

Maven学习总结(五)——聚合与继承 - 孤傲苍狼

一、聚合

如果我们想一次构建多个项目模块，那我们就需要对多个项目模块进行聚合

1.1、聚合配置代码

```
1 <modules>
2     <module>模块一</module>
3     <module>模块二</module>
4     <module>模块三</module>
5 </modules>
```

例如：对项目的Hello、HelloFriend、MakeFriends这三个模块进行聚合

```
1 <modules>
2     <module>../Hello</module>
3     <module>../HelloFriend</module>
4     <module>../MakeFriends</module>
5 </modules>
```

其中module的路径为相对路径。

二、继承

继承为了消除重复，我们把很多相同的配置提取出来，例如：groupId, version等

2.1、继承配置代码

```
1 <parent>
2     <groupId>me.gacl.maven</groupId>
3     <artifactId>ParentProject</artifactId>
4     <version>0.0.1-SNAPSHOT</version>
5     <relativePath>../ParentProject/pom.xml</relativePath>
6 </parent>
```

2.2、继承代码中定义属性

继承代码过程中，可以定义属性，例如：

```
1 <properties>
2     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
3     <junit.version>4.9</junit.version>
```

```
4     <maven.version>0.0.1-SNAPSHOT</maven.version>
5 </properties>
```

访问属性的方式为\${junit.version}，例如：

```
1 <dependency>
2     <groupId>junit</groupId>
3     <artifactId>junit</artifactId>
4     <version>${junit.version}</version>
5     <scope>test</scope>
6 </dependency>
```

2.3、父模块用dependencyManagement进行管理



```
1 <dependencyManagement>
2     <dependencies>
3         <dependency>
4             <groupId>junit</groupId>
5             <artifactId>junit</artifactId>
6             <version>${junit.version}</version>
7             <scope>test</scope>
8         </dependency>
9         <dependency>
10            <groupId>cn.itcast.maven</groupId>
11            <artifactId>HelloFriend</artifactId>
12            <version>${maven.version}</version>
13            <type>jar</type>
14            <scope>compile</scope>
15        </dependency>
16    </dependencies>
17 </dependencyManagement>
```



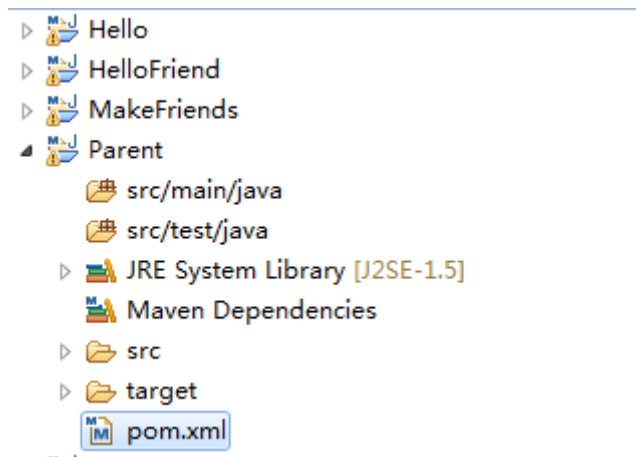
这样的好处是子模块可以有选择行的继承，而不需要全部继承。

三、聚合与继承的关系

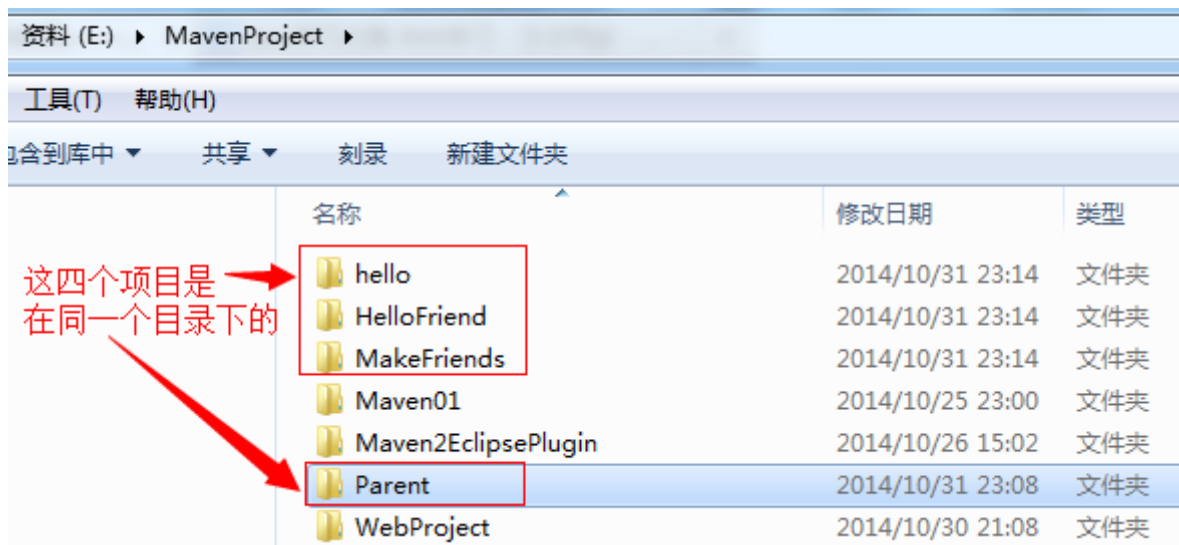
聚合主要为了快速构建项目，继承主要为了消除重复

四、聚合与继承实战演练

创建四个Maven项目，如下图所示：



这四个项目放在同一个目录下，方便后面进行聚合和继承



Parent项目是其它三个项目的父项目，主要是用来配置一些公共的配置，其它三个项目再通过继承的方式拥有Parent项目中的配置，首先配置Parent项目的pom.xml，添加对项目的Hello、HelloFriend、MakeFriends这三个模块进行聚合以及jar包依赖，pom.xml的配置信息如下：

Parent项目的pom.xml配置



```
1 <project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
  4.0.0.xsd">
3     <modelVersion>4.0.0</modelVersion>
4
5     <groupId>me.gacl.maven</groupId>
6     <artifactId>Parent</artifactId>
7     <version>0.0.1-SNAPSHOT</version>
8     <packaging>pom</packaging>
9
10    <name>Parent</name>
11    <url>http://maven.apache.org</url>
12
13    <!-- 对项目的Hello、HelloFriend、MakeFriends这三个模块进行聚合 -->
```

```

14 <modules>
15     <module>../Hello</module>
16     <module>../HelloFriend</module>
17     <module>../MakeFriends</module>
18 </modules>
19
20 <!-- 定义属性 -->
21 <properties>
22     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23     <junit.version>4.9</junit.version>
24     <maven.version>0.0.1-SNAPSHOT</maven.version>
25 </properties>
26
27 <!-- 用dependencyManagement进行jar包依赖管理 -->
28 <dependencyManagement>
29     <!-- 配置jar包依赖 -->
30     <dependencies>
31         <dependency>
32             <groupId>junit</groupId>
33             <artifactId>junit</artifactId>
34             <!-- 访问junit.version属性 -->
35             <version>${junit.version}</version>
36             <scope>test</scope>
37         </dependency>
38         <dependency>
39             <groupId>me.gacl.maven</groupId>
40             <artifactId>Hello</artifactId>
41             <!-- 访问maven.version属性 -->
42             <version>${maven.version}</version>
43             <scope>compile</scope>
44         </dependency>
45         <dependency>
46             <groupId>me.gacl.maven</groupId>
47             <artifactId>HelloFriend</artifactId>
48             <!-- 访问maven.version属性 -->
49             <version>${maven.version}</version>
50         </dependency>
51     </dependencies>
52 </dependencyManagement>
53 </project>

```



在Hello项目的pom.xml中继承Parent项目的pom.xml配置



```

1 <project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
3
4 <modelVersion>4.0.0</modelVersion>
5 <artifactId>Hello</artifactId>
6
7 <!-- 继承Parent项目中的pom.xml配置 -->
8 <parent>
9 <groupId>me.gacl.maven</groupId>
10 <artifactId>Parent</artifactId>
11 <version>0.0.1-SNAPSHOT</version>
12 <!-- 使用相对路径 -->
13 <relativePath>../Parent/pom.xml</relativePath>
14 </parent>
15
16 <dependencies>
17 <dependency>
18 <groupId>junit</groupId>
19 <artifactId>junit</artifactId>
20 </dependency>
21 </dependencies>
22 </project>

```



在HelloFriend项目的pom.xml中继承Parent项目的pom.xml配置



```

1 <project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
3 <modelVersion>4.0.0</modelVersion>
4 <artifactId>HelloFriend</artifactId>
5 <name>HelloFriend</name>
6
7 <!-- 继承Parent项目中的pom.xml配置 -->
8 <parent>
9 <groupId>me.gacl.maven</groupId>
10 <artifactId>Parent</artifactId>
11 <version>0.0.1-SNAPSHOT</version>
12 <relativePath>../Parent/pom.xml</relativePath>
13 </parent>
14 <dependencies>
15 <dependency>

```

```

16      <!-- Parent项目的pom.xml文件配置中已经指明了要使用的Junit的版本号，因此在这里添
      加junit的依赖时，
17      可以不指明<version></version>和<scope>test</scope>，会直接从Parent项目的pom.xml
      继承 -->
18      <groupId>junit</groupId>
19      <artifactId>junit</artifactId>
20  </dependency>
21  <!-- HelloFriend项目中使用到了Hello项目中的类，因此需要添加对Hello.jar的依赖
22  Hello.jar的<version>和<scope>也已经在Parent项目的pom.xml文件配置中已经指明了
23  因此这里也可以省略不写了
24  -->
25  <dependency>
26      <groupId>me.gacl.maven</groupId>
27      <artifactId>Hello</artifactId>
28  </dependency>
29  </dependencies>
30 </project>

