ASP.NET Server Controls for Silverlight in the Silverlight 3 SDK

Contents

[Introduction 2](#_Toc234295500)

[Moving ASP.NET Silverlight Control Functionality from Silverlight 2 to Silverlight 3 2](#_Toc234295501)

[Effects of Installing the Silverlight 3 SDK and Uninstalling the Silverlight 2 SDK for the ASP.NET Silverlight Controls 3](#_Toc234295502)

[Working with Existing Silverlight Solutions in the Silverlight 3 SDK 4](#_Toc234295503)

[Using the Silverlight 2 ASP.NET Controls with the Silverlight 3 SDK 4](#_Toc234295504)

[Using the Existing Silverlight Controls 5](#_Toc234295505)

[Modifying ASP.NET Silverlight Controls to Work with Silverlight 3 6](#_Toc234295506)

[Setting the Silverlight Minimum Version 6](#_Toc234295507)

[Deep Linking (Navigation) 6](#_Toc234295508)

[Hardware Acceleration 6](#_Toc234295509)

[Cache Visualization 6](#_Toc234295510)

[Obtaining the Source Code for the ASP.NET Silverlight Controls 7](#_Toc234295511)

[Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK 7](#_Toc234295512)

[Cross-Domain XAP 7](#_Toc234295513)

[CPU Utilization While the Application is Idle 7](#_Toc234295514)

[Upgrading Silverlight Projects and Solutions 8](#_Toc234295515)

[Sharing Silverlight Projects and Solutions 8](#_Toc234295516)

[Effect on Designer Tools of Removing ASP.NET Silverlight Controls 9](#_Toc234295517)

[Implementing ASP.NET Silverlight Control Functionality in Silverlight 3 10](#_Toc234295518)

[Setting the Source Location of the XAP or XAML File 10](#_Toc234295519)

[Setting the PlugInNotInstalledTemplate Property 10](#_Toc234295520)

[Using Stretch Mode 10](#_Toc234295521)

[Setting the Minimum Version 11](#_Toc234295522)

[Handling the Load Event or Other Events 11](#_Toc234295523)

[Programmatically Changing the Plug-In in Server Code 12](#_Toc234295524)

[Creating a Media Player 12](#_Toc234295525)

[FAQ 13](#_Toc234295526)

[How can I use the ASP.NET MediaPlayer Control with a new skin file? 13](#_Toc234295527)

[How can I work with media using Silverlight 3? 13](#_Toc234295528)

[Can I continue to use the ASP.NET Silverlight controls? Is this supported? 13](#_Toc234295529)

[How do I use ASP.NET controls but update them to be compatible with Silverlight 3? 13](#_Toc234295530)

# Introduction

The ASP.NET Server Controls for Silverlight™ (“ASP.NET Silverlight controls”), which are the ASP.NET MediaPlayer and Silverlight controls, have been removed from the Silverlight SDK for Silverlight version 3. This document provides guidance for developers who worked with these ASP.NET controls in previous versions of Silverlight. The information in this document is useful for developers who are moving from the Silverlight 2 SDK to the Silverlight 3 SDK and who either want to upgrade their existing applications or create new applications and are considering using the ASP.NET Silverlight controls.

For Silverlight version 3 and subsequent versions, the recommendation is for developers to use the object tag (and the Silverlight.js JavaScript file) to host Silverlight applications in a Web page. This document provides information about how to use the object tag to implement functionality that was previously available in the ASP.NET Silverlight control.

For applications that use the ASP.NET MediaPlayer control, developers can switch to using an object tag (and the Silverlight.js JavaScript file) to host the Silverlight media player application and use the player skins and interaction logic that is available through Expression Encoder.

Additional options for developers who are using the ASP.NET Silverlight or MediaPlayer controls are described in this document.

# Moving ASP.NET Silverlight Control Functionality from Silverlight 2 to Silverlight 3

If you are upgrading from the Silverlight 2 SDK to the Silverlight 3 SDK on a development computer, you have the following options for implementing the functionality represented by the ASP.NET Silverlight controls:

* Change your Web pages to use the object tag (and the Silverlight.js JavaScript file) and make any additional required changes. For more information, see [Implementing ASP.NET Silverlight Control Functionality in Silverlight 3](#_Implementing_ASP.NET_Silverlight_1) later in this document.
* Continue to use the ASP.NET Silverlight controls from the Silverlight 2 SDK in Web applications that are created with Silverlight 3. You can use the existing binary code for the controls, or you can download and compile the source code for the controls. For more information, see [Using the Silverlight 2 ASP.NET Controls with the Silverlight 3 SDK](#_Using_the_Silverlight) later in this document. If you choose this option, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.

