

pom.xml

2018.11.19 09:24:27 字数 1537 阅读 125

什么是pom?

pom.xml文件是Maven进行工作的主要配置文件。在这个文件中我们可以配置Maven项目的groupId、artifactId和version等Maven项目必须的元素；可以配置Maven项目需要使用的远程仓库；可以定义Maven项目打包的形式；可以定义Maven项目的资源依赖关系等等。

pom.xml标签解析

```
1 <!--project是pom.xml的根元素，包含了一些约束的信息，比如xmlns, xsi-->
2 <project>
3     <!--指定了当前pom的版本-->
4     <modelVersion>4.0.0</modelVersion>
5     <!--maven2.0必须是这样写，现在是maven2唯一支持的版本-->
6     <!-- 基础设置 -->
7     <groupId>反写公司的网址+项目名称</groupId>
8     <artifactId>项目名称+模块名</artifactId>
9     <version>当前项目版本号</version>
10    <!-- 默认是jar，还可以打包成war/zip/pom-->
11    <packaging>...</packaging>
12    <!-- 项目描述名，一般是写项目文档的时候才使用 -->
13    <name>...</name>
14    <!-- 项目的地址-->
15    <url>...</url>
16    <!-- 项目的描述 -->
17    <description>...</description>
18    <!-- 开发人员的列表 -->
19    <developers>...</developers>
20    <!-- 许可证的信息 -->
21    <licenses>...</licenses>
22    <!-- 组织信息 -->
23    <organization>...</organization>
24
25
26    <!-- 依赖列表，下面可以包含多个依赖项dependency-->
27    <dependencies>
28        <dependency>
29            <!-- 指定坐标确定依赖项的位置 -->
30            <groupId></groupId>
31            <artifactId></artifactId>
32            <version></version>
33            <type></type>
34            <!-- 依赖包的依赖范围-->
35            <scope></scope>
36            <!-- 这个标签只有true和false两个值，是用来设置依赖是否可选 -->
37            <optional></optional>
38            <!-- 排除依赖传递列表 -->
39            <exclusions>
40                <exclusion>
41                    </exclusion>
42            </exclusions>
43        </dependency>
44    </dependencies>
45
46
47    <!-- 依赖管理，里面包含多个依赖，但是它并不会被运行，即不会被引用到实际的依赖中-->
48    <!-- 这个标签主要是用来定义在父模块中，供子模块继承用 -->
49    <dependencyManagement>
50        <dependencies>
51            <dependency>
52                </dependency>
53        </dependencies>
54    </dependencyManagement>
55
56
57    <!-- 常用于给构件的行为提供相应的支持 -->
```

不知名的蛋挞

拥有8钻 (约0.93元)

关注

- resultMap使用总结
- 阅读 18
- spring中过滤器 (filter)、拦截器 (interceptor) 和切面 (aop) 的执...
- 阅读 14
- JAVA自定义注解和AOP配合使用
- 阅读 8

精彩继续

卖不掉的房子，被套在北京的我们

阅读 35980

此人是梁山内奸？宋江死后他混得最好，还贴身侍奉宋徽宗

阅读 3080

```
63         <artifactId></artifactId>
64         <version></version>
65     </plugin>
66 </plugins>
67 </build>
68
69 <!-- 用于在子模块中对父模块的pom的继承 -->
70 <parent>...</parent>
71 <!-- 用来聚合多个模块，让多个模块进行编译，不用一个一个来 -->
72 <modules>
73     <module>
74     </module>
75 </modules>
76
77 </project>
```

1. groupId、artifactId和version

在Maven中，使用groupId、artifactId和version组成groupId:artifactId:version的形式来唯一确定一个项目。对于一个最简单的pom.xml的定义必须包含modelVersion、groupId、artifactId和version这四个元素，当然这其中的元素也是可以从它的父项目中继承的。

- groupId：指公司里面开发的某个项目，即组织名称，例如：zttc.itat.maven，在F盘目录下，将是：zttc/itat/maven目录。
- artifactId：项目里面某一个具体的模块，例如：maven-ch01，在F盘目录下，将是：zttc/itat/maven/maven-01目录。
- version：版本号，例如：版本号为1.0，在M2_REPO目录下，将是：org/codehaus/mojo/my-project/1.0目录。

