**네트워크 게임 프로그래밍**

**Term Project 추진계획서**

**7조**

**2019184030 조성원**

**2022180005 김상혁**

**2020180040 진현서**

**목 차**

**1. 애플리케이션 기획**

1) 게임 소개

2) 게임 구성

3) 스크린 샷

4) 게임 플로우

5) 개발 환경

**2. high level design**

1) sever/client high level design

2) server Architecture

3) client Architecture

**3. low level design**

1) Packet List

2) Function List

**4. 팀원 별 역할분담**

**5. 개발일정**

**1. 애플리케이션 기획**

**게임 소개**

- 게임 이름 : Unity Of Mind

- 게임 장르 : 아케이드 게임 , 협동 게임

- 게임 플레이 방법 : 앞으로 다가오는 블럭을 피해서 끝까지 살아남는 협동 게임

- 플레이어 수 : 3명

조성원 팀원이 컴퓨터 그래픽스 과목에서 과제로 제출한 프로젝트를 기반으로 제작하였습니다.

플레이어들은 앞으로 다가오는 블럭을 피해야 함. 블럭에 맞으면 체력이 감소 ( 1/3 감소) . 모든 플레이어가 체력을 공유함, 체력이 0이 되면 게임 오버, 모든 스테이지를 통과하면 게임 클리어

