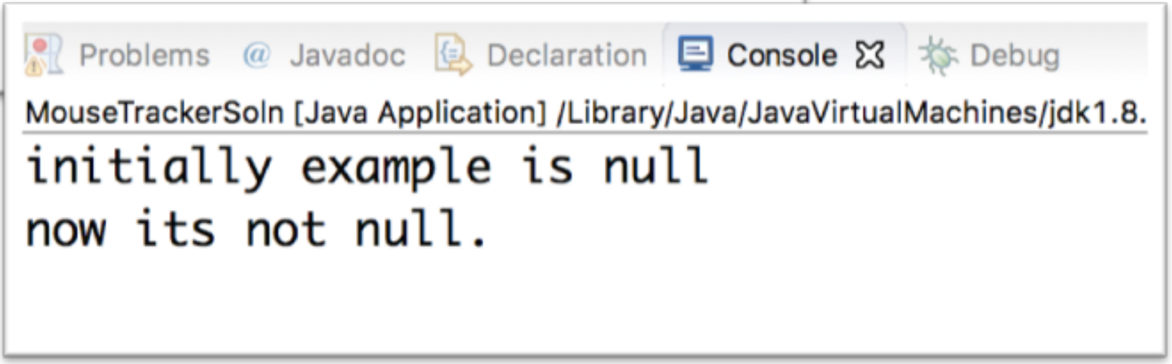
```
public class MouseTrackerSoln extends GraphicsProgram {

    /* Instance variable for the square to be tracked */
    GRect square = null;

    public void run() {
        square = makeSquare();
        addMouseListeners();
    }

    public void mouseMoved(MouseEvent e) {
        int x = e.getX() - SQUARE_SIZE/2;
        int y = e.getY() - SQUARE_SIZE/2;
        square.setLocation(x, y);
    }
```

Null

Objects have a special value called **null** which means this variable is not associated with a value yet.



```
public void run() {
    GOval example = null;
    if(example == null) {
        println("initially example is null");
    }
    example = new GOval(5, 5);
    if(example != null) {
        println("now its not null.");
    }
}
```



```
MouseTrackerSoln [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.
initially example is null
now its not null.
```

Null

GObjects returned by getElementAt might be null!

```
// may be a GObject, or null if nothing at (x, y)
GObject maybeAnObject = getElementAt(x, y);
if (maybeAnObject != null) {
   // do something with maybeAnObject
} else {
   // null – nothing at that location
}
```

