Java面向对象程序设计课程设计实验报告七

实验人：王维

学号：320210942931

班级：2021级计算机科学与技术二班

一：补全程序题

1. 11.3.1（2）

* 源代码：

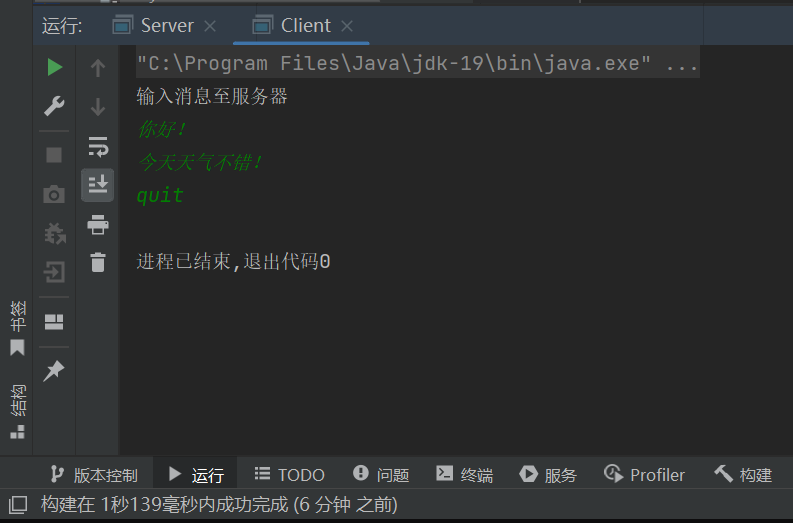
（1）Client.java

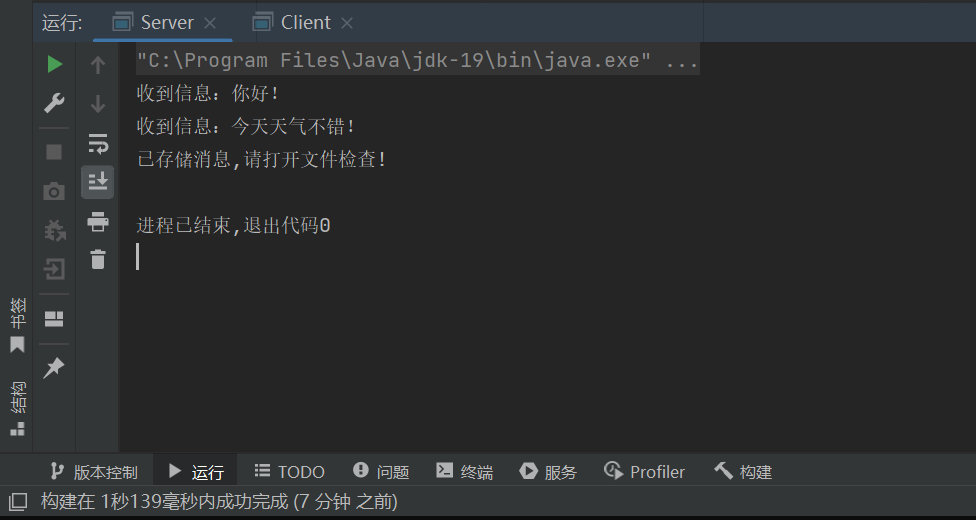
|  |
| --- |
| package ExperimentSourceCode\_7**.**fill**;**/\*client.java\*/  **import** java**.**net**.\*;**  **import** java**.**io**.\*;**  public class Client **{**  public static void main**(**String args**[])** **throws** Exception **{**  Socket soc**=new** Socket**(**"127.0.0.1"**,**4001**);** //将target mechine替换为目标机器的ip地址或主机名  BufferedReader br**=new** BufferedReader**(new** InputStreamReader**(**System**.**in**));**  System**.**out**.**println**(**"输入消息至服务器"**);**  String message**=**""**;**  String temp**;**  PrintStream ps**=new** PrintStream**(**soc**.**getOutputStream**());**  **while(!((**temp**=**br**.**readLine**()).**equals**(**"quit"**))){**  ps**.**println**(**temp**);**  **}**  ps**.**close**();**  soc**.**close**();**  **}**  **}** |

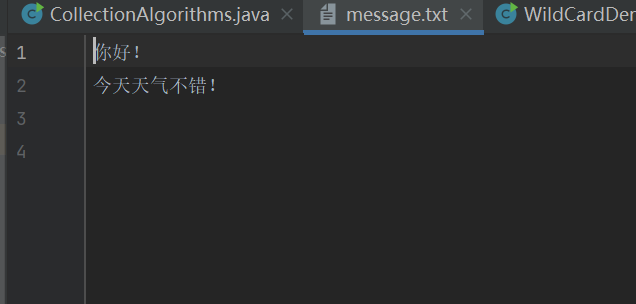
（2） Server.java

|  |
| --- |
| package ExperimentSourceCode\_7**.**fill**;**  **import** java**.**net**.\*;**  **import** java**.**io**.\*;**  public class Server **{**  public static void main**(**String args**[])** **throws** Exception**{**  ServerSocket ss**=new** ServerSocket**(**4001**);**  Socket soc**=**ss**.**accept**();**  BufferedReader br**=new** BufferedReader**(**  **new** InputStreamReader**(**soc**.**getInputStream**()));**  String message**=**""**;**  String temp**;**  **do{**  temp**=**br**.**readLine**();**  **if(**temp**==null)** **break;**  System**.**out**.**println**(**"收到信息："**+**temp**);**  message**=**message**+**temp**+**"\n"**;**  **}while(true);**  br**.**close**();**  PrintStream ps**=new** PrintStream**(new** FileOutputStream**(**"src/ExperimentSourceCode\_7/Data/message.txt"**));**  ps**.**println**(**message**);**  ps**.**close**();**  br**.**close**();**  soc**.**close**();**  System**.**out**.**println**(**"已存储消息,请打开文件检查!"**);**  **}**  **}** |

