CSCI-GA.3033.003 Scripting Languages

08/06/2012

Objects in VBA
Properties, Call-backs

Outline

- Classes and properties
- Visual programming and callbacks

Defining Classes

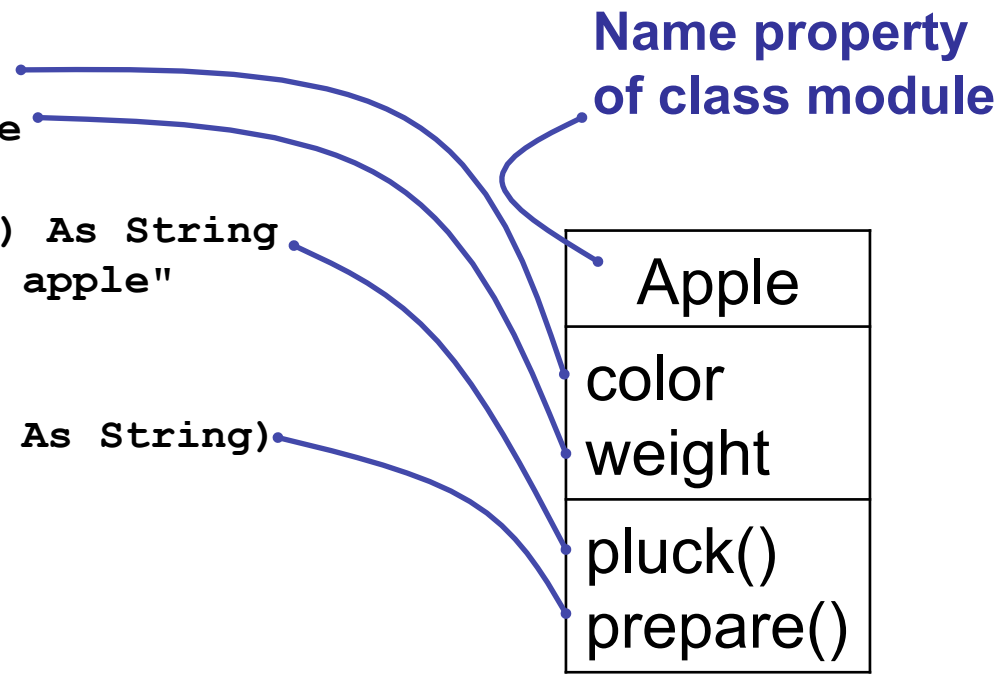
Properties

Methods

```
Public color As String
Public weight As Double

Public Function pluck() As String
  pluck = Me.color & " apple"
End Function

Public Sub prepare(how As String)
  Dim dish As String
  Select Case how
    Case "slice"
      dish = "salad"
    Case "squeeze"
      dish = "juice"
    Case Else
      dish = "dessert"
  End Select
  Debug.Print "yum, " & pluck() & " " & dish
End Sub
```



