

단순 인스턴싱 샘플 (DX12)

*이 샘플은 Windows Anniversary 업데이트(14393)와 호환됩니다.*

# 설명

이 샘플은 Direct3D 12 API로 인스턴싱을 사용하는 방법을 보여줍니다.

# 샘플 사용하기



|  |  |  |
| --- | --- | --- |
| 액션 | 게임 패드 | 키보드 |
| 카메라 회전 | 왼쪽 엄지 스틱 | WASD / 왼쪽 마우스 및 왼쪽 마우스 버튼 |
| 인스턴스 개수 변경 | LB / RB | Q / E |
| 센터 | 해당 없음 | 홈 |
| 시뮬레이션 재설정 | A | 스페이스 |
| 종료 | 보기 버튼 | Esc |

# 구현 정보

인스턴스 된 지오메트리를 렌더링 할 때 고려해야 할 4 가지 영역이 있습니다:

1. **기하 데이터**

이 샘플의 경우 큐브의 면을 설명하는 버텍스와 인덱스가 포함된 버텍스 및 인덱스 버퍼가 포함됩니다. 또한 *파이프 라인 상태 객체* 가 큐브를 세계 좌표에 렌더링하는 데 필요합니다. 이러한 모든 구성 요소는 인스턴스화되지 않은 형상과 동일한 방식으로 설정 및 조작됩니다. ( SimpleInstancing12.cpp의 *CreateDeviceDependentResources*를 참조하세요)

1. **인스턴스 데이터**

표준 D3D12 인스턴스 렌더링의 경우 인스턴스 데이터 당 하나 이상의 정점 버퍼를 통해 제공됩니다. 이 정점 버퍼는 다른 정점 버퍼와 같은 방식으로 생성됩니다. 이 샘플은 2 개의 정점 버퍼를 사용합니다. 하나는 정적이며 인스턴스별 색상 데이터 (샘플의 수명 동안 변경되지 않음)를 포함합니다. 다른 하나는 동적이며 인스턴스 별 위치 및 방향 정보를 포함합니다 (모든 프레임 변경).

1. **인스턴스 배치**

지오메트리를 인스턴스화하기 위해 D3D는 제공된 버텍스 데이터를 해석하는 방법에 대한 정보가 필요합니다. 이것은 표준 렌더링이 수행되는 것과 거의 같은 방식으로 *D3D12\_INPUT\_ELEMENT\_DESC* 구조의 배열을 사용하여 수행됩니다. 그러나 이 구조에는 추가 요소가 추가됩니다. 기하학적 데이터는 일반적으로 *InputSlotClass* 요소용 *D3D12\_INPUT\_CLASSIFICATION\_PER\_VERTEX\_DATA*값을 사용하지만 인스턴스 별 데이터는 *D3D12\_INPUT\_CLASSIFICATION\_PER\_INSTANCE\_DATA*값을 사용합니다. *InputSlot* 요소는 또한 각 데이터 조각이 추출되는 정점 스트림을 나타내는 데 사용됩니다.

버텍스 세이더는 버텍스 구조를 사용합니다 이 버텍스 구조는 기하학적 데이터와 인스턴스 별 데이터가 모두 덩어리로 처리된 것처럼 정의됩니다 (*D3D12\_INPUT\_ELEMENT\_DESC* 배열로 기술 된 레이아웃을 반영한다).

***알림:*** *이 샘플은* AlignedByteOffset *요소에 대한* D3D12\_APPEND\_ALIGNED\_ELEMENT *상수를 사용하여 입력 레이아웃에서 데이터를 자동으로 올바르게 정렬합니다. 문제의 버텍스 버퍼의 구조에 올바르게 정렬된 데이터가 있는 경우에만 작동합니다. 정점 데이터에서 요소를 건너 뛰거나 무시하는 경우 정확한 정렬 오프셋이 필요합니다.*

1. **렌더링**

이전 포인트를 고정해 놓으면 인스턴스 데이터를 렌더링하는 것은 간단합니다. *ID3D12GraphicsCommandList::IASetVertexBuffers* API는 입력으로 사용 된 버텍스 버퍼를 설정하는 데 사용됩니다 (이 경우 입력 버퍼). *ID3D12GraphicsCommandList::DrawIndexedInstanced*API는 렌더링에 사용됩니다. 나머지 렌더링 설정은 표준 비 인스턴스 렌더링과 동일하게 수행됩니다.

# 업데이트 기록

초기 출시 2018 년 4 월.