# Controller方法返回值



不管是Model还是ModelAndView，其本质都是使用Request对象向jsp传递数据。

## 返回ModelAndView

controller方法中定义ModelAndView对象并返回，对象中可添加model数据、指定view。



## 返回void

在Controller方法形参上可以定义request和response，使用request或response指定响应结果：

1、使用request转发页面，如下：

request.getRequestDispatcher("页面路径").forward(request, response);

request.getRequestDispatcher("/WEB-INF/jsp/success.jsp").forward(request, response);

2、可以通过response页面重定向：

response.sendRedirect("url")

response.sendRedirect("/springmvc-web2/itemEdit.action");

1. 可以通过response指定响应结果，例如响应json数据如下：

response.getWriter().print("{\"abc\":123}");

以下代码一次测试，演示上面的效果

/\*\*

\* 返回void测试

\*

\* **@param** request

\* **@param** response

\* **@throws** Exception

\*/

@RequestMapping("queryItem")

**public** **void** queryItem(HttpServletRequest request, HttpServletResponse response) **throws** Exception {

// 1 使用request进行转发

// request.getRequestDispatcher("/WEB-INF/jsp/success.jsp").forward(request,

// response);

// 2 使用response进行重定向到编辑页面

// response.sendRedirect("/springmvc-web2/itemEdit.action");

// 3 使用response直接显示

response.getWriter().print("{\"abc\":123}");

}

## 返回字符串--map

### 逻辑视图名

controller方法返回字符串可以指定逻辑视图名，通过视图解析器解析为物理视图地址。

//指定逻辑视图名，经过视图解析器解析为jsp物理路径：/WEB-INF/jsp/itemList.jsp

**return** "itemList";

### Redirect重定向

Contrller方法返回字符串可以重定向到一个url地址

如下商品修改提交后重定向到商品编辑页面。

/\*\*

\* 更新商品

\*

\* **@param** item

\* **@return**

\*/

@RequestMapping("updateItem")

**public** String updateItemById(Item item) {

// 更新商品

**this**.itemService.updateItemById(item);

// 修改商品成功后，重定向到商品编辑页面

// 重定向后浏览器地址栏变更为重定向的地址，

// 重定向相当于执行了新的request和response，所以之前的请求参数都会丢失

// 如果要指定请求参数，需要在重定向的url后面添加 ?itemId=1 这样的请求参数

**return** "redirect:/itemEdit.action?itemId=" + item.getId();

}

### forward转发

Controller方法执行后继续执行另一个Controller方法

如下商品修改提交后转向到商品修改页面，修改商品的id参数可以带到商品修改方法中。

/\*\*

\* 更新商品

\*

\* **@param** item

\* **@return**

\*/

@RequestMapping("updateItem")

**public** String updateItemById(Item item) {

// 更新商品

**this**.itemService.updateItemById(item);

// 修改商品成功后，重定向到商品编辑页面

// 重定向后浏览器地址栏变更为重定向的地址，

// 重定向相当于执行了新的request和response，所以之前的请求参数都会丢失

// 如果要指定请求参数，需要在重定向的url后面添加 ?itemId=1 这样的请求参数

// return "redirect:/itemEdit.action?itemId=" + item.getId();

// 修改商品成功后，继续执行另一个方法

// 使用转发的方式实现。转发后浏览器地址栏还是原来的请求地址，

// 转发并没有执行新的request和response，所以之前的请求参数都存在

**return** "forward:/itemEdit.action";

}

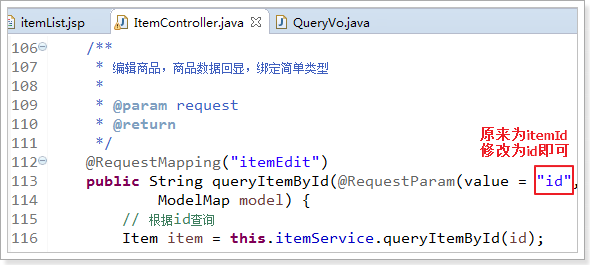
//结果转发到editItem.action，request可以带过去

**return** "forward: /itemEdit.action";

需要修改之前编写的根据id查询商品方法

因为请求进行修改商品时，请求参数里面只有id属性，没有itemId属性

修改，如下图：：



# 转发和重定向--handler

## 转发-- forward

### 本类转发

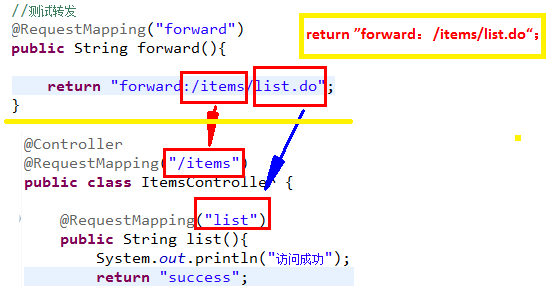
方式一：return ”forward：list.do“；

方式二：return ”forward：/user/list.do“；



### 跨类转发

转发方式：return ”forward：/items/list.do“；



## 重定向-- redirect

### 本类重定向

重定向方式：

方式一：return ”redirect：list.do“；

方式二：return ”redirect：/user/list.do“；



### 跨类重定向

转发方式：return ”redirect：/items/list.do“；

