用户管理系统V2.0

- 新增功能：
  - 登录之后显示用户主页，并隐藏登录界面
  - 用户主页新增退出登录按钮，隐藏主页，显示登录界面
  - 用户页面新增用户个人信息输入框，并实现点击更新信息按钮，实现更新用户在数组中的信息
  - 添加用户管理页面，实现查看用户用户 以及刷新功能

```java
package com.oop.userManageV2;

public class User {


    // 面向对象部分： 属性  方法
    // 属性：
    String userName;
    String password;
    int score;
    boolean isLogin;
    String phoneNumber;
    String address;
    String email;

    // 构造方法
    public User(String name, String password) {
        userName = name;
        this.password = password;
        // 当参数名与属性名相同时，又需要在这个方法中将参数的值赋给属性，此时可以使用this.
属性名 = 参数名;
        // this.后面的都是属性 指代调用属性
        System.out.println("带参数的构造方法执行");
    }

    public User() {
        System.out.println("空参数的构造方法执行");
    }


    //方法
    public void showInfo() {
        System.out.println("账号：" + userName + " 密码：" + password + " 积分：" +
score + " 在线状态：" + isLogin);
    }

    public boolean login(String pwdIn) {
        // 使用传入的密码与自己的密码进行比较
        if (pwdIn.equals(password)) {
            score += 100;
            isLogin = true;
            System.out.println("登录成功");
            return true;
        } else {
            System.out.println("密码错误");
            return false;
        }
    }
```

```
    // 主函数  创建对象
    public static void main(String[] args) {
        // 创建一个用户对象 ，使用的是第一个带参数的构造方法
        User user1 = new User("admin", "admin123");
//        user1.userName="admin";
        // user1.password = "admin123";

        user1.login("admin123");
        user1.showInfo();
        // 创建一个用户对象 ，使用的是第二个不带参数的构造方法
        User user2 = new User();
        user2.userName = "user2";
        user2.password = "user2123";
        user2.showInfo();
    }


    public void logOut() {
        isLogin = false;
    }
}
```

```java
package com.oop.userManageV2;

import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class UserAction implements ActionListener {

    // 前端组件:
    // 1: 在此处创建一个输入框变量 不要初始化创建对象
    // 登录注册的输入框
    JTextField nameJtfL;// 登录的输入框
    JPasswordField pwdJpfL;

    // UserUI 对象
    UserUI userUI;// 可以调用userHomeUI显示

    // 用户主页个人信息输入框数组
    JTextField[] infoJtfs;


    // 创建一个可以存储多个用户对象的用户数组
    User[] userList = new User[10];
    int userCount = 0;

    // 往数组中插入一些模拟数据
    {// 代码块 第一时间创建对象时执行
        for (int i = 0; i < 8; i++) {
            User user = new User("admin" + i, "123" + i);
            userList[i] = user;
            userCount++;
        }
        System.out.println("用户数据生成模拟完成");

    }

    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("按钮被点击了");
        String ac = e.getActionCommand();// 获取按钮的动作指令文本，如果按钮没有设置动作指令文本，默认为是按钮上文本

        // 获取输入框中的文本
        String name = nameJtfL.getText();
        System.out.println(name);
        String pwd = pwdJpfL.getText();
        System.out.println(pwd);

        if (ac.equals("登录")) {
            // 根据名字查询是否已经注册了
```

```
            for (int i = 0; i < userCount; i++) {
                User user = userList[i];
                if (user.userName.equals(name)) {
                    // 再判断密码
                    boolean isLogin = user.login(pwd);
                    if (isLogin) {
                        // 跳转主页
                        userUI.hideLoginUI();
                        userUI.userHomeUI(user);
                        break;
                    }
                }
            }
        } else if (ac.equals("注册")) {

            int count = 0;
            for (int i = 0; i < userCount; i++) {
                User user = userList[i];
                if (user.userName.equals(name)) {
                    System.out.println("用户已注册");
                    count++;
                }
            }
            if (count == 0) {
                // 将输入框的文本作为参数创建一个对象
                User user = new User(name, pwd);
                userList[userCount] = user;
                userCount++;
