浙江工业大学

JavaEE技术实验报告

实验名称：Struts2基础应用——基于Struts2框架的用户登录模块

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15日

# 一、实验主要步骤

**一、基础实验——Struts2 框架搭建**

（一）实验目的

1、掌握 Struts2 应用的基本开发步骤和常规配置；

2、观察表单参数与 Action 属性的赋值关系，观察 Action 的 execute()方法及其返

回值，并能够正确应用；

3、观察配置文件 struts.xml 中的主要元素及属性，并能够正确应用；

4、理解 Struts2 框架中 MVC 设计模式的体现，理解 Action，FilterDispatcher，

struts.xml 的主要作用，并能够正确应用。

（二）基本知识与原理

1、Struts2 是从 WebWork 框架上发展起来的 MVC 框架；

2、FilterDispatcher 是 Struts2 中的核心控制器，客户端对服务器端的请求将被

FilterDispatcher 过滤；若请求需要调用某个 Action，则框架将根据配置文件

struts.xml，找到需要调用的 Action 类；

3、Action 类是一个符合一定命名规范的 JavaSE 类，作为业务控制器使用；Action

中的 execute()方法用于调用 Model 层的业务逻辑类，并根据返回结果决定页

面导航；

4、若 Action 类中需要使用表单提交的请求参数，那么必须在 Action 类中声明与

表单域的名字对应的变量，并为变量提供 getters/setters 方法；

5、Action 类需要在 struts.xml 中进行配置才能使用；

6、编译运行基于 Struts2 框架的 Web 工程，需要导入 struts2 的 8 个核心 jar 包：

表 2-1 strut2 的 8 个核心 jar 包

**文件名**

**说明**

struts2-core-2.3.15.1.jar

Struts 2 框架的核心类库

xwork-core-2.3.15.1.jar

XWork 类库，Struts 2 的构建基础

Ognl-3.0.6.jar

Struts 2 使用的一种表达式语言类库（三）实验内容及步骤

1、登录 http://struts.apache.org/download.cgi 站点，下载 Struts2 的最新版（Full

Distribution）；

2、在 Eclipse 中新建 Web 工程 struts-prj1；

3、将 Struts2 中的 8 个核心包增加到 Web 应用中，即复制到“%workspace%

struts-prj1\WebContent\WEB-INF\lib”路径下，如下图所示；

图 2-1 Struts2 的 8 个核心包

4、在 struts-prj1 中新建 login.jsp 页面，作为用户登录的视图（代码片段如下），

注意表单（form）中 action 属性的值，注意用户名与密码输入框中 name 属性

的值；

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

请输入用户名：<input **name="loginUser.account"** type="text"><BR>

请输入密码：<input **name="loginUser.password"** type="password">

<input type="submit" value="登录">

</form>

5、在 struts-prj1 中新建 loginSuccess.jsp 和 loginFail.jsp 页面，分别作为登录成功

或登录失败的视图，在页面中显示“登录成功”或“登录失败”；

6、在 struts-prj1 中新建 cn.edu.zjut.bean 包，并在其中创建 UserBean.java，用于

记录登录用户信息（代码如下），注意该 JavaBean 中属性名的写法；

package cn.edu.zjut.bean;

public class UserBean {

private **String account=""**;

freemarker-2.3.19.jar

Struts 2 的标签模板使用类库

javassist-3.11.0.GA.jar

代码生成工具包

commons-lang3-3.1.jar

Apache 语言包，是 java.lang 包的扩展

commons-io-2.0.1.jar

Apache IO 包

commons-fileupload-1.3.jar

Struts 2 文件上传依赖包private **String password=""**;

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

7、在 struts-prj1 中新建 cn.edu.zjut.service 包，并在其中创建 UserService.java，

用于实现登录逻辑，为简化登录逻辑，将登录成功的条件设置为：用户名和

密码相同（代码如下）；

package cn.edu.zjut.service;

import cn.edu.zjut.bean.UserBean;

public class UserService {

public boolean login(UserBean loginUser) {

if (loginUser.getAccount().equals(loginUser.getPassword())) {

return true;

}

return false;

}

}

8、在 struts-prj1 中新建 cn.edu.zjut.action 包，并在其中创建 UserAction.java，调

用登录逻辑，并根据登录结果不同而返回不同的内容（代码如下），注意该

Action 中的属性名及相应的 getters 和 setters 方法、execute()方法及返回值；

package cn.edu.zjut.action;

import cn.edu.zjut.bean.UserBean;

import cn.edu.zjut.service.UserService;

public class UserAction {

private **UserBean loginUser**;

public UserBean **getLoginUser()** {

return loginUser;

}

public void **setLoginUser(UserBean loginUser)** {

this.loginUser = loginUser;

}

public **String** execute() {UserService userServ = new UserService();

if (userServ.login(loginUser)) {

**return "success"**;

}

**return "fail"**;

}

}

9、在工程 struts-prj1 的 src 目录中创建 struts.xml 文件，用于配置 Action 并设置

页面导航（代码片段如下），注意 action 标签中 name 属性和 class 属性的值，

以及 result 子标签的属性；

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

<!DOCTYPE struts PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"

"http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

<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>

</package>

</struts>

10、编辑 Web 应用的 web.xml 文件，增加 Struts2 核心 Filter 的配置（代码片段

如下）；

<!-- 定义 Struts2 的核心 Filter -->

<filter>

<filter-name>struts2</filter-name>

<filter-class>

org.apache.struts2.dispatcher.ng.filter

.StrutsPrepareAndExecuteFilter

</filter-class>

</filter>

<!-- 让 Struts2 的核心 Filter 拦截所有请求 -->

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

11、将 struts-prj1 部署在 Tomcat 服务器上；

12、通过浏览器访问 login.jsp 页面，并记录运行结果；

13、尝试对以上第 4、6、8、9 中的关键代码进行修改，观察修改后的运行结果。

（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；修改后的关键代码，及相应的运行结果或报错信息；

（2）根据实验过程，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）根据实验过程，总结表单参数与 Action 属性的赋值关系，并记录下来；

