LEARNING MADE EASY

7th Edition

Microsoft® 365 Excel® VBA Programming for dummies®

A Wiley Brand

Build add-ins your whole office can use

Save time and reduce errors automating Excel with VBA

Write code faster with Copilot AI

Dick Kusleika

Microsoft® 365 Excel® VBA Programming

for dummies®
A Wiley Brand

Microsoft® 365 Excel® VBA Programming

7th Edition

by Dick Kusleika

for dummies®
A Wiley Brand

Microsoft® 365 Excel® VBA Programming For Dummies®, 7th Edition

Published by: **John Wiley & Sons, Inc.,** 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

Copyright © 2025 by John Wiley & Sons, Inc. All rights reserved, including rights for text and data mining and training of artificial technologies or similar technologies.

Media and software compilation copyright © 2025 by John Wiley & Sons, Inc. All rights reserved, including rights for text and data mining and training of artificial technologies or similar technologies.

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. 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: THE PUBLISHER AND THE AUTHOR 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 WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

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 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.

Library of Congress Control Number is available from the publisher.

ISBN 978-1-394-29239-4 (pbk); ISBN 978-1-394-29241-7 (ebk); ISBN 978-1-394-29240-0 (ebk)

Contents at a Glance

Introduction 1

Part 1: Starting Excel VBA Programming 7

CHAPTER 1: Getting to Know VBA 9

CHAPTER 2: Building Simple Macros 17

Part 2: Using VBA with Excel 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 65

Part 3: Creating Code Automatically 81

CHAPTER 6: Using the Excel Macro Recorder 83

CHAPTER 7: Generating Code with Copilot. 95

Part 4: Programming Concepts 107

CHAPTER 8: Essential VBA Language Elements 109

CHAPTER 9: Working with Range Objects 131

CHAPTER 10: Using VBA and Worksheet Functions 147

CHAPTER 11: Controlling Program Flow and Making Decisions 163

CHAPTER 12: Automatic Procedures and Events. 183

CHAPTER 13: Error-Handling Techniques 205

CHAPTER 14: Bug Extermination Techniques 217

CHAPTER 15: VBA Programming Examples 231

Part 5: Communicating with Your Users 255

CHAPTER 16: Simple Dialog Boxes 257

CHAPTER 17: UserForm Basics 275

CHAPTER 18: Using UserForm Controls 293

CHAPTER 19: UserForm Techniques and Tricks. 313

CHAPTER 20: Accessing Your Macros through the User Interface 341

Part 6: Putting It All Together 355

CHAPTER 21: Creating Worksheet Functions 357

CHAPTER 22: Creating Excel Add-Ins 379

Part 7: The Part of Tens 391

CHAPTER 23: Ten Handy Visual Basic Editor Tips 393

CHAPTER 24: Resources for VBA Help 403

CHAPTER 25: Ten VBA Do's and Don'ts 409

Index 415

Table of Contents

INTRODUCTION. 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

PART 1: STARTING EXCEL VBA PROGRAMMING. 7

CHAPTER 1: **Getting to Know VBA**. 9

Introducing 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

CHAPTER 2: **Building Simple Macros**. 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

PART 2: USING VBA WITH EXCEL . . . 29

CHAPTER 3: **Working in the Visual Basic Editor** . . . 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 . . . 45
Using the Editor Format tab . . . 47
Using the General tab . . . 48
Using the Docking tab . . . 49

CHAPTER 4: **Introducing the Excel Object Model** . . . 51
Working with the Excel Object Model . . . 51
Examining 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 . . . 62
Automatically listing properties and methods . . . 64

CHAPTER 5: **VBA Sub and Function Procedures** . . . 65
Understanding Subs versus Functions . . . 65
Looking at Sub procedures . . . 66
Looking at Function procedures . . . 66
Naming Subs and Functions . . . 67
Executing Sub Procedures . . . 68
Executing the Sub procedure directly . . . 70
Executing the procedure from the Macro dialog box . . . 71
Executing a macro by using a shortcut key . . . 71
Executing the procedure from a button or shape . . . 73

Executing the procedure from another procedure 74
Executing the procedure from the Immediate window. 75
Executing Function Procedures. 75
Calling the function from a Sub procedure 76
Calling a function from the Immediate window 77
Calling a function from a worksheet formula 78

PART 3: CREATING CODE AUTOMATICALLY. 81

CHAPTER 6: **Using the Excel Macro Recorder** 83
Recording Basics 83
Preparing to Record 85
Choosing Between Relative Mode and Absolute Mode. 85
Recording in absolute mode 86
Recording in relative mode 87
Watching the Macro Recorder in Action 89
Specifying Recording Options for Your Macro 90
Streamlining Code Generated by the Macro Recorder 92

