SpringAOP开发步骤

1. 添加maven依赖

|  |
| --- |
| *<!-- https://mvnrepository.com/artifact/org.springframework/spring-aop --> <!--spring AOPA相关类-->* <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-aop</artifactId>  <version>4.3.7.RELEASE</version> </dependency>   *<!-- https://mvnrepository.com/artifact/org.aspectj/aspectjweaver -->* <dependency>  <groupId>org.aspectj</groupId>  <artifactId>aspectjweaver</artifactId>  <version>1.9.8.RC2</version>  <scope>runtime</scope> </dependency>  *<!-- https://mvnrepository.com/artifact/aspectj/aspectjrt -->* <dependency>  <groupId>aspectj</groupId>  <artifactId>aspectjrt</artifactId>  <version>1.5.4</version> </dependency>    *<!-- https://mvnrepository.com/artifact/org.springframework/spring-aspects -->* <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-aspects</artifactId>  <version>4.3.7.RELEASE</version> </dependency>    *<!-- https://mvnrepository.com/artifact/aopalliance/aopalliance -->* <dependency>  <groupId>aopalliance</groupId>  <artifactId>aopalliance</artifactId>  <version>1.0</version> </dependency> |

1. 在application.xml配置文件 中开启aop功能

|  |
| --- |
| *<!--开始spring的AOP功能-->*  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:context="http://www.springframework.org/schema/context"  xmlns:aop="http://www.springframework.org/schema/aop"    xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd   http://www.springframework.org/schema/aop  http://www.springframework.org/schema/aop/spring-aop.xsd  ">  <aop:aspectj-autoproxy></aop:aspectj-autoproxy> |
|  |

1. 定义切面类(就是一个普通Java),在切面类中的添加横切关注点(日志，事务管理 ，权限验证)

切面类需要用到@component注解和@Aspect注解

|  |
| --- |
| @Component *//纳入spring容器* @Aspect *//标注该类为切面类* public class LogAop {   *//定义切入点表达式  //以下表达式表示拦截 com.service.UserService 类中的任意方法* @Pointcut("execution(public int com.service.UserService.\*(..) )")  public void poingCut(){};    @Before(value = "poingCut()") //前置通知  public void addLog(){  *//在调用*com.service.UserService*包中的任意类的任意方法前，都打印以下语句* System.*out*.println("添加日志。。。。。。");  }  } |

1. 在切面类中定义切点入(拦截哪些类的哪些方法，添加模切关注点)