

浙江大学计算机学院

Java 程序设计课程报告

2021—2022 学年秋冬学期

题目	多用户多会话网络在线聊天软件
学号	3190104534
学生姓名	庞懿非
所在专业	计算机科学与技术
所在班级	计算机科学与技术 1903

目 录

1 引言.....	1
1.1 设计目的.....	1
1.2 设计说明.....	2
2 总体设计.....	3
2.1 功能模块设计.....	3
2.2 流程图设计.....	3
3 详细设计.....	5
3.1 客户端整体布局设计	5
3.2 客户端登录页面设计.....	5
3.3 客户端注册页面设计.....	7
3.4 客户端与服务器交互页面设计.....	9
3.5 客户端会话页面设计.....	11
3.6 客户端其他设计.....	12
3.7 服务器设计.....	13
3.8 服务器相关类设计.....	14
4 测试与运行.....	16
4.1 程序测试.....	16
4.2 程序运行.....	16
5 总结.....	22

1 引言

本次开发的内容为多用户多会话网络在线聊天软件，是一个综合了 GUI 界面设计以及 Socket 网络通信的实验，具有相当大的挑战性。由于过往的编程大多是以 CLI 作为输入输出，任何的命令都是以字母的形式进行解析并且显示。在 GUI 编程中，内容展示以及事件驱动都与以往的变成模式有较大的不同，更需要我们进行全面的设计与整体的规划。网络通信 Socket 是一种常见的网络通信的工具，使用 TCP 建立连接并且可靠地传输字节流信息。项目还可以结合数据库进行信息的存储和管理，使得应用性得到进一步提升。

1.1 设计目的

在线聊天软件我们每天都在使用，大到钉钉、微信、QQ，小到匿名提问箱 Tape 等小型聊天工具，其实本质上都可以认为是一个多用户多会话网络在线聊天软件。项目中我实现的多用户多会话网络在线聊天软件是一个网络聊天软件的雏形，其功能远远比不上流行的聊天软件，但是仍然可以很好的使用。

本多用户多会话网络在线聊天软件完全由 Java 实现，具体功能如下：

- (1) 服务端先设定端口 9999，并启动服务器。服务器记录所有的用户以及会话的信息。
- (2) 客户端开启之后自动与服务器连接，连接成功后进入登录界面。
- (3) 客户端可以注册用户，输入“用户名、密码、确认密码”完成注册，每个用户有一个唯一标识 ID，用户名可以重复，注册成功后回到登录界面。
- (4) 客户端可以选择登录并且通过服务器验证，登录成功后进入与服务器的交互页面。
- (5) 在与服务器的交互页面中客户端可以点击按钮向服务器发起请求并且获得服务器所有的会话的信息以及用户的基本信息。
- (6) 在与服务器的交互页面中客户端可以新建会话并且指定会话的名字（即类比新建群和群名），建立成功后可以查询到结果。
- (7) 在与服务器的交互页面中客户端可以输入会话 ID 并且点击进入会话，进入成功后切换到会话界面。
- (8) 在会话界面，客户端可以发送信息，并且可以接收到同一个会话的其他客户发送的信息，从而实现相互交流。

- (9) 在会话界面，客户端可以选择返回到与服务器交互的页面，如果之后仍然进入该会话，期间会保留全部会话信息，并且可以使用滚动条查看。如果进入其他会话，信息将被清空。
- (10) 在会话界面，客户端可以退出会话，服务器会删除用户在会话中的信息并且页面切换到与服务器交互的页面。
- (11) 使用 JDBC 与 Mysql 连接，存储所有的用户数据以及会话数据，并且记录统计数据对未来的功能提升（比如用户等级信息）奠定基础，这一重要的部分在设计中，但是没有完全实现所以最后去掉了这一模块。

1. 2 设计说明

本程序采用 Java 程序设计语言，在 Eclipse 平台下编辑、编译与调试。具体程序均由本人开发而成。图形界面开发使用了 [WindowBuilder](#) 的图形开发工具，Socket 使用基本的 Socket 类完成。整个项目全部是自己完成，没有抄袭或者使用他人努力成果。

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) 客户端发起请求，请求内容包括：登录、注册、数据查询、新建会话、进入会话、发送消息、退出会话；
- (2) 服务端响应请求，存储所有的用户和会话数据；
- (3) 客户端的页面切换，文本信息显示。