# Effects of Installing the Silverlight 3 SDK and Uninstalling the Silverlight 2 SDK for the ASP.NET Silverlight Controls

Installing the Silverlight 3 SDK does not automatically remove the Silverlight 2 SDK. However, you can remove the Silverlight 2 SDK manually if you want.

**Note**   Uninstalling the Silverlight 2 Tools for Visual Studio 2008 SP1 does not remove the Silverlight 2 SDK. The Silverlight 2 SDK documentation is available at the following URL: <http://go.microsoft.com/fwlink/?LinkId=157094>.

If you leave the Silverlight 2 SDK on your computer after you install the Silverlight 3 SDK, the ASP.NET Silverlight controls remain in the Visual Studio® Toolbox. You can continue to use the ASP.NET controls from the Silverlight 2 SDK for applications that you build with the Silverlight 3 SDK. If you do, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.

If you uninstall the Silverlight 2 SDK, files are removed from the following location:

%WINDIR%/Program Files/Microsoft SDKs/Silverlight/v2.0

This step removes the assembly for the ASP.NET Silverlight controls (System.Web.Silverlight) and removes any additional resources in the folder, such as the MediaPlayer XAML (skin) files. (This step does not remove the assembly or skin files from the local folders in individual Web applications. However, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.) The assembly is found in the following location:

%WINDIR%/Program Files/Microsoft SDKs/Silverlight/v2.0/Libraries/Server

The MediaPlayer control skins are found in the following location:

%WINDIR%/Program Files/Microsoft SDKs/Silverlight/v2.0/Libraries/Server/MediaPlayerSkins

**Note**    By default, the assembly for the ASP.NET Silverlight controls is not installed in the GAC. If you have manually installed the assembly in the GAC, it will remain in the GAC when the Silverlight 2 SDK is uninstalled. In that case, if you uninstall the Silverlight 2 SDK, you must also manually remove the assembly from the GAC.

If you manually remove the Silverlight 2 SDK, the ASP.NET Silverlight controls are also removed from the Visual Studio 2008 Toolbox.

# Working with Existing Silverlight Solutions in the Silverlight 3 SDK

Silverlight solutions in Visual Studio 2008 consist of the Silverlight project (the project used to build code and associated XAML) and a Hosting Web Application that contains ASP.NET Web pages and HTML pages.

Silverlight 3 Tools for Visual Studio 2008 SP1 include the Silverlight 3 SDK. When you open a project with Silverlight 3 Tools, any Silverlight 2 project will be updated to target the Silverlight 3 runtime. The modification that occurs does not affect the Hosting Web Application.

When you build the solution, Visual Studio rebuilds the XAP file (which contains the compiled assembly, XAML, and resources) and copies the file to the associated Hosting Web Application’s clientbin folder.

The existing Hosting Web Application might still contain the assembly (System.Web.Silverlight) that contains the ASP.NET controls in the Application’s Bin folder. (For more information, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.) The assembly was always copied as part of the build process for projects in earlier versions of Silverlight.

**Note**   If you manually added the ASP.NET controls assembly to the GAC, the assembly is typically not in the clientbin folder.

Any XAML (skin) files that were used by the ASP.NET MediaPlayer control will normally have been copied to a local folder in the Hosting Web Application. These skins are not removed when you uninstall the Silverlight 2 SDK.

If you have not manually uninstalled the Silverlight 2 SDK, the ASP.NET Silverlight controls will still be in the Visual Studio 2008 Toolbox. You can still add these controls to ASP.NET Web pages.

When you run the Hosting Web Application, the application will continue to work. (However, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.) Similarly, if you move the application to a production server, it will continue to work. However, if you use Silverlight 3 features, you must make certain changes, as explained in this document.

