

LEARNING MADE EASY



6th Edition

# Excel<sup>®</sup> VBA Programming

for  
**dummies**<sup>®</sup>  
A Wiley Brand



Work faster and smarter  
by automating with VBA

Create applications and  
be the office hero

Learn VBA to get the  
most out of Excel

**Dick Kusleika**





# Excel<sup>®</sup> VBA Programming

6th Edition

**by Dick Kusleika**

**for  
dummies<sup>®</sup>**  
A Wiley Brand

## Excel® VBA Programming For Dummies®

Published by: **John Wiley & Sons, Inc.**, 111 River Street, Hoboken, NJ 07030-5774, [www.wiley.com](http://www.wiley.com)

Copyright © 2022 by John Wiley & Sons, Inc., Hoboken, New Jersey

Media and software compilation copyright © 2012 by John Wiley & Sons, Inc. All rights reserved.

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

**Trademarks:** Wiley, For Dummies, the Dummies Man logo, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and may not be used without written permission. Microsoft and Excel are trademarks or registered trademarks of Microsoft Corporation in the United States and other countries. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: WHILE THE PUBLISHER AND AUTHORS HAVE USED THEIR BEST EFFORTS IN PREPARING THIS WORK, THEY MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES REPRESENTATIVES, WRITTEN SALES MATERIALS OR PROMOTIONAL STATEMENTS FOR THIS WORK. THE FACT THAT AN ORGANIZATION, WEBSITE, OR PRODUCT IS REFERRED TO IN THIS WORK AS A CITATION AND/OR POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE PUBLISHER AND AUTHORS ENDORSE THE INFORMATION OR SERVICES THE ORGANIZATION, WEBSITE, OR PRODUCT MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING PROFESSIONAL SERVICES. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR YOUR SITUATION. YOU SHOULD CONSULT WITH A SPECIALIST WHERE APPROPRIATE. FURTHER, READERS SHOULD BE AWARE THAT WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ. NEITHER THE PUBLISHER NOR AUTHORS SHALL BE LIABLE FOR ANY LOSS OF PROFIT OR ANY OTHER COMMERCIAL DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR OTHER DAMAGES.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit <https://hub.wiley.com/community/support/dummies>.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit [www.wiley.com](http://www.wiley.com).

Library of Congress Control Number: 2021951691

ISBN 978-1-119-84307-8 (pbk); ISBN 978-1-119-84308-5 (ebk); ISBN 978-1-119-84309-2 (ebk)

# Contents at a Glance

<b>Introduction</b>	1
<b>Part 1: Starting Excel VBA Programming</b>	7
CHAPTER 1: Getting to Know VBA	9
CHAPTER 2: Building Simple Macros	17
<b>Part 2: Employing VBA with Excel</b>	29
CHAPTER 3: Working in the Visual Basic Editor	31
CHAPTER 4: Introducing the Excel Object Model	51
CHAPTER 5: VBA Sub and Function Procedures	67
CHAPTER 6: Using the Excel Macro Recorder	83
<b>Part 3: Programming Concepts</b>	97
CHAPTER 7: Essential VBA Language Elements	99
CHAPTER 8: Working with Range Objects	119
CHAPTER 9: Using VBA and Worksheet Functions	135
CHAPTER 10: Controlling Program Flow and Making Decisions	151
CHAPTER 11: Automatic Procedures and Events	171
CHAPTER 12: Error-Handling Techniques	193
CHAPTER 13: Bug Extermination Techniques	205
CHAPTER 14: VBA Programming Examples	219
<b>Part 4: Communicating with Your Users</b>	243
CHAPTER 15: Simple Dialog Boxes	245
CHAPTER 16: UserForm Basics	263
CHAPTER 17: Using UserForm Controls	281
CHAPTER 18: UserForm Techniques and Tricks	301
CHAPTER 19: Accessing Your Macros through the User Interface	329
<b>Part 5: Putting It All Together</b>	343
CHAPTER 20: Creating Worksheet Functions	345
CHAPTER 21: Creating Excel Add-Ins	365
<b>Part 6: The Part of Tens</b>	377
CHAPTER 22: Ten Handy Visual Basic Editor Tips	379
CHAPTER 23: Resources for VBA Help	389
CHAPTER 24: Ten VBA Do's and Don'ts	395
<b>Index</b>	401



