Scott Dorman
Joe Healy
Nikita Polyakov
Kevin Wolf

Foreword by Brandon Watson
Senior Director, Developer Platform & Ecosystem

Sams Teach Yourself Windows® Phone 7 Application Development in 24 Hours

SAMS

Scott Dorman
Kevin Wolf
Nikita Polyakov
Joe Healy

Sams **Teach Yourself**

Windows® Phone 7 Application Development in 24 Hours

SAMS 800 East 96th Street, Indianapolis, Indiana, 46240 USA

Sams Teach Yourself Windows® Phone 7 Application Development in 24 Hours

Copyright © 2012 by Pearson Education

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-672-33539-6
ISBN-10: 0-672-33539-5

Library of Congress Cataloging-in-Publication Data

Sams teach yourself Windows Phone 7 application development in 24 hours / Scott Dorman ... [et al.].

p. cm.

Includes index.

ISBN-13: 978-0-672-33539-6

ISBN-10: 0-672-33539-5

1. Windows phone (Computer file) 2. Application software—Development. 3. Smartphones—Programming. 4. Mobile computing. I. Dorman, Scott, 1973- II. Title: Windows Phone 7 application development in 24hours.

QA76.76.A65S254 2012

004.16'7—dc23

2012007173

Printed in the United States of America

First Printing April 2012

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales

1-800-382-3419

corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales

international@pearsoned.com

Editor-in-Chief
Greg Wiegand

Executive Editor
Neil Rowe

Acquisitions Editor
Brook Farling

Development Editor
Mark Renfrow

Managing Editor
Kristy Hart

Project Editor
Betsy Harris

Copy Editor
Karen Annett

Senior Indexer
Cheryl Lenser

Proofreader
Paula Lowell

Technical Editor
Richard Bailey

Publishing Coordinator
Cindy Teeters

Book Designer
Gary Adair

Senior Compositor
Gloria Schurick

Contents at a Glance

Part I: Fundamentals

HOUR 1 Introducing Windows Phone 7 1
2 Considerations for Desktop Developers 21
3 Choosing an Application Framework 31
4 Learning Silverlight Basics 45
5 Understanding Page-Based Navigation 73
6 Using the Sensors 87

Part II: Developing Applications with Silverlight

HOUR 7 Working with the GPS and the Bing Map Control 99
8 Developing Finger-Friendly Applications 111
9 Consuming Web Data Services 129
10 Working with REST, Syndication Feeds, and OData 141
11 Integrating with Windows Phone Hubs 159
12 Integrating with Microsoft Services 177
13 Performance and Debugging 191
14 Understanding Push Notification Services 207
15 Understanding Live Tiles and Toast Notifications 217
16 Working with Data 231

Part III: Developing Games with the XNA Framework

HOUR 17 Introduction to XNA Programming 249
18 Building a Simple Game 263
19 Adding Polish to Your Game 281

Part IV: Diving Deeper

HOUR 20 Internationalizing Your Application 293
21 Working with Phone Internals 309
22 Being a Good Mobile Citizen 325
23 Distributing Applications with the Marketplace 343
24 Marketing and Selling Your Application 359
Resources 371
Index 373

Table of Contents

Introduction **xvii**

Audience and Organization xviii
Conventions Used In This Book xix
Closing Thoughts xix

Part I: Fundamentals

HOUR 1: Introducing Windows Phone 7 **1**

Experiencing Life in Motion 2
Ensuring a Common Foundation 3
Introducing the Application Platform 6
Introducing Metro 11
Summary 18
Q&A 19

HOUR 2: Considerations for Desktop Developers **21**

Understanding the Requirements 21
Identifying Users and Their Mobile Needs 22
Getting Data for Your Application 25
Building the User Interface 27
Summary 29
Q&A 29
Workshop 29

HOUR 3: Choosing an Application Framework **31**

Introducing the Application Frameworks 31
Understanding Silverlight 32
Understanding the XNA Framework 37

Summary 41
Q&A 42
Workshop 43
Exercise 43

HOUR 4: Learning Silverlight Basics 45
Understanding Silverlight 45
Silverlight Coding Fundamentals 46
Exploring Silverlight Controls 49
Understanding Data Binding 62
Summary 70
Q&A 70

HOUR 5: Understanding Page-Based Navigation 73
Introducing the Page Navigation Model 74
Using the Navigation Service 75
Working with Page Navigation Events 79
Passing Data with the Navigation Context 82
Summary 83
Q&A 83
Workshop 84
Exercise 85

HOUR 6: Using the Sensors 87
Working with the Accelerometer 87
Using the Compass 91
Responding to the Gyroscope 93
Determining Motion 95
Summary 97
Q&A 97
Workshop 97

Part II: Developing Applications with Silverlight

HOUR 7: Working with the GPS and the Bing `Map` Control **99**
Building a Location-Aware Application 99
Working with the GPS 101
Using the Silverlight Bing `Map` Control 104
Drawing on the Map 106
Summary 107
Q&A 108
Workshop 108
Exercise 109

HOUR 8: Developing Finger-Friendly Applications **111**
Working Within Phone Screen Limitations 111
Understanding Touch Targets 113
Working with Gestures 116
Summary 126
Q&A 126
Workshop 127
Exercises 127

HOUR 9: Consuming Web Data Services **129**
Using Raw Web Communication 129
Using Windows Communication Foundation 135
Supporting Sockets 139
Summary 139
Q&A 140
Workshop 140
Exercise 140

HOUR 10: Working with REST, Syndication Feeds, and OData **141**
Consuming REST Services 141
Working with Syndication Feeds 147
Using the OData Libraries 151

Summary 156
Q&A 156
Workshop 157
Exercise 157

HOUR 11: Integrating with Windows Phone Hubs 159
Introducing the Windows Phone Hubs 160
Integrating with the Music + Videos Hub 164
Integrating with the Pictures Hub 170
Summary 174
Workshop 174
Exercise 175

HOUR 12: Integrating with Microsoft Services 177
Integrating with Microsoft Bing 177
Translating Text 185
Summary 188
Workshop 188
Exercise 189

HOUR 13: Performance and Debugging 191
Tuning Application Startup 191
Measuring and Tuning Your Application 195
Working with Asynchronous and Background Tasks 198
Tuning Silverlight Performance 201
Summary 203
Q&A 203
Workshop 204
Exercises 205

HOUR 14: Understanding Push Notification Services 207
What Are Push Notification Services? 207
How Do Notification Services Work? 208
Opening a Notification Channel 210

Sending and Receiving Notifications 212
Summary 215
Q&A 216
Workshop 216
Exercise 216

HOUR 15: Understanding Live Tiles and Toast Notifications **217**

What Are Live Tile and Toast Notifications? 217
Working with Toast Notifications 219
Live Tile Notifications 223
Summary 227
Q&A 228
Workshop 228
Exercise 229

HOUR 16: Working with Data **231**

Determining Network Connectivity 231
Managing Local Data 232
Working with Data from a Service 243
Keeping Data Synchronized 245
Enforcing Business Rules 246
Summary 246
Q&A 246
Workshop 247
Exercises 248

Part III: Developing Games with the XNA Framework

HOUR 17: Introduction to XNA Programming **249**

Understanding the Game Loop 249
Adding Game Content 253
Rendering Textures 254
Rendering Text 255
Adding a Background 258

Handling User Input 259
Summary 261
Q&A 261
Workshop 261

HOUR 18: Building a Simple Game **263**
Game Physics Made Simple 263
Detecting Collisions 269
Using Accelerometer-Controlled Movement 272
Adding the Challenges 274
Summary 278
Q&A 278
Workshop 278
Exercise 279

HOUR 19: Adding Polish to Your Game **281**
Adding Gameplay 281
Enhancing the Ball-Dropping Experience 287
Adding Haptic and Audible Feedback 289
Summary 290
Q&A 290
Workshop 291

Part IV: Diving Deeper

HOUR 20: Internationalizing Your Application **293**
Understanding Internationalization 293
Understanding Culture 295
Globalization at Work 297
Localizing Application Resources 300
Summary 306
Q&A 307
Workshop 307
Exercise 308

HOUR 21: Working with Phone Internals 309
Understanding Launchers and Choosers 309
Sending Messages 311
Interacting with the Marketplace 314
Playing Media Content 316
Making Phone Calls 317
Searching and Web Browsing 319
Working with the Camera and Pictures 320
Playing FM Radio 322
Summary 323
Q&A 323
Workshop 324
Exercise 324

HOUR 22: Being a Good Mobile Citizen 325
Theming 325
Preserving Application State 334
Implementing an About Page 340
Summary 341
Q&A 341
Workshop 342
Exercise 342

HOUR 23: Distributing Applications with the Marketplace 343
Introducing the Marketplace 343
Joining the Marketplace 348
Configuring Your Application 349
Submitting Your Application 352
Using Trial Services 356
Summary 357
Q&A 357
Workshop 357
Exercise 358

HOUR 24: **Marketing and Selling Your Application** 359

Marketing Using Basic Techniques 359
Using Trial Services 361
Generating Sales Using the About Page 367
Understanding the Effects of Globalization 367
Entering Contests 369
Summary 370
Q&A 370

Resources 371

Author Resources 371
Other Helpful Resources 371

Index 373

About the Authors

Scott Dorman has been designated by Microsoft as a C# Most Valued Professional in recognition for his many contributions to the C# community. Scott has been involved with computers in one way or another for as long as he can remember. He has been working with computers professionally since 1993 and with .NET and C# since 2001. Currently, Scott's primary focus is developing commercial software applications using Microsoft .NET technologies. Scott runs a software architecture–focused user group, speaks extensively (including at Microsoft TechEd and community-sponsored code camps), and contributes regularly to online communities such as The Code Project and StackOverflow. Scott also maintains a .NET Framework and C#-focused technology blog at http://geekswithblogs.com/sdorman.

