What You Can Do with Outlook 2007

Whether you've been using Microsoft Outlook for just a few days or for several years, you've certainly figured out that it does more than email—much, much more. It's not unusual to find people who have organized their entire lives with Outlook. Companies that develop add-ins for Outlook view the application as a great platform because so many people "live in Outlook."

But if you asked each Outlook user how he or she puts the program to work, you would receive a different answer every time, because people have their own ideas on how to organize the critical information in their lives. Wouldn't it be great if Microsoft could make Outlook so customizable that everyone could use it in his or her own way? It might not be 100 percent possible, but the programming environment included with Outlook 2007 is rich enough to let you make great strides toward bending Outlook to your will, or that of the organization you work for.

This book shows you how to use the programming tools that come with the Outlook application to make it your own. It's OK if you have never programmed before. This book shows you the basics! Progamming Outlook is much easier than you think. For experienced programmers, we cover how to work with Outlook data, how to put its special features to work, and how to work around some of its quirks. Even professional developers can find useful information here on the basic building blocks of Outlook programming. Our focus in this book, though, is on what you can build "out of the box" without Visual Studio or any additional programming tools.

The highlights of this chapter include discussions of the following:

- What kinds of programming projects are possible in Outlook 2007
- What tools you will use
- How to decide which tool to use for a particular project
- How to make an initial sketch of your plans for Outlook programming projects

1.1 Why program with Outlook?

Maybe you're an information technology professional managing a network of users and need a way to report on the data visible in Outlook. Or perhaps you're one of those network users and can imagine ways to make your work more productive if only Outlook would do _____ (you fill in the blank). Maybe you even use Outlook at home, as well as at the office, and wish you knew how to extend its capabilities as a personal information manager to organize more of your activities. The good news in this book is that Outlook 2007 makes it easier than ever to customize the application to streamline repetitive tasks, add new capabilities, and integrate with other Office applications.

To help you get excited about the chapters ahead, take a look at this list of things you can do when you learn how to program with Outlook:

- Create your own custom rules to handle incoming messages
- Search and replace data, such as telephone area codes
- Create custom reports by integrating Outlook data into HTML-format messages, Word documents, and Excel worksheets
- Schedule a follow-up call for a meeting
- Create Outlook forms that duplicate the paper phone message, vacation request, and other business forms that you use

1.2 Outlook programming tools

Let's start by previewing the primary tools that you will be using:

- Visual Basic for Applications (VBA)
- Outlook forms
- Visual Basic Scripting Edition (VBScript)
- Programming models for other Office applications

1.2.1 Visual Basic for Applications

Outlook includes a rich development environment for creating macros, event handlers, and other procedures—Visual Basic for Applications, or VBA for short. Other Office programs also have VBA, but so do AutoCAD and other applications that have licensed VBA as their programming environment.

Note: Microsoft has a new application-centric development environment called Visual Studio Tools for Applications (VSTA), built on the .NET Framework, but in Office 2007, only InfoPath supports VSTA. Other Office 2007 applications, including Outlook, still use VBA as their integrated programming environment.

Figure 1.1 shows the VBA programming environment. (Most of the screen shots in this book were taken using Windows Vista. If you use Windows XP or Windows 2003, your screen will look slightly different, but Outlook code should function the same regardless of the operating system version.) The VBA programming environment includes many tools to help you learn how to write VBA code:

- Visual forms designer (to create Windows dialogs in VBA, not custom Outlook forms)
- Intelligent editor with color coding and dropdown lists to avoid code errors
- Detailed index to Outlook programming techniques
- Properties windows and other tools

