Course Name:
Advanced Java

Lecture 19
Topics to be covered

- Java Swing
  - Lists
  - Trees
  - Tables
  - Styled Text Components
  - Progress Indicators
  - Component Organizers

AWT to Swing

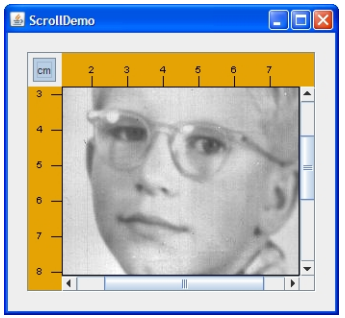
- AWT: Abstract Windowing Toolkit
  - import java.awt.*
- Swing: new with Java2
  - import javax.swing.*
  - Extends AWT
  - Tons o' new improved components
  - Standard dialog boxes
  - Look-and-feel
  - Event listeners

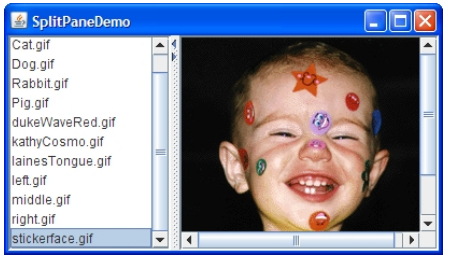
Top Level Containers: JFrame

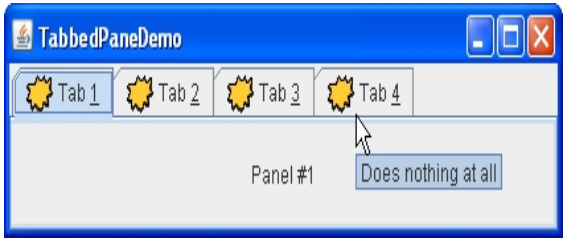
- **`javax.swing.JFrame`**:
  - Top-level window with a title and a border.
  - Usually used as a program's main window

Internal Containers

- Not Top level containers
- Can contain other non-top level components
- Examples:
  - `JScrollPane`: Provides a scrollable view of its components
  - `JSplitPane`: Separates two components
  - `JTabbedPane`: User chooses which component to see







Containers - Layout

- Each container has a layout manager
  - Determines the size, location of contained widgets.
- Setting the current layout of a container:
  `void setLayout(LayoutManager lm)`
- *`LayoutManager`* implementing classes:
  - `BorderLayout`
  - `BoxLayout`
  - `FlowLayout`
  - `GridLayout`

FlowLayout

- Components “flow” onto form left-to-right and top-to-bottom
- Components take on “normal” size

The Code

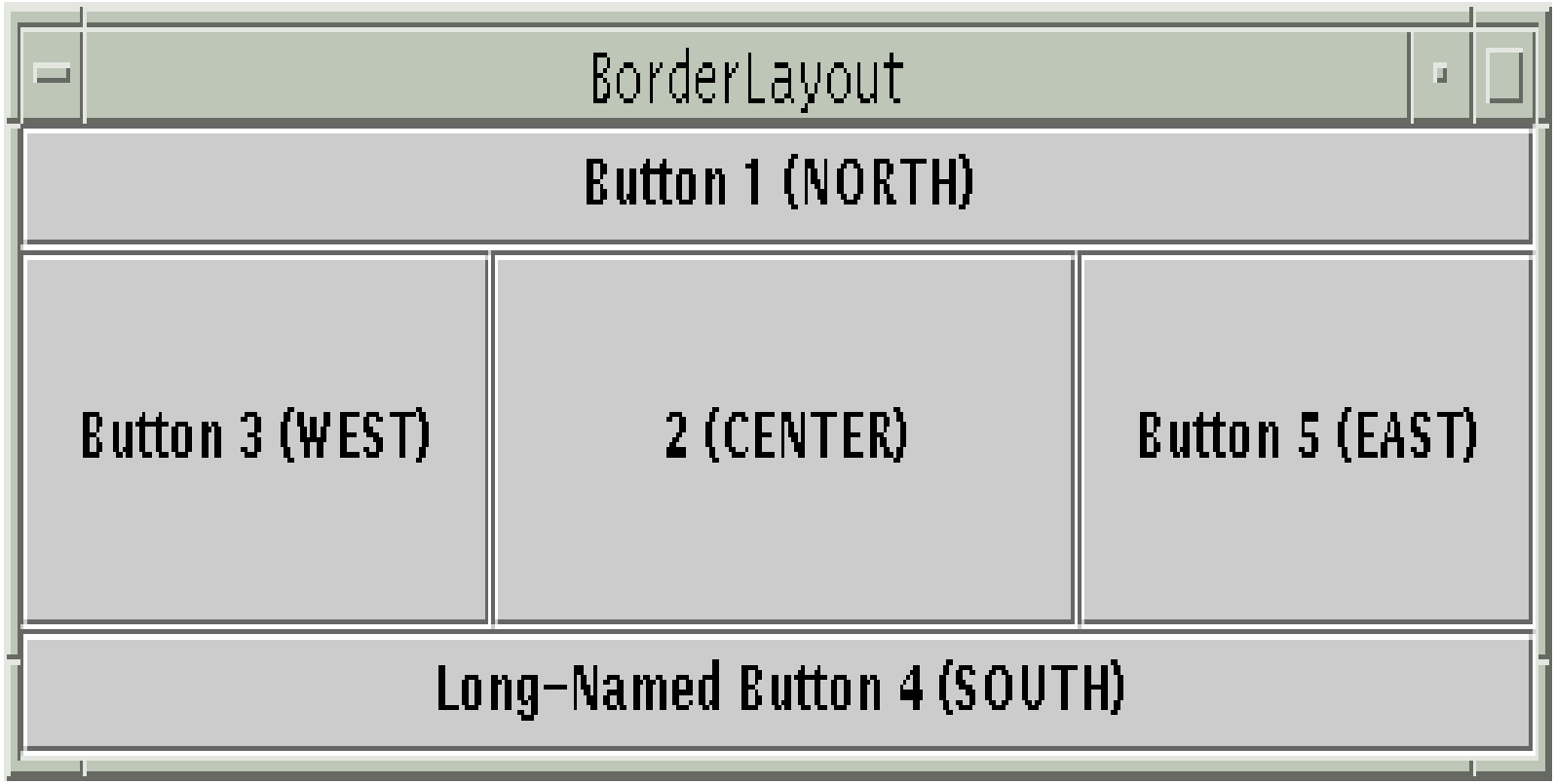
```
Container contentPane = getContentPane();

    contentPane.setLayout(new FlowLayout());
    contentPane.add(new JButton("Button 1"));
    contentPane.add(new JButton("2"));
    contentPane.add(new JButton("Button 3"));
    contentPane.add(new JButton("Long-
Named Button 4"));
    contentPane.add(new JButton("Button 5"));
```

The FlowLayout API

- Three constructors:
  - public FlowLayout()
  - public FlowLayout(int alignment)
  - public FlowLayout(int alignment, int horizontalGap, int verticalGap)