```



在MakeFriends项目的pom.xml中继承Parent项目的pom.xml配置



```

1 <project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
  4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4   <artifactId>MakeFriends</artifactId>
5   <!-- 继承Parent项目中的pom.xml配置 -->
6   <parent>
7       <groupId>me.gacl.maven</groupId>
8       <artifactId>Parent</artifactId>
9       <version>0.0.1-SNAPSHOT</version>
10      <relativePath>../Parent/pom.xml</relativePath>
11  </parent>
12  <dependencies>
13      <dependency>
14          <!-- Parent项目的pom.xml文件配置中已经指明了要使用的Junit的版本号，因此在这里添加
              junit的依赖时，
15          可以不指明<version></version>和<scope>test</scope>，会直接从Parent项目的pom.xml
              继承 -->
16          <groupId>junit</groupId>
17          <artifactId>junit</artifactId>
18      </dependency>
19      <dependency>
20          <!-- MakeFriends项目中使用到了HelloFriend项目中的类，因此需要添加对HelloFriend.jar的

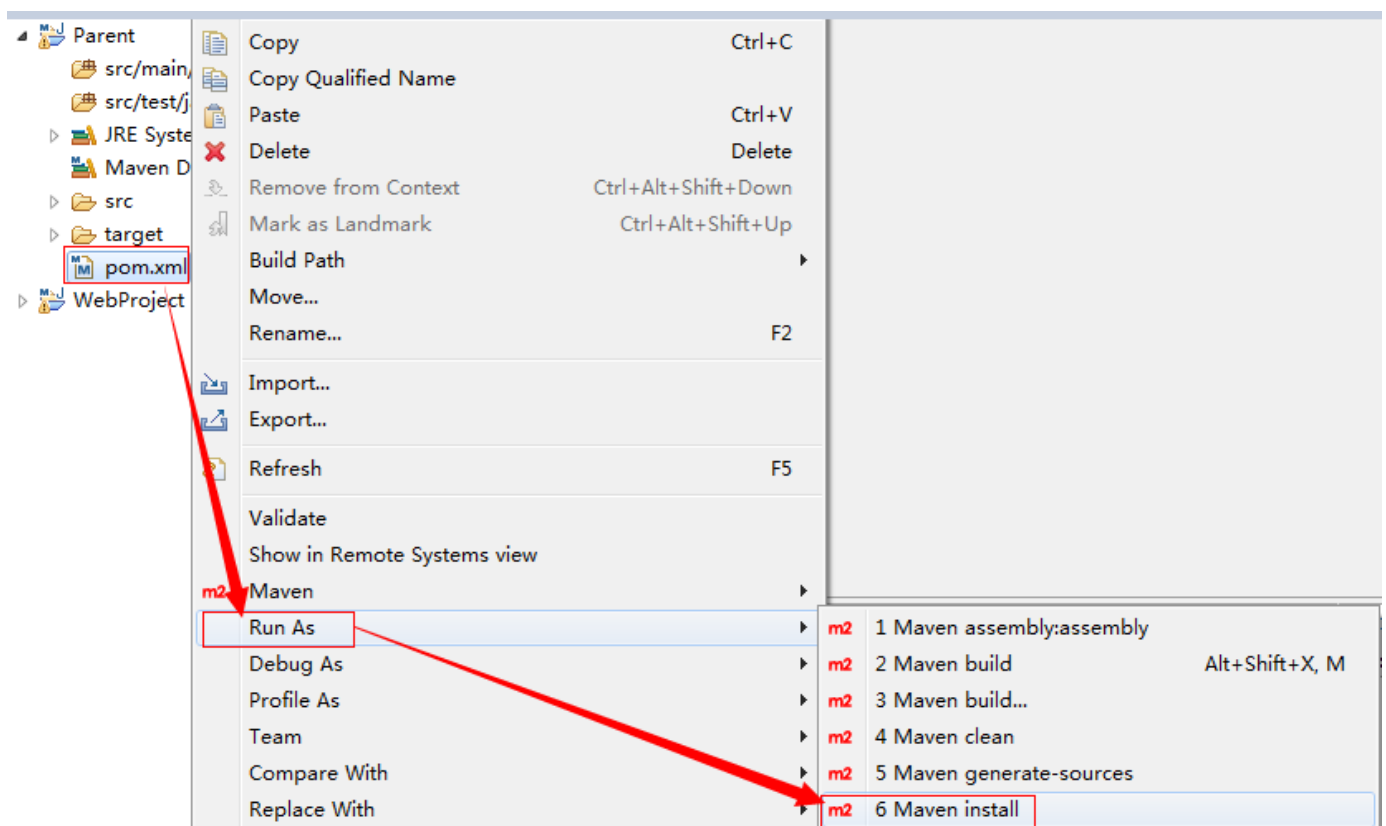
```

依赖

```
21      HelloFriend.jar的<version>和<scope>也已经在Parent项目的pom.xml文件配置中已经指明了
22      因此这里也可以省略不写了
23      -->
24      <groupId>me.gacl.maven</groupId>
25      <artifactId>HelloFriend</artifactId>
26    </dependency>
27  </dependencies>
28 </project>
```



以上的四个项目的pom.xml经过这样的配置之后，就完成了在Parent项目中聚合Hello、HelloFriend、MakeFriends这三个子项目(子模块)，而Hello、HelloFriend、MakeFriends这三个子项目(子模块)也继承了Parent项目中的公共配置，这样就可以使用Maven一次性构建所有的项目了，如下图所示：

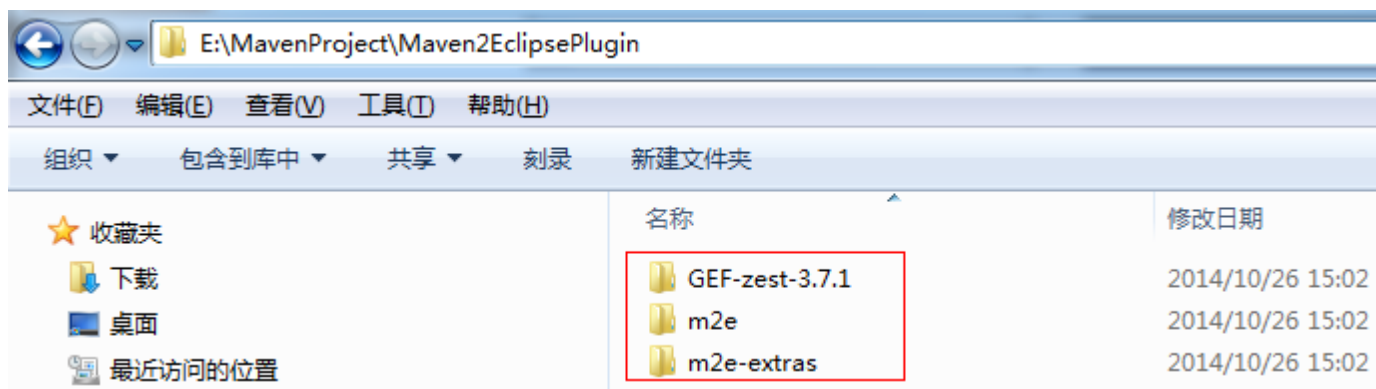


选中Parent项目的pom.xml文件→【Run As】→【Maven install】，这样Maven就会一次性同时构建Parent、Hello、HelloFriend、MakeFriends这四个项目，如下图所示：