# Table of Contents

---

<b>INTRODUCTION</b>	1
About This Book	1
Typographical conventions	2
Macro security	3
Foolish Assumptions	5
Icons Used in This Book	5
Beyond the Book	6
Where to Go from Here	6
 <b>PART 1: STARTING EXCEL VBA PROGRAMMING</b>	 7
 <b>CHAPTER 1: Getting to Know VBA</b>	 9
Understanding VBA Basics	9
Knowing What VBA Can Do	10
Inserting a bunch of text	11
Automating a task you perform frequently	11
Automating repetitive operations	11
Creating a custom command	11
Creating a custom button	12
Developing new worksheet functions	12
Creating custom add-ins for Excel	12
Getting the Most from VBA	12
Knowing what VBA does best	12
Recognizing the disadvantages of using VBA	13
Understanding VBA Concepts	14
Ensuring Excel Compatibility	16
 <b>CHAPTER 2: Building Simple Macros</b>	 17
Displaying the Developer Tab	17
Creating a Macro	18
Preparing the Environment	18
Recording a Macro	19
Running the Macro	21
Viewing a Macro in the Visual Basic Editor	22
Modifying the Macro	24
Saving Workbooks That Contain Macros	25
Understanding Macro Security	25

<b>PART 2: EMPLOYING VBA WITH EXCEL</b> .....	29
<b>CHAPTER 3: Working in the Visual Basic Editor</b> .....	31
Getting to Know the Visual Basic Editor.....	31
Activating the VBE .....	32
Exploring VBE components .....	32
Working with the Project Explorer .....	34
Adding a new VBA module.....	35
Working with a Code Pane.....	36
Minimizing and maximizing windows.....	36
Looking at the parts of a module .....	37
Getting VBA code into a module .....	38
Entering code directly.....	38
Using the macro recorder .....	41
Copying VBA code .....	44
Customizing the VBE.....	44
Using the Editor tab .....	44
Using the Editor Format tab.....	47
Using the General tab.....	48
Using the Docking tab.....	48
<b>CHAPTER 4: Introducing the Excel Object Model</b> .....	51
Working with the Excel Object Model.....	51
Understanding the object hierarchy.....	52
Referring to objects.....	53
Diving into Object Properties and Methods .....	57
Setting object properties .....	58
Taking action with object methods.....	60
Triggering actions with object events.....	61
Finding Out More from VBA Resources .....	62
Using VBA's Help system .....	62
Using the Object Browser.....	63
Automatically listing properties and methods .....	64
<b>CHAPTER 5: VBA Sub and Function Procedures</b> .....	67
Understanding Subs versus Functions.....	67
Looking at Sub procedures .....	68
Looking at Function procedures .....	68
Naming Subs and Functions .....	69
Executing Sub procedures.....	70
Executing the Sub procedure directly .....	72
Executing the procedure from the Macro dialog box .....	73
Executing a macro by using a shortcut key .....	74
Executing the procedure from a button or shape .....	75
Executing the procedure from another procedure .....	77
Executing the procedure from the Immediate window.....	77



Executing Function Procedures . . . . .	78
Calling the function from a Sub procedure . . . . .	78
Calling a function from the Immediate window . . . . .	79
Calling a function from a worksheet formula . . . . .	80
<b>CHAPTER 6: Using the Excel Macro Recorder . . . . .</b>	<b>83</b>
Recording Basics . . . . .	84
Preparing to Record . . . . .	85
Choosing Between Relative and Absolute Mode . . . . .	86
Recording in absolute mode . . . . .	86
Recording in relative mode . . . . .	87
Watching the Macro Recorder in Action . . . . .	89
Specifying Recording Options for Your Macro . . . . .	91
Streamlining Code Generated by the Macro Recorder . . . . .	92
 <b>PART 3: PROGRAMMING CONCEPTS . . . . .</b>	 <b>97</b>
<b>CHAPTER 7: Essential VBA Language Elements . . . . .</b>	<b>99</b>
Using Comments in Your VBA Code . . . . .	99
Using Variables, Constants, and Data Types . . . . .	101
Understanding variables . . . . .	101
What are VBA's data types? . . . . .	103
Declaring and scoping variables . . . . .	103
Working with constants . . . . .	110
Using premade constants . . . . .	110
Working with strings . . . . .	111
Working with dates . . . . .	112
Using Assignment Statements . . . . .	113
Assignment statement examples . . . . .	113
About that equal sign . . . . .	114
Smooth operators . . . . .	114
Working with Arrays . . . . .	116
Declaring arrays . . . . .	116
Multidimensional arrays . . . . .	117
Dynamic arrays . . . . .	117
Using Labels . . . . .	118
 <b>CHAPTER 8: Working with Range Objects . . . . .</b>	 <b>119</b>
Referring to Range Objects . . . . .	119
Referring to a Range Using Properties . . . . .	121
The Cells property . . . . .	122
The Offset property . . . . .	122
The Resize property . . . . .	123

