Objects and Object-oriented Programming (OOP)

Larry Caretto
Mechanical Engineering 209

Computer Programming for Mechanical Engineers

March 28, 2017

Reminder: Midterm Exam April 6

California State University Northridge

Outline

- Basics of object-oriented programming: objects, properties, and methods
- Important objects in Excel
- Collections of Objects
- Help from Object Browser
- Object variables
- Relative and absolute references
- Offset Property

California State University Northridge 2

Background

- We have used statements like the following in previous VBA coding
  - Range("D6").Value Value Property
  - Cells(6,4).Value Value Property
  - Range("R12").NumberFormat Property
  - Range("B6:D12").ClearContents Method
- These are examples of object-oriented programming (OOP)
  - We are interested in using OOP to control Excel worksheets from VBA

California State University Northridge 3

Basic Ideas

- Concept of object and property and object method
  - <Object>.<property> can be read or write
  - <Object>.<method> for actions with object
- Can have properties that are also objects
  - <Object>.<propertyObject>.<object>
  - <Object>.<propertyObject>.<method>
  - E.g. Range("A1").Font.Size

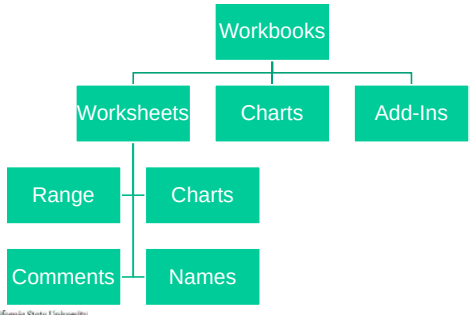
California State University Northridge 4

Object-Oriented Programming

- OOP uses "objects" which have properties and methods
- A range is an object which has properties (e.g. Value and NumberFormat)
  - It also has methods like ClearContents
- In OOP we use constructs like Object.Property and Object.Method
- We have properties that we can read, write, or do both

California State University Northridge 5

Simplified Excel Object Hierarchy



California State University Northridge 6

Object Shorthand

- When we write a statement like Range("A6").Value = 12 we change the value of cell A6 on the current worksheet, known as the ActiveSheet
- When we want to work with multiple sheets (e.g., to copy data from one worksheet to another) we need to explicitly state the worksheet to which we are referring

California State University Northridge 7

Examples

- For multiple worksheets, we can use a fuller reference as follows
  - Worksheets("<sheetName>").Range().Value
- Copy data from one worksheet to another with a command like the following:

  Worksheets("Sheet1").Range("B3:C6").Value =Worksheets("Sheet2").Range("A1:B4").Value
- Can also refer to worksheets by numerical index: Worksheets(k)

California State University Northridge 8

Object Variables

- Can declare variables to refer to objects
  - Used to simplify code with repeated references to one object

```
Dim rng As Range
Set rng = Worksheets("Sheet3").Cells(5,12)
If rng.Value < 0 Then rng.Value = -rng.Value
Set rng = Nothing  'Clear object variable
```

  - Notice use of Set for setting object variables to a value and clearing object variables at end of code

California State University Northridge 9

Worksheet Object Variable

- Useful approach for referring to different worksheets several times in VBA code

```
Dim sht1 As Worksheet
Dim sht2 As Worksheet
Set sht1 = Worksheets("Sheet1")
Set sht2 = Worksheets("Sheet2")
sht2.Range("B3:C6").Value = _
sht1.Range("A1:B4").Value
Set sht1 = Nothing
Set sht2 = Nothing
```

California State University Northridge 10

Example: Find Last Row

- VBA code to find the last row with data in a selected column on a worksheet

```
Function getLastRow(sheetName As String, _
                    column As Long) As Long
    Dim sht As Worksheet
    Set sht = Worksheets(sheetName)
    getLastRow = sht.Cells(sht.Rows.Count, _
                 sht.column).End(xlUp).Row
    Set sht = Nothing
End Function
```

- Example: =getLastRow("Sheet1",5) returns number of last row with data in column E of Sheet1

How does this code work?

California State University Northridge 11

How does this code work?

