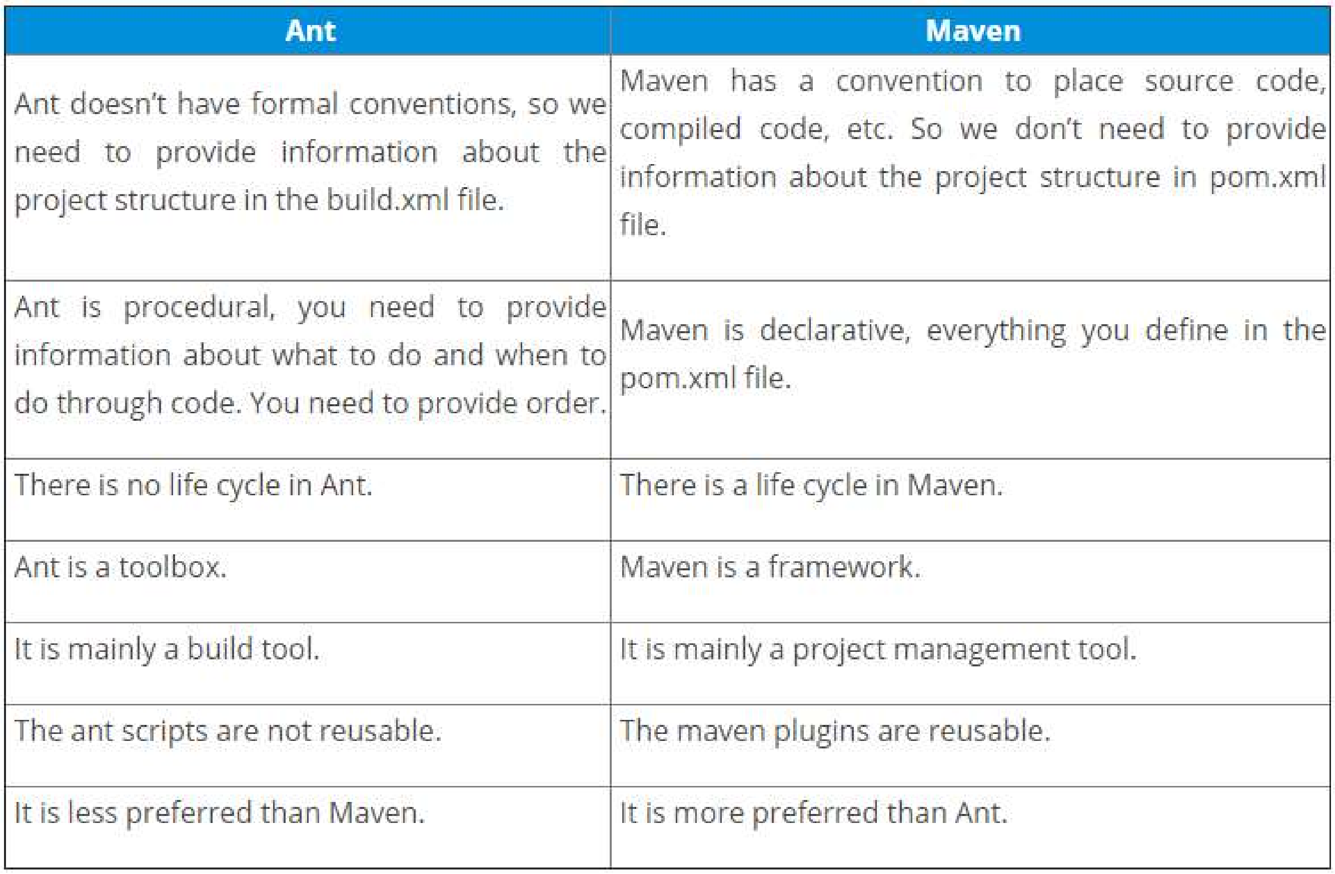
Self Online Training

Maven Interview Questions

1. List the differences between ANT and Maven.



Ant	Maven
Ant doesn't have formal conventions, so we need to provide information about the project structure in the build.xml file.	Maven has a convention to place source code, compiled code, etc. So we don't need to provide information about the project structure in pom.xml file.
Ant is procedural, you need to provide information about what to do and when to do through code. You need to provide order.	Maven is declarative, everything you define in the pom.xml file.
There is no life cycle in Ant.	There is a life cycle in Maven.
Ant is a toolbox.	Maven is a framework.
It is mainly a build tool.	It is mainly a project management tool.
The ant scripts are not reusable.	The maven plugins are reusable.
It is less preferred than Maven.	It is more preferred than Ant.