- The alignment argument must have one of the values :
  - FlowLayout.LEFT, FlowLayout.CENTER, FlowLayout.RIGHT.
- horizontalGap and verticalGap specify the number of pixels to put between components.
  - default gap value = 5 pixels.
- Properties:
  - Alignment, Hgap, Vgap: int, RW,

BorderLayout



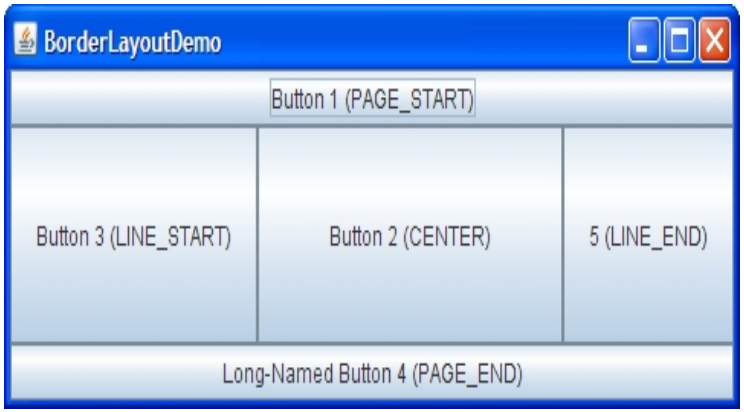
- Container divided into five regions: West, North, East, South, Center.

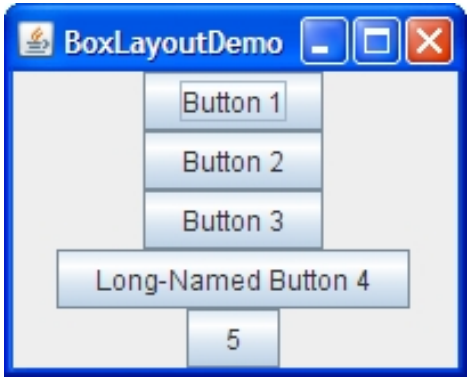
Example

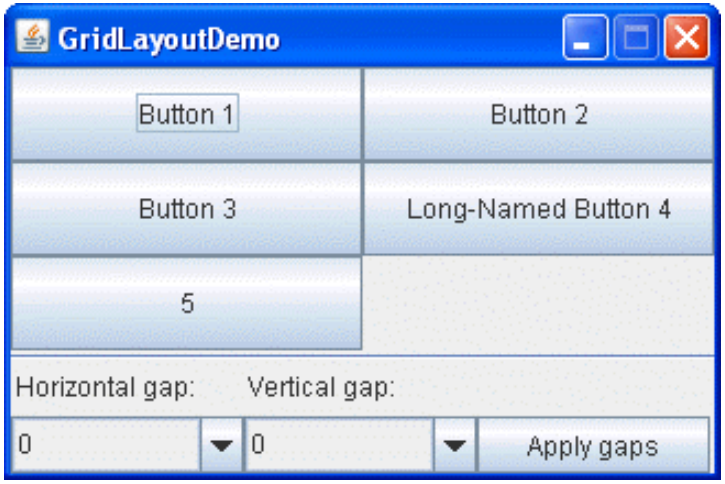
```
public class BorderLayout1 extends JApplet {
    public void init() {
      Container cp = getContentPane();
      cp.setLayout(new BorderLayout());  // default is
    FlowLayout
       cp.add(new JButton("North") , BorderLayout.NORTH);
//   cp.add(BorderLayout.NORTH, new JButton("North"));
    // also ok!
//  cp.add(new JButton("North"), “North”);   // also ok!
     cp.add(BorderLayout.SOUTH, new JButton("South"));
     cp.add(BorderLayout.EAST, new JButton("East"));
     cp.add(BorderLayout.WEST, new JButton("West"));
    cp.add(BorderLayout.CENTER, new JButton("Center"));
} }
```

