GENERAL

microsoft silverlight tutorial

Published: September 3, 2026 | Index ID: #-

Microsoft Silverlight Tutorial: A Comprehensive Guide for Modern Developers

For many developers, the name “Silverlight” conjures memories of the early days of rich internet applications (RIAs). Though its popularity has waned, Silverlight remains an important chapter in the evolution of web and desktop development. This tutorial offers a deep dive into Microsoft Silverlight, covering everything from its history and core features to practical, hands on instructions that let you create, test, and deploy your own Silverlight applications.

Whether you’re a seasoned developer looking to refresh your knowledge or a newcomer curious about legacy technologies, this guide will equip you with the concepts, tools, and best practices needed to master Silverlight. By the end, you’ll have a solid foundation in Silverlight development and a clear understanding of why it’s still relevant for certain use cases, such as internal corporate portals or cross platform desktop apps.

What Is Microsoft Silverlight?

Microsoft Silverlight is a cross platform, plug in based framework designed to build rich, interactive applications that run in web browsers and on desktops. Introduced in 2007, Silverlight was conceived as a lightweight alternative to the more heavyweight Windows Presentation Foundation (WPF). It leveraged XAML (Extensible Application Markup Language) for UI design and C# or VB.NET for application logic, mirroring the familiar .NET ecosystem.

Silverlight’s primary goal was to bring rich media, animation, and data driven experiences to the web without requiring users to install bulky software. It supported vector graphics, audio/video playback, and real time communication through the **SignalR**library. The framework also offered a sandboxed environment that ensured applications could run safely across different browsers and operating systems.

Key Features of Silverlight

Silverlight introduced several groundbreaking features that set it apart from traditional web technologies:

- Declarative UI with XAML: Designers and developers could collaborate more effectively by separating visual layout from business logic.
- Rich media support: Native playback of MP3, WMA, and Windows Media video, with support for streaming and adaptive bitrate.
- Animation and storytelling: Storyboards and visual states made it easy to create smooth transitions and interactive storytelling.
- Data binding and MVVM: Strong data binding capabilities facilitated the Model View ViewModel pattern, promoting clean architecture.
- Cross platform deployment: Applications could run on Windows, macOS, Linux, and even mobile devices (via Silverlight for Windows Phone).
- Extensibility: Custom controls, behaviors, and third party libraries could be integrated seamlessly.

Silverlight Versions Overview

Silverlight evolved through several major releases, each adding new capabilities and improving performance:

1. Silverlight 1.0 (2007): Basic XAML support, simple controls, and limited media.
2. Silverlight 2 (2008): Introduced MVVM support, improved media playback, and added the WebBrowser control.
3. Silverlight 3 (2009): Added 3D graphics, improved animation, and introduced the Silverlight Toolkit.
4. Silverlight 4 (2010): Enhanced printing, file I/O, and clipboard integration.
5. Silverlight 5 (2011): Introduced Windows 8 support, native Win32 API access, and improved performance.

Although Microsoft discontinued active development in 2015, the last version (Silverlight 5) remains the most feature rich and widely supported.

Setting Up Your Development Environment

Before you dive into coding, you'll need to set up a development environment that supports Silverlight. The following steps assume you are using Windows, but Silverlight projects can be built on macOS and Linux with the right tooling.

Prerequisites

- Visual Studio 2010 or later (Community edition is free).
- Silverlight 5 Runtime and SDK.
- Internet Explorer 11 or a modern browser that still supports the Silverlight plug in (Chrome, Firefox, Edge).
- Optional: Visual Studio 2017/2019 for legacy project support.

Installing Visual Studio

Download the Visual Studio installer from Microsoft's official website. During installation, select the *.NET desktop development* workload and ensure the *Silverlight development tools* are included. If you already have Visual Studio, you can add the Silverlight workload via the *Modify* option in the installer.

Installing Silverlight SDK

After installing Visual Studio, download the **Silverlight 5 SDK** from the Microsoft Download Center. Run the installer and follow the prompts. This will add the necessary project templates, runtime components, and debugging tools.

Configuring Browser Plug in

Most modern browsers have phased out NPAPI support, which Silverlight relies on. If you're using Chrome or Edge, you'll need to enable the plug in manually or use a legacy browser like Internet Explorer 11. In IE, go to *Tools !' Manage Add ons* and ensure the Silverlight plug in is enabled.

Building Your First Silverlight Application

With the environment ready, let's create a simple "Hello, World" Silverlight app to see the workflow in action.

Create a New Project

1. Open Visual Studio and select File !' New !' Project.
2. Under Visual C# !' Silverlight, choose Silverlight Application.
3. Give your project a name (e.g., SilverlightHelloWorld) and click Create.
4. When prompted, select the target browser (Internet Explorer 11) and click OK.

Understanding the Project Structure

A new Silverlight project contains the following key files:

- App.xaml – Application-level resources and startup logic.
- MainPage.xaml – The default user interface page.
- MainPage.xaml.cs – Code behind for MainPage.
- SilverlightApplication.csproj – Project configuration file.

Open *MainPage.xaml* to see the default XAML markup. It includes a **Grid** layout with a **TextBlock**displaying “Hello, World!”

Adding XAML Markup

Replace the existing **Grid** content with a simple form that collects a user’s name and displays a greeting:

```
&lt;Grid&gt;
    &lt;StackPanel HorizontalAlignment="Center" VerticalAlignment="Center"&gt;
        &lt;TextBlock Text="Enter your name:" Margin="0,0,0,10"/&gt;
        &lt;TextBox x:Name="NameInput" Width="200" Height="25" Margin="0,0,0,10"/&gt;
        &lt;Button Content="Greet" Width="100" Click="GreetButton_Click"/&gt;
        &lt;TextBlock x:Name="GreetingText" Margin="0,10,0,0" FontSize="16" FontWeight="Bold"/&gt;
    &lt;/StackPanel&gt;
&lt;/Grid&gt;
```

Adding Code Behind

Open *MainPage.xaml.cs* and implement the **GreetButton_Click** event handler:

```
private void GreetButton_Click(object sender, RoutedEventArgs e)
{
    string name = NameInput.Text.Trim();
    if (!string.IsNullOrEmpty(name))
    {
        GreetingText.Text = $"Hello, {name}!";
    }
    else
    {
        GreetingText.Text = "Please enter a name.";
    }
}
```

Run the project by pressing **F5**. The application will launch in the chosen browser, and you should see the greeting functionality working.

XAML Basics

XAML (Extensible Application Markup Language) is the heart of Silverlight UI development. It allows developers to define controls, layouts, and data bindings declaratively. Understanding XAML fundamentals is essential for building complex interfaces.

Layout Panels

Silverlight offers several layout panels that automatically arrange child elements:

- StackPanel: Arranges children vertically or horizontally.
- Grid: Provides a flexible two dimensional layout with rows and columns.
- Canvas: Allows absolute positioning via Canvas.Left and Canvas.Top.
- DockPanel: Dock children to the top, bottom, left, right, or center.

Controls

Common Silverlight controls include **Button**, **TextBox**, **ListBox**, **DataGrid**, **MediaElement**, and **Slider**. Each control has a rich set

Tags: #microsoft #silverlight #tutorial