（4）根据实验过程，总结 Action 的 execute()方法的作用和特点，并记录下来；

（5）根据实验过程，查找相关资料，写出本实验中配置文件 struts.xml 里各元素

及其属性的作用；

（6）碰到的问题及解决方案或思考；

（7）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、提高实验——Struts2 标签**

（一）实验目的

1、进一步熟悉 Struts2 应用的基本开发步骤和常规配置；

2、进一步熟悉 Action 及配置文件 struts.xml 的应用方法；

3、掌握 Struts2 标签的基本使用方法；

4、能熟练使用 Struts2 的常用标签，能参考 Struts2 标签的使用说明文档，对各

类标签进行灵活应用；

（二）基本知识与原理

1、使用 Struts2 标签的形式来表达页面逻辑，可以尽量避免在视图中使用 Java

代码，让逻辑与显示分离，提高视图的可维护性；

2、Struts2 标签库的主要 tld 文件为 struts-tags.tld，在 struts2-core-2.3.15.1.jar 包中，

另一个与 Ajax 相关的标签库 tld 文件为 struts-dojo-tags.tld，在 struts2-dojo

-plugin-2.3.15.1.jar 包中；

3、Struts2 标签的使用步骤和使用 JSTL 相同，只需在 JSP 页面中使用 taglib 指令

引入标签库中 tld 文件的 uri，并指定前缀即可，例如：

<%@ taglib prefix="s" uri="/struts-tags"%>；

4、根据 Struts2 标签的主要作用，可以将其分为：用于生成页面元素的 UI 标签、

用于实现流程控制的控制类标签、用于控制数据的数据标签和用于支持 Ajax的标签。

（三）实验内容及步骤

1、为能使用与 Ajax 相关的标签，将 Struts2 中的 struts2-dojo-plugin-2.3.15.1.jar

包增加到工程 struts-prj1 中，即复制到“%workspace%struts-prj1\WebContent\

WEB-INF\lib”路径下，刷新工程；

2、在工程 struts-prj1 中增加用户注册功能，新建 register.jsp 页面作为用户注册的

视图，页面使用 Struts2 的 UI 标签来生成表单元素，包括用户名、密码、确

认密码、真实姓名、性别、生日、联系地址、联系电话和电子邮箱（代码片

段如下）；

**<%@ taglib prefix="s" uri="/struts-tags"%>**

**<%@ taglib prefix="sx" uri="/struts-dojo-tags"%>**

<html>

<head>

<s:head theme=*"xhtml"*/>

<sx:head parseContent=*"true"* extraLocales=*"UTF-8"*/> </head>

<body>

**<s:form action="register" method="post">**

<s:textfield name="loginUser.account" label="请输入用户名"/>

<s:password name="loginUser.password" label="请输入密码"/>

……

<s:radio name="loginUser.sex" list="#{1 : '男', 0 : '女'}" label

="请输入性别"/>

<sx:datetimepicker name="loginUser.birthday" displayFormat

="yyyy-MM-dd" label="请输入生日"/>

……

<s:submit value="注册"/>

<s:reset value="重置"/>

**</s:form>**

</body>

</html>

3、在 struts-prj1 中新建 regFail.jsp 页面，作为注册失败的视图，在页面中显示“注

册失败”；

4、在 struts-prj1 中新建 regSuccess.jsp 页面，作为注册成功的视图，使用 Struts2

的数据标签和控制标签来生成注册成功的信息，并将登录用户信息保存在会

话范围内（代码片段如下）；

<!-- 数据标签 property -->

<s:property value="loginUser.name"/>

<!-- 控制标签 if/else --><s:if test="%{loginUser.sex==\"1\"}">

<s:text name="先生，"/>

</s:if>

<s:else>

<s:text name="女士，"/>

</s:else>

您注册成功了！

<!-- 数据标签 set -->

<s:set name="user" value="loginUser" scope="session"/>

5、修改 UserBean.java，增加属性用于记录注册用户信息（代码片段如下）；

public class UserBean {

private String account="";

private String password="";

private String repassword="";

private String name="";

private String sex="";

private String birthday="";

private String address="";

private String phone="";

private String email="";

//省略 getters/setters 方法

}

6、修改 UserService.java，增加用户注册逻辑，为简化注册逻辑，将注册成功的

条件设置为：用户名、密码和确认密码相同，而且不为空字串；

7、修改 UserAction.java 中的 execute()方法，参照基础实验部分写入代码，用于

调用注册逻辑，并根据注册成功与否而返回不同的内容；

8、修改 struts.xml 文件，对用户注册进行配置并设置页面导航；

9、将 struts-prj1 重新布署在 Tomcat 服务器上，通过浏览器访问 register.jsp 页面，

并记录运行结果；

10、查找 Struts2 标签的相关资料，并尝试将其它的 Struts2 标签应用在实验步骤

2、3、4 中，观察应用后的运行结果。

（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）应用各种 Struts2 标签的关键代码，及相应的运行结果或报错信息；

