


Macros and Excel

Innovation and Technology Center

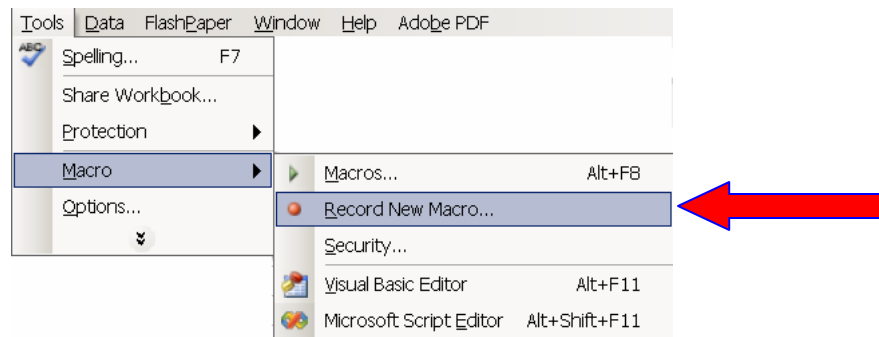
Macros are a sequence of keystrokes, mouse actions and other programming instructions that you record and get stored in a macro module, that can be used later on. By utilizing Macros, you eliminate repetitive work and work more efficiently. You automate complex tasks.

There are 3 Parts to a Macro:

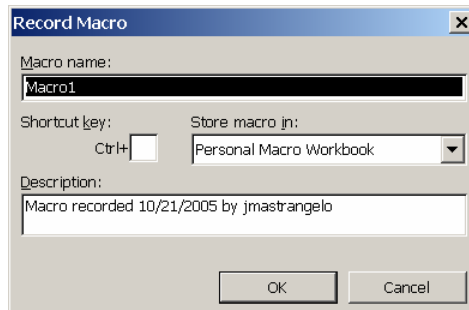
Macro name Shortcut key (optional) Macro steps	
--	---

Macros are commands expressed in Visual Basic that you execute when you run the macro.

To initiate a macro, go to Tools, Macro, Record.



You will then be prompted to give the Macro a name and a place to store it. If you will be using the macro for a number of Excel sheets, you would store it in Personal Macro Workbook. If it is only going to be used for the specific workbook, select this workbook.

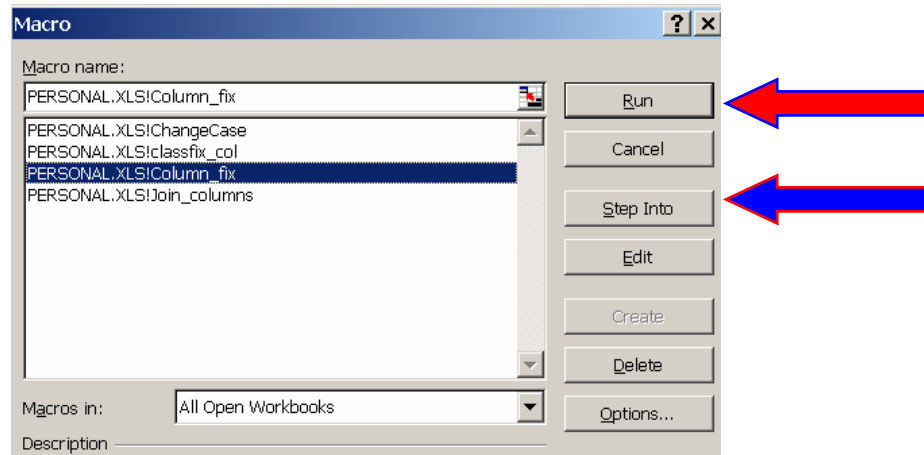


Macros and Excel Innovation and Technology Center

As a rule, it is best to select Personal Macro Workbook. A macro that is commonly useful when working with spreadsheets is one that will adjust all of the columns. We will first select record, name the macro, and then adjust all of the columns manually from A-Z. When complete, hit stop recording.



Once created, you can then call up the macro by going into Tools, Macro, Macros and select the macro you named. Proceed to select **run**.



