Events and Fields

Nick Troccoli

Reading:
Art & Science of Java, Ch. 10 & 6

Learning Goals

- Know how to respond to mouse events in **`GraphicsPrograms`**
- Know how to use *fields* to store information outside of methods

Plan for today

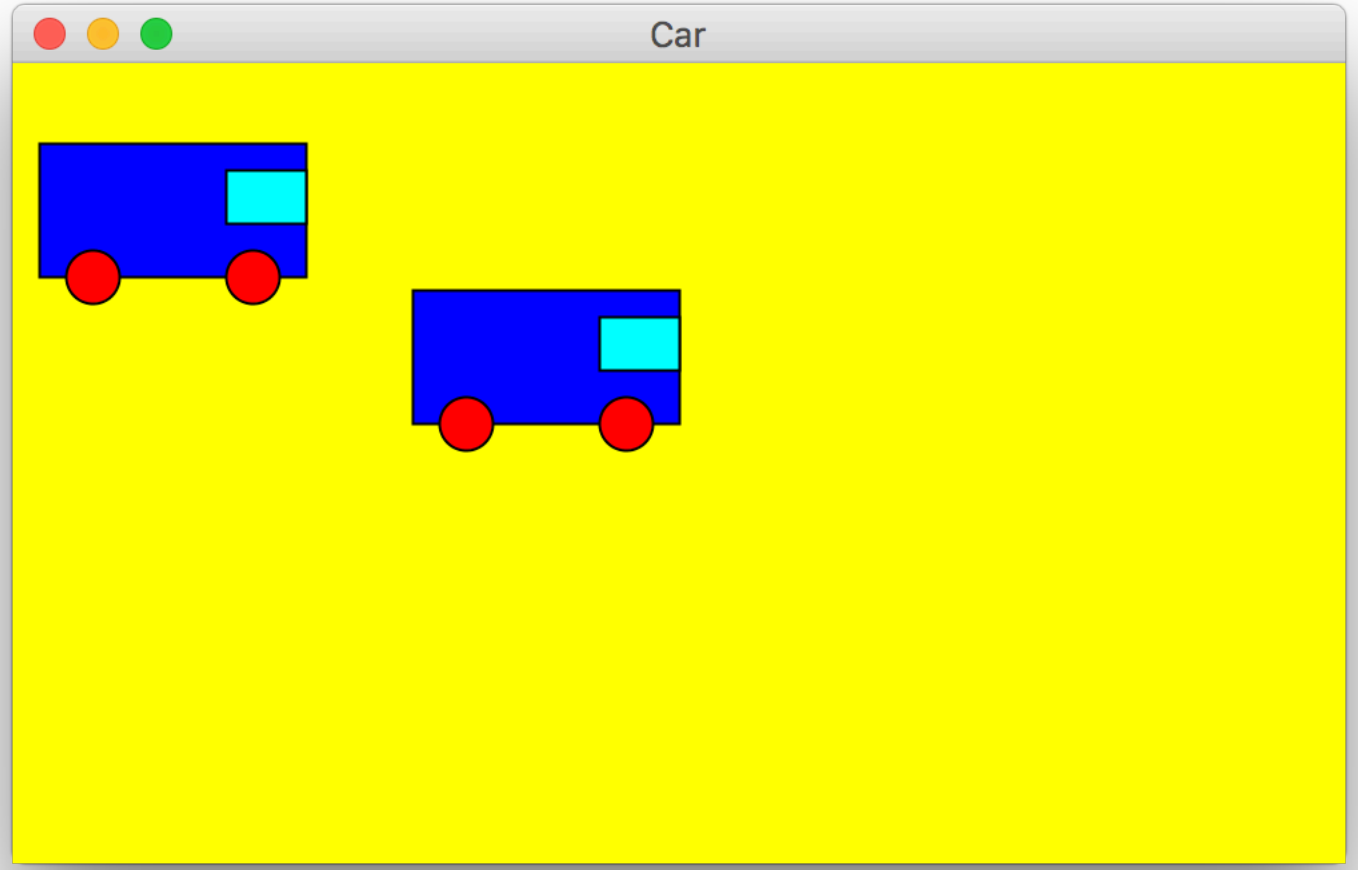
- Graphics review
- Event-driven programming
- Announcements
- Fields

