## 一,后置通知

```
1,配置文件
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</pre>
    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">
    <!-- 开启注解扫描,base-package的值就是需要进行注解开发的类所在的包,若只有
俩三个包的话,可以写cn.java或者cn(1)到指定的包中扫描类、方法、属性上是否有注解
    <context:component-scan base-package="cn.java"></context:component-
scan>
    <!-- 配置对象 -->
    <bean id="book" class="cn.java.aop.Book"></bean>
    <bean id="mybook" class="cn.java.aop.My_Book"></bean>
    <!-- 配置aop配置 -->
    <aop:config proxy-target-class="true">
         <!-- 配置切入点 -->
         <aop:pointcut expression="execution(* cn.java.aop.Book.*(..))"</pre>
             id="MypointCut" />
         <!-- 配置切面:把增强用到方法上去 -->
         <aop:aspect ref="mybook">
             <!-- 配置增强类型: method:增强类里面使用哪个方法作为前置 -->
```

```
<aop:before method="before1" pointcut-ref="MypointCut" />
             <!-- 后置增强 -->
             <aop:after-returning method="after_mehtod" pointcut-
ref="MypointCut"/>
         </aop:aspect>
    </aop:config>
</beans>
2, book实体类
package cn.java.aop;
public class Book {
public void add(){
    System.out.println("add.....");
}
}
3, my_book实体类
package cn.java.aop;
public class My_Book {
public void before1(){
    System.out.println("前置增强.....");
}
public void after_mehtod(){
    System.out.println("这是一个后置增强-----");
}
}
4,测试代码
package cn.java.aop;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class test {
    @Test
    public void t1() {
        ApplicationContext context = new
ClassPathXmlApplicationContext("application.xml");
        Book book = (Book) context.getBean("book");
        book.add();
    }
}
5,运行结果截图
add.....
这是一个后置增强-----
```

## 二,环绕通知

```
1, application. xml配置文件
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</p>
    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">
    <!-- 开启注解扫描,base-package的值就是需要进行注解开发的类所在的包,若只有
俩三个包的话,可以写cn.java或者cn (1)到指定的包中扫描类、方法、属性上是否有注解
-->
    <context:component-scan base-package="cn.java"></context:component-
scan>
    <!-- 配置对象 -->
    <bean id="book" class="cn.java.aop.Book"></bean>
    <bean id="mybook" class="cn.java.aop.My Book"></bean>
    <!-- 配置aop配置 -->
    <aop:config proxy-target-class="true">
         <!-- 配置切入点 -->
         <aop:pointcut expression="execution(* cn.java.aop.Book.*(..))"</pre>
             id="MypointCut" />
         <!-- 配置切面:把增强用到方法上去 -->
         <aop:aspect ref="mybook">
             <!-- 配置增强类型: method:增强类里面使用哪个方法作为前置 -->
             <aop:before method="before1" pointcut-ref="MypointCut" />
```

```
<!-- 后置增强 -->
             <aop:after-returning method="after mehtod" pointcut-
ref="MypointCut"/>
             <!-- 环绕增强 -->
             <aop:around method="around1" pointcut-ref="MypointCut"/>
         </aop:aspect>
    </aop:config>
</beans>
2, book实体类
package cn.java.aop;
public class Book {
public void add(){
    System.out.println("add.....");
}
}
3, my book实体类
package cn.java.aop;
import org.aspectj.lang.ProceedingJoinPoint;
public class My_Book {
    public void before1() {
         System.out.println("前置增强.....");
    public void after_mehtod() {
         System.out.println("这是一个后置增强-----");
    // 环绕通知
    public void around1(ProceedingJoinPoint proceedingJoinPoint) throws
Throwable {
        //方法之前
         System.out.println("方法之前=======");
        //执行被增强的方法
         proceedingJoinPoint.proceed();
        //方法之后
        System.out.println("方法之后=======");
        test around();
    public void test around(){
         System.out.println("方法之后的参数======");
    }
}
4,测试代码
package cn.java.aop;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
```

```
public class test {
   @Test
   public void t1() {
       ApplicationContext context = new
ClassPathXmlApplicationContext("application.xml");
       Book book = (Book) context.getBean("book");
       book.add();
   }
}
5,测试运行结果
前置增强....
方法之前=========
add.....
方法之后=========
方法之后的参数======
这是一个后置增强-----
```