| 자유 게시판 | >

소스 오류?질문

진워커 챗봇 중수 1:1 채팅
2020.06.13. 20:33 조회 103

끝말잇기 소스인데 끝말잇기 시작하면 잠시후 시작합니다 라고뜨고 시작을 안하네요 뭐가문제일까요

```
importPackage(javax.net.ssl);
importPackage(java.lang);
importPackage(java.net);
importPackage(java.io);

importPackage(org.jsoup);

/* ------------------------------------------------------------------------------ */

const sdcard = android.os.Environment.getExternalStorageDirectory().getAbsolutePath();

const GAME_TIMER_OUT = 15;
const GAME_WORD_FILTER = [];
const ROOM_TIMER_OUT = 60;
const BOT_DELAY_TIME = 1;

/* ------------------------------------------------------------------------------ */

let gamePlayerList = [], // 플레이어 목록
 gamePlayerData = {}, // 플레이어 데이터

 gameStartTime = 0, // 게임 시작 시간
 gamePlayTime = 0, // 게임 플레이 시간

 gameUsedWord = [], // 사용된 단어
 gameLastChar = "", // 매칭 확인용 글자 (사[과] -> [과]자)
 gamePower = false, // 게임 전원
 gameRound = 1, // 게임 라운드
 gameStop = true, // 게임 일시정지
 gameTurn = 1, // 게임 턴

 modeOwnCom = true, // 한방단어 모드

 gameTimerCount = 0, // 타이머 카운트
 gameTimerStop = false, // 타이머 일시정지
 gameTimerPower = true, // 타이머 전원

 roomName = "", // 방 이름
 roomCreat = false, // 방 생성 여부
 roomTimerCount = 0, // 방 타이머 카운트
 roomTimerPower = true, // 방 타이머 전원

 aiLevel = 0, // AI 난이도
 aiName = "AI:", // AI 이름
```

```
 aiCreat = false, // AI 생성 여부
 aiPower = false; // AI 전원

/* ---------------------------------------------------------------------------- */

const Bot = {};

const PHP =
{
 getData : function(type, value)
 {
  // Https 허용

  if (android.os.Build.VERSION.SDK_INT > 9)
  {
   var policy = new android.os.StrictMode.ThreadPolicy.Builder().permitAll().build();
    android.os.StrictMode.setThreadPolicy(policy);
  }

  // javax.net.ssl.SSLHandshakeException (SSL 인증서 오류 방지)

  let sslContext = SSLContext.getInstance("SSL");
  sslContext.init(null, [new JavaAdapter(X509TrustManager, {
   getAcceptedIssuers : () => { return null; },
   checkClientTrusted : () => { return ; },
   checkServerTrusted : () => { return ; },
  })], new java.security.SecureRandom());

  HttpsURLConnection.setDefaultSSLSocketFactory(sslContext.getSocketFactory());
  HttpsURLConnection.setDefaultHostnameVerifier(new JavaAdapter(HostnameVerifier, {verify : (hostname, session)
=> { return true; }}));

  let connect = new URL("https://ttyy3388.josbar.io/getData.php?type=" + type + ((value) ? "&value=" + value :
"")).openConnection();

  if(connect != null)
  {
   connect.setConnectTimeout(5000);
   connect.setUseCaches(false);

   var inputStreamReader = new InputStreamReader(connect.getInputStream()),
    bufferedReader = new BufferedReader(new InputStreamReader(connect.getInputStream())),
    text = bufferedReader.readLine(), line = "";

   while((line = bufferedReader.readLine()) != null)
   {
    text += "₩n" + line;
   }

   inputStreamReader.close();
   bufferedReader.close();
   connect.disconnect();
  }

  return text.toString();
```

```
  return text.toString();
 },

 postData : function(type, value)
 {
  Jsoup.connect("https://ttyy3388.josbar.io/postData.php").data("type", type, "value",
value).method(Connection.Method.POST).execute().parse();
 },
};

const DB =
{
 StartWord : {}, // JSON Object
 GameData : {}, // JSON Object
 WordList : [], // JSON Array

 load : () =>
 {
  DB.StartWord = JSON.parse(PHP.getData("StartWord", null));
  DB.WordList = JSON.parse(PHP.getData("WordList", null));
  DB.GameData = JSON.parse(PHP.getData("GameData", null));
 },

 /* ---------------------------------------------------------------------------- */

 getWordMean : function(word)
 {
  return ((DB.isWord(word)) ? JSON.parse(PHP.getData("WordMean", word)) : null);
 },

 getDoumChar : function(lastChar)
 {
  let data = lastChar.charCodeAt() - 0xAC00;
  if (data < 0 || data > 11171) return ;

  const RIEUL_TO_NIEUN = [4449, 4450, 4457, 4460, 4462, 4467];
  const RIEUL_TO_IEUNG = [4451, 4455, 4456, 4461, 4466, 4469];
  const NIEUN_TO_IEUNG = [4455, 4461, 4466, 4469];

  let onset = Math.floor(data / 28 / 21) + 0x1100,
   nucleus = (Math.floor(data / 28) % 21) + 0x1161,
   coda = (data % 28) + 0x11A7, isDoumChar = false, doumChar;

  if (onset == 4357)
  {
   isDoumChar = true;
