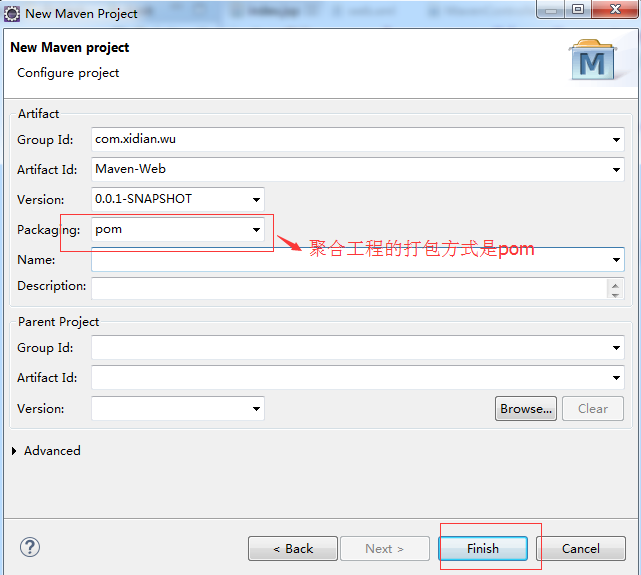
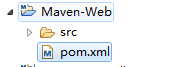
### 一 Maven创建一个聚合工程（框架）