程序的总体功能如图 1 所示：

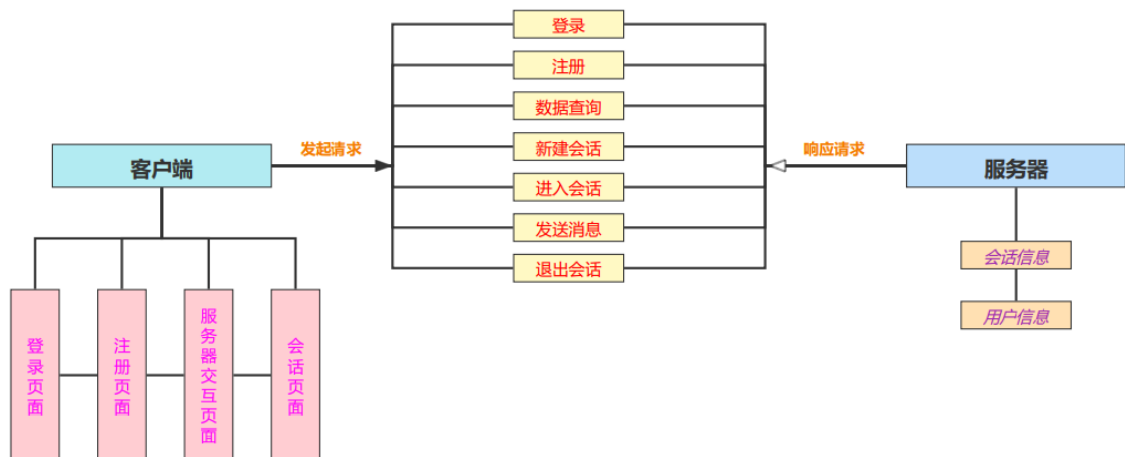


图 1 总体功能图

2.2 流程图设计

客户端总体流程如图 2 所示：

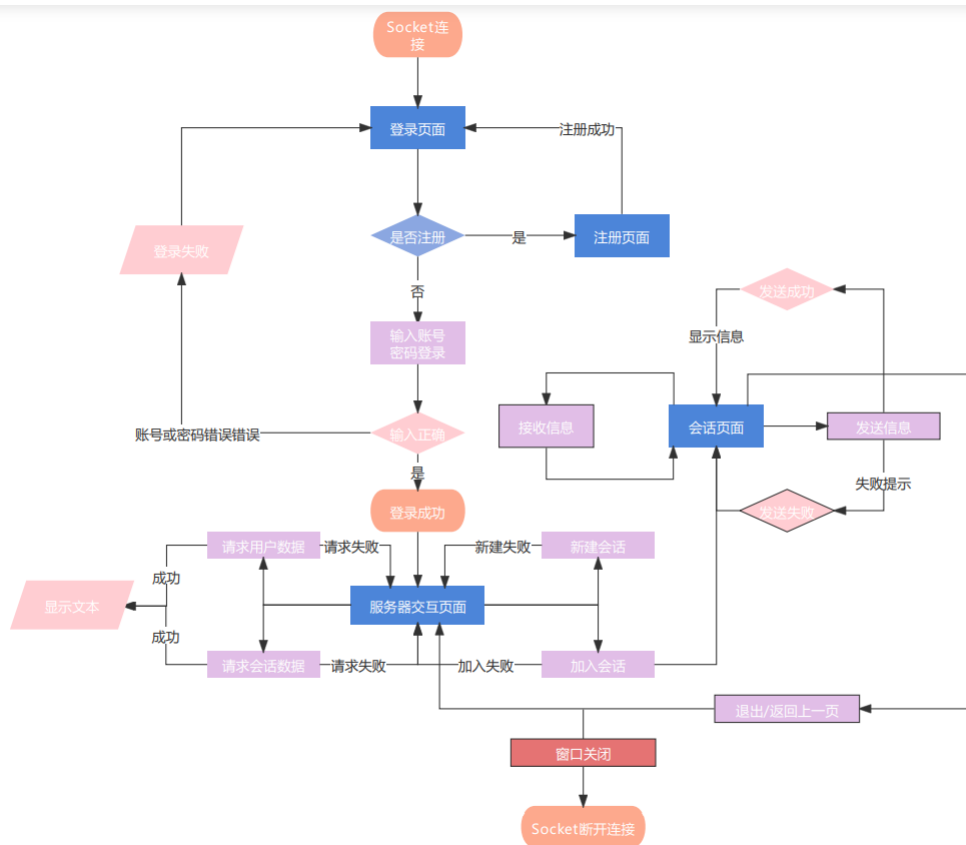


图 2 客户端总体流程

客户端总体流程如图 3 所示：

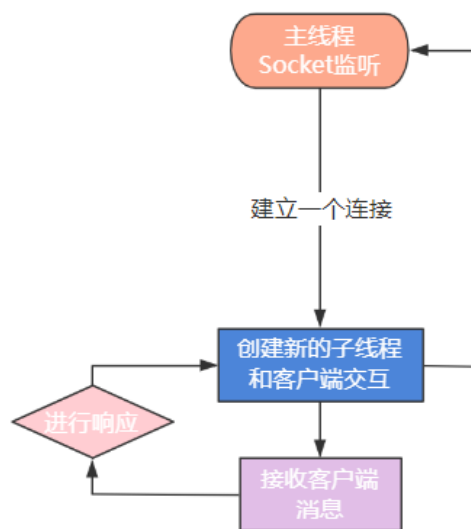


图 3 服务器总体流程

3 详细设计

项目设计主要分为客户端和服务端，一个是服务请求方，一个是服务响应方，二者功能不相同，但是在设计中逻辑上是连贯的。

3.1 客户端整体布局设计

客户端页面的整体布局为：Absolute Layout，采用了 4 个 JPanel 容器交替在页面上进行显示，同时采用了各类型文本框（包括普通文本框、密码框、格式化文本框等）以及按钮、面板、等组件。初始设定界面为登录界面，通过登录界面可以跳转到注册页面或者服务器交互页面，之后通过服务器交互页面可以进入会话页面。整个过程中有且仅有一个 JPanel 容器可以显示。

