L’AOP in Spring con AspectJ – Parte I

Cosa imparerai

- Come implementare l’AOP in Spring con AspectJ





Aspect Oriented Programming

Ricordiamo cos'è L'AOP…

L'Aspect Oriented Programming (AOP) è un paradigma di programmazione complementare all'OOP.

L'AOP è in grado di descrivere dei comportamenti che sono trasversali all'applicazione.

Con l'AOP, questi comportamenti vengono separati ed isolati dal dominio applicativo in moduli dedicati al singolo aspetto.

L'AOP garantisce:

- ❑ Separazione dei comportamenti trasversali dal dominio applicativo
- ❑ Riutilizzo del codice

Aspect Oriented Programming in Spring

AOP in Spring

Il primo modulo creato in Spring per l'AOP è **Spring AOP**, nato con lo scopo di fornire un'integrazione con il container IoC.

Dalla versione 2.x, Spring AOP è stato esteso con il supporto ad **AspectJ**.

AspectJ è un'estensione al linguaggio Java che implementa l'AOP.

Informazioni e dettagli su AspectJ sono disponibili su

https://www.eclipse.org/aspectj/

Aspect Oriented Programming in Spring con AspectJ

Per poter utilizzare *AspectJ* in un'applicazione Spring, è necessario aggiungere due jar:

- **aspectjrt.jar**
- **aspectjweaver.jar**

Se usiamo Maven, è necessario aggiungere la dipendenza dai due artifact (versione 1.8 o superiore).

```
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>…</version>
</dependency>

<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjweaver</artifactId>
    <version>1.9.6</version>
</dependency>
```

Aspect Oriented Programming in Spring con AspectJ

Vediamo come creare Aspect e gestire tutti gli elementi dell'AOP con *AspectJ*…

Per utilizzare le annotation di AspectJ è necessario abilitare il supporto Spring AOP basato su AspectJ.

Possiamo abilitare il support usando le configurazioni Java (tramite annotation) o XML.

Configurazione Java

Tramite l'annotation **org.springframework.context.annotation.EnableAspectJAutoProxy**

```
@Configuration
@ImportResource("services.xml")
@EnableAspectJAutoProxy
public class Config {
    @Value("${cliente.tipo-predefinito}")
    private String tipoPredefinito;
```

Aspect Oriented Programming in Spring con AspectJ

Vediamo come creare Aspect e gestire tutti gli elementi dell'AOP con *AspectJ*…

Configurazione XML

Tramite l'elemento **<aop:aspectj-autoproxy/>**

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="https://www.springframework.org/schema/beans"
                xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
                xmlns:aop="https://www.springframework.org/schema/aop"
                xsi:schemaLocation="https://www.springframework.org/schema/beans

                                https://www.springframework.org/schema/beans/spring-beans-4.0.xsd
                                https://www.springframework.org/schema/aop
                                https://www.springframework.org/schema/aop/spring-aop-4.0.xsd">
    <aop:aspectj-autoproxy />
    …
</beans>
```

Di cosa abbiamo parlato in questa lezione

- Come implementare l'AOP in Spring con AspectJ