Kevin Wolf is a software engineer with 20 years of professional experience developing and deploying software. Kevin's background includes everything from developing hardware and software for real-time embedded systems, to CMS/Sales Force Automation systems, to large decision-support, line-of-business applications. For the past five years, Kevin's focus as an independent consultant has been on developing applications for mobile devices such as Windows Mobile, iPhone, and Android. In January of 2010, Kevin was recognized as a Microsoft MVP in the area of Device Application Development for his work promoting Windows Mobile.

Nikita Polyakov is a Microsoft Windows Phone Development MVP and has been a long-time community voice for most things mobile from Microsoft in the local community in Florida and online. Nikita helped and cochaired many local user groups and community initiatives. Nikita is a software professional with 10 years of experience building custom software using Microsoft platforms for many local and national customers. Nikita is passionate about the power of technology and enabling people to realize their full potential through quality software solutions.

Joe Healy is a Microsoft Developer Evangelist based in Florida. Joe's geographical responsibility is to provide "developer care" for the Gulf States District for the East Region Developer team. As such, he has responsibilities to bring .NET development to the masses in his area. Joe serves a multitude of clients, from corporate accounts to broad-reach events and user groups. He lectures on various development and architectural topics focused around the .NET Frameworks, Visual Studio .NET, and associated servers.

Dedications

Scott Dorman: *This book is first and foremost dedicated to Nathan, whom I hope follows in my footsteps and someday writes books of his own. Thank you for giving me a unique perspective and showing me the world through the eyes of a child.*

Kevin Wolf: *To my father who introduced me to computers at an early age by taking me to work with him on Saturday mornings and letting me play with a green screen terminal while he organized the punch cards. Thank you for providing me the building blocks necessary to get where I am today.*

Rest in Peace, Dad,
Donald Elwood Wolf 1944–2010

Nikita Polyakov: *Dedicated to my late grandfathers, my family, and my friends for your support and encouragement. Also everyone who mentored or took a chance on me.*

Joe Healy: *Dedicated to my geek girls, Ryan and Morgan, and my wife, Anne, for putting up with my "coding moods." Thanks to my mom and dad for my IIe in high school, and loaning me the $2,000 for the AT-Clone in college. I'm sure the accrued interest is pretty high, since I never paid you back. Thanks to my management at Microsoft for encouraging me and tolerating my many impulses and even more CLMs. And thanks to cutter, scumpy, and foghat: At some point in my computer career, you guys all taught me I still had a long way to go. Fla-wimo-dev, y'all rock. Nuff said.*

Acknowledgments

Scott Dorman: I want to thank Joe for giving me the idea of getting all of us together to write this book. Everyone did a great job supporting each other, and prodding when needed, to make this book a reality. Thank you to everyone at Sams Publishing, without whom the book would have never been published. I also want to thank Eric Lippert for recommending our technical editor, Richard, who has provided great comments and insight. Of course, without the Windows Phone 7 team, there wouldn't be anything for us to write about in the first place. Finally, I have to thank my family for being so patient and understanding of the time I wasn't able to spend with them while working on this book. Although Nathan is still too young right now to understand it, I hope he will appreciate it as he gets older.

Kevin Wolf: Special thanks go out to Russ Fustino of Russ' ToolShed Network for his contributions to Hour 24, "Marketing and Selling Your Application." Sharing his experience including thinking "out of the box" in the areas of marketplace publishing, marketing basics, globalization, and instrumentation will hopefully help all who read this book. This was accomplished by sharing his real-world metrics on the "Shortcuts" Phone 7 series of apps. These apps help Phone 7 users increase productivity for a wide variety of products, including Visual Studio, Expression Studio, Windows 7, Office, Firefox, Internet Explorer, SharePoint, and SQL Server.

Nikita Polyakov: Thanks to Joe, Kevin, and Scott for putting up with me over this entire process, and all the editors and staff who had to read my version of English. Also everyone in the Microsoft developer community for inspiring me much in the same way I hope I do today for others. Last but not least, thank you to everyone at Microsoft for a stellar product and the exceptional documentation of Windows Phone 7. A special thank you to my father, who gave me my very first English language book—*Visual Basic*.

We Want to Hear from You

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger.

Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message.

When you write, please be sure to include this book's title and author, as well as your name and contact information. I will carefully review your comments and share them with the author and editors who worked on the book.

Email: feedback@samspublishing.com

Mail: Neil Rowe
Executive Editor
Sams Publishing
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

Introduction

When mobile phones were first introduced in the early 1970s, they were capable of doing nothing more than placing telephone calls. When the first 1G cellular network was introduced in 1979, mobile phones hadn't changed much in features but were beginning to decrease in size. In 1991, the first 2G network was introduced and began the transition from simple mobile phones to "smart" phones that could send and receive email, browse the web, and take pictures, in addition to being able to place phone calls. Nearly 30 years after their introduction, in 2001, the cellular networks began providing the necessary infrastructure to support more modern mobile phones with faster data connections and even greater capabilities.

Since then, mobile phones have become nearly ubiquitous in modern life. The features and capabilities have continued to improve, expand, and evolve. These advances have come with increasing network capabilities and speeds, such as 3G, HSDPA, and 4G. Despite all of these advances, the mobile phone landscape hadn't really changed much.

In 2007, Apple introduced the first iPhone with overwhelming response. The iPhone helped redefine the mobile phone landscape and introduced consumers to a much friendlier, easier-to-use touch-aware device. Quickly following Apple's success, Google introduced the first Android-based phones in late 2008. Once again, the mobile phone landscape changed dramatically. Both the iPhone and Android phones changed the mobile landscape and solidified the concept that these devices are more than just phones. The incredibly large assortment of applications available for both allows you to create a phone that fits your individual lifestyle…almost. The one thing that all of these devices have in common is that, although it has become easier to do more things with your phone, it still requires you to "stop and stare." For example, to see how many unread emails you have in your Inbox, you must still navigate to the email application, start it, and then you can see how many unread emails you have waiting.

During this time, Microsoft kept fairly quiet about its response to both the iPhone and Android phones, seemingly content to continue releasing Windows Mobile 6–based phones. However, with the release of Windows Phone 7, all of that has changed.

Windows Phone 7 introduced an entirely new mobile operating system, user interface, and development experience. Borrowing a page from the Apple playbook, Microsoft established minimum hardware requirements for all phones that run Windows Phone 7. The user interface, code-named Metro, is a typography-based design language that is modern, clean, and simple. The idea behind Metro is similar to that of the visual universal language you see in airports and large cities to clearly direct you to the content you want. If the current smartphones available could be categorized as "stop and stare," Windows Phone 7 phones could be categorized as "glance and go" through the use of hubs and tiles. For example, checking how many unread emails you have in your Inbox on a Windows Phone 7 device simply means unlocking the screen and glancing at the Outlook tile.

From an application-development perspective, Microsoft is leveraging the existing .NET developer base, potentially turning any C# or VB developer who understands Silverlight or the XNA Framework into a mobile application developer. This allows you to take your existing development skills with you when you build a mobile application.

Audience and Organization

This book is targeted toward the .NET programmer who is building his or her first mobile application or first Windows Phone 7 application. If you are first learning how to program, this book can help you on your way but it isn't intended to be a beginning programming book. The book is designed with the purpose of getting you familiar with how to develop a Windows Phone 7 application and allowing you to become productive as quickly as possible.

This book is divided into four parts, each one focusing on a different aspect of Windows Phone 7 application development. These parts progress from the simple fundamentals to more advanced topics, so I recommend reading them in order.

- Part I, "Fundamentals," provides the Windows Phone 7 essentials, including an overview of the development tools, the Metro user interface, considerations for desktop developers moving to a mobile device, and how to choose your application development model.
- Part II, "Developing Applications with Silverlight," teaches you the fundamentals of page-based navigation, developing finger-friendly applications, integrating with a hub, how to store data using isolated storage and how to consume OData and WCF services. You also learn how to use the notification model, Toast notifications, and Live Tiles.
- Part III, "Developing Games with the XNA Framework," starts with an introduction to the XNA Framework and goes on to build a simple XNA-based game.
- Part IV, "Diving Deeper," introduces the advanced concepts of localization and internationalization. Finally, we look at what it takes to create an application that is a good mobile citizen and how to distribute your application using the Windows Marketplace.

Throughout the book, we try to use examples that show real-world problems and how to solve them using a mobile application.

Conventions Used in This Book

This book uses several design elements and conventions to help you prioritize and reference the information it contains:

By the Way

By the Way boxes provide useful sidebar information that you can read immediately or circle back to without losing the flow of the topic at hand.

Did you Know?

Did You Know? boxes highlight information that can make your programming more effective.

Watch Out!

Watch Out! boxes focus your attention on problems or side effects that can occur under certain situations.

In addition, this book uses various typefaces to help you distinguish code from regular English. Code is presented in a `monospace` font. Placeholders—words or characters that represent the real words or characters you would type in code—appear in *`italic monospace`*. When you are asked to type or enter text, that text appears in **bold**.

Some code statements presented in this book are too long to appear on a single line. In these cases, a line-continuation character (➥) is used to indicate that the following line is a continuation of the current statement.

Closing Thoughts

Windows Phone 7 represents a radical shift in the mobile landscape—for application developers, consumers, and Microsoft—bringing in an entirely new way to think about mobile applications and how people use them. You won't be an expert in Windows Phone 7 application development when you finish this book, but we hope you will feel comfortable about being able to create robust and highly effective mobile applications for Windows Phone 7.

This page intentionally left blank

HOUR 3

Choosing an Application Framework

What You'll Learn in This Hour:

- Introducing the application frameworks
- Understanding Silverlight
- Understanding the XNA Framework

The application framework you choose impacts not only the amount of work you must do to complete the application but also the inherent features available to you. Different application frameworks are more suited to different types of applications and determine the default look and feel of your application, instrumentation, and available resources. There are two application frameworks, Microsoft Silverlight and Microsoft XNA Framework, available, each offering different strengths and weaknesses. By the end of this hour, you will be introduced to both the Silverlight and XNA frameworks.

Introducing the Application Frameworks

Silverlight is primarily for text-based applications, whereas the XNA Framework is primarily used for games. Both of these frameworks are flexible and can be adapted so that your imagination is the only limit to what can be created.