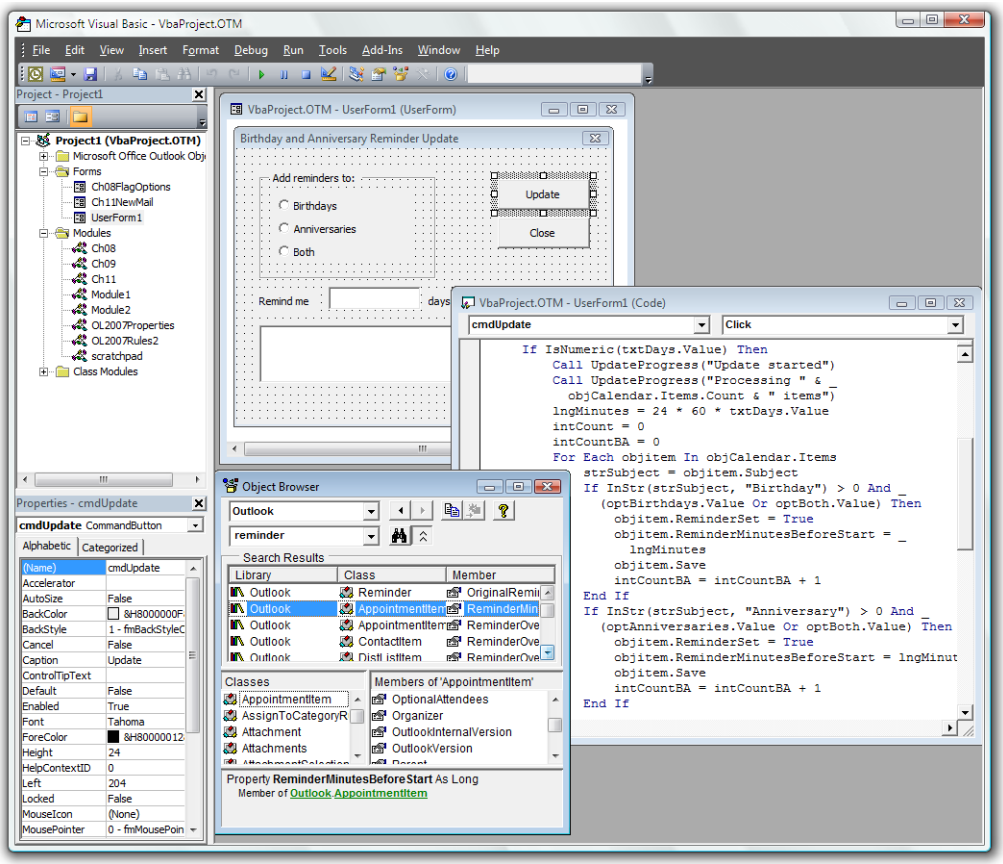

Figure 1.1 *VBA includes a rich form and code environment (compare with Figure 1.4).*

VBA code can enhance many of the operations that take place when you work with your Outlook information, such as creating new items or switching from one folder to another. As you'll see, most of those operations have corresponding events in the Outlook programming library that let you respond to such operations automatically.

VBA also gives you the ability to design pop-up dialog boxes to get information from and windows that stay on the screen to provide information to the user. For example, you might build a VBA form to display how many vacation days you have used so far this year or the time that you last received messages in your Inbox.

Furthermore, you can use VBA to create macros that you can add to the Outlook toolbar to launch a telephone message form, search for and replace text, run rules on demand, and expand Outlook's capabilities in many other ways. You can even create VBA procedures that the Outlook Rules Wizard can execute as "run a script" rule actions.

You might have created macros in Word or Excel by turning on a macro recorder that watches your actions and then builds the appropriate code. Outlook does not include a comparable macro recorder, but the examples in this book should give you all the basic building blocks you will need to construct truly useful Outlook macros.

Note that the VBA techniques discussed in this book also apply if you want to move up from Outlook's integrated development environment to building more sophisticated Outlook tools with Microsoft's Visual Studio or other development tools. They also apply to VBA code written in Word, Excel, Access, or other Office applications that need to automate Outlook.

1.2.2 Custom Outlook forms

The second stop on the road to Outlook programming proficiency is learning how to customize the basic Outlook forms.

Every item that you open in Outlook—whether it's an email message, a contact, or an appointment—uses a particular form to display its data. (If you have programmed in Microsoft Access, you may already be familiar with using forms as templates to display different data records.) You can customize these forms to show or hide fields or whole pages, respond to user input and actions, and launch other Outlook operations. If you work within an organization that uses Microsoft Exchange as its mail server, you may be able to collaborate with other people by using custom Outlook forms. With a little more effort, it is also possible (although much less common) to use custom forms for collaboration with other Outlook users across the Internet.

