# 1、maven项目分配，以淘淘商城为例说明

## 1、项目结构

继承：

依赖：

。。。。。。

前台工程

Taotao-parent

Taotao-common

Taotao-manager(pom工程)聚合工程

Taotao-controller(war包)

Taotao-service(jar包)

Taotao-mapper(jar包)

Taotao-pojo(jar包)

## 2、后台管理系统工程结构：

taotao-parent -- 管理依赖jar包的版本，全局，公司级别

|--taotao-common --- 通用组件、工具类

|--taotao-manage -- 后台系统

|-- taotao.manage.web

|-- taotao.manage.service

|-- taotao.manage.mapper

|-- taotao.manage.pojo

## 3、继承和依赖关系的处理

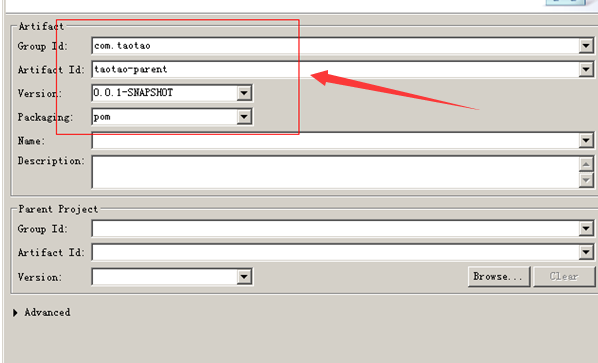
### 1、解释：

#### 1、父工程都是pom，web层是war，其余的依赖是jar

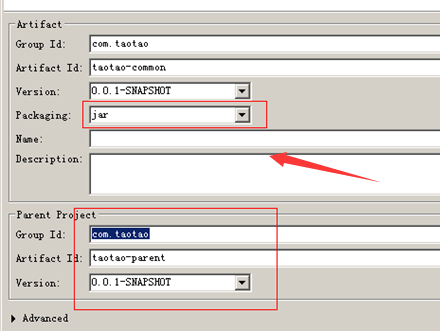
#### 2、groupId 都是com.taotao

#### 3、ArtifactId 为eclipse中各个项目的名字，下面图片中，可以查看详细信息

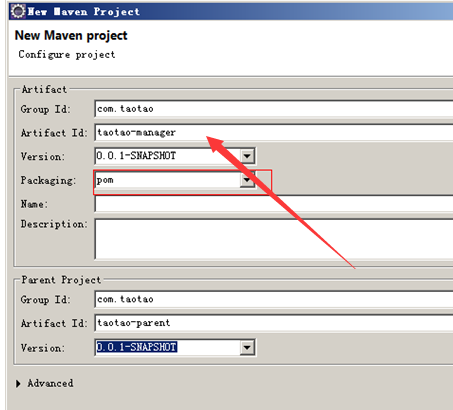
### 2、taotao-parent ->>> pom （全局）



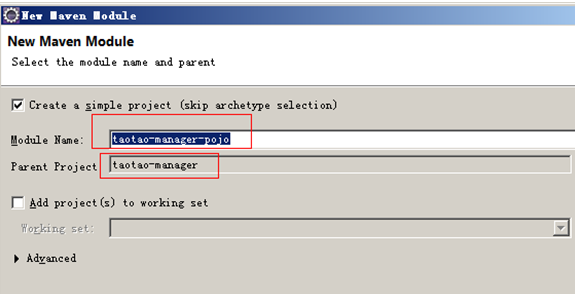
### 3、taotao-common --->jar包 （工具项目类，继承于parent）

