네트워크 게임 프로그래밍  
 Term Project

**-알케미스트-**

2011182007 게임공학과 김영서

20131560 소프트웨어 공학과 민웅기

2013156037 소프트웨어 공학과 정낙현

2017.11.8

목차

애플리케이션 기획

개요

ui/ux 기획

요구사항 분석

High Level 디자인

전체 Flowchart

Client측면 Flowchart

Server측면 Flowchart

쓰래드 동기화

Low Level디자인

데이터 통신을 위한 구조체

Client 함수

Server함수

팀원 역할 분담

개발 환경

개발 일정

## 애플리케이션 기획

### 개요

게임 제목:

알케미스트

장르:

탄막슈팅 디펜스게임

컨셉:

사방의 랜덤 한 방향에서 적들이 출현하며 PC(Player Character)는 탄막을 발사하여 적을 제거한다.

이때 적을 처치하는게 늦어 필드에 일정 수치 이상의 적들이 소환되면 패배로 정한다.

High Score기록을 목표로 하는 게임이다.

규칙:

플레이어는 키보드를 조작하여 캐릭터조작을 제어할 수 있다.

적은 초기에 1초마다 랜덤한 방향의 외곽에서 생성되며 시간이 지날수록 생성주기가 빨라진다.

적이 필드에 100개 이상 존재하면 게임오버가 되고 High Score가 기록된다.

멀티플레이시 접속자수에 비례하여 몬스터 생성주기가 더 빨라진다.

멀티플레이시 모든 유저가 라이프를 잃으면 GameOver처리되며 팀 단위로 Leader보드에 기록된다.

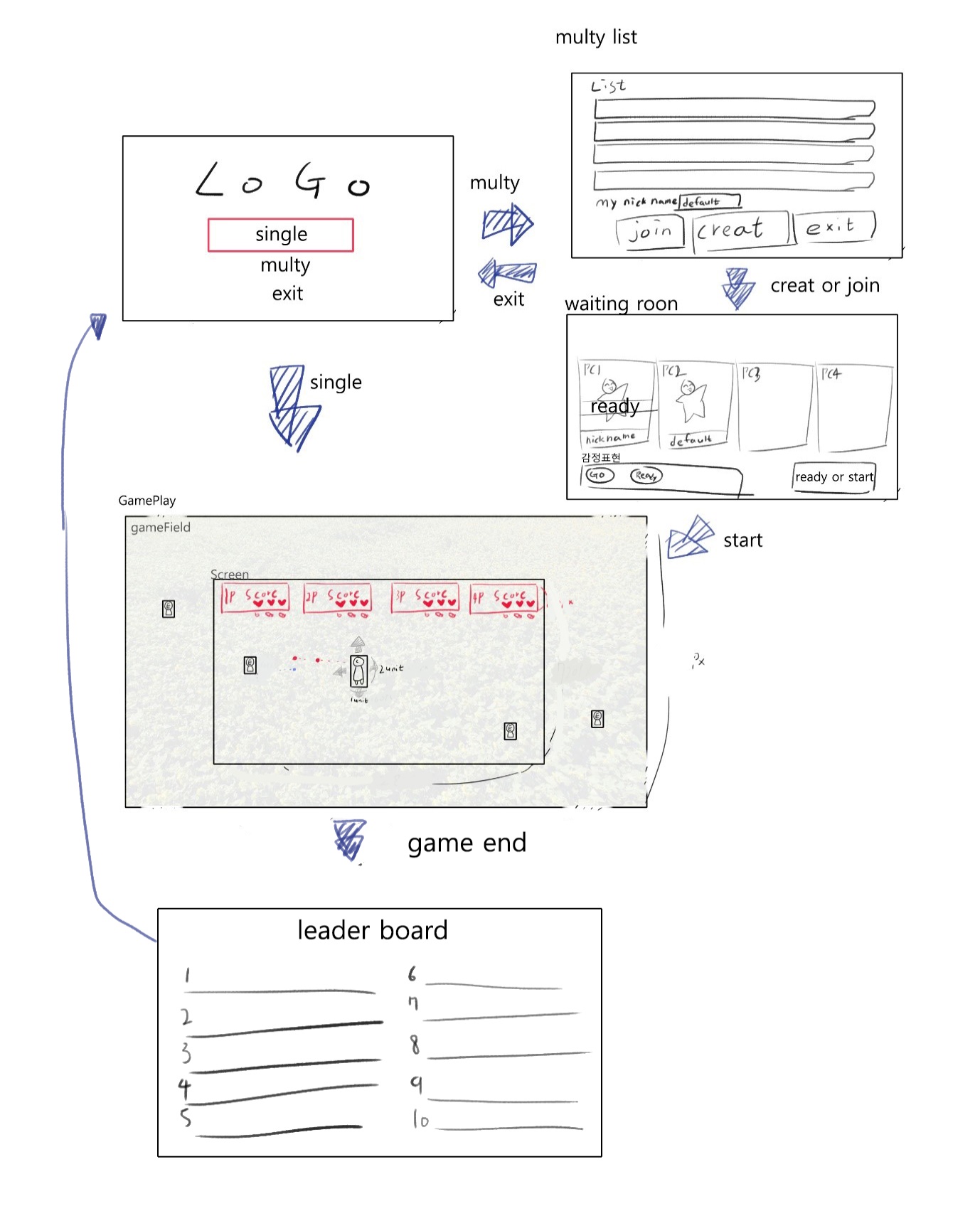
게임의 조작법:

