

# Tema 2 – Spring Boot

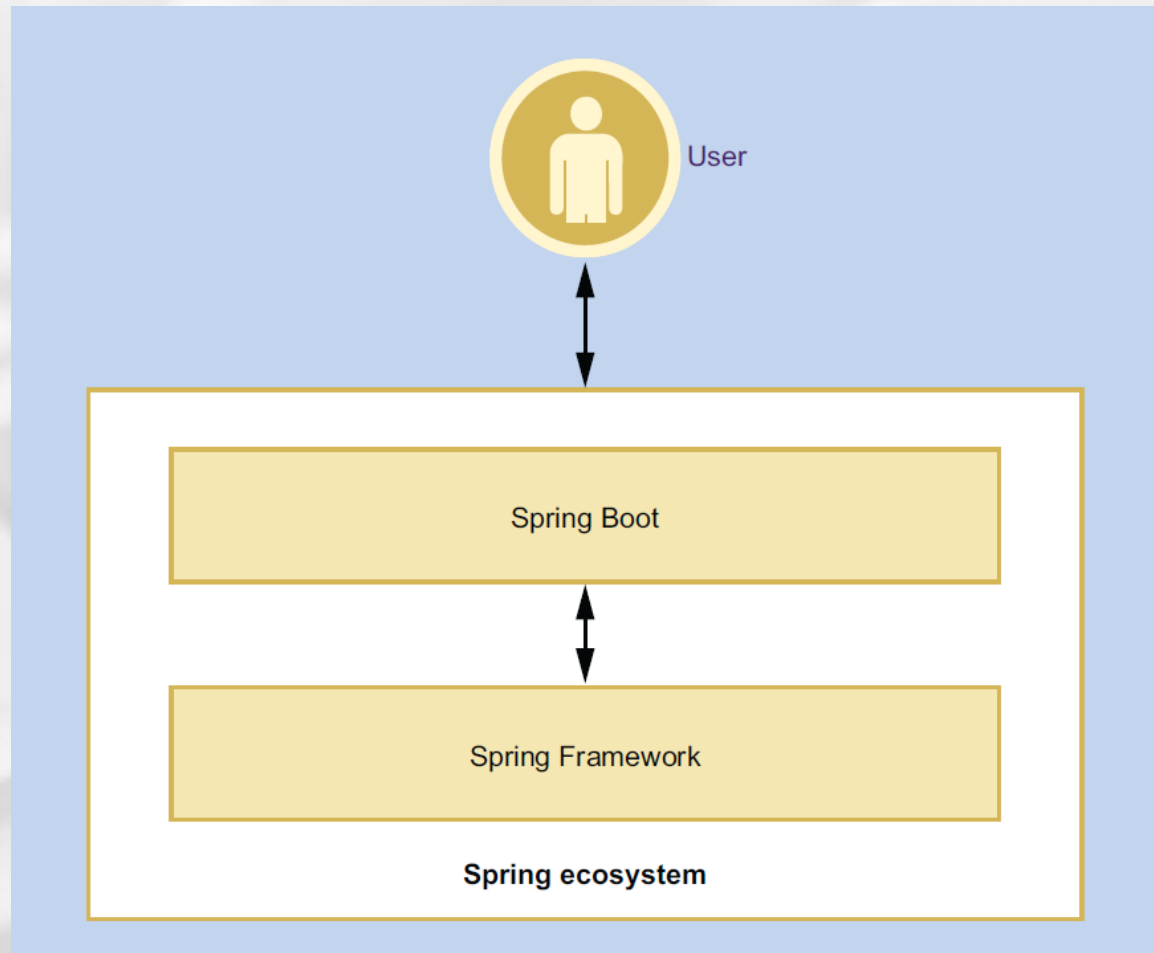
Grado en Ingeniería Informática en Tecnologías  
de la Información

