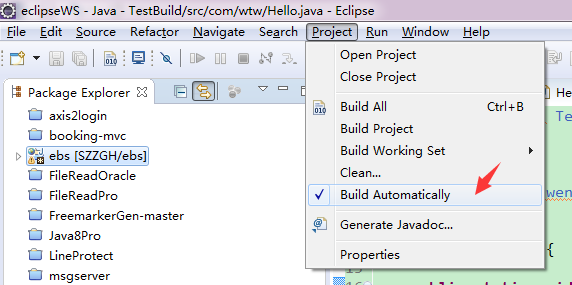
# 一、关于Eclipse编译方面的问题

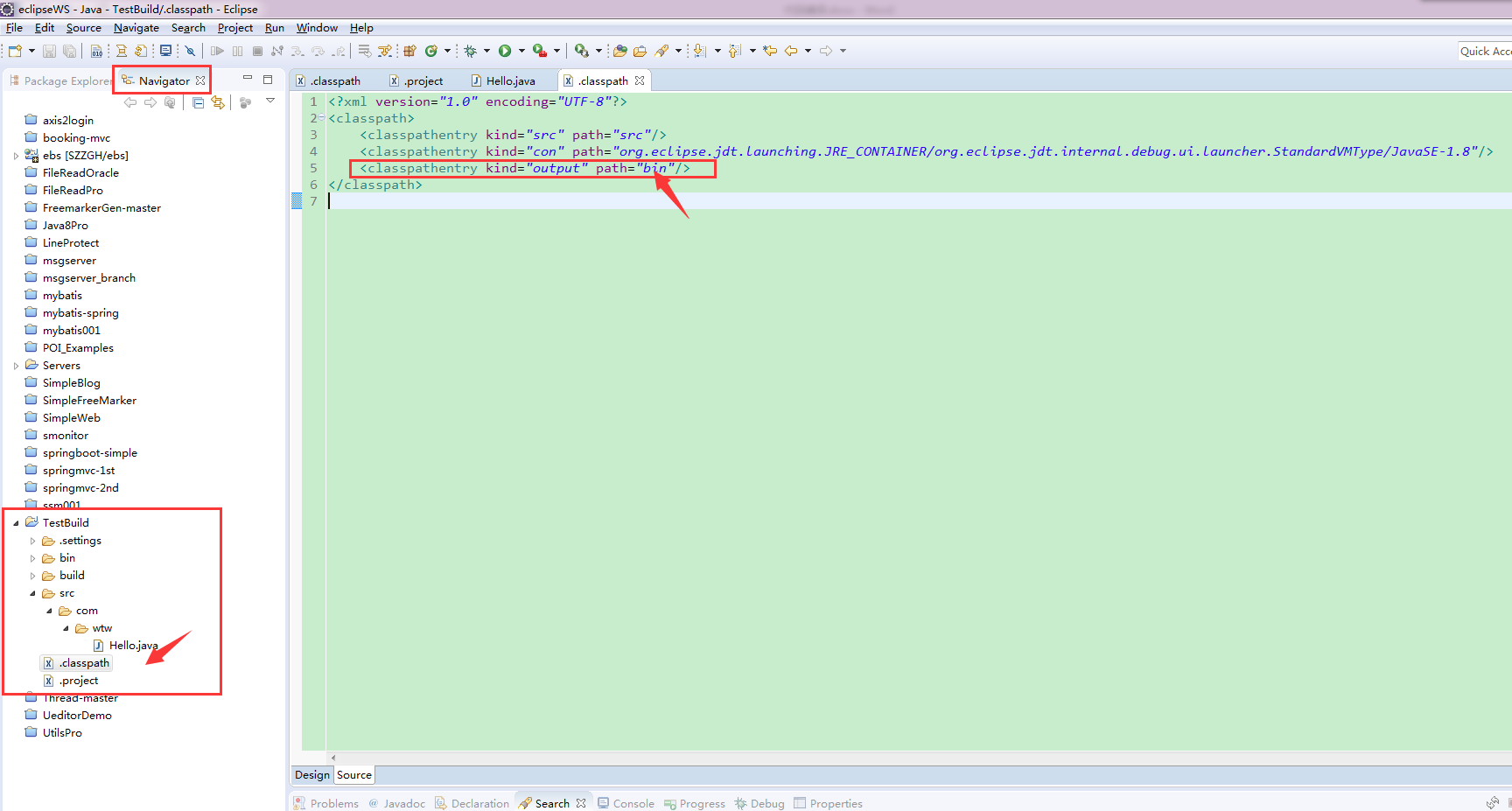
1. 自动编译



当你在编写完一个java文件并保存时，eclipse会自动编译该java文件，生成class文件。

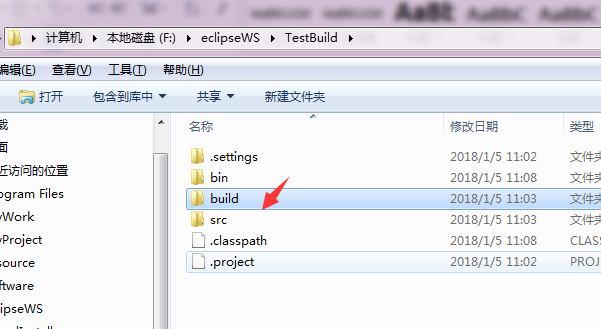
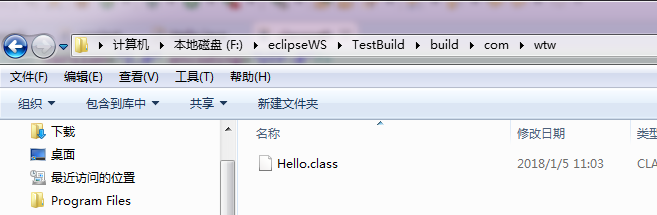
1. class文件生成到哪个目录？

当我们在Eclipse中新建一个项目时，会自动生成.classpath和.project文件。



这里的bin目录表示编译后的class文件存放目录。可修改为其他目录，如改成build，

之后的class文件都会生成到build目录下。

# 二、Apache Ant

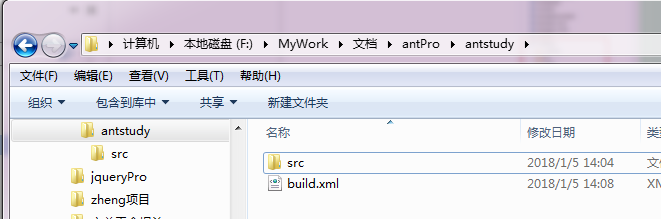
1.下载Ant <http://ant.apache.org/>

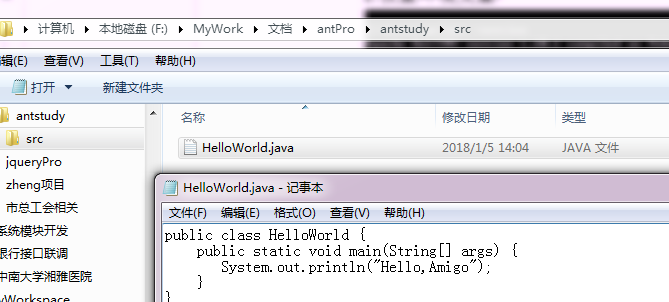
2.设置环境变量



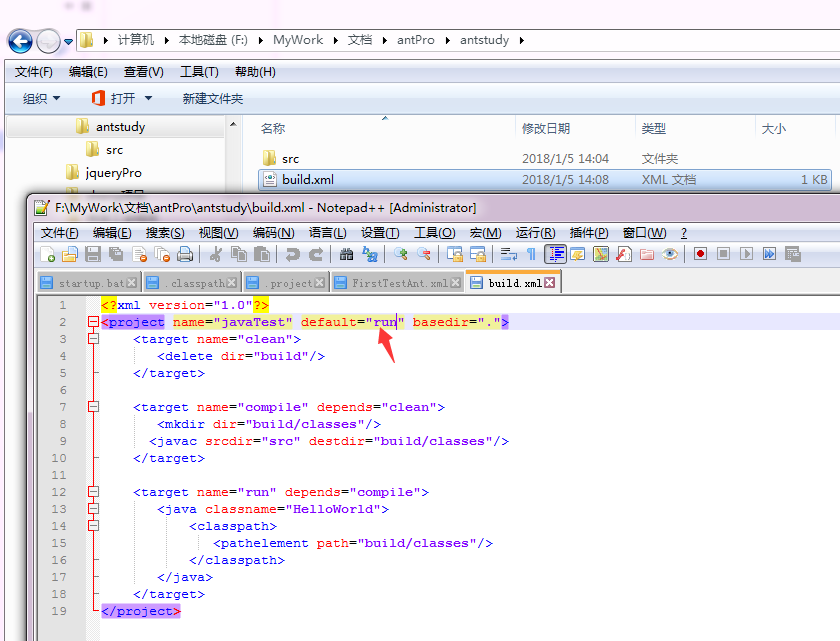
## Ant 编译Java项目

1. 建立Java项目

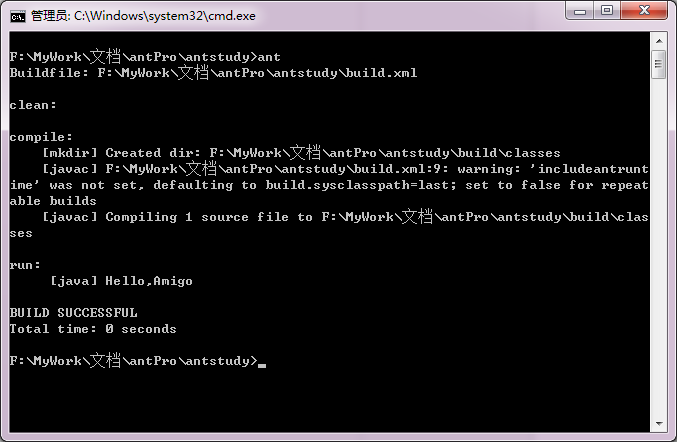


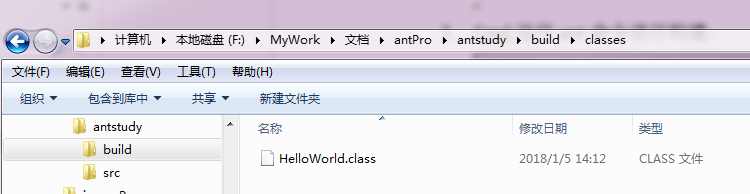


1. 编写ant build.xml脚本



1. Cmd运行ant命令进行构建





## Ant –Writing a Simple Buildfile

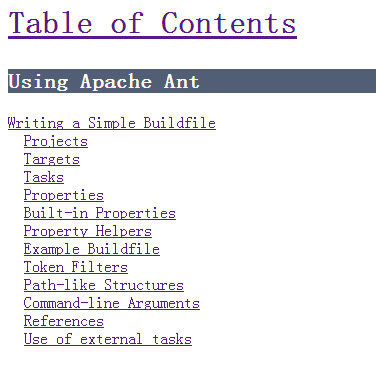
Ant的build.xml的写法参考官方网站： <http://ant.apache.org/manual/index.html>

中文博客可参考 <https://www.cnblogs.com/jenniferhuang/p/3865132.html>

Ant命令相对简单，ant [options] [target [target2 [target3] ...]]

