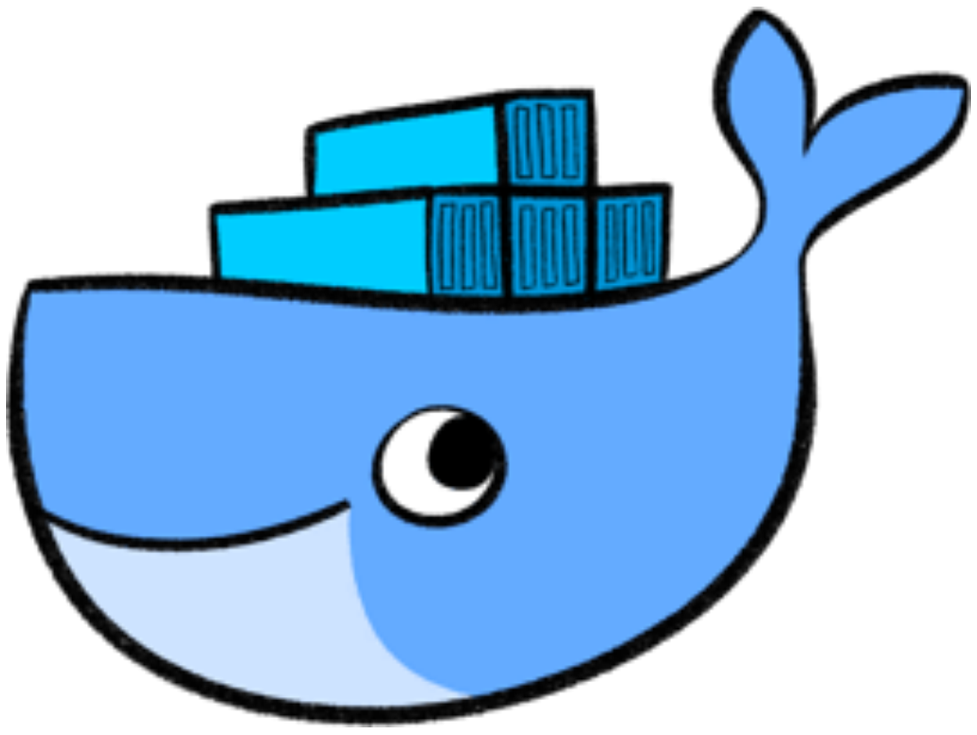
Docker

안태경

???



