THE EXPERT'S VOICE® IN .NET

Accelerated VB 2005

The fastest path to VB 2005 mastery.

Guy Fouché and Trey Nash

Apress®

Accelerated VB 2005

Guy Fouché and Trey Nash

Apress®

Accelerated VB 2005

ISBN-13 (pbk): 978-1-59059-801-6

ISBN-10 (pbk): 1-59059-801-6

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Lead Editor: James Huddleston
Technical Reviewers: Fabio Claudio Ferracchiati, Dianne Siebold
Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Gennick, Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Jeff Pepper, Paul Sarknas, Dominic Shakeshaft, Jim Sumser, Matt Wade
Project Manager: Sofia Marchant
Copy Edit Manager: Nicole Flores
Copy Editor: Nicole Abramowitz
Assistant Production Director: Kari Brooks-Copony
Production Editor: Kelly Winquist
Compositor: Gina Rexrode
Proofreader: Dan Shaw
Indexer: Becky Hornyak
Artist: April Milne
Cover Designer: Kurt Krames
Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail `orders-ny@springer-sbm.com`, or visit `http://www.springeronline.com`.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail `info@apress.com`, or visit `http://www.apress.com`.

The source code for this book is available to readers at `http://www.apress.com` in the Source Code/Download section.

To Jim & Kay Liegl: for their friendship and the jaunts in the Jeep
To Charlotte Fouché: for her laughter and her compassion toward others
To Frank Reed: for the music and trumpet duets after my lessons were long over
To Jodi Fouché: for her poetry, being my biggest fan, and unequivocal love

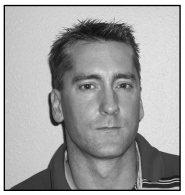
—Guy Fouché

Contents at a Glance

Contents

About the Authors

GUY FOUCHÉ is a business intelligence and decision support system consultant in the Dallas, Texas, area. He has developed a large number of Visual Basic systems in a variety of industries, supporting companies of all shapes and sizes. His VB programming experience dates back to Version 1. Yes, Version 1. Guy spends his evenings playing one of his eight trumpets and expanding his composition skills using the current generation of music technologies. On the weekend, he puts as many miles as he can on his bright yellow Honda F4i sport motorcycle. Guy and Jodi enjoy taking nine-day trips in their Jeep 4×4, taking photographs and writing travelogs along the way. You can view their photography at `http://photography.fouche.ws`.



TREY NASH is a principal software engineer working on security solutions at a market-leading, security-software company. Prior to that, he developed Bluetooth solutions for the release of Microsoft Vista, and he called Macromedia Inc. home for five years before that. At Macromedia, he worked on a cross-product engineering team for several years, designing solutions for a wide range of products throughout the company, including Flash and Fireworks. He specialized in COM/DCOM using C/C++/ATL until the .NET revolution. He's been glued to computers ever since he scored his first, a TI-99/4A, when he was a mere 13 years old. He astounded his parents by turning a childhood obsession into a decent paying career, much to their dismay. Trey received his bachelor of science and his master of engineering degrees in electrical engineering from Texas A&M University. When he's not sitting in front of a computer, you can find him working in his garage, playing his piano, brushing up on a foreign language (Russian and Icelandic are the current favorites), or playing ice hockey.

About the Technical Reviewers

FABIO CLAUDIO FERRACCHIATI is a senior developer for Brain Force (`www.brainforce.com`). A prolific writer on leading-edge technologies, he's contributed to more than a dozen books on .NET, C#, Visual Basic, and ASP.NET. He's a .NET MCSD and lives in Milan, Italy.

DIANNE SIEBOLD is a developer specializing in .NET, C#, ADO, and SQL Server technologies. Currently, she's a programmer writer at Microsoft and specializes in writing white papers, documenting APIs and SDKs, and general commentary on the world of programming. Reach her by e-mail at `dsiebold@earthlink.net`.

Acknowledgments

We offer a huge thank you to everyone at Apress who has had input into these pages!

—Guy Fouché and Trey Nash

In addition, a note of my appreciation goes to:

Nicole Abramowitz, for helping me through the copy-edit process.
Fabio Claudio Ferracchiati, for his technical reviews and bug-stomping skills.
Dianne Siebold, for her technical reviews and contributions to the text.
Sofia Marchant, for managing this project and answering countless questions from a new author.
Dominic Shakeshaft, for giving me my first experiences as a technical reviewer.
Jim Huddleston, for offering me this opportunity and guiding me every step of the way.

—Guy Fouché

Introduction

Visual Basic 2005 (VB 2005) is relatively easy to learn for anyone familiar with another object-oriented language. Even someone familiar with Visual Basic 6.0, who is looking for an object-oriented language, will find VB 2005 easy to pick up. However, though VB 2005, coupled with .NET, provides a quick path for creating simple applications, you still must know a wealth of information and understand how to use it correctly in order to produce sophisticated, robust, fault-tolerant applications. We teach you what you need to know and explain how best to use your knowledge so that you can quickly develop true VB 2005 expertise.

Idioms and design patterns are invaluable for developing and applying expertise, and we show you how to use many of them to create applications that are efficient, robust, fault-tolerant, and exception-safe. Although many are familiar to C++ and Java programmers, some are unique to .NET and the Common Language Runtime (CLR). We show you how to apply these indispensable idioms and design techniques to seamlessly integrate your VB 2005 applications with the .NET runtime, focusing on the new capabilities of VB 2005.

