Interactors Recap

Main Interactors:

JButtons

```java
Button button = new JButton("Press me");
```

Press Me!

JTextFields

```java
TextField field = new JTextField(10);
add(field, SOUTH);

... field.getText(); // returns string in field
```

Type something!

JLabels

```java
Label label = new JLabel("Hi");
```

Hi

Basically just a Label :(
Regions

4 of them!

Center is usually canvas, sides are for interactors
# JButton Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>new JButton(&quot;text&quot;)</td>
<td>Creates new button with given text string</td>
</tr>
<tr>
<td>jb.getBackground()</td>
<td>get or set background color on button</td>
</tr>
<tr>
<td>jb.setBackground(color);</td>
<td></td>
</tr>
<tr>
<td>jb.isEnabled()</td>
<td>get or set whether button is clickable</td>
</tr>
<tr>
<td>jb.setEnabled(boolean);</td>
<td></td>
</tr>
<tr>
<td>jb.getFont()</td>
<td>get or set text font used for button text</td>
</tr>
<tr>
<td>jb.setFont(font);</td>
<td></td>
</tr>
<tr>
<td>jb.getForeground()</td>
<td>get or set text color on button</td>
</tr>
<tr>
<td>jb.setForeground(color);</td>
<td></td>
</tr>
<tr>
<td>jb.getIcon()</td>
<td>get or set icon image showing on button</td>
</tr>
<tr>
<td>jb.setIcon(icon);</td>
<td></td>
</tr>
<tr>
<td>jb.getText()</td>
<td>set or return text showing on the button</td>
</tr>
<tr>
<td>jb.setText(&quot;text&quot;);</td>
<td></td>
</tr>
</tbody>
</table>
### JTextField Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>new JTextField(&quot;text&quot;)</td>
<td>Create new text field of given size</td>
</tr>
<tr>
<td>new JTextField(columns)</td>
<td></td>
</tr>
<tr>
<td>jtf.addActionListener(this);</td>
<td>causes action events to occur when the user presses Enter on the field</td>
</tr>
<tr>
<td>jtf.getActionCommand()</td>
<td>set/return a string to identify the action events that will occur in this field</td>
</tr>
<tr>
<td>jtf.setActionCommand(&quot;cmd&quot;);</td>
<td></td>
</tr>
<tr>
<td>jtf.getText()</td>
<td>set/return text in the field</td>
</tr>
<tr>
<td>jtf.setText(&quot;text&quot;);</td>
<td></td>
</tr>
</tbody>
</table>
## JLabels Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>new JLabel(&quot;text&quot;)</code></td>
<td>Create new label with given text</td>
</tr>
<tr>
<td><code>jl.getFont()</code></td>
<td>get/set text font used for label text</td>
</tr>
<tr>
<td><code>jl.setFont(font);</code></td>
<td>get or set text color on label</td>
</tr>
<tr>
<td><code>jl.setForeground()</code></td>
<td>set or return horizontal alignment of text in the label; pass</td>
</tr>
<tr>
<td><code>jl.setForeground(color);</code></td>
<td>JLabel.LEFT, CENTER, or RIGHT</td>
</tr>
<tr>
<td><code>jl.getHorizontalAlignment()</code></td>
<td>set/return text in the label</td>
</tr>
<tr>
<td><code>jl.setText(&quot;text&quot;);</code></td>
<td></td>
</tr>
</tbody>
</table>
How to initialize Interactors

```java
public void init() {
    JButton button = new JButton("Graph");
    add(button, NORTH);

    JTextField textField = new JTextField(20);
    textField.addActionListener(this);
    add(textField, NORTH);

    addActionListeners(); // Listen for all button clicks
}
```
Adding Interactors

Order Matters!

- Since I added the JButton first, it appears first from left to right
public void actionPerformed(ActionEvent e) {
    String command = e.getActionCommand(); //return a string of what is clicked
    Object obj = e.getSource();
    if (command.equals("Graph")) {
        // do whatever you want Graph to do
    } else if (command.equals("something else")) {
        // do something else
    } else if (obj == textField) {
        String text = textField.getText() //returns whatever is typed in textField
        // do something with what is entered in textField
    }
}
Classes Recap
Classes = **Blueprints**

3 important things:

- What information does this variable store? *Instance Variables*
- What can you do with this variable type? *Methods*
- How do you create this variable type? *Constructor*
Example: GOval

What information does this class need to store?
Example: GOval

What information does this class need to store?

```java
private double x;
private double y;
```
Example: G Oval

What information does this class need to store?

```java
private double x;
private double y;
private double width;
private double height;
```
Example: GOval

What information does this class need to store?

```java
private double x;
private double y;
private boolean filled;
private double width;
private double height;
private Color color;
```

e tc, etc, etc...
Example: GOval

What can you do with this variable type?

