

Server 구조 기획

KGCA 22기 설은경

(KGCA 22th EunGyeong, Seol)

목차

[1. 게임의 개요 2](#_Toc321753269)

[2. 서버 구조 3](#_Toc321753270)

[A. Login Server 4](#_Toc321753271)

[B. Lobby Server 7](#_Toc321753272)

[C. Game Server 10](#_Toc321753273)

# 게임의 개요

* 게임의 개요

|  |  |  |
| --- | --- | --- |
| 게임의 제목 | Unkown | * 근 미래를 배경으로 어둠과 은신을 주제로 시야에 제한을 두어 긴장감과 심리전에 가장 큰 초점을 맞춤. * 신중하고 전략적인 움직임을 통해 기존의 FPS와 다른 느낌을 추구. * 스킬, 아이템을 이용하여 다양한 플레이 유도. |
| 플렛폼 | PC온라인 |
| 장르 | FPS |
| 플레이 인원 | 6~8인 |
| 플레이 타임 | 5분 |

* 게임 컨셉

21xx년 미래,

대량살상 무기 금지조약으로

세계 열강들은 다른 곳으로 눈을 돌리게 되는데

소수의 고성능의 장비를 갖춘 인원으로

적을 타격 하는 것이

가장 효율적이라 여긴 각국은

경쟁적으로 특수부대를

비밀리에 육성하기 시작하고

세계 각지에서 그들간의

보이지 않는 전쟁이 시작된다.

( 출처 : 레벨 디자인 초안\_min in made, 하진욱 )

# 서버 구조

* 서버 분할
  + 서버는 총 3개로 Login, Lobby, Game서버로 나뉜다.
  + DB에 저장해야 하는 Data의 중요도가 낮다고 판단하여 파일연결로 처리.
  + DB와의 연결은 로그인과 로비만 처리한다.
  + **Login Srv** : Client가 처음 접속하는 서버로 player의 로그인 처리를 담당.
  + **Lobby Srv** : player가 로그인을 성공하고 다음으로 접속하는 서버.

원하는 방으로 들어가거나 자신이 방을 생성할 수 있으며,

게임을 시작하기 전에 사람들과 채팅을 할 수 있다.

* + **Game Srv** : 게임 로직을 담당하는 서버.
* 다이어그램 표시 규칙
  + 다이어 그램에 WorkThread의 행동은 생략되어 있다.

: 시작과 끝

: 동작

: 로직 처리

: 패킷 종류

## Login Server

Accept Thread

서버 셋팅 및 동작

서버 종료?

Yes

사용가능

클라 접속?

패킷 처리

로그인?

중복체크?

로그인 요청

접속 종료

ID중복 체크

새로 만들기

접속 끊기

중복

Yes

성공

No

실패

***- Packet 프로토콜 정의***

|  |  |
| --- | --- |
| **SC\_LOGIN\_CONNECT\_OK** | 클라이언트의 접속( accept )가 성공하면 접속 성공패킷을 보내 접속이 성공했다는 패킷을 보낸다. |
| 데이터 없음 |
| **CS\_LOGIN\_CHECK\_ID** | player는 계정을 새로 만들기 위해 ID중복체크 패킷을 서버로 보낸다.  ( ID의 공백체크는 전송 전에 선행되어야 함 ) |
| **Int** : id 데이터 크기  **TCHAR** : id |
| **SC\_LOGIN\_CHECK\_ID\_RESULT** | ID 중복체크에 대한 DB검색 결과를 클라에게 보내준다. |
| **Int** : result값 ( -1(음수) : 중복 / 1(양수) : 사용가능 ) |
| **CS\_LOGIN\_CREATE\_ACCOUNT** | 계정 새로 만들기 패킷을 받는다. |
| **Int** : id데이터 크기  **TCHAR** : id  **Int** : password 데이터 크기  **TCHAR** : password  **Int** : e-mail 데이터 크기  **TCHAR** : e-mail |
| **SC\_LOGIN\_CREATE\_RESULT** | 새로만든 계정에 대한 결과를 클라에게 |
| **Int** : result값 ( -1(음수) : 실패 / 1(양수) : 성공 ) |
| **CS\_LOGIN\_LOGIN** | 만들어진 계정으로 로그인 시도 |
| **Int** : id데이터 크기  **TCHAR** : id  **Int** : password 데이터 크기  **TCHAR** : password |
| **SC\_LOGIN\_LOGIN\_RESULT** | 로그인 시도에 대한 결과 |
| **Int** : result값 ( -1(음수) : id가 없음/ 0 : password 틀림 / n(양수) : 로그인 성공(세션ID) )  *--------------로그인이 성공했을때만 존재하는 data--------------------------*  ***Int*** *: ip주소의 데이터 크기*  ***TCHAR*** *: ip주소*  ***Int*** *: port번호* |

