



- [Java Core »](#)
- [Hibernate »](#)
- [Spring »](#)
- [Struts »](#)
- [Maven »](#)
- [Unit Test »](#)

Spring AOP Example – Pointcut , Advisor

Written on March 26, 2010 at 12:55 am by [mkyong](#)

In last [Spring AOP advice examples](#), all the methods in a class will be intercept automatically. But for most cases, you may need a way to intercept only one or two methods, this is what 'Pointcut' is. It's allow you to intercept a method by it's method name. In addition, a 'Pointcut' must be associated with an 'Advisor'.

In [Spring](#) AOP, comes with three very [technical terms](#) - **Advices, Pointcut , Advisor**, put it in unofficial way...

- [Advice](#) – Indicate the action to take either before or after [the method](#) execution.
- [Pointcut](#) – Indicate which method should be intercept, by method name or [regular expression](#) name.
- [Advisor](#) – Group 'Advice' and 'Pointcut' into a single unit, and pass it to a proxy factory object.

Example

Take the [last AOP advice example](#) again, a simple customer service class

```
package com.mkyong.customer.services;  
  
public class CustomerService  
{  
    private String name;  
    private String url;  
  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

```

    public void setUrl(String url) {
        this.url = url;
    }

    public void printName(){
        System.out.println("Customer name : " + this.name);
    }

    public void printURL(){
        System.out.println("Customer website : " + this.url);
    }

    public void printThrowException(){
        throw new IllegalArgumentException();
    }
}

```

Bean configuration file (Spring-Customer.xml) with ‘Around advice’ associate with customerService.

```

<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

    <bean id="customerService" class="com.mkyong.customer.services.CustomerService" >
        <property name="name" value="Yong Mook Kim" />
        <property name="url" value="http://www.mkyong.com" />
    </bean>

    <bean id="hijackAroundMethodBeanAdvice"
        class="com.mkyong.aop.HijackAroundMethod" />

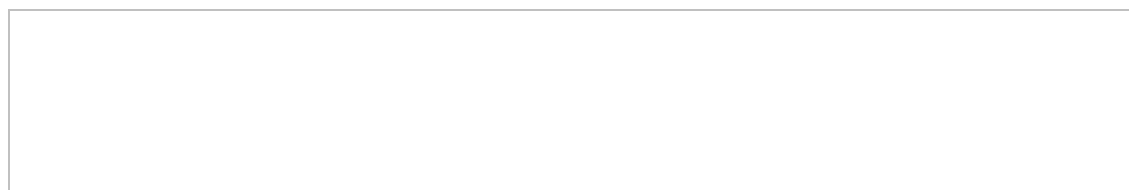
    <bean id="customerServiceProxy"
        class="org.springframework.aop.framework.ProxyFactoryBean">

        <property name="target" ref="customerService" />

        <property name="interceptorNames">
            <list>
                <value>hijackAroundMethodBeanAdvice</value>
            </list>
        </property>
    </bean>
</beans>

```

Run it



```

package com.mkyong.common;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.mkyong.customer.services.CustomerService;

public class App
{
    public static void main( String[] args )
    {
        ApplicationContext appContext =
            new ClassPathXmlApplicationContext(new String[] { "Spring-Customer.xml" });

        CustomerService cust =
            (CustomerService)appContext.getBean( "customerServiceProxy" );

        System.out.println( "*****" );
        cust.printName();
    }
}

```

```

        System.out.println("*****");
        cust.printURL();
        System.out.println("*****");
        try{
            cust.printThrowException();
        }catch(Exception e){
        }
    }
}

```

output

```

*****
Method name : printName
Method arguments : []
HijackAroundMethod : Before method hijacked!
Customer name : Yong Mook Kim
HijackAroundMethod : Before after hijacked!
*****
Method name : printURL
Method arguments : []
HijackAroundMethod : Before method hijacked!
Customer website : http://www.mkyong.com
HijackAroundMethod : Before after hijacked!
*****
Method name : printThrowException
Method arguments : []
HijackAroundMethod : Before method hijacked!
HijackAroundMethod : Throw exception hijacked!

```

It will intercept all the customer service's methods. Can we intercept just a printName() method?

Pointcuts – Name match example

You can intercept a printName() method via 'pointcut' and 'advisor'. Create a **NameMatchMethodPointcut** pointcut bean, and put the method name you want to intercept in the 'mappedName' [property](#) value.