（3）根据实验过程，查找相关资料，总结 Struts2 中标签及其属性的作用和用法

并记录下来；（4）碰到的问题及解决方案或思考；

（5）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——Struts2 的国际化**

（一）实验目的

1、进一步熟悉 Struts2 标签的基本使用方法；

2、能使用 Struts2 标签实现国际化；

3、了解配置文件 struts.properties 以及国际化资源文件的作用和基本使用方法。

（二）基本知识与原理

1、通过将不同语言版本的字符保存在属性文件中，Struts2 的国际化机制能够在

不需要修改程序主体的前提下，实现不同语言版本的应用；

2、在 Web 应用中选择需要进行国际化的内容，不在页面中直接输出该信息，而

是通过 Struts2 标签输出一个键值，该键值在不同语言环境下对应不同的字符

串；如：<s:textfield name="loginUser.account" lable="请输入用户名"/>代码中

的“请输入用户名”是需要国际化的内容，将其以键值代替为<s:textfield

name="loginUser.account" key="login.account.lable"/>；

3、需要进行国际化的内容以键值对（key=value）的形式写入 Struts2 的国际化资

源文件中，如“login.account.lable=请输入用户名”；该资源文件名可以自定

义，但是后缀必须是 properties；资源文件应放在 Web 应用的类加载路径下；

每一个语言版本需要创建一个资源文件；

4、通过 Struts2 的配置文件 struts.properties 来配置资源文件的基础名，若资源文

件的基础名为 message，则 message\_zh\_CN.properties 是对应的中文资源文件，

message\_en\_US.properties 是对应的美国英语资源文件；

5、浏览器将根据其默认的语言版本，自动调用相应语言的资源文件，从而在页

面中展示不同的语言效果。

（三）实验内容及步骤

1、在工程 struts-prj1 的 src 目录下新建一个 cn.edu.zjut.local 包，把所有的资源文

件放置其中，如：message\_zh\_CN.properties、message\_en\_US.properties 等；2、将工程 struts-prj1 中用户登录模块的三个 JSP 页面进行国际化处理，选择需