**조작법**

- Left 키 , Right 키 : 각 방향에 맞는 이동

- z 키 : 플레이어 색을 빨간색으로 변경

- x 키 : 플레이어 색을 초록색으로 변경

- c 키 : 플레이어 색을 파란색으로 변경

- v 키 : 플레이어 크기 축소 / 확대

- 1 / 3 키 : 카메라 모드 변환

**스테이지 별 컨셉**

1 스테이지 : 블럭이 없는 공간으로 이동해 블럭을 통과

2 스테이지 : 블럭의 색과 캐릭터의 색을 일치시켜 블럭을 통과

3 스테이지 : 캐릭터의 크기를 축소시켜 빈 공간으로 블럭을 통과

**스크린 샷**

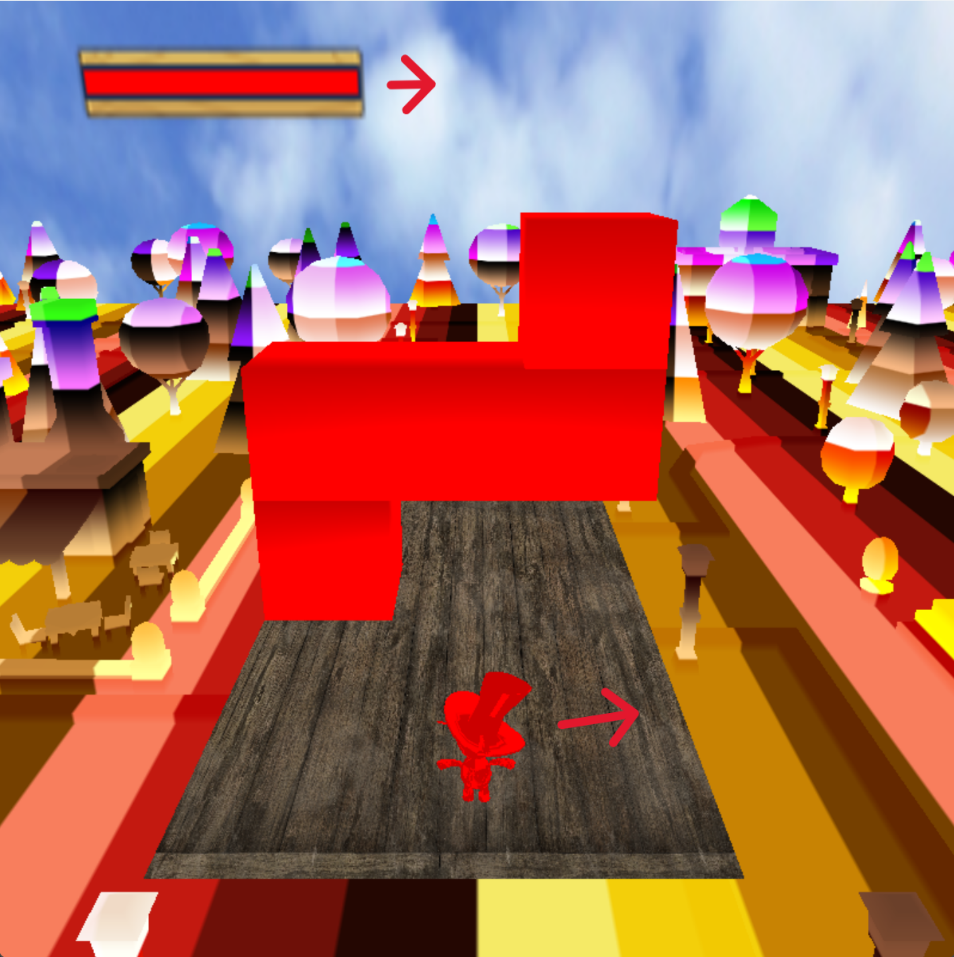


**게임 종료 버튼**

**플레이 버튼**

**로비**

**1스테이지**

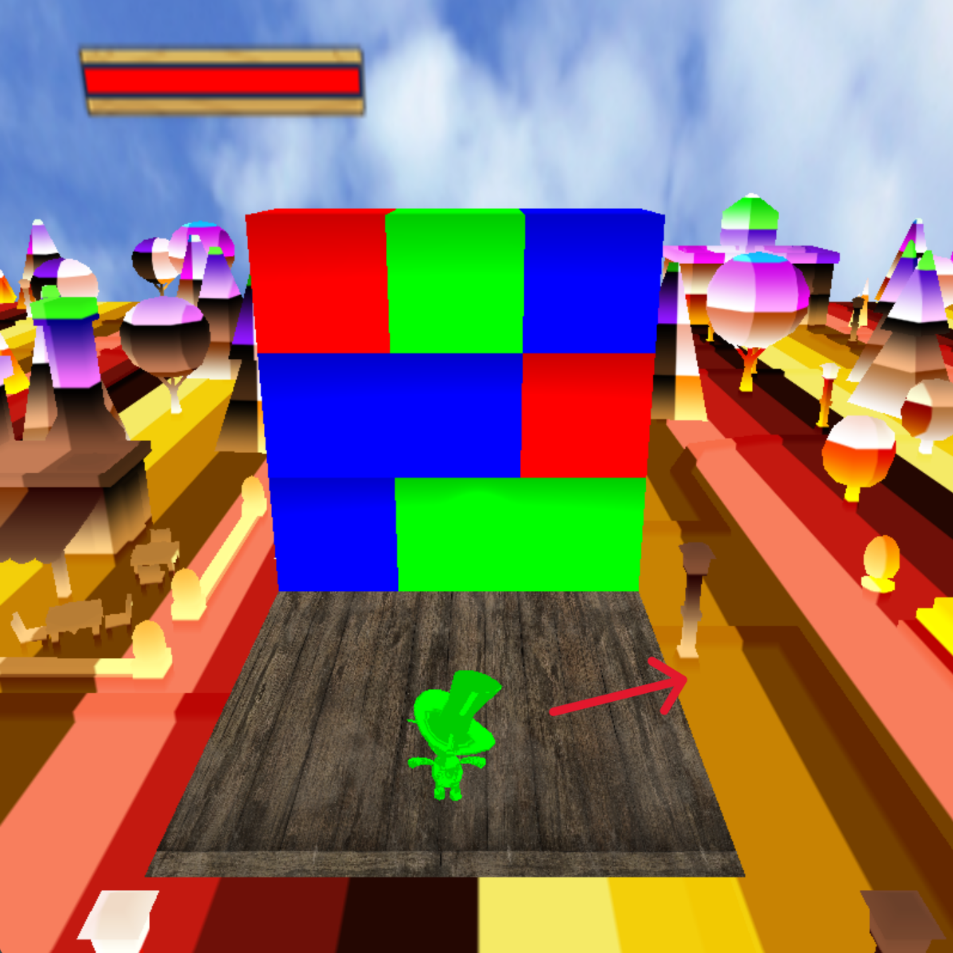


**- 블럭이 스테이지 마다 10번씩 앞으로 앞으로 다가옴**

**- 블럭이 다가오면 플레이어는 빈공간으로 이동해서 피해야 함**

**- 블럭에 맞으면 HP 감소**

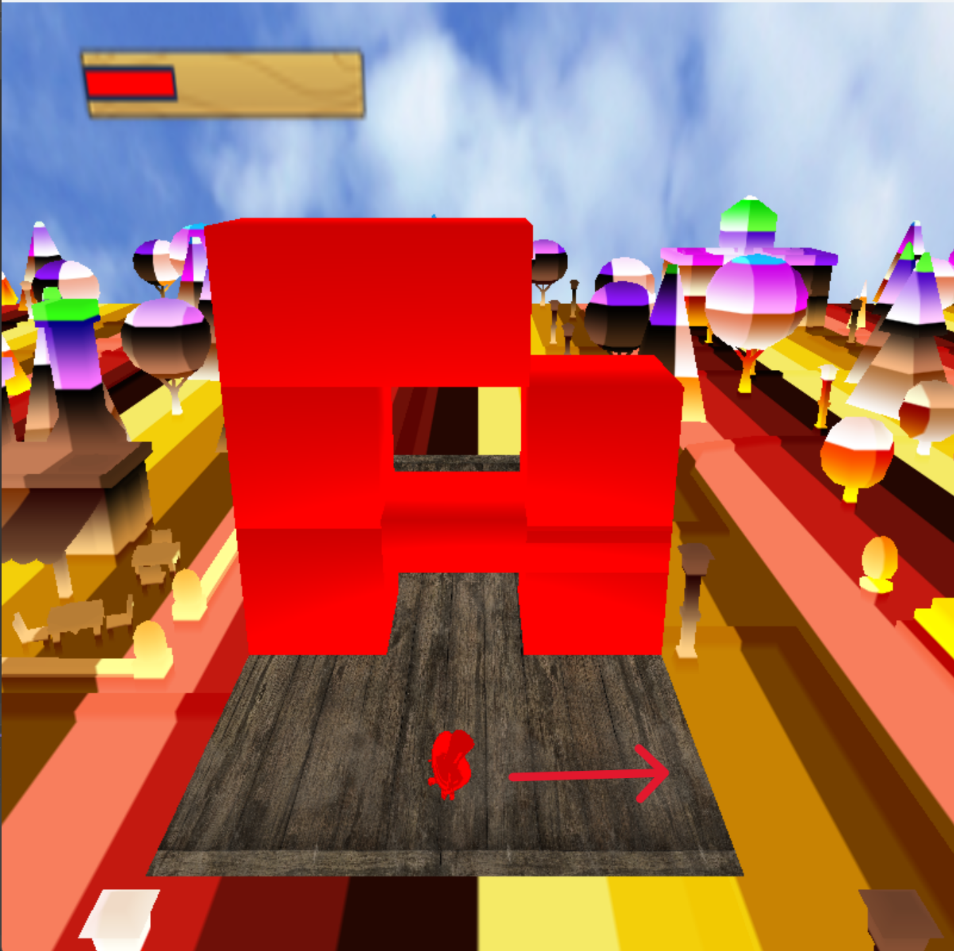
**플레이어 HP ( HP 가 0이 되면 게임오버 , 한번 충돌시 1/3 감소)**



