Spring Boot Maven 插件

要在基於 Maven 的 Spring Boot 專案中使用 `spring-boot-maven-plugin`，請按照以下步驟配置和利用插件來構建、運行和管理您的應用程序。以下是一個全面的指南：

---

1. 確保您的專案是 Spring Boot 專案

在使用 `spring-boot-maven-plugin` 之前，請確認您的專案已設置為 Spring Boot 專案。這通常涉及：

- **擴展自 `spring-boot-starter-parent`（推薦）**：
  - 在您的 `pom.xml` 中，將 `spring-boot-starter-parent` 設置為父項目，以管理 Spring Boot 依賴項和插件版本。
  - 範例：

```xml
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.5.4</version> <!-- 替換為您的 Spring Boot 版本 -->
    <relativePath/> <!-- 從存儲庫查找父項目 -->
</parent>
```

- **或使用 `spring-boot-dependencies` BOM（Bill of Materials）**：
  - 如果無法使用 `spring-boot-starter-parent`，請在 `dependencyManagement` 部分導入 `spring-boot-dependencies` BOM。
  - 範例：

```xml
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-dependencies</artifactId>
            <version>2.5.4</version> <!-- 替換為您的 Spring Boot 版本 -->
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```

使用 `spring-boot-starter-parent` 簡化配置，因為它會自動管理插件版本。

---

2. 在 pom.xml 中添加 spring-boot-maven-plugin

要使用插件，您需要在 pom.xml 的 <build><plugins> 部分聲明它。

- **如果使用 spring-boot-starter-parent**：
  - 不需要指定版本，因為它由父項目管理。
  - 範例：

```
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

- **如果不使用 spring-boot-starter-parent**：
  - 明確指定版本，與正在使用的 Spring Boot 版本匹配。
  - 範例：

```
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
            <version>2.5.4</version> <!-- 替換為您的 Spring Boot 版本 -->
        </plugin>
    </plugins>
</build>
```

---

3. 利用插件目標

spring-boot-maven-plugin 提供了幾個目標來幫助構建、運行和管理您的 Spring Boot 應用程序。以下是最常用的目標：

- spring-boot:run
  - 使用嵌入式 Web 伺服器（例如 Tomcat）直接從 Maven 運行您的 Spring Boot 應用程序。
  - 適合開發和測試。

- 命令：

```
mvn spring-boot:run
```

- **spring-boot:repackage**
  - 將 mvn package 生成的 JAR 或 WAR 文件重新打包為可執行的「肥 JAR」或 WAR，其中包含所有依賴項。
  - 如果插件已配置，則此目標會在 package 階段自動執行。
  - 命令：

```
mvn package
```

  - 運行後，您可以使用以下命令啟動應用程序：

```
java -jar target/myapp.jar
```

- **spring-boot:start 和 spring-boot:stop**
  - 用於在 pre-integration-test 和 post-integration-test 階段分別啟動和停止應用程序的集成測試。
  - 範例：

```
mvn spring-boot:start
mvn spring-boot:stop
```

- **spring-boot:build-info**
  - 生成包含構建信息（例如構建時間、版本）的 build-info.properties 文件。
  - 這些信息可以使用 Spring Boot 的 BuildProperties Bean 或 @Value 注釋在應用程序中訪問。
  - 命令：

```
mvn spring-boot:build-info
```

---

4. 自定義插件配置（可選）

您可以通過在 pom.xml 中添加配置選項來自定義 spring-boot-maven-plugin 的行為。以下是一些常見的自定義：

- **指定主類**：
  - 如果插件無法自動檢測主類，請手動指定它。
  - 範例：

```
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
```

```
            <configuration>
                <mainClass>com.example.MyApplication</mainClass>
            </configuration>
        </plugin>
    </plugins>
</build>
```

- **從肥 JAR 中排除依賴項**：
  - 排除由運行時環境提供的依賴項（例如外部 Servlet 容器）。
  - 範例：

```
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
            <configuration>
                <excludes>
                    <exclude>
                        <groupId>com.example</groupId>
                        <artifactId>some-dependency</artifactId>
                    </exclude>
                </excludes>
            </configuration>
        </plugin>
    </plugins>
</build>
```

- **設置應用程序參數**：
  - 配置要傳遞給應用程序的參數，以便使用 `spring-boot:run` 運行。
  - 在 `pom.xml` 中的範例：

```
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
            <configuration>
                <arguments>
                    <argument>--server.port=8081</argument>
                </arguments>
            </configuration>
```

```
            </plugin>
        </plugins>
    </build>
```

  - 或者，通過命令行傳遞參數：

```
mvn spring-boot:run -Dspring-boot.run.arguments=--server.port=8081
```

- **構建 WAR 文件**：
  - 如果要將 WAR 文件部署到外部 Servlet 容器，請確保在 `pom.xml` 中將項目打包設置為 `war`：

```
<packaging>war</packaging>
```

  - 您可能還需要排除嵌入式 Servlet 容器（例如 Tomcat），如果由環境提供。將以下依賴項添加為 `provided`：

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-tomcat</artifactId>
        <scope>provided</scope>
    </dependency>
</dependencies>
```

---

5. 關鍵說明

- **默認行為**：
  - 當您運行 `mvn package` 時，插件會在 `package` 阶段自動重新打包 JAR/WAR，使其可執行。
  - 重新打包的工件是主要工件，原始 JAR/WAR 會附加分類器（例如 `original`）。
- **開發工作流程**：
  - 使用 `mvn spring-boot:run` 進行快速開發和測試。
  - 使用 `mvn package` 來構建可部署的可執行 JAR 或 WAR。
- **集成測試**：
  - 使用 `spring-boot:start` 和 `spring-boot:stop` 來管理應用程序生命週期，進行集成測試。

總結

要使用 `spring-boot-maven-plugin`：1. 確保您的專案是 Spring Boot 專案，通常通過擴展 `spring-boot-starter-parent` 或使用 `spring-boot-dependencies` BOM。2. 將插件添加到 `pom.xml` 的 `<build><plugins>` 部分。3. 使用 Maven 命

令，例如：- `mvn spring-boot:run` 運行應用程序。- `mvn package` 來構建可執行的 JAR/WAR。- 根據需要使用其他目標，例如 `spring-boot:start`、`spring-boot:stop` 和 `spring-boot:build-info`。4. 可選地，在 `pom.xml` 中自定義插件配置以進行高級用例。

此設置使您能夠高效地使用 Maven 構建、運行和管理 Spring Boot 應用程序。