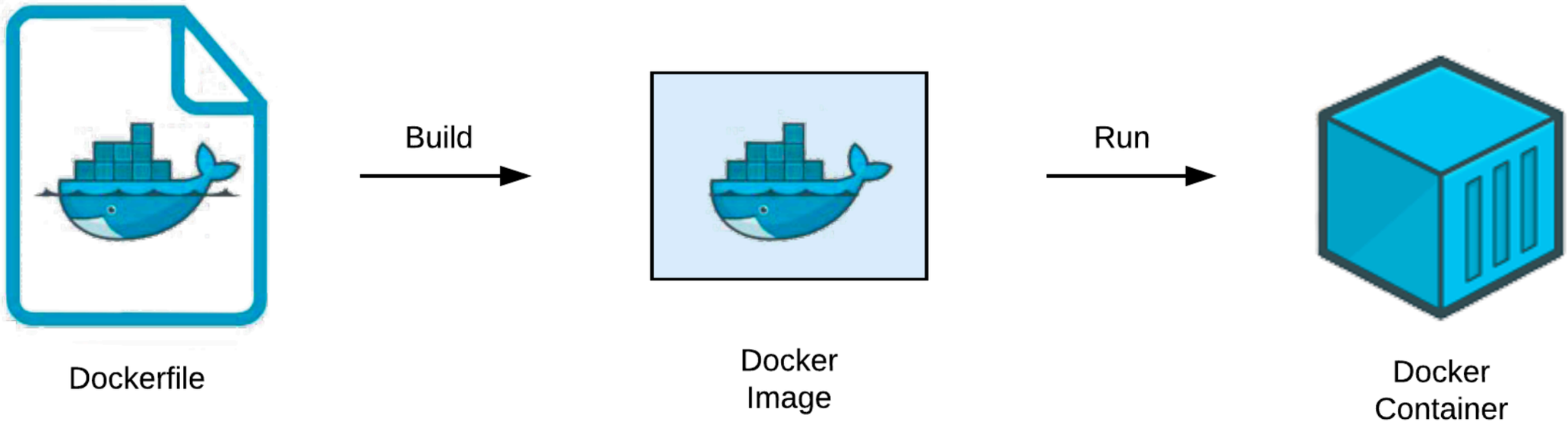
🐳 도커란? 🐳

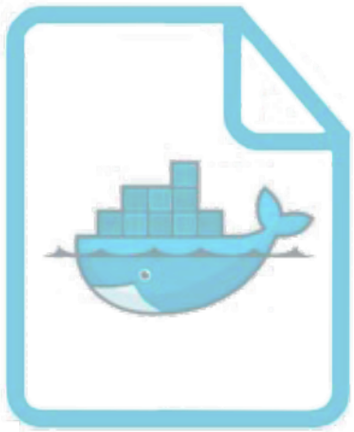
- 다양한 **프로그램**이나 실행환경을 **컨테이너**로 추상화

- **컨테이너**를 관리하며 **프로그램의 배포&관리**를 단순화

- 백엔드 프로그램, 데이터베이스 서버 등 어떤 프로그램도 **컨테이너화** 할 수 있고 aws, google cloud 등 어디서든 실행 가능

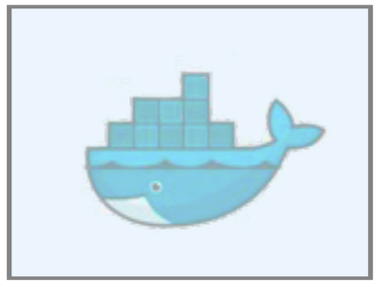
도커의 동작 과정





Dockerfile

Build



Docker
Image

Run

Docker
Container

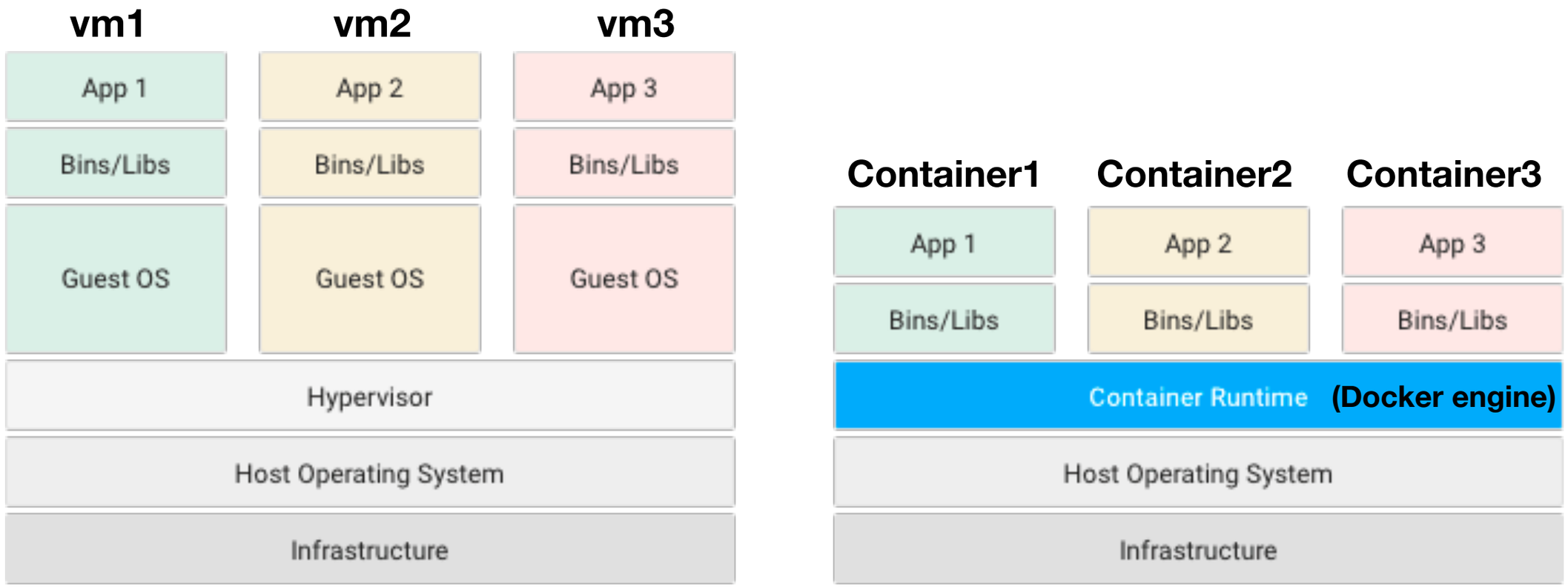
Docker Container

- 기존에는 하나의 시스템에서 둘 이상의 소프트웨어를 동시에 실행하면 문제가 생김
- 개별 소프트웨어에 필요한 **독립적인 실행환경**을 만들어줌
- 별도의 OS를 가상화하지 않음 (프로세스를 격리)

