



JavaEE 实验三实验报告

姓　　名　　 Mcrivers

班　　级　　 软件工程2102班

学　　号

提交日期　　 2023.11.12

目录

[一、基础实验——Struts2 框架搭建 1](#_Toc150714473)

[1. 1](#_Toc150714474)

[2. 总结 jsp 页面、Action 类、Service 类、JavaBean、Filter 和struts.xml 文件的作用，整理 Struts2 应用中从请求到响应的完整流程，思考并总结 Struts2 框架中 MVC 的体现 3](#_Toc150714475)

[3. 总结表单参数与 Action 属性的赋值关系 4](#_Toc150714476)

[4. 总结 Action 的 execute()方法的作用和特点 4](#_Toc150714477)

[5. 写出本实验中配置文件struts.xml 里各元素及其属性的作用 4](#_Toc150714478)

[6. 碰到的问题及解决方案或思考 5](#_Toc150714479)

[7. 实验收获及总结 5](#_Toc150714480)

[二、提高实验——Spring 和 Struts2 的整合 5](#_Toc150714481)

[1. 运行结果截图 5](#_Toc150714482)

[2.总结 Spring 整合 Struts2 框架的关键步骤 6](#_Toc150714483)

[3.总结本实验中的 UserAction 与基础实验中的写法关键区别 6](#_Toc150714484)

[4.总结配置文件applicationContext.xml 中bean 元素的 prototype 属性及其取值的含义 7](#_Toc150714485)

[5. 总结 web.xml 文件中添加监听器的目的 7](#_Toc150714486)

[6. 应用各种 Struts2 标签的关键代码，及运行结果或报错信息 7](#_Toc150714487)

[7. 总结 Struts2 中标签及其属性的作用和用法 8](#_Toc150714488)

[8. 碰到的问题及解决方案或思考 9](#_Toc150714489)

[9. 实验收获及总结 9](#_Toc150714490)

[三、扩展实验——Action 与 ActionSupport 9](#_Toc150714491)

[1.运行结果截图 9](#_Toc150714492)

[2. 总结 Action 自定义方法的四种调用和配置方式 13](#_Toc150714493)

[3.总结 validate()方法或validateXxx()方法的作用、使用时的要点或注意事项，总结在 JSP 页面中显示错误信息和提示信息的方法 14](#_Toc150714494)

[4. 总结校验器校验的使用和配置方法，结合相应案例将其记录下来 15](#_Toc150714495)

[5. 总结 Struts2 中常用的内置类型转换器及其使用方法 16](#_Toc150714496)

[6. 碰到的问题及解决方案或思考 17](#_Toc150714497)

[7. 实验收获及总结 17](#_Toc150714498)

# 一、基础实验——Struts2 框架搭建

## 1.

**运行结果截图：**

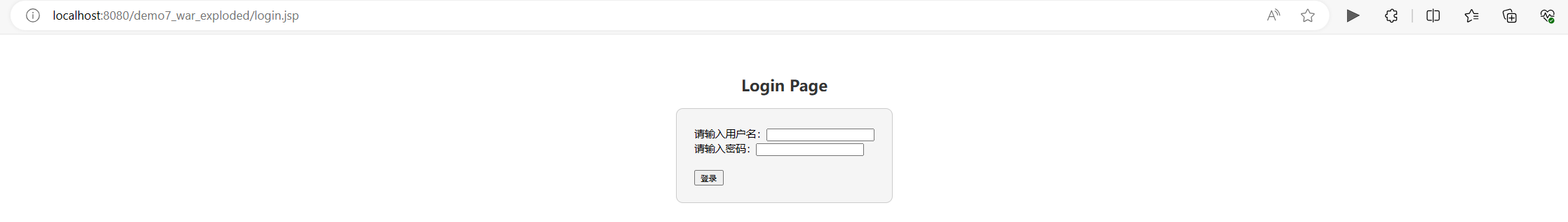


图1.1 login.jsp登录界面

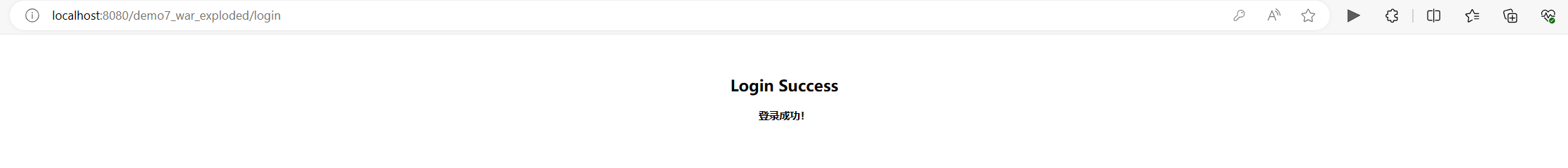


图1.2 loginSuccess.jsp登录成功页面

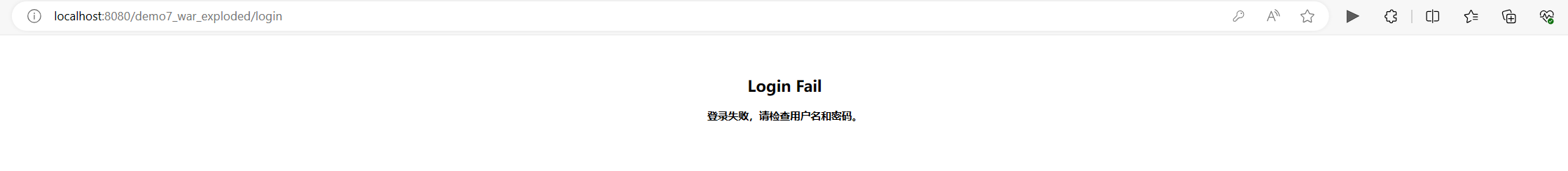


图1.3 loginFail.jsp登录失败页面

**修改代码：**

①将login.jsp中name="loginUser.username"改为name="loginUser.usernnnnname"

