

Maven 完整学习手册（完整版）

第一章：Maven 基础概念

1. Maven 是什么？

Maven 是一个基于 Java 的项目管理与构建工具。核心作用包括：

- 依赖管理
- 构建生命周期管理
- 标准化目录结构
- 插件机制
- 多模块项目支持

2. 为什么使用 Maven？

- 自动下载依赖，避免手动管理 jar 包
- 统一构建流程，减少重复工作
- 标准化目录结构，降低团队成员上手成本
- 配合私有仓库可提升企业级开发效率

第二章：安装与环境配置

1. 安装步骤：

- 下载 Maven
- 解压到任意目录
- 配置 M2_HOME、MAVEN_HOME

- 设置 PATH 添加 bin

2. 检查安装是否成功：

`mvn -v`

3. 镜像加速（推荐使用阿里云镜像）：

编辑 settings.xml 增加 mirror 节点。

第三章：标准目录结构（约定优于配置）

`src/main/java`

`src/main/resources`

`src/test/java`

`pom.xml`

第四章：POM 文件详解

POM（Project Object Model）定义项目的基础信息：

1. 基本标签：

`groupId`

`artifactId`

`version`

`packaging`

2. 配置依赖：

标签中定义第三方依赖。

3. 构建设置（ build 标签 ）：

可自定义插件、资源目录、过滤器等。

第五章：依赖管理机制

1. 依赖传递

间接依赖自动引入。

2. 版本冲突原则

- 路径最短优先
- 声明顺序优先

3. 排除依赖

可使用 exclusion 标签排除不需要的传递依赖。

第六章：常用 Maven 命令

mvn clean

mvn compile

mvn test

mvn package

mvn install

mvn deploy

mvn dependency:tree

第七章：Maven 生命周期

1. clean 生命周期

执行清理操作。

2. default 生命周期（核心构建流程）

validate compile test package install deploy

3. site 生命周期

生成项目信息与文档。

第八章：插件系统

1. 编译插件（maven-compiler-plugin）

用于设置 Java 编译版本。

2. 测试插件（surefire-plugin）

负责运行测试。

3. Shade 插件

用于打包可执行 JAR。

第九章：多模块项目（Multi-Module）

父项目（packaging = pom）管理统一依赖版本。

子模块共享父 POM 配置。

第十章：私有仓库（Nexus / Maven 私服）

企业一般会使用私服存储内部依赖，提升构建速度并管理版本。

第十一章：Maven 最佳实践

- 使用 BOM 统一管理版本
 - 使用私服
 - 避免 SNAPSHOT 依赖
 - 使用 dependencyManagement 精准控制版本
-

第十二章：常见问题与解决方案

1. 依赖下载失败

检查网络或追加镜像。

2. 依赖冲突

使用 `mvn dependency:tree` 排查。

3. 打包失败

检查 JDK 版本与插件版本是否匹配。

第十三章：面试常考题

- Maven 生命周期的执行顺序？
- 什么是依赖传递？
- 如何解决依赖冲突？
- Maven 多模块如何工作？
- Maven 与 Gradle 的对比？

第十四章：附录 - Maven 常见标签参考

dependency、plugin、build、profiles、properties 等标签详解。

文档完