.NET 2.0 for Delphi Programmers

Jon Shemitz

Apress®

.NET 2.0 for Delphi Programmers

ISBN-13: 978-1-59059-386-8

ISBN-10: 1-59059-386-3

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

Lead Editor: Jim Sumser
Technical Reviewer: Hallvard Vassbotn
Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Gennick, Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Dominic Shakeshaft, Jim Sumser, Keir Thomas, Matt Wade
Project Manager: Sofia Marchant
Copy Edit Manager: Nicole LeClerc
Copy Editor: Ami Knox
Assistant Production Director: Kari Brooks-Copony
Production Editor: Lori Bring
Compositor: Susan Glinert
Proofreader: Liz Welch
Indexer: Rebecca Plunkett
Artist: April Milne
Cover Designer: Kurt Krames
Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail `orders-ny@springer-sbm.com`, or visit `http://www.springeronline.com`.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail `info@apress.com`, or visit `http://www.apress.com`.

The source code for this book is available to readers at `www.apress.com` in the Source Code section.

To Anders Hejlsberg, for Turbo Pascal, Delphi, and now C#;

And to the vegetable garden that I didn't grow in 2005
so that I'd have time to finish this book;

And, most of all, to Tané, Sam, and Arthur, with thanks
for all your patience and encouragement.

Contents at a Glance

PART 4 Appendixes

Contents

Table Cross-Reference

About the Author

JON SHEMITZ has been programming since he was 12, when he learned Focal on a PDP-8. He's been programming professionally since he graduated from Yale in 1981, and has done everything from shrink-wrap programming to consulting. Jon has used Borland Pascals since Turbo Pascal 1, and has been doing .NET programming in C# since 2002. This is Jon's second book: he's written dozens of programming articles; contributed to four other books; and has given programming talks on two continents.

Jon does contract programming, consulting, and training—you can contact him at `www.midnightbeach.com`.

About the Technical Reviewer

HALLVARD VASSBOTN is a senior systems developer at and partial owner of Infront AS (`www.infront.no`), developing state-of-the-art real-time financial information and trading systems (`www.theonlinetrader.com`). Hallvard has been a professional programmer since 1985, and has written numerous articles for *The Delphi Magazine* and tech edited several popular Delphi books.

You can read his technical blog at `hallvards.blogspot.com/`.

Hallvard lives in Oslo, Norway, with the three diamonds of his heart, Nina, Ida, and Thea. You can reach him at `hallvardvassbotn@c2i.net`.

Acknowledgments

This book represents a lot of effort over several years. It wouldn't have been possible without the help of many talented people—most of whom I've never met.

Dan Appleman and the editorial board at Apress had the good taste to agree that a book about .NET for Delphi programmers would sell better than the Delphi for .NET reference that they originally agreed to publish. More importantly, they've been willing to wait for a book that's longer and later than they originally expected. Sofia Marchant, my project manager, has been answering my questions for nearly three-and-a-half years, and she put together a production team that smoothly and painlessly turned my 4 meg of Word and TIF files into a printed book. Ami Knox, the copy editor, made my punctuation and capitalization consistent, and caught awkward phrases that made it past all my rewrites; I'd especially like to thank Ami for the extra effort involved in dealing with my 'scare quotes.' Liz Welch, the proofreader, did a great job transferring the syntax highlighting from my manuscript to the printed page, and she also caught several mistakes that made it past both Ami and me.

Google made it much easier for me to answer various questions. Without Google, the research would have taken much longer, and I might never even have found some of the more interesting details. VMWare generously supplied me with a free "authors copy" of *VMware Workstation*, which made it much easier (and safer) to install and uninstall beta software. My orthopedist, Dr. Howard Schwartz, was very patient with my impatience when a bike accident tore shoulder ligaments and disabled me for three months near the end of the first draft.

