课程设计报告二：聊天室系统

1. 课程设计要求与目的

目的：编写一个小型Java聊天室系统，掌握Java网络通信、多线程、IO文件操作等高级应用编程技能。

要求：以课本第15章 Java网络通信例15.3、15.4的源代码为基础，编写一个小型Java聊天室系统。

完成如下功能：

1、多客户端模式下，实现客户与客户的单独通信，要求信息通过服务器中转

2、端到端的通信，实现并行通信模式（不再是你说一句，我说一句，一端的信息发送不受另一端的影响）

3、添加图形界面（选做）

1. 系统设计

设计思想:

利用socket套接字通信

多线程处理不同任务

用Properties在本地存储注册账号

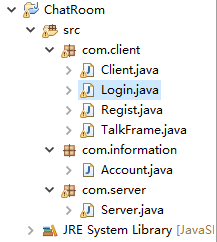
下载安装windowbuilder插件并用其设计图形界面

整体设计（类与类的关系）：

Server类（服务器），包含几个继承Runnable的内部类，用于处理客户 端请求。

Client类、Login类、TalkFrame类、Regist类。Login类用于登陆，Regist类用于注册，TalkFrame类用于对话，Client实例则被这三个类调用。

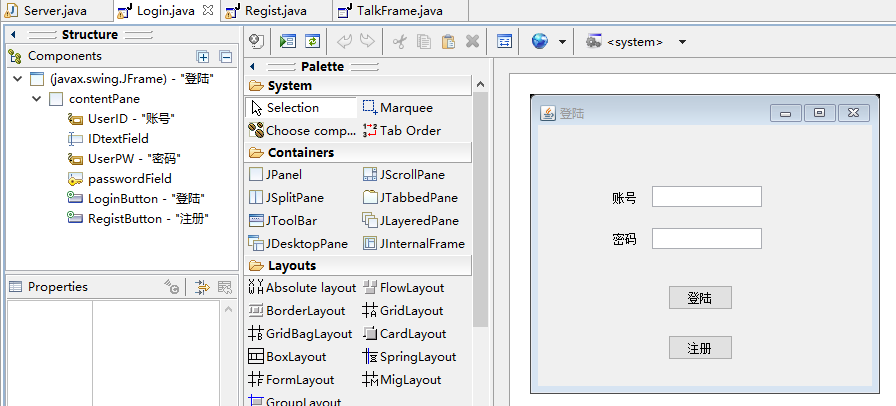
Account类，这个类很简单，只有id和password两个属性和相应的set方法。



