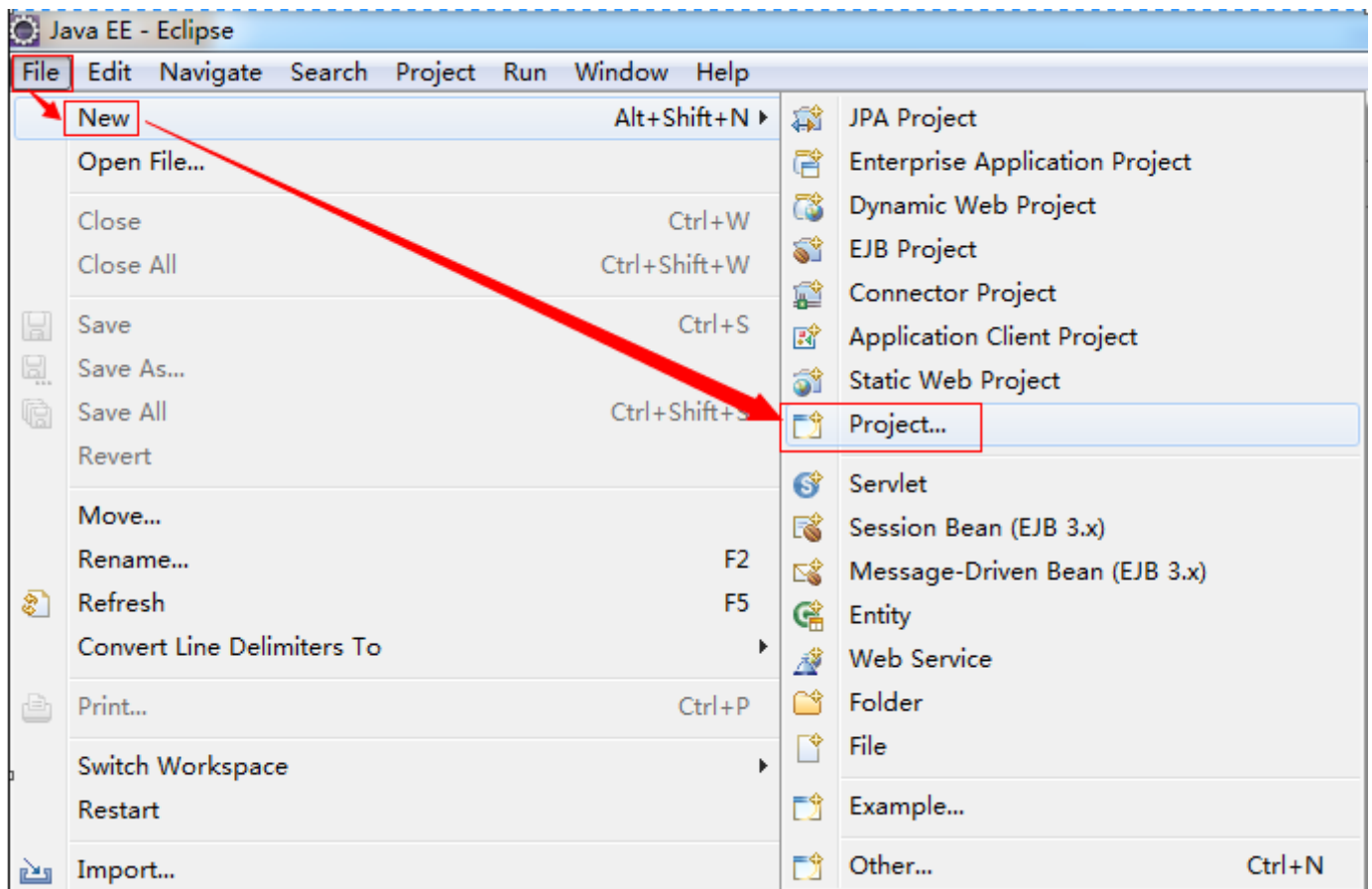


# Maven学习总结(七)——eclipse中使用Maven创建Web项目 - 孤傲苍狼

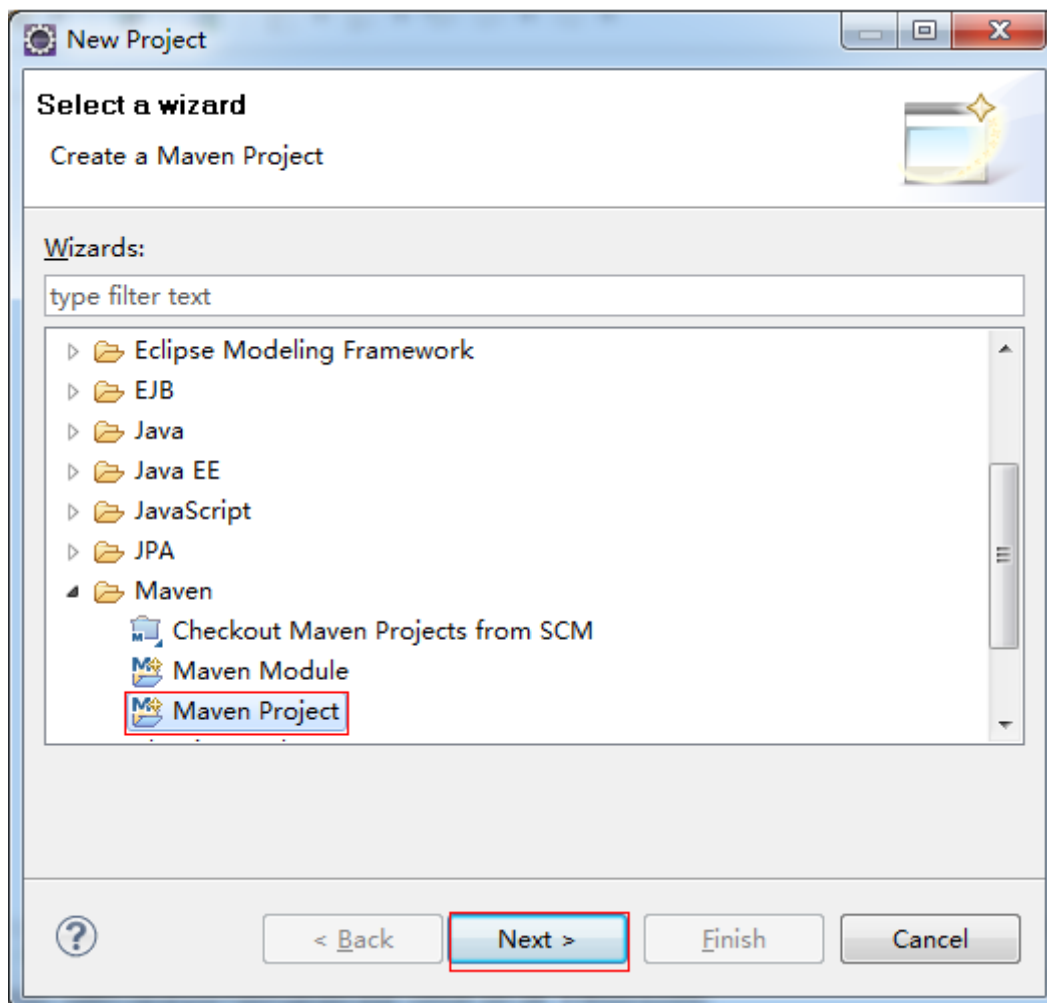
## 一、创建Web项目

### 1.1 选择建立Maven Project

选择File -> New ->Project，如下图所示：

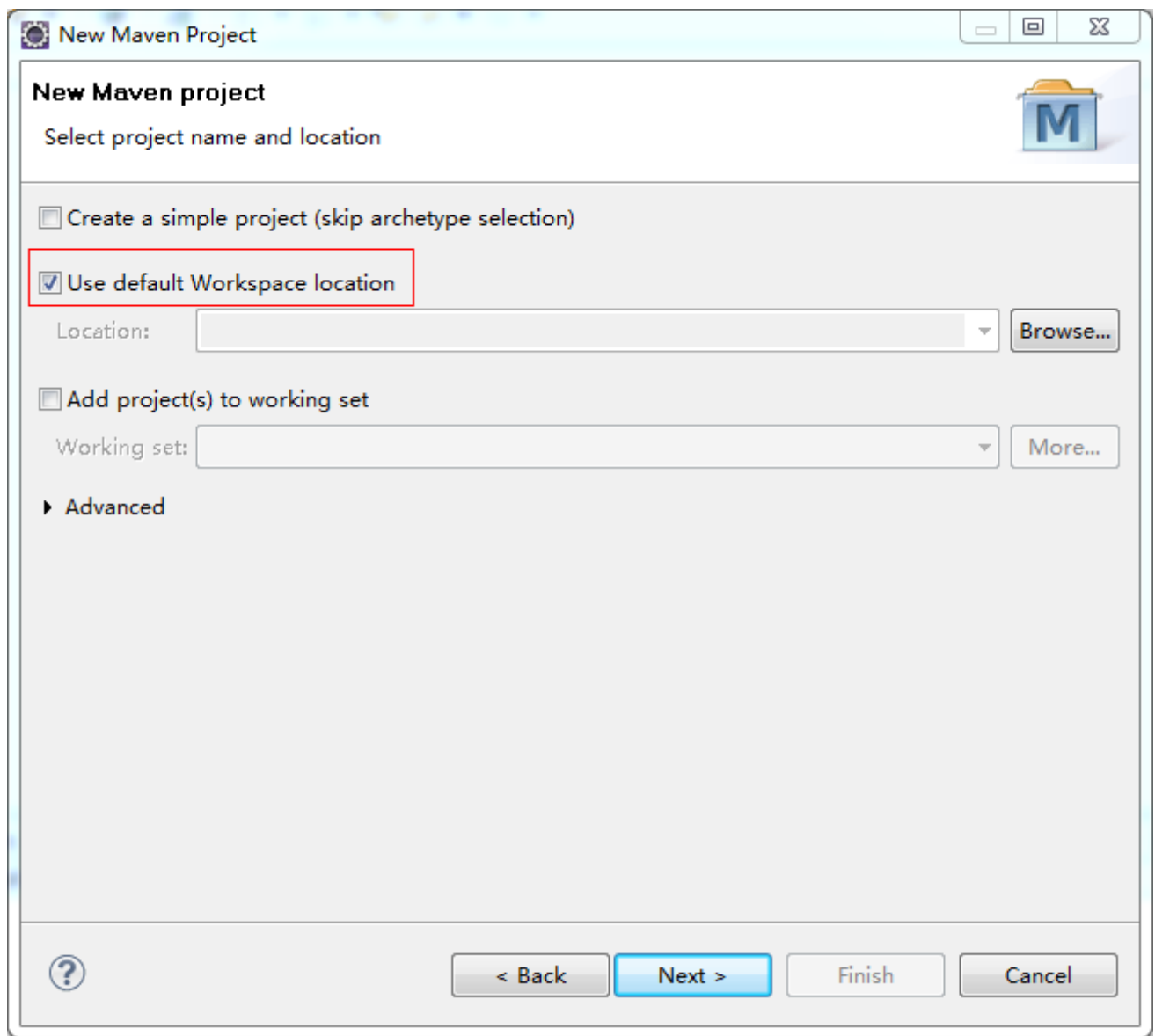


