SpringMVC中的***Interceptor拦截器***是非常重要和相当有用的，它的主要作用是***拦截指定的用户请求，并进行相应的预处理与后处理***。其拦截的时间点在“处理器映射器根据用户提交的请求映射出了所要执行的处理器类，并且也找到了要执行该处理器类的处理器适配器，***在处理器适配器执行处理器之前***”。当然，***在处理器映射器映射出所要执行的处理器类时，已经将拦截器与处理器组合为了一个处理器执行链，并返回给了中央调度器。***

1. **一个拦截器的执行**
2. 自定义拦截器

自定义拦截器，需要实现HandlerInterceptor接口。而该接口中含有三个方法：

* preHandle(request,response,Object handler)：

该方法在处理器方法执行之前执行。其返回值为boolean，***若为true，则紧 接着会执行处理器方法***，且会将afterCompletion()方法放入到一个专门的方 法栈中等待执行。

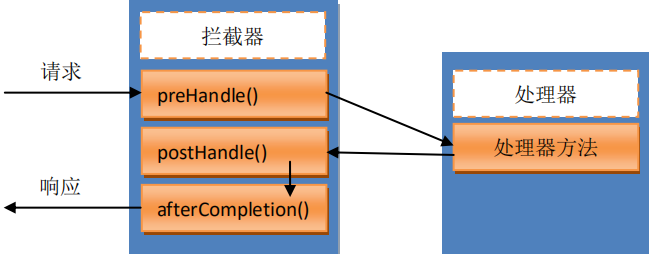
* postHandle(request,response,Object handler,modelAndView)：

该方法在处理器方法执行之后执行。处理器方法若最终未被执行，则该方法 不会执行。由于该方法是在处理器方法执行完后执行，且该方法参数中包含 ModelAndView，所以该方法可以修改处理器方法的处理结果数据，且可以修 改跳转方向。

* afterCompletion(request,response,Object handler,Exception ex)：

当preHandle()方法返回true时，会将该方法放到专门的方法栈中，等到对请 求进行响应的所有工作完成之后才执行该方法。即该方法是在中央调度器渲 染（数据填充）了响应页面之后执行的，此时对ModelAndView再操作也对 响应无济于事。

拦截器中方法与处理器方法的执行顺序：

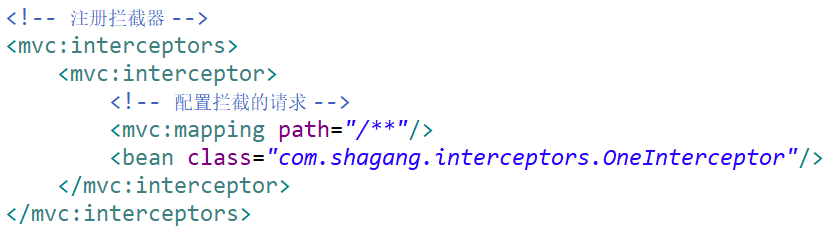


换一种一表现方式，也可以这样理解：



1. 注册拦截器

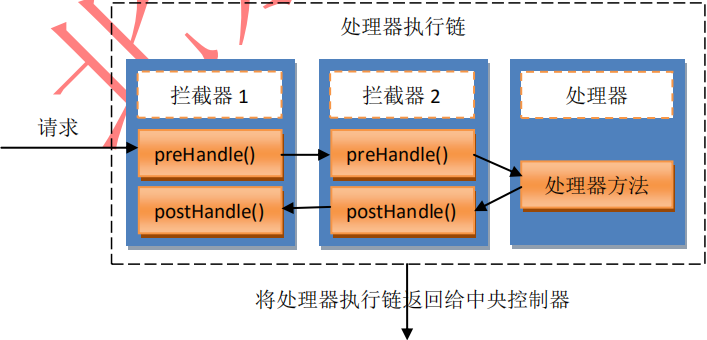
***<mvc:mapping/>用于指定当前所注册的拦截器可以拦截的请求路径，而/\*\*表示拦截所有请求。***

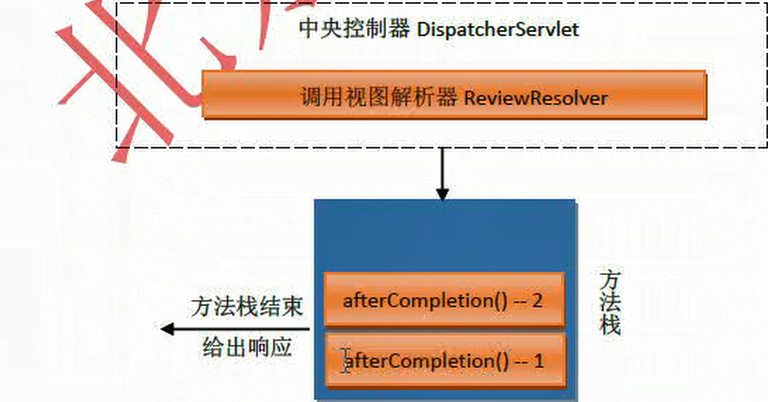


1. 修改index页面
2. 修改处理器
3. 修改show页面
4. 控制台输出结果
5. **多个拦截器的执行**
6. 再定义一个拦截器
7. 多个拦截器的注册与执行
8. 控制台执行结果

当有多个拦截器时，***形成拦截器链***。***拦截器链的执行顺序，与其注册顺序一致。***需要再次强调一点的是，当某一个拦截器的preHandle()方法返回true并被执行到时，会向一个专门的方法栈中放入该拦截器的afterCompletion()方法。

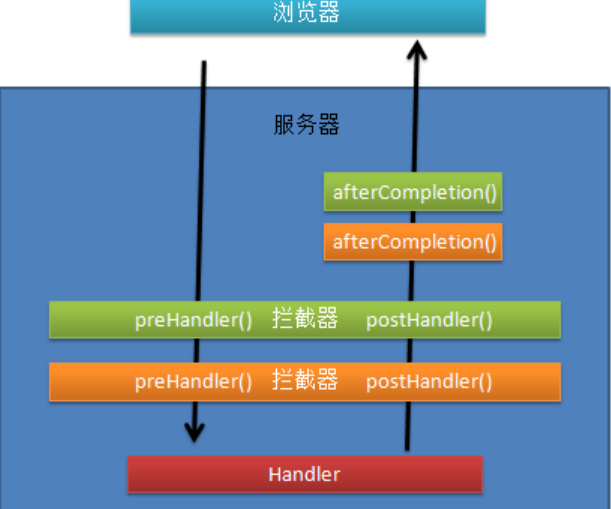
多个拦截器中方法与处理器方法的执行顺序如下图：





从图中可以看出，只要有一个preHandle()方法返回false，则上部的执行链将被断开，其后续的处理器方法与postHandle()方法将无法执行。但，***无论执行链执行情况怎样，只要方法栈中有方法，即执行链中只要有preHandle()方法返回true，就会执行方法栈中的afterCompletion()方法。最终都会给出响应。***

换一种一表现方式，也可以这样理解：



1. 阅读源码
2. **权限拦截器举例**

只有经过登录的用户方可访问处理器，否则，将返回“无权访问”提示。

本例的登录，由一个JSP页面完成。即***在该页面里将用户信息放入session中***。也就是说，只要访问过该页面，就说明登录了。没访问过，则为未登录用户。

1. 定义index页面
2. 定义Controller
3. 定义welcome页面
4. 定义权限拦截器
5. 定义fail页面
6. 注册权限拦截器
7. 定义login页面
8. 定义logout页面
9. 项目测试