

# Chuck's VBA Reference

## Supports Excel 2007

### Hotkeys

CTRL + F3 – Displays Named Range box.

ALT + F1 – After highlighting data, creates chart based upon the data.

F5 – Run selected macro.

F9 – Calculates all worksheets in all open workbooks.

SHIFT + F9 – Calculates active worksheet.

ALT + F11 – Opens VB Editor.

Press F2 to display formula and highlight included cells.

Press F4 to change cell references from absolute and relative.

Force a recalculation by pressing Ctrl + Alt + F9

### Excel 2007 Resources

<http://msdn.microsoft.com/en-us/office/aa905419.aspx>

<http://www.ozgrid.com/forum/showthread.php?t=57762>

## Table of Contents

Chuck's VBA Reference .....	1
Hotkeys.....	1
Excel 2007 Resources .....	1
Speeding Up Macros .....	5
Stopping Screen Flickering with Macros .....	5
Suspending Alerts .....	5
Dynamic Web Query .....	5
Colon Equal Sign (:=) .....	5
Determine Last Row in a Column with Data .....	6
Autofits Columns.....	6
Set Activesheet .....	6
Clearing Contents.....	6
Inserting a Formula into a Cell.....	6
Save Workbook .....	6
Shell Command .....	6
Parse Comma Separated Values.....	6
Environmental Variables .....	7
Where is Data Stored on Computer.....	7
Add Worksheet to Workbook.....	8
Formatting Numbers in a String .....	8
Display Error Message.....	8
Convert Column Number to Letter .....	8
Delete Worksheet .....	9
Hiding/Unhiding Several Sheets .....	9
Object Browser .....	9
Password Protection .....	9
Printing Worksheet with Header and Footer .....	9
Delete a Workbook .....	9
Formatting Codes for Headers and Footers .....	10
Adding Workbooks.....	10
Opening Workbooks.....	11
Open Text File into New Workbook.....	11
Save Workbook with New Name .....	11
Retrieve Current Directory (Folder) .....	11
Closing and Saving a Workbook.....	11
Copying and Moving a Sheet .....	11
Select a Row/Column in an Active Region .....	12
Range Reference and Offset.....	12
InputBox .....	12
Inserting/Deleting Columns and Rows.....	13

Inserting/Deleting Cells .....	13
Naming a Range .....	13
Hide/Unhide Rows .....	13
Parsing Text in a Single Column (Space delimited).....	13
Replace Function.....	14
Entering Formulas into a Range.....	14
Determine Number of Rows Containing Data in a Column .....	14
Create Chart.....	14
Create Chart Using Named Ranges.....	14
Changing Chart Details .....	14
Looping Through Charts .....	14
Named Data Ranges.....	15
Dynamic Named Range .....	15
Series Formula for Charting.....	16
Inserting a Formula into a Range.....	16
Downloading Data from the Internet (Yahoo Quotes) .....	16
Verify File Exists .....	17
Recalculating Data.....	17
User Defined Formulas.....	17
Display Built In Properties .....	17
Display Names of Built In Properties .....	18
Using Excel Formulas.....	18
Sample Moving Average Calculation .....	18
Improved Moving Average Calculation.....	18
String Comparisons "LIKE" .....	18
Returning Max Value Across Many Worksheets .....	19
Windows API – Video Mode .....	19
Determine if a Workbook is Open .....	19
Determine if a Worksheet Exists .....	20
Determine if Directory/Folder Exists .....	20
VLOOKUP .....	22
Common Dialog.....	22
Make Directories and Copy File.....	22
Copy Excel Charts to Powerpoint.....	23
Create Powerpoint .....	24
Read Object Types on Powerpoint Page .....	26
Copy Excel Data to Powerpoint .....	30
Copy Range to Powerpoint.....	30
Copy More Stuff to Powerpoint.....	31
Even More Excel to Powerpoint .....	33
Color Active Cell .....	35
Time Delay .....	35
Writing Data to Word from Excel .....	35
Find Item in Range and Replace .....	35

Like and Wildcard.....	36
Display Workbook Name .....	36
Moving Copies of Shapes .....	36
Deleting Pictures on a Sheet.....	37
Load Webpage into Spreadsheet.....	38
Retrieve Webpage (Entire or Table) .....	38
Status Bar.....	40
SQL Query – Execute MSSQL Stored Procedure .....	40
SQL Retrieve Data 1.....	42
SQL Retrieve Data 2.....	44

## Speeding Up Macros

```
Application.Calculation = xlCalculationManual  
Application.Calculation = xlCalc
```

Make sure to turn automatic calculating in the event of an error.

## Stopping Screen Flickering with Macros

```
Application.ScreenUpdating = false  
Application.ScreenUpdating = true
```

Also increases the speed of macros.

## Suspending Alerts

```
Application.DisplayAlerts = false  
Application.DisplayAlerts = true
```

## Dynamic Web Query

```
Dim QuerySheet As Worksheet  
Dim DataSheet As Worksheet  
Dim sURL As String  
Set DataSheet = ActiveSheet  
sURL = "  
http://finance.yahoo.com/q/hp?s=MSFT&a=03&b=11&c=2006&d=03&e=11&f=2007&g=d"  
Columns("AA:AG").Select  
Selection.ClearContents  
With ActiveSheet.QueryTables.Add(Connection:="URL;" & sURL,  
Destination:=DataSheet.Range("S1"))  
    .BackgroundQuery = True  
    .TablesOnlyFromHTML = False  
    .Refresh BackgroundQuery:=False  
    .SaveData = True  
End With
```

## Colon Equal Sign (:=)

Given...

```
Public Sub Showname(surname As String, firstname As String)  
    MsgBox "Hello " + firstname + " " + surname  
End Sub
```

Can be called using :=.

```
Public Sub Testshow()  
    Showname firstname:="homer", surname:="simpson"  
End Sub
```

## Determine Last Row in a Column with Data

```
Dim LastRow As Long
LastRow = [A65536].End(xlUp).Row
```

Or

```
Dim lastRow As long
lastRow = Worksheets("Data").Cells(65536, 1).End(xlUp).Row
```

## Autofits Columns

```
Columns("AA:AG").EntireColumn.AutoFit
```

## Set Activesheet

```
Dim DataSheet As Worksheet
Set DataSheet = ActiveSheet
'do something
'then release memory
Set DataSheet = Nothing
```

## Clearing Contents

```
Columns("C:I").Select
Selection.ClearContents

Cells.Select 'selects entire sheet
Selection.ClearContents
```

## Inserting a Formula into a Cell

```
Worksheets("Stock").Cells(i, col).Formula = "=average(g1:g10)"
```

## Save Workbook

```
Application.Dialogs(xlDialogSaveAs).Show Application.Path &
"\mybook.xls"
```

## Shell Command

```
Dim retVal
retVal = Shell(ThisWorkbook.Path & "\lfile.exe " & sInput & " " &
sOutput, 1)

Dim notepadRet
notepadRet = Shell("Notepad.exe " & dlg.filename, vbNormalFocus)
```

