**네트워크 게임 프로그래밍**

**Term project**

**[추진계획서]**

|  |  |  |
| --- | --- | --- |
| 문서명 | : | 추진계획서 |
| 팀장 | : | 임해인(2017184028) |
| 팀원 |  | 김경욱(2016182005)  전하은(2019182034) |
| 최초작성일 | : | 20221013 |
| 최종수정일 | : | 202201106 |
| 비고 | : | Ver. 2.0 |

목차

[1. 게임 개요 3](#_Toc118664113)

[1. 게임 소개 3](#_Toc118664114)

[2. 규칙 설명 3](#_Toc118664115)

[3. 기획 수정사항 4](#_Toc118664116)

[4. 씬 구성 5](#_Toc118664117)

[2. High Level 6](#_Toc118664118)

[~~1.~~ ~~쓰레드 구성(ver 1.0)~~ 6](#_Toc118664119)

[1. 쓰레드 구성(ver. 2.0) 6](#_Toc118664120)

[2. Scene 기준 플로우 차트 7](#_Toc118664121)

[3. Low Level 12](#_Toc118664122)

[1. 식별용 열거형 12](#_Toc118664123)

[2. 구조체 12](#_Toc118664124)

[3. 구현 필요 함수 15](#_Toc118664125)

[4. 팀 정보 17](#_Toc118664126)

[1. 팀원 역할분담 17](#_Toc118664127)

[2. 팀 정보 17](#_Toc118664128)

[5. 개발환경 17](#_Toc118664129)

[6. 개발일정 18](#_Toc118664130)

[3. 주요 분기점 18](#_Toc118664131)

[4. 임해인 19](#_Toc118664132)

[5. 김경욱 20](#_Toc118664133)

[6. 전하은 21](#_Toc118664134)

# 게임 개요

## 게임 소개

임해인(윈도우 게임 프로그래밍 2018) 텀 프로젝트 작업물

텍스트, 클립아트이(가) 표시된 사진

자동 생성된 설명

~~2~~ 3명의 플레이어가 최후의 순간 금화를 들고 있기 위해

치열한 싸움을 하며 코인을 사수하고,

서로 뺏고 뺏기는 다이나믹 한 플레이가 있는 게임

## 규칙 설명

1. **타이머가 0**이 되는 순간 **코인을 가지고 있는 유저가 승리**합니다.
2. **코인**을 **들고** 있으면 **이동속도가 감소**합니다.
3. **스킬, 데쉬, 공격**을 활용하여 상대방의 **코인을 뺏거나**, 코인을 들고 **도망**갈 수 있습니다.

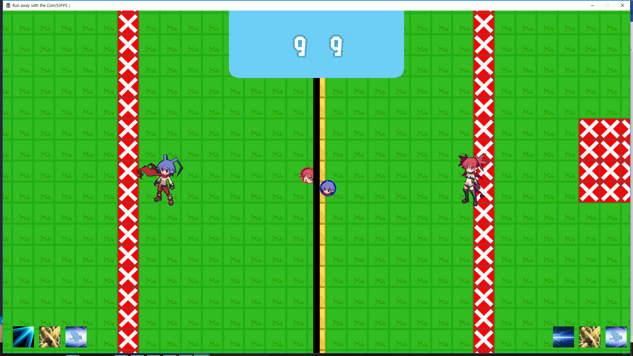
## 기획 수정사항

### 개요

이 파트는 기존 프로젝트(윈도우 프로그래밍) 텀 프로젝트를 네트워크 게임 프로그래밍 텀 프로젝트로 활용하는 과정에서 발생하는 기존 프로젝트에서의 수정사항을 다루는 부분입니다.

### 내용

1. 기준 플레이 인원 2인 -> 3인
   * 기존 프로젝트의 네트워크 환경 구현 간 난이도 조정을 위한 3인 플레이로 변경
2. 클라이언트 2분할 뷰 -> 전체화면 뷰
   * 한 개의 기기에서 동시에 플레이가 아닌 네트워크 플레이기 때문에, 2분할에서 전체화면으로 변경