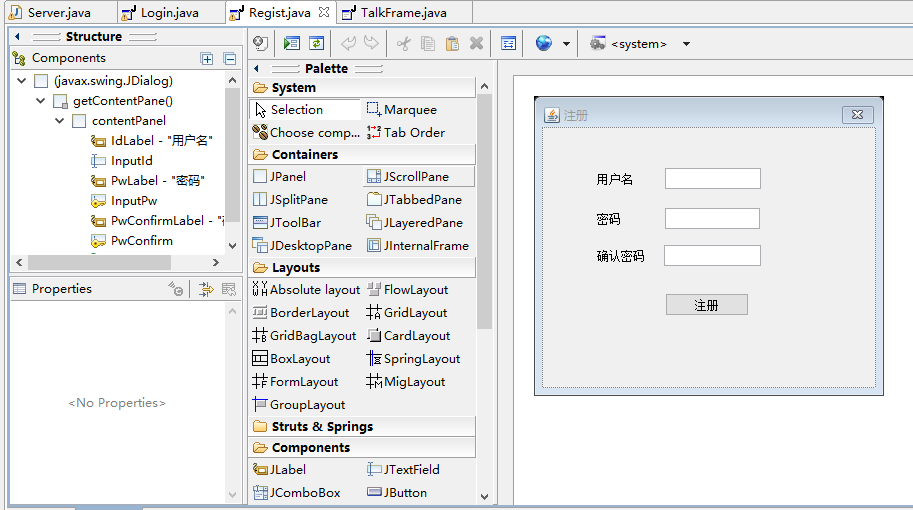
局部设计（单个类的设计）：

Client类中只定义了Login、Regist、TalkFrame中需要的socket和流，和 对这些属性初始化的构造函数。

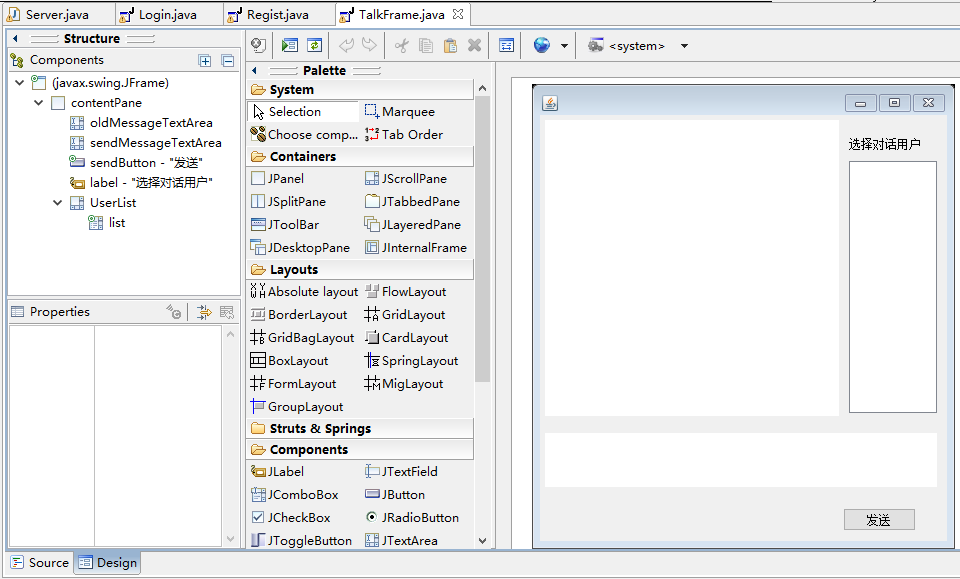
Login类：用于登陆，包含账号label、密码label、账号JTextField、密码 JPasswordField、登陆按钮和注册按钮这些组件，还有一些监听器。



Regist类：用于注册，包含用户名label、密码labl、确认密码label、用 户名JTextField、密码JPasswordField、、确认密码JpasswordField和注册按钮 这些组件，还有一些监听器。



TalkFrame类：用于对话，包含选择对话用户label、UserList在线用户JscrollPane 面板、list用户列表、显示对话文本区域、发送信息文本区域和发送消息按钮这些组件， 以及对应的监听器。TalkFrame中有两个继承了Runnable的内部类，分别处理在线用户 显示和消息的发送。



Server类：这个类我设计得有些复杂。有三个ServerSocket分别监听登陆端口、注册端口、显示在线用户端口（我实在是没有办法才这样设计的，不然无法很好地处理客户端传递过来的信息）。还有好几个继承Runnable的内部类，用于处理注册、登陆、显示在线用户、用户对话。

1. 系统实现

采用的主要技术、关键算法步骤（或代码摘要）及相应解释

由于涉及了账号注册的功能，必须能在本地保存账号，但我不会在Java中数据库插入数据库，所以只能想办法保存数据，一开始用的是hashtable，但保存到存盘有点困难，在网上搜到的方法是使用XML文件存取可序列化的对象的类，由于hashtable已经实现了序列化接口，所以可以这样实现，我用这种方法写了一遍，发现非常麻烦，再次百度，发现了properties这个集合类，查看JDK文档，了解了properties的load和store方法，这两个方法能很容易地将账号存储在磁盘的文件中。进而，我改用了properties存储账号。

**public** Properties userInformation;

与磁盘文件建立联系

userInformation = **new** Properties();

uis = **new** FileInputStream("f:/userInfo.properties");

uos = **new** FileOutputStream("f:/userInfo.properties", **true**);

userInformation.load(uis);

将用户名和密码存储到内存中 userInformation.setProperty(account.getId(),account.getPW());

将用户名和密码保存到文件中

userInformation.store(uos, **null**);

在处理server接收到的信息时，我用了三个ServerSocket进行处理，RegistServer监听规定的注册端口，对接收的注册信息进行处理；LoginServer监听登陆端口，对登陆后的对话信息进行规定，要求客户端传递的信息必须以接收者id+“：”+发送者id+“：”+发送信息的形式传递给服务器，服务器对传递过来的字符串进行截取，将信息发送给接收者客户端。

监听发送消息按钮，在字符串前面加上规定的信息然后传递给服务器

sendButton.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent ae) {

String str = sendMessageTextArea.getText();

String sendMessage = recieverId+":"+id+":"+"\n";

**if**(!sendMessage.equals("")){

client.pw.print(sendMessage);

client.pw.flush();

oldMessageTextArea.append(sendMessage);

oldMessageTextArea.setText("");

}

}

});

服务器截取信息，获取接收者id，找到对应socket，将信息传递过去

String s = br.readLine();

