Chapter 1

Getting to Know Silverlight

Silverlight is Microsoft's implementation of a cross-browser, cross-platform client framework that allows designers and developers to deliver Rich Internet Applications (RIA) embedded in Web pages. Silverlight is fitted with a flexible media pipeline that makes it extremely easy to implement media-rich controls in your Web applications.

This chapter gives you a brief introduction to the Silverlight framework. The following sections discuss what Silverlight is and why you would want to use it to develop Web-based applications. They also define the architecture behind the Silverlight framework and Silverlight applications to help you understand how Silverlight fits into the Web services picture.

IN THIS CHAPTER

What Is Silverlight?

You can look at the Silverlight framework as a combination of three very different architectures: the browser plug-in, presentation framework, and .NET framework. The culmination of these frameworks allows Silverlight to bridge the gap between presentation user interface (UI) using declarative languages and functional programming using a subset of the .NET framework.

The lightweight browser plug-in provides the necessary interaction with the browser enabling the same Silverlight application to run on multiple platforms. The plug-in must be installed by the user before Silverlight applications can be viewed in the browser.

Silverlight applications are implemented as embedded objects in Web pages. When the browser encounters a Silverlight object in the Web page, the plug-in downloads an XAP package from the Web server that contains the binaries and resources for the Silverlight application and then begins code execution inside the Web page.

Silverlight applications are run as client-side applications without the need to refresh the browser to update the UI. However, because of the built-in .NET framework, Silverlight applications can easily integrate with server-side controls and services. Using Silverlight's implementation of the .NET framework, developers can easily integrate existing libraries and code into Silverlight applications.

Silverlight's presentation framework is a subset of the Window Presentation Foundation (WPF), which is based on the eXtensible Application Markup Language (XAML) programming language. The XAML language is simply based on the XML language with application elements that map to objects and properties in the .NET framework. Because it is based on the XML language format it can be easily parsed and integrated with many technologies.

The XAML language lends itself to the UI design side of Silverlight because it is simple to implement and understand. All of the UI generated by XAML is vector based, allowing it to be dynamically reshaped and resized easily while maintaining a crisp visual effect.

Silverlight's implementation of WPF provides a rich set of controls such as buttons, calendars, text boxes, scroll viewers, a data grid, and much more. These controls are easy to implement in the XAML language as well as easy to access from .NET managed code.

Silverlight's media pipeline makes it simple to stream media such as WMV, MP3, and JPEG files to your application UI. This allows you to add rich UI elements that can give users a true Web experience.

Why Use Silverlight?

The biggest reason to use Silverlight is that it seamlessly integrates the XAML declarative language with the .NET framework. XAML adoption is growing rapidly because of how easy it is to implement amazing UI interfaces. Many developers already have applications and libraries written in .NET. That code can usually be easily modified to fit Silverlight applications.

The Silverlight platform appeals to both designers and to developers because it provides a dynamic platform that makes it easy to develop powerful Web applications that incorporate rich graphics, audio, and video. There is a distinct division between the declarative XAML designers use and the .NET managed code that developers use, allowing each side to implement its piece and easily integrate the two.

Another reason Silverlight appeals to both designers and developers is that Microsoft offers powerful tools that make it easy to implement Silverlight applications. Microsoft's Expression Suite provides useful tools to implement Silverlight UI and encode media. Microsoft's Visual Studio provides a dynamic development interface with tools that speed up and increase productivity.

Some other reasons to use Silverlight are listed here:

- **It is a cross-browser, cross-platform technology, which provides a consistent user experience everywhere it runs.**
- **The Silverlight plug-in installs in seconds and leaves a very small footprint.**

- **After you install the plug-in, users no longer need to install anything on their workstations to run Silverlight applications.** The applications are available to them from whatever browser they are accessing.
- **It runs a client-side application that can read data and update the UI without interrupting the user by refreshing the whole page.**
- **It can run asynchronous communications with the server allowing the UI to continue to function while waiting for the server response.**
- **It delivers rich video, audio, and graphics.**

There are some disadvantages to implementing Silverlight applications as opposed to traditional client applications. It is important that you understand those limitations when deciding to use Silverlight.

The following is a list of Silverlight limitations that may impact your applications:

- **The user must have access to the Internet for Silverlight applications to run.** Also, if your Silverlight applications have a lot of media that is bandwidth intensive, slow Internet connections can impact the user experience.
- **Limited access to the file system.** Browser-based applications are limited in their access to the files system. However, you can ask the user to access the local file system to read files and there is an isolated local storage that you can use for small amounts of persistent data.
- **Browser settings may limit Silverlight applications.** Users have the ability to disable scripting and controls in their browsers. These settings may limit or inhibit running Silverlight applications.
- **The data that is dynamically implemented in Silverlight applications is not visible to search engines.** If you have data that needs to be visible to search engines to give your Web site exposure, you need to find some way to expose that data outside of Silverlight.

