



Docker 넓고 얇게 알아보기



컨테이너

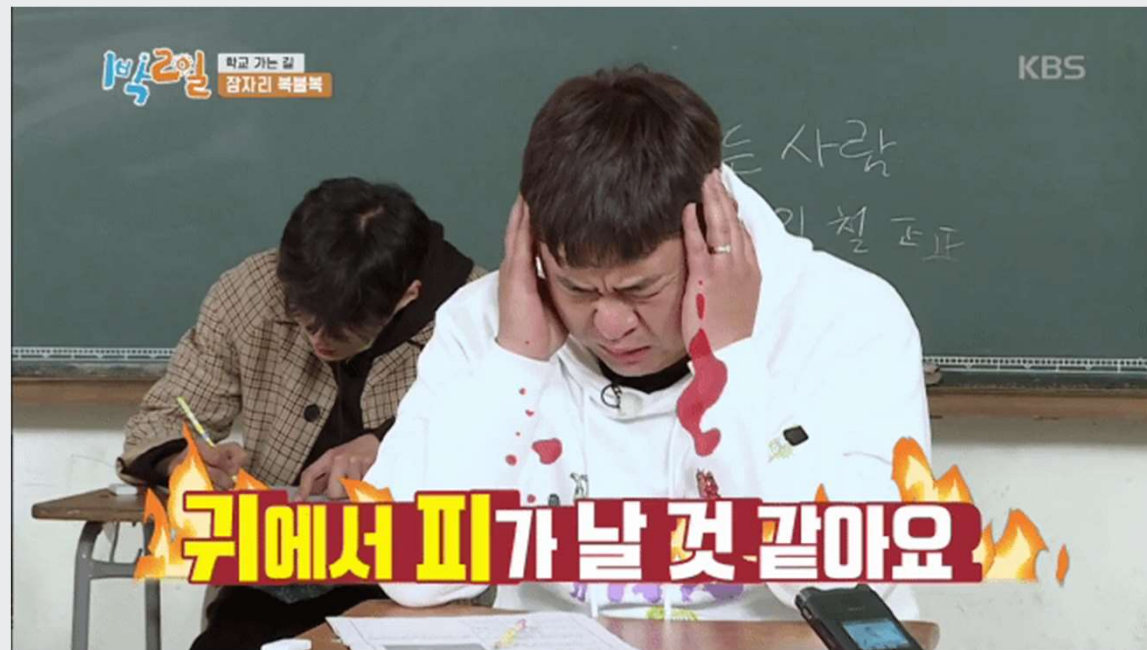
쿠버네티스

경량 가상화

MSA

DevOps


도커



## Part 1

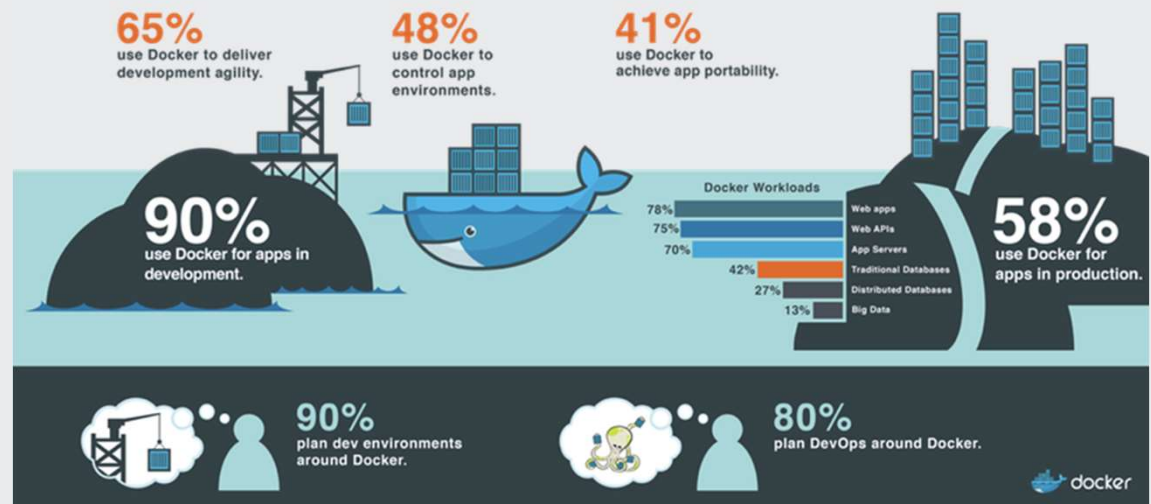
# 도커의 역사

```
$ docker images
NAME          ID                CREATED          PARENT
base          fad7c890e7ca7be4 2 days ago
busybox       c5860e37a1bcb678 20 hours ago
$ docker run -a busybox echo hello world
hello world
$
```



