

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.

<p>LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: WHILE THE PUBLISHER AND AUTHORS HAVE USED THEIR BEST EFFORTS IN PREPARING THIS WORK,</p>
------------------------------------------------------------------------------------------------------------------------------------------------

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)

# Excel® VBA Programming For Dummies®

To view this book's Cheat Sheet, simply go to [www.dummies.com](http://www.dummies.com) and search for “Excel VBA Programming For Dummies Cheat Sheet” in the Search box.

## Table of Contents

[Cover](#)

[Title Page](#)

[Copyright](#)

[Introduction](#)

[About This Book](#)

[Foolish Assumptions](#)

[Icons Used in This Book](#)

[Beyond the Book](#)

[Where to Go from Here](#)

[\*\*Part 1: Starting Excel VBA Programming\*\*](#)

[\*\*Chapter 1: Getting to Know VBA\*\*](#)

[Understanding VBA Basics](#)

[Knowing What VBA Can Do](#)

[Getting the Most from VBA](#)

[Understanding VBA Concepts](#)

[Ensuring Excel Compatibility](#)

## **Chapter 2: Building Simple Macros**

[Displaying the Developer Tab](#)  
[Creating a Macro](#)  
[Preparing the Environment](#)  
[Recording a Macro](#)  
[Running the Macro](#)  
[Viewing a Macro in the Visual Basic Editor](#)  
[Modifying the Macro](#)  
[Saving Workbooks That Contain Macros](#)  
[Understanding Macro Security](#)

## **Part 2: Employing VBA with Excel**

### **Chapter 3: Working in the Visual Basic Editor**

[Getting to Know the Visual Basic Editor](#)  
[Working with the Project Explorer](#)  
[Working with a Code Pane](#)  
[Customizing the VBE](#)

### **Chapter 4: Introducing the Excel Object Model**

[Working with the Excel Object Model](#)  
[Diving into Object Properties and Methods](#)  
[Finding Out More from VBA Resources](#)

### **Chapter 5: VBA Sub and Function Procedures**

[Understanding Subs versus Functions](#)  
[Naming Subs and Functions](#)  
[Executing Sub procedures](#)  
[Executing Function Procedures](#)

### **Chapter 6: Using the Excel Macro Recorder**

[Recording Basics](#)  
[Preparing to Record](#)  
[Choosing Between Relative and Absolute Mode](#)

[Watching the Macro Recorder in Action](#)

[Specifying Recording Options for Your Macro](#)

[Streamlining Code Generated by the Macro Recorder](#)

## **Part 3: Programming Concepts**

### **Chapter 7: Essential VBA Language Elements**

[Using Comments in Your VBA Code](#)

[Using Variables, Constants, and Data Types](#)

[Using Assignment Statements](#)

[Working with Arrays](#)

[Using Labels](#)

### **Chapter 8: Working with Range Objects**

[Referring to Range Objects](#)

[Referring to a Range Using Properties](#)

[Working with Range Object Properties](#)

[Taking Action with Range Object Methods](#)

### **Chapter 9: Using VBA and Worksheet Functions**

[Understanding Functions](#)

[Using Built-In VBA Functions](#)

[Using Worksheet Functions in VBA](#)

[Using Custom Functions](#)

### **Chapter 10: Controlling Program Flow and Making Decisions**

[Going with the Flow, Dude](#)

[The GoTo Statement](#)

[Decisions, Decisions](#)

[Knocking Your Code for a Loop](#)

[Using For Each-Next Loops with Collections](#)

### **Chapter 11: Automatic Procedures and Events**

[Preparing for the Big Event](#)



[Knowing Where to Put the Event Code](#)

[Writing an Event-Handler Procedure](#)

[Triggering Workbook Events](#)

[Using Activation Events](#)

[Programming Worksheet-Related Events](#)

[Understanding Events Not Associated with Objects](#)

## **Chapter 12: Error-Handling Techniques**

[Types of Errors](#)

[An Erroneous Macro Example](#)

[Alternate Ways of Handling Errors](#)

[Handling Errors: The Details](#)

[An Intentional Error](#)

## **Chapter 13: Bug Extermination Techniques**

[Species of Bugs](#)

[Identifying Bugs](#)

[Debugging Techniques](#)

[Using the Debugger's Tools](#)

[Bug Reduction Tips](#)

## **Chapter 14: VBA Programming Examples**

[Working with Ranges](#)

[Changing Excel Settings](#)

[Working with Charts](#)

[VBA Speed Tips](#)

## **Part 4: Communicating with Your Users**

### **Chapter 15: Simple Dialog Boxes**

[Interacting with the User in VBA](#)

