GUI Basics

Object Orientated Programming in Java

Benjamin Kenwright

Outline

- Essential Graphical User Interface (GUI) Concepts
  - Libraries, Implementation, Mechanics, ..
  - Abstract Windowing Toolkit (AWT)
  - Java Foundation Classes (JFC)
- Today's Practical
- Review/Discussion

Graphical User Interfaces (GUI)

- Note this is a huge area many books are devoted solely to this topic
- Today we will provide an overview on getting started with Java GUIs

Why is the Graphical User Interface (GUI) Important?

Why is the Graphical User Interface (GUI) Important?

- Visual feedback/input
- Allows higher productivity
- Faster learning curve/usability
  - Intuitive to the user
- Display/show more information/details
  - Picture is worth a thousand words
  - Allows colour/animations
  - Provides more opportunities (e.g., video/games)
- …

GUI Overview

- To create a Java GUI, you need to understand
  - Containers
  - Event
  - Event Handlers
  - Layout managers
  - Components
  - Special features

AWT and JFC/Swing

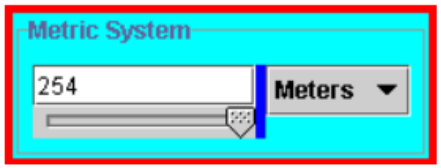
- Early Java development used graphic classes defined in the
- Abstract Windowing Toolkit (AWT).
  - ▷ See the java.awt packages.
- In Java 2, JFC/Swing classes were introduced.
  - ▷ See the javax.swing packages
- Many AWT components have improved Swing counterparts.
  - ▷ An example, the AWT Button class corresponds to a more versatile Swing class called JButton.
- Swing does not generally replace the AWT; still use for AWT events and the underlying AWT event processing model

Containers

- A container is a special component that can hold other components.
- The AWT class, as well as the Swing class, are containers
- Other containers include
  - Frames
    - A frame is a container that is free standing and can be positioned anywhere on the screen.
    - Frames give the ability to do graphics and GUIs through applications
  - Dialog boxes
  - Panels
  - Panes
  - Toolbars

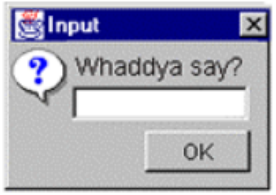
Containers
(Top Level and General)

Applet



Panel

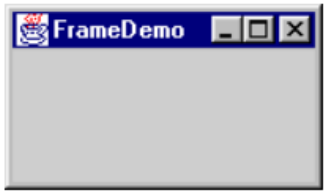
Tabbed Pane



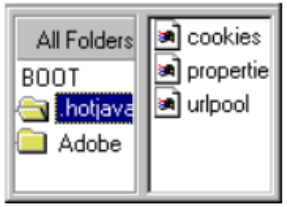
Dialog

Scroll Pane

Toolbar



Frame

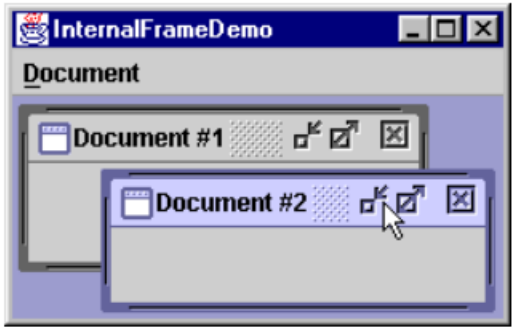


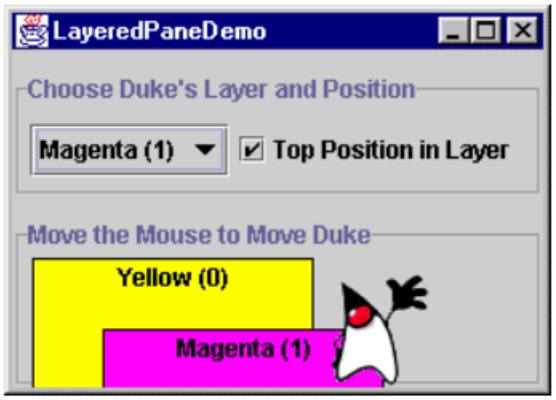
Split Pane

[Source: java.sun.com]

