

Spring Aop pointcut expression

Теги: [Spring](#) [AOP](#) [Pointcut](#)

В Spring AOP обычно необходимо использовать язык выражений pointcut AspectJ для определения pointcut. Важно то, что в Spring поддерживается только подмножество индикаторов pointcut AspectJ.

AspectJ Pointcut индикатор поддерживается Spring

Индикатор AspectJ	описание
<code>args()</code>	Метод выполнения, который ограничивает параметры сопоставления точки подключения типом выполнения
<code>@args()</code>	Метод выполнения, который ограничивает пометки параметров соответствия точки подключения аннотациями выполнения
<code>execution()</code>	Соответствующий метод выполнения точки подключения
<code>this()</code>	Ограничить ссылочный тип Бина точки соединения, соответствующей прокси-серверу AOP, указанным типом Бина
<code>target()</code>	Ограничить точки подключения от сопоставления целевых объектов до классов указанных типов
<code>@target()</code>	Ограничить точки подключения от сопоставления целевого объекта с указанным классом аннотаций
<code>within()</code>	Ограничить точки подключения, чтобы соответствовать указанному типу
<code>@within()</code>	Ограничить точки подключения, чтобы соответствовать указанному типу аннотации
<code>@annotation</code>	Ограничить сопоставление точек подключения с указанными аннотациями

Обычно в Spring AOP используются:

1. `execution(modifiers-pattern? ret-type-pattern declaring-type-pattern? name-pattern(param-pattern)`
2. `throws-pattern?)`

Сопоставить все

```
execution("* *.*(..)")
```

Сопоставить все методы, начинающиеся с set

```
execution("* *.set*(..)")
```

Сопоставить все методы в указанном пакете

```
execution("* com.david.biz.service.impl.*(..)")
```

Сопоставить все методы в указанном пакете и его подпакетах

```
execution("* com.david.*(..)")
```

Соответствует указанному пакету, а его тип параметра подпакета - String

```
execution("* com.david.*(java.lang.String))
```

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:p="http://www.springframework.org/schema/p" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:context="http://www.springframework.org/schema/context" xmlns:tx="http://www.springframework.org/schema/tx"
5.     xmlns:aop="http://www.springframework.org/schema/aop" xmlns:jee="http://www.springframework.org/schema/jee"
6.     xmlns:task="http://www.springframework.org/schema/task"
7.     xsi:schemaLocation="
8.         http://www.springframework.org/schema/beans
9.         http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
10.        http://www.springframework.org/schema/context
11.        http://www.springframework.org/schema/context/spring-context-3.0.xsd
12.        http://www.springframework.org/schema/aop
13.        http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
14.        http://www.springframework.org/schema/tx
15.        http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
16.        http://www.springframework.org/schema/jee
17.        http://www.springframework.org/schema/jee/spring-jee-3.0.xsd
18.        http://www.springframework.org/schema/task
19.        http://www.springframework.org/schema/task/spring-task-3.1.xsd
20.    ">
21.    <context:component-scan base-package="com.david.*"/>
22.    <aop:aspectj-autoproxy />
23.    <context:property-placeholder location="classpath:META-INF/config.properties" />
24.    <!-- Определить источник данных -->
25.    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
26.        destroy-method="close">
27.        <property name="driverClass" value="${jdbc.driver}" />

```

```

28.         <property name="jdbcUrl" value="${jdbc.ams.url}" />
29.         <property name="user" value="${jdbc.ams.username}" />
30.         <property name="password" value="${jdbc.ams.password}" />
31.         <property name="initialPoolSize" value="${initialSize}" />
32.         <property name="minPoolSize" value="${minPoolSize}" />
33.         <property name="maxPoolSize" value="${maxActive}" />
34.         <property name="acquireIncrement" value="${acquireIncrement}" />
35.         <property name="maxIdleTime" value="${maxIdleTime}" />
36.     </bean>
37.
38.     <! - Определить класс шаблона jdbc ->
39.     <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
40.         <property name="dataSource" ref="dataSource" />
41.     </bean>
42.
43.     <bean id="sqlMapClient" class="org.springframework.orm.ibatis.SqlMapClientFactoryBean">
44.         <property name="dataSource" ref="dataSource" />
45.         <property name="configLocation" value="classpath:META-INF/sqlmap/sqlmap.xml" />
46.     </bean>
47.
48.     <tx:advice id="txAdvice" transaction-manager="transactionManager">
49.         <tx:attributes>
50.             <tx:method name="addBook" propagation="REQUIRED" />
51.             <tx:method name="addUserBook" propagation="MANDATORY" />
52.             <tx:method name="deleteBook" propagation="REQUIRES_NEW" />
53.             <tx:method name="addNewBook" propagation="NEVER" />
54.             <tx:method name="addUser" propagation="NESTED" />
55.         </tx:attributes>
56.     </tx:advice>
57.     <bean id="transactionManager"
58.         class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
59.         <property name="dataSource" ref="dataSource" />
60.     </bean>
61.
62. </beans>

```

