|  |  |  |
| --- | --- | --- |
| EmberFall 개발 일지 | | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | 회의 내용 |
| 1월3주 회의는 스테이지에 사용될 에셋을 결정하는 시간을 갖고, 종합설계관련 기존에 나왔던 다양한 논의를 진행한다 .  악마, 애니메이션은 선별을 완료했지만, 맵 에셋은 선별을 하지 못했기 때문에 고민할 시간이 필요했다.  **HiveMind**제작자의 **Modular Haunted Village, Modular Medieval Town**를 선별했으며, 부족한 오브젝트는 무료 에셋을 이용하여 충당한다.  서버와 클라이언트가 같은 프로젝트에 있을 때 문제점에 대해 고민해보았다.  서버와 클라이언트의 파일 이름, 클래스 이름이 같은 경우 어떻게 처리해야 할지 논의하였다.  같은 경우가 발생할 때 네임스페이스를 서버에서 감싸는 것으로 해결한다.  데드레커닝 관련하여 예측의 적중여부를 저장해야하기 때문에 입력처리에 관한 결과를 저장할 수 있어야한다.  그렇다면 저장은 어디에 하는 것이 좋을지 고민을 해보았을 때,  입력처리를 하는 클래스에 입력을 받자마자 위치를 배열로 저장하는 것이 좋을 것 같다는 결론을 내렸다.  바운딩박스의 충돌감지와 처리는 시간이 오래걸리기 때문에 큰 박스의 처리를 서버에서 하고, 작은 박스는 클라이언트에서 처리를 한다.  이때, 데드레커닝과 연관되어 누구는 맞고, 누구는 안맞는 경우가 생길 수 있기 때문에 별도의 처리를 해야한다.  이는 서버에서 처리 할 예정이다.  애니메이션에서 멀티쓰레드 렌더링을 사용하는 이유에 대한 의견이 나와 의견을 주고받았다.  클라이언트에서 애니메이션의 행렬 계산을 싱글쓰레드로 작업하기 어렵기 때문에 멀티쓰레드를 사용한다는 의견이 나왔으며, SIMD로 계산하고 애니메이션의 개수가 많지 않기 때문에 불필요하다는 의견이 나왔다.  싱글쓰레드로 제작을 해보고, 문제점이 발견된다면 멀티쓰레드로 변경해보는 것이 좋다는 의견이다.  마우스 입력은 어떻게 서버에 보내는가?  클라이언트에서 회전을 처리하고, 처리 결과를 서버에 전송하는 것으로 한다.  클라이언트의 목표에 수정사항이 생겼다.  1월31일에 Release 버전1.0을 완성하는 것으로 변경하였고,  개강 전까지 애니메이션을 제작하는 것을 목표로 변경하였다. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| EmberFall 개발 일지 | | 작성자 : 김승범 | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | | 이번 주 진행 사항 |
| **1월 12일**  Texture 을 완성하였다. 차주 진행할 Mesh Filter / Mesh Renderer 구현을 위해 필요한 Default Buffer 를 작성하고  ResourceManager 에 대해 고민하였다.  **1월 13일**  장면 내의 리소스를 관리하는 방법에 대해 고민하였다.  기존 리소스 관리 방법은 다음과 같은 문제점이 있는 것으로 분석하였다.   1. 비슷한 코드가 너무 자주 반복된다. 상속과 같은 방법으로 코드를 재사용 할 수 있도록 변경해야 한다. 2. 텍스쳐는 일반 리소스와 다른 성질을 지닌다. 하지만 기존 방식은 텍스쳐와 버퍼를 비슷한 방법으로  처리한다. 이 때문에 불필요한 과정이나, 복잡한 코드가 만들어진다. 3. Mesh – Model – ModelInterface 로 이어지는 Mesh 참조 과정이 너무 복잡하다. 4. 관절이 없는 Mesh 와 관절이 있는 Skeleton Mesh 는 서로 다르다. 하지만 기존 방식을 사용하면  두 Mesh 를 같은 자료형을 통해 관리하여야 한다. 이는 이전 사례로 보아 불필요한 소요를 발생시킬  것이다.   