

Reykjavik University
New Technology,
T-611-NYTI, 2006-1
Teacher: Ólafur Andri Ragnarsson



Windows Presentation Foundation (WPF)

Guðmundur Bjargmundsson
gummudu03@ru.is

Contents

Introduction.....	2
Background.....	3
WPF explained.....	5
Brief description.....	5
Design goals.....	6
The engine.....	6
The architecture.....	7
Deployment.....	7
Interoperability.....	7
Graphical Services.....	7
Media Services.....	8
Imaging.....	8
Text Services.....	8
Documents.....	9
Annotations.....	9
User Interface.....	9
Data Binding.....	9
Input.....	10
XAML.....	10
WPF/E.....	10
Use of WPF.....	11
Final remarks.....	12
References.....	13

Introduction

Graphics is becoming more and more important every year and with constantly improving displays and monitors being produced the users and consumers are demanding a richer user experience with both work and leisure in mind. Growth of gaming and media industries fuels the demand.

People have come to realize that their time is precious and they want to use every minute for something they enjoy. More people, everyday, are using the internet for leisure. So called mini-games and/or coffee break games are becoming ever more popular especially for how easily accessible they usually are. Just browse to a web page, wait a couple of seconds, no install, no fuss, usually no problems, and the wasting of time can begin.

The problem is that it has always been difficult to deliver a rich client experience in a web browser. There are indeed many ways to do so, for example ActiveX, Java Applets, DHTML, java-script and most recently AJAX based technologies. But none of these have really caught on for games and rich media for all sorts of reasons. Probably the most successful way of delivering rich content today, and was not mentioned before, is via. Flash Player by Macromedia and it is at this time the de facto standard for internet games.

Flash Player has enjoyed enormous success and is so widespread that it is installed on over 90% of machines. The reason for this success is probably because Flash Player uses vector graphics which are scaleable and have only a small data footprint, also the player has been free of charge and much content has been available from the get go. Someone could describe that as the recipe for success.

But in spite of all it's success the Flash Player has it's shortcomings, especially when it comes to tying Flash applications to remote systems or databases, and there are many new projects aiming to bring more rich content to the web browser.

One of which is the topic of this paper. Some believe this new technology will eventually go head to head in battle with the Flash Player. This is Windows Presentation Foundation, or WPF.

WPF is the new graphics subsystem from Microsoft. It is based on vector graphics and it might turn out to prove rumors right as the Flash killer.

This paper aims to introduce WPF to the reader, what it entails, why Microsoft made it and what it has to offer.

Background

In 2001 Microsoft started developing the next Windows operating system, then codenamed Longhorn. Originally Longhorn was only supposed to be as a minor step from Windows XP with a release in 2003, but somewhere along the way it was decided that Longhorn should become something larger. Following through with their decision Microsoft started Longhorn development afresh by redesigning and rewriting the entire system. Longhorn got built upon on the Windows Server 2003 codebase to further meet the users demands and as a strong base for future needs. On July 22nd 2005 Microsoft dumped the Longhorn codename and announced the Vista name.

Part of the process, of redesigning and rewriting a new Windows OS, called for a re-evaluation on the OS application programming interface. Microsoft decided to write a new API to supersede the then current Windows API (Win32). Microsoft saw a golden opportunity to base the new API upon the .NET framework and thus utilize all of the frameworks good qualities like the code being managed and garbage collected and such.

The programming interface is called WinFX and it was originally only intended for Vista, but later on it was decided to make it available for Windows XP SP2 and Windows Server 2003 SP1. WinFX and Vista are backwards compatible and support Win32 code.

The Windows API will still be present in Vista, but it will not give direct access to all the new functionality introduced with WinFX. In addition, WinFX is intended to give .NET programmers easier access to the functionality present in Windows itself.[8]

WinFX consists of three fundamental frameworks:

- *Windows Presentation Foundation* (WPF), formerly code-named *Avalon*.
- *Windows Communication Foundation* (WCF), formerly code-named *Indigo*.
- *Windows Workflow Foundation* (WF).[8]

WPF is the subsystem of WinFX which will be the topic of this paper. WPF is the graphical subsystem and provides various services for applications like deployment, accessibility, input, media, audio, video, graphics, documents, text and more.

The main reason for Microsoft to develop WPF is that the tried and trusted Graphics Design Interface, or GDI, had become well outdated after serving it's purpose well since Windows 1.0. When GDI was designed the requirements were very different from what is required now, twenty years later. Back then there were no specialized graphics cards or processors in commercial distribution and everything was oriented around saving space and being simple. Nothing compared to todays situation where nearly all computers have specialized graphics processing hardware and rich user interface demands.

