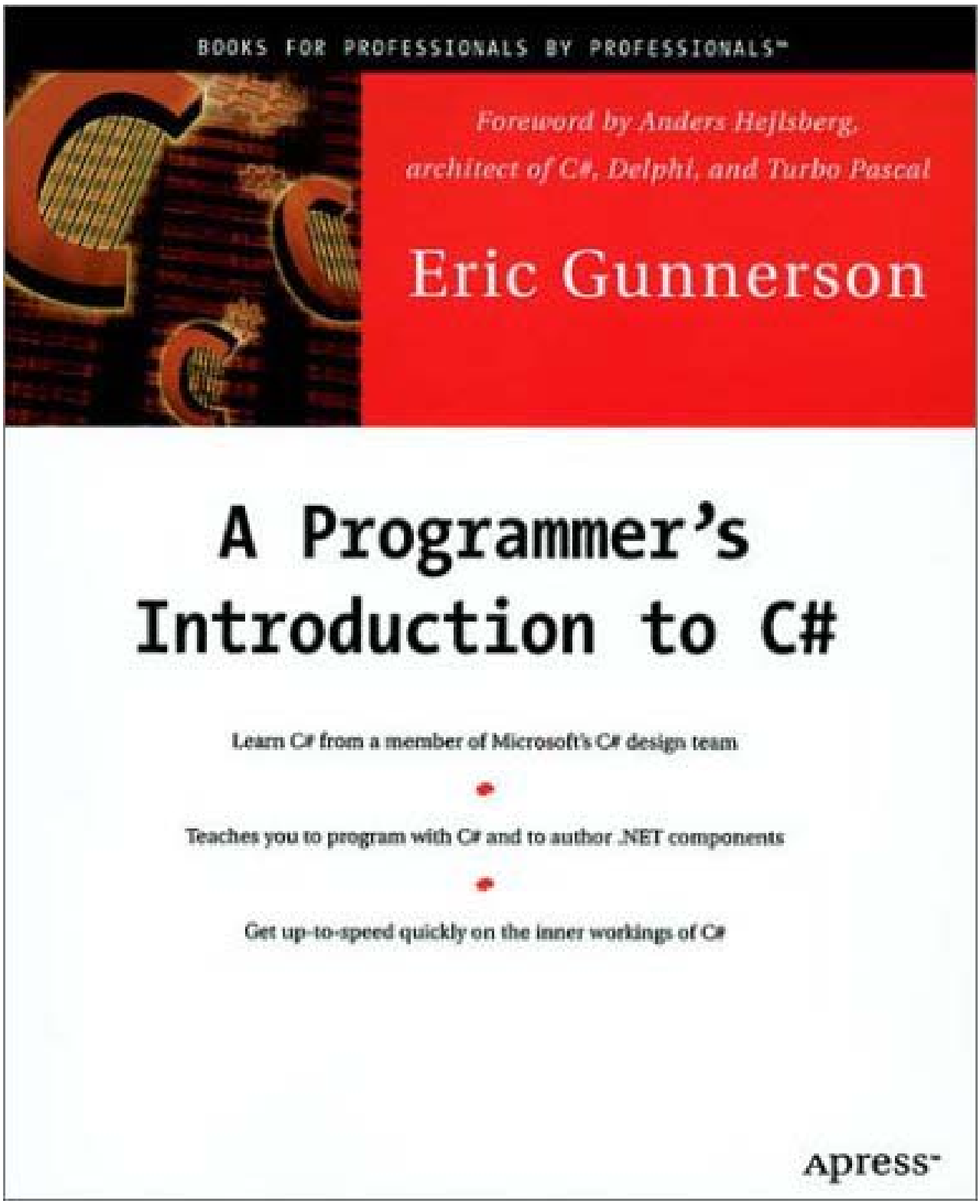


**Release Team[oR] 2001**
**[x] Programming**

**A Programmer's Introduction to C#**

by Eric Gunnerson

ISBN: 1893115860

Apress © 2000, 358 pages

This book takes the C programmer through the all the details—from basic to advanced-- of the new Microsoft C# language.

Companion Web Site

Table of Contents

Colleague Comments

Back Cover

**Synopsis**

Written as an introduction to the new C#, this guide takes the experienced C programmer a few steps beyond the basics. It covers objects, data types, and flow control, and even delves into some background on the new Microsoft NET Frameworks environment. Keeping in mind that this is for those familiar with C (and even Java), the book goes into some of the advanced features and improvements found in this new language. It also offers a comparison between C#, C++, Visual Basic, and Java.

**Table of Contents**

**Back Cover**

- Provides in-depth information about the functionality of the language and C# "Quick Start"
- Shows you how to write components that fit seamlessly into the .NET Frameworks
- Includes C# reference information tailored for C++, Java and Visual Basic Programmers
- Suitable for intermediate to advanced developers and includes coverage of advanced topics in C#

Eric Gunnerson, A member of the Microsoft C# design team, has written a comprehensive C# tutorial addressed to the experienced programmer. *A Programmer's Introduction to C#* explains how C# works, why it was designed the way it was, and how C# fits into Microsoft's new .NET Frameworks. This book teaches programmers how to write C# components and how to truly leverage the power of the new .NET Runtime.

Gunnerson's first chapters are for the impatient programmer. In them, he provides an introduction to object-oriented programming with C# along with a C# "Quick Start" for those who want a fast track to programming in C#. This is followed by a more comprehensive section in which he uses his unique insider's view to explain each of the new C# language features in detail. He covers fundamentals such as classes, structs, attributes, statements and flow of execution, arrays, delegates and events, exception handling, and the unique interoperability provided by the .NET Frameworks.

