



*“Practical, down-to-earth, tips,
tricks and advice for Excel users”*

How to make a cell flash in Excel and why I wouldn't do it!

www.excelsupersite.com

How to make a cell flash in Excel and why I wouldn't do it!

I was asked recently by a colleague of mine ***"is there a way in which we can make a cell flash in Excel to highlight some pertinent information?"*** The idea behind the question was that the data in the spreadsheet is entered and maintained by an employee and then provided to their boss to actually work with the information.

So... was there a way in which to highlight areas in the spreadsheet for the boss so that they did not have to scour the entire document to find the information they needed to look at?

To answer the question – yes, there is a way to make a cell background flash in Excel, however, as I'll discuss later, I would not apply this as a solution for them in this particular case.

How to make a cell flash in Excel

If we are trying to achieve this effect in Microsoft Word, it is actually very straight forward. Simply, select **Format | Font** from the menu, click the **Text Effects** tab, and choose **Blinking Background** – Viola done!! Nice and simple and very easy to achieve.

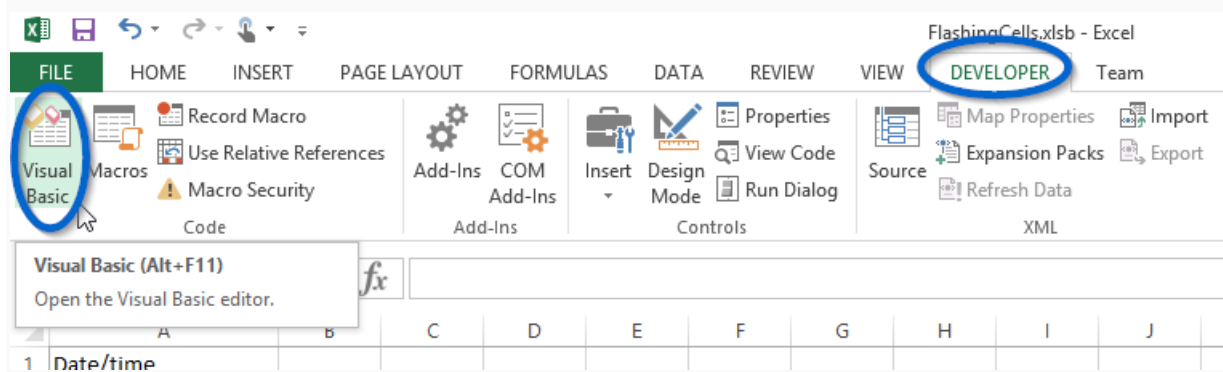
Now, you might expect that it would be just as simple to achieve the same in Microsoft Excel, but unfortunately (or fortunately as you'll see) that is not the case. It is, in fact, a **LOT** more difficult

and requires us to jump into the world of VBA (Visual Basic for Applications) and do a little macro programming.

So how do we actually do it...

