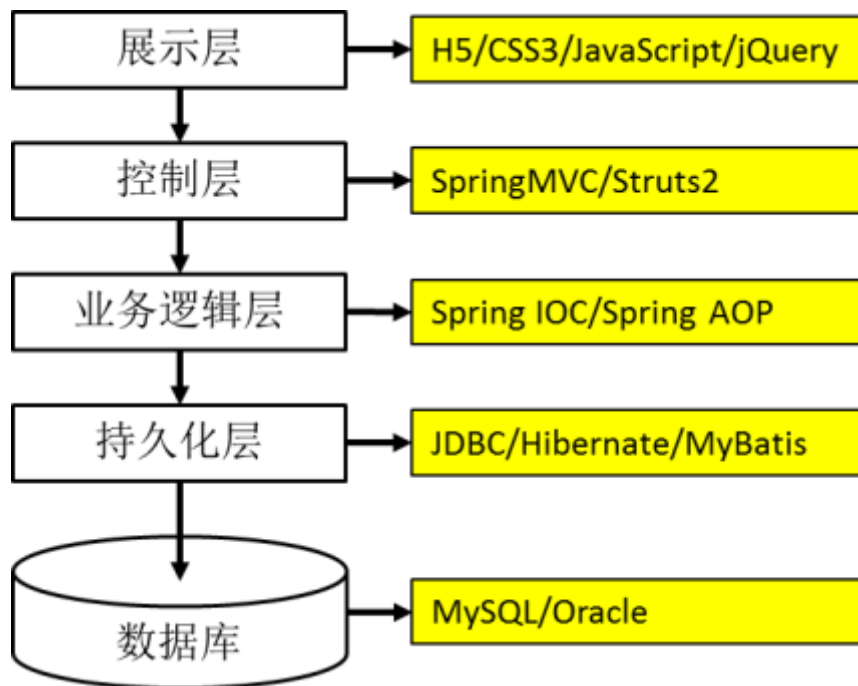


1. Maven

1.1真的需要吗？

Maven 是干什么用的？这是很多同学在刚开始接触 Maven 时最大的问题。之所以会提出这个问题，是因为即使不使用 Maven 我们仍然可以进行 B/S 结构项目的开发。从表述层、业务逻辑层到持久化层再到数据库都有成熟的解决方案——不使用 Maven 我们一样可以开发项目啊？



这里给大家纠正一个误区，Maven 并不是直接用来辅助编码的，它战斗的岗位并不是以上各层。所以我们有必要通过企业开发中的实际需求来看一看哪些方面是我们现有技术的不足。

1.2究竟为什么？

为什么要使用 Maven？它能帮助我们解决什么问题？

1. 添加第三方 jar 包

在今天的 JavaEE 开发领域，有大量的第三方框架和工具可以供我们使用。要使用这些 jar 包最简单的方法就是复制粘贴到 WEB-INF/lib 目录下。但是这会导致每次创建一个新的工程就需要将 jar 包重复复制到 lib 目录下，从而造成工作区中存在大量重复的文件，让我们的工程显得很臃肿。

而使用 Maven 后每个 jar 包本身只在本地仓库中保存一份，需要 jar 包的工程只需要以坐标的方式简单的引用一下就可以了。不仅极大的节约了存储空间，让项目更轻巧，更避免了重复文件太多而造成的混乱。

2. jar 包之间的依赖关系

























jar 包往往不是孤立存在的，很多 jar 包都需要在其他 jar 包的支持下才能够正常工作，我们称之为

jar 包之间的依赖关系。最典型的例子是：commons-fileupload-1.3.jar 依赖于 commons-io-2.0.1.jar，如果没有 IO 包，FileUpload 包就不能正常工作。

那么问题来了，你知道你所使用的所有 jar 包的依赖关系吗？当你拿到一个新的从未使用过的 jar 包，你如何得知他需要哪些 jar 包的支持呢？如果不了解这个情况，导入的 jar 包不够，那么现有的程序将不能正常工作。再进一步，当你的项目中需要用到上百个 jar 包时，你还会人为的，手工的逐一确认它们依赖的其他 jar 包吗？这简直是不可想象的。

而引入 Maven 后，Maven 就可以替我们自动的将当前 jar 包所依赖的其他所有 jar 包全部导入进来，无需人工参与，节约了我们大量的时间和精力。用实际例子来说明就是：通过 Maven 导入 commons-fileupload-1.3.jar 后，commons-io-2.0.1.jar 会被自动导入，程序员不必了解这个依赖关系。

下图是 Spring 所需 jar 包的部分依赖关系

- ▲  spring-core : 4.0.0.RELEASE [compile]
 -  commons-logging : 1.1.1 [compile]
- ▲  spring-context : 4.0.0.RELEASE [compile]
 - ▲  spring-aop : 4.0.0.RELEASE [compile]
 -  aopalliance : 1.0 [compile]
 -  spring-beans : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
 -  spring-beans : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
- ▲  spring-expression : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
- ▲  spring-jdbc : 4.0.0.RELEASE [compile]
 -  spring-beans : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
- ▲  spring-tx : 4.0.0.RELEASE [compile]
 -  aopalliance : 1.0 [compile]
 -  spring-beans : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
- ▲  spring-orm : 4.0.0.RELEASE [compile]
 -  aopalliance : 1.0 [compile]
 -  spring-beans : 4.0.0.RELEASE [compile]
 -  spring-core : 4.0.0.RELEASE [compile]
 -  spring-jdbc : 4.0.0.RELEASE [compile]
 -  spring-tx : 4.0.0.RELEASE [compile]

3. 获取第三方 jar 包

JavaEE 开发中需要使用到的 jar 包种类繁多，几乎每个 jar 包在其本身的官网上的获取方式都不尽相同。为了查找一个 jar 包找遍互联网，身心俱疲，没有经历过的人或许体会不到这种折磨。不仅如此，费劲心血找的 jar 包里有的时候并没有你需要的那个类，又或者又同名的类没有你要的方法——以不规范的方式获取的 jar 包也往往是不规范的。

使用 Maven 我们可以享受到一个完全统一规范的 jar 包管理体系。你只需要在你的项目中以坐标的方式依赖一个 jar 包，Maven 就会自动从中央仓库进行下载，并同时下载这个 jar 包所依赖的其他 jar 包

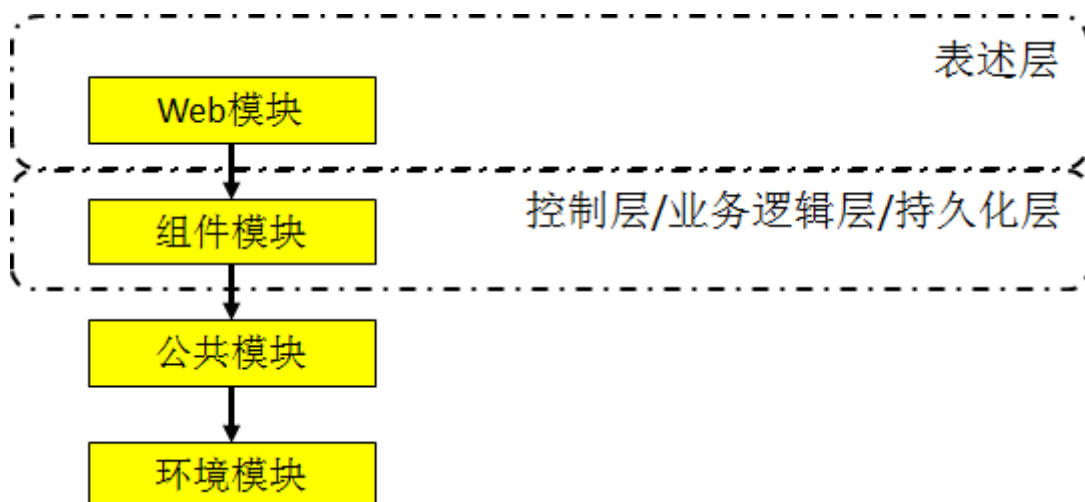
——规范、完整、准确！一次性解决所有问题！

Tips: 在这里我们顺便说一下, 统一的规范几乎可以说成是程序员的最高信仰。如果没有统一的规范, 就意味着每个具体的技术都各自为政, 需要以诸多不同的特殊的方式加入到项目中; 好不容易加入进来还会和其他技术格格不入, 最终受苦的是我们。而任何一个领域的统一规范都能够极大的降低程序员的工作难度, 减少工作量。例如: USB 接口可以外接各种设备, 如果每个设备都有自己独特的接口, 那么不仅制造商需要维护各个接口的设计方案, 使用者也需要详细了解每个设备对应的接口, 无疑是非常繁琐的。

4. 将项目拆分成多个工程模块

随着 JavaEE 项目的规模越来越庞大, 开发团队的规模也与日俱增。一个项目上千人的团队持续开发很多年对于JavaEE 项目来说再正常不过。那么我们想象一下: 几百上千的人开发的项目是同一个 Web 工程。那么架构师、项目经理该如何划分项目的模块、如何分工呢? 这么大的项目已经不可能通过package 结构来划分模块, 必须将项目拆分成多个工程协同开发。多个模块工程中有的的是 Java 工程, 有的是 Web 工程。

那么工程拆分后又如何进行互相调用和访问呢? 这就需要用到 Maven 的依赖管理机制。



上层模块依赖下层, 所以下层模块中定义的 API 都可以为上层所调用和访问。

2. Maven 是什么

2.1 Maven 简介

Maven 是Apache 软件基金会组织维护的一款自动化构建工具, 专注服务于 Java 平台的项目构建和依赖管理。Maven 这个单词的本意是: 专家, 内行。读音是['meɪv(ə)n]或['meɪn]。



