Angelfire Build your own FREE website at Angelfire.com **Share:** del.icio.us | digg | reddit | Twitter | facebook

Ads by Google

Excel,Macros,VBA Training Advanced Excel2003/2007 Training Excel VBA Macros Training in B'lore www.gitsacademy.com	**Submit Your Resume** Professionals With 2+ Yrs Exp. Get Top Paying Jobs on Shine.com. Apply www.Shine.com	**Export Outlook Emails** Export a selected group of emails & attachments to Excel. Free trial. www.encoresuite.com	**Excel VBA Developer Jobs** Companies are Hiring in India Submit Resume & Find the Right Job! www.monsterindia.com

Julian's Macro Tips

(last update 30/6/2010)

email (without spaces) : **julian.excel @ gmail.com** or

julian7_s @ yahoo.com

Excel Macros (VBA) tips for beginners. I do not use macros that have too many lines of codes as this may confuse beginers. I remembered when I started learning macro programming and going to sites by Chip Pearson & John Walkenbach and started seeing stars instead of VB!!!. Simplified macros are used for easy understanding for people like YOU and ME. By the way by, I am trained as an Accountant and not a programmer. I hope the this site gets you started in Macro Programming & Good Luck.

NOTE : Run the macros at your own risk. Macros cannot be un-done. Therefore always save your workbook before running any macros.

Auto Run [24/12/2001] *(back to top)*

Making your macros run automatically when opening your workbook. You can either use the Auto Open method or the Workbook Open method. These macros will display the message "Hello" when you open the workbook.

```
Sub Auto_Open()
Msgbox "Hello"
End Sub
```

This code would be located in the module. However if you use the second method, the code must be in the workbook (double click "This Workbook" in the explorer window). Click on the drop down list (that says General) and select Workbook. Click on the drop down list (that says declarations) and select Open.

```
Private Sub Workbook_Open()
Msgbox "Hello"
End Sub
```

Active Cell [5/1/2002] *(back to top)*

An active cell is the current cell that is selected. This term is used in many macros. This can be used as a marker. A good example is when you need to move from your current cell. **Refer to Moving your cursor macro**.

Adding Items to a combobox [15/3/2002] *(back to top)*

To add a combobox refer to **User Form**. To populate a combobox or a listbox is the same. You could add from the code or even from a range of cells in your spreadsheet. To add from the code, just add this line to your code.

```
ComboBox1.AddItem "Product A"
ComboBox1.AddItem "Product B"
```

Learn IT At Home Excel Word Window Front Page Power Point

Counting Rows & Columns & Sheets [27/10/2001] *(back to top)*

When you have selected a range, it is sometimes useful to know how many rows or columns you have selected as this information can be used in your macros (for eg when you have reached the end, you will know it is time to stop the macros. This macro will do the trick.

```
Sub Count()
myCount = Selection.Rows.Count    'Change Rows to Columns to count columns
```

```
MsgBox myCount
End Sub
```

The next macro counts the number of sheets instead. **Refer to Protecting all sheets macro** which uses this method.

```
Sub Count2()
myCount = Application.Sheets.Count
MsgBox myCount
End Sub
```

Carriage Return [10/11/2002] *(back to top)*
Sometimes you may want to put a line of text on the next row and not let it continue on the first row. See this example in a message box.

```
Sub TwoLines()
MsgBox "Line 1" & vbCrLf & "Line 2"
End Sub
```

Close All Files [23/3/2009] *(back to top)*
Sometimes you may want to close all files without saving. Doing it manually is a hassle with the question "Do you wanna save?"

```
Sub CloseAll()
Application.DisplayAlerts = False
myTotal = Workbooks.Count
For i = 1 To myTotal
   ActiveWorkbook.Close
Next i
End Sub
```

Copying A Range [5/1/2002] *(back to top)*
Copy data from a specific range can be done with this macro. Here data is copied from the current sheet to the activecell. (Refer to Active Cell)

```
Sub CopyRange()
Range("A1:A3").Copy Destination:=ActiveCell
End Sub
```

To copy from a range in another sheet (eg Sheet3) to the active cell you need to change the code to;

```
Sheets("sheet3").Range("A1:A3").Copy Destination:=ActiveCell
```

