Docker 소개

김기훈

Agenda

- Docker 소개
- Installation on Linux
- Installation on Windows (Boot2docker)
- Docker Commands
- dockerfile
- Docker Container link
- Docker Compose

Docker 소개

About Docker

Docker is a **platform** for developers and sysadmins to **develop, ship, and run applications**.

Docker lets you **quickly assemble applications** from components and eliminates the friction that can come when shipping code.
Docker lets you **get your code** tested and deployed into production **as fast as possible**.

# Docker Engine

our lightweight and powerful **open source container virtualization technology** combined with a work flow for building and containerizing your applications.

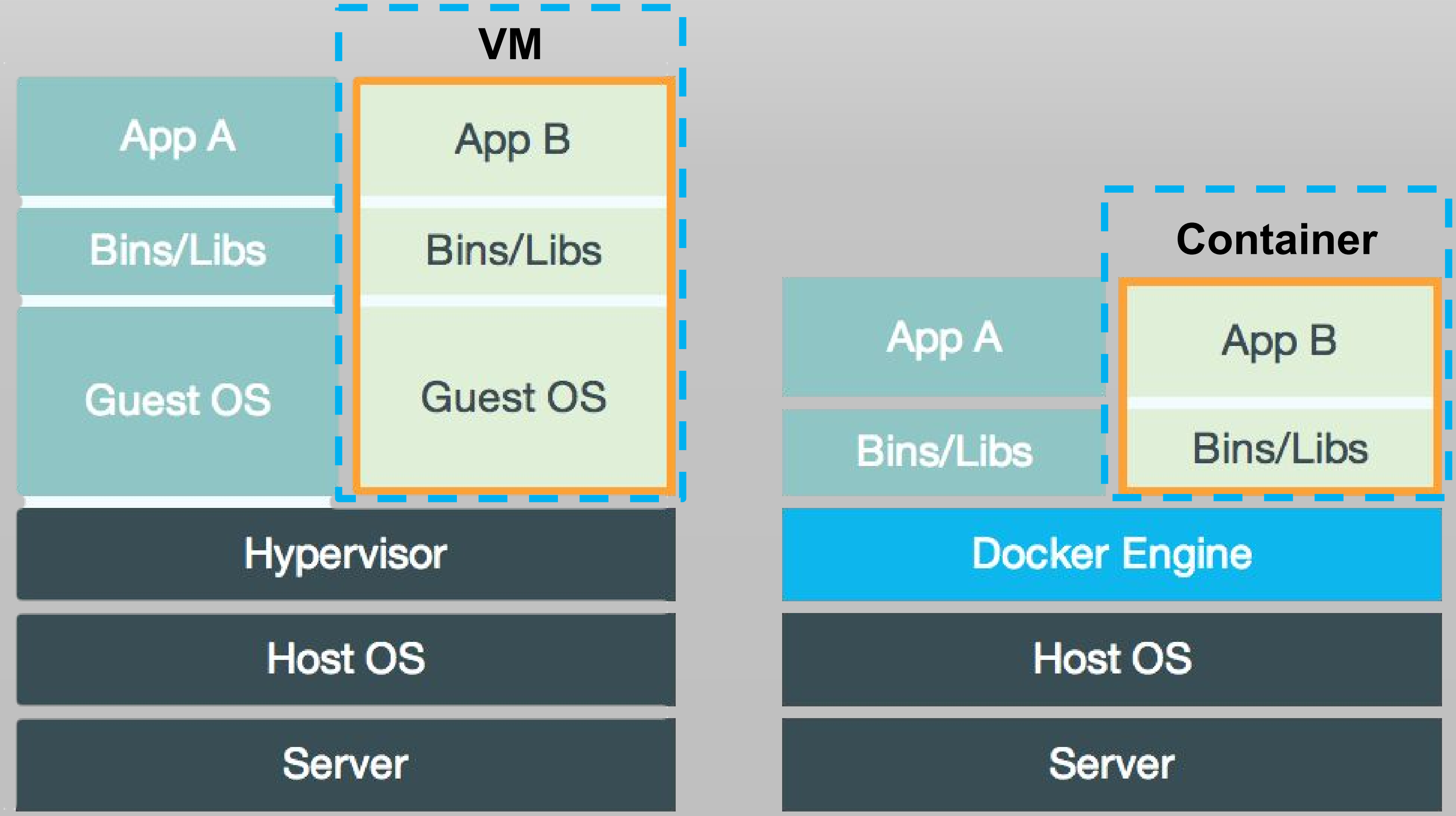
# Docker Hub

our SaaS service for **sharing and managing** your application stacks.

https://docs.docker.com/misc/

VM vs Container

Application에 고립된(Isolated) 환경을 제공하려는 목적은 동일하나 방법이 다름!



https://www.docker.com/whatisdocker

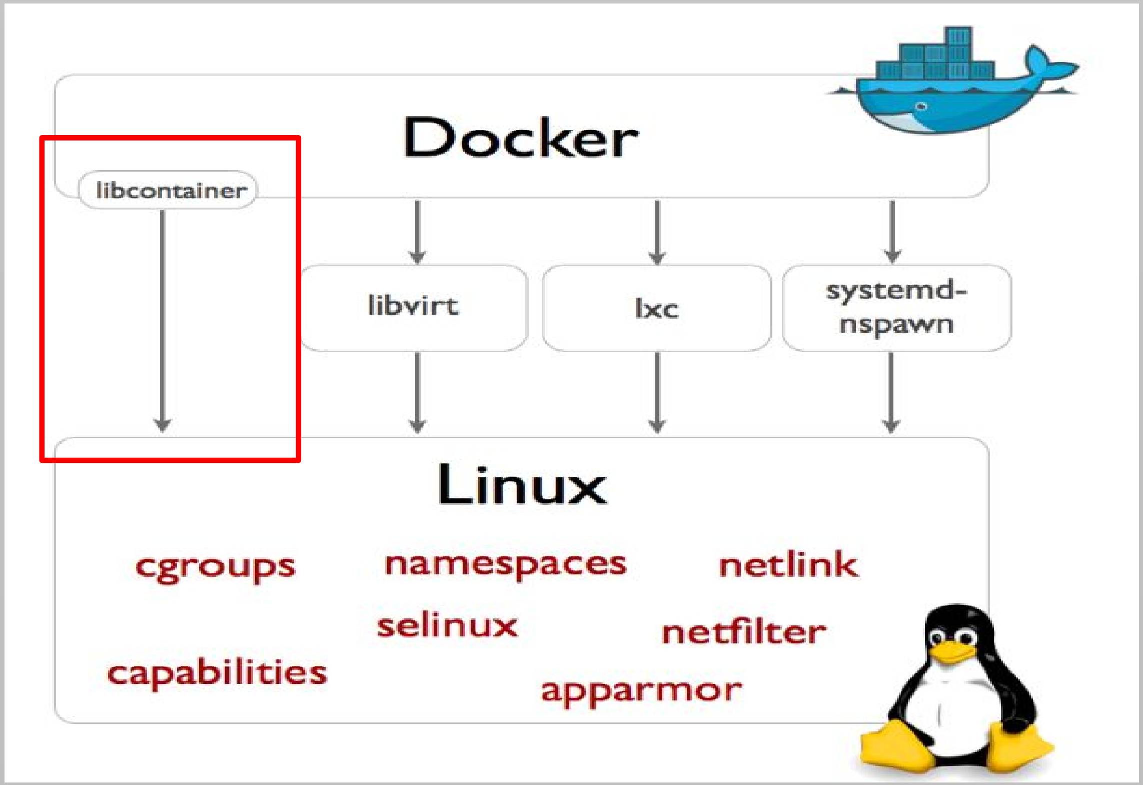
Linux Container

- LXC (Linux Containers) is **an operating-system-level virtualization environment** for running multiple **isolated** Linux systems (containers) on **a single Linux control host**.
- The **Linux kernel** provides the **cgroups** functionality that allows limitation and prioritization of resources (CPU, memory, block I/O, network, etc.) without the need for starting any virtual machines, and **namespace** isolation functionality that allows complete isolation of an applications' view of the operating environment, including process trees, networking, user IDs and mounted file systems.
- LXC combines kernel's cgroups and support for isolated namespaces to provide an isolated environment for applications.
- Docker can also use LXC as one of its execution drivers, enabling image management and providing deployment services.

