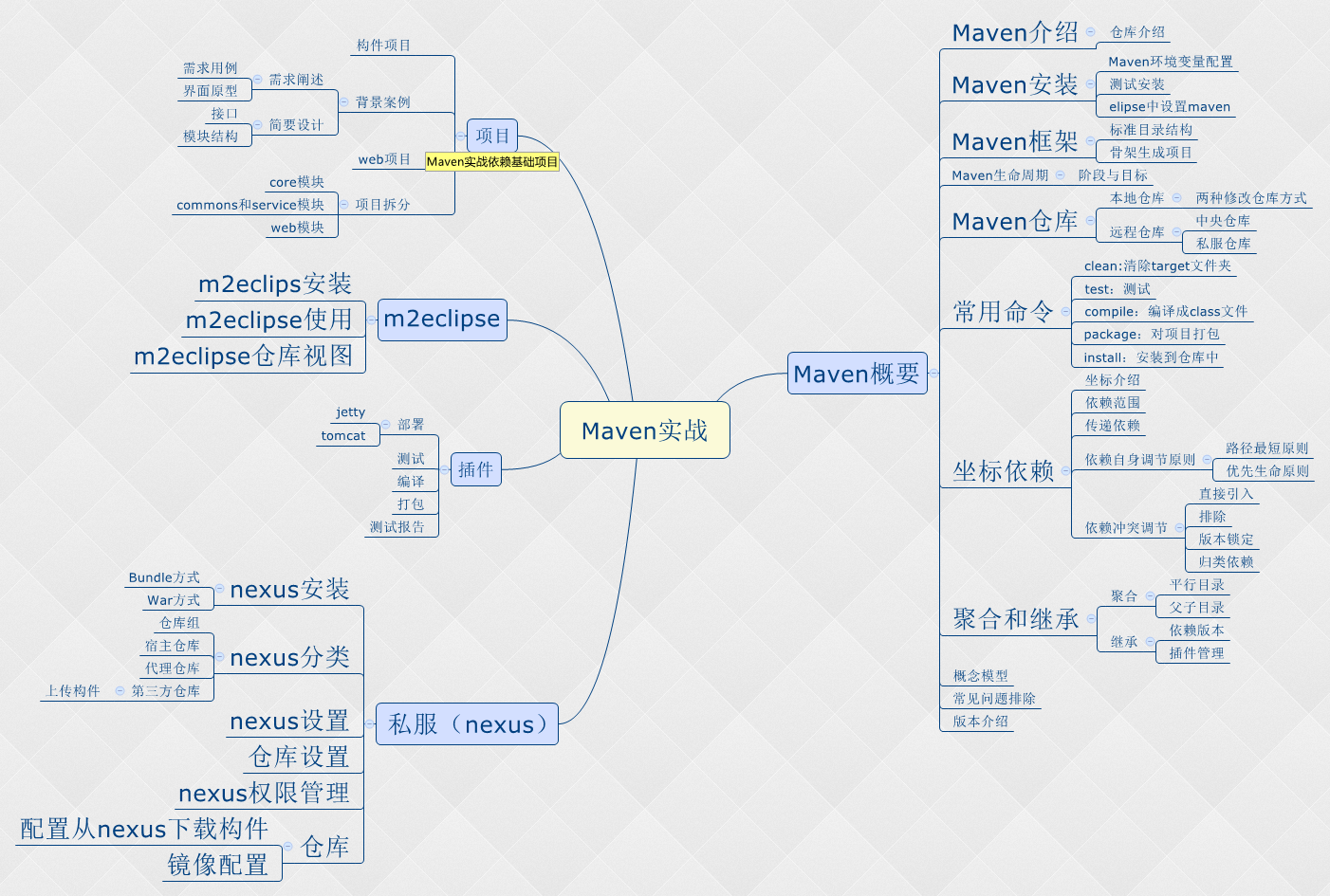
# Maven实践

## 文档结构

### 1.1目录视图

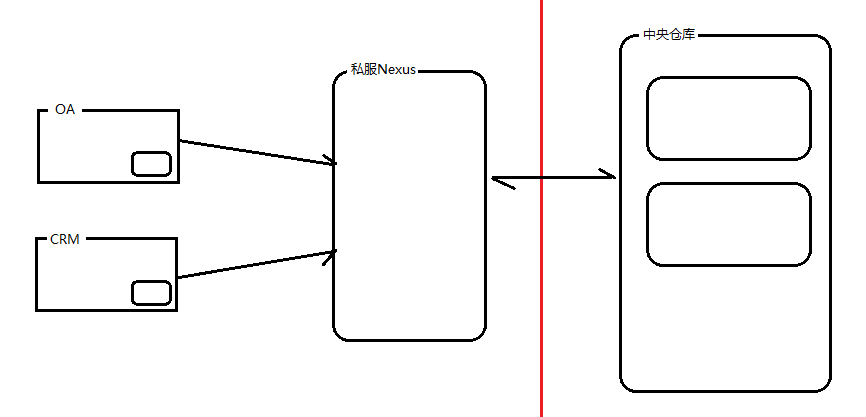


## 第2节 Maven概要

### 2.1Maven介绍

maven是Apache的一个开源项目，可以对项目进行构建，依赖管理。

可以通过命令对项目进行编译，测试，打包，安装，部署，测试报告以及文档生成等。

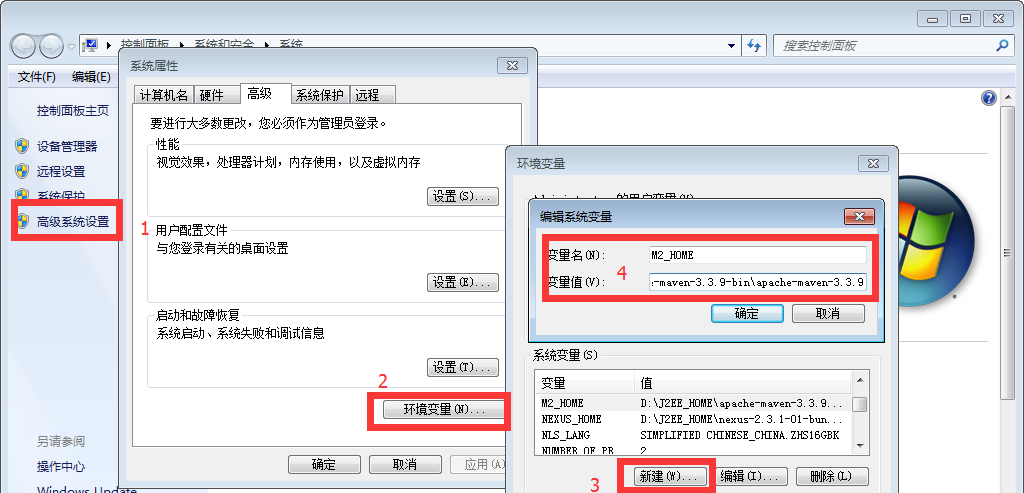


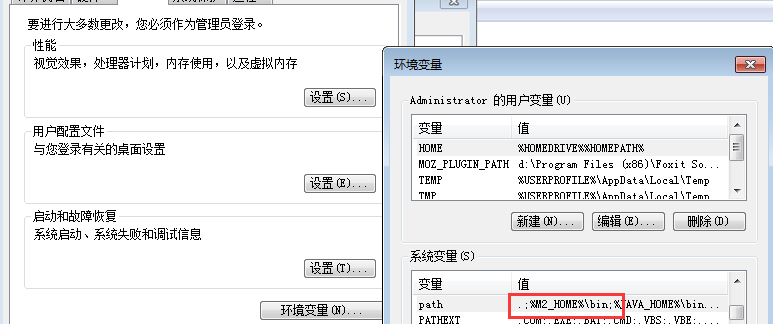
GAV(groupId, artifactId,version) 相当于X,Y,Z坐标。

### 2.2Maven安装

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

1. 确保JAVA\_HOME环境变量指定JDK的安装目录，也就是保证JDK能够正确运行。
2. 解压apache-maven-3.3.9-bin.zip；
3. 配置环境变量，将bin目录配置到path环境变量。







### 2.3Maven框架

1）Maven安装文件介绍：

bin文件夹 命令脚本文件 包括windows和linuxs下面的脚本；

boot文件夹 他是一个类加载框架，maven使用还框架来加载自己的类库；

conf文件夹 主要包含一个主要的文件settings.xml文件；

lib文件夹 包括了maven运行所需要的所有java类库，自身及第三方类库；

2）maven标准项目结构

使用命令新建maven项目

mvn -B archetype:generate -DarchetypeGroupId=org.apache.maven.archetypes -DgroupId=com.mycompany.app -DartifactId=my-app

|  |
| --- |
| [INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.0  [INFO] ----------------------------------------------------------------------------  [INFO] Parameter: basedir, Value: G:\eclipse\workspace\tmp  [INFO] Parameter: package, Value: com.mycompany.app  [INFO] Parameter: groupId, Value: com.mycompany.app  [INFO] Parameter: artifactId, Value: my-app  [INFO] Parameter: packageName, Value: com.mycompany.app  [INFO] Parameter: version, Value: 1.0-SNAPSHOT  [INFO] project created from Old (1.x) Archetype in dir: G:\eclipse\workspace\tmp\my-app  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 26.547 s  [INFO] Finished at: 2018-12-16T16:22:32+08:00  [INFO] Final Memory: 15M/170M  [INFO] ------------------------------------------------------------------------ |

${basedir}/my-app

└───pom.xml

│

└───src

├───main

│ └───java

│ └───com

│ └───mycompany

│ └───app

│ App.java

└───test

└───java

└───com

└───mycompany

└───app

AppTest.java

3）

### 2.4Maven生命周期

<http://maven.apache.org/ref/3.6.0/maven-core/lifecycles.html>

Maven defines 3 lifecycles in META-INF/plexus/components.xml:

default Lifecycle

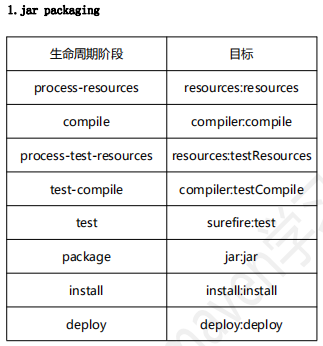
clean Lifecycle

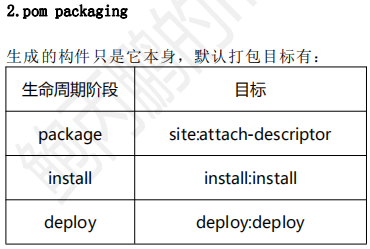
site Lifecycle

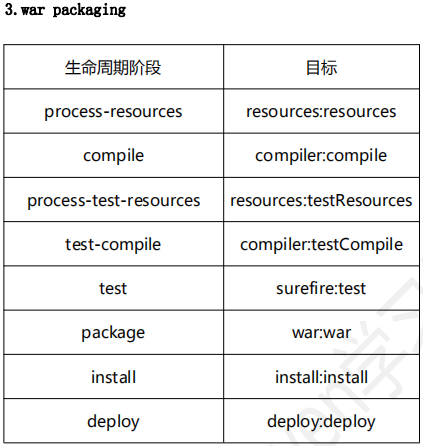
|  |
| --- |
| 1. <phases> 2. <phase>validate</phase> 3. <phase>initialize</phase> 4. <phase>generate-sources</phase> 5. <phase>process-sources</phase> 6. <phase>generate-resources</phase> 7. <phase>process-resources</phase> 8. <phase>compile</phase> 9. <phase>process-classes</phase> 10. <phase>generate-test-sources</phase> 11. <phase>process-test-sources</phase> 12. <phase>generate-test-resources</phase> 13. <phase>process-test-resources</phase> 14. <phase>test-compile</phase> 15. <phase>process-test-classes</phase> 16. <phase>test</phase> 17. <phase>prepare-package</phase> 18. <phase>package</phase> 19. <phase>pre-integration-test</phase> 20. <phase>integration-test</phase> 21. <phase>post-integration-test</phase> 22. <phase>verify</phase> 23. <phase>install</phase> 24. <phase>deploy</phase> 25. </phases> |

