

CSSE 220 Day 16

Event Based Programming

Check out *EventBasedProgramming* from SVN

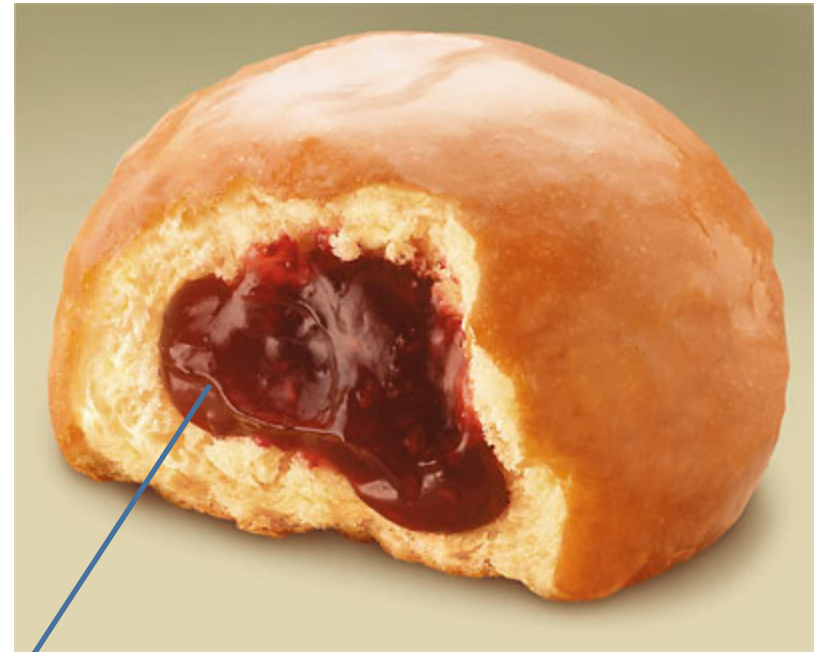
Finish the sentence

Using interfaces can help reduce _____
between classes.

1. Coupling
2. Cohesion
3. Encapsulation
4. Polymorphism

Graphical User Interfaces in Java

- We say what to draw
- Java windowing library:
 - Draws it
 - Gets user input
 - **Calls back** to us with events
- We **handle** events



Hmm, donuts

Goocy

Handling Events

- Many kinds of events:
 - Mouse pressed, mouse released, mouse moved, mouse clicked, button clicked, key pressed, menu item selected, ...
- We create **event listener objects**
 - that implement the right **interface**
 - that handle the event as we wish
- We **register** our listener with an **event source**
 - Sources: buttons, menu items, graphics area, ...

Using Inner Classes

- Classes can be defined **inside** other classes or methods
- Used for “smallish” helper classes
- Example: **Ellipse2D.Double**



- Often used for **ActionListeners...**

Anonymous Classes

- Sometimes very small helper classes are only used once
 - This is a job for an anonymous class!
- **Anonymous** → no name
- A special case of inner classes
- Used for the simplest **ActionListeners...**

