即时通讯程序

这是本人大二下学期的课程设计，本博客简单的讲解各个部分代码。整个项目已打包上转到github，以下是链接：github网页

用到的技术：javaSwing，mysql8，以及简单的java多线程。

项目需求

实现一个Java版即时聊天程序。

【功能提示】

1、用户登录及登录验证：用户能够使用固定帐号（帐号程序内置即可，无需完成额外的注册功能）登录系统，系统能对预定的帐号、密码进行验证。

2、两两聊天功能。

3、建群及群聊功能，并且在下次登录时，群聊组依然存在。

4、共享白板：聊天时，可创建白板，在白板上共同绘制图形，可保存白板绘制的图片。

5、好友管理：能够显示好友列表，并能够添加、修改、删除好友。

6、在线、离线状态显示：能够显示好友的在线状态或离线状态。

7、聊天记录管理：能够以文件或数据库形式将聊天记录进行存储，并能打开、显示、删除所存储的聊天记录。

需求分析

系统总体设计

数据库的设计或文件结构的设计

本系统的数据库采用mysql。

只需要一个id作为主键，自动递增，然后一个user还有一个psw三列即可，具体如图：

聊天记录：一个count作为主键，自动递增然后一个sendorreceive列选择这条记录是发送的还是接受的，最终一个information列描述具体信息。同时，每个用户的聊天记录存在不同表中。具体如图：

群组关系：一个count作为主键，自动递增，然后一列是群名称，一列是群成员，所有群组关系存放在同一个表中。具体如图。

好友关系：和群组关系差不都，一个count作为主键自动递增，一个frendA列作为一个好友，一个frendB列作为另一个好友，如果两个名字出现在同一行，则证明两个人是好友。具体如图。

群组聊天记录：一个count作为主键自动递增，一列groupname存放该聊天记录发送的群，一列user存放发送人的名字，一列information存放发送的信息。如图。

项目结构

Client包

chatmanage内容

```
package Client;

import java.io.*;
import java.net.Socket;
import java.net.UnknownHostException;

public class ChatManage {
    private ChatManage(){}
    private static final ChatManage instance = new ChatManage();
    public static ChatManage getCM (){
        return instance;
    }
    ClientWindow window;
    Socket socket;
    /*传入ip、账号、密码*/
    String IP;
    String user;
    String password;
    /*定义输入输出流*/
    BufferedReader reader;
    PrintWriter writer;
    /*绑定window*/
    public void setWindow(ClientWindow window){
        this.window = window;
    }
    public void connect(String ip, String user , String password) {
        this.IP = ip;
        this.user = user;
        this.password = password;
        new Thread(){
            @Override
            public void run(){
                try{
                    /*接收ip地址传到socket中*/
                    socket = new Socket(IP,8848);
                    writer = new PrintWriter(
                            new OutputStreamWriter(
                                    socket.getOutputStream(),"UTF-8"
                            )
                    );
                    reader = new BufferedReader(
                            new InputStreamReader(
```

```
                                socket.getInputStream(),"UTF-8"
                        )
                );
                String line;
                /*传账号密码，用,分割*/
                writer.write(user + "," + password);
                writer.flush();
                /*收到数据输出到窗口文本区中*/
                while((line = reader.readLine()) != null){
                    window.appendText("收到： "+line);
                }
                /*关闭输入输出流*/
                writer.close();
                reader.close();
                writer = null;
                reader = null;
            } catch (UnknownHostException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }.start();
}
/*发出输入框的数据*/
public void send(String text) {
    if (writer != null){
        writer.write(text + "\n");
        writer.flush();
    }else {
        window.appendText("连接已断开");
    }
}

}
```

client内容

```
package Client;

import java.awt.*;

public class Client {
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            @Override
            public void run() {
                try {
                    ClientWindow frame = new ClientWindow();
                    frame.ClientWindow();
                    frame.setVisible(true);
                    ChatManage.getCM().setWindow(frame);
                }catch (Exception e){
                    e.printStackTrace();
                }
            }
        });
```

```
    }
}
```

