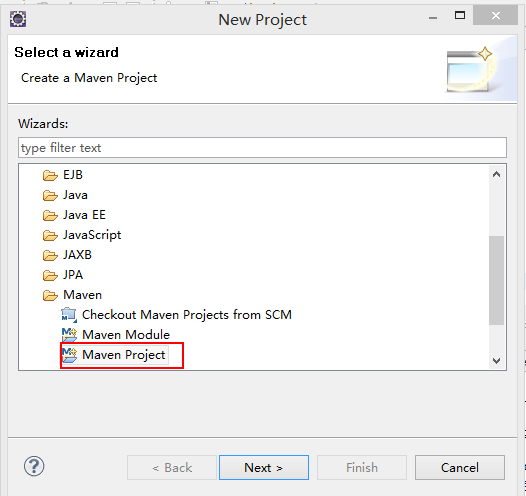
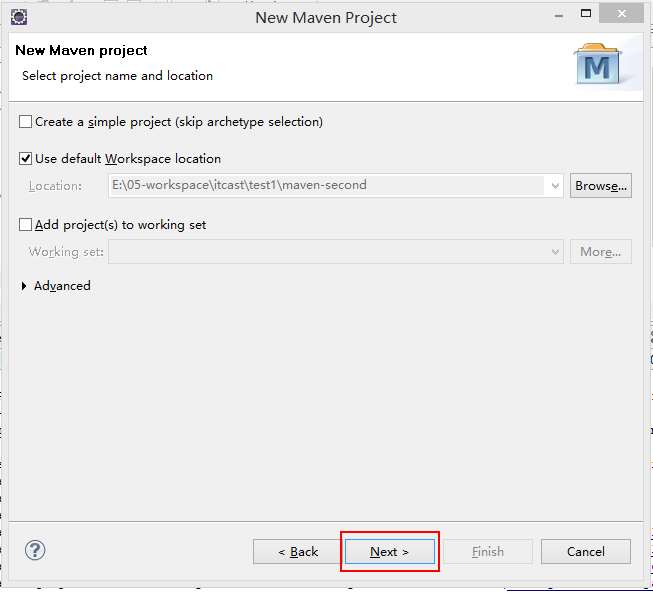
# 通过骨架创建maven工程

