1.What is Maven?

Maven is an open-source build automation and project management tool for Java applications. It simplifies compilation, packaging, testing, and deployment through a standardized approach.

2. How Does Maven Work?

- Manages dependencies by downloading necessary JAR files.
- Uses `pom.xml` (Project Object Model) for configuration.
- Automates software development life cycle (compile → test → package → deploy).

3. Maven in DevOps

- Maven is useful in DevOps when:
- A project has many dependencies.
- Dependencies require frequent updates.
- Fast compilation and packaging (JAR/WAR/EAR) are needed.

4.Types of Applications:

- WAR – Web applications
- JAR – Java applications
- EAR – Enterprise applications

Limitation: Supports only Java-based applications.

5. Maven Repositories

- Central Repository – Public repository for dependencies.
- Remote Repository – Hosted by organizations for internal use.
- Local Repository – Cached dependencies on a developer's system.

6. Maven Life Cycle

Maven follows 3 stages with 7 build steps:

Stages:

Default – compile, validate, test, package, verify

Clean – pre-clean, clean, post-clean

Site – pre-site, site, post-site

Steps:

Validate – Ensures all dependencies exist.

Compile – Converts source code into bytecode.

Test – Runs JUnit test cases.

Package – Creates JAR/WAR files.

Verify – Checks for errors.

Install – Saves the package in the local repository.

Deploy – Uploads the package to a remote repository.

7. Maven vs. ANT

Feature	Maven	ANT
Type	Build automation & project management tool	Build automation tool only
Configuration File	`pom.xml` (Project Object Model)	`build.xml`
Dependency Management	Yes, handles automatically	No, manual handling required
Life Cycle	Yes, follows a predefined lifecycle	No lifecycle, tasks are scripted
Build Process	Convention over configuration (standardized)	Requires custom scripting
Learning Curve	Easier due to predefined structure	Harder due to manual configurations
Plugins Support	Yes, many built-in plugins	Limited plugins support
Test Integration	Supports JUnit/TestNG for automated testing	No built-in testing support
Usage	Preferred for large-scale Java projects	Used for simple build automation
Speed	Faster due to lifecycle automation	Slower due to manual scripting

Apache ANT: An older build tool that lacks dependency management and lifecycle support.

8. Maven Commands & Compilation

How to Compile Maven in Linux:

Connect to EC2 instance.

Install Java, Git, and Maven.

Run:

`sh`

```
CopyEdit

mvn clean package
```

9.Common Maven Commands:

`mvn clean` – Deletes compiled files.

mvn compile – Compiles the source code.

mvn package – Creates JAR/WAR files.

mvn install – Installs the package in the local repository.

10. Uses of Maven

- Automates project builds.
- implifies dependency management.
- standardizes the development process.