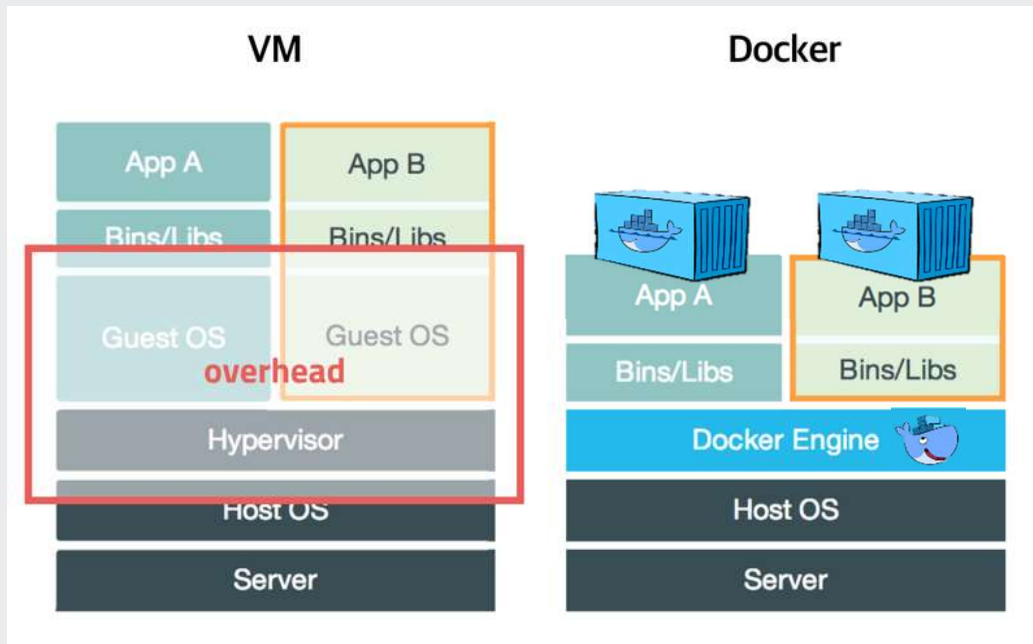
Pycon2013, dotCloud의 창업자 Solomon Hykes

## The future of Linux Containers 세션 이후 빠르게 성장한 도커



## Part 2

# 도커란?



컨테이너: 격리된 공간에서 프로세스가 동작하는 기술

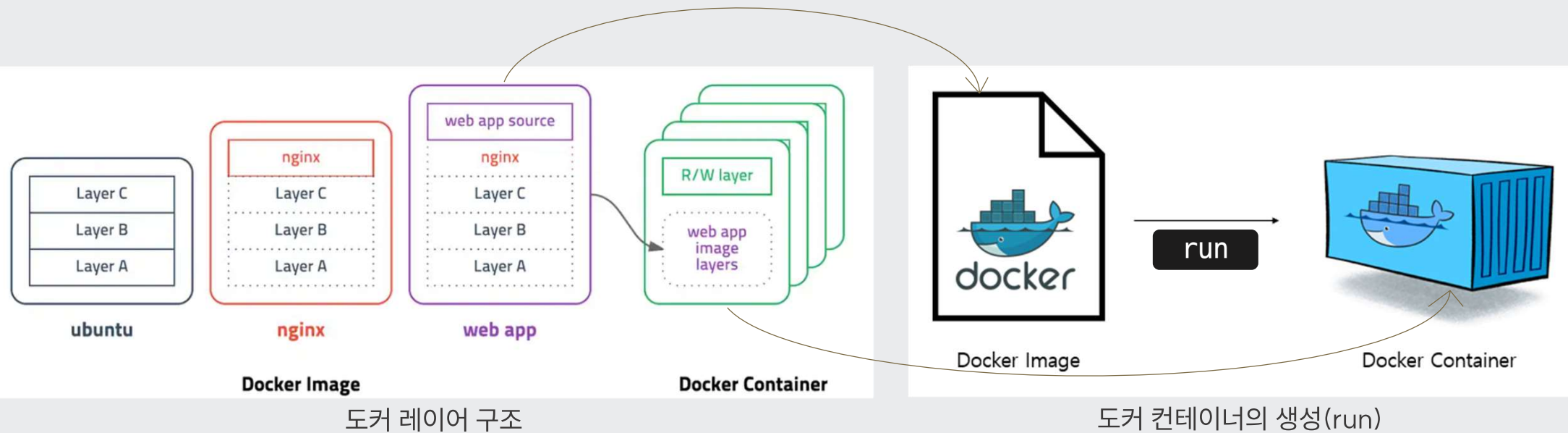
도커: 컨테이너 기반의 오픈소스 가상화 플랫폼

가상화: 물리적인 HW 객체를 논리적인 객체로 추상화  
한 대의 물리적인 서버를 마치 여러 대의 서버처럼 활용  
or 여러 대의 서버를 하나의 서버처럼 묶어서 사용



