텀 프로젝트 최종보고서

네트워크 게임 프로그래밍

10 팀

2019184102 박병준

2017180011 박정환

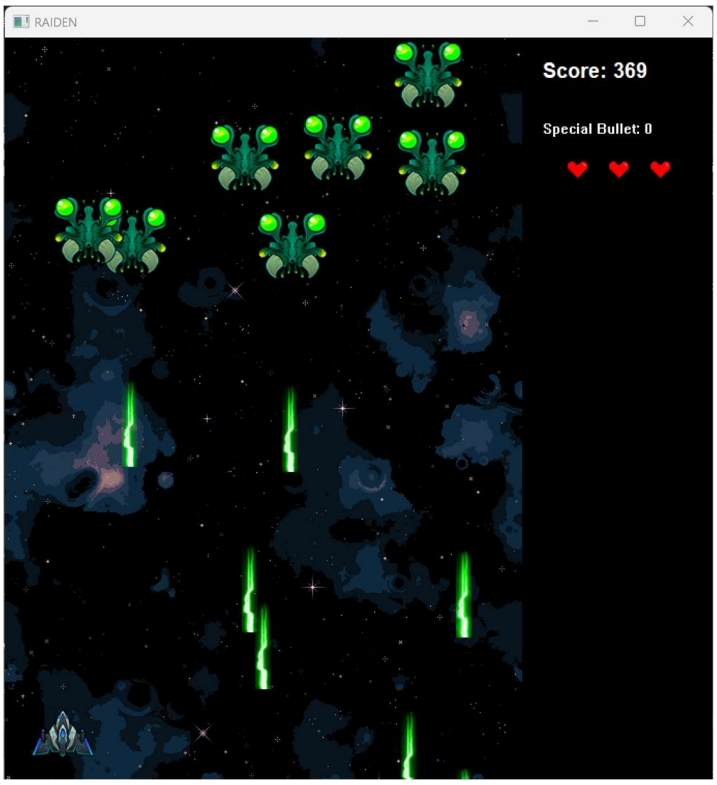
2018180022 방영규

목차

1. 게임 소개

- 게임 개요

클라이언트 프로그램 출처: 박정환 학생의 윈도우 프로그래밍 프로젝트. 라이덴 모작 게임.



네트워크 게임 방향: 2인 멀티플레이 게임
게임을 실행하면 다른 플레이어와 매칭이 될 때까지 대기한다.
매칭이 성공하고 게임이 시작되면 각자 본인의 전투기를 조작하여 게임을 플레이 한다.
플레이어는 스페이스바 키를 입력하여 총알을 발사한다.
무작위 위치에 출현하는 적을 총알을 발사하여 제거한다.

