目录

[**一、** **项目选题** 2](#_Toc106824764)

[**程序功能：** 2](#_Toc106824765)

[**二、** **开发环境和工具** 2](#_Toc106824766)

[**三、** **项目使用说明手册** 2](#_Toc106824767)

[**四、** **系统结构** 2](#_Toc106824768)

[**1.** **项目目录结构** 2](#_Toc106824769)

[**2.** **系统执行过程** 3](#_Toc106824770)

[**五、** **系统内核心实现** 3](#_Toc106824771)

[**1.** **显示聊天页面** 3](#_Toc106824772)

[**2.** **发送信息给服务器** 4](#_Toc106824773)

[**3.** **服务器端转发数据** 5](#_Toc106824774)

[**六、** **项目运行截图** 5](#_Toc106824775)

[6](#_Toc106824776)

1. **项目选题**

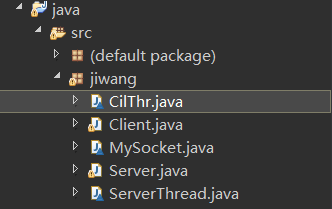
基于Socket套接字实现的多人实时聊天程序

**程序功能：**

1. 为每个请求的用户机分配一个端口号，记录ip及socket
2. 将用户发送的信息转发给聊天室内其他端口，并显示
3. **开发环境和工具**

|  |  |
| --- | --- |
| **系统环境** | Windows10 1903 |
| **开发工具** | Eclipse |
| **jdk版本** | 1.8.0\_331 |

1. **项目使用说明手册**
2. 运行客户端代码
3. 输入自己的用户名并确认
4. 在聊天框输入想要发送的信息
5. 点击发送即可将消息发送给其他用户
6. **系统结构**
7. **项目目录结构**

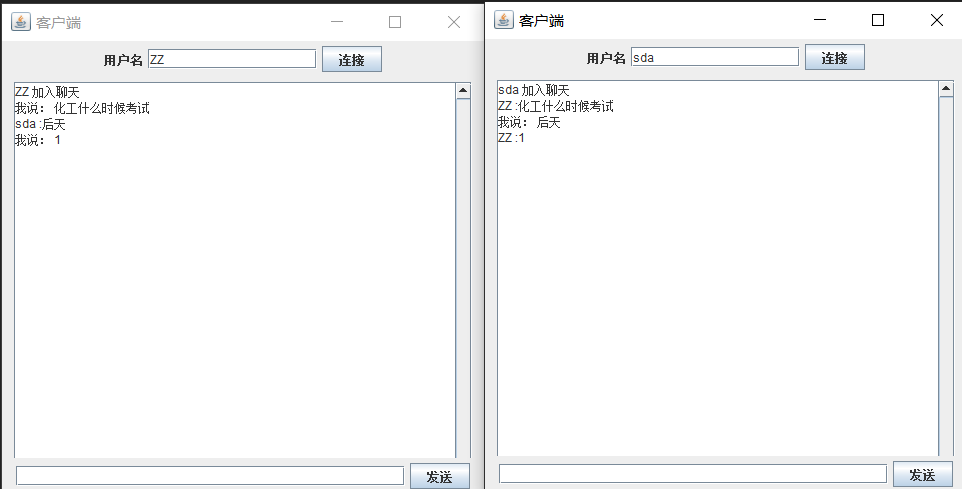


Client为客户端程序，创建一个Socket，并向Server请求建立链接，cliThr用于客户向套接字发送数据。Server为服务器程序，监听端口号为9000，保存每个链接的套接字。MySocket用于把每一台客户端的socket 和用户名联系起来；ServerThread为每一个链接到套接字的客户分配一个线程来收集发送的数据并转发

1. **系统执行过程**

当Server服务器启动时，首先会创建一个服务器端口，端口号为9000，然后持续监听每一次客户段连接，用accept（）接受，每响应一个客户端就产生一个myscokcet，把用户的名字和socket放到MySocket里再添加到集合里，这样就可以保存每一个客户端的名字和socket，启动线程 把当前客户端的socket和加入到服务器的所有socket传到线程通过构造器，并开启线程run方法：接受客户信息并转发给其他客户端。

在启动Client客户端后，就会连接到9000端口，显示聊天页面，在输入用户名，传给服务器响应过后即可使用文本框输入消息，给其他用户发现。



1. **系统内核心实现**
2. **显示聊天页面**

jf=new JFrame("客户端");

jLabel=new JLabel("用户名");

tf1=new JTextField(15);

tf2=new JTextField(35);

start=new JButton("连接");

send=new JButton("发送");

ta=new JTextArea(40,40);

init();//调用方法实现界面

}

public void init(){

JPanel jp1=new JPanel();

JPanel jp2=new JPanel();

JPanel jp3=new JPanel();

ta.setEditable(false);//让文本框不能输入，只是显示出来

send.setEnabled(false);//设置发送按钮，不能按，因为还没有连接上服务器

jp1.add(jLabel);

jp1.add(tf1);

jp1.add(start);

jp2.add(ta);

jp2.add(new JScrollPane(ta,JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS,JScrollPane.HORIZONTAL\_SCROLLBAR\_ALWAYS));//设置文本区域的滚动条

jp3.add(tf2);

jp3.add(send);

jf.add(jp1, BorderLayout.NORTH);//设置布局

jf.add(jp2,BorderLayout.CENTER);

jf.add(jp3,BorderLayout.SOUTH);

jf.setVisible(true);

jf.setLocationRelativeTo(null);

jf.setSize(500,500);

jf.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

handler();//添加按钮监听

new CilThr(socket,ta).start();

1. **发送信息给服务器**

class CilThr extends Thread{

Socket socket;

JTextArea ta;

public CilThr(Socket socket,JTextArea ta){//获取客户端的socket和文本区域

this.socket=socket;

this.ta=ta;

}

@Override

public void run() {

while (true){//一直在接收中

try {

InputStream inputStream=socket.getInputStream();//字节输入流

DataInputStream din=new DataInputStream(inputStream);//数据输入流

String text=din.readUTF();//获取输入的数据

ta.append(text);//添加到文本区域

} catch (IOException e) {

e.printStackTrace();

}

1. **服务器端转发数据**

public void run() {

while(true){//接收一个用户的多次发送信息

try {

InputStream inputStream=socket.getInputStream();//字节输入流

DataInputStream din=new DataInputStream(inputStream);//数据输入流

String text=din.readUTF();//获取输入的信息

String user="";//设置当前用户名为空，以防下一次赋值

for(int i=0;i<sockets.size();i++){//遍历sockets集合

if(socket==sockets.get(i).getSocket()){//判断当前发放信息的用户

user=sockets.get(i).getUser();//获取当前发放信息的用户名

}

}

for(int i=0;i<sockets.size();i++){

Socket sc=sockets.get(i).getSocket();//获取每个用户的socket

if(sc!=socket){//判断 不需要自己发信息给自己

OutputStream outputStream=sc.getOutputStream();//字节输出流

DataOutputStream dout=new DataOutputStream(outputStream);//数据输出流

dout.writeUTF(user+":"+text+"\n");//把当前客户端的名字和信息发送给除自己以外的其他客户端

}

}

} catch (IOException e) {

e.printStackTrace();

}

1. **项目运行截图**