Plan for today

- Graphics review
- Event-driven programming
- Announcements
- Fields

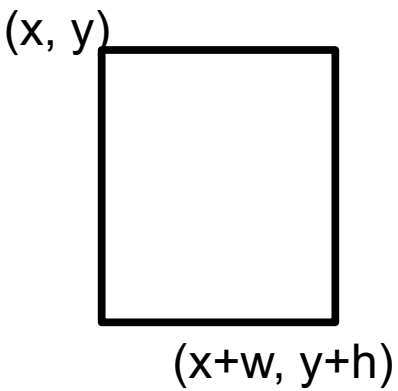
Last time: Graphics

- extending **GraphicsProgram** lets you add GObjects to a 2D canvas

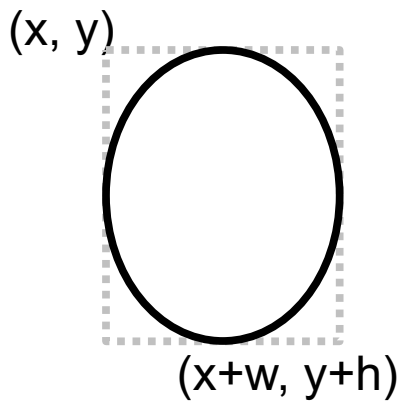


Types of GObjects

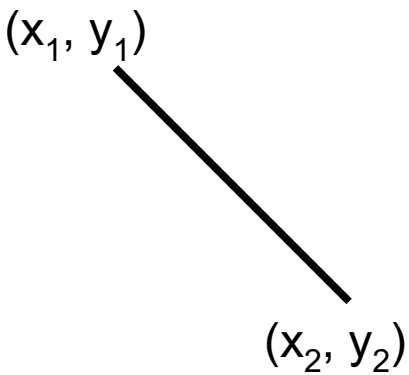
GRect



GOval



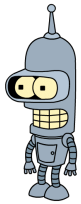
GLine



GLabel

Hello there!

GImage

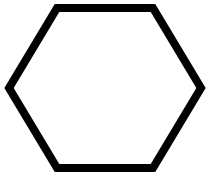


GArc

G3DRect

GRoundRect

GPolygon



Creating GObjects

Graphical object	Description
`new GImage("`***filename***`", `***x***`, `***y***`)`	image from the given file, drawn at *(x, y)*
`new GLabel("`***text***`", `***x***`, `***y***`)`	text with bottom-left at *(x, y)*
`new GLine(`***x1***`, `***y1***`, `***x2***`, `***y2***`)`	line between points (*x1*, *y1*), (*x2*, *y2*)
`new GOval(`***x***`, `***y***`, `***w***`, `***h***`)`	largest oval that fits in a box of size *w* * *h* with top-left at (*x*, *y*)
`new GRect(`***x***`, `***y***`, `***w***`, `***h***`)`	rectangle of size *w* * *h* with top-left at (*x*, *y*)

- for others, see:
  - http://cs.stanford.edu/people/eroberts/jtf/javadoc/student/

Using GObjects

- All graphical objects have these methods inside them (and more):

Method	Description
obj`.move(`***dx, dy***`)`	adjusts location by the given amount
obj`.setBackground(`***Color***`)`	sets overall window's background color
obj`.setFilled(`***boolean***`)`	whether to fill the shape with color
obj`.setFillColor(`***Color***`)`	what color to fill the shape with
obj`.setColor(`***Color***`)`	what color to outline the shape with
obj`.setLocation(`***x, y***`)`	change the object's x/y position
obj`.setSize(`***w, h***`)`	change the objects width*height size

Graphics example

```
// Create a 100x100 GRect at (50, 50)
GRect rect = new GRect(50, 50, 100, 100);

// Set some properties
rect.setFilled(true);
rect.setColor(Color.RED);

// Add to the canvas
add(rect);
```

Graphics example

```
// Create a 100x100 GRect at (50, 50)
GRect rect = new GRect(50, 50, 100, 100);

// Set some properties
rect.setFilled(true);
rect.setColor(Color.RED);

// Add to the canvas
add(rect);
```

Graphics example

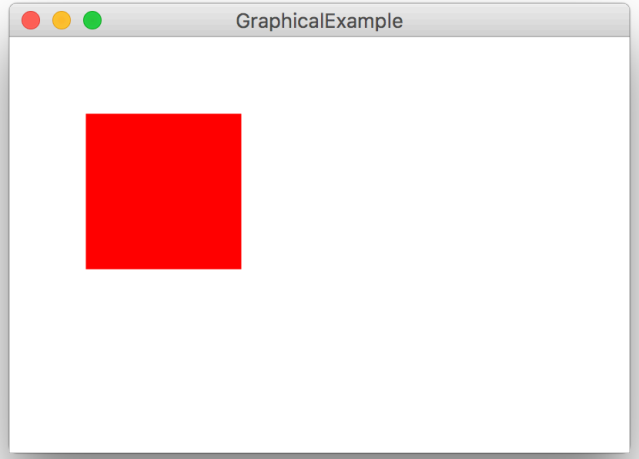
```
// Create a 100x100 GRect at (50, 50)
GRect rect = new GRect(50, 50, 100, 100);

// Set some properties
rect.setFilled(true);
rect.setColor(Color.RED);

// Add to the canvas
add(rect);
```

Graphics example

```
// Create a 100x100 GRect at (50, 50)
GRect rect = new GRect(50, 50, 100, 100);

// Set some properties
rect.setFilled(true);
rect.setColor(Color.RED);

// Add to the canvas
add(rect);
```