Regardless of which framework you decide to work with, you will still have the benefits of the Windows Phone Application Platform. The toolset is free, a rich application programming interface (API) set is available across the entire platform, all applications are sandboxed, and finished products can utilize Marketplace distribution routes.

By the Way

Choosing Your Framework

To choose an application framework, you need to have a very good understanding of your application idea itself. To help, consider writing a paragraph about what you want your application to do. You can also note other applications with a similar style to what you envision. Remember, a little preparation goes a long way.

By the Way

Directly Porting Applications

Although not necessarily a best practice, some developers have insisted on having the exact same user interface (UI) ported from an iPhone or Android application to Windows Phone. Typically, such directly ported applications do not fit the Metro style and trying to adapt a Metro template in Silverlight is not usually easy.

As a result, iPhone or Android developers may find using the XNA Framework more adapted to their particular user-interface styles and provides an excellent route for directly porting the user interface from other platforms.

Understanding Silverlight

Silverlight for Windows Phone is based on the Silverlight 3.0 out-of-browser model and is targeted at media-based applications and rich, interactive text applications. The Windows Phone Silverlight runtime has been optimized for the tight memory environment of Windows Phone.

From a programmer's perspective, Silverlight is an event-driven programming framework consisting of declarative markup, known as Extensible Application Markup Language (XAML), with supporting code. Silverlight projects can also have graphics, audio, and video resources. For controls, Silverlight uses a visual control tree, and when an item is added to that control tree, it is there until you remove it.

Silverlight offers a visual control framework for tasks such as input, user-interface (UI) rendering, media playback, controls, layouts, data binding, gestures, software input panels, and more. The Windows Phone developer tools also provide Microsoft Visual Studio project templates for basic applications, Panorama applications, and Pivot applications, which conform to the Metro design guidelines that can be used to jump-start your application development.

GO TO ▶ HOUR 1, "INTRODUCING WINDOWS PHONE 7," for more information on the Metro UI design guidelines.

Silverlight offers a number of unique advantages to application developers and are the most common applications in the Windows Phone Marketplace. These applications are easy to build and are easy to polish.

Visual controls make a developer's life easier and represent prebuilt templates for a number of common tasks. Silverlight for Windows Phone ships with a large number of stock controls, including `Button`, `CheckBox`, `TextBox`, `TextBlock`, `WebBrowser`, `MediaElement`, the Bing `Map` control, and more. The default control toolbox for Windows Phone Controls is shown in Figure 3.1.

By the Way

Silverlight for Windows Phone Versus Silverlight

We are not talking about web-deployed Silverlight applications (coming to the phone via a web browser) that you might be familiar with from other forms of Silverlight programming. On Windows Phone, all Silverlight applications are "out-of-browser" on the device programs and must be installed to the phone through the Windows Marketplace. These applications cannot be downloaded to the phone through the web.

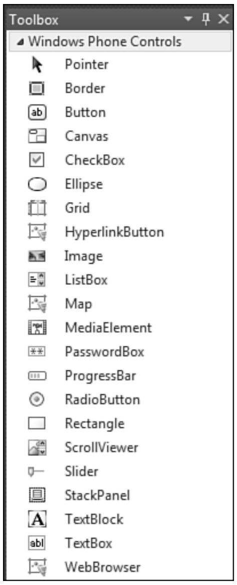
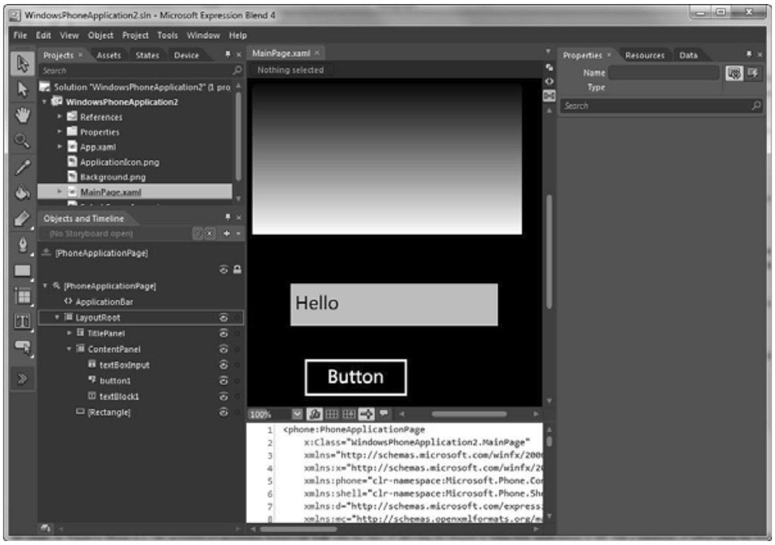


FIGURE 3.1
Default control palette in the Visual Studio Toolbox.

Silverlight also has great visual designers in both Visual Studio and Microsoft Expression Blend to assist with the layout and formatting of Phone pages. Expression Blend, shown in Figure 3.2, focuses on the application designer experience rather than the developer experience and can be used for advanced layout and styling of Silverlight applications. Without the designer, it would be necessary to run the program to visually check layouts and rendering.



```
<phone:PhoneApplicationPage
    x:Class="WindowsPhoneApplication2.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006
    xmlns:x="http://schemas.microsoft.com/winfx/20
    xmlns:phone="clr-namespace:Microsoft.Phone.Co
    xmlns:shell="clr-namespace:Microsoft.Phone.Sh
    xmlns:d="http://schemas.microsoft.com/express
    xmlns:mc="http://schemas.openxmlformats.org/m
```

FIGURE 3.2
Expression Blend visual designer.

Being able to apply styles and resources for a common look and feel can save developers a large amount of effort and work. In web development, Cascading Style Sheets (CSS) are used to provide a common way to define a look and feel that may be applied across all assets in your project. Similarly, Silverlight offers Resource Dictionaries and Styles as a means to style common controls. Styles are commonly used to apply font items such as foreground, style, and family, as well as apply layouts to various flow templates.

Many applications make use of multiple pages and need a way to navigate between those pages. The Silverlight for Windows Phone application framework supports a navigation style similar to that used by web browsers, providing a history stack and backward and forward navigation methods. Applications may navigate from one page to another using document URIs and the NavigationService provided by the page framework. Navigation history is automatically maintained, keeping track of where you navigated from. Using the page navigation framework, Silverlight applications can easily integrate into the Windows Phone navigation model.

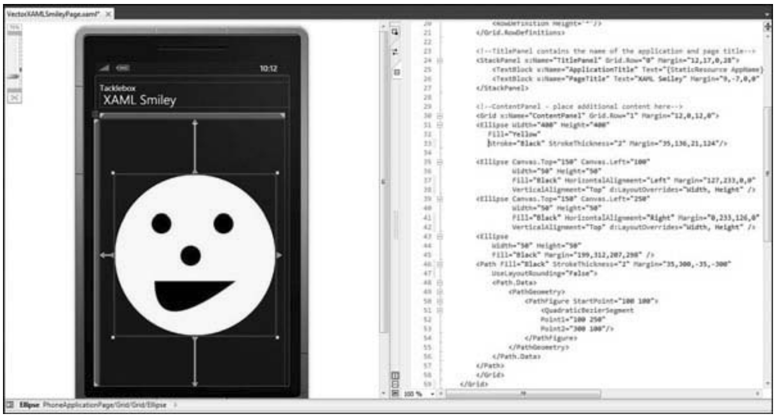
Most programmers have experience in event-driven programming metaphors. Traditionally, line-of-business applications built using VB, C#, or even Java use an event-driven metaphor. Events are raised and methods (either procedures or functions), which subscribe to those events, call the appropriate code in response. For certain types of applications, such as informational or forms-based applications, this is the preferred model and Silverlight fully embraces it.

In keeping with its Windows Presentation Foundation (WPF) base, Silverlight supports vector-based image formats and has a high-performance, vector-based rendering engine built in. Unlike raster, or bitmap, images, which can lose picture quality (called pixelation) when they are scaled to larger sizes, vector images use geometric paths for drawing so they dynamically scale without pixelation. Figure 3.3 shows the XAML for a vector-based smiley face image inside the Visual Studio design surface.

Performance of Vector Graphics

Vector graphics may be slower to render than prerendered, static raster images. Watch out for performance hiccups rendering a complex vector graphic where you might have been able to use a static bitmap instead.

Watch Out!


VectorXAMLSmileyPage.xaml*
10:12
Tacklebox
XAML Smiley
100 %
Ellipse PhoneApplicationPage/Grid/Grid/Ellipse

```
        <RowDefinition Height="*"/>
    </Grid.RowDefinitions>

    <!--TitlePanel contains the name of the application and page title-->
    <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
        <TextBlock x:Name="ApplicationTitle" Text="{StaticResource AppName}
        <TextBlock x:Name="PageTitle" Text="XAML Smiley" Margin="9,-7,0,0"
    </StackPanel>

    <!--ContentPanel - place additional content here-->
    <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
    <Ellipse Width="400" Height="400"
       Fill="Yellow"
       Stroke="Black" StrokeThickness="2" Margin="35,136,21,124"/>

    <Ellipse Canvas.Top="150" Canvas.Left="100"
             Width="50" Height="50"
             Fill="Black" HorizontalAlignment="Left" Margin="127,233,0,0"
             VerticalAlignment="Top" d:LayoutOverrides="Width, Height" />
    <Ellipse Canvas.Top="150" Canvas.Left="250"
             Width="50" Height="50"
             Fill="Black" HorizontalAlignment="Right" Margin="0,233,126,0"
             VerticalAlignment="Top" d:LayoutOverrides="Width, Height" />
    <Ellipse
        Width="50" Height="50"
        Fill="Black" Margin="199,312,207,298" />
    <Path Fill="Black" StrokeThickness="2" Margin="35,300,-35,-300"
        UseLayoutRounding="False">
        <Path.Data>
            <PathGeometry>
                <PathFigure StartPoint="100 100">
                    <QuadraticBezierSegment
                    Point1="100 250"
                    Point2="300 100"/>
                </PathFigure>
            </PathGeometry>
        </Path.Data>
    </Path>
    </Grid>
</Grid>
```

FIGURE 3.3
Vector-rendered smiley face inside Visual Studio.