在New窗口中选择 Maven -> Maven Project。点击【next】如下图所示：



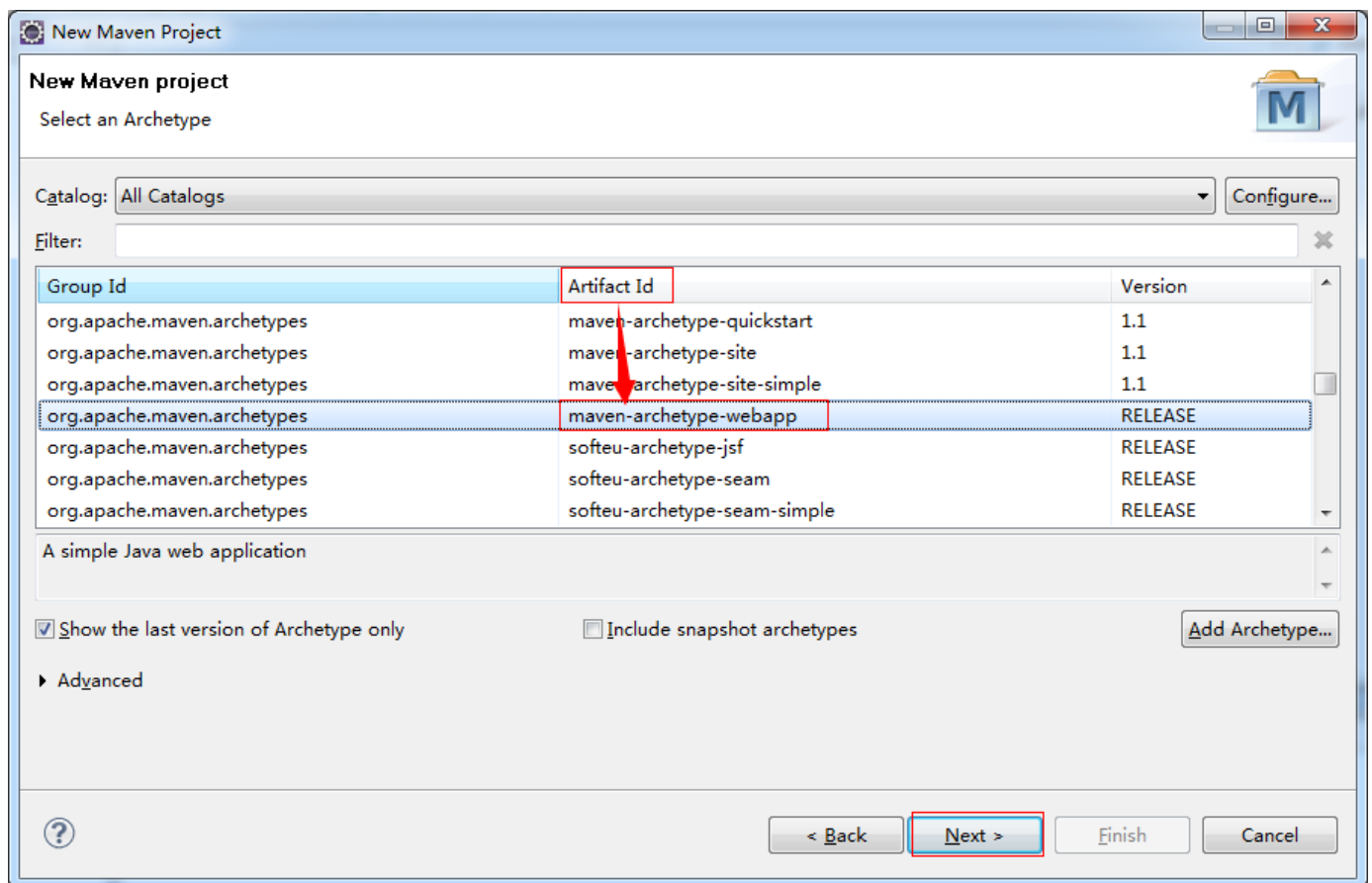
## 1.2 选择项目路径

根据项目的实际情况选择项目的存放目录，也可以选择【Use default Workspace location】默认工作空间。如下图所示：



### 1.3 选择项目类型

在Artifact Id中选择maven-archetype-webapp，如下图所示：

