Java 期末大作业设计报告

吴荆璞 17343117　　张莉斌 16340290

一、概述

我们小组设计了一个简易的图形界面聊天室，可以注册并登录多个用户，用户在进入聊天室后，可与聊天室内的其他用户进行文字聊天。实现过程主要分为聊天功能的实现以及 UI 界面的设计两步。其中用到了类和对象、超类与继承、接口及其实现、异常处理、多线程、文件存储、网络编程以及图形界面这几个知识点。

二、需求分析及设计构思

我们希望实现一个多人聊天室，这包含服务端、客户端两个部分，分别通过 Server、Client 两个类实现，所有客户端发送的消息，都经过服务端转发至所有在线的客户端，以实现多人聊天的功能。

实现 Client 及 Server 两个类以后，就可以在控制台进行聊天了，但这样很不方便，因此考虑增加图形界面。

UI 部分根据老师给的几个文件参考先做了第一个登陆界面，然后就应该还要有注册界面，为了增加一些特点，又加了一个修改文件，注册之后应该将用户的信息储存起来，所以又加了文件操作，文件操作这里用了 properties 配置文件，依据键=值的格式，可以比较简单的实现读取信息。 然后根据之前的客户端服务器端的文件，实现在对话框显示聊天内容。

三、实现过程

首先考虑 Client 类，它通过服务端这个中介与其它客户端进行通信，因此首先要建立与服务端的连接，客户端从键盘读入字符串并将其写入输出流供服务端接收，同时它需要接收并显示来自服务端的消息（也是字符串形式）。

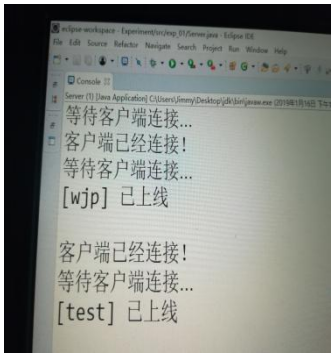
每个客户是一个线程，在构造函数中通过 socket 建立与服务端的连接，在输入了合法的用户名之后，用户从键盘输入的信息均被视为待发送的消息，这里

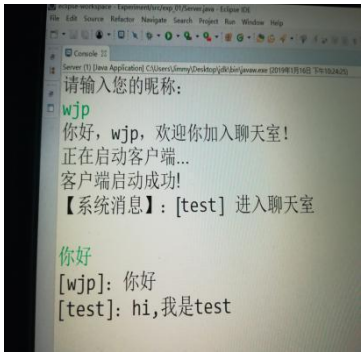
我采用了在作业 5-2（单人聊天）中用到的 OutputStreamWriter 类、PrintWriter 类，以将要发送的信息写进客户端的输出流。除发送消息外，还需要接受并显示来自服务端的消息，这里采用一个 ServerHandler 线程来实现，它用到 BufferedReader 类，从输入流中读取信息并在控制台显示。

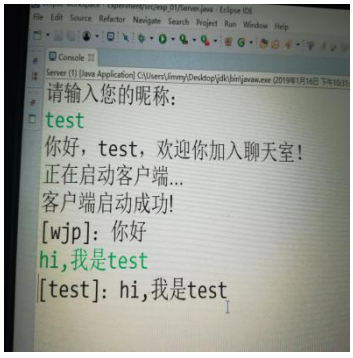
接下来实现 Server 类，它通过 ServerSocket 的 accept 方法连接到客户端，每个客户端由一个 ServerHandler 线程处理。与客户端类似地，Server 端也要有接受、发送消息两个功能。不同的是，Server 端发送的消息来自由客户端接受到的消息而非键盘读入，且消息要发给所有的客户端。这里利用了 list 存储针对所有用户的 PrintWriter，单独写一个 sendMessage 函数，用于向所有用户发送消息，连接到一个用户时，创建对应的 PrintWriter 并将其加入 list,用户下线是将其删除。