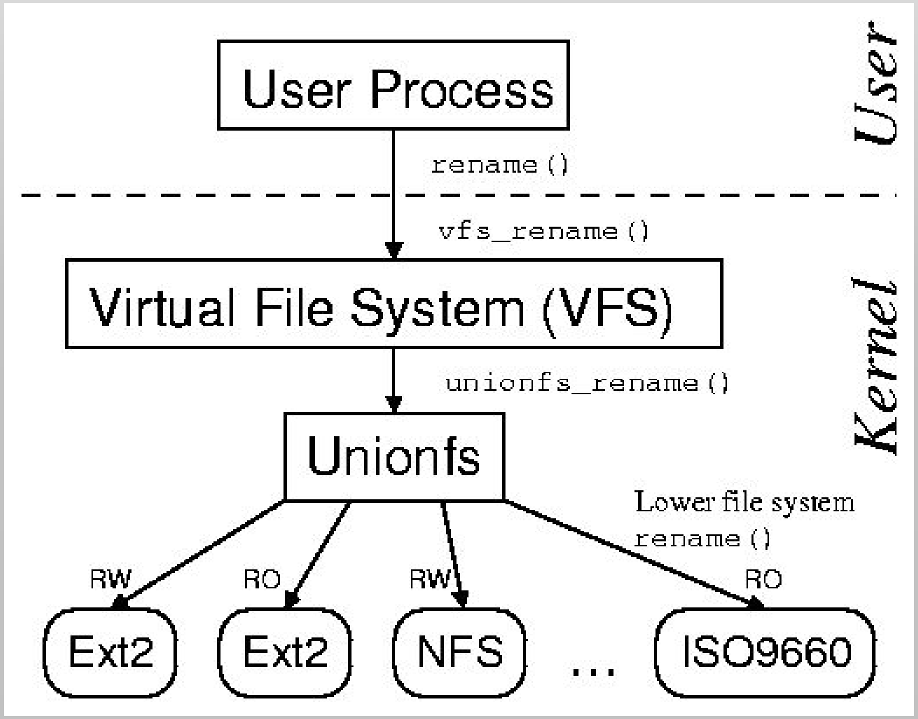
https://en.wikipedia.org/wiki/LXC

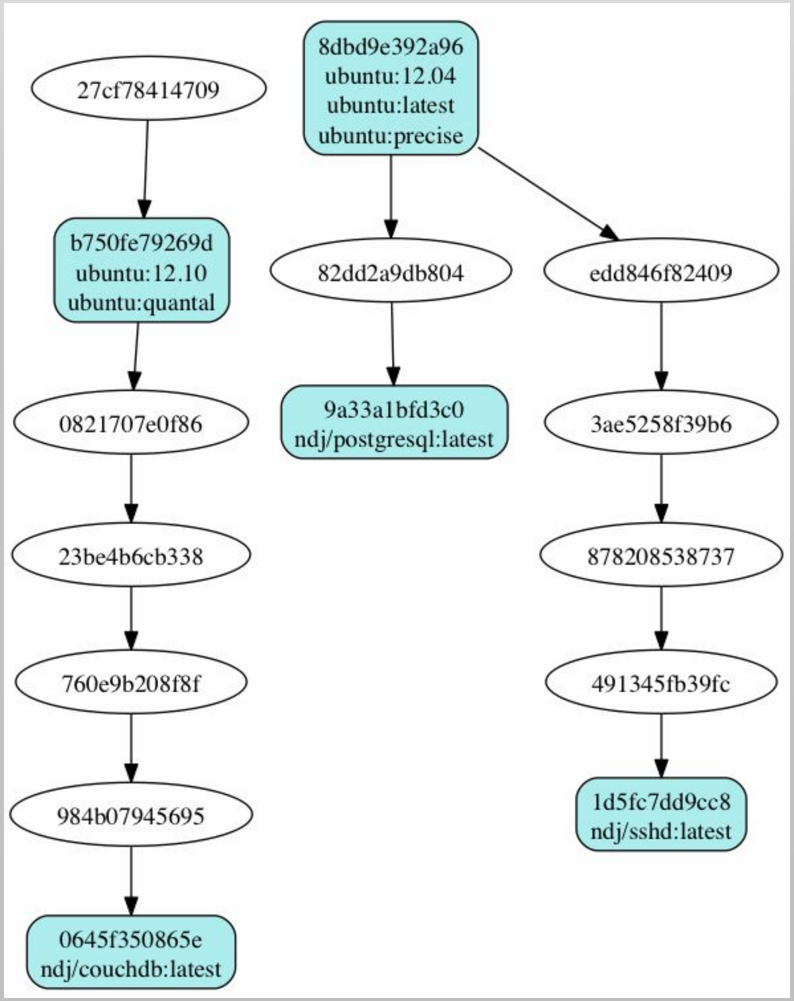
libcontainer

- without depending on LXC
- Docker version 0.9 ~

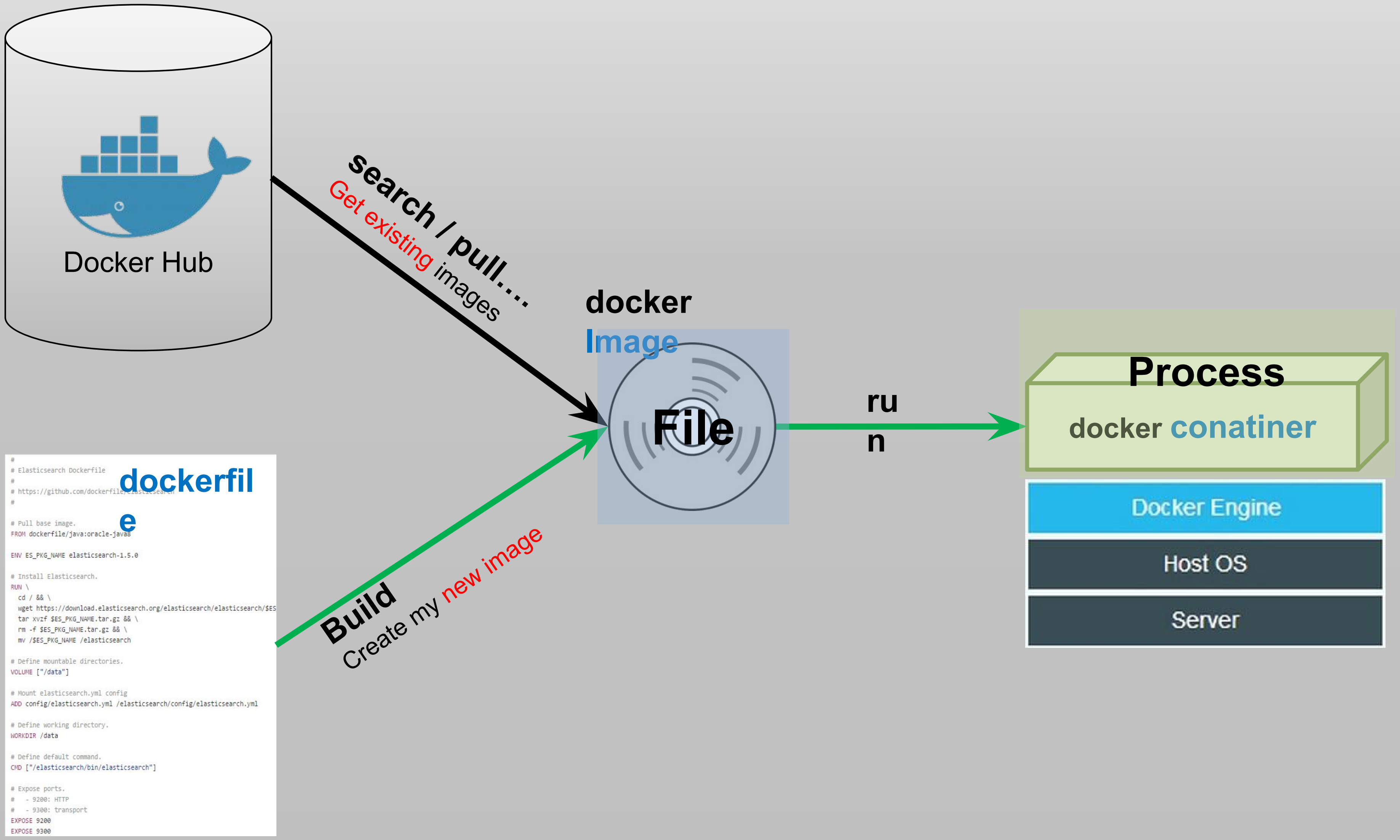


UnionFS


User Process
rename()
User
vfs_rename()
Virtual File System (VFS)
Kernel
unionfs_rename()
Unionfs
Lower file system
rename()
RW
RO
RW
RO
Ext2
Ext2
NFS
...
ISO9660

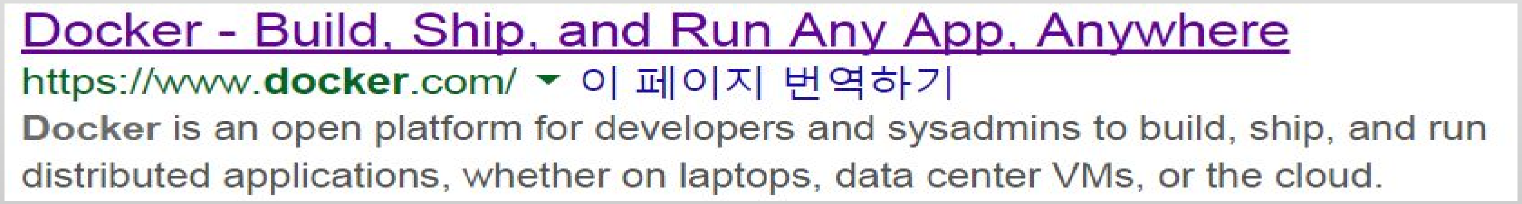

27cf78414709
b750fe79269d ubuntu:12.10 ubuntu:quantal
0821707e0f86
23be4b6cb338
760e9b208f8f
984b07945695
0645f350865e ndj/couchdb:latest
8dbd9e392a96 ubuntu:12.04 ubuntu:latest ubuntu:precise
82dd2a9db804
9a33a1bfd3c0 ndj/postgresql:latest
edd846f82409
3ae5258f39b6
878208538737
491345fb39fc
1d5fc7dd9cc8 ndj/sshd:latest

dockerfile vs Image vs Container



```
#
# Elasticsearch Dockerfile
#
# https://github.com/dockerfile/elasticsearch
#

# Pull base image.
FROM dockerfile/java:oracle-java8

ENV ES_PKG_NAME elasticsearch-1.5.0

# Install Elasticsearch.
RUN \
  cd / && \
  wget https://download.elasticsearch.org/elasticsearch/elasticsearch/$ES
  tar xvzf $ES_PKG_NAME.tar.gz && \
  rm -f $ES_PKG_NAME.tar.gz && \
  mv /$ES_PKG_NAME /elasticsearch

# Define mountable directories.
VOLUME ["/data"]

# Mount elasticsearch.yml config
ADD config/elasticsearch.yml /elasticsearch/config/elasticsearch.yml

# Define working directory.
WORKDIR /data

# Define default command.
CMD ["/elasticsearch/bin/elasticsearch"]

# Expose ports.
#   - 9200: HTTP
#   - 9300: transport
EXPOSE 9200
EXPOSE 9300
```

Docker가 원하는 것!


Docker - Build, Ship, and Run Any App, Anywhere
https://www.docker.com/ ▾ 이 페이지 번역하기
Docker is an open platform for developers and sysadmins to build, ship, and run distributed applications, whether on laptops, data center VMs, or the cloud.

- **Build** : 컨테이너 안에 어플리케이션을 담아!
- **Ship** : 컨테이너를 여러 곳에 배포해!
- **Run** : 컨테이너를 실행해!
- **Any App** : Linux에서 동작하는 어플리케이션!
- **Anywhere** : Laptop! Virtual Machine! Cloud! 등등!

"어플리케이션을 컨테이너에만 담아두면
어느 곳에든 쉽고 빠르게 배포하고 실행 할 수 있게 해줄게!!"

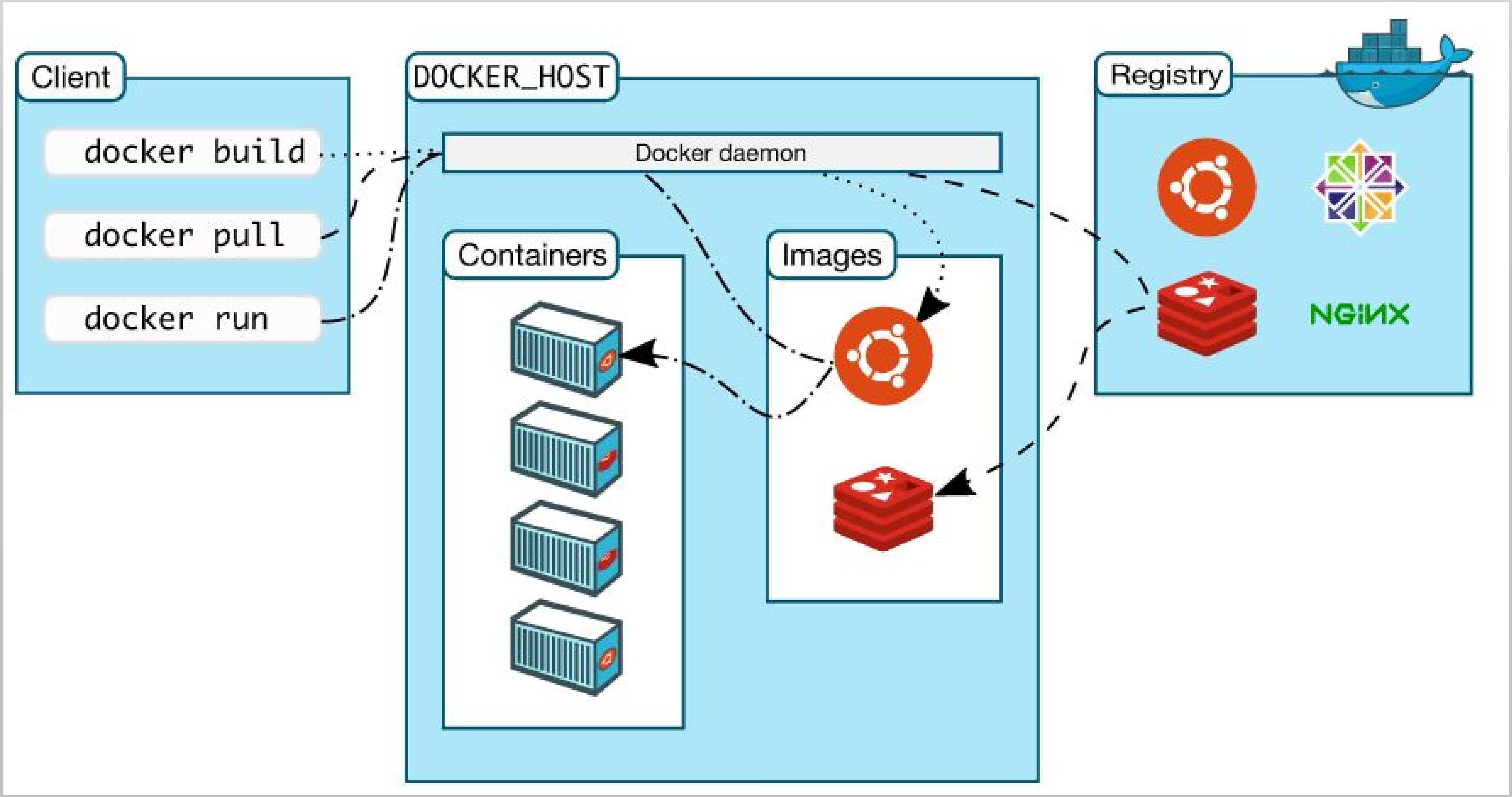
정말 Anywhere……?