Silverlight for Windows Phone is a very powerful event-driven programming framework allowing developers to bring their applications to life with a minimum of effort. Although the use of XAML in Silverlight is usually considered a benefit, it can also be a potential drawback, requiring study in the areas of layout, control binding, and other areas to become proficient.

Great Silverlight Examples

Many thousands of applications are available through the Windows Phone Marketplace. The following sections contain a couple of examples of what can be accomplished using Silverlight for Windows Phone.

Facebook

The Facebook application, shown in Figure 3.4, by Microsoft offers a great look at an application styled for Windows Phone. The various levels of information are presented in a flowing manner, and the text fits in well with the Metro style across the phone. Explore how the various screens interact in this application before you begin designing your own.

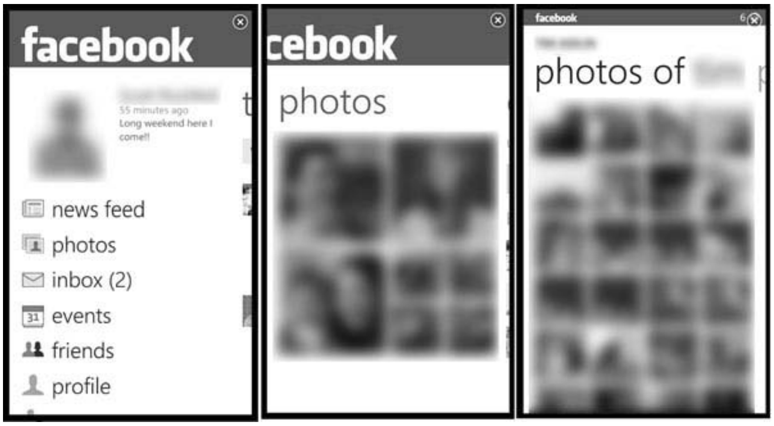


FIGURE 3.4 The Facebook application interface.

Popper 2

Not all Silverlight applications need to adhere to the Metro style. Bill Reiss, of Blue Rose Games (http://www.bluerosegames.com/brg), has published a fantastic bubble breaker game called *Popper 2*, shown in Figure 3.5. Games don't necessarily work very well in the standard Metro styling for Windows Phone, so *Popper 2* uses an alternative user interface to render a Silverlight game on Windows Phone. These innovations include custom menus, screen flow, and much more. This is a great application to explore if you're interested in an alternative Silverlight user-interface style.

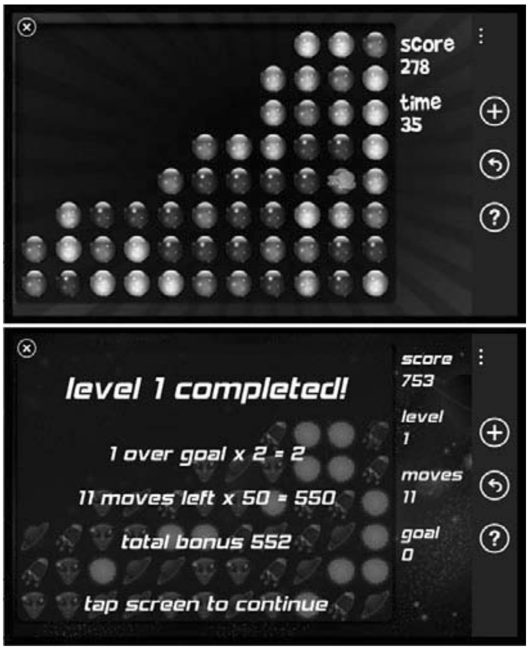


FIGURE 3.5 *Popper 2* menu and screen.

Understanding the XNA Framework

The XNA Framework is primarily targeted at game development. Applications built using the XNA Framework run on a game-optimized managed runtime on top of the Windows Phone operating system. As a general rule, if you're going to write a game, it will be easier to write one on XNA than on Silverlight. XNA offers a unique set of advantages targeted for those wanting very graphically intense applications.

The framework itself is fairly minimalist in regard to user-interface trimmings, such as controls and designers, but provides a lightning-fast runtime execution environment. Programs execute in a loop-based environment, with screen and logic update blocks typically firing 30 times per second. All items on the screen are updated every time the screen is updated, which means it does not show up if you don't draw it. Unlike Silverlight, by default there is no event-driven programming mechanism in XNA.

To build an XNA game, you use Visual Studio and XNA Game Studio Express. In Part III, "Developing Games with the XNA Framework," you will dive deeper into using the XNA Framework.

The XNA Framework, built on top of the .NET Framework, uses a specialized common language runtime optimized for games. This runtime is available not only for Windows Phone, but also for Xbox, Windows 7, Vista, Zune, and Windows XP. A program written for one XNA platform may be retargeted with minimal effort to another platform, so a game written for Windows Phone can potentially be extended to target Windows 7 or any of the other XNA runtime environments fairly easily. This cross-targeting capability makes XNA attractive if you want the potential to port your applications to other platforms.

Because the XNA Framework is targeted for game development, it offers a lower memory footprint than Silverlight and provides a high-performance memory framework designed for the demands of near-real-time games. The math and floating-point computational engines are optimized for the phone. Performance is specifically geared toward graphics with high-performance 2D sprite rendering for rotations, transforms, filtering, and others. For 3D scenarios, the phone also offers hardware-accelerated 3D APIs. As a result, XNA provides the optimal platform for graphics rendering.

Game developers often seek unique and innovative game experiences that will make their products stand out. There are no predefined application templates for XNA as there are for Silverlight, which means each application must design and code its own user interface from scratch. This also means there are no design guidelines related to user interfaces for XNA games.

By the Way

XNA Starter Kits

Although there are no predefined application templates for XNA games, there are many free XNA Starter Kits that can help jump-start your game development. These are available at the following websites:

- http://create.msdn.com/en-US/education/catalog/sample/ui_controls
- http://create.msdn.com/en-US/education/catalog/?contenttype=0&devarea=16&platform=0&sort=1

Working with graphics can be very mathematically intense. To help simplify this, XNA embraces the graphical concept of the sprite, which is a flat, 2D texture drawn on the screen to represent a graphical game object. Programmatically, a sprite is an image resource combined with a position telling XNA where to draw it.

When working with images, instructions can be passed that tell how the graphics should be shaded and is a key advantage for the XNA Framework. The four configurable shading effects provided are the following:

- SkinnedEffect
- EnvironmentMapEffect
- DualTextureEffect
- AlphaTestEffect

By the Way

Image Manipulation in XNA

XNA provides support for drawing, rotating, stretching, scaling, and filtering 2D images in a high-performing manner. By default, the Silverlight framework has no 3D graphics support, whereas the XNA Framework fully supports 3D graphics and offers hardware acceleration for 3D graphical objects for the best performance possible.

In addition to shading effects, the XNA Framework also provides different blend modes to determine how colors interact with each other when they are overlaid and combined. This type of blending is necessary in advanced graphical rendering. Whereas Silverlight for Windows Phone has only Alpha blending, the XNA Framework supports a large number of blend modes, including the following:

- Inversion
- Alpha
- Zero
- One

If you install the Reach graphics demo from the MSDN Creators site, you can see a number of different rich renderings provided by the XNA Framework. This demo runs in both Windows and Windows Phone. Figure 3.6 shows some screenshots from the Reach graphics demo.

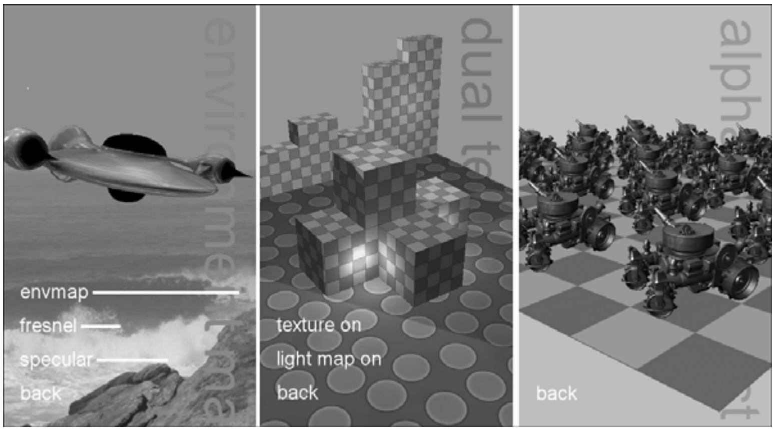


FIGURE 3.6
Reach graphics demo application.

Although there are no predefined application templates for the XNA Framework, there is a Content Pipeline project template, which translates various media sources into XNA consumable formats. This pipeline is extensible for custom formats, but a large number of formats are supported by default. Standard graphics and audio formats are supported as well as AutoDesk, DirectX, Textures, Microsoft Cross-Platform Audio Creation Tool (XACT), and more. If you have media assets in a large variety of formats you need to utilize, XNA is an excellent choice.

As you have seen, the XNA Framework is designed to be a high-performance game programming framework. As a result, there is not really any visual designer support for user-interface work in XNA. There are also no built-in controls, such as text boxes. In fact, rendering text in XNA can be one of the most difficult tasks because of the lack of support for wrapping text controls, flowable control panels, and other text-rendering elements.

Great XNA Examples

You will find a number of great examples of XNA-based games, such as *Glyder: Adventure World*, or *Flowerz*, in the Windows Phone Marketplace. Two sample games, *Catapult Wars* and *Marble Maze*, with complete source code are also available from http://create.msdn.com as part of the Microsoft Educational Resource training materials.

Catapult Wars, shown in Figure 3.7, makes use of moving backgrounds, sounds, graphics, fonts, and other content assets inside the XNA Framework. Reproducing this example on Silverlight would be difficult, although not impossible. (With enough time and custom code, anything is possible!)