   (RIEUL_TO_NIEUN.indexOf(nucleus) != -1) ? onset = 4354 : (RIEUL_TO_IEUNG.indexOf(nucleus) != -1) ? onset =
4363 : isDoumChar = false;
  }
  else if (onset == 4354)
  {
   if (NIEUN_TO_IEUNG.indexOf(nucleus) != -1)
   {
    onset = 4363;
    isDoumChar = true;
```

```
  }
 }
 if (isDoumChar)
 {
  onset -= 0x1100; nucleus -= 0x1161; coda -= 0x11A7;
  doumChar = String.fromCharCode(((onset * 21) + nucleus) * 28 + coda + 0xAC00);
 }

 return doumChar;
},

getWordMeanMessage : function(word)
{
 if (DB.isWord(word))
 {
  let mean = DB.getWordMean(word)[0];
  return (mean.length > 30) ? mean.substr(0, 25) + ".." : mean;
 }
 else
 {
  return null;
 }
},

getWordMeanListMessage : function(word)
{
 if (DB.isWord(word))
 {
  let meanList = DB.getWordMean(word);
   len = meanList.length, text = [];

  for (i = 0 ; i < len ; i ++)
  {
   text[i] = "「" + java.lang.String.format("%03d", Integer(i + 1)) + "」「명사」 " + meanList[i];
   do { text[0] += "₩u200b".repeat(500); } while (false)
  }

  return text.join("₩n₩n");
 }
 else
 {
  return null;
 }
},

getRandomWord : function()
  {
 return DB.WordList[Math.floor(Math.random() * DB.WordList.length)];
},

getRandomStartWord : function()
{
 let list = Object.keys(DB.StartWord);

 while (true)
```

```
 while (true)
 {
  result = list[Math.floor(Math.random() * list.length)];

  if (DB.StartWord[result].length > 1000)
  {
   break;
  }
 }

 return result;
},

getFirstChar : function(word)
{
 return word[0];
},

getLastChar : function(word)
{
 return word[word.length - 1];
},

getLastCharMessage : function(lastChar)
{
 return lastChar = (DB.getDoumChar(lastChar)) ? lastChar + "(" + DB.getDoumChar(lastChar) + ")" : lastChar;
},

/* ---------------------------------------------------------------------- */

isLoaded : function()
{
 return (Object.keys(DB.StartWord).length && Object.keys(DB.WordList).length &&
Object.keys(DB.GameData).length) ? true : false;
},

isWord : function(word)
{
 return DB.WordList.indexOf(word) != -1;
},

isUsedWord : function(word)
{
 return gameUsedWord.indexOf(word) != -1;
},

isStartWord : function(startWord)
{
 return DB.StartWord[startWord] ? true : false;
},

isOwnComWord : function(word)
{
 if (DB.isStartWord(DB.getLastChar(word))) return false;
 else return (DB.isStartWord(DB.getDoumChar(DB.getLastChar(word)))) ? false : ((modeOwnCom) ? false : true);
```

```
},

isForbiddenWord : function(word)
{
 return GAME_WORD_FILTER.indexOf(word) != -1;
},

isSuitableWord : function(word)
{
 if (!word)
 {
  Bot.replyRoom("단어를 입력 해 주세요.");
  return false;
 }
 else if (word.length < 2)
 {
  Bot.replyRoom("두 글자 이상의 단어를 입력 해 주세요.");
  return false;
 }
 else if (DB.isForbiddenWord(word))
 {
  Bot.replyRoom(word + "(은)는 금칙어 입니다.");
  return false;
 }
 else if (DB.isUsedWord(word))
 {
  Bot.replyRoom("이미 사용한 단어 입니다.");
  return false;
 }
 else
 {
  if (!DB.isWord(word))
  {
   Bot.replyRoom("₩"" + word + "₩" (은)는 명사가 아니거나 사전에 등록되지 않은 단어입니다.");
   return false;
  }
  else if (DB.isOwnComWord(word))
  {
   Bot.replyRoom("현재 한방 단어가 금지된 상태입니다.");
   return false;
  }
  else if (gameLastChar != DB.getFirstChar(word))
  {
   if (DB.getDoumChar(gameLastChar) == DB.getFirstChar(word))
   {
    return true;
   }
   else
   {
    Bot.replyRoom(DB.getLastCharMessage(gameLastChar) + "(으)로 시작하는 단어를 입력 해 주세요.");
    return false;
   }
  }
  else
  {
```

```
    {
     return true;
    }
   }
  },
};

const AI =
{
 getWord : function(lastChar)
 {
  let startWordData = DB.StartWord[lastChar],
   startDoumWord = DB.StartWord[DB.getDoumChar(lastChar)];

  if (startWordData)
  {
   if (startDoumWord)
   {
    startWordData = (startWordData + "," + startDoumWord).split(",");
   }
  }
  else
  {
   if (startDoumWord)
   {
    startWordData = startDoumWord;
   }
   else return null;
  }

  let startWordList = [], // @로 시작하는 단어 리스트
   startWordLastChar = [], // @로 시작하는 단어의 끝 글자
   startWordDoumChar = null, // @로 시작하는 단어의 끝 글자 (두음 적용)
   startWordStartWord = [], // @로 시작하는 단어의 끝 글자로 시작하는 단어
   startWordStartWordNum = [], // @로 시작하는 단어의 끝 글자로 시작하는 단어의 갯수

   easyValue = 0, normalValue = 0, hardValue = 0,
   easyCount = 0, normalCount = 0, hardCount = 0,
