1. **Maven là gì ?**

**Maven** là công cụ quản lý và thiết lập tự động 1 dự án phần mềm. Chủ yếu dùng cho các lập trình viên java, nhưng nó cũng có thể được dùng để xây dựng và quản lý các dự án dùng C#, Ruby, Scala hay ngôn ngữ khác. Maven phục vụ mục đích tương tự như Apache Ant, nhưng nó dựa trên khái niệm khác và cách hoạt động khác. Maven được quản lý bởi Apache Software Foundation, nó là một phần của dự án Jakarta Project.

Maven dùng khái niệm *Project Object Model* (POM) để mô tả việc build project, các thành phần phụ thuộc và các module. Nó định nghĩa trước các target cho việc khai báo task, trình biên dịch, đóng gói và thứ tự hoạt động để mọi việc diễn ra tốt nhất.

Maven tải các thư viện, plug-in từ 1 hoặc nhiều repositories. Maven cung cấp việc hỗ trợ tải file từ repository trung tâm của Maven 2 và các repositories khác. Cũng có thể upload các sản phẩm lên repository sau khi đã build xong.

Maven được xây dựng dựa trên kiến trúc plugin - cho phép sử dụng bất kỳ ứng dụng điều khiển nào thông qua đầu vào tiêu chuẩn. Về mặt lý thuyết, điều này cho phép bất kỳ ai cũng có thể viết các plugin để giao tiếp với các build tool (trình biên dịch, unit test tool, etc) cho bất kỳ ngôn ngữ nào khác. Nhưng trong thực tế, hỗ trợ và sử dụng cho các ngôn ngữ khác đã bị thu hẹp. Hiện tại thì plugin cho.Net đã có và vẫn được duy trì, và một plugin cho C/C++ cũng được duy trì cho Maven 1.

1. **Các đặc trưng chính của Maven**

**a.Thiết lập dự án đơn giản, tuân theo các cách tiếp cận thực tế tốt nhất**: Maven tránh tối đa việc phải cấu hình nhiều bằng cách cung cấp các project templates (named archetypes).
**b.Quản lý phụ thuộc**: Các phụ thuộc được update, downloading và validating tự động.

**c. Cô lập các phụ thuộc dự án & các plugin**: Các phụ thuộc dự án được trích rút từ dependency repositories, các plugins lại được trích rút từ plugin repositories
=> tránh xung đột khi plugin khởi tạo & download các phụ thuộc thêm vào.
**d. Build dự án dựa trên mô hình**: Cho phép build nhiều dự án với các định dạng như .jar, .war, .ear, ejb, .pom, .maven- plugin, .metadata, ...

**e. Sinh site cho dự án**: Cho phép sinh ra các tài liệu dạng website&PDF cho dự án
**f. Quản lý phát hành**: Tự động tích hợp với source control system và quản lý release của dự án mà không cần cấu hình thêm.
**g. Hệ thống kho thành phẩm phong phú**: Các phụ thuộc dự án được download, đẩy lên các kho Maven.

1. **Maven-POM**Maven POM là một file XML đặc biệt trong dự án Maven, viết tắt của Project Object Model.

POM định nghĩa cấu trúc của dự án Maven và chứa các thông tin quan trọng như các phụ thuộc, plugin, mô tả dự án, cài đặt biên dịch và quản lý dự án. Mỗi dự án Maven đều có một file POM, được đặt tên là `pom.xml`, nằm ở thư mục gốc của dự án. POM cung cấp một cách tiêu chuẩn hóa để quản lý dự án và các phụ thuộc của nó, giúp đơn giản hóa việc xây dựng, kiểm thử, đóng gói và triển khai ứng dụng.



1. **Maven- Build Life Cycle**

Maven Build Life Cycle là một chuỗi các pha (phase) mà Maven thực hiện để xây dựng và quản lý một dự án. Mỗi phase trong life cycle là một bước cụ thể trong quá trình xây dựng dự án, và các phase này được thực thi theo một thứ tự nhất định.

Maven xác định ba life cycle chính là: default, clean và site. Mỗi life cycle này bao gồm một loạt các phase.

Ví dụ, trong default life cycle, các phase bao gồm:

1. `validate`: Xác nhận tính hợp lệ của dự án, ví dụ: kiểm tra xem các POM có hợp lệ không.

2. `compile`: Biên dịch mã nguồn của dự án.

3. `test`: Chạy các test unit.

4. `package`: Đóng gói các file compiled thành một artifact (jar, war, etc.).

5. `verify`: Kiểm tra kết quả của việc kiểm thử.

6. `install`: Cài đặt artifact vào local repository của Maven để có thể sử dụng trong các dự án khác trên máy cục bộ.

7. `deploy`: Sao chép artifact vào remote repository để chia sẻ với các thành viên khác trong nhóm phát triển hoặc công cộng.

Mỗi phase trong life cycle có thể được kích hoạt bằng cách gọi một lệnh Maven cụ thể. Ví dụ, để chạy default life cycle, bạn có thể sử dụng lệnh `mvn clean install`, trong đó `clean` và `install` là hai phase trong default life cycle.

1. Maven- Repositories

Maven Repositories là nơi lưu trữ các artifacts (các gói phần mềm, thư viện, plugin, etc.) và metadata của chúng. Maven sử dụng các repositories để tải xuống các dependencies của dự án và cung cấp cơ sở hạ tầng cho việc chia sẻ và quản lý các artifacts trong cộng đồng phát triển phần mềm.

