Programación Frontend y Backend

BLOQUE SPRING

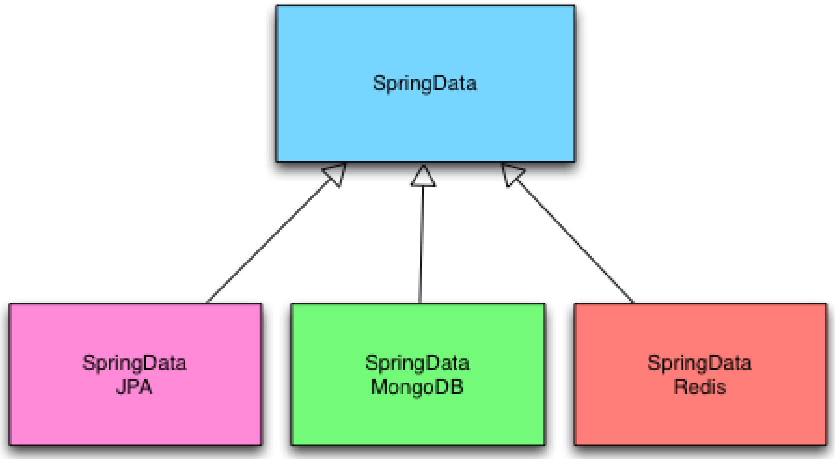
Spring DATA

01 Introducción

Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro

Spring Data

Spring Data es uno de los frameworks que se encuentra dentro de la plataforma de Spring. Su objetivo es simplificar al desarrollador la persistencia de datos contra distintos repositorios de información.



GOBIERNO DE ESPAÑA MINISTERIO DE INDUSTRIA, COMERCIO Y TURISMO
Escuela de organización industrial
Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro
SISTEMA NACIONAL DE GARANTÍA JUVENIL

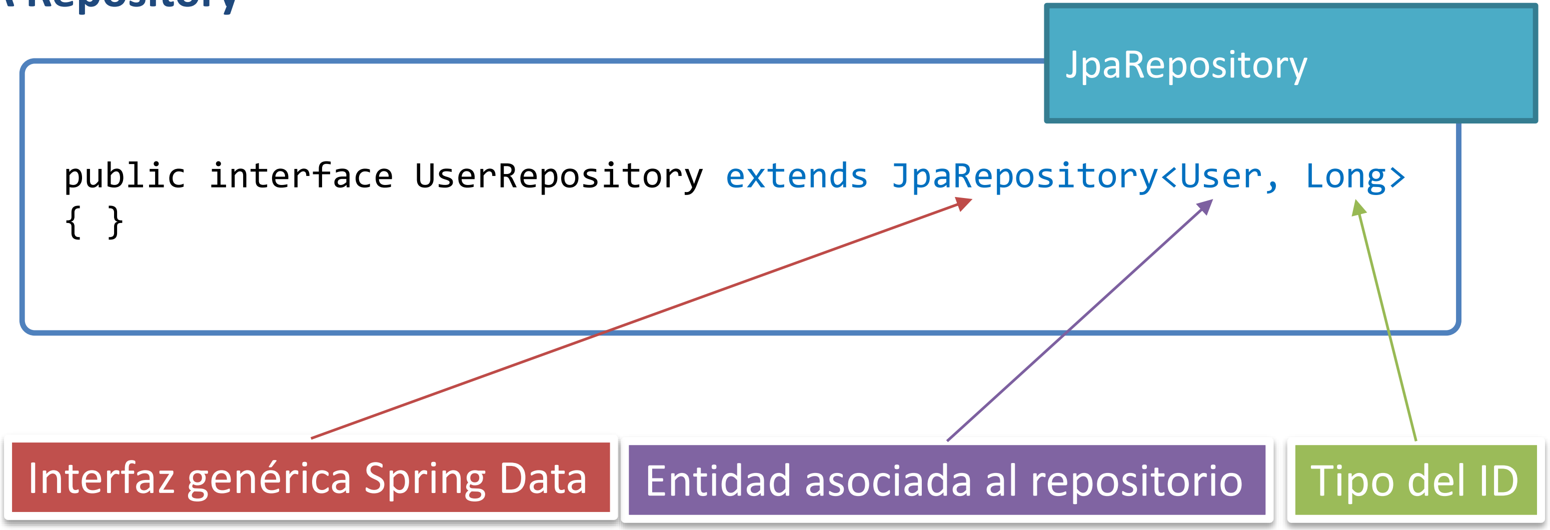
JPA Repository

Permite la definición de objetos de acceso a datos simplemente especificando una interfaz que proporciona:

