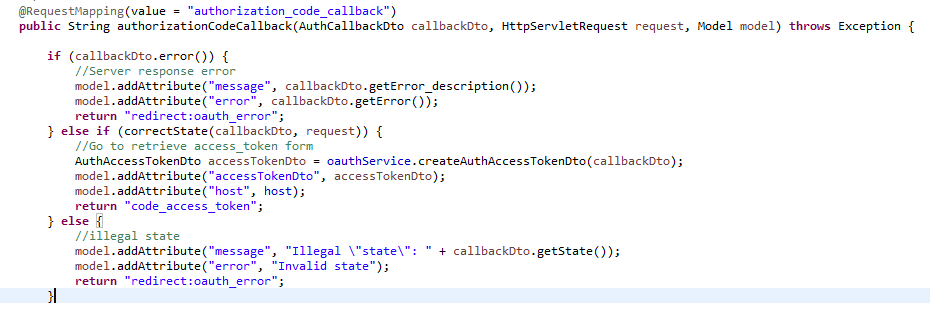
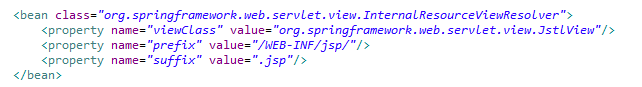
**[Spring MVC 中的 forward 和 redirect](https://www.cnblogs.com/lexiaofei/p/7044429.html)**

****



        Spring MVC 中，我们在返回逻辑视图时，框架会通过 viewResolver 来解析得到具体的 View，然后向浏览器渲染。通过配置，我们**配置某个 ViewResolver 如下：**

Xml代码  [收藏代码](http://blog.csdn.net/zzjjiandan/article/details/19491253)

1. <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
2. <description>
3. 假如逻辑试图名为 "hello"，因此 viewResolver 将解析成 /WEB-INF/jsp/hello.jsp
4. </description>
5. <property name="order" value="10" />
6. <property name="prefix" value="/WEB-INF/jsp/" />
7. <property name="suffix" value=".jsp" />
8. </bean>

* 如果逻辑视图名为 hello，框架通过 forward 的方式转发到了 /WEB-INF/jsp/hello.jsp。
* 如果逻辑视图名是 /hello，框架还是转发到了 /WEB-INF/jsp/hello.jsp，即 /WEB-INF/jsp//hello.jsp 等同于 /WEB-INF/jsp/hello.jsp。

        现在有个问题，如果 /hello 就是某个 controller 的映射，我想转发到这个 controller，怎么办？

**我们可以通过forward 前缀来达到转发到其它资源的目的：**

Java代码  [收藏代码](http://blog.csdn.net/zzjjiandan/article/details/19491253)

1. **public String handle() {**
2. // return "forward:/hello" => 转发到能够匹配 /hello 的 controller 上
3. // return "hello" => 实际上还是转发，只不过是框架会找到该逻辑视图名对应的 View 并渲染
4. // return "/hello" => 同 return "hello"
5. **return "forward:/hello";**
6. }

        同理，如果我们想重定向到某个资源，我们可以通过 **redirect 前缀来达到重定向到其它资源的目的：**

Java代码  [收藏代码](http://blog.csdn.net/zzjjiandan/article/details/19491253)

1. **public String handle() {**
2. // 重定向到 /hello 资源
3. **return "redirect:/hello";**
4. }

下面是关于redirect和forward方式的区别：

**redirect**方式相当于”response.sendRedirect()”. 这种方式浏览器地址栏最后显示的路径是转发后的新的路径。工作方式是这样的, **服务器端会首先发一个response给浏览器, 然后浏览器收到这个response后再发一个requeset给服务器, 然后服务器发新的response给浏览器. 这时页面收到的request对象是重新从浏览器发来的**。这种方式的特点：

* A.在转发前后有两个不同的request对象,转发前后的两个控制器在request上的参数(request.getParameter())和request属性(request.getAttribute())不能共享;
* B.如果转发前后的两个控制器都配置在spring 拦截器范围内,这样拦截器会拦截前后两个request,即会拦截两次;
* C.最后返回到浏览器后,因为地址栏显示的是转发后的url,所以刷新页面时只会执行后面的url映射的控制器.

**forward**方式相当于request.getRequestDispatcher().**forward**(request,response)，这种方式的外部特征是浏览器地址显示的路径是转发前的路径。工作方式是这样,**forward 发生在服务器内部,在前一个控制器处理完毕后,直接进入下一个控制器处理, 并将最后的response发给浏览器.**这种方式的特点：

* A.转发前后是同一个request,后一个控制器可共享前一个控制器的参数与属性;
* B.因为是同一个request,拦截器只会拦截前一个url,如果前一个url在映射时未配置到拦截器拦截，则拦截后一个url，即只拦截一次;
* C.最后返回到浏览器后,因为地址栏显示的是转发前的url,所以刷新页面时会依次执行前后两个控制器.