In many programming environments, you must start from scratch every time you want to create a new window for the user to interact with. Out-

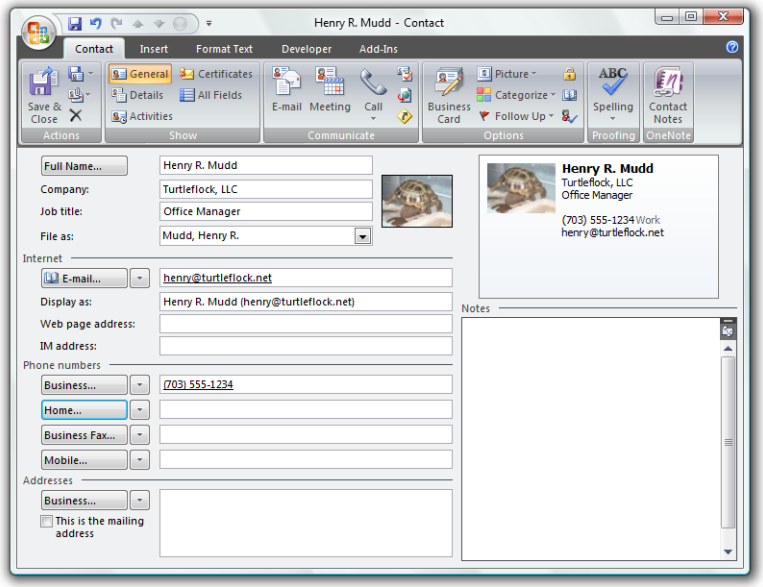

Figure 1.2
This Contact form has not been customized.

look is different in that it presents you with a group of built-in forms. To build a custom form, you start with one of the built-in forms and then add your own special touches.

For example, people often ask how to show a contact's age, not just record the birthday. Figure 1.2 shows the default Contact form as it normally looks. The Birthday field is on the Details page, and so is not visible. In Figure 1.3, you see the same form, only this time it has been customized with a *form region* to provide a control for entering the birthday from the main page and a box to calculate the age. Form regions are a new feature in Outlook 2007 that allows you add to or replace pages on custom Outlook forms. We'll learn more about them in Chapter 5, "Introducing Form Regions."

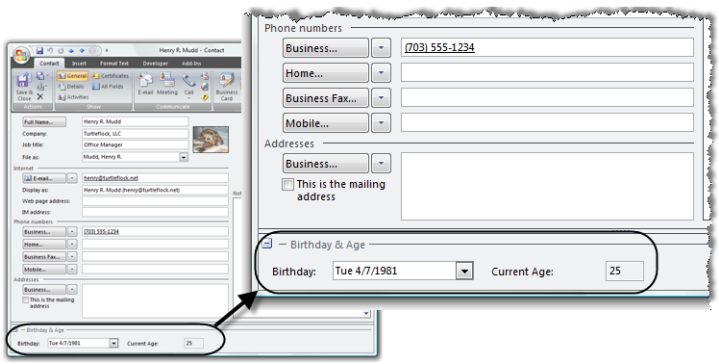

Figure 1.3
This Contact form has been customized with a form region to show the birthday and age of the contact.

Was any programming code required to do this? Not really. All it took was a formula, not that different from those formulas you might have written for Microsoft Excel worksheets.

Custom form regions are just the newest way to customize Outlook forms. You can also create custom forms by adding controls and custom fields to five or six pages on any of the basic six standard forms that come with Outlook.

Given that no code is required just to add controls and fields, is this truly Outlook programming? Sure it is! Many of the changes you want to make to Outlook might involve nothing more than adding new fields and pages to existing forms to hold that extra data. Without writing any code at all, you can perform simple validation to make sure that the data meets your criteria for correctness and develop formulas such as the one for calculating a person's age.

