Customizing the Interface

- What Can You Customize?
  - Customizing Menus
  - Customizing Buttons
  - Customizing Tools
  - Customizing Pop-up Menus
  - Customizing the Project Window
- Start-up and Shutdown Scripts
- Saving a Custom Interface
What Can be Customized

- Menu Bar
- Button Bar
- Tool Bar
- Pop-up Set
- Project Window
DocGUI Object Model

- Control
  - An interface object

- ControlSet
  - A collection of interface objects
Control Properties

• To access control properties double click on the button bar or activate the project window and select the project: customize menu option.
Control Properties

- Page 91-92
  - Apply
  - Click
  - Cursor
  - Disabled
  - Help
  - Help Topic
  - Icon
  - Invisible
  - Label
  - Shortcut
  - Tag
  - Update
Control Properties

- Properties of a document GUI
  - Access the “Customize Types” dialog box by clicking on the “Edit…” button in the “Customize” dialog window.
  - See pages 92-93
    - Action Script
    - Action Update Script
    - Document Base Name
    - Icon
    - Name
    - New Script
    - New Update Script
    - Open Script
    - Title
Event Programming

• An event is an action creating a change in the application
• Examples Events:
  – Click – clicking the mouse on a control
  – Apply – clicking or dragging a tool in a display area
  – Update – a change in the active window
Object model for Control Bar
Creating a Menu System

- Adding and organizing menu options and items
- Executing a task through a menu item
- Disabling and hiding a menu item
- Assigning access and shortcut keys
Menu System Object Model

• *MenuBar* has a collection of *Menus* which in turn have a collection of *Choices* and *Spaces*. 
Modifying Menus With Avenue

'** Creating a new menu choice called "Theme Display"
'** that runs a script turning on and off themes
'** selected in a list.
newChoice = Choice.Make
newChoice.SetLabel("Theme Display")
newChoice.SetClick("my first script")

'** Adds a new choice to the Window menu in the
'** View MenuBar
theViewGUI = av.FindGUI ("View")
theMenuBar = theViewGUI.GetMenuBar
theWindowMenu = theMenuBar.FindByLabel("Window")
theWindowMenu.Add( newChoice, 0)
Creating Buttons

• Adding and organizing buttons
• Executing a task through a button
• Disabling and hiding a menu item
• Assigning access and shortcut keys
Button Bar
Object Model

- *ButtonBar* has a collection of *Buttons* and *Spaces*. 
Modifying Buttons With Avenue

'** Creates a new button that runs a script
'** turning on and off themes selected in a list.
newButton = Button.Make
newButton.SetClick("my first script")

'** Gets the Bex icon and makes it the button's icon
allIcons = IconMgr.GetIcons
iconList = NameDictionary.Make( 100 )
for each anIcon in allIcons
    iconList.Add( anIcon )
end
bexIcon = iconList.Get("Bex")
newButton.SetIcon(bexIcon)

'** Adds the new button to the View ButtonBar
theViewGUI = av.FindGUI("View")
theButtonBar = theViewGUI.GetButtonBar
theButtonBar.Add( newButton, 0)
Creating Tool Buttons

- Adding and organizing tool buttons
- Executing a task sequence through a tool
- Associating a cursor shape to a tool
- Disabling and hiding a tool
- Adding a help line to tools
Tool Bar
Object Model

- *ToolBar* has a collection of *Tools, Spaces, and ToolMenus*. A *ToolMenu* has a collection of *Tools*.
Modifying Tools With Avenue

''** Creating a new tool that runs a script
''** turning on and off themes selected in a list.
newTool = Tool.Make
newTool.SetApply("my first script")

''** Gets the Bex icon and makes it the Tool's icon
allIcons = IconMgr.GetIcons
iconList = NameDictionary.Make(100)
for each anIcon in allIcons
    iconList.Add(anIcon)
end
bexIcon = iconList.Get("Bex")
newTool.SetIcon(bexIcon)

''** Adds a new tool to the View ToolBar
theViewGUI = av.FindGUI ("View")
theToolBar = theViewGUI.GetToolBar
theToolBar.Add( newTool, 0)
Creating Pop-up Menus

- Adding and organizing pop-up options and items
- Executing a task through a pop-up
- Disabling and hiding a pop-up item
Pop-up Menu Object Model

- `PopupSet` has a collection of `Popups` (a subclass of `Menu`), which has a collection of `Choices` and `Spaces`. 
Modifying Pop-up Menus With Avenue

```avenue
*** Creating a new menu choice called "Theme Display"
*** that runs a script turning on and off themes
*** selected in a list.
newChoice = Choice.Make
newChoice.SetLabel("Theme Display")
newChoice.SetClick("my first script")

*** Adds a new choice to the Pop-up menu in the View
theViewGUI = av.FindGUI("View")
theMenuBar = theViewGUI.GetMenuBar
theWindowMenu = theMenuBar.FindByLabel("Window")
theWindowMenu.Add( newChoice, 0)
```
Customizing the Project Window

• Creating new document types
• Modifying document type attributes
Start-up and Shutdown Scripts

• In the project window, select the *project: properties* menu item.
Saving the Custom Interface

• Simply save the project file, and the modified GUI will be saved.

• If you want to save the GUI as your default, open the customize dialog box, and click the “Make Default” button.
  – NOTE: Please don’t do this on the machines in the lab!!