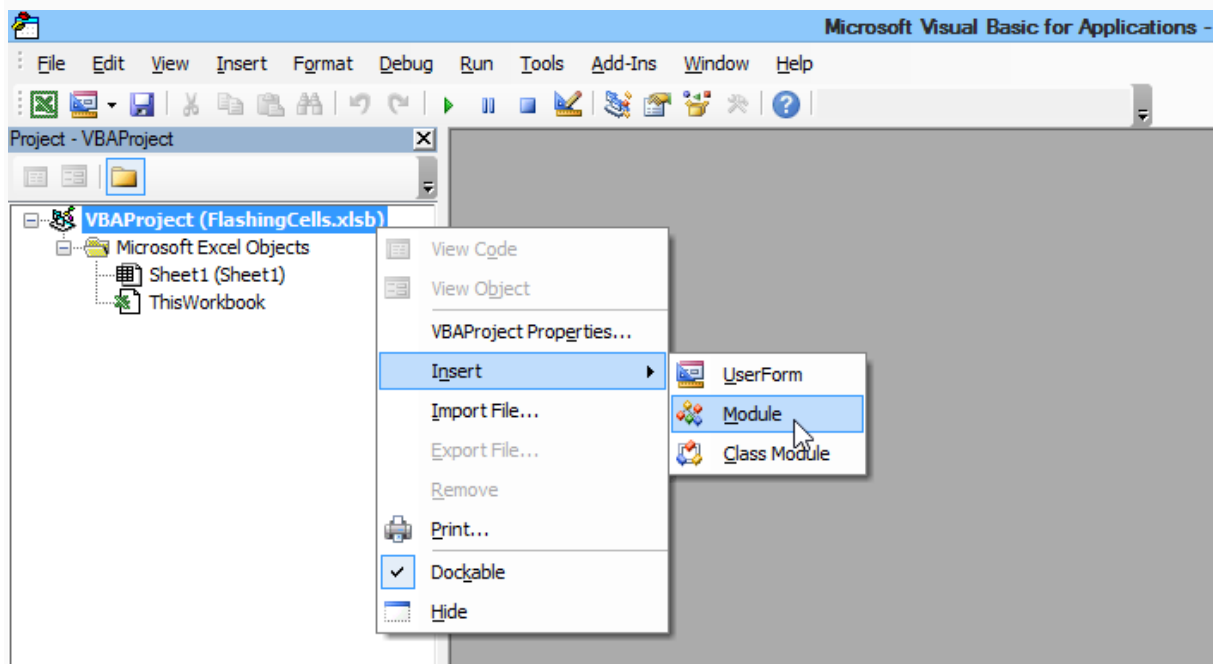
For our example, we are going to make the cell **A3** have a flashing background.

Step 1 On the **Developer Tab**, select the **Visual Basic Editor** from the menu.



Note: The **Developer Tab** is not shown by default in Excel, you need to add it in. If you do not have the Developer Tab showing, [follow this short tutorial](http://www.excelsupersite.com/how-to-show-the-developer-tab-in-excel/). (url: <http://www.excelsupersite.com/how-to-show-the-developer-tab-in-excel/>)

Step 2 Right-click the **VBA Project** item in the tree at left and choose **Insert | Module** from the pop-up menu.



Step 3 Type or copy and paste the following code into the module

```
Public NextFlash As Double
```

```
Public Const FlashRng As String = "Sheet1!A3"
```

```
Sub StartFlashing()
```

```
    If Range(FlashRng).Interior.ColorIndex = 3 Then
```

```
        Range(FlashRng).Interior.ColorIndex = xlColorIndexNone
```

```
    Else
```

```
        Range(FlashRng).Interior.ColorIndex = 3
```

```
    End If
```

```
    NextFlash = Now + TimeSerial(0, 0, 1)
```

```
    Application.OnTime NextFlash, "StartFlashing", , True
```