상하좌우 방향키를 이용하여 캐릭터를 상하좌우로 이동시킬 수 있다.

z를 눌러 총알을 발사할 수 있다.

x를 눌러 폭탄은 사용할 수 있다.

### Ui/ Ux 기획서:



해상도: 1600\*900 (16:9 비율)

1unit: 1280 / 16\*1.5

필드크기: 해상도 \*2

캐릭터: 1\*2 unit

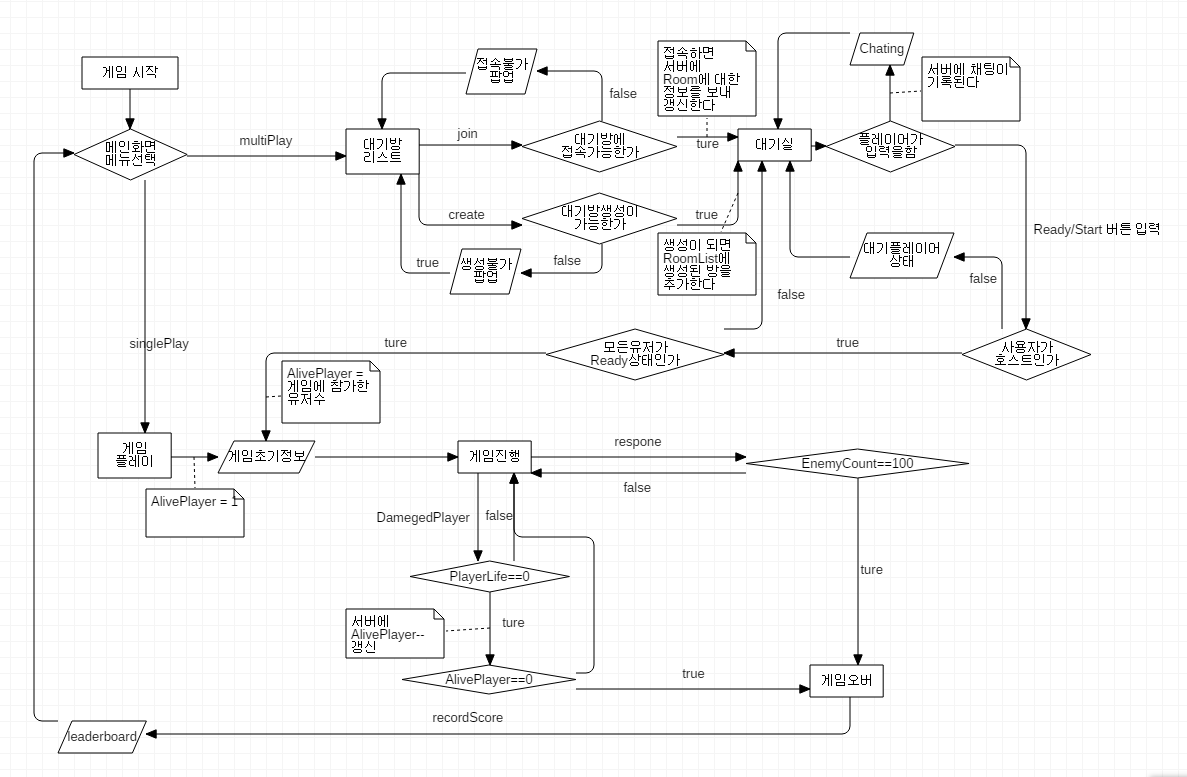
적: (2/3)\*1unit

### 요구사항 분석

* 통신 프로토콜은 경우에 따라 TCP와 UDP를 선택적으로 사용한다.
* 서버는 4개의방에 4명씩 총 16명의 클라이언트를 수용한다.
* 서버의 IP주소와 포트번호는 고정되어 있다고 가정하여 서버IP와 포트번호를 로컬에 text
* 파일을 읽어서 이용한다.
