**Giới thiệu**

Trong thực tế không phải lúc nào chúng ta cũng nên để mọi thứ trong code của mình. Có những thông số tốt hơn hết nên được truyền từ bên ngoài vào ứng dụng, để giúp ứng dụng của bạn dễ dàng thay đổi giữa các môi trường khác nhau.

Để phục vụ điều này, chúng ta sẽ tìm hiểu về khái niệm config ứng dụng **Spring Boot** với application.properties

Sau bài này bạn có thể xem thêm nội dung sau:

1. [Hướng dẫn sử dụng Spring Properties với @ConfigurationProperties](https://loda.me/huong-dan-su-dung-spring-properties-voi-configuration-properties-loda1556418741178)
2. [「Spring-boot」Hướng dẫn sử dụng Spring Profiles](https://loda.me/spring-boot-huong-dan-su-dung-spring-profiles-loda1552901817707)

**Cài đặt**

*pom.xml*

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<packaging>pom</packaging>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.0.5.RELEASE</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<groupId>me.loda.spring</groupId>

<artifactId>spring-boot-learning</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-boot-learning</name>

<description>Everything about Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<!--spring mvc, rest-->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

Cấu trúc thư mục:



**application.properties**

Trong **Spring Boot**, các thông tin cấu hình mặc định được lấy từ file resources/applications.properties.

Ví dụ, bạn muốn **Spring Boot** chạy trên port 8081 thay vì 8080:

*applicatoin.properties*

server.port = 8081

Hoặc bạn muốn log của chương trình chi tiết hơn. Hãy chuyển nó sang dậng Debug bằng cách config như sau:

logging.level.root=DEBUG

Đây là cách chúng ta có thể can thiệp vào các cấu hình của ứng dụng từ bên ngoài. Cho phép thay đổi linh hoạt tùy môi trường.

**@Value**

Trong trường hợp, bạn muốn tự config những giá trị của riêng mình, thì **Spring Boot** hỗ trợ bạn với annotation @Value

Ví dụ, tôi muốn cấu hình cho thông tin database của tôi từ bên ngoài ứng dụng

*application.properties*

loda.mysql.url=jdbc:mysql://host1:33060/loda

@Value được sử dụng trên thuộc tính của class, Có nhiệm vụ lấy thông tin từ file properties và gán vào biến.

public class AppConfig {

// Lấy giá trị config từ file application.properties

@Value("${loda.mysql.url}")

String mysqlUrl;

}

Thông tin truyền vào annottaion @Value chính là tên của cấu hình đặt trong dấu ${name}

**Ví dụ**

Vẫn là ví dụ trên, chúng ta sẽ làm hoàn chỉnh.

Thông tin file *application.properties* bao gồm:

*application.properties*

server.port = 8081

logging.level.root=INFO

loda.mysql.url=jdbc:mysql://host1:33060/loda

Tạo ra class DatabaseConnector có nhiệm vụ kết nối tới database.

*DatabaseConnector.java*

public abstract class DatabaseConnector {

private String url;

/\*\*

\* Hàm này có nhiệm vụ Connect tới một Database bất kỳ

\*/

public abstract void connect();

public String getUrl() {

return url;

}

public void setUrl(String url) {

this.url = url;

}

}

Kế thừa nó là MySqlConnector

*MySqlConnector.java*

public class MySqlConnector extends DatabaseConnector {

@Override

public void connect() {

System.out.println("Đã kết nối tới Mysql: " + getUrl());

}

}

Chương trình sẽ được cấu hình trong AppConfig

*AppConfig.java*

import org.springframework.beans.factory.annotation.Value;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class AppConfig {

// Lấy giá trị config từ file application.properties

@Value("${loda.mysql.url}")

String mysqlUrl;

@Bean

DatabaseConnector mysqlConfigure() {

DatabaseConnector mySqlConnector = new MySqlConnector();

// Set Url

System.out.println("Config Mysql Url: " + mysqlUrl);

mySqlConnector.setUrl(mysqlUrl);

// Set username, password, format, v.v...

return mySqlConnector;

}

}

Chạy thử chương trình:

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class App {

public static void main(String[] args) {

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

DatabaseConnector databaseConnector = context.getBean(DatabaseConnector.class);

databaseConnector.connect();

}

}

Output:

2019-05-18 17:16:45.489 INFO 14004 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8081 (http) with context path ''

2019-05-18 17:16:45.493 INFO 14004 --- [ restartedMain] m.loda.spring.applicationproperties.App : Started App in 4.402 seconds (JVM running for 5.932)

Đã kết nối tới Mysql: jdbc:mysql://host1:33060/loda

Bạn sẽ thấy là chương trình đã chạy trên port 8081. Và cấu hình về đường dẫn mysql của tôi tự tạo ra cũng được **Spring Boot** đọc lên và đưa vào giá trị này vào biến.

**Kết**