

Java Foundation Classes

GUI Components: Swing & AWT

Pluggable Look & Feel

Java 2D API

Drag and Drop

Accessibility API

Pluggable Look & Feel

| | | |
|------------|--------------------|----------------|
| Java | MetalLookAndFeel | any -- default |
| CDE/Motif | MotifLookAndFeel | any |
| MS Windows | WindowsLookAndFeel | Win32 |
| Mac OS | MacLookAndFeel | Mac OS |

Swing is not a replacement for AWT (Abstract Window Toolkit)

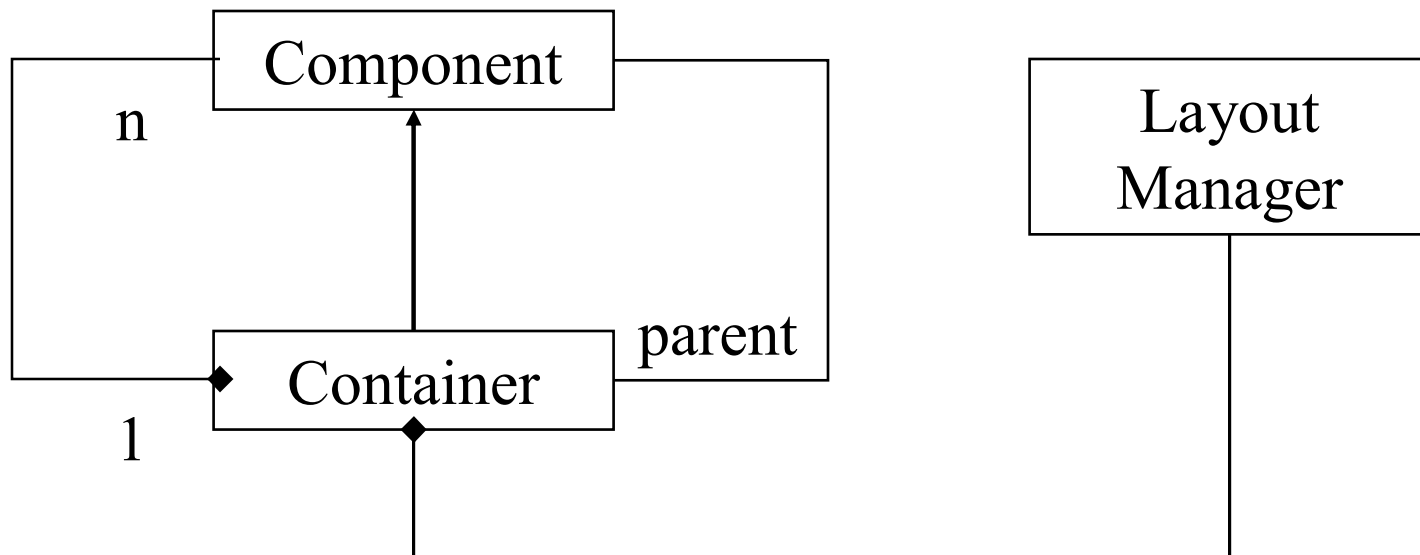
Swing classes extend AWT

| | Swing Heavyweight components | Swing Lightweight components |
|-----|---|------------------------------|
| AWT | Applet, Frame, Window, Dialog | |
| | Component, Container, Graphics, Color, Font, Toolkit, Layout Managers | |

Heavyweight components are opaque, windows, drawn by native window system, not as portable.

Lightweight components can have transparent backgrounds, drawn by JVM, portable.

Java UIs consist of containers using layout managers to control the geometry of UI components.



