

# WHAT ARE MACROS AND WHAT THEY ARE USED FOR?



WYN HOPKINS  
SENIOR MANAGER  
Access Analytic  
whopkins@accessanalytic.com.au

## What are macros and what they are used for?

Good question, I typed this in to the Excel help menu and got this:

**Macros automate frequently-used tasks; many are created with VBA and are written by software developers.**

Let me expand on this, if you carry out a repetitive process in Excel you may be able to record the steps you take and then “play” them back the next time you need to perform that task. These recorded steps are the macro.

The programming language they are recorded in is (VBA) Visual Basic for Applications.

**Like any language you can learn a few words and phrases very quickly, but mastery comes with practice and time.**

## History

Macros have had a bad reputation.

In my view, this reputation has arisen due to the Macro Recorder allowing users with no programming experience or awareness the ability to create their own macros. I know these issues because I used to be one of those users, happily automating all my month end reconciliation and reporting processes, reducing the time it took to complete

my tasks by 50% - how wonderful. But, and it’s a big BUT, at no time did I consider making my macros robust, structured or documented, nor did any manager I ever worked for challenge me on this. However, simply changing a sheet name in my Excel file, or inserting a row in the wrong place would cause my macro to fail and either show the completely unhelpful “debug” button, or even worse, not even flag that something was wrong.

As they were my macros I knew how to fix them and what not change. I understood where not to insert or delete rows and columns, so there was no problem with the way I was doing things.

That is, until I left those companies and moved on to new roles leaving my replacement to cross their fingers and hope they picked up on the nuances of my macros. This is all too often how macros are developed and why then then break and cause so much frustration.

## A recorded macro

```
(General) ActualJnl
Sub ActualJnl()
' ABU Actual Journal Macro

Sheets("Actual Journal").Select
Columns("A:A").EntireColumn.AutoFit
Range("A4").Select
Range(Selection, Selection.End(xlDown)).Select
Selection.ClearContents
Sheets("Actual Payroll Charges").Select
Calculate
Range("Q2").Select
ActiveSheet.Range("$A$3:$P$37088").AutoFilter Field:=7, Criteria:=""
ActiveSheet.Range("$A$3:$P$37088").AutoFilter Field:=9, Criteria:=""<>
Columns("A:C").EntireColumn.AutoFit
Range("A4:A37385").Select
Selection.Copy
Sheets("ABU Actual J").Select
Range("A4").Select
Selection.PasteSpecial Destination:=Selection, Operation:=xlNone, SkipBlanks _
:=False, Transpose:=False
Range("A4").Select
Range(Selection, Selection.End(xlDown)).Select
Range(Selection, Selection.End(xlDown)).Select
ActiveSheet.Range("$A$4:$A$2000").RemoveDuplicates Columns:=1, Header:=xlNo
Range("E5").Select
Sheets("Actual Payroll Charges").Select
Columns("A:B").Select
Selection.EntireColumn.Hidden = True
ActiveSheet.Range("$A$3:$P$37088").AutoFilter Field:=9
```

Sheet and Ranges Hardcoded

No documentation or structure

I have come a long way from recording my macros and now develop Excel based tools and applications for clients. This is what I do differently: I write the code directly in modules, structure it efficiently and clearly, avoid hard coding any references, document it well and add error handling to deal with those completely unexpected glitches.

A large part of development time is spent on planning my approach and considering all the what-if scenarios to avoid any future changes to the files causing unintended results.

## A written macro

```
(General) CreateJournal
Option Explicit
Private Const cstrModule As String = "modJournals"

Sub CreateJournal()
'run through journal sheets setting required filters

Dim ValueFilterCol As Long
Dim tblName As String
Dim strJournalRange As String

Const cstrProcedure = "Create Journal"
On Error GoTo HandleError

With Application
    .Calculate
    .Calculation = xlCalculationManual
    .ScreenUpdating = False
End With

Set wksThis = ActiveSheet

'each journal sheet has a unique table name so code needs to identify which
'table we're referencing in order to apply the filters correctly

Select Case ActiveSheet.Name

    Case Sheet76.Name
        tblName = "tblJnlOUFA"

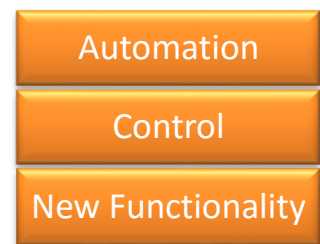
    Case Sheet84.Name
        tblName = "tblJnlOUFB"
```

Error Handling

Documented Code

Structured Layout

## What can you do with VBA?



I never realised until I started with Access Analytic just what was possible with VBA. It opens up Excel to a massive (limitless) range of features and functionality that assist with making processes more efficient or performing tasks that are not possible to do manually.