***- 세부 사항***

|  |
| --- |
| * DB와의 연결은 mdb파일연결로 되어있다. * 기본 설정으로 id와 비밀번호는 최소 6자부터 20자까지로 제한한다( 추후 달라질 수 있음 ) |

## Lobby Server

( Srv.docx\_ 1페이지 )

***- Packet 프로토콜 정의***

|  |  |
| --- | --- |
| **GL\_CONNECT\_SERVER** | 게임서버가 로비서버에 연결하고 자신이 서버라는 신호를 보낸다 |
| 데이터 없음 |
| **GL\_PLAYER\_DISCONNECT** | 게임서버에서 게임을 종료해 버린 player의 정보을 보낸다. |
| **Int** : 게임( 방) 번호  **Int** : 해당 player의 sessionId |
| **LG\_START\_GAME** | 로비서버에서 게임서버로 게임 시작을 알림. |
| **Int** : 게임 (방) 번호  **Int** : 인원 |
| **SC\_LOBBY\_CONNECT\_OK** | Client가 서버에 정상적으로 접속했을 때 넘겨 주는 패킷 |
| 데이터 없음 |
| **CS\_LOBBY\_INSERT\_LOBBY** | 서버에 자신의 정보를 보내고 다른 정보를 요청하는 패킷 |
| **Int** : 자신의 SessionID  **Int** : 속해 있는 방 번호( 없다면 0을 보낸다 ) |
| **SC\_LOBBY\_OTHER\_CHARINFO** | 로비의 캐릭터 정보를 담은 패킷 |
| **Int** : 패킷에 들어있는 캐릭터 정보의 개수  **Int** : 캐릭터의 sessionId  **Int** : id데이터의 크기  **TCHAR** : id  (└캐릭터 수 만큼 반복해서 들어간다.) |
| **SC\_LOBBY\_ROOMINFO** | 첫 접속 시 Client가 받아야 하는 방들의 정보를 담은 패킷 |
| **Int** : 패킷에 들어있는 방 정보의 개수  **Int** : 방의 Id  **Int** : 해당 방의 인원  **Int** : 방 문구의 데이터 크기  **TCHAR** : 방 문구  **Int** : 방 상태( 0 : 일반 / 1(양수) : 게임 중 )  (└방 수만큼 반복해서 들어간다.) |
| **CS\_LOBBY\_CREATE\_ROOM** | 방 만들기를 요청 |
| **Int** : 방의 Id  **Int** : 방 문구의 데이터 크기  **TCHAR** : 방 문구 |
| **SC\_ROOM\_RESULT\_CREATE** | 방 만들기 요청에 대한 결과 |
| **Int** : result( 0 : 생성 가능 / n(양수) : 생성된 방의 id )  **Int** : 자신의 팀 정보( 생성 실패하면 팀 데이터는 없다. ) |
| **SC\_LOBBY\_OPEN\_ROOM** | 방이 만들어 졌다는 패킷을 로비의 사람들에게.. |
| **Int** : 방의 ID  **Int** : 방의 제목 데이터 크기  **TCHAR** : 방의 제목 |
| **SC\_LOBBY\_CLOSE\_ROOM** | 방이 사라졌다는 패킷을 로비의 사람들에게.. |
| **Int** : 방의 ID |
| **CS\_LOBBY\_INSERT\_ROOM** | 방에 들어간다 |
| **Int** : 방의 Id |
| **SC\_ ROOM \_RESULT\_INSERT** | 방에 들어가는 요청에 대한 결과 |
| **Int** : result ( -1 : 게임중 / 0 : 인원초과 / n(양수1, 2) : 입장(팀) )  **Int** : Team정보를 받음 |
| **SC\_ LOBBY \_INSERT\_ROOM** | 캐릭터가 방에 들어갔다고 알림 |
| **Int** : 방의 Id  **Int** : SessionId |
| **SC\_ ROOM \_OTHER\_CHARINFO** | 방에 있는 사람들의 정보를 받음 |