- Métodos CRUD genéricos (**C**reate-**R**ead-**U**pdate-**D**elete).
- Métodos de consulta a partir de su nomenclatura.
- Métodos de consulta a partir de queries JPQL o mediante queries con nombre.

Todo esto **sin que sea necesario desarrollar la implementación**.
<u>Spring lo hace automáticamente</u> a partir de la interfaz.

JPA Repository



```
public interface UserRepository extends JpaRepository<User, Long>
{ }
```

JpaRepository
Interfaz genérica Spring Data
Entidad asociada al repositorio
Tipo del ID

Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro

JPA Repository

Method Summary	
long	**count**() Returns the number of entities available.
void	**delete**(ID id) Deletes the entity with the given id.
void	**delete**(Iterable<? extends T> entities) Deletes the given entities.
void	**delete**(T entity) Deletes a given entity.
void	**deleteAll**() Deletes all entities managed by the repository.
void	**deleteInBatch**(Iterable<T> entities) Deletes the given entities in a batch which means it will create a single Query.
boolean	**exists**(ID id) Returns whether an entity with the given id exists.
List<T>	**findAll**() Returns all instances of the type.
Page<T>	**findAll**(Pageable pageable) Returns a Page of entities meeting the paging restriction provided in the Pageableobject.
List<T>	**findAll**(Sort sort) Returns all entities sorted by the given options.
T	**findOne**(ID id) Retrives an entity by its primary key.
void	**flush**() Flushes all pending changes to the database.
List<T>	**save**(Iterable<? extends T> entities) Saves all given entities.
T	**save**(T entity) Saves a given entity.
T	**saveAndFlush**(T entity) Saves an entity and flushes changes instantly.

JPA Repository

A tener en cuenta:

Prefijos: find…By, read…By, query…By, count…By y get…By

Añadir antes del primer By:

First: Devuelve solo el primer elemento de la lista. findFirstBy

Top + Número: Devuelve el número de elementos indicado: findTop3By

Distinct: Nos permite seleccionar un único resultado. findTitleDistinctBy

Spring Data

Introducción

Keyword	Sample	JPQL snippet
And	findByLastnameAndFirstname	… where x.lastname = ?1 and x.firstname = ?2
Or	findByLastnameOrFirstname	… where x.lastname = ?1 or x.firstname = ?2
Is,Equals	findByFirstname,findByFirstnameIs ,findByFirstnameEquals	… where x.firstname = ?1
Between	findByStartDateBetween	… where x.startDate between ?1 and ?2
LessThan	findByAgeLessThan	… where x.age < ?1
LessThanEqual	findByAgeLessThanEqual	… where x.age <= ?1
GreaterThan	findByAgeGreaterThan	… where x.age > ?1
GreaterThanEqual	findByAgeGreaterThanEqual	… where x.age >= ?1
After	findByStartDateAfter	… where x.startDate > ?1
Before	findByStartDateBefore	… where x.startDate < ?1
IsNull	findByAgeIsNull	… where x.age is null
IsNotNull,NotNull	findByAge(Is)NotNull	… where x.age not null
Like	findByFirstnameLike	… where x.firstname like ?1
NotLike	findByFirstnameNotLike	… where x.firstname not like ?1

