Maven Fundamentals

- **Explain what is Maven? How does it work?**

  Maven is a popular open source build tool for enterprise Java projects, designed to take much of the hard work out of the build process. Maven uses a declarative approach, where the project structure and contents are described, rather then the task-based approach used in Ant or in traditional make files, for example. This helps enforce company-wide development standards and reduces the time needed to write and maintain build scripts.

- **Explain what is POM and its significance**

A Project Object Model or POM is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects. Examples for this is the build directory, which is `target`; the source directory, which is `src/main/java`; the test source directory, which is `src/test/java`; and so on. When executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal.

Some of the configuration that can be specified in the POM are the project dependencies, the plugins or goals that can be executed, the build profiles, and so on. Other information such as the project version, description, developers, mailing lists and such can also be specified.

- **Explain what a Maven artifact is?**

An artifact is a file, usually a JAR, that gets deployed to a Maven repository.A Maven build produces one or more artifacts, such as a compiled JAR and a "sources" JAR.Each artifact has a group ID (usually a reversed domain name, like com.example.foo), an artifact ID (just a name), and a version string. The three together uniquely identify the artifact.A project's dependencies are specified as artifacts.

- **List out the dependency scope in Maven?**

  Scopes and configurations are used by each tool to define dependencies and how they affect different classpaths, such as the compilation and runtime classpaths. Maven defines 6 scopes: compile, runtime, provided, system, test, and import

Maven Fundamentals

- **List out what are the build phases in Maven?**

Phase	Handles	Description
prepare-resources	resource copying	Resource copying can be customized in this phase.
validate	Validating the information	Validates if the project is correct and if all necessary information is available.
compile	compilation	Source code compilation is done in this phase.
Test	Testing	Tests the compiled source code suitable for testing framework.
package	packaging	This phase creates the JAR/WAR package as mentioned in the packaging in POM.xml.
install	installation	This phase installs the package in local/remote maven repository.
Deploy	Deploying	Copies the final package to the remote repository.

- **Mention the three build lifecycle of Maven?**

There are three built-in build lifecycles: default, clean and site. The **default** **lifecycle** handles your project deployment, the **clean** **lifecycle** handles project cleaning, while the **site** **lifecycle** handles the creation of your project's site documentation.

- **List out what are the aspects does Maven Manages?**

Maven handles following aspects,

- Build.
- Documentation.
- Reporting.
- Dependencies.
- SCMs.
- Releases.
- Distribution.

Maven Fundamentals

- Mailing list

➢ Explain what a Maven Repository is? What are their types?

A repository in Maven holds build artifacts and dependencies of varying types.

There are exactly two types of repositories: **local** and **remote**:

- the **local** repository is a directory on the computer where Maven runs. It caches remote downloads and contains temporary build artifacts that you have not yet released.
- **remote** repositories refer to any other type of repository, accessed by a variety of protocols such as file:// and https://. These repositories might be a truly remote repository set up by a third party to provide their artifacts for downloading (for example, repo.maven.apache.org). Other "remote" repositories may be internal repositories set up on a file or HTTP server within your company, used to share private artifacts between development teams and for releases.

➢ Explain how you can exclude dependency?

- Open the dependency POM and find the transitive dependency you want to exclude. Copy groupId and artifactId.
- In your project POM, underneath your active dependency, enter exclusions and using code completion paste the copied info of the dependency you want to exclude.

➢ For POM what are the minimum required elements?

The minimum requirement for a POM are the following:

- project root
- modelVersion - should be set to 4.0.0
- groupId - the id of the project's group.
- artifactId - the id of the artifact (project)
- version - the version of the artifact under the specified group