CHAPTER 7: **Generating Code with Copilot**. 95
Installing Tools for Copilot 95
Writing Your First AI-Generated Code 97
Using Copilot Chat to Modify Generated Code. 102
Continuing the Copilot Chat. 104

PART 4: PROGRAMMING CONCEPTS 107

CHAPTER 8: **Essential VBA Language Elements**. 109
Using Comments in Your VBA Code 109
Using Variables, Constants, and Data Types .111
Understanding variables 111
What are VBA’s data types? 113
Declaring and scoping variables 114
Working with constants 120
Using premade constants 121
Working with strings 122
Working with dates 122
Using Assignment Statements. 123
Assignment statement examples 124
About that equal sign 124
Smooth operators 125
Working with Arrays 126
Declaring arrays. 127
Multidimensional arrays. 127
Dynamic arrays 128
Using Labels 129

CHAPTER 9: **Working with Range Objects** 131
Referring to Range Objects 131
Referring to a Range Using Properties 133
The Cells property 134
The Offset property 134
The Resize property 135
Working with Range Object Properties 136
The Value property 136
The Text property 137
The Count property 138
The Column and Row properties 138
The Address property 138
The HasFormula property 139
The Font property 140
The Interior property 140
The Formula property 142
The NumberFormat property 143
Taking Action with Range Object Methods 143
The Select method 144
The Copy and Paste methods 144
The Clear method 145
The Delete method 145

CHAPTER 10: **Using VBA and Worksheet Functions** 147
Understanding Functions 147
Using Built-In VBA Functions 148
Working with dates and times 148
Manipulating strings 150
Identifying objects and data 151
VBA functions that do more than return a value 152
Discovering VBA functions 153
Using Worksheet Functions in VBA 156
Worksheet function examples 157
Entering worksheet functions 159
More about using worksheet functions 160
Using Custom Functions 160

CHAPTER 11: **Controlling Program Flow and Making Decisions** 163
Going with the Flow, Dude 163
The GoTo Statement 164
Decisions, Decisions 166
The If-Then structure 166
The Select Case structure 170

Knocking Your Code for a Loop 174
For-Next loops 174
Do-While loop. 179
Do-Until loop 180
Using For Each-Next Loops with Collections 180

CHAPTER 12: **Automatic Procedures and Events** 183
Preparing for the Big Event 183
Learning when to use event procedures 185
Programming event-handler procedures 186
Knowing Where to Put the Event Code 186
Writing an Event-Handler Procedure 187
Triggering Workbook Events 189
The Open event for a workbook 189
The BeforeClose event for a workbook 192
The BeforeSave event for a workbook 193
Using Activation Events 194
Activate and deactivate events in a sheet 194
Activate and deactivate events in a workbook 195
Workbook activation events 196
Programming Worksheet-Related Events 197
The BeforeDoubleClick event 197
The BeforeRightClick event 198
The Change event 198
Understanding Events Not Associated with Objects 200
The OnTime event 200
Keypress events 203

CHAPTER 13: **Error-Handling Techniques** 205
Types of Errors 205
An Erroneous Macro Example 206
Not-quite-perfect macros 207
Perfecting the macro. 208
Giving up on perfection 210
Alternative Ways of Handling Errors. 210
Revisiting the EnterSquareRoot procedure 210
Trapping errors with the On Error statement 212
Handling Errors: The Details 212
Resuming after an error 212
Error handling in a nutshell 214
An Intentional Error. 215

CHAPTER 14: **Bug Extermination Techniques** 217
Species of Bugs 217
Identifying Bugs 219

Debugging Techniques 220
Examining your code. 220
Using the MsgBox function 220
Inserting Debug.Print statements 222
Using the VBA debugger. 223
Using the Debugger's Tools 223
Setting breakpoints in your code 223
Using the Watches window 226
Using the Locals window 228
Bug Reduction Tips 228

CHAPTER 15: **VBA Programming Examples** 231
Working with Ranges 232
Copying a range. 232
Copying a variable-size range 233
Selecting to the end of a row or column 235
Selecting a row or column 236
Moving a range 236
Looping through a range efficiently: Part I 237
Looping through a range efficiently: Part II 239
Prompting for a cell value 239
Determining the selection type 240
Identifying a multiple selection 241
Changing Excel Settings 242
Changing Boolean settings. 242
Changing non-Boolean settings 243
Working with Charts 243
AddChart versus AddChart2 244
Modifying the chart type 246
Looping through the ChartObjects collection 246
Modifying chart properties 247
Applying chart formatting 247
VBA Speed Tips 249
Turning off screen updating 249
Turning off automatic calculation 250
Eliminating those pesky alert messages 251
Simplifying object references 252
Declaring variable types 253
Using the With-End With structure 253