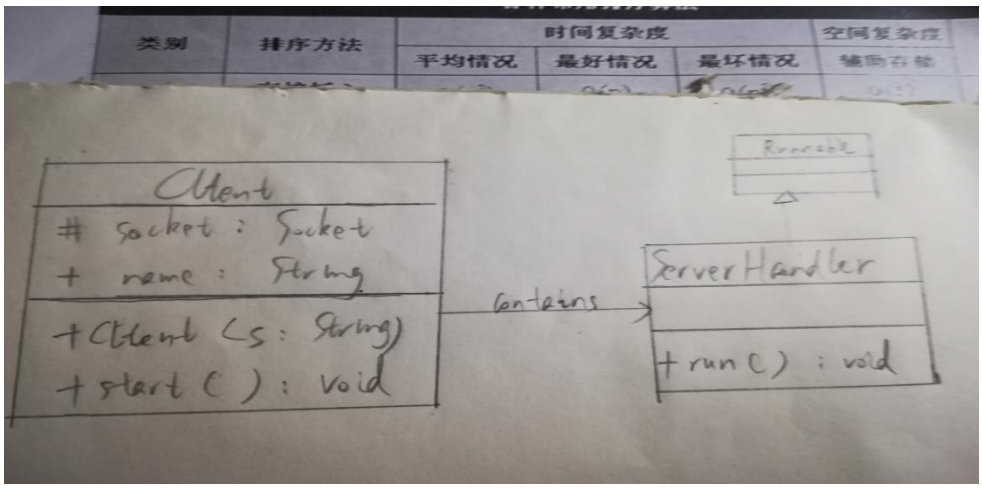
```
private List<PrintWriter> allOut;  //存储所有用户的 PrintWriter，用于写入数据以被用户接收
```

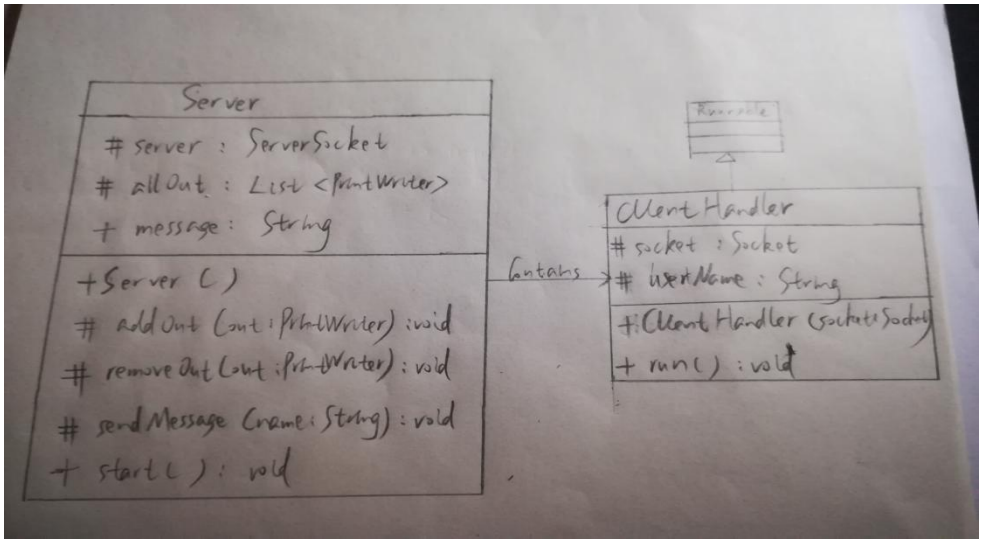
以上就实现了聊天室的基本功能，两个类的类图及运行结果见下图：











UI 部分比较简单，老师提供了很多的文件，直接套上去就可以了，当然还可以做出更美观的界面，这里只是追求做出来没有继续做精做良。

模块说明：

chatFrame 模块

其余的几个 UI 文件和这个都类似，这里实现了 client 界面部分。

```
this.setLayout(new BorderLayout());
chatContent = new JTextArea(12, 34); // 创建一个文本域
// 创建一个滚动面板，将文本域作为其显示组件
JScrollPane showPanel = new JScrollPane(chatContent);
chatContent.setEditable(false);     // 设置文本域不可编辑

chatContent.append("你好，"+name+"，欢迎加入游戏聊天室！\n");
```

