浙江工业大学

JavaEE技术实验报告

实验名称：**Servlet** 与 **JSP** 技术第一个用户登录模块

学 院： 计算机学院

班 级： 软件工程1801

姓 名： 陈南

学 号： 201806062101

组 号： 无

时 间： 2020年10月10日

# 一、实验主要步骤

## 一、基础实验 ——Servlet 与 JSP 基础开发

（一）实验目的

1. 掌握 HttpServlet 的概念，相关 API 以及开发步骤；
2. 掌握 JSP 技术的基本语法；
3. 掌握 JSP 各隐含变量的使用方法；
4. 掌握使用 Servlet 和 JSP 集成开发简单用户登录功能。

（二）基本知识与原理

1. Servlet 是用于实现 Web 应用程序设计的 Java 技术解决方案，旨在扩展

Web

服务器的能力，它是由 Servlet 容器（例如：Tomcat）创建并管理。

1. JSP（Java Server Pages）页面是包含 Java 代码和 HTML 标签的 Web 页面。

它由 JSP 标签和 HTML 标签混合而成的 Web 页面，主要用于进行用户交互。

（三）实验内容及步骤

1. 下载并解压安装 Eclipse 的 Java EE 集成开发环境
2. 在 Eclipse 中新建动态 Web 工程(Dynamic Web Project ) javaweb-prj1
3. 为工程 javaweb-prj1 添加 Servlet 开发的外部库文件 servlet-api.jar，操作步

骤如下：

1. 右键单击工程 javaweb-prj1 图标，在弹出的菜单中选择

[Build

Path]->[Configure Build Path]，如图 1-3 所示；

1. 在弹出的窗口中选择[Java Build Path]的[Libraries]选项卡，点击按钮[Add

External JARS]，接着在弹出的窗口中切换到 Tomcat 安装目录（例如：

C:\Program Files (x86)\Apache Software Foundation\Tomcat 6.0 ）下的 lib 目录，然后选中并打开加载 servlet-api.jar 库文件，最后点击按钮[OK]

4、在 javaweb-prj1 中，右键单击目录[WebContent]，新建用户登录页面 login.jsp，

如图 1-5 所示，具体代码如下所示（注意表单 form 属性 action 的值，用户名与密码输入框中属性 name 的值）；

图 1-5 创建 JSP 页面

<%@ page language="java" contentType="text/html; charset=GBK" pageEncoding="GBK"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=GBK">

<title>用户登录页面</title>

</head>

<body>

<form action="login" method="post">

请输入用户名：<input name="username" type="text"><br> 请输入密码：<input name="password" type="password">

<input type="submit" value="登录">

</form>

</body>

</html>

5、在 javaweb-prj1 中，右键单击[Java Resources]->[src]目录，新建一个名称为 cn.edu.zjut 的包，然后在该包下创建一个 Servlet：LoginController.java（如图

1-6 所示），实现如下功能：用于接收 login.jsp 页面提交的用户名和密码，如果用户名和密码均为 zjut，则输出“登录成功，欢迎您！”，否则输出“用户名或密码错误！”

package cn.edu.zjut; import java.io.\*; import javax.servlet.\*; import javax.servlet.http.\*; public class LoginController extends HttpServlet { protected void doPost(HttpServletRequest request,

HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html;charset=utf-8");

PrintWriter out=response.getWriter();

String username = request.getParameter("username"); String password = request.getParameter("password"); if("zjut".equals(username) && "zjut".equals(password)){

out.println("登录成功，欢迎您！");

}else{

out.println("用户名或密码错误！");

}

}

}

6、在 javaweb-prj1 的[WebContent]->[WEB-INF]目录下新建 web.xml 文件，为

LoginController 配置 url-pattern 映射，具体代码如下所示：

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:web="http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">

<display-name>javaweb-prj1</display-name>

<servlet>

<servlet-name>LoginController</servlet-name>

<display-name>LoginController</display-name>

<description></description>

<servlet-class>cn.edu.zjut.LoginController</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginController</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

</web-app>

7、将 javaweb-prj1 部署在 Tomcat 服务器上，具体操作步骤如下：

1. 下载安装好 Tomcat（本例为版本 6.0，实际应用中可选择 6.0 以上版本）；
2. 打开 Eclipse，选择菜单[Window]->[Preferences]
3. 在弹出的窗口左侧栏中选择[Server]->[Runtime Environments]，然后点击右侧栏的按钮[add]
4. 在弹出的服务器窗口中选择[Apache Tomcat v6.0]，并点击按钮[next]，此时进入 Tomcat 的服务器配置窗口，点击[Browse]按钮，接着