 컨테이너의 장점 

- OS를 가상화하는 기존 방식(Virtual Machine)보다 가벼움

- 운영체제 커널에서 직접 구동하기 때문에 빠르고 메모리를 적게 차지함

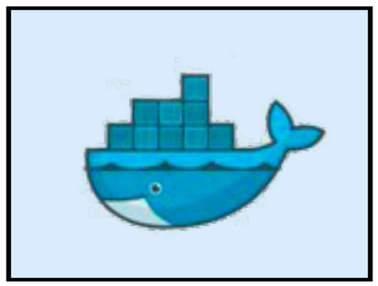


Virtual Machines

Containers

Dockerfile

Build



Docker
Image

Run

Docker
Container

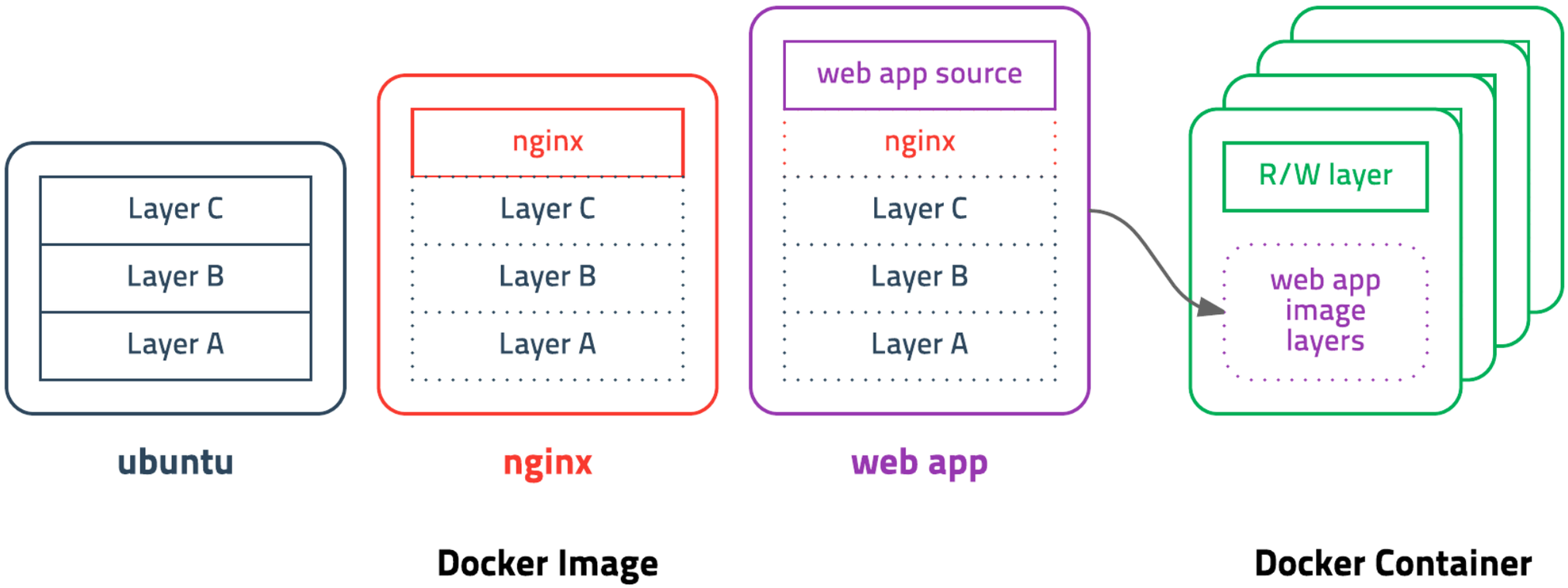
Docker Image

- 컨테이너 실행에 필요한 파일과 설정값을 포함한 단위
- 같은 이미지에서 **여러 개의 컨테이너** 생성 가능
- 컨테이너의 상태가 바뀌거나 삭제되어도 **이미지는 변하지 않음**
- Docker Hub나 Docker Registry를 만들어 관리

