네프 프로토콜 구상 – 서버쪽 [2017-11-13-월~... 2017-12-10]

서버쪽은 사실상 클라이언트로부터 메시지를 받는 것에 대해서만 응답함.

클라이언트 -> 서버로 메시지 받음.

S_ClientThread [로그인전 제어] : 로그인, 회원가입, (강제)접속종료

- 로그인

[LOGIN]/id/pwd : 로그인 [아이디와 비밀번호를 받아서 로그인을 한다]

로그인에 성공하면, 이후는 S_WaitingRoom.controlStart() 에서 통제한다.

writer1 [LOGIN]/SUCCESS/id

writer2 [LOGIN]/SUCCESS or FAILURE

[LOGIN]/FAILURE/INVAILD_VALUE or ALREADY_CONNECTED

- 회원가입

[JOIN]/id/pwd : 회원가입 [아이디와 비밀번호를 받아서 회원가입을 한다]

writer2 [JOIN]/SUCCESS or FAILURE

[JOIN]/FAILURE/ALREADY_JOIN

- (강제)접속종료

[QUIT] : (강제) 접속 종료. 클라이언트의 접속 종료를 받음을 의미함.

/ 강제 접속 종료일 경우 catch 등 예외처리 문에 추가

S_WaitingRoom [대기실 제어] : 이름, 방[대기실 등] 입장, 방 생성, 채팅[메시지], (강제)접속종료

- 클라이언트 이름

[NAME]/userName : 연결되어 있는 클라이언트의 이름을 받음을 의미함.

- 방[대기실 등] 입장

[JOIN_ROOM]/roomNum/roomPwd

: 연결되어 있는 클라이언트가 들어갈 방의 번호와 비밀번호를 받음을 의미함.

- 이후,

방이 찼는지 검사하고, (찼으면 writer2 [JOIN_ROOM]/FAILURE/FULL명령어를 전송)

방의 비밀번호가 맞는지 검사한 후,

(찼으면 writer2 [JOIN_ROOM]/FAILURE/WRONGPWD명령어를 전송)

- 이후,

대기실 사람들에게 자신이 나옴을 알린다. **sendToRoom.** [EXIT]/userName

// 사용자의 새 방 번호를 지정한다.

// 사용자를 새로운 방에 넣는다.

사용자에게 방에 입장하였음을 알린다.

writer1 [JOIN_ROOM]/roomNumer/roomName/roomOwnerName

writer2 [JOIN_ROOM]/SUCCESS

새방의 다른 사람들에게 자신의 입장을 알린다. **sendToRoom.** [ENTER]/serName

사용자에게 새로운 방에 있는 사용자 이름 리스트를 전송한다.

sendTo [PLAYERS]/userName.....

// 이후는 S_GameRoom.controlStart()에서 통제함.

- 방 생성

[CREATE_ROOM]/roomName/roomPwd/userName : 클라리언트가 생성할 방의 이름과 비밀번호, 방을 생성한 방장의 아이디를 받음을 의미함.

- 이후,

// 대기실에서 나간다.

대기실 사람들에게 자신이 나옴을 알린다. **sendToOthers** [EXIT]userName

// 새로운 테트리스 방을 만든다.

// 자신의 방번호를 조정한다.

// 방을 만든 당사자는 만든 방에 들어간다.

// 방에 입장하였음을 알린다.

게임방에 있는 사람들에게도 자신이 들어감을 알린다. **sendToRoom** [ENTER]/userName

게임방에 있는 사람들의 이름 리스트를 전송한다. **sendTo** [PLAYERS]/userName 목록 ....

대기실의 다른 클라이언트들에게 새로운 방이 추가되었음을 알린다.

sendToRoom [CREATE_ROOM]/roomNum/roomName

// 이후는 GameRoomServer.controlStart()에서 통제함.

- 채팅[메시지]

[MSG]/message : 클라이언트의 메시지를 받음을 의미함.

sendToRoom 해당 방의 클라이언트들에게 메시지를 알려준다. [MSG]/message

- (강제)접속종료

[QUIT] : 클라이언트의 접속 종료를 받음을 의미함.

// 해당 클라이언트를 전체 및 대기실에서 나가게 한다.

