

采用 maven 进行发布和版本管理

采用 maven 代替 ant 进行发布，把全部代码按组件分为若干子项目，每个子项目单独进行版本发布。不再需要把 lib 文件每个子项目中复制了，也不容易出现构建路径出错造成的问题。严格按照 svn 和 maven 建议的方式管理版本。

maven 是什么？

- maven 概览

为什么用 maven？

- 本地仓库、私服、中央仓库
- maven 生命周期

三套独立的生命周期：

Clean Lifecycle 在进行真正的构建之前进行一些清理工作。

Default Lifecycle 构建的核心部分，编译，测试，打包，部署等等。

Site Lifecycle 生成项目报告，站点，发布站点。

clean 生命周期：

pre-clean 执行一些需要在 clean 之前完成的工作

clean 移除所有上一次构建生成的文件

post-clean 执行一些需要在 clean 之后立刻完成的工作

default 生命周期：

validate

generate-sources

process-sources

generate-resources

process-resources 复制并处理资源文件，至目标目录，准备打包。

compile 编译项目的源代码。

process-classes

generate-test-sources

process-test-sources

generate-test-resources

process-test-resources 复制并处理资源文件，至目标测试目录。

test-compile 编译测试源代码。

process-test-classes

test 使用合适的单元测试框架运行测试。这些测试代码不会被打包或部署。

prepare-package

package 接受编译好的代码，打包成可发布的格式，如 JAR 。

pre-integration-test

integration-test

post-integration-test

verify

install 将包安装至本地仓库，以让其它项目依赖。

deploy 将最终的包复制到远程的仓库，以让其它开发人员与项目共享。

- svn 版本 :trunk branches tag 的不同含义
- maven 中 jar 包版本： snapshot release 的含义
- maven 插件

了解插件体系、常用插件、插件与生命周期的关系。

maven 的具体使用

- 1、安装 maven
- 2、修改 maven 设置
- 3、安装 eclipse 的 maven 插件
- 4、在 tomcat 安装热发布插件 jrebel
- 5、新建项目过程

1、安装 maven

安装 maven3 安装最新的 3.2.5 版

本, <http://maven.apache.org/download.cgi>

下载后, 解压到例如:

d:\java\apache-maven-3.2.5

然后设置环境变量:

新建一个系统变量: M2_HOME, 路径即为解压后的安装路径, 如:

d:\java\apache-maven-3.2.5

再配置 path 环境变量, 在 path 值的末尾添加 "%M2_HOME%\bin"

打开 cmd 窗口: 输入 mvn -version, 出现如下内容表示安装成功。

2、安装 eclipse 的 maven 插件

m2e - <http://m2eclipse.sonatype.org/sites/m2e>

3、修改 maven 设置

- 更改 maven 本地仓库地址

在 D 盘新建目录, 例如: d:\java\maven, 用作本地仓库 (默认的在 '我的文档' 中, 重装系统容易丢失)

打开安装目录下, conf 文件夹下的 settings.xml 文件, 找到第 localRepository 一行, 把注释去掉, 修改成例如:

```
<localRepository>d:\java\maven</localRepository>
```

再把 setting.xml 复制到本地仓库根下，如 d:\java\maven

- 添加 server

在 settings.xml 中，servers 中添加：

```
<server>
```

```
  <id>eap-release</id>
```

```
  <username>deployment</username>
```

```
  <password>deployment123</password>
```

```
</server>
```

```
<server>
```

```
  <id>eap-SNAPSHOT</id>
```

```
  <username>deployment</username>
```

```
  <password>deployment123</password>
```

```
</server>
```

这是私服中正式版本和快照版本的用户和密码设置，使大家可以向私服

发布版本

- 修改 eclipse 的 maven 属性

从 窗口 - 首选项 - maven- installations 处”，点击 add，选择 maven 的安装目录；

窗口 - 首选项 - maven- user settings 处”，选择自定义的 settings.xml

位置

- 解出 parent 项目

从 svn 解出 eap2-parent 项目，及需要的其它项目。 eap2-parent 是用于管理的父项目，所有 EAP 平台组件项目均继承该项目。

在 eap2-parent 项目的 pom.xml 中添加属性 tomcatPath，指向本机用于测试的 tomcat

```
<properties>
```