运行任何一个阶段的时候，它前面的所有阶段都会被运行，这也就是为什么我们运行 mvn install 的时候，代码会被编译，测试，打包。此外，Maven 的插件机制是完全依赖 Maven 的生命周期的，因此理解生命周期至关重要









### 2.5Maven仓库

1）仓库的概念

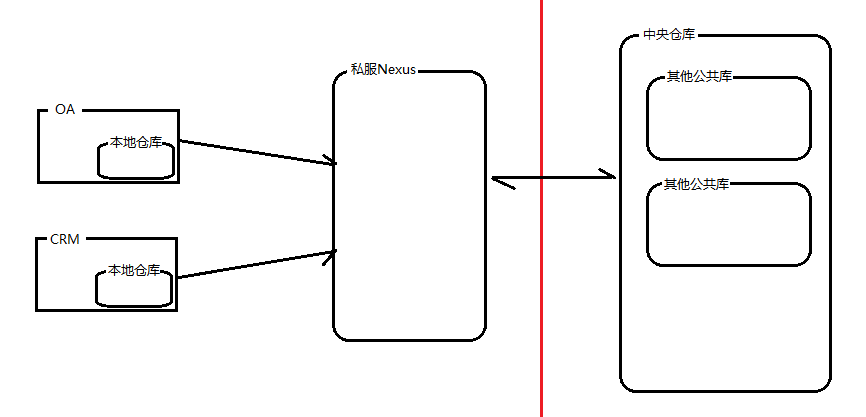
在 Maven 中，任何一个依赖、插件或者项目构建的输出，都可以称之为构件。

Maven 在某个统一的位置存储所有项目的共享的构件，这个统一的位置，我们就称之为仓库。（仓库就是存放依赖和插件的地方）任何的构件都有唯一的坐标，Maven 根据这个坐标定义了构件在仓库中的唯一存储路径。

2）仓库的分类

Maven 的仓库只有两大类：1.本地仓库 2.远程仓库。

在远程仓库中又分成了 3 种：① 中央仓库 ② 私服 ③其它公共库



3）仓库详解

**本地仓库**

是 Maven 在本地存储构件的地方。Maven 本地仓库的默认位置：无论是

Windows 还是 Linux，在用户的目录下都有一个.m2/repository/的仓库目录，这就是 Maven 仓库的默认位置。

【注】maven 的本地仓库，在安装 maven 后并不会创建，它是在第一次执行 maven 命令的时候才被创建。

【更改默认的本地仓库位置】

更改本地仓库有两种方式：① 基于用户范围 ② 基于全局范围

**更改配置用户范围的本地仓库**：先在/.m2/目录下创建 settings.xml 文件，

然后在~/.m2/settings.xml，设置 localRepository 元素的值为想要的仓库地址

<settings>

<localRepository>D:\maven\_new\_repository</localRepository>

</settings>

这时候，maven 的本地仓库地址就变成了 D:\maven\_new\_repository ，注：此时配置的 maven 的本地仓库是属于用户范围的。

**更改配置全局范围的本地仓库**：在 M2\_HOME/conf/settings.xml 中更改配置，

更改配置的方法同上

【注】此时更改后，所有的用户都会受到影响，而且如果 maven 进行升级，那么所有的配置都会被清除，所以要提前复制和备份 M2\_HOME/conf/settings.xml 文件，一般情况下不推荐配置全局的 settings.xml

**远程仓库**

### 2.6Maven常用命令

1）mvn compile 编译

|  |
| --- |
| G:\eclipse\workspace\my-app>mvn compile  [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building Maven Quick Start Archetype 1.0-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO] 插件 版本号 目标  [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dep  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\main\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---  [INFO] Changes detected - recompiling the module!  [WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is platform dependent  [INFO] Compiling 1 source file to G:\eclipse\workspace\my-app\target\classes  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 1.751 s  [INFO] Finished at: 2018-12-16T15:12:17+08:00  [INFO] Final Memory: 12M/111M  [INFO] ------------------------------------------------------------------------ |
| target  ├───classes  │ └───com  │ └───mycompany  │ └───app  │ App.class  │  └───maven-status  └───maven-compiler-plugin  └───compile  └───default-compile  createdFiles.lst  inputFiles.lst |

2）mvn test 测试

|  |
| --- |
| G:\eclipse\workspace\my-app>mvn test  [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building Maven Quick Start Archetype 1.0-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO]  [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dep  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\main\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---  [INFO] Nothing to compile - all classes are up to date  [INFO]  [INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dep  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\test\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---  [INFO] Nothing to compile - all classes are up to date  [INFO]  [INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---  [INFO] Surefire report directory: G:\eclipse\workspace\my-app\target\surefire-reports  -------------------------------------------------------  T E S T S  -------------------------------------------------------  Running com.mycompany.app.AppTest  \*\*\*\*\*\*\*\*testApp\*\*\*\*\*\*\*\*  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.064 sec  Results :  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 2.797 s  [INFO] Finished at: 2018-12-16T16:31:02+08:00  [INFO] Final Memory: 8M/151M  [INFO] ------------------------------------------------------------------------ |

3）mvn package 打包

|  |
| --- |
| G:\eclipse\workspace\my-app>mvn package  [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building Maven Quick Start Archetype 1.0-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO]  [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dep  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\main\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---  [INFO] Nothing to compile - all classes are up to date  [INFO]  [INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dep  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\test\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---  [INFO] Changes detected - recompiling the module!  [WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is platform dependent  [INFO] Compiling 1 source file to G:\eclipse\workspace\my-app\target\test-classes  [INFO]  [INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---  [INFO] Surefire report directory: G:\eclipse\workspace\my-app\target\surefire-reports  -------------------------------------------------------  T E S T S  -------------------------------------------------------  Running com.mycompany.app.AppTest  \*\*\*\*\*\*\*\*testApp\*\*\*\*\*\*\*\*  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.084 sec  Results :  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0  [INFO]  [INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ my-app ---  [INFO] Building jar: G:\eclipse\workspace\my-app\target\my-app-1.0-SNAPSHOT.jar  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 3.576 s  [INFO] Finished at: 2018-12-16T15:14:30+08:00  [INFO] Final Memory: 16M/166M  [INFO] ------------------------------------------------------------------------ |
|  |

4）mvn install 安装到maven仓库

|  |
| --- |
| G:\eclipse\workspace\my-app>mvn install  [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building Maven Quick Start Archetype 1.0-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO]  [INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\main\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---  [INFO] Changes detected - recompiling the module!  [WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is platform dependent!  [INFO] Compiling 1 source file to G:\eclipse\workspace\my-app\target\classes  [INFO]  [INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---  [WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!  [INFO] skip non existing resourceDirectory G:\eclipse\workspace\my-app\src\test\resources  [INFO]  [INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---  [INFO] Changes detected - recompiling the module!  [WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is platform dependent!  [INFO] Compiling 1 source file to G:\eclipse\workspace\my-app\target\test-classes  [INFO]  [INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---  [INFO] Surefire report directory: G:\eclipse\workspace\my-app\target\surefire-reports  -------------------------------------------------------  T E S T S  -------------------------------------------------------  Running com.mycompany.app.AppTest  \*\*\*\*\*\*\*\*testApp\*\*\*\*\*\*\*\*  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.063 sec  Results :  Tests run: 1, Failures: 0, Errors: 0, Skipped: 0  [INFO]  [INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ my-app ---  [INFO] Building jar: G:\eclipse\workspace\my-app\target\my-app-1.0-SNAPSHOT.jar  [INFO]  [INFO] --- maven-install-plugin:2.4:install (default-install) @ my-app ---  [INFO] Installing G:\eclipse\workspace\my-app\target\my-app-1.0-SNAPSHOT.jar to C:\Users\BYF\.m2\repository\com\mycompany\app\my-app\1.0-SNAPSHOT\my-a  pp-1.0-SNAPSHOT.jar  [INFO] Installing G:\eclipse\workspace\my-app\pom.xml to C:\Users\BYF\.m2\repository\com\mycompany\app\my-app\1.0-SNAPSHOT\my-app-1.0-SNAPSHOT.pom  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 3.528 s  [INFO] Finished at: 2018-12-16T16:16:05+08:00  [INFO] Final Memory: 17M/162M  [INFO] ------------------------------------------------------------------------ |
|  |

5）mvn clean清除

|  |
| --- |
| G:\eclipse\workspace\my-app>mvn clean  [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building Maven Quick Start Archetype 1.0-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO]  [INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ my-app ---  [INFO] Deleting G:\eclipse\workspace\my-app\target  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 0.368 s  [INFO] Finished at: 2018-12-16T16:36:55+08:00  [INFO] Final Memory: 6M/117M  [INFO] ------------------------------------------------------------------------ |
|  |