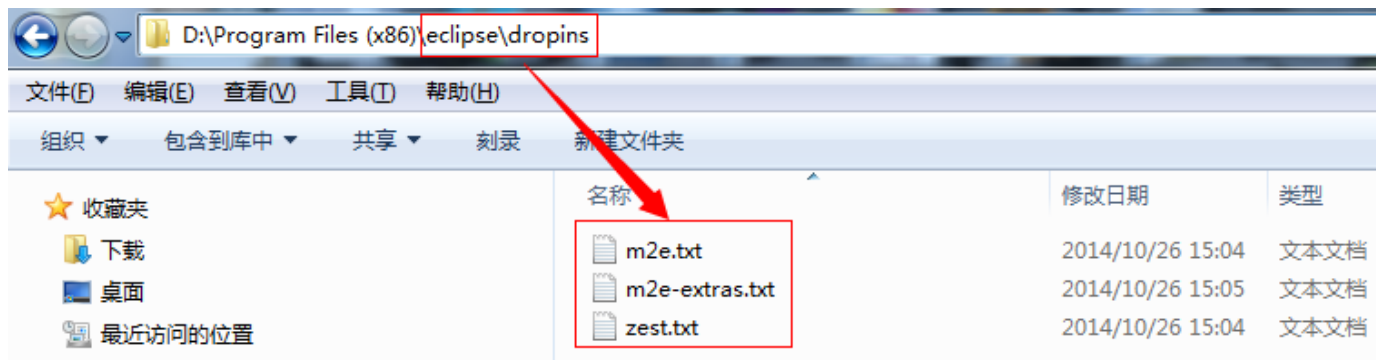
```
[INFO] Parent ..... SUCCESS [ 0.372 s]
[INFO] Hello ..... SUCCESS [ 1.935 s]
[INFO] HelloFriend ..... SUCCESS [ 0.473 s]
[INFO] MakeFriends ..... SUCCESS [ 0.459 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

一、安装Maven插件

下载下来的maven插件如下图所示：，插件存放的路径是：E:/MavenProject/Maven2EclipsePlugin



1. 进入到eclipse中的dropins目录下，新建三个txt文件（zest.txt, m2e.txt, m2e-extras.txt）。如下图所示：

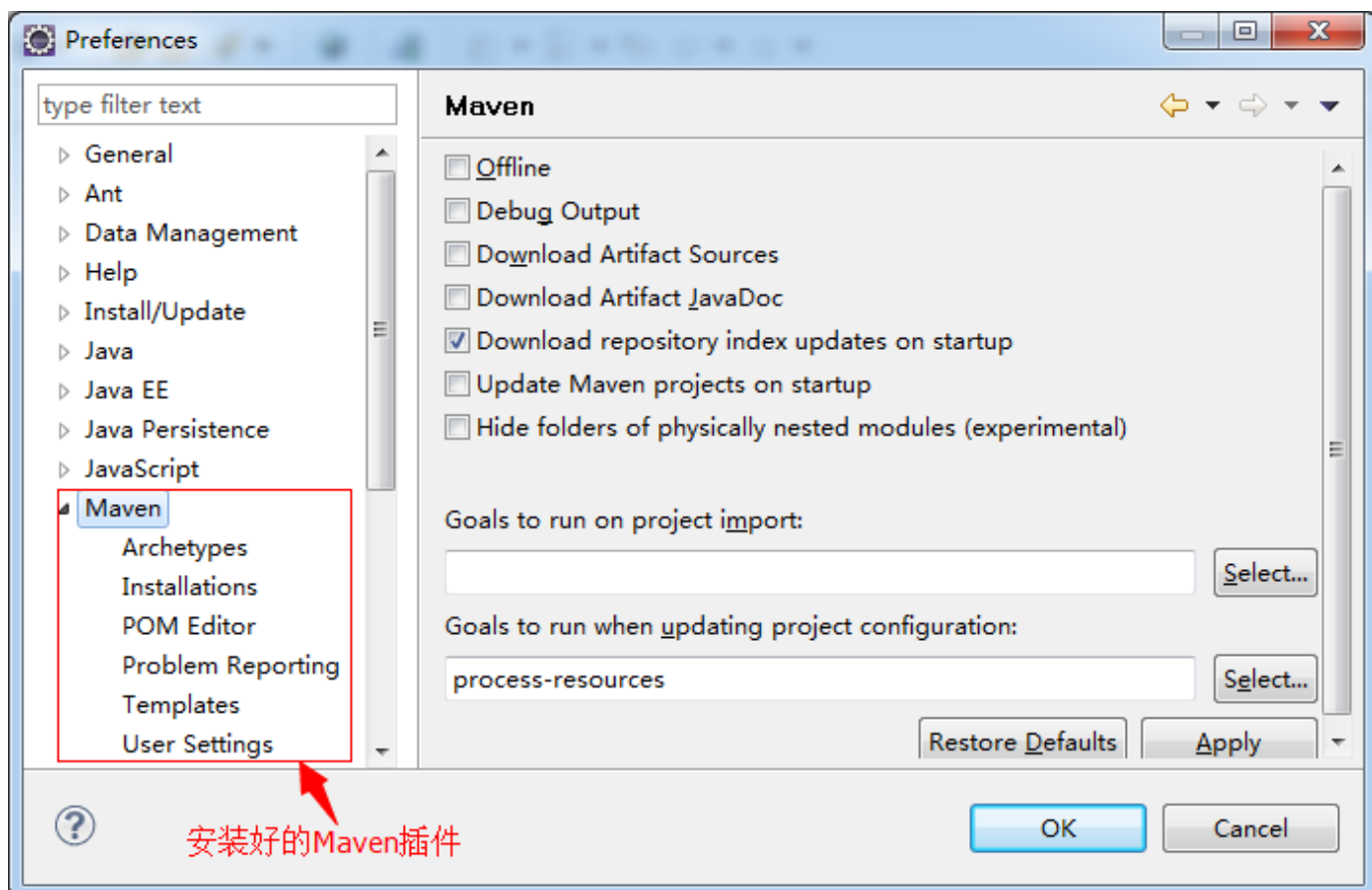


m2e.txt文件的内容如下：path=E:/MavenProject/Maven2EclipsePlugin/m2e m2e-extras.txt文件的内容如下：path=E:/MavenProject/Maven2EclipsePlugin/m2e-extras zest.txt文件的内容如下：