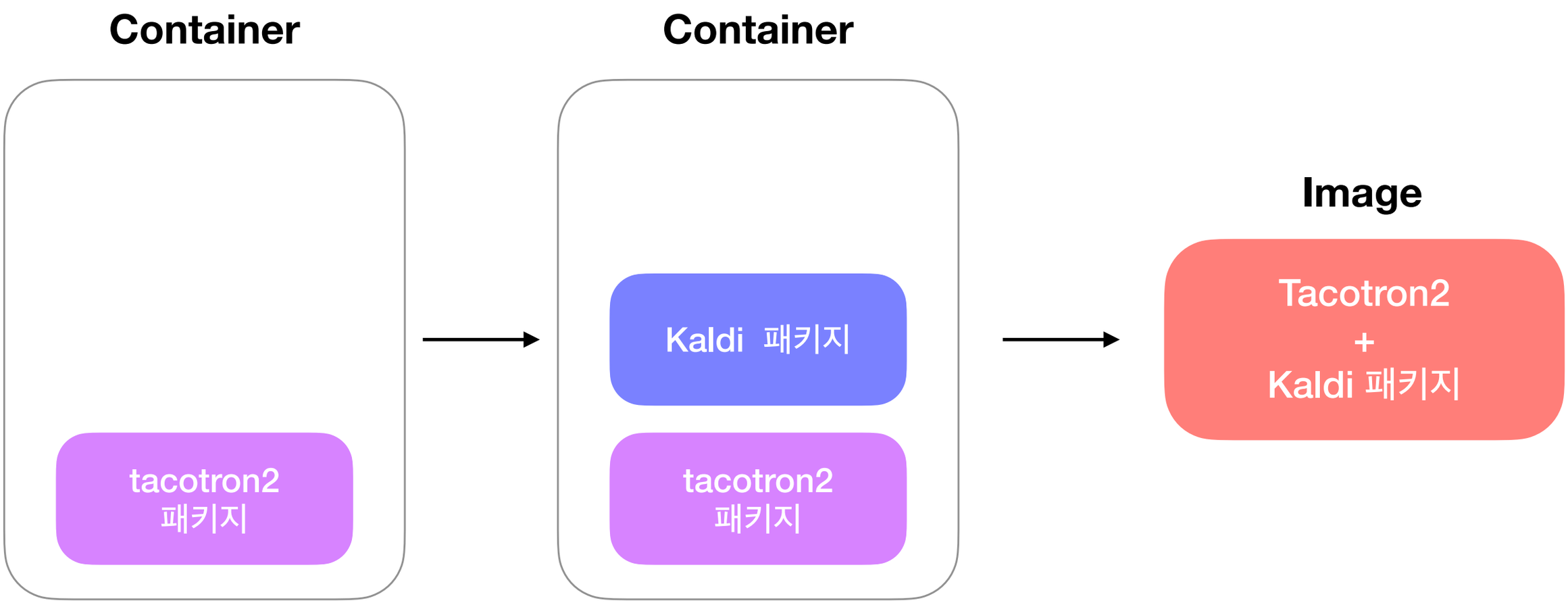
Docker Layer

- 용량이 큰 이미지를 수정하면 다시 그 이미지를 다운받아야 하는 문제를 방지하기 위해 쓰는 방법

- 컨데이너 생성할 때에도 레이어 방식 적용



Docker Layer

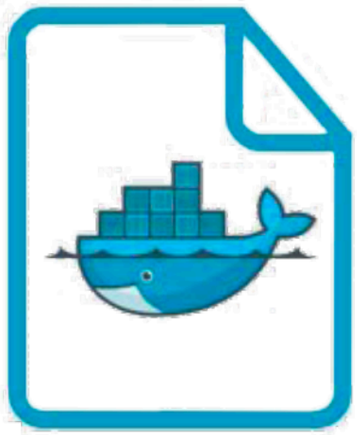


타코트론2 이미지를 띄운
기존 컨테이너에

칼디를 설치하고

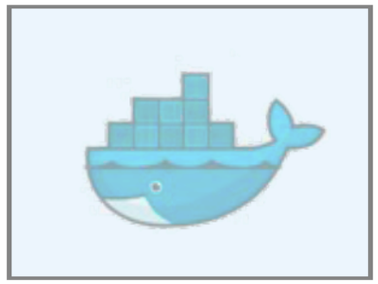
해당 컨테이너를
새로운 이미지로 저장할 수 있다
(방법은 뒤에 설명 예정)

참고 자료 : https://www.scalyr.com/blog/create-docker-image/



Dockerfile

Build



Docker
Image

Run

Docker
Container

Docker File

- 이미지를 만들기 위해 DSL(Domain-specific language)를 사용하여 이미지 생성과정을 기록한 파일(스크립트)

- 서버를 관리할 때 설치한 프로그램, 패키지, 설정파일 등등등을 따로 기록하지 않고 docker file로 관리할 수 있음

- 버전 단위로 관리되고 언제든 생성과정을 확인하고 수정 가능

Branch: master ▾ serve / docker /

Create new file | Upload files | Find file | History

Manoj Rao update docker tags while building ··· ✓ Latest commit 8533136 23 days ago

..		
Dockerfile.cpu	Merge branch 'stage_release' into quick_start_scripts	2 months ago
Dockerfile.gpu	Merge branch 'quick_start_scripts' of https://github.com/pytorch/serve ...	2 months ago
README.md	update docker tags while building	23 days ago
config.properties	updated sanity test_script and added model-store back in docker scripts	2 months ago
dockerd-entrypoint.sh	Added local python package installation support	4 months ago

README.md

Create TorchServe docker image

```
cd serve/docker
git clone https://github.com/pytorch/serve.git
```