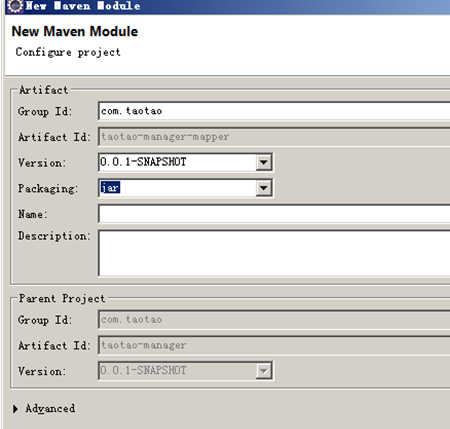


### 4、taotao-manage --->pom （继承于parent ，依赖common）



#### 1、com.taotao.manage.pojo --->jar（继承manage，不需要依赖）



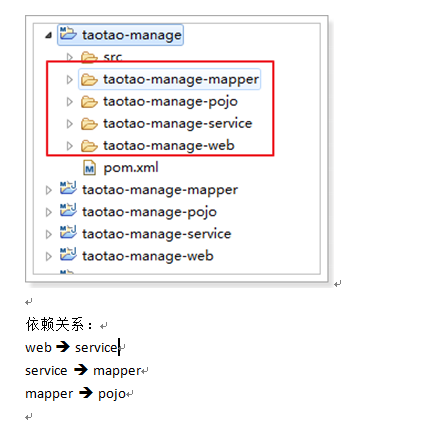


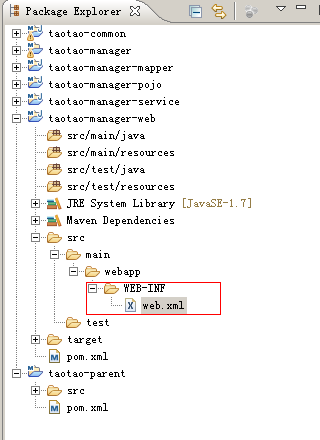
#### 2、com.taotao.manage.mapper -->jar（继承manage，依赖pojo ）

