This book is dedicated to my mother, Linda, who passed after a long fight with cancer on January 6, 2008. Your courageous battle was and will always be an encouragement to me and all who knew you. I love you and miss you. I also would like to dedicate this book to my dad, Ken, who lost the gift most precious to him. Your strength has been an inspiration to me. I love you, Dad.
## Contents at a Glance

**About the Author** ............................................................... xiii
**Acknowledgments** ........................................................... xv
**Introduction** ....................................................................... xvii

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Welcome to Silverlight 2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Introduction to Visual Studio 2008</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>Layout Management in Silverlight</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>Silverlight Form Controls</td>
<td>57</td>
</tr>
<tr>
<td>5</td>
<td>Data Binding and Silverlight List Controls</td>
<td>85</td>
</tr>
<tr>
<td>6</td>
<td>Data Access and Networking</td>
<td>117</td>
</tr>
<tr>
<td>7</td>
<td>Local Storage in Silverlight</td>
<td>135</td>
</tr>
<tr>
<td>8</td>
<td>Introduction to Expression Blend</td>
<td>167</td>
</tr>
<tr>
<td>9</td>
<td>Styling in Silverlight</td>
<td>189</td>
</tr>
<tr>
<td>10</td>
<td>Transformations and Animation</td>
<td>221</td>
</tr>
<tr>
<td>11</td>
<td>Custom Controls</td>
<td>245</td>
</tr>
</tbody>
</table>

**INDEX** ........................................................................... 269
Contents

About the Author ................................................................. xiii
Acknowledgments ................................................................. xv
Introduction ............................................................................ xvii

CHAPTER 1 Welcome to Silverlight 2 ................................. 1

The Evolution of the User Interface ....................................... 1
Rich Internet Application Solutions ...................................... 3
What Is Silverlight? .............................................................. 3
Benefits of Silverlight .......................................................... 5
  Cross-Platform/Cross-Browser Support .......................... 6
  Cross-Platform Version of the .NET Framework ............ 6
  XAML, a Text-Based Markup Language ....................... 7
  Use of Familiar Technologies ......................................... 7
  Small Runtime and Simple Deployment ....................... 8
The Silverlight Development Environment ......................... 9
Summary .............................................................................. 11

CHAPTER 2 Introduction to Visual Studio 2008 ..................... 13

Just What Is Visual Studio? .................................................... 13
What’s New in Visual Studio 2008? ..................................... 14
  JavaScript IntelliSense and Debugging ....................... 14
  Multi-Targeting Support ............................................. 26
  Transparent IntelliSense Mode ................................... 28
Building Your First Silverlight Application in Visual Studio . 29
  Try It Out: Hello World in Silverlight 2 ..................... 29
Hosting Your Silverlight Application: Web Site or
  Web Application? ....................................................... 33
Summary .............................................................................. 34
CHAPTER 3  Layout Management in Silverlight .......................................................... 35
  Layout Management ................................................................................................. 35
  The Canvas Panel ...................................................................................................... 36
    Try It Out: Using the Canvas Panel ......................................................................... 37
    Filling the Entire Browser Window with Your Application ...................................... 41
  The StackPanel Control ............................................................................................ 42
    Try It Out: Using the StackPanel Control .............................................................. 42
    Try It Out: Nesting StackPanel Controls ............................................................... 45
  The Grid Control ........................................................................................................ 47
    Try It Out: Using the Grid Control ......................................................................... 48
    Try It Out: Nesting a Grid and Spanning a Column .............................................. 52
  Summary .................................................................................................................... 55

CHAPTER 4  Silverlight Form Controls ................................................................. 57
  Setting Control Properties ....................................................................................... 57
    Attribute Syntax ...................................................................................................... 57
    Element Syntax ....................................................................................................... 58
    Type-Converter–Enabled Attributes ...................................................................... 58
    Attached Properties ............................................................................................... 59
  Nesting Controls Within Controls ........................................................................... 59
  Handling Events in Silverlight .................................................................................. 61
    Try It Out: Declaring an Event in XAML .............................................................. 61
    Try It Out: Declaring an Event Handler in Managed Code .................................. 65
  The Border Control ................................................................................................... 69
  User Input Controls ................................................................................................... 73
    Try It Out: Working with the TextBox Control .................................................... 73
    Try It Out: Working with the RadioButton and 
    CheckBox Controls ............................................................................................... 77
  Extended Controls .................................................................................................... 81
    Adding an Extended Control .................................................................................. 81
    Try It Out: Using the GridSplitter ........................................................................ 82
  Summary .................................................................................................................... 84

