Pro WPF in C# 2008

Windows Presentation Foundation with .NET 3.5

SECOND EDITION

Matthew MacDonald

Apress®

Pro WPF in C# 2008: Windows Presentation Foundation with .NET 3.5, Second Edition

ISBN-13 (pbk): 978-1-4302-0576-0

ISBN-10 (pbk): 1-59059- 955-1

ISBN-13 (electronic): 978-1-59059-955-6

ISBN-10 (electronic): 1-4302-0576-8

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

Lead Editor: Matt Moodie
Technical Reviewer: Christophe Nasarre
Editorial Board: Clay Andres, Steve Anglin, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Kevin Goff, Matthew Moodie, Joseph Ottinger, Jeffrey Pepper, Frank Pohlmann, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Project Manager: Sofia Marchant
Copy Editor: Kim Wimpsett
Associate Production Director: Kari Brooks-Copony
Production Editor: Laura Esterman
Compositor: Diana Van Winkle
Proofreader: Nancy Sixsmith
Indexer: Broccoli Information Management
Artists: Diana Van Winkle, 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 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit http://www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at http://www.apress.com/info/bulksales.

The source code for this book is available to readers at http://www.apress.com.

For my wonderful family,
Faria and Maya

Contents at a Glance

Contents

About the Author

MATTHEW MACDONALD is an author, educator, and Microsoft MVP in Windows client development. He's a regular contributor to programming journals and the author of more than a dozen books about .NET programming, including *Pro .NET 2.0 Windows Forms and Custom Controls in C#* (Apress, 2005) and *Pro ASP.NET 3.5 in C# 2008* (Apress, 2007). He lives in Toronto with his wife and daughter.

About the Technical Reviewer

CHRISTOPHE NASARRE is a software architect and development lead for Business Objects, a multinational software company focused on business intelligence solutions. During his spare time, Christophe writes articles for *MSDN Magazine*, MSDN, and ASPToday. Since 1996, he has also worked as a technical editor on numerous books on Win32, COM, MFC, .NET, and WPF. In 2007, he wrote his first book, *Windows via C/C++* from MSPress.

Acknowledgments

No author can complete a book without a small army of helpful individuals. I'm deeply indebted to the whole Apress team, including Sofia Marchant and Laura Esterman, who shepherded this second edition through production, Kim Wimpsett, who speedily performed the copy edit, and many other individuals who worked behind the scenes indexing pages, drawing figures, and proofreading the final copy. I also owe a special thanks to Gary Cornell, who always offers invaluable advice about projects and the publishing world.

Christophe Nasarre deserves my sincere thanks for his unfailingly excellent and insightful tech review comments—they've helped me to fill gaps and improve the overall quality of this book. I'm also thankful for the legions of die-hard bloggers on the various WPF teams, who never fail to shed light on the deepest recesses of WPF. I encourage anyone who wants to learn more about the future of WPF to track them down. Finally, I'd never write any book without the support of my wife and these special individuals: Nora, Razia, Paul, and Hamid. Thanks, everyone!

Introduction

When .NET first appeared, it introduced a small avalanche of new technologies. There was a whole new way to write web applications (ASP.NET), a whole new way to connect to databases (ADO.NET), new typesafe languages (C# and VB .NET), and a managed runtime (the CLR). Not least among these new technologies was Windows Forms, a library of classes for building Windows applications.

Although Windows Forms is a mature and full-featured toolkit, it's hardwired to essential bits of Windows plumbing that haven't changed much in the past ten years. Most significantly, Windows Forms relies on the Windows API to create the visual appearance of standard user interface elements such as buttons, text boxes, check boxes, and so on. As a result, these ingredients are essentially uncustomizable.

For example, if you want to create a stylish glow button you need to create a custom control and paint every aspect of the button (in all its different states) using a lower-level drawing model. Even worse, ordinary windows are carved up into distinct regions, with each control getting its own piece of real estate. As a result, there's no good way for the painting in one control (for example, the glow effect behind a button) to spread into the area owned by another control. And don't even think about introducing animated effects such as spinning text, shimmering buttons, shrinking windows, or live previews because you'll have to paint every detail by hand.

The Windows Presentation Foundation (WPF) changes all this by introducing a new model with entirely different plumbing. Although WPF includes the standard controls you're familiar with, it draws every text, border, and background fill *itself*. As a result, WPF can provide much more powerful features that let you alter the way any piece of screen content is rendered. Using these features, you can restyle common controls such as buttons, often without writing any code. Similarly, you can use transformation objects to rotate, stretch, scale, and skew anything in your user interface, and you can even use WPF's baked-in animation system to do it right before the user's eyes. And because the WPF engine renders the content for a window as part of a single operation, it can handle unlimited layers of overlapping controls, even if these controls are irregularly shaped and partially transparent.

