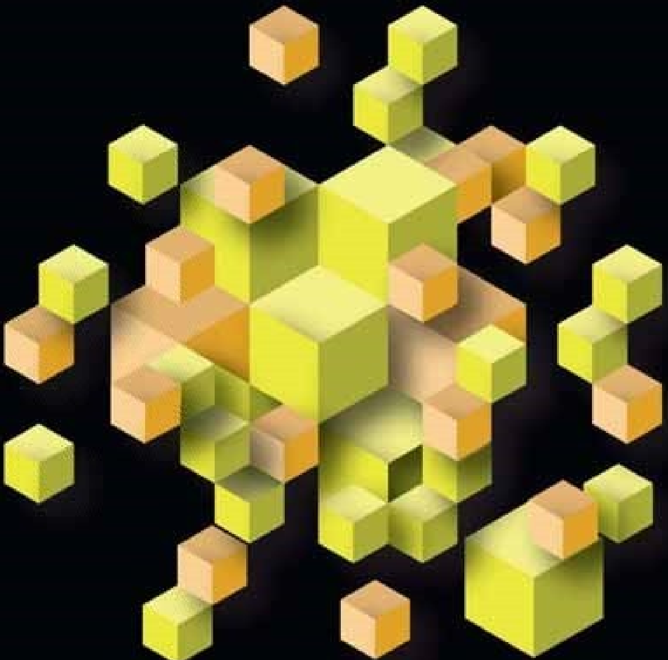
Discover the world of 2D and 3D game programming in C#
with XNA and Silverlight for Windows Phone 7 devices



Windows Phone 7
Game Development

Adam Dawes

Apress®

Windows Phone 7 Game Development

■ ■ ■

Adam Dawes

Apress®

Windows Phone 7 Game Development

ISBN-13 (pbk): 978-1-4302-3306-0

ISBN-13 (electronic): 978-1-4302-3307-7

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

President and Publisher: Paul Manning
Lead Editors: Mark Beckner, Ewan Buckingham
Technical Reviewer: Don Sorcinelli
Editorial Board: Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jonathan Hassell, Michelle Lowman, Matthew Moodie, Duncan Parkes, Jeffrey Pepper, Frank Pohlmann, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Coordinating Editor: Mary Tobin
Copy Editor: Nancy Sixsmith
Compositor: MacPS, LLC
Indexer: Brenda Miller
Artist: April Milne
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media, LLC., 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 `www.springeronline.com`.

For information on translations, please e-mail `rights@apress.com`, or visit `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 `www.apress.com/info/bulksales`.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at `www.apress.com`.

For Ritu and Kieran.

Contents at a Glance

Contents

About the Author

■ **Adam Dawes** is a software developer and systems architect working at a cutting-edge online service development company.

He has been a compulsive programmer since the age of four, when he was first introduced to a monochrome Commodore PET. The love affair has continued through three subsequent decades, flourishing through the days of the 8-bit dinosaurs to today's era of multicore processors and pocket supercomputers.

A constant throughout Adam's career has been his fondness for computer games. From the very first time *Nightmare Park* displayed its devious maze of pathways in green symbols back in 1980, he has been a games player across a variety of genres and styles. These days, he spends his spare time playing the latest 3D titles on his PC, or enjoying some of the classics in his stand-up arcade machine or sit-in cockpit driving cabinet. Creating his own games has always been a hobby and, while he has no intention of becoming part of the professional games industry, he has a lot of fun developing his own titles.

Adam lives with his wife Ritu and son Kieran in southeast England. His web site is at `www.adamdawes.com` (all his finished projects can be downloaded there) and he can be emailed at `adam@adamdawes.com`. He would particularly like to see the results of your own game development projects.

About the Technical Reviewer

■ **Don Sorcinelli** has been involved with planning, developing, and deploying enterprise applications for over 15 years. His involvement in these processes expanded to include the PDA platforms starting in the late 1990s. He is currently a Product Engineer focused on Mobile Device Management solutions.

