用户管理系统

- 类与对象： 类的创建 对象的创建以及使用
  - 界面开发
  - 事件监听器
- API: 数组 输入 循环 条件语句

项目的四个要素：

- 用户交互 ：
  - 界面开发 事件监听
- 功能逻辑控制：
  - 对应的功能
- 数据模型
  - 项目需要使用的数据
- 数据存储
  - 数据结构
- 用户留言管理系统：
  - 普通用户： 可以登录 注册 查看信息 修改信息 可以留言 管理留言
  - 管理员： 可以登录 查看所有用户信息 支持所有用户的管理
- 文件存储/本地文件保险柜：
  - 实现一个可视化界面，支持对于文件的上传 下载 查看等功能

用户留言管理系统：

- 普通用户： 可以登录 注册 查看信息 修改信息 可以留言 管理留言
- 管理员： 可以登录 查看所有用户信息 支持所有用户的管理
- 开发版本：

- 1.0: 支持实现一个登录注册界面，可以实现登录之后跳转到一个个人主页，并且个人主页上可以显示自己的信息 以及包含退出登录的按钮
- 2.0： 实现对于个人主页上的信息修改 】
- 3.0： 实现增加留言信息以及查看留言信息
- 4.0： 实现对于留言的管理 【删除留言】

类与对象：

- 类与对象的概念：
  - 类：class 类别
  - 对象：object 具体的事物的意思
    - 电脑： 类
    - 振宇正在使用电脑： 对象
  - 学会分析一个类别的属性 特征 行为 功能 | 数据集 指令集
- 在代码中，类本身也是一个数据类型，是一种复合型数据类型结构，可以包含多个数据以及对于这些数据操作
- 命名规则：
  - 项目名 文件名 类名： 要求首字母大写，多个单词拼接，每个单词首字母大写
  - 包名： 全小写，不能使用数字符号开头 字母开头 ， 包的层级之间使用.分隔
  - 属性：
    - 变量: 对象的成员属性 要求首字母小写，后续多个单词可以使用首字母大写
      - PS: name userName
    - 常量： 全部大写 EXIT_ON_CLOSE
  - 方法名： 首字母小写，多个单词拼接，后面每个单词的首字母大写
    - showInfo login register
  - 变量名：要求首字母小写，后续多个单词可以使用首字母大写
- 类的定义：
  - 语法： public class 类名{ }
  - 步骤： 右键 包名 - new - Java Class - 命名 - 回车
  - 内容：
    - 面向对象的层次讲： 属性 和 方法 用户(账号 密码 | 登录 登出)
      - 属性： 表示这个类别都有的数据信息 通常使用变量表示
        - 格式： 访问权限修饰符 属性的数据类型 属性变量名;

- 方法：表示这个类别都可以使用/具有的功能/行为 通常我们需要使用一段代码结构表示
    - 格式： 访问权限修饰符 返回值类型 方法名(参数类型 参数变量名){ }
- 从代码功能结构上讲还包含：
    - 构造方法：
        - 格式： public 类名(){ }
            - 特性：
                - 1：没有返回值 void这种结构
                - 2：以类名为方法名
                - 3：任何类都有一个隐式的空参数构造方法
                - 4：一旦显式的写了构造方法，不管是否带参数都会覆盖隐式的
                - 5：构造方法可以设定参数 ， 参数可以将值赋给属性，实现创建对象时属性的数据初始化
                - 6：一个类中可以创建多个构造方法，必须参数不一致
                - 7：构造方法在创建对象时执行，每个对象创建时都会执行一次构造方法
                - 8：创建对象可以选择合适的构造方法创建对象
    - 构造代码块：{} 创建对象时执行
    - 静态资源： static 修饰的属性和方法 ， 可以使用类名直接调用 与对象无关
        - static修饰的属性在内存中只会创建唯一的一份

```java
package com.oop.zyf1013;

public class User {


    // 面向对象部分：  属性   方法

    // 属性：
    String userName;
    String password;
    int score;
    boolean isLogin;

    // 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");
    }

    // 构造代码块：  可以用来执行一些创建对象时就需要执行的代码 初始化等等
    {
        System.out.println("代码块1");
    }

    {
        System.out.println("代码块2");
    }

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

    public void 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");
        } else {
            System.out.println("密码错误");
```

```
        }
    }


    // 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();
    }

}
```

界面开发与动作监听器：

- 界面开发：
  - 界面的组成：
    - 可视化组件部分： 窗体 按钮 输入框
    - 元素规则： 尺寸 颜色 字体布局
    - 内容：文本 图片
  - 界面开发包： java.awt \ javax.swing
    - 在自己的类中使用其他包中的类时需要导入这些所在的包路径
    - 一定要注意自己的类中有没有导入无关的包路径
    - 导入包：
      - import java.awt.*;
      - import javax.swing.JButton;
    - 界面开发包中的类：
      - awt: FlowLayout
      - swing: JFrame JButton JTextField JPasswordField JLabel
  - 开发步骤:
    - 1: 创建一个新的类 UserUI ,创建一个loginUI方法， 创建一个主函数，其中创建一个UserUI对象 调用loginUI方法

        - 2： 在loginUI方法中创建一个窗体类对象，使用窗体类对象的set系列方法设置相关属性： 尺寸 标题 布局 可视化 关闭方式
        - 3： 创建两个文本标签 一个登录按钮 两个输入框 添加到窗体中
- 事件监听器：
    - 监听器的使用步骤:
        - 1: 创建一个新的类UserAction 实现ActionListener接口
        - 2：重写ActionListener中的方法 actionPerformed
        - 3：使用需要实现监听的按钮对象调用addActionListener方法添加一个UserAction对象
    - 如何区分不同的按钮
    - 获取输入框的文本：
        - String str = 输入框对象名.getText();

```java
package com.oop.zyf1013;

public class User {


    // 面向对象部分：  属性   方法

    // 属性：
    String userName;
    String password;
    int score;
    boolean isLogin;

    // 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");
    }

    // 构造代码块：  可以用来执行一些创建对象时就需要执行的代码 初始化等等
    {
        System.out.println("代码块1");
    }

    {
        System.out.println("代码块2");
    }

    //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();
    }


}
```

```java
package com.oop.zyf1013;

import javax.swing.*;
import java.awt.*;

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


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

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

        jf.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;
    }

    // 主界面
    // 注册界面
    // 信息修改界面
```

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

}
```

```java
package com.oop.zyf1013;

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

public class UserAction implements ActionListener {

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

    // 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
                        JFrame jf = new JFrame();
```

```
                        jf.setSize(500, 500);
                        jf.setVisible(true);
                        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);
            }
        }

    }
}
```

练习:

- 完成使用用户类和数组的登录注册程序， 并写一篇博客