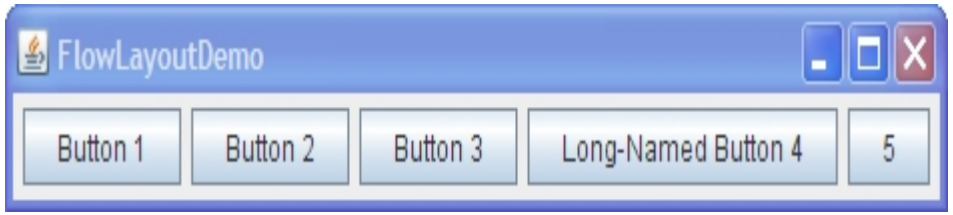
- Default for most things

Containers - Layout

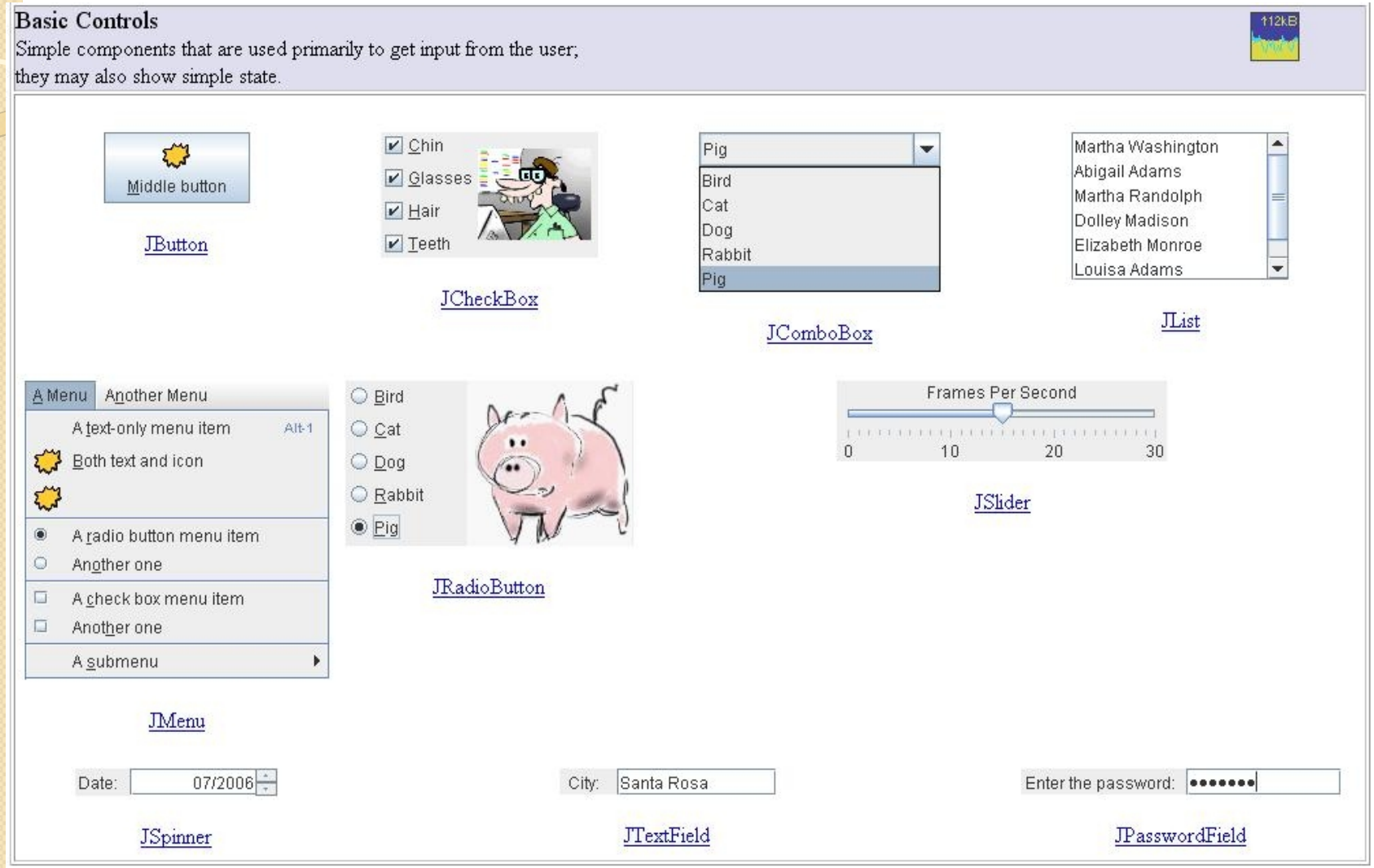




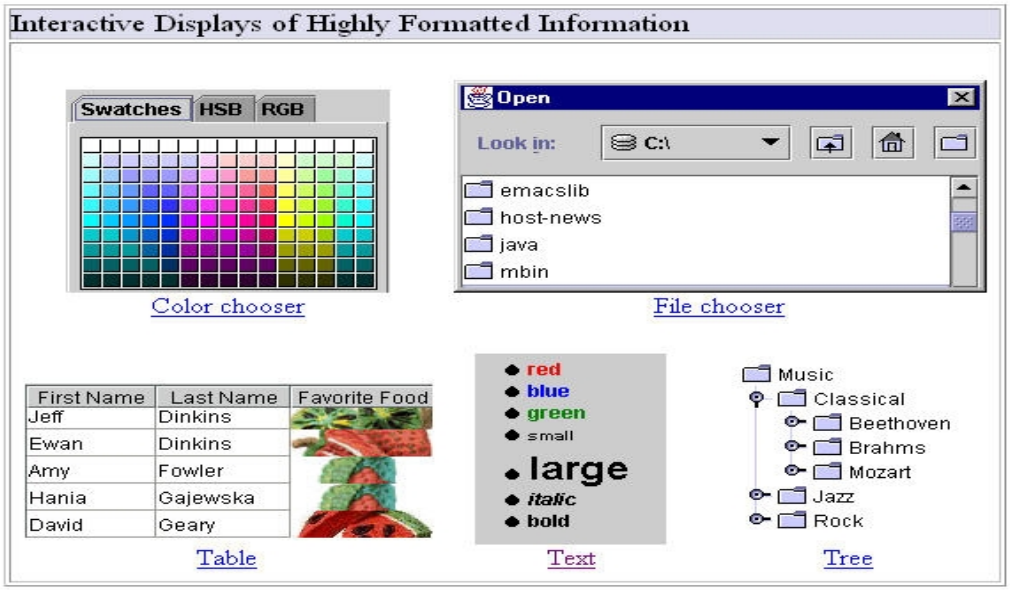


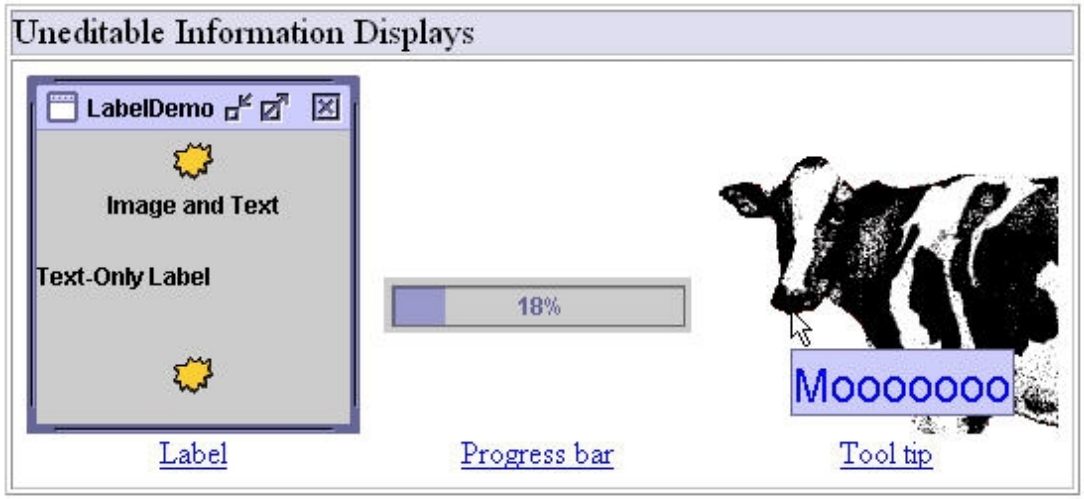


Swing Components



Swing Components

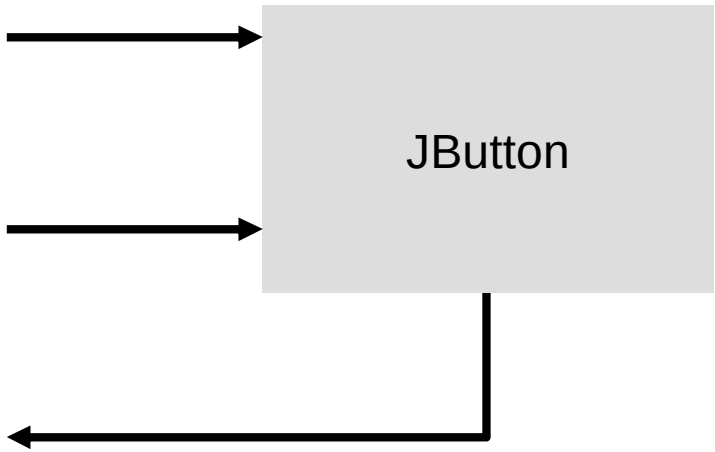




GUI Component API

- Java: GUI component = class

- Properties
  -
- Methods
  -
- Events
  -



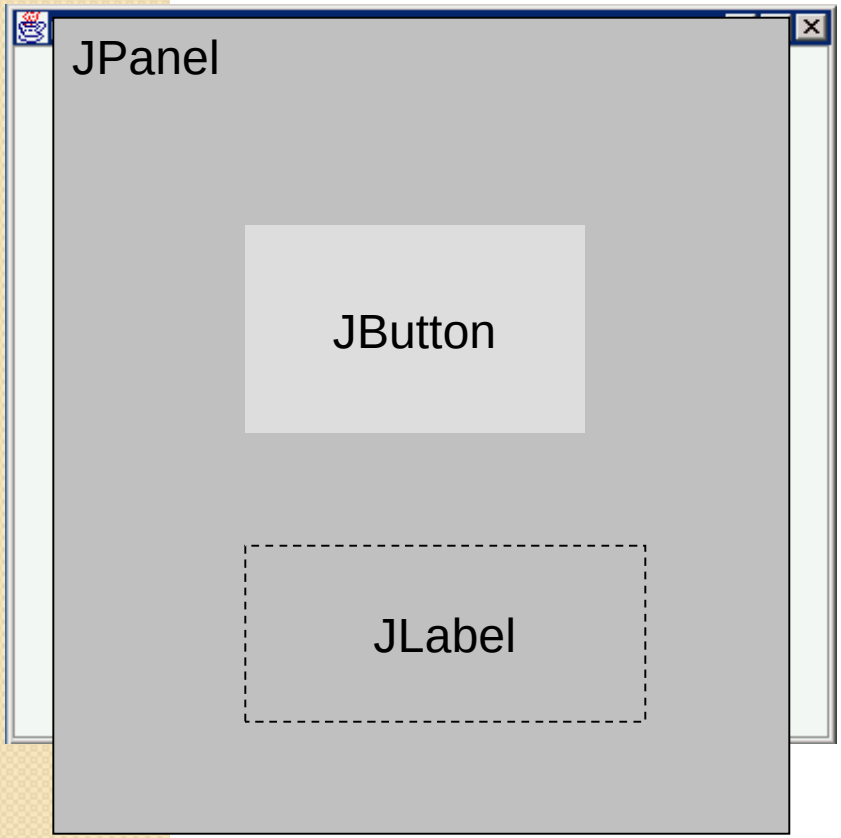
Using a GUI Component

1. Create it
   - Instantiate object: b = new JButton(“press me”);
2. Configure it
   - Properties: b.text = “press me”; [avoided in java]
   - Methods: b.setText(“press me”);
3. Add it
