# 常见问题

## 1、maven下载依赖失败，提示项目错误

找到 无法下载依赖 的本地目录地址，然后将其删除，让maven重新下在即可。

## 2、缺少jdk tools.jar，提示项目错误

在pom.xml中添加依赖

<dependency>

<groupId>jdk.tools</groupId>

<artifactId>jdk.tools</artifactId>

<version>1.7</version>

<scope>system</scope>

<systemPath>${JAVA\_HOME}/lib/tools.jar</systemPath>

</dependency>

## 3、maven打包编译时后台一直输出警告信息

[WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is platform dependent!

解决方案：添加如下信息

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

## 4、构建cdh版本的hadaoop

在pom.xml中，添加资源依赖

<repositories>

<repository>

<id>cloudera</id>

<url>https://repository.cloudera.com/artifactory/cloudera-repos/</url>

</repository>

</repositories>

## 5. jar冲突解决方法

Maven默认处理策略

声明优先 > 最短路径优先

排除jar

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-hystrix</artifactId>

<version>1.4.4.RELEASE</version>

<exclusions>

<exclusion>

<groupId>com.google.guava</groupId>

<artifactId>guava</artifactId>

</exclusion>

</exclusions>

</dependency>

版本锁定

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

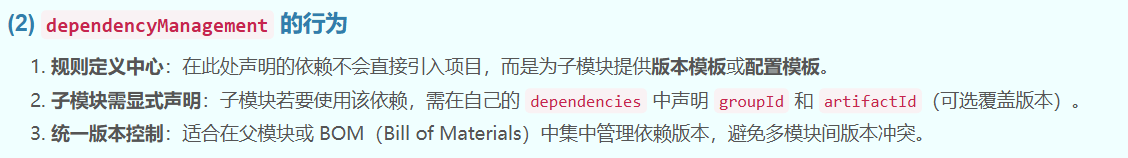
<version>${spring.versio}</version>

</dependency>

</dependencies>

</dependencyManagement>

# dependencies 与 dependencyManagement 的区别







解决的问题

### dependencies

父 pom 中定义的依赖，可以直接被子 pom 继承，子 pom 中不用再次声明。这听起来不错，但是仔细想想还是有一些不好的地方：

* 父 pom 定义的依赖，会被所有模块继承。即使有的模块可能根本不需要这个依赖，这就导致打包后的 jar 臃肿，甚至产生 jar 冲突的问题。
* 子模块的 pom 中可以覆盖继承的依赖，这可能导致依赖版本管理出现混乱。

### dependencyManagement

便可以很好的解决上面两个问题：

* dependencyManagement 中声明的依赖，仅仅是声明，并没有真正的引入依赖。这样子模块可以从声明的依赖中，选择自己需要的，而不会继承所有父 pom 中的依赖。
* dependencyManagement 中除了声明依赖，还指定了依赖的版本。这样在子模块的依赖中，就无须指定版本，而是使用 dependencyManagement 指定的版本。

# 常用

## 设置代码编译级别与编码方式

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

<encoding>UTF8</encoding>

</configuration>

</plugin>

## 打包插件

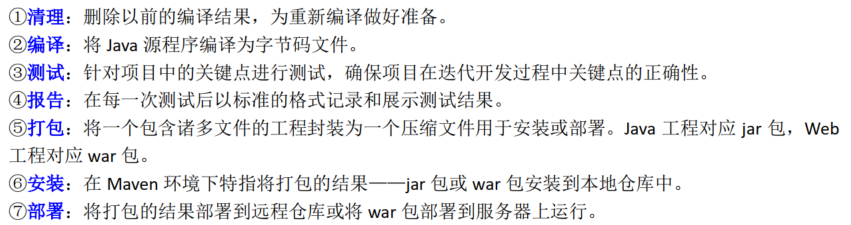


## profile动态选择配置文件

<https://www.cnblogs.com/0201zcr/p/6262762.html>

# 知识点

## 构建环节



## 依赖范围

compile： 此范围的依赖 对 主程序与测试程序有效，此依赖 参与打包。

provided： 此范围的依赖 对 主程序与测试程序有效，此依赖 不参与打包。

test： 此范围的依赖 仅对 测试程序有效，此依赖 不参与打包。