利用Spring的AOP记录代码跟踪日志

作者：nontrace
原文链接：https://ld246.com/article/1376231687617
来源网站：链滴

    转眼间又有好一段时间没更新博客了，今天写一篇利用Spring的AOP 来记录代跟踪日志的
一般的程序过都会记录日志，而且会在方法开始和结束的时候都会打一条日志，这样当程序出错时候，可以根据日志信息很快的定位错误，但是这需要在每个方法都要写重复而又单调的记录方法开和结束的代码，
而面向切面的编程（AOP）正好可以完成在方法开始或结束的时候插入代码的工作，而Spring对OP有很好的支持，下面我们就来用Spring的AOP来完成记录代码跟踪日志的工作。

第一、先创建一个javaWeb项目，不一定是Web项目，因为我们这里用不到web的东西，然后所必需的jar包考进来,
spring-context-3.1.3.RELEASE.jar
spring-core-3.1.3.RELEASE.jar
spring-expression-3.1.3.RELEASE.jar
spring-beans-3.1.3.RELEASE.jar
spring-asm-3.1.3.RELEASE.jar
然后还有AOP所需要的包，
spring-aop-3.1.3.RELEASE.jar
spring-aspects-3.1.3.RELEASE.jar
由于Spring是用aspectJ实现的AOP所以还需要aspectJ的包
aspectjrt.jar
aspectjtools.jar
aspectjweaver.jar
org.aspectj.matcher.jar
还有面向切面编程联盟的一个包，aopalliance-1.0.jar
再就是commons-logging.jar包了
总共13个jar包，少一个都不行！！！
第二，编写我们的测试类
    假设我们测试类TestServiceImpl 里的方法添加日志，先写该类的接口ITestService
package com.service;
import java.util.List;
/**
 * 项目名称：test_Spring_AOP
 * 类 名 称：ITest
 * 类 描 述:
 * 建 人：yaohx
 * 创建时间：2013-8-2 上午10:42:02
 * 修 改 人：yaohx
 * 修时间：2013-8-2 上午10:42:02
 * 修改备注：
 * @version
 */
public interface ITestService {
 public void test(String testStr,int age,List list);
然后是TestServiceImpl类

package com.service;
import java.util.List;
/**
 * 项目名称：test_Spring_OP
 * 类 名 称：User
 * 类 描 述:
 * 创 建 人：yaohx
 * 创建时间：2013--2 上午10:08:40
 * 修 改 人：yaohx
 * 修改时间：2013-8-2 上午10:08:40
 * 改备注：
 * @version
 */
public class TestServiceImpl implements ITestService {
 public void test(String testSt,int age,List list){
 System.out.println("Test.test。。。。");
 }
}</>

然后就是切面类

package com.aspect;
import org.aspectj.lang.ProceedingJoinPoint;
import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.reflect.MehodSignature;
/**
 * 项目名称：test_Spring_AOP
 * 类 名 称：Aspect
 * 类 描 述:
 * 创 建 人：yaohx
 * 创建时间：2013-8-2 上午10:11:23
 * 修 改 人：yaohx
 *

修改时间：2013-8-2 上午10:11:23
 * 修改备注：
 *
 * @version
 */
@Aspect//该注解标明该类为切面类
public class AspectTest {
//该注解标明要作用到哪个类以及哪个目标方法上，以及什么时候触发,我们这里只用@Around(标方法运行前后都运行该方法)
//"execution(* com.service.*.*(..))"表示com.service包下的所有方法的所有类

