浙江大学计算机学院

Java 程序设计课程报告

2023—2024 学年 冬 学期

题目	GUI 与网络编程——miniWeChat
学号	3210102037
学生姓名	徐铭
所在专业	计算机科学与技术
所在班级	计科 2101

目　录

1 引言

本次开发的是一个简化版微信，同时使用 GUI 编程，这是一个综合性的题目，可以对 Java 语言中的各项功能有更好的理解和使用，通过具体的程序来加深对 Java 语言的掌握，提高自己的编程水平，为以后的工作打下一定的基础。

1．1 设计目的

微信是我们都在用的一个社交软件，其主要功能有很多，但是我们的日常生活其实并不需要使用这么多功能，我编写的 miniWeChat 就实现了微信最本质的功能，具体功能如下：

(1) 客户端：

- 用户输入用户名登录 miniWeChat；
- 登陆后查看在线的好友；
- 与在线好友开启聊天，且两人的客户端实时同步消息；
- 对方断开连接时会弹出消息并关闭聊天窗口；

(2) 选服务端：

- 启动后接受客户端连接；
- 管理已连接的所有客户端；
- 负责转发不同客户端之间的消息；

1．2 设计说明

本程序采用 Java 程序设计语言，在 IDEA 平台下编辑、编译与调试。具体程序由本人独立完成。

2 总体设计

2．1 功能模块设计

本程序需实现的主要功能有

(1) 客户端：

- 用户输入用户名登录 miniWeChat；
- 登陆后查看在线的好友；
- 与在线好友开启聊天，且两人的客户端实时同步消息；
- 对方断开连接时会弹出消息并关闭聊天窗口；

(3) 选服务端：

- 启动后接受客户端连接；
- 管理已连接的所有客户端；
- 负责转发不同客户端之间的消息；

程序的总体功能如图 1 所示：

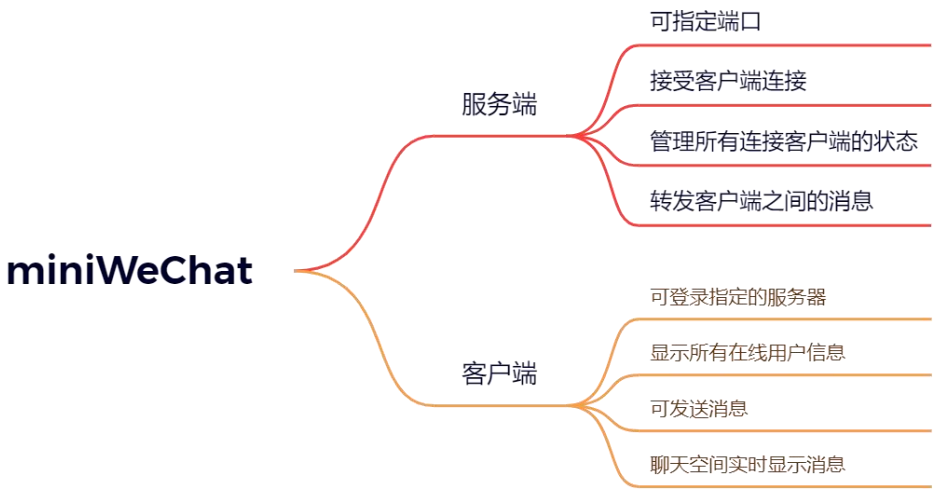

图 1 总体功能图

（图表文字行距均为单倍行距）

2. 2 流程图设计

程序总体流程如图 2 所示：

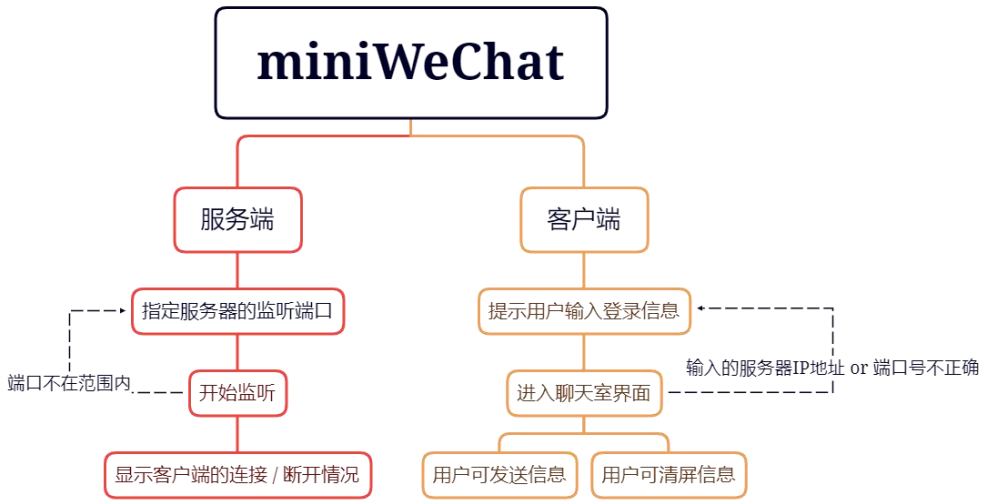

图 2 总体流程图

3　详细设计

3．1　总体描述

系统的整体框架如图 3 所示，主要分为 Client，Client_GUI， Server，Server_GUI 和 Message 这 5 个类，各个类的具体功能和实现将在接下来的部分一一介绍。

