Autowire in Spring

Cosa imparerai

- Cos’è l’autowire e come utilizzarlo in Spring

paolopreite

Autowire in Spring

Per gestire le dipendenze tra bean, è possibile utilizzare i metadati di configurazione.

In particolare, possiamo utilizzare:

- L'elemento **<property ref="..." />** dove ref conterrà il nome del bean da inizializzare prima del nostro
- L'attributo **depends-on** inserito nell'elemento **<bean />** per indicare al container i bean da cui dipende il bean che stiamo definendo

Il container di Spring può anche fare di più: attraverso i metadati di configurazione è possibile anche chiedere al container di definire in automatico le dipendenze tra i bean.

Questo meccanismo si chiama autowire.

Autowire in Spring

Se utilizziamo l'autowire, non abbiamo bisogno di configurare le dipendenze nei metadati di configurazione, ma lasciamo al container l'onere di risolverle per noi.

L'Autowiring in Spring utilizza la Setter Injection o la Constructor Injection.

IMPORTANTE: **l'autowiring funziona solo per l'injection di bean, non può essere utilizzato per fare l'injection di tipi primitivi e stringhe**.

```
<bean id="ordine3" class="it.test.service.OrdineServiceValorizzareVariabili">
    <property name="prodotto" ref="prodottoService"></property>
</bean>
```

Autowire in Spring

Facciamo un esempio…

Il bean *RubricaService* invoca il bean *MessaggiService* per richiede la lista dei messaggi inviati da un utente.

Se non utilizziamo l'autowire, dobbiamo utilizzare l'elemento *<property ref="…" />* del bean per definire la dipendenza di *RubricaService* da *MessaggiService*.

```java
public class MessaggiService {

    public MessaggiService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return "Messaggio 1 - Messaggio 2 - ...";
    }
}
```

```java
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }
}
```

```xml
<bean id="rubricaService" class="it.test.service.RubricaService">
    <property name="messaggiService" ref="messaggiService"></property>
</bean>
```

Autowire in Spring

Se utilizziamo l'autowire, invece, non dobbiamo utilizzare l'elemento *<property ref="..." />* del bean per definire la dipendenza di *RubricaService* da *MessaggiService*.

In questo caso è necessario configurare l'attributo **autowire** (se si utilizzano i metadati XML-based) sul bean *RubricaService*, chiedendo al container di definire in automatico la dipendenza dal bean *MessaggiService*.

```java
public class MessaggiService {

    public MessaggiService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return "Messaggio 1 - Messaggio 2 - ...";
    }
}
```

```java
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }
}
```

```xml
<bean id="messaggiService" class="it.test.service.MessaggiService">
</bean>

<bean id="rubricaService" class="it.test.service.RubricaService" autowire="...">
</bean>
```

Autowire in Spring

Quali sono i valori ammessi per l'attributo autowire?

- **no**: è il valore predefinito e indica che non è richiesto l'autowiring. Questo è il caso in cui siamo noi a definire le dipendenze tra i bean mediante l'elemento *<property ref="…" />*.

- **byName**: se specifichiamo questo valore, il container cerca un bean che ha lo stesso nome della proprietà che deve essere «*autowired*».

```
<bean id="messaggiService" class="it.test.service.MessaggiService">
</bean>

<bean id="rubricaService" class="it.test.service.RubricaService" autowire="byName">
</bean>
```

```
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }

    public MessaggiService getMessaggiService() {
        return messaggiService;
    }

    public void setMessaggiService(MessaggiService messaggiService) {
        this.messaggiService = messaggiService;
    }
}
```

Affinché venga eseguito l'autowire **byName**, è necessario che il bean *RubricaService* abbia una variabile d'istanza con lo stesso nome del bean *MessaggiService* e che sia definito il metodo ***set(…)*** di questa variabile.

Autowire in Spring

Quali sono i valori ammessi per l'attributo autowire?