- only **64** bits OS
- kernels version
- Windows, Mac OS…… coming soon..
- Docker is supported on the following versions
  - Red Hat Enterprise Linux 7, Linux 6.6 or later
  - CentOS 7.X, 6.5 or higher
  - Ubuntu 14.04 (LTS), 12.04 (LTS), 13.10
  - 나머지 확인 : https://docs.docker.com/installation

Why Docker?

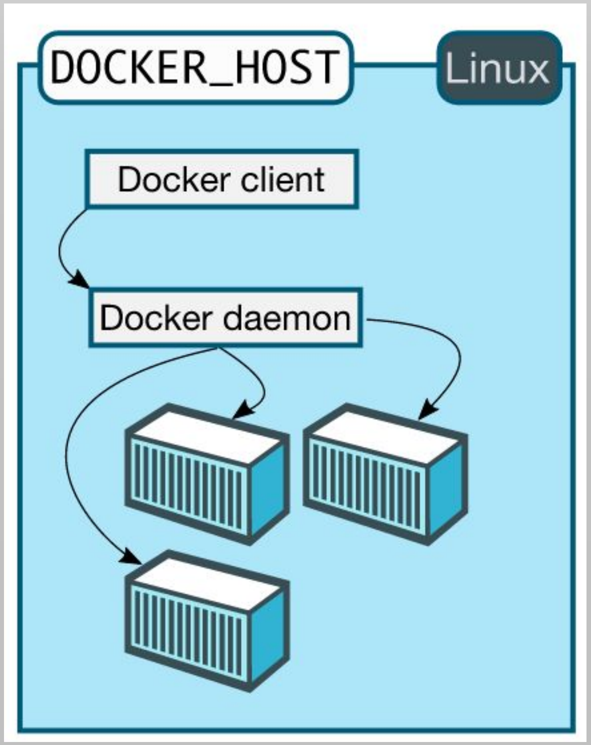
- Faster delivery of your applications
- Deploy and scale more easily
- Get higher density and run more workloads
- Faster deployment makes for easier management

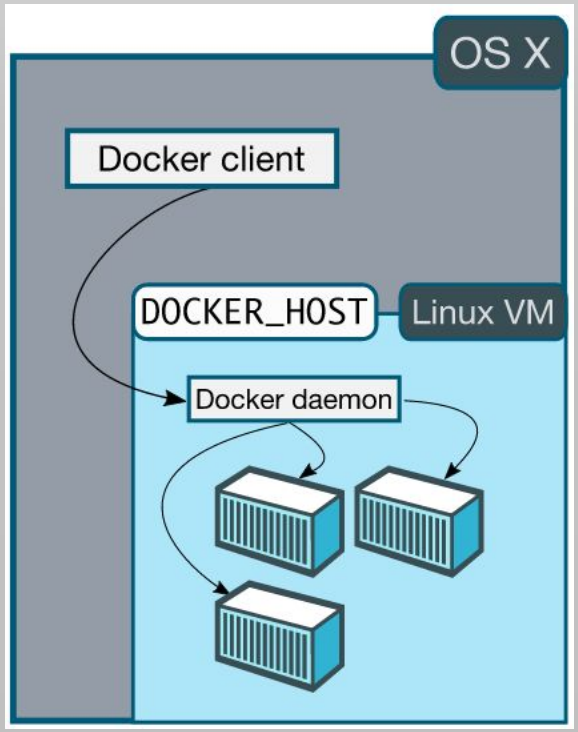
Docker's architecture

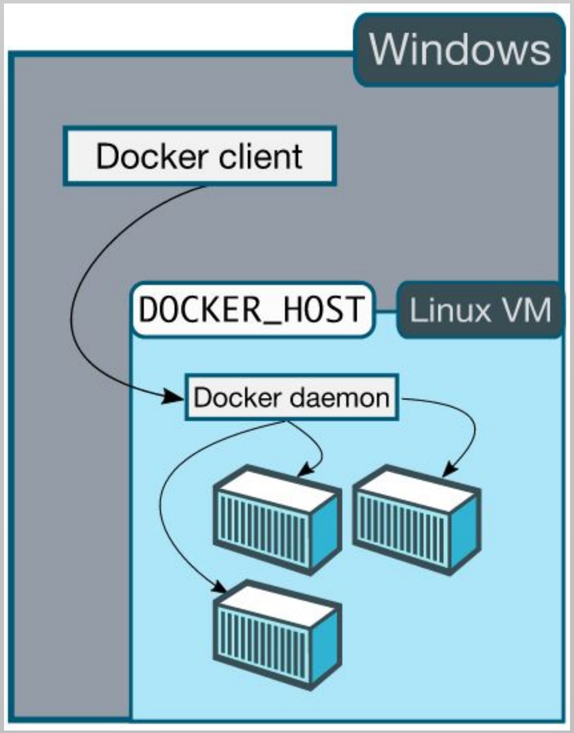


https://docs.docker.com/introduction/understanding-docker/

Docker on Linux, Mac, Windows







Installation on Linux

https://docs.docker.com/installation/ubuntulinux/

Proxy setting

# Proxy

```
$ sudo vi /etc/environment
    http_proxy=http://host:port
    https_proxy=http://host:port
    ftp_proxy=http://host:port
$ sudo vi /etc/apt/apt.conf
    Acquire::http::proxy "http://host:port";
    Acquire::ftp::proxy "ftp://host:port";
    Acquire::https::proxy "https://host:port";
$ sudo apt-get update
```

# 인증서

```
$ sudo cp mycert.crt /usr/local/share/ca-certificates/
$ update-ca-certificates
```

Ubuntu Trusty 14.04(LTS)

```
# Installation
$ wget -qO- https://get.docker.com/ | sh

# docker proxy
$ sudo vi /etc/default/docker
    export http_proxy=“http://host:port”
$ sudo service docker restart

# Uninstallation
$ sudo apt-get purge lxc-docker
$ rm -rf /var/lib/docker
```

Installation on Windows

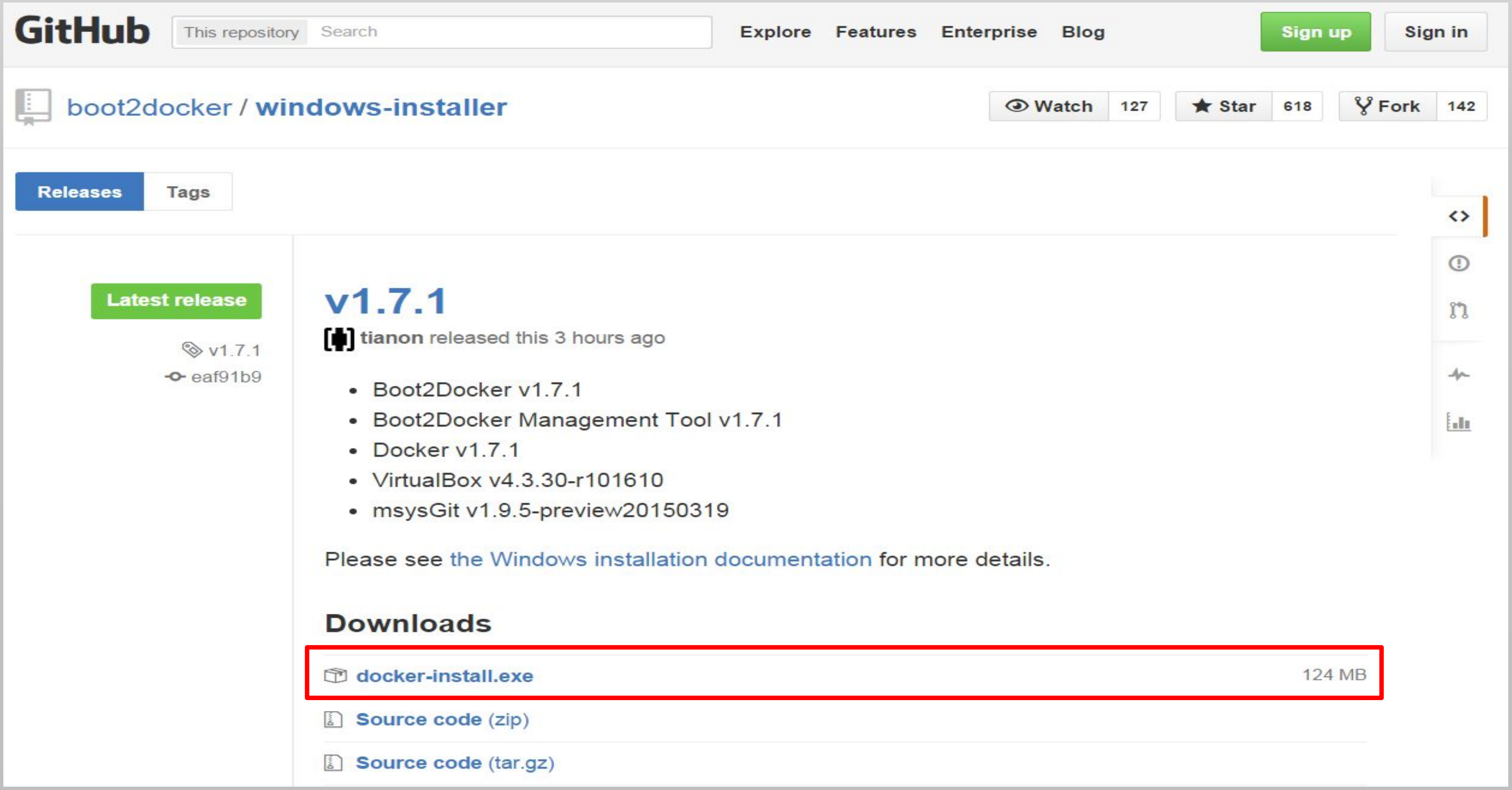
https://docs.docker.com/installation/windows/

Boot2Docker 설치

https://github.com/boot2docker/boot2docker

Download Boot2Docker

https://github.com/boot2docker/windows-installer/releases/latest



v1.7.1

tianon released this 3 hours ago

- Boot2Docker v1.7.1
- Boot2Docker Management Tool v1.7.1
- Docker v1.7.1
- VirtualBox v4.3.30-r101610
- msysGit v1.9.5-preview20150319

Please see the Windows installation documentation for more details.