Don frequently presents on Windows Mobile topics for users, developers, and IT professionals. As a result, he was awarded Most Valuable Professional status for Windows Mobile Devices by Microsoft Corporation in January 2004 for his work with the Windows Mobile community.

Currently, Don is co-manager of the Boston/New England Windows Mobile User and Developer Group, and webmaster of BostonPocketPC.com (`http://www.bostonpocketpc.com`). He can be contacted at `donsorcinelli@bostonpocketpc.com`.

Acknowledgments

I must start by thanking my parents for all the opportunities they gave me when I was growing up and for encouraging my computer habit from a very young age.

Thank you to everyone at Apress for their assistance in getting this book written and delivered; in particular to Mark Beckner for allowing me the opportunity in the first place, to Mary Tobin for her tireless assistance and encouragement, and to Nancy Sixsmith for making the book much more readable and for putting up with correcting the same grammatical errors over and over again.

I owe thanks, too, to Don Sorcinelli for his invaluable input throughout the whole book, and to Mike Ormond for arranging for me to get my hands on a real live Windows Phone 7 device during the writing of the book.

And finally, of course, thanks without end to my wife Ritu and my son Kieran, for their constant encouragement and for tolerating my shutting myself in my study and writing every evening and weekend. I'll be spending much more time with you both now, I promise!

Introduction

This Goal of This Book

Gaming on the move has become very popular during recent years. With the arrival of the Nintendo Gameboy, people realized that they could take their games out and about with them, and as technology has become more sophisticated these games have grown, too. They now encompass complex game mechanics, advanced 2D and 3D graphics, and engrossing stories and game worlds that the player can literally become lost in.

Alongside this phenomenon is the explosion in popularity of mobile communication devices. Nearly everyone carries a phone with them every time they leave the house. These devices have become much more than just phones, however; they provide contact management, e-mail, web browsing, satellite navigation, and entertainment.

Writing games for mobile devices allows both these trends to be brought together into the same place. It is very easy for people to "pick up and play" a game on their mobile device as they always have it in their pocket—whether they are progressing through a sprawling role-playing game on a train or simply want the few minutes of casual diversion that mobile gaming can provide while waiting for an appointment.

Windows Phone 7 Game Development aims to bring you the knowledge and techniques that you will need to create your own games for devices running the Microsoft's powerful Windows Phone 7 operating system. Starting with the basics of the platform and its development environment, and progressing through to advanced topics such as 3D graphics, it will guide you step by step toward creating a simple and manageable environment into which you can write your own mobile games and distribute them to the world for fun or profit. Example projects are provided to demonstrate all the techniques discussed and are ideal as a basis for experimentation.

Both of the application environments supported by Windows Phone 7 are addressed, exploring how games can be produced in the dedicated gaming environment, XNA, and also in the more general-purpose and user-interface-driven Silverlight.

Who This Book Is For

This book is written for those who are already familiar with programming one of the two main managed Visual Studio languages: C# or Visual Basic.NET. It is assumed that you already have a grasp of the fundamentals of programming and are familiar with using the environment for PC-based application development. This is not an introduction to programming or to Visual Studio itself.

You will, however, be given a complete guide to setting up the development environment for Windows Phone 7 programming, getting your first programs to compile, and interactively debugging your games as they run either on the Windows Phone 7 emulator included with the phone's free software development kit or on a real device.

In order to develop software for your device, you will need to use the Visual Studio 2010 development environment. If you already have Visual Studio 2010, you can integrate the Windows Phone 7 development tools into your existing environment; if you do not have it, you can obtain Visual

Studio 2010 Express for Windows Phone free of charge via a simple download from the Microsoft web site.

Although most of the projects in the book can be developed using the provided emulator, it is strongly recommended that you also have access to a real device to test your games.

The examples in this book are all written using C#, the only development language fully supported for Windows Phone 7 development. Developers who are more familiar with VB.NET should find that the language code and concepts translate over to C# fairly easily, so this should not present too much of a barrier to entry.