Có 3 loại repositories chính trong Maven:

* Local Repository: Là một thư mục trên máy tính của bạn, Maven lưu trữ các artifacts được tải xuống từ remote repositories vào local repository này. Mỗi khi bạn cài đặt một artifact mới từ remote repository hoặc sử dụng một artifact mà Maven chưa từng sử dụng trước đó, nó sẽ được lưu vào local repository để sử dụng lại trong tương lai.
* Central Repository: Là một remote repository mặc định trong Maven, cung cấp hàng ngàn artifacts và là nguồn lưu trữ chính thức cho Maven. Nó là một phần của Remote Repository, được quản lý và duy trì bởi cộng đồng Maven. Maven Central chứa các artifacts được chia sẻ bởi các tổ chức và cá nhân từ khắp nơi trên thế giới và là nguồn lợi quan trọng cho phát triển phần mềm Java.
* Remote Repository: Là các máy chủ từ xa hoặc kho lưu trữ trực tuyến, nơi chứa các artifacts và metadata của chúng. Maven sẽ tải các dependencies từ remote repository khi cần thiết. Maven Central là một trong những remote repository phổ biến nhất, nơi lưu trữ hàng ngàn artifacts mở và phần lớn các thư viện phổ biến được sử dụng trong các dự án Java.

Ví dụ: Maven Central sử dụng để tải các dependencies. Khi bạn cần sử dụng một thư viện như Apache Commons hoặc Google Gson trong dự án của mình, Maven sẽ tải các artifacts tương ứng từ Maven Central để sử dụng trong dự án của bạn.

1. **Maven- Plugins**

Maven- Plugins là các thành phần mở rộng được sử dụng để thực hiện các công việc cụ thể trong quá trình xây dựng dự án. Các plugin cung cấp các mục tiêu (goals) có thể được kích hoạt thông qua các lệnh Maven. Các mục tiêu này thường thực hiện các nhiệm vụ như biên dịch mã nguồn, đóng gói ứng dụng, chạy các test, triển khai ứng dụng, và nhiều nhiệm vụ khác.

Ví dụ: Cấu hình của plugin trong file pom.xml



* groupId, artifactId, và version xác định plugin được sử dụng.
* configuration chứa các cấu hình cho plugin, trong trường hợp này là phiên bản của Java được sử dụng cho biên dịch (source và target là 1.8).

Khi chạy lệnh mvn compile, plugin Maven Compiler sẽ được kích hoạt và mã nguồn Java của dự án sẽ được biên dịch bằng phiên bản Java được cấu hình.

1. **Maven – Build, Test and Run Maven project**
* Build**:**
* Sử dụng mục tiêu package để đóng gói dự án thành một artifact (jar, war, etc.).
* Lệnh: mvn package
* Sau khi thực thi lệnh này, Maven sẽ thực hiện các bước biên dịch, kiểm tra, đóng gói và tạo ra artifact tương ứng (thường là file JAR) trong thư mục target.
* Test:
* Sử dụng mục tiêu test để chạy các test unit của dự án.
* Lệnh: mvn test
* Maven sẽ tìm kiếm các test trong thư mục src/test/java và thực thi chúng. Kết quả kiểm thử sẽ được hiển thị trên giao diện dòng lệnh.
* Run:
* Đối với các ứng dụng Java, sau khi đã xây dựng (build) và kiểm thử (test), bạn có thể chạy ứng dụng bằng cách sử dụng plugin Maven Exec.
* mvn exec:java -Dexec.mainClass="<Tên lớp chính> "
1. **Maven- Auild Automation**

Maven- Build Automation là quá trình tự động hóa các công việc liên quan đến việc xây dựng, kiểm thử và triển khai phần mềm, giúp giảm thiểu sự phụ thuộc vào công việc thủ công và giảm rủi ro do sai sót con người.

Lo

1. **Maven – Manage Dependencies**

Manage Dependencieslà một trong những tính năng quan trọng nhất, giúp dự án Java có thể sử dụng các thư viện và frameworks một cách dễ dàng và hiệu quả. Dưới đây là cách Maven quản lý dependencies:

* Khai báo Dependencies trong File POM: Trong file `pom.xml`, bạn cần khai báo các dependencies mà dự án của bạn sử dụng. Điều này bao gồm groupId, artifactId và version của mỗi dependency.



* Tải Dependencies từ Remote Repositories: Khi bạn chạy lệnh Maven để xây dựng dự án hoặc cài đặt dependencies, Maven sẽ tải các dependencies từ các remote repositories như Maven Central hoặc các remote repositories khác mà bạn đã cấu hình.
* Cài Đặt Dependencies vào Local Repository: Các dependencies được tải về từ remote repositories sẽ được cài đặt vào local repository trên máy tính của bạn. Lần sau khi bạn sử dụng dependencies đó trong dự án khác, Maven sẽ sử dụng dependencies từ local repository thay vì phải tải lại từ remote repository.
* Xử lý Phiên Bản (Version Handling): Maven giúp quản lý các phiên bản của dependencies một cách nhất quán và dễ dàng. Bạn có thể chỉ định phiên bản chính xác của một dependency hoặc sử dụng các biểu thức như `LATEST`, `RELEASE`, `1.0.\*` để chỉ định phiên bản.
* Phạm vi (Scope): Maven hỗ trợ các phạm vi (scope) như `compile`, `test`, `provided`, `runtime`,... để quy định cách dependencies được sử dụng trong các giai đoạn khác nhau của quá trình phát triển.