CE94930 – Introduction to Visual BASIC

- Last week
- Debugging
- List controls
- Common dialogues
- “Roll-your-own” dialogues

Last week...

- “Opaque” vs “transparent” backgrounds
- Consistency of visual information
- “Help” information for controls: F1 vs TOC

Debugging

- Single-step
- Debug window -- immediate statements
- Breakpoints
- Debug.print
- Event synchronization

List controls

- List controls: user convenience and program reliability
- Different types: simple, combo, dropdown
- Scrollbars automatic
- Properties store items
- Methods add/remove items

Arrays

- Example:
```
Dim list_o_things( 10 ) as String
```
- 11 elements: indices 0..10
- User-defined index range:
```
Dim list_o_things( 1 To 10 ) as String
```
- Multi-dimensions
- Varying number of elements:
```
Dim list_o_things() as String
...
ReDim list_o_things( 1 to 10 )
```
- ReDim Preserve

List properties

- List -- array containing items
  My_stuff.List(5)
- Index 0 is first item
- ListCount -- number of items in list
- ListIndex -- index of currently-selected item (-1 if none selected)
- ItemData -- array of auxiliary item information

List methods

- Methods: routines that operate on objects
- My_stuff.AddItem “this is an item”
- Property NewIndex is index where item added (useful for sorted lists)
- My_stuff.RemoveItem idx
- My_stuff.Clear

List events

- Simple lists: click, dblclick
- Others: change (text portion), click

List operations

- Traversing a list
- Searching for an item
- Multiple selection
- Item uniqueness?

Traversing a list

```
if( mylist.ListCount > 0 )then
    for i = 0 to mylist.ListCount-1
        debug.print mylist.List(i)
        if( mylist.Selected(i) )then
            debug.print “was selected”
        endif
    next i
endif
```

Specialized lists

- File-system controls: DriveListBox, DirListBox and FileListBox
- Similar to ordinary lists
- Read-only List property
- Change event

Common dialogues

- Provide common “look & feel”
- File open/save, help, font selection, colour selection
- Single control, lots of properties
- Property values predefined, use “global.bas”

Other standard dialogues

- single-line input: InputBox$
- simple messages: MsgBox

Creating custom dialogues

- Standard dialogues lack flexibility
- Application-specific dialogues
- Dialogue is just another form
- Modal dialogues
- No synchronization problems

Application structure

- Multiple forms
- “Show” method, modal
- “Unload” statement
- Modeless forms acceptable when no dependencies among forms
- Fully-qualified names: form.object.property
- Scope: objects are visible, code isn’t