



maven 配置及打包依赖，打包war到指定目录

原创

2015年12月26日 11:47:16

 10814

打jar包时把依赖打到jar

[html]

```
01.     <plugins>
02.
03.         <plugin>
04.             <artifactId>maven-assembly-plugin</artifactId>
05.             <configuration>
06.                 <descriptorRefs>
07.                     <descriptorRef>jar-with-dependencies</descriptorRef>
08.                 </descriptorRefs>
09.             <archive>
10.                 <manifest>
11.                     <mainClass></mainClass>
12.                 </manifest>
13.             </archive>
14.         </configuration>
15.         <executions>
16.             <execution>
17.                 <id>make-assembly</id>
18.                 <phase>package</phase>
19.                 <goals>
20.                     <goal>single</goal>
21.                 </goals>
22.             </execution>
23.         </executions>
24.     </plugin>
25.
26. </plugins>
```

把依赖和jsp等文件放在指定目录（直接是项目，不是war）

[html]



0



```
01. <plugin>
02.     <groupId>org.apache.maven.plugins</groupId>
03.     <artifactId>maven-war-plugin</artifactId>
04.     <configuration>
05.         <webappDirectory>D:/apache-tomcat-7.0.64/webapps/bj12385</webappDirectory>
06.         <warSourceDirectory>D:/apache-tomcat-7.0.64/webapps/bj12385</warSourceDirectory>
07.     </configuration>
08. </plugin>
```

以下下为互联网上搜的

就拿这个属性配置来说：

Xml代码

```
01. <properties>
02.     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
03.     <spring.framework.version>3.0.5.RELEASE</spring.framework.version>
04.     <cxfr.version>2.3.2</cxfr.version>
05.     <tiles.version>2.2.2</tiles.version>
06. </properties>
```

我需要对整个项目统一字符集编码，就需要设定<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>；
如果我需要对spring统一版本号，可以设置这么个变量
<spring.framework.version>3.0.5.RELEASE</spring.framework.version>，当然，maven不会那么乖乖的自动识别这些配置。当然，<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>默认还是可以识别的！

插件配置：

资源配置插件：

Xml代码

```
01. <plugin>
02.     <groupId>org.apache.maven.plugins</groupId>
03.     <artifactId>maven-resources-plugin</artifactId>
04.     <version>2.4.3</version>
05.     <configuration>
06.         <encoding>${project.build.sourceEncoding}</encoding>
07.     </configuration>
08. </plugin>
```



0



xml、properties文件都是资源文件，编码的时候遇到中文总要进行转码！用什么编码？UTF-8，那就记得强制
<encoding>\${project.build.sourceEncoding}</encoding>，虽然<project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>默认可以识别，但是我还是觉得不放心。

这些插件有时候即使是不配置，它也是默认存在的！

编译插件：

这个插件就是个默认配置，不过我还是谨慎的配置了jdk编译版本跟字符集设置：

Xml代码

```
01. <plugin>
02.   <groupId>org.apache.maven.plugins</groupId>
03.   <artifactId>maven-compiler-plugin</artifactId>
04.   <version>2.0.2</version>
05.   <configuration>
06.     <source>1.6</source>
07.     <target>1.6</target>
08.     <encoding>${project.build.sourceEncoding}</encoding>
09.   </configuration>
10. </plugin>
```

<source>1.6</source>：源代码编译版本；

<target>1.6</target>：目标平台编译版本；

<encoding>\${project.build.sourceEncoding}</encoding>：字符集编码。

如果你用eclipse建立maven项目时，新建的项目往往JDK版本很低——1.5！怎么办？修改settings.xml文件：

Xml代码

```
01. <profile>
02.   <id>jdk-1.6</id>
03.   <activation>
04.     <activeByDefault>true</activeByDefault>
05.     <jdk>1.6</jdk>
06.   </activation>
07.   <properties>
08.     <maven.compiler.source>1.6</maven.compiler.source>
09.     <maven.compiler.target>1.6</maven.compiler.target>
10.     <maven.compiler.compilerVersion>1.6</maven.compiler.compilerVersion>
11.   </properties>
12. </profile>
```

eclipse会很听话的构建一个JDK1.6版本的maven项目！😄



0



测试插件：

Xml代码

```
01. <plugin>
02.   <groupId>org.apache.maven.plugins</groupId>
03.   <artifactId>maven-surefire-plugin</artifactId>
04.   <version>2.7.2</version>
05.   <configuration>
06.     <parallel>methods</parallel>
07.     <threadCount>10</threadCount>
08.     <argLine>-Dfile.encoding=UTF-8</argLine>
09.     <!-- <skip>true</skip> -->
10.     <!-- <testFailureIgnore>true</testFailureIgnore> -->
11.   </configuration>
12. </plugin>
```