### 2.7Maven坐标依赖

**1）scope：依赖范围**

 compile:编译范围,默认scope,在classpath中存在

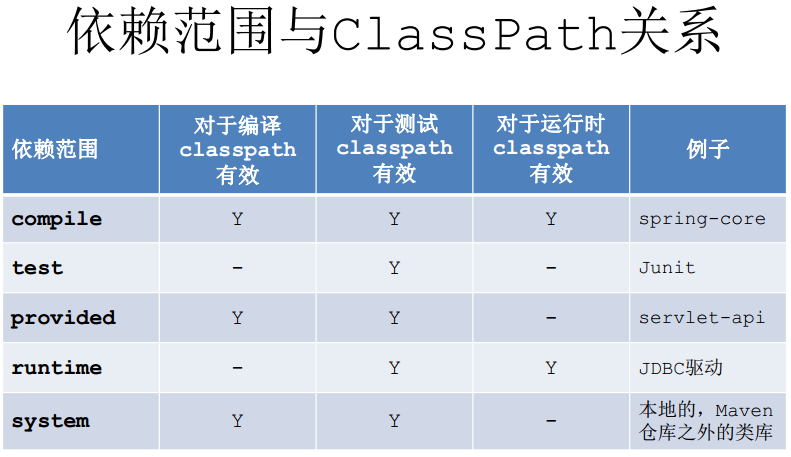
 provided:已提供范围,比如容器提供Servlet API

 runtime:运行时范围,编译不需要,接口与实现分离

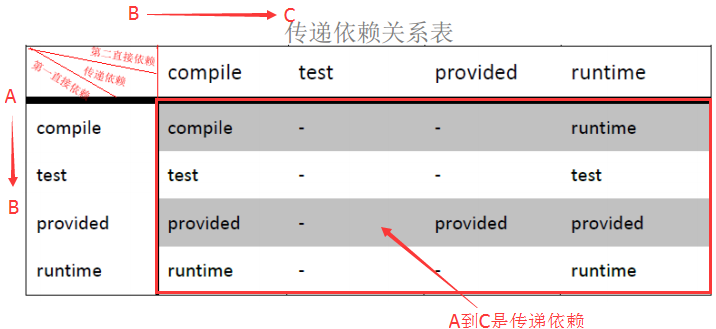
 test:测试范围,单元测试环境需要

 system:系统范围,自定义构件，指定systemPath

 import：导入依赖



**2）Maven传递依赖**



传递依赖的依赖范围要么是两个直接依赖的其中一个值

传递依赖会取 两个依赖范围中的一个值，或者中断。

总结：当传递依赖中断的时候，直接在项目中引入依赖即可。

**3）依赖冲突解决**

传递依赖： A依赖B,B依赖C，那么C是A的传递依赖，B是A的直接依赖。

解决依赖有两个原则：

1.路径最短原则

例如：a-log4j1.1 a-b-log4j1.2 最终会用到log4j1.1

2.最先声明优先。

例如：a模块的pom.xml引入log4j1.2，a的父模块也引用log4j1.1，那么最先使用在前边声明的版本log4j1.1。

**4）依赖调解**

A->B->D->X(1.6)

A->D-X(2.0)

X是A的传递依赖。B是A的直接依赖，D也是A的直接依赖。

依赖调解原则：

第一原则：路径近者优先原则 x2.0传递给A最近

第二原则：第一声明者优先原则 当路径相等时，则由POM声明的依赖顺序决定

### 2.8Maven聚合和继承

1）聚合（平行结构）

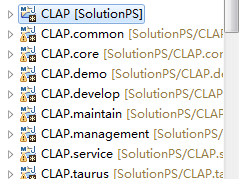


--pom.xml

|  |
| --- |
| <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.mycompany.app</groupId>  <artifactId>dma-aggregate</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>pom</packaging>  <modules>  <module>../dma-commons</module>  <module>../dma-core</module>  <module>../dma-service</module>  </modules>  </project> |

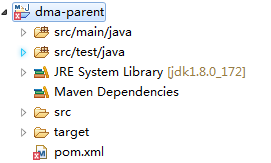
|  |
| --- |
| [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building dma-aggregate 0.0.1-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO] ------------------------------------------------------------------------  [INFO] Reactor Summary:  [INFO]  [INFO] dma-commons ........................................ SUCCESS [ 1.831 s]  [INFO] dma-core ........................................... SUCCESS [ 1.569 s]  [INFO] dma-service ........................................ SUCCESS [ 2.241 s]  [INFO] dma-aggregate ...................................... SUCCESS [ 0.001 s]  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 5.752 s  [INFO] Finished at: 2018-12-17T22:30:12+08:00  [INFO] Final Memory: 15M/125M  [INFO] ------------------------------------------------------------------------ |

2）聚合（父子结构）



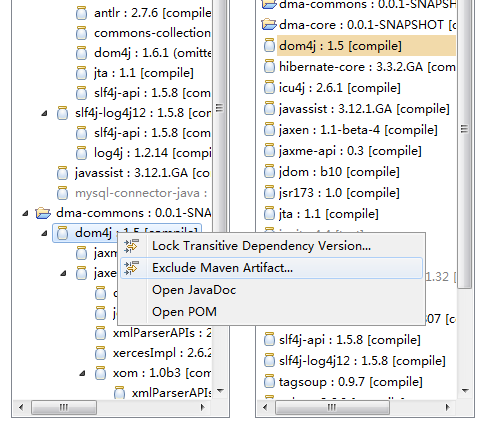
|  |
| --- |
| <modules>  <module>CLAP.common</module>  <module>CLAP.core</module>  <module>CLAP.service</module>  <module>CLAP.management</module>  <module>CLAP.develop</module>  <module>CLAP.maintain</module>  <module>CLAP.taurus</module>  <!-- 集成测试及演示项目 -->  <module>CLAP.demo</module>  </modules> |

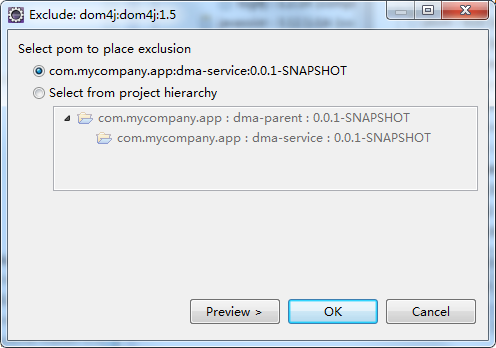
3）继承



|  |
| --- |
| <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.mycompany.app</groupId>  <artifactId>dma-parent</artifactId>  <version>0.0.1-SNAPSHOT</version>  <packaging>pom</packaging>    <!-- 子模块继承的同时，聚合各个子模块 -->  <modules>  <module>../dma-commons</module>  <module>../dma-core</module>  <module>../dma-service</module>  </modules>    <dependencyManagement>  <dependencies>  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>3.3.2.GA</version>  </dependency>  <!-- jar包版本不对会报错误 -->  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-log4j12</artifactId>  <version>1.5.8</version>  </dependency>  <dependency>  <groupId>javassist</groupId>  <artifactId>javassist</artifactId>  <version>3.12.1.GA</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.4</version>  <scope>test</scope>  </dependency>  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>5.1.32</version>  <scope>runtime</scope>  </dependency>  <dependency>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-service</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <dependency>  <groupId>jstl</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version>  </dependency>  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>servlet-api</artifactId>  <version>2.5</version>  <scope>provided</scope>  </dependency>    </dependencies>  </dependencyManagement>    <build>  <pluginManagement>  <plugins>  <plugin>  <groupId>org.mortbay.jetty</groupId>  <artifactId>jetty-maven-plugin</artifactId>  <configuration>  <scanIntervalSeconds>10</scanIntervalSeconds>  <webApp>  <contextPath>/example</contextPath>  </webApp>  <connectors>  <connector implementation=*"org.eclipse.jetty.server.nio.SelectChannelConnector"*>  <port>8888</port>  <maxIdleTime>60000</maxIdleTime>  </connector>  </connectors>  </configuration>  </plugin>  </plugins>  </pluginManagement>  </build>  </project> |

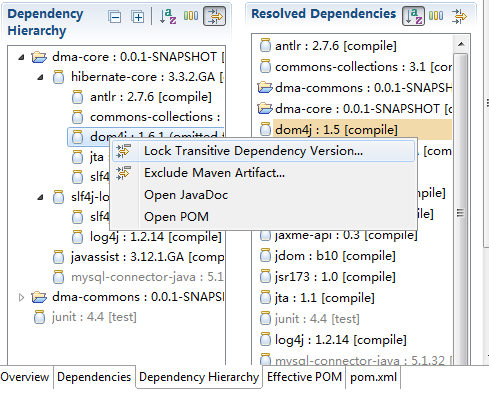
4）依赖冲突管理--排除exclude

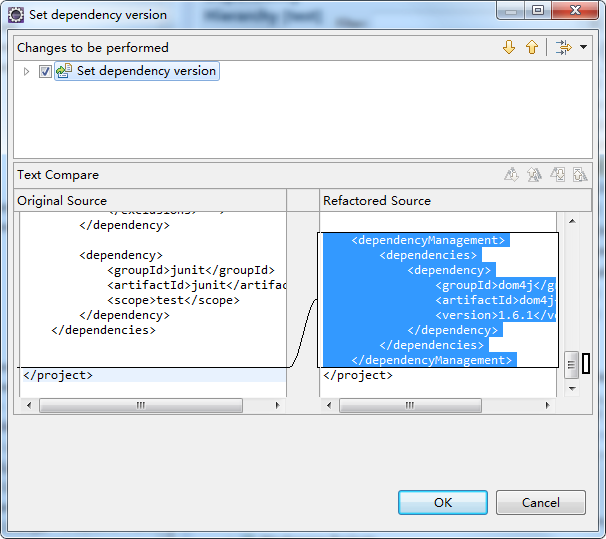




|  |
| --- |
| <!-- 由于commons模块的dom4j是低版本1.5，排除该模块的dom4j -->  <dependency>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-commons</artifactId>  <version>0.0.1-SNAPSHOT</version>  <exclusions>  <exclusion>  <artifactId>dom4j</artifactId>  <groupId>dom4j</groupId>  </exclusion>  </exclusions>  </dependency> |

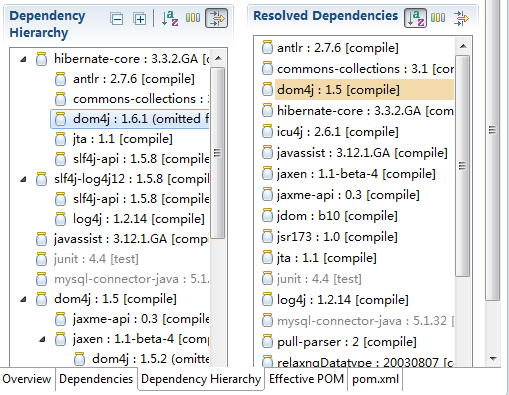
5）依赖冲突管理--版本锁定





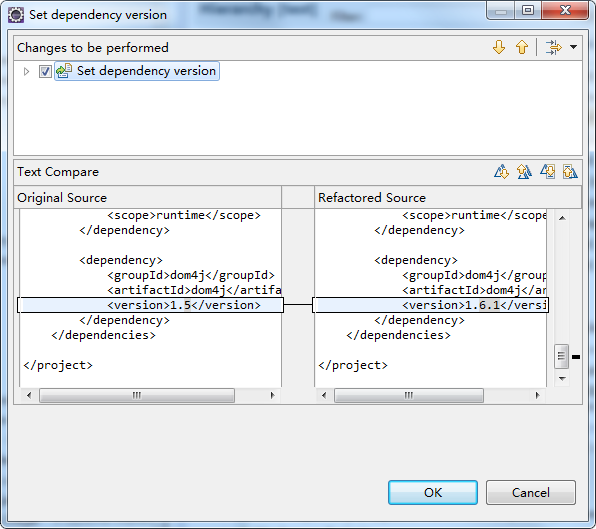
|  |
| --- |
| ...  <!-- 版本锁定，指定依赖的jar包版本 -->  <dependencyManagement>  <dependencies>  <dependency>  <groupId>dom4j</groupId>  <artifactId>dom4j</artifactId>  <version>1.6.1</version>  </dependency>  </dependencies>  </dependencyManagement>  </project> |

6）传递依赖冲突解决



|  |
| --- |
| <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <exclusions>  <exclusion>  <groupId>dom4j</groupId>  <artifactId>dom4j</artifactId>  </exclusion>  </exclusions>  </dependency>  <dependency>  <groupId>dom4j</groupId>  <artifactId>dom4j</artifactId>  <version>1.5</version>  </dependency> |

