# Maven

## What is Maven?

Maven is a free, open-source build automation tool used for Java-based projects. It helps developers manage dependencies, compile source code, and package applications. Maven was developed by the Apache Software Foundation.

## What is a Build Tool in DevOps?

A build tool is a software program that automates the process of converting source code into an executable format. The source code is usually stored in a version control system like GitHub.

## Types of Build Tools

Different programming languages have different build tools:

1. Java - Apache Maven, Apache ANT (older version).

2. Python - PyBuilder.

3. .NET - MSBuild (Microsoft Build Engine).

4. Node.js - Gulp, Grunt, Gradle, Webpack.

## How Does Maven Work?

Maven follows a specific workflow:

1. Developers write code and push it to a GitHub repository.

2. Maven compiles the code and packages it into different formats:

- EAR (Enterprise Application Archive)

- WAR (Web Application Archive)

- JAR (Java Archive)

3. These packages are deployed to a web server (like Tomcat).

4. End users can then access the application.

## What is POM.xml?

POM.xml (Project Object Model) is a key file in a Maven project. It:

- Manages project dependencies (external libraries needed for the project).

- Defines the build process.

- Ensures that all necessary tools are included in the project.

Each project contains a unique POM.xml file. It is also called a 'Super POM' or 'Parent POM'.

## Plugins in Maven

Plugins add extra features to Maven. They are used to install dependencies (external features).

There are two types of plugins:

1. Built-in Plugins – Installed once and used automatically.

2. Added Plugins – Installed separately and reusable.

## Maven Repositories

Maven uses three types of repositories to store dependencies:

1. Central Repository – Online repository like GitHub.

2. Remote Repository – Used by organizations like IBM to store private dependencies.

3. Local Repository – Stored on a developer’s computer.

## Maven Lifecycle

Maven follows three main stages:

1. Default Phase (Main Build Process):

- Compile: Converts source code into bytecode.

- Validate: Ensures code is correct.

- Test: Runs unit tests.

- Package: Creates a JAR, WAR, or EAR file.

- Install: Saves the package in the local repository.

- Verify: Checks if the package is correctly built.

2. Clean Phase (Before Compilation):

- Pre-clean: Checks for old build files.

- Clean: Deletes old JAR/WAR/EAR files.

- Post-clean: Saves new build files.

3. Site Phase (Documentation & Deployment):

- Pre-site: Prepares files for deployment.

- Site: Generates documentation and reports.

- Post-site: Finalizes documentation.

- Site-deploy: Uploads documentation to a server.

## What is Apache ANT?

Apache ANT (Another Neat Tool) is an older build automation tool used primarily for Java projects. It was developed by Apache Software Foundation before Maven and was widely used for building and deploying applications.

| **Feature** | **Apache ANT** | **Apache Maven** |
| --- | --- | --- |
| **Build File** | Uses build.xml | Uses pom.xml |
| **Dependency Management** | No built-in dependency management | Yes, automatically downloads dependencies |
| **Lifecycle** | No predefined lifecycle | Has a structured lifecycle (compile, test, package, deploy) |
| **Plugins** | Requires manual scripting | Supports plugins for automation |
| **Configuration** | More complex (manual scripting) | Simpler (convention over configuration) |
| **Flexibility** | More flexible but requires more effort | Less flexible but easier to use |
| **Reusability** | Scripts need to be written for each project | Standardized process across projects |