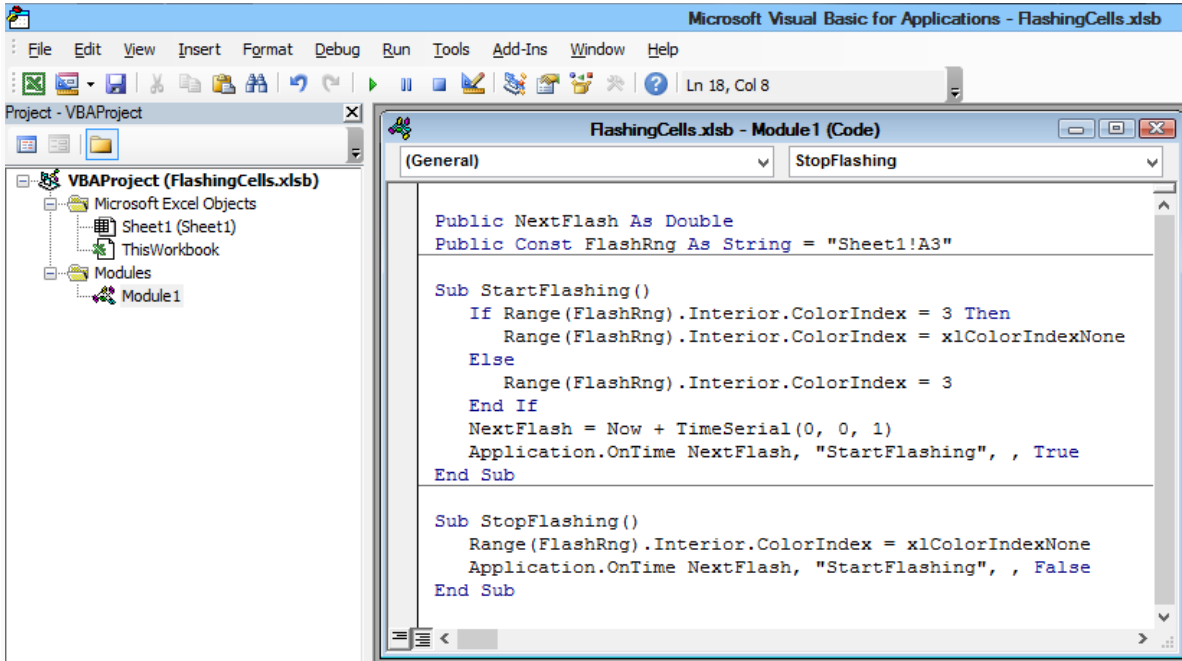
```
End Sub
```

```
Sub StopFlashing()
```

```
    Range(FlashRng).Interior.ColorIndex = xlColorIndexNone
```

```
    Application.OnTime NextFlash, "StartFlashing", , False
```

```
End Sub
```



```

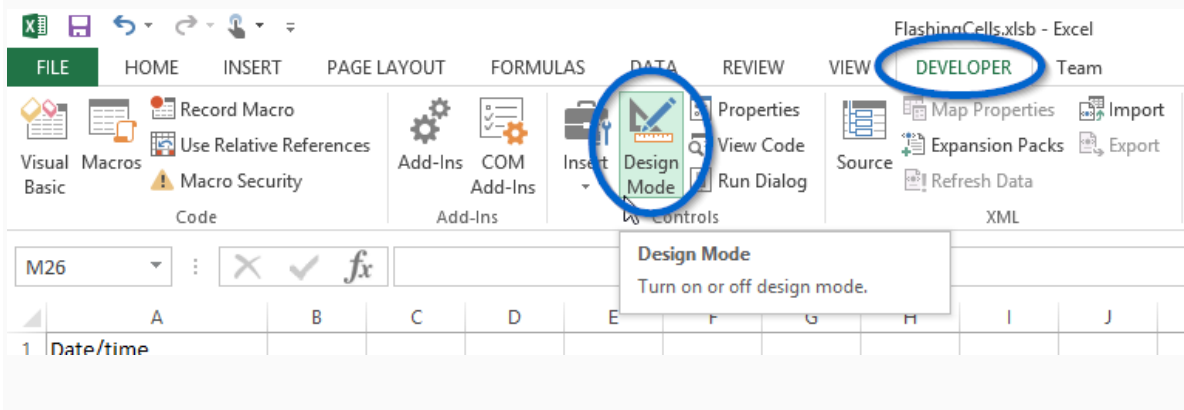
Microsoft Visual Basic for Applications - FlashingCells.xlsb
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 18, Col 8
Project - VBAProject
VBAProject (FlashingCells.xlsb)
  Microsoft Excel Objects
    Sheet1 (Sheet1)
  ThisWorkbook
  Modules
    Module1
FlashingCells.xlsb - Module1 (Code)