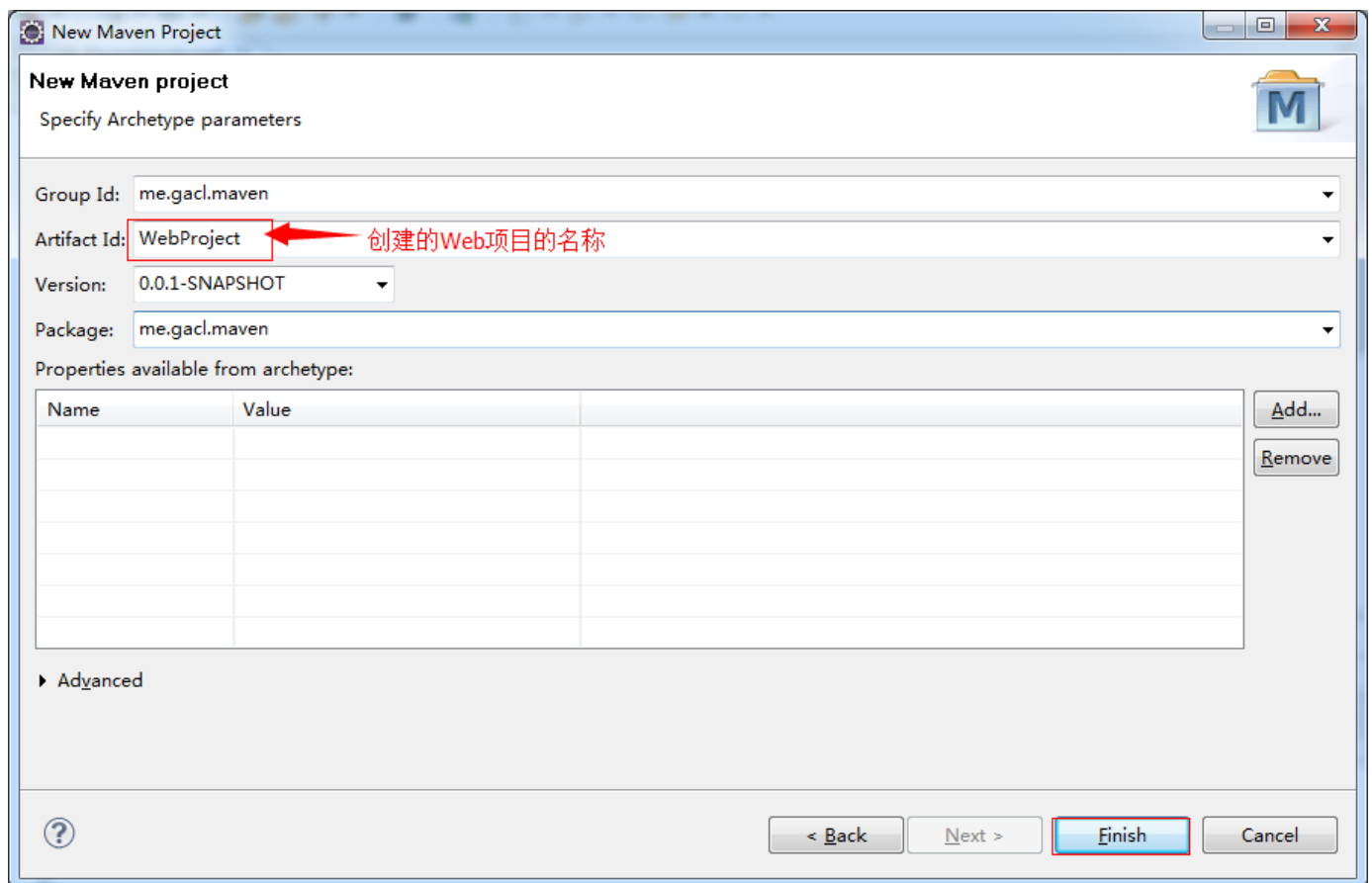


#### 1.4 输入Group ID和 Artifact ID以及Package

Group ID一般写大项目名称。Artifact ID是子项目名称。

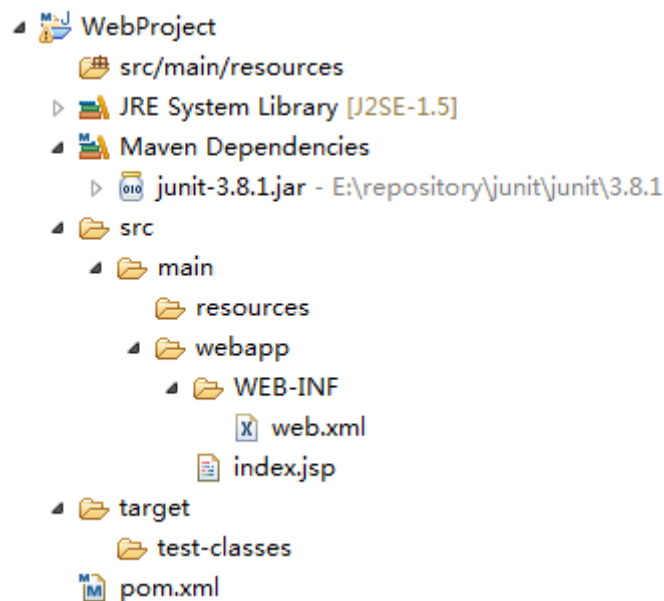
例如Spring的web包，Group ID: org.springframework, artifactId: spring-web。