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

```
ncoding>
```

```
    <!-- 设置 tomcat 测试环境目录，请自行修改 -->
```

```
    <user.tomcatPath>E:/tomcat/webapps/ROOT/WEB-INF</
```

```
user.tomcatPath>
```

```
    <!-- 设置 tomcat 测试环境中指定的企业 ID -->
```

```
    <user.comId>abcd</user.comId>
```

```
</properties>
```

4、在 tomcat 安装热发布插件 jrebel

- 复制文件

复制 jrebel.jar 和 jrebel.lic 到 tomcat/lib 目录下；

- 在 tomcat/web-inf/classes 中添加 rebel.xml ，内容为：

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<application
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns="http://www.zeroturnaround.com"      xsi:schemaLocation="http
```

```
://www.zeroturnaround.com      http://www.zeroturnaround.com/alder
```

```
aan/rebel-2_0.xsd">
```

```
  <classpath>
```

```
    <dir name="E:\tomcat\webapps\ROOT\WEB-INF\classes"/>
```

```
  </classpath>
```

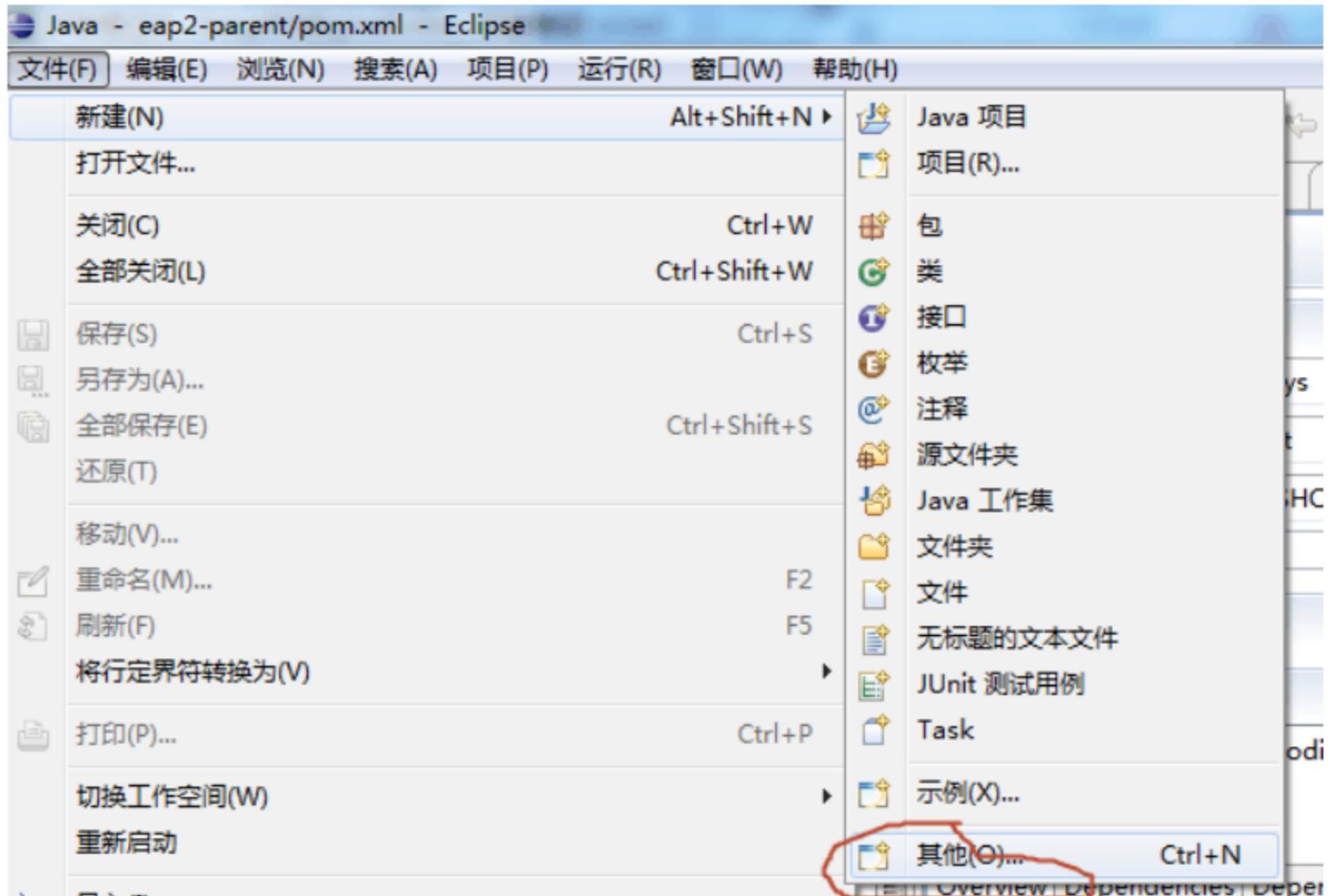
```
</application>
```

- 在 catalina.bat 添加参数：

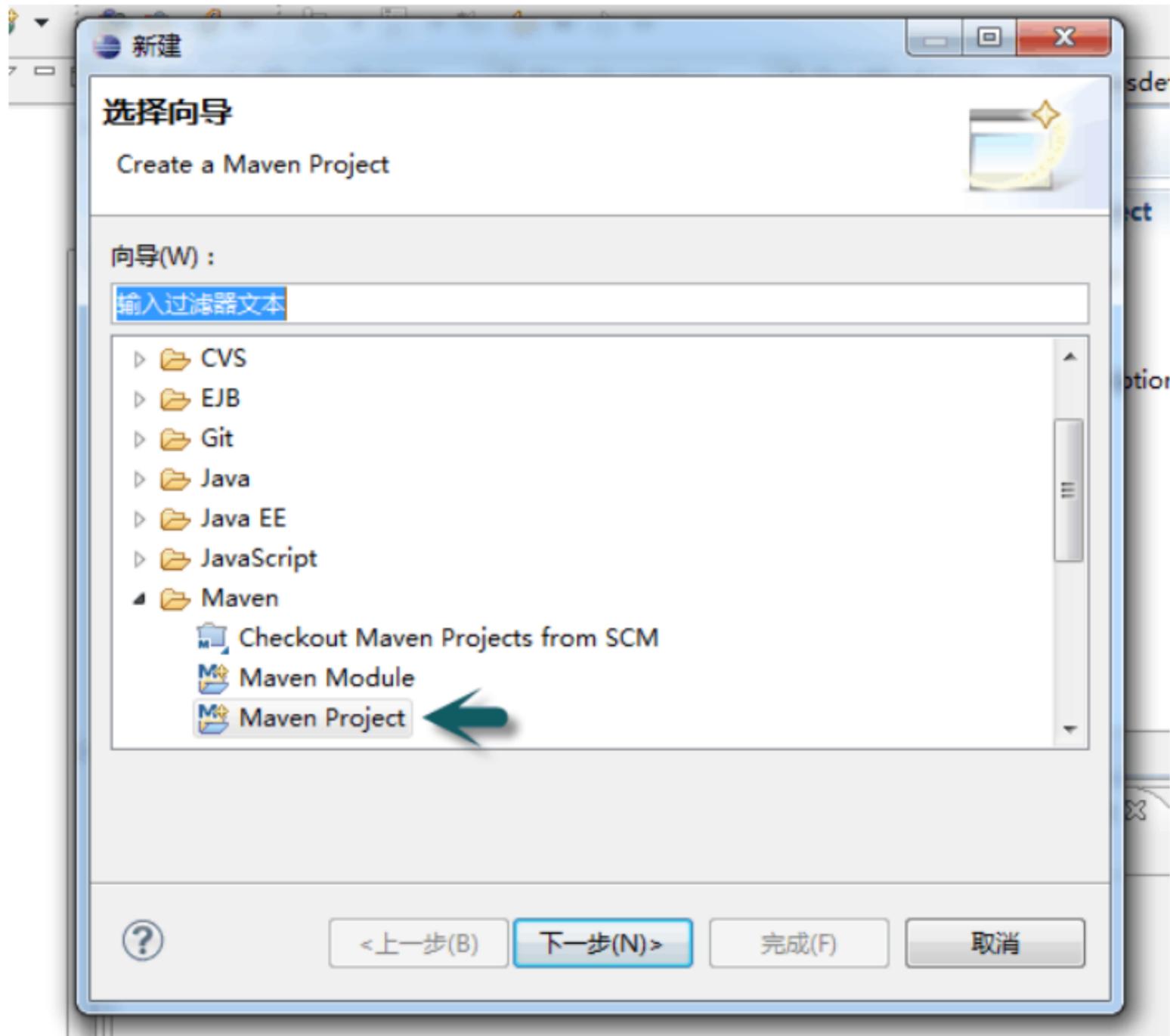
```
set JAVA_OPTS=-noverify -javaagent:E:/tomcat/lib/jrebel.jar
