


IS4300 HCI



Non-Quiz

What does "JFC" stand for?

- Java Fundamental Classes
- Java Foundation Creator
- Java Fried Chicken
- Java Foundation Classes
- Java Framework Creator

Non-Quiz

What is "pluggable look and feel"?

- Swing components can be plugged in to each other.
- Swing components can be embedded in each other.
- Easy tailoring of individual Swing components to look and behave uniquely.
- Easy switching between Windows, Mac, and other "look and feels"
- Library of Swing plug-ins.

Ethnography Homework





Which design principles do GUIs support?

1. Feedback
2. Speak the User's Language
3. Clearly Marked Exits
4. Consistency
5. Prevent Errors
6. Minimize User Memory Load
7. Flexibility / Shortcuts
8. Simple Design
9. Good Error Messages
10. Help and Documentation
11. Use Appropriate Affordances
12. Visibility / Obviousness



What's a GUI?

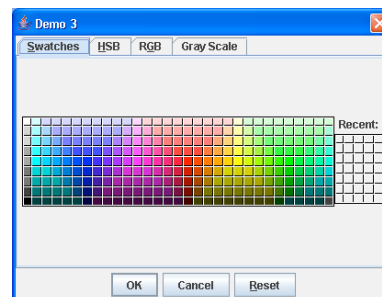
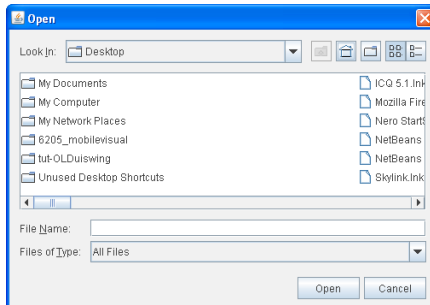
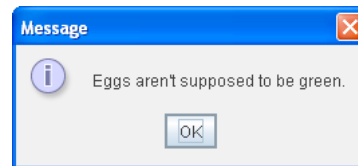
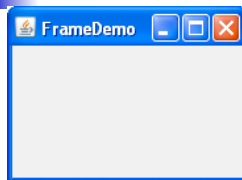
Standard Elements of a GUI

- WIMP
 - Windows
 - Icons
 - Menus
 - Pointers

What is a Widget? Examples?

- The "Macintosh 7" 1984
 - Button
 - Slider
 - Pulldown menu
 - Check box
 - Radio buttons
 - Text entry fields
 - File pick/save
- *These have become standard*

