SPRING DATA JPA

El objetivo de Spring es que cuando se escriba una aplicación, el desarrollador se enfoque únicamente en la lógica del negocio y no en el código repetitivo o que se puede generar de forma automática.

JPA es la Java Persistence API, que es la forma estandarizada que ofrece Java para usar los proveedores de ORM (Object Relational Mapping), que simplifican el trabajo con bases de datos. Spring Data JPA es el módulo de Spring que permite trabajar con JPA en nuestras aplicaciones, encapsulando toda la complejidad del acceso a datos. JPA maneja la mayor parte de la complejidad del acceso a la base de datos basada en JDBC y las asignaciones objeto-relacionales.

Spring Data JPA aprovecha las anotaciones de JPA para definir las entidades (la información que se guardará en base de datos) y las relaciones entre ellos, así como también información particular del mecanismo de persistencia (nombres de tablas, longitudes de campos, tipos de datos, etc.), los cuales podemos personalizar si tenemos alguna necesidad particular y los valores por default no nos funcionan.

JPA es una especificación que define una API para mapeos objeto-relacionales y para gestionar objetos persistentes. Hibernate y EclipseLink son dos implementaciones populares de esta especificación.

Spring Data JPA se integra perfectamente con Spring Boot, el cual se encarga de incializar los repositorios e indicar el proveedor de persistencia (por default Hibernate). Hibernate es una implementación del estándar Java Persistence API (JPA). Actúa como una capa adicional sobre JDBC y permite implementar una capa de persistencia independiente de la base de datos. Hibernate proporciona una implementación de mapeo objeto-relacional que mapea los registros de la base de datos a objetos Java y genera las sentencias SQL necesarias para replicar todas las operaciones a la base de datos.

Spring Data JPA añade una capa sobre JPA. Esto significa que utiliza todas las características definidas por la especificación JPA, especialmente los mapeos de entidades y asociaciones, la gestión del ciclo de vida de las entidades y las capacidades de consulta de JPA. Además, Spring Data JPA añade sus propias características, como una implementación sin código del patrón de repositorio y la creación de consultas a la base de datos a partir de nombres de métodos.

Algunas veces necesitamos inicializar o hacer alguna comprobación al momento de iniciar una aplicación. Para estos casos existe la interface CommandLineRunner, la cual indica que una clase se debe ejecutar después de inicializar los componentes de la aplicación pero antes de comenzar a recibir peticiones.

Las interfaces Application Runner y Command Line Runner permiten ejecutar el código una vez iniciada la aplicación Spring Boot. Puedes utilizar estas interfaces para realizar cualquier acción inmediatamente después de que la aplicación se haya iniciado.