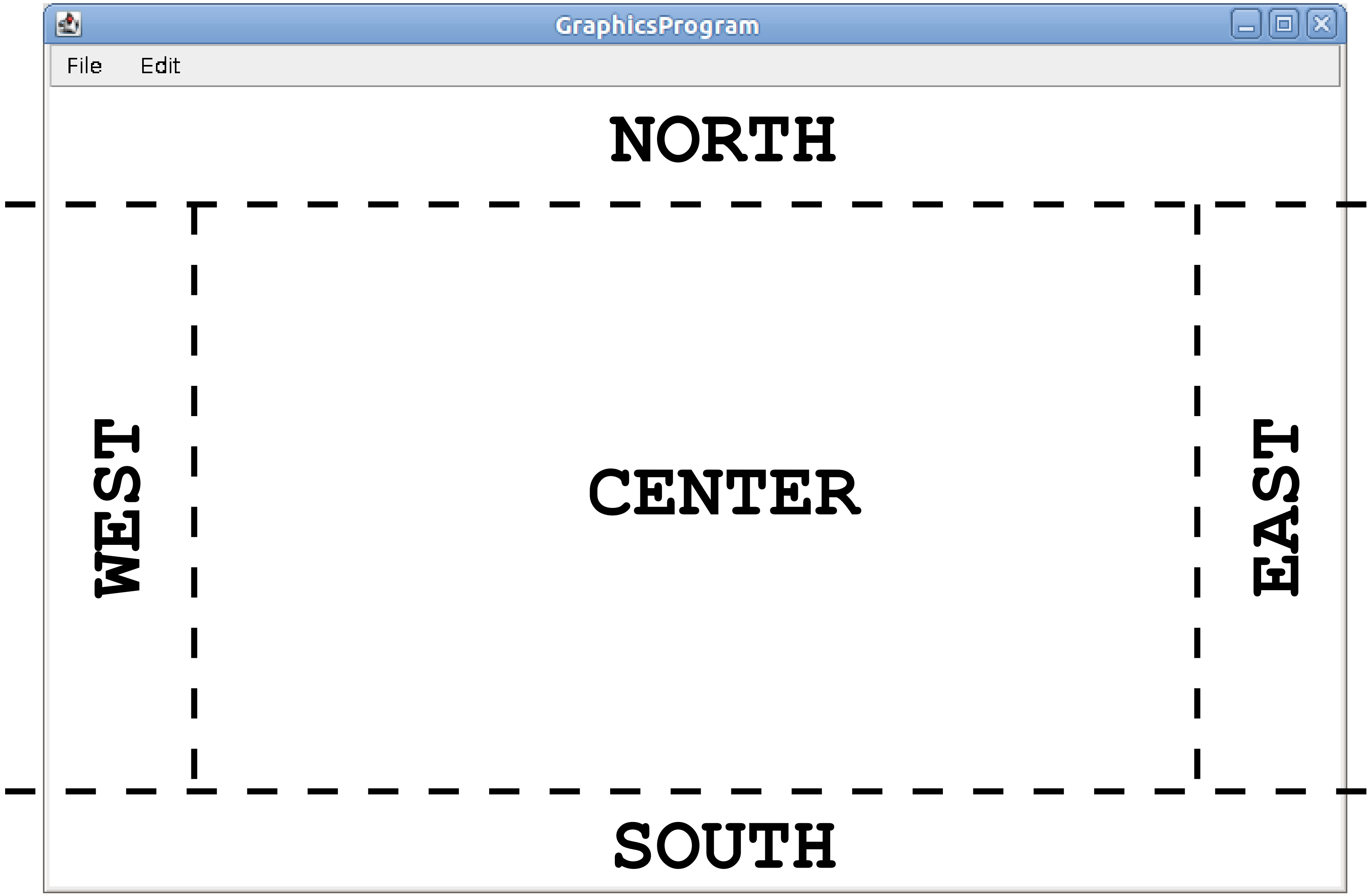
Interactors

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 javax.swing.*;
```

- You can add an interactor to the appropriate part of the window by calling

```
add(interactor, location);
```

- ***location*** can be `NORTH`, `SOUTH`, `EAST`, or `WEST`.
    - (If you forget the location, the interactor you add will swallow up half the window.)

The Shocking Exposé

Structuring a Program

- When your program starts up, *before it calls* `run`, it calls a method named `init` tasked with setting up interactors.
- 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.
- You can construct a new slider that ranges over the integer values in the range [***min***, ***max***] with the specified initial value by writing

```
new JSlider(min, max, initial)
```

- You can then read the value on the slider by calling

```
slider.getValue()
```

`JLabels`

- You can add descriptive labels to the sides of the world by using the `JLabel` type.
- The user can't really interact with a `JLabel`, but it still counts as an interactor.
- You can create `JLabels` by writing

```
new Jlabel(text)
```

Time-Out for Announcements!

Announcements

- Assignment 6 is due a week from today.
  - ***Recommendation:*** Complete Steganography by Monday and start working on Histogram Equalization.
- Second midterm exam is Tuesday, March 3 from 7PM – 10PM.
  - More details next week.
  - Need to take the exam at an alternate time? Contact me by next Tuesday!

Back to CS106A!

Working with Buttons

- Pushbuttons are one of the most common types of interactors.
- There are three steps to setting up a program that works with buttons:
  - Create and add the buttons to the display in the `init` method.
  - Tell Java that you want to listen in to button events.
  - Write a button handler to respond to those events.

Creating Buttons

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

```
new JButton(label)
```

Responding to Commands

- As with mouse events, responding to interactor events requires two steps.
- First, tell Java that you want to respond to commands by calling

```
addActionListeners();
```

  after you've added your buttons.
- Then, respond to events by writing a method

```
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.
  - More on that later.

Text Input

- You can get text input from the user by using the `JTextField` interactor.
- You can construct a `JTextField` by writing

  `new JTextField(`***numColumns***`)`

  where ***numColumns*** controls the displayed width of the text field.
- You can then call

  field`.getText()`

- to get the text from the field.

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:

```
text.addActionListener(this);
```

- Can then use `e.getSource()` to find the text box.
- Once you've done the above, you can also

```
text.setActionCommand(command-string);
```

- Can then use `e.getActionCommand()` to find the text box.