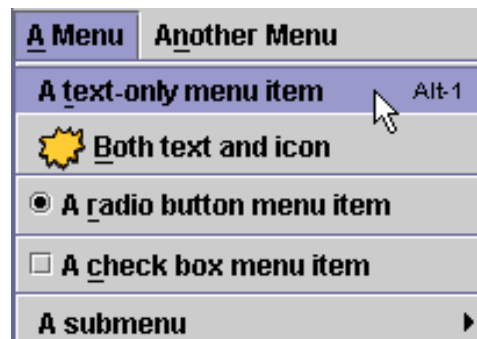
WIMP Elements in Swing: Windows



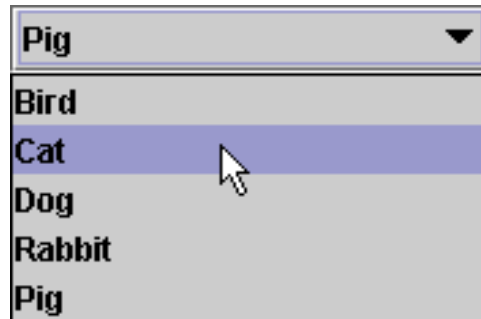
WIMP Elements in Swing: Buttons



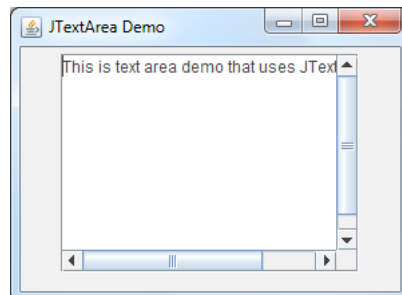
WIMP Elements in Swing: Menus



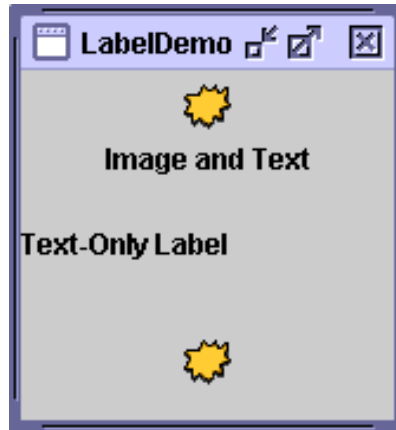
WIMP Elements in Swing: Combo Box



WIMP Elements in Swing: Text Field / Area



WIMP Elements in Swing: Labels



WIMP Elements in Swing: Tool tips



WIMP Elements in Swing: Embedded Panels

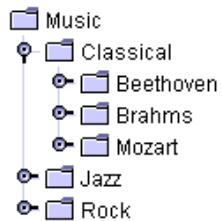


A Label on a Panel

Color and font test:

- ◆ red
- ◆ blue
- ◆ green
- ◆ small

WIMP Elements in Swing: Advanced



First Name	Last Name	Favorite Food
Jeff	Dinkins	
Ewan	Dinkins	
Amy	Fowler	
Hania	Gajewska	
David	Geary	



Pros and Cons of "Standard" Widget Sets?

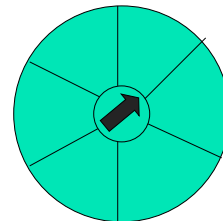
- Pro: Collection of good interaction techniques that work well
 - uniformity is good for usability
 - Improves external consistency

- Cons: Significant stagnation
 - Failing to customize interaction techniques to tasks
 - Efficiency could be improved

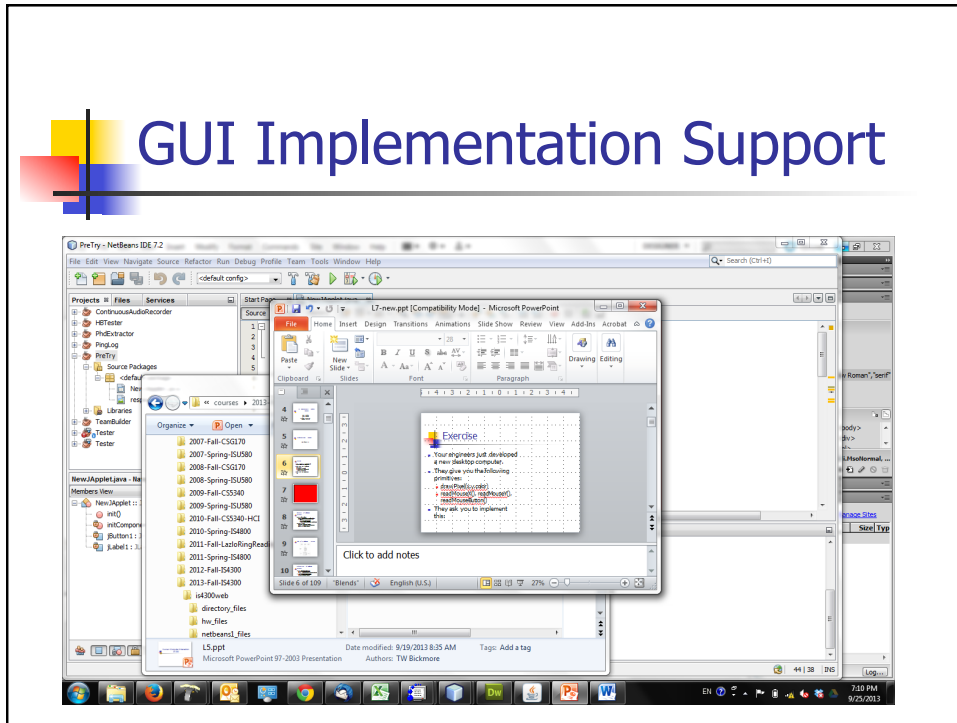
Example of non-standard widget: Pie menus