后面的聊天内容都是通过这句 chatContent.append(″？″)发送到对话框

message 是服务器端的输出流。

```
//接收服务端发过来的消息，并输出
class ServerHandler implements Runnable {
    public void run(){
        try {
            BufferedReader in =new BufferedReader(new InputStreamReader (socket.getInputStream(),"UTF-8"));
            String message =null;
            while((message=in.readLine())!=null) {
                System.out.println(message);
                chatContent.append(message + "\n");
            }
        }catch(Exception e){
        }
    }
}
```

LoginFrame

```
    private JLabel accountJLabel=new JLabel("请输入账号");
    private JTextField accountJTextField=new JTextField(10);
    private JLabel passwordJLabel=new JLabel("请输入密码");
    private JPasswordField passwordJPasswordField=new JPasswordField(10);//用*代替输入的密码做到保密性
    private JButton loginJButton=new JButton("登录");
    private JButton registerJButton=new JButton("注册");
    private JButton exitJButton=new JButton("退出");
```

passwordJPasswordField 用了点符号代替密码的显示，做到保密。

FileOperation

这个模块利用了 Properties 文件来操作的，Java Properties 类 Java 数据结构 Properties 继承于 Hashtable.表示一个持久的属性集.属性列表中每个键及其对应值都是一个字符串。Properties 类存在于胞 Java.util 中，该类继承自 Hashtable

1. getProperty (String key) ， 用指定的键在此属性列表中搜索属性。也就是通过参数 key ，得到 key 所对应的 value。
2. load (InputStream inStream) ，从输入流中读取属性列表（键和元素对）。通过对指定的文件（比如说上面的 test.properties 文件）进行装载来获取该文件中的所有键 - 值对。以供 getProperty (String key) 来搜索。
3. setProperty (String key, String value) ，调用 Hashtable 的方法 put 。他通过调用基类的 put 方法来设置 键 - 值对。
4. store (OutputStream out, String comments) ， 以适合使用 load 方法加载到 Properties 表中的格式，将此 Properties 表中的属性列表（键和元素对）写入输出流。与 load 方法相反，该方法将键 - 值对写入到指定的文件中去。
5. clear () ，清除所有装载的 键 - 值对。该方法在基类中提供。

设置键值对，用#分割

```
public static void updateUser(String account,String password,String name) {
    pps.setProperty(account,password+"#"+name);//调用基类的set方法来设置 键 - 值对
    //pps.store(FileReader, "Update " + account + " name");写入输入流
    listInfo();
}
```

利用键值获取所有信息

```
//通过账号获取当前用户所有信息
public static void getInfoByAccount(String account) {
    userInfo=pps.getProperty(account);//用指定的键在此属性列表中搜索属性
    if(userInfo!=null) {
        String[] infos=userInfo.split("#");
        information.account=account;
        information.password=infos[0];
        information.name=infos[1];
    }
    return;
}
```

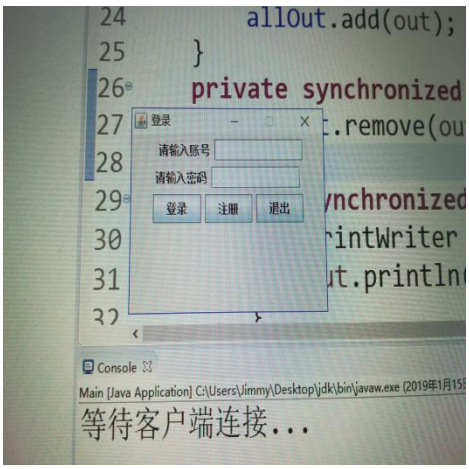
类初次被加载的时候，会按照 static 块的顺序来执行每个 static 块，只在类加载的时候执行一次可以优化代码