Departamento de Ingeniería de Sistemas Informáticos y Telemáticos

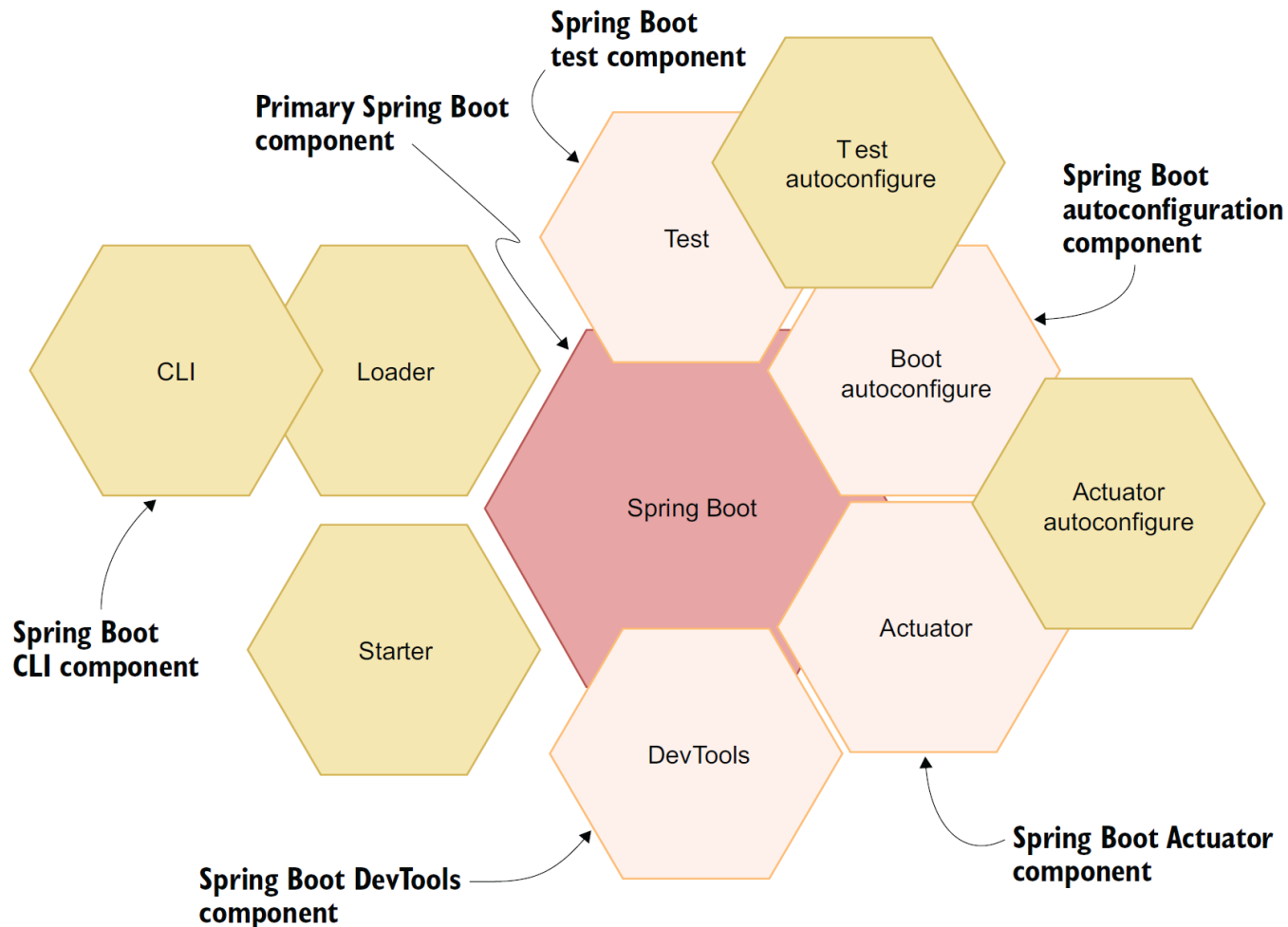
Área de Lenguajes y Sistemas Informáticos

Dr. Luis V. Calderita

# Recordatorio




# Spring Boot Componentes



# Primer proyecto Spring Boot

- Usaremos una aplicación web:
  - Spring Initializr: <https://start.spring.io>
- Esta aplicación web nos permite seleccionar las opciones de nuestro proyecto Spring Boot
  - ATB de construcción, el lenguaje, la versión de Spring Boot, incorporar los metadatos del proyecto, el tipo de empaquetamiento, la versión de la JVM que tenemos instalada, las dependencias que necesitamos.
- Spring Initializr permite explorar y generar el proyecto en .zip