Chapter Overview

The following is a brief description of each chapter. The chapters tend to build on one another, so it is recommended that you read them in sequence to avoid knowledge gaps in later chapters.

Chapter 1 introduces Windows Phone 7 and using the Visual Studio 2010 development environment to create Windows Phone 7 games and applications. It explains how to set up simple .NET projects running against the emulator and real devices, explores debugging techniques, and begins to look at the two application environments: XNA and Silverlight.

Chapter 2 dives into XNA, exploring in detail the structure of XNA projects, the approach to displaying and updating graphics, how *sprites* can be used to create complex 2D graphics output, and how to work with fonts and text.

Chapter 3 takes the concepts explored so far and builds them into a simple reusable game framework that simplifies many of the tedious elements of setting up a game project. This allows you to focus on the game itself rather than getting weighed down with object management. This chapter also introduces the first of the example game projects in this book: *Cosmic Rocks*.

Chapter 4 covers the subject of user input. All sorts of input devices are available on Windows Phone 7 devices, from touch screens and keyboards through to accelerometers, and they are explored in detail to show how they can be used to allow your games to be controlled.

Chapter 5 turns up the volume and reveals the options for game audio. Covering simple sound effects to MP3 music playback, everything you need to know about sound for your games can be found here.

Chapter 6 begins to explore rendering with vertices and matrices instead of using sprites. Matrix transformations are uncovered and explained so that graphics can be rotated, scaled, and translated; and concepts such as texture mapping, blending, and alpha effects in this environment are explored.

Chapter 7 lifts the XNA feature set up into the third dimension, explaining how to create 3D game worlds. Subjects covered include perspective and orthographic projections, the depth buffer, and lighting so that your scenes really come to life.

Chapter 8 continues the exploration of XNA in the third dimension and introduces a number of useful new rendering features. These features include importing 3D objects from third-party modeling packages, moving and manipulating the camera within a game world, using particle effects, creating background imagery with sky boxes, applying fog to a 3D scene, and using XNA's Effect objects to add new features and capabilities to your game.

Chapter 9 provides some useful reusable components that can be used in any game. A simple mechanism for loading and saving user settings and a high-score table implementation are provided to allow you to focus on writing your game rather than having to reinvent these features yourself.

Chapter 10 exposes the Windows Phone 7 application life cycle and *tombstoning*, an essential topic that you will need to get to grips with so that your game can live side by side with other applications that the user chooses to open on their device.

Chapter 11 moves away from XNA and begins to explore Windows Phone 7's other application environment: Silverlight. While not specifically geared around games, Silverlight still has plenty of capabilities and great potential for game development. This chapter introduces the environment and explores how it is used.

Chapter 12 takes a more detailed look at the controls that are available for use in Silverlight projects, and also explores topics such as page navigation, orientation, and full-screen mode.

Chapter 13 focuses on game development in Silverlight, building a simple but flexible sprite control, and covering topics such as hardware acceleration, storyboard animation, game timing, and user input. It also begins development of this book's second example game project: *Diamond Lines*.

Chapter 14 steps through a series of additional Silverlight topics, exploring subjects including navigation through the different stages of a game, music and sound effects, game settings, high-score tables, and Silverlight's view of the application life cycle.

Chapter 15 sets up shop inside the Windows Phone Marketplace. This is the outlet that you need to use to distribute your game to the rest of the world, and possibly make some money from it, too. The chapter contains a guide to the Marketplace submission requirements, as well as tips on testing your game, managing versions, creating trial versions, and promoting your game to encourage people to try it.

Chapter 16 brings things to a close by stepping back from the phone and looking at how both XNA and Silverlight games can be brought to life on other platforms. XNA games can be played on Windows PCs (as well as the Xbox 360), and Silverlight games can be run in a variety of web browsers. This chapter shows you how to painlessly convert your games so that they run in these environments.