2. What is Maven?

Maven is a project management and comprehension tool. Maven provides developers a complete build lifecycle framework. The development team is easily able to automate the project's build infrastructure in almost no time as Maven uses

3. What are the main features of Maven?
Some of the main features of Maven are:

Simple to use: Maven provides easy project settings that are based on genuine practices.
Fast: You can receive a fresh project or module that began in fewer seconds in Maven.
Easy to learn: Maven usage and commands are pretty easy to learn across all projects. Therefore ramp-up time for new developers coming onto a project is very less.
Dependency management: Maven provides superior dependency management including automatic updates and transitive dependencies.
Multiple Projects: You can easily work on multiple projects at the same time by using Maven.
Huge Library: Maven has a large and growing repository of libraries and metadata to use out of the box.

4. What does the build tool?
Generates source code (if the auto-generated code is used)
Generates documentation from source code
Compiles source code
Packages compiled code into a JAR or ZIP file
Installs the packaged code in the local repository, server repository, or central repository

5. Why should one use Maven?
It aids in setting up the project very quickly and it avoids complicated build files like build.xml. the pom.xml file is at the core of Maven.POM.xml is a collection of dependencies of your Java Project which one can specify to Maven. After this Maven will download all of them from the internet and store it to some repository i.e. local repository, central repository, and remote repository.

application that repository will be used for any dependencies lookup. In this way, your deployment file will be very light.

6. Mention the steps for installing Maven on windows.
Maven can be downloaded and installed on windows, linux, and MAC OS platforms. To install Maven on windows, you need to perform the following steps:

Download Maven and extract it.
Add JAVA_HOME and MAVEN_HOME in the list of environment variables.
Add the environment path in Maven variable.
The last step is the verification of Maven by checking its version.

7. What does it mean when you say Maven uses Convention over Configuration?
In the case of Configuration, developers have to create the build processes manually and they have to specify each and every configuration in detail. But, Maven uses convention where the developers need not create the build processes manually.

Also, for convention, users do not need to specify the configuration in detail. Once a developer creates a project in Maven then Maven will automatically create a structure. Developers just have to place the files appropriately. There is no need to specify any configuration details in pom.xml file.

8. What are the aspects Maven manages?
Maven provides developers ways to manage following –

Builds
Documentation
Reporting
Dependencies
SCMs
Releases
Distribution

9. How do you know the version of mvn you are using?
Type the following command – mvn -version

10. What is POM?
POM stands for Project Object Model. In Maven, it is a fundamental Unit of Work and it is an XML file. You can find it in the base directory of the project. It consists of information about the project and various configuration details used by Maven to build the project(s).

11. What information does POM contain?
POM contains the following configuration information -

project dependencies
plugins
goals
build profiles
project version
developers
mailing list

12. What is Maven artifact?
An artifact is a file, normally a JAR that gets deployed to a Maven repository. A Maven build creates one or more artifacts, such as a compiled JAR and a source JAR. Each artifact consists of a group ID, an artifact ID, and a version string. The three together uniquely identify the artifact. A project's dependencies are specified as artifacts.

13. What is the Maven Build lifecycle?
A Build Lifecycle can be defined as a well-defined sequence of phases. It clearly defines the order in which the goals are to be executed. Each build phase contains a sequence of goals. If one life cycle is executed, all build phases in that life cycle are executed. If a build phase is executed, all build phases before it in the pre-defined

14. Name the 3 build lifecycle of Maven.
The three build lifecycles are –

clean: cleans up artifacts created by prior builds.

default: used to build the application.

site: generates site documentation for the project.

15. What is the command to build your Maven site?
Type the command – mvn site

16. What would the command mvn clean do?
This command deletes the target directory with all the build data before starting the build process.