Package是默认给你建一个包，不写也可以。如下图所示：



## 1.5 项目建立好后的文件结构

刚建立好后的文件结构如下图如下图所示：



## 二、将Web项目自动部署到tomcat服务器

### 2.1、在pom.xml文件中配置tomcat服务器

配置web项目的pom.xml文件，配置如下：



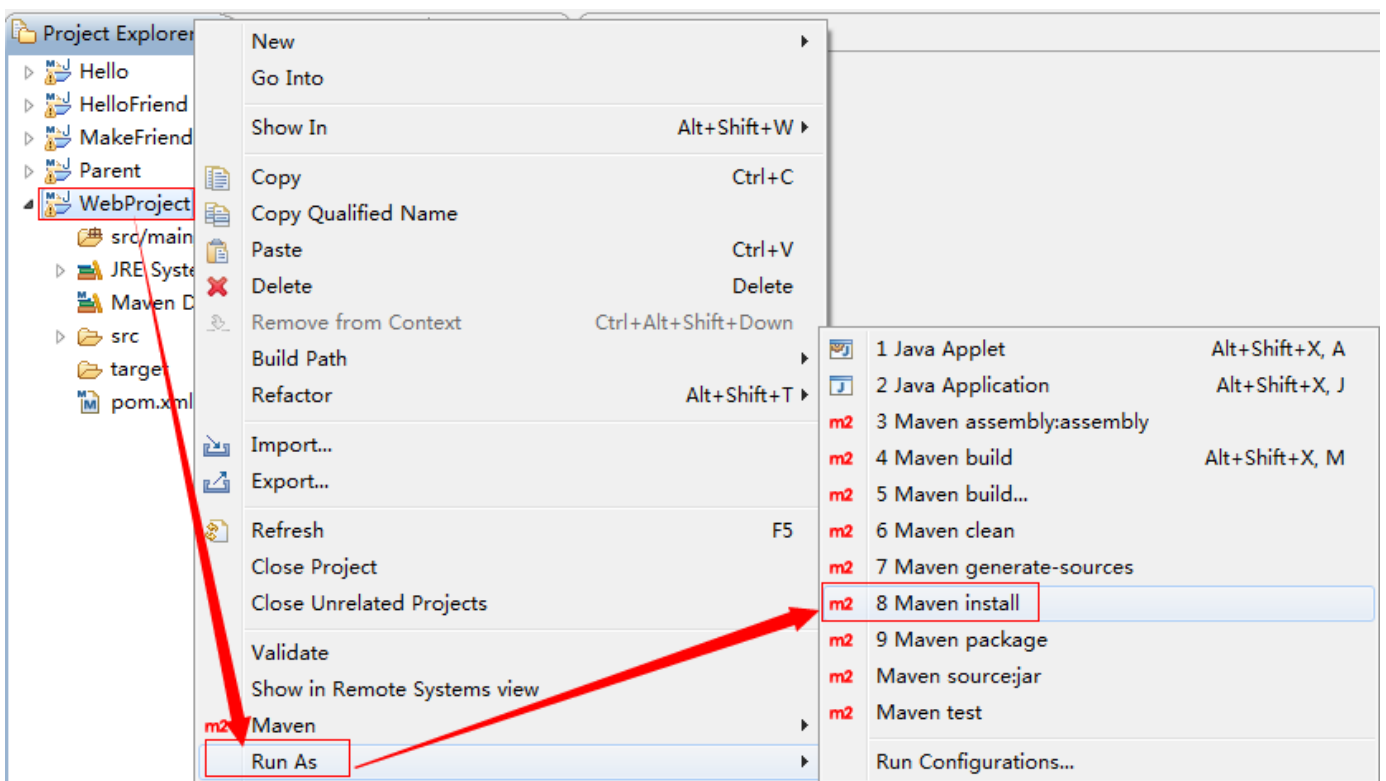
```
1 <project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4   <groupId>me.gacl.maven</groupId>
5   <artifactId>WebProject</artifactId>
6   <packaging>war</packaging>
7   <version>0.0.1-SNAPSHOT</version>
8   <name>WebProject Maven Webapp</name>
9   <url>http://maven.apache.org</url>
10  <dependencies>
11    <dependency>
12      <groupId>junit</groupId>
13      <artifactId>junit</artifactId>
14      <version>3.8.1</version>
15      <scope>test</scope>
16    </dependency>
17  </dependencies>
18
19  <!-- 将Web项目自动部署到tomcat服务器的相关 配置信息-->
20  <build>
21    <!-- 将WebProject项目打包成WebProject.war自动部署到tomcat服务器的webapps目录下面 -->
22    <finalName>WebProject</finalName>
23    <plugins>
24      <plugin>
25        <groupId>org.codehaus.cargo</groupId>
26        <artifactId>cargo-maven2-plugin</artifactId>
27        <version>1.2.3</version>
28        <configuration>
29          <container>
30            <!-- 指明使用的tomcat服务器版本 -->
31            <containerId>tomcat7x</containerId>
32            <!--指明tomcat服务器的安装目录 -->
33            <home>D:/apache-tomcat-7.0.53</home>
34          </container>
35          <configuration>
36            <type>existing</type>
37            <!--指明tomcat服务器的安装目录 -->
38            <home>D:/apache-tomcat-7.0.53</home>
39          </configuration>
40        </configuration>
41        <executions>
42          <execution>
43            <id>cargo-run</id>
44            <phase>install</phase>
45            <goals>
```

```
46         <goal>run</goal>
47     </goals>
48 </execution>
49 </executions>
50 </plugin>
51 </plugins>
52 </build>
53 </project>
```

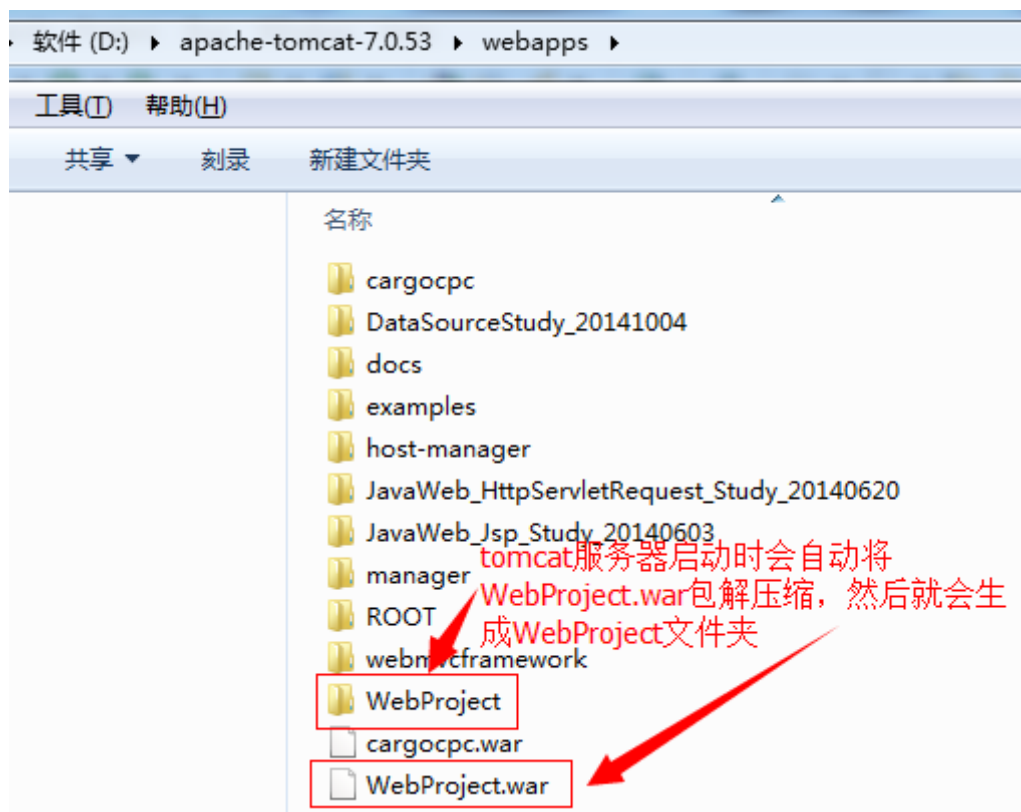


## 2.2、将web项目发布到tomca服务器的webapps目录

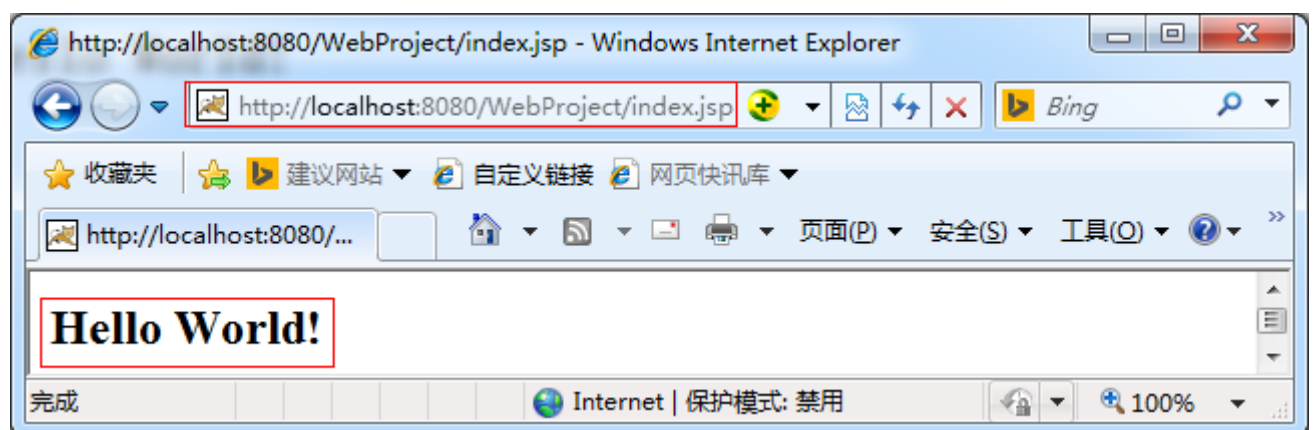
选中Web项目(或者选中Web项目的pom.xml文件)→【Run As】→【Maven install】，如下图所示：



执行完【Maven install】命令之后，就可以将WebProject项目打包成WebProject.war包发布到tomca服务器的webapps目录下，如下图所示：



