Macros 101: An Introduction to Macros, and Some Macros for Your Editorial Toolbox

Overview

- Basic definitions
- Benefits of macros
- Background
- Some how-tos

Basic Definitions

- Macros are computer programs. But don't let the definition scare you!
- They are sets of instructions telling your computer what to do.
- The instructions can be short, simple sequences or long, complex ones.

Why Macros?

- Streamline your editorial processes (e.g., word lists, applying style codes).
- Save time and avoid burnout from repetitive steps (e.g., unembedding endnotes; changing 16 April 2019 → April 16, 2019).
- Customize your editorial software.

Examples of when macros can help

- A client wants your author queries in line and highlighted.
- You want a fast way to add words to a style sheet.
- You need to repeatedly capitalize or lowercase a term without tracking the change. Or, conversely, you need to show the change.
- You want to automatically mark where you stopped editing.
- You want a visual reminder of when track changes is turned off.

Background: VBA, the Language of MS Office Macros

- Macros in MS Office are written in a computer language called **VBA** (Visual Basic for Applications).
- VBA automatically runs in the background when you use a macro or create one.
- As you work more and more with macros, you'll automatically learn some basics of this programming language.

Three ways to add a macro

- Write a macro in VBA (like making a cake from scratch).
- Record your own clicks (a box cake).
- Use someone else's macro (a store-bought cake).

I don't make cakes from scratch—yet! Writing a full macro in VBA lies outside the scope of this presentation. For a comprehensive guide on writing macros from scratch, see Allen L. Wyatt, *Microsoft Word VBA Guidebook* (details at end).

Some How-Tos

Security: enabling a macro

- Because macros are real programs, your computer may want to protect itself from them.
- Adjust MS Word's security setting to allow macros.
- Go to **Developer** tab → **Code** area → **Macro Security**.
- Make sure "Disable All Macros with Notification" is checked (a good balance of security and permission).

Recording a macro

1. On **Developer** tab, click **Record Macro**.
2. Type in the Macro name you have chosen (see "Naming conventions," below).
3. Add a **description**. A month from now, you might forget what the macro does!
4. If you want your macro to be available in all your documents (usually you do), **store the macro** in "All documents (Normal.dotm)."
5. Choose **Button** or **Keyboard** assignment. (I prefer keyboard, as

mousing slows me down.)

6. If you clicked **Keyboard**, you'll get a dialog box that asks what keyboard shortcut you want to use. In the **Press new shortcut key** box, type the sequence you want to use. For example, you might try **Alt+Q** to add a special type of author query.
7. If the shortcut is already used for another function, you'll see "Currently assigned to: XXXX." If you want to keep this shortcut function XXX, then try a different shortcut key.
8. Once you have a good shortcut key, click **Assign**, then **Close**.
9. Now do whatever sequence of keystrokes or clicks you want to be recorded as a macro.
10. When you are finished, either click either the **little blue square** on your status bar at the bottom, or click **Stop Recording** on the Code section of the **Developer** tab.

Naming conventions

- Must begin with a letter
- Cannot have spaces
- The only symbol allowed is underscore
- Example names: lowercase_rev_show, lowercase_rev_hide, MsCleanup, AcmeAQ

Recording a macro: example

- Let's say your client wants author queries (AQs) in the text boldfaced and highlighted **<<AU: like this>>**.
- Place your cursor where you want an AQ in the manuscript, and start typing <<AU: >> Now select the <<AU: >> and make it boldface and yellow highlighted. As a last step in recording this macro, you'll probably also want to leave the cursor right after the colon, so that when you use the macro, your cursor will be exactly where you want to start typing your query. Now either click the **little blue square** on your status bar at the bottom or click **Stop Recording** on the Code section of the Developer tab. That's it!

Using a "premade" macro

1. From a document or elsewhere (online, for example), copy the text of a macro. Start at the line below the word "Sub" (stands for *subroutine*) to the line above "End Sub."
2. Go into any Word document.
3. Click **Alt+F8**. Alternatively, click **Developer** → Code → **Macros**.
4. In the **Macro Name** box, type the name you want to give your macro. Click **Create**.

5. Immediately under Sub [Whatever your macro name is], paste the text of the macro you copied.
6. Make sure there is only one “Sub” line and one “End Sub” line.
7. Click the right-hand **X** at the top of the window.
8. You’re done! To run the macro, click **Alt+F8** and type the name of the macro you want to run in the **Macro Name** box. Click **Run**.

