拦截器

实现方式

import org.springframework.stereotype.**Component**;  
import org.springframework.web.servlet.ModelAndView;  
import org.springframework.web.servlet.handler.HandlerInterceptorAdapter;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;**@Component**//添加进IOC容器public class TestInterceptor extends HandlerInterceptorAdapter {  
 **@Override** public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {  
 System.*err*.println(**"前置拦截器。。。controller方法调用之前"**);  
 return super.preHandle(request,response,handler);  
 }  
 **@Override** public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler, ModelAndView modelAndView) throws Exception {  
 System.*err*.println(**"POSTHandle---------------controller方法调用之后，视图渲染之前"**);  
 super.postHandle(request,response,handler,modelAndView);  
 }  
 **@Override** public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler, Exception ex) throws Exception {  
 System.*err*.println(**"请求结束之后，主要清理工作"**);  
 super.afterCompletion(request,response,handler,ex);  
 }  
 /\*  
 \* 该方法是用来处理异步请求，当Controller中有异步请求方法的时候会触发该方法。  
 \* 异步请求先执行preHandle、然后执行afterConcurrentHandlingStarted  
 \* 异步线程完成之后执行preHandle、postHandle、afterCompletion  
 \* \*/  
 **@Override** public void afterConcurrentHandlingStarted(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {  
 super.afterConcurrentHandlingStarted(request, response, handler);  
 }  
}

集成WebMvcConfigurerAdapter，拦截的路径要带两个\*\*，/api/\*\*

import org.springframework.beans.factory.annotation.**Autowired**;  
import org.springframework.context.annotation.**Configuration**;  
import org.springframework.web.servlet.config.annotation.InterceptorRegistry;  
import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;  
**@Configuration**public class MyWebMvcConfigurerAdapter extends WebMvcConfigurerAdapter {  
 **@Autowired** TestInterceptor testInterceptor;  
 **@Override** public void addInterceptors(InterceptorRegistry registry) {  
 registry.addInterceptor(testInterceptor).addPathPatterns(**"/api/\*\*"**);  
 super.addInterceptors(registry);  
 }  
}