String accountId =s.substring(0,s.indexOf(":"));

String message = s.substring(s.indexOf(":")+1);

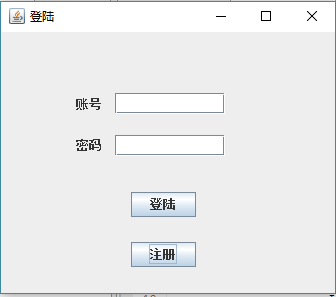
receiveClient = clientConnection.get(accountId);

PrintWriter pw= **new** PrintWriter(receiveClient.getOutputStream());

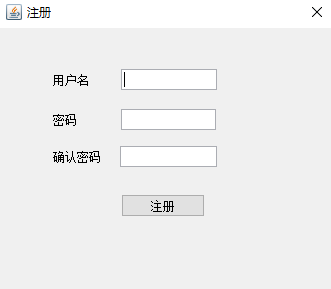
pw.println(message);

1. 系统测试

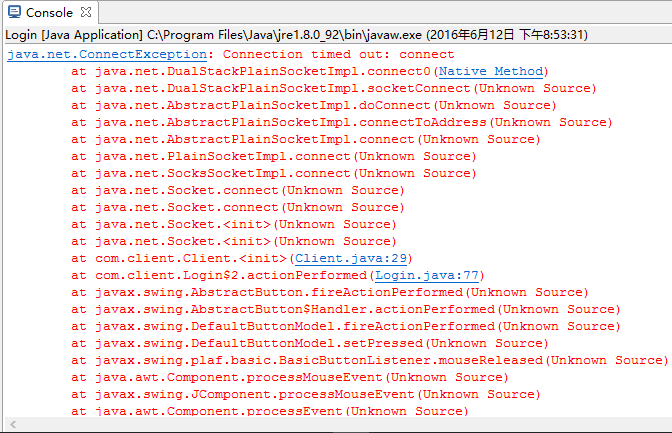
登陆界面



注册界面



我写的程序失败了，每次在登陆或注册时都报如下错误：



客户端一直不能和服务器连接！！！有点郁闷！

我正在想办法解决！后面的测试暂时不能进行，我会在解决后补上的！

1. 课程设计总结

通过这次实验，我掌握了socket的通信过程，熟练的掌握了各种流的操作。由于学过QT，所以我试着百度了一下Java能不能这样设计界面，结果百度到了windowbuilder插件，下载安装后感觉挺好用的，事件的处理和Qt、Android（了解过）都很相似，很快就掌握了用法。一开始时一直纠结怎么同时处理登陆、注册|、对话时信息和信息接收方传递不同进程传递的消息，想了几天，最终想到了注册用另一个端口监听，再对对话信息进行规定。

总之，这次实验我学到了好多，所写的只是一小部分，两个星期以来一直在对程序修修改改，流程图也画了一些，由于时间紧迫不能一一列出。暂且就写这样吧！

补：1.连接错误是把address错写成了127.0.0.0（本应该是127.0.0.1）

2.程序总是卡死在登陆界面和注册界面上，debug好久发现程序停在

BufferedInputStream bis = **new** BufferedInputStream(is);

ois = **new** ObjectInputStream(bis);

这一段代码上，百度发现原因是建立ObjectInputStream对象是需要先接收一定的header数据，接收到这些数据之前会处于阻塞状态。以下为Java API文档的说明 ：

Creates an ObjectInputStream that reads from the specified InputStream.

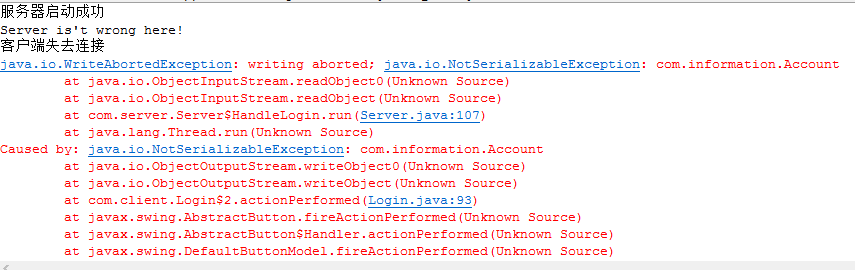
A serialization stream header is read from the stream and verified.

This constrUCtor will block until the corresponding ObjectOutputStream

has written and flushed the header.