PART 1

■■■

The Beginning

CHAPTER 1

Windows Phone and .NET

It is a genuine pleasure to develop software for Windows Phone 7 devices using Visual Studio .NET.

Microsoft's latest version of its mobile operating system provides a very different environment from the versions that came before it. Virtually everything about Windows Phone is new to the mobile platform: it includes an entirely redesigned user interface, implements the powerful XNA audio/visual libraries for high-performance gaming, and has standard high-specification hardware requirements that all devices are obliged to meet.

There is one key element of Windows Phone that has stayed essentially the same as the platforms that preceded it: the use of the .NET programming environment to create games and applications. This brings with it some exceedingly powerful and flexible programming languages and one of the best development environments available anywhere.

The development platform for Microsoft's mobile devices has advanced substantially over the last decade. During the early years of the original Windows Mobile/Pocket PC operating system, programming involved using the suite of eMbedded Visual tools. They came supporting two different languages: eMbedded Visual Basic and eMbedded Visual C++.

eMbedded Visual Basic was based on the same technologies as Visual Basic for Applications (VBA). It was similar in a number of ways to Visual Basic 6 (VB6), the desktop version of VB that was current at the time, but had many shortcomings, such as the lack of strongly typed variables and poor object orientation features. Programs were written using a stand-alone integrated development environment (IDE), which had its own peculiarities and different ways of working than VB6.

eMbedded Visual C++ presented more of a challenge because of differences not only in the IDE but also in the code. Although established C++ programmers would no doubt have managed to pick up this language without too many problems, those who were less well versed in the intricacies of C++ might have found the amount of new information they needed to learn a significant barrier to entry.

All this changed with the release of Visual Studio .NET and the .NET Compact Framework (.NET CF). .NET CF provides a set of class libraries that are parallel to the desktop .NET Framework. The libraries are not identical because parts of the full .NET Framework functionality are missing from .NET CF. However, a substantial set of identical functionality does exist, and any programmer who is comfortable developing C# or VB .NET applications for Windows will be instantly at home developing against .NET CF, too.

Windows Phone development uses the very latest Visual Studio 2010. The IDE has made advances in a number of ways since that of the earlier versions of Visual Studio, but best of all, Microsoft has chosen to release an "Express" version of Visual Studio that supports Windows Phone development completely free of charge. Although there are charges and fees involved in some areas of development and in distribution of finished applications (as we will see later in this book when we discuss this subject in more detail), these are generally fairly modest and do not create the barriers to entry that having to purchase the full versions of Visual Studio presented in the past.

The Windows Phone development environment also integrates into the full versions of Visual Studio seamlessly if you have such a version already installed.

On Windows Phone devices, all applications are written using managed .NET code. The Silverlight and XNA runtime libraries are preinstalled on the device, so no awkward installation is required of your end users. Finished games and applications are distributed through a central Marketplace operated by Microsoft, which has support for useful features such as update notifications, trial versions, and protection against piracy.

A major advantage of developing for Windows Phone using Visual Studio .NET is that the exact same IDE is used as for desktop Windows development. There is no need to learn the details or keyboard shortcuts of a new IDE; instead, you will be working within the environment you are already used to, which includes all your user interface tweaks and preferences changes. Developing an application for Windows Phone is simply a question of creating a different project type.

Programming within Visual Studio .NET also means that the Windows Phone developer can take advantage of the maturity of the Visual Studio.NET development environment. Microsoft has spent many years improving the user interfaces and functionality of Visual Studio, and countless versions and releases have cumulated in an extremely powerful and user-friendly studio for application design, development, and debugging. All this is at your disposal when developing Windows Phone games and applications.

The Framework also retains much of the power of its desktop cousin, including extensive object orientation features, strong variable typing, generics, flexible collections, and powerful XML processing functions.

Finally, the use of .NET, Silverlight, and XNA means that there are great opportunities for cross-platform development with only a small amount of additional effort. Silverlight games can be modified to run in the browser on a desktop PC, and XNA games can be made to run on a PC or Xbox 360 in addition to the phone.