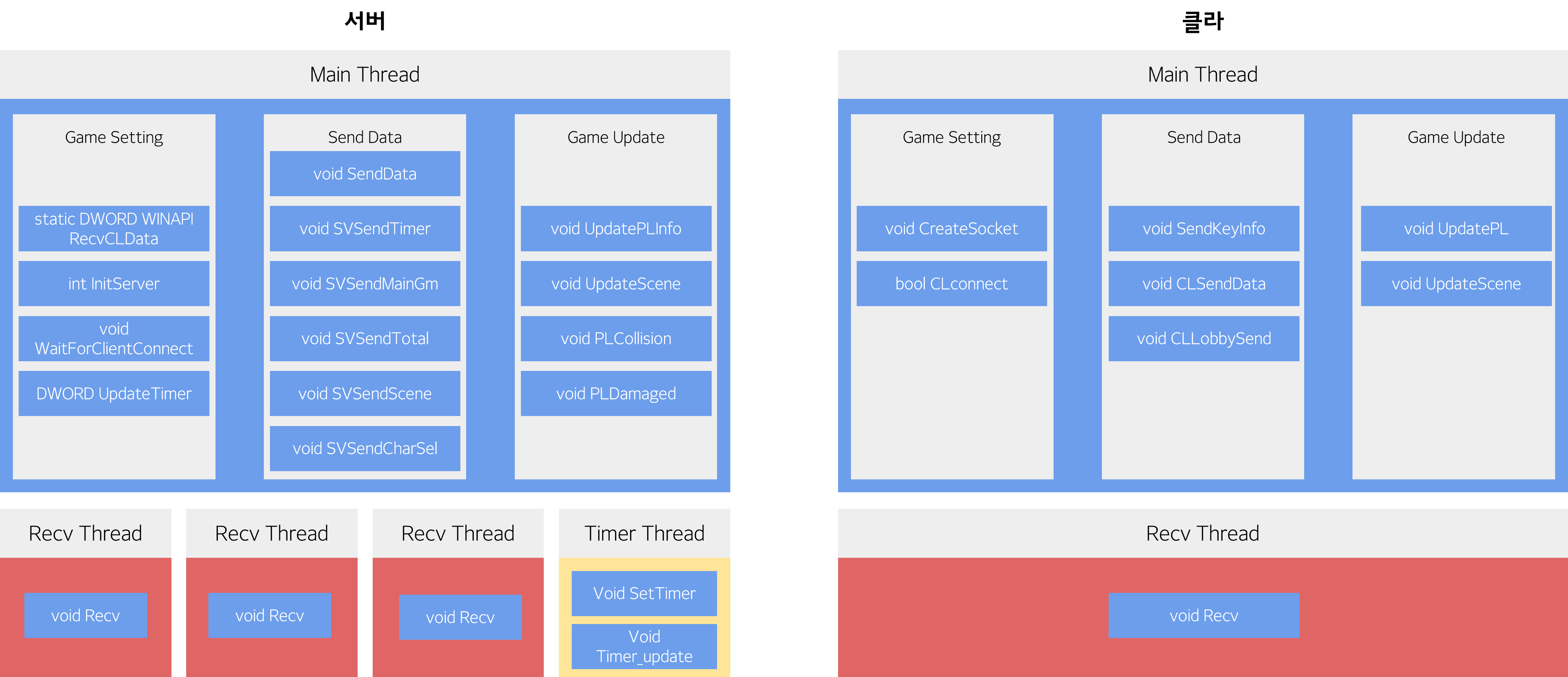
1. LOBBY 내 다른 유저 접속 정보 추가
   * 로비 내에 다른 유저들이 몇 명이 접속했고, Ready 버튼을 눌렀는지 확인할 수 있는 정보 제공 기능 추가
2. 캐릭터 선택 창 3인 선택창으로 수정
3. 게임 룰 소개 창 별도 페이지로 분리

## 씬 구성

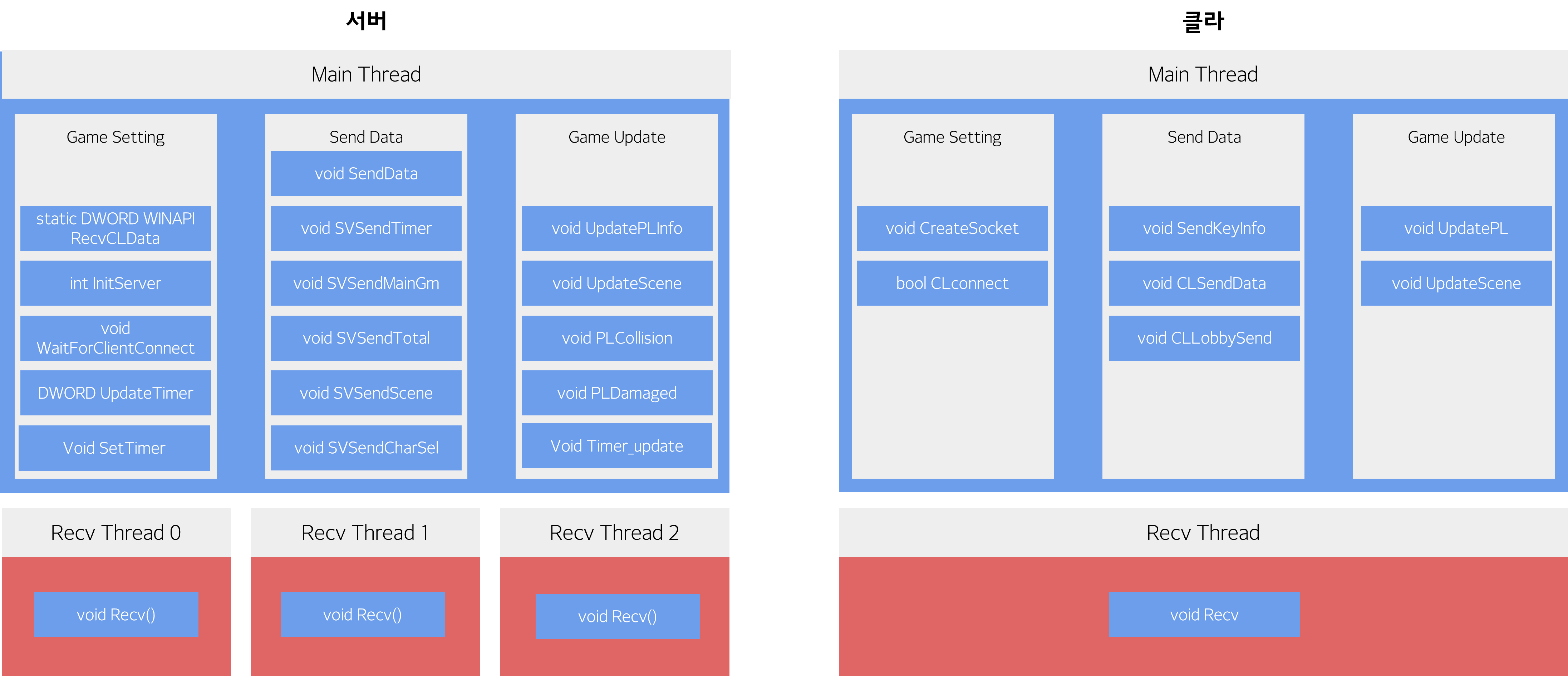
|  |  |  |
| --- | --- | --- |
| 이름 | 이미지 | 기능 |
| LOBBY | 텍스트, 책이(가) 표시된 사진  높은 신뢰도로 생성된 설명 | * 게임 타이틀 표시 * 유저 접속 정보 표시 * 룰 북 팝업 기능 제공 |
| Character  Select | 나무, 사진이(가) 표시된 사진  매우 높은 신뢰도로 생성된 설명 | * 캐릭터 선택 기능 * 타 유저 캐릭터 선택 실시간 업데이트 * 타이머 제공 * 랜덤 선택 기능 제공 |
| InGame |  | * 메인 게임 콘텐츠 제공 * 게임 종료 시 승리 및 패배 반응 제공 |

