



Windows Embedded Compact 7: Extensible Platform to Power Dynamic User Experiences

Windows Embedded Technical Article

December 2010

Applies to: Windows Embedded Compact 7

Summary: Windows® Embedded Compact 7 provides the tools and components to build your own customized and stand-out user experiences that engage users and extend the world of Windows across Windows® 7-based computers and other devices. Compact 7 also provides a clear separation between developers and designers, allowing ease-of-development to enable quicker development and faster time to market.

Introduction

In today's device marketplace, providing a rich, compelling, branded user experience for devices is a must. Users expect a device to be dynamic and robust, with a nimble, easy-to-use user interface (UI). User experience is a strong motivator that influences how and what users purchase, adopt, and recommend to others.

Providing the means for developers and designers to create such user experiences for devices more seamlessly, and bring them to market with greater ease and less cost, is also a must for companies that produce embedded devices. User tastes and expectations can change quickly, and increasing the speed to bring a device to market is vital to encourage and maintain device sales.

Windows® Embedded Compact 7 provides the tools and components to build your own customized and stand-out user experiences that engage users and extend the world of Windows across Windows® 7-based computers and other devices. Compact 7 also provides a clear separation between developers and designers, allowing for quicker overall device development and faster time to market.

Using Compact 7, you can:

- Create an immersive and intuitive user interface for your device using Microsoft® Silverlight® for Windows Embedded, which allows you to build unique, differentiated UI using a simpler, more seamless design and development process.
- Extend the world of Windows and connect computers with your device and vice versa for a more complete user experience.
- Use Internet Explorer® Embedded to build a custom browser for a specific device or market.

Create an Immersive and Intuitive User Interface

Windows Embedded Compact 7 provides you with Silverlight for Windows Embedded for UI design and development. Silverlight for Windows Embedded offers several advantages that you can use to delineate the responsibilities of developers and designers. Making these roles far clearer and cleaner through a more unified production process enables both to more effectively create rich, robust, and unique UI for devices.

What Is Silverlight for Windows Embedded?

Silverlight for Windows Embedded is a native code (Visual C++®) UI framework that delivers impressive graphics with optimized performance on embedded devices. The UI framework is designed to build UI shells and core applications for devices and can handle tasks such as events, initializing, and manipulating visual elements, and runs very efficiently on specialized embedded hardware.

Note: Hardware acceleration can be utilized if a graphics processing unit (GPU) is available on the device and other requirements are met.

The following figure provides a high-level architectural view of Silverlight for Windows Embedded.