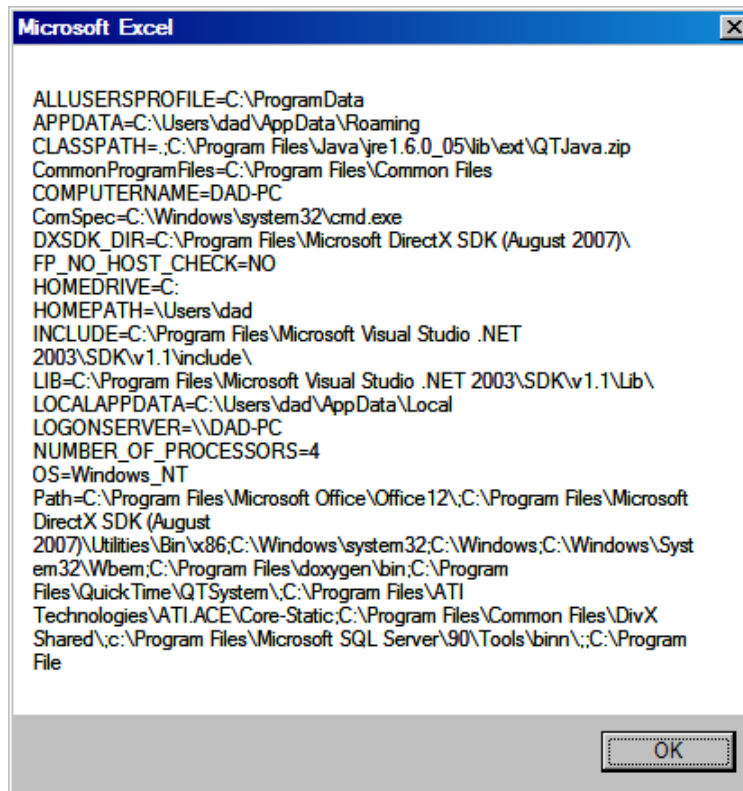
## Parse Comma Separated Values

```
Dim sValues as string
Dim vData as variant
vData = split(sValues, ",")
for i = 0 to ubound(sValues) - 1
'do something with sValues(i)
Next i
```

## Environmental Variables

The following snippet of code reads environmental variables and displays the following message box.

```
Dim EnvString As String
Dim i As Integer
i = 1
Dim txt As String
Do
    EnvString = Environ(i)
    txt = txt & EnvString & vbCrLf
    i = i + 1
Loop Until EnvString = ""
Dim ret
MsgBox txt
```



## Where is Data Stored on Computer

Some systems force data to be saved by default to the user's profile. In other cases it goes to the sub folder related to the application.

```
Dim Fname As Variant
Dim history As String
Dim userprofile As String

userprofile = Environ("USERPROFILE")
history = userprofile & "\\Documents\history\"
```

```
Fname = history
Dim ct As Integer
ct = 0

Dim sFName As String
sFName = dir(Fname & "*.csv")
Do While Len(sFName) > 0
    lstFiles.AddItem sFName 'add to list box
    ct = ct + 1
    sFName = dir
Loop

'nothing found...maybe it's stored in documents\history file
If ct < 1 Then
    Fname = ThisWorkbook.Path & "\history\"
    sFName = dir(Fname & "*.csv")
    Do While Len(sFName) > 0
        lstFiles.AddItem sFName
        ct = ct + 1
        sFName = dir
    Loop
End If
```

## Add Worksheet to Workbook

```
Set ws as Worksheet
Set ws = Nothing
Worksheets.Add after:=Worksheets(Worksheets.count)
Worksheets(Worksheets.count).name = sht
Set ws = Worksheets(Worksheets.count)
ws.Select
ActiveSheet.Cells.Select
Selection.ClearContents
ws.Cells(1, 1) = "Date"
set ws = Nothing
```

## Formatting Numbers in a String

```
Dim num as string
Num = "132.345675"
ws.cells(5, 10) = Format$(Num, "###.###")
```

## Display Error Message

```
MsgBox Err.Number & vbCrLf & Err.Description
```

## Convert Column Number to Letter

```
Public Function ColumnLetter(ColumnNumber As Integer) As String
    If ColumnNumber > 26 Then
        ColumnLetter = Chr(Int((ColumnNumber - 1) / 26) + 64) & _
            Chr(((ColumnNumber - 1) Mod 26) + 65)
    Else
        ColumnLetter = Chr(ColumnNumber + 64)
    End If
End Function
```

## Delete Worksheet

```
Sheets("Sheet1").Delete
```

Or

```
Worksheets(1).Delete
```

Or

```
Sheet1.Delete (code name works even if user changes name or position of  
a sheet).
```

## Hiding/Unhiding Several Sheets

```
Dim i As Integer  
For i = 2 To Sheets.Count  
    Sheets(i).Visible = False  
Next i
```

Repeat code and set visible to true to unhide sheets.

## Object Browser

Press F2 to view classes and members.

## Password Protection

```
Worksheets("Sheet1").Protect Password:="aaa"  
Worksheets("Sheet1").Unprotect Password:="aaa"
```

## Printing Worksheet with Header and Footer

Prints Preview.

```
With Worksheets("Sheet1")  
    .PageSetup.Orientation = xlLandscape  
    .PageSetup.PrintArea = "$B$2:$C$5"           'range  
    .PageSetup.RightHeader = Date  
    .PageSetup.CenterHeader = "My Test Header"  
    .PageSetup.LeftFooter = "Page &P of &N"      'page number  
    .PageSetup.RightFooter = "&KFF00FF Right"    'color  
    .PrintOut Preview:=True  
End With
```

## Delete a Workbook

```
Kill "C:\business\invoices\invoice.xls"
```

To remove a folder use "rmdir" in the following example.

```
Rmdir "C:\business\invoices"
```

## Formatting Codes for Headers and Footers

Format code	Description
&L	Left aligns the characters that follow.
&C	Centers the characters that follow.
&R	Right aligns the characters that follow.
&E	Turns double-underline printing on or off.
&X	Turns superscript printing on or off.
&Y	Turns subscript printing on or off.
&B	Turns bold printing on or off.
&I	Turns italic printing on or off.
&U	Turns underline printing on or off.
&S	Turns strikethrough printing on or off.
&"fontname"	Prints the characters that follow in the specified font. Be sure to include the double quotation marks.
&nn	Prints the characters that follow in the specified font size. Use a two-digit number to specify a size in points.
&color	Prints the characters in the specified color. User supplies a hexadecimal color value.

VBA code	Description
&D	Prints the current date.
&T	Prints the current time.
&F	Prints the name of the document.
&A	Prints the name of the workbook tab.
&P	Prints the page number.
&P+number	Prints the page number plus the specified number.
&P-number	Prints the page number minus the specified number.
&&	Prints a single ampersand.
&N	Prints the total number of pages in the document.
&Z	Prints the file path.
&G	Inserts an image.

Color requires "&K" followed by hexadecimal number. E.g. "&KFF00FF".

## Adding Workbooks

`Workbooks.Add`

Make a new copy.

```
Workbooks.Add Template:="Invoice.xls"
```

To make a copy of a template include the path.

```
Workbooks.Add Template:="C:\Business\Invoice.xls"
```

## Opening Workbooks

```
Workbooks.Open Filename:="C:\Business\Invoice.xls"
```

If workbook is password protected.

```
Workbooks.Open Filename:="C:\Business\Invoice.xls" Password:="aaa"
```

## Open Text File into New Workbook

```
Workbooks.OpenText Filename:=ThisWorkbook.Path & "\data.txt",  
comma:=True
```

Use tab:=true for tab delimited data.

To start at a particular row in the text file add "startrow:=3". The text file beginning at the 3<sup>d</sup> row will be added to the workbook. To save the Workbook with the imported text file run this code.

```
ThisWorkbook.SaveAs Filename:=("filename.xls"), FileType =  
xlWorkbookNormal
```

## Save Workbook with New Name

```
ActiveWorkbook.SaveAs Filename:="Invoice2.xls"
```

## Retrieve Current Directory (Folder)

```
MsgBox CurDir
```

## Closing and Saving a Workbook

```
ActiveWorkbook.Close savechanges:=True
```

## Copying and Moving a Sheet

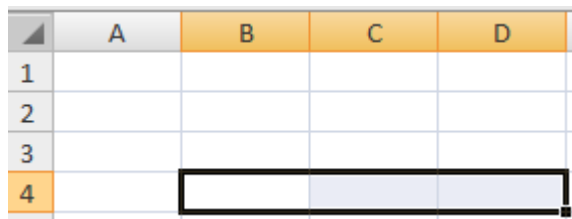
```
Sheets("Sheet1").copy after:=sheets("Sheet3")
```

Use above line to copy a sheet and to place it within the workbook. Use the following line to move a sheet.

```
Sheets("Sheet3").move before:=Sheets(1)
```

## Select a Row/Column in an Active Region

```
Worksheets("Sheet2").Select  
Range("B1:D15").Select  
Selection.Rows(4).Select
```



The image shows a portion of an Excel spreadsheet. The columns are labeled A, B, C, and D. The rows are labeled 1, 2, 3, and 4. The cell B4 is selected, and the entire row 4 is highlighted in light blue. A thick black border surrounds the selected area, extending from column B to column D and from row 4 to row 4.

