SPECIAL EDITION

EXPANDED WITH NEW MATERIAL
ON **.NET 3.0**, **C# 3.0**, AND **LINQ**

THE EXPERT'S VOICE® IN .NET

Pro C# with .NET 3.0

Exploring the .NET universe using curly brackets

Andrew Troelsen

Broaden your C# skills with a **FREE BONUS eBOOK!**
Visit **www.apress.com/promo** for download details.

Apress®

Pro C# with .NET 3.0

Special Edition

Andrew Troelsen

Apress®

Pro C# with .NET 3.0, Special Edition

ISBN-13: 978-1-59059-823-8

ISBN-10: 1-59059-823-7

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

Lead Editor: Ewan Buckingham
Technical Reviewer: Christophe Nasarre
Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Gennick, Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Dominic Shakeshaft, Jim Sumser, Matt Wade
Project Manager: Grace Wong
Copy Edit Manager: Nicole Flores
Copy Editors: Nicole Flores, Ami Knox
Assistant Production Director: Kari Brooks-Copony
Senior Production Editor: Kelly Winquist
Compositor: Dina Quan
Proofreader: Linda Seifert
Indexer: Broccoli Information Management
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. You will need to answer questions pertaining to this book in order to successfully download the code.

I would like to dedicate this book to my mother, Mary Troelsen.
Mom, thanks for all of your support over the years and the years to come.
Oh yeah, and thanks for not busting my chops when I came home
with the red Mohawk.
Luv ya,
Pooch

Contents at a Glance

Contents

PART 3 Programming with .NET Assemblies

PART 4 Programming with the .NET Libraries

PART 5 Web Applications and XML Web Services

PART 6 Programming with .NET 3.0 Extensions

About the Author

ANDREW TROELSEN is a Microsoft MVP (Visual C#) and a partner, trainer, and consultant with Intertech Training (`http://www.Intertech.com`), a .NET and J2EE developer education center. He is the author of numerous books, including *Developer's Workshop to COM and ATL 3.0* (Wordware Publishing, 2000), *COM and .NET Interoperability* (Apress, 2002), *Pro VB 2005 and the .NET 2.0 Platform* (Apress, 2005), and the award-winning *C# and the .NET Platform* (Apress, 2003). Andrew has also authored numerous articles on .NET for MSDN online, DevX.com, and MacTech (where he explored the platform-independent aspects of the .NET platform), and he is a frequent speaker at various .NET conferences and user groups.

Andrew currently lives in Minneapolis, Minnesota, with his wife, Amanda. He spends his free time waiting for the Wild to win the Stanley Cup, the Vikings to win the Super Bowl (which he thinks may never happen), and the Timberwolves to grab numerous NBA championship titles (where he has similar doubts).

About the Technical Reviewer

CHRISTOPHE NASARRE is a development architect for Business Objects, a company that develops desktop and web-based business intelligence solutions. In his spare time, Christophe writes articles for *MSDN* magazine, MSDN/Vista, and ASP Today, and he has been reviewing books on Win32, COM, MFC, .NET, and WPF since 1996.

Acknowledgments

While completing a "special edition" of an existing text is far less painful than authoring a new book beginning with page 1, this manuscript would not have been possible without the assistance and talent offered by numerous individuals. First of all, many thanks to the entire Apress crew. As always, each of you did an outstanding job massaging my raw manuscript into a polished product. Next, I must thank my technical reviewers, who did a truly wonderful job of keeping me honest. Of course, any remaining errors (spelling, coding, or otherwise) that may have snuck into this book are my sole responsibility.

Thanks to my friends and family who (yet again) tolerated my lack of time and sometimes grumpy demeanor. More thanks to my friends and coworkers at Intertech Training. Your support (directly and indirectly) is greatly appreciated. Finally, thanks to my wife, Mandy, and "all the kids" for their love and encouragement.

Introduction

I remember a time years ago when I proposed a book to Apress regarding a forthcoming software SDK code-named Next Generation Windows Services (NGWS). As you may be aware, NGWS eventually became what we now know as the .NET platform. My research of the C# programming language and the .NET platform took place in parallel with the authoring of the initial manuscript. It was a fantastic project; however, I must confess that it was more than a bit nerve-racking writing about a technology that was undergoing drastic changes over the course of its development. Thankfully, after many sleepless nights, the first edition of *C# and the .NET Platform* was published in conjunction with the release of .NET 1.0 Beta 2, circa the summer of 2001.