Working with Range Object Properties .....	124
The Value property .....	124
The Text property .....	125
The Count property .....	126
The Column and Row properties .....	126
The Address property .....	126
The HasFormula property .....	127
The Font property .....	128
The Interior property .....	128
The Formula property .....	130
The NumberFormat property .....	131
Taking Action with Range Object Methods .....	131
The Select method .....	132
The Copy and Paste methods .....	132
The Clear method .....	133
The Delete method .....	133
<b>CHAPTER 9: Using VBA and Worksheet Functions .....</b>	<b>135</b>
Understanding Functions .....	135
Using Built-In VBA Functions .....	136
Working with dates and times .....	136
Manipulating strings .....	138
Identifying objects and data .....	139
VBA functions that do more than return a value .....	140
Discovering VBA functions .....	141
Using Worksheet Functions in VBA .....	144
Worksheet function examples .....	144
Entering worksheet functions .....	147
More about using worksheet functions .....	148
Using Custom Functions .....	148
<b>CHAPTER 10: Controlling Program Flow and Making Decisions .....</b>	<b>151</b>
Going with the Flow, Dude .....	151
The GoTo Statement .....	152
Decisions, Decisions .....	154
The If-Then structure .....	154
The Select Case structure .....	158
Knocking Your Code for a Loop .....	162
For-Next loops .....	162
Do-While loop .....	167
Do-Until loop .....	168
Using For Each-Next Loops with Collections .....	168

<b>CHAPTER 11: Automatic Procedures and Events</b>	171
Preparing for the Big Event	171
Learning when to use event procedures	173
Programming event-handler procedures	174
Knowing Where to Put the Event Code	174
Writing an Event-Handler Procedure	175
Triggering Workbook Events	177
The Open event for a workbook	177
The BeforeClose event for a workbook	179
The BeforeSave event for a workbook	180
Using Activation Events	181
Activate and deactivate events in a sheet	181
Activate and deactivate events in a workbook	182
Workbook activation events	184
Programming Worksheet-Related Events	185
The BeforeDoubleClick event	185
The BeforeRightClick event	185
The Change event	186
Understanding Events Not Associated with Objects	188
The OnTime event	188
Keypress events	190
<b>CHAPTER 12: Error-Handling Techniques</b>	193
Types of Errors	193
An Erroneous Macro Example	194
Not-quite-perfect macros	195
Perfecting the macro	196
Giving up on perfection	198
Alternate Ways of Handling Errors	198
Revisiting the EnterSquareRoot procedure	198
Trapping errors with the On Error statement	200
Handling Errors: The Details	200
Resuming after an error	200
Error handling in a nutshell	202
An Intentional Error	203
<b>CHAPTER 13: Bug Extermination Techniques</b>	205
Species of Bugs	205
Identifying Bugs	207
Debugging Techniques	208
Examining your code	208
Using the MsgBox function	208
Inserting Debug.Print statements	210
Using the VBA debugger	210

Using the Debugger's Tools .....	211
Setting breakpoints in your code .....	211
Using the Watches window .....	214
Using the Locals window .....	216
Bug Reduction Tips .....	217
<b>CHAPTER 14: VBA Programming Examples .....</b>	<b>219</b>
Working with Ranges .....	220
Copying a range .....	220
Copying a variable-size range .....	221
Selecting to the end of a row or column .....	223
Selecting a row or column .....	224
Moving a range .....	224
Looping through a range efficiently .....	225
Looping through a range efficiently (Part II) .....	227
Prompting for a cell value .....	227
Determining the selection type .....	228
Identifying a multiple selection .....	229
Changing Excel Settings .....	230
Changing Boolean settings .....	230
Changing non-Boolean settings .....	231
Working with Charts .....	231
AddChart versus AddChart2 .....	232
Modifying the chart type .....	234
Looping through the ChartObjects collection .....	234
Modifying chart properties .....	235
Applying chart formatting .....	235
VBA Speed Tips .....	237
Turning off screen updating .....	237
Turning off automatic calculation .....	238
Eliminating those pesky alert messages .....	239
Simplifying object references .....	240
Declaring variable types .....	241
Using the With-End With structure .....	242
<b>PART 4: COMMUNICATING WITH YOUR USERS .....</b>	<b>243</b>
<b>CHAPTER 15: Simple Dialog Boxes .....</b>	<b>245</b>
Interacting with the User in VBA .....	246
Displaying Messages with the MsgBox Function .....	247
Displaying a simple message box .....	247
Getting a response from a message box .....	248
Customizing message boxes .....	250
Getting Data with an Input Box .....	252