Comparing Silverlight 1.0 and 2

The difference between Silverlight 1.0 and 2 is very significant. The biggest change is the implementation of the .NET framework. If you are familiar with Silverlight 1.0 then you will be used to coding the application functionality in JavaScript. You still can implement functionality using JavaScript; however, you can now also implement functionality using C#, Visual Basic, Python, Ruby, and managed JavaScript.

Another major change is the introduction of the XAP package. In Silverlight 1.0, the XAML code was referenced directly by the Silverlight object embedded in the browser. In Silverlight 2, however, the embedded object references an XAP package that contains the XAP file, assemblies, and resources necessary to run the Silverlight application.

NOTE **Because of the number of changes from Silverlight 1.0 to Silverlight 2, there are several code-breaking changes. Visit Microsoft's Web site at `http://msdn.microsoft.com/en-us/library/cc189007(VS.95).aspx` to see a list of code breakers that may apply to your Silverlight 1.0 applications.**

To help you get a feel for the changes between Silverlight 1.0 and 2, Table 1.1 provides a list of features implemented in Silverlight 2 versus what was implemented in Silverlight 1.0 that was collected from the MSDN documentation.

TABLE 1.1

Silverlight 1.0 versus 2 Feature Comparison

Features	Silverlight 1.0	Silverlight 2
2D Vector Animation/Graphics	YES	YES
AJAX Support	YES	YES
Cross-Browser (Firefox, IE, Safari)	YES	YES
Cross-Platform (Windows, Mac)	YES	YES
Framework Languages (Visual Basic, Visual C#, IronRuby, IronPython)	NO	YES
HTML DOM Integration	YES	YES
HTTP Networking	YES	YES
Isolated Storage	NO	YES
JavaScript Support	YES	YES
JSON, REST, SOAP/WS-*, POX, and RSS Web Services (as well as support for Sockets)	NO	YES
Cross Domain Network Access	NO	YES
LINQ to Objects	NO	YES
Canvas Layout Support	YES	YES
StackPanel, Grid, and Panel Layout Support	NO	YES
Managed Control Framework	NO	YES
Full suite of Controls (`TextBox`, `RadioButton`, `Slider`, `Calendar`, `DatePicker`, `DataGrid`, `ListBox`, and others)	NO	YES
Deep Zoom Technology	NO	YES
Managed HTML Bridge	NO	YES
Managed Exception Handling	NO	YES
Media — Content Protection	NO	YES
Media — 720P High Definition (HD) Video	YES	YES
Media — Audio/Video Support (VC-1, WMV,WMA, MP3)	YES	YES

Features	Silverlight 1.0	Silverlight 2
Media — Image Support (JPEG, PNG)	YES	YES
Media Markers	YES	YES
Rich Core Framework (for example, Generics, collections)	NO	YES
Security Enforcement	NO	YES
Silverlight ASP.NET Controls (asp:media, asp:xaml)	YES	YES
Type Safety Verification	NO	YES
Windows Media Server Support	YES	YES
XAML Parser (based on WPF)	YES	YES
XMLReader/Writer	NO	YES

Silverlight Framework Architecture

One of the most appealing features of Silverlight is its ability to easily integrate into several different architectures. The Silverlight framework allows applications to access Web services, databases, Web servers, and other data sources to acquire data that is displayed in the UI. It also allows the application to integrate with the Web page DOM as well as AJAX and JavaScript to enhance the functionality of the Web page.

Figure 1.1 outlines a basic architecture of the Silverlight framework and how it fits into the full stack from the Web browser to Web servers and other services.

Notice in Figure 1.1 that the Silverlight framework sits in between the Web browser DOM and the Web services. JavaScript and AJAX flow up the stack and can be integrated and accessed from all layers of the Silverlight framework.

Figure 1.1 shows the Silverlight framework broken down into the Silverlight Plug-in, Core Presentation Framework, and NET Silverlight Framework. The following sections discuss each layer of the Silverlight architecture.