Since that point, I have been extremely happy and grateful to see that this text was very well received by the press and, most important, by readers. Over the years it was nominated as a Jolt Award finalist (I lost . . . crap!) and for the 2003 Referenceware Excellence Award in the programming book category (I won? Cool!).

The second edition of this text (*C# and the .NET Platform, Second Edition*) provided me the opportunity to expand upon the existing content with regard to version 1.1 of the .NET platform. Although the second edition of the book did offer a number of new topics, a number of chapters and examples were unable to make it into the final product.

Once the text entered its third edition (*Pro C# 2005 and the .NET 2.0 Platform*), the manuscript was updated to account for the numerous bells and whistles brought about by .NET 2.0 (new C# programming constructs, generics, updates to core APIs, etc.), and it included new material that had long been written but not yet published (such as content on the common intermediate language [CIL] and dynamic assemblies, and expanded ASP.NET coverage).

In this special edition of the text, I have added six new chapters dedicated to the new programming APIs brought about with the release of .NET 3.0. Over these chapters, you will come to understand the role of Windows Presentation Foundation (WPF), Windows Communication Foundation (WCF), and Windows Workflow Foundation (WF). As well, this new text provides details regarding the forthcoming C# 3.0 programming language and LINQ programming technologies (LINQ to SQL and LINQ to XML). While C# 3.0 and LINQ are currently beta technologies, the final two chapters of this text will provide a solid road map for the changes to come.

As with the earlier editions, this special edition presents the C# programming language and .NET base class libraries using a friendly and approachable tone. I have never understood the need some technical authors have to spit out prose that reads more like a GRE vocabulary study guide than a readable book. As well, this new edition remains focused on providing you with the information you need to build software solutions today, rather than spending too much time examining esoteric details that few individuals will ever actually care about.

We're a Team, You and I

Technology authors write for a demanding group of people (I should know—I'm one of them). You know that building software solutions using any platform is extremely detailed and is very specific to your department, company, client base, and subject matter. Perhaps you work in the electronic publishing industry, develop systems for the state or local government, or work at NASA or a branch of the military. Speaking for myself, I have developed children's educational software, various n-tier

systems, and numerous projects within the medical and financial industries. The chances are almost 100 percent that the code you write at your place of employment has little to do with the code I write at mine (unless we happened to work together previously!).

Therefore, in this book, I have deliberately chosen to avoid creating examples that tie the example code to a specific industry or vein of programming. I choose to explain C#, OOP, the CLR, and the .NET 2.0/3.0 base class libraries using industry-agnostic examples. Rather than having every blessed example fill a grid with data, calculate payroll, or whatnot, I'll stick to subject matter we can all relate to: automobiles (with some geometric structures and employees thrown in for good measure). And that's where you come in.

My job is to explain the C# programming language and the core aspects of the .NET platform the best I possibly can. As well, I will do everything I can to equip you with the tools and strategies you need to continue your studies at this book's conclusion.

Your job is to take this information and apply it to your specific programming assignments. I obviously understand that your projects most likely don't revolve around automobiles with pet names, but that's what applied knowledge is all about! Rest assured, once you understand the concepts presented within this text, you will be in a perfect position to build .NET solutions that map to your own unique programming environment.

An Overview of This Book

Pro C# with .NET 3.0, Special Edition is logically divided into six distinct parts, each of which contains some number of chapters that somehow "belong together." If you've read the third edition of this text (*Pro C# 2005 and the .NET 2.0 Platform*), you will notice that the first 25 chapters of this special edition are identical (beyond a number of errata integrations) to those. However, Part 6 of this book (which is devoted to building .NET 3.0/C# 3.0 and LINQ applications) is indeed entirely new. This being said, here is a part-by-part and chapter-by-chapter breakdown of the text.

Part 1: Introducing C# and the .NET Platform

The purpose of Part 1 is to acclimate you to the core aspects of the .NET platform, the .NET type system, and various development tools (many of which are open source) used during the construction of .NET applications. Along the way, you will also check out some basic details of the C# programming language.