# High Level

## ~~쓰레드 구성(ver 1.0)~~

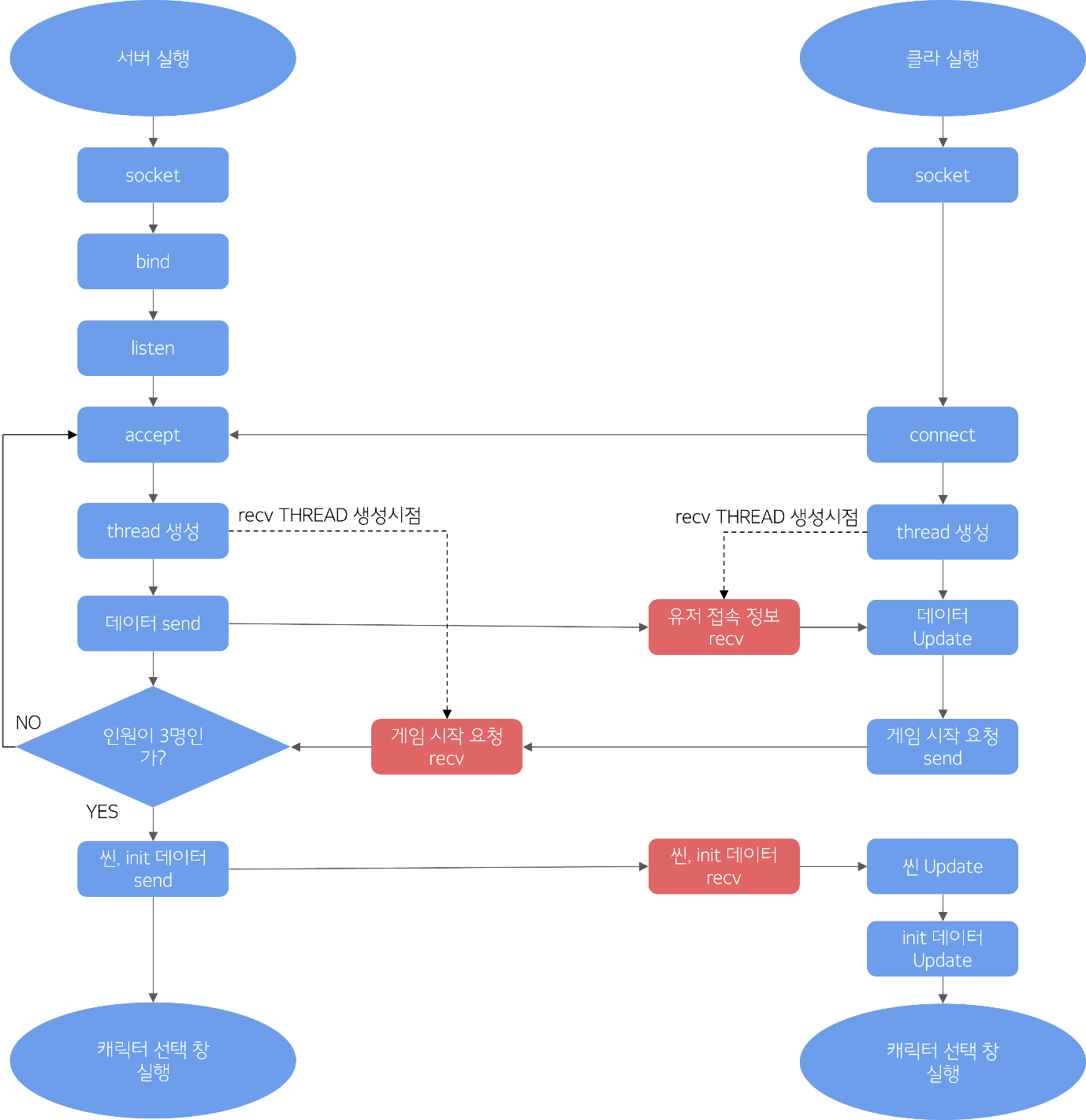


## 쓰레드 구성(ver. 2.0)

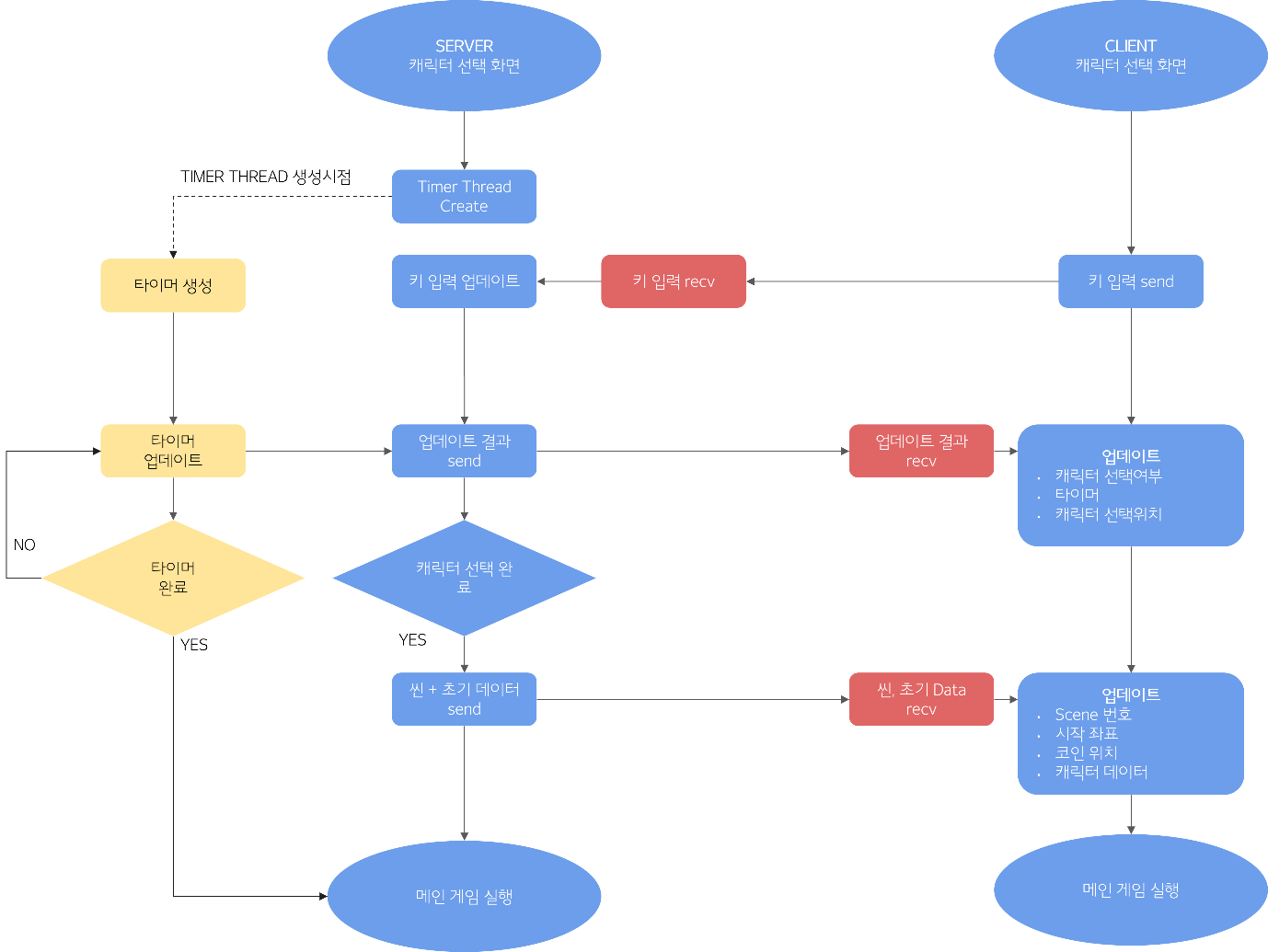


## Scene 기준 플로우 차트

### LOBBY



### ~~Character Select (ver 1.0)~~

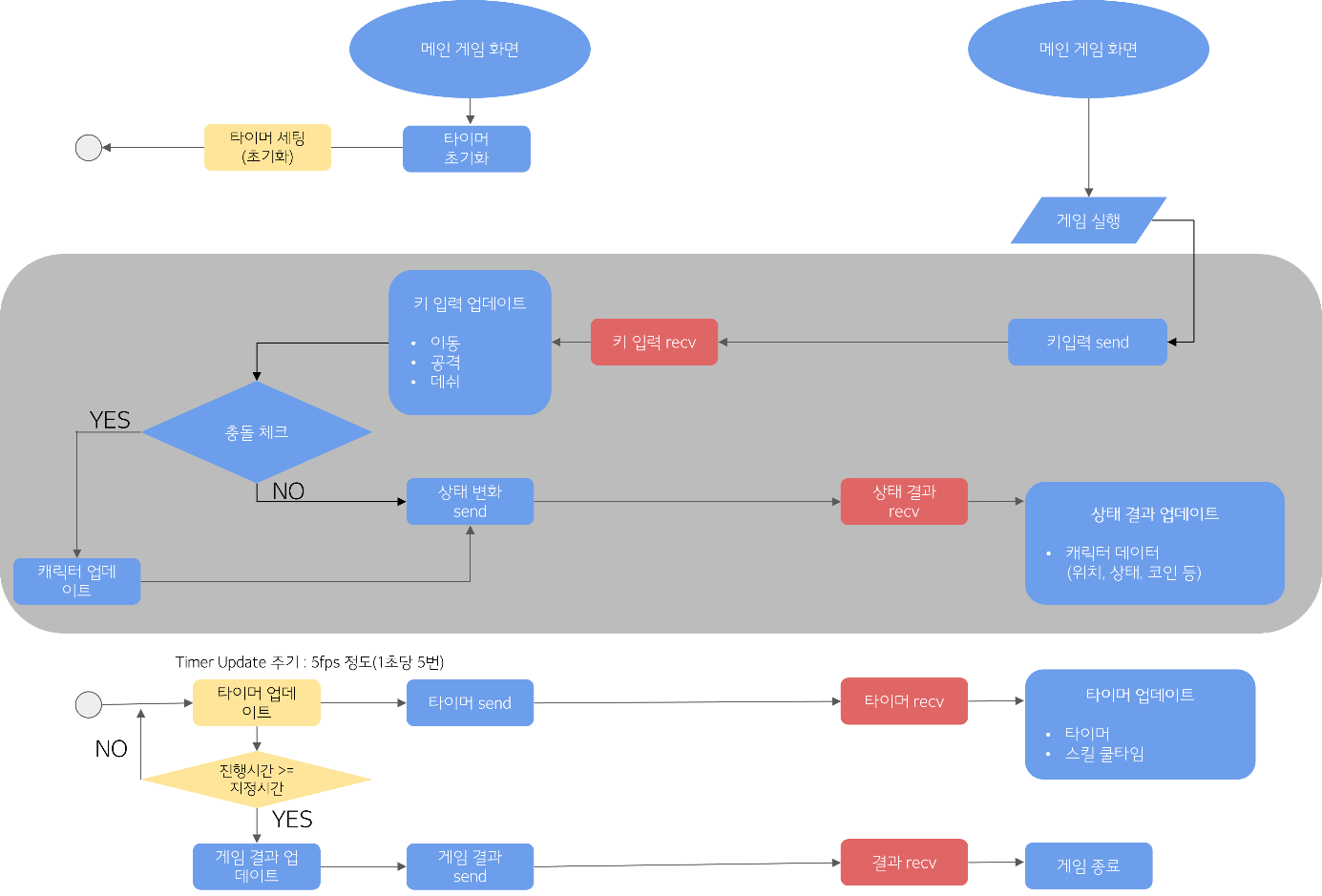


### Character Select (ver 2.0)

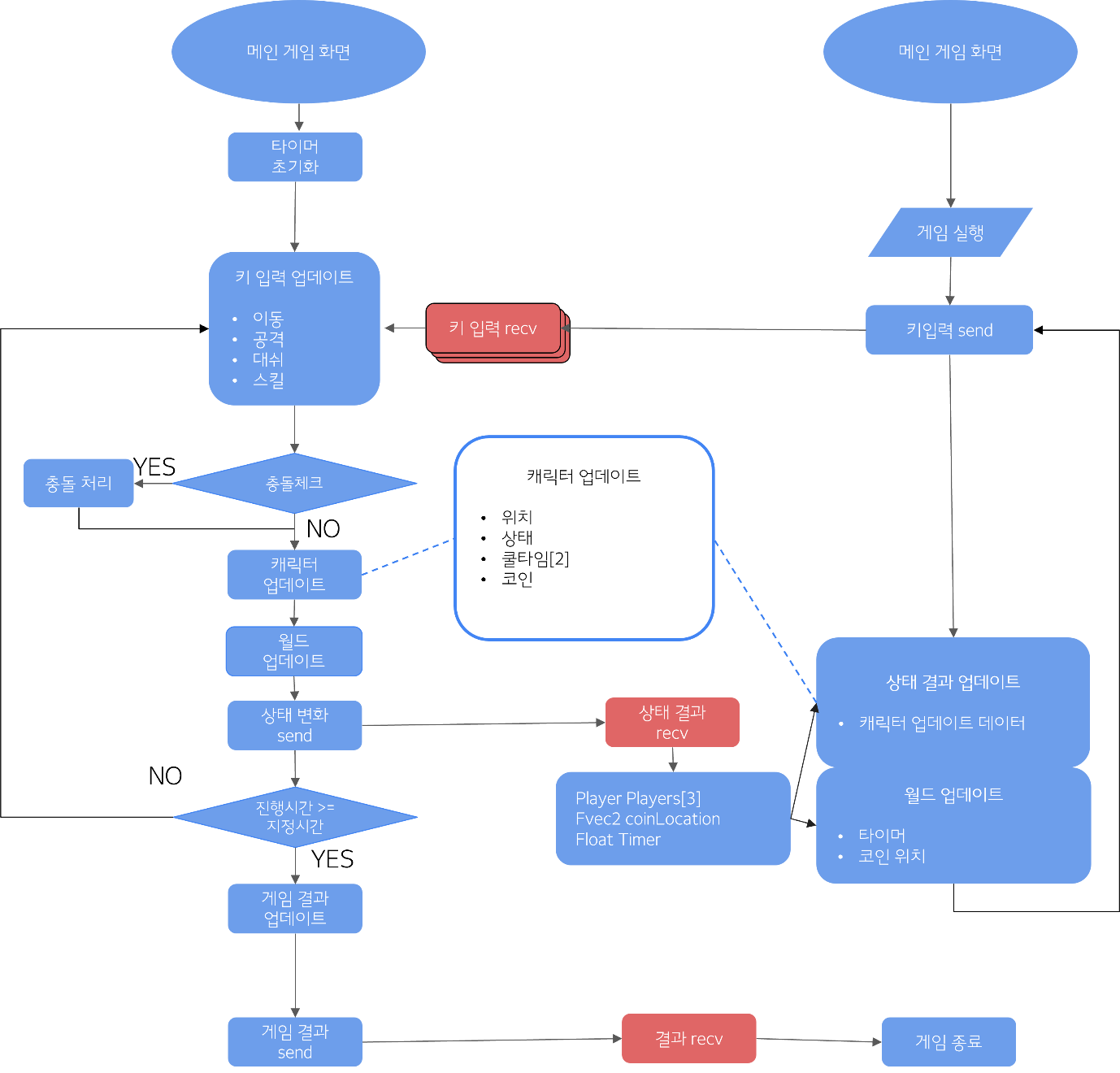
텍스트, iPod, 벡터그래픽이(가) 표시된 사진

자동 생성된 설명

### ~~InGame(ver 1.0)~~



### InGame(ver 2.0)



# Low Level

## 식별용 열거형

**enum** **Scene** {

Lobby,

Char\_sel,

Main\_game,

End\_game

}

**enum** **CharState** {

Idle,

Walk,

Attack,

Attacked,

Dash,

Skill,

Win,

Lose,

}