- Cells(<row>,<column>) refers to a range by row and column number

```
sht.Cells(sht.Rows.Count, sht.column)
```

  - Here we use the object variable, sht, to refer to a particular worksheet
  - The entry "Rows.Count" gives the last row in the worksheet = 1,048,576

```
sht.Cells(sht.Ro…, sht.col…).End(xlUp)
```

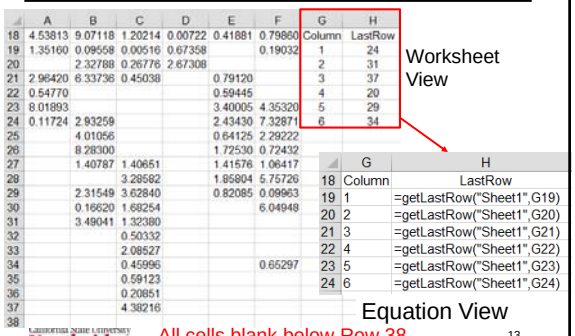
  - The End(xlUp) method finds the first row above the current row (here 1,048,576) that is not blank

```
sht.Cells(sht.R…,sht.c…).End(xlUp).Row
```

  - The final "Row" returns the number of the row

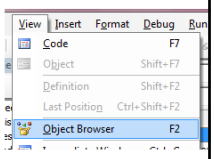
California State University Northridge 12

LastRow Example



Object Browser

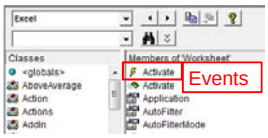
- Shows all objects and methods
- Access from VB Editor
- Press F2 or select Object Browser from View menu



Objects

Methods

Properties

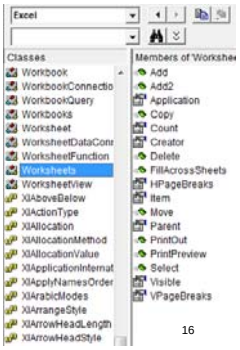


Collections

- Some objects represent collections of individual objects
  - Worksheet is an object that is a single worksheet
  - Worksheets is an object that represents the collection of all worksheets in a workbook
  - What do you think Workbooks is?
  - Workbooks is a collection of all open workbooks

Object Browser Example

- Select Excel from top pulldown menu
- Select desired object from Classes menu
- Right-click desired class member
- Select Help from resulting right- click menu



Worksheets.Count Help

Worksheets.Count Property (Excel)

Office 2013 and later | Other Versions

Contribute to this content

Use GitHub to suggest and submit changes. See our guidelines for contributi

Returns a **Long** value that represents the number of objects in the collection.

Syntax

expression .**Count**

expression A variable that represents a **Worksheets** object.

Use of Macro Recorder

- Macro recorder can show how to write VBA code to perform certain functions
- Code may not be most efficient
  - Generates code that selects a cell and then works with the selection
- Code may also do more than you want
  - Example: changing one aspect of a font will produce code that sets all other font properties to their default values

Code Generated for Font Size

```
With Selection.Font
    .Name = "Arial"
    .Size = 12
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ThemeColor = xlThemeColorLight1
    .TintAndShade = 0
    .ThemeFont = xlThemeFontNone
End With
```

19

With-End With Structure

- Used to simplify setting of several properties of same object
- Used on previous slide for setting properties of selected font
- General form

```
With <object>
   .<property1> = <value1>
   .<propertyN> = <valueN>
End With
```

Multiple statements allowed here

20

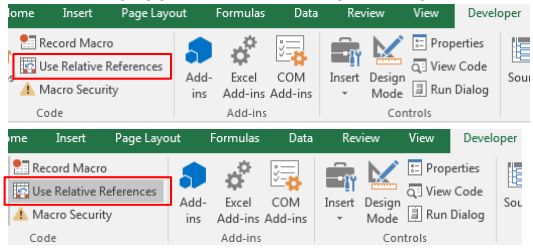
Necessary Font Size Code

- Any one of the following are equivalent statements are required to set font (assumes `Selection` is cells C3:F12)
- `Selection.Font.Size = 12`
- `Range("C3:F12").Font.Size = 12`
- `Range("C3","F12").Font.Size _ = 12`
- `Range(Cells(3,3),Cells(12,6) _ ).Font.Size = 12`

21

Relative/Absolute References

Developer tab with Relative References not selected (top) and selected (bottom)



22

Absolute Reference Macro

```
Sub Macro1()
' Macro1 Macro
    Range("J69").Select
    Selection.Copy
    Range("K74").Select
    ActiveSheet.Paste
End Sub
```

23

Combined Copy/Paste

- Can add paste destination as argument of copy function

```
Sub Macro2()
' Macro2 Macro
    Range("J69").Copy _
                  Range("K74")
End Sub
```

24

Relative Reference Macro

```
Sub Macro3()
' Macro3 Macro
  'Range("J69").Select (equivalent)
    Application.CutCopyMode = False
    Selection.Copy
    ActiveCell.Offset(5, 1) _
                .Range("A1").Select
    ActiveSheet.Paste
End Sub
```