대기실의 클라이언트들에게 userName가 접속을 끊었음을 알려준다.

sendToOthers [DISCONNECT]/userName

S_GameRoom[게임방 제어] : 채팅[메시지], 방 나가기, 레디, 자리(열고 닫기), 강퇴, 게임 시작, (강제)접속종료

- 채팅[메시지]

[MSG]/message : 클라이언트의 메시지를 받음을 의미함.

sendToRoom 해당 방의 클라이언트들에게 메시지를 알려준다. [MSG]/message

- 방 나가기

[EXIT_ROOM]/userName : 해당 클라이언트가 방을 나감

=> // 해당 클라이언크가 방장인지 확인한다.

-> if 방장이 아닐 경우,

// 해당 클라이언트만 방에서 나가게 한다.

sendToRoom. 게임방 사람들에게 자신이 나옴을 알린다. [EXIT]userName

// 사용자의 대기실 방 번호를 0으로 지정한다.

// 대기실로 보낸다.

사용자에게 방에서 나가서 대기실로 감을 알린다. **sendTo** [EXIT_ROOM]

대기실의 다른 사람들에게 자신의 입장을 알린다.

sendToRoom. [ENTER]/userName

사용자에게 대기실에 있는 사용자 이름 리스트를 전송한다.

sendTo [PLAYERS]/userName.....

// 루프를 빠져나온다. 이후는 S_WaitingRoom.controlStart()에서 다시 통제함.

-> else 방장일 경우,

// 방이 터진다. [터지기 전에 모든 사람들을에게 이를 알리고 방에서 나가게 해야한다]

사용자들에게 방이 터졌음을 알린다. **sendToRoomOther** [ROOM_DESTROYED]

// 기존 게임방에 속해있는 각각의 클라이언트들에 대해서 수행한다.

// 해당 방내의 모든 클라이언트를 방에서 나가게 한다.

// 사용자들의 대기실 방 번호 0으로 지정한다.

// 대기실에 사용자들을 넣는다.

대기실의 다른 사람들에게 자신의 입장을 알린다. **sendToRoom.** [ENTER]/userName

사용자에게 새 방에 있는 사용자 이름 리스트를 전송한다. [PLAYERS]/ userName.....

// 방을 터트린다.

// 루프를 빠져나온다. 이후는 S_WaitingRoom.controlStart()에서 다시 통제함.

- 레디 [당사자만이 보낼 수 있음]

[READY]/index/Ready or Not ready : index번째 해당 클라이언트가 게임을 준비함.

=> 해당 클라이언트 레디의 true/false 변환

=> 방의 클라이언트들에게 해당 클라이언트의 레디 값이 변경되었음을 알림.

sendToRoomOthers [READY]/index/Ready or Not ready

- 자리(열고 닫기) [방장만이 보낼 수 있음]

[PLACE]/index/Open or Close : index번째 자리가 열리고, 닫힘을 의미함.

=> 만약 Open일 경우, 제한인원이 증가하고, Close이면 제한인원이 감소함

=> 방의 클라이언트들에게 해당 자리 값이 변경되었음을 알림.

sendToRoomOthers [PLACE]/index/Open or Close

- 강퇴 [방장만이 보낼 수 있음]

[FIRED]/userName : userName인 클라이언트가 강퇴 당함을 의미.

=> // userName가 있던 방의 사람들에게 userName 클라이언트가 강퇴되었음을 전한다.

sendToRoom [FIRED]/userName

// userName인 클라이언트를 방에서 나가게 한다.

// userName인 클라이언트의 방 번호를 0으로 설정한다.

// userName인 클라이언트를 대기실로 보낸다.

userName인 클라이언트가 대기실에 들어갔음을 대기실 사람들에게 알린다.

sendToRoom [ENTER]/userName

사용자에게 대기실에 있는 사람들의 이름 리스트를 전송한다.

sendTo [PLAYERS]/ userName.....

- 게임 시작 [방장만이 보낼 수 있음]

[GAME_START]

=> 방장포함 2명이상인지 검사하고, 모두 레디를 박았는지 검사한다.

