

Spring Framework

Introducción a Spring Boot



EE. SS. M^a AUXILIADORA

2º DAM

Autor: Manuel Torres Molina

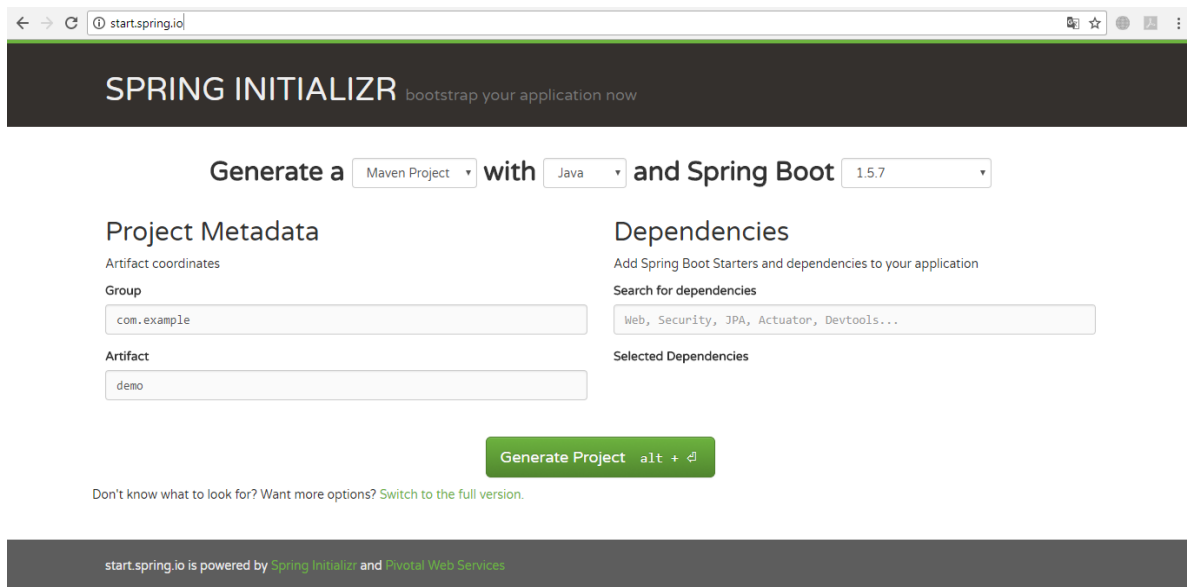
ACCESO A DATOS

Spring Framework

Introducción a Spring Boot

Nuestra primera aplicación Spring Boot

Podemos crear un proyecto inicial desde la web start.spring.io.



The screenshot shows the Spring Initializr web application. At the top, it says "SPRING INITIALIZR bootstrap your application now". Below this, there are dropdown menus to "Generate a" (Maven Project), "with" (Java), and "and Spring Boot" (1.5.7). There are two main sections: "Project Metadata" and "Dependencies". Under "Project Metadata", there are input fields for "Group" (com.example) and "Artifact" (demo). Under "Dependencies", there is a search bar with the text "Web, Security, JPA, Actuator, Devtools..." and a "Selected Dependencies" section. A green button labeled "Generate Project" with a keyboard shortcut "alt + ⌘" is at the bottom. A footer note says "start.spring.io is powered by Spring Initializr and Pivotal Web Services".

Las dependencias que vamos a utilizar de primeras en la creación de este primer proyecto serían Web y Thymeleaf, el resto de dependencias las iremos añadiendo a lo largo del desarrollo de nuestro proyecto el fichero pom.xml según vayamos requiriendo más funcionalidades.

El proyecto generado podremos llevarlo a nuestro workspace de eclipse en una carpeta descomprimida.

Para actualizar nuestro proyecto con las dependencias sería poner a través del terminal de consola dentro de la carpeta del proyecto que está en el workspace la orden:

mvn clean install

Maven

MAVEN es una herramienta de software que nos permite la gestión integral de nuestro proyecto en cualquiera de sus fases.

Los proyectos Maven son reusables. Su ciclo de vida tendría las siguientes fases:

- Compile
- Test
- Package
- Install
- Deploy

La configuración de nuestro proyecto mediante Maven vendrá definida por el fichero pom.xml.