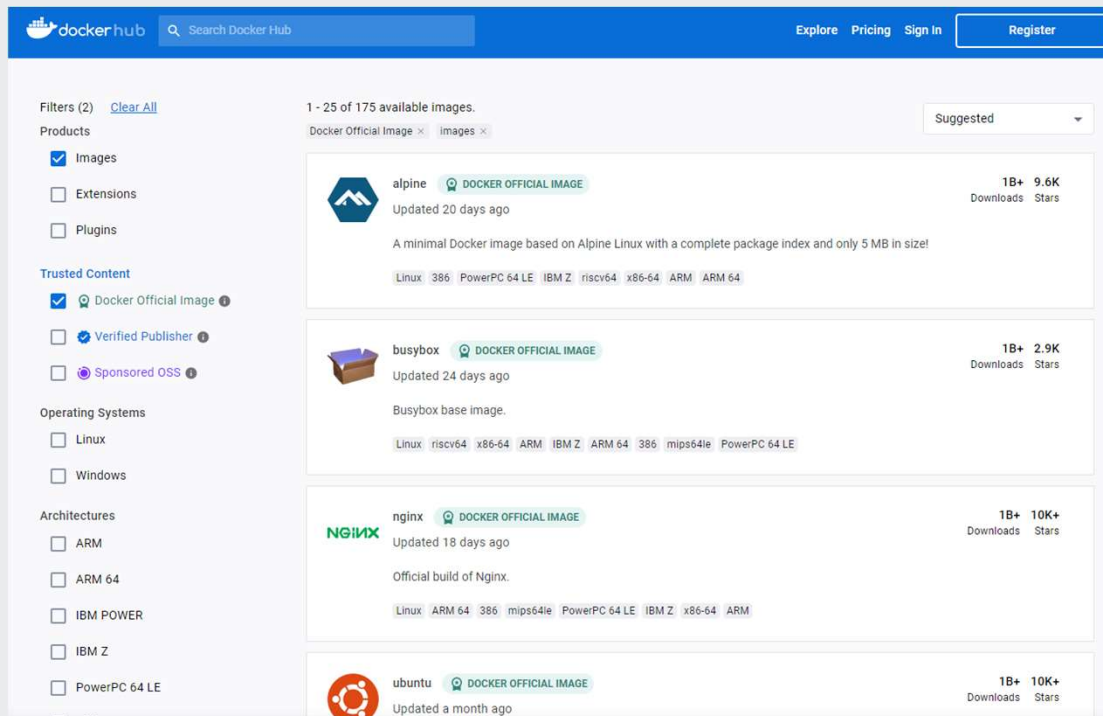
## Part 2

# 도커 이미지란?

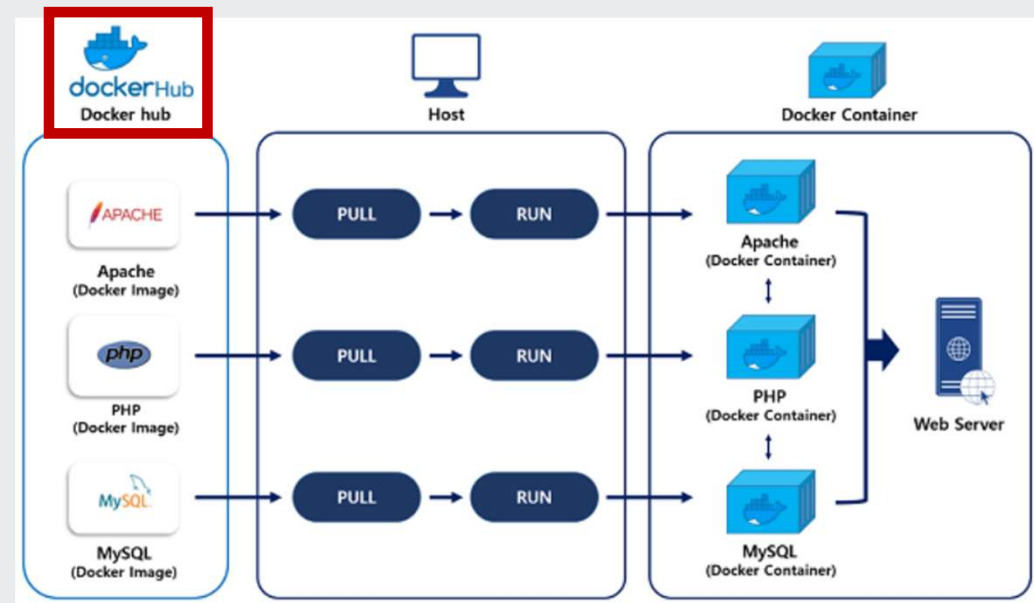


## Part 2

# 도커 허브란?

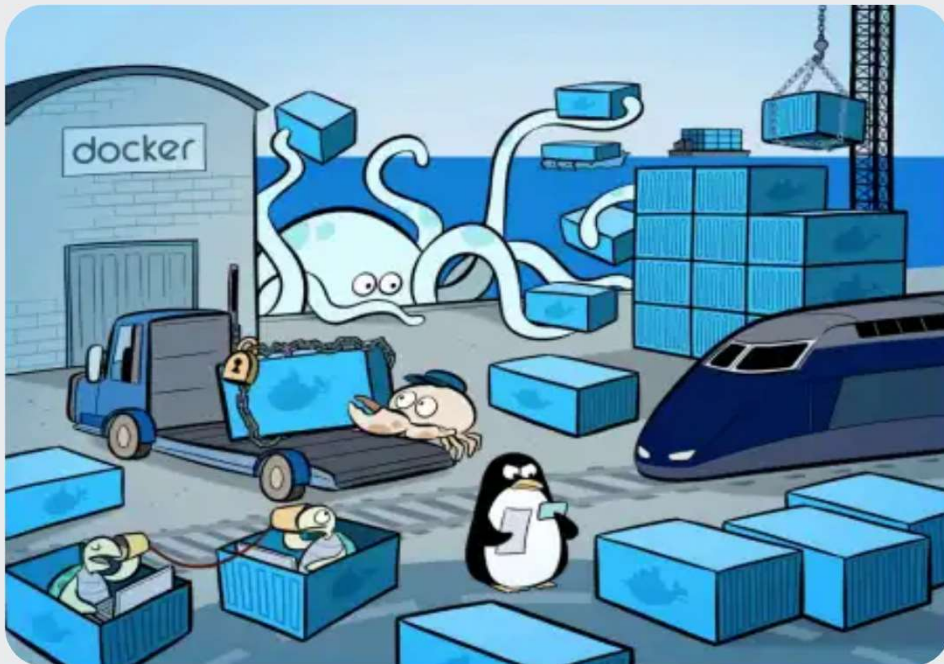


도커에서 제공하는 컨테이너 이미지 저장소



## Part 2

# 도커의 장점



### 빠른 속도 (레이어 방식)

- 새로운 컨테이너를 만드는데 걸리는 시간 겨우 1-2초
- 하나의 서버에 여러 개의 컨테이너를 수행하더라도 독립적으로 실행되어 가벼운 VM을 쓰는 느낌