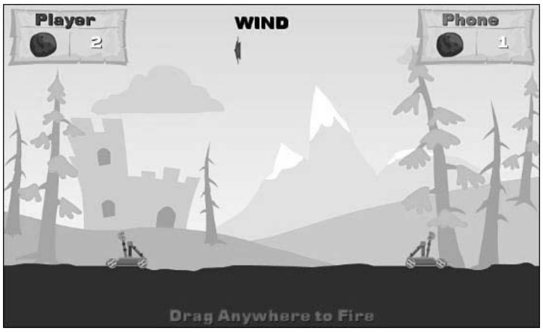


FIGURE 3.7
Catapult Wars.

Presented in tutorial format, *Marble Maze*, shown in Figure 3.8, is another open source sample from Microsoft. The end result shows off the 3D capabilities of Windows Phone 7 and XNA on a very simple level. Even with 3D modeling and code, the game is still smooth and performs well. Developers seeking to learn XNA can use *Marble Maze* as a starting base for exploring 3D programming on Windows Phone.

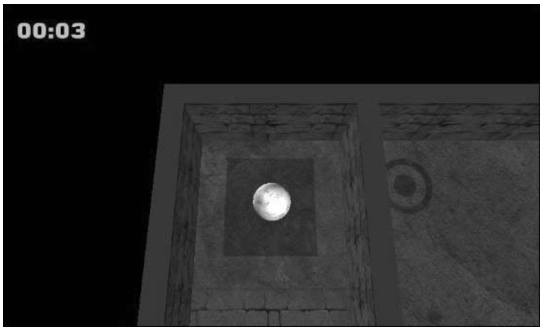


FIGURE 3.8
Marble Maze.

Summary

Silverlight and XNA both have distinct advantages for various application types through built-in constructs, which support the particular types of applications they target. Silverlight is targeted toward textual applications with embedded graphics and video. The Metro styling offers an attractive launching point for building applications. Although Silverlight also provides animation storyboards and timelines, which can be used to develop many simple 2D games, the XNA Framework is targeted at game development and provides a high-performance runtime capable of complex graphical rendering.

Think about the application you are writing, explore some of the applications on the market today to get an idea of what's possible, and begin prototyping with the framework of your choice. Table 3.1 shows a feature comparison between Silverlight and the XNA Framework.

TABLE 3.1 Decision Matrix—Silverlight Versus XNA

Criteria	Silverlight	XNA Framework
2D graphics	✓	✓✓✓
3D graphics		✓✓✓
Controls	✓✓✓	
Cross-targeting	✓	✓✓✓
Event based	✓✓✓	
Game	✓	✓✓✓
Loop based		✓✓✓
Metro UI	✓✓✓	
Text based	✓✓✓	

Q&A

Q. ***What is XNA?***

A. XNA is a high-speed framework designed specifically for game programming; XNA offers a built-in game loop that redraws the screen every 30 seconds. Content Pipelines offer a route for XNA games to consume a variety of audio and graphic content formats.

Q. ***What is Silverlight for Windows Phone?***

A. Silverlight for Windows Phone is a version of Silverlight specifically targeted toward building applications on Windows Phone. Preconstructed templates for Phone, Panorama, and Pivot are available that harness the Metro UI styling built in. Silverlight is a great choice for building text-driven phone applications and applications integrating in media or audio.

Workshop

Quiz

1. What are some of the advantages of the XNA application framework?
2. What are some of the advantages of the Silverlight application framework?

Answers

1. Raw speed is a key advantage of the XNA Framework, as well as the free-form user interface, the built-in game loop, and the extensive graphics and audio support.
2. Some advantages of the Silverlight application framework include the text-based templates for the Metro user interface and the familiar programming metaphor.

Exercise

1. Explore a completed 2D XNA game by downloading and running *Catapult Wars* in the Windows Phone Emulator.
2. Explore a completed 3D XNA game by downloading and running *Marble Maze*.
3. Download the free Facebook application to your Windows Phone and navigate through the various menu styles and information contexts to familiarize yourself with the application.
4. Download the trial of *Popper 2* from the Zune marketplace and explore the game, noting how Silverlight applications are not constrained to Metro UI applications.
5. Think about the application you are envisioning. Write a paragraph about it, and sketch a few screens to help determine which application model fits your desired outcome.

This page intentionally left blank

Index

A

About page
- creating, 340-341
- sales from, 367
- tombstoning and, 341

accelerometer, 5, 87-90
- combining with collision detection, 274-277
- controlling movement with, 272-273

Accelerometer class, 87-90

accent colors, 326, 329-331

accessing APIs, 310

accuracy of position (GPS), 103

Activated event, 336

AddChild method (MapLayer class), 108

adding
- background, 258-259
- game content, 253

addresses (email), choosing and saving, 312-314

advertising in Trial mode applications, 361

All screen (People hub), 160

analytics tools, 10

Android applications, porting user interface from, 32

angles, converting between degrees and radians, 264

animation
- enhancing, 287-288
- frame rates, 264

APIs (application programming interfaces), accessing, 310

App Hub website, 349

AppID
- for Bing API, obtaining, 177-178
- for Translator API, 185

Application Bar, 13-14
- icon color, 329
- localization of, 305-306

application certification, 165

application development life cycle, 10

application frameworks, 31

- choosing, 32, 41-42
- .NET Framework, localization model, 300
- porting user interface from iPhone or Android applications, 32
- Silverlight, 32-35, 45
  - application examples, 35-37
  - Bing Map control, 104-107
  - common language runtime version, 46
  - controls, 49-61
  - data binding, 62-69
  - events, 47-48
  - markup and code-behind, 46-47
  - page navigation model. *See* page navigation model
  - Silverlight Toolkit, 61-62
- XNA Framework, 37-41

application icon, modifying, 13

application identifiers, finding, 315

Application Manifest file, HubType value, 166

Application Platform, 6-10

Application Policy Guidelines document, 346

application programming interfaces (APIs), accessing, 310

application requirements, understanding when porting desktop applications to Windows Phone, 21-22

application resources, localization of, 300-306

application startup, tuning, 191-194

application state, 337

- preserving, 334-339

Application Tiles on Start screen, 12

application title, modifying, 13

applications

- categories, choosing, 360-361
- configuring for Marketplace, 349-352
- debugging and testing for Music + Videos hub, 166-167
- descriptions, writing, 361
- installing large applications, 345
- keywords for, 361
- logo, creating, 360
- marketing, 359-361
  - by entering contests, 369-370
- Marketplace guidelines, 345-346
- sales from About page, 367
- searching in Marketplace, 315
- Silverlight performance tuning, 201-202
- sleeping on startup, 192
- submitting
  - to Marketplace, 352-355
  - in multiple languages, 367-369
- Trial mode, 356, 361-362
  - Shortcuts application case study, 362-367
- tuning performance, 195-198

assisted GPS, 5

asynchronous tasks, 198-200

Atom, 147-151

audible feedback in games, 289-290

Author class, 64

AuthorsCollection class, 68

automatic screen rotation, 90

automatic scrolling in StackPanel control (Silverlight), 56-57

Available Resources list, viewing, 328-329

axes, reversed, 273

Azure, 9

B

Back button, 4
- overriding, 74

background
- adding, 258-259
- colors, 325, 329-331
- tasks, 198-201

backward navigation, 77-78
ball position, updating, 267-269
BallDropping state, 287-288
Base Class Library, 8
BeginRun method, 252
billing support, 10
binding
- data. *See* data binding in Silverlight
- images with themes, 330

Bing API, integration with, 177-184
- AppID, obtaining, 177-178
- protocols for, 178-182
- source types, 179

Bing Map control (Silverlight), 3, 60
- in location-aware applications, 104-107

Bing Maps Developer Accounts, 105
Bing search, 2
Bing Translator API, 185-188
black color scheme, power management and, 326
blend modes in XNA Framework, 39
Bluetooth, support for, 5
browsers, launching, 319-320
business intelligence tools, 10
business rules, enforcing, 246
Button control (Silverlight), 47, 50
buttons, required, 3-4

C

camera
- choosing photos, 321-322
- requirements, 5
- taking photos, 320-321

Camera button, 4
CameraCaptureTask, 320-321
CancelAsync method, 132
CanGoBack property, 77
Canvas control (Silverlight), 52-53
capabilities
- list of, 351-352
- verifying, 351

Capability Detection Tool, 351
capitalization guidelines, 16-17
***Catapult Wars*, 40**
categories
- for applications, choosing, 360-361
- in Marketplace, 347-348

certification, 10, 165, 218
channels. *See* notification channels
CheckBox control (Silverlight), 50-51
choosers, 309-311
- for People hub, 160

choosing
- application categories, 360-361
- application frameworks, 32, 41-42
- email addresses, 312-314
- phone numbers, 317-319
- photos, 321-322

class members
- Accelerometer class, 88
- Compass class, 91
- CompassReading structure, 92
- Gyroscope class, 93
- GyroscopeReading structure, 94
- Motion class, 95
- MotionReading structure, 96

Click event (Silverlight), 48
Closing event, 336
cloud services, 129
- with HttpWebRequest class, 134-135
- list of, 9
- with raw web communication, 129-130

How can we make this index more useful? Email us at indexes@samspublishing.com

with sockets, 139
with WebClient class, 131-133
with Windows Communication Foundation, 135-138
code-behind in Silverlight, 46-47
collections, binding to, 67-69
collision detection, 269-272
combining with accelerometer, 274-277
colors in themes, 325-326, 329-331
ColumnAttribute properties, 233
common foundation
hardware requirements, 3-5
software requirements, 5-6
common language runtime in Silverlight, 46
communication. *See* network communication
compass, 5, 91-92
Compass class, 91-92
CompassReading structure, 92
configuring applications for Marketplace, 349-352
connections, determining network connectivity, 231-232
contacts, 2, 160-161
container controls in Silverlight, 52-57
content assets
adding to games, 253
loading early, 252
Content Pipeline project template in XNA Framework, 40
content processor for sound effects, 289
ContentManager.Load<T> method, 254
contests, entering, 369-370
controls
in Silverlight, 32, 49
container controls, 52-57
input and display controls, 49-52
Panorama and Pivot controls, 57-59
special controls, 59-61
in Silverlight Toolkit, 61-62
theme resources and, 327-329
converting
between degrees and radians, 264
speed to mph/kph, 104
coordinates, origin, 266
costs of Marketplace registration, 3
culture codes, 295-296
CultureInfo class, 296
cultures, 295-297
currency formats, internationalization of, 293