Special Containers

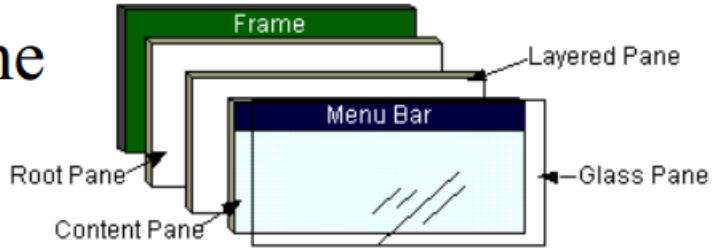
Internal frame





Layered pane

Root pane



Events

- Every time the user types a character or pushes a mouse button, an event occurs
- Any object can be notified of the event.
- All the objects have to do implement the appropriate interface and be registered as an event listener on the appropriate event source

Events, cont.

- Several events implemented in java.awt.AWTEvent subclasses (java.awt.Event is deprecated).
  - Defines a lot of constants

```
public abstract class AWTEvent extends EventObject {
  public void setSource(Object newSource);
  public int getID();
  public String toString();
  public String paramString();
  protected void consume();
  protected boolean isConsumed();
}
```

Events Handlers

- In the declaration for the event handler class, one line of code specifies that the class either implements a listener interface (or extends a class that implements a listener interface).
  - public class MyClass implements ActionListener
- In the event handler class the method(s) in the listener interface must be implemented
  - public void actionPerformed(ActionEvent e) { /* code that "reacts" to the event */ }
- Register an instance of the event handler class as a listener on one or more components.
  - myComponent.addActionListener(myClassInstance)

Events Handlers, cont.

```
class AL implements ActionListener {
    public void actionPerformed (ActionEvent e){
      int xValue = Integer.parseInt(x.getText());
      model.setX(xValue);
      int yValue = Integer.parseInt(y.getText());
      model.setY(yValue);
      String temp = Integer.toString(model.calc());
      prod.setText(temp);
    }
}
```

- Often an event handler that has only a few lines of code is implemented using an anonymous inner class.

Events Handlers, cont.

- SwingApplication has two event handlers.
  - ▷Window closing (window events).
    - frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
- Button clicks (action events).
  - ▷see previous slide.
- Types of events (listeners defined in java.awt.event)

Click button	⇒	**`ActionListener`**
Close frame	⇒	**`WindowListener`**
Press mouse button	⇒	**`MouseListener`**
Move mouse	⇒	**`MouseMotionListener`**
Component visible	⇒	**`ComponentListener`**
Component gets focus	⇒	**`FocusListener`**

WindowListener and MouseListener

```
public interface WindowListener extends EventListerner {
   void windowActivated(WindowEvent e);
   void windowClosed(WindowEvent e);
   void windowClosing(WindowEvent e);
   void windowDeactivated(WindowEvent e);
   void windowDeiconified(WindowEvent e);
   void windowIconified(WindowEvent e);
   void windowOpened(WindowEvent e);
}
```

```
public interface MouseListener extends EventListener {
  public void mouseClicked(MouseEvent e);
  public void mousePressed(MouseEvent e);
  public void mouseReleased(MouseEvent e);
  public void mouseEntered(MouseEvent e);
  public void mouseExited(MouseEvent e);
}
```

Layout Managers

- A layout manager is an object that determines the manner in which components are displayed in a container
- There are several predefined layout managers defined in the Java standard class library