要进行国际化的内容，以键值对的形式写入资源文件中（代码片段如下）；

# message\_en\_US.properties

login.account.lable=Please input your account

login.password.lable=Please input your password

login.submit.button=submit

# message\_zh\_CN.properties

login.account.lable=请输入用户名

login.password.lable=请输入密码

login.submit.button=登录

3、使用 JDK 中的 native2ASCII 工具，将 message\_zh\_CN.properties 重新编码，

将中文字符都转化为 unicode 码，从而避免乱码问题；

图 2-2 使用 native2ASCII 工具

# message\_zh\_CN.properties

login.account.lable=\u8bf7\u8f93\u5165\u7528\u6237\u540d

login.password.lable=\u8bf7\u8f93\u5165\u5bc6\u7801

login.submit.button=\u767b\u5f55

4、在工程 struts-prj1 的 src 目录中创建 struts.properties 文件，通过它加载资源文

件（具体代码如下）；

**struts.custom.i18n.resources=cn.edu.zjut.local.message**

struts.i18n.encoding=GBK

或者在JSP页面中临时加载资源文件（代码片段如下）；

**<s:i18n name="cn.edu.zjut.local.message">**

<s:form action="login" method="post"> …… </s:form>

**</s:i18n>**

5、修改 login.jsp、loginSuccess.jsp 和 loginFail.jsp 页面，通过 Struts2 标签实现国

际化（代码片段如下）；

<!-- login.jsp -->

<s:textfield name="loginUser.account" **key="login.account.lable"**/>

<s:password name="loginUser.password" **key="login.password.lable"**/>

<s:submit name="submit" **key="login.submit.button"**/>

6、将 struts-prj1 重新布署在 Tomcat 服务器上；

7、设置浏览器的首语言选项（如下图所示），通过浏览器访问 login.jsp 页面，

观察并记录运行结果；图 2-3 浏览器的语言首选项

8、尝试修改用户注册模块，实现国际化，并记录运行结果。

（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）应用国际化的关键代码，及相应的运行结果或报错信息；

（3）查找相关资料，总结配置文件 struts.properties 的作用并记录下来；

（4）碰到的问题及解决方案或思考；

（5）实验收获及总结。

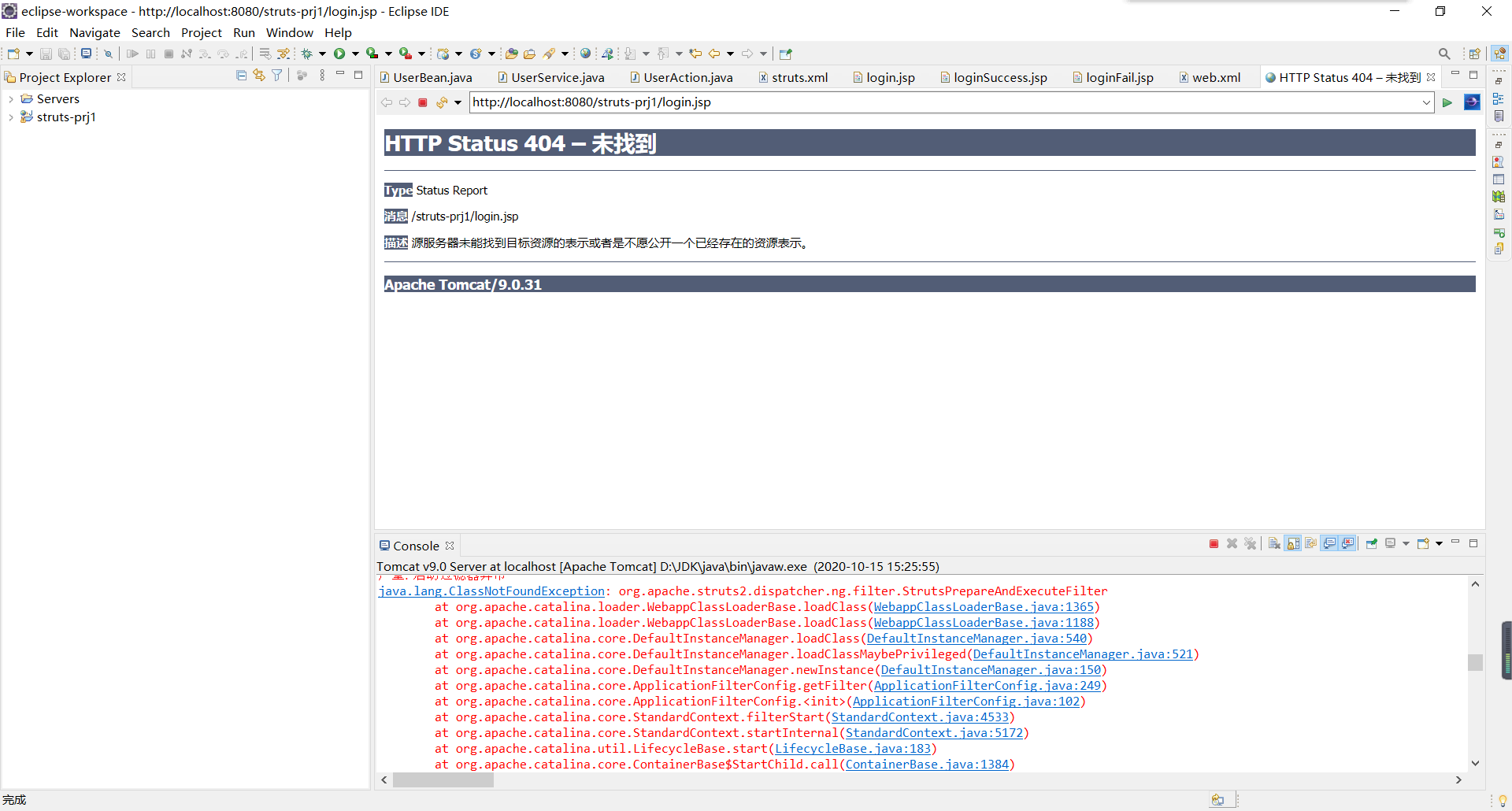
2、上交程序源代码，代码中应有相关注释

# 二、实验结果及问题分析

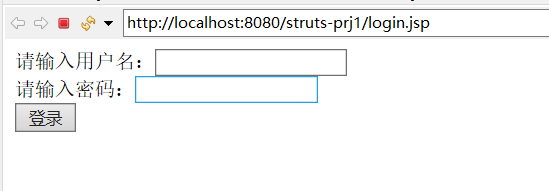
**一、基础实验——Struts2 框架搭建**

（1）实验结果：

**web.xml修改前**



**web.xml修改后（将org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter改成org.apache.struts2.dispatcher.filter.StrutsPrepareAndExecuteFilter）**



**尝试修改4,6,8,9中的关键代码之后仍能完成登录（以下是修改后的代码，原先的代码是版本1，更改了两次其中的关键代码，分别称为版本2和版本3，三个坂本都能完成登录）**

UserBean.java

/\*版本1

public class UserBean {

private String account="";

private String password="";

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}\*/

//版本3

public class UserBean {

private login login;

public login getLogin() {

return login;

}

public void setLogin(login login) {

this.login = login;

}

}

UserAction.java

package cn.edu.zjut.action;

import cn.edu.zjut.bean.UserBean;

import cn.edu.zjut.service.UserService;

/\*版本1

public class UserAction {

private UserBean loginUser;

public UserBean getLoginUser() {

return loginUser;

}

public void setLoginUser(UserBean loginUser) {

this.loginUser = loginUser;

}

public String execute() {

UserService userServ = new UserService();

if (userServ.login(loginUser)) {

return "success";

}

return "fail";

}

}\*/

/\*版本2

public class UserAction {

private String password;

private String account;

public String execute() {

if (getPassword().equals(getAccount())) {

return "success";

}

return "fail";

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

}\*/

//版本3

public class UserAction {

private UserBean loginUser;

public UserBean getLoginUser() {

return loginUser;

}

public void setLoginUser(UserBean loginUser) {

this.loginUser = loginUser;

}

public String execute() {

UserService userServ = new UserService();

if (userServ.login(loginUser)) {

return "success";

}

return "fail";

}

}

UserService.java

package cn.edu.zjut.service;

import cn.edu.zjut.bean.UserBean;

/\*版本1

public class UserService {

public boolean login(UserBean loginUser) {

if (loginUser.getAccount().equals(loginUser.getPassword())) {

return true;

}

return false;

}

}

\*/

//版本3

public class UserService {

public boolean login(UserBean loginUser) {

if (loginUser.getLogin().getAccount().equals(loginUser.getLogin().getPassword())) {

return true;

}

return false;

}

}

新增一个类login

login.java

package cn.edu.zjut.bean;

public class login {

private String account;

private String password;

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

login.jsp

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>登录页面</title>

</head>

<body>

<%-- 版本1

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

请输入用户名：<input name="loginUser.account" type="text"><BR>

请输入密码：<input name="loginUser.password" type="password"><BR>

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

--%>

<%-- 版本2

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

请输入用户名：<input name="account" type="text"><BR>

请输入密码：<input name="password" type="password"><BR>

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

--%>

<%-- 版本3 --%>

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

请输入用户名：<input name="loginUser.login.account" type="text"><BR>

请输入密码：<input name="loginUser.login.password" type="password"><BR>

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

</body>

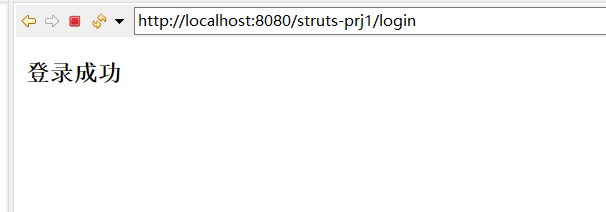
</html>

**运行结果：**

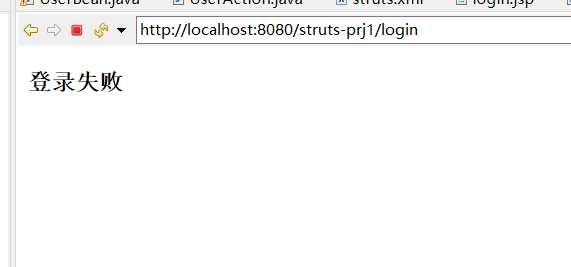
**登录页面**



**登录成功页面（用户名和密码相同）**



**登录失败页面（用户名和密码不相同）**



（2）实验结果：

Jsp的作用：作为view的部分，用于给用户显示页面，接受处理客户端的请求。

Action类的作用：struts2中的第二层控制器 ，可以处理请求的一些参数，其将业务逻辑的调用封装到一个单独的工作单元中，处理客户端的请求。

Service类的作用：属于业务层，对业务逻辑进行处理。

Javabean的作用：作为model的部分，实现数据的共享和存储。

Filter的作用：过滤所有页面，使之能够成功进入到struts2框架中。

Struts.xml文件的作用：配置文件。

完整流程：

1. 发送请求
2. struts2框架的主控器(FilterDispatcher)接收该请求，判断这个请求是交给Action处理还是交给web组件处理，在此处理过程中需要一个辅助对象
3. Action调用业务方法，之后返回业务结果。
4. Action返回逻辑视图名给StrutsPrepareAndExecuteFilter。
5. StrutsPrepareAndExecuteFilter Forword到物理视图。
6. 视图页面生成响应内容。
7. 最后浏览器输出响应。