Spring Data

Introducción

StartingWith	findByFirstnameStartingWith	… where x.firstname like ?1(parameter bound with appended %)
EndingWith	findByFirstnameEndingWith	… where x.firstname like ?1(parameter bound with prepended %)
Containing	findByFirstnameContaining	… where x.firstname like ?1(parameter bound wrapped in %)
OrderBy	findByAgeOrderByLastnameDesc	… where x.age = ?1 order by x.lastname desc
Not	findByLastnameNot	… where x.lastname <> ?1
In	findByAgeIn(Collection<Age> ages)	… where x.age in ?1
NotIn	findByAgeNotIn(Collection<Age> age)	… where x.age not in ?1
True	findByActiveTrue()	… where x.active = true
False	findByActiveFalse()	… where x.active = false
IgnoreCase	findByFirstnameIgnoreCase	… where UPPER(x.firstame) = UPPER(?1)

¡Pregunta! ¿qué hace cada query?:

public Todo findById(Long id);

public List<Todo> findByTitleOrDescription(String title, String description);

public long countByTitle(String title);

public List<Todo> findDistinctByTitle(String title);

public List<Todo> findFirstByTitleOrderByTitleAsc(String title);

public List<Todo> findTop3ByTitleOrderByTitleAsc(String title);

Spring Data

@Query

```
public interface UserRepository extends JpaRepository<User, Long> {
  @Query("select u from User u where u.emailAddress = ?")
  User findByEmailAddress(String emailAddress);
}
```

@Query y @Param

```
public interface UserRepository extends JpaRepository<User, Long> {
  @Query("select u from User u
          where u.firstname = :firstname or u.lastname = :lastname")
  User findByLastnameOrFirstname(@Param("lastname") String lastname,
                                 @Param("firstname") String firstname);
}
```

02 Configuración

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, COMERCIO Y TURISMO
EOI Escuela de organización industrial

Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro

SISTEMA NACIONAL DE GARANTÍA JUVENIL

Spring Data

Configuración

```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter</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>
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

Configuración

```
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.11</version>
</dependency>
<dependency>
        <groupId>org.modelmapper</groupId>
        <artifactId>modelmapper</artifactId>
        <version>2.3.4</version>
</dependency>
```

GOBIERNO DE ESPAÑA MINISTERIO DE INDUSTRIA, COMERCIO Y TURISMO
EOI Escuela de organización industrial
Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro
SISTEMA NACIONAL DE GARANTÍA JUVENIL

application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/ej_eoi?serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=1234
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.database-platform = org.hibernate.dialect.MySQL5Dialect
```

03 Ejemplo

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, COMERCIO Y TURISMO
Escuela de organización industrial

Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro

SISTEMA NACIONAL DE GARANTÍA JUVENIL

Spring Data

Ejemplo

```
@RequestMapping(method = RequestMethod.GET, produces = MediaType.APPLICATION_JSON_VALUE)
public ResponseEntity<List<ClienteDto>> findAll() {
    ModelMapper mapper = new ModelMapper();
    List<ClienteDto> clientesDto;
    List<Cliente> clientes = service.findAll();
    java.lang.reflect.Type targetListType = new TypeToken<List<ClienteDto>>() {}.getType();
    clientesDto = mapper.map(clientes, targetListType);
    return new ResponseEntity<>(clientesDto,HttpStatus.OK);
}
```

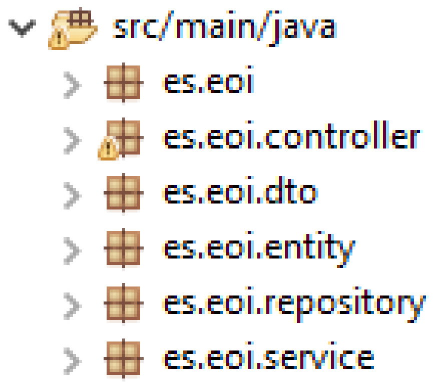
04

Ejercicios

Ejercicios JRepository

Vamos a modificar el ejercico anterior de **CLIENTES – CUENTAS – BANCOS**, suprimiremos el repositorio actual y utilizaremos un repositorio JPA Repository

Deberemos crear la siguiente estructura de paquetes:



```
src/main/java
  es.eoi
  es.eoi.controller
  es.eoi.dto
  es.eoi.entity
  es.eoi.repository
  es.eoi.service
```