   easyLevel = [], normalLevel = [], hardLevel = [],
   easyWord = null, normalWord = null, hardWord = null, replyWord = null,

   /*
    easyValue = 51% ~ 100%
    normalValue = 21% ~ 50%
    hardValue = 00% ~ 20%
   */

   len = startWordData.length,
   maxValue = 0, count = 0, i;

  for (i = 0 ; i < len ; i ++)
  {
   startWordLastChar[i] = startWordData[i][startWordData[i].length - 1];

   if (gameUsedWord.indexOf(startWordData[i]) == -1)
```

```
    {
     startWordList[count ++] = startWordData[i];
    }
   }

   for (i = 0 ; i < count ; i ++)
   {
    startWordDoumChar = DB.getDoumChar(startWordLastChar[i]);
    startWordStartWord = DB.StartWord[startWordLastChar[i]];

    if (!startWordStartWord) { if (startWordDoumChar) startWordStartWord = DB.StartWord[startWordDoumChar]; }
    else { if (startWordDoumChar) startWordStartWord = (startWordStartWord + "," +
DB.StartWord[startWordDoumChar]).split(","); }

    startWordStartWordNum[i] = 0;

    if (startWordStartWord)
    {
     startWordStartWordNum[i] = startWordStartWord.length;

     if (maxValue < startWordStartWordNum[i])
     {
      maxValue = startWordStartWordNum[i];
     }
    }
   }

   easyValue = maxValue * 0.5;
   hardValue = maxValue * 0.2;

   len = startWordList.length;

   for (i = 0 ; i < len ; i ++)
   {
    if (easyValue < startWordStartWordNum[i])
    {
     easyLevel[easyCount ++] = startWordList[i];
    }
    else if (hardValue < startWordStartWordNum[i] && startWordStartWordNum[i] >= easyCount)
    {
     normalLevel[normalCount ++] = startWordList[i];
    }
    else
    {
     hardLevel[hardCount ++] = startWordList[i];
    }
   }

   let getWord = (array) =>
    {
     let data = [], len = array.length, count = 0, temp = "", i, j;

     for (i = 0 ; i < len ; i ++)
     {
      if (Object.keys(DB.GameData).indexOf(array[i]) != -1)
```

```
    if (Object.keys(DB.GameData).indexOf(array[i]) != -1)
    {
     data[count ++] = array[i];
    }
   }
   if (count == 1)
   {
    return data[0];
   }
   else if (count >= 2)
   {
    /*
    for (i = 0 ; i < count - 1 ; i ++)
    {
     for (j = i + 1 ; j < count ; j ++)
     {
      if (DB.GameData[data[i]].Num < DB.GameData[data[j]].Num)
      {
       temp = data[j];
       data[j] = data[i];
       data[i] = temp;
      }
     }
    } */

    return data[Math.floor(Math.random() * count)];
   }
   else
   {
    return array[Math.floor(Math.random() * len)];
   }
  };

 switch (aiLevel)
 {
  case 3 :

   replyWord = (hardCount) ? getWord(hardLevel) :
      (normalCount) ? getWord(normalLevel) :
      (easyCount) ? getWord(easyLevel) : null;
   break;

  case 2 :

   replyWord = (normalCount) ? getWord(normalLevel) :
      (easyCount) ? getWord(easyLevel) : null;
   break;

  case 1 :

   replyWord = (easyCount) ? getWord(easyLevel) : null;
   break;
 }

 return replyWord;
```

```
},

getReply : function(lastChar)
{
 let aiWord = AI.getWord(lastChar);

 if (!aiWord)
 {
  Thread.sleep(BOT_DELAY_TIME * 1000);

  if (gamePlayerList.length == 2)
  {
   Bot.replyRoom
   (
    "[ " + aiName + " ] 가 입력 할 단어가 없어 아웃됩니다.₩n" +
    "승자는 [ " + Game.getNextPlayer() + " ] 님 입니다!"
   );

   Game.off();
  }
  else
  {
   Game.setPlayerDelete(aiName);

   Bot.replyRoom("[ " + aiName + " ] 가 입력 할 단어가 없어 아웃됩니다.");

   Bot.replyRoom
   (
    "[ " + Game.getNowPlayer() + " ] 님은 새로운 단어를 입력해 주세요₩n₩n" +
    "< 남은 플레이어 >₩n" + Game.getPlayerListMessage()
   );
  }
 }
 else
 {
  gameLastChar = DB.getLastChar(aiWord);

  Thread.sleep(BOT_DELAY_TIME * 1000);

  Bot.replyRoom("[ " + aiName +" ] : " + aiWord);

  Bot.replyRoom
  (
   "< " + DB.getWordMeanMessage(aiWord) + " >₩n₩n" +
   "[ " + aiName + " ] 가 ₩"" + aiWord + "₩"단어를 입력했습니다.₩n" +
   "[ " + Game.getNextPlayer() + " ] 님은 ₩"" + DB.getLastCharMessage(gameLastChar) + "₩"(으)로 시작하는 단어를
입력해 주세요"
  );

  Game.setNextEvent(aiWord);
 }
},

getRandomReply : function()
{
```

```
{
 let randomWord = "";

 while (true)
 {
  randomWord = DB.getRandomWord();

  if (DB.isOwnComWord(randomWord) || DB.isUsedWord(randomWord))
  {
   randomWord = DB.getRandomWord();
  }
  else
  {
   AI.getReply(DB.getLastChar(randomWord));
   break;
  }
 }
},

isAITurn : function()
{