# Using the Silverlight 2 ASP.NET Controls with the Silverlight 3 SDK

After you install the Silverlight 3 SDK and Silverlight 3 Tools for Visual Studio 2008 SP1, when you create a new Silverlight application, the Hosting Web Application contains only ASP.NET Web pages and HTML pages that use the object tag (and the Silverlight.js JavaScript file) to incorporate Silverlight functionality. The object tag is the preferred model for implementing Silverlight functionality in Silverlight 3. However, the following options are available to you if you want to use the ASP.NET Silverlight controls in your applications:

* Use the existing ASP.NET Silverlight controls in the Silverlight 3 solution. However, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.
* Get the source code for the ASP.NET controls, and then create and deploy your own assembly containing the controls.

## Using the Existing Silverlight Controls

If you have not uninstalled the Silverlight 2 SDK, the ASP.NET Silverlight controls are still in the Visual Studio 2008 Toolbox and you can drag an ASP.NET Silverlight control onto ASP.NET Web pages. When you do, the control is added to the page and an @ Register directive is added at the top of the page that maps the control’s prefix and name (unless the register directive is already defined in the Web.config file). You must manually add an ASP.NET ScriptManager control to the page as the first control inside the form element and before the markup for the ASP.NET Silverlight control. The following example shows markup for the ScriptManager and Silverlight controls.

<form id="form1" runat="server" style="height:100%;">

**<asp:ScriptManager ID="ScriptManager1" runat="server"></asp:ScriptManager>**

<div style="height:100%;">

<asp:Silverlight ID="Xaml1" runat="server"

Source="../ClientBin/sample.xap" MinimumVersion="2.0" Width="100%"

Height="100%" />

</div>

</form>

If you have uninstalled the Silverlight 2 SDK, the controls are no longer available in Visual Studio 2008. You can add the ASP.NET Silverlight controls to the new Hosting Web Application by doing the following:

1. Get a copy of the System.Web.Silverlight assembly from one of the following sources:

* On a computer where the Silverlight 2 SDK is still installed, get the assembly from the Silverlight 2 SDK installation folders. For information about the location of the assembly in the Silverlight 2 SDK, see [Effects of Installing the Silverlight3 SDK and Uninstalling the Silverlight 2 SDK for the ASP.NET Silverlight Controls](#_Uninstalling_the_SL2)  earlier in this document.
* The Bin folder of the Hosting Web Application for an application that was created using the Silverlight 2 SDK.
* The Web Application that contains the source code for the ASP.NET Silverlight controls. You can then build a new version of the assembly yourself. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

1. Copy the assembly to the local Bin folder of the Hosting Web Application.
2. Add a reference to the System.Web.Silverlight assembly in the assemblies section of the Web.config file.
3. Add tag mapping for the control in the Web.config file or in an @ Register directive in the page. The following example shows the @ Register directive in the page.

<%@ Register Assembly="System.Web.Silverlight"

Namespace="System.Web.UI.SilverlightControls" TagPrefix="asp" %>

If you intend to use the ASP.NET Silverlight controls from Silverlight 2 SDK, see [Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK](#_Known_Issues_in) later in this document.

## Modifying ASP.NET Silverlight Controls to Work with Silverlight 3

If you add new Silverlight 3 functionality to an existing Silverlight project, and if you want to continue to use the ASP.NET Silverlight controls, you must modify the controls to be compatible with Silverlight 3. This section provides details about how to do this.

### Setting the Silverlight Minimum Version

To use the ASP.NET Silverlight control in Silverlight 3, you must modify the MinimumVersion property of the controls to 3.0.40624.0, as shown in the following example:

<asp:Silverlight runat="server" Source="/clientbin/sample.xap"

MinimumVersion="3.0.40624.0" />

### Deep Linking (Navigation)

If you want to add deep linking functionality to an existing Silverlight application, and you want to continue to use the ASP.NET Silverlight controls, you must add an iframe element to the Web pages that contain the controls. The iframe element must have the ID \_sl\_historyFrame (case-sensitive), and the element's style must set the element to not be visible. The following example shows the markup for the Silverlight control and the required iframe element.

<asp:Silverlight runat="server" Source="/clientbin/sample.xap" />

<iframe id="\_sl\_historyFrame"

style='visibility:hidden;height:0;width:0;border:0px'>

</iframe>

### Hardware Acceleration

Silverlight 3 supports hardware acceleration. A new property, EnableGPUAcceleration, lets you toggle this feature on and off. The property is supported as a parameter of the object tag in Silverlight 3.

The following example shows how to set the EnableGPUAcceleration property in the object tag.

