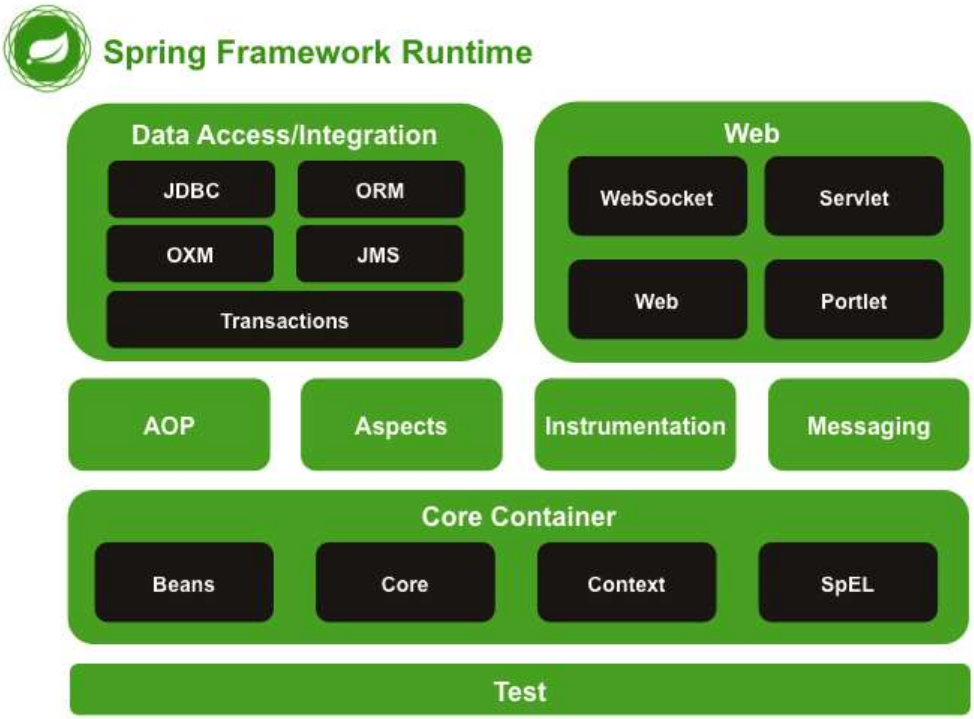
DESARROLLO WEB CON SPRING



UNIDAD 07
SPRING DATA JPA

Eric Gustavo Coronel Castillo
gcoronelc.github.io
I N S T R U C T O R

CONTENIDO

INTRODUCCIÓN **3**

¿Qué es Spring Data JPA? 3
¿JPA y Spring Data JPA son iguales? 3
¿Por qué se desarrolló Spring Data JPA? 4

INICIAR PROYECTO **5**

Creación del proyecto 5
Dependencias 5
Parámetros de Conexión 6

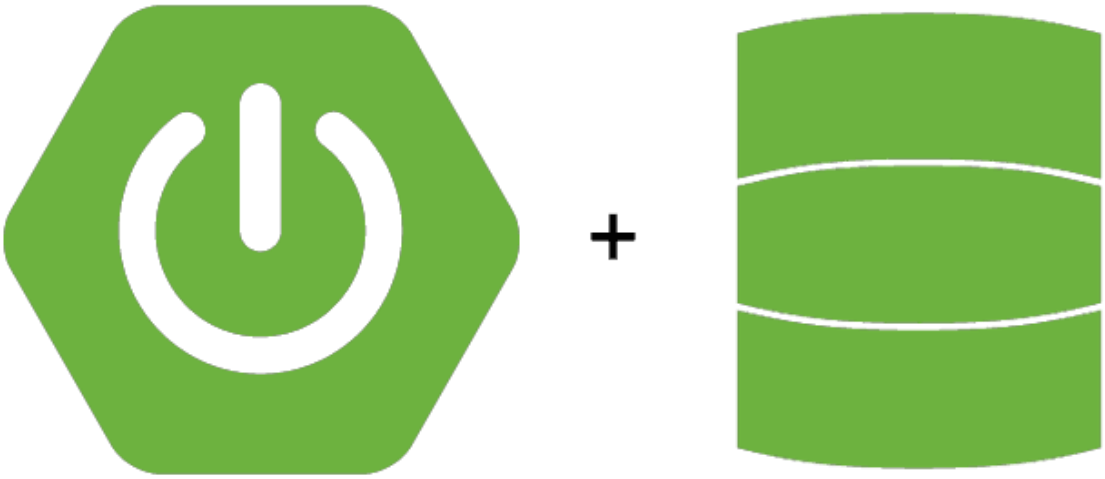
EJEMPLO ILUSTRATIVO **7**

Clase Entidad 7
Capa Repository 8
Capa Service 9
Interface *9*
Implementación *9*
Prueba 11