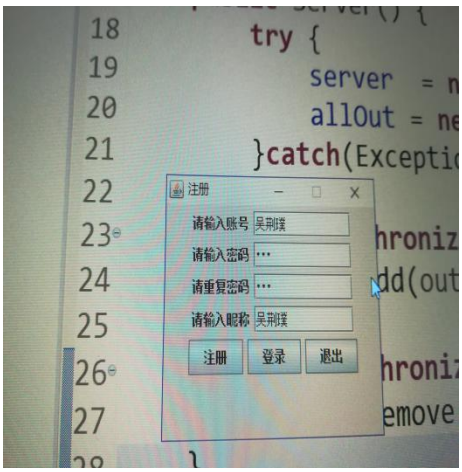
```
    static {//
        pps=new Properties();
        FileReader reader = null;
        try {
            reader = new FileReader(fileName);//按字符读取流中数据
        } catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        try {
            pps.load(reader);//加载输入流
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        finally {
            try {
                reader.close();//关闭
            }
            catch (Exception ex) {
            }
        }
    }
```

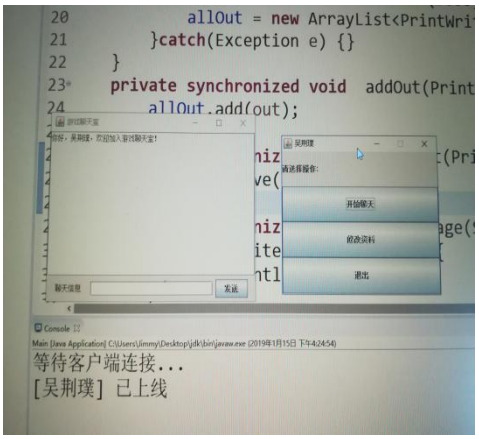
四、实际效果

在这一部分结合图片展示用户的注册、登录、聊天、退出的基本功能及其实际效果。

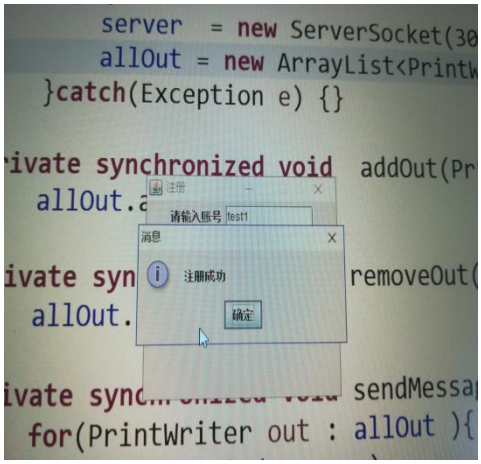
首先运行 Server 类开启服务端，之后运行 Main 类，进入初始的登录注册界面。我们先注册一个名为吴荆璞的用户，注册成功后回到登录界面登录，登录成功后会显示三栏选项：进入聊天室、修改资料、退出，此处我们选择进入聊天室。这时聊天室内就有了第一个成员。