1.2.3 Visual Basic Scripting Edition

A time will come, however, when you want your custom Outlook forms to do more. Maybe you will want to generate a task for a follow-up telephone call from an appointment and have Outlook automatically fill in the contact name for you. Perhaps you want to be able to enter the birth date for a contact's spouse or partner and have Outlook automatically create a new recurring event in your Calendar folder. When you are ready to go beyond entering data and manipulating it in simple ways, you can move up to *VBScript*, the shorthand name for Visual Basic Scripting Edition, the programming language behind Outlook forms.

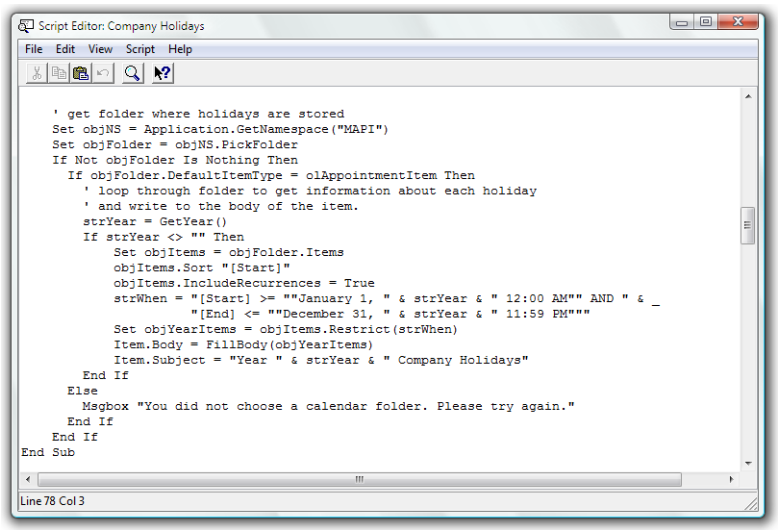

Figure 1.4 *The Outlook form script editor is just a text editor (compare with Figure 1.1).*

You might have heard of VBScript in the context of Web pages. VBScript is one of several languages that can control what you see when you interact with a Web page. It also works with the Windows Script Host (WSH) scripting environment that Microsoft has included with Windows since Windows 98. With WSH, you can write routines that are stored as simple text files and can be run at a command line.

Scripts don't run as fast as other kinds of programs, but they enjoy the advantage of small size and portability. Having a script associated with an Outlook form hardly increases the size of the form at all.

VBScript is a little scary, though. It's like walking a tightrope without a net, because the built-in editor for building VBScript programs is, well, a text editor. Figure 1.4 shows a sample script for a form to distribute a list of company holidays within an organization. The form script editor has none of the color-coding or automatic syntax checking that you get with VBA.

One sneaky technique that you will learn in this book is to write and test your Outlook form code in the superior VBA code environment, make a few minor adjustments to adapt it to VBScript, and then copy and paste it into the script window of an Outlook form. This method cuts down on programming time immensely.

1.2.4 Folder home pages

When you start Outlook for the first time, you see Outlook Today, a summary of your Inbox, Calendar, and Tasks folders. Outlook Today is actually a Web page included with Outlook, a specific example of a *folder home page*. Every folder in Outlook can be set up on its Properties dialog to display a Web page instead of the contents of the folder. The Web page can even include the list of items in the folder, through a special ActiveX control called the Outlook View Control, but it can also show something unrelated to Outlook, such as a SharePoint site or an Intranet help desk page. Folder home pages can also help document for users what kinds of activities they can perform in an Exchange public folder.

Two things make folder home pages interesting to network administrators who want to explore Outlook's configuration scripting support:

- With Group Policy Objects or the Office Customization Tool, you can control which default folders in Outlook show home pages and what pages they show.
- Folder home pages can run VBScript code to access Outlook automation objects and configure such things as rules.

Figure 1.5 shows an example of a folder home page used to deploy Outlook settings. We'll explore such techniques in Chapter 22, "Rules, Views, and Administrator Scripting Tasks."

