컴퓨터공학특론2

프로젝트

담당교수: 문대경 교수님

컴퓨터공학과

60202029 이정호

1. Client

main 함수에서 소켓을 열고 connect를 한 후 실행문을 실행하고 닫아줍니다. 실행문 전에는 유저 아이디를 받습니다. 빈칸의 경우는 유저아이디로 취급하지 않습니다. 실행문은 무한 루프문으로 입력을 항상 받습니다. 그후 구현한 split 함수를 통해 ‘ ‘기준으로 문자열을 자릅니다. 명령어가 옳은 지 비교한 후, 옳다면 Json형태로 바꿔줍니다. sendJsonString 함수는 rapidjson으로 만들어진 document와 array를 가져오고 또한 보낼 sock과 명령어를 가져옵니다. 이를 통해 { “command”:”명령어”, “argument”:[]} 형태의 json을 만들어줍니다. 그리고 sendJsonString 내부에서 이러한 json을 string형태로 sock에 보내줍니다. bot 명령어의 경우 bot 함수를 실행시키며 bot 함수는 random한 정수를 뽑아 명령을 실행시킵니다. chatBot의 경우 chat할 유저 이름을 서버에서 정하기 위해 만들어진 명령입니다. main함수에서 실행문외에 하나의 thread를 만듭니다. 이 thread는 항상 sock을 통해 받은 데이터를 읽고 수행합니다. 수행하는 종류에는 어디선가 일어난 공격을 알려주는 것, GameOver를 알려주는 것, Item을 얻는 것, 몬스터의 좌표를 출력하는 것, 유저를 출력하는 것, chat의 유저이름이 틀렸다는 것, chat message를 출력하는 것, 잡다한 것을 출력하는 것이 있습니다. 프로젝트의 명령 외에도 potion hp, potion str의 명령이 있습니다. 이것은 hp 포션을 쓰는 것, str 포션을 쓰는 것을 의미합니다.

2. Server

Server는 작업을 처리하는 thread pool 과 몬스터의 gen을 맡는 thread, 몬스터의 attack을 맡는 thread가 있습니다. 또한 select구문을 이용해서 io multiplexing을 사용합니다. 그리고 main의 경우 계속해서 입력을 받아 jobqueue에 넘겨주고 thread pool은 jobqueue의 작업을 가져가서 처리합니다. thread pool에서는 processClient를 실행하는데 이는 데이터를 입력 받고 그에 맞는 명령을 실행하는 역할을 합니다. Json은 위의 설명대로 받아오며 이러한 json을 가지고 명령을 비교하게 됩니다. move 명령은 이동을 attack의 경우 자신의 좌표 중심으로 사각형의 9칸을 때립니다. 이때 누군가 맞은 경우 printAttack 함수를 통해 client 모두에게 전체방송을 하게 됩니다. 그리고 만약 몬스터가 죽은 경우 그 몬스터의 데이터를 모두 삭제하고 가지고 있던 hp, str 포션들을 그 유저가 가져가게 됩니다. sendJsonString 함수를 통해 그 유저에게 아이템 획득에 대하여 알려줍니다. sendJsonString 함수를 통해 모든 데이터를 client에게 전달하게 되며 모든 자료형은 위의 json대로 진행됩니다. user이름의 경우 client class에 저장하게 됩니다. 이로 인해 monsters, users 명령을 처리할 수 있습니다. chat과 chatBot의 경우처럼 유저 id가 필요한 경우에는 activeClients를 순회하며 client에 저장된 userId를 찾게 됩니다. login의 경우 만약 기존에 login이 되어있어 데이터가 있다면 소켓을 끊고 데이터를 갱신합니다. Potion의 경우 hp 혹은 str 포션인지를 체크하고 맞게 입력했다면 포션의 소유유무를 파악하고 사용합니다. hp의 경우는 증가하고 str의 경우 1분간 str이 6이되며 재사용시 시간만 늘어나는 구조로 설계했습니다.

Gen의 경우 monster의 id를 0~9로 주며 없는 번호를 채워 넣는 방식으로 진행됩니다.

Attack의 경우 몬스터 별로 진행되며 activeClient를 통해 유저 아이디를 가져오고 그를 이용하여 redis에 user를 검색하고 hp를 조정합니다. 만약 hp가 0이하라면 user의 모든 데이터를 삭제하고 sock과의 연결을 중단합니다. 사용자의 종료를 알기 위해 client의 소멸자에 expire를 300으로 설정하였고 삭제하는 경우도 존재하므로 미리 user 데이터의 존재 여부를 확인합니다. 접속을 5분이내 다시하였다면 사용중이던 힘포션을 제외한 모든 데이터는 유지되며 사용중이던 힘포션의 효과는 제거됩니다.

3. 보너스

server에서 설명한 것 외에 보너스를 위해 processClientHttp함수를 만들고 sendJsonStringHttp함수를 만들었습니다. 처리방식은 위와 동일하지만 processClientHttp함수의 경우 rest api를 사용하여 명령을 받습니다. 또한 sendJsonStringHttp의 경우도 마찬가지로 rest api의 응답을 위해 만들어졌습니다. Client에 serverPort라는 변수를 추가해 어느 포트에서 연결했는지 알아보고 http인지 아닌지 workerThreadProc안에서 비교하여 processClient나 processClientHttp로 보냅니다. sendJsonStringToAll의 경우도 rest api를 사용하는 client를 제외한 곳에만 정보를 보냅니다. sendErrorHttp의 경우 not found를 보내주는 함수입니다. createPassiveSocket에 port번호를 받게 하여 소켓을 두개 열 때 편리하도록 하였습니다.

참고로 attack의 경우 client에만 전체방송을 해줍니다. 모든 에러는 Not Found로 처리했습니다.

4. 명령어 목록

|  |  |
| --- | --- |
| Client 명령어 | Rest api 명령어 |
| move x y  ex) move 1 2 | 메소드: POST URL: /move  \*Body:  username 이름  x 좌표  y 좌표 |
| attack | 메소드: POST URL: /attack  \*Body:  username 이름 |
| monsters | 메소드: GET URL: /monsters |
| users | 메소드: GET URL: /users |
| chat username text  ex) chat aaa 안녕하신가 | 메소드: POST URL: /chat  \*Body:  username 이름  to 수신인  text 텍스트 |
| bot |  |
| potion kind  ex) potion hp  ex) potion str | 메소드: POST URL: /potion  \*Body:  username 이름  kind 종류 |

\* POST의 body의 경우 x-www-form-urlencoded 로 전송해야 합니다.