Struts框架中MVC的体现：分为M（model），V（view），C（controller）三个模式，javabean作为M部分，jsp作为V部分，action作为C部分。

（3）实验结果：表单参数与Action的赋值关系

【1】在Action类中定义表单属性，两者属性名称必须一致,提供setter即可接收到表单传过来的参数。接受参数的方法结构简单，方便。

【2】把表单传递过来的参数封装成一个类， 然后调用其中的属性。Action中只需定义该类的对象，jsp中需要处理其中的name。

（4）实验结果：execute（）方法的作用和特点

Action中的execute（）方法会默认调用，execute（）方法会返回一个字符串。

1. 实验结果：struts.xml中各元素及其属性的作用。

<package>

用包机制将功能相似的action放到一个package中

name属性：包名，用来标识<package/> ，可任意命名。

abstract属性：用于设置为抽象包，abstract= “true”。

extends属性：用于设置对其它包的继承，通常继承一个struts2的内置包"struts-default"。

struts2支持多继承：<package name="struts2" extends= "struts-default, jfreechart-default">。

父包必须定义在子包之前。

Namespace属性：管理大型项目中action，解决action重名问题，在不同命名空间的action可用相同名字，默认的命名空间为"" 。

<package>中可以有多个<action>

<action>：

name属性:请求的action名称。

class属性：请求的Action类名。

method属性：指定Action中的方法名，没有指定method则默认执行Action中的execute方法。

converter属性：指定Action使用的类型转换器。

<result>:

name属性：对应Action的execute方法返回的字符串（逻辑视图名称），默认为success

type属性：返回结果的方式，默认为dispatcher

type取值：1、dispatcher: 处理结果为作服务器内部跳转(跳转到web组件)；2.redirect: 作客户端重定向(重定向到web组件)；3.chain: 作服务器内部跳转(跳转到其他Action)；4.redirectAction: 作客户端重定向(重定向到其他Action)

1. 实验结果：

遇到的问题：

1. 需要将<filter-class>改成org.apache.struts2.dispatcher.filter.StrutsPrepareAndExecuteFilter。
2. 改完第一个之后发现tomcat启动不了，根据报错信息发现是少添加了一个包，添加进lib之后即可。
3. 在改关键代码时有一些迷茫，在回忆上课时举得例子之后才找到了思路，之后成功配置完代码。
4. 实验收获及总结：
5. 熟悉并尝试了struts2框架，明白了他与MVC之间的关系，学会了如何正确配置和运用struts2框架。
6. 再一次明确了MVC的框架结构，使用流程等。

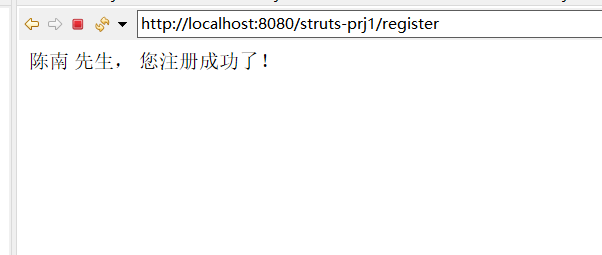
**二、提高实验——Struts2 标签**

（1）实验结果：

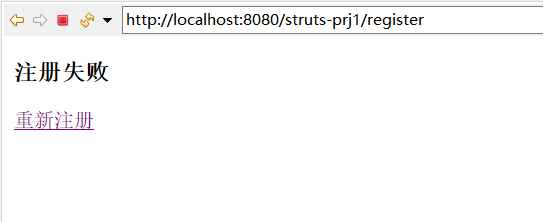
**注册页面：**



**注册成功页面：**



**注册失败页面：**



1. 报错信息



查询资料后发现是因为struts2.5版本之后有了一些奇特的变化，将name改成var之后可以正常运行。



1. Struts2标签及其属性的作用和写法

【1】checkbox

name:对应action中的属性名称；

fieldValue:选中时action变量被赋的值，当action的字段是布尔型的时候可以省略这个属性；

value:只能为true或false，来决定生成的checkbox是否被选中，尽量不用这个属性；

lable:jsp页面为checkbox设定的显示内容。

【2】checkboxlist

name:对应action中的属性名称，属性的类型即是listKey的值对应的类型；

list:required，这个属性的值必须为可迭代类型，比如list，set，array。用以显示所有的选择项。如果list的值为Map，则map的key对应option的值，map的value对应option的显示内容；

listKey:生成的checkbox的input元素的value属性；

listValue:生成的checkbox的input元素的显示内容；

【3】combobox

Name：对应action中的属性名称，属性的类型即是listKey的值所对应的类型，这个属性的值也即是生成的text field的name的值;

list：Required，这个属性的值必须为可迭代类型或者是数组类型，比如list，set，array，用以显示所有的选择项。如果list的值为Map，则map的key对应option的值，map的value对应option的显示内容;

listKey：生成的select 的option元素的value属性，也即是选择某个option后，text field显示的内容;

listValue：生成的select的option元素的显示内容;

headerKey：设置第一个选项的值;

headerValue：设置第一个选项的显示内容;

【4】doubleselect

Name:对应action中的变量名称，生成的parent select的name属性;

doubleName:Required，对应action中变量的名称，生成的child select的name属性;

List:Required，生成的parent select元素显示的可选列表，参照combobox的list属性;

doubleList:Required，生成的child select元素显示的可选列表，相当于child select标签的list属性;

listKey:生成的parent select 的option元素的value属性;

listValue:生成的parent select的option元素的显示内容;

doubleListKey:Child select的选项的value属性;

doubleListValue:Child select的option元素显示的内容;

headerKey:设置parent select的第一个选项的值;

headerValue:设置parent select第一个选项的显示内容;