# Spring Initializr: <https://start.spring.io>

 **spring initializr**

---

**Project**  
☒ Maven Project  
☐ Gradle Project

**Language**  
☒ Java ☐ Kotlin  
☐ Groovy

**Spring Boot**  
☐ 3.0.0 (SNAPSHOT) ☐ 3.0.0 (M5)  
☐ 2.7.5 (SNAPSHOT) ☒ 2.7.4  
☐ 2.6.13 (SNAPSHOT) ☐ 2.6.12

**Project Metadata**  

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 19 ☒ 17 ☐ 11 ☐ 8

**Dependencies** ADD ... CTRL + B  
**Spring Web** WEB  
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

GENERATE CTRL + G

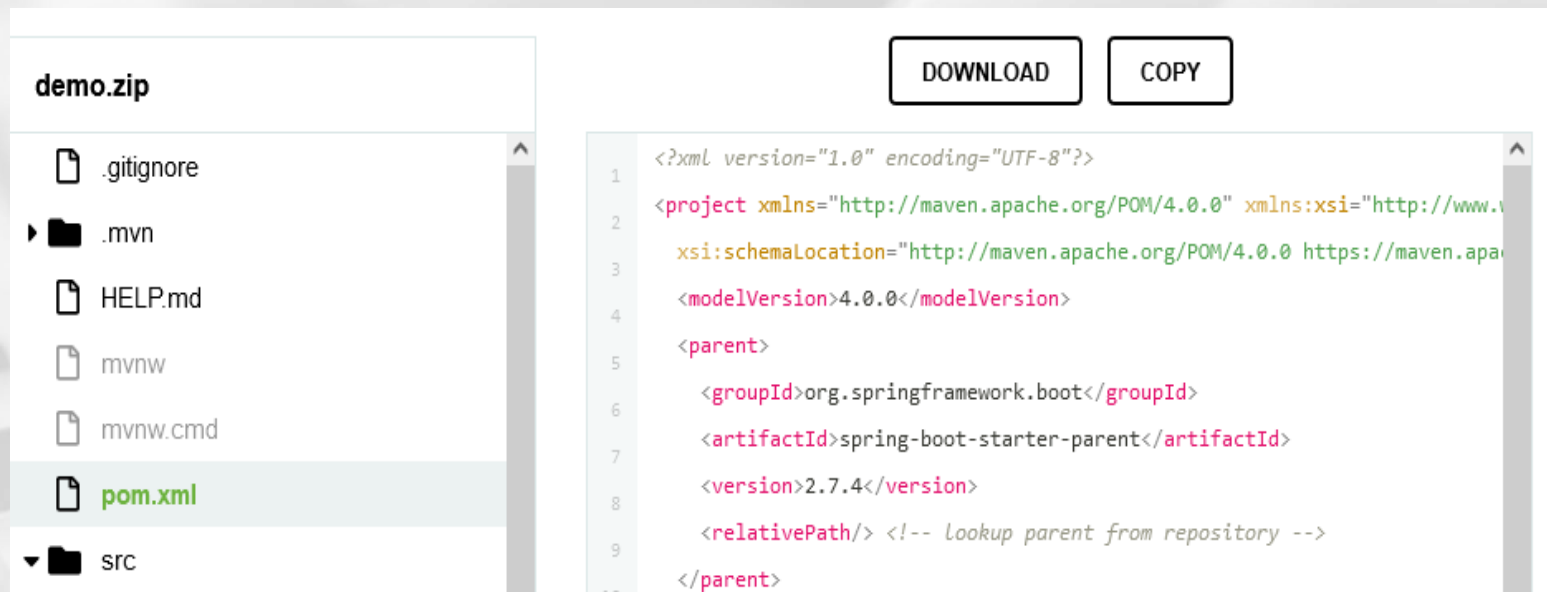
EXPLORE CTRL + SPACE

SHARE...

