

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

Accelerated VB 2005

Copyright © 2007 by Guy Fouché and Trey Nash

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

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

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

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 information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

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

About the Authors	xv
About the Technical Reviewers	xvii
Acknowledgments	xix
Introduction	xxi
CHAPTER 1 VB 2005 Overview	1
CHAPTER 2 VB 2005 and the CLR	11
CHAPTER 3 VB Syntax	25
CHAPTER 4 Classes and Structures	49
CHAPTER 5 Methods, Properties, and Fields	85
CHAPTER 6 Inheritance, Polymorphism, and Encapsulation	101
CHAPTER 7 Interfaces	117
CHAPTER 8 Operator Overloading	137
CHAPTER 9 Exception Handling	153
CHAPTER 10 Working with Strings	185
CHAPTER 11 Arrays and Collections	215
CHAPTER 12 Delegates and Events	235
CHAPTER 13 Generics	253
CHAPTER 14 Threading	289
CHAPTER 15 Canonical Forms	335
APPENDIX A Resources	395
APPENDIX B Running the Examples	397
INDEX	399

Contents

About the Authors	xv
About the Technical Reviewers	xvii
Acknowledgments	xix
Introduction	xxi
CHAPTER 1 VB 2005 Overview	1
Differences Between VB 2005, C#, and VB6	1
.NET Runtime	1
VB 2005 and C#	2
VB 2005 and VB6	3
CLR Garbage Collection	3
Common Type System	4
A Simple VB 2005 Program	5
What's New in VB 2005	6
New Commands	6
Generics	7
Operator Overloading	8
My Namespace	8
Summary	9
CHAPTER 2 VB 2005 and the CLR	11
From VB to IL	11
From IL to Platform	13
Understanding Assemblies	14
Assembly Management	18
Private Assemblies	19
Shared Assemblies	19
Loading Assemblies	21
Cross-Language Compatibility	22
Metadata: Better Than COM	22
Reflection	23
Summary	23

CHAPTER 3	VB Syntax	25
	Types and Variables	25
	Strong Typing	25
	Type Categories	27
	Value Types	29
	Reference Types	32
	Type Conversion	33
	Namespaces	40
	Defining Namespaces	41
	Using Namespaces	42
	Statements	43
	Control Flow Constructs	44
	If...Then...Else	44
	Select...Case	45
	Iteration and Looping Constructs	45
	For Each...Next	45
	For...Next	46
	Do While and Do Until	47
	Continue	47
	Summary	48
CHAPTER 4	Classes and Structures	49
	Class Definitions	50
	Constructors	51
	Accessibility	51
	Interfaces	53
	MyBase and MyClass Keywords	54
	NotInheritable Classes	57
	MustInherit Classes	57
	Nested Classes	58
	Item Property Indexers	62
	Partial Classes	64
	Value Type Definitions	64
	Constructors	65
	The Meaning of Me	66
	Finalizers	67
	Interfaces	67
	Boxing and Unboxing	68
	When Boxing Occurs	71
	Efficiency and Confusion	73

System.Object	73
Equality and What It Means	75
The IComparable Interface	75
Creating Objects	75
The New Keyword	75
Shared Constructor	76
Instance Constructor and Creation Ordering	78
Destroying Objects	79
Finalizers	79
Exception Handling	80
Disposable Objects	80
The IDisposable Interface	80
The Using Keyword	82
Summary	84

■ CHAPTER 5 **Methods, Properties, and Fields** 85

Methods	85
Shared Methods	85
Instance Methods	86
Method Parameter Types	87
Method Overloading	90
Overridable and MustOverride Methods	90
A Final Few Words on Overridable Methods	93
Properties	93
Accessors	94
Declaring Properties	94
Fields	96
Field Initialization	98
Summary	100

■ CHAPTER 6 **Inheritance, Polymorphism, and Encapsulation** 101

Inheritance	101
Accessibility of Members	101
Implicit Conversion and a Taste of Polymorphism	102
Member Hiding	104
Inheritance, Containment, and Delegation	107
Choosing Between Interface and Class Inheritance	107
Delegation and Composition vs. Inheritance	109
Encapsulation	111
Summary	115

CHAPTER 7	Interfaces	117
	Interfaces Are Reference Types	117
	Defining Interfaces	118
	What Can Be in an Interface?	120
	Interface Inheritance	121
	Implementing Multiple Interfaces	122
	Hiding Interface Members	123
	Implementing Interfaces in Structures	124
	Beware of Side Effects of Value Types Implementing Interfaces	125
	Using Generics with Interfaces	125
	Using a Generic Interface	125
	Using a Generic Method in an Interface	126
	Contracts	127
	Implementing Contracts with Classes	127
	Implementing Contracts with Interfaces	129
	Choosing Between Interfaces and Classes	130
	Polymorphism with Interfaces	134
	Summary	135
CHAPTER 8	Operator Overloading	137
	Just Because You Can Doesn't Mean You Should	137
	Operators That Can Be Overloaded	137
	Types and Formats of Overloaded Operators	138
	Operators Shouldn't Mutate Their Operands	139
	Does Parameter Order Matter?	140
	Overloading the Addition Operator	141
	Comparison Operators	142
	Conversion Operators	146
	Boolean Operators	148
	Summary	152
CHAPTER 9	Exception Handling	153
	Handling Exceptions	153
	Avoid Using Exceptions to Control Flow	154
	Mechanics of Handling Exceptions in VB 2005	154
	Throwing Exceptions	154
	Unhandled Exceptions in .NET 2.0	155
	Syntax Overview of the Try Statement	155
	Rethrowing Exceptions and Translating Exceptions	157

