

What are artifact repository manager ?

Set up artifact repository manager ? Nexus on cloud server digital ocean .

Artifact : app build into single file , different artifact format jar,war , tar , zip etc.

Artifact repository : where u store those artifact .

Artifact repo need to support these septic type of format .

Repository of each file/artifact type .

They produce different type of artifact eg : java , python , .net etc.

U need different repository for them .

Different software need for each repository > just one application managing all .

And that application is artifact repository manager .(nexus , j frog etc).

It allow u to store or build different artifacts ,retrieve artifact later ,central storage .

Public repository manager : library /framework u you as dependencies :

Eg: maven public repository : for java/jar files that are public available , to make something publicly available I can upload it int maven central repo.

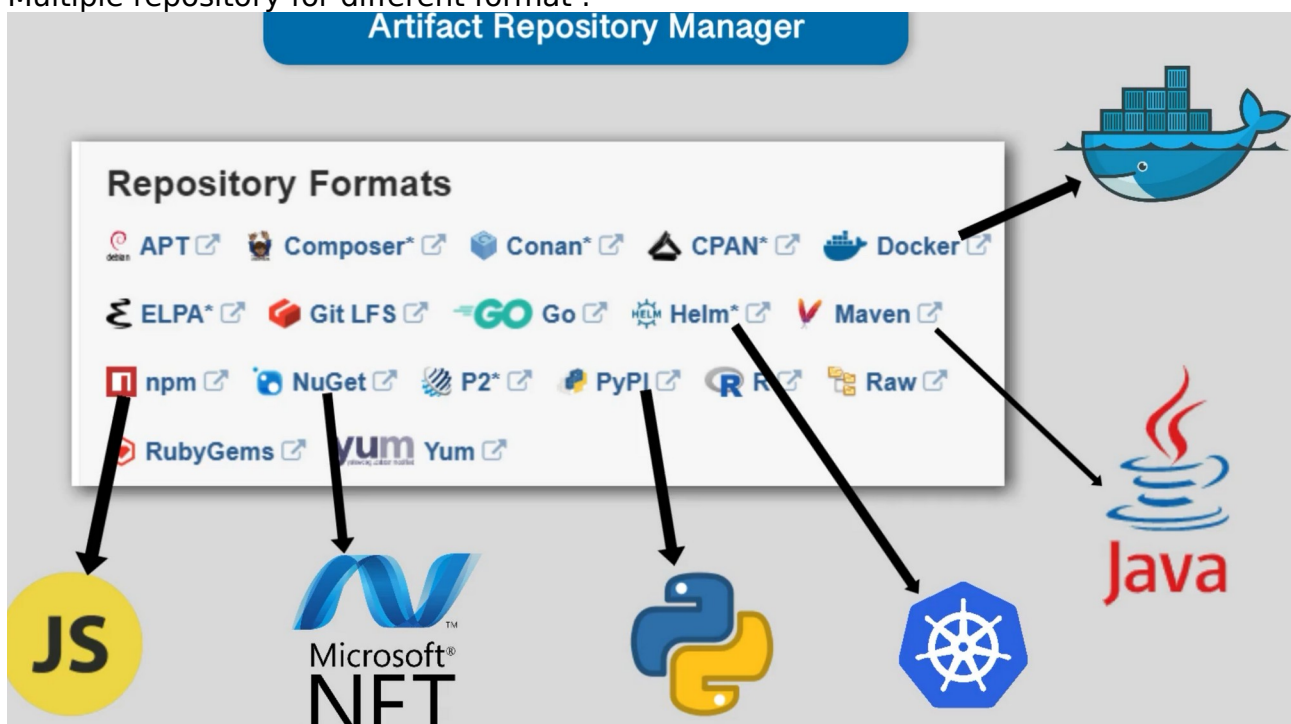
N pm repo manager : for javascript :

Using nexus u can host your own repository : maven or n pm etc . To share the artifact used within the company .

Proxy repo : so whether u r using company or central repo it is is nexus

Open source and commercial version are available in nexus .

Multiple repository for different format .



FEATURES of repository manager :

Integrate with IDAP : configure access management .