PART 5: COMMUNICATING WITH YOUR USERS 255

CHAPTER 16: **Simple Dialog Boxes**. 257
Interacting with the User in VBA 258
Displaying Messages with the MsgBox Function 259

Displaying a simple message box 259
Getting a response from a message box 260
Customizing message boxes 262
Getting Data with an Input Box 264
Understanding the InputBox syntax. 265
Using the InputBox function 265
Using the InputBox method. 267
Allowing the User to Select a File or Folder 268
Constructing a GetOpenFilename statement 268
Selecting a file with GetOpenFilename. 269
Picking a file with GetSaveAsFilename 271
Getting a folder name. 271
Displaying Excel's Built-In Dialog Boxes 272

CHAPTER 17: **UserForm Basics** 275
Knowing When to Use a UserForm. 275
Creating UserForms: An Overview 277
Working with UserForms 278
Inserting a new UserForm 278
Adding controls to a UserForm 279
Changing properties for a UserForm control 280
Viewing the UserForm Code pane 281
Displaying a UserForm 281
Using information from a UserForm 282
A UserForm Example 283
Creating the UserForm 283
Adding the CommandButtons. 284
Adding the OptionButtons 285
Adding event-handler procedures 287
Creating a macro to display the dialog box 289
Making the macro available. 289
Testing the macro 290

CHAPTER 18: **Using UserForm Controls**. 293
Getting Started with Dialog Box Controls 293
Adding controls 294
Introducing control properties 295
Learning Dialog Box Controls Details 297
CheckBox control. 297
ComboBox control 298
CommandButton control 299
Frame control. 300
Image control 300
Label control. 301
ListBox control 301

MultiPage control .303
OptionButton control .304
ScrollBar control .304
SpinButton control .305
TabStrip control .306
TextBox control .306
ToggleButton control .307
Working with Dialog Box Controls .308
Moving and resizing controls .308
Aligning and spacing controls .308
Accommodating keyboard users .309
Testing a UserForm .311
Dialog Box Aesthetics .311

CHAPTER 19: **UserForm Techniques and Tricks** 313
Using Dialog Boxes .314
A UserForm Example .314
Creating the dialog box .314
Writing code to display the dialog box .317
Making the macro available .317
Trying out your dialog box .318
Adding event-handler procedures .319
Validating the data .321
Now the dialog box works .321
A ListBox Control Example .322
Filling a ListBox Control .322
Determining the selected item .324
Determining multiple selections .325
Selecting a Range .326
Using Multiple Sets of Option Buttons .328
Using a Spin Button and a Text Box .329
Using a UserForm as a Progress Indicator .331
Creating the progress-indicator dialog box .332
The procedures .333
How this example works .334
Creating a Modeless Tabbed Dialog Box .335
Displaying a Chart in a UserForm .337
A Dialog Box Checklist .338

CHAPTER 20: **Accessing Your Macros through the User Interface** 341
Customizing the Ribbon .341
Customizing the Ribbon manually .342
Adding a macro to the Ribbon .344
Customizing the Ribbon with XML .344

Customizing the Excel UI with VBA . . . 349
Adding commands to the Add-ins Ribbon tab . . . 349
Adding a new item to the Cell shortcut menu . . . 350
Adding customizations automatically . . . 352
Understanding shortcut menus and the single document interface . . . 353

PART 6: PUTTING IT ALL TOGETHER . . . 355

CHAPTER 21: **Creating Worksheet Functions** . . . 357
Create Custom Functions to Simplify Your Work . . . 358
Understanding VBA Function Basics . . . 359
Writing Functions . . . 360
Working with Function Arguments . . . 360
A function with no argument . . . 361
A function with one argument . . . 362
A function with two arguments . . . 364
A function with a range argument . . . 365
A function with an optional argument . . . 367
Introducing Wrapper Functions . . . 369
The NumberFormat function . . . 369
The ExtractElement function . . . 369
The SayIt function . . . 370
The IsLike function . . . 371
Working with Functions That Return an Array . . . 372
Returning an array of month names . . . 372
Returning a sorted list . . . 373
Using the Insert Function Dialog Box . . . 374
Displaying the function's description . . . 375
Adding argument descriptions . . . 376