【5】optiontransferselect

name:对应action中的变量名称，生成的left select的name属性

doubleName:Required，对应action中变量的名称，生成的complete component的name属性;

List:Required：生成的left select元素显示的可选列表，参照combobox的list属性;

doubleList：Required，生成的complete select元素显示的可选列表，参照 list属性;

listKey：生成的left select 的option元素的value属性;

listValue：生成的left select的option元素的显示内容;

doubleListKey：complete select的选项的value属性;

doubleListValue：complete select的option元素显示的内容;

leftTitle：Left select title;

rightTitl：Right select title;

Multiple：布尔型，如果设置为true，标签会为左边的选项预先选中，这要通过Array类型的value：属性来确定哪些是预选中的;

DoubleMultiple：参照multiple属性，起作用于右面的select组件;

EmptyOption：布尔型，决定左面的select是否添加一个空选项;

doubleEmptyOption：布尔型，决定右面的select是否添加一个空选项；

【6】radio

name：对应action中的属性名称，属性的类型即是listKey的值所对应的类型；

List:Required，这个属性的值必须为可迭代类型或者是数组类型，比如list，set，array。用以显示所有的选择项。如果list的值为Map，则map的key对应radio的值，map的value对应radio的显示内容;

listKey:生成的radio 的input元素的value属性;

listValue:生成的radio的input元素的显示内容;

【7】select和optgroup

name:对应action中的属性名称，属性的类型即是listKey的值所对应的类型；

list：Required，这个属性的值必须为可迭代类型或者是数组类型，比如list，set，array，用以显示所有的选择项。如果list的值为Map，则map的key对应option的值，map的value对应option的显示内容；

listKey：生成的select 的option元素的value属性，也即是选择某个option后，text field显示的内容；

listValue：生成的select的option元素的显示内容；

headerKey：设置第一个选项的值；

headerValue：设置第一个选项的显示内容；

【8】hidden

hidden标签在页面产生一个hidden的域，用来存放不需要显示而不想丢失的值。

例子

<s:hidden name="foo" />

【9】a

errorText：这是ajax的a标签属性，用来显示错误信息；

href：设置链接；

【10】if/elseif/else

用于控制输出的内容

【11】iterator

status：制定迭代时的IteratorStatus实例,通过该实例即可判断当前迭代元素的属性;

value：指定迭代的集合,若没有指定则使用ValueStack栈顶的集合;

id:指定集合里元素的id;

【12】generator

count：生成集合元素中的总数；

Seperator：必填，指定分隔符；

Val：必填，指定被解析的字符串；

Converter：负责将集合中的每个字符串转换成对象；

Id：将生成的集合发在request的属性中；

【13】bean

用于创建一个javabean的实例；

【14】property

用于输出指定值

default：可选属性，如果需要输出的属性值为null，则显示该属性指定的值

value：可选属性，指定需要输出的属性值，如果没有指定该属性，则默认输出ValueStack栈顶的值

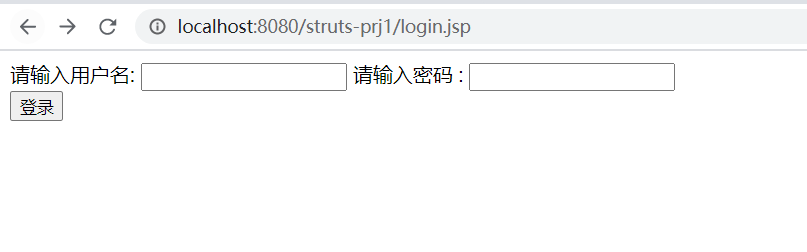
id：可选属性，指定该元素的标识

1. 遇到的问题及解决：
2. 完成注册和登录两个功能时由于用的是同一个action类，loginUser和registerUser作为两个私有变量存在，问题的关键在于进行注册账号时返回到的是login的页面，后来观察代码思考之后增添一个判断语句，判断有无loginUser的请求存在，如果有则进入到login的业务逻辑，没有再进入register的业务逻辑中，这样就做到了一个action同时判断注册和登录。
3. 再用set标签时发现了包版本与代码之间的问题，思路在于更改包的版本或是书写符合包版本的代码。
4. 实验总结及收获
5. 熟悉了struts2标签
6. 尝试了用两个action分别判断注册和登录、用同一个action同时判断注册和登录两种写法的书写模式以及配置方法。