17. What are the different phases of a Maven Build Lifecycle?
Following are the phases of Maven build lifecycle –

validate – validate the project and check if everything is correct and all necessary information is available.

compile – this phase compiles the source code of your project.

test – tests the compiled source code by using a suitable unit testing framework. These tests should not require the code to be packaged or deployed

package – takes the compiled code and packages it in its distributable format.

integration-test – processes and deploys the package if possible into an environment where integration tests can be run.

install – installation of the package into the local repository. This is done to use it as a dependency in other projects locally.

deploy – done in an integration environment or release environment. Here the final package is copied to the remote repository for sharing with other developers and projects.

18. What is a goal in Maven terminology?
A goal represents a specific task that contributes to the building and managing of a project. It is bound to zero or more build phases. A goal that is not bound to any build phase could be executed outside of the build lifecycle by invocating it directly.

19. What would the command mvn clean dependency:copy-dependencies package do?
This command will clean the project, copy the dependencies and package the project (executing all phases up to package).

20. What phases does a Clean Lifecycle consist of?
The clean lifecycle consists of the following phases –

pre-clean

clean

post-clean

21. What phases does a Site Lifecycle consist of?
The phases in Site Lifecycle are –

pre-site

site

site-deploy

22. What is Build Profile?
A Build profile is a set of configuration values that can be used to set or override default values of Maven build. Using a build profile, you can customize build for different environments such as Production and the Development environments.

23. What are different types of Build Profiles?
Build profiles are of three types –

Per Project – Defined in the project pom.xml file.

Per-User – Defined in Maven settings XML file (%USER_HOME%/.m2/settings.xml)

Global –Defined in Maven global settings xml file (%M2_HOME%/conf/settings.xml)

24. How can you activate profiles?
A Maven Build Profile can be activated in the following ways –

Explicitly using command console input.
Through maven settings.
Based on environment variables (User/System variables).
OS Settings (for example, Windows family).
Present/missing files.

25. What is a Maven Repository?
A repository is a place i.e. a directory where all the project jars, library jar, plugins or any other project specific artifacts are stored and this can be used by Maven easily.

26. What types of Maven repository?

Local: Maven local repository is a folder location that is present on your machine. It is created when you run any maven command for the first time. Maven local repository is a location where you can find your project's all dependencies (library jars, plugin jars etc).

Central: It is a repository provided by the Maven community. It contains a huge collection of commonly used libraries. When Maven does not find any dependency in local repository, it starts searching in central repository using the following URL: http://repo1.maven.org/maven2/.

Remote: Sometimes, Maven is not able to find a mentioned dependency in the central repository as well then it stops the build process and an output error message is displayed on the console. To avoid such a situation, Maven provides the idea of Remote Repository which is nothing but the developer's own custom repository containing required libraries or other project jars.

27. Can you tell me the default location of your local repository?
~/m2./repository.

28. Tell me the command to install JAR file in local repository.
mvn install

29. Is there a particular sequence in which Maven searches for dependency libraries?
Following is the search pattern –

Search for dependency in the local repository, if not found, move to step 2 else do the further processing.
Search for dependency in the central repository first, if not found and the remote repository is mentioned then move to step 4 else it is downloaded to the local repository for future reference.
If a remote repository has not been mentioned, Maven simply stops the processing and throws an error (Unable to find dependency).

processing and throws an error.

30. What are the uses of Maven Plugins?
Maven Plugins are used to –

create jar file.
create war file.
compile code files.
unit testing of code.
create project documentation.
create project reports.

31. What are the types of Maven Plugins?
Maven provides the following two types of Plugins –

Build plugins –They come into picture during the build and should be configured in the <build/> element of pom.xml

Reporting plugins –They get executed during the site generation and they should be configured in the <reporting/> element of the pom.xml

32. When does Maven use the External Dependency concept?
Maven dependency management uses the concept of Maven Repositories (Local, Central, Remote). Suppose dependency is not present in any of remote repositories and central repository then in such case Maven uses the concept of External Dependency.

33. What are the things that you must define for each external dependency?
External dependencies (library jar location) can be configured in pom.xml in same way as other dependencies are configured.

First, specify groupId the same as the name of the library.
Then specify artifactId the same as the name of the library.
Thirdly, specify scope as a system.