Understanding the InputBox syntax. . . . .	253
Using the InputBox function . . . . .	253
Using the InputBox method. . . . .	255
Allowing the User to Select a File or Folder . . . . .	256
Constructing a GetOpenFilename statement . . . . .	256
Selecting a file with GetOpenFilename. . . . .	256
Picking a file with GetSaveAsFilename . . . . .	259
Getting a folder name. . . . .	259
Displaying Excel's Built-In Dialog Boxes . . . . .	260
<b>CHAPTER 16: UserForm Basics . . . . .</b>	<b>263</b>
Knowing When to Use a UserForm. . . . .	263
Creating UserForms: An Overview . . . . .	265
Working with UserForms . . . . .	266
Inserting a new UserForm . . . . .	266
Adding controls to a UserForm . . . . .	267
Changing properties for a UserForm control . . . . .	268
Viewing the UserForm Code pane . . . . .	269
Displaying a UserForm . . . . .	269
Using information from a UserForm . . . . .	270
A UserForm Example . . . . .	271
Creating the UserForm . . . . .	271
Adding the CommandButtons. . . . .	272
Adding the OptionButtons . . . . .	272
Adding event-handler procedures . . . . .	275
Creating a macro to display the dialog box. . . . .	277
Making the macro available. . . . .	277
Testing the macro . . . . .	278
<b>CHAPTER 17: Using UserForm Controls . . . . .</b>	<b>281</b>
Getting Started with Dialog Box Controls . . . . .	281
Adding controls . . . . .	282
Introducing control properties . . . . .	283
Learning Dialog Box Controls Details . . . . .	285
CheckBox control. . . . .	285
ComboBox control . . . . .	286
CommandButton control . . . . .	287
Frame control. . . . .	288
Image control. . . . .	288
Label control. . . . .	289
ListBox control. . . . .	289
MultiPage control . . . . .	291
OptionButton control . . . . .	292
RefEdit control . . . . .	292

ScrollBar control .....	293
SpinButton control .....	294
TabStrip control .....	295
TextBox control .....	295
ToggleButton control .....	296
Working with Dialog Box Controls .....	296
Moving and resizing controls .....	297
Aligning and spacing controls .....	297
Accommodating keyboard users .....	297
Testing a UserForm .....	300
Dialog Box Aesthetics .....	300
<b>CHAPTER 18: UserForm Techniques and Tricks .....</b>	<b>301</b>
Using Dialog Boxes .....	302
A UserForm Example .....	302
Creating the dialog box .....	302
Writing code to display the dialog box .....	305
Making the macro available .....	305
Trying out your dialog box .....	306
Adding event-handler procedures .....	307
Validating the data .....	309
Now the dialog box works .....	309
A ListBox Control Example .....	309
Filling a ListBox Control .....	310
Determining the selected item .....	312
Determining multiple selections .....	313
Selecting a Range .....	314
Using Multiple Sets of Option Buttons .....	316
Using a Spin Button and a Text Box .....	317
Using a UserForm as a Progress Indicator .....	319
Creating the progress-indicator dialog box .....	320
The procedures .....	321
How this example works .....	322
Creating a Modeless Tabbed Dialog Box .....	323
Displaying a Chart in a UserForm .....	325
A Dialog Box Checklist .....	326
<b>CHAPTER 19: Accessing Your Macros through the User Interface .....</b>	<b>329</b>
Customizing the Ribbon .....	329
Customizing the Ribbon manually .....	330
Adding a macro to the Ribbon .....	332
Customizing the Ribbon with XML .....	333
Customizing the Excel UI with VBA .....	337

Adding commands to the Add-ins Ribbon tab . . . . .	338
Adding a new item to the Cell shortcut menu. . . . .	339
Adding customizations automatically. . . . .	340
Understanding shortcut menus and the single document interface. . . . .	341
<b>PART 5: PUTTING IT ALL TOGETHER . . . . .</b>	<b>343</b>
<b>CHAPTER 20: Creating Worksheet Functions . . . . .</b>	<b>345</b>
Create Custom Functions to Simplify Your Work . . . . .	345
Understanding VBA Function Basics. . . . .	347
Writing Functions. . . . .	347
Working with Function Arguments . . . . .	348
A function with no argument. . . . .	349
A function with one argument. . . . .	349
A function with two arguments . . . . .	351
A function with a range argument . . . . .	352
A function with an optional argument . . . . .	354
Introducing Wrapper Functions . . . . .	356
The NumberFormat function. . . . .	356
The ExtractElement function . . . . .	357
The SayIt function . . . . .	358
The IsLike function . . . . .	358
Working with Functions That Return an Array . . . . .	359
Returning an array of month names . . . . .	359
Returning a sorted list. . . . .	360
Using the Insert Function Dialog Box . . . . .	362
Displaying the function's description . . . . .	362
Adding argument descriptions . . . . .	364
<b>CHAPTER 21: Creating Excel Add-Ins . . . . .</b>	<b>365</b>
Add-Ins Defined . . . . .	365
Reasons to Create Add-Ins. . . . .	366
Working with Add-Ins . . . . .	367
Understanding Add-In Basics . . . . .	368
Looking at an Add-In Example. . . . .	369
Setting up the workbook . . . . .	369
Testing the workbook . . . . .	372
Adding descriptive information. . . . .	372
Protecting the VBA code. . . . .	373
Creating the add-in . . . . .	374
Opening the add-in . . . . .	374
Distributing the add-in . . . . .	375
Modifying the add-in. . . . .	375