#### 3、com.taotao.manage.service -->jar（继承manage，依赖mapper）

#### 4、com.taotao.manage.web--->war包（继承manage，依赖mapper）

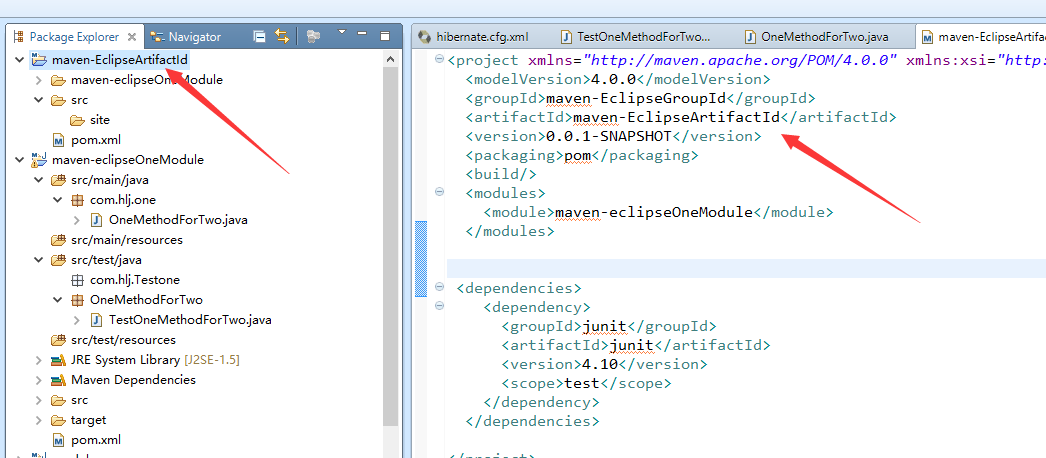
### 5、项目搭建完成的基本结构如下



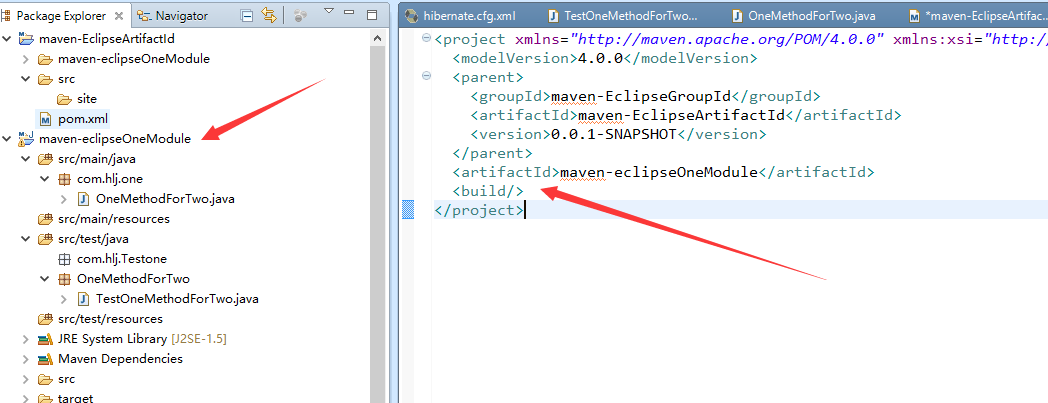


# 2、自己搭建的项目测试

## 1、建立一个父工程 maven-EclipseArtifactId



## 2、建立一个module 模块maven-eclipseOneModule

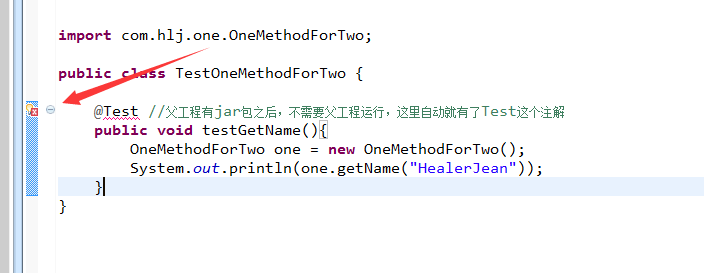


## 3、maven-eclipseOneModule 中添加一个java类

|  |
| --- |
| **public** **class** OneMethodForTwo {  **public** String getName(String name){    **return** name;  }  } |

## 4、maven-eclipseOneModule 中添加一个测试类，这个时候下面的test注解是错误的，不能使用，因为工程中没有junit依赖

|  |
| --- |
| **public** **class** TestOneMethodForTwo {    @Test  **public** **void** testGetName(){  OneMethodForTwo one = **new** OneMethodForTwo();  System.*out*.println(one.getName("HealerJean"));  }  } |



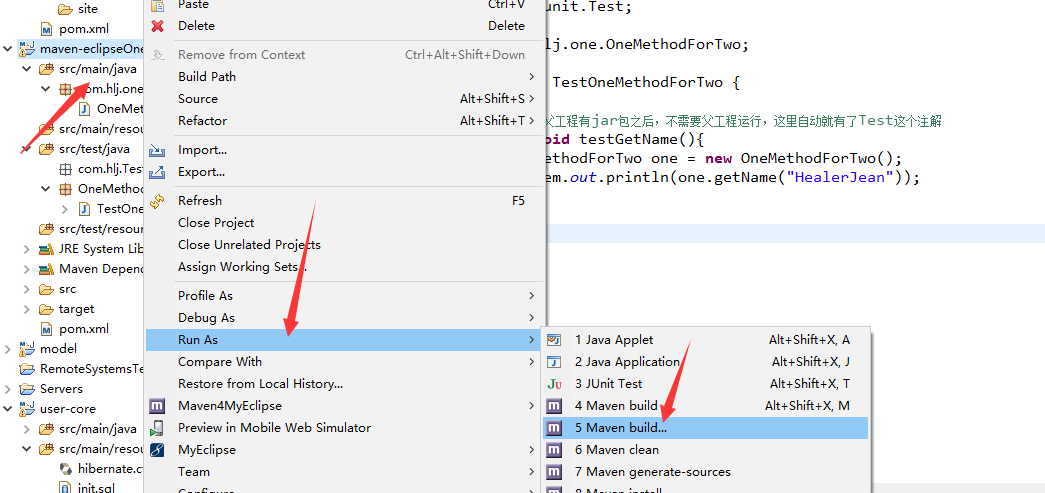
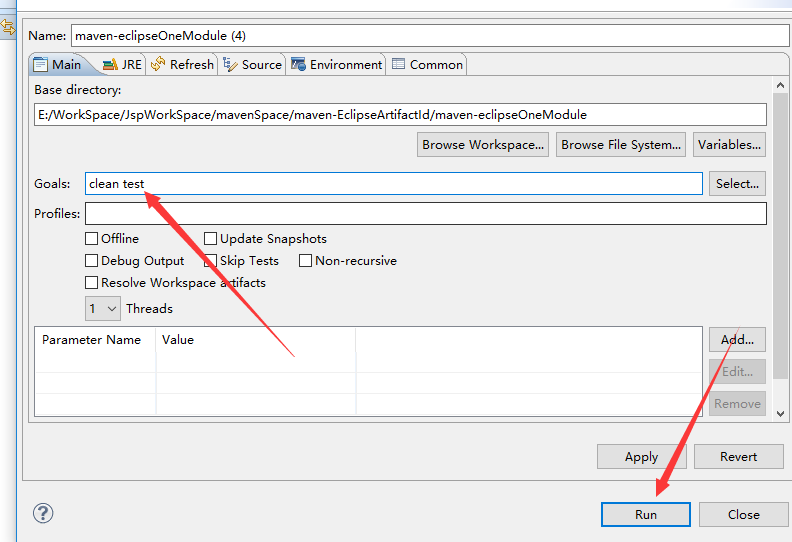
## 5、这个时候到父工程中添加一个junit依赖