2.2 什么是构建

构建并不是创建, 创建一个工程并不等于构建一个项目。要了解构建的含义我们应该由浅入深的从以下三个层面来看:

1. 纯 Java 代码

大家都知道，我们 Java 是一门编译型语言，.java 扩展名的源文件需要编译成.class 扩展名的字节码文件才能够执行。所以编写任何 Java 代码想要执行的话就必须经过编译得到对应的.class 文件。

2. Web 工程

当我们需要通过浏览器访问 Java 程序时就必须将包含 Java 程序的 Web 工程编译的结果“拿”到服务器上的指定目录下，并启动服务器才行。这个“拿”的过程我们叫部署。

我们可以将未编译的 Web 工程比喻为一只生的鸡，编译好的 Web 工程是一只煮熟的鸡，编译部署的过程就是将鸡炖熟。

3. 实际项目

在实际项目中整合第三方框架，Web 工程中除了 Java 程序和 JSP 页面、图片等静态资源之外，还包括第三方框架的 jar 包以及各种各样的配置文件。所有这些资源都必须按照正确的目录结构部署到服务器上，项目才可以运行。

所以综上所述：构建就是以我们编写的 Java 代码、框架配置文件、国际化等其他资源文件、JSP 页面和图片等静态资源作为“原材料”，去“生产”出一个可以运行的项目的过程。

那么项目构建的全过程中都包含哪些环节呢？

2.3 构建过程的几个主要环节

1. 清理：删除以前的编译结果，为重新编译做好准备。
2. 编译：将 Java 源程序编译为字节码文件。
3. 测试：针对项目中的关键点进行测试，确保项目在迭代开发过程中关键点的正确性。
4. 报告：在每一次测试后以标准的格式记录和展示测试结果。
5. 打包：将一个包含诸多文件的工程封装为一个压缩文件用于安装或部署。Java 工程对应 jar 包，Web 工程对应 war 包。
6. 安装：在 Maven 环境下特指将打包的结果——jar 包或 war 包安装到本地仓库中。
7. 部署：将打包的结果部署到远程仓库或将 war 包部署到服务器上运行。

2.4 自动化构建

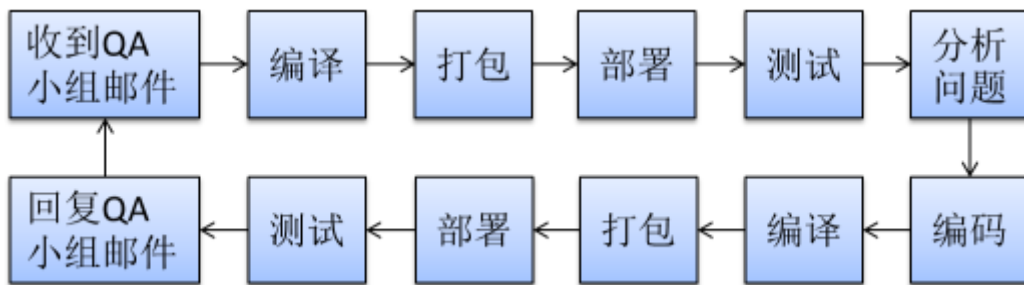
其实上述环节我们在 Eclipse 中都可以找到对应的操作，只是不太标准。那么既然 IDE 已经可以进行构建了我们为什么还要使用 Maven 这样的构建工具呢？我们来看一个小故事：

这是阳光明媚的一天。托马斯向往常一样早早的来到了公司，冲好一杯咖啡，进入了自己的邮箱——很不幸，QA 小组发来了一封邮件，报告了他昨天提交的模块的测试结果——有 BUG。“好吧，反正也不是第一次”，托马斯摇摇头，进入 IDE，运行自己的程序，编译、打包、部署到服务器上，然后按照邮件中的操作路径进行测试。“嗯，没错，这个地方确实有问题”，托马斯说道。于是托马斯开始尝试修复这个 BUG，当他差不多有眉目的时候已经到了午饭时间。

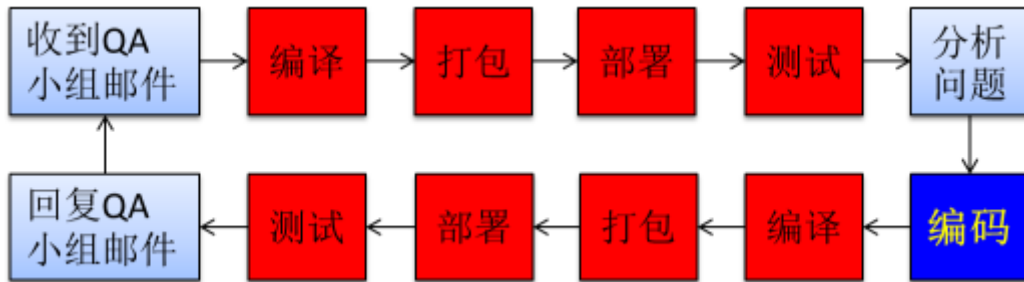
下午继续工作。BUG 很快被修正了，接着托马斯对模块重新进行了编译、打包、部署，测试之后确认没有问题了，回复了 QA 小组的邮件。

一天就这样过去了，明媚的阳光化作了美丽的晚霞，托马斯却觉得生活并不像晚霞那样美好啊。

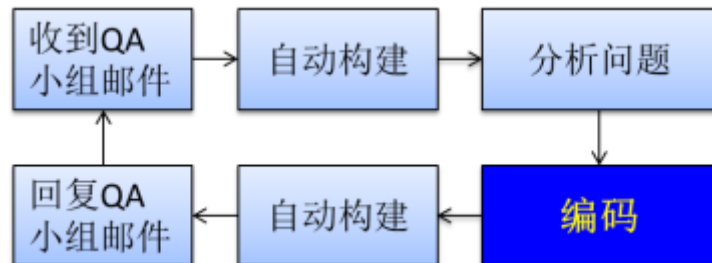
让我们来梳理一下托马斯这一天中的工作内容



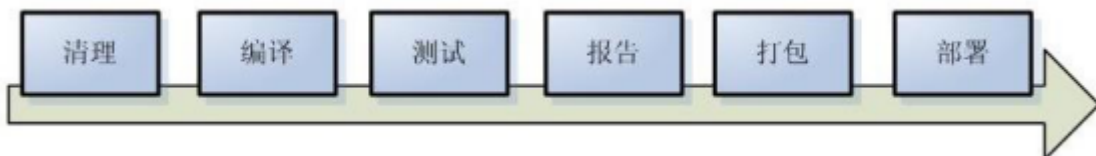
从中我们发现，托马斯的很大一部分时间花在了“编译、打包、部署、测试”这些程式化的工作上面，而真正需要由“人”的智慧实现的分析问题和编码却只占了很少一部分。



能否将这些程式化的工作交给机器自动完成呢？——当然可以！这就是自动化构建



此时 Maven 的意义就体现出来了，它可以自动的从构建过程的起点一直执行到终点



2.5 安装 Maven 核心程序

1. 检查 JAVA_HOME 环境变量

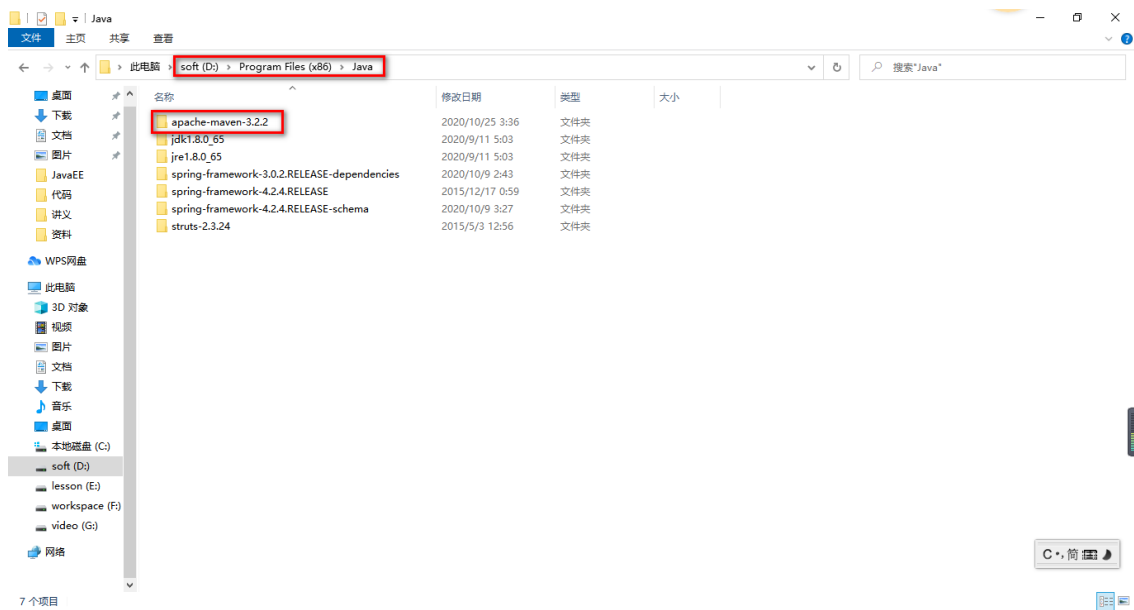
```
C:\Users\Administrator>echo %JAVA_HOME%;
D:\Program Files (x86)\Java\jdk1.8.0_65;
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.18363.1139]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>echo %JAVA_HOME%;
D:\Program Files (x86)\Java\jdk1.8.0_65;

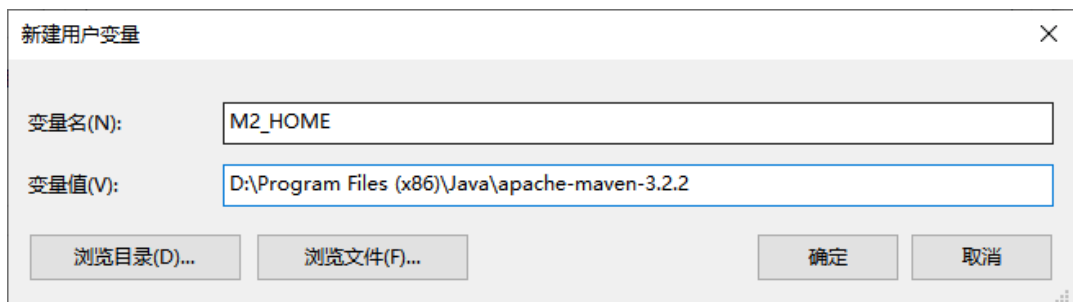
C:\Users\Administrator>
```