Reference

Documentation of OO in VBA

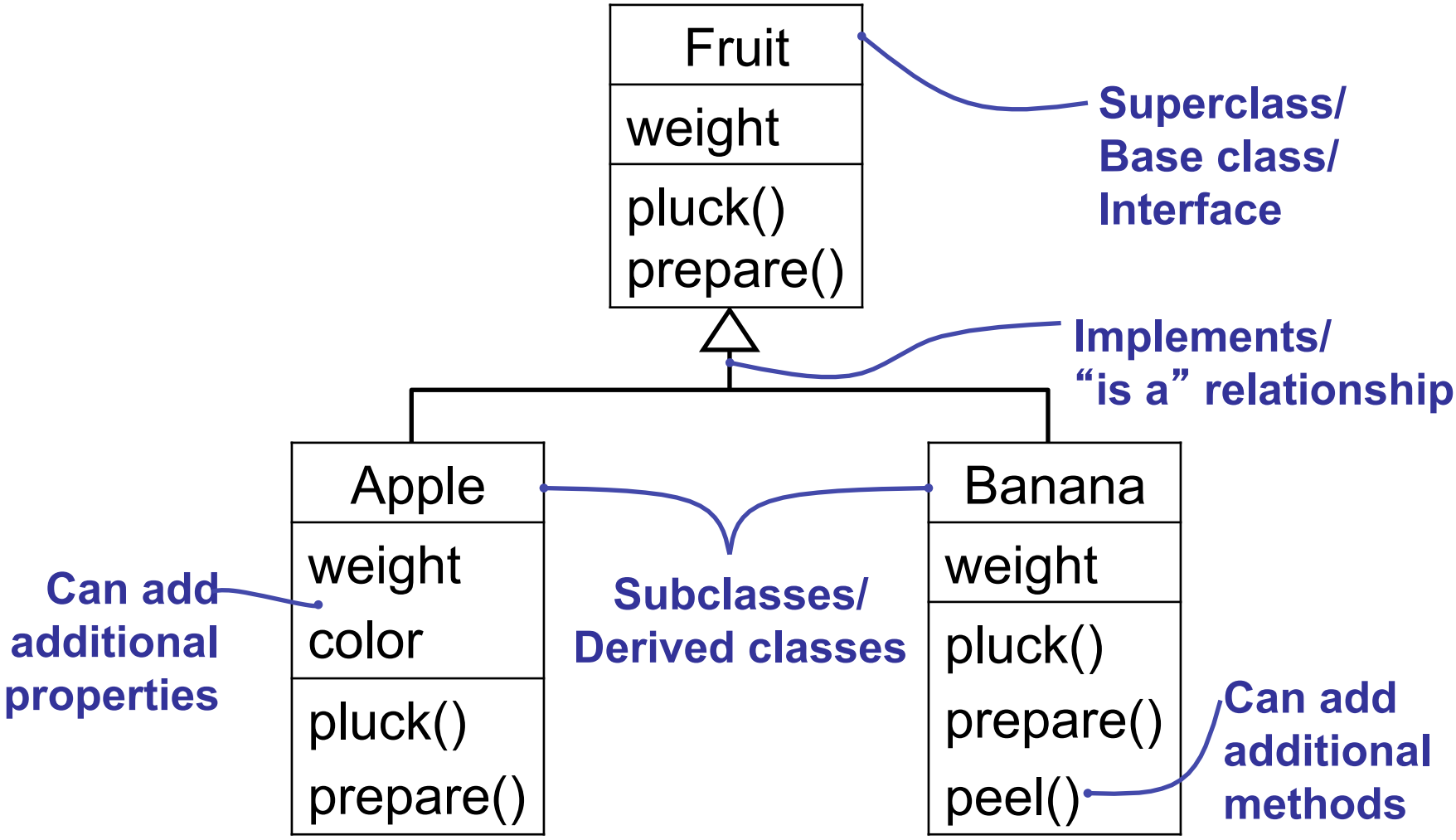
- MSDN library
  - →Development tools and languages
  - →Visual Studio 6.0
  - →Visual Basic 6.0
  - →Product documentation
  - →Using Visual Basic
  - →Programmer's guide
  - →Part 2: What can you do with Visual Basic
  - →Programming with objects

Defining Properties

- Syntax
  - **`Property Get`** *id* **`(`**[*arg****`,`**] *arg***`)`** … **`End Property`**
  - **`Property Let`** *id* **`(`**[*arg****`,`**] *arg***`)`** … **`End Property`**
  - **`Property Set`** *id* **`(`**[*arg****`,`**] *arg***`)`** … **`End Property`**
- Remarks
  - To return value from **`Get`**, assign to *id*
  - **`Let`** assigns regular value, **`Set`** assigns object
  - Extra arguments, if any, are for indexing
  - Each routine can be **`Private`** or **`Public`**
  - **`Public`** variable is just shortcut for routines

Concepts

Inheritance



Subtyping Example

```
Dim someFruit As Fruit
If … Then
  Set someFruit = New Apple
Else
  Set someFruit = New Banana
End If
' compiler knows that the method exists,
' even if it doesn't know which version
' will get called
someFruit.prepare("slice")
```

