

¿Qué es un Framework?

Es un conjunto estructurado de herramientas, librerías, buenas prácticas y componentes reutilizables que proporcionan una base para desarrollar aplicaciones de manera más rápida, organizada y estandarizada.

En lugar de empezar un proyecto desde cero, el framework ya ofrece piezas listas (como manejo de base de datos, seguridad, enrutamiento, plantillas de interfaz, etc.), lo que permite al desarrollador enfocarse en la lógica de negocio y no en reinventar funciones básicas

¿Qué es Spring Framework?

Es un framework de código abierto para Java que facilita el desarrollo de aplicaciones empresariales y web modernas, seguras y escalables. Se basa en principios como la Inversión de Control (IoC) y la Inyección de Dependencias (DI), lo que reduce el acoplamiento y mejora la mantenibilidad del código. Además, incluye módulos como Spring MVC para crear APIs REST, Spring Data para acceder a bases de datos SQL y NoSQL, Spring Security para manejar autenticación y autorización, y Spring Boot, que simplifica la configuración y permite levantar proyectos rápidamente. Gracias a su ecosistema (Spring Cloud, Spring Batch, entre otros), se ha convertido en un estándar para construir desde aplicaciones monolíticas hasta arquitecturas de microservicios en la nube.

¿Qué es Spring Boot?

es un proyecto dentro del ecosistema de Spring que simplifica la creación y ejecución de aplicaciones en Java, eliminando gran parte de la configuración manual que requería el Spring Framework tradicional. Su principal ventaja es la auto-configuración, que detecta automáticamente las dependencias y ajusta la aplicación para que funcione sin necesidad de archivos XML extensos. Además, incluye un servidor embebido (como Tomcat o Jetty), lo que permite ejecutar la aplicación como un simple `java -jar` sin necesidad de desplegar en servidores externos. También ofrece herramientas como Spring Boot Starter (paquetes preconfigurados de dependencias), Spring Boot Actuator (para monitoreo y métricas en producción) y compatibilidad con Spring Initializr (para generar proyectos rápidamente)

Características de Spring Boot



Made with Napkin

¿Qué es Spring Initializr?

Spring Initializr es una herramienta oficial de Spring que permite generar proyectos Spring Boot de manera rápida y personalizada.

Funciona como un asistente en línea (<https://start.spring.io/>) o integrado en IDEs como IntelliJ, Eclipse o NetBeans, donde selecciona:

- Versión de Spring Boot.
- Lenguaje (Java, Kotlin, Groovy).
- Gestor de dependencias (Maven o Gradle).
- Versión de Java.
- Dependencias necesarias (Web, JPA, MongoDB, Security, Lombok, etc.).

Con esos datos, Initializr genera un proyecto base listo para importar en tu IDE y comenzar a programar sin tener que crear desde cero el pom.xml o la estructura de carpetas.

start.spring.io

3. Basic Data Struct... 1. Introduction — P... Mi perfil Activate Adobe Acrobat

spring initializr

Project

☐ Gradle - Groovy

☐ Gradle - Kotlin

☒ Maven

Language

☒ Java

☐ Kotlin

☐ Groovy

Spring Boot

☐ 4.0.0 (SNAPSHOT)

☐ 4.0.0 (M2)

☐ 3.5.6 (SNAPSHOT)

☒ 3.5.5

☐ 3.4.10 (SNAPSHOT)

☐ 3.4.9

Project Metadata

Group

com.example

Artifact

demo

Name

demo

Description

Demo project for Spring Boot

Package name

com.example.demo

Packaging

☒ Jar

☐ War

Java

☐ 24

☒ 21

☐ 17

Dependencies

ADD DEPENDENCIES... CTRL + B

No dependency selected

GENERATE CTRL + G

EXPLORE CTRL + SPACE

...

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Boot DevTools

DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Spring Data MongoDB

NOSQL

Store data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time.

Lombok

DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

Spring Web

WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

```

1 // Definimos el paquete donde se encuentra la clase.
2 // Esto organiza el código dentro de la aplicación.
3 package com.DASistemas.DASistemas.controller;
4
5 // Importamos las anotaciones necesarias de Spring Framework
6 import org.springframework.web.bind.annotation.GetMapping;
7 import org.springframework.web.bind.annotation.RestController;
8
9 /*
10  * La anotación @RestController indica que esta clase es un controlador
11  * de tipo REST en Spring Boot. Esto significa que va a manejar peticiones HTTP
12  * y devolverá datos directamente en formato JSON, texto u otro tipo de respuesta.
13  */
14 @RestController
15 public class HolaController {
16
17     /*
18     * La anotación @GetMapping("/hola") le dice a Spring que este método
19     * responderá a las solicitudes HTTP GET que lleguen a la URL "/hola".
20     * Ejemplo: si el servidor está en http://localhost:8080,
21     * la ruta sería http://localhost:8080/hola
22     */
23     @GetMapping("/hola")
24     public String holamundo() {
25         /*
26         * Este método devuelve un String como respuesta al cliente.
27         * En este caso, simplemente retorna el mensaje:
28         * "Hola mundo desde Spring Boot".
29         */
30         return "Hola mundo desde Spring Boot";
31     }
32 }
33
34

```

Ejercicio: Aplicación de gestión de estudiantes

La universidad desea construir un sistema sencillo en Spring Boot para manejar la información de estudiantes. Por ahora, se requiere únicamente consultar la lista de estudiantes registrados en una base de datos MongoDB.

Cada estudiante debe tener la siguiente información:

- id (identificador único)
- nombre completo
- email institucional
- programa académico
- semestre actual

Requisitos:

- Deben implementar una entidad que represente al estudiante.

- Deben crear un repositorio para acceder a los datos en MongoDB.
- Deben desarrollar un servicio que gestione la lógica de negocio.
- Deben construir un controlador que exponga un endpoint para consultar todos los estudiantes.

Objetivo del ejercicio

Organizar el proyecto de manera que cada componente (entidad, repositorio, servicio y controlador) tenga una sola responsabilidad, demostrando así la aplicación del primer principio SOLID: Responsabilidad Única.