| **Int** : 패킷에 들어있는 캐릭터 정보의 개수  **Int** : 캐릭터의 sessionId  **Int** : id데이터의 크기  **TCHAR** : id  **Int** : Team정보  (└캐릭터 수 만큼 반복해서 들어간다.)  **Int** : 방장의 SessionID |
| **SC\_ ROOM \_LEADER** | 방장이 나가고 다음 방장이 누구인지 보낸다 |
| **Int** : 캐릭터의 SessionId |
| **CS\_ ROOM \_OUT\_ROOM** | 방에서 나온다. |
| 데이터 없음 |
| **SC\_ROOM\_CHAR\_OUT** | 방을 나간 캐릭터 정보를 방의 사람들에게 |
| **Int** : SessionId |
| **SC\_LOBBY\_ROOMPLAYER\_DISCOUNT** | 로비에 방의 인원이 줄었다는 것을 알린다. |
| **Int** : 방 번호 |
| **CS\_ROOM\_CHAR\_READY** | 캐릭터가 ready상태가 변함 |
| 데이터 없음 |
| **SC\_ROOM\_CHAR\_READY** | 캐릭터의 ready상태가 변했다는 것을 모두에게 알림 |
| **Int** : SessionId  **Int** : ready 상태 |
| **CS\_ROOM\_TEAM\_CHANGE** | 캐릭터의 team을 변경 |
| **Int** : Team |
| **SC\_ROOM\_TEAM\_CHANGE** | 캐릭터의 team변경 내용을 모두에게 알림 |
| **Int** : SessionID  **Int** : Team |
| **CS\_ROOM\_CHAT** | 채팅 |
| **Int** : 문장 데이터 크기  **TCHAR** : 문장 |
| **SC\_ROOM\_CHAT** | 채팅 |
| **Int** : SessionID  **Int** : 문장 데이터 크기  **TCHAR** : 문장 |
| **CS\_ROOM\_START** | 방장이 게임 시작을 누른다 |
| **데이터 없음** |
| **SC\_ROOM\_START\_RESULT** | 게임 시작이 가능한가에 대한 결과값 return |
| **BOOL** : 결과값 |
| **SC\_ROOM\_GAME\_START** | 게임서버로 이동하라고 보냄 |
| **Int** : ip데이터 크기  **TCHAR** : ip  **Int** : port 번호 |
| **SC\_LOBBY\_GAME\_START** | 로비에서 방 정보를 게임 중으로 바꾸라고 보냄 |
| **Int** : Room 번호 |

***- 세부 사항***

|  |
| --- |
| * session을 관리하는 list가 기본적인 sessionMgr이 가진 list를 제외하고 lobbylist + roomlist(\*방개수) 만큼 있다. * 클라이언트가 CS\_LOBBY\_INSERT\_LOBBY를 보냈을 때 속해 있는 방 번호가 0이면 SC\_LOBBY\_OTHER\_CHARINFO와 SC\_LOBBY\_ROOMINFO를 받고 방 번호가 있으면 lobbylist에 추가되지 않고 해당 방을 확인하고 방으로 넘긴다. * 방 만들기를 요청 받아 방이 만들어지면 해당 session을 lobbylist에서 해당 roomlist로 옮긴다. * 각 roomlist의 처음 사람이 방장이며 방장이 나가는 경우 바로 다음(그 다음 header)사람이 방장이 된다. * 방에서 나가면 클라는 다시 CS\_LOBBY\_INSERT\_LOBBY를 보내게 된다. * 처음 방에 들어올 때 TEAM는 순서대로 랜덤(?)으로 배정된다. * 팀이 균등하지 않거나 인원이 최소 6명이 되지 않으면 게임을 시작 할 수 없다. * 게임 시작을 위해 서버의 접속이 끊긴 경우(CS\_ROOM\_PLAY패킷을 이미 받아 놓은 경우) ROOM의 사람수가 줄지 않으며 roomlist에서 제외되지 않는다. * roomlist가 empty인 경우 방은 초기화 된다. |