2. 解压 Maven 核心程序压缩包

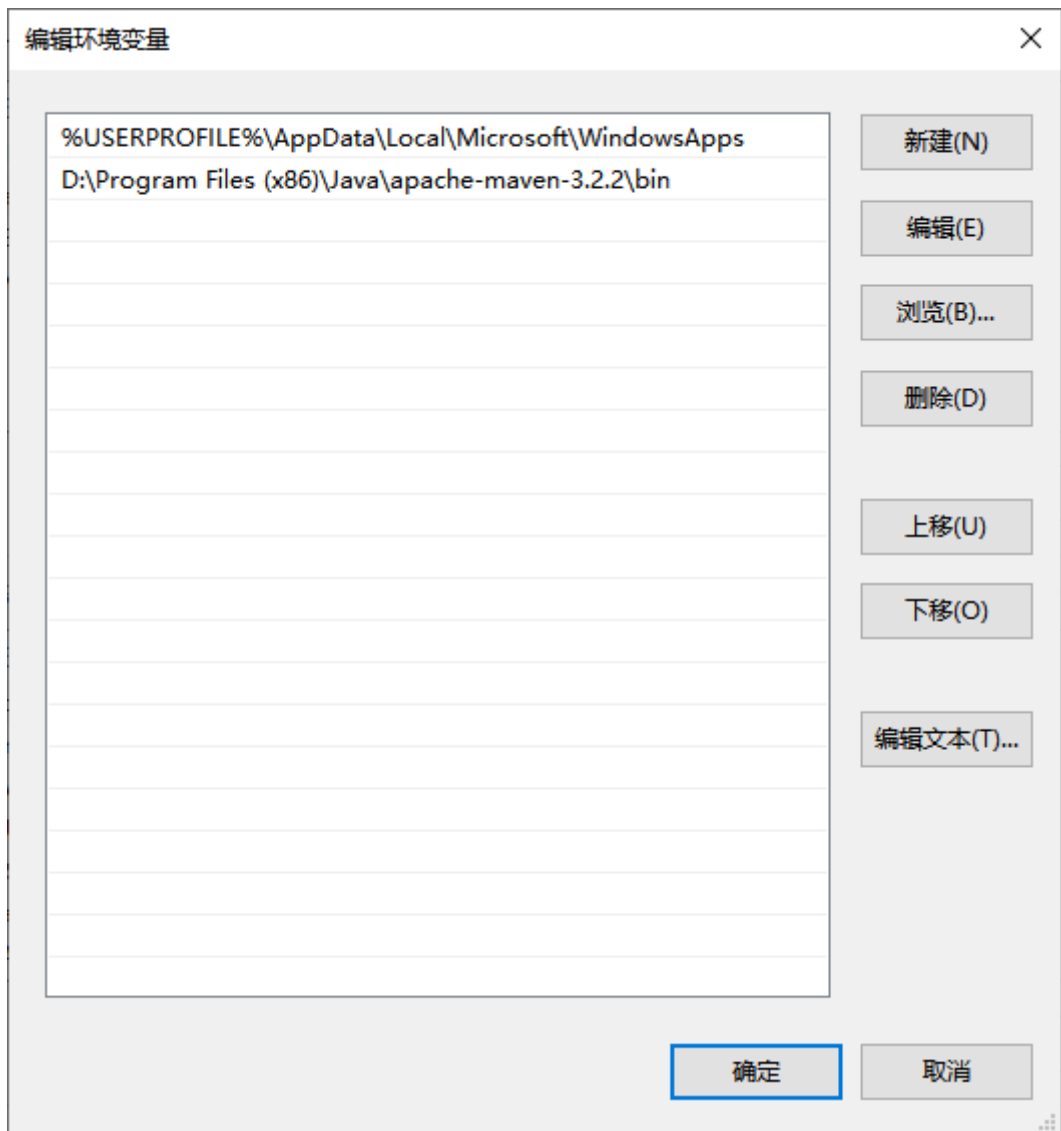


3. 配置 Maven 相关环境变量

1. 配置 MAVEN_HOME 或者 M2_HOME



2. path



3. 验证:运行 mvn -v 命令查看

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.18363.1139]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>mvn -v
Apache Maven 3.2.2 (45f7c06d68e745d05611f7fd14efb6594181933e; 2014-06-17T21:51:42+08:00)
Maven home: D:\Program Files (x86)\Java\apache-maven-3.2.2
Java version: 1.8.0_65, vendor: Oracle Corporation
Java home: D:\Program Files (x86)\Java\jdk1.8.0_65\jre
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 10", version: "10.0", arch: "amd64", family: "dos"
C:\Users\Administrator>
```

2.6 Maven 核心概念

Maven 能够实现自动化构建是和它的内部原理分不开的，这里我们从 Maven 的九个核心概念入手，看看 Maven 是如何实现自动化构建的

1. POM
2. 约定的目录结构
3. 坐标
4. 依赖管理
5. 仓库管理
6. 生命周期
7. 插件和目标
8. 继承
9. 聚合

3. 连网问题

Maven 的核心程序中仅仅定义了抽象的生命周期，而具体的操作则是由 Maven 的插件来完成的。可是

Maven 的插件并不包含在 Maven 的核心程序中，在首次使用时需要联网下载。

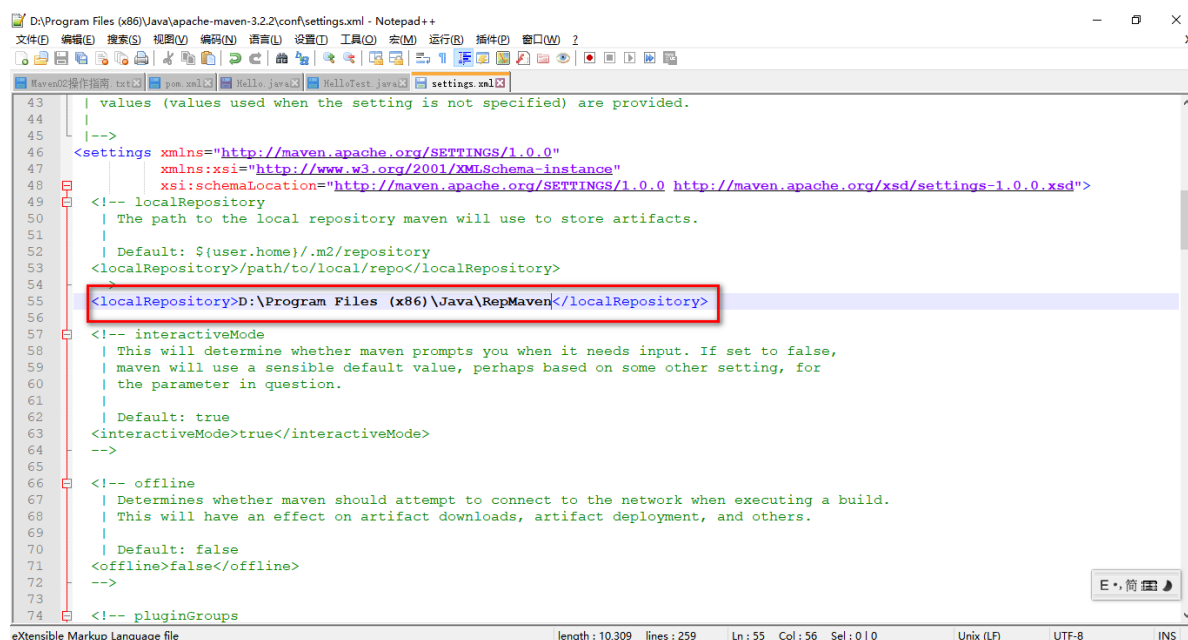
下载得到的插件会被保存到本地仓库中。本地仓库默认的位置是：~\.m2\repository。

如果不能联网可以使用我们提供的 RepMaven.zip 解压得到。具体操作参见“Maven 操作指南.txt”。

Maven 核心程序如果在本地仓库中找不到需要的插件,那么它会自动连接外网,到中央仓库下载.

如果此时无法连接外网,则构建失败

- 修改默认本地仓库位置
 - 找到 Maven 解压目录 \conf\settings.xml
 - 在 settings.xml 中找到 localRepository 标签
 - 将标签内容从注释中取出,修改为已经准备好仓库路径.



