前面的代码中所使用的代理对象，均是由ProxyFactoryBean代理工具类生成的。而该代理工具类存在如下缺点：

* 一个代理对象只能代理一个Bean，即如果有两个Bean同时都要织入同一切面，这时，不仅要配置这两个Bean，即两个目标对象，同时还要配置两个代理对象。
* 在客户类中获取Bean时，使用的是代理类的id，而非我们定义的目标对象Bean的id。我们真正想执行的是应该是目标对象。从形式上看，不符合正常逻辑。

Spring提供了自动生成代理器，用于解决ProxyFactoryBean的问题。常用的自动代理生成器有两个：

* 默认advisor自动代理生成器
* Bean名称自动代理生成器

注：自动代理生成器均继承自Bean后处理器BeanPostProcessor。容器中所有Bean在初始化时均会自动执行Bean后处理器中的方法，故其无需id属性。所以自动代理生成器的Bean也没有id属性，客户类直接使用目标对象bean的id。

1. **默认advisor自动代理生成器**

DefaultAdvisorAutoProxyCreator代理的生成方式是，将所有的目标对象与Advisor自动结合，生成代理对象。无需给生成器做任何的注入配置。注意，只能与Advisor配合使用。



1. **Bean名称自动代理生成器**

DefaultAdvisorAutoProxyCreator会为每个目标对象织入所有匹配的Advisor，不具有选择性，且切面只能是顾问Advisor。而BeanNameAutoProxyCreator的代理生成方式是，根据bean的id，来为符合相应名称的类生成相应代理对象，且切面既可以是顾问Advisor又可以是通知Advice。