## Game Server

( Srv.docx\_ 2페이지 )

***- Packet 프로토콜 정의***

|  |  |
| --- | --- |
| **GL\_CONNECT\_SERVER**  **(로비서버의 패킷 설명과 동일)** | 게임서버가 로비서버에 연결하고 자신이 서버라는 신호를 보낸다 |
| 데이터 없음 |
| **GL\_PLAYER\_DISCONNECT**  **(로비서버의 패킷 설명과 동일)** | 게임서버에서 게임을 종료해 버린 player의 정보을 보낸다. |
| **Int** : 게임( 방) 번호  **Int** : 해당 player의 sessionId |
| **LG\_START\_GAME**  **(로비서버의 패킷 설명과 동일)** | 로비서버에서 게임서버로 게임 시작을 알림. |
| **Int** : 게임 (방) 번호  **Int** : 인원 |
| **SC\_GAME\_CONNECT\_OK** | Client가 서버에 정상적으로 접속했을 때 넘겨 주는 패킷 |
| 데이터 없음 |
| **CS\_GAME\_INGAME** | 게임서버에 자신의 정보를 보내고 위치 정보를 요청한다 |
| **Int** : SessionID  **Int** : roomID  **Int** : Team |
| **SC\_GAME \_CHARINFO\_INGAME** | 접속한 player의 좌표값 포함 정보를 자신을 포함한 모두에게 보낸다  / 접속했을 때 먼저 접속해 있는 캐릭터의 정보를 접속한 player에게 보낸다 |
| **Int** : 패킷에 들어있는 캐릭터 정보의 개수  **Int** : 캐릭터의 sessionId  **Int** : 캐릭터의 상태  **Float**: 위치 X  **Float**: 위치 Y  **Float**: 위치 Z  **Float**: 방향 X  **Float**: 방향 Y  **Float**: 방향 Z  (└캐릭터 수 만큼 반복해서 들어간다.) |
| **SC\_GAME\_START\_GAME** | 접속 할 캐릭터가 모두 정상적으로 접속했으니 게임을 시작하라는 명령을 보낸다 |
| 데이터 없음 |
| **CS\_GAME\_MOVE** | 캐릭터의 상태가 변함(이동시작 혹은 멈춤 등) |
| **Int** : 캐릭터의 상태  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z  **Float** : 방향 X  **Float** : 방향 Y  **Float** : 방향 Z |
| **SC\_GAME\_MOVE\_CHAR** | 캐릭터의 변한 상태를 모두에게 보낸다 |
| **Int** : SessionID  **Int** : 캐릭터의 상태  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z  **Float** : 방향 X  **Float** : 방향 Y  **Float** : 방향 Z |
| **CS\_GAME\_ATTECT** | 총을 쏘는 등의 공격을 가했을 때 보내는 패킷 |
| **Int** : 무기 종류( ex )1:라이플 / 2: 샷건 등등 )  **Int** : 피격 대상 SessionID  **Int** : 피격체의 부위( ex )1 : 머리 / 2 : 몸통 등등 ) |
| **SC\_GAME\_ATTECT** | 공격의 결과 공격이 성공했을 때 모두에게 공격 정보를 보낸다 |
| **Int** : 공격자 SessionID  **Int** : 무기 종류  **Int** : 피격 대상 SessionID  **Int** : 피격 부위  **Int** : 감소 에너지 |
| **SC\_GAME\_CHAR\_DIE** | 공격당한 캐릭터의 에너지가 0이하가 되면 죽음(?) 패킷을 보낸다 |
| **Int** : 공격자 SessionID  **Int** : 무기 종류  **Int** : 피격 대상 SessionID  **Int** : 피격 부위 |
| **CS\_GAME\_SYNC** | 위치 및 상태 동기화 패킷 |