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

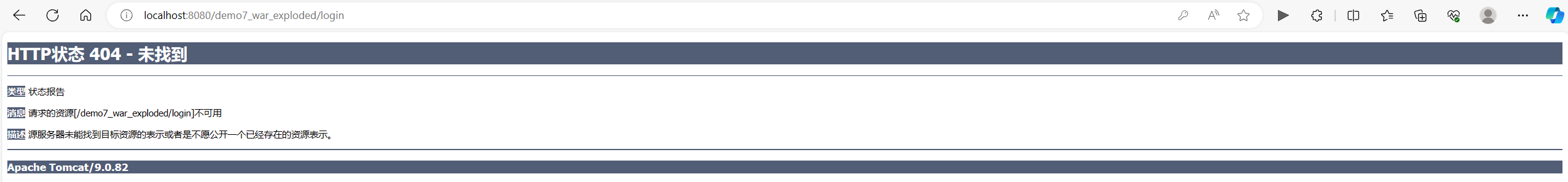


图1.4 报错信息

②将UserBean.java中的username和password改为usernnnnname和passwwwwword

private String usernnnnname = "";  
private String passwwwwword = "";

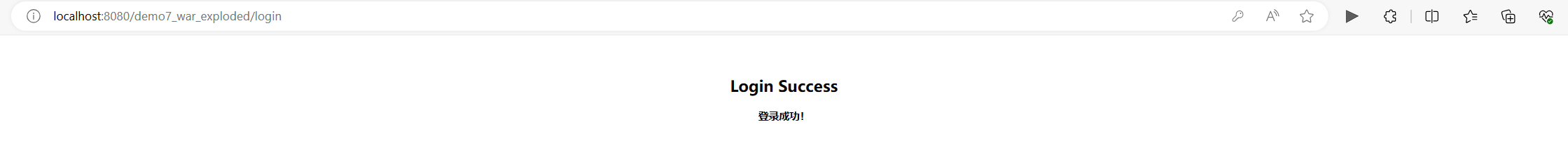


图1.5 运行结果（正常运行）

③将UserAction.java中的loginUser改为loginnnnnUser

