Java 网络编程实验报告

一、 实验目的

1.1 理解计算机网络的工作原理

1.2 掌握网络编程的基本概念

1.3 掌握 Java 中套接字编程技术

1.4 掌握 Java 中 URL 等类的使用

二、 相关知识

2.1 TCP socket ：通过超时重传和确认的两大机制保证数据传输的可靠性、有序性。当需要可靠、有序的数据传输时，选择 TCP 套接字可以提高传输效率，省去大量确认工作。

2.2 UDP socket ：轻便，灵活 ，快速，不需要建立并维持相关的数据通道，将数据打包完成后发送到指定 IP 地址的端口即可，设置端口便可接收打包好的数据包

三、 实验内容

3.1 使用 UDP 编程实现 P2P 的双人通信

3.2 使用 TCP 编程实现多人聊天室的建立

四、 实现思路

4.1 UDP socket 建立双人聊天室

4.1.1 代码实现思路

由于使用的时 UDP 编程，所以客户端和服务端的代码思路基本一致。

1. 构造函数中，使用 javax.swing 的组件，将聊天框、信息输入框、信息发送按钮布局完毕
2. 给“信息发送”按钮添加 addActionListener，实现 actionPerformed
3. 将输入框中 outMessage 中的内容存入字节数组中，使用 send 将其打包为 DatagramPacket,通过 DatagramSocket 发送到对应

IP 地址的对应端口中，并在聊天框中显示发送的内容

4. 使用 DatagramSocket.receive()方法接收发来的数据包，将其解析为 String message，添加到 inMessage 中并显示在聊天框中

4.1.2 代码截选

```
/*
 * 点击发送按钮时做出响应
 * 将outMessage中的内容打包发送
 */
public void actionPerformed(ActionEvent event) {
    byte  b[] = outMessage.getText().trim().getBytes();//除去不必要的空格得到内容变换为字节数组中的内容
    try {
        InetAddress address = InetAddress.getByName("127.0.0.1");//IP地址
        DatagramPacket data = new DatagramPacket(b, b.length, address, 1122);//将数据打包发送到对应IP的对应端口
        @SuppressWarnings("resource")
        DatagramSocket mail = new DatagramSocket();
        mail.send(data);//发送数据包
        inMessage.append("发送的数据是："+outMessage.getText().trim()+"\n");//将要发送的内容也显示在聊天框中
        inMessage.setCaretPosition(inMessage.getText().length());
    } catch (Exception e) {
    }
}

@SuppressWarnings("resource")
public void run() {
    DatagramPacket packet = null;
    DatagramSocket mail = null;
    byte b[] = new byte[8192];
    try {
        packet = new DatagramPacket(b, b.length);//接收数据包
        mail = new DatagramSocket(3344);////构造数据报套接字并将其绑定到本地主机上3344端口
    } catch (Exception e) {
    }
    while(true) {
        try {

            mail.receive(packet);//接收发来的数据包
            String message = new String(packet.getData(), 0, packet.getLength());//提取数据包中的信息
            inMessage.append("收到的数据来自："+packet.getAddress());//显示在显示框中
            inMessage.append("\n收到的数据是："+message+"\n");
            inMessage.setCaretPosition(inMessage.getText().length());
        } catch (Exception e) {
        }
    }
}
```

图 1：实现点击按钮后的发送和 Thread 的 run 方法

4.2 TCP socket 建立多人聊天室

4.2.1 代码实现思路

1. Utils.java 实现一系列收发的静态方法
   a. SendMsg：发送 message 到指定套接字
   b. RecvMsg：返回接收的 message
   c. GetTimeStr：获得当前的时间戳
   d. AddMsgRec：将 message 添加入可变容器 sb 中
   e. AddMsgRec：重载，将 message 显示在界面上

2. ChatServer.java 服务器端实现代码
   a. 使用 userList 记录聊天成员的端口、IP、名字和进入时间
   b. 使用 socket 的 accept 方法来接收客户端的连接
   c. 当客户端登陆时，检查用户名是否相同，相同发送报文，登陆

失败 FAIL，不相同把新用户添加到用户列表中，发送 SUCCESS

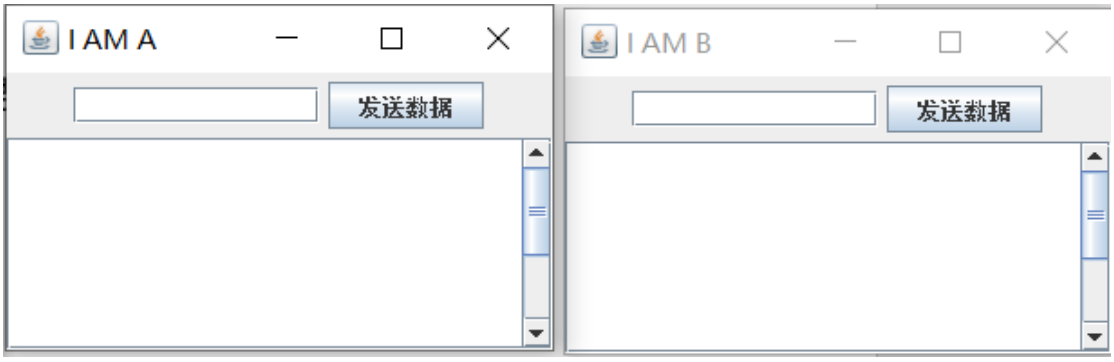
d. 当 bStart 为 True 时，通过报文的特定字段，来判断客户端的操作，分为四种：LOGIN TALKTO TALKTO_ALL LOGOUT，分别是登录、私聊、群聊、退出登录

e. 当用户退出聊天室时，更加用户界面，在列表中删除该用户

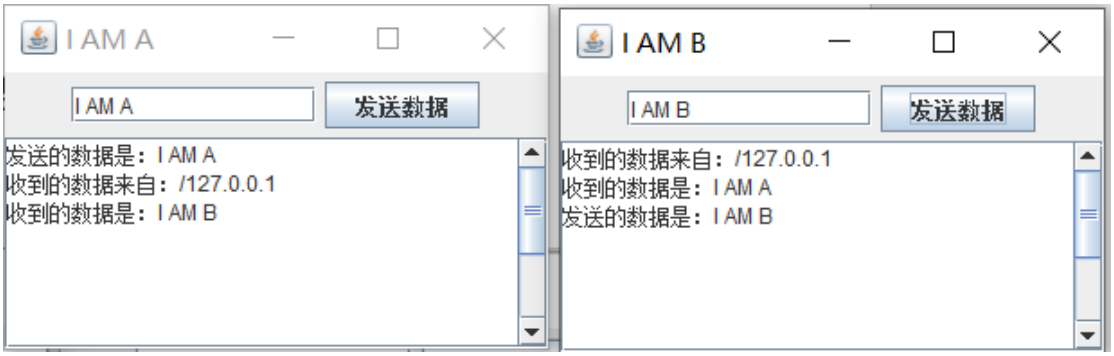
3. ChatClient.java 客户端实现代码
   a. 客户端的界面组装（具体看详细代码文件）
   b. 使用 EventQueue.invokeLater 使得 run 方法在指配的线程中被调用
   c. 使用 random 随机生成用户名
   d. 处理登陆事件按钮，获取服务器 IP 地址，端口号和用户名并启动线程
   e. 给发送按钮添加监听事件，调用方法 sendChatMsg()发送聊天信息
   f. 默认聊天对象为所有人，根据对象（私聊、所有人）构造消息报文，并将聊天报文发送到服务器并添加到聊天框并清空输入框
   g. Logout 函数发送用户退出聊天室报文，更新界面并关闭套接字
   h. CilentThread 函数连接 TCP 服务器，登陆成功即更新界面刷新人员列表，登陆失败则显示登陆失败信息；接收服务器报文，根据报文的关键字刷新界面信息

五、 结果截图

5.1 UDP 双人聊天室

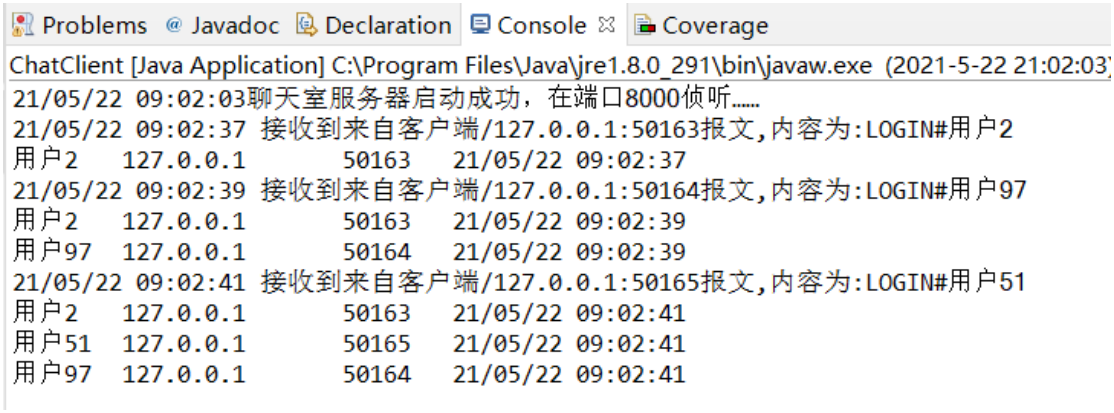


图二：UDP 双人聊天室截图

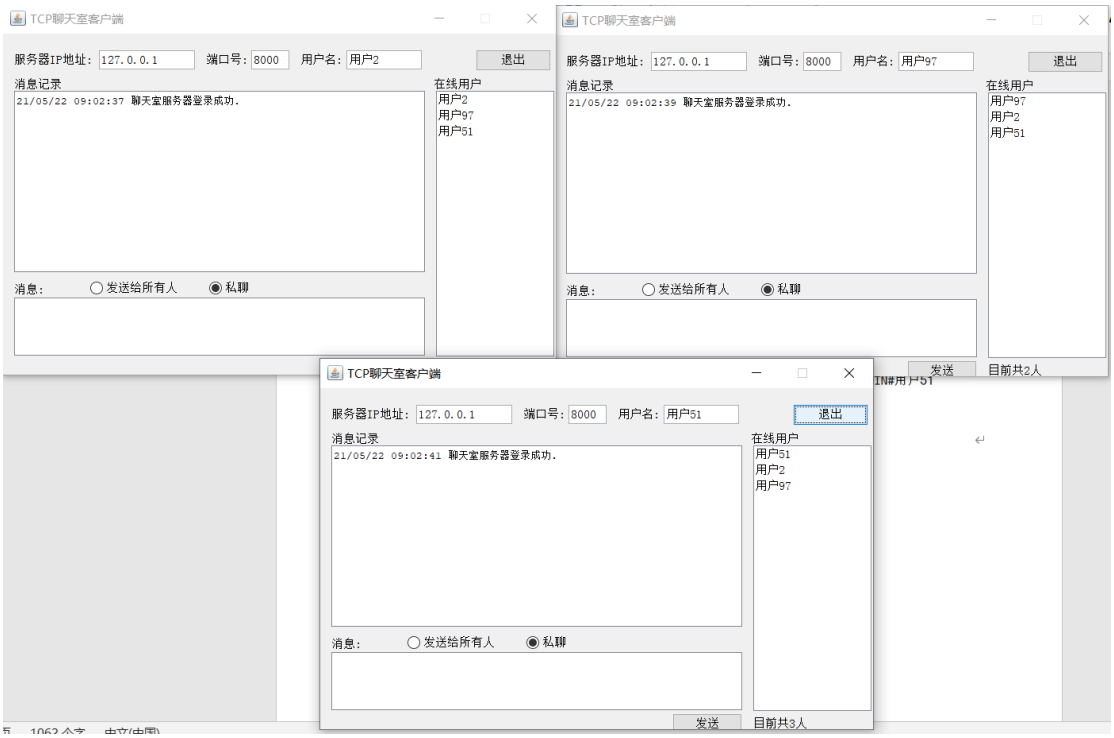


图三：互发信息截图

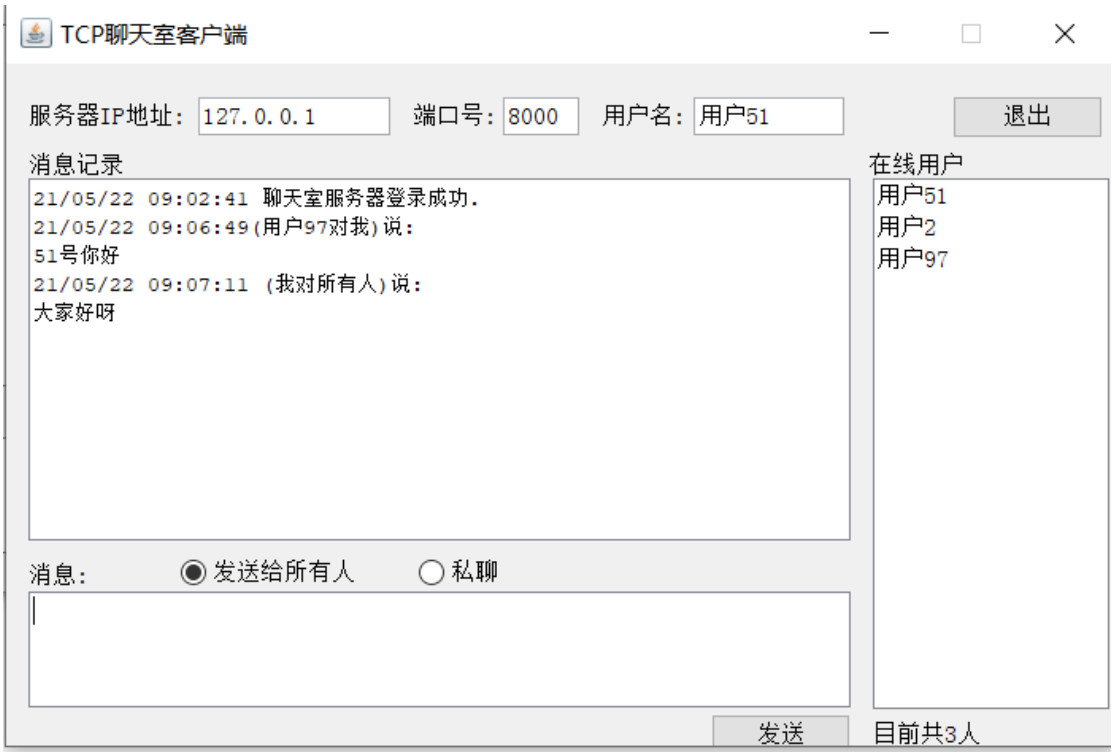
5.2 TCP 多人聊天室



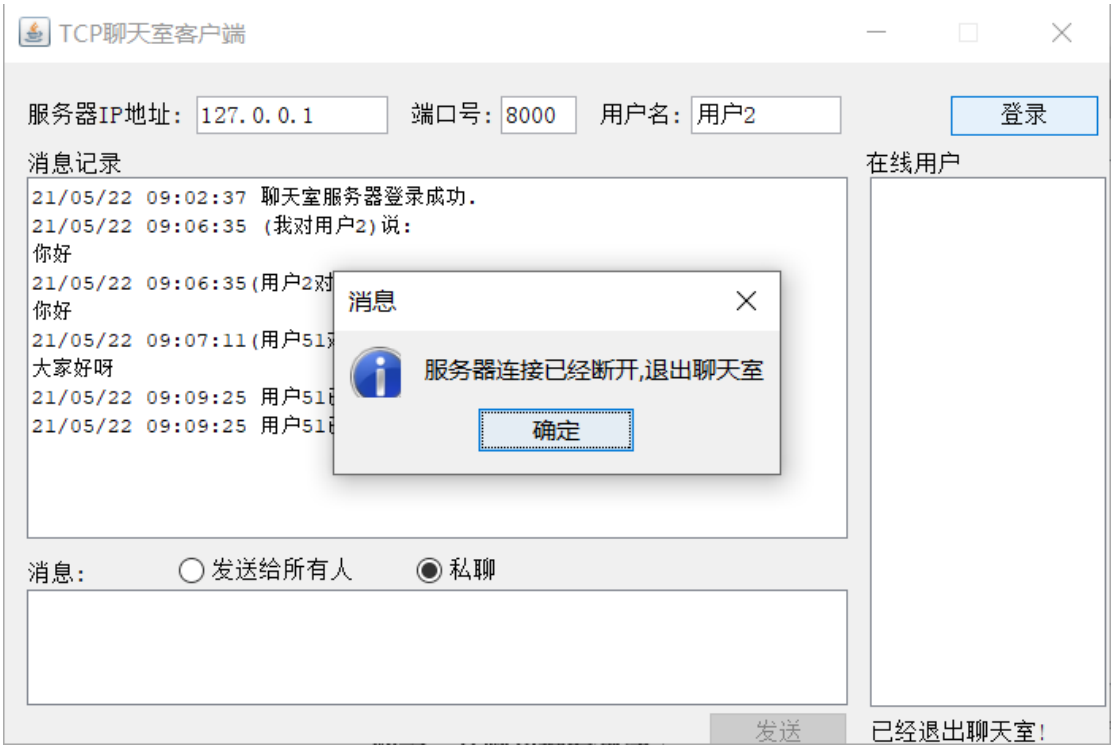
图四：三个用户登陆聊天室截图



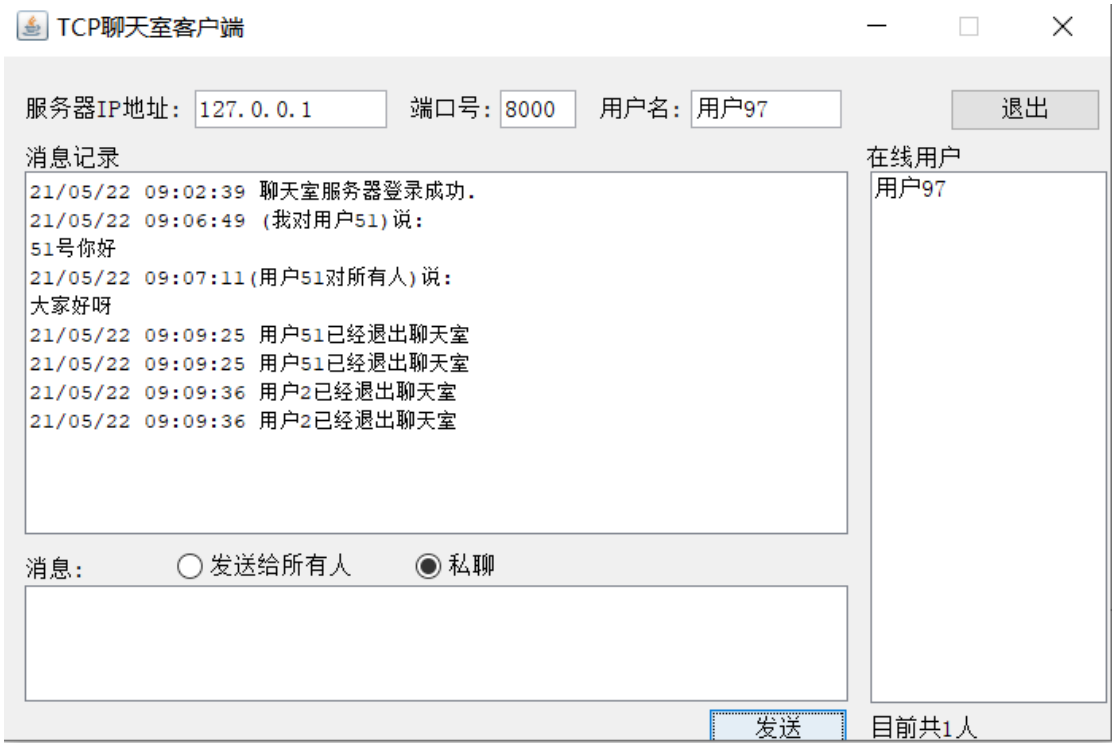
图五：三个客户端截图



图六：私聊和群发测试



图七：退出聊天室



图八：显示用户退出

六、 实验收获

附件：实验代码