Direct3D 12 파이프라인 (part2)

Primitive

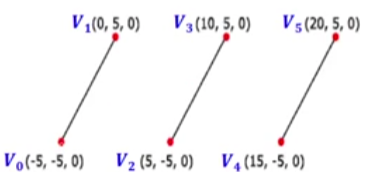
IASetPrimitiveTopology 라는 함수로 선언이 가능하다.

파이프라인이 정점 버퍼 데이터를 해석하는 법을 나타낸다.

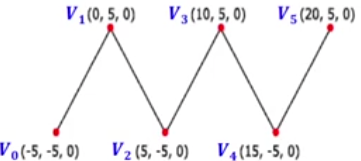
텍스트, 장치이(가) 표시된 사진

자동 생성된 설명\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D3D\_PRIMITIVE\_TOPOLOGY\_POINTLIST (점들의 리스트)

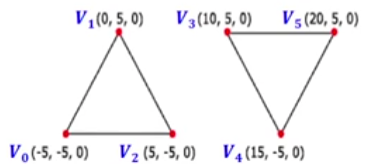
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D3D\_PRIMITIVE\_TOPOLOGY\_LINELIST (선분들의 리스트)

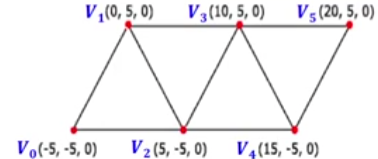
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D3D\_PRIMITIVE\_TOPOLOGY\_LINESTRIP (선분들의 스트립)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D3D\_PRIMITIVE\_TOPOLOGY\_TRIANGLELIST (삼각형의 리스트)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D3D\_PRIMITIVE\_TOPOLOGY\_TRIANGLESTRIP (삼각형의 스트립)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

그렇기 때문에 Primitive를 조립 할 수 있는 충분한 정점을 넣어주어야 한다.

Ex) TRINANGLELIST는 삼각형 n개라면 3\*n개의 정점

LINESTRIP은 선분n개라면 n+1개의 정점을 넣어주어야 한다.

Index Buffer

삼각형을 그리는데 1번(P1,P2,P3) 2번(P2,P4,P3), 3번(P3,P4,P5)를 그린다면

정점 버퍼로는 위와 같이 9개를 넣어주어야 한다.

그러나 인덱스 버퍼를 사용하면 정점 버퍼에는 P1,P2,P3,P4,P5만 넣어두고 인덱스 버퍼에는 (0,1,2),(1,3,2),(2,3,4)를 넣어주면 된다.

즉 여러 번 반복되어 드는 메모리와 시간을 단축시켜준다.

DrawIndexedInstanced(인덱스 개수, 인스턴스 개수, 시작 인덱스 위치, 각 정점 인덱스에 더해지는 값, 인스턴스 인덱스에 더해지는 값)으로 그려준다.

정점 버퍼 뷰와 인덱스 버퍼 뷰는 서술자와 서술자 힙이 필요없다.

IASetVertexBuffers(정점 버퍼 시작 슬롯, 정점 버퍼 개수, 정점 버퍼 주소)

인덱스 버퍼는 인덱스 버퍼 주소만 받는다. (한번에 하나씩만 설정 가능)

Root Signature

Root Parameters의 집합이다.

Root Parameters는 Shader들에게 전달되어야 하는 글로벌 변수이다.

1. Root Constant (루트 상수)

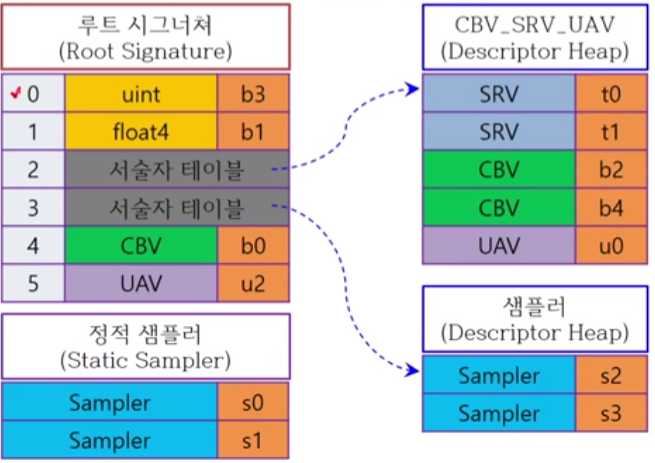
매개변수가 상수들을 직접 적어서 Shader변수에 값을 전달하며 힙이 필요없고 빠르다

1. Root Descriptor (루트 서술자)

매개 변수가 table에다가 Descriptor을 직접 포함하며 서술자 힙이 필요없다.

버퍼 리소스에 대한 SRV/UAV, 상수 버퍼에 대한 CBV이다.

1. Descriptor Table (서술자 테이블)

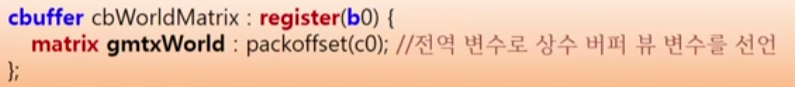
서술자 힙에 간접적으로 접근하므로 힙이 필요하고 시간이 걸린다.

상수 버퍼

여러 개의 Shader 상수들의 그룹을 동시에 갱신 가능한 특별한 버퍼다.

크기가 최소 하드웨어 할당 크기 256바이트의 배수여야 한다.

B는 상수 버퍼 u는 아놀드 액세스 버퍼 t는 Texture s는 Sampler를 의미한다.



상수 버퍼는 4096개의 벡터를 저장할 수 있으며 14개까지 생성 가능하다.

위는 0번째 상수 버퍼이며 gmtxWorld라는 이름의 행렬을 가지고 있다.

SetGraphicsRoot32BitConstants(루트 파라미터 인덱스, 전달할 수, 데이터 배열, offset)을 하면 쉐이더 변수의 값을 빠르게 전달할 수 있다.