* 싱글플레이는 data를 client에서 모두 처리하지만 멀티플레이에서는 data를 서버로 옮겨 처리한후 같은 방의 player에게 보내준다.
* teamScore는 멀티 쓰레드에서 임계영역 방식을 사용하여 동기화한다.
* client와 server간 통신에서 프레임 주기에 대해 동기화 하기위해 이벤트를 사용한다.
* 수용인원 : 4개의 방에 4명씩 총 16명

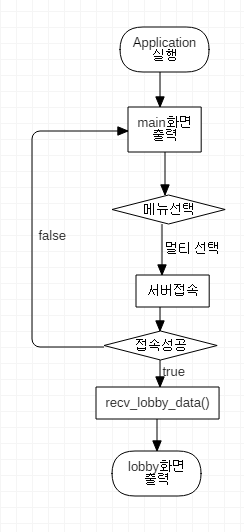
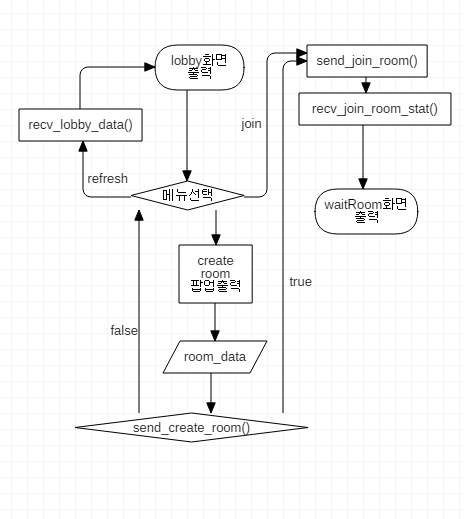
## High-level 디자인

### 전체 Flowchart

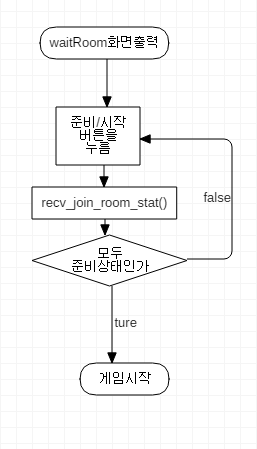
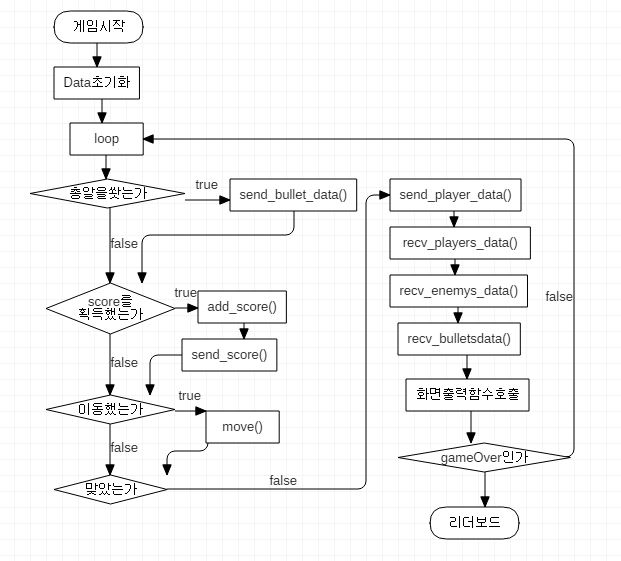