path=E:/MavenProject/Maven2EclipsePlugin/GEF-zest-3.7.1

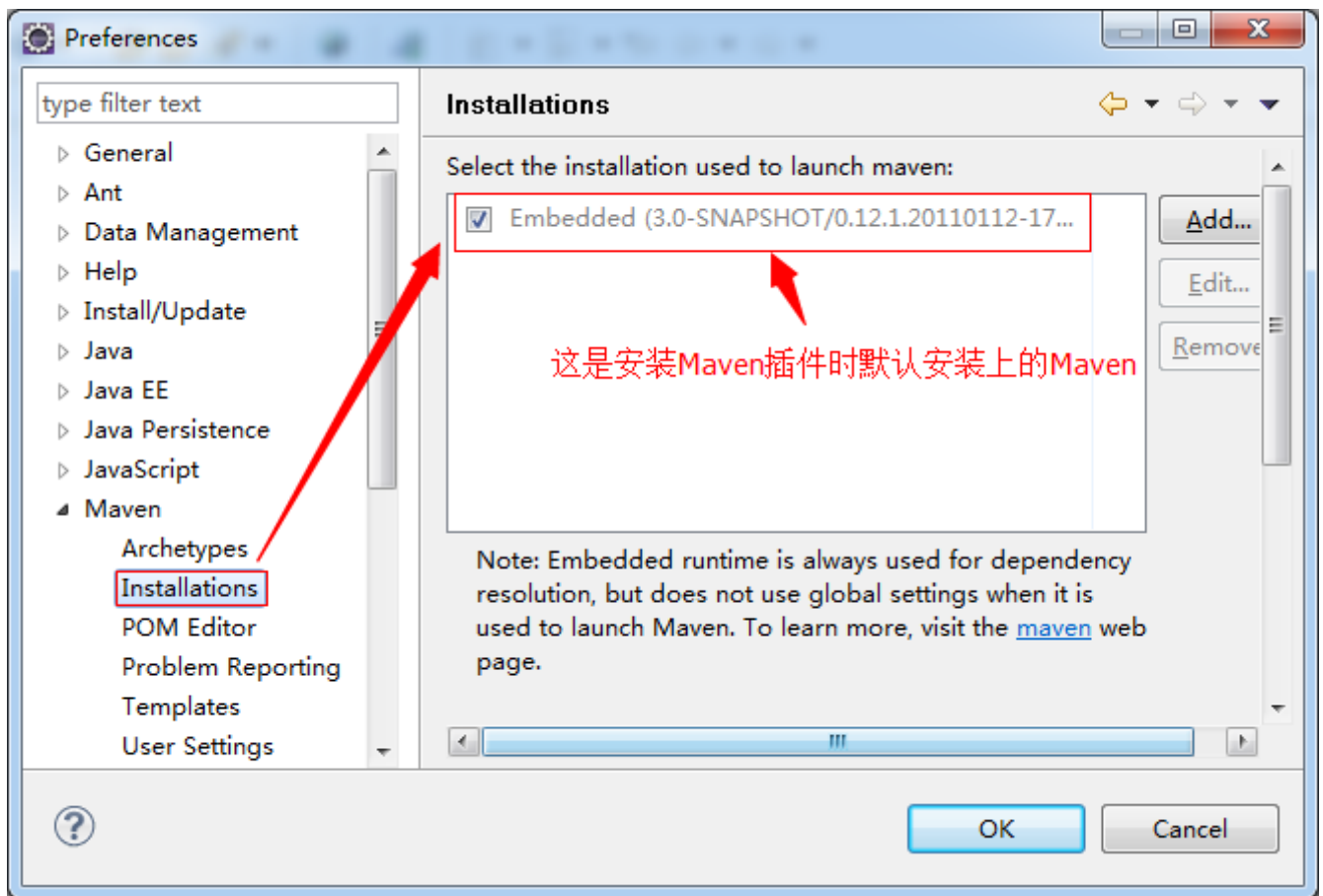
path路径的值为插件在本机上存放的路径地址

重新启动eclipse, 点击windows→preferences, Maven插件安装成功后看到如下画面：

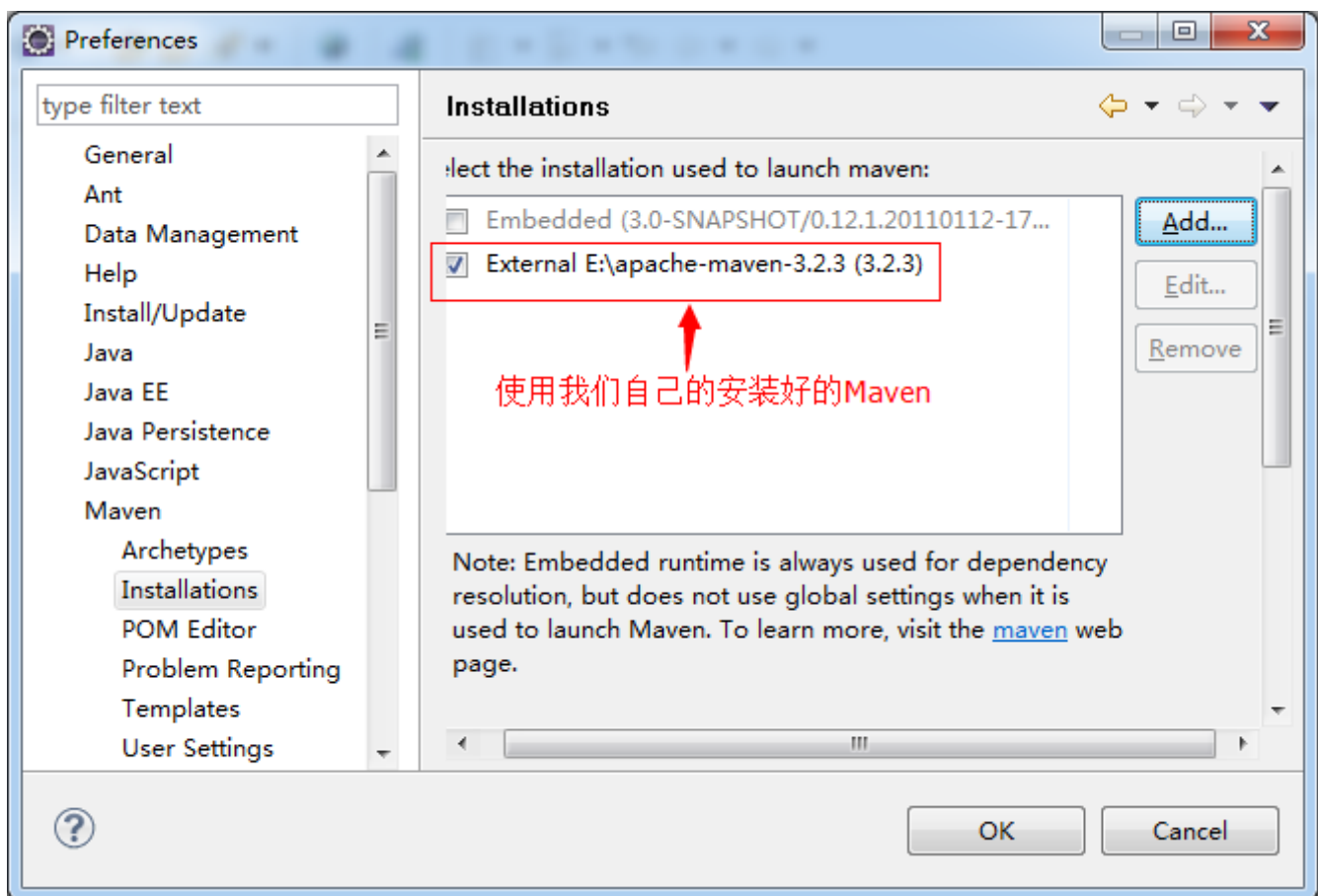
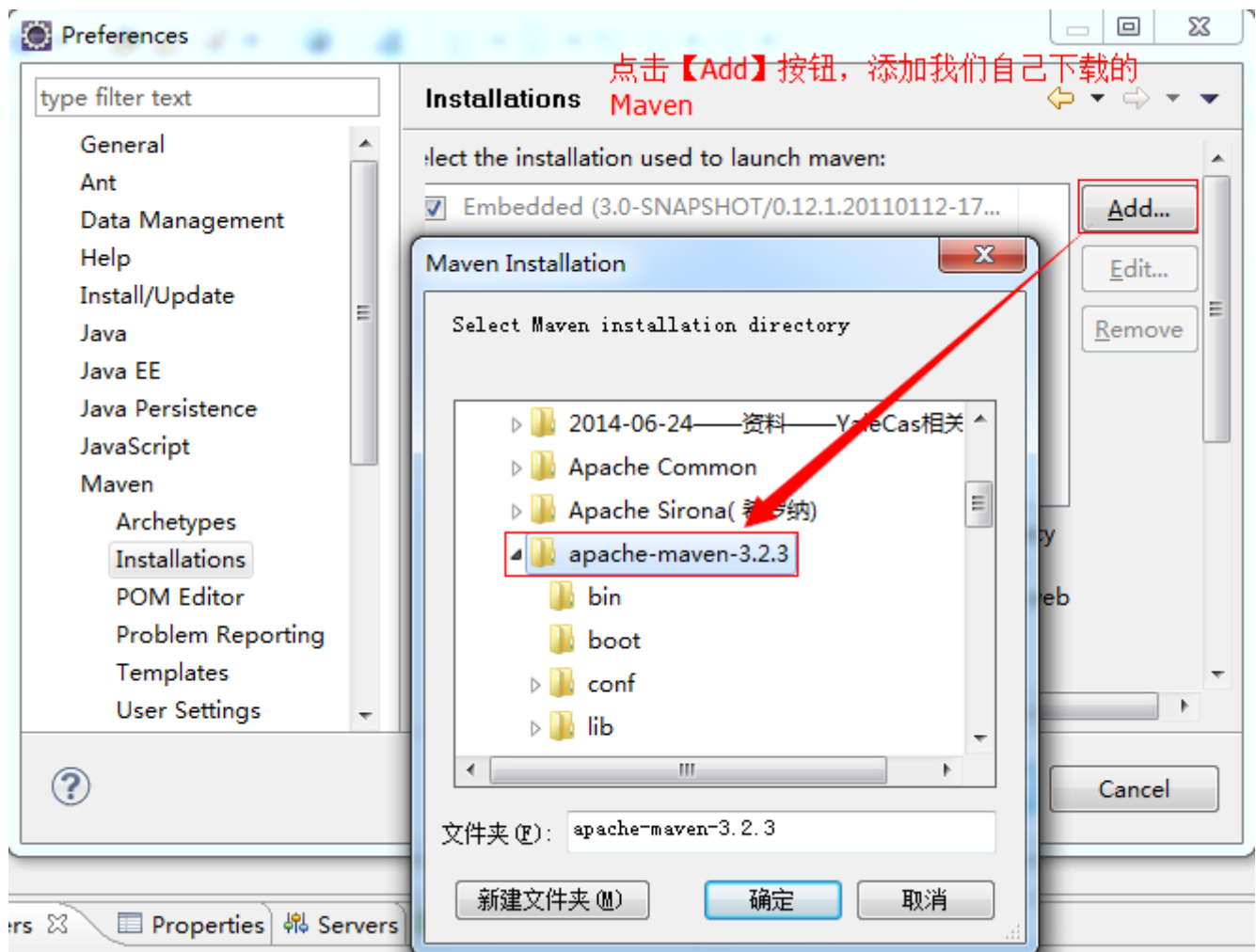


二、配置Maven插件

2.1、配置使用的Maven

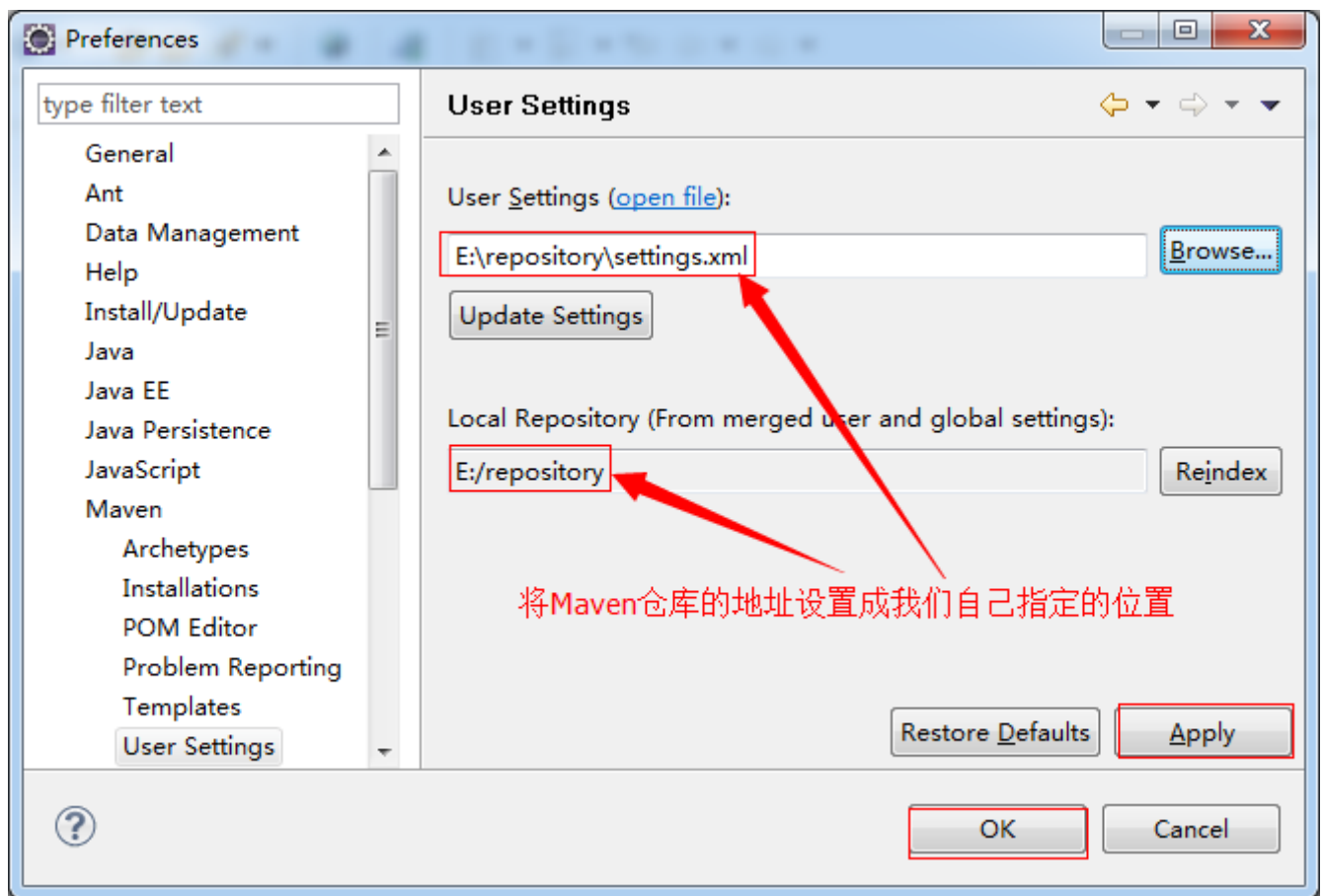
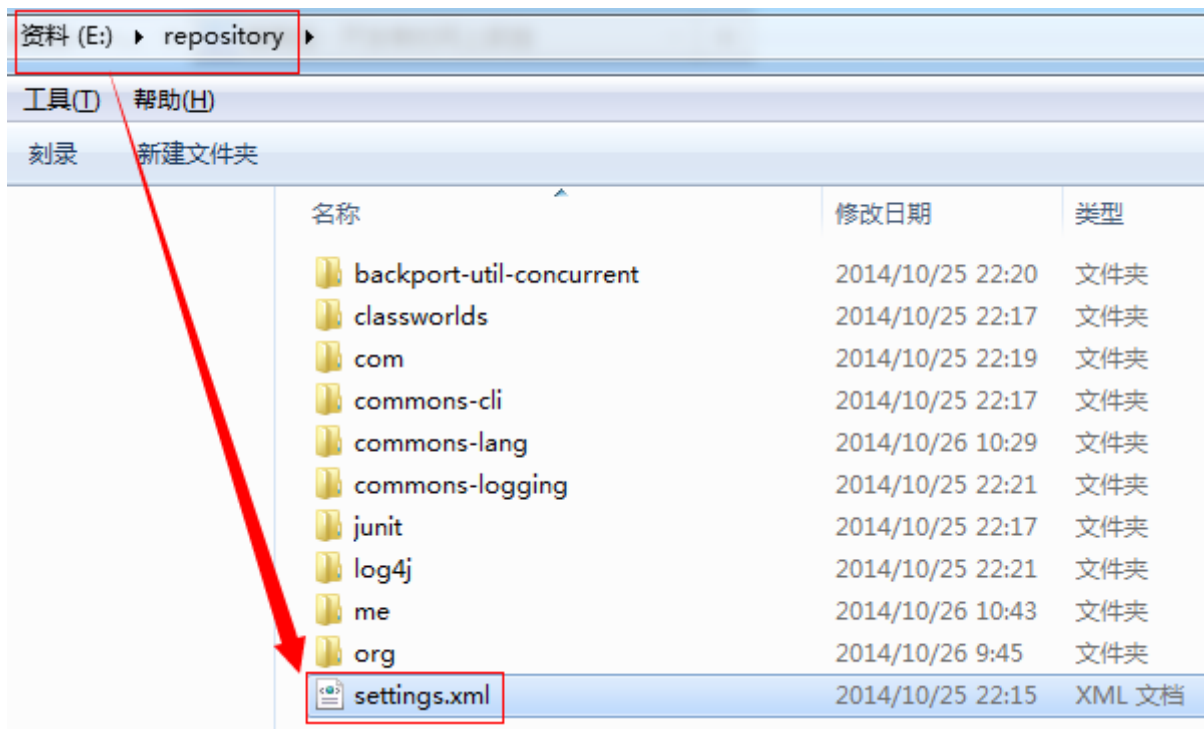


我们不使用默认安装的那个Maven，配置我们自己下载安装好的那个Maven，如下图所示：



2.2、配置User Settings

User Settings的默认配置如下图所示:



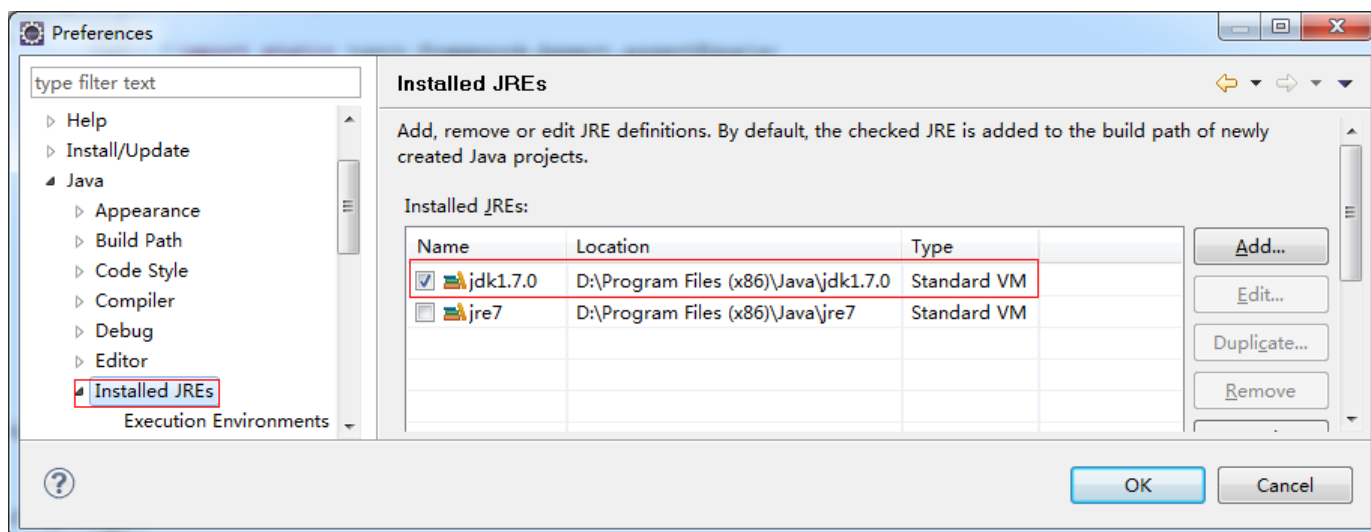
经过以上两步，Maven插件就算是设置好了。

2.3、配置Maven编译时使用的JDK

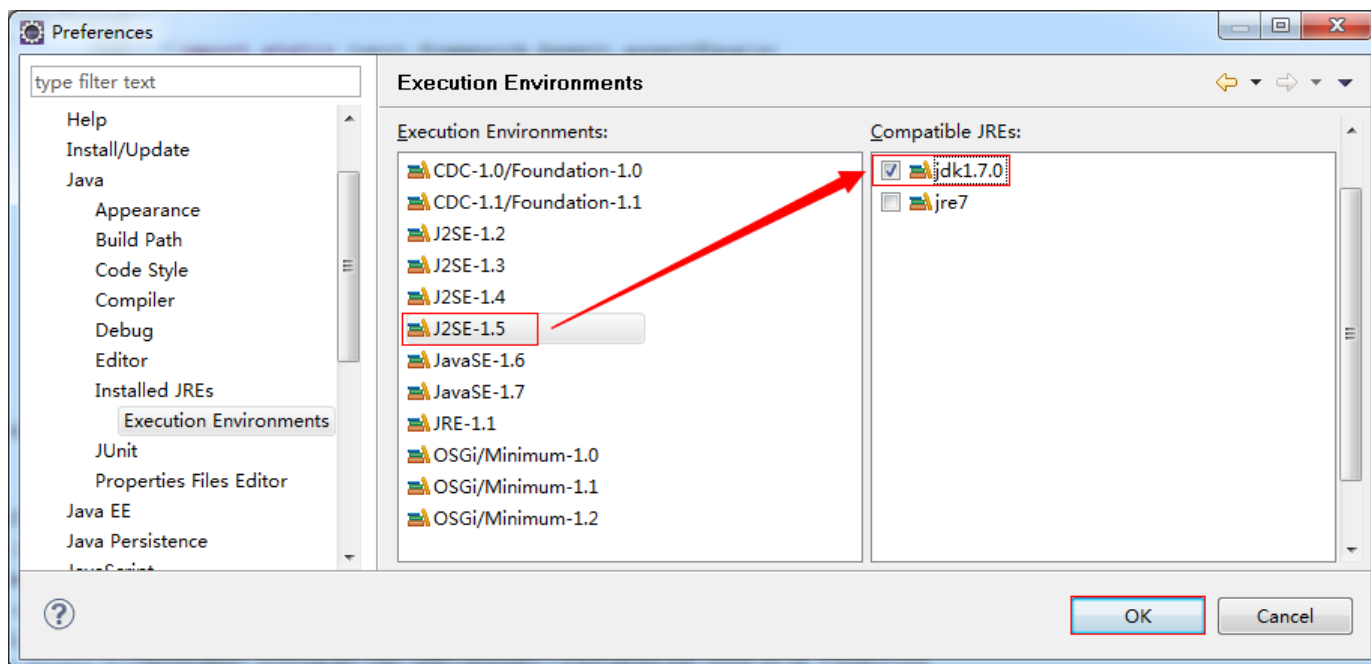
因为Maven必须要JDK1.6以上才能够正常运行，所以需要配置一下Eclipse使用的JDK。

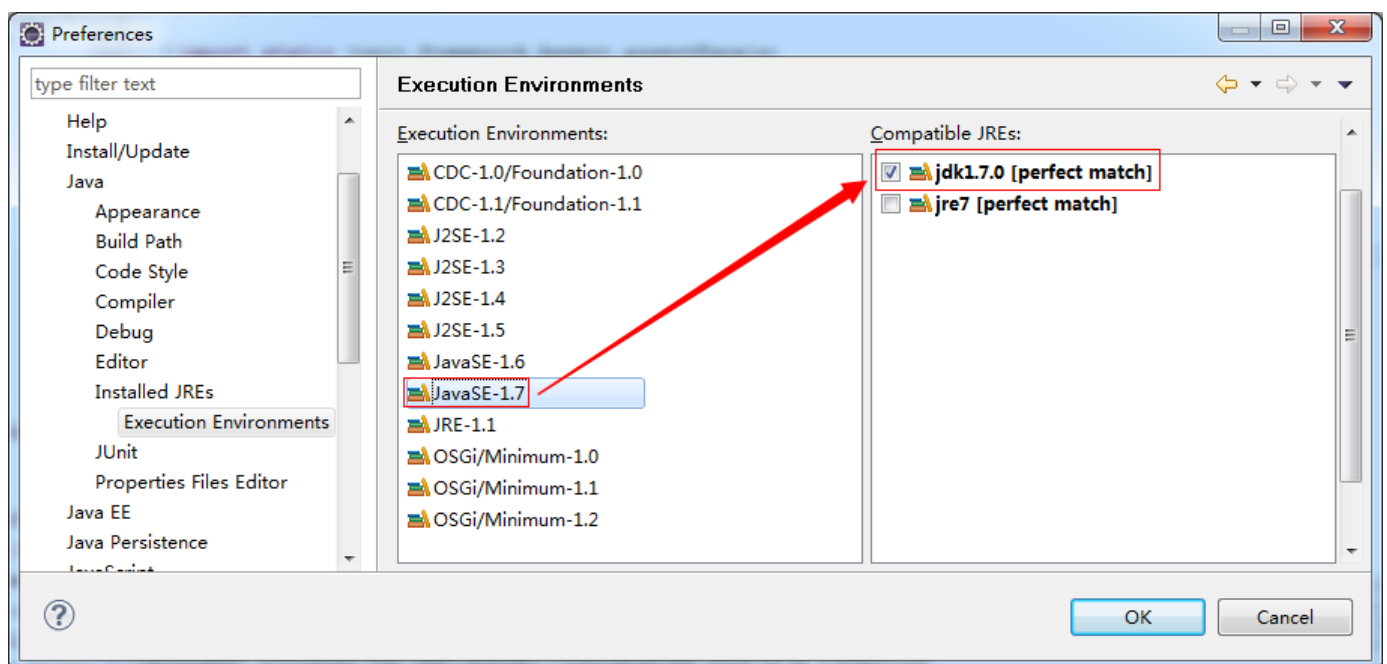
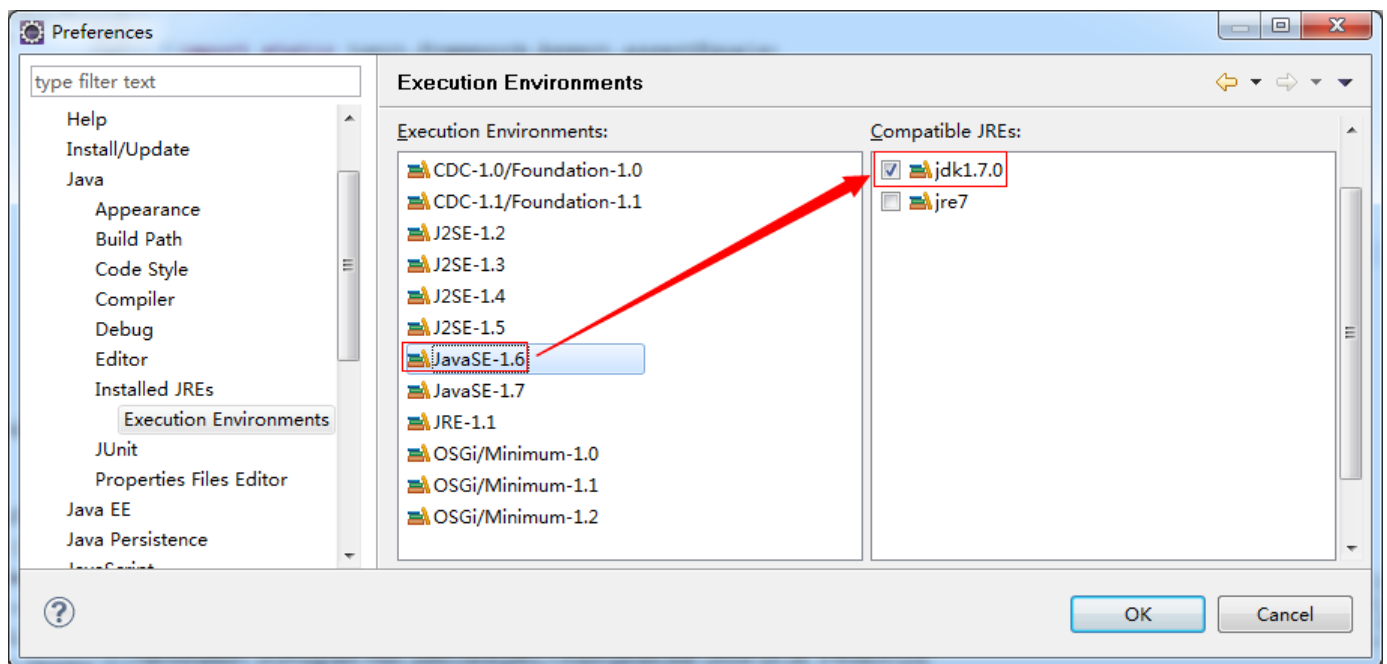
进入window->preferences窗口，选择java->Installed JREs，通过add按钮增加jdk对对应的路径加进来，否则在编译的时候会报错：

No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?



接着，进入Installed JREs的子项Execute Environment：在左侧选择JavaSE-1.5、JavaSE-1.6、JavaSE-1.7，右侧选择编译时使用的jdk版本，如下图所示：

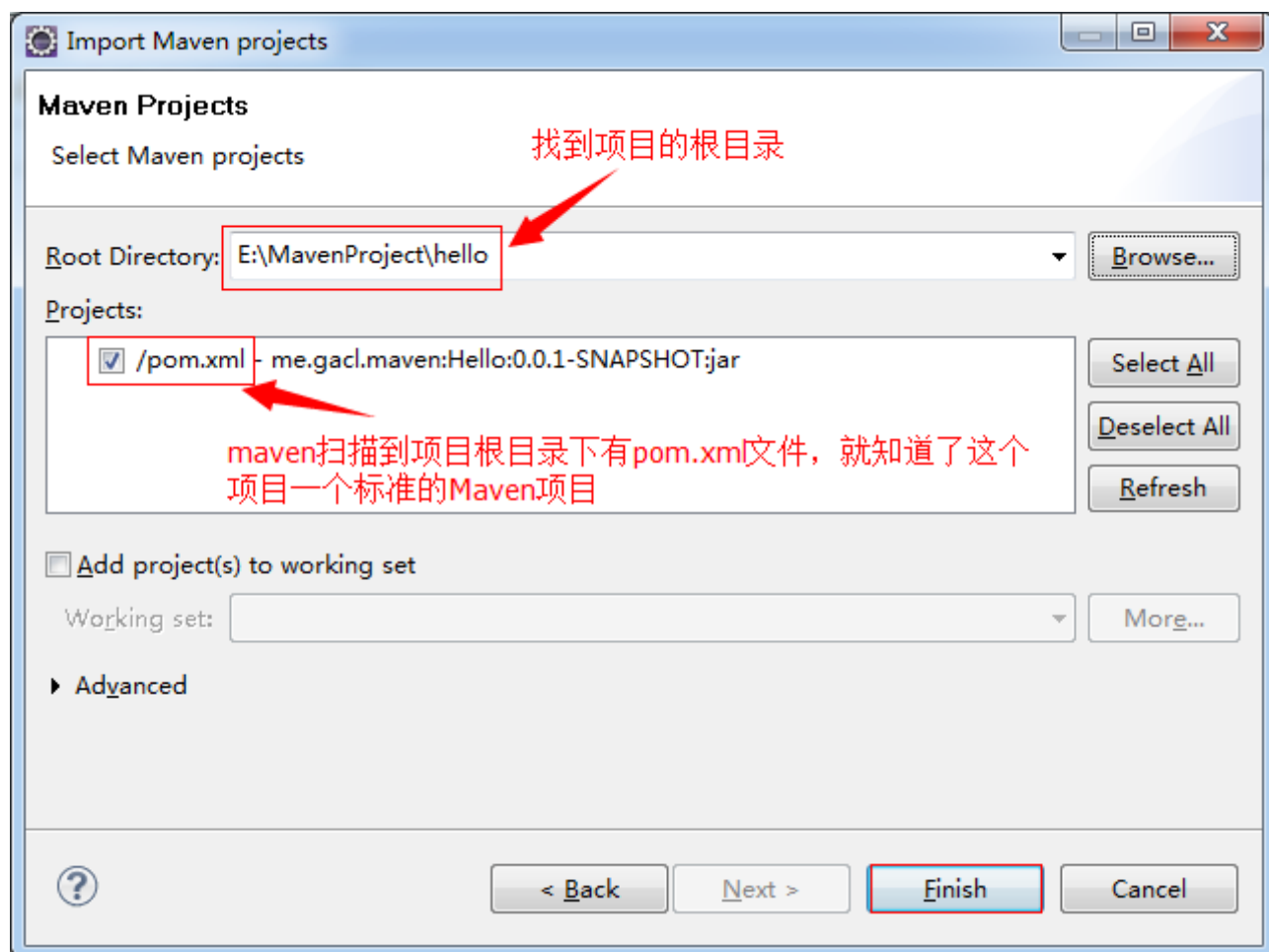
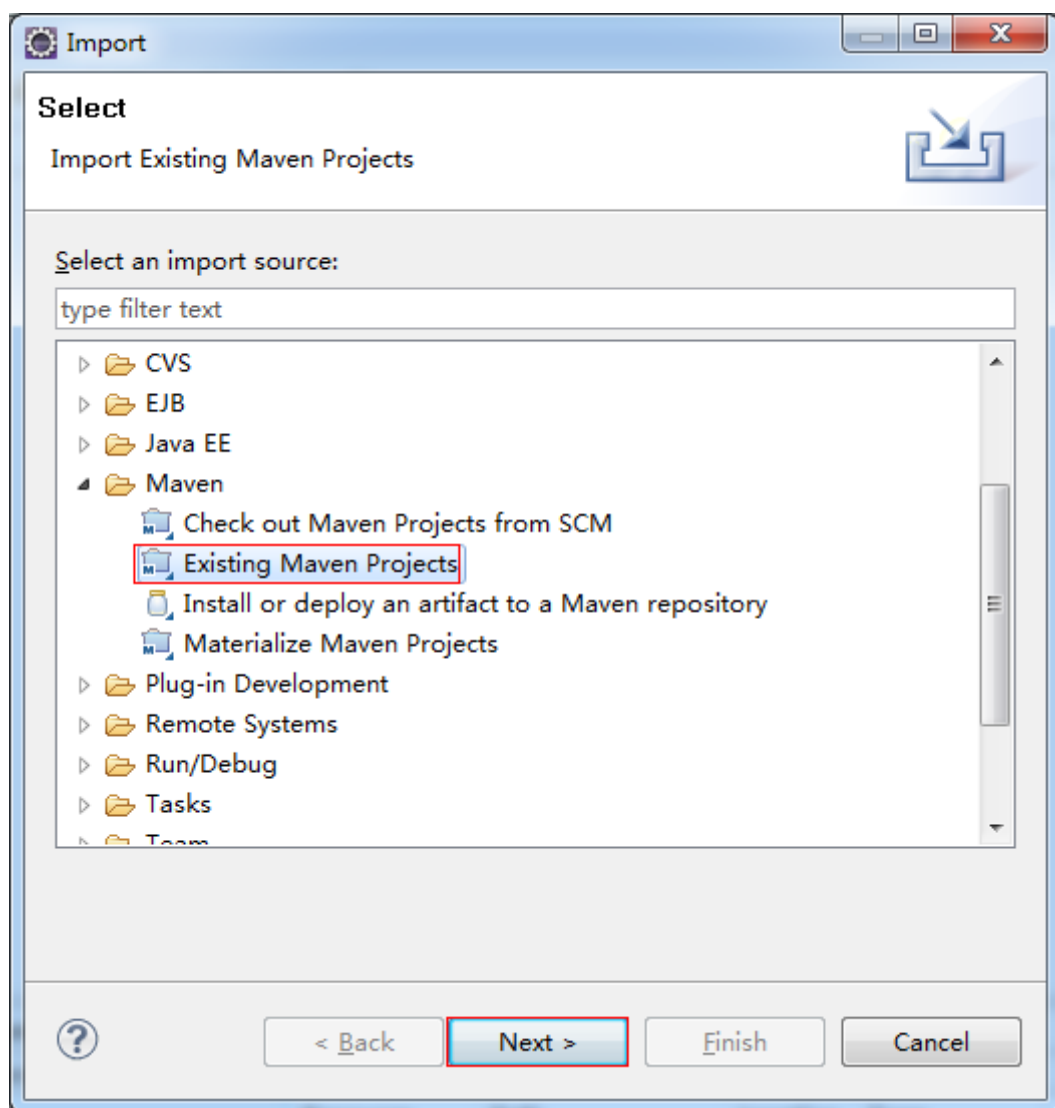