   - panel.add(b);
4. Listen to it
   - Events: Listeners

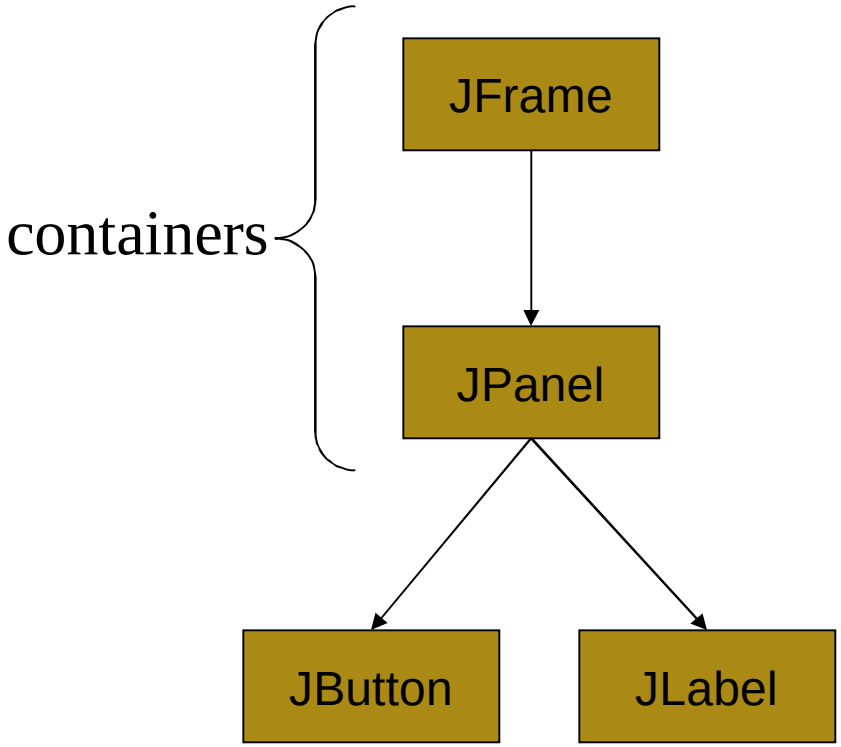
Anatomy of an Application GUI

GUI

JFrame

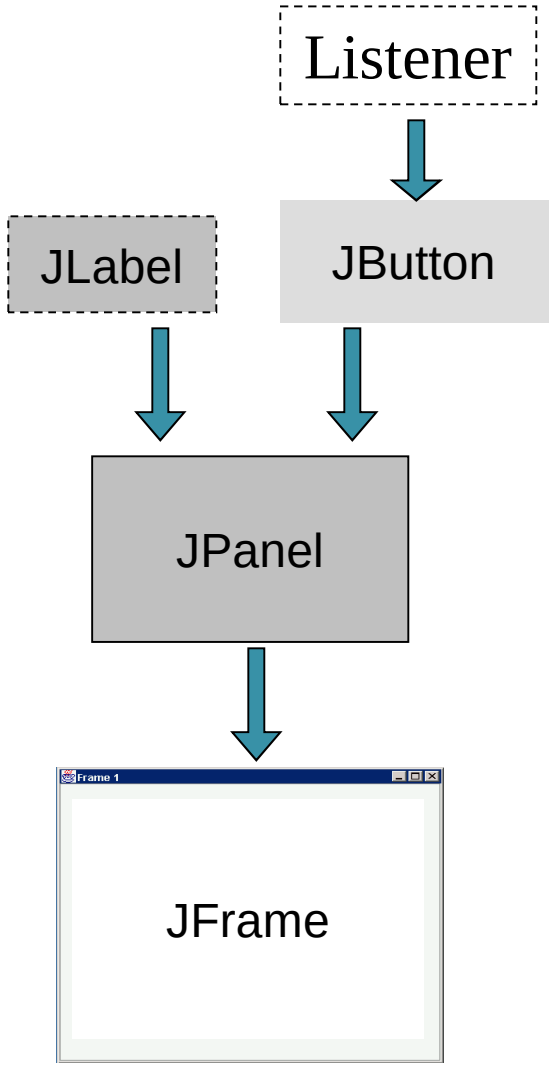


Internal structure



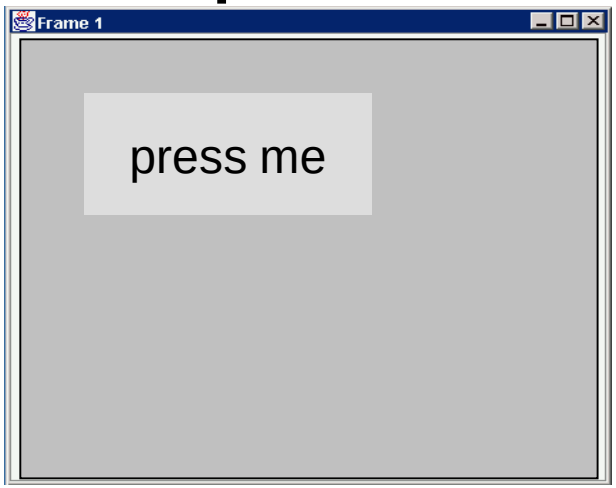
Build from bottom up

- Create:
  - Frame
  - Panel
  - Components
  - Listeners
- Add: (bottom up)
  - listeners into components
  - components into panel
  - panel into frame



Code

```
JFrame f = new JFrame(“title”);
JPanel p = new JPanel( );
JButton b = new JButton(“press me”);

p.add(b);                    // add button to
  panel
f.setContentPane(p);    // add panel to
  frame

f.show();
```