### client측면 Flowchart

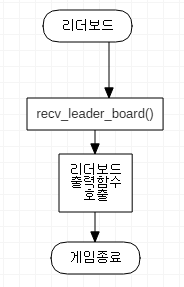
Application lobby

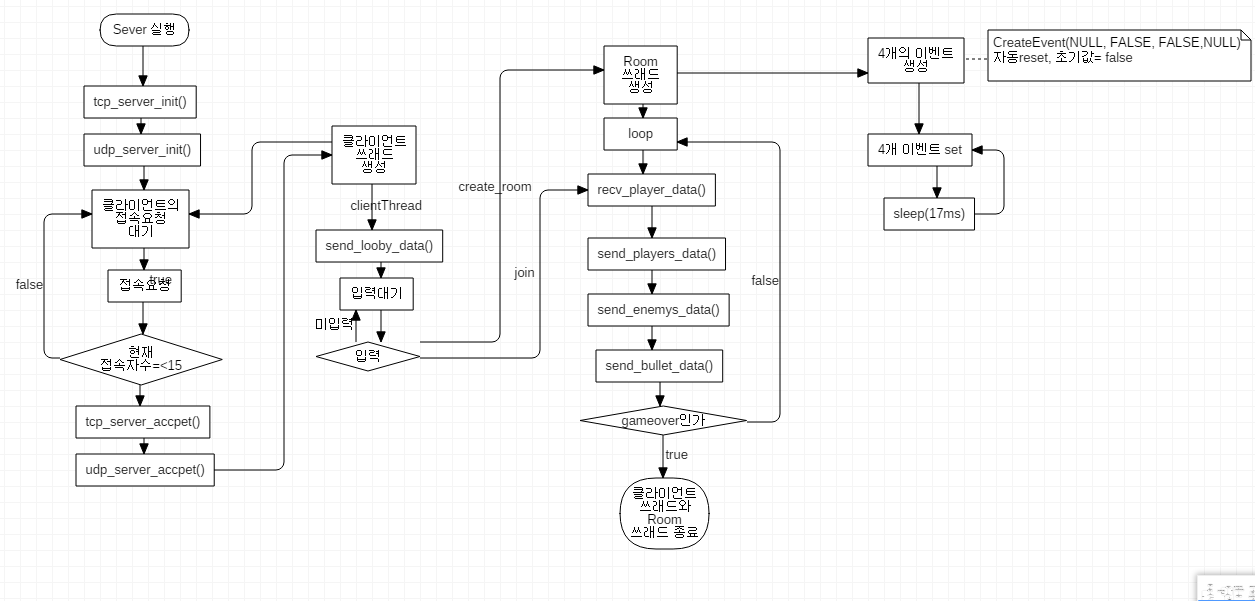
waitRoom gameStart

leaderBoard



### server측면 Flowchart

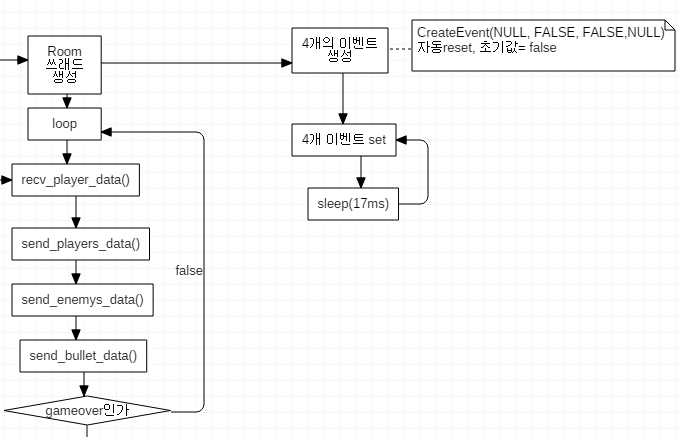


### 쓰레드 동기화

임계영역

TeamScore의 경우 클라이언트로부터 획득한 점수를 Server에서 합치는 것이므로 동시에 4개의 클라이언트가 1개의 변수에 접근하게 되므로 동기화에 문제가 없도록 임계영역으로 구현해준다.

쓰레드 이벤트



클라이언트에서 1초에 60번 주기로 데이터를 보내므로 서버도 1초에 60번 데이터를 받도록 해줘야 한다.

loop문에 wait\*를 걸어주면 한바퀴 돌때마다 자동reset때문에 멈추게 된다.

17ms마다 이벤트를 set함으로서 17ms마다 1번씩 loop문을 돌 수 있게 되어 1초에 60번에 데이터를 받게된다.