 if (aiPower && (Game.getNowPlayer() == aiName))
 {
  return true;
 }
}
};

const Game =
{
 start : function()
 {
  DB.load();

  gameStartTime =
   ((new Date().getHours() > 12) ? "오후 " +
   (new Date().getHours() - 12) : "오전 ") +
   new Date().getHours() + "시 " + new Date().getMinutes() + "분";

  gamePlayTime = new Date().getTime();
  gameLastChar = DB.getRandomStartWord();
  gamePower = true;

  Game.startGameTimer();
 },

 off : function()
 {
  gamePlayerData = [];
  gamePlayerList = [];
  gamePlayerLife = [];

  gameStartTime = 0;
  gamePlayTime = 0;
```

```
 gameUsedWord = [];
 gameLastChar = "";
 gamePower = false;
 gameFirst = true;
 gameTurn = 1;

 gameTimerCount = 0;
 gameTimerStop = false;
 gameTimerPower = false;

 roomName = "";
 roomCreat = false;
 roomTimerCount = 0;
 roomTimerPower = false;

 aiLevel = 0;
 aiName = "AI:";
 aiCreat = false;
 aiPower = false;
},

restart : function()
{
 for (let i of gamePlayerData)
 {
  gamePlayerData[i]['Score'] = 0;
  gamePlayerData[i]['Life'] = 3;
 }

 gameTimerCount = 0;
 gameUsedWord = [];
 gameLastChar = "";
 gameFirst = true;
 gameTurn = 1;

 gameTimerCount = 0;
},

/* ---------------------------------- Timer ---------------------------------- */

startGameTimer : function()
{
 gameTimerPower = true;

 new Thread
 ({
  run : function() { try
  {
   while (gameTimerPower)
   {
    Thread.sleep(1000);

    if (gameTimerCount >= GAME_TIMER_OUT)
    {
     Game.setPlayerTimeOut();
```

```
      Game.setPlayerTimeOut();
      gameTimerCount = 0;
     }
     else
     {
      gameTimerCount ++;

      if (gameTimerCount == 5)
      {
       Bot.replyRoom("10초 남았습니다.");
      }
     }
    }
   }
   catch(e)
   {
    Bot.reply
    (
     "GAME TIMER ERROR₩n₩n" +
     "Error : " + e
    );
   }
 }}).start();
},

startRoomTimer : function()
{
 roomTimerPower = true;

 new Thread
 ({
  run : function() { try
  {
   while (roomTimerPower)
   {
    Thread.sleep(1000);

    if ((roomTimerCount >= ROOM_TIMER_OUT) && (!gamePower))
    {
     Bot.replyRoom("60초가 지나 자동으로 방을 삭제합니다.");
     Game.off(); break;
    }
    else
    {
     roomTimerCount ++;
     (roomTimerCount == 30) ? Bot.replyRoom("30초 후 방이 삭제됩니다.") :
     (roomTimerCount == 50) ? Bot.replyRoom("10초 후 방이 삭제됩니다.") : null;
    }
   }
  }
  catch(e)
  {
   Bot.reply
   (
    "ROOM TIMER ERROR₩n₩n" +
```

```
    "Error : " + e
   );
  }
 }}).start();
},

stopGameTimer : function()
{
 gameTimerCount = 0;
 gameTimerPower = false;
},

stopRoomTimer : function()
{
 roomTimerCount = 0;
 roomTimerPower = false;
},

/* ---------------------------------------------------------------------------- */
/* ----------------------------------- Room ----------------------------------- */

setRoomCreat : function(room)
{
 roomName = room;
 roomCreat = true;
 roomTimerPower = true;
},

/* ---------------------------------------------------------------------------- */
/* ----------------------------------- Print ----------------------------------- */

getPlayerListMessage : function()
{
 let list = ["첫", "두", "세", "네", "다섯", "여섯", "일곱", "여덟", "아홉", "열"], text = [], len = gamePlayerList.length, i;
 for (i = 0 ; i < len ; i ++) { text[i] = "○ " + list[i] + "번째 : " + gamePlayerList[i]; }

 return text.join("₩n");
},

/* ---------------------------------------------------------------------------- */
/* ---------------------------------------------------------------------------- */

getPlayerPosition : function(player)
{
 return gamePlayerList.indexOf(player);
},

getNowPlayer : function()
{
  return gamePlayerList[gameTurn - 1];
},

getNextPlayer : function()
{
 return gamePlayerList[(gameTurn == gamePlayerList.length) ? 0 : gameTurn];
```

```
 return gamePlayerList[(gameTurn == gamePlayerList.length) ? 0 : gameTurn];
},

/* ---------------------------------------------------------------------------- */
/* ---------------------------------------------------------------------------- */

setPlayerAdd : function(player)
{
 gamePlayerList.push(player);

 gamePlayerData[player] =
 {
  'Score' : 0,
  'Life' : 3,
 }
},

setPlayerDelete : function(player)
{
 if (gameTurn == gamePlayerList.length) { Game.setNextTurn(); }

