实验10 Spring MVC框架：异常处理及拦截器

**【实验目的及要求】**

1. 熟悉拦截器的配置与使用；
2. 掌握异常处理的配置与使用；
3. 要求所有回答的文本格式：**五号，宋体、1.5倍行距，保留段单元格背景。**

**【实验步骤】**

**1、拦截器应用**

（1）在IDEA中新建Maven项目TestInterceptor，在pom.xml文件中配置好相关依赖。

（2）在webapp文件夹中新建jsp文件index.jsp，提供进入系统管理端的链接，内容如下。

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>成绩管理系统</title>  </head>  <body>  <a href=*"tomain"*>系统管理端</a>  </body>  </html> |

**说明**：该页面用于模拟Web系统的首页，首页不需要登录就可以访问，但系统管理端需要登录后才能访问。

（3）在WEB-INF/jsp文件夹中新建jsp文件main.jsp，显示当前登录用户信息，以及注销和返回主页链接，内容如下。

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <%@ taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"* %>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>后台管理</title>  </head>  <body>  <c:if test=*"*${sessionScope.username!=null}*"*>  <p>当前用户：${sessionScope.username}</p>  <p><a href=*"tologout"*>注销</a></p>  <p><a href=*"*${pageContext.request.contextPath}*/index.jsp"*>返回首页</a></p>  </c:if>  <p>欢迎进入系统管理端</p>  </body>  </html> |

**说明**：该页面用于模拟Web系统的后台管理端，只有有效登录的用户才能访问。该页面提供了注销及返回首页的链接。

（3）在WEB-INF/jsp文件夹中新建jsp文件login.jsp，提供用户登录表单，内容如下。

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>用户登录</title>  </head>  <body><center>  <form id=*"form1"* name=*"form1"* method=*"post"* action=*"*${pageContext.request.contextPath}*/userlogin"*>  <table width=*"344"* height=*"95"* border=*"0"* cellpadding=*"0"* cellspacing=*"1"* style="background-color: *#3A8ECD*; color: *#000*;">  <tr>  <td height=*"30"* colspan=*"2"* align=*"center"* valign=*"middle"* bgcolor=*"#FFFFFF"*>用户登录</td>  </tr>  <tr>  <td width=*"114"* height=*"30"* align=*"right"* valign=*"middle"* bgcolor=*"#FFFFFF"*>登录名：</td>  <td width=*"227"* height=*"20"* align=*"left"* valign=*"middle"* bgcolor=*"#FFFFFF"*>  <input type=*"text"* name=*"username"* /></td>  </tr>  <tr>  <td height=*"30"* align=*"right"* valign=*"middle"* bgcolor=*"#FFFFFF"*>密码：</td>  <td height=*"20"* align=*"left"* valign=*"middle"* bgcolor=*"#FFFFFF"*><label for=*"textfield2"*></label>  <input type=*"password"* name=*"pwd"* /></td>  </tr>  <tr>  <td height=*"30"* colspan=*"2"* align=*"center"* valign=*"middle"* bgcolor=*"#FFFFFF"*><input type=*"submit"* name=*"button"* id=*"button"* value=*"提交"* />&nbsp;&nbsp;  <input type=*"reset"* name=*"button2"* id=*"button2"* value=*"重置"* /></td>  </tr>  </table>  </form></center>  </body>  </html> |

（5）修改WEB-INF目录中的web.xml配置文件，内容如下。

|  |
| --- |
| <?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"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"* id=*"WebApp\_ID"* version=*"3.0"*>  <welcome-file-list>  <welcome-file>index.jsp</welcome-file>  </welcome-file-list>  <!--Spring MVC前端器 -->  <servlet>  <servlet-name>grademanage</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:config/springmvc-config.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>grademanage</servlet-name>  <url-pattern>/</url-pattern>  </servlet-mapping>  </web-app> |

