DevOps

[Maven 2](#_Toc194177113)

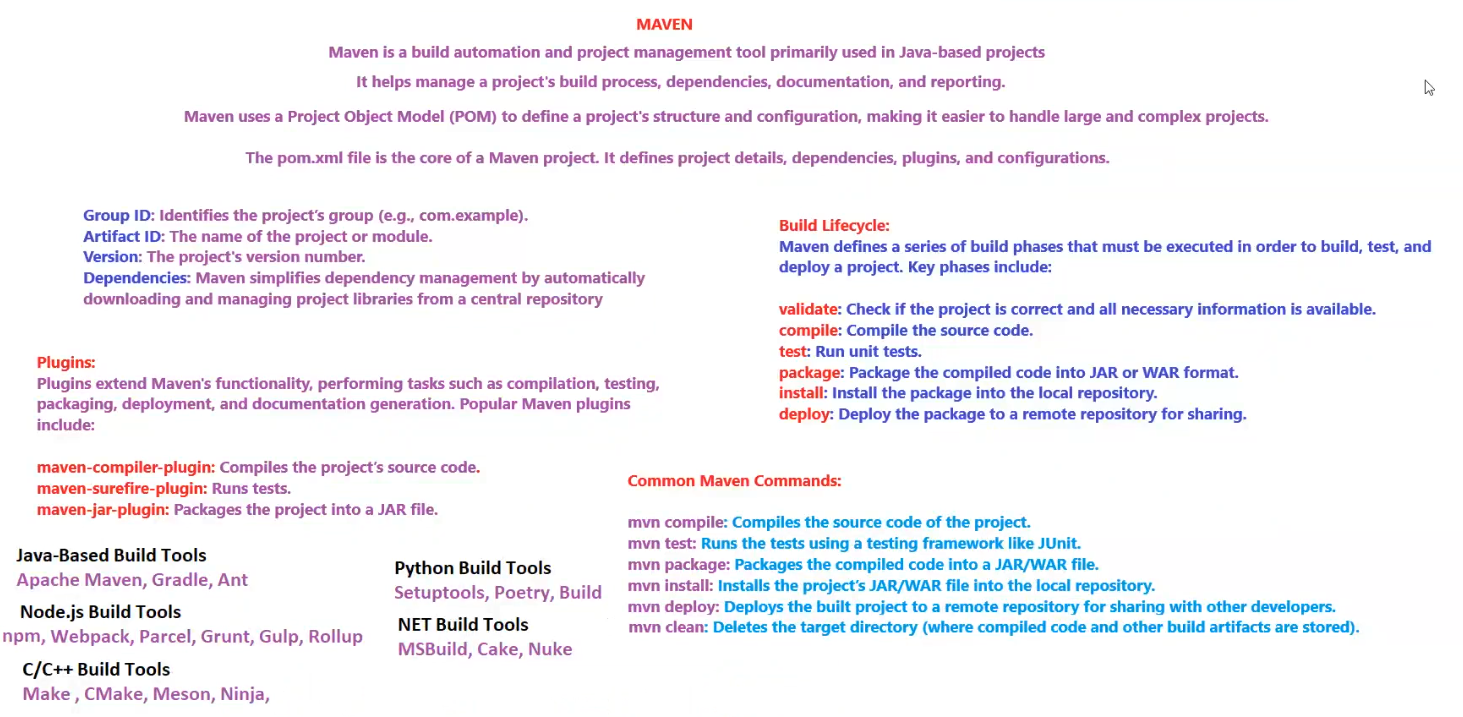
[Build Automation & Project Management Tool 2](#_Toc194177114)

[Maven Phases & Goals 4](#_Toc194177115)

[Maven, its build lifecycle, POM file, phases, plugins, and goals 5](#_Toc194177116)

# Maven

## Build Automation & Project Management Tool



Maven is a build automation and project management tool primarily used in **Java-based projects**.  
It helps manage a project's **build process, dependencies, documentation, and reporting**.

Maven uses a **Project Object Model (POM)** to define a project's **structure and configuration**, making it easier to handle large and complex projects.

The pom.xml file is the core of a Maven project. It defines:

* **Project details**
* **Dependencies**
* **Plugins**
* **Configurations**

**Key Components of a POM File**

* **Group ID**: Identifies the project’s group (e.g., com.example).
* **Artifact ID**: The name of the project or module.
* **Version**: The project's version number.
* **Dependencies**: Manages project libraries automatically by downloading and managing them from a central repository.

**Plugins in Maven**

Plugins extend Maven's functionality by performing tasks such as **compilation, testing, packaging, deployment, and documentation generation**.

