[Maven - Setup, Build and Deploy](javascript:void(0);)

**Objectives**

* Explain the need and benefits of Maven
  + Managing dependent libraries, building and deploying project, generate documents from source code, compile source code, create package (jar or war)
    - Ref - https://www.javatpoint.com/maven-tutorial
* Setup maven for command line
  + Zip file extraction, environment variables JAVA\_HOME, MAVEN\_HOME, PATH, mvn -version
    - Ref - https://www.mkyong.com/maven/how-to-install-maven-in-windows/
* Create, build and deploy a maven project
  + archetypes, proxy environment variables, group id, artifact id, central maven repository in internet, local repository location, pom.xml, maven architecture, dependency tree
    - Ref - https://maven.apache.org/guides/introduction/introduction-to-archetypes.html
* Explain the maven build lifecycle
  + clean, validate, compile, test-compile, test, package, install, deploy, goal hierarchy
    - Ref - https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html
* Explain the various aspects available in pom.xml
  + groupId, artifactId, version, packaging, dependencies, dependency scope (compile, provided, runtime, test, system), properties, profiles
    - Ref - https://maven.apache.org/pom.html
* Explain the importance of profiles in maven
  + Types of Build Profiles, naming environment specific resources, setting profile in maven settings.xml, profile activation via environment variables
    - Ref - https://www.tutorialspoint.com/maven/maven\_build\_profiles.htm

**Install Maven in Desktop**   
Follow instructions below to setup maven in the desktop:

1. Download latest version of maven from internet or get the installation zip file from trainer
2. Extract the zip file into D: drive
3. Set the windows environment variables JAVA\_HOME, MAVEN\_HOME
4. Include PATH with MAVEN\_HOME\bin
5. Test maven setup with ‘mvn -version’ command in command line

**Create web project using Maven**   
Follow instructions below to create a web project using Maven:

1. Open command line
2. Change the folder to the root folder of the Eclipse workspace
3. Create the maven web project structure.

mvn archetype:generate DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4 -DinteractiveMode=false

1. Build the project using the following command:

mvn clean package

**Explanation**

1. Java libraries are available in central maven repository in internet ([https://repo.maven.apache.org](https://repo.maven.apache.org/))
2. Maven downloads dependent jar files from the central maven repository and places them in C:\Users\[EMP\_ID]\.m2\repository folder. This is the local maven repository.
3. This transfer is a one-time activity and download happens only when new dependencies are added in pom.xml.

Refer diagram in Maven Repositories topic of <http://tutorials.jenkov.com/maven/maven-tutorial.html> for understanding the Maven Architecture.

**Setup Maven project in Eclipse**   
Follow steps below to setup the Maven project in Eclipse.

1. Import the maven project folder in previous hands on in Eclipse (File > Import > Maven > Existing Maven Projects and select the new maven project folder)
2. Make following change in pom.xml
   1. Change packaging as war <packaging>war</packaging >
   2. Change compiler version from 1.7 to 1.8
3. Add Tomcat Server Environment as Library (Right Click Project > Build Path > Configure Build Path > Add Library > Server Runtime > Apache Tomcat v9.0)
4. Add the project to Tomcat Server by right clicking the Tomcat Server in Servers perspective and use "Add and Remove" option.
5. Start the server and check if the application is running.

**Understanding Maven Build Cycle and Goals**   
Execute the following commands in the order specified below.  
After executing each command, check the content created in target folder.

|  |  |
| --- | --- |
| **Command** | **Expection action by Maven** |
| mvn clean | Deletes the target folder |
| mvn clean validate | Validates if there is not issues to initiate build |
| mvn clean compile | Compiles and creates the classes from main folder |
| mvn clean test-compile | Compiles and creates the classes from test folder |
| mvn clean test | Executes the unit test cases |
| mvn clean package | Creates the JAR/WAR based on the configuration in pom.xml |
| mvn clean install | Copies the JAR to local maven repository |
| mvn clean deploy | This will fail since the remote server details are not configured. |