Chapter 1: The Philosophy of .NET

This first chapter functions as the backbone for the remainder of the text. We begin by examining the world of traditional Windows development and uncover the shortcomings with the previous state of affairs. The primary goal of this chapter, however, is to acquaint you with a number of .NET-centric building blocks, such as the common language runtime (CLR), Common Type System (CTS), Common Language Specification (CLS), and the base class libraries. Also, you will also take an initial look at the C# programming language and the .NET assembly format, and you'll examine the platform-independent nature of the .NET platform and the role of the Common Language Infrastructure (CLI).

Chapter 2: Building C# Applications

The goal of this chapter is to introduce you to the process of compiling and debugging C# source code files using various tools and techniques. First, you will learn how to make use of the command-line compiler (`csc.exe`) and C# response files. Over the remainder of the chapter,

you will examine numerous IDEs, including TextPad, SharpDevelop, Visual C# 2005 Express, and (of course) Visual Studio 2005. As well, you will be exposed to a number of open source tools (Vil, NAnt, NDoc, etc.), which any .NET developer should have in his or her back pocket.

Part 2: The C# Programming Language

This part explores all the gory details of the C# programming language, including the new syntactical constructs introduced with .NET 2.0. As well, Part 2 exposes you to each member of the CTS (classes, interfaces, structures, enumerations, and delegates) and the construction of generic types.

Chapter 3: C# Language Fundamentals

This chapter examines the core constructs of the C# programming language. Here you will come to understand basic class construction techniques, the distinction between value types and reference types, boxing and unboxing, and the role of everybody's favorite base class, `System.Object`. Also, Chapter 3 illustrates how the .NET platform puts a spin on various commonplace programming constructs, such as enumerations, arrays, and string processing. Finally, this chapter examines a number of 2.0-specific topics, including "nullable data types."

Chapter 4: Object-Oriented Programming with C#

The role of Chapter 4 is to examine the details of how C# 2.0 accounts for each "pillar" of OOP: encapsulation, inheritance, and polymorphism. Once you have examined the keywords and the syntax used to build class hierarchies, you will then look at the role of XML code comments.

Chapter 5: Understanding Object Lifetime

This chapter examines how the CLR manages memory using the .NET garbage collector. Here you will come to understand the role of application roots, object generations, and the `System.GC` type. Once you understand the basics, the remainder of this chapter covers the topics of "disposable objects" (via the `IDisposable` interface) and the finalization process (via the `System.Object.Finalize()` method).

Chapter 6: Understanding Structured Exception Handling

The point of this chapter is to discuss how to handle runtime anomalies in your code base through the use of structured exception handling. Not only will you learn about the C# keywords that allow you to handle such problems (`try`, `catch`, `throw`, and `finally`), but you will also come to understand the distinction between application-level and system-level exceptions. In addition, this chapter examines various tools within Visual Studio 2005 that allow you to debug the exceptions that have escaped your view.

Chapter 7: Interfaces and Collections

The material in this chapter builds upon your understanding of object-based development by covering the topic of interface-based programming. Here you will learn how to define types that support multiple behaviors, how to discover these behaviors at runtime, and how to selectively hide particular behaviors using *explicit interface implementation*. To showcase the usefulness of interface types, the remainder of this chapter examines the `System.Collections` namespace.

Chapter 8: Callback Interfaces, Delegates, and Events

The purpose of Chapter 8 is to demystify the delegate type. Simply put, a .NET *delegate* is an object that "points" to other methods in your application. Using this pattern, you are able to build systems that allow multiple objects to engage in a two-way conversation. After you have examined the use of .NET delegates (including numerous 2.0-specific features such as anonymous methods), you will then be introduced to the C# `event` keyword, which is used to simplify the manipulation of raw delegate programming.

Chapter 9: Advanced C# Type Construction Techniques

This chapter deepens your understanding of the C# programming language by introducing a number of advanced programming techniques. For example, you will learn how to overload operators and create custom conversion routines (both implicit and explicit), build type indexers, and manipulate C-style pointers within a `*.cs` code file.

Chapter 10: Understanding Generics

As of .NET 2.0, the C# programming language has been enhanced to support a new feature of the CTS termed *generics*. As you will see, generic programming greatly enhances application performance and type safety. Not only will you explore various generic types within the `System.Collections.Generic` namespace, but you will also learn how to build your own generic methods and types (with and without constraints).

