**Artifactory OSS Install in Linux using Docker**

$ docker run -p 8081:8081 -p 8082:8082 -d releases-docker.jfrog.io/jfrog/artifactory-oss:latest

# BROWSE - http://15.207.221.219:8082

# Username/Password - admin/password //new pass: Welcome@123

What is Artifactory?

🡪Repository mgmt. tool

🡪package mgmt..

[What](https://www.jfrog.com/confluence/pages/viewpage.action?pageId=100110787) kind of Package:

<https://www.jfrog.com/confluence/pages/viewpage.action?pageId=100110787>

Who developed artifactory: Company name is🡪Jfrog

Its written in Java

Difference Between OSS – Pro:

OSS: Minimul package support. Its free

Professional – all package support .Its paid

<https://www.jfrog.com/confluence/display/JFROG/Artifactory+Comparison+Matrix>

Why do we need it?

* Without jfrog we have to store packages on shared drive.
* In DevOps we are generating lots of packages.
* As we more MS we are storing large number of package.
* Automated Package versioning
* We are using 3rd party library.. we are storing that library in jfrog artifactory
* Integration with DevOps Tool
* Roles Bassed access mgmt.
* Package security
* Easy Clean up
* Backup and Archieve
* Centralized store
* Other tools are Nexus, archieva

Why we are not using Github to store artifact

GitHub – We are storing code.

Artifactory- Binary , yum repo, apt repo. Its packages.. docker repo, maven repo

How to Install it:

**Artifactory Pro Install in Linux Manually**

$ cd /opt/

$ wget https://releases.jfrog.io/artifactory/artifactory-pro/org/artifactory/pro/jfrog-artifactory-pro/7.23.3/jfrog-artifactory-pro-7.23.3-linux.tar.gz

$ tar -zxvf jfrog-artifactory-pro-7.23.3-linux.tar.gz

$ cd jfrog-artifactory-pro-7.23.3

$ cd app/bin

$ ./artifactory.sh start

$ ./artifactory.sh status

Using default router's certificate and private key

router is running (PID: 10621)

metadata is running (PID: 10767)

event is running (PID: 10878)

frontend is running (PID: 10994)

Artifactory is running, on pid=11080

# BROWSE - http://15.207.221.219:8082

# Username/Password - admin/password

How Does Build Promotion Work?

https://jfrog.com/knowledge-base/how-does-build-promotion-work/

Workflow in artifactory:

Will create two repo – maven and docker

Local repo – Read and Write

Remote repo – Read only copy of

Group Repo /Virtual- Collection of local and remote repo

**Maven Release Type:**

-Release

-snapshot

A "release" is **the final build for a version which does not change**. A "snapshot" is a build which can be replaced by another build which has the same name. It implies that the build could change at any time and is still under active development. You have different artifacts for different builds based on the same code

<https://repo.maven.apache.org/>

<http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-virtual-group-repo/>

<http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-int-local-release/>

<http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-int-local-snapshot/>

<http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-remote-repo/>

User and roles:

<http://18.225.37.142:8082/ui/login/>

Modify POM.xml add distribution element which contain address to be deployed.

<distributionManagement>

**<repository>**

<id>kishor</id>

<name>Internal Release</name>

<url><http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-int-local-release> </url>

**</repository>**

**<snapshotRepository>**

<id>kishor</id>

<name>snapshot softx</name>

<url><http://ec2-3-22-172-90.us-east-2.compute.amazonaws.com/artifactory/maven-int-local-snapshot/></url>

**</snapshotRepository>**

</distributionManagement>

Modify settings.xml and add server element of accessing repo..

<server>

<id>kishor</id> 🡪check from distribution mgmt. from pom file. It should be same

<username> admin</username>

<password>Welcome@123</password>

</server>

<http://3.22.172.90/> 🡪 use ip instead of public dns name

Run command mvn deploy

<http://ec2-18-225-37-142.us-east-2.compute.amazonaws.com/artifactory/maven-virtual-group-repo/>

**Avoid custom of base url**

OLD New

Web ui PORT|Upload port web ui PORT|Upload Port

8081| 8082|8081

AKCp8kqMTK3yZTomdf3hKcPkyrCxFMsZ7wrhXWG5yv6u7fBzeHzrs3d7pxRnzKZyNM6ybgYAh

How to download package using artifactory in maven?

1. Create a group repo which contain all the repo from where you are downloading package..

<http://3.22.172.90:8082/artifactory/maven-virtual-group-repo/>

1. Locate which package you want to download update pom

<dependencies>

<dependency>

<groupId>…</groupId>

<artifactId>..</artifactId>

<version>1.0</version>

</dependency>

</dependencies>

1. Delete package from local repository
2. Modify settings.xml and add mirror element with group repo

<mirror>

<id>kishor</id>

<name></name>

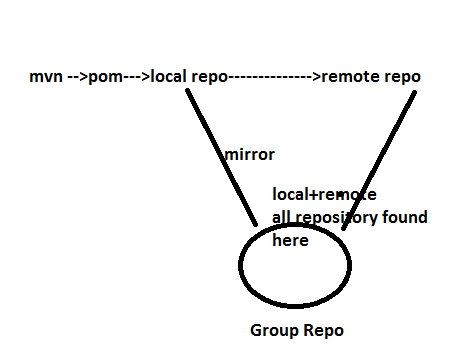
<url>[http://3.22.172.90:8081/artifactory/maven-virtual-group-repo/</url](http://3.22.172.90:8081/artifactory/maven-virtual-group-repo/%3c/url)>

<mirrorOf>\*</mirrorOf>

</mirror>

Why mirror? Every time maven looking package in local and remote repo.. If we activate mirror its looking for package in mirror repo

 Mirror is **used to redirect all traffic to a specific repository URL and block everything else**



<https://www.baeldung.com/maven-settings-xml>

We can use mirrors in cases where we want to **define** **an alternative mirror for a particular repository**. This overrides what's in the pom.xml.

For example, we can force Maven to use a single repository by mirroring all repository requests:

Defining repositories in the project pom.xml is a good practice. However, we shouldn't put security settings, such as credentials, into our source code repository with the pom.xml. Instead, we **define this** **secure information in the settings.xml file**

We should note that the ID of the server in the settings.xml needs to match the ID element of the repository mentioned in the pom.xml. The XML also allows us to use placeholders to pick up credentials from environment variables.

1. mvn package