The code above selects the section above.

```
Worksheets("Sheet2").Select  
Range("B10").Select  
ActiveCell.EntireColumn.Select
```

The code above selects the entire column of column "B".

```
Worksheets("Sheet2").Select  
Range("B10").Select  
ActiveCell.EntireRow.Select
```

The code above selects the entire row of row "10".

```
Worksheets("Sheet2").Select  
Range("B10:D15").Select  
ActiveCell.CurrentRegion.Select  
Selection.Clear
```

The above code uses "B10:D15" as a starting point. It expands the selection to include all data.

## Range Reference and Offset

```
Worksheets("Sheet2").Select  
Dim myrange As Range  
Set myrange = Range("C5")  
myrange.Offset(4, 4).Resize(8, 8).Select
```

The above code moves the top-left corner of the selection 4 rows and 4 columns beyond "C5". It resizes the selection area from one cell to an area of 8 by 8 cells.

## InputBox

```
Dim ret as string  
Ret = InputBox("Enter a range", "Range value", "A1:C3")
```

You must add code to manage a return value of nothing "".

## Inserting/Deleting Columns and Rows

Inserting columns.

```
Worksheets("Sheet1").Columns("M:P").Insert
```

Deleting columns.

```
Worksheets("Sheet1").Columns("M:P").Delete
```

Inserting rows.

```
Worksheets("Sheet1").Rows("5:25").Insert
```

Deleting rows.

```
Worksheets("Sheet1").Rows("5:25").Delete
```

## Inserting/Deleting Cells

Inserting cells.

```
Sheets("Sheet1").Range("M12:M20").Insert shift:=xlShiftDown
```

Deleting cells

```
Sheets("Sheet1").Range("M12:M20").Delete shift:=xlShiftUp
```

## Naming a Range

```
Range("A1:D15").Name = "MyData"
```

## Hide/Unhide Rows

```
Rows("3:10").Hidden = true
```

```
Columns("D:M").Hidden = True
```

Replace true with false to unhide rows and columns.

## Parsing Text in a Single Column (Space delimited)

```
Range("A1:A10").TextToColumns , Destination:=Range("B1"), Space:=True
```

	A	B	C
1	AB	A	B
2	DE	D	E
3	GE	G	E
4	CE	C	E
5	G9	G	9
6	E9	E	9
7	AC	A	C
8	NB	N	B
9	MS	M	S
10	UI	U	I

## Replace Function

```
Dim sWord as string  
sWord = Replace("sweat", "a", "e")
```

This replaces the word "sweat" with "sweet".

## Entering Formulas into a Range

```
Range("A1").Value = "=SUM(A2:A5)"
```

## Determine Number of Rows Containing Data in a Column

Add in cell =COUNTA(\$A:\$A) to return number of rows in column with data.

## Create Chart

```
ActiveSheet.Shapes.AddChart.Select  
ActiveChart.SetSourceData source:=range("Data!$A$4:$E$10")  
ActiveChart.ChartType = xlColumnClustered
```

Press ALT + F1.

## Create Chart Using Named Ranges

```
ActiveSheet.Shapes.AddChart.Select  
ActiveChart.SetSourceData source:=range("MyDates, MyInfo, MyInfo3")  
ActiveChart.ChartType = xlColumnClustered
```

Press ALT + F1.

## Changing Chart Details

```
With ActiveChart  
    .HasTitle = True  
    .ChartTitle.Text = "MSFT"  
    .Axes(xlCategory, xlPrimary).HasTitle = True  
    .Axes(xlCategory, xlPrimary).AxisTitle.Text = "Dates"  
    .Axes(xlCategory, xlPrimary).CategoryType = xlTimeScale  
    .Axes(xlValue, xlPrimary).HasTitle = True  
    .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Values"  
    .HasAxis(xlCategory, xlPrimary) = True  
    .HasAxis(xlValue, xlPrimary) = True  
    .HasDataTable = False  
    .HasLegend = False  
End With
```

## Looping Through Charts

```
Dim cht As ChartObject  
  
For Each cht In ActiveSheet.ChartObjects  
    cht.Select  
    cht.Chart.ChartTitle.Text = "here"  
Next
```

## Named Data Ranges

	A	B	C	D	E
1	Start	5/1/2000			
2	End	5/1/2001			
3					
4	Date	FirmA	FirmB	FirmC	FirmD
5	1/1/2003	9.40	88.26	32.14	85.33
6	1/2/2003	18.40	94.23	21.92	27.54
7	1/3/2003	27.20	114.65	13.54	33.00
8	1/4/2003	39.00	22.68	27.12	67.26
9	1/5/2003	56.2	44.02	38.99	28.14
10	1/6/2003	56.4	21.9	49.54	27.55

The 'Edit Name' dialog box shows the following details:

- Name: AllData
- Scope: Workbook
- Comment: (empty)
- Refers to: =OFFSET(Data!\$A\$4, 1, 1, 6, 4)

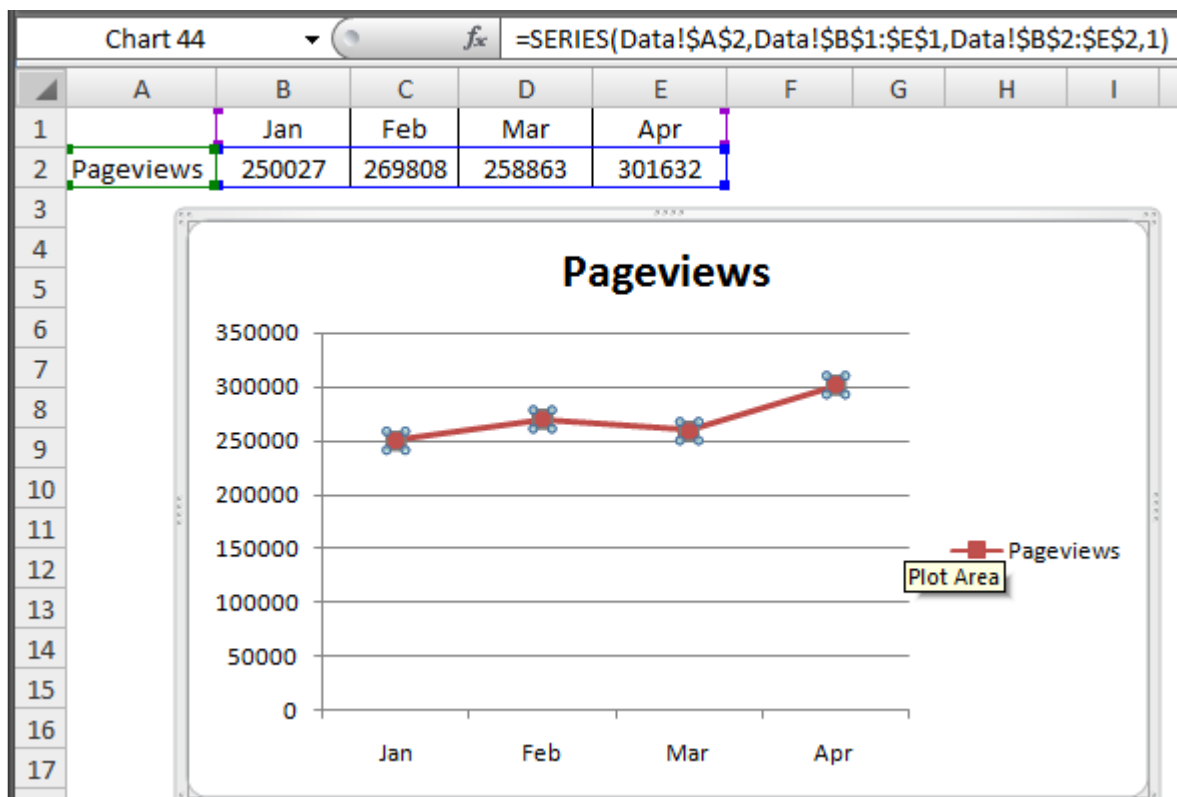
The named range "AllData" may be used as the Data Source for a chart. Press CTRL+F3 to display Edit Name box.