clientWindow内容

```
package Client;

import javax.swing.*;
import javax.swing.border.EmptyBorder;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

/*客户端主窗实现*/
public class ClientWindow extends JFrame {
    private JPanel contentPanel;
    private JTextField ip;
    private JTextField user;
    private JTextField password;
    private JTextField send;
    private JTextArea textArea;
    public void ClientWindow(){
        setTitle("Client");
        setAlwaysOnTop(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(500,500,700,500);
        contentPanel = new JPanel();
        contentPanel.setBorder(new EmptyBorder(5,5,5,5));
        setContentPane(contentPanel);
        /*
        *ip栏、账号密码栏
        *备注：其中127.0.0.1是回送地址，即本机ip
        */
        ip = new JTextField();
        ip.setText("127.0.0.1");
        ip.setColumns(10);
        user = new JTextField();
        user.setText("账号");
        user.setColumns(10);
        password = new JTextField();
        password.setText("密码");
        password.setColumns(10);

        /*发送栏*/
        send = new JTextField();
        /*回车发送*/
        send.addKeyListener(new KeyAdapter() {
            @Override
            public void keyPressed(KeyEvent e) {
                if (e.getKeyCode() == KeyEvent.VK_ENTER){
                    ChatManage.getCM().send(send.getText());
                    appendText("我： " + send.getText());
                    send.setText("");
                }
            }
        });
```

```
        send.setText("");
        send.setColumns(10);
        /*登陆/发送键*/
        JButton sendButton = new JButton("登陆/发送");
        sendButton.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                ChatManage.getCM().send(send.getText());
                appendText("我：  \n" + send.getText());
                send.setText("");
            }
        });
        /*连接按钮*/
        JButton loginButton = new JButton("连接服务器");
        loginButton.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                ChatManage.getCM().connect(ip.getText(),user.getText(),password.getText()
                );
            }
        });
        /*画图按钮*/
        JButton drawButton = new JButton("画图");
        drawButton.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                DrawFrame draw = new DrawFrame();
                draw.setVisible(true);
            }
        });
        /*整体界面设计*/
        JScrollPane scrollPane = new JScrollPane();
        GroupLayout groupLayoutContentPane = new GroupLayout(contentPanel);
        groupLayoutContentPane.setHorizontalGroup(
                groupLayoutContentPane.createParallelGroup(GroupLayout.Alignment.TRAILING)
                        .addGroup(groupLayoutContentPane.createSequentialGroup()
                                .addGroup(groupLayoutContentPane.createParallelGroup(GroupLayout.Alignment.TRAILING)
                                        .addComponent(scrollPane, GroupLayout.Alignment.LEADING,GroupLayout.DEFAULT_SIZE,
600,Short.MAX_VALUE)
                                        .addGroup(groupLayoutContentPane.createSequentialGroup()
                                                /*ip、账号密码以及登陆按钮*/
                                                .addComponent(ip,GroupLayout.DEFAULT_SIZE,120,Short.MAX_VALUE)
                                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(user,GroupLayout.DEFAULT_SIZE,120,Short.MAX_VALUE)
                                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(password,GroupLayout.DEFAULT_SIZE,120,Short.MAX_VALUE)
                                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(loginButton,GroupLayout.DEFAULT_SIZE,80,Short.
MAX_VALUE))
                                        .addGroup(groupLayoutContentPane.createSequentialGroup()
                                                /*发送栏以及发送按钮*/
                                                .addComponent(send,GroupLayout.DEFAULT_SIZE,400,Short.MAX_VALUE)
                                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(sendButton,GroupLayout.DEFAULT_SIZE,150,Short.MAX_VALUE)
                                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                                .addComponent(drawButton,GroupLayout.DEFAULT_SIZE,20,Short.MAX_VALUE)
                                        )
                                )
```

```
                        .addGap(0))
        );
        groupLayoutContentPane.setVerticalGroup(
                groupLayoutContentPane.createParallelGroup(GroupLayout.Alignment.LEADING)
                        .addGroup(groupLayoutContentPane.createSequentialGroup()
                                .addGroup(groupLayoutContentPane.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(ip,GroupLayout.PREFERRED_SIZE,36,GroupLayout.PREFERRED_SIZE)
                                        .addComponent(user,GroupLayout.PREFERRED_SIZE,36,GroupLayout.PREFERRED_SIZE)
                                        .addComponent(password,GroupLayout.PREFERRED_SIZE,36,GroupLayout.
PREFERRED_SIZE)
                                        .addComponent(loginButton,GroupLayout.PREFERRED_SIZE,36,GroupLayout.
PREFERRED_SIZE))
                                .addGap(4)
                                .addComponent(scrollPane,GroupLayout.DEFAULT_SIZE,300,Short.MAX_VALUE)
                                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED)
                                .addGroup(groupLayoutContentPane.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(send,GroupLayout.PREFERRED_SIZE,40,GroupLayout.PREFERRED_SIZE)
                                        .addComponent(sendButton,GroupLayout.PREFERRED_SIZE,30,GroupLayout.
PREFERRED_SIZE)
                                        .addComponent(drawButton,GroupLayout.PREFERRED_SIZE,30,GroupLayout.
PREFERRED_SIZE)
                                )
                        )
        );
        textArea = new JTextArea();
        scrollPane.setViewportView(textArea);
        textArea.setText("准备就绪");
        contentPanel.setLayout(groupLayoutContentPane);

    }
    /*追加文本*/
    public void appendText(String text) {
        textArea.append("\n" + text);
    }
}
```