在弹出的目录选择窗口中选择 Tomcat 安装目录（例如：C:\Program Files

(x86)\Apache Software Foundation\Tomcat 6.0 ），点击按钮[ 确定]完成

Tomcat 服务器的选择，然后依次点击按钮[Finish]和按钮[OK]来保存服务器

配置方案

1. 右键单击 javaweb-prj1 工程，在弹出的菜单中依次选择[Run As]->[Run on Server]，如所示；在弹出的窗口中选择刚才配置的[Tomcat v6.0 Server]，并点击按钮[Finish]，此时服务器会自动启动并加载运行 javaweb-prj1 工程，

同时服务器启动的相关信息会出现在 Console 控制栏

1. 打开任一浏览器，输入网址：http://localhost:8080/javaweb-prj1/login.jsp，即可开始访问 login.jsp 页面，进行相关运行调试。 8、运行 login.jsp 页面，输入用户名和密码并记录运行结果；

9、修改 login.jsp 页面，使用表格对表单域进行对齐排列，运行并观察结果； 10、修改 login.jsp 页面，使用 JavaScript 对用户名表单域 username 和密码表

单域 password 进行校验(校验规则：不能为空且不能超过 6 位)，运行并观察结果。

（四）实验要求

1、填写并上交实验报告，报告中应包括：

1. 运行结果截图；
2. 修改后的关键代码，及相应的运行结果或报错信息；（3）实验收获及总结。

2、上交程序源代码，主要代码中应加注详细的注释。

## 二、提高实验 ——Servlet 与 JSP 集成的 MVC 方案

（一）实验目的

1. 掌握 JavaBeans 的编写要点；
2. 掌握 MVC 设计模式的三个组成要素：模型 Model、视图 View 和控制器

Controller；

1. 能在 Web 应用程序设计中熟练使用 MVC 模式。

（二）基本知识与原理

1. MVC 模式将交互式应用分成模型（Model）、视图（View）和控制器（Controller）三个部分。
2. 模型（Model）是指从现实世界中抽象出来的对象模型，是应用逻辑的反映。模型封装了数据和数据的操作，是实际进行数据处理计算的地方。 3、视图（View）是应用和用户之间的接口，它负责将应用显现给用户和显示模型的状态。
3. 控制器（Controller）负责视图和模型之间的交互，控制对用户输入的响应、响应方式和流程。它主要负责两方面的动作：把用户的请求分发到相应的模型；将模型的改变及时反映到视图上。
4. MVC 设计模式将业务逻辑和显示逻辑进行了良好的分离，使代码更加清晰，可维护性更好。（三）实验内容及步骤

1、在工程 javaweb-prj1 中新建用户模型 UserBean.java，具体代码如下所示： package cn.edu.zjut.model; public class UserBean {

// 属性声明

private String username; private String password;

//构造方法

public UserBean(){}

// get 方法

public String getUsername(){ return username; } public String getPassword(){ return password; }

// set 方法

public void setUsername(String username){ this.username = username;

}

public void setPassword(String password){ this.password = password;

}

}

2、在 javaweb-prj1 中新建视图 loginSuccess.jsp 和 loginFailed.jsp 页面，分别

用于显示登录成功和登录失败后的页面，具体代码如下所示：

<!-- loginSuccess.jsp 页面源代码 -->

<%@ page language="java" contentType="text/html; charset=GBK" pageEncoding="GBK" import="cn.edu.zjut.model.UserBean"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=GBK"> <title>登录成功</title>

</head>

<body>

<% UserBean user=(UserBean)(request.getAttribute("USER")); %> 登录成功,欢迎您,<%=user.getUsername() %>!

</body>

</html>

<!-- loginFailed.jsp 页面源代码 -->

<%@ page language="java" contentType="text/html; charset=GBK" pageEncoding="GBK"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=GBK">

<title>登录失败</title>

</head>

<body>

登录失败，用户名或密码错误！

</body>

</html>

3、修改 javaweb-prj1 中的控制器 LoginController.java，实现如下功能：

1. 调用 UserBean 用户模型记录用户信息；
2. 如果用户名和密码均为 zjut，则跳转到视图 loginSuccess.jsp，否则跳转到

视图 loginFailed.jsp，具体代码如下所示：

package cn.edu.zjut; import java.io.\*; import javax.servlet.\*; import javax.servlet.http.\*; import cn.edu.zjut.model.\*;

public class LoginController extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse

response) throws ServletException, IOException { String username = request.getParameter("username");

String password = request.getParameter("password");

UserBean user=new UserBean(); user.setUsername(username); user.setPassword(password); if(checkUser(user)){ request.setAttribute("USER", user); RequestDispatcher dispatcher = request .getRequestDispatcher("/loginSuccess.jsp"); dispatcher.forward(request, response); }else{

response.sendRedirect("/javaweb-prj1/loginFailed.jsp");