In this chapter, we will take a closer look at the .NET Framework, at setting up and using Visual Studio, and at creating your first simple Windows Phone application. We will also examine some of the options that are available for game development.

Looking Closely at Visual Studio Development for Windows Phone

Let's start by taking a look at the versions of Visual Studio that we can use for developing software for Windows Phone.

We can develop either using Visual Studio 2010 Express for Windows Phone, which is available free of charge, or by using one of the full versions of Visual Studio. Both products will produce the same resulting games and applications. Visual Studio 2010 has system requirements that necessitate a reasonably modern PC, running either Windows Vista with Service Pack 2 or Windows 7. Both x86 and x64 versions of the operating system can be used, but the Starter Editions of Windows are not supported. A DirectX 10 or later graphics card with a WDDM 1.1 driver is required for developing games using XNA. In practice, this requirement will be met by most graphics cards from the last couple of years, although many mobile graphics chips present in laptop computers might lack these capabilities.

There is no support for developing for Windows Phone in earlier versions of Visual Studio. The good news is that Visual Studio 2010 will install side by side with earlier versions of Visual Studio without causing any problems, so if you need to keep Visual Studio 2005 or 2008 on your PC, you can.

Language Choices

.NET development offers the choice of a number of different languages including C#, VB, and C++. Currently, however, only C# is supported for Windows Phone development. Microsoft is promising support for VB in the near future, so check the latest version of the development environment if you want to use VB for your development.

If you are already familiar with VB, you should find that both reading and writing C# will come naturally with a little practice. In this book we will focus on C# for all of our samples and example code.

IDE Features

As would be expected from Visual Studio, a number of very useful features are available to help develop and debug Windows Phone applications.

Emulators

Visual Studio offers a Windows Phone emulator to help test and debug your programs. Although it is essential to use a real device regularly during your application development process to ensure that everything works properly on actual hardware, being able to use the emulator for general coding and testing is extremely useful. A screenshot of the Windows Phone emulator is shown in Figure 1–1.

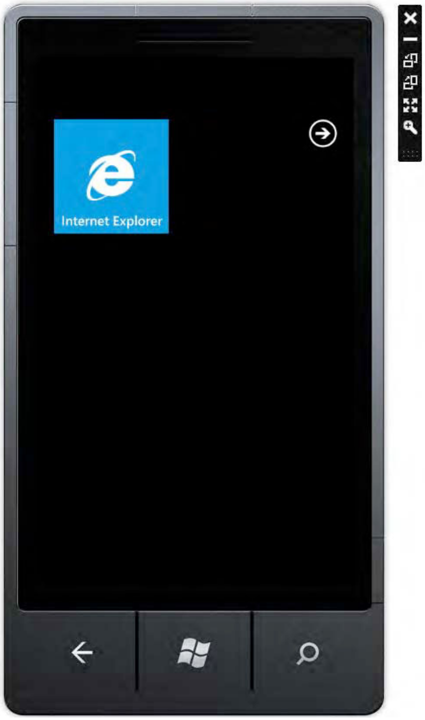


Figure 1–1. *Windows Phone emulator*

The emulator offers a full implementation of the physical device and is capable of running genuine Windows Phone applications. It offers access to a number of features of the device, including the ability to simulate networking, screen rotation, and touch screen input using the mouse cursor. (In fact, if you are developing on a Windows 7 computer with a touch-enabled screen, you can perform touch and multitouch operations on the emulator by touching your PC screen.)