- A circular pop-up menu
 - only angle of mouse motion counts
 - *Maya, Blender, Grand Theft Auto V, Android Browser*

- What are Fitts' law properties?
 - minimum distance to travel
 - minimum required accuracy (dependent on # of options)
 - very fast (dependent on # of options)

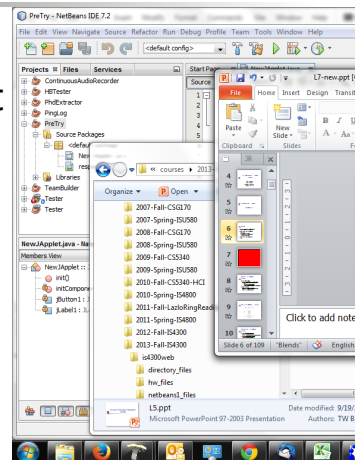


GUI Implementation Support

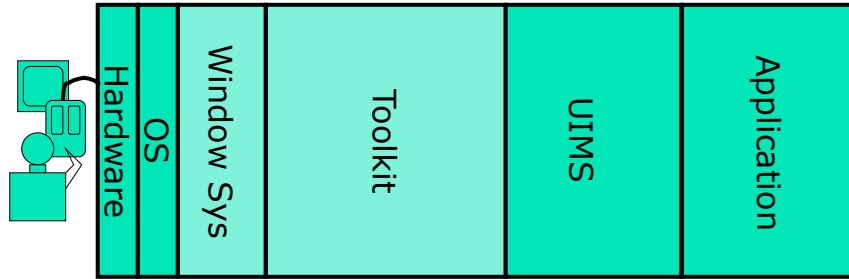


Exercise

- Your engineers just developed a new desktop computer, and want a modern GUI on it.
- They give you the following primitives:
 - drawPixel(x,y,color)
 - readMouseX(), readMouseY(), readMouseButton(), readKey()
- What functionality do you need?



Myers: Levels of Abstraction in UI Software



UIMS Example: Labview

This screenshot shows a LabVIEW UIMS example for a control system. The interface includes:

- Control Panel:** Start/Stop, Serial Port, Program Running, and four channels with Refresh ADC Input buttons and status indicators.
- Block Diagram:** A control loop starting with a Set Point (Meters) control, followed by a Setpoint Gain multiplier, a Theta >> input, and a Control >> block containing an LP Plant Model. The output is visualized using Picture and Create Visualization blocks.

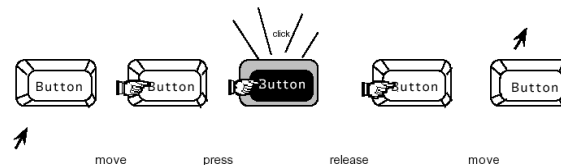
Myers' Chapter

Graphical User Interface Programming

- Why use GUI tools?
 - Makes authoring easier & more economical
 - Improves quality
 - Improves usability

OOP and Toolkit Widgets

- Why are they so well suited?
 - Natural metaphor (direct manipulation)
 - Subclassing to create custom widgets
 - Encapsulation (data & behavior)



OO Toolkit Concepts

#1 Specialization via Subclassing

```

java.lang.Object
  java.awt.Component
    java.awt.Container
      javax.swing.JComponent
        javax.swing.text.JTextComponent
          javax.swing.JTextField
            javax.swing.JTextArea
  
```

Years:

OO Toolkit Concepts

#2 Composition


- Put together interactive objects at larger scale than atomic interactors
- Container objects

Inner Panel

Coke

Sprite

Coffee



Inner Inner Panel

Ice

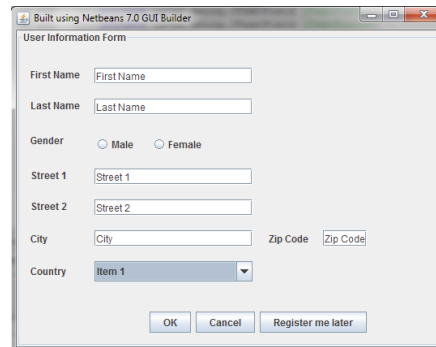
No Ice

Hot

Order!

OO Toolkit Concepts #3 Layout

- How a container organizes its widgets within itself.



Build using NetBeans 7.0 GUI Builder

User Information Form

First Name

Last Name

Gender Male Female

Street 1

Street 2

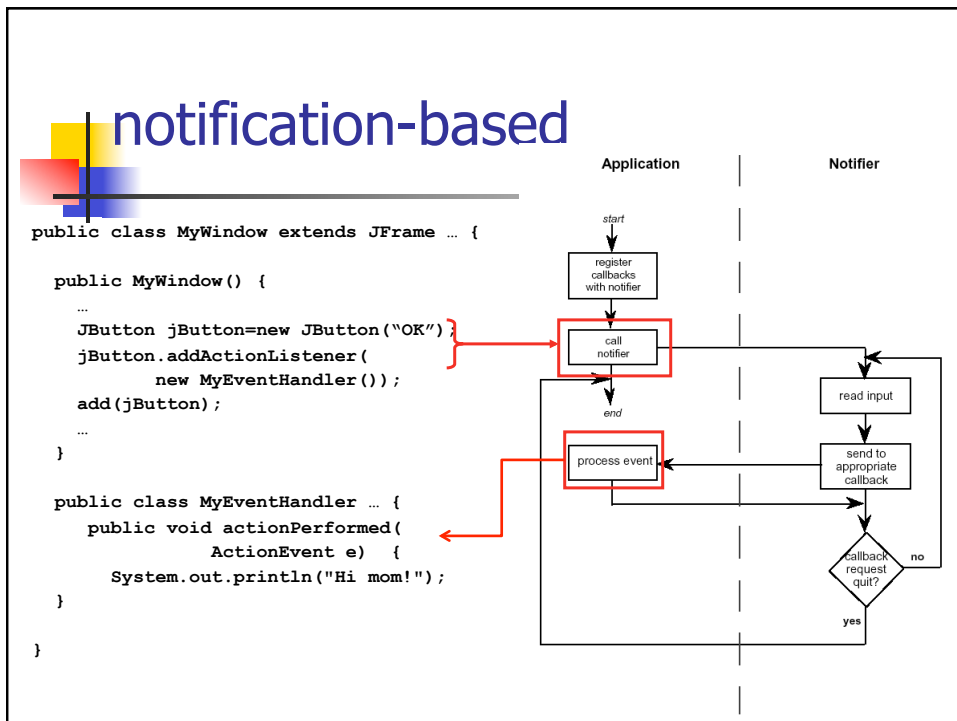
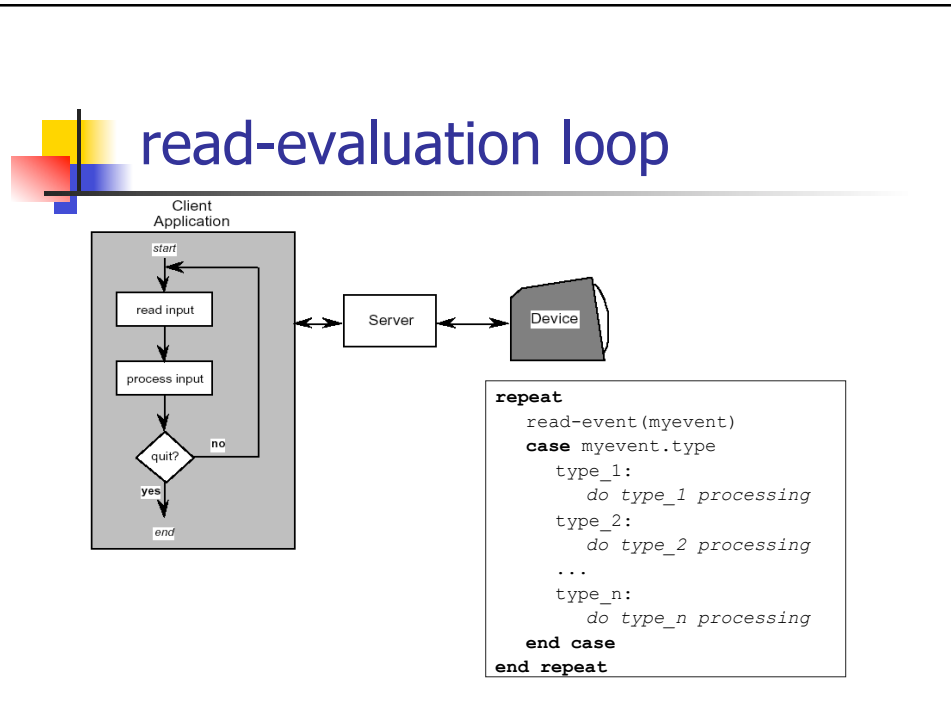
City Zip Code