7）传递依赖冲突解决--本地修改冲突的版本



|  |
| --- |
| <dependency>  <groupId>dom4j</groupId>  <artifactId>dom4j</artifactId>  <version>1.6.1</version>  </dependency> |

### 2.9其他

1）概念模型

1. 常见问题排除

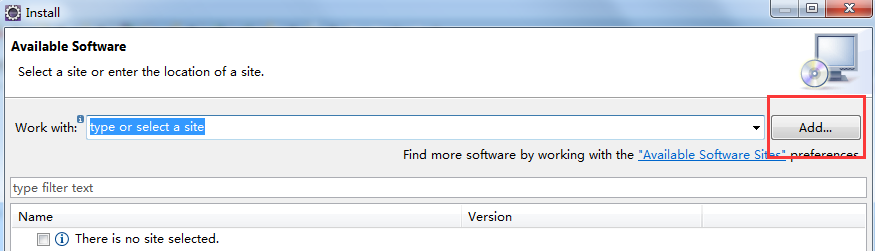
org.apache.maven.archiver.MavenArchiver.getManifest错误

eclipse导入新的maven项目时，pom.xml第一行报错：

org.apache.maven.archiver.MavenArchiver.getManifest(org.apache.maven.project.MavenProject, org.apache.maven.archiver.MavenArchiveConfiguration)

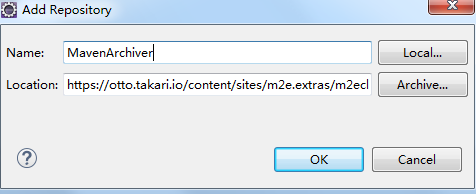
解决办法：

1、Help-->Install New SoftWare



2、MavenArchiver

https://otto.takari.io/content/sites/m2e.extras/m2eclipse-mavenarchiver/0.17.2/N/LATEST/



3.按照安装提示，一直next，重启eclipse之后，执行maven-update project

3）版本介绍

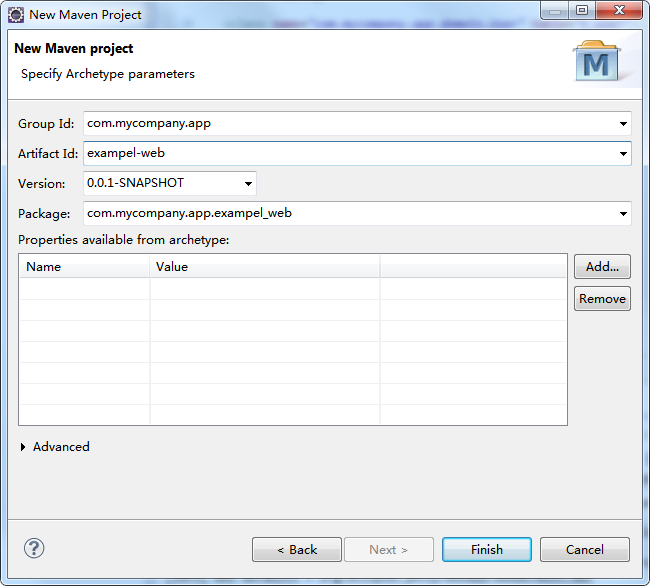
4）常量

|  |
| --- |
| <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <catalina.home>D:\J2EE\_HOME\tomcat7</catalina.home>  <hibernate.version>3.3.2.GA</hibernate.version>  </properties>  ...  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>${hibernate.version}</version>  </dependency> |

5）归类依赖

## 第3节 Maven项目

### 3.1构建项目



### 3.2背景案例

1. 客户需求

注册账号服务

根据账号查询密码服务

2需求阐述

了解了客户服务之后，从软件工程的角度来分析下服务需求。

2.1需求用例

注册账号

主要场景：

1. 用户访问注册页面
2. 用户输入用户名和密码
3. 用户提交注册请求
4. 系统检测账号是否被注册过
5. 系统保护账号信息
6. 用户登录

扩展场景：

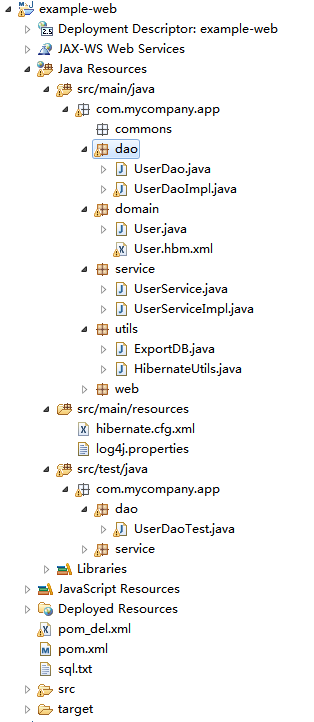
4a:系统检测到账号已经被注册

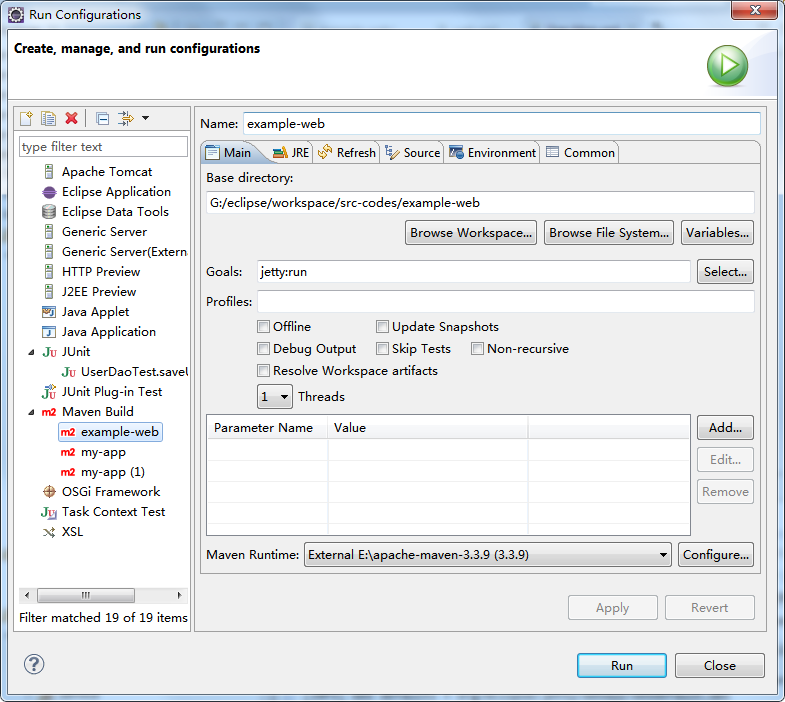
1. 提示账号被注册
2. 跳转到步骤2

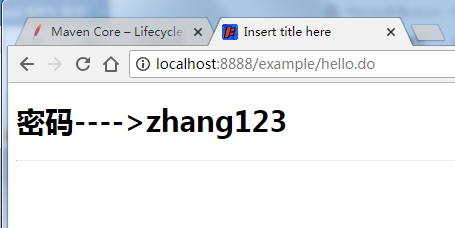
### 3.3web项目

--sql.txt

|  |
| --- |
| drop table if exists t\_user;  create table t\_user (  id varchar(255) not null,  name varchar(255),  password varchar(255),  primary key (id)  ); |

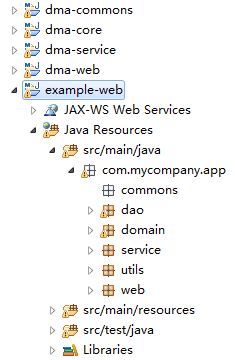






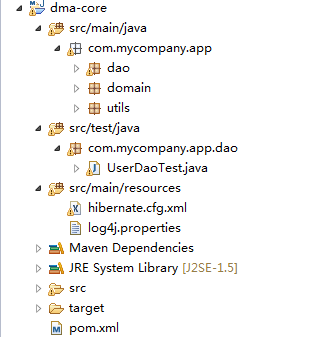
### 3.4项目拆分





1）dma-core



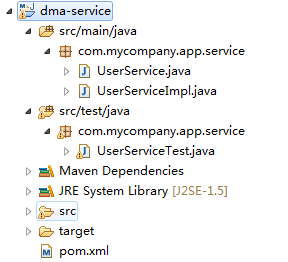


-pom.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <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.mycompany.app</groupId>  <artifactId>dma-core</artifactId>  <version>0.0.1-SNAPSHOT</version>  <name>dma-core</name>  <!-- FIXME change it to the project's website -->  <url>http://www.example.com</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <catalina.home>D:\J2EE\_HOME\tomcat7</catalina.home>  <hibernate.version>3.3.2.GA</hibernate.version>  </properties>  <dependencies>  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>${hibernate.version}</version>  </dependency>  <!-- jar包版本不对会报错误 -->  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-log4j12</artifactId>  <version>1.5.8</version>  </dependency>  <dependency>  <groupId>javassist</groupId>  <artifactId>javassist</artifactId>  <version>3.12.1.GA</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.4</version>  <scope>test</scope>  </dependency>  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>5.1.32</version>  <scope>runtime</scope>  </dependency>  </dependencies>    </project> |

2）dma-service



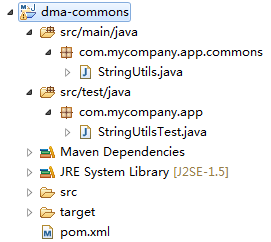


--pom.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <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.mycompany.app</groupId>  <artifactId>dma-service</artifactId>  <version>0.0.1-SNAPSHOT</version>  <name>dma-service</name>  <!-- FIXME change it to the project's website -->  <url>http://www.example.com</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <catalina.home>D:\J2EE\_HOME\tomcat7</catalina.home>  <hibernate.version>3.3.2.GA</hibernate.version>  </properties>  <dependencies>  <dependency>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-core</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.4</version>  <scope>test</scope>  </dependency>  </dependencies>  </project> |

3）dma-commons



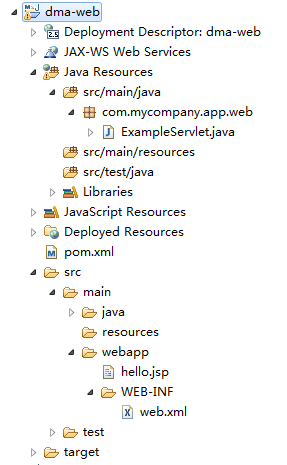


--pom.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <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.mycompany.app</groupId>  <artifactId>dma-commons</artifactId>  <version>0.0.1-SNAPSHOT</version>  <name>dma-commons</name>  <!-- FIXME change it to the project's website -->  <url>http://www.example.com</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  </properties>  <dependencies>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>4.11</version>  <scope>test</scope>  </dependency>  </dependencies>    </project> |