* 运行结果：







1. 11.3.2（2）

* 源代码：

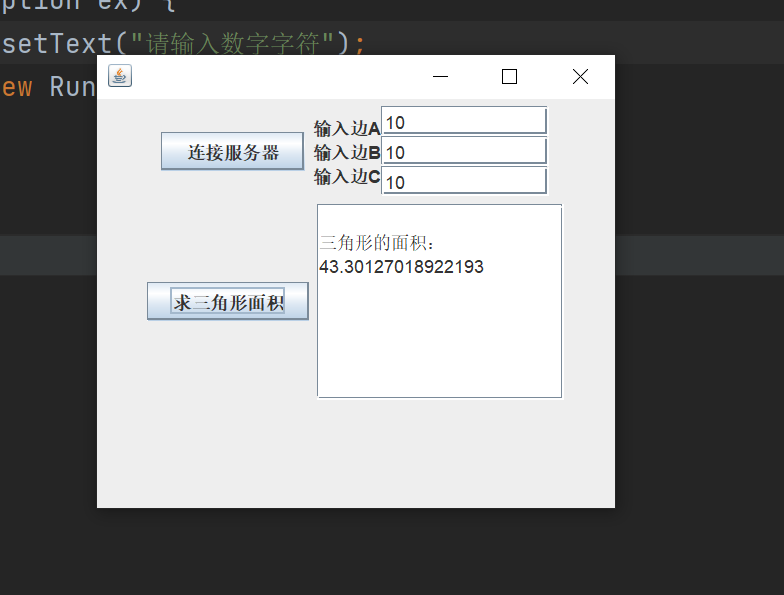
（1）JsClient.java

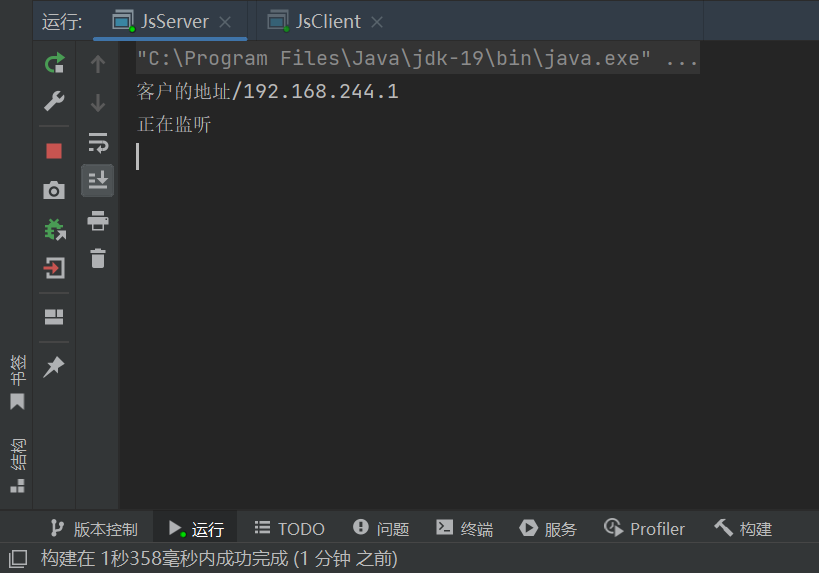
|  |
| --- |
| package ExperimentSourceCode\_7**.**fill**;**//JsClient.java  **import** java**.**net**.\*;**  **import** java**.**io**.\*;**  **import** java**.**awt**.\*;**  **import** java**.**awt**.**event**.\*;**  **import** javax**.**swing**.\*;**  class JsClient **extends** JFrame **implements** Runnable**,**ActionListener**{**  JButton connection**,**jsbutton**;**  JTextField inputA**,**inputB**,**inputC**;**  JTextArea showResult**;**  Socket socket**;**  DataInputStream in**=null;**  DataOutputStream out**=null;**  Thread thread**;**  public JsClient**(){**  socket**=new** Socket**();**  connection**=new** JButton**(**"连接服务器"**);**  jsbutton**=new** JButton**(**"求三角形面积"**);**  inputA**=new** JTextField**(**"0"**,**12**);**  inputB**=new** JTextField**(**"0"**,**12**);**  inputC**=new** JTextField**(**"0"**,**12**);**  Box boxV1**=**Box**.**createVerticalBox**();**  boxV1**.**add**(new** JLabel**(**"输入边A"**));**  boxV1**.**add**(new** JLabel**(**"输入边B"**));**  boxV1**.**add**(new** JLabel**(**"输入边C"**));**  Box boxV2**=**Box**.**createVerticalBox**();**  boxV2**.**add**(**inputA**);**  boxV2**.**add**(**inputB**);**  boxV2**.**add**(**inputC**);**  Box baseBox**=**Box**.**createHorizontalBox**();**  baseBox**.**add**(**boxV1**);**  baseBox**.**add**(**boxV2**);**  Container con**=**getContentPane**();**  con**.**setLayout**(new** FlowLayout**());**  showResult**=new** JTextArea**(**8**,**18**);**  con**.**add**(**connection**);**  con**.**add**(**baseBox**);**  con**.**add**(**jsbutton**);**  con**.**add**(new** JScrollPane**(**showResult**));**  jsbutton**.**addActionListener**(this);**  connection**.**addActionListener**(this);**  thread**=new** Thread**(this);**  setBounds**(**100**,**100**,**360**,**310**);**  setVisible**(true);**  setDefaultCloseOperation**(**JFrame**.**EXIT\_ON\_CLOSE**);**  **}**  public void run**(){**  **while(true){**  **try{**  double area**=**in**.**readDouble**();**  showResult**.**append**(**"\n三角形的面积：\n"**+**area**);**  showResult**.**setCaretPosition**((**showResult**.**getText**()).**length**());**  **}**  **catch(**IOException e**){**  showResult**.**setText**(**"与服务器已断开"**);**  jsbutton**.**setEnabled**(false);**  **break;**  **}**  **}**  **}**  public void actionPerformed**(**ActionEvent e**){**  **if(**e**.**getSource**()==**connection**){**  **try{**  **if(**socket**.**isConnected**()){}**  **else{**  InetAddress address**=**InetAddress**.**getByName**(**"127.0.0.1"**);**  InetSocketAddress socketAddress**=new** InetSocketAddress**(**address**.**getLocalHost**(),**4444**);**  socket**.**connect**(**socketAddress**);**  in**=new** DataInputStream**(**socket**.**getInputStream**());**  out**=new** DataOutputStream**(**socket**.**getOutputStream**());**  jsbutton**.**setEnabled**(true);**  thread**.**start**();**  **}**  **}**  **catch(**IOException ee**){}**  **}**  **if(**e**.**getSource**()==**jsbutton**){**  **try{**  double a**=**Double**.**parseDouble**(**inputA**.**getText**()),**  b**=**Double**.**parseDouble**(**inputB**.**getText**()),**  c**=**Double**.**parseDouble**(**inputC**.**getText**());**  **if(**a**+**b**>**c**&&**a**+**c**>**b**&&**b**+**c**>**a**){**  out**.**writeDouble**(**a**);**  out**.**writeDouble**(**b**);**  out**.**writeDouble**(**c**);**  **}else{**  inputA**.**setText**(**"你输入的三个数不构成三角形"**);**  **}**  **}**  **catch(**Exception ee**){**  inputA**.**setText**(**"请输入数字字符"**);**  **}**  **}**  **}**  public static void main**(**String args**[]){**  JsClient win**=new** JsClient**();**  **}**  **}** |

