|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| 1주 | 2020. 12. 27~2021. 1. 2 | 작성자 | 김덕규 |
| 주간 목표 | 최현욱  - MMO서버와 MO서버의 차이점 공부  - 게임 서버 프로그래밍 강의 때 사용한 더미 클라이언트의 구조 분석  홍범도  **-** 주요 D3D인터페이스에 대한 정리  - 애니메이션에 대한 공부  김덕규  - 프레임워크 제작 시작  - 프레임워크 구조 설계 및 장치 초기화 | | |
| 이번주 한 일 | 최현욱  [12. 27]  제안서 수정 및 보완, 예상 질문 준비  [12. 28]  제안서 발표  [12. 29]  - 책과 강의자료를 통해 MMO서버와 MO서버의 차이점을 공부. 가장 큰 차이점은 클라이언트 간의 동기화 방식. 실시간 반응에 민감한 MO게임은 초당 30~40번 월드 상태를 클라이언트에게 전달하여 클라이언트간 동기화를 구현한다. 위 방식대로 진행하면 클라이언트는 항상 몇ms 전의 상태를 그리게 될 텐데, 이 부분은 상당히 문제가 된다고 판단. 해결 방법을 찾아야 할 듯.  [12. 30]  - 책을 찾아본 결과, 해당 문제를 해결하는 방법으로 ‘데드 레커닝’이 있음을 확인하였다. 두 책 모두 가속도 공식을 예시로 들며 설명하고 있는데, 무슨 관계인지 쉽게 이해가 안된다. 인터넷에서 예시를 찾아 봐야할 듯.  [1. 1]  - 데드 레커닝에 대한 예시를 인터넷 검색하여 찾아봄. 인터넷에서 많은 글들을 보니 이해가 되었다. 핵심은 클라이언트가 서버로부터 받은 정보는 몇ms과거의 상태이므로 그 시간과 서버로부터 받은 캐릭터의 속도를 통해 위치를 예측하는 것.  [1. 2]  - 강의 때 사용한 더미 클라이언트 구조 분석  홍범도  이번 주는 코드 진행보다는 내가 지금까지 배운 D3D인터페이스들에 대한 정리를 하고 다음주부터 본격적으로 애니메이션 적용에 대해 공부하는 시간을 가졌다.  [12. 27]  제안서 수정 및 보완, 예상 질문 준비  [12. 28]  제안서 발표  [12. 29]  1. InputLayout 정리 ( 정점의 성분별 작업에 대한 정리를 해서 D3D와 셰이더에게 알려주는 인터페이스 말해뭐해 할정도로 당연한 거임 )  2. CommandList 정리 ( Graphics, Compute용으로 사용할  CommandList, CommandQueue, ,CommandAllocator  에 대한 개념 정리함 )  [12. 30]  1. RootSignature 정리 ( Graphics, Compute용으로 사용할 RootSignature에 맞는 서술자 와 루트 상수들을 셰이더와의 연관관계에 대해 정리함 )  2. RenderTarget 정리 ( SwapChain과 이펙트와 관련된 셰이더기법의 사용시 RenderTarget을 담아 처리할 ID3D12Resource의 사용은 불가피함 그래서 이에 관한 개념 정리가 필요하다봄 )  [12. 31]  오늘부터 본격적인 Animation에 대해 공부  내가 혼자서 맡은 분야인 만큼 전문성이 필요하다 생각하여 깊고깊게 공부하기로 함  1. 키프레임 애니메이션  ( 구간 프레임에 애니메이션 키값을 등록하고 나머지 값은 자동 생성하는 것인데 여러가지 방법 중 일단 사원수를 이용한 선형 보간법을 생각해보기로 함 )  2. 계층구조 이해 ( 부모 - 자식 간의 구조를 가지는 애니메이션 에서  부모 중 최상위에 해당하는 행렬을 가장 나중에 곱하고 바로 인접한 부모 부터 곱하는 방식이라 애니메이션할때 사용할 구조인 것 같음 )  [1. 1]  계층구조 , 분할메쉬를 통한 계층적 애니메이션 구현 공부 ( 목표인 스키닝에 비해 관절 구조의 끊김이 있을 수 있으나 애니메이션에 가장 근본이 되는 애니메이션을 우선 구현해보는 것이 중요하다 생각함 )  김덕규  [12. 