```
43 | values (values used when the setting is not specified) are provided.
44 |
45 |
46 <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
47 |         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
48 |         xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">
49 |   <!-- localRepository
50 |    | The path to the local repository maven will use to store artifacts.
51 |    |
52 |    | Default: ${user.home}/.m2/repository
53 |    |>
54 |   <localRepository>path/to/local/repo</localRepository>
55 |
56 |   <!-- interactiveMode
57 |    | This will determine whether maven prompts you when it needs input. If set to false,
58 |    | maven will use a sensible default value, perhaps based on some other setting, for
59 |    | the parameter in question.
60 |    |
61 |    | Default: true
62 |    |>
63 |   <interactiveMode>true</interactiveMode>
64 |
65 |   <!-- offline
66 |    | Determines whether maven should attempt to connect to the network when executing a build.
67 |    | This will have an effect on artifact downloads, artifact deployment, and others.
68 |    |
69 |    | Default: false
70 |    |>
71 |   <offline>false</offline>
72 |
73 |   <!-- pluginGroups
74 |    |>
```

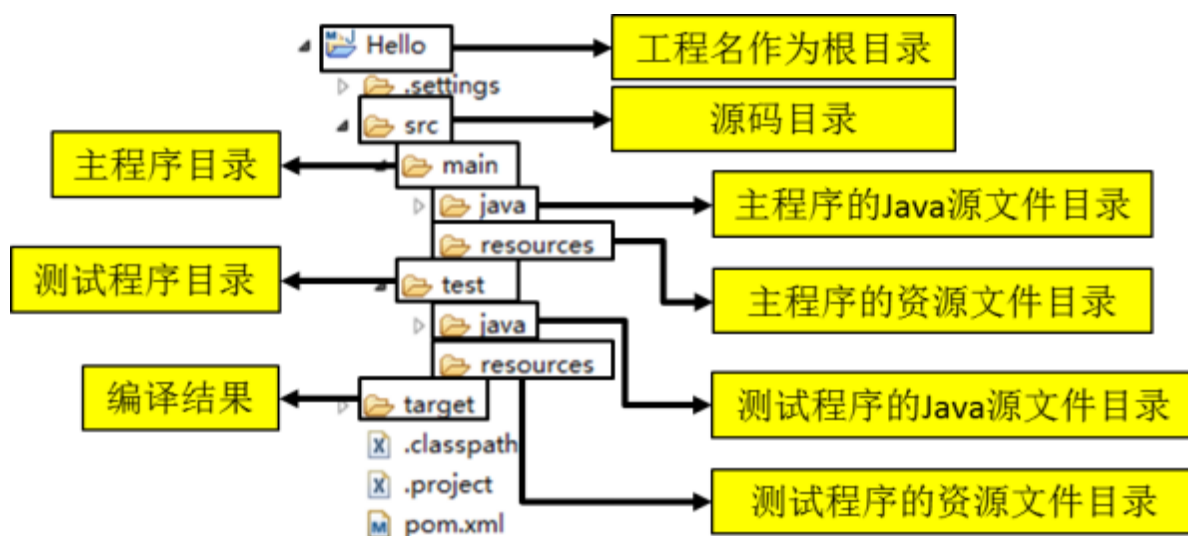

4 约定的目录结构

约定的目录结构对于 Maven 实现自动化构建而言是必不可少的一环，就拿自动编译来说，Maven 必须能找到 Java 源文件，下一步才能编译，而编译之后也必须有一个准确的位置保存编译得到的字节码文件。

我们在开发中如果需要对第三方工具或框架知道我们自己创建的资源在哪，那么基本上就是两种方式：

1. 通过配置的形式明确告诉它
2. 基于第三方工具或框架的约定

Maven 对工程目录结构的要求就属于后面的一种。



现在 JavaEE 开发领域普遍认同一个观点：约定>配置>编码。意思就是能用配置解决的问题就不编码，能基于约定的就不进行配置。而 Maven 正是因为指定了特定文件保存的目录才能够对我们的 Java 工程进行自动化构建。

5 第一个 Maven 程序

5.1 目录结构

```
Hello
|---src
|---|---main
|---|---|---java
|---|---|---resources
|---|---test
|---|---|---java
|---|---|---resources
|---pom.xml
```

5.2 POM文件内容

```
<?xml version="1.0" ?>
<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.admiral.maven</groupId>
  <artifactId>Hello</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <name>Hello</name>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.0</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

5.3 编写主程序代码

在 src/main/java/com/admiral/maven 目录下新建文件 Hello.java，内容如下

```
package com.admiral.maven;
public class Hello {
    public String sayHello(String name){
        return "Hello "+name+"!";
    }
}
```

5.4 编写测试代码

在 /src/test/java/com/admiral/maven 目录下新建测试文件 HelloTest.java

```
package com.admiral.maven;
import org.junit.Test;
import static junit.framework.Assert.*;
public class HelloTest {
    @Test
    public void testHello(){
        Hello hello = new Hello();
        String results = hello.sayHello("litingwei");
        assertEquals("Hello litingwei!",results);
    }
}
```

