**java防止xss攻击，且富文本框内容可用（带白名单）教程**

**1 - 过滤器类XssFilter**

**package** com.shop.xss\_sq;

**import** java.io.IOException;

**import** javax.servlet.Filter;

**import** javax.servlet.FilterChain;

**import** javax.servlet.FilterConfig;

**import** javax.servlet.ServletException;

**import** javax.servlet.ServletRequest;

**import** javax.servlet.ServletResponse;

**import** javax.servlet.http.HttpServletRequest;

**public** **class** XssFilter **implements** Filter {

FilterConfig filterConfig = **null**;

**public** **void** init(FilterConfig filterConfig) **throws** ServletException {

**this**.filterConfig = filterConfig;

}

**public** **void** destroy() {

**this**.filterConfig = **null**;

}

//对request进行包装

**public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain) **throws** IOException, ServletException {

chain.doFilter(**new** XssHttpServletRequestWrapper((HttpServletRequest) request), response);

}

}

**2 – XssHttpServletRequestWrapper类 继承HttpServletRequestWrapper**

**package** com.shop.xss\_sq;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletRequestWrapper;

**import** org.apache.oro.text.perl.Perl5Util;

**public** **class** XssHttpServletRequestWrapper **extends** HttpServletRequestWrapper {

**public** XssHttpServletRequestWrapper(HttpServletRequest servletRequest) {

**super**(servletRequest);

}

**public** String[] getParameterValues(String parameter) {

String[] values = **super**.getParameterValues(parameter);

**if** (values == **null**) **return** **null**;

**int** count = values.length;

String[] encodedValues = **new** String[count];

**for** (**int** i = 0; i < count; i++) {

encodedValues[i] = cleanXSS(values[i]);

}

**return** encodedValues;

}

**public** String getParameter(String parameter) {

String value = **super**.getParameter(parameter);

**if** (value == **null**) **return** **null**;

**return** cleanXSS(value);

}

// public String getHeader(String name) {

// String value = super.getHeader(name);

// if (value == null) return null;

//

// return cleanXSS(value);

//

// }

//这里可以自己实现转义，也可以直接用工具类进行转义，比如说org.apache.common.lang.StringEscapeUtils和org.springframework.web.util.HtmlUtils

**private** String cleanXSS(String value) {

/\*StringBuilder buffer = new StringBuilder(value.length() + 16);

for (int i = 0; i < value.length(); i++) {

char c = value.charAt(i);

switch (c) {

case '>':

buffer.append(">");// 转义大于号

break;

case '<':

buffer.append("<");// 转义小于号

break;

case '\'':

buffer.append("'");// 转义单引号

break;

case '\"':

buffer.append("\""); // 转义双引号

break;

case '&':

buffer.append("&");// 转义&

break;

default:

buffer.append(c);

break;

}

}

return buffer.toString();\*/

StringBuffer buffer = **new** StringBuffer();

Perl5Util preg = **new** Perl5Util();

// preg.substitute(buffer,"s/<script[^>]\*?>.\*?<\\/script>//gmi",value);

//(?# a,/a,img/br ...标签之外,都删除)

preg.substitute(buffer,"s#<[/]\*?(?!a|img|strong|/strong|span|/span|br|/a|table|/table|tr|/tr|td|/td|div|/div|p|/p)[^<>]\*?>#abc#gmi",value);

/\*

preg.substitute(buffer,"s/([\r\n])[\\s]+//gmi",html);

\*/

value =buffer.toString();

buffer.setLength(0);

**return** value;

//直接用spring的HtmlUtils 进行html转义

// if (value != null) {

// value = HtmlUtils.htmlEscape(value);

// }

// return value;

//

}

}

**3 – 配置web.xml**

<filter>

<filter-name>xssFilter</filter-name>

<filter-class>com.shop.xss\_sq.XssFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>xssFilter</filter-name>

<url-pattern>/\*</url-pattern>

<dispatcher>REQUEST</dispatcher>

</filter-mapping>

**4 – 加入依赖pom.xml**

<!-- xss\_sq begin -->

<dependency>

<groupId>oro</groupId>

<artifactId>oro</artifactId>

<version>2.0.8</version>

</dependency>

<!-- xss\_sq end -->