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<param name="EnableGPUAcceleration" value="true"** />

</object>

### Cache Visualization

Silverlight 3 also supports cache visualization, which provides graphical feedback that indicates which objects are bitmap-cached. A new property, EnableCacheVisualization, lets you toggle this feature on and off. The property is supported as a parameter of the object tag in Silverlight 3.

The following example shows how to set the EnableCacheVisualization property in the object tag.

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<param name="EnableCacheVisualization" value="true" />**

</object>

## Obtaining the Source Code for the ASP.NET Silverlight Controls

The source code for the ASP.NET controls, including associated JavaScript files and MediaPlayer skins, is available for download from the MSDN Code Gallery Web site at the following URL:

<http://go.microsoft.com/fwlink/?LinkId=156721>

You can modify the controls and then use them in Silverlight 2 or Silverlight 3 applications. For more information about how to work with the source code for the controls, including how to register them in the Web.config file, see the documentation that accompanies the source code.

The source code is based on the controls that are in the Silverlight 2 SDK, but contains additional properties for the hardware acceleration and cache visualization features that are available in Silverlight 3. For information about these additions and other fixes, see the document that accompanies the project. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

# Known Issues in Upgrading to Silverlight 3 and Using the ASP.NET Server Controls from Silverlight 2 SDK

## Cross-Domain XAP

If you load a XAP source file from a cross-domain location (that is, you load the XAP file from a location other than the server that provides the Web page), and you enable scripting in the Silverlight application (by setting HtmlAccess="enabled" for the ASP.NET Silverlight controls), a JavaScript error may occur.

You can resolve this issue by doing any of the following:

* Modify the Web page to use the **object** tag (and accompanying Silverlight.js file).
* Set **HtmlAccess** to false on the Silverlight controls if they do not require scripting or HTML DOM access.
* Compile the controls from the source code that is available for download from the Code Gallery build. The Code Gallery versions of the control have a fix for this issue. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

## CPU Utilization While the Application is Idle

If you are loading a large XAML or XAP file with the ASP.NET Silverlight controls, in some cases the client CPU is being used, even though there is no activity in the Silverlight application. You can resolve this issue by doing any of the following:

* Modify your application to use the **object** tag, and do not handle the sourceDownloadProgress or the sourceDownloadComplete events for the plug-in.
* Add JavaScript like the following example to the Web application:

<asp:Silverlight

OnPluginSourceDownloadComplete="downloadComplete" ... />

<script type="text/javascript">

function downloadComplete(sender, args) {

var sb =

sender.get\_element().content.FindName("LoadingAnimation2");

if (sb) sb.Stop();

}

</script>

* Compile the controls from the source code that is available for download from the Code Gallery build. The Code Gallery versions of the control have a fix for this issue. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

## Upgrading Silverlight Projects and Solutions

When you upgrade a Silverlight 2 application to Silverlight 3, and you have uninstalled the Silverlight 2 SDK, you might be unable to run the Hosting Web Application if you use the **Clean Solution** command from the **Build** menu.

Uninstalling the Silverlight 2 SDK removes the ASP.NET Silverlight controls assembly from the installation location on your machine. Using the **Clean Solution** command also removes that assembly from the Bin folder of the Hosting Web Application. When you re-build the solution, the build action can no longer locate the installed ASP.NET Silverlight controls assembly and copy it to the Bin folder.

This doesn’t occur if you clean only the Silverlight application project within the solution.

You can resolve this issue by doing any of the following:

* Modify your application to use the object tag.
* Reinstall the Silverlight 2 SDK.
* Manually copy the ASP.NET Silverlight controls assembly to the Bin folder from some other location.

## Sharing Silverlight Projects and Solutions

If you are sharing a Silverlight 2 solution or project as a sample, the shared solution might include the Hosting Web Application but not include the assembly for the ASP.NET Silverlight controls (System.Web.Silverlight). In that case, running the Hosting Web Application on the target machine might fail.

Possible reasons that the solution does not include the assembly are the following:

* During the build process in Visual Studio, you ran the **Clean Solution** command. In that case, the results of build are removed from the Bin folder of the Hosting Web Application. If you then share the solution, the required assembly for the ASP.NET Silverlight controls is not included. The consumer of the solution might not have that assembly installed locally.
* You share only the Silverlight project from the solution, and the developer who is using the project does not have a Hosting Web Application that has access to the ASP.NET Silverlight controls assembly.
* The consumer of the solution expects to use the ASP.NET Silverlight controls and has only the Silverlight 3 SDK installed, and the ASP.NET Silverlight controls assembly is not located in the Bin folder of the Hosting Web Application that the consumer uses for the application.

