# 建立目标对象及所用增强

## 目标对象接口：

**package** com.yang.springaoppractice.service;

/\*\*

\* 业务模拟接口

\* **@author** Administrator

\*

\*/

**public** **interface** ServiceSimulator {

**void** select();

**void** delete();

}

## 目标对象类：

**package** com.yang.springaoppractice.service;

**import** org.springframework.stereotype.Component;

/\*\*

\* 业务模拟实现类，用于作为添加动态代理的target

\*

\* **@Component**("serviceSimulator") 自动生成该类的bean，名称为"serviceSimulator"

\*

\* **@author** Administrator

\*

\*/

@Component("serviceSimulator")

**public** **class** ServiceSimulatorImpl **implements** ServiceSimulator {

/\*\*

\* 模拟select方法

\*/

@Override

**public** **void** select() {

**try** {

System.***out***.println("Simulating select......");

Thread.*currentThread*().*sleep*(500);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

/\*\*

\* 模拟delete方法

\*/

@Override

**public** **void** delete() {

**try** {

System.***out***.println("Simulating delete......");

Thread.*currentThread*().*sleep*(300);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

## 所用增强(advice)：

**package** com.yang.springaoppractice.advice;

**import** java.lang.reflect.Method;

**import** org.springframework.aop.MethodBeforeAdvice;

**import** org.springframework.stereotype.Component;

/\*\*

\* 前置增强（MethodBeforeAdvice）

\*

\* **@Component**("methodStartReminderAdvice") 自动生成该类的bean，名称为"methodStartReminderAdvice"

\*

\* **@author** Administrator

\*

\*/

@Component("methodStartReminderAdvice")

**public** **class** MethodStartReminderAdvice **implements** MethodBeforeAdvice{

@Override

**public** **void** before(Method arg0, Object[] arg1, Object arg2) **throws** Throwable {

/\*\*

\* 该语句在目标对象方法前执行

\*/

System.***out***.println("The method "

+ arg0.getName()

+ " in the class "

+ arg2.getClass().getName()

+ " is starting......");

}

}

## 第一种：通过定义的Pointcut（切点）

### 定义pointcut：

**package** com.yang.springaoppractice.advisor;

**import** java.lang.reflect.Method;

**import** org.springframework.aop.support.StaticMethodMatcherPointcut;

**import** com.yang.springaoppractice.service.ServiceSimulatorImpl;

/\*\*

\* 静态普通方法名匹配切点切面的切点（可过滤class，method）

\*

\* 定义切点（point cut）

\*/

**public** **class** MethodStartReminderPointcut **extends** StaticMethodMatcherPointcut {

/\*\*

\* 定义只有当目标对象class为ServiceSimulatorImpl.class及调用的方法为select()时，

\* 才触发代理

\*/

@Override

**public** **boolean** matches(Method method, Class<?> targetClass) {

**if**("select".equals(method.getName())

&& ServiceSimulatorImpl.**class**.isAssignableFrom(targetClass)){

**return** **true**;

}

**else**{

**return** **false**;

}

}

}

### 配置xml：

<!-- 配置生成advisor（增强） bean：

通过加入 MethodStartReminderAdvice增强

及MethodStartReminderPointcut切点的配置 -->

<bean id=*"methodStartReminderAdvisor"*

class=*"org.springframework.aop.support.DefaultPointcutAdvisor"*

p:advice-ref=*"methodStartReminderAdvice"*>

<property name=*"pointcut"*>

<bean class=*"com.yang.springaoppractice.advisor.MethodStartReminderPointcut"*/>

</property>

</bean>

## 第二种：通过定义的Advisor（切面）

### 定义advisor：

**package** com.yang.springaoppractice.advisor;

**import** java.lang.reflect.Method;

**import** org.springframework.aop.support.StaticMethodMatcherPointcutAdvisor;

**import** com.yang.springaoppractice.service.ServiceSimulatorImpl;

/\*\*

\* 静态普通方法名匹配切点切面（可过滤class，method）

\*

\* 表面上在此定义的是切面（advisor），实质在此定义的仅有切点（point cut）,没有增强（advice）

\* 增强（advice）是在xml配置中加入的

\*

\* ps:切面（advisor）= 切点（point cut）+ 增强（advice）

\*

\* **@author** Administrator

\*

\*/

**public** **class** MethodStartReminderAdvisor **extends** StaticMethodMatcherPointcutAdvisor {

/\*\*

\* 定义只有当目标对象class为ServiceSimulatorImpl.class及调用的方法为select()时，

\* 才触发代理

\*/

@Override

**public** **boolean** matches(Method method, Class<?> targetClass) {

**if**("select".equals(method.getName())

&& ServiceSimulatorImpl.**class**.isAssignableFrom(targetClass)){

**return** **true**;

}

**else**{

**return** **false**;

}

}

}

### 配置xml：

<!-- 配置生成advisor（增强） bean：

通过加入 MethodStartReminderAdvice增强的配置-->

<bean id=*"methodStartReminderAdvisor"*

class=*"com.yang.springaoppractice.advisor.MethodStartReminderAdvisor"*

p:advice-ref=*"methodStartReminderAdvice"*/>

以上的两种方法生成一模一样的advisor bean

第一种更符合思维逻辑，更容易理解

第二种更简便