# Configuración

- Project: Maven
- Lenguaje: Java
- Spring Boot versión:
  - La seleccionada es la versión actual estable.
- Project Metada: Name, Artifact, Group...
- Packaging: Jar
- Java:
  - La versión que tengáis instalada
- Dependencies: Spring Web

# Spring Initializr: Opción Explore



The screenshot displays the 'Explore' option in Spring Initializr. On the left, a file explorer for 'demo.zip' shows a list of files: .gitignore, .mvn, HELP.md, mvnw, mvnw.cmd, and pom.xml (highlighted in green), and a 'src' folder. On the right, there are 'DOWNLOAD' and 'COPY' buttons. Below these, a preview of the pom.xml file is shown with line numbers 1 through 10. The XML content is as follows:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/maven-v4_0_0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.7.4</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
```

# Estructura del proyecto: POM

El proyecto actual declara a Spring Boot starter parent como su padre, por tanto, indica que este proyecto es un proyecto hijo de Spring Boot.

Esto asegura que varias características, como la gestión de plugins y dependencias, puedan ser gestionadas por Spring Boot.

```
<?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
    https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.7.4</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>
```



# Estructura del proyecto: POM

Detalles del artefacto del proyecto actual.

- Groupid, lo identifica entre otros proyectos
- artifactId, Identifica el artefacto y es el nombre del jar sin la versión
- Version, versión actual del proyecto
- Name, nombre para los humanos
- Description, describe el proyecto

```
</parent>
<groupId>com.example</groupId>
<artifactId>demo</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>demo</name>
<description>Demo project for Spring Boot</description>
<properties>
  <java.version>17</java.version>
</properties>
<dependencies>
```

# Estructura del proyecto: POM

Lista de dependencias seleccionadas:

- Spring Boot starter web
- Spring Boot starter test

```
<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>

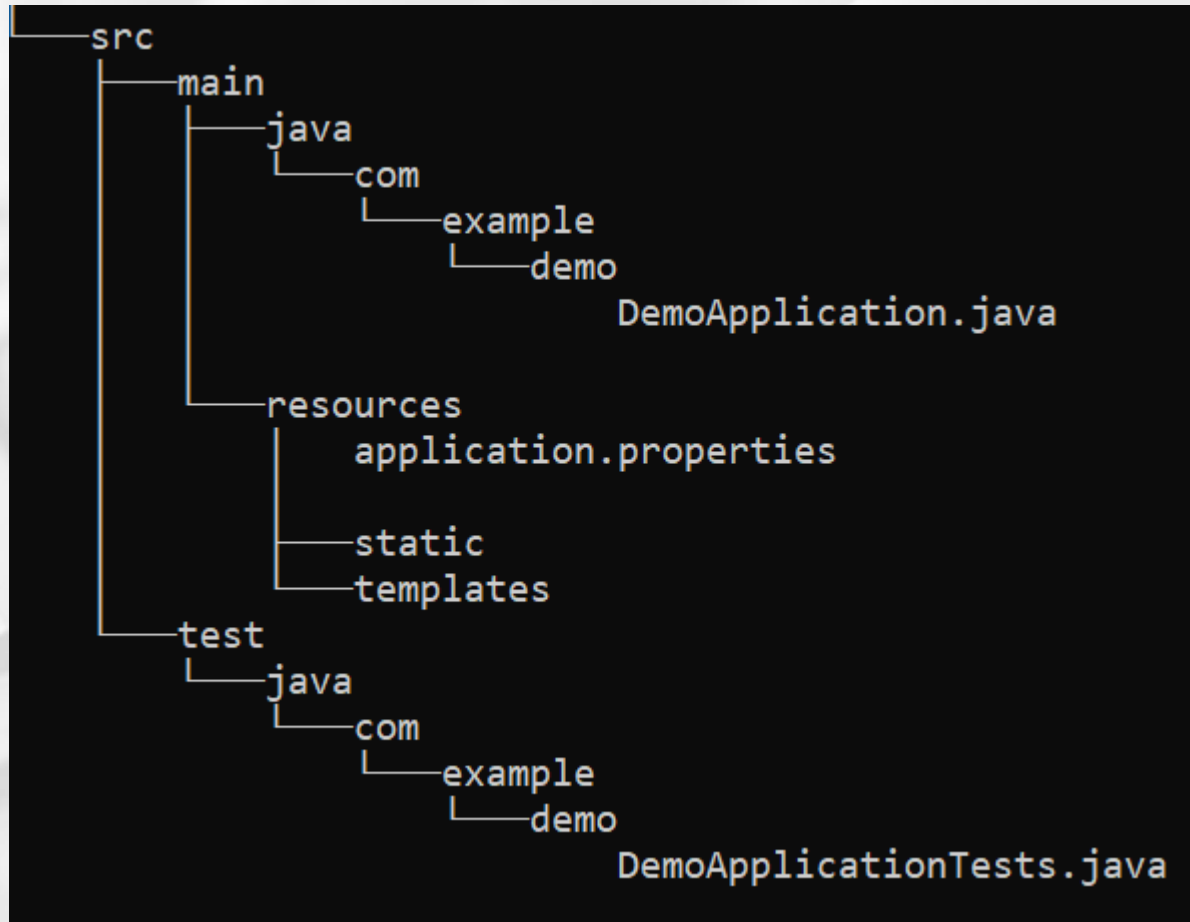
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
</project>
```

Spring Boot Maven Plugin. Es un plugin de Maven para realizar varias actividades relacionadas con la gestión de la aplicación

# Estructura de carpetas