Now lets look at the Macro code to see what the code looks like. Select [Step into](#).

```
Sub Column_fix()  
,  
    Column_fix Macro  
    Macro recorded 9/13/2005 by jmastrangelo  
,  
,  
  
    Columns("A:A").EntireColumn.AutoFit  
    Columns("B:B").EntireColumn.AutoFit  
    Columns("C:C").EntireColumn.AutoFit  
    Columns("D:D").EntireColumn.AutoFit  
    Columns("E:E").EntireColumn.AutoFit  
    Columns("F:F").EntireColumn.AutoFit  
    Columns("G:G").EntireColumn.AutoFit  
    Columns("H:H").EntireColumn.AutoFit  
    Columns("I:I").EntireColumn.AutoFit  
    Columns("J:J").EntireColumn.AutoFit  
    Columns("K:K").EntireColumn.AutoFit  
    Columns("L:L").EntireColumn.AutoFit  
    Columns("M:M").EntireColumn.AutoFit  
    Columns("N:N").EntireColumn.AutoFit  
    Columns("O:O").EntireColumn.AutoFit  
    Columns("P:P").EntireColumn.AutoFit  
    Columns("Q:Q").EntireColumn.AutoFit  
    Columns("R:R").EntireColumn.AutoFit  
    Columns("S:S").EntireColumn.AutoFit  
    Columns("T:T").EntireColumn.AutoFit  
    Columns("U:U").EntireColumn.AutoFit  
    Columns("V:V").EntireColumn.AutoFit  
    Columns("W:W").EntireColumn.AutoFit  
    Columns("X:X").EntireColumn.AutoFit  
    Columns("Y:Y").EntireColumn.AutoFit  
    Columns("Z:Z").EntireColumn.AutoFit  
End Sub
```

You can see that the code is Auto fitting all columns A-Z.

Macros and Excel
Innovation and Technology Center

You can find many Macros on internet sites. However, Beware because macros can contain malicious code.



Some of my favorite Macros are joining two columns and changing the case text from Upper to Proper.

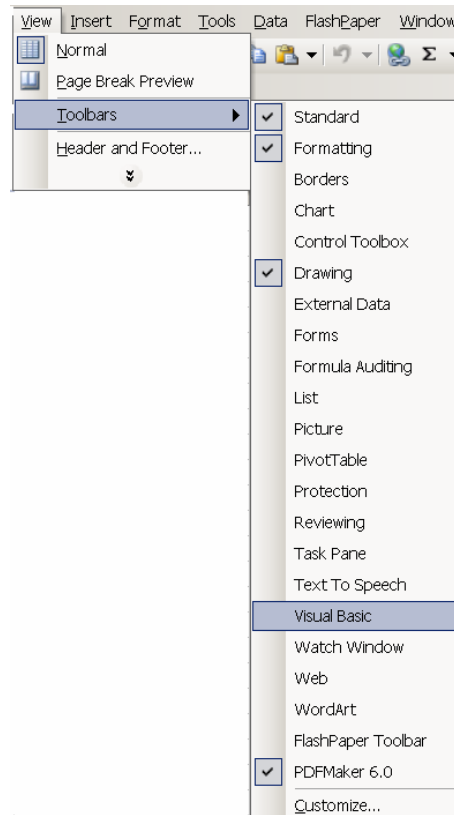
The code for each is below. We will create 2 macros and assign a button to them for execution that will accomplish this task.

[Join two columns](#)

```
Sub Join_columns()  
,  
' Join_columns Macro  
' Macro recorded 9/13/2005 by jmastrangelo  
  'Join cells in selected portion of a row together  
  Application.ScreenUpdating = False  
  Application.Calculation = xlCalculationManual  
  On Error Resume Next  
  Dim iRows As Long, mRow As Long, ir As Long, ic As Long  
  iRows = Selection.Rows.Count  
  Set lastcell = Cells.SpecialCells(xlLastCell)  
  mRow = lastcell.Row  
  If mRow < iRows Then iRows = mRow 'not best but better than nothing  
  iCols = Selection.Columns.Count  
  For ir = 1 To iRows  
    newcell = Trim(Selection.Item(ir, 1).Value)  
    For ic = 2 To iCols  
      trimmed = Trim(Selection.Item(ir, ic).Value)  
      If Len(trimmed) <> 0 Then newcell = newcell & " " & trimmed  
      Selection.Item(ir, ic) = ""  
    Next ic  
    Selection.Item(ir, 1).Value = newcell  
  Next ir  
  Application.Calculation = xlCalculationAutomatic  
  Application.ScreenUpdating = True  
End Sub
```

Macros and Excel Innovation and Technology Center

We will open up an Excel spreadsheet and select Toolbars, Visual Basic.



The Visual Basic tool bar will appear.



When you select the hammer and wrench another tool box appears. Hit the button.



Macros and Excel Innovation and Technology Center

When you select the button on the tool bar, a cross will appear in place of your mouse. You will have to decide where you would like to place the button that will execute your macro. Click the mouse once, on the position you would like the button to appear. The button will be populated as displayed below.



If you double click on the mouse, you will be launched to the Visual Basic Editor and you will see the display below.

The screenshot shows the Visual Basic Editor interface. The code window displays the following code:

```
Private Sub CommandButton1_Click()  
End Sub
```

A blue arrow points to the 'End Sub' line. A yellow callout box with a red border contains the following text:

Copy the macro and place it in between Private Sub and End Sub. You can change the appearance of the button by going into the properties menu. Select the item you want and make the changes. When you have made all of your changes, hit save and then hit Alt F11 to get you back into Excel.