## 创建工程

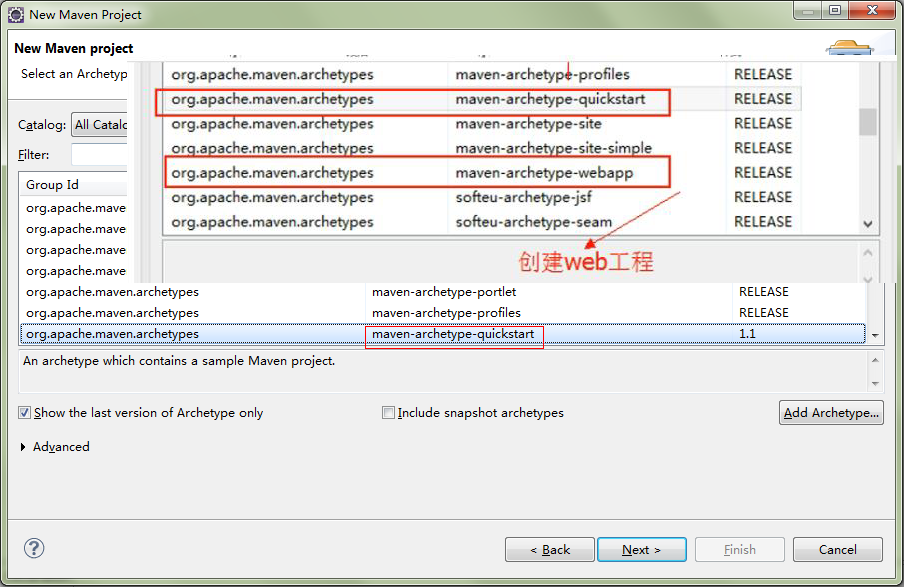
### 第一步：选择new→maven→Maven Project



### 第二步：next

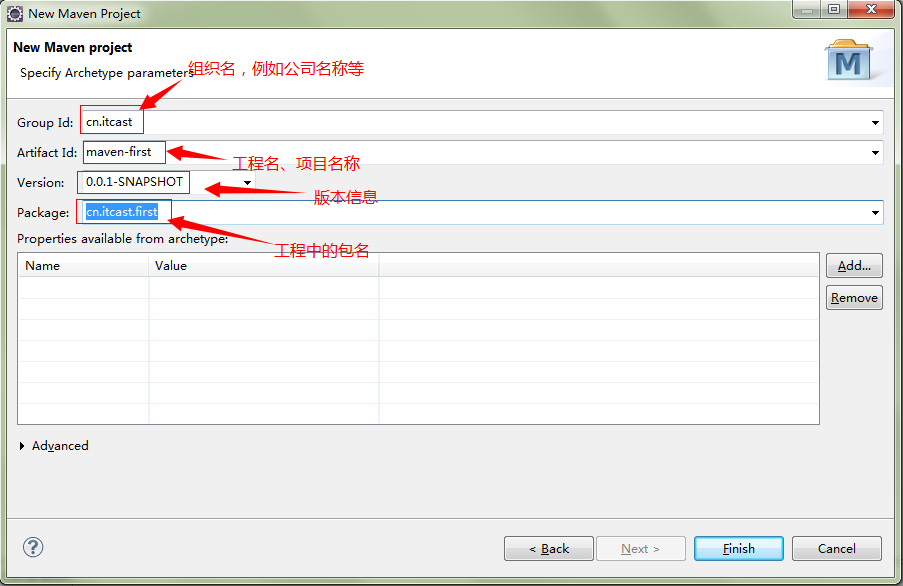


### 第三步：next

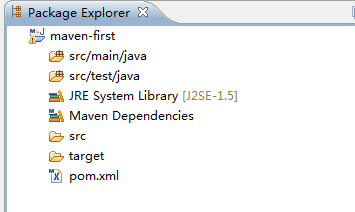


选择maven的工程骨架，这里我们选择quickstart（java工程）。 Webapp（web工程）

### 第四步：next



输入GroupId、ArtifactId、Version、Package信息点击finish完成。



## 编写代码

### 创建MavenFirst.java

在src/main/java中创建cn.itcast.maven包，然后创建MavenFirst.java

|  |
| --- |
| **package** cn.itcast.maven;  **public** **class** MavenFirst {  **public** String sayHello(String name) {  **return** "hello " + name;  }  } |

