采用第2种注册过滤器方式来控制分页参数的大小(如果请求的分页参数大于50时,过滤器强制设置pagesize为50)

package com.kanghe.ncdcloud.system.business.config;  
  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.core.annotation.Order;  
import org.springframework.stereotype.Component;  
  
import javax.servlet.\*;  
import javax.servlet.annotation.WebFilter;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletRequestWrapper;  
import java.io.IOException;  
  
*/\*\*  
 \** ***@Author:*** *yangl  
 \** ***@Date:*** *2019/4/18 10:25  
 \* 该过滤器用于限制查询分页大小 默认最大50条*

//@Order(1)表示排序

*\*/*@Component  
@Order(1)  
@WebFilter(filterName = "pageSizeFilter", urlPatterns = "/\*")  
public class PageSizeFilter implements Filter {  
  
 @Value("${business.system.service.query.max.page.size:50}")  
 int maxPageSize;  
 */\*\*  
 \* 分页参数常量  
 \*/* private String PAGE\_SIZE = "pageSize";  
  
 @Override  
 public void init(FilterConfig filterConfig) {  
  
 }  
  
 @Override  
 public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException {  
 chain.doFilter(new HttpServletRequestWrapper((HttpServletRequest) request) {  
 @Override  
 public String[] getParameterValues(String name) {  
 if (PAGE\_SIZE.equals(name)) {  
 int pageSize = Integer.*valueOf*(super.getParameter(PAGE\_SIZE));  
 if (pageSize > maxPageSize) {  
 return new String[]{String.*valueOf*(maxPageSize)};  
 }  
 }  
 return super.getParameterValues(name);  
 }  
 }, response);  
 }  
  
 @Override  
 public void destroy() {  
  
 }  
}

**chain.doFilter的原样chain.doFilter(ServletRequest request, ServletResponse response**

**)**

**上面代码的**

**chain.doFilter(xxx, response);**

**其中xxx(也就是个ServletRequest)代表上面代码的**

**new HttpServletRequestWrapper((HttpServletRequest) request) {  
 @Override  
 public String[] getParameterValues(String name) {  
 if (PAGE\_SIZE.equals(name)) {  
 int pageSize = Integer.valueOf(super.getParameter(PAGE\_SIZE));  
 if (pageSize > maxPageSize) {  
 return new String[]{String.valueOf(maxPageSize)};  
 }  
 }  
 return super.getParameterValues(name);  
 }  
 }**

注意:

**chain.doFilter(ServletRequest request, ServletResponse response**

**)**

1. **请求过来的方法的参数都是被封装在ServletRequest,我们目的就是把ServletRequest的某个参数拿出来校验,并设回去**
2. **在HttpServletRequestWrapper里重写父类ServletRequestWrapper的getParameterValues方法(重写getParameterValues方法的作用是当某个请求过来的时候,会执行我这里重写的getParameterValues方法,而不是父类的getParameterValues,并且请求过来的参数都放在该方法的name,所以我们只在这个方法里做手脚就行),方法里通过调用父类的getParameter拿到请求参数的PAGE\_SIZE的值,如果该值大于50时,则返回50字符串集合,否则原路返回(return super.getParameterValues(name),该返回其实也是个字符串集合)(因为我们重写的public String[] getParameterValues(String name)返回就是字符串集合)**

**请求参数有多少个,就执行重写的getParameterValues()方法多少次**