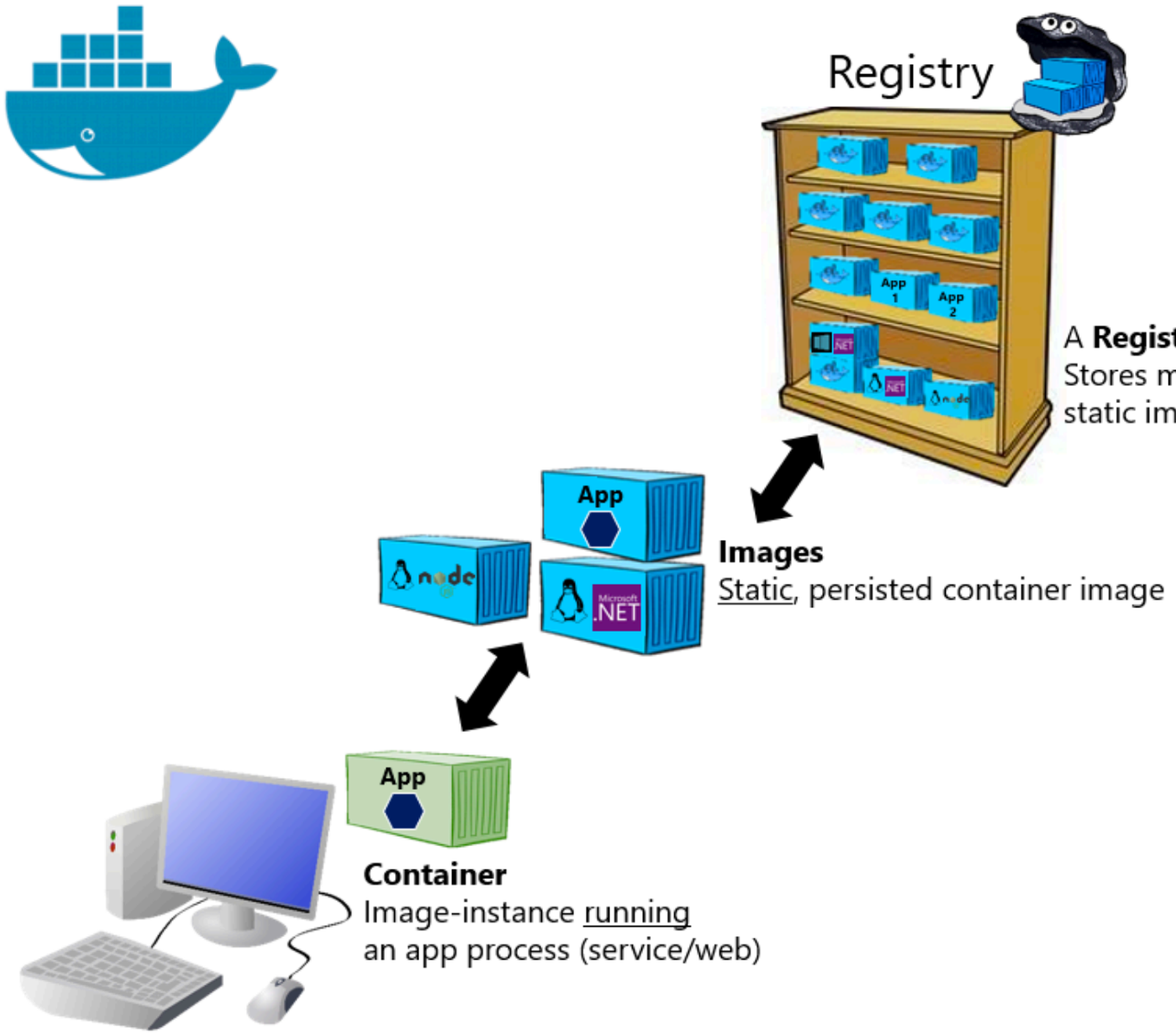
For creating CPU based image :

```
docker build --file Dockerfile.cpu -t torchserve:latest .
```

For creating GPU based image :

```
docker build --file Dockerfile.gpu -t torchserve:latest .
```

Basic taxonomy in Docker



Hosted Docker Registry

Docker Trusted Registry on-prem.

On-premises
('n' private organizations)

Docker Hub Registry

Docker Trusted Registry on-cloud

Azure Container Registry

AWS Container Registry

Google Container Registry

Quay Registry

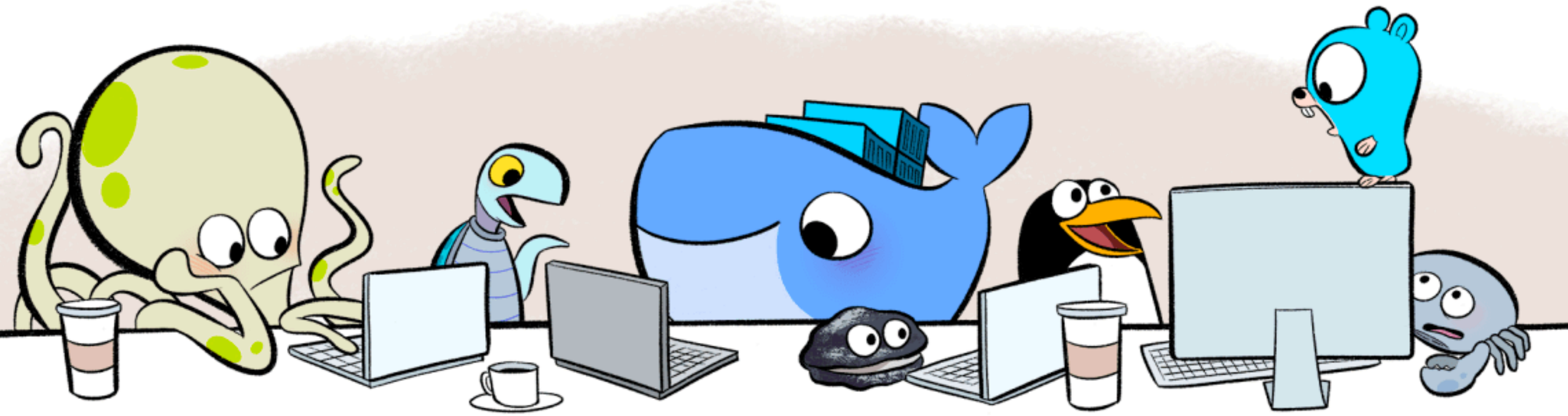
Other Cloud

Public Cloud
(specific vendors)

아주 간단한 도커 컨테이너 실행

```
(base)  gaetokk  ~  docker run --rm -it ubuntu:16.04 /bin/bash
Unable to find image 'ubuntu:16.04' locally
16.04: Pulling from library/ubuntu
e92ed755c008: Pull complete
b9fd7cb1ff8f: Pull complete
ee690f2d57a1: Pull complete
53e3366ec435: Pull complete
Digest: sha256:db6697a61d5679b7ca69dbde3dad6be0d17064d5b6b0e9f7be8d456ebb337209
Status: Downloaded newer image for ubuntu:16.04
root@83c61c63e48c:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@83c61c63e48c:/# cat /etc/issue
Ubuntu 16.04.6 LTS \n \l

root@83c61c63e48c:/# exit
exit
```



