



Maven

Maven是一个 **Java 项目管理和构建工具**，由 Apache 提供

✓ 核心功能

1. **自动构建项目**（如编译、打包 jar/war 文件等）
 - 类似于 Python 中的 `setup.py` 或 C 的 `Makefile`
2. **依赖管理**（自动下载第三方库）
 - 你只需写清楚要用什么库、什么版本，Maven 会**自动**从远程仓库下载并管理
3. **标准化项目结构**
 - Maven 有一套默认的项目目录结构（比如 `src/main/java` 是源码目录）
4. **生命周期和插件系统**
 - Maven 提供了构建生命周期（compile → test → package → install → deploy）
 - 可通过插件实现单元测试、代码检查、打包部署等操作

🏗️ 项目结构（默认）

```
project-name/
├── pom.xml          # Maven 的核心配置文件
├── src/
│   ├── main/
│   │   └── java/    # 源代码
│   └── test/
│       └── java/    # 测试代码
```

📦 pom.xml 是 Maven 的配置文件

- 项目名称、版本
- 使用的依赖（比如 Spring、JUnit）
- 编译插件、构建方式
- 构建目标等


◆ 示例：

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.example</groupId>
  <artifactId>myproject</artifactId>
  <version>1.0-SNAPSHOT</version>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.13.2</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

常见命令

mvn compile	# 编译
mvn test	# 运行测试
mvn package	# 打包（生成 jar 或 war）
mvn clean	# 清理生成的文件
mvn install	# 安装到本地仓库

 总结一句话：

Maven 是 Java 世界中最常用的项目构建和依赖管理工具之一，它能自动下载依赖、规范项目结构，并统一构建流程

想象你要做一顿饭（写一个 Java 项目）

你要做一道菜（写一个 Java 程序），你需要：

1. 食材（代码）
2. 调料包（别人写好的库，比如数据库库、网络库）
3. 炊具（编译器、打包工具）

没有 Maven，你需要自己做这些事：

1. 自己去网上找每个需要的库（比如某个 JSON 解析库）
2. 手动下载.jar文件
3. 手动放到项目里
4. 还要记住哪些库用哪个版本，出错了自己排查冲突
5. 自己写脚本来编译打包、测试代码

有了 Maven，就像用了个“全自动厨房”：

你只要：

1. 写一个清单（`pom.xml`）：
 - 告诉 Maven：我要做什么菜（项目名）
 - 用什么调料（用哪些库、版本是多少）
2. 然后一键运行 Maven 命令：
 - `mvn compile`：自动编译
 - `mvn package`：自动打包成 jar
 - `mvn test`：自动运行测试
 - 所有需要的 jar 文件，它会自动帮你联网下载、放好、用对！

所以 Maven 是什么？

一句话总结：

Maven 是 Java 的“自动厨师”，你只需写清楚菜谱（pom.xml），它会帮你买菜（下载库）、做菜（编译打包）、试味（测试）

举个真实例子：

写 Java 想用 Google 的 Gson 库来处理 JSON，本来要：

- 去官网找 gson.jar
- 下载、复制到项目
- 配置 classpath

现在你只要在 `pom.xml` 里写一段：

```
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
  <version>2.10.1</version>
</dependency>
```

然后运行命令，Maven 就自动帮你做好一切