


| 일반 소스 공유 | >

# [소스공유] 채팅기록 저장 && 불러오기


 **프로그** 챗봇 고수  1:1 채팅

2019.12.22. 20:44 조회 409

 댓글 8

 URL 복사



 첨부파일 1

공유할 소스 용도

=> 채팅기록 저장 && 불러오기

지금까지 질문만 한것 같아서 간단한 소스 하나 공유해 봅니다. 주석없는 소스는 아래 첨부해 두었습니다.(고인물분들께 서는 주석 설명이 틀려도 너그럽게 용서(?)해 주시면 감사하겠습니다.)

제가 만들면서 참고한 게시글들입니다. 초보분들은 이 게시글 먼저 보시고 소스 보시면 이해하시기 수월하실 거예요.

파일 생성, 저장, 불러오기, 삭제에 관한 게시글: <https://cafe.naver.com/nameyee/3513>

객체 개념을 잡아주는 게시글: <https://cafe.naver.com/nameyee/4908>, <https://cafe.naver.com/nameyee/8232>

## 소스 코드

```
=>
fs = FileStream
Lw="Wu200b".repeat(500) //공백문자를 500번 사용한다는 뜻으로 전달하는 메세지를 전체보기...형식으로 전송합니다
function response(room, msg, sender, isGroupChat, replier, ImageDB, packageName, threadId){
var day = new Date();
var today = day.getDay(); //오늘의 요일 불러오기(일요일은0, 월요일은1,화요일은2... 형식입니다)
var 시 = day.getHours(); //시각
var 분 = day.getMinutes(); //분
msg = msg.replace(/Wu202E/g,"")
sender = sender.replace(/Wu202E/g,"")
path = "sdcard/chatreport/"+room+".txt" //대화내용을 저장할 경로입니다. sdcard/chatreport에 객체형식으로 저장됩니다.
path1 = "sdcard/chatreport/count.txt" //횟수(아래 나옵니다)를 저장할 경로입니다. sdcard/chatreport에 count.txt라는 이름으로 저장됩니다.
path2 = "sdcard/chatreport/type2/"+room+".txt" //대화내용을 저장하는 경로입니다. sdcard/chatreprot/type2에 {}형식으로 저장됩니다.
path3 = "sdcard/chatreport/day.txt" //요일을 저장할 경로입니다. sdcard/chatreport에 day.txt라는 이름으로 저장됩니다.

if(room=="원하는방1" || room=="원하는방2"){
if(fs.read(path)==null) fs.write(path,"{}"); //path에 저장된 파일이 없다면 객체 형식으로 생성합니다.
if(fs.read(path1)==null) fs.write(path1,"{}");
if(fs.read(path2)==null) fs.write(path2,"{}");
if(fs.read(path3)==null) fs.write(path3,"{}");
json = JSON.parse(fs.read(path)); //json에 path에 저장된 내용을 집어넣습니다.
jsin = JSON.parse(fs.read(path1));
jsona = JSON.parse(fs.read(path2));
dday = JSON.parse(fs.read(path3));

if(json[sender] == undefined) { //만약 json에서 sender의 값이 정의되지 않았다면
json[sender] = {}; //sender의 값을 객체형식으로 생성합니다.({보낸이:{}})
json[sender]["name"] = sender; //sender의 값인 name의 값을 sender로 정의합니다.({보낸이:{name: "보낸이"}})
json[sender]["chat"] = []; //sender의 또다른 값인 chat의 값을 배열형식으로 생성합니다.({보낸이:{name: "보낸이","chat": []}})
}
```

```

if(jsin[sender] == undefined) {
jsin[sender] = {}; //sender의 값을 객체형식으로 생성합니다.({"보낸이":{}})
jsin[sender]["delete"] = 0; //sender의 값인 delete의 값을 0으로 놓습니다.({"보낸이":{"delete":0}})
}

if(jsona[room] == undefined) jsona[room] = []; //jsona에서 room의 값이 지정되지 않았다면 room의 값을 배열형식으
로 생성합니다. ({"방이름":[]})

if(dday["day"] == undefined) {
dday["day"] = today; //"day"의 값을 today(위에서 정의했습니다.)로 지정해줍니다.({"day":today의 값})
}

fs.write(path3, JSON.stringify(dday)) //path3에 위치한 문서(day.txt)에 오늘의 날짜를 업데이트(?) 해 줍니다.
check = dday["day"] //check에 "day"의 값을 넣어줍니다.
if(check != today) { //만약 원래 기록되어 있던 "day"의 값과 today의 값이 다르다면(요일이 다르다면)
dday["day"] = today //"day"의 값을 today값으로 바꿔줍니다.
fs.write(path3, JSON.stringify(dday)) //변경사항을 path3에 위치한 문서에 업데이트합니다.
fs.remove(path1); //횃수를 초기화합니다.(path1에 저장된 문서를 제거합니다.)
}

json[sender]["chat"].push(["["+sender+"]"+" ["+시+":"+분+" "+msg) //sender의 값인 "chat"의 값에 '[sender] [몇시:몇
분] 채팅내용' 형식으로 집어넣습니다.({보낸이:{name: "보낸이","chat": ["[보낸이] [몇시:몇분] 대화내용"]}})
fs.write(path, JSON.stringify(json)) //path에 저장된 문서에 채팅기록을 실시간으로 업데이트합니다.
jsona[room].push(["["+sender+"]"+" ["+시+":"+분+" "+msg)
fs.write(path2, JSON.stringify(jsona))
}

try{
if((room == "원하는방1" || room == "원하는방2") &&msg.startsWith("!채팅.)) { //특정 유저가 보낸 메세지를 최신순->오래된순 형식으
로 보여줍니다.
nickname = msg.substr(4) //메세지의 4번째단어까지 지웁니다.(지워지는 단어: !채팅.)
chat = json[nickname]["chat"] //chat에 저장된 nickname(유저)의 값인 "chat"의 값을 집어넣습니다.
chat1 = chat.reverse().slice(0,11) //chat1에 특정유저가 보낸 최근 메세지 10개를 집어넣습니다.
if(chat.length>1) { //만약 caht의 길이가 1보다 길다면
replier.reply(nickname+"님이 최근에 보낸 채팅입니다(최신순):\n"+Lw+"\n"+chat1.join("\n\n")) //chat1을 출력합니
다.
} else { //아니면(1보다 길지 않다면)
replier.reply("해당 유저의 채팅기록이 존재하지 않습니다!")
}
}

if((room == "원하는방1" || room == "원하는방2") &&msg.startsWith("!채.)) { //특정 유저가 보낸 메세지를 오래된순->최신순 형식
으로 보여줍니다.
nickname = msg.substr(3)
chat = json[nickname]["chat"]
chat1 = chat.reverse().slice(0,11)
if(chat.length>1) {
replier.reply(nickname+"님이 최근에 보낸 채팅입니다(오래된순):\n"+Lw+"\n"+chat1.reverse().join("\n\n"))
} else {
replier.reply("해당 유저의 채팅기록이 존재하지 않습니다!")
}
}

if((room == "원하는방1" || room == "원하는방2") &&msg.startsWith("!!채팅.)) {