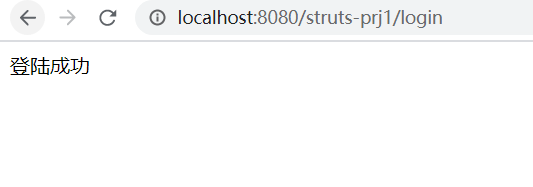
**三、扩展实验——Struts2 的国际化**

（1）实验截图

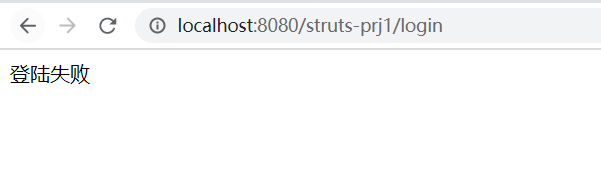
**login.jsp中文版**



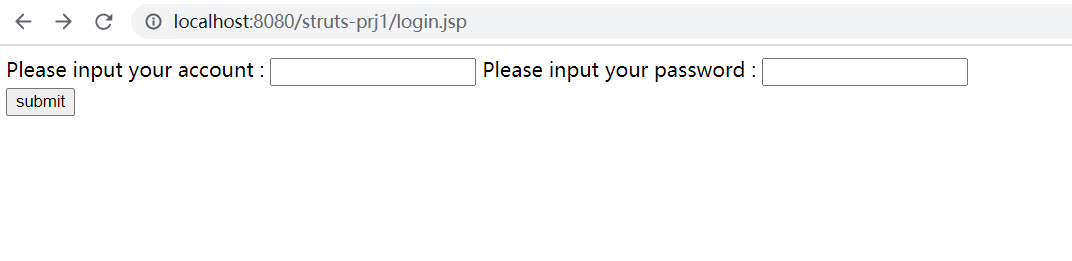
**登录成功**



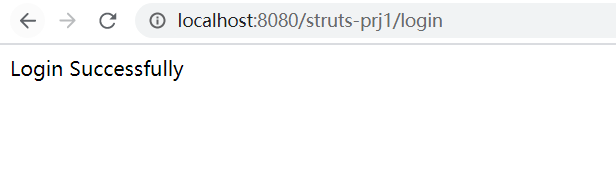
**登录失败**



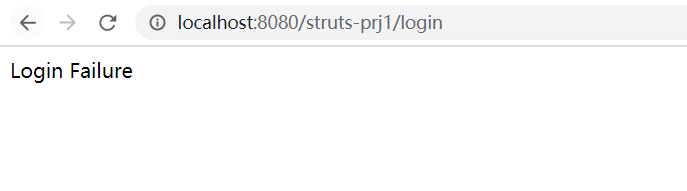
**login.jsp英文版**



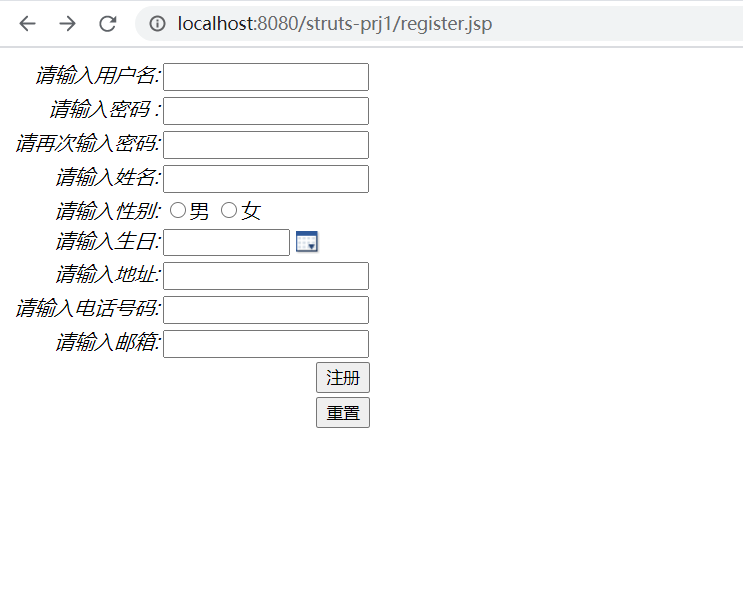
**登录成功**



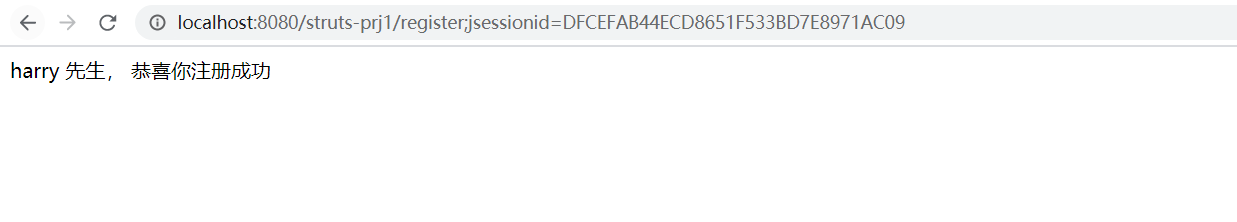
**登录失败**



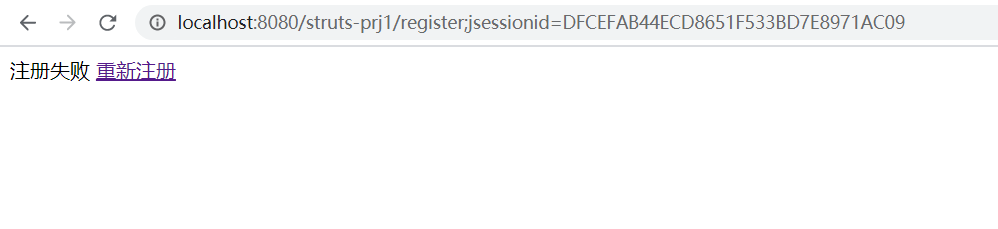
**register.jsp中文版**



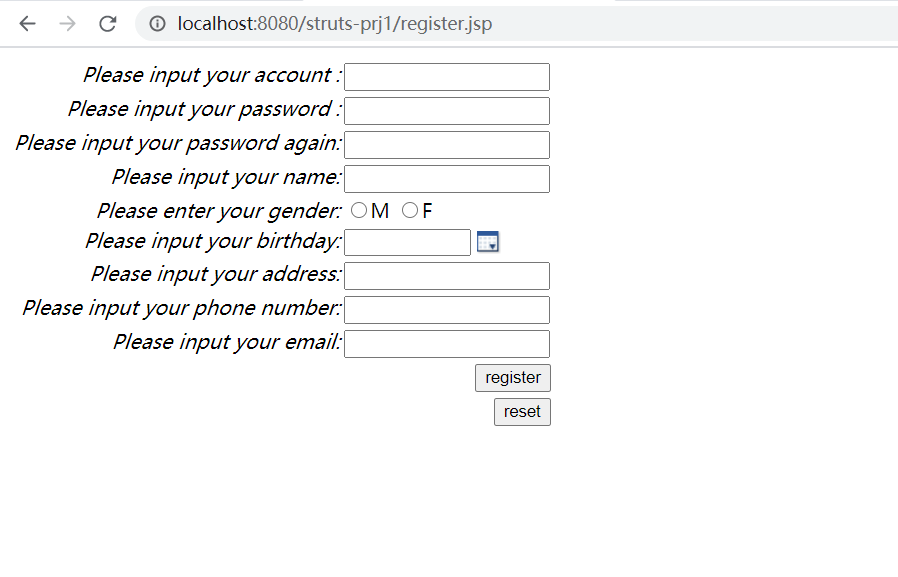
**注册成功**



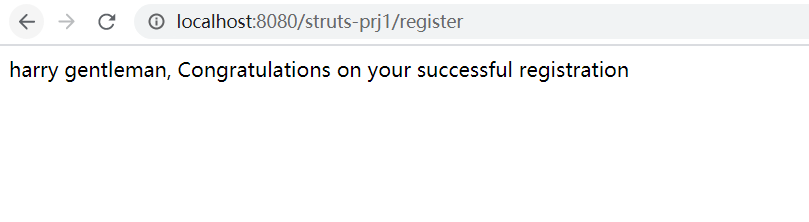
**注册失败**



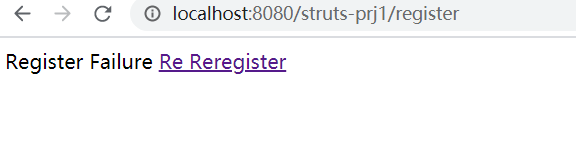
**register.jsp英文版**



**注册成功**



**注册失败**



1. 国际化的关键代码：

**message\_zh\_CN.properties**

login.account.lable=Please input your account

login.password.lable=Please input your password

login.fail.label=Login Failure

login.success.label=Login Successfully

login.submit.button=submit

registerUser.account.lable=Please input your account

registerUser.password.lable=Please input your password

registerUser.repassword.lable=Please input your password again

registerUser.sex.lable=Please enter your gender

registerUser.name.lable=Please input your name

registerUser.birthday.lable=Please input your birthday

registerUser.address.lable=Please input your address

registerUser.phone.lable=Please input your phone number

registerUser.email.lable=Please input your email

registerUser.gentleman.label=gentleman,

registerUser.madam.label=madam,

register.submit.button=register

register.reset.button=reset

register.success.label=Congratulations on your successful registration

register.fail.label=Register Failure

register.reregister.label=Re Reregister

**message\_en\_US.properties**

login.account.lable=\u8BF7\u8F93\u5165\u7528\u6237\u540D

login.password.lable=\u8BF7\u8F93\u5165\u5BC6\u7801

login.fail.label=\u767B\u9646\u5931\u8D25

login.success.label=\u767B\u9646\u6210\u529F

login.submit.button=\u767B\u5F55

registerUser.account.lable=\u8BF7\u8F93\u5165\u7528\u6237\u540D

registerUser.password.lable=\u8BF7\u8F93\u5165\u5BC6\u7801

registerUser.repassword.lable=\u8BF7\u518D\u6B21\u8F93\u5165\u5BC6\u7801

registerUser.sex.lable=\u8BF7\u8F93\u5165\u6027\u522B

registerUser.name.lable=\u8BF7\u8F93\u5165\u59D3\u540D