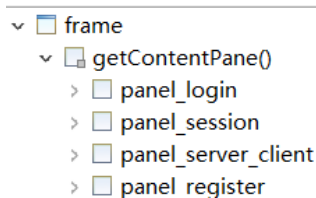


图 4 client 类的 GUI 图层次图

3.2 客户端登录页面设计

panel_login 是登录界面的容器，其中包括了提示字段、输入框、按钮等部分，一些组件增加了鼠标事件监听来进行交互。

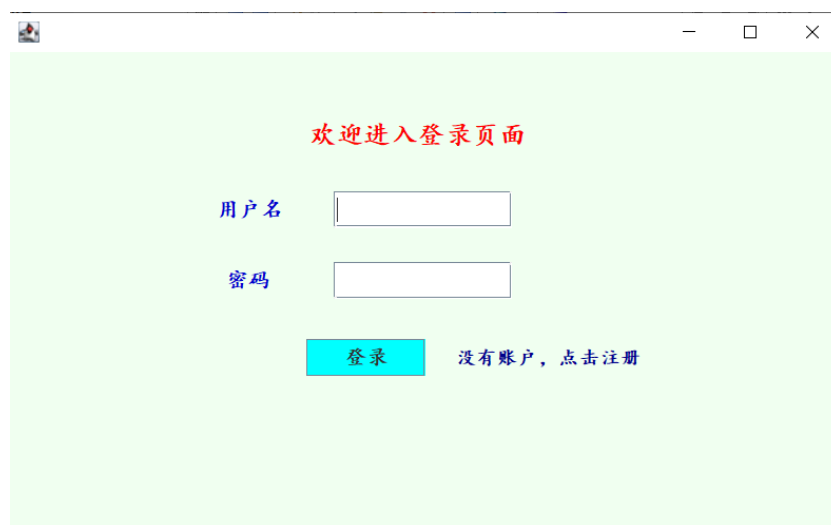


图 5 panel_login 容器的界面图

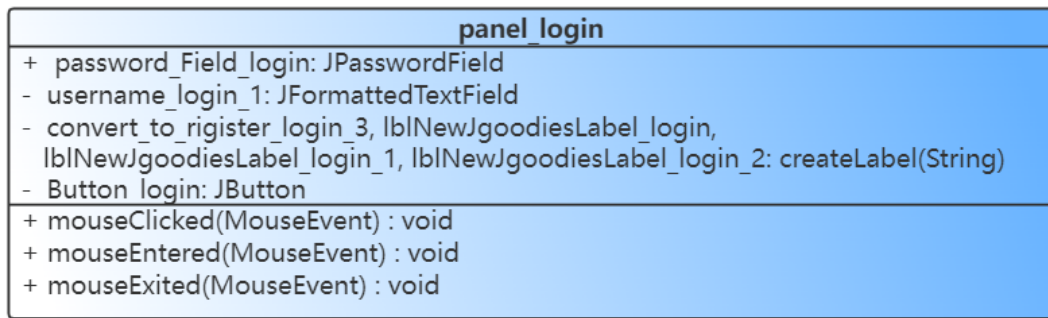


图 6 panel_login 容器的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

(1) 成员变量

- ① password_Field_login 是 JPasswordField 的实例, 可写, 用来填写密码, 密码在界面中的具体内容不可见。
- ② 图 5 的几个提示字符都是 createLabel (String) 的实例, “没有账户, 点击注册” 的部分含有跳转链接到注册页面。
- ③ username_login_1 是 JFormattedTextField 的实例, 可写, 用来输入用户名。
- ④ Button_login 是 JButton 的实例, 用于行使 “登录” 的功能。

(2) 方法

- ① mouseClicked(MouseEvent) 方法

成员变量 4 的按钮以及成员变量 2 的 “注册” 部分都有这个方法, 其中: 按钮部分会提交输入框中的数据到服务器, 子线程监听响应, 如果登录成功, 服务器会将这个客户端的 socket 与用户绑定, 用户界面会切换到服务器交互界面。注册部分点击之后, 应用会切换到注册页面, 通过 setVisible 方法对 Jpanel 容器进行控制。

- ② mouseEntered(MouseEvent) 和 mouseExited(MouseEvent) 方法

这两个方法的作用就是当鼠标移动和离开到某一组件上时, 组件发生特定的变化, 比如鼠标在登录按钮上, 登录按钮会变成橘色; 鼠标在 “没有账户, 点击注册” 的提示文本上时, 文本变成 “注册” 两个字。

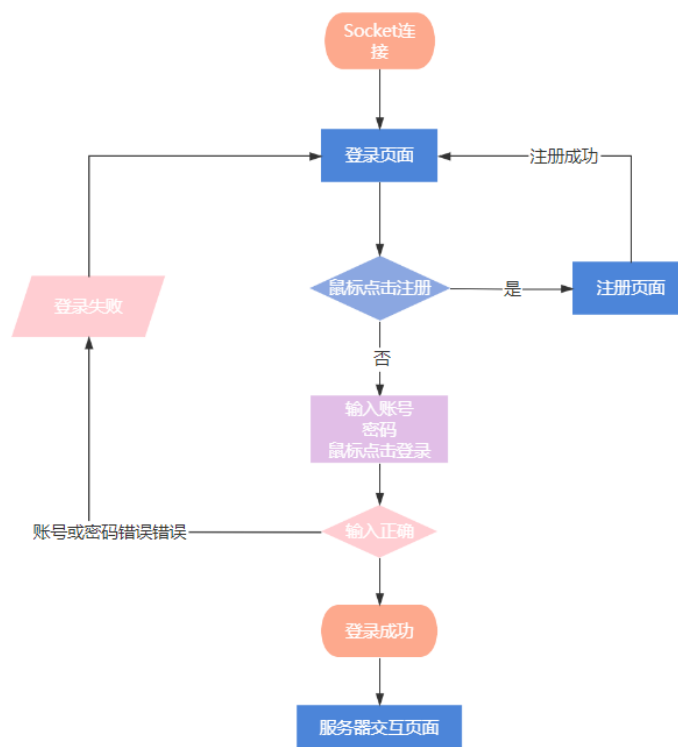


图 7 登录页面设计流程图

3.3 客户端注册页面设计

panel_register 是注册界面的容器，其中包括了提示字段、输入框、按钮等部分，按钮组件增加了鼠标事件监听来进行注册数据的提交。

注册页面

用户名

密码

确认密码

图 8 panel_register 容器的界面图

panel_register
+ passwordField_register, passwordField_register_confirm : JPasswordField - formattedTextField_register_1: JFormattedTextField - lblNewJgoodiesLabel_register, lblNewJgoodiesLabel_register_1, lblNewJgoodiesLabel_register_2, lblNewJgoodiesLabel_register_3: createLabel(String) - Button_register_Register: JButton
+ mouseClicked(MouseEvent) : void + mouseEntered(MouseEvent) : void + mouseExited(MouseEvent) : void

