  大家都知道Maven的优点是依赖管理，特别是前期使用ANT的开发者都有很多感触。最近要开发一个java工程，定的要使用maven，会使用 hadoop和hbase的客户端，而引入一个hadoop-client的jar或者hbase的jar包，会依赖十几个其他的jar包，而这些jar 包的功能我又用不上，所以这种依赖反倒成了工程瘦身的负担。关键我还有强迫症，见到这些对工程无用的包，我就抓狂。所以在网上百找千寻，找到了几个方法：

1. 项目间传递

    如果我的当前项目是project1，project1要依赖project2，project1依赖project2的配置中加上<optional>true</optional>，表示依赖可选，

Java代码  [收藏代码](javascript:void())

1. <dependency>
2. <groupId>com.projecct</groupId>
3. <artifactId>project2</artifactId>
4. <version>1.0</version>
5. <scope>compile</scope>
6. <optional>true</optional>
7. </dependency>

 那么以后所有声明依赖project1的项目如果也依赖project2，就必须写手动声明。比如project3依赖project1和 project2，如果project3只声明了对project1的依赖，那么project2不会自动加入依赖，需要重新声明对project2的依 赖。

这种方式排除不了我项目中对第三方jar包所依赖的其他依赖，因为我不可能去修改第三方jar包的pom文件，所以只适合在项目组内部使用。

2. 依赖过滤

（1）单依赖过滤

       同依赖过滤直接处理：可以过滤一个或者多个，如果过滤多个要写多个<exclusion>。这个也解决不了我的问题，或者说解决太麻烦，我那里知道hbase要依赖那些包，记不住。

Java代码  [收藏代码](javascript:void())

<dependency>

     <groupId>org.apache.hbase</groupId>

     <artifactId>hbase</artifactId>

     <version>0.94.17</version>

     <exclusions>

       <exclusion>

           <groupId>commons-logging</groupId>

           <artifactId>commons-logging</artifactId>

        </exclusion>

     </exclusions>

</dependency>

（2）多依赖过滤

     把所以依赖都过滤了。手起刀落~啊，世界都安静了。

Java代码  [收藏代码](javascript:void())

1. <dependency>
2. <groupId>org.apache.hbase</groupId>
3. <artifactId>hbase</artifactId>
4. <version>0.94.17</version>
5. <exclusions>
6. <exclusion>
7. <groupId>\*</groupId>
8. <artifactId>\*</artifactId>
9. </exclusion>
10. </exclusions>
11. </dependency>

自从搞定了这个依赖过滤啊，我腰不酸，腿不疼了，手不抖了，一口气啊，5000行代码，都不觉得累了~！看这里！看这里！看哪里！ 想依赖那里就依那里！妈妈再也不用担心我的学习了！