La risoluzione delle view con il ViewResolver

Cosa imparerai

- Come definire il ViewResolver in una web app Spring MVC

La risoluzione delle view con il ViewResolver

Come viene idenfiticata la view che dovrà gestire la generazione dell'output?

Il controller utilizza un oggetto di tipo **ViewResolver** che identifica la view sulla base di uno dei seguenti casi:

- **il metodo ritorna String**: il valore della stringa ritornata dal metodo coincide con il nome della view
- **il metodo ritorna un oggetto di tipo View:** ed il nome della view è incapsulato nell'oggetto

In pratica…

1. **La View** identifica la risorsa che deve generare l'output da inviare all'utente (JSP, file Freemarker, ecc…)

2. **Il ViewResolver** identifica la View sulla base di una stringa.

La risoluzione delle view con il ViewResolver

Spring MVC non ha un ViewResolver di default. Infatti, se non specifichiamo il view resolver ed invochiamo una URL la nostra web app andrà in errore…

```java
@Controller
@RequestMapping(path = "/")
public class HelloController {
    @GetMapping(path = "/hello")
    public String hello() {
        return "redirect:/bye";
    }

    @GetMapping(path = "/helloForward")
    public String helloForward() {
        return "forward:/bye";
    }

    @GetMapping(path = "/bye")
    public String bye() {
        return "bye";
    }
}
```

La risoluzione delle view con il ViewResolver

Definire un ViewResolver

In Spring è possibile definire diversi tipi ViewResolver, in base alla tecnologia utilizzata per realizzare le view.

Se utilizziamo la notazione Java-based, il ViewResolver va configurato nella classe che contiene la configurazione.

```
@EnableWebMvc
@Configuration
@ComponentScan(basePackages = { "it.test.web.controller" })
public class WebConfig  {
    @Bean
    public ViewResolver viewResolver() {
        InternalResourceViewResolver bean = new InternalResourceViewResolver();

        bean.setPrefix("/WEB-INF/view/");
        bean.setSuffix(".jsp");

        return bean;
    }
}
```

La risoluzione delle view con il ViewResolver

Definire un ViewResolver

Un tipo di ViewResolver è **InternalResourceViewResolver** (package *org.springframework.web.servlet.view*) che consente di identificare una view interne al progetto (tipicamente file JSP).

Per definire l'**InternalResourceViewResolver** è necessario:

- ❑ Definire un bean di tipo **InternalResourceViewResolver**
- ❑ Definire nel bean il path dove si trovano i file
- ❑ Definire l'estensione dei file

```
@EnableWebMvc
@Configuration
@ComponentScan(basePackages = { "it.test.web.controller" })
public class WebConfig  {
    @Bean
    public ViewResolver viewResolver() {
        InternalResourceViewResolver bean = new InternalResourceViewResolver();

        bean.setPrefix("/WEB-INF/view/");
        bean.setSuffix(".jsp");

        return bean;
    }
}
```

A questo punto possiamo già utilizzare le JSP per creare le view della web app.

La risoluzione delle view con il ViewResolver

Definire un ViewResolver

L'**InternalResourceViewResolver** consente anche di utilizzare JSTL all'interno delle JSP.

Per fare questo è necessario:

1. Aggiungere la dipendenza da JSTL nel POM (se usiamo Maven) o nel file di Gradle

```
<!-- https://mvnrepository.com/artifact/javax.servlet/jstl -->
<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>
```

2. Specificare al bean che stiamo usando JSTL utilizzando come view class JstlView (org.springframework.web.servlet.view.JstlView)

```
public class WebConfig  {
    @Bean
    public ViewResolver viewResolver() {
        InternalResourceViewResolver bean = new InternalResourceViewResolver();

        bean.setViewClass(JstlView.class);
        bean.setPrefix("/WEB-INF/view/");
        bean.setSuffix(".jsp");

        return bean;
    }
}
```

La risoluzione delle view con il ViewResolver

Quanti ViewResolver esistono?

Tanti!

FreeMarkerViewResolver

JasperReportsViewResolver

XsltViewResolver

…

Di cosa abbiamo parlato in questa lezione

- Come definire il ViewResolver in una web app Spring MVC