

Silverlight Overview

What is Silverlight?

Silverlight (formerly known as WPF/E or Windows Presentation Foundation / Everywhere) is a cross-platform browser plug-in being developed by Microsoft that is designed to provide a richer internet experience for enterprise and consumer applications. Silverlight 1.0 has been called a “Flash killer” in that it will provide enhanced animation, vector graphics, and video playback capabilities. Although these features are interesting for improving the multimedia experience on the web, the real potential for Silverlight lies in the feature set to be released with version 1.1.

Silverlight 1.1 will contain a substantial subset of the .NET 3.5 runtime (also running cross-browser and cross-platform) and will enable a new generation of smart/rich internet applications. Silverlight 1.1 has the potential to become an ASP.NET or even an HTML application killer, especially where sophisticated, feature-rich applications are involved. Silverlight will essentially host a full .NET runtime engine within a sandbox on the client computer and allow a .NET smart client application to run on any platform, all with the same deployment characteristics as a thin-client application (once the Silverlight plug-in has been installed). Essentially, Silverlight creates a “best of both worlds” environment, in which users can experience responsive, media-rich internet applications that leverage local computing power and resources, while administrators enjoy the benefits of “no-touch” deployments and centralized control.

Silverlight Adoption Curve

Just like any browser plug-in, Silverlight will face an initial adoption curve, and the value of the plug-in as a target platform for development will be determined by the number of end-user installations. However, given the excitement over the powerful capabilities of Silverlight, the speed of a Silverlight download (20 seconds over broadband), and Microsoft’s vast distribution channel (Microsoft will likely include it in every Internet Explorer and Windows installation as well as on every new PC), Silverlight adoption will likely proceed at a brisk pace and reach high levels of market penetration quickly.

ASP.NET Migration to Silverlight

Once the Silverlight plug-in has gained market acceptance, ASP.NET application architects are likely to ask why they should prefer development in ASP.NET rather than Silverlight. On the one hand, they could write a “Web 2.0” application using AJAX, JavaScript, HTML, CSS, and a back-end programming language, with the necessity to manage session state and scalability on the server and with potential cross-browser compatibility issues. Alternatively, they could write a Silverlight application in a pure .NET environment with a WPF (WinForms-like) UI that is highly functional and has clean model-view-controller separation.

A .NET Silverlight application will be easier to write, more responsive, more scalable, more performant, and just as easy to deploy as an HTML application once the Silverlight plug-in has been installed. Cross-platform objections will be mitigated, as Silverlight will be supported across all major browsers and operating systems. Meanwhile, HTML / CSS / JavaScript will continue to have cross-platform issues given each browser's propensity to interpret code in a slightly different manner.

Furthermore, with Silverlight, offline or partially connected applications are fully supported, as the application actually executes in a sandbox on the client and can be disconnected from the internet. Since the application state is maintained on the client, the server does not need to keep track of the client state and the application becomes far more scalable, easier to develop, and simple to load balance.

Smart Client Migration to Silverlight

Those architects currently building smart client applications may wish to target Silverlight as a platform if they desire broad cross-platform penetration of their application. Since a Silverlight application is inherently smart client, the basic programming model remains the same and only differences in the Silverlight runtime from the full .NET runtime need to be addressed.

The two major differences in the Silverlight runtime are its lack of WinForms support (in favor of WPF) and lack of support for database connectivity. The decision not to include WinForms in Silverlight was made to reduce the download size, simplify cross-platform porting of the .NET runtime, and because WPF is the natural evolution that will eventually replace WinForms.

The decision to remove support for database connectivity was made since a Silverlight application will be deployed as an n-tier internet application, and it is poor practice to expose the database directly to the internet. Connecting to the database in a 2-tier or client-server model across the internet exposes both the connection string as well as the database port to the entire world and creates inherent security and scalability issues. Instead, a Silverlight application will fetch data using a web service or other equivalent n-tier data provider (such as DevForce).

DevForce and Silverlight

DevForce will dramatically simplify the development of a Silverlight application. All of the value that DevForce provides in building .NET applications today will also apply to building Silverlight applications. In addition, the DevForce persistence layer and Business Object Server will make accessing data in a Silverlight application far simpler. Instead of having to write and expose a web service for every data access, a developer will simply query for objects and DevForce will automatically fetch them from the object server, send them across the internet, and cache them on the client.

Applications currently built with DevForce are well positioned to transition to Silverlight in the future. The existing business objects, business logic, and best practices will remain the same and carry forward into the Silverlight world. The persistence layer in DevForce will have nearly the same API regardless of whether the Silverlight or full .NET runtime is being used as the host environment. This means developers can start building their applications with DevForce today, and choose later to deploy as an n-tier Silverlight application without having to rewrite their persistence code.

A single code base can also be used to support the Silverlight runtime as well as the full .NET runtime. In scenarios where cross-platform or zero-deployment characteristics are required, the Silverlight application can be used. When users are ready for more powerful client-side features such as Outlook/Office integration, high volume offline storage, or access to advanced third-party components, they can upgrade to the full featured .NET application.

When Will Silverlight be Available?

While a release date for Silverlight 1.1 has not yet been announced, the alpha version is currently available and it is speculated that there will be a final release in mid 2008. IdeaBlade has a close partnership with Microsoft with early access to all Microsoft technologies and will be supporting Silverlight development with DevForce as soon as it is released.