5.5 运行几个基本的Maven命令

mvn clean	:清理
mvn compile	:编译
mvn test-compile	:清理
mvn test	:测试
mvn package	:打包

注意：运行Maven命令时一定要进入pom.xml文件所在的目录！

```

Downloaded from central: https://repo.maven.apache.org/maven2/commons-cli/commons-cli/1.0.0/commons-cli-1.0.0.jar (30 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.0.6/maven-settings-2.0.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interactivity-api/1.0-alpha-4/plexus-interactivity-api-1.0-alpha-4.jar (13 kB at 5.0 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-descriptor/2.0.6/maven-plugin-descriptor-2.0.6.jar (37 kB at 14 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.6/maven-monitor-2.0.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1/classworlds-1.1.jar (38 kB at 14 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.0.6/maven-model-2.0.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.0-alpha-9-stable-1/plexus-container-default-1.0-alpha-9-stable-1.jar (104 kB at 47 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.6/maven-artifact-2.0.6.jar (87 kB at 31 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.0.6/maven-settings-2.0.6.jar (49 kB at 16 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/2.0.5/plexus-utils-2.0.5.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.6/maven-monitor-2.0.6.jar (10 kB at 3.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interpolation/1.13/plexus-interpolation-1.13.jar (45 kB at 10 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/2.0.5/plexus-utils-2.0.5.jar (10 kB at 3.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/maven-model/2.0.6/maven-model-2.0.6.jar (10 kB at 3.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-filtering/1.1/maven-filtering-1.1.jar (10 kB at 3.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/plexus/plexus-build-api/0.0.4/plexus-build-api-0.0.4.jar (36 kB at 25 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.jar (121 kB at 33 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interpolation/1.13/plexus-interpolation-1.13.jar (43 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-filtering/1.1/maven-filtering-1.1.jar (43 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/plexus/plexus-build-api/0.0.4/plexus-build-api-0.0.4.jar (6.8 kB at 1.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/2.0.5/plexus-utils-2.0.5.jar (223 kB at 54 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.0-alpha-9-stable-1/plexus-container-default-1.0-alpha-9-stable-1.jar (104 kB at 47 kB/s)
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ Hello ---
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.0.9/maven-plugin-api-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.0.9/maven-plugin-api-2.0.9.pom (1.5 kB at 3.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven/2.0.9/maven-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven/2.0.9/maven-2.0.9.pom (19 kB at 40 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/8/maven-parent-8.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/8/maven-parent-8.pom (24 kB at 49 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/4/apache-4.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/4/apache-4.pom (4.5 kB at 9.1 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.9/maven-artifact-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.9/maven-artifact-2.0.9.pom (1.6 kB at 3.3 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.5.1/plexus-utils-1.5.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.5.1/plexus-utils-1.5.1.pom (2.3 kB at 4.9 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.0.9/maven-core-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.0.9/maven-core-2.0.9.pom (7.8 kB at 16 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.0.9/maven-settings-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.0.9/maven-settings-2.0.9.pom (2.1 kB at 4.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.0.9/maven-model-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.0.9/maven-model-2.0.9.pom (3.1 kB at 6.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-parameter-documenter/2.0.9/maven-plugin-parameter-documenter-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-parameter-documenter/2.0.9/maven-plugin-parameter-documenter-2.0.9.pom (2.0 kB at 4.1 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-profile/2.0.9/maven-profile-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-profile/2.0.9/maven-profile-2.0.9.pom (2.0 kB at 4.3 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-repository-metadata/2.0.9/maven-repository-metadata-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-repository-metadata/2.0.9/maven-repository-metadata-2.0.9.pom (1.9 kB at 3.9 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-error-diagnostics/2.0.9/maven-error-diagnostics-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-error-diagnostics/2.0.9/maven-error-diagnostics-2.0.9.pom (1.7 kB at 3.6 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-project/2.0.9/maven-project-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-project/2.0.9/maven-project-2.0.9.pom (2.7 kB at 5.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact-manager/2.0.9/maven-artifact-manager-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact-manager/2.0.9/maven-artifact-manager-2.0.9.pom (2.7 kB at 5.6 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-registry/2.0.9/maven-plugin-registry-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-registry/2.0.9/maven-plugin-registry-2.0.9.pom (2.0 kB at 4.1 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-descriptor/2.0.9/maven-plugin-descriptor-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-descriptor/2.0.9/maven-plugin-descriptor-2.0.9.pom (2.1 kB at 4.3 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.9/maven-monitor-2.0.9.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.9/maven-monitor-2.0.9.pom (1.3 kB at 2.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-toolchain/1.0/maven-toolchain-1.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-toolchain/1.0/maven-toolchain-1.0.pom (3.4 kB at 7.1 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/0.1/maven-shared-utils-0.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/0.1/maven-shared-utils-0.1.pom (4.0 kB at 8.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-components/18/maven-shared-components-18.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-components/18/maven-shared-components-18.pom (4.9 kB at 10 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/code/findbugs/jsr305/2.0.1/jsr305-2.0.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/code/findbugs/jsr305/2.0.1/jsr305-2.0.1.pom (965 B at 1.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-incremental/1.1/maven-shared-incremental-1.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-incremental/1.1/maven-shared-incremental-1.1.pom (4.7 kB at 9.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-components/19/maven-shared-components-19.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-components/19/maven-shared-components-19.pom (6.4 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.2.1/maven-plugin-api-2.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.2.1/maven-plugin-api-2.2.1.pom (1.5 kB at 3.1 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven/2.2.1/maven-2.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven/2.2.1/maven-2.2.1.pom (22 kB at 43 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/11/maven-parent-11.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/11/maven-parent-11.pom (32 kB at 61 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/5/apache-5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/5/apache-5.pom (4.1 kB at 8.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.2.1/maven-core-2.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.2.1/maven-core-2.2.1.pom (12 kB at 24 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.2.1/maven-settings-2.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.2.1/maven-settings-2.2.1.pom (2.2 kB at 4.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.2.1/maven-model-2.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.2.1/maven-model-2.2.1.pom (3.2 kB at 7.0 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interpolation/1.11/plexus-interpolation-1.11.pom
Downloaded from central: https://
```



```
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-api/2.2/plexus-compiler-api-2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-api/2.2/plexus-compiler-api-2.2.pom (865 B at 1.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler/2.2/plexus-compiler-2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler/2.2/plexus-compiler-2.2.pom (3.6 kB at 7.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-components/1.3.1/plexus-components-1.3.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-components/1.3.1/plexus-components-1.3.1.pom (3.1 kB at 6.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus/3.3.1/plexus-3.3.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus/3.3.1/plexus-3.3.1.pom (20 kB at 42 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/spice/spice-parent/17/spice-parent-17.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/spice/spice-parent/17/spice-parent-17.pom (6.8 kB at 14 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/forge/forge-parent/10/forge-parent-10.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/forge/forge-parent/10/forge-parent-10.pom (14 kB at 28 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.8/plexus-utils-3.0.8.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.8/plexus-utils-3.0.8.pom (3.1 kB at 6.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus/3.2/plexus-3.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus/3.2/plexus-3.2.pom (19 kB at 37 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-manager/2.2/plexus-compiler-manager-2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-manager/2.2/plexus-compiler-manager-2.2.pom (690 B at 1.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/2.2/plexus-compiler-javac-2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/2.2/plexus-compiler-javac-2.2.pom (769 B at 1.6 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compilers/2.2/plexus-compilers-2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compilers/2.2/plexus-compilers-2.2.pom (1.2 kB at 2.6 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.5.5/plexus-container-default-1.5.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.5.5/plexus-container-default-1.5.5.pom (2.8 kB at 5.7 kB/s)

Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.4.5/plexus-utils-1.4.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.4.5/plexus-utils-1.4.5.pom (2.3 kB at 4.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-classworlds/2.2.2/plexus-classworlds-2.2.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-classworlds/2.2.2/plexus-classworlds-2.2.2.pom (4.0 kB at 8.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean-reflect/3.4/xbean-reflect-3.4.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean-reflect/3.4/xbean-reflect-3.4.pom (2.8 kB at 5.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean/3.4/xbean-3.4.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean/3.4/xbean-3.4.pom (19 kB at 38 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/log4j/log4j/1.2.12/log4j-1.2.12.pom
Downloaded from central: https://repo.maven.apache.org/maven2/log4j/log4j/1.2.12/log4j-1.2.12.pom (145 B at 309 B/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-logging/commons-logging-api/1.1/commons-logging-api-1.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/commons-logging/commons-logging-api/1.1/commons-logging-api-1.1.pom (5.3 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.pom (2.5 kB at 5.2 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/gollections/google-1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/gollections/google-1.pom (1.6 kB at 3.3 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.pom (747 B at 1.6 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.0.9/maven-plugin-api-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.0.9/maven-plugin-api-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.0.9/maven-core-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.5.1/plexus-utils-1.5.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.9/maven-artifact-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-api/2.0.9/maven-plugin-api-2.0.9.jar (13 kB at 27 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-parameter-documenter/2.0.9/maven-plugin-parameter-documenter-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-settings/2.0.9/maven-settings-2.0.9.jar (49 kB at 77 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-profile/2.0.9/maven-profile-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.9/maven-artifact-2.0.9.jar (89 kB at 97 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.0.9/maven-model-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-parameter-documenter/2.0.9/maven-plugin-parameter-documenter-2.0.9.jar (21 kB at 20 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-repository-metadata/2.0.9/maven-repository-metadata-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-core/2.0.9/maven-core-2.0.9.jar (160 kB at 148 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-error-diagnostics/2.0.9/maven-error-diagnostics-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/1.5.1/plexus-utils-1.5.1.jar (211 kB at 171 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-profile/2.0.9/maven-profile-2.0.9.jar (35 kB at 28 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-registry/2.0.9/maven-plugin-registry-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-project/2.0.9/maven-project-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-repository-metadata/2.0.9/maven-repository-metadata-2.0.9.jar (25 kB at 16 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-error-diagnostics/2.0.9/maven-error-diagnostics-2.0.9.jar (14 kB at 8.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-descriptor/2.0.9/maven-plugin-descriptor-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact-manager/2.0.9/maven-artifact-manager-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-model/2.0.9/maven-model-2.0.9.jar (87 kB at 53 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.9/maven-monitor-2.0.9.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-registry/2.0.9/maven-plugin-registry-2.0.9.jar (29 kB at 15 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-toolchain/1.0/maven-toolchain-1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-project/2.0.9/maven-project-2.0.9.jar (122 kB at 62 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/0.1/maven-shared-utils-0.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-monitor/2.0.9/maven-monitor-2.0.9.jar (10 kB at 4.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/code/findbugs/jsr305/2.0.1/jsr305-2.0.1.jar (32 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-plugin-descriptor/2.0.9/maven-plugin-descriptor-2.0.9.jar (37 kB at 16 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-incremental/1.1/maven-shared-incremental-1.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact-manager/2.0.9/maven-artifact-manager-2.0.9.jar (58 kB at 25 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-component-annotations/1.5.5/plexus-component-annotations-1.5.5.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-toolchain/1.0/maven-toolchain-1.0.jar (33 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-api/2.2/plexus-compiler-api-2.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/0.1/maven-shared-utils-0.1.jar (155 kB at 59 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-manager/2.2/plexus-compiler-manager-2.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-component-annotations/1.5.5/plexus-component-annotations-1.5.5.jar (4.2 kB at 1.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-incremental/1.1/maven-shared-incremental-1.1.jar (14 kB at 4.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/2.2/plexus-compiler-javac-2.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/code/findbugs/jsr305/2.0.1/jsr305-2.0.1.jar (32 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-classworlds/2.2.2/plexus-classworlds-2.2.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.5.5/plexus-container-default-1.5.5.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-api/2.2/plexus-compiler-api-2.2.jar (25 kB at 8.0 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-manager/2.2/plexus-compiler-manager-2.2.jar (4.6 kB at 1.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/log4j/log4j/1.2.12/log4j-1.2.12.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean-reflect/3.4/xbean-reflect-3.4.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/2.2/plexus-compiler-javac-2.2.jar (19 kB at 5.5 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-logging/commons-logging-api/1.1/commons-logging-api-1.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-classworlds/2.2.2/plexus-classworlds-2.2.2.jar (46 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.5.5/plexus-container-default-1.5.5.jar (217 kB at 58 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean-reflect/3.4/xbean-reflect-3.4.jar (134 kB at 34 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-logging/commons-logging-api/1.1/commons-logging-api-1.1.jar (45 kB at 11 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.jar (121 kB at 27 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/log4j/log4j/1.2.12/log4j-1.2.12.jar (358 kB at 78 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.jar (640 kB at 131 kB/s)
[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 F:\workspace_eclipse01\javaase\Hello\target\classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:21 min
[INFO] Finished at: 2020-10-25T05:19:07+08:00
[INFO] -----
F:\workspace_eclipse01\javaase\Hello>
```

5. POM

Project Object Model：项目对象模型。将 Java 工程的相关信息封装为对象作为便于操作和管理的模型。

Maven 工程的核心配置。可以说学习 Maven 就是学习 pom.xml 文件中的配置。

6. 坐标

6.1 几何中的坐标

[1] 在一个平面中使用x、y 两个向量可以唯一的确定平面中的一个点。

[2] 在空间中使用x、y、z 三个向量可以唯一的确定空间中的一个点。

6.2 Maven 的坐标

使用如下三个向量在 Maven 的仓库中唯一的确定一个 Maven 工程。

[1]groupId：公司或组织的域名倒序+当前项目名称

[2]artifactId：当前项目的模块名称

[3]version：当前模块的版本

```
<groupId>com.admiral.maven</groupId>
<artifactId>Hello</artifactId>
<version>0.0.1-SNAPSHOT</version>
```