Underlying the new features in WPF is a powerful new infrastructure based on DirectX, the hardware-accelerated graphics API that's commonly used in cutting-edge computer games. This means that you can use rich graphical effects without incurring the performance overhead that you'd suffer with Windows Forms. In fact, you even get advanced features such as support for video files and 3-D content. Using these features (and a good design tool), it's possible to create eye-popping user interfaces and visual effects that would have been all but impossible with Windows Forms.

Although the cutting-edge video, animation, and 3-D features often get the most attention in WPF, it's important to note that you can use WPF to build an ordinary Windows application with standard controls and a straightforward visual appearance. In fact, it's just as easy to use common controls in WPF as it is in Windows Forms. Even better, WPF enhances features that appeal directly to business developers, including a vastly improved data binding

model, a new set of classes for printing content and managing print queues, and a document feature for displaying large amounts of formatted text. You'll even get a new model for building page-based applications that run seamlessly in Internet Explorer and can be launched from a website, all without the usual security warnings and irritating installation prompts.

Overall, WPF combines the best of the old world of Windows development with new innovations for building modern, graphically rich user interfaces. Although Windows Forms applications will continue to live on for years, developers embarking on new Windows development projects should consider WPF.

Tip If you've done a substantial amount of work creating a Windows Forms application, you don't need to migrate it wholesale to WPF to get access to new features such as animation. Instead, you can add WPF content to your existing Windows Forms application, or you can create a WPF application that incorporates your legacy Windows Forms content. Chapter 25 discusses all your interoperability options.

About This Book

This book is an in-depth exploration of WPF for professional developers who know the .NET platform, the C# language, and the Visual Studio development environment. Previous experience with Windows Forms is useful but not required to get the most out of this book.

This book provides a complete description of every major WPF feature, from XAML (the markup language used to define WPF user interfaces) to 3-D drawing and animation. Along the way, you'll occasionally work with code that involves other features of the .NET Framework, such as the ADO.NET classes you use to query a database. These features aren't discussed here. Instead, if you want more information about .NET features that aren't specific to WPF, you can refer to one of the many dedicated .NET titles from Apress.

Chapter Overview

This book includes 26 chapters. If you're just starting out with WPF, you'll find it's easiest to read them in order, as later chapters often draw on the techniques demonstrated in earlier chapters.

The following list gives you a quick preview of each chapter:

Chapter 1: Introducing WPF describes the architecture of WPF, its DirectX plumbing, and the new device-independent measurement system that resizes user interfaces automatically.

Chapter 2: XAML describes the XAML standard that you use to define user interfaces. You'll learn why it was created and how it works, and you'll create a basic WPF window using different coding approaches.

Chapter 3: The Application introduces the WPF application model. You'll see how to create single-instance and document-based WPF applications.

Chapter 4: Layout delves into the layout panels that allow you to organize elements in a WPF window. You'll consider different layout strategies, and you'll build some common types of windows.

Chapter 5: Content describes the WPF content control model, which allows you to place elements *inside* other elements to customize the look of common controls such as buttons and labels.

Chapter 6: Dependency Properties and Routed Events describes how WPF extends .NET's property and event system. You'll see how WPF uses dependency properties to provide support for key features such as data binding and animation, and how it uses event routing to send events bubbling or tunneling through the elements in your user interface.

Chapter 7: Classic Controls considers some of the common controls every Windows developer is familiar with, such as buttons, text boxes, and labels—and their WPF twists.

Chapter 8: Windows examines how windows work in WPF. You'll also learn how to create irregularly shaped windows and use Vista glass effects.

Chapter 9: Pages and Navigation describes how you can build pages in WPF and keep track of navigation history. You'll also see how to build a browser-hosted WPF application that can be launched from a website without a tedious installation step.

Chapter 10: Commands introduces the WPF command model, which allows you to wire multiple controls to the same logical action.

Chapter 11: Resources describes how resources let you embed binary files in your assembly and reuse important objects throughout your user interface.

Chapter 12: Styles explains the WPF style system, which lets you apply a set of common property values to an entire group of controls.

Chapter 13: Shapes, Transforms, and Brushes introduces the 2-D drawing model in WPF. You'll learn to create shapes, alter elements with transforms, and paint exotic effects with gradients, tiles, and images.

Chapter 14: Geometries, Drawings, and Visuals delves deeper into 2-D drawing. You'll learn to create complex paths that incorporate arcs and curves, how to use complex graphics efficiently, and how to use the lower-level visual layer for optimized drawing.

Chapter 15: Control Templates shows you how you can give any WPF control a dramatic new look (and new behavior) by plugging in a customized template. You'll also see how templates allow you to build a skinnable application.