Counter [17/2/2002] *(back to top)*
To use a counter in your macro, just assign any cell to retain the value. In this example the cell A1 is chosen. Each time the macro is run, it adds the value 1 to the cell A1.

```
Sub Count()
mycount = Range("a1") + 1
Range("a1") = mycount
End Sub
```

Current Date [24/12/2001] *(back to top)*
It's a good idea to insert the current date when you save the file so that you can tell if it's the latest version. Of course this is shown under file properties but how many people know where to find it? You could also put the current date in the footer of your print out. It is ideal if the date does not change unless the file is saved. You can use this code. (On the drop down list that says declaration, select before save and you will see the 1st line of code shown below - more details **refer to Auto Run macro**)

```
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)
Range("A1") = Now       'Select any cell you want
End Sub
```

Current Cell Content [24/12/2001] *(back to top)*
Sometimes we need to know what the cell contains ie dates, text or formulas before taking a course of action. In this example a message box is displayed. Replace this with a macro should you require another course of action.

```
Sub ContentChk()
If Application.IsText(ActiveCell) = True Then
MsgBox "Text"          'replace this line with your macro
Else
If ActiveCell = "" Then
MsgBox "Blank cell"    'replace this line with your macro
Else
End If
If ActiveCell.HasFormula Then
MsgBox "formula"       'replace this line with your macro
Else
End If
If IsDate(ActiveCell.Value) = True Then
```

```
MsgBox"date"          'replace this line with your macro
Else
End If
End If
End Sub
```

Current Cell Position [10/3/2002] *(back to top)*
Sometimes we need to know the current cell position. This would do the trick.

```
Sub MyPosition()
myRow = ActiveCell.Row
myCol = ActiveCell.Column
Msgbox myRow & "," & myCol
End Sub
```

Learn IT At Home   Excel   Word   Window   Front Page   Power Point

Deleting Empty Rows [27/10/2001] *(back to top)*
To delete empty rows in a selected range we can use the following macro. The macro here uses the For Next Loop. First the macro counts the rows in a selected range to determine the when the macro should stop. The For Next statement acts as a counter.

```
Sub DelEmptyRow()
Rng = Selection.Rows.Count
ActiveCell.Offset(0, 0).Select
Application.ScreenUpdating = False
For i = 1 To Rng
If ActiveCell.Value = "" Then   'You can replace "" with 0 to delete rows with 'the value zero
Selection.EntireRow.Delete
Else
ActiveCell.Offset(1, 0).Select
End If
Next i
Application.ScreenUpdating = True
End Sub
```

The statement "Application.ScreenUpdating = False" prevents the screen from updating to ensure the macro runs faster and the screen will not flicker. Don't forget to set it back to "True".

Deleting Range Names [15/03/2002] *(back to top)*
To delete all the range names in your workbook, this macro will do the trick.

```
Sub DeleteNames()
Dim NameX As Name
For Each NameX In Names
ActiveWorkbook.Names(NameX.Name).Delete
Next NameX
End Sub
```

Duplicates (Highlight duplicates in Bold Red) [27/10/01] *(back to top)*
There are times you need to highlight duplicate data in your worksheet. This macro does the trick.

```
Sub DupsRed()
Application.ScreenUpdating = False
Rng = Selection.Rows.Count
For i = Rng To 1 Step -1
myCheck = ActiveCell
ActiveCell.Offset(1, 0).Select
For j = 1 To i
If ActiveCell = myCheck Then
Selection.Font.Bold = True
Selection.Font.ColorIndex = 3
End If
ActiveCell.Offset(1, 0).Select
Next j
ActiveCell.Offset(-i, 0).Select
Next i
Application.ScreenUpdating = True
End Sub
```

Emailing Workbook [2/12/2001] *(back to top)*
To email your current workbook the following code.

```
Sub Email()
ActiveWorkbook.SendMail recipients:="julsn@yahoo.com"
End Sub
```

EDIT macros [30/6/2003] *(back to top)*
Refer to Text Manipulation.

Errors in macros [1/2/2002] *(back to top)*

Ever had a macro running perfectly one day and the next day errors keep on popping up even though you never made changes to that macro? This is no fault of yours. Due to the excel VBA design, macro files get badly fragmented due to heavy editing of macros, insertion of modules & userforms. What you need to do is copy your macros else where, delete the macros, save the file without macros. Open the file again and import the macros and save it once more with the macros. You macros will run properly until it gets fragmented again at a later stage.

