01 Introduction to macros

*(Videos: **My first macro – Part 1** youtu.be/hi4QCQy1QWg and **Part 2**: youtu.be/KFOVs3qBomY)*

What is a macro?

One way to think of it is that it's an 'app', like the ones you use on your phone. So it's a bit of computer wizardry that 'does things'. We might say...

> A macro is an app.
> A macro is a computer program.
> A macro is a bit of computer code.
> A macro is a computer subroutine.

Unlike with apps, you do have to 'handle' the computer code, but if you know how to copy and paste, you'll be fine.

Macros use a computer language called **Visual Basic for Applications** (**VBA**), and just as you might copy and paste some text out of Word into an email (or out of an email into Word), so you need to copy the text of macros into VBA. That's as complicated as it gets.

The elements of a macro

Here's a very simple macro – I've coloured the important bits.

```
Sub SwapCharacters()
Selection.MoveEnd 1
Selection.Cut
Selection.MoveLeft 1
Selection.Paste
End Sub
```

I said a macro is a subroutine, right? So the **Sub**(routine) and **End Sub**(routine) are the 'markers' that show where the macro starts and ends. It's important, when copying and pasting macros, to maintain this pattern of **Sub**/**End Sub**.

The **SwapCharacters** is the name of the macro. This is important, because that's how you tell the computer what you want it to do: "Please run the macro called **SwapCharacters**".

The name must be unique – the computer gets confused if you paste two macros into VBA that have the same name.

To take advantage of what the macros in this book can do, you need to know where to store them (within the VBA application) and how to run them from within Word.

Storing your macros

Macros can be stored in various places in your computer, but the simplest place is within Word's Normal template. This is the most convenient place because they then are available for use with any file(s) that you are working on.

(You might hear people saying that it is dangerous to store macros in the Normal template. It is true that there *were* once problems with doing so, but that was back in the days of Word 97 and 2003. As far as I'm aware, this hasn't been a problem since Word 2007 onwards.)

The macros are stored inside the Normal template, one after the other, in a Visual Basic file called *Normal.NewMacros*. Here's part of my *Normal.NewMacros* file to show you what the macros look like when viewed in VBA (please don't worry about the content of these macros, or what they do – just note the way that they are stored):

```
..............................................................................................................................
Sub SubscriptSwitch()
```

```vba
' F4
Selection.Font.Subscript = Not Selection.Font.Subscript
End Sub
```

....

```vba
Sub SuperscriptSwitch()
' F5
Selection.Font.Superscript = Not Selection.Font.Superscript
End Sub
```

....

```vba
Sub Mu()
' Version
Selection.TypeText Text:=ChrW(956)
End Sub
```

....

```vba
Sub FontRemove()
' Version
On Error GoTo ReportIt
Selection.Font.Reset
Exit Sub
ReportIt:
beep
End Sub
```

....

There are just three important things to understand here:

• The macros are *all stored together in a single file*, but they don't have to be in any particular order – they are run from Word by name.

• It's a single file, so *you can select all the macros (Ctrl-A) and copy them (Ctrl-C)*. You can then paste them somewhere else, perhaps in a Word file, as a way of keeping a backup copy.

• *When adding or removing macros be very careful not to break that repeated pattern of* `Sub ... End Sub` which I've again highlighted to make it stand out.

Tip: Below, I explain about how to add a macro, but rather than just reading it in theory, why not choose a particular macro and actually install it. You could try the transpose characters macro that I gave as an example above.

```vba
Sub TransposeChars()
    Selection.MoveRight 1, Extend:=wdExtend
    Selection.Cut
    Selection.MoveLeft 1
    Selection.Paste
End Sub
```

It transposes adjacent characters, say from 'Pual' to 'Paul' – you just put the cursor between the 'u' and the 'a' and run the macro.

Adding macros

Macros can be added simply by copying them from this book and pasting them into *Normal.NewMacros*. So, in this book, make sure that you select the complete macro, from

```vba
Sub Something()
```
to
```vba
End Sub
```

and press Ctrl-C to copy it. Then open VBA (see below), decide where to put the new macro and press Ctrl-V to paste it in. I tend to put new macros down at the bottom of the file (click Ctrl-End to get to the bottom), but it really doesn't matter, because Word calls them by name.

(N.B. This book is arranged as two files: this file has the descriptions of the macros, and the other file, 'TheMacros', has the actual macro listings.)

