

咕泡学院VIP课程

# 工程化专题之Maven

咕泡学院 James老师

QQ 2904270631

课程安排

1. 认识Maven
2. 优势
   1. 约定优于配置
   2. 简单
   3. 测试支持
   4. 构建简单
   5. CI
   6. 插件丰富
3. 下载
   1. <https://maven.apache.org/download.cgi>
   2. 安装
   3. maven-model-builder-3.3.9.jar/org/apache/maven/model
   4. 配置 MVM\_HOME
      1. Windows path
      2. Linux .bash\_profile
      3. MAVEN\_OPTS
      4. 配置setting.xml

<mirror>

<id>alimaven</id>

<name>aliyun maven</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

</mirror>

<mirror>

<id>ui</id>

<mirrorOf>central</mirrorOf>

<name>Human Readable Name for this Mirror.</name>

<url>http://uk.maven.org/maven2/</url>

</mirror>

<mirror>

<id>osc</id>

<mirrorOf>central</mirrorOf>

<url>http://maven.oschina.net/content/groups/public/</url>

</mirror>

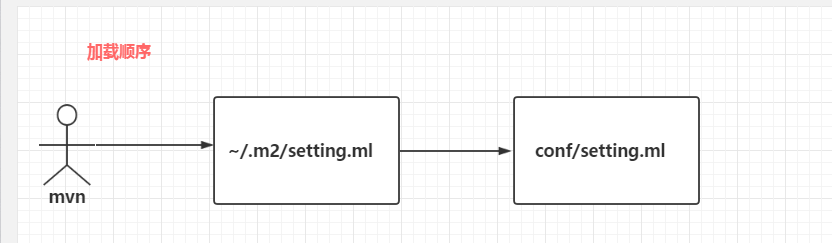
<mirror>

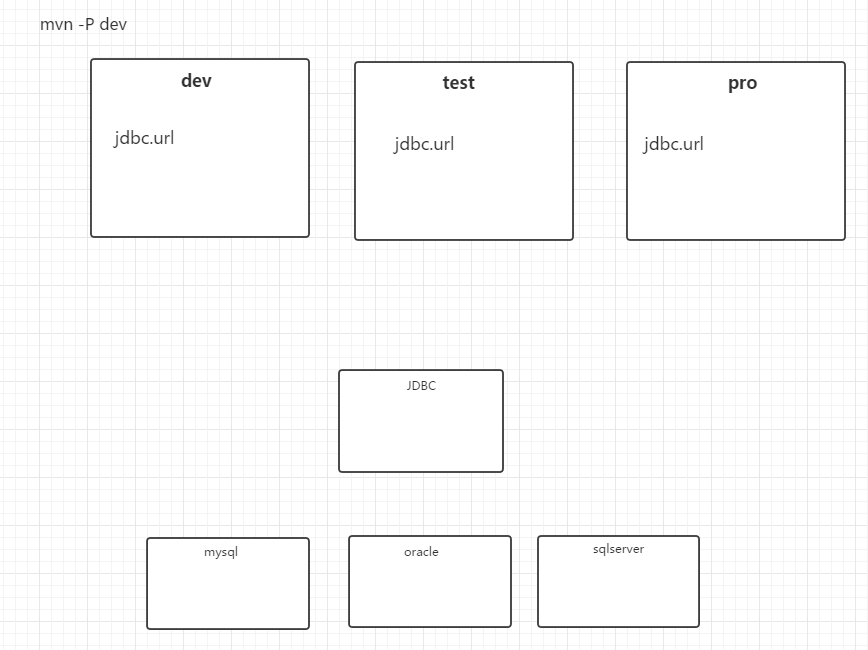
<id>osc\_thirdparty</id>

<mirrorOf>thirdparty</mirrorOf>

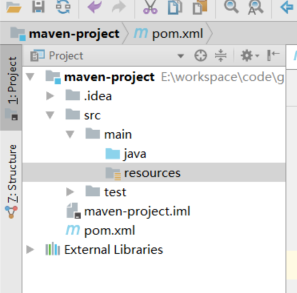
<url>http://maven.oschina.net/content/repositories/thirdparty/</url>