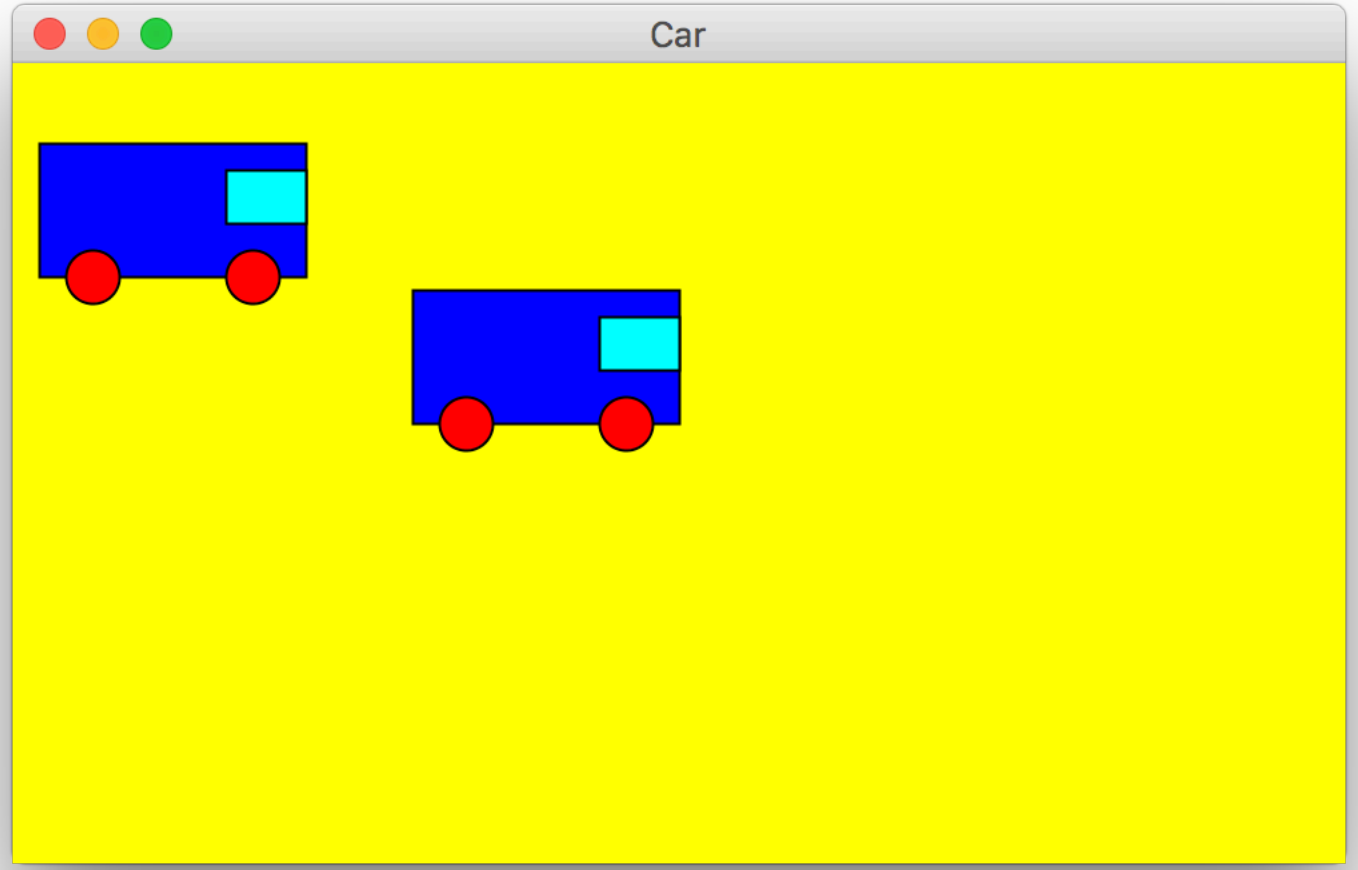
Graphics example

```
// Create a 100x100 GRect at (50, 50)
GRect rect = new GRect(50, 50, 100, 100);

// Set some properties
rect.setFilled(true);
rect.setColor(Color.RED);

// Add to the canvas
add(rect);
```



Wrap-up: Graphics

- Everything we've learned about methods, parameters, loops, if/else, etc. applies to graphics too!



Plan for today

- Graphics review
- Event-driven programming
- Announcements
- Fields

Events

- Program launches

Events

- Program launches
- Mouse motion
- Mouse clicking
- Keyboard keys pressed
- Device rotated
- Device moved
- GPS location changed
- and more…

Events

- Program launches
- Mouse motion
- Mouse clicking
- Keyboard keys pressed
- Device rotated
- Device moved
- GPS location changed
- and more…