I have the good fortune to live with three fine writers—my partner, Tané Tachyon, and our sons, Sam and Arthur Shemitz. All three of them have had to put up with innumerable problem paragraphs, and inevitably made good suggestions that helped move things along.

Weyert de Boer, Mark Boler, Marcel van Brakel, Alessandro Federici, Marc Hoffman, Chuck Jazdzewski, Ian Marteens, Jon Skeet, and Danny Thorpe all read drafts of various chapters. Their comments helped improve the book, and their positive feedback helped keep me working.

I'd particularly like to thank Jon Skeet and Marcel van Brakel. Jon helped me understand the .NET *memory model* (Chapter 17) and how it affects interprocessor synchronization. Marcel read every chapter at least once, and made detailed comments on most of them. I've benefited greatly from his deep knowledge, his helpful suggestions, and his uncanny ability to find every point where I waved my hands vaguely and hoped no one would notice.

Finally, I can't say enough about Hallvard Vassbotn, my technical reviewer. This project was much more work (and took much longer) than he could possibly have anticipated when he signed on, yet he read every chapter two or three times—and caught errors and made suggestions, each time. Hallvard also wrote the Delphi syntax chapter when I was considering dropping it after my bike accident. I've enjoyed working with him, and have been thoroughly impressed by his intelligence, energy, and diligence. Naturally, any mistakes that remain are entirely my fault.

April 2006
Santa Cruz, California

Preface

It's rough being a Delphi programmer. We know we have a wonderful, productive environment—but jobs are few and far between. We know that we can write any sort of application with Delphi—yet Delphi is seen as a GUI builder and a database front-end. We've all seen (or at least heard of) systems where the 'interesting parts' are written in C or C++, in DLLs, and Delphi is just used for the GUI interface. We may know C++ and have significant Win32 experience—and yet not been considered for C++ jobs because we didn't know MFC or ATL.

.NET changes that. All .NET languages use the same Framework Class Library (FCL). Learn the FCL—in any language—and you're a .NET programmer. "Learn once, work anywhere."

What split the Windows programming world into mutually incompatible Delphi shops, VB shops, and C++ shops was never the languages themselves. Picking up any particular language has always been easy. The barriers to entry have always been the different libraries. Using a different language meant learning a new library. Learning a new library meant that every little thing required a documentation search; your productivity was near zero for weeks on end. But with .NET, once you learn the Framework Classes, you can easily move from project to project and from job to job.

What's more, in this bigger, broader job market, Delphi skills are a big advantage. .NET is not a knock-off or successor to Delphi, and there are significant differences between Delphi and .NET—but .NET **is** a lot like Delphi. .NET has components, events, exceptions, interfaces, properties, and objects that descend via single inheritance from a common ancestor. All just like Delphi. .NET has more in common with Delphi than it does with either MFC, ATL, or VB, and so Delphi programmers will find .NET easier to learn than VB or C++ programmers will.

This book presents .NET from a Delphi programmer's viewpoint. It doesn't ask you to plow through things you already know in the hopes of picking up a few choice bits of new information; it presents the core concepts of the .NET world in terms of the Delphi concepts you're familiar with. The examples are in either C# or Delphi, not both—unless I'm trying to highlight a syntax difference.

From your employer's point of view, .NET offers managed code plus most of Delphi's traditional productivity advantages, without Delphi's traditional drawback of being a niche product that few programmers know. From your point of view, .NET offers something like a hundred times as many possible jobs—and it puts the fun back in programming. Garbage collection frees us from the tyranny of Free What You Create and all the petty discipline of avoiding memory leaks. We can write functions that return objects; we never have to worry about a "tombstoned pointer" to a prematurely freed object leading to memory corruption.

.NET is fun. .NET is productive. .NET offers what you've always loved about Delphi, without locking you into a narrow ghetto. This book will help you transfer your Delphi skills to the broader, brighter world outside the ghetto walls.

