

**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](http://www.apress.com/promo) for download details.

**apress®**

# Pro C# with .NET 3.0

## Special Edition



Andrew Troelsen

## **Pro C# with .NET 3.0, Special Edition**

**Copyright © 2007 by Andrew Troelsen**

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: 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

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: 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](mailto: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](mailto: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. 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

About the Author .....	xxxvii
About the Technical Reviewer .....	xxxix
Acknowledgments .....	xli
Introduction .....	xliii

## PART 1 ■ ■ ■ Introducing C# and the .NET Platform

■ CHAPTER 1	The Philosophy of .NET .....	3
■ CHAPTER 2	Building C# Applications .....	33

## PART 2 ■ ■ ■ The C# Programming Language

■ CHAPTER 3	C# Language Fundamentals .....	65
■ CHAPTER 4	Object-Oriented Programming with C# 2.0 .....	139
■ CHAPTER 5	Understanding Object Lifetime .....	179
■ CHAPTER 6	Understanding Structured Exception Handling .....	197
■ CHAPTER 7	Interfaces and Collections .....	221
■ CHAPTER 8	Callback Interfaces, Delegates, and Events .....	255
■ CHAPTER 9	Advanced C# Type Construction Techniques .....	289
■ CHAPTER 10	Understanding Generics .....	321

## PART 3 ■ ■ ■ Programming with .NET Assemblies

■ CHAPTER 11	Introducing .NET Assemblies .....	347
■ CHAPTER 12	Type Reflection, Late Binding, and Attribute-Based Programming .....	391
■ CHAPTER 13	Processes, AppDomains, Contexts, and CLR Hosts .....	425
■ CHAPTER 14	Building Multithreaded Applications .....	449
■ CHAPTER 15	Understanding CIL and the Role of Dynamic Assemblies .....	477

## PART 4 ■ ■ ■ Programming with the .NET Libraries

■ CHAPTER 16	The System.IO Namespace .....	517
■ CHAPTER 17	Understanding Object Serialization .....	545
■ CHAPTER 18	The .NET Remoting Layer .....	565
■ CHAPTER 19	Building a Better Window with System.Windows.Forms .....	605
■ CHAPTER 20	Rendering Graphical Data with GDI+ .....	649
■ CHAPTER 21	Programming with Windows Forms Controls .....	699
■ CHAPTER 22	Database Access with ADO.NET .....	759

## PART 5 ■ ■ ■ Web Applications and XML Web Services

■ CHAPTER 23	ASP.NET 2.0 Web Pages and Web Controls .....	829
■ CHAPTER 24	ASP.NET 2.0 Web Applications .....	889
■ CHAPTER 25	Understanding XML Web Services .....	919

## PART 6 ■ ■ ■ Programming with .NET 3.0 Extensions

■ CHAPTER 26	Establishing a .NET 3.0 Programming Environment .....	957
■ CHAPTER 27	Introducing Windows Presentation Foundation .....	969
■ CHAPTER 28	Introducing Windows Communication Foundation .....	1021
■ CHAPTER 29	Introducing Windows Workflow Foundation .....	1047
■ CHAPTER 30	C# 3.0 Language Features .....	1075
■ CHAPTER 31	An Introduction to LINQ .....	1107
■ INDEX	.....	1151



# Contents

About the Author .....	xxxvii
About the Technical Reviewer .....	xxxix
Acknowledgments .....	xli
Introduction .....	xliii

## PART 1 ■ ■ ■ Introducing C# and the .NET Platform

■ CHAPTER 1	<b>The Philosophy of .NET .....</b>	<b>3</b>
	Understanding the Previous State of Affairs .....	3
	Life As a C/Win32 API Programmer .....	3
	Life As a C++/MFC Programmer .....	4
	Life As a Visual Basic 6.0 Programmer .....	4
	Life As a Java/J2EE Programmer .....	4
	Life As a COM Programmer .....	5
	Life As a Windows DNA Programmer .....	5
	The .NET Solution .....	6
	Introducing the Building Blocks of the .NET Platform	
	(the CLR, CTS, and CLS) .....	6
	The Role of the Base Class Libraries .....	7
	What C# Brings to the Table .....	7
	Additional .NET-Aware Programming Languages .....	8
	Life in a Multilanguage World .....	9
	An Overview of .NET Assemblies .....	10
	Single-File and Multifile Assemblies .....	11
	The Role of the Common Intermediate Language .....	11
	Benefits of CIL .....	13
	Compiling CIL to Platform-Specific Instructions .....	14
	The Role of .NET Type Metadata .....	14
	The Role of the Assembly Manifest .....	15
	Understanding the Common Type System .....	15
	CTS Class Types .....	16
	CTS Structure Types .....	16
	CTS Interface Types .....	17



CTS Enumeration Types .....	17
CTS Delegate Types .....	17
CTS Type Members .....	18
Intrinsic CTS Data Types .....	18
Understanding the Common Language Specification .....	19
Ensuring CLS Compliance .....	20
Understanding the Common Language Runtime .....	20
The Assembly/Namespace/Type Distinction .....	22
Accessing a Namespace Programmatically .....	24
Referencing External Assemblies .....	25
Using ildasm.exe .....	26
Viewing CIL Code .....	27
Viewing Type Metadata .....	28
Viewing Assembly Metadata .....	28
Deploying the .NET Runtime .....	29
The Platform-Independent Nature of .NET .....	29
Summary .....	31

<b>CHAPTER 2</b>	<b>Building C# Applications .....</b>	<b>33</b>
	Installing the .NET Framework 2.0 SDK .....	33
	The C# Command-Line Compiler (csc.exe) .....	34
	Configuring the C# Command-Line Compiler .....	34
	Configuring Additional .NET Command-Line Tools .....	35
	Building C# Applications Using csc.exe .....	36
	Referencing External Assemblies .....	37
	Compiling Multiple Source Files with csc.exe .....	38
	Referencing Multiple External Assemblies .....	39
	Working with csc.exe Response Files .....	39
	The Default Response File (csc.rsp) .....	40
	The Command-Line Debugger (cordbg.exe) .....	40
	Debugging at the Command Line .....	41
	Building .NET Applications Using TextPad .....	41
	Enabling C# Keyword Coloring .....	42
	Configuring the *.cs File Filter .....	43
	Hooking Into csc.exe .....	43
	Associating Run Commands with Menu Items .....	44
	Enabling C# Code Snippets .....	45
	Building .NET Applications Using SharpDevelop .....	46
	Learning the Lay of the Land: SharpDevelop .....	47
	The Project and Classes Scouts .....	47
	The Assembly Scout .....	48
	Windows Forms Designers .....	49
	Building .NET Applications Using Visual C# 2005 Express .....	50

The Big Kahuna: Building .NET Applications Using Visual Studio 2005 . . . . .	51
Learning the Lay of the Land: Visual Studio 2005 . . . . .	52
The Solution Explorer Utility . . . . .	52
The Class View Utility . . . . .	53
The Code Definition Window . . . . .	54
The Object Browser Utility . . . . .	54
Integrated Support for Code Refactoring . . . . .	55
Code Expansions and Surround with Technology . . . . .	57
The Visual Class Designer . . . . .	57
Object Test Bench . . . . .	60
The Integrated Help System . . . . .	60
A Partial Catalogue of Additional .NET Development Tools . . . . .	61
Summary . . . . .	62