CHAPTER 5  Data Binding and Silverlight List Controls ....................................... 85
  Data Binding ............................................................................................................. 85
    The Binding Class ................................................................................................... 86
    Try It Out: Simple Data Binding in Silverlight .................................................... 86
CONTENTS

CHAPTER 6 Data Access and Networking .................................................. 117

Data Access in Silverlight Applications ............................................. 117
Accessing Data Through Web Services .............................................. 118
  Try It Out: Accessing Data Through a WCF Service ....................... 118
  Using a Standard WCF Service with Silverlight ......................... 130
Accessing Services from Other Domains ......................................... 130
Accessing Data Through Sockets ..................................................... 131
Summary ......................................................................................... 133

CHAPTER 7 Local Storage in Silverlight ............................................... 135

Working with Isolated Storage .......................................................... 135
  Using the Isolated Storage API ..................................................... 136
  Try It Out: Creating a File Explorer for Isolated Storage ............. 139
Managing Isolated Storage ............................................................... 162
  Viewing and Clearing Isolated Storage ....................................... 162
  Try It Out: Increasing the Isolated Storage Quota .................... 163
Summary ......................................................................................... 166

CHAPTER 8 Introduction to Expression Blend .................................. 167

Key Features in Expression Blend 2 .................................................... 168
  Visual XAML Editor .................................................................................................. 168
  Visual Studio 2008 Integration ............................................................................. 168
  Split-View Mode ................................................................................................. 169
  Visual State Manager and Template Editing Support .................. 170
  World-Class Timeline ....................................................................................... 170
  Try It Out: Working with Projects in Expression Blend 2 .......... 171
Exploring the Workspace .............................................. 175
   Toolbox ......................................................... 175
   Project Panel .................................................. 178
   Properties Panel ............................................... 178
   Objects and Timeline Panel ..................................... 180
Laying Out an Application with Expression Blend ..................... 180
   Working with the Grid Control in Expression Blend ............. 180
   Try It Out: Editing a Layout Grid with Expression Blend ....... 180
Summary ........................................................................ 188

CHAPTER 9 Styling in Silverlight ........................................ 189
   Inline Properties ................................................... 189
      Try It Out: Setting Inline Properties with Visual Studio ..... 190
      Try It Out: Setting Inline Properties with Expression Blend . 197
   Silverlight Styles .................................................. 206
      Try It Out: Using Styles As Static Resources ................... 208
      Defining Styles at the Application Level ....................... 215
      Silverlight Style Hierarchy ..................................... 217
Summary ........................................................................ 219

CHAPTER 10 Transformations and Animation .......................... 221
   Introduction to Silverlight 2 Animation .......................... 221
      Silverlight Storyboards .......................................... 222
      Types of Animation in Silverlight .............................. 223
   Programmatically Controlling Animations ........................ 225
   Using Expression Blend to Create Animations ................. 228
      Viewing a Storyboard in the Expression Blend Timeline .. 228
      Try It Out: Creating an Animation with Expression Blend .. 229
   Creating Transformations in Silverlight ........................ 236
      Transformation Types ........................................... 236
      Try It Out: Using Expression Blend to Transform Silverlight
      Objects .............................................................. 239
Summary ........................................................................ 243
CHAPTER 11  Custom Controls ................................................................. 245

   When to Write Custom Controls .................................................. 245
   Silverlight Control Toolkit .......................................................... 246
   Silverlight Control Model ............................................................ 248
      Parts and States Model ............................................................. 248
      Dependency Properties .......................................................... 249
   Creating Custom Controls in Silverlight 2 ................................. 250
      Implementing Custom Functionality ......................................... 251
      Try It Out: Building a Custom Control .................................... 251
   Summary ..................................................................................... 268

INDEX .............................................................................................. 269
ROBERT LAIR has been working with .NET technologies since before its alpha release, and built the original IBuySpy Store and Portal applications that were used by Microsoft to introduce ASP.NET to the development community. He is the author of Pure ASP.NET (Sams, 2002), a reference for web development in the .NET Framework, and portions of ASP.NET for Developers, as well as numerous magazine articles on the topic of .NET. Robert has also been a speaker at a number of .NET technical conferences. Technologies in which Robert specializes include Silverlight, CRM–Live service integration, mainframe modernization to .NET, ASP.NET custom application development, and SharePoint development and integration.

