Il container IoC

Cosa imparerai

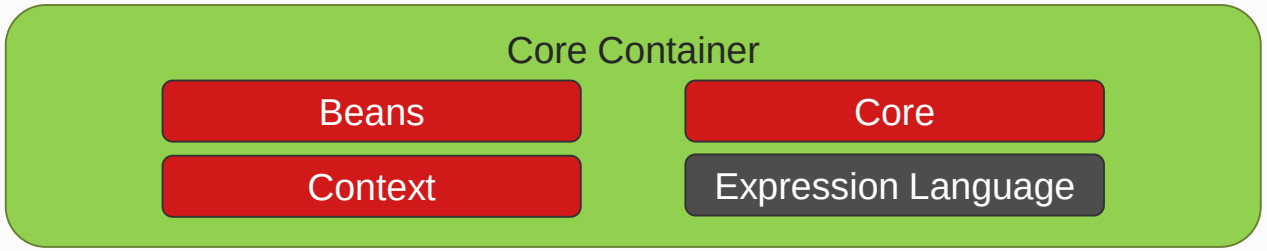
- Come funziona il container IoC in Spring





Il container IoC

Come abbiamo detto, i moduli Spring che consentono di implementare l'IoC sono *beans*, *core* e *context*.



In particolare, il modulo **spring-context**, basato sul modulo *spring-beans*, contiene l'interfaccia **ApplicationContext** che eredita le funzionalità della *BeanFactory* e ne aggiunge altre, tra cui:

- supporto per l'internazionalizzazione
- propagazione di eventi
- caricamento di risorse
- supporto a JEE

Il container IoC

L'interfaccia **org.springframework.context.ApplicationContext** rappresenta il container IoC in Spring ed è responsabile di:

- creazione di istanze
- configurazione ed assemblaggio dei bean

Il container riceve istruzioni su:

- quali oggetti istanziare
- come configurarli e assemblarli attraverso i metadati di configurazione

I metadati sono rappresentati in XML, Java annotation o codice Java e consentono di definire quali oggetti compongono la tua applicazione e le dipendenze tra essi.

Il container IoC

In Spring sono presenti diverse implementazioni dell'interfaccia ApplicationContext.

Nelle applicazioni standalone, è possibile creare un'istanza delle seguenti classi:

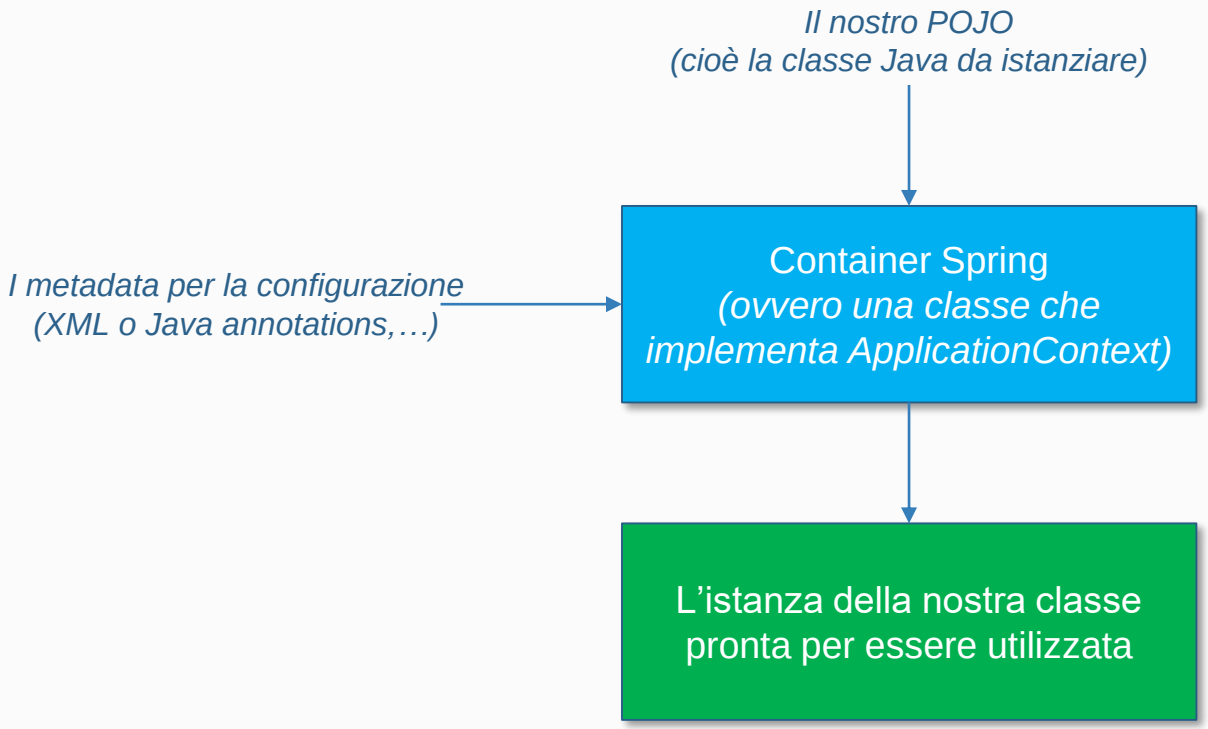
- ClassPathXmlApplicationContext
- FileSystemXmlApplicationContext
- XmlWebApplicationContext
- AnnotationConfigApplicationContext