} } boolean checkUser(UserBean user){ if("zjut".equals(user.getUsername()) &&

"zjut".equals(user.getPassword())) { return true;

}else{

return false;

}

}

}

1. 使用 javaweb-prj1 工程中的 web.xml，为 LoginController 配置 url-pattern 映射；
2. 将 javaweb-prj1 工程重新部署在 Tomcat 服务器上；
3. 运行 login.jsp 页面，输入用户名和密码并运行，观察运行结果，如果出现运行错误，请尝试修正；
4. 对 javaweb-prj1 工程做如下修改：
5. 修改视图 login.jsp，新增一个下拉框的表单域，用于选择用户类型（值为

“普通用户”和“管理员”）；

1. 修改用户模型 UserBean.java，新增一个用户类型 type 属性；
2. 修改控制器 LoginController.java，要求用户名和密码均为“zjut”，并且用

户类型为“管理员”才转到 loginSuccess.jsp 页面，否则转到 loginFailed 页面。

8、运行并观察修改后的结果，领会采用视图、模型和控制器分离表示逻辑和业务逻辑的优点。（四）实验要求

1、填写并上交实验报告，报告中应包括：

1. 运行结果截图；
2. 根据实验过程，查找相关资料，总结 MVC 设计模式的优缺点；
3. 实验收获及总结。

2、上交程序源代码，代码中应有相关注释。

## 三、扩展实验 ——JDBC 与 与 DAO 设计模式

（一）实验目的

1. 掌握数据库操作的基本 SQL 语句；
2. 掌握连接数据库的基本步骤；
3. 掌握利用 JDBC 操纵数据库的基本 API 和方法；
4. 掌握 DAO 设计模式；
5. 掌握在一个项目中集成 MVC 设计模式和 DAO 设计模式；
6. 进一步理解 MVC 设计模型中模型和现实中实体的对应关系；
7. 理解 DAO 设计模式中业务逻辑和数据访问逻辑相分离的优点。（二）基本知识与原理
8. JDBC（Java Data Base Connectivity）是用于执行 SQL 语句的 Java API，为

多种

关系数据库提供统一访问，它由一组用 Java 语言编写的类和接口组成。JDBC 提供了一种基准用于构建更高级的工具和接口，使数据库开发人员能够编写数据库应用程序。

1. DAO（Data Access Object）设计模式将所有对数据源的访问操作抽象封装在

一

个公共 API 接口中，其中定义了应用程序中将会用到的所有事务方法。当程序开发人员需要和数据源进行交互的时候可直接使用这个接口，无需操纵数据库。

（三）实验内容及步骤

1. 下载和安装 PostgreSQL 数据库(版本 8.3)，下载链接地址：

http://www.enterprisedb.com/products-services-training/pgdownload#windows

；

1. 创建一个用户名为 dbuser，密码为 dbpassword 的数据库登录角色，然后为该

角色创建一个名称为 myDB 的数据库，并在该数据库中创建一个名称为

usertable 的数据表，具体表结构如表 1-1 所示：表 1-1 usertable 数据表

字段名称类型中文含义

username Character varying(10) 登录用户名

password Character varying(10) 登录密码

type int 用户类型，1 表示管理员， 2 表示普通用户。

1. 在表 usertable 添加 3 条记录登录用户名密码用户类型

zjut zjut 1 admin admin 1

Temp temp 2

1. 修改工程 javaweb-prj1：添加 PostgreSQL 驱动程序库文件 postgresql-8.4-702

.jdbc4.jar 到工程中；

1. 修改工程 javaweb-prj1：创建数据库操作类 UserDAO.java，实现按用户名和

密码校验用户是否合法的功能：

package cn.edu.zjut.dao;

import java.sql.\*;import javax.sql.\*;import javax.naming.\*; import cn.edu.zjut.model.UserBean; public class UserDAO{

private static final String GET\_ONE\_SQL =

"SELECT \* FROM usertable WHERE username=? and password=?"; public UserDAO( ){ }

public Connection getConnection(){

Connection conn = null;

String driver = "org.postgresql.Driver";

String dburl = "jdbc:postgresql://127.0.0.1:5432/myDB";

String username = "dbuser"; //数据库登录用户名 String password = "dbpassword"; //数据库登录密码 try{

Class.forName(driver); //加载数据库驱动程序

conn = DriverManager.getConnection(dburl,username,password);

}catch( Exception e ){ e.printStackTrace(); } return conn; } public boolean searchUser(UserBean user){

// 按用户名和密码校验用户是否合法

Connection conn = null;