The Silverlight plug-in

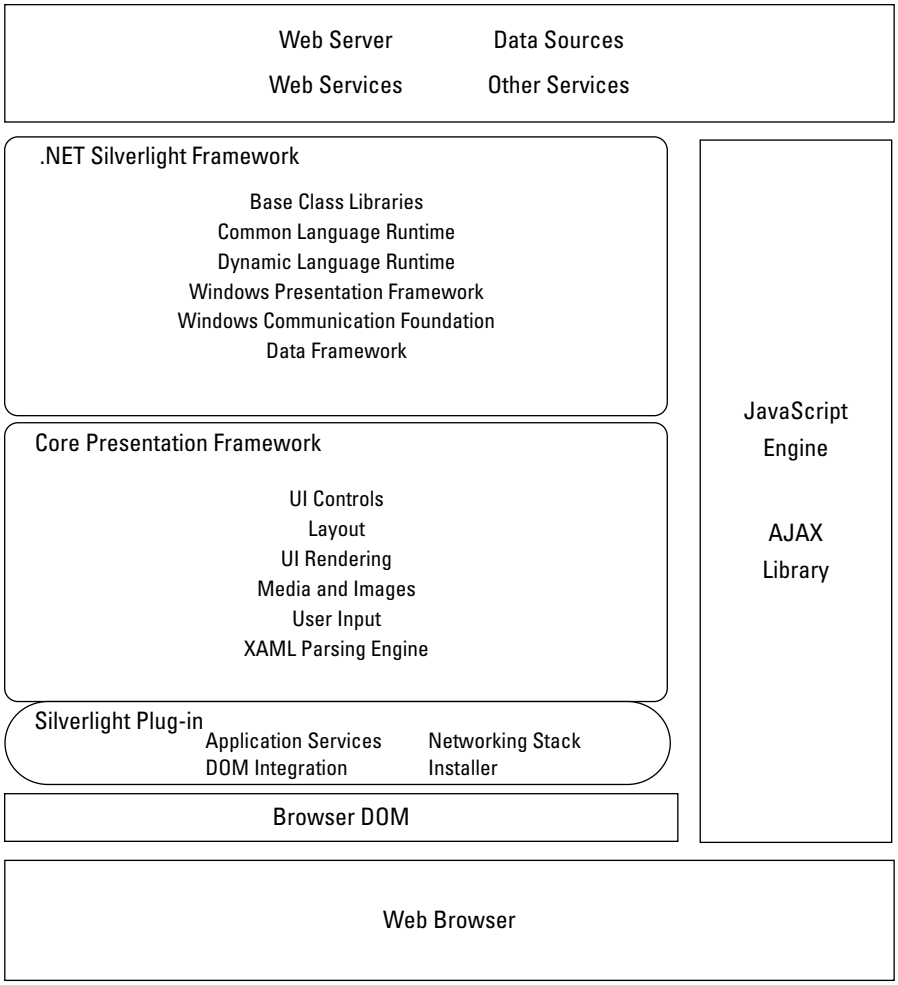
The Silverlight plug-in is a very lightweight component that is necessary for users to access Silverlight applications. The plug-in download and install take only a few moments and do not take up much hard drive space.

NOTE **When a Web page containing a Silverlight application is displayed, the user should be given a link to download the plug-in from Microsoft if the plug-in is not already installed.**

The Silverlight plug-in is responsible for accessing the Silverlight object in the Web page, downloading and accessing the XAP package, setting up the program environment, and beginning execution of the application.

FIGURE 1.1

Silverlight framework architecture diagram



The Silverlight core presentation framework

The Silverlight core presentation framework is a subset of the Windows Presentation Foundation language. The core presentation framework provides the libraries and utilities necessary to parse the Silverlight XAML files, present the UI to the browser, and handle interaction from the user.

The following is a list of components that are included with the core presentation framework listed in Figure 1.1:

- **XAML parser:** Parses the XAML files into objects.
- **UI renderer:** Handles rendering of XAML objects, such as vector/bitmap graphics, animation, and text, into UI elements that are displayed in the applications.
- **Layout:** Utilizes canvas, grid, and other controls to dynamically position and size UI elements.
- **Controls:** Implements extensible controls, such as buttons, sliders, calendars, and text boxes, which provide customizable functionality to applications.
- **Media pipeline:** Provides streaming of audio and video files as well as playback and other management.
- **Data binding:** Enables data objects to be directly linked to UI elements in a one-way or two-way relationship. Changes to values can then be reflected automatically using the link.
- **Input:** Handles the input from user input devices such as a mouse, keyboard, and other input requests.
- **DRM:** Implements digital rights management to protect media files in Web applications.

The .NET Silverlight framework

The .NET Silverlight framework is a subset of the .NET programming platform. Silverlight uses much of the .NET framework and enhances some of the libraries to provide additional functionality necessary for Silverlight applications.

The .NET Silverlight framework provides the libraries and utilities necessary to implement managed code that accesses remote services and data, interacts with code from the core presentation framework, accesses SQL and other data sources, and provides the functionality to your Silverlight applications.