（2）JsServer.java

|  |
| --- |
| package ExperimentSourceCode\_7**.**fill**;**//JsServer.java  **import** java**.**io**.\*;**  **import** java**.**net**.\*;**  public class JsServer**{**  public static void main**(**String args**[]){**  ServerSocket server**=null;**  ServerThread thread**;**  Socket client**=null;**  **while(true){**  **try{**  server**=new** ServerSocket**(**4444**);**  **}**  **catch(**IOException e1**){**  System**.**out**.**println**(**"正在监听"**);**  **}**  **try{**  client**=**server**.**accept**();**  System**.**out**.**println**(**"客户的地址"**+**client**.**getInetAddress**());**  **}**  **catch(**IOException e**){**  System**.**out**.**println**(**"正在等待客户"**);**  **}**  **if(**client**!=null){**  **new** ServerThread**(**client**).**start**();**  **}**  **else{**  **continue;**  **}**  **}**  **}**  **}**  class ServerThread **extends** Thread**{**  Socket socket**;**  DataOutputStream out**=null;**  DataInputStream in**=null;**  String s**=null;**  ServerThread**(**Socket t**){**  socket**=**t**;**  **try{**  in**=new** DataInputStream**(**socket**.**getInputStream**());**  out**=new** DataOutputStream**(**socket**.**getOutputStream**());**  **}**  **catch(**IOException e**){}**  **}**  public void run**(){**  **while(true){**  double a**=**0**,**b**=**0**,**c**=**0**,**area**=**0**;**  **try{**  a**=**in**.**readDouble**();**  b**=**in**.**readDouble**();**  c**=**in**.**readDouble**();**  double p**=(**a**+**b**+**c**)/**2.0**;**  area**=**Math**.**sqrt**(**p**\*(**p**-**a**)\*(**p**-**b**)\*(**p**-**c**));**  out**.**writeDouble**(**area**);**  **}**  **catch(**IOException e**){**  System**.**out**.**println**(**"客户离开"**);**  **break;**  **}**  **}**  **}**  **}** |

* 运行结果：





二．设计实验：设计一个基于TCP或UDP的聊天程序

1. 基本思路：设计一个聊天程序，传输数据协议使用TCP，采用最简单的

群聊方式，发送的消息所有链接的客户端均可见，运行模式如下简图：

LOCK

Client3

Client2

Client1

Server

MySQL

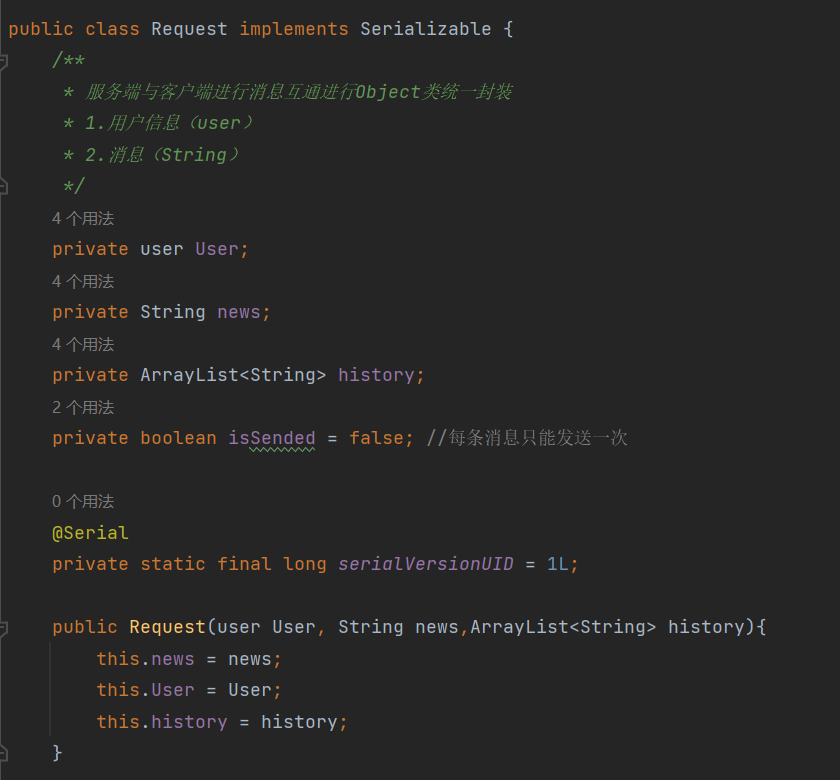
**————**:登陆验证，客户端和服务端交换一次数据，与MySQL进行数据交互时需要开启同步锁