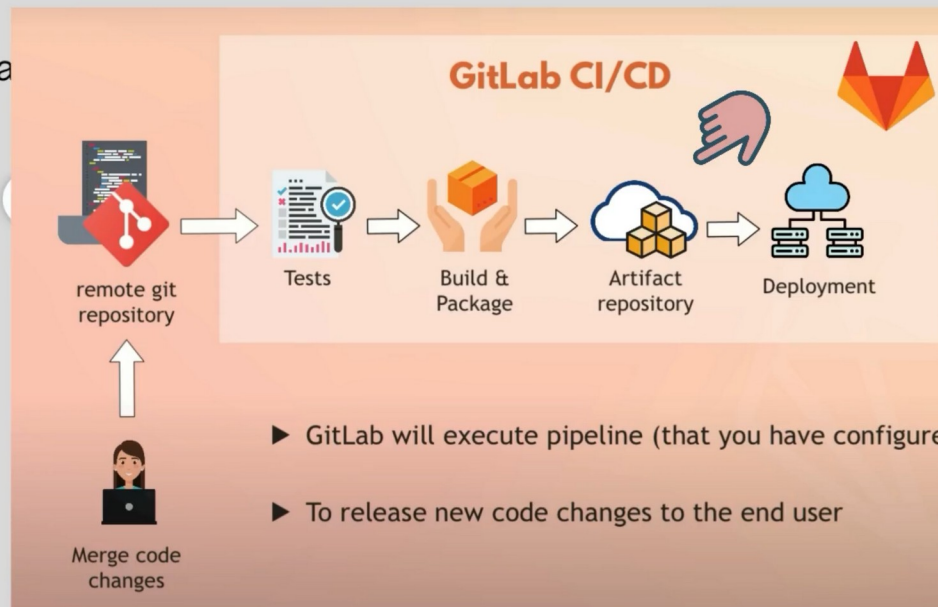
Flexible and powerful rest API for integration with other tools .

Why not manually managed !/

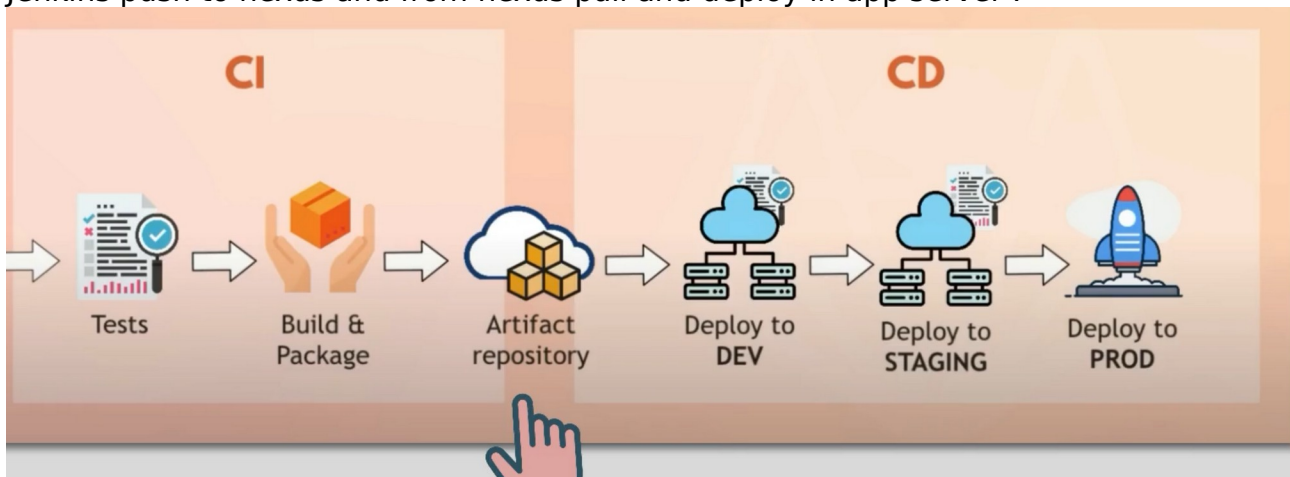
Features of Repository Manager

👍 Integrate with LDAP

👍 Flexible a



Jenkins push to nexus and from nexus pull and deploy in app server .



Backup and restore.

Multiple formate support : docker , zip, jar , tar etc.