**Popular Maven Plugins**

* **maven-compiler-plugin** → Compiles the project’s source code.
* **maven-surefire-plugin** → Runs unit tests.
* **maven-jar-plugin** → Packages the project into a JAR file.

**Maven Build Lifecycle**

Maven defines a series of **build phases** that must be executed to build, test, and deploy a project.

**Key Phases:**

1. **validate** → Checks if the project is correct and all necessary information is available.
2. **compile** → Compiles the source code.
3. **test** → Runs unit tests.
4. **package** → Packages the compiled code into JAR/WAR format.
5. **install** → Installs the package into the local repository.
6. **deploy** → Deploys the package to a remote repository for sharing.

**Common Maven Commands**

* mvn compile → **Compiles** the source code of the project.
* mvn test → **Runs unit tests** using a testing framework like JUnit.
* mvn package → **Packages** the compiled code into a JAR/WAR file.
* mvn install → **Installs** the project's JAR/WAR file into the local repository.
* mvn deploy → **Deploys** the built project to a remote repository for sharing.
* mvn clean → **Deletes** the target directory (where compiled code and other build artifacts are stored).

**Build Tools for Different Technologies**

**Java-Based Build Tools**

* **Apache Maven, Gradle, Ant**

**Node.js Build Tools**

* **npm, Webpack, Parcel, Grunt, Gulp, Rollup**

**C/C++ Build Tools**

* **Make, CMake, Meson, Ninja**

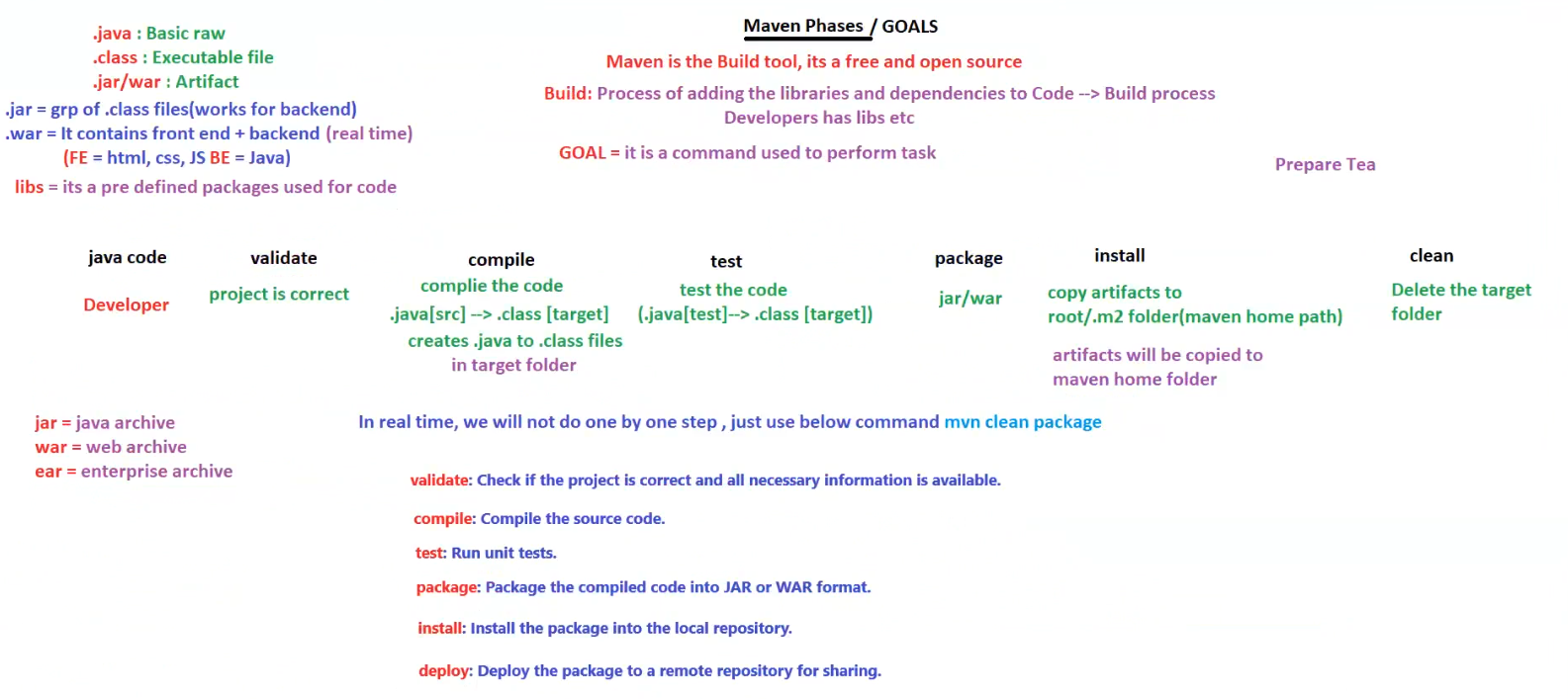
**Python Build Tools**

* **Setuptools, Poetry, Build**

**.NET Build Tools**

* **MSBuild, Cake, Nuke**

## Maven Phases & Goals



**Introduction to Maven**

* **Maven is a build tool** that is **free and open-source**.
* It helps in the **build process** by managing **libraries and dependencies**.
* **GOAL**: A command used to perform specific tasks during the build process.

**File Types in Java Projects**

* **.java** → Basic raw source code.
* **.class** → Executable file compiled from .java.
* **.jar/.war** → Artifacts (final packaged output).
  + **.jar** → A group of .class files (used for backend).
  + **.war** → Contains **frontend + backend** (used in real-time applications).
  + **Frontend**: HTML, CSS, JS | **Backend**: Java.
  + **.ear** → Enterprise Archive (for enterprise applications).
* **libs** → Predefined packages used in the code.

**Maven Build Lifecycle & Phases**

Maven follows a structured process to build, test, and deploy Java projects.

| **Phase** | **Description** |
| --- | --- |
| **Validate** | Checks if the project is correct and all necessary information is available. |
| **Compile** | Compiles the source code (.java → .class). |
| **Test** | Runs unit tests (.java[test] → .class). |
| **Package** | Packages compiled code into .jar or .war format. |
| **Install** | Copies artifacts to ~/.m2 (Maven home path). |
| **Deploy** | Deploys the package to a remote repository for sharing. |
| **Clean** | Deletes the target folder (removes compiled code and artifacts). |

**Real-Time Usage**

* Instead of running each phase manually, you can use a single command:
* mvn clean package

This will **delete previous builds, compile the project, run tests, and package it**.

