Maven

@M了个J
李明杰

https://github.com/CoderMJLee

https://space.bilibili.com/325538782

小码哥教育 SEEMYGO

实力IT教育 www.520it.com

Tomcat部署项目的方式

① 将web项目整个文件夹，放在%TOMCAT_HOME%/webapps目录中，文件夹名作为ContextPath

② 将web项目打包成war，放在%TOMCAT_HOME%/webapps目录中，war文件名作为ContextPath

③ 在%TOMCAT_HOME%/conf/server.xml的Host标签中添加以下内容（ContextPath是path属性值）

- <Context docBase="项目路径" path="/xxx" />

④ 在%TOMCAT_HOME%/conf/Catalina/localhost中新建一个xml文件，xml文件名作为ContextPath

- <Context docBase="项目路径" />

小码哥教育 SEEMYGO

传统开发中的常见痛点

- jar包的下载、升级、依赖、冲突
- 不同IDE之间的项目共享
- 单元测试
- 打包发布

传统项目 vs Maven项目

Maven

- Apache Maven，主要用于自动化构建和管理Java项目
  - 基于项目对象模型（POM，Project Object Model）的概念
  - 下载地址：https://maven.apache.org/download.cgi

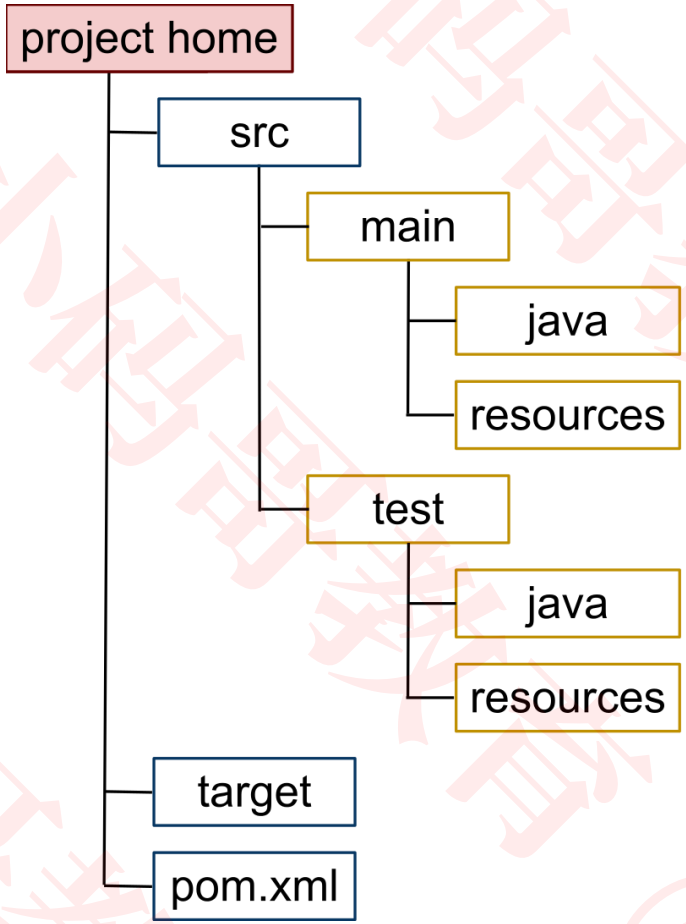
- 必须配置好JAVA_HOME，Maven对JDK版本的要求：http://maven.apache.org/docs/history.html