Country

OK Cancel Register me later

OO Toolkit Concepts #4 Event Handling

1. When anything happens in the UI
 - Mouse clicked, Window moved, Key pressed, etc
2. Windowing System creates a record
3. The event record is added to a UI event queue
4. The application (or toolkit) pulls events from the queue and acts on them in order

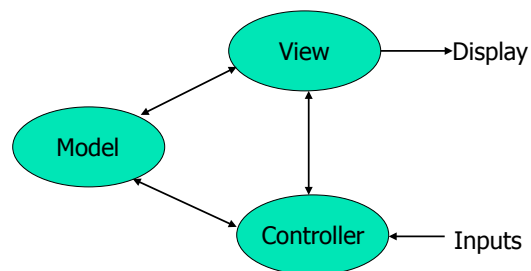


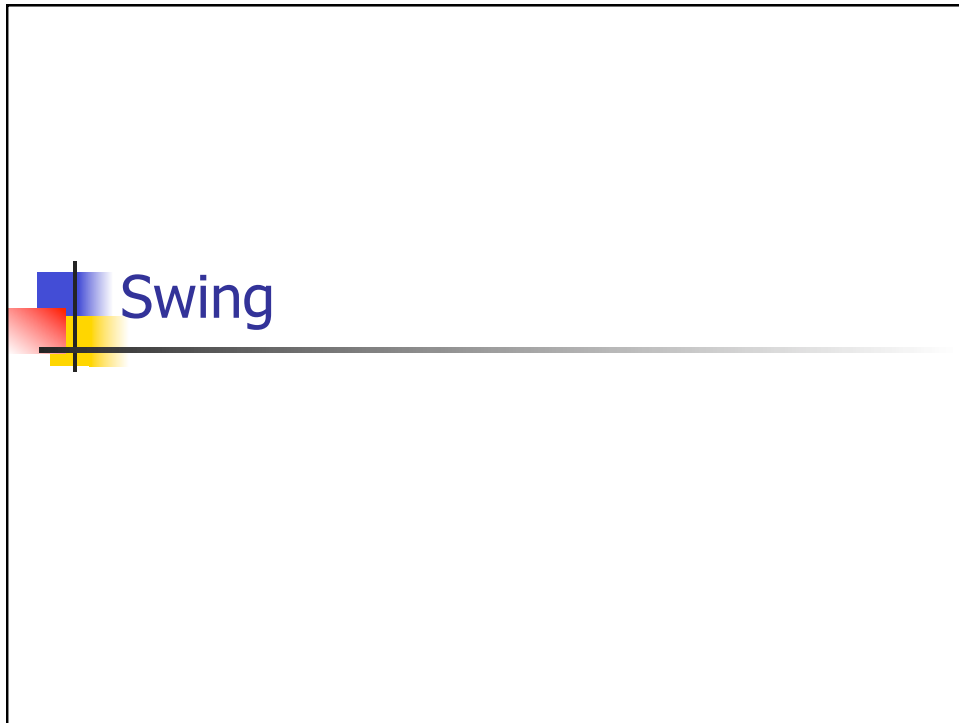
OO Toolkit Concepts

#5 Virtual toolkits & MVC


- Aka Cross-Platform Development Systems
- Provide a layer of abstraction above "native" toolkit
- Two approaches:
 - Map to native widgets
 - Provide own widgets

Model-View-Controller Architecture for Widgets





Java GUI APIs

- 
- AWT
 - The original – now mostly obsolete as a toolkit (event handling mechanism still used in Swing)
 - Used “heavyweight” components
 - Swing (~1997)
 - The current(?) standard.
 - Native window, but draws all widgets
 - Pluggable look-and-feel
 - SWT (Standard Widget Toolkit)
 - Open source widget toolkit.
 - JavaFX
 - Becoming new standard UI toolkit (?)

Pluggable Look-and-Feel

The image displays four separate SwingApplication windows, each illustrating a different look-and-feel theme. Each window contains a button labeled "I'm a Swing button!" and a text label "Number of button clicks: 4".

- Java:** Standard Windows-style appearance with a light gray background and a standard button.
- GTK+:** Dark theme with a black background and a button that has a dashed border.
- Windows:** Light theme with a light gray background and a standard button.
- Mac:** macOS-style appearance with a light gray background, rounded corners, and a button with a blue gradient and rounded corners.

Netbeans

The screenshot shows the NetBeans IDE interface for a project named "ContactEditor". The interface is divided into several panes:

- Projects:** A tree view on the left showing the project structure, including "Source Packages", "my.contacteditor", and "ContactEditorUI.java".
- Navigator:** A pane below the Projects pane showing a hierarchical view of the "Form ContactEditorUI" component, including "Other Components" and "[JFrame]".
- Source Editor:** The central area showing the source code for "ContactEditorUI.java". A tooltip indicates: "The Inspector window displays a tree hierarchy of components in the opened form." Below this is a large gray box labeled "Visual Design & Source Editors".
- Palette:** A pane on the right showing a collection of Swing components like "Panel", "Split Pane", "Tool Bar", "Text Field", etc., under the heading "Swing Containers".
- Properties:** A pane on the right showing the properties for the selected component, "Form ContactEditorUI", including "Code Generation", "Variables Modifier", "Local Variables", "Generate Fu", "Generate Im", "Listener Gen", "Layout Generation Style", "Resources and Internationalization", and "Set Component Names".



Group Exercise

- Project Teams
- Brainstorm alternative “main screens” for your project



GUI Implementation Support

Learning Objectives:

- Why GUIs?
- What is a GUI?
- Why is implementation support needed?
- What kinds of implementation support are available?
- Basic concepts in OO GUI toolkit
- Basics of Java Swing
- Use Netbeans to Design a GUI



Swing Homework I4 – Create a Restaurant Ordering App

- Two JLabels, one with an icon.
- Two JButtons, one with an icon.
- Etc.

- Email a screen shot of your app running, together with a zip archive of your project directory to is4300f16@

- Due in 1 week



To do

- Read
 - Design I (Benyon Ch 5 & 9).
- Due Monday: P2 – Requirements Analysis
- Start Homework I4 – Swing & Netbeans