Covers C# 2 and C# 3

C# IN DEPTH

Jon Skeet

Foreword by Eric Lippert

MANNING

C# in Depth

C# in Depth

JON SKEET

MANNING
Greenwich
(74° w. long.)

For online information and ordering of this and other Manning books, please visit www.manning.com. The publisher offers discounts on this book when ordered in quantity. For more information, please contact:

Special Sales Department
Manning Publications Co.
Sound View Court 3B fax: (609) 877-8256
Greenwich, CT 06830 email: orders@manning.com

Manning Publications Co.
Sound View Court 3B
Greenwich, CT 06830

Copyeditor: Liz Welch
Typesetter: Gordan Salinovic
Cover designer: Leslie Haimes

ISBN 1933988363
Printed in the United States of America
1 2 3 4 5 6 7 8 9 10 – MAL – 13 12 11 10 09 08

For family, friends, colleagues,
and all those who love C#

brief contents

contents

foreword

There are two kinds of pianists.

There are some pianists who play not because they enjoy it, but because their parents force them to take lessons. Then there are those who play the piano because it pleases them to create music. They don't need to be forced; on the contrary, they sometimes don't know when to stop.

Of the latter kind, there are some who play the piano as a hobby. Then there are those who play for a living. That requires a whole new level of dedication, skill, and talent. They may have some degree of freedom about what genre of music they play and the stylistic choices they make in playing it, but fundamentally those choices are driven by the needs of the employer or the tastes of the audience.

Of the latter kind, there are some who do it primarily for the money. Then there are those professionals who would want to play the piano in public even if they weren't being paid. They enjoy using their skills and talents to make music for others. That they can have fun and get paid for it is so much the better.

Of the latter kind, there are some who are self-taught, who "play by ear," who might have great talent and ability but cannot communicate that intuitive understanding to others except through the music itself. Then there are those who have formal training in both theory and practice. They can explain what techniques the composer used to achieve the intended emotional effect, and use that knowledge to shape their interpretation of the piece.

Of the latter kind, there are some who have never looked inside their pianos. Then there are those who are fascinated by the clever escapements that lift the damper felts a fraction of a second before the hammers strike the strings. They own key levelers and capstan wrenches. They take delight and pride in being able to understand the mechanisms of an instrument that has five to ten thousand moving parts.

Of the latter kind, there are some who are content to master their craft and exercise their talents for the pleasure and profit it brings. Then there are those who are not just artists, theorists, and technicians; somehow they find the time to pass that knowledge on to others as mentors.

I have no idea if Jon Skeet is any kind of pianist. But from my email conversations with him as one of the C# team's Most Valuable Professionals over the years, from reading his blog and from reading every word of this book at least three times, it has become clear to me that Jon is that latter kind of software developer: enthusiastic, knowledgeable, talented, curious and analytical—a teacher of others.

C# is a highly pragmatic and rapidly evolving language. Through the addition of query comprehensions, richer type inference, a compact syntax for anonymous functions, and so on, I hope that we have enabled a whole new style of programming while still staying true to the statically typed, component-oriented approach that has made C# a success.

Many of these new stylistic elements have the paradoxical quality of feeling very old (lambda expressions go back to the foundations of computer science in the first half of the twentieth century) and yet at the same time feeling new and unfamiliar to developers used to a more modern object-oriented approach.

Jon gets all that. This book is ideal for professional developers who have a need to understand the "what" and "how" of the latest revision to C#. But it is also for those developers whose understanding is enriched by exploring the "why" of the language's design principles.

Being able to take advantage of all that new power will require new ways of thinking about data, functions, and the relationship between them. It's not unlike trying to play jazz after years of classical training—or vice versa. Either way, I am looking forward to finding out what sorts of functional compositions the next generation of C# programmers come up with. Happy composing, and thanks for choosing the key of C# to do it in.

ERIC LIPPERT
Senior Software Engineer, Microsoft

preface

I have a sneaking suspicion that many authors have pretty much stumbled into writing books. That's certainly true in my case. I've been writing about Java and C# on the Web and in newsgroups for a long time, but the leap from that to the printed page is quite a large one. From my perspective, it's been an "anti-Lemony Snicket"—a series of *fortunate* events.

I've been reviewing books for various publishers, including Manning, for a while. In April 2006 I asked whether it would be OK to write a blog entry on a book that looked particularly promising: *PowerShell in Action*. In the course of the ensuing conversation, I somehow managed to end up on the author team for *Groovy in Action*. I owe a huge debt of thanks to my wife for even allowing me to agree to this—which makes her sound like a control freak until you understand we were expecting twins at the time, and she had just gone into the hospital. It wasn't an ideal time to take on extra work, but Holly was as supportive as she's always been.

Contributing to the Groovy book took a lot of hard work, but the writing bug firmly hit me during the process. When talking with the principal author, Dierk König, I realized that I wanted to take on that role myself one day. So, when I heard later that Manning was interested in publishing a book about C#3, I started writing a proposal right away.

My relationship with C# itself goes further back. I started using it in 2002, and have kept up with it ever since. I haven't been using it professionally for all that time—I've been flitting back and forth between C# and Java, depending on what my employers wanted for the projects I was working on. However, I've never let my interest in it drop, posting on the newsgroups and developing code at home. Although I