链滴

spring boot 通过 aop 实现全局打印请求参数和返回结果

作者：gongxiongzhuang

原文链接：https://ld246.com/article/1551866577094

来源网站：链滴

为了更好的跟踪请求参数和请求连接还有返回的结果，可以使用spring的面向切面aop实现请求的日输出。

第一步：添加spring boot aop 依赖

```
<!-- 面向切面spring aop -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-aop</artifactId>
</dependency>
```

第二步：添加日志切面类

```
/**
 * @Description 打印请求参数和返回结果
 * @Date 2019/2/26 11:37
 * @Created by gongxz
 */
@Aspect
@Configuration
public class LoggerAspect {

    private Logger logger = LoggerFactory.getLogger(this.getClass());

    // 定义切点Pointcut
    @Pointcut("execution(* com.springboot.controller..*.*(..))")
    public void executeService() {
    }

    @Around("executeService()")
    public Object doAround(ProceedingJoinPoint pjp) throws Throwable {
        RequestAttributes ra = RequestContextHolder.getRequestAttributes();
        ServletRequestAttributes sra = (ServletRequestAttributes) ra;
        HttpServletRequest request = sra.getRequest();

        String url = request.getRequestURL().toString();
        String method = request.getMethod();
        //String uri = request.getRequestURI();
        String queryString = request.getQueryString();
        Object[] args = pjp.getArgs();
        String params = "";
        //获取请求参数集合并进行遍历拼接
        if(args.length>0){
            if("POST".equals(method)){
                Object object = args[0];
                Map map = getKeyAndValue(object);
                params = JSON.toJSONString(map);
            }else if("GET".equals(method)){
                params = URLDecoder.decode(queryString,"UTF-8");
            }
        }
```

```
        logger.info("请求开始===地址:"+url);
        logger.info("请求开始===类型:"+method);
        logger.info("请求开始===参数:"+params);

        // result的值就是被拦截方法的返回值
        Object result = pjp.proceed();
        logger.info("请求结束===返回值:" + JSON.toJSON(result));
        return result;
    }

    public static Map<String, Object> getKeyAndValue(Object obj) {
        Map<String, Object> map = new HashMap<>();
        // 得到类对象
        Class userCla = obj.getClass();
        /* 得到类中的所有属性集合 */
        Field[] fs = userCla.getDeclaredFields();
        for (Field f : fs) {
            f.setAccessible(true); // 设置些属性是可以访问的
            Object val;
            try {
                val = f.get(obj);
                // 得到此属性的值
                map.put(f.getName(), val);// 设置键值
            } catch (IllegalArgumentException | IllegalAccessException e) {
                e.printStackTrace();
            }

        }
        return map;
    }



}
```

第二步：直接请求，日志效果如下