|  |
| --- |
| <project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>maven-EclipseGroupId</groupId>  <artifactId>maven-EclipseArtifactId</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>pom</packaging>  <build/>  <modules>  <module>maven-eclipseOneModule</module>  </modules>    <dependencies>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.10</version>  <scope>test</scope>  </dependency>  </dependencies>    </project> |

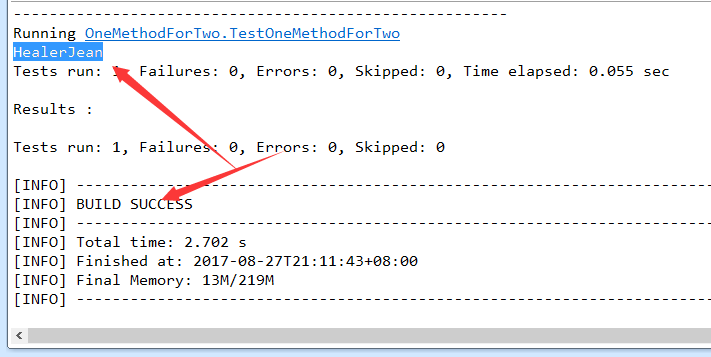
## 6、@Test错误消失，运行这个module 的test测试，使用build..进行执行方法，因为这样可以自己输入命令，这里输入clean test 这样就会执行