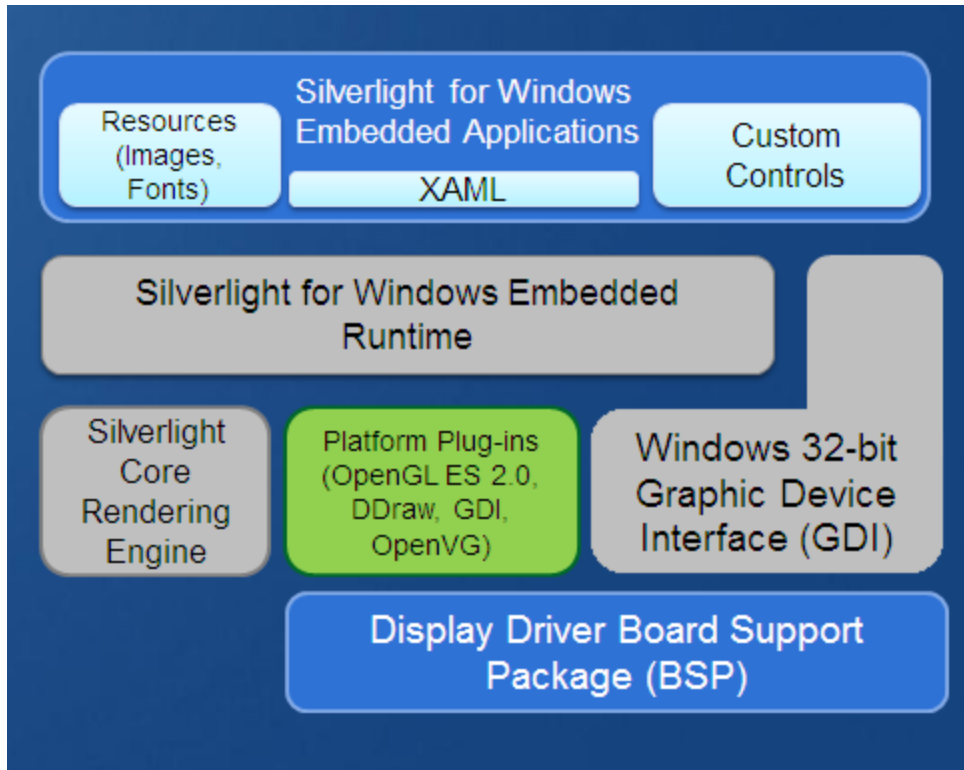


Figure 1. Silverlight for Windows Embedded architecture

For more information about Silverlight for Windows Embedded and its components, see Microsoft Silverlight for Windows Embedded (<http://www.microsoft.com/windowseembedded/en-us/products/windowsece/silverlightforwe.msp>) and Silverlight for Windows Embedded (<http://msdn.microsoft.com/en-us/library/ee502198.aspx>).

Why Use Silverlight for Windows Embedded?

Silverlight for Windows Embedded provides a number of advantages when designing and developing device UI. Some of these advantages include:

- Providing a level of developer and designer differentiation that speeds up UI development and creation.
- Decreasing the time to market spent building and refining device user experiences.
- Performance advantages that allow for rich graphics and a robust rendering engine.
- Providing developers and designers with the rich and powerful features of Silverlight for Windows Embedded.

Developer and Designer Separation

Separating the roles and responsibilities between the developer and the designer is a key element in building device UI quickly and as seamlessly as possible. Silverlight for Windows Embedded is ideal for helping establish and maintain this separation.

By using the designer visual tool Microsoft Expression Blend® 3 (a tool that allows for graphic UI creation), designers can now create UI that serves as an immediate, translatable deliverable. Rather than a designer having to build UI in Visual C++ and play the role of a developer, the designer builds the UI, and the developer handles the program logic required for the device UI.

The Expression Blend project the designer creates is actionable and can be implemented immediately by a developer, unlike a document or wire frame created by a designer that must be translated into code by the developer. Fully-featured UI shells and applications can be created by the designer. The developer then only needs to take the Expression Blend 3-designed UI and enable any events, buttons, or other UI features rather than building the entire UI from scratch. This contributes to a consistent device UI instead of a collection of separate screens built at different times using an inconsistent method (like the designer sometimes building the UI and the code or the developer doing the same thing). This cleaner process removes the burden of attempting to translate a designer's vision from scratch for the developer, and ensures better separation of the designer and developer roles that is reinforced with each iteration of the device UI.

The following figures illustrate the kinds of shells that designers and developers can build using Expression Blend 3 and Silverlight for Windows Embedded.

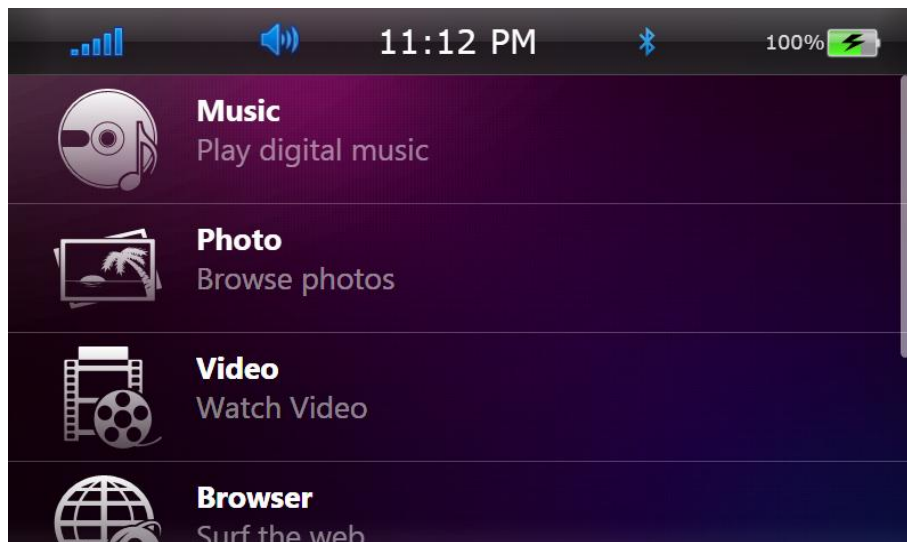


Figure 2. Sample shell



Figure 3. Sample shell

Overall UI improvements in Silverlight for Windows Embedded on the developer and designer side also help make the process more robust, simple, and unified. Some examples include:

- **The new Silverlight engine – Silverlight 3.**

The core of Silverlight for Windows Embedded is the same as Silverlight. This allows designers and developers to quickly and easily access and incorporate Silverlight and Expression Blend updates.

- **Databinding.**

Databinding allows designers to associate data objects to visual elements (like an image and text in a list box to represent an album cover, the artist's name, or other related information) in Extensible Application Markup Language (XAML). (For the sake of this example, this is all done by the designer.) The designer then hands off the shell or application with these data associations to the developer, who can then simply attach the data set to the visual element at runtime based in code. The databinding process provides a considerable performance boost, as Silverlight for Windows Embedded iterates through the data set instead of developer-written code. The visual elements are dynamically created for each data column in every data row in a data set.

- **Binary XAML (BAML).**

BAML is binary XAML and is new to Silverlight for Windows Embedded. It allows developers to bundle a significant amount of data into a small package that can be quickly parsed. It also allows for quicker XAML parsing on demand. This means the XAML does not have to be kept in a device's memory all at once, which greatly boosts the performance of the device. BAML also reduces the overall XAML load and parsing

time performed by Silverlight for Windows Embedded tools that compile the XAML into binaries. This is especially helpful in situations where XAML files exceed a device's footprint when parsed into memory, such as with extremely complex human-machine interface (HMI) devices or automotive infotainment devices.

- **Automatic cache compositioning.**

During the rasterization phase Silverlight for Windows Embedded converts vectors into pixels on the screen. The composition process then layers memory buffer contents (bitmap images) on top of one another, taking into consideration opacity and transformations, such as scale, skew, or translation. The composed image that results displays directly on the device screen.

Note: Composition phase hardware is accelerated if a GPU is available on the device.

Automatic cache compositioning provides a number of benefits, including quick transformations for fast animations, easier XAML exchanges from designer to developer (and vice versa), and that XAML no longer must be modified with cache attributes.

The following figure illustrates how the Silverlight for Windows Embedded framework enables developers and designers to build and provide a consistent user experience across desktop applications, devices, and web applications.





Figure 4. Building user experiences with Silverlight for Windows Embedded

This framework allows developers and designers to create innovative user experiences. Companies like yours can now take advantage of the huge design community of Silverlight-based web and desktop developers to create rich, robust device UI.

Note: Silverlight for Windows Embedded and web applications will not run on your device by itself. However, Silverlight for Windows Embedded allows you to use XAML to create similar user experiences across desktop applications.

Decrease Your Time to Market

Decreasing your device's time to market is a vital consideration. Using Windows Embedded Compact 7 to build device UI, it now takes weeks, rather than months, to get your unique, branded, and robust UI running on your device. Some of the ways that Compact 7 reduces your time to market include:

- Interoperability between Expression Blend 3 XAML projects and Silverlight for Windows Embedded. This interoperability (how template Visual C++ project code is built on an Expression Blend 3 XAML project) provides the following benefits:
 - Design markup XAML and Visual C++ code separation makes it faster and easier to change or completely replace UI based on user requests, market findings, and usability studies.
 - Collaboration between developer and designer is fostered and the overall development process speeds up, making UI and UX design and development much simpler and clearer in its delineation of responsibilities.
 - Faster prototyping when using Expression Blend 3 and Windows® Internet Explorer® 7 on the desktop creates a more efficient design and development process.
- The Windows Embedded Silverlight Tools Wizard. The wizard allows for Expression Blend project and Visual C++ code to work together as it generates objects and classes in Visual C++ code from Expression Blend 3 XAML projects. The wizard provides the following benefits:
 - If a designer changes or updates the project UI, the project UI can be run through the tool again by the developer and the corresponding Visual C++ code is changed or updated accordingly.
 - The design XAML and image assets can be completely packaged into a single resource. This means you will not have to worry about where the XAML files or image assets reside.
 - Boilerplate code, UI shell, and UI applications can be generated and taken advantage of again as needed. This means the UI is reusable and does not need to be built from scratch for each instance it is needed.