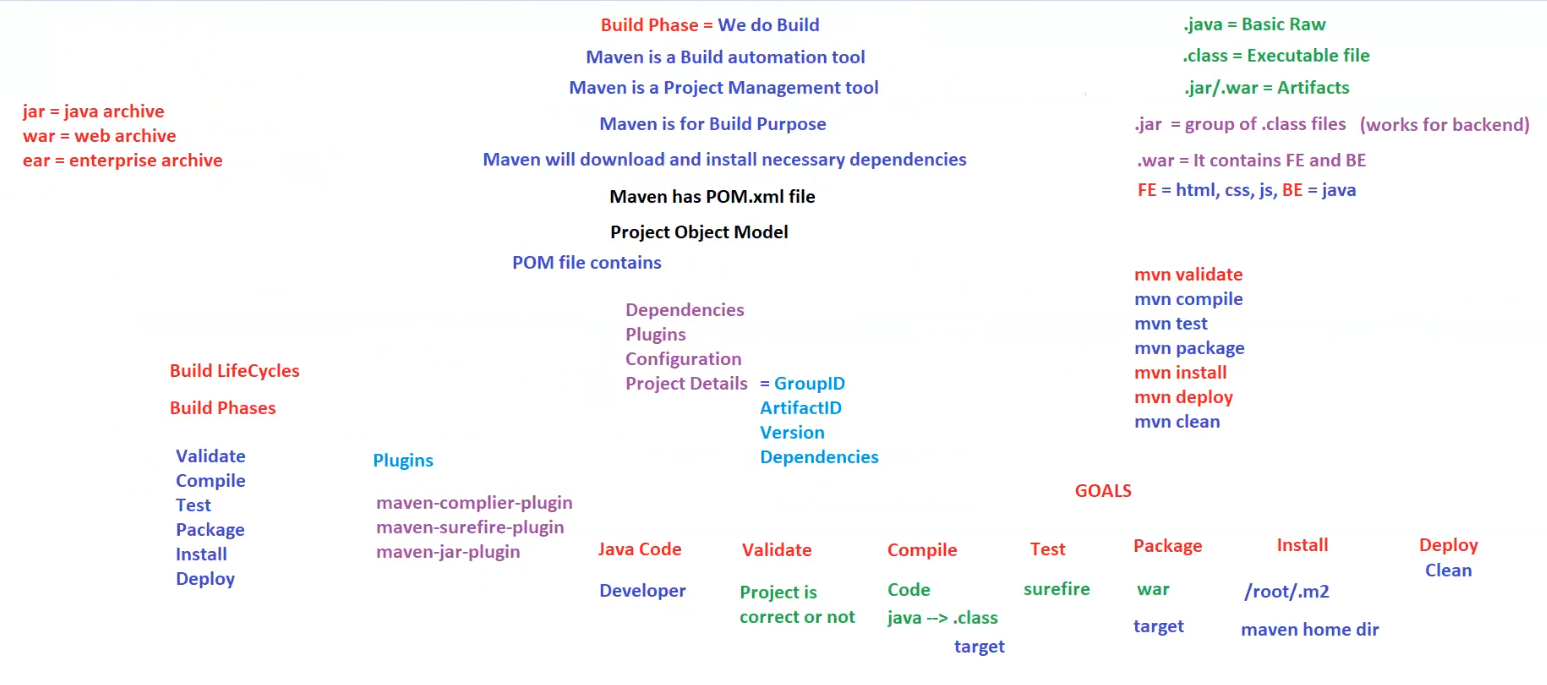
**Summary of Maven Phases**

* **validate** → Check if the project is correct.
* **compile** → Compile the source code.
* **test** → Run unit tests.
* **package** → Create a .jar/.war file.
* **install** → Install the package into the local repository.
* **deploy** → Deploy the package to a remote repository.
* **clean** → Remove old build artifacts.

## Maven, its build lifecycle, POM file, phases, plugins, and goals

**1. What is Maven?**

* **Maven is a build automation tool** for Java projects.
* It is also a **project management tool**.
* **Purpose:** Manages dependencies, compiles, tests, packages, and deploys projects.
* Maven automatically **downloads and installs necessary dependencies**.



**2. Maven POM (Project Object Model)**

* **Core file:** pom.xml
* Defines:
  + **Dependencies** (External libraries required)
  + **Plugins** (Extend Maven functionality)
  + **Configuration** (Build settings)
  + **Project Details:**
    - **Group ID**
    - **Artifact ID**
    - **Version**
    - **Dependencies**

**3. File Types in Java Projects**

* **.java** → Raw source code.
* **.class** → Compiled executable file.
* **.jar/.war** → Final artifacts.
  + **.jar** (Java Archive) → Backend services.
  + **.war** (Web Archive) → Frontend + Backend.
  + **.ear** (Enterprise Archive) → Large enterprise applications.
* **.jar** → Group of .class files.
* **.war** → Contains **Frontend (HTML, CSS, JS) + Backend (Java)**.

**4. Maven Build Lifecycle & Phases**

Maven follows an ordered process to build a project.

| **Phase** | **Description** |
| --- | --- |
| **Validate** | Checks if the project is correct and necessary information is available. |