**————**:服务端当消息池发生改变就发送消息，客户端一直监听接受消息

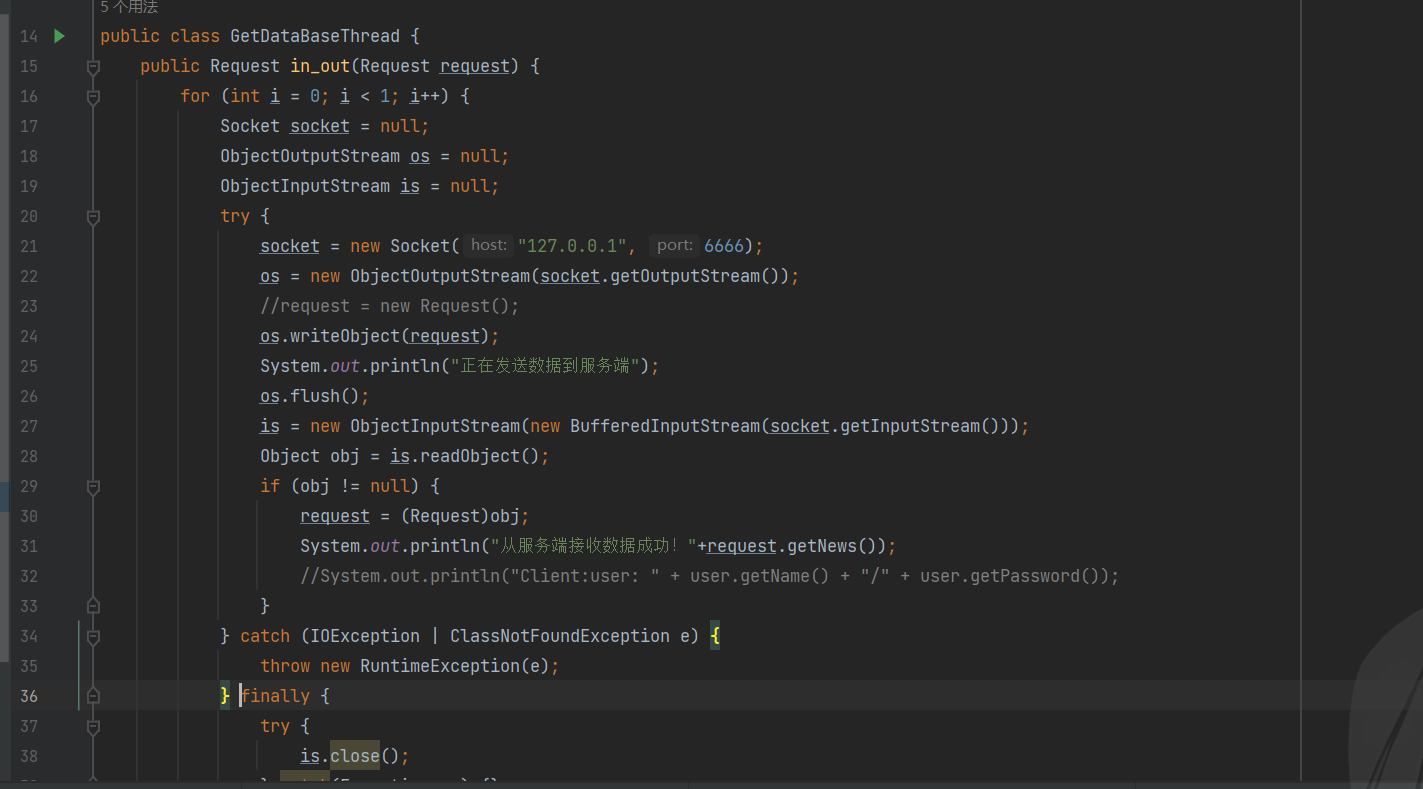
**————**：客户端发送数据，服务端一直监听接受数据

1. 各部分具体细节实现：
2. 登录：

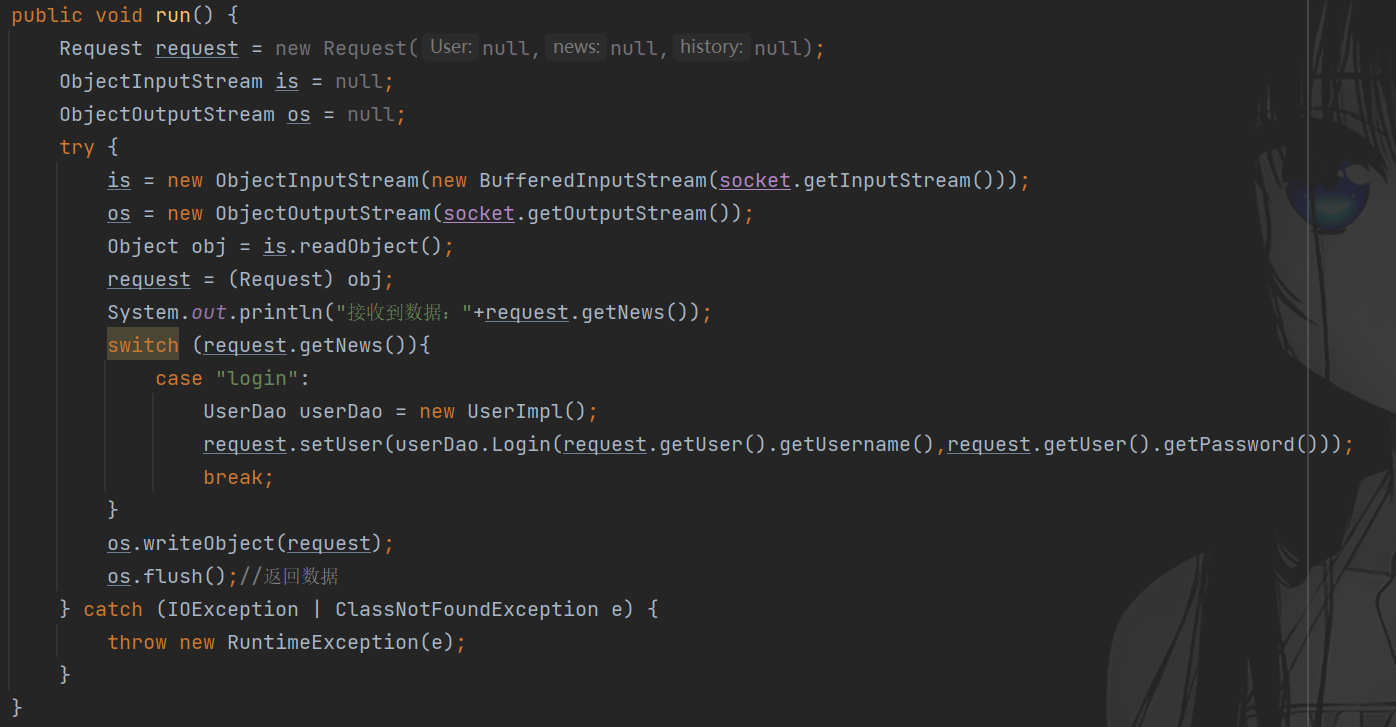
* TCP传输数据：登陆验证和聊天时采用不同的端口，采用序列化传输对象（Object）