4）dma-web





--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/maven-v4\_0\_0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-web</artifactId>  <packaging>war</packaging>  <version>0.0.1-SNAPSHOT</version>  <name>dma-web Maven Webapp</name>  <url>http://maven.apache.org</url>  <dependencies>  <dependency>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-service</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <dependency>  <groupId>com.mycompany.app</groupId>  <artifactId>dma-commons</artifactId>  <version>0.0.1-SNAPSHOT</version>  </dependency>  <!-- app-web模块 -->  <dependency>  <groupId>jstl</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version>  </dependency>  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>servlet-api</artifactId>  <version>2.5</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <version>3.8.1</version>  <scope>test</scope>  </dependency>  </dependencies>  <build>  <finalName>dma-web</finalName>  <plugins>  <plugin>  <groupId>org.mortbay.jetty</groupId>  <artifactId>jetty-maven-plugin</artifactId>  <configuration>  <scanIntervalSeconds>10</scanIntervalSeconds>  <webApp>  <contextPath>/example</contextPath>  </webApp>  <connectors>  <connector implementation=*"org.eclipse.jetty.server.nio.SelectChannelConnector"*>  <port>8888</port>  <maxIdleTime>60000</maxIdleTime>  </connector>  </connectors>  </configuration>  </plugin>  </plugins>  </build>  </project> |

--web.xml

|  |
| --- |
| <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"*>  <display-name>Archetype Created Web Application</display-name>      <servlet>  <servlet-name>ExampleServlet</servlet-name>  <servlet-class>com.mycompany.app.web.ExampleServlet</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>ExampleServlet</servlet-name>  <url-pattern>/hello.do</url-pattern>  </servlet-mapping>      </web-app> |

--hello.jsp

|  |
| --- |
| <%@ 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>Insert title here</title>  </head>  <body>  <h1>密码---->${password}</h1>  <hr/>  </body>  </html> |

## 第4节 m2eclipse

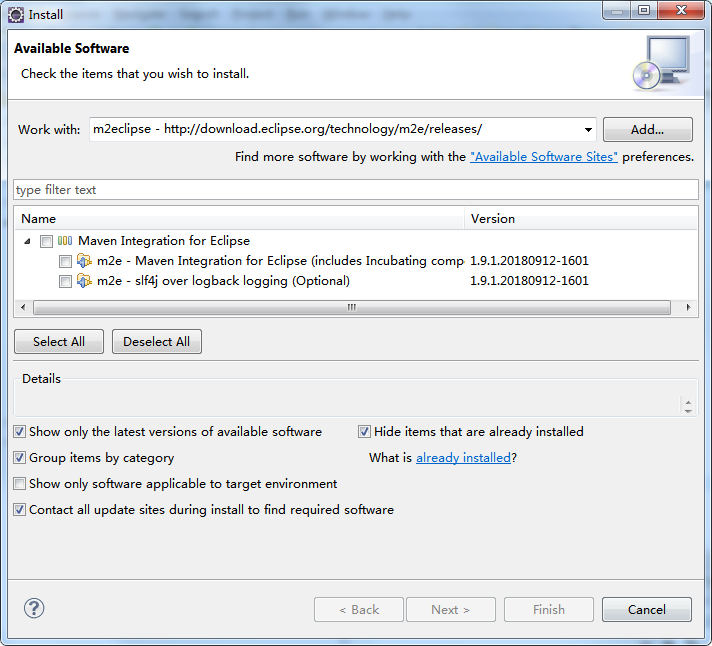
1）m2eclipse安装

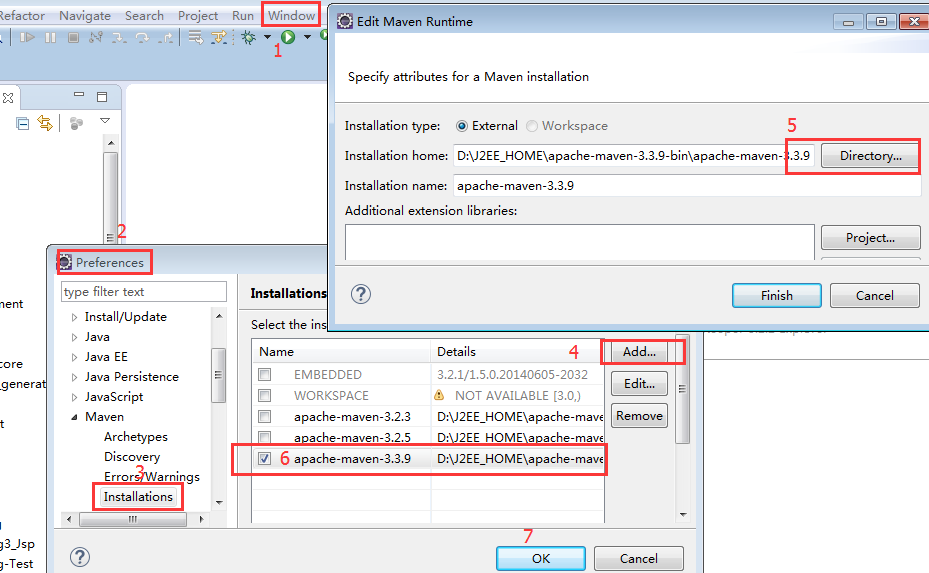
有的eclipse版本已经集成了maven插件，例如eclipse-jee-kepler-SR2-win32-x86\_64.zip；

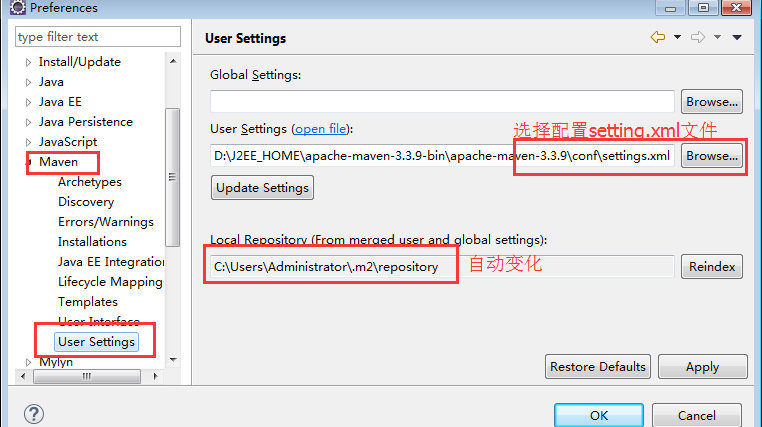
如果eclipse没有集成maven插件，需要手动安装。

1.在线安装

地址：<http://download.eclipse.org/technology/m2e/releases/>







下载插件包安装...

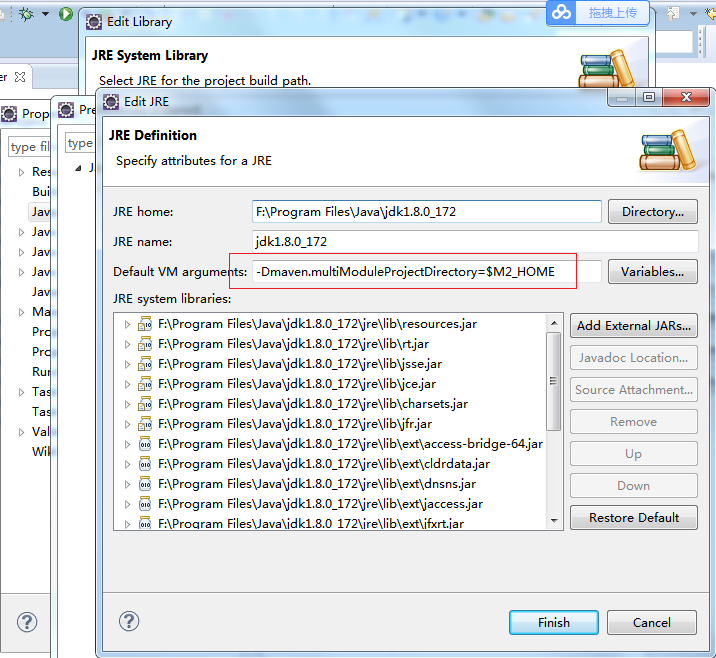
2）m2eclipse使用

当我们在测试的时候，报下面错误

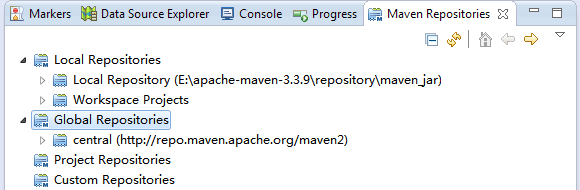
-Dmaven.multiModuleProjectDirectory system property is not set. Check $M2\_HOME environment variable and mvn script match.

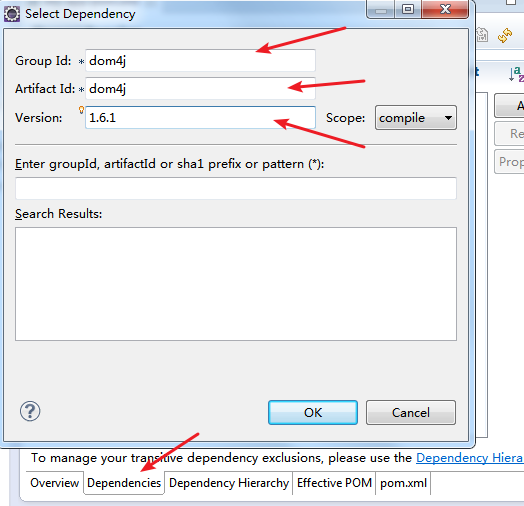
解决方法：

-Dmaven.multiModuleProjectDirectory=$M2\_HOME



3）m2eclipse仓库视图





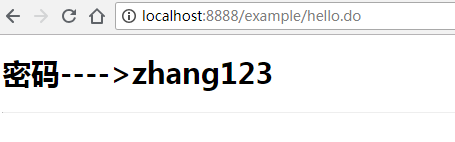
## 第5节 Maven插件

1）部署

jetty:run

|  |
| --- |
| <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-war-plugin</artifactId>  <version>2.6</version>  </plugin>  <plugin>  <groupId>org.mortbay.jetty</groupId>  <artifactId>jetty-maven-plugin</artifactId>  <configuration>  <scanIntervalSeconds>10</scanIntervalSeconds>  <webApp>  <contextPath>/example</contextPath>  </webApp>  <connectors>  <connector implementation=*"org.eclipse.jetty.server.nio.SelectChannelConnector"*>  <port>8888</port>  <maxIdleTime>60000</maxIdleTime>  </connector>  </connectors>  </configuration>  </plugin> |

http://localhost:8888/example/hello.do



package cargo:run

|  |
| --- |
| <plugin>  <groupId>org.codehaus.cargo</groupId>  <artifactId>cargo-maven2-plugin</artifactId>  <version>1.4.9</version>  <configuration>  <container>  <containerId>tomcat7x</containerId>  <home>E:\apache-tomcat-7.0.82-windows-x64\apache-tomcat-7.0.82</home>  </container>  <configuration>  <type>standalone</type>  <home>${project.build.directory}/tomcat7x</home>  <properties>  <cargo.servlet.port>8084</cargo.servlet.port>  </properties>  </configuration>  </configuration>  </plugin> |

|  |
| --- |
| <plugin>  <groupId>org.codehaus.cargo</groupId>  <artifactId>cargo-maven2-plugin</artifactId>  <version>1.5</version>  <configuration>  <container>  <containerId>tomcat7x</containerId>  <home>D:\J2EE\_HOME\tomcat7</home>  </container>  <configuration>  <type>existing</type>  <home>D:\J2EE\_HOME\tomcat7</home>  </configuration>  </configuration>  </plugin> |

tomcat:run

启动tomcat

|  |
| --- |
| <role rolename="admin-gui"/>  <role rolename="manager-gui"/>  <role rolename="manager-script"/>  <user username="admin" password="admin" roles="manager-gui,admin-gui,manager-script"/> |