```

5、新建项目过程

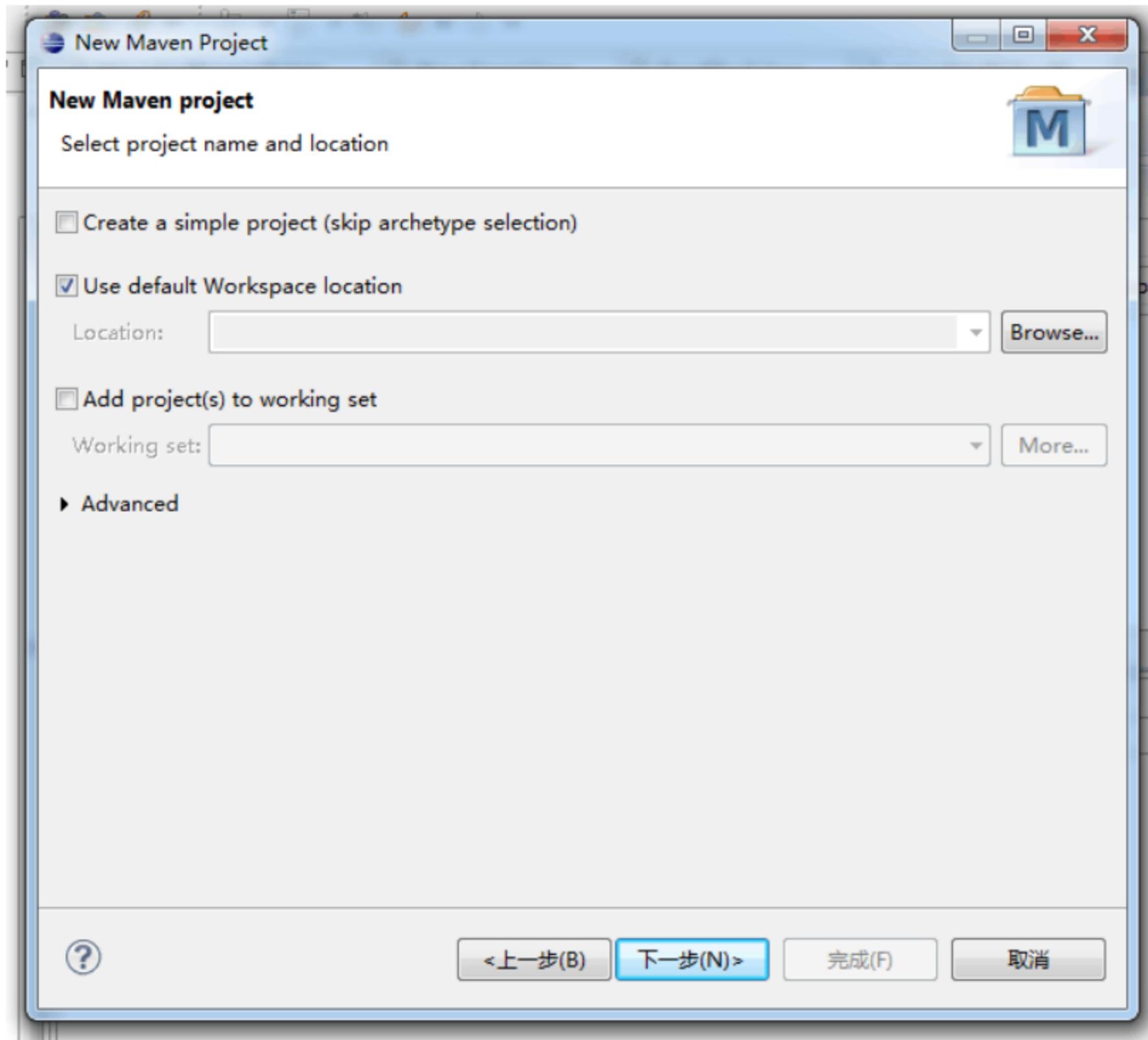
- 新建 ‘其它 项目



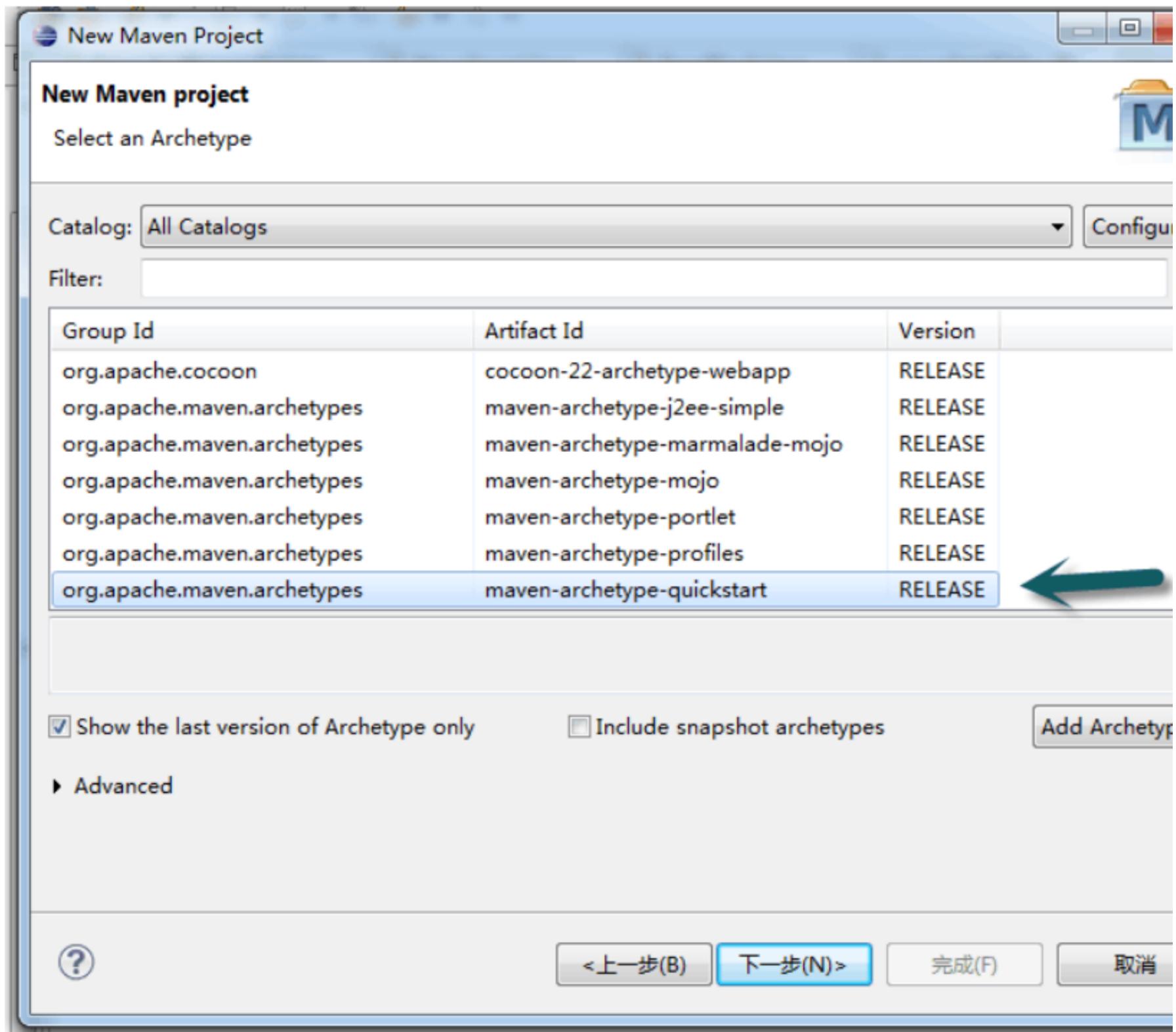
选择 maven 项目



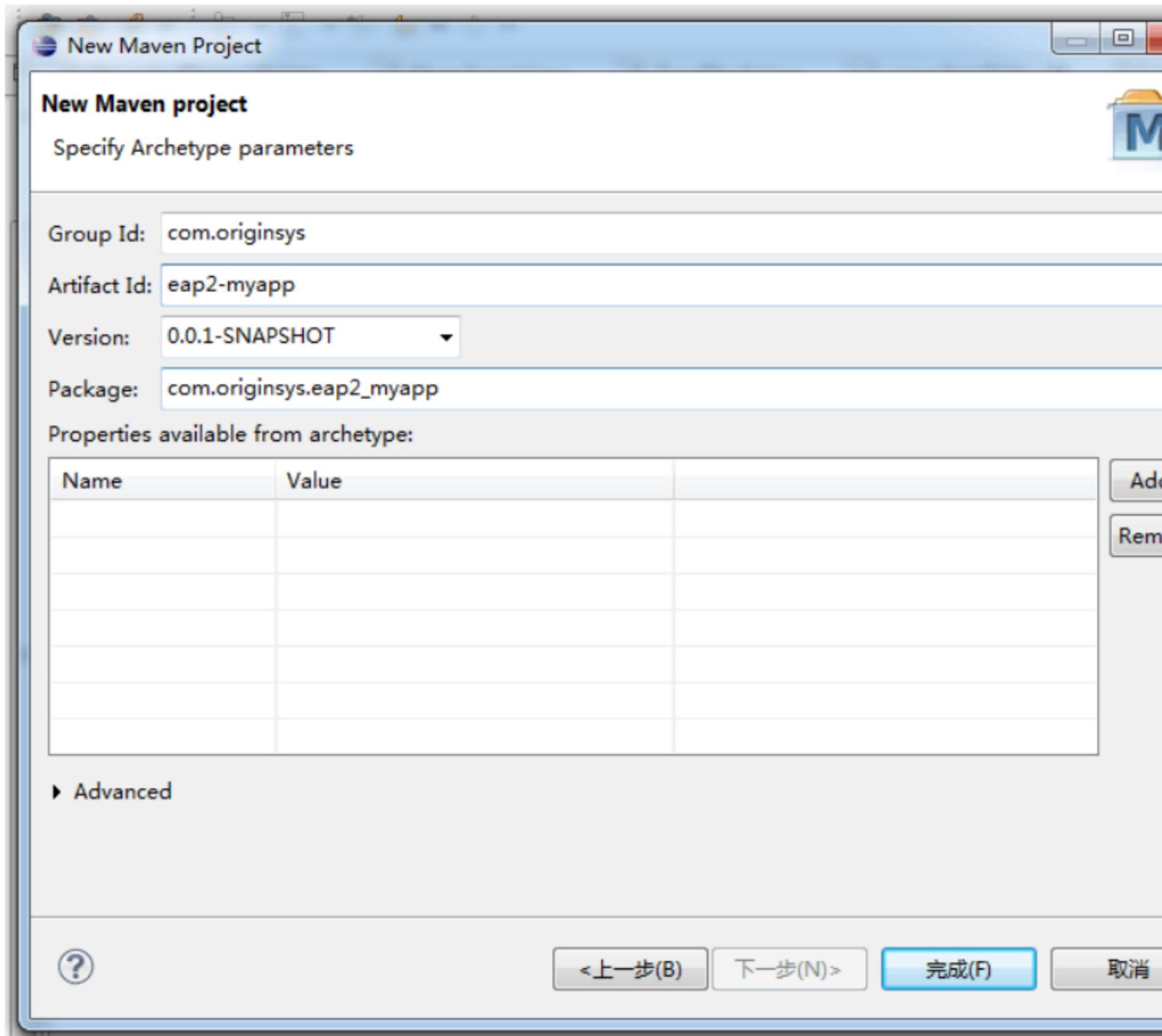
下一步



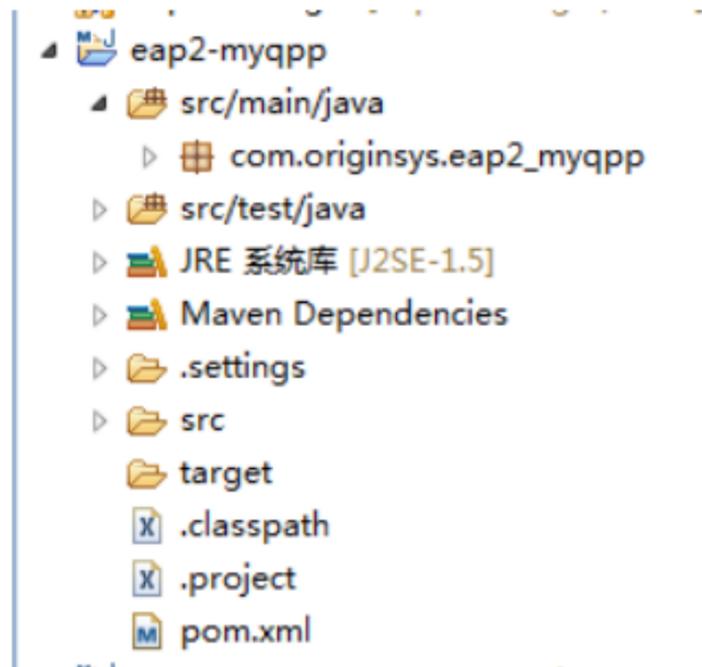
- 下一步，选择 quickstart 模板



- 输入 groupId 为 com.originsys ，Artifact Id 为你的组件名称 ，输入版本号



- 生成的项目结构：



其中：

src/main/java 是默认的源程序文件夹，删除其中自动创建的包 com.originsys.eap2_myqpp，添加自己的包及程序；

src/test/java 是放 junit 的测试文件的，我们没有使用 junit，这个目录可忽略；

- pom.xml 是最重要的，需要以下修改：

插入 <parent>，eap2-parent/pom.xml 中有公共的配置：