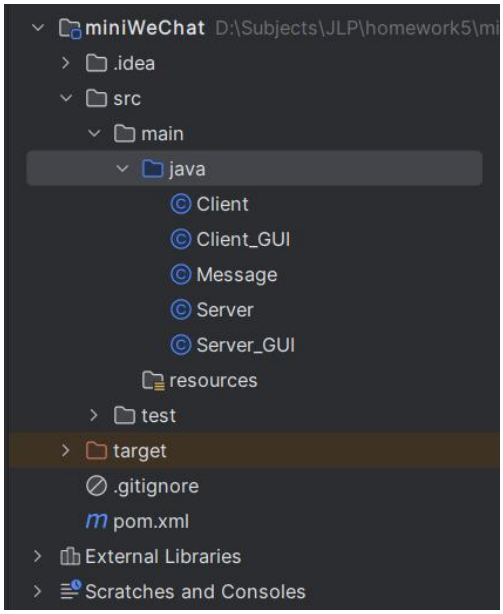

图 3 系统整体框架

3．2　Message 的设计

Message 类主要用于封装在服务器和客户端之间传递的消息。通过 type 来区分消息类型，content 存储实际的消息内容。这种设计使得消息的传递更加灵活，可以根据 type 的不同执行不同的处理逻辑。由于实现了 Serializable 接口，该类的对象可以在网络上传输：

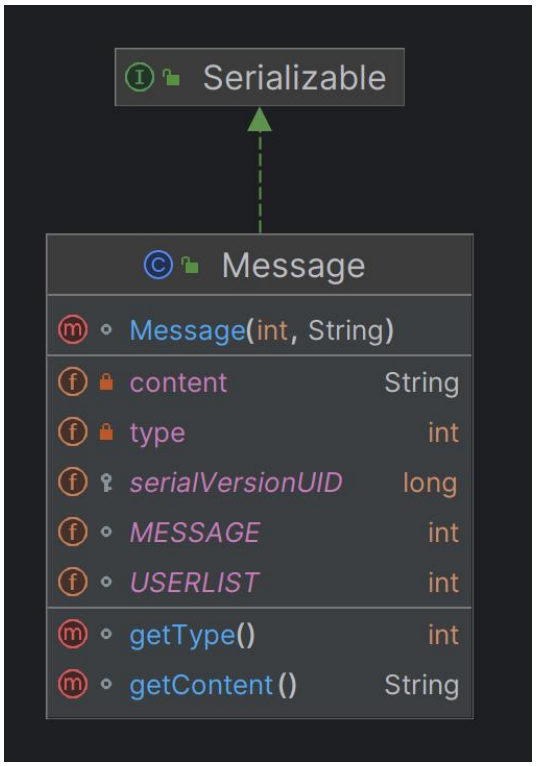

图 4 Message 类的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

（1）变量：

serialVersionUID: 用于版本控制的序列化 ID，保证在不同版本之间兼容性。

type: 表示消息的类型，是一个整数。常量 MESSAGE 表示普通消息，USERLIST 表示用户列表。

content: 表示消息的实际内容，是一个字符串。

（2）构造函数：

① Message(int type, String content)

构造函数用于创建 Message 对象，需要传入消息类型和消息内容作为参数。

② getType():

int getType()

用于获取消息的类型，返回整数。

③ getContent():

String getContent()

用于获取消息的内容，返回字符串。

3．3　Client_GUI 的设计

Client_GUI 类充当了客户端图形用户界面的管理者，通过界面与用户进行交互，同时通过 Client 处理与服务器的通信，实现了基本的聊天功能。下面标明 Client_GUI 类的主要成员变量、方法的 UML 图如图 3-5 所示：

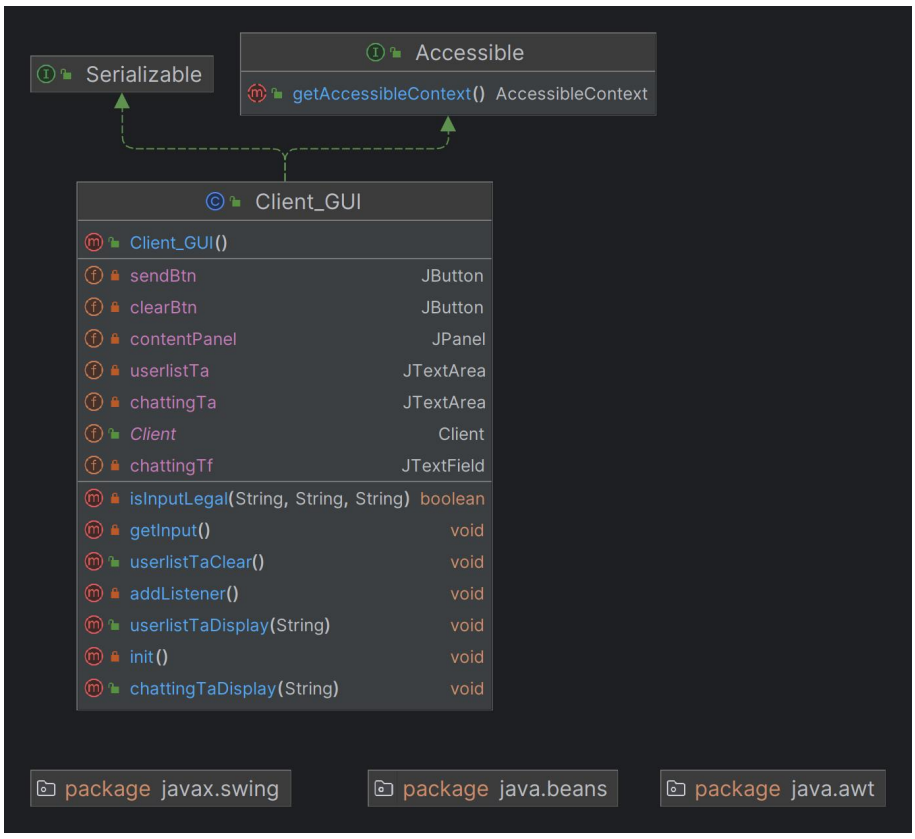

图 5　Client_GUI 类的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

Client_GUI 类主要是实现客户端的图形用户界面，以下是代码主要功能的总结：

（1）界面初始化：

创建发送按钮(sendBtn)和清屏按钮(clearBtn)。

创建聊天文本框(chattingTf)，用于输入聊天消息。

创建聊天文本区域(chattingTa)，用于显示聊天消息。

创建用户列表文本区域(userlistTa)，用于显示在线用户列表。

创建内容面板(contentPanel)，包含聊天区域和用户列表区域。

（2）用户输入获取：

通过 getInput 方法弹出对话框，获取用户输入的服务器 IP、端口号和用户名并检查用户输入的合法性。

（3）事件监听：

通过 addListener 方法添加事件监听器，监听发送按钮和清屏按钮的动作。发送按钮用于发送聊天消息，清屏按钮用于清空聊天区域。

（4）界面显示：

提供方法(chattingTaDisplay, userlistTaDisplay, userlistTaClear)用于在界面上显示聊天消息和用户列表信息。包含一个 Client 类的实例，用于处理与服务器的通信。

3．4 Client 设计

Client 是处理客户端的主要程序，也是启动客户端的唯一入口，它同时负责管理客户端界面的创建和与服务器之间的 socket 通信，下面是 Client 类的 UML 图：

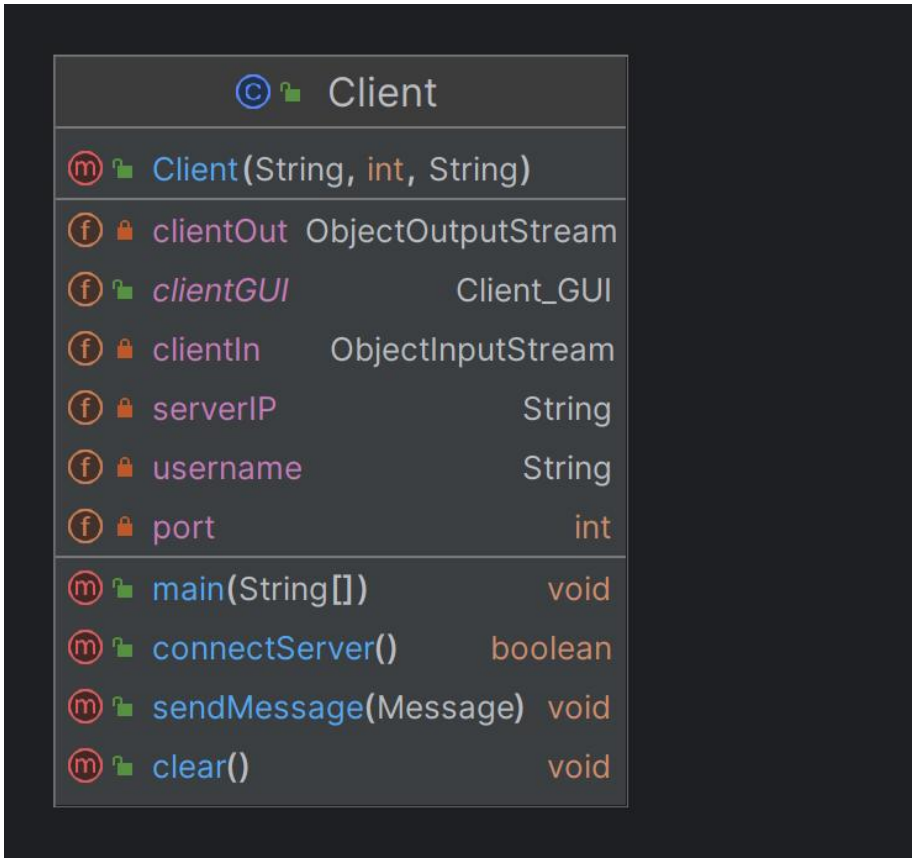

图 6　Client 类的 UML 图

（1）成员变量:

ObjectInputStream clientIn: 客户端 Socket 输入流，用于接收服务器发来的消息。

ObjectOutputStream clientOut: 客户端 Socket 输出流，向服务器发送消息。

String serverIP: 服务器的 IP 地址。

int port: 服务器的端口。

String username: 客户端的用户名。

Client_GUI clientGUI: Client_GUI 类的实例，用于处理客户端的图形用户界面。

（2）方法：

④ main(String[] args): 主方法，启动客户端，创建 Client_GUI 实例。

⑤ Client(String serverIP, int port, String username): 构造函数，初始化服务器 IP、端口和用户名。

⑥ connectServer(): 连接到服务器的方法。创建客户端 Socket，建立与服务器的连接，初始化输入输出流，向服务器发送用户名，创建并启动一个线程用于监听来自服务器的消息。

⑦ clear(): 清除客户端界面的方法，用于清空服务器 IP 和端口。

⑧ sendMessage(Message message): 发送消息的方法，将消息对象通过输出流发送给服务器。

⑨ 内部类 InteractWithServer:继承自 Thread 类,用于在后台监听服务器发送的消息。

1) run(): 线程运行方法，持续监听服务器发来的消息。接收消息，根据消息类型显示在客户端图形界面的聊天文本区域或用户列表文本区域。若与服务器的连接断开，弹出提示窗口，并隐藏客户端图形界面。

注意:

Message 类的实例用于封装消息，其中包含消息的类型和内容。客户端在连接到服务器后，会创建一个用于监听的线程 InteractWithServer，该线程负责接收和处理来自服务器的消息。

3．5　Server_GUI 设计

Server_GUI 是绘制服务器界面的主要程序，接受用户的输入来设置端口，下面是 Server_GUI 类的 UML 图:

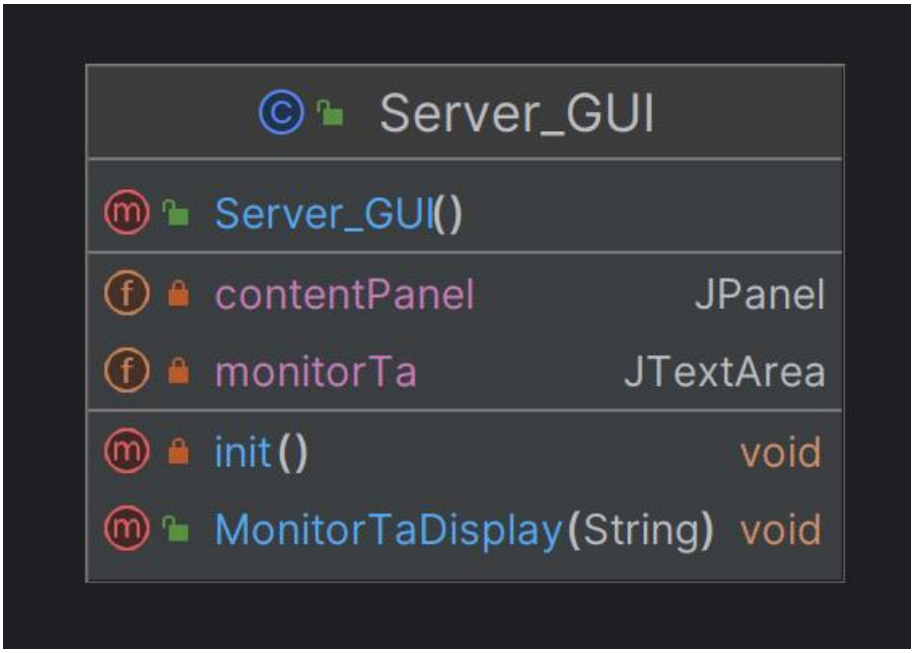

图 7　Server_GUI 类的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

（1）成员变量：

JPanel contentPanel: 用于容纳界面组件的面板。

JTextArea monitorTa: 用于显示服务器监控信息的文本区域。

（2）方法：

① Server_GUI(): 构造函数，用于初始化界面。

② init(): 初始化方法，包含以下步骤：

1) 获取监听端口，使用 JOptionPane 弹出对话框，让用户输入准备监听的端口号。通过循环确保用户输入的端口号是合法的整数值（1-65535）；

2) 窗口设置：设置窗口的标题为"服务器界面"，大小为 750x600，启用关闭按钮，将窗口放置在屏幕中央；

3) 创建监控文本区域:创建一个大小为 500x80 的文本区域 monitorTa，设置背景颜色为淡黄色，并禁止编辑。

4) 创建滚动窗格： 使用 JScrollPane 包装监控文本区域，以支持文本内容超出可视区域时的滚动。

5) 添加组件： 将滚动窗格添加到 contentPanel 面板中。

6) 设置位置： 将滚动窗格放置在面板的(0, 0)位置；

7) 面板透明化： 将 contentPanel 设置为透明，以便看到背景。

8) 显示窗口： 通过 getContentPane().add(contentPanel) 将面板添加到窗口中，使其可见。

9) 创建并启动服务器实例： 创建 Server 类的实例 sersoc，并通过 sersoc.listen() 启动服务器监听。

① MonitorTaDisplay(String msg): 用于在监控文本区域显示消息。

1) 追加消息： 将传入的消息 msg 追加到 monitorTa 文本区域中。

2) 滚动到底部： 将滚动条拉到 monitorTa 文本区域底部，以便查看最新的监控信息。

注意：

Server_GUI 类主要负责创建并显示服务器监控界面，提供一个可视化的窗口用于展示服务器的运行状态和消息。用户通过输入端口号启动服务器监听，相关信息通过 monitorTa 显示。通过 JOptionPane 实现简单的用户输入和信息提示功能。服务器实例 sersoc 通过 Server 类创建,并在 init 方法中启动监听。

3.6 Client 设计

Server 是处理服务端的主要程序，也是启动服务端的唯一入口，它同时负责管理服务器界面的创建和与客户端之间的 socket 通信，下面是 Server 类的 UML 图:

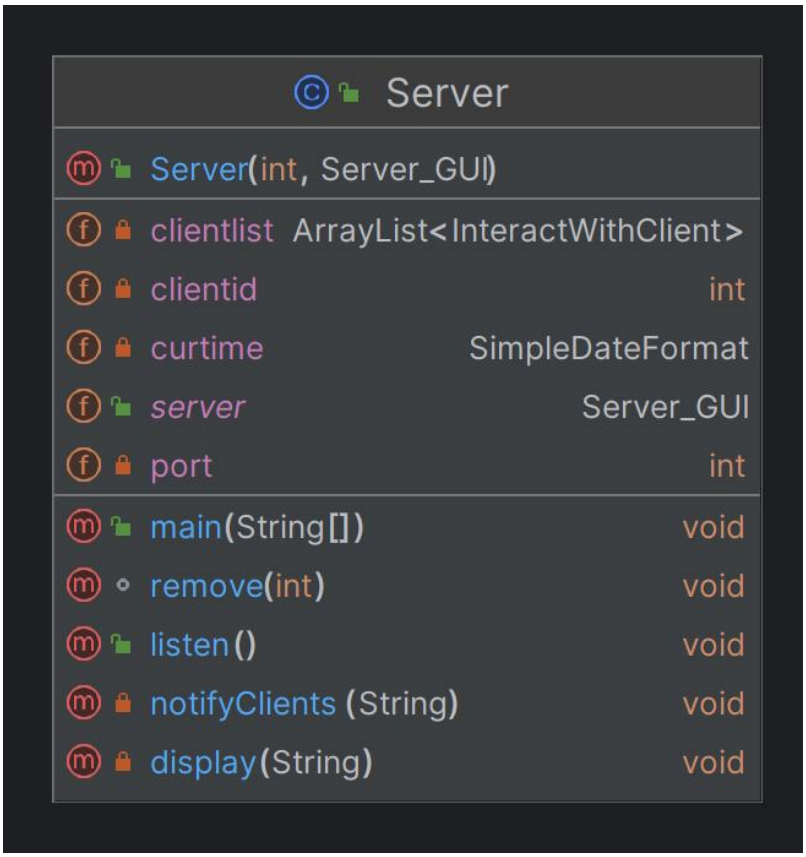

图 8　Server 类的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

（1）成员变量：

int port: 监听的端口号，服务器启动时指定。

ArrayList<InteractWithClient> clientlist: 保存所有已连接客户端的 ArrayList。

int clientid: 每个客户端的唯一 ID，用于标识不同的客户端。

SimpleDateFormat curtime: 日期时间格式化对象，用于记录事件发生时间，以及在监控区域显示的时间格式。

Server_GUI server: Server_GUI 类的实例，用于在界面上显示服务器监控信息。

（2）方法：

① Server(int port, Server_GUI server): 构造函数，创建 Server 实例时初始化相关参数。

② listen(): 主要函数，负责监听并等待客户端连接。通过创建 ServerSocket 对象，不断接受客户端的连接请求，并为每个连接创建一个新的 InteractWithClient 线程处理。

③ notifyClients(String message): 通知所有已连接客户端的方法。根据传入的消息内容，构建 Message 对象，然后通过所有客户端的输出流向它们发送消息。

④ remove(int id): 从客户端列表中移除指定 ID 的客户端。在客户端断开连接时调用。

⑤ display(String msg): 在 Server_GUI 上显示消息的方法。将传入的消息格式化，并通过 Server_GUI 的 MonitorTaDisplay 方法在监控文本区域显示。

⑥ InteractWithClient: 内部类，用于处理与单个客户端的交互。包含以下成员：

int id: 客户端的唯一 ID。

Socket socket: 与客户端通信的套接字。

ObjectInputStream socketInput: 从客户端接收消息的输入流。

ObjectOutputStream socketOutput: 向客户端发送消息的输出流。

String userName: 连接的客户端的用户名。

String linktime: 客户端连接的时间。

Message clientmessage: 从客户端接收的消息。

run(): 作为线程运行的方法。在循环中接收客户端消息，然后根据消息类型做相应的处理。如果客户端关闭，则从客户端列表中移除，并通知其他客户端。

sendMessage(Message msg): 向客户端发送消息的方法。通过输出流向客户端发送消息。

4 测试与运行

4．1 程序测试

在程序代码基本完成后，经过不断的调试与修改，最后测试本次所设计的miniWeChat 能够正常运行，在基本功能上与微信没有太大差别，没有出现明显的错误和漏洞，但是在一些细节方面仍然需要完善，总的来说本次设计在功能上已经基本达到要求，其他细节方面有待以后完善。

4．2 程序运行

服务端运行初始界面如图 9 所示：

图 9 服务端初始界面

服务端主界面如图 10 所示：

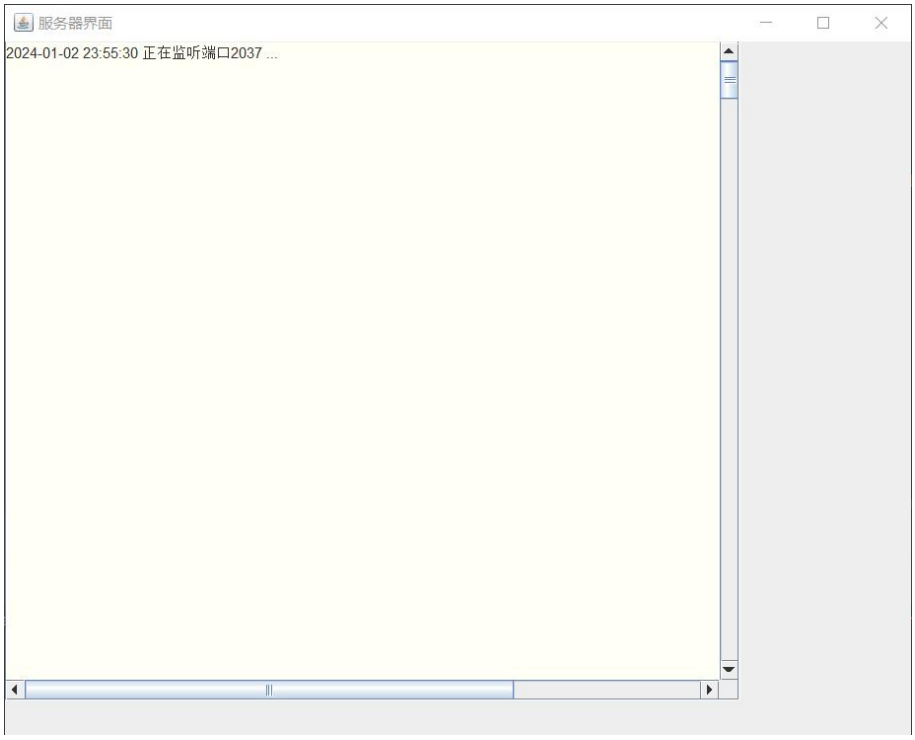

图 10 服务端主界面

客户端登录界面如图 11 所示：

图 11 客户端登录界面

客户端连接成功界面如图 12 所示：

图 12 客户端连接成功界面

客户端主界面如图 13 所示：

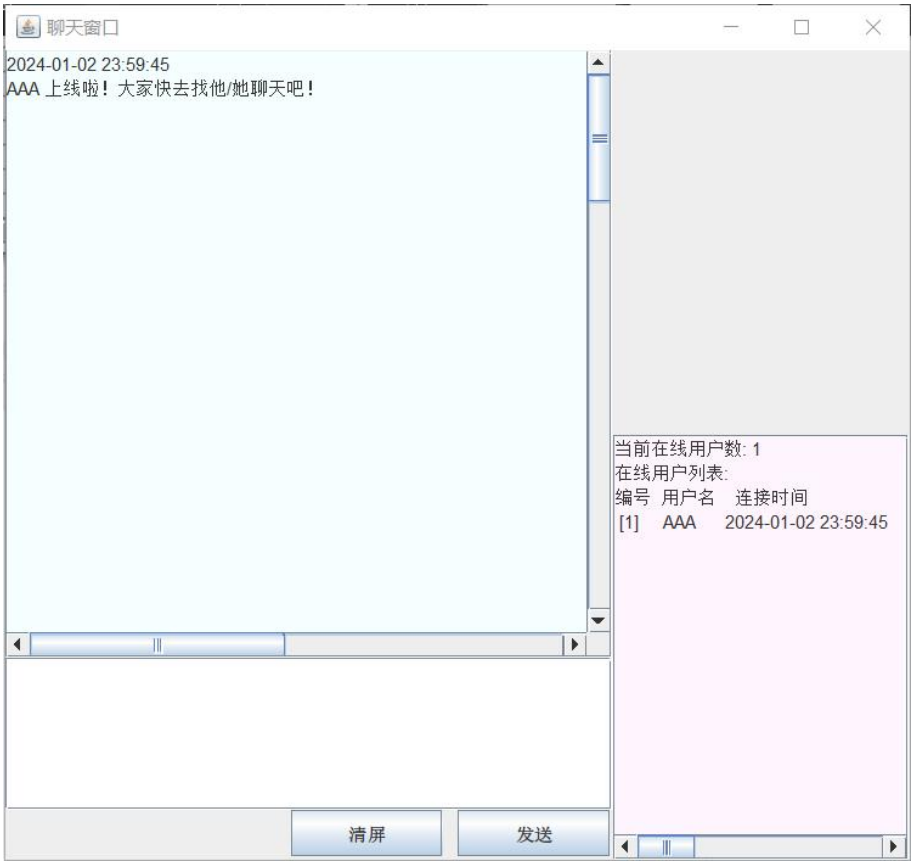

图 13 客户端主界面

多个客户端同时发信息如图 14-16 所示：

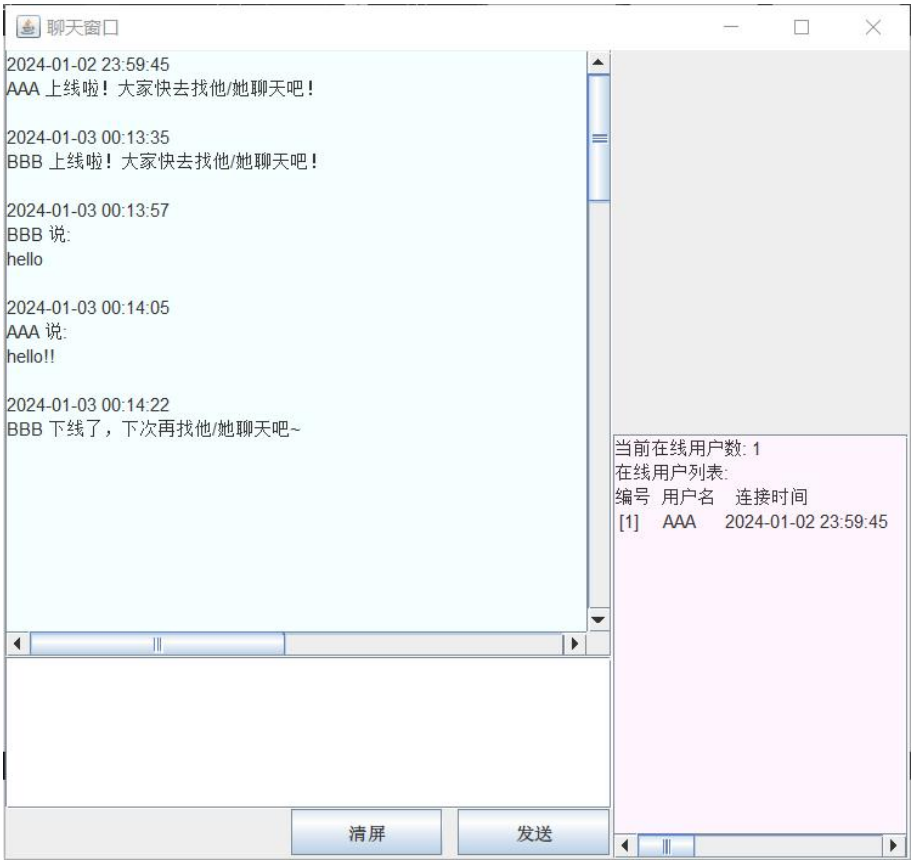

图 14 客户端 AAA 界面

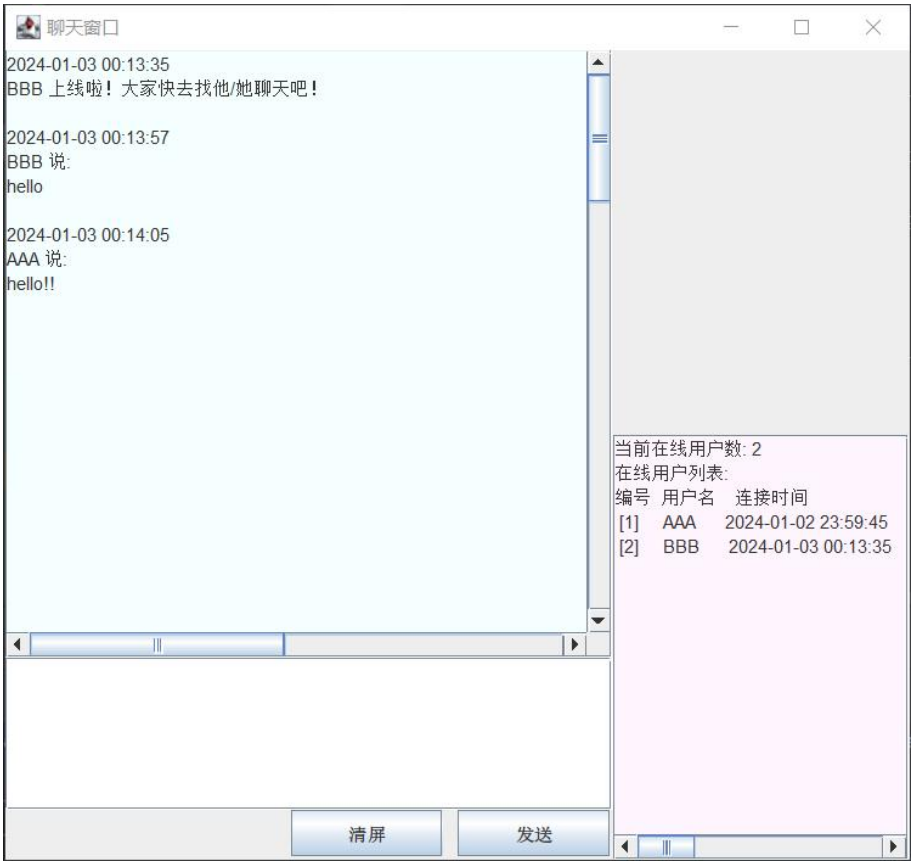

图 15 客户端 BBB 界面

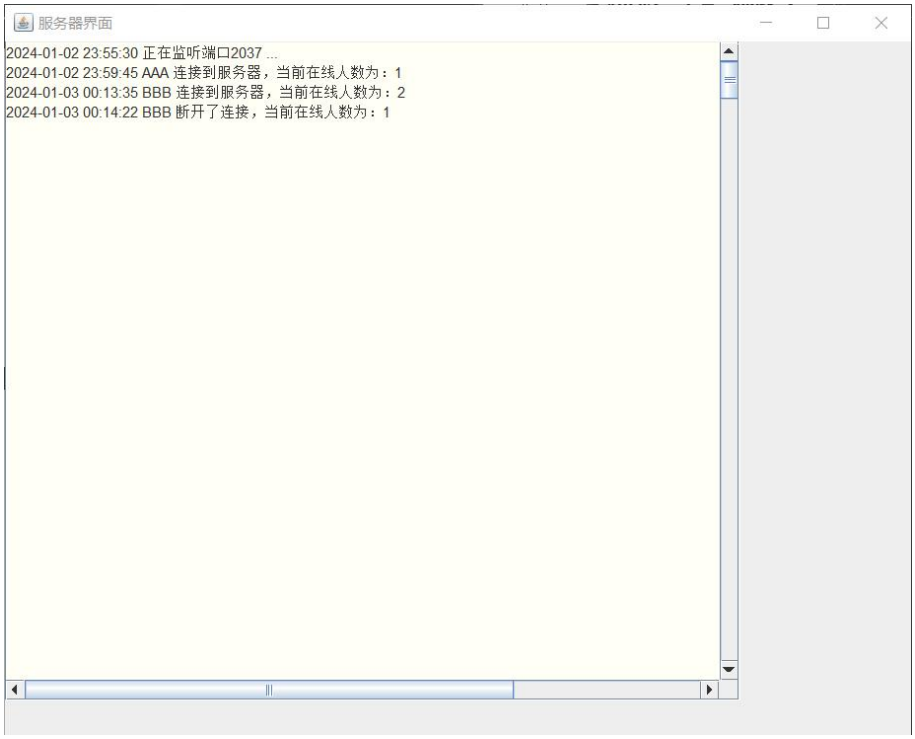
服务器界面
2024-01-02 23:55:30 正在监听端口2037 ...
2024-01-02 23:59:45 AAA 连接到服务器，当前在线人数为：1
2024-01-03 00:13:35 BBB 连接到服务器，当前在线人数为：2
2024-01-03 00:14:22 BBB 断开了连接，当前在线人数为：1

图 16 服务器界面

5. 总结

经过编写这个 miniWeChat，我认识到应该注意细节问题，虽然是很小的问题，但可以提高自己编程的能力，而且还可以培养自己编程的严谨性，同时还可以为以后的编程积累经验。编写完这个项目，我发现自己有很多的不足，感触最深的就是我们真的要扎扎实实的打基础!并且我感觉到只要我们自己肯下功夫学习，我们也可以做出很好的东西，不需要每次都抄袭别人的，只有自己的才是最珍贵的!

通过该课程设计，全面系统的理解了程序构造的一般原理和基本实现方法。把死板的课本知识变得生动有趣，激发了学习的积极性。把学过的计算机编译原理的知识强化，能够把课堂上学的知识通过自己设计的程序表示出来，加深了对理论知识的理解。现在通过自己动手做实验，从实践上认识了操作系统是如何处理命令的，课程设计中程序比较复杂，在调试时应该仔细。

参考文献

[1] 耿祥义. Java 大学实用教程[M]. 北京：清华大学出版社，2009.

[2] 耿祥义. Java 课程设计[M]. 北京：清华大学出版社，2008.

[3] 王鹏. Java Swing 图形界面开发与案例详解[M]. 北京：清华大学出版社，2008.

[4] 丁振凡. Java 语言实验教程[M]. 北京：北京邮电大学出版社，2005.

[5] 郑莉. Java 语言程序设计[M]. 北京：清华大学出版社，2006.