Part 3: Programming with .NET Assemblies

Part 3 dives into the details of the .NET assembly format. Not only will you learn how to deploy and configure .NET code libraries, but you will also come to understand the internal composition of a .NET binary image. This part also explains the role of .NET attributes and the construction of mutilthreaded applications. Later chapters examine some fairly low-level details (such as object context) and the syntax and semantics of CIL.

Chapter 11: Introducing .NET Assemblies

From a very high level, *assembly* is the term used to describe a managed `*.dll` or `*.exe` file. However, the true story of .NET assemblies is far richer than that. Here you will learn the distinction between single-file and multifile assemblies, and how to build and deploy each entity. You'll examine how private and shared assemblies may be configured using XML-based `*.config` files and publisher policy assemblies. Along the way, you will investigate the internal structure of the global assembly cache (GAC) and the role of the .NET Framework 2.0 configuration utility.

Chapter 12: Type Reflection, Late Binding, and Attribute-Based Programming

Chapter 12 continues our examination of .NET assemblies by checking out the process of runtime type discovery via the `System.Reflection` namespace. Using these types, you are able to build applications that can read an assembly's metadata on the fly. You will learn how to dynamically activate and manipulate types at runtime using *late binding*. The final topic of this chapter explores the role of .NET attributes (both standard and custom). To illustrate the usefulness of each of these topics, the chapter concludes with the construction of an extendable Windows Forms application.

Chapter 13: Processes, AppDomains, Contexts, and CLR Hosts

Now that you have a solid understanding of assemblies, this chapter dives much deeper into the composition of a loaded .NET executable. The first goal is to illustrate the relationship between processes, application domains, and contextual boundaries. Once these terms have been qualified, you will then understand exactly how the CLR itself is hosted by the Windows operating system and deepen your understanding of `mscoree.dll`. The information presented here is a perfect lead-in to Chapter 14.

Chapter 14: Building Multithreaded Applications

This chapter examines how to build multithreaded applications and illustrates a number of techniques you can use to author thread-safe code. The chapter opens by revisiting the .NET delegate type in order to understand a delegate's intrinsic support for asynchronous method invocations. Next, you will investigate the types within the `System.Threading` namespace. You will look at numerous types (`Thread`, `ThreadStart`, etc.) that allow you to easily create additional threads of execution.

Chapter 15: Understanding CIL and the Role of Dynamic Assemblies

The goal of this chapter is twofold. In the first half (more or less), you will examine the syntax and semantics of CIL in much greater detail than in previous chapters. The remainder of this chapter covers the role of the `System.Reflection.Emit` namespace. Using these types, you are able to build software that is capable of generating .NET assemblies in memory at runtime. Formally speaking, assemblies defined and executed in memory are termed *dynamic assemblies*.

Part 4: Programming with the .NET Libraries

By this point in the text, you have a solid handle on the C# language and the details of the .NET assembly format. Part 4 leverages your newfound knowledge by exploring a number of namespaces within the base class libraries, including file I/O, the .NET remoting layer, Windows Forms development, and database access using ADO.NET.

Chapter 16: The System.IO Namespace

As you can gather from its name, the `System.IO` namespace allows you to interact with a machine's file and directory structure. Over the course of this chapter, you will learn how to programmatically create (and destroy) a directory system as well as move data into and out of various streams (file based, string based, memory based, etc.).

Chapter 17: Understanding Object Serialization

This chapter examines the object serialization services of the .NET platform. Simply put, *serialization* allows you to persist the state of an object (or a set of related objects) into a stream for later use. *Deserialization* (as you might expect) is the process of plucking an object from the stream into memory for consumption by your application. Once you understand the basics, you will then learn how to customize the serialization process via the `ISerializable` interface and a set of new attributes introduced with .NET 2.0.

Chapter 18: The .NET Remoting Layer

Contrary to popular belief, XML web services are not the only way to build distributed applications under the .NET platform. Here you will learn about the .NET remoting layer. As you will see, the CLR supports the ability to easily pass objects between application and machine boundaries using marshal-by-value (MBV) and marshal-by-reference (MBR) semantics. Along the way, you will learn how to alter the runtime behavior of a distributed .NET application in a declarative manner using XML configuration files.