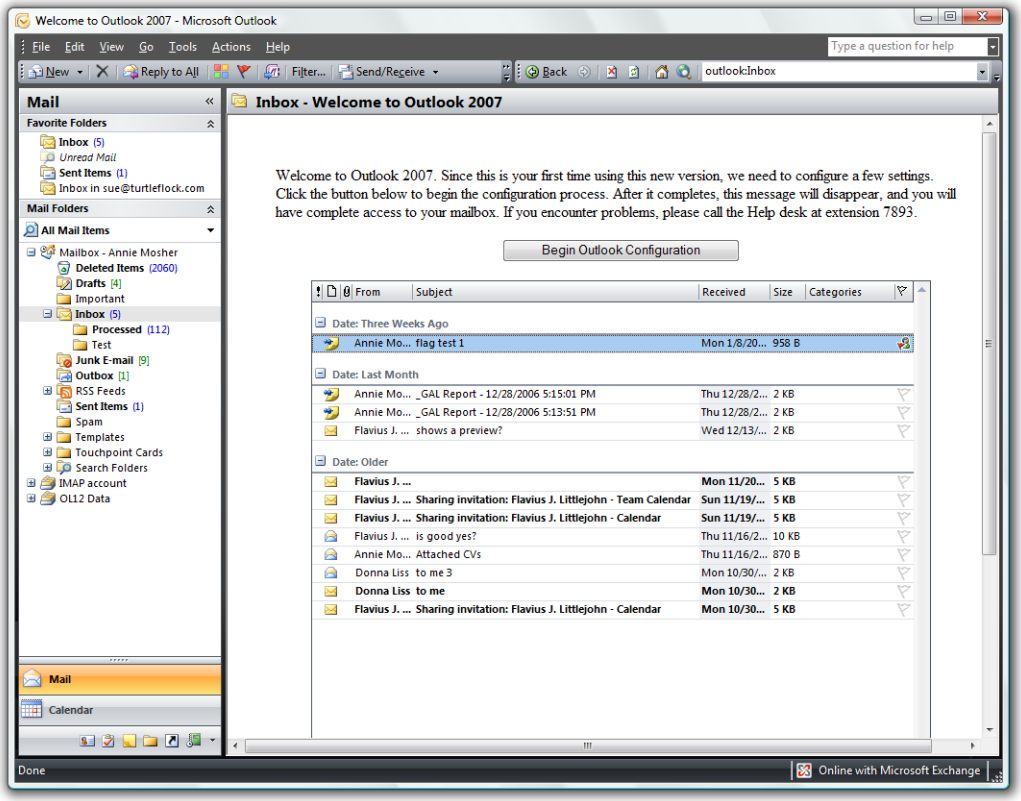

Figure 1.5 *Folder home pages can display data from any Outlook folder and run script code like any other Web page.*

What happened to CDO?

Previous versions of Outlook relied on a programming interface called Collaboration Data Objects (CDO) for many programming tasks that were not possible through the Outlook object model, such as displaying the Address Book dialog or getting the mobile phone number from a user in the Exchange Global Address List. Outlook 2007's object model has approximately doubled compared with Outlook 2003, and it can now handle virtually all the programming tasks that once required CDO—and do it without triggering security prompts. Therefore, Microsoft is no longer shipping CDO 1.21 with Outlook 2007. It is, however, available as a Web download for use with legacy applications that require it. This book does not cover programming with CDO 1.21.

1.2.5 Office integration and other object models

Outlook can create Word or Excel documents, and Microsoft Office programs such as Excel and Word can create messages, appointments, and other Outlook items. This integration is possible thanks to something called the *Outlook object model*, a programming library that opens Outlook to automation not just through Outlook VBA but from other applications' code environments as well. Furthermore, all the Office programs and many other Windows components also have object models that reveal what those programs can do, the types of items (or objects) they can work with, and the characteristics of those items. Word objects, for example, are crucial to creating Outlook messages with complex formatting.

1.3 How to start

At this point, you might feel that the hardest task in Outlook programming is knowing where to start. Do you use VBScript or VBA? Do you work with a form first and then write the programming code or vice versa?

I would recommend that you start by choosing one or more compelling projects—ideas that will save you time in the long run, make repetitive tasks less burdensome, or perhaps just display information that is hard to extract from the standard Outlook interface. Try to be as specific as possible. Don't decide to build a project to make Outlook work just like GoldMine (a popular sales contact management program). Instead, pick a particular GoldMine feature that you want Outlook to duplicate.