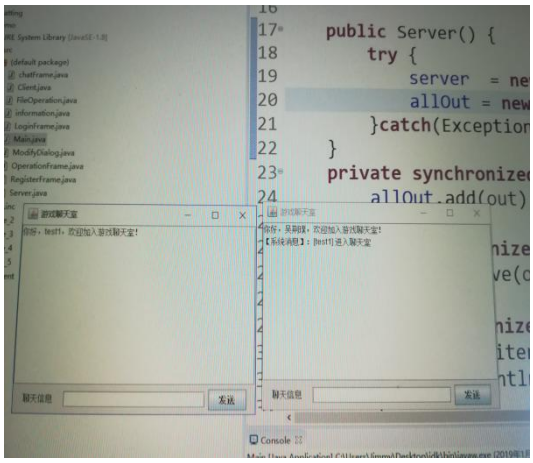






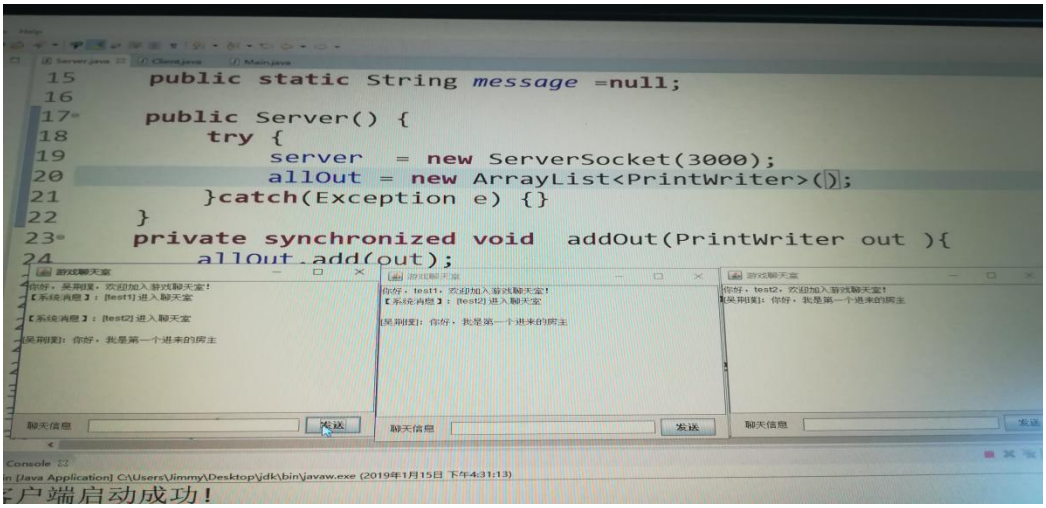
接下来我们再注册、登录两个成员 test1、test2 并让他们也进入聊天室，可以观察到，在新成员进入聊天室时，已经在聊天室内的用户将收到来自系统的相应提示。