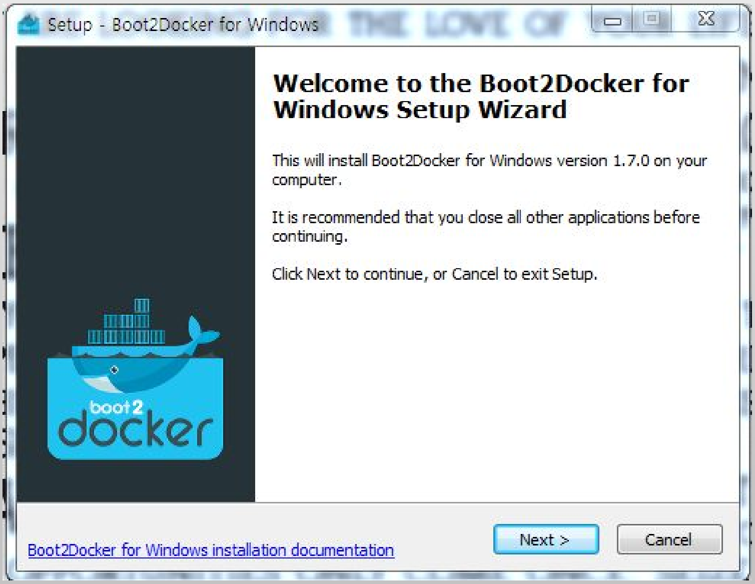
Downloads

docker-install.exe 124 MB

Source code (zip)

Source code (tar.gz)

1. Welcome to the Boot2Docker



Setup - Boot2Docker for Windows

Welcome to the Boot2Docker for Windows Setup Wizard

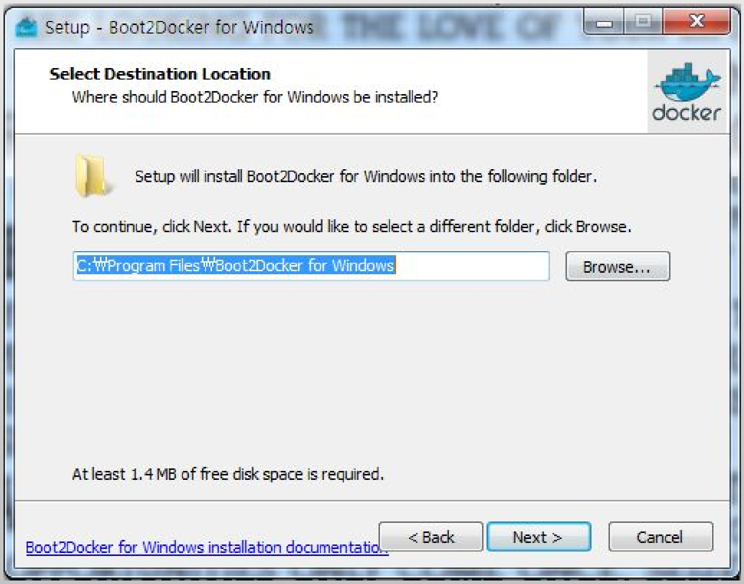
This will install Boot2Docker for Windows version 1.7.0 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Boot2Docker for Windows installation documentation

2. Select Destination Location



Setup - Boot2Docker for Windows

Select Destination Location

Where should Boot2Docker for Windows be installed?

Setup will install Boot2Docker for Windows into the following folder.

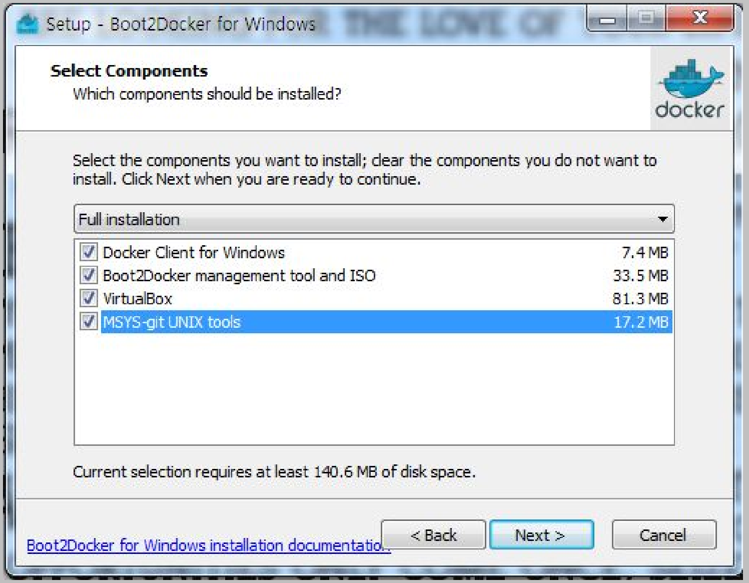
To continue, click Next. If you would like to select a different folder, click Browse.

C:₩Program Files₩Boot2Docker for Windows

At least 1.4 MB of free disk space is required.

Boot2Docker for Windows installation documentatio

3. Select Components



Setup - Boot2Docker for Windows

Select Components

Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

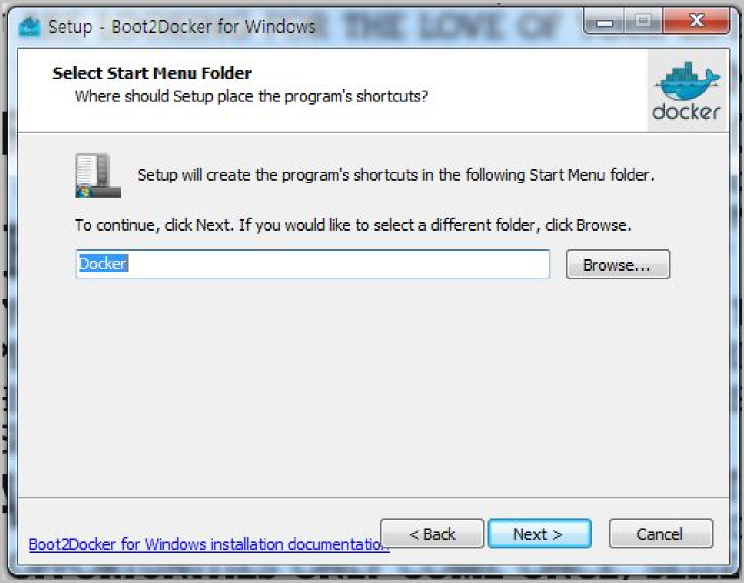
Full installation

✓	Docker Client for Windows	7.4 MB
✓	Boot2Docker management tool and ISO	33.5 MB
✓	VirtualBox	81.3 MB
✓	MSYS-git UNIX tools	17.2 MB

Current selection requires at least 140.6 MB of disk space.

Boot2Docker for Windows installation documentatio

4. Select Start Menu Folder



Setup - Boot2Docker for Windows

Select Start Menu Folder

Where should Setup place the program's shortcuts?

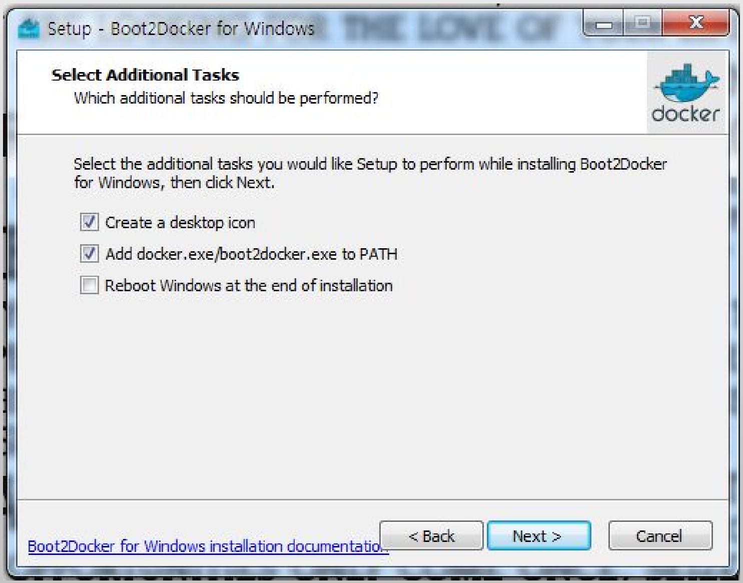
Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Docker

Boot2Docker for Windows installation documentatio

5. Select Additional Tasks



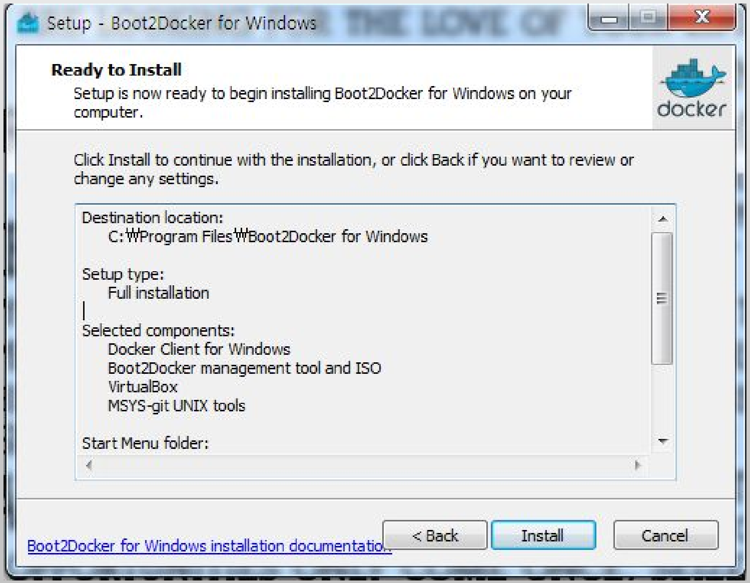
Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing Boot2Docker for Windows, then click Next.

- [✓] Create a desktop icon
- [✓] Add docker.exe/boot2docker.exe to PATH
- [ ] Reboot Windows at the end of installation

Boot2Docker for Windows installation documentation

6. Ready to Install



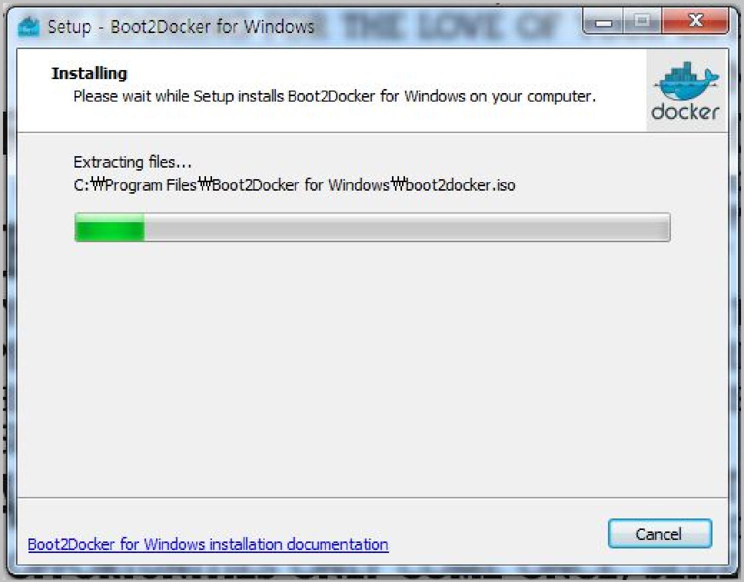
Ready to Install
Setup is now ready to begin installing Boot2Docker for Windows on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

```
Destination location:
    C:₩Program Files₩Boot2Docker for Windows

Setup type:
    Full installation

Selected components:
    Docker Client for Windows
    Boot2Docker management tool and ISO
    VirtualBox
    MSYS-git UNIX tools

Start Menu folder:
```

Boot2Docker for Windows installation documentation

7. Installing

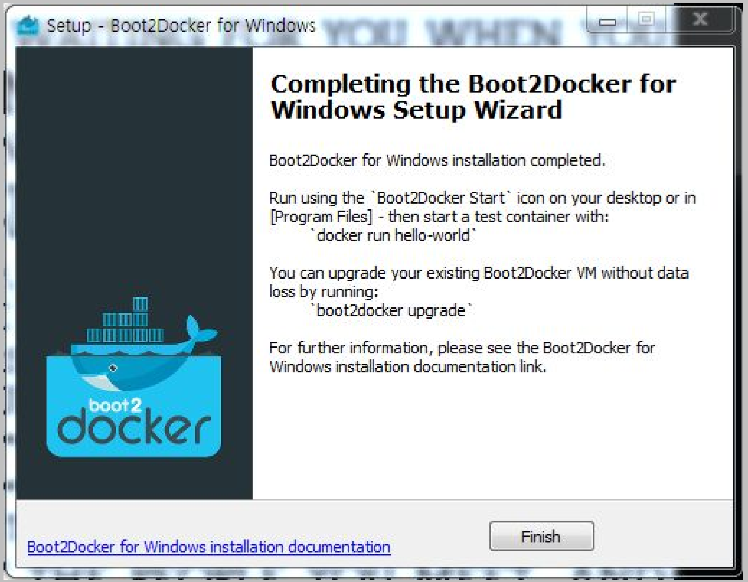


Installing
Please wait while Setup installs Boot2Docker for Windows on your computer.