<b>PART 6: THE PART OF TENS</b> .....	377
<b>CHAPTER 22: Ten Handy Visual Basic Editor Tips</b> .....	379
Applying Block Comments .....	380
Copying Multiple Lines of Code at Once .....	381
Jumping between Modules and Procedures .....	382
Teleporting to Your Functions .....	382
Staying in the Right Procedure .....	383
Stepping through Your Code .....	383
Stepping to a Specific Line in Your Code .....	384
Stopping Your Code at a Predefined Point .....	385
Seeing the Beginning and End of Variable Values .....	386
Turning Off Auto Syntax Check .....	387
<b>CHAPTER 23: Resources for VBA Help</b> .....	389
Letting Excel Write Code for You. ....	390
Referencing the Help System. ....	390
Pilfering Code from the Internet. ....	390
Leveraging User Forums. ....	391
Visiting Expert Blogs .....	392
Mining YouTube for Video Training .....	393
Attending Live and Online Training Classes .....	393
Learning from the Microsoft Office Dev Center .....	393
Dissecting the Other Excel Files in Your Organization .....	394
Asking Your Local Excel Guru .....	394
<b>CHAPTER 24: Ten VBA Do's and Don'ts</b> .....	395
Do Declare All Variables .....	395
Don't Confuse Passwords with Security .....	396
Do Clean Up Your Code .....	396
Don't Put Everything in One Procedure .....	397
Do Consider Other Software .....	397
Don't Assume That Everyone Enables Macros .....	397
Do Get in the Habit of Experimenting .....	398
Don't Assume That Your Code Will Work with Other Excel Versions .....	398
Do Keep Your Users in Mind .....	399
Don't Forget about Backups. ....	399
<b>INDEX</b> .....	401



# Introduction

---

**G**reetings, prospective Excel programmer. . .

You no doubt have your reasons for picking up a book on VBA programming. Maybe you got a new job (congratulations). Maybe you're trying to automate some of the repetitive data crunching tasks you have to do. Maybe you're just a nerd at heart. Whatever the reason, thank you for choosing this book.

Inside, you find everything you need to get up and running with VBA fast. Even if you don't have the foggiest idea of what programming is all about, this book can help. Unlike most programming books, this one is filled with information designed to include just what you need to know to quickly ramp your VBA programming skillset.

## About This Book

---

Go to any large bookstore (in person or online), and you'll find many Excel books. A quick overview can help you decide whether this book is really right for you. This book

- » Is designed for intermediate to advanced Excel users who want to get up to speed with Visual Basic for Applications (VBA) programming.
- » Requires no previous programming experience.
- » Covers the most commonly used commands.
- » Is appropriate for recent versions of Excel.
- » Just might make you crack a smile occasionally.

If you're using an older version of Excel, this book *might* be okay, but some things have changed. You'd probably be better off with the preceding edition.

Oh, yeah — this is *not* an introductory Excel book. If you're looking for a general-purpose Excel book, check out either of the following books, which are both published by Wiley:

- » *Excel 2019 For Dummies*, by Greg Harvey
- » *Excel Bible*, by Michael Alexander and Dick Kusleika

These books are also available in editions for earlier versions of Excel.

Notice that the title of this book isn't *The Complete Guide to Excel VBA Programming For Dummies*. This book doesn't cover all aspects of Excel programming — but then again, you probably don't want to know *everything* about this topic.

If you consume this book and find that you're hungry for a more comprehensive Excel programming book, you might try *Microsoft Excel 2019 Power Programming with VBA*, also published by Wiley. And yes, editions for older versions of Excel are also available.

To make the content more accessible, I divided this book into six parts:

- » **Part 1, Starting with Excel VBA Programming**
- » **Part 2, Employing VBA with Excel**
- » **Part 3, Programming Concepts**
- » **Part 4, Communicating with Your Users**
- » **Part 5, Putting It All Together**
- » **Part 6, The Part of Tens**

## Typographical conventions

Sometimes, I refer to key combinations — which means you hold down one key while you press another. For example, Ctrl+Z means you hold down the Ctrl key while you press Z.

For menu commands, I use a distinctive character to separate items on the Ribbon or menu. For example, you use the following command to create a named range in a worksheet:

Formulas ⇨ Defined Names ⇨ Define Name

Formulas is the tab at the top of the Ribbon, Defined Names is the Ribbon group, and Define Name is the Ribbon tool you click.

The Visual Basic Editor still uses old-fashioned menus and toolbars. So Tools⇨Options means choose the Tools menu and then choose the Options menu item.

Excel programming involves developing *code* — that is, the instructions VBA follows. All code in this book appears in a monospace font, like this:

```
Range("A1:A12").Select
```

Some long lines of code don't fit between the margins in this book. In such cases, I use the standard VBA line-continuation character sequence: a space followed by an underscore character. Here's an example:

```
Selection.PasteSpecial Paste:=xlValues, _  
    Operation:=xlNone, SkipBlanks:=False, _  
    Transpose:=False
```

When you enter this code, you can type it as written or place it on a single line (omitting the space and underscore combination).