## PART 2 ■ ■ ■ The C# Programming Language

CHAPTER 3	<b>C# Language Fundamentals . . . . .</b>	<b>65</b>
	The Anatomy of a Simple C# Program . . . . .	65
	Variations on the Main() Method . . . . .	66
	Processing Command-Line Arguments . . . . .	67
	Specifying Command-Line Arguments with Visual Studio 2005 . . . . .	68
	An Interesting Aside: The System.Environment Class . . . . .	68
	Defining Classes and Creating Objects . . . . .	69
	The Role of Constructors . . . . .	70
	Is That a Memory Leak? . . . . .	72
	Defining an “Application Object” . . . . .	72
	The System.Console Class . . . . .	73
	Basic Input and Output with the Console Class . . . . .	73
	Formatting Console Output . . . . .	74
	.NET String Formatting Flags . . . . .	75
	Establishing Member Visibility . . . . .	76
	Establishing Type Visibility . . . . .	78
	Default Values of Class Member Variables . . . . .	78
	Default Values and Local Variables . . . . .	79
	Member Variable Initialization Syntax . . . . .	79
	Defining Constant Data . . . . .	80
	Referencing Constant Data . . . . .	81
	Defining Read-Only Fields . . . . .	82
	Static Read-Only Fields . . . . .	83
	Understanding the static Keyword . . . . .	83
	Static Methods . . . . .	84
	Static Data . . . . .	84

Static Constructors .....	86
Static Classes .....	88
Method Parameter Modifiers .....	89
The Default Parameter-Passing Behavior .....	89
The out Modifier .....	90
The ref Modifier .....	90
The params Modifier .....	91
Iteration Constructs .....	92
The for Loop .....	92
The foreach Loop .....	93
The while and do/while Looping Constructs .....	93
Decision Constructs and the Relational/Equality Operators .....	94
The if/else Statement .....	94
The switch Statement .....	95
Understanding Value Types and Reference Types .....	96
Value Types, References Types, and the Assignment Operator .....	97
Value Types Containing Reference Types .....	99
Passing Reference Types by Value .....	101
Passing Reference Types by Reference .....	102
Value and Reference Types: Final Details .....	103
Understanding Boxing and Unboxing Operations .....	104
Some Practical (Un)Boxing Examples .....	105
Unboxing Custom Value Types .....	106
Working with .NET Enumerations .....	107
The System.Enum Base Class .....	109
The Master Class: System.Object .....	110
The Default Behavior of System.Object .....	112
Overriding Some Default Behaviors of System.Object .....	113
Overriding System.Object.ToString() .....	114
Overriding System.Object.Equals() .....	114
Overriding System.Object.GetHashCode() .....	115
Testing the Overridden Members .....	116
Static Members of System.Object .....	117
The System Data Types (and C# Shorthand Notation) .....	117
Experimenting with Numerical Data Types .....	120
Members of System.Boolean .....	120
Members of System.Char .....	121
Parsing Values from String Data .....	121
System.DateTime and System.TimeSpan .....	122
The System.String Data Type .....	123
Basic String Operations .....	123
Escape Characters .....	124
Working with C# Verbatim Strings .....	125
The Role of System.Text.StringBuilder .....	126

.NET Array Types . . . . .	127
Arrays As Parameters (and Return Values) . . . . .	128
Working with Multidimensional Arrays . . . . .	128
The System.Array Base Class . . . . .	130
Understanding C# Nullable Types . . . . .	131
Working with Nullable Types . . . . .	132
The ?? Operator . . . . .	133
Defining Custom Namespaces . . . . .	133
A Type's Fully Qualified Name . . . . .	134
Defining using Aliases . . . . .	136
Creating Nested Namespaces . . . . .	137
The "Default Namespace" of Visual Studio 2005 . . . . .	138
Summary . . . . .	138

## ■ CHAPTER 4    **Object-Oriented Programming with C# 2.0** . . . . . 139

Understanding the C# Class Type . . . . .	139
Understanding Method Overloading . . . . .	141
Self-Reference in C# Using this . . . . .	142
Defining the Public Interface of a Class . . . . .	143
Reviewing the Pillars of OOP . . . . .	145
Encapsulation . . . . .	145
Inheritance . . . . .	145
Polymorphism . . . . .	146
The First Pillar: C#'s Encapsulation Services . . . . .	147
Enforcing Encapsulation Using Traditional Accessors and Mutators . . . . .	148
Another Form of Encapsulation: Class Properties . . . . .	149
Internal Representation of C# Properties . . . . .	151
Controlling Visibility Levels of Property get/set Statements . . . . .	153
Read-Only and Write-Only Properties . . . . .	153
Static Properties . . . . .	153
The Second Pillar: C#'s Inheritance Support . . . . .	154
Controlling Base Class Creation with base . . . . .	156
Regarding Multiple Base Classes . . . . .	157
Keeping Family Secrets: The protected Keyword . . . . .	157
Preventing Inheritance: Sealed Classes . . . . .	158
Programming for Containment/Delegation . . . . .	159
Nested Type Definitions . . . . .	160
The Third Pillar: C#'s Polymorphic Support . . . . .	162
The virtual and override Keywords . . . . .	162
Revisiting the sealed Keyword . . . . .	164
Understanding Abstract Classes . . . . .	164
Enforcing Polymorphic Activity: Abstract Methods . . . . .	165
Member Hiding . . . . .	169

C# Casting Rules . . . . .	170
Determining the “Type of” Employee . . . . .	172
Numerical Casts . . . . .	172
Understanding C# Partial Types . . . . .	173
Documenting C# Source Code via XML . . . . .	174
XML Code Comment Format Characters . . . . .	176
Transforming XML Code Comments . . . . .	177
Summary . . . . .	177
 <b>■ CHAPTER 5   Understanding Object Lifetime . . . . .</b>	 179
Classes, Objects, and References . . . . .	179
The Basics of Object Lifetime . . . . .	180
The CIL of new . . . . .	181
The Role of Application Roots . . . . .	182
Understanding Object Generations . . . . .	184
The System.GC Type . . . . .	185
Forcing a Garbage Collection . . . . .	186
Building Finalizable Objects . . . . .	188
Overriding System.Object.Finalize() . . . . .	189
Detailing the Finalization Process . . . . .	191
Building Disposable Objects . . . . .	191
Reusing the C# using Keyword . . . . .	192
Building Finalizable and Disposable Types . . . . .	194
A Formalized Disposal Pattern . . . . .	194
Summary . . . . .	196
 <b>■ CHAPTER 6   Understanding Structured Exception Handling . . . . .</b>	 197
Ode to Errors, Bugs, and Exceptions . . . . .	197
The Role of .NET Exception Handling . . . . .	198
The Atoms of .NET Exception Handling . . . . .	199
The System.Exception Base Class . . . . .	199
The Simplest Possible Example . . . . .	200
Throwing a Generic Exception . . . . .	202
Catching Exceptions . . . . .	203
Configuring the State of an Exception . . . . .	204
The TargetSite Property . . . . .	204
The StackTrace Property . . . . .	205
The HelpLink Property . . . . .	206
The Data Property . . . . .	206
System-Level Exceptions (System.SystemException) . . . . .	208

Application-Level Exceptions (System.ApplicationException) . . . . .	208
Building Custom Exceptions, Take One . . . . .	209
Building Custom Exceptions, Take Two . . . . .	210
Building Custom Exceptions, Take Three . . . . .	210
Processing Multiple Exceptions . . . . .	212
Generic catch Statements . . . . .	213
Rethrowing Exceptions . . . . .	214
Inner Exceptions . . . . .	214
The Finally Block . . . . .	215
Who Is Throwing What? . . . . .	216
The Result of Unhandled Exception . . . . .	217
Debugging Unhandled Exceptions Using Visual Studio 2005 . . . . .	218
Summary . . . . .	219