Error Trapping [4/1/2002] *(back to top)*

Trapping errors are important as users can do marvelous things to mess up you macros. Here you can use either of these 2 statements.

```
- On Error Resume Next   OR
- On Error Goto ErrorTrap1
  ... more lines of code
  ErrorTrap1:
  ... more code (what to do if there is an error)
```

The first statement will allow the macro to continue the next line of code upon hitting an error but the second statement will run an alternative code should there be an error.

Learn IT At Home   Excel   Word   Window
Front Page   Power Point

Excel Functions [8/2/2002] *(back to top)*

Using Excel functions in VBA is almost the same as using them in a spreadsheet. For example to round an amount to 2 decimal places in a spreadsheet would be;

```
=round(1.2345,2)
```

In VBA you would need to use the term Application followed by the function ie;

```
ActiveCell = Application.round(ActiveCell, 2)
```

For more examples **see Rounding Numbers**

Expiry Dates for Workbook / Macro [4/1/2002] *(back to top)*

See Security in Excel.

For, Next Loop [5/1/2002] *(back to top)*

See Deleting Empty Rows or Protect All Sheets. A point to note is, try not to use the For, Next loop because this method is very slow unless of course you don't know how to write your macro another way.

Flickering Screen [27/10/2001] *(back to top)*

Sometimes when you run a macro, the screen flickers a lot due to the screen updating itself. This slows the macro done especially when the macro has a lot of work to do. You need to include the statement as shown below.

Also **see Deleting Empty Rows**

```
Application.ScreenUpdating = False
```

You need to set the screen updating back to true at the end of the macro.

Functions [31/1/2002] *(back to top)*

Creating function is useful as complicated formulas can be made easier in code than in a spread sheet. Formulas can be protected so that users cannot see or modify them. The example I use will calculate tax using the **Select Case Statement**. Here's the scenario.

First $2500 is tax free.
Next $2500 is taxable at 5%.
Anything above $5000 is taxable at 10%.
In cell A1 type Income and in cell B1 type in your income in numbers say $20000.
In cell A2 type Tax payable and in cell B2 type =tax(B1).
Put the following code in a module. The tax payable here would be $1625.

```
Public Function tax(income As Single)
Select Case income
Case Is <= 2500
tax = 0
Case Is <= 5000
tax = (income - 2500) * 0.05
Case Else
tax = (income - 5000) * 0.1 + 125
End Select
End Function
```

Goto (a range) [27/10/2001] *(back to top)*

To specify a macro to go to a specific range you can use the Goto method. Here I have already named a range in my worksheet called "Sales". You may also use an alternative method ie the Range select method. Naming a range in excel is recommended rather than specifying an absolute cell reference.

```
Sub GoHere()
Application.Goto Reference:="Sales"   OR   Range("Sales").Select
End Sub
```

Going to the 1st Sheet [27/10/2001] *(back to top)*

You can select the first sheet of the workbook without knowing the name of the sheet by referring to it by the index.

```
Sub FirstSheet()
Sheets(1).Select
End Sub
```

GoTo Sheet [18/03/2005] *(back to top)*

Sometimes we have many sheets or sheets with long names & we cannot view them all. You can select which sheet to go to by using this macro.

```
Sub Go2sheet()
myShts = ActiveWorkbook.Sheets.Count
For i = 1 To myShts
myList = myList & i & " - " & ActiveWorkbook.Sheets(i).Name & " " & vbCr
Next i
Dim mySht As Single
mySht = InputBox("Select sheet to go to." & vbCr & vbCr & myList)
Sheets(mySht).Select
End Sub
```

Hiding Sheets [27/10/2001] *(back to top)*

To hide your worksheet from users you can use the following code.

```
Sub HideSheet()
Sheet1.Visible = xlSheetVeryHidden
End Sub
```

If you hide your sheets this way, users will not be able to unhide them using the menus. Only using VB codes will be able to display the sheets again.

