Interactors
Anatomy of a Window
Introducing Interactors

• An **interactor** is a widget that can be added to a window.

• The user can then interact with the program through the interactors.
Adding Interactors

• To use most interactors, you will need to import:
  ```java
  import acm.gui.*;
  import javax.swing.*;
  ```

• You can add an interactor to the appropriate part of the window by calling:
  ```java
  add(interactor, location);
  ```

• `location` can be NORTH, SOUTH, EAST, or WEST.
The Shocking Exposé
Structuring a Program

- **Inside `init`:**
  - Create interactors.
  - Add interactors to the program.
- **Inside `run`:**
  - Set up any graphics, state, etc.
  - Run the program.
Slider Controls

• The JSlider control lets the user visually choose from a range of integers.

• Constructor:
  ```java
  new JSlider(min, max, initial)
  ```

• To construct a vertical slider bar:
  ```java
  new JSlider(SwingConstants.VERTICAL, min, max, initial)
  ```
Time-Out for Announcements!
CS Casual Dinner

• Second biquarterly CS Casual Dinner for Women in Computer Science is tonight at 6PM in Gates 519.

• Everyone is welcome; highly recommended!

• Keith's office hours shortened to 4:30PM – 6:00PM tonight.
Second Midterm Exam

- Second midterm exam one week from today: **Wednesday, March 5** from 7PM – 10PM.
- Topics covered: up through and including today's lecture on interactors.
- Review session: **Sunday, March 2** from 1PM – 3PM in Hewlett 200.
- Alternate exam requests due at 3:15PM today.
  - Contact us *immediately* if you need to take an alternate exam and haven't done so yet.
  - We'll email back information on the alternate exam by tomorrow night.
Assignment 5

• Assignment 5 due Friday.
• Questions?
  • Stop by the LaIR!
  • Ask on QuestionHut!
  • Email your section leader!
  • Stop by Vikas's or Keith's office hours!
Back to CS106A!
Buttons

- The `JButton` type represents a button.
- You can create one using

  ```java
  new JButton(label)
  ```
Responding to Commands

• As with mouse events, responding to interactor events requires two steps.
• Tell Java that you want to respond to commands by calling
  ```java
  addActionListeners();
  ```
• Respond to events by writing a method
  ```java
  public void actionPerformed(ActionEvent e)
  ```
Determining the Cause

- You can tell where an `ActionEvent` came from in one of two ways:
  - Calling `e.getActionCommand()`, which returns a string containing the name of the source.
    - Most common use case: the name of the `JButton` that was clicked.
  - Calling `e.getSource()`, which returns a reference to the interactor that caused the event.
Text Input

• Three common text input controls:
  • **JTextField**
    • Takes in any text as input.
  • **IntField**
    • Only accepts *int* values; will prompt if you give bad data.
  • **DoubleField**
    • Only accepts *double* values; will prompt if you give bad data.
Responding to Text

- If the user presses ENTER or RETURN in a text box, you will not automatically be notified of this.
- One way to get notification:
  ```java
text.addActionListener(this);
```
- Can then use `e.getSource()` to find the text box.
- Once you've done the above, you can also
  ```java
text.setActionCommand(command-string);
```
- Can then use `e.getActionCommand()` to find the text box.