6.3 如何通过坐标到仓库中查找 jar 包？

1. 将 gav 三个向量连起来

```
com.admiral.maven + Hello + 0.0.1-SNAPSHOT
```

2. 以连起来的字符串作为目录结构到仓库中查找

```
com/admiral/maven/Hello/0.0.1-SNAPSHOT/Hello-0.0.1-SNAPSHOT.jar
```

注意：我们自己的Maven 工程必须执行安装操作才会进入仓库。安装的命令是：mvn install

7. 依赖

Maven 中最关键的部分，我们使用 Maven 最主要的就是使用它的依赖管理功能。要理解和掌握 Maven

的依赖管理，我们只需要解决一下几个问题：

①依赖的目的是什么

当 A jar 包用到了 B jar 包中的某些类时，A 就对 B 产生了依赖，这是概念上的描述。那么如何在项目中以依赖的方式引入一个我们需要的 jar 包呢？

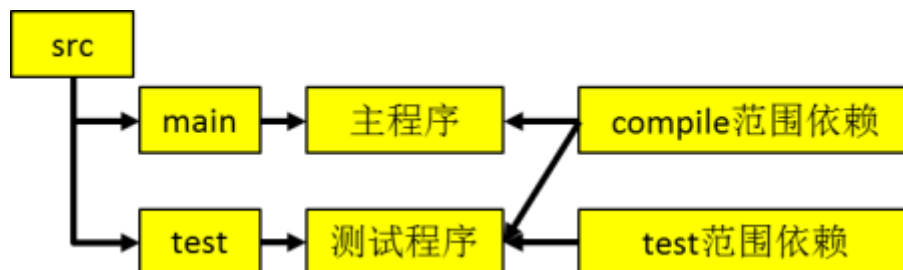
答案非常简单，就是使用 dependency 标签指定被依赖 jar 包的坐标就可以了。

```
<dependency>
  <groupId>com.admiral.maven</groupId>
  <artifactId>Hello</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <scope>compile</scope>
</dependency>
```

②依赖的范围

大家注意到上面的依赖信息中除了目标 jar 包的坐标还有一个 scope 设置，这是依赖的范围。依赖的范围有几个可选值，我们用得到的是：compile、test、provided 三个。

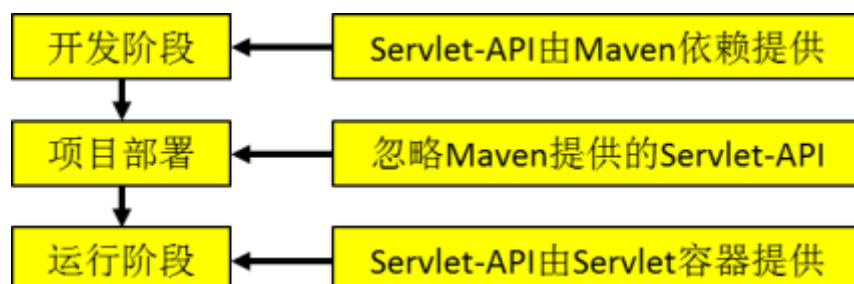
(1) 从项目结构角度理解 compile 和 test 的区别



结合具体例子：对于 HelloFriend 来说，Hello 就是服务于主程序的，junit 是服务于测试程序的。HelloFriend 主程序需要 Hello 是非常明显的，测试程序由于要调用主程序所以也需要 Hello，所以 compile 范围依赖对主程序和测试程序都应该有效。

HelloFriend 的测试程序部分需要 junit 也是非常明显的，而主程序是不需要的，所以 test 范围依赖仅仅对于主程序有效。

(2) 从开发和运行这两个不同阶段理解 compile 和 provided 的区别



(3) 有效性总结

	compile	test	provided
主程序	√	×	√
测试程序	√	√	√
参与部署	√	×	×

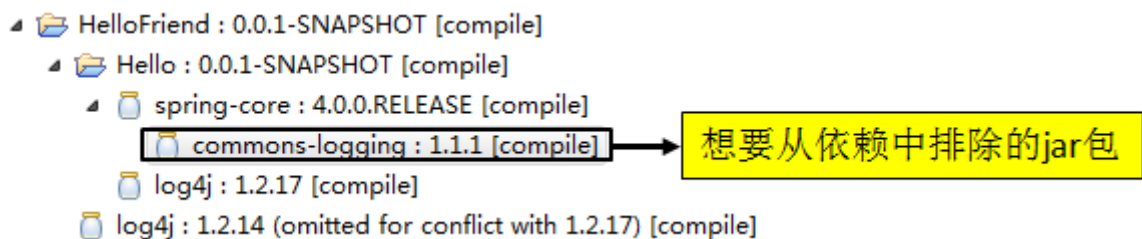
③依赖的传递性

A 依赖 B, B 依赖 C, A 能否使用 C 呢? 那要看 B 依赖 C 的范围是不是 compile, 如果是则可用, 否则不可用。

Maven 工程	依赖范围	对 A 的可见性		
A	B	C	compile	√
D	test	×		
E	provided	×		

④依赖的排除

如果我们在当前工程中引入了一个依赖是 A, 而 A 又依赖了 B, 那么 Maven 会自动将 A 依赖的 B 引入当前工程, 但是个别情况下 B 有可能是一个不稳定版, 或对当前工程有不良影响。这时我们可以在引入 A 的时候将 B 排除。



- 配置方式

```

<dependency>
  <groupId>com.admiral.maven</groupId>
  <artifactId>HelloFriend</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <type>jar</type>
  <scope>compile</scope>
  <exclusions>
    <exclusion>
      <groupId>commons-logging</groupId>
      <artifactId>commons-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>

```

```
HelloFriend : 0.0.1-SNAPSHOT [compile]
└─ Hello : 0.0.1-SNAPSHOT [compile]
    ├── spring-core : 4.0.0.RELEASE [compile]
    ├── log4j : 1.2.17 [compile]
    └─ log4j : 1.2.14 (omitted for conflict with 1.2.17) [compile]
```

```
<properties>
  <admiral.spring.version>4.1.1.RELEASE</admiral.spring.version>
</properties>
```

- 引用前面声明的版本号

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>${admiral.spring.version}</version>
  </dependency>
</dependencies>
```

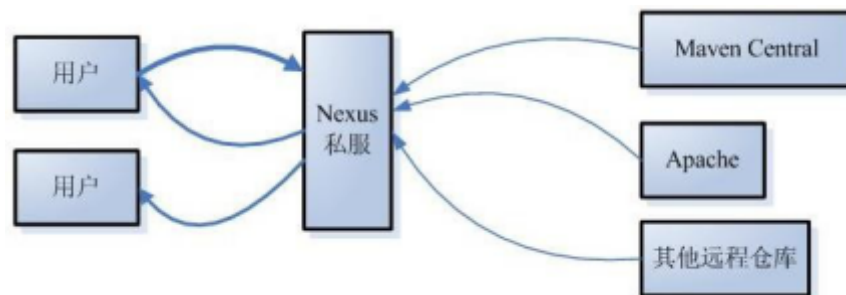
8. 仓库

8.1 分类

[1] 本地仓库：为当前本机电脑上的所有 Maven 工程服务。

[2] 远程仓库：

(1) 私服：架设在当前局域网环境下，为当前局域网范围内的所有 Maven 工程服务。



(2) 中央仓库：架设在 Internet 上，为全世界所有 Maven 工程服务。

(3) 中央仓库的镜像：架设在各个大洲，为中央仓库分担流量。减轻中央仓库的压力，同时更快的响应用户请求。

8.2 仓库中的文件

[1] Maven 的插件

[2] 我们自己开发的项目的模块

[3] 第三方框架或工具的 jar 包

不管是什么样的 jar 包，在仓库中都是按照坐标生成目录结构，所以可以通过统一的方式查询或依赖。

9. 生命周期

9.1 什么是 Maven 的生命周期

- Maven 生命周期定义了各个构建环节的执行顺序，有了这个清单，Maven 就可以自动化的执行构建命令了。
- Maven 有三套相互独立的生命周期，分别是：
 - ① Clean Lifecycle 在进行真正的构建之前进行一些清理工作。
 - ② Default Lifecycle 构建的核心部分，编译，测试，打包，安装，部署等等。
 - ③ Site Lifecycle 生成项目报告，站点，发布站点。

它们是相互独立的，你可以仅仅调用 clean 来清理工作目录，仅仅调用 site 来生成站点。当然你也可以直接运行 mvn clean install site 运行所有这三套生命周期。

每套生命周期都由一组阶段(Phase)组成，我们平时在命令行输入的命令总会对应于一个特定的阶段。比如，运行 mvn clean，这个 clean 是 Clean 生命周期的一个阶段。有 Clean 生命周期，也有 clean 阶段。

9.2 Clean 生命周期

Clean 生命周期一共包含了三个阶段：

- ① pre-clean 执行一些需要在 clean 之前完成的工作
- ② clean 移除所有上一次构建生成的文件
- ③ post-clean 执行一些需要在 clean 之后立刻完成的工作

9.3 Site 生命周期

- ① pre-site 执行一些需要在生成站点文档之前完成的工作
- ② site 生成项目的站点文档
- ③ post-site 执行一些需要在生成站点文档之后完成的工作，并且为部署做准备

④site-deploy 将生成的站点文档部署到特定的服务器上

这里经常用到的是 site 阶段和 site-deploy 阶段，用以生成和发布 Maven 站点，这可是 Maven 相当强大的功能，Manager 比较喜欢，文档及统计数据自动生成，很好看。

9.4 Default 生命周期

Default 生命周期是 Maven 生命周期中最重要的一个，绝大部分工作都发生在这个生命周期中。这里，只解释一些比较重要和常用的阶段：

validate

generate-sources

process-sources

generate-resources

process-resources 复制并处理资源文件，至目标目录，准备打包。

compile 编译项目的源代码。

process-classes generate-test-sources process-test-sources generate-test-resources

process-test-resources 复制并处理资源文件，至目标测试目录。

test-compile 编译测试源代码。

process-test-classes

test 使用合适的单元测试框架运行测试。这些测试代码不会被打包或部署。

prepare-package

package 接受编译好的代码，打包成可发布的格式，如JAR。pre-integration-test

integration-test

post-integration-test verify

install 将包安装至本地仓库，以让其它项目依赖。

deploy 将最终的包复制到远程的仓库，以让其它开发人员与项目共享或部署到服务器上运行。

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.18363.1139]
(c) 2019 Microsoft Corporation。保留所有权利。

F:\workspace_eclipse01\javaee\HelloFriend>mvn compile
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.admiral.maven:HelloFriend >-----
[INFO] Building HelloFriend 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloFriend ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloFriend ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1.462 s
[INFO] Finished at: 2020-10-26T02:45:03+08:00
[INFO]
F:\workspace_eclipse01\javaee\HelloFriend>
```