Metadata tagging : labelling and taggin artifact . Maintain version

Clean up policies : every commit to the master branch create new artifact of the application this will result in generation many artifact by nexus : nexus will run out of storage space so clean up policies are important :like automatically clean up old artifact .

Search functionality :

User token support for system user non human system authentication : as nexus is a tool u don't use manually most of the time its for integration for other technology

Installing nexus on digitalocean cloud:

Prerequisite of installing nexus is to having java versions 8

First move in to cd/opt folder from root.

And we gonna download nexus package from this folder .

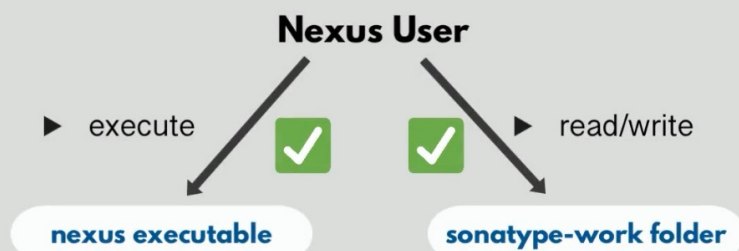
Search for nexus download from sonar type .

Paste the tar file of nexus

Now untar the downloaded package .

```
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# ls
digitalocean nexus-3.53.0-01-unix.tar.gz
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# tar -zxvf nexus-3.53.0-01-unix.tar.gz
```

```
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# ls -l
total 205600
drwxr-xr-x  4 root root      4096 May  3 16:07 digitalocean
drwxr-xr-x 10 root root      4096 May  3 16:16 nexus-3.53.0-01
-rw-r--r--  1 root root 210520190 May  2 21:37 nexus-3.53.0-01-unix.tar.gz
drwxr-xr-x  3 root root      4096 May  3 16:16 sonatype-work
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# chown -R nexus:nexus nexus-3.53.0-01
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# chown -R nexus:nexus sonatype-work
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# ls -l
total 205600
drwxr-xr-x  4 root root      4096 May  3 16:07 digitalocean
drwxr-xr-x 10 nexus nexus    4096 May  3 16:16 nexus-3.53.0-01
-rw-r--r--  1 root root 210520190 May  2 21:37 nexus-3.53.0-01-unix.tar.gz
drwxr-xr-x  3 nexus nexus    4096 May  3 16:16 sonatype-work
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt#
```



Work with nexus : set up users and permissions , different repo types and format.
Different btw components vs assists , publish artefacts to nexus first using maven and then gradle as a build tool

Configure cleanup policies. Interact with nexus rest api

First : nexus need bigger size server >
8gb memory 4 cpus 160 gb ssd 5tb transfer

Add firewall rule setup port ssh 22 port and install java 8(nexus needs).

Cd /opt : move in this folder .

/opt# wget pastTheLinkFromSonaTypeNexusTarFileDownload.

/opt# ls

Now untar file : /opt# tar -zxvf nexus-3.53.0-1-unix.tar.gz

/opt# ls : digitalocean nexus-3.53.0-01 nexustar and sonatype-work

Nexus folder: contains runtime and application of nexus ;;bin etc lib public system

Sonatype-work : contain own config for nexus and data . Subdirectories depending on your nexus configuration , u can install plugins directory , also contains ip address that accessed nexus , logs of nexus app , your uploaded fils and meta data , you can use this folder for backup ,

Starting nexus :

Services should not run with root user permissions

Best practice : create own user for service (eg : nexus)

Only the permission for the specific service .

/opt# adduser nexus

Ls -l // to check the permissions of nexus

```
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt# ls -l
total 205600
drwxr-xr-x  4 root root      4096 May  3 16:07 digitalocean
drwxr-xr-x 10 root root      4096 May  3 16:16 nexus-3.53.0-01
-rw-r--r--  1 root root 210520190 May  2 21:37 nexus-3.53.0-01-unix.tar.gz
drwxr-xr-x  3 root root      4096 May  3 16:16 sonatype-work
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt#
```

We need to change both of these directory permissions to the Nexus user should have both permission : nexus executable (executable) , sonatype-work folder (read,write) .

