

# **Excel® 2007 VBA Programmer's Reference**

John Green  
Stephen Bullen  
Rob Bovey  
Michael Alexander

ISBN: 978-0-470-04643-2

## **Chapter 1 Primer in Excel VBA**

SKU: 9785CH0000062



# Excel® 2007 VBA Programmer's Reference

Published by  
**Wiley Publishing, Inc.**  
10475 Crosspoint Boulevard  
Indianapolis, IN 46256  
www.wiley.com

Copyright © 2007 by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-0-470-04643-2

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

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 either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4355, or online at <http://www.wiley.com/go/permissions>.

**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 or to obtain technical support, please contact our Customer Care Department within the U.S. at (800) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Library of Congress Cataloging-in-Publication Data

Excel 2007 VBA programmer's reference / John Green ... [et al.].

p. cm.

Includes index.

ISBN 978-0-470-04643-2 (paper/website)

1. Microsoft Excel (Computer file) 2. Business—Computer programs. I. Green, John, 1945-  
HF5548.4.M523E92988 2007  
005.54—dc22

2007004976

**Trademarks:** Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Microsoft and Excel are registered trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Wrox Blox: Excel 2007 VBA Programmers Reference By John Green, Stephen Bullen, Rob Bovey, Michael Alexander - ISBN: 9780470046432 Copyright 2008, Wiley Publishing Inc. This PDF is exclusively for your use in accordance with the WroxBlox/eChapters Terms of Service. No part of it may be reproduced or transmitted in any form by any means without prior written permission from the publisher. Redistribution or other use that violates the Wrox Blox Terms of Service or otherwise violates the U.S. copyright laws is strictly prohibited.

# 1

## Primer in Excel VBA

This chapter is intended for those who are not familiar with Excel and the Excel macro recorder, or who are inexperienced with programming using the Visual Basic language. If you are already comfortable with navigating around the features provided by Excel, have used the macro recorder, and have a working knowledge of Visual Basic and the Visual Basic Editor, you might want to skip straight to Chapter 2.

If this is not the case, this chapter has been designed to provide you with the information you need to be able to move on comfortably to the more advanced features presented in the following chapters. Specifically, this chapter covers the following topics:

- The Excel macro recorder
- User-defined functions
- The Excel object model
- VBA programming concepts

Excel VBA is a programming application that allows you to use Visual Basic code to run the many features of the Excel package, thereby allowing you to customize your Excel applications. Units of VBA code are often referred to as *macros*. More formal terminology is covered in this chapter, but you will continue to see the term *macro* as a general way to refer to any VBA code.

In your day-to-day use of Excel, if you carry out the same sequence of commands repetitively, you can save a lot of time and effort by automating those steps using macros. If you are setting up an application for other users who don't know much about Excel, you can use macros to create buttons and dialog boxes to guide them through your application as well as automate the processes involved.

If you are able to perform an operation manually, you can use the *macro recorder* to capture that operation. This is a very quick and easy process and requires no prior knowledge of the VBA language. Many Excel users record and run macros and feel no need to learn about VBA.

## Chapter 1: Primer in Excel VBA

---

However, the recorded results might not be very flexible, in that the macro can only be used to carry out one particular task on one particular range of cells. In addition, the recorded macro is likely to run much more slowly than code written by someone with knowledge of VBA. To set up interactive macros that can adapt to change and also run quickly, and to take advantage of more advanced features of Excel such as customized dialog boxes, you need to learn about VBA.

**Don't get the impression that we are dismissing the macro recorder. The macro recorder is one of the most valuable tools available to VBA programmers. It is the fastest way to generate working VBA code, but you must be prepared to apply your own knowledge of VBA to edit the recorded macro to obtain flexible and efficient code. A recurring theme in this book is recording an Excel macro and then showing how to adapt the recorded code.**

In this chapter, you learn how to use the macro recorder and you see all the ways Excel provides to run your macros. You see how to use the *Visual Basic Editor* to examine and change your macros, thus going beyond the recorder and tapping into the power of the VBA language and the *Excel object model*.