可参考：http://ant.apache.org/manual/index.html

|  |
| --- |
| 核心结构：  <project>  <Properties>自定义的属性/内置的属性</ Properties>  <target>  <task名>内建了很多Task，参考<http://ant.apache.org/manual/tasklist.html>，  如javac，jar，mkdir等等。  </task名>  </target>  </project> |

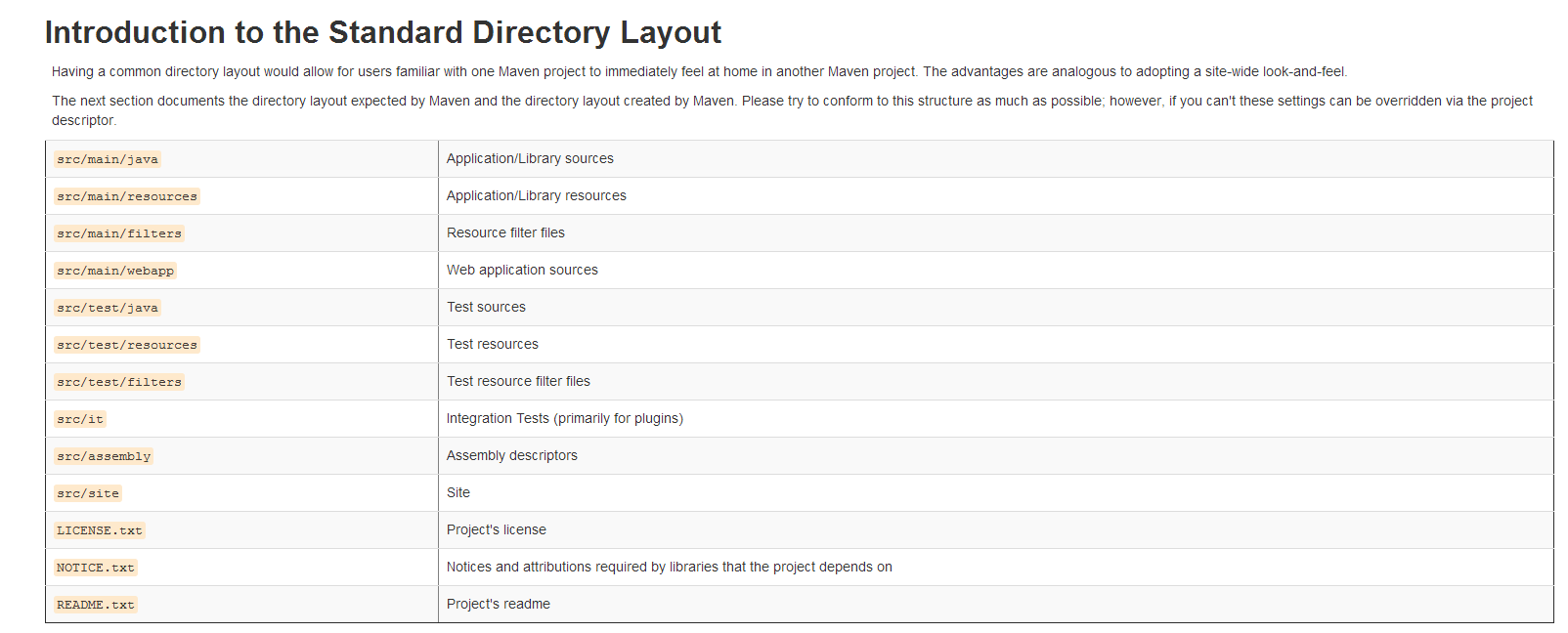




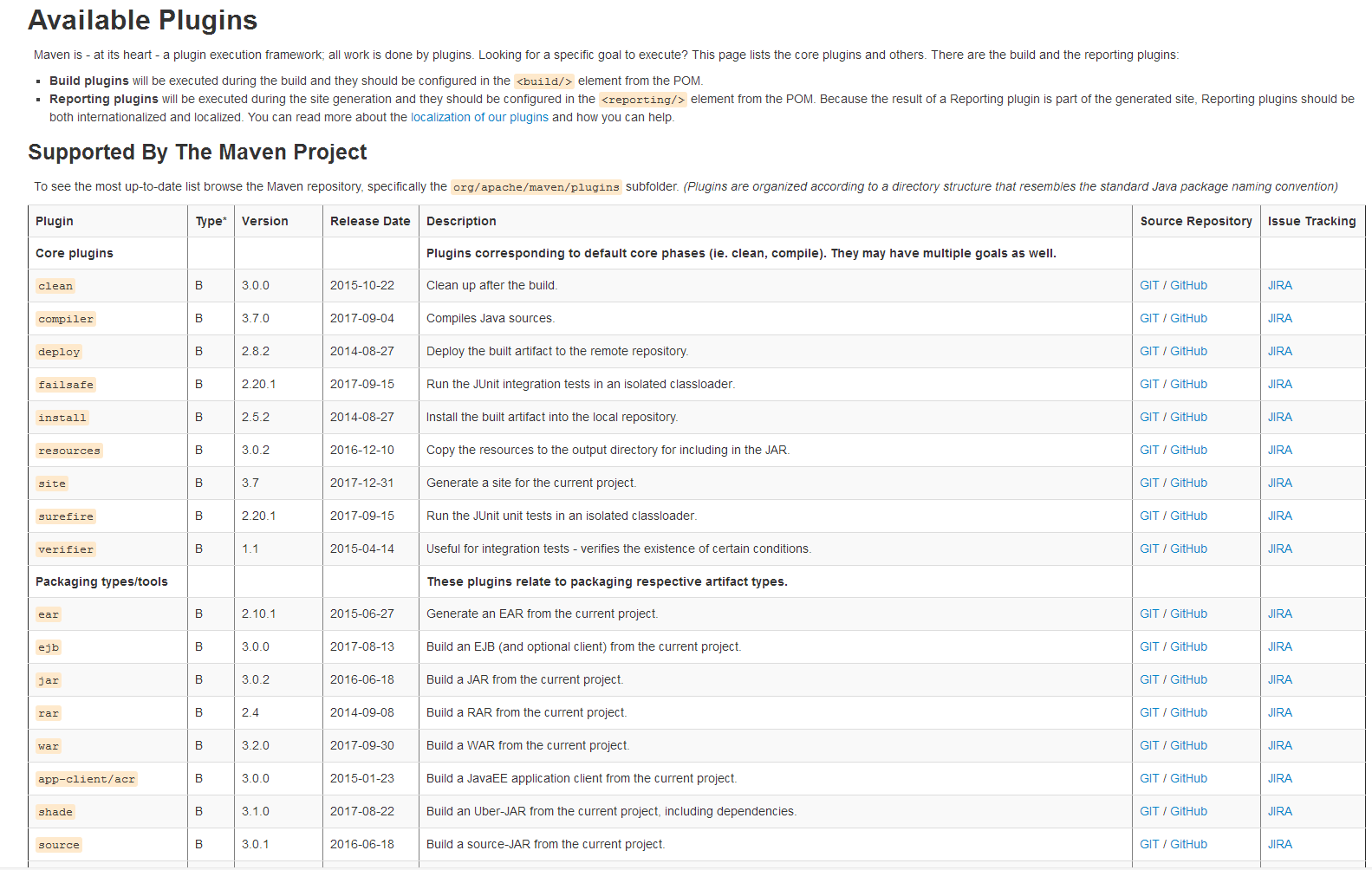
# 三、 Apache Maven

官网 <http://maven.apache.org/>

## Maven标准目录结构：



## Maven插件：



## Maven Phases：

Although hardly a comprehensive list, these are the most common *default* lifecycle phases executed.

* **validate**: validate the project is correct and all necessary information is available
* **compile**: compile the source code of the project
* **test**: test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
* **package**: take the compiled code and package it in its distributable format, such as a JAR.
* **integration-test**: process and deploy the package if necessary into an environment where integration tests can be run
* **verify**: run any checks to verify the package is valid and meets quality criteria
* **install**: install the package into the local repository, for use as a dependency in other projects locally
* **deploy**: done in an integration or release environment, copies the final package to the remote repository for sharing with other developers and projects.

There are two other Maven lifecycles of note beyond the *default* list above. They are

* **clean**: cleans up artifacts created by prior builds
* **site**: generates site documentation for this project

Phases are actually mapped to underlying goals. The specific goals executed per phase is dependant upon the packaging type of the project. For example, *package* executes *jar:jar* if the project type is a JAR, and *war:war* if the project type is - you guessed it - a WAR.

## Maven Setting 配置：

<http://maven.apache.org/settings.html> .