图 9 panel_register 容器的 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

- ① passwordField_register 和 passwordField_register_confirm 是 JPasswordField 的实例，可写，用来填写密码，密码在界面中的具体内容不可见，二者的内容一致才可以提交。
- ② 图 8 的几个提示字符都是 createLabel(String) 的实例，用来提示输入。
- ③ formattedTextField_register_1 是 JFormattedTextField 的实例，可写，用来输入用户名。
- ④ Button_register_Register 是 JButton 的实例，用于行使“注册”的功能，检查内容并且提交数据。

(2) 方法

① mouseClicked(MouseEvent) 方法

成员变量 4 的按钮有这个方法，其中：按钮部分会确保在两个密码输入框中的内容一致的情况下提交输入框中的数据到服务器，子线程监听响应，如果注册成功，服务器会将这个用户信息添加到列表中，用户界面会切换到登录界面。

② mouseEntered(MouseEvent) 和 mouseExited(MouseEvent) 方法

这两个方法的作用就是当鼠标移动和离开到某一组件上时，组件发生特定的变化，只是为了增加效果，没有其他作用，这里不做过多赘述。

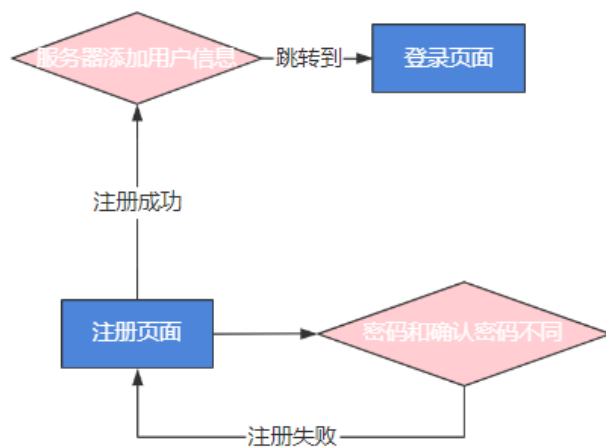


图 10 注册页面设计流程图

3. 4 客户端与服务器交互页面设计

panel_server_client 是客户端服务器交互界面的容器，其中包括了提示字段、输入框、按钮等部分，按钮组件增加了鼠标事件监听来进行注册数据的提交。



图 11 panel_server_client 容器的界面图

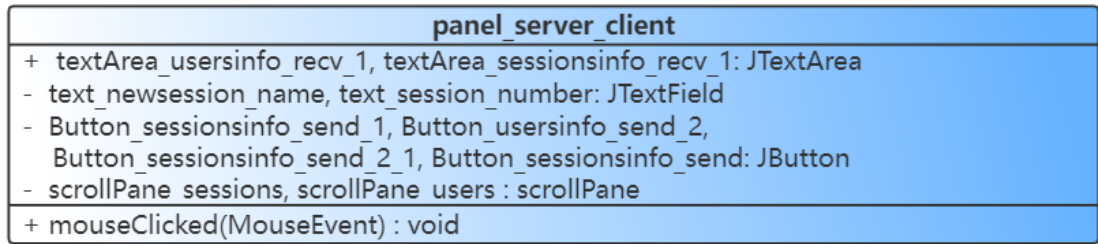


图 12 panel_server_client 容器的 UML 图

以下是 UML 图中有关数据和方法的详细说明：

(1) 成员变量

① 2 个 JTextArea 的实例分别被一个 scrollPane 实例环绕，从而可以在需要的时候使用滑轮查看在区域范围之外的信息。

② 4 个 JButton 的实例，各自的功能都不相同，具体的功能再方法中介绍。

其他成员变量与前文组件类似，只是简单的输入文本，这里不做介绍。

(2) 方法

① mouseClicked(MouseEvent) 方法

4 个按钮都有这个方法，其中：“查询所有会话”按钮点击之后向服务器发送查询所有会话的请求，服务器返回所有的会话信息，首行为标题，内容为会话的唯一标识 ID 和会话的名称，服务器返回的信息经过处理显示在左上方的文本框中。“查询所有用户”与前类似，返回所有用户信息处理后显示在左下方的文本框中。“创建新会话”按钮点击之后，按照最左边可写文本框的内容作为会话的名称建立新的会话。“进入该会话”按钮根据会话 ID（只能是数字，否则会提示）向服务器请求加入会话，成功就跳转到会话页面，否则提示错误。