Extracting files...
C:₩Program Files₩Boot2Docker for Windows₩boot2docker.iso

Boot2Docker for Windows installation documentation

8. Complete



Completing the Boot2Docker for Windows Setup Wizard

Boot2Docker for Windows installation completed.

Run using the `Boot2Docker Start` icon on your desktop or in [Program Files] - then start a test container with:
`docker run hello-world`

You can upgrade your existing Boot2Docker VM without data loss by running:
`boot2docker upgrade`

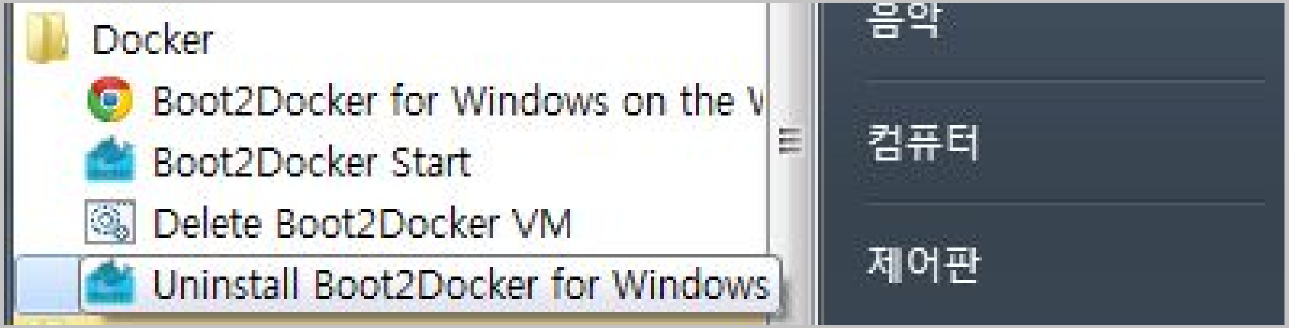
For further information, please see the Boot2Docker for Windows installation documentation link.

Boot2Docker for Windows installation documentation

Boot2Docker 삭제

Boot2docker 삭제

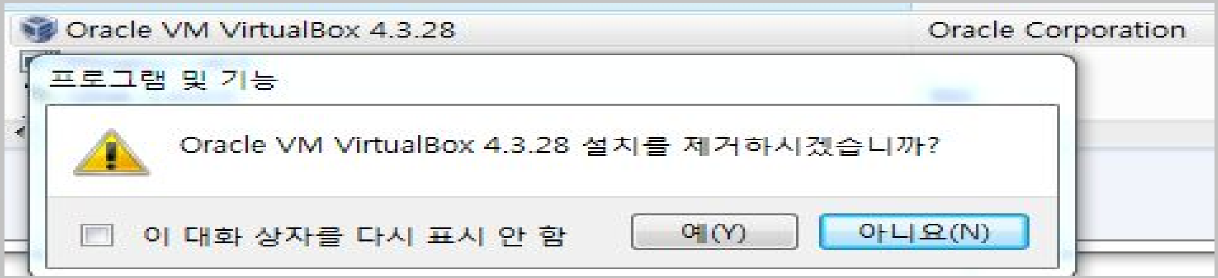
- Delete Boot2Docker VM
- Uninstall Boot2Docker for Windows


Docker
Boot2Docker for Windows on the V
Boot2Docker Start
Delete Boot2Docker VM
Uninstall Boot2Docker for Windows
음악
컴퓨터
제어판

VirtualBox 삭제

1. Delete C:\Windows\System32\drivers\Vbox*.sys

2. Uninstall Virtualbox



Oracle VM VirtualBox 4.3.28 Oracle Corporation

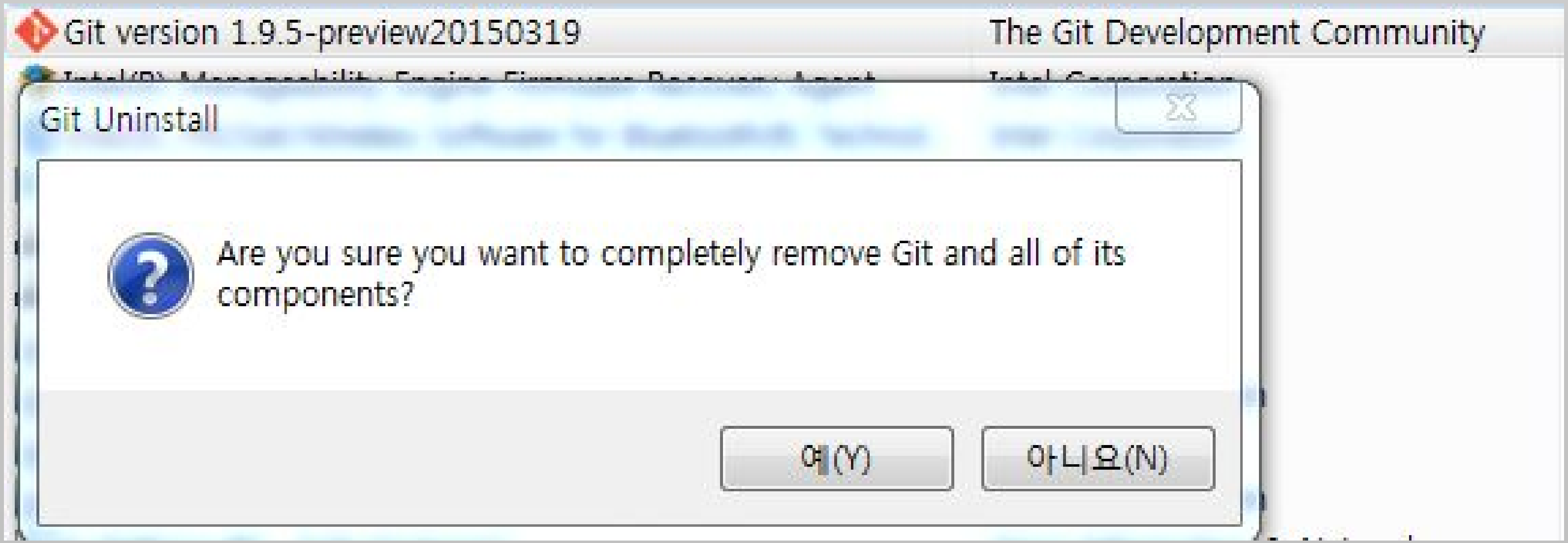
프로그램 및 기능

Oracle VM VirtualBox 4.3.28 설치를 제거하시겠습니까?

3. Delete HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Vbox*

Git 삭제

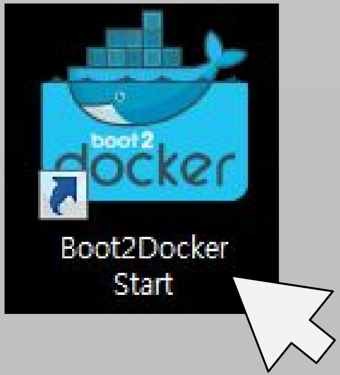
- Uninstall Git


Git version 1.9.5-preview20150319 The Git Development Community

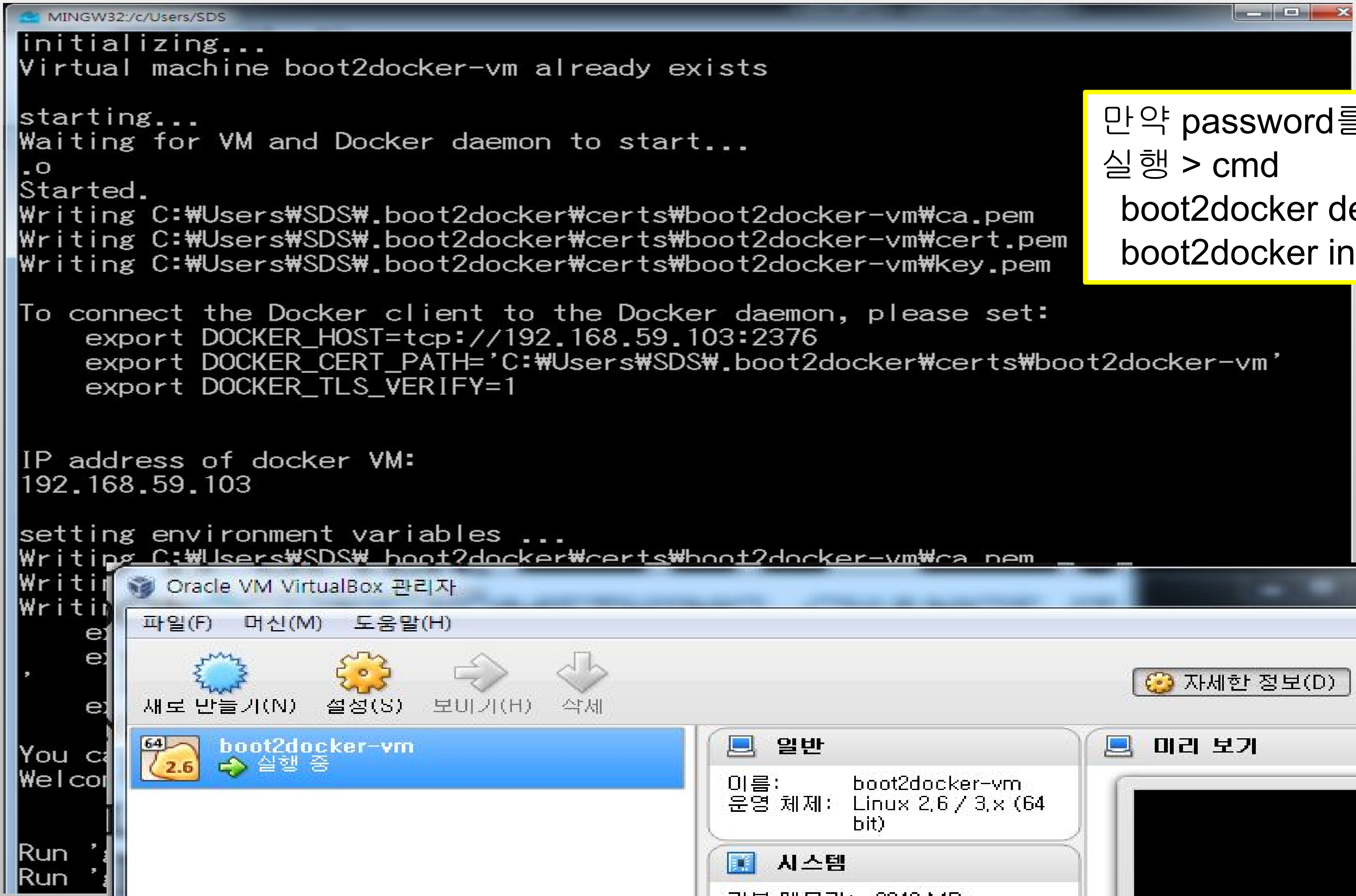
Git Uninstall

Are you sure you want to completely remove Git and all of its components?