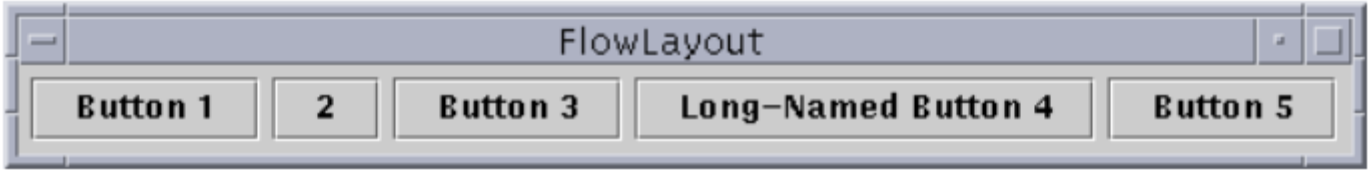
Flow Layout	(in **`java.awt`**)
Border Layout	(in **`java.awt`**)
Card Layout	(in **`java.awt`**)
Grid Layout	(in **`java.awt`**)
GridBag Layout	(in **`java.awt`**)
Box Layout	(in **`javax.swing`**)
Overlay Layout	(in **`javax.swing`**)

Layout Managers, cont.

- Every container has a default layout manager, but we can also explicitly set the layout manager for a container
- Each layout manager has its own particular rules governing how the components will be arranged
- Some layout managers pay attention to a component's preferred size or alignment, and others do not
- The layout managers attempt to adjust the layout as components
- are added and as containers are resized

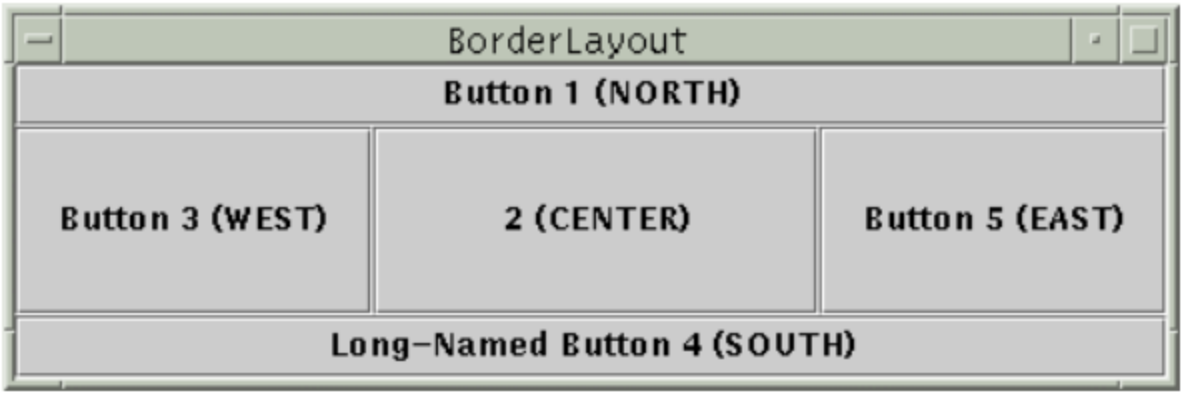
Flow Layout

- A flow layout puts as many components on a row as possible, then moves to the next row
- Rows are created as needed to accommodate all of the components
- Components are displayed in the order they are added to the container
- The horizontal and vertical gaps between the components can be explicitly set
- Default for JPanel



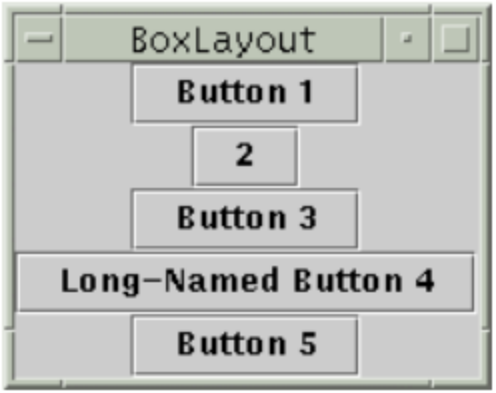
Border Layout

- A border layout defines five areas into which components can be added
- The default for most GUIs



