

The Excel VBA Programming Tutorial for Beginners

By Brad Jones

Read the original article here: http://www.makeuseof.com/tag/excel-vba-programming-tutorial/

Image: The search art. Chart Parent. Name: "Exponential Nor particus. News, is reading to the search art. Name: "Exponential Nor particus. News, is reading to the search art. Name: "Exponential Nor particus. News, is reading to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the search art. Name: "Exponential Nor particus. News, is reader to the searc. Name: "Exponential Nor particus. Name: "Ex
<pre>the eachart.chart Parent.Hame = "Exponential No ving a ving (1 ≤ v ≤ k) SeriesCollection.NewSeriesGilection.Versets("data").Range("e2:e" & nowCourt: Format.Line.Weight = 2 Name = "Price"</pre> the instruction is the series of
Parent. Hame The boxe Not Parent bit Not Not <th< th=""></th<>
And the series conjection newseries sheets ("data"). Range ("e2:e" & Powcount") A = F G H I J K L Monor (Control Found (Contr
Series Collection. New Series Collection. New Series Collection Series Collection. New Seri
<pre>Avalues = Sheets("data").Range("e2:e" & rowCount) Name = "Price"</pre>
Walkes = Skets("data"). Range("na. & stochasticoscillaton" .Walkes = Skets("data"). Range("e2:e" & rowcourt: .Pormat.Line.Weight = 2 .Name = "Price" .Name = "Price"
.Format.Line.Weight = 2 .Range("e2:e" & rowcount) .Range("e2:e" & rowcount) .Name = "Price" .Range("a2:a" & rowcount) .Range("in & stochasticoscillator
Name = "Price" & rowCount: End With a stochasticoscillator Name = "Price" & rowCount: Second (R) (Range("n. & sillator (Range("n. & s
End where "Price" = 2 4 of azia" & "modify" and the stochasticosci a
Bange("i1") = "Lower ("in") =
Chartfyne ", & stociate and a la storperiod - 1 & jul 1 & ".i" & r
AxisGroup = Alline
.Values = Sheets("data").Range("h2:h" & rowCour & stochastiCoscillatorPeriod - 1 & "]C[-5];KC[-3]/
.Name = "EMA" 13
Border.ColorIndex = 3 is
.Format.Line.Weight = 2
End with $Pic(-3) = R[0](c(-3) - Reaction + 1) = Pic(-3) + R[0](c(-3) - Reaction + 1) = Pic(-3) + R[0](c(-3) - Reaction + 1) = Pic(-3) + R[0](c(-3) + 1) + Pic(-3) + $
Axes(xlValue, xlPrimer), Axistice: Workset (Morket and Control and Artice and
Axes(xlvalue, xlPrime') Maximuscale = "
aves(xlvalue, xiriary). ManfaironBouchartice vo
Aver(xiValue, xir, xilegenarous, above provide and a second secon
Promise ("k" & Stoch(" & smoothed and the smoothed and th
symbol (see the see the see the second secon
Title Text 2200 Text 200 + Smallerind
(nartis)
End With

If you're using Excel, you have to try this power tool!

Visual Basic for Applications (VBA) is the Microsoft Office programming language that allows you to create macros and userforms, add a message box, execute code inside a document in response to a trigger, and much more. With VBA you can supercharge your Excel spreadsheets. And you just have to learn a little bit about coding.

This guide will help you try your hand at VBA with a simple project: a button that converts the value of a chosen cell from GBP to USD. We will introduce you to the ways that VBA and Excel can intersect. This short tutorial will put you on a path toward creating your own more complex projects.

Here's how to get started with VBA in Excel 2016 (CA/UK).



Access Developer Controls

Before we can dive into VBA, it might be necessary to open Excel and adjust the settings to display the **Developer** tab as part of the Ribbon. To do so, head to **File > Options > Customize Ribbon**.

Excel Options		?	×
General Formulas	General options for working with Excel.		
Proofing	User Interface options		
Save Language Advanced Customize Ribbon	 ✓ Show Mini Toolbar on selection ✓ Show Quick Analysis options on selection ✓ Enable Live Preview ScreenTip style: Show feature descriptions in ScreenTips ▼ 		
Quick Access Toolbar	When creating new workbooks		
Add-ins Trust Center	Use this as the default font: Body Font Font size: Default view for new sheets: Normal View Include this many sheets: 1		
	Personalize your copy of Microsoft Office		
	User name: bradjonze@googlemail.com Always use these values regardless of sign in to Office.		