## Dynamic Named Range

```
=OFFSET(Data!$A$1, 0, 1, COUNTA(Data!$A:$A), 1)
```

The above code allows a row of data to be inserted into the named range.

## Series Formula for Charting



With a blank chart highlighted, click on the formula text box and enter the above formula. This is sufficient to adding the Chart data range.

## Inserting a Formula into a Range

```
Range("G2:G25").Value = "=Average(F2:F5)"
```

## Downloading Data from the Internet (Yahoo Quotes)

```
Public Sub test()
    Application.ScreenUpdating = False
    On Error GoTo MyError
    Dim symbol As String
    symbol = "GOOG"

    Dim sURL As String
    Dim data As String

    data = "&f=hpogabj1vb4deyrr5p5p6km3m4ja5b6k4j5e7e8e9t7"
    sURL = "http://finance.yahoo.com/d/quotes.csv?s=" & data

    'connects to internet
    With Sheets("Sheet1").QueryTables.Add(Connection:="URL;" & sURL,
        Destination:=Sheets("Sheet1").Cells(1, 1))
        .BackgroundQuery = True
    End With
End Sub
```

```
.TablesOnlyFromHTML = False  
.Refresh BackgroundQuery:=False  
.SaveData = True  
End With  
  
Application.ScreenUpdating = True  
Exit Sub  
  
MyError:  
Application.ScreenUpdating = True  
End Sub
```

This results in data being downloaded and placed into Cell(1, 1).

## Verify File Exists

```
Msgbox Dir(filepath & filename)
```

Return empty if doesn't exist.

## Recalculating Data

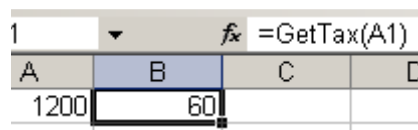
Force a recalculation by pressing Ctrl + Alt + F9

Or

Add Application.Volatile and press F9

## User Defined Formulas

1. Add module.
2. Add function.
3. Use function in cell.



1	A	B	C	D
	1200	60		

## Display Built In Properties

```
MsgBox ThisWorkbook.BuiltinDocumentProperties("Last Save Time")
```

Title	Creation Date	Company
Subject	Last Save Time	Number of Bytes

Author	Total Editing Time	Number of Lines
Keywords	Number of Pages	Number of Paragraphs
Comments	Number of Words	Number of Slides
Template	Number of Characters	Number of Notes
Last Author	Security	Number of Hidden Slides
Revision Number	Category	Number of Multimedia Clips
Application Name	Format	Hyperlink Base
Last Print Date	Manager	Number of Characters (with spaces)

## Display Names of Built In Properties

```
rw = 1
Worksheets(1).Activate
For Each p In ActiveWorkbook.BuiltInDocumentProperties
    Cells(rw, 1).Value = p.Name
    rw = rw + 1
Next
```

## Using Excel Formulas

```
Result = Application.WorksheetFunction.Min(rng)
```

Replace 'Min' with other functions.

## Sample Moving Average Calculation

```
Public Function MA(r As Range, s As Integer)
    If s = 0 Then MA = 0
    MA = Application.WorksheetFunction.Average(r) / s
End Function
```

## Improved Moving Average Calculation

```
Public Function MA(r As Range)
    MA = Application.WorksheetFunction.Average(r) / r.Rows.Count
End Function
```

## String Comparisons "LIKE"

```
Function ISLIKE(text As String, pattern As String) As Boolean
    ISLIKE = text Like pattern
End Function
```

```
Public Sub test()  
  MsgBox ISLIKE("ahuck", "[abc]*")  
End Sub
```

<b>Characters in <i>pattern</i></b>	<b>Matches in <i>string</i></b>
?	Any single character.
*	Zero or more characters.
#	Any single digit (0–9).
[ <i>charlist</i> ]	Any single character in <i>charlist</i> .
[! <i>charlist</i> ]	Any single character not in <i>charlist</i> .

Case Insentive – Use UCASE

```
Public Sub test()  
  MsgBox ISLIKE(UCASE("mhuck"), UCASE("[abc]*"))  
End Sub
```

## Returning Max Value Across Many Worksheets

```
=MAX(Shee1:Shee4!B1)
```

## Windows API – Video Mode

Declare Function GetSystemMetrics Lib "user32" (ByVal nIndex As Long) As Long

```
Sub Video()  
  w = GetSystemMetrics(SM_CXSCREEN)  
  h = GetSystemMetrics(SM_CYSCREEN)  
  MsgBox w & ":" & h  
End Sub
```

## Determine if a Workbook is Open

```
On Error Resume Next  
Dim x As Workbook  
Set x = Workbooks("Book2.xls")  
If Err = 0 Then
```

```
MsgBox "Open"  
Else  
MsgBox "Closed"  
End If
```

## Determine if a Worksheet Exists

```
Dim x As Object  
On Error Resume Next  
Set x = ActiveWorkbook.Sheets("Sheet4")  
If Err = 0 Then  
MsgBox "Exists!"  
Else  
MsgBox "Does not exist!"  
End If
```

## Determine if Directory/Folder Exists

```
If Dir("c:\downloads2", vbDirectory) = "" Then  
MsgBox "False"  
Else  
MsgBox "True"  
End If
```

## Iterate Through All Named Ranges

```
Dim n As Name  
For Each n In ActiveWorkbook.Names  
MsgBox n.Name  
Next n
```

## Load Variable Values into a Range

```
Dim var(5) As Single  
var(0) = 12.45  
var(1) = 34.56  
var(2) = 65.67  
var(3) = 87.45  
var(4) = 90.01
```

```
Range("A1:A5").Value = Application.WorksheetFunction.Transpose(var)
```

## Load Range into Variant Array

```
Dim var As Variant
var = Range("A1:L600").Value
MsgBox UBound(var, 1) & ":" & UBound(var, 2)
```

### **Read Range, Load Array, Calculate, Return Values**

```
Dim var As Variant
var = Range("A1:L600").Value
```

```
Dim i As Integer
Dim j As Integer
For i = 1 To UBound(var, 1)
  For j = 1 To UBound(var, 2)
    var(i, j) = i * j * 2.5
  Next j
Next i
```

```
Range("A1:L600") = var
```

## VLOOKUP

**Column #2 (B)**

**Press F4 to toggle between absolute and relative addresses.**

## Common Dialog

```

dlg.InitDir = ThisWorkbook.Path
dlg.filename = ""
dlg.Filter = "Excel 1997-2003 (*.xls)|*.xls|(*.xlsm)|(*.xlsm)"
dlg.ShowOpen

```

```

Dim filename As String
filename = dlg.filename

```

```

'file exists
If Dir(filename) <> "" Then
    'do something
End If

```

## Make Directories and Copy File

```

MkDir ThisWorkbook.Path & "\members\" & Trim(CStr(gNextMemberID))
MkDir ThisWorkbook.Path & "\members\" & Trim(CStr(gNextMemberID)) & "\" &
CStr(nYear)
Dim source As String
Dim dest As String
source = ThisWorkbook.Path & "\data\Member_Template.xls"
dest = ThisWorkbook.Path & "\members\" & Trim(CStr(gNextMemberID)) & "\" &
CStr(nYear) & "\" & Trim(CStr(gNextMemberID)) & ".xls"
FileCopy source, dest