Inheritance in VBA

Class module "Fruit" (Interface)

Would be abstract in other languages

```
Public weight As Double
Public Function pluck() As String 'empty routine
End Function
Public Sub prepare(how As String) 'empty routine
End Sub
```

"Implements" in subclass identifies superclass

Override private

with mangled name

```
Implements Fruit
Public color As String
Private wght As Double
Private Property Let Fruit_weight(ByVal RHS As Double)
  wght = RHS
End Property
Private Property Get Fruit_weight() As Double
  Fruit_weight = wght
End Property
Private Function Fruit_pluck() As String
  Fruit_pluck = color & " apple"
End Function
Private Sub Fruit_prepare(how As String)
  Debug.Print how & "d " & Fruit_pluck()
End Sub
```

Class module "Apple" (subclass)

Pure Interface Inheritance

- Interface inheritance (in VBA):
  - Subclass must provide own implementation for all methods in superclass
  - Can use object of subclass where object of superclass is expected
- Implementation inheritance (not in VBA):
  - Subclass can leave some methods from superclass unchanged
  - Those method implementations are reused
- To reuse code in VBA, must call it by hand

Soap-box

Missing OO Features in VBA

- Implementation inheritance
- Constructors
  - Can write **`Class_Initialize()`** method
  - Or write subroutine in separate module to allocate **`new`** instance and initialize it
- Static fields and static methods

