|  |  |  |  |
| --- | --- | --- | --- |
| 씬 | 보내는 것 | 주기 | 이동 |
| 메뉴 | 레디정보 | 매 프레임 | Client -> Server |
| 씬 정보 | 매 프레임 | Server -> Client |
| 플레이어 번호 | 씬 정보가 바뀔 때 1회 | Server -> Client |
| 인 게임 | 키 입력 | 매 프레임 | Client -> Server |
| 플레이어 정보 | 매 프레임 | Server -> Client |
| 적 정보 | 매 프레임 | Server -> Client |
| 적 총알 정보 | 매 프레임 | Server -> Client |
| 아이템 정보 | 매 프레임 | Server -> Client |
| 플레이어 총알 | 매 프레임 | Server -> Client |
| 게임오버 | 점수 및 랭크 | 게임이 끝날 경우 1회 | Server -> Client |

데이터 전송 주기

* 매 프레임은 30프레임으로 한다.

서버

Object의 좌표를 얻는 함수

POINT CGameObject::GetPos()

int CGameObject::GetXPos()

int CGameObject::GetYPos()

Object의 Hp를 얻는 함수

int CGameObject::GetHp()

Object의 크기를 얻는 함수

int CGameObject::GetSize()

Object의 Type을 얻는 함수

int CGameObject::GetType()

Object의 Alive값을 얻는 함수

bool CGameObject::GetAlive()

Object의 좌표를 설정하는 함수

void CGameObject::SetPos(int x, int y)

void CGameObject::SetXPos(int x)

void CGameObject::SetYPos(int y)

Object의 Hp를 설정하는 함수

void CGameObject::SetHp(int hp)

Object의 크기를 설정하는 함수

void CGameObject::SetSize(int size)

Object의 Type을 설정하는 함수

void CGameObject::SetType(int type)

Object의 Alive값을 설정하는 함수

void CGameObject::SetAlive(bool life)

Bullet이 생성될 때 좌표와 타입에 따라 다르게 생성

CBullet(POINT pos, int typeNum)

Bullet의 Type값을 얻는 함수

int CBullet::getType()

Bullet과 적이 충돌했는지 체크하는 함수

bool CBullet::IsCrashtoEnemy(CGameObject enemy)

Bullet과 플레이어가 충돌했는지 체크하는 함수

bool CBullet::IsShootPlayer(PlayerInfo player)

Item이 생성될 때 아이템 정보에 따라 다르게 생성

CItem(ItemInfo itemInfo)

Item의 방향을 설정하는 함수

void CItem::SetDir(char c, bool b)

Item의 방향을 얻는 함수

bool CItem::GetDir(char c)

Item의 인덱스를 얻는 함수

int CItem::GetIndex()

Item의 플레이어가 아이템을 얻었는지 체크하는 함수

bool CItem::IsGetItem(PlayerInfo player)

Monster가 생성될 때 적 정보에 따라 다르게 생성

CMonster(EnemyInfo enemyInfo)

Monster의 안의 적이 가지고 있는 총알 벡터

vector<CBullet> m\_EnemyBullet

Monster의 방향을 설정하는 함수

void CMonster::SetDir(char c, bool b)

Monster의 방향을 얻는 함수

bool CMonster::GetDir(char c)

Monster의 인덱스를 얻는 함수

int CMonster::GetIndex()

Monster의 인덱스를 설정하는 함수

void CMonster::SetIndex(int index)

랭크파일을 열어 랭크 벡터에 저장하는 함수.

istream&ReadInputFile(istream& in, vector<Score>& vec)

스레드 함수. 모든 정보 처리를 이 함수에서 함

DWORD WINAPI ProcessClient(LPVOID arg)

들어오는 클라이언트 순서로 플레이어의 정보 초기값을 설정해줌

void Server::SetInitData(PlayerInfo& a, int num)

모든 클라이언트들이 레디했는지 확인하는 함수

bool Server::IsAllClientReady()

플레이어의 좌표 등 바뀌는 정보를 송신

void Server::SendAllPlayerInfo( PlayerInfo P[])

적 생성 함수

void Server::MakeEnemy()

아이템 생성 함수

void Server::MakeItem()

스킬 – 적 간 충돌체크

void Server::SkillCollision(vector<CMonster> &Target)

플레이어 – 아이템 간 충돌체크

void Server::CheckItembyPlayerCollision( vector<CItem>& item, PlayerInfo& player)

플레이어 총알 – 적 간 충돌체크

void Server::CheckEnemybyPlayerBulletCollision(vector<CBullet> &Bullet, vector<CMonster> &Target)

적 총알 – 플레이어 간 충돌체크

void Server::CheckPlayerbyEnemyBulletCollision(vector<CBullet>Bullet, PlayerInfo& player)

이전에 저장된 랭킹에 현재 정보 삽입 및 정렬 함수

void Server::SetRank(ofstream &out, vector<Score>& vec, Score temp)