```

## Copy Excel Charts to Powerpoint

```
Sub CreateNewPowerPointPresentation()  
  ' to test this code, paste it into an Excel module  
  ' add a reference to the PowerPoint-library this is done from the  
  ' Tools ---> References menu path and you  
  ' need to find the microsoft powerpoint check box and check it.  
  ' The excel can use ppt objects within itself  
  ' create a new folder named C:\Foldername or edit the filenames in the code  
  Dim pptApp As PowerPoint.Application  
  Dim pptPres As PowerPoint.Presentation  
  Dim pptSlide As PowerPoint.Slide  
  Dim pptShape As PowerPoint.Shape  
  Dim i As Integer, strString As String  
  Dim Graphcount As Integer  
  Dim count As Integer  
  
  count = 0 'initialise count variable  
  i = 1  
  'returns the number of charts on the sheet at the time the macro is run.  
  'User customises the worksheets name to sheet that holds all the charts  
  Graphcount = Worksheets(1).ChartObjects.count  
  
  Set pptApp = CreateObject("PowerPoint.Application")  
  Set pptPres = pptApp.Presentations.Add(msoTrue) ' create a new presentation  
  ' or open an existing presentation  
  ' Set pptPres = pptApp.Presentations.Open("C:\Foldername\Filename.ppt")  
  
  Do While i < Graphcount + 1 ' starts a loop to copy charts  
  
    ActiveSheet.ChartObjects(i).Activate ' selects the chart object by its index number  
    ActiveChart.ChartArea.Select  
    ActiveChart.ChartArea.Copy  
  
    With pptPres.Slides  
      Set pptSlide = .Add(.count + 1, ppLayoutTitleOnly) ' add a slide  
    End With  
  
    With pptSlide  
      .Shapes(1).TextFrame.TextRange.Text = "Slide Title"  
      ' take this line out if you dont want a generic slide title to appear on each slide  
      .Shapes.PasteSpecial ppPasteDefault  
      With .Shapes(.Shapes.count) ' sizes the graph on the slide
```

```
.Left = 120  
.Top = 125.125  
.Width = 480  
.Height = 289.625  
End With  
End With
```

```
Application.CutCopyMode = False ' end cut/copy from Excel  
Set pptSlide = Nothing  
i = i + 1 ' increment the graph count to copy the next chart on the excel sheet  
Loop
```

```
On Error Resume Next ' ignore errors
```

```
On Error GoTo 0 ' resume normal error handling  
Set pptPres = Nothing
```

```
pptApp.Visible = True ' display the application  
'pptApp.Quit ' or close the PowerPoint application  
Set pptApp = Nothing
```

```
End Sub
```

## Create Powerpoint

```
'*****  
' Creates powerpoint presentation in Excel  
' Base code: http://vbadud.blogspot.com/2006/07/vba-creating-powerpoint-presentation.html  
' Add refence to PPT by selecting Tools, References...  
' Select Microsoft Powerpoint x.yy Object Library  
'*****
```

```
Option Explicit
```

```
Sub Create_PowerPoint_Slides()
```

```
On Error GoTo Err_PPT
```

```
Dim oPA As PowerPoint.Application  
Dim oPP As PowerPoint.Presentation  
Dim oPS As PowerPoint.Slide  
Dim oShape As PowerPoint.Shape  
Dim sPath As String  
Dim sFile As String
```

Dim i As Integer

```
'create filename (unique)
sPath = ThisWorkbook.Path
sFile = "MyfileName" & CStr(Minute(Now)) & "_" & CStr(Second(Now))
```

```
'create application, presentation
Set oPA = New PowerPoint.Application
oPA.Visible = msoTrue
Set oPP = oPA.Presentations.Add(msoTrue)
```

```
'add slides
For i = 1 To 10
    oPP.Slides.Add 1, ppLayoutBlank
Next i
```

```
'set slide to first slide
Set oPS = oPP.Slides(1)
```

```
'add shape
Set oShape = oPS.Shapes.AddTextbox(msoTextOrientationHorizontal, 140#, 246#, 400#, 36#)
oShape.TextFrame.WordWrap = msoTrue
oShape.TextFrame.TextRange.Text = "Comments For File : " & sFile
With oShape
    .Fill.Visible = msoTrue
    .Fill.Solid
    .Fill.ForeColor.RGB = RGB(204, 255, 255)
    .Line.Weight = 3#
    .Line.Visible = msoTrue
    .Line.ForeColor.SchemeColor = ppForeground
    .Line.BackColor.RGB = RGB(255, 255, 255)
End With
```

```
'save file and close
oPP.SaveAs sPath & "\" & sFile & ".ppt"
oPP.Close
oPA.Quit
```

```
'clear memory
If Not oPS Is Nothing Then Set oPS = Nothing
If Not oPP Is Nothing Then Set oPP = Nothing
If Not oPA Is Nothing Then Set oPA = Nothing
```

```
'ERROR in code jump here
```

```
Err_PPT:
  If Err <> 0 Then
    MsgBox Err.Description
    Err.Clear
    Resume Next
  End If
End Sub
```

## Read Object Types on Powerpoint Page

```
'http://www.pptfaq.com/FAQ00008.htm
Option Explicit
```

```
Sub Object_Types_on_This_Slide()
  'Refers to each object on the current page and returns the Shapes.Type
  'Can be very useful when searching through all objects on a page
  Dim it As String
  Dim i As Integer
  Dim Ctr As Integer
  'Read-only Long
  'No need to select the object in order to use it
  With ActiveWindow.Selection.SlideRange.Shapes.Count
  'But it is easier to watch when the object is selected
  'This next line is for demonstration purposes only.
  'It is not necessary
  ActiveWindow.Selection.SlideRange.Shapes(i).Select

  Select Case .Type

    'Type 1
    Case msoAutoShape
      it = "an AutoShape. Type : " & .Type

    'Type 2
    Case msoCallout
      it = "a Callout. Type : " & .Type

    'Type 3
```

```
Case msoChart
    it = "a Chart. Type : " & .Type

'Type 4
Case msoComment
    it = "a Comment. Type : " & .Type

'Type 5
Case msoFreeform
    it = "a Freeform. Type : " & .Type

'Type 6
Case msoGroup
    it = "a Group. Type : " & .Type

' If it's a group them iterate thru
' the items and list them

    it = it & vbCrLf & "Comprised of..."
    For Ctr = 1 To .GroupItems.Count
        it = it & vbCrLf & _
            .GroupItems(Ctr).Name & _
            ". Type:" & .GroupItems(Ctr).Type
    Next Ctr

'Type 7
Case msoEmbeddedOLEObject
    it = "an Embedded OLE Object. Type : " & .Type

'Type 8
Case msoFormControl
    it = "a Form Control. Type : " & .Type

'Type 9
Case msoLine
    it = "a Line. Type : " & .Type

'Type 10
Case msoLinkedOLEObject
    it = "a Linked OLE Object. Type : " & .Type
    With .LinkFormat
        it = it & vbCrLf & "My Source: " & _
            .SourceFullName
    End With
```

'Type 11

Case msoLinkedPicture

it = "a Linked Picture. Type : " & .Type

With .LinkFormat

it = it & vbCrLf & "My Source: " & \_  
.SourceFullName

End With

'Type 12

Case msoOLEControlObject

it = "an OLE Control Object. Type : " & .Type

'Type 13

Case msoPicture

it = "a embedded picture. Type : " & .Type

'Type 14

Case msoPlaceholder

it = "a text placeholder (title or regular text--" & \_  
"not a standard textbox) object." & \_  
"Type : " & .Type

'Type 15

Case msoTextEffect

it = "a WordArt (Text Effect). Type : " & .Type

'Type 16

Case msoMedia

it = "a Media object .. sound, etc. Type : " & .Type

With .LinkFormat

it = it & vbCrLf & " My Source: " & \_  
.SourceFullName

End With

'Type 17

Case msoTextBox

it = "a Text Box."

