**작업 진행 보고서**  
네트워크 게임 프로그래밍

7팀 : 김상혁, 진현서, 조성원

**구현 내용  
서버 구현**

**송수신 스레드**

클라이언트가 접속할 경우 송수신을 담당할 Thread를 클라이언트당 1개씩 생성합니다.

클라이언트가 전송하는 정보를 MYCMD struct를 통해 송수신하는 정보를 Enum 형태로 구별하여, 패킷 종류에 따라서 서버에서 관리하는 HP정보를 수정하거나, 다른 클라이언트들에게 정보를 재전송 하는 등의 처리를 수행합니다.

**클라이언트 구현**

**마스터 클라이언트 기법을 활용한 벽 생성 동기화**

벽의 생성을 동기화 하기위해서 마스터 클라이언트가 총괄하여 블록의 생성정보를 동기화하도록 방향성을 잡았습니다.

클라이언트의 매칭이 끝나고 게임플레이가 시작될 시 클라이언트중 Master클라이언트를 선정하도록 구성하였고. 마스터클라이언트가 대표하여 블록을 생성하고 나면, 생성된 블록의 정보를 Server에 송신합니다. 그리고 블록의 정보를 송신받은 Server는 다시 마스터 클라이언트를 제외한 남은 클라이언트들에게 해당 정보를 재송신하여  
해당 블록 정보를 받은 게스트 클라이언트들이 최종적으로 정보를 기반으로 블록을 똑같이 생성하여 블록을 동기화 하게 설계하였습니다.

**Block Collision 및 Player의 정보의 상호 전파**

Player와 Block Collision은 마스터 여부와 관계없이 각자의 Client의 로직에서 독립적으로 해당 조건 검사를 하여 서버로 정보를 전송을 하게 됩니다.

블록 충돌 패킷을 서버가 수신할 경우 서버에서 중앙 통제하는 HP를 수정하여 HP갱신 패킷을 모든 클라이언트에게 재전송하게 되고,

Player의 Info 정보를 수신할 경우, 수신한 대상 Client를 제외한 나머지 모든 Client에게 정보를 재전송 하는 방식으로 Client간 정보를 서로에게 전달하는 방식으로 동기화 작업을 처리하였습니다.

**조성원**

**11월 7일 목요일**

<Packet>

플레이어 위치 , 색상 , 크기 정보 구조체 패킷화 작업

struct ClientInfoPacket

**11월 8일 금요일**

<Client>

뷰어 클라이언트 객체 변수 및 함수 정의 및 작성

ViewerPlayer.h , ViewerPlayer.cpp 파일 코드 작성

**11월 9일 토요일**

<Client>

ViewerPlayer 가 모든 클라이언트의 화면에 적용이 되는지 테스트

**11월 14일 목요일**

<Client>

뷰어 클라이언트를 생성 , 검색 , 삭제 하는 함수 작성

void CreateClientPlayer(int ClientID);

ViewerPlayer\* FindClientPlayer(int ClientID);

void RemoveClientPlayer(int ClientID);

**11월 15일 금요일**

<Client>

내 클라이언트의 접속을 끊는 함수 작성

void DisConnectClient();

접속을 끊은 클라이언트가 접속을 끊었다는 패킷을 전송하면 내 클라이언트가 반영하는 함수작성

void DisConnectClientInfo(int ClientID, bool isMaster);

**11월 17일 일요일**

<Client>

내 플레이어의 위치, 색깔, 크기 정보가 변경되었을 때 다른 클라이언트들에게 알려주는 함수 작성

void SendPlayerInfo();

<Client>

다른 클라이언트의 위치, 색깔, 크기 정보가 변경되었을 때 뷰어플레이어 객체를 업데이트 해주는 함수 작성

void ClientInfoData(int ClientID , const ClientInfoPacket& cInfo);

**11월 19일 화요일**

<Client>

뷰어 플레이어의 X 좌표의 위치를 업데이트하는 함수 작성

void updateViewerPosX(ViewerPlayer\* pViewer , float PosX);

**11월 21일 목요일**

<Client>

다른 클라이언트의 위치가 바뀌면 그 클라이언트 ID 를 가지는 뷰어 플레이어의 색과 크기를 변경하는 함수를 작성함

void updateViewerColor(ViewerPlayer\* pViewer , float r , float g , float b);

void updateViewerScale(ViewerPlayer\* pViewer , float x , float y);

인게임 시작 전 ViewerPlayer 객체들을 생성하는 함수 작성

void InitViewerPlayer(int MyClientID);

<Server>

어떤 플레이어의 정보가 변경되었을 때 서버에서 다른 모든 클라이언트에게 그 정보를 쏴주는 함수 작성

void PlayerInfo(SOCKET socket , int clientID , const ClientInfoPacket& Info);