Performance Advantages

Embedded device designers and developers must often optimize their device-based UI to perform on limited hardware platforms. The advent of rich UI has also intensified performance issue concerns by adding more complex paradigms for displaying and interacting with devices. The central performance tradeoff now is balancing UI complexity with hardware capability. In short, a complicated, multilayered user experience requires greater capability in the hardware platform.

Silverlight for Windows Embedded delivers rich and robust graphics that provide an optimized performance on embedded devices. Given that accelerating the rendering process is important to achieving optimal performance results, by using Silverlight for Windows Embedded as the rendering engine, all of the optimization is accomplished through Silverlight for Windows Embedded to increase performance. For example, raw memory needs, like lower-level graphics, are handled by Silverlight for Windows Embedded. This helps solve performance bottlenecks commonly experienced by designers and developers, as they can now create consumable device UI that meets hardware, software, and user requirements and needs that are at the forefront of design for embedded devices.

Using Silverlight for Windows Embedded provides the best of both worlds, namely rich graphics and a robust rendering engine.

Silverlight for Windows Embedded Features

You can use Silverlight for Windows Embedded to build a complete user experience for your device that looks and behaves seamlessly for the user, as well as your own unique, differentiated, branded UI for your device.

Note: For examples of such UI, see the figures displayed on page 4 of this document.

Some of the features in Silverlight for Windows Embedded include:

- The Silverlight Visual C++ application programming interface (API) and programming model that integrates with the Windows Embedded Compact 7 operating system. This helps shorten the learning curve for developers familiar with programming in Visual C++ for Compact 7.
- Visual C++ classes that developers can use to create or customize visual appearance and behavior of UI elements.
- Support for advanced UI features including gradients, transformations, and animations, so that developers can create interactive Silverlight-based controls in UI for embedded applications.
- Compatibility with Silverlight 3 XAML and a set of equivalent classes for supported XAML elements for developers and designers familiar with using Silverlight 3 XAML.



- Run-time support for displaying XAML UI so that organizations like yours can design UI for applications entirely in XAML.
- Ability to dynamically change the UI at run-time by using Visual C++.
- Separation of programming logic and UI design to encourage XAML designers to focus on designing user experiences and developers to focus on integration, programming logic, and run-time behavior.

Extend the World of Windows to Your Device

Most user experiences are still largely centered on the computer, which serves as the hub for things like content storage (photos), productivity (work and business-related endeavors), and media playback (music and movies). This will likely not change in the near future, and the computer will continue to be the center of most user experiences.

However, devices will continue to play key contributing roles to user experiences, and extending them to the world of Windows and making device integration more seamless and inviting to computers (and vice versa) is vital. Some examples of how Windows Embedded Compact 7 connects computers with your device for a more holistic and complete user experience include:

- The ability for devices to connect, consume, and play back rich media with Compact 7 features like:
 - The Microsoft DirectShow® multimedia pipeline that provides richer media streaming support with updated MPEG-2, MPEG-4, HTTP, and high-definition support.
 - Simplified media management with the new Media Library.
 - A flexible plug-in architecture that supports other components.
- Seamless connection to the Windows® 7 operating system like:
 - Simplified device management through integration with Device Stage™.
 - Transfer of user data and media using Windows 7 with Media Transfer Protocol (MTP) support.
 - Play-to functionality to stream multimedia to and from computers running Windows 7.
- Working with Digital Living Network Alliance (DLNA)–based devices.

Internet Explorer Embedded

Internet Explorer® Embedded is a feature-rich browser control that provides the ability to build a custom browser for a specific device or market. The Internet Explorer Embedded browser is a version of Windows® Internet Explorer® 7. In addition to the features that come with Internet Explorer 7, Internet Explorer Embedded includes the following:



- Customizable XAML UI that allows you to:
 - Use Silverlight for Windows Embedded to make your custom browser UI consistent with the rest of your device.
 - Make your custom browser UI appear unified to the overall device user experience rather than as a disparate experience.

The following figure provides an example of a seamless custom browser and device user experience based on Internet Explorer Embedded.