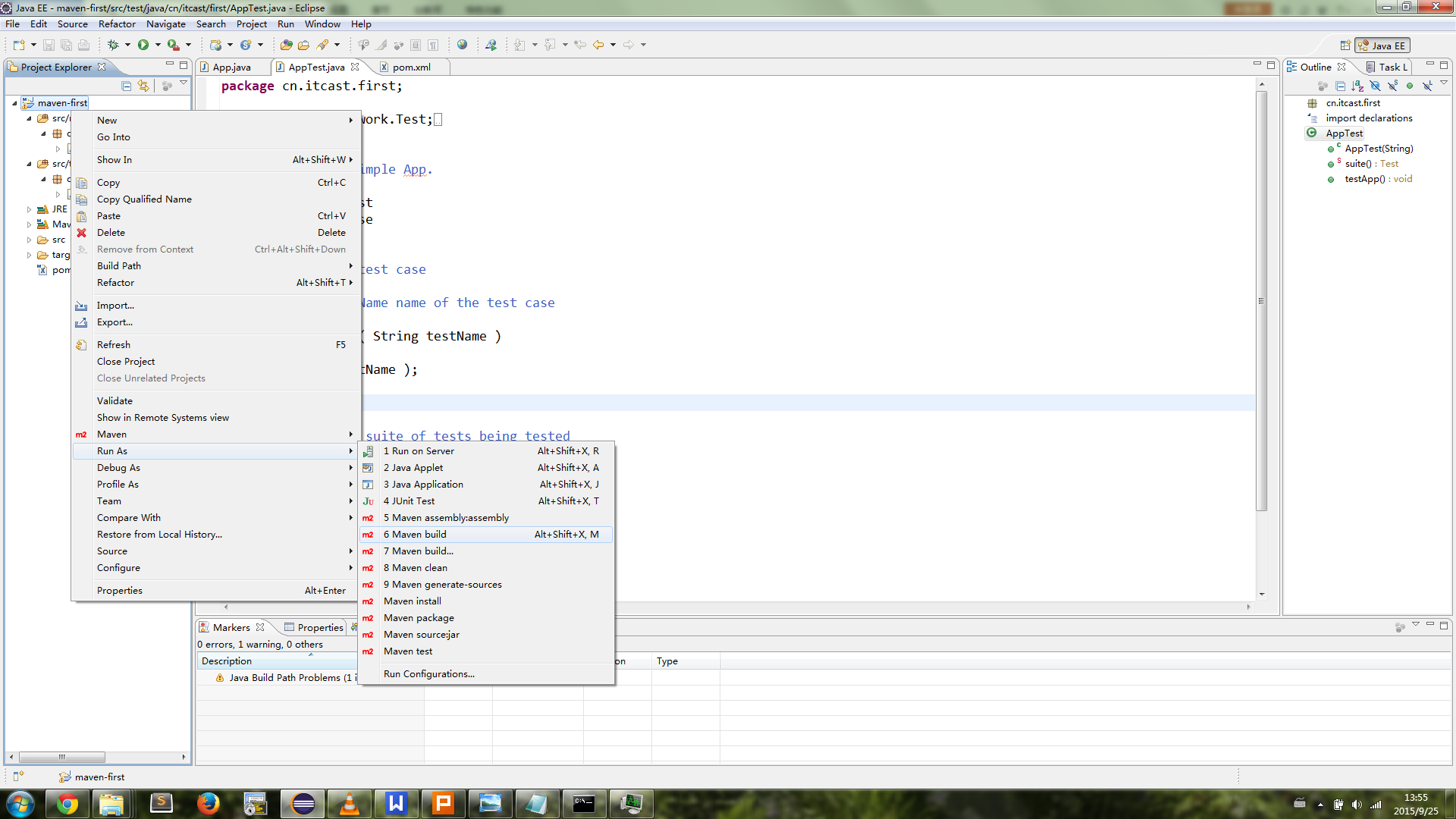
### 创建TestMavenFirst.java

在src/test/java中创建cn.itcast.maven包，然后创建TestMavenFirst.java

|  |
| --- |
| **package** cn.itcast.maven;  **import** org.junit.Assert;  **import** org.junit.Test;  **public** **class** TestMavenFirst {  @Test  **public** **void** testSayHello() {  MavenFirst first = **new** MavenFirst();  String result = first.sayHello("zhangsan");  Assert.*assertEquals*("hello zhangsan", result);  }  } |

## 进行测试

在Eclipse的maven插件中执行maven命令，需要在maven工程或者pom.xml文件上点击右键，选择Run as→maven build.. test

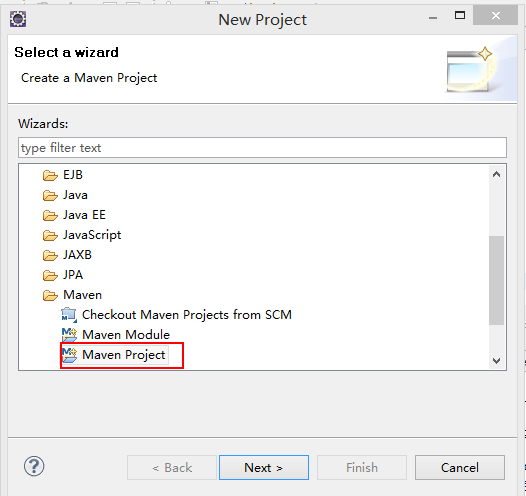


# 不通过骨架创建maven工程

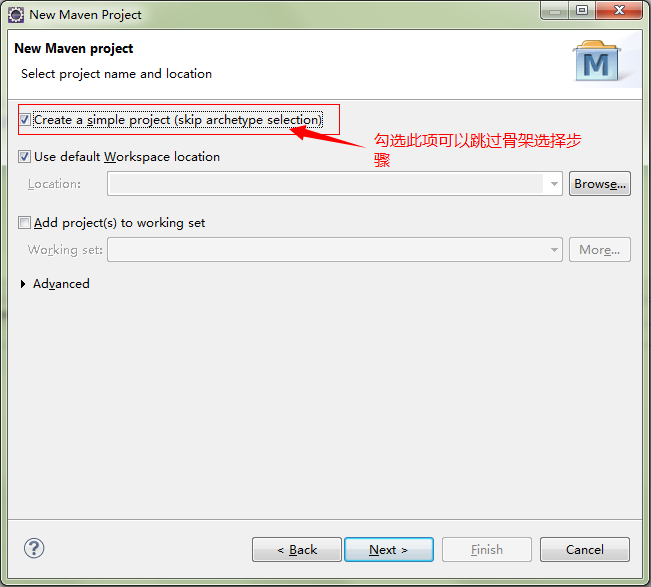
通过选择骨架创建maven工程，每次选择骨架时都需要联网下载，如果网络不通或者较慢的情况下会有很长时间的等待。使用很是不方便，所以创建工程时可以不选择骨架直接创建工程。

