



Certified Tech Developer

The Ultimate Degree

Back End I

Ejercitación patrón MVC (vista con plantilla)

Objetivo

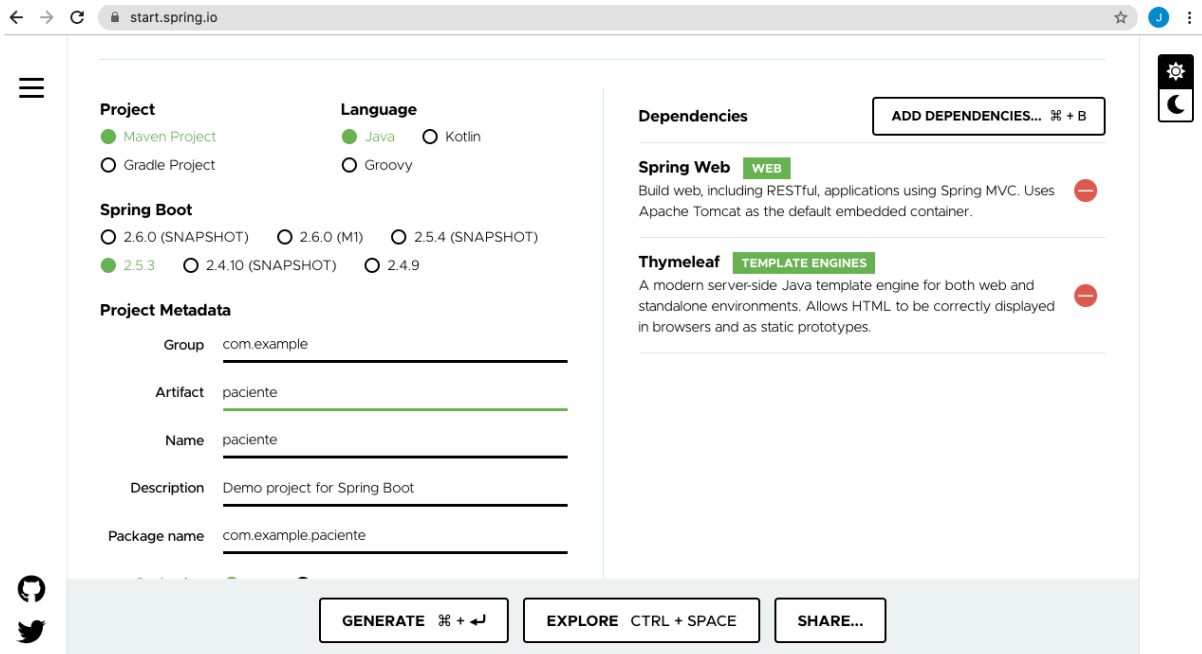
Continuando con las ejercitaciones realizadas en la clase asincrónica y con el docente, vamos a crear un proyecto Paciente en Spring MVC siguiendo las instrucciones.

- Ejercicio individual
- Complejidad: baja

Instrucciones

1. Crear un proyecto desde <https://start.spring.io>.

Recordá poner el nombre del proyecto y agregar las dependencias Spring Web y Thymeleaf.