Hiding Excel [3/9/2002] *(back to top)*

You can hide the Excel application with this macro. This disables the user from using the excel menus. Don't forget to set it back to visible.

```
Sub HideExcel()
Application.Visible = False
End Sub
```

Input Box [27/10/2001] *(back to top)*

When you need to get input from users, you can use input boxes. This macro will ask for the user's name and will display a message "Hello" plus the user's name.

```
Sub GetInput()
Dim MyInput      'This line of code is optional
MyInput = InputBox("Enter your name")
MsgBox ("Hello ") & MyInput
End Sub
```

Inserting Rows [4/1/2002] *(back to top)*

To insert rows required by a user is easy. Here the input box is used so that a user can define the number of rows required.

```
Sub InsertRow()
Dim Rng
Rng = InputBox("Enter number of rows required.")
Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(Rng - 1, 0)).Select
Selection.EntireRow.Insert
End Sub
```

Here the macro uses the range technique where a range is first selected and then subsequently rows are inserted.

IF, Then Statement [27/10/2001] *(back to top)*

See Protect All Sheets

Joining Text Together [10/3/2003] *(back to top)*

There are times where we import text file into Excel an we get text that are separated. I received an email asking how put these text together. Select across your cells first and run this macro.

```
Sub JoinText()
myCol = Selection.Columns.Count
For i = 1 To myCol
ActiveCell = ActiveCell.Offset(0, 0) & ActiveCell.Offset(0, i)
ActiveCell.Offset(0, i) = ""
```

```
Next i
End Sub
```

Killing Files [1/12/2001] *(back to top)*
Killing or deleting files is easy. However the files must not be in used.

```
Sub Killfile()
Dim MyFile As String   'This line of code is optional
On Error Resume Next  'On hitting errors, code resumes next code
MyFile = "c:\folder\filename.xls"
kill MyFile
End Sub
```

Wildcards can be use. Replace the file name with * (use with caution!).

Killing The Current File [8/2/2002] *(back to top)*
Killing the current file you need to change it's status to read only.

```
Sub Killed()
Application.DisplayAlerts=False
ThisWorkbook.ChangeFileAccess xlReadOnly
Kill ThisWorkbook.FullName
ThisWorkbook.Close False
End Sub
```

Lower Case [27/10/2001] *(back to top)*
To change text in a selected range to lower case use this code.

```
Sub LowerCase()
Dim cell As Range
For Each cell In Selection.Cells
If cell.HasFormula = False Then
cell = LCase(cell)
End If
Next
End Sub
```

Last Available Row [23/3/2009] *(back to top)*
Many users need to know the next available row to input data. This code locates the next available row in column A

```
Sub LastRow()
Range("a65536").End(xlUp).Offset(1, 0).Select
End Sub
```

Learn IT At Home   Excel   Word   Window
Front Page   Power Point

Message Box [17/2/2002] *(back to top)*
When you need to communicate with users, you can use message boxes. This macro will display a message "This macro is created by Julian". The Message Box appearance can be customised to show whether it is Information, Critical Messages. Here the icon in the message box would be different. The buttons can also be customise to show extra Yes, No, Ok buttons. **(Refer to vbYesNo macro)**. This macro will show you 3 different styles.

```
Sub MyMessage()
MsgBox "This macro is created by Julian"
MsgBox "The icon is different", vbInformation
MsgBox "The top title is different", vbExclamation, "Julian's Tips"
End Sub
```

Modeless Forms [10/11/2002] *(back to top)*
Sometimes you want to allow users to be able to switch between your form and your spreadsheet by clicking on either one. All you need to do is set the form property of Show Modal to False or you can try this. However this is only for Excel 2000 & above.

```
Sub myForm()
UserForm.show vbModeless
End Sub
```

Moving your cursor [27/10/2001] *(back to top)*
Sometimes you need to move your cursor around your worksheet to re-position it before running the next step of a macro. The movement here uses the row, column position method. **Also see (Visible Rows)**

```
Sub Down()
ActiveCell.Offset(1, 0).Select
End Sub
Sub up()
ActiveCell.Offset(-1, 0).Select
End Sub
```

```
Sub Right()
ActiveCell.Offset(0, 1).Select
End Sub
Sub Left()
ActiveCell.Offset(0, -1).Select
End Sub
```

Protecting / Unprotecting a sheet [27/10/2001] *(back to top)*

The macros below will protect/unprotect the current worksheet with a password.

```
Sub ProtectSheet()
Dim Password   'This line of code is optional
Password = "1234"
ActiveSheet.Protect Password, True, True, True
End Sub
Sub UnProtectSheet()
Password = "1234"
ActiveSheet.Unprotect Password
End Sub
```