测试部署好的Web项目，如下图所示：



浏览器正常输出了index.jsp页面中的内容，这说明我们的部署成功了。这就是在Eclipse中使用Maven将Web项目自动部署到tomcat服务器的过程。

在平时的Javaweb项目开发中为了便于后期的维护，我们一般会进行分层开发，最常见的就是分为domain（域模型层）、dao（数据库访问层）、service（业务逻辑层）、web（表现层），这样分层之后，各个层之间的职责会比较明确，后期维护起来也相对比较容易，今天我们就是使用Maven来构建以上的各个层。

项目结构如下：

```
system-parent
|----pom.xml
|----system-domain
|           |----pom.xml
|----system-dao
|           |----pom.xml
|----system-service
|           |----pom.xml
```



```
|----system-web
      |----pom.xml
```

一、创建system-parent项目 创建system-parent，用来给各个子模块继承。

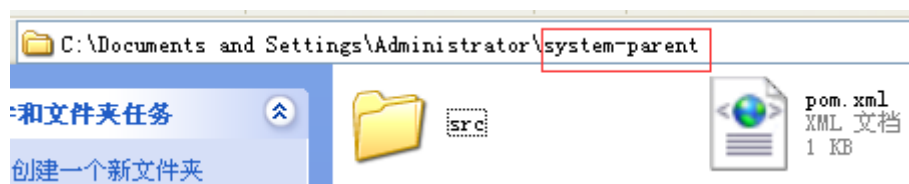
进入命令行，输入以下命令：

```
mvn archetype:create -DgroupId=me.gacl -DartifactId=system-parent -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

如下图所示：

```
C:\Documents and Settings\Administrator>mvn archetype:create -DgroupId=me.gacl -DartifactId=system-parent -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

命令执行完成之后可以看到在当前目录(C:\Documents and Settings\Administrator)生成了system-parent目录，里面有一个src目录和一个pom.xml文件，如下图所示：



将src文件夹删除，然后修改pom.xml文件，将<packaging>jar</packaging>修改为<packaging>pom</packaging>，pom表示它是一个被继承的模块，修改后的内容如下：



```
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</groupId>
6   <artifactId>system-parent</artifactId>
7   <version>1.0-SNAPSHOT</version>
8   <packaging>pom</packaging>
9
10  <name>system-parent</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
```

```
19     <groupId>junit</groupId>
20     <artifactId>junit</artifactId>
21     <version>3.8.1</version>
22     <scope>test</scope>
23 </dependency>
24 </dependencies>
25 </project>
```

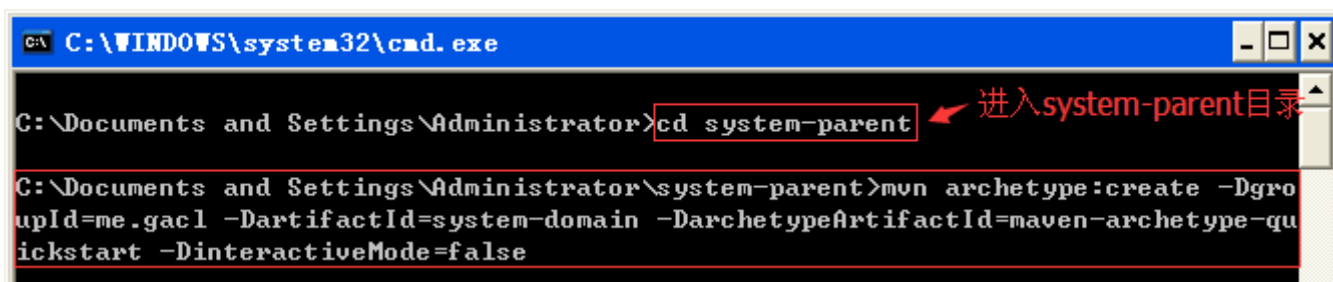


## 二、创建system-domain模块

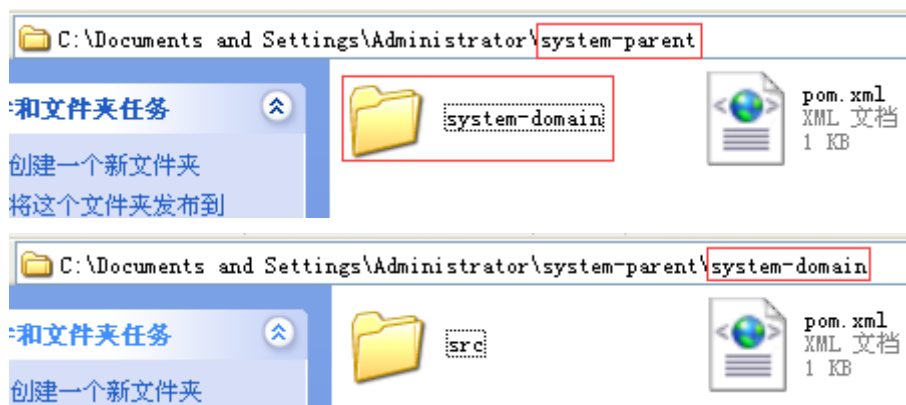
在命令行进入创建好的system-parent目录，然后执行下列命令：

```
mvn archetype:create -DgroupId=me.gacl -DartifactId=system-domain -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

如下图所示：



命令执行完成之后可以看到在system-parent目录中生成了system-domain，里面包含src目录和pom.xml文件。如下图所示：



同时，在system-parent目录中的pom.xml文件自动添加了如下内容：

```
<modules>
  <module>system-domain</module>
</modules>
```

这时，system-parent的pom.xml文件如下：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <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">
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>me.gacl</groupId>
6   <artifactId>system-parent</artifactId>
7   <version>1.0-SNAPSHOT</version>
8   <packaging>pom</packaging>
9
10  <name>system-parent</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
19      <groupId>junit</groupId>
20      <artifactId>junit</artifactId>
21      <version>3.8.1</version>
22      <scope>test</scope>
23    </dependency>
24  </dependencies>
25  <modules>
26    <module>system-domain</module>
27  </modules>
28 </project>
```



修改system-domain目录中的pom.xml文件，把<groupId>me.gacl</groupId>和<version>1.0-SNAPSHOT</version>去掉，加上<packaging>jar</packaging>，因为groupId和version会继承system-parent中的groupId和version，packaging设置打包方式为jar

修改过后的pom.xml文件如下：



```
1 <?xml version="1.0"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>me.gacl</groupId>
7         <artifactId>system-parent</artifactId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10
11     <artifactId>system-domain</artifactId>
12     <packaging>jar</packaging>
13
14     <name>system-domain</name>
15     <url>http://maven.apache.org</url>
16 </project>
```

