# Maven scope 为provided时，依赖的jar包依然被打入了项目的lib中

之前一直没有太在意maven scope这个属性，导致一些包运行冲突或者jar包过于臃肿。于是我也重新学习了相关的概念。

scope元素

scope元素主要用来控制依赖的使用范围，指定当前包的依赖范围和依赖的传递性，也就是哪些依赖在哪些classpath中可用。常见的可选值有：compile, provided, runtime, test, system等。

compile:默认值。compile表示对应依赖会参与当前项目的编译、测试、运行等，是一个比较强的依赖。打包时通常会包含该依赖，部署时会打包到lib目录下。比如：spring-core这些核心的jar包。

provided:provided适合在编译和测试的环境，和compile功能相似，但provide仅在编译和测试阶段生效，provide不会被打包，也不具有传递性。比如：上面讲到的spring-boot-devtools、servlet-api等，前者是因为不需要在生产中热部署，后者是因为容器已经提供，不需要重复引入。

runtime:仅仅适用于运行和测试环节，在编译环境下不会被使用。比如编译时只需要JDBC API的jar，而只有运行时才需要JDBC驱动实现。

test:scope为test表示依赖项目仅参与测试环节，在编译、运行、打包时不会使用。最常见的使用就是单元测试类了。

system:system范围依赖与provided类似，不过依赖项不会从maven仓库获取，而需要从本地文件系统提供。使用时，一定要配合systemPath属性。不推荐使用，尽量从Maven库中引用依赖。

import: import scope只能用在dependencyManagement里面。表示从其它的pom中导入dependency的配置。

那么问题来了， 这是我项目的pom配置

<?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/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<dependencies>

<!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.8.7</version>

<scope>provided</scope>

</dependency>

我发现不管怎样，gson-2.8.7.jar这个jar包永远存在我的jar包的lib里面。

大爷的不是说provided就不会打到最终jar包吗？这让我百思不得其解，最终在不断地寻找下发现了问题所在。原来是spring-boot-maven-plugin这个插件造成的，他有什么用呢，对于初学者来说可能还比较懵逼。Spring Boot的Maven插件（Spring Boot Maven plugin）能够以Maven的方式为应用提供Spring Boot的支持，即为Spring Boot应用提供了执行Maven操作的可能。Spring Boot Maven plugin能够将Spring Boot应用打包为可执行的jar或war文件，然后以通常的方式运行Spring Boot应用。简单来说我们不需要Tomcat这些容器，执行java -jar xxx.jar就可以把这个项目运行起来了，还是挺方便的。

这个如果要排除指定的包该怎么做呢？下面看个例子就知道了

#1. 通过指定groupId和artifactId排除某个特定的依赖

<plugin>

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

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

<version>2.3.7.RELEASE</version>

<configuration>

<excludes>

<exclude>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

</exclude>

</excludes>

<mainClass>com.example.jucdemo.JucDemoApplication</mainClass>

</configuration>

<executions>

<execution>

<id>repackage</id>

<goals>

<goal>repackage</goal>

</goals>

</execution>

</executions>

</plugin>

#2. 指定groupId排除groupId相关的所有依赖

<build>

<plugins>

<plugin>

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

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

<configuration>

<excludeGroupIds>com.google.code.gson</excludeGroupIds>

</configuration>

</plugin>

</plugins>

</build>

然后看看结果

gson-2.8.7.jar这个包已经再也看不到了。

防止后人走弯路，大家多学习