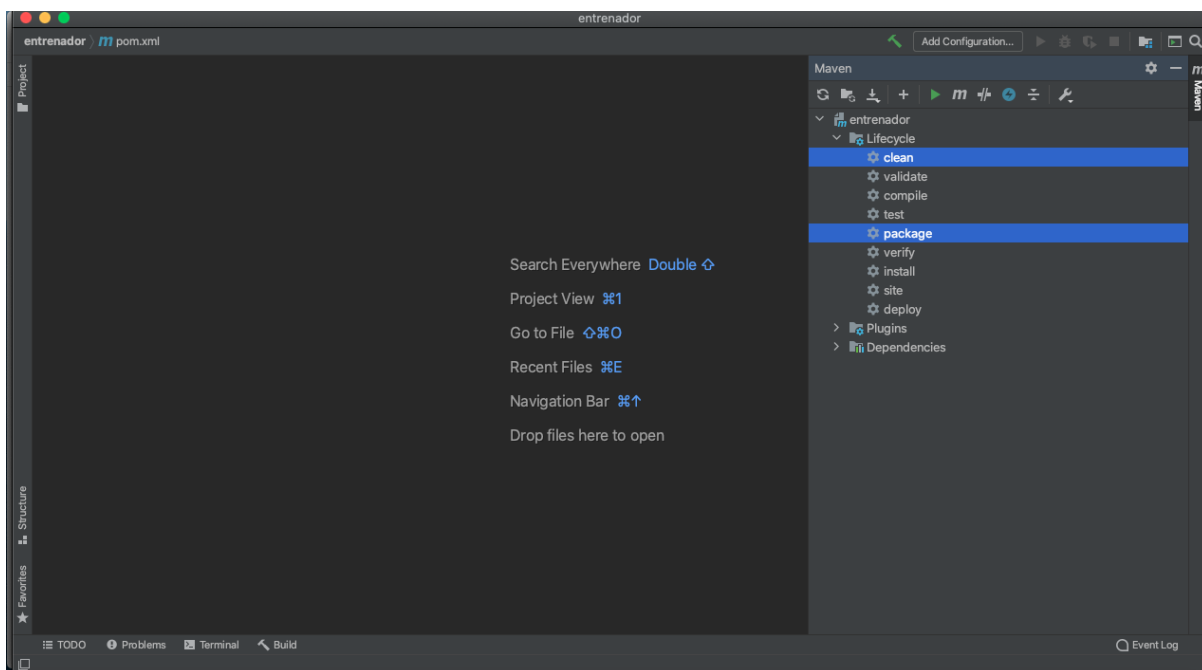


The screenshot shows the start.spring.io web interface. The browser address bar shows 'start.spring.io'. The interface is divided into several sections:

- Project:**
 - ☒ Maven Project
 - ☐ Gradle Project
- Language:**
 - ☒ Java
 - ☐ Kotlin
 - ☐ Groovy
- Spring Boot:**
 - ☐ 2.6.0 (SNAPSHOT)
 - ☐ 2.6.0 (M1)
 - ☐ 2.5.4 (SNAPSHOT)
 - ☒ 2.5.3
 - ☐ 2.4.10 (SNAPSHOT)
 - ☐ 2.4.9
- Project Metadata:**
 - Group: com.example
 - Artifact: paciente
 - Name: paciente
 - Description: Demo project for Spring Boot
 - Package name: com.example.paciente
- Dependencies:**
 - Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container. (Add button)
 - Thymeleaf** (TEMPLATE ENGINES): A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes. (Add button)

At the bottom, there are three buttons: **GENERATE** (⌘ + ↵), **EXPLORE** (CTRL + SPACE), and **SHARE...**

2. Generar el proyecto y descomprimir el archivo .zip para abrirlo en IntelliJ IDEA. En IntelliJ IDEA, dirigirse a "New -> Existing source".
3. En la solapa de Maven, hacer clean package y presionar "Play".





4. Crear un controller en el paquete Controller.

```
@Controller
public class PacienteController {

    @GetMapping("/index")
    public String welcome(Model model) {
        model.addAttribute("nombre", "diez");
        return "index";
    }
}
```

Vemos cómo el método **welcome** tiene el Model por parámetro. Esta es la manera que tenemos de mandar datos del controlador a la vista.

En la última línea, return "index" va a hacer mención a la plantilla HTML.

5. Crear un archivo index.html dentro de resources/templates.

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <title>Users</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.mi
n.css"
integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8
ERdknLPM0" crossorigin="anonymous">
```



```
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
integrity="sha384-5sAR7xN1Nv6T6+dT2mhtzEpVJvfS3NScPQTr0xhwjIuvcA67KV2R5Jz
6kr4abQsz" crossorigin="anonymous">
<link rel="stylesheet" href="../css/shards.min.css">
</head>
<body>
<td th:text="${nombre}"></td>

</body>
</html>
```

6. Correr la clase main, dentro de PacienteApplication. Luego, ir al navegador y poner <http://localhost:8080/index>. Allí, verás cómo nuestra vista está mostrando el valor de nombre que vino desde el modelo.