네트워크 게임프로그래밍

Term Project 추진계획서

**2019184004 김성준**

**2021180006 김진선**

**2021184016 백하빈**

목차

* 어플리케이션 기획
  + 게임 특징
  + 개발환경
* High level Design
  + 전체 흐름도
  + 로그인 흐름도
  + 인 게임 흐름도
* Low Level Design
  + 상수
  + client low-level
  + server low-level
  + 스레드
  + -패킷
* 팀원 별 역할분담
* 개발일정

1. **어플리케이션 기획**
2. 소개
   1. 게임 개요

* 장르: 3d 그래픽 pvp
* 게임컨셉 : 서로 탱크로 공격하여 먼저 죽이는 사람이 이기는 게임
* 플레이어 수: 2인

1. 게임 요소

* 게임 진행 : 상대의 hp를 0으로 만들 시 승리
* 조작법
  + w/a/s/d : 이동
  + q : 종료
  + 마우스: 카메라 움직임
    - 좌클릭: 포탄발사
    - 우클릭: 카메라 인칭 변경
* 아이템
  + 맵에 놓인 아이템을 먹을 시 특수효과 발동 (자신에게 도움이 되거나, 상대를 방해)

1. hp회복

* 줄어든 체력을 일부 회복
* 체력이 최대치일 시 적용되지 않음

1. 속도 증가

* 본인의 이동 속도 증가

1. 상대 freeze

* 상대방 움직임 일정시간동안 제안

1. **High Level Design**

1)전체 흐름도

텍스트, 친필, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

2)로그인 흐름도

텍스트, 친필, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

3)인게임 흐름도

텍스트, 친필, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

1. **Low Level Design**

1) 상수

enum DATA\_TYPE { SEND, RECV };

-작업자스레드에서 처리할 데이터를 구분짓기 위해서 사용

enum COMP\_TYPE { OP\_ATTACK, OP\_STATE\_CHANGE, OP\_MOVE };

-패킷처리를 할때 어떤 패킷의 내용을 처리할지 구분하기위해서 사용

enum ITEM { HEAL, SPEEDUP, FREEZE };

-어떤 아이템을 처리할지 구분하기위해 사용

constexpr int MAX\_USER = 2;

-동접자 수

constexpr int NAME\_SIZE = 20;

-아이디

constexpr char CS\_LOGIN = 0;

constexpr char CS\_MOVE = 1;

constexpr char CS\_ATTACK = 2;

constexpr char CS\_ITEM = 3;

constexpr char CS\_LOGOUT = 4;

constexpr char SC\_LOGIN\_INFO = 0;

constexpr char SC\_LOGIN\_OK = 1;

constexpr char SC\_LOGIIN\_FAIL = 2;

constexpr char SC\_MOVE\_OBJECT = 3;

constexpr char SC\_DIE\_OBJECT = 4;

constexpr char SC\_STATE\_CHANGE = 5;

constexpr char SC\_ATTACK = 6;

constexpr char SC\_LOGOUT = 7;

constexpr char SC\_GAMESTART = 8;

constexpr char SC\_PLAYING\_TIME = 9;

constexpr char SC\_RELOAD\_GUN=10;

2) client low level

텍스트, 폰트, 스크린샷, 흑백이(가) 표시된 사진

자동 생성된 설명client에서 socket의 생성 및 초기화는 메인에서 진행한다.

필요 함수 : void sendPakcet() : 서버로 패킷을 전송한다.

      void RecvPacket() : 클라이언트에서 패킷을 전송한다.

3)server low-level

텍스트, 친필, 폰트, 서예이(가) 표시된 사진

자동 생성된 설명

float x,y: 클라이언트로부터 받는 탱크의 좌표값

short hp: 플레이어 체력

float speed: 탱크 속도

bool online: 플레이어가 접속중인지 판단 전부 접속하면 게임 시작패킷 보냄

short bullet\_cnt : 클라이언트 총알 개수

void send\_login\_info\_packet(int id);

클라이언트가 로그인이 성공적일경우 서버에서 클라이언트에게 초기 정보를 부여

void send\_move\_packet(int id);

이동된 탱크의 좌표를 보내줌

void send\_change\_state\_packet(int id);

아이템or공격으로 인한 상태변화를 전송해줌

void send\_player\_die\_packet(int id);

체력이 다 떨어지면 죽음 패킷 전송

bool CheckCollision(){};

객체간 충돌처리 진행

void send\_Reload\_gun\_packet(int id);

총알을 다 썼을때 전송하는 패킷

std::array<Session,MAX\_USER> g\_players;

동접수가 많지 않으므로 플레이어 컨테이너는 array를 사용 session에서 정의한 구조체를 바탕으로 글로벌플레이어 변수 선언

공유자원인 g\_players로 인한 임계영역 사용

하나의 프로세스에 대해서만 임계영역을 설정할 필요가 있기 때문에 무거운 커널영역에 있는 객체인 mutex를 사용하기 보다는 criticalSection을 사용할예정

4)스레드

스케치, 그림, 도표, 텍스트이(가) 표시된 사진

자동 생성된 설명

1)Main Thread

소켓을 생성하고 초기화 함 listen상태로 계속해서 클라이언트의 접속을 받기 위해 대기함

2.Client Thread

accept()를 성공적으로 받으면 스레드를 할당한 뒤 클라이언트로부터 데이터를 받음

3.WokerThread

클라이언트로부터 전송받은 패킷의 세부정보를 해석하고 처리하는 스레드

4.do\_send

해석된 데이터를 클라이언트로 전송하는 스레드

ex) 클라이언트로부터 접속을 받은 서버는 클라이언트1스레드를 할당한다. 만약 클라이언트로부터 데이터를 수신받았다면 클라이언트 내부 루프에서 switch문을 통해 DATA\_TYPE::RECV: 로 이동 후WokerThread를 통해 데이터를 처리한다,  처리가 끝난 데이터들에 대해서 DATA\_TYPE::SEND: 로 이동 후 do\_send스레드를 통해 데이터를 클라이언트에게 송신한다.