[Displaying Messages with the MsgBox Function](#)

[Getting Data with an Input Box](#)

[Allowing the User to Select a File or Folder](#)

[Displaying Excel's Built-In Dialog Boxes](#)

### **Chapter 16: UserForm Basics**

[Knowing When to Use a UserForm](#)

[Creating UserForms: An Overview](#)

[Working with UserForms](#)

[A UserForm Example](#)

## **Chapter 17: Using UserForm Controls**

[Getting Started with Dialog Box Controls](#)

[Learning Dialog Box Controls Details](#)

[Working with Dialog Box Controls](#)

[Dialog Box Aesthetics](#)

## **Chapter 18: UserForm Techniques and Tricks**

[Using Dialog Boxes](#)

[A UserForm Example](#)

[A ListBox Control Example](#)

[Selecting a Range](#)

[Using Multiple Sets of Option Buttons](#)

[Using a Spin Button and a Text Box](#)

[Using a UserForm as a Progress Indicator](#)

[Creating a Modeless Tabbed Dialog Box](#)

[Displaying a Chart in a UserForm](#)

[A Dialog Box Checklist](#)

## **Chapter 19: Accessing Your Macros through the User Interface**

[Customizing the Ribbon](#)

[Customizing the Excel UI with VBA](#)

## **Part 5: Putting It All Together**

### **Chapter 20: Creating Worksheet Functions**

[Create Custom Functions to Simplify Your Work](#)

[Understanding VBA Function Basics](#)

[Writing Functions](#)

[Working with Function Arguments](#)

[Introducing Wrapper Functions](#)

[Working with Functions That Return an Array](#)

[Using the Insert Function Dialog Box](#)

## **Chapter 21: Creating Excel Add-Ins**

[Add-Ins Defined](#)

[Reasons to Create Add-Ins](#)

[Working with Add-Ins](#)

[Understanding Add-In Basics](#)

[Looking at an Add-In Example](#)

## **Part 6: The Part of Tens**

### **Chapter 22: Ten Handy Visual Basic Editor Tips**

[Applying Block Comments](#)

[Copying Multiple Lines of Code at Once](#)

[Jumping between Modules and Procedures](#)

[Teleporting to Your Functions](#)

[Staying in the Right Procedure](#)

[Stepping through Your Code](#)

[Stepping to a Specific Line in Your Code](#)

[Stopping Your Code at a Predefined Point](#)

[Seeing the Beginning and End of Variable Values](#)

[Turning Off Auto Syntax Check](#)

### **Chapter 23: Resources for VBA Help**

[Letting Excel Write Code for You](#)

[Referencing the Help System](#)

[Pilfering Code from the Internet](#)

[Leveraging User Forums](#)

[Visiting Expert Blogs](#)

[Mining YouTube for Video Training](#)

[Attending Live and Online Training Classes](#)

[Learning from the Microsoft Office Dev Center](#)

[Dissecting the Other Excel Files in Your Organization](#)

[Asking Your Local Excel Guru](#)

### **Chapter 24: Ten VBA Do's and Don'ts**

[Do Declare All Variables](#)  
[Don't Confuse Passwords with Security](#)  
[Do Clean Up Your Code](#)  
[Don't Put Everything in One Procedure](#)  
[Do Consider Other Software](#)  
[Don't Assume That Everyone Enables Macros](#)  
[Do Get in the Habit of Experimenting](#)  
[Don't Assume That Your Code Will Work with Other Excel Versions](#)  
[Do Keep Your Users in Mind](#)  
[Don't Forget about Backups](#)

## **Index**

### **About the Author**

### **Connect with Dummies**

### **End User License Agreement**

# List of Tables

## **Chapter 7**

[TABLE 7-1 VBA's Built-In Data Types](#)

[TABLE 7-2 Variable's Scope](#)

[TABLE 7-3 VBA's Operators](#)

[TABLE 7-4 VBA's Logical Operators](#)

## **Chapter 9**

[TABLE 9-1 VBA Functions with Useful Side Benefits](#)

[TABLE 9-2 VBA's Most Useful Built-In Functions](#)

## **Chapter 10**

[TABLE 10-1 Programming Constructs for Making Decisions](#)

## **Chapter 11**

[TABLE 11-1 Workbook Events](#)

[TABLE 11-2 Worksheet Events](#)

## **Chapter 12**

[TABLE 12-1 Using the On Error Statement](#)

[TABLE 12-2 Using the Resume Statement](#)

## **Chapter 15**

[TABLE 15-1 MsgBox Function Arguments](#)

[TABLE 15-2 Constants Used in the MsgBox Function](#)