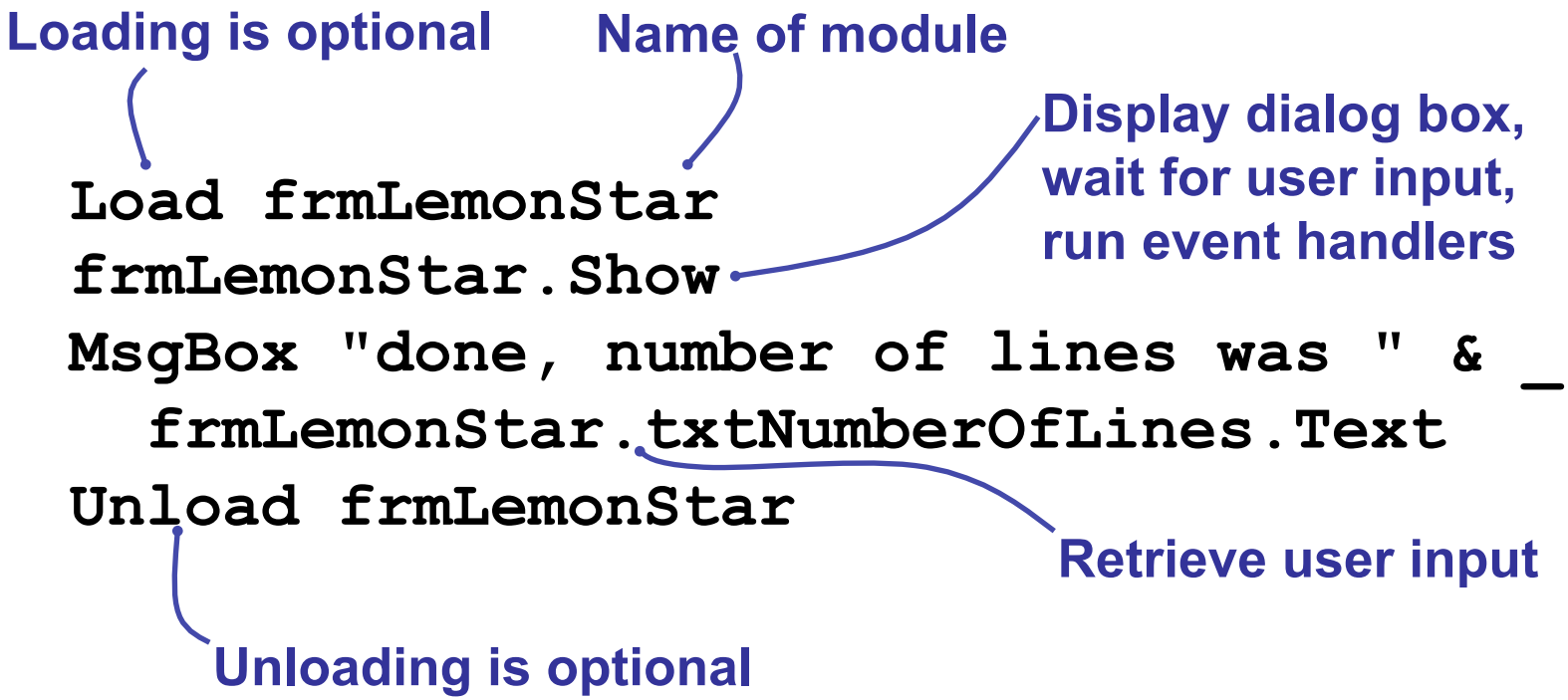
⇒ *OO features changed in VB.NET*

Outline

- Classes and properties
- Visual programming and callbacks

Using Dialogs

- Dialog = window to request user input
- User form = module defining dialog



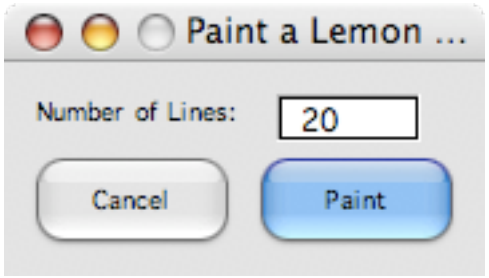
Concepts

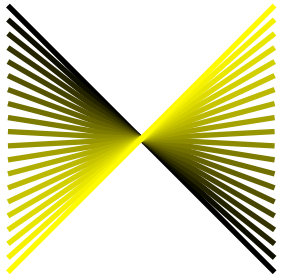
Hungarian Notation

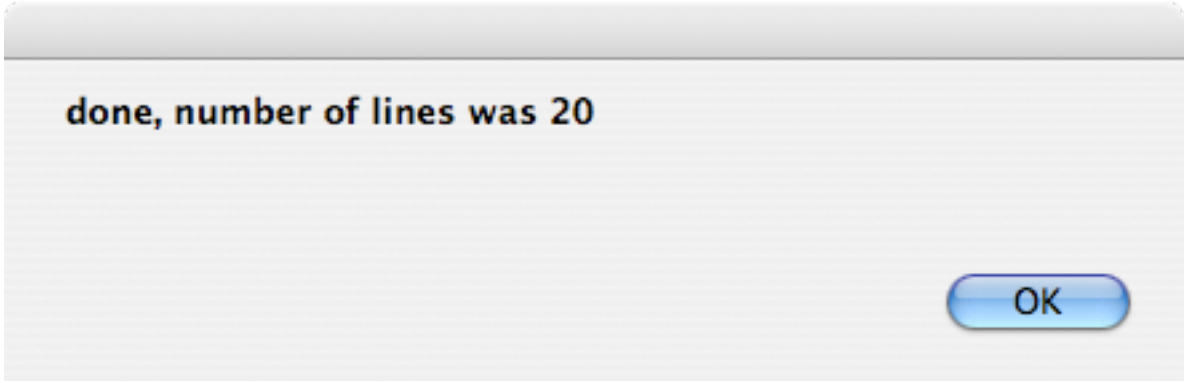
- Convention: start identifier with indication of type

`chk`	Check box
`cmd`	Command button
`frm`	User form
`fra`	Frame
`lst`	List box
`cmb`	Combo box
`mnu`	Menu
`opt`	Option button
`lbl`	Label
`txt`	Text box

Dialog Example







Drag&Drop Dialog Design

- Right-click on project→Insert→User form
- View→Properties Window
  - Change name, caption, maybe width/height
- View→Toolbox
  - Drag and drop controls onto user form
  - Rename, resize, set initial value
  - Set “default” or “cancel” property for buttons
- Right-click on form→Tab order
  - Order when user “tabs through” controls