## ■ CHAPTER 7      **Interfaces and Collections** . . . . . 221

Defining Interfaces in C# . . . . .	221
Implementing an Interface in C# . . . . .	222
Contrasting Interfaces to Abstract Base Classes . . . . .	224
Invoking Interface Members at the Object Level . . . . .	224
Obtaining Interface References: The as Keyword . . . . .	225
Obtaining Interface References: The is Keyword . . . . .	225
Interfaces As Parameters . . . . .	226
Interfaces As Return Values . . . . .	228
Arrays of Interface Types . . . . .	228
Understanding Explicit Interface Implementation . . . . .	229
Resolving Name Clashes . . . . .	231
Building Interface Hierarchies . . . . .	232
Interfaces with Multiple Base Interfaces . . . . .	233
Implementing Interfaces Using Visual Studio 2005 . . . . .	234
Building Enumerable Types (IEnumerable and IEnumerator) . . . . .	235
Understanding C# Iterator Methods . . . . .	237
Building Cloneable Objects (ICloneable) . . . . .	238
A More Elaborate Cloning Example . . . . .	240
Building Comparable Objects (IComparable) . . . . .	242
Specifying Multiple Sort Orders (IComparer) . . . . .	245
Custom Properties, Custom Sort Types . . . . .	246
The Interfaces of the System.Collections Namespace . . . . .	247
The Role of ICollection . . . . .	248
The Role of IDictionary . . . . .	248
The Role of IDictionaryEnumerator . . . . .	249
The Role of IList . . . . .	249

The Class Types of System.Collections . . . . .	249
Working with the ArrayList Type . . . . .	250
Working with the Queue Type . . . . .	251
Working with the Stack Type . . . . .	252
System.Collections.Specialized Namespace . . . . .	253
Summary . . . . .	254

## ■ CHAPTER 8      **Callback Interfaces, Delegates, and Events** . . . . . 255

Understanding Callback Interfaces . . . . .	255
Understanding the .NET Delegate Type . . . . .	259
Defining a Delegate in C# . . . . .	259
The System.MulticastDelegate and System.Delegate Base Classes . . . . .	262
The Simplest Possible Delegate Example . . . . .	263
Investigating a Delegate Object . . . . .	264
Retrofitting the Car Type with Delegates . . . . .	266
Enabling Multicasting . . . . .	268
A More Elaborate Delegate Example . . . . .	270
Delegates As Parameters . . . . .	271
Analyzing the Delegation Code . . . . .	274
Understanding Delegate Covariance . . . . .	275
Understanding C# Events . . . . .	277
Events Under the Hood . . . . .	278
Listening to Incoming Events . . . . .	279
Simplifying Event Registration Using Visual Studio 2005 . . . . .	280
A “Prim-and-Proper” Event . . . . .	281
Understanding C# Anonymous Methods . . . . .	282
Accessing “Outer” Variables . . . . .	284
C# Method Group Conversions . . . . .	285
Summary . . . . .	286

## ■ CHAPTER 9      **Advanced C# Type Construction Techniques** . . . . . 289

Building a Custom Indexer . . . . .	289
A Variation of the Garage Indexer . . . . .	291
Internal Representation of Type Indexers . . . . .	292
Indexers: Final Details . . . . .	293
Understanding Operator Overloading . . . . .	293
Overloading Binary Operators . . . . .	294
And What of the += and -- Operators? . . . . .	295
Overloading Unary Operators . . . . .	296
Overloading Equality Operators . . . . .	296
Overloading Comparison Operators . . . . .	297

The Internal Representation of Overloaded Operators . . . . .	298
Interacting with Overloaded Operators from Overloaded Operator–Challenged Languages . . . . .	299
Final Thoughts Regarding Operator Overloading . . . . .	301
Understanding Custom Type Conversions . . . . .	301
Recall: Numerical Conversions . . . . .	301
Recall: Conversions Among Related Class Types . . . . .	301
Creating Custom Conversion Routines . . . . .	302
Additional Explicit Conversions for the Square Type . . . . .	304
Defining Implicit Conversion Routines . . . . .	304
The Internal Representation of Custom Conversion Routines . . . . .	306
The Advanced Keywords of C# . . . . .	307
The checked Keyword . . . . .	307
The unchecked Keyword . . . . .	309
Working with Pointer Types . . . . .	310
The sizeof Keyword . . . . .	316
C# Preprocessor Directives . . . . .	316
Specifying Code Regions . . . . .	317
Conditional Code Compilation . . . . .	318
Summary . . . . .	319

## CHAPTER 10    **Understanding Generics** . . . . . 321

Revisiting the Boxing, Unboxing, and System.Object Relationship . . . . .	321
The Problem with (Un)Boxing Operations . . . . .	322
Type Safety and Strongly Typed Collections . . . . .	323
Boxing Issues and Strongly Typed Collections . . . . .	325
The System.Collections.Generic Namespace . . . . .	326
Examining the List<T> Type . . . . .	327
Creating Generic Methods . . . . .	329
Omission of Type Parameters . . . . .	330
Creating Generic Structures (or Classes) . . . . .	332
The default Keyword in Generic Code . . . . .	333
Creating a Custom Generic Collection . . . . .	334
Constraining Type Parameters Using where . . . . .	335
The Lack of Operator Constraints . . . . .	338
Creating Generic Base Classes . . . . .	339
Creating Generic Interfaces . . . . .	340
Creating Generic Delegates . . . . .	341
Simulating Generic Delegates Under .NET 1.1 . . . . .	342
A Brief Word Regarding Nested Delegates . . . . .	343
Summary . . . . .	343



## PART 3 ■ ■ ■ Programming with .NET Assemblies

■ CHAPTER 11	<b>Introducing .NET Assemblies</b> .....	347
	The Role of .NET Assemblies .....	347
	Assemblies Promote Code Reuse .....	347
	Assemblies Establish a Type Boundary .....	348
	Assemblies Are Versionable Units .....	348
	Assemblies Are Self-Describing .....	348
	Assemblies Are Configurable .....	348
	Understanding the Format of a .NET Assembly .....	349
	The Win32 File Header .....	349
	The CLR File Header .....	350
	CIL Code, Type Metadata, and the Assembly Manifest .....	351
	Optional Assembly Resources .....	352
	Single-File and Multifile Assemblies .....	352
	Building and Consuming a Single-File Assembly .....	354
	Exploring the Manifest .....	357
	Exploring the CIL .....	358
	Exploring the Type Metadata .....	358
	Building a C# Client Application .....	359
	Building a Visual Basic .NET Client Application .....	360
	Cross-Language Inheritance in Action .....	362
	Building and Consuming a Multifile Assembly .....	362
	Exploring the ufo.netmodule File .....	364
	Exploring the airvehicles.dll File .....	364
	Consuming a Multifile Assembly .....	364
	Understanding Private Assemblies .....	365
	The Identity of a Private Assembly .....	365
	Understanding the Probing Process .....	366
	Configuring Private Assemblies .....	366
	Configuration Files and Visual Studio 2005 .....	368
	Introducing the .NET Framework 2.0 Configuration Utility .....	369
	Understanding Shared Assemblies .....	371
	Understanding Strong Names .....	371
	Strongly Naming CarLibrary.dll .....	373
	Assigning Strong Names Using Visual Studio 2005 .....	374
	Installing/Removing Shared Assemblies to/from the GAC .....	374
	The Role of Delayed Signing .....	375
	Consuming a Shared Assembly .....	376
	Exploring the Manifest of SharedCarLibClient .....	378
	Configuring Shared Assemblies .....	378
	Freezing the Current Shared Assembly .....	379
	Building Shared Assembly Version 2.0.0.0 .....	379

Dynamically Redirecting to Specific Versions of a Shared Assembly. . . . .	381
Revisiting the .NET Framework 2.0 Configuration Utility. . . . .	382
Investigating the Internal Composition of the GAC. . . . .	382
Understanding Publisher Policy Assemblies. . . . .	384
Disabling Publisher Policy. . . . .	385
Understanding the <codeBase> Element. . . . .	385
The System.Configuration Namespace. . . . .	387
The Machine Configuration File. . . . .	388
The Assembly Binding “Big Picture”. . . . .	388
Summary. . . . .	389