</mirror>





1. 新建一个Maven项目
   1. 项目结构



* 1. pom.xml
     1. groupId com.gupaoedu
     2. artfactId 功能命名
     3. version 版本号
     4. packaging 打包方式 默认是jar
     5. dependencyManagement
        1. 只能出现在父pom
        2. 统一版本号
        3. 声明 (子POM里用到再引)
     6. Dependency
        1. Type 默认jar
        2. scope
           1. compile 编译 例如spring-core
           2. test 测试
           3. provided编译 例如 servlet
           4. runtime运行时 例如JDBC驱动实现
           5. system 本地一些jar 例如短信jar
           6. 依赖传递

第一列表示直接依赖的scope，第一行表示间接依赖的scope

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | compile | test | provided | runtime |
| compile | compile | - | - | runtime |
| test | test | - | - | test |
| provided | provided | - | provided | provided |
| runtime | runtime | - | - | runtime |

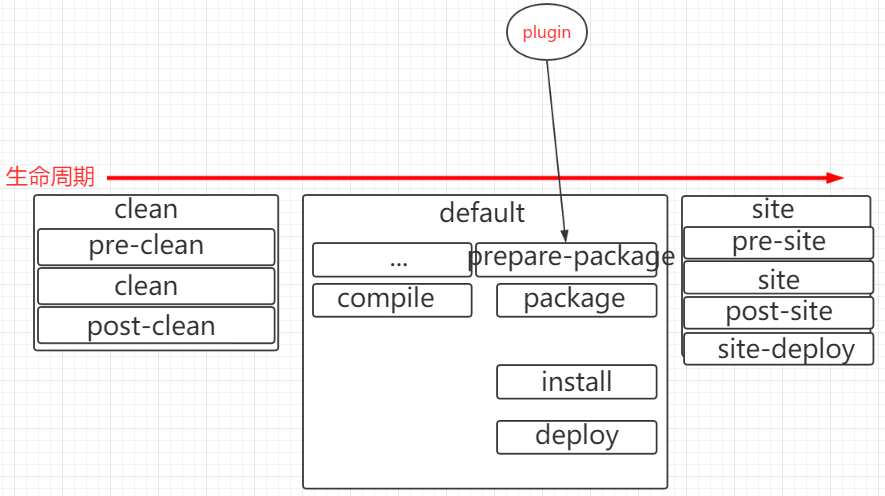
* + - * 1. 依赖仲裁

最短路径原则

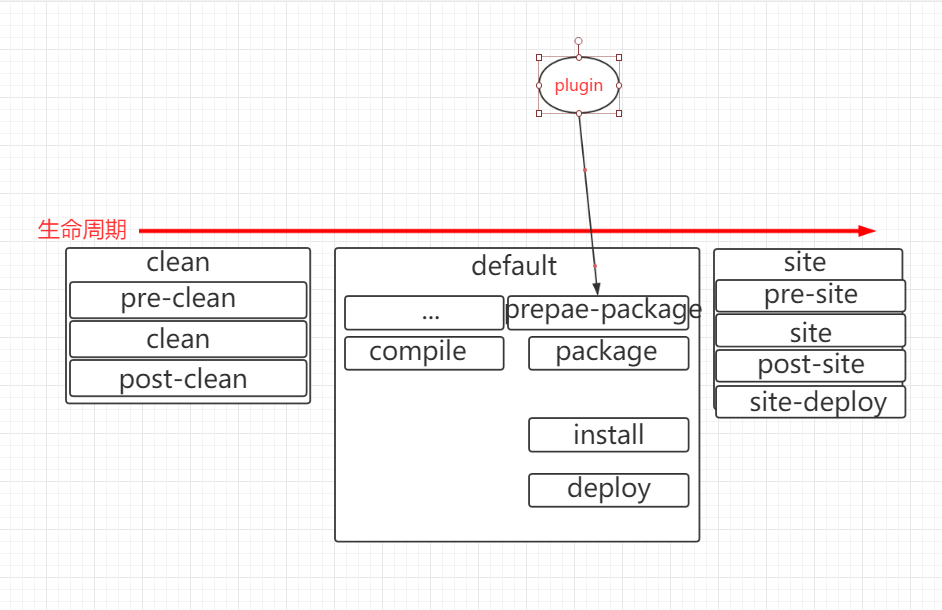
加载先后原则

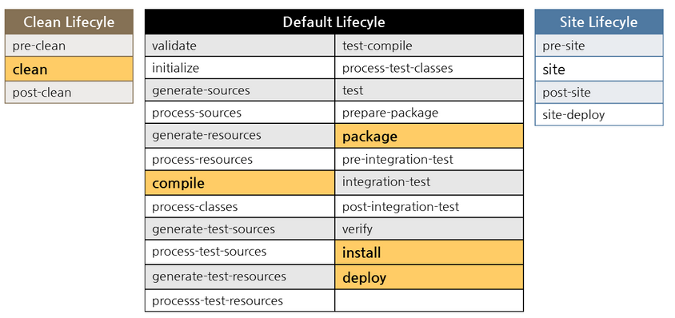
* + - * 1. exclusions

排除包



* + 1. 生命周期 lifecycle/phase/goal
       1. A Build Lifecycle is Made Up of Phases
       2. A Build Phase is Made Up of Plugin Goals





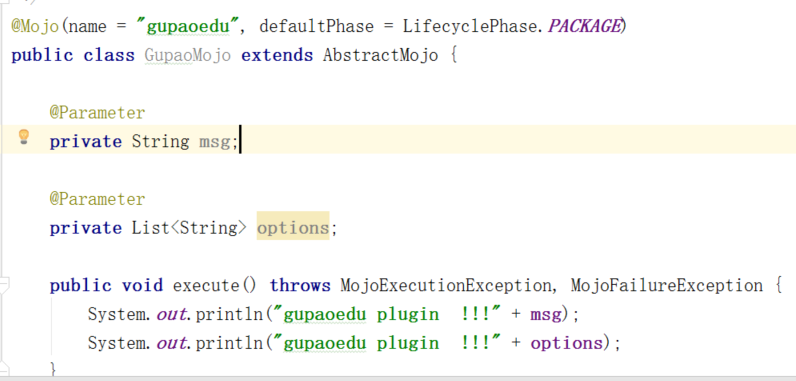
1. 版本管理
   1. 1.0-SNAPSHOT