Figure 5. Seamless custom browser UI and device user experience

- High fidelity browsing experiences like:
 - An updated Internet Explorer browsing engine.
 - Desktop-style browsing that supports tabs, panning, zooming navigation (including full-screen, thumbnail, and normal views), and multitouch capabilities.
- Access to rich multimedia content (like web applications) with support for Adobe Flash 10. This is important, as most dynamic web content currently uses Flash.
 - Flash 10 is often used as a way to build specialized device UI within a browser. It allows a single web application to function on a portable or distributed specialized device.

- Immersive experiences with natural input like:
 - Support for touch input.
 - Multitouch browser integration for mobile device experiences. A developer can make the mobile device application multitouch capable and behave like a browser on your device.
 - Support for custom user gestures. Custom gesture recognition can be developed that allows the device operating system to send the device gesture engine continuous touch information. If those touch points are recognized by any of the standard or customer gesture recognition, the custom application is sent a simple gesture event for it to respond to. This functionality frees the developer from the complexity of gesture recognition when activating the device UI.
- APIs that can communicate over TCP/IP. A developer can use these APIs to:
 - Enable a native or managed device application to use web services and extend to the cloud.
 - Connect your device to a cloud computing service through a web application and access such applications through your custom browser.
 - Create web applications on your device that can satisfy device needs like portability, battery life, or small form factor requirements.
 - Connect your device to WiFi networks.

Conclusion

Windows Embedded Compact 7 provides the tools and components to build your own customized and stand-out user experiences that engage users and extend the world of Windows across Windows 7-based computers and other devices. Compact 7 also provides a clear separation between developers and designers, allowing ease-of-development for both to enable quicker development and faster time to market.

Using Compact 7, you can:

- Create an immersive and intuitive user interface for your device using Silverlight for Windows Embedded, which allows you to build unique, differentiated UI while also providing a simpler, more seamless design and development process.
- Extend the world of Windows and connect computers with your device, and vice versa for a more holistic and complete user experience.
- Use Internet Explorer Embedded to build a custom browser for a specific device or market.



For more information:

Visit the Windows Embedded website:

<http://www.windowseembedded.com>

Glossary

The following acronyms appear in this document:

- **Application Programming Interface (API)**. A set of routines that an application uses to request and carry out lower-level services performed by a computer's operating system.
- **Binary XAML (BAML)**. Allows developers to bundle a significant amount of data into a small package for quicker XAML parsing on demand.
- **Digital Living Network Alliance (DLNA)**. An alliance of leading companies in the consumer electronics, mobile and personal computer industries focused on delivering an interoperability framework of design guidelines based on open industry standards to complete the cross-industry digital convergence. Microsoft is a member of this alliance.
- **Extensible Application Markup Language (XAML)**. An XML-based language used to represent a tree of objects. Events generated by these objects can be handled using any Microsoft .NET programming language.
- **Graphics Processing Unit (GPU)**. A specialized microprocessor that offloads and accelerates 3D or 2D graphics rendering from the microprocessor used in embedded systems, mobile phones, personal computers, workstations, and game consoles.
- **Human-Machine Interface (HMI)**. The user interface in a manufacturing or process control system.
- **Media Transfer Protocol (MTP)**. A Microsoft enhancement to the picture transfer protocol (PTP), starting with Windows Media Player 10 in Windows® XP. MTP extends PTP to music players and other devices and supports meta-data such as titles and artist names.
- **Moving Picture Experts Group 2 (MPEG-2)**. The designation for a group of audio and video coding standards agreed on in 1994 by MPEG (Moving Pictures Experts Group) and published as ISO standard 13818.



- **Moving Picture Experts Group 4 (MPEG-4).** The designation for a group of audio and video coding standards agreed on in 1998 by MPEG. MPEG-4 is primarily designed to handle low bit rate content, from 4,800 bit/s to approximately 4 Mbit/s. The primary uses for the MPEG-4 standard are web (streaming media) and CD distribution, conversational (videophone), and broadcast television. MPEG-4 absorbs many of the features of MPEG-1 and MPEG-2.
- **User Interface (UI).** The portion of a program with which a user interacts.

Copyright

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2011 Microsoft Corporation. All rights reserved.