```

1. @Service("bookService")
2. public class BookServiceImpl implements BookService {
3.
4.     private static final Logger logger = LogManager.getLogger(BookServiceImpl.class);
5.     public static final String ADD_BOOK = "insert into t_book(id,name) values(1,'duck-j2ee')";
6.
7.     public static final String DELETE_BOOK = "delete from t_book where id=1";
8.
9.     private JdbcTemplate jdbcTemplate;
10.    @Autowired
11.    private BookDao bookDao;
12.
13.    public void addBook() throws Exception {
14.        Book book = new Book();
15.        book.setName("ibatis");
16.        book.setPrice(11);
17.        bookDao.insert(book);
18.        генерировать новое исключение UnRollbackException («Проверено исключение, откат не будет»);
19.    }
20.
21.    public void deleteBook(int id) {
22.        try {
23.            bookDao.deleteById(id);
24.        } catch (SQLException e) {
25.            logger.error("", e);
26.        }
27.    }
28.
29.    @LoggingRequired
30.    public void addNewBook(String name, int price) {
31.        try {
32.            Book book = new Book();
33.            book.setName(name);
34.            book.setPrice(price);
35.            bookDao.insert(book);
36.            List<Book> lists = bookDao.selectAll();
37.            System.out.println(lists);
38.        } catch (SQLException e) {
39.            logger.error("", e);
40.        }
41.    }
42.
43.    public void addUserBook() {
44.        jdbcTemplate.execute("insert into t_book(id,name) values(3,'UserBook')");
45.    }
46.
47.    /**
48.     * Setter method for property <tt>jdbcTemplate</tt>.
49.     *
50.     * @param jdbcTemplate value to be assigned to property jdbcTemplate
51.     */
52.    public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
53.        this.jdbcTemplate = jdbcTemplate;
54.    }
55.
56.    /**
57.     * @see com.david.biz.service.BookService#queryAll()
58.     */

```

```

59.     public List<Book> queryAll() {
60.         try {
61.             return bookDao.selectAll();
62.         } catch (SQLException e) {
63.             logger.error("", e);
64.         }
65.         return null;
66.     }
67.
68. }

```

```

1. /**
2.  * execution (modifiers-pattern? ret-type-pattern declaring-type-pattern? name-pattern (param-pattern)
3.      throws-pattern?)
4.  * arg () Метод выполнения, который ограничивает параметр сопоставления точки подключения указанным типом
5.  * @ args () Ограничить выполнение параметров сопоставления точек подключения, отмеченных аннотациями выполнения
6.  * execute () используется для соответствия методу выполнения точки подключения
7.  * this () ограничивает точку подключения, соответствующую ссылке Bean-адреса прокси-сервера AOP, в качестве класса типа
8.  * target () Ограничить точку подключения, чтобы сопоставить целевой объект с указанным типом класса
9.  * @ target () Ограничить точки подключения, чтобы соответствовать конкретным объектам выполнения, эти объекты должны ии
10. * @ annotation () ограничивает сопоставление точек подключения с указанными аннотациями
11. *
12. *
13. *
14. * @author zhangwei_david
15. * @version $ Id: LogAspect.java, v 0.1 29 ноября 2014 г. 13:10:13 zhangwei_david Exp $
16. */
17. @Component
18. @Aspect
19. public class LogAspect {
20.     private static final Logger logger = LogManager.getLogger(LogAspect.class);
21.
22.     /**
23.      * Соответствующим параметром является любой тип, любое число и метод в пакете com, david.biz или подпакете.
24.      */
25.     @Pointcut("args(..)&&within(com.david.biz..*)")
26.     public void arg() {
27.
28.     }
29.
30.     @Pointcut("@args(com.david.aop.LoggingRequired)")
31.     public void annotationArgs() {
32.
33.     }
34.
35.     @Pointcut("@annotation(com.david.aop.LoggingRequired)")
36.     public void logRequiredPointcut() {
37.
38.     }
39.
40.     @Pointcut("args(java.lang.String,*)")
41.     public void argsWithString() {
42.
43.     }
44.
45.     @Pointcut("target(com.david.biz.service.impl.BookServiceImpl)")
46.     public void targetPointcut() {
47.
48.     }
49.
50.     @Pointcut("@target(org.springframework.stereotype.Service)")
51.     public void targetAnnotation() {
52.
53.     }
54.
55.     // @Around("execution(* org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter.handle(..)")
56.     // public Object aa(ProceedingJoinPoint pjp) throws Throwable {
57.     //     try {
58.     //         Object retVal = pjp.proceed();
59.     //         System.out.println(retVal);
60.     //         return retVal;
61.     //     } catch (Exception e) {
62.     //         // System.out.println ("Исключение");
63.     //         return null;
64.     //     }
65.     // }
66.     @Before(value = "logRequiredPointcut()")
67.     public void before(JoinPoint joinPoint) {
68.         LogUtils.info(logger,
69.             «Выражение точки подключения @annotation (com.david.aop.LoggingRequired) -method = {0} было посеи
70.             joinPoint.getSignature().getName());
71.     }
72.
73.     @Before(value = "arg()")
74.     public void beforeArg(JoinPoint joinPoint) {
75.         LogUtils.info(logger,
76.             «Выражение точки соединения: args (..) && в (com.david.biz .. *) метод = {0}, args = {1}, target
77.             joinPoint.getSignature().getName(), ToStringBuilder.reflectionToString(
78.             joinPoint.getArgs(), ToStringStyle.SHORT_PREFIX_STYLE), joinPoint.getTarget()
79.             .getClass().getName());
80.     }
81.
82.     @Before(value = "argsWithString()")

```