27]  제안서 수정 및 보완, 예상 질문 준비, 발표 준비  [12. 28]  제안서 발표  [12. 29]  프레임워크 제작 1일차 – 장치초기화   1. Device 클래스 제작  * 장치에 대한 모든 초기화 및 렌더링 담당. 렌더링 부분은 따로 추후에 Renderer 클래스 제작을 통해 기능을 분리시킬 예정. * m\_mapResourceState라는 멤버 변수를 통해 ID3D12Resource\* 변수의 리소스 현재 상태를 저장해두고 이를 변경할 때 호출해야하는 번거로움을 없앰. 렌더 타겟의 경우에는 m\_mapRenderTarget라는 멤버 변수를 통해 관리한다. 생성한 렌더 타겟을 const char\* 형태의 키 값을 통해 조회할 수 있게한다. * Device 클래스는 추후에 설명할 Gameframework가 소유할 클래스이기 때문에 렌더링을 한 함수내에 정의하기 힘들다. 따라서 Render()함수를 Render\_Begin()과 Render\_End()로 구분한다. * Render\_Begin()에서는 CommandAllocator와 CommandList를 리셋 시켜준 후, 현재 백버퍼의 상태를 렌더 타겟으로 바꿔주고 커맨드리스트에 렌더타겟 뷰와 뎁스스텐실 뷰를 초기화하고 렌더타겟을 세팅해준다. * Render\_End()에서 현재 백퍼퍼의 상태를 프리젠트로 바꿔주고 커맨드 큐에 커맨드 리스트를 넣고 프리젠트를 한다.      1. Gameframework 클래스 제작  * 게임의 전반적인 부분을 담당하는 게임 프레임워크 클래스이다. 디바이스 클래스와 씬 클래스를 가지고 있는다. * 중재자 패턴을 이용하기 위해 싱글 톤 클래스로 제작했고, 객체 간의 상호작용을 프레임워크 클래스 혹은 씬 클래스에서 함으로써 커플링을 방지할 예정이다. * 기본적으로 Update(), LateUpdate(), Render()함수를 가지고 있다.   [12. 30]  프레임워크 제작 2일차 – 기본적인 구조 제작   1. Object 클래스 제작  * 오브젝트 클래스는 내부 내용은 없고, 상속을 위한 인터페이스 클래스이다. Update(), LateUpdate(), Render()가 있으며, 각 함수는 PURE로 순수 가상함수로 제작되어 있다.  1. Scene 클래스 제작  * 씬 클래스는 오브젝트 클래스와 마찬가지로 내부 내용은 없고 상속을 해 사용할 인터페이스 클래스이다. Update(), LateUpdate(), Render(), Enter(), Exit()함수가 있으며, Enter(), Exit()함수는 씬이 바뀔 때 적용해야할 정보들을 삽입할 생각이다. * 씬 클래스는 list<Object\*> m\_LstObject[OBJ\_END]; 변수를 가지고 있다. 씬 클래스 내부에서 오브젝트를 다룰 예정이다.   [12. 31]  프레임워크 제작 3일차 – framework.h 정리   1. 프레임워크 제작 중에 필요한 매크로, include, enum 등 문서 정리. 각 문서별로 담고 있는 내용을 나눠놓음 | | |
| 다음주 할 일 | [최현욱]  1학기때 3DGP를 수강할 때 사용한 클라이언트에 서버를 연결하여 이동 및 회전을 구현해보기. 우리가 만들 클라이언트와 100%호환되지는 않겠지만 수정은 그렇게 어렵지 않을 거라 생각함.  [홍범도]  1. MicroSoft사의 애니메이션 예제 분석하기  2. 애니메이션 구현을 위한 DEMO 프레임워크 하나 만들기 ( MicroSoft사의 애니메이션 예제 참고 && 아마 이게 다음주 다 잡아먹지 않을까 생각함 ( 필요한가에 대한 생각 필요 ) )  [김덕규]  1. 프레임워크 제작 – 제작을 위한 씬 및 오브젝트 생성. 오브젝트가 담고 있을 컴포넌트의 원시 클래스 제작.  2. 사각형을 화면에 그리고, 가능하면 텍스쳐 입혀보기 | | |
| 문제점 |  | | |