以下是实现聊天功能的代码

public class ClientFrame extends Frame {

 private TextField textFieldContent = new TextField();

 private TextArea textAreaContent = new TextArea();

 private Socket socket = null;

 private OutputStream out = null;

 private DataOutputStream dos = null;

 private InputStream in = null;

 private DataInputStream dis = null;

 private boolean flag = false;

 /\*\*

  \*

  \*

  \* @param args

  \*/

 public static void main(String[] args) {

  new ClientFrame().init();

 }

 /\*\*

  \*

  \*/

 private void init() {

  this.setSize(300, 300);

  setLocation(250, 150);

  setVisible(true);

  setTitle("WeChatRoom");

  // 添加控件

  this.add(textAreaContent);

  this.add(textFieldContent, BorderLayout.SOUTH);

  textAreaContent.setFocusable(false);

  pack();

  // 关闭事件

  addWindowListener(new WindowAdapter() {

   public void windowClosing(WindowEvent e) {

    System.out.println("用户试图关闭窗口");

    disconnect();

    System.exit(0);

   }

  });

  // textFieldContent添加回车事件

  textFieldContent.addActionListener(new ActionListener() {

   public void actionPerformed(ActionEvent e) {

    onClickEnter();

   }

  });

  // 建立连接

  connect();

  new Thread(new ReciveMessage()).start();

 }

 private class ReciveMessage implements Runnable {

  @Override

  public void run() {

   flag = true;

   try {

    while (flag) {

     String message = dis.readUTF();

     textAreaContent.append(message + "\n");

    }

   } catch (EOFException e) {

    flag = false;

    System.out.println("客户端已关闭");

    // e.printStackTrace();

   } catch (SocketException e) {

    flag = false;

    System.out.println("客户端已关闭");

    // e.printStackTrace();

   } catch (IOException e) {

    flag = false;

    System.out.println("接受消息失败");

    e.printStackTrace();

   }

  }

 }

 /\*\*

  \* 功能：当点击回车时出发的事件

  \*/

 private void onClickEnter() {

  String message = textFieldContent.getText().trim();

  if (message != null && !message.equals("")) {

   String time = new SimpleDateFormat("h:m:s").format(new Date());

   textAreaContent.append(time + "\n" + message + "\n");

   textFieldContent.setText("");

   sendMessageToServer(message);

  }

 }

 /\*\*

  \* 功能：给服务器发送消息

  \*

  \* @param message

  \*/

 private void sendMessageToServer(String message) {

  try {

   dos.writeUTF(message);

   dos.flush();

  } catch (IOException e) {

   System.out.println("发送消息失败");

   e.printStackTrace();

  }

 }

 /\*\*

  \* 功能：申请socket链接

  \*/

 private void connect() {

  try {

   socket = new Socket("localhost", 8888);

   out = socket.getOutputStream();

   dos = new DataOutputStream(out);

   in = socket.getInputStream();

   dis = new DataInputStream(in);

  } catch (UnknownHostException e) {

   System.out.println("申请链接失败");

   e.printStackTrace();

  } catch (IOException e) {

   System.out.println("申请链接失败");

   e.printStackTrace();

  }

 }

 /\*\*

  \* 功能：关闭流和链接

  \*/

 private void disconnect() {

  flag = false;

  if (dos != null) {

   try {

    dos.close();

   } catch (IOException e) {

    System.out.println("dos关闭失败");

    e.printStackTrace();

   }

  }

  if (out != null) {

   try {

    out.close();

   } catch (IOException e) {

    System.out.println("dos关闭失败");

    e.printStackTrace();

   }

  }

  if (socket != null) {

   try {

    socket.close();

   } catch (IOException e) {

    System.out.println("socket关闭失败");

    e.printStackTrace();

   }

   ;

  }

 }

}

以下是添加好友的部分代码

public class AddFriend {

public static final String ALLOW="Y";

public static final String DISALLOW="N";

private String friend\_1;

private String friend\_2;

private String f1\_allow;

private String f2\_allow;

private String currentUser;

private String friendName;

}

@RequestMapping("/addFriend")

public void addFriend(@RequestBody AddFriend friend,HttpServletResponse response) throws Exception{

friendService.addFriend(friend);

String hintMessage="请求已发出！等待其接受！！";

JSONObject message = new JSONObject();

message.put("hintMessage", hintMessage);

System.out.println(message.toJSONString());

response.setContentType("text/html;charset=UTF-8");

response.getWriter().println(message.toJSONString());

response.getWriter().close();

}

//同意“好友请求”

@RequestMapping("/agreeReq")

public void agreeReq(@RequestBody AddFriend addFriend,HttpServletResponse response) throws Exception{

friendService.friendReqResp(addFriend);

Friend friend = new Friend();

friend.setFriend\_1(addFriend.getFriend\_1());

friend.setFriend\_2(addFriend.getFriend\_2());

friendService.allowFriendReq(friend);

String hintMessage="相见恨晚，快去聊天吧！！";

JSONObject message = new JSONObject();

message.put("hintMessage", hintMessage);

System.out.println(message.toJSONString());

response.setContentType("text/html;charset=UTF-8");

response.getWriter().println(message.toJSONString());

response.getWriter().close();

}

//拒绝“好友请求”

@RequestMapping("/rejectReq")

public void rejectReq(@RequestBody AddFriend addFriend,HttpServletResponse response) throws Exception{

friendService.friendReqResp(addFriend);

String hintMessage="铁石心肠，成大事者也！！";

JSONObject message = new JSONObject();

message.put("hintMessage", hintMessage);

System.out.println(message.toJSONString());

response.setContentType("text/html;charset=UTF-8");

response.getWriter().println(message.toJSONString());

response.getWriter().close();

}