

Nexus on Docker

Updated on 19/07/2022

++nexus container++install nexus in docker++containerize nexus++nexus on docker++nexus docker++

<https://hub.docker.com/r/sonatype/nexus3>

Create/Start nexus container:

Create a volume to persist the Nexus repositories(stored at /nexus-data):

```
$ docker volume create --name nexus-data  
$ docker run -d -p 8081:8081 --name nexus -v nexus-data:/nexus-data sonatype/nexus3
```



If any additional ports are to be used for connecting to certain registries,for example, to connect to docker registry at 8082.



```
$ docker run -d -p 8081:8081 -p 8082:8082 --name nexus -v nexus-data:/nexus-data  
sonatype/nexus3
```

Setting up Docker Registry listening on 8082:

In order to get this work,

- 1) **HTTP** should be checked and is provided with a port# (in this case, 8082). This port should have been opened while creating the container. In this case, 8082 is provided with the above “docker run” command.
- 2) **Allow anonymous docker pull** is NOT checked.
- 3) **Enable Docker V1 API** is checked.

 **Repositories** /  appointme-docker

 Delete repository  Rebuild index

Settings

Name: appointme-docker
Format: docker
Type: hosted
URL: http://34.125.93.230:8081/repository/appointme-docker/
Online: If checked, the repository accepts incoming requests

Repository Connectors

Connectors allow Docker clients to connect directly to hosted registries, but are not always required. Consult our [documentation](#) for which connector is appropriate for your use case. For information on [scaling the repositories](#) see our [scaling documentation](#).

HTTP:
Create an HTTP connector at specified port. Normally used if the server is behind a secure proxy.
 8082

HTTPS:
Create an HTTPS connector at specified port. Normally used if the server is configured for https.

Allow anonymous docker pull:
 Allow anonymous docker pull (Docker Bearer Token Realm required)

The port in the Nexus container should have been opened.

Docker Registry API Support

Enable Docker V1 API:

Allow clients to use the V1 API to interact with this repository

Storage

Blob store:

default

Strict Content Type Validation:

Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

Hosted

Deployment policy:

Controls if deployments of and updates to artifacts are allowed

Allow redeploy

Proprietary Components:

Components in this repository count as proprietary for namespace conflict attacks (requires Sonatype Nexus Firewall)

Proprietary Components:

Components in this repository count as proprietary for namespace conflict attacks (requires Sonatype Nexus Firewall)

Cleanup

Cleanup Policies:

Components that match any of the Applied policies will be deleted

Available

Filter

Empty list box for Available policies

Applied

Empty list box for Applied policies

>
<

Configure daemon.json as below on the Docker server



```
>> cat /etc/docker/daemon.json
{
  "insecure-registries" : ["34.125.93.230:8082","34.125.209.72:8082"]
}
```

Connecting to the private Nexus registry:

```
>> docker login 34.125.93.230:8082
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

To upload non-java files to the repository manager:

```
curl -v -H "Connection: close" -F "r=snapshots" -F "g=com.mytasks" -F "a=tasks" -F "v=1.1-1" -F "p=phar"
-F "file=@/var/www/html/phar/myapp.phar" -u admin:admin123
http://localhost:8081/nexus/content/repositories/snapshots/myapp.phar
```

For reference, here are all the available form parameters for this endpoint:

r : repository

hasPom - whether you are supplying the pom or you want one generated. If you are uploading a pom, then the parameters **g**, **a**, **v**, **p**, and **c** are not needed as part of the command as those values are extracted from the pom.ml

e - extension

g - group id

a - artifact id

v - version

p - packaging

c - classifier (optional, not shown in examples above)

file - each file to be uploaded, use one file parameter per file.

The similar upload through Maven:

```
mvn deploy:deploy-file -DgroupId=com.somecompany -DartifactId=project -Dversion=1.0.0  
-DgeneratePom=true -Dpackaging=jar -DrepositoryId=nexus  
-Durl=http://localhost:8081/nexus/content/repositories/releases -Dfile=target/project-1.0.0.jar
```

To download files:

```
curl -u admin:admin123  
"http://localhost:8081/nexus/content/repositories/snapshots/myapp.phar?r=snapshots&g=com.m  
ytasks&a=tasks&v1.1-1&p=phar&c=" > myapp.phar
```