<!-- 未登录权限过滤 web.xml文件中进行配置-->

<filter>

<filter-name>AccessLimit</filter-name> <!-- 过滤器名称 -->

<filter-class>com.zte.sms.filters.AccessLimitFilter</filter-class>

<!-- 过滤器全名 -->

<init-param>

<!-- 初始化参数，在实现类的init函数值通过getInitParameter(name)函数通过param-name获取 -->

<param-name>urls</param-name>

<param-value>/login.jsp,/sysuser/login.do</param-value>

</init-param>

</filter>

<filter-mapping> <!-- 设置过滤器过滤规则 -->

<filter-name>AccessLimit</filter-name> <!-- 过滤器名称 -->

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

<!-- 对制定pattern 形式的url请求进行过滤 -->

</filter-mapping>

//过滤器实现类

**public** **class** AccessLimitFilter **implements** Filter {

String arr[];

@Override

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

arr=**null**;

}

@Override

**public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain)

**throws** IOException, ServletException {

//进行过滤

HttpServletRequest req = (HttpServletRequest) request;

HttpServletResponse resp = (HttpServletResponse) response;

//获取当前请求路径

String url = req.getServletPath();

//查看当前路径是否需要进行过滤，如不过滤则进行放行

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

**if**(url.equals(arr[i])){

chain.doFilter(request, response);

**return**;

}

}

//当前请求路径需要过滤，查看该请求发出时是否包含了可以放行的条件，即用户已经登录过了

HttpSession session = req.getSession(**false**);

**if**(session==**null**||session.getAttribute("user")==**null**){

//当前请求发出时用户并未进行登录，将其过滤返回至登录界面

resp.sendRedirect(req.getContextPath()+url);

**return** ;

}

//已经登录，放行

chain.doFilter(request, response);

}

@Override

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

//初始化函数，获取配置文件web.xml中的初始化不需要过滤的url

String urls = filterConfig.getInitParameter("urls");

//将获取到的url分割放置于数组内

arr = urls.split(",");

}

}