一. SpringMVC 中的Interceptor 拦截器

作用:  SpringMVC 中的Interceptor 拦截器也是相当重要和相当有用的，它的主要作用是拦截用户的请求并进行相应的处理。比如通过它来进行权限验证，或者是来判断用户是否登陆，或者是像12306 那样子判断当前时间是否是购票时间。

 1.1定义Interceptor实现类

   SpringMVC 中的Interceptor 拦截请求是通过HandlerInterceptor 来实现的。在SpringMVC 中定义一个Interceptor 非常简单，主要有两种方式，第一种方式是要定义的Interceptor类要实现了Spring 的HandlerInterceptor 接口，或者是这个类继承实现了HandlerInterceptor 接口的类，比如Spring 已经提供的实现了HandlerInterceptor 接口的抽象类HandlerInterceptorAdapter ；第二种方式是实现Spring的WebRequestInterceptor接口，或者是继承实现了WebRequestInterceptor的类。

   （一）实现HandlerInterceptor接口

   HandlerInterceptor 接口中定义了三个方法，我们就是通过这三个方法来对用户的请求进行拦截处理的。

   （1 ）preHandle (HttpServletRequest request, HttpServletResponse response, Object handle) 方法，顾名思义，该方法将在请求处理之前进行调用。SpringMVC 中的Interceptor 是链式的调用的，在一个应用中或者说是在一个请求中可以同时存在多个Interceptor 。每个Interceptor 的调用会依据它的声明顺序依次执行，而且最先执行的都是Interceptor 中的preHandle 方法，所以可以在这个方法中进行一些前置初始化操作或者是对当前请求的一个预处理，也可以在这个方法中进行一些判断来决定请求是否要继续进行下去。该方法的返回值是布尔值Boolean 类型的，当它返回为false 时，表示请求结束，后续的Interceptor 和Controller 都不会再执行；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 ）postHandle (HttpServletRequest request, HttpServletResponse response, Object handle, ModelAndView modelAndView) 方法，由preHandle 方法的解释我们知道这个方法包括后面要说到的afterCompletion 方法都只能是在当前所属的Interceptor 的preHandle 方法的返回值为true 时才能被调用。postHandle 方法，顾名思义就是在当前请求进行处理之后，也就是Controller 方法调用之后执行，但是它会在DispatcherServlet 进行视图返回渲染之前被调用，所以我们可以在这个方法中对Controller 处理之后的ModelAndView 对象进行操作。postHandle 方法被调用的方向跟preHandle 是相反的，也就是说先声明的Interceptor 的postHandle 方法反而会后执行，这和Struts2 里面的Interceptor 的执行过程有点类型。Struts2 里面的Interceptor 的执行过程也是链式的，只是在Struts2 里面需要手动调用ActionInvocation 的invoke 方法来触发对下一个Interceptor 或者是Action 的调用，然后每一个Interceptor 中在invoke 方法调用之前的内容都是按照声明顺序执行的，而invoke 方法之后的内容就是反向的。

   （3 ）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。顾名思义，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。这个方法的主要作用是用于进行资源清理工作的。

Spring框架的Interceptor与Servlet的Filter有相似之处,比如二者都是AOP编程思想的提现,都能实现权限检查,日志记录等,不同的是:

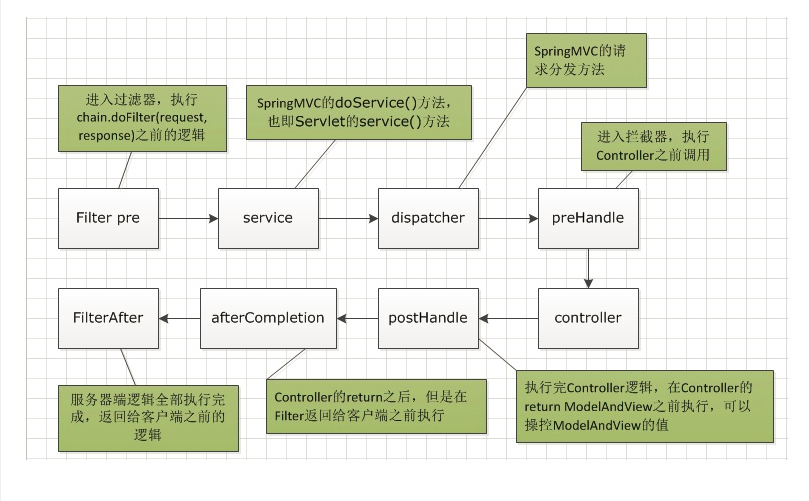
1. 使用范围不同:Filter是Servlet规范规定的,只能用于Web程序中,而拦截器既可以用于Web程序,也可以用于Application,Swing程序中;
2. 规范不同:Filter是在Servlet规范中定义的,是Servlet容器支持的,而拦截器是在Spring容器内的,是Spring框架支持的;
3. 使用的资源不同:同其他的代码块一样,拦截器也是一个Spring的组件间,归Spring管理,配置在Spring文件中,一次能使用Spring里的任何资源,对象,例如Service对象,数据源,事务管理等,通过IOC注入到拦截器即可;而Filter则不能;
4. 深度不同:Filter只能在Servlet前后起作用.而拦截器能够深入到方法前后,异常抛出前后等,因此拦截器的使用具有更大的弹性,所以在Spring框架的程序中,要优先使用拦截器;
5. 拦截器是基于java的反射机制的，而过滤器是基于函数回调。
6. 拦截器是AOP的一种实现,底层通过动态代理模式实现;
7. 拦截器只能对servlet请求起作用,而过滤器则可以对几乎所有的请求起作用;
8. 拦截器可以访问Servlet上下文,值栈里的对象,而过滤器不能;
9. 在Servlet的生命周期中,拦截器可以多次被调用;
10. 多个的执行顺序:根据filter mapping配置的先后顺序;根据在springmvc.xml中配置的顺序,但是可以通过order控制顺序;

二.Filter与Servlet的对比:

实际上Filter和Servlet极其相似，区别只是Filter不能直接对用户生成响应。实际上Filter里doFilter()方法里的代码就是从多个Servlet的service()方法里抽取的通用代码，通过使用Filter可以实现更好的复用。

filter是一个可以复用的代码片段，可以用来转换[HTTP请求](https://www.baidu.com/s?wd=HTTP%E8%AF%B7%E6%B1%82&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YkuH6snhP9PWfdPv79uyRs0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHbYrHDknWf1rHc1n16kPjb3r0)、响应和头信息。Filter不像Servlet，它不能产生一个请求或者响 应，它只是修改对某一资源的请求，或者修改从某一的响应。

***Filter/Servlet/Interceptor的执行顺序图***



https://blog.csdn.net/chenleixing/article/details/44573495

<https://elim.iteye.com/blog/1750680>

https://blog.csdn.net/sundenskyqq/article/details/8549932