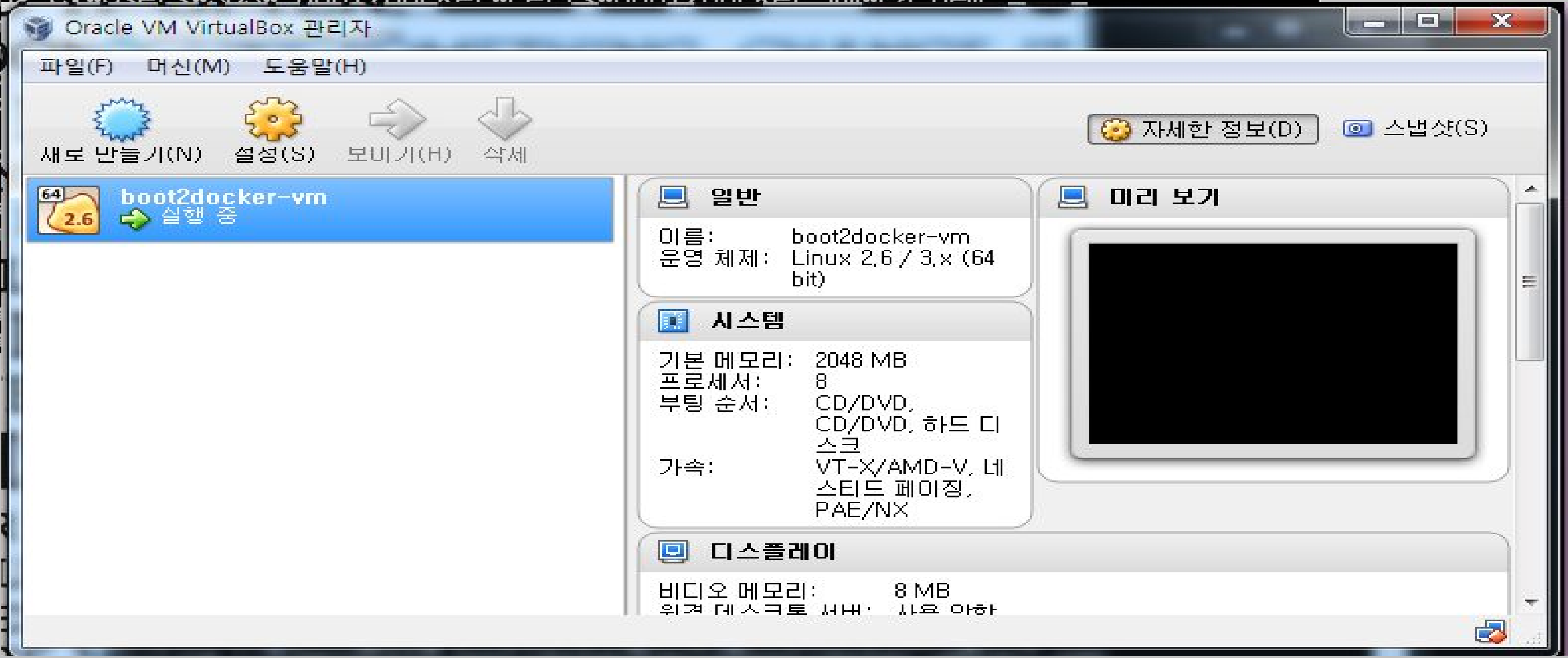
Boot2Docker 시작



만약 password를 묻는 다면?
실행 > cmd
  boot2docker delete
  boot2docker init

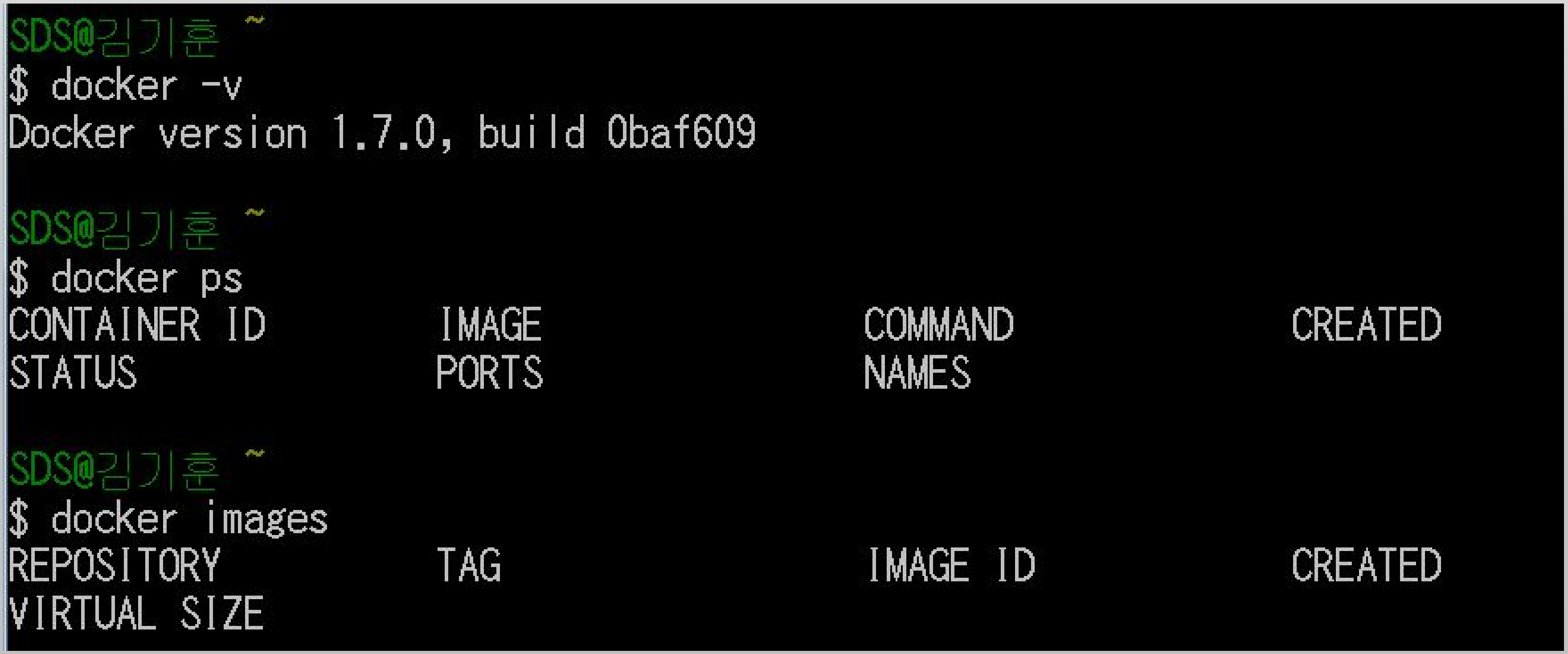


```
initializing...
Virtual machine boot2docker-vm already exists

starting...
Waiting for VM and Docker daemon to start...
.o
Started.
Writing C:₩Users₩SDS₩.boot2docker₩certs₩boot2docker-vm₩ca.pem
Writing C:₩Users₩SDS₩.boot2docker₩certs₩boot2docker-vm₩cert.pem
Writing C:₩Users₩SDS₩.boot2docker₩certs₩boot2docker-vm₩key.pem

To connect the Docker client to the Docker daemon, please set:
    export DOCKER_HOST=tcp://192.168.59.103:2376
    export DOCKER_CERT_PATH='C:₩Users₩SDS₩.boot2docker₩certs₩boot2docker-vm'
    export DOCKER_TLS_VERIFY=1


IP address of docker VM:
192.168.59.103

setting environment variables ...
Writing C:₩Users₩SDS₩.boot2docker₩certs₩boot2docker-vm₩ca.pem
Writi
Writi
    e
    e
,
    e

You c
Welco


Run '
Run '
```



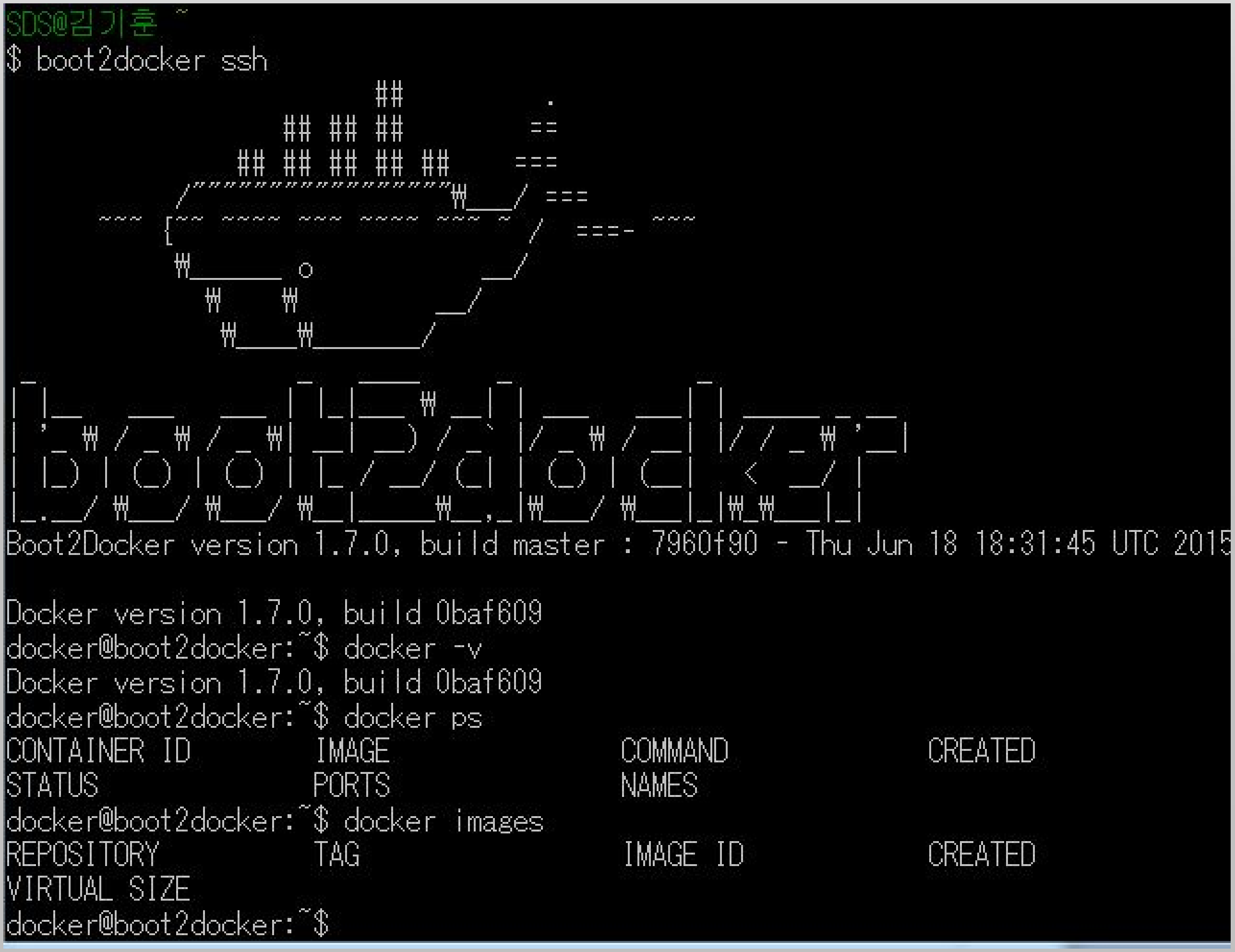
```
이름:          boot2docker-vm
운영 체제:     Linux 2.6 / 3.x (64
               bit)

기본 메모리:   2048 MB
프로세서:      8
부팅 순서:     CD/DVD,
               CD/DVD, 하드 디
               스크
가속:          VT-X/AMD-V, 네
               스티드 페이징,
               PAE/NX

비디오 메모리:          8 MB
원격 데스크톱 서버:     사용 안함
```