**说明**：文件中配置了Spring MVC前端控制器。

（6）在/main/resource/config目录中新建springmvc-config.xml配置文件，内容如下。

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"*  **xmlns:mvc=*"http://www.springframework.org/schema/mvc"***  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xsi:schemaLocation=*"http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-4.3.xsd*  ***http://www.springframework.org/schema/mvc***  ***http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd***  *http://www.springframework.org/schema/context*  *http://www.springframework.org/schema/context/spring-context-4.3.xsd"*>  <context:component-scan base-package=*"com.controller"* />  <bean id=*"viewResolver"* class=  *"org.springframework.web.servlet.view.InternalResourceViewResolver"*>  <!-- 设置前缀 -->  <property name=*"prefix"* value=*"/WEB-INF/jsp/"* />  <!-- 设置后缀 -->  <property name=*"suffix"* value=*".jsp"* />  </bean>  <mvc:interceptors>  <mvc:interceptor>  <mvc:mapping path=*"/tomain"*/>  <bean class=*"com.interceptor.UserInterceptor"*/>  </mvc:interceptor>  </mvc:interceptors>  </beans> |

**说明**：配置了控制器扫描、视图解析器和拦截器（仅拦截以/tomain为结尾的请求）。请留意红色文字部分，其是配置拦截器时需要添加的约束信息。

（7）在com.interceptor包中，新建拦截器类UserInterceptor，内容如下。

|  |
| --- |
| package com.interceptor;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import org.springframework.web.servlet.HandlerInterceptor;  import org.springframework.web.servlet.ModelAndView;  public class UserInterceptor implements HandlerInterceptor {  public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception{  HttpSession session = request.getSession();  String username = (String)session.getAttribute("username");  if (username != null)  return true;    request.getRequestDispatcher("/WEB-INF/jsp/login.jsp").forward(request, response);  return false;  }  public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler, ModelAndView modeAndView) throws Exception{  }  public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler, Exception ex) throws Exception {  }  } |

**说明**：拦截器需要实现接口HandlerInterceptor，上述拦截器仅在preHandle方法中，添加了用户登录验证功能，即已登录则通过，继续转入控制器执行，否则转发至登录页面login.jsp。

（8）在com.po包中，新建实体类User，内容如下。

|  |
| --- |
| **package** com.po;  **public** **class** User {  **private** String username;  **private** String pwd;  **public** String getUsername() {  **return** username;  }  **public** **void** setUsername(String username) {  **this**.username = username;  }  **public** String getPwd() {  **return** pwd;  }  **public** **void** setPwd(String pwd) {  **this**.pwd = pwd;  }  } |

**说明**：实体类User在这个例子中仅用于绑定数据。

（9）在com.controller包中，新建控制器类UserController，内容如下。

|  |
| --- |
| **package** com.controller;  **import** javax.servlet.http.HttpSession;  **import** org.springframework.stereotype.Controller;  **import** org.springframework.ui.Model;  **import** org.springframework.web.bind.annotation.RequestMapping;  **import** org.springframework.web.bind.annotation.RequestMethod;  **import** com.po.User;  @Controller  **public** **class** UserController {  @RequestMapping(value="/tologout",method=RequestMethod.***GET***)  **public** String toLogout(HttpSession session) {  session.invalidate();  **return** "login";  }    @RequestMapping(value="/userlogin",method=RequestMethod.***POST***)  **public** String userLogin(User user, HttpSession session, Model model) {  **if** (user.getUsername() != "" || user.getUsername() != **null**)  session.setAttribute("username", user.getUsername());    **return** "main";  }    @RequestMapping(value="/tomain", method=RequestMethod.***GET***)  **public** String toMain() {  **return** "main";  }  } |

**说明**：控制器UserController用于处理用户有关的请求，可以处理3个不同的请求。注意，仅“/tomain”请求，会被拦截器拦截。

（10）在Web服务环境中测试上述程序的有效性。

（11）将index.jsp、main.jsp、login.jsp、UserInterceptor和UserController等看作对象，绘图（可以用活动图）表示各对象之间的跳转关系。

|  |
| --- |
|  |

**2、异常处理应用**

（1）以注解方式，完成上述项目中，处理器算术异常及其它异常的捕捉及视图响应。

**要求：项目名称以自己姓名拼音的首字母取名，如张晓峰的项目名为zxf；该项目要求提交源码的压缩文件，以“学号\_姓名”命名。**