A continuación, vemos una estructura de este fichero.

```
<?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.udemy</groupId>
  <artifactId>backendninja</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>backendninja</name>
  <description>Proyecto del curso</description>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.5.6.RELEASE</version>
    <relativePath /> <!-- lookup parent from repository -->
  </parent>

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

  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>
    <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>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
  </dependencies>
</project>
```

```

        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
        </dependency>
        <dependency>
            <groupId>com.querydsl</groupId>
            <artifactId>querydsl-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-log4j12</artifactId>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
            <plugin>
                <groupId>com.mysema.maven</groupId>
                <artifactId>apt-maven-plugin</artifactId>
                <version>1.1.3</version>
                <executions>
                    <execution>
                        <goals>
                            <goal>process</goal>
                        </goals>
                        <configuration>
                            <outputDirectory>target/generated-
sources/java</outputDirectory>

                            <processor>com.querydsl.apt.jpa.JPAAnnotationProcessor</processor>
                            </configuration>
                        </execution>
                    </executions>
                    <dependencies>
                        <dependency>
                            <groupId>com.querydsl</groupId>
                            <artifactId>querydsl-apt</artifactId>
                            <version>${querydsl.version}</version>
                        </dependency>
                    </dependencies>
                </plugin>
            </plugins>
        </build>
    </project>

```

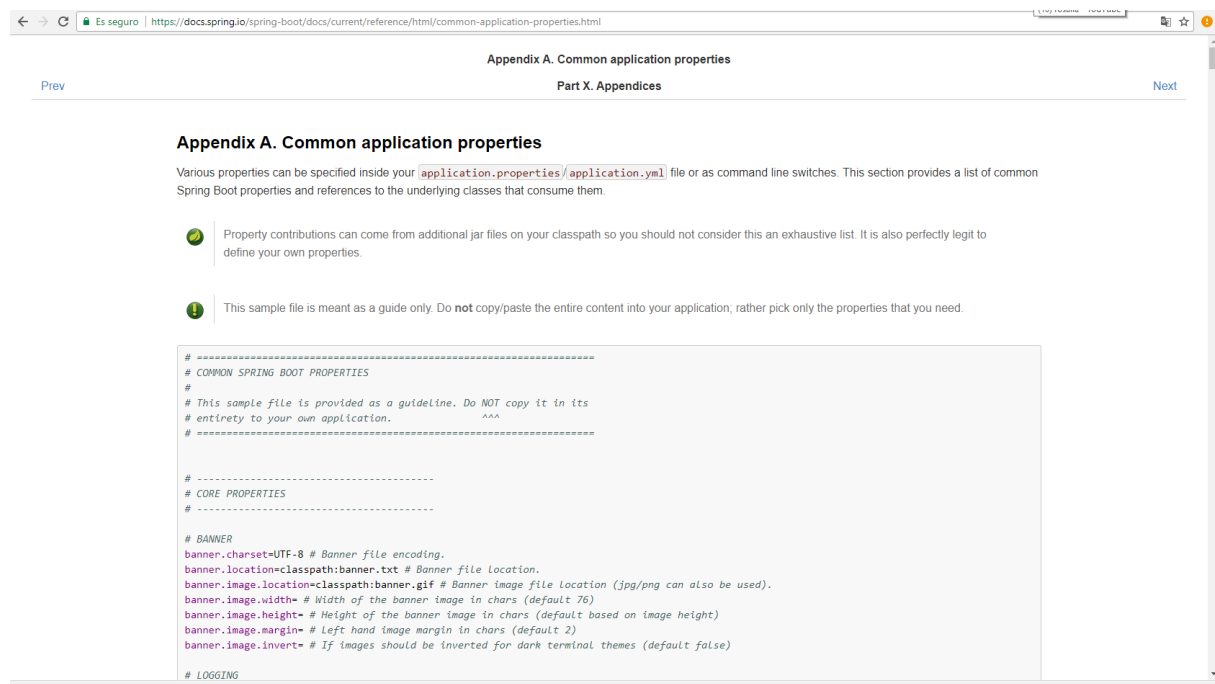
Bastaría si quisiéramos añadir alguna funcionalidad o librería, irnos a la página del repositorio de Maven mvnrepository.com y buscar que funcionalidad queremos y añadir el código de dependencia en el fichero pom.xml y realizar al igual que antes un ***mvn clean install*** desde un terminal.

Introducción a Spring Boot

Para añadir propiedades de la configuración de nuestro proyecto Spring Boot, tendremos que irnos al fichero `application.properties` y añadir las que consideremos necesarias.