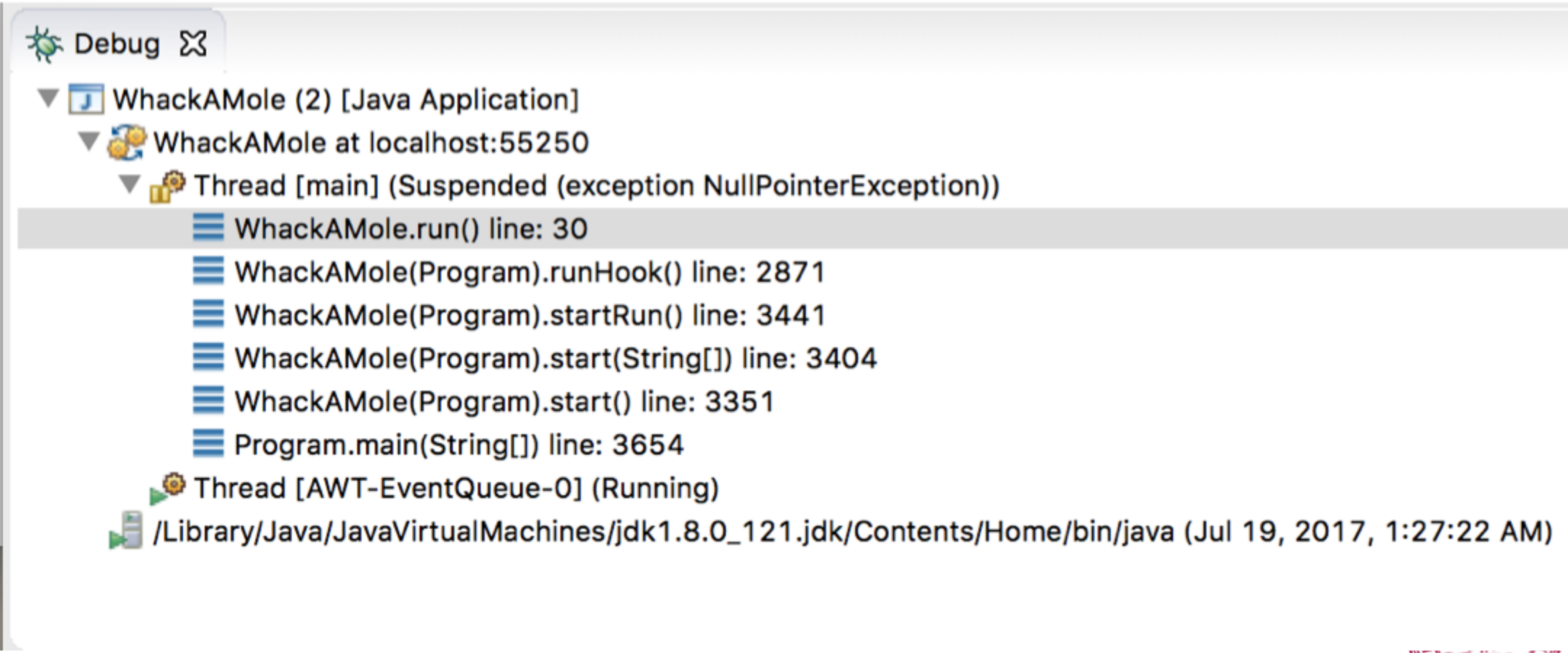
Null

Calling methods on an object that is **null** will crash your program!

```
// may be a GObject, or null if nothing at (x, y)
GObject maybeAnObject = getElementAt(x, y);
if (maybeAnObject != null) {
   int x = maybeAnObject.getX(); // OK
} else {
   int x = maybeAnObject.getX(); // CRASH!
}
```

