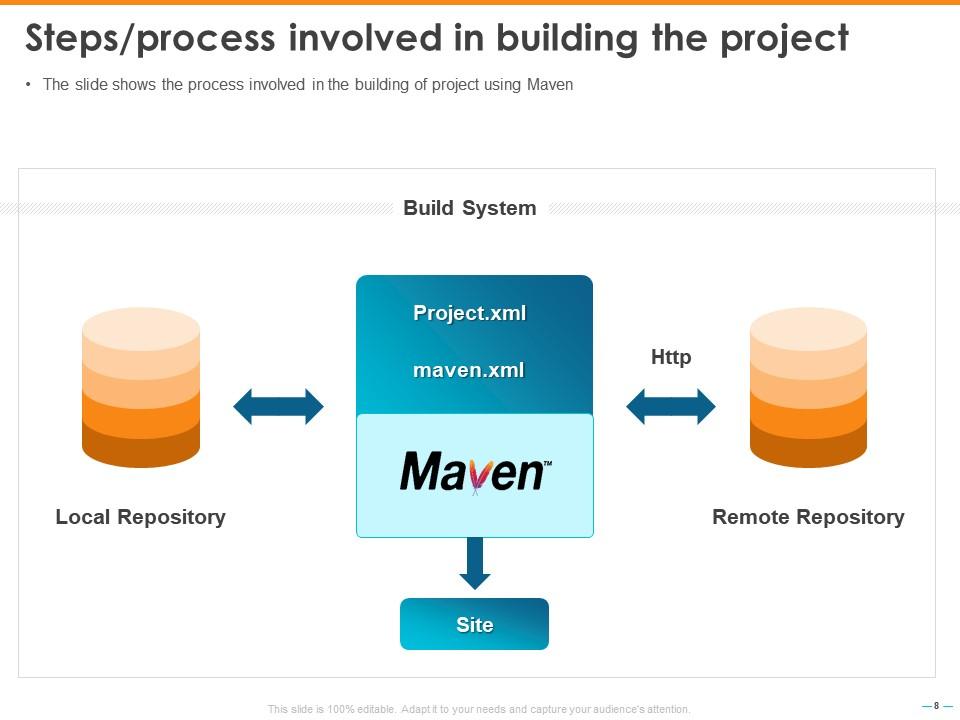
**Build tool- Maven: Maven Installation, Maven Build requirements**

What is Maven?

Apache Maven is a powerful build automation tool used primarily for Java projects. It simplifies project management by handling dependencies, compiling code, running tests, and packaging applications.



**Installing Maven on Ubuntu**

sudo apt update

sudo apt install maven -y

mvn -version

**Maven Build Requirements**

To build a project using Maven, ensure:

* **Maven is installed** (mvn -version)
* **Java is installed** (java -version)
* **A pom.xml file is present** in your project directory.

**Maven POM Builds (pom.xml), Maven Build Life Cycle, Maven Local Repository (.m2), Maven Global Repository, Group ID, Artifact ID, Snapshot, Maven Dependencies, Maven Plugins**

The **pom.xml** (Project Object Model) file is the heart of a Maven project. It defines project dependencies, build configurations, and plugins required to compile, test, and package the application.

To create a simple **Maven project**, run the following command:

mvn archetype:generate -DgroupId=com.example -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

* -DgroupId=com.example → This is like a namespace (usually your domain name in reverse).
* -DartifactId=my-app → This is the name of your project.
* -DarchetypeArtifactId=maven-archetype-quickstart → This tells Maven to use a simple Java project template.
* -DinteractiveMode=false → This makes the process automatic (no need to answer questions).

After running this command, a new folder called **my-app** will be created.

cd my-app

ls

ls -R

**CODE**

package com.example;

public class App {

public static void main(String[] args) {

System.out.println("Hello, Maven!");

}

}

mvn clean compile

Note:Clean: Remove any previously compiled files.

* Compile: Compile the source code.

mvn exec:java -Dexec.mainClass="com.example.App"

Note:This command tells Maven to run the main method in the com.example.App class.

mvn package

Note:This will create a JAR file in the target directory (

java -cp target/my-app-1.0-SNAPSHOT.jar com.example.App

When you create a Maven project, it automatically generates a pom.xml file inside your project folder (my-app).

