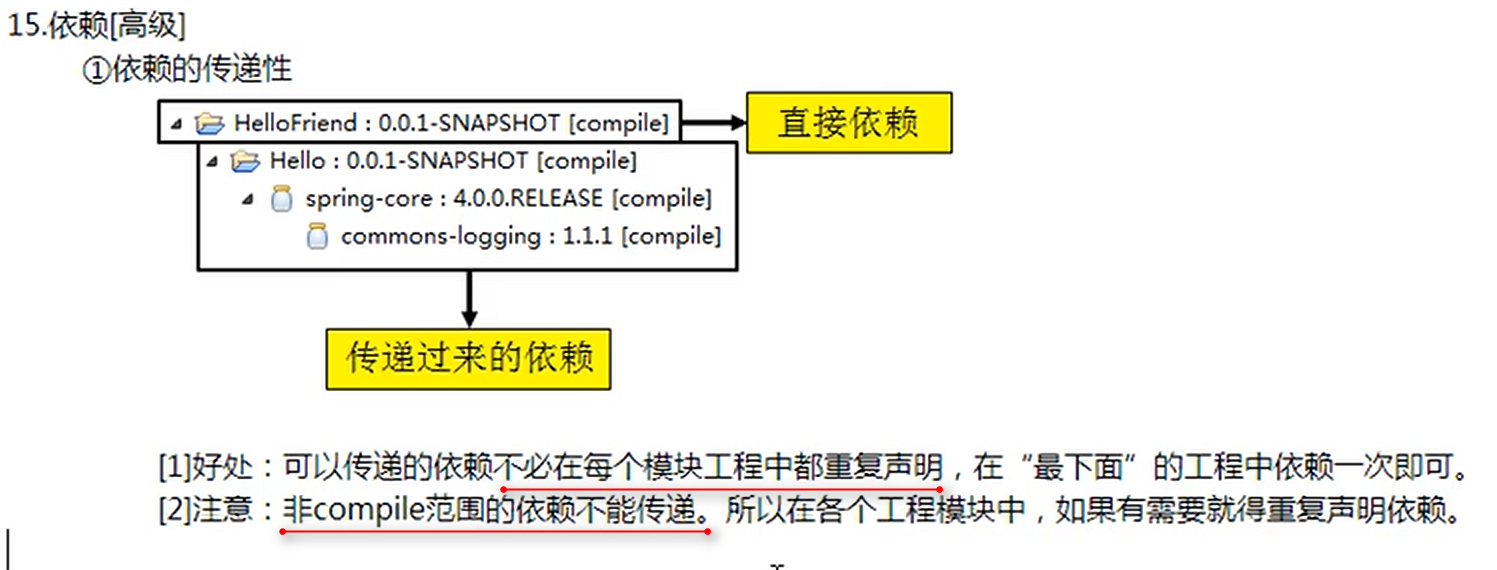
背景：三个工程分别是Hello、HelloFriend、MakeFriend

Hello 工程被打包成jar，HelloFriend 工程被打包成jar；

HelloFriend依赖Hello工程的jar包；MakeFriend依赖HelloFriend工程的jar包

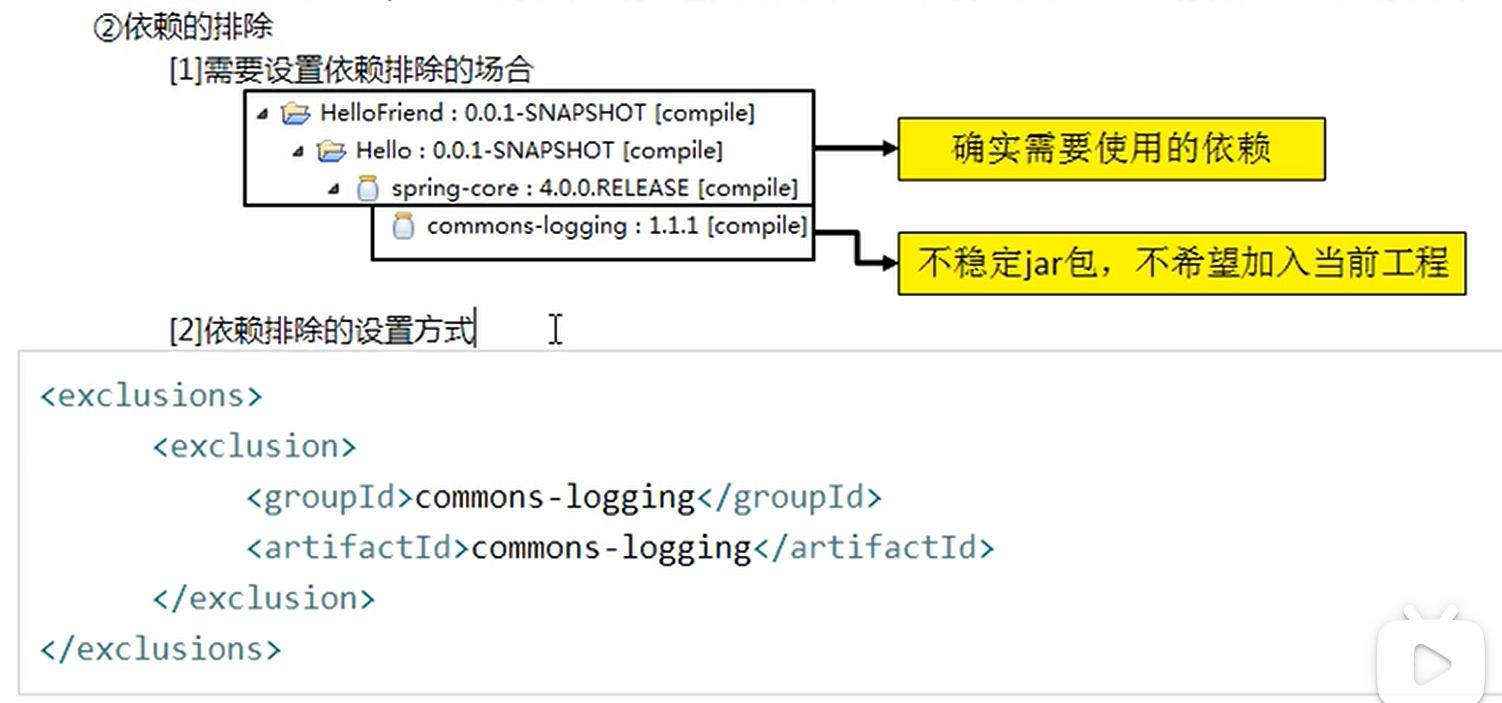
## 依赖

### 依赖的传递性



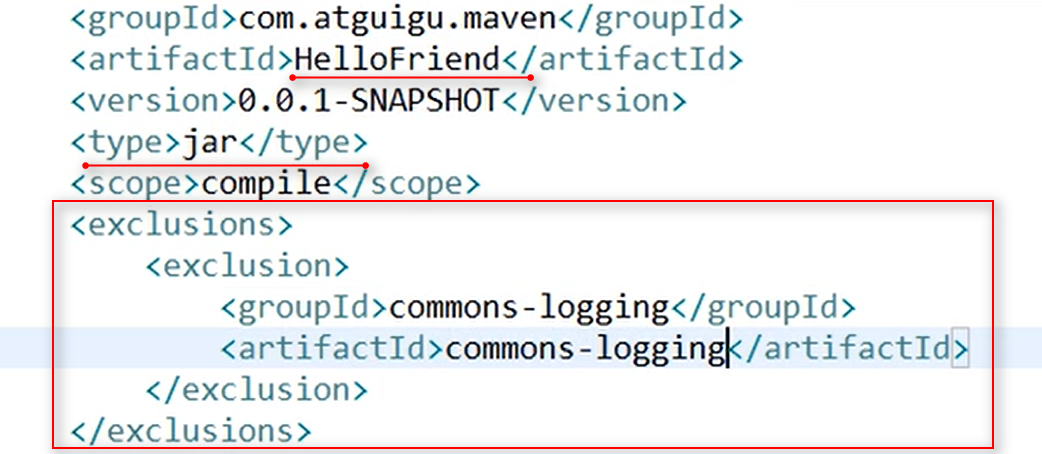
非complie范围的依赖指的是test、provide

### 依赖的排除



在Hello工程中依赖了spring-core的jar包，而该jar包依赖commons-logging；

在MakeFriend中引入了HelloFriend的jar包，会把spring-core及commons-logging都引入，**如果在MakeFriend工程中不想使用commons-logging的jar包**，可以采用如下方式进行依赖排除



### 依赖的原则



说明：

情景1：Hello工程依赖log4j.1.2.17；HelloFriend工程依赖log4j.1.2.14；则在MakeFriends工程中log4j的依赖版本是多少呢？