Now we have to change the own of the directory using :

chown -R(recursively) nexus:nexus nexus-3.53.0-01

chown -R(recursively) nexus:nexus sonatype-work

Ls-l

Set nexus config so that it runs nexus user : /opt# vim nexus-3.53.0-01/bin/nexus.rc

Run_as_user="nexus"

Switch from root to nexus user :

Su - nexus

Start nexus service :/opt/nexus-3.53.0-01/bin/nexus start

Starting nexus

Ps aux | grep nexus : to make sure it is running

Netstat -lnpt

To access nexus service from browser we need to open port 8081

So add firewall configuration for port 8081 to open in bound connection

:type custom protocol tcp port range 8081 from all sources . Take ip address of the droplet

157.33.44.44:8081 search on url browser. Also add artifact to the nexus repository .

Sign in as add admin and password from the terminal

```
nexus@ubuntu-s-4vcpu-8gb-fra1-01:~$ ls /opt/sonatype-work/nexus3
admin.password  cache  elasticsearch  generated-bundles  karaf.pid  lock  orient  restore-from-backup
blobs           db     etc             instances          keystores  log   port   tmp
nexus@ubuntu-s-4vcpu-8gb-fra1-01:~$ cat /opt/sonatype-work/nexus3/admin.password
4527ff88-5b80-41bd-9341-fde4e808fa6d
nexus@ubuntu-s-4vcpu-8gb-fra1-01:~$
```

Repository section on nexus

The screenshot shows the Sonatype Nexus Repository OSS 3.53.0-01 interface. The left sidebar contains the 'Administration' menu with options like Repository, Repositories, Blob Stores, Log4j Visualizer, Proprietary Repositories, Content Selectors, Cleanup Policies, Routing Rules, Security, Privileges, Roles, Users, Anonymous Access, and LDAP. The main area is titled 'Repositories Manage repositories' and features a 'Create repository' button and a 'Filter' dropdown. A table lists existing repositories with columns for Name, Type, Format, Status, URL, Health check, and Firewall Rules.

Name ↑	Type	Format	Status	URL	Health check	Firewall R
maven-central	proxy	maven2	Online - Ready to Connect		Analyze	
maven-public	group	maven2	Online			
maven-releases	hosted	maven2	Online			
maven-snapshots	hosted	maven2	Online			
nuget-group	group	nuget	Online			
nuget-hosted	hosted	nuget	Online			
nuget.org-proxy	proxy	nuget	Online - Remote Available		Analyze	

Types of repository :

Proxy repository : linked to a remote repository .eg maven central . It saves network bandwidth

Hosted repository:primary repository for storing artifact and components all the artefacts owned by company is stored here .

Maven snapshots are internal development versions all ed snapshots .

Group repository : to combine multiple repo or other groups together .

Publish artifact to nexus repository :

Upload jar files from maven and gradle to the hosted repository

Gradle and maven command for pushing to remote repo .

But first we have to configure both tools to connect with nexus (nexus repo url and credential).