## 创建工程

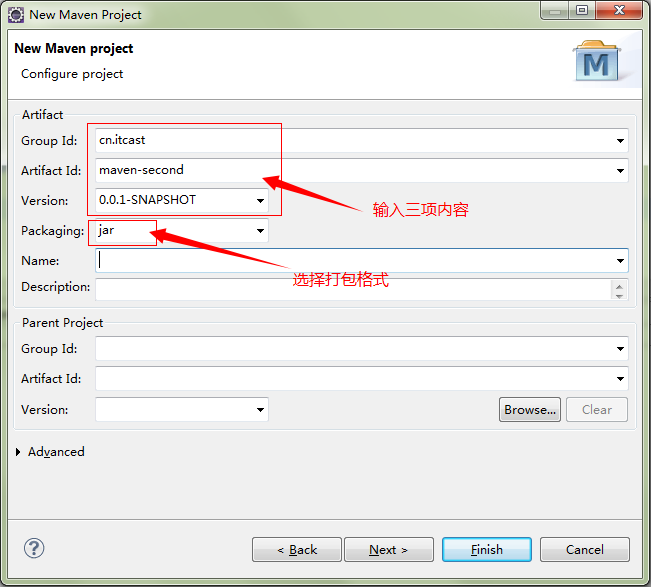
### 第一步：选择new→maven→Maven Project



### 第二步：next



### 第三步：next



Packaging：指定打包方式，默认为jar。选项有：jar、war、pom。

### 第四步：点击finish，完成maven工程创建。

### 第五步： 修改pom文件

在Maven-second工程中依赖使用maven-first工程的代码

|  |
| --- |
| <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>cn.itcast.maven</groupId>  <artifactId>maven-second</artifactId>  <version>0.0.1-SNAPSHOT</version>  <dependencies>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.12</version>  </dependency>  <dependency>  <groupId>cn.itcast.maven</groupId>  <artifactId>maven-first</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  </dependencies>  </project> |