## Low-level 디자인

### 데이터 통신을 위한 구조체

struct Point

|  |  |  |  |
| --- | --- | --- | --- |
| type | 이름 | 설명 | 범위 |
| int | posX | x좌표 값 | Filed x축 넓이 |
| int | posY | y좌표 값 | Filed y축 넓이 |

struct PlayerData

|  |  |  |  |
| --- | --- | --- | --- |
| type | 이름 | 설명 | 범위 |
| byte | playerNumbering | 서로 다른 User가 조작하는 Player를 구분하기 위한 넘버링 | 1~16 |
| struct Point | playerPos | Player의 포지션 | - |
| byte | direction | Player가 바라보는 방향  12시방향부터 시계방향으로 4방위 | 1~4 |
| byte | life | Player의 남은 목숨 수 | 0~3 |
| byte | bomb | Player의 남은 폭탄 수 | 0~3 |
| bool | isDamaged | 일정시간내에 피해를 받았는지 여부 | true, false |

struct RoomData

|  |  |  |  |
| --- | --- | --- | --- |
| type | 이름 | 설명 | 범위 |
| byte | hostNumber | 방의 Leader인 User의 playerNumbering | 1~16 |
| String | roomName | 방의 이름 | 30자 내 |
| byte | fullPlayer | 방에 설정된 최대 인원 | 0~4 |
| byte | curPlayer | 현재 방에 접속한 인원 | 0~fullPlayer |
| bool | isStarted | 게임중 여부 | true, false |
| byte | readyPlayerCount | 준비중인 플레이어 수 | 0~3 |

struct Bullet

|  |  |  |  |
| --- | --- | --- | --- |
| type | 이름 | 설명 | 범위 |
| struct Point | start\_pos | 발사된 위치 | 필드내 |
| byte | direction | 총알이 향하는 방향 | 1~16 |
| byte | speed | 총알이 향하는 speed | 1~10 |
| float | shot\_time | 총알을 발사한 시간 |  |
| byte | shoter | 총알을 발사한 개체  1: player, 2: enemy |  |

List players\_data PlayerData의 집합 ListLenght =< 방의 최대 인원

List lobby\_data RoomData의 집합 ListLength =< 방의 개수

Client 디자인

### Client 함수

GameSys Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| input() | 사용자의 key조작 입력을 받아 move()를 호출하여 Player를 움직인다. | - | void |
| ui\_state\_draw() | player state ui를 그려준다. | int player수,  List player\_data, | void |
| enemys\_draw() | enemy를 그려준다. | int enemys\_data length,  List enemys\_data | void |
| players\_draw() | player를 그려준다. | List players\_data length,  List players\_data | void |
| add\_score() | score를 인자 값을 더한다 | int add\_val | int now\_score |
| damaged() | GameObject 개체가 damege를 입는다. player 혹은 enemy | GameObject 개체 | int left\_life |

TcpController Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| err\_display() | 서버와 연결 하던 중 문제가 발생할 경우 알림 창을 띄워 알려준다. | string msg | void |
| err\_quit() | 서버와 연결 하던 중 문제가 발생할 경우 로비로 돌아간다. | string msg | void |
| server\_tcp\_connect() | 윈속 초기화, 소켓 생성, 접속 시도 | string server\_ip,  int port, | void |
| lobby | | | |
| recv\_lobby\_data() | buf 에 서버로부터 roomData들의 정보를 받는다. | bufSize | buf |
| send\_create\_room() | room을 만들면 room에대한 정보를 보낸다. | string name,  byte full\_player,  struct PlayerData player\_data | void |
| send\_join\_room() | 서버에 인자로 준 방에 접속요청을 보낸다. | int roomNember | 성공: 1,  실패: ERRORCODE |
| wait\_room | | | |
| recv\_join\_room\_stat() | buf 에 서버로부터 join하려는 방의 상황을 받아온다 | bufSize | buf |
| send\_ready\_state() | ready여부를 서버에 보낸다. | bool is\_ready | void |
| send\_emotion\_chat() | 로비에서 감정표현 채팅을 보낸다  ->enum 감정표현들 | byte emotion | void |
| recv\_emotion\_chat() | buf 에 서버로부터 로비에서 감정표현 채팅을 수신한다 | bufSize | void |
| game | | | |
| send\_bullet\_data() | bullet을 발사하면 bullet 정보를 서버에 보낸다. | List Bullet | void |
| send\_score() | score를 획득 할 때마다 데이터를 보낸다.  ->(score) | int score | void |
| leader\_board | | | |
| recv\_leader\_board() | buf 에 서버로부터 leader board의 정보를 받는다. | bufSize | buf |