## Maven POM参考（非常重要），类似于ant的build.xml:

POM stands for "Project Object Model"

<http://maven.apache.org/pom.html>

关于Maven的配置细节具体参考官方文档，如build的具体配置。

Quick Overview

|  |
| --- |
| 1. <project xmlns="http://maven.apache.org/POM/4.0.0" 2. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" 3. xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 4. http://maven.apache.org/xsd/maven-4.0.0.xsd"> 5. <modelVersion>4.0.0</modelVersion> 7. <!-- The Basics --> 8. <groupId>...</groupId> 9. <artifactId>...</artifactId> 10. <version>...</version> 11. <packaging>.（ pom, jar, maven-plugin, ejb, war, ear, rar, par.）..</packaging> 12. <dependencies>...</dependencies> 13. <parent>...</parent> 14. <dependencyManagement>...</dependencyManagement> 15. <modules>...</modules> 16. <properties>...</properties> 18. <!-- Build Settings --> 19. <build>...</build> 20. <reporting>...</reporting> 22. <!-- More Project Information --> 23. <name>...</name> 24. <description>...</description> 25. <url>...</url> 26. <inceptionYear>...</inceptionYear> 27. <licenses>...</licenses> 28. <organization>...</organization> 29. <developers>...</developers> 30. <contributors>...</contributors> 32. <!-- Environment Settings --> 33. <issueManagement>...</issueManagement> 34. <ciManagement>...</ciManagement> 35. <mailingLists>...</mailingLists> 36. <scm>...</scm> 37. <prerequisites>...</prerequisites> 38. <repositories>...</repositories> 39. <pluginRepositories>...</pluginRepositories> 40. <distributionManagement>...</distributionManagement> 41. <profiles>...</profiles> 42. </project> |

That is not to say that the POM cannot affect the flow of the lifecycle - it can. For example, by configuring the maven-antrun-plugin, one can effectively embed ant tasks inside of the POM.—Maven中可以嵌入Ant

## Maven scope：

**scope**:  
This element refers to the classpath of the task at hand (compiling and runtime, testing, etc.) as well as how to limit the transitivity of a dependency. There are five scopes available:

* **compile** - this is the default scope, used if none is specified. Compile dependencies are available in all classpaths. Furthermore, those dependencies are propagated to dependent projects.
* **provided** - this is much like compile, but indicates you expect the JDK or a container to provide it at runtime. It is only available on the compilation and test classpath, and is not transitive.
* **runtime** - this scope indicates that the dependency is not required for compilation, but is for execution. It is in the runtime and test classpaths, but not the compile classpath.
* **test** - this scope indicates that the dependency is not required for normal use of the application, and is only available for the test compilation and execution phases. It is not transitive.
* **system** - this scope is similar to provided except that you have to provide the JAR which contains it explicitly. The artifact is always available and is not looked up in a repository.

Maven命令：

|  |
| --- |
| 依赖范围控制哪些依赖在哪些classpath 中可用，哪些依赖包含在一个应用中。让我们详细看一下每一种范围：  compile （编译范围）  compile是默认的范围；如果没有提供一个范围，那该依赖的范围就是编译范围。编译范围依赖在所有的classpath 中可用，同时它们也会被打包。  provided （已提供范围）  provided 依赖只有在当JDK 或者一个容器已提供该依赖之后才使用。例如， 如果你开发了一个web 应用，你可能在编译 classpath 中需要可用的Servlet API 来编译一个servlet，但是你不会想要在打包好的WAR 中包含这个Servlet API；这个Servlet API JAR 由你的应用服务器或者servlet 容器提供。已提供范围的依赖在编译classpath （不是运行时）可用。它们不是传递性的，也不会被打包。  runtime （运行时范围）  runtime 依赖在运行和测试系统的时候需要，但在编译的时候不需要。比如，你可能在编译的时候只需要JDBC API JAR，而只有在运行的时候才需要JDBC  驱动实现。  test （测试范围）  test范围依赖 在一般的编译和运行时都不需要，它们只有在测试编译和测试运行阶段可用。  system （系统范围）  system范围依赖与provided 类似，但是你必须显式的提供一个对于本地系统中JAR 文件的路径。这么做是为了允许基于本地对象编译，而这些对象是系统类库的一部分。这样的构件应该是一直可用的，Maven 也不会在仓库中去寻找它。如果你将一个依赖范围设置成系统范围，你必须同时提供一个 systemPath 元素。注意该范围是不推荐使用的（你应该一直尽量去从公共或定制的 Maven 仓库中引用依赖）。 |

mvn [options] [<goal(s)>] [<phase(s)>]



<http://blog.csdn.net/yhj19920417/article/details/72627227>

## maven build… 构建不是一个命令，是抽象的，具体体现在如mvn compile,mvn package等

## mvn help:effective-pom 查看有效的pom.xml

## Maven内置变量说明：

${basedir} 项目根目录

${project.build.directory} 构建目录，缺省为target

${project.build.outputDirectory} 构建过程输出目录，缺省为target/classes

${project.build.finalName} 产出物名称，缺省为${project.artifactId}-${project.version}

${project.packaging} 打包类型，缺省为jar

${project.xxx} 当前pom文件的任意节点的内容

Maven filter

filter 规则

maven 通过过滤器来修改部署时的不同配置。可参考<http://blog.csdn.net/fengchao2016/article/details/72726101>

profile，resources，filter

## [是goal还是phase？Maven插件(plugin)goal的执行与生命周期(lifecycle)phase的关系](http://blog.csdn.net/bluishglc/article/details/6632280)

<http://blog.csdn.net/bluishglc/article/details/6632280>

这其实并不是一个复杂的问题，但是似乎没有什么资料对此做过清晰的说明，本文将对这个问题做一个详细的解释。

**背景知识**

maven对构建(build)的过程进行了抽象和定义，这个过程被称为构建的生命周期(lifecycle)。生命周期(lifecycle)由多个阶段(phase)组成,每个阶段(phase)会挂接一到多个goal。goal是maven里定义任务的最小单元，相当于ant里的target。

**以phase为目标构建**

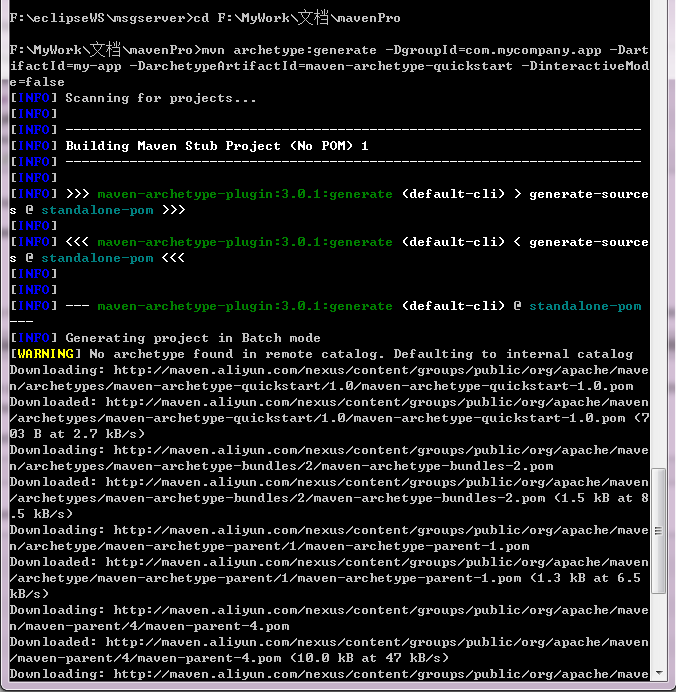
以phase为目标进行构建是最常见的，如我们平时经常执行的mvn compile,mvn test,mvn package...等等,compile,test,package都是maven生命周期(lifecycle)里的phase,通过mvn命令，你可以指定一次构建执行到那一个阶段，在执行过程中，所有经历的执行阶段(phase)上绑定的goal都将得到执行。例如，对于一个jar包应用，当执行mvn package命令时，maven从validate阶段一个阶段一个阶段的执行，在执行到compile阶段时，compiler插件的compile goal会被执行，因为这个goal是绑定在compile阶段(phase)上的。

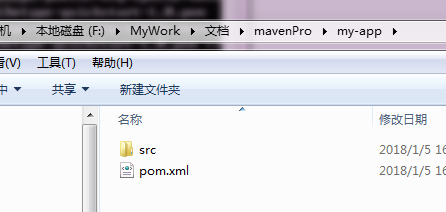
**以goal为目标构建**

虽然以phase为目标的构建最常见，但是有时候我们会发现，一些插件的goal并不适合绑定到任何阶段(phase)上，或者是，这些goal往往是单独执行，不需要同某个阶段(phase)绑定在一起，比如hibernate插件的导入\导出goal多数情况下是根据需要要手动执行的(当然，也可以绑定到某个阶段上，比如进行单元测试时，可考虑将其绑定到test阶段上)。再比如jetty(6.1.26)插件，它的goal都是将打包或未打包的工程部署到jetty里然后启动jetty容器的，多数情况下，人们都是独立运行这些goal的，比如：人们希望当键入mvn jetty:run后，工程就能完成编译后启动jetty,

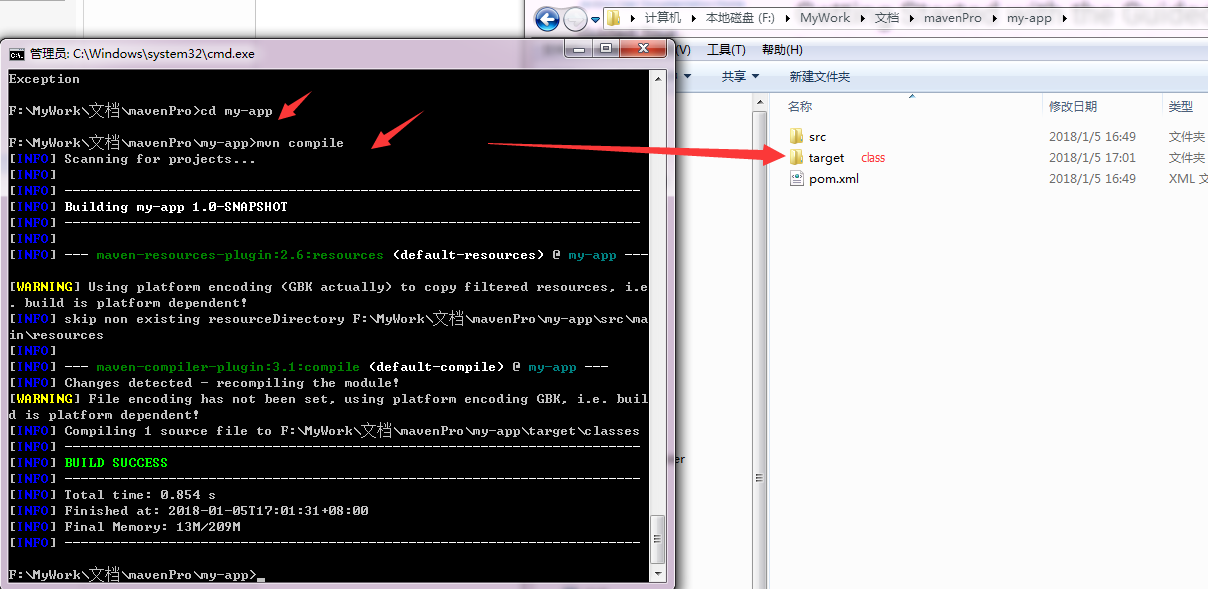
## 新建maven项目

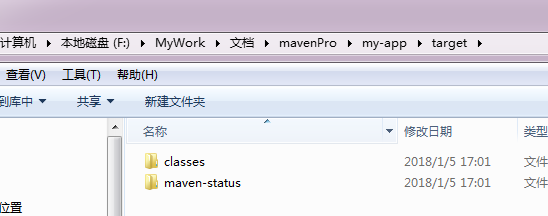
<http://maven.apache.org/guides/getting-started/index.html> 官方参考





