

# 2024.07.08

## 오전

vs code 에서

```
Group : com.encore
```

-> 프로젝트를 정의하는 고유한 식별자 정보

```
Artifact : de31
```

-> 세부 프로젝트를 식별하는 정보

폴더생성하고 java파일 하나생성

mvn package

build error 뜨면 자바버전 17로 bashrc 변경

```
> test-classes
```

```
demo-0.0.1-SNAPSHOT.jar
```

그러면

생성된다...

```
docker run -it --name myweb2 -p 8080:8080 ubuntu:14.04
```

cp 컨테이너 안에 복사..

```
hadoop@worker2:~/workspace/demo/target$ docker cp ./demo-0.0.1-SNAPSHOT.jar  
myweb2:/root/
```

컨테이너 들어가기

```
docker exec -it myweb2 bash
```

```
apt install software-properties-common
```

```
add-apt-repository ppa:openjdk-r/ppa
```

```
apt update
```

```
apt upgrade
```

```
apt install openjdk-17-jdk
```

컨테이너 환경에서 배포하기

```
root@dd3cf3a62399:/# java -jar /root/demo-0.0.1-SNAPSHOT.jar
```

tomcat은 was역할을한다.

```
docker commit -a "encore" -m "first image" myweb2 encore:0.1
```

도커는 컨테이너 안에서 돌아가는데... 해당하는 컨테이너를 도커 이미지로 만드는법

도커이미지를 파일로 저장...

**docker save -o myweb.tar encore:0.1**

이미지를 지우기전에 컨테이너를 먼저 지운다...

docker ps -a -q를 하면 컨테이너의 id값이 출력이 되는데

리눅스앞에서 \$붙으면 변수

docker rm \$(docker ps -a -q) 컨테이너 일괄삭제..

이미지 상세 정보 보기

**docker inspect encore:0.1**

docker inspect encore:0.1 > ~/encore.txt

ubuntu위에 encore:0.1를 레이어를 하나 더 올려싸...

이미지를 지울때에는

docker rmi encore:0.1

파일로 저장한 image 복원해보기

docker load -i ./myweb.tar

복원된 이미지로 컨테이너 만들기

docker run -it --name myweb3 -p 80:8080 encore:0.1

실행중인 컨테이너 들어가기...

docker attach myweb3

**docker run -itd --name myweb4 -p 80:8080 -v ~/java\_work/de31/:/root/ encore:0.1**

docker exec myweb4 ls /root/

docker attach myweb4

apt install maven

메이븐이 에러가 난다;;;

host에서 mvn package하고

cd root로 가서

**java -jar ./target/de31-0.0.1-SNAPSHOT.jar**

이러면 동기화가 되었으므로 `http://ip/api/vi/get-api/request1?`

`name=John&email=john@example.com&phone=1234567890` 들어가도 된다 즉 port지워도 된다..

마운트했으니깐 ㅇㅈ

dto (data transfer object) - 다른 레이어간의 데이터 교환, 데이터를 교환하는 용도로 사용하는 객체

`http://ip:port/api/vi/get-api/request3?`

`name=John&email=john@example.com&phone=1234567890`

도커에서 이미지 안올렸으니 컨테이너에서 하지말고.. local에서 8080으로 한다..

## 오후

docker image

docker pull ubuntu:22.04

안되면 apt update

apt install openjdk-17-jdk

apt install maven

mkdir /workspace/

docker commit -a "mort" -m "spring boot" myimage myimage:0.1

docker run -itd --name myspring1 -p 80:8080 -v ~/workspace/demo/:/workspace/ myimage:0.1

mvn clean 에러 안난다;; ubuntu 버전차이인건가?

docker attach

docker exec myspring1 /bin/bash

docker exec -it myspring1 /bin/bash

exec - container안에 명령어 전달하

attach로 들어가서 exit하면 죽는다

하지만 exec는 안죽는

mvn clean을 하면 target 파일이 죽는다....

```
<version>0.0.2-SNAPSHOT</version>
```

버전 0.0.2로 바꾸고

mvn package

```
drwxrwxr-x 3 root root 4096 Jul 8 05:30 classes/
-rw-r--r-- 1 root root 20095867 Jul 8 05:30 demo-0.0.2-SNAPSHOT.jar
```

이제는 포트를 지워도 된다 왜냐하면 바꾼거니깐...

//

docker volume

docker container inspect myspring1

docker volume ls

**docker run -it --name volume\_dummy alicek106/volume\_test**

cd /home/testdir\_2

cat test

testdir\_2!

docker run -it --name volume\_override -v ~/workspace/demo:/home/testdir\_2

alicek106/volume\_test

cd /home/testdir\_2/

내 로컬 호스트께 나온다...

docker run -it --name volume\_override2 -v ~/tttt:/home/testdir\_2 alicek106/volume\_test

빈공간을 만들어서 마운트하면 빈공간이 보인다!!!

\*\*

docker run -it --name vol\_from\_cont --volumes-from volume\_override ubuntu:22.04

복사한것을 복사...

도커볼륨은 저장소...가상머신이지만 리얼머신에 어딘가에 저장되어있다;;

docker volume create --name myvol

docker inspect myvol

를 하면 mount하는 방법이나온다...

저기는 권한이 root이므로 sudo su로 들어가서

cd /var/lib/docker/volumes/myvol/\_data

\*\*

docker run -itd --name myvol\_1 -v myvol:/root/ ubuntu:22.04

docker run -itd --name myvol\_2 -v myvol:/root/ ubuntu:22.04

\*\*

docker exec -it myvol\_1 /bin/bash

cd /root

echo로 아무파일하기...

그러면 myvol\_2에도 a.txt가 있고 /var/lib/docker/volumes/myvol/\_data 에도 a.txt가 있다...

```
docker exec myvol_2 cat /root/a.txt
```

```
docker exec myvol_2 ls /root
```

```
cp ~/java_work/de31/* -R /var/lib/docker/volumes/myvol/_data
```

- sudo su  
cd /var/lib/docker/volumes/myvol/\_data  
cp -R /home/kafka/java\_work/de31/\* /var/lib/docker/volumes/myvol/\_data

\*\*

**post방식만들어보기!!**

탈런드 설치하기..

**netplan 오류**

```
dpkg -l | grep netplan
```