### 이식성이 좋음

- 개발자의 코드를 컨테이너화
- 개발환경과 운영 환경을 맞추는데 시간버릴 필요 X

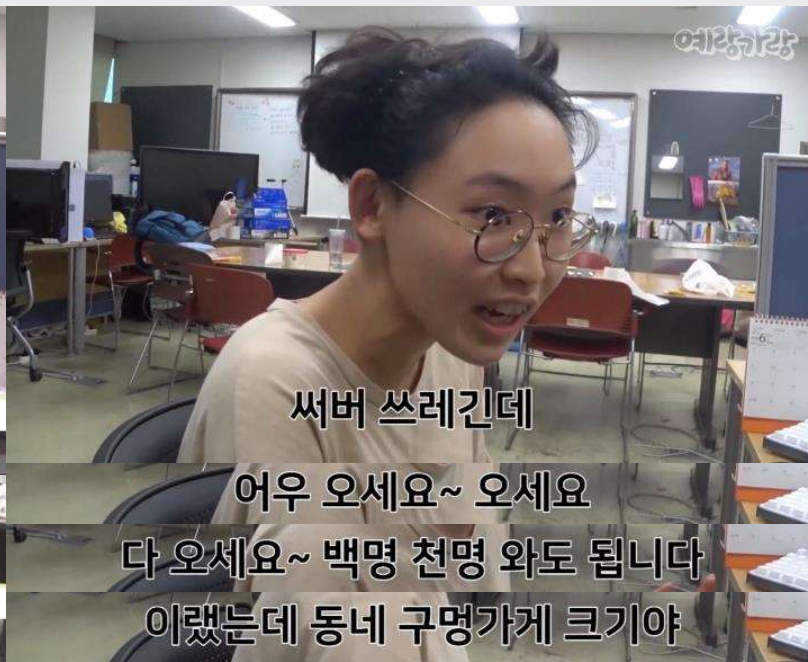
### 적은 용량으로 빠른 확장이 가능함

- 스케일 인/아웃이 쉬움  
(이전에는 OS와 애플리케이션이 묶여서 스케일 아웃 진행)
- 컨테이너 구조에서는 컨테이너(애플리케이션)만 추가하면 됨





전설의 티켓팅 백침 짤



MSA + Docker 로 해결하자!

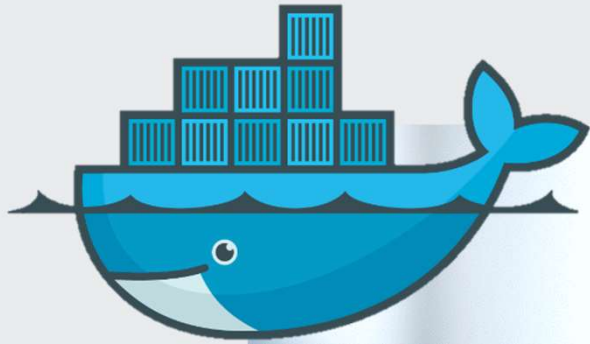




# 도커의 장점-스케일 인/아웃

SCALE UP (스케일업)	VS.	SCALE OUT (스케일아웃)
	구성	
CPU 변경, RAM 추가 등으로 하드웨어 장비의 성능을 높임 수직 확장이며, 성능 확장에 한계가 있음	확장성	하나의 장비에서 처리하던 일을 여러 장비에 나눠서 처리함 수평 확장이며, 지속적 확장 가능
성능 증가에 따른 비용 증가폭이 큼	비용	비교적 저렴한 서버 사용으로 비용 부담이 적음
한 대의 서버에 부하가 집중되어 장애 영향도가 큼	장애	읽기/쓰기가 여러 대의 서버에 분산 처리 장애 시 전면 장애의 가능성이 적음





Moby Dock 너란 고래... 너무 멋있잖나?