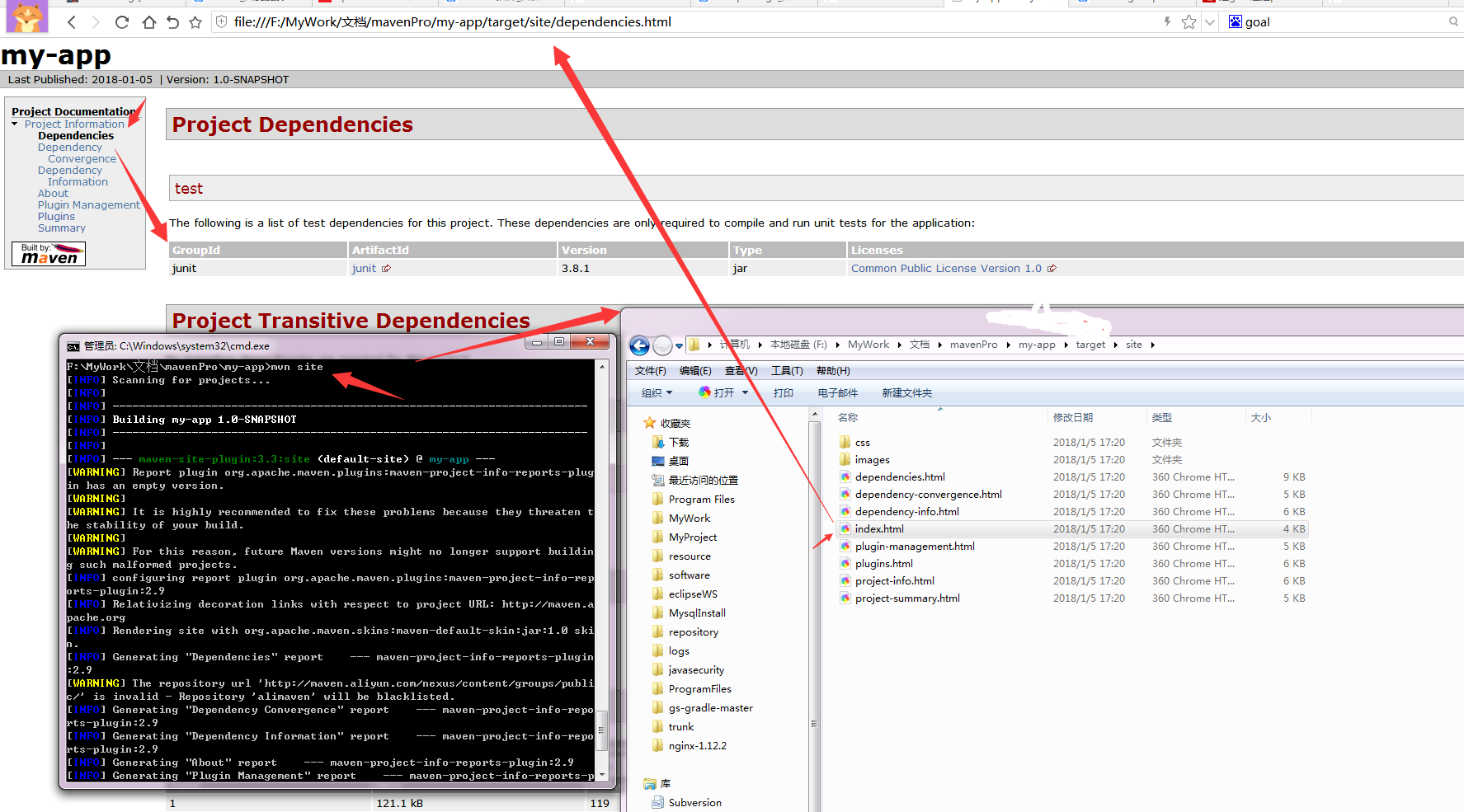
Mvn compile 编译





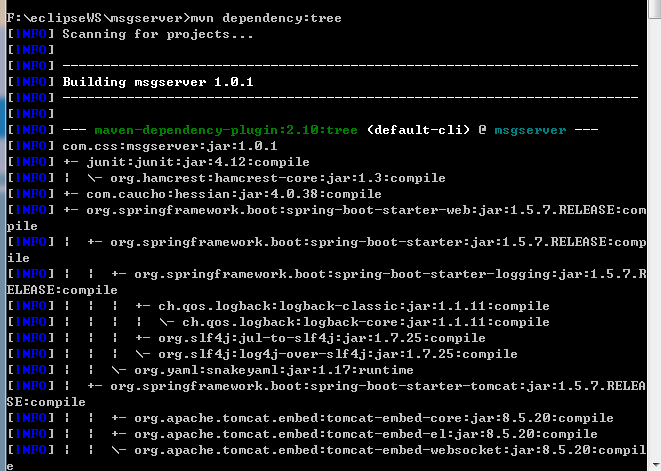
Mvn clean： 删除target

Mvn site：为项目生成站点文档

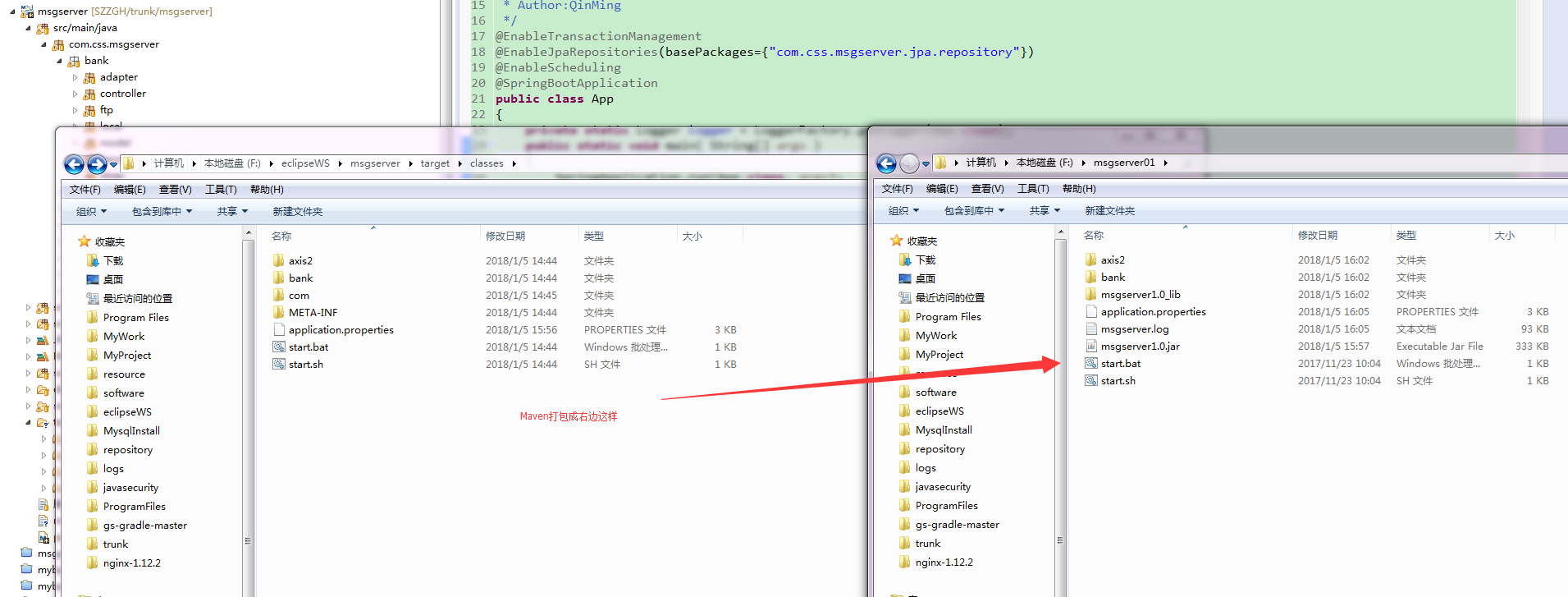


mvn denpendcy:tree

打印出msgserver系统整个项目的依赖树



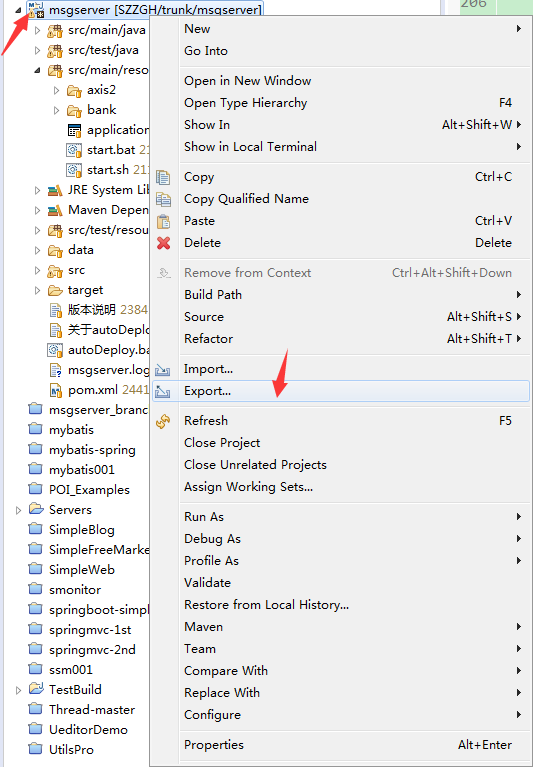
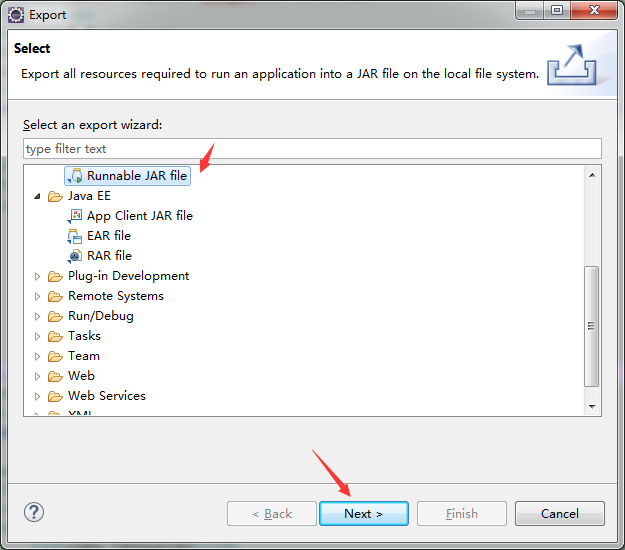
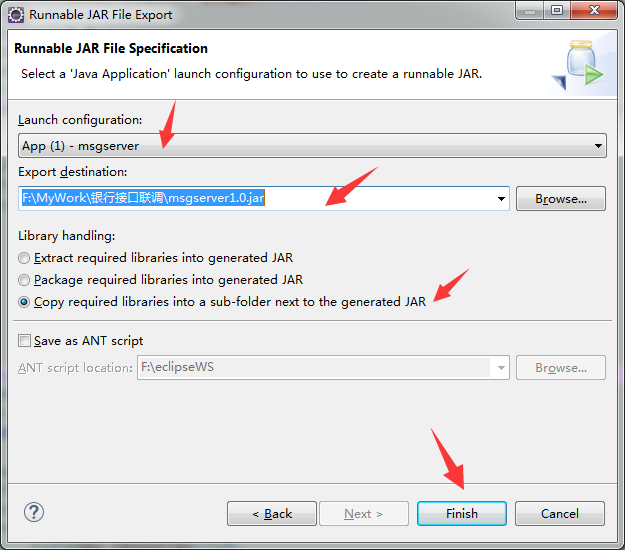
|  |
| --- |
| [INFO] com.css:msgserver:jar:1.0.1  [INFO] +- junit:junit:jar:4.12:compile  [INFO] | \- org.hamcrest:hamcrest-core:jar:1.3:compile  [INFO] +- com.caucho:hessian:jar:4.0.38:compile  [INFO] +- org.springframework.boot:spring-boot-starter-web:jar:1.5.7.RELEASE:com  pile  [INFO] | +- org.springframework.boot:spring-boot-starter:jar:1.5.7.RELEASE:comp  ile  [INFO] | | +- org.springframework.boot:spring-boot-starter-logging:jar:1.5.7.R  ELEASE:compile  [INFO] | | | +- ch.qos.logback:logback-classic:jar:1.1.11:compile  [INFO] | | | | \- ch.qos.logback:logback-core:jar:1.1.11:compile  [INFO] | | | +- org.slf4j:jul-to-slf4j:jar:1.7.25:compile  [INFO] | | | \- org.slf4j:log4j-over-slf4j:jar:1.7.25:compile  [INFO] | | \- org.yaml:snakeyaml:jar:1.17:runtime  [INFO] | +- org.springframework.boot:spring-boot-starter-tomcat:jar:1.5.7.RELEA  SE:compile  [INFO] | | +- org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.20:compile  [INFO] | | +- org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.20:compile  [INFO] | | \- org.apache.tomcat.embed:tomcat-embed-websocket:jar:8.5.20:compil  e  [INFO] | +- org.hibernate:hibernate-validator:jar:5.3.5.Final:compile  [INFO] | | +- javax.validation:validation-api:jar:1.1.0.Final:compile  [INFO] | | \- com.fasterxml:classmate:jar:1.3.4:compile  [INFO] | +- com.fasterxml.jackson.core:jackson-databind:jar:2.8.10:compile  [INFO] | | +- com.fasterxml.jackson.core:jackson-annotations:jar:2.8.0:compile  [INFO] | | \- com.fasterxml.jackson.core:jackson-core:jar:2.8.10:compile  [INFO] | +- org.springframework:spring-web:jar:4.3.11.RELEASE:compile  [INFO] | \- org.springframework:spring-webmvc:jar:4.3.11.RELEASE:compile  [INFO] | \- org.springframework:spring-expression:jar:4.3.11.RELEASE:compile  [INFO] +- org.springframework.boot:spring-boot-starter-aop:jar:1.5.7.RELEASE:com  pile  [INFO] | \- org.springframework:spring-aop:jar:4.3.11.RELEASE:compile  [INFO] +- org.apache.camel:camel-spring-boot-starter:jar:2.18.0:compile  [INFO] | +- org.apache.camel:camel-spring-boot:jar:2.18.0:compile  [INFO] | | \- org.apache.camel:camel-spring:jar:2.18.0:compile  [INFO] | \- org.apache.camel:camel-core-starter:jar:2.18.0:compile  [INFO] | \- com.github.ben-manes.caffeine:caffeine:jar:2.3.5:compile  [INFO] +- org.apache.camel:camel-ftp:jar:2.18.0:compile  [INFO] | +- org.apache.camel:camel-core:jar:2.18.0:compile  [INFO] | | \- org.slf4j:slf4j-api:jar:1.7.25:compile  [INFO] | +- com.jcraft:jsch:jar:0.1.54:compile  [INFO] | +- commons-net:commons-net:jar:3.3:compile  [INFO] | +- com.sun.xml.bind:jaxb-core:jar:2.2.11:compile  [INFO] | \- com.sun.xml.bind:jaxb-impl:jar:2.2.11:compile  [INFO] +- org.apache.axis2:axis2:pom:1.7.6:compile  [INFO] +- org.apache.axis2:axis2-transport-local:jar:1.7.6:compile  [INFO] +- org.apache.axis2:axis2-transport-http:jar:1.7.6:compile  [INFO] | +- org.apache.httpcomponents:httpclient:jar:4.5.3:compile  [INFO] | | +- org.apache.httpcomponents:httpcore:jar:4.4.6:compile  [INFO] | | \- commons-codec:commons-codec:jar:1.10:compile  [INFO] | \- commons-httpclient:commons-httpclient:jar:3.1:compile  [INFO] +- org.apache.axis2:axis2-kernel:jar:1.7.6:compile  [INFO] | +- org.apache.ws.commons.axiom:axiom-api:jar:1.2.20:compile  [INFO] | | +- jaxen:jaxen:jar:1.1.6:compile  [INFO] | | +- org.apache.geronimo.specs:geronimo-stax-api\_1.0\_spec:jar:1.0.1:c  ompile  [INFO] | | \- org.apache.james:apache-mime4j-core:jar:0.7.2:compile  [INFO] | +- org.apache.ws.commons.axiom:axiom-impl:jar:1.2.20:runtime  [INFO] | | \- org.codehaus.woodstox:woodstox-core-asl:jar:4.2.0:runtime  [INFO] | | \- org.codehaus.woodstox:stax2-api:jar:3.1.1:runtime  [INFO] | +- org.apache.geronimo.specs:geronimo-ws-metadata\_2.0\_spec:jar:1.1.2:c  ompile  [INFO] | +- org.apache.geronimo.specs:geronimo-jta\_1.1\_spec:jar:1.1:compile  [INFO] | +- javax.servlet:servlet-api:jar:2.3:compile  [INFO] | +- commons-fileupload:commons-fileupload:jar:1.3.3:compile  [INFO] | +- wsdl4j:wsdl4j:jar:1.6.3:compile  [INFO] | +- org.apache.ws.xmlschema:xmlschema-core:jar:2.2.1:compile  [INFO] | +- org.apache.neethi:neethi:jar:3.0.3:compile  [INFO] | +- org.apache.woden:woden-core:jar:1.0M10:compile  [INFO] | +- commons-logging:commons-logging:jar:1.1.1:compile  [INFO] | +- javax.ws.rs:jsr311-api:jar:1.1.1:compile  [INFO] | \- commons-io:commons-io:jar:2.1:compile  [INFO] +- org.apache.axis2:axis2-jaxws:jar:1.7.6:compile  [INFO] | +- org.apache.geronimo.specs:geronimo-annotation\_1.0\_spec:jar:1.1:comp  ile  [INFO] | +- org.apache.geronimo.specs:geronimo-jaxws\_2.2\_spec:jar:1.0:compile  [INFO] | +- org.apache.axis2:axis2-saaj:jar:1.7.6:compile  [INFO] | | \- org.apache.geronimo.specs:geronimo-saaj\_1.3\_spec:jar:1.0.1:compi  le  [INFO] | +- org.apache.axis2:axis2-metadata:jar:1.7.6:compile  [INFO] | | \- com.sun.xml.ws:jaxws-tools:jar:2.1.3:compile  [INFO] | +- org.apache.geronimo.specs:geronimo-javamail\_1.4\_spec:jar:1.6:compil  e  [INFO] | +- xml-resolver:xml-resolver:jar:1.2:compile  [INFO] | +- com.sun.xml.bind:jaxb-xjc:jar:2.2.6:compile  [INFO] | +- xalan:xalan:jar:2.7.0:compile  [INFO] | \- javax.xml.bind:jaxb-api:jar:2.2.6:compile  [INFO] +- org.apache.axis2:axis2-adb:jar:1.7.6:compile  [INFO] | +- org.apache.ws.commons.axiom:axiom-dom:jar:1.2.20:runtime  [INFO] | \- org.apache.geronimo.specs:geronimo-activation\_1.1\_spec:jar:1.0.2:co  mpile  [INFO] +- org.apache.axis2:axis2-spring:jar:1.7.6:compile  [INFO] | +- org.springframework:spring-core:jar:4.3.11.RELEASE:compile  [INFO] | +- org.springframework:spring-beans:jar:4.3.11.RELEASE:compile  [INFO] | \- org.springframework:spring-context:jar:4.3.11.RELEASE:compile  [INFO] +- javax.servlet:jstl:jar:1.2:compile  [INFO] +- org.springframework.boot:spring-boot-test:jar:1.5.7.RELEASE:compile  [INFO] | \- org.springframework.boot:spring-boot:jar:1.5.7.RELEASE:compile  [INFO] +- org.springframework.boot:spring-boot-starter-test:jar:1.5.7.RELEASE:te  st  [INFO] | +- com.jayway.jsonpath:json-path:jar:2.2.0:test  [INFO] | | \- net.minidev:json-smart:jar:2.2.1:test  [INFO] | | \- net.minidev:accessors-smart:jar:1.1:test  [INFO] | | \- org.ow2.asm:asm:jar:5.0.3:test  [INFO] | +- org.assertj:assertj-core:jar:2.6.0:test  [INFO] | +- org.mockito:mockito-core:jar:1.10.19:test  [INFO] | | \- org.objenesis:objenesis:jar:2.1:test  [INFO] | +- org.hamcrest:hamcrest-library:jar:1.3:test  [INFO] | +- org.skyscreamer:jsonassert:jar:1.4.0:test  [INFO] | | \- com.vaadin.external.google:android-json:jar:0.0.20131108.vaadin1  :test  [INFO] | \- org.springframework:spring-test:jar:4.3.11.RELEASE:test  [INFO] +- org.springframework.boot:spring-boot-test-autoconfigure:jar:1.5.7.RELE  ASE:compile  [INFO] | \- org.springframework.boot:spring-boot-autoconfigure:jar:1.5.7.RELEAS  E:compile  [INFO] +- org.apache.rampart:rampart-core:jar:1.7.1:compile  [INFO] | +- org.apache.axis2:axis2-mtompolicy:jar:1.7.6:compile  [INFO] | +- org.apache.axis2:mex:jar:impl:1.7.6:compile  [INFO] | +- org.apache.ws.security:wss4j:jar:1.6.16:compile  [INFO] | | \- org.apache.santuario:xmlsec:jar:1.5.7:compile  [INFO] | +- org.bouncycastle:bcprov-jdk15on:jar:1.49:compile  [INFO] | +- org.opensaml:opensaml:jar:2.5.1-1:compile  [INFO] | | +- org.opensaml:openws:jar:1.4.2-1:compile  [INFO] | | | \- org.opensaml:xmltooling:jar:1.3.2-1:compile  [INFO] | | | +- joda-time:joda-time:jar:2.9.9:compile  [INFO] | | | +- org.bouncycastle:bcprov-jdk15:jar:1.45:compile  [INFO] | | | +- ca.juliusdavies:not-yet-commons-ssl:jar:0.3.9:compile  [INFO] | | | \- net.jcip:jcip-annotations:jar:1.0:compile  [INFO] | | +- velocity:velocity:jar:1.5:compile  [INFO] | | +- org.owasp.esapi:esapi:jar:2.0GA:compile  [INFO] | | +- xml-apis:xml-apis:jar:1.4.01:runtime  [INFO] | | \- xerces:xercesImpl:jar:2.10.0:runtime  [INFO] | +- org.apache.rampart:rampart-policy:jar:1.7.1:compile  [INFO] | \- org.apache.rampart:rampart-trust:jar:1.7.1:compile  [INFO] +- org.apache.rampart:rampart:jar:1.7.1:compile  [INFO] +- org.freemarker:freemarker:jar:2.3.26-incubating:compile  [INFO] +- commons-beanutils:commons-beanutils:jar:1.9.3:compile  [INFO] | \- commons-collections:commons-collections:jar:3.2.2:compile  [INFO] +- org.apache.commons:commons-digester3:jar:3.2:compile  [INFO] | \- cglib:cglib:jar:2.2.2:compile  [INFO] | \- asm:asm:jar:3.3.1:compile  [INFO] +- com.thoughtworks.xstream:xstream:jar:1.4.7:compile  [INFO] | +- xmlpull:xmlpull:jar:1.1.3.1:compile  [INFO] | \- xpp3:xpp3\_min:jar:1.1.4c:compile  [INFO] +- org.springframework.boot:spring-boot-starter-data-jpa:jar:1.5.7.RELEAS  E:compile  [INFO] | +- org.springframework.boot:spring-boot-starter-jdbc:jar:1.5.7.RELEASE  :compile  [INFO] | | +- org.apache.tomcat:tomcat-jdbc:jar:8.5.20:compile  [INFO] | | | \- org.apache.tomcat:tomcat-juli:jar:8.5.20:compile  [INFO] | | \- org.springframework:spring-jdbc:jar:4.3.11.RELEASE:compile  [INFO] | +- org.hibernate:hibernate-core:jar:5.0.12.Final:compile  [INFO] | | +- antlr:antlr:jar:2.7.7:compile  [INFO] | | \- org.jboss:jandex:jar:2.0.0.Final:compile  [INFO] | +- javax.transaction:javax.transaction-api:jar:1.2:compile  [INFO] | +- org.springframework.data:spring-data-jpa:jar:1.11.7.RELEASE:compile  [INFO] | | +- org.springframework.data:spring-data-commons:jar:1.13.7.RELEASE:  compile  [INFO] | | +- org.springframework:spring-orm:jar:4.3.11.RELEASE:compile  [INFO] | | +- org.springframework:spring-tx:jar:4.3.11.RELEASE:compile  [INFO] | | \- org.slf4j:jcl-over-slf4j:jar:1.7.25:compile  [INFO] | \- org.springframework:spring-aspects:jar:4.3.11.RELEASE:compile  [INFO] +- org.hibernate:hibernate-entitymanager:jar:5.0.12.Final:compile  [INFO] | +- dom4j:dom4j:jar:1.6.1:compile  [INFO] | +- org.hibernate.common:hibernate-commons-annotations:jar:5.0.1.Final:  compile  [INFO] | +- org.hibernate.javax.persistence:hibernate-jpa-2.1-api:jar:1.0.0.Fin  al:compile  [INFO] | \- org.javassist:javassist:jar:3.21.0-GA:compile  [INFO] +- org.aspectj:aspectjweaver:jar:1.8.10:compile  [INFO] +- mysql:mysql-connector-java:jar:5.1.44:compile  [INFO] +- com.alibaba:druid:jar:1.1.3:compile  [INFO] | +- com.alibaba:jconsole:jar:1.8.0:system  [INFO] | \- com.alibaba:tools:jar:1.8.0:system  [INFO] +- net.sf.json-lib:json-lib:jar:jdk15:2.2.3:compile  [INFO] | +- commons-lang:commons-lang:jar:2.4:compile  [INFO] | \- net.sf.ezmorph:ezmorph:jar:1.0.6:compile  [INFO] \- org.jboss.logging:jboss-logging:jar:3.3.1.Final:compile  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 03:42 min  [INFO] Finished at: 2018-01-05T16:38:51+08:00  [INFO] Final Memory: 31M/197M  [INFO] ------------------------------------------------------------------------ |

