Distributed Database Systems

Java AWT Programming

605.741

David Silberberg

Overview

- This is a review, not a course on JAVA
- The assumption: you have JAVA programming experience
- Covers some of the user interface concepts that you will need
- This lecture covers AWT - you may use SWING for your project
- Documentation can be found on line
  - Goto java.sun.com
  - Navigate to documentation

Main Routine

```
import java.awt.*;

public class MyFrame extends Frame {

    // The main method is ignored by applets
    public static void main(String[] args) {
        // Marty's code has QuittableFrame
        Frame mainFrame = new MyFrame();

        // match applet dimensions
        mainFrame.resize(<width>, <height>);
        mainFrame.pack();
        mainFrame.setVisible(true);
    }

    public MyFrame()
    {
        // Build Frame
        buildFrame(this);
    }
}
```

Panel

- Borderless window that can contain GUI elements
- Need to put it into a frame
- It is used to group components of a Panel

```
Panel panel = new Panel();
panel.add(component);
panel.add(anotherComponent);
```

BorderLayout

```
Button button1 = new Button("Button 1");
Button button2 = new Button("Button 2");
Button button3 = new Button("Button 3");
Button button4 = new Button("Button 4");
Button button5 = new Button("Button 5");

setLayout(new BorderLayout());
this.add("North", button1);
this.add("East", button2);
this.add("South", button3);
this.add("West", button4);
this.add("Center", button5);

this.remove(button1);
…
this.removeAll();
```

GridLayout

```
TextField field1 = new TextField();
TextField field2 = new TextField(30);
TextField field3 = new TextField("Init string");
TextField field4 = new TextField("Init string", 30);

setLayout(new GridLayout(rows, cols, <hgap>, <vgap>));

add(field1);
…

String val = field1.get<Selected>Text();
```

ScrollPane

- Holds only one component
- Usually, it holds something that is too large to display all at once

---

```
ScrollPane sp = new ScrollPane();
ScrollPane sp = new
    ScrollPane(ScrollPane.SCROLLBARS_ALWAYS);
ScrollPane sp = new ScrollPane(ScrollPane.AS_NEEDED);
ScrollPane sp = new
    ScrollPane(ScrollPane.SCROLLBARS_NEVER);
```

List Boxes

```
List list = new List();
list.addItem(field1);
list.addItem(field2);
…
add(list);
…

for(j=0; j<numListItems; j++) {
    if (list.isSelected(j)) {
        …
    }
}
```

OR

```
String[] itemList = list.getSelectedItems();
```

TextArea

- Sets up larger text areas
- Different constructors

```
TextArea area1, area2, area3, area4;
area1 = new TextArea(rows, columns);
area2 = new TextArea("Initial text");
area3 = new TextArea("Initial text", rows, columns);
area4 = new TextArea("Initial text", rows, columns, scrollbarType);
```

```
TextArea.SCROLLBARS_BOTH
TextArea.SCROLLBARS_VERTICAL_ONLY
TextArea.SCROLLBARS_HORIZONTAL_ONLY
TextArea.SCROLLBARS_NEITHER
```

Label

```
Label label = new Label("label string",
                        Label.LEFT | Label.RIGHT | Label.LEFT);
label.setFont(new Font("Helvetica", Font.BOLD, 12);
```

Button

```
Button button = new Button("label");
button.setLabel("new label");
String buttonLabel = button.getLabel();
```

Checkbox

```
Checkbox cb = new Checkbox("label", <true|false>);
panel.add(cb);
boolean bool = cb.getState();
cb.setState(bool);
```

CheckboxGroup

```
CheckboxGroup cbg = new CheckboxGroup();
Checkbox cb1 = new (label, state, cbg);
Checkbox cb2 = new (label, state, cbg);

Checkbox cb = cbg.getCurrent();
Checkbox cb = cbg.selectedCheckbox();
cbg.setCurrent(cb);
```

Listeners

- Listeners are defined as interfaces in Java
- ActionListener

```
import java.awt.*;
import java.awt.event.*;

public class MyActionListener implements ActionListener {

   // variables defined

   public void MyActionListener(…) {
      // initializations
   }
```

Action Listener (cont.)

```
        public void actionPerformed (
            ActionEvent event) {

            String
               event.getActionCommand();
            // do something with string

            // or

            Object obj = event.getSource();
            // do something with object

        }
}
```

```
// Other code
{
        …
        MyActionListener al = new
                                MyActionListener(…);
        Button button = new Button(label);
        button.addActionListener(al);
        …
}
```

Button Listener

```
public class MyButtonListener implements ActionListener {
    public MyButtonListener (…) {
        …
    }

    public void ActionPerformed(ActionEvent ae) {
        String command = ae.getActionCommand();

        if (command.equals("Select")) {
            …
        }
    // or
        if (button == getSource()) {
            …
        }
        …
    }
}
```

Button Listener (cont.)

- In calling code:

```
MyButtonListener buttonListener = new MyButtonListener(…);

Button button1 = new Button("Text of Button");
add(button1);
button1.addActionListener(buttonListener);
```