Server包

Beginserver

```
package Server;

public class BeginServer {
    public static void main(String[] args) {
        new ServerListen().start();
    }
}
```

chatHistory

```
package Server;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
```

```
public class ChatHistory {
    String sender = null;
    String[] infomation = null;
    String receiver = null;
    /*获取发送者和接受者*/
    public void getSenderAndReceiver(String sender , String receiver){
        this.sender = sender;
        this.receiver = receiver;
    }
    /*存入发送聊天记录*/
    public void pushSendRecord(String username ,String infomation) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlmakefriend = "INSERT INTO "+username+"Record"+" (SendorReceive,information) VALUES('send','"+"
to:"+this.receiver+": "+infomation+"');";
        try {
            preparedStatement = connection.prepareStatement(sqlmakefriend);
            preparedStatement.executeUpdate();
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement);
        }
    }
    /*存入接收聊天记录*/
    public void  pushReceiveRecord(String username , String infomation) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlmakefriend = "INSERT INTO "+username+"Record"+" (SendorReceive,information) VALUES
('receive','"+infomation+"');";
        try {
            preparedStatement = connection.prepareStatement(sqlmakefriend);
            preparedStatement.executeUpdate();
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement);
        }
    }
    /*查询私聊内容*/
    public ArrayList<String> searchPrivateRecord(String user) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlgetRecord = "select information from "+user+"record";
        ResultSet resultSet = null;
        ArrayList<String> records= new ArrayList<String>();
        try {
            preparedStatement = connection.prepareStatement(sqlgetRecord);
            resultSet = preparedStatement.executeQuery();
            while(resultSet.next()){
                records.add(resultSet.getString(1));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement,resultSet);
        }return records;
```

```
    }
}
```