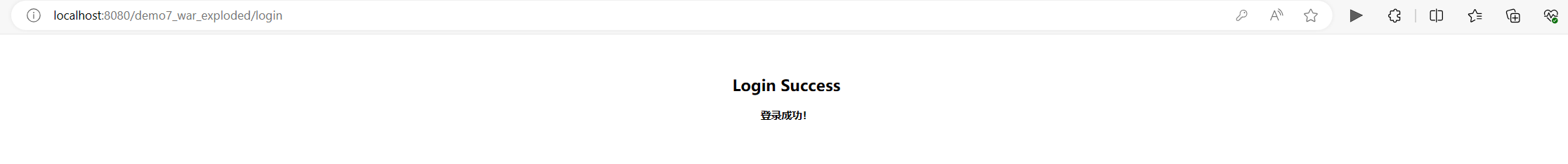


图1.6 运行结果（正常运行）

④将UserAction.java中的getLoginUser和setLoginUser改为getLoginnnnnUser和setLoginnnnnUser

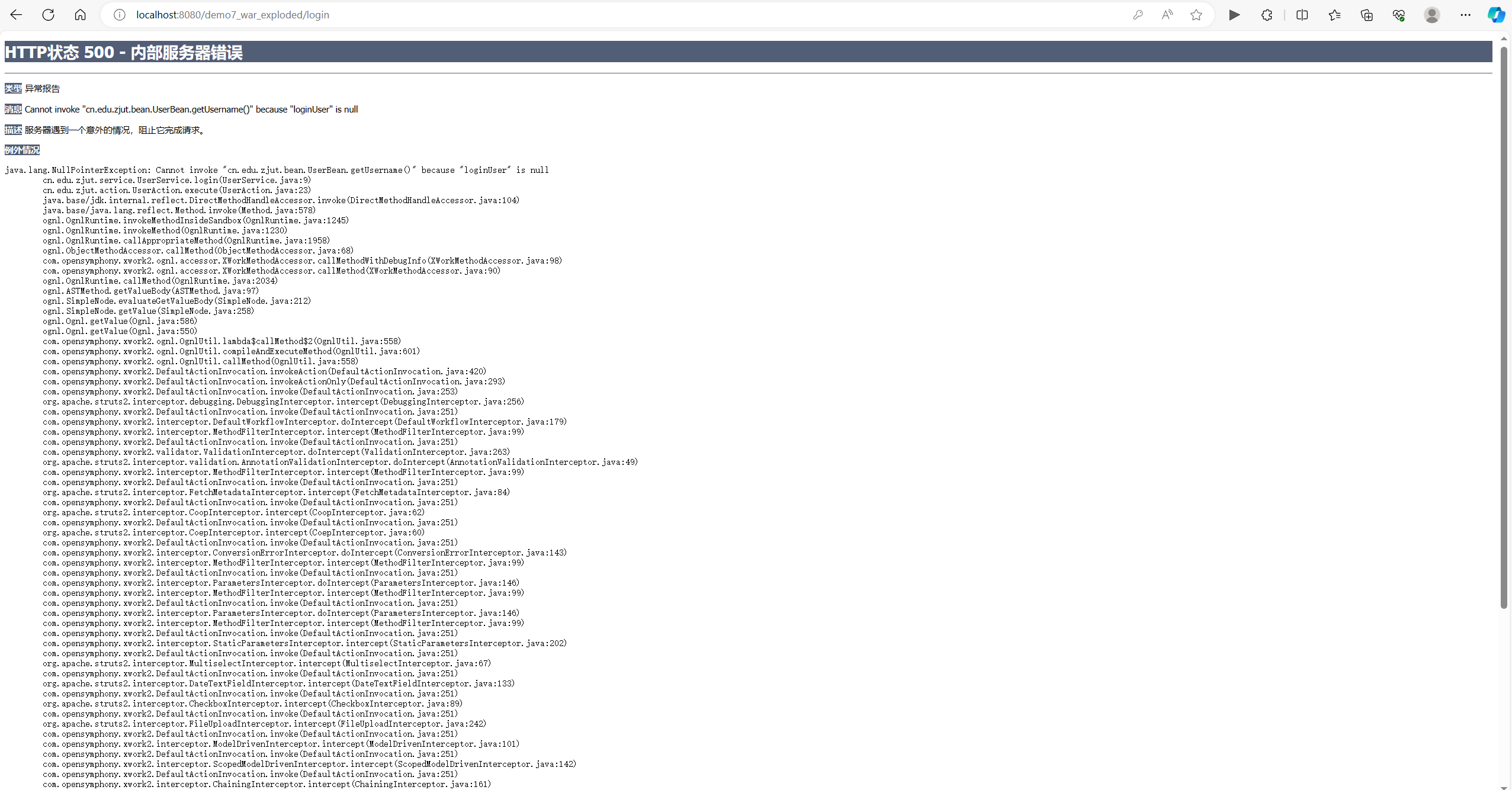


图1.7 报错情况

⑤将struts.xml中的name="login"改成name="loginnnnn"

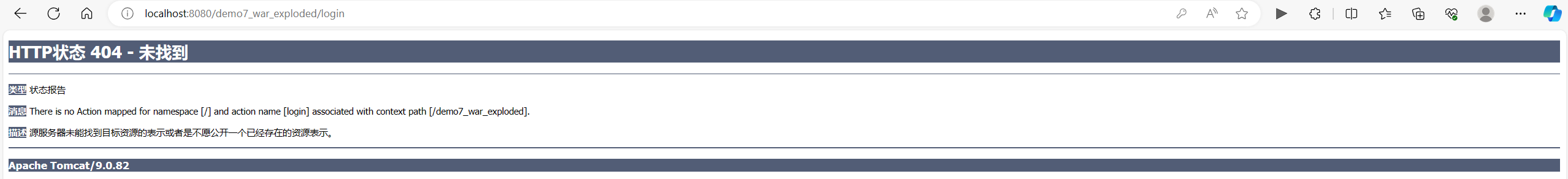


图1.8 报错情况

⑥将struts.xml中的class="cn.edu.zjut.action.UserAction"改成

class="cn.edu.zjut.action.Useraction"



