InfoQ Maven专栏（十）——Maven 3，是时候升级了

August 5th, 2011

去年10月份Apache Maven发布了3.0正式版，而在上个月的22号，Eclipse基金会宣布了Eclipse 3.7（Indigo）的发布，该版本Eclipse最大的新特性之一就是集成了Maven。下载Eclipse IDE for Java Developers版 本的用户会发现，Eclipse已经能够自动识别Maven项目了。Indigo中内置的Maven版本是3.0.2，这在一定程度上说明Maven 3已经非常稳定了。不过我相信一定还有很多Maven 2用户在犹豫是否升级，本文会介绍一些Maven 3最重要的特性，旨在帮助读者扫除疑虑，尽早享受Maven 3所能带来的各种便利。

确保兼容性

在升级软件的时候，兼容性显然是首先要考虑的问题，如果原本在Maven 2下能成功构建的项目，在Maven 3下立刻就失败了，而且难以简单修复，那显然是不可接受的。 好在Maven用户大可打消这一顾虑，Maven 3自设计之初就一直考虑与Maven 2的兼容性，这不仅是指兼容Maven 2的核心，还包括大量的org.apache.maven.plugins与org.codehaus.mojo插件。虽然由于某些插件代码的特殊性，无 法做到100%完全的兼容，但已经基本不会遇到问题了。

如果你还有担心，那可以先仔细阅读官方发布的Maven 3.x兼容性笔记和Maven 3.x插件兼容性列表。这两份文档记录的兼容性问题主要涉及的是一些应当被弃用的特性，并且都给出了Maven 3下的解决方案。

改进性能

Maven 3的性能较之于Maven 2是有了很大的进步的，这体现在内存占用的减少和构建时间的减少两个方面。特别是Maven 3引入的并行构建特性，能够分析项目模块之间的依赖关系，然后并行地构建那些相互间没有依赖关系的模块，从而充分利用如今普遍的多核CPU资源。

以下两条命令分别表示用4个线程构建，以及根据CPU核数每个核分配1个线程进行构建：

```
1 $ mvn -T 4 clean install
2 $ mvn -T 1C clean install
```

Maven的提交者之一Anders Hammar在其文章迁移到Maven 3的十大理由中介绍了一个简单的实验，分别用Maven 2.2.1，Maven 3.0.2（单线程），和Maven 3.0.2（4线程）构建同样的包含32个模块的Maven源代码，得到了如下的结果：

Table 1. 用”mvn package”构建Maven SCM trunk（32个模块）

	时间/内存
Maven 2.2.1	3:20/53M
Maven 3.0.2（单线程）	3:15/27M
Maven 3.0.2（4线程）	2:26/28M

可以看到Maven 3下内存的占用减少了近一半！而开启并行构建后，时间的节省也是非常可观的。而项目越大，这种性能的改进就越为明显。如果你的开发环境没有充裕的内存，而你的项目又非常大，那光内存节省这一条就足以让你立刻转向Maven 3了。

改进模块间依赖解析

Maven 2中一个比较令人头疼的问题是，当你在构建一个多模块项目的时候，为了使前面的模块能在后面模块classpath中生效，你必须将其install到本地仓库中之后，Maven才能解析使用。几乎所有Maven 2用户或早或晚都遇到了这个困惑，“为什么我已经 mvn clean package 了模块A，可构建模块B的时候还是无法看到A的更新呢？”这个问题在Maven 3中得以解决了，在构建多模块项目的时候，Maven 3会从反应堆（reactor）中解析模块间依赖，也就是说只要模块A执行了package，那模块B就能根据相对路径找到并解析使用A生成的jar文件。

提倡最佳实践

刚从Maven 2转到Maven 3的用户很可能会发现执行构建的时候命令行会打印很多如下的警告：

```
[WARNING] 'build.plugins.plugin.version' for org.apache.maven.plugins:maven-javadoc-plugin is m
[WARNING] It is highly recommended to fix these problems because they threaten the stability of
```

大部分如下的警告是因为用户在配置插件或者依赖的时候没有指定版本，无法保证构建的可重现性，从而为构建引入了潜在的风险。这样的警告是一种既保持兼容性，又提倡最佳实践而做出的权衡。类似的改进还包括弃用profile.xml特性、明确分离项目依赖和插件依赖等等。

改进日志输出

这是一个很微小的改进，却突显了Maven开发者对Maven用户的关怀，我个人非常喜欢这点改进。简单得来说，Maven 3的构建日志更容易阅读了。插件的输出之间都有空行隔开，每个被运行插件的版本、目标、以及所处模块的artifactId都得以清晰显示。当构建出现错误的时候，这样的输出能帮助用户更快地找到问题所在。

站点（注意！）

站点这一特性是Maven 2的核心，但是在Maven 3中，该特性被完全移到了maven-site-plugin中，这就导致了相关的配置也需要转移。Maven 2中与站点相关的配置是在POM的reporting元素下的，如：

```
<project>
  ...
  <reporting>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-javadoc-plugin</artifactId>
        <version>2.7</version>
      </plugin>
    </plugins>
  </reporting>
</project>
```

在Maven 3中，所有站点相关的配置都应该出现在maven-site-plugin下面：

```
<project>
 ...
 <build>
   ...
   <plugins>
     ...
     <plugin>
       <groupId>org.apache.maven.plugins</groupId>
       <artifactId>maven-site-plugin</artifactId>
       <version>3.0-beta-3</version>
       <configuration>
         <reportPlugins>
           <plugin>
             <groupId>org.apache.maven.plugins</groupId>
             <artifactId>maven-javadoc-plugin</artifactId>
             <version>2.7</version>
           </plugin>
         </reportPlugins>
       </configuration>
     </plugin>
   </plugins>
 </build>
</project>
```

小结与致谢

本文从兼容性、新特性、性能改进、以及重要细节等方面全面介绍了Maven 3。Maven 3是Maven从成熟走向巅峰的标志，如果你还未升级，我强烈建议你至少尝试一下，Maven的安装是非常简单的，只需要下载一个zip包、解压、然后设置简单的环境变量即可，马上去下载吧！

由于能力及精力所限，我已经很难再写更多既不重复又符合很多读者口味的Maven文章，因此暂且计划将该专栏告一段落。我衷心感谢张凯峰的策划和编辑，感谢读者的支持，另外也感谢我的家人，特别是我三岁的女儿，那些给写稿的时间本该属于她们。最后，我还是会持续关心Maven的发展，有机会也一定会分享更多的经验和心得。

本文已经首发于InfoQ中文站，版权所有，原文为《Maven实战（十）——Maven 3，是时候升级了》
