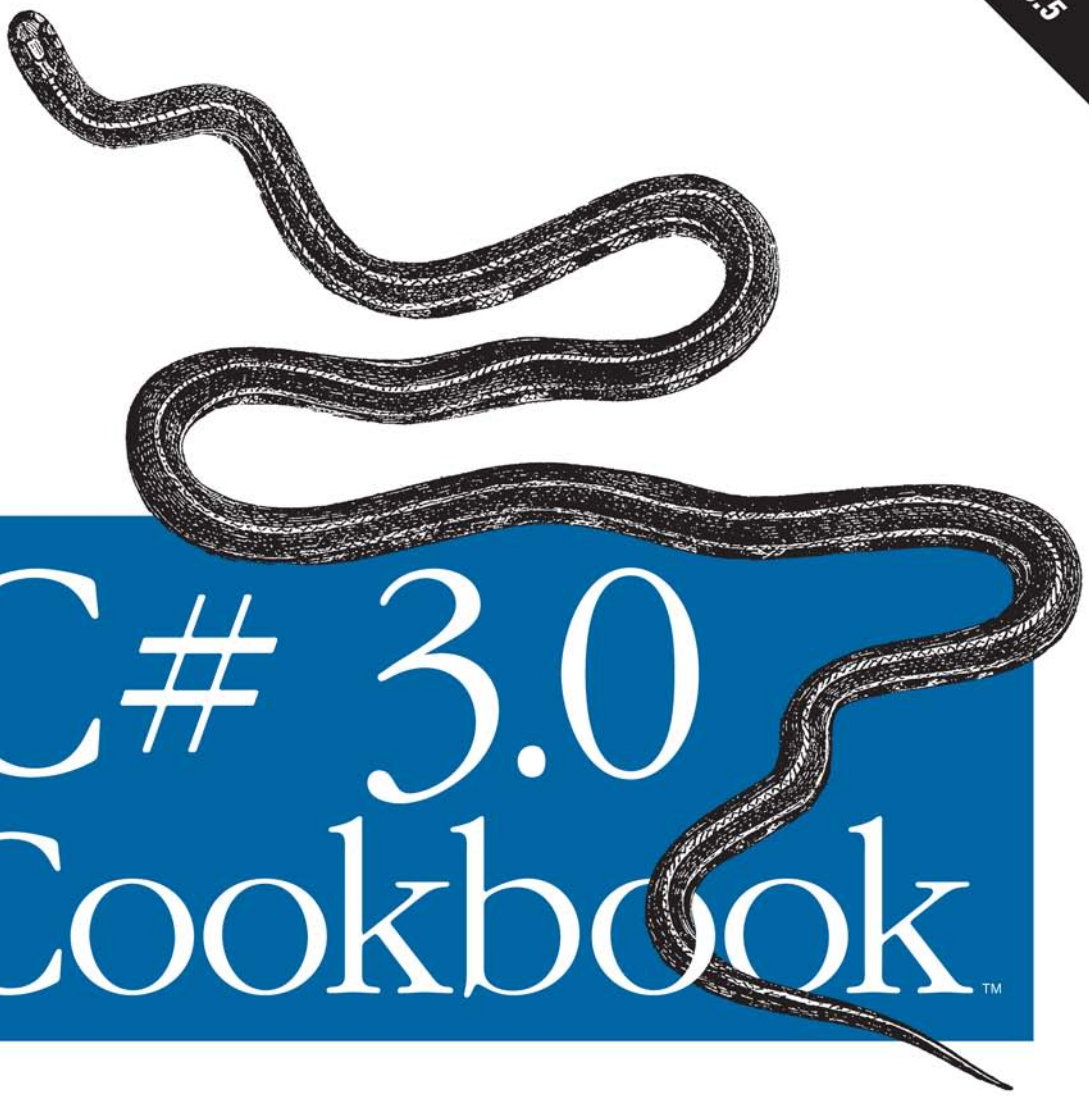
More Than 250 Solutions for C# 3.0 Programmers

3rd Edition
Thoroughly revised for C# 3.0 and .NET 3.5



C# 3.0 Cookbook™

O'REILLY®

Jay Hilyard & Stephen Teilhet

C# 3.0 Cookbook™

Other Microsoft .NET resources from O'Reilly

Related titles

Building a Web 2.0 Portal with ASP.NET 3.5
C# 3.0 Design Patterns
Learning C#
Programming ASP.NET
Programming C#
Visual C# 2005: A Developer's Notebook™

.NET Books Resource Center

dotnet.oreilly.com is a complete catalog of O'Reilly's books on .NET and related technologies, including sample chapters and code examples.

ONDotnet.com provides independent coverage of fundamental, interoperable, and emerging Microsoft .NET programming and web services technologies.