상하좌우 방향키 – 이동
스페이스바 – 공격

시프트 + 스페이스바 – 특수 공격

2. 개발 환경


Visual Studio

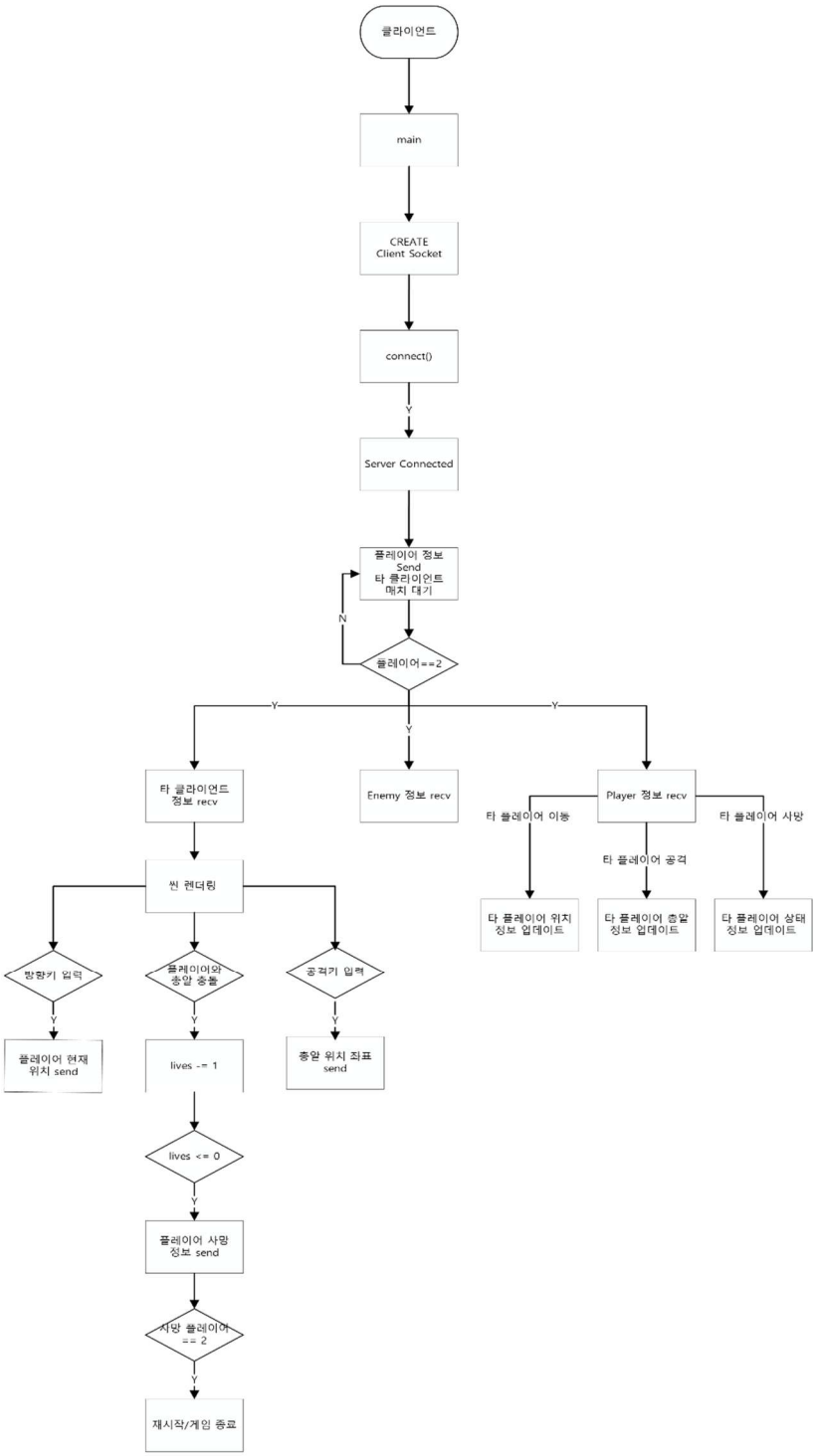
TCP / IP

Windows API


Github Desktop

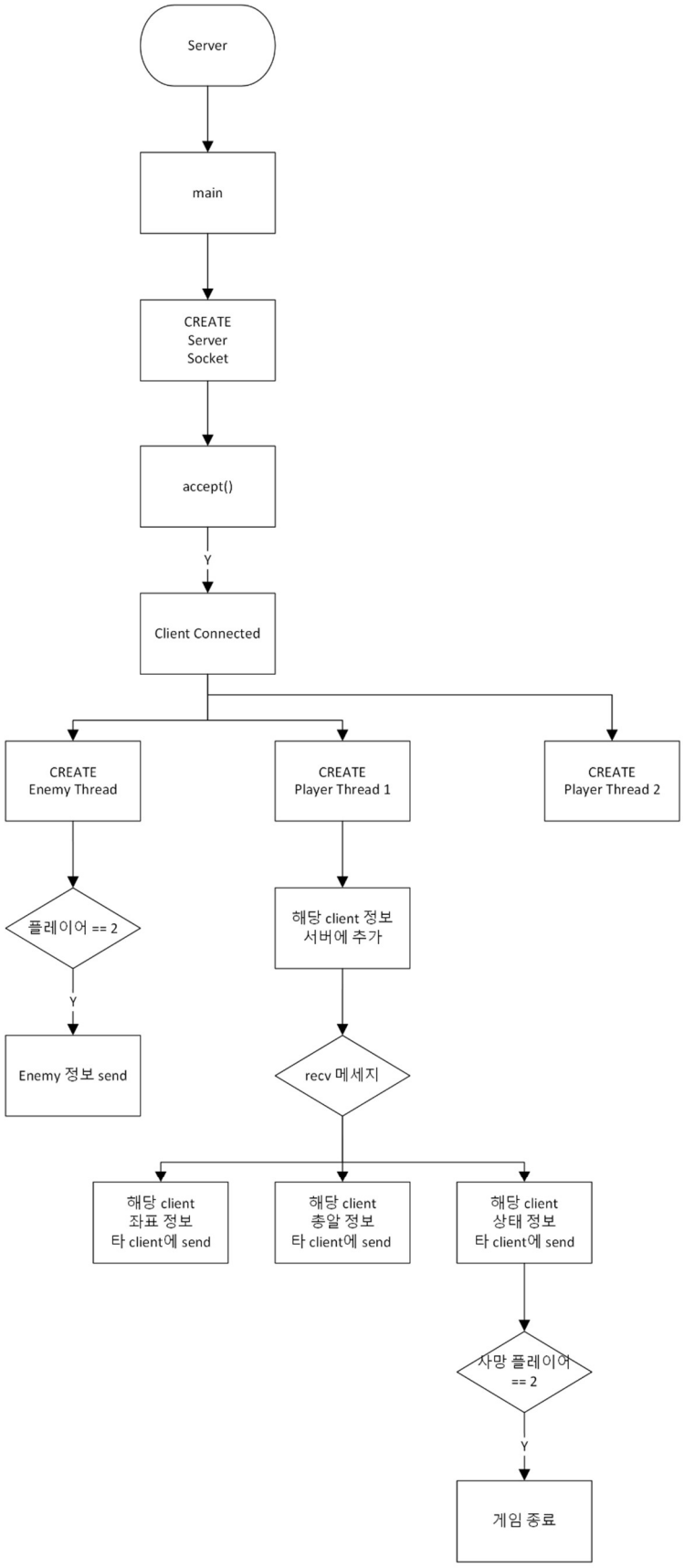
3. High Level

- Client



클라이언트는 main Thread만 사용.