PreparedStatement pstmt = null;

ResultSet rst=null; try{

conn = getConnection();

pstmt = conn.prepareStatement(GET\_ONE\_SQL); pstmt.setString(1, user.getUsername()); pstmt.setString(2, user.getPassword()); rst = pstmt.executeQuery(); if(rst.next()){

return true;

}

}catch(SQLException se){ se.printStackTrace(); return false; }finally{ try{ pstmt.close(); conn.close();

}catch(SQLException se){ se.printStackTrace(); }

} return false;

}

}

6、修改工程 javaweb-prj1 中的控制器 LoginController.java：

1. 引入 UserDAO 类，具体代码如下：

……

import cn.edu.zjut.dao.UserDAO;

……

1. 重写 checkUser(UserBean user)方法：实现通过数据库表 usertable 的记录

来校验用户名和密码的合法性，具体修改代码如下所示： ……

boolean checkUser(UserBean user){

UserDAO ud=new UserDAO(); if( ud.searchUser(user) ) { return true;

}

return false;

}

……

1. 将 javaweb-prj1 工程重新部署在 Tomcat 服务器上；
2. 运行 login.jsp 页面，输入用户名和密码并运行，观察运行结果，如果出现运行错误，请尝试修正；
3. 对 javaweb-prj1 工程进行如下修改：
4. 修改登录成功的条件为用户名、密码和用户类型三者匹配；
5. 新增一个用户注册视图 register.jsp，用于普通用户的注册；
6. 修改 UserDAO 类，新增一个 insert(UserBean user)方法，用于插入一条记录