## Macro security

It's a cruel world out there. It seems that some scam artist is always trying to take advantage of you or cause some type of problem. The world of computing is equally cruel. You probably know about computer viruses, which can cause some nasty things to happen to your system. But did you know that computer viruses can also reside in an Excel file? It's true. In fact, it's relatively easy to write a computer virus by using VBA. An unknowing user can open an Excel file and spread the virus to other Excel workbooks and to other systems.

Over the years, Microsoft has become increasingly concerned about security issues. This is a good thing, but it also means that Excel users need to understand how things work. You can check Excel's security settings by choosing File⇨Options⇨Trust Center⇨Trust Center Settings. There is a plethora of options in there, and people have been known to open that dialog box and never be heard from again.

If you click the Macro Settings tab (on the left side of the Trust Center dialog box), your options are as follows:

- » **Disable VBA macros without notification.** Macros will not work, regardless of what you do.
- » **Disable VBA macros with notification.** When you open a workbook with macros, you see the Message Bar open with an option you can click to enable macros, or (if the Visual Basic Editor window is open) you get a message asking if you want to enable macros.
- » **Disable VBA macros except digitally signed macros.** Only macros with a digital signature are allowed to run (but even for those signatures you haven't marked as trusted, you still get the security warning).
- » **Enable VBA macros.** All macros run with no warnings. This option is not recommended because potentially dangerous code can be executed.

Consider this scenario: You spend a week writing a killer VBA program that will revolutionize your company. You test it thoroughly and then send it to your boss. They call you into their office and claim that your macro doesn't do anything at all. What's going on? Chances are, your boss's security setting doesn't allow macros to run. Or maybe they chose to go along with Microsoft's default suggestion and disable the macros when they opened the file.

Bottom line? Just because an Excel workbook contains a macro does not guarantee that the macro will ever be executed. It all depends on the security setting and whether the user chooses to enable or disable macros for that file.

To work with this book, you need to enable macros for the files you work with. My advice is to use the second security level. Then, when you open a file that you've created, you can simply enable the macros. If you open a file from someone you don't know, you should disable the macros and check the VBA code to ensure that it doesn't contain anything destructive or malicious. Usually, it's pretty easy to identify suspicious VBA code.

Another option is to designate a trusted folder. Choose File⇨Options⇨Trust Center⇨Trust Center Settings. Select the Trusted Locations option and then designate a particular folder as a trusted location. Store your trusted workbooks there, and Excel won't bug you about enabling macros. For example, if you download the sample files for this book, you can put them in a trusted location.

# Foolish Assumptions

People who write books usually have a target reader in mind. The following points more or less describe the hypothetical target reader for this book:

- » You have access to a PC at work — and probably at home. And those computers are connected to the internet.
- » You're running a fairly recent version of Excel.
- » You've been using computers for several years.
- » You use Excel frequently in your work, and you consider yourself to be more knowledgeable about Excel than the average bear.
- » You need to make Excel do some things that you currently can't make it do.
- » You have little or no programming experience.
- » You understand that the Help system in Excel can actually be useful. Face it — this book doesn't cover everything. If you get on good speaking terms with the Help system, you'll be able to fill in some of the missing pieces.
- » You need to accomplish some work, and you have a low tolerance for thick, boring computer books.

## Icons Used in This Book

Throughout this book, icons in the margins highlight certain types of valuable information that call out for your attention. Here are the icons you'll encounter and a brief description of each.



TIP

The Tip icon marks tips and shortcuts that can save you a great deal of time (and maybe even allow you to leave the office at a reasonable hour).



REMEMBER

Remember icons mark the information that's especially important to know. To siphon off the most important information in each chapter, just skim through these icons.



TECHNICAL  
STUFF

The Technical Stuff icon marks information of a highly technical nature that you can normally skip over.



WARNING

The Warning icon tells you to watch out! It marks important information that may save you from losing data and ruining your whole day.

## Beyond the Book

This book has its very own website where you can download the sample files. To get these files, point your web browser to

<https://www.dummies.com/go/excelvbaprogrammingfd6e>

Having the sample files will save you a lot of typing. Better yet, you can play around with them and experiment with various changes. In fact, experimentation is the best way to master VBA.

In addition, this book comes with a free access-anywhere Cheat Sheet that includes keyboard shortcuts related to Excel VBA programming. To get this Cheat Sheet, simply go to [www.dummies.com](http://www.dummies.com) and type **VBA Excel Programming For Dummies Cheat Sheet** in the Search box and click on the Cheat Sheets tab.

## Where to Go from Here

This book contains everything you need to learn VBA programming at a mid-advanced level. The book starts off with the basics of recording macros and builds, chapter by chapter.

If you're completely new to Excel macros, start with Part 1 to get a refresher on the fundamentals of recording macros. If you have experience recording macros, but want to better understand the VBA behind them, read to Parts 2 and 3. There, you gain a concise understanding of how VBA works, along with the basic foundation you need to implement your own code.