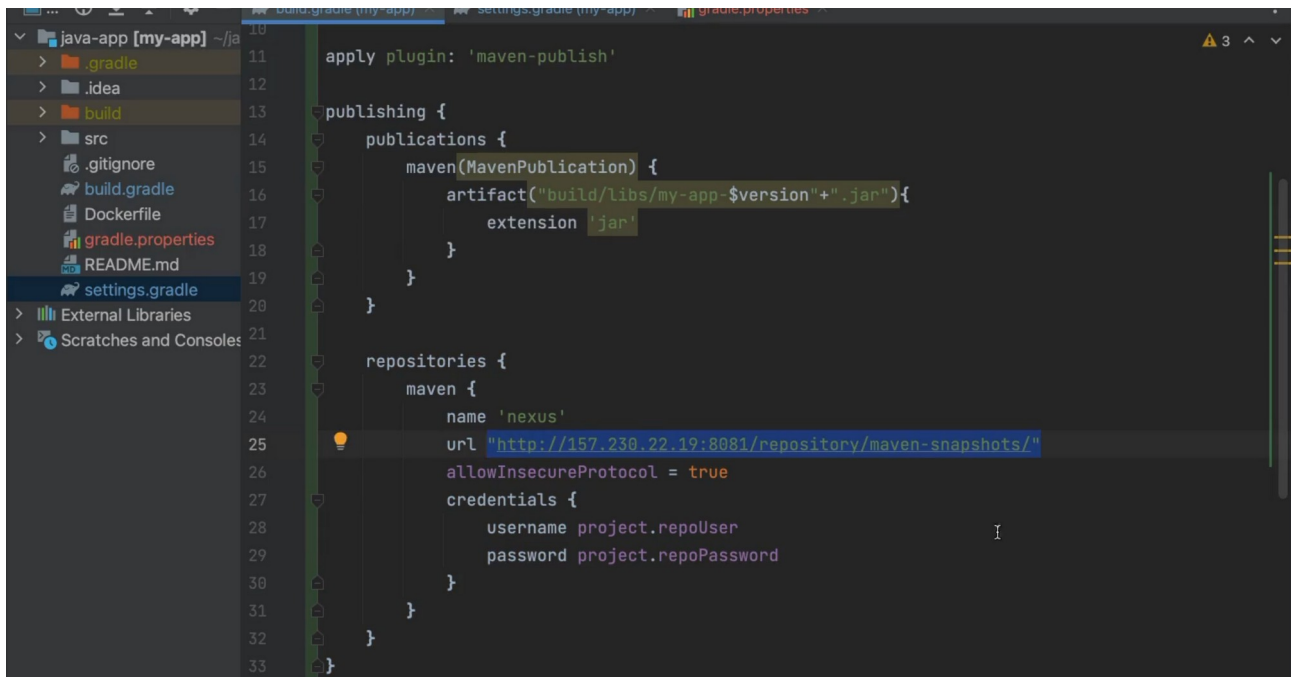
Nexus users in permission to upload artifact file .

We need to create nexus user to upload jar file from our laptop

Creating nexus user = Go to security : users : create role :

Gradle project configure with nexus :

java app = Apply plugin:'maven-publish'. For publishing a jar to maven formate repository because both gradle and maven uses the same maven formate to upload the java artifacts . This will allow gradle to connect to nexus and push to maven repo .



This is help gradle to upload the jar file to the nexus repo after build command . .

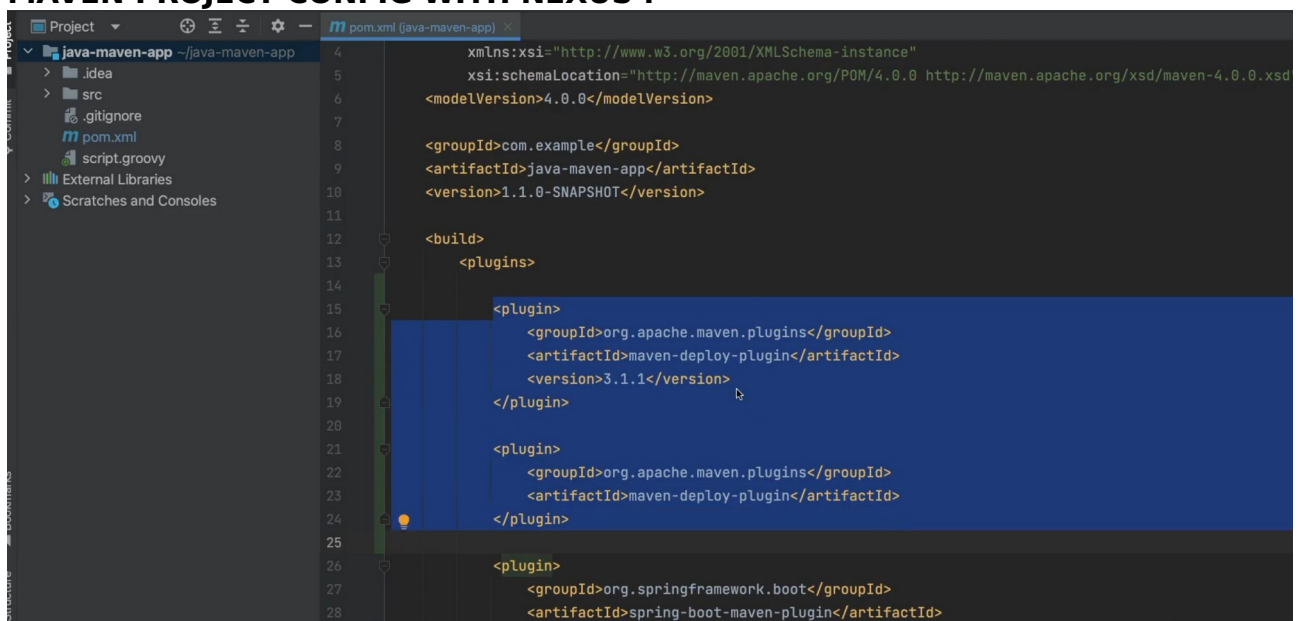
Gradle project jar upload : gradle build , build folder will have jar file .