2. scope

scope释就是依赖包的依赖范围。

【scope的使用场景和说明】

1.compile

表示被依赖项目需要参与当前项目的编译，当然后续的测试，运行周期也参与其中，是一个比较强的依赖，打包的时候也通常需要包含进去。

2.provided

相当于compile，但是在打包阶段做了exclude的动作，也就是当我们打包的时候就不会把这个依赖加进去。最常见的是j2ee规范相关的servlet-api和jsp-api等jar包，一般Tomcat会提供，无需在打包到war包中，如果不配置为provided，把这些包打包到工程war包中，在tomcat6以上版本会出现冲突无法正常运行程序（版本不符的情况）。

3.runtime

一般是运行和测试环境使用，编译时候不用加入classpath，打包时候会打包到目标包中。一般是通过动态加载或接口反射加载的情况比较多。与compile相比，跳过编译而已。

4.test

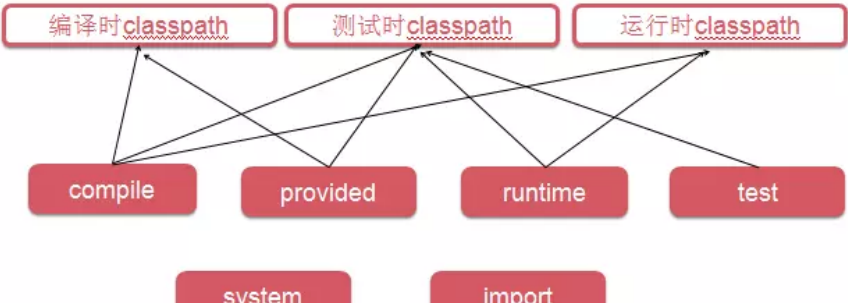
表示依赖项目仅仅参与测试相关的工作，包括测试代码的编译，执行，但是在项目编译和打包都不会使用这个依赖。比较典型的如junit。只有在src/test/java文件夹下面的才会用到这个作用域。

5.system

从参与度来说，也provided相同，不过被依赖项不会从maven仓库抓，而是从本地文件系统拿，一定需要配合systemPath属性使用。

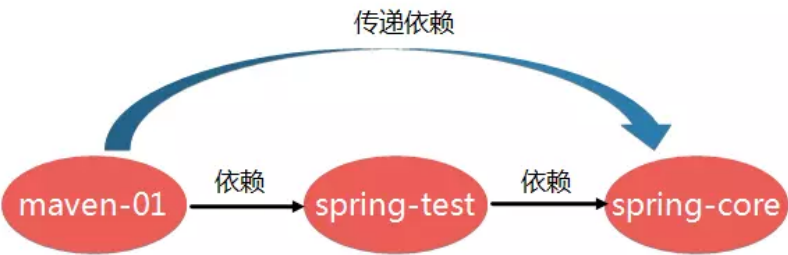
写下你的评论...

评论0 赞 ...



【scope的依赖传递】

比如我们引入某一个依赖spring-test,依赖传递特性会很方便帮助我们下来它相关的依赖，而不必有时会因为引入jar有问题而烦恼，但是也有弊端，存在一些不必要的依赖，可能会造成冲突。



【scope依赖冲突】

假设有如下依赖关系：

- A项目----->依赖于L.jar1.0版本
- B项目----->依赖于L.jar2.0版本
- C项目----->既依赖A也依赖B

那么传递给C的L.jar是哪一个版本的包呢？

A和L的关系叫做直接依赖，C和L的依赖叫做间接依赖。当我们有了间接依赖之后，我们先声明哪个依赖，就会用哪个依赖。比如userService先依赖于user-core（先写），再依赖于user-log（后写），user-core的log包是1.0.4的版本，user-log依赖的log包是1.2.7版本，则传到userService的log包就是user-core的log包（1.0.4版本）。

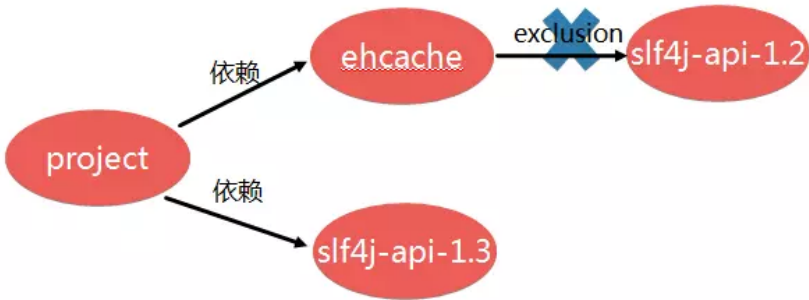
但是user-core的commons-logging包是1.0.4的版本，user-log依赖的commons-logging包是1.1.1版本，但传到userService的log包就是user-log的log包（1.1.1版本），这是为什么呢？打开依赖关系库发现commons-logging是dbunit的间接依赖：