이러한 문제를 해결하기 위한 리소스 관리, 렌더링 엔진 구조로 Unity Engine 의 컴포넌트 시스템을 차용하기로  한다. 당분간의 목표를 이 컴포넌트에 대한 공부를 하는 것으로 잡았다.  **1월 14일, 1월 15일**  해당없음  **1월 16일**  Texture 클래스를 실제로 적용시켜보니, 리소스 연결부에 문제가 있어서 수정하였다.  이번 주차의 목표인 Mesh 로드와 렌더링을 위한 Mesh Loader 프로젝트를 생성하고,  Assimp 라이브러리를 추가하였다. 바이너리 파일에 필요한 정보들을 직렬화하는데 필요한 Serializer 를 만들었다.  **1월 17일**  골격이 없는 Mesh 를 PlainMesh, 골격이 있는 Mesh 를 BonnedMesh 로 분류하고, MeshLoader 를 구성하기로  하였다. 이를 구현하기 위한 Assimp 라이브러리와 그 사용법에 대해 공부하였다.  **1월 18일**  팀원들과 주간 회의를 가졌다. PlaneMeshLoader 를 만들기 위해 Mesh Loading 의 기능을 하는 기본 클래스를  만들어, 이를 상속하도록 만들었다. 본격적으로 Scene 을 구성하기 위한 컴포넌트 패턴에 대해 공부하였다. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| EmberFall 개발 일지 | | 작성자 : 김성준 | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | | 이번 주 진행 사항 |
| **01-12**  SRWLock 클래스 구현 및 SessionManager의 Session 맵 접근 시 SRW Lock 을 사용하도록 수정  클라이언트가 연결되어 새로 추가되거나 연결 종료되어 삭제될 때는 Write Lock, 클라이언트에게 데이터를 송신할 때에는 Write 작업은 일어나지 않으므로 Read Lock을 사용하도록 수정함  **01-13**  클라이언트가 종료될 때 정상적으로 종료되지 않던 버그 수정.  수신 처리 함수에서 정상 종료에 대한 처리를 진행하지 않고 있던 것을 발견하고 수정함  **01-14**  승범이 부탁했던 쓰레드 풀을 작성함.  std::future와 std::function, std::packaged\_task를 이용하여 쓰레드가 수행하는 작업의 반환 값을 얻을 수 있게 하고, 클래스의 정적 멤버로 쓰레드 ID를 추가해 사용할 때 쓰레드 ID를 확인하고 그에 맞는 작업을 할 수 있도록 함  **01-15**  SessionManager에 Send 관련 함수를 작성하고 에코서버 테스트를 진행함.  데이터 송, 수신 작업에서 일어나는 버그를 고치고 오늘 하루 정상적으로 돌아가는 것 까지 확인함.  **01-16 - 17**  이후에 게임데이터를 송수신하기 위해 쓸 버퍼를 작성함. 여러 쓰레드에서 송신한 데이터를 담아두기 위한 새로운 버퍼 정의  I/O 쓰레드에서 패킷을 바로바로 처리하는 것이 이상적이나 그럴 경우 관리해야하는 공유변수가 너무 많아지기에 이런 방식을 선택함.  더블 버퍼링처럼 동작하도록함  Main Thread (GameLoop) -> ReadOnlyBuffer에만 접근하며 수신한 패킷들을 처리함  Worker Threads (I/O Thread) -> WriteOnlyBuffer 에만 접근하며 수신한 패킷들을 저장함  Main Thread가 데이터를 읽고자 할 때 WriteOnlyBuffer와 ReadOnlyBuffer 의 교체가 이루어짐  다음과 같이 동작하도록 로직을 작성  1. Swap 함수 호출 이후의 모든 Write 작업은 멈추고 해당 작업을 수행하려는 쓰레드는 Block  2. Swap 함수 호출 이전의 모든 Write 작업은 그대로 진행됨. 