## CHAPTER 12    **Type Reflection, Late Binding, and Attribute-Based Programming**

The Necessity of Type Metadata. . . . .	391
Viewing (Partial) Metadata for the EngineState Enumeration. . . . .	392
Viewing (Partial) Metadata for the Car Type. . . . .	393
Examining a TypeRef. . . . .	394
Documenting the Defining Assembly. . . . .	394
Documenting Referenced Assemblies. . . . .	394
Documenting String Literals. . . . .	395
Understanding Reflection. . . . .	395
The System.Type Class. . . . .	396
Obtaining a Type Reference Using System.Object.GetType(). . . . .	397
Obtaining a Type Reference Using System.Type.GetType(). . . . .	397
Obtaining a Type Reference Using typeof(). . . . .	398
Building a Custom Metadata Viewer. . . . .	398
Reflecting on Methods. . . . .	398
Reflecting on Fields and Properties. . . . .	398
Reflecting on Implemented Interfaces. . . . .	399
Displaying Various Odds and Ends. . . . .	399
Implementing Main(). . . . .	399
Reflecting on Method Parameters and Return Values. . . . .	401
Dynamically Loading Assemblies. . . . .	402
Reflecting on Shared Assemblies. . . . .	404
Understanding Late Binding. . . . .	406
The System.Activator Class. . . . .	406
Invoking Methods with No Parameters. . . . .	407
Invoking Methods with Parameters. . . . .	407
Understanding Attributed Programming. . . . .	408
Attribute Consumers. . . . .	408
Applying Predefined Attributes in C#. . . . .	409
Specifying Constructor Parameters for Attributes. . . . .	411

The Obsolete Attribute in Action . . . . .	411
C# Attribute Shorthand Notation . . . . .	411
Building Custom Attributes . . . . .	412
Applying Custom Attributes . . . . .	413
Restricting Attribute Usage . . . . .	414
Assembly-Level (and Module-Level) Attributes . . . . .	415
The Visual Studio 2005 AssemblyInfo.cs File . . . . .	415
Reflecting on Attributes Using Early Binding . . . . .	416
Reflecting on Attributes Using Late Binding . . . . .	417
Putting Reflection, Late Binding, and Custom Attributes in Perspective . . . . .	418
Building an Extendable Application . . . . .	419
Building CommonSnappableTypes.dll . . . . .	419
Building the C# Snap-In . . . . .	420
Building the Visual Basic .NET Snap-In . . . . .	420
Building an Extendable Windows Forms Application . . . . .	421
Summary . . . . .	424

## ■ CHAPTER 13 Processes, AppDomains, Contexts, and CLR Hosts . . . . . 425

Reviewing Traditional Win32 Processes . . . . .	425
An Overview of Threads . . . . .	426
Interacting with Processes Under the .NET Platform . . . . .	427
Enumerating Running Processes . . . . .	429
Investigating a Specific Process . . . . .	430
Investigating a Process's Thread Set . . . . .	430
Investigating a Process's Module Set . . . . .	432
Starting and Stopping Processes Programmatically . . . . .	434
Understanding .NET Application Domains . . . . .	435
Enumerating a Process's AppDomains . . . . .	436
Programmatically Creating New AppDomains . . . . .	437
Programmatically Unloading AppDomains . . . . .	439
Understanding Object Context Boundaries . . . . .	440
Context-Agile and Context-Bound Types . . . . .	441
Defining a Context-Bound Object . . . . .	442
Inspecting an Object's Context . . . . .	442
Summarizing Processes, AppDomains, and Context . . . . .	444
Hosting the Common Language Runtime . . . . .	444
Side-by-Side Execution of the CLR . . . . .	445
Loading a Specific Version of the CLR . . . . .	446
Additional CLR Hosts . . . . .	447
Summary . . . . .	447

<b>CHAPTER 14</b>	<b>Building Multithreaded Applications</b>	449
	The Process/AppDomain/Context/Thread Relationship	449
	The Problem of Concurrency and the Role of Thread	
	Synchronization	450
	A Brief Review of the .NET Delegate	451
	The Asynchronous Nature of Delegates	453
	The BeginInvoke() and EndInvoke() Methods	453
	The System.IAsyncResult Interface	453
	Invoking a Method Asynchronously	454
	Synchronizing the Calling Thread	455
	The Role of the AsyncCallback Delegate	456
	The Role of the AsyncResult Class	457
	Passing and Receiving Custom State Data	458
	The System.Threading Namespace	459
	The System.Threading.Thread Class	460
	Obtaining Statistics About the Current Thread	460
	The Name Property	461
	The Priority Property	462
	Programmatically Creating Secondary Threads	462
	Working with the ThreadStart Delegate	463
	Working with the ParameterizedThreadStart Delegate	465
	Foreground Threads and Background Threads	466
	The Issue of Concurrency	467
	Synchronization Using the C# lock Keyword	469
	Synchronization Using the System.Threading.Monitor Type	471
	Synchronization Using the System.Threading.Interlocked Type	471
	Synchronization Using the [Synchronization] Attribute	472
	Programming with Timer Callbacks	473
	Understanding the CLR ThreadPool	475
	Summary	476
 <b>CHAPTER 15</b>	 <b>Understanding CIL and the Role of Dynamic Assemblies</b>	 477
	Reflecting on the Nature of CIL Programming	477
	Examining CIL Directives, Attributes, and Opcodes	478
	The Role of CIL Directives	478
	The Role of CIL Attributes	479
	The Role of CIL Opcodes	479
	The CIL Opcode/CIL Mnemonic Distinction	479
	Pushing and Popping: The Stack-Based Nature of CIL	480

Understanding Round-Trip Engineering. . . . .	481
The Role of CIL Code Labels. . . . .	483
Interacting with CIL: Modifying an *.il File. . . . .	484
Compiling CIL Code Using ilasm.exe . . . . .	485
Compiling CIL Code Using SharpDevelop . . . . .	486
Compiling CIL Code Using ILIDE#. . . . .	486
The Role of peverify.exe . . . . .	487
Understanding CIL Directives and Attributes. . . . .	487
Specifying Externally Referenced Assemblies in CIL . . . . .	488
Defining the Current Assembly in CIL . . . . .	488
Defining Namespaces in CIL. . . . .	489
Defining Class Types in CIL . . . . .	489
Defining and Implementing Interfaces in CIL . . . . .	490
Defining Structures in CIL. . . . .	491
Defining Enums in CIL. . . . .	491
Compiling the CILTypes.il file . . . . .	491
.NET Base Class Library, C#, and CIL Data Type Mappings . . . . .	492
Defining Type Members in CIL . . . . .	493
Defining Field Data in CIL . . . . .	493
Defining Type Constructors in CIL . . . . .	494
Defining Properties in CIL . . . . .	494
Defining Member Parameters. . . . .	495
Examining CIL Opcodes. . . . .	495
Considering the .maxstack Directive . . . . .	497
Declaring Local Variables in CIL. . . . .	498
Mapping Parameters to Local Variables in CIL. . . . .	498
The Hidden this Reference . . . . .	499
Representing Iteration Constructs in CIL. . . . .	499
Building a .NET Assembly with CIL . . . . .	500
Building CILCars.dll . . . . .	500
Building CILCarClient.exe . . . . .	503
Understanding Dynamic Assemblies . . . . .	504
Exploring the System.Reflection.Emit Namespace. . . . .	505
The Role of the System.Reflection.Emit.ILGenerator . . . . .	506
Emitting a Dynamic Assembly . . . . .	506
Emitting the Assembly and Module Set. . . . .	508
The Role of the ModuleBuilder Type . . . . .	509
Emitting the HelloClass Type and the String Member Variable. . . . .	510
Emitting the Constructors . . . . .	511
Emitting the HelloWorld() Method. . . . .	512
Using the Dynamically Generated Assembly. . . . .	512
A Brief Word Regarding System.CodeDOM. . . . .	513
Summary . . . . .	514