```

```

nickname = msg.substr(5)
chat = json[nickname]["chat"]
chat1 = chat.reverse().slice(0,21) //chat1에 특정 유저가 보낸 메세지 20개를 집어넣습니다.
if(chat.length>1) {
replier.reply(nickname+"님이 최근에 보낸 채팅입니다(최신순):\n"+Lw+"\n"+chat1.join("\n\n"))
}else{
replier.reply("해당 유저의 채팅기록이 존재하지 않습니다!");
}
}

if((room=="원하는방1" || room=="원하는방2") &&msg.startsWith("!!챗.")) {
nickname = msg.substr(4)
chat = json[nickname]["chat"]
chat1 = chat.reverse().slice(0,21)
if(chat.length>1) {
replier.reply(nickname+"님이 최근에 보낸 채팅입니다(오래된순):\n"+Lw+"\n"+chat1.reverse().join("\n\n"))
}else{
replier.reply("해당 유저의 채팅기록이 존재하지 않습니다!");
}
}

} catch(e) {
replier.reply("존재하지 않는 유저입니다! 이름을 다시 확인해주세요.")
}

if((room=="원하는방1" || room=="원하는방2") &&msg=="!!채팅") { //현재 방에서 온 메세지를 최신순->오래된순 형식으로 보
여줍니다.
chat = jsona[room] //chat에 jsona의 room의 값을 집어넣습니다.
chat1= chat.reverse().slice(1,11) //chat1에 현재방에서 최근에 온 메세지 10개를 집어넣습니다.
if(chat.length>1) {
replier.reply("방 "+room+"의 최근 채팅리스트입니다(최신순):\n"+Lw+"\n"+chat1.join("\n\n"))
} else {
replier.reply("채팅방의 채팅기록이 존재하지 않습니다!");
}
}

if((room=="원하는방1" || room=="원하는방2") &&msg=="!!챗") { //현재 방에서 온 메세지를 오래된순->최신순 형식으로 보여
줍니다.
chat = jsona[room]
chat1= chat.reverse().slice(1,11)
if(chat.length>1) {
replier.reply("방 "+room+"의 최근 채팅리스트입니다(오래된순):\n"+Lw+"\n"+chat1.reverse().join("\n\n"))
} else {
replier.reply("채팅방의 채팅기록이 존재하지 않습니다!");
}
}

if((room=="원하는방1" || room=="원하는방2") &&msg=="!!채팅") { //현재 방에서 온 메세지를 오래된순->최신순 형식으로 보
여줍니다.
chat = jsona[room]
chat1 = chat.reverse().slice(1,21) //chat1에 현재방에서 최근에 온 메세지 20개를 집어넣습니다.
if(chat.length>1) {
replier.reply("방 "+room+"의 최근 채팅리스트입니다(최신순):\n"+Lw+"\n"+chat1.join("\n\n"))
} else {
replier.reply("채팅방의 채팅기록이 존재하지 않습니다!");
}
}