Para consultar todas estas propiedades tendremos que irnos a la siguiente dirección:

<https://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html>



Estructura de un proyecto Spring Boot

- **Spring Elements**. Se almacenarán todos los Beans declarados.
- **src/main/java**. Contendrá todas las clases java que declaremos con sus correspondientes paquetes. Estarán los controladores, servicios y repositorios entre otros. También contendrá la clase `BackendninjaApplication.java` que servirá para arrancar la aplicación Spring Boot.
- **src/main/resources**. Están todos los recursos de nuestra aplicación (vistas, imágenes, estilos, etc.) y nuestro fichero `application.properties`.
- **src/test/java**. La misma estructura que las de nuestra carpeta de clases java.
- **JRE System Library**. Librerías utilizadas.
- **Maven Dependencies**. Todas nuestras dependencias del proyecto.
- **src**. La estructura que tiene en nuestro Workspace de carpetas y archivos.
- **target**. Todos los recursos de Maven que genera en su ciclo de vida.
- Quedaría unos archivos de configuración de Maven y el fichero `pom.xml`

Transformar un fichero .properties a un .yaml

Vamos a transformar el fichero de configuración application.properties en uno application.yml.

La estructura de los dos quedaría de la siguiente manera:

Fichero application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/mydb?useSSL=false
spring.datasource.username=root
spring.datasource.password=
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.hibernate.naming.strategy= org.hibernate.cfg.ImprovedNamingStrategy
spring.jpa.properties.hibernate.dialect= org.hibernate.dialect.MySQL5Dialect
```

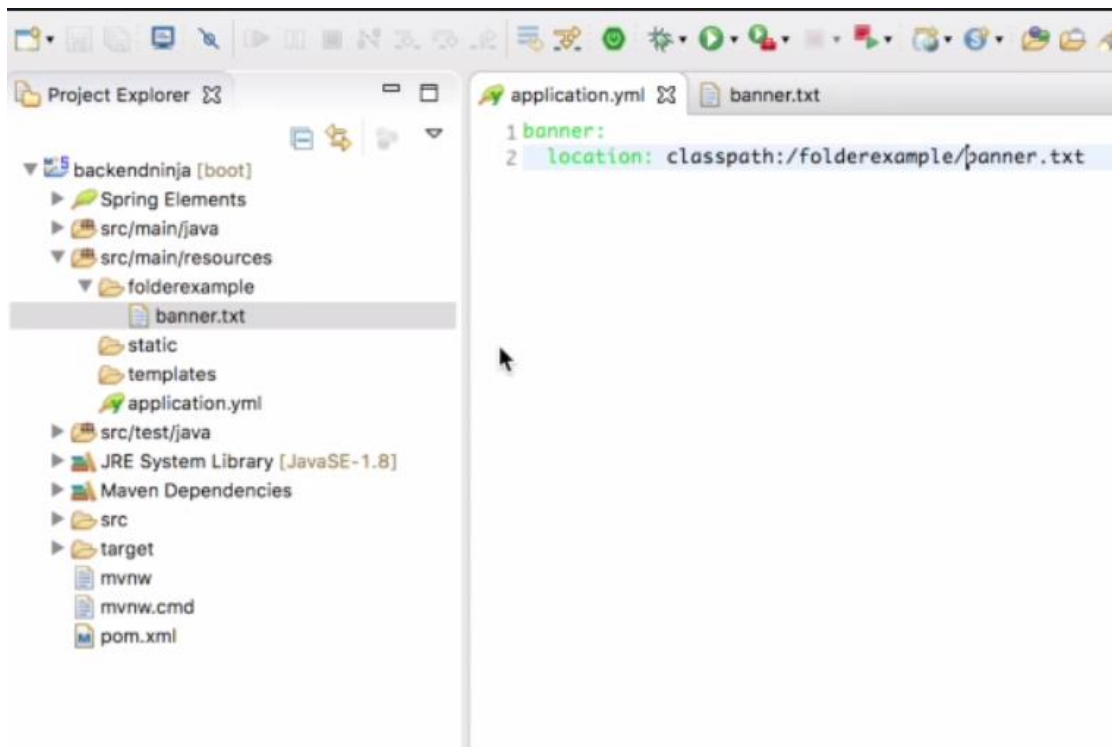
Fichero application.yml

```
spring:
  datasource:
    url: jdbc:mysql://localhost:3306/mydb?useSSL=false
    username: root
    password:
  jpa:
    show-sql: true
    hibernate:
      ddl-auto: update
      naming:
        strategy: org.hibernate.cfg.ImprovedNamingStrategy
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQL5Dialect
```

Cambiar el Banner de la aplicación Spring Boot

Se puede cambiar creando un fichero .txt en el paquete **src/main/resources** y le pondríamos el nombre banner.txt. Podemos modificar las propiedades de este banner en el **appendix A**.

Por ejemplo, si queremos poner el fichero del banner en otro paquete que no sea el anterior por defecto. Habría que añadir estas propiedades en el fichero **application.yml**.



Existen páginas para crear banner personalizados que generen imágenes o textos con caracteres:

<https://spring-boot-banner-gen.cfapps.io/banner>

<http://patorjk.com/software/taag/#p=display&f=Graffiti&t=backend%20oninja!>