Introduction

Welcome to Macros and Add-ins

This is a book about developing macros and add-ins for Visual Studio .NET. Every programmer who uses Visual Studio .NET, regardless of the type of project he or she is developing, and regardless of his or her skill level, sooner or later wonders: Why can't Visual Studio .NET do such-and-such? Or: Wouldn't it be nice if Visual Studio .NET had this feature I'm thinking of? Or: There must be a way to simplify this task that I do over and over!

Surely you've thought of ways you could make Visual Studio .NET better, whether it's simply by automating a repetitive task that you have to do 158 times today or by implementing a full-scale improvement Visual Studio .NET that, for example, would add an entire window that contains suggestions for misspelled words in the comments in your code.

For the first case, simplifying repetitive tasks, you can either use a macro, which is a simple, interpreted piece of code, or an add-in, which is a compiled library that Visual Studio .NET loads and runs. In the second case, creating a new window for the Visual Studio .NET environment, you use an add-in. For numerous other tasks, you can use either a macro or an add-in.

This is all possible thanks to the developers at Microsoft, who, when they created Visual Studio .NET, included an *automation library* that provides a set of the objects and classes for manipulating the Visual Studio .NET environment. This library is available to you whether you're writing a macro or an add-in.

For the macros, Microsoft has built on the previous generation of macro tools for its products. Whereas these earlier tools featured Visual Basic for Applications (VBA), Visual Studio .NET uses Visual Basic .NET for its macro language. This means that not only can your macros drive Visual Studio .NET, but they can access the full, rich .NET framework library.

For the add-ins, Microsoft used a process that already existed in the Microsoft Office products. (For those of you who are already familiar with add-ins, I'm talking about

the IDTExtensibility2 interface that Microsoft has used in the past.) This means that if you already know how to develop add-ins for Microsoft Office, you've got a head start on developing add-ins for Visual Studio .NET. Likewise, after you read the chapters in this book on how to develop add-ins for Visual Studio .NET, you will know how to design add-ins for Microsoft Office. For this reason, I have devoted a chapter to using Visual Studio .NET to develop add-ins for Microsoft Office products.

All that said, this book does much more than just explain how to develop and use macros and add-ins. By working through the entire book, you will also learn about .NET programming in general. Specifically, you learn:

- How the Common Language Runtime works.
- What Microsoft Intermediate Language is.
- How to develop satellite DLLs that provide for globalization of your software.
- What COM is, how .NET differs from COM, and how you can make the two work together.
- And much more!

In short, at the conclusion of this book, you will have a very strong knowledge of .NET programming in general.

Required Tools

The tools you need to get the most from this book depend on what you want to do:

- *If you plan to develop add-ins*, you must have the Professional Edition, or either of the Enterprise editions of Visual Studio .NET. (Theoretically, you could develop add-ins using the Standard edition of C++, VB.NET, or C#, but you would have an uphill battle. For that reason, if you're seriously interested in developing add-ins, I recommend that you purchase a Professional or Enterprise Edition.)
- *If you plan to develop macros*, you can use any edition of Visual Studio .NET, provided it's a .NET edition. Though you can develop macros for earlier editions of Visual Studio, to take advantage of all the features I describe in this book, you will want to have a .NET edition.

Minimal system requirements include the following: You must have Windows NT 4.0 or later (that is, anything except Windows 3.1, 95, and 98; and if you're going with XP, you'll want the Professional, not the Home Edition). Microsoft also lists some requirements on its Web site; however, I recommend that you go way beyond these requirements, at least in RAM. Remember, these are *minimal* requirements, meaning Visual Studio .NET will run, but not necessarily very quickly, giving you plenty of time to go refill the coffee cup as the virtual memory loads and unloads. Here are the minimum RAM requirements Microsoft suggests:

Microsoft Windows XP Professional. 160 megabytes (MB) of RAM

Windows 2000 Professional. 96 MB of RAM

Windows 2000 Server. 192 MB of RAM

Windows NT 4.0 Workstation. 64 MB of RAM

Windows NT 4.0 Server. 160 MB of RAM

I strongly recommend that you double or even triple these numbers, especially if you're developing add-ins, because often you'll run two instances of Visual Studio .NET on a single computer. For the development of this book, I used Windows 2000 Professional with 320 Meg of RAM, and there were times my system had trouble keeping up. However, in general 320 Meg on a Win2000 box did fine. Remember, these days, RAM is pretty cheap, and if you're a serious developer, RAM is definitely something you will want to invest in. Go for 512 Meg if you can.