Events

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

Events

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

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

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
}
```

Example: ClickForFace

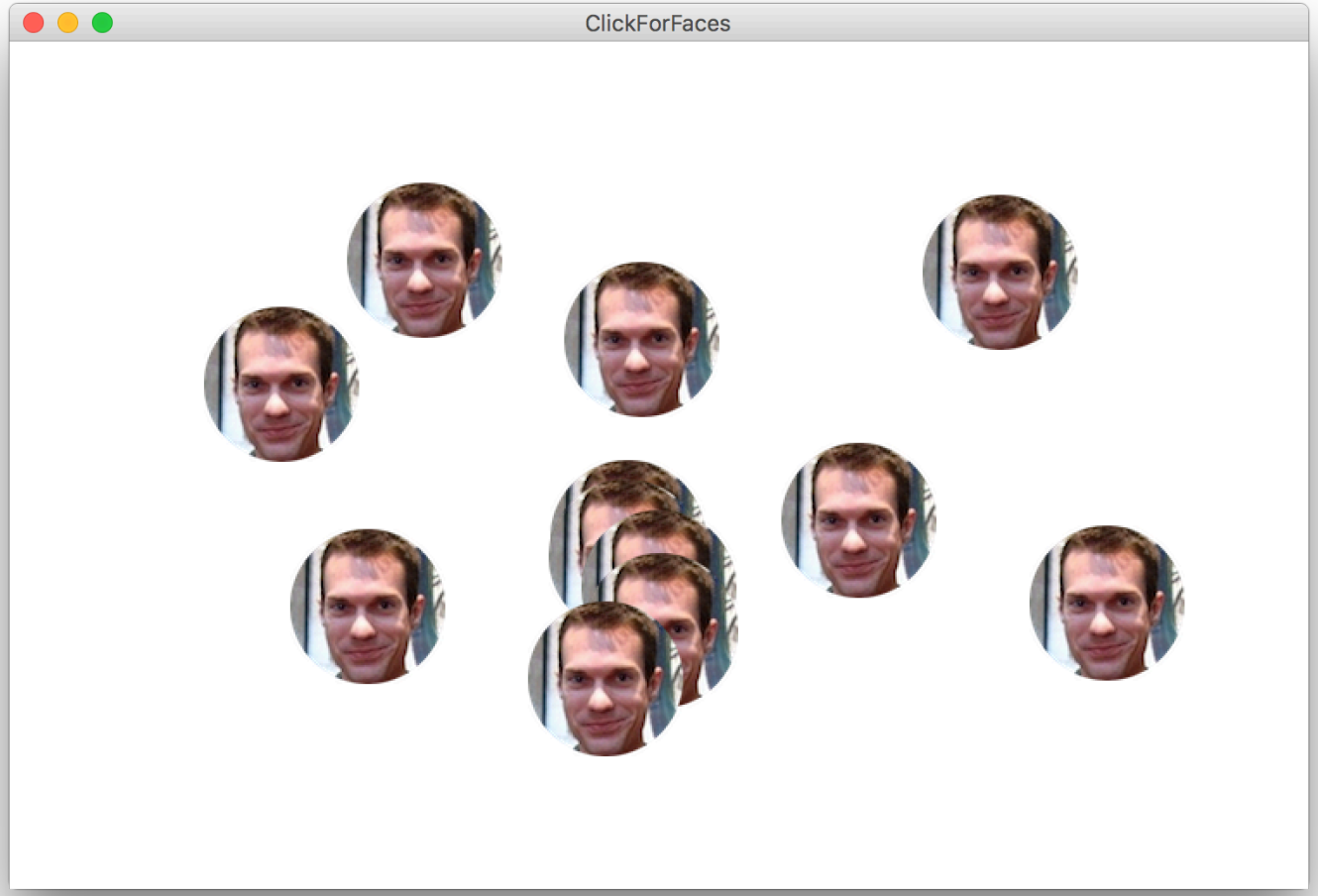
```
import acm.program.*;
import acm.graphics.*;
import java.awt.*;
import java.awt.event.*;    // NEW

public class ClickForFace extends GraphicsProgram {

    // Add a face at 50, 50 on mouse click
    public void mouseClicked(MouseEvent event) {
        GImage face = new GImage("res/martyFace.png",
            50, 50);
        add(face);
    }
}
```

MouseEvent objects

- A `MouseEvent` contains information about the event that just occurred:

Method	Description
***e*.`getX()`**	the *x*-coordinate of mouse cursor in the window
***e*.`getY()`**	the *y*-coordinate of mouse cursor in the window

Example: ClickForFaces



Example: ClickForFaces

```
public class ClickForFaces extends GraphicsProgram {

    // Add a face at where the user clicks
    public void mouseClicked(MouseEvent event) {
        // Get information about the event
        double mouseX = event.getX();
        double mouseY = event.getY();

