# 监听器

一、监听对象创建/销毁的监听器接口

Interface ServletRequestListener 监听request对象的创建或销毁

Interface HttpSessionListener 监听session对象的创建或销毁

Interface ServletContextListener 监听servletContext对象的创建或销毁

二、监听对象属性的变化

Interface ServletRequestAttributeListener 监听request对象属性变化: 添加、移除、修改

Interface HttpSessionAttributeListener 监听session对象属性变化: 添加、移除、修改

Interface ServletContextAttributeListener 监听servletContext对象属性变化

总结：先写类，实现接口； 再配置

配置：

<listener>

<listener-class>实现类全名 </listener-class>

</listener>

三、session相关监听器

Interface HttpSessionBindingListener 监听对象绑定到session上的事件

实体类 实现该接口，当实体类加入到session中时触发，对象在session中移除时触发

对象实现接口； 再把对象绑定/解除绑定到session上就会触发监听代码。

这个session监听器，和上面的声明周期、属性监听器区别？

🡪 不用再web.xml配置

🡪 因为监听的对象是自己创建的对象，不是服务器对象！

Interface HttpSessionActivationListener(了解) 监听session序列化及反序列化的事件

# 过滤器

项目开发中涉及到重复代码的实现（如：验证用户是否登录）。

|-- interface Filter 过滤器核心接口

Void init(filterConfig); 初始化方法，在服务器启动时候执行

Void doFilter(request,response,filterChain); 过滤器拦截的业务处理方法

Void destroy(); 销毁过滤器实例时候调用

|-- interface FilterConfig 获取初始化参数信息

|  |  |
| --- | --- |
| String | [**getInitParameter**](mk:@MSITStore:E:\API\servlet-api_en.chm::/javax/servlet/FilterConfig.html#getInitParameter(java.lang.String))(java.lang.String name) |
| Enumeration | [**getInitParameterNames**](mk:@MSITStore:E:\API\servlet-api_en.chm::/javax/servlet/FilterConfig.html#getInitParameterNames())**()** |

|-- interface FilterChain 过滤器链参数；一个个过滤器形成一个执行链；

void doFilter(ServletRequest request, ServletResponse response) ; 执行下一个过滤器或放行



多个过滤器时，设置直接返回还是进入下一个过滤器使用：

chain.doFilter(request, response);

web服务器根据Filter在web.xml文件中的注册顺序，决定先调用哪个Filter，当第一个Filter的doFilter方法被调用时，web服务器会创建一个代表Filter链的FilterChain对象传递给该方法。在doFilter方法中，开发人员如果调用了FilterChain对象的doFilter方法，则web服务器会检查FilterChain对象中是否还有filter，如果有，则调用第2个filter，如果没有，则调用目标资源。