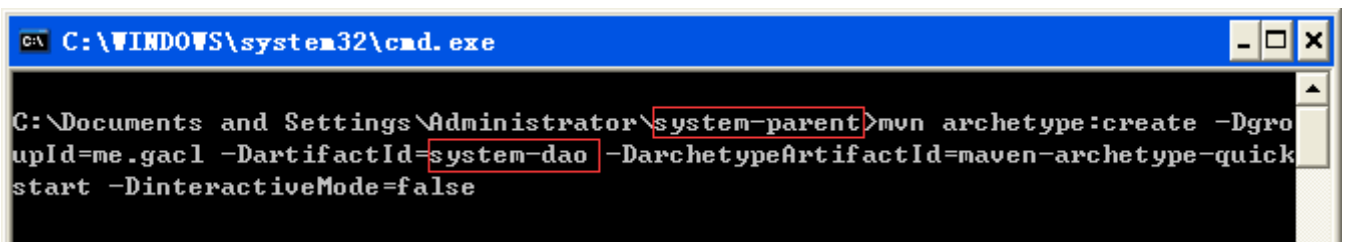


### 三、创建system-dao模块

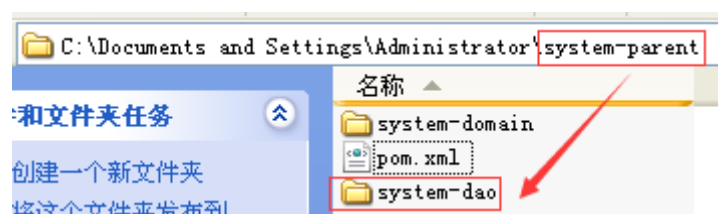
在命令行进入创建好的system-parent目录，然后执行下列命令：

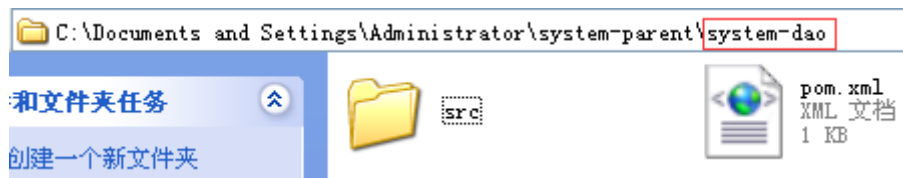
```
mvn archetype:create -DgroupId=me.gacl -DartifactId=system-dao -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

如下图所示：



命令执行完成之后可以看到在system-parent目录中生成了system-dao，里面包含src目录和pom.xml文件。如下图所示：





同时，在system-parent目录中的pom.xml文件自动变成如下内容：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <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">
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>me.gacl</groupId>
6   <artifactId>system-parent</artifactId>
7   <version>1.0-SNAPSHOT</version>
8   <packaging>pom</packaging>
9
10  <name>system-parent</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
19      <groupId>junit</groupId>
20      <artifactId>junit</artifactId>
21      <version>3.8.1</version>
22      <scope>test</scope>
23    </dependency>
24  </dependencies>
25  <modules>
26    <module>system-domain</module>
27    <module>system-dao</module>
28  </modules>
29 </project>
```



修改system-dao目录中的pom.xml文件，，把<groupId>me.gacl</groupId>和<version>1.0-SNAPSHOT</version>去掉，加上<packaging>jar</packaging>，因为groupId和version会继承system-parent中的groupId和version，packaging设置打包方式为jar，同时添加对system-domain模块的依赖，修改后的内容如下：



```
1 <?xml version="1.0"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>me.gacl</groupId>
7         <artifactId>system-parent</artifactId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10
11     <artifactId>system-dao</artifactId>
12     <packaging>jar</packaging>
13
14     <name>system-dao</name>
15     <url>http://maven.apache.org</url>
16     <properties>
17         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
18     </properties>
19     <dependencies>
20         <!--system-dao需要使用到system-domain中的类，所以需要添加对system-domain模块的依赖-->
21         <dependency>
22             <groupId>me.gacl</groupId>
23             <artifactId>system-domain</artifactId>
24             <version>${project.version}</version>
25         </dependency>
26     </dependencies>
27 </project>
```

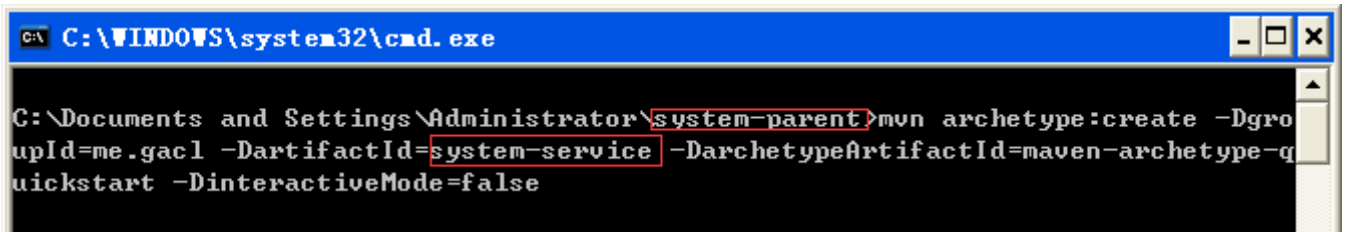


## 四、创建system-service模块

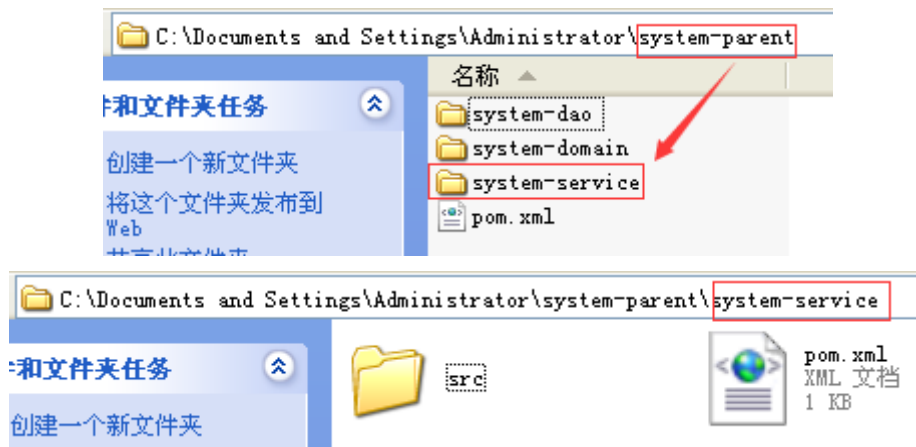
在命令行进入创建好的system-parent目录，然后执行下列命令：

```
mvn archetype:create -DgroupId=me.gacl -DartifactId=system-service -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

如下图所示：



命令执行完成之后可以看到在system-parent目录中生成了system-service，里面包含src目录和pom.xml文件。如下图所示：



同时，在system-parent目录中的pom.xml文件自动变成如下内容：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <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-
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>me.gacl</groupId>
6   <artifactId>system-parent</artifactId>
7   <version>1.0-SNAPSHOT</version>
8   <packaging>pom</packaging>
9
10  <name>system-parent</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
```

```

16
17 <dependencies>
18   <dependency>
19     <groupId>junit</groupId>
20     <artifactId>junit</artifactId>
21     <version>3.8.1</version>
22     <scope>test</scope>
23   </dependency>
24 </dependencies>
25 <modules>
26   <module>system-domain</module>
27   <module>system-dao</module>
28   <module>system-service</module>
29 </modules>
30 </project>

```



修改system-service目录中的pom.xml文件，，把<groupId>me.gacl</groupId>和<version>1.0-SNAPSHOT</version>去掉，加上<packaging>jar</packaging>，因为groupId和version会继承system-parent中的groupId和version，packaging设置打包方式为jar，同时添加对system-dao模块的依赖，system-service依赖system-dao和system-domain，但是我们只需添加system-dao的依赖即可，因为system-dao已经依赖了system-domain。修改后的内容如下：



```

1 <?xml version="1.0"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>me.gacl</groupId>
7     <artifactId>system-parent</artifactId>
8     <version>1.0-SNAPSHOT</version>
9   </parent>
10
11   <artifactId>system-service</artifactId>
12   <packaging>jar</packaging>
13
14   <name>system-service</name>
15   <url>http://maven.apache.org</url>
16   <properties>
17     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

```



```
18 </properties>
19 <dependencies>
20 <!--
21 system-service依赖system-dao和system-domain,
22 但是我们只需添加system-dao的依赖即可, 因为system-dao已经依赖了system-domain
23 -->
24 <dependency>
25 <groupId>me.gacl</groupId>
26 <artifactId>system-dao</artifactId>
27 <version>${project.version}</version>
28 </dependency>
29 </dependencies>
30 </project>
```

