Le nuove annotation Spring per la configurazione dei Bean – Parte I

Cosa imparerai

- Come configurare i metadati utilizzando le nuove annotation di Spring

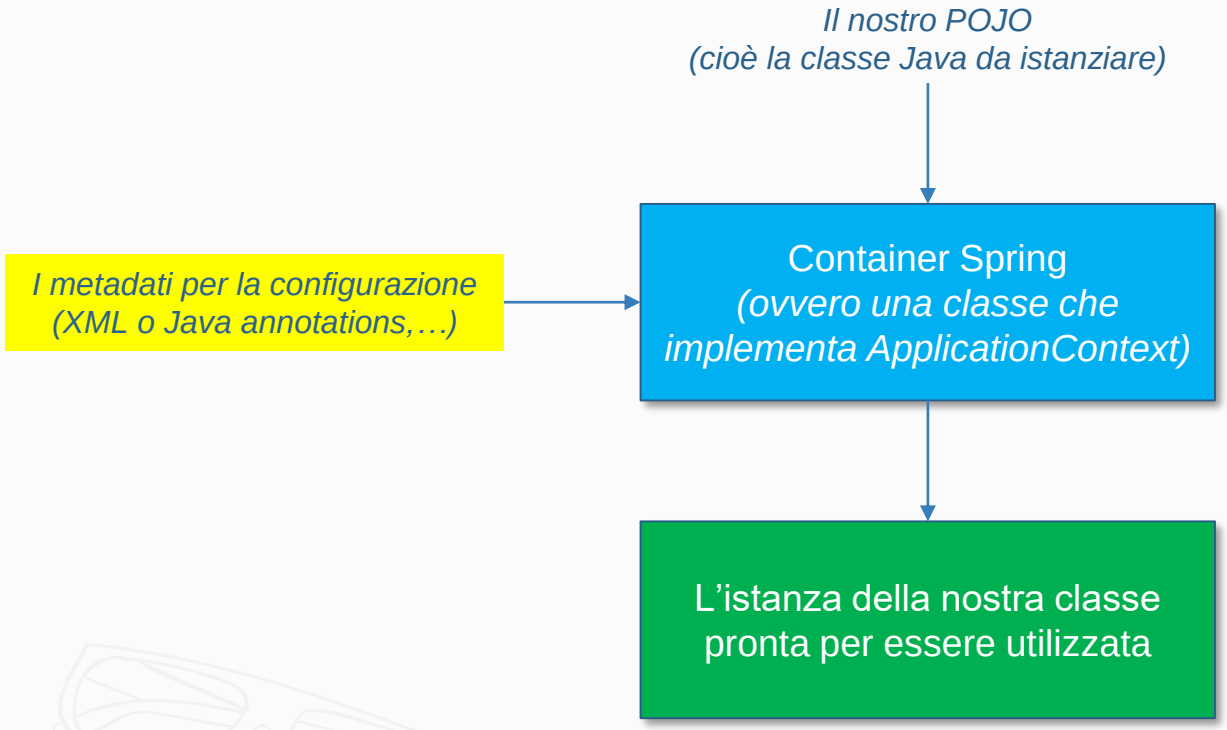


paolopreite

Le nuove annotation Spring per la configurazione dei Bean

Il container Spring per istanziare i bean utilizza dei metadati di configurazione.

Questi metadati possono essere espressi con notazione XML o Java annotation (abbiamo visto le annotation @Component, le annotation JSR-330).



Le nuove annotation Spring per la configurazione dei Bean

Al posto di utilizzare la notazione XML, possiamo utilizzare le nuove annotation che consentono di definire e configurare i bean.

Le **due** principali annotation sono: **@Bean** e **@Configuration**

Queste annotation si trovano nel package **org.springframework.context**.

L'annotation @Bean

Viene utilizzata per indicare che un metodo istanzia, configura e inizializza un nuovo oggetto che deve essere gestito dal container.

L'annotation @Bean svolge lo stesso ruolo dell'elemento <bean /> della notazione XML.

Le nuove annotation Spring per la configurazione dei Bean

L'annotation @Configuration

Indica che la classe ha come scopo principale quello di fornire le definizioni dei bean.

Le classi annotate con *@Configuration* consentono di definire anche le dipendenze tra i bean.

L'annotation *@Configuration* annotation è l'analogo del file XML e consente di evitare l'utilizzo della notazione XML-based.

L'implementazione da utilizzare per utilizzare la configurazione tramite annotation è:

org.springframework.context.annotation.AnnotationConfigApplicationContext

Le nuove annotation Spring per la configurazione dei Bean

Facciamo un esempio…

```java
@Configuration
public class Config {
    @Bean
    public ClienteService getClienteService() {
        return new ClienteService();
    }

    @Bean
    public OrdineService getOrdineService() {
        return new OrdineService();
    }

    @Bean
    public ProdottoService getProdottoService() {
        return new ProdottoService();
    }
}
```

```java
public class Main {
    public static void main(String[] args) {
        AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(Config.class);

        ProdottoService prodotto = context.getBean(ProdottoService.class);

        System.out.println(prodotto.sayHello());
    }
}
```

Le nuove annotation Spring per la configurazione dei Bean

Possiamo utilizzare l'implementazione **AnnotationConfigApplicationContext** anche per i bean definiti tramite annotation *@Component* o annotation *JR-330*.

L'annotation *@ComponentScan(basePackages="...")* consente di definire il package dove sono situati i bean.

Esempio…

Il bean

```
import org.springframework.stereotype.Component;

@Component
public class ProdottoService {
    public ProdottoService() {
        super();
    }

    public String sayHello() {
        String ret = "TV 32''";

        return ret;
    }
}
```

XML-based Metadata configuration

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

    <context:component-scan base-package="it.test.service" />
</beans>
```

Il context…

```
public class Main {
    public static void main(String[] args) {
        ApplicationContext context =
                new ClassPathXmlApplicationContext("services.xml");

        OrdineService o = context.getBean("ordineService", OrdineService.class);

        System.out.println(o.sayHello());
    }
}
```

Annotation Metadata configuration

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan(basePackages = "it.test.nuove.component")
public class ConfigScan {

}
```

Il context…

```
public class Main {
    public static void main(String[] args) {
        AnnotationConfigApplicationContext context =
                new AnnotationConfigApplicationContext(ConfigScan.class);

        ProdottoService prodotto = context.getBean(ProdottoService.class);

        System.out.println(prodotto.sayHello());
    }
}
```

Di cosa abbiamo parlato in questa lezione

- Come configurare i metadati utilizzando codice Java