[TABLE 15-3 Constants Used as Return Values for the MsgBox Function](#)

[TABLE 15-4 InputBox Function Arguments](#)

[TABLE 15-5 GetOpenFilename Method Arguments](#)

[TABLE 15-6 GetSaveAsFilename Method Arguments](#)

## **Chapter 16**

[TABLE 16-1 Toolbox Controls](#)

## **Chapter 17**

[TABLE 17-1 Common Control Properties](#)

## **Chapter 18**

[TABLE 18-1 Settings for the MultiSelect Property](#)

## **Chapter 20**

[TABLE 20-1 Commission Rates by Sales](#)

# **List of Illustrations**

## **Chapter 2**

[FIGURE 2-1: The Developer tab is normally hidden, but it's easy to unhide.](#)

[FIGURE 2-2: The Record Macro dialog box appears when you're about to record a m...](#)

[FIGURE 2-3: The completed Record Macro dialog box.](#)

[FIGURE 2-4: The VBE displays the VBA code in Module1 of Book1.](#)

[FIGURE 2-5: If your workbook contains macros, and you attempt to save it in a m...](#)

[FIGURE 2-6: The Macro Settings section of the Trust Center dialog box.](#)

[FIGURE 2-7: Excel's warning that the file to be opened contains macros.](#)

[FIGURE 2-8: Excel's warning that the workbook just opened contains macros. You ...](#)

## **Chapter 3**

[FIGURE 3-1: The VBE is your customizable friend.](#)

[FIGURE 3-2: This Project Explorer lists projects that can be expanded to show m...](#)

[FIGURE 3-3: Code pane overload isn't a pretty sight.](#)

[FIGURE 3-4: The GuessName procedure displays this dialog box.](#)

[FIGURE 3-5: The Editor tab of the Options dialog box.](#)

[FIGURE 3-6: An example of Auto List Members.](#)

[FIGURE 3-7: Auto Quick Info offers help about the MsgBox function.](#)

[FIGURE 3-8: Change how the VBE looks with the Editor Format tab.](#)

[FIGURE 3-9: The General tab of the Options dialog box.](#)

[FIGURE 3-10: The Docking tab of the Options dialog box.](#)

## **Chapter 4**

[FIGURE 4-1: This message box displays a Range object's Value property.](#)

[FIGURE 4-2: The VBE displays a list of arguments while you type.](#)

[FIGURE 4-3: An example from VBA's Help system.](#)

[FIGURE 4-4: Browsing for objects with the Object Browser.](#)

[FIGURE 4-5: The Auto List Members feature helps you identify properties and met...](#)

## **Chapter 5**

[FIGURE 5-1: Using the built-in VBA InputBox function to get a number.](#)

[FIGURE 5-2: Displaying the cube root of a number via the MsgBox function.](#)

[FIGURE 5-3: The Macro dialog box lists all available Sub procedures.](#)

[FIGURE 5-4: The Macro Options dialog box lets you set options for your macros.](#)

[FIGURE 5-5: The Ribbon, showing the controls available when you click Insert on...](#)

[FIGURE 5-6: When you add a button to a worksheet, Excel automatically displays ...](#)

[FIGURE 5-7: Executing a Function in the Immediate window returns the answer imm...](#)

[FIGURE 5-8: The CubeRoot function appears in the User Defined category of the I...](#)

[FIGURE 5-9: Using the CubeRoot function in formulas.](#)

## **Chapter 6**

[FIGURE 6-1: A convenient window arrangement for watching the macro recorder do ...](#)

[FIGURE 6-2: The Record Macro dialog box provides several options.](#)

## **Chapter 7**

[FIGURE 7-1: Pressing Ctrl+spacebar displays a list of variable names, reserved ...](#)

[FIGURE 7-2: Each VBA module has a Declarations section, which appears before an...](#)

## **Chapter 8**

[FIGURE 8-1: A noncontiguous range selection.](#)

[FIGURE 8-2: This message box displays the Address property of a 5 × 5 rang...](#)

## **Chapter 9**

[FIGURE 9-1: Calculating the length of your name.](#)

[FIGURE 9-2: A way to display a list of VBA functions.](#)

[FIGURE 9-3: Using a worksheet function in your VBA code.](#)

[FIGURE 9-4: The range, named PriceList, contains prices for parts.](#)

[FIGURE 9-5: Getting a list of worksheet functions that you can use in your VBA ...](#)

## **Chapter 10**

[FIGURE 10-1: A message displayed by the CheckCell procedure.](#)

[FIGURE 10-2: Using a loop to apply background shading to rows.](#)