**~~enum~~****~~ProtocalInfo~~**~~{~~ *~~//현재 전송하는 정보~~*

~~Timer,~~

~~PlayerData,~~

~~Scene\_Data~~

*~~//….~~*

~~}~~

*//ver 2.0 : add*

**enum** **SC\_ProtocalInfo** { *//현재 전송하는 정보*

SC\_Ingame\_send,

SC\_lobby\_send,

SC\_scene\_send,

SC\_LobbytoCharsel\_init,

SC\_CharseltoIngame\_init,

SC\_IngametoFinish\_init,

SC\_FinishtoLobby\_init,

}

*//ver 2.0 : add*

**enum** **CS\_ProtocalInfo** { *//현재 전송하는 정보*

CS\_ingame\_send, *//or CS\_ingame\_send\_temp*

CS\_lobby\_send,

}

## 구조체

**struct** **fvec2**{ *//플레이어 좌표 구조체*

float x;

float y;

}

**~~struct~~****~~Gtimer~~**~~{~~

~~float \_left\_time;~~

~~float \_skill\_cooltime1;~~

~~float \_skill\_cooltime2;~~

~~}~~

**~~struct~~****~~player\_data~~**~~{~~

~~int \_char\_type;~~

~~fvec2 \_location;~~

~~int \_look;~~

~~int \_state;~~

~~bool \_coin;~~

~~}~~

*//ver 2.0 : add*

**struct** **Character\_data** {

int \_char\_type;

fvec2 \_location;

int \_look;

int \_state;

bool \_coin;

float \_skill\_cooltime1;

float \_skill\_cooltime2;

}

*//ver 2.0 : add*

**struct** **SC\_Ingame\_send** { *//프레임마다 전송하는 데이터*

Character\_data \_player[3];

fvec2 \_coin\_location;

float \_left\_time;

}

*//ver 2.0 : edit*

**struct** **SC\_lobby\_send**{ *//새로운 acc 있을 때마다 보내줌*

int \_acc\_count;

