OIT11

<https://github.com/walbourn/directx-sdk-samples>

This is the DirectX SDK's Direct3D 11 sample updated to use Visual Studio 2012 and the Windows SDK 8.0 without any dependencies on legacy DirectX SDK content. This sample is a Win32 desktop DirectX 11.0 application for Windows 10, Windows 8.1, Windows 8, Windows 7, and Windows Vista Service Pack 2 with the DirectX 11.0 runtime.

**This is based on the legacy DirectX SDK (June 2010) Win32 desktop sample. This is not intended for use with Windows Store apps, Windows RT, or universal Windows apps.**

# Description



Demonstrates a technique for achieving order-independent transparency in Direct3D 11.

## Overview

This sample, demonstrates a technique for achieving order-independent transparency in Direct3D 11. In other words, primitives can be drawn in any order, and transparency will be handled correctly. The technique works by buffering transparent pixels as they are rendered. These pixels are then sorted by depth, and rendered in back-to-front order with transparency. This sample uses compute shaders to generate a prefix sum and to sort the pixel fragments before the final render pass.

The algorithm executes a series of passes:

1. Determine the number of transparent fragments in each pixel by drawing each of the transparent primitives into an overdraw accumulation buffer.
2. Create a prefix sum for each pixel location. This holds the sum of all the fragments in each of the preceding pixels. The last pixel will hold a count of all fragments in the scene.
3. Render the fragments to a deep frame buffer that holds both depth and color for each of the fragments. The prefix sum buffer is used to determine the placement of each fragment in the deep buffer.
4. Sort the fragments and render to the final frame buffer. The prefix sum is used to locate fragments in the deep frame buffer.

# Dependencies

DXUT-based samples typically make use of runtime HLSL compilation. Build-time compilation is recommended for all production Direct3D applications, but for experimentation and samples development runtime HLSL compilation is preferred. Therefore, the D3DCompile\*.DLL must be available in the search path when these programs are executed.

* When using the Windows 8.x SDK and targeting Windows Vista or later, you can include the D3DCompile\_46 or D3DCompile\_47 DLL side-by-side with your application copying the file from the REDIST folder.

%ProgramFiles(x86)%\Windows kits\8.0\Redist\D3D\arm, x86 or x64

%ProgramFiles(x86)%\Windows kits\8.1\Redist\D3D\arm, x86 or x64

%ProgramFiles(x86)%\Windows kits\10\Redist\D3D\arm, x86 or x64

# More Information

[Where is the DirectX SDK (2015 Edition)?](https://walbourn.github.io/where-is-the-directx-sdk-2015-edition/)

[DXUT for Win32 Desktop Update](https://walbourn.github.io/dxut-for-win32-desktop-update/)

[Games for Windows and DirectX SDK blog](https://walbourn.github.io/)