Conferences

O'Reilly brings diverse innovators together to nurture the ideas that spark revolutionary industries. We specialize in documenting the latest tools and systems, translating the innovator's knowledge into useful skills for those in the trenches. Visit *conferences.oreilly.com* for our upcoming events.

Safari Bookshelf (*safari.oreilly.com*) is the premier online reference library for programmers and IT professionals. Conduct searches across more than 1,000 books. Subscribers can zero in on answers to time-critical questions in a matter of seconds. Read the books on your Bookshelf from cover to cover or simply flip to the page you need. Try it today for free.

THIRD EDITION

C# 3.0 Cookbook™

Jay Hilyard and Stephen Teilhet

O'REILLY®
Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo

C# 3.0 Cookbook™, Third Edition
by Jay Hilyard and Stephen Teilhet

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*safari.oreilly.com*). For more information, contact our corporate/institutional sales department: (800) 998-9938 or *corporate@oreilly.com*.

Editor: John Osborn
Production Editor: Adam Witwer
Production Services: nSight, Inc.
Cover Designer: Karen Montgomery
Interior Designer: David Futato
Illustrators: Robert Romano and Jessamyn Read

Printing History:

January 2004: First Edition.
January 2006: Second Edition.
December 2007: Third Edition.

ISBN-10: 0-596-51610-X
ISBN-13: 978-0-596-51610-9
[M]

To Brooke
My wife, my best friend, and the most supportive
person I know. This one was for you;
you earned it.

—Jay Hilyard

To my loving wife Kandis and my two wonderful
sons, Patrick and Nicholas.

—Stephen Teilhet

Table of Contents

Preface

C# is a language targeted at developers for the Microsoft .NET platform who have already worked with a C-like language such as C, C++, or Java. Unlike previous versions of C or C++ for the Microsoft Windows platform, C# code runs under a *managed execution environment*. Microsoft portrays C# as a modern and innovative language for .NET development and continues to deliver on that with new features such as Language Integrated Query (LINQ). The new features in C# 3.0 allow for more of a declarative and functional style of programming, when that is appropriate, while it still has great object-oriented features as well. The main idea is to use the style of programming that fits your problem, and C# will support your endeavor.

C# allows you to perform many C/C++-like functions, such as direct memory access via pointers and operator overloading, that are not supported in Visual Basic .NET. C# is the system-level programming language for .NET. You can still do great application-level work in C#, but it really shines when you need to build code a little closer to the Framework.

If you have seen C#, you may have noticed that it looks a lot like Java; Java programmers will feel very much at home in C# once they learn the Framework SDK. C# can also be a great language for Visual Basic .NET programmers when they need a little more control over what the code is doing and don't want to have to write C++ to gain an advantage. On the Web, you'll find a large community of people doing really neat things with C# and tons of sample code on sites such as *http://www.codeplex.com* and *http://www.codeproject.com*.

We started writing this book together based on programming problems we ran into when we were first learning C# and have continued to expand it based on new challenges and capabilities in the language. In this edition, we have reworked the approach of many solutions to take advantage of LINQ and have also created entirely new solutions based on LINQ and the other new features in C# 3.0. We hope that it will help you get past some of the common (and not-so-common) pitfalls and initial questions everyone has when learning a new language as well as the slightly off-the-beaten-path items that come up during a development cycle. There

are recipes addressing things we found missing from the .NET Framework Class Library (FCL), even though Microsoft has provided tons of functionality to keep folks from reinventing the wheel. Some of these solutions you might immediately use, and some may never darken your door, but we hope this book helps you get the most out of C# and the .NET Framework.

The book is laid out with respect to the types of problems you will solve as you progress through your life as a C# programmer. These solutions are called *recipes*; each recipe consists of a single problem, its solution, a discussion of the solution and other relevant related information, and finally, where you can look for more information about the classes used from the FCL, other books addressing this topic, related articles, and other recipes. The question-answer format provides complete solutions to problems, making the book easy to read and use. Nearly every recipe contains a complete, documented code sample, showing you how to solve the specific problem, as well as a discussion of how the underlying technology works and a list of alternatives, limitations, and other considerations when appropriate.

Who This Book Is For

You don't have to be an experienced C# or .NET developer to use this book—it is designed for users of all levels. This book provides solutions to problems that developers face every day as well as some that may come along less frequently. The recipes are targeted at the real-world developer who needs to solve problems now, not learn lots of theory before being able to solve the problem. While reference or tutorial books can teach general concepts, they do not generally provide the help you need in solving real-world problems. We choose to teach by example, the natural way for most people to learn.