Acknowledgments

There are a number of people to whom I would like to express my appreciation—people who have helped me in many ways. This book proved to be a much greater challenge than I had anticipated, as throughout the course of writing, a number of life events crept in, including the death of my mom. Without these people, this book would never have been possible.

First and foremost, I would like to thank my loving and inspiring wife, Debi. While I spent hours in my office writing, she picked up the slack around the house, and through it all provided me with love, support, and a lighthouse that I could always turn to and refocus on. I cannot thank her enough for the sacrifices she has endured, and for always being there. I would also like to thank my son, Max, who gave up a great deal of time with his dad. I hope this book is proof that you can accomplish anything if you put your mind to it and have the perseverance to see it through. I look forward to once again giving you the time you both deserve and to being a family again.

I would like to thank the many people at Apress who made this book happen. I would especially like to thank Tracy Brown Collins, Ewan Buckingham, Marilyn Smith, Elizabeth Berry, Dominic Shakeshaft, and Stefan Turalski. Without all of your hard work, this book would never have happened. Thank you all.

Through life’s many challenges, everyone needs someone to whom they can turn. I would like to thank a great friend of mine, Paul Strozier, for all of his support and encouragement, and especially for his friendship. True friends are priceless. Thank you, Paul, for everything.

And finally, I would like to shout out from Gooblicious to Warclock, Essy, and the members of Slighted and Distant Beliefs, who provided me with friendship and an escape when I needed one.
Welcome to *Beginning Silverlight 2: From Novice to Professional*. This book will provide you with an introduction to Silverlight: what it is, what it means to you as a developer, and how to begin developing Silverlight-enabled applications. You'll not only read about the features of the Silverlight development environment, but also work through many hands-on examples that demonstrate exactly how to use those features to create rich Internet applications (RIAs).

Who Should Read This Book

This book is written for application developers who want to get started with Silverlight 2. It assumes that you have some experience developing applications using technologies related to Microsoft’s ASP.NET, and have worked with Microsoft Visual Studio. You should be familiar with the JavaScript, C#, and XML languages.

How This Book Is Organized

Each chapter focuses on a particular area of Silverlight and contains one or more “Try It Out” exercises that allow you to apply what you have learned. Here is a summary of what each chapter includes:

- Chapter 1, “Welcome to Silverlight 2,” gives you an introduction to RIAs and Silverlight. You will also learn about the tools used in developing Silverlight-enabled applications.
- Chapter 3, “Layout Management in Silverlight,” discusses Silverlight’s flexible layout management system, which lets you specify how controls will appear in your applications. It describes Silverlight 2’s layout management controls in depth.
- Chapter 4, “Silverlight Form Controls,” introduces the common form controls that are provided with Silverlight 2. You will continue to work with these controls throughout the book.
- Chapter 5, “Data Binding and Silverlight List Controls,” looks at the Silverlight 2 controls that display lists of data and how to bind data to these controls. You’ll see that these controls are flexible and can show data in unique ways.
Chapter 6, “Data Access and Networking,” describes how data access in Silverlight applications works compared with data access in traditional applications. It then explores mechanisms for accessing data in Silverlight applications, focusing on the use of web services.

Chapter 7, “Local Storage in Silverlight,” covers localized storage in Silverlight 2, which is handled by its isolated storage feature. You’ll learn how to store user-specific data for your application and have that data persist over browser instances.

Chapter 8, “Introduction to Expression Blend,” gets you started with Microsoft Expression Blend, which lets you edit XAML documents visually.

Chapter 9, “Styling in Silverlight,” describes how you can control the styles of your Silverlight application’s user interface elements. You’ll learn about defining style properties inline using both Visual Studio and Expression Blend, as well as how to use Silverlight styles.