```
C:\Users\luiky\Downloads\demo>tree /f
Listado de rutas de carpetas
El número de serie del volumen es 1A9D-C0BE
C:.\
|
|  .gitignore
|  HELP.md
|  mvnw
|  mvnw.cmd
|  pom.xml
|
|  .mvn
|  |  └─ wrapper
|  |         maven-wrapper.jar
|  |         maven-wrapper.properties
```

# Estructura de carpetas



# Contenido del proyecto generado

- pom.xml (Project Object Model)
  - Fichero con la configuración y dependencias del proyecto para Maven
- Un wrapper de Maven
  - Para construir el proyecto sin necesidad de instalar Maven. (mvnw)
- El código fuente del proyecto (src)
  - Incluye una clase Java con el método main (DemoApplication.java)
  - Esta clase permite iniciar la aplicación Spring Boot
  - Es un fichero que puede ejecutarse desde un IDE

# Contenido del proyecto generado

- Código de prueba del proyecto (test)
  - Incluye una clase para tests vacía (`DemoApplicationTests.java`) para escribir casos de prueba
  - Spring Boot incluye, automáticamente, frameworks para testing como Junit

# Contenido del proyecto generado

- Recursos del proyecto (resources)
  - Contiene un archivo de configuración vacío llamado *application.properties*. Sirve para proporcionar una configuración adicional al comportamiento de la app
  - También ha creado las subcarpetas *static* y *template* para los recursos Web.
    - Static, para ficheros CSS, imágenes...
    - Template, para los ficheros HTML

# Clase principal de SPRING BOOT

- src/.../DemoApplication.java

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```



# DemoApplication.java

- Spring Boot no te obliga a construir un WAR o un EAR y desplegarlo después en un servidor
- Spring Boot permite la ejecución como una app normal de Java usando un método main()
- Spring Boot se encarga, por ejemplo, de por defecto embeber el servidor Apache Tomcat al seleccionar la dependencia Spring Web

# @SpringBootApplication annotation.

```
@SpringBootApplication  
public class DemoApplication {
```

- Es una anotación que agrupa a otras 3:
  - @EnableAutoConfiguration, autoconfigura la app basándose en las dependencias JAR presentes en el *classpath*
  - @ComponentScan, escanea los componentes de Spring, desde el paquete raíz a todos sus hijos.
    - Componente es un Java *Bean* anotado con @Component o @Bean
  - @SpringBootConfiguration, permite que la configuración de la clase anotada se encontrada automáticamente por Spring Boot

# Ejecutando el proyecto

- Spring Boot Maven plugin goals.
  - *mvnw spring-boot:run*, ejecuta la app. Debe lanzarse desde el directorio dónde está el pom.xml
  - Matamos el proceso con ctrl-c.
  - Este Goal (*spring-boot:run*) es válido para Maven y Maven Wrapper:
    - *mvn spring-boot:run*
    - *mvnw spring-boot:run*

# Ejecutando el proyecto

- **mvnw** spring-boot:run