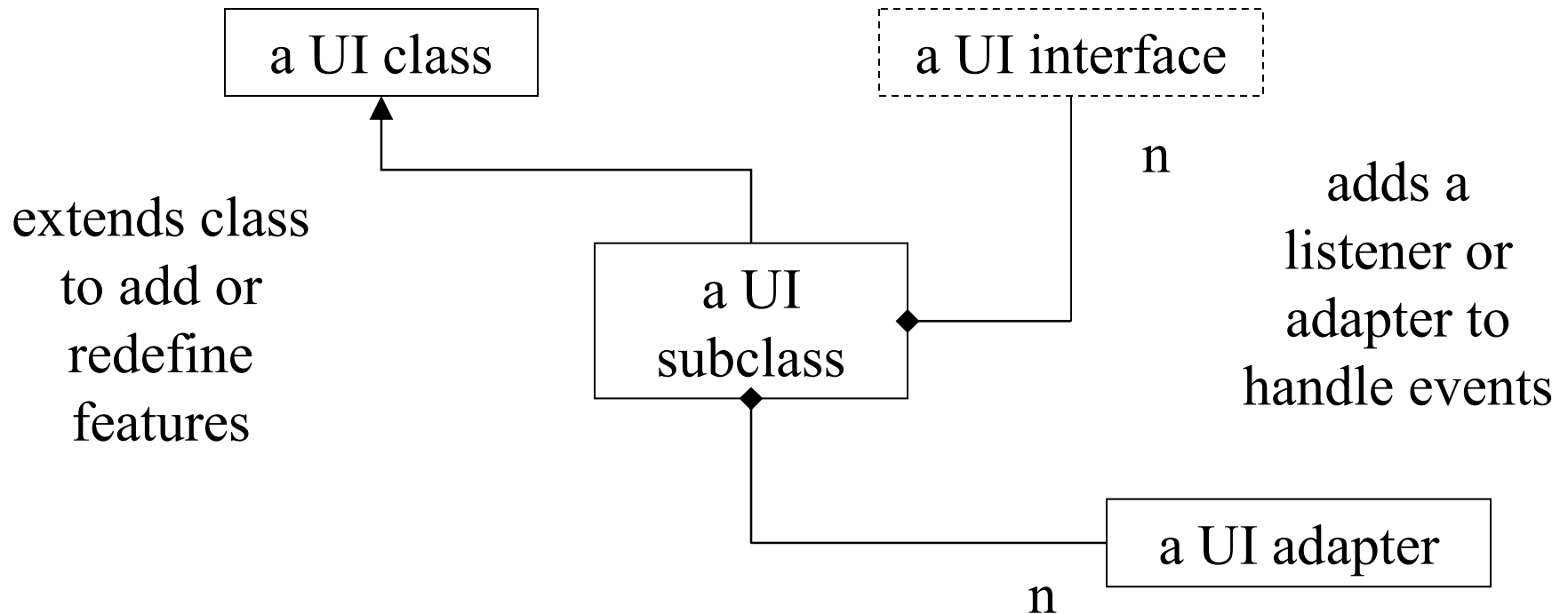
Java UI components can register listeners (or adapters) to react to events that are “fired off” by components.

components have `add*Listener()` methods for expected events.

listeners are java interfaces.

- classes** containers, components, layout managers, events, adapters
JFrame, JLabel, BorderLayout, MouseAdapter
- interfaces** event listeners
ActionListener, MouseListener
interfaces are messy w 2+ methods.
every implementor must define all methods
- adapters** convenience classes
JFC provides an adapter class for every listener
interfaces w/ 2+ methods.
Adapter extends an interface w/ methods w/ null behavior.
MouseAdapter, WindowAdapter

Java has single inheritance for classes but can implement multiple interfaces.



Inner Classes

Inner class's definition is nested w/in outer class

Named or unnamed inner classes

Scope rules allow reference of outer class variables

The listener's interface member functions are invoked when an event is received.

ActionListener, AdjustmentListener, ContainerListener, FocusListener, ItemListener, KeyListener, MouseListener, MouseMotionListener, TextListener, WindowListener.

Events are represented by event classes.

MouseEvent has getX(), getY(), getPoint(), getClickCount(), translatePoint(int x, int y), isPopupTrigger() interfaces.

Not all events have a specific class -- some event classes represent related events and use integer constants to identify the actual event.

WindowEvent handles: activating, deactivating, closing, opening, iconifying, and de-iconifying windows with getWindow() interface

Event handling wiring "*problem*"

Designer of class wants to support an "application specific" behavior when a user initiates an event (request for behavior), but doesn't know what the event will be.

Provides a typed "*callback slot*" reference and set methods to "add" or "remove" an event handler object.

Application developer wants to use or extend existing GUI classes and be able to add "application specific" behaviors (methods) to be performed when user requests them via the GUI control.

Must provide correct type "*callback object*" when setting event handler.

| | |
|-----------------|--|
| ActionEvent | button activated, list item double clicked, menuItem selected. textField (enter typed) |
| AdjustmentEvent | scrollbar thumb moved |
| ItemEvent | checkbox is toggled, checkboxMenuItem selected choice item selected, list item selected |
| TextEvent | textComponent text changes |
| ComponentEvent | component has moved (automatically handled) |
| ContainerEvent | component added / removed (auto handled) |
| FocusEvent | focus has been gained or lost |
| PaintEvent | need to repaint (automatically handled) |
| WindowEvent | window state has changed: minimized, |
| InputEvent | |
| KeyEvent | |
| MouseEvent | |
| MouseWheelEvent | |