| **Int** : 캐릭터의 상태  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z  **Float** : 방향 X  **Float** : 방향 Y  **Float** : 방향 Z |
| **SC\_GAME\_SYNC** | 위치 및 상태 동기화 패킷 |
| **Int** : SessionID  **Int** : 캐릭터의 상태  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z  **Float** : 방향 X  **Float** : 방향 Y  **Float** : 방향 Z |
| **CS\_GAME\_INSTALL\_BOOM\_** | 공격팀의 player가 폭탄을 설치 |
| 데이터 없음 |
| **SC\_GAME\_INSTALL\_BOOM** | 폭탄설치 사실을 알림 |
| Int : 설치한 player의 SessionID |
| **SC\_GAME\_BOOM\_COUNTDOWN** | 폭탄을 설치하고의 제한 시간의 흐름 |
| 데이터 없음( or Int : 남은 카운트 ) |
| **CS\_GAME\_UNINSTALL\_BOOM** | 수비팀의 player가 폭탄을 제거 |
| 데이터 없음 |
| **SC\_GAME\_UNINSTALL\_BOOM** | 폭탄 제거를 알림 |
| Int : 폭탄을 제거한 player의 SessionID |
| **SC\_GAME\_TIME\_COUNTDOWN** | 1초마다 countdown |
| 데이터 없음( or int : 남은 카운트 ) |
| **SC\_GAME\_END** | 게임 종료 |
| **Int** : 게임종료의 종류 ( ex) 1: 팀 전멸/2: 시간초과/ 3: 목표성공 등 )  **Int** : 승리 팀  **Int** : 공격팀 Kill수( 수비팀 DeathCount )  **Int** : 수비팀 Kill수( 공격팀 DeathCount ); |
| **CS\_GAME\_INSTALL\_ITEM** | item설치 |
| **Int** : item종류  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z |
| **SC\_GAME\_INSTALL\_ITEM** | item설치 |
| **Int** : 설치자의 Team( 혹은 SessionID )  **Int** : item종류  **Int** : item ID  **Float** : 위치 X  **Float** : 위치 Y  **Float** : 위치 Z |
| **CS\_GAME\_RUN\_ITEM** | Item 작동 |
| **Int** : item ID  **Int** : 데미지 |
| **SC\_GAME\_RUN\_ITEM** | Item 작동 |
| **Int** : item에 공격당한(?) player의 SessionID  **Int** : item ID  **Int** : 데미지 |
| **CS\_GAME\_GOTO\_LOBBY** | 게임이 끝나고 방으로(로비서버로) 돌아간다 |
| 데이터 없음 |
| **SC\_GAME\_GOTO\_LOBBY** | 로비서버로 돌아가기 위해 ip와 포토번호를 넘긴다 |
| **Int** : IP번호 데이터 크기  **TCHAR** : IP  **Int** : Port 번호 |
| **SC\_GAME\_CHAR\_GOTO\_LOBBY** | Player가 로비서버로 이동함 |
| **Int** : 세션 ID |
| **SC\_GAME\_CHAR\_DISCONNECT** | Player가 접속을 종료했음을 알림 |
| **Int** : 세션 ID |

***- 세부 사항***

|  |
| --- |
| * 로비서버로부터 방을 생성하는 패킷을 받아 방을 연다. * 게임이 끝나고 사람들이 모두 접속을 끊게 되면 방을 초기화 하고 닫는다. * 게임에서 player가 종료하면 해당 player의 정보는 로비서버까지 전해 줘야 한다. * 게임이 끝나고 로비서버로의 이동을 위해 접속을 종료하게 되는 player의 경우 접속을 끊기 전에 **CS\_GAME\_RETURN\_TO\_ROOM**패킷을 보내야 한다. |