The majority of the problems addressed in this book are frequently faced by C# developers, but some of the more advanced problems call for more intricate solutions that combine many techniques. Each recipe is designed to help you quickly understand the problem, learn how to solve it, and find out any potential trade-offs or ramifications to help you solve your problems quickly, efficiently, and with minimal effort.

To save you even the effort of typing in the solution, we provide the sample code for the book on the O'Reilly web site to facilitate the "editor inheritance" mode of development (copy and paste) as well as to help less-experienced developers see good programming practice in action. The sample code provides a running test harness that exercises each of the solutions, but enough of the code is provided in each solution in the book to allow you to implement the solution without the sample code. The sample code is available from the book's catalog page: *http://www.oreilly.com/catalog/9780596516109*.

What You Need to Use This Book

To run the samples in this book, you need a computer running Windows XP or later. A few of the networking and XML solutions require Microsoft Internet Information Server (IIS) Version 5.1 or later, and the FTP recipes in the Networking chapter require a locally configured FTP server.

To open and compile the samples in this book, you need Visual Studio .NET 2008. If you are proficient with the downloadable Framework SDK and its command-line compilers, you should not have any trouble following the text of this book and the code samples.

Platform Notes

The solutions in this book were developed using Visual Studio .NET 2008. The differences between C# 3.0 and C# 2.0 are significant, and the sample code has changed from the second edition to reflect that.

It is worth mentioning that although C# is now at version 3.0, the .NET Framework has progressed to version 3.5. .NET 3.0 introduced Windows Communication Foundation, Windows Presentation Foundation, and Windows Workflow Foundation as additional functionality to the 2.0 framework base, but C# was not changed. Now in C# 3.0, there is a bunch of new functionality, mostly due to LINQ and the ability to do more functional programming.

How This Book Is Organized

This book is organized into 20 chapters, each of which focuses on a particular topic in creating C# solutions. The following paragraphs summarize each chapter to give you an overview of this book's contents:

Chapter 1, *Language Integrated Query (LINQ)*
: This chapter covers Language Integrated Query (LINQ) and its usage with objects, ADO.NET, and XML. There are recipes using many of the Standard Query Operators and showing how to use some of the query operators that are not keywords in the language, but are still quite powerful.

Chapter 2, *Strings and Characters*
: This chapter covers both the `String` and `Char` data types. Recipes show such things as how to compare strings in various ways, encode/decode strings, break strings apart, and put them back together again.

Chapter 3, *Classes and Structures*
: This large chapter contains recipes dealing with both class and structure data types. This chapter covers a wide range of recipes, from design patterns to converting a class to a full-blown command-line argument-processing system.

Chapter 4, *Generics*

This chapter focuses on the generics capacity in C#, which allows you to have code operate uniformly on values of different types. There are recipes to help your general understanding of generics as well as when they are appropriate to use, what support is provided in the Framework for them, and how to create custom implementations of collections using generics.

Chapter 5, *Collections*

This chapter examines recipes that make use of collections. The collection recipes make use of—as well as extend the functionality of—the array (single, multi, and jagged), the `List<T>`, and the `Hashtable`. The generic-based collections are explored, and the various ways to create your own strongly typed collection are also discussed.

Chapter 6, *Iterators, Partial Types, and Partial Methods*

In this chapter, two of the features of C# are used to solve very different programming problems. We show how you can implement iterators for generic and nongeneric types and implement `foreach` functionality using iterators, as well as custom iterator implementations. The other feature of C# in this chapter is partial types and methods. We show how you can use partial types and methods to do such things as better segmenting your code and how to generate code that is more easily extensible.

Chapter 7, *Exception Handling*

The recipes in this chapter focus on the best ways to implement exception handling in your application. Preventing unhandled exceptions, reading and displaying stack traces, and throwing/rethrowing exceptions are included recipes. In addition, specific recipes show how to overcome some tricky situations, such as exceptions from late-bound called methods.

Chapter 8, *Diagnostics*

This chapter presents recipes that use data types that fall under the `System.Diagnostics` namespace. Recipes deal with the `Trace/Debug` classes, event logs, processes, performance counters, and custom debugger displays for your types.

Chapter 9, *Delegates, Events, and Lambda Expressions*

This chapter's recipes show how delegates, events, and lambda expressions can be used in your applications. Recipes allow manipulation of delegates that call more than one method, synchronous delegates, and asynchronous delegates. Lambda expressions are explored, and recipes show their usage in place of old-style delegates as well as their use in implementing closures and functors.

Chapter 10, *Regular Expressions*

This chapter covers a useful set of classes that are employed to run regular expressions against strings. Recipes enumerate regular expression matches, break up strings into tokens, find/replace characters, and verify the syntax of a regular expression. We also include a recipe that contains many common regular expression patterns.