5)패킷

-client to server

struct CS\_LOGIN\_PACKET {

char type;

char name[NAME\_SIZE];

};

-로그인시도할때 보내는 패킷

struct CS\_MOVE\_PAKCET {

char type;

char direction;

};

-키보드로 입력받은 움직이를 보내는 패킷

struct CS\_ITEM\_PACKET {

char type;

char item;

};

-아이템을 먹었을때 보내는 패킷

struct CS\_LOGOUT\_PACKET {

char tpye;

};

로그아웃할때 보내는 패킷

-server to client

struct SC\_LOGIN\_INFO\_PACKET {

char type;

int id;

int hp;

short x, y;

};

로그인 성공시 클라에게 보내는 초기정보 패킷

struct SC\_LOGIN\_OK\_PACKET {

char type;

};

성공적으로 로그인됐다는 패킷

struct SC\_LOGINFAIL\_PACKET {

char type;

};

로그인 실패 패킷

struct SC\_MOVE\_PACKET {

char type;

int id;

short x, y;

};

움직임 입력한 클라이언트에게 보내는 위치값 패킷

struct SC\_DIE\_PACKET {

char type;

};

플레이어 사망 패킷

struct SC\_STATE\_CHANGE\_PACKET {

char type;

int id;

char state;

};

아이템으로 인한 상태 변화 계산 후 보내는 패킷

SC\_RELOAD\_PACKET {

char type;

int id;   
};

1. **팀원간 역할분담**

|  |  |  |  |
| --- | --- | --- | --- |
| **구분** | **구분 내용** | **구현 함수** | **담당자** |
| **서버** | **Session** | **Session class** | **김진선** |
| **로그인** | **send\_login\_info\_packet(int id)**  **send\_login\_fail\_packet(int id)** | **백하빈** |
| **클라이언트로부터 받은 패킷처리** | **WokerThread(int id, char\* packet)** | **김성준**  **김진선** |
| **패킷세부내용 해체후처리** | **send\_move\_packet()**  **send\_change\_satet\_packet()**  **send\_player\_die\_packet()**  **send\_Reload\_gun\_packet()** |
| **send 스레드 구현** | **do\_send()** | **백하빈** |
| **충돌처리** | **collision()** | **김성준,백하빈,김진선** |
| **클라이언트** | **서버로부터 받은 데이터 처리** | **RecvPacket()** | **백하빈 김진선** |
| **서버로 데이터 송신** | **sendPacket()** | **백하빈 김진선** |
| **클라이언트 연결 구현** | **connect()까지** | **김성준** |

1. **개발일정**

10/31 - 11/2

|  |  |  |
| --- | --- | --- |
| 10/31 | 11/1 | 11/2 |
|  |  |  |

11/3 -  11/9

서버-클라 로그인까지 구현목표

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11/3 | 11/4 | 11/5 | 11/6 | 11/7 | 11/8 | 11/9 |
|  | 김진선 : Session 클래스 구현  김성준 : 패킷 및 프로토콜 구현  백하빈: 클라이언트 네트워크 연결 구현 |  |  | 김성준 : 클라이언트 스레드 구현  김진선 : 클라이언트 코드 수정  백하빈 : 로그인패킷을 이용한 접속성공,실패 구현 | 테스트 | |

11/10 - 11/16

구체적 패킷 해석 처리(움직임)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11/10 | 11/11 | 11/12 | 11/13 | 11/14 | 11/15 | 11/16 |
|  | 김성준 : WokerThread -  send\_move\_pakcet()  백하빈 : SendPacket(), RecvPacket() move패킷만 | |  | 테스트 | |  |

11/17 - 11/23

구체적 패킷 해석 처리(공격,체력깎임,사망)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11/17 | 11/18 | 11/19 | 11/20 | 11/21 | 11/22 | 11/23 |
|  | 김진선 : WorkerThread - send\_player\_die\_packet()  send\_state\_change\_pacet()  백하빈 : SendPacket() RecvPacket() attack 및 hp 패킷만 | |  | 테스트 | |  |

11/24 - 11/30

구체적 패킷 해석처리(아이템으로 인한 상태변화,재장전)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11/24 | 11/25 | 11/26 | 11/27 | 11/28 | 11/29 | 11/30 |
|  | 김성준 : WorkerThread - send\_state\_change\_packet()  send\_Reload\_gun\_packet()  김진선 : SendPacket() RecvPacket() item및 reload패킷만 | |  | 테스트 | |  |

12-1 - 12/7

충돌처리

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 12/1 | 12/2 | 12/3 | 12/4 | 12/5 | 12/6 | 12/7 |
| 전인원 : 맵+상대객체에 대한 모든 충돌처리 | | | 최종테스트 | | | |