|  |  |  |  |
| --- | --- | --- | --- |
| **[ Catch Bear (캐치 베어) ]** | | | |
| 3 주 | 2021. 1. 9 ~ 2021. 1. 15 | 작성자 | 박소영 |
| 이번주  한 일 | **[0] 공동**  1. 14) 정기 모임 (3주차)  - 3주차에 공부한 내용 공유  김우찬: Lock-Based Stack, Lock-free Stack, 쓰레드 매니저  박소영: 키입력과 타이머, Material, Component 패턴, 카메라(기본적인)  고은비: Tree Object Wireframe Rendering, FBX Exporter 구현(텍스처, 애니메이션)  - 4주차에 할 일 공유  **[1] 김우찬 (서버)**  주간 목표  - Lock-base Stack/Queue, Lock-Free Stack/Queue 공부  - 메모리 관리 공부  - 소켓 프로그래밍 복습  진척도  - 멀티스레드 마무리(Lock-Base Stack/Queue, Lock-Free Stack/Queue) (100%)  - 메모리 관리 (60%)  - 소켓 프로그래밍 (0%)  Stack/Queue을 사용할 때 Lock을 사용하지 않는 Lock-Free Stack / Queue에 관해 공부함.  이제부터 ThreadManager를 이용하여 쓰레드를 관리함. 본격적으로 네트워크 들어가기 전에 메모리 관리 공부중.  DeadLock을 탐지하는 DeadLockProfiler를 만들어봄 (그래프 Dfs를 이용한 방법)  본격적인 네트워크 들어가기 전에 메모리 공부중. 이번주는 Reference Counting과 스마트 포인터에 대해 공부함.  **[2] 박소영 (클라이언트)**  주간 목표 및 진척도  - 게임수학 복습(주로 행렬, 변환행렬 부분) (100%)  - 키보드 입력, 타이머 구현 (100%)  - Scene, SceneManager 구현 (100%)  - 카메라 공부 및 구현 (50%)  1. Material을 추가하여 원래 따로 만들던 Mesh, Shader, Texture를 묶어서 한번에 관리하도록 만들었다.  2. Component 패턴을 추가했다. 고정적으로 사용할 오브젝트(카메라 같은)는 array에 담아서 관리하고, 유저가 만드는 오브젝트들은 vector에 담아서 관리한다.  3. Scene과 SceneManager를 추가했다. 기존에는 클라이언트 부분에서 객체를 만들었는데, 이제 클라이언트에서 불러오는 씬 안에서 객체를 생성하여 화면에 띄운다.  4. 게임수학을 직전 학기(3-2)에 들어서 행렬, 변환행렬 복습이 생각보다 빨리 끝났다. 남은 시간에 카메라를 공부하고 구현했다. 카메라는 조금 더 공부하고 기획한 게임에 맞게 수정이 필요하다.  **[3] 고은비 (클라이언트)**  1. 환경 오브젝트 메쉬를 FBX Exporter에서 추출한 바이너리 파일을 기반으로 와이어 프레임으로 렌더링(텍스처링X)  2. FBX Exporter 구현 완료: 텍스처(재질), 애니메이션  3. 메쉬 텍스처링 구현중 (uvs, normals, 텍스처 이름 읽어 들이는 것 까지 디버깅으로 확인, materials 아직 X)  메쉬를 와이어프레임으로 렌더링해 메쉬의 좌표들이 잘 저장되었는지 확인하였다. 이때, FBX와 DirectX12의 차이점인 좌표계, 정점 와인딩 순서, 행렬들이 FBX에서 DirectX12의 속성들로 잘 변경됐는지 확인하면서 제대로 저장되고 불러들이는지 확인하였다. | | |
| 다음주  할 일 | **[0] 공동**  1. 21) 정기 모임  **[1] 김우찬 (서버)**  - 메모리 관리(Memory pool) 공부, 소켓 프로그래밍 복습, IOCP 구현하여 네트워크 라이브러리 제작  **[2] 박소영 (클라이언트)**  - 여태까지의 클라이언트 프레임워크 흐름도 정리  - 조명 공부 및 구현  - 스카이박스 추가  - 노멀매핑 공부  **[3] 고은비 (클라이언트)**  - 환경 오브젝트(먼저), 플레이어 메쉬에 텍스처와 재질을 적용해서 렌더링  - 클라이언트에서 애니메이션 정보 불러오기 | | |
| 문제점 | **[0] 김우찬 (서버)**  Lock-Free-Stack/Queue 부분이 코드도 많고 내용이 어려워 진도가 더디다. 메모리 관리랑 소켓 프로그래밍 부분은 빠르게 넘기고 네트워크 라이브러리 제작을 빨리 들어가야 할 것 같다.  **[1] 박소영 (클라이언트)**  프레임워크를 만들어가면서 화면에 제대로 띄워지면 구현이 완료됐다고 좋아했는데, 회의때 내가 한 부분을 팀원들에게 설명해주는데 말이 잘 안나왔다. 구현이 완료됐다고 바로 다음으로 넘아가지 말고 내가 완벽히 알아서 다른 사람들에게 설명해줄 수 있을 정도로 자세히 공부해야겠다고 생각했다. 그래서 다음주에는 여태까지 구현한 클라이언트 프레임워크의 흐름도를 정리해보기로 했다.  **[2] 고은비 (클라이언트)**  - FBX Exporter에서 쓴 파일들을 클라이언트에서 읽을 때, 파일의 데이터들을 저장하고 관리할 변수들을 관리하는 것이 조금 까다로웠다. (변수를 어디서 선언하고 어떻게 처리해야 할지 + 클라이언트에서 쉽게 사용할 수 있도록 FBX Exporter를 수정하는 것)  - FBX Exporter에 시간을 너무 많이 들여서 진도가 느린 것 같다. 얼른 재질 데이터들까지 제대로 읽어와서 다음주에 최대한 빨리 메쉬에 텍스처와 재질을 적용해서 렌더링하는 것을 끝내고, 애니메이션을 시작해야겠다! | | |