修改settings.xml

|  |
| --- |
| <server>  <id>tomcat</id>  <username>admin</username>  <password>admin</password>  </server> |

|  |
| --- |
| <plugin>  <groupId>org.apache.tomcat.maven</groupId>  <artifactId>tomcat7-maven-plugin</artifactId>  <version>2.0-SNAPSHOT</version>  <configuration>  <url>http://localhost:8080/manager/html</url>  <server>tomcat7x</server>  </configuration>  </plugin> |

2）测试

3）编译

4）打包

5）测试报告

## Nexus私服

### 6.1Nexus安装

1）nexus安装

**Bundle方式安装**

* 下载后的文件：nexus-2.3.1-01-bundle.zip

安装：直接解压到某个目录即可解压后，会有两个目录：

nexus-2.3.1-01 运行所需要的文件

sonatype-work ：私服的默认仓库，配置文件，日志文件，仓库文件

* 将bin目录添加到windows系统环境变量中
* 配置java文件的路径

${NEXUS\_HOME}\bin\jsw\conf\wrapper.conf文件

修改wrapper.java.command=java为你的java.exe文件的路径，并去掉#注释

|  |
| --- |
| # Java Application  wrapper.java.command=F:\Program Files\Java\jdk1.7.0\_17\bin\java |

* 安装nexus服务，并启动nexus

|  |
| --- |
| C:\Users\BYF>nexus install  wrapper | nexus installed.  C:\Users\BYF>nexus start  wrapper | Starting the nexus service...  wrapper | Waiting to start...  wrapper | Waiting to start...  wrapper | Waiting to start...  wrapper | Waiting to start...  wrapper | Waiting to start...  wrapper | nexus started. |

例如：wrapper.java.command=${Your\_Java\_Home}\jdk1.7\bin\java

注意：执行nexus start时，报错${NEXUS\_HOME}\logs\wrapper.log

|  |
| --- |
| jvm 1 | 2018-12-16 22:22:45 INFO [er\_start\_runner] - org.sonatype.nexus.bootstrap.jsw.JswLauncher - Current directory: G:\eclipse\nexus-2.3.1-01-bundle\nexus-2.3.1-01  jvm 1 | Error in WrapperListener.start callback. java.lang.StackOverflowError |

把jdk版本换成1.7.(换jdk方式1：改环境变量；方式2：修改nexus/bin/jsw/conf/wrapper.conf 第15行 wrapper.java.command=java 改为 wrapper.java.command=/jdk路径/bin/java )

|  |
| --- |
| C:\Windows\system32>nexus install  wrapper | OpenSCManager failed - 拒绝访问。 (0x5) |

使用管理员打开cmd进行安装。

* 登录nexus

地址：http://localhost:8081/nexus

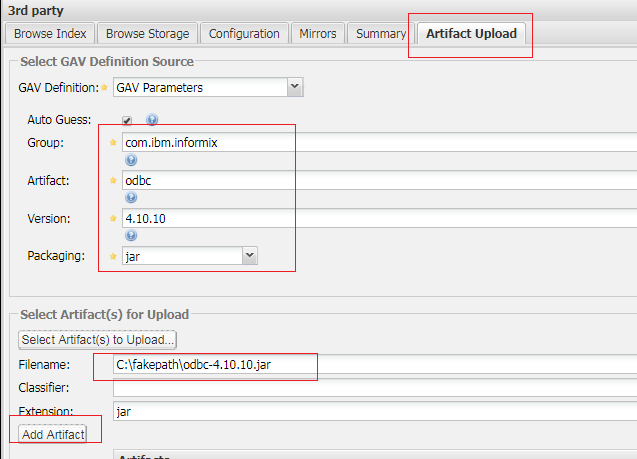
默认用户名密码 admin/admin123

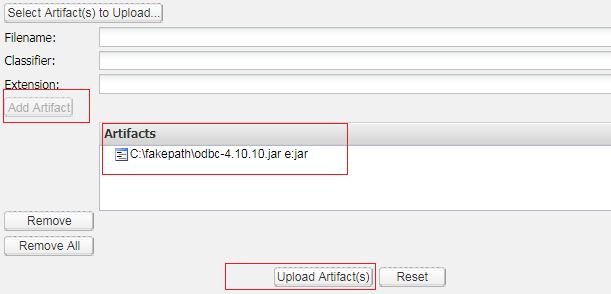
|  |
| --- |
| wrapper | Starting the nexus service...  wrapper | --> Wrapper Started as Service  wrapper | Launching a JVM...  wrapper | Unable to execute Java command. 参数错误。 (0x57)  wrapper | ""F:\Program Files\Java\jdk1.8.0\_172\bin\java"" -Djava.library.path="bin/jsw/lib" -classpath "bin/jsw/lib/wrapper-3.2.3.jar;./lib/appcontext-3.2.jar;./lib/javax.servlet-3.0.0.v201112011016.jar;./lib/jetty-continuation-8.1.8.v20121106.jar;./lib/jetty-deploy-8.1.8.v20121106.jar;./lib/jetty-http-8.1.8.v20121106.jar;./lib/jetty-io-8.1.8.v20121106.jar;./lib/jetty-jmx-8.1.8.v20121106.jar;./lib/jetty-rewrite-8.1.8.v20121106.jar;./lib/jetty-security-8.1.8.v20121106.jar;./lib/jetty-server-8.1.8.v20121106.jar;./lib/jetty-servlet-8.1.8.v20121106.jar;./lib/jetty-util-8.1.8.v20121106.jar;./lib/jetty-webapp-8.1.8.v20121106.jar;./lib/jetty-xml-8.1.8.v20121106.jar;./lib/logback-classic-1.0.7.jar;./lib/logback-core-1.0.7.jar;./lib/nexus-bootstrap-2.3.1-01.jar;./lib/nexus-logging-extras-appender-2.3.1-01.jar;./lib/plexus-interpolation-1.15.jar;./lib/sisu-jetty8-1.4.2.jar;./lib/slf4j-api-1.7.2.jar;./conf/" -Dwrapper.key="39BzIK0bWOJSDMBg" -Dwrapper.port=32000 -Dwrapper.jvm.port.min=31000 -Dwrapper.jvm.port.max=31999 -Dwrapper.pid=7332 -Dwrapper.version="3.2.3" -Dwrapper.native\_library="wrapper" -Dwrapper.service="TRUE" -Dwrapper.cpu.timeout="10" -Dwrapper.jvmid=1 org.sonatype.nexus.bootstrap.jsw.JswLauncher ./conf/jetty.xml  wrapper | Critical error: wait for JVM process failed  wrapper | The nexus service was launched, but failed to start. |

使用jdk1.7.

### 6.2Mavne介绍上传构件

1）上传构件

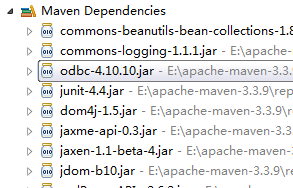




2）添加pom依赖

|  |
| --- |
| <dependency>  <groupId>com.ibm.informi</groupId>  <artifactId>odbc</artifactId>  <version>4.10.10</version>  </dependency>1 |

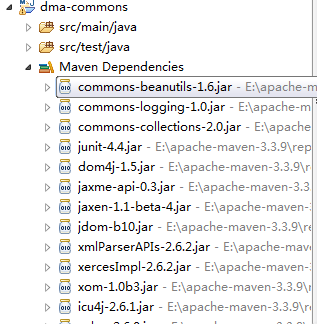
3）保存后可以看到下载的依赖

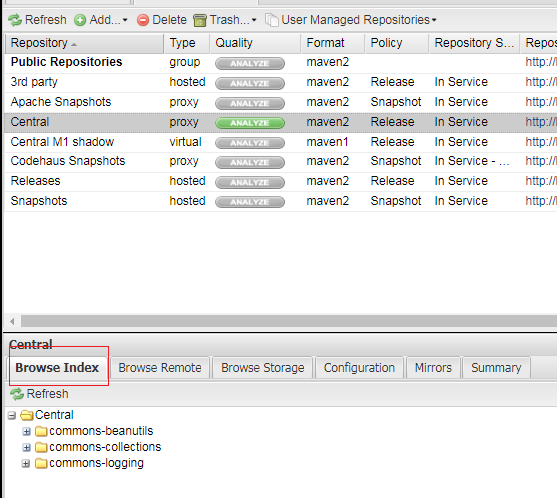


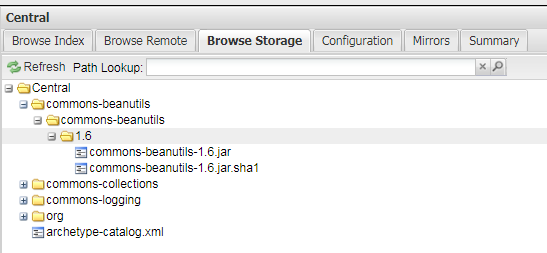
### 6.3Maven从Nexus下载构建

1）配置pom文件，单个项目从Nexus私服上下载依赖

|  |
| --- |
| <!-- 配置jar包的私服仓库 -->  <repositories>  <repository>  <id>app\_repository</id>  <name>app\_repository</name>  <url>http://127.0.0.1:8081/nexus/content/groups/public/</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </repository>  </repositories>  <!-- 配置插件的私服仓库 -->  <pluginRepositories>  <pluginRepository>  <id>app\_repository</id>  <name>app\_repository</name>  <url>http://127.0.0.1:8081/nexus/content/groups/public/</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </pluginRepository>  </pluginRepositories> |







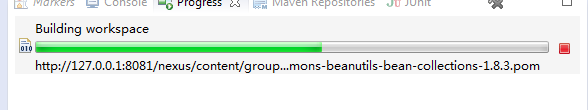
2）配置Maven的settings.xml使所有的maven项目都到私服上下载依赖

|  |
| --- |
| ...  <!-- 添加一个profile，指定id为：app-profile，并配置在pom中配置的私服仓库 -->  <profile>  <id>app-profile</id>  <!-- 配置jar包的私服仓库 -->  <repositories>  <repository>  <id>app\_repository</id>  <name>app\_repository</name>  <url>http://127.0.0.1:8081/nexus/content/groups/public/</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </repository>  </repositories>  <!-- 配置插件的私服仓库 -->  <pluginRepositories>  <pluginRepository>  <id>app\_repository</id>  <name>app\_repository</name>  <url>http://127.0.0.1:8081/nexus/content/groups/public/</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </pluginRepository>  </pluginRepositories>  </profile>  </profiles>  <!-- 激活指定的profile：app-profile，也就是profile中的id的值 -->  <activeProfiles>  <activeProfile>app-profile</activeProfile>  </activeProfiles> |

