



链滴

maven 中 dependencyManagement import scope 依赖方式解决单继承问题的理解

作者: [zhaozhizheng](#)

原文链接: <https://ld246.com/article/1608172640812>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

在maven多模块项目中，为了保持模块间依赖的统一，常规做法是在parent model中，使用dependencyManagement预定义所有模块需要用到的dependency(依赖)

parent: 复用父类元素

dependencyManagement: 子模块中可以选择性继承父类的依赖，此标签中的依赖子pom不会自继承，子pom需要额外声明。但是，当依赖版本在父POM中声明后，子模块在使用依赖的时候就无声明版本，确保版本一致。

Import: 只在dependencyManagement元素下才有效果，作用是将目标POM中的dependencyManagement配置导入并合并到当前POM的dependencyManagement元素中。

properties属性: 通过元素用户可以自定义一个或多个Maven属性，然后在POM的其他地方使用\${姓名}的方式引用该属性，这种做法的最大意义在于消除重复和统一管理。

Maven总共有6类属性，内置属性、POM属性、自定义属性、Settings属性、java系统属性和环境变量属性；

```
<dependencyManagement>
  <dependencies>
    <!-- Feign是一种声明式、模板化的HTTP客户端:以HTTP接口的形式暴露自身服务 -->
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-feign</artifactId>
      <version>${spring-cloud-starter-feign.version}</version>
    </dependency>
    <!-- 支持面向方面的编程即AOP，包括spring-aop和AspectJ -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-aop</artifactId>
      <version>${spring.boot.version}</version>
    </dependency>
    <dependency>
      <groupId>org.aspectj</groupId>
      <artifactId>aspectjrt</artifactId>
      <version>${aspectjrt.version}</version>
    </dependency>
  </dependencies>
</dependencyManagement>
...
```

然后，子model根据实际需要引入parent中预定义的依赖

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-aop</artifactId>
  </dependency>
</dependencies>
```

好处:

1. 依赖统一管理(parent中定义，需要变动dependency版本，只要修改一处即可);

2. 代码简洁(子model只需要指定groupId、artifactId即可)

3. dependencyManagement只会影响现有依赖的配置，但不会引入依赖，即子model不会继承parent中dependencyManagement所有预定义的dependency，只引入需要的依赖即可，简单说就是“需引入依赖”或者“按需继承”；因此，在parent中严禁直接使用dependencies预定义依赖，坏处子model会自动继承dependencies中所有预定义依赖；

但是，问题也出现了：

单继承：maven的继承跟java一样，单继承，也就是说子model中只能出现一个parent标签；

parent模块中，dependencyManagement中预定义太多的依赖，造成pom文件过长，而且很乱；

如何让这些依赖可以分类并清晰的管理？

问题解决：import scope依赖

如何使用：

1. maven2.9以上版本

2. 将dependency分类，每一类建立单独的pom文件

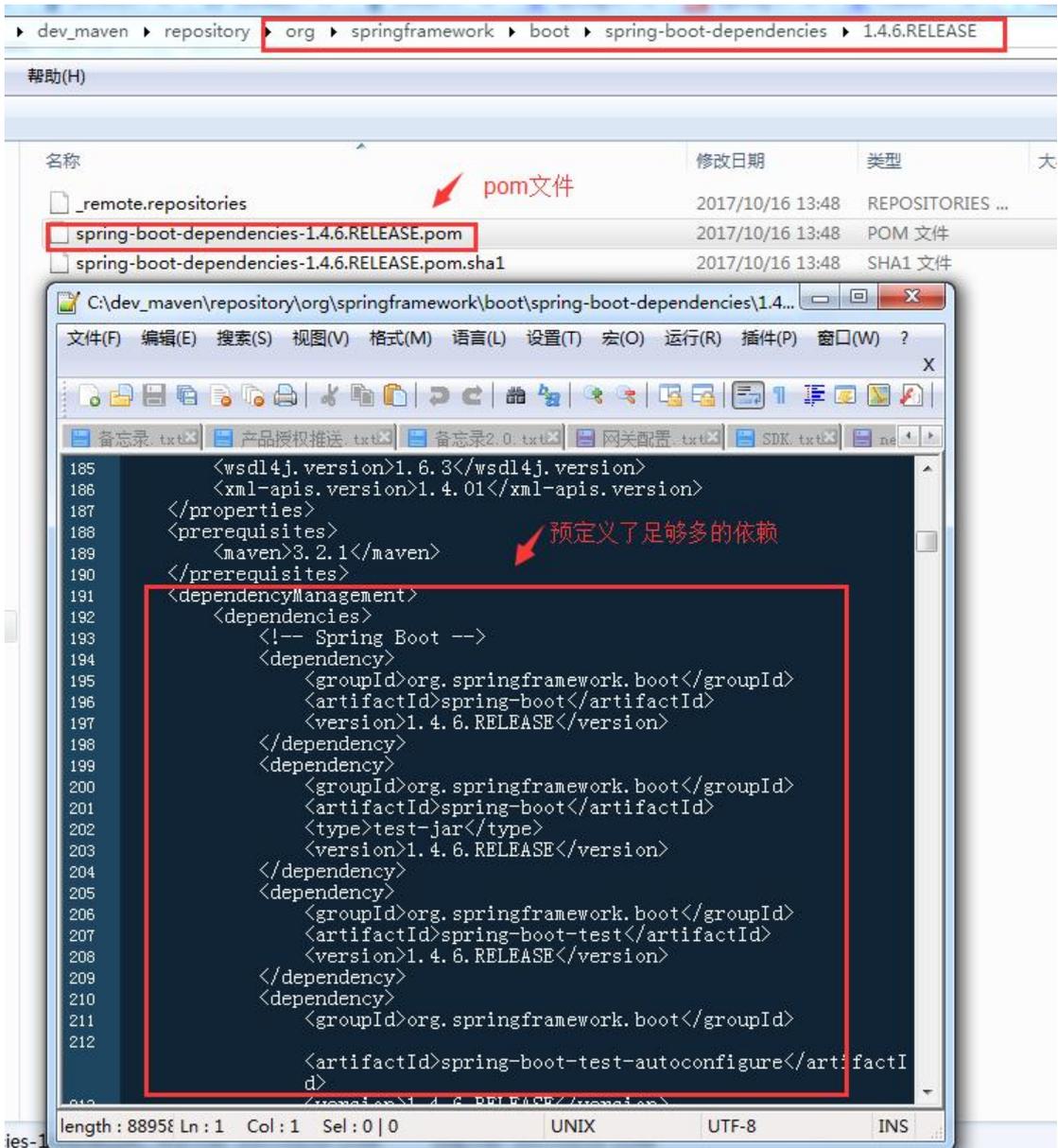
3. 在需要使用到这些依赖的子model中，使用dependencyManagement管理依赖，并import scope依赖

4. 注意：scope=import只能用在dependencyManagement里面,且仅用于type=pom的dependency

示例：