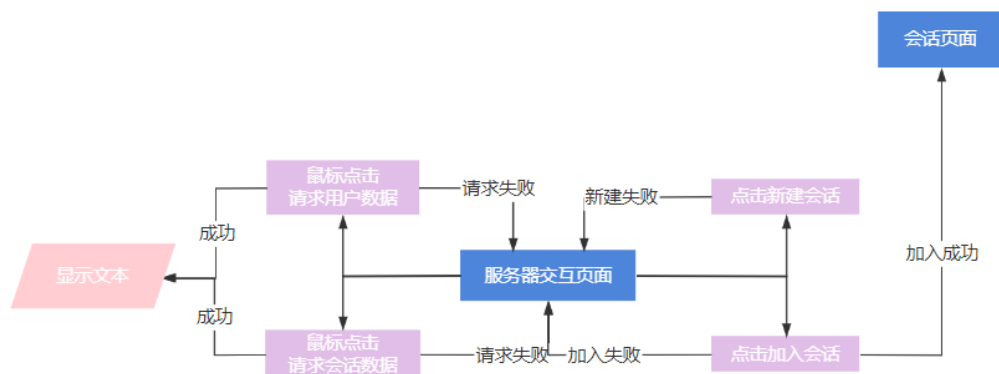


图 13 客户端与服务器交互页面设计流程图

3.5 客户端会话页面的设计

panel_session 是会话界面的容器，其中包括了提示字段、输入框、按钮等部分，按钮组件增加了鼠标事件监听来进行注册数据的提交。

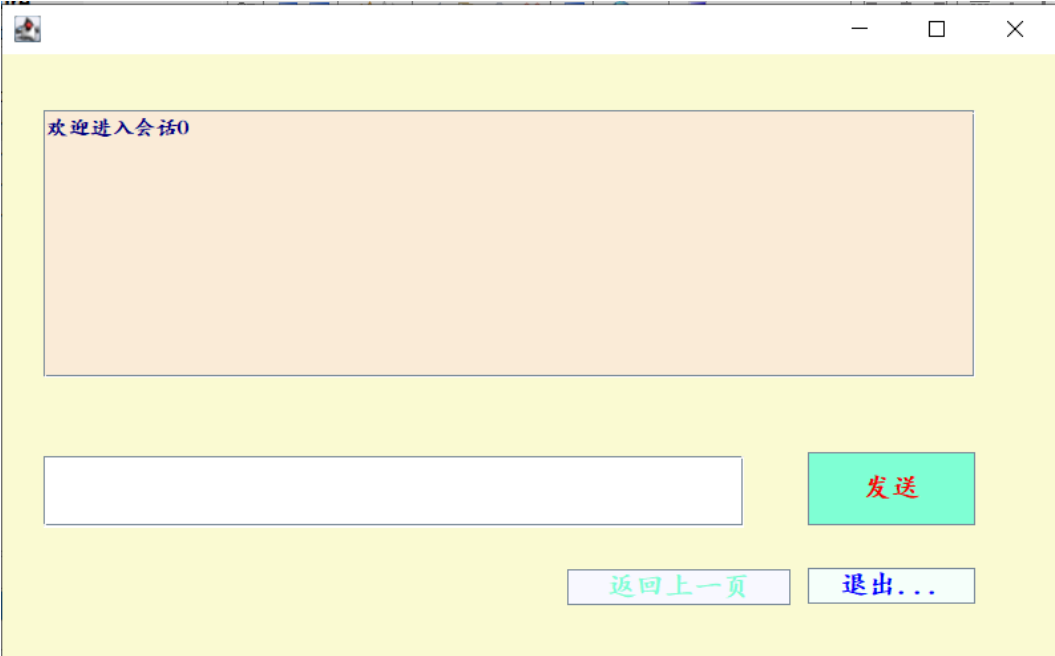


图 14 panel_session 容器的界面图



图 15 panel_session 容器的 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

成员变量与前文组件类似含义，文本区域和按钮区域，这里不做赘述。

(2) 方法

① mouseClicked(MouseEvent) 方法

3 个按钮都有这个方法，其中：“发送”按钮点击之后向服务器发送白色输入框中的信息并且自己打印信息到框中，服务器将信息发送给同一个 session 中的用户。“返回上一页”页面跳转到与服务器交互页面，会话页面不关闭，持续

接收消息。“退出”按钮点击之后，将向服务器请求清除该用户在会话中的数据，得到肯定答复后回到与服务器交互页面。

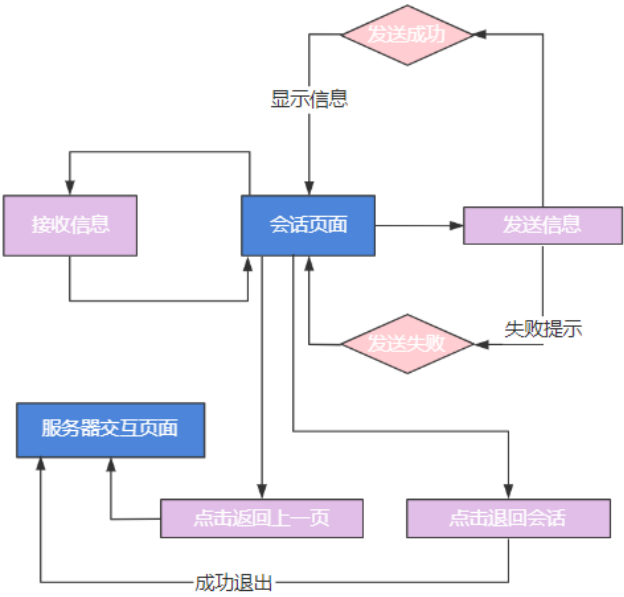


图 16 客户端与服务器交互页面设计流程图

3. 6 客户端其他设计

(1) 客户端 client 类简述

客户端类的内容大部分以 GUI 的方式进行设计和逻辑梳理，内部包含大量的组件，有些组件需要更大的可用范围所以作为了成员变量而不是方法的局部变量，在类创建的时候先调用第一个初始化函数 initialize1 去建立 Socket 连接，并且保存自己的 socket 值；之后立即初始化图形界面。

(2) 客户端子线程简述

在图形化界面初始化结束之后，会建立一个子线程用于接收服务器传来的消息并且处理，这些消息都根据特殊的文本来标识，与页面中鼠标的操作紧密相关，协作控制页面的变化。



图 17 内部类 recvmsg 的 UML 图

其中,在线程创建的时候将 Socket 变量传入,之后在 run 中循环接收 Socket 通信的值并且调用 process 函数进行针对性处理,使得客户端界面可以正常的运转。

(3) 使用 `JOptionPane.showMessageDialog` 调用消息框作为消息提示。

3.7 服务器设计

服务器只有一台,所以使用了很多静态变量,方便全局访问以及方便在 main 函数中调用。服务器开启之后循环监听端口,如果建立连接就建立一个子线程去与连接的客户端进行应答,主线程一直进行侦听。

