Rendering with DirectX 12

[Proposal Document On Git](https://github.com/GH-DS/project-neortls007idev/tree/Setup/Docs)

# Introduction

Games are often judged for their high-fidelity graphics and getting high fidelity graphics to render on screen with a standard 60 FPS is no easy task. DirectX 12 allows using hardware at a deeper level and giving access to features such as Mesh Shading, Variable Rate Shading and Real Time Ray Tracing. The hardware technology for raytracing has progress by leaps and bounds over the last few years and setting up DirectX 12 renderer and the DirectX Raytracing is one of the only few options that allows utilization of this new hardware for Real time Raytracing while being able to meet the standard requirements.

# Detailed Feature Description

## MIDTERM – No Lighting 2D And 3D Rendering Working.

### Clearing the screen - *As a developer, I want to be able to set the color and see the single color across the entire window.[2]*

### Hello Triangle *- As a developer, I want to be able to see a hardcoded shader triangle rendered onto the screen.[3]*

### Unlit Rendering – *As a developer, I want to able to see 2D and 3D game rendering across the window without any lighting.[1]*

## FINAL TERM – Lighting Works

### Rendering with Lighting *– As a developer, I want to see a 3D tech demo rendering with lighting.[1]*

### Raytracing Pipeline *– As a developer, I want the raytracing pipeline setup complete.* [1][2]

## STRETCH GOALS

### Real Time Raytracing – *As a user, I want to see a demonstration of real time raytracing with a few different objects in the scene.[2]*

# Technical Issues

|  |  |  |
| --- | --- | --- |
| 1.1.1 | Requires setting up render targets, Descriptor Heaps, Command allocators and Lists, Fences and Events | Medium |
| 1.1.2 | Requires setting up vertex and pixel shader stages, samplers | Medium |
| 1.1.3 | Requires setting up buffers, Resource trackers, Descriptor allocators, Dynamic Descriptor Heaps | High |
| 1.2.1 | Have not worked with deferred lighting yet so I am not sure | High |
| 1.2.2 | Requires dealing with accelerated structures, more research needed | High |
| 1.3.1 | Not sure how I will be demonstrating it, possibly with cube and spheres or gem models | High |

# Performance

* If performance issues are encountered –
  + Finding hotspot using a profiler

# Testing

* How will you test your project to prove correctness of the implementation?
* Which features can be tested using automation?
  + DirectX objects creation and destruction
* Which features must be tested by hand?
  + The correctness of the graphics
  + Changing Shader code for debugging purposes
* Is any special software needed?
  + Nvidia N-Sight Profiler
  + Intel V-tune Profiler
  + Tracy Profiler

# Bibliography

|  |  |
| --- | --- |
| [1] | M.-K. L. a. P. Gautron, "DX12 Raytracing tutorial - Part 1," Nvidia Developer, [Online]. Available: https://developer.nvidia.com/rtx/raytracing/dxr/DX12-Raytracing-tutorial-Part-1. [Accessed 2 February 2021]. |
| [2] | M.-K. L. a. P. Gautron, "DX12 Raytracing tutorial - Part 2," Nvidia Developer, [Online]. Available: https://developer.nvidia.com/rtx/raytracing/dxr/DX12-Raytracing-tutorial-Part-2. [Accessed 2 February 2021]. |
| [3] | Jeremiah, "Learning DirectX 12 – Lesson 1 – Initialize DirectX 12," 3D Game Engine Programming, 14 December 2017. [Online]. Available: https://www.3dgep.com/learning-directx-12-1/. [Accessed 31 January 2021]. |
| [4] | Microsoft, "Direct3D 12 programming guide," Microsoft, 19 April 2019. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/direct3d12/directx-12-programming-guide. [Accessed 31 January 2021]. |
| [5] | F. Luna, Introduction to 3D Game Programming with DirectX 12, Mercury Learning & information, 2016. |