**Giới thiệu**

Yêu cầu bạn phải đọc bài viết về @Conditional trước:

1. [[Spring Boot] Hướng dẫn tạo Bean có điều kiện với @Conditional](https://loda.me/spring-boot-huong-dan-tao-bean-co-dieu-kien-voi-conditional-loda1559137415075)

Tôi đã giới thiệu với các bạn các sử dụng các loại @Conditional có sẵn trong Spring Boot. Tuy nhiên, trên thực tế, sẽ có những lúc yêu cầu các loại điều kiện nằm ngoài phạm vi của Spring Boot cung cấp.

Khi đó, chúng ta phải tự tạo @Condinal cho mình.

**Tự tạo @Conditional**

Để tạo ra một điều kiện cho mình, bạn cần kế thừa lớp Condition, và implement lại function matches

matches là nơi bạn kiểm tra điều kiện xem có thoả mãn không.

/\*

Một điều kiện, phải kế thừa lớp Condition của Spring Boot cung cấp

\*/

public class WindowRequired implements Condition{

@Override

public boolean matches(ConditionContext conditionContext, AnnotatedTypeMetadata annotatedTypeMetadata) {

// Nếu OS ra window trả ra true.

return System.getProperty("os.name").toLowerCase().contains("win");

}

}

Và khi đã định nghĩa được cho mình một điều kiện riêng, bạn có thể sử dụng như sau:

@Configuration

public class AppConfiguration {

public static class SomeBean{

}

/\*

SomeBean chỉ được tạo ra khi

thỏa mãn điều kiện

\*/

@Conditional(WindowRequired.class)

@Bean

SomeBean someBean(){

return new SomeBean();

}

}

**Tự tạo Annotation @Conditional**

Nếu việc viết @Conditional(WindowRequired.class) chưa làm bạn hài lòng, bạn có thể tự tạo ra một Annotation giống với Spring Boot.

Ví dụ như @ConditionalOnClass trong [bài viết trước](https://loda.me/spring-boot-huong-dan-tao-bean-co-dieu-kien-voi-conditional-loda1559137415075)

Thì hãy làm như sau:

@Target({ ElementType.TYPE, ElementType.METHOD })

@Retention(RetentionPolicy.RUNTIME)

@Documented

// Đánh dấu annotation này bởi @Conditional(WindowRequired.class)

@Conditional(WindowRequired.class)

public @interface ConditionalOnWindow {

/\*

Trong trường hợp bạn muốn viết ngắn gọn,

hay tạo ra 1 Annotation mới và gắn @Conditional(WindowRequired.class)

trên nó

Như vậy khi cần sử dụng chỉ cần gọi @ConditionalOnWindow là được

\*/

}

Khi sử dụng:

@Configuration

public class AppConfiguration {

public static class SomeBean{

}

/\*

SomeBean chỉ được tạo ra khi

thỏa mãn điều kiện

\*/

@ConditionalOnWindow

@Bean

SomeBean someBean(){

return new SomeBean();

}

}

**Chạy thử**

@SpringBootApplication

public class App {

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(App.class, args);

try {

SomeBean someBean = context.getBean(SomeBean.class);

System.out.println("SomeBean tồn tại!");

}catch (Exception e){

System.out.println("SomeBean chỉ được tạo nếu chạy trên Window");

}

}

}

Hãy bỏ Annotation @ConditionalOnWindow đi và chạy thử, sau đó thêm lại, xem kết quả của 2 lần chạy.

**Kết hợp nhiều điều kiện với OR**

Bạn có thể kết hợp nhiều điều kiện với nhau bởi phép OR.

Spring Boot hỗ trợ điều này bằng cách kế thừa lớp AnyNestedCondition

/\*\*

\* Class kế thừa AnyNestedCondition sẽ chấp nhận mọi

\* điều kiện @Conditional bên trong nó

\*/

public class WindowOrMacRequired extends AnyNestedCondition{

public WindowOrMacRequired(){

super(ConfigurationPhase.REGISTER\_BEAN);

}

/\*

Bạn phải định nghĩa các Điều kiện bên trong class

kế thừa AnyNestedCondition

\*/

@Conditional(WindowRequired.class)

public class RunOnWindow{}

/\*

Lúc này, cả 2 điều kiện Window và Mac sẽ được kết hợp vs

nhau khi kiểm tra, nếu thoả mãn 1 trong 2 là đc

\*/

@Conditional(MacRequired.class)

public class RunOnMac{}

}

Sử dụng

@Configuration

public class AppConfiguration {

public static class SomeBean{

}

/\*

SomeBean chỉ được tạo ra khi

thỏa mãn điều kiện

\*/

@Conditional(WindowOrMacRequired.class)

@Bean

SomeBean someBean(){

return new SomeBean();

}

}

**Kết hợp nhiều điều kiện với AND**

Bạn có thể kết hợp các điều kiện bằng phép AND bằng cách kế thừa lớp AllNestedConditions.

Cách kế thừa của nó giống với AnyNestedCondition nên tôi sẽ không cần đề cập thêm.

Ngoài ra, có một cách khác là sử dụng nhiều custom @Conditional cùng một lúc.

@Bean

@ConditionalOnWindow

@Conditional(MacRequired.class)

SomeBean someBean() {

return new SomeBean();

}

**Kết**

Tới đây, bạn có thể nắm vững được cách tạo điều kiện cho cấu hình ứng dụng của mình, nó sẽ rất có ích khi bạn làm việc và tách biệt được hai môi trường dev và production.