The difficult thing is knowing how to open *Normal.NewMacros* in VBA – it is different on different computer systems!

Installing a macro from scratch

(See also video: My First Macro – Part 1 (6:04): youtu.be/hi4QCQy1QWg)
(See also video: My First Macro – Part 2 (5:45): youtu.be/KFOVs3qBomY)
(See also video: Macro Starter Pack (5:42): youtu.be/IeMnmtJT2Ys)

Macros can be added simply by copying them from an electronic book, from a website or from an email, and pasting them into a program called Visual Basic for Applications (VBA), where they will be stored in the 'Normal template', as it's called. I'll try to explain in a number of small steps.

Step 1: Copy the macro
Wherever the macro comes from, you first have to select it and then do a Ctrl-C to copy it. (*On a Mac, that's Command-C, ⌘-C.*)

However, you do need to make very sure that you select the *complete* macro, i.e. from the

```
Sub SomethingOrOther()
```

down to and including the

```
End Sub
```

before you press Ctrl-C (*Mac: ⌘-C*) to copy it.

Step 2: Open VBA
VBA is a separate application that works alongside Word. The computer programs in VBA are called macros. These macros can be used from within Word without VBA actually being on screen. However, to install your macros in the first place, you have to open VBA, as follows:

Click on Alt-F8 (*Option-F8 on a Mac*), and it should open the Macros window. (If not, on Word 2003/4 you can use the menu: Tools–Macro–Macros, or on 2007 onwards View–Macros.)
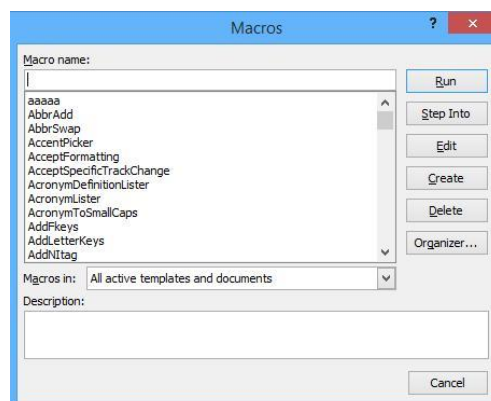
In the middle of this window is a list of all the macros that are currently installed in your computer. But of course, if no-one has yet put any macros in your computer, the list will have no items in it:
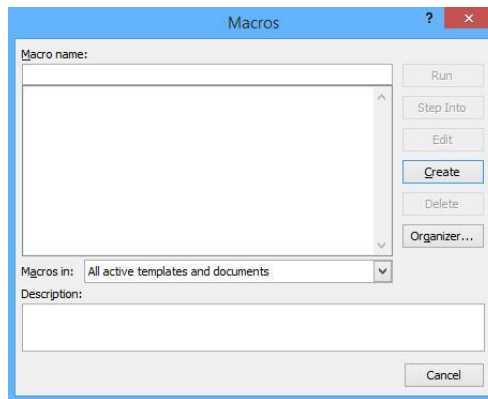
Now, in the top box (Macro name:) type the single word Dummy, and click the Create icon (fourth down on the right). VBA will now open, showing:

```
Sub Dummy()
'
'  Dummy Macro
'
'

End Sub
```

Step 3: Paste in your new macro
Select the whole of the Dummy Macro, from the `Sub Dummy()` line up to and including the `End Sub` line, and then click Ctrl-V (*Mac: ⌘-V*). This will put your new macro in place of the Dummy macro.

Step 4: Close VBA
You can usually do this with Alt-Q (⌘-*Q*), but you can also do it by clicking in the top right 'X' (Close) icon – but notice that there's another 'X' just below it, so click in the very top icon.

Step 5: Running your new macro
To run your new macro, use Alt-F8 (*Option-F8*), to open the Macros window again (or Tools–Macro–Macros or View–Macros). Look in the list of macros for the one you want, click on it and then click Run (top right button).

Once you are familiar with this, there's a simpler way to add a macro: Press Alt-F8 to open the Macros window, click on any one of the macro names, click on the Edit button, then you can paste in the new macro. However, you must be very careful where you put the new macro! You must be careful not paste the new macro inside one of the existing macros; it must go *after* and End Sub, and *before* the next Sub SomethingOrOther. Probably the safest is to use Ctrl-End to go to the very end of the VBA macros, and paste the new macro in there.