Come utilizzare FreeMarker con Spring MCV

Cosa imparerai

- Quali sono le dipendenze necessarie e come configurare la web app per utilizzare FreeMarker

Come utilizzare FreeMarker con Spring MCV

Per utilizzare FreeMarker in una web app Spring MVC dobbiamo:

1. **definire la dipendenza da FreeMarker e dal modulo Spring Context Support**

Se usiamo Maven, la dipendenza da aggiungere al POM è:

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
    <version>5.3.2</version> (o altra versione…)
</dependency>

<dependency>
    <groupId>org.freemarker</groupId>
    <artifactId>freemarker</artifactId>
    <version>2.3.30</version> (o altra versione…)
</dependency>
```

ATTENZIONE: il groupId corretto è org.freemarker. Ci sono vecchie version di FreeMarker che hanno come groupId freemarker.

Come utilizzare FreeMarker con Spring MCV

2. definire il ViewResolver adatto ad interpretare i template di FreeMarker

Se usiamo la notazione Java-based dobbiamo implementare un metodo che ritorni un oggetto di tipo **FreeMarkerViewResolver** (package *org.springframework.web.servlet.view.freemarker*) che conterrà l'estensione data ai file dei template (generalmente lasciamo ***.ftl***)

```
@Bean
public FreeMarkerViewResolver freemarkerViewResolver() {
    FreeMarkerViewResolver bean = new FreeMarkerViewResolver();

    bean.setCache(true);   ← usiamo questo metodo per abilitare il caching dei template FreeMarker
    bean.setPrefix("");
    bean.setSuffix(".ftl");

    return bean;
}
```

Come utilizzare FreeMarker con Spring MCV

3. configurare l'oggetto che conterrà la confgurazione utilizzata da FreeMarker

Se usiamo la notazione Java-based dobbiamo implementare un metodo che ritorni un oggetto di tipo **FreeMarkerConfigurer** (package *org.springframework.web.servlet.view.freemarker*) e che conterrà il path dove inseriremo i nostri file **.ftl**.

```
@Bean
public FreeMarkerConfigurer freeMarkerConfigurer() {
    FreeMarkerConfigurer configurer = new FreeMarkerConfigurer();
    configurer.setTemplateLoaderPath("/WEB-INF/view-ftl/");

    return configurer;
}
```

Come utilizzare FreeMarker con Spring MCV

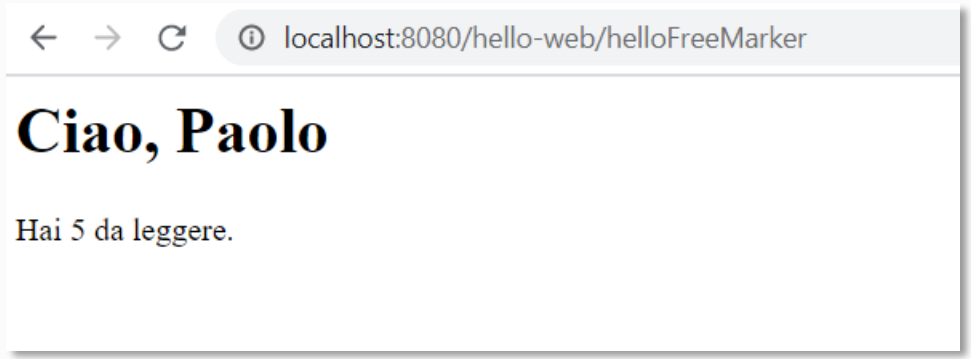
A questo punto possiamo creare i nostri template e mapparli nel controller…

```java
@Controller
@RequestMapping(path = "/")
public class HelloController {
    @GetMapping(path = "/helloFreeMarker")
    public String helloFreeMarker(ModelMap model) {
        model.addAttribute("userFirstname", "Paolo");
        model.addAttribute("unreadMessages", "5");

        return "index";
    }
```

index.ftl

```
<!DOCTYPE>
<html>
    <head></head>
    <body>
        <h1>Ciao, ${userFirstname}</h1>
        <p>Hai ${unreadMessages} da leggere.</p>
    </body>
</html>
```



Di cosa abbiamo parlato in questa lezione

- Quali sono le dipendenze necessarie e come configurare la web app per utilizzare FreeMarker