Protecting all sheets [27/10/2001] *(back to top)*

To protect all the sheets this macro uses all the methods contained in this page (see counting sheets). The **If, Then statement** is also used here. This tests for a condition and if the condition is TRUE, then the macro continuous the next line of code. In this case it will END the macro. If the condition is NOT TRUE, then it will go to the following line which in this case is to select the next sheet. You will also notice the **For, Next statement** is also used. This acts as a counter to tell the macro how many loops to run. In this case if there are 3 sheets, the macro will run 3 times protecting all the 3 sheets.

```
Sub protectAll()
Dim myCount      'This line of code is optional
Dim i            'This line of code is optional
myCount = Application.Sheets.Count
Sheets(1).Select   'This line of code selects the 1st sheet
For i = 1 To myCount
ActiveSheet.Protect
If i = myCount Then
End
End If
ActiveSheet.Next.Select
Next i
End Sub
```

Learn IT At Home
Excel Word Window
Front Page Power Point

Protecting your VB code [10/3/2002] *(back to top)*

To protect your VB code from being seen by others, all you need to do is go to the project explorer, point at your project and right click on it. Select VBA project properties, click on the protection tab and check the Lock project for viewing and key your password. That's it.

Random numbers [27/10/2001] *(back to top)*

For macros to generate random numbers, the code is takes this format - Int ((upperbound - lowerbound +1) * Rnd + lowerbound). Where the Upperbound is the largest number random number to be generated and Lowerbound is the lowest.

```
Sub RandomNo()
Randomize
MyNumber = Int((49 - 1 + 1) * Rnd + 1)
MsgBox ("The random number is ") & (MyNumber)
End Sub
```

In this case the random numbers that will be generate is between 1 and 49.

Range Names [3/9/2002] *(back to top)*

Assigning range names to a range of cells.

```
Sub RngName()
Selection.Name = "myRange"
End Sub
```

Resizing a Range [3/9/2002] *(back to top)*

Resizing a range is simple. You can apply this to inserting rows & columns or to expand a selected range. This macro resizes the range to 7 rows by 7 columns.

```
Sub ResizeRng()
Selection.Resize(7,7).Select
End Sub
```

Rounding Numbers [8/2/2002] *(back to top)*

Here I will show how to perform different types of rounding. Key in 12345 in any active cell and run the following code.

```
Sub Round()
ActiveCell = Application.round(ActiveCell, -3)
End Sub
```

This code round to the nearest 1000 thus giving the value 12000.

```
ActiveCell = Application.Ceiling(ActiveCell, 1000)
Replace with this line of code and it will round up to the next 1000 ie 13000
ActiveCell = Application.Floor(ActiveCell, 1000)
Replace with this line of code and it will round down to the next 1000 ie 12000
```

Running A Sub Routine [5/1/2002] *(back to top)*

To run another macro from within a macro you need to use the Call statement.

```
Sub Macro1()
Msgbox("This is Macro1")
Call Macro2   'This calls for Macro2 to run
End Sub
```

Learn IT At Home
Excel Word Window
Front Page Power Point

Saving a file [23/3/2009] *(back to top)*

There are times you may want a macro to save a file automatically after running a macro. The second macro will save the file with a name called "MyFile". You may specify the path if you need to. The last macro saves all opened workbooks.

```
Sub Save()
ActiveWorkbook.Save
End Sub

Sub SaveName()
ActiveWorkbook.SaveAs Filename:="C:\MyFile.xls"
End Sub

Sub SaveAll()
myFile = ActiveWorkbook.Name
   ActiveWorkbook.Save
   ActiveWindow.ActivateNext
Do While myFile <> ActiveWorkbook.Name
   ActiveWorkbook.Save
   ActiveWindow.ActivateNext
Loop
End Sub
```

Security in Excel [4/2/2002] *(back to top)*

Level 1 - To protect your excel files, there are a few steps required to make it more difficult for other users to by pass security. To prevent changes made to the worksheet, you need to protect your worksheet. See protecting sheets. To prevent sheets from being renamed, moved or deleted, protect the workbook. However protection of worksheets and workbook can easily be hacked using macros as shown by an Excel developer. I believe the next level of protection is protecting your macros. To protect your macros, point at your project in the explorer window, right click on it and select VBA project properties, click on the Protection tab, check on Lock Project for Viewing and next key in your password and you're done. Now the project cannot be viewed or amended.