You can also use VBA to create your own worksheet functions. Excel comes with hundreds of built-in functions, such as `SUM` and `IF`, which you can use in cell formulas. However, if you have a complex calculation that you use frequently and that is not included in the set of standard Excel functions — such as a tax calculation or a specialized scientific formula — you can write your own *user-defined function*.

## Using the Macro Recorder

Excel's macro recorder operates very much like the recorder that stores the greeting on your telephone answering machine. To record a greeting, you first prepare yourself by rehearsing the greeting to ensure that it says what you want. Then you switch on the recorder and deliver the greeting. When you have finished, you switch off the recorder. You now have a recording that automatically plays when you leave a call unanswered.

Recording an Excel macro is very similar. You first rehearse the steps involved and decide at what points you want to start and stop the recording process. You prepare your spreadsheet, switch on the Excel recorder, carry out your Excel operations, and switch off the recorder. You now have an automated procedure that you and others can reproduce at the press of a button.

## Recording Macros

Say you want a macro that types six month names as three-letter abbreviations, Jan to Jun, across the top of your worksheet, starting in cell B1. I know this is rather a silly macro because you could do this easily with an AutoFill operation, but this example will serve to show you some important general concepts:

- ❑ First, think about how you are going to carry out this operation. In this case, it is easy — you will just type the data across the worksheet. Remember, a more complex macro might need more rehearsals before you are ready to record it.

- ❑ Next, think about when you want to start recording. In this case, you should include the selection of cell B1 in the recording, because you want to always have Jan in B1. If you don't select B1 at the start, you will record typing Jan into the active cell, which could be anywhere when you play back the macro.
- ❑ Next, think about when you want to stop recording. You might first want to include some formatting such as making the cells bold and italic, so you should include that in the recording. Where do you want the active cell to be after the macro runs? Do you want it to be in the same cell as Jun, or would you rather have the active cell in column A or column B, ready for your next input? Assume that you want the active cell to be A2, at the completion of the macro, so you will select A2 before turning off the recorder.
- ❑ Now you can set up your screen, ready to record.

In this case, start with an empty worksheet with cell A1 selected. If you can't see the Developer tab above the Ribbon, you will need to click the round Microsoft Office button that you can see in the top-left corner of the Excel screen shown in Figure 1-1. Click Excel Options at the bottom of the dialog box and select Personalize. Select the checkbox for Show Developer tab in the Ribbon and click OK. Now you can select the Developer section of the Ribbon and click Record Macro to display the Record Macro dialog box, shown in Figure 1-1.