## 编写代码

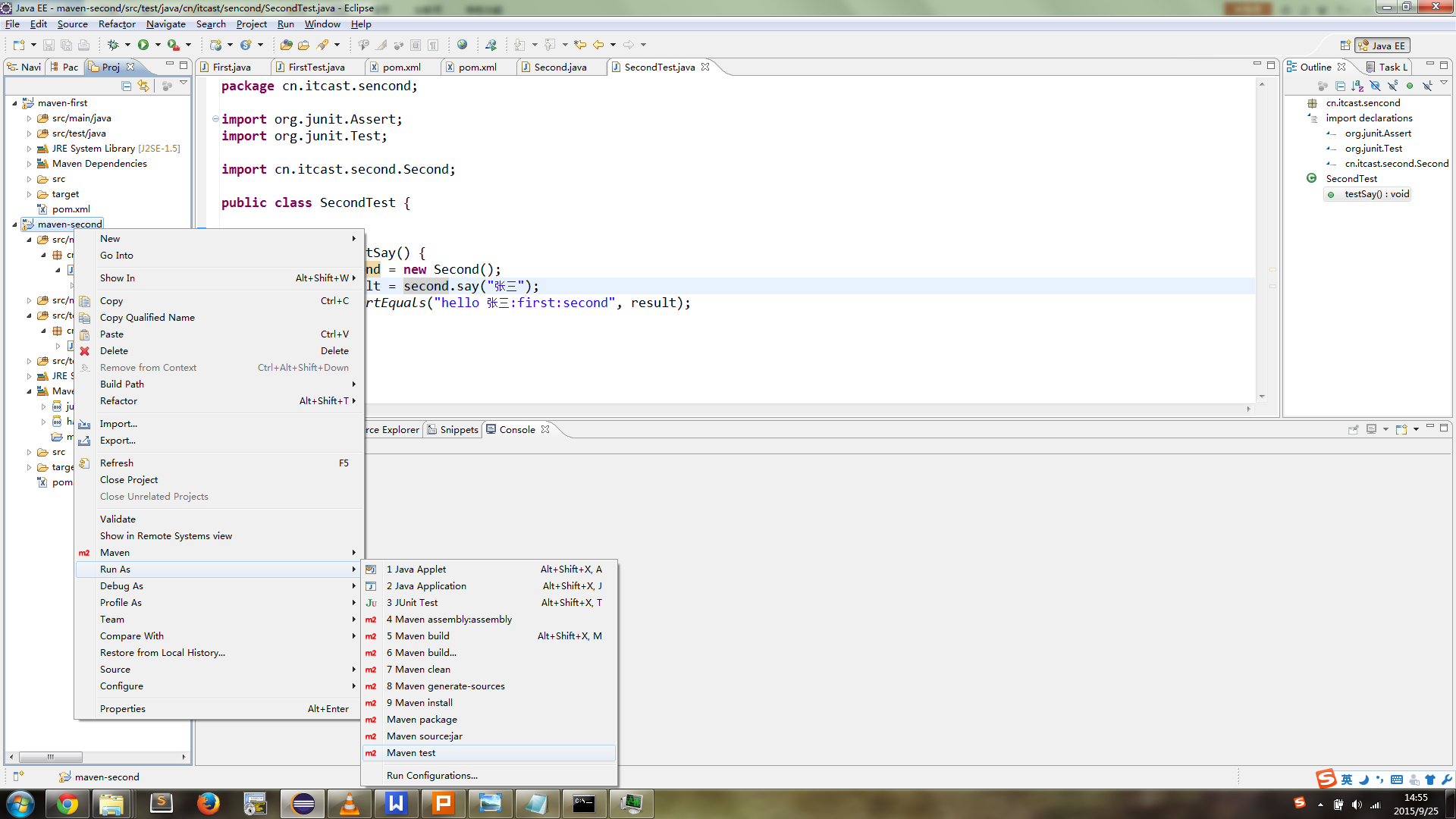
### 创建MavenSecond.java

|  |
| --- |
| **package** cn.itcast.maven;  **public** **class** MavenSecond {    **public** String sayHello(String name) {  MavenFirst first = **new** MavenFirst();  **return** first.sayHello(name) + ":second";  }  } |

### 创建TestMavenSecond.java

|  |
| --- |
| **package** cn.itcast.maven;  **import** org.junit.Assert;  **import** org.junit.Test;  **public** **class** TestMavenSecond {  @Test  **public** **void** testSayHello() {  MavenSecond second = **new** MavenSecond();  String result = second.sayHello("zhangsan");  Assert.*assertEquals*("hello zhangsan:second", result);  }  } |

## 测试工程



如果maven-first工程没有安装则会出现以下错误：

|  |
| --- |
| [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building maven-second 0.0.1-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [WARNING] The POM for cn.itcast:maven-first:jar:0.0.1-SNAPSHOT is missing, no dependency information available  [INFO] ------------------------------------------------------------------------  [INFO] BUILD FAILURE  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 0.218s  [INFO] Finished at: Fri Sep 25 15:06:00 CST 2015  [INFO] Final Memory: 4M/15M  [INFO] ------------------------------------------------------------------------  [ERROR] Failed to execute goal on project maven-second: Could not resolve dependencies for project cn.itcast:maven-second:jar:0.0.1-SNAPSHOT: **Could not find artifact cn.itcast:maven-first:jar:0.0.1-SNAPSHOT** -> [Help 1]  [ERROR]  [ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.  [ERROR] Re-run Maven using the -X switch to enable full debug logging.  [ERROR]  [ERROR] For more information about the errors and possible solutions, please read the following articles:  [ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/DependencyResolutionException |

提示找不到maven-first的jar包。当系统运行时是从本地仓库中找依赖的jar包的，所以必须先将maven-first安装才能正常运行，需要在maven-first工程上运行 mvn install命令安装到本地仓库。