Organization

As I wrote this book, I tried to write the book I wish I'd had when I was learning .NET. I tried to remember what I found confusing, and what key points made for Aha! moments. At the same time, I imagined the reactions of people I've worked with, or met online, and went into more detail on the points where they would be confused or argumentative. Hopefully, the result will spare you a lot of trial and error.

I assume you know Delphi well enough to get paid to write it, but I've tried very hard to avoid ambiguity and unexplained jargon. You should be able to read this book straight through, and understand it all well enough to go out and get yourself in trouble—you should not have to reread any section two or three times before it makes sense. (I also know that many people will **not** read the book straight through, and have provided plenty of parenthetical cross-references for the reader who wants to skip around, or who will only open the book on an "as needed" basis.)

The first part of the book is for a native code programmer (i.e., Win32 or Linux) with no *managed code* experience. Chapter 1 explains what managed code is, and how it makes you even more productive than you are with Delphi. Chapter 2 introduces the .NET programming model, and how it differs from the familiar Delphi programming model. Chapter 3 has garbage collection details, while Chapter 4 goes into more detail about how Just In Time (JIT) compilation works, and why .NET uses JIT compilation. Most of the examples in the first four chapters are in Delphi, except where I use a little C# to introduce generics in Chapter 2.

The second part of the book is (mostly) a Delphi programmer's introduction to C#. While you can probably decipher C# examples on your own, I think you'll find that Part 2 makes it easier—and that reading the C# chapters will make it **much** easier to actually write C#.

The third part of the book covers the .NET Framework Class Library, or FCL. This part is nearly as long as the first two parts put together, and is very much the heart of the book. The Microsoft documentation is a fine reference when you know what class to use, but it's not a particularly good introduction. I've tried to provide the conceptual overview that you need to make sense of the documentation and/or to ask questions that Google can answer. After reading Chapters 11 through 18, you should understand the FCL design philosophy well enough that you'll find it easy to learn new parts of the library. There are Delphi examples in every chapter, but most of the FCL examples are in C#.

Note This is not a Delphi book—this is a book about .NET, for Delphi programmers.

Typography

Inline code looks like `this`, and I use bold for **emphasis** and italics as a sort of quote, to introduce *new terms*.

I also distinguish single quotes from double quotes. A double quote is a 'strong' or 'true' quote, while a single quote is a *scare quote*—a 'weak' or 'sort of' quote. (Other former philosophy majors will find this convention familiar; my copy editor suggested that I may need to explain it to everyone else.)

That is, if I say that Benjamin Franklin said "Thank you," I'm saying that I am 100% sure that Benjamin Franklin said "Thank you" on at least one occasion. I use double quotes when I'm actually quoting something I've read or heard. By contrast, if I say that Benjamin Franklin said 'Those who will sacrifice Freedom for the sake of Security will soon find they have Neither,' I'm saying that Benjamin Franklin said something like that. I use single quotes when I'm paraphrasing, or when I'm using slang or a neologism.

The Sample Code

There are over 150 sample projects mentioned in this book. For the most part, I only print the few most interesting lines of each. In some cases, I don't even do that—I describe a technique, and refer you to a sample project for the details. To run the projects and/or read the code that I don't print, you'll have to download the appropriate zip file from the Apress web site, and install it on a machine with a .NET development environment.

You can get the code by going to the Source Code section of the Apress web site, `www.apress.com`, where you'll find complete download and installation instructions. I urge you to download the sample code. Reading the code and pressing F1 on various identifiers is a great way to dip into the .NET documentation. More importantly, while I've made every effort to keep the book self-contained so that you can read it away from a computer, some techniques are best grasped by experimentation. Using my working code as a starting point can be very helpful here. (Most of the projects are just snippets that demonstrate a single point, but there **are** a few that contain code you may want to borrow.) All the sample code—from the code that demonstrates various useful techniques to the utility units in my `common` directory—is distributed under a license that lets you use my code in any way you like, so long as you leave my copyright notice in the source code.