

DirectX Developers - DirectX in Windows Phone

This article provides basic information about native C++ and DirectX game development on Windows Phone 8. It will list the major differences of DirectX in Windows 8 vs Windows Phone 8 and also give suggestions on how to do certain functions in an alternative way because of them not being available in both platforms.

Introduction



With Windows Phone 8 game developers will be able to use native C++ and DirectX APIs to create their games. However, Microsoft has made some changes to the DirectX features on Windows Phone compared to the desktop version to better optimize it for a mobile environment. We will be taking a look at the general differences.

Prerequisites

This article is meant for developers with some prior knowledge about DirectX. Basic knowledge of game development programming is also recommended.

DirectX in Windows Phone 8

Direct3D

DirectX in Windows Phone 8 uses version 11 of Direct3D with feature level set to 9_3 (rather than 11_1 that Windows 8 uses). This naturally means that features higher than 9_3 aren't supported. We have put together a comparison table of differences in feature levels 11_1 and 9_3 using [the MSDN page on Direct3D Feature Levels](#) as a reference.

Feature	Feature level 11_1	Feature level 9_3
Shader Model	5.0	2.0
Geometry Shader	Yes	No
Stream Out	Yes	No
DirectCompute/ComputerShader	Yes	N/A
Hull and Domain Shaders	Yes	No
Texture Resource Arrays	Yes	No
Cubemap Resource Arrays	Yes	No
BC4/BC5 Compression	Yes	No
BC6H/BC7 Compression	Yes	No
Alpha-to-coverage	Yes	No
10-bit XR High Color Format	Yes	N/A
Logic Operations(Output Merger)	Yes	No
Target-independent rasterization	Yes	No
Multiple render Target(MRT) with ForcedSampleCount 1	Yes	No
UAV slots	64	N/A
Max Texture Dimension	16384	4096
Max Cubemap Dimension	16384	4096
Max Volume Extent	2048	256
Max Texture Repeat	16384	8192
Max Primitive Count	2 ³² -1	1048575
Max Input Slots	32	16
Simultaneous Render Targets	8	4

From these we can see that the feature level 9_3 has quite a lot less features than the 11_1, but this also ensures that Direct3D applications run smoothly on mobile hardware.

Some Direct3D APIs that are part of the desktop environment are **not supported in WP8**. These APIs are:

- ID3D11CommandList
- ID3D11ComputeShader
- ID3D11DomainShader
- ID3D11GeometryShader
- ID3D11HullShader
- ID3D11UnorderedAccessView

In addition to these, Windows Phone 8 specific feature changes have been made to the Swap Chain. In practice this means some of the features that are available in desktop Direct3D are not supported or are stripped down on WP8. Here is a list of WP8 specific changes:

- No multisampling is allowed.
- Only single buffer can be used.
- Scaling must use stretching option.
- SwapEffect must be set to discard.

Runtime Shader Compilation

Runtime Shader Compilation in Windows Phone 8 Direct3D is disabled and only offline compilation is possible due to mobile hardware limitations.

Other DirectX Features

Other than Direct3D, some other central DX features in WP8 have been disabled. The disabled features are:

- **Direct2D:** This means the developer will need to use Direct3D even if he/she wants to make a 2D game. The DirectX Toolkit is a good alternative to use in WP8. It contains the SpriteBatch-class from XNA which is good for 2D graphics rendering.
- **DirectWrite:** As an alternative for text rendering, we suggest using the SpriteFont-class from DirectX Toolkit.
- **XInput:** This API is used to get the Xbox Controller input, so it being removed isn't a big matter when the developer is making a mobile-only game application.
- **Windows Imaging Component (WIC):** WIC being removed means that loading textures from image files such as JPGs and PNGs is more difficult and using DDS-texture images is recommended. The DirectX Toolkit also has a helper function to make DDS-texture loading easier.
- **DirectAudio:** Not available in WP8. The developer can use XAudio2 or WASAPI APIs instead.

Development Environment Requirements

When developing games for Windows Phone 8, all the requirements for normal WP8 application development apply. The developer will need a Windows 8 PC with Visual Studio 2012. The Windows Phone 8 SDK comes with Visual Studio 2012 Express.

For the developer to be able to use the Emulator to deploy applications on, the development PC will need to meet certain hardware requirements. A graphics card with DirectX 11 support is needed. It is required that the machine also has a HyperV and SLAT supported CPU (Intel's i3, i5, i7 -series) and at least 4GB of RAM. This enables the developer to choose from either developing with the emulator or to deploy the developed application to a developer unlocked device, or both. An Windows Phone Dev Center account is needed to unlock a device for development.

If the emulator can't be run on the developer's machine, the application can also be solely deployed and debugged on a device with Windows Phone 8 connected to the development PC via USB.

Fastest Way to Get Started: the Visual Studio Template

Visual Studio helps the developer to get started on a WP8 Direct3D application with the "**Windows Phone Direct3D App**" – **project template**. This template has some of the application functionality such as D3D device and touch input already implemented. The template also has D3D object rendering implemented for a rotating 3D cube. This saves the developer from setting everything up by himself and helps him to get in the middle of things better.

Further Reading

To further get into C++/DirectX development on Windows Phone you can check out the articles [Windows Phone Native C++ and DirectX - First Direct3D App, setting up Touch and Sensors](#) and [DirectX on Windows Phone: 2D Game Example using DirectX](#)