Application Code

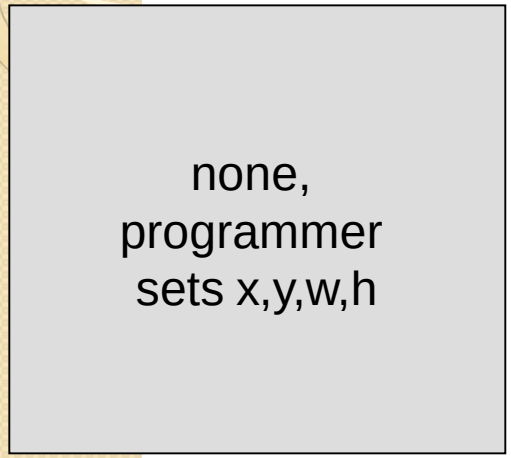
```
import javax.swing.*;

class hello {
  public static void main(String[] args){
      JFrame f = new JFrame("title");
      JPanel p = new JPanel();
      JButton b = new JButton("press me");

      p.add(b);                    // add button to
  panel
      f.setContentPane(p);         // add panel to
  frame

      f.show();
  }
}
```

Layout Managers

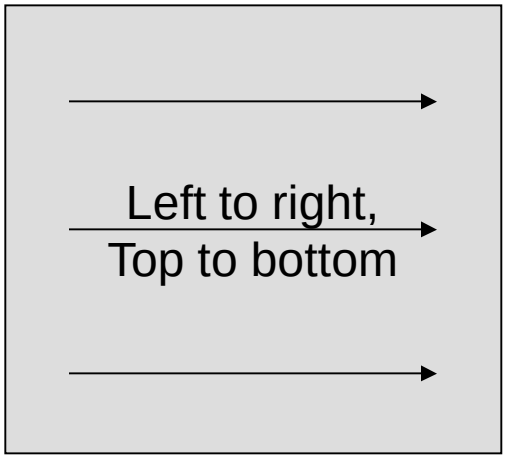
- Automatically control placement of components in a panel
- Why?