## PART 4 ■ ■ ■ Programming with the .NET Libraries

<b>CHAPTER 16</b>	<b>The System.IO Namespace</b>	517
	Exploring the System.IO Namespace	517
	The Directory(Info) and File(Info) Types	518
	The Abstract FileSystemInfo Base Class	519
	Working with the DirectoryInfo Type	519
	The FileAttributes Enumeration	521
	Enumerating Files with the DirectoryInfo Type	521
	Creating Subdirectories with the DirectoryInfo Type	522
	Working with the Directory Type	523
	Working with the DriveInfo Class Type	524
	Working with the FileInfo Class	525
	The FileInfo.Create() Method	526
	The FileInfo.Open() Method	527
	The FileInfo.OpenRead() and FileInfo.OpenWrite() Methods	528
	The FileInfo.OpenText() Method	528
	The FileInfo.CreateText() and FileInfo.AppendText() Methods	528
	Working with the File Type	529
	New .NET 2.0 File Members	530
	The Abstract Stream Class	531
	Working with FileStreams	532
	Working with StreamWriters and StreamReaders	533
	Writing to a Text File	534
	Reading from a Text File	535
	Directly Creating StreamWriter/StreamReader Types	536
	Working with StringWriters and StringReaders	536
	Working with BinaryWriters and BinaryReaders	538
	Programmatically “Watching” Files	540
	Performing Asynchronous File I/O	542
	Summary	543
<b>CHAPTER 17</b>	<b>Understanding Object Serialization</b>	545
	Understanding Object Serialization	545
	The Role of Object Graphs	546
	Configuring Objects for Serialization	547
	Public Fields, Private Fields, and Public Properties	548
	Choosing a Serialization Formatter	548
	The IFormatter and IRemotingFormatting Interfaces	549
	Type Fidelity Among the Formatters	550

Serializing Objects Using the BinaryFormatter . . . . .	550
Deserializing Objects Using the BinaryFormatter . . . . .	551
Serializing Objects Using the SoapFormatter . . . . .	552
Serializing Objects Using the XmlSerializer . . . . .	553
Controlling the Generated XML Data . . . . .	553
Persisting Collections of Objects . . . . .	555
Customizing the Serialization Process . . . . .	556
A Deeper Look at Object Serialization . . . . .	557
Customizing Serialization Using ISerializable . . . . .	558
Customizing Serialization Using Attributes . . . . .	560
Versioning Serializable Objects . . . . .	561
Summary . . . . .	563

<b>CHAPTER 18</b>	<b>The .NET Remoting Layer . . . . .</b>	<b>565</b>
	Defining .NET Remoting . . . . .	565
	The .NET Remoting Namespaces . . . . .	566
	Understanding the .NET Remoting Framework . . . . .	567
	Understanding Proxies and Messages . . . . .	567
	Understanding Channels . . . . .	568
	Revisiting the Role of .NET Formatters . . . . .	569
	All Together Now! . . . . .	569
	A Brief Word Regarding Extending the Default Plumbing . . . . .	570
	Terms of the .NET Remoting Trade . . . . .	570
	Object Marshaling Choices: MBR or MBV? . . . . .	570
	Activation Choices for MBR Types: WKO or CAO? . . . . .	572
	Stateful Configuration of WKO Types: Singleton or Single Call? . . . . .	573
	Summarizing the Traits of MBR Object Types . . . . .	574
	Basic Deployment of a .NET Remoting Project . . . . .	574
	Building Your First Distributed Application . . . . .	575
	Building the General Assembly . . . . .	575
	Building the Server Assembly . . . . .	576
	Building the SimpleRemoteObjectClient.exe Assembly . . . . .	577
	Testing the Remoting Application . . . . .	578
	Understanding the ChannelServices Type . . . . .	578
	Understanding the RemotingConfiguration Type . . . . .	580
	Revisiting the Activation Mode of WKO Types . . . . .	581
	Deploying the Server to a Remote Machine . . . . .	582
	Leveraging the TCP Channel . . . . .	582
	A Brief Word Regarding the IpcChannel . . . . .	583
	Remoting Configuration Files . . . . .	584
	Building Server-Side *.config Files . . . . .	584
	Building Client-Side *.config Files . . . . .	585

Working with MBV Objects .....	586
Building the General Assembly .....	586
Building the Server Assembly .....	587
Building the Client Assembly .....	588
Understanding Client-Activated Objects .....	590
The Lease-Based Lifetime of CAO/WKO-Singleton Objects .....	592
The Default Leasing Behavior .....	592
Altering the Default Lease Characteristics .....	594
Server-Side Lease Adjustment .....	595
Client-Side Lease Adjustment .....	596
Server-Side (and Client-Side) Lease Sponsorship .....	596
Alternative Hosts for Remote Objects .....	597
Hosting Remote Objects Using a Windows Service .....	597
Hosting Remote Objects Using IIS .....	601
Asynchronous Remoting .....	602
The Role of the [OneWay] Attribute .....	604
Summary .....	604

## ■ CHAPTER 19    **Building a Better Window with System.Windows.Forms** .. 605

Overview of the System.Windows.Forms Namespace .....	605
Working with the Windows Forms Types .....	606
Building a Main Window by Hand .....	607
Honoring the Separation of Concerns .....	608
The Role of the Application Class .....	609
Fun with the Application Class .....	609
The System.EventHandler Delegate .....	611
The Anatomy of a Form .....	611
The Functionality of the Control Class .....	612
Fun with the Control Class .....	614
Responding to the MouseEventArgs .....	615
Determining Which Mouse Button Was Clicked .....	616
Responding to Keyboard Events .....	617
The Functionality of the Form Class .....	618
The Life Cycle of a Form Type .....	619
Building Windows Applications with Visual Studio 2005 .....	621
Enabling the Deprecated Controls .....	623
Dissecting a Visual Studio 2005 Windows Forms Project .....	623
Handling Events at Design Time .....	625
The Program Class .....	625
Autoreferenced Assemblies .....	626



Working with MenuStrips and ContextMenuStrips . . . . .	626
Adding a TextBox to the MenuStrip . . . . .	629
Creating a Context Menu . . . . .	630
Checking Menu Items . . . . .	632
Working with StatusStrips . . . . .	633
Designing the Menu System . . . . .	634
Designing the StatusStrip . . . . .	634
Working with the Timer Type . . . . .	637
Toggling the Display . . . . .	638
Displaying the Menu Selection Prompts . . . . .	639
Establishing a “Ready” State . . . . .	639
Working with ToolStrips . . . . .	639
Working with ToolStripContainers . . . . .	643
Building an MDI Application . . . . .	646
Building the Parent Form . . . . .	646
Building the Child Form . . . . .	647
Spawning Child Windows . . . . .	647
Summary . . . . .	648

## ■ CHAPTER 20    **Rendering Graphical Data with GDI+** . . . . . 649

A Survey of the GDI+ Namespaces . . . . .	649
An Overview of the System.Drawing Namespace . . . . .	650
The System.Drawing Utility Types . . . . .	651
The Point(F) Type . . . . .	651
The Rectangle(F) Type . . . . .	652
The Region Class . . . . .	653
Understanding the Graphics Class . . . . .	653
Understanding Paint Sessions . . . . .	655
Invalidating the Form’s Client Area . . . . .	656
Obtaining a Graphics Object Outside of a Paint Event Handler . . . . .	657
Regarding the Disposal of a Graphics Object . . . . .	658
The GDI+ Coordinate Systems . . . . .	659
The Default Unit of Measure . . . . .	660
Specifying an Alternative Unit of Measure . . . . .	661
Specifying an Alternative Point of Origin . . . . .	662
Defining a Color Value . . . . .	663
The ColorDialog Class . . . . .	664
Manipulating Fonts . . . . .	665
Working with Font Families . . . . .	666
Working with Font Faces and Font Sizes . . . . .	667
Enumerating Installed Fonts . . . . .	669
The FontDialog Class . . . . .	671
Survey of the System.Drawing.Drawing2D Namespace . . . . .	672