Chapter 16: Data Binding introduces WPF data binding. You'll see how to bind any type of object to your user interface, whether it's an instance of a custom data class or the full-fledged ADO.NET DataSet. You'll also learn how to convert, format, and validate data.

Chapter 17: Data Templates, Data Views, and Data Providers shows some of the tricks for designing professional data-driven interfaces. Along the way, you'll build rich data lists that incorporate pictures, controls, and selection effects.

Chapter 18: Lists, Trees, Toolbars, and Menus considers WPF's family of list controls. You'll see data-oriented controls such as grids and trees, and command-oriented controls such as toolbars and menus.

Chapter 19: Documents introduces WPF's rich document support. You'll learn to use flow documents to present large amounts of text in the most readable way possible, and you'll use fixed documents to show print-ready pages. You'll even use the RichTextBox to provide document editing.

Chapter 20: Printing demonstrates WPF's new printing model, which lets you draw text and shapes in a print document. You'll also learn how to manage page settings and print queues.

Chapter 21: Animation explores WPF's animation framework, which lets you integrate dynamic effects into your application using straightforward, declarative markup.

Chapter 22: Sound and Video describes WPF's media support. You'll see how to control playback for sound and video, and how to throw in synchronized animations and live effects.

Chapter 23: 3-D Drawing explores the support for drawing 3-D shapes in WPF. You'll learn how to create, transform, and animate 3-D objects. You'll even see how to place interactive 2-D controls on 3-D surfaces.

Chapter 24: Custom Elements explores how you can extend the existing WPF controls and create your own. You'll see several examples, including a template-based color picker, a masked text box, and a decorator that performs custom drawing.

Chapter 25: Interacting with Windows Forms examines how you can combine WPF and Windows Forms content in the same application—and even in the same window.

Chapter 26: Multithreading and Add-Ins describes two advanced topics. You'll use multithreading to create responsive WPF applications that perform time-consuming work in the background. You'll use the add-in model to create an extensible application that can dynamically discover and load separate components.

Chapter 27: ClickOnce Deployment shows how you can deploy WPF applications using the ClickOnce setup model introduced in .NET 2.0.

What You Need to Use This Book

WPF exists in two versions. The original version was released with .NET 3.0 and shipped with Windows Vista. The second (slightly improved) version was released with .NET 3.5. Incidentally, the second version of WPF is named WPF 3.5 to match the version of the .NET Framework.

This book assumes you're using the latest-and-greatest version, .NET 3.5. All the downloadable examples use Visual Studio 2008 projects and target .NET 3.5. However, most of the concepts you'll learn apply equally well to .NET 3.0. For more information about the refinements that were added to WPF in .NET 3.5, refer to the section "The Evolution of WPF" in Chapter 1.

In order to *run* a WPF 3.5 application, your computer must have Microsoft Windows Vista or Microsoft Windows XP with Service Pack 2. You also need the .NET Framework 3.5.

Note In this book, frequent mention is made to Windows Vista and Windows XP—the two client operating systems that WPF supports. It's easy to overlook that WPF actually runs on two related server versions of Windows: Windows Server 2003 and Windows Server 2008.

In order to *create* a WPF 3.5 application (and open the sample projects included with this book), you need Visual Studio 2008, which includes the .NET Framework 3.5.

There's one other option. Instead of using any version of Visual Studio, you can use Expression Blend—a graphically oriented design tool—to build and test WPF applications. Overall, Expression Blend is intended for graphic designers who spend their time creating serious eye candy, while Visual Studio is ideal for code-heavy application programmers. This book assumes you're using Visual Studio. If you'd like to learn more about Expression Blend, you can consult one of many dedicated books on the subject.

Some of the examples in this book use ADO.NET data access code to query a SQL Server database. To try out these examples, you can use the script file that's included with the sample code to install the database (on SQL Server version 2000 or later). Alternatively, you can use a file-based database component that's also included with the sample code. This component retrieves the same data from an XML file, simulating the work of the full database component without requiring a live instance of SQL Server.

Code Samples and URLs

It's a good idea to check the Apress website or `http://www.prosetech.com` to download the most recent up-to-date code samples. You'll need to do this to test most of the more sophisticated code examples described in this book because the less significant details are usually left out. This book focuses on the most important sections so that you don't need to wade through needless extra pages to understand a concept.

To download the source code, surf to `http://www.prosetech.com` and look for the page for this book. You'll also find a list of links that are mentioned in this book, so you can find important tools and examples without needless typing.

Feedback

This book has the ambitious goal of being the best tutorial and reference for programming WPF. Toward that end, your comments and suggestions are extremely helpful. You can send complaints, adulation, and everything in between directly to `apress@prosetech.com`. I can't solve your .NET problems or critique your code, but I will benefit from information about what this book did right and wrong (or what it may have done in an utterly confusing way).