Introduction to WPF Technology

Publish Date: Aug 01, 2012

Overview

Microsoft Windows Presentation Foundation (WPF) is a user interface (UI) framework for building Windows client applications with immersive and intuitive user experiences. It is a subset of the Microsoft .NET Framework and was first introduced by Microsoft as part of .NET 3.0. It combines the application UI, 2D graphics, 3D graphics, documents, and multimedia into a single framework to help developers create rich and interactive applications. Its vector-based rendering engine takes advantage of the hardware acceleration of modern graphic cards. This makes the UI faster, scalable, and resolution independent.

Table of Contents

1. Satisfying User Experience Expectations

WPF was created to help developers meet increasing expectations of the experience and usability of software applications. From the user's perspective, the UI experience is the application and providing a more intuitive user experience through a better interface can improve productivity, help create loyal customers, and increase sales. WPF provides developers with the tools needed to stand out among competitors and help test operators accomplish their tasks faster.

2. Evolution of UI Technology

The primary technologies behind Windows-based UIs, such as graphics device interface (GDI) and USER subsystems were introduced with Windows 1.0 over 25 years ago. Although better technologies were released over the years, such as OpenGL and DirectX, which made it easier to work with 2D and 3D graphics as seen in the computer-aided design (CAD), scientific visualization programs and games, these technologies were still rarely used in mainstream Windows applications. The reason is that good graphics hardware has only recently become more common and the post-GDI technologies were much more difficult to implement. Now that graphics hardware is less expensive and user expectations are endlessly mounting, WPF is the answer for developers seeking to create a modern user experience with tools that can exploit the capabilities of hardware while still being easy to use.

The predecessor to WPF is Windows Forms, but WPF differs fundamentally in that it builds on top of DirectX, a technology that originally focused on multimedia and game programming. This adds the ability to implement advanced visual behavior such as animated cartoons, videos, or immersive 3D environments as well as take advantage of hardware acceleration when it is available. As it is still a subset of the .NET Framework, the WPF programming experience should be similar to Windows Forms or ASP.NET; however, the programming model is closer to web development with the "code-behind" approach. It should also be noted that WPF was not created to replace Windows Forms for every application as there are still cases where Windows Forms are good enough. Rather, WPF will continue to grow in popularity as a modern UI design tool and must be considered for any application that stands to benefit from a richer user experience.

3. Separation of Design and Business Logic

WPF also fulfills the need for technology and tools that make it natural to separate the UI from the rest of the implementation as much as possible. The WPF system lets developers decouple visual behavior from the underlying program logic and provides a unified API and supporting tools for creating sophisticated UIs.

The biggest innovation in WPF is XAML, a new language and new file type that provides a declarative model for application programming. With XAML, developers can create a workflow where separate parties can work on the UI and the logic of an application, using potentially different tools. This is discussed in more detail in the next tutorial, Benefits of WPF.

4. Getting Familiar With XAML

XAML stands for eXtensible Application Markup Language—pronounced "zammel." With this new markup, UIs can be defined without the need to program, very similar to creating an HTML web page. Like Windows Forms, you build WPF forms using the interactive designer to drag and drop items on the UI and customize in the properties box. But unlike Windows Forms applications where the designer generates code in C#, VB.NET to create controls on the form, in a WPF application, the interactive designer generates a XAML script. The XAML code specifies the control objects and their properties that make up the UI. When you run the program, the XAML compiler converts the XAML into instances of objects using the .NET Framework. This is unlike most other markup languages, which are typically an interpreted language without such a direct tie to a backing type system.

5. Summary

The popularization of smart consumer electronics like mobile devices and media consoles has had two interesting effects on the development of software not typically considered as mainstream. The perceived standard of a good user experience has increased and continues to grow based on the user's interaction with highly sophisticated consumer products.

The first outcome is that traditional software products like financial applications, human resources software, and engineering applications must face higher expectations from their users and invest resources to improve usability and design of software interfaces to stay competitive.

The second effect is on the tools for software interface development. The advent of WPF as Microsoft's premier technology for programming visually impressive Windows client applications is the result of demand from developers for better and faster methods to designing the user experience.

As long as software applications for personal use continue to improve more rapidly, other industries must follow suit to rethink and redesign the face of their software.

6. Additional Resources

Next Tutorial: Benefits of WPF Technology

Back to **Windows Presentation Foundation Home Page**

Learn about **NI Measurement Studio** Tools for .NET Developers