到 usertable 表中；

1. 运行并观察结果。

（四）实验要求

1、填写并上交实验报告，报告中应包括：

1. 运行结果截图；
2. 查找相关资料，总结 DAO 设计模型的优点；
3. 实验收获及总结。

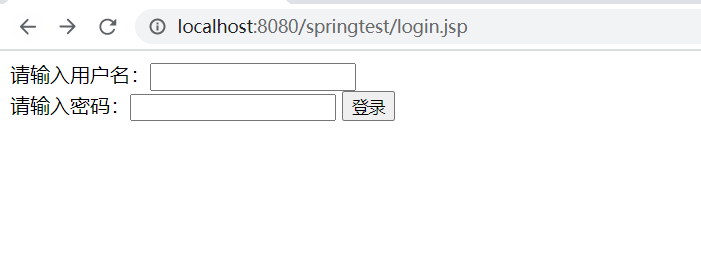
2、上交程序源代码，代码中应有相关注释。

# 二、实验结果及问题分析

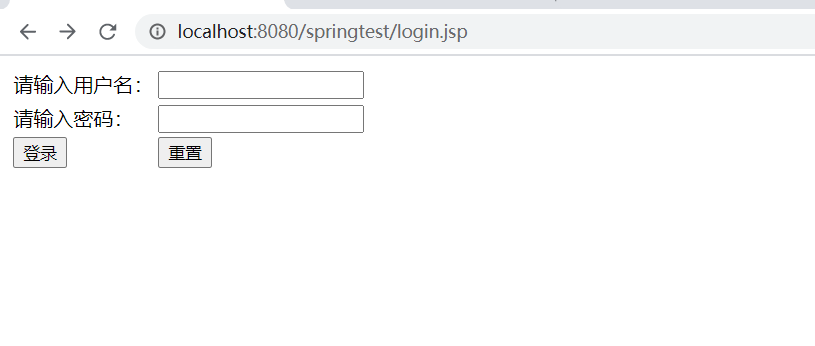
## 基础实验 ——Servlet 与 JSP 基础开发

（1）实验结果：

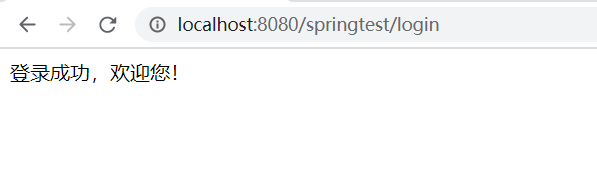
**login.jsp修改前**



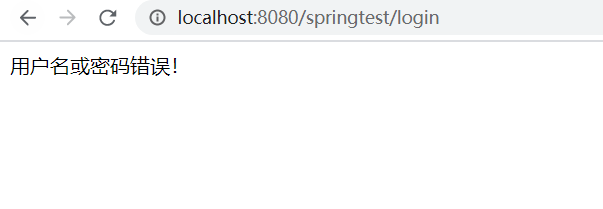
**login.jsp修改后**



**登录成功**



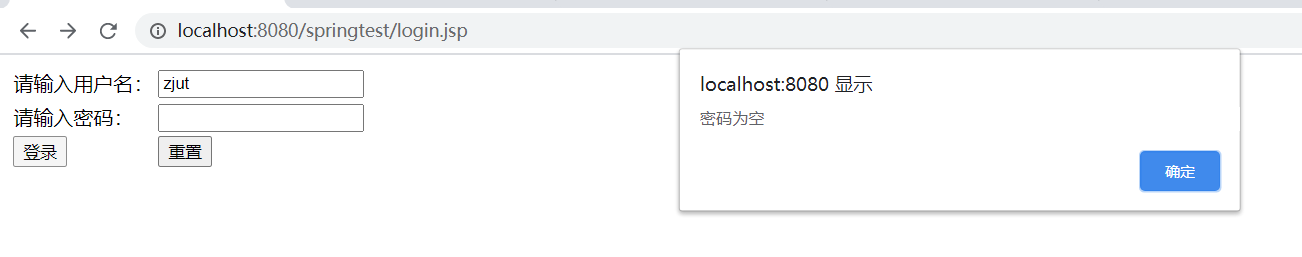
**登录失败**



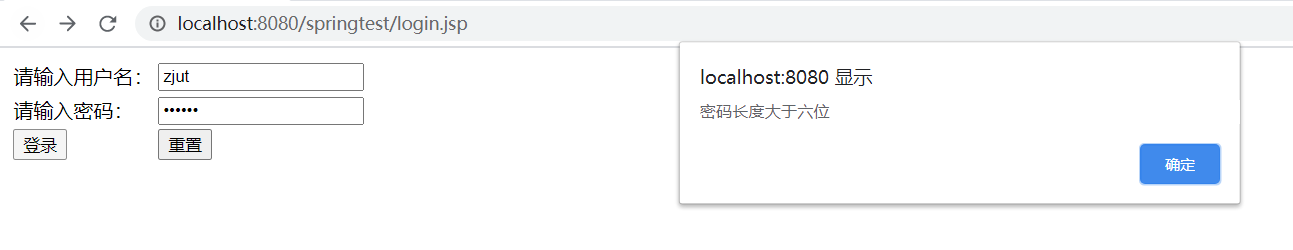
**用户名和密码输入不合规范：用户名为空**



**用户名和密码输入不合规范：密码为空**



**用户名和密码输入不合规范：密码超过6位**



（2）修改关键的代码

1.js表单判断

<script type="text/javascript">

function mycheck(){

if(document.form1.username.value==""){

alert("用户名为空");

return false;

}

if(document.form1.password.value==""){

alert("密码为空");

return false;

}

if(document.form1.password.value.length>=6){

alert("密码长度大于六位");

return false;

}

}

</script>

2.表格对齐

<table>

<tr>

<td>请输入用户名：</td>

<td><input type="text" name="username">

</tr>

<tr>

<td>请输入密码：</td>

<td><input type="password" name="password"></td>

</tr>

<tr>

<td> <input type="submit" value="登录"/> </td>

<td><input type="reset" value="重置" /></td>

</tr>

</table>

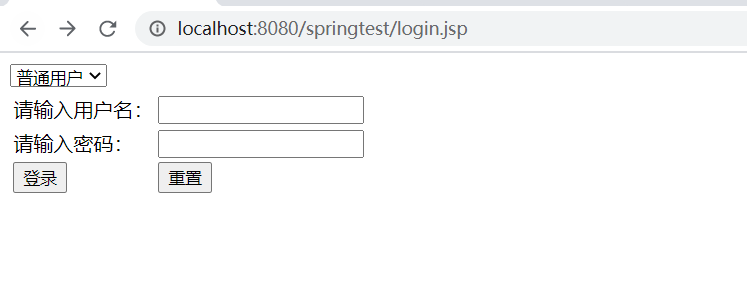
3.实验收获及问题

（1）尝试了onclick和onsubmit两种方式，一开始使用onclick但是发现没有反应，后来发现onclick反应是在按钮上的，onsubmit是在表单提交的时候才会有响应。