**- 플레이어는 블럭에 보이는 색과 똑같은 색으로 캐릭터를 바꾸어서 블럭을 통과해야 함**

**- 다른 색으로 통과 시 HP 감소**

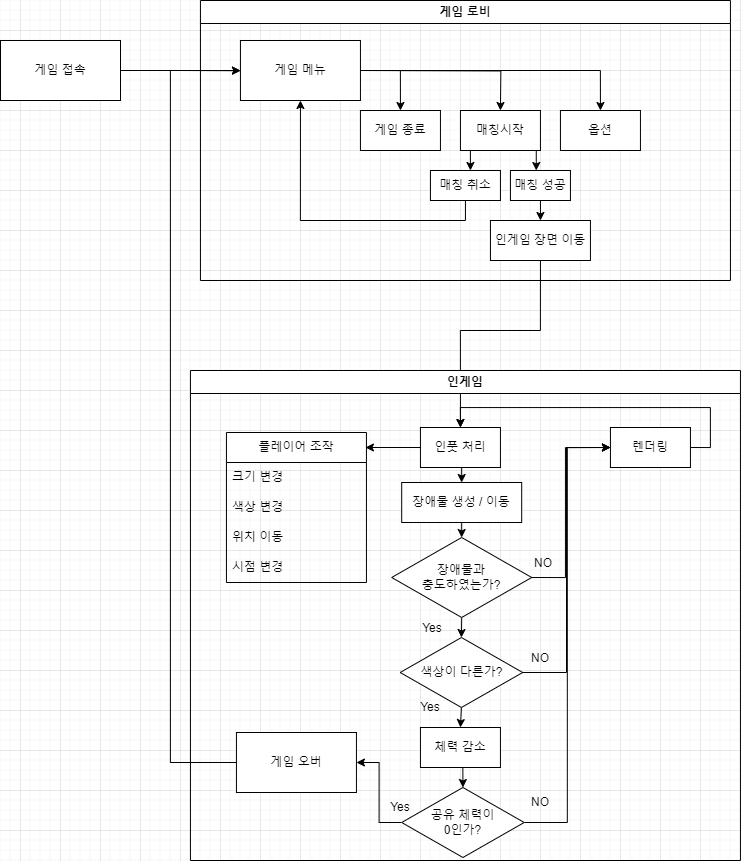
**2스테이지**



**- 플레이어는 캐릭터의 크기를 줄여서 블럭을 통과해야 함**

**3스테이지**

**게임 플로우**



**개발 환경**

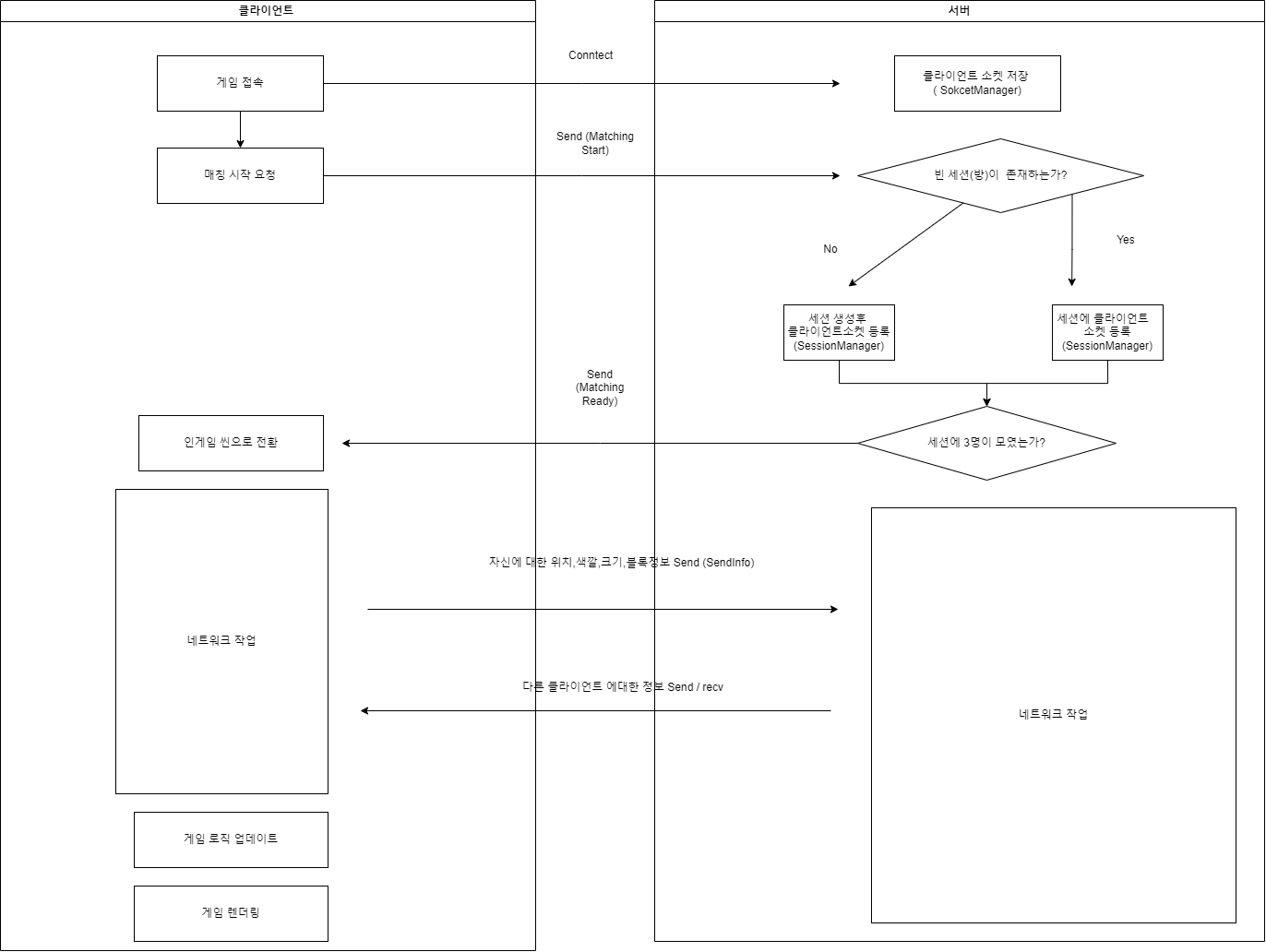
- IDE : Visual Studio

- API : OpenGL

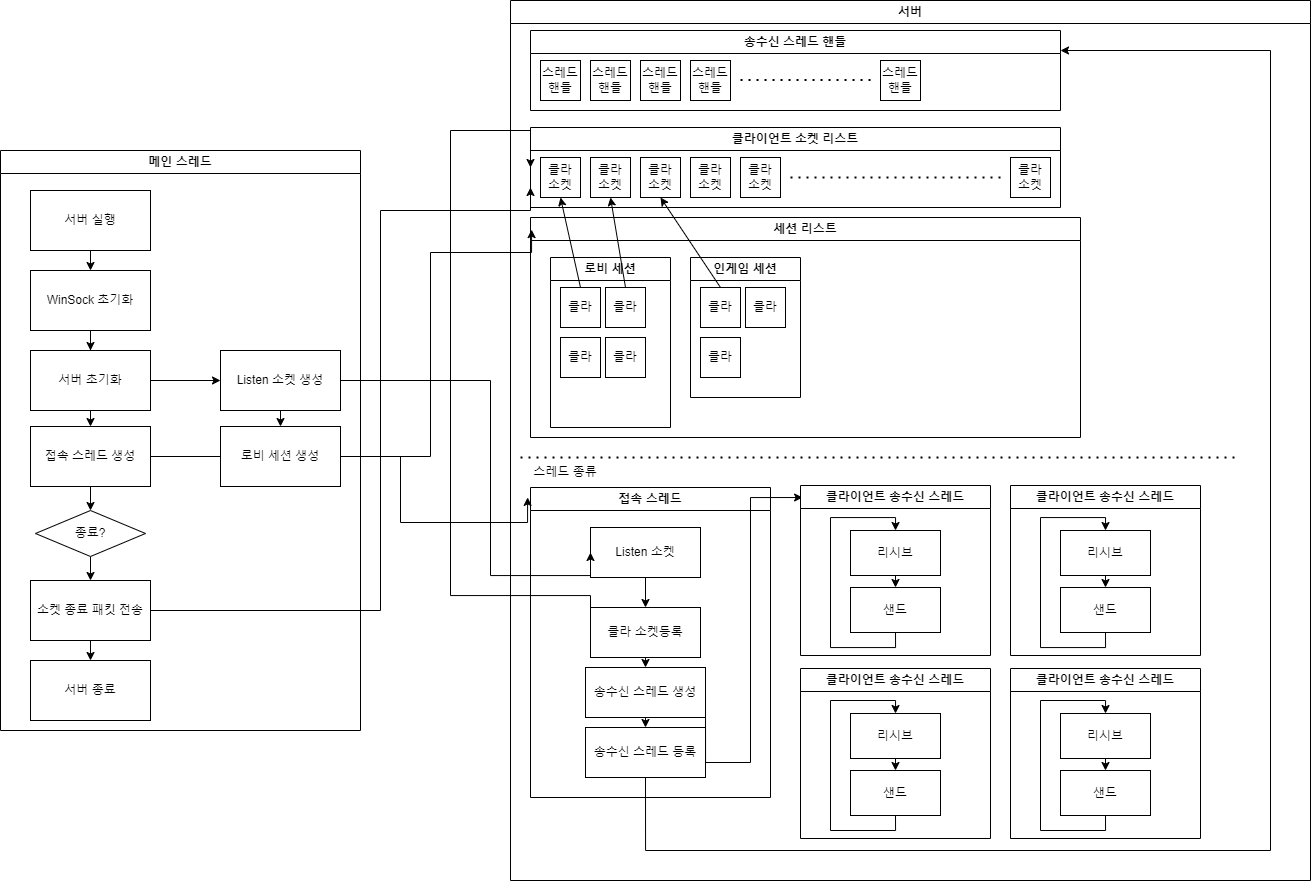
- 사용 언어 : C , C++

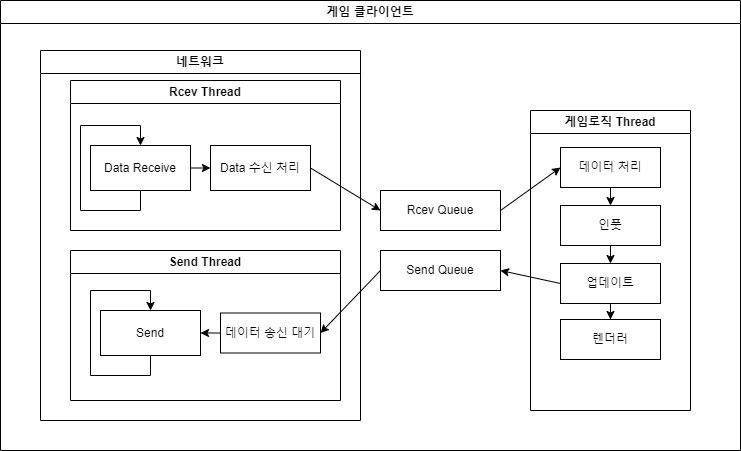
**2. High Level Design**

**Server / Client 데이터 송수신 플로우**



**- Server Architecture**



**- Client Architecture**

**3. Low Level Design**

**1. Packet 처리**

**Server Function**

-void RecvInfo() : 모든 클라이언트르 부터 정보를 받는 함수.

-void SendInfo() : 모든 클라이언트로 부터 받은 정보를 다시 Echo 해주는 기능