```
<parent>

    <groupId>com.originsys</groupId>

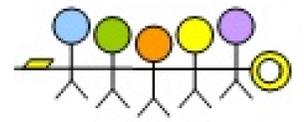
    <artifactId>eap2-parent</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <relativePath>../eap2-parent/pom.xml</relativePath>

</parent>
```

加入后，重复的属性设置就不需要了，去掉 <groupId><url><properties> 等标签。



```
<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 >

<parent >
<groupId >com.originsys </groupId >
<artifactId >eap2-parent </artifactId >
<version >0.0.1-SNAPSHOT </version >
<relativePath >../eap2-parent/pom.xml </relativePath >
</parent >
```

[parent 是干嘛的]

[maven 是如何判读 parent 在 或者是不在的呢???

[maven 继承 ???]

```
<artifactId >eap2-ysy </artifactId >
<packaging >jar </packaging >
<name>eap2-ysy </name>
<dependencies >
<dependency >
```

在 dependencies 中加入对 eap-core-common 的依赖，这个包中已配置了大多数公共 jar

```
<dependency>  
  
    <groupId>com.originsys</groupId>  
  
    <artifactId>core-common</artifactId>  
  
    <version>2.14.1-SNAPSHOT</version>  
  
</dependency>
```

- 完成

新建项目完成，查看 Maven Dependencies，可看到所有依赖的 jar 包都已引入。在项目中的 src/main/java 源文件夹中建立包和编写程序即可。

- 编译、发布到应用、发布到仓库

自动编译：Maven 项目同时也是 eclipse 项目，编写或修改 java 文件后可自动编译到 target/classes 中，如未能自动编译，打开 cmd 窗口，cd 至 workspace 下的项目目录中，输入 maven 命令：`mvn -U eclipse:clean eclipse:eclipse` 执行即可。

编译：在 pom.xml 上点击右键，选择：运行方式 - maven build...，在弹出的窗口中输入执行目标（goals）：`:clean compile`，运行即先清空 target，然后对源代码全部编译到 target/classes 中。同时，复制插件也绑定在了 compile 生命周期中，执行 compile 同时，将会把全部依赖的

jar 包复制到 tomcat 应用的 lib 下，把 eap2-parent 中的 classes(系统配置)复制到 tomcat/

发布到应用：用 `compile` 首次编译过后，日常修改后可执行 `:validate` 命令，发布到 tomcat 应用中。

发布到仓库：用 `clean deploy` 命令，发布到公司组件仓库。

- 提交到 svn

在 eclipse 中项目根目录中点击右键，从小组 (team) 菜单中点击“共享项目”，然后选择 svn，并在选择已有的资源库位置中选择：`svn://115.28.14.199`，在使用指定的模块名中，例如组件名为 myapp，则输入：`eap2-myapp/trunk`

即创建一个新的组件项目，并放入主线目录中。

共享完成后，再从 team 菜单中点击“提交”，提交到 svn，提前前看一看是否存在不必要的文件，如 `.settings.xml`、`target`、`.project` 等，在 svn 配置忽略的资源中排除，只上传所有的源程序及 `pom.xml` 文件。

6、使用 maven 开发过程

- parent/pom.xml 的定义

这是所有项目需要继承的父项目，其中定义了私服仓库地址，应用了复制文件的插件及安装依赖包的插件。

有两个 `propertie` 需要根据自己的环境修改：`user-tomcatPath`、`user.comId`

复制文件规则如下：

依赖包复制和系统配置复制：

绑定在 `compile` 目标中，当执行目录：`compile` 或者之后的目标如 `install`、`desploy` 等时均会调用。

1. 把依赖的 `jar` 包复制到 `tomcat/webapps/WEB-INF/lib`
2. 把 `eap2-parent/classes` 中配置文件复制到
`tomcat/webapps/ROOT/WEB-INF/classes`
3. 在 `tomcat/webapps/WEB-INF/SysFile/` 建立企业文件夹 `[comId]`
4. 把 `eap2-parent/SysFile` 中文件复制到
`tomcat/webapps/ROOT/WEB-INF/SysFile/[comId]` 文件夹中

如果是一个新的 `tomcat` 环境，上述复制后所需文件已齐备，`tomcat` 启动后，应用将可正常运行但缺少一些静态资源，只需要把 `resource` 文件夹一次性手工复制到 `tomcat/webapps/ROOT` 下即可。

开发过程中复制：

绑定在 `validate` 目标中，当执行 `validate` 目标时，从 `target` 文件夹复制到 `tomcat` 应用下。

7 从 svn 项目中检出的项目，转换成 maven 项目

使用命令把 eclipse 项目转换成 maven 项目：然后刷新即可。

```
ca: 管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>f:

F:\>cd work

F:\work>dir/w
驱动器 F 中的卷没有标签。
卷的序列号是 000D-5DA7

F:\work 的目录

[.]                [..]                [.metadata]         [eap-lib]
[eap2-core-common] [eap2-manager]     [eap2-office]      [eap2-parent]
[eap2-sso]          [eap2-ysy]         [eap2-ysyTest]     [eap2.0]
[eap2_bmap]        [eap2_product]    [Ibatis_test1]    [maven-jar]
[name]             [oa]               [sso]               [sso - 副本]
[Test1]            [tt]               [Ysy]
                  0 个文件          0 字节
                  23 个目录 51,620,700,160 可用字节

F:\work>cd eap2-sso

F:\work\eap2-sso>mvn eclipse:eclipse
```