<http://blog.csdn.net/sunp823/article/details/51694662>

1.使用背景

Web项目中需要判断http接口用户Post上来的数据是否合法，如果不合法要另做处理，用户Post上来的数据是Json形式的，我们用了@RequestBody标记自动将json形式的提交封装为一个Model对象，这样一来，我们就不能判断在自动封装过程中是否出现了异常，此时便想起了springMVC中的interceptor，用于处理请求之前，做一些处理，我们可以取消@RequestBody标记，然后在interceptor中取得请求体，检查是否符合json要求，即是不是一个valid interceptor，但是这里出现了一个问题：   
httpServletRequest的请求内容，只能被读取一次，在interceptor中读取了的话，在controller中便不能读取了，解决方式是，读取到的请求内容存起来，然后在controller中直接使用。   
这里用到了interceptor的一种，HandlerInterceptor，可以写一个小例子来记一下是怎么使用的！

2.HandlerInterceptor概述

在SpringMVC 中定义一个Interceptor是比较非常简单，主要有两种方式：   
第一种：实现HandlerInterceptor 接口，或者是继承实现了HandlerInterceptor 接口的类，例如HandlerInterceptorAdapter；   
第二种：实现Spring的WebRequestInterceptor接口，或者是继承实现了WebRequestInterceptor的类。   
现在主要结合一个例子说一下第一种方式：实现HandlerInterceptor接口。   
HandlerInterceptor接口主要定义了三个方法：   
**1. boolean preHandle (HttpServletRequest request, HttpServletResponse response, Object handle)方法：**该方法将在请求处理之前进行调用，只有该方法返回true，才会继续执行后续的Interceptor和Controller，当返回值为true 时就会继续调用下一个Interceptor的preHandle 方法，如果已经是最后一个Interceptor的时候就会是调用当前请求的Controller方法；   
**2.void postHandle (HttpServletRequest request, HttpServletResponse response, Object handle, ModelAndView modelAndView)方法：**该方法将在请求处理之后，DispatcherServlet进行视图返回渲染之前进行调用，可以在这个方法中对Controller 处理之后的ModelAndView 对象进行操作。   
**3.void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handle, Exception ex)方法：**该方法也是需要当前对应的Interceptor的preHandle方法的返回值为true时才会执行，该方法将在整个请求结束之后，也就是在DispatcherServlet 渲染了对应的视图之后执行。用于进行资源清理。

3.简单的一个例子：

**1.xml需要配置：两种配置方式（对所有的请求记性拦截，对特定的请求进行拦截）**

<mvc:interceptors>

<!--对所有的请求记性拦截-->

<!--<beans:bean class="com.sunp.common.interceptor.Myinterceptor"/>-->

<!--对特定的请求进行拦截-->

<mvc:interceptor>

<mapping path="/kfc/brands/brand1/\*"/>

<beans:bean class="com.sunp.common.interceptor.Myinterceptor"/>

</mvc:interceptor>

</mvc:interceptors>

**2.interceptors类**

package com.sunp.common.interceptor;

import com.alibaba.fastjson.JSON;

import com.alibaba.fastjson.JSONObject;

import com.mkyong.common.model.Shop;

import org.springframework.web.servlet.HandlerInterceptor;

import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.BufferedReader;

import java.io.IOException;

/\*\*

\* Created by hzsunpeng on 2016/6/14.

\*/

public class Myinterceptor implements HandlerInterceptor {

public boolean preHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o) throws Exception {

System.out.println("preHandle run!");

return true;

}

public void postHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, ModelAndView modelAndView) throws Exception {

System.out.println("postHandle run!");

}

public void afterCompletion(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, Exception e) throws Exception {

System.out.println("afterCompletion run!");

}

}

**3.controller类(为了使用@Controller @PathVariable @ResponseBody注释等需要加一些配置文件，这里不再列出)**

package com.sunp.common.controller;

import com.sunp.common.model.Shop;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.\*;

import javax.servlet.http.HttpServletRequest;

/\*\*

\* Created by hzsunpeng on 2016/6/7.

\*/

@Controller

@RequestMapping("/kfc/brands")

public class JSONController {

@RequestMapping(value = "/brand1/{name}",method = RequestMethod.GET)

public @ResponseBody Shop getShopInJSON(@PathVariable String name)

{

Shop sp = new Shop();

sp.setName(name);

sp.setStaffName(new String[]{"Staffname1","Staffname2"});

System

return sp;

}

//测试用

@RequestMapping(value = "/json",method = RequestMethod.POST)

public String index(HttpServletRequest request)

{

return "index";

}

}

**4.model Shop类**

package com.sunp.common.model;

import java.io.Serializable;

/\*\*

\* Created by hzsunpeng on 2016/6/7.

\*/

public class Shop implements Serializable {

private String name;

private String[] staffName;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String[] getStaffName() {

return staffName;

}

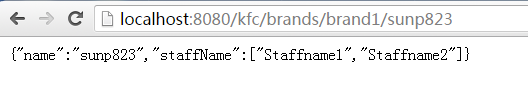
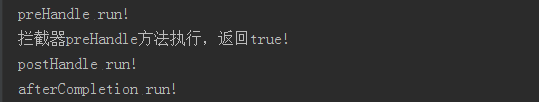
public void setStaffName(String[] staffName) {

this.staffName = staffName;

}

}

**运行结果：**

   
控制台输出：   


版权声明：本文为博主原创文章，未经博主允许不得转载。