D

data
downloading at application startup, 193-194
organization when porting desktop applications to Windows Phone, 26-27
overflow in Accelerometer class, 89
passing within page navigation model, 82-83
retrieval, understanding when porting desktop applications to Windows Phone, 25
storage when porting desktop applications to Windows Phone, 25
synchronization, 26, 245
validation, 246
from web services, 243-244
data binding in Silverlight, 62-64
to collections, 67-69
creating bindings, 65-67
data contract serializable classes, creating, 240-241
data frameworks, 9
data sources
cloud services, 129
with HttpWebRequest class, 134-135
with raw web communication, 129-130

with sockets, 139
with WebClient class, 131-133
with Windows Communication Foundation, 135-138
web services, 141
OData libraries, consuming, 151-156
REST services, consuming, 141-146
syndication feeds, consuming, 147-151
data templates, ListBox binding with, 69
database models, creating, 232
databases
isolated storage, 238-243
SQL CE relational database, 232-238
DataContext class, creating custom, 235-238
DataContractAttribute class, 240
DataMemberAttribute class, 240
DataServiceCollection class, 153
DataServiceContext class, 153
date formats, internationalization of, 294
DateTime class, instrumenting code with, 196
Deactivated event, 336
debugging, 10, 166-167
declarative binding example, 66
deep links, 355
DefaultTask element (Silverlight), 74
degrees, converting to radians, 264
DependencyObject (Silverlight), 63
descriptions for applications, writing, 361
deserializing lists of data, 242
desktop applications, porting to Windows Phone, 21
application requirements, understanding, 21-22
building user interface, 27-29
data organization, 26-27
data storage and retrieval, 25
data synchronization, 26
user requirements, understanding, 22-25
deterministic progress bars, 194
development life cycle, 10
development tools, 6-10
dialing phone numbers, 317
direction of vectors, 263-265
Direction property, 265
directions. *See* location-aware applications
discoverability via keywords, 361
Dispatcher property, 199
display controls in Silverlight, 49-52
display language, defined, 294
display mode, locking, 273
displaying images, 51
documentation of Application Policy Guidelines, 346
downloading data at application startup, 193-194
DownloadProgressChanged event, 132
DownloadStringAsync method, 131
DownloadStringCompleted event, 131
DragCompleted event (gestures), 121
DragDelta event (gestures), 121
DragStart event (gestures), 121
Draw method, 252, 255
drawing on maps, 106-107
drift, gyroscopes and, 93

E

E0F0E49A-3EB1-4970-B780-45DA41EC7C28.xml, 172
EAS profiles, 3
Edson, Dave, 89
ElapsedEventHandler event (Silverlight), 47

element trees, 46

element-to-element binding, 67

11th color in themes, 326

email addresses, choosing and saving, 312-314

email messages, sending, 311-312

EmailAddressChooserTask, 312

EmailComposeTask, 311-312, 341

emulation. *See also* Windows Phone Emulator

CameraCaptureTask and, 320

GPS emulation, 103

in performance tuning, 196

endpoints, 136

EndRun method, 252

enforcing business rules, 246

enumerations for game state machines, 281-287

event handlers for navigation buttons, 76

event-driven programming model, game loop model versus, 249

events

in GeoCoordinateWatcher class, 102

in page navigation model, 79-81

for pan gesture, 121

for pinch and stretch gesture, 124

in Silverlight, 34, 47-48

subscribing to, 48

tombstoning related events, 336-337

exception handling, 130, 331-334

Expression Blend, 9, 33

extending serialized classes, 242

Extensible Application Markup Language (XAML), 46

F

Facebook application, 36

FavoriteLocation class, extending, 242

field users, requirements of, 23

finding application identifiers, 315

flick gesture, 120-121

combined with pan gesture, 123

FM radio, playing, 322-323

fonts

licensing, 256

Segoe WP system font, 15-17

frame rates, 264

frameworks, list of, 8-9

full screen for games, 252

G

Gallery screen (Pictures hub), 161

game ball position, updating, 267-269

game loop, explained, 249-252

games

accelerometer-controlled movement, 272-273

background, adding, 258-259

collision detection, 269-272

combining with accelerometer, 274-277

content, adding, 253

enhancing animation, 287-288

examples of, 40-41

full screen, 252

game loop, explained, 249-252

orientation, changing, 252

physics of, 263-269

state machine for, 281-287

text, rendering, 255-258

textures, rendering, 254-255

user input, handling, 259-260

vibrations and sound effects, 289-290

XNA Framework. *See* XNA Framework

Games hub, 2, 160-162
GeoCoordinate class, 101, 103-104
GeoCoordinateWatcher class, 101-102
gestures, 116-119
- flick gesture, 120-121
- multitouch gestures, testing, 124
- pan gesture, 121-124
- pinch and stretch gesture, 124-126

“glance and go,” 2
global positioning system. *See* GPS
globalization
- localization versus, 293-294
- settings, 297-298, 300
- submitting applications in multiple languages, 367-369

globalizing application manifest file, 350
GPS (global positioning system). *See also* location-aware applications
- classes, 101-104
- emulation, 103
- position accuracy, 103

GPS Simulator, 103
gradient fills, setting, 47
graphics rendering in XNA Framework, 38
Grid control (Silverlight), 53, 55
Gyroscope class, 93-94
GyroscopeReading structure, 94

H

haptic feedback in games, 289-290
hardware requirements, 3-5
hardware sensors
- accelerometer, 87-90
- compass, 91-92
- gyroscope, 93-94
- motion detection, 95-96

HasNetworkAccess method, 130
height in container controls, 55
History screen (Music + Video hub), 162
- activating from, 166
- integration with, 165

hit area, 113-115
Hogdal, Greg, 89
Home button, 4
home page, defining, 74
HttpWebRequest class, 134-135
“hub-and-spoke” metaphor for localization, 300
hubs, 159
- Games hub, 161-162
- list of, 2, 160
- Marketplace hub. *See* Marketplace hub
- Music + Videos hub, 162-163
  - integration with, 164-170
- Office hub, 163-164
- People hub, 160-161
- Pictures hub, 161
  - integration with, 170-173

HubType value in Application Manifest file, 166
hybrid model for data from web services, 244
HyperlinkButton control (Silverlight), 50

I

icon color on Application Bar, 329
icons
- Application Bar icons, 14
- application icon, modifying, 13

identity verification for Marketplace, 349
IDisposable interface, 236
Image control (Silverlight), 51
images
- binding with themes, 330
- displaying, 51

in Silverlight, 35
in XNA Framework, 38-39
indeterminate progress bars, 194-195
INETA (International Association of .NET User Groups) OData feed, 151
INotifyPropertyChanged interface (Silverlight), 64
input controls in Silverlight, 49-52
InputScope property (Silverlight), 51-52
installing large applications, 345
InstantAnswer, 179
instrumentation, 365
instrumenting your code, 196
with DateTime class, 196
with Stopwatch class, 197
integration
with Bing API, 177-184
AppID, obtaining, 177-178
protocols for, 178-182
source types, 179
with Music + Videos hub, 164-170
with Pictures hub, 170-173
with Translator API, 185-188
interface. *See* user interface
International Association of .NET User Groups (INETA) OData feed, 151
internationalization
cultures, 295-297
globalization settings, 297-298, 300
localization
of application resources, 300-306
globalization versus, 293-294
.NET Framework model, 300
invariant cultures, 296
iPhone applications, porting user interface from, 32
isolated storage, 238-243
IsolatedStorageFileStream class, 243
IsolatedStoragePersistance class, 241, 244
isolation of WebBrowser control (Silverlight), 61
IsTrial() method, 356

J–K

joining Marketplace, 348-349
JSON (JavaScript Object Notation), 142
with Bing API, 179-181
keyboard requirements, 4
keywords for applications, 361

L

landscape view, 15
languages, submitting applications in multiple languages, 367-369
large applications, installing, 345
launchers, 309-311
for Marketplace hub, 163
for People hub, 160
launching
web browsers, 319-320
web searches, 319
Launching event, 336
Lees, Andy, 1
licensing for fonts, 256
“life in motion,” 2
light sensors, 5
limited functionality in Trial mode applications, 362
ListBox binding, 68
with data templates, 69
listings
Accelerometer class, 88
activating from Now Playing or History screen (Music + Videos hub), 166
adding navigation buttons, 75
Application Manifest with HubType value set, 166
Atom syndication format sample, 148