CHAPTER 22: **Creating Excel Add-Ins** . . . 379
Add-Ins Defined . . . 379
Reasons to Create Add-Ins . . . 380
Working with Add-Ins . . . 381
Understanding Add-In Basics . . . 382
Looking at an Add-In Example . . . 383
Setting up the workbook . . . 383
Testing the workbook . . . 386
Adding descriptive information . . . 386
Protecting the VBA code . . . 387
Creating the add-in . . . 387
Opening the add-in . . . 388
Distributing the add-in . . . 389
Modifying the add-in . . . 389

PART 7: THE PART OF TENS 391

CHAPTER 23: **Ten Handy Visual Basic Editor Tips** 393
Applying Block Comments 394
Copying Multiple Lines of Code at One Time 395
Jumping between Modules and Procedures 396
Teleporting to Your Functions 396
Staying in the Right Procedure 397
Stepping through Your Code 397
Stepping to a Specific Line in Your Code 398
Stopping Your Code at a Predefined Point 399
Seeing the Beginning and End of Variable Values 400
Turning Off Auto Syntax Check 401

CHAPTER 24: **Resources for VBA Help** 403
Letting Excel Write Code for You 403
Referencing the Help System 404
Pilfering Code from the Internet 404
Leveraging User Forums 405
Visiting Expert Blogs 406
Mining YouTube for Video Training 406
Attending Live and Online Training Classes 407
Learning from the Microsoft Office Dev Center 407
Prompting a Large Language Model Artificial Intelligence Engine 407
Asking Your Local Excel Guru 407

CHAPTER 25: **Ten VBA Do's and Don'ts** 409
Do Declare All Variables 409
Don't Confuse Passwords with Security 410
Do Clean Up Your Code 410
Don't Put Everything in One Procedure 411
Do Consider Other Software 411
Don't Assume That Everyone Enables Macros 411
Do Get in the Habit of Experimenting 412
Don't Assume That Your Code Will Work with Other Excel Versions 412
Do Keep Your Users in Mind 413
Don't Forget about Backups 413

INDEX 415

Introduction

Welcome, 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 skill set.

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 truly 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 one of the preceding editions of this book.

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, both published by Wiley:

- *Microsoft 365 Excel For Dummies,* by David Ringstrom
- *Microsoft Excel 365 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 Microsoft 365 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 *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 seven parts:

- Part 1: Starting Excel VBA Programming
- Part 2: Using VBA with Excel
- Part 3: Creating Code Automatically
- Part 4: Programming Concepts
- Part 5: Communicating with Your Users
- Part 6: Putting It All Together
- Part 7: 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 name of the tab at the top of the Ribbon, Defined Names is the name of the Ribbon group, and Define Name is the name of the Ribbon tool you click.

The Visual Basic Editor still uses old-fashioned menus and toolbars. So Tools➪Options means to 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 people on your network.

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. A plethora of options appear 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 see a message asking whether 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 see 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. 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 doesn't 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 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 these paragraphs.

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

`www.dummies.com/go/excelvbaprogrammingfd7e`

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` and type `Microsoft 365 Excel VBA 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 acquainted with the fundamentals of recording macros. If you have experience recording macros but want to better understand the VBA behind them, read Parts 2, 3, and 4. 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 5.

1 Starting Excel VBA Programming

IN THIS PART . . .

Get to know Visual Basic for Applications.

Work through a real, live Excel programming session.

IN THIS CHAPTER

» Forming 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 legendary heart surgeon has to first take freshman biology, you must learn some basic concepts and terminology so that the more practical aspects you use later make sense.

Introducing VBA Basics

VBA, which stands for *Visual Basic for Applications,* is a programming language developed by Microsoft and built right into Excel (and a few other programs, like Word, Outlook, and PowerPoint) at no extra charge. In a nutshell, VBA is a tool that people use to write programs that automate Excel, such as a program to format a spreadsheet full of data 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.

TECHNICAL STUFF

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 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 items or info 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 may 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. It happily performs the same actions and presses the same keystrokes over and over. And it's faster than a hundred caffeinated interns.

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 person sitting in the next cubicle will be duly 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 quite 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 offers 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 it is. (You can explore how to do it in Chapter 21.) Even better, the Insert Function dialog box displays your custom functions, making them appear built-in. 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 that they have no worksheets that users can see. Add-ins are all VBA, and they usually automate tasks on other workbooks.

Getting the Most from VBA

VBA provides a ton of benefits that are tempered by 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 you tell it to. The earlier section "Knowing What VBA Can Do" describes some specific tasks you can accomplish by using VBA. The following sections can 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.)