-void CreateSession(Socket& socket) : 로비에서 클라이언트 소켓을 받아서 세션을 만들어주는 함수.

-void ParticipateSession(Socket& socket) : 로비에서 클라이언트 소켓을 받아 세션에 참가하는함수.

-void MathcingCancle() : 로비에서 클라이언트로 부터 요청을 받아 세션에서 소켓을 제거하는 함수.

-void MathcingAccept() : 클라리언트로 요청을 받아서 세션이 없으면 CreateSession(Socket& socket) 을 호출하여주고

세션이 있으면 ParticipateSession(Socket& socket) 을 실행 해주는 함수.

-void ReStart() -recive /send : 세션을 유지한채로 다시한번 인게임씬으로 시작하게 해주는함수.

-void ReturnMenu() -recive /send : 세션을 해제시키고 메뉴화면 으로 돌아가는 함수

-void MakeBlock() : 블록을 랜덤하게 만들고 충돌하지 않아야 할 블록 정보를 벡터에 저장하는 함수

**Client Function**

-void SendInfo() : 클라이언트에게 자신의 색깔,위치,크기 정보를 넘겨주는 함수

-void RecvInfo() : 서버로부터 다른 클라이언트의 색깔,위치,크기 정보를 받아오는서 처리하는 함수

-void MathcingStart() : 서버에게 매칭을 해달라고 요청 하는 코드 함수

-void MathcingCancle() : 서버에게 매칭을 취소해달라고 요청 하는 코드함수.

-void MathcingReady() : 매칭이 완료되 서버에서 신호를 받아 인게임 씬으로 이동 하는 함수

-void ReStart() -recive /send : 게임오버되고 인게임으로 돌아가 게임을 다시 수행하는 함수

-void ReturnMenu() -recive /send : 게임오버되고 메인메뉴로 돌아가는 함수

-void TryPopPackets(std::vector<Packet>& packets); : 서버로부터 수신한 패킷 목록을 가져오는 작업

-void CreateBlock(Cube\* cubes, int count); : 서버로부터 수신한 블록정보를 기반으로 블록을 생성하는 코드

-void SyncClientPlayer(PlayerStateInfo& info, int clientID); : 다른 클라이언트의 플레이어 유닛 정보를 반영하는 함수

-void SyncStageHp(PlayerStateInfo& info, int clientID); : 다른 클라이언트의 플레이어 유닛 정보를 반영하는 함수

