1. **Explain about Maven.**

Maven provides the basic and advanced concepts of the Apache technology. It is a project management tool based on the POM(Project Object Model) used to build , dependency and documentation. Its like an ant but maven has more advantages than Ant. Both are build tools provided by Apache.

**Problems without Maven:**

1. Adding jars into every project: Set of jars needs to be added into project and dependencies also .
2. Building and deploying the projects : We have to build and deploy the project , so that it can work.
3. Creating the right project structure: we need to create the right project structure to get the project executed.

**Why we need maven:**

Maven overcome the above mentioned issues and it mainly does the following tasks,

* It simplifies the build process
* It provides the uniform build process
* It provides project information (such as log documentation, mailing list, unit test report,dependency list and cross referenced sources)
* It is easy to migrate the new features of maven

**Advantages:**

* Build
* Documentation
* Reports
* Releases
* Distribution

**What is build tool:**

Build tool is take care of everything to do the build process. It does the following

* Generate the source code
* Create the documentation front he generated source code(such as log documentation
* Compile the generated source code
* Package the complied code into zip or jar
* Deploy the packaged jar into local repository, server repository or central repository

**Difference between Ant and Maven:**

|  |  |
| --- | --- |
| **Ant** | **Maven** |
| It a build tool used for building the projects | it is the framework or tool used for project management |
| Less preferred | More preferred |
| Ant scripts are not reusable | Maven plugins are reusable |
| It is procedural. We need to specify the project information what to do when to do through the code in order in build.xml file | It is declarative. we need to define everything in pom.xml file |
| It doesn’t have formal conventions , so we need to provide the project structure in build.xml file | It has special conventions to place the source code, compiled code etc. So no need to define the project structure in pom.xml file |

**Maven Repository:**

Repository is directory of packaged jar file with POM.xml file. Maven has 3 types of repositories and dependencies are searched in the following order.

1. Local Repository :

Local repository is located in our localsystem. It is creaed by maven when executing any of the maven command

The default location of the local repository is %USER\_HOME%\.m2

Ex: C:\Users\vasantha.periyasamy\.m2

But we can change the local repository path in setting.xml file located in %MAVEN\_HOME%conf/

Contents of the Setting.xml

**<settings** xmlns="http://maven.apache.org/SETTINGS/1.0.0"

   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

   xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd"**>**

**<localRepository>**e:/mavenlocalrepository**</localRepository>**   // **change the local repository path here**

...

**</settings>**

1. Central Repository :

It is located on the web and created by maven community itself. The path of the central repository is <http://repo1.maven.org/maven2/>

The central repository has lots of libraries that can be searched by using the <http://search.maven.org/#browse>

1. Remote Repository :

It is located on the web and created by different vendors. There are some libraries which are missing in the central repository can be fetched from the remote repository. The remote repository need to be defined in the POM.xml file.

Adding the Junit library in POM.xml file

**<dependencies>**

**<dependency>**

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

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

**<version>**4.8.2**</version>**

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

**</dependency>**

**</dependencies>**

**Maven POM.XML :**

**POM** stands for the project management tool. It contains the information about the projects and configuration management information that is used by the maven to build the projects such as dependency information , build directory, source directory , test source directory, plugin and goals.

**Elements of the POM.xml:**

1. **Project :** It is the root elements f the pom.xml
2. **modelVersion :** it is the sub element of the project and it specifies the maven plugin version to be used
3. **groupid :** it is the sub element of the project. It is the unique id of the project group.
4. **Artifactid :** it is the unique id of the artifact. It is the sub element of the project. It is something either used or produced by the project
5. **version :** it is the sub element of the project. It specifies the version of the artifact to be produced or used.
6. **Packaging:** it specifies either jar or war
7. **Name :** it specifies the name of the peoject
8. **url:** it specifies the url of the project
9. **dependencies:** it is the sub element of the project and it specifies the dependency
10. **dependency:** it is the sub element of the dependencies and it specifies the individual dependency details
11. **scope:** it specifies the scope of the maven project. It can be compile, runtime, provided, test and system.

**Lifecycle event of the maven tool:**

The mvnpackage command complete the life cycle events of the project such as,

There are three types of lifecycle :

1) Build Life Cycle: (Default life cycle)

1. Validate – validate the project is correct and necessary information is available
2. Compile – compiling the source code
3. Test – testing the source code
4. Package – create WAR OR JAR as mentioned in POM.xml
5. Integration test – process and deploy the package into an environment where the integration tests run
6. Verify – verify the quality criteria
7. Install - install the package in local repository to be used as the dependency for other projects in local
8. Deploy – copies the final package to remote repository and share with other developers
9. Clean life cycle:

It has the following phases,

* Pre-clean
* Clean
* Post-clean

It cleans the output of the build by deleting the build directory . All these phases are invoked when we execute the maven clean goal

1. Site life cycle:

It has the following phases

* Pre-site
* Site
* Post-site
* Site-deploy

It mainly used for project documentation, deployment and reporting purposes. All these phases will be invoked by executing the maven site goal.

**Maven Plugins:**

Maven plugins are central part of the maven framework. There are 2 types of maven plugins

1. **Build plugins** : used at the time of build. It should be declared inside the <build> element
2. **Reporting plugins :** used at the time of site generation. It should be declared inside the <reporting> element.

**What is Maven Release ?**

* Maven release plugin is used to release project with maven. It save lot of repetitive and manual work to release a project.
* It has two steps.

1. Prepare :-
2. Perform :-