

maven dependency中scope=compile 和 provided区别

转载 春天的早晨 于 2017-03-09 16:18:20 发布 62673 收藏 39

分类专栏: [maven](#) [Spring](#) 文章标签: [maven](#)

问题再现

上次这边朋友问我一个问题，就是他们在pom.xml中的dependency中，看到有一些是provided的情况，比如如下：

```
1 <dependency>
2   <groupId>com.liferay.portal</groupId>
3   <artifactId>portal-impl</artifactId>
4   <version>6.1.0</version>
5   <scope>provided</scope>
6 </dependency>
```

他们问我 **scope** 在何种情况下要设置为provided,以及和scope设置为compile的区别。

解释

其实这个问题很简单。

对于scope=compile的情况 (**默认scope**),也就是说这个项目在编译，测试，运行阶段都需要这个artifact(模块)对应的jar包在classpath中。


而对于scope=provided的情况，则可以认为这个provided是目标容器已经provide这个artifact。换句话说，**它只影响到编译，测试阶段。在编译测试阶段，我们需要这个artifact对应的jar包在classpath中，而在运行阶段，假定目标的容器（比如我们这里的liferay容器）已经提供了这个jar包，所以无需我们这个artifact对应的jar包了。**

实例（scope=provided）

比如说，假定我们自己的项目ProjectABC 中有一个类叫C1,而这个C1中会import这个portal-impl的artifact中的类B1，那么在编译阶段，我们肯定需要这个B1，否则C1通不过编译，因为我们的scope设置为provided了，所以编译阶段起作用，所以C1正确的通过了编译。测试阶段类似，故忽略。

那么最后我们要把ProjectABC部署到Liferay服务器上了，这时候，我们到 `$liferay-tomcat-home\webapps\R00T\WEB-INF\lib` 下发现，里面已经有了一个portal-impl.jar了，换句话说，容器已经提供了这个artifact对应的jar,所以，我们在运行阶段，这个C1类可以直接用容器提供的portal-impl.jar中的B1类，而不会出任何问题。

实际插件的行为

刚才我们讲述的是理论部分，现在我们看下，实际插件在运行时候，是如何来区别对待scope=compile和scope=provided的情况的。做一个实验就可以很容易发现，当我们用 `maven`  install生成最终的构件包ProjectABC.war后，**在其下的WEB-INF/lib中，会包含我们被标注为scope=compile的构件的jar包，而不会包含我们被标注为scope=provided的构件的jar包。这也避免了此类构件当部署到目标容器后产生包依赖冲突。**

依赖范围

maven中三种classpath

编译，测试，运行

- 1.compile: **默认范围**，编译测试运行都有效
- 2.provided: 在编译和测试时有效
- 3.runtime: 在测试和运行时有效
- 4.test: 只在测试时有效
- 5.system: 在编译和测试时有效，与本机系统关联，可移植性差


pom.xml常用元素介绍

project 包含pom一些约束的信息

modelVersion 指定当前pom的版本

groupId (主项目标识，定义当前maven属于哪个项目，反写公司网址+项目名)、

artifactId (实际项目模块标识，项目名+模块名)、

`version`  (当前项目版本号，第一个0标识大版本号，第二个0标识分支版本号，第三个0标识小版本号，0.0.1，snapshot快照，alpha内部测试，beta公测，release稳定，GA正式发布)

name项目描述名

url项目地址

description项目描述

developers开发人员列表

licenses许可证

organization: 组织

dependencies: 依赖列表

dependency: 依赖项目 里面放置坐标

scope: 包的依赖范围 test

optional : 设置依赖是否可选

exclusions: 排除依赖传递列表

dependencyManagement 依赖的管理

build: 为构建行为提供支持

plugins: 插件列表

parent: 子模块对父模块的继承

modules: 聚合多个maven项目

参考:

<http://supercharles888.blog.51cto.com/609344/981316/>