-void CollisionBlock(); : 다른 클라이언트의 플레이어 유닛 정보를 반영하는 함수

-void OnGameOver(); : 게임이 오버됬다는 패킷 작업

-void CreateOtherClientPlayer() : 다른 클라이언트를 시각적으로 볼 수 있는 뷰어 객체를 생성하는 함수

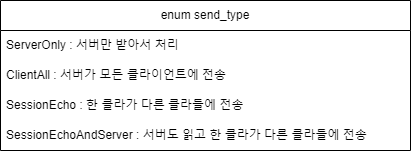
-void RenderOtherClient() : 다른 클라이언트를 개별적으로 랜더링하는 함수

-void FindOtherClientPlayer(int clientID) : 클라이언트 아이디로 매칭되는 뷰어객체 가져오는 함수

**2. Packet Protocol**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Size | 4byte | 1byte | 2byte | 4byte | 가변적 |
| Type | unsigned Int | send\_type | enum | unsigned int | struct |
| Name | **Header** | **SendType** | **PacketType** | **DataSize** | **Data** |

**Enum SendType**



**Enum PacketType**

enum class Packet

{

PlayerState

StartMatching,

CancelMatching,

MatchingSuccess,

MoveSession,

EndSession,

PlayerStateInfo,

OnCollisionEvent,

OnGameEnd,

OnGameClear,

UpdateHealthInfo,

UpdateTimeInfo,

OnCollisionObjectCreated,

CloseSessionPlayer,

CloseClient,

CloseServer

};

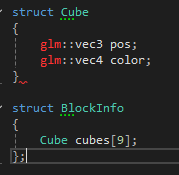
**3. Network Data Struct**

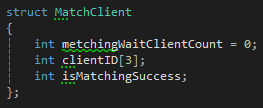












**4 . 팀원 별 역할 분담**

**조성원**

매칭 화면 제작

매칭 취소 기능 제작

전송할 정보들 구조체로 묶는 작업

서버에서 블럭 패턴을 만들고 충돌 정보를 저장하는 작업

다른 클라이언트의 동작을 볼 수 있는 뷰어 객체를 만드는 작업

다른 클라이언트 유닛 객체 렌더링 분리 작업

**진현서**

패킷 프로토콜 설계 및 패킷 코드 작성

패킷으로 주고 받기 위한 인게임 struct 리스트화

Lock 을 통해서 네트워크 스레드에서 게임 스레드로 데이터 가져오기

프레임당 플레이어 위치정보 전송

프레임당 위치정보 수신밑 반영 작업

충돌시 서버에 패킷 전송.

게임 오버시 세션 종료 패킷 전송 작업

**김상혁**

클라이언트 소켓을 세션 단위 그룹화 작업

로비 에 대한 로직구현

클라이언트에서 텍스트를 생성하여 렌더링 해 주는 코드추가

프레임워크 공용 클래스 설계 및 작업

패킷 프로토콜 설계 및 패킷 코드 작성

**5. 개발 일정**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  |  |  |  | 11/1 | 11/2 |
|  |  |  |  |  |  | 시험공부 |
| 11/3 | 11/4 | 11/5 | 11/6 | 11/7 | 11/8 | 11/9 |
| 시험공부 | 패킷으로 주고 받을 struct 리스트화 | 시험공부 | 교양 시험 | 패킷 프로토콜 설계 및 패킷 코드 작성 | 패킷 프로토콜 설계 및 패킷 코드 작성 2일차 | Lock 을 통해서 네트워크 스레드에서 게임 스레드로 데이터 가져오기 |
| 11/10 | 11/11 | 11/12 | 11/13 | 11/14 | 11/15 | 11/16 |
| Lock 을 통해서 네트워크 스레드에서 게임 스레드로 데이터 가져오기2일차 | Client의 TryPopPackets() 네트워크 함수 구현 | 시험공부 | 전공 시험 | Client의CreateBlock()함수 구현 | 3D게임프로그래밍 2차과제 | 3D게임프로그래밍 2차과제 |
| 11/17 | 11/18 | 11/19 | 11/20 | 11/21 | 11/22 | 11/23 |
| 프레임당 플레이어 위치정보 전송  SyncClientPlayer() | 종합설계기획서 작성 | 3D게임프로그래밍 2차과제 | 3D게임프로그래밍 2차과제 | 프레임당 위치정보 수신 및 반영 작업 | 충돌시 서버에 패킷 전송.  CollisionBlock() |  |
| 11/24 | 11/25 | 11/26 | 11/27 | 11/28 | 11/29 | 11/30 |
|  | SyncStageHp() | 게임 오버 시 세션 종료 패킷 전송 작업 | OnGameOver(); 구현작업 | OnGameOver(); 구현작업 | 1차 병합 |  |
| 12/1 | 12/2 | 12/3 | 12/4 | 12/5 | 12/6 |  |
|  |  |  |  |  |  |  |