 gamePlayerList.splice(Game.getPlayerPosition(player), 1);
 gameTimerCount = 0;
},

setPlayerExit : function(player)
{
 let nowPlayer = Game.getNowPlayer(),
  nextPlayer = Game.getNextPlayer();

 if (gamePlayerList.length == 2)
 {
  Bot.replyRoom("승자는 [ " + nextPlayer + " ] 님 입니다!");
  Game.off();
 }
 else
 {
  Game.setPlayerDelete(player);

  if (nowPlayer == player)
  {
   Bot.replyRoom
   (
    "[ " + nextPlayer + " ] 님은 새로운 단어를 입력해 주세요₩n₩n" +
    "< 남은 플레이어 >₩n" + Game.getPlayerListMessage()
   );

   AI.isAITurn() ? AI.getRandomReply() : null;
  }
  else
  {
   if ((gameTurn - 1) > Game.getPlayerPosition(player))
   {
    gameTurn -= 1;
   }
```

```
    Bot.replyRoom("< 남은 플레이어 >₩n" + Game.getPlayerListMessage());
   }
  }
 },

 setPlayerTimeOut : function()
 {
  Game.setPlayerLife();
  (gamePower) ? Game.setPlayerScore(0, null) : null; // 0 : 뺌기, 1 : 더하기
 },

 setPlayerScore : function(value, word)
 {
  if (value)
  {
   gamePlayerData[Game.getNowPlayer()]['Score'] += (20 * (word.length - 1)) + Math.floor(Math.random() * (10 +
(gameRound / 10)))
  }
  else
  {
   let minusValue = Math.floor(Math.random() * (10 * (gameRound / 2)));

   if (gamePlayerData[Game.getNowPlayer()]['Score'] > 0)
   {
    if (gamePlayerData[Game.getNowPlayer()]['Score'] - minusValue < 0)
    {
     gamePlayerData[Game.getNowPlayer()]['Score'] = 0
    }
    else
    {
     gamePlayerData[Game.getNowPlayer()]['Score'] -= minusValue;
    }
   }
  }
 },

 // Life 감소 방식에서 Score 감소 방식으로 변경 (3번 라이프 감소 시 탈락으로)
 setPlayerLife : function()
 {
  let nowPlayer = Game.getNowPlayer(),
   nextPlayer = Game.getNextPlayer();

  gamePlayerData[nowPlayer]['Life'] -= 1;

  if (gamePlayerData[nowPlayer]['Life'] > 0)
  {
   Bot.replyRoom
   (
    "< 시간 초과 >₩n₩n" +
    "[ " + nowPlayer + " ] 님의 라이프가 1 감소합니다.₩n₩n" +
    "남은 라이프 : " + gamePlayerData[nowPlayer]['Life']
   );
  }
  else
```

```
 else
 {
  if (gamePlayerList.length == 2)
  {
   Bot.replyRoom
   (
    "< 시간 초과 >₩n₩n" +
    "[ " + nowPlayer + " ] 님이 라이프가 0이 되어 게임을 종료합니다.₩n" +
    "승자는 [ " + nextPlayer + " ] 님 입니다!"
   );
   Game.off();
  }
  else
  {
   Game.setPlayerDelete(nowPlayer);

   Bot.replyRoom
   (
    "< 시간 초과 >₩n₩n" +
    "[ " + nowPlayer + " ] 님이 라이프가 0이 되어 아웃되었습니다."
   );

   Bot.replyRoom
   (
    "[ " + nextPlayer + " ] 님은 새로운 단어를 입력해 주세요₩n₩n" +
    "< 남은 플레이어 >₩n" + Game.getPlayerListMessage()
   );

   AI.isAITurn() ? AI.getRandomReply() : null;
  }
 }
},

/* ---------------------------------------------------------------------------- */
/* ---------------------------------------------------------------------------- */

setNextTurn : function()
{
 gameTimerCount = 0;

 if (gameTurn == gamePlayerList.length)
 {
  gameRound ++;
  gameTurn = 1;
 }
 else
 {
  gameTurn += 1;
 }
},

setNextEvent : function(word)
{
 gameUsedWord.push(word);
 Game.setPlayerScore(1, word);
```

```
 Game.setNextTurn();
},

/* ---------------------------------------------------------------------------- */
/* ---------------------------------------------------------------------------- */

main : function(room, message, sender)
{
 let isGameWord = message.substr(0, 2) == "::",
  isCommand = message.charAt(0) == "/";

 (isCommand) ? Game.command(room, message, sender) : "";

 if (gamePower && isGameWord)
 {
  let word = message.substring(2).trim(),
   len = gamePlayerList.length, num = 0;

  for (num = 0 ; num < len ; num ++)
  {
   if ((sender == gamePlayerList[num]) && ((gameTurn - 1) == num))
   {
    if (DB.isSuitableWord(word))
    {
     gameLastChar = DB.getLastChar(word);

     Bot.replyRoom
     (
      "< " + DB.getWordMeanMessage(word) + " >₩n₩n" +
      "[ " + Game.getNowPlayer() + " ] 님이 ₩"" + word + "₩"단어를 입력하셨습니다.₩n" +
      "[ " + Game.getNextPlayer() + " ] 님은 ₩"" + DB.getLastCharMessage(gameLastChar) + "₩"(으)로 시작하는 단어
를 입력해 주세요"
     );

     PHP.postData("GameData", word);
     Game.setNextEvent(word);

     AI.isAITurn() ? AI.getReply(DB.getLastChar(word)) : null;
    }
   }
  }
 }
},

command : function(room, message, sender)
{
 let command = message.split(" ")[0],
  input = message.split(" ")[1],
  select = message.split(" ")[2],
  type = message.split(" ")[3],