registerUser.birthday.lable=\u8BF7\u8F93\u5165\u751F\u65E5

registerUser.address.lable=\u8BF7\u8F93\u5165\u5730\u5740

registerUser.phone.lable=\u8BF7\u8F93\u5165\u7535\u8BDD\u53F7\u7801

registerUser.email.lable=\u8BF7\u8F93\u5165\u90AE\u7BB1

register.submit.button=\u6CE8\u518C

register.reset.button=\u91CD\u7F6E

registerUser.gentleman.label=\u5148\u751F\uFF0C

registerUser.madam.label=\u5973\u58EB\uFF0C

register.success.label=\u606D\u559C\u4F60\u6CE8\u518C\u6210\u529F

register.fail.label=\u6CE8\u518C\u5931\u8D25

register.reregister.label=\u91CD\u65B0\u6CE8\u518C

**register.jsp（部分代码）**

<s:form action="register" method="post">

<s:textfield name="registerUser.account" key="registerUser.account.lable"/>

<s:password name="registerUser.password" key="registerUser.password.lable"/>

<s:password name="registerUser.repassword" key="registerUser.repassword.lable"/>

<s:textfield name="registerUser.name" key="registerUser.name.lable"/>

<s:radio name="registerUser.sex" list="#{1 : '男', 0 : '女'}" key ="registerUser.sex.lable"/>

<sx:datetimepicker name="registerUser.birthday" displayFormat ="yyyy-MM-dd" key="registerUser.birthday.lable"/>

<s:textfield name="registerUser.address" key="registerUser.address.lable"/>

<s:textfield name="registerUser.phone" key="registerUser.phone.lable"/>

<s:textfield name="registerUser.email" key="registerUser.email.lable"/>

<s:submit name="register.submit" key="register.submit.button"/> <s:reset name="register.reset" key="register.reset.button"/>

</s:form>

**regSuccess.jsp（部分代码）**

<s:if test="%{registerUser.sex==\"1\"}">

<s:text name="registerUser.gentleman.label"/>

</s:if>

<s:else>

<s:text name="registerUser.madam.label"/>

</s:else>

<s:text name="register.success.label"/>

**regFail.jsp（部分代码）**

<s:text name="register.fail.label"/>

<a href="register.jsp"><s:text name="register.reregister.label"/> </a>

1. struts.properties 的作用
2. 用来全局化配置struts.custom.i18n.resources常量即可加载全局国际化资源文件
3. .properties是属性文件，用来存放数据，一般以键值的形式存放。
4. 该文件包含了系列的key-value对象，每个key就是一个Struts2属性，该key对应的value就是一个Struts2属性值。
5. 碰到的问题及解决
6. 碰到乱码，查询资料之后更改编译器的配置之后就行更改。
7. text标签、a标签、radio标签等一些标签中国际化的使用，查阅相关资料之后进行修改。
8. 注册时姓名输入中文后在注册成功的页面出现中文字乱码的情况，查阅资料之后将struts.properties中的编码方式改成UTF-8，即可。
9. 实验总结
10. 了解并使用了struts2框架中国际化的使用。
11. 对Struts2框架中一些标签的国际化书写有了了解和实践，还可以有更多的尝试。