Chapter 19: Building a Better Window with System.Windows.Forms

This chapter begins your examination of the `System.Windows.Forms` namespace. Here you will learn the details of building traditional desktop GUI applications that support menu systems, toolbars, and status bars. As you would hope, various design-time aspects of Visual Studio 2005 will be examined, as well as a number of .NET 2.0 Windows Forms types (`MenuStrip`, `ToolStrip`, etc.).

Chapter 20: Rendering Graphical Data with GDI+

This chapter covers how to dynamically render graphical data in the Windows Forms environment. In addition to discussing how to manipulate fonts, colors, geometric images, and image files, this chapter examines hit testing and GUI-based drag-and-drop techniques. You will learn about the new .NET resource format, which as you may suspect by this point in the text is based on XML data representation.

Chapter 21: Programming with Windows Forms Controls

This final Windows-centric chapter will examine numerous GUI widgets that ship with the .NET Framework 2.0. Not only will you learn how to program against various Windows Forms controls, but you will also learn about dialog box development and Form inheritance. As well, this chapter examines how to build *custom* Windows Forms controls that integrate into the IDE.

Chapter 22: Database Access with ADO.NET

ADO.NET is the data access API of the .NET platform. As you will see, you are able to interact with the types of ADO.NET using a connected and disconnected layer. Over the course of this chapter, you will have the chance to work with both modes of ADO.NET, and you'll learn about several new .NET 2.0 ADO.NET topics, including the data provider factory model, connection string builders, and asynchronous database access.

Part 5: Web Applications and XML Web Services

Part 5 is devoted to the construction of ASP.NET web applications and XML web services. As you will see in the first two chapters of this section, ASP.NET 2.0 is a major upgrade from ASP.NET 1.*x* and includes numerous new bells and whistles.

Chapter 23: ASP.NET 2.0 Web Pages and Web Controls

This chapter begins your study of web technologies supported under the .NET platform using ASP.NET. As you will see, server-side scripting code is now replaced with "real" object-oriented languages (such as C#, VB .NET, and the like). This chapter will introduce you to key ASP.NET topics such as working with (or without) code-behind files, the role of ASP.NET web controls, validations controls, and interacting with the new "master page" model provided by ASP.NET 2.0.

Chapter 24: ASP.NET 2.0 Web Applications

This chapter extends your current understanding of ASP.NET by examining various ways to handle state management under .NET. Like classic ASP, ASP.NET allows you to easily create cookies, as well as application-level and session-level variables. However, ASP.NET also introduces a new state management technique: the application cache. Once you have looked at the numerous ways to handle state with ASP.NET, you will then come to learn the role of the `System.HttpApplication` base class (lurking within the `Global.asax` file) and how to dynamically alter the runtime behavior of your web application using the `Web.config` file.

Chapter 25: Understanding XML Web Services

This chapter examines the role of .NET XML web services. Simply put, a *web service* is an assembly that is activated using standard HTTP requests. The beauty of this approach is the fact that HTTP is the one wire protocol almost universal in its acceptance, and it is therefore an excellent choice for building platform- and language-neutral distributed systems. You will also check out numerous surrounding technologies (WSDL, SOAP, and UDDI) that enable a web service and external client to communicate in harmony.

Part 6: Programming with .NET 3.0 Extensions

The bulk of Part 6 is devoted to the new APIs introduced with .NET 3.0: WPF, WCF, and WF. In addition, you will examine the details of C# 3.0 and the LINQ programming model, both of which (at the time of this writing) are in beta.

Chapter 26: Establishing a .NET 3.0 Programming Environment

Before you can build .NET 3.0–aware software, or explore C# 3.0 and LINQ development, your very first task is to install a number of freely downloadable SDKs and Visual Studio 2005 CTP modules. In this chapter, you'll be provided with a blow-by-blow account of setting up a .NET 3.0/C# 3.0/LINQ development machine, and you'll learn how to repair a critical installation bug along the way.

Chapter 27: Introducing Windows Presentation Foundation

Windows Presentation Foundation, or simply WPF, is a brand-new model for building .NET desktop applications. This chapter takes you behind the scenes of this new API by examining the problems WPF attempts to solve, the role of desktop markup (aka XAML), and the use of code-behind files. Along the way, you will come to understand the major services found within WPF (graphical rendering, animations, etc.), examine the new control programming model, and be introduced to the concept of an XBAP application.