```

```

}
}

if((room == "원하는방1" || room == "원하는방2") && msg == "!!채팅") {
  chat = jsona[room]
  chat1 = chat.reverse().slice(1,21)
  if(chat.length > 1) {
    replier.reply("방 "+room+"의 최근 채팅리스트입니다(오래된순):Wn"+Lw+"Wn"+chat1.reverse().join("WnWn"))
  } else {
    replier.reply("채팅방의 채팅기록이 존재하지 않습니다!");
  }
}

if(room == "원하는방1" || room == "원하는방2") {
  chat = jsona[room]
  if(chat.length > 500) { //만약 chat의 길이가 500을 넘는다면(저장된 대화량이 500을 넘는다면)
    replier.reply("채팅수가 500을 넘어 채팅기록이 자동으로 삭제 되었습니다.")
    fs.remove(path2) //해당방의 채팅기록을 자동으로 삭제합니다.
    fs.remove(path)
  }
}

if((room == "원하는방1" || room == "원하는방2") && msg.startsWith("!삭제")) { //저장된 메시지를 삭제하는 명령어입니다.
  chat = json[sender]["delete"]++ //!삭제 명령어를 사용했다면 sender의 값인 "delete"의 값을 +1 해주고, 이를 chat에 집
  어넣습니다.
  fs.write(path1, JSON.stringify(json)) //바뀐정보를 path1에 위치한 문서에 업데이트해줍니다.
  if(chat < 2) { //만약 chat이("delete"의 값) 2보다 작다면
    fs.remove(path) //현재방의 채팅기록을 삭제합니다.
    fs.remove(path2)
    replier.reply("메세지가 성공적으로 삭제되었습니다")
  } else if(chat >= 2) { //만약 chat이 2보다 크다면(!삭제 명령어를 2번 이상 사용했다면) 삭제를 불가능하게 해 줍니다. 이
  는 번번한 삭제를 방지하기 위한 것이며, 사용여부는 자유입니다.
    replier.reply(sender+"님은 이미 2번이상 채팅을 삭제하셨습니다! 채팅삭제는 각 사람마다 하루에 최대 2번 가능합니
    다.")
  }
}
}
}

```



프로그래밍의 게시물 더보기 >

❤️ 좋아요 3 💬 댓글 8

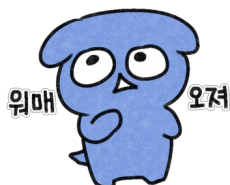
🔗 공유 | 신고

댓글 등록순 최신순 🔄

댓글알림 ☐



별명을 입력해주세요



2019.12.22. 20:46 답글쓰기



doami2005

rm = '(room == "원하는방1" || room == "원하는방2")'