Layout Manager Heuristics

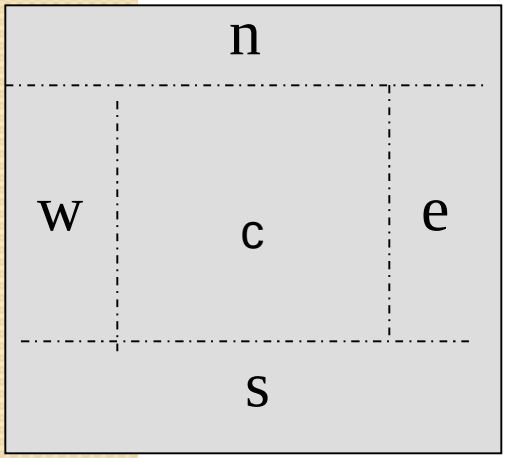
null



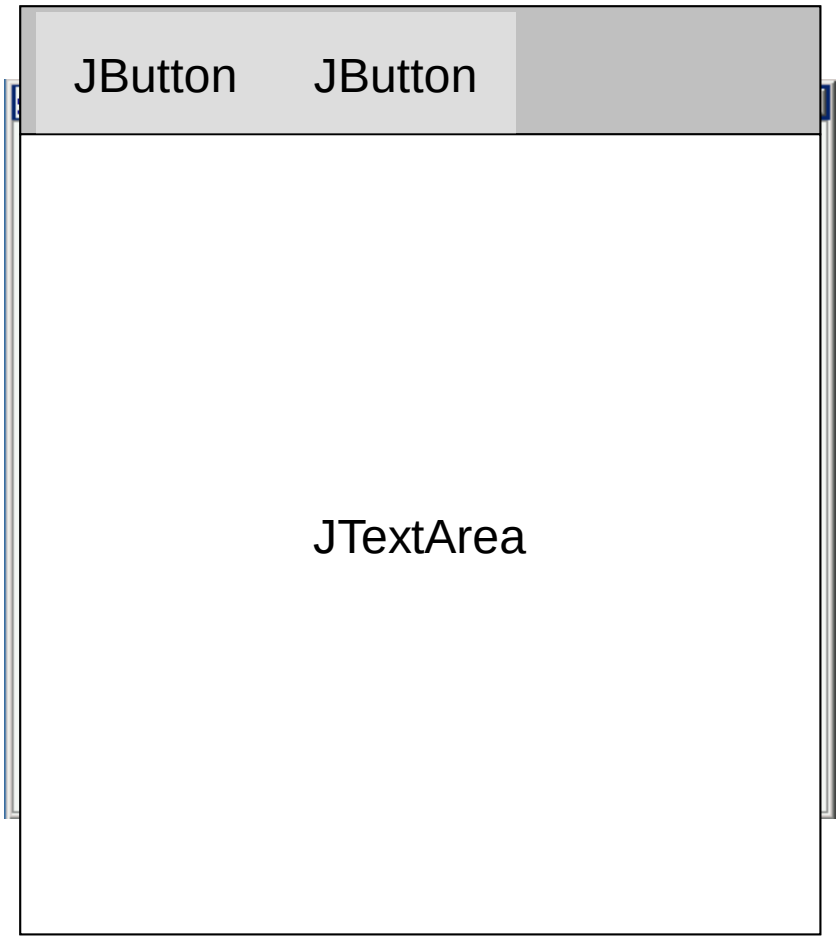
FlowLayout



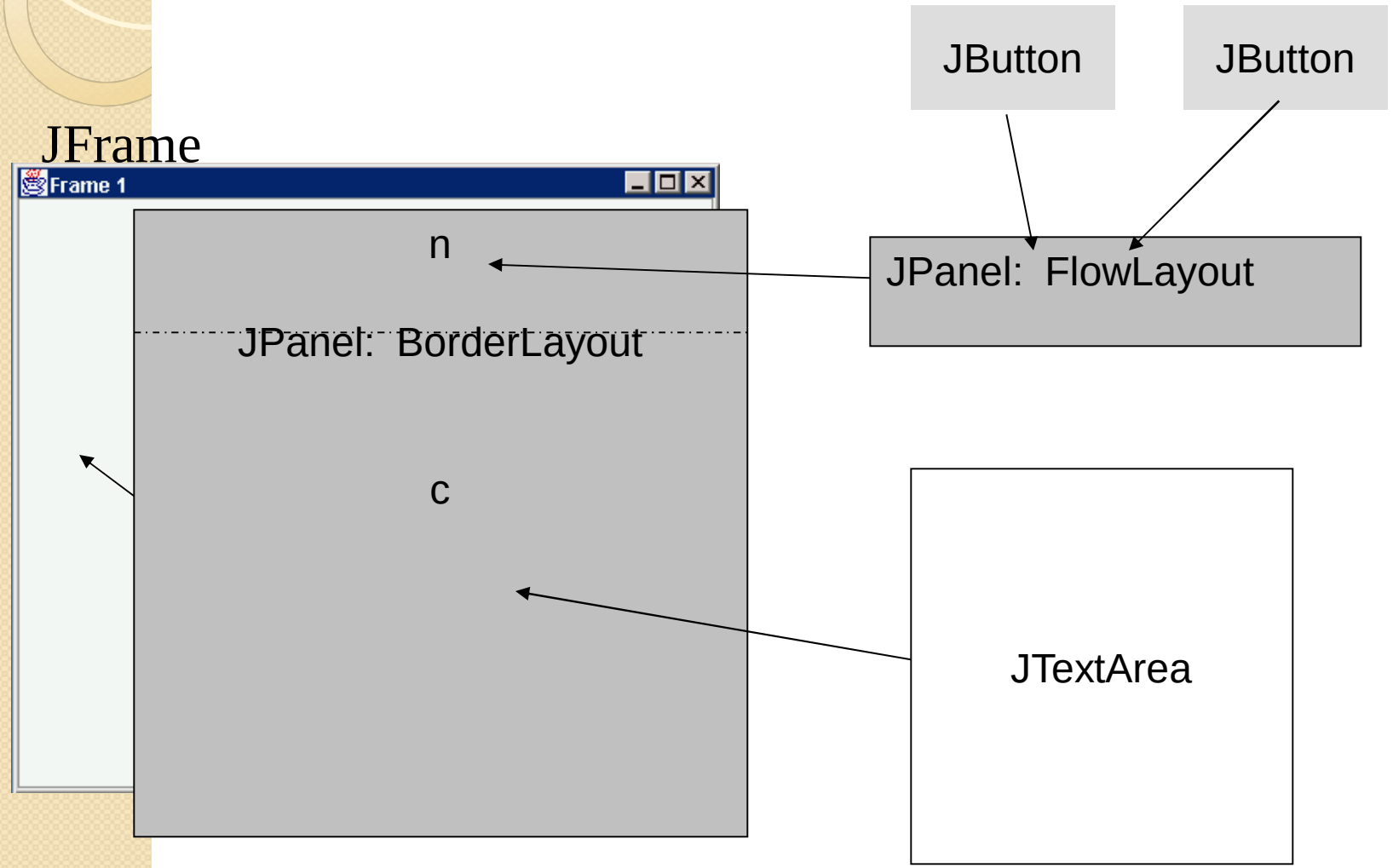
BorderLayout



Combinations



Combinations



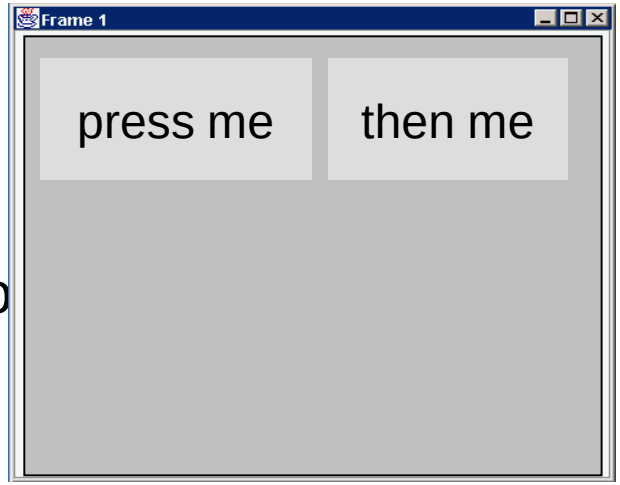
Code: null layout

```
JFrame f = new JFrame("title");
JPanel p = new JPanel( );
JButton b = new JButton("press me");

b.setBounds(new Rectangle(10,10,
  100,50));
p.setLayout(null);          // x,y layout
p.add(b);
f.setContentPane(p);
```

Code: FlowLayout

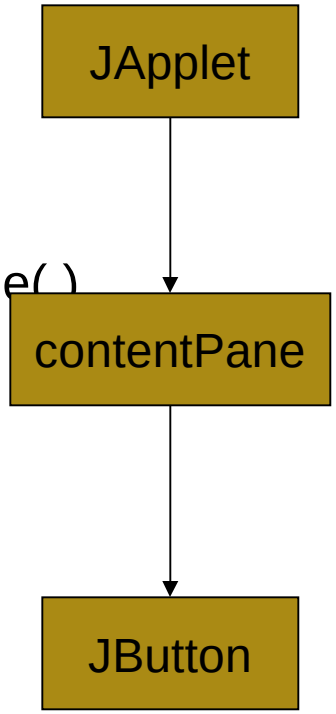
```
JFrame f = new JFrame("title");
JPanel p = new JPanel( );
FlowLayout L = new FlowLayout( );
JButton b1 = new JButton("press me");
JButton b2 = new JButton("then me");

p.setLayout(L);
p.add(b1);
p.add(b2);
f.setContentPane(p);
```

Set layout mgr before adding compo



Applets

- JApplet is like a JFrame
- Already has a panel
  - Access panel with JApplet.getContentPane( )



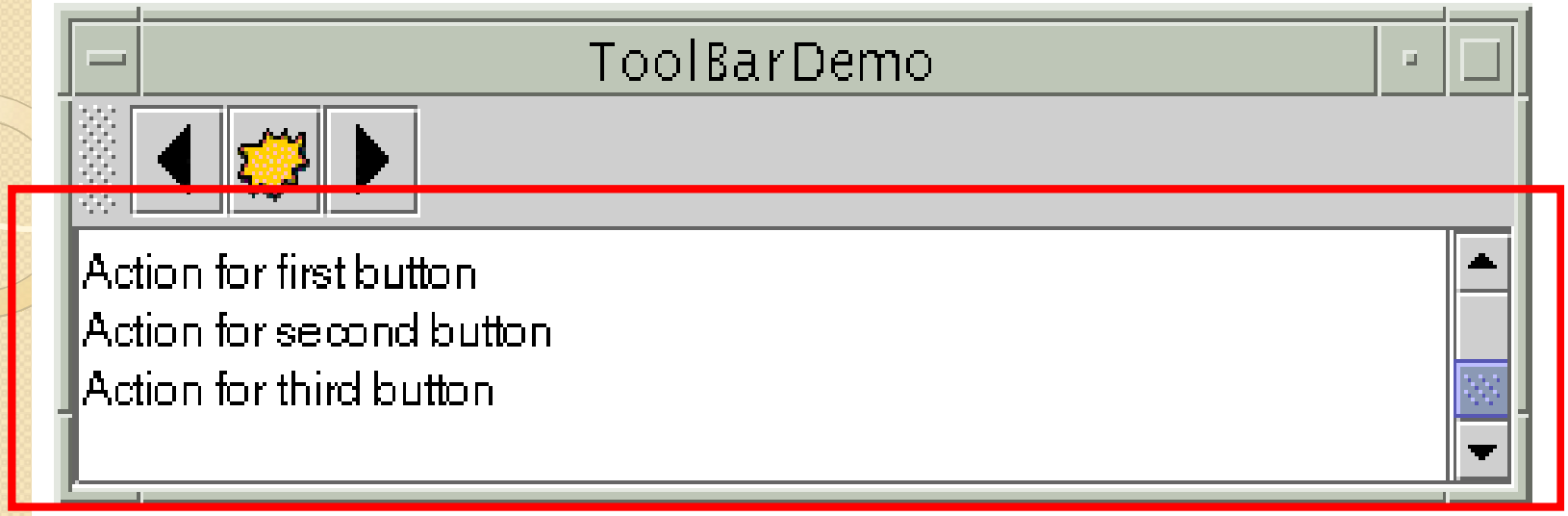

```
import javax.swing.*;

class hello extends JApplet {
  public void init(){
      JButton b = new JButton(“press me”);
      getContentPane().add(b);
  }
}
```