If you are the developer who is sharing the solution, you can make sure that the missing ASP.NET Silverlight controls assembly is available by doing one of the following:

* Modify the Hosting Web Application to use the HTML pages and use the **object** tag.
* If you have run the **Clean Solution** command, manually drag the required assembly to the Bin folder of the Hosting Web Application after the solution has been built.

If you are the developer who is consuming the shared solution, you can do one of the following:

* Remove the pages that use the ASP.NET Silverlight controls and use the **object** tag.
* Install the Silverlight 2 SDK and rebuild the Hosting Web Application. In that case, the ASP.NET Silverlight controls assembly is automatically copied to the Bin folder of the Hosting Web Application.
* Manually copy the ASP.NET Silverlight controls assembly from another application to the Bin folder of the Hosting Web Application.
* Download and compile the ASP.NET Silverlight controls and copy the resulting assembly to the solution. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

# Effect on Designer Tools of Removing ASP.NET Silverlight Controls

The removal of the ASP.NET Silverlight controls for the Silverlight 3 SDK results in changes in some designer tools.

For Visual Studio 2008, the Silverlight 3 Tools for Visual Studio 2008 SP1 create a Hosting Web Application that contains ASP.NET Web pages and HTML pages that reference the object tag. There is no reference to the ASP.NET controls in the Hosting Web Application. However, if you do not uninstall the Silverlight 2 SDK, the Toolbox still contains the ASP.NET Silverlight controls.

For Expression Encoder, there is no change. Expression Encoder projects produce HTML pages that use the object tag and Silverlight.js for XAPs and for XAML files.

# Implementing ASP.NET Silverlight Control Functionality in Silverlight 3

The examples in this section show various ways to implement functionality in Silverlight 3 that is available in the ASP.NET Silverlight controls. The examples show how to use a XAP file in Silverlight 3. You can replace the XAP file with a XAML reference instead.

In these cases, you will use the **object** tag (and the Silverlight.js JavaScript file), for handling the installation of Silverlight for end users. For more details, see the Silverlight documentation.

## Setting the Source Location of the XAP or XAML File

In Silverlight 2, you specify the source location of a XAP (or XAML) file by setting the Source property of the Silverlight control, as shown in the following example:

<asp:Silverlight runat="server" Source="/clientbin/sample.xap" />

In Silverlight 3, you can perform this task by setting a parameter of the plug-in, as shown in the following example:

<object type="application/x-silverlight">

**<param name="Source" value="/clientbin/sample.xap" />**

</object>

## Setting the PlugInNotInstalledTemplate Property

In Silverlight 2, you can include a PluginNotInstalledTemplate element in the markup for a Silverlight control in order to specify text that will appear if the client does not have Silverlight installed. The following example shows how this is specified in markup.

<asp:Silverlight runat="server" Source="/clientbin/sample.xap">

**<PluginNotInstalledTemplate>**

**<%-- Custom template --%>**

**</PluginNotInstalledTemplate>**

</asp:Silverlight>

In Silverlight 3, you can use the following syntax in the object tag:

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<!-- Custom template -->**

</object>

## Using Stretch Mode

Silverlight 3 supports zooming for the XAP or XAML file. In the Silverlight 2 SDK, the ASP.NET controls provided a stretch feature that is not supported in Silverlight 3. If you want to retain this functionality, you can add code to achieve a similar result.

In the Silverlight 2 SDK, you use markup like the following in the Silverlight control to implement stretch mode:

<asp:Silverlight runat="server" Source="/clientbin/sample.xap"

ScaleMode="Stretch" />

In Silverlight 3, zoom functionality requires JavaScript that handles the Silverlight OnResize event. In the event handler, you apply a ScaleTransform object to the root of the XAML document. The following example shows the object tag that binds a function to the OnResize event.

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<param name="OnResize" value="HandleResize" />**

<!-- The default plugin-not-installed template goes here -->

</object>

The following example shows the stub of a JavaScript function that can handle the event.

<script type="text/javascript">

function HandleResize(obj) {

// Use ActualWidth and ActualHeight to calculate the transform.