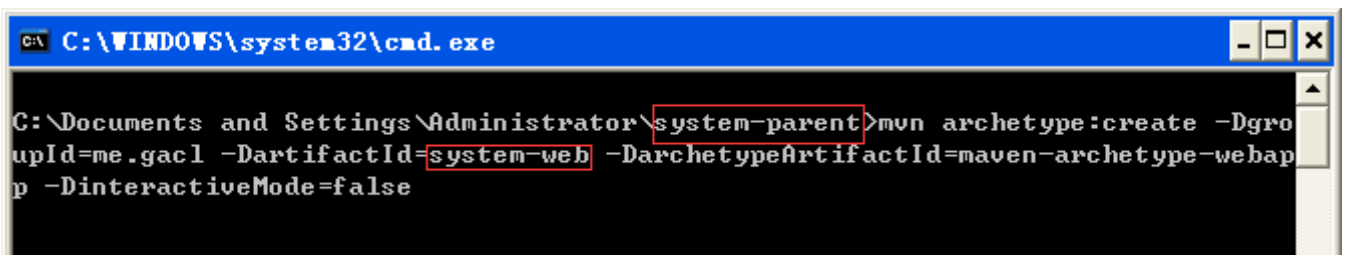


## 五、创建system-web模块

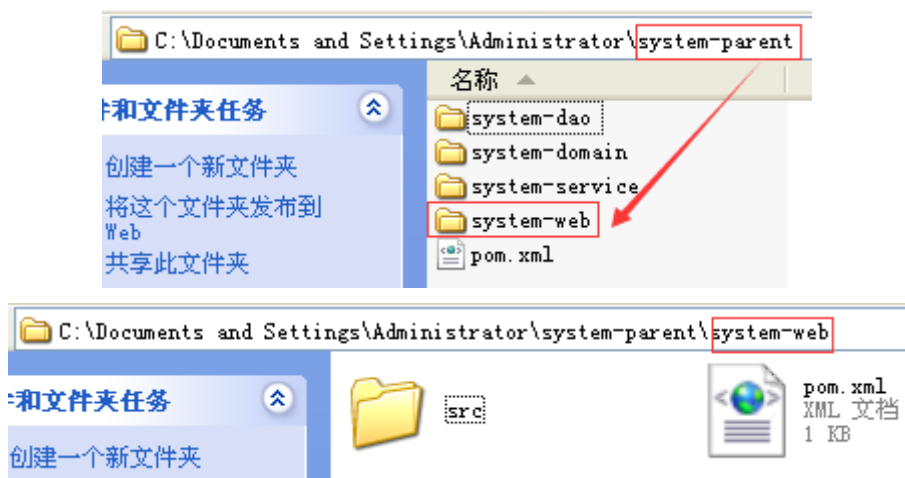
在命令行进入创建好的system-parent目录，然后执行下列命令：

```
mvn archetype:create -DgroupId=me.gacl -DartifactId=system-web -DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false
```

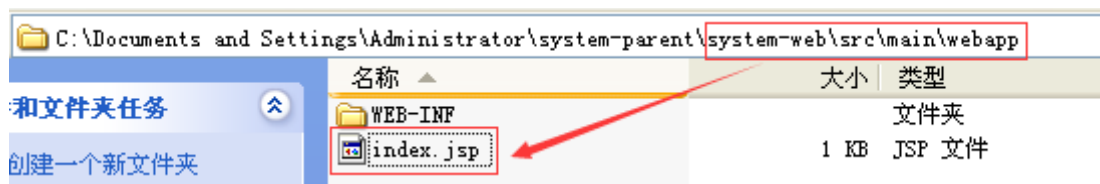
如下图所示：



命令执行完成之后可以看到在system-parent目录中生成了system-web，里面包含src目录和pom.xml文件。  
如下图所示：



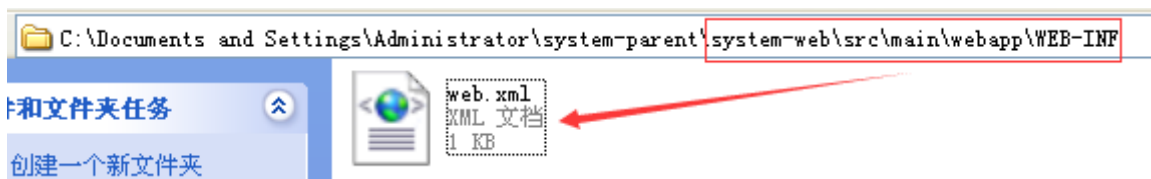
在\system-web\src\main\webapp目录中还生成了一个简单的index.jsp, 如下图所示:



里面的内容为

```
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
```

system-web\src\main\webapp\WEB-INF目录中生成了web.xml



同时, 在system-parent目录中的pom.xml文件自动变成如下内容:



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <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">
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>me.gacl</groupId>
6   <artifactId>system-parent</artifactId>
7   <version>1.0-SNAPSHOT</version>
8   <packaging>pom</packaging>
9
10  <name>system-parent</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
```

```

19     <groupId>junit</groupId>
20     <artifactId>junit</artifactId>
21     <version>3.8.1</version>
22     <scope>test</scope>
23 </dependency>
24 </dependencies>
25 <modules>
26     <module>system-domain</module>
27     <module>system-dao</module>
28     <module>system-service</module>
29     <module>system-web</module>
30 </modules>
31 </project>

```



修改system-web目录中的pom.xml文件，，把<groupId>me.gacl</groupId>和<version>1.0-SNAPSHOT</version>去掉，因为groupId和version会继承system-parent中的groupId和version，同时添加对system-service模块的依赖，修改后的内容如下：



```

1 <?xml version="1.0"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>me.gacl</groupId>
7         <artifactId>system-parent</artifactId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10
11     <artifactId>system-web</artifactId>
12     <packaging>war</packaging>
13
14     <name>system-web Maven Webapp</name>
15     <url>http://maven.apache.org</url>
16     <dependencies>
17         <!--
18         system-web依赖system-service
19         -->
20         <dependency>
21             <groupId>me.gacl</groupId>

```

```
22     <artifactId>system-service</artifactId>
23     <version>${project.version}</version>
24 </dependency>
25 </dependencies>
26 <build>
27     <finalName>system-web</finalName>
28 </build>
29 </project>
```



注意，web项目的打包方式是war。

## 六、编译运行项目

经过上面的五个步骤，相关的模块全部创建完成，怎么运行起来呢。由于最终运行的是system-web模块，所以我们对该模块添加jetty支持，方便测试运行。修改system-web项目的pom.xml如下：



```
1 <?xml version="1.0"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd" xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>me.gacl</groupId>
7         <artifactId>system-parent</artifactId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10
11     <artifactId>system-web</artifactId>
12     <packaging>war</packaging>
13
14     <name>system-web Maven Webapp</name>
15     <url>http://maven.apache.org</url>
16     <dependencies>
17         <!--
18         system-web依赖system-service
19         -->
20         <dependency>
21             <groupId>me.gacl</groupId>
22             <artifactId>system-service</artifactId>
23             <version>${project.version}</version>
```

```
24     </dependency>
25 </dependencies>
26 <build>
27     <finalName>system-web</finalName>
28     <plugins>
29         <!--配置Jetty插件-->
30         <plugin>
31             <groupId>org.mortbay.jetty</groupId>
32             <artifactId>maven-jetty-plugin</artifactId>
33         </plugin>
34     </plugins>
35 </build>
36 </project>
```



在命令行进入system-parent目录，然后执行下列命令：