When you choose a project, don't start writing code or moving fields around on a form right away! Instead, take some time to outline what you want the project to accomplish, using what programmers call *pseudo code*.

But wait! You say you don't know how to writing programming code. ("That's why I bought this book!," you protest.) No, I'm not asking you to write a program (not yet), only to lay the groundwork. When you write pseudo code, you're walking through the logic of what you want to happen, without worrying about the exact language required to make it work.

For example, let's say that you want to enhance Outlook's appointment form with a button that would create a new task for a follow-up telephone call to the person you met with. The pseudo code might look something like this:

```
User clicks button
    Show task form
    Copy details of meeting to task body
    Copy contact from meeting to task
    Set task due date for one week from the meeting date
    If task due date falls on a weekend or holiday
        Then adjust the due date to the next business day
    Save the task
```

Nothing in this list looks like programming, but it describes in detail what you want Outlook to do when the user clicks the follow-up call button that you'll add to the form. It won't take much to move from this pseudo code to the programming code that implements those steps.

Once you decide what project to tackle and have an idea of what the finished project should do, how do you decide which tool is appropriate? Table 1.1 provides some recommendations for tools appropriate to particular situations. Don't take these recommendations as hard and fast rules. In many cases, you can approach a project in several ways. As you work through the examples in the chapters that follow, you will develop a better feel for which Outlook tool works best and which approach you feel more comfortable implementing.

Note that Table 1.1 does not include such approaches as add-ins, task panes, custom toolbars, and smart tags. While professional developers include such elements in their Outlook-integrated applications, creating

Table 1.1 *Choosing Outlook Tools*

If you want to . . .	Try this approach . . .
Show additional information in an individual item window	Modify an Outlook form or design a custom form region
Show additional information about an item in the reading pane	Design a custom form region
Take some action in response to something that the user does with an Outlook item	Modify an Outlook form with VBScript code
Click a button on the Outlook toolbar to make something happen to the current item or items	Write a macro in VBA
Make something happen when the user starts Outlook, switches to a different folder, or performs other actions that don't involve a particular Outlook item	Write an event handler in VBA
Process an incoming message or meeting request	Write a procedure in VBA that can be invoked from an Outlook rule using a "run a script" action or write a VBA event handler
Display status information as the user performs various Outlook tasks	Create a user form in VBA with a routine that keeps the status information up-to-date
Show data from multiple Outlook folders in a single view	Use a folder home page with multiple instances of the Outlook View Control

Table 1.2 *Key Outlook Development Components*

Component	Features
Microsoft Outlook	.NET Programmability Support
	Visual Basic Scripting Support
Office Shared Features	Digital Signature for VBA Projects
	Visual Basic for Applications
Office Tools	Microsoft Forms 2.0 .NET Programmability Support
	Microsoft Script Editor (HTML Source Editing)/Web Scripting/Web Debugging (supported on Windows Vista only with Visual Studio 2005)
	Smart Tag .NET Programmability Support

them requires additional development tools such as Microsoft Visual Studio. In this book, we're going to stick to the programming you can do just with Outlook and the other Office programs.

1.4 Key Outlook programming components

If you use the default settings to install Office or Outlook, you should have almost all the built-in development tools you need. Table 1.2 lists those components and where you'll find them in the feature installation state lists when you run Setup.exe to install or update Outlook or Office 2007. Note that the Web Debugging component, which is used to debug Outlook custom form VBScript code, is not supported on computers using Windows Vista as the operating system unless you also have Visual Studio 2005 (not Visual Studio 2005 Express) installed. Also, the .NET support components are needed only if you are using Visual Studio. VBA and VBScript code do not need them.

As noted in the "What Happened to CDO?" sidebar, Collaboration Data Objects is no longer a part of an Outlook installation, but is available as a separate download for backward compatibility.