Chapter 11, *Data Structures and Algorithms*

This chapter ventures a bit outside of what is provided for you in the .NET Framework Class Library and implements certain data structures and algorithms that are not in the FCL, or possibly are not in existence exactly the way you would like to use them, but are ones that you have used to solve problems before. Items such as queues, maps, trees, and hashes are examined.

Chapter 12, *Filesystem I/O*

This chapter deals with file system interactions in four distinct ways. The first way is to look at typical file interactions; the second way looks at directory- or folder-based interactions; the third way deals with paths and temporary files; and the fourth way deals with advanced file system I/O topics.

Chapter 13, *Reflection*

This chapter shows ways to use the built-in assembly inspection system provided by the .NET Framework to determine what types, interfaces, and methods are implemented within an assembly and how to access them in a late-bound fashion.

Chapter 14, *Web*

This chapter covers accessing a web site and its content as well as programmatically determining web site configuration. Among the recipes in this chapter are using the web browser control and setting up caching triggers to refresh cached data when a database table changes.

Chapter 15, *XML*

If you use .NET, it is likely that you will be dealing with XML to one degree or another; in this chapter, we explore some of the uses for XML and how to program against it using LINQ to XML, the XmlReader/XmlWriter, and XmlDocument. There are examples using both XPath and XSLT, and topics such as the validation of XML and transformation of XML to HTML are shown.

Chapter 16, *Networking*

This chapter explores the connectivity options provided by the .NET Framework and how to programmatically access network resources. Recipes for using TCP/IP directly, named pipes for communication, building your own port scanner, and more are covered here.

Chapter 17, *Security*

There are many ways to write secure code and protect data using the .NET Framework, and in this chapter, we explore areas such as controlling access to types, encryption and decryption, securely storing data, and using programmatic and declarative security.

Chapter 18, *Threading and Synchronization*

This chapter addresses the subject of using multiple threads of execution in a .NET program and issues such as how to implement threading in your application, protecting resources from and allowing safe concurrent access, storing per-thread data, and how to use the synchronization primitives in .NET to write thread-safe code.

Chapter 19, *Toolbox*
: This chapter has recipes for those random sorts of operations that developers run into over and over again, such as determining locations of system resources, sending email, and working with services. It also covers some less frequently accessed but helpful application pieces such as message queuing, running code in a separate AppDomain, and finding the versions of assemblies in the GAC.

Chapter 20, *Numbers and Enumerations*
: This chapter focuses on the numeric and enumeration data types used in C# code. Recipes cover such things as numeric conversions, using bitwise operators on numbers, and testing strings to determine whether they contain a numeric value. The display, conversion, and testing of enumeration types and recipes on using enumerations that consist of bit flags are also shown.

In some cases, certain recipes are related. In these cases, the See Also section of the recipe as well as some text in the Discussion will note the relationships.

What Was Left Out

This book is not a reference or a primer about C#. Some good primers and reference books are *C# in a Nutshell*, *C# Language Pocket Reference*, and *Learning C#*, all titles available from O'Reilly. The MSDN Library is also invaluable. It is included with Visual Studio .NET 2008 and available online at *http://msdn.microsoft.com/library/default.asp*.

This book is not about how to use Visual Studio .NET 2008 to build, compile, and deploy applications. See *Mastering Visual Studio .NET* (O'Reilly) for excellent coverage of these topics.

Conventions Used in This Book

This book uses the following typographic conventions:

Italic
: Used for URLs, names of directories and files, options, and occasionally for emphasis.

`Constant width`
: Used for program listings and for code items such as commands, options, switches, variables, attributes, keys, functions, types, classes, namespaces, methods, modules, properties, parameters, values, objects, events, event handlers, XML tags, HTML tags, macros, the contents of files, and the output from commands.

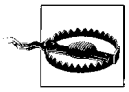
`Constant width bold`
: Used in program listings to highlight an important part of the code.

`Constant width italic`
: Used to indicate replaceable parts of code.

//…
: Ellipses in C# code indicate text that has been omitted for clarity.

<!--…-->
: Ellipses in XML Schemas and documents' code indicate text that has been omitted for clarity.

This icon indicates a tip, suggestion, or general note.



This icon indicates a warning or caution.

About the Code

Nearly every recipe in this book contains one or more code samples. These samples are included in a single solution and are pieces of code and whole projects that are immediately usable in your application. Most of the code samples are written within a class or structure, making it easier to use within your applications. In addition to this, any `using` directives are included for each recipe so that you will not have to search for which ones to include in your code.

Complete error handling is included only in critical areas, such as input parameters. This allows you to easily see what is correct input and what is not. Many recipes omit error handling. This makes the solution easier to understand by focusing on the key concepts.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books *does* require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation *does* require permission.