Level 2 - The next step is to ***force*** the user to ***enable*** your macro when opening your file. The best way is to use a macro to hide the important sheets (see Hiding sheets) when saving your file. Upon opening the file, a macro will be used to unhide these sheets. If the user disables the macros when opening the worksheet, they will not be able to view your worksheet unless they allow the macro to run.

Level 3 - The final step is to put an expiry date for your worksheet or your macro. However this has a draw back as the user may change the system date of the computer to by pass the step. Alternatively you can use a counter **(Refer Counter Macro)** to allow a fixed number of access to your worksheet or macro. Here you need to save the counter value each time the file or macro is used. Upon reaching the defined limit, disable the macro or disable the access of your worksheet.

The steps mentioned above are not 100% fool proof. But it will keep normal users out but not hackers and crackers. Here I will not supply the code as this can be lengthy and may be difficult to understand but I believe these steps may be useful to some of you out there.

Select Case Statement [31/1/2002] *(back to top)*

This is a useful statement to use when you have many conditions. Too many IFs in your code will only make you more confuse. **See Functions macro**.

Sentence Case [6/6/2005] *(back to top)*

To change text in a selected range to sentence case use this code. This code was supplied by **Simon Huggins.** He did a fine job of making the code work for both earlier & current versions of Excel. Thanks for the contribution.

```
<> Sub SentenceCase()
For Each cell In Selection.Cells
s = cell.Value
Start = True
For i = 1 To Len(s)
ch = Mid(s, i, 1)
    Select Case ch
    Case "."
    Start = True
    Case "?"
    Start = True
    Case "a" To "z"
    If Start Then ch = UCase(ch): Start = False
    Case "A" To "Z"
    If Start Then Start = False Else ch = LCase(ch)
    End Select
Mid(s, i, 1) = ch
Next
cell.Value = s
Next
End Sub
```

Select Data Range [3/3/2006] *(back to top)*

This is a useful when you need to select the whole range of data to copy to another sheet (especially a large range).

```
Sub SelAllData()
Application.ScreenUpdating = False
Dim myLastRow As Long
Dim myLastColumn As Long
Range("A1").Select
 On Error Resume Next
   myLastRow = Cells.Find("*", [A1], , , xlByRows, xlPrevious).Row
   myLastColumn = Cells.Find("*", [A1], , , xlByColumns, xlPrevious).Column
   myLastCell = Cells(myLastRow, myLastColumn).Address
myRange = "a1:" & myLastCell
Application.ScreenUpdating = True
Range(myRange).Select
End Sub
```

Learn IT At Home   Excel   Word   Window
Front Page   Power Point

Text Manipulation [30/6/2003] *(back to top)*

I received many queries regarding text manipulation. Here are some useful text functions which you could use to EDIT your text.

```
Sub myEdit()
MsgBox Left("abcd", 2)     'Displays 2 characters from Left
MsgBox Right("abcd", 2)    'Displays 2 characters from Right
MsgBox Len("abcd")         'Displays number of characters
End Sub
```

Text Box Calculations (In user forms) [24/12/2006] *(back to top)*

To perform calculations using text boxes in user forms, you will need to validate the data first, otherwise you will be in for surprises. Assuming you want to add two values in 2 separate textbox and assign the answer to another.

```
Textbox1 = Val(textbox2)+Val(textbox3)
```

Timer [1/2/2002] *(back to top)*

To create a macro to measure time before executing the next line of code use this simple code.

```
Sub timer()
Application.Wait Now + TimeValue("00:00:10")
MsgBox("10 sec has elasped")
End Sub
```

Title Case [27/10/2001] *(back to top)*

To change text in a selected range to title case use this code.

```
Sub TitleCase()
Dim cell As Range
For Each cell In Selection.Cells
If cell.HasFormula = False Then
cell = Application.Proper(cell)
```

```
End If
Next
End Sub
```

Top of the screen [10/11/2002] *(back to top)*
To make the activecell be at the top of the screen & to the left on the screen try this.

```
Sub TopLeft()
ActiveCell.Select
With ActiveWindow
.ScrollColumn = ActiveCell.Column
.ScrollRow = ActiveCell.Row
End With
End Sub
```

Upper Case [27/10/2001] *(back to top)*
To change text in a selected range to upper case use this code.

```
Sub UpperCase()
Dim cell As Range
For Each cell In Selection.Cells
If cell.HasFormula = False Then
cell = UCase(cell)
End If
Next
End Sub
```