Working with Pens . . . . .	673
Working with Pen Caps . . . . .	675
Working with Brushes . . . . .	677
Working with HatchBrushes . . . . .	678
Working with TextureBrushes . . . . .	679
Working with LinearGradientBrushes . . . . .	681
Rendering Images . . . . .	682
Dragging and Hit Testing the PictureBox Control . . . . .	684
Hit Testing Rendered Images . . . . .	687
Hit Testing Nonrectangular Images . . . . .	688
Understanding the .NET Resource Format . . . . .	691
The System.Resources Namespace . . . . .	691
Programmatically Creating an *.resx File . . . . .	692
Building the *.resources File . . . . .	693
Binding the *.resources File into a .NET Assembly . . . . .	693
Working with ResourceWriters . . . . .	694
Generating Resources Using Visual Studio 2005 . . . . .	694
Programmatically Reading Resources . . . . .	697
Summary . . . . .	698

## ■ CHAPTER 21    **Programming with Windows Forms Controls** . . . . . 699

The World of Windows Forms Controls . . . . .	699
Adding Controls to Forms by Hand . . . . .	700
The Control.ControlCollection Type . . . . .	701
Adding Controls to Forms Using Visual Studio 2005 . . . . .	702
Working with the Basic Controls . . . . .	703
Fun with Labels . . . . .	704
Fun with TextBoxes . . . . .	705
Fun with MaskedTextBoxes . . . . .	707
Fun with Buttons . . . . .	709
Fun with CheckBoxes, RadioButtons, and GroupBoxes . . . . .	711
Fun with CheckedListBoxes . . . . .	714
Fun with ListBoxes . . . . .	715
Fun with ComboBoxes . . . . .	716
Configuring the Tab Order . . . . .	718
The Tab Order Wizard . . . . .	718
Setting the Form's Default Input Button . . . . .	719
Working with More Exotic Controls . . . . .	719
Fun with MonthCalendars . . . . .	719
Fun with ToolTips . . . . .	721
Fun with TabControls . . . . .	722
Fun with TrackBars . . . . .	724
Fun with Panels . . . . .	726

Fun with the UpDown Controls . . . . .	727
Fun with ErrorProviders . . . . .	729
Fun with TreeViews . . . . .	731
Fun with WebBrowsers . . . . .	736
Building Custom Windows Forms Controls . . . . .	737
Creating the Images . . . . .	739
Building the Design-Time UI . . . . .	739
Implementing the Core CarControl . . . . .	740
Defining the Custom Events . . . . .	741
Defining the Custom Properties . . . . .	741
Controlling the Animation . . . . .	743
Rendering the Pet Name . . . . .	743
Testing the CarControl Type . . . . .	743
Building a Custom CarControl Form Host . . . . .	744
The Role of the System.ComponentModel Namespace . . . . .	746
Enhancing the Design-Time Appearance of CarControl . . . . .	746
Defining a Default Property and Default Event . . . . .	748
Specifying a Custom Toolbox Bitmap . . . . .	748
Building Custom Dialog Boxes . . . . .	750
The DialogResult Property . . . . .	751
Understanding Form Inheritance . . . . .	752
Dynamically Positioning Windows Forms Controls . . . . .	754
The Anchor Property . . . . .	755
The Dock Property . . . . .	755
Table and Flow Layout . . . . .	756
Summary . . . . .	758

<b>■ CHAPTER 22 Database Access with ADO.NET . . . . .</b>	<b>759</b>
A High-Level Definition of ADO.NET . . . . .	759
The Two Faces of ADO.NET . . . . .	760
Understanding ADO.NET Data Providers . . . . .	760
Microsoft-Supplied Data Providers . . . . .	762
Select Third-Party Data Providers . . . . .	763
Additional ADO.NET Namespaces . . . . .	763
The System.Data Types . . . . .	764
The Role of the IDbConnection Interface . . . . .	765
The Role of the IDbTransaction Interface . . . . .	765
The Role of the IDbCommand Interface . . . . .	766
The Role of the IDbDataParameter and IDataParameter Interfaces . . . . .	766
The Role of the IDbDataAdapter and IDataAdapter Interfaces . . . . .	767
The Role of the IDataReader and IDataRecord Interfaces . . . . .	767
Abstracting Data Providers Using Interfaces . . . . .	768
Increasing Flexibility Using Application Configuration Files . . . . .	769

The .NET 2.0 Provider Factory Model .....	770
Registered Data Provider Factories .....	771
A Complete Data Provider Factory Example .....	772
The <connectionStrings> Element .....	774
Installing the Cars Database .....	775
Connecting to the Cars Database from Visual Studio 2005 .....	776
Understanding the Connected Layer of ADO.NET .....	778
Working with Connection Objects .....	779
Working with .NET 2.0 ConnectionStringBuilder .....	780
Working with Command Objects .....	781
Working with Data Readers .....	782
Obtaining Multiple Result Sets Using a Data Reader .....	784
Modifying Tables Using Command Objects .....	784
Inserting New Records .....	786
Deleting Existing Records .....	787
Updating Existing Records .....	787
Working with Parameterized Command Objects .....	788
Specifying Parameters Using the DbParameter Type .....	788
Executing a Stored Procedure Using DbCommand .....	790
Asynchronous Data Access Under .NET 2.0 .....	792
Understanding the Disconnected Layer of ADO.NET .....	793
Understanding the Role of the DataSet .....	794
Members of the DataSet .....	794
Working with DataColumns .....	796
Building a DataColumn .....	797
Enabling Autoincrementing Fields .....	797
Adding a DataColumn to a DataTable .....	798
Working with DataRows .....	798
Understanding the DataRow.RowState Property .....	799
Working with DataTables .....	800
Working with .NET 2.0 DataTableReaders .....	802
Persisting DataSets (and DataTables) As XML .....	803
Binding DataTables to User Interfaces .....	804
Programmatically Deleting Rows .....	806
Applying Filters and Sort Orders .....	807
Updating Rows .....	809
Working with the DataView Type .....	810
Working with Data Adapters .....	811
Filling a DataSet Using a Data Adapter .....	812
Mapping Database Names to Friendly Names .....	813
Updating a Database Using Data Adapter Objects .....	813
Setting the InsertCommand Property .....	814
Setting the UpdateCommand Property .....	815
Setting the DeleteCommand Property .....	815

Autogenerating SQL Commands Using CommandBuilder Types . . . . .	816
Multitabled DataSets and DataRelation Objects . . . . .	817
Navigating Between Related Tables . . . . .	820
We're Off to See the (Data) Wizard. . . . .	822
Strongly Typed DataSets. . . . .	823
The Autogenerated Data Component. . . . .	824
Summary . . . . .	825

## PART 5 ■ ■ ■ Web Applications and XML Web Services

■ CHAPTER 23 <b>ASP.NET 2.0 Web Pages and Web Controls</b> . . . . .	829
The Role of HTTP . . . . .	829
Understanding Web Applications and Web Servers . . . . .	830
Working with IIS Virtual Directories . . . . .	831
The ASP.NET 2.0 Development Server . . . . .	832
The Role of HTML. . . . .	832
HTML Document Structure . . . . .	833
HTML Form Development . . . . .	833
Building an HTML-Based User Interface . . . . .	834
The Role of Client-Side Scripting . . . . .	836
A Client-Side Scripting Example. . . . .	836
Validating the default.htm Form Data . . . . .	837
Submitting the Form Data (GET and POST). . . . .	837
Building a Classic ASP Page . . . . .	838
Responding to POST Submissions . . . . .	839
Problems with Classic ASP . . . . .	840
Major Benefits of ASP.NET 1.x . . . . .	840
Major Enhancements of ASP.NET 2.0 . . . . .	841
The ASP.NET 2.0 Namespaces. . . . .	841
The ASP.NET Web Page Code Model . . . . .	842
Working with the Single-File Page Model . . . . .	843
Working with the Code-Behind Page Model . . . . .	847
Details of an ASP.NET Website Directory Structure . . . . .	851
The Role of the Bin Folder. . . . .	852
The Role of the App_Code Folder. . . . .	853
The ASP.NET 2.0 Page Compilation Cycle. . . . .	853
Compilation Cycle for Single-File Pages . . . . .	853
Compilation Cycle for Multifile Pages . . . . .	854
The Inheritance Chain of the Page Type . . . . .	855
The System.Web.UI.Page Type . . . . .	856