CURSOS VIRTUALES **12**

CUPONES 12
FUNDAMENTOS DE PROGRAMACIÓN 12
JAVA ORIENTADO A OBJETOS 13
PROGRAMACIÓN CON JAVA JDBC 14
PROGRAMACIÓN CON ORACLE PL/SQL 15

INTRODUCCIÓN



¿Qué es Spring Data JPA?

Spring Data JPA es una librería que forma parte de la familia Spring Data. Spring Data JPA facilita la implementación de repositorios basados en JPA. Este marco tiene soporte mejorado para capas de acceso a datos basadas en JPA.

Spring Data JPA le ayuda a centrarse en la lógica empresarial en lugar de la complejidad técnica y el código estándar.

¿JPA y Spring Data JPA son iguales?

La respuesta es no.

JPA es una especificación que estandariza la forma en que los objetos Java se asignan a un sistema de base de datos relacional. Al ser solo una especificación, JPA consta de un conjunto de interfaces, como EntityManagerFactory, EntityManager y anotaciones que lo ayudan a asignar un objeto de entidad Java a una tabla de base de datos.

Hay varios proveedores de JPA, como HIbernate, EclipseLink u Open JPA, que puede utilizar.

Spring Data JPA es una abstracción de acceso a datos JPA. Al igual que JPA, Spring Data JPA no puede funcionar sin un proveedor de JPA. Genera consultas JPA en su nombre a través de convenciones de nombres de métodos.

JPA maneja la mayor parte de la complejidad del acceso a la base de datos basada en JDBC y los ORM (asignaciones relacionales de objetos). Además de eso, Spring Data JPA reduce la cantidad de código repetitivo requerido por JPA.

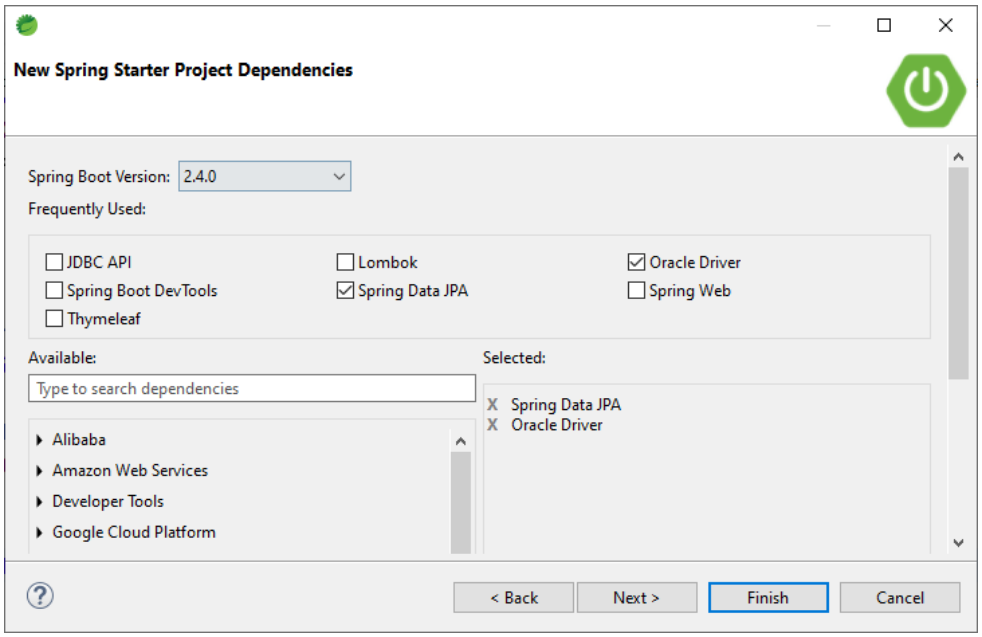
¿Por qué se desarrolló Spring Data JPA?