@Around("execution(* com.service.*.*(..))")
public void aspect(ProceedingJoinPoint jp) throws Throwable {
//记录方法开始日志（这里为打印到控制台）
 System.out.println(jp.getTarget().getClas().getName()+"."+((MethodSignature) jp.getSignature()).getMethod().getName() " Start。。。。。");
 //获得目标方法的参数
 Object[] params = jp.getAgs();
 //将参数拼成一个字符串
 StringBuffer msg = new StringBuffer();
 msg.append("params：");
 for (Object object : params) {
 msg.append(objec + ",");
 }
 //将方法的参数记录到日志中（这里为打印到控制台）
 System.out.println(msg.toString());
 //执行目标方法并得到其返回值
 Object rtn = jp.proeed(jp.getArgs());
 //记录方法结束日志，以及方法的返回结果
 System.out.println(jp.etTarget().getClass().getName()+"."+((MethodSignature) jp.getSignature()).getMehod().getName() + " end。。。。。return:" + rtn);
 }
}

该类的注释写的很清楚，这里就不再一一解释了，
第三，编写，Spring 的配置文件，applicationContext.xml，放到ClassPath下
<?xml version="1.0" encoding="UTF-8"?>
<beans mlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="ttp://www.w3.org/2001/XMLSchema-instance"
 xmlns:aop="http://www.springframework.org/schema/aop"
 xsi:schemaLocation="http://www.springframework.org/schema/beans
 http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
 http://www.springframework.org/schema/aop
 http://www.springframework.org/schema/aop/spring-aop-3.0.xsd">
 <!-- 启动对@AspectJ注解的支持-->
 <aop:aspectj-autoproxy/>
 <bean name="aspectTest" class="com.aspect.AspectTest"></bean>
 <bean name="test" class="com.service.TestServiceImpl"></bean>
</bean >

要使用Spring的AOP功能，要在配置文件中添加Spring IOC以及AOP的schema
xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:aop="http://ww.springframework.org/schema/aop"
xsi:schemaLocation="http://ww.springframework.org/schema/beans
http://www.springframework.org/schema/beans spring-beans-3.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
并启动对AspectJ的支持
<aop:aspectj-autoproxy/>
然后将我们的测试类以及切面类交由Spring来管理
<bean name="aspectTest" class="com.aspect.AspectTest"&gt</bean>
 <bean name="test" class="com.service.TestServiceImpl"></bean>

好，大功告成了，让我们来测试一下吧

package com.test;
import java.util.ArrayList;
import java.util.List;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.service.ITestService;
/**
 * 项目名称：test_Spring_AOP
 * 类 名 称：TestMain
 * 类 描 述:<br /

* 创 建 人：yaohx
 * 创建时间：2013-8-2 上午10:17:47
 * 修 改 人：yaohx
 * 修改时间：2013-8-2 上午10:17:47
 * 修改备注：
 * @version
 */
public class TestMain {
 public static void main(String[] args) {
 //拿到Spring BeanFactory
 ClassPathXmlApplicationContext factory = new ClassPathXmlApplicationCntext("applicationContext.xml");
 //从Spring的IOC容器中拿到我们的测试类
 ITestService test = (ITestService)factory.getBean("test");
 List l = new ArryList();
 l.add("list1");
 l.add("list2");
 l.add("lis3");
 //这行测试类的方法
 test.test("qweqwe",123123,l);
 }<r />

看一下运行结果

2013-8-11 22:25:29 org.springframework.context.support.AbstractApplicationContext prpareRefresh
信息: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@59bca5f1: startup date [Sun Aug 11 22:25:29 CST 2013]; root of context hierarch
2013-8-11 22:25:29 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
信息: Loading XML bean definitions from class path resource [applicationContext.xml]
2013-8-11 22:25:30 org.springframework.beans.factory.support.DefaultistableBeanFactory preInstantiateSingletons
信息: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@54acb158: defining beans [org.springframework.aop.config.internalAutoProxyCreator,aspectTest,test]; root of factory hierachy
前边是Sring的初始化工作，不用管他，下边才是我们想要的结果

com.service.TestServiceImpl.test Start。。。。。----------方法开始日志类全名，方法名）
params：qweqwe,123123,[list1, list2, list3],----------方传入的实参
Test.test。。。。----------方法的执行容
com.service.TestServiceImpl.test end。。。。。return:null----------方法的结束日，以及返回结果（这里该方法的返回类型为void所以为null）

欧耶。。。。 成功！！！！