| **Compile** | Converts .java files to .class files. |
| **Test** | Runs unit tests (JUnit, TestNG). |
| **Package** | Packages the compiled code into .jar or .war files. |
| **Install** | Installs the package into the local Maven repository (~/.m2). |
| **Deploy** | Deploys the packaged file to a remote repository. |
| **Clean** | Deletes previously compiled artifacts and the target folder. |

🔹 **Command to run a full build:**

mvn clean package

This command **cleans**, **compiles**, **tests**, and **packages** the project.

**5. Maven Plugins**

Plugins enhance Maven’s capabilities by performing tasks such as compilation, testing, packaging, and deployment.

Common Plugins:

* **maven-compiler-plugin** → Compiles Java code.
* **maven-surefire-plugin** → Runs unit tests.
* **maven-jar-plugin** → Creates .jar files.

**6. Maven Commands**

| **Command** | **Description** |
| --- | --- |
| mvn validate | Validates project structure. |
| mvn compile | Compiles Java files. |
| mvn test | Runs unit tests. |
| mvn package | Packages compiled code into .jar or .war. |
| mvn install | Installs package into local repository (.m2). |
| mvn deploy | Deploys project to remote repository. |
| mvn clean | Deletes compiled files and build artifacts. |

**7. Maven Goals**

* **Validate:** Ensures project is correct.
* **Compile:** Converts Java files to .class.
* **Test:** Runs unit tests.
* **Package:** Creates .war or .jar.
* **Install:** Moves artifacts to ~/.m2.
* **Deploy:** Pushes to a remote repository.
* **Clean:** Removes old build files.

**Final Thoughts**

This image summarizes Maven's structure, including its build lifecycle, POM file, commands, and plugins. It serves as a **quick reference** for developers working with Maven.