

SegClient Library Document

KGCA 22기 설은경

(KGCA 22th EunGyeong, Seol)

목차

[Class **SNetwork** 3](#_Toc323569794)

[Class **SPacket** 4](#_Toc323569795)

[Class **SPacketQueue** 5](#_Toc323569796)

[Protocol정의 6](#_Toc323569797)

# Class **SNetwork**

* Singleton.
* GetNetwork.함수()로 호출 가능
* Init( BOOL isNon )
  + 함수 초기화
  + 매개변수는 기본값 TRUE
  + 소켓을 block형으로 설정하는지에 대한 flag값( 그냥 Init()하면 됨 )
  + Return : TRUE(성공)/ FALSE(실패)
* ConnectToSrv( char\* ip, int port )
  + 서버와 연결하는 함수
  + 로그인 서버의 ip주소와 포트번호를 입력한다.
  + 로그인 서버의 포트번호는 7900번으로 고정되어 있지만,   
    서버컴퓨터는 56번이지만 변할 수 있다.
  + 연결과 함께 쓰레드가 동작을 시작한다.
  + Return : TRUE(성공)/ FALSE(실패)
* ReConnect( char\* ip, int port )
  + 서버와 재연결 혹은 서버 교체(?)를 하는 함수
  + 로그인에서 로비, 혹은 로비에서 게임서버 등 서버를 바꿀 때 사용하는 함수로  
    ip와 port번호는 입력하고 선언해 주면 알아서 지금의 서버에 연결을 끊고  
    새로 입력된 ip와 port번호로 연결된다.
  + Return : TRUE(성공)/ FALSE(실패)
* DisConnect()
  + 서버와의 연결을 끊는 함수로 기본적으로 class의 소멸자에서 자동 호출됨
  + 굳이 사용할 필요는 없지만 만일을 위해 알아 두세요ㅋ
* SendPacket( SPacket\* packet )
  + 서버로 패킷을 전송하는 함수
  + 완성된 패킷을 넘기면 전송하고 결과를 return한다
  + Return : TRUE(성공)/ FALSE(실패)

# Class **SPacket**

* 전송하거나 해석할 패킷을 만들 함수
* ID입력
  + SPacket sendPacket( ID );
  + SPacket sendPacket;  
    sendPacket.SetID( ID );
* ID얻기
  + Unsigned short GetID();
* 패킷의 재사용(?)을 위해 패킷을 초기화
  + Void PacketClear();
* 데이터 입력 : 문자열을 제외  
  ( BYTE, short, int, DWORD, float, \_\_int64, SPacket 의 형태)
  + SPacket << value;
* 데이터 입력 : 문자열
  + SPacket.PutDataA( char\*, int );
  + SPacket.PutDataW( TCHAR\*, int );
  + 위의 두 함수를 특별히 구분해야 하지 않는다면 SPacket.PutData로 사용하면 됨
  + 두번째 매개 변수인 int는 앞의 문자열 데이터의 크기( 문자열의 길이가 아님 )
  + Ex) char형 문자열 abc 라면 int는 3  
     TCHAR형 문자열 abc 라면 int는 6이 된다.
* 데이터 출력 : 문자열을 제외  
  ( BYTE, short, int, DWORD, float, \_\_int64, SPacket 의 형태)
  + SPacket >> 변수;
* 데이터 출력 : 문자열
  + SPacket.GetDataA( char\*, int );
  + SPacket.GetDataW( TCHAR\*, int );
  + 위의 두 함수를 특별히 구분해야 하지 않는다면 SPacket.GetData로 사용하면 됨
  + 두번째 매개 변수인 int는 받을 문자열의 데이터 크기( 문자열의 길이가 아님 )
  + 예제는 위와 같다.

# Class **SPacketQueue**

* Singleton.
* GetPacketQ.함수()로 호출 가능
* 패킷이 담겨 있는 Queue
* Int GetDataCount();
  + 현재 패킷큐에 담겨있는 패킷의 개수를 나타냄
* Void GetPacket( SPacket& packet );
  + 패킷처리를 위해 패킷을 꺼내오는 함수
  + 넘겨 받은 packet에 한 개의 패킷을 담아 준다
  + 단! 이 함수가 호출되었다고 패킷큐에 패킷이 하나 줄어드는 것은 아니다
  + Return: TRUE/ FALSE(큐에 패킷이 없을 때 꺼내면 FALSE를 return )

# Protocol정의

* **// Server -> Client**
* **// 접속한 유저에게 혹은 접속해 있는 유저에게 방에 대한 정보를 보내는 패킷**
* **// int - roomCount**
* **// --------------------------------------------------------**
* **// int - RoomId**
* **// int - PlayerCountInRoom**
* **// int - RoomTitle데이터의 크기**
* **// TCHAR - RoomTitle**
* **// int - StateOfRoom(ex) 0:일반, 1(양수)게임중)**
* **//----------------------------------------------------------**
* **ㄴroomCount만큼 반복으로 들어가 있음**
* **SC\_LOBBY\_ROOMINFO,**

|  |  |
| --- | --- |
| // Server -> Client | 어디서 어디로 보내는 패킷인지를 표시 |
| // 접속한 유저에게 혹은 접속해 있는 유저에게 방에 대한 정보를 보내는 패킷 | 패킷에 대한 설명 |
| // --------------------------------------------------------  // int - RoomId  // int - PlayerCountInRoom  // int - RoomTitle데이터의 크기  // TCHAR - RoomTitle  // int - StateOfRoom(ex) 0:일반, 1(양수)게임중)  //----------------------------------------------------------  ㄴroomCount만큼 반복으로 들어가 있음 | 반복시작 라인 |
| 데이터형과 그 데이터가 나타내는 데이터의 종류를 표시  **TCHAR와 char는 문자열** |
| 반복끝 라인 |
| SC\_LOBBY\_ROOMINFO, | 패킷의 ID |