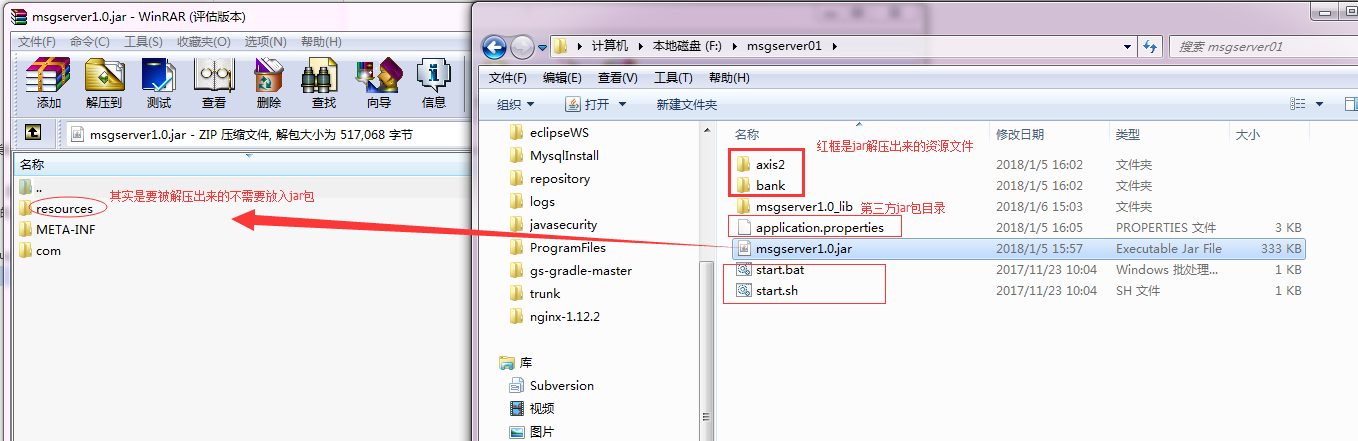


## Maven构建msgserver，实现自动化打包部署

### 现状：

msgserver是工会与中国银行之间的接口中间系统，以往在修改了代码之后，需要在IDE如Eclipse中使用Export命令打包成jar，再将jar包中的资源文件和第三方jar解压出来，并按照如下目录结构部署：



（之前现状图）

以上的打包和部署方式有几点不足：

* 需要依赖IDE（Eclipse）
* 每次修改代码后需要手动Export，解压，步骤繁琐麻烦。
* 项目本身采用了Maven，但仅仅用来下载依赖了，并没有发挥其自动化构建的优势。

针对以上几点，我对msgserver进行了自动化构建优化（其实是在学习Jenkins持续集成的构建时不太清楚，顺便把Ant，Maven，Grandle都学习了一下，现在应该说是学以致用吧），