Applet Methods

- Called by browser:

- init( ) - initialization
- start( ) - resume processing (e.g. animations)
- stop( ) - pause
- destroy( ) - cleanup
- paint( ) - redraw stuff (‘expose’ event)

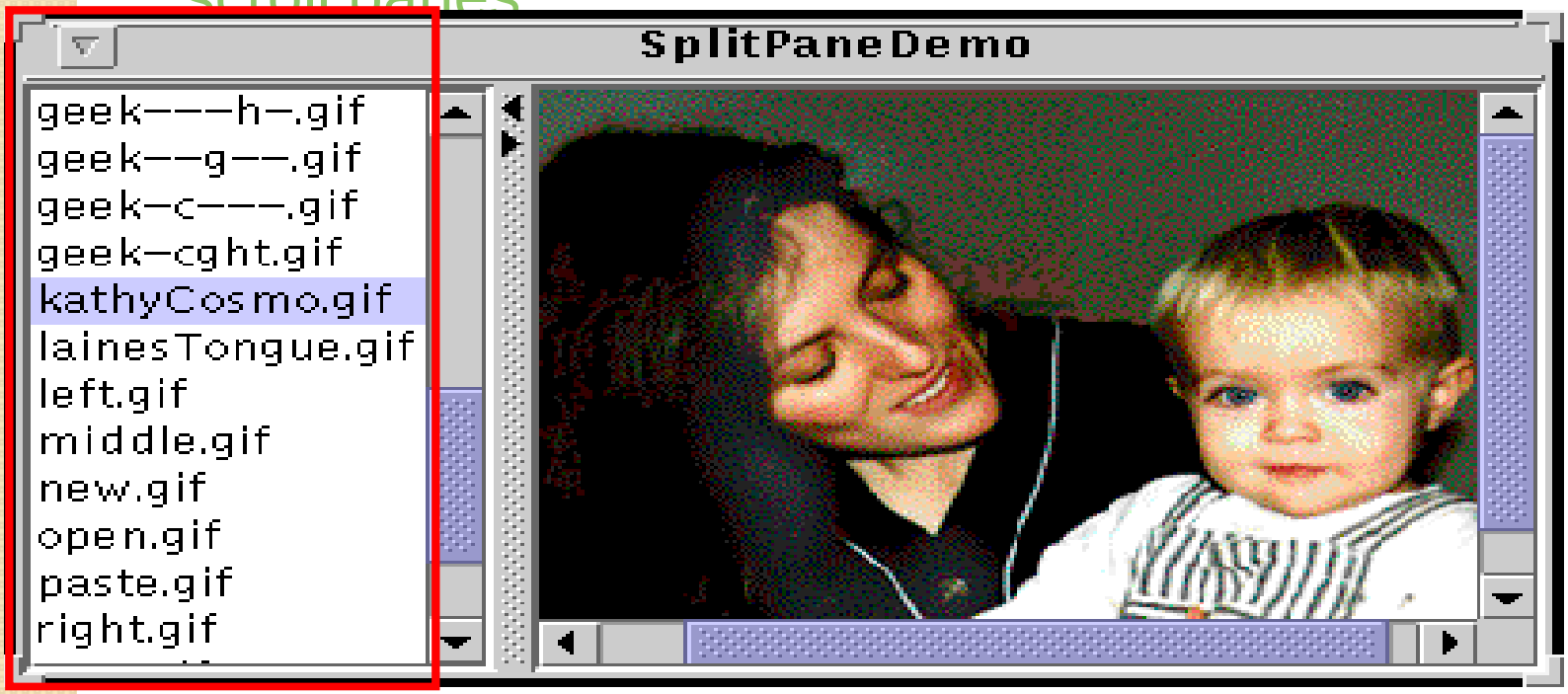
How to Use Scroll Panes




```
textArea = new JTextArea(5, 30);
JScrollPane scrollPane = new
    JScrollPane(textArea);
        ...
contentPane.setPreferredSize(new Dimension(400,
    100));
        ...
contentPane.add(scrollPane,
    BorderLayout.CENTER);
```

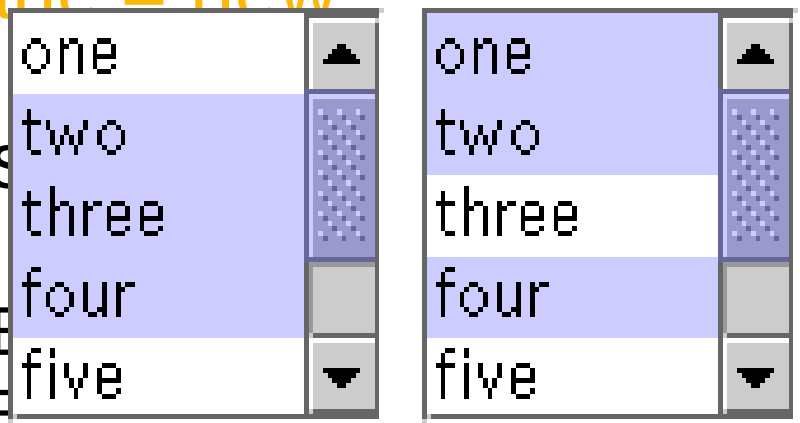
How to Use Lists

A JList presents the user with a group of items, displayed in one or more columns, to choose from. Lists can have many items, so they are often put in scroll panes.



```
// Create the list of images and put it in a scroll pane
 JList list = new JList();
list.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
    ...
 JScrollPane listScrollPane = new JScrollPane(list);
```

- possible selection modes
  - SINGLE_SELECTION
  - SINGLE_INTERVAL_SELE
  - MULTIPLE_INTERVAL_SE



Creating a Model

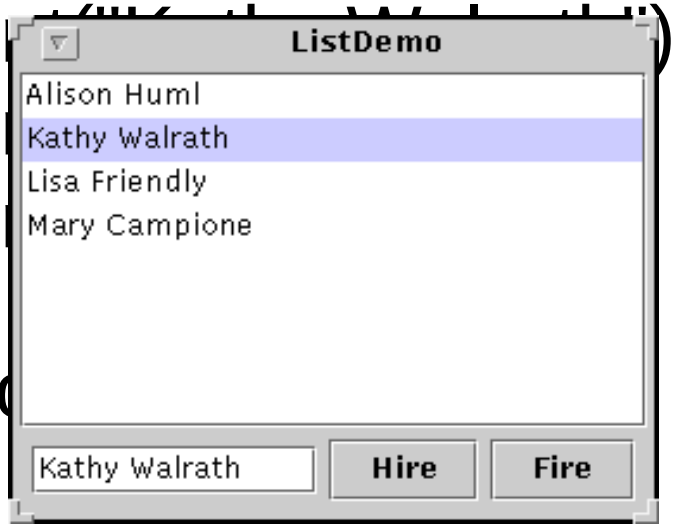
There are three ways to create a list model:

- DefaultListModel –Take care of few things.
- AbstractListModel — you manage the data and invoke the "fire" methods. For this approach, you must subclass AbstractListModel and implement the getSize and getElementAt methods inherited from the ListModel interface.
- ListModel — you manage everything.