```
mvn clean install
```

如下图所示：

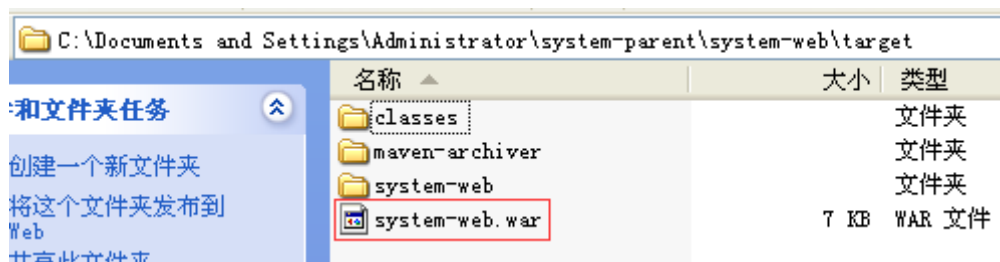
The first screenshot shows the command prompt with the command `mvn clean install` entered in the directory `C:\Documents and Settings\Administrator\system-parent`.

The second screenshot shows the output of the command. It includes information about skipping web.xml, installing the maven-jetty-plugin, and installing the system-web war and pom files. A red arrow points to the 'Reactor Summary' section, which lists the successful build of all modules: system-parent, system-domain, system-dao, system-service, and system-web Maven Webapp. The output also shows the total build time and the final memory usage.

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Administrator\system-parent>mvn clean install

C:\WINDOWS\system32\cmd.exe
[INFO] WEB-INF\web.xml already added, skipping
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ system-web ---
[INFO] Installing C:\Documents and Settings\Administrator\system-parent\system-w
eb\target\system-web.war to D:\maven\repository\me\gac\system-web\1.0-SNAPSHOT\
system-web-1.0-SNAPSHOT.war
[INFO] Installing C:\Documents and Settings\Administrator\system-parent\system-w
eb\pom.xml to D:\maven\repository\me\gac\system-web\1.0-SNAPSHOT\system-web-1.0
-SNAPSHOT.pom
[INFO] -----
[INFO] Reactor Summary:
[INFO]
[INFO] system-parent ..... SUCCESS [ 0.297 s]
[INFO] system-domain ..... SUCCESS [ 1.437 s]
[INFO] system-dao ..... SUCCESS [ 0.360 s]
[INFO] system-service ..... SUCCESS [ 0.359 s]
[INFO] system-web Maven Webapp ..... SUCCESS [ 0.422 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.985 s
[INFO] Finished at: 2015-01-22T17:06:02+08:00
[INFO] Final Memory: 14M/34M
[INFO] -----
C:\Documents and Settings\Administrator\system-parent>
```

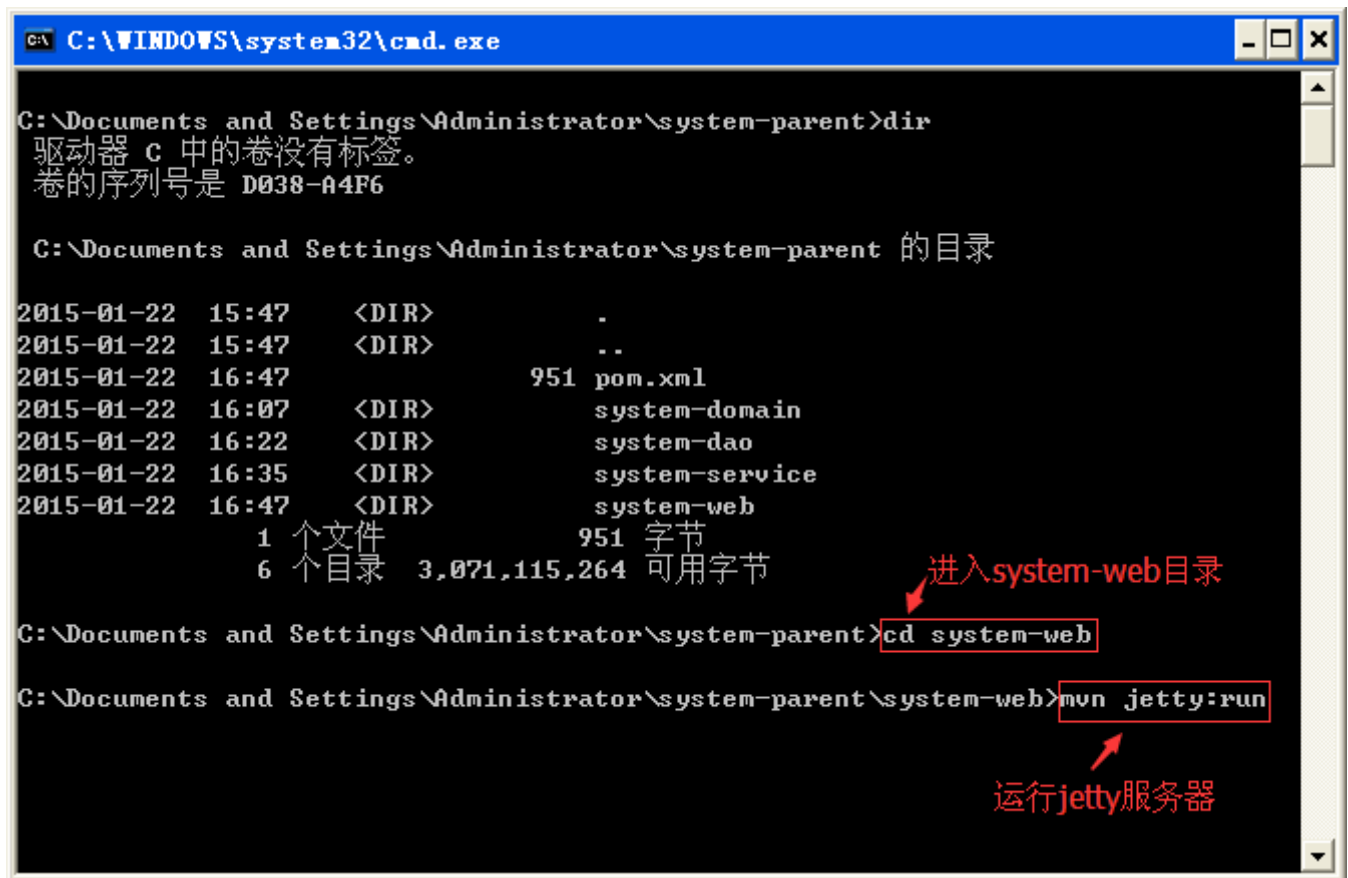
命令执行完后，在system-web目录下多出了target目录，里面有了system-web.war，如下图所示：



命令行进入system-web目录，执行如下命令，启动jetty

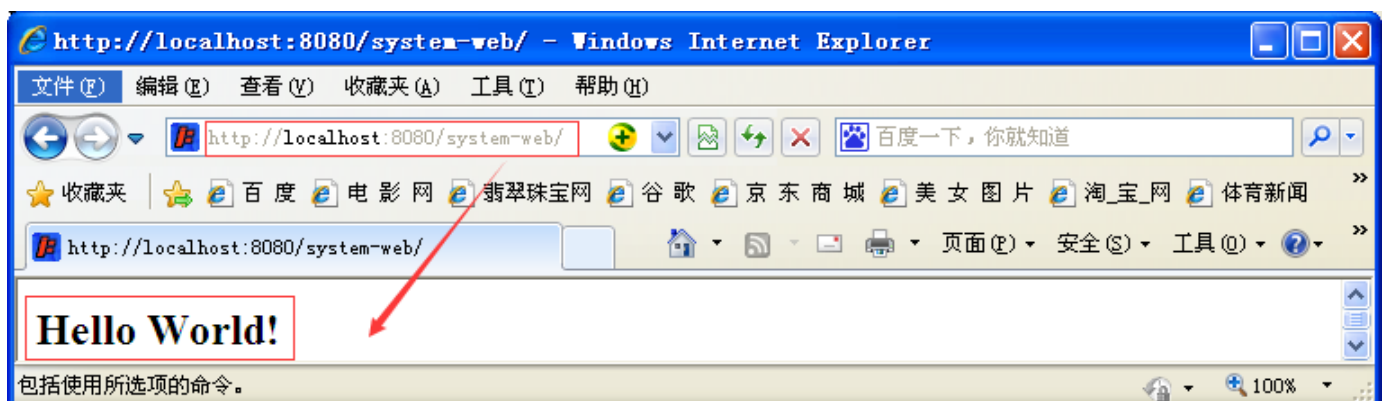
```
mvn jetty:run
```

如下图所示：



```
[INFO] Started SelectChannelConnector@0.0.0.0:8080
[INFO] Started Jetty Server
```

启动jetty服务器后，访问http://localhost:8080/system-web/ 运行结果如下图所示：



## 七、导入Eclipse中进行开发

操作步骤如下所示：