- Server


Server
main
CREATE
Server
Socket
accept()
Y
Client Connected
CREATE
Enemy Thread
CREATE
Player Thread 1
CREATE
Player Thread 2
플레이어 == 2
Y
Enemy 정보 send
해당 client 정보
서버에 추가
recv 메세지
해당 client
좌표 정보
타 client에 send
해당 client
총알 정보
타 client에 send
해당 client
상태 정보
타 client에 send
사망 플레이어
== 2
Y
게임 종료

서버는 main, Enemy, Client_1 Thread, Client_2 Thread 총 4개의 Thread로 구성

main

클라이언트와 accept하여 서버와 클라이언트를 연결하는 역할.

2명의 플레이어가 연결될 때까지 대기한다.

Enemy

클라이언트에게 무작위의 위치에 생성되는 적 위치를 전송하는 역할.

Client

클라이언트에 대한 정보를 관리하고 클라이언트에게 데이터를 전송하는 역할.

4. Low Level

- Client

  - 구조체

    struct BulletData{ int x,y; bool destroy, send;}

    서버한테 보내줄 총알 정보. 총알 좌표랑 생존(파괴)여부를 포함

  - 함수

    RecvClientID(SOCKET sock)

    서버로부터 클라이언트 id를 수신하고 클라이언트 id를 저장

    void SendPlayerInfo(Fighter& player, SOCKET& sock), void RecvPlayerInfo(Fighter& anotherplayerFighter, SOCKET& sock)

    서버랑 플레이어 정보를 주고받는 역할

    RecvBulletPos(GameManager& gameManager, SOCKET& sock), SendBulletPos(GameManager& gameManager, SOCKET& sock)

    총알정보를 서버한테 보내주고 서버가 다른 클라이언트한테 보내주는 총알 정보를 받는 역할

    RecvEnemyInfo(GameManager&gameManager, SOCKET& sock)

    서버에서 적의 위치 좌표를 받는 역할

    RecvPlayerDead (GameManager& gameManager), SendPlayerDead(GameManager& gameManager)

    플레이어의 사망정보를 서버와 주고받는 역할

    void RecvPlayerPos(Fighter& anotherplayerFighter, SOCKET& sock)
    void SendPlayerPos(Fighter& anotherplayerFighter, SOCKET& sock)

플레이어 위치 좌표를 서버와 주고받는 역할

void SendGameStart(SOCKET sock)
게임 시작을 서버한테 보내주는 역할

- Server

  - 구조체

    struct BulletData { int x, y; bool destroy, send; }

    총알 전송 유무를 파악.

    struct clientinfo {int id; SOCKET client; }

    클라이언트 정보를 주고받기 위한 패킷

    struct PlayerSock {SOCKET client_sock;

    vector<BulletData>BulletVector; int x = 0, y = 0;

    bool isGameStarted = false; bool dead = false; }

    서버에서 클라이언트 관리용으로 쓰는 구조체

  - 함수

    void RecvGameStart(PlayerSock* PS, int clientId)

클라이언트한테 게임 시작를 받는 역할

void SendPlayerBullet(PlayerSock* send_PS, PlayerSock* recv_PS),
void RecvPlayerBullet(PlayerSock* PS)
플레이어 총알정보를 클라이언트랑 주고받는 역할

void SendPlayerDead(PlayerSock* send_PS, PlayerSock* recv_PS),
void RecvPlayerDead(PlayerSock* PS)
플레이어 사망정보를 클라이언트랑 주고받는 역할

void SendClientID(PlayerSock* PS, int clientId)
클라이언트한테 받은 클라이언트 정보를 다른 클라이언트한테 보내주는 역할

5. 개발 결과

- 역할 분담

이름	클라이언트	서버
방영규	SendPlayerInfo() RecvPlayerInfo() SendBulletInfo() RecvBulletInfo() SendPlayerDead() RecvPlayerDead()	ClientThread() SendBulletInfo() RecvBulletInfo() RecvPlayerDead() SendPlayerDead()
박정환	RecvEnemyInfo() RecvClientID() SendGameStart()	EnemyThread() SendClientID() RecvGameStart()
박병준	Sendmoveplayer() RecvThread()	SendInitPos()