# Eclipse命令

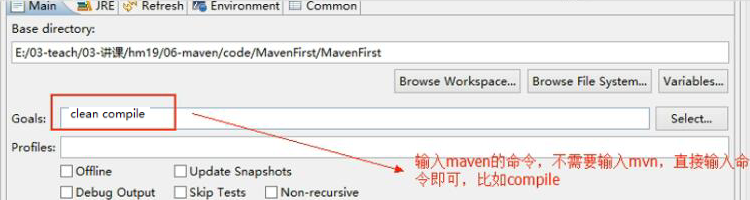
## 自带命令

maven build和maven build... 并不是maven的命令。

maven build...只是提供一个命令输入功能，可以在此功能中输入自定义的maven命令。

maven build的功能就是执行上次自定义命令。

maven build...创建自定义命令



## 传统命令

Maven clean

Maven install

Maven package

Maven test

# [java项目和java-web项目](http://www.cnblogs.com/nelson-hu/p/7163736.html)

## 概念

1. java项目

　　.project:是工程构建配置文件

　　.classpath:保存的是项目所用的外部引用包的路径

　　.settings：记录项目配置变化的记录文件夹

　　src:sourcefolder项目源代码.java文件的存放位置

bin：项目开发时的classes文件存放目录：项目编译后.class文件的存放位置

注： 右键点击项目名称 - buildpath - source标签：编辑项目的sourcefolder和outputfolder

2. java - web项目

　　.project:是工程构建配置文件

　　.classpath:保存的是项目所用的外部引用包的路径

　　.settings：记录项目配置变化的记录文件夹

　　src:sourcefolder项目源代码.java文件的存放位置

　　build/classes：项目开发时的classes文件存放目录：既项目编译后.class文件的存放位置

　　webcontent/web-info/classes:项目发布到tomcat时， .class文件的存放目录

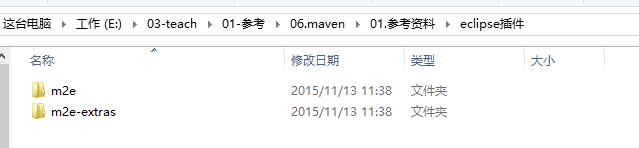
## 唯一区别



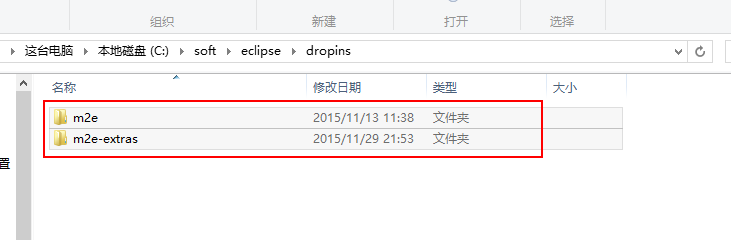
# M2Eclipse插件(了解)

对于底版本的M2Eclipse是eclipse中的maven插件，高版本自带

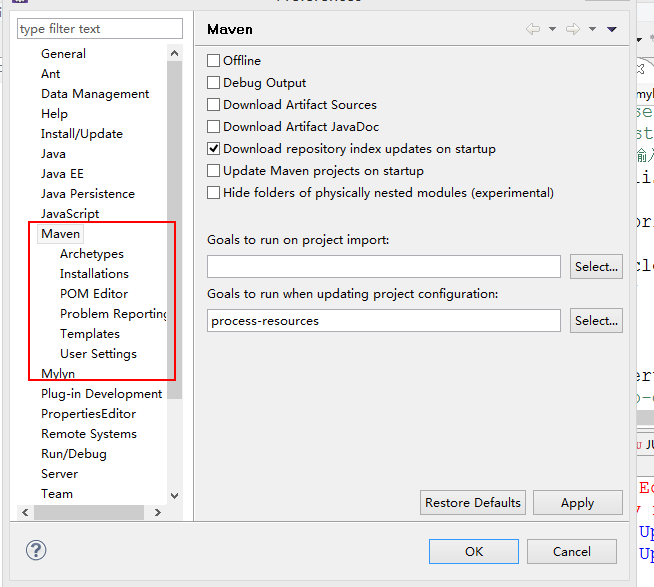
## 第一步：将以下包中的插件进行复制。



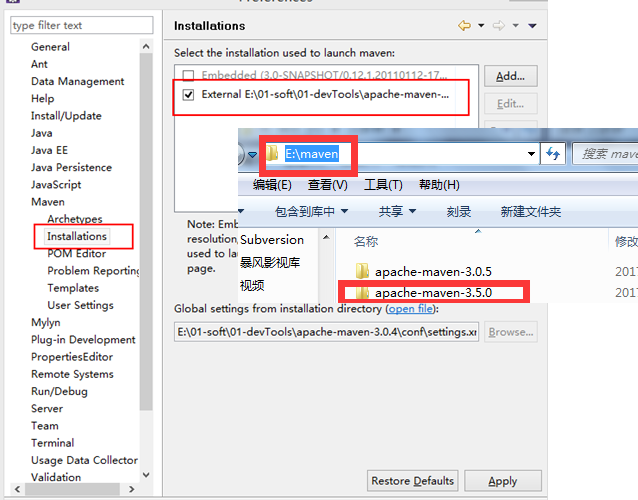
## 第二步：粘贴到eclipse中的dropins目录中



## 第三步：查看eclipse中是否有maven插件



## 第四步：设置maven安装目录



## 第五步：设置用户配置