2명이상이면서, 모두 레디를 박았으면, 방에 게임시작 메시지를 보낸다.

writer1 [GAME_START]

writer2 [GAME_START]/SUCCESS

[GAME_START]/FAILURE/NOT_START_ALONE or NOT_READY_ALL

방장 혼자밖에 없을 경우, 게임을 시작할 수 없다.

한명이라도 레디를 안 박으면, 게임을 시작할 수 없다.

- (강제)접속종료

[QUIT] : 클라이언트의 접속 종료를 받음을 의미함.

// 해당 클라이언트를 전체 및 대기실에서 나가게 한다.

대기실의 클라이언트들에게 userName가 접속을 끊었음을 알려준다.

sendToOthers [DISCONNECT]/userName

[게임은 서바이벌 모드로 진행. 점수 상관없이 오래 사는 놈이 이김]

S_TetrisRoom [테트리스 시작한 방 제어] : 테트리스 명령어, 채팅, ~~이김~~, 게임오버[대기] & 게임 종료, 방 나가기, (강제)접속종료

- 테트리스 명령어

[TETRIS_CMD]/userName/cmd [cmd = 해당 명령어. EX LEFT, RIGHT, UP, DOWN, Q 등]

: 클라이언트로부터 받은 테트리스 명령어를 의미함.

=> 해당 방의 클라이언트들에게 메시지를 알림

sendToRoomOther [TETRIS_CMD]/userName/keyCode

~~- 채팅[메시지]~~

~~[MSG]/message : 클라이언트의 메시지를 받음을 의미함.~~
~~=> 해당 방의 클라이언트들에게 메시지를 알림~~
~~**sendToRoom** 해당 방의 클라이언트들에게 메시지를 알려준다. [MSG]/message~~

~~- 이김~~

- 게임오버[대기] & 게임 종료

[GAME_OVER]/userName : 해당 클라이언트가 게임오버 당함
=> // 해당 클라이언트를 게임오버 처리시킨다.
if 모두 게임오버가 아닐 경우, 해당 방의 클라이언트들에게 메시지를 알려준다.
sendToRoom [GAME_OVER]/userName
else 만약, 해당 방의 모든 클라이언트들이 게임오버 당했다면,
승자의 이름과 함께 게임이 종료되었다고 알림.
이 경우, 맨 마지막에 게임오버 당한 사람이 승자이다.
sendToRoom [GAME_END]/winnerName
=> 이후
방에 있는 클라이언트들을 모두 대기실로 나오게 하고, 방을 없애야 한다.
방이 터졌을 때의 과정과 동일. [기존 방에서 나갈 때 메시지를 전송할 필요는 없지만...]

- 방 나가기

[EXIT_ROOM]/userName : 해당 클라이언트가 방을 나감
=> // 해당 클라이언트가 방을 나간다.
if 해당 클라이언트가 아직 게임오버 당하지 않았을 경우,
// 해당 클라이언트를 게임오버 처리시킨다.
if 모든 클라이언트들이 게임오버 당했었다면, -> 최후의 1인이 방을 나간 것임.
// 방이 터진다. // 어차피 남은 사람도 없다.
else 아직 플레이 하고 있는 사람이 있을 경우,
방에 해당 클라이언트가 게임을 포기하고 나갔음을 알린다.
sendToRoom. [GIVE_UP]/userName
else 해당 클라이언트가 이미 게임오버 당했었다면,
게임방에 해당 클라이언트가 나감을 알린다. **sendToRoom.** [EXIT]/userName

// 사용자의 방 번호를 0으로 지정한다.
// 대기실로 보낸다.
사용자에게 방에서 나가서 대기실로 감을 알린다. **sendTo** [EXIT_ROOM]
대기실의 다른 사람들에게 자신의 입장을 알린다. **sendToRoom.** [ENTER]/userName
사용자에게 새 방에 있는 사용자 이름 리스트를 전송한다.
sendTo [PLAYERS]/ userName.....
// 루프를 빠져나온다. 이후는 S_WaitingRoom.controlStart()에서 다시 통제함.

- (강제)접속종료

[QUIT] : 클라이언트의 접속 종료를 받음을 의미함.

// 해당 클라이언트를 전체 및 대기실에서 나가게 한다.

대기실의 클라이언트들에게 userName가 접속을 끊었음을 알려준다.

sendToOthers [DISCONNECT]/userName