Gradle publish command : apply plugin:'maven-publish' make this GRADLE PUBLISH command available in gradle .

Go to user view in browse : to see the components jar file in snapshot

WE HAVE NOW FIRST JAR ARTIFACT IN NEXUS REPO USING GRADLE BUILD TOOL .

MAVEN PROJECT CONFIG WITH NEXUS :



FIRST BLOCK IS TO DEFINE PLUGIN AND SECOND TO USE THAT . Refresh the project so that maven can download that plugin

Let u configure repo for my snapshot

```
<distributionManagement>
  <snapshotRepository>
    <id>nexus-snapshots</id>
    <url>http://157.230.22.19:8081/repository/maven-snapshots/</url>
  </snapshotRepository>
</distributionManagement>
```

.m2 file where maven global credentials can be defined . Which will be accessible for all your maven projects.

```
nana@macbook /Users/nana/java-app [master]
% cd ~
nana@macbook /Users/nana
% ls -a | grep .m2
.m2
nana@macbook /Users/nana
% ls .m2/
repository
nana@macbook /Users/nana
% cd .m2
nana@macbook /Users/nana/.m2
% ls
repository
nana@macbook /Users/nana/.m2
% vim settings.xml
```

```
<settings>
  <servers>
    <server>
      <id>nexus-snapshots</id>
      <username>nana</username>
      <password>xxxxx</password>
    </server>
  </servers>
</settings>
```

Username : nexus user name and nexus user password .

Mvn package to build the artifact .

Ls target/ = mvn deploy = we can load the jar to the nexus repo .

NEXUS API :

Query nexus repo for different information .

Eg : which components , version ,repo etc.

This info is needed in your cicd pipeline =

Build-> test ->release ->deploy ->operate.

How to access rest end points :

Use command like curl or wget to execute http request

Provide user and credential of a nexus user .

Use the nexus user for the required permissions .