```
C:\Users\luiky\Downloads\demo>mvnw spring-boot:run
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:demo >-----
[INFO] Building demo 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> spring-boot-maven-plugin:2.7.4:run (default-cli) > test-compile @ demo >>>
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.10.1:compile (default-compile) @ demo ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\Users\luiky\Downloads\demo\target\classes
[INFO]
```

# Ejecutando el proyecto

```
[INFO] --- spring-boot-maven-plugin:2.7.4:run (default-cli) @ demo ---
[INFO] Attaching agents: []

  ____ _
 / ___ \ | |
/ /___\ \| |
\_____/___\___|
:: Spring Boot ::                (v2.7.4)

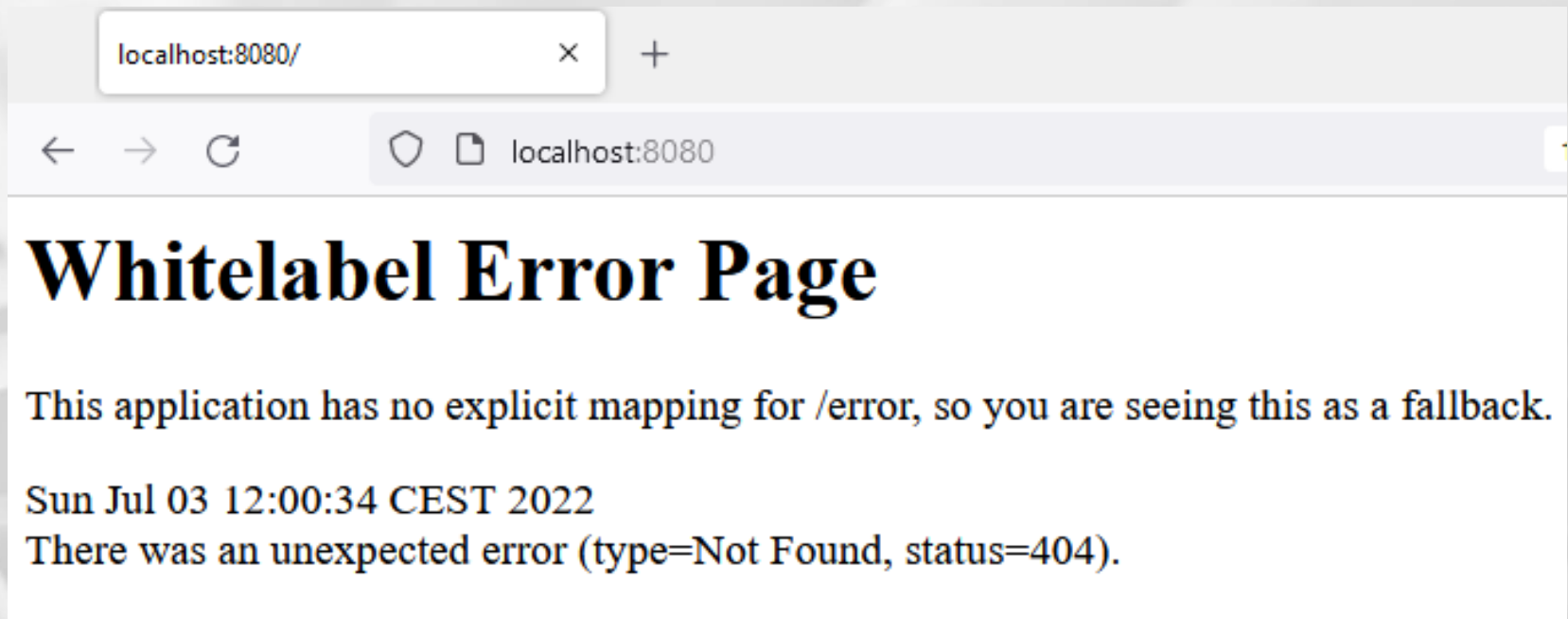
2022-10-03 11:38:40.304 INFO 7780 --- [main] com.example.demo.DemoApplication
n PID 7780 (C:\Users\luiky\Downloads\demo\target\classes started by luiky in C:\Users\luiky\Downl
2022-10-03 11:38:40.306 INFO 7780 --- [main] com.example.demo.DemoApplication