'Type 18 = msoScriptAnchor, not defined in PPT pre-2000 so we use the numeric value

'Case msoScriptAnchor

Case 18

it = " a ScriptAnchor. Type : " & .Type

```
'Type 19 = msoTable, not defined in PPT pre-2000 so we use the numeric value
'Case msoTable
Case 19
    it = " a Table. Type : " & .Type

'Type 19 = msoCanvas, not defined in PPT pre-2000 so we use the numeric value
'Case msoCanvas
Case 20
    it = " a Canvas. Type : " & .Type

'Type 21 = msoDiagram, not defined in PPT pre-2000 so we use the numeric value
'Case msoDiagram
Case 22
    it = " a Diagram. Type : " & .Type

'Type 22 = msolnk, not defined in PPT pre-2000 so we use the numeric value
'Case msolnk
Case 22
    it = " an Ink shape. Type : " & .Type

'Type 23 = msolnkComment, not defined in PPT pre-2000 so we use the numeric value
'Case msolnkComment
Case 23
    it = " an InkComment. Type : " & .Type

'Type -2
Case msoShapeTypeMixed
    it = "a Mixed object (whatever that might be)." & _
        "Type : " & .Type

'Just in case
Case Else
    it = "a mystery!? An undocumented object type?" & _
        " Haven't found one of these yet!"
End Select

MsgBox ("I'm " & it)
End With
Next i
End Sub
```

## Copy Excel Data to Powerpoint

```
Sub UsedRangePPT()  
    Dim objPPT As Object  
    Dim shtTemp As Object  
    Dim intSlide As Integer  
  
    Set objPPT = CreateObject("Powerpoint.application")  
    objPPT.Visible = True  
    objPPT.Presentations.Add  
    objPPT.ActiveWindow.ViewType = 1 'ppViewSlide  
  
    For Each shtTemp In ThisWorkbook.Sheets  
        shtTemp.Range("A1", shtTemp.UsedRange).CopyPicture xlScreen, xlPicture  
        intSlide = intSlide + 1  
        ' new slide for each chart  
        objPPT.ActiveWindow.View.GotoSlide  
Index:=objPPT.ActivePresentation.Slides.Add(Index:=objPPT.ActivePresentation.Slides.Count +  
1, Layout:=12).SlideIndex  
        objPPT.ActiveWindow.View.Paste  
        With  
objPPT.ActiveWindow.View.Slide.Shapes(objPPT.ActiveWindow.View.Slide.Shapes.Count)  
            .Left = (.Parent.Parent.SlideMaster.Width - .Width) / 2  
        End With  
    Next  
  
    Set objPPT = Nothing  
End Sub
```

## Copy Range to Powerpoint

```
Sub UsedRangePPT()  
    Dim objPPT As Object  
    Dim shtTemp As Object  
    Dim intSlide As Integer  
  
    Set objPPT = CreateObject("Powerpoint.application")  
    objPPT.Visible = True  
    objPPT.Presentations.Add  
    objPPT.ActiveWindow.ViewType = 1 'ppViewSlide  
  
    For Each shtTemp In ThisWorkbook.Sheets
```

```
shtTemp.Range("A1:C5").CopyPicture xlScreen, xlPicture
intSlide = intSlide + 1
' new slide for each chart
objPPT.ActiveWindow.View.GotoSlide
index:=objPPT.ActivePresentation.Slides.Add(index:=objPPT.ActivePresentation.Slides.count +
1, Layout:=12).SlideIndex
objPPT.ActiveWindow.View.Paste
With
objPPT.ActiveWindow.View.Slide.Shapes(objPPT.ActiveWindow.View.Slide.Shapes.count)
.Left = (.Parent.Parent.SlideMaster.Width - .Width) / 2
End With
Next

Set objPPT = Nothing
End Sub
```

## Copy More Stuff to Powerpoint

```
Sub Chart2PPT()
Dim objPPT As Object
Dim objPrs As Object
Dim objSld As Object
Dim shtTemp As Object
Dim chtTemp As ChartObject
Dim objShape As Shape
Dim objGShape As Shape
Dim intSlide As Integer
Dim blnCopy As Boolean

Set objPPT = CreateObject("Powerpoint.application")
objPPT.Visible = True
objPPT.Presentations.Add
objPPT.ActiveWindow.ViewType = 1 'ppViewSlide

For Each shtTemp In ThisWorkbook.Sheets
    blnCopy = False
    If shtTemp.Type = xlWorksheet Then
        For Each objShape In shtTemp.Shapes 'chtTemp In shtTemp.ChartObjects
            blnCopy = False
            If objShape.Type = msoGroup Then
                ' if ANY item in group is a chart
                For Each objGShape In objShape.GroupItems
                    If objGShape.Type = msoChart Then
```

```
        blnCopy = True
        Exit For
    End If
Next
End If
If objShape.Type = msoChart Then blnCopy = True

If blnCopy Then
    intSlide = intSlide + 1
    objShape.CopyPicture
    ' new slide for each chart
    objPPT.ActiveWindow.View.GotoSlide
Index:=objPPT.ActivePresentation.Slides.Add(Index:=objPPT.ActivePresentation.Slides.Count +
1, Layout:=12).SlideIndex
    objPPT.ActiveWindow.View.Paste
End If
Next
If Not blnCopy Then
    ' copy used range
    intSlide = intSlide + 1
    shtTemp.UsedRange.CopyPicture
    ' new slide for each chart
    objPPT.ActiveWindow.View.GotoSlide
Index:=objPPT.ActivePresentation.Slides.Add(Index:=objPPT.ActivePresentation.Slides.Count +
1, Layout:=12).SlideIndex
    objPPT.ActiveWindow.View.Paste
End If
Else
    intSlide = intSlide + 1
    shtTemp.CopyPicture
    ' new slide for each chart
    objPPT.ActiveWindow.View.GotoSlide
Index:=objPPT.ActivePresentation.Slides.Add(Index:=objPPT.ActivePresentation.Slides.Count +
1, Layout:=12).SlideIndex
    objPPT.ActiveWindow.View.Paste
End If
Next

Set objPrs = Nothing
Set objPPT = Nothing
End Sub
```