(General) StopFlashing
Public NextFlash As Double
Public Const FlashRng As String = "Sheet1!A3"

Sub StartFlashing ()
  If Range(FlashRng).Interior.ColorIndex = 3 Then
    Range(FlashRng).Interior.ColorIndex = xlColorIndexNone
  Else
    Range(FlashRng).Interior.ColorIndex = 3
  End If
  NextFlash = Now + TimeSerial(0, 0, 1)
  Application.OnTime NextFlash, "StartFlashing", , True
End Sub

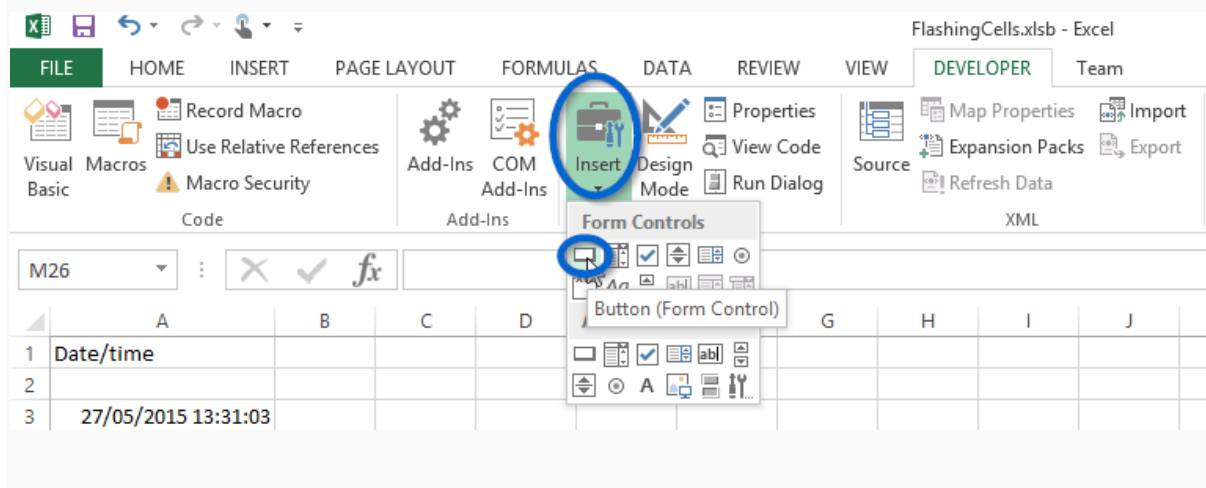
Sub StopFlashing()
  Range(FlashRng).Interior.ColorIndex = xlColorIndexNone
  Application.OnTime NextFlash, "StartFlashing", , False
End Sub

