**Maven**

* It is a build automation tool. And it is used for project management hence it is also known as project management tool. It is open source tool.
* It is a software program that automates the process of transforming the source code into a deployable and executable format.
* Maven was developed by Apache company.
* Maven is used for java based projects or applications.

**Build tools:**

1.Java:Apache Maven, Apache ANT

2.Python:Pybuilder

3..NET:MSbuild(Microsoft build engine)

4.nodejs:Gulp,Grunt,Gradle Web pack

**Process:**

It converts Source code into war/ear/jar by maven

war→web applications

jar→java application

ear→Enterprise application

copying packages(ear/jar/war) to the tomcat is called deployment.

Developers will develop the code and they push into the GitHub then we need to build (maven) then it generate packages(ear/jar/war) then copying all the packages onto the web server. Then we will tell details to the end users

* Maven works with POM.XML(Latest version)/POM.XML2(Older version)
* POM(Project object model) is used for managing the application.
* It contains all the dependency libraries
* POM.XML should be unique.
* Each project contains one POM.XML file
* POM.XML is written by developers.
* POM.XML has super POM/parent POM.
* Maven consists plugins (Installing the dependencies is known as plugins).
* Dependencies are external features adding to our project.
* Added plugins are reusable
* Maven has life cycle.

**It has 3 Repositories’:**

1.Local repo:own laptops/systems

2.Central repo:repo which is available online

3.Remote repo:organizational repo

**Maven in devops should be utilized in 3 scenarios:**

1.If the initiative has number of significant dependencies.

2.If the dependency version needs to be up graded frequently.

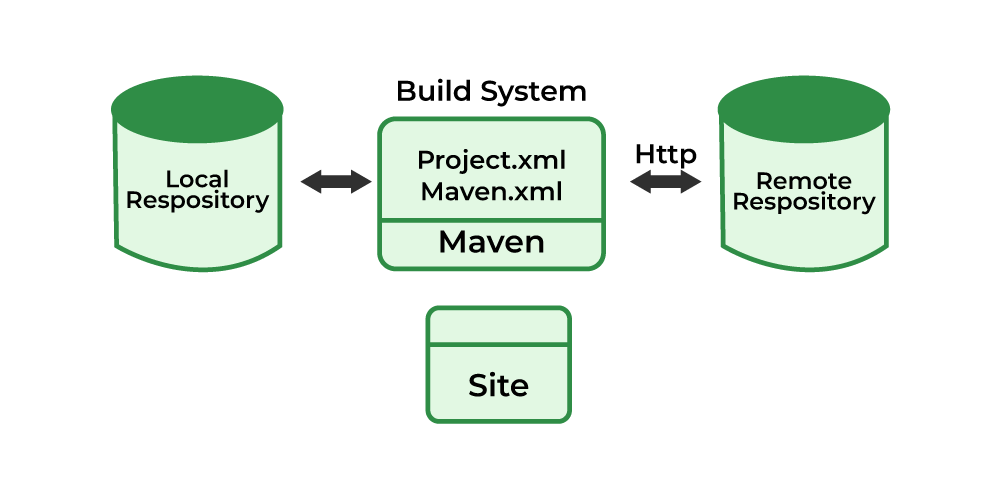
3.The task which involves rapid documentation,compilation and building of source code as jar or zip files.

**Apache ANT(older version of maven)**

* It is developed by apache company.
* It can build any kind of projects
* It doesn’t has life cycle.
* Build.xml is written by developers.
* Junit test cases not there in ANT.
* Scripts are not reusable

| **Apache maven** | **Apache ANT** |
| --- | --- |
| Maven has a standard project layout. | Ant does not have a standard project layout. |
| Maven has a predefined structure of project build. | We need to define everything from the directory, target, project, etc., in Ant |
| Dependencies need not be updated manually. | Dependencies need to be updated manually. |
| Maven is more preferred by users. | ANT is less preferred by users. |
| Maven has built-in plugins and commands to build the project. | Ant takes orders from developers and builds the project. |
| Maven scripts are simple. | ANT scripts are readable |
| ANT tasks can be used in Maven | Maven tasks cannot be used in ANT. |
| Maven has a pom.xml file. | Ant does not have a pom file, but it works with XML scripts. |
| For any customization, it uses pre-made plugins to add features easily. | You can customize and write your own tasks to do specific things. |
| It easier for common Java projects because much of it is automated. | It takes time to write and manage everything manually. |

**Architecture of Maven**



**Local Repository**: A folder on your computer where Maven stores dependencies (like libraries and plugins) that are already downloaded from remote repositories. It helps Maven avoid re-downloading dependencies and saves both time and bandwidth.

**Remote Repository**: A centralized online repository that holds all the required dependencies and plugins. When Maven cannot find a dependency in the local repository, it fetches it from the remote repository using HTTP.

**Build System**: The heart of Maven, which processes the pom.xml(Project Object Model) file. The build system helps define the project structure, dependencies, plugins, and the build lifecycle. It ensures that tasks such as compiling, testing, packaging, and deploying are executed in the correct order.

**Site**: Maven can generate project documentation and reports, such as test results and code analysis. These outputs can be hosted as a "site" for developers or stakeholders to access and review.

Maven has 2 types of life cycles.

**Maven lifecycle:**

It has 7 steps

1.validation:After creation of pom.xml.Downloading all the dependencies.

2.compile:Compile the source code

3.Test:Run junit test cases.

4.Packages:Create software package.That software provides war/jar/ear file

5.Verification:It verifies all the integrated test cases whether it is configured or not.

6.Installation:Copy the package to the local repository

7.Deployment:It upload the complete package to the remote repository.

**3 steps process of maven**

**1.Default:**Compile,validate,test

**2.clean:**pre clean(Before compilation),clean(while compilation),post clean(after compilation)-------->It deletes all unnecessary files.

**3.site:**It is like folder where we deploy our application.It generates comprehensive project,test results,figures are produced in the form of metrics(exact values).

**3 types of site:**

i.pre site

ii..site

iii.post site

For better understanding and collaboration between team members we use this site.

**Fields present in POM.XML file**

**1.groupid:**It contains client name or project name eg..Axis,ICICI

**2.artifactid:**feature names eg..personal loan,gold loan,credit card,debit card

**3.version id:**It consists of 3 fields:

Ex:8.2.0 here 8 is major 2 is minor and 0 is hot fix

**i.major releas**e:if 8 changes to 9 it means that project got releases.

**ii.minor release:**if 2 changes to 3 then new features are added to already exhausted version.

**iii.hot fix/bug fix:**it indicates that devlopers are still developing the code

**Differences between release and snapshot**

| **Releases** | **Snapshot** |
| --- | --- |
| It is a final version of a build which will not change. | It is under development stage or version build compile/artifacts can change. |
| Release looks like 8.2.0 only numerical values are present | Snapshot looks like 8.2.0-snapshot |
| Releases will maintain versions | It doesnot maintain any versions. |

**Note**:Maven command will always start will maven

Ex:mvn clean

Maven packages

**Maven installation steps:**

Take 1 EC2 Instance and login

2.Connect to linux environment

3.Install java

4.install git

5.clone the git url

6.cd filename →file should be pom.xml

7.mvn clean package→apart from war/jar/ear all other are deleted.and they are stored in target dictionary/folder.