Null-safety con Spring

Cosa imparerai

- Quali sono gli strumenti messi a disposizione da Spring per gestire variabili *null*





paolopreite

Null-safety con Spring

Il terrore di tutti i programmatori Java

java.lang.NullPointerException

Questa eccezione è una *RuntimeException* e viene generata quando un'applicazione tenta di utilizzare un riferimento a un oggetto con valore *null*.

```
public class Main {
    public static void main(String[] args) {
        ApplicationContext context = null;

        OrdineService o = context.getBean(OrdineService.class);

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

```
Exception in thread "main" java.lang.NullPointerException: Cannot invoke
"org.springframework.context.ApplicationContext.getBean(java.lang.Class)" because
"context" is null
        at it.test.Main.main(Main.java:12)
```

Null-safety con Spring

Java non mette a disposizione strumenti per il controllo dello stato di un oggetto.

Per sapere se un oggetto è *null* l'unica cosa che possiamo fare è uno statement *if(object != null)…*

```
public static void main(String[] args) {
    ApplicationContext context = null;

    if(context != null) {
        OrdineService o = context.getBean(OrdineService.class);

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

Null-safety con Spring

Per sopperire a questa mancanza di Java, il framework Spring mette a disposizione degli svilluppatori alcune annotation per gestire il «*nullability*» di API e attributi.

Queste annotation si trovano nel package **org.springframework.lang** e sono:

- **@Nullable**: utilizzata per indicare che un parametro, un valore restituito o un attributo può essere *null*.
- **@NonNull**: utilizzata per indicare che un parametro, un valore restituito o un attributo deve essere sempre diversa da *null*.
- **@NonNullApi**: utilizzata a livello di package, indica che parametri e valori restituiti da tutti i metodi presenti nel package sono sempre diversi da *null*.
- **@NonNullFields**: utilizzata a livello di package, indica che gli attributi di un bean sono sempre diversi da *null*.

Null-safety con Spring

Quando usare queste annotation

Le annotation hanno lo scopo di:

- fornire una dichiarazione esplicita per il «*nullability*» di un componente
- fornire nell'IDE (come IntelliJ IDEA o Eclipse) warning relativi alla «*nullability*» di un componente, al fine di evitare eccezioni *NullPointerException* in fase di esecuzione.

Le annotation sono anche usate per rendere *null-safe* le API di Spring nei progetti Kotlin, dal momento che Kotlin supporta nativamente il *null-safety*.

Di cosa abbiamo parlato in questa lezione

- Quali sono gli strumenti messi a disposizione da Spring per gestire variabili *null*