1.5 Showing developer commands

Some of the developer features in Outlook don't appear by default. To see developer commands on individual items, follow these steps:

1. From the main Outlook menu, choose Tools | Options.
2. Switch to the Mail Format tab.
3. Click Editor Options.

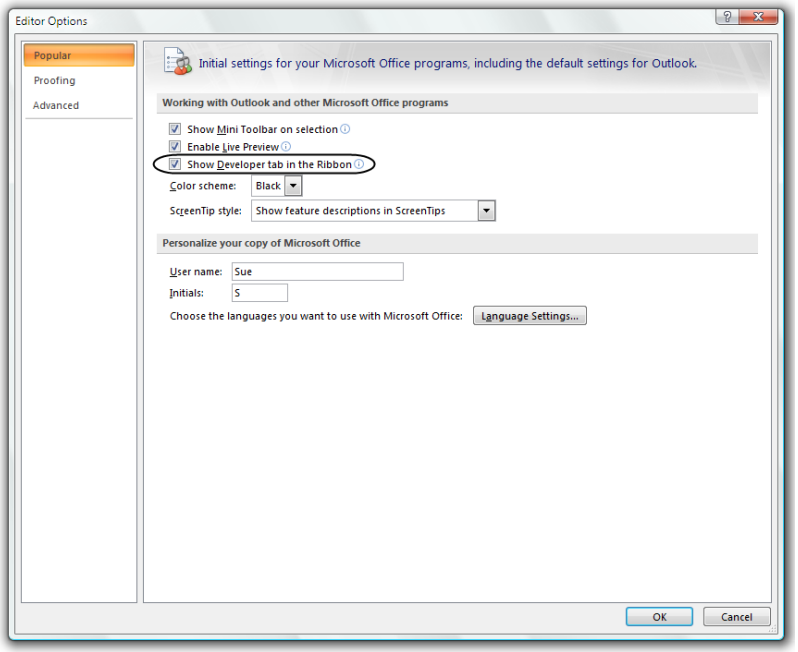

Figure 1.6 *Some Outlook developer commands won't be visible until you turn on the Developer tab.*

Where's the .NET code?

You may have noticed that Table 1.2 lists several components that provide .NET programmability support to Outlook but that none of the approaches in Table 1.1 mentioned .NET. We are not going to cover .NET programming in this book or show any code samples in VB.NET or C#. I know that may sound odd, given that Visual Studio and its .NET languages comprise Microsoft's latest and greatest programming environment. Microsoft even has a special edition of Visual Studio (Visual Studio Tools for Office) for creating add-ins for Outlook. However, the programming tools in the versions of Outlook and Office that you buy at the store or that come preinstalled on a new computer use not .NET, but the older VBA and VBScript programming languages. Future versions of Office may replace VBA with a new .NET programming environment called Visual Studio Tools for Applications that makes its debut in InfoPath 2007, but in Outlook 2007, the customization languages appropriate for administrators and power users (the main audience for this book) are still VBA and VBScript.

That said, the Outlook object model works the same, regardless of what language you use, and so I hope that professional developers working in .NET languages will find some of the material in this book useful to their understanding of how to accomplish the basic programming tasks in Outlook.

4. In the Editor Options dialog (see Figure 1.6), check the box for "Show Developer tab in the Ribbon."

The "Ribbon" is Office 2007's new command interface, replacing toolbars and menus on Word documents, Excel worksheets, PowerPoint presentations, and individual Outlook items.

1.6 Summary

At first, Microsoft Outlook programming can seem complex because there is more than one tool and no clear indicator of where to start. This book is divided into sections that introduce Outlook skills one at a time, with examples that you can easily try on your own computer. After an introduction in Part I to VBA design, in Part II you will learn about Outlook form design. If you are completely new to writing code, Part III will give you the basics that you'll need to write both VBA and VBScript code. If you already have coding experience, feel free to skip ahead to Part IV, which dives into the specifics of writing code for the Outlook object model, both in VBA and in VBScript behind custom forms. Finally, in Part V, you'll find out how to integrate Outlook with Word and Excel to print reports, work with rules and views, manage forms and some key user settings, and modify the toolbar on Outlook's main menu.

Code samples in this book are available from http://www.outlook-code.com, along with book suggestions for professional developers, more code samples, discussion forums, and other Outlook programming resources.