1) what is maven?

*Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework.
*Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.
*Maven can set-up the way to work as per standards in a very short time.
*As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.
* It allows developers to create projects, dependency, and documentation using Project Object Model and plugins.

2) what is usage of Maven?

*Maven can add all the dependencies required for the project automatically by reading pom file.
*One can easily build their project to jar,war etc. as per their requirements using Maven.
*Maven makes easy to start project in different environments and one doesn’t needs to handle the dependencies injection, builds, processing, etc.
*Adding a new dependency is very easy. One has to just write the dependency code in pom file.

3) How to create a maven project?

*From the File menu, select New > Project.
*The New Project screen opens.
*Expand Maven, select Maven Project, and click Next.
*The New Maven project wizard opens.
*Leave the default, Use default Workspace location box selected and click Next.
*The Select an archetype page opens.
*click Add Archetype and supply the following values:
*Archetype Group Id: com.lightbend.lagom
*Archetype Artifact Id: maven-archetype-lagom-java
*Version: The Lagom version number. Be sure to use the current stable release.
*Repository URL: Leave blank
*Click OK.
*The next page of the wizard opens, providing fields to identify the project and displaying the hello and stream properties from the archetype.
*To identify your project, enter the following:
*Group Id - Usually a reversed domain name, such as com.example.hello.
*Artifact Id - Maven also uses this value as the name for the top-level project folder. You might want to use a value such as my-first-system.
*Version - A version number for your project.
*Package - By default, the same as the groupId.
*Click Finish and the projects created by the archetype display in the Package Explorer.
*Run the project:
*Right-click the parent project folder.
*Eclipse puts all of the Maven project folders at the same level, so be sure to select the correct one. For example, if you used my-first-system as the Maven artifact ID, right-click my-first-system.
*Select Run as … > Maven Build.
*In the Goals field, enter lagom:runAll.
*Select the JRE tab and make sure it is pointing at a JRE associated with a JDK.
*Click Run.

4) what commands in maven? and what is the purpose of maven?

*Maven Commands and purpose:

1.mvn --version =Prints out the version of Maven you are running.

2.mvn clean = Clears the target directory into which Maven normally builds your project.

3.mvn package = Builds the project and packages the resulting JAR file into the target directory.

4.mvn package -Dmaven.test.skip=true -> Builds the project and packages the resulting JAR file into the target directory - without running the unit tests during the build.

5.mvn clean package = Clears the target directory and Builds the project and packages the resulting JAR file into the target directory.

6.mvn clean package -Dmaven.test.skip=true -> Clears the target directory and builds the project and packages the resulting JAR file into the target directory - without running the unit tests during the build.

7.mvn verify = Runs all integration tests found in the project.

8.mvn clean verify = Cleans the target directory, and runs all integration tests found in the project.

9.mvn install = Builds the project described by your Maven POM file and installs the resulting artifact (JAR) into your local Maven repository

10.mvn clean install -Dmaven.test.skip=true =Clears the target directory and builds the project described by your Maven POM file without running unit tests, and installs the resulting artifact (JAR) into your local Maven repository.

11.mvn dependency:copy-dependencies =Copies dependencies from remote Maven repositories to your local Maven repository.

12.mvn clean dependency:copy-dependencies = Cleans project and copies dependencies from remote Maven repositories to your local Maven repository.

13.mvn clean dependency:copy-dependencies package =Cleans project, copies dependencies from remote Maven repositories to your local Maven repository and packages your project.

14.mvn dependency:tree =Prints out the dependency tree for your project - based on the dependencies configured in the pom.xml file.

15.mvn dependency:tree -Dverbose =Prints out the dependency tree for your project - based on the dependencies configured in the pom.xml file. Includes repeated, transitive dependencies.

16.mvn dependency:tree -Dincludes=com.fasterxml.jackson.core -> Prints out the dependencies from your project which depend on the com.fasterxml.jackson.core artifact.

17.mvn dependency:tree -Dverbose -Dincludes=com.fasterxml.jackson.core -> Prints out the dependencies from your project which depend on the com.fasterxml.jackson.core artifact. Includes repeated, transitive dependencies.

18.mvn dependency:build-classpath = Prints out the classpath needed to run your project (application) based on the dependencies configured in the pom.xml file.