                System.out.println("注册成功");
                System.out.println("现有的用户个数: " + userCount);
            }
        } else if (ac.equals("退出登录")) {
            // 第一步 获取输入框中的内容
            String userName = infoJtfs[0].getText();// 获取用户名知道是谁登录
            // 在数组中查找用户
            for (int i = 0; i < userCount; i++) {
                User user = userList[i];
                if (user.userName.equals(userName)) {
                    user.logOut();
                }
            }
            userUI.hideUserHomeUI();
            userUI.loginUI();
        } else if (ac.equals("更新信息")) {
            // 第一步 获取输入框中的内容
            String userName = infoJtfs[0].getText();// 获取用户名知道是谁登录
            // 获取需要修改的数据
            String password = infoJtfs[1].getText();
            String phoneNumber = infoJtfs[3].getText();
            String email = infoJtfs[4].getText();
            String address = infoJtfs[5].getText();
```

```
            // 在数组中查找用户
            for (int i = 0; i < userCount; i++) {
                User user = userList[i];
                if (user.userName.equals(userName)) {
                    user.password = password;
                    user.phoneNumber = phoneNumber;
                    user.email = email;
                    user.address = address;
                    break;
                }
            }
            System.out.println("信息修改成功");
        } else if (ac.equals("刷新")) {
            userUI.hideAdminUI();
            userUI.showAdminUI(userList);
        }

    }
}
```

```java
package com.oop.userManageV2;

import javax.swing.*;
import javax.swing.table.AbstractTableModel;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import java.awt.*;
import javax.swing.table.DefaultTableModel;

public class UserUI {
    // 创建一个监听对象
    UserAction userAction = new UserAction();

    // 窗体
    JFrame loginFrame;
    JFrame userHomeFrame;

    // 登录界面
    public void loginUI() {
        loginFrame = new JFrame("用户登录界面");
        loginFrame.setSize(400, 300);
        loginFrame.setLocationRelativeTo(null);
        loginFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        FlowLayout layout = new FlowLayout();
        loginFrame.setLayout(layout);

        JLabel nameJla = new JLabel("账号: ");
        JTextField nameJtf = new JTextField(30);
        JLabel pwdJla = new JLabel("密码: ");
        JPasswordField pwdJtf = new JPasswordField(30);
        JButton loginJbt = new JButton("登录");
        JButton registerJbt = new JButton("注册");

        loginFrame.add(nameJla);
        loginFrame.add(nameJtf);
        loginFrame.add(pwdJla);
        loginFrame.add(pwdJtf);
        loginFrame.add(loginJbt);
        loginFrame.add(registerJbt);

        loginFrame.setVisible(true);

        // 登录按钮加载监听事件
        loginJbt.addActionListener(userAction);
        registerJbt.addActionListener(userAction);

        // 使用监听对象调用它的nameJtfL 初始化
        userAction.nameJtfL = nameJtf;
        userAction.pwdJpfL = pwdJtf;
```

```
        // 将UserUI对象传到监听器中 ， this表示调loginUI方法的对象
        userAction.userUI = this;
    }

    // 主界面
    public void userHomeUI(User user) {
        userHomeFrame = new JFrame();
        userHomeFrame.setTitle("用户个人主页");
        userHomeFrame.setSize(400, 300);
        userHomeFrame.setLocationRelativeTo(null);
        FlowLayout flow = new FlowLayout();
        userHomeFrame.setLayout(flow);


        String[] infoJlaStrs = {"账号: ", "密码: ", "积分: ", "电话: ", "邮箱: ", "地址:
", "在线状态: "};
        String[] infoStrs = {user.userName, user.password, user.score + "",
                user.phoneNumber, user.email, user.address, user.isLogin ? "在线" : "离
线"};
        int infoSize = infoJlaStrs.length;
        JTextField[] userInfoJtfs = new JTextField[infoSize];

        for (int i = 0; i < infoSize; i++) {
            JLabel jla = new JLabel(infoJlaStrs[i]);