Chapter 10, “Transformations and Animation,” covers creating animations in Silverlight 2. You’ll see how Expression Blend 2 helps you create complex animations and transformations.

Chapter 11, “Custom Controls,” explains the basics of creating custom controls in Silverlight 2. First, it covers when it might be appropriate to write custom controls in Silverlight 2, and then it describes how to build a custom control that has several different states.

By the time you finish this book, you will have a firm foundation in Silverlight 2, and will be able to create your own Silverlight-enabled applications.
This chapter introduces Silverlight, a Microsoft cross-browser, cross-platform plug-in that allows you to create rich interactive (or Internet) applications (RIAs) for the Web. It begins with a brief look at the evolution of user interfaces, and then provides an overview of Silverlight. You’ll learn how Silverlight fits into RIA solutions, the benefits it brings to developers, and the tools involved in developing Silverlight-enabled applications.

The Evolution of the User Interface

Software user interfaces are constantly evolving and improving. I remember back when I was still working with an early version of Windows and looking at Mac OS with envy. Then I remember seeing Linux systems with radical new desktop interfaces. More recently, I found myself looking again at the Mac OS X Dock (see Figure 1-1) and wanting that for my Windows XP machine—to the point where I purchased a product that mimicked it. I was dedicated to Windows through it all, but I was envious of some of the user experiences the different environments offered.

Figure 1-1. The Mac OS Dock feature

The evolution of the user interface continues in the Windows Vista operating system. One example is the interface for switching between applications. In past versions of Windows, when you pressed Alt+Tab to switch from one program to another, you would see a rather ugly interface offering nothing but icons. Today, when you press Alt+Tab in Vista, you get a much more user-friendly interface, presenting a clipping of the content of each window as you tab through your choices, as shown in Figure 1-2.
In addition, Vista offers an even cooler way to switch between applications using the Desktop Window Manager. When you press the Windows key along with Tab, Vista displays all open windows in a cascading shuffle effect, which allows you to see a large-scale version of each window (see Figure 1-3). And if there is animated content in any of the windows, it actually shows up in the view! So, if you have a video or a game playing in one of the windows, you will see that in action as you shuffle through the windows.

These features reflect how developers have built standard desktop applications, which are meant to be installed and executed on individual client machines. Desktop applications allow for very rich and responsive user interfaces and additional features, such as offline support. Performance of the application depends on the machine on which it is installed. A challenge for desktop applications is deployment. The application needs to have a code base for each target platform, and every machine needs to have the application installed and maintained.
In contrast, we have web applications, which are HTML-focused programs designed to run within a browser and across platforms. For the Microsoft-based developer, this has recently meant developing with ASP.NET and building web services to offer services over the Internet. The focus of most of the logic and code has been placed on the server for the benefit of application performance. The price has been a poor user interface.

With recent technologies, the line between the desktop and web approaches for developing applications has started to blur. As a result, a third approach has surfaced. This new approach is termed RIA, which is defined as a web application that has the features and functionality found in traditional desktop applications.

**Rich Internet Application Solutions**

The concept of RIA has been around for quite some time, but the term *rich Internet application* was first used in 2002 in a Macromedia white paper. Before then, the terms *remote scripting* and *X Internet* were used to describe the concept.

Today, many different solutions fit the description of RIAs, but there is one consistent characteristic: all RIA solutions involve a runtime that runs on the client machine and architecturally sits between the user and the server.

In recent years, the technology that is most commonly used in RIAs is Flash. When Flash was introduced, it brought to the Web rich user experiences never seen before. However, due to the lack of tools allowing Microsoft .NET developers to integrate Flash into their applications, to those developers, Flash just seemed like a tool for adding some pretty effects to a web page, but nothing functional.

Then a wonderful thing happened when Adobe purchased Macromedia. All of the sudden, Flash was married to some of the development tools offered by Adobe. Microsoft retaliated by announcing Silverlight, formerly known as Windows Presentation Foundation Everywhere (WPF/E). Silverlight is the technology that many .NET developers have been waiting for.

But what exactly is Silverlight? And what impact does Silverlight actually have on us as .NET developers? Well, I’m glad you asked.

