**Filter接口**

　　Servlet API提供了一个Filter接口，编写的过滤器必须实现该接口。

**过滤器如何实现拦截**

* 当客户端发生请求后，在HttpServletRequest 到达Servlet 之前，过滤器拦截客户的HttpServletRequest 。
* 根据需要检查HttpServletRequest ，也可以修改HttpServletRequest 头和数据。
* 在过滤器中调用doFilter方法，对请求放行。请求到达Servlet后，对请求进行处理并产生HttpServletResponse发送给客户端。
* 在HttpServletResponse 到达客户端之前，过滤器拦截HttpServletResponse 。
* 根据需要检查HttpServletResponse ，可以修改HttpServletResponse 头和数据。
* 最后，HttpServletResponse到达客户端。

**Filter的生命周期**

**（1）Filter接口中有三个重要的方法。**

* init()方法：初始化参数，在创建Filter时自动调用。当我们需要设置初始化参数的时候，可以写到该方法中。
* doFilter()方法：拦截到要执行的请求时，doFilter就会执行。这里面写我们对请求和响应的预处理。
* destroy()方法：在销毁Filter时自动调用。
* 这三个方法都由Server自动调用，不同我们操心

**（2）Filter的生命周期**

　　  Filter的创建和销毁由web服务器控制。

* 服务器启动的时候，web服务器创建Filter的实例对象，并调用其init方法，完成对象的初始化功能。filter对象只会创建一次，init方法也只会执行一次。
* 拦截到请求时，执行doFilter方法。可以执行多次。
* 服务器关闭时，web服务器销毁Filter的实例对象。

**Filter对象——FilterConfig（同ServletConfig）**

　　用 户在配置filter时，可以使用<init-param>为filter配置一些初始化参数，当web容器实例化Filter对象，调用其 init方法时，会把封装了filter初始化参数的filterConfig对象传递进来。因此开发人员在编写filter时，通过 filterConfig对象的方法，就可获得：

* String getFilterName()：得到filter的名称。
* String getInitParameter(String name)： 返回在部署描述中指定名称的初始化参数的值。如果不存在返回null.
* Enumeration getInitParameterNames()：返回过滤器的所有初始化参数的名字的枚举集合。
* public ServletContext getServletContext()：返回Servlet上下文对象的引用。