와

if(rm) {

이거 제대로 작동 안합니다

문자열은 boolean으로 나타내면 무조건 참이기 때문에...(공백이 아닌 이상)

room조건 성립 안하고 모든 방에 적용됩니다

2019.12.22. 21:23 답글쓰기



줄러

zzzzzzzzzzzz

2019.12.22. 21:24 답글쓰기



프로그 작성자

그렇군요..수정하겠습니다

2019.12.22. 21:26 답글쓰기



프로그 작성자

수정했습니다. 지적 감사합니다



2019.12.22. 21:32 답글쓰기



doami2005

프로그



2019.12.22. 21:32 답글쓰기



doami2005

프로그 참고로 이프문은 안에 있는 것이 참이지만 보기 때문에  
rm에서 ' 2개만 뺏어도 작동할 거예요

2019.12.22. 21:34 답글쓰기



프로그 작성자

doami2005 하나 더 배워갑니다. 감사드려요 ㅎㅎ

2019.12.22. 21:34 답글쓰기

Hibot

댓글을 남겨보세요



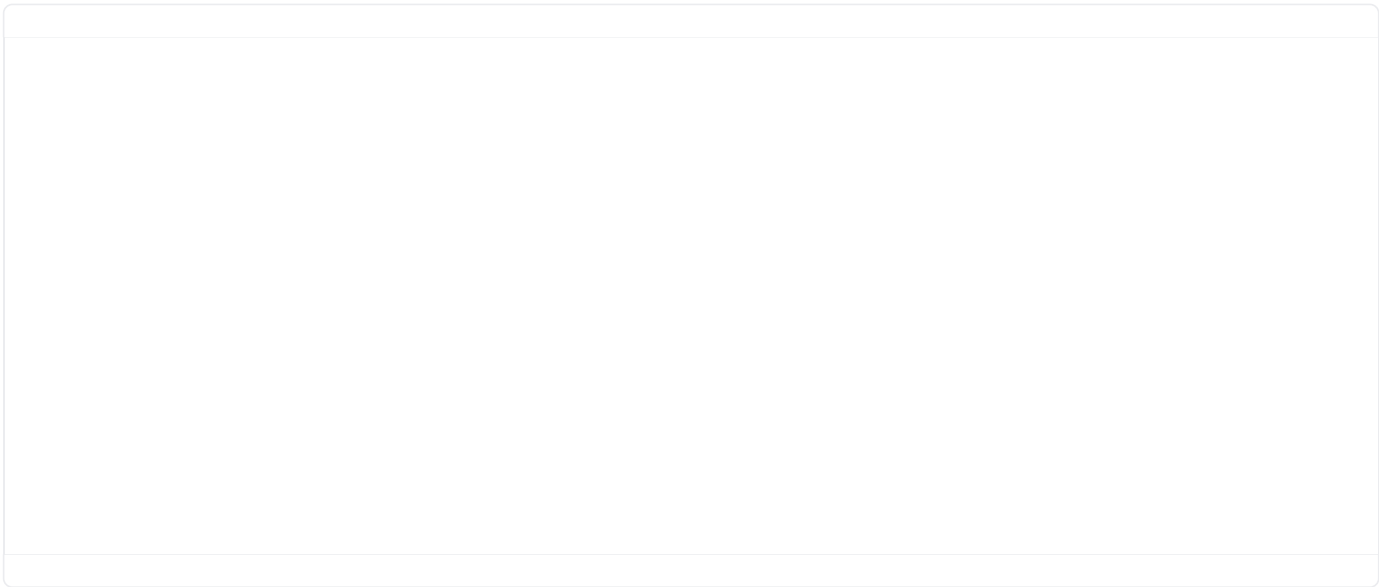
등록

글쓰기

답글

목록

▲ TOP



'| 일반 소스 공유 |' 게시판 글

이 게시판 새글 구독하기 ☐

명단 - 리스트 🗨️ [20]	NotreDame	2019.12.23.
그저 만들어본 특정한 사람이 특정한 말을 하면 반응하는 소스 [1]	jerome012342	2019.12.23.
[소스공유] 채팅기록 저장 && 불러오기 [8]	프로그	2019.12.22.
공지 기능 [6]	죽창인데요	2019.12.22.
메시	알랑뽕까	2019.12.22.

전체보기

이 카페 인기글

오픈이발 괴롭히기(메봇R)

ERROR  
♡0 💬7

가르치기 리로드..

Milk2  
♡0 💬9



혹시 여기서 변수를 저장하는 부분이 어딘지

[ 치\*\*, 한\*\* ] 님 강제 탈퇴

AlphaDo  
♡1 💬2

카톡방 나가기

hajuhee01  
♡0 💬15

마녀사냥 금지

AlphaDo  
♡3 💬8

안녕하세요

tomohong  
♡1 💬3

path질문

젤리파덕  
♡0 💬2

타 메신저 사용시 무한반복

NaN  
♡0 💬1