Surely GDI has been much improved over the years, and even with the addition of GDI+ from .NET framework which does improve it quite some bit, it does hardly utilize the graphics accelerating capabilities of todays hardware. Surely developers could utilize the hardware via DirectX ever since it came out, but it's API was not specifically design for user applications but rather games and has almost exclusively

been used for such, with few exceptions and mostly scientific and simulation programs.

So it was with some cause that Microsoft decided to start developing Windows Vista and that decision involved redesigning the underlying graphics and multimedia engine. The company imbued their new API with aspects that hadn't been in scope for earlier versions and special attention was given to likely future development. The WPF API are also tailored to suit the needs of the most common form of development process for graphical interfaces where programmers have to re-design and re-implement ideas from graphics designers so they fit the underlying technology used to create the final version of the interface. This is redundant work which, fairly often, ends with the designers vision getting blurred and skewed.

Following the success of markup languages for web development, WPF introduces a new language known as eXtensible Application Markup Language (XAML), which is a variant of XML. It is meant to be a faster way to develop application user interfaces and a better way for designers and developers to collaborate. Using XAML to develop user interfaces also allows for separation of model and view; this is generally considered a good architectural principle.[1]

WPF explained

The following text explains the infrastructure and architecture of Windows Presentation Framework without getting into detailed specifics and programming related matters. It is a top view general explanation that any IT person should be able to follow.

Brief description

The Microsoft Windows Presentation Foundation provides an integrated platform for building highly immersive, visually differentiated applications.[3]

The Windows Presentation Foundation (or WPF), formerly code named Avalon, is the graphical subsystem feature of Microsoft Windows. It will be included in Vista, the next version of the Microsoft Windows operating system. WPF is also available for installation on Windows XP SP2 and Windows Server 2003 SP1 as part of WinFX Runtime Components, a managed-code programming suite that extends the Microsoft .NET Framework. It provides a consistent programming model for building applications, whether they are installed on a system or are loaded into a web browser. It also enables richer control, design, and development of the visual aspects of Windows programs. It aims to unify a host of application services: user interface, 2D and 3D drawing, fixed and adaptive documents, vector graphics, raster graphics, animation, data binding, audio and video.[1]

As the strategic presentation foundation for future releases of Windows, it provides a model which takes full advantage of the underlying graphical hardware and offers support for everything from standalone applications to Web browser-based applications to documents.[2]

Built on the .NET Framework foundation and utilizing Direct3D for vector-based rendering, it provides a powerful solution for building immersive applications of all kinds. Furthermore, Windows Presentation Foundation enables designers to be an integral part of the development process by providing declarative programming models for "toolability" and flexibility.[2]

All rendering in Windows Presentation Foundation takes place through Direct3D. By moving more graphics processing to the GPU, the CPU is released for other work, increasing performance while simultaneously improving graphics quality. These benefits aren't limited to the 3D world, however - 2D graphics also take advantage of the same services; GDI is not used within Windows Presentation Foundation except for integration with legacy elements.[2]

Windows Presentation Foundation Services:

- *Base Services* - XAML, Property System, Input and Eventing, Accessibility.
- *Media Services* - 2D, 3D, Audio, Video, Text, Imaging, Animation, Effects, Composition Engine.
- *Document Services* - XPS Documents, Open Packaging Conventions.
- *User Interface Services* - Application Services, Deployment, Controls, Layout, Data Binding.[2]

By providing a single framework based on managed code for all these services, the Windows Presentation Foundation makes it possible to build new rich user experiences that were hitherto difficult or impossible. For .NET developers, its framework will be familiar, and it will ultimately reduce the number of lines of code required to build data-bound applications. In addition, the Windows Presentation Foundation introduces new and enhanced services such as animation, while retaining interoperability with existing code written for GDI/GDI+.

One of the overriding objectives in building the Windows Presentation Foundation has been *integration*. Services such as animation and data binding are used in exactly the same way whether the target is 2-D or 3-D graphics, user interface elements such as buttons and text boxes, or even media.

Windows Presentation Foundation applications are rendered using a new vector-based composition engine. Hardware acceleration is used for the rendering process on DirectX graphics cards, with a fallback software solution for older display hardware. The coordinate system provides double precision, resolution-independent pixel addressing that enables support for the high-dpi displays that are becoming increasingly prevalent.