The following is a list of components that are included with the .NET Silverlight framework listed in Figure 1.1:

- **Common Language Runtime (CLR):** Provides the memory management, type checking, exception handling, and garbage collection for Silverlight applications
- **Base Class Libraries:** Provides the base set of .NET libraries that handle basic programming functionality such as string handling, regular expressions, collections, and input/output
- **Dynamic Language Runtime (DLR):** Provides the framework to dynamically compile and execute JavaScript, IronPython, and IronRuby managed code in Silverlight applications
- **Windows Presentation Foundation (WPF):** Provides libraries to parse, access, and modify controls defined in XAML files as well as create and dynamically add new ones

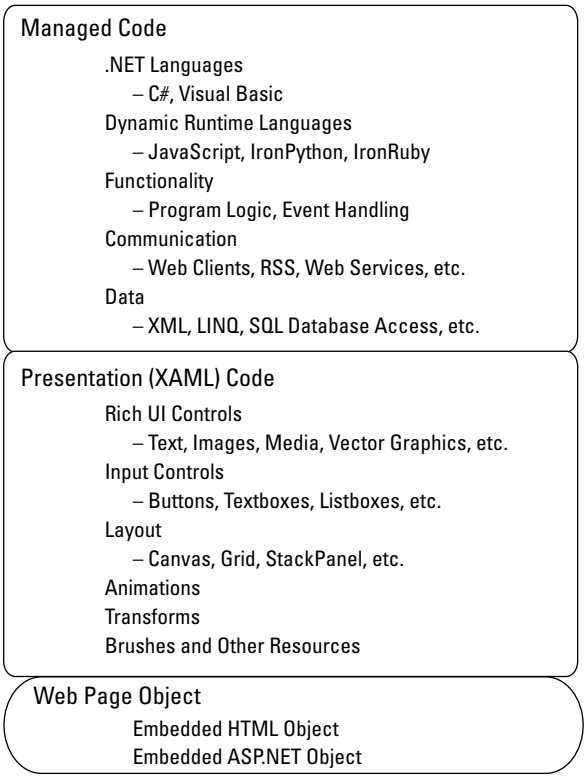
- **Windows Communication Foundation (WCF):** Provides libraries to interface with remote services, such as Web servers, RSS feeds, and other Web services
- **Data Framework:** Provides a set of libraries to parse and serialize XML data as well as support for Language Integrated Query (LINQ) requests to XML sources and SQL databases

Silverlight application architecture

Silverlight applications can be broken down into three distinct components, each of which is written in a different language. Figure 1.2 shows a diagram of these components as well as some of their uses. These components directly map to the components of the Silverlight framework architecture that are described in following chapters.

FIGURE 1.2

Silverlight application architecture diagram



The bottom layer in Figure 1.2 is the Web Page Object. This is an HTML element or ASP.NET control that tells the browser to use the Silverlight plug-in to load a Silverlight application.

The middle layer in Figure 1.2 is the Presentation Code. This is one or more XAML files that define the UI elements, user input controls, transformations, and animations that will be implemented in the Silverlight application. Most of the UI will be written in XAML because it is simple to implement especially with the aid of applications such as Microsoft's Expression Blend.

The top layer in Figure 1.2 is the Managed Code. These are .NET code-behind files that are actually part of the same class as the XAML files. This provides the managed code with the ability to access and modify the UI elements defined in the XAML file to implement functionality in Silverlight applications. These files can be written in C#, Visual Basic, JavaScript, IronPython, and IronRuby. Using Silverlight implementation of the .NET framework, managed code is able to implement data access and communication to a variety of services.

Summary

Silverlight is a cross-browser, cross-platform client framework that allows designers and developers to deliver rich Internet applications embedded in Web pages. Silverlight is fitted with a flexible media pipeline that makes it extremely easy to implement media-rich controls in your Web applications.

The Silverlight framework is made up of three main components: the browser plug-in, core presentation framework, and .NET Silverlight framework. This framework is based on a subset of other well-established frameworks such as the .NET platform and the WPF framework.

Silverlight applications are also made up of three distinct components that correspond to each aspect of the framework. Silverlight objects are embedded in Web pages. The UI elements of Silverlight applications are defined in XAML files that are parsed and rendered using the core presentation framework. The functionality of Silverlight applications is implemented using managed .NET code-behind files that share the same class with the XAML file.

In this chapter, you learned:

- What Silverlight is
- How Silverlight fits in the Web services and applications picture
- The advantages of using Silverlight
- Silverlight's limitations
- What components make up the Silverlight framework
- How the components of the Silverlight framework relate to each other
- How the components of the Silverlight framework relate to the components of Silverlight applications