User Forms [8/3/2005] *(back to top)*
Adding user forms in your macro is simple. With user forms you can create GUIs (Graphical User Interface) for user who do not have much excel knowledge and make you excel programs more professional looking. Go to your Visual Basic Editor window & click on Insert, select user form and a user for will appear along with the toolbox. Now you can add labels, buttons, text boxes and many more items. The property window will allow you to customise your user form. To display your user form use these codes.

```
<>UserForm1.show     'to load form
Unload Me           'to close the form with a macro
```

Loading User forms with MultiPage [8/3/2005] *(back to top)*
To select a page in a Multipage object is fairly simple. Just remember the page 1 has a value of 0, page 2 a value of 1 and so forth. To load a form with a specific page in mind, try using these codes.

```
Sub page2()
UserForm1.MultiPage1.Value = 1  'this sets page 2
UserForm1.Show  'this displays the user form after page 2 has been set
End Sub
```

vbYesNo [17/2/2002] *(back to top)*
There are times you may want users to click Yes or No. Just insert this line of code. Here the Select Case statement is used.

```
YesNo = MsgBox("This macro will ... Do you want to continue?", vbYesNo + vbCritical, "Caution")
Select Case YesNo
Case vbYes
'Insert your code here if Yes is clicked
Case vbNo
'Insert your code here if No is clicked
End Select
```

Visible Rows (selecting) [24/12/2006] *(back to top)*
The normal way of selecting the next row cannot be used where there are hidden rows or filtered data. To select the next visible row in a filtered list we test each row until we find the next visible row.

```
Sub NextVisibleRow()
ActiveCell.Offset(1, 0).Select
Do While ActiveCell.EntireRow.Hidden = True
ActiveCell.Offset(1, 0).Select
Loop
End Sub
```

Take note that there are many ways of writing a macro which produces the same effect. Macro programming takes a lot of imagination and creativity. There is no one correct method. The macros on this site may not be very efficient but I have used a simple approach and this is a good start to learn macro programming for those who are new in this area. I have also simplified the macros so that it would be easier for you to understand.

I do get lots of email from all over the world saying that this is a great site for beginners. Truly this is encouraging & I THANK YOU ALL for your support. Please note that I may not answer all your emails as I get more emails than I can cope with & I'm pretty busy with work too. I will try to answer some of them if I can but it may take some time. Also note that I do not provide the full source codes but I will try to point you in the right direction. Once again a big Thank You for visiting my site.

Julian's Excel Tips

Julian's Excel Solutions

Link to other Excel Developers

GOOGLE Group on Excel Programming

Training Provider In Malaysia
Centrilinc Sdn Bhd - **Centrilinc is registered with the Microsoft Partner Program, and courses conducted by us are claimable under the HRDF / PSMB (Pembangunan Sumber Manusia Berhad) SBL Scheme.**

Microsoft Office Training

In today's highly competitive job market, success will determined by your ability in fully utilising the power of computers. Build that competitive edge over the others. Corporate / personal training available in **Petaling Jaya, Kuala Lumpur** and surrounding areas.

I have been conducting computer training for over 10 years. With almost 20 years of using spreadsheets in the field of finance I can assure you that quality training with real life examples is what you will get.

For more details email **julian.excel @ gmail.com** Don't get left behind in the IT world. On site training available for organisations.

Courses available in Microsoft Excel are :

Excel - Basic (1 day) / Intermediate (1 day)	Managing A Database in Excel (1 day)
Excel - Advance (2 days)	Excel Functions & Formulas (2 days)
Excel Pivot Tables (1 day)	Excel Macro Programming - Basic (2 days)
Excel - Charts (1 day)	Excel Macro Programming - Intermediate (2 days)
Excel - Financial Analysis & Modeling (2 days)	Not listed above? Just tell me what you need

<< A Passion To Excel In All Things >>

Back to

SHRIRAM Capital
INCLUSIVE GROWTH SINCE 1974
• TRANSPORT FINANCE • CONSUMER FINANCE • GENERAL INSURANCE
• LIFE INSURANCE • CHIT FUNDS • EQUIPMENT FINANCE
• STOCK BROKING • WEALTH MANAGEMENT
www.shriramcapital.in
Replay

Site Sponsors WHOWHERE? Gamesville Rhapsody WIRED