The Windows Presentation Foundation also introduces a new declarative programming model code-named "XAML," which enables user interfaces to be specified as a hierarchy of objects with properties and logic. XAML neatly separates user interface design from code, enabling graphical designers to create compelling, highly refined user interfaces and developers to focus on the application logic. This collaborative model of application development allows developers and designers to work closely and efficiently together. In addition, XAML is easy for tools to create and consume, and can be compiled into an application alongside classes written in any CLS-compliant language such as Visual Basic and C#. [3]

Design goals

The design principles behind Windows Presentation Foundation can be categorized as follows:

- *Integration*: Offers a unified API that spans many services.
- *Vector graphics*. At its heart, the composition engine is vector-based, allowing for scaling, and takes full advantage of the GPU.
- *Declarative programming*. Allows for separation of model and view, so UI can be designed and edited in XML based markup called XAML.
- *Easy deployment*. With support for both standalone applications and Web-browser applications, Windows Presentation Foundation offers the best of both deployment models.
- *Document lifecycle*. Support for a new set of document and print technologies. [2]

The engine

Windows Presentation Foundation takes full advantage of the powerful Graphical Processing Units that are part of modern PC systems. At its heart, the composition engine is vector-based, allowing for scaling of all output to match the resolution of a specific machine. The rendering architecture uses Direct3D for all output: on video cards that implement DirectX 7 or later in hardware, Windows Presentation Foundation renders output using the GPU wherever possible. In situations where hardware rendering cannot be used, software rendering is available as a fallback. Lastly, a floating-point logical pixel system and 32-bit ARGB color support provide a

rich high-fidelity experience that anticipates future technology needs, such as high-DPI displays.[2]

Most elements in Windows Presentation Foundation derive from a base Visual class. The visual system composites all the data and produces the output onto the screen. This is where we integrate video and audio, 2D, 3D and animation together, as well as deliver certain text layout services. Lower in the architectural model sits the composition engine, which has the responsibility of rendering a visual tree to the screen, taking account of everything from transparent layers to window region invalidation.[2]

The architecture

Developers today are faced with a myriad choice of disparate technologies and APIs, depending on whether they are targeting 2D graphics (GDI or GDI+), user interface (USER32 or Windows Forms), media (DirectShow), or 3D (Direct3D or OpenGL). Windows Presentation Foundation provides a single model that is orthogonal across all these services and allows seamless integration of content within a single application. You can use the same constructs for animation, data binding and styling, regardless of whether you are targeting 2D, 3D or text content.[2]

Deployment

WPF supports both standalone applications and browser based applications.

Web-browser applications run from within Internet Explorer, either occupying the entire window or within an inline frame. They offer the ease of deployment for which Web applications are famed, as well as operating within a partial trust sandbox that protects the client machine against malicious applications. Yet they can still take advantage of the local client hardware and use 3D and media services for the richest Web experience available today. On the other hand, standalone applications are locally installed via ClickOnce or MSI technologies and offer full access to the underlying platform.[2]

Interoperability

Most developers who see Windows Presentation Foundation will be concerned about the potential impact on their existing applications. Will they have to rewrite everything, or is there a better story? What about plug-ins and controls? Do they all have to be rewritten too?

The good news is that Windows Presentation Foundation provides good interoperability: you can use Windows Presentation Foundation inside existing code, or you can use existing code inside Windows Presentation Foundation. For the maximum richness, however, it's true that a 100% Windows Presentation Foundation experience is going to offer benefits that will be missing from a mixed approach, for example data binding, animation and layout services across all elements.[2]

Graphical Services

As highlighted in the introduction, the current Windows graphics platform (GDI) has a heritage spanning nearly two decades. Over the last few years huge innovation in the 3D space, but GDI developers have seen little benefit from this. Graphics hardware advances today are mainly focused on the 3D pipeline, and Windows Presentation

Foundation is designed as a full strategic replacement for older 2D-based technologies, using the graphics capabilities of modern PCs to their fullest. All the different elements of graphics go through the same pipeline: 2D, 3D, text, imaging and video.

Windows Presentation Foundation provides mainstream graphics services for applications and content. It provides an application model that takes advantage of the 3D hardware support prevalent in modern graphics cards. Windows Presentation Foundation places a greater emphasis on vector-based content; it offers resolution independence, with virtual pixels that are mapped to logical pixels and support high DPI screens; the native co-ordinate system is based on a double data type. A side effect of the new rendering model is that image transformations can be applied without additional work (for example, it's possible to apply a blur effect or a scaling factor to any content). Windows Presentation Foundation is also an integrated component of the broader graphics ecosystem: for example, it takes advantage of the new Windows Vista display driver model, it shares capabilities with the print model, and it has high fidelity remoting via Terminal Services.[2]

Media Services

Windows Presentation Foundation provides shape primitives for 2D graphics along with a built-in set of brushes, pens, geometries and transforms.