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.18363.1139]
(c) 2019 Microsoft Corporation。保留所有权利。

F:\workspace_eclipse01\javaee\HelloFriend>mvn test-compile
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.admiral.maven:HelloFriend >-----
[INFO] Building HelloFriend 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloFriend ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloFriend ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ HelloFriend ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ HelloFriend ---
[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 F:\workspace_eclipse01\javaee\HelloFriend\target\test-classes
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1.836 s
[INFO] Finished at: 2020-10-26T02:46:27+08:00
[INFO]
F:\workspace_eclipse01\javaee\HelloFriend>_
```

```
C:\Windows\System32\cmd.exe
F:\workspace_eclipse01\javaee\HelloFriend>mvn package
[INFO] Scanning for projects...
[INFO] -----< com.admiral.maven:HelloFriend >-----
[INFO] Building HelloFriend 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloFriend ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloFriend ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ HelloFriend ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ HelloFriend ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ HelloFriend ---
[INFO] Surefire report directory: F:\workspace_eclipse01\javaee\HelloFriend\target\surefire-reports

-----
T E S T S
-----
Running com.admiral.maven>HelloFriendTest
Hello litingwei! I am John
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.055 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ HelloFriend ---
[INFO] Building jar: F:\workspace_eclipse01\javaee\HelloFriend\target>HelloFriend-0.0.1-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.703 s
[INFO] Finished at: 2020-10-26T02:47:49+08:00
[INFO] -----
F:\workspace_eclipse01\javaee\HelloFriend>
```

9.5 生命周期与自动化构建

运行任何一个阶段的时候，它前面的所有阶段都会被运行，例如我们运行 `mvn install` 的时候，代码会被编译，测试，打包。这就是 Maven 为什么能够自动执行构建过程的各个环节的原因。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。

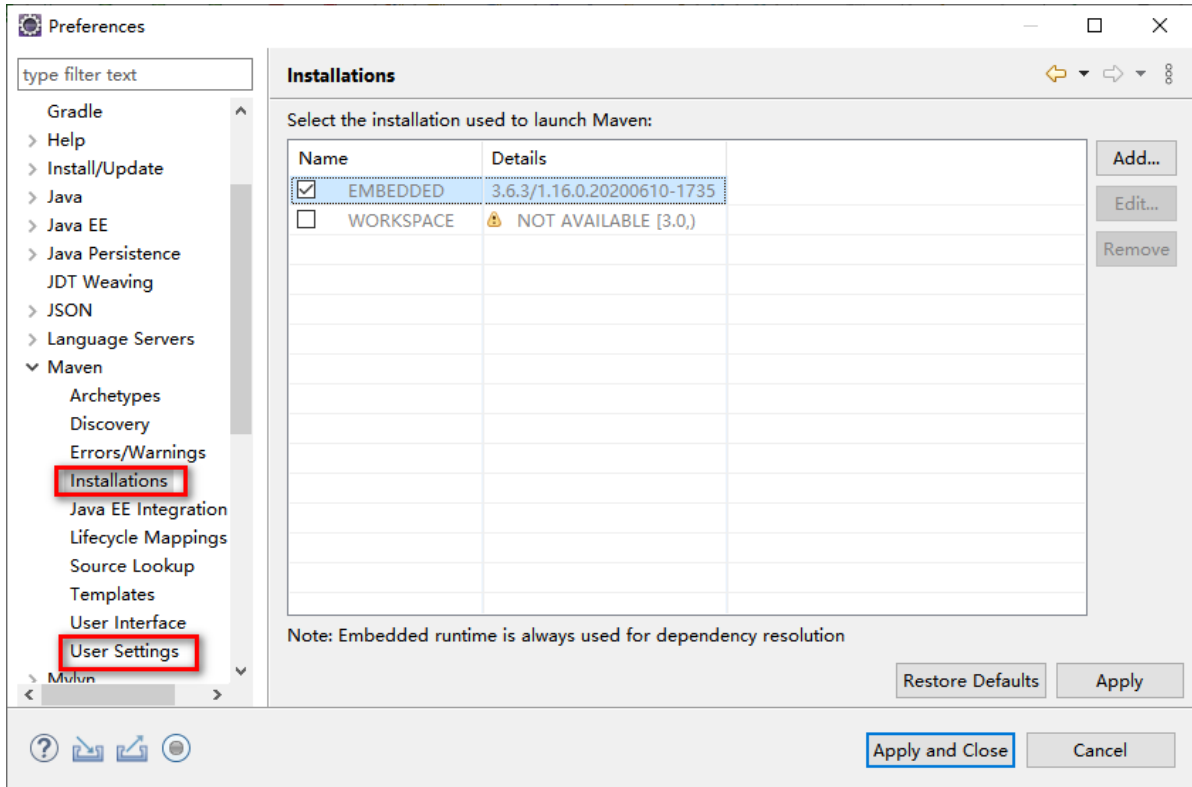
10. 插件和目标

- Maven 的核心仅仅定义了抽象的生命周期，具体的任务都是交由插件完成的。
- 每个插件都能实现多个功能，每个功能就是一个插件目标。
- Maven 的生命周期与插件目标相互绑定，以完成某个具体的构建任务。

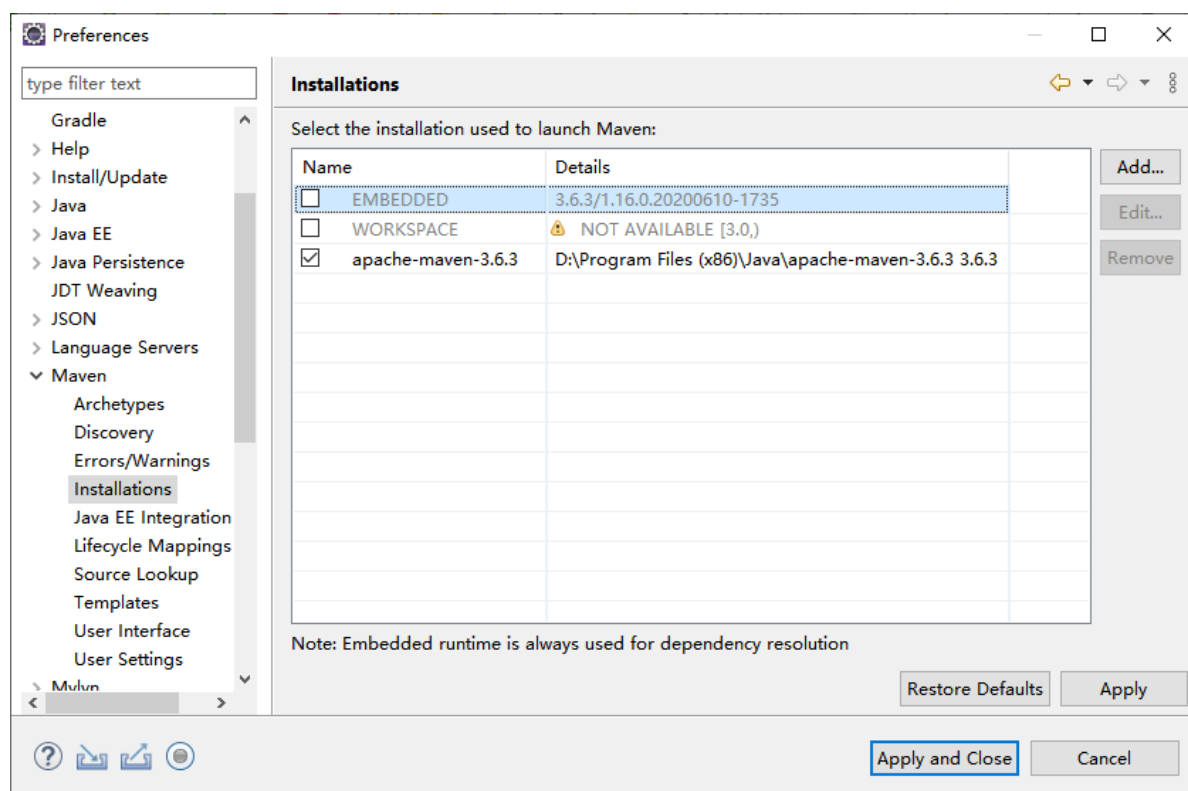
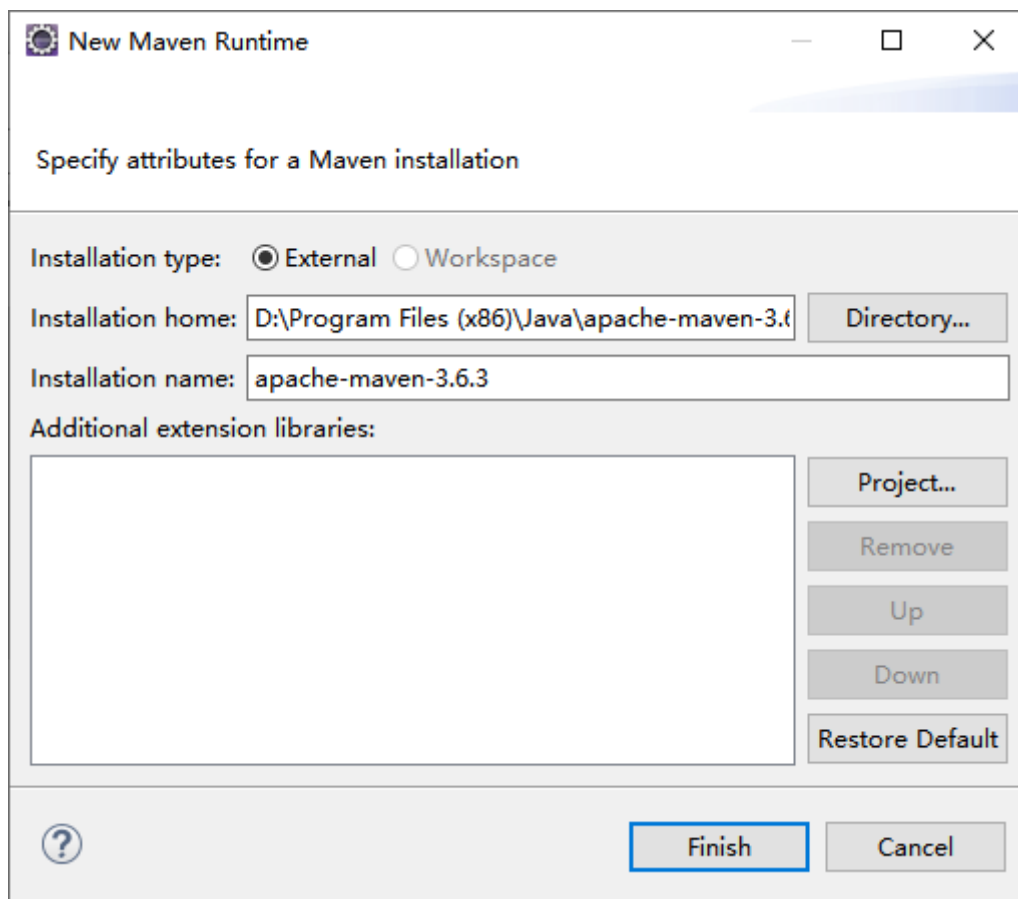
例如：`compile` 就是插件 `maven-compiler-plugin` 的一个目标；`pre-clean` 是插件 `maven-clean-plugin` 的一个目标。

生命周期阶段	插件目标	插件
compile	compile	maven-compiler-plugin
test-compile	testCompile	maven-compiler-plugin

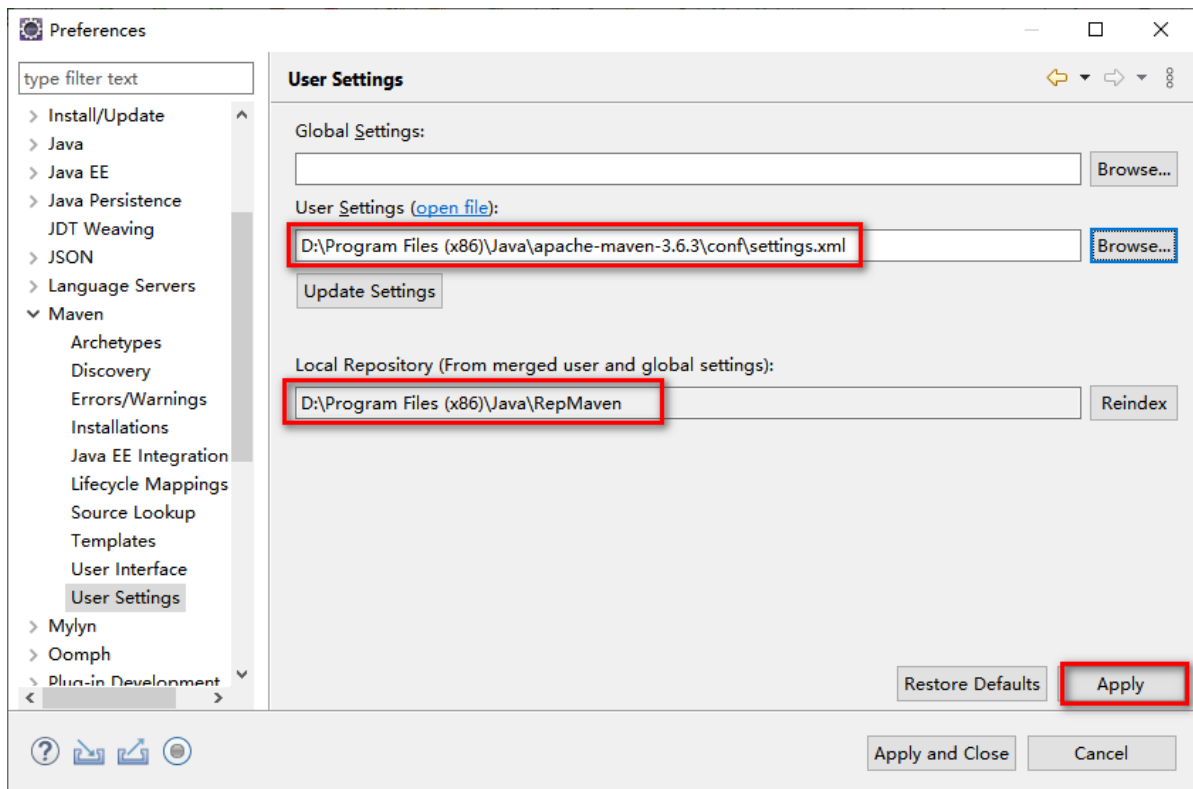
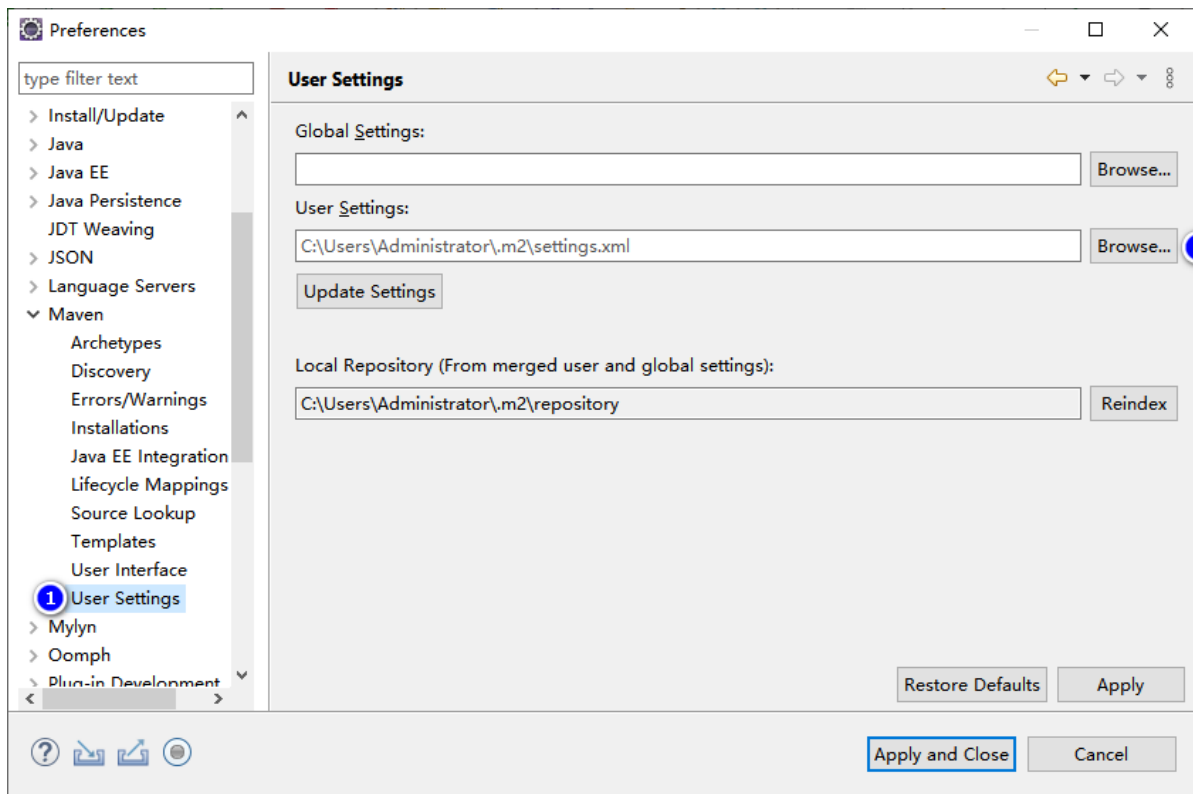
11. Eclipse Maven 插件的设置



- Installations 的设置



- User Settings



12. 使用 Eclipse 创建 Maven 项目

12.1 创建 Java 项目

12.2 创建 Web 项目