图 1.9 报错情况

⑦将struts.xml中的name="success"和name="fail"互换

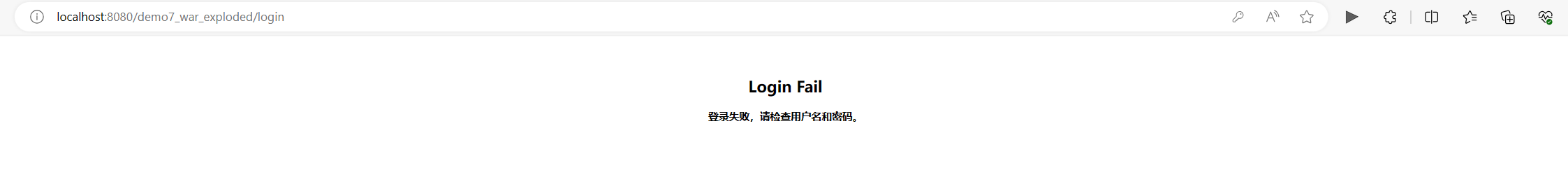


图1.10 用户名和密码匹配时

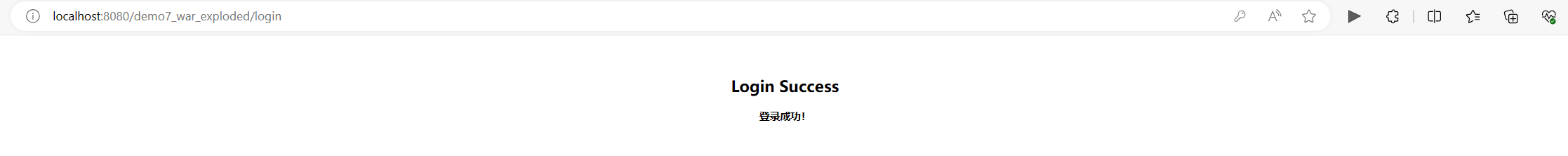


图 1.11 用户名和密码不匹配时

## 2. 总结 jsp 页面、Action 类、Service 类、JavaBean、Filter 和struts.xml 文件的作用，整理 Struts2 应用中从请求到响应的完整流程，思考并总结 Struts2 框架中 MVC 的体现

**文件的作用：**

**JSP 页面：**提供用户界面，展示数据和接收用户输入。

**Action 类：**

①处理业务逻辑和用户请求。

②包含用于接收和处理用户提交数据的属性。

**Service 类：**

①处理具体的业务逻辑，与数据交互。

②在Action中被调用，执行具体的业务操作。

**JavaBean：**存储数据，比如用户数据

**Filter：**拦截所有请求，将其传递给Struts2框架处理。

**struts.xml 文件：**

①定义Struts2应用的配置信息，包括Action的映射、结果页面等。

②定义了请求到Action的映射，以及Action执行后返回的结果。

**完整流程：**

①请求到达应用

②Filter拦截请求，将其传递给Struts2框架

③Struts2框架初始化

④Struts2根据请求信息在struts.xml中查找对应的Action配置。

⑤根据配置实例化相应的Action类，执行Action的构造方法。

⑥Struts2尝试将请求参数绑定到Action的属性。

⑦调用Action的execute()方法。

⑧根据Action执行的结果，查找struts.xml中配置的对应视图。

⑨渲染结果页面，将数据呈现给用户。

⑩在请求处理完成后，销毁Struts2框架的相关资源。

**MVC体现：**

Model： JavaBean和Service类负责数据的存储和业务逻辑的处理。

View： JSP页面负责展示数据和接收用户输入。

Controller： Action类充当控制器的角色，处理用户请求，调用相应的业务逻辑，并决定返回哪个视图。

## 3. 总结表单参数与 Action 属性的赋值关系

表单参数的命名应该与Action中的属性名一致，表单中的参数名应与Action类中的属性名一致，这样Struts2框架会尝试将请求参数与Action类中的属性进行匹配。比如，表单中的loginUser.username和loginUser.password会被赋值给UserAction类中的username属性和password属性。

## 4. 总结 Action 的 execute()方法的作用和特点

**作用：**用于处理请求的业务逻辑，可以在该方法中编写与请求相关的业务逻辑代码，比如处理表单提交等

特点： ①自动调用

②返回值处理，比如返回值为"success"、"error"等字符串，根据这些字符串导航到相应的结果页面

## 5. 写出本实验中配置文件struts.xml 里各元素及其属性的作用

**<struts>元素：**

作用： 定义整个Struts2配置文件的根元素。

**<package>元素：**

作用： 定义一个Struts2的包，用于组织和管理相关的Action、拦截器、结果等。

属性：

name：指定包的名称，用于在配置文件中唯一标识该包。

extends：指定该包继承自哪个包，可以继承自Struts2的默认包struts-default。

**<action>元素：**

作用： 定义一个具体的Action。

属性：

name：指定Action的名称，用于在配置文件中唯一标识该Action。

class：指定Action类的完整类名，表示该Action的实现类。

**<result>元素：**

作用： 定义Action执行后的结果视图。

属性：

name：指定结果的名称，用于在Action中进行返回值匹配。

## 6. 碰到的问题及解决方案或思考

将login.jsp中name="loginUser.username"改为name="loginUser.usernnnnname"时项目无法正确运行；如果将UserAction类中的setter方法和getter方法对应地改为setLoginnnnnUser()和getLoginnnnnUser()，则Struts框架可以正确进行数据绑定，从而使程序可以正常运行，这是符合预期的。**但是，如果显式定义UserAction类的无参构造方法，则即使不更改setter方法和getter方法，程序也能正常运行。推测是默认的无参构造方法没有对loginUser进行定义，从而使Struts框架只能根据表单参数进行数据绑定；而显式定义的无参构造方法对loginUser进行了显式定义，从而使数据绑定可以正常进行。**

## 7. 实验收获及总结

在这个实验中，我学到了如何在Struts2项目中配置和使用关键的文件和组件，包括struts.xml配置文件、Action类、JavaBean、JSP页面和Filter。同时，我学习到了请求到响应的完整流程、数据绑定和业务逻辑处理和Struts2的MVC体系结构等。这对我对JavaEE的学习有深刻的影响。

# 二、提高实验——Spring 和 Struts2 的整合

## 1. 运行结果截图



图2.1 login.jsp登录页面



图2.2 register.jsp注册页面



图2.3 regSuccess.jsp注册成功页面



图2.4 regFail.jsp注册失败页面

## 2.总结 Spring 整合 Struts2 框架的关键步骤

①导入一系列有关的jar包

②业务逻辑类的包中创建接口，比如 IUserService

③在业务逻辑类中实现上述接口，例如 UserService 类实现 IUserService 接口，编写相应的业务逻辑

④在 UserAction 类中，通过提供相应的 setter 方法，注入业务逻辑类的实例

⑤创建 Spring 配置文件applicationContext.xml，配置业务逻辑类的 bean，并在 UserAction 类中通过 Spring 的 IoC 容器注入相关的 bean