chatServer

```
package Server;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.Socket;
import java.sql.SQLException;
import java.util.ArrayList;

public class ChatServer extends Thread{
    Socket socket;
    String username;

    /*获取socket*/
    public void ChatServer(Socket socket){
        this.socket = socket;

    }
    /*得到输出流*/
    public void out(String out){
        try{
            socket.getOutputStream().write((out+"\n").getBytes("UTF-8"));
        }catch (IOException e){
            ServerManager.getServerManager().remove(this);
            e.printStackTrace();
        }
    }
    @Override
    public void run(){
        /*连接到服务器*/
        out("您已连接服务器");
        try{
            /*获取socket输入流*/
            BufferedReader bufferedReader = new BufferedReader(
                    new InputStreamReader(
                            socket.getInputStream(),"UTF-8"
                    )
            );
            String line = null;
            //String username = null;
            /*验证登陆*/
            /*if ((*/line = bufferedReader.readLine();/*) == null){*/
                String[] login = line.split(",");
                Login login1 = new Login();
                login1.getUserAndPassword(login[0],login[1]);
                if (!login1.login()){
                    this.run();
                }
                    this.username = login[0];
            //}
            /*登陆成功*/
            out("您已登陆");
```

```
/*根据输入内容执行操作*/
while ((line = bufferedReader.readLine()) != null){
    String[] information = line.split("\\|");
    if (information[0].equals("makegroup")){
        String[] member = information[1].split(",");
        String groupname = information[2];
        GroupServer groupServer = new GroupServer();
        groupServer.getMemberAndGroupName(member,groupname);
        groupServer.makeGroup();
        out("您已建群,群名： " + groupname + " ,群成员: " + information[1] +"\n");
    }
    else if (information[0].equals("searchgroupmember") ){
        String groupname = information[1];
        GroupServer groupServer = new GroupServer();
        groupServer.getGroupName(groupname);
        String[] member;
        member = groupServer.searchGroupMember(groupname);
        out("群名: " + groupname);
        out("成员: ");
        for (int i = 0; i < member.length; i++) {
            out(member[i]);
        }
    }
    else if (information[0].equals("sendto")){
        String[] userAndInf = information[1].split(":");
        ChatHistory chatHistory = new ChatHistory();
        ServerManager.getServerManager().privateChat(this,userAndInf[1],userAndInf[0]);
        chatHistory.getSenderAndReceiver(this.username , userAndInf[0]);
        chatHistory.pushSendRecord(this.username,userAndInf[1]);
    }
    else if (information[0].equals("searchfriends")){
        FriendsServer friendsServer = new FriendsServer();
        friendsServer.getusername(this.username);
        String[] friends;
        friends = friendsServer.getFriends();
        out("好友： ");
        for (int i = 0; i < friends.length; i++) {
            if (!friends[i].equals(null)) {
                out(friends[i]);
            }
        }
    }
    else if (information[0].equals("makefriends")){
        FriendsServer friendsServer = new FriendsServer();
        friendsServer.getusernameAndfriendname(this.username,information[1]);
        friendsServer.makefriend();
        out("已添加好友！ ");
    }
    else if (information[0].equals("showrecord")){
        ChatHistory chatHistory = new ChatHistory();
        ArrayList<String> record =chatHistory.searchPrivateRecord(this.username);
        int size = record.size();
        String[] rec = (String[])record.toArray(new String[size]);
        out("聊天记录： ");
        for (int i = 0; i < rec.length; i++) {
            out(rec[i]);
        }
    }
```

```
                else if (information[0].equals("sendtogroup")){
                    String[] gui = information[1].split(":");
                    GroupServer groupServer = new GroupServer();
                    groupServer.sendtogroup(this.username,gui[0],gui[1]);
                    out("已发送到群组："+gui[0]);
                }
                else if (information[0].equals("showgrouprecord")){
                    GroupServer groupServer = new GroupServer();
                    ArrayList<String>grouprecord = groupServer.getgrouprecord(information[1]);
                    int size = grouprecord.size();
                    String[] rec = (String[])grouprecord.toArray(new String[size]);
                    out("群组"+information[1]+"聊天记录：");
                    for (int i = 0; i < rec.length; i++) {
                        out(rec[i]);
                    }
                }
                else if (information[0].equals("deletefriend")){
                    FriendsServer friendsServer = new FriendsServer();
                    friendsServer.delefriend(this.username,information[1]);
                    out("已删除好友："+information[1]);
                }
                else if (information[0].equals("deletegroupmember")){
                    String[] groupnameandmember = information[1].split(":");
                    GroupServer groupServer = new GroupServer();
                    groupServer.deletegroupmember(groupnameandmember[0],groupnameandmember[1]);
                    out("群成员："+groupnameandmember[1]+"已被删除");
                }
                else if (information[0].equals("deletegroup")){
                    GroupServer groupServer = new GroupServer();
                    groupServer.deletgroup(information[1]);
                    out("群组："+information[1]+"已被删除");
                }
                else if (information[0].equals("state")){
                    ServerManager.getServerManager().showstate(this);
                }
            }
            /*收尾*/
            bufferedReader.close();
            ServerManager.getServerManager().remove(this);
        } catch (IOException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    }
```

friendServer

```
package Server;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class FriendsServer {
```

```
String username;
String friendname;
/*获取用户名*/
public void getusername(String username){
    this.username = username;
}
public void getusernameAndfriendname(String username , String friendname){
    this.username = username;
    this.friendname = friendname;
}
/*获取好友列表*/
public String[] getFriends() throws SQLException {
    Connection connection = JDBC.getConnection();
    PreparedStatement preparedStatement = null;
    String sqlgetfrends = "SELECT * FROM frend WHERE frendA= ?";
    ResultSet resultSet = null;
    int count =0 ;
    String[] foundfriends = new String[10];
    try {
        preparedStatement = connection.prepareStatement(sqlgetfrends);
        preparedStatement.setString(1,this.username);
        resultSet = preparedStatement.executeQuery();
        while(resultSet.next()){
            foundfriends[count] = resultSet.getString(3);
            count++;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }finally {
        JDBC.close(connection,preparedStatement,resultSet);
    }
    return foundfriends;
}
/*添加好友*/
public void makefriend() throws SQLException {
    Connection connection = JDBC.getConnection();
    PreparedStatement preparedStatement = null;
    String sqlmakefriend = "INSERT INTO frend(frendA,frendB) VALUES('"+this.username+"','"+this.friendname+"');";
    try {
        preparedStatement = connection.prepareStatement(sqlmakefriend);
        preparedStatement.executeUpdate();
    }catch (SQLException e){
        e.printStackTrace();
    }finally {
        JDBC.close(connection,preparedStatement);
    }
}
public void  delefriend(String username, String friendname) throws SQLException {
    Connection connection = JDBC.getConnection();
    PreparedStatement preparedStatement = null;
    String sqlmakefriend = "delete from frend where frendA = '"+username+"' and frendB = '"+friendname+"';";
    try {
        preparedStatement = connection.prepareStatement(sqlmakefriend);
        preparedStatement.executeUpdate();
    }catch (SQLException e){
        e.printStackTrace();
    }finally {
        JDBC.close(connection,preparedStatement);
```

```
        }
    }

}
```