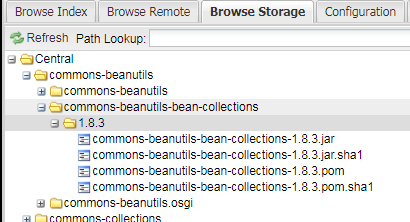
在pom中添加任何一个依赖坐标

|  |
| --- |
| <dependency>  <groupId>commons-beanutils</groupId>  <artifactId>commons-beanutils-bean-collections</artifactId>  <version>1.8.3</version>  </dependency> |

首先从本地的私服下载依赖

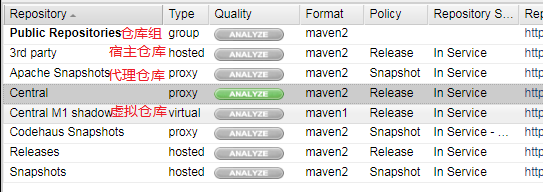


刷新私服中央仓库可以看到缓存了pom中下载的依赖

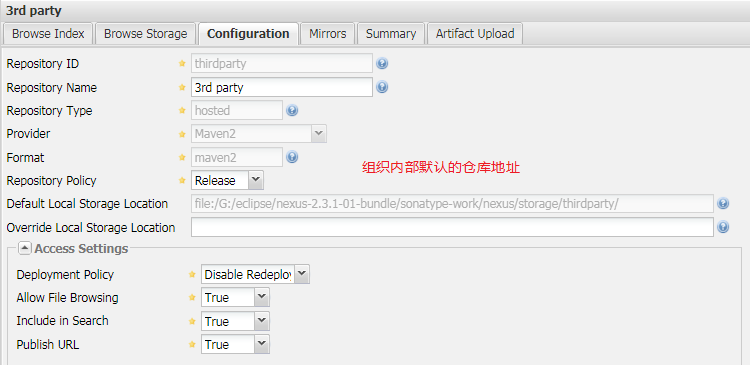


2）nexus分类

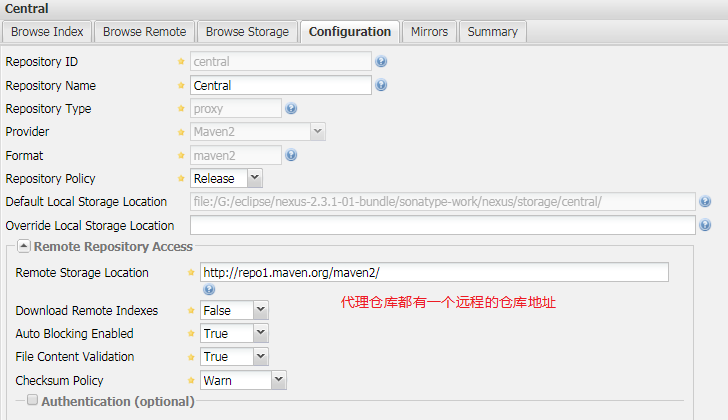
仓库类型



**宿主仓库**

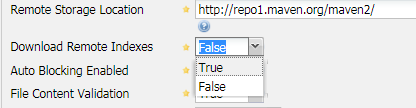


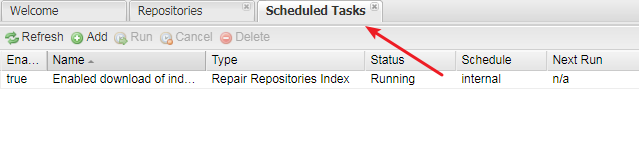
代理仓库



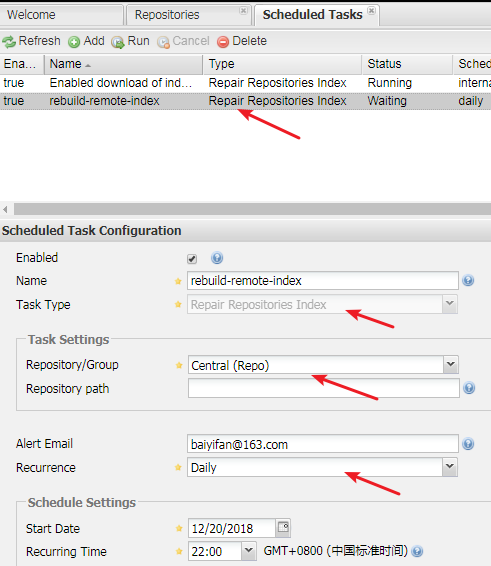
### 6.4nexus设置

1）下载远程仓库索引

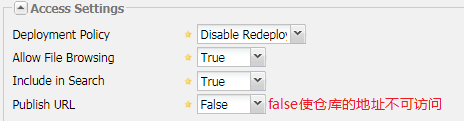




2）新建任务（重建远程中央仓库索引）

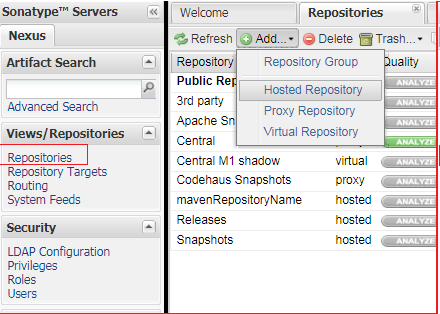


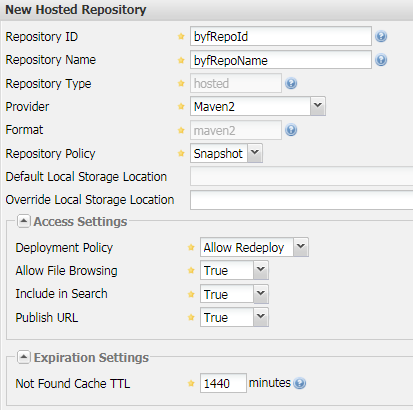
4）仓库设置



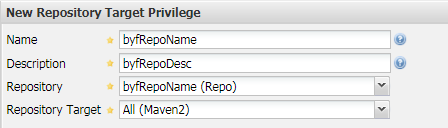
### 6.5nexus权限管理

·添加宿主仓库

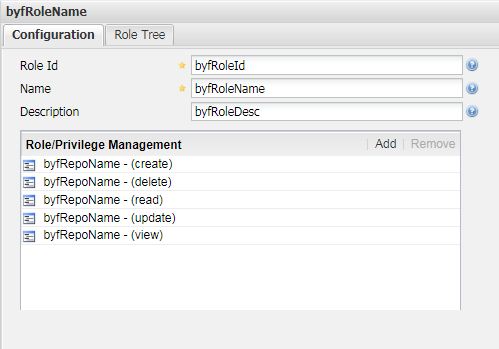




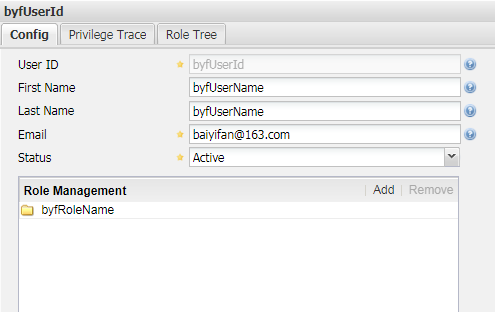
·创建权限



·创建角色



·创建用户



### 6.6修改maven中央仓库

1）修改maven全局配置文件settings.xml

|  |
| --- |
| <!-- 添加一个profile，指定id为：central-profile，并配置在pom中配置的私服仓库 -->  <profile>  <id>central-profile</id>  <!-- 配置jar包的私服仓库 -->  <repositories>  <repository>  <id>central</id>  <name>central\_repository</name>  <!-- 中央仓库配置的文件maven-model-builder-3.3.9-sources.jar\org\apache\maven\model\pom-4.0.0.xml -->  <url>https://repo.maven.apache.org/maven2</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </repository>  </repositories>  <!-- 配置插件的私服仓库 -->  <pluginRepositories>  <pluginRepository>  <id>central</id>  <name>central\_repository</name>  <!-- 中央仓库配置的文件maven-model-builder-3.3.9-sources.jar\org\apache\maven\model\pom-4.0.0.xml -->  <url>https://repo.maven.apache.org/maven2</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </pluginRepository>  </pluginRepositories>  </profile> |

2）激活中央仓库的profile

|  |
| --- |
| <!-- 激活指定的profile：app-profile，也就是profile中的id的值 -->  <activeProfiles>  <!-- <activeProfile>app-profile</activeProfile> -->  <activeProfile>central-profile</activeProfile>  </activeProfiles> |

3）使用maven项目测试：maven test

|  |
| --- |
| [INFO] Scanning for projects...  [INFO]  [INFO] ------------------------------------------------------------------------  [INFO] Building dma-service 0.0.1-SNAPSHOT  [INFO] ------------------------------------------------------------------------  [INFO] Downloading: https://repo.maven.apache.org/maven2/com/mycompany/app/dma-core/0.0.1-SNAPSHOT/maven-metadata.xml  [INFO] Downloading: https://repo.maven.apache.org/maven2/com/mycompany/app/dma-parent/0.0.1-SNAPSHOT/maven-metadata.xml  [INFO] Downloading: https://repo.maven.apache.org/maven2/com/mycompany/app/dma-parent/0.0.1-SNAPSHOT/dma-parent-0.0.1-SNAPSHOT.pom |

### 6.7maven镜像配置

1）设置maven镜像

|  |
| --- |
| <!-- 配置maven镜像 -->  <mirror>  <id>mirrorId</id>  <!-- \* 表示所有的仓库都要经过如下的url -->  <mirrorOf>\*</mirrorOf>  <name>Human Readable Name for this Mirror.</name>  <url>http://127.0.0.1:8081/nexus/content/groups/public/</url>  </mirror> |

2）设置中央仓库的release如果为false，则无法从中央仓库下载release版本依赖，设置为true则可以下载成功。

|  |
| --- |
| <!-- 添加一个profile，指定id为：central-profile，并配置在pom中配置的私服仓库 -->  <profile>  <id>central-profile</id>  <!-- 配置jar包的私服仓库 -->  <repositories>  <repository>  <id>central</id>  <name>central\_repository</name>  <!-- 中央仓库配置的文件maven-model-builder-3.3.9-sources.jar\org\apache\maven\model\pom-4.0.0.xml -->  <url>https://\*</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </repository>  </repositories>  <!-- 配置插件的私服仓库 -->  <pluginRepositories>  <pluginRepository>  <id>central</id>  <name>central\_repository</name>  <!-- 中央仓库配置的文件maven-model-builder-3.3.9-sources.jar\org\apache\maven\model\pom-4.0.0.xml -->  <url>https://\*</url>  <releases>  <enabled>true</enabled>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </pluginRepository>  </pluginRepositories>  </profile> |

删除maven仓库中的依赖，关闭nexus，并删除nexus的snapshot缓存G:\eclipse\nexus-2.3.1-01-bundle\sonatype-work\nexus

