

CMSC 132: Object-Oriented Programming II



Nelson Padua-Perez
William Pugh

Department of Computer Science
University of Maryland, College Park

Creating GUIs

■ Resources

- Appendix C of textbook
- Javadoc for the JDK
- Swing tutorial
- Course slides and code handouts
- Java Ranch

GUI (Graphical User Interface)

- You create GUIs by defining objects such as
 - Text fields
 - Labels
 - Buttons
 - Checkboxes
 - Radioboxes
 - Menus
 - Tables
 - Etc.

Java GUI Classes

- **AWT (Abstract Window Toolkit) (java.awt.*)**
 - Old GUI framework for Java (Java 1.1)
 - Some reliance on native code counterparts
 - Platform independence problems
- **Swing (javax.swing.*)**
 - New GUI framework first introduced in Java 1.2
 - Includes AWT features plus many enhancements
 - Pure Java components (no reliance on native code)
 - Pluggable look and feel architecture
- **SWT (Standard Widget Toolkit; from Eclipse)**

GUI Classes

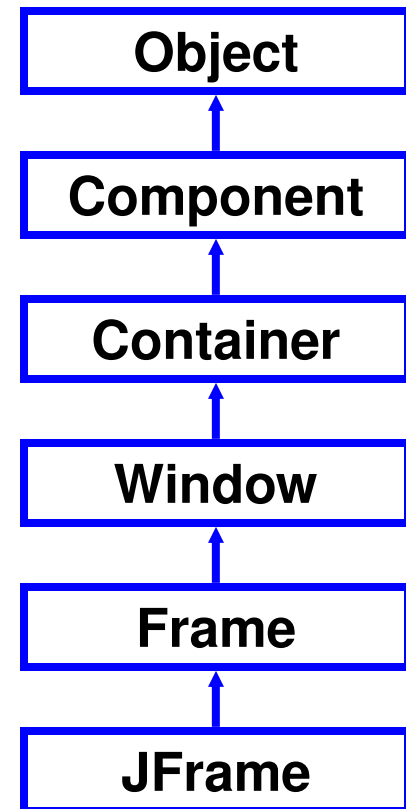
- **GUI classes can be organized in three groups**
 - **Container classes – Hold other GUI Components**
 - **JFrame -**
 - **JPanel**
 - **JApplet**
 - **Component classes – (e.g., JButton, JTextfield, etc.)**
 - **Helper classes – Describe properties of other GUI Components**
 - **Color**
 - **Graphics**
 - **Dimension**
 - **Other**

How to Create a GUI

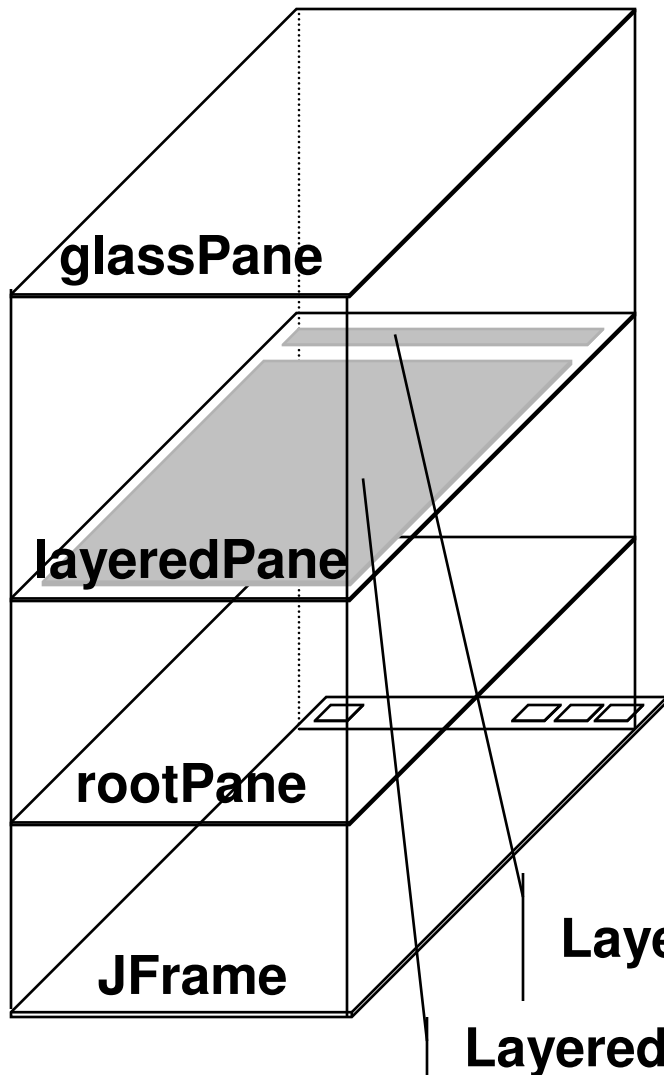
- **Define a Frame or Applet to hold the components. From now on we will use only frames.**
- **Add components to the frame**
- **Add actions to your GUI by adding listeners to GUI components**

JFrame Hierarchy

- Several super classes and well as implemented interfaces
- Many, member methods including inherited methods that allow for operations such as resizing, setting properties, adding components, etc.



JFrame Structure



- Most things go into content pane
 - `getContentPane()`
- Use glassPane for pop up menus, some animations
- Methods
 - `getRootPane()`
 - `getLayeredPane()`
 - `getContentPane()`
 - `getGlassPane()`
- Can set...Pane explicitly

LayeredPane manages (optional) JMenuBar

LayeredPane contains contentPane

Examples

■ **Examples in gui.jar**

- **For now our examples have no action associated with them.**
- **Let's see example in ex1 package**
- **ex2 is basically ex1 but reduced to one class**

Event Dispatching Thread (From Wikipedia)

- **Background thread to process events from the AWT graphical interface event queue**
- **These events are mainly update events that**
 - **Cause components to redraw themselves**
 - **Represent input events**
- **Swing uses a single-threaded painting model where only the Event Dispatching thread is the only valid thread that can update user interface components**
- **Updating components from other threads is a source of common bugs**

Event Dispatching Thread

- Following code fragment allows the current thread to execute the GUI code in the dispatching thread.
- `createAndDisplayGUI` – method that actually defines the GUI

```
javax.swing.SwingUtilities.invokeLater(new Runnable() {  
    public void run() {  
        createAndDisplayGUI();  
    }  
});
```

GUI Elements – Layout

■ Definition

- Arrangement of widgets in container

■ Layout specification

- Logical terms (2nd row, 1st column, left)
 - Preferred approach
- Actual coordinates (100 pixels, 5 inches)
 - Can be too rigid, limited to certain window sizes

■ Layout manager

- Entity translating layout specifications into actual coordinates at runtime, depending on conditions

Java Layout Managers

- **FlowLayout**
 - Lays out components from left to right
- **GridLayout**
 - Lays out components in a grid of user specified size
- **BorderLayout**
 - Designates portions of the container as North, South, East, West, and Center
- **CardLayout**
 - Adds components one on top of another
- **GridBagLayout**
 - Customizable manager that can use rows and columns of varying lengths

Example

■ Example in gui.jar

- ex3 (when manager = `LayoutManager.Grid`)
- ex3 (when manager = `LayoutManager.Flow`)