



SPRING

Jérémy PERROUAULT



SPRING DATA-JPA

Généralisation des
Repositories

FONCTIONNEMENT

Déclaration d'interfaces qui étendent l'interface *JpaRepository*<T, ID>

- **@Repository** devient implicite
- **@Transactional** devient implicite

FONCTIONNEMENT

Utilisation d'une convention pour nommer les méthodes

- `findByNom(String nom)`
- `findByReference(String reference)`
- `findBy...(arguments attendus)`
- `findByNomContaining(String nom)`

Possibilité d'ajouter des Queries spécifiques

- Utilisation de l'annotation `@Query("<Requête JP-QL>")` sur la méthode
- Utilisation de l'annotation `@Modifying` sur la méthode
 - Si c'est une requête de mise à jour (update ou delete)

FONCTIONNEMENT

```
public interface IProduitRepository extends JpaRepository<Produit, Integer> {  
    public List<Produit> findByPrix(Double prix);  
  
    public Produit findByLibelle(String libelle);  
  
    public List<Produit> findByLibelleContaining(String libelle);  
  
    @Query("select p from Produit p where p.libelle = :lelibelle")  
    public Produit findUnProduit(@Param("lelibelle") String libelleProduit);  
}
```

Si la méthode retourne un Produit

- Spring retournera *null* si rien n'a été trouvé
- Spring lèvera une exception si plusieurs éléments ont été trouvés

Si la méthode retourne une liste de Produit

- Spring retournera une liste !

FONCTIONNEMENT

Si la classe modèle est annotée de requête(s) nommée(s)

- Spring DATA-JPA est capable d'exécuter ces requêtes
- Il suffit que le nom de la méthode dans la **DAO** soit identique au nom de la requête nommée !

Sur la classe Produit

```
@NamedQueries({  
    @NamedQuery(  
        name="Produit.unNomDeMethode",  
        query="select p from Produit p where p.libelle = :libelle"  
    )  
})  
@Entity  
public class Produit {  
  
}
```

Dans la DAO

```
public List<Produit> unNomDeMethode(@Param("libelle") String libelle);
```

FONCTIONNEMENT

Convention	Effet
findByAttribut	Rechercher sur la valeur d'un attribut
findByAttribut1AndAttribut2	Rechercher sur la valeur de deux attributs
findByAttributContaining	Rechercher une valeur contenue dans un attribut
findAllByOrderByAttributDesc	Rechercher tout et ranger par ordre décroissant sur l'attribut
findFirstByAttribut	Rechercher le premier élément sur la valeur d'un attribut
findTop10ByAttribut	Rechercher les 10 premiers éléments sur la valeur d'un attribut
findTop10ByAttributContaining	Rechercher les 10 premiers éléments dont la valeur d'un attribut contient
findTop10ByAttributContainingOrder ByIdDesc	Rechercher les 10 derniers éléments dont la valeur d'un attribut contient
countByLibelleContaining	Compter le nombre d'éléments dont la valeur d'un attribut contient

FONCTIONNEMENT

```
public interface IProduitRepository extends JpaRepository<Produit, Integer> {  
    public List<Produit> findByLibelle(String libelle);  
  
    public Produit findFirstByLibelle(String libelle);  
  
    public List<Produit> findTop10ByLibelleContainingOrderByIdDesc(String libelle);  
  
    public int countByLibelleContaining(String libelle);  
}
```


FONCTIONNEMENT

Important à savoir

- Les méthodes *find* par défaut de Spring DATA-JPA retournent
 - Soit une `List<T>` (c'est le cas de *findAll*)
 - Soit un `Optional<T>` (c'est le cas de *findById*)
- Si la convention n'est pas respectée et que la méthode n'est pas annotée de **@Query**
 - Spring génère une erreur au démarrage de l'application

FONCTIONNEMENT

Il est possible de paginer les résultats

- En utilisant un objet de type **Pageable**

```
public List<Produit> findByLibelleContainingOrderByIdDesc(String libelle, Pageable pageable);
```

```
//Première page de 2 éléments  
repoProduit.findByLibelleContainingOrderByIdDesc("g", PageRequest.of(0, 2));  
  
//Deuxième page de 2 éléments  
repoProduit.findByLibelleContainingOrderByIdDesc("g", PageRequest.of(1, 2));
```

*La pagination peut aller plus loin : possibilité de trier avec **PageRequest** !*

CONFIGURATION

Dépendance

- spring-data-jpa

CONFIGURATION

Configuration XML (ne pas oublier les en-têtes)

```
<!-- Scan des JpaRepositories -->  
<jpa:repositories base-package="fr.formation.repo" />
```

Configuration par classe

- Annotation de la classe de configuration de
`@EnableJpaRepositories("fr.formation.dao")`

EXERCICE

Implémenter DATA-JPA