1. Filter是拦截Request请求的对象：在用户请求访问资源前处理ServletRequest和ServletResponse，它用于日志记录、加解密、Session检查、图像文件保护等。
2. Filter的实现必须继承javax.servlet.Filter接口，这个接口保护三个生命周期：init、doFilter、destory

•void init(FilterConfig filterConfig):在容器初始化Filter时被调用，一般是在应用开始时。

* FilterConfig是由Servlet容器传进init方法中的。
* FilterConfig有ServletContext getServletContext()方法。
* FilterConfig的String getFilterName()可以获取Filter的名称。
* FilterConfig的 Enumeration<String> getInitParameterNaems()方法获取参数的名称集合。
* FilterConfig的 String getInitParameter(String name)方法获取对于名称参数的值。

•void doFilter(ServletRequest request, ServletResponse response, FilterChain filterChain):

* 当Servlet容器每次处理Filter相关资源时，都会调用gaiFilter实例的doFilter方法。
* 在doFilter的实现中，最后一行需要调用FilterChain的doFilter方法。FilterChain.doFilter(request,response)
* 在FilterChain的最后一个Filter调用FilterChain.doFilter()时才会触发处理资源方法。
* FilterChain接口只有一个方法：doFilter(ServletRequest, ServletResponse)

•void destroy()：在Servlet容器要销毁Filter时触发，一般在应用停止时触发。

1. 配置Filter的步骤：
   1. 确认哪些资源需要使用这个Filter拦截。
   2. 配置Filter的初始化参数，这些参数可以在Filter的init方法的FilterConfig中读取到。
   3. 给Filter取一个名称。
   4. 使用@WebFilter

filterName String 指定过滤器的 name 属性，等价于 <filter-name>

value String[] 该属性等价于 urlPatterns 属性。但是两者不应该同时使用。urlPatterns String[] 指定一组过滤器的 URL 匹配模式。等价于 <url-pattern> 标签。

servletNames String[] 指定过滤器将应用于哪些 Servlet。取值是 @WebServlet 中的 name 属性的取值，或者是 web.xml 中 <servlet-name> 的取值。

dispatcherTypes DispatcherType 指定过滤器的转发模式。具体取值包括：

ASYNC、ERROR、FORWARD、INCLUDE、REQUEST。

initParams WebInitParam[] 指定一组过滤器初始化参数，等价于 <init-param> 标签。

asyncSupported Boolean 声明过滤器是否支持异步操作模式，等价于 <async-supported> 标签。

Description String 该过滤器的描述信息，等价于 <description> 标签。

displayName String 该过滤器的显示名，通常配合工具使用，等价于 <display-name> 标签。

largeIcon String Filter的大图名称

smallIcon String Filter的小图名称

示例：

@WebFilter(filterName = "DataCompressionFilter", urlPatterns = {"/\*"})  
public class DataCompressionFilter implements Filter {}

使用部署描述

<filter>  
 <filter-name>DataCompressionFilter</filter-name>  
 <filter-class>filterclass</filter-class>  
</filter>  
<filter-mapping>  
 <filter-name>DataCompressionFilter</filter-name>  
 <url-pattern>/\*</url-pattern>  
</filter-mapping>

1. Filter顺序

链式Filter必须使用部署描述来管理Filter,部署在前面的Filter会先被触发。

示例:

<filter>  
 <filter-name>filter1</filter-name>  
 <filter-class></filter-class>  
</filter>  
<filter>  
 <filter-name>filter2</filter-name>  
 <filter-class></filter-class>  
</filter>

filter1会比filter2先被触发