Author class, 64
AuthorsCollection class, 68
Bing Map control, 60
calling services asynchronously, 200
CameraCaptureTask, 321
CancelAsync method, 132
CanGoBack property, 77
Canvas control, 52
CheckBox and RadioButton controls, 51
Compass class, 91
data contract serializable class, creating, 240
DataContext, creating custom, 235-238
declarative binding example, 66
default unhandled exception code, 332
Dispatcher property, 199
EmailAddressChooserTask, 312
EmailComposeTask, 311
exception handling, 130
FavoriteLocation class, extending, 242
Game1 class, 250
Game1 class updates, 254
GameState enumeration, 281
GeoCoordinateWatcher class, 102
GeoCoordinateWatcher events, 102
gradient fills, setting, 47
Grid control and column definitions, 54
Grid control and row definitions, 53
Grid control for complex layouts, 54
Gryoscope class, 93
HasNetworkAccess method, 130
History screen integration, 165
INETA OData service, browsing, 151
InputScope property, setting, 52
instrumenting code with DateTime class, 196
instrumenting code with Stopwatch class, 197
isolated storage usage, 238
IsolatedStoragePersistance class, 241, 244
IsTrial() partial code, 356
JSON search results, 181
ListBox binding with data template, 69
Live Tile notification XML data structure, 224
MainPage.xaml updated for location awareness, 100
MapLayer class, adding, 106
MarketplaceDetailTask, 314
MarketplaceHubTask, 314
MarketplaceReviewTask, 315
MarketplaceSearchTask, 315
MediaElement control, 61
MediaPlayerLauncher, 316
Motion class, 95
navigation button event handlers, 76
network availability, determining, 232
New screen integration, 165
Now Playing screen integration, 164
OData entry, 152
opening notification channels, 210-211
Owner model (database models), 233
page navigation events, 79
Panorama control, 58
partial WMAppManifest.xml file, 350
Pet model (database models), 234
PetDataContext, 235
PhoneApplicationService defined in App.xaml, 336
PhoneCallTask, 317
PhoneNumberChooserTask, 317
Pivot control, 57
playing FM radio, 322
programmatic binding example, 67

programmatically binding images using themes, 330
pushpins, creating on maps, 107
receiving notifications, 214
RSS sample, 147
SaveEmailTask, 313
SavePhoneNumberTask, 318
searching in background threads, 201
SearchTask, 319
sending Live Tile notifications, 224-225
sending notifications, 212-213
sending Toast notifications, 219-220
serializing and deserializing lists of data, 242
ShellToastNotificationReceived event handler, 222
simple ListBox binding, 68
simple XAML, 47
SMSComposeTask, 312
Sphere class, 265
StackPanel control, 56
subscribing to events, 48
TextBlock controls with theme resources applied, 327
TitleFont.spritefont XML file, 256
Toast notification XML data structure, 219
trial-to-paid conversion, 365
WebBrowserTask, 319
WebClient class complete example, 133
XML/REST search results, 182
Yahoo! PlaceFinder response as JSON, 143
Yahoo! PlaceFinder response as XML, 142

lists of data, serializing and deserializing, 242

Live Tile notifications, 208
characteristics of, 217-218
sending and receiving, 223-227
in Windows Phone 7.1, 219

Live Tiles, 2

LoadContent method, 252, 254

loading content assets early, 252

local data management
isolated storage, 238-243
SQL CE relational database, 232-238

localization
of application manifest file, 350
of application resources, 300-306
globalization versus, 293-294
.NET Framework model, 300

localization targets tables, preparing, 301

localized strings, text length, 302

location frameworks, 9

location services, 5, 9

location-aware applications. *See also* GPS
with Bing Map control, 104-107
building, 99-101
GPS services classes, 101-104

locking display mode, 273

logo, creating, 360

M

magnitude of vectors, 263-265

Magnitude property, 265

MainPage.xaml, 74

MainPage.xaml.cs, 74

manifest file, configuring applications for Marketplace, 349-352

MapLayer class, 106-108

maps. *See* Bing Map control; location-aware applications

***Marble Maze*, 41**

marketing applications, 359-361
by entering contests, 369-370

Marketplace hub, 2, 160, 163, 343
- application guidelines in, 345-346
- categories, 347-348
- configuring applications for, 349-352
- joining, 348-349
- registration costs, 3
- submitting applications, 352-355
  - in multiple languages, 367-369
- Trial mode applications, 356, 361-362
  - Shortcuts application case study, 362-367
- versions of, 343-345

Marketplace interaction tasks, 314-316

MarketplaceDetailTask, 314, 341, 367

MarketplaceHubTask, 314

MarketplaceReviewTask, 315, 341

MarketplaceSearchTask, 315

markup in Silverlight, 46-47

Marquee screen (Music + Video hub), 162

measuring application performance, 195-198

media content, playing, 316-317

media frameworks, 8

MediaElement control (Silverlight), 60-61

MediaHistory class, 164

MediaHistoryItem class, 164

MediaPlayerLauncher task, 316-317

members. *See* class members

MemoryStream class, 243

messages, sending, 311-312

Metro design language, 11-12
- screen orientations, 15
- text guidelines, 15-18
- user-interface framework, 12
  - Start screen, 12
  - Status and Application Bars, 13-14

Microsoft Advertising pubCenter, 362

Microsoft Bing API. *See* Bing API

Microsoft Office, 2, 160, 163-164

Microsoft Silverlight Toolkit for Windows Phone, gesture listening service, 117

Microsoft XNA Game Studio. *See* XNA Framework

Microsoft.Devices.Sensors assembly, 88

Microsoft.Phone.Controls.Toolkit assembly, 117

Microsoft.Phone.UserData. Contacts class, 160

models, creating database models, 232

modifying application title/icon, 13

Motion class, 95-96

motion detection, 95-96

MotionReading structure, 96

MP3 files as sound effects, 289

multicast protocol, 139

multiple EAS profiles, 3

multitasking
- asynchronous tasks, 199-200
- background tasks, 200-201
- UI thread, 198

multitouch experience, testing, 124, 259

music, playing, 316-317, 322-323

Music + Videos hub, 2, 160, 162-163
- integration with, 164-170

N

navigation
- page navigation model, 73-74
  - backward navigation, 77-78
  - events, 79-81
  - with multiple navigation buttons, 75-76
  - overriding Back button navigation, 74

page navigation life cycle, 81-82
page URI syntax, 76
passing data within, 82-83
recursive navigation, 79
styles in Silverlight, 34
targets for HyperlinkButton control (Silverlight), 50
nested objects, serializing, 241
.NET Framework, localization model, 300
network communication
with cloud services, 129
HttpWebRequest class, 134-135
raw web communication, 129-130
sockets, 139
WebClient class, 131-133
Windows Communication Foundation, 135-138
data storage and retrieval and, 25
determining connectivity, 231-232
NetworkInterfaceType class, 232
neutral cultures, 295
New screen (Music + Video hub), 162, 165
notification channels
limits on, 209
opening, 210-212
notification services, 9
Push Notification Services, 207-208
Live Tile notifications. *See* Live Tile notifications
opening notification channels, 210-212
operational overview, 208-209
sending and receiving notifications, 212-215
Toast notifications. *See* Toast notifications
troubleshooting, 227
Now Playing screen (Music + Videos hub)
activating from, 166
integration with, 164

O

object graphs, 241
OData Data Service Utility, 153
OData libraries, consuming, 151-156
Office hub, 2, 160, 163-164
OnBackKeyPress method (Silverlight), 74
OneTime binding, 63
OneWay binding, 63
OneWayToSource binding, 63
OnExiting method, 252
OnNavigatedFrom event, 337
OnNavigatedTo event, 337
Open Data Protocol libraries, 151-156
opening notification channels, 210-212
organizing data, 26-27
orientation, 15
automatic screen rotation, 90
of games, changing, 252
origin (for coordinate system), 266
overriding Back button navigation, 74

P

Page Load for data binding, 67
page navigation life cycle, 81-82
page navigation model, 73-74
backward navigation, 77-78
events, 79-81
with multiple navigation buttons, 75-76
overriding Back button navigation, 74
page navigation life cycle, 81-82
page URI syntax, 76
passing data within, 82-83
recursive navigation, 79
page state, 337

pan gesture, 121-124

Panorama control (Silverlight), 57-59, 159

passing data within page navigation model, 82-83

payment time frames for sales, 355

People hub, 2, 160-161

performance

counters, 201-202

isolated storage and, 243

tuning. *See* tuning

persistent state data, 337

phone numbers

choosing and saving, 317-319

dialing, 317

PhoneApplicationService, 335

PhoneCallTask, 317

PhoneNumberChooserTask, 317

PhotoChooserTask, 321-322

photos. *See also* Pictures hub

choosing, 321-322

taking, 320-321

physics of games, 263-269

Pictures hub, 2, 160-161. *See also* photos

integration with, 170-173

pinch and stretch gesture, 124-126

PinchCompleted event (gestures), 124

PinchDelta event (gestures), 124

PinchStart event (gestures), 124

Pivot control (Silverlight), 57-59

platform requirements, 5-6

playing

media content, 316-317

music, 322-323

***Popper 2* game, 36-37**

portal services, 10

porting

desktop applications to Windows Phone, 21

building user interface, 27-29

data organization, 26-27

data storage and retrieval, 25

data synchronization, 26

application requirements, understanding, 21-22

user requirements, understanding, 22-25

user interface from iPhone or Android applications, 32

portrait view, 15

position accuracy (GPS), 103

position of game ball, updating, 267-269

Position property, 265

power management, black color scheme and, 326

Power/Sleep button, 4

preparing localization targets tables, 301

preserving application state, 334-339

primary keys, 245

processor requirements, 5

programmatic binding example, 67

progress bars, 194-195

properties in GeoCoordinate class, 103-104

protocols for Bing API, 178-182

proximity sensors, 5

pubCenter API, 362

publication, 10

punctuation guidelines, 17

Push Notification Services, 207-208

Live Tile notifications

characteristics of, 217-218

sending and receiving, 223-227

in Windows Phone 7.1, 219

opening notification channels, 210-212

operational overview, 208-209

sending and receiving notifications, 212-215

Toast notifications

characteristics of, 217-218

sending and receiving, 219-223

in Windows Phone 7.1, 219

troubleshooting, 227

pushpins, creating on maps, 107

Q–R

query string, 82

radians, converting to degrees, 264

radio, playing, 322-323

RadioButton control (Silverlight), 50-51

Raw notifications, 208
- sending and receiving, 212-215

Really Simple Syndication (RSS), 147-151

receiving notifications, 212-215
- Live Tile notifications, 223-227
- Toast notifications, 219-223

Recent screen (People hub), 160

recursive navigation, 79

region format, defined, 294

registration, 10
- costs for Marketplace hub, 3
- as Marketplace application developer, 348-349

Reiss, Bill, 36

Render Thread Frames Per Second performance counter, 202

rendering
- background, 259
- text, 255-258
- textures, 254-255

Requests screen (Games hub), 161

requirements
- application requirements, understanding when porting desktop applications to Windows Phone, 21-22
- of field users, 23
- hardware, 3-5
- software, 5-6
- user requirements, understanding when porting desktop applications to Windows Phone, 22-25

researching other applications, 359

resolution of screen, limitations, 111-113

resources, localization of, 300-306

REST services, consuming, 141-146

retrieving
- current culture, 297
- data, 25

reversed axes, 273

rotation, automatic screen rotation, 90

route speeds in GPS Simulator, 104

RSS feeds with Bing API, 180

RSS (Really Simple Syndication), 147-151

runtime environments in XNA Framework, 38

S

sales
- from About page, 367
- payment time frames, 355

SaveEmailTask, 313

SavePhoneNumberTask, 318

saving
- email addresses, 312-314
- phone numbers, 317-319

screen
- automatic rotation of, 90
- gestures, 116-119
  - flick gesture, 120-121
  - pan gesture, 121-124
  - pinch and stretch gesture, 124-126
- limitations, 111-113
- touch targets, 113-115