由于服务器本身结构很简单,主要是由静态变量以及 main 方法组成,处理各个客户端的部分都在子线程中完成

(1) 成员变量

```
server
  lock_sendtoclients : Object
  serverSocket : ServerSocket
  server1 : server
  id_user : int
  session_id : int
  socket : Socket
  users : ArrayList<user>
  socketList : ArrayList<Socket>
  sessionList : ArrayList<session>
  {...}
  server()
  main(String[]) : void
```

图 18 server 类

- ① `lock_sendtoclients` 为线程锁;
- ② `serverSocket` 为服务器的唯一 Socket 值;
- ③ `server1` 为服务器引用;
- ④ `id_user, session_id` 用来记录用户和会话的数目,并且作为唯一标识 ID 赋值给用户和会话;
- ⑤ `socket` 用来暂时存储连接的客户端的 Socket 值;
- ⑥ `users, socketList, sessionList` 使用 ArrayList 记录所有用户、Socket、以及会话的信息。

(2) 子线程调用类 each_client 分析

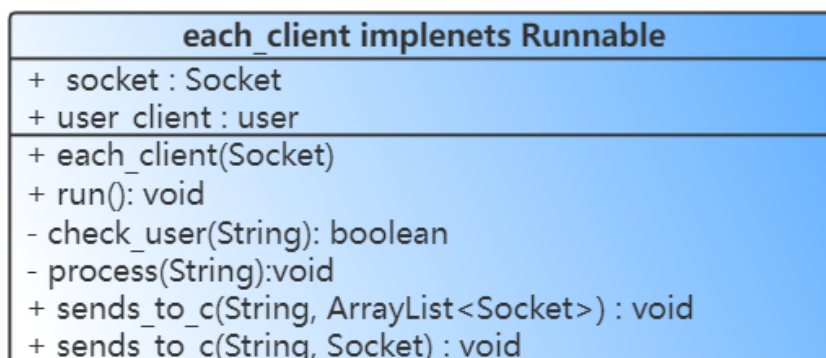


图 19 each_client 类的 UML 图

变量 socket 在类实例化的时候进行赋值，user_client 在该客户端登录成功后赋值，并且进行绑定，知道客户端退出。

方法 check_user 即为客户端身份验证的函数，函数值为 true 的时候 user_client 即绑定成功。process 方法对特定的输入进行针对性的响应，通过 sends_to_c 发送到制定客户端的 Socket 中。

3. 8 服务器相关类的设计

与服务器相关的类有 user 类以及 session 类，分别表示用户和会话类，二者设计上都非常简单，这里做简单介绍。

(1) user 类

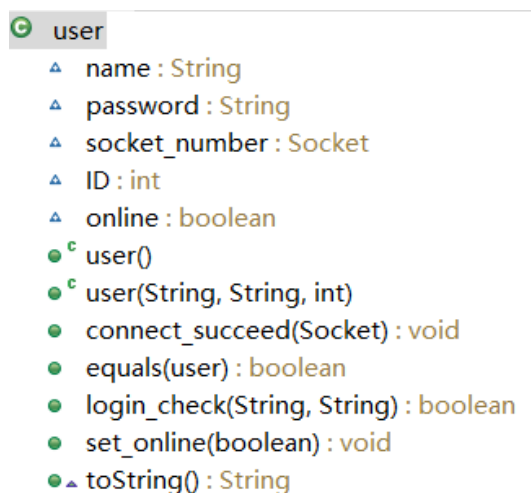
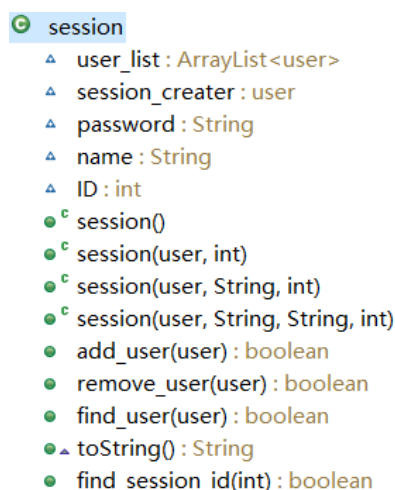


图 20 user 类

user 类在设计上定义了 name, password, socket_number, ID, online 5 个成员变量，其中含义都比较清晰。ID 为用户的唯一标识，即主

键，online 表示该用户是否登录，在身份验证的时候进行判断和设置。Socket 的值在每次验证之后进行判定，通过 connect_succeed 绑定此次登录的 Socket 值。user 用户改写了 equals 方法为比较 ID 值，从而确定是否为同一个用户；同时，user 用户还改写了 toString 方法，使得其打印用户的名字、唯一 ID 以及 online 状态。

(2) session 类

The image shows the class view of the 'session' class in an IDE. It lists several attributes with blue triangle icons: 'user_list' of type 'ArrayList<user>', 'session_creator' of type 'user', 'password' of type 'String', 'name' of type 'String', and 'ID' of type 'int'. Below the attributes, it lists methods with green circle icons: 'session()', 'session(user, int)', 'session(user, String, int)', 'session(user, String, String, int)', 'add_user(user) : boolean', 'remove_user(user) : boolean', 'find_user(user) : boolean', 'toString() : String', and 'find_session_id(int) : boolean'.

```
session
  ▲ user_list : ArrayList<user>
  ▲ session_creator : user
  ▲ password : String
  ▲ name : String
  ▲ ID : int
  ● session()
  ● session(user, int)
  ● session(user, String, int)
  ● session(user, String, String, int)
  ● add_user(user) : boolean
  ● remove_user(user) : boolean
  ● find_user(user) : boolean
  ● toString() : String
  ● find_session_id(int) : boolean
```

图 21 session 类

①成员变量

ID 为一个 session 的唯一标识，user_list 记录所有参与到这个 session 的用户，password 为可以选择设置的会话的密码（本工程没有使用），name 为会话名称（类比 QQ 群名），session_creator 即为会话的创建者。