```
<pre> <span>&lt;/span> <span>dependencyManagement</span> <span>&gt;</span>
  <span>&lt;/span> <span>dependencies</span> <span>&gt;</span>
    <span>&lt;/span> <span>dependency</span> <span>&gt;</span>
      <span>&lt;/span> <span>groupId</span> <span>&gt;</span> org.springframework
rk.boot<span>&lt;/span> <span>groupId</span> <span>&gt;</span>
      <span>&lt;/span> <span>artifactId</span> <span>&gt;</span> spring-boot-dep
ndencies<span>&lt;/span> <span>artifactId</span> <span>&gt;</span>
      <span>&lt;/span> <span>!--</span> <span>重要：版本号要和父模块中预定义的spring boot版本号
持一致</span> <span>--&gt;</span>
      <span>&lt;/span> <span>version</span> <span>&gt;</span> ${spring.boot.versi
n}<span>&lt;/span> <span>version</span> <span>&gt;</span>
      <span>&lt;/span> <span>type</span> <span>&gt;</span> pom<span>&lt;/span> <span>&lt;/span> <span>type</span> <span>&gt;</span>
      <span>&lt;/span> <span>scope</span> <span>&gt;</span> import<span>&lt;/span> <span>&lt;/span> <span>scope</span> <span>&gt;</span>
      <span>&lt;/span> <span>dependency</span> <span>&gt;</span>
      <span>&lt;/span> <span>dependencies</span> <span>&gt;</span>
      <span>&lt;/span> <span>dependencyManagement</span> <span>&gt;</span> </pre>
```

maven编译后，下载了pom文件，在maven的本地仓库下查看pom文件如下：



好处分析:

1. 单一职责原则，根据依赖的分类，细化每一个单一职责的pom文件
2. 解决单继承问题，通过import pom文件达到依赖的目的（典型的非继承模式），从而不用从父类引用依赖
3. 父模块的pom就会非常干净，容易维护

参考文章:

[Maven实战（三）——多模块项目的POM重构](#)

[使用import scope解决maven继承（单）问题](#)

[Maven中的import scope](#)

[maven之非继承引用dependency](#)