User Interface Components

Layout

- Placement of UI components in a window
  - Size & Position
- Each component occupies a rectangular region in the window
- Components are told where and how big they are
- Each component has a
  - Preferred size
  - Minimum size
  - Maximum size

Layout

- Ideally, each component will be assigned its preferred size
- To accommodate the current window size, a component may get more or less space than it prefers
- Window resize behavior
  - When the user resizes a window, the sizes and positions of components in the window must be adjusted to accommodate the new size

Panels

- The area of a frame window can be divided into rectangular regions called “panels”
  - Implemented by the JPanel class
- Each panel contains sub-components
  - A panel is a “container” component, because it contains other components
- Component Tree
  - Frame window at the root
  - Panels for the interior nodes
  - Simple components (e.g., buttons, text fields, etc.) at the leaves
- Each panel’s components are lain out separately

Layout Managers

- There are different algorithms for laying out components in a panel
- Each panel has a “layout manager” object that is responsible for deciding the sizes and positions of the panel’s components
- Java provides several LayoutManager classes that implement different layout algorithms
  - BorderLayout, BoxLayout, CardLayout, FlowLayout, GridLayout, GridBagLayout, GroupLayout, SpringLayout, …

Simple Layout

- FlowLayout
  - Example: FlowTest
- BorderLayout
  - Example: BorderTest
- GridLayout
  - Example: GridTest

Built-in Components

- Guide to Swing Components
- Component Events
  - User actions generate events
  - Listeners respond to events
  - Example: Font Selector
    - ActionListener, ChangeListener
  - Listeners Supported by Swing Components

Design Exercise: Simple Web Browser

- Web Browser
- Web Browser (Refactored)