User ‘docker’ directly



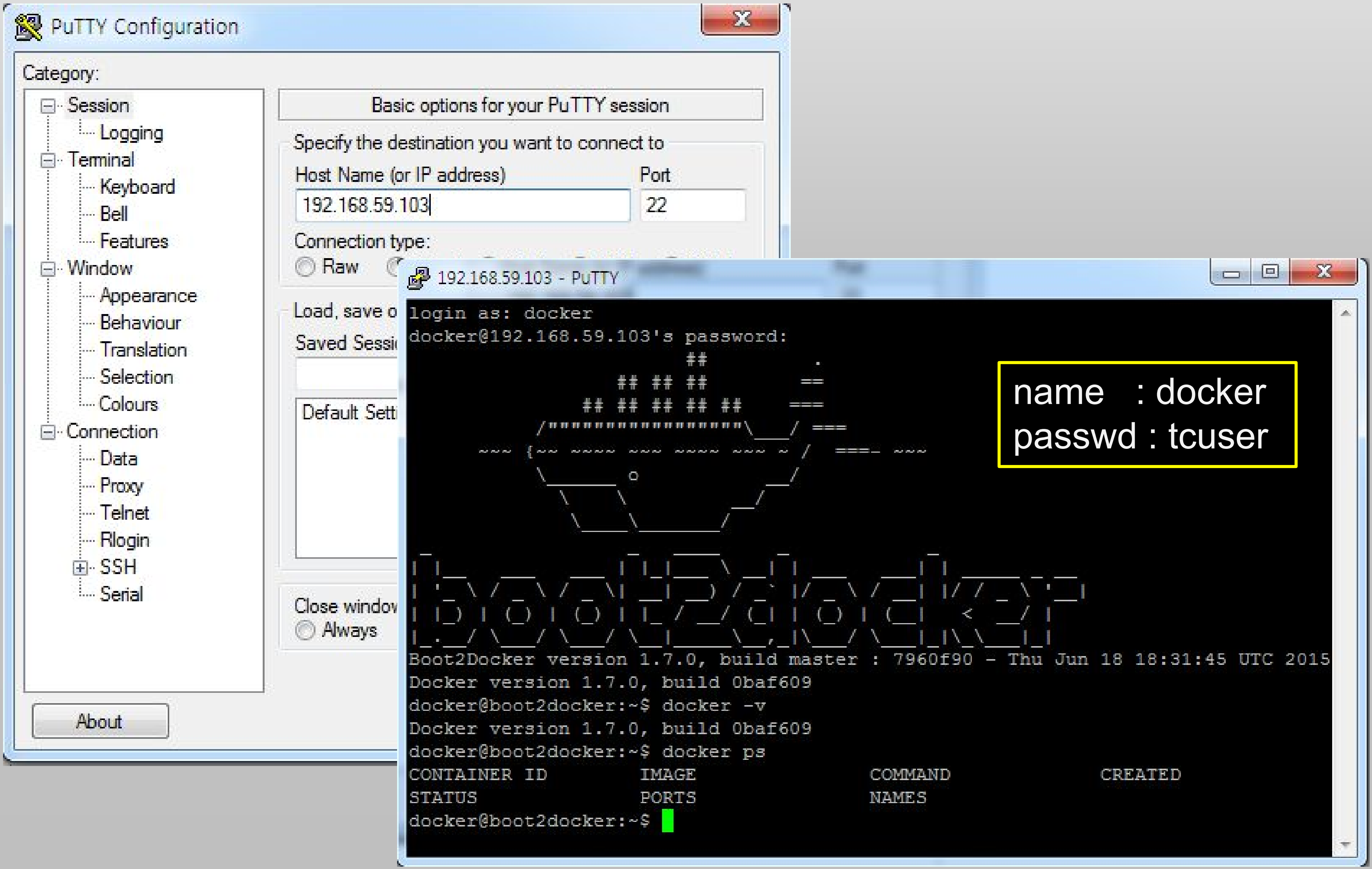
```
SDS@김기훈 ~
$ docker -v
Docker version 1.7.0, build 0baf609

SDS@김기훈 ~
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES

SDS@김기훈 ~
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
VIRTUAL SIZE
```

‘boot2docker ssh’ to log into the VM



```
SDS@김기훈 ~
$ boot2docker ssh
                        ##         .
                  ## ## ##        ==
               ## ## ## ## ##    ===
           /"""""""""""""""""₩___/ ===
      ~~~ {~~ ~~~~ ~~~ ~~~~ ~~~ ~ /  ===- ~~~
           ₩______ o           __/
             ₩    ₩         __/
              ₩____₩_______/
 _                 _   ____     _            _
| |__   ___   ___ | |_|___ ₩ __| | ___   ___| | _____ _ __
| '_ ₩ / _ ₩ / _ ₩| __| __) / _` |/ _ ₩ / __| |/ / _ ₩ '__|
| |_) | (_) | (_) | |_ / __/ (_| | (_) | (__|   <  __/ |
|_.__/ ₩___/ ₩___/ ₩__|_____₩__,_|₩___/ ₩___|_|₩_₩___|_|
Boot2Docker version 1.7.0, build master : 7960f90 - Thu Jun 18 18:31:45 UTC 2015

Docker version 1.7.0, build 0baf609
docker@boot2docker:~$ docker -v
Docker version 1.7.0, build 0baf609
docker@boot2docker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES
docker@boot2docker:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
VIRTUAL SIZE
docker@boot2docker:~$
```

ssh using Putty

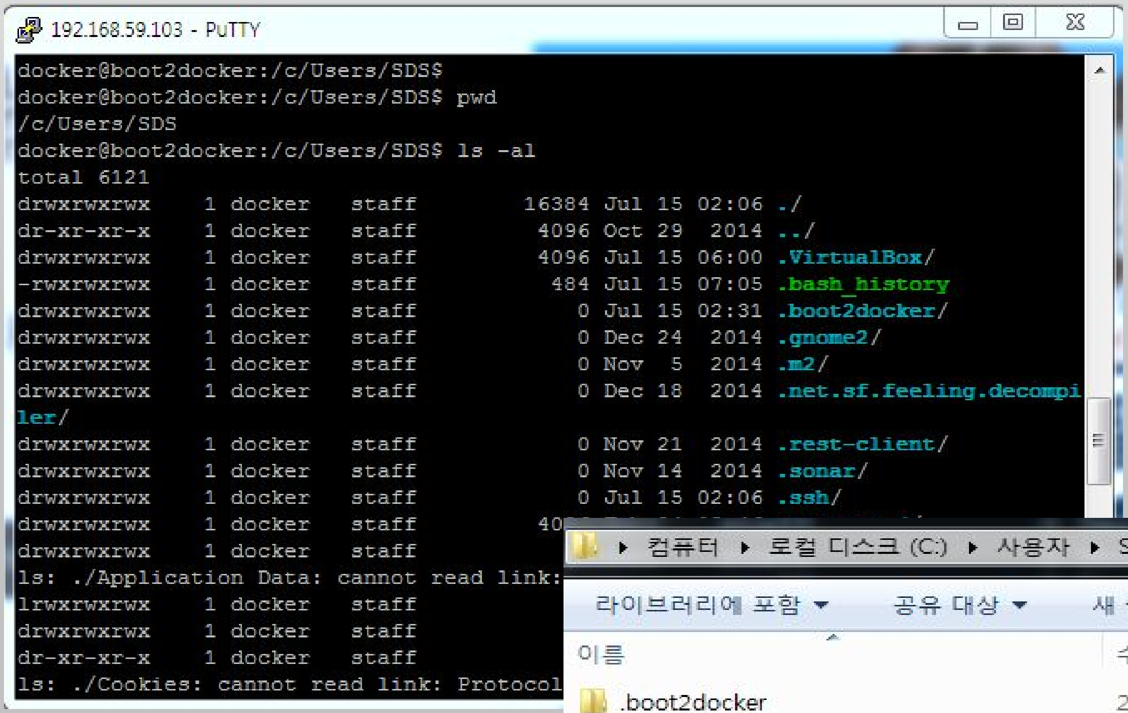


Host Name (or IP address): 192.168.59.103
Port: 22

```
login as: docker
docker@192.168.59.103's password:
Boot2Docker version 1.7.0, build master : 7960f90 - Thu Jun 18 18:31:45 UTC 2015
Docker version 1.7.0, build 0baf609
docker@boot2docker:~$ docker -v
Docker version 1.7.0, build 0baf609
docker@boot2docker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES
docker@boot2docker:~$
```

name : docker
passwd : tcuser

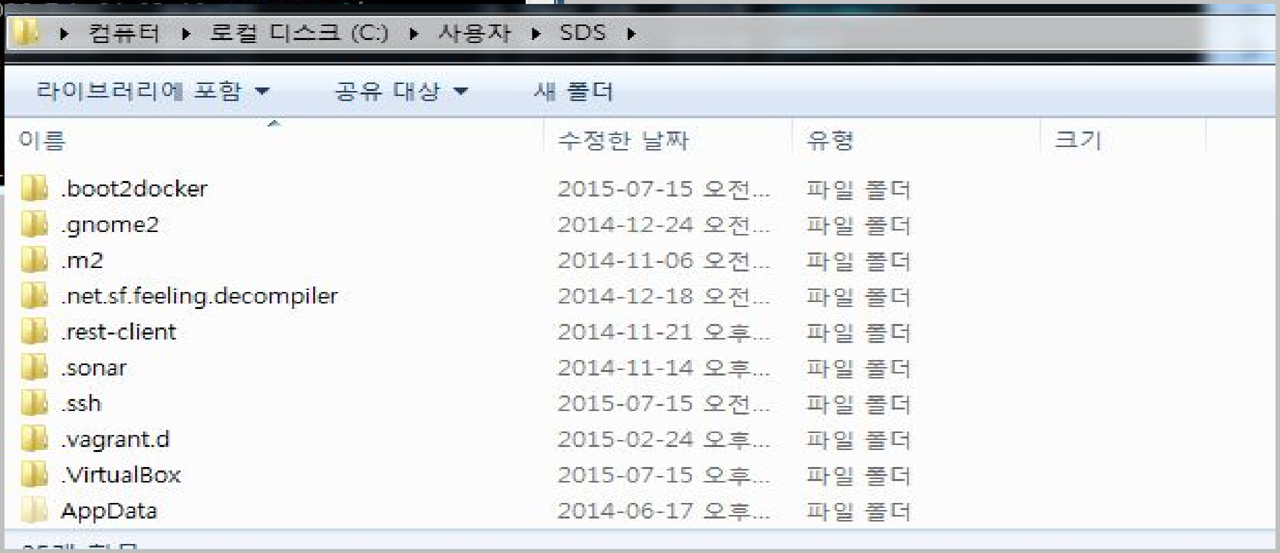
folder sharing



```
docker@boot2docker:/c/Users/SDS$
docker@boot2docker:/c/Users/SDS$ pwd
/c/Users/SDS
docker@boot2docker:/c/Users/SDS$ ls -al
total 6121
drwxrwxrwx    1 docker   staff        16384 Jul 15 02:06 ./
dr-xr-xr-x    1 docker   staff         4096 Oct 29  2014 ../
drwxrwxrwx    1 docker   staff         4096 Jul 15 06:00 .VirtualBox/
-rwxrwxrwx    1 docker   staff          484 Jul 15 07:05 .bash_history
drwxrwxrwx    1 docker   staff            0 Jul 15 02:31 .boot2docker/
drwxrwxrwx    1 docker   staff            0 Dec 24  2014 .gnome2/
drwxrwxrwx    1 docker   staff            0 Nov  5  2014 .m2/
drwxrwxrwx    1 docker   staff            0 Dec 18  2014 .net.sf.feeling.decompi
ler/
drwxrwxrwx    1 docker   staff            0 Nov 21  2014 .rest-client/
drwxrwxrwx    1 docker   staff            0 Nov 14  2014 .sonar/
drwxrwxrwx    1 docker   staff            0 Jul 15 02:06 .ssh/
drwxrwxrwx    1 docker   staff         40
drwxrwxrwx    1 docker   staff
ls: ./Application Data: cannot read link:
lrwxrwxrwx    1 docker   staff
drwxrwxrwx    1 docker   staff
dr-xr-xr-x    1 docker   staff
ls: ./Cookies: cannot read link: Protocol
```