2022-10-03 11:38:41.059 INFO 7780 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer
2022-10-03 11:38:41.073 INFO 7780 --- [main] o.apache.catalina.core.StandardService
2022-10-03 11:38:41.073 INFO 7780 --- [main] org.apache.catalina.core.StandardEngine
2022-10-03 11:38:41.165 INFO 7780 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/]
2022-10-03 11:38:41.165 INFO 7780 --- [main] w.s.c.ServletWebServerApplicationContext
2022-10-03 11:38:41.437 INFO 7780 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer
2022-10-03 11:38:41.445 INFO 7780 --- [main] com.example.demo.DemoApplication
```

- Si todo ha ido bien: la aplicación se inicia y se ejecuta en el puerto HTTP 8080

# localhost:8080

- Página por defecto en localhost:8080



# Formato Logging Consola

- **Date and Time:** Precisión de milisegundo. Fácilmente ordenable.
- **Log Level:** ERROR, WARN, INFO, DEBUG, or TRACE.
  - ERROR indica un problema grave
  - INFO o DEBUG normalmente pueden ser ignorados
- **Process ID:** Identificador del proceso (un número)
- --- separador para distinguir el comienzo del mensaje de registro
- **Thread Name:** Nombre del hilo que está haciendo el logging. Entre corchetes [ ]
- **Logger name:** Suele ser el nombre de la clase (normalmente abreviado)
- **The log message:** La información real a registrar

# Ejemplo logging consola

```
2022-09-22 10:28:07.180 INFO 770 --- [main] o.s.b.d.f.s.MyApplication : Starting
2022-09-22 10:28:07.185 INFO 770 --- [main] o.s.b.d.f.s.MyApplication : No active
2022-09-22 10:28:11.996 INFO 770 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat
2022-09-22 10:28:12.031 INFO 770 --- [main] o.apache.catalina.core.StandardService : Starting
2022-09-22 10:28:12.031 INFO 770 --- [main] org.apache.catalina.core.StandardEngine : Starting
2022-09-22 10:28:12.191 INFO 770 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initial
2022-09-22 10:28:12.191 INFO 770 --- [main] w.s.c.ServletWebServerApplicationContext : Root Web
2022-09-22 10:28:13.660 INFO 770 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat
2022-09-22 10:28:13.675 INFO 770 --- [main] o.s.b.d.f.s.MyApplication : Started
```

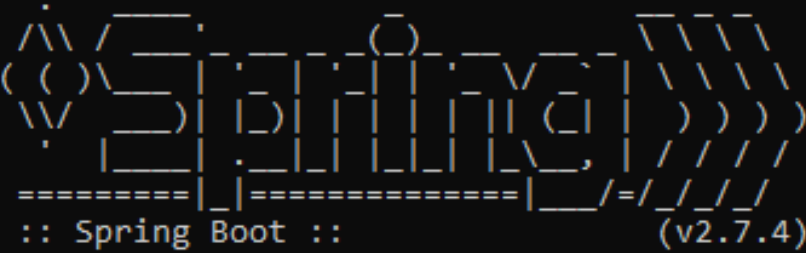
```
o.s.b.d.f.s.MyApplication : Starting MyApplication using Java 1.8.0_345 on myhost with PID 770
o.s.b.d.f.s.MyApplication : No active profile set, falling back to 1 default profile: "default"
o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
o.apache.catalina.core.StandardService : Starting service [Tomcat]
org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.65]
o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 4486 ms
o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
o.s.b.d.f.s.MyApplication : Started MyApplication in 9.458 seconds (JVM running for 12.129)
```



# Generación de un ejecutable JAR

- Desde el directorio que contiene el pom.xml:
  - mvnw package o mvn package
- Genera un .jar en la subcarpeta target, ejecutable con java -jar:
  - java -jar demo-0.0.1-SNAPSHOT.jar
  - Ctrl-c para terminar