<parallel>methods</parallel>：方法级并发；

<threadCount>10</threadCount>：是线程数，当前是10；

<argLine>-Dfile.encoding=UTF-8</argLine>：就是参数命令行，这一行很关键。测试插件有个小问题，它不依赖项目的字符集设置，它依赖于操作系统。如果你用命令行操作，看中文提示测试结果还好。但在win+eclipse（UTF-8设置）下，那就是乱码了；

<skip>true</skip>是忽略测试；

<testFailureIgnore>true</testFailureIgnore>：忽略测试异常；

测试报告插件：

要测试，就要有报告，知晓测试覆盖率，这就需要clover：

Xml代码

```
01. <plugin>
02.   <groupId>maven</groupId>
03.   <artifactId>maven-clover-plugin</artifactId>
04.   <version>2.4.2</version>
05.   <configuration>
06.     <encoding>${project.build.sourceEncoding}</encoding>
07.   </configuration>
08. </plugin>
```

安全起见，我还是要强制写遍字符集编码设置：<encoding>\${project.build.sourceEncoding}</encoding>

可以使用 mvn cobertura:cobertura 产生测试覆盖率报告！



0



eclipse插件：

在eclipse下进行maven项目开发，还是需要eclipse相应的插件：

Xml代码

```
01. <plugin>
02.   <groupId>org.apache.maven.plugins</groupId>
03.   <artifactId>maven-eclipse-plugin</artifactId>
04.   <version>2.8</version>
05.   <configuration>
06.     <wtpmanifest>true</wtpmanifest>
07.     <wtpapplicationxml>true</wtpapplicationxml>
08.     <wtpversion>2.0</wtpversion>
09.     <downloadSources>true</downloadSources>
10.     <downloadJavadocs>true</downloadJavadocs>
11.   </configuration>
12. </plugin>
```

<wtpmanifest>true</wtpmanifest>：产生manifest文件

<wtpapplicationxml>true</wtpapplicationxml>：产生web.xml文件

<wtpversion>2.0</wtpversion>：这是eclipse里面的wtp版本

<downloadSources>true</downloadSources>：下载源代码

<downloadJavadocs>true</downloadJavadocs>：下载源码文档

虽然这么配置了，但是由maven项目转为eclipse的wtp类型的项目时，所产生的文件根本没有实际意义。web.xml文件的servlet版本很低，也许是我没有配置。对应的xml字符集的头标识也没有。

war包插件：

既然是web项目，就需要打war包，那就需要这个插件：

Xml代码

```
01. <plugin>
02.   <groupId>org.apache.maven.plugins</groupId>
03.   <artifactId>maven-war-plugin</artifactId>
04.   <version>2.1.1</version>
05.   <configuration>
06.     <encoding>${project.build.sourceEncoding}</encoding>
07.     <warName>platform</warName>
08.     <webappDirectory>${project.build.directory}/platform</webappDirectory>
09.     <warSourceDirectory>WebContent</warSourceDirectory>
10.   </configuration>
11. </plugin>
```

<encoding>\${project.build.sourceEncoding}</encoding>强制字符集编码



0



<warName>platform</warName>war包名字——platform.war
<webappDirectory>\${project.build.directory}/platform</webappDirectory>产生war前，用于存放构建war包的目录——target/platform。
<warSourceDirectory>WebContent</warSourceDirectory>：我把web工程搞成了eclipse下的WTP类型。我不喜欢maven产生的webapp目录，更喜欢WebContent！

tomcat插件：
我习惯于用tomcat作为JSP容器，对jetty多少有点陌生。不但如此，我还通过在Bamboo中配置命令让这个应用可以在tomcat中定时部署：

Xml代码

```
01. <plugin>
02.   <groupId>org.codehaus.mojo</groupId>
03.   <artifactId>tomcat-maven-plugin</artifactId>
04.   <configuration>
05.     <charset>${project.build.sourceEncoding}</charset>
06.     <url>http://localhost:8080/manager</url>
07.     <server>tomcat.server</server>
08.     <path>/platform</path>
09.     <port>8080</port>
10.     <warFile>${project.build.directory}/platform.war</warFile>
11.     <warSourceDirectory>WebContent</warSourceDirectory>
12.     <uriEncoding>${project.build.sourceEncoding}</uriEncoding>
13.   </configuration>
14. </plugin>
```

<charset>\${project.build.sourceEncoding}</charset>：字符集强制编码
<url>http://localhost:8080/manager/text</url>：我这里已经使用了Tomcat7，如果是Tomcat6就不需要/text！
<server>tomcat.server</server>这是个用户名设置，需要配置maven的settings.xml文件：

Xml代码

```
01. <servers>
02.   <server>
03.     <id>tomcat.server</id>
04.     <username>admin</username>
05.     <password>123456</password>
06.   </server>
07. </servers>
```