서버에서 도커를 사용해 봅시다

sudo docker load -i {이미지파일}

```
taekyung@walnut:~/projects1/tts_train$ sudo docker load -i kaldi_with_taco2.tar.gz
f0218ecf3d62: Loading layer [==================================================>]  2.076GB/2.076GB
Loaded image: kaldi:with_taco2
```

sudo docker run

```
taekyung@walnut:~/projects1/tts_train$ sudo docker run -d -e CUDA_VISIBLE_DEVICES=1 -it --mount type=bind,source=/mnt/data1/taekyung/pr
ojects/tts_train/A096_ADAPTIVE_TTS/tf_model/tacotron2,target=/0 --name taco2_tk_1 --rm --runtime=nvidia --shm-size=1g --ulimit memlock=
-1 --ulimit stack=67108864 kaldi:with_taco2
efe3fc9da92231780cf3d2f8d0a120bb032d0714ce40d9ef16f9e2ee18331045
```

컨테이너를 background에서 실행 환경변수 설정 터미널 입력 컨테이너에 파일을 마운트

```
sudo docker run -d -e CUDA_VISIBLE_DEVICES=0 -it --mount
type=bind,source=/mnt/data1/taekyung/projects/tts_train/
```

입력한 경로가 디렉토리 /0/이 된다 컨테이너 이름

```
A096_ADAPTIVE_TTS/tf_model/tacotron2,target=/0 --name
```

프로세스 종료 시 자동으로 컨테이너 제거 nvidia-docker 사용 shared memory size 메모리 최대 사이즈

```
taco2_tk --rm --runtime=nvidia –shm-size=1g --ulimit
```

컨테이너에 띄울 이미지 이름 (이름:태그)

```
memlock=-1 --ulimit stack=67108864 tacotron2:v_1_0_0
```

https://docs.docker.com/engine/reference/commandline/run/

sudo docker container ls

```
taekyung@peanut:~$ sudo docker container ls
CONTAINER ID        IMAGE                 COMMAND                  CREATED             STATUS              PORTS               NAMES
152896a076dc        waveglow:v_1_0_0      "/usr/local/bin/nvid…"   2 weeks ago         Up 2 weeks          6006/tcp            tmp_wg_jsyoon
96f5728e6f45        waveglow:v_1_0_0      "/usr/local/bin/nvid…"   2 weeks ago         Up 2 weeks          6006/tcp            tmp_ydkwon_glow
655bd30c5488        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   2 weeks ago         Up 2 weeks          6006/tcp            tmp-jsyoon
238c86c08435        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   3 weeks ago         Up 3 weeks          6006/tcp            tmp_ydkwon
a62c6213dfd0        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   3 weeks ago         Up 3 weeks          6006/tcp            tmp
85230b953163        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   5 weeks ago         Up 5 weeks          6006/tcp            taco2_tk_2
ff24b5fb79c4        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   5 weeks ago         Up 5 weeks          6006/tcp            taco2_tk_1
262b68568598        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   5 weeks ago         Up 5 weeks          6006/tcp            taco2_tk_0
43fcc4503ba9        tacotron2:v_1_0_0     "/usr/local/bin/nvid…"   5 weeks ago         Up 5 weeks          6006/tcp            taco2
```

sudo docker exec -it {컨테이너명} bash

```
taekyung@walnut:~/tf_model/tacotron2$ ls
config  data  log  model  output  pretrained  src
taekyung@walnut:~/tf_model/tacotron2$ sudo docker exec -it taco2_tk_1 bash
root@efe3fc9da922:/workspace# ls
README.md  docker-examples  nvidia-examples
root@efe3fc9da922:/workspace# cd ../0
root@efe3fc9da922:/0# ls
config  data  log  model  output  pretrained  src
root@efe3fc9da922:/0# cd src
root@efe3fc9da922:/0/src# ls
01_female.sh  02_f_68.sh  02_male_1.sh  data  log  main.py  model  test_serving_api.py  utils
root@efe3fc9da922:/0/src# python main.py
```

sudo docker stop {컨테이너명}

```
taekyung@peanut:~$ sudo docker stop taco2_tk_1
taco2_tk_1
```

기존 컨테이너에서 새로운 이미지 만들기

```
$sudo docker commit {컨테이너명}

$sudo docker images (<none>에 해당하는 image id 확인)

$sudo docker tag {image id} {이미지명}

$sudo docker save -o {파일명}.tar.gz {이미지명:태그}
```