Chapter 28: Introducing Windows Communication Foundation

Windows Communication Foundation (WCF) is a .NET 3.0 programming API specifically geared toward the development of distributed applications. As you will learn in this chapter, WCF's major goal is to integrate a number of previously independent APIs (COM+, MSMQ, .NET remoting, XML web services, etc.) into a single unified (and extendable) object model. Although WCF is indeed a new API, you will be happy to know that your current knowledge of the .NET remoting layer (Chapter 18) and XML web services (Chapter 25) will greatly increase your understanding of what is taking place behind the curtains.

Chapter 29: Introducing Windows Workflow Foundation

Windows Workflow Foundation (WF) is the final major component of .NET 3.0. This chapter begins by defining exactly what workflows are and where you may wish to make use of them in your programming assignments. Then this chapter examines a number of WF activities, the role of the WF runtime engine, and how to make use of the Visual Studio 2005 "Orcas" workflow designer.

Chapter 30: C# 3.0 Language Features

At the time of this writing, C# 3.0 is still a beta product. However, once you have installed the correct development tools (described in Chapter 26), you are able to explore all of the new constructs you will find in the future release of .NET's flagship programming language. In this chapter, you will learn about implicitly typed local variables, object initialization syntax, extension methods, anonymous types, and the role of lambda expressions. The information presented here will be a perfect (and, quite frankly, mandatory) foundation for the final chapter of this book.

Chapter 31: An Introduction to LINQ

The final chapter of this text dives into the details of the LINQ programming model, which will be released in conjunction with C# 3.0, sometime in the middle of 2007 (or so we hope!). Simply put, LINQ attempts to provide a single symmetrical model to access "data," regardless of its location. As you will see, LINQ allows you to build query expressions (which have been purposely designed to look like SQL queries) to access and manipulate data located in arrays, collections, relational databases, and XML documents.

What's Included in the Free Bonus Download

This special edition includes bonus materials with additional content in PDF. This content includes:

- A carefully selected sampler of chapters from 18 other *Pro* and *Expert* books from the Apress library, including advanced books about ASP.NET 2.0 and SQL Server 2005. These chapters total more than 1,500 information-rich pages in eBook form, with complementary examples at `http://www.apress.com`.
- A full selection of our .NET 2.0 road maps that illustrate how you, the reader, can link together Apress books to chart a path for custom learning.

Obtaining This Book's Source Code

All of the code examples contained within this book (minus small code snippets here and there) are available for free and immediate download from the Source Code/Download area of the Apress website. Simply navigate to `http://www.apress.com`, select the Source Code/Download link, and look up this title by name. Once you are on the "homepage" for *Pro C# with .NET 3.0, Special Edition*, you may download a self-extracting `*.zip` file. After you unzip the contents, you will find that the code has been logically divided by chapter.

Do be aware that Source Code notes like the following in the chapters are your cue that the example under discussion may be loaded into Visual Studio 2005 for further examination and modification:

Source Code This is a source code note referring you to a specific directory!

Simply double-click the `*.sln` file found in the correct subdirectory to load the project into Visual Studio 2005.

Obtaining Updates for This Book

As you read through this text, you may find an occasional grammatical or code error (although I sure hope not). If this is the case, my apologies. Being human, I am sure that a glitch or two may be present, despite my best efforts. You can obtain the current errata list from the Apress website (located once again on the "homepage" for this book) as well as information on how to notify me of any errors you might find.

Contacting Me

If you have any questions regarding this book's source code, are in need of clarification for a given example, or simply wish to offer your thoughts regarding the .NET platform, feel free to drop me a line at the following e-mail address (to ensure your messages don't end up in my junk mail folder, please include "C# SpEd" in the Subject line somewhere): `atroelsen@Intertech.com`.

Please understand that I will do my best to get back to you in a timely fashion; however, like yourself, I get busy from time to time. If I don't respond within a week or two, do know I am not trying to be a jerk or don't care to talk to you. I'm just busy (or, if I'm lucky, on vacation somewhere).

So, then! Thanks for buying this text (or at least looking at it in the bookstore while you try to decide if you will buy it). I hope you enjoy reading this book and putting your newfound knowledge to good use.

Take care,
Andrew Troelsen