The 3D capabilities in WPF are limited compared to what's available in Direct3D. However, WPF provides tighter integration with other features like user interface (UI), documents and media. This makes it possible to have 3D UI, 3D documents and 3D media.

There is support for most common image formats. In addition, WPF supports the WMV, MPEG and AVI media formats.

WPF supports time-based animations, in contrast to the frame-based approach. This delinks the speed of the animation from how slow or fast the system is performing. Text rendering is supported using ClearType. This provides for Sub-pixel positioning, natural advance widths and Y-direction anti-aliasing. WPF also supports OpenType font features.[1]

Imaging

Windows Presentation Foundation provides support for most common image formats via a library of codecs.[2]

Windows Presentation Foundation has rich and extensible support for image metadata, with in-place updates. There are also straightforward image query capabilities.[2]

Text Services

Windows Presentation Foundation uses sub-pixel positioned ClearType, which allows the full precision of ClearType to be displayed on the screen.[2]

This is the first Windows programming interface to expose OpenType features to software developers, supporting both OpenType TrueType and OpenType CFF fonts.[1]

All of the text rendering in Windows Presentation Foundation can be hardware-accelerated given a suitable card, using composites, filters, and blends to offload work

that would otherwise be performed on the CPU. The enhanced text support in Windows Presentation Foundation is particularly noticeable for users of East Asian fonts, where historically ClearType has been missing and fonts haven't taken full advantage of the platform.[2]

Documents

Windows Presentation Foundation supports Windows Vista two new presentation technology specifications for handling the document and print lifecycle:

- *Open Packaging Conventions*; describes a series of file format packaging conventions and associated services that can be used by any application. The specification for Open Packaging Conventions is based on a ZIP file, over which there is a parts/relationships abstraction and then services that allow you to access the ZIP file.
- *XML Paper Specification*; describes the markup and rendering rules for a fixed, paginated document that uses the Open Packaging Conventions. The XML Paper Specification also includes a specific fixed format implementation called *XPS Document*.[2]

Annotations

With just a few lines of code, developers can enable support for adding and delete annotations to documents. Two types of annotations are supported in Windows Presentation Foundation: highlighting and notes.

Highlighting works just like a highlighter pen for paper: you can select a range of content to be set with a different color. On the other hand, note annotations allow you to add ink or text content to a piece of content, for example, to add a comment.[2]

Windows Presentation Foundation introduces a new set of document and print technologies. Applications that need to persist data to a local store can use the *Open Packaging Conventions*, a ZIP-based packaging convention shared with Office 12 that supports core properties and custom metadata, digital signatures and rights management functionality. For applications that want to share documents for collaboration across multiple machines, even without the application installed, the *XML Paper Specification* allows visuals to be fixed in a printable, portable format.[2]

User Interface

A set of built-in controls is provided as part of WPF, containing items such as button, menu, and list box.

A powerful concept in WPF is the ability to perform control/content composition, where a control can contain any other control or layout.[1]

Data Binding

Data has been a first-class citizen since the start of Windows Presentation Foundation; there is support for data in the property engine as well as declaratively from XAML.[2]

WPF has a built-in set of data services to enable application developers to bind and manipulate data within applications. There exists support for three types of data binding:

- one time: where the client ignores updates on the server

- one way: where the client has read-only access to data
- two way: where client can read from and write data to the server

Binding of data has no bearing on its presentation. WPF provides data templates to control presentation of data.[1]

Input

Windows Presentation Foundation gives full access to the mouse, keyboard, stylus, and speech while providing higher-level services for text input and commands. The WPF framework makes it easy for developers to define new input devices and methods and supplies an easy to understand Command pattern functionality to build upon.

XAML

Windows Presentation Foundation introduces XAML (eXtensible Application Markup Language), an XML-based language for instantiating and populating nested object hierarchies. While XAML isn't exclusively tied to Windows Presentation Foundation, it is inherently suitable for tasks such as UI definition and construction. The design of XAML allows applications to parse and manipulate UI logic at run-time for dynamic workflow scenarios. Importantly, the XAML / code-behind model embodied in Windows Presentation Foundation allows designers and developers to work collaboratively on client application design and development, using tools such as Expression "Sparkle" as well as third-party specialist tools including ZAM 3D and Mobiform Aurora.[2]

WPF/E

WPF/E is a subset of WPF, and stands for "Windows Presentation Foundation Everywhere". It is basically a mobile version of WPF, based on XAML and Javascript. 3D features are not included, but XPS, vector-based drawing, and hardware acceleration, are.[1]

Windows Presentation Foundation/Everywhere is an extension to WPF to provide a subset of WPF features, such as hardware accelerated video, vector graphics, and animations to platforms other than Windows Vista. Specifically, WPF/E will be provided as a plug-in for Windows XP, Mozilla Firefox, Apple Safari and mobile devices.