Finally, if you're familiar with programming concepts and just want to get a quick run-through of some of the more advanced techniques like creating your custom functions and add-ins, feel free to jump to Part 4.

# **1**

# **Starting Excel VBA Programming**

**IN THIS PART . . .**

Get to know Visual Basic for Applications.

Work through a real-live Excel programming session.



- » Getting a conceptual overview of VBA
- » Finding out what you can do with VBA
- » Discovering the advantages and disadvantages of using VBA
- » Getting the lowdown on what VBA is
- » Staying Excel compatible

# Chapter 1

# Getting to Know VBA

If you're eager to jump into VBA programming, hold your horses. This chapter is completely devoid of any hands-on training material. It does, however, contain some essential background information that assists you in becoming an Excel programmer. Just like a great heart surgeon has to take freshman biology, you must learn some basic concepts and terminology so that the more practical aspects you do later make sense.

## Understanding VBA Basics

VBA, which stands for *Visual Basic for Applications*, is a programming language developed by Microsoft — you know, the company that tries to get you to buy a new version of Windows every few years. Excel, along with the other members of Microsoft Office, includes the VBA language (at no extra charge). In a nutshell, VBA is the tool that people use to write programs that automate Excel, such as a program to format a spreadsheet into a monthly report. In the next section, I discuss in more detail the types of tasks you can automate with VBA.

Imagine a robot that knows all about Excel. This robot can read instructions, and it can also operate Excel quickly and accurately. When you want the robot to do something in Excel, you write a set of robot instructions by using special codes. Then you tell the robot to follow your instructions while you sit back and drink a glass of lemonade. With VBA, you don't need the extra chair at your desk for an actual robot. Just feed the special codes into VBA and it does the work.



## A FEW WORDS ABOUT TERMINOLOGY

Excel programming terminology can be a bit confusing. For example, VBA is a programming language, but it also serves as a macro language. In this context, a *macro* is a set of instructions Excel performs to imitate keystrokes and mouse actions. What do you call something written in VBA and executed in Excel? Is it a *macro*, or is it a *program*? Excel's Help system often refers to VBA procedures as *macros*, so that terminology is used in this book. But you can also call this stuff a *program*.

You'll see the term *automate* throughout this book. This term means that a series of steps are completed automatically. For example, if you write a macro that adds color to some cells, prints the worksheet, and then removes the color, you have *automated* those three steps.

By the way, *macro* doesn't stand for **M**essy **A**nd **C**onfusing **R**epeated **O**peration. Rather, it comes from the Greek *makros*, which means large — which also describes your paycheck after you become an expert macro programmer.

## Knowing What VBA Can Do

You're probably aware that people use Excel for thousands of different tasks. Here are just a few examples:

- » Analyzing scientific data
- » Budgeting and forecasting
- » Creating invoices and other forms
- » Building charts from data
- » Keeping lists of things such as customers' names, students' grades, or holiday gift ideas (a nice fruitcake would be lovely)

The list could go on and on, but you get the idea. The point is simply that Excel is used for a wide variety of tasks, and everyone reading this book has different needs and expectations regarding Excel. One thing virtually every reader has in common is the *need to automate some aspect of Excel*. That, dear reader, is what VBA is all about.

For example, you might create a VBA program to import some numbers and then format and print your month-end sales report. After writing and testing the program, you can execute the macro with a single command, causing Excel to

automatically perform many time-consuming procedures. Rather than struggle through a tedious sequence of commands, you can simply click a button and then hop on over to TikTok and kill some time while your macro does the work.

The following sections briefly describe some common uses for VBA macros. One or two of these may push your button.

## **Inserting a bunch of text**

If you often need to enter your company name, address, and phone number in your worksheets, you can create a macro to do the typing for you. You can extend this concept as far as you like. For example, you might develop a macro that automatically enters a list of all salespeople who work for your company.

## **Automating a task you perform frequently**

Assume you're a sales manager and you need to prepare a month-end sales report to keep your boss happy. If the task is straightforward, you can develop a VBA program to do it for you. Your boss will be impressed by the consistently high quality of your reports, and you'll be promoted to a new job for which you are highly unqualified.

## **Automating repetitive operations**

If you need to perform the same action on, say, 12 different Excel workbooks, you can record a macro while you perform the task on the first workbook, and then let the macro repeat your action on the other workbooks. The nice thing about this is that Excel never complains about being bored. Excel's macro recorder is similar to recording live action on a video recorder. However, it doesn't require a camera, and the battery never needs to be recharged.

## **Creating a custom command**

Do you often issue the same sequence of Excel commands? If so, save yourself a few seconds by developing a macro that combines these commands into a single custom command, which you can execute with a single keystroke or button click. You probably won't save *that* much time, but you'll probably be more accurate. And the guy sitting in the next cubicle will be really impressed.