  printMessage = function(type)
  {
   switch (type)
   {
```

```
 {
  case 1 : Bot.replyRoom("이미 생성된 방이 있습니다."); break;
  case 2 : Bot.reply("생성된 방이 없습니다."); break;
  case 3 : Bot.reply("끝말잇기 방이 생성된 채팅방에서만 입력이 가능합니다."); break;
  case 4 : Bot.replyRoom("이미 끝말잇기 게임이 진행중입니다."); break;
  case 5 : Bot.replyRoom("진행중인 게임이 없습니다."); break;
  case 6 : Bot.replyRoom("2명 이상이 참가해야 게임 시작이 가능합니다."); break;
  case 7 : Bot.replyRoom("방장만 입력이 가능합니다."); break;
  case 8 : Bot.replyRoom("중복 참여로 참가가 거부되었습니다."); break;
  case 9 : Bot.replyRoom("2명 이상이 참가해야 게임 시작이 가능합니다."); break;
  case 10 : Bot.replyRoom("생성된 AI가 없습니다."); break;
  case 11 : Bot.replyRoom("게임에 참가중인 상태가 아닙니다."); break;
 }
},

 roomManager = gamePlayerList[0],
 roomCheck = (roomName == room),
 managerCheck = (roomManager == sender),
 playerCheck = (gamePlayerList.indexOf(sender) != -1);

if (command == "/끝말잇기") {
if (input == "도움말")
{
 Bot.reply
 (
  "< 끝말잇기 도움말 >₩n₩n" +
  "● 게임시작 방법 : 생성 > 인원모집 (참가) > 시작₩n" +
  "● 단어입력 방법 : ₩"::단어₩"₩n" + ("₩u200b".repeat(500)) + "₩n" +
  "----------------------------------------------------------₩n" +
  "/끝말잇기 [생성 / 시작 / 참가 / 기권(자기차례일때만 기권하세요) / 나가기 / 종료]₩n" +
  "/끝말잇기 [데이터]₩n" +
  "----------------------------------------------------------₩n" +
  "/끝말잇기 AI 대결 [초보, 중수, 고수]₩n" +
  "/끝말잇기 AI 추가 [초보, 중수, 고수]₩n" +
  "/끝말잇기 AI 삭제₩n" +
  "----------------------------------------------------------₩n" +
  "/끝말잇기 한방단어 [켜기 / 끄기]₩n" +
  "----------------------------------------------------------₩n" +
  "/끝말잇기 검색 단어 [단어]₩n" +
  "/끝말잇기 검색 시작단어 [글자]"
 );
}
if (input == "데이터") { if (roomCreat) { if (roomCheck) { if (gamePower)
{
 let time = Math.floor((new Date().getTime() - gamePlayTime) / 1000),
  minute = Math.floor(time / 60), second = Math.floor(time % 60),
  len = gamePlayerList.length, i, j, text = [], temp = "";

 for (i = 0 ; i < len ; i ++)
 {
  text[i] = "○ " + gamePlayerList[i] + " : " +
   gamePlayerData[gamePlayerList[i]]['Score'] + "S : " +
   gamePlayerData[gamePlayerList[i]]['Life'] + "L";
 }
```

```
  for (i = 0 ; i < len - 1 ; i ++)
  {
   for (j = i + 1 ; j < len ; j ++)
   {
    if (Number(text[i].split(" : ")[1].split("S")[0]) < Number(text[j].split(" : ")[1].split("S")[0]))
    {
     temp = text[j];
     text[j] = text[i];
     text[i] = temp;
    }
   }
  }

  Bot.replyRoom
  (
   "< 현재 게임 상태 >₩n₩n" +
   "[ 시간 데이터 ]₩n" +
   "● 시작 시간 : " + gameStartTime + "₩n" +
   "● 플레이 시간 : " + ((!minute) ? second + "초" : minute + "분 " + second + "초") + "₩n₩n" +
   "[ 게임 데이터 ]₩n" +
   "● 플레이어 정보₩n" + text.join("₩n") + "₩n₩n" +
   "● 사용한 단어 : " + ((gameUsedWord.length) ? gameUsedWord.length + "개₩n○ " + gameUsedWord.join(" - ")
: "아직 사용한 단어가 없습니다.")
  );
 } else printMessage(5); } else printMessage(3); } else printMessage(2); }
 if (input == "생성") { if (!roomCreat)
 {
  Game.setPlayerAdd(sender);
  Game.setRoomCreat(room);
  Game.startRoomTimer();

  Bot.replyRoom
  (
   "[ " + sender + " ] 님이 끝말잇기 게임을 생성하였습니다.₩n₩n" +
   "게임 참가를 원하시면 ₩"/끝말잇기 참가₩"를 입력해 주세요."
  );
 } else printMessage(1); }
 else if (input == "참가") { if (roomCreat) { if (roomCheck) { if (!gamePower) { if (!playerCheck)
 {
  Game.setPlayerAdd(sender);

  Bot.replyRoom
  (
   "[ " + sender + " ] 님이 끝말잇기에 참가하셨습니다.₩n₩n" +
   "현재 참가자 : " + gamePlayerList.join(", ")
  );
 } else printMessage(8); } else printMessage(4); } else printMessage(3); } else printMessage(2); }
 else if (input == "시작" || input == "종료" || input == "재시작") { if (roomCreat) { if (roomCheck) { if
(managerCheck)
 {
  if (input == "시작") { if (!gamePower)
  {
   if (gamePlayerList.length >= 2)
   {
    Game.startRoomTimer();
```

```
    Game.stopRoomTimer();
    Bot.replyRoom("잠시 후 게임을 시작합니다!");

    Game.start();

    Bot.replyRoom
    (
     "게임을 시작합니다!₩n₩n" +
     "< 플레이어 목록 >₩n" +
     Game.getPlayerListMessage() + "₩n₩n" +
     "[ " + roomManager + " ] 님은 ₩"" + DB.getLastCharMessage(gameLastChar) + "₩"(으)로 시작하는 단어를 입력
해 주세요"
    );
   } else printMessage(9); } else printMessage(4);
  }
  else if (input == "종료" || input == "재시작") { if (gamePower)
  {
   if (input == "종료")
   {
    Bot.replyRoom("게임이 종료됩니다.");
    Game.off();
   }
   else if (input == "재시작")
   {
    Bot.replyRoom("현재 인원으로 게임을 재시작합니다.");