~~int \_my\_num;~~

}

*//공통*

**struct** **SC\_scene\_send**{ *// 씬데이터*

int \_scene\_num;

int \_protocol\_num;

}

*// 클라이언트 초기화할 때 사용하는 구조체들*

**struct** **SC\_LobbytoCharsel\_init**{ *//로비 -> 캐 선택*

int \_scene\_num;

int \_protocol\_num;

int \_char\_num; *//접속한 순서대로 플레이어에게 부여되는 플레이어 넘버*

int \_char\_type;

bool \_ready;

}

**struct** **SC\_CharseltoIngame\_init** { *//캐선택 -> 인게임*

int \_scene\_num;

int \_protocol\_num;

Gtimer \_timer; *//타이머, 스킬 쿨타임 초기화*

fvec2 \_location;

int \_state;

bool \_coin;

}

**struct** **SC\_IngametoFinish\_init**{ *//인게임 -> 겜종료*

int \_scene\_num;

int \_protocol\_num;

int \_CharState;

}

**struct** **SC\_FinishtoLobby\_init**{ *//겜종료 -> 로비*

int \_scene\_num;

int \_protocol\_num;

}

**struct** **CS\_ingame\_Send**{

int vkey; *// GetAsyncKeyState(vkey)로 동시키입력 여부를 확인하기 위한 변수*

SHORT pressedVkey;

}

**struct** **CS\_ingame\_send\_tmp**{*// GetAsyncKeyState(vkey)로 동시키입력이 동작X시 사용*

short \_horizontal\_key; *// -1 : left || 0 : NULL || 1 : right*

short \_vertical\_key; *// -1 : down || 0 : NULL || 1 : up*

short \_skill\_key; *// 0 : NULL || 1 : skill || 2 : attack || 3 : dash*

}

**struct** **CS\_lobby\_send** {

CS\_ingame\_send \_input;

bool \_ready;

}

## 구현 필요 함수

|  |  |
| --- | --- |
| **서버** | |
| int InitServer() | 소켓을 생성한다 |
| void SendData(~~player\_data&~~) | 클라이언트에게 플레이어 데이터를 전송한다. |
| void WaitForClientConnect() | 접속한 순서대로 클라이언트에게 넘버를 부여하고 클라이언트가 3명이 모두 접속 했는지 확인한다. |
| **static** DWORD WINAPI RecvCLData (LPVOID) | 클라이언트로부터 데이터를 받고 업데이트한 정보를 클라이언트에 전송한다. |
| ~~void SVSendTimer()~~ | ~~타이머(탑 화면의 시간, 스킬 쿨타임)를 전송한다.~~ |
| ~~DWORD UpdateTimer()~~ | ~~타이머 업데이트~~ |
| void SVSendTotal() | 현재 전송하는 Protocal Info와 접속한 Player Data를 전송한다. |
| void SVSendScene() | 다음 넘길 Scene을 전송한다. |
| void SVSendLobby() | 로비에서 필요한 데이터를 넘긴다 |
| void SVSendCharSel() | 캐릭터 선택창에서의 플레이어 데이터(선택한 캐릭터, Ready 여부)를 전송한다. |
| void SVSendMainGm() | 메인 게임 내에서 플레이어 데이터(선택한 캐릭터, 위치, 방향, 상태, 코인 여부) 전송한다. |
| void UpdatePLInfo() | 플레이어 데이터(상태, 좌표값, 방향, 코인여부)를 업데이트한다. |
| void UpdateScene() | Scene 정보를 업데이트한다. |
| void PLCollision() | 플레이어와 플레이어, 플레이어와 코인, 플레이어와 장애물 사이의 충돌여부에 따라 플레이어 상태를 업데이트한다. |
| void PLDamaged() | 스킬 키를 입력이 들어왔을 때 플레이어가 스킬에 의한 데미지를 입었는지 검사하고 플레이어 상태를 업데이트한다. |

|  |  |
| --- | --- |
| **클라이언트** | |
| void CreateSocket() | 소켓을 생성한다 |
| bool CLConnect() | 네트워크에 연결하고 연결 성공 유무에 따라 값을 반환한다. |
| void CLSendData (LPVOID) | 서버로 패킷을 전송한다. (각 Scene로 오버라이딩)  (Lobby, Char\_sel, ingame, end\_game) |
| ~~void CLLobbySend(CS\_lobby\_send )~~ | ~~로비에서의 데이터(레디유무,선택캐릭터)를 서버에게 보낸다.~~ |
| ~~void SendKeyInfo(CS\_ingame\_send& )~~ | ~~입력한 키값을 서버에게 보낸다.~~ |
| ~~void RecvTimer()~~ | ~~탑 화면 시간과 스킬 쿨타임 정보를 서버로부터 받는다.~~ |
| void UpdatePL(player\_data ) | 서버로부터 넘겨받은 플레이어 정보로 해당 값을 업데이트한다. |
| void UpdateScene() | Scene을 업데이트한다. |

# 팀 정보

## 팀원 역할분담

임해인 [ 팀장 ]

