USER INTERFACE SOFTWARE

Styles of Tools

- Design Tools
- User Interface toolkits
- Graphic User Interface Builder Tools

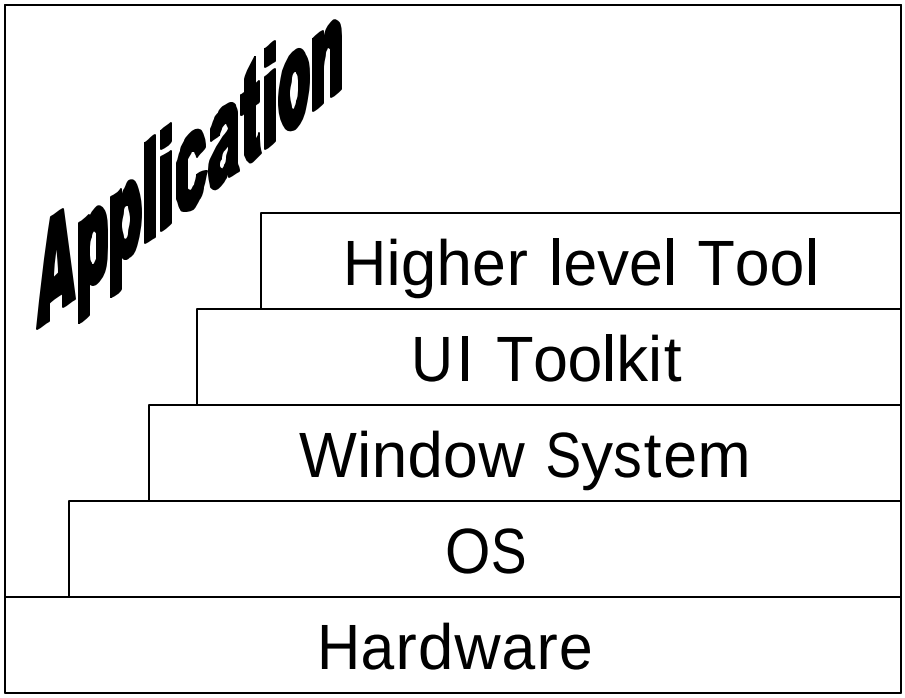
User Interface Software

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

GUI System Architecture

What does it look like?

Layered Architecture



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**

User Interface Toolkit

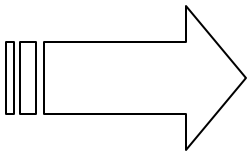
- **What application programmer typically programs with**
- **Combination of interface objects and management behaviors**
- **Usually object-oriented now**
- **Library of software components and routines that programmer puts together**
  - X Windows: *X Toolkit & Motif*
  - Macintosh: *Mac Toolbox, MacApp*
  - 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?

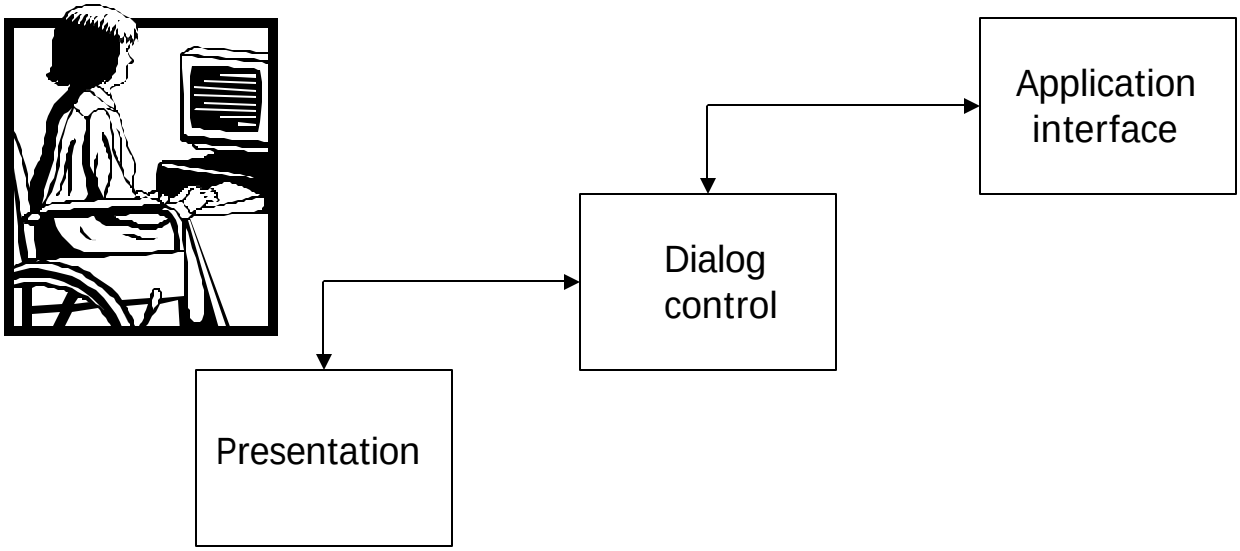
- **What exactly does it provide?**
- **How is it organized?**