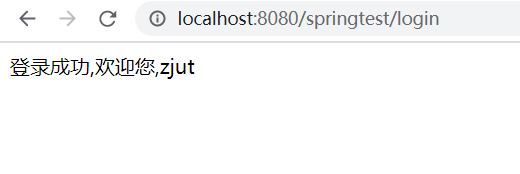
## 提高实验 ——Servlet 与 JSP 集成的 MVC 方案

（1）实验结果

**login.jsp修改后**

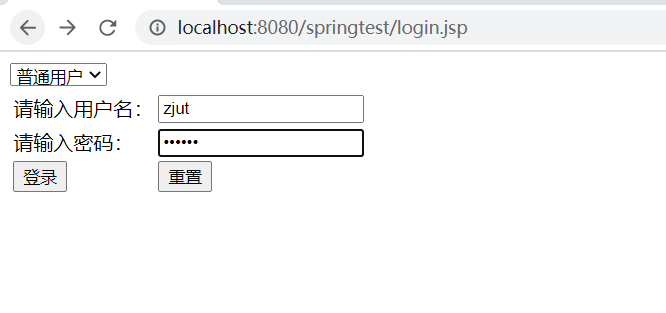


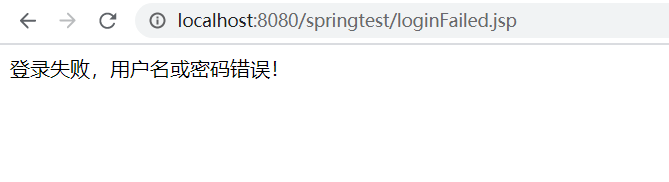
**登录成功**



**登录失败**







（2）总结MVC模式设计优缺点

优点：1.将业务逻辑和数据访问从表示层分离出来。2.让应用程序开发更加简单。3.维护更加方便。4. 各施其职，互不干涉。

缺点：1. 增加了系统结构和实现的复杂性。2. 视图与控制器间的过于紧密的连接。3. 视图对模型数据的低效率访问。

（3）关键代码

String type = request.getParameter("type");//获取表单中的用户类型

int type2 = Integer.parseInt(type);//将string数据转换成int

boolean checkUser(UserBean user){

if(user.getType==1&&"zjut".equals(user.getUsername())&&"zjut".equals(user.getPassword())) {

return true;

}else{

return false;

}

}

//判断他同时满足管理员身份、用户名和密码为zjut

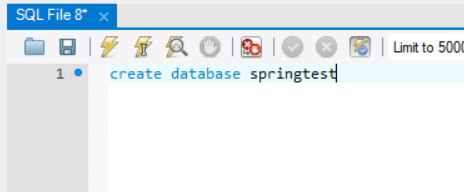
（4）实验收获及问题

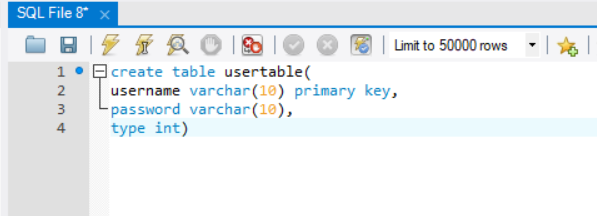
1.再一次的运用了MVC的开发模式，还是要注意一些细小的细节，努力做到一次将程序跑通。