Inner Classes and Scope

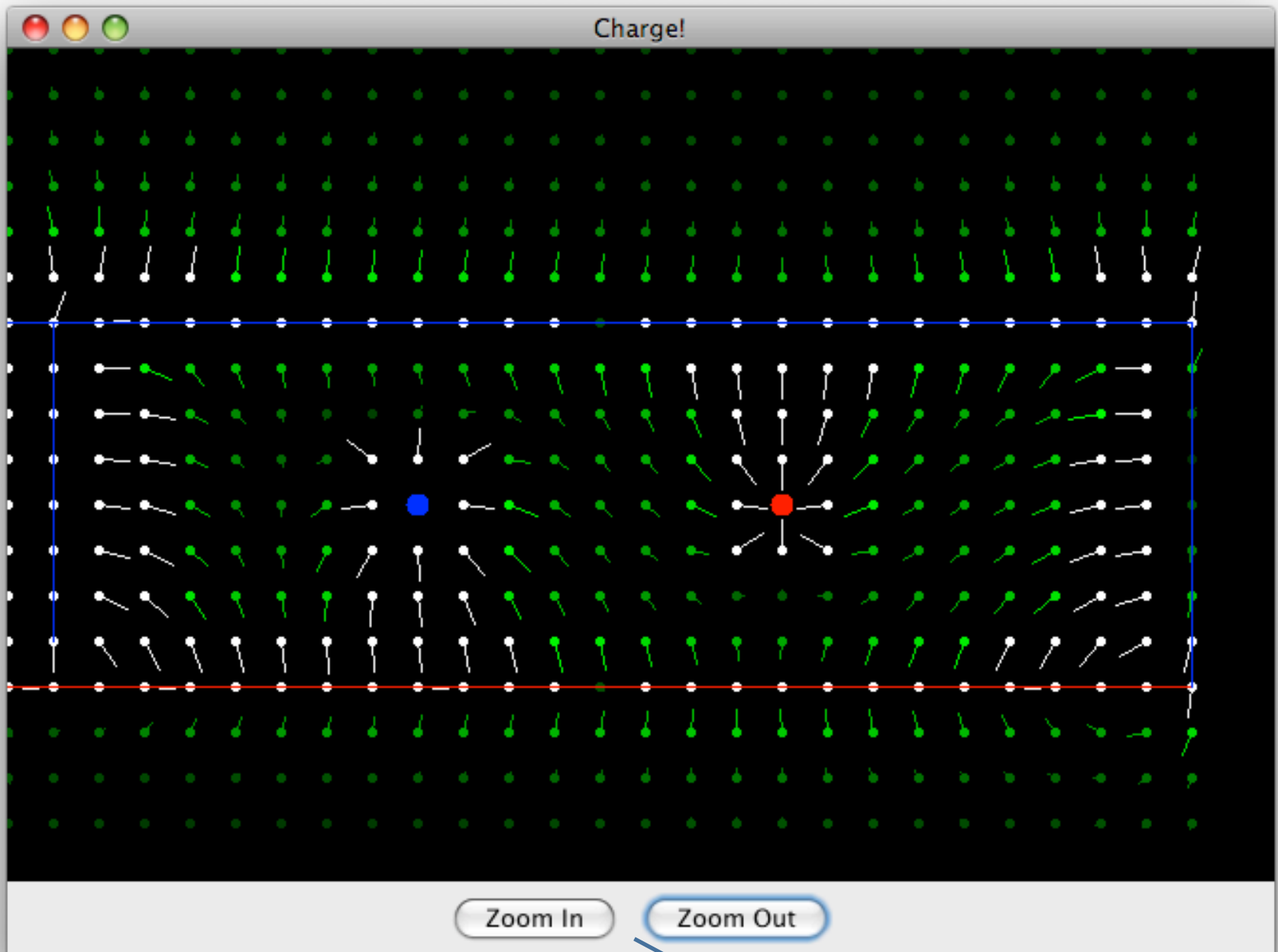
- Inner classes can access any variables in surrounding scope
- Caveats:
 - Local variables must be **final**
 - Can only use instance fields of surrounding scope if we're inside an instance method
- Example:
 - Prompt user for what porridge tastes like

Layout in Java windows

TIME TO MAKE THE BUTTONS

Key Layout Ideas

- JFrame's add(Component c) method
 - Adds a new component to be drawn
 - Throws out the old one!
- JFrame also has method add(Component c, Object constraint)
 - Typical constraints:
 - BorderLayout.NORTH, BorderLayout.CENTER
 - Can add one thing to each “direction”, plus center
- JPanel is a container (a thing!) that can display multiple components



So, how do we do this?

Repaint (and then no more)

- To update graphics:
 - We tell Java library that we need to be redrawn:
 - `space.repaint()`
 - Library calls `paintComponent()` when it's ready
- **Don't call `paintComponent()` yourself!
It's just there for Java's call back.**

Mouse Listeners



```
public interface MouseListener {  
    public void mouseClicked(MouseEvent e);  
    public void mouseEntered(MouseEvent e);  
    public void mouseExited(MouseEvent e);  
    public void mousePressed(MouseEvent e);  
    public void mouseReleased(MouseEvent e);  
}
```

Work Time

- LinearLightsOut
- Hint: you'll want a class with fields for all the various JButtons in the lights out project. That way after the window is created you can get and set the text of the buttons.