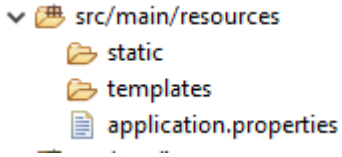
Las aplicaciones de Spring Boot JSP pueden ser más habituales de lo que en un primer momento nos parece . Sí , si es cierto que usar Spring Boot es usar tecnologías modernas que facilitan que despleguemos arquitecturas orientadas a MicroServicios . Pero hay que recordar que bueno … seguimos teniendo 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.

```
<?xml version="1.0" encoding="UTF-8"?>
<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">
	<modelVersion>4.0.0</modelVersion>

	<groupId>com.arquitecturajava</groupId>
	<artifactId>sprinbootweb</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<packaging>jar</packaging>

	<name>sprinbootweb</name>
	<description>Demo project for Spring Boot</description>

	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>2.1.0.RELEASE</version>
		<relativePath /> <!-- lookup parent from repository -
->
	</parent>
```

```
        <properties>
<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncod
ing>
                <java.version>1.8</java.version>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-
web</artifactId>
                </dependency>

                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-
test</artifactId>
                        <scope>test</scope>
                </dependency>

        </dependencies>

        <build>
                <plugins>
                        <plugin>
<groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-
plugin</artifactId>
                        </plugin>
                </plugins>
```

```
        </build>


</project>
```

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

```
<dependency>
                        <groupId>org.apache.tomcat.embed</groupId>
                        <artifactId>tomcat-embed-jasper</artifactId>
                        <scope>provided</scope>
                </dependency>
```

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
```

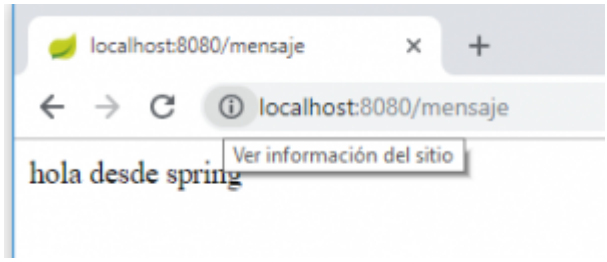
Es momento de crear un controlador que nos redirija a la página mensaje.jsp .

```
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.

```
<html>
<body>
hola desde spring
</body>
</html>
```

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