**What Is Silverlight?**

As I stated in the previous section, all RIAs have one characteristic in common: a client runtime that sits between the user and the server. In the case of Microsoft’s RIA solution, Silverlight is this client runtime. Specifically, Silverlight is a cross-platform, cross-browser plug-in that renders user interfaces and graphical assets on a canvas that can be inserted into an HTML page.
The markup used to define a Silverlight canvas is called Extensible Application Markup Language (XAML, pronounced “zammel”). XAML is an XML-based language that is similar to HTML in some ways. Like HTML, XAML defines which elements appear, as well as the layout of those elements. However, unlike HTML, XAML goes far beyond simple element definition and layout. Using XAML, you can also specify timelines, transformations, animations, and events.

The following is an example of a Silverlight canvas defined in XAML:

```xml
<Canvas
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   Width="640" Height="480"
   Background="White"
   x:Name="Page">
   <Rectangle
      RenderTransformOrigin="0.5,0.5"
      x:Name="rectangle"
      Width="292"
      Height="86"
      Fill="#FFFF0000"
      Stroke="#FF000000"
      StrokeThickness="3"
      Canvas.Left="115"
      Canvas.Top="70"/>
</Canvas>
```

Figure 1-4 shows this canvas in Microsoft Expression Blend, the design tool used to edit and create XAML for Silverlight applications. You can see that this XAML simply defines a rectangle on a canvas, as well as the properties associated with that rectangle, including its name, location, size, color, and border.

This simple example is just intended to give you an idea of what XAML looks like. You’ll learn more about XAML in upcoming chapters. For now, let’s continue by looking at the benefits of Silverlight.
Benefits of Silverlight

Naturally, Silverlight offers all of the same benefits of RIAs, but there are a few features that set it apart from other RIA solutions, including the following:

- It offers cross-platform/cross-browser support.
- It provides a cross-platform version of the .NET Framework.
- XAML is a text-based markup language.
- Silverlight uses familiar technologies.
- It’s easy to deploy the Silverlight runtime to clients.

Let’s take a closer look at each of these benefits.
Cross-Platform/Cross-Browser Support

When ASP.NET was released a number of years ago, one of the benefits touted was cross-browser support. Developers would need to have only one code base, and that code base would work in all modern browsers. For the most part, this is true. No matter which browser you are using, the application will function. However, in order to receive all of the bells and whistles offered by the ASP.NET controls, you must use the latest version of Internet Explorer. If you are using any other browser, you actually get a downgraded version of the web site, which contains fewer features.

Validation controls are a prime example. If you are using a browser that ASP.NET recognizes as an “upscale” browser, you can take advantage of client-side validation. If you are using any other browser, the validation controls still function, but require a postback to the server to do the validation. So, although ASP.NET is cross-browser, users can get different experiences, depending on which browser they are using.

With Silverlight, this changes. Microsoft is once again pulling out the term *cross-browser*, and also adding *cross-platform*, and this time they mean it. As a developer, you can create a Silverlight application and rest assured that it will run exactly the same on all supported platforms and browsers.

Currently, two platforms are supported. Naturally, the first is Windows-based platforms, and the second is Mac OS platforms. As for browser support, Internet Explorer and Firefox are currently covered. Microsoft has committed support for Safari as well, so it may be on the list by the time you’re reading this book.

This leaves one large platform unsupported: Linux. Although Microsoft does not have plans to support Linux, others do. The Mono project, which is sponsored by Novell, is an open source initiative to develop and run .NET client and server applications on Linux, Solaris, Mac OS X, Windows, and Unix. The Mono team has indicated that it will soon have a Silverlight implementation, currently called the Moonlight runtime. With this addition, developers will be able to develop Silverlight applications for Windows, Macintosh, and Linux systems with one code base. Furthermore, the user experience will be identical, no matter which platform you are using.

Cross-Platform Version of the .NET Framework

Silverlight 1.0 was released by Microsoft in the summer of 2007, but this version supported only Ecma languages that are interpreted in the client. And although Silverlight 1.0 works well for developers who are already familiar with client-side scripting, many developers have their eyes on the second release of Silverlight, version 2. Silverlight 1.0 is more or less in direct competition with Flash—some have called it Microsoft’s “Flash killer.” However, things really get exciting with Silverlight 2.
Silverlight 2 contains its own cross-platform version of the .NET Framework, which means it has its own version of the common language runtime (CLR), the full type system, and a .NET Framework programming library that you can use in Visual Studio 2008 to build rich user experiences in the browser.