JApplet, JFrame, JDialog, JWindow

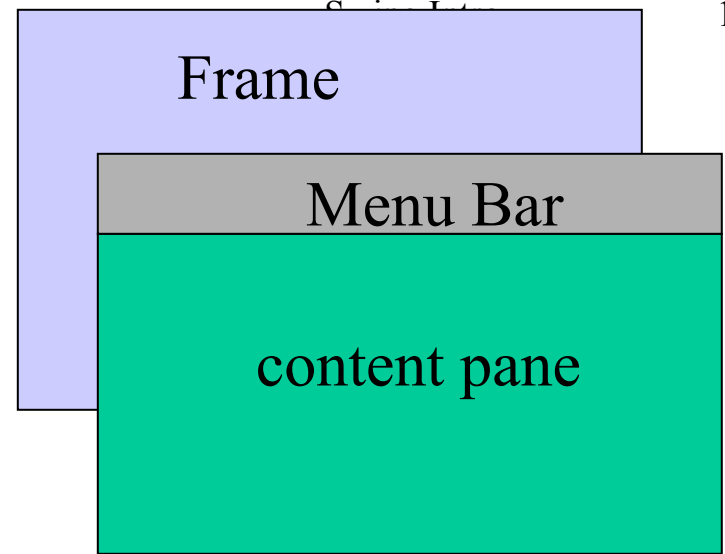
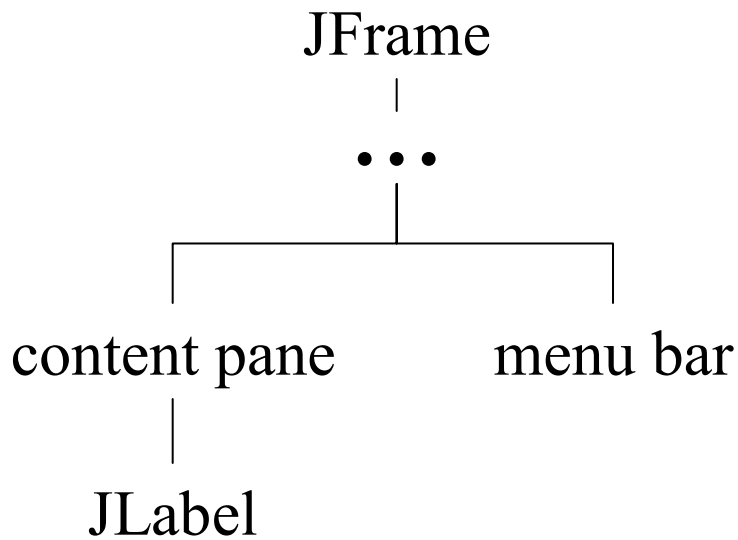
Top level containers are the root of a containment hierarchy.

Interface between the native windowing environment and window manager and java application or applet

JApplet, JFrame and JDialog contain a JRootPane which contains a contentPane container.

components and layout managers must be added and set to the content pane not JApplet, JFrame or JDialog. (else exception thrown)





```
JFrame frame = new JFrame("frame title");
```

Constructor creates menubar and sets layout managers.

Need to get the content pane to add components to the frame

```
frame.getContentPane().add(greenLabel, BorderLayout.CENTER);
```

Adding a menu bar to the frame

```
frame.setJMenuBar(greyMenuBar);
```

Respond to window closing

```
frame.addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent e) {System.exit(0);}
})
```

```
frame.pack(); // resolves all containment management
frame.setVisible(); // displays the frame
```

Menus in menubars or as popup

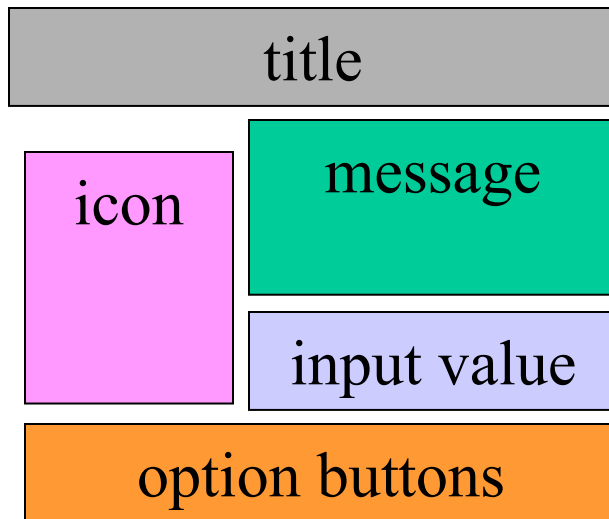
Simple menu setup steps: (order can vary)