**pom.xml**

**<?xml version="1.0" encoding="UTF-8"?>**

**<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.example</groupId>**

**<artifactId>my-app</artifactId>**

**<version>1.0-SNAPSHOT</version>**

**<properties>**

**<maven.compiler.source>1.8</maven.compiler.source>**

**<maven.compiler.target>1.8</maven.compiler.target>**

**</properties>**

**<dependencies>**

**<!-- Example: JUnit for testing -->**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>4.13.2</version>**

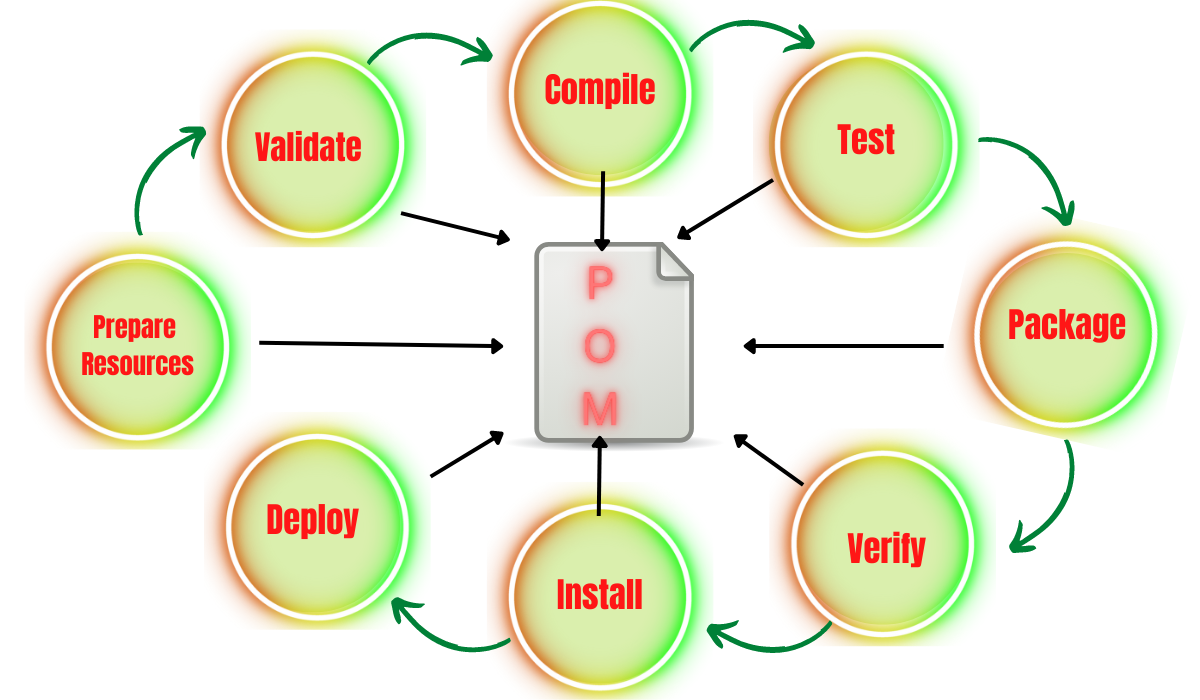
**<scope>test</scope>**

**</dependency>**

**</dependencies>**

**</project>**

**Maven Build Life Cycle**



| **hase** | **Description** |
| --- | --- |

|  |  |
| --- | --- |
| validate | Checks if the project structure is correct. |

|  |  |
| --- | --- |
| compile | Compiles Java source files. |

|  |  |
| --- | --- |
| test | Runs unit tests. |

|  |  |
| --- | --- |
| package | Creates a JAR or WAR file. |

|  |  |
| --- | --- |
| verify | Runs integration tests. |

|  |  |
| --- | --- |
| install | Installs the JAR/WAR in the local repository (.m2). |

|  |  |
| --- | --- |
| deploy | Uploads artifacts to a remote repository. |

To build a JAR:

mvn package

**Maven SNAPSHOT Versioning**

* **SNAPSHOT** versions (1.0-SNAPSHOT) are **unstable and frequently updated**.
* Maven fetches new builds automatically from repositories.

| **Version** | **Type** | **Behavior** |
| --- | --- | --- |

|  |  |  |
| --- | --- | --- |
| 1.0 |  | Stable, never changes. |

|  |  |  |
| --- | --- | --- |
| 1.0-SNAPSHOT | Snapshot | Updated frequently. |

**Maven Dependencies**

* Maven manages external libraries using <dependencies> in pom.xml.
* Dependencies are automatically downloaded from Maven Central.

## **Maven Plugins**

* Maven plugins **extend the build functionality**.

Plugin Purpose

maven-compiler-plugin Compiles Java code.

maven-jar-plugin Creates JAR files.

maven-surefire-plugin Runs unit tests.

maven-deploy-plugin Deploys artifacts to repositories.

**What is an Artifact in Maven?**

An **artifact** in Maven is a file generated when you build your project. It is usually a **JAR** (for Java projects) or **WAR** (for web applications). This file can be used in other projects or shared with others.