Acabamos de ver como usar Java equals y hashcode

Las aplicaciones de Spring Boot JSP pueden ser más habituales de lo que en un primer momento nos parece . Sí , si es cierto que usar Spring Boot es usar tecnologías modernas que facilitan que despleguemos arquitecturas orientadas a MicroServicios . Pero hay que recordar que bueno … seguimos teniendo aplicaciones no tan “modernas” y quizás podamos necesitar que se ejecuten dentro de Spring Boot . Cuando nosotros construimos un proyecto web por defecto en Spring Boot el fichero pom.xml tiene la siguiente estructura.

```
&amp;lt;?xml version="1.0" encoding="UTF-8"?&amp;gt;
&amp;lt;project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd"&amp;gt;
        &amp;lt;modelVersion&amp;gt;4.0.0&amp;lt;/modelVersion&amp;gt;

&amp;lt;groupId&amp;gt;com.arquitecturajava&amp;lt;/groupId&amp;gt;
&amp;lt;artifactId&amp;gt;sprinbootweb&amp;lt;/artifactId&amp;gt;
        &amp;lt;version&amp;gt;0.0.1-SNAPSHOT&amp;lt;/version&amp;gt;
        &amp;lt;packaging&amp;gt;jar&amp;lt;/packaging&amp;gt;

        &amp;lt;name&amp;gt;sprinbootweb&amp;lt;/name&amp;gt;
        &amp;lt;description&amp;gt;Demo project for Spring
Boot&amp;lt;/description&amp;gt;

        &amp;lt;parent&amp;gt;
&amp;lt;groupId&amp;gt;org.springframework.boot&amp;lt;/groupId&amp;gt
;
                &amp;lt;artifactId&amp;gt;spring-boot-starter-
```

```
parent&amp;lt;/artifactId&amp;gt;
&amp;lt;version&amp;gt;2.1.0.RELEASE&amp;lt;/version&amp;gt;
                &amp;lt;relativePath /&amp;gt; &amp;lt;!-- lookup
parent from repository --&amp;gt;
        &amp;lt;/parent&amp;gt;

        &amp;lt;properties&amp;gt;
&amp;lt;project.build.sourceEncoding&amp;gt;UTF-8&amp;lt;/project.buil
d.sourceEncoding&amp;gt;
&amp;lt;project.reporting.outputEncoding&amp;gt;UTF-8&amp;lt;/project.
reporting.outputEncoding&amp;gt;
&amp;lt;java.version&amp;gt;1.8&amp;lt;/java.version&amp;gt;
        &amp;lt;/properties&amp;gt;

        &amp;lt;dependencies&amp;gt;
                &amp;lt;dependency&amp;gt;
&amp;lt;groupId&amp;gt;org.springframework.boot&amp;lt;/groupId&amp;gt
;
                        &amp;lt;artifactId&amp;gt;spring-boot-starter-
web&amp;lt;/artifactId&amp;gt;
                &amp;lt;/dependency&amp;gt;

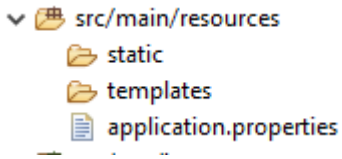
                &amp;lt;dependency&amp;gt;
&amp;lt;groupId&amp;gt;org.springframework.boot&amp;lt;/groupId&amp;gt
;
                        &amp;lt;artifactId&amp;gt;spring-boot-starter-
test&amp;lt;/artifactId&amp;gt;
&amp;lt;scope&amp;gt;test&amp;lt;/scope&amp;gt;
                &amp;lt;/dependency&amp;gt;

        &amp;lt;/dependencies&amp;gt;
```

```
        &amp;lt;build&amp;gt;
                &amp;lt;plugins&amp;gt;
                        &amp;lt;plugin&amp;gt;
&amp;lt;groupId&amp;gt;org.springframework.boot&amp;lt;/groupId&amp;gt
;
                                &amp;lt;artifactId&amp;gt;spring-boot-
maven-plugin&amp;lt;/artifactId&amp;gt;
                        &amp;lt;/plugin&amp;gt;
                &amp;lt;/plugins&amp;gt;
        &amp;lt;/build&amp;gt;

&amp;lt;/project&amp;gt;
```

Junto con ello tenemos una sencilla carpeta de resources en donde nos aparecen los ficheros y carpetas fundamentales para plantillas.



El problema es que nosotros no usamos plantillas de thymeleaf sino que tenemos ficheros JSP y una aplicación bastante clásica de Spring . ¿Como podemos integrar JSP sin volvernos muy locos?.

Spring Boot JSP

En este caso es relativamente sencillo . Lo primero que tendremos que hacer es añadir la dependencia de

```
&amp;lt;dependency&amp;gt;
```

```
&amp;lt;groupId&amp;gt;org.apache.tomcat.embed&amp;lt;/groupId&amp;gt;
                        &amp;lt;artifactId&amp;gt;tomcat-embed-
jasper&amp;lt;/artifactId&amp;gt;
&amp;lt;scope&amp;gt;provided&amp;lt;/scope&amp;gt;
                &amp;lt;/dependency&amp;gt;
```

Que nos aporta soporte para JSP dentro del proyecto de Spring Boot . El segundo paso es crear las carpetas clásicas donde nuestros ficheros JSP van a ir.



El siguiente paso es abrir el fichero de application.properties y dar de alta la carpeta con la extensión que necesita el ViewResolver.

```
spring.mvc.view.prefix=/WEB-INF/vistas/
spring.mvc.view.suffix=.jsp
```

Es momento de crear un controlador que nos redirija a la página mensaje.jsp .

```
package com.arquitecturajava.sprinbootweb;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
public class HolaController {
```

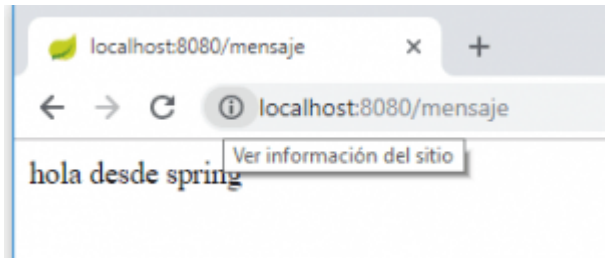
```
        @RequestMapping("/mensaje")
        public String mensaje() {

                return "mensaje";
        }
}
```

Una vez creado el controlador es momento de ver el contenido de la página JSP . En este caso se trata de algo muy sencillo.

```
&amp;lt;html&amp;gt;
&amp;lt;body&amp;gt;
hola desde spring
&amp;lt;/body&amp;gt;
&amp;lt;/html&amp;gt;
```

Unicamente nos queda arrancar la aplicación de Spring boot y solicitar la url de http://localhost:8080/mensaje . Automáticamente Spring boot nos redigirá a la vista JSP que tenemos.



Acabamos de construir nuestro primer proyecto de Spring Boot JSP . Utilizando una tecnología moderna para poner en funcionamiento tecnologías antiguas.

Otros artículos relacionados

1. Spring Boot JPA y su configuración
2. Spring Boot @Lazy Components
3. Spring Boot AOP y rendimiento
4. @RepositoryRestResource y Spring Framework

Externos

1. Spring Boot