```

<bean id="customerPointcut"
      class="org.springframework.aop.support.NameMatchMethodPointcut">
    <property name="mappedName" value="printName" />
</bean>

```

Create a **DefaultPointcutAdvisor** advisor bean, and associate both advice and pointcut.

```

<bean id="customerAdvisor"
      class="org.springframework.aop.support.DefaultPointcutAdvisor">
    <property name="pointcut" ref="customerPointcut" />
    <property name="advice" ref="hijackAroundMethodBeanAdvice" />
</bean>

```

Replace the proxy's 'interceptorNames' to 'customerAdvisor' (it was 'hijackAroundMethodBeanAdvice').

```

<bean id="customerServiceProxy"
      class="org.springframework.aop.framework.ProxyFactoryBean">

    <property name="target" ref="customerService" />

    <property name="interceptorNames">
        <list>
            <value>customerAdvisor</value>
        </list>
    </property>
</bean>

```

Full bean configuration file

```

<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

    <bean id="customerService" class="com.mkyong.customer.services.CustomerService" >

```

```

<property name="name" value="Yong Mook Kim" />
<property name="url" value="http://www.mkyong.com" />
</bean>

<bean id="hijackAroundMethodBeanAdvice"
      class="com.mkyong.aop.HijackAroundMethod" />

<bean id="customerServiceProxy"
      class="org.springframework.aop.framework.ProxyFactoryBean">

    <property name="target" ref="customerService" />

    <property name="interceptorNames">
        <list>
            <value>customerAdvisor</value>
        </list>
    </property>
</bean>

<bean id="customerPointcut"
      class="org.springframework.aop.support.NameMatchMethodPointcut">
    <property name="mappedName" value="printName" />
</bean>

<bean id="customerAdvisor"
      class="org.springframework.aop.support.DefaultPointcutAdvisor">
    <property name="pointcut" ref="customerPointcut" />
    <property name="advice" ref="hijackAroundMethodBeanAdvice" />
</bean>

</beans>

```

Run it again, output

```

*****
Method name : printName
Method arguments : []
HijackAroundMethod : Before method hijacked!
Customer name : Yong Mook Kim
HijackAroundMethod : Before after hijacked!
*****
Customer website : http://www.mkyong.com
*****

```

Now, you only intercept the printName() method.

PointcutAdvisor class

Spring comes with **PointcutAdvisor** class to save you declared advisor and pointcut into different beans, you can use **NameMatchMethodPointcutAdvisor** to combine both into a single bean.

```

<bean id="customerAdvisor"
      class="org.springframework.aop.support.NameMatchMethodPointcutAdvisor">

    <property name="mappedName" value="printName" />
    <property name="advice" ref="hijackAroundMethodBeanAdvice" />

</bean>

```

Pointcut – Regular expression example

Beside the matching method by name, you can also match the method's name by using regular expression pointcut – **RegexpMethodPointcutAdvisor**.

```

<bean id="customerAdvisor"
      class="org.springframework.aop.support.RegexpMethodPointcutAdvisor">
    <property name="patterns">
        <list>
            <value>.*URL.*</value>
        </list>
    </property>
</bean>

```