所以不能在Client类里定义ObjectInputStream流！我在创建client对象后再创建ObjectInputStream流了！

3.刚改完一个bug，又出来一个！汗！完成一个程序，debug占了80%时间



这个异常说的是Account类未序列化！

Debug发现程序在client.oos.writeObject(account)这一句抛出异常，把Account同步后还是抛出同样的异常！

终于找到了解决方法：

如果将一个不能串行化的对象传递给writeObject方法，那么它抛出一个NotSerializable Exception。对象只能在它的类实现了Serializable接口的情况下被串行化

自己新建的类class，新建的时候引入Serializable接口就行了

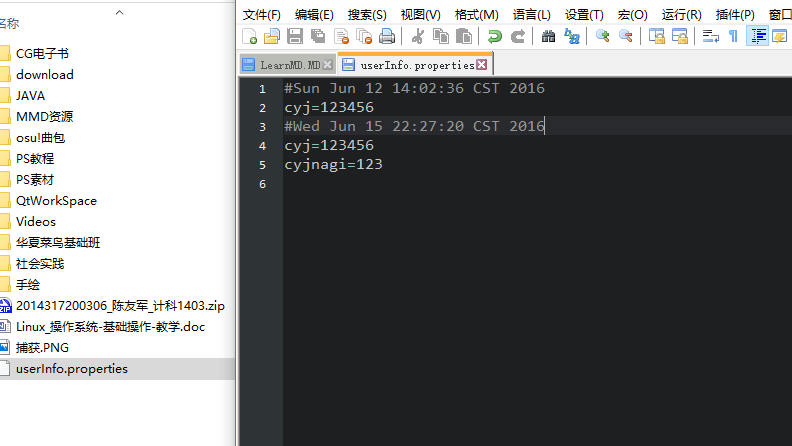
public class class implements Serializable{}

然后再根据提示在类里加一个private static final long serialVersionUID = 1L;就行了。

之后就可以使用writeObject(class的实例)之类的函数了。

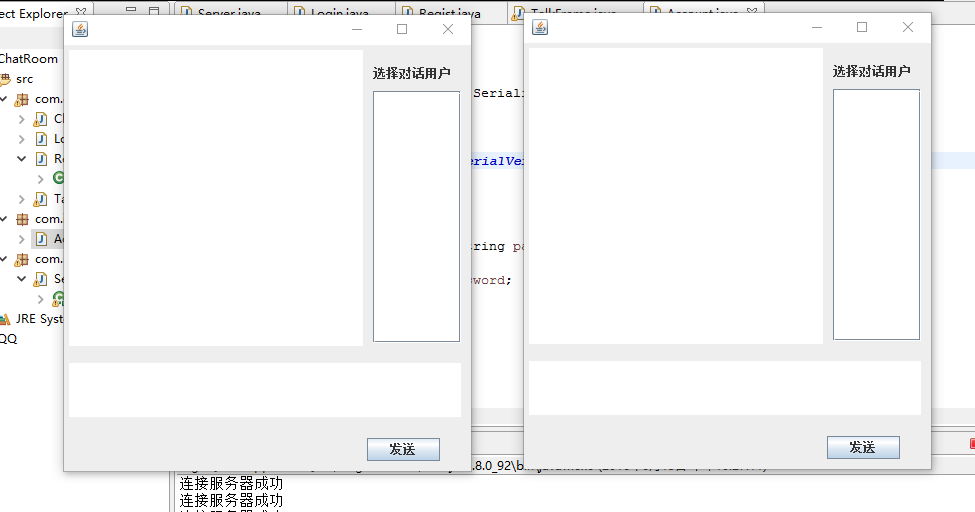
1. 我写的Account类未串行化，现在去改！

注册成功，登陆成功，userInfo.properties记录如下：



终于成功出现了对话框！改了那么久终于出现了，有点激动！

不过，又出现问题了！



明明登陆了两个用户，选择对话用户框却没有出现对方id！另外，当我关掉程序后，服务器陷入死循环，异常如下：



**快要期末考试了！但是显示用户模块还是有问题，没时间改了！，我重新写了一个简化版的，这个写成功了！等暑假再慢慢改这个有问题的版本吧！**

这个重新写的大体上和之前的差不多，就是把注册功能和显示在线用户的功能删除了，改为三个固定的账号了。当然，也可以手动在userInfo.properties文件中添加账户！不过聊天界面的那三个账号选项是固定的，只能通过代码改！

花了一天重新写这个聊天器，发现了上个版本的好多错误！感觉自己对这些知识掌握的更加熟练了，写代码的速度也变快了！

**还有，运行程序前先阅读我写的markdown文件（readme.md）!**

我是一边写（改）代码一边写实验报告，所以写得有些乱！就这么多吧！

陈友军

2014317200306

计科1403

2016.6.19