## Part 3

# 간단한 도커 사용법

- Docker 버전 확인

```
# docker version
```

- 컨테이너 목록 확인

```
# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

- 도커 이미지 목록 확인

```
# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
------------	-----	----------	---------	------



## Part 3

# 간단한 도커 사용법

- Docker 이미지 받아오기: `docker pull <IMAGE_NAME>`

```
$ docker pull centos
Using default tag: latest
latest: Pulling from library/centos
8a29a15cefae: Pull complete
Digest:
sha256:fe8d824220415eed5477b63addf40fb06c3b049404242b31982106ac204f6700
Status: Downloaded newer image for centos:latest
docker.io/library/centos:latest
```

- Pull 이후 이미지 목록 확인

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
centos	latest	470671670cac	8 weeks ago	237MB



## Part 3

# 간단한 도커 사용법

- 컨테이너 실행

```
$ docker run -it centos:latest bash  
[root@d3fef9c0f9e9 /]#
```

- Pull 해온 이미지가 없는 상태에서 컨테이너 실행

```
$ docker run -it -rm centos:latest bash  
Unable to find image 'centos:latest' locally  
latest: Pulling from library/centos  
8d30e94188e7: Pull complete  
Digest:  
sha256:2ae0d2c881c7123870114fb9cc7afabd1e31f9888dac8286884f6cf59373ed9b  
Status: Downloaded newer image for centos:latest  
  
[root@881189373f8b /]#
```



## Part 3

# 간단한 도커 사용법

- Dockerfile 작성

```
FROM ubuntu:18.04
LABEL maintainer="Sumin Lee <sumin@gmail.com>"
# install apache
RUN apt-get update &&\
    apt-get install -y apache2
RUN echo "TEST WEB" > /var/www/html/index.html
EXPOSE 80
CMD ["/usr/sbin/apache2ctl", "-DFOREGROUND"]
```

- Build로 이미지 생성

```
$ docker build -t webserver:v1 .
```

- 이미지 실행하여 컨테이너 생성

```
$ docker run -d -p 80:80 --name web webserver:v1
```





## Part 3

# 간단한 도커 사용법

- 컨테이너 삭제, 이미지 삭제

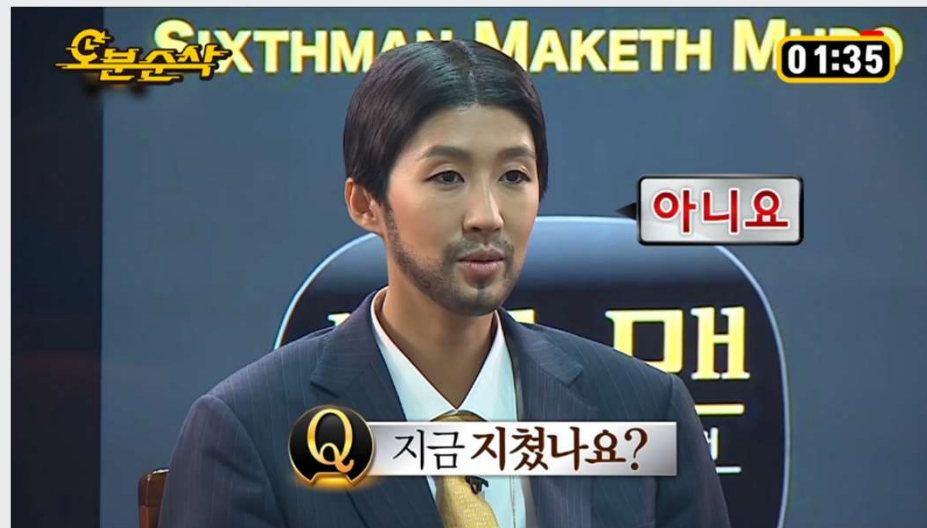
```
$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                NAMES
2a00b9b2b7cc   webserver:v1   "/usr/sbin/apache2ct...  1 minutes ago Up 1 minutes   0.0.0.0:80->80/tcp    web

$ docker rm -f web
web

$ docker rmi webserver:v1
Untagged: wevserver:v1
Deleted: sha256:487a3619305e68483059caa21eb54d1d812ced4282df9e2ba05ec46ed9a2b8f4
Deleted: sha256:9b6621e819f094c16ea9f63af90f7cb564a48133c05504fad0f28563865f957d
```



이론 설명은 여기까지 입니다...