## Creating a custom button

You can customize your Quick Access toolbar with your own buttons that execute the macros you write. Office workers tend to be very impressed by buttons that perform magic. And if you *really* want to impress your fellow employees, you can even add new buttons to the Ribbon.

## Developing new worksheet functions

Although Excel includes hundreds of built-in functions (such as SUM and AVERAGE), you can create *custom* worksheet functions that can greatly simplify your formulas. You'll be surprised by how easy this is. (You can explore how to do this in Chapter 20.) Even better, the Insert Function dialog box displays your custom functions, making them appear built-in. Very snazzy stuff.

## Creating custom add-ins for Excel

You might be familiar with some of the add-ins that ship with Excel. For example, the Analysis ToolPak is a popular add-in. You can use VBA to develop your own special-purpose add-ins. Add-ins are just like regular Excel workbooks except they don't have any worksheets the users can see. They're all VBA and they usually automate tasks on other workbooks.

# Getting the Most from VBA

VBA provides a ton of benefits that are tempered with a few disadvantages you'll want to keep in mind. One advantage of VBA, for example, is that you can use it to automate almost anything you do in Excel. All you have to do is write instructions in VBA, and Excel carries out those instructions when instructed to do so. The previous section, "Knowing What VBA Can Do," describes some specific tasks you can accomplish by using VBA. The following sections help you decide whether VBA is the best tool to achieve the results you want.

## Knowing what VBA does best

Here are some benefits of automating a task by using VBA:

- » Excel always executes the task in exactly the same way. (In most cases, consistency is a good thing.)

- » Excel performs the task much faster than you can do it manually. (Unless, of course, you're Clark Kent.)
- » If you're a good macro programmer, Excel always performs the task without errors (which probably can't be said about you, no matter how careful you are).
- » If you set up things properly, someone who doesn't know anything about Excel can perform the task by running the macro.
- » You can do things in Excel that are otherwise impossible — which can make you a very popular person around the office.
- » For long, time-consuming tasks, you don't have to sit in front of your computer and get bored. Excel does the work while you hang out at the water cooler.

## Recognizing the disadvantages of using VBA

Using VBA can present some disadvantages (or *potential* disadvantages). Understanding these limitations upfront helps you avoid running down rabbit trails to nowhere. Most people use VBA to save time, not waste it!

Keep the following points in mind when deciding whether to use VBA:

- » Other people who need to use your VBA programs must have their own copies of Excel. It would be nice if you could press a button that transforms your Excel/VBA application into a stand-alone program, but that isn't possible. (And probably never will be.)
- » Sometimes, things go wrong. In other words, you can't blindly assume that your VBA program will always work correctly under all circumstances. Welcome to the world of *debugging* (fixing errors) and, if others are using your macros, providing technical support.
- » VBA is a moving target. As you know, Microsoft is continually upgrading Excel. Even though Microsoft puts great effort into compatibility between versions, you might discover that the VBA code you've written doesn't work properly with older versions or with a future version of Excel. For more information on the importance of Excel compatibility, see the section "Ensuring Excel Compatibility," later in this chapter.

# Understanding VBA Concepts

Just to let you know what you're in for, here's a quick-and-dirty summary of what VBA is all about.

» **You perform actions in VBA by writing (or recording) code in a VBA module.** You view and edit VBA modules by using the Visual Basic Editor (VBE).

» **A VBA module consists of Sub procedures.** A Sub procedure has nothing to do with underwater vessels or tasty sandwiches. Rather, it's a chunk of computer code that performs some action on or with objects (discussed in a moment). The following example shows a simple Sub procedure called `AddEmUp`. This amazing program, when executed, displays the result of 1 plus 1:

```
Sub AddEmUp( )  
    Sum = 1 + 1  
    MsgBox "The answer is " & Sum  
End Sub
```

A Sub procedure that doesn't perform properly is said to be *substandard*.

» **A VBA module can also have Function procedures.** A Function procedure returns a single value. You can call it from another VBA procedure or even use it as a function in a worksheet formula. An example of a Function procedure (named `AddTwo`) follows. This Function accepts two numbers (called arguments) and returns the sum of those values:

```
Function AddTwo(arg1, arg2)  
    AddTwo = arg1 + arg2  
End Function
```

A Function procedure that doesn't work correctly is said to be *dysfunctional*.

» **VBA manipulates objects.** Excel provides dozens and dozens of objects that you can manipulate. Examples of objects include a workbook, a worksheet, a cell range, a chart, and a shape. You have many more objects at your disposal, and you can manipulate them by using VBA code.

» **Objects are arranged in a hierarchy.** Objects can act as *containers* for other objects. At the top of the object hierarchy is Excel. Excel itself is an object called *Application*. The Application object contains other objects, such as Workbook objects and Add-In objects. The Workbook object can contain other objects, such as Worksheet objects and Chart objects. A Worksheet object can contain objects such as Range objects and PivotTable objects. The term *object model* refers to the arrangement of these objects. (Object-model mavens can find out more in Chapter 4.)