**XAML, a Text-Based Markup Language**

Another advantage to Silverlight is that its foundation is based on a text-based markup language. For other RIA solutions such as Flash, the base is a compiled file. This is not nearly as friendly to developers as a text-based format, for obvious reasons.

XAML is very easy to write and modify. As an example, let’s say you want to change the opacity of an object. If you were using Flash to do this, you would need to open the Flash project file, find the right layer and object, and then make the adjustment there. You then would need to recompile and republish the file. In contrast, with Silverlight, you simply open the XAML file, change the opacity property of the object, and save the file.

Another advantage of XAML is that it can be created dynamically at runtime. If you think about it, the implications of this are huge. Consider the similarities between HTML and XAML. Both are text-based markup languages that have a decent similarity to XML. HTML is the base foundation of files published on the Internet. Since HTML was introduced, a number of technologies have been built on top of it. In the Microsoft camp, for example, Active Server Pages (ASP) was first introduced to allow developers to dynamically modify HTML at runtime. Today, we have ASP.NET. XAML has the same potential, since it is a text-based markup language on which developers can expand.

**Use of Familiar Technologies**

Microsoft is very good at creating tools that make application development easy. The Visual Studio integrated development environment (IDE) has been around for quite some time, and although new features are continually added to the tool, the environment itself has remained remarkably consistent.

Silverlight development is no different. At the core of developing Silverlight 2 applications is Visual Studio 2008, the latest version in Visual Studio’s long history. This gives Silverlight a distinct advantage, as developers do not need to learn how to use a new development environment.

In addition to Visual Studio, Microsoft has released a suite of tools called Expression Studio. Included in this suite is Microsoft Expression Blend, which is used to edit and create XAML for Silverlight applications. While Expression Blend looks completely different, it still has many of the same elements as Visual Studio. In addition, Expression Blend 2 works off of the same project as Visual Studio. This means that as you make changes in each of the editors—opening a project in Visual Studio, and then opening the same project in Expression Blend to edit the XAML—the edited files will request to be refreshed when opened again in the other tool.
Small Runtime and Simple Deployment

Since Silverlight requires that a client runtime be installed on the client machine, it is vital that this runtime has a small footprint and downloads quickly. Microsoft worked very hard to get the installation size as small as possible. The developers clearly succeeded with Silverlight 1.0, as the download size is a tiny 1MB. For Silverlight 2, however, they had a harder chore ahead of them, since Silverlight 2 contains its own .NET Framework and object library. Microsoft went to each .NET Framework team and allocated it a size to fit its portion. The end result is astonishing—Silverlight 2 is approximately 4MB in size.

As for pushing the Silverlight runtime out to clients, Microsoft has provided a very easy detection mechanism. If the client does not have the proper Silverlight runtime installed, it will display a logo, as shown in Figure 1-5.

![Silverlight runtime required logo](image)

Figure 1-5. Silverlight runtime required logo

When users click the icon in the logo, they are taken to a web page that walks them through the process of installing the Silverlight runtime. Once the runtime is finished installing, the Silverlight application is immediately available to the user, as shown in the example in Figure 1-6.
The Silverlight Development Environment

In the past, setting up an environment to work with Microsoft’s latest and greatest has been relatively straightforward, typically involving only the setup of the latest version of Visual Studio and the appropriate software development kit. However, with Silverlight, the situation is quite a bit different due to the introduction of many new tools. Let’s take a look at these new tools:

*Silverlight 2 Runtime:* This is the Silverlight client runtime, which is required on every computer that wishes to view a Silverlight-enabled web application.

*Silverlight 2 Software Development Kit (SDK):* This SDK is a collection of samples, Silverlight QuickStarts, documentation, and controls that are used to develop Silverlight applications. This SDK is not required, but it is recommended that all Silverlight developers download it.