Exceptions Thrown in Finally Blocks.	159
Exceptions Thrown in Finalizers	160
Exceptions Thrown in Static Constructors.	161
Achieving Exception Neutrality	163
Basic Structure of Exception-Neutral Code.	163
Constrained Execution Regions.	168
Critical Finalizers and SafeHandle.	170
Creating Custom Exception Classes.	174
Working with Allocated Resources and Exceptions.	177
Providing Rollback Behavior	181
Summary	184

■ CHAPTER 10 Working with Strings 185

String Overview	185
String Literals.	186
Format Specifiers and Globalization.	187
Object.ToString(), IFormattable, and CultureInfo.	187
Creating and Registering Custom CultureInfo Types	189
Format Strings	191
Console.WriteLine() and String.Format()	192
Examples of String Formatting in Custom Types	193
ICustomFormatter	195
Comparing Strings.	197
Working with Strings from Outside Sources.	199
StringBuilder.	201
Searching Strings with Regular Expressions	203
Searching with Regular Expressions	203
Searching and Grouping.	204
Replacing Text with Regex	208
Regex Creation Options	211
Summary	213

■ CHAPTER 11 Arrays and Collections. 215

Introduction to Arrays	215
Type Convertibility and Covariance.	216
Sortability and Searchability	217
Synchronization	218
Multidimensional Arrays.	218
Multidimensional Jagged Arrays.	220

Collection Types	221
Comparing ICollection(Of T) with ICollection	222
Collection Synchronization	223
Types That Produce Collections	224
Lists	224
Dictionaries	226
System.Collections.ObjectModel	226
How Iteration Works	229
Summary	233
 CHAPTER 12 Delegates and Events	235
Overview of Delegates	235
Delegate Creation and Use	236
Single Delegate	236
Delegate Chaining	238
Iterating Through Delegate Chains	241
Open-Instance Delegates	242
Strategy Pattern	246
Events	248
Custom Events	250
Summary	252
 CHAPTER 13 Generics	253
Introduction to Generics	253
Efficiency and Type Safety of Generics	254
Generic Type Placeholder Naming Conventions	256
Generic Type Definitions and Constructed Types	256
Generic Classes and Structures	257
Generic Interfaces	258
Generic Methods	259
Generic Delegates	261
Generic Type Conversion	265
Nullable Types	265
Constructed Types Control Accessibility	267
Constraints	267
Constraints on Nonclass Types	271
Generic System Collections	272

Select Problems and Solutions	274
Conversion and Operators Within Generic Types	274
Creating Constructed Types Dynamically	285
Summary	287
CHAPTER 14 Threading	289
Threading in VB 2005 and .NET	289
Starting Threads	290
The IOU Pattern and Asynchronous Method Calls	292
States of a Thread	293
Terminating Threads	295
Halting and Waking Threads	297
Waiting for a Thread to Exit	298
Foreground and Background Threads	299
Thread-Local Storage	300
Synchronizing Threads	303
Lightweight Synchronization with the Interlocked Class	304
Monitor Class	309
Locking Objects	319
Events	323
Win32 Synchronization Objects and WaitHandle	324
Using the ThreadPool	326
Asynchronous Method Calls	327
Timers	332
Summary	333
CHAPTER 15 Canonical Forms	335
Reference-Type Canonical Forms	335
Default to NotInheritable Classes	336
Use the NVI Pattern	336
Is the Object Cloneable?	338
Is the Object Disposable?	345
Does the Object Need a Finalizer?	347
What Does Equality Mean for This Object?	355
If You Override Equals(), Override GetHashCode().	362
Does the Object Support Ordering?	365
Is the Object Formattable?	368

Is the Object Convertible?	371
Prefer Type Safety at All Times	373
Using Immutable Reference Types	378
Value-Type Canonical Forms	381
Override Equals() for Better Performance	382
Do Values of This Type Support Any Interfaces?	387
Implement Type-Safe Forms of Interface Members and Derived Methods	388
Design Checklists	391
Checklist for Reference Types	391
Checklist for Value Types	393
Summary	394
 ■ APPENDIX A Resources	395
Books	395
Articles	396
Web	396
 ■ APPENDIX B Running the Examples	397
Example Types	397
Code Snippets	397
Classes, Structures, and Interfaces	397
Console Applications	397
A Few Words Regarding Modules	398
 ■ INDEX	399

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 4x4, 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.