- **byType**: se specifichiamo questo valore, il container cerca un bean che ha lo stesso tipo della proprietà che deve essere «*autowired*». Se il container individua più bean dello stesso tipo, viene generata un'eccezione, che indica che non è possibile utilizzare l'autowire byType per quel bean.

```
<bean id="messaggiService" class="it.test.service.MessaggiService">
</bean>

<bean id="rubricaService" class="it.test.service.RubricaService" autowire="byType">
</bean>
```

```
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService() {
        super();
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }

    public MessaggiService getMessaggiService() {
        return messaggiService;
    }

    public void setMessaggiService(MessaggiService messaggiService) {
        this.messaggiService = messaggiService;
    }
}
```

Affinché venga eseguito l'autowire **byType**, è necessario che esista un bean di tipo *MessaggiService* e che nel bean *RubricaService* sia definito il metodo ***set(…)*** della variabile d'istanza *messaggiService*.

Autowire in Spring

Quali sono i valori ammessi per l'attributo autowire?

- **constructor**: analogo a byType ma si applica agli argomenti del costruttore e non alle variabili di istanza. Se non c'è esattamente un bean del tipo di argomento del costruttore nel contenitore, viene generato un errore irreversibile.

```
<bean id="messaggiService" class="it.test.service.MessaggiService">
</bean>

<bean id="rubricaService" class="it.test.service.RubricaService" autowire="constructor">
</bean>
```

```
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService(MessaggiService messaggiService) {
        super();
        this.messaggiService = messaggiService;
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }

    public MessaggiService getMessaggiService() {
        return messaggiService;
    }
}
```

Affinché venga eseguito l'autowire **constructor**, è necessario che esista un bean di tipo *MessaggiService* e che nel bean *RubricaService* sia definito un costruttore che abbia tra gli argomenti una variabile di tipo *MessaggiService*.

Autowire in Spring

Vantaggi dell'autowiring

- Richiede meno codice e meno configurazione da parte dello sviluppatore;
- Può ridurre notevolmente la necessità di inserire variabili di istanza o argomenti nel costruttore;
- Riduce il tempo di sviluppo e di evoluzione di un'applicazione poiché, se vengono definite nuove dipendenze, non è necessario specificarle in quanto il container le risolverà in automatico.

Autowire in Spring

Svantaggi e limiti dell'autowiring

- Nessun controllo da parte del programmatore. Le dipendenze vengono gestite in automatico dal container
- Le dipendenze esplicite vanno in override sull'autowiring.
- L'autowire non si può fare su variabili che sono tipi di dato primitivi (int, byte, …), String, e classi che non siano bean.

Autowire in Spring

È possibile escludere un bean dall'autowiring?
Si.

È sufficiente impostare l'attributo **autowire-candidate** a **false** sul bean da escludere.

```
<bean id="messaggiService" class="it.test.service.MessaggiService" autowire-candidate="false">
</bean>

<bean id="rubricaService" class="it.test.service.RubricaService" autowire="constructor">
</bean>
```

```
public class RubricaService {
    private MessaggiService messaggiService;

    public RubricaService(MessaggiService messaggiService) {
        super();
        this.messaggiService = messaggiService;
    }

    public String getMessaggi(Long idUser) {
        return messaggiService.getMessaggi(idUser);
    }
```

```
Exception in thread "main" org.springframework.beans.factory.UnsatisfiedDependencyException: Error creating bean with name 'rubricaService' defined in class
path resource [services-xml.xml]: Unsatisfied dependency expressed through constructor parameter 0; nested exception is
org.springframework.beans.factory.NoSuchBeanDefinitionException: No qualifying bean of type 'it.test.service.MessaggiService' available: expected at least 1
bean which qualifies as autowire candidate. Dependency annotations: {}
```

Affinché il container non generi eccezione, in questo caso devo esplicitare la dipendenza di *RubricaService* dal bean *MessaggiService*.

```
<bean id="rubricaService" class="it.test.service.RubricaService" autowire="constructor">
    <constructor-arg name="messaggiService" ref="messaggiService"></constructor-arg>
</bean>
```

Autowire in Spring

È meglio usare l'autowire o la specifica esplicita delle dipendenze?

La specifica esplicita delle dipendenze tra i bean offre maggiore controllo e chiarezza per cui è preferibile utilizzarla nel caso di progetti di grosse dimensioni.

In un certo senso, la definizione esplicita delle dipendenze è una sorta di documentazione tecnica che aiuta a capire le relazioni tra gli oggetti dell'applicazione.

Di cosa abbiamo parlato in questa lezione

- Cos’è l’autowire e come utilizzarlo in Spring