Implementar una capa de acceso a datos de una aplicación usando JPA o JDBC es engorroso. Demasiado código repetitivo para ejecutar consultas simples, realizar paginación y auditoría.

Para evitar todo esto, se desarrolló Spring Data JPA. Spring Data JPA mejora la implementación de capas de acceso a datos al reducir el esfuerzo que realmente se necesita.

INICIAR PROYECTO

Creación del proyecto



Dependencias

```
<?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>2.4.0</version>
      <relativePath/> <!-- lookup parent from repository -->
   </parent>
   <groupId>com.desarrollasoftware</groupId>
   <artifactId>JPAConsole</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <name>JPAConsole</name>
   <description>Modelo de una aplicacion</description>

   <properties>
      <java.version>11</java.version>
   </properties>
```

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

    <dependency>
      <groupId>com.oracle.database.jdbc</groupId>
      <artifactId>ojdbc8</artifactId>
      <scope>runtime</scope>
    </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>
```

Parámetros de Conexión

Estos parámetros se deben incluir en el archivo de propiedades (application.properties), también puedes establecer el puerto del servidor.

```
# Oracle settings
spring.datasource.url=jdbc:oracle:thin:@localhost:1521/XE
spring.datasource.username=eureka
spring.datasource.password=admin
spring.jpa.database-platform=org.hibernate.dialect.Oracle12cDialect
spring.jpa.hibernate.ddl-auto=none
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

EJEMPLO ILUSTRATIVO

Clase Entidad

```
package com.desarrollasoftware.app.entity;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "CLIENTE")
public class Cliente implements Serializable{

   private static final long serialVersionUID = 1L;

   @Id
   @Column(name = "CHR_CLIECODIGO")
   private String codigo;

   @Column(name = "VCH_CLIEPATERNO")
   private String paterno;

   @Column(name = "VCH_CLIEMATERNO")
   private String materno;

   @Column(name = "VCH_CLIENOMBRE")
   private String nombre;

   @Column(name = "CHR_CLIEDNI")
   private String dni;

   @Column(name = "VCH_CLIECIUDAD")
   private String ciudad;

   @Column(name = "VCH_CLIEDIRECCION")
   private String direccion;

   @Column(name = "VCH_CLIETELEFONO")
   private String telefono;

   @Column(name = "VCH_CLIEEMAIL")
```

```
   private String email;

   public Cliente() {
   }

   // Métodos getters y setters

   @Override
   public String toString() {
      return "Cliente [codigo=" + codigo + ", paterno=" + paterno
         + ", materno=" + materno + ", nombre=" + nombre
         + ", dni=" + dni + ", ciudad=" + ciudad + ", direccion="
         + direccion + ", telefono=" + telefono
         + ", email=" + email + "]";
   }

}
```

Capa Repository

```
package com.desarrollasoftware.app.repository;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

import com.desarrollasoftware.app.entity.Cliente;

@Repository
public interface ClienteRepository extends CrudRepository<Cliente, String>{

}
```

Capa Service

Interface

```
package com.desarrollasoftware.app.service;

import java.util.List;

import com.desarrollasoftware.app.entity.Cliente;

public interface ClienteService {

   List<Cliente> listarTodos();
   Cliente buscarPorId(String id);
   void grabar(Cliente cliente);
   void eliminar(String id);

}
```

Implementación

```
package com.desarrollasoftware.app.service.impl;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.desarrollasoftware.app.entity.Cliente;
import com.desarrollasoftware.app.repository.ClienteRepository;
import com.desarrollasoftware.app.service.ClienteService;

@Service
public class ClienteServiceImpl implements ClienteService{

   @Autowired
   private ClienteRepository repository;

   @Override
   public List<Cliente> listarTodos() {
      List<Cliente> lista = (List<Cliente>) repository.findAll();
      return lista;
   }
```

```
    @Override
    public Cliente buscarPorId(String id) {
        Cliente cliente = repository.findById(id).orElse(null);
        return cliente;
    }

    @Override
    public void grabar(Cliente cliente) {
        repository.save(cliente);
    }

    @Override
    public void eliminar(String id) {
        repository.deleteById(id);
    }

}
```

