Software for Building User Interfaces

Look under the hood

Agenda

- UI Software Principles
- Styles of tools
  - UI toolkits
  - GUI builder tools

User Interface Software

- What support is provided for building graphical user interfaces?
  - UI toolkits
  - GUI builder tools

- Let’s examine some background...
GUI System Architecture

What does it look like?

Layered Architecture

- Higher level Tool
- UI Toolkit
- Window System
- OS
- Hardware

Window System

- Virtual display abstraction
- Coordinates different input devices
- Provides window primitives
- Important components
  - Graphics model
  - Input model
- May or may not include window manager

UI Toolkit

- What application programmer typically programs with
- Combination of interface objects and management behaviors
- Usually object-oriented
- Library of software components and routines that programmer puts together
  - X Windows: X Toolkit & Motif
  - Macintosh: Mac Toolbox, MacApp
  - Windows: Windows Developers’ Toolkit
  - Java: Swing
Higher Level Tools

- Provide assistance or some automation in developing UIs
- Many names
  - User Interface Management System (UIMS)
  - User Interface Builder
  - User Interface Development Environment

Separation of Concerns

- Application
  - Core functionality
  - Operations
  - Data
- Interface
  - Interface components
  - Graphics
  - I/O

Should these be separated in code?
Why?
Why not?

How Does a Toolkit Work?

- What exactly does it provide?
- How is it organized?

Toolkit Workings

- User takes actions, interacts with interface
- Those actions must be delivered to application in meaningful ways
- Application takes appropriate actions, perhaps updating display
Seeheim Model

Conversational model

DOMINANT model for long time

Object Model

- UI is collection of interactor objects (often called widgets)
- User directly manipulates them
- Objects responsible for transmitting user actions to application in meaningful ways

Locus of Control

- "Traditional" software
  - Control is in system, query user when input needed
- Event-driven software
  - Control is with user (wait for action)
  - Code reacts to user actions
  - More difficult to write code this way, harder to modularize, etc.

Event-Driven Program

- Initialize display & system
- Repeat
  - Wait for and get next user action
  - Decipher action
  - Take appropriate action
  - Update display
- Until Done
Event-Driven Program

Callback Routine

- Software procedure, part of application
- Invoked when particular action occurs to UI component, such as pressing a PushButton
- Procedure is invoked with event parameters

Example – X & Motif

- Object-oriented hierarchy of UI interactors called widgets
  - Associate callback routines in your code with them
  - Interface is built up by putting child widgets “onto” parent widgets

Widget

Graphical user interface interactor object
Widget Hierarchy

- Widgets organized into inheritance hierarchy

```
Primitive
  |___ Text
  |___ Label
  |___ Button
  |___ Scroll Bar
  |___ Push Button
  |___ Drown Button
  |___ Toggle Button
```

- Visual appearance
- Set of tailorable attributes

```
PushButton {
  Color Background;
  int MarginLeft;
  int MarginRight;
  int BorderWidth;
  Pixmap ArmPixmap;
  Boolean FillOnArm;
  CallbackList ActivateCallback;
}
```

- Interactive behavior

Widget Use

- Set up widget attributes
- Create widget object (as child of parent widget)
- Define callback or event procedure for widget

```
n = 0;
        xmstr = XmStringCreate("Color", XmSTRING_DEFAULT_CHARSET);
        XtSetArg(args[n], XmNlabelString, xmstr); n++;
        XtSetArg(args[n], XmNbackground, red); n++;
        colorbut = XtCreateManagedWidget("colorbutton",
                                          XmPushButtonWidgetClass, focusrowcol, args, n);
        XtAddCallback(colorbut, XmNactivateCallback, colorChangeCB, id);
```

Widget and Callback

```
void
    colorChangeCB(Widget w, XtPointer userdata, XtPointer eventdata)
    { // Actions }
```
Main Program Event Loop

```java
void CheckXEvents()
{
    XEvent xev;
    while (XtAppPending(_context)) {
        XtAppNextEvent(_context, &xev);
        XtDispatchEvent(&xev);
    }
}
```

OO Systems

- Java’s GUI programming done with AWT and Swing
- More distributed model (separate threads)
- Primary action here is dispatching events to objects (widgets) as messages
- Delegation important
  - Can make particular objects responsible for event handling

Example - Java AWT

```java
public void mouseReleased(MouseEvent e){
    System.out.println("Changing color");
    if (bHighlight)
        frame.setBackground(highlight);
    else
        frame.setBackground(normal);
    bHighlight = !bHighlight;
}
```

GUI Builder Tools

- Why build graphical (visual) interface with textual commands?
- Why not show what you want it to look like?
- Visual builder tools: Visual Basic, Visual C++, Borland Delphi, Symantec Café
Tool Methods

- Work area (interface being built)
- Drag and drop interactors/widgets onto work area
- Specify position, color, look, etc.
- Often provide Build/Test modes

Example: dtbuilder (Motif)

Visual Basic

- Microsoft tool for building Windows UIs
- Not most powerful, but pretty easy to use
- Let’s see more details...
  - Might want to use it in your project
Compiling an Executable

Making Menus

Alignment

Different Controls

- Checkboxes
- Scrollbars
- ...
Interested in This?

- Take CS 6456, Intro to UI Software
- Should have a good programming background