1. AOP术语：
   1. 切面（Aspect）：横切关注点（跨越应用程序多个模块的功能）被模块化的特殊对象。
   2. 通知（Advice）：切面必须要完成的工作
   3. 目标（Target）：被通知的对象
   4. 织入（waving）：AOP有三种织入方式

编译期织入：要求使用特殊的JAVA编译器

类装载织入：要求使用特殊的类装载器

动态代理织入：在运行期为目标类添加增强生成子类的方式

Spring采用动态代理织入，而AspectJ采用编译期织入和类装载期织入

* 1. 代理（Proxy）：向目标对象应用通知之后创建的对象
  2. 连接点（JoinPoint）：程序执行的某个特定位置。由两个信息确定：方法表示的程序执行点；相对点表示的方位。每个类都拥有多个连接点，即连接点是程序类中客观存在的事务。
  3. 切点（pointcut）：AOP通过切点定位到特定的连接点。类比：连接点相当于数据库中的记录，切点相当于查询条件。一个切点匹配多个连接点，通过Pointcut接口描述，使用类和方法作为连接点的查询条件。

1. AspecJ支持5种类型的通知注解：
   1. @Before：前置通知，在方法执行之前执行
   2. @After：后置通知，在方法执行之后执行
   3. @AfterReturning：返回通知，在方法返回结构之后执行
   4. @AfterThrowing：异常通知，在方法抛出异常之后执行
   5. @Around：环绕通知，围绕着方法执行
2. 基于注解方式：
   1. 配置AOP时需要在配置文件中加入：<aop:aspect – autoproxy></aop:aspect – autoproxy>是注解起作用。
   2. 把横切关注点的代码抽象到切面的类中，切面要为IOC容器中的bean，即加入@Component注解，还要加入@Aspect注解。
   3. 在方法前加入各种通知注解，注解中要标明execution（“”）的目标范围，具体到方法。
   4. 可以在切面方法中加入JoinPoint参数，可以获取目标方法的参数，方法名等。
   5. 获取目标方法的参数可以使用JoinPoint.args()方法，也可在execution（“”）后面加参的方法，如：@Before（value=”execution( 目标范围 && args(参数1，参数2…) ) ”）,还要在切面方法中加入相应匹配的参数。
3. 后置通知是在连接点完成之后执行，无论目标方法是否发生异常都会执行。
4. 返回通知在目标方法正常执行之后执行，可以访问到方法的返回值。
5. 环绕通知相当于动态代理，可以在环绕通知的方法内写前置通知，后置通知，异常通知。环绕通知的方法内要传ProceedingJoinPoint类型的参数，环绕通知要有返回值。
6. 对于多个切面，可以使用@Order注解指定切面的优先级，值越小，优先级越高。
7. 引介增强是的链接点是类级别的，用过引介增强，我们可以为目标类添加一个接口的实现。
8. Spring提供了六种切点(Pointuct)类型：静态方法切点，动态方法切点，注解切点，表达式切点，流程切点，符合切点
9. 切面(Advisor)：一般切面，切点切面，引介切面。
10. 增强(advice),切点(Pointuct)和切面(Advisor)关系

增强：包含横切代码（即性能监视，日志功能等代码）和连接点（join ）信息（即方 法前，方法后等的方位信息）

切点：根据增强中的连接点信息找出具体的织入点（将横切代码织入的位置），即为 查询条件

切面：切点和切面组合在一起为切面

12.各种切面总结

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **@AspectJ** | **<aop:aspect>** | **Advisor** | **<aop:advisor>** |
| **增强类型** | 前置增强 | @Before | <aop:before> | MethodBeforeAdvice | 同Advisor |
| 后置增强 | @AfterReturning | <aop:after-returning> | AfterReturningAdvice | 同Advisor |
| 环绕增强 | @Around | <aop:around> | MethodInterceptor | 同Advisor |
| 异常增强 | @AfterThrowing | <aop:after-throwing> | ThrowsAdvice | 同Advisor |
| Final增强 | @After | <aop:after> | 无对应接口 | 同Advisor |
| 引介增强 | @DeclareParents | <aop:declare-parents> | IntroductionInterceptor | 同Advisor |
| **切点定义** | | 支持AspectJ切点表达式，可以通过@Pointcut注解定义命名切点 | 支持AspectJ切点表达式，可以通过<aop:pointcut>定义命名切点 | 直接通过基于Pointcut的实现类定义切点 | 和<aop:aspect>  相同，不过切点函数不能绑定参数 |
| **连接点方法入参绑定** | | 使用JoinPoint，ProceedingsJoinPoint连接点对象  使用切点函数指定参数名的绑定 | 同@AspectJ  <aop:after-returning> | 通过增强接口方法入参绑定 | 同Advisor |
| **连接点方法返回值或抛出异常绑定** | | 在后置增强中，使用@AfterReturning的returning成员绑定方法返回值，  在抛出异常增强中，使用@AfterThrowing的throwing成员绑定方法抛出的异常 | 在后置增强中，使用<aop:after-returning>的returning属性绑定方法返回值，  在抛出异常增强中使用<aop:after-throwing>的throwing属性绑定方法抛出的异常。 | 通过增强接口方法入参绑定 | 同Advisor |