UdpController Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| err\_display() | 서버와 연결 하던 중 문제가 발생할 경우 알림 창을 띄워 알려준다. |  | void |
| err\_quit() | 서버와 연결 하던 중 문제가 발생할 경우 로비로 돌아간다. |  | void |
| server\_udp\_connect() | 윈속 초기화, 소켓 생성, 접속 시도  Client는 단 1개 Server만 통신하므로 write, read를 쓰기위해 |  | void |
| game | | | |
| send\_player\_data() | 프레임 마다 Player의 정보를 서버에 보낸다.  ->(player numbering, position, position, direct, life, bomb, isDamaged ) | struct PlayerData | void |
| recv\_players\_data() | 프레임 마다 buf 에 서버로부터 player들의 정보를 받는다 | buf\_size | buf |
| recv\_enemys\_data() | 프레임 마다 buf 에 서버로부터 enemy들의 정보를 받는다. | buf\_size | buf |
| recv\_bullets\_data() | 프레임 마다 buf 에 서버로부터bullet들의 정보를 받는다. | buf\_size | buf |

### Server 함수

ServerSys Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| get\_bullet\_pos() | 총알의 시작점, 속도, 시간을 통해 총알의 위치를 반환한다. | Bullet | stuct Point |
| collision\_check() | Bullet과 Enemy 혹은 Player와의 충돌을 체크한다. | List player, List bullet, List Enemy | 충돌하면 true |
| client\_thread() | 클라이어트가 접속을 요청할 때마다 연결해준다. |  | void |
| room\_thread() | 호스트 클라이언트가 방을 만들때마다 동기화기능이 있는 쓰래드를 만들어준다. |  | void |

TcpController Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| err\_display() | 서버와 연결 하던 중 문제가 발생 할 경우 출력을 통해 알려준다. |  | void |
| tcp\_server\_init() | 윈속 초기화, 소켓 생성, 소켓 주소 할당 |  | void |
| tcp\_client\_accept() | client와 연결 대기 & 연결 이후 스레드 생성 |  | void |
| lobby | | | |
| send\_lobby\_data() | 로비에 있는 room\_data들의 정보를 보낸다  ->room data list |  | void |
| recv\_create\_room() | buf에 서버로부터 생성할 방의 정보를 받는다. |  | 성공: 1  실패: 0 |
| send\_join\_room\_stat() | 접속을 요청받았을 때 요청한 플레이어에게 접속하려는 방의 정보를 준다  ->(같은방의 player들의 name, numbering, hostnumber) |  | void |
| wait\_room | | | |
| recv\_ready() | buf 에 서버로부터 준비 여부를 받는다 | buf\_size | buf |
| recv\_emotion\_chat() | buf 에 서버로부터 플레이어들의 감정표현채팅을 수신한다 | buf\_size | buf |
| send\_emotion\_chat() | 플레이어들에게 서버에 수신한 채팅메시지를 보낸다  ->enum 감정표현들 |  | void |
| game | | | |
| recv\_bullet\_data() | buf에 client에서 온 bullet data List를 받는다 | buf\_size | buf |
| recv\_score() | buf에 client로부터 각 각 점수를 받아온다 | buf\_size | buf |
| recv\_end\_game() | 게임이 끝난 후 buf에 client로부터 GameData를 받는다 | buf\_size | buf |
| leader\_board | | | |
| send\_leader\_board | 게임이 끝나면 서버에 로컬로 저장된 leaderboard를 불러와 방의 모든 client들에게 보내준다.  ->(1위부터 10위까지의 name, score, time) | string path\_leader\_board | void |

UdpController Class

|  |  |  |  |
| --- | --- | --- | --- |
| 함수 | 설명 | 인자 | 반환값 |
| main | | | |
| err\_display() | 서버와 연결 하던 중 문제가 발생 할 경우 출력을 통해 알려준다. |  | void |
| udp\_server\_init() | 윈속 초기화, 소켓 생성, 소켓 주소 할당 |  | void |
| udp\_client\_accept() | client와 연결 대기 & 연결 이후 스레드 생성  client에서 connect를 사용했기 때문에 |  | void |
| game | | | |
| recv\_player\_data() | 프레임 마다 client로부터 buf에 Player의 정보를 받아온다  ->(player numbering, position, position, direct, life, bomb, isDamaged ) | buf\_size | buf |
| send\_players\_data() | 프레임 마다 모든 client로부터 받은 Player정보들을 모든client에게 보내준다.  ->(player data list) |  | void |
| send\_enemys\_data() | 프레임 마다 EnemyData정보들을 모든 client에게 보내준다 |  | void |
| send\_bullet\_data() | 프레임 마다 client에서 받은 bullet의 위치를 함수를 통해 얻어 그 값을 client들에게 보내준다. |  | void |