“Premade” macro: example

- Let’s say you keep forgetting to turn track changes back on after you’ve turned them off for a minor correction.
- Paul Beverley’s **VisibleTrackOff4** macro turns the page yellow when you’ve turned off track changes and back to white when tracking is turned on.
- Copy the macro: download large file from Free Macros, and copy macro titled Sub VisibleTrackOff4(). Paste the macro into your own macros.
- You can add a keystroke to this macro to make it easier to run. For example, Ctr+Y would toggle the track changes off (and make page yellow) and back on (page becomes white again).

Assigning keystrokes to macros

1. Right-click anywhere in the ribbon.
2. Click **Customize the Ribbon**.
3. On the bottom left, next to **Keyboard shortcuts**, click **Customize**.
4. Scroll down under **Categories**, and click **Macros**.
5. Select the macro you want.
6. In the **Press new shortcut key**, type the keystrokes you want to use.
7. When you are happy with the keystroke you’ve selected, click **Assign** and then **Close**.

Resources

- Paul Beverley’s free macros, from his website Archive Publications: www.archivepub.co.uk/book.html (download the zip folder, which includes the huge book of macros themselves, plus another huge book of explanations).
- Beverley’s videos explaining macros step by step (YouTube). Here’s a good starter video: www.youtube.com/watch?v=hi4QCQy1QWg&t.
- Allen Wyatt’s newsletter WordTips has sections on macros: https://wordribbon.tips.net/WordTipsMacros.html.

- Allen L. Wyatt, *Microsoft Word VBA Guidebook*, 2nd ed. (Orem, UT: Sharon Parq Assoc., 2013).

See next page for some macros to copy.

Three Useful Macros for Copyediting

Try copying and pasting these macros into your set of macros. Copy the text between the green-highlighted lines, leaving the "bread" (the Sub and End Sub lines), and follow the directions at "Using a 'premade' macro" on page 3. The gray-highlighted lines are simply informational lines. They are not part of the program. The straight apostrophe symbol tells the software that the line is just info, not a command. You can add your own notes in these macros by simply preceding the line with the straight apostrophe.

Note that in the last two macros, I've used the filename "Practice manuscript." You need to substitute the filename of your own document for the macro to work. So if your file is titled Rennais_09_version2.doc then you need to replace "Practice manuscript" with "Rennais_09version2"

This macro saves file and leaves a bookmark "AAStoppedhere" where cursor is located. I assign the keystroke Ctrl+S to this macro, so that whenever I press Ctrl+S, it saves the file and adds "AAStoppedhere" wherever the cursor was:

```
Sub StoppedHere()
' stoppedhere Macro
' Inserts "AAStopped here" as a bookmark and saves file.
   With ActiveDocument.Bookmarks
      .Add Range:=Selection.Range, Name:="AAStoppedhere"
      .DefaultSorting = wdSortByName
      .ShowHidden = False
   End With
   ActiveDocument.Save
End Sub
```

Inserts selected term to bottom of style sheet:

```
Sub Style_Sheet()

' Style_Sheet Macro
' Adds selected term to style sheet and returns to original file.

    Selection.Copy
    Windows("Style Sheet").Activate
    Selection.EndKey Unit:=wdStory
    Selection.TypeParagraph
    Selection.Paste
    ' Replace "Practice manuscript" with "[whatever your main file's name
is]"
    Windows("Practice manuscript").Activate

End Sub
```

Inserts selected name to bottom of style sheet, and inverts name to Last, First (e.g., Brown, John):

```
Sub InvertNameToStyle()

' Inserts name in style sheet and inverts it.
' Style sheet filename must be "Style Sheet" (w/out quotes)
' Look at the next-to-last line and insert the
'     main file's filename instead of "Practice manuscript" between the
'     quote marks

    Dim fileName As String
    Dim pathName As String
    Dim o As Document
    Set o = ActiveDocument
    If InStrRev(o.Name, ".") <> 0 Then
        fileName = Left(o.Name, InStrRev(o.Name, ".") - 1)
    Else
        fileName = o.Name
    End If

Selection.Copy
    Windows("Style Sheet").Activate
    Selection.EndKey Unit:=wdStory
    Selection.TypeParagraph
    Selection.PasteAndFormat (wdPasteDefault)
    Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdExtend
    Selection.Cut
    Selection.MoveUp Unit:=wdLine, Count:=1
    Selection.EndKey Unit:=wdLine
    Selection.MoveRight Unit:=wdCharacter, Count:=1
    Selection.PasteAndFormat (wdFormatOriginalFormatting)
    Selection.MoveLeft Unit:=wdCharacter, Count:=1
    Selection.TypeText Text:=","
'   Replace "Practice manuscript" below with "[whatevery your main's
file name is]"
    Windows("Practice manuscript").Activate
End Sub
```