### **优化步骤**：

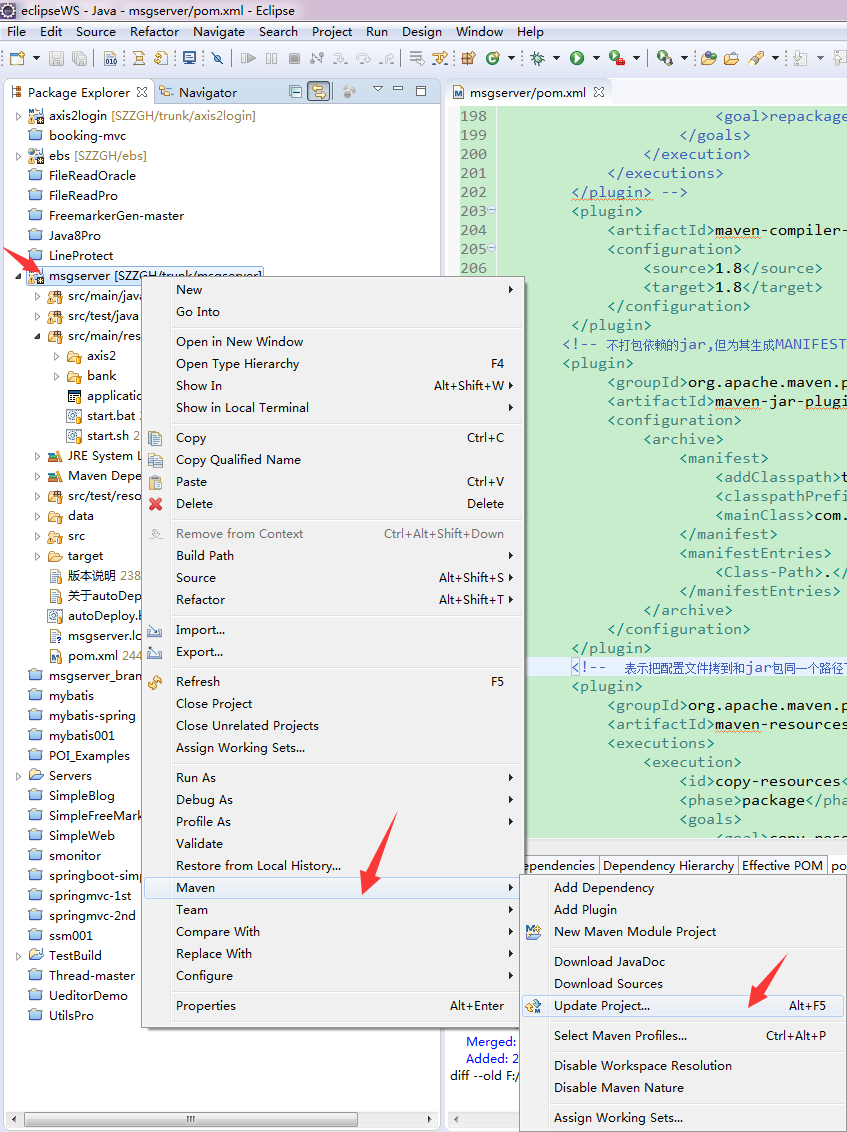
1.通过Maven官网学习了pom.xml文件的结构及其配置，Maven插件，Maven生命周期等基础知识。

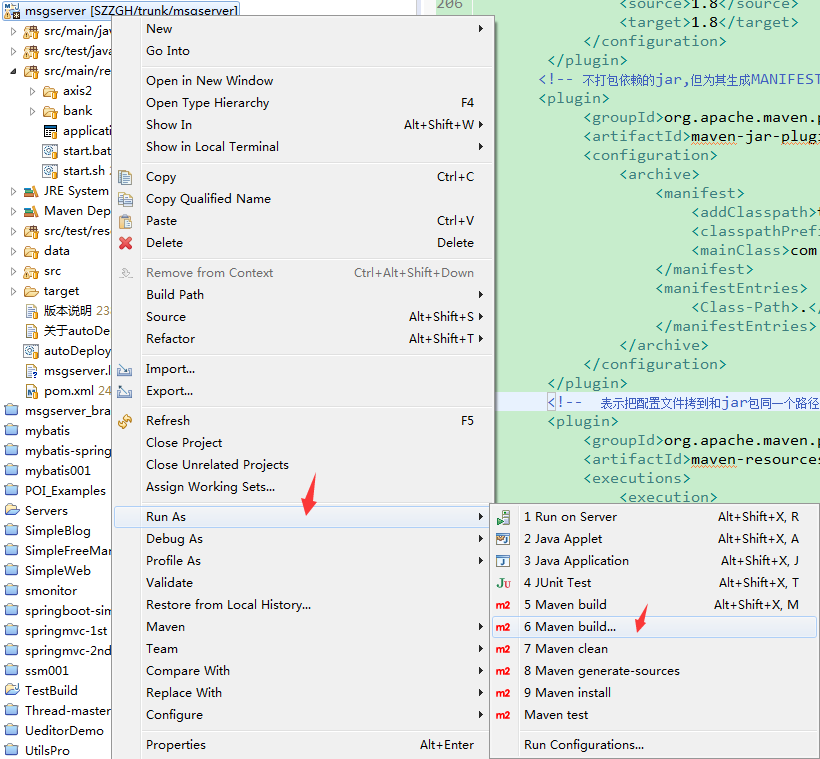
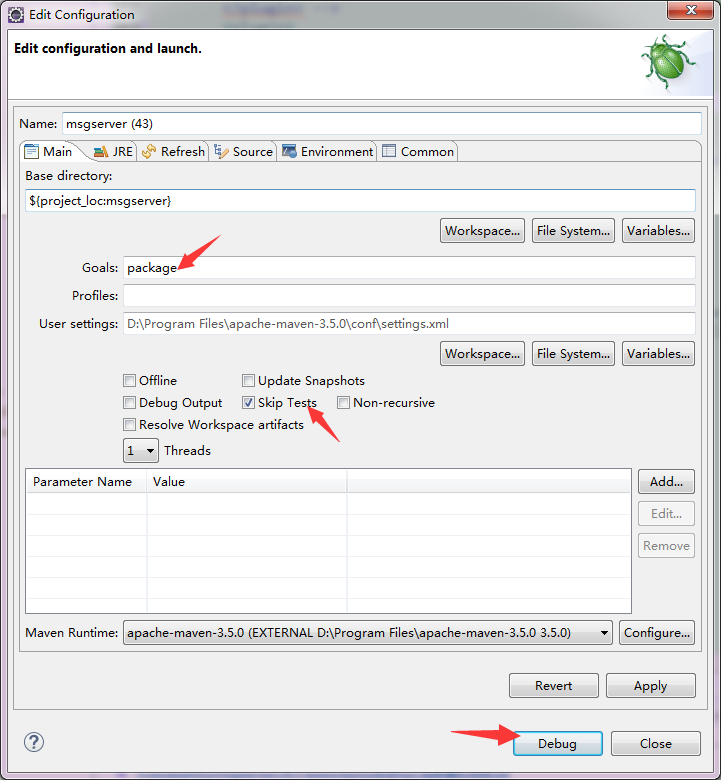
2.着手修改msgserver的pom.xml文件，加入自动化构建相关配置，其实主要是加入Maven插件，如maven-compiler-plugin，maven-jar-plugin，maven-resources-plugin，maven-dependency-plugin，也包括处理资源文件< resources>.

在加入这些插件之前我并不知道需要加入哪些插件，是通过观察之前部署的目录结构总结以下几点：资源文件不需要打包进jar，第三方jar存放在单独目录，第三方jar信息需要维护进MANIFEST.MF（这里配置时需要注意蛮多细节）。

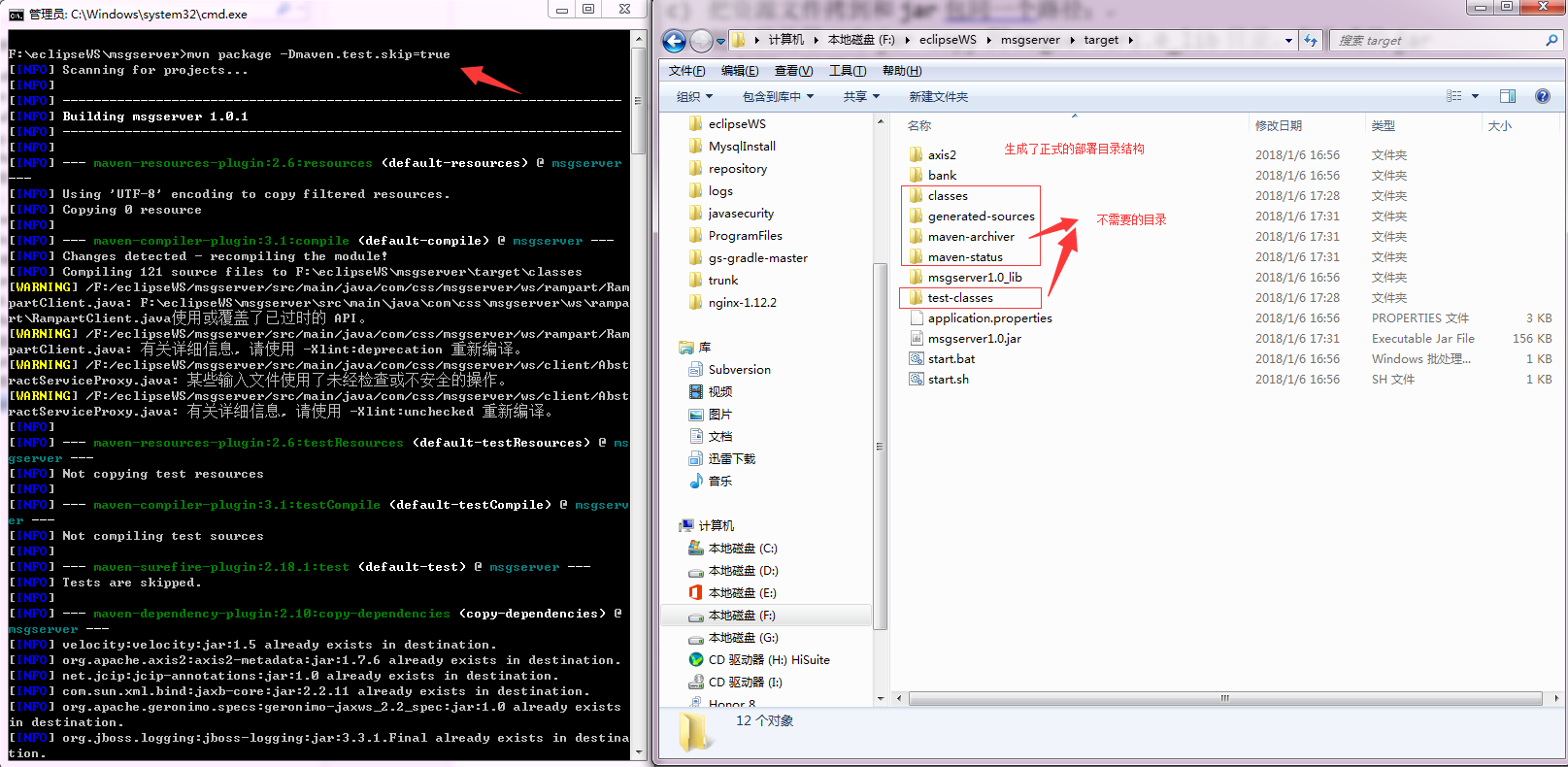
1. 以mvn package -Dmaven.test.skip=true 命令反复测试，逐步加入了如下功能：
   1. 资源文件不打进jar包；
   2. 不打包依赖的第三方jar,但为其生成MANIFEST.MF文件；
   3. 把资源文件拷到和jar包同一个路径；
   4. 把依赖的第三方jar copy到msgserver1.0\_lib目录，和生成的jar放在同一级目录下。
2. 最终通过mvn package -Dmaven.test.skip=true可以生成项目部署的结构了