*Visual Studio 2008:* As noted, this is the latest version of Microsoft’s IDE, the successor to Visual Studio 2005 (see Figure 1-7). Installing Visual Studio 2008 also automatically installs Microsoft .NET Framework 3.5. There are many new features in Visual Studio 2008 that make it a highly recommended upgrade in general; for serious Silverlight developers, Visual Studio 2008 is a must. Chapter 2 covers Visual Studio 2008 in more depth.
**Silverlight Tools for Visual Studio 2008:** This is an add-on for Visual Studio that provides a Silverlight project system for developing Silverlight applications using C# or Visual Basic. This add-on is required if you wish to take advantage of Visual Studio to build Silverlight applications. The project system includes the following components:

- Visual Basic and C# project templates
- IntelliSense and code generators for XAML
- Debugging of Silverlight applications
- Web reference support
- Integration with Expression Blend

**Microsoft Expression Blend 2:** This is a “what you see is what you get” (WYSIWYG) editor for XAML (see Figure 1-8). Expression Blend is similar to Adobe’s Flash MX product. It allows you to lay out the XAML canvas, add timelines, and create transformations—all in a very user-friendly and visual way. Expression Blend 2 is covered in Chapter 8.
Summary

In this chapter, we looked at the evolution of user interfaces in applications, as well as the history of RIAs. I then introduced Silverlight, and talked about the benefits it brings to developers today and how it fits into RIA solutions. Finally, you learned about the tools involved in developing Silverlight-enabled applications.

Now it is time to get your hands dirty and start building some Silverlight applications! In the next chapter, I will provide an introduction to Microsoft Visual Studio 2008, one of the primary tools used to build Silverlight applications.
The previous chapter mentioned the tools required to develop RIAs that utilize the Silverlight technology. At the core of all of these tools is Microsoft’s flagship development product, Visual Studio. This chapter provides an introduction to the latest version, Visual Studio 2008. You will learn about some of the new features that are particularly helpful for developers building RIAs with Silverlight, and then work through an exercise to try out Visual Studio 2008’s enhanced JavaScript IntelliSense and debugging support. Finally, you will have an opportunity to create your first Silverlight application using Visual Studio 2008. Let’s get started with a brief introduction to the Visual Studio IDE.

Just What Is Visual Studio?

Any developer who has developed applications using technologies related to Microsoft’s Visual Basic, ASP, or .NET has used some version of Visual Studio on a regular basis. This is because Visual Studio is Microsoft’s primary development product. Whether you are developing desktop applications, web applications, mobile applications, web services, or just about any other .NET solution, Visual Studio is the environment you will be using.

Visual Studio is an IDE that allows .NET developers to implement a variety of .NET solutions within the confines of one editor. An IDE is a software application that contains comprehensive facilities to aid developers in building applications. Visual Studio fits this description for a number of reasons. First, Visual Studio offers a very rich code-editing solution. It includes features such as source code color-coding and code completion. Second, it offers an integrated debugger, which allows you to place breakpoints in your source code to stop execution at any given point, as well as step through the source line by line, analyzing the state of objects and fields at any given point in the execution. Add to these features rich support for application deployment, installation, and integration with database services, and you can understand how Visual Studio is an extremely valuable tool for developers.
Note This book assumes a basic understanding of Visual Studio. If you’re new to Visual Studio, I recommend that you get started with a book devoted to the subject, such as Beginning C# 2008, Second Edition by Christian Gross (Apress, 2008).

THE HISTORY OF VISUAL STUDIO

Visual Studio has quite a history. The first version was called Visual Studio 97, which was most commonly known for Visual Basic 5.0. In 1998, Microsoft released Visual Studio 6.0. That version included Visual Basic 6.0, as well as Microsoft’s first web-based development tool, Visual InterDev 1.0, which was used to develop ASP applications.

Next came the introduction of Microsoft .NET and ASP.NET 1.0, prompting Visual Studio.NET. As Microsoft was enhancing and releasing new versions of Microsoft .NET and ASP.NET, it also continued enhancing Visual Studio by releasing Visual Studio 2003 and then Visual Studio 2005. In addition, Microsoft has introduced a line of free development tools known as the Visual Studio Express tools, as well as the Visual Studio Team System, which can be used by large programming teams to build enterprise-level systems.

This brings us to the latest version of Visual Studio, which Microsoft developed under the code name Orcas and has now dubbed Visual Studio 2008.

What’s New in Visual Studio 2008?