```

83.     public void beforeArgWithString(JoinPoint joinPoint) {
84.         LogUtils.info (logger, «Выражение точки соединения: метод args (java.lang.String, *) = {0}, args = {1},
85.             joinPoint.getSignature().getName(), ToStringBuilder.reflectionToString(
86.                 joinPoint.getArgs(), ToStringStyle.SHORT_PREFIX_STYLE), joinPoint.getTarget()
87.                 .getClass().getName());
88.     }
89.
90.     @Before(value = "annotationArgs()")
91.     public void beforeAnnotationArgs(JoinPoint joinPoint) {
92.         LogUtils
93.             .info(
94.                 logger,
95.                 «Выражение точки соединения: @args (com.david.annotation.validate.Length, *) method = {0},
96.                 joinPoint.getSignature().getName(), ToStringBuilder.reflectionToString(
97.                     joinPoint.getArgs(), ToStringStyle.SHORT_PREFIX_STYLE), joinPoint.getTarget()
98.                     .getClass().getName());
99.     }
100.
101.     @Before(value = "targetPointcut()")
102.     public void beforeTarget(JoinPoint joinPoint) {
103.         LogUtils
104.             .info(
105.                 logger,
106.                 «Выражение точки подключения: метод target (com.david.biz.service.impl.BookServiceImpl) = {0}, a
107.                 joinPoint.getSignature().getName(), ToStringBuilder.reflectionToString(
108.                     joinPoint.getArgs(), ToStringStyle.SHORT_PREFIX_STYLE), joinPoint.getTarget()
109.                     .getClass().getName());
110.     }
111.
112.     @Before(value = "targetAnnotation()")
113.     public void beforeTargetAnnotation(JoinPoint joinPoint) {
114.         LogUtils
115.             .info(
116.                 logger,
117.                 «Выражение точки подключения: метод @target (org.springframework.stereotype.Service) = {0},
118.                 joinPoint.getSignature().getName(), ToStringBuilder.reflectionToString(
119.                     joinPoint.getArgs(), ToStringStyle.SHORT_PREFIX_STYLE), joinPoint.getTarget()
120.                     .getClass().getName());
121.     }
122. }

```

```

1. /**
2.  *
3.  * @author zhangwei_david
4.  * @version $ Id: T.java, v 0.1 1 декабря 2014 г. 9:35:44 zhangwei_david Exp $
5.  */
6. @RunWith(SpringJUnit4ClassRunner.class)
7. @ContextConfiguration(locations = "file:H:/workspace4study/WebApp/src/main/webapp/WEB-INF/applicationContext.xml")
8. public class BookServiceTest {
9.
10.     @Autowired
11.     private BookService bookService;
12.
13.     @Test
14.     public void testB() {
15.         bookService.addNewBook("JUnit Test", 1000);
16.     }
17.
18. }

```

```

1. 2014-12-01 11:14:39 [main: 1577] - [INFO] Выражение точки подключения @annotation (com.david.aop.LoggingRequired) -method
2. 2014-12-01 11:14:39 [main: 1587] - [INFO] Выражение точки подключения: args (..) && в (com.david.biz .. *) метод = addNew
3. 2014-12-01 11:14:39 [main: 1588] - [INFO] Выражение точки подключения: args (java.lang.String, *) method = addNewBook, a
4. 2014-12-01 11:14:39 [main: 1588] - [INFO] Выражение точки подключения: target (com.david.biz.service.impl.BookServiceImp
5. 2014-12-01 11:14:39 [main: 1589] - [INFO] Выражение точки подключения: @target (org.springframework.stereotype.Service)
6. 2014-12-01 11:14:39 [main: 1589] - [INFO] Выражение точки подключения: args (..) && в (com.david.biz .. *) метод = insert
7. 2014-12-01 11:14:39 [main: 1590] - [INFO] Выражение точки подключения: @args (com.david.annotation.validate.Length, *) m
8. 2014-12-01 11:14:39 [main: 1591] - [INFO] Выражение точки подключения: args (java.lang.String, *) method = insert, args =

```