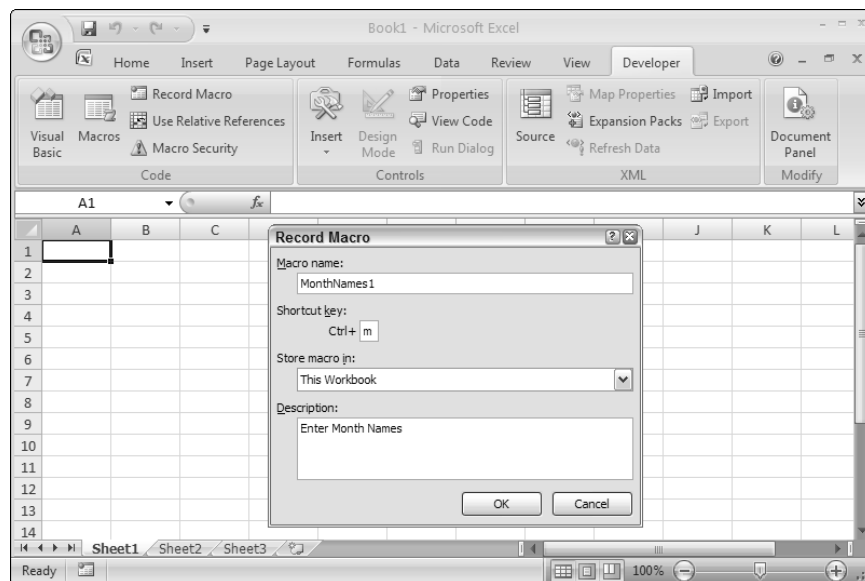


Figure 1-1

In the Macro name: box, replace the default entry, such as Macro1, with the name you want for your macro. The name should start with a letter and contain only letters, numbers, and the underscore character, with a maximum length of 255 characters. The macro name must not contain special characters such as exclamation points (!) or question marks (?), nor should it contain blank spaces. It is also best to use a short but descriptive name that you will recognize later. You can use the underscore character to separate words, but it is easy to just use capitalization to distinguish words.

## Chapter 1: Primer in Excel VBA

---

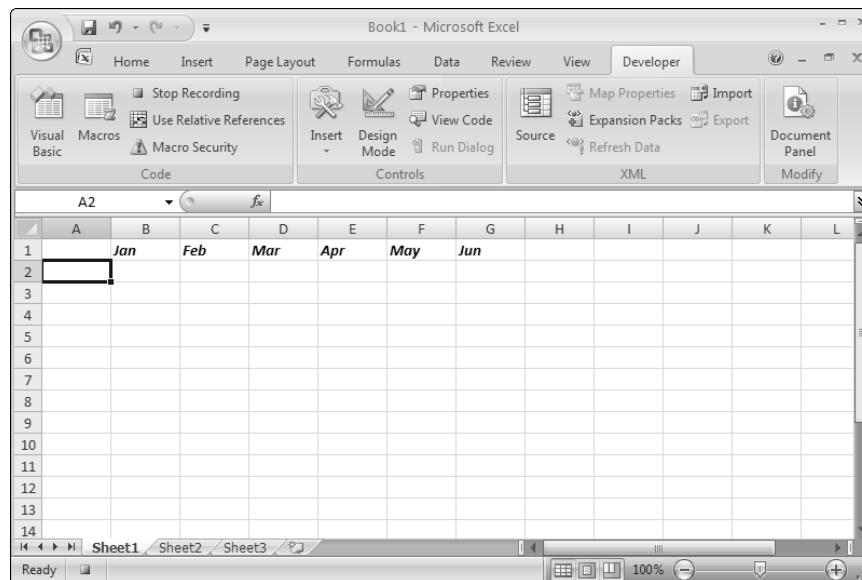
Call the macro `MonthNames1`, because you will create another version later.

In the **Shortcut key:** box, you can type in a single letter. This key can be pressed later, while holding down the **Ctrl** key, to run the macro. Use a lowercase `m`. Alternatively, you can use an uppercase `M`. In this case, when you later want to run the macro, you need to use the keystroke combination **Ctrl+Shift+M**. It is not mandatory to provide a shortcut key; you can run a macro in a number of other ways, as you will see.

In the **Description:** box, you can add text that will be added as comments to the macro. These lines will appear at the top of your macro code. They have no significance to VBA, but provide you and others with information about the macro.

All Excel macros are stored in workbooks. You are given a choice regarding where the recorded macro will be stored. The **Store macro in:** combo box lists three possibilities. If you choose **New Workbook**, the recorder will open a new empty workbook for the macro. **Personal Macro Workbook** refers to a special hidden workbook, which is discussed in a moment. Choose **This Workbook** to store the macro in the currently active workbook.

When you have filled in the **Record Macro** dialog box, click the **OK** button. You will see a new **Stop Recording** button appear on the left side of the status bar at the bottom of the screen, as shown in **Figure 1-2**. You will also notice that the **Start Recording** button in the Ribbon has been replaced by a new **Stop Recording** button.



**Figure 1-2**

You should now click cell B1, type in `Jan`, and fill in the rest of the cells as shown in **Figure 1-2**. Then select B1:G1 and click the **Bold** and **Italic** buttons on the **Home** tab of the **Ribbon**. Click the A2 cell and then stop the recorder. You can stop the recorder by clicking the **Stop Recording** button on the **Ribbon** or by clicking the **Stop Recording** button on the status bar.