Writing Event Handlers

```
Sub paintLemonStar(nLines As Integer)
  Dim n As Integer, i As Integer
  n = nLines - 1
  For i = 0 To n
    Dim l As Shape
    Set l = ActiveWindow.Selection.SlideRange.Shapes.AddLine( _
      BeginX:=300, BeginY:=200 + i * 100 / n, _
      EndX  :=400, EndY  :=300 - i * 100 / n)
    l.Line.ForeColor.RGB = RGB(i * 255 / n, i * 255 / n, 0)
  Next i
End Sub

Private Sub cmdPaint_Click()
  paintLemonStar CInt(txtNumberOfLines.Text)
  Hide
End Sub

Private Sub cmdCancel_Click()
  End
End Sub
```

Auxiliary subroutine

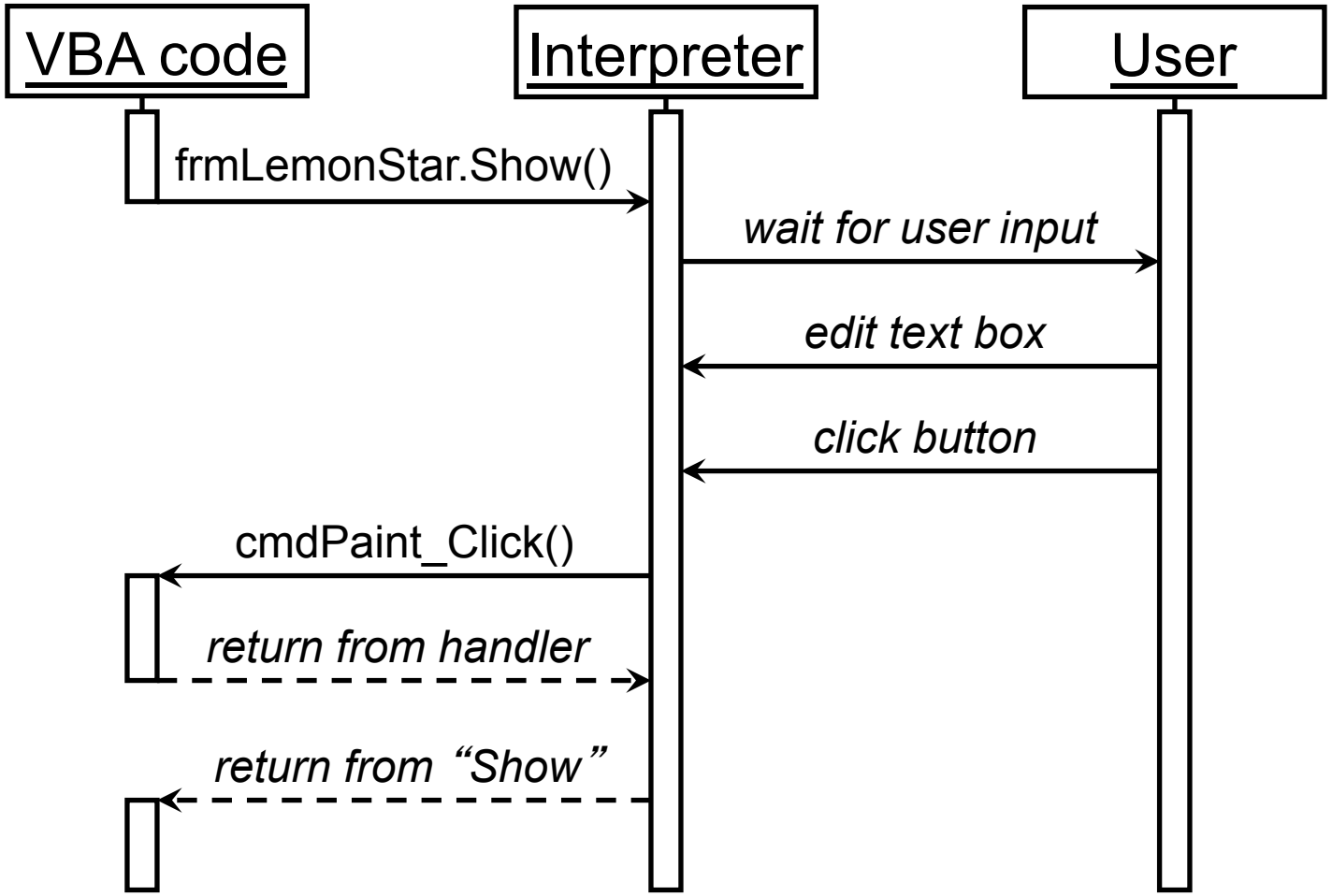
Callbacks with mangled names

Retrieve user input

Code Sheets

- Right-click→View code
  - Just like other modules
  - Contains event handlers as well as regular subroutines
- Double-click on control
  - Goes to handler, creating it if necessary
  - Caution: handler name not changed automatically with control name!
- Dynamic dialog: assign properties at runtime
  - E.g., **`Visible=True`** reveals hidden controls
  - **`UserForm_Initialize`** sets default, e.g. for list box

Call-backs


VBA code
Interpreter
User
frmLemonStar.Show()
wait for user input
edit text box
click button
cmdPaint_Click()
return from handler
return from "Show"

Reference

Common Controls and Events

- Controls:

  Label, TextBox, ComboBox, ListBox, CheckBox, OptionButton, ToggleButton, Frame, CommandButton, TabStrip, MultiPage, ScrollBar, SpinButton, Image

- Events:

  Change, Click, DblClick, Enter, Exit, Initialize, KeyPress, SpinDown, and many more

- In editor for user form code sheet:
  - Left drop-down list: control objects on this form
  - Right drop-down list: events on that object

Callback Mechanisms

VBA form	Subroutine in form with mangled name
VBA class	**`WithEvent`** / **`RaiseEvent`** statements
Java	Pass object on which to call method
Perl, Python, JavaScript	Pass anonymous function (lambda)
C, C++	Pass function pointer