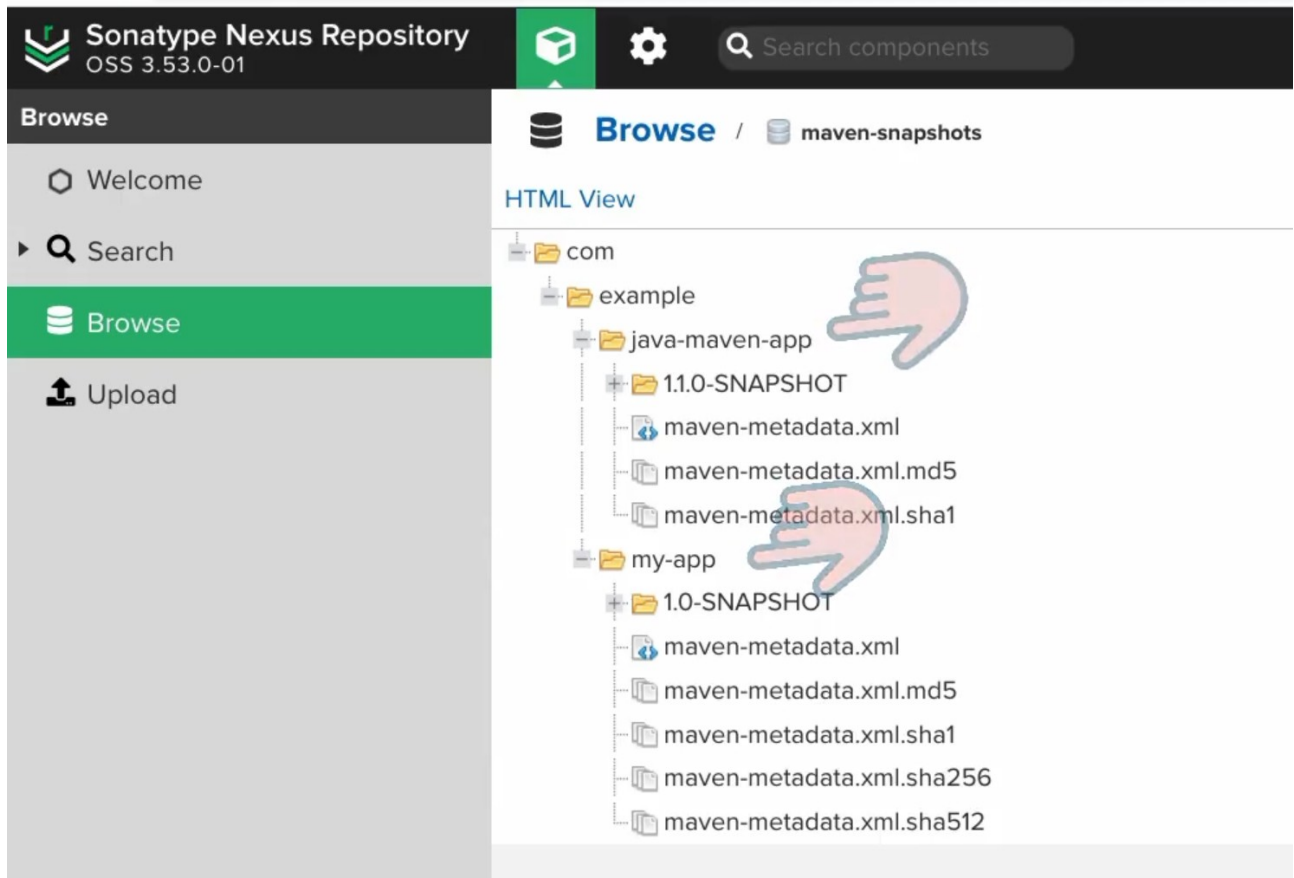
We want to see what repo are there in our nexus query repo ??

```
nana@macbook /Users/nana  
% curl -u user:pwd -X GET 'http://157.230.22.19:8081/service/rest/v1/repositories' ⌘
```

User : pwd : u can set admin : password to see all the repo admin has access to .

To get the components of specific repo ?

```
nana@macbook /Users/nana
% curl -u nana:techworld -X GET 'http://157.230.22.19:8081/service/rest/v1/components?repository=maven-snapshots'
```



The screenshot shows the Sonatype Nexus Repository OSS 3.53.0-01 interface. The left sidebar has 'Browse' selected. The main area shows a tree view of components. A hand icon points to the 'example' folder, and another hand icon points to the 'my-app' folder. The 'my-app' folder contains a '1.0-SNAPSHOT' subfolder with several metadata files.

Searching components via its unique id :

```
nana@macbook /Users/nana
% curl -u nana:techworld -X GET 'http://157.230.22.19:8081/service/rest/v1/components/bWF2ZW4tc25hcHNob3RzOmMwYmYyYWI2YzVlOTNhNGE3NmY3OTkxMDI5ZTA5NGM4'
```

BLOB STORE:

Nexus must have some storage to store all the uploaded files .
Blob store is what nexus is using to manage the storage all the components.
Blob store is a storage of binary file it can be local or cloud storage
Each blob store can be used to store multiple repo and repo groups .