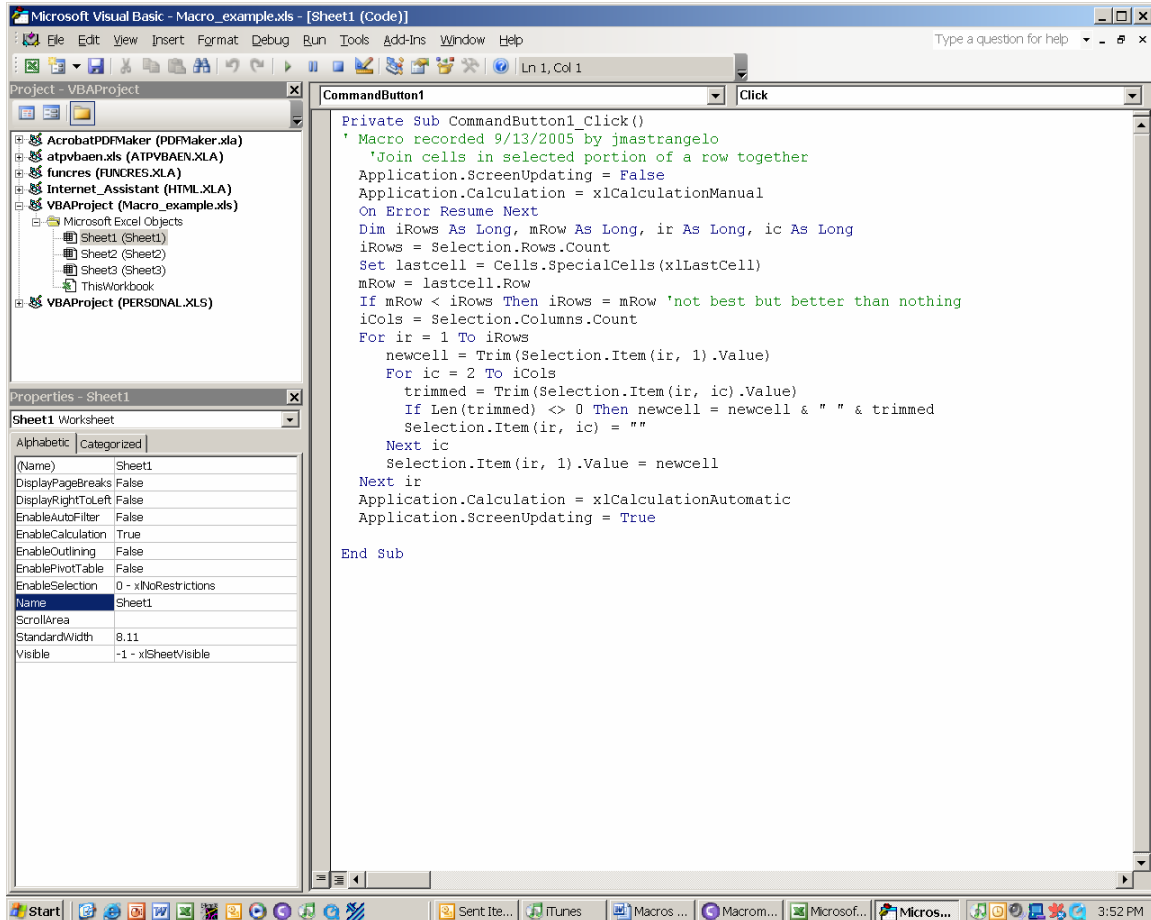
The Properties window for CommandButton1 is visible on the left, showing various properties such as Name, Accelerator, AutoLoad, AutoSize, BackColor, BackStyle, Caption, Enabled, Font, ForeColor, Height, Left, Locked, MouseIcon, MousePointer, Picture, PicturePosition, Placement, PrintObject, Shadow, TakeFocusOnClick, Top, Visible, Width, and WordWrap.

A yellow callout box with a red border on the left side of the Properties window contains the following text:

This is the properties area, where you can change the color of the button and caption

Macros and Excel Innovation and Technology Center

After it has been copied, it will look like the display below in the Visual Basic Editor. Hit save and then F11 to get out.



In the below example, the caption has been changed from CommandButton1 to Join Data. The macro is designed to take the data in columns A and B and join them into one. If you highlight the data in Column A and Column B and then hit the button, the data will join. Try to avoid highlighting an entire column because the macro will run too long.

	A	B	C	D	E	F
1	John	Michaels				
2	Sue	Canny				
3	Ruth	Jones				
4	Neil	Mackey				
5						
6						
7						

Macros and Excel
Innovation and Technology Center

The results of the macro are displayed below.

John Michaels
Sue Canny
Ruth Jones
Neil Mackey

Now you can repeat the process and use the following code to create a macro that will convert UPPER CASE LETTERS to Proper Case.

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

Assign a button to this code and save the sheet. You can then paste your data in anytime you need to change a case on data.



You did it!

Macros and Excel
Innovation and Technology Center



Thank you for participating in the class

Contact User Services for assistance with all your desktop needs

X4397