1 创建聚合工程Maven-Web，其打包方式必须是pom

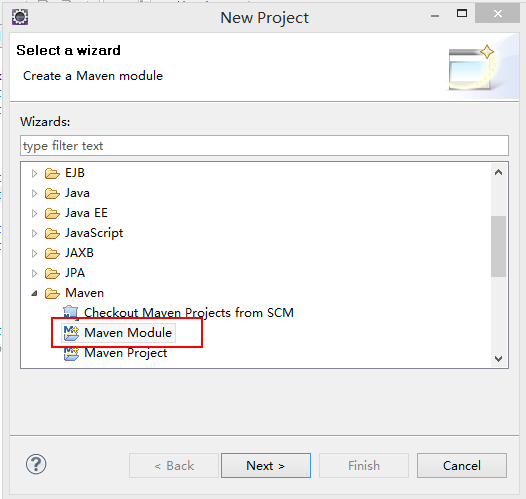


1.1 生成的项目目录：

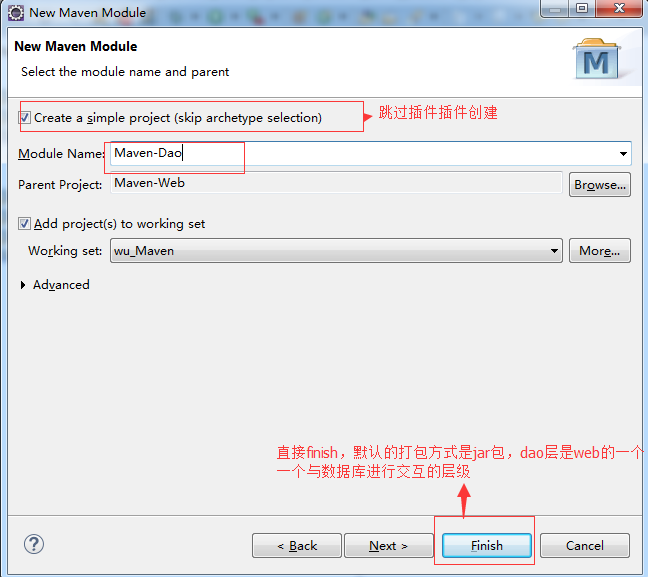


2 创建持久层工程（Maven-Dao）

第一步：在Maven-Web工程上，点击new –> other...-->maven-->Maven Model



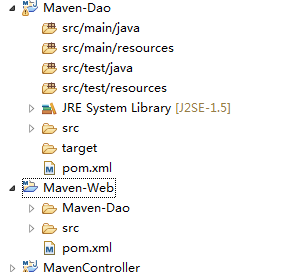
第二步：next



**Dao层的打包方式是jar包，因为它主要是和数据库进行交互，不存在Jsp页面的处理。**

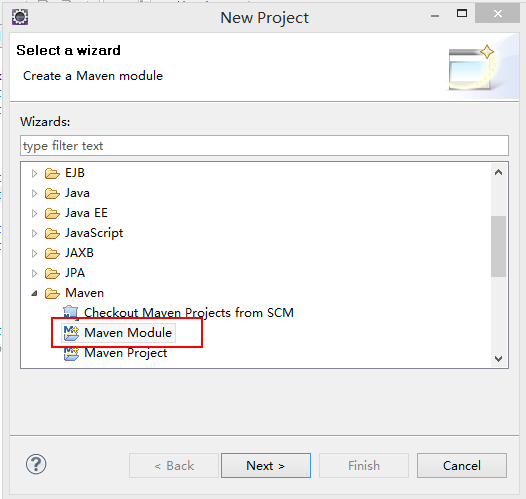
**jar文件**包括[Java](http://lib.csdn.net/base/javase" \o "Java SE知识库" \t "http://blog.csdn.net/mafan121/article/details/_blank)普通类、资源文件和普通文件，在maven中即是打包src/main/java和src/main/resources资源文件夹下的所有文件。在打包的时候会自动生成MATA-INF文件夹，用于存储maven的pom信息和MANIFEST.MF文件

第三步：生成的项目结构

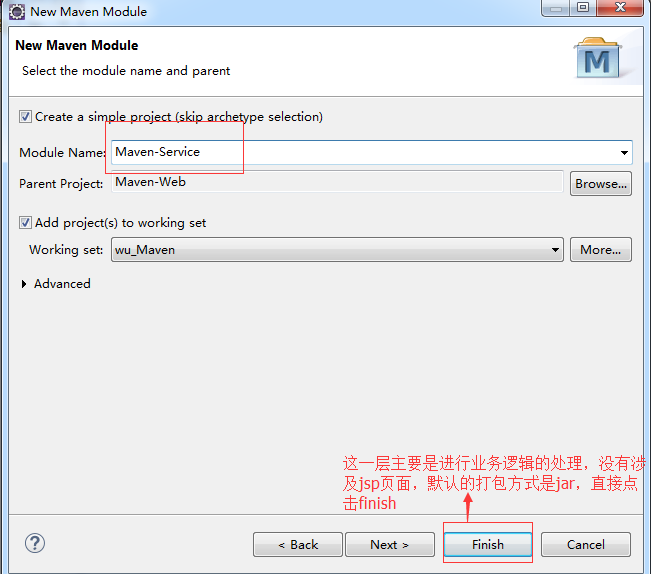


3 创建服务层工程（Maven-Service）

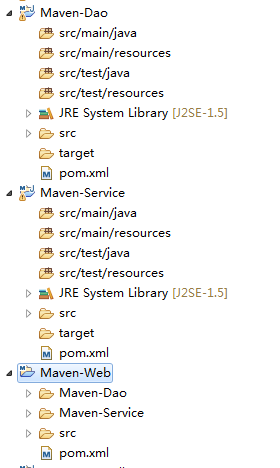
第一步：在Maven-Web工程上，点击new –> other...-->maven-->Maven Model