        // Add a face at the mouse location
        GImage face = new GImage("res/martyFace.png",
            mouseX, mouseY);
        add(face);
    }
}
```

Example: ClickForFaces

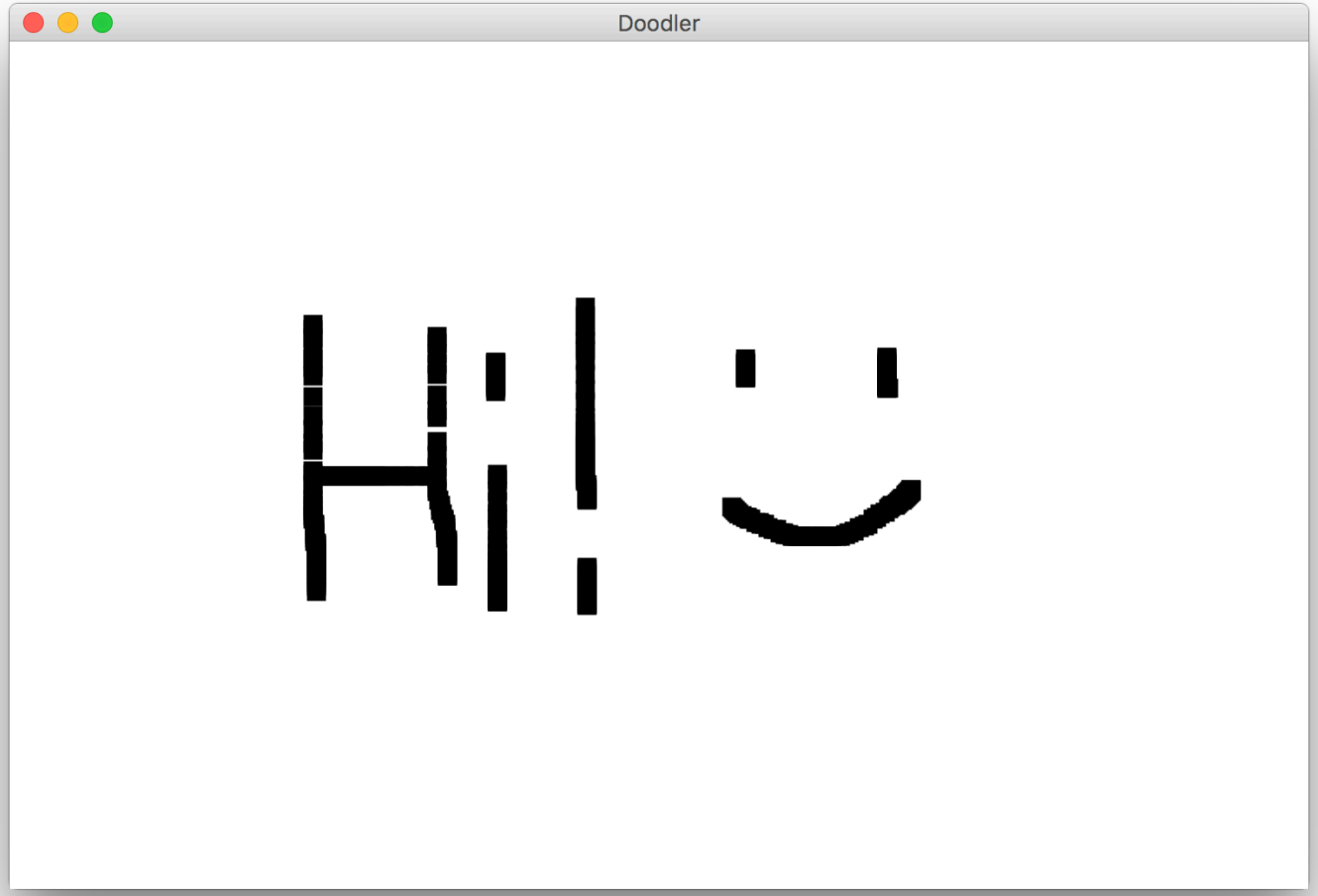
```
public class ClickForFaces extends GraphicsProgram {

    // Add a face at where the user clicks
    public void mouseClicked(MouseEvent event) {
        // Get information about the event
        double mouseX = event.getX();
        double mouseY = event.getY();