- `getFilled()`
- `setFilled(true)`
Example: G Oval

What can you do with this variable type?

- `getFilled()`
- `getColor()`
- `setFilled(true)`
- `setColor(Color.RED)`
Example: GOval

What can you do with this variable type?

**getters**
- `getFilled()`
- `getColor()`

**setters**
- `setFilled(true)`
- `setColor(Color.RED)`
Example: GOval

What can you do with this variable type?

**getters**
- getFilled()
- getColor()
- getWidth()
- getHeight()
- ...

**setters**
- setFilled(true)
- setColor(Color.RED)
- setWidth(true)
- setHeight(true)
- ...


Example: GOval

How does the constructor create a GOval?
Example: GOval

How does the constructor **create** a GOval?

```java
GOval myOval = new GOval(0, 0, 30, 30)
```
Example: GOval

How does the constructor **create** a GOval?

```java
GOval myOval = new GOval(0, 0, 30, 30)
```

- `setX(0)`
- `setY(0)`
- `setWidth(30)`
- `setHeight(30)`
Milestones

Interactors

Class 1: NameSurferEntry

Class 2: NameSurferDatabase

DrawBackground

Put it all together! (redraw())
Assignment 6: NameSurfer

One of my personal favorites!
Interactors

Add a bunch of text boxes and buttons to the screen.
Milestone 1: Interactors

If “enter” is pressed in JTextBox, or JButton “Graph” pressed, should add Graph.

- Name is case-insensitive
- If no data found, don’t add line.
Interactors Testing

Use the fact that it starts out as a console program to your advantage!

Check that your interactors are working.
NameSurferEntry

Goal: create a custom variable that stores the information in one NameSurfer entry
Milestone 2: NameSurferEntry

NameSurferEntry: One entry in our NameSurfer database.

- NameSurferEntry(String line)

Name and popularity in each decade from 1900 to 2000.

“Sam 58 69 99 131 168 236 278 380 467 408 466”

Need to parse string and store information.
Milestone 2: NameSurferEntry

“Maya 0 0 0 0 0 0 0 835 646 244 114”

Edge case: 0’s mean a name wasn’t in the top 1000 names for that decade.

You will draw those at the bottom of the NameSurfer graph.
Classes = **Blueprints**

3 important things:

- What information does this variable store?  
  * **Instance Variables**

- What can you do with this variable type?  
  * **Methods**

- How do you create this variable type?  
  * **Constructor**
Class = NameSurferEntry

3 important things:

- What information does this variable store? ???
- What can you do with this variable type? getName, getRank, toString
- How do you create this variable type? NameSurferEntry(String line);
Class = **NameSurferEntry**

3 important things:

- What information does this variable store?  
  What data structure(s) will you use?

- What can you do with this variable type?  
  **getName, getRank, toString**

  public methods MUST stay the same!

- How do you create this variable type?  
  **NameSurferEntry(String line);**

  hint: str.split(" ");
NameSurferDatabase

Goal: store a collection of NameSurferEntries.
Milestone 3: NameSurferDatabase

Each line in text file contains info for one NameSurferEntry.

Read the file and store a collection of NameSurferEntries.

(data structure?)
Class = NameSurferDatabase

3 important things:

- What information does this variable store?  
- What can you do with this variable type?  
- How do you create this variable type?
  
  - findEntry(String name)
  
  - NameSurferDatabase(String filename);
Milestone 3: NameSurferDatabase

You **may** (and probably should) implement your own private helper methods.

However, you may **NOT** modify the public method headers.
Milestone 4: Drawing the Background

- 11 Buckets
- 10 Vertical GLines
Milestone 5: Bringing All Together

- Give the interactors their real functionality
- Find the NameSurferEntry for an input name and graph it
- Colors go: black, red, blue, magenta, black, red, blue...
Milestone 6: You insert here!

We give you datasets, you do something cool with them! (You can submit this to the graphics contest!)

See: Data Visualization Extension Handout
Common Pitfalls!

Off-by-One in Graphics

- 11 “Buckets”
  - 11 Vertical Lines
  - 11 GLabels
- 10 Connecting GLines
Common Pitfalls!

Can only use getHeight() AFTER the canvas has been added

- Not in constructor, not in init()

Not using constants!

- Use the constants we give you!
Common Pitfalls!!

Rank goes from 1 to 1000!

- Must handle edge case when getRank() is 0
  - (should graph to the bottom of the graph)
But first: Sandcastle!

Goal: get practice using collections.

(seem familiar?)

*cough cough* section 6 problem 7
But first: Sandcastle!

Goal: get practice using collections.

(seem familiar?)

*cough cough* section 6 problem 7