GroupServer

```
package Server;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;

public class GroupServer {
    String[] member;
    String groupname;
    /*获取成员和群名*/
    public void getMemberAndGroupName(String[] member , String groupname){
        this.member = member;
        this.groupname = groupname;
    }
    /*获取群名*/
    public void getGroupName(String groupname){
        this.groupname = groupname;
    }
    /*建群*/
    public void makeGroup() throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;

        for (int i = 0; i < member.length; i++) {
            try {
                String sqlmakeGroup = "INSERT INTO groupship (groupname,groupmember) VALUES ("+groupname+","
+member[i]+");";
                preparedStatement = connection.prepareStatement(sqlmakeGroup);
                preparedStatement.executeUpdate();
            }catch (SQLException e) {
                e.printStackTrace();
            }
        }
        JDBC.close(connection,preparedStatement);
    }
    /*搜索群成员，返回成员数组*/
    public String[] searchGroupMember(String groupname) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String[] foundmember = new String[10];
        String sqlsearchMember = "SELECT * FROM groupship WHERE groupname= ?";
        ResultSet resultSet = null;
        try {
            preparedStatement = connection.prepareStatement(sqlsearchMember);
            preparedStatement.setString(1,groupname);
            resultSet = preparedStatement.executeQuery();
            int count = 0;
            while (resultSet.next()) {
```

```
                foundmember[count] = resultSet.getString(3);
                count++;
            }
            } catch (SQLException throwables) {
            throwables.printStackTrace();
        } finally {
            JDBC.close(connection,preparedStatement,resultSet);
        }
        return foundmember;
    }
    /*群组聊天记录*/
    public void sendtogroup(String username , String groupname , String infomation) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlmakefriend = "INSERT INTO "+"grouprecord"+" (groupname,username,information) VALUES
('"+groupname+"','"+username+"','"+infomation+"');";
        try {
            preparedStatement = connection.prepareStatement(sqlmakefriend);
            preparedStatement.executeUpdate();
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement);
        }
    }
    public ArrayList<String> getgrouprecord(String groupname ) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlgetRecord = "select username,information from "+"grouprecord where groupname = "+groupname+";";
        ResultSet resultSet = null;
        ArrayList<String> records= new ArrayList<String>();
        try {
            preparedStatement = connection.prepareStatement(sqlgetRecord);
            resultSet = preparedStatement.executeQuery();
            while(resultSet.next()){
                records.add(resultSet.getString(1)+":");
                records.add(resultSet.getString(2));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement,resultSet);
        }return records;
    }
    /*删除群员*/
    public void deletegroupmember(String groupname , String groupmember) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlmakefriend = "delete from groupship where groupname = '"+groupname+"' and groupmember =
'"+groupmember+"';";
        try {
            preparedStatement = connection.prepareStatement(sqlmakefriend);
            preparedStatement.executeUpdate();
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement);
        }
```

```
    }
    /*删除群*/
    public void deletgroup(String groupname) throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqlmakefriend = "delete from groupship where groupname = '"+groupname+"';";
        try {
            preparedStatement = connection.prepareStatement(sqlmakefriend);
            preparedStatement.executeUpdate();
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
            JDBC.close(connection,preparedStatement);
        }
    }
}
```