기존 컨테이너에서 새로운 이미지 만들기

```
taekyung@peanut:~/tf_model$ sudo docker commit kaldi_test_1
sha256:247562440e260fbdbae414de78b4fbea912ada8a8849ecdc17d8e31440b11e33
taekyung@peanut:~/tf_model$ sudo docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED             SIZE
<none>                              <none>              247562440e26        10 seconds ago      8.97GB
tensorflow/tensorflow               latest-gpu-py3      e2a4af785bdb        4 months ago        4.11GB
tf_hardware_optimized_serving_mkl   v_1_0_0             8026bc6dde3d        6 months ago        545MB
nvidia/cuda                         9.0-base            1443caa429f9        8 months ago        137MB
waveglow                            v_1_0_0             889d1caed66c        8 months ago        6.88GB
tacotron2                           v_1_0_0             d1ed5ee0e59f        8 months ago        6.88GB
tf_hardware_optimized_serving_gpu   v_1_0_0             b3eaf80faabc        10 months ago       2.48GB
kwb425/tensorflow-serving-gpu       latest              b3eaf80faabc        10 months ago       2.48GB
nvcr.io/nvidia/tensorflow           19.06-py3           9f1f8aff1e6e        11 months ago       6.88GB
taekyung@peanut:~/tf_model$ sudo docker tag 247562440e26 kaldi_with_taco2
taekyung@peanut:~/tf_model$ sudo docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED             SIZE
kaldi_with_taco2                    latest              247562440e26        56 seconds ago      8.97GB
tensorflow/tensorflow               latest-gpu-py3      e2a4af785bdb        4 months ago        4.11GB
tf_hardware_optimized_serving_mkl   v_1_0_0             8026bc6dde3d        6 months ago        545MB
nvidia/cuda                         9.0-base            1443caa429f9        8 months ago        137MB
waveglow                            v_1_0_0             889d1caed66c        8 months ago        6.88GB
tacotron2                           v_1_0_0             d1ed5ee0e59f        8 months ago        6.88GB
kwb425/tensorflow-serving-gpu       latest              b3eaf80faabc        10 months ago       2.48GB
tf_hardware_optimized_serving_gpu   v_1_0_0             b3eaf80faabc        10 months ago       2.48GB
nvcr.io/nvidia/tensorflow           19.06-py3           9f1f8aff1e6e        11 months ago       6.88GB
taekyung@peanut:~/tf_model$ sudo docker save -o kaldi_taco2.tar.gz kaldi_with_taco2:latest
taekyung@peanut:~/tf_model$ ls
kaldi  kaldi_taco2.tar.gz  ser_measure  tacotron2  waveglow
```

서버에서 도커 사용 시 주의할 점

- 도커는 기본적으로 root 권한이 필요함
- 서버에서 도커 명령어만 sudo를 허용해 놓은 상태
- 실행한 컨테이너 안에서의 동작은 root 권한

-> 도커 안에서 만들어진 파일의 권한 확인하기!

- vscode 사용 시 파일 변경 사항이 바로 동기화되지 않음

```
taekyung@coconut:~/projects1/tts_train$ ls -l
total 9996660
drwxr-xr-x  7 taekyung staff       4096  4월  2 15:04 A096_ADAPTIVE_TTS
-rw-r--r--  1 taekyung staff       1189  4월  3 17:15 data_processing.sh
drwxr-xr-x 12 taekyung staff       4096  4월 23 14:52 kaldi
-rw-------  1 root     root  9043612160  4월 23 22:08 kaldi_with_taco2.tar.gz
-rw-r--r--  1 taekyung staff        766  4월  3 18:23 move_trans.sh
-rw-r--r--  1 taekyung staff       1326  4월  3 17:40 move_txt.sh
-rw-r--r--  1 taekyung staff       1017  4월  3 18:00 move_wav.sh
-rw-r--r--  1 taekyung staff     844624  4월  3 16:39 saved.txt
drwxr-xr-x 12 taekyung staff       4096  4월  3 18:12 spk
drwxr-xr-x 12 taekyung staff       4096  4월  3 15:55 subset_data
-rw-r--r--  1 taekyung staff 1192064283  4월  3 15:53 subset_data.tar.gz
drwxr-xr-x  2 taekyung staff       4096  4월  3 18:08 subset_txt_data
-rw-r--r--  1 taekyung staff          0  4월  2 14:37 use
lrwxrwxrwx  1 taekyung staff         68  4월  2 14:40 wav_data -> /mnt/data1/yo
```

```
root@11dfd2c00bf6:/0# chmod 644 kaldi_with_taco2.tar.gz
root@11dfd2c00bf6:/0# ls -l
total 9996660
drwxr-xr-x  7 5021 staff       4096 Apr  2 06:04 A096_ADAPTIVE_TTS
-rw-r--r--  1 5021 staff       1189 Apr  3 08:15 data_processing.sh
drwxr-xr-x 12 5021 staff       4096 Apr 23 05:52 kaldi
-rw-r--r--  1 root root  9043612160 Apr 23 13:08 kaldi_with_taco2.tar.gz
-rw-r--r--  1 5021 staff        766 Apr  3 09:23 move_trans.sh
-rw-r--r--  1 5021 staff       1326 Apr  3 08:40 move_txt.sh
-rw-r--r--  1 5021 staff       1017 Apr  3 09:00 move_wav.sh
-rw-r--r--  1 5021 staff     844624 Apr  3 07:39 saved.txt
drwxr-xr-x 12 5021 staff       4096 Apr  3 09:12 spk
drwxr-xr-x 12 5021 staff       4096 Apr  3 06:55 subset_data
-rw-r--r--  1 5021 staff 1192064283 Apr  3 06:53 subset_data.tar.gz
drwxr-xr-x  2 5021 staff       4096 Apr  3 09:08 subset_txt_data
-rw-r--r--  1 5021 staff          0 Apr  2 05:37 use
lrwxrwxrwx  1 5021 staff         68 Apr  2 05:40 wav_data -> /mnt/data1/yo
```