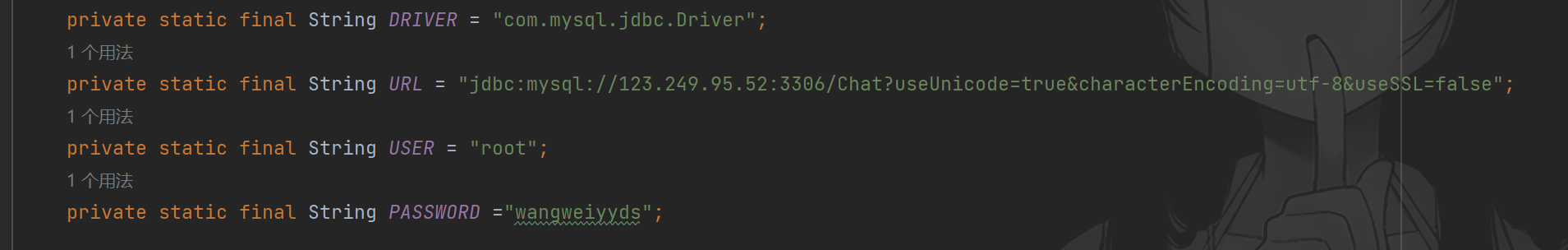
* 客户端发送数据：发送数据，用for循环实现多次迭代



* 服务端接受数据：每接受到一个服务端的链接请求，就开启一个线程进行，在switch循环中找到对应请求，对数据库进行操作，接受数据库返回的信息：

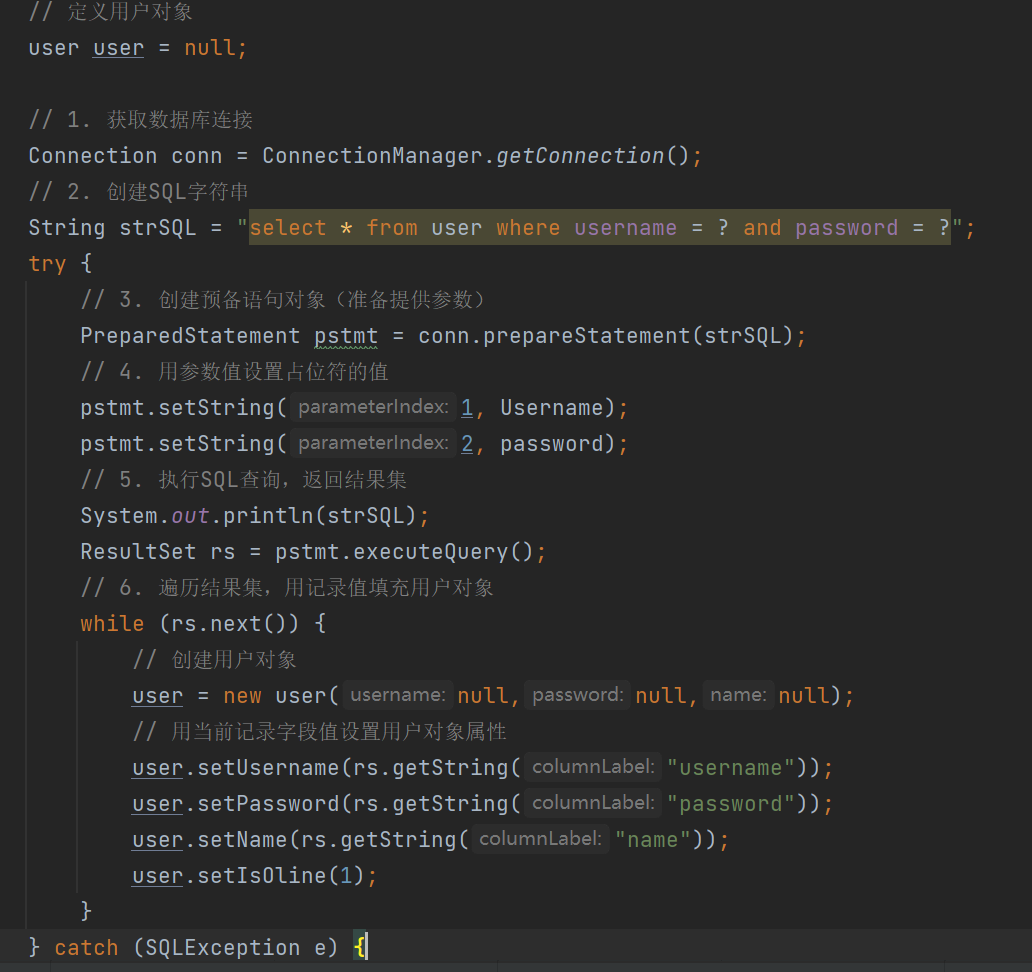


* 对数据库的操作：
* 获取数据库链接：

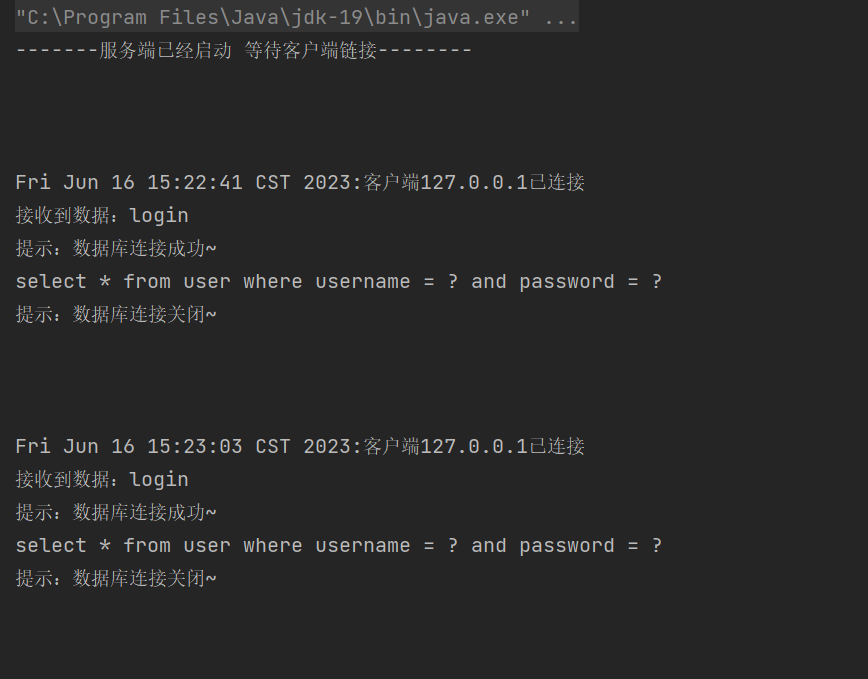