## 팀원 별 역할분담

|  |  |
| --- | --- |
| 김영서 | 클라이언트 개발,  클라이언트tcpController gamepart 구현  서버 tcpController gamepart 구현  임계영역설정 |
| 민웅기 | 리소스 제작,  클라이언트 lobby, waitRoom구현  서버 lobby, waitRoom구현  leaderboard구현 |
| 정낙현 | 클라이언트 기획서에 맞게 리팩토링,  클라이언트udpController gamepart 구현  서버 udpController gamepart 구현  이벤트구현 |

## 개발 환경

Python 3.6.3

pycham

pico2d

GitHub

StarUML

## 개발 일정

공백의 칸은 가장 최근작업의 연장

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 월 | 화 | 11/8수 | 11/9목 | 11/10금 | 11/11토 | 11/12일 |
| 김영서 |  |  | 클라이언트  코딩 |  | 클라이언트 코딩 |  | 클라이언트 병합 및 완성 마감일 |
| 민웅기 |  |  |  |  | 캐릭터, 적  리소스제작 | 필드, 탄환  리소스제작 |
| 정낙현 |  |  |  | 클라이언트 리팩토링 |  | 클라이언트 리팩토링 |
|  | 11/13월 | 11/14화 | 11/15수 | 11/16목 | 11/17금 | 11/18토 | 11/19일 |
| 김영서 | tcp\_server\_init | client\_thread |  | server\_tcp\_  connect | tcp\_client\_  accept |  |  |
| 민웅기 |  |  |  |  | recv\_lobby\_  data | send\_create\_  room |  |
| 정낙현 | udp\_server\_init |  | server\_udp\_  connect | udp\_client\_  accept |  | send\_player\_  data |  |
|  | 11/20월 | 11/21화 | 11/22수 | 11/23목 | 11/24금 | 11/25토 | 11/26일 |
| 김영서 | send\_bullet\_  data |  | send\_score | recv\_bullet\_  data | recv\_score |  |  |
| 민웅기 | send\_join\_  room | send\_lobby\_data |  |  | recv\_create\_  room | send\_join\_  room\_stat |  |
| 정낙현 | recv\_players\_  data |  | recv\_  enemys\_  data | recv\_  bullets\_  data |  | recv\_  player\_  data |  |
|  | 11/27월 | 11/28화 | 11/29수 | 11/30목 | 12/1금 | 12/2토 | 12/3일 |
| 김영서 | recv\_end\_  game |  | recv\_  leader\_  board | send\_  leader\_  board | 2주차 코드 클라이언트 반영 |  |  |
| 민웅기 | recv\_join\_  room\_stat | send\_ready\_  state |  |  | send\_emotion\_  chat | recv\_  emotion\_  chat |  |
| 정낙현 | send\_  players\_  data |  | send\_  enemys\_  data() | send\_  bullet\_  data |  | room\_  thread |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 12/4월 | 12/5화 | 12/6수 | 12/7목 | 12/8금 | 12/9토 | 12/10일 |
| 김영서 | 3주차 코드  클라이언트  반영 |  | 4주차 코드 클라이언트  반영 |  | 종합코드  QA |  | 중간 완성  각각 결산물 merge |
| 민웅기 | recv\_ready | recv\_emotion\_  chat |  |  | 종합코드  QA |  |
| 정낙현 | send\_emotion\_  chat |  | ser\_send\_  emotion\_  chat |  | 종합코드  QA |  |
|  | 12/11월 | 12/12화 | 12/13수 | 12/14목 | 12/15금 | 12/16토 | 12/17일 |
| 김영서 | 중간결산물 테스트 플레이  버그수집 | | 디버그 및  텀 프로젝트 보고 문서작업 | | | | 수고  하셧습니다. |
| 민웅기 |
| 정낙현 |
|  | 12/18월 |  |  |  |  |  |  |
| 김영서 | 최종 제출일 |  |  |  |  |  |  |
| 민웅기 |  |  |  |  |  |  |
| 정낙현 |  |  |  |  |  |  |