Screen Fill Rate performance counter, 202

screen orientation. ***See*** **orientation**

screen requirements, 5

scrolling in StackPanel control (Silverlight), 56-57

ScrollViewer control (Silverlight), 56-57

Search button, 4

searching. ***See also*** **Bing API**
- for Marketplace applications, 315
- with web searches, 319

SearchTask, 319

security, isolated storage and, 239

Segoe WP system font, 15, 17

selecting. *See* choosing

sending

- Live Tile notifications, 223-227
- messages, 311-312
- notifications, 212-215
- Toast notifications, 219-223

sensors

- accelerometer, 87-90
- compass, 91-92
- frameworks for, 8
- gyroscope, 93-94
- motion detection, 95-96
- requirements, 4-5

serializable classes, creating, 240-241

serializing lists of data, 242

services

- calling asynchronously, 200
- data from, 243-244

shading in XNA Framework, 39

Share Picker, integration with Pictures hub, 172-173

ShellToastNotificationReceived event handler, 222

Shortcuts application case study (Trial mode applications), 362-367

Silverlight, 32-35, 45

- application examples, 35
    - Facebook, 36
    - *Popper 2*, 36-37
- Bing Map control in location-aware applications, 104-107
- common language runtime version, 46
- controls, 49
    - container controls, 52-57
    - input and display controls, 49-52
    - Panorama and Pivot controls, 57-59
    - special controls, 59-61
- data binding, 62-64
    - to collections, 67-69
    - creating bindings, 65-67
- events, 47-48
- markup and code-behind, 46-47
- page navigation model, 73-74
    - backward navigation, 77-78
    - events, 79-81
    - with multiple navigation buttons, 75-76
    - overriding Back button navigation, 74
    - page navigation life cycle, 81-82
    - page URI syntax, 76
    - passing data within, 82-83
    - recursive navigation, 79
- performance tuning, 201-202
- Silverlight Toolkit controls, 61-62
- themes, Windows Phone themes versus, 328
- versions of, 33
- when to use, 8, 41-42

Silverlight Toolkit controls, 61-62

sizing in container controls, 55

sleeping at application startup, 192

Slider control (Silverlight), 51

SMS text messages, sending, 311-312

SMSComposeTask, 312

SOAP, with Bing API, 180

social services, 9

sockets, 139

software requirements, 5-6

sound effects in games, 289-290

source types for Bing API, 179

speaking text with Translator API, 185-188

special controls in Silverlight, 59-61

speed

- converting to mph/kph, 104
- in GPS Simulator, 104
- in vectors, 263-265

Sphere class, 265

splash screens, 191-193

Spotlight screen (Games hub), 161

SpriteBatch class, 254

SpriteBatch.Begin method, 254

SpriteBatch.Draw method, 254-255, 259

SpriteBatch.DrawString method, 256

SpriteBatch.End method, 254

SpriteFont class, 256

sprites in XNA Framework, 38

SQL CE relational database, 232-238

StackPanel control (Silverlight), 55-56

star sizing, 55

Start screen, 12

startup, 191-194

state
- preserving, 334-339
- types of, 337

state machines for games, 281-287

Status Bar, 13-14

Stopwatch class, instrumenting code with, 197

storing data, 25. *See also* databases

strings, text length of localized strings, 302

styles in Silverlight, 34

submitting applications
- to Marketplace, 352-355
- in multiple languages, 367-369

subscribing to events, 48

synchronization, 26, 245

syndication feeds, consuming, 147-151

SyndicationFeed class, 148

SyndicationItem class, 149

System.Device assembly, 101

System.Device.Location namespace, 101

System.ServiceModel.Syndication assembly, 148

T

TaskEventArgs class, 310

tasks, launchers and choosers, 309-311

taxes, responsibility for, 355

TCP (Transmission Control Protocol), 139

telephone numbers
- choosing and saving, 317-319
- dialing, 317

templates, ListBox binding with, 69

testing, 10
- applications for Music + Videos hub, 166-167
- background colors, 331
- multitouch experience, 124, 259

texels, 254

text
- rendering, 255-258
- translation with Translator API, 185-188

text guidelines for design, 15-18

text length for localized strings, 302

text messages, sending, 311-312

text resources, localizing, 302-303

TextBlock controls, theme resources applied to, 327

TextBox control (Silverlight), 51-52

Texture2D class, 254

textures, rendering, 254-255

theme resources, 327-329

themes, 325-331

threads
- background tasks, 200-201
- UI thread, 198

tiles, 2

time stamps for data synchronization, 245

time trials, 367

title of application, modifying, 13

Toast notifications, 208
- characteristics of, 217-218
- sending and receiving, 219-223
- in Windows Phone 7.1, 219

tombstoning, 334-339
- About page actions and, 341

tone (of text), 16

touch targets, 113-115

transient state data, 337

Translator API, integration with, 185-188

Transmission Control Protocol (TCP), 139

Trial mode applications, 356, 361-362
- Shortcuts application case study, 362-367

trial-to-paid conversions, capturing, 365-367

troubleshooting
- handling unhandled exceptions, 331-334
- MarketplaceDetailTask, 314
- Push Notifications Service, 227
- taking pictures with Zune connection, 320

tuning
- application performance, 195-198
- application startup, 191-194
- importance of, 191
- progress bar usage, 194-195
- Silverlight performance, 201-202

TwoWay binding, 63

typography
- in Metro design language, 11
- text guidelines, 15-18

U

UDP (User Datagram Protocol), 139

UI Design and Interaction Guide for Windows Phone 7, 11

UI Element (Silverlight), 63

UI thread
- asynchronous and background tasks, 198
- HttpWebRequest class and, 134

unhandled exceptions, handling, 331-334

unicast protocols, 139

uniform resource identifier (URI)
- format for Bing API, 178
- syntax, 76

UnloadContent method, 252

Update method, 252

UpdateSourceTrigger property (Silverlight), 64

updating game ball position, 267-269

URI (uniform resource identifier)
- format for Bing API, 178
- syntax, 76

usability, 11

User Datagram Protocol (UDP), 139

user input for games, handling, 259-260

user interface. *See also* Metro design language
- building when porting desktop applications to Windows Phone, 27-29
- gestures, 116-119
  - flick gesture, 120-121
  - pan gesture, 121-124
  - pinch and stretch gesture, 124-126
- hubs, 159
  - Games hub, 161-162
  - list of, 160
  - Marketplace hub, 163
  - Music + Videos hub, 162-170
  - Office hub, 163-164
  - People hub, 160-161
  - Pictures hub, 161, 170-173
- porting from iPhone or Android applications, 32
- screen limitations, 111-113
- themes, 325-331
- touch targets, 113-115
- in XNA Framework, 38

User Interface (UI) Thread Frames Per Second performance counter, 202

User Interface Design and Interaction Guide, 114
user requirements, understanding when porting desktop applications to Windows Phone, 22-25
user-interface framework, 12
    Start screen, 12
    Status and Application Bars, 13-14

V

validation, 10, 246
vector-based images in Silverlight, 35
vectors, magnitude and direction, 263-265
verifying
    capabilities, 351
    identity for Marketplace, 349
versions of Silverlight, 33
vibrations in games, 289-290
videos, playing, 316-317. *See also* Music + Video hub
viewing Available Resources lists, 328-329
visual designers in Silverlight, 33
Visual Studio 2010, 7
Visual Studio Toolbox, 49
voice (of text), 16
Volume button, 4

W

WCF (Windows Communication Foundation), 135-138
    data contracts, creating, 240-241
web browsers, launching, 319-320
web communication with cloud services, 129-130
    HttpWebRequest class, 134-135
    WebClient class, 131-133
web searches, launching, 319
web services, 141
    data from, 243-244
    OData libraries, consuming, 151-156
    REST services, consuming, 141-146
    syndication feeds, consuming, 147-151
web-based Marketplace, 344
WebBrowser control (Silverlight), 61
WebBrowserTask, 319, 341
WebClient class, 131-133
What's New screen (People hub), 160
What's New screen (Pictures hub), 161
width in container controls, 55
Wilcox, Jeff, 195
Windows Azure, 9
Windows Communication Foundation (WCF), 135-138
    data contracts, creating, 240-241
"Windows Phone 7 Push Notification Recipe," 215
Windows Phone 7.1, Live Tiles and Toast notifications in, 219
Windows Phone Application Platform, 6-10
Windows Phone Connect Tool, 167
Windows Phone Design Templates, 12
Windows Phone Emulator, 7
    CameraCaptureTask and, 320
WMAppManifest.xaml file, 74
WMAppManifest.xml file, 350
WPF themes, Windows Phone themes versus, 328
writing application descriptions, 361

X

XAML (Extensible Application Markup Language), 46
Xbox LIVE games, 2, 160-162
Xbox LIVE screen (Games hub), 161
XML, with Bing API, 181-182

XNA Framework, 37-40
coordinate origin, 266
accelerometer updates, 272
background, adding, 258-259
game content, adding, 253
game examples, 40-41
game loop, explained, 249-252
text, rendering, 255-258
textures, rendering, 254-255
user input, handling, 259-260
when to use, 8, 41-42
XNA Starter Kits, 38

Y–Z

Yahoo! PlaceFinder response data
as JSON, 143
as XML, 142

zoom controls in Bing Map control, 105
Zune desktop software. *See also* Music + Video hub
CameraCaptureTask and, 320
Marketplace in, 344
Zune screen (Music + Video hub), 162

How can we make this index more useful? Email us at indexes@samspublishing.com