⑥在struts.xml中，将 <action> 元素的 class 属性设置为对应 Spring 配置文件中的 bean 的 id

⑦在 web.xml 文件中配置 Spring 的监听器

⑧在 JSP 页面中使用 Struts2 的标签库，例如 <s:form>、<s:textfield> 等

⑨在 UserAction 类中添加注册逻辑，修改业务逻辑类和相应的 jsp 页面

## 3.总结本实验中的 UserAction 与基础实验中的写法关键区别

①UserAction 类中的 userService 属性的类型是 IUserService 接口，而不是具体的实现类 UserService

②使用了 Spring 框架，通过 setUserService 方法实现对 userService 对象的依赖注入

## 4.总结配置文件applicationContext.xml 中bean 元素的 prototype 属性及其取值的含义

当作用域设置为 prototype 时，Spring 每次从容器中获取 bean 时都会创建一个新的实例，而不是返回容器中现有的实例

## 5. 总结 web.xml 文件中添加监听器的目的

这个监听器的目的是启动 Spring 容器，加载指定的 Spring 上下文，从而确保 Spring 容器在应用启动时被正确初始化。这样Spring 中定义的 bean、配置等都可以在整个应用中被访问和使用。

## 6. 应用各种 Struts2 标签的关键代码，及运行结果或报错信息

<**s:form** action="register" method="post">  
 <**s:textfield** name="loginUser.username" label="请输入用户名"/>  
 <**s:password** name="loginUser.password" label="请输入密码"/>  
 <**s:password** name="loginUser.Repassword" label="确认密码"/>  
 <**s:textfield** name="loginUser.realName" label="真实姓名"/>  
 <**s:radio** name="loginUser.sex" list="#{1 : '男', 0 : '女'}" label="请选择性别"/>  
 <**s:textfield** name="loginUser.birthday" label="请输入生日(yyyy-MM-dd)">  
 <**s:param** name="value">  
 <**s:date** name="loginUser.birthday" format="yyyy-MM-dd"/>  
 </**s:param**>  
 </**s:textfield**>  
 <**s:textfield** name="loginUser.address" label="请输入联系地址"/>  
 <**s:textfield** name="loginUser.phoneNumber" label="请输入联系电话"/>  
 <**s:textfield** name="loginUser.email" label="请输入电子邮箱"/>  
 <!-- 其他字段根据需要添加 -->  
  
 <**s:submit** value="注册"/>  
 <**s:reset** value="重置"/>  
</**s:form**>



图2.5 运行结果

## 7. 总结 Struts2 中标签及其属性的作用和用法

**s:form 标签：**

作用： 用于生成 HTML 表单。

常用属性：

action：指定表单提交的目标 URL。

method：指定表单提交的 HTTP 方法，可以是 "post" 或 "get"。

其他属性：可以设置表单的其他属性，如 enctype、onsubmit 等。

**s:textfield 标签：**

作用： 用于生成文本框。

常用属性：

name：指定文本框的名称。

label：指定文本框的标签。

其他属性：可以设置文本框的其他属性，如 size、maxlength、onfocus 等。

**s:password 标签：**

作用： 用于生成密码框。

常用属性：

name：指定密码框的名称。

label：指定密码框的标签。

其他属性：可以设置密码框的其他属性，如 size、maxlength、onfocus 等。

**s:submit 和 s:reset 标签：**

作用： 用于生成提交按钮和重置按钮。

常用属性：

value：指定按钮上显示的文本。

align：指定按钮的对齐方式。

**s:radio 标签：**

作用： 用于生成单选按钮。

常用属性：

name：指定单选按钮的名称。

list：指定单选按钮的选项。

**s:date 标签：**

作用： 用于格式化日期。

常用属性：

name：指定日期的属性名。

format：指定日期的显示格式。

**s:property 标签：**

作用： 用于输出属性值。

常用属性：

value：指定要输出的属性名。

## 8. 碰到的问题及解决方案或思考

**问题：**在UserService中判断用户名username、密码password、确认密码repassword三者是否相等时，发现判断结果始终为假，导致注册操作始终不能成功。

**分析：**在分析代码后，发现时register.jsp中将name="loginUser.Repassword"写成了name="Repassword"，导致Struts2框架无法进行数据绑定，从而无法将表单参数赋值给UserAction类中的loginUser属性，导致判断用户名、密码、确认密码的结果始终为假。

**解决方案：**将name="Repassword" name="改成loginUser.Repassword"。

## 9. 实验收获及总结

通过完成本实验，我了解了 Spring 框架与 Struts2 框架整合的基本步骤，学习了 Struts2 常用标签的基本使用方法。同时，我进一步深入理解了控制反转和依赖注入；通过修改 Spring 配置文件 applicationContext.xml 和 Struts2 配置文件 struts.xml，我了解了配置文件对整合过程的影响。

# 三、扩展实验——Action 与 ActionSupport

## 1.运行结果截图

**3. 将 struts-prj1 重新部署，通过浏览器访问 login.jsp 与register.jsp 页面**



图3.1 login.jsp页面



图3.2 register.jsp页面

**4、查找相关资料，尝试使用 Action 自定义方法的其它三种调用和配置方式**

**动态方法调用方式（DMI）**

关键配置：

Struts.xml:

<constant name="struts.enable.DynamicMethodInvocation" value="true"/>  
<package name="strutsBean" extends="struts-default" namespace="/">  
 <action name="login" class="cn.edu.zjut.action.UserAction">  
 <result name="success">/loginSuccess.jsp</result>  
 <result name="fail">/loginFail.jsp</result>  
 </action>  
 <action name="register" class="cn.edu.zjut.action.UserAction">  
 <result name="regsuccess">/regSuccess.jsp</result>  
 <result name="regfail">/regFail.jsp</result>  
 </action>  
</package>



图3.2 URL配置



图3.3 URL配置



图3.4登录页面

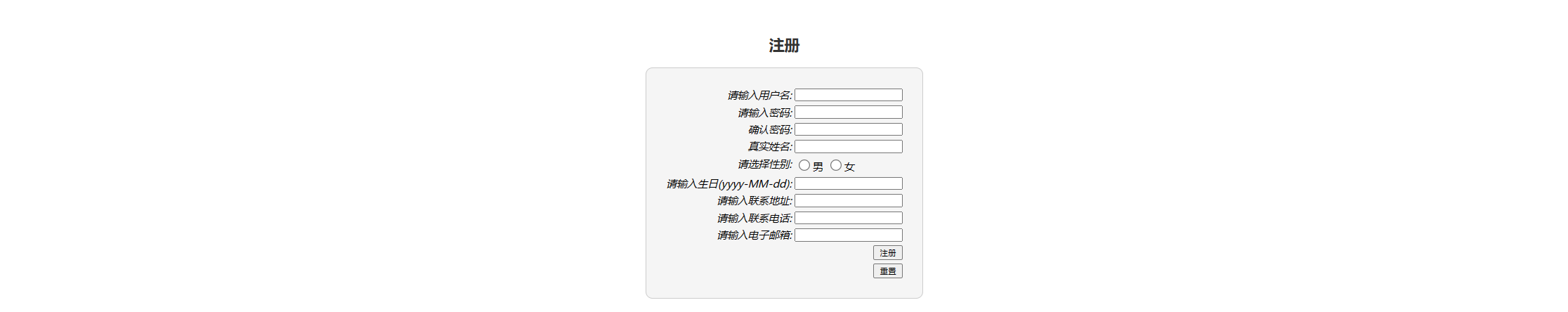


图3.5注册页面

提交按钮的 method 属性：



图3.6注册页面

**<s:submit value="注册" method="register"/>**

**7、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问login.jsp 页面，观察并记录运行结果**

观察到，当表单为空时点击提交，页面会“保持不变“，实际上是重新导航到了login.jsp



图3.7 登录前

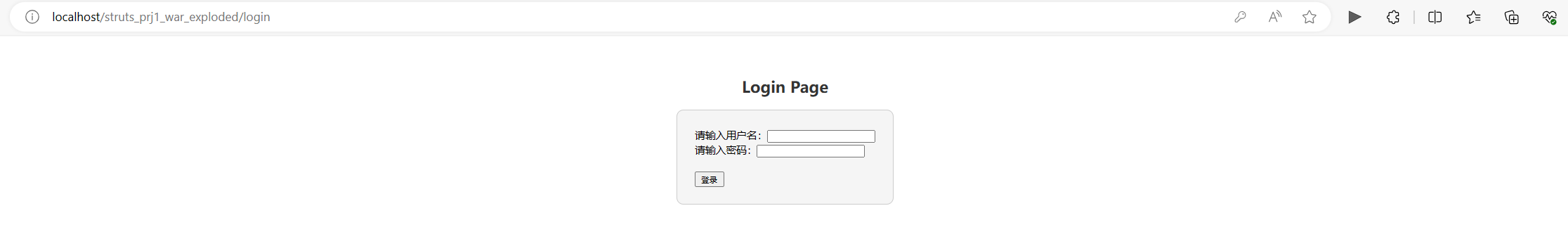


图3.8 登录后（在表单为空的情况下）

**8、修改 login.jsp 页面，在表单前增加 fielderror 标签：，再通过浏览器访问 login.jsp 页面，观察并记录运行结果**

观察到，在表单为空时点击提交，系统会提示输入正确的用户名

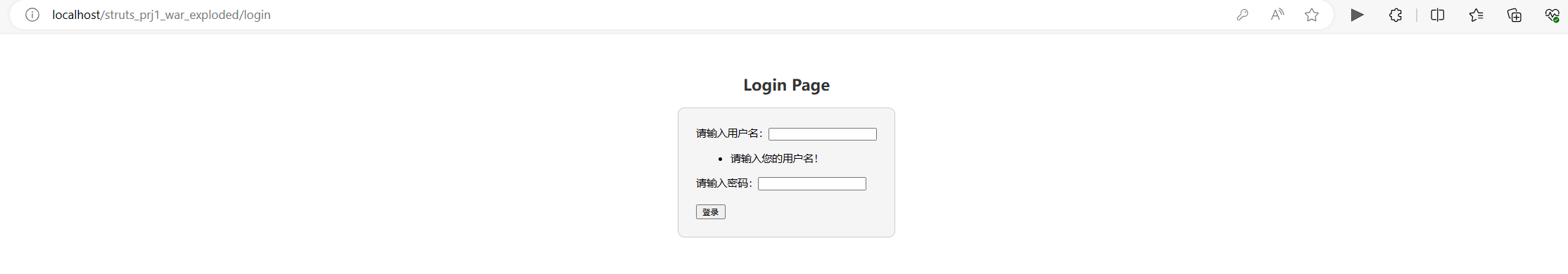


图3.9 登录后（在表单为空的情况下）

12、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问login.jsp 页面，观察并记录运行结果

观察到，登录失败时会在登录界面提示“用户名或密码错误，请重新输入！”，登录成功时，会在登录成功界面提示“登录成功！！！！！！”