Prueba

```
package com.desarrollasoftware.app;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

import com.desarrollasoftware.app.entity.Cliente;
import com.desarrollasoftware.app.service.ClienteService;

@SpringBootApplication
public class JpaConsoleApplication implements CommandLineRunner {

   @Autowired
   private ClienteService service;

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

   @Override
   public void run(String... args) throws Exception {
      List<Cliente> lista = service.listarTodos();
      for (Cliente p : lista) {
         System.out.println(p.getNombre());
      }
   }

}
```

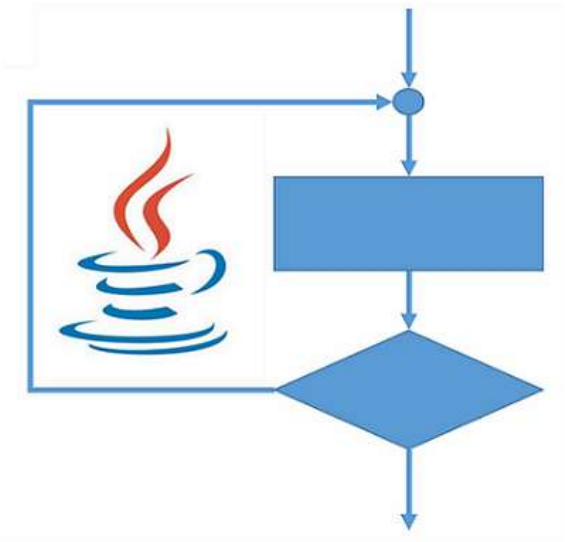
CURSOS VIRTUALES

CUPONES

En esta URL se publican cupones de descuento:

http://gcoronelc.github.io

FUNDAMENTOS DE PROGRAMACIÓN



Tener bases sólidas de programación muchas veces no es fácil, creo que es principalmente por que en algún momento de tu aprendizaje mezclas la entrada de datos con el proceso de los mismos, o mezclas el proceso con la salida o reporte, esto te lleva a utilizar malas prácticas de programación que luego te serán muy difíciles de superar.

En este curso aprenderás las mejores practicas de programación para que te inicies con éxito en este competitivo mundo del desarrollo de software.

URL del Curso: **https://www.udemy.com/course/fund-java**

Avance del curso: **https://n9.cl/gcoronelc-fp-avance**

Cupones de descuento: **http://gcoronelc.github.io**

JAVA ORIENTADO A OBJETOS

En este curso aprenderás a crear software aplicando la Orientación a objetos, la programación en capas, el uso de patrones de software y swing.

Cada tema está desarrollado con ejemplos que demuestran los conceptos teóricos y finalizan con un proyecto aplicativo.

URL del Curso: **https://bit.ly/2B3ixUW**

Avance del curso: **https://bit.ly/2RYGXIt**

Cupones de descuento: **http://gcoronelc.github.io**

PROGRAMACIÓN CON JAVA JDBC

En este curso aprenderás a programas bases de datos Oracle con JDBC utilizando los objetos Statement, PreparedStatement, CallableStatement y a programar transacciones correctamente teniendo en cuenta su rendimiento y concurrencia.

Al final del curso se integra todo lo desarrollado en una aplicación de escritorio.

URL del Curso: **https://bit.ly/31apy0O**

Avance del curso: **https://bit.ly/2vatZOT**

Cupones de descuento: **http://gcoronelc.github.io**

PROGRAMACIÓN CON ORACLE PL/SQL

En este curso aprenderás a programas las bases de datos ORACLE con PL/SQL, de esta manera estarás aprovechando las ventas que brinda este motor de base de datos y mejoraras el rendimiento de tus consultas, transacciones y la concurrencia.

Los procedimientos almacenados que desarrolles con PL/SQL se pueden ejecutarlo de Java, C#, PHP y otros lenguajes de programación.

URL del Curso: **https://bit.ly/2YZjfxT**

Avance del curso: **https://bit.ly/3bciqYb**

Cupones de descuento: **http://gcoronelc.github.io**