②方法

add_user 方法添加用户到会话中；remove_user 移除会话中的某个用户；find_user 找到会话中的某个用户是否存在；该类也重写的 toString 方法为输出会话的 ID 和会话名。

通过上述类以及相关分析，项目整体的思路已经比较清晰了，具体理解还需要参考实际的代码以及相应的逻辑。

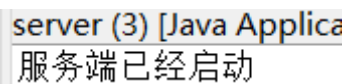
4 测试与运行

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，最后工程可以很好地运行，基本功能与简单的查询工具没有太大差别，没有出现明显的错误和漏洞，在数据库交互以及功能完备性上还可以加强，并且如果使用更多 GUI 组件或者自定义的 GUI 组件可能会更加美观。总体上本次设计在功能上已经基本达到要求，但是还有优化空间。

4.2 程序运行

程序运行分为 2 个部分，先启动服务器（在控制台进行输出），在启动客户端（GUI 图形界面操作）。



server (3) [Java Applica
服务端已经启动

图 22 服务端启动



图 22 客户端启动

点击注册新建账户。注册用我的名字为用户名，123 位密码。



图 23 客户端注册



图 24 客户端注册成功

使用刚注册的客户端重新登录。



图 25 客户端登录

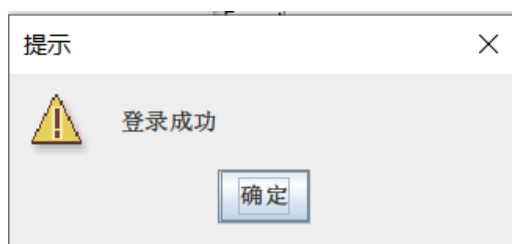


图 26 客户端登录成功

点击查询所有会话和查询所有用户进行查询以及显示。



图 27 查询会话和用户信息

发现没有会话，但有系统默认的两个用户以及自己，自己的状态是 on（在线）。之后，点击创建新会话，命名为 Java 学习群。

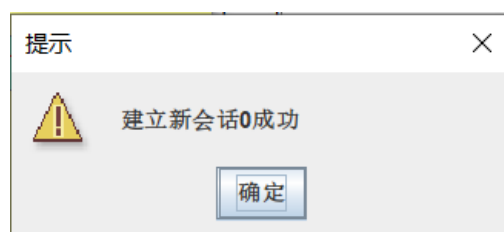


图 28 新会话建立成功

点击查询所有会话进行更新。



图 29 重新查询所有会话

填写会话 ID 为 0 进入会话。

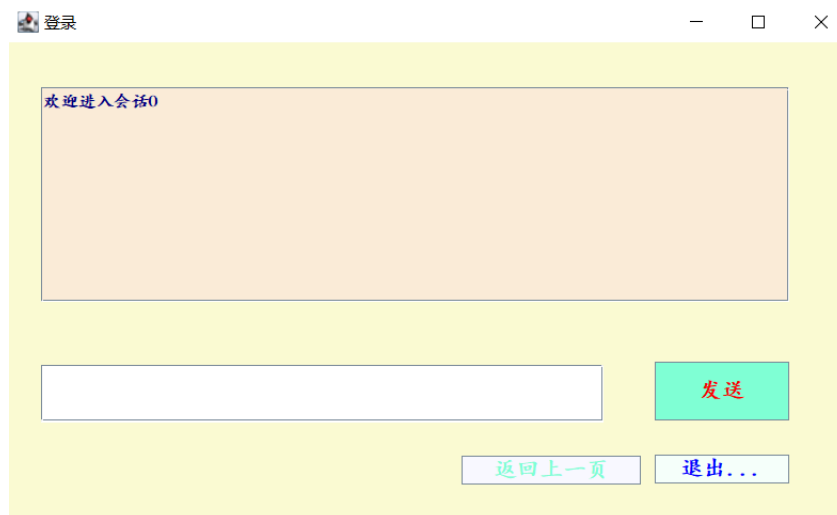


图 30 进入会话 0

再打开两个客户端，使用 aaa 和 bbb 登录，并且都进入会话 0，进行聊天

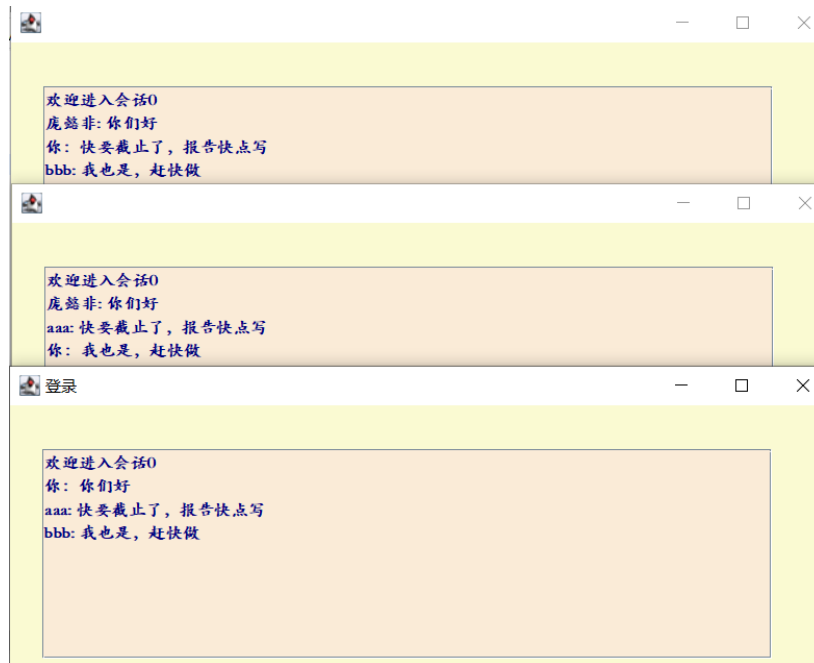


图 31 3 个用户在会话 0 聊天

将第一个打开的客户端返回上一下, 新增加一个 OS 的会话, 并且和用户 aaa 都进入这个会话。



图 32 新建另一个会话

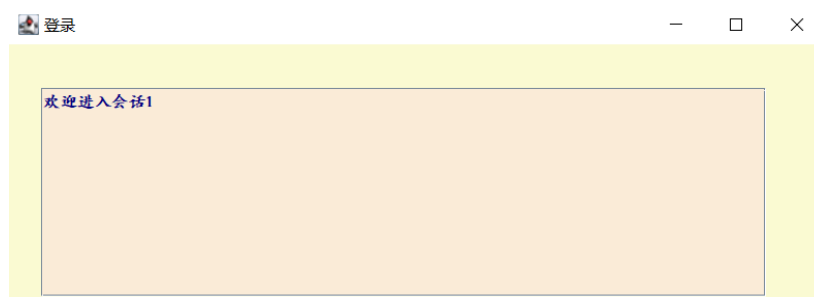