* 登录验证操作：在user表中寻找与用户名和密码匹配的对象，找到了就返回该uer对象，没有返回就返回null



* 登录演示：







1. 聊天：

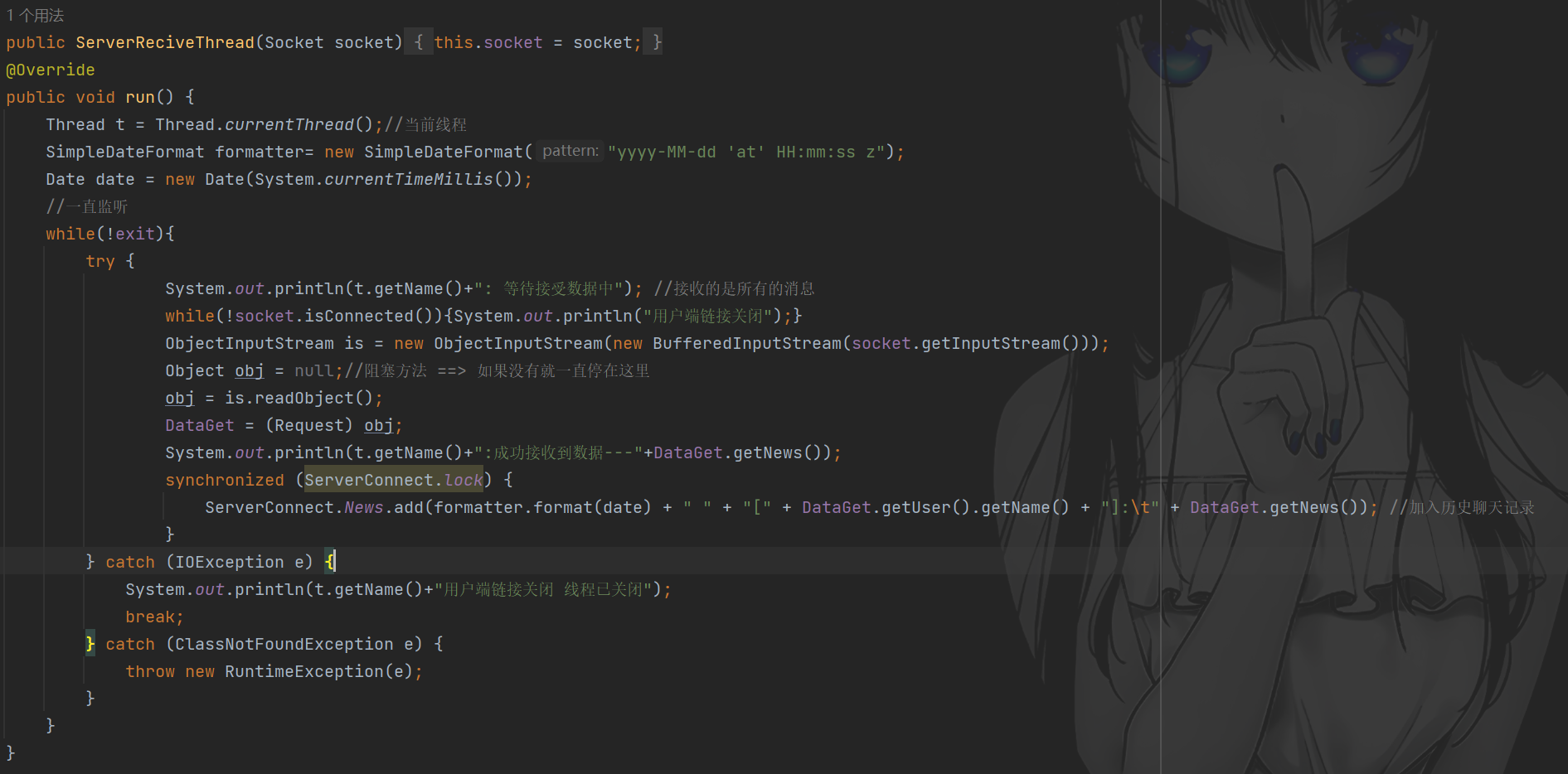
* 服务端：每收到一个链接请求，就开启一个线程池，开启线程又开两个线程，一个负责接受客户端数据，并通过调用静态变量ArrayList加入消息池，一个负责监听消息池，发生变化时发送到客户端
* 监听客户端链接，开启线程：