别急，这时候还没完！
在tomcat的tomcat-users.xml中补充相关内容：

Xml代码



0



```
01. <role rolename="manager-gui"/>
02. <role rolename="magager-script"/>
03. <user username="admin" password="123456" roles="manager-gui,manager-script" />
```

这是Tomat7的配置，在Tomcat6中应该是：

Xml代码

```
01. <role rolename="manager"/>
02. <user username="admin" password="123456" roles="manager" />
```

<port>8080</port>:运行时端口
<path>/platform</path>：是运行时路径——http://host:port/platform
<warFile>\${project.build.directory}/platform.war</warFile>：我们以war包方式发布，需要指定war包路径。
<warSourceDirectory>WebContent</warSourceDirectory>：同时要指定war包源码路径。这里我使用eclipse的WTP类型工程WebContent目录。
<uriEncoding>\${project.build.sourceEncoding}</uriEncoding>：强制字符集编码！

插件命令：

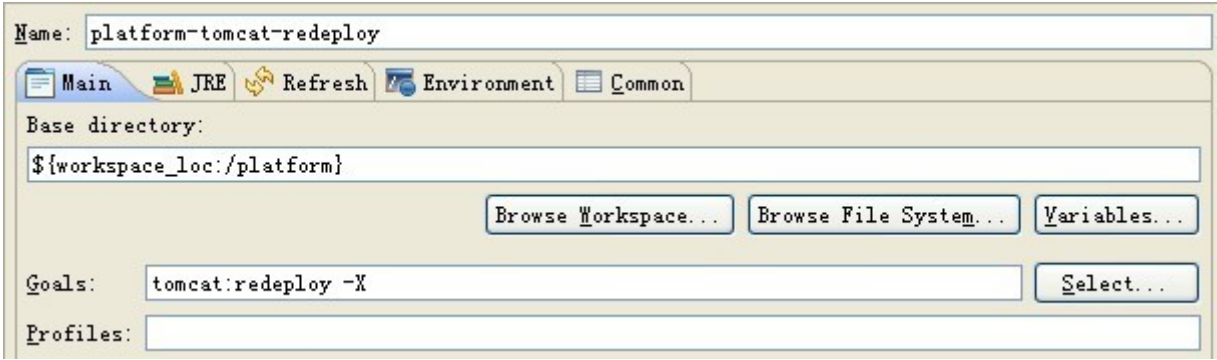
Shell代码

```
01. #Run Tomcat
02. mvn tomcat:run
03. #Stop Tomcat
04. #Deploy Tomcat
05. mvn tomcat:deploy
06. #Undeploy Tomcat
07. mvn tomcat:undeploy
08. #Redeploy Tomcat
09. mvn tomcat:redeploy
```

对于我来说，我最常用的就是mvn tomcat:run 跟 mvn tomcat:redeploy，而且我会把他们配置在eclipse中：



0



加上个-X，我可以清楚的知道每一个插件的配置，以及详细的Debug信息！

版权声明：GOOD DEEP



maven 打包排除指定文件



Gavid0124 2015年10月16日 13:57 16101

记一次我对maven深刻变革的认知。当我还不知道Maven本质上其实只是一个插件框架，当我还不知道maven有生命周期的概念，不知道maven的生命周期是抽象的，更不知道maven的每一个构建任务...

(<http://blog.csdn.net/Gavid0124/article/details/49177969>)

maven学习系列8----将resources目录下的文件打包到jar包外

maven默认情况下会把src/main/resources下的文件和class文件一起打到jar包内部，但是有很多场景下都需要把resources下的文件打包到jar包外面，这样修改resource...



ITsenlin 2016年11月09日 23:44 10380

(<http://blog.csdn.net/ITsenlin/article/details/53107304>)

Maven使用—拷贝Maven依赖jar包到指定目录

转载：一、导出到默认目录 targeted/dependency 从Maven项目中



u013514928 2017年09月11日 10:11 329

导出项目依赖的jar包：进入工程pom.xml 所在的目录下，执行如下





命令：1 mvn depe...

(<http://blog.csdn.net/u013514928/article/details/77930183>)

Maven build jar 导出指定资源和排除指定资源配置


jar导出配置如下：src/main/resources ...

 zj380475045 2017年06月27日 15:44  287

(<http://blog.csdn.net/zj380475045/article/details/73800352>)

使用maven profile指定配置文件打包适用多环境

开发过程, 我们习惯把数据源配置, 项目常量, 日志配置等基础数据配置写到一个个单独的的文件中. 如jdbc.properties等各种.格式的文件. 如何不频繁修改配置文件, 随时打包不同基础数据配...

 hjiacheng 2017年02月26日 17:18  7001

(<http://blog.csdn.net/hjiacheng/article/details/57413933>)

Maven根据不同环境打包不同配置文件 li295214001 2016年07月27日 14:42 10194

开发项目时会遇到这个问题：开发环境，测试环境，生产环境的配置文件不同，打包时经常要手动更改配置文件，更改的少还可以接受，但是如果需要更多个配置文件，手动的方法就显得非常笨重了。 下面介绍一种方...

