Example VBA Macro

Background

Visual Basic for Applications (VBA) is the language Excel macros are written in. VBA is based on VB6, which was last updated in 1998. Still, for many folks in locked-down corporate environments, it remains the most accessible method of scripting tasks.

If you're looking to write a VBA macro, keep in mind that VBA is *very old* at this point, and that *good* examples may be hard to come by.

I started coding by tinkering with existing macros I used on the job. Not many years after that I was a full-time developer moving on to C# and SQL. Not everybody has the opportunities necessary for that, but it's a very valid path.

How To Use This Example

1. Open Excel
2. Start a new blank workbook
3. Add the Developer tab if you don't already have it
    1. Right click on the ribbon up top
    2. Select "Customize the Ribbon"
    3. In the right column, scroll down and check "Developer"
    4. Hit "OK"
4. Click the "Developer" > Visual Basic
5. Put the damned VBA window somewhere more useful than where it just popped up
    - In the future the VBA window can be opened by pressing Alt+F11
6. Click "This Workbook" under the tree for your file
7. Copy the below macro and paste it there
8. Copy the below table and paste it into your spreadsheet
9. Run the macro by either
    - Clicking the "play" button in the VBA window
    - Hitting "F5" in the VBA window
    - Adding a button to your spreadsheet and assigning this macro to it
        - "Developer" > Insert > Button > draw the button > select your macro > click "OK"
10. View the results
11. Think about how easy it would be to tweak this example to fit your business logic

Example Macro

```
Option Explicit

Sub ExampleScript()
    ' declare the variables we're going to use
    ' variable = bucket data goes in
    Dim mainSheet As Worksheet
    Dim rowNum As Long
    Dim id As String
    Dim name As String
    Dim zip As String
    Dim firstZipChar As String
    Dim tenFingies As Boolean
    Dim fingieString As String
    Dim outputOne As String
    Dim longName As String

    ' grab the first sheet from this workbook
    Set mainSheet = ThisWorkbook.Sheets(1)

    ' one in ten chance of an error
    ' this is just an example of a validation step you can take
    Randomize
    If Rnd() < 0.1 Then
        MsgBox "We're sorry, a duck has lodged in the flux capacitor." & vbCrLf & vbCrLf & "Please try aga
        Exit Sub
    End If
```

```
    ' clear previous output
    If mainSheet.UsedRange.Rows.Count >= 2 Then
        mainSheet.Range("E2", "F" & mainSheet.UsedRange.Rows.Count).ClearContents
    End If

    ' start at the row below the headers
    rowNum = 2

    ' loop while non-blank row data remains
    ' you can also use mainSheet.UsedRange to iterate rows, but this method is more reliable and less bugg
    Do While mainSheet.Range("A" & rowNum).Value <> vbNullString

        ' update the user on progress
        Application.StatusBar = "Doing shit: " & FormatPercent(rowNum / mainSheet.UsedRange.Rows.Count, 2)

        ' grab data
        id = mainSheet.Range("A" & rowNum).Value
        name = mainSheet.Range("B" & rowNum).Value
        zip = mainSheet.Range("C" & rowNum).Value
        fingieString = mainSheet.Range("D" & rowNum).Value

        ' format inputs
        id = Trim(id)
        name = Trim(name)
        zip = Trim(zip)
        fingieString = LCase(Trim(fingieString))

        If fingieString = "no" Or fingieString = "false" Then
            tenFingies = False
        Else
            tenFingies = True
        End If

        ' do some business logic
        firstZipChar = Left(zip, 1)

        If firstZipChar = "0" And tenFingies Then
            outputOne = "X"
        Else
            outputOne = ""
        End If

        If Len(name) > 13 Then
            longName = name
        Else
            longName = ""
        End If

        ' push the data to the spreadsheet
        mainSheet.Range("E" & rowNum).Value = outputOne
        mainSheet.Range("F" & rowNum).Value = longName

        ' iterate our row counter
        rowNum = rowNum + 1
    Loop

    ' reset the status bar
    Application.StatusBar = False

    ' dump the object reference we used before
    Set mainSheet = Nothing
End Sub
```

Corresponding Spreadsheet Table

ID	Name	Zip	Has > 10 Fingers	Output One	Output Two
220781234-0	Sleve McDichael	3096153	TRUE		
828417753-8	Onsen Sweemey	004423	FALSE		
172934172-1	Darryl Archideli	141707	yes		
573651857-0	Anatoli Smorin	21040 CEDEX	no		
052674485-5	Rey McSriff	16-120	x		
996602934-4	Glenallen Mixon	029000	x		
715035692-3	Mario McIlwain	02714	TRUE		
217007453-8	Raul Chamgelrian	63101	FALSE		
363790986-0	Kevin Nogily	asdf	FALSE		
880016640-4	Bobson Dugnutt	36009 CEDEX	no		
712333006-4	Willie Dustice	32213	yes		
945958918-6	Jeromy Gride	37416	yes		
346548127-5	Scott Dorque				
302236511-5	Shown Furcotte	235048	no		
531808967-3	Dean Wesery	02250	yes		
889639658-1	Mike Truk		FALSE		
416585373-9	Dwigt Rortugal	433 68	TRUE		
092887470-7	Tim Sandeale	422338	yes		
674373347-7	Karl Dandleton	01101	no		
417973345-X	Mike Hernandez	39290	no		
599343442-2	Todd Bonzalez	V6S	strongbad is on point		

Further Tips

- Remember Adam Savage's 2nd and 4th commandments for makers: "Make stuff that improves your life" and "Learn with a project."
  - Don't overwhelm yourself by learning stuff that won't be useful right away
  - Don't learn additional stuff because you feel obligated to; as you improve your life satisfaction and curiosity will drive you to learn more
- You can get limited help by placing your cursor on a keyword (like ".Range" or "vbNullString") and hitting F1

Useful VBA Settings

Found under in the VBA window under Tools > Options. I don't remember which of these are checked by default anymore

- check Editor > Auto Syntax Check, Auto List Members
- check General > Compile > Compile On Demand and Background Compile
- check General > Error Trapping > Break on Unhandled Errors
- check Docking > Locals Window and Immediate Window