      1. repository 删除
      2. mvn clean package -U (强制拉一次)
   2. 主版本号.次版本号.增量版本号-<里程碑版本>

1.0.0-RELAESE

1. 常用命令
   1. compile
   2. clean 删除target/
   3. test test case junit/testNG
   4. package 打包
   5. install 把项目install到local repo
   6. deploy 发本地jar发布到remote
2. 插件
   1. 常用插件
      1. <https://maven.apache.org/plugins/>
      2. <http://www.mojohaus.org/plugins.html>
      3. findbugs 静态代码检查

<!-- findbugs插件 ：静态检查代码的错误-->  
<plugin>  
 <groupId>org.codehaus.mojo</groupId>  
 <artifactId>findbugs-maven-plugin</artifactId>  
 <version>3.0.4</version>  
 <configuration>  
 <!-- 设置分析工作的等级，可以为Min、Default和Max -->  
 <effort>Low</effort>  
 <!-- Low、Medium和High (Low最严格) -->  
 <threshold>Medium</threshold>  
 <failOnError>true</failOnError>  
 <includeTests>true</includeTests>  
 <!--findbugs需要忽略的错误的配置文件-->  
 <!-- <excludeFilterFile>compile.bat</excludeFilterFile> -->  
 </configuration>  
 <executions>  
 <execution>  
 <id>run-findbugs</id>  
 <!-- 在install 阶段触发执行findbugs检查，比如执行 mvn clean install，就会执行findbugs:check命令-->  
 <phase>install</phase>  
 <goals>  
 <goal>check</goal> <!--指定执行check命令-->  
 </goals>  
 </execution>  
 </executions>  
</plugin>