* 클라이언트 네트워크
* 클라이언트 처리

김경욱

* 서버 멀티 쓰레드
* 서버 소켓 프로그래밍

전하은

* 서버 멀티 쓰레드
* 서버 소켓 프로그래밍

## 팀 정보

깃허브 주소

* <https://github.com/haein0303/NetWorkGame_Final_Term>

# 개발환경

개발 환경 버전 정보

|  |  |
| --- | --- |
| **항목** | **버전** |
| Win Sdk ver | 10.0.22000.0 |
| Visual Studio 2022 | v143 |
| Std Ver | C++ 17 |

# 개발일정

## 주요 분기점

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  | 11/1 | 2 | 3 | 4 | 5 |
|  |  |  |  | 추진계획서  1차 검수 |  |  |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 계획서 작성 회의 | 추진계획서 완성 | 추진계획서  2차 검수 |  |  |  |  |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|  | 1차 회의 |  | 로비 네트워크 기능 확인가능 |  | 캐릭터 선택  네트워크 기능 확인 가능 |  |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|  | 2차회의 |  | 인게임 기초 네트워크 송수신 구현 완료 |  | 인게임 네트워크 기능 확인 가능 |  |
| 27 | 28 | 29 | 30 | 12/1 | 2 | 3 |
|  | 3차회의 |  |  |  | 예비일 / 최종 점검 가능하게 준비 |  |
| 4 | 5 | 6 | 7 | 8 |  |  |
|  |  | 문서화 |  |  |  |  |

## 임해인

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  | 11/1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |  |  |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  | [클라이언트]  Scene\_Lobby\_UI  수정 | [클라이언트]  FrameWork\_Recv 개발 | [클라이언트]  Scene\_Char\_select\_UI 수정 |  |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|  | [1차 자체 점검 및 회의] | [클라이언트]  Scene\_Char\_select\_UI 수정2 | [클라이언트]  Recv 쓰레드 분리 | [1차 예비일] | [클라이언트]  CLConnect() /  CLLobbySend()  제작 |  |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|  |  |  | [클라이언트]  UpdateScene()  제작 | [클라이언트]  RecvTimer()  - Char\_sel  - Ingame  적용 | [클라이언트]  CLSendData ()  제작 |  |
| 27 | 28 | 29 | 30 | 12/1 | 2 | 3 |
|  | [클라이언트]  UpdatePL(player\_data ) – 제작(위치) | [클라이언트]  UpdatePL(player\_data ) – 제작(상태) | [클라이언트]  UpdatePL(player\_data ) – 제작(애니메이션 동기화) |  | [클라이언트]  1차 점검 |  |
| 4 | 5 | 6 | 7 | 8 |  |  |
|  |  |  |  |  |  |  |
| 메모 | 작업 선호 요일 : 수, 목, 금, 일  작업 불가일 : 11월 20일 ~ 21일 | | | | | |

## 김경욱

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  | 11/1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |  |  |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  | [서버]  프레임워크 제작  /  통신 프로토콜  제작 |  | [서버]  타이머 thread /  InitServer()  제작 |  |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|  | [1차 자체 점검 및 회의] |  |  |  |  |  |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|  | [2 차 자체 점검 및 회의] |  | [서버]  WaitForClientConnect() /  제작 | [서버]  SendData() | [서버]  UpdateTimer() /  SVSendScene() 제작 | [서버]  SVSendTimer()  제작 |
| 27 | 28 | 29 | 30 | 12/1 | 2 | 3 |
|  |  | [서버]  SVSendCharSel() 제작 | [서버]  PLCollision()  제작 | [서버]  RecvCLData ()  제작 | [클라이언트]  1차 점검 |  |
| 4 | 5 | 6 | 7 | 8 |  |  |
|  |  |  |  |  |  |  |
| 메모 | 작업 선호 요일 : 수, 목, 금, 토, 일 | | | | | |

## 전하은

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 일 | 월 | 화 | 수 | 목 | 금 | 토 |
|  |  | 11/1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |  |  |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  |  |  | [클라이언트]  Scene\_ingame 분할화면 하나로 합치기 | [클라이언트]  Scene\_Igame\_input  수정 |  |  |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|  | [1차 자체 점검 및 회의]  1.개발진척확인  2.일정재조율  3.기능구현체크 |  |  | [클라이언트]  Scene\_Igame\_UI  네비게이터 수정 |  |  |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|  | [2 차 자체 점검 및 회의]  1. 개발진척확인  2. 기능구현체크 | [클라이언트]  SendKeyInfo()  제작 |  | [서버]  SVSendMainGm  UpdatePLInfo()  제작 |  |  |
| 27 | 28 | 29 | 30 | 12/1 | 2 | 3 |
|  | [3차 자체 점검 및 회의]  1. 최종검수 및 디버그 일정 조율  2. 문제 있을 경우 활용 기간 | [서버]  PLDamaged() |  | [클라이언트]  UpdatePL(player\_data ) – 제작(코인) |  |  |
| 4 | 5 | 6 | 7 | 8 |  |  |
|  |  |  |  |  |  |  |
| 메모 | 작업 선호 요일 : 월, 화, 목  예외일 : 7, 8, 16, 17 | | | | | |