* 开启接收消息和发送消息两条线程：



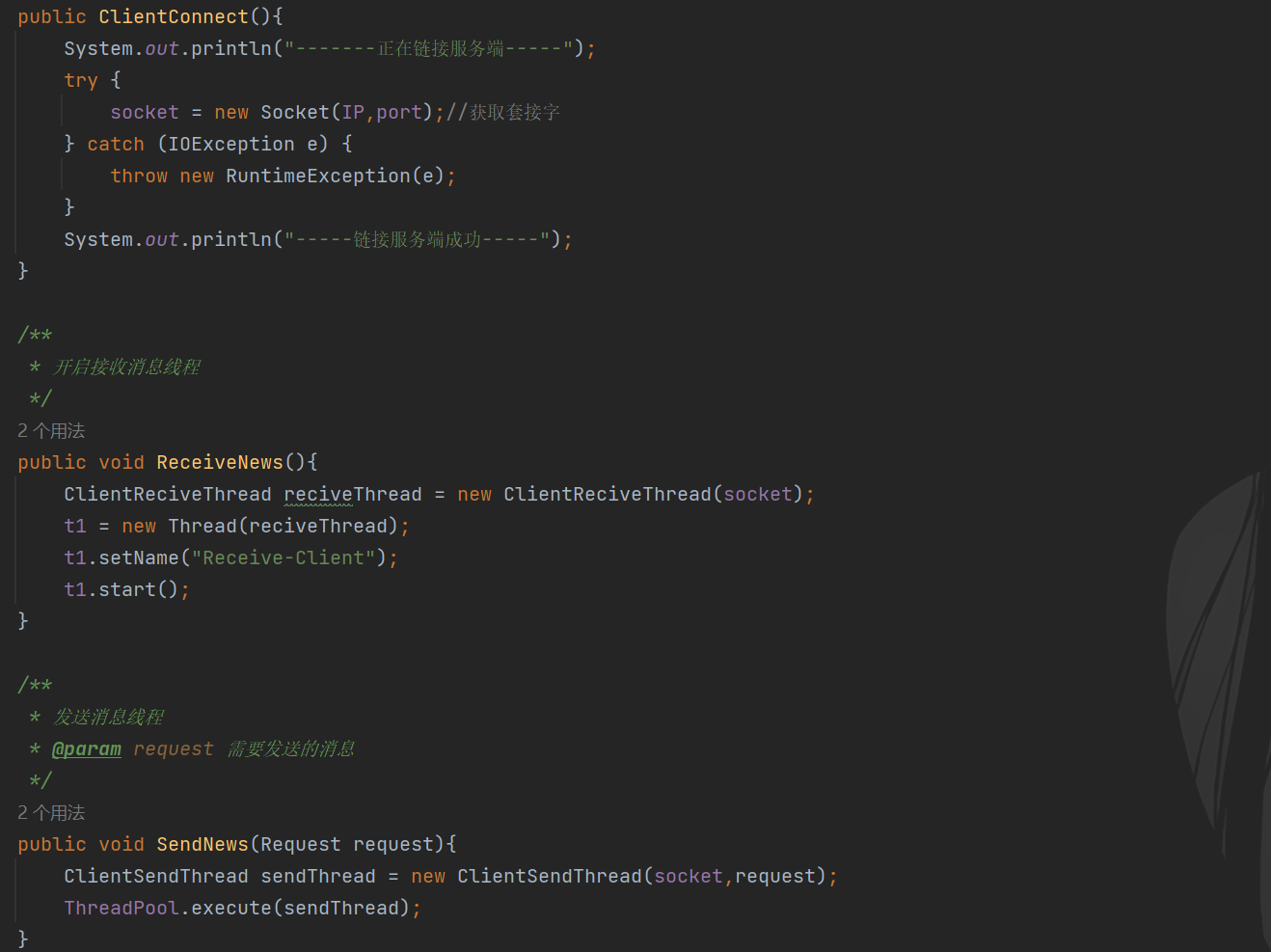
* 接受消息线程：接受数据，调用static属性的消息池，将结果写入（开启同步锁）