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

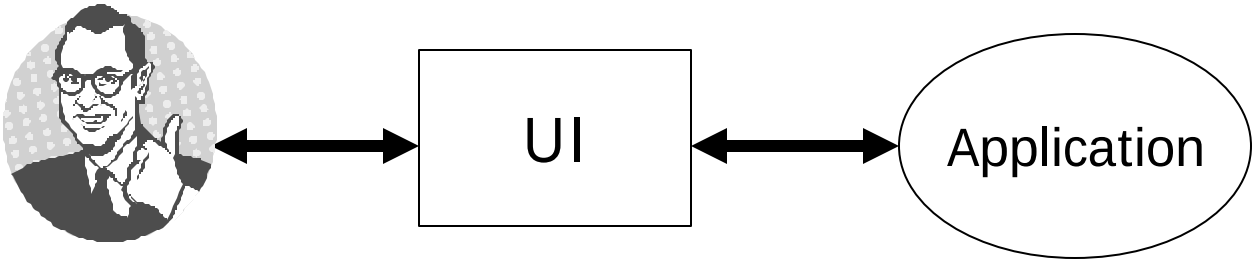
Conversational model



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



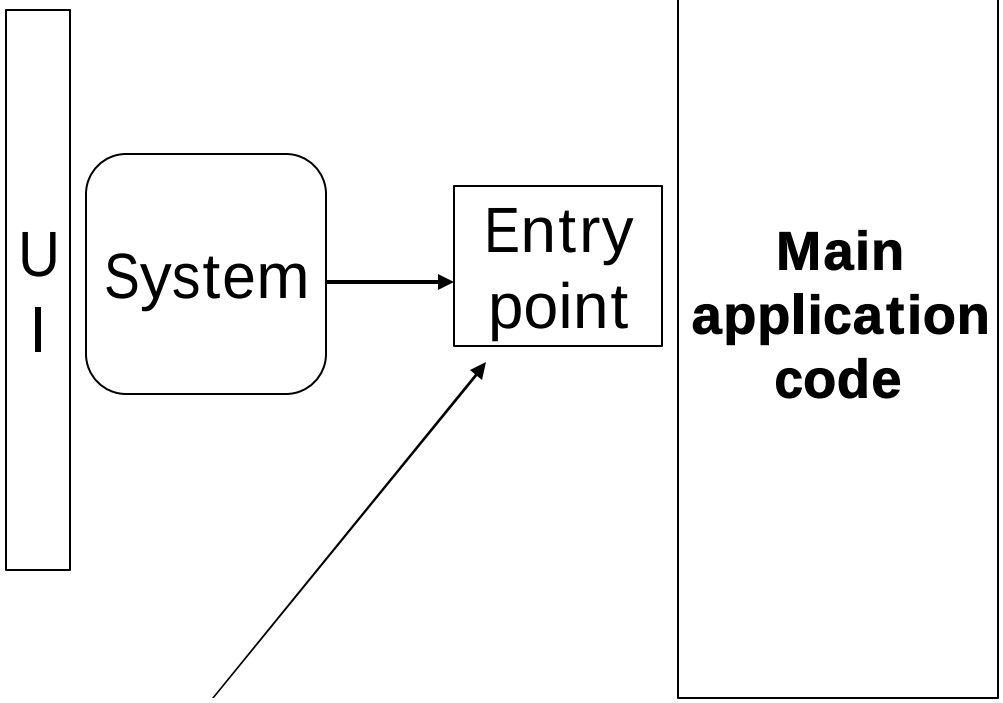
User

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**



Callback Routine

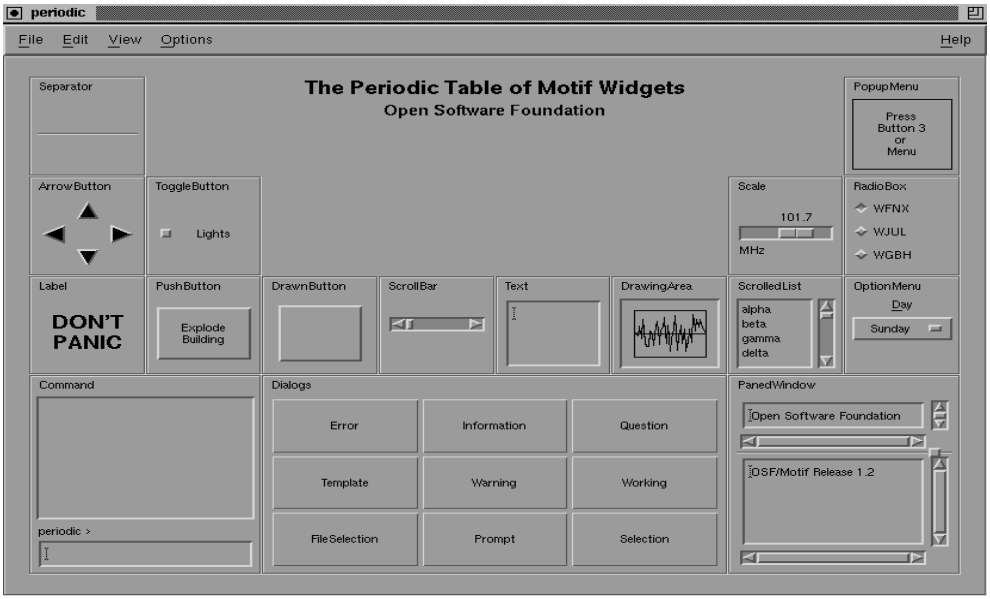
- **Software procedure, part of application**
- **Invoked when particular action occurs to UI component, such as pressing a PushButton**
- **Procedure is invoked with event parameters**

Example – X & Motif

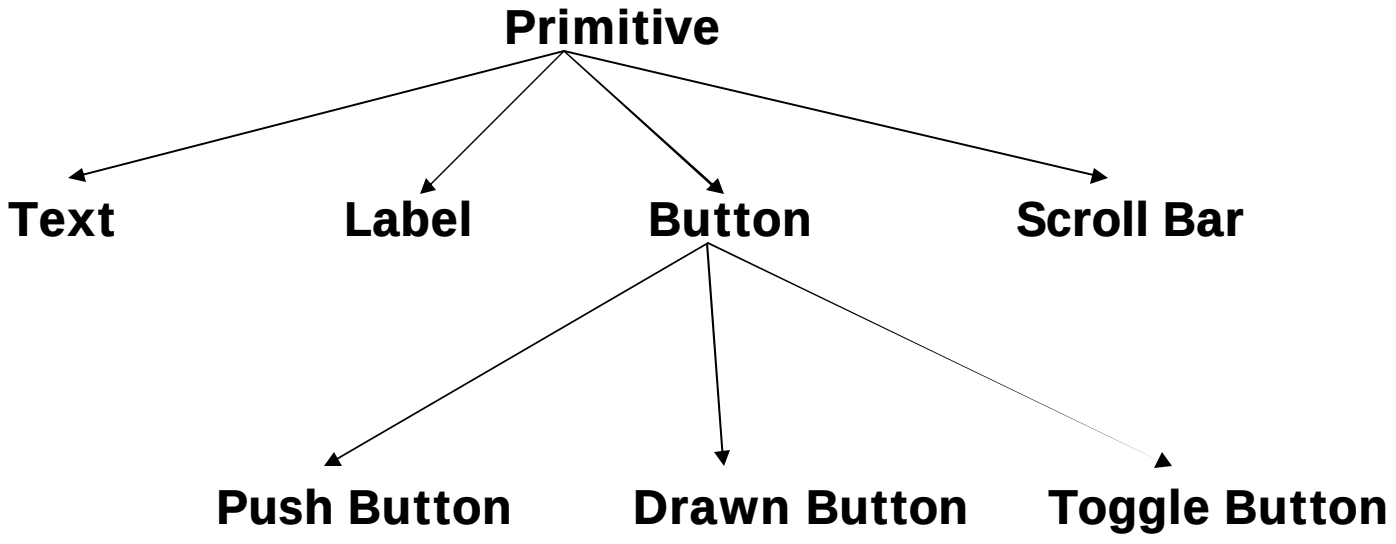
- **Object-oriented hierarchy of UI interactors called widgets**
  - Associate callback routines in your code with them
- **Interface is built up by putting child widgets “onto” parent widgets**

Widget

Graphical user interface interactor object



- **Widgets organized into inheritance hierarchy**



Widget

- **Visual appearance**
- **Set of tailorable attributes**

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

- **Interactive behavior**

Widget and Callback

```
    n = 0;
    xmstr =
   XmStringCreate("Color",XmSTRING_DEFAULT_C
   HARSET);
    XtSetArg(args[n], XmNlabelString,
   xmstr); n++;
    XtSetArg(args[n], XmNbackground, red);
   n++;
colorbut =
   XtCreateManagedWidget("colorbutton",

   xmPushButtonWidgetClass,focusrowcol,
   args, n);
    XtAddCallback(colorbut,
   XmNactivateCallback,
                  colorChangeCB, id);


void
colorChangeCB(Widget w, XtPointer userdata,
   XtPointer evtdata)
{
    // Actions
}
```

Main Program Event Loop

```
voidCheckXEvents()
{
   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

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 Cafe**

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)