            JTextField jtf = new JTextField(infoStrs[i]);
            if (i == infoSize - 1) {
                jtf.setPreferredSize(new Dimension(260, 25));

            } else {
                // 设置尺寸
                jtf.setPreferredSize(new Dimension(310, 25));
            }
            userHomeFrame.add(jla);
            userHomeFrame.add(jtf);
            userInfoJtfs[i] = jtf;
            // 禁用 账号 积分 在线标识修改
            if (i == 0 || i == 2 || i == 6) {
                jtf.setEditable(false);
            }
        }

        JButton btn1 = new JButton("更新信息");
        JButton btn2 = new JButton("退出登录");
        btn1.addActionListener(userAction);
        btn2.addActionListener(userAction);
        userHomeFrame.add(btn1);
        userHomeFrame.add(btn2);

        userHomeFrame.setVisible(true);

        // 将存储了输入框的数组 传到监听器中
```

```
        userAction.infoJtfs = userInfoJtfs;
    }

    // 调用时隐藏登录界面
    public void hideLoginUI() {
        loginFrame.setVisible(false);
    }

    // 调用时隐藏主页
    public void hideUserHomeUI() {
        userHomeFrame.setVisible(false);
    }

    // 注册界面
    // 信息修改界面
    JFrame adminFrame;
    public void hideAdminUI(){
        adminFrame.setVisible(false);
    }

    // 管理界面
    public void showAdminUI(User[] userList) {
        adminFrame = new JFrame("管理员界面");
        adminFrame.setSize(820, 520);
        adminFrame.setLocationRelativeTo(null);
        adminFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        FlowLayout flow = new FlowLayout();
        adminFrame.setLayout(flow);

        JTable table;
        UserTableModel tableModel;
// 设置列名
        Object[] columnNames = {"账号", "密码", "电话", "地址", "积分", "邮箱", "在线状
态"};

        // 初始化表格模型
        tableModel = new UserTableModel(null, columnNames);
        table = new JTable(tableModel);

        // 添加数据
        // 添加示例数据
        Object[][] data = new Object[userList.length][7];
        for (int i = 0; i < data.length; i++) {
            User user = userList[i];
            if (user != null) {
                data[i][0] = user.userName;
                data[i][1] = user.password;
                data[i][2] = user.phoneNumber;
                data[i][3] = user.address;
                data[i][4] = user.score;
                data[i][5] = user.email;
```

```
                data[i][6] = user.isLogin ? "在线" : "离线";
            }
        }
        for (int i = 0; i < data.length; i++) {
            Object[] row = data[i];
            tableModel.addRow(row);
        }

        // 将表格放入滚动面板中
        JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setPreferredSize(new Dimension(800, 400));
        adminFrame.add(scrollPane);

        JButton btn = new JButton("刷新");
        btn.addActionListener(userAction);
        adminFrame.add(btn);

        adminFrame.setVisible(true);

    }


    public static void main(String[] args) {
        UserUI userUI = new UserUI();
        userUI.loginUI();

        // 创建一个可以存储多个用户对象的用户数组
        User[] userList = new User[10];
        int userCount = 0;

        // 往数组中插入一些模拟数据
        {// 代码块 第一时间创建对象时执行
            for (int i = 0; i < 8; i++) {
                User user = new User("admin" + i, "123" + i);
                userList[i] = user;
                userCount++;
            }
            System.out.println("用户数据生成模拟完成");

        }
        userUI.showAdminUI(userList);
    }

}


class UserTableModel extends DefaultTableModel {
    private static final long serialVersionUID = 1L;

    public UserTableModel(Object[][] data, Object[] columnNames) {
        super(data, columnNames);
```

```
    }

    @Override
    public boolean isCellEditable(int row, int column) {
        // 根据需要设置哪些列可编辑，这里假设密码列不可编辑
        if (column == 1) {
            return false;
        }
        return true;
    }
}
```

练习：

- 跟着回放把代码完成，实现讲解新功能如何让实现的博客