경원 개발할 영역 씬 매니저 생성, 타이틀 씬 제작, 매치메이킹 버튼 생성, 클라이언트 send/recv, send/recv 처리

용주 개발할 영역 네트워크 매니저, 자료구조 관리 (수정됨)

승호 개발할 영역 쓰레드 매니저, 서버 send/recv(callback?) 처리

[클라이언트]

NetworkManager => Singleton

Variable:

SOCKET socket

Recv buffer

Send buffer

Function:

Init(): 게임 Init시 호출

윈속 초기화 / 소켓 생성 / connect 시도

성공 시 send()를 통해서 CS\_MATCHMAKING 전송

비동기 recv 호출 => CALLBACK ::processPacket();

Send( const int packet\_id )

받아온 패킷의 id에 따라서 패킷을 보낸다.

Process(const int packet\_id)

SC\_MATCHMAKING\_PACKET: 매칭 완료 패킷을 받아 타이틀 화면에다가 알림 창 띄우고 씬 로드 및 전환

나머지는 후술할 PACKET의 역할에 맞게 패킷 처리

~ NetworkManager():

소켓 close / 윈속 클리어

TimerManager

Variable

EPOCH\_TIME

now\_time

delta\_time

Function:

getDelta()

델타값 반환

useDelta()

1/60 초 이상이면 그만큼 delta\_time 깎고 진행 (렌더링 과정, 동기화 과정)

::ProcessPacket()

Recv 패킷을 NetworkManager::Process(id)에서 처리

[서버] :

로비스레드 생성 <- 스레드 매니저에서 하기

accept

queue에 받은 소켓 푸시

로비스레드 중임------------------------

queue에 소켓이 있는지 확인 -> 있다면 recv 바인드

recv callback() 불리면 -> 매치메이킹 자료구조에 푸시 (매칭 잡는 중이라는 뜻)

자료구조에 2명이상 있으면 게임 스레드 생성 후 일임 <- 스레드 매니저에서 하기

게임스레드 중임----------------------------

recv() 바인드 <- 네트워크 매니저에서 처리?

send() 바인드 <- 네트워크 매니저에서 처리?

업데이트 진행 (1/60초 마다 업데이트 후 동기화) <- 타이머를 이용해서

NetworkManager

Variable:

용주 니가짜 ㅅㅂ

ThreadManager

Variable:

lobby thread 생성해줘

vector<thread> 게임 스레드들 관리

Timer

클라와 같음

~~struct CS\_LOGIN\_PACKET : public BASE\_PACKET~~

struct CS\_MATCHCATCH\_PACKET : public BASE\_PACKET

struct CS\_MOVE\_PACKET : public BASE\_PACKET

struct CS\_FIRE\_PACKET : public BASE\_PACKET

struct SC\_MATCHMAKING\_PACKET : public BASE\_PACKET

struct SC\_MOVE\_FAULT\_PACKET : public BASE\_PACKET

struct SC\_MOVE\_PACKET : public BASE\_PACKET

struct SC\_FIRE\_PACKET : public BASE\_PACKET

struct SC\_GUN\_UPDATE\_PACKET : public BASE\_PACKET

struct SC\_LIVE\_UDPATE\_PACKET : public BASE\_PACKET

struct SC\_GAMEOVER\_PACKET : public BASE\_PACKET