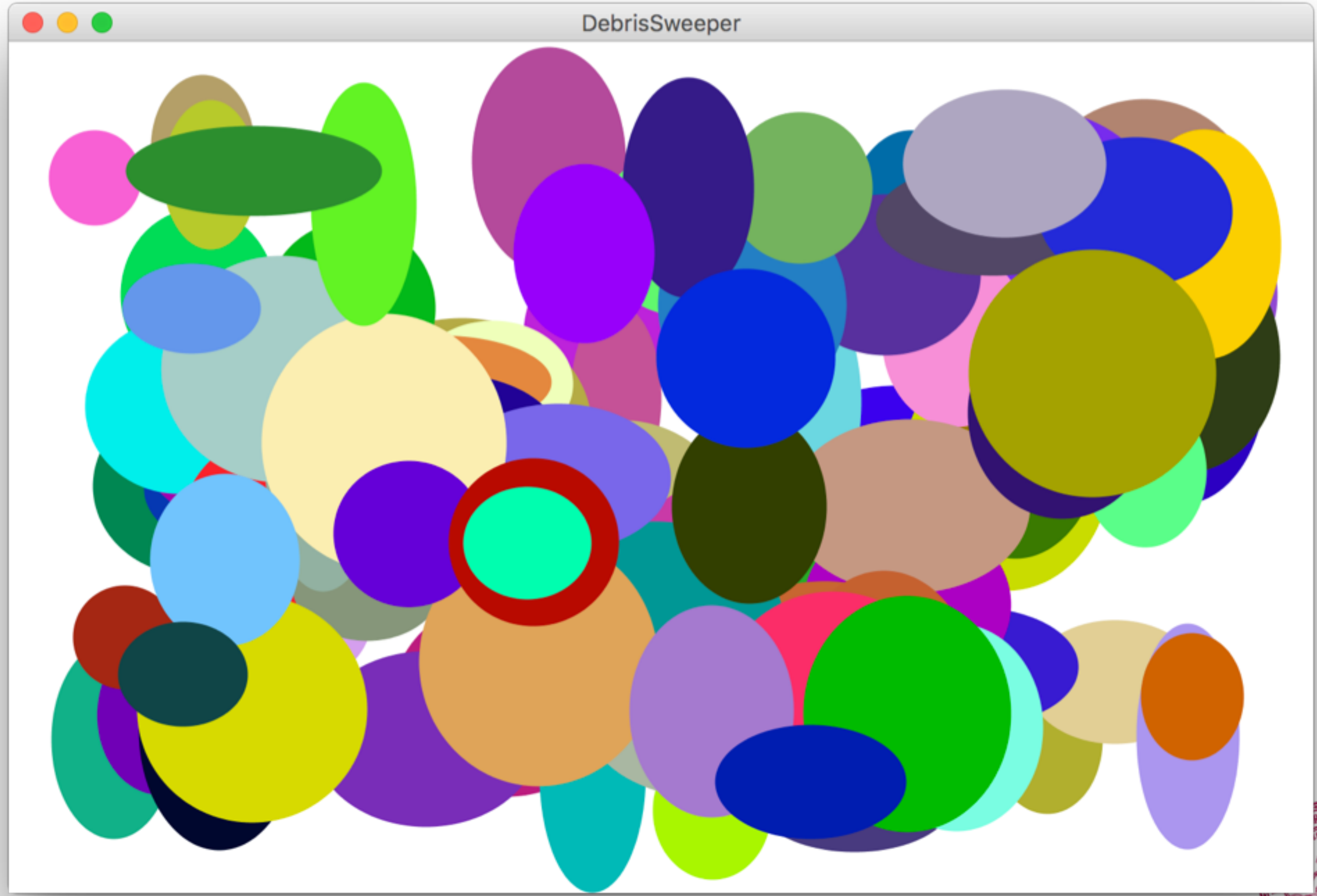
Null

Calling methods on an object that is **null** will crash your program! (throws a NullPointerException)



```
Debug
WhackAMole (2) [Java Application]
  WhackAMole at localhost:55250
    Thread [main] (Suspended (exception NullPointerException))
      WhackAMole.run() line: 30
      WhackAMole(Program).runHook() line: 2871
      WhackAMole(Program).startRun() line: 3441
      WhackAMole(Program).start(String[]) line: 3404
      WhackAMole(Program).start() line: 3351
      Program.main(String[]) line: 3654
    Thread [AWT-EventQueue-0] (Running)
  /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (Jul 19, 2017, 1:27:22 AM)
```

Debris Sweeper



New Concepts

New Commands

- `addMouseListeners();`
- `getElementAt(`*`x`*`, `*`y`*`);`
- `remove(`*`obj`*`);`

New Ideas

- The Listener Model
- Instance Variables
- **`null`**

Responding to Mouse Events

1. The **run** method should call **addMouseListeners**
2. Write definitions of any listener methods needed

mouseClicked(*e*)	Called when the user clicks the mouse
mousePressed(*e*)	Called when the mouse button is pressed
mouseReleased(*e*)	Called when the mouse button is released
mouseMoved(*e*)	Called when the user moves the mouse
mouseDragged(*e*)	Called when the mouse is dragged with the button down

The parameter *e* is **MouseEvent** object, which provides more data about event, such as the location of mouse.

Responding to Keyboard Events

1. The **run** method should call **addKeyListeners**
2. Write definitions of any listener methods needed

keyPressed(*e***)**	Called when the user presses a key
keyReleased(*e***)**	Called when the key comes back up
keyTyped(*e***)**	Called when the user types (presses and releases) a key

The parameter *e* is a **KeyEvent** object, which indicates which key is involved.

And Here We Are…

Catch Me If You Can?