<https://www.baeldung.com/spring-retry>

Have a look at Maven Central for the latest versions of the [spring-retry](https://search.maven.org/search?q=spring-retry) and [spring-aspects](https://search.maven.org/search?q=a:spring-aspects) dependencies.

Các cách để sử dụng

* Annotations
* RetryTemplate
* Callbacks.

**Enabling Spring Retry**

Để bật Spring Retry trong một ứng dụng, chúng ta cần thêm chú thích @EnableRetry vào lớp @Configuration của chúng ta:

@Configuration @EnableRetry **public** **class** **AppConfig** { ... }

## ****Using Spring Retry****

### @Retryable**Without Recovery**

Để thêm chức năng thử lại vào các phương thức, chúng tôi có thể sử dụng chú thích @Retryable:

@Service

public interface MyService {

@Retryable(value = RuntimeException.class)

void retryService(String sql);

}

Trong ví dụ này, việc thử lại được thực hiện khi một RuntimeException được ném.

Theo hành vi mặc định của @ Retryable, quá trình thử lại có thể xảy ra tối đa ba lần, với độ trễ là một giây giữa các lần thử lại.

### @Retryable **and**@Recover

Bây giờ hãy thêm một phương thức khôi phục bằng cách sử dụng chú thích @Recover:

@Service

public interface MyService {

@Retryable(value = SQLException.class)

void retryServiceWithRecovery(String sql) throws SQLException;

@Recover

void recover(SQLException e, String sql);

}

Trong ví dụ này, việc thử lại được thực hiện khi một SQLException được ném ra. Chú thích @Recover xác định phương thức khôi phục riêng biệt khi phương thức @Retryable không thành công với một ngoại lệ được chỉ định.

Do đó, nếu phương thức retryServiceWithRecovery tiếp tục ném một SqlException sau 3 lần thử, phương thức recovery () sẽ được gọi.

Do đó, nếu phương thức retryServiceWithRecovery tiếp tục ném một SqlException sau 3 lần thử, phương thức recovery () sẽ được gọi.

Trình xử lý khôi phục phải có tham số đầu tiên là kiểu Throwable (tùy chọn) và cùng kiểu trả về. Các đối số sau đây được điền từ danh sách đối số của phương thức không thành công theo cùng một thứ tự.

### **Customizing**@Retryable's**Behavior**

Để tùy chỉnh hành vi của một lần thử lại, chúng tôi có thể sử dụng các tham số maxAttempts và backoff:

@Service

public interface MyService {

@Retryable( value = SQLException.class,

maxAttempts = 2, backoff = @Backoff(delay = 100))

void retryServiceWithCustomization(String sql) throws SQLException;

}

Trong ví dụ trên, sẽ có tối đa 2 lần thử và độ trễ là 100 mili giây.

## RetryTemplate