JDBC（此类是封装jdbc操作，便于写程序，如果会mybatis，也可以用mybatis代替程序中的sql相关操作，当时没学，现在学了但是暂时没时间重写）

```
package Server;

import javax.xml.transform.Result;
import java.sql.*;

public class JDBC {
    public Connection connection;
    public PreparedStatement preparedStatement;
    public Result result;
    private static final String driverClass = "com.mysql.cj.jdbc.Driver";
    private static final String url = "jdbc:mysql://localhost:3306/";
    private static final String user = "yourroot";
    private static final String password = "yourpassword";
    /*注册驱动*/
    static {
        try{
            Class.forName(driverClass);
        }catch (ClassNotFoundException e){

        }
    }
    /*获取连接*/
    public static Connection getConnection() throws SQLException {
        Connection connection = DriverManager.getConnection( url , user , password);
        return connection;
    }
    /*关闭重载*/
    public static void close(Connection connection , Statement statement)  {
        if (statement != null){
            try {
                statement.close();
            }catch (SQLException e){
            }
        }
        if (connection != null){
            try {
                connection.close();
            }catch (SQLException e){
```

```
            }
        }}
        /*关闭资源*/
        public static void close(Connection connection , Statement statement , ResultSet resultSet){
            if (resultSet != null){
                try{
                    resultSet.close();
                }catch (SQLException e){
                }
            }
            if (statement != null){
                try {
                    statement.close();
                }catch (SQLException e){
                }
            }
            if (connection != null){
                try {
                    connection.close();
                }catch (SQLException e){
                }
            }
        }
    }
```

Login

```
package Server;

import java.sql.*;

public class Login {
    /*账号密码以及其获取函数*/
    private String user ;
    private String password;
    public void getUserAndPassword(String user, String password){
        this.user = user;
        this.password = password;
    }

    /*通过sql查找*/
    public boolean login() throws SQLException {
        Connection connection = JDBC.getConnection();
        PreparedStatement preparedStatement = null;
        String sqllogin = "SELECT * FROM chatuser WHERE username= '"+this.user+"' AND password = '"+this.
password+"' ;";
        ResultSet resultSet = null;
        try{
            connection = JDBC.getConnection();
            preparedStatement = connection.prepareStatement(sqllogin);
            resultSet = preparedStatement.executeQuery();
            if (resultSet.next()){
                return true;
            }
        }catch (SQLException e){
            e.printStackTrace();
        }finally {
```

```
            JDBC.close(connection,preparedStatement,resultSet);
        }
        return false;
    }

}
```

ServerListen

```
ackage Server;

import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class ServerListen extends Thread{
    @Override
    public void run() {
        try{
            ServerSocket serverSocket = new ServerSocket(8848);
            while (true){
                Socket socket = serverSocket.accept();
                /*连接客户端传送到新线程*/
                ChatServer chatServer = new ChatServer();
                chatServer.ChatServer(socket);
                chatServer.start();
                /*整合到服务器管理集合*/
                ServerManager.getServerManager().add(chatServer);
            }
        }catch (IOException e){
            e.printStackTrace();
        }
    }
}
```

ServerManager

```
package Server;

import java.sql.SQLException;
import java.util.Vector;

public class ServerManager {
    /*将不同客户端变成一个个单例*/
    private void ServerManager(){}
    private static final ServerManager serverManager = new ServerManager();
    public static ServerManager getServerManager(){
        return serverManager;
    }
    /*将不同客户端放入vector容器中*/
    Vector <ChatServer> vector = new Vector<ChatServer>();
    public void add(ChatServer chatServer){
        vector.add(chatServer);
    }
    /*断开连接*/
    public void remove(ChatServer chatServer){
        vector.remove(chatServer);
```

```
    }
    /*私聊*/
    public void privateChat(ChatServer chatServer , String out , String receiver) throws SQLException {
        for (int i = 0; i < vector.size() ; i++) {
            ChatServer chatServer1 = vector.get(i);
            /*用户名符合时发送*/
            if (chatServer1.username.equals(receiver)){
                ChatHistory chatHistory = new ChatHistory();
                chatServer1.out(chatServer.username+": "+out);
                chatHistory.pushReceiveRecord(receiver,"from:"+chatServer.username+":"+out);
            }
        }
    }
    public void showstate(ChatServer chatServer){
        for (int i = 0; i < vector.size(); i++) {
            ChatServer chatServer1 = vector.get(i);
            /*在线时发送用户名+在线*/
            if (!chatServer.equals(chatServer1)){
                chatServer.out(chatServer1.username+"在线");
            }
        }
    }
}
```

FROM：https://juejin.cn/post/6859332834092548110