        // Add a face at the mouse location
        GImage face = new GImage("res/martyFace.png",
            mouseX, mouseY);
        add(face);
    }
}
```

Event methods

- There are many different types of mouse events.
  - Each takes the form:
    `public void` ***`eventMethodName`***`(MouseEvent event) { ...`

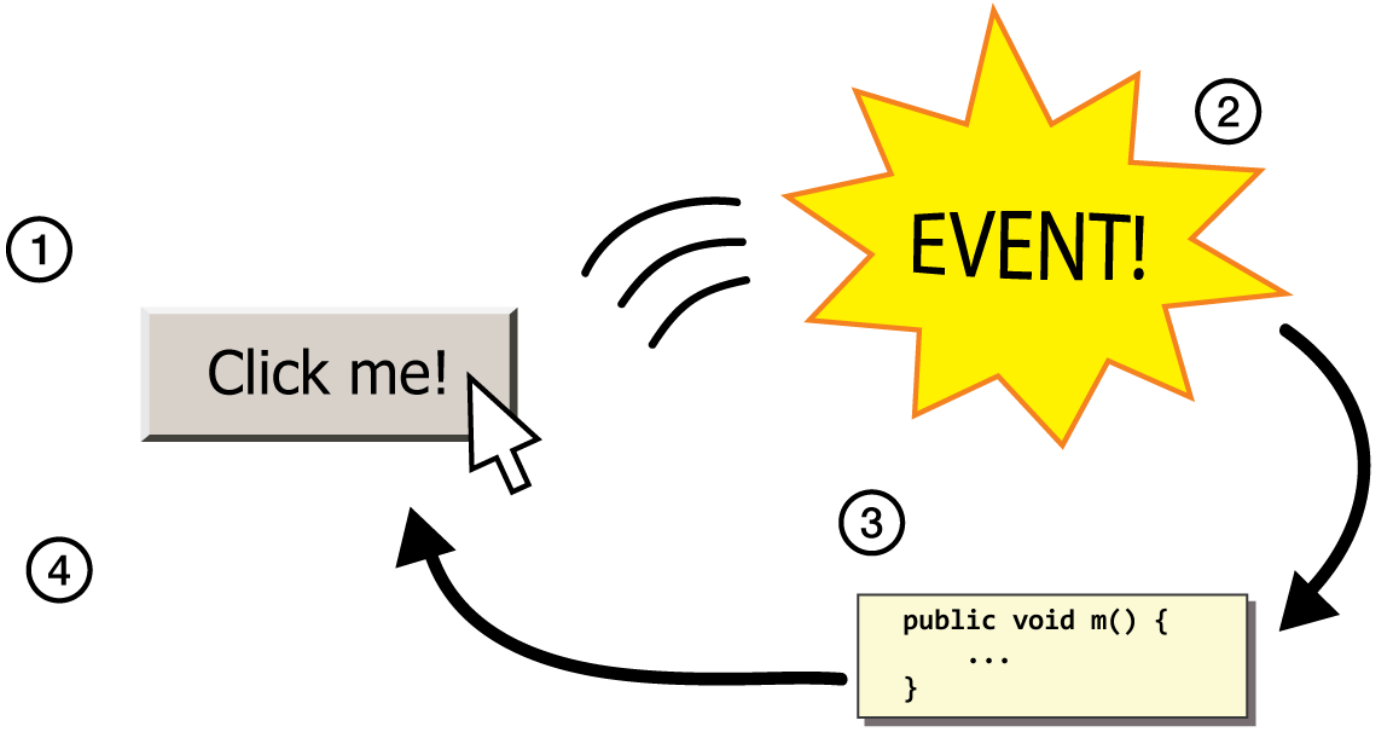
Method	Description
`mouseMoved`	mouse cursor moves
`mouseDragged`	mouse cursor moves while button is held down
`mousePressed`	mouse button is pressed down
`mouseReleased`	mouse button is lifted up
`mouseClicked`	mouse button is pressed and then released
`mouseEntered`	mouse cursor enters your program's window
`mouseExited`	mouse cursor leaves your program's window

Example: Doodler



The event cycle

1) User performs some action, like moving / clicking the mouse.
2) This causes an event to occur.
3) Java executes a particular method to handle that event.
4) The method's code updates the screen appearance in some way.



Plan for today

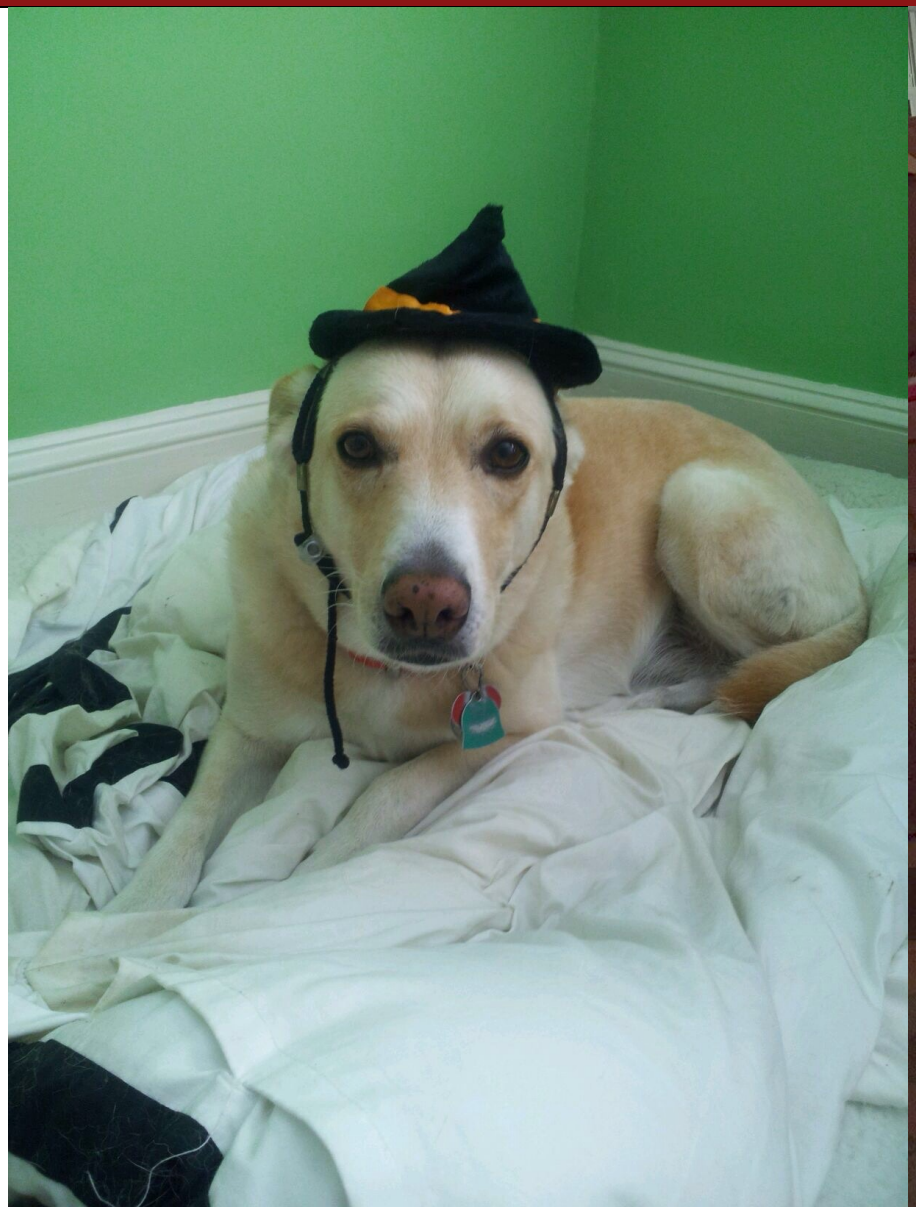
- Graphics review
- Event-driven programming
- Announcements
- Fields

Announcements!

- Assignment 3 (Hangman) is due Monday 5/1
- Hi ProFros!
- Puppy (kind of) pictures
- A moment of reflection

Meet Daisy

Daisy is festive!



Revisiting Doodler