These extensions will allow the browsers and other applications to use WPF/E graphical capabilities. The browser extensions will be in the line of Adobe Flash, a highly popular graphic plug-in available for most browsers. Internet Explorer will have native support for WPF in Windows Vista, and will support WPF/E in older versions.[1]

It is rumoured that Microsoft will deliver WPF/E along with Internet Explorer 7 and thus could quickly gain much distribution.

Use of WPF

There's a clear role for a client application model that offers standalone and browser-based applications, ease of deployment, great tools support, and deep platform integration.[1]

That is what WPF is for. It is used to create rich client applications, standalone and browser-based, that further enhance the user experience than before. The user experience is enhanced by features like easier deployment through web browsers.

Utilizes users investment in hardware better so they get more for their money.

Easier development by having all these different elements accessible from one framework and easy to use interface.

Eases collaboration between designers and developers by having the framework markup based, therefore allowing separation of model and view and also custom tools which need only emit the markup which can in turn be interpreted by the framework.

Final remarks

The lone fact, that WPF will be a fundamental component in Windows Vista, might be enough to conclude that it will have a major impact on the development of Graphical User Interfaces of the future. Making WPF able to run on older Windows platforms along with the WPF/E version might be enough to secure it as a new standard for developing rich content and end user experiences. To sweeten the deal Microsoft has already introduced WPF internally and all elements of Windows Vista along with every component of the highly anticipated Office 12 Suite will use WPF as their graphical foundation. Microsoft is really swinging to back up this technology and they are, without a doubt, a company that can prove that “Money Talks”.

XAML and WPF/E will play a major role in the reception of WPF, along with Microsoft Expression, the Graphics Development Suite that is aimed at bridging the gap between software developers and graphics designers. If XAML lives up to what it is supposed to deliver by solidifying and simplifying the connection between graphics and code it will prove to be a major benefit. Another big issue is whether Expression can penetrate into the Adobe ruled graphics development market it will guarantee a smoother, more pleasant, reception for WPF. Its outbreak will happen that much more quickly.

Future use for WPF might not only reach business applications but also games, rich advertisement content and other applications that would want to expand into rich media. WPF might even prove to be a catalyst for rich media taking over as a standard in web presentation, releasing the much-out-of-date HTML from its duty as king and queen of displaying web content.

In the future it is likely that Microsoft will introduce new versions of the WinFX API, along with updates and new versions of WPF. Today it seems this technology has full backing from Microsoft. GDI has lived for nearly two decades. It isn't improbable that WPF will live for, at least, as long.

References

- [1] **Windows Presentation Foundation**. Wikipedia. Retrieved March 28, 2006, from Wikipedia. <http://en.wikipedia.org/wiki/Windows_Presentation_Foundation>
- [2] **A Guided Tour of Windows Presentation Foundation**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/wpf101.asp>>
- [3] **Architectural Overview of the Windows Presentation Foundation Beta 1 Release**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/hgtobeta1.asp>>
- [4] **An Introduction to Windows Presentation Foundation**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/introwpf.asp>>
- [5] **Introducing the New Avalon Graphics Model**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/avalongraphics.asp>>
- [6] **Timing Is Everything**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnavalon/html/avalon01062004.asp>>
- [7] **The Blinking Lights Division**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnavalon/html/avalon03262004.asp>>
- [8] **WinFX**. Wikipedia. Retrieved March 28, 2006, from Wikipedia. <<http://en.wikipedia.org/wiki/WinFX>>
- [9] **The "Avalon" Input System**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/avaloninput.asp>>
- [10] **Windows Presentation Foundation**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/winfx/reference/presentation/default.aspx>>
- [11] **Windows Presentation Foundation on the Web: Web Browser Applications**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/wpfandwbas.asp>>
- [12] **Windows Presentation Foundation Security Sandbox**. MSDN Library. Retrieved March 28, 2006, from MSDN Library online. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/wpfsecuritysandbox.asp>>
- [13] **More than meets the eye in Microsoft plan**. News.com. Retrieved March 28, 2006, from News.com online. <http://news.com.com/More+than+meets+the+eye+in+Microsoft+plan/2100-1012_3-6053893.html?tag=carsl>