34. What is Archetype?
An archetype is a Maven plugin whose task is to create a project structure as per its template.

35. What is the command to create a new project based on an archetype?
Type the following command - mvn archetype:generate

36. What is SNAPSHOT in Maven?
SNAPSHOT can be defined as a special version that indicates a current development copy. Unlike the regular versions, Maven checks for a new SNAPSHOT version in its remote repository. Maven does it for every build.

37. What is the difference between Snapshot and Version?
In the case of Version, if Maven once downloads the mentioned version say data-service:1.0, it will never try to download a newer 1.0 available in the repository. To download the updated code, the data-service version is then upgraded to 1.1.

In the case of SNAPSHOT, Maven will automatically fetch the latest SNAPSHOT (data-service:1.0-SNAPSHOT) every time the team builds its project.

38. What is a transitive dependency in Maven?
Transitive dependency is nothing but to avoid the need to discover and specify the libraries that your own dependencies require, and including them automatically.

39. What does dependency management mean with respect to transitive dependency?

It simply means to directly specify the versions of artifacts to be used when they are encountered in transitive dependencies. For example project C can include B as a dependency in its dependencyManagement section and directly control which version of B is to be used when it is ever referenced.

If you find two dependency versions at the same depth in the dependency tree, then you use the first declared dependency. This is nothing but dependency mediation.

41. What is the dependency scope? Name all the dependency scope.
Dependency scope typically includes dependencies as per the current stage of the build. The various Dependency scopes are –

compile – This scope indicates that dependency is available in the classpath of the project. It is the default scope.

provided – This indicates that the dependency is to be provided by JDK or web-Server/Container at runtime.

runtime – This scope tells that you dont need dependency is for compilation but you need it for for execution.

test – This scope states that the dependency is only available for the test compilation and execution phases.

system – This scope indicates that you must provide the system path.

import – This scope is only used when the dependency is of type pom. This scope tells that the specified POM should be replaced with the dependencies in the POM's <dependencyManagement> section.

42. What is the minimal set of information for matching dependency reference against a dependencyManagement section?
{groupId,artifactId,type,classifier}.

43. Is it possible to refer a property defined in your pom.xml file?
To refer a property that is defined in your pom.xml, the property name uses the names of the XML elements that define the value, with "pom" being allowed as an

project version, ${pom.build.finalName} refers to the final name of the file created

during the built project packaging etc.
NOTE: This is a frequently asked Maven Interview Question that you must know.

44. What are the default values for the packaging element? If there is no packaging element defined? What is the default value for that?
The valid packaging values include jar, war, ear, and pom. If no packaging value has been specified, then by default it will be a jar.

45. What is the use of the execution element in pom file?
The <execution> element contains the information required for the execution of a plugin.

46. What is a project's fully qualified artifact name?
<groupId>:<artifactId>:<version>

47. If you fail to define any information, where does your pom inherits that information from?
All POMs are inherited from a parent despite explicitly defined or not. This base POM is called Super POM and it contains values that are inherited by default.

48. How profiles are specified in Maven?
Profiles are specified by making use of a subset of the elements that are available in the POM itself.

49. What are the elements in POM that a profile can freely modify when specified in the POM?
<repositories>, <pluginRepositories>,<dependencies>, <plugins> ,<properties>, <modules><reporting>,<dependencyManagement>,<distributionManagement>

50. What are the benefit of storing JARS/external dependencies in the local repository instead of a remote one?
It uses less storage and also makes checking out a project quicker, without the need

51. What is a system dependency?
Dependency with scope system is always available and is not looked up in the repository, they are usually used to tell Maven about dependencies that are provided by the JDK or the VM. Thus, system dependencies are especially useful for resolving dependencies on artifacts that JDK normally provides.

52. What is the use of optional dependency?
You can mark any transitive dependency as optional using the "optional" element. As an example, A depends upon B and B depends upon C. Now B marked C as optional. Then A will not use C.

53. What is dependency exclusion?
You can exclude any transitive dependency using the "exclusion" element. Suppose if A depends upon B and B depends upon C then A can be marked C as excluded.

54. How to run the clean plugin automatically during the build?
For this, you just need to place the clean plugin inside the execution tag in pom.xml file.

55. What is the meaning of the message "You cannot have two plugin executions with the same or missing elements"?
It simply means that you have executed a plugin multiple times with the same <id>. To correct this you just need to provide each <execution> with a unique <id>.

Like

Be the first to like this.

Self Online Training /