3）重新测试maven项目，这时，首先从私服仓库组下载依赖，然后从中央仓库下载配置

|  |
| --- |
| .....  [INFO] Downloaded: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/plugin-tools/maven-plugin-annotations/3.1/maven-plugin-annotations-3.1.jar (0 B at 0.0 KB/sec)  [INFO] Surefire report directory: G:\eclipse\workspace\dma-commons\target\surefire-reports  [INFO] Downloading: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.pom  [INFO] Downloaded: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.pom (0 B at 0.0 KB/sec)  [INFO] Downloading: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-providers/2.12.4/surefire-providers-2.12.4.pom  [INFO] Downloaded: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-providers/2.12.4/surefire-providers-2.12.4.pom (0 B at 0.0 KB/sec)  [INFO] Downloading: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.jar  [INFO] Downloaded: http://127.0.0.1:8081/nexus/content/groups/public/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.jar (0 B at 0.0 KB/sec)  -------------------------------------------------------  T E S T S |

### 6.8项目发布到Nexus

1）在pom.xml中添加发布的仓库位置

|  |
| --- |
| <!-- 项目发布到Nexus私服 -->  <distributionManagement>  <repository>  <id>app\_release</id>  <name>app\_release\_name</name>  <url>http://127.0.0.1:8081/nexus/content/repositories/releases/</url>  </repository>    <snapshotRepository>  <id>app\_snapshot</id>  <name>app\_snapshot\_name</name>  <url>http://127.0.0.1:8081/nexus/content/repositories/snapshots/</url>  </snapshotRepository>    </distributionManagement> |

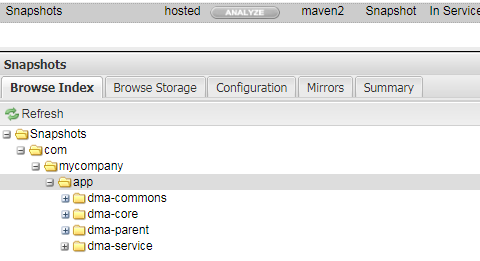
2）给maven授予发布到指定仓库的用户权限，在settins.xml中增加登录发布仓库的用户，该用户需要具备发布的权限

|  |
| --- |
| <!-- 授权将项目发布到id指定的仓库 -->  <server>  <id>app\_release</id>  <username>deployment</username>  <password>deployment123</password>  </server>    <server>  <id>app\_snapshot</id>  <username>deployment</username>  <password>deployment123</password>  </server> |

3）发布日志

|  |
| --- |
| [INFO] Uploading: http://127.0.0.1:8081/nexus/content/repositories/snapshots/com/mycompany/app/dma-service/0.0.1-SNAPSHOT/maven-metadata.xml  [INFO] Uploaded: http://127.0.0.1:8081/nexus/content/repositories/snapshots/com/mycompany/app/dma-service/0.0.1-SNAPSHOT/maven-metadata.xml (778 B at 13.1 KB/sec)  [INFO] Uploading: http://127.0.0.1:8081/nexus/content/repositories/snapshots/com/mycompany/app/dma-service/maven-metadata.xml  [INFO] Uploaded: http://127.0.0.1:8081/nexus/content/repositories/snapshots/com/mycompany/app/dma-service/maven-metadata.xml (288 B at 6.4 KB/sec)  [INFO] ------------------------------------------------------------------------  [INFO] Reactor Summary:  [INFO]  [INFO] dma-parent ......................................... SUCCESS [ 1.427 s]  [INFO] dma-commons ........................................ SUCCESS [ 2.183 s]  [INFO] dma-core ........................................... SUCCESS [ 1.920 s]  [INFO] dma-service ........................................ SUCCESS [ 1.776 s]  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 7.438 s  [INFO] Finished at: 2018-12-23T17:13:58+08:00  [INFO] Final Memory: 13M/159M  [INFO] ------------------------------------------------------------------------ |

4）查看仓库的索引和缓存



5）使用自定义用户，发布到自定义仓库

|  |
| --- |
| <!-- 项目发布到Nexus私服 -->  <distributionManagement>  <repository>  <id>app\_release</id>  <name>app\_release\_name</name>  <url>http://localhost:8081/nexus/content/repositories/byfRepoId/</url>  </repository>    <!-- <snapshotRepository>  <id>app\_snapshot</id>  <name>app\_snapshot\_name</name>  <url>http://127.0.0.1:8081/nexus/content/repositories/snapshots/</url>  </snapshotRepository> -->    </distributionManagement> |

|  |
| --- |
| <!-- 授权将项目发布到id指定的仓库 -->  <server>  <id>app\_release</id>  <username>byfId</username>  <password>byf123</password>  </server>    <server>  <id>app\_snapshot</id>  <username>byfId</username>  <password>byf123</password>  </server> |

6）仓库

## 第7节 JUnit用例覆盖率插件

### 7.1插件Surefire

By default, the Surefire Plugin will automatically include all test classes with the following wildcard patterns:

"\*\*/Test\*.java" - includes all of its subdirectories and all Java filenames that start with "Test".

"\*\*/\*Test.java" - includes all of its subdirectories and all Java filenames that end with "Test".

"\*\*/\*Tests.java" - includes all of its subdirectories and all Java filenames that end with "Tests".

"\*\*/\*TestCase.java" - includes all of its subdirectories and all Java filenames that end with "TestCase".

1）使用插件，测试按正则表达式匹配，或排除的单元测试

|  |
| --- |
| <project>  [...]  <build>  <plugins>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-surefire-plugin</artifactId>  <version>2.20.1</version>  <configuration>  <!-- <includes>  <include>\*\*/UserDao\*.java</include>  </includes> -->  <excludes>  <exclude>UserDao\*</exclude>  </excludes>  </configuration>  </plugin>  </plugins>  </build>  [...]  </project> |

2）测试用例覆盖率

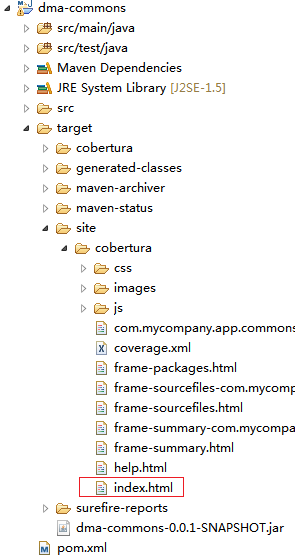
<https://www.cnblogs.com/qyf404/archive/2015/12/12/5040593.html>

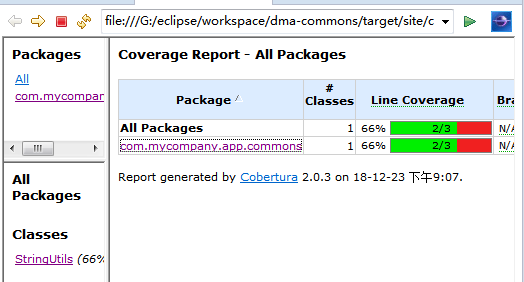
|  |
| --- |
| <plugin>  <groupId>org.codehaus.mojo</groupId>  <artifactId>cobertura-maven-plugin</artifactId>  <version>2.6</version>  <configuration>  <formats>  <format>xml</format>  <format>html</format>  </formats>  </configuration>  <executions>  <execution>  <id>cobertura-report</id>  <phase>test</phase>  <goals>  <goal>cobertura</goal>  </goals>  </execution>  </executions>  </plugin> |

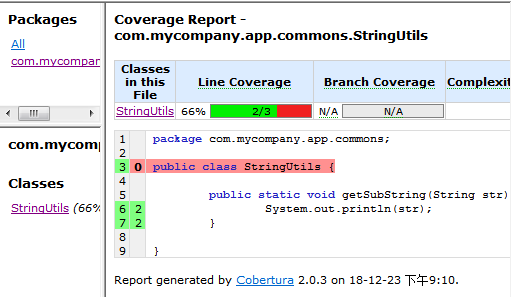
Maven-010-maven 编译报错：Failure to ... in ... was cached in the local repository, resolution will not be reattempted until the update interval of nexus has elapsed or updates are forced.

构件（org.apache.maven.plugins:maven-shade-plugin:jar:2.4.3）已经被占用，无法进行操作，进入本地仓库对应目录，将其对应版本文件夹删除，重新执行打包命令，成功编译打包

|  |
| --- |
| [INFO] Cobertura Report generation was successful.  [INFO] ------------------------------------------------------------------------  [INFO] Reactor Summary:  [INFO]  [INFO] dma-parent ......................................... SUCCESS [ 0.776 s]  [INFO] dma-commons ........................................ SUCCESS [ 4.572 s]  [INFO] dma-core ........................................... SUCCESS [ 4.523 s]  [INFO] dma-service ........................................ SUCCESS [ 4.345 s]  [INFO] ------------------------------------------------------------------------  [INFO] BUILD SUCCESS  [INFO] ------------------------------------------------------------------------  [INFO] Total time: 14.712 s  [INFO] Finished at: 2018-12-23T21:10:31+08:00  [INFO] Final Memory: 19M/282M  [INFO] ------------------------------------------------------------------------ |







3）测试覆盖率检测cobertura:check

|  |
| --- |
| <plugin>  <groupId>org.codehaus.mojo</groupId>  <artifactId>cobertura-maven-plugin</artifactId>  <version>2.7</version>  <configuration>  <formats>  <format>xml</format>  <format>html</format>  </formats>  <check>  <!--每个类的分支覆盖率 -->  <branchRate>50</branchRate>  <!--每个类的代码行覆盖率 -->  <lineRate>50</lineRate>  <!--校验失败则中止.即为true时如果校验失败则打包执行失败,反之则只是打出告警日志 -->  <haltOnFailure>true</haltOnFailure>  <!--整体的分支覆盖率 -->  <totalBranchRate>50</totalBranchRate>  <!--整体的代码行覆盖率 -->  <totalLineRate>50</totalLineRate>  <!--每个包的分支行覆盖率 -->  <packageBranchRate>50</packageBranchRate>  <!--每个包的代码行覆盖率 -->  <packageLineRate>50</packageLineRate>  <regexes>  <!--通过表达式设定覆盖率伐值 -->  <regex>  <pattern>com.mycompany.app.my\_app.\*.java</pattern>  <branchRate>90</branchRate>  <lineRate>80</lineRate>  </regex>  </regexes>  </check>  </configuration>  </plugin> |

|  |
| --- |
| [INFO]  [INFO] <<< cobertura-maven-plugin:2.7:check (default-cli) < [cobertura]test @ my-app <<<  [INFO]  [INFO] --- cobertura-maven-plugin:2.7:check (default-cli) @ my-app ---  [INFO] Cobertura 2.1.1 - GNU GPL License (NO WARRANTY) - See COPYRIGHT file  [INFO] Cobertura: Loaded information on 1 classes.  [ERROR] com.mycompany.app.my\_app.App failed coverage check. Line coverage rate of 0.0% is below 50.0%  [ERROR] com.mycompany.app.my\_app failed coverage check. Line coverage rate of 0.0% is below 50.0%  [ERROR] project failed coverage check. Line coverage rate of 0.0% is below 50.0%  [INFO] ------------------------------------------------------------------------  [INFO] BUILD FAILURE |

