一，基于aspectJ的注解aop

前置增强：@Before(value=″execution(* cn. java. zhujie. aop. Book. *(..))″)

环绕增强：

```
//环绕增强
	@Around(value="execution(* cn.java.zhujie.aop.Book.*(..))")
	public void around1(ProceedingJoinPoint proceedingJoinPoint) throws Throwable{
		System.out.println("环绕前===========");
		proceedingJoinPoint.proceed();
		System.out.println("环绕后------------");
	}
```

1，使用注解方式实现aop操作

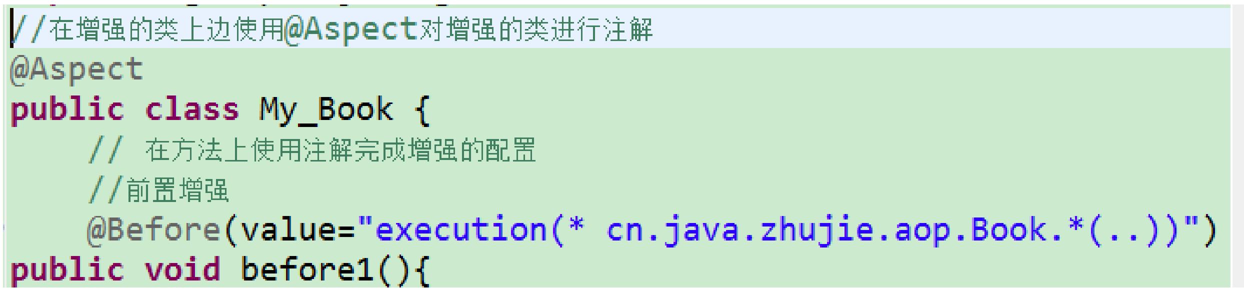
第一步：创建对象

```
<!-- 配置对象 -->
	<bean id="book" class="cn.java.zhujie.aop.Book"></bean>
	<bean id="mybook" class="cn.java.zhujie.aop.My_Book"></bean>
```

第二步：在spring核心配置文件中，开启aop扫描操作

```
<!-- 开启aop操作 -->
	<aop:aspectj-autoproxy></aop:aspectj-autoproxy>
```

第三步：在增强的类上面使用注解完成aop操作



```
//在增强的类上边使用@Aspect对增强的类进行注解
@Aspect
public class My_Book {
    // 在方法上使用注解完成增强的配置
    //前置增强
    @Before(value="execution(* cn.java.zhujie.aop.Book.*(..))")
public void before1(){
```

2，具体的步骤

（1）book实体类

```
package cn.java.zhujie.aop;
public class Book {
	public void add() {
		System.out.println("add----------");
	}
}
```

（2），my_book实体类

```
package cn.java.zhujie.aop;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;
//在增强的类上边使用@Aspect对增强的类进行注解
```

```
@Aspect
public class My_Book {
        // 在方法上使用注解完成增强的配置
        //前置增强
        @Before(value="execution(* cn.java.zhujie.aop.Book.*(..))")
public void before1(){
        System.out.println("before1------------");
}
}
```

（3）配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:aop="http://www.springframework.org/schema/aop"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:jdbc="http://www.springframework.org/schema/jdbc"
xmlns:jee="http://www.springframework.org/schema/jee"
        xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:util="http://www.springframework.org/schema/util"
        xmlns:task="http://www.springframework.org/schema/task"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
        http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.0.xsd
        http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd
        http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd
        http://www.springframework.org/schema/jee
http://www.springframework.org/schema/jee/spring-jee-4.0.xsd
        http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd
        http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util-4.0.xsd
        http://www.springframework.org/schema/task
http://www.springframework.org/schema/task/spring-task-4.0.xsd"
        default-lazy-init="false">
        <!-- 配置对象 -->
        <bean id="book" class="cn.java.zhujie.aop.Book"></bean>
        <bean id="mybook" class="cn.java.zhujie.aop.My_Book"></bean>
        <!-- 开启aop操作 -->
        <aop:aspectj-autoproxy></aop:aspectj-autoproxy>
</beans>
```

（4）测试代码

```
package cn.java.testSet;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import cn.java.zhujie.aop.Book;
public class TestZhujieAop {
@Test
public void testDemo(){
        ApplicationContext context =
                    new ClassPathXmlApplicationContext("bean1.xml");
        Book book = (Book)context.getBean("book");
        book.add();
}
}
```