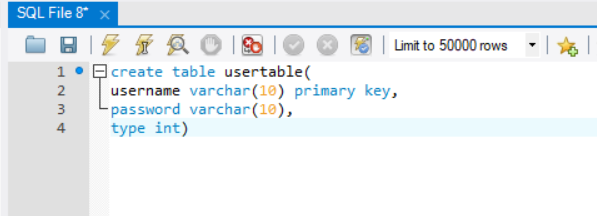
## 扩展实验 ——JDBC 与 与 DAO 设计模式

（1）实验结果

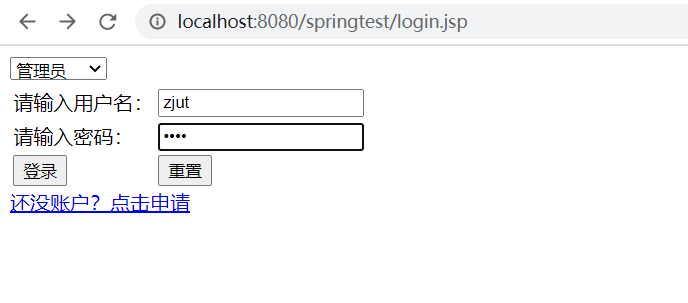
**数据库创建**

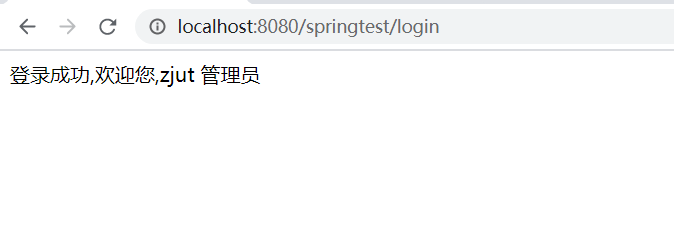




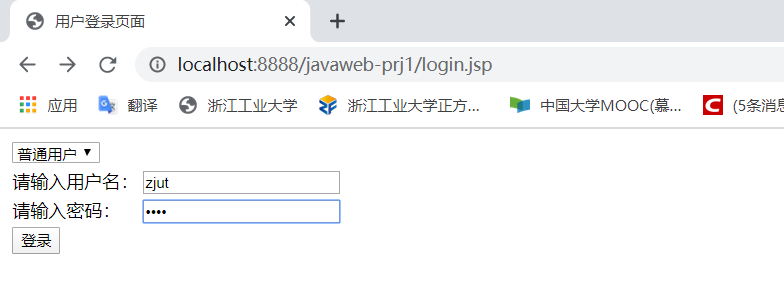


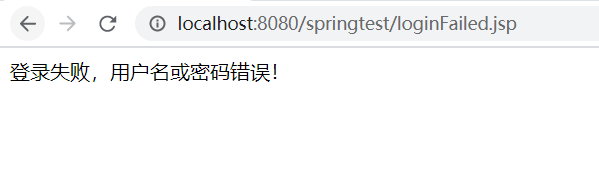
**用管理员身份登录成功**



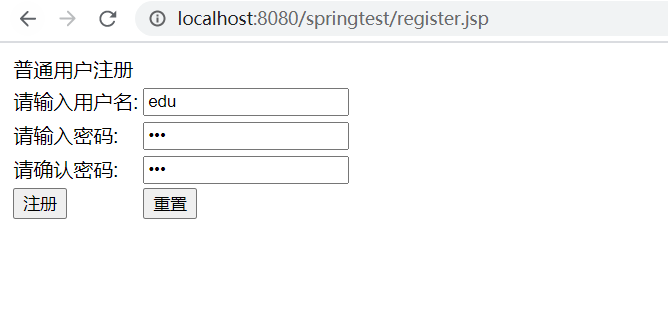


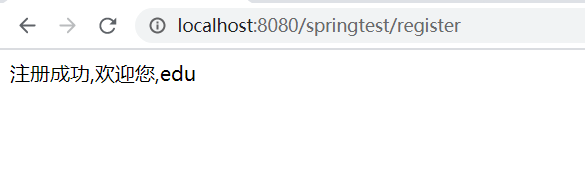
**登录失败**



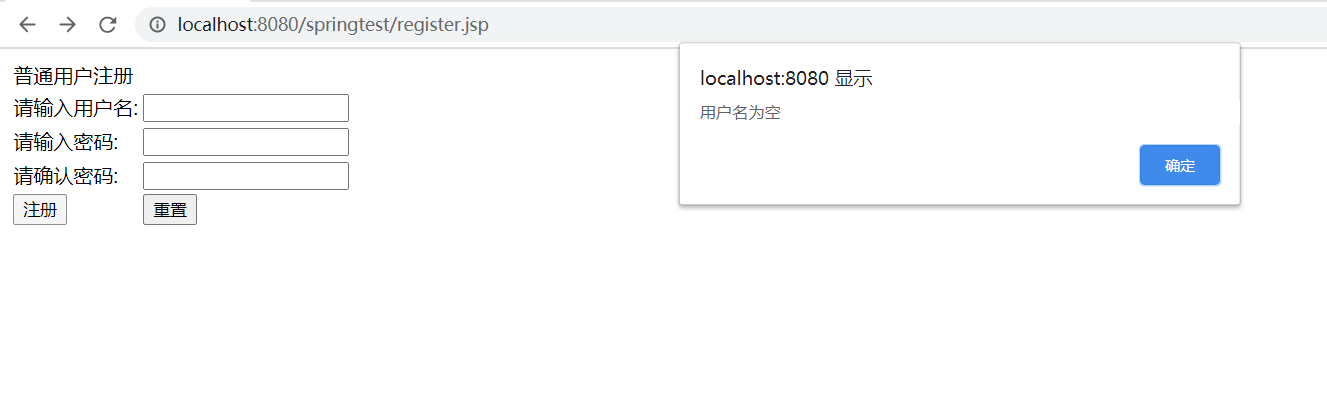


**普通用户注册**





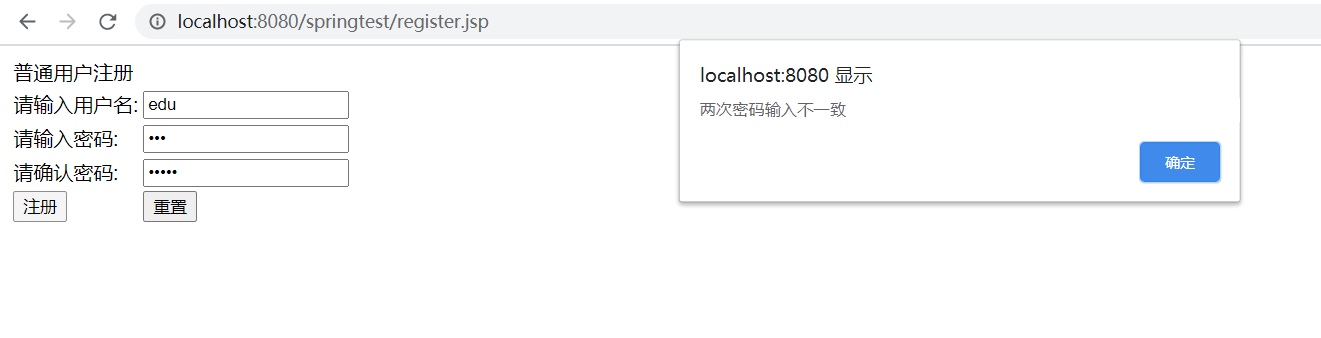
**注册失败各种情况：输入为空**



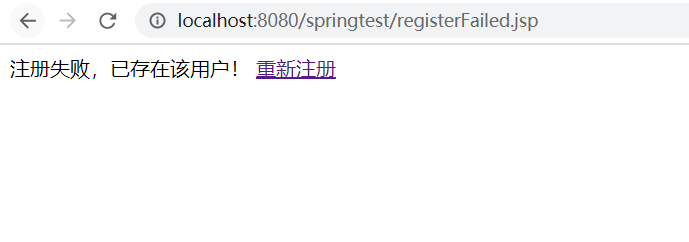
**注册失败各种情况：密码长度不符合**



**注册失败各种情况：密码不一致**



**注册失败各种情况：用户名已存在**



（2）总结DAO设计模型的优（摘自百度，觉得比较详细，就全部抄下来了）

1.DAO模式抽象出数据访问方式，业务逻辑层访问数据源时完全感觉不到数据源的存在。软件工厂中有一条很重要的法则：一个对象对其他对象的了解越少越好，了解越少就意味着依赖越少，可复用性越高。

2.DAO将数据访问集中在独立的一层，因为所有的数据访问都由DAO代理，这层独立的DAO将数据访问的实现和系统的其余部分剥离，将数据访问集中，使得系统更具可维护性。

3.DAO降低了业务逻辑层的复杂度。DAO管理复杂的数据访问，从而简化了业务逻辑层。所有与数据访问的实现有关的代码都不写在业务逻辑层里，业务逻辑层可集中处理业务逻辑，提高了代码的可读性和生产率。

4.DAO有助于提升系统的可移植性。DAO模式通过将数据访问划分为抽象层和实现层，分离数据使用和数据访问的实现细节。这意味着业务层与数据访问的底层细节无关，也就是说，可以在保持上层机构不变的情况下，通过切换底层实现来修改数据访问的具体机制，提高了系统的可复用性。