(<http://blog.csdn.net/li295214001/article/details/52044800>)

maven 过滤文件夹打包

2016年09月24日 16:02 9.42MB

下载



Maven打包的三种方式 daiyutage 2016年12月19日 14:33 32757

Maven可以使用mvn package指令对项目进行打包，如果使用Java -jar xxx.jar执行运行jar文件，会出现"no main manifest attribute, in xx..."

(<http://blog.csdn.net/daiyutage/article/details/53739452>)

maven打war包到指定目录下和tomcat下 QH_JAVA 2016年08月14日 20:25 13031

一、maven打war包到指定目录下 初步解决方法：maven中更改target目录可以用子目录，但是只能是相对于当前项目的目录，虽然也能将war包打到项目外的目录下面，但是项目下会出现一个奇...

(http://blog.csdn.net/QH_JAVA/article/details/52205448)



0



java项目中利用maven打war包，其pom配置



hgg923

2016年10月13日 09:35

📖 8208

1.8 4.1.6.RELEASE 5.1.30 XXX \${project.basedir}/src/main/resources true ...

(<http://blog.csdn.net/hgg923/article/details/52804006>)

maven的web工程打包为war并部署到服务器



u011314442

2017年03月20日 14:24

📖 6381

1.在maven工程上右键 --> export --> 选择WAR file --> next 2. 点击Browse... 选择导出后存放位置 3. 将工程名改为ROOT.war...

(<http://blog.csdn.net/u011314442/article/details/64124489>)

Maven 项目打包需要注意到的那点事儿



defonds

2015年01月28日 17:04

📖 114604

关于 Maven 打 war 包《使用 Eclipse 的 Maven 2 插件开发一个 JEE 项目》详细介绍了如何在 Eclipse 使用 Maven 新建一个 JEE 项目并对其进行断点跟踪调试...

(<http://blog.csdn.net/defonds/article/details/43233131>)

Maven 的 Web 项目使用 war 插件针对不同环境打包

这一节我们介绍 Maven 的 Web 项目使用 war 插件针对不同环境打包。这是我在 BAE 部署自己的博客应用的时候整理的一种方案，供大家参考。最重要的思路其实就一条：打包之前替换配置文件，从...



lw_power

2016年09月10日 15:49

📖 6250

(http://blog.csdn.net/lw_power/article/details/52495296)

恭喜：一个公式教你秒懂天下英语

老司机教你一个数学公式秒懂天下英语



maven打包，实现将jar包中的路径，打到对应的目录下

如果你想通过pom.xml文件的配置实现的话，你可以这样 1、打jar包时过滤配置文件 src/main/resources **/* true ...



wangliutao1


2017年03月04日 10:59

📖 1588

(<http://blog.csdn.net/wangliutao1/article/details/60322463>)

一段实用的maven pom:将项目依赖打包到文件夹或打入jar中


使用了maven-assembly-plugin插件和maven-dependency-plugin，将依赖打入jar和文件夹

 aitangyong 2016年09月13日 16:15 📖 6937

(<http://blog.csdn.net/aitangyong/article/details/52526876>)

MAVEN将所有依赖打进一个jar包并复制到指定目录的简单示例

pom的配置信息：

 blizzardlyk 2016年08月13日 14:39 📖 3877

(<http://blog.csdn.net/blizzardlyk/article/details/52199247>)

Maven打包，指定classes路径

2013年04月02日 10:30 9KB [下载](#)



maven实现按需打包指定接口

 BeMoreQuiet 2017年10月21日 10:02 📖 461

Maven实现接口按需打包。


(<http://blog.csdn.net/BeMoreQuiet/article/details/78301540>)

恭喜：一个公式教你秒懂天下英语

老司机教你一个数学公式秒懂天下英语




maven 配置及打包依赖，打包war到指定目录

 focusjava 2016年11月15日 16:14 📖 1264

打jar包时把依赖打到jar [html] view plain copy plugins> plugin> ...

(<http://blog.csdn.net/focusjava/article/details/53172463>)

更改maven打包文件的默认输出

 rj042 2011年09月29日 18:36 📖 37927

众所周知，maven是个项目管理工具，maven是个好东西啊，这里就不多介绍了，网上有很多！这里我主要是讲一下在使用maven之后的一些心得。 开发j2ee Web项目的同学，在开...

(<http://blog.csdn.net/rj042/article/details/6834557>)