## Part 4

# 도커의 미래

컨테이너 오케스트레이션: 여러 서버에 걸친 다수의 컨테이너를 사용하는 환경설정을 관리하는 것



# kubernetes

: 다수의 컨테이너를 배포, 관리, 확장할 때 수반되는 다수의 수동 프로세스를 자동화하는  
오픈소스 컨테이너 오케스트레이션 플랫폼

### KUBERNETES IS DEPRECATING DOCKER AS A CONTAINER RUNTIME AFTER V1.20

The screenshot shows the Kubernetes Blog page with the article "Don't Panic: Kubernetes and Docker" dated Wednesday, December 02, 2020. The article authors are Jorge Castro, Duffie Cooley, Kat Cosgrove, Justin Garrison, Noah Kantrowitz, Bob Killen, Rey Lejano, Dan "POP" Papandrea, Jeffrey Sica, and Davanum "Dims" Srinivas. The article states that Kubernetes is deprecating Docker as a container runtime after v1.20. It reassures users that they do not need to panic and that Docker can still be used as a development tool. It also mentions that Docker support will be removed in a future version of Kubernetes and that users of managed services like GKE, EKS, or AKS will need to ensure their worker nodes are using a supported container runtime before Docker support is removed.

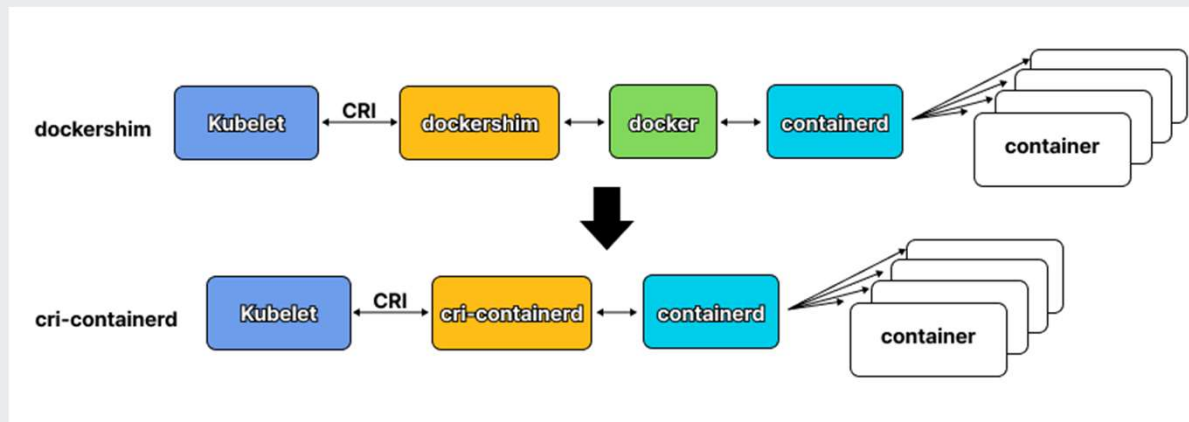
쿠버네티스, v1.20 부터 도커 지원 중단!?!?!?



## Part 4

# 도커의 미래

도커가 생성하는 이미지는 도커에만 특정된 이미지가 아닌  
OCI(Open Container Initiative)와 호환되는 이미지



CRI를 준수하는 다른  
컨테이너 런타임으로 바꾸면 OK!

도커가 **CRI**를 사용하는 런타임으로 더 이상 사용되지 않는 것일 뿐,  
도커를 더 이상 개발 도구로 쓸 수 없다는 뜻은 아니다!



도커 대신 이미지를 빌드할 수 있는 툴:  
kaniko, buildash, Podman, Skopeo 등

 감사합니다 