1. extend a JFrame for top container
2. instantiate a JMenuBar
3. instantiate a JMenu for every menu choice in the menu bar
4. instantiate each JMenuItem, JRadioButtonMenuItem, or JCheckboxMenuItem choice in the JMenu
 - set JMenuItem icon
 - set mnemonics and/or key accelerators
 - add any submenus (pull right menus)
 - add any item separators
3. add the JMenuItemItems to JMenu
- 4a. add a common menuItemListener for all (set of) menu items
- 4b. add an ActionListener each menu item.
5. add the JMenu to the JMenuBar
6. set the MenuBar in the frame

Dialogs

Swing has many pre-build dialogs

JOptionPane class has prebuilt, message, warning, information, question dialogs with default icons and “ok” button. Also customizable.



```
JOptionPane.showMessageDialog( null, "Hello 585",  
    "HI!", JOptionPane.INFORMATION_MESSAGE);
```

<< walkthrough Pipe.java >>

JFileChooser class has options to display various file choosing dialogs.

```
JFileChooser.showOpenDialog(Component parent)
```

```
JFileChooser.showSaveDialog(Component parent)
```

```
JFileChooser.showDialog(Component parent,  
String title_approve)
```

After file chooser is closed
(modal dialog)

```
getSelectedFile()
```

obtains file reference

```
setCurrentDirectory(File)
```

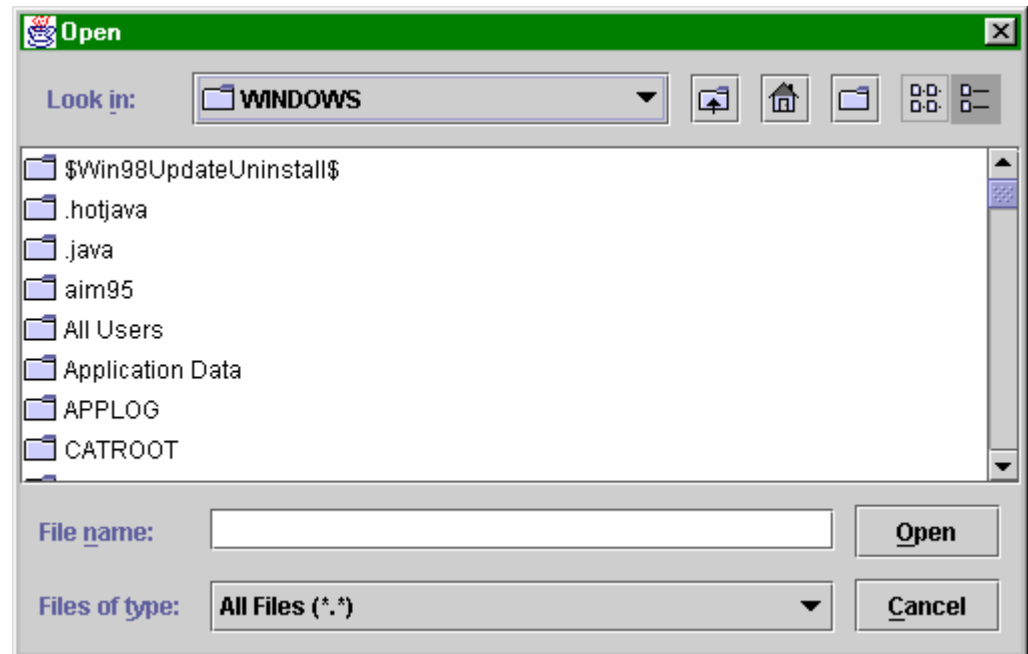
```
setFileFilter(FileFilter)
```

```
setFileSelectionMode(int)
```