* 发送消息线程：发送数据，监听static属性的消息池，将结果写回客户端（开启同步锁）



* 客户端
* 链接服务端并开启接收消息、发送消息线程



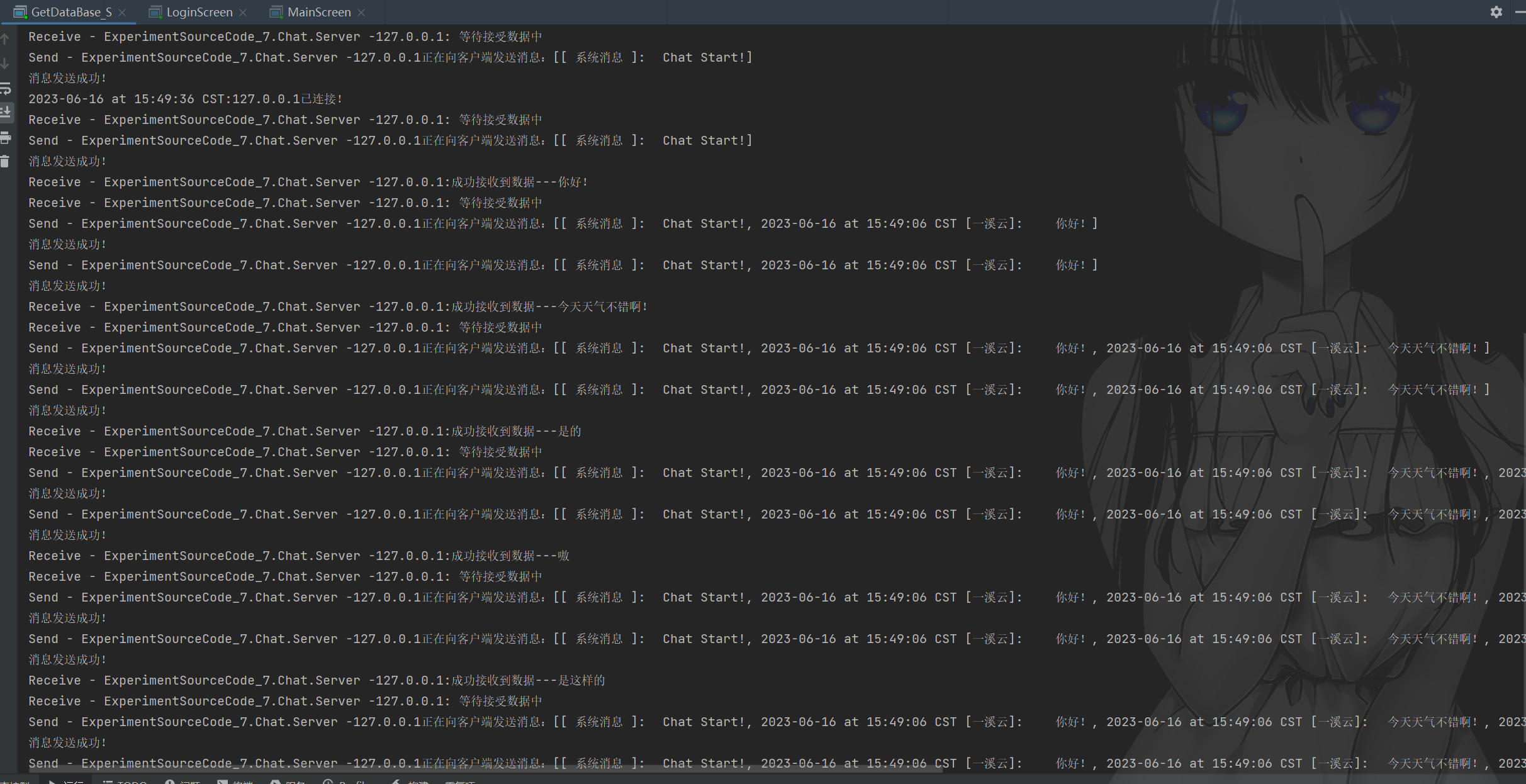
* 接受消息线程：调用static静态方法，线程接受到数据并回调

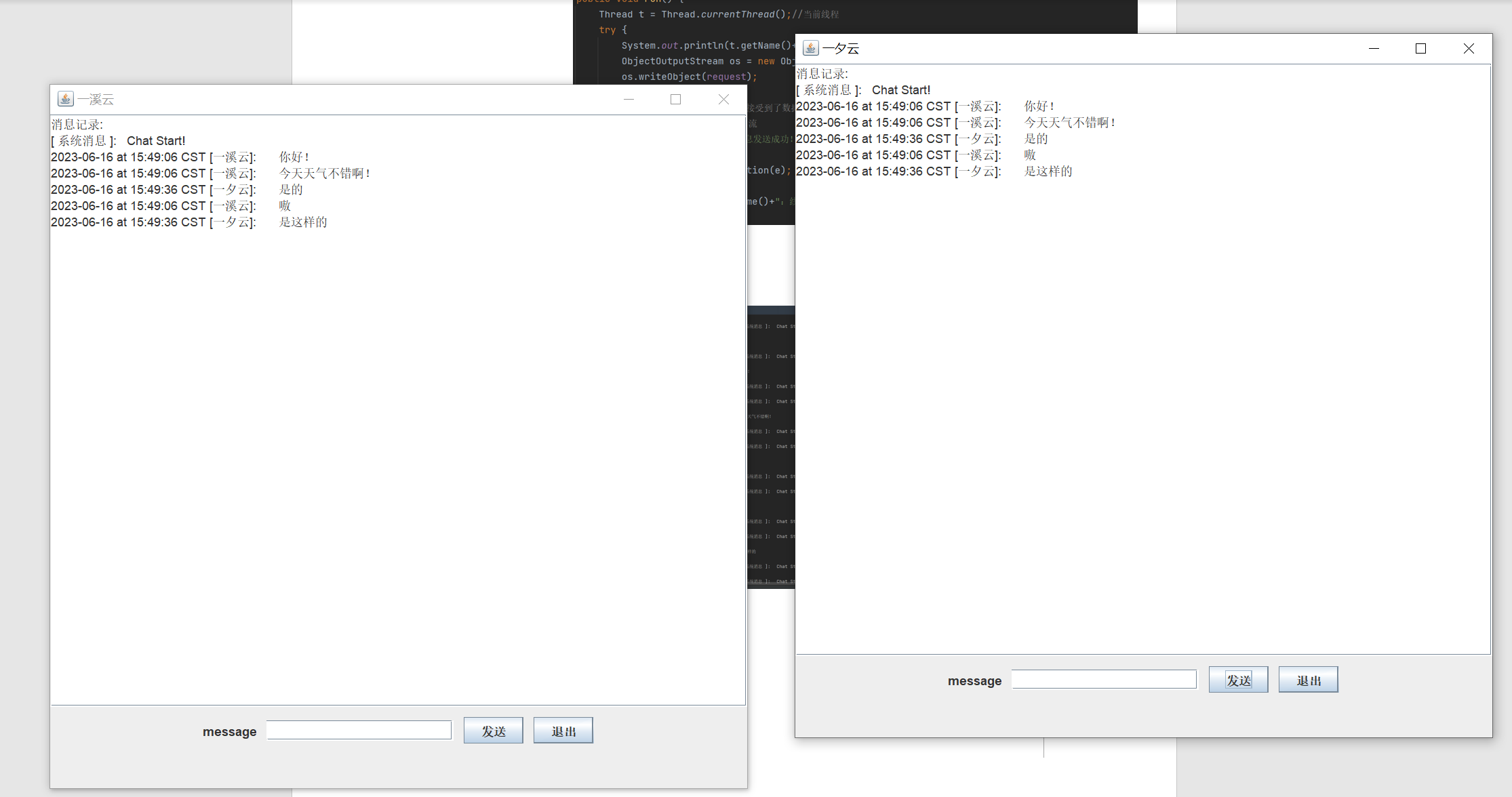


* 发送消息线程：



* 聊天系统演示：





1. 源代码：源代码过多，详见附件，已上传，以下是相关账号、密码的信息：