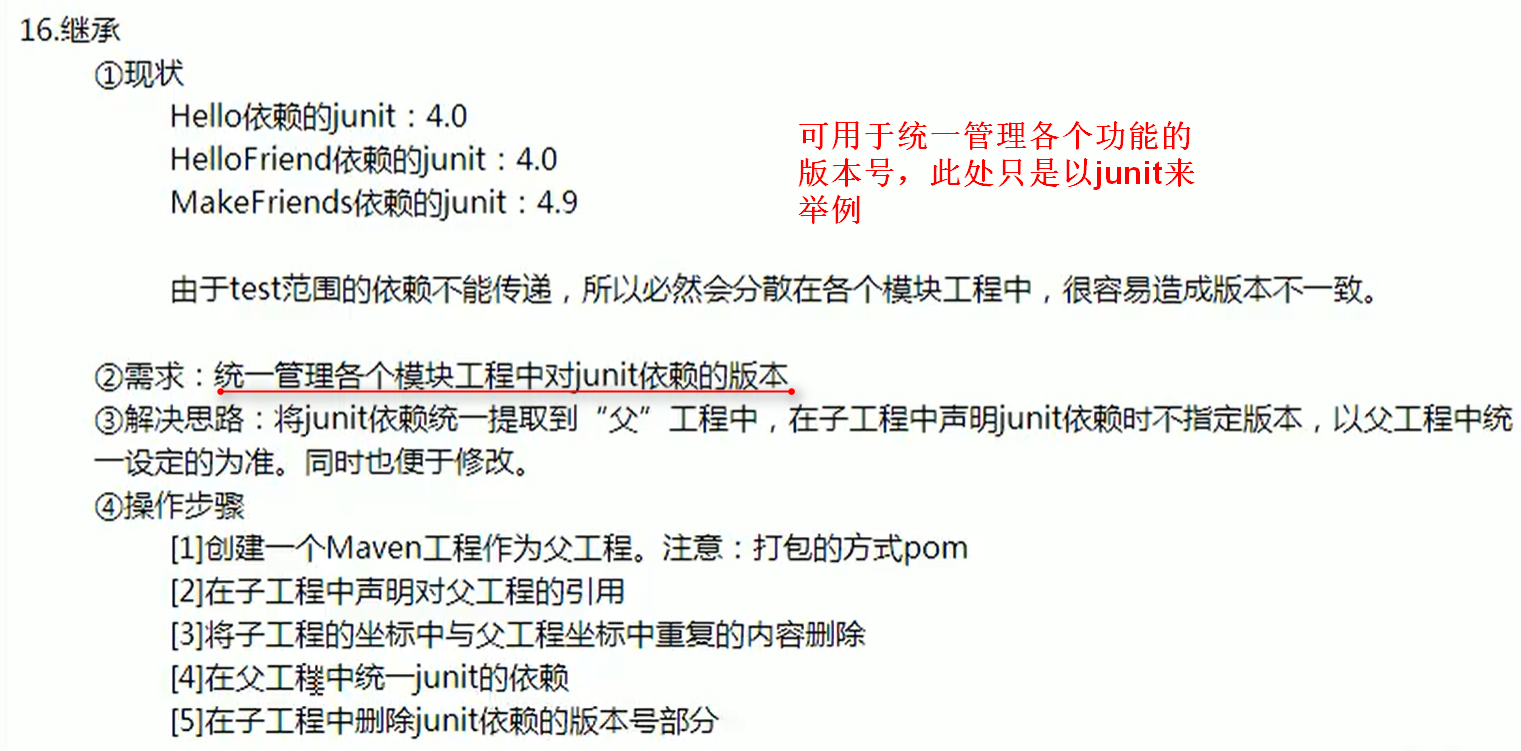
### 统一管理依赖的版本







## 继承



⑤注意：配置继承后，执行安装命令时要先安装父工程；使用聚合后，就不用担心这个问题了。

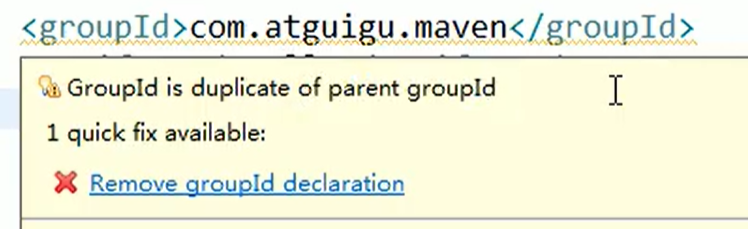
操作步骤：











此时在子工程中可以将groupId和version删除掉



在父工程中做以上配置，实现对版本的统一管理；

注意：<dependencyManagement>标签中只是声明各个依赖及版本号，子工程中如果用不到父工程中声明的依赖，子工程里并不会引入相关的jar包；子工程如果用到父工程中声明的依赖可以不用加版本号了；

## 聚合

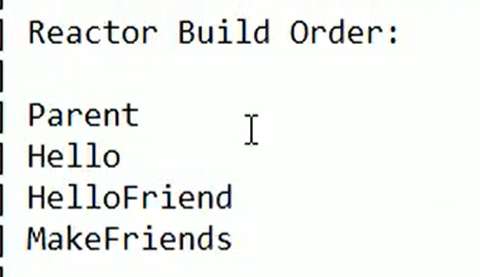


说明：此处打乱指定模块的顺序，还是能够成功的；

1、打乱顺序



2、控制台构建前输出的构建顺序



3、构建成功后控制台的输出

