## 是什么:

Maven是一个用来把源代码构建成可发布的构件的工具，即将Java项目构建相关的过程自动化的工具。它成功的原因是它定义了构建软件的一般的接口（?）。

## 解决什么问题:

这个项目需要构建什么？需要现在什么类库？把类库放哪里？构建中我该运行什么目标？

## 如何解决的:

Maven核心只是解析一些 XML 文档，管理生命周期与插件，把职责委派给一组 Maven 插件。从远程仓库获取依赖和插件的这一事实允许了构建逻辑的全局性重用。

Maven有一个定义好的生命周期和一组知道如何构建和装配软件的通用插件。

一个构建周期由一系列的构建阶段组成，每一个构建阶段由一系列的目标组成。

在POM中项目的坐标和结构被声明，构建被配置，与其它项目的关联也被定义。

## 依赖管理

Maven内嵌有依赖管理的功能。你只需要在pom文件里指定依赖jar包的名称、版本号，Maven会自动下载并放到你的Maven本地仓库中。如果这些外部jar包依赖了其它的库，它们也会被下载到你的Maven本地仓库。

**可选依赖：<optional>true</ optional >**

编译这个项目的时候你需要这个依赖类库，但是你不希望在使用你类库的项目中这个依赖类库同时作为传递性运行时依赖出现，就可以将其设计为可选依赖。

**依赖范围：<scope>…</cope>**

范围控制哪些依赖在哪些classpath中可用，哪些依赖包含在一个应用中。

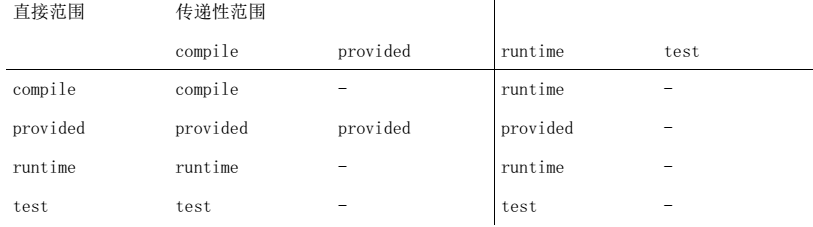
Compile：编译范围依赖在所有的classpath中可用，同时它们也会被打包；

Provided：Provided依赖只有在当JDK或者一个容器已提供该依赖之后才使用；Provided依赖在编译classpath可用，它们不是传递性的，也不会被打包；

Runtime：运行时依赖在运行和测试系统的时候需要，但在编译的时候不需要；

Test：它们只有在测试编译和测试运行阶段可用。

每种依赖范围不仅仅影响声明项目中的依赖范围，它也对所传递性依赖起作用：



**依赖冲突消解：<exclusion> …</exclusion>**

有很多时候你需要排除一个传递性依赖。为了用另外的依赖来替换这个传递性依赖，你需要排除这个传递性以依赖，然后在你的项目中再声明一个依赖。

## 该怎么用

在Eclipse中利用m2eclipse插件，Maven的功能在Eclipse上就能操作和实现。