In the final portion of the book, Gunnerson provides a useful overview of the .NET Frameworks. A section on the .NET Common Language Runtime and Framworks shows how to write components that function well in the runtime and how to use the basic runtime features (such as I/O). Gunnerson also devoted time to more advanced topics such as regular expressions and collections. Final chapters include Making Friends with the .NET Frameworks, System.Array and the Collection Classes, .NET Fraeworks Overview, Deeper into C# and Defensive Programming. Also included is a detailed C# language comparison that will be indispensable for programmers currently working in C++, Java, or Visual Basic.

All of the source code for this book in online at http://www.apress.com.

## About the Author

Eric Gunnerson is a software design engineer in Microsoft's Visual C++ QA group and a member of the C# design team. In the course of his professional career, he has worked primarily on database products and tools – and is proud of the fact that nearly half of the companies he has worked for remain in business.

# A Programmer's Introduction to C#

**ERIC GUNNERSON**

 ISBN (pbk): 1-893115-86-0

Printed and bound in the United States of America 2345678910


**Editorial Directors:** Dan Appleman, Gary Cornell, Karen Watterson
**Technical Reviewers:** David Staheli, Shawn Vita, Gus Perez, Jerry Higgins, Brenton Webster
**Editor:** Andy Carroll
**Projects Manager:** Grace Wong
**Production Editor:** Janet Vail
**Page Compositor and Soap Bubble Artist:** Susan Glinert
**Artist:** Karl Miyajima
**Indexer:** Nancy Guenther
**Cover and Interior Design:** Derek Yee Design

Distributed to the book trade in the United States by Springer-Verlag New York, Inc., 175 Fifth Avenue, New York, NY, 10010 and outside the United States by Springer-Verlag GmbH & Co. KG, Tiergartenstr. 17, 69112 Heidelberg, Germany
In the United States, phone 1-800-SPRINGER orders@springer-ny.com ; http://www.springer-ny.com Outside the United States, contact orders@springer.de; http://www.springer.de ; fax +49 6221 345229
For information on translations, please contact Apress directly at 901 Grayson Street, Suite 204, Berkeley, CA, 94710 Phone: 510-549-5931; Fax: 510-549-5939; info@apress.com ; http://www.apress.com



**Dedication**

*To Tony Jongejan, for introducing me to programming and being ahead of his time.*

**Acknowledgments**

**THOUGH WRITING A BOOK** is often a lonely undertaking, no author can do it without help.

I'd like to thank all those who helped me with the book, including all those team members who answered my incessant questions and read my unfinished drafts. I would also like to thank my managers and Microsoft, both for allowing me to work on such a unique project and for allowing me to write a book about it.

Thanks to the Apress team for making a bet on an unproven author and for not pestering me when I waited to turn in content.

Thanks to all the artists who provided music to write to—all of which was commercially purchased—with special thanks to Rush for all their work.

Finally, I'd like to thank all those who supported me at home; my wife Kim and daughter Samantha who didn't complain when I was working, even when it was during our vacation, and for my cat for holding my arms down while I was writing.

# Foreword

WHEN YOU CREATE a new programming language, the first question you're asked invariably is, why? In creating C# we had several goals in mind:

- *To produce the first component-oriented language in the C/C++ family.* Software engineering is less and less about building monolithic applications and more and more about building components that slot into various execution environments; for example, a control in a browser or a business object that executes in ASP+. Key to such components is that they have properties, methods, and events, and that they have attributes that provide declarative information about the component. All of these concepts are first-class language constructs in C#, making it a very natural language in which to construct and use components.
- *To create a language in which everything really is an object.* Through innovative use of concepts such as boxing and unboxing, C# bridges the gap between primitive types and classes, allowing any piece of data to be treated as an object. Furthermore, C# introduces the concept of value types, which allows users to implement lightweight objects that do not require heap allocation.
- *To enable construction of robust and durable software.* C# was built from the ground up to include garbage collection, structured exception handling, and type safety. These concepts completely eliminate entire categories of bugs that often plague C++ programs.
- *To simplify C++, yet preserve the skills and investment programmers already have.* C# maintains a high degree of similarity with C++, and programmers will immediately feel comfortable with the language. And C# provides great interoperability with COM and DLLs, allowing existing code to be fully leveraged.

We have worked very hard to attain these goals. A lot of the hard work took place in the C# design group, which met regularly over a period of two years. As head of the C# Quality Assurance team, Eric was a key member of the group, and through his participation he is eminently qualified to explain not only how C# works, but also why it works that way. That will become evident as you read this book.

I hope you have as much fun using C# as those of us on the C# design team had creating it.
*Anders Hejlsberg*
*Distinguished Engineer*
*Microsoft Corporation*

# About This Book

**C# IS ONE OF THE MOST EXCITING** projects I've ever had the privilege to work on. There are many languages with different strengths and weaknesses, but once in a while a new language comes along that that meshes well with the hardware, software, and programming approaches of a specific time. I believe C# is such a language. Of course, language choice is often a "religious issue." [1]

I've structured this book as a tour through the language, since I think that's the best and most interesting way to learn a language. Unfortunately, tours can often be long and boring, especially if the material is familiar, and they sometimes concentrate on things you don't care about, while overlooking things you're interested in. It's nice to be able to short-circuit the boring stuff and get into the interesting stuff. To do that, there are two approaches you might consider:
To start things off quickly, there's Chapter 3, "C# QuickStart," which is a quick overview of the language, and gives enough information to start coding.
Chapter 34, "C# Compared to Other Languages," offers language-specific comparisons for C++, VB, and Java for programmers attuned to a specific language, or for those who like to read comparisons.