Eclipse中：

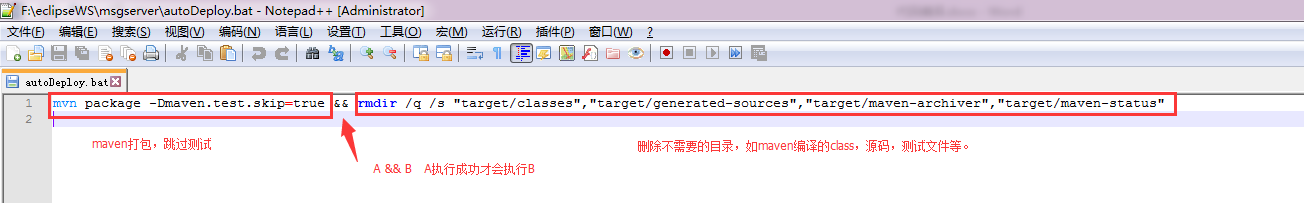


或者Cmd直接Maven命令：

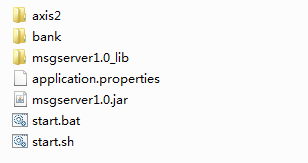


1. 发现还存在一些不需要的文件，于是写一个windows批处理程序，最终只要“在Windows机器上安装Maven(http://maven.apache.org/)并配置相关环境变量，svn拉取源码后，然后执行autoDeploy.bat批处理命令，执行完成后会在当前目录下生产target文件夹，部署时只需要将其中的内容拷贝到服务器上。”。批处理程序autoDeploy.bat内容如下：

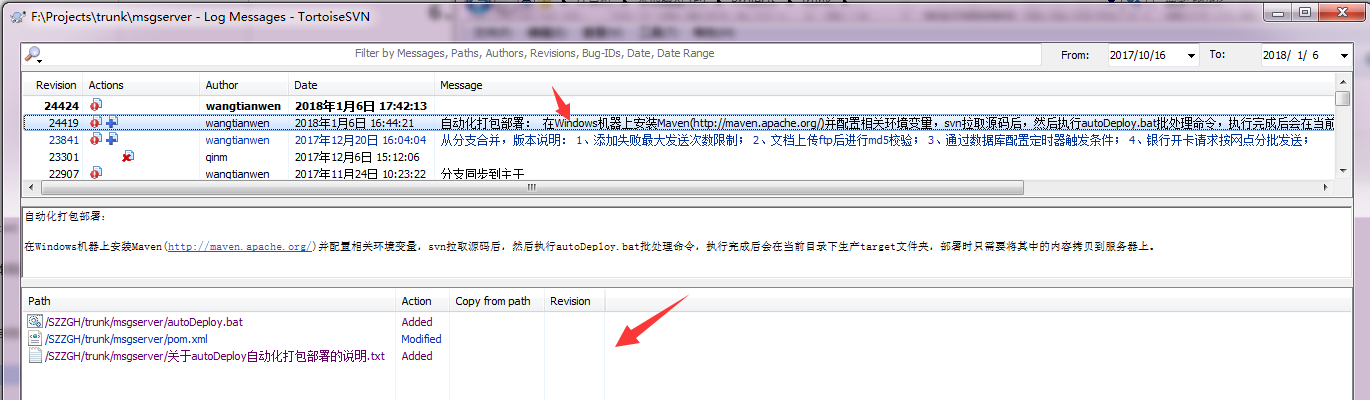


|  |
| --- |
| mvn package -Dmaven.test.skip=true && rmdir /q /s "target/classes","target/generated-sources","target/maven-archiver","target/maven-status","target/test-classes" |

### 得到最后干净的目录结构：



附源代码：及svn提交说明：



pom.xml

|  |
| --- |
| <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.css</groupId>  <artifactId>msgserver</artifactId>  <version>1.0.1</version>  <packaging>jar</packaging>  <name>msgserver</name>  <url>http://maven.apache.org</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <camel-version>2.18.0</camel-version>  <axis-version>1.7.6</axis-version>  </properties>  <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <!-- <version>2.0.0.M4</version> -->  <version>1.5.7.RELEASE</version>  </parent>  <dependencies>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  </dependency>  <dependency>  <groupId>com.caucho</groupId>  <artifactId>hessian</artifactId>  <version>4.0.38</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-aop</artifactId>  </dependency>  <dependency>  <groupId>org.apache.camel</groupId>  <artifactId>camel-spring-boot-starter</artifactId>  <version>${camel-version}</version>  </dependency>  <dependency>  <groupId>org.apache.camel</groupId>  <artifactId>camel-ftp</artifactId>  <version>${camel-version}</version>  </dependency>  <!-- axis2 -->  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2</artifactId>  <version>${axis-version}</version>  <type>pom</type>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-transport-local</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-transport-http</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-kernel</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-jaxws</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-adb</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>org.apache.axis2</groupId>  <artifactId>axis2-spring</artifactId>  <version>${axis-version}</version>  </dependency>  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>jstl</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-test</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-test-autoconfigure</artifactId>  </dependency>  <!-- https://mvnrepository.com/artifact/org.apache.rampart/rampart-core -->  <dependency>  <groupId>org.apache.rampart</groupId>  <artifactId>rampart-core</artifactId>  <version>1.7.1</version>  </dependency>  <!-- https://mvnrepository.com/artifact/org.apache.rampart/rampart -->  <dependency>  <groupId>org.apache.rampart</groupId>  <artifactId>rampart</artifactId>  <version>1.7.1</version>  </dependency>  <dependency>  <groupId>org.freemarker</groupId>  <artifactId>freemarker</artifactId>  </dependency>  <dependency>  <groupId>commons-beanutils</groupId>  <artifactId>commons-beanutils</artifactId>  </dependency>  <dependency>  <groupId>org.apache.commons</groupId>  <artifactId>commons-digester3</artifactId>  <version>3.2</version>  </dependency>  <dependency>  <groupId>com.thoughtworks.xstream</groupId>  <artifactId>xstream</artifactId>  <version>1.4.7</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-data-jpa</artifactId>  </dependency>  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-entitymanager</artifactId>  </dependency>  <dependency>  <groupId>org.aspectj</groupId>  <artifactId>aspectjweaver</artifactId>  </dependency>  <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  </dependency>  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>druid</artifactId>  <version>1.1.3</version>  </dependency>  <dependency>  <groupId>net.sf.json-lib</groupId>  <artifactId>json-lib</artifactId>  <version>2.2.3</version>  <classifier>jdk15</classifier>  </dependency>  <dependency>  <groupId>org.jboss.logging</groupId>  <artifactId>jboss-logging</artifactId>  <!-- <version>3.3.1.Final</version> -->  </dependency>  <!-- 下载之用 -->  <!-- https://mvnrepository.com/artifact/com.alibaba/druid -->  <!--dependency> <groupId>com.alibaba</groupId> <artifactId>druid</artifactId>  <version>1.0.31</version> </dependency -->  <!-- https://mvnrepository.com/artifact/log4j/log4j -->  <!--dependency> <groupId>log4j</groupId> <artifactId>log4j</artifactId>  <version>1.2.17</version> </dependency -->  <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->  </dependencies>  <build>  <finalName>msgserver1.0</finalName>  <!-- 资源文件不打进jar包 -->  <resources>  <resource>  <directory>src/main/resources</directory>  <excludes>  <exclude>\*\*/\*</exclude>  </excludes>  <filtering>true</filtering>  </resource>  </resources>  <plugins>  <!-- <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-maven-plugin</artifactId>  <executions>  <execution>  <goals>  <goal>repackage</goal>  </goals>  </execution>  </executions>  </plugin> -->  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  <!-- 不打包依赖的jar,但为其生成MANIFEST.MF文件。特别注意这里的mainClass和manifestEntries配置-->  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-jar-plugin</artifactId>  <configuration>  <archive>  <manifest>  <addClasspath>true</addClasspath>  <classpathPrefix>msgserver1.0\_lib/</classpathPrefix>  <mainClass>com.css.msgserver.App</mainClass>  </manifest>  <manifestEntries>  <Class-Path>.</Class-Path>  </manifestEntries>  </archive>  </configuration>  </plugin>  <!-- 表示把配置文件拷到和jar包同一个路径下 -->  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-resources-plugin</artifactId>  <executions>  <execution>  <id>copy-resources</id>  <phase>package</phase>  <goals>  <goal>copy-resources</goal>  </goals>  <configuration>  <encoding>UTF-8</encoding>  <outputDirectory>  ${project.build.directory}  </outputDirectory>  <resources>  <resource>  <directory>src/main/resources/</directory>  </resource>  </resources>  </configuration>  </execution>  </executions>  </plugin>  <!-- 把依赖的jar copy到msgserver1.0\_lib目录，和生成的jar放在同一级目录下 -->  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-dependency-plugin</artifactId>  <executions>  <execution>  <id>copy-dependencies</id>  <phase>prepare-package</phase>  <goals>  <goal>copy-dependencies</goal>  </goals>  <configuration>  <outputDirectory>${project.build.directory}/msgserver1.0\_lib</outputDirectory>  <overWriteReleases>false</overWriteReleases>  <overWriteSnapshots>false</overWriteSnapshots>  <overWriteIfNewer>true</overWriteIfNewer>  </configuration>  </execution>  </executions>  </plugin>  </plugins>  </build>  <repositories>  <repository>  <id>spring-milestones</id>  <name>Spring Milestones</name>  <url>https://repo.spring.io/libs-milestone</url>  <snapshots>  <enabled>false</enabled>  </snapshots>  </repository>  </repositories>  </project> |