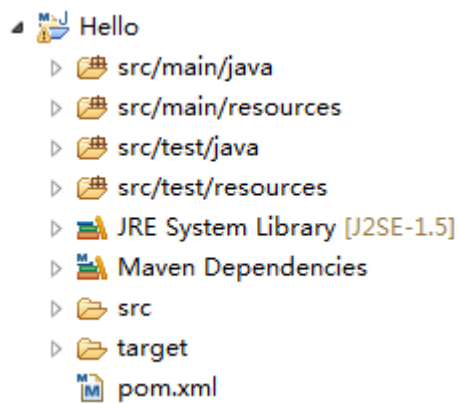
三、Eclipse中使用Maven插件

3.1、导入Maven项目

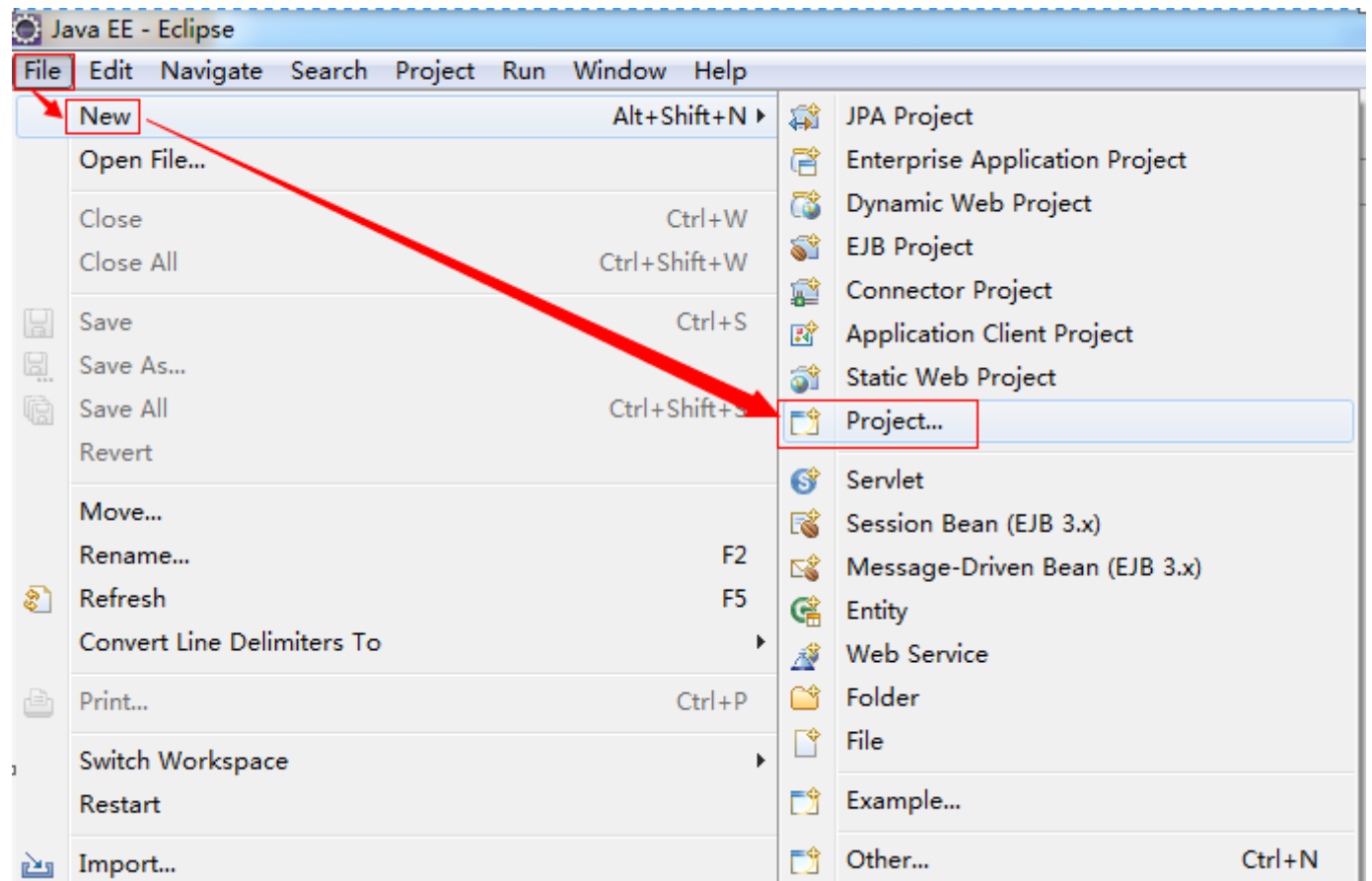
File→import

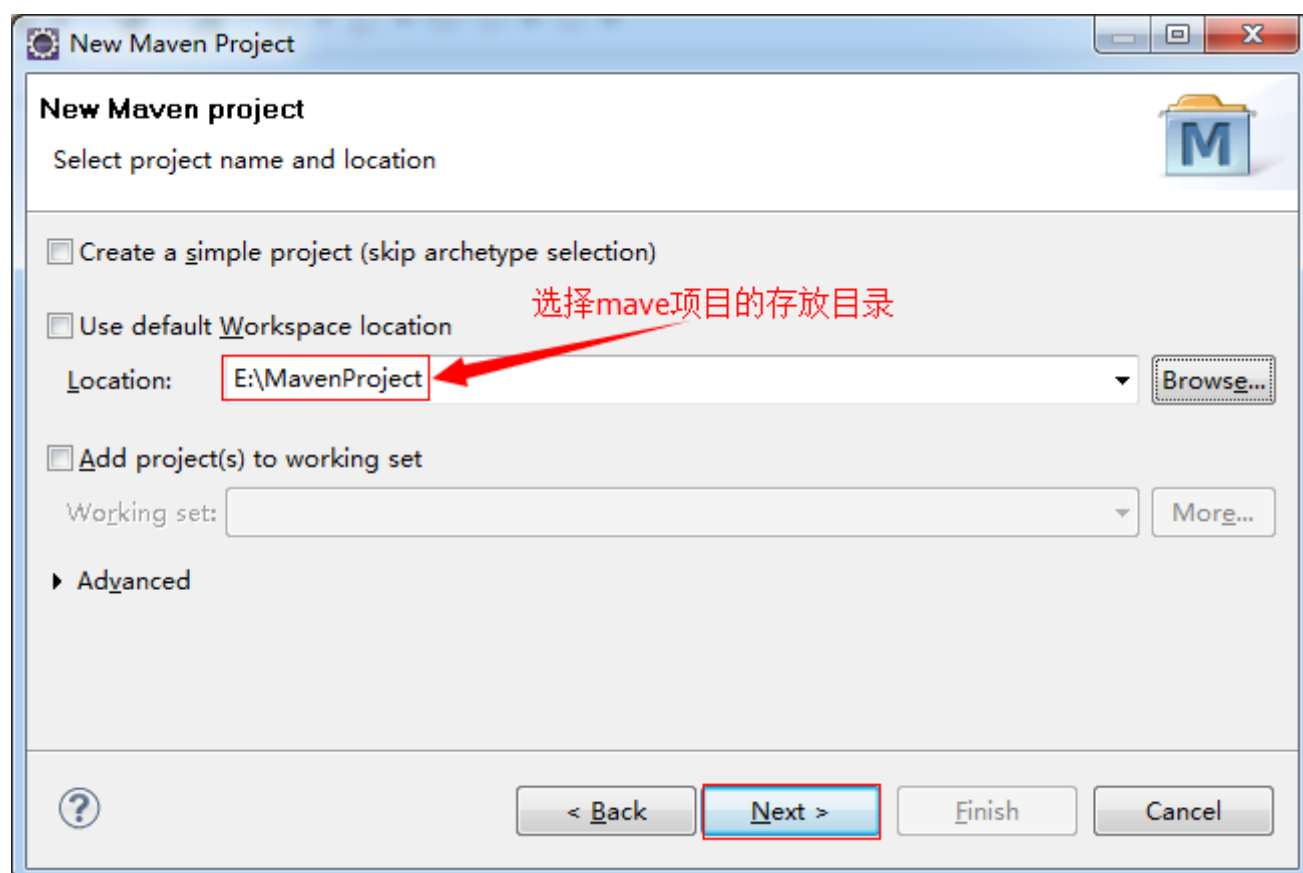
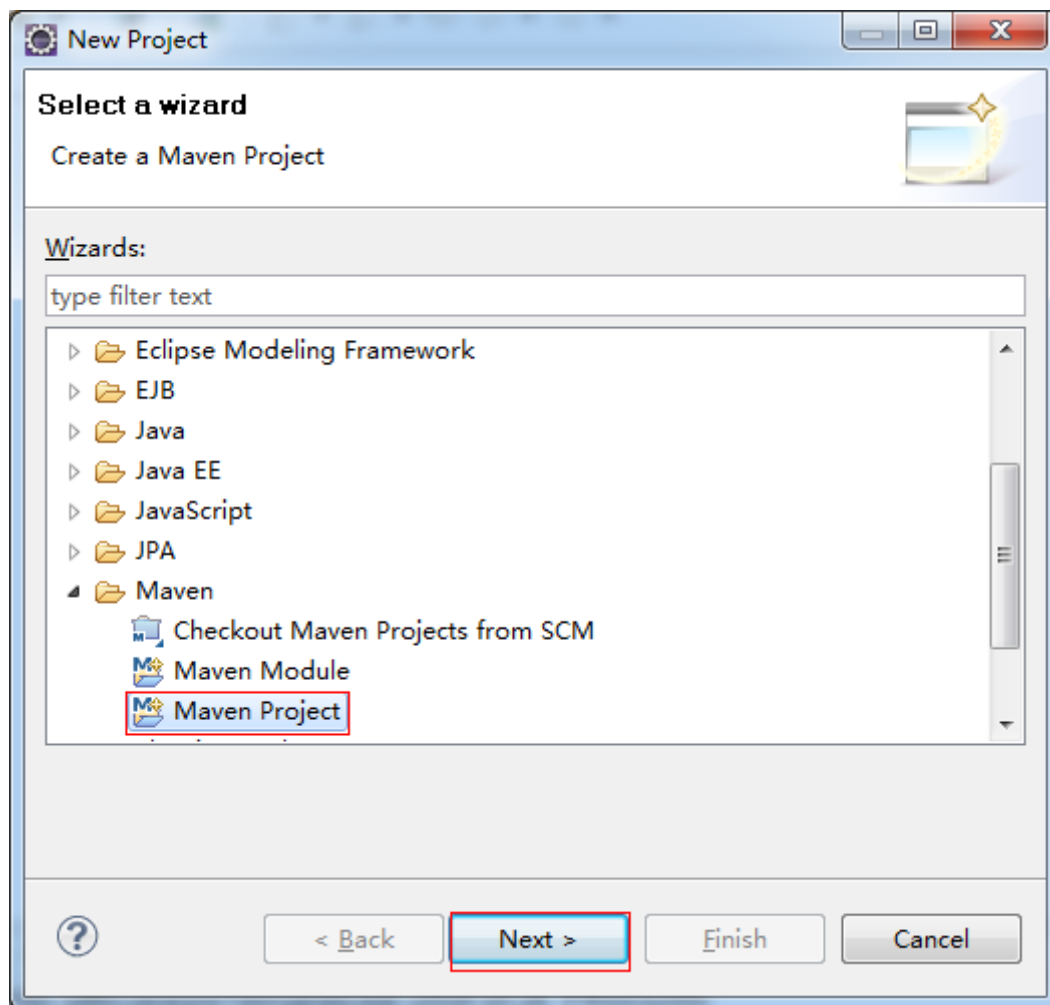


点击【Finish】按钮，完成项目的导入，如下图所示：



3.2、新建Maven项目





New Maven Project

New Maven project

Select an Archetype

Catalog: All Catalogs Configure...

Filter: ✕

Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1
org.apache.maven.archetypes	maven-archetype-webapp	RELEASE
org.apache.maven.archetypes	softeu-archetype-jsf	RELEASE