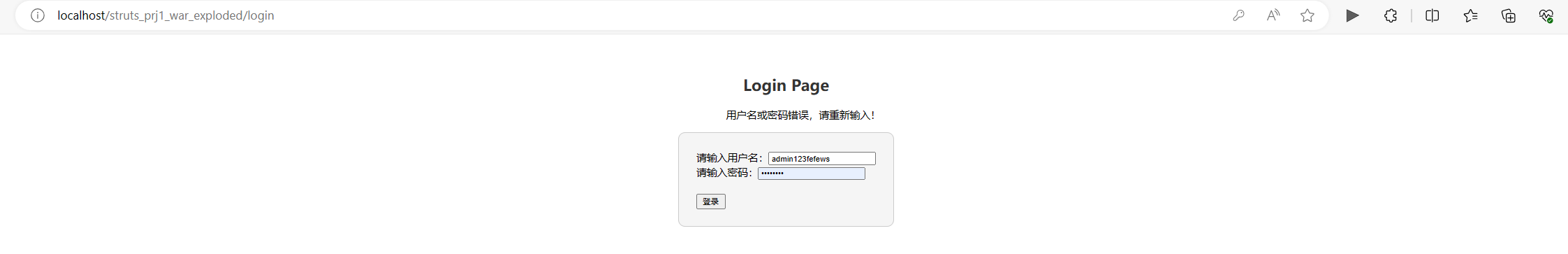


图3.10 登录失败时

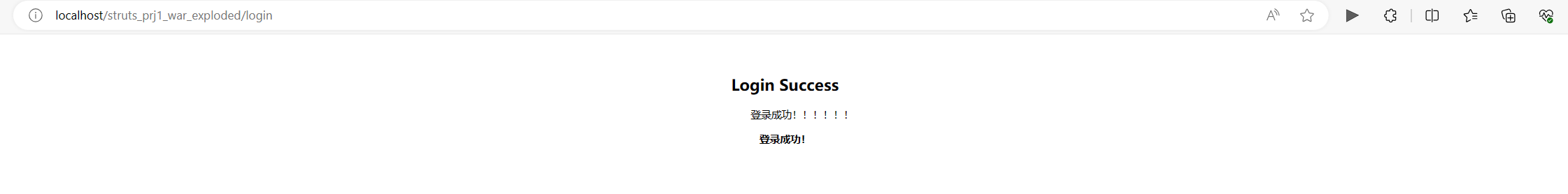


图3.11 登录失败时

**14、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问login.jsp 页面，观察并记录运行结果**

观察到，输入框下方出现了额外的“用户名不能为空”和“密码不能为空”字样。



图3.12 当用户名密码全为空时点击登录

**17、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问register.jsp 页面，当用户输入的生日不合法时，观察并记录运行结果**

观察到，如果输入非法日期，页面中会出现错误提示。



图3.13 输入非法日期时的结果

**19、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问register.jsp 页面，当用户输入的生日不合法时，观察并记录运行结果**



图3.14 输入非法日期时的结果

**21、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问register.jsp 页面，观察并记录运行结果**



图3.14 输入错误邮箱格式时的结果

## 2. 总结 Action 自定义方法的四种调用和配置方式

①通过struts.xml配置method属性：

在struts.xml文件中，通过method属性指定要调用的自定义方法。例如：

<action name="example" class="com.example.ExampleAction" method="customMethod">  
<result>/success.jsp</result>  
</action>

②使用动态方法调用方式（DMI）：

通过请求参数指定方法名。例如：

http://example.com/actionName!customMethod.action

customMethod是自定义的方法名

③使用提交按钮的method属性：

在表单的提交按钮中，可以使用method属性指定要调用的方法。例如：

<s:submit value="Submit" method="customMethod" />

④使用通配符配置Action：

通过通配符配置可以为某个Action指定一组共享的配置，包括调用的方法。例如：

<action name="example\_\*" class="com.example.ExampleAction" method="{1}">  
<result>/success.jsp</result>  
</action>

## 3.总结 validate()方法或validateXxx()方法的作用、使用时的要点或注意事项，总结在 JSP 页面中显示错误信息和提示信息的方法

**validate() 方法或 validateXxx() 方法的作用和使用要点：**

1. 作用：

validate() 方法： 这是一个在执行 Action 方法之前由 Struts 2 调用的生命周期方法。它用于执行对请求参数的验证逻辑，并通过 addActionError(), addFieldError(), 或 addActionMessage() 等方法向错误集合或消息集合中添加错误信息。如果有错误信息存在，Struts 2 将阻止进一步的 Action 方法调用，并将控制返回给结果页面。

validateXxx() 方法： validateXxx() 是一种约定的方式，其中 Xxx 通常是 Action 方法的名称，用于执行特定方法的验证逻辑。例如，如果有一个名为 execute() 的 Action 方法，则可以定义 validateExecute() 方法来执行与该方法相关的验证。