5.DAO组件依赖于数据库系统，提供数据库访问接口，只要数据库没有重构，DAO层通常无须改写。DAO层透明地分离数据库与业务逻辑层，业务逻辑层专注于业务逻辑的实现，而无须关心持久层访问的实现。

6.DAO模式的进一步改良。由于DAO层已实现所有的数据访问，业务逻辑层只需调用DAO接口，因此业务逻辑层要使用Facade模式包装DAO。为实现跨数据库平台移植，支持不同数据访问机制之间的可配置切换，需在DAO层引入Factorv模式、Proxy模式和Strategy模式，则可方便地在不同数据存储方式间切换。然而，采用DAO模式，系统在请求数据端和数据服务端之间增加一层，增加了系统的复杂度；新增加的一层需要额外的设计与实现，增加了工作量；还需引入工厂甚至抽象工厂，增加了设计的复杂度。总的来说，DAO将数据持久层与业务逻辑层分离，提高了软件的可扩展性，可维护性和可复用性。

（3）关键代码

try{

conn = getConnection();

Statement st = conn.createStatement();

String sql="INSERT INTO usertable VALUES('"+user.getUsername()+"','"+user.getPassword()+"','"+user.getType()+"')";

//添加用户的sql语句

st.executeUpdate(sql);

}

if(checkUser(user)){

//如果不存在该用户，则将注册的用户信息添加到表中，并转到注册成功的页面中

UserDao ud=new UserDao();

ud.insertUser(user);

request.setAttribute("USER", user);

RequestDispatcher dispatcher = request .getRequestDispatcher("/registerSuccess.jsp");

dispatcher.forward(request, response);

}else{

//如果存在该用户，则转到注册失败的页面中

response.sendRedirect("/springtest/registerFailed.jsp");

}

}

（4）实验收获及问题

1.用eclipse实现了数据库的连接。

2.简单的运用了sql语句，顺便复习了一下sql语句的使用。