|  |
| --- |
| 한국산업기술대학교 게임공학과 |
| 포트폴리오 |
| 클라이언트 부분 |

|  |
| --- |
| 박진수(Park Jin Soo)  2019-9-1 |

목차

[1. 3D 3인칭 4:4 대전 액션 게임 2](#_Toc18260404)

[기본 설명 2](#_Toc18260405)

[세부 설명 3](#_Toc18260406)

[제작하면서 어려웠던 점. 5](#_Toc18260407)

1. 3D 3인칭 4:4 대전 액션 게임

****

**게임 이미지**

기본 설명

**게임 제목**: 알약전사

**영상 링크**: -

**프로젝트 소스:** <https://github.com/GogleEarth/Pills-Fighter>

**사용 언어:** C++

**개발 환경**: Visual Studio 2017, WinAPI, DirectX12

**장르:** 3D 3인칭 멀티 대전 액션 게임

**제작 인원**: 3명

**역할:** 메인 클라이언트

**제작 기간**: 약 11개월 소요

**목표**: DirectX12를 이용한 프레임워크 구축 및 게임 시스템, 다양한 그래픽 효과 구현

**게임 내용**: 로비에서 방을 만들어 인원을 모아 팀과 캐릭터를 정한 후, 게임이 시작되면 각 캐릭터마다 주어진 무기를 사용하여 상대팀의 점수를 0으로 만들 때까지 싸우는 방식.

세부 설명

1. 시스템 구조:

클라이언트 실행 시 DirectX12를 사용하기 위한 장치 생성, 메모리 할당 등 초기화 작업을 한 후 타이틀을 생성함. 타이틀, 로비, 룸 등을 Scene클래스로 분류하였고 Scene을 생성할 때 해당 Scene에서 사용하는 리소스를 생성하도록 했음. 해당 Scene에서 사용할 리소스들을 생성한 후 루프를 돌며 1프레임마다 오브젝트에 시간을 적용시켜 Animate, 충돌 처리 등을 처리한 후 화면에 렌더링하도록 하였음.

1. 클라이언트 상세 설명:
2. GameFramework 클래스

클래스 생성 시 DirectX12 디바이스, 명령 큐, 명령 리스트 등 화면에 렌더링하기 위해 필요한 것들을 생성한 후, 매 프레임마다 서버로부터 시간을 받아 오브젝트들을 Animate 시켜서 화면에 렌더링해주는 뼈대가 되는 클래스. 서버로부터 패킷을 받아 조립하여 처리하거나, 키보드, 마우스 입력 등을 받아 처리하며 필요할 경우 Scene 클래스에서도 처리할 수 있게 처리함.

1. Scene 클래스

GameFramework로부터 생성되며 Scene 클래스는 상황에 따라 맞는 클래스를 생성하기 위해 상속하여 정리하였음. Scene 클래스의 모든 함수는 기본적으로 GameFramework 클래스로부터 호출되며 UI 텍스쳐, 메쉬 정보, 재질 정보 등 화면에 렌더링하기 위해 사용할 리소스 생성, 오브젝트 Animate, 렌더 타겟에 Rendering, 패킷이나 키보드, 마우스 입력 처리 등이 그에 해당됨..

1. Shader 클래스

오브젝트, UI 등 렌더 타겟에 Rendering하기 위해 필요한 렌더링 파이프라인을 생성하는 클래스. Shader 클래스도 상속을 통해 정리하였음. 그리고 배치 처리를 하기 위해서 Shader 클래스에서 오브젝트를 관리할 수 있게 구현하였음. 렌더링 파이프라인을 명령 리스트에 Set해야만 그 렌더링 파이프라인으로 렌더링이 되는데 똑같은 파이프라인을 여러 번 Set할 필요는 없기 때문에 이와 같이 구현함.

1. Mesh 클래스

정점 버퍼를 생성하여 최종적으로 Draw 명령을 명령 리스트에 집어넣는 클래스. 정점 버퍼는 파일을 읽어 들이거나 손으로 코딩하여 만들며, 상속을 통해서 정리하였음.

1. Texture 클래스

.dds 파일을 읽어 리소스를 생성하고 쉐이더 리소스를 명령 리스트를 통해 Set하여 렌더링할 때 텍스쳐 리소스를 GPU에서 사용할 수 있게 하는 클래스.

1. Material 클래스

파일을 읽어 재질 정보를 저장하고 Texture 클래스를 생성하여 리소스를 생성하는 클래스. 재질 정보를 상수 버퍼에 Set하고 Texture를 서술자 테이블에 Set하는 역할을 함.

1. Model 클래스

Mesh 클래스와 Material 클래스를 관리하는 클래스. 계층구조를 표현할 수 있도록 트리 형태로 구현하였으며 모델 정보가 파일에 쓰여 있으면 Mesh 클래스나 Material 클래스는 Model 클래스를 통해서 생성됨.

1. Object 클래스

Model 클래스, 월드 변환 행렬, HP, 속도 등 하나의 오브젝트가 표현되기 위해 필요한 변수들을 멤버 변수로 갖는 클래스 Scene 클래스 혹은 배치 처리를 하는 Shader 클래스에 의해 렌더링 되거나 Animate 됨.

1. Camera 클래스

투영 변환 행렬, 카메라 변환 행렬 등 게임에서 카메라와 같은 역할을 하는 클래스. 게임 내에서는 기본적으로 하나의 Camera가 존재하며 Player를 따라 다니게 하였음.

제작하면서 어려웠던 점.

1. **사용하기 힘든 그래픽 라이브러리**

아직 DirectX12에 대해 정리가 되지 않은 상태에서 프로그래밍을 시작했기 때문에 시스템 하나하나를 구현할 때마다 시행착오를 거치는데 그 때마다 포기하고 싶은 생각이나 스트레스 때문에 많이 힘들었습니다. 그래도 참고 견디면서 하다 보니 원하는 결과가 나왔을 때에는 나쁘지만은 않다는 생각을 했습니다.

1. **벡터와 행렬 등 수학적 지식 부족**

오브젝트를 회전하거나 원하는 카메라 연출을 하기 위해서 벡터나 행렬에 대해 잘 알아야 하는데 그런 지식이 많이 부족했기 때문에 원하는 결과를 만들어 내기가 힘들었습니다.

1. **새로운 지식 습득**

환경 매핑, 모션 블러, 글로우 이펙트 등 저에게는 아주 생소한 것들을 게임에 추가하기 위해서 인터넷에 검색하거나 책을 찾아보면서 공부를 했는데 원하는 결과가 잘 안 나오고 대부분의 정보는 영어로 되어있어서 많이 해석하고 이해하는데 많이 힘들었습니다.