Interacting with the Incoming HTTP Request . . . . .	857
Obtaining Browser Statistics . . . . .	858
Access to Incoming Form Data . . . . .	858
The IsPostBack Property . . . . .	859
Interacting with the Outgoing HTTP Response . . . . .	859
Emitting HTML Content . . . . .	860
Redirecting Users . . . . .	861
The Life Cycle of an ASP.NET Web Page . . . . .	861
The Role of the AutoEventWireUp Attribute . . . . .	862
The Error Event . . . . .	863
Understanding the Nature of Web Controls . . . . .	865
Qualifying Server-Side Event Handling . . . . .	865
The AutoPostBack Property . . . . .	866
The System.Web.UI.Control Type . . . . .	866
Enumerating Contained Controls . . . . .	867
Dynamically Adding (and Removing) Controls . . . . .	869
Key Members of the System.Web.UI.WebControls.WebControl Type . . . . .	870
Categories of ASP.NET Web Controls . . . . .	871
A Brief Word Regarding System.Web.UI.HtmlControls . . . . .	871
Building a Simple ASP.NET 2.0 Website . . . . .	872
Working with Master Pages . . . . .	872
Defining the Default.aspx Content Page . . . . .	875
Designing the Inventory Content Page . . . . .	877
Designing the Build a Car Content Page . . . . .	881
The Role of the Validation Controls . . . . .	883
The RequiredFieldValidator . . . . .	885
The RegularExpressionValidator . . . . .	886
The RangeValidator . . . . .	886
The CompareValidator . . . . .	887
Creating Validation Summaries . . . . .	887
Summary . . . . .	888

## CHAPTER 24    **ASP.NET 2.0 Web Applications** . . . . . 889

The Issue of State . . . . .	889
ASP.NET State Management Techniques . . . . .	891
Understanding the Role of ASP.NET View State . . . . .	891
Demonstrating View State . . . . .	892
Adding Custom View State Data . . . . .	893
A Brief Word Regarding Control State . . . . .	894
The Role of the Global.asax File . . . . .	894
The Global Last Chance Exception Event Handler . . . . .	896
The HttpApplication Base Class . . . . .	897

Understanding the Application/Session Distinction .....	897
Maintaining Application-Level State Data .....	898
Modifying Application Data .....	899
Handling Web Application Shutdown .....	900
Working with the Application Cache .....	901
Fun with Data Caching .....	901
Modifying the *.aspx File. ....	903
Maintaining Session Data .....	906
Additional Members of HttpSessionState .....	908
Understanding Cookies .....	909
Creating Cookies .....	909
Reading Incoming Cookie Data .....	911
Configuring Your ASP.NET Web Application Using Web.config .....	912
Enabling Tracing via <trace> .....	913
Customizing Error Output via <customErrors> .....	914
Options for Storing State via <sessionState> .....	915
The ASP.NET 2.0 Site Administration Utility .....	916
Configuration Inheritance .....	917
Summary .....	918

<b>■ CHAPTER 25    Understanding XML Web Services .....</b>	<b>919</b>
The Role of XML Web Services .....	919
Benefits of XML Web Services .....	919
Defining an XML Web Service Client .....	920
The Building Blocks of an XML Web Service .....	921
Previewing XML Web Service Discovery .....	921
Previewing XML Web Service Description. ....	921
Previewing the Transport Protocol .....	922
The .NET XML Web Service Namespaces .....	922
Examining the System.Web.Services Namespace .....	922
Building an XML Web Service by Hand .....	923
Testing Your XML Web Service Using WebDev.WebServer.exe .....	924
Testing Your Web Service Using IIS .....	925
Viewing the WSDL Contract .....	925
The Autogenerated Test Page. ....	925
Providing a Custom Test Page .....	925
Building an XML Web Service Using Visual Studio 2005. ....	926
Implementing the TellFortune() Web Method .....	928
The Role of the WebService Base Class .....	929

Understanding the [WebService] Attribute . . . . .	929
The Effect of the Namespace and Description Properties . . . . .	930
The Name Property . . . . .	930
Understanding the [WebServiceBinding] Attribute . . . . .	931
Ignoring BP 1.1 Conformance Verification . . . . .	932
Disabling BP 1.1 Conformance Verification . . . . .	932
Understanding the [WebMethod] Attribute . . . . .	932
Documenting a Web Method via the Description Property . . . . .	932
Avoiding WSDL Name Clashes via the MessageName Property . . . . .	933
Building Stateful Web Services via the EnableSession Property . . . . .	933
Exploring the Web Service Description Language (WSDL) . . . . .	935
Defining a WSDL Document . . . . .	936
The <types> Element . . . . .	937
The <message> Element . . . . .	938
The <portType> Element . . . . .	938
The <binding> Element . . . . .	939
The <service> Element . . . . .	939
Revisiting the XML Web Service Wire Protocols . . . . .	940
HTTP GET and HTTP POST Bindings . . . . .	940
SOAP Bindings . . . . .	941
The wsdl.exe Command-Line Utility . . . . .	942
Transforming WSDL into a Server-Side XML Web Service Skeleton . . . . .	943
Transforming WSDL into a Client-Side Proxy . . . . .	944
Examining the Proxy Code . . . . .	944
The Default Constructor . . . . .	945
Synchronous Invocation Support . . . . .	946
Asynchronous Invocation Support . . . . .	946
Building the Client Application . . . . .	947
Generating Proxy Code Using Visual Studio 2005 . . . . .	947
Exposing Custom Types from Web Methods . . . . .	948
Exposing Arrays . . . . .	949
Exposing Structures . . . . .	949
Exposing ADO.NET DataSets . . . . .	950
A Windows Forms Client . . . . .	951
Client-Side Type Representation . . . . .	952
Understanding the Discovery Service Protocol (UDDI) . . . . .	953
Interacting with UDDI via Visual Studio 2005 . . . . .	954
Summary . . . . .	954



## PART 6 ■ ■ ■ Programming with .NET 3.0 Extensions

■ CHAPTER 26	<b>Establishing a .NET 3.0 Programming Environment</b> . . . . .	957
	Introducing the .NET 3.0 Technologies . . . . .	957
	Introducing C# 3.0 and LINQ Technologies . . . . .	959
	Welcome to Beta Land! . . . . .	959
	Installing the .NET Framework 3.0 Runtime Components . . . . .	960
	Installing the Windows Software Development Kit . . . . .	961
	Choosing an Installation Option . . . . .	962
	Investigating the SDK's Contents . . . . .	962
	Installing the Visual Studio 2005 "Orcas" Development Tools . . . . .	964
	Installing WPF and WCF Project Support . . . . .	964
	Installing the Visual Studio 2005 Extensions for Windows Workflow Foundation . . . . .	964
	Installing C# 3.0 and the LINQ Community Technology Preview . . . . .	966
	Repairing Visual Studio IntelliSense . . . . .	966
	Summary . . . . .	968
■ CHAPTER 27	<b>Introducing Windows Presentation Foundation</b> . . . . .	969
	The Motivation Behind WPF . . . . .	969
	Providing a Separation of Concerns via XAML . . . . .	970
	Providing an Optimized Rendering Model . . . . .	971
	Investigating the WPF Assemblies . . . . .	972
	The Role of the Application Class . . . . .	973
	The Role of the Window Class . . . . .	974
	Building a (XAML-Free) WPF Application . . . . .	976
	Extending the Window Class Type . . . . .	978
	Creating a Simple User Interface . . . . .	979
	Introducing XAML . . . . .	980
	Defining MainWindow in XAML . . . . .	981
	Defining the Application Object in XAML . . . . .	982
	Processing the XAML Files via msbuild.exe . . . . .	983
	Transforming Markup into a .NET Assembly . . . . .	984
	Mapping XAML to C# Code . . . . .	984
	The Role of BAML . . . . .	986
	XAML-to-Assembly Process Summary . . . . .	987
	Separation of Concerns Using Code-Behind Files . . . . .	988
	Experimenting with XAML Using XamlPad . . . . .	989