지난 로우레벨에서 불필요해서 제거한 함수들

~~void SetReady()~~ **~~– 플레이어의 ready 정보를 토대로 레디 상태를 설정하는 함수~~**

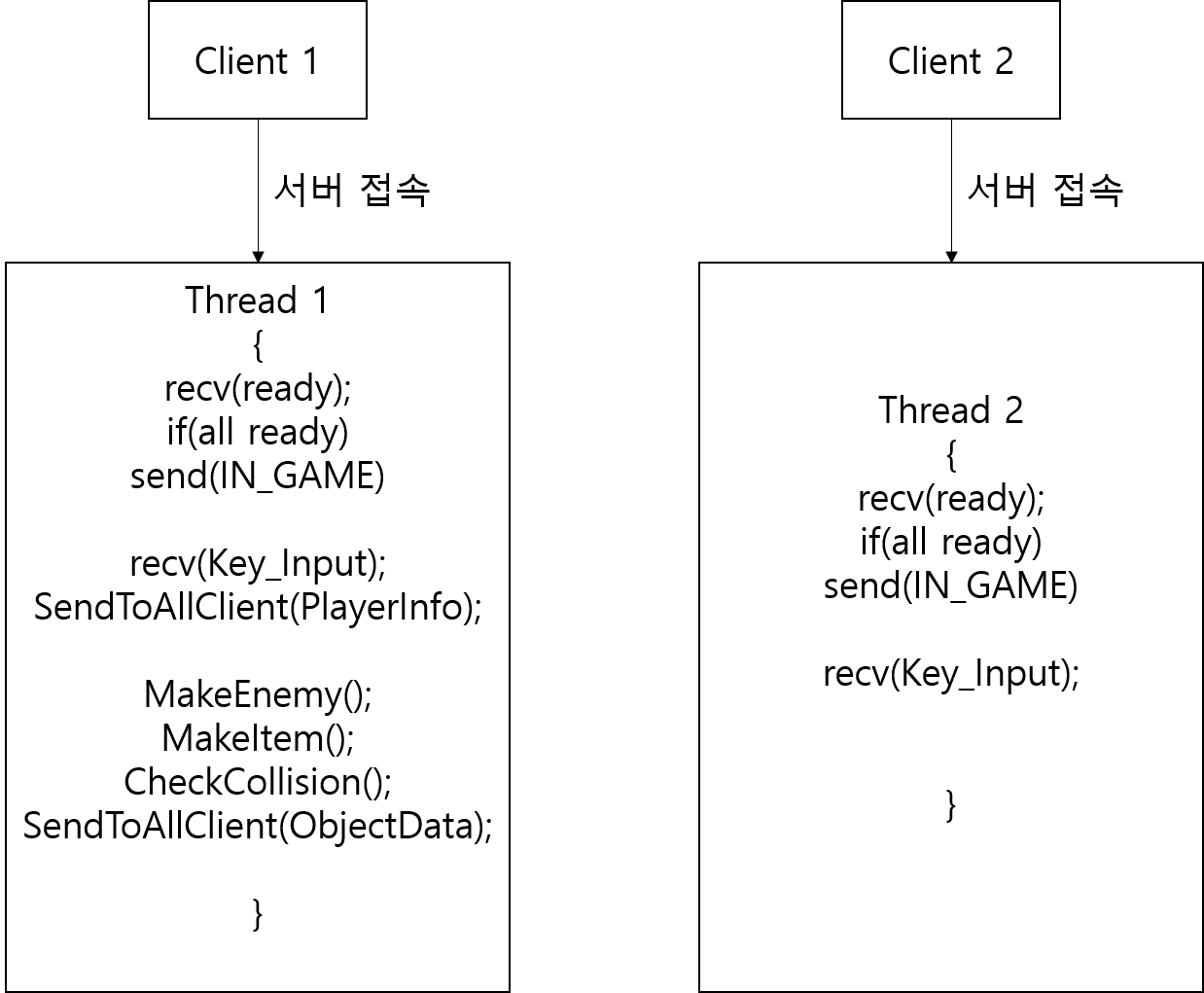
~~void RecvInitData(ClientInfo)~~ **~~- ready 정보 받고 ClientInfo 의 상태를 바꾼다~~**

~~void RecvInGameData(PlayerInfo, EnemyInfo)~~ **~~-게임 실행 중 필요한 데이터(아이템 획득 여부, 적 죽음 여부, 점수 등) 수신한다.~~**

~~void SendInGameData(PlayerInfo, EnemyInfo)~~ **~~-게임 실행 중 필요한 데이터(아이템 획득 여부, 적 죽음 여부, 점수 등) 송신한다.~~**

~~bool IsAllPlayerDead(int Client1PHp, int Client2PHp)~~ **~~-모든 플레이어가 죽으면 게임을 종료한다.~~**

~~void SendRankData()~~ **~~-10위권 내의 정보를 클라이언트에게 전송한다.~~**



동기화와 공유자원 문제

한 공유자원에 두개 이상의 쓰레드가 접근하면 문제가 생긴다. 그래서 아예 한 쓰레드에서 모든 데이터를 관리하는 방법으로 해결했다.

Client 2

Thread 2

{

recv(ready);

if(all ready)

send(IN\_GAME)

recv(Key\_Input);

}

서버 접속