org.apache.maven.archetypes	softeu-archetype-seam	RELEASE
org.apache.maven.archetypes	softeu-archetype-seam-simple	RELEASE

An archetype which contains a sample Maven project.

☒ Show the last version of Archetype only ☐ Include snapshot archetypes Add Archetype...

► Advanced

? < Back Next > Finish Cancel

New Maven Project

New Maven project

Specify Archetype parameters

Group Id: me.gacl.maven 随便填写都可以

Artifact Id: MakeFriends 随便填写都可以

Version: 0.0.1-SNAPSHOT 这里的四个选项就是用来设置Maven项目的坐标位置的

Package: me.gacl.maven 这里的四个选项就是用来设置Maven项目的坐标位置的

Properties available from archetype:

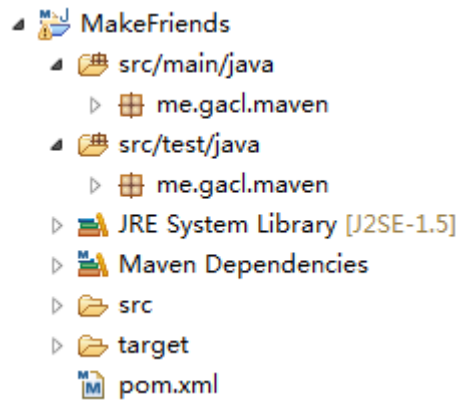
Name	Value

Add... Remove

► Advanced

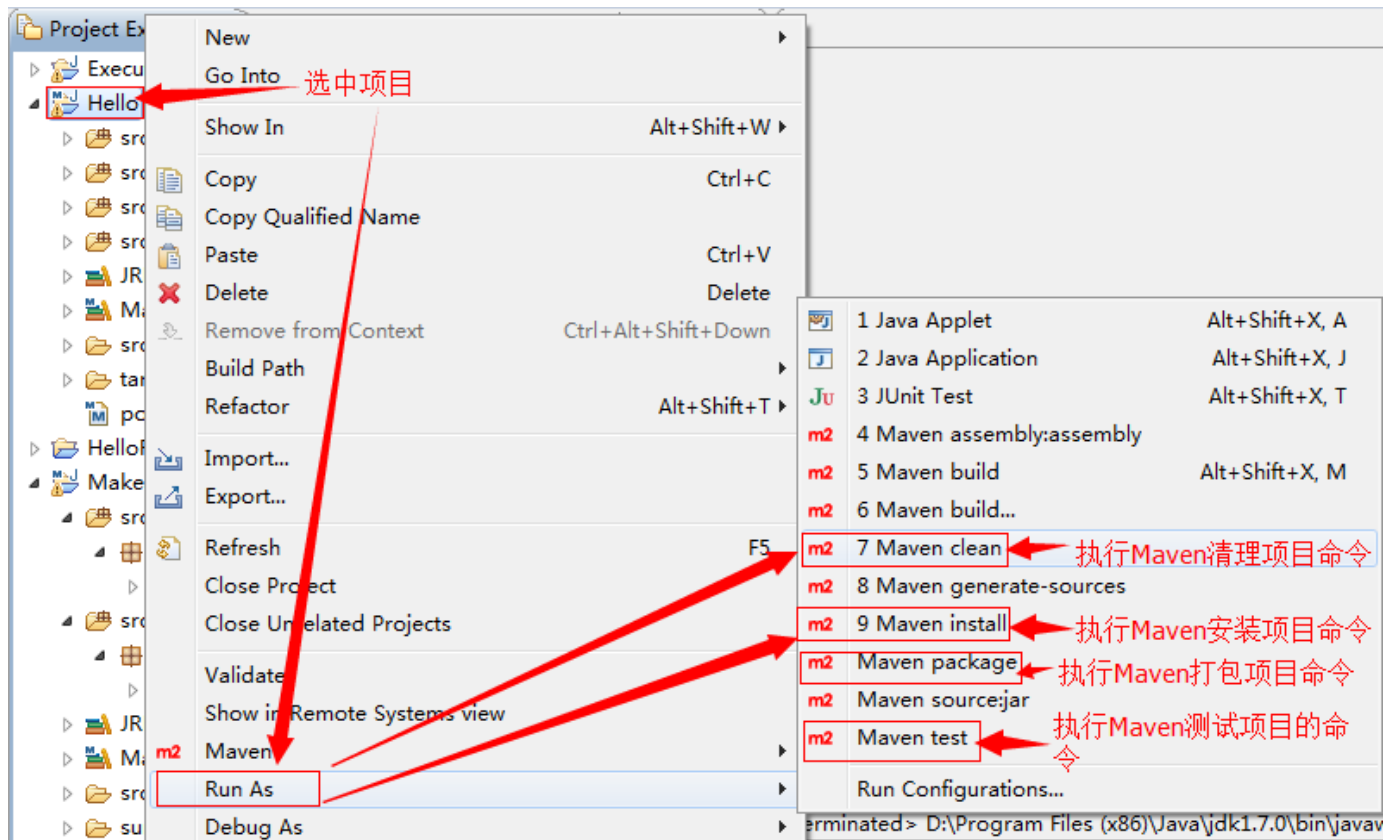
? < Back Next > Finish Cancel

点击【Finish】按钮，完成项目的创建，创建好的项目如下图所示：

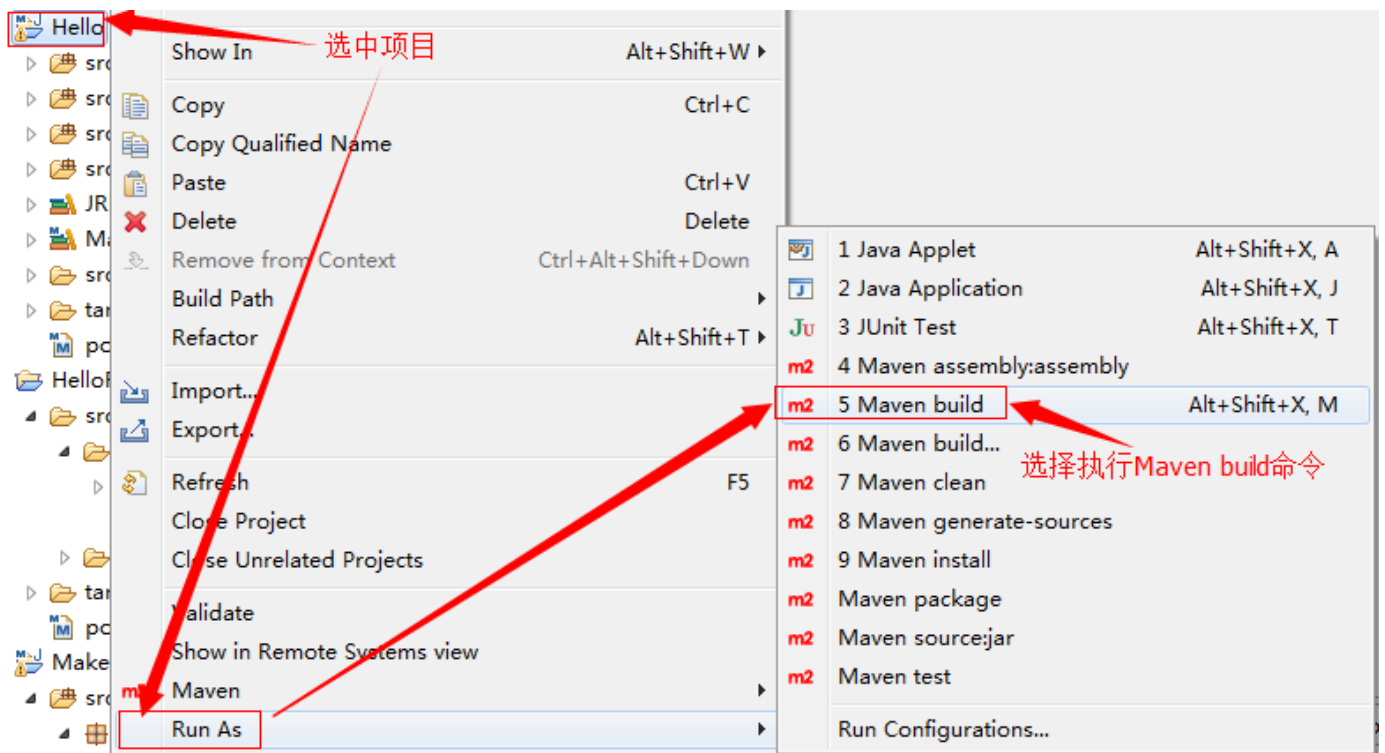


3.3、在Eclipse执行mvn命令

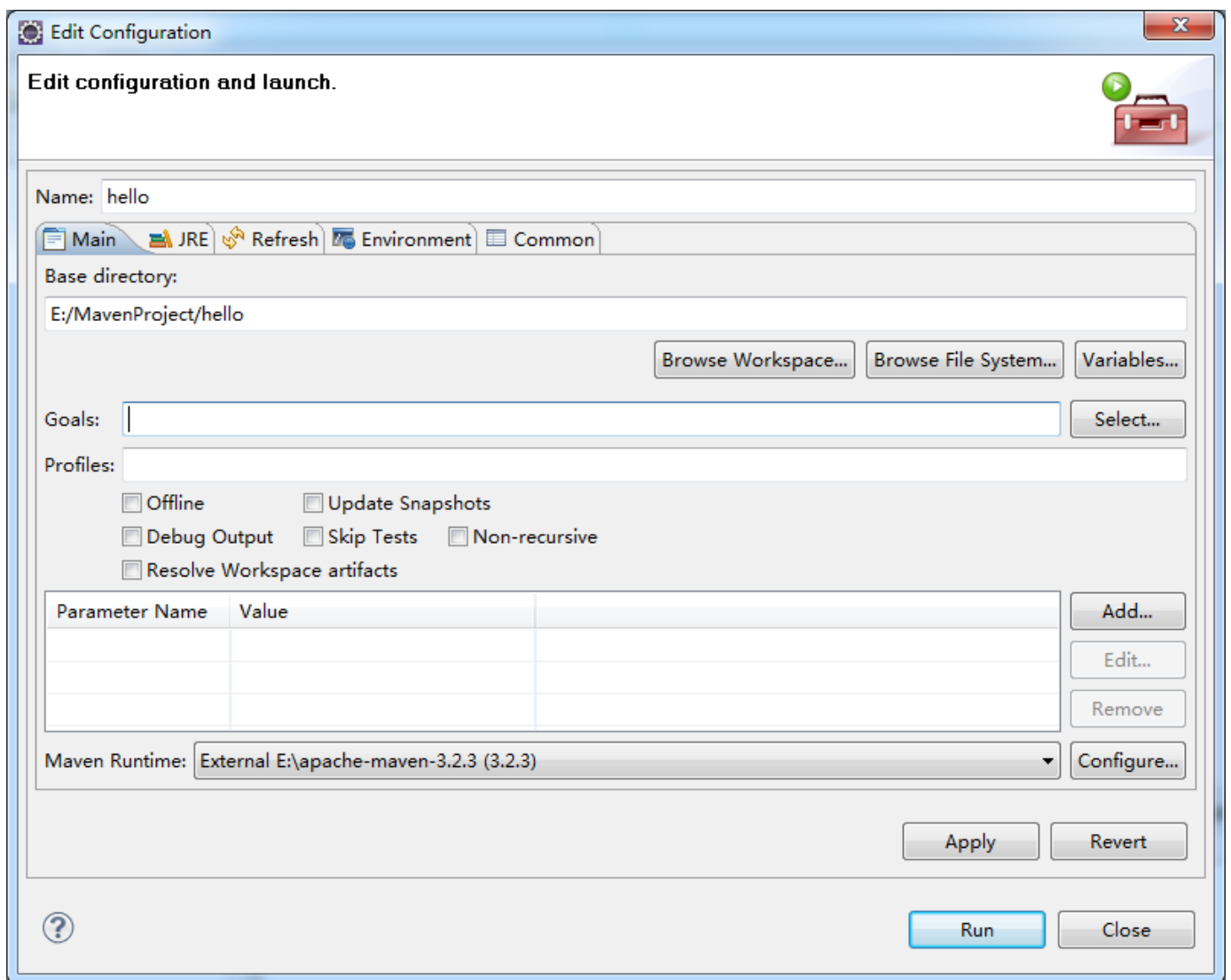
选中项目，点击鼠标右键→【Run As】或者【Debug As】→选择相应的Maven命令执行，如下图所示：



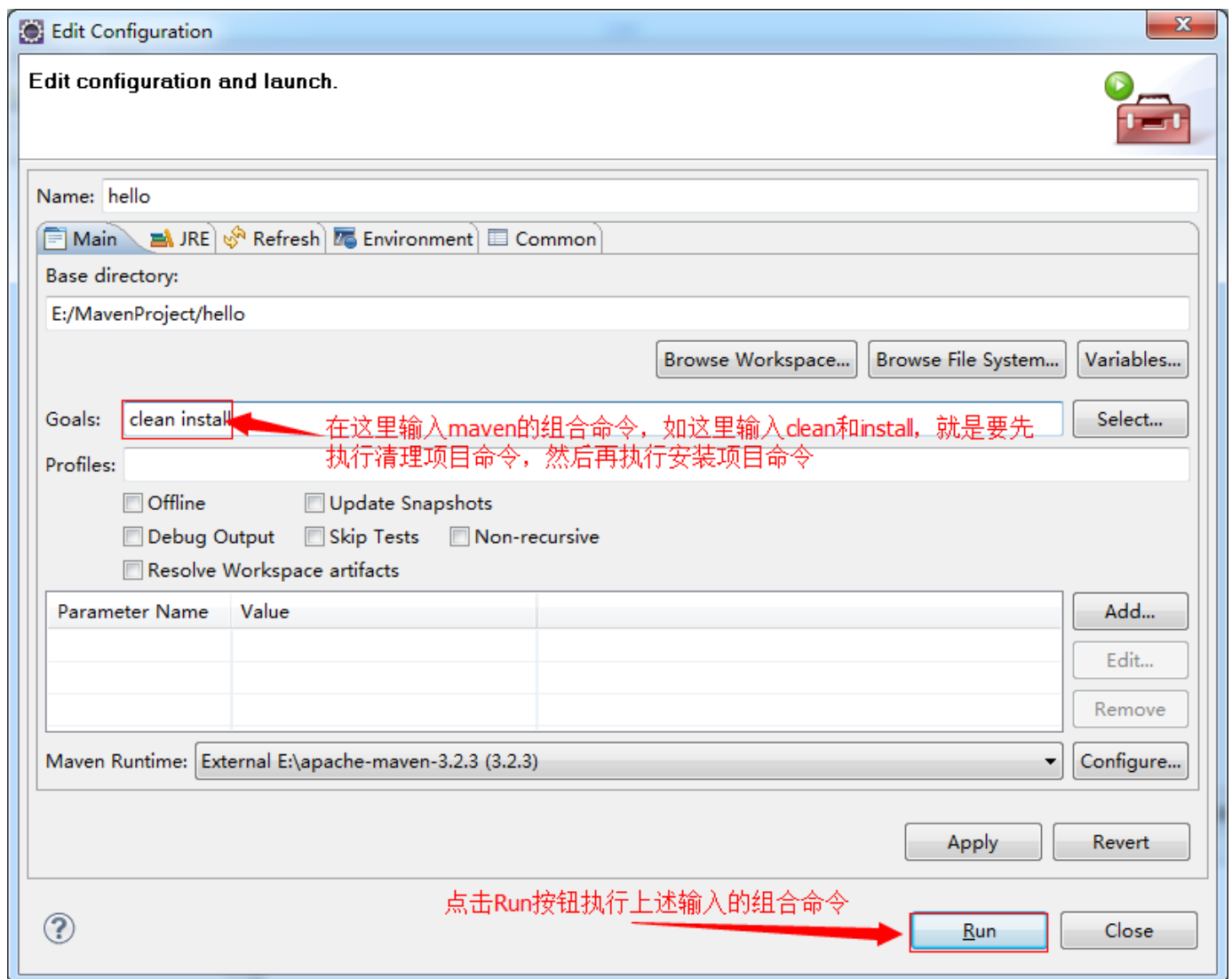
这种方式每次只能执行一个Maven命令，如果想像在cmd命令行那样使用组合命令，那么可以这样做



此时会弹出如下所示的对话框



在Goals输入框中输入要执行的组合命令，如下图所示：



以上就是Maven与Eclipse整合使用的内容。