Building a WPF Application Using Visual Studio 2005 “Orcas” . . . . .	991
Generating XAML Using Microsoft Expression Interactive Designer . . . . .	992
Controlling Content Layout Using Panels . . . . .	993
Positioning Content Within Canvas Panels . . . . .	994
Positioning Content Within WrapPanel Panels . . . . .	996
Positioning Content Within StackPanel Panels . . . . .	998
Positioning Content Within Grid Panels . . . . .	999
Positioning Content Within DockPanel Panels . . . . .	1000
Building a Window’s Frame Using Nested Panels . . . . .	1001
Understanding WPF Controls . . . . .	1002
Configuring WPF Controls . . . . .	1003
Working with WPF Control Properties . . . . .	1004
Handling WPF Control Events . . . . .	1006
Applying Control Styles . . . . .	1008
Introducing WPF Graphical Rendering Services . . . . .	1010
Breaking Down the Graphical Services of WPF . . . . .	1011
Working with Basic Shapes . . . . .	1012
Introducing WPF’s Animation Services . . . . .	1014
Generating XAML Using Microsoft Expression Graphical Designer . . . . .	1016
A Brief Word Regarding XAML Browser Applications . . . . .	1017
Summary . . . . .	1019

## ■ CHAPTER 28    **Introducing Windows Communication Foundation . . . . .** 1021

The Motivation Behind WCF . . . . .	1021
Investigating the Core WCF Assemblies . . . . .	1023
The ABCs of WCF . . . . .	1023
Understanding WCF Contracts . . . . .	1024
Understanding WCF Bindings . . . . .	1024
Understanding WCF Addresses . . . . .	1026
Building a Complete WCF Application . . . . .	1026
The Interrelated Assemblies of a WCF Application . . . . .	1026
Defining and Implementing the Contract . . . . .	1027
Hosting the WCF Service . . . . .	1030
Specifying the ABCs . . . . .	1031
The Role of the ServiceHost Type . . . . .	1032
Details of the <system.ServiceModel> Element . . . . .	1034
Defining Service Behaviors . . . . .	1035
Communicating with the WCF Service . . . . .	1036
Generating Proxy Code Using svcutil.exe . . . . .	1036
Generating Proxy Code Using Visual Studio 2005 . . . . .	1038

WCF Data Type Representation . . . . .	1039
Updating the ICarOrder Service Contract . . . . .	1040
Recoding the CarOrderServiceClient Assembly . . . . .	1041
Data Formatting Using the XmlSerializer . . . . .	1042
Transporting Data in Binary Format . . . . .	1043
Building WCF Configuration Files Using the Service Configuration Editor . . . .	1043
Summary . . . . .	1045

## ■ CHAPTER 29    **Introducing Windows Workflow Foundation . . . . .** 1047

The Motivation Behind Windows Workflow Foundation . . . . .	1047
The Building Blocks of WF . . . . .	1048
The Integrated Services of WF . . . . .	1049
A First Look at WF Activities . . . . .	1050
The Role of Sequential Workflows and State Machine Workflows . . . . .	1051
Getting into the Flow of Workflow . . . . .	1053
The WF Assemblies and Core Namespaces . . . . .	1054
Building a Simple Workflow-Enabled Application . . . . .	1055
Examining the Initial Workflow Code . . . . .	1055
Adding a Code Activity . . . . .	1056
Adding a While Activity . . . . .	1057
Examining the WF Engine Hosting Code . . . . .	1060
Adding Custom Start-Up Parameters . . . . .	1061
Invoking Web Services Within Workflows . . . . .	1063
Working with the IfElse Activity . . . . .	1067
Building a Reusable WF Code Library . . . . .	1069
Authoring a Simple Workflow . . . . .	1070
Creating a Windows Forms Workflow-Enabled Application . . . . .	1071
A Brief Word Regarding Custom Activities . . . . .	1072
Summary . . . . .	1074

## ■ CHAPTER 30    **C# 3.0 Language Features . . . . .** 1075

Working with the C# 3.0 Command-Line Compiler . . . . .	1075
Understanding Implicitly Typed Local Variables . . . . .	1076
Restrictions on Implicitly Typed Variables . . . . .	1078
Implicitly Typed Local Arrays . . . . .	1079
Final Points Regarding Implicit Data Typing . . . . .	1079
Understanding Extension Methods . . . . .	1080
Defining Extension Methods . . . . .	1080
Invoking Extension Methods from an Instance Level . . . . .	1081
Invoking Extension Methods Statically . . . . .	1082
Importing Types That Define Extension Methods . . . . .	1083
Building and Using Extension Libraries . . . . .	1084

Understanding Object Initializers .....	1085
Calling Custom Constructors with Initialization Syntax .....	1087
Initializing Inner Types .....	1088
Understanding Collection Initialization .....	1089
Understanding Anonymous Types .....	1090
The Internal Representation of Anonymous Types .....	1091
The Implementation of ToString() and GetHashCode() .....	1092
The Semantics of Equality for Anonymous Types .....	1093
Anonymous Types Containing Anonymous Types .....	1095
Understanding the Role of Lambda Expressions .....	1096
Lambda Expressions As a Better Anonymous Method .....	1096
Dissecting a Lambda Expression .....	1099
The Two Flavors of Lambda Expressions .....	1100
Retrofitting the CarDelegate Example Using Lambda Expressions .....	1101
Lambda Expressions with Multiple (or Zero) Parameters .....	1104
Summary .....	1105

## ■ CHAPTER 31    **An Introduction to LINQ** ..... 1107

Defining the Role of LINQ .....	1107
The Core LINQ Assemblies .....	1109
A First Look at LINQ Query Expressions .....	1110
Revising Implicitly Typed Local Variables .....	1112
Revisiting Extension Methods .....	1113
Using LINQ to Query Generic Collections .....	1114
Defining LINQ Queries .....	1115
Revisiting Anonymous Types .....	1116
Using LINQ to Query Nongeneric Collections .....	1116
The Internal Representation of Query Operators .....	1118
Building Query Expressions with Query Operators (Revisited) .....	1118
Building Query Expressions Using the Sequence Type and Lambdas .....	1119
Building Query Expressions Using the Sequence Type and Anonymous Methods .....	1120
Building Query Expressions Using the Sequence Type and Raw Delegates .....	1120
Investigating the LINQ Query Operators .....	1122
Building LINQ Query Expressions .....	1122
Basic Selection Syntax .....	1123
Obtaining Subsets of Data .....	1125
Reversing Result Sets .....	1126
Sorting Expressions .....	1126
Transforming Query Results and the Role of Deferred Execution .....	1128

Querying Relational Databases Using LINQ to SQL .....	1129
The Role of Entity Classes .....	1130
The Role of the DataContext Type .....	1130
A Simple LINQ to SQL Example .....	1130
Building a Strongly Typed DataContext .....	1131
The [Table] and [Column] Attributes: Further Details .....	1133
Generating Entity Classes Using sqlmetal.exe .....	1134
Examining the Generated Entity Classes .....	1135
Defining Relationships Using Entity Classes .....	1136
The Strongly Typed DataContext .....	1137
Programming Against the Generated Types .....	1138
Building Entity Classes Using Visual Studio 2005 .....	1140
Inserting New Items .....	1141
Updating Existing Items .....	1142
Deleting Existing Items .....	1142
Manipulating XML Documents Using LINQ to XML .....	1142
The System.Xml.XLinq Namespace .....	1143
Programmatically Creating XML Documents .....	1144
Loading and Parsing XML Content .....	1146
Navigating an In-Memory Document .....	1146
Selecting Elements Using LINQ to XML .....	1146
Modifying Data in an XML Document .....	1148
Summary .....	1149
<b>INDEX .....</b>	<b>1151</b>

# 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

**W**hile 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 multithreaded 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 `mscorlib.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](mailto: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