그 외 여러가지 도커의 기능

도커 컴포넌트

- Docker Engine : 코어 기능 수행, Dockerfile을 통한 이미지 생성, 컴포넌트 구동, Docker 커맨드 실행
- Docker Kinematic : GUI 툴, 이미지 관리, 컴포넌트 구동
- Docker Registry : 이미지 공유
- Docker Compose : 여러 컨테이너를 통합 관리
- Docker Machine : 도커 실행환경을 커맨드로 생성하기 위한 툴
- Docker Swarm : 여러 도커 호스트를 클러스터화 하기 위한 툴

Docker Engine

Docker Swarm

Docker Compose

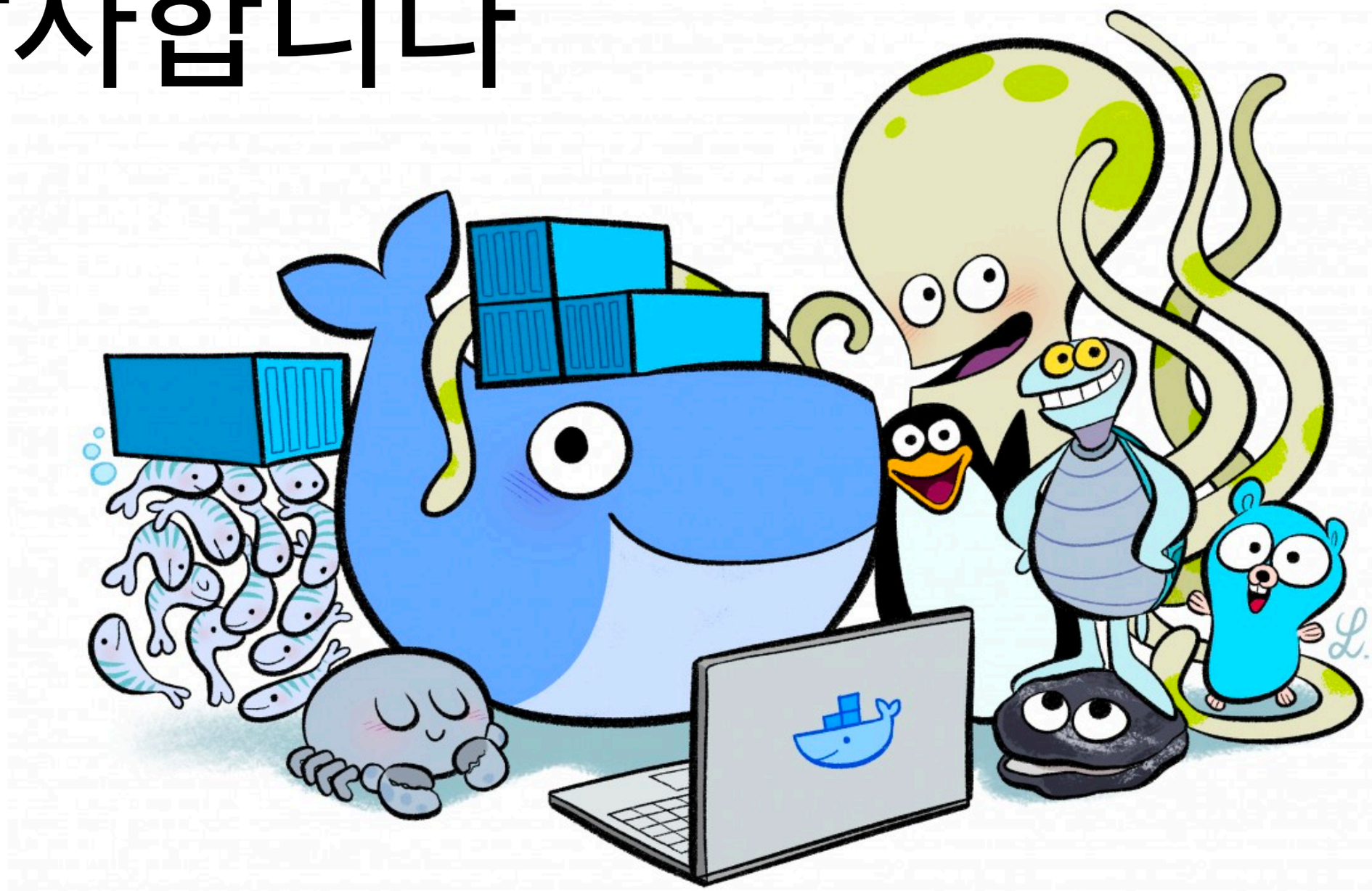
Docker Machine

Docker Registry

Kitematic

http://kimstar.kr/7695/

감사합니다



참고 : https://subicura.com/2017/01/19/docker-guide-for-beginners-1.html
https://subicura.com/2017/01/19/docker-guide-for-beginners-2.html