第二步：next

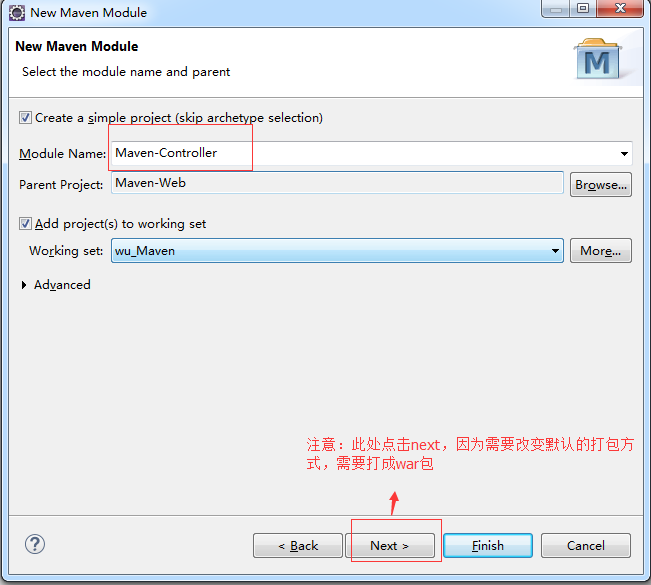


第三步：生成的项目结构:



4 创建controller层工程（Maven-Controller）

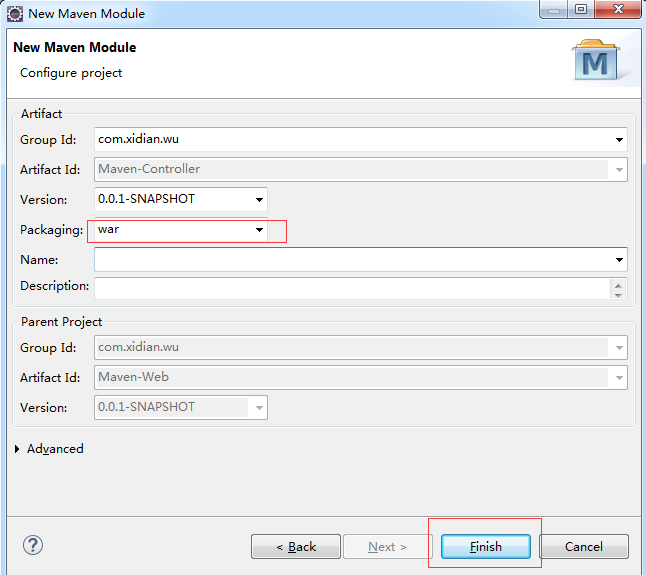
第一步： 第一步：在Maven-Web工程上，点击new –> other...-->maven-->Maven Model



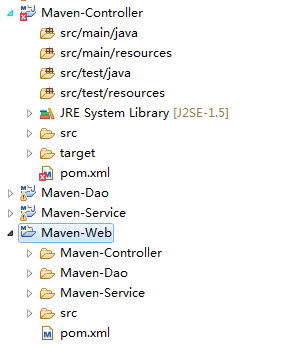
Controller层打包成war包。

**war文件**包含全部的web应用程序，即所有的java类，配置信息和jsp、js等静态资源。

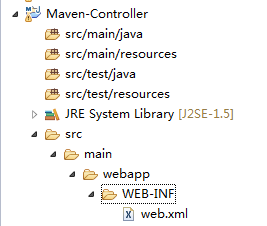
第二步：next



第三步：生成的目录结构



第四步：报错很正常，因为缺少web.xml 给Maven-Controller 添加web.xml



Web.xml 内容：

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns=*"http://java.sun.com/xml/ns/javaee"* xmlns:web=*"http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"*  xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"*  id=*"WebApp\_ID"* version=*"2.5"*>  <welcome-file-list>  <welcome-file>index.html</welcome-file>  <welcome-file>index.htm</welcome-file>  <welcome-file>index.jsp</welcome-file>  <welcome-file>default.html</welcome-file>  <welcome-file>default.htm</welcome-file>  <welcome-file>default.jsp</welcome-file>  </welcome-file-list>  </web-app> |