* + 1. versions 统一升级版本号
       1. mvn versions:set -DnewVersion=1.1
    2. source 打包源代码
    3. assembly 打包zip、war
    4. tomcat7

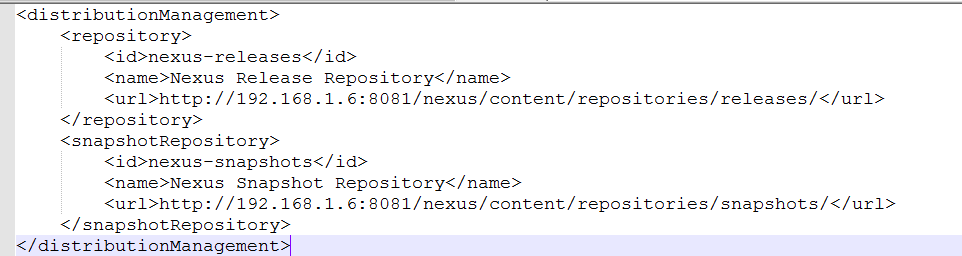
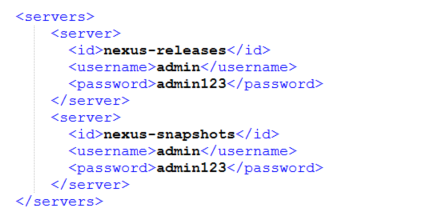
1. 自定义插件 <https://maven.apache.org/guides/plugin/guide-java-plugin-development.html>
   1. <packaging>maven-plugin</packaging>
   2. extends AbstractMojo
   3. 
   4. mvn install
   5. 参数传递
2. Profile
   1. 使用场景 dev/test/pro

<build>  
 <resources>  
 <resource>  
 <directory>${basedir}/src/main/resources</directory>  
 <excludes>  
 <exclude>conf/\*\*</exclude>  
 </excludes>  
 </resource>  
 <resource>  
 <!--保留文件-->  
 <directory>${basedir}/src/main/resources/conf/${profiles.active}</directory>  
 </resource>  
 </resources>  
</build>  
<!--按不同的环境打包-->  
<profiles>  
 <!--dev-->  
 <profile>  
 <id>dev</id>  
 <properties>  
 <!--定义变量在bulid中使用-->  
 <profiles.active>dev</profiles.active>  
 </properties>  
 <activation>  
 <!--默认打包的环境-->  
 <activeByDefault>true</activeByDefault>  
 </activation>  
 </profile>  
 <!--test-->  
 <profile>  
 <id>test</id>  
 <properties>  
 <profiles.active>test</profiles.active>  
 </properties>  
 </profile>  
 <!--prd-->  
 <profile>  
 <id>prd</id>  
 <properties>  
 <profiles.active>prd</profiles.active>  
 </properties>  
 </profile>  
</profiles>

打包测试环境 执行命令:

mvn clean package -P test

* 1. setting.xml 家和公司两套

1. 仓库
   1. 下载
   2. 安装 解压
   3. 使用<http://books.sonatype.com/nexus-book/reference3/index.html>
      1. <http://192.168.1.6:8081/nexus>
      2. admin/admin123
   4. 发布
      1. pom.xml 配置
         1. 
         2. 
   5. 下载jar配置
      1. 配置mirror
      2. Profile
2. archetype 模版化
   1. 生成一个archetype
      1. mvn archetype:create-from-project
      2. cd /target/generated-sources/archetype
      3. mvn install
   2. 从archetype创建项目 mvn archetype:generate -DarchetypeCatalog=local