Le prime tre hanno bisogno di file XML che dovranno contenere i metadati di configurazione dei bean.

La classe AnnotationConfigApplicationContext, invece, utilizza codice Java per i metadati.

Il container IoC

Il diagramma mostra una visione ad alto livello del funzionamento del container IoC di Spring.



Il container IoC

Come istanziare ed utilizzare un container

1. Creo un'istanza della classe che implementa il container da utilizzare

```
ApplicationContext context = new ClassPathXmlApplicationContext("my-beans.xml");
```

2. Invoco il metodo **getBean(...)** per recuperare l'istanza di un bean

```
MyClass service = context.getBean("myclass", MyClass.class);
```

Alcune classi che implementano ApplicationContext

- *org.springframework.context.support.ClassPathXmlApplicationContext*

Ha diversi costruttori tra cui:

- *ClassPathXmlApplicationContext(String... configLocations)*
- *ClassPathXmlApplicationContext(String configLocation)*
- *ClassPathXmlApplicationContext(String[] configLocations, ApplicationContext parent)*

Utilizza file XML per il recupero dei metadati

Il percorso del (o dei) file XML viene passato in ingresso al costruttore.

La classe **legge file XML che si trovano nel classpath dell'applicazione**, ovvero:

- nella directory **classes** della web application
- in un **jar** della directory **lib**.

Alcune classi che implementano ApplicationContext

- *org.springframework.context.support.FileSystemXmlApplicationContext*

Ha diversi costruttori tra cui:

- *FileSystemXmlApplicationContext(String... configLocations)*
- *FileSystemXmlApplicationContext(String configLocation)*
- *FileSystemXmlApplicationContext(String[] configLocations, ApplicationContext parent)*

Utilizza file XML per il recupero dei metadati

Il percorso del (o dei) file XML viene passato in ingresso al costruttore.

La classe **legge file XML che si trovano nel file system**, ad es. c:/my-app/config/my-file.xml

Alcune classi che implementano ApplicationContext

- *org.springframework.context.annotation.AnnotationConfigApplicationContext*

Ha diversi costruttori tra cui:

- *AnnotationConfigApplicationContext()*
- *AnnotationConfigApplicationContext(Class<?>... componentClasses)*
- *AnnotationConfigApplicationContext(DefaultListableBeanFactory beanFactory)*

Utilizza classi che contengono le annotation per la definizione dei metadati

Di cosa abbiamo parlato in questa lezione

- Come funziona il container IoC in Spring