**11월 22일 금요일**

<Client>

뷰어 플레이어 캐릭터의 모델을 마법사 모델로 변경

스테이지 인덱스 오류 수정

인게임에 hp 스크린 추가

**11월 23일 토요일**

<Test>

플레이어 위치 동기화 테스트 및 남은 진행 상황 공유

**11월 28일 목요일**

<Test>

클라이언트 간 벽 속도 동기화 테스트를 통해 버그를 발견하고 서로 공유

**12월 1일 일요일**

<Test>

마스터 클라이언트 위임 문제 버그 발견 및 수정

**12월 2일 월요일**

<Test>

클라이언트 종료 시에 나머지 클라이언트 간 위치 , 벽 동기화가 잘 이루어지는 지 테스트

클라이언트 종료 시에 버그를 발견하고 공유

**12월 3일 화요일**

<Test>

팀원들과 모여서 여러가지 상황을 테스트

**12월 4일 수요일**

Progress Report 작성

**김상혁**

**작업 일정 요약**

**10월 30일 ~ 11월 5일**

1. **공통 프레임워크 제작**
   * **PacketProtocol** 설계 및 MYCMD 구조 제작.
   * **ServerManager 클래스 작업**:
     + Listener 소켓 구현.
     + PackDecode 작업.
   * **ClientManager 작업**:
     + PacketDecode와 ConnectionSocket 작업.
   * **Thread 단위 작업 설계** 완료.
   * **LobbySession 작업**:
     + ConnectionClient를 이용해 SendConnectServer 구현.
     + SendMatchingStart, MatchingAccept 작업 수행.

**11월 6일 ~ 11월 12일**

1. **로비 테스트 검증 작업**.
2. **LobbySession 작업**:
   * SendMatchingCancel, MatchingOff 구현.
3. **작업 브랜치 통합**:
   * Merge 및 종합 빌드 테스트.
   * 개발 일정에 대한 피드백 진행.
4. **텍스트 동적 렌더링 코드 추가**.
5. **버그 해결**:
   * 접속 종료 시 클라이언트가 나갔다는 패킷 전송이 안되는 문제 수정.

**11월 13일 ~ 11월 16일**

1. **LobbySession 검증 테스트**:
   * SendMatchingCancel 및 MatchingOff 테스트 완료.
2. **작업 브랜치 Merge**:
   * 종합 빌드 테스트.
   * 일정에 따른 추가 피드백 진행.

**11월 17일 ~ 11월 23일**

1. **벽 동기화 작업**
   * 벽 관련 동기화 문제 수정 작업 중.
   * 11월 20일에 버그 완벽히 해결 완료.
2. **버그 픽스**:
   * 접속 종료 시 클라이언트 나간다는 패킷 미전송 문제 해결.
3. **작업 브랜치 Merge**:
   * 통합 및 종합 테스트를 통해 빌드 점검.

**11월 24일 ~ 11월 30일**

1. **벽 동기화 버그 수정**:
   * 버그를 완벽히 해결.
2. **버그 해결**:
   * 리소스가 계속 생성되는 문제로 인한
   * 메모리 비정상 증가 해결.
3. **작업 브랜치 Merge**:
   * 종합 빌드 테스트 및 피드백 반영 완료.

**12월 1일 ~ 12월 4일**

1. **벽 동기화 최종 점검**:
   * 모든 관련 기능을 안정적으로 구현.
2. **최종 정검 및 버그 픽스**:
   * 시스템 전반을 마지막으로 점검하며 남은 버그 수정 완료.

**진현서**

**11월 14일~11월 16일**

**함수 추가**

void BlockCreate();

void BlockCreateReceive();

void HpUpdate(int hp)

커맨드 구분용 CMDCODE 3개 추가.

BlockDataRecv,

BlockCollision,

HPSync,

패킷 송수신 구조체 2개 추가.

struct BlockCreateInfo

struct HPInfo

**11월 17일~11월 29일**

클라이언트 작업

Client::BlockCreate() 마스터클라이언트의 블록정보 전송 함수 작업.

BlockCreateReceive() 게스트 클라이언트 블록정보 수신 및 반영 함수 작업,

Client::HpUpdate() 수정 및 HP 정보 수신 함수 작업.

Client::BlockCollision() 충돌시 충돌 이벤트 전송 함수 작업.

서버 작업

case ENUM::BlockDataRecv:

MakeBlockSend(socket);

break;

case ENUM::BlockCollision:

BlockCollision(socket);

블록 데이터 처리 작업 추가

충돌 이벤트 처리 작업 추가

**11월 20일**

충돌 로직에서 클라이언트가 Wall 의 EmptyIdx를 초기화 하는 시점에 대한 버그가 발생하여 해당 버그를 픽스

**11월 24일**

Server 측 코드에서 HP가 증가해야하는데, 감소하는 로직으로 구성되어 있어 해당 로직 수정

**11월 25일**

마스터 클라이언트가 게임을 플레이하던 중 종료할 시 남아있는 클라이언트중 새로운 마스터 클라이언트를 선정하여 마스터 클라이언트의 권한을 위임하고, 로직을 지속적으로 정상 작동하도록 수정하는 작업.

**11월 28일**

마스터 클라이언트를 선정하는 과정에서 발생한 버그 로직을 수정.

**11월 29일**

클라이언트가 접속을 종료하였을 때 Viewer객체가 Delete 되는 로직을, 객체 재사용성을 위하여 Rendering Disable 방식으로 전환하는 작업

Server에서 \_readyCount를 0으로 초기화 하는 시점을 게임이 매칭되어 시작하는 시점으로 로직 이동,

**12월 01**일

Client가 접속을 종료하였을 때 Socket을 보관하는 ConcurrentQueue에서 socket 정보를 지우는 과정에서 발생한 로직 오류를 수정함.

**12월 03일**

게임이 종료되고 재시작될 때 발생되는 Init 정보들을 Reset 함수로 분리하여 Reset을 통해 필요한 정보들만 초기화 하는 작업 완료.

ViewerPlayer에서 렌더링 정보가 초기화 될 수 있도록 작업 수정 완료

재시작 시 HP바 동기화가 되지 않는 버그 수정 완료.