

# Java AWT

CS2704: Object-Oriented Software Design  
and Construction

Constantinos Phanouriou  
Department of Computer Science  
Virginia Tech

CS2704 (Spring 2000)

1

## Abstract Windowing Toolkit (AWT)

- AWT classes form 3 categories:
  - Graphics
    - colors, fonts, images, polygons, ...
    - draws stuff, defines events
  - Components
    - GUI components: buttons, menus, frames (window), panel (w/o window), dialog (window)
    - Container class contains components, layout managers
  - Layout Managers
    - Gives rule for graphical layout of components in a container
    - Ex: layout components in a grid

CS2704 (Spring 2000)

2

## Graphics classes of the java.awt package

<i>java.awt.Color</i>	<i>java.awt.Image</i>
<i>java.awt.Cursor</i>	<i>java.awt.Insets</i>
<i>java.awt.Dimension</i>	<i>java.awt.MediaTracker</i>
<i>java.awt.Event</i>	<i>java.awt.Point</i>
<i>java.awt.EventQueue</i>	<i>java.awt.Polygon</i>
<i>java.awt.Font</i>	<i>java.awt.PrintJob</i>
<i>java.awt.FontMetrics</i>	<i>java.awt.Rectangle</i>
<i>java.awt.Graphics</i>	<i>java.awt.Toolkit</i>

CS2704 (Spring 2000)

3

## Component classes of the java.awt package

<i>java.awt.Component</i>	<i>java.awt.Container</i>
<i>java.awt.Checkbox</i>	<i>java.awt.Panel</i>
<i>java.awt.Choice</i>	<i>java.awt.ScrollPane</i>
<i>java.awt.List</i>	<i>java.awt.Window</i>
<i>java.awt.Button</i>	<i>java.awt.Dialog</i>
<i>java.awt.Canvas</i>	<i>java.awt.FileDialog</i>
<i>java.awt.Label</i>	<i>java.awt.Frame</i>
<i>java.awt.Scrollbar</i>	
<i>java.awt.TextComponent</i>	<i>java.awt.MenuComponent</i>
<i>java.awt.TextArea</i>	<i>java.awt.MenuBar</i>
<i>java.awt.TextField</i>	<i>java.awt.MenuItem</i>
	<i>java.awt.Menu</i>

CS2704 (Spring 2000)

4


## Layout classes of the java.awt package

*java.awt.FlowLayout*  
*java.awt.GridLayout*  
*java.awt.GridBagConstraints*  
*java.awt.BorderLayout*  
*java.awt.CardLayout*  
*java.awt.GridBagLayout*

CS2704 (Spring 2000)

5

## GUI Components

- Button 
- Canvas
  - Roll your own GUI components
- Checkbox
  - Maintains boolean state
- Checkboxgroup
  - Combines checkboxes into radio buttons

CS2704 (Spring 2000)

6

## GUI Components (cont.)

- Choice
  - Menu of options drops down from button
  - Currently selected option is button label
- Label
  - Displays 1 line of text (read-only)



CS2704 (Spring 2000)

7

## GUI Components (cont.)

- List
  - Displays list of strings as N rows
  - Adds scrollbar if necessary
  - Allows single and multiple selection
  - Method to return selected item indexes



CS2704 (Spring 2000)

8

## GUI Components (cont.)

- Scrollbar
  - You specify orientation (horz, vert, and min/max range)
  - Event returned specifies
    - #lines or pages to scroll, or
    - absolute position to scroll to

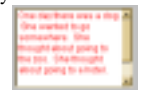


CS2704 (Spring 2000)

9

## GUI Components (cont.)

- TextComponent
  - Displays text
  - Can be editable (getText() returns text)
  - Can be selectable
- TextField
  - One line TextComponent
  - setEchoCharacter() allows password entry
- TextArea
  - Multiline TextComponent



CS2704 (Spring 2000)

10

## GUI Components (cont.)

- Window
  - Top level window without
    - borders,
    - menubar
  - show() makes window visible
  - toFront(), toBack()
  - pack() resizes window to fit components in it
  - dispose() frees resources when window is not needed
  - Example use: pop-up menu

CS2704 (Spring 2000)

11

## GUI Components (cont.)

- Dialog
  - A window for dialog box
  - Can be modal
- Frame
  - A window with title, menubar, icon, cursor
- Panel
  - Alternative to window
  - e.g., for display of Applet in Web browser



CS2704 (Spring 2000)

12

## AWT in JDK1.0 vs JDK1.1

- JDK1.1 makes several changes:
  - Some method names change to use set/get style of Java beans
  - You can use 1.0 methods in 1.1, but
    - compiler will tell you new names
    - the old names won't work in future JDK releases