[FIGURE 10-3: These cells were filled using a nested For-Next loop.](#)

[FIGURE 10-4: Using loops to create a checkerboard pattern.](#)

## Chapter 11

[FIGURE 11-1: The Project window displays items for a single project.](#)

[FIGURE 11-2: Choosing an event in the ThisWorkbook object's module.](#)

[FIGURE 11-3: This event-handler procedure is executed when the workbook is open...](#)

[FIGURE 11-4: Using a Workbook\\_Open event-handler to keep track of how many time...](#)

[FIGURE 11-5: When a chart sheet is activated, the user sees a message like this...](#)

[FIGURE 11-6: Performing data validation with an event procedure.](#)

## Chapter 12

[FIGURE 12-1: The InputBox function displays a dialog box asking the user for a ...](#)

[FIGURE 12-2: Excel displays this error message when the procedure attempts to c...](#)

[FIGURE 12-3: Running the procedure when a chart is selected generates this erro...](#)

[FIGURE 12-4: A runtime error in the procedure generates this semi-helpful error...](#)

[FIGURE 12-5: If an error occurs, the user can decide whether to try again.](#)

## Chapter 13

[FIGURE 13-1: An error message like this often means that your VBA code contains...](#)

[FIGURE 13-2: Using a message box to display the value of three variables.](#)

[FIGURE 13-3: Pressing Ctrl+Break halts execution of your code and gives you som...](#)

[FIGURE 13-4: A Debug.Print statement sends output to the Immediate window.](#)

[FIGURE 13-5: The highlighted statement marks a breakpoint in this procedure.](#)

[FIGURE 13-6: A typical scene in Break mode.](#)

[FIGURE 13-7: The Add Watch dialog box lets you specify a condition that causes ...](#)



[FIGURE 13-8: The Watches window displays all watches.](#)

[FIGURE 13-9: The Locals window displays all local variables and their content.](#)

## **Chapter 14**

[FIGURE 14-1: This range can consist of any number of rows.](#)

[FIGURE 14-2: Use the VBA InputBox function to get a value from the user.](#)

[FIGURE 14-3: Excel doesn't like it when you try to copy a multiple selection.](#)

[FIGURE 14-4: You can instruct Excel to not display these types of alerts while ...](#)

## **Chapter 15**

[FIGURE 15-1: A simple message box.](#)

[FIGURE 15-2: A simple message box, with two buttons.](#)

[FIGURE 15-3: The MsgBox function's buttons argument determines what appears in ...](#)

[FIGURE 15-4: This dialog box, displayed by the MsgBox function, displays a titl...](#)

[FIGURE 15-5: The InputBox function displays this dialog box.](#)

[FIGURE 15-6: Another example of using the InputBox function.](#)

[FIGURE 15-7: Using the Application.InputBox method to get a range.](#)

[FIGURE 15-8: The GetOpenFilename method displays a customizable dialog box and ...](#)

[FIGURE 15-9: Displaying one of Excel's dialog boxes by using VBA.](#)

[FIGURE 15-10: Using the Customize Ribbon tab to identify a command name.](#)

## **Chapter 16**

[FIGURE 16-1: You can get information from the user by displaying a UserForm.](#)

[FIGURE 16-2: A new UserForm object.](#)

[FIGURE 16-3: Use the Properties windows to change the properties of UserForm co...](#)

[FIGURE 16-4: The UserForm with two CommandButton controls.](#)

[FIGURE 16-5: This is the UserForm after adding three OptionButton controls insi...](#)

[FIGURE 16-6: Assign a shortcut key to execute the ChangeCase macro.](#)

[FIGURE 16-7: Adding the ChangeChase macro to the Quick Access toolbar.](#)

[FIGURE 16-8: The UserForm is in action.](#)

[FIGURE 16-9: The text has been converted to uppercase.](#)

## **Chapter 17**

[FIGURE 17-1: A UserForm in the VBE, with a few controls added.](#)

[FIGURE 17-2: Use the Properties window to make design-time changes to a control...](#)

[FIGURE 17-3: Change some properties by selecting from a drop-down list of valid...](#)

[FIGURE 17-4: CheckBox Controls in a UserForm.](#)

[FIGURE 17-5: ComboBox controls in a UserForm.](#)

[FIGURE 17-6: CommandButton controls.](#)

[FIGURE 17-7: An Image control displays a photo.](#)

[FIGURE 17-8: Label controls can take on many different looks.](#)

[FIGURE 17-9: ListBox controls.](#)

[FIGURE 17-10: Use a MultiPage control to create a tabbed dialog box.](#)