### 1、Run as build..

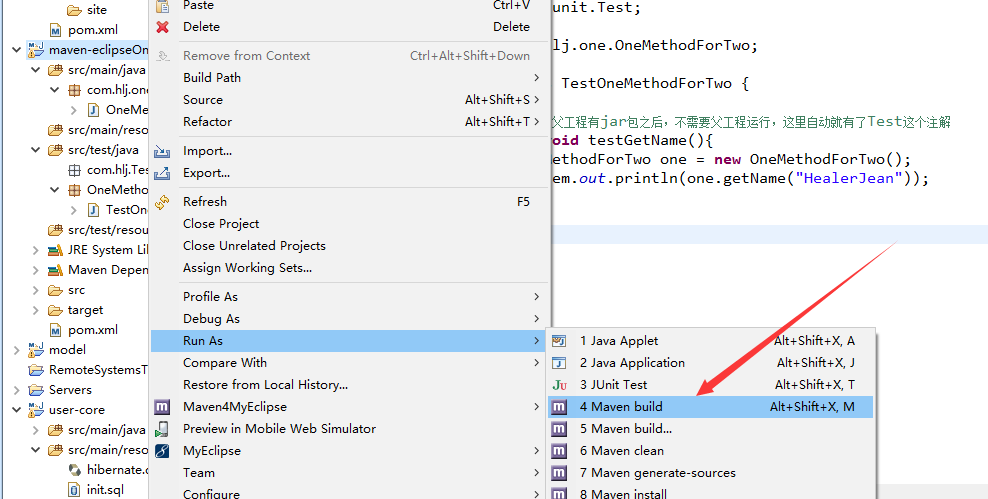
### 2、clean test

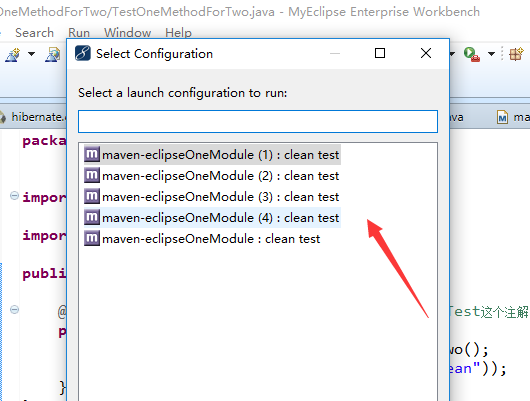
 

## 7、控制台输出 HealerJean 测试成功

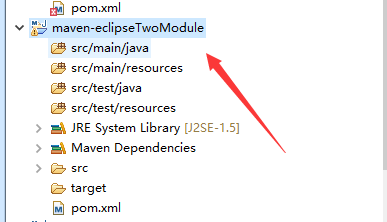


## 8、Maven build 用法，执行已经操作过的记录



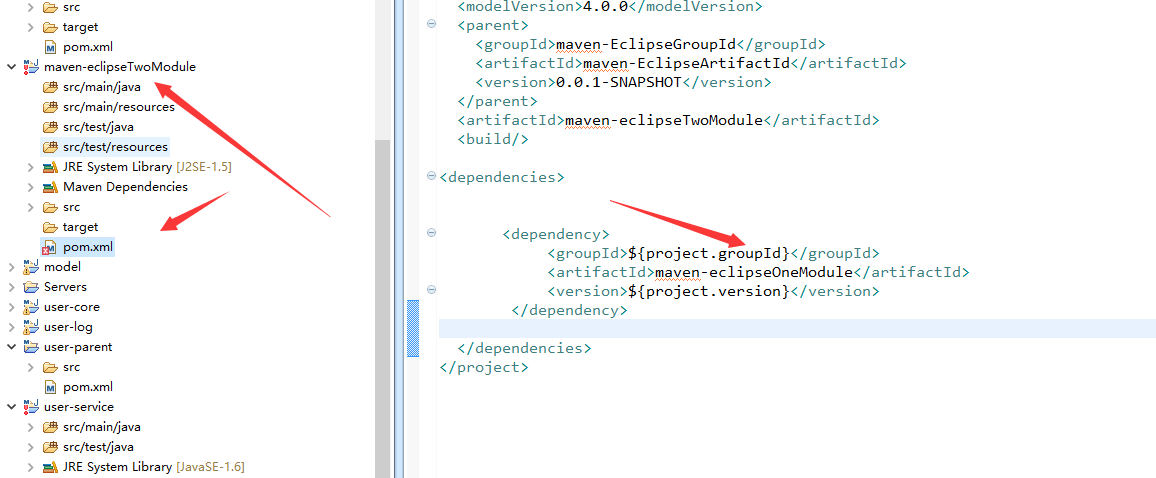


# 3、建立第二个module，之间相互调用



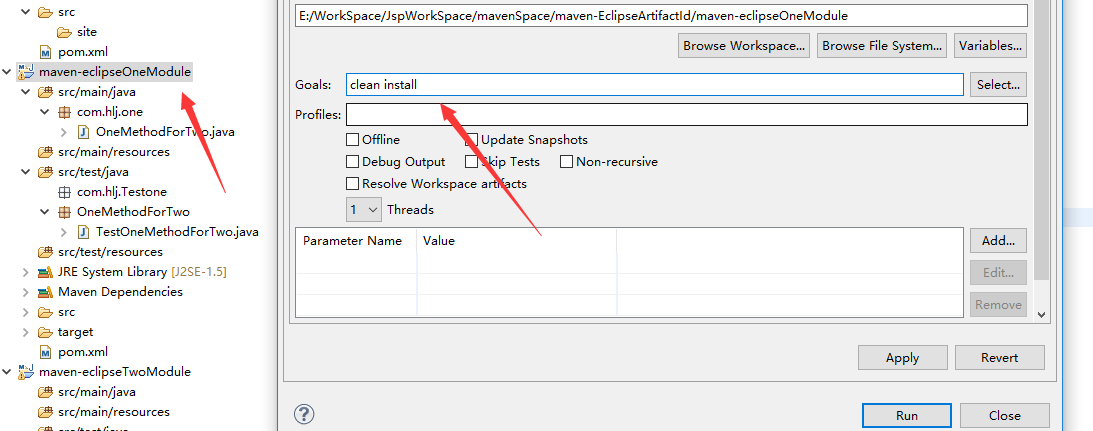
## 1、twoModule中添加one的依赖，发现pom项目报错

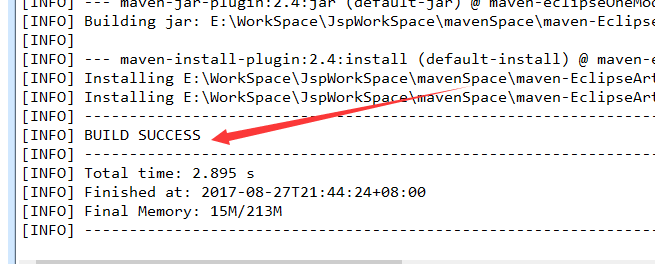
|  |
| --- |
| <dependency>  <groupId>${project.groupId}</groupId>  <artifactId>maven-eclipseOneModule</artifactId>  <version>${project.version}</version>  </dependency> |