}

</script>

## Setting the Minimum Version

In Silverlight 2, you set the minimum Silverlight plug-in version in the markup for the Silverlight control as shown in the following example:

<asp:Silverlight runat="server" Source="/clientbin/sample.xap"

**MinimumVersion="2.0.0.0"** />

In Silverlight 3, in the object tag, set the minRuntimeVersion property to 3.0.40624.0, as shown in the following example:

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<param name="minRuntimeVersion" value="3.0.40624.0" />**

<!-- The default plugin-not-installed template goes here -->

</object>

## Handling the Load Event or Other Events

To handle events that are raised by the Silverlight plug-in, in Silverlight 2, you can bind a function to an event in the Silverlight control markup as shown in the following example:

<asp:Silverlight runat="server" Source="/clientbin/sample.xap"

**OnClientPluginLoaded="onLoad"** />

In Silverlight 3, you can bind plug-in events to handlers in the object tag. The following example shows the object tag in which the plug-in’s onLoad event is bound to a function named onLoad.

<object type="application/x-silverlight">

<param name="Source" value="/clientbin/sample.xap" />

**<param name="onload" value="onLoad" />**

<!-- The default plugin-not-installed template goes here -->

</object>

The following example shows the stub of a JavaScript function that can handle the event.

<script type="text/javascript">

function onLoad(obj) {

// To access the object with <asp:Silverlight>:

// obj.get\_element()

// To access the object with the object tag:

// obj

}

</script>

## Programmatically Changing the Plug-In in Server Code

If you used the ASP.NET controls and you modified any properties of the control as a result of user input or other business logic, you must make changes to the application when you substitute the object tag for the control. Typically, you must write custom code that dynamically modifies markup in the page. You can do this by using inline code snippets such as the following:

<% =myParam %>

This snippet might call a user-defined function in server code or it might reference a user-defined property. You might also create server code that adds markup dynamically to a Placeholder or Literal control. The following example shows how to use a Literal control to dynamically add markup to a parameter of an object tag.

<object type="application/x-silverlight">

<param name="Source" **value='<asp:Literal id="Source" runat="server"/>'** />

</object>

You can then specify a value for the source by setting the Text property of the Literal control. Use the ResolveClientUrl method to map application paths, as shown in the following example:

void Page\_Load(object sender, EventArgs args) {

Source.Text = **ResolveClientUrl("~/clientbin/sample.xap")**;

}

## Creating a Media Player

If you want your application to include a media player, you have the following options:

* Use the new managed media player skins that are shipped with Expression Encoder and associate these with an object tag.
* Continue to use the ASP.NET MediaPlayer control from the Silverlight 2 SDK to define the player, the skins, and the associated media source.
* Download the source code for the MediaPlayer control from the code gallery project and modify it. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1) earlier in this document.

**Note**    You can use the object tag and use the associated skin files and JavaScript files from the ASP.NET MediaPlayer control. These files are defined in the source code project for the ASP.NET MediaPlayer control. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1).

# FAQ

## How can I use the ASP.NET MediaPlayer Control with a new skin file?

When you uninstall the Silverlight 2 SDK, the folder that contains the XAML (skin) files is removed. If an existing application needs to reference the skin file that you want to use, you can get the skin file from the local files for that application. If you want to work with a skin file that is not already in use in an existing application, you must either reinstall the Silverlight 2 SDK or get the skin file from the downloadable source code. For information about downloading the source files, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1) earlier in this document.

## How can I work with media using Silverlight 3?

You can work with media files using Silverlight 3 and use the Silverlight samples that you can obtain when you use Expression Encoder 3.

## Can I continue to use the ASP.NET Silverlight controls? Is this supported?

You can continue to use the ASP.NET Server Controls for Silverlight, either by using the Silverlight 2 SDK release, or by downloading the source code for the controls and compiling it yourself.

## How do I use ASP.NET controls but update them to be compatible with Silverlight 3?

You can download the source files for the ASP.NET Silverlight controls from the ASP.NET Server Controls for Silverlight Code Gallery project, compile the controls, and add them to Silverlight 3 applications. For more information, see [Obtaining the Source Code for the ASP.NET Silverlight Controls](#_Obtaining_the_Source_1) earlier in this document.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

 2009 Microsoft Corporation. All rights reserved.

Microsoft, Silverlight, and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.