C++	Pass object on which to call “()” operator
SmallTalk	Pass code block
PHP	Pass name of function as string

Events on Classes

- In event source class id_{src}:
  - **`Public Event`** id_{event}**`(`**arg^***`)`**
  - **`RaiseEvent`** id_{event}**`(`**$expr^*$**`)`**
- In event sink module:
  - **`WithEvents`** $id_{handler}$ **`As`** id_{src}
  - **`Sub`** $id_{handler}$**`_`**id_{event}**`(`**arg^***`)`** … **`End Sub`**

Reusing Dialog Boxes

- Don't write "open file" dialog from scratch!
- **`Application.Dialogs(`***id*`).Show`**
  - E.g., *id* = **`xlDialogOptionsGeneral`**
  - To find others: Help→Visual Basic→Search "built-in dialog argument"
- **`Display`** instead of **`Show`** prevents handlers
  - Return value: -2=Close, -1=OK, 0=Cancel, >0 other command buttons
  - Retrieve user input from controls with **`.Value`**

Soap-box

How to Learn a Language

I. Use peers & gurus
II. Install tools
III. Read tutorial
IV. Find language & library reference
V. Read example programs
VI. Write example programs:
I/O, types, control flow, libraries
VII. Understand error messages
VIII. Practice

Common VBA Mistakes

Error	Message	Common cause
Runtime error '91'	Object variable or With block variable not set	Missing "Set" before object assignment
Compile error	Invalid character	Missing space before "_" at end of line
Compile error	Expected: =	Parentheses around arguments, missing "Call"
	And many more …	

Soap-box

Suggestions for VBA Practice

- hw02 gets you points, but you may want to do more at your own leisure
- Powerpoint
  - Center shape on slide
  - Draw object model
  - Plot a polynomial
- Excel
  - Create graph, set fonts and grid preferences
  - Generate email from name+title+address sheet

Administrative

Last Slide

- First homework due today at 6pm
- Second homework on the course website

- Today's lecture
  - Properties
  - Call-backs
- Next lecture
  - Associative arrays
  - Regular expressions
  - Basics of Perl