## Even More Excel to Powerpoint

```
Public Sub labelMSGraphs()
```

```
' go through slides and get obj type 7 and place the name above them
```

```
Dim i, j, iCount, iRow, iCol As Integer
```

```
Dim iSlideNum As Integer
```

```
Dim sText As String
```

```
Dim sFileName As String
```

```
Dim bNewSlide As Boolean
```

```
Dim oPPTApp As PowerPoint.Application
```

```
Dim oPPTShape As PowerPoint.Shape
```

```
Dim oPPTFile As PowerPoint.Presentation
```

```
Dim oGraph As Graph.Chart
```

```
Dim strPresPath As String, strExcelFilePath As String, strNewPresPath As String
```

```
sFileName = Application.GetOpenFilename
```

```
'if user cancels
```

```
If sFileName = "False" Then Exit Sub
```

```
Set oPPTApp = CreateObject("PowerPoint.Application")
```

```
oPPTApp.Visible = msoTrue
```

```
'open file, will be read only... PP is single session
```

```
Set oPPTFile = oPPTApp.Presentations.Open(sFileName)
```

```
'make sure all shapes are ungrouped, so we can access each
```

```
'Call unGroupShapes(oPPTFile)
```

```
'loop through slides
```

```
For Each opptslide In oPPTFile.Slides
```

```
    iCount = iCount + 1
```

```
    iRow = iRow + 1
```

```
    iSlideNum = opptslide.SlideIndex
```

```
'loop through shapes in slide
```

```
For Each oPPTShape In opptslide.Shapes
```

```
    iRow = iRow + 1
```

```
        '.Cells(iRow, 2).Value = oPPTShape.Name
```

```
        '.Cells(iRow, 3).Value = oPPTShape.Type

        'check if is graph object
        If oPPTShape.Type = msoEmbeddedOLEObject Then
            'ok, it's OLE, now check if chart
            If Left(UCCase(oPPTShape.OLEFormat.progID), 7) = "MSGGRAPH" Then
                'LABEL IT
                'Set oPPTShape = opptSlide
                Debug.Print "Shape name: " & oPPTShape.Name & " -- Top: " & oPPTShape.Top
                'opptfile.Slides(1).
                oPPTFile.Slides(iSlideNum).Shapes.AddTextbox(msoTextOrientationHorizontal, _
                    Left:=oPPTShape.Left, Top:=oPPTShape.Top - 15, Width:=200,
                    Height:=50).TextFrame.TextRange.Text _
                    = oPPTShape.Name

                End If
            End If

        Next
    Next

    MsgBox "DONE!", vbOKOnly + vbInformation

almostExit:
    'oPPTFile.Close

exitMe:
    Set oGraph = Nothing
    Set oPPTShape = Nothing
    Set oPPTFile = Nothing
    Set oPPTApp = Nothing
End Sub

Public Sub unGroupShapes(ByVal oPres As PowerPoint.Presentation)
    'this will ungroup any grouped shapes
    Dim iCount As Integer

    Debug.Print "Beginning UngroupShapes routine. Presentation: " & oPres.Name

    iCount = 0
    For Each oslide In oPres.Slides
```

```
iCount = iCount + 1
For Each oShape In oslide.Shapes
    If oShape.Type = msoGroup Then
        oShape.Ungroup
        'Debug.Print "Ungrouping shape: " & oShape.Name
    End If

Next
Next

Debug.Print "slides: " & Str(iCount)
```

End Sub

## Color Active Cell

```
Private Sub Workbook_SheetSelectionChange(ByVal Sh As Object, ByVal Target As Range)
    Static OldRange As Range
    On Error Resume Next
    Target.Interior.ColorIndex = 6 ' yellow - change as needed
    OldRange.Interior.ColorIndex = xlColorIndexNone
    Set OldRange = Target
End Sub
```

## Time Delay

Application.Wait (Now + TimeValue("0:00:1")) delays the code for a second.

## Writing Data to Word from Excel

```
Public Sub CreateWordDoc(sOut As String)
    ' requires reference to the Word object library
    Dim wdApp As Word.Application
    Dim wdDoc As Word.Document
    Set wdApp = CreateObject("Word.Application")
    wdApp.Visible = True
    Set wdDoc = wdApp.Documents.Add
    wdDoc.Content.InsertAfter sOut
    Set wdDoc = Nothing
    Set wdApp = Nothing
End Sub
```

## Find Item in Range and Replace

```
With Worksheets(1).Range("a1:a500")
    Set c = .Find(2, lookin:=xlValues)
```

```
If Not c Is Nothing Then
    firstAddress = c.Address
    Do
        c.Value = 5
        Set c = .FindNext(c)
    Loop While Not c Is Nothing And c.Address <> firstAddress
End If
End With
```

## Like and Wildcard

```
For Each c In [A1:C5]
    If c.Font.Name Like "Cour*" Then
        c.Font.Name = "Times New Roman"
    End If
Next
```

## Display Workbook Name

```
WkFileName = ActiveWorkbook.Name
MsgBox WkFileName
'WkFileName = ActiveWorkbook.FullName
'MsgBox WkFileName
```

## Moving Copies of Shapes

```
Public Sub AddIcon(iconNumber As Integer, cellRow As Integer, cellColumn As Integer, offset As Integer)
```

```
    If iconNumber < 0 Or iconNumber > 5 Then Exit Sub
```

```
    Sheets("Sheet3").Select
    Select Case iconNumber
        Case ICON_PLAN_START:
            ActiveSheet.Shapes("START").Select
        Case ICON_PLAN_IMPLEMENT:
            ActiveSheet.Shapes("ENDE").Select
        Case ICON_PLAN_ACTION:
            ActiveSheet.Shapes("P_ACTION").Select
        Case ICON_ACTUAL_START:
            ActiveSheet.Shapes("GESTARTET").Select
        Case ICON_ACTUAL_IMPLEMENT:
            ActiveSheet.Shapes("ERLEDIGT").Select
        Case ICON_ACTUAL_ACTION:
            ActiveSheet.Shapes("I_ACTION").Select
    End Select
```

```
Selection.Copy
Sheets("Sheet1").Select
Cells(cellRow, cellColumn).Select
ActiveSheet.Paste
Selection.ShapeRange.Left = Cells(cellRow, cellColumn).Left + offset
Selection.ShapeRange.Top = Cells(cellRow, cellColumn).Top
End Sub
```

## Deleting Pictures on a Sheet

```
'clears all icon pictures from sheet 1
Public Sub DeleteAllShapes()
    Dim ws As Worksheet
    Set ws = Worksheets(1)
    ws.Select

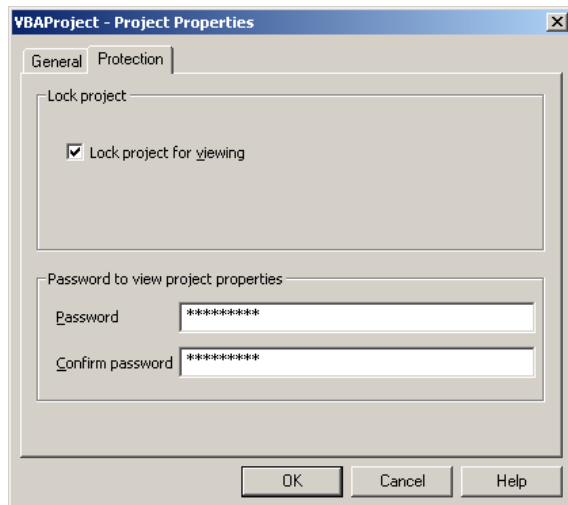
    Dim shp As Shape

    For Each shp In ws.Shapes
        If shp.Name = "START" Or shp.Name = "ENDE" Or shp.Name = "P_ACTION" Or shp.Name =
"GESTARTET" Or shp.Name = "ERLEDIGT" Or shp.Name = "I_ACTION" Then
            shp.Delete
        End If
    Next shp

    Set ws = Nothing
End Sub
```

## Password Protect Macros

Tools, Properties



## Load Webpage into Spreadsheet

```
Public Sub test()  
    Workbooks.Open Filename:="http://www.google.com"  
End Sub
```

## Retrieve Webpage (Entire or Table)

```
Public Sub test1()  
    Dim objWeb As QueryTable  
    Dim sWebTable As String  
    sWebTable = 2  
    Set objWeb =  
ActiveSheet.QueryTables.Add(Connection:="URL;http://www.google.com",  
Destination:=Range("A1"))  
    With objWeb  
        .WebSelectionType = xlEntirePage ' = xlSpecifiedTables  
        '.WebTables = sWebTable  
        .Refresh BackgroundQuery:=False  
        '.SaveData = True  
        .Refresh  
    End With
```

## Web Controller

```
'add reference to Microsoft Internet Controls  
Public Sub test()  
    Me.WebBrowser1.Navigate "http://www.google.com"
```

```
End Sub
```

## Detect Cell Change

```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
    If Target.Address = "$C$3" Then Target = "Here"
End Sub
```

## Arrays and Ranges

Copies one range of values to another range.

```
Range("A1:A3", "B2").Copy Destination:=Range("E5")
```

Copies an array to a range.

```
Dim s As Variant
s = Array(1, 2, 3)
Range("a1:c1") = s
```

Copies a range to an array.

```
Dim t As Variant
t = Range("A1:C1")
Range("b6:d6") = t
```

Copies a row of data to a column.

```
Dim s As Variant
s = Array(1, 2, 3)
Range("a1:a3") = application.Transpose(s)
```

