**拦截器（Intercepter）简介：**

1. Spring MVC的拦截器（Intercepter）类似于Java Servlet中的过滤器（Filter），用 于拦截用户的请求并做出相应的处理。
2. 拦截器常应用于登录验证、编码处理、权限验证、记录请求信息日志等。

**定义拦截器：**

定义一个拦截器有两种方式：一种是通过实现HandlerInterceptor接口或继承 HandlerInterceptor接口的实现类来定义，另一种是通过实现 WebRequestInterceptor接口或继承WebRequestInterceptor接口的实现类来定义。例如 下面通过实现HandlerInterceptor接口来定义拦截器，如下：



实现了HandlerInterceptor接口后，覆写其中的3个方法：

·preHandle( )：该方法在控制器的处理请求方法前执行，其返回值表示是否继续 向下操作，返回true表示继续向下执行，返回false表示中断 后续操作。

·postHandle( )：该方法在控制器的请求处理方法调用后、解析视图之前执行， 可以通过此方法对请求域中的模型和视图做进一步修改。

·afterCompletion( )：该方法在控制器的处理请求方法执行完成后，视图渲染结束 后执行，可以通过此方法实现一些资源清理、记录日志信息 等工作。

以上3个方法并不是强制性的需要覆写，但至少应该覆写一个方法，这样才使得该拦截 器的存在有意义。

**配置拦截器：**

自定义拦截器后，还需要在Spring MVC的配置文件中进行配置，使其生效。配置代码 如下：



在上述中一共配置了3个拦截器。

1. <mvc:interceptors>标签中用于配置一组拦截器，所有的拦截器都需要配置在该标签 体内。

2）<bean>标签定义的是全局拦截器，会对所以的请求进行拦截；<bean>的class属性用 于指定定义的拦截器所在路径。

3）<mvc:interceptor>标签可以自定义拦截器作用的路径，其中

<mvc:mapping>用于配置拦截器作用的路径，由属性path指定；当path值为“/\*\*”时，表示拦截所有请求；当path值为“/goTest”时，表示拦截所有以“/goTes t”结尾的路径。

<mvc:exclude-mapping>标签用于配置拦截器不需要拦截的路径。

<bean>标签用于指定定义的拦截器所在路径。

**拦截器的执行流程：**

根据拦截器的配置数量多少与配置的顺序，拦截器执行的流程也有所区别。

1. 当配置了一个拦截器时，程序首先执行拦截器中的preHandle( )方法，如果此方法返 回true，程序将继续执行控制中处理请求方法，当请求处理方法执行后，在返回视 图前将执行postHandle( )方法，返回视图后才执行afterCompletion( )方法。
2. 当配置了多个拦截器时，将按照拦截器配置的顺序一个接着一个执行 preHandle( )方法，如果此时有某个preHandle( )方法返回值为false，则中断；当 全部的preHandle( )方法执行完成后，将执行请求处理方法。而它们的postHandle( ) 方法和afterCompletion( )方法则按照配置顺序的反序执行。

**拦截器的应用：**

拦截器在开发中最常用的应用有登录验证、编码处理、权限验证、记录请求信息日志等。

如下是拦截器的作用是用于登录验证。