```
public void mouseDragged(MouseEvent event) {
    double mouseX = event.getX();
    double mouseY = event.getY();
    double rectX = mouseX – SIZE / 2.0;
    double rectY = mouseY – SIZE / 2.0;
    GRect rect = new GRect(rectX, rectY, SIZE,
        SIZE);
    rect.setFilled(true);
    add(rect);
}
```

What if we wanted the *same* GRect to track the mouse, instead of making a new one each time?

Plan for today

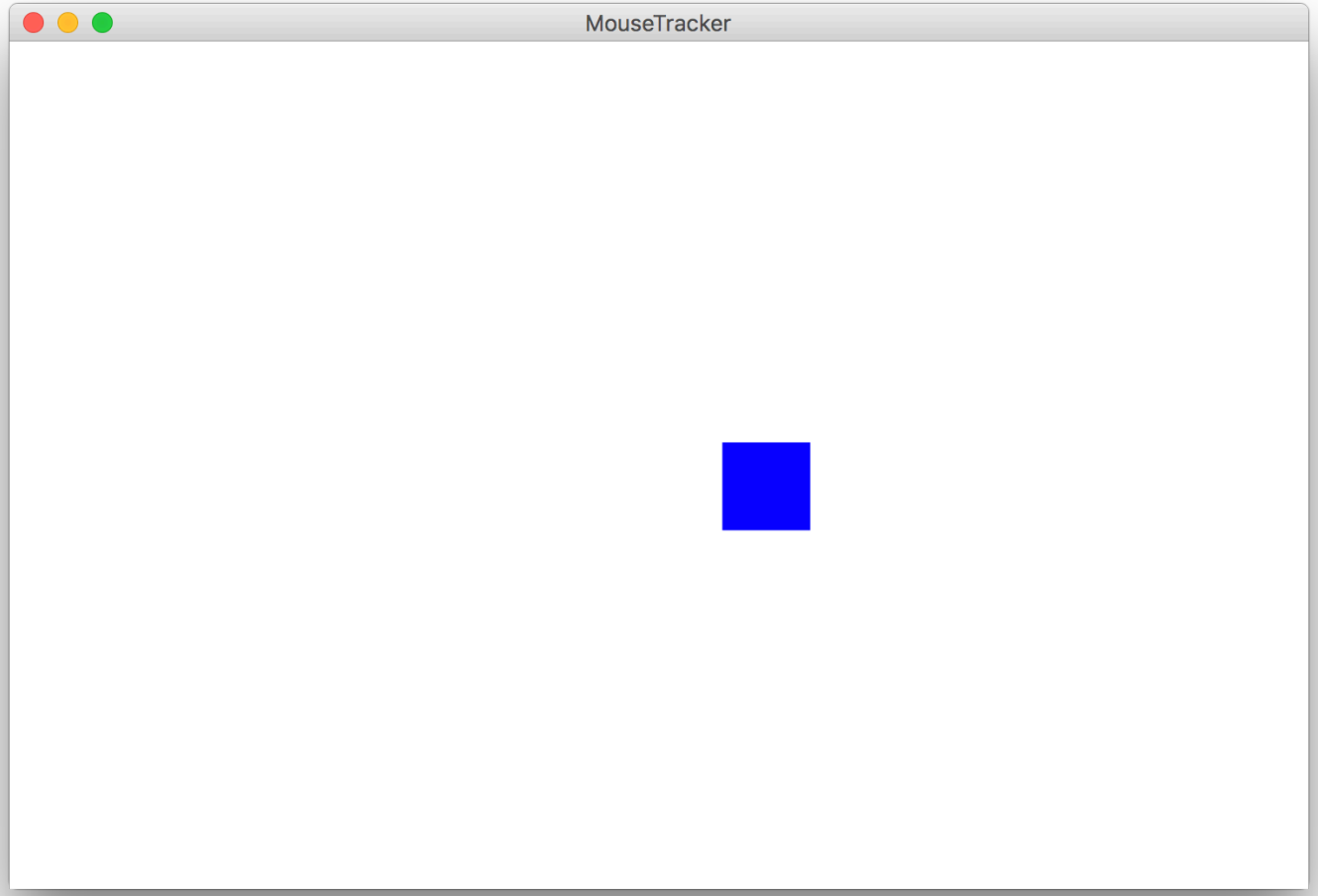
- Graphics review
- Event-driven programming
- Announcements
- Fields

Fields

