**聊天室**

王宇鹏 最后更新 2017-08-21

**功能介绍**

**服务器**

* 启动服务器 ServerMain
* 服务器命令

| **命令** | **说明** |
| --- | --- |
| 1 | 获取在线用户列表 |
| 2 | 获取聊天室列表 |
| 3 | 关闭服务器 |

**客户端**

* 启动客户端 ClientMain
* 客户端命令

| **命令** | **说明** |
| --- | --- |
| $login 用户名 密码 | 登录或注册 |
| $logout | 退出登录 |
| $quit | 退出客户端 |
| $create 房间名 房间简介 | 创建房间 |
| $enter 房间名 | 进入聊天室 |
| $exit | 退出房间，回到大厅 |
| $rooms | 聊天室列表 |
| $users( 房间名) | （聊天室里）在线用户列表 |
| $xxx | 在聊天室中直接发言 |
| $@昵称 xxx | 对另一人密语 |
| $hongbao 金额,个数(,拼手气) | 发（拼手气）红包 |
| $qiang 红包编号 | 抢红包 |

**代码结构**

* src
  + client：客户端
    - ChatClient：客户端类，包括连接退出、命令消息传递方法、命令消息接收线程等
    - ClientMain：客户端入口，处理接收命令输入并传递
  + model：模型实体类
    - ChatRoom：聊天室类
    - Command：命令枚举
    - Hongbao：红包类
    - Message：消息类
    - MsgType：消息类型枚举
    - User：用户类
  + server：服务器
    - ChatServer：服务器类，包括连接退出等方法、命令消息接收处理线程
    - ServerMain：服务器入口，接收日志信息
  + utils：工具类
    - FileHelper：文件读写
    - ReadWriteInfo：用户、聊天室、聊天记录读写
    - SerializeHelper：序列化、反序列化
    - StringHelper：字符串处理

**代码介绍**

**client 客户端**

**ChatClient**

主要成员变量：

* Selector selector
* SocketChannel socketChannel
* boolean isConnected：当前状态是否连接
* boolean isLogin：当前用户是否登录
* String username;：当前用户名

主要方法：

* void connect(String host, int port)

连接服务器：传入host和port，开启Selector，初始化InetSocketAddress，打开SocketChannel，设置为非阻塞模式，注册Selector，开启客户端线程

* void login(String username, String passwd)

登录或注册：传入用户名和密码，将消息发送给服务器

* void createChatRoom(String roomName, String roomInfo)

创建聊天室：传入聊天室名字和简介，将消息发送给服务器

* void enterChatRoom(String roomName)

进入聊天室：传入聊天室名字，将消息发送给服务器

* void exitChatRoom()

退出聊天室：将消息发送给服务器

* void sendMsgToRoom(String msg)

发送消息到聊天室：传入聊天内容，将消息发送给服务器

* void sendMsgToUser(String toName, String msg)

发送消息给某用户：传入对方用户名和聊天内容，将消息发送给服务器

* void sendHongbao(String totalMoney, String count, String isRandom)

发红包：传入红包金额、个数和是否是拼手气红包，将消息发送给服务器

* void qiangHongbao(String hongbaoId)

抢红包：传入红包编号，将消息发送给服务器

ClientThread线程：

Selector.select() > 0时循环，SelectionKey可读时，建立SocketChannel，将ByteBuffer的数据写到ByteArrayOutputStream，并转换为byte数组，通过反序列化读到定义的Message类的实例，即从服务器传递来的消息，通过判断消息的类型来做相应的处理，如读到登录、创建聊天室、加入聊天室、发消息等的结果，用于显示给客户端

**ClientMain**

客户端主函数，开启客户端线程，读取输入，通过判断输入的符合格式的命令，调用各自对应的方法，将消息发送给服务器

**model 模型实体类**

**ChatRoom**

主要成员变量：

* String roomName：聊天室名称
* String roomInfo：聊天室简介
* Set<String> users = Collections.synchronizedSet(new HashSet<>());：聊天室用户集合
* Set<String> hongbaos = Collections.synchronizedSet(new HashSet<>());：聊天室红包集合

主要方法：

* Set<String> getUsers()

获取聊天室用户：返回该聊天室的用户集合

* void addUser(String username)

用户进入聊天室：传入用户名，添加到该聊天室的用户集合

* boolean hasUser(String username)

用户是否在聊天室：传入用户名，判断用户是否在该聊天室的用户集合

* void removeUser(String username)

用户离开聊天室：传入用户名，从该聊天室的用户集合中移除该用户

* void addHongbao(String hongbaoId)

加入红包：传入红包编号，添加到该聊天室的红包集合

* boolean hasHongbao(String hongbaoId)

聊天室是否存在某红包：传入红包编号，判断红包是否在该聊天室的红包集合

**Command**

枚举类，包括登录、退出、创建聊天室、进入聊天室、离开聊天室、查询聊天室列表、聊天室成员、发送消息到聊天室、私聊、发红包、抢红包等，用于消息传递

**Hongbao**

主要成员变量：

* static AtomicInteger id：红包编号
* boolean isRandom：是否是拼手气红包
* int totalMoney：红包总金额
* int count：红包个数
* Map<String, Integer> hbUsrMap = Collections.synchronizedMap(new HashMap<>());：该红包的用户-抢到金额的键值对