Box Layout

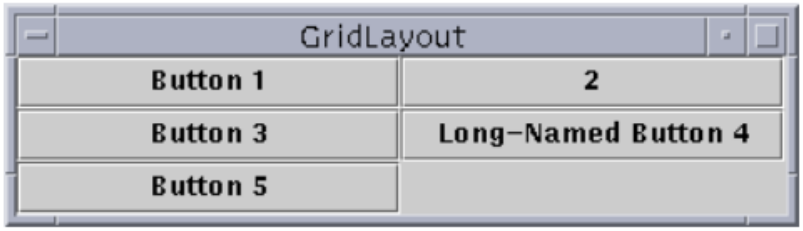
- A box layout organizes components either horizontally (in one row) or vertically (in one column)
- Special rigid areas can be added to force a certain amount of spacing between components
- By combining multiple containers using box layout, many different configurations can be created
- Multiple containers with box layouts are often preferred to one container that uses the more complicated gridbag layout manager



Other Layout Managers

Card layout. The area contains different components at different times.

Gridbag layout. The most sophisticated and flexible.



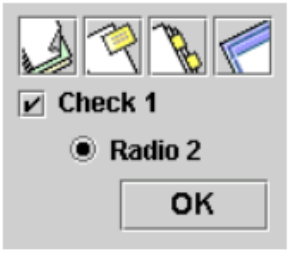
Grid layout. All equal size in a grid.

"Atomic" Components

- The root in the component hierarchy is JComponent.
- The JComponent provides the following functionality to its descendants, e.g., JLabel, JRadioButton, and JTextArea.
  - Tool tips
  - Borders
  - Keyboard-generated actions
  - Application-wide pluggable look and feel
  - Various properties
  - Support for layout
  - Support for accessibility
  - Double buffering