```
private type name;  // declared outside of any method
```

- **field**: A variable that lives outside of any method.
  - The *scope* of a field is throughout an entire file (class).
  - Useful for data that must persist throughout the program, or that cannot be stored as local variables or parameters (event handlers).
  - *Overuse of fields:* Because fields have a large scope, it is considered bad style to use too many fields, or to make something a field that could instead be a local variable, parameter, return, etc.
    - **<u>DO NOT USE FIELDS ON HANGMAN!!</u>**

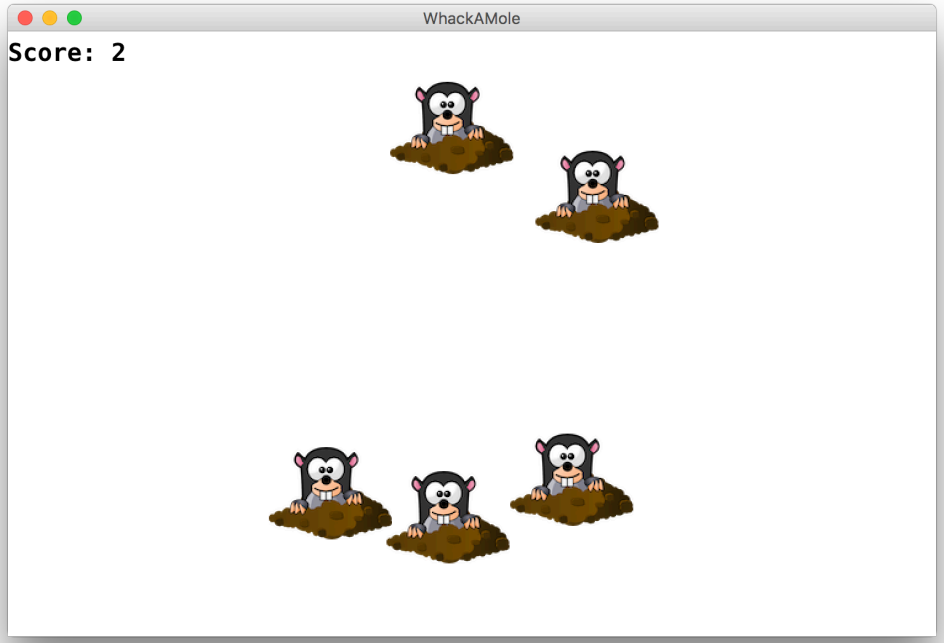
Example: MouseTracker



Putting it all together

Whack-A-Mole

- Let's use fields and mouse events to make Whack-A-Mole!
  - A mole should appear every second at a random location
  - If the user clicks a mole, remove it and increase their score by 1
  - There should be a GLabel in the left corner showing their score



Example: Whack-A-Mole

Demo

Exception

- If the user clicks an area with no mole, the program crashes.
  - A program crash in Java is called an **exception**.
  - When you get an exception, Eclipse shows red error text.
  - The error text shows the line number where the error occurred.
  - Why did this error happen?
  - How can we avoid this?

Problems | Javadoc | Declaration | Console

```
WhackAMole [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (Apr 27, 2017, 10:46:49 PM)
Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException
        at acm.graphics.GObjectList.remove(GContainer.java:187)
        at acm.graphics.GCanvas.remove(GCanvas.java:518)
        at acm.program.GraphicsProgram.remove(GraphicsProgram.java:215)
        at WhackAMole.mouseClicked(WhackAMole.java:52)
        at java.awt.AWTEventMulticaster.mouseClicked(AWTEventMulticaster.java:270)
        at java.awt.Component.processMouseEvent(Component.java:6536)
        at javax.swing.JComponent.processMouseEvent(JComponent.java:3324)
        at java.awt.Component.processEvent(Component.java:6298)
```

Null

- **`null`**: A special constant value meaning, "no object."
  - `getElementAt` returns `null` if no object is at that position.
  - You can check for `null` using the == and != operators.

```
GObject mole = getElementAt(x, y);
if (mole != null) {
    remove(mole);
}
```

Recap

- Graphics review
- Event-driven programming
- Announcements
- Fields