이 때 Swap 함수를 호출한 쓰레드는 Block  3. Swap 함수 호출 이전의 모든 Write 작업이 종료된 것이 보장되었을 때 Swap 함수가 실행됨  4. RecvBuffer 클래스가 아닌 PacketHandler의 Reset은 안전을 보장하지 않음.  16~17 이틀간 클래스를 만들고 에코서버에 접목하여 통신이 원활하게 이루어 지는지 테스트함. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| EmberFall 개발 일지 | | 작성자 : 김성준 | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | | 이번 주 진행 사항 |
| **01-18**  통신과정에서 Session 클래스의 mConnected 플래그를 검사하는 과정이 없어 오류가 나는 부분을 수정, 게임 컨텐츠 개발을 위한 GameScene 클래스를 새로 생성하고 Session 클래스에서 송신하는 과정에서 Overlapped 구조체를 매번 new delete하는 문제를 수정하기 위해 MemoryPool을 만들어 재사용 할 수 있도록 함 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| EmberFall 개발 일지 | | 작성자 : 정영기 | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | | 이번 주 진행 사항 |
| **01-12**  조명 제작물에서 수정해야 할 문제가 생겨 텍스쳐의 학습을 일시 중단 후, 조명의 제작물을 이어서 작업하였다.  기존의 인덱스 버퍼를 사용하지 않고 정점 버퍼만을 사용하는 단계로 나아가는 과정에서 조명이 제대로 나오지 않는 문제가 생겼다.  **01-13**  셰이더 코드와 C++코드에서 약간의 보완을 거쳤지만, 근본적인 문제는 해결을 하지 못했다.  **01-14**  기존 셰이더 코드에서 노멀벡터의 변환과정에서 이동변환요소가 섞여들어가 잘못된 결과가 도출된 것을 발견하였고, 수정하였다.  **01-15**  텍스쳐 이론과 인터넷 강의를 수강 하였으며, 종합 설계에서 담당한 블러링에 관한 이론은 학습하였다.  **01-16**  복습을 위해 프로젝트를 새롭게 만들었다.  기존의 프로젝트에서 월드변환행렬을 인스턴싱으로 셰이더 코드에 전달하였던 것을 structed buffer를 이용하여 전송 하였고, 함수를 분리하여 가시성을 높혔으며, 조명의 회전을 구현하였다.  텍스쳐의 이론을 이어서 학습하였다.  **01-17**  복습을 위해 새롭게 만든 프로젝트에 생긴 오류를 해결하였고, 텍스쳐의 제작물을 완성하였고, 제작물에 대한 피드백을 반영하여 수정하였다.  **01-18**  블렌딩에 대한 이론을 학습 하였고, 개인프로젝트에 적용하는 시간을 가졌다.  블룸효과에 대한 이론을 학습하였다. | | | |

|  |  |  |
| --- | --- | --- |
| EmberFall 개발 일지 | | |
| **3주차** | **2025.01.12 ~ 2025.01.18** | 회의 내용 |
| 1. 김승범 ( 클라이언트 )   간단한 정육면체와 평면을 띄우기 위한 간단한 Scene 구현  이를 달성하기 위한 Mesh Loader 완성 및 컴포넌트 구조 예제 완성   1. 김성준 ( 서버 )   - OpenGL로 간단한 테스트용 클라이언트를 작성하고 통신할 패킷들을 정할 예정  **-** 클라이언트 작성 후 입력처리, 간단한 물리 연산 작성  - FlatBuffer 공부   1. 정영기 ( 클라이언트 )   스텐실, 기하셰이더, 계산셰이더, 테셀레이터, 피킹, 쉐도우 관련 학습 예정 | | |
| **특이사항** | | |
| 김승범 – 1월 14/15 일에는 현장 프로젝트 최종 발표를 위한 자료 조사와 발표 자료 제작하였음 | | |