Design patterns document best practices in application design that many different programmers have discovered and rediscovered over time. In fact, .NET itself implements many well-known design patterns. You will see these practices detailed throughout this book. Also, it is important to note that the invaluable tool chest of techniques is evolving constantly.

.NET 2.0 provides a unique and stable cross-platform execution environment. VB 2005 is only one of the languages that targets this powerful runtime. You will find that many of the techniques explored in this book are also applicable to any language that targets the .NET runtime.

As you'll see, it doesn't take years of trial-and-error experience to become a VB 2005 expert. You simply need to learn about the right tools and the correct ways to use them. That's why we wrote this book for you.

About This Book

We assume that you already have a working knowledge of some object-oriented programming language, such as C++, Java, or Visual Basic. If you already know some VB 2005, you may find yourself skimming or even skipping Chapters 1 through 3.

Chapter 1, "VB 2005 Overview," gives a quick glimpse of what a simple VB 2005 application looks like.

Chapter 2, "VB 2005 and the CLR," expands on Chapter 1 and quickly explores the managed environment within which VB 2005 applications run. We introduce you to assemblies, which are the basic building blocks of applications into which VB 2005 code files are compiled. Additionally, you'll see how metadata makes assemblies self-describing.

Chapter 3, "VB Syntax," introduces the VB 2005 language syntax. We introduce you to the two fundamental kinds of types within the CLR: value types and reference types. We also describe namespaces and how you can use them to logically partition types and functionality within your applications.

Chapters 4 through 15 provide in-depth descriptions on how to employ useful idioms, design patterns, and best practices in your VB 2005 programs and designs. We've tried hard to put these chapters in logical order, but occasionally one chapter may reference a technique or topic covered in a later chapter. It is nearly impossible to avoid this situation, but we've tried to minimize it as much as possible.

Chapter 4, "Classes and Structures," provides details about defining types in VB 2005. You'll learn more about value types and reference types in the CLR. We also discuss the inefficiencies inherent in boxing, and discuss object creation, initialization, and destruction.

Chapter 5, "Methods, Properties, and Fields," discusses using methods to add behavior to your types, using properties to enforce encapsulation, and using fields to represent the state of your object. You'll explore method parameter types, overloading, property modifiers, and field initializers.

Chapter 6, "Inheritance, Polymorphism, and Encapsulation," discusses member accessibility and hiding. You'll learn how to choose between delegation and containment, and you'll discover ways to tightly control access to object state.

Chapter 7, "Interfaces," details interfaces and the role they play in the VB 2005 language. Interfaces provide a functionality contract that types may choose to implement. You'll learn the various ways that a type may implement an interface, as well as how the runtime chooses which methods to call when an interface method is called.

Chapter 8, "Operator Overloading," details how you may provide custom functionality for the built-in operators of the VB 2005 language when applied to your own defined types. You'll see how to overload operators responsibly, since not all managed languages that compile code for the CLR are able to use overloaded operators.

Chapter 9, "Exception Handling," shows you the exception-handling capabilities of the VB 2005 language and the CLR. Creating exception-safe and exception-neutral code is tricky in VB 2005, and you'll see that creating fault-tolerant, exception-safe code doesn't require the use of `Try`, `Catch`, or `Finally` constructs at all. We also describe some of the new capabilities within the .NET 2.0 runtime that allow you to create more fault-tolerant code than was possible in .NET 1.1.

Chapter 10, "Working with Strings," describes how strings are a first-class type in the CLR and how to use them effectively in VB 2005. A large portion of the chapter covers the string-formatting capabilities of various types in the .NET Framework and how to make your defined types behave similarly by implementing `IFormattable`. Additionally, we introduce you to the globalization capabilities of the framework and show you how to create custom `CultureInfo` instances for cultures and regions that the .NET Framework doesn't already know about.

Chapter 11, "Arrays and Collections," covers the various array and collection types available in VB 2005. You can create two types of multidimensional arrays, as well as your own collection types, while utilizing collection-utility classes. You'll also learn how to implement `IEnumerable` so that your collection types will work well with `For Each` statements.

Chapter 12, "Delegates and Events," shows you the mechanisms used within VB 2005 to provide callbacks. Historically, all viable frameworks have always provided a mechanism to implement callbacks. VB 2005 goes one step further and encapsulates callbacks into callable objects called *delegates*. Also, you'll see how the .NET Framework builds upon delegates to

provide a publish-subscribe event-notification mechanism, allowing your design to decouple the source of the event from the consumer of the event.

Chapter 13, "Generics," introduces you to probably the most exciting feature added to VB 2005 and the CLR. Using generics, you can provide a shell of functionality within which to define more specific types at run time. Generics are most useful with collection types and provide great efficiency compared to the collections of previous .NET versions.

Chapter 14, "Threading," covers the tasks required in creating multithreaded applications in the VB 2005 managed virtual execution environment. You'll see how delegates, through use of the "I owe you" (IOU) pattern, provide an excellent gateway into the process thread pool. Arguably, synchronization is the most important concept when getting multiple threads to run concurrently. This chapter covers the various synchronization facilities available to your applications.

Chapter 15, "Canonical Forms," is a dissertation on the best design practices for defining new types and how to make them so you can use them naturally and so consumers won't abuse them inadvertently. We touch upon some of these topics in other chapters, but discuss them in detail in this chapter. This chapter concludes with a checklist of items to consider when defining new types using VB 2005.