CS 4244: Internet Programming

User Interface Programming in Java
Java Foundation Classes (JFC)

- Set of classes within J2SE for GUI development and graphics
- JFC APIs or packages include
  - AWT
  - Swing
  - Java 2D
  - Applets
  - Accessibility
  - Internationalization
  - ...
Swing and AWT Architecture

- AWT provides
  - Basic facilities for creating GUIs
  - Drawing graphics

- Swing is newer GUI toolkit
  - Extension of AWT
  - All GUI components within Swing are lightweight, making it more portable
  - Larger and more comprehensive than AWT
Simple GUI

Do you really want to quit

Yes, please!  No thanks!
import java.awt.*;     // AWT classes
import javax.swing.*;  // Swing components and classes
import javax.swing.border.*; // Borders for Swing components
import java.awt.event.*; // Basic event handling

/*
 * This example is from the book "Java Foundation Classes in a Nutshell".
 * Written by David Flanagan. Copyright (c) 1999 by O'Reilly & Associates.
 * You may distribute this source code for non-commercial purposes only.
 * You may study, modify, and use this example for any purpose, as long as
 * this notice is retained. Note that this example is provided "as is",
 * WITHOUT WARRANTY of any kind either expressed or implied.
 */
public class DisplayMessage {
    public static void main(String[] args) {
        /*
         * Step 1: Create the components
         */
        JLabel msgLabel = new JLabel();   // Component to display the question
        JButton yesButton = new JButton(); // Button for an affirmative response
        JButton noButton = new JButton();  // Button for a negative response
    }
}
Simple GUI

- JLabel
  - Do you really want to quit
  - Yes, please!
  - No thanks!

- JButton
Code for the GUI

/*
 * Step 2: Set properties of the components
 */

msgLabel.setText(args[0]);  // The msg to display
msgLabel.setBorder(new EmptyBorder(10,10,10,10));
yesButton.setText((args.length >= 2)?args[1]:"Yes");
noButton.setText((args.length >= 3)?args[2]:"No");
Code for the GUI

/*
 * Step 3: Create containers to hold the components
 */

    JFrame win = new JFrame("Message"); // The main application window
    JPanel buttonbox = new JPanel(); // A container for the two buttons
Simple GUI

Jframe (win)

Jpanel (buttonbox)
/*
 * Step 4: Specify LayoutManagers to arrange components in the containers
 */

win.getContentPane().setLayout(new BorderLayout());
buttonbox.setLayout(new FlowLayout());
Simple GUI

BorderLayout

FlowLayout

Do you really want to quit

Yes, please!  No thanks!
Code for the GUI

/*
 * Step 5: Add components to containers, with optional layout constraints
 */
buttonbox.add(yesButton);     // add yes button to the panel
buttonbox.add(noButton);     // add no button to the panel

// add JLabel to window, telling the BorderLayout to put it in the middle
win.getContentPane().add(msgLabel, "Center");

// add panel to window, telling the BorderLayout to put it at the bottom
win.getContentPane().add(buttonbox, "South");
Simple GUI

Do you really want to quit

Yes, please!  No thanks!
Code for the GUI

/*
 * Step 6: Arrange to handle events in the user interface.
 */

yesButton.addActionListener(new ActionListener() {
    // Note: inner class
    // This method is called when the Yes button is clicked.
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
});

noButton.addActionListener(new ActionListener() {
    // Note: inner class
    // This method is called when the No button is clicked.
    public void actionPerformed(ActionEvent e) {
        System.exit(1);
    }
});
/*
 * Step 6: Arrange to handle events in the user interface.
 */

yesButton.addActionListener(new ActionListener() { // Note: inner class
    // This method is called when the Yes button is clicked.
    public void actionPerformed(ActionEvent e) { System.exit(0); }
});

noButton.addActionListener(new ActionListener() { // Note: inner class
    // This method is called when the No button is clicked.
    public void actionPerformed(ActionEvent e) { System.exit(1); }
});
/*
 * Step 7: Display the GUI to the user
 */
win.pack(); // Set the size of the window based its children's sizes.
win.show();  // Make the window visible.
/*
 * Step 7: Display the GUI to the user
 */

win.pack();  // Set the size of the window based its children's sizes.

win.show();  // Make the window visible.
Review of Swing classes

- JFrame
- JPanel
- JButton
- JLabel

Containers with layout
Some other Swing classes

- JTextField
- JEditorPane
- Jmenu, JMenuBar, JMenuItem
- Others at
  http://java.sun.com/docs/books/tutorial/uiswing/components/components.html
Taken from “The Java Tutorial”
http://java.sun.com/docs/books/tutorial/uiswing/components/menu.html
JMenuBar, JMenu and JMenuItem

```java
menuBar = new JMenuBar();
setJMenuBar(menuBar);
menu = new JMenu("A Menu");

menuBar.add(menu);

menuItem = new JMenuItem("A text-only menu item");
menu.add(menuItem);
```