    Game.restart();
   }
  } else printMessage(5); } else printMessage(7); } else printMessage(3); } else printMessage(2); }
 }
 else if (input == "기권" || input == "나가기") { if (roomCreat) { if (roomCheck) { if(gamePower) { if(playerCheck)
 {
  if (input == "기권")
  {
   Bot.replyRoom("[ " + sender + " ] 님이 게임을 기권했습니다.");
   Game.setPlayerExit(sender);
  }
  else if (input == "나가기")
  {
   Object.keys(gamePlayerData)[Game.getPlayerPosition(sender)] = null;
   Bot.replyRoom("[ " + sender + " ] 님이 게임을 나갔습니다.");
   Game.setPlayerExit(sender);
  }
 } else printMessage(11); } else printMessage(5); } else printMessage(3); } else printMessage(2); }
 else if (input == "검색") { if (select == "단어" || select == "시작단어")
 {
  if (!DB.isLoaded())
  {
   Bot.reply("DB를 자동으로 등록한 후 검색을 시작합니다.");
   DB.load();
  }

  if (select == "단어")
  {
   let inputWord = type;
```

```
  if (DB.isWord(inputWord))
  {
   Bot.reply
    (
    "<" + inputWord + "의 사전 검색 결과 : " + DB.getWordMean(inputWord).length + "개>₩n₩n" +
     DB.getWordMeanListMessage(inputWord)
    );
  }
  else Bot.reply("사전에 등록되지 않은 단어입니다.");
 }
 else if (select == "시작단어")
 {
  let startWord = type;

  if (DB.isStartWord(startWord))
  {
   Bot.reply
   (
    "<" + startWord + "(으)로 시작하는 단어 " + DB.StartWord[startWord].length + "개>₩n₩n" +
    DB.StartWord[startWord].join(", ")
   );
  }
  else Bot.reply(startWord + "로 시작하는 단어는 사전에 없습니다.");
 }
}}
else if (input == "한방단어") { if (roomCreat) { if (roomCheck) { if (managerCheck) { if(!gamePower)
{
 if (select == "켜기")
 {
  Bot.replyRoom("한방단어 모드가 켜졌습니다.");
  modeOwnCom = true;
 }
 else if (select == "끄기")
 {
  Bot.replyRoom("한방단어 모드가 꺼졌습니다.");
  modeOwnCom = false;
 }
} else printMessage(4); } else printMessage(7); } else printMessage(3); } else printMessage(2); }
else if (input == "AI")
{
 if (select == "대결" || select == "도전") { if (!roomCreat) { if (!gamePower)
 {
  if (type == "초보" || type == "중수" || type == "고수")
  {
   Game.setPlayerAdd(sender);
   Game.setRoomCreat(room);

   aiCreat = true;
   aiPower = true;

   switch (type)
   {
    case "초보" : aiLevel = 1; aiName += "초보"; break;
    case "중수" : aiLevel = 2; aiName += "중급"; break;
    case "고수" : aiLevel = 3; aiName += "고수"; break;
```

```
     case "고수" : aiLevel = 3; aiName += "고수"; break;
    }

    Game.setPlayerAdd(aiName);

    Bot.replyRoom("잠시 후 게임을 시작합니다!");
    Game.start();

    Bot.replyRoom
    (
     "게임을 시작합니다!₩n₩n" +
     "< 플레이어 목록 >₩n" +
     Game.getPlayerListMessage() + "₩n₩n" +
     "[ " + sender + " ] 님은 ₩"" + DB.getLastCharMessage(gameLastChar) + "₩"(으)로 시작하는 단어를 입력해 주세
요"
    );
   }
  } else printMessage(1); } else printMessage(4); }
  if (select == "추가" || select == "삭제") { if (roomCreat) { if (roomCheck) { if (managerCheck) {if(!gamePower) {
  if (select == "추가")
  {
   /// let list = ["초보", "중수", "고수", "쉬움", "보통", "어려움"]
   if (type == "초보" || type == "중수" || type == "고수")
   {
    aiCreat = true;
    aiPower = true;

    switch (type)
    {
     case "초보" : aiLevel = 1; aiName += "초보"; break;
     case "중수" : aiLevel = 2; aiName += "중급"; break;
     case "고수" : aiLevel = 3; aiName += "고수"; break;
    }

    Game.setPlayerAdd(aiName);

    Bot.replyRoom
    (
      "[ " + aiName + " ] 를 게임을 추가했습니다.₩n₩n" +
      "현재 참가자 : " + gamePlayerList.join(", ")
    );
   }
  }
  else if (select == "삭제")
  {
   if (aiCreat)
   {
    aiName = "AI:";
    aiCreat = false;
    aiPower = false;
    Game.setPlayerDelete(aiName);
    Bot.replyRoom(aiName + " 가 삭제되었습니다.");
   }
   else printMessage(10);
  }
```

```
  } else printMessage(4); } else printMessage(7); } else printMessage(3); } else printMessage(2); }}}
 },
}

function response(room, message, sender, isGroupChat, replier, imageDB, packageName, threadId) { try
{
 Bot.reply = (chatting) => { replier.reply(chatting); };
 Bot.replyRoom = (chatting) => { (roomName) ? Api.replyRoom(roomName, chatting) : null; };

 Game.main(room, message, sender);
}
catch(e)
{
 Bot.reply
 (
  "Error Code : " + e.name + "₩n₩n" +
  "Content : " + e.message + "₩n₩n" +
  "Line : " + e.lineNumber
 );
}}

function onStartCompile()
{
 Game.stopGameTimer();
 Game.stopRoomTimer();
}
```