```

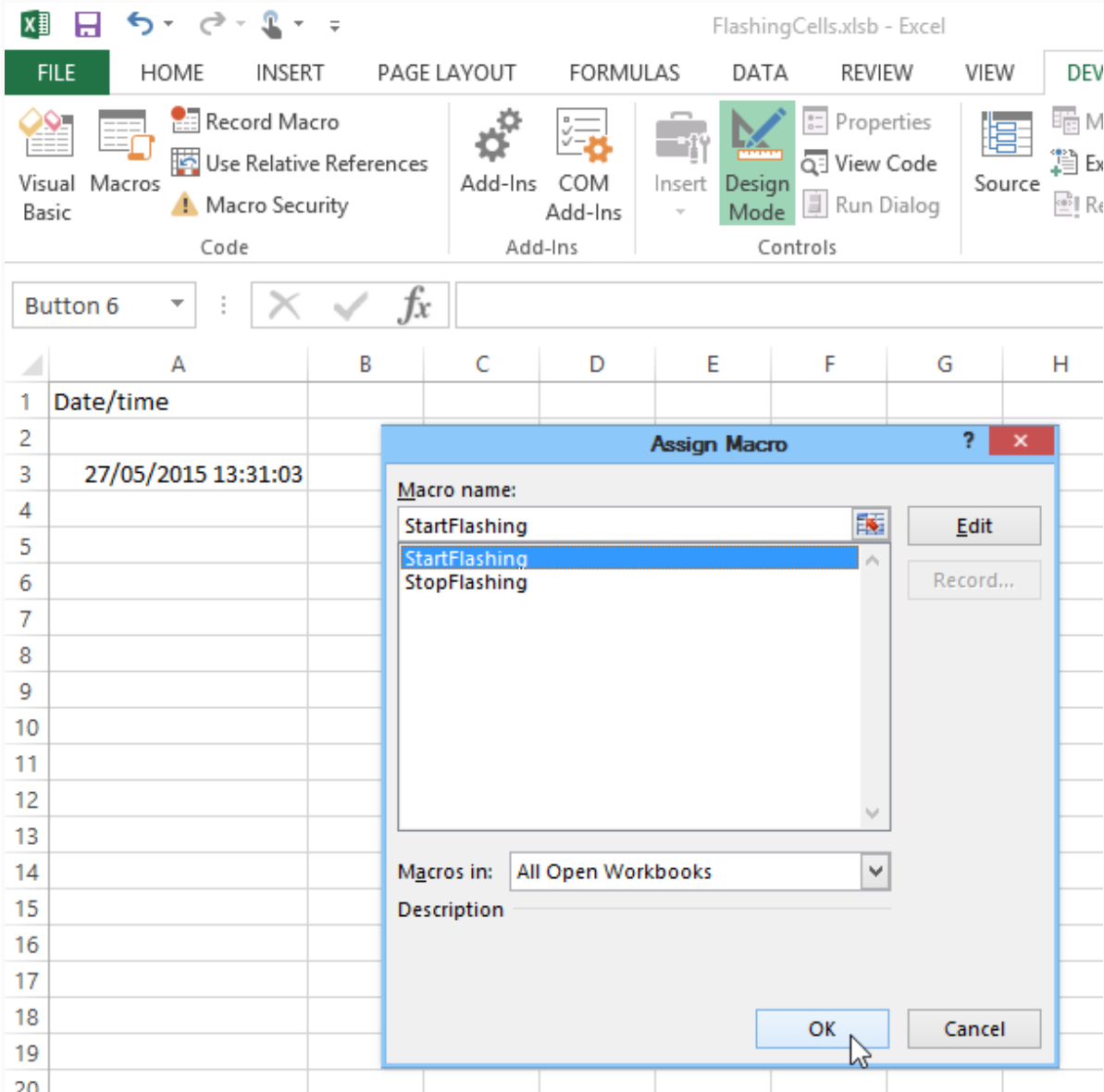
Step 4 We now need a way to turn the flashing on and off, so we'll add 2 buttons to our spreadsheet. In the Excel spreadsheet, click the **Developer Tab** and then make sure "Design Mode" is selected.



Step 5 Click **Insert** then click the **Button (Form Control)** option



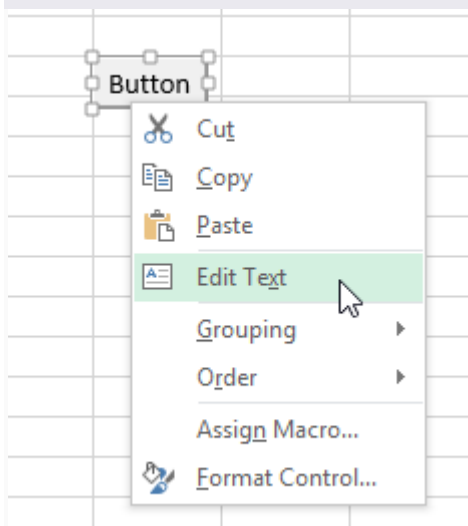
Step 6 Click on your spreadsheet (around cell C2, but it doesn't matter exactly where) to add the button. The **Assign Macro** dialog box will show. Select the "StartFlashing" macro and then click **OK**.