California State University Northridge 25

Offset Property

- Form: <range specification>. _ Offset(<rowOffset>,<columnOffset>)
- **rowOffset** is an integer that is negative, zero or positive for **rows** above, the same as, or lower than the specified range row
- **columnOffset** is an integer that is negative, zero or positive for **columns** to the left of, the same as, or to the right of the specified range column

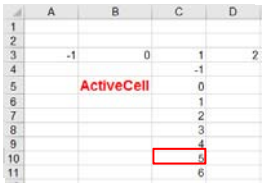
California State University Northridge 26

Offset Command Illustrated

- VBA code with offset property

```
ActiveCell.Offset(5, 1) _
    .Range("A1").Select
```

- What is selected cell?
- Cell C10
- Why is this selected cell?
- It is 5 rows down and 1 column to the right of the Active Cell



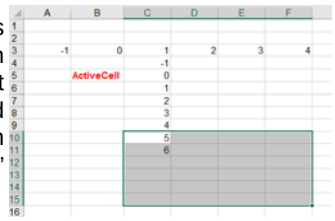
California State University Northridge 27

Offset Command with Select

- Revised VBA code with offset

```
ActiveCell.Offset(5, 1) _
    .Range("A1:D6").Select
```

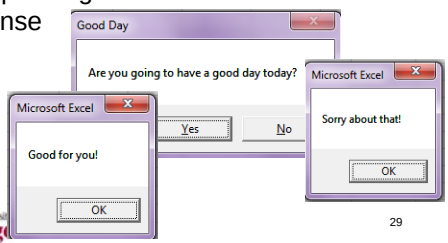
Offset cell 5 rows down and 1 column to left is upper-left cell "A1" in selected 6-row, 4-column range, "C10:F15"



California State University Northridge

Functions as Subs

- Functions may be used as subs if the return value is not needed
- Example msgBox Function with Yes/No response



California State University Northridge 29

Message Box Code

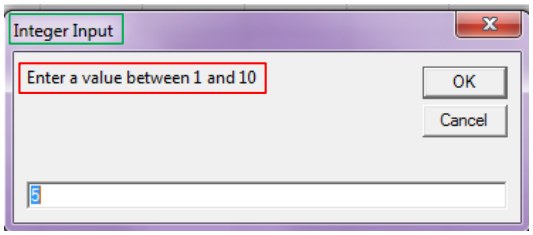
```
response = MsgBox("Are you going to" _
           & "have a good day today?", _
             vbYesNo, "Good Day")
If response = vbYes Then
    MsgBox ("Good for you!")
Else
    MsgBox ("Sorry about that!")
End If
```

MsgBox can return a value

MsgBox does not have to return a value

California State University Northridge 30

Input Box




```
y = InputBox("Enter a value between" _
    & "1 and 10", "Integer Input", 5)
```

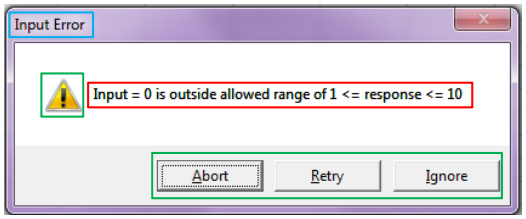
Called "prompt" on next slide

Overall Input Box Code

```
Do
  y = InputBox(prompt,"Integer Input", yMean)
  If y >= yMin And y <= yMax Then Exit Do
  response = MsgBox("Input out of range", _
           vbAbortRetryIgnore, "Input Error")
  If response = vbAbort Then
     MsgBox "Program will halt" : Exit Sub
  ElseIf response = vbIgnore Then
     Exit Do
  End If
Loop
```

Note: infinite loop requires Exit Do statement(s) somewhere

Error Message




```
response = MsgBox("Input = " & y & " is" _
 "outside allowed range of 1 <= response" _
 " <= 10",vbAbortRetryIgnore+vbExclamation, _
 "Input Error")
```


More on Argument Lists

- Some argument lists have optional arguments
  - These optional arguments have default values which need not be entered by user
- Some methods have a very large number of arguments many of which are optional
- It is possible to specify only the arguments you want to a function

Example: Add a worksheet

- Here is the method and its parameters
  - Worksheets.Add(<before>,<after>, <count>,<type>)
  - All these arguments are optional
  - Worksheets.Add(,,2,) to add two worksheets
    - When <before> and <after> are both omitted, the new worksheet is added before the active worksheet
    - Could also use Worksheets.Add(count:=2)

Example: Add a worksheet II

- Worksheets.Add(<before>,<after>, <count>,<type>)
  - Worksheets.Add(,,2,)
  - Worksheets.Add(count:=2)
- This := method for specifying only some optional parameters is a general one
- Worksheets.add returns an optional object variable referring to the new work-sheet, e.g., wks = worksheets.add()