- 添加MAVEN_HOME\bin到PATH中

```
C:\Users\MJ>mvn -v
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: F:\Dev\Java\apache-maven-3.6.3\bin\..
Java version: 14.0.1, vendor: Oracle Corporation, runtime: F:\Dev\Java\jdk-14.0.1
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

Maven项目的常见目录

- Maven使用“约定优于配置”的思想（Convention over Configuration）
- 创建Maven项目时，Maven将创建默认项目结构，开发人员只需要相应地放置文件



类型	目录
源代码	${basedir}/src/main/java
资源	${basedir}/src/main/resources
测试源代码	${basedir}/src/test/java
测试资源	${basedir}/src/test/resources
jar、war等	${basedir}/target
字节码	${basedir}/target/classes
测试字节码	${basedir}/target/test-classes
web项目资源	${basedir}/src/main/webapp

pom.xml

- pom.xml是Maven项目的核心配置文件，根元素是project。project的常用子元素如下表所示

元素	描述
modelVersion	pom的版本，目前都使用4.0.0，是必要元素
groupId	组织名称，一般用域名的倒写，比如com.mj.hello
artifactId	项目名称
version	项目的版本号，比如1.0.0、3.0、1.0-SNAPSHOT、1.0-RELEASE
packaging	打包方式，比如pom、jar（默认值）、maven-plugin、ejb、war、ear、rar
properties	属性信息，比如文本编码等
dependencies	依赖信息
build	构建信息，比如插件配置等

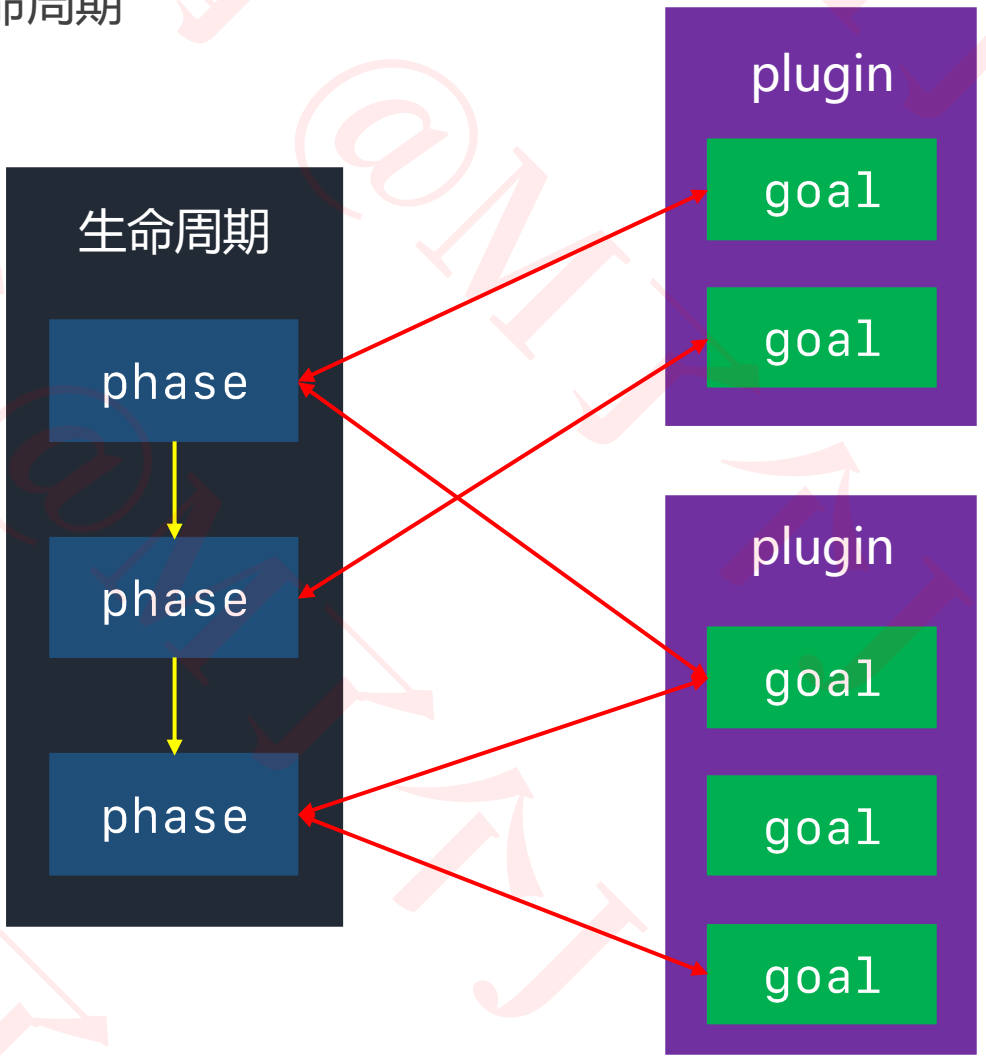
- groupId、artifactId、version，组成一个Maven坐标（Coordinate）
  - 能够确定唯一的一个项目

构建生命周期（Build Lifecycle）

- 构建生命周期，描述了构建的过程。Maven内置了3个构建生命周期
  - clean（清理）
  - default（默认，重点关注）
  - site（站点）

- 构建生命周期由phase（阶段）组成
  - phase可以跟plugin goal（插件目标）绑定
  - plugin goal代表1个特定的任务
  - 一旦某个phase被执行，就会执行其绑定的所有goal

- 通过命令mvn 插件:help可以查看插件包含的所有goal
  - 比如mvn archetype:help



default生命周期

- default生命周期由下表中的phase组成

phase	描述
validate	确认项目正确并且所有必要的信息均可用
compile	编译项目的源代码（src/main）
test	使用合适的单元测试框架测试编译后的源代码（src/main、src/test）
package	获取编译后的代码，并将其打包为可分发的格式，例如jar
verify	对集成测试的结果进行任何检查，以确保符合质量标准
install	将软件包安装到本地存储库中，以作为本地其他项目中的依赖项
deploy	在构建环境中完成后，将最终软件包复制到远程存储库中，以便与其他开发人员和项目共享

- 使用命令mvn package就会按顺序执行validate、compile、test、package阶段

小码哥教育 SEEMYGO

常用命令

命令	描述
mvn archetype:generate	创建Maven项目
mvn clean	清除target
mvn clean package	先执行clean，再执行package

小码哥教育 SEEMYGO

仓库查询

- https://search.maven.org/
- https://mvnrepository.com/

dependency中scope的取值

- compile：默认。编译依赖关系在所有类路径中均可用。此外，这些依赖项会传递到相关项目
- provided：仅在编译和测试类路径上可用，并且不可传递。希望JDK或容器在运行时提供它
- runtime：依赖关系不是编译所必需的，而是运行所必需的。它在运行时和测试类路径中，但不在编译类路径中
- test：依赖关系对于正常使用该应用程序不是必需的，并且仅在测试编译和执行阶段可用。它不是可传递的
- system：必须显式提供jar的位置（可以通过systemPath标签指定），不会去Maven仓库中查找