——>user-core; 而user-log中是: commons-logging——>user-log, 依赖关系比user-log少一级。

总结: 当依赖级别相同的时候, 哪个先写就先用哪个; 当依赖级别不同的时候, 用级别/层次最短的那一个, 也就是短路优先。

【依赖排除】

依赖排除的特性也是为了解决依赖冲突的一个方法, <exclusions>标签能很方便去除依赖传递过程中不必要的依赖。



```
1 <dependency>
2   <groupId>org.springframework</groupId>
3   <artifactId>spring-test</artifactId>
4   <version>4.2.2.RELEASE</version>
5   <!-- 依赖排除 -->
6   <exclusions>
7     <exclusion>
8       <groupId>commons-logging</groupId>
9       <artifactId>commons-logging</artifactId>
10    </exclusion>
11  </exclusions>
12 </dependency>
```

3. 插件

Maven本质上是一个插件框架, 它主要的任务都是由插件来完成。定位到: %本地仓库%\org\apache\maven\plugins, 可以看到一些下载好的插件:

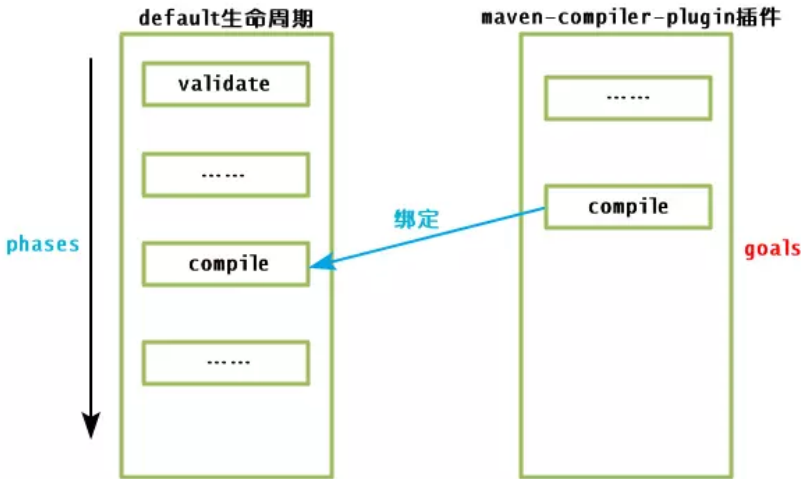


Maven的生命周期是抽象的, 实际需要插件来完成任务, 这一过程是通过将插件的目标 (goal) 绑定到生命周期的具体阶段 (phase) 来完成的。就是说插件目标有一下两种方式被执行:

例如，一个 Java 项目可以使用 Maven 编译器插件来编译目标，通过运行以下命令编译：

```
1 | mvn compiler:compile
```

- 通过绑定到生命周期阶段，而在生命周期阶段执行时被调用
如：将maven-compiler-plugin插件的compile目标绑定到default生命周期的compile阶段，完成项目的源代码编译：



👍 0人点赞 >

👎

📄 Maven

...



不知名的蛋挞
拥有8钻 (约0.93元)

关注

"小礼物走一走，来简书关注我"

赞赏

写下你的评论...

全部评论 0

只看作者


按时间倒序 按时间正序

推荐阅读

更多精彩内容 >

Spring Cloud

Spring Cloud为开发人员提供了快速构建分布式系统中一些常见模式的工具（例如配置管理，服务发现，断路器，智...

卡卡罗2017

Spring boot参考指南

Spring Boot 参考指南 介绍 转载自:https://www.gitbook.com/book/qbgb...

毛宇鹏

MAVEN简介之——pom.xml

写下你的评论...

评论0

👍 赞

...

简书

首页

下载APP

搜索

Q

Aa

beta

登录

注册

pom文件作为MAVEN中重要的配置文件，对于它的配置是相当重要。文件中包含了开发者需遵循的规则、缺陷管理系统、组...

千万之路刚开始

[Maven]-pom.xml文件详解

一、概述 当我们使用Maven来管理和构建我们的项目的时候，我们会不可避免的遇到pom文件。虽然已经配置过po...

骑着乌龟去看海