As a high level catch-all statement I'd say that if you can explain to a VBA developer what you want to happen then they can make Excel (or other application) do it.

**Automating** processes that would otherwise be a laborious manual series of repetitive actions is a very common use. Examples include:

- cleaning up messy system generated data that is in the wrong format for analysis in Excel
- generating reports for multiple departments or cost centres
- pulling information from multiple locations into one file
- generating emails containing files in PDF format and automatically distributing them to a distribution list
- amending the filename of a thousand PDF files to include today's date. So not even Excel related!

**Control** refers to the ability to restrict or prompt the user in what they can and should be doing.

Examples include:

- protection of files, restricting access to certain sheets or areas based on user passwords
- preventing users from using cut and paste or forcing the use of Paste Values
- guiding the user through the completion of a spreadsheet via message boxes and user forms
- automatically running data integrity checks prior to printing or saving a file
- automatically refresh pivot tables when selecting a sheet

- creating an audit trail of all changes made to a file.

The **new functionality** that VBA can provide is limited only by the developer's skills and experience.

I hesitate to even start a list of examples in this area but just to give some idea:

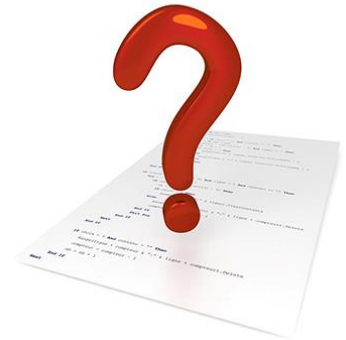
### From the small

- writing your own personalised custom functions (called User Defined Functions) .e.g. a JOIN function allowing you to highlight a column of email addresses and then in a single cell joins them together separated by a semi colon ";" – ideal for pasting into an email
- adding your own shortcuts to the right click menu
- reading and writing to files without opening them
- cleaning up corrupt cell styles (issues are now more common place now that more users have 2 screens to work between and copying between them also copies corrupt styles)
- closing a file automatically after a set period of time to prevent a user leaving a critical file open which prevents others getting access.

### To the large

- fully automated budgeting and reporting systems

- collating data from hundreds of cost centres
- stock control, recruitment, roster planning systems
- highly interactive dashboards and reporting systems with amazing graphical displays
- fully self-contained applications and add-ins that can audit your file, quickly create financial models or add advanced name management and analysis.



### Where now?

If you're happy recording macros and the results work for you then consider adding some key messages and instructions for future users of your file, normally consisting of "Don't do this, don't do that etc".

Your next step should be to read some of the literature on VBA and/or attend a course to understand how to take your macros to the next level and make them more robust for the next person in your role.

For those of you who've been "burnt" by macros in the past, don't write them off as voodoo magic that can't be trusted, get an experienced expert to write the code you need. Alternatively invest in the training required to understand the best practice approach to developing efficient robust solutions.

### About the author

Wyn gained his Chartered Accountant qualification at PricewaterhouseCoopers in the UK in 2000. He has extensive international business analyst experience, having worked with a variety of the UK's FTSE 100 financial services companies including Halifax Bank of Scotland (HBOS) and Barclays Bank. Wyn has specialist knowledge and industry experience in oil and gas, through financial services, telecommunications and agribusiness.

### Further information

#### Books, articles and demonstrations

<http://chandoo.org/wp/excel-vba/books/>

<http://www.accessanalytic.com.au/article/implementing-multiple-levels-of-access-in-a-spreadsheet.html>

[http://www.rondebruin.nl/windows\\_articles.htm](http://www.rondebruin.nl/windows_articles.htm)

<http://www.excelhero.com/>

<http://www.contextures.com/excelvbagettingstarted.html>

#### Training

<http://www.accessanalytic.com.au/course/excel-vba-training-course.html>

To learn more about VBA, Access Analytic run public courses and in-house training. Contact Access Analytic Senior Manager Wyn Hopkins for discuss your training needs on **+61 8 621 8500** or by [email](#).

|