

Announcements for This Lecture

<h3 style="text-align: center;">Prelim II</h3> <ul style="list-style-type: none"> • Tonight in Olin 155 <ul style="list-style-type: none"> ▪ 7:30 – 9 pm ▪ Will be graded tonight ▪ Grades by tomorrow • Make-up Thursday 4:30 <ul style="list-style-type: none"> ▪ For students that contacted me with prelim conflicts ▪ Graded by the weekend • Review slides are online 	<h3 style="text-align: center;">Assignments</h3> <ul style="list-style-type: none"> • A6 due Thursday <ul style="list-style-type: none"> ▪ Hopefully you have done all but Steganography ▪ To be graded this weekend • Assignment A7 posted Friday <ul style="list-style-type: none"> ▪ Last assignment of semester ▪ Sizeable project; longer than the previous ones ▪ Will give you until Saturday after last day of classes
---	---

4/17/12
GUI Layout
1

The Limitations of JFrame

- JFrame is just a Window
 - Can resize it
 - Can close it
 - Not much else
- To do more, you need **GUI components**
 - Items inside a JFrame
 - Ex: Buttons, Text Boxes
- Two main Java packages
 - `java.awt`: “old GUI”
 - `javax.swing`: “Swing GUI”

4/17/12
GUI Layout
2

AWT vs. Swing

<h3 style="text-align: center;">Abstract Window Toolkit</h3> <ul style="list-style-type: none"> • Uses the standard interface <ul style="list-style-type: none"> ▪ Mac looks like Mac ▪ Windows like Windows • Violates Java “portability” <ul style="list-style-type: none"> ▪ Demo: AWTFile.java • Very rarely used today <ul style="list-style-type: none"> ▪ Handling input is messy ▪ But superclass of Swing classes, so have to include 	<h3 style="text-align: center;">Swing API</h3> <ul style="list-style-type: none"> • Codename that “stuck” • Has pluggable look & feel <ul style="list-style-type: none"> ▪ Mac can look like Windows ▪ Default same on all platforms ▪ Demo: SwingFile.java • Now the default component collection in Java <ul style="list-style-type: none"> ▪ Very easy to use ▪ Programmers like uniformity
--	--

4/17/12
GUI Layout
3

Swing Components

<p>JButton: a pushbutton that can be clicked by mouse</p> <p>JCheckBox: can be on (true) or off (false)</p> <p>JComboBox: a popup menu of user choices</p> <p>JLabel: a text label</p> <p>JList: scrolling list of user-chooseable items</p> <p>JScrollbar: a scroll bar</p> <p>JTextField: allows editing of a single line of text</p> <p>JTextArea: multiline region for displaying and editing text</p>	<p>JPanel: used for containing and grouping components</p> <p>JDialog: window used for user input</p> <p>JFrame: top-level window with frame and border</p>
--	--

4/17/12
GUI Layout
4

Main Challenges in GUI Applications

<h3 style="text-align: center;">Layout</h3> <ul style="list-style-type: none"> • Arranging items the screen <ul style="list-style-type: none"> ▪ Java has many components ▪ But where do they go? • Challenge: Resizing <ul style="list-style-type: none"> ▪ Want components to “behave nicely” as you resize ▪ Change size of components ▪ Change padding in between • LayoutManagers do both 	<h3 style="text-align: center;">Input Handling</h3> <ul style="list-style-type: none"> • Many types of input <ul style="list-style-type: none"> ▪ button pushed ▪ text typed ▪ mouse clicked ... • Want app to react to input <ul style="list-style-type: none"> ▪ Otherwise GUI looks pretty, but does nothing • Topic of next lecture
---	---

4/17/12
GUI Layout
5

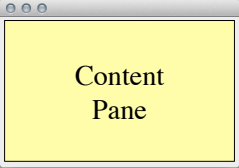
Example: BorderLayout

- Container has 5 directions
 - When add, specify direction
 - Ex: BorderLayout.CENTER
 - One component to a direction
- Resizing stretches components
 - Sides stretch vertically
 - Bottom/top stretch horizontally
 - Center stretches in both ways
- **Demo:** TestBorder.java

4/17/12
GUI Layout
6

Adding Components to JFrames


- Never add to JFrame directly.
 - `getContentPane().add(...)`
- `getContentPane()` provides **double buffering**
 - Draws components to image, then displays image
 - Faster way to redraw if window moves



4/17/12 7

Alternate Layouts: FlowLayout


- Use a left-to-right “flow”
 - If one row fills, start on next row
 - Components “wrap” around
 - By default, components are centered in the container.
- Resizing affects layout only
 - Components retain their size
 - But the wrap may be affected
- **Demo:** TestFlow.java



4/17/12 GUI Layout 8

Alternate Layouts: GridLayout

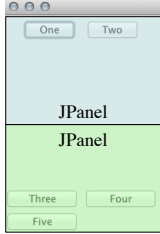
- Uses a rectangular grid
 - Specify no. of rows, columns
 - Ex: `new GridLayout(3,2);`
 - Tries to fill each gridbox
 - Leaves blanks if cannot
- Resizing affects components
 - Stretched to equal size
 - Cannot make different sizes
- **Demo:** TestGrid.java



4/17/12 GUI Layout 9

Nesting Layouts

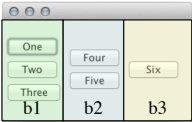
- Want more interesting layouts
 - **Idea:** nest layouts in each other
 - Can get fine padding control
- Useful class: JPanel
 - Invisible component
 - Container for other components
 - Can take a LayoutManager
- **Demo:** PanelGrouping.java



4/17/12 GUI Layout 10

BoxLayout: Ideal for Nesting

- BoxLayout
 - Arranges components in line
 - No wrap (like FlowLayout)
 - Either horizontal/vertical
- Box: JPanel w/ BoxLayout
 - Box `b1 = new Box(BoxLayout.Y_AXIS);`
 - Makes layout quick
- **Demo:** BoxGrouping.java

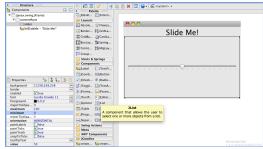


- Nested boxes
 - Three vertical boxes
 - Inside horizontal box

4/17/12 GUI Layout 11

Designing GUIs in the Eclipse IDE

- Integrated Development Environment
 - Editor for managing complex programs
 - Often have debuggers
 - DrJava is simple example
- Eclipse is THE Java IDE
 - Use in later classes
 - Defacto IDE in industry
- Has tools for visual GUI design
 - Layout code generated automatically



4/17/12 GUI Layout 12