As for other tools, consider that programming is changing. These days, thanks to the miracle of hypertext, we have access to endless bits and bytes of documentation, all online, which we can access without having to worry about our computers grinding to a halt. And for the purposes of this book, you will need access to the online help that ships with Visual Studio .NET. So if you didn't install it already, do so before you begin reading this book, because rather than rewriting the online help, listing every single method of every single object, I provide you with an augmentation to the online help. Also, at times I refer you to the online help for more information. (But note: I do *not* assume prior knowledge of macros and add-ins, and I certainly don't expect you to read the online help before reading this book. The online help is simply a reference; this book will teach you what you need to know.)

Finally, I'll state the obvious by adding that you should have easy access to the Internet, preferably high-speed, for several reasons. First, this book has an accompanying Web site, where you can find additional information from me, more add-ins and macros, and a forum where you can share ideas and thoughts with other programmers. Second, even with the online help loaded, you will be spending a good deal of time at Microsoft's development site, http://msdn.microsoft.com, specifically in the library at http://msdn.microsoft.com/library/. If you aren't familiar with these two sites, I encourage you to take a peek at them the next time you're online, for it is at these sites that you can find nearly all the answers to anything you might ever get stuck on, well beyond the topics of macros and add-ins.

Background Knowledge

I assume you are familiar with .NET Programming basics, such as how to create solutions and projects, how to use some of the basic framework classes such as those in the IO namespace, how to run console applications, how to use the form designers, how to run the debugger, and similar information.

In addition, it will help if you know C++ programming, although most of the work in this book is in C# and VB.NET. A bit of COM and ActiveX familiarity will help, too, although for the uninitiated, I explain most of what you'll need to know of COM and ActiveX for developing add-ins. (The reason you need to know a bit about these technologies is that Visual Studio .NET is a COM automation server, and you'll be developing add-ins as COM in-process servers.)

Some knowledge of Visual Basic is also recommended, either the older versions of the language or the newer VB.NET language, because you'll be writing macros in

VB.NET. But for VB, too, I walk the uninitiated through the essentials. And I devote an entire chapter, Chapter 2, "Just Enough VB.NET," to introducing VB.NET. And rest assured, if you're an expert in C++ or C#, you probably will pick up VB.NET in no time, as the syntax is simple and the language is easy to learn.

I also suggest you learn a bit of C#, if you don't know it already, because many of the examples in this book are in C#. More important, the C# language is, in all likelihood, the future of .NET. Microsoft created C# from the ground floor with .NET in mind (while borrowing a good bit of the syntax from Java and, to an extent, C++). Basically C# is the language of .NET, and so, if you're going to develop for .NET, you will want to at least explore this exciting new language. If you know another programming language (especially C++ or Java), I guarantee you'll be able to pick up the basics of C# in a single day. The syntax is straightforward (it's based on C++ and Java) and the language is easy to learn.

About the Book

This book is divided into four parts:

Part 1: Automating Your Work. In Part I, I focus primarily on macro development, showing you how to automate processes in Visual Studio .NET. Even if your primary interest is add-ins, I encourage you to read this part, as there is a great deal of information overlap between macros and add-ins; most of what you can do in macros you can also do in add-ins. Moreover, in the chapters on add-ins, I assume you've at least given the chapters in Part I a quick read. And note that Part I includes a chapter that introduces the Visual Basic .NET language.

Part 2: Enhancing Visual Studio. Here I introduce the concepts and technology behind add-ins and then take you through the whole world of add-in development.

Part 3: VS.NET and Other Products. In this part I describe how you can use Visual Studio .NET to write add-ins for Microsoft Office, and how you can integrate Microsoft Office products such as Word, Excel, and Outlook directly into Visual Studio .NET. There are two ways of integrating: by automating the office products, using their functionality (such as checking the spelling of source comments or emailing a source file to another developer); and by embedding a spreadsheet or other Office document right into Visual Studio .NET. I wrap up Part 3 with a discussion of other tools useful in automation, such as the Windows Scripting Host and other languages such as Delphi and Python. (Yes, you can write automation macros using nearly any language you want, provided you're comfortable stepping away from the main macro development tools built into Visual Studio .NET.)

Part 4: Deploying and Supercharging. Once you've developed a macro, an add-in for Visual Studio .NET, or an add-in for Microsoft Office, you'll want to know the ins and outs of getting your product onto another computer. Part IV begins by explaining how to get your add-in deployed, and ends by describing how to supercharge Visual Studio .NET.

About the Web Site

In addition to reading this book, I invite you to visit our official Web site at www.wiley.com/compbooks/cogswell, where you'll find :

- Code examples submitted by other readers
- Pages describing additional issues you may encounter, submitted by readers such as yourself
- A discussion forum where you can talk about any issues relating to macros, add-ins, and .NET development in general
- Links to more add-ins and macros that you can download
- And much more.

The Web site was designed to help you, the software developer, get the most possible out of Visual Studio .NET as you develop add-ins and macros. Please join us!