主要方法：

* int getAverageMoney()

获取平均金额：返回每个人抢到普通红包的金额

* int updateLeftCount()

更新剩余红包个数：返回剩余没有被抢的红包个数

* synchronized void qiang(String nameQiang)

抢红包：传入抢红包的用户名，如果剩余0个红包，就返回；如果剩余1个红包，就把用户和剩余金额加入Map<用户名，抢到的金额>中；其余情况，判断是否是拼手气红包，是的话，抢到金额为符合条件的随机数，否则为平均金额，更新剩余金额，把用户和抢到金额加到Map中。

* String getInfoOrMax()

抢红包的信息：返回字符串，内容为抢到红包的用户名和金额数，按金额数从高到低排序，如果是拼手气红包，则标注手气最佳的用户名

**Message**

主要成员变量：

* Map<MsgType, String> fields：消息类型-消息内容的键值对
* Command command：命令枚举

主要方法：

* byte[] toBytes()

序列化消息：返回消息序列化后的byte数组

* void sendRawMessage(SocketChannel sc, Message message)

发送消息：传入SocketChannel和Message，向SocketChannel写入包装后的Message

**MsgType**

枚举类，包括用户名、密码、聊天室名称、聊天室简介、私聊用户名、聊天室用户、聊天内容、红包编号、红包总金额、红包个数、响应状态等，用于消息传递

**User**

主要成员变量：

* String username：用户名
* String password：密码
* SocketChannel socketChannel

**server 服务器**

**ChatServer**

主要成员变量：

* Selector selector
* Map<String, ChatRoom> rooms = Collections.synchronizedMap(new HashMap<>());：保存所有聊天室的Map
* Map<String, User> users = Collections.synchronizedMap(new HashMap<>());：保存所有在线用户的Map
* Map<String, Hongbao> hongbaos = Collections.synchronizedMap(new HashMap<>());：红包Map
* Map<String, String> userFile = Collections.synchronizedMap(new HashMap<>());:读取本地保存用户
* Map<String, String> roomFile = Collections.synchronizedMap(new HashMap<>());:读取本地保存聊天室
* String host
* int port

主要方法：

* void closeClient(SelectionKey sk, SocketChannel sc)

关闭客户端连接：传入SelectionKey和SocketChannel，遍历在线用户Map，移除相应的用户，以及聊天室用户集合里的用户

线程：

开启Selector和ServerSocketChannel，初始化InetSocketAddress，绑定端口，设置为非阻塞模式，注册Selector。Selector.select() > 0时循环，SelectionKey可读时，建立SocketChannel，将ByteBuffer的数据写到ByteArrayOutputStream，并转换为byte数组，通过反序列化读到定义的Message类的实例，即从客户端传递来的消息，通过判断消息的类型来做相应的处理，如登录、创建聊天室、加入聊天室、发消息、发红包、抢红包等，处理结束后，分别将结果信息传递给客户端

**ServerMain**

服务器主函数，开启服务器线程，接收日志信息

**utils 工具类**

**FileHelper**

主要方法：

* ArrayList<String> readByLinesList(String path)

按行读取文件到列表：传入文件地址，读取文件的每一行，把每一行的内容加入到列表中，返回列表

* boolean saveAs(String content, String path, boolean append)

写到文件：传入内容、文件路径、是否追加写入，返回写入是否成功

**ReadWriteInfo**

主要方法：

* Map<String, String> readUserInfoFromFile(String path)

从文件中读取用户信息：传入文件地址，按行读取文件，将用户名和密码传入Map中，返回Map

* void writeUserInfoToFile(String path, String name, String passwd)

把用户信息写到文件中：传入文件地址、用户名和密码，按行写到文件

* Map<String, String> readRoomInfoFromFile(String path)

从文件中读取聊天室信息：传入文件地址，按行读取文件，将聊天室名称和简介传入Map中，返回Map

* void writeRoomInfoToFile(String path, String name, String info)

把聊天室信息写到文件中：传入文件地址、聊天室名称和简介，按行写到文件

* String readMsgInfoFromFile(String path, String roomName)

从文件中读取聊天信息：传入文件地址和聊天室名称，按行读取文件，返回指定聊天室的聊天记录

* void writeMsgInfoToFile(String path, String roomName, String msg)

把聊天信息写到文件中：传入文件地址、聊天室名称和聊天记录，按行写到文件

**SerializeHelper**

主要方法：

* byte[] serialize(Object object)

序列化：传入对象，序列化为byte数组

* Object deSerialize(byte[] bytes)

反序列化：传入byte数组，反序列化为对象

**StringHelper**

主要方法：

* String moneyMulti100(String money)

金额单位转化为分：传入abc.de格式的金额，返回abcde的整数金额（乘以100）

* String moneyDivideBy100(int money)

金额单位转化为元：传入整数金额abcde，返回abc.de格式的金额（除以100）