Attributes of blob store :

Type file of blob store represent file system bases storage

```
root@ubuntu-s-4vcpu-8gb-fra1-01:~# ls /opt/
digitalocean  nexus-3.53.0-01  nexus-3.53.0-01-unix.tar.gz  sonatype-work
root@ubuntu-s-4vcpu-8gb-fra1-01:~# ls /opt/sonatype-work/nexus3/
blobs  db  etc  instances  keystores  log  port  tmp
cache  elasticsearch  generated-bundles  karaf.pid  lock  orient  restore-from-backup
root@ubuntu-s-4vcpu-8gb-fra1-01:~# cd /opt/sonatype-work/nexus3/
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt/sonatype-work/nexus3# ls blobs/
default
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt/sonatype-work/nexus3# ls blobs/default
A68E86E5-A0F36E3F-0AE371DC-89DDABED-D02C5D94-deletions.index  content  reconciliation
A68E86E5-A0F36E3F-0AE371DC-89DDABED-D02C5D94-metrics.properties  metadata.properties
root@ubuntu-s-4vcpu-8gb-fra1-01:/opt/sonatype-work/nexus3#
```

Another type for blob storage is S3 which is cloud based storage
S3 is recommended for repository managers which are deployed in AWS .

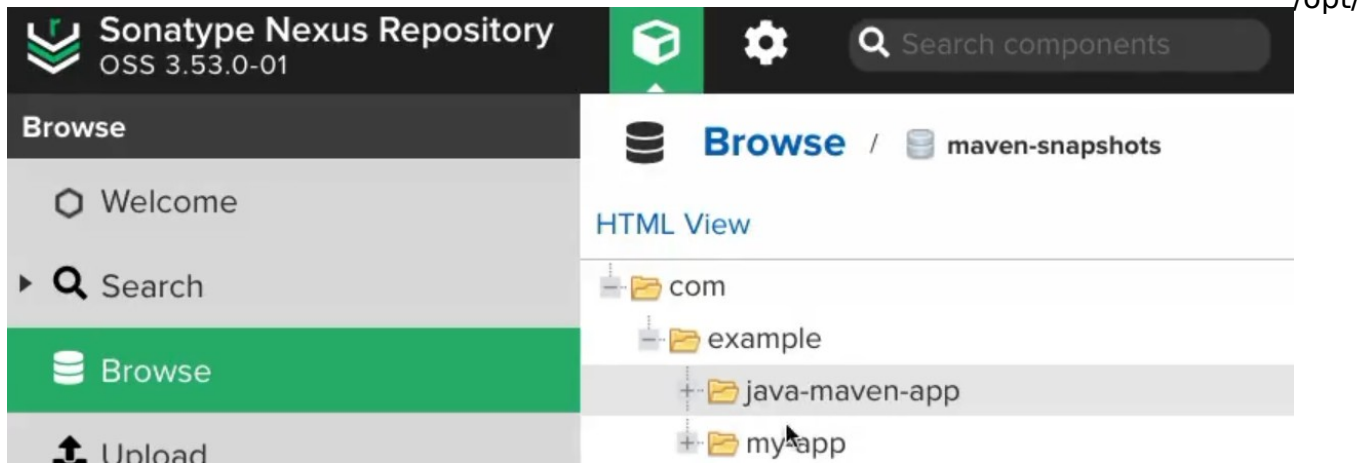
Blob store state : started means running as accepted .

Failed means configuration issue .

Blob count : components artifacts are stored in different blobs .

Ls /blobs /default/content .

Creating new blob : path : absolute path to the desired file system location .



sonatype-work/nexus3/blobs/mystore

Also it should be fully accessible by the os user account running nexus user .

Key points :

Once blobs created can't be modified .

Blobs created by repo cant be deleted.

U need to know approx how much space each repo will take .

Which blob u will use for which repo.

Once a repo is allocated to blob it sticks there permanently , blobs can be move from one to other if u need more space , blobs can not split , one repo can not be assigned to multiple blobs .

Ls blobs/ will show your blobs.

Way to assign blob to a repo : when u create a repo u can select with storage blob u need to assign it to.

COMPONENTE VS ASSEST :

Java -maven -app , my app are components :
Pom , jar , xml these are assets . That all belong to a single components .
Components : abstract , what we are uploading
Assets : file , physical packages . 1 components = 1 or more assets .

Docker formate give assets unique identifier (called docker layers);
Docker layer = Assets.
Eg: 2 docker images => 2 components , but same assets .

CLEAN UP POLICIES :

Rule what help to clean up the artifacts from the repo , in order to free up storage for new component

. * regular expression .

Attach policy with repo :

Configuring the cleanup policy >

Go to system -> task =

This will just mark them to be clean up to actually delete we need to do task =
compact blob store .

Compact store when executed the files will be physically deleted from the blob storage

.