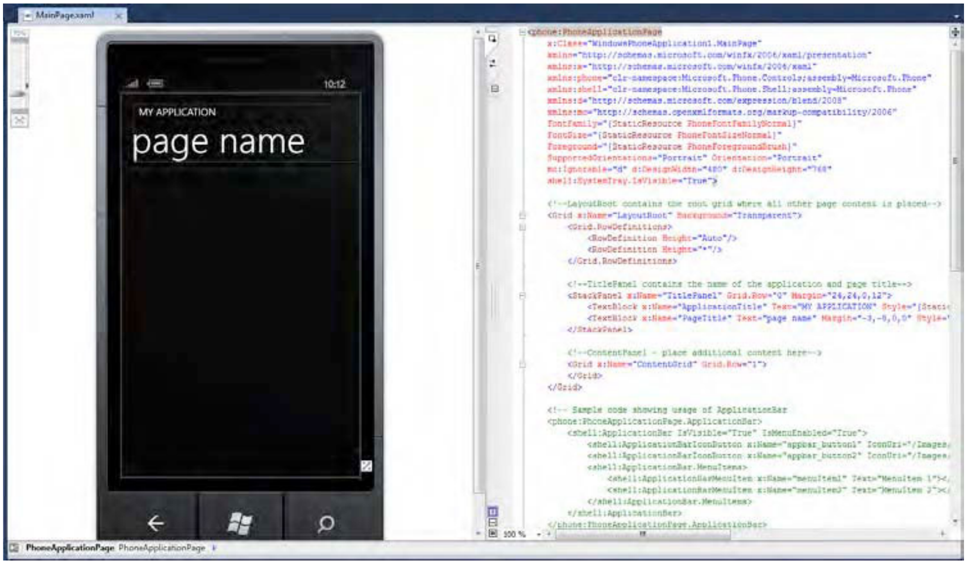
Running your application in an emulator is as simple as can be: just select to use the emulator within the Visual Studio IDE and start your application. The emulator will appear, and your program will run.

When we fire up the emulator shortly, note that it takes a little while to initialize itself. This would get frustrating very quickly when you are in a repeat modify/compile/test cycle, but the emulator can be left running in the background when you stop your code from executing and will then resume much more quickly the next time you begin a debug session. When the emulator's Close button is clicked, it will prompt for confirmation that the emulator is really no longer required, which is very useful protection against accidentally closing the window without actually meaning to.

Silverlight Page Designer

A fully featured page designer is available to lay out windows and controls for use within Silverlight programs. The designer goes as far as to display an image of the device around the edge of your page to help visualize its appearance.

Silverlight pages can be created and modified by either using the designer or by editing the underlying eXtensible Application Markup Language (XAML, which is generally pronounced "zammal"). We will look at XAML and the Silverlight page designer in a great deal more detail later in this book. Visual Studio will display both the designer and the XAML editor as side-by-side panels within the IDE, as shown in Figure 1–2, and any change made to either will be immediately reflected in the corresponding panel. This provides a very flexible mechanism for page design, allowing each panel to work together to perform their actions more efficiently.



```
<phone:PhoneApplicationPage
    x:Class="WindowsPhoneApplication1.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot contains the root grid where all other page content is placed-->
    <Grid x:Name="LayoutRoot" Background="Transparent">
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!--TitlePanel contains the name of the application and page title-->
        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="24,24,0,12">
            <TextBlock x:Name="ApplicationTitle" Text="MY APPLICATION" Style="{Stati
            <TextBlock x:Name="PageTitle" Text="page name" Margin="-3,-8,0,0" Style="
        </StackPanel>

        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentGrid" Grid.Row="1">
        </Grid>
    </Grid>

    <!-- Sample code showing usage of ApplicationBar
    <phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="True">
            <shell:ApplicationBarIconButton x:Name="appbar_button1" IconUri="/Images/
            <shell:ApplicationBarIconButton x:Name="appbar_button2" IconUri="/Images/
            <shell:ApplicationBar.MenuItems>
                <shell:ApplicationBarMenuItem x:Name="menuItem1" Text="MenuItem 1"></
                <shell:ApplicationBarMenuItem x:Name="menuItem2" Text="MenuItem 2"></
            </shell:ApplicationBar.MenuItems>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>
```

Figure 1–2. *The Silverlight page editor showing the designer and page source together*