Windows : C:\Users\{사용자}
Docker VM : /c/Users/{사용자}

Volume 을 통해 데이터 공유



이름	수정한 날짜	유형	크기
.boot2docker	2015-07-15 오전...	파일 폴더	
.gnome2	2014-12-24 오전...	파일 폴더	
.m2	2014-11-06 오전...	파일 폴더	
.net.sf.feeling.decompiler	2014-12-18 오전...	파일 폴더	
.rest-client	2014-11-21 오후...	파일 폴더	
.sonar	2014-11-14 오후...	파일 폴더	
.ssh	2015-07-15 오전...	파일 폴더	
.vagrant.d	2015-02-24 오후...	파일 폴더	
.VirtualBox	2015-07-15 오후...	파일 폴더	
AppData	2014-06-17 오후...	파일 폴더	

Proxy setting

- Proxy

```
$ sudo vi /var/lib/boot2docker/profile
  export HTTP_PROXY=http://host:port
  export HTTPS_PROXY=http://host:port
  DOCKER_TLS=no
```

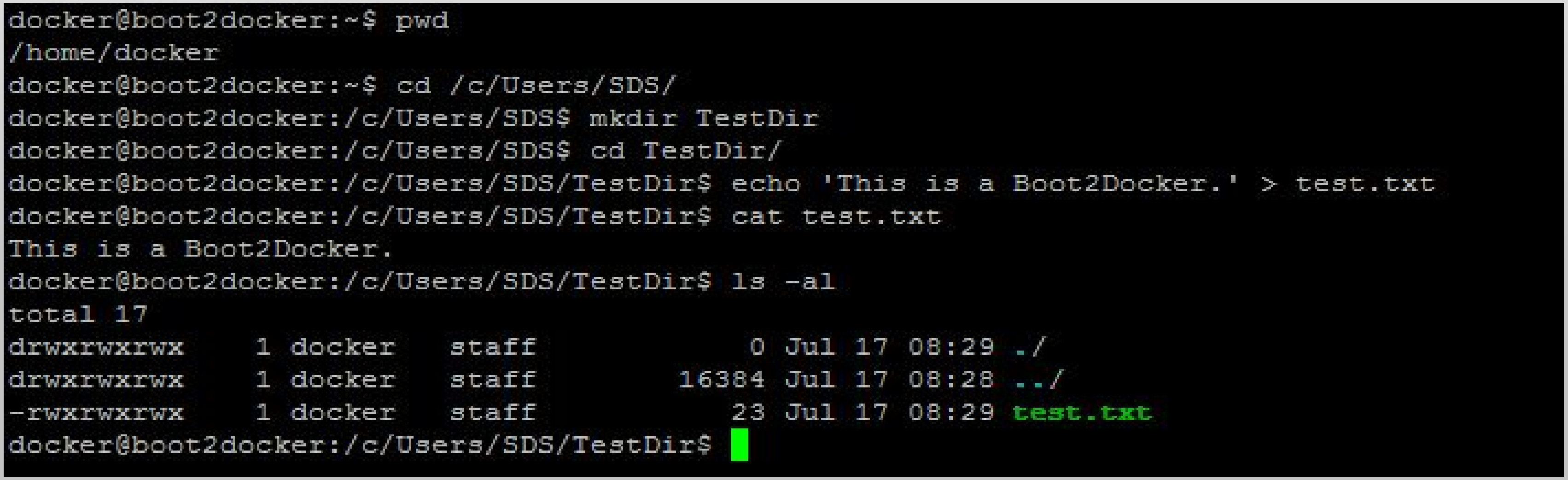
- 인증서

```
$ sudo cat mycert.crt >> /etc/ssl/certs/ca-certificates.crt
```

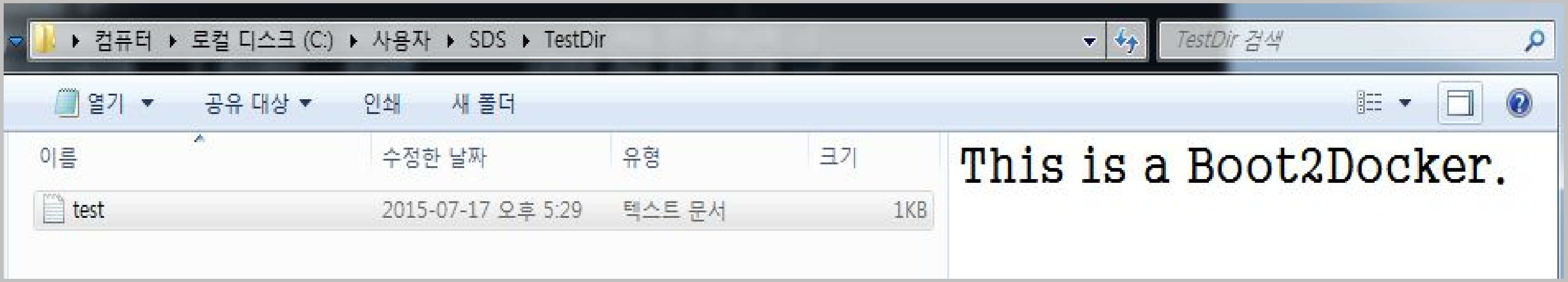
- Restart

```
$ sudo /etc/init.d/docker restart
```

folder sharing test



```
docker@boot2docker:~$ pwd
/home/docker
docker@boot2docker:~$ cd /c/Users/SDS/
docker@boot2docker:/c/Users/SDS$ mkdir TestDir
docker@boot2docker:/c/Users/SDS$ cd TestDir/
docker@boot2docker:/c/Users/SDS/TestDir$ echo 'This is a Boot2Docker.' > test.txt
docker@boot2docker:/c/Users/SDS/TestDir$ cat test.txt
This is a Boot2Docker.
docker@boot2docker:/c/Users/SDS/TestDir$ ls -al
total 17
drwxrwxrwx    1 docker   staff            0 Jul 17 08:29 ./
drwxrwxrwx    1 docker   staff        16384 Jul 17 08:28 ../
-rwxrwxrwx    1 docker   staff           23 Jul 17 08:29 test.txt
docker@boot2docker:/c/Users/SDS/TestDir$
```

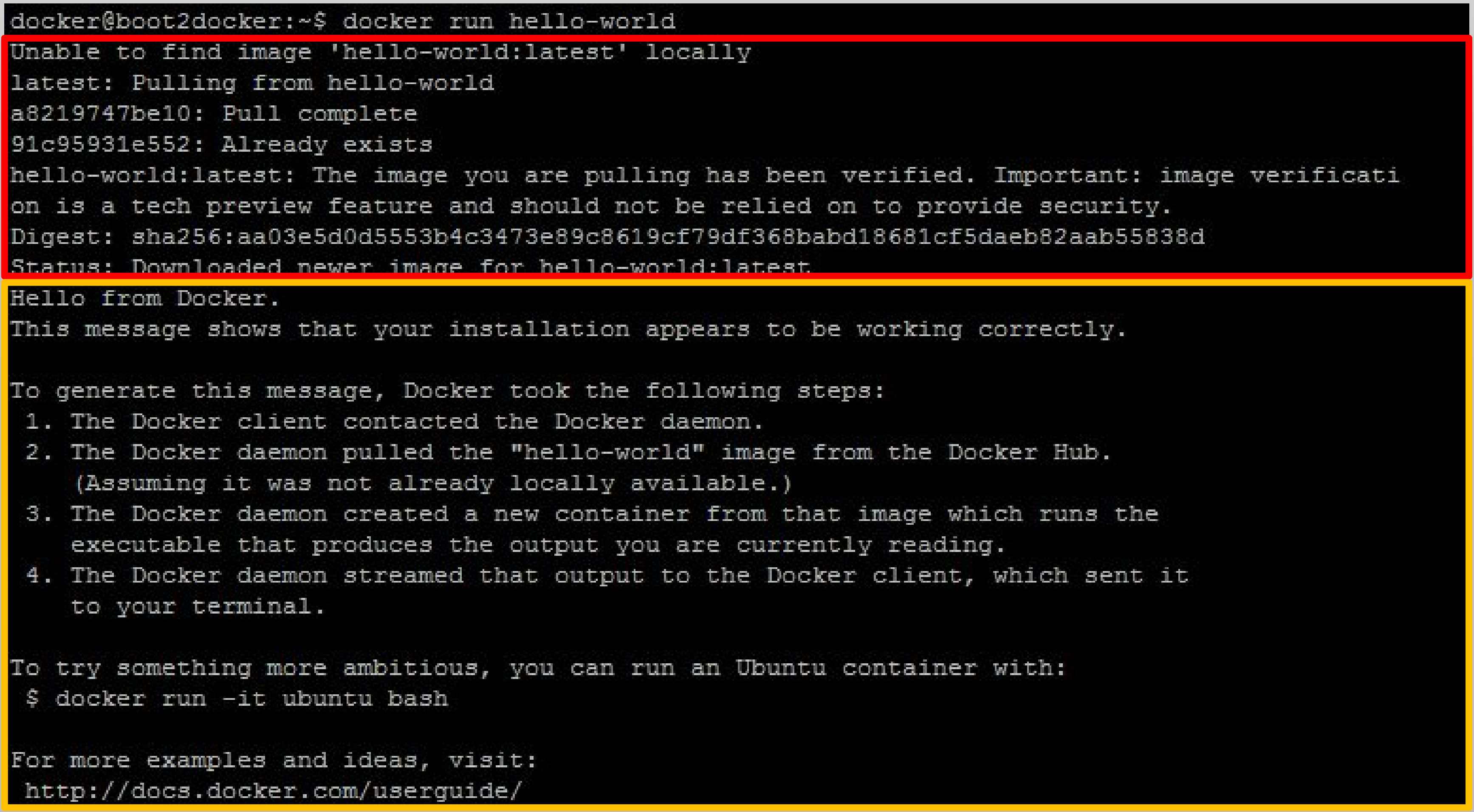


이름	수정한 날짜	유형	크기
test	2015-07-17 오후 5:29	텍스트 문서	1KB

This is a Boot2Docker.

hello-world

$ docker run hello-world



```
docker@boot2docker:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from hello-world
a8219747be10: Pull complete
91c95931e552: Already exists
hello-world:latest: The image you are pulling has been verified. Important: image verificati
on is a tech preview feature and should not be relied on to provide security.
Digest: sha256:aa03e5d0d5553b4c3473e89c8619cf79df368babd18681cf5daeb82aab55838d
Status: Downloaded newer image for hello-world:latest
Hello from Docker.
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (Assuming it was not already locally available.)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

For more examples and ideas, visit:
 http://docs.docker.com/userguide/
```