Adding Items to and Removing Items from a List

Here is the code that creates a list model object, puts the initial items in it, and uses the list model to create a list:

- ListModel listModel = new DefaultListModel();
- listModel.addElement("Alison Huml");
- listModel.addEleme
- listModel.addEleme
- listModel.addEleme Campione");
- list = new JList(listM



Event Handling

- JList fires list selection events whenever the selection changes.
- You can process these events by adding a list selection listener to the list with the addListSelectionListener method.
- A list selection listener must implement one method: valueChanged

Trees

- The simplest and most common way to use JTree is to create objects of type DefaultMutableTreeNode to act as the nodes of the tree.
- The MutableTreeNode interface extends TreeNode. The DefaultMutableTreeNode class implements the MutableTreeNode.
- Nodes that have no children will be displayed as leaves.
- The toString method returns the selected node.

- Once you have some nodes, you hook them together in a tree structure via parentNode.add(childNode).
- add() is a method of DefaultMutableTreeNode.
- Finally, you pass the node to the JTree constructor.
- Note that, since trees can change size based upon user input (expanding and collapsing nodes), trees are usually placed inside a JScrollPane.

For example, here is a very simple tree

- DefaultMutableTreeNode root = new DefaultMutableTreeNode("Root"); DefaultMutableTreeNode child1 = new DefaultMutableTreeNode("Child 1"); root.add(child1); DefaultMutableTreeNode child2 = new DefaultMutableTreeNode("Child 2"); root.add(child2); JTree tree = new JTree(root); someWindow.add(new JScrollPane(tree));

Tables

- A table is a component that displays rows and columns of data.
- Table is implemented by JTable class .Its constructor is:

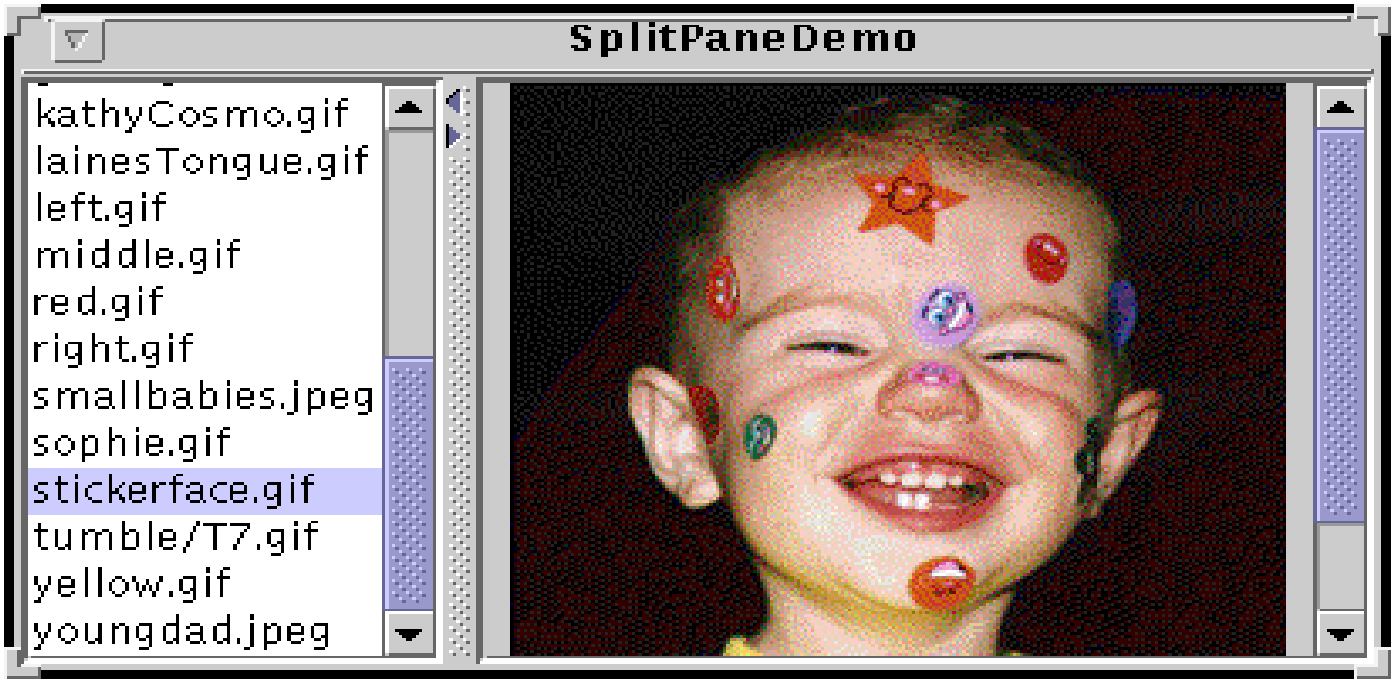
JTable(Object data[][],Object colHeads[])

Here data is a 2-dimensional array of information to be presented and colHeads is 1-dimensional array of column headings.

Steps for using table in an Applet

- Create JApplet object.
- Create JScrollPane object.
- Add table to scroll pane
- Add scroll pane to the content pane of applet.

Split Pane


SplitPaneDemo
kathyCosmo.gif
lainesTongue.gif
left.gif
middle.gif
red.gif
right.gif
smallbabies.jpeg
sophie.gif
stickerface.gif
tumble/T7.gif
yellow.gif
youngdad.jpeg

the code

```
//Create a split pane with the two scroll panes in it.
 splitPane = new
   JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
                                listScrollPane,
   pictureScrollPane);
splitPane.setOneTouchExpandable(true);
      splitPane.setDividerLocation(150);

//Provide minimum sizes for the two components in
   the split pane
      Dimension minimumSize = new
   Dimension(100, 50);
      listScrollPane.setMinimumSize(minimumSize);

   pictureScrollPane.setMinimumSize(minimumSize
   );
```

Formatted Text Fields

- Formatted text fields provide a way for developers to specify the valid set of characters that can be typed in a text field. Specifically, the JFormattedTextField class adds a *formatter* and an object *value* to the features inherited from the JTextField class. The formatter translates the field's value into the text it displays, and the text into the field's value.

- Using the formatters that Swing provides, you can set up formatted text fields to type dates and numbers in localized formats. Another kind of formatter enables you to use a character mask to specify the set of characters that can be typed at each position in the field. For example, you can specify a mask for typing phone numbers in a particular format, such as (XX) X-XX-XX-XX-XX.
- Using the formatters that Swing provides, you can set up formatted text fields to type dates and numbers in localized formats. Another kind of formatter enables you to use a character mask to specify the set of characters that can be typed at each position in the field. For example, you can specify a mask for typing phone numbers in a particular format, such as (XX) X-XX-XX-XX-XX.