Microsoft has introduced a variety of new features in Visual Studio 2008, many of which are geared toward helping developers build RIAs with Silverlight and related Microsoft technologies, such as the Windows Communication Foundation (WCF), ADO.NET Data Services, and Ajax. Here we will look at some of the new features in Visual Studio 2008 that are particularly helpful to Silverlight application developers.

JavaScript IntelliSense and Debugging

Client-side scripting is a major component of developing RIAs. With the adoption of technologies like Ajax and Silverlight, developers can integrate client-side scripting into applications to enhance the user experience.

In response to the growing necessity for integrating client-side scripting into ASP.NET applications, Microsoft has implemented an extensive upgrade to Visual Studio’s JavaScript IntelliSense and debugging support. Here, we’ll look at the IntelliSense and debugging improvements, and then try a test run to see them in action.
IntelliSense Improvements

The first major improvement of JavaScript IntelliSense in Visual Studio 2008 is type inference. Since JavaScript is a dynamic language, a variable can be one of many different types, depending on its current state. For example, in the following code snippet, the variable \( x \) represents a different type each time it is assigned.

```javascript
function TypeInference()
{
    var x;
    x = document.getElementById("fieldName");
    // \( x \) is now an HTML element
    alert(x.tagName);
    x = 10;
    // \( x \) is now an integer
    alert(x.toFixed());
    x = new Date();
    // \( x \) is now a date
    alert(x.getDay());
}
```

In this example, the variable \( x \) represents three different types during the execution of the function:

- First, it represents an HTML element. When the user types \( x \) followed by a period, the code-completion choices will be specific to an HTML element, as shown in Figure 2-1.

![Figure 2-1. Code completion with type inference for an HTML element](image-url)
• In the next line, \( x \) is assigned to the value 10. At this point, \( x \) has become an integer, and the code-completion choices that appear are specific to an integer, as shown in Figure 2-2.

![Figure 2-2. Code completion with type inference for an integer](image)

• Finally, \( x \) is assigned to a date type. At this point, \( x \) represents a date type, and the code-completion choices include date-specific properties and methods.

The second notable enhancement to JavaScript IntelliSense in Visual Studio 2008 is the support for IntelliSense in external script files. In fact, there are many levels to this enhancement. First, developers will have IntelliSense while they are editing the external script files. Second, by adding a reference to other external script files, developers can get IntelliSense for functions and fields from other script files. Finally, developers will receive IntelliSense in the actual pages that reference the external script files.

Another new feature of JavaScript IntelliSense is the ability to add XML comments to your code, which will provide additional information in the IntelliSense display. These are similar to standard C# XML comments, which have been available in C# since it was initially released. The following example shows some XML comments added to a JavaScript function.

```javascript
function HelloWorld(FirstName, LastName)
{
    /// <summary>Returns a hello message to the given name</summary>
    /// <param name="FirstName">Person's First Name</param>
    /// <param name="LastName">Person's Last Name</param>
    /// <return>string</return>
    return ("Hello " + FirstName + " " + LastName);
}
```
This is a function called `HelloWorld`, which simply accepts a first and last name and returns a hello message customized for that person. This function is located in a file called `JScripts.js`. Notice the four XML comments added to the start of the function. These provide a summary of the function, give a description of the function’s parameters, and indicate the value returned by the function. With these extra lines in place, when you add the function in your code, IntelliSense will now display this additional information. First, when you start typing `HelloWorld`, Visual Studio’s JavaScript IntelliSense will help you complete the method call. After you have typed `HelloWorld` and the opening parenthesis, it will display the two parameters and their descriptions, as shown in Figure 2-3.

![Figure 2-3. IntelliSense for a JavaScript function with parameter tags](image)

Now that we have reviewed the JavaScript IntelliSense features added to Visual Studio 2008, let’s take a look at the new JavaScript debugging features, which are equally as useful and long-awaited.

### New Debugging Features

In previous versions of Visual Studio, ASP.NET developers were severely limited in the debugging they could do in client-side scripting. Some of the more industrious developers would find a third-party JavaScript debugging tool to assist them. However, the majority of developers would simply use hacks, such as adding alerts throughout their client-side scripting. When an alert was not hit, they could identify where the error had occurred and at least determine the basic location where attention was required.

In Visual Studio 2008, JavaScript debugging is now integrated directly into the IDE, and believe it or not, it actually works!