**2020180040 진현서**

**2022180005 김상혁**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  |  |  |  | 11/1 | 11/2 |
|  |  |  |  |  | 클라이언트 실행되게 종속성 걸기 작업  Text 렌더링 되게 코드추가작업 |  |
| 11/3 | 11/4 | 11/5 | 11/6 | 11/7 | 11/8 | 11/9 |
|  |  |  | 모두가 공용적으로 사용할수있는프레임워크(클래스단위)작업  -ClientManager |  |  | 모두가 공용적으로 사용할수있는프레임워크(클래스단위)작업  -SessionManager |
| 11/10 | 11/11 | 11/12 | 11/13 | 11/14 | 11/15 | 11/16 |
| 모두가 공용적으로 사용할수있는프레임워크(클래스단위)작업  -ThreadManager |  |  | 모두가 공용적으로 사용할수있는프레임워크(클래스단위)작업  - |  |  | 모두가 공용적으로 사용할수있는프레임워크(클래스단위)작업  - |
| 11/17 | 11/18 | 11/19 | 11/20 | 11/21 | 11/22 | 11/23 |
| 로비작업-  게임시작시 Session 생성작업  CreateSession(Socket& socket)  ParticipateSession(Socket& socket) |  |  | 로비작업-  네트워크에서  Session 단위  처리 작업  MathcingCancle  MathcingAccept |  |  | 로비작업-  클라이언트쪽에서  Session 요청 처리작업  MathcingStart() :  MathcingCancle() |
| 11/24 | 11/25 | 11/26 | 11/27 | 11/28 | 11/29 | 11/30 |
| 로비작업-  클라이언트쪽에서  Session 요청 처리작업  MathcingReady() : |  |  | 로비작업-  게임종료시 Session 정리작업  ReStart(Socket& socket) |  |  | 로비작업-  게임종료시 Session 정리작업  ReturnMenu() |
| 12/1 | 12/2 | 12/3 | 12/4 | 12/5 | 12/6 |  |
|  |  |  |  |  |  |  |

**2019184030 조성원**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  |  |  |  | 11/1 | 11/2 |
|  |  |  |  |  | 클라이언트 Block 생성 코드를 서버로 옮기기 위한 클라이언트 코드 수정 |  |
| 11/3 | 11/4 | 11/5 | 11/6 | 11/7 | 11/8 | 11/9 |
| 기존에 클라이언트에서 만든 블록 생성과 충돌 영역 정보를 서버에서 처리하도록 작업  - MakeBlock() |  | 클라이언트와 서버가 주고받을 플레이어 및 블럭 데이터 정보 Struct 화 작업 |  | 클라이언트와 서버가 주고받을 플레이어 및 블럭 데이터 정보 Struct 화 작업 |  |  |
| 11/10 | 11/11 | 11/12 | 11/13 | 11/14 | 11/15 | 11/16 |
|  |  |  |  | 게임 시작 시 매칭화면 UI로 넘어가도록 클라이언트 수정 |  | 매칭화면일때 매칭 취소가 가능하도록 클라이언트 수정 |
| 11/17 | 11/18 | 11/19 | 11/20 | 11/21 | 11/22 | 11/23 |
|  |  | 다른 클라이언트를 시각적으로 확인할 수 있는 뷰어객체 생성  CreateOtherClientPlayer() |  | 다른 클라이언트를 시각적으로 확인할 수 있는 뷰어객체 생성  CreateOtherClientPlayer() |  | 클라이언트 아이디로 매칭되는 뷰어객체 가져오는 함수 작업  FindOtherClientPlayer(int clientID) |
| 11/24 | 11/25 | 11/26 | 11/27 | 11/28 | 11/29 | 11/30 |
| 뷰어 객체들을 분리해서 렌더링 하도록 코드 작성  RenderOtherClient() |  | 뷰어 객체들을 분리해서 렌더링 하도록 코드 작성  RenderOtherClient() |  |  |  |  |
| 12/1 | 12/2 | 12/3 | 12/4 | 12/5 | 12/6 |  |
|  |  |  |  |  |  |  |