[FIGURE 17-11: Two sets of OptionButton controls, each contained in a Frame cont...](#)

[FIGURE 17-12: Two RefEdit controls.](#)

[FIGURE 17-13: A ScrollBar control with a Label control below it.](#)

[FIGURE 17-14: SpinButton controls.](#)

[FIGURE 17-15: TextBox controls.](#)

[FIGURE 17-16: ToggleButton controls.](#)

[FIGURE 17-17: Choose Format ⇒ Align to change the alignment of UserForm con...](#)

[FIGURE 17-18: The Tab Order dialog box.](#)

[FIGURE 17-19: Use labels to provide direct access to controls that don't have a...](#)

## **Chapter 18**

[FIGURE 18-1: This dialog box logs dinner choices for guests.](#)

[FIGURE 18-2: Executing the LogDinnerGuest procedure displays the dialog box.](#)

[FIGURE 18-3: Use the custom dialog box for data entry.](#)

[FIGURE 18-4: Determining which item in a list box is selected.](#)

[FIGURE 18-5: Determining the selected items in a list box that allows multiple ...](#)

[FIGURE 18-6: This dialog box lets the user select a range.](#)

[FIGURE 18-7: This dialog box contains three sets of OptionButton controls.](#)

[FIGURE 18-8: A UserForm with a spin button and a companion text box.](#)

[FIGURE 18-9: This UserForm functions as a progress indicator for a lengthy macr...](#)

[FIGURE 18-10: The progress-indicator UserForm.](#)

[FIGURE 18-11: The three tabs of a MultiPage control.](#)

[FIGURE 18-12: Displaying a chart in a UserForm.](#)

## **Chapter 19**

[FIGURE 19-1: The Customize Ribbon tab of the Excel Options dialog box.](#)

[FIGURE 19-2: The View tab with a new group named Text To Speech.](#)

[FIGURE 19-3: RibbonX code displayed in the Custom UI Editor.](#)

[FIGURE 19-4: The VBA callback procedure that is executed by clicking the Ribbon...](#)

[FIGURE 19-5: Proof that adding a new Ribbon command using XML is actually possi...](#)

[FIGURE 19-6: A control on the Add-in tab isn't flashy, but it gets the job done...](#)

[FIGURE 19-7: The Cell shortcut menu showing a custom menu item: Change Case.](#)

## **Chapter 20**

[FIGURE 20-1: Using the Commission function in a worksheet.](#)

[FIGURE 20-2: Using the Commission2 function, which takes two arguments.](#)

[FIGURE 20-3: Using a custom function to sum only odd numbers.](#)

[FIGURE 20-4: Using the ExtractElement function to return an element from a stri...](#)

[FIGURE 20-5: Using the MonthNames function to return a 12-element array.](#)

[FIGURE 20-6: Using a custom function to return a sorted range.](#)

[FIGURE 20-7: By default, the Insert Function dialog box doesn't provide a descr...](#)

[FIGURE 20-8: The custom function now displays a description.](#)

[FIGURE 20-9: By default, the Function Arguments dialog box displays Function ar...](#)

## **Chapter 21**

[FIGURE 21-1: The Add-Ins dialog box lists all the add-ins known to Excel.](#)

[FIGURE 21-2: The UserForm for the Change Case add-in.](#)

[FIGURE 21-3: Use the Properties section to enter descriptive information about ...](#)

[FIGURE 21-4: The Add-Ins dialog box has the new add-in selected.](#)

[FIGURE 21-5: Making an add-in not an add-in.](#)

## **Chapter 22**

[FIGURE 22-1: A single apostrophe in front of any line turns that line into a co...](#)

[FIGURE 22-2: The Edit toolbar allows you to select entire blocks of code and ap...](#)

[FIGURE 22-3: Holding down the Ctrl key while dragging code creates a copy of th...](#)

[FIGURE 22-4: Pressing Shift+F2 with your cursor on a function or variable name ...](#)

[FIGURE 22-5: Click the Procedure View button to show only the active procedure.](#)

[FIGURE 22-6: Press F8 key to step through each line of your macro at your own p...](#)

[FIGURE 22-7: You can click and drag the yellow arrow while stepping through you...](#)

[FIGURE 22-8: A breakpoint is marked by a dot in the left margin along with shad...](#)

[FIGURE 22-9: Showing the ending characters in a variable tooltip.](#)

[FIGURE 22-10: Leaving an unfinished line of code, even for a second, results in...](#)

[FIGURE 22-11: Uncheck the Auto Syntax Check option to prevent warning messages ...](#)

# Introduction

---

Greetings, 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](#).

**Part 1**  
**Starting Excel VBA  
Programming**