[Bài đọc] Spring Data Repositories

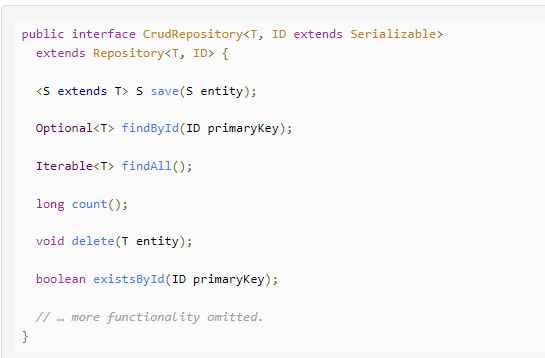
Các khái niệm chính

Mục đích của Spring Data repository là giúp giảm thiểu lượng code thông thường lặp đi lặp lại ở tầng truy xuất dữ liệu.

Interface chính của Spring Data repository là Repository. Interface này xác định lớp entity mà nó quản lý và kiểu dữ liệu của trường ID.

Interface Repository là một marker interface - chỉ sử dụng để đánh dấu chứ không có khai báo các phương thức.

Interface CrudRepository là một interface con của Repository, chứa các phương thức thông dụng dành cho các thao tác CRUD (thêm - sửa - xoá -hiển thị) thông dụng.



* Save(S entity): Lưu một entity
* findById(ID primaryKey): Tìm kiếm một entity dựa vào ID
* findAll(): Trả về tất cả các entity
* count(): Trả về số lượng các entity
* delete(T entity): Xoá một entity
* existsById(ID primaryKey): Kiểm tra xem một entity với ID nhất định có tồn tại hay không

Ngoài ra, có interface PagingAndSortingRepository cung cấp các phương thức hỗ trợ cho việc phân trang và sắp xếp các entity.



Ngoài ra, chúng ta còn có thể tự định nghĩa thêm các phương thức khác tuỳ theo nhu cầu của mình. Các câu lệnh truy vấn sẽ được sinh ra dựa theo tên của phương thức (hoặc chúng ta có thể tự viết câu truy vấn). Ví dụ:

Đếm số lượng User theo lastname:

interface UserRepository extends CrudRepository<User, Long> {

long countByLastname(String lastname);

}

Xoá User theo lastname:

interface UserRepository extends CrudRepository<User, Long> {

long deleteByLastname(String lastname);

List<User> removeByLastname(String lastname);

}

Các phương thức truy vấn

Các chức năng CRUD thông dụng thường cần đến các câu lệnh truy vấn dành cho các hệ quản trị dữ liệu khác nhau. Với việc sử dụng Spring Data, chúng ta có thể khai báo các câu truy vấn đó thông qua 4 bước:

1. Khai báo một interface kế thừa từ Repository hoặc các interface con của Repository. Ví dụ:

interface PersonRepository extends Repository<Person, Long> { … }

1. Khai báo các phương thức truy vấn trong interface
2. interface PersonRepository extends Repository<Person, Long> {
3. List<Person> findByLastname(String lastname);

}

1. Cấu hình Spring để tạo các thực thể proxy của các interface này, có thể sử dụng cấu hình XML hoặc javaConfig. Ví dụ:
2. import org.springframework.data.jpa.repository.config.EnableJpaRepositories;
3. @EnableJpaRepositories

class Config {}

1. Inject các thực thể repository vào để sử dụng. Ví dụ:
2. class SomeClient {
3. private final PersonRepository repository;
4. SomeClient(PersonRepository repository) {
5. this.repository = repository;
6. }
7. void doSomething() {
8. List<Person> persons = repository.findByLastname("Matthews");
9. }

}

Định nghĩa các interface repository

Khai báo một interface kế thừa Repository, CrudRepository, PagingAndSortingRepository... hoặc các repository con khác của interface Repository.

Định nghĩa các phương thức truy vấn

Các thực thể proxy có 2 cách để tạo ra các câu truy vấn:

* Dựa vào tên của phương thức
* Dựa vào câu truy vấn do chúng ta khai báo

Cơ chế của Spring Data cho phép duyệt qua các từ khoá như find…By, read…By, query…By, count…By, và get…By để tạo các câu truy vấn. Có thể đưa thêm các từ khoá như Distinc, Like, And, Or...

Ví dụ:

interface PersonRepository extends Repository<User, Long> {

List<Person> findByEmailAddressAndLastname(EmailAddress emailAddress, String lastname);

*// Enables the distinct flag for the query*

List<Person> findDistinctPeopleByLastnameOrFirstname(String lastname, String firstname);

List<Person> findPeopleDistinctByLastnameOrFirstname(String lastname, String firstname);

*// Enabling ignoring case for an individual property*

List<Person> findByLastnameIgnoreCase(String lastname);

*// Enabling ignoring case for all suitable properties*

List<Person> findByLastnameAndFirstnameAllIgnoreCase(String lastname, String firstname);

*// Enabling static ORDER BY for a query*

List<Person> findByLastnameOrderByFirstnameAsc(String lastname);

List<Person> findByLastnameOrderByFirstnameDesc(String lastname);

}

Hạn chế số lượng kết quả truy vấn

Có thể sử dụng các từ khoá như *top* hoặc *first* để quy định số lượng kết quả truy vấn. Có thể xác định một giá trị số để quy định số lượng kết quả trả về, hoặc nếu không có giá trị số thì mặc định 1 kết quả sẽ trả về.

Ví dụ:

User findFirstByOrderByLastnameAsc();

User findTopByOrderByAgeDesc();

Page<User> queryFirst10ByLastname(String lastname, Pageable pageable);

Slice<User> findTop3ByLastname(String lastname, Pageable pageable);

List<User> findFirst10ByLastname(String lastname, Sort sort);

List<User> findTop10ByLastname(String lastname, Pageable pageable);

Last modified: Tuesday, 5 March 2019, 2:31 PM

[Previous Activity](https://james.codegym.vn/mod/page/view.php?id=8950&forceview=1)

Top of Form