Lecture 13: Toolkits

UI Hall of Fame or Shame?

Today's Topics

- Widgets
- Toolkit layering
- Look-and-feel
**Widgets**

- Reusable user interface components
  - Also called controls, interactors, gizmos, gadgets
- Widget is a view + controller
  - Embedded model
    - Application data must be copied into the widget
    - Changes must be copied out
  - Linked model
    - Application provides model satisfying an interface
    - Enables "data-bound" widgets, e.g. a table showing thousands of database rows, or a combo box with thousands of choices

**Widget Pros and Cons**

- **Advantages**
  - Reuse of development effort
    - Coding, testing, debugging, maintenance
    - Iteration and evaluation
  - External consistency
- **Disadvantages**
  - Constrain designer’s thinking
  - Encourage menu & forms style, rather than richer direct manipulation style
  - May be used inappropriately

**Example**

- Cheese sandwich delivery
  - Bread (choose one): white, wheat, rye
  - Cheese (choose one or more): swiss, cheddar, provolone, american, brie, goudda
  - Toasted: yes or no
  - Deliver to MIT building (choose one):
    - Baker, EC, Next, Simmons, Stata, ...

**Widgets for 1-of-N Choices**

- N radio buttons
- N mutually-exclusive toggle buttons
- Drop-down list
- Single-selection listbox
Widgets for 1-of-2 Choices

- Checkbox
- Toggle button

Widgets for K-of-N Choices

- N checkboxes
- N toggle buttons
- Multiple-selection listbox
- 2 listboxes

Widgets for Commands

- Menubar menu item
- Toolbar button
- Context menu item
- Keyboard shortcut
- Push button
- Hyperlink

- Action object (aka command)
  - Tooltips on mouseover
  - Enable/disable

Widgets for Window Organization

- Mutually-exclusive panes
  - Tabbed panes
  - Listbox + pane
- Multiple content panes
  - Splitters
- Scrolling content
  - Scroll panes
  - Paging buttons
Widgets for Dialogs

- Modal dialog box
- Modeless dialog box
- Sidebar
- Sheet

Toolkits

- User interface toolkit consists of:
  - Components (view hierarchy)
  - Stroke drawing
  - Pixel model
  - Input handling
  - Widgets
  - (often) Automatic layout

Toolkit Examples

<table>
<thead>
<tr>
<th>components</th>
<th>MS Win windows</th>
<th>Swing JComponents</th>
<th>HTML elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>strokes</td>
<td>GDI Graphics</td>
<td>-- (none)</td>
<td></td>
</tr>
<tr>
<td>pixels</td>
<td>bitmaps Image</td>
<td>inlined images</td>
<td></td>
</tr>
<tr>
<td>input</td>
<td>messages</td>
<td>listeners</td>
<td>Javascript event handlers</td>
</tr>
<tr>
<td>widgets</td>
<td>button, menu,</td>
<td>JButton, JMenu,</td>
<td>form controls &amp; links</td>
</tr>
<tr>
<td></td>
<td>textbox, ...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
**Cross-Platform Toolkit Layering**

```
MS Windows  Motif  Mac OS
  |      |     |
  
  AWT
  |      |
  Swing
  |      |
  subArctic
```

**Cross-Platform Widgets: AWT Approach**

- AWT, HTML
  - Use native widgets, but only those common to all platforms
    - Tree widget available on MS Win but not X, so AWT doesn’t provide it
  - Very consistent with other platform apps, because it uses the same code

```
java.awt.List      peer   MSWin.List
```

**Cross-Platform Widgets: Swing approach**

- Swing, Amulet
  - Reimplement all widgets
  - Not constrained by least common denominator
  - Consistent behavior for application across platforms
Pluggable Look-and-Feel

- Swing also reimplements platform look-and-feel

```
java.awt.List    delegate  painting
               install    components & listeners
                MetaListUI  WindowsListUI
```

Cross-Platform Widgets: SWT Approach

- SWT
  - Use native widgets where available
  - Reimplement missing native widgets