Copies an array to an array to a range.

```
Dim num(3) As Integer
num(0) = 12
num(1) = 25
num(2) = 38
```

```
Dim s As Variant
s = num
Range("A1:C1") = s
```

Bigger Arrays and a Range

	A	B	C
1	12	0	0
2	13	14	0
3	0	0	43
4			

```
Dim num(2, 3) As Integer
num(0, 0) = 12
num(1, 0) = 13
num(1, 1) = 14
num(2, 2) = 43
```

```
Dim s As Variant
s = num
Range("A1:C3") = s
```

## More Arrays to Ranges

```
Dim A as Variant 'MUST be variant, no brackets
A = Range("SomeRange").Resize(10,20) 'reads 10x20 array starting at 'to
```

```
write back to sheet
Range("SomeRange").Resize(10,20) = A
```

## Status Bar

```
oldStatusBar = Application.DisplayStatusBar
Application.DisplayStatusBar = True
Application.StatusBar = "Please be patient..."
Workbooks.Open filename:="LARGE.XLS"
Application.StatusBar = False
Application.DisplayStatusBar = oldStatusBar
```

## SQL Query – Execute MSSQL Stored Procedure

```
'connect to SQL database
```

```
/******
```

```
c = "Provider=SQLOLEDB.1;Data Source=CH1SQL1;Initial Catalog=PP_M;User  
ID=Dog;Password=Cat"
```

```
If selRespGroup <= "0" Then
    selRespGroup = ""
End If
```

```
q = "EXEC sp_get_PDCA_measures_for_calc_tool2"  
q = q & " " & selRespGroup & "" & ", " & selResponsibility & ""
```

```
row = 3
```

```
col = 2
```

```
Call RunQuery(c, q, row, col)
```

```
/******
```

```
Sub RunQuery(c As String, q As String, row As Long, begcol As Long)
```

```
Dim cn As ADODB.Connection
```

```
Dim cmd As ADODB.Command
```

```
Dim rs As ADODB.Recordset
```

```
Dim f As ADODB.Field
```

```
Dim col As Long
```

```
Dim kount As Integer
```

```
On Error Resume Next
```

```
Set cn = New ADODB.Connection
```

```
cn.ConnectionString = c
```

```
cn.Open
```

```
If Err.Number <> 0 Then
```

```
MsgBox "Connection error: " & Err.description
```

```
Exit Sub
```

```
End If
```

```
kount = 0
```

```
Set cmd = New ADODB.Command
```

```
Set cmd.ActiveConnection = cn
```

```
cmd.CommandText = q
```

```
cmd.CommandType = adCmdText
```

```
Set rs = New ADODB.Recordset
```

```
Set rs.Source = cmd
```

```
cn.Errors.Clear
```

```
rs.Open
```

```
If Err.Number <> 0 Then
```

```
MsgBox "Query error: " & Err.description & " SQL=" & q
```

```
Exit Sub
```

```
End If
```

```
Do While Not rs.EOF
```

```
col = begcol
```

```
For Each f In rs.Fields
```

```
Cells(row, col) = f.Value
```

```
col = col + 1
```

```
Next f
```

```
rs.MoveNext
row = row + 1
kount = kount + 1
Loop
rs.Close
cn.Close
End Sub
```

## SQL Retrieve Data 1

```
' *****
' Configure sConnection string
' Define server name: Data Source=BSSQL1
' Define SQL catalog: Initial Catalog=PP_M
' Define user ID: User ID=Dog
' Define password: Password=Cat
'*****
Public Sub RetrieveSQLData()

'variable declarations
Dim dbConnection As ADODB.Connection
Dim dbCmd As ADODB.Command
Dim dbRS As ADODB.Recordset
Dim dbField As ADODB.Field
Dim nCol As Long
Dim nRow As Long

'clear Sheet1
Sheets("Sheet1").Select
Cells.Select
Selection.ClearContents
Range("A1").Select

'connect to database
On Error Resume Next
Dim sConnection As String
sConnection = "Provider=SQLOLEDB.1;Data Source=BSSQL1;Initial Catalog=PP_M;User
ID=Dog;Password=Cat"
Set dbConnection = New ADODB.Connection
dbConnection.ConnectionString = sConnection
dbConnection.Open
If Err.Number <> 0 Then
MsgBox "Connection error: " & Err.Description
```

```
Exit Sub
End If

'assign query to command
Dim sQuery As String
sQuery = "Select * from Chuck_Measures where left(ProjectNum, 3) < 206"
Set dbCmd = New ADODB.Command
Set dbCmd.ActiveConnection = dbConnection
dbCmd.CommandText = sQuery
dbCmd.CommandType = adCmdText

'retrieve data into database
Set dbRS = New ADODB.Recordset
Set dbRS.Source = dbCmd
dbConnection.Errors.Clear
dbRS.Open
If Err.Number <> 0 Then
    MsgBox "Query error: " & Err.Description & " SQL=" & sQuery
    Exit Sub
End If

'add header from retrieved data
Dim i As Integer
For i = 0 To dbRS.Fields.Count - 1
    Worksheets("Sheet1").Cells(0, i + 1) = dbRS.Fields(i).Name
Next i

'iterate through retrieved data and place into worksheet
nRow = 2
Do While Not dbRS.EOF
    nCol = 1
    For Each dbField In dbRS.Fields
        Worksheets("Sheet1").Cells(nRow, nCol) = dbField.Value
        nCol = nCol + 1
    Next dbField
    dbRS.MoveNext
    nRow = nRow + 1
Loop

'close connectino
dbRS.Close
dbConnection.Close

'housekeeping
```

```
Set dbConnection = Nothing  
Set dbRS = Nothing
```

```
End Sub
```

## SQL Retrieve Data 2

```
Public Sub RetrieveSQLData()
```

```
'variable declarations
```

```
Dim dbConnection As ADODB.Connection
```

```
Dim dbCmd As ADODB.Command
```

```
Dim dbRS As ADODB.Recordset
```

```
Dim dbField As ADODB.Field
```

```
Dim nCol As Long
```

```
Dim nRow As Long
```

```
'connect to database
```

```
On Error Resume Next
```

```
Dim sConnection As String
```

```
sConnection = "Provider=SQLOLEDB.1;Data Source=BSSQL1;Initial Catalog=PP_M;User  
ID=Dog;Password=Cat"
```

```
Set dbConnection = New ADODB.Connection
```

```
dbConnection.ConnectionString = sConnection
```

```
dbConnection.Open
```

```
If Err.Number <> 0 Then
```

```
    MsgBox "Connection error: " & Err.Description
```

```
    Exit Sub
```

```
End If
```

```
'assign query to command
```

```
Dim sQuery As String
```

```
sQuery = "Select * from Chuck_Measures" ' where left(ProjectNum, 3) < 206"
```

```
Set dbCmd = New ADODB.Command
```

```
Set dbCmd.ActiveConnection = dbConnection
```

```
dbCmd.CommandText = sQuery
```

```
dbCmd.CommandType = adCmdText
```

```
'retrieve data into database
```

```
Set dbRS = New ADODB.Recordset
```

```
Set dbRS.Source = dbCmd
```

```
dbConnection.Errors.Clear
```

```
dbRS.Open
```

```
If Err.Number <> 0 Then
    MsgBox "Query error: " & Err.Description & " SQL=" & sQuery
    Exit Sub
End If

'clear used area of sheet
Worksheets("Sheet1").Activate
ActiveSheet.UsedRange.Clear

'add header from retrieved data
Dim i As Integer
For i = 0 To dbRS.Fields.Count - 1
    Worksheets("Sheet1").Cells(1, i + 1).Value = dbRS.Fields(i).Name
Next i

'copy data into worksheet
Worksheets("Sheet1").Cells(3, 1).CopyFromRecordset dbRS

'close connectino
dbRS.Close
dbConnection.Close

'housekeeping
Set dbConnection = Nothing
Set dbRS = Nothing

End Sub
```