Accessory components

preview, icons

displays



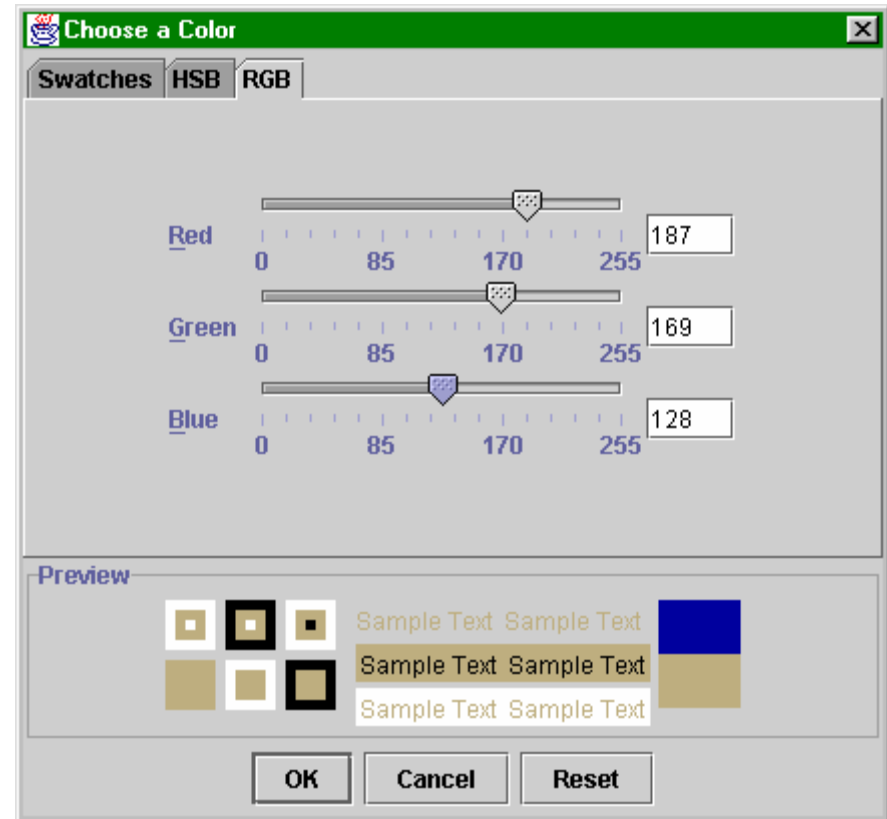
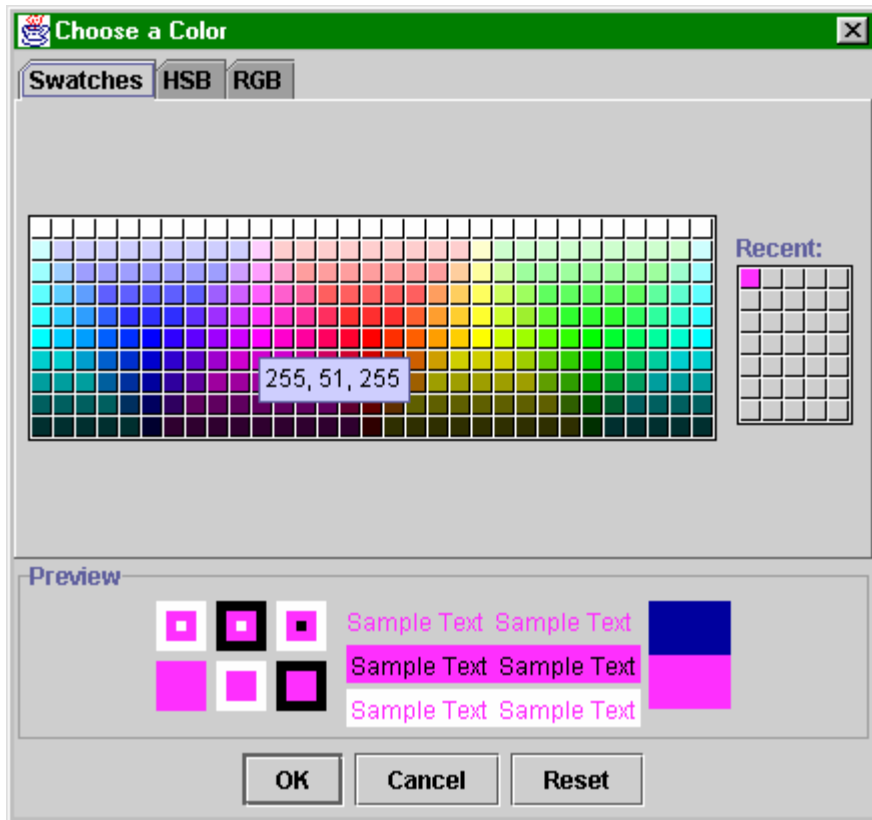
Color chooser

JColorChooser class has options to display color choosing dialogs.

`JColorChooser.createDialog()` creates a color chooser

`JColorChooser.show(...args)` creates a chooser from *args*

`getColor()` returns selected color



Run the swing set demo

```
...\jdk*\demo\jfc\SwingSet2 java -jar SwingSet2.html
```

```
...\jdk*\demo\jfc\SwingSet2 java -jar SwingSet2.jar
```