图 32 庞懿非和 aaa 进入会话 1

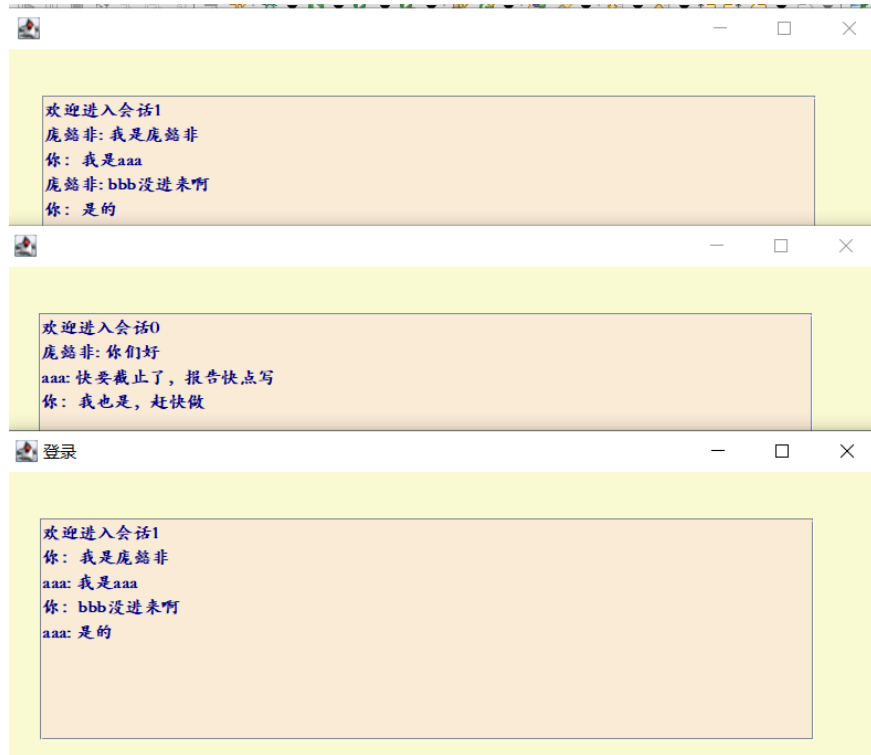


图 33 aaa 和庞懿非在会话 1 聊天，此时 bbb 还在会话 0

可以看到，通过进入不同的会话可以实现多会话聊天。（bbb 没有进入会话 1 所以没有接收到会话 1 的内容）

```

客户庞懿非已经退出连接
现在还剩下2个客户在线
Socket[addr=/127.0.0.1,port=62803,localport=9999]
客户bbb已经退出连接
现在还剩下1个客户在线
Socket[addr=/127.0.0.1,port=62802,localport=9999]
客户aaa已经退出连接
现在还剩下0个客户在线

```

图 34 客户端关闭，服务器控制台输出

当客户退出，服务器打印相应信息（如上图所示）。

5. 总结

多用户多会话聊天软件整体来看，逻辑上是比较清晰的。通过客户端发起请求和服务器相应请求，只需要定义好相应的请求方式就可以保证二者之间的信息互通。

当然，在具体实现中还是有很大的难度。难点主要为：如何设计客户端和服务端之间的交互逻辑，比如发送什么类型的信息表示登录，以及什么样的返回信息表示登录成功等等，所有的操作中发送和接收内容都需要做具体的规定。另一个难点则是 **GUI** 的图形界面设计。由于缺乏经验，在项目设计的时候出现了很多的问题，比如文字超出界面，预览图 and 实际效果不同，页面的跳转和切换，界面过于僵硬等问题，都需要花费很大的时间去一点点的解决。在设计的时候，我只解决了其中的一小部分问题，页面切换使用 **setVisible** 函数对多个 **Jpanel** 进行设置，同时使用 **ScrollPane** 对文本内容进行滑动显示，实际上在设计的时候还使用了 **JTable** 以及 **JList** 等组件，但都因为难以做出满意美观的效果而放弃，最后整体上我使用的组件都相对简单，努力使用简单的组件绘制轻巧的图形界面。

在设计初期我的设想是设计一个类似 **QQ** 的 **GUI** 界面，并且功能也要累死，包括实时显示群内用户信息（实时更新），通过 **tab** 选择切换新的窗口（即进入另一个群进行聊天），以及使用 **Mysql** 对用户、会话信息进行记录，可以的话对文本信息进行记录。不过实现起来我花费了较多时间研究组件而且也没有取得很好的结果，导致数据库连接的这一部分我也只能放弃，但是我觉得实现起来并不是很难，只是第一次做有点生疏，需要花费更多的时间了。

从最后的结果来看，这个项目的结果我还是满意的，他也可以很好地模拟早期聊天软件的功能，对于我自己来说是一个巨大的进步。