```
<property name="advice" ref="hijackAroundMethodBeanAdvice" />
</bean>
```

It will only match the method which has ‘URL ‘ within the method name. It’s also quite useful for the DAO transaction management, where you can declare “.*DAO.*” to include all your DAO classes.

You can download this Spring Pointcut & Advisor example here – [Spring-AOP-Pointcuts-Advisor-Example.zip](#)

Be the first of your friends to like this.

Popular Tutorials



- [Struts Tutorials](#)



- [Spring Tutorials](#)



- [Maven Tutorials](#)



- [Hibernate Tutorials](#)

1 Comment

1. [Google PPC: Content or Search? / Adsense](#) says:
[March 26, 2010 at 4:18 pm](#)

[...] Spring AOP Example – Pointcut , Advisor | Spring [...]

[Reply](#)

Leave a Reply

Name (required)

Mail (will not be published) (required)

Website☐ Notify me of followup comments via e-mail

Popular Tutorials

hibernate

- [Hibernate Tutorials](#)

Hibernate is a object/relational persistence tool for Java developers. Hibernate lets you to use object-oriented way to develop your persistent classes, It's also provide many data manipulation API like ...

navel

-

[Maven Tutorials](#)

Apache Maven is a software [project management](#) tool, not just a build tool like ant. It's provides a new concept of a project object model (POM) file, to manage project's ...

[Apache Archiva Tutorials](#)

Apache Archiva is a Build Artifact Repository Manager, a repository management that helps to create a Maven repository in our end.

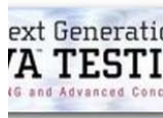
To simple, it helps to create a Maven repository ...

[Spring Tutorials](#)

The Spring framework is a powerful and extensible Inversion of control(IoC) container, provides developers a good habit to program to interface, rather than classes to decouple your components, helps developers ...

[JUnit Tutorials](#)

JUnit is an [open source](#) testing framework, and an instance of the xUnit architecture for unit testing frameworks. JUnit is simple and suitable used for pure unit testing, for ...

[TestNG Tutorials](#)

TestNG (Next Generation) is a testing framework inspired from JUnit and NUnit, but introducing some new functionalities like dependency testing, grouping concept to make testing more powerful and easier to ...

[Java XML Tutorials](#)

Java comes with two XML parsers – DOM and SAX to process XML file. The others popular third party [XML parser](#) is JDOM. Here's few examples to show how to ...

[Java Regular Expression Tutorials](#)







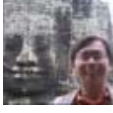

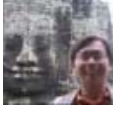

Java has comprehensive support for Regular Expression functionality through the java.util.regex package. The regular expression language is easy to learn but hard to master, the better way to learn it ...



- [Struts Tutorials](#)

The Struts 1.x framework is a the most famous, classic and proven success MVC framework. Often times, you will listen something like, meaningless to learn Struts 1.x, it's a dead ...

Recent Comments

-  Stephane : For a MySql schema with 100 tables, each having some NOT NULL DEFAULT... [More](#)
-  Neeraj : No need to set window.onload function at the end of page... [More](#)
-  anonymous : Just go to the following site: warez-bb.org and register. Than... [More](#)
-  sahil : thanks , but There is no getServletContext(); in WebApplication.get() [More](#)
-  Courtney : Hi my name is courtney and Um srry this must be a really really... [More](#)
-  jagadish : I gone through the notes and it is very good for begeers. ~Thanks,jaga [More](#)
-  mkyong : Sorry, are you tested in IE6? caused it worked fine in IE 7 & 8 [More](#)
-  dattai : that's very cool Thanks alot [More](#)
-  mkyong : use jdom to read the xml file and constructs any string format you... [More](#)
-  mkyong : Java cache? What to do with XML? [More](#)

Authors

Hi, my name is Yong Mook Kim, person behind Mkyong.com.

This website is providing daily Java / J2EE web development tutorials, which involve Spring, Hibernate, Struts, Maven, Java Core, Unit Test...

Friends & Links

- [China Wholesale](#)
- [Dropship](#)
- [Investment Life](#)

- [Webmaster Forum](#)
- [MySQL Tutorials](#)
- [PHP Tutorials](#)

- [ChanKelwin](#)
- [Internet Blogger](#)
- [TanWanMei](#)
- [TenthOfMarch](#)
- [Find Free Icons](#)
- [Excel Consultant](#)
- [IT Support Liverpool](#)

Copyright © 2008-2010 [Mkyong.com](#) | [RSS Feed](#) | [Privacy Policy](#) | [Write for Cash](#) | [Advertise with Us](#) | [Contact Us](#)