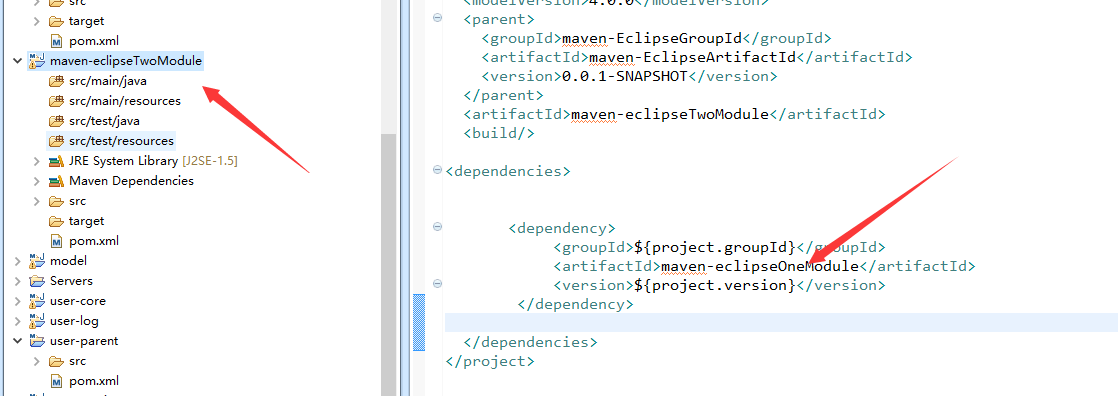
## 2、解决1中的问题，oneModule中先将jar包放入m2的依赖中去，也就是install oneModule

### clean install



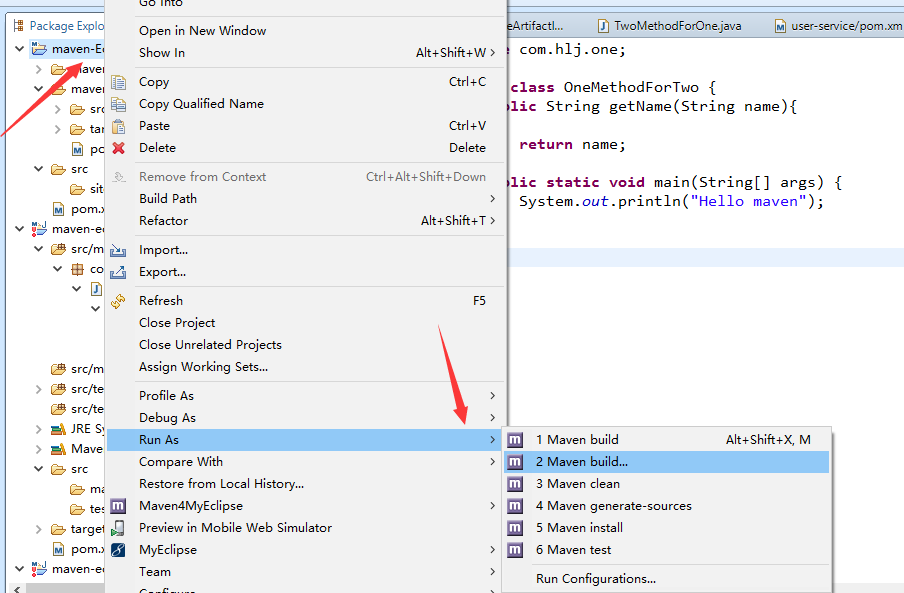


## 3、问题解决



## 4、项目之间的依赖就是这么解决的

# 4、主项目打包，oneModule和twoModule都在maven-EclipseArtifactId中，所以这样对这个项目打包就会帮助给以下两个打包了



# 4、依赖参数

## 1.、scope 以mysql举例，只有测试和运行的时候才会依赖，所以这里设置为runtime

## 1、mysql scope为runtime

|  |
| --- |
| <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>5.1.24</version>  <scope>runtime</scope>  </dependency> |

## 2、junit scope test ，只有在编译test时候才会依赖

|  |
| --- |
| <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.10</version>  <scope>test</scope>  </dependency> |