Create a Button

To create our currency converter, we first need to insert the button element. In a second step, we'll attach our VBA code to that button.

Open a new Excel spreadsheet, then navigate to the **Developer** tab. Use the **Insert** dropdown in the **Controls** section to select an **ActiveX Command Button**.



Copyright © 2016 MakeUseOf. All Rights Reserved ®.



Drag the button out to an appropriate size and place it somewhere convenient – you can easily change this later on.

=EMBED("Forms.Con	nmandButto	on.1","")					
D	E	F	G	Н	T	J	к	L
					Comn	onandButton:	1	

Now we'll attach the code. Right-click the button and select **Properties**. We'll make two changes; we're going to change the **Name** that we'll use to refer to the button while coding, and the **Caption** that displays text on the button itself. You can choose whatever you like for these labels, but remember that you'll need to swap out ConverterButton for whatever you use in its place while we're adjusting the code.

Properties		×	
ConverterBut	ton CommandButton	•	
Alphabetic Ca	tegorized		
(Name)	ConverterButton		
Accelerator		1	
AutoLoad	False		
AutoSize	False		
BackColor	8H8000000F&		
BackStyle	1 - fmBackStyleOpaque		
Caption	Convert		
Enabled	True		
Font	Calibri		

Now it's time to give the button some functionality.



Add Some Code

Coding with VBA takes place in a separate environment to the standard Excel interface. To access it, make sure **Design Mode** is active in the **Developer** tab, then right-click the button we created and select **View Code**.

Data	Rev	view	View	Devel	oper	Add-ins	Team	Format	♀ Tell n
)M I-ins	Insert	Design Mode Con	E Prop S View Run	erties Code Dialog	Source	Map Pr Expans Refresh	roperties ion Packs 1 Data XMI	Export	

.CommandButton.1","")

F	G	Н	- I	J	К	L
		0	с	0		9
		0	_	Convert		4
		Ĭ	*	Cut		
		0	Ē	<u>C</u> opy		
				Paste		
				Propert <u>i</u> es		_
			Q.	<u>V</u> iew Code		
				CommandBu	tton <u>O</u> bject	F

You'll see a window like the one below:

ConverterButton	Click	-
Private Sub Conver	terButton_Click()	
End Sub		

Copyright © 2016 MakeUseOf. All Rights Reserved ®.



The start and the end of our code is already in place — the two blue pieces of text bookend our function, while the text in black states that we're stipulating the action that should take place when the user **clicks on the button** we created. If you chose a different name to ConverterButton, you should see the respective term in your version of this window.

To carry out the currency conversion procedure, we'll use the following line of code between the two that have already been created for us:

```
ActiveCell.Value = (ActiveCell * 1.28)
```

To break it down further, this piece of code states that the new value of the cell that the user has selected will be the current value multiplied by 1.28 — the exchange rate from GBP to USD. Here's how that looks in the VBA window:

Book2 - Sheet1 (Code)	Click	
Private Sub Conver ActiveCell.Value = End Sub	terButton_Click() (ActiveCell * 1.28)	
≡≡		

Next, close the VBA editor window and head back to Excel.



Test Your Work

It's now time to see whether our code works — but there's an important step to take before we can do that. We need to disable **Design Mode** to stop any further modifications to the button, and make it functional.



Next, enter a number into a cell, select that cell, and click your button to see it work its magic. Hopefully, you'll see the value increase by around a quarter, meaning that the conversion has been carried out correctly.

Next Steps

Now that you've created a button and used it to execute VBA code in Excel, you can use the same basic method to carry out all kinds of different projects. You might want to create a simulated die that returns a random value when you press the button, perhaps as **part of a larger game**. Alternatively, you could create a button that checks the contents of a specified cell against another one elsewhere in the same document.

Embarking on projects like these exposes you to the different features of VBA. Our currency converter is about as simple as it gets – but it's the first step towards bigger things. If you're just starting out, pick projects that relate your learning to a basic goal or task you're curious about. Step by step, you will become more familiar with how VBA works.

Do you have a question about this VBA project? Ask for assistance or offer up some help in the comments!

Read more stories like this at

