**Jenkins Maven Build Nexus Upload**

This document explains about the how the Jenkins automates the Maven build and upload the artifacts to Nexus repository.

About Maven Project:

----------------------------

The maven project has below folder structure and it has pom.xml file to define the configuration details about the artifact.

my-app

|-- pom.xml

`-- src

|-- main

| `-- java

| `-- com

| `-- mycompany

| `-- app

| `-- App.java

`-- test

`-- java

`-- com

`-- mycompany

`-- app

`-- AppTest.java

The src/main/java directory contains the project source code, the src/test/java directory contains the test source, and the pom.xml file is the project's Project Object Model, or POM.

Maven LifeCycle Phases:

--------------------------------

**Maven Phases**

Although hardly a comprehensive list, these are the most common *default* lifecycle phases executed.

* **validate**: validate the project is correct and all necessary information is available
* **compile**: compile the source code of the project
* **test**: test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
* **package**: take the compiled code and package it in its distributable format, such as a JAR.
* **integration-test**: process and deploy the package if necessary into an environment where integration tests can be run
* **verify**: run any checks to verify the package is valid and meets quality criteria
* **install**: install the package into the local repository, for use as a dependency in other projects locally
* **deploy**: done in an integration or release environment, copies the final package to the remote repository for sharing with other developers and projects.

There are two other Maven lifecycles of note beyond the *default* list above. They are

* **clean**: cleans up artifacts created by prior builds
* **site**: generates site documentation for this project

Note: In the above we can use only the below command to build the artifact.

**“ maven clean build”**

Jenkins Pipeline contains the multiple below sections.

1. Configuration Section
2. Environment Variable Declaration
3. Jenkins Initialization stage
4. Upload artifact to nexus

Configuration Section

----------------------------

Here we have defined about the Agent details where our task is going to execute. Creating a pod with docker image and it has the java, maven installed in it.

Environment Variable Declaration:

------------------------------------------------

In this section we have defined the required environment variable and it hold the below details.

1. Nexus url
2. Java nexus repo name
3. Jenkins Credentials to connect to Nexus repository.

Jenkins Initialization Stage:

-----------------------------------

It will clone our java repository into the docker container to use for further stages.

Maven Build & Upload Artifact to Nexus:

-----------------------------------------------------

The following command used to create maven artifact and the artifact file will be generated based on the below values in pom.xml file.

**<groupId>org.apache.maven</groupId>**

**<artifactId>super-pom</artifactId>**

**<version>4.0.0-SNAPSHOT</version>**

There are two type of artifacts will be there.

* 1. Release artifact file

If you define the below tag in pom.xml will call it as Release version.

**<version>4.0.0-SNAPSHOT</version>**

* 1. SNAPSHOT artifact file

If you define like below it is called as SNAPSHOT release .

**<version>4.0.0-SNAPSHOT</version>**

To upload the build artifact to NEXUS we can use different methods available but we are using linux “CURL” command here because our nexus is exposed as https url.

The below command is used to upload the artifact to nexus.

curl -kv -u ${NEXUS\_CREDENTIALS\_USR}:${NEXUS\_CREDENTIALS\_PSW} \

-F "maven2.generate-pom=false" \

-F "maven2.asset1=@pom.xml" \

-F "maven2.asset1.extension=pom" \

-F "maven2.asset2=@target/$JARFILE;type=application/java-archive" \

-F "maven2.asset2.extension=jar" \

${NEXUS\_URL}/service/rest/v1/components?repository=${NEXUS\_REPOSITORY}