The screenshot shows the Microsoft Excel interface with the 'Assign Macro' dialog box open. The dialog box has a title bar that says 'Assign Macro'. Inside, there is a section labeled 'Macro name:' with a list box containing 'StartFlashing' and 'StopFlashing'. 'StartFlashing' is selected. To the right of the list box are 'Edit' and 'Record...' buttons. Below the list box is a dropdown menu labeled 'Macros in:' with 'All Open Workbooks' selected. At the bottom of the dialog box are 'OK' and 'Cancel' buttons. A mouse cursor is pointing at the 'OK' button. In the background, the Excel ribbon is visible with the 'DEVELOPER' tab selected, and the 'Design Mode' group is active. The spreadsheet shows a date in cell C3: '27/05/2015 13:31:03'.

Step 7 Right click on the button you just added and select **Edit Text**, from the pop-up menu, to give the button a more meaningful name. Edit the text to something like “*Start Flashing*”. When complete, click off the button to anywhere else on your spreadsheet to apply the changes.

Note: you can also modify the size of this button at this stage as well if you like.



Step 8 Repeat steps 5 through 7 to add a second button onto your spreadsheet, but this time in step 6, assign the “*StopFlashing*” macro to the button and in step 7, rename your button to something like “*Stop Flashing*”.

Ok, we’re done and don’t forget to save your workbook.

Click the “*Start Flashing*” button and the background of cell A3 should now start to flash! Click the “*Stop Flashing*” and the flashing should stop.

How does it work?

So how does it work? First, the constant *FlashRng* defines the range of cell(s) that will flash. You can make different cells flash by changing this value. If the flash range has a red background, this macro sets it back to normal; if not, it makes the background red. Then it sets itself up to be called again in one second using the application’s *OnTime* method.

Next, we check the background colour of our range of cells. If the flash range has a red background (i.e. *ColorIndex=3*), then the macro changes it back to normal; if the background is not red, the code then makes the cell background red.

The *TimeSerial* function returns a numeric time-value corresponding to the number of hours, minutes, and seconds passed to it as input. *TimeSerial* takes only whole numbers, so the shortest time period it can calculate is one second which we use here.

We set the *NextFlash* variable to the current time (NOW) plus one second (*TimeSerial(0,0,1)*), and we call the application object’s *OnTime* method to launch *StartFlashing* again at that time (i.e. every second in our case).

Each time the macro is called the cell background of our *FlashRng* is changed from red to normal or vice versa.

The *StopFlashing* macro simply restores the normal background and calls *OnTime* to cancel the pending event. To cancel an

event, you have to pass the exact same time that was used to schedule it, which is why we had to store that time in the public variable available to both macros.

That’s a lot of work just to get flashing text, but it does the job. And of course, you can apply it to any range you like by changing the value of the *FlashRng* constant.

So why wouldn't I use this as a solution??

So you are probably wondering now *“Why after all that work would we not use this as a solution?”*

My reasoning is actually quite straight forward.

In order for Excel to make a cell flash, the spreadsheet **MUST** constantly run your code (as I explained above, our macro is called every second, in order to “flash” the cell background) so it can keep checking your criteria to determine whether or not to apply any formatting. This may not seem a big deal at first, particularly if you are only dealing with small spreadsheets. However, constantly have code running in your spreadsheet will add **VERY SIGNIFICANT** load onto the processing of it and if you work with large and/or complex spreadsheets that undertake lots of calculations, putting unnecessary load on them is a **BAD thing** and something you want to avoid wherever possible.

So there you have it, yes we can make a cell's background in Excel flash, but I generally wouldn't recommend actually applying this to any of your spreadsheets.

It comes down to... Just because you can, doesn't mean you should!

Please Share

If you liked this article or know someone who could benefit from this information, please feel free to share it with your friends and colleagues and spread the word on Facebook, Twitter or LinkedIn.

Download

You can also download a copy of the spreadsheet I used in this article so you can explore this tip further – [How to make a cell flash in Excel and why I wouldn't do it!](#).