2. 使用要点或注意事项：

命名规范： 如果定义了 validateXxx() 方法，Xxx 应该是相应的 Action 方法的名称。例如，对于 execute() 方法，对应的验证方法是 validateExecute()。

参数校验： 在 validate() 或 validateXxx() 方法中，可以使用 addFieldError() 或 addActionError() 添加字段级别或 Action 级别的错误信息，以进行参数的验证。

验证失败处理： 如果验证失败，Struts 2 将不会执行 Action 方法，并将控制返回到结果页面。在结果页面中，可以使用 <s:fielderror>, <s:actionerror>, <s:actionmessage> 等标签显示错误信息。

**在 JSP 页面中显示错误信息和提示信息的方法：**

1. 显示字段级错误信息（Field Errors）：

使用 <s:fielderror> 标签：这将显示特定字段的错误信息，其中 fieldName 是字段的名称。

2. 显示 Action 级错误信息（Action Errors）：

使用 <s:actionerror> 标签：这将显示与整个 Action 关联的错误信息。

3. 显示提示信息（Action Messages）：

使用 <s:actionmessage> 标签：这将显示与整个 Action 关联的提示信息。

4. 显示全局错误信息（Global Errors）：

使用 <s:actionerror> 标签，但不指定 fieldName：这将显示字段级别和 Action 级别的所有错误信息。

**关键代码：**

public void validateLogin()  
{  
 String username = loginUser.getUsername();  
 String pwd = loginUser.getPassword();  
 if (username == null || username.equals(""))  
 {  
 this.addFieldError("loginUser.username", "请输入您的用户名！");  
 }  
 if (pwd == null || pwd.equals(""))  
 {  
 this.addFieldError("loginUser.password", "请输入您的密码！");  
 }  
}

<**s:actionerror**/>  
<form action="login" method="post">  
 请输入用户名：<input name="loginUser.username" type="text"><br>  
 <**s:fielderror** fieldName="loginUser.username" />  
 请输入密码：<input name="loginUser.password" type="password">  
 <**s:fielderror** fieldName="loginUser.password" />  
 <br><br>  
 <input type="submit" value="登录">  
</form>

<h2>Login Success</h2>  
<**s:actionmessage**/>  
<p>登录成功！</p>

## 4. 总结校验器校验的使用和配置方法，结合相应案例将其记录下来

**校验器的使用方法：**

通过创建 validation.xml 文件，可以定义校验规则，然后在 struts.xml 中配置对应的Action。例如：

<validators>  
<field name="username">  
<field-validator type="requiredstring">  
<message>用户名不能为空</message>  
</field-validator>  
<field-validator type="stringlength">  
<param name="minLength">3</param>  
<param name="maxLength">20</param>  
<message>用户名长度必须在3到20之间</message>  
</field-validator>  
</field>  
<field name="password">  
<field-validator type="requiredstring">  
<message>密码不能为空</message>  
</field-validator>  
<field-validator type="stringlength">  
<param name="minLength">6</param>  
<param name="maxLength">20</param>  
<message>密码长度必须在6到20之间</message>  
</field-validator>  
</field>  
</validators>

**2.校验器的配置方法**

a. 常用的校验器类型：

requiredstring： 检查字符串是否非空。

stringlength： 检查字符串的长度是否在指定范围内。

regex： 使用正则表达式检查字符串格式。

email： 检查字符串是否符合邮箱地址格式。

int，long，double： 分别检查整数、长整数、浮点数格式。

b. 校验器的参数配置：

校验器通常支持一系列的参数，例如 fieldName（字段名）、message（错误消息）、minLength、maxLength、expression（正则表达式）等，具体配置取决于校验器类型。

c. 全局校验器配置：

可以在 struts.xml 文件中全局配置校验器，以便在多个Action中共享相同的校验规则。

## 5. 总结 Struts2 中常用的内置类型转换器及其使用方法

**String 类型转换器：**

用途： 默认的类型转换器，将请求参数直接转换为String类型。

使用方法： 通常无需额外配置，Struts 2会自动使用String类型转换器。

**基本数据类型转换器：**

用途： 将请求参数转换为基本数据类型，如int、long、double等。

使用方法： 无需额外配置，Struts 2会根据属性类型自动选择合适的基本数据类型转换器。

**日期类型转换器：**

用途： 将字符串转换为Date类型。

使用方法： 在struts.xml中配置全局日期格式，或在Action属性上使用

## 6. 碰到的问题及解决方案或思考

问题：在配置登录界面的校验器时，发现用户名即使为空也没有任何错误提示，而密码为空却有错误提示

分析：发现时实验文档中的命名方式和我的命名方式不统一，实验文档中把用户名命名成loginUser.account，而我把用户名命名成loginUser.username

解决方案：把loginUser.username改成loginUser.account

## 7. 实验收获及总结

在本次实验中，我学会了Action 类的自定义方法、Action 接口与 ActionSupport 类、数据校验和校验器、显示错误信息和提示信息以及内置类型转换器。这让我对Struts2框架有了更深刻的认识。