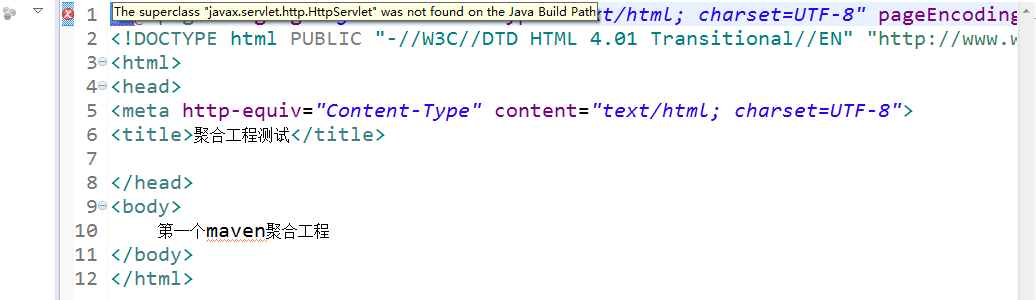
再添加index.jsp



Index.jsp 内容：

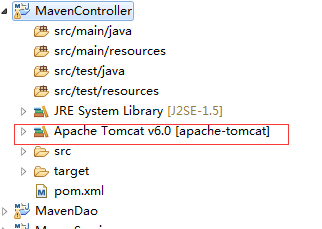
|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>  <title>聚合工程测试</title>  </head>  <body>  第一个maven聚合工程  </body>  </html> |

此时如果还是出现错误：



错误：The superclass "javax.servlet.http.HttpServlet" was not found on the Java Build Path

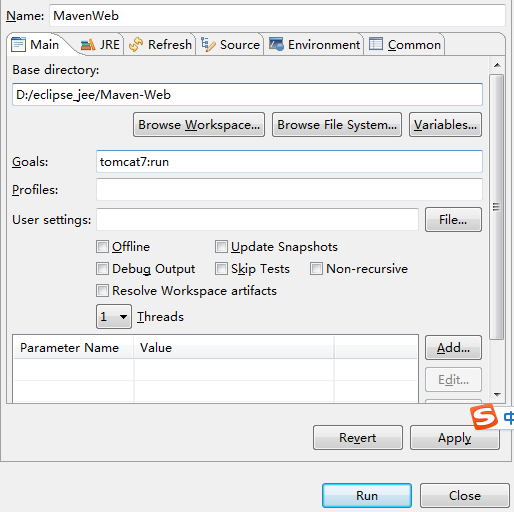
则需要导入tomcat：依赖包，导入后就ok



第五步：配置Maven-Web 的pom.xml （添加<build>标签中的的插件）：

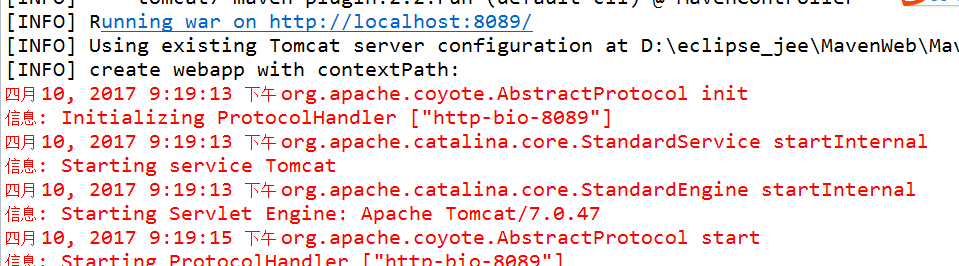
|  |
| --- |
| <project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>com.xidian.wu</groupId>  <artifactId>Maven-Web</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>pom</packaging>  <modules>  <module>Maven-Dao</module>  <module>Maven-Service</module>  <module>Maven-Controller</module>  </modules>  <build>  <plugins>  <plugin>  <!-- 配置插件 -->  <groupId>org.apache.tomcat.maven</groupId>  <artifactId>tomcat7-maven-plugin</artifactId>  <configuration>  <port>8089</port>  <path>/</path>  </configuration>  </plugin>  </plugins>  </build> |

第六步启动 ：右击Maven-Web ----> Run As--->Maven Bulid...



第七步：run

第八步：显示启动成功：



第九步：测试：