```
C:\Users\luiky\Downloads\demo\target>java -jar demo-0.0.1-SNAPSHOT.jar
```



The image shows a terminal window with the Spring Boot logo. The logo consists of a stylized 'S' and 'B' made of dashed lines, with the text 'Spring Boot' in the center. Below the logo, it says 'v2.7.4'.

# Limpiando el proyecto

- Desde el directorio que contiene el pom.xml:
  - mvnw clean o mvn clean
- Elimina todos los archivos compilados.
  - Elimina el directorio target completamente
  - Elimina, por tanto, el .jar generado

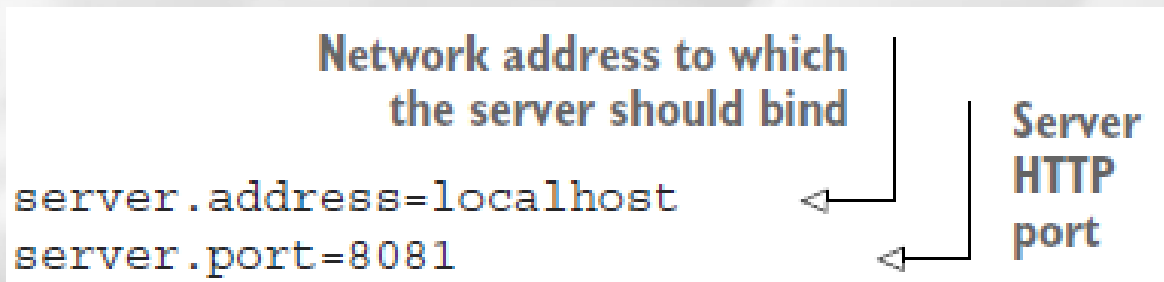
```
C:\Users\luiky\Downloads\demo>mvnw clean
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:demo >-----
[INFO] Building demo 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:3.2.0:clean (default-clean) @ demo ---
[INFO] Deleting C:\Users\luiky\Downloads\demo\target
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 0.352 s
[INFO] Finished at: 2022-10-11T13:35:57+02:00
[INFO] -----
```

# Administración de la configuración

- Spring Initializr genera un archivo ***application.properties*** vacío
- Se encuentra en *src/main/resources*
- Permite externalizar aspectos relacionados con la configuración de la aplicación
- Este archivo de propiedades permite especificar las configuraciones en un formato de pares clave-valor.
- La clave está separada del valor por el carácter =

# Fichero de propiedades

- Ejemplo de *application.properties*



Network address to which  
the server should bind

server.address=localhost

server.port=8081

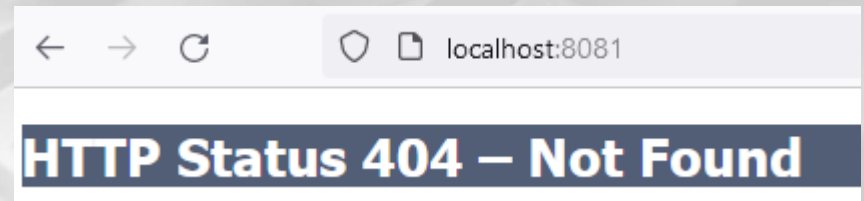
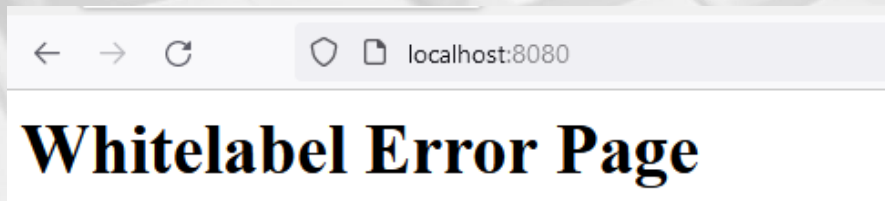
Server  
HTTP  
port

The diagram shows two lines of code from an application.properties file. The first line, 'server.address=localhost', is annotated with a bracket and the text 'Network address to which the server should bind'. The second line, 'server.port=8081', is annotated with a bracket and the text 'Server HTTP port'.

- Propiedades comunes de una aplicación de Spring
  - [application-properties.html](http://application-properties.html)

# Practicando

- Ejecutar dos aplicaciones web de Spring boot al mismo tiempo. Una en el puerto 8080 y otra en el 8081.
- La app-web del puerto 8081 no mostrará la *whitelabel error page*



# Recursos

- Maven
  - <https://maven.apache.org/>
- JDK
  - <https://www.oracle.com/java/technologies/downloads/>
- Spring Initializr:
  - <https://start.spring.io>
- Common application properties
  - <application-properties.html>