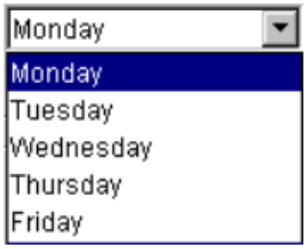
Basic Components

Button

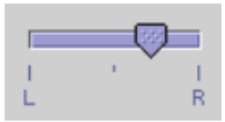


Menu

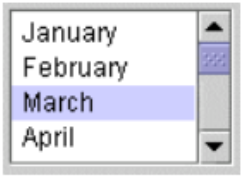
Combo Box



Slider



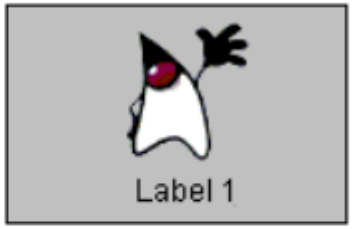
List



Text Field

Non-Editable Displays

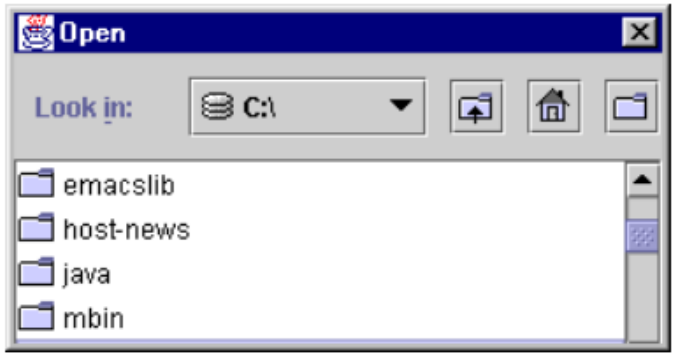
Label



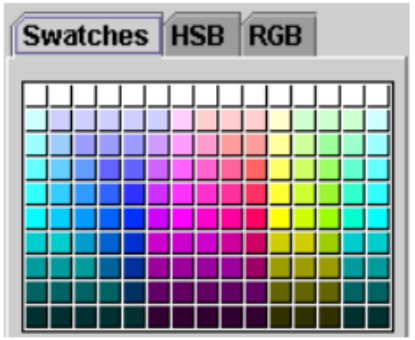
Progress bar

Tool tip

Editable Displays



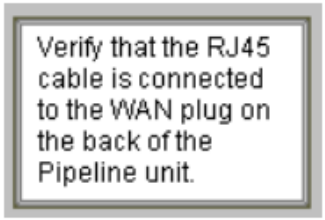
File Chooser



Color Chooser

First Na...	Last Name
Mark	Andrews
Tom	Ball
Alan	Chung
Jeff	Dinkins

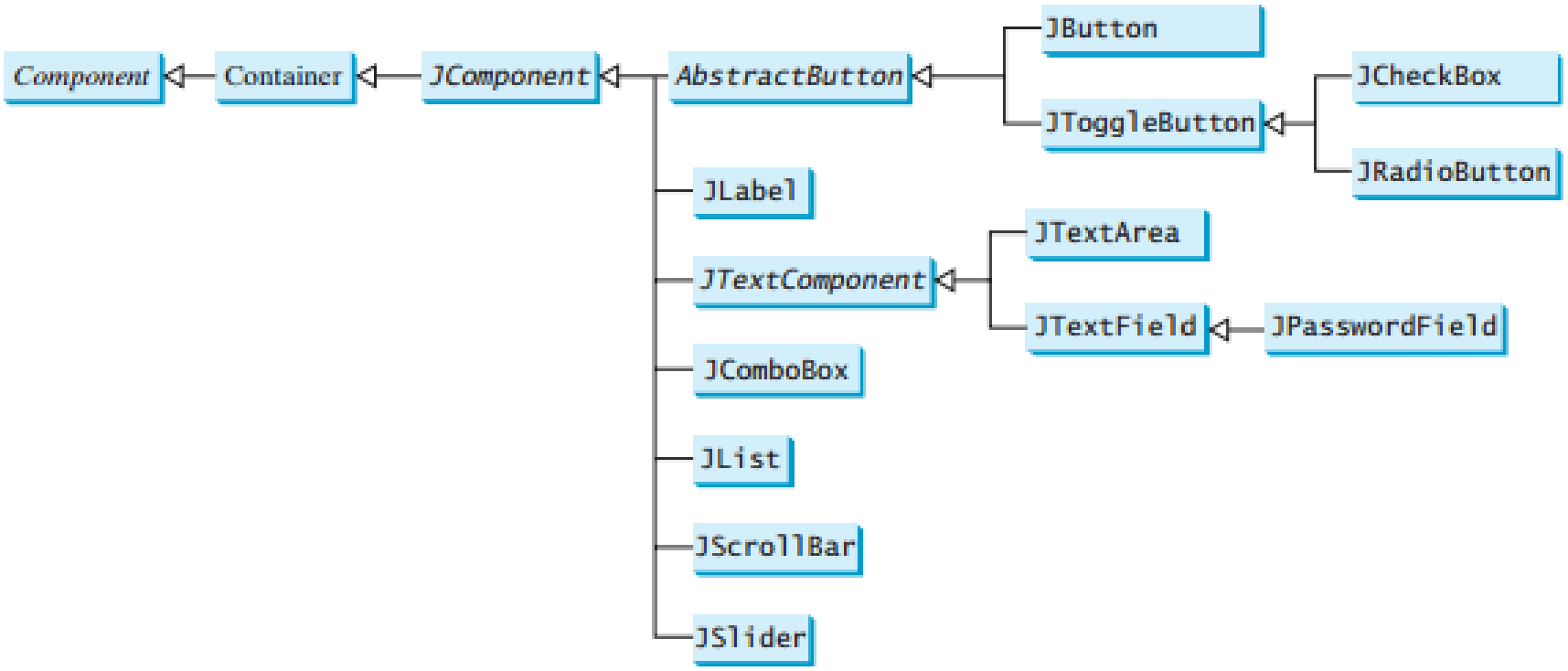
Table



Text

Tree

Review Popular GUI Components used to Create User Interfaces (Swing)



This Week

- Read Associated Chapters
- Review Slides
- Online Quizzes
- Java Exercises

Summary

- Overview Basic GUI Principles
- Abstract Windowing Toolkit (AWT)
- Java Foundation Classes (JFC)
- Hands-On/Practical

Questions/Discussion