진워커님의 게시글 더보기 >

좋아요 **0** 댓글 **0** 공유 | 신고

댓글

댓글알림

Hibot

댓글을 남겨보세요

등록

글쓰기 답글 목록 ▲ TOP

'| 자유 게시판 |' 게시판 글

이 게시판 새글 구독하기

부계정 만들었는데 톡 하나만 보내주실수 있을까요?	문어다리는10개	2020.06.13.
혹시 제 계정 추가해서 메시지좀 보내주실 수 있을까요?	책사주세요	2020.06.13.
소스 오류?질문	진워커	2020.06.13.
이벤트리스너 호출 실패 [36]	Ad	2020.06.13.
이제 뭐 만들지 [13]	사로로	2020.06.13.

1 2 3 전체보기

이 카페 인기글

로코가 뭐길래 사람들이 환장하는걸까?

쿨칠

♡1 ⋯18

이카페는 팩트를 쓰면 날아가는군요.

tomohong

♡2 ⋯9

카톡봇 사람들은 꼭봐야됨

지민

♡4 ⋯8

카톡 정지에 관해서

리스트

♡0 ⋯15

계산기 소스

JSR

♡0 ⋯15

성빈님

린

♡0 ⋯1

간단 계산기 입니다

카카오톡이 이런걸 추천하다니

getSchoolFood

도미 doami2005

♡0 ⋯5

1 2 3 4 5