CS2704 (Spring 2000)

13

## Information Dialog

- Code:
  - Class InfoDialog
  - Class MultiLineLabel (used in InfoDialog)



CS2704 (Spring 2000)

14

## Information Dialog

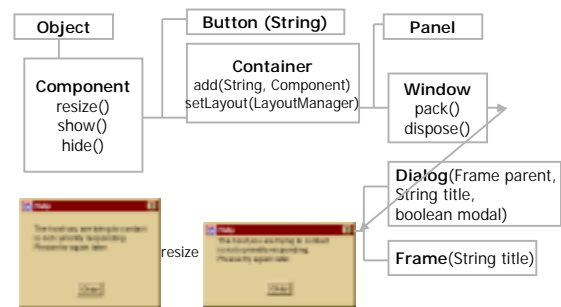
- Demonstrates:
  - Creating a window with message and button
  - Layout Managers for Containers
  - Hierarchical relationship of components



CS2704 (Spring 2000)

15

## Classes used by Example



CS2704 (Spring 2000)

16

## Example: Information Dialog

```
import java.awt.*;
public class InfoDialog extends Dialog {
```

*Let's look at main() first*

```
}
```



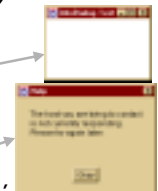
CS2704 (Spring 2000)

17

## Example - Main()

```
public static void main(String[] args) {
    Frame f = new Frame("InfoDialog Test");
    f.resize(100, 100);
    f.show();

    InfoDialog d = new InfoDialog(f, "Help",
        "The host you are trying to contact\n" +
        "is not currently responding.\n" +
        "Please try again later.");
    d.show();
}
```



CS2704 (Spring 2000)


18

## Example

```
import java.awt.*;
public class InfoDialog extends Dialog {
    protected Button button;
    protected MultiLineLabel label;

    public InfoDialog(Frame parent, String title,
                     String message)
    {
        // Create a dialog with the specified title
        super(parent, title, false);

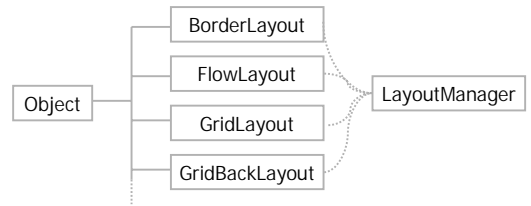
        // Create and use a BorderLayout manager
        // with specified margins
        this.setLayout(new BorderLayout(15, 15));
    }
}
```



*Why?*

CS2704 (Spring 2000) 19

## Layout Manger Classes




Layout Manager arranges components (Button, Dialog, List, ...) within containers (Panel, Dialog, Window, Frame) - not Button, List, ...

CS2704 (Spring 2000) 20

## BorderLayout Manager

```
import java.awt.*;
import java.applet.Applet;
public class buttonDir
    extends Applet {
    public void init() {
        setLayout(new BorderLayout());
        add("North", new Button("North"));
        add("South", new Button("South"));
        add("East", new Button("East"));
        add("West", new Button("West"));
        add("Center", new Button("Center"));
    }
} //BorderLayout(15, 15) produces gaps between regions!
```



CS2704 (Spring 2000) 21

## Recall...

```
import java.awt.*;
public class InfoDialog extends Dialog {
    protected Button button;
    protected MultiLineLabel label;

    public InfoDialog(Frame parent, String title,
                     String message)
    {
        // Create a dialog with the specified title
        super(parent, title, false);


        // Create and use a BorderLayout manager
        // with specified margins
        this.setLayout(new BorderLayout(15, 15));
    }
}
```

CS2704 (Spring 2000) 22

## Example (cont.)

```
// Create the message component and add it to the window
label = new MultiLineLabel(message, 20, 20);
this.add("Center", label);

// Create an Okay button in a Panel;
// add the Panel to the window
// Use a FlowLayout to center the button
// and give it margins.
button = new Button("Okay");
Panel p = new Panel();
p.setLayout(new FlowLayout(FlowLayout.CENTER, 15, 15));
p.add(button);
this.add("South", p);
this.pack(); // resize window
}
```




CS2704 (Spring 2000) 23

## Example (cont.)

```
// Pop down the window when the button is clicked.
// Java 1.0 event handling

public boolean action(Event e, Object arg)
{
    if (e.target == button) {
        this.hide();
        this.dispose();
        return true;
    } else return false;
}
```

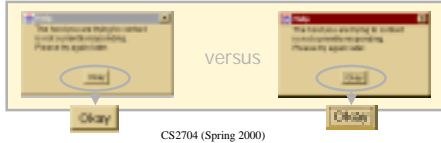
`button = new Button("Okay");`



CS2704 (Spring 2000) 24

## Example (cont.)

```
// When the window gets the keyboard focus,  
// give it to button.  
// This allows keyboard shortcuts to pop down the dialog.  
  
public boolean gotFocus(Event e, Object arg) {  
    button.requestFocus();  
    return true;  
}
```



## Java 1.1 Event Handling

```
java.awt.event.ActionEvent  
java.awt.event.AdjustmentEvent  
java.awt.event.ComponentEvent  
java.awt.event.ContainerEvent  
java.awt.event.FocusEvent  
java.awt.event.InputEvent  
java.awt.event.KeyEvent  
java.awt.event.MouseEvent  
java.awt.event.PaintEvent  
java.awt.event.WindowEvent  
java.awt.event.ItemEvent  
java.awt.event.TextEvent
```

CS2704 (Spring 2000)

26

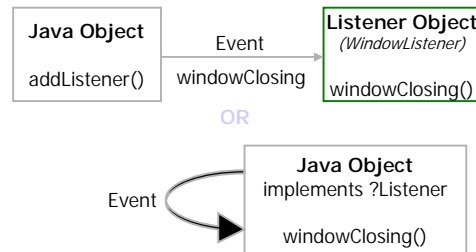
## Java 1.1 Event Handling

```
java.awt.event.ActionListener  
java.awt.event.AdjustmentListener  
java.awt.event.ComponentListener  
java.awt.event.ContainerListener  
java.awt.event.FocusListener  
java.awt.event.ItemListener  
java.awt.event.KeyListener  
java.awt.event.MouseListener  
java.awt.event.MouseMotionListener  
java.awt.event.TextListener  
java.awt.event.WindowListener
```

CS2704 (Spring 2000)

27

## Java 1.1 Event Handling



## interface java.awt.event.MouseMotionListener

```
public abstract interface MouseMotionListener  
    extends EventListener {  
  
    // Public Instance Methods  
    public abstract void mouseDragged(MouseEvent e);  
    public abstract void mouseMoved(MouseEvent e);  
}
```

CS2704 (Spring 2000)

29

## interface java.awt.event.MouseListener

```
public abstract interface MouseListener  
    extends EventListener {  
  
    // Public Instance Methods  
    public abstract void mouseClicked(MouseEvent e);  
    public abstract void mouseEntered(MouseEvent e);  
    public abstract void mouseExited(MouseEvent e);  
    public abstract void mousePressed(MouseEvent e);  
    public abstract void mouseReleased(MouseEvent e);  
}
```

CS2704 (Spring 2000)

30

## Illustration of event handling: Scribble Applet



CS2704 (Spring 2000)

31

## Example: Scribble Applet

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

public class Scribble extends Applet
    implements MouseListener, MouseMotionListener {
    private int last_x, last_y;

    public void init() {
        // Tell this applet what MouseListener and MouseMotionListener
        // objects to notify when mouse and mouse motion events occur.
        // Since we implement the interfaces ourself,
        // our own methods are called.
        this.addMouseListener(this);
        this.addMouseMotionListener(this);
    }
}
```

CS2704 (Spring 2000)

32

## Example (cont.)

```
// A method from the MouseListener interface.
// Invoked when the user presses a mouse button.
public void mousePressed(MouseEvent e) {
    last_x = e.getX();
    last_y = e.getY();
}

// A method from the MouseMotionListener interface.
// Invoked when the user drags the mouse
// with a button pressed.
public void mouseDragged(MouseEvent e) {
    Graphics g = this.getGraphics();
    int x = e.getX(), y = e.getY();
    g.drawLine(last_x, last_y, x, y);
    last_x = x; last_y = y;
}
```

CS2704 (Spring 2000)

33

## Example (cont.)

```
// The other, unused methods of the MouseListener interface.
public void mouseReleased(MouseEvent e) {}
public void mouseClicked(MouseEvent e) {}
public void mouseEntered(MouseEvent e) {}
public void mouseExited(MouseEvent e) {}

// The other method of the MouseMotionListener interface.
public void mouseMoved(MouseEvent e) {}
}
```

CS2704 (Spring 2000)

34

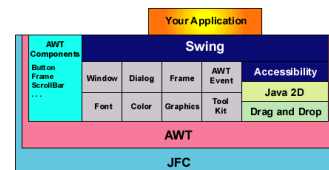
## Java JFC

- Java Foundation Classes (or JFC) is a comprehensive set of graphical user interface classes
- JFC is comprised of:
  - AWT
  - Swing – More GUI widgets based on AWT
  - Java 2D – 2D graphical routines
  - Drag and Drop
  - Accessibility APIs

CS2704 (Spring 2000)

35

## Swing and AWT



CS2704 (Spring 2000)

36