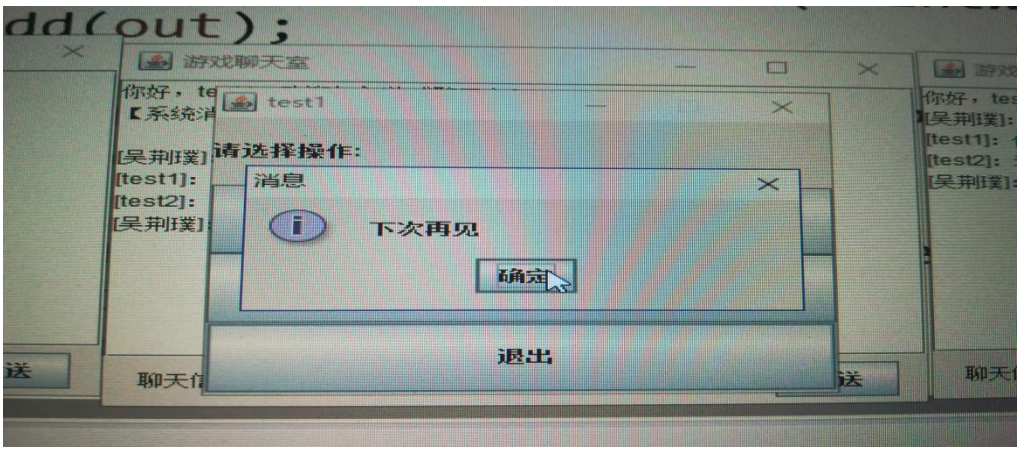


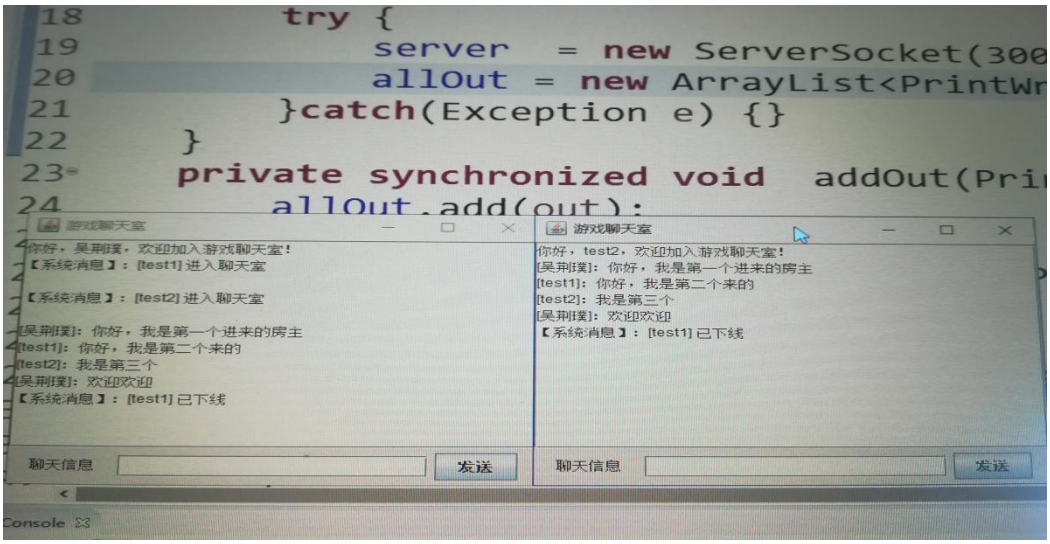


这时我们让在最先进入的用户说一句话，可以看到消息显示在了三个用户的聊天界面中，之后依次让各用户说一句话，均在各用户的聊天界面中有了相应显示。

之后我们试着让用户 test1 退出，这时聊天室内只剩吴荆璞和 test2 两个用户，而他们将收到来自系统的 test1 的下线提示。







五、总结与反思

这个程序由两个人完成，我们都是第一次与别人合作写代码，在沟通上存在

很多不足，代码的注释也不是很明晰、全面，以致对于对方的代码的理解有一定的偏差，走了一些弯路，这在几次当面沟通后得到了解决。

我们本来设想的是实现一个类似你画我猜的“你说我猜”小游戏，其基础是一个多人聊天室，于是我们便先构思设计一个聊天室。结果后来发现要实现上述游戏需要限定轮数、发言次序，需要限定时间并让每个人轮流画，其他人猜，而这又需要随机产生一定的词语发给该用户，并判断其他人的回答是否与之相符，并根据次序进行计分等等。我们觉得这个问题有一点复杂，特别是在我们对线程的使用还不是很熟练时，感觉有一些困难，且时间也比较有限，于是便止步于聊天室了，只在基本通信功能的基础上增加了登录注册、聊天、退出的 UI 界面以及聊天记录文件存储的功能。

与我们不久前初级实训中用 C++做的 Agenda 议程管理系统不同，这个程序逻辑上非常简单，只是利用一些封装好的类，来实现网络交互、图形界面、文件存储等功能。因此可以说这是一个比较简陋的项目。这和我们对 java 的熟悉程度不如 C++有很大的关系，不过在这个过程中，我们依然对 java 网络编程、线程、图形界面等在 C++中没有学过的内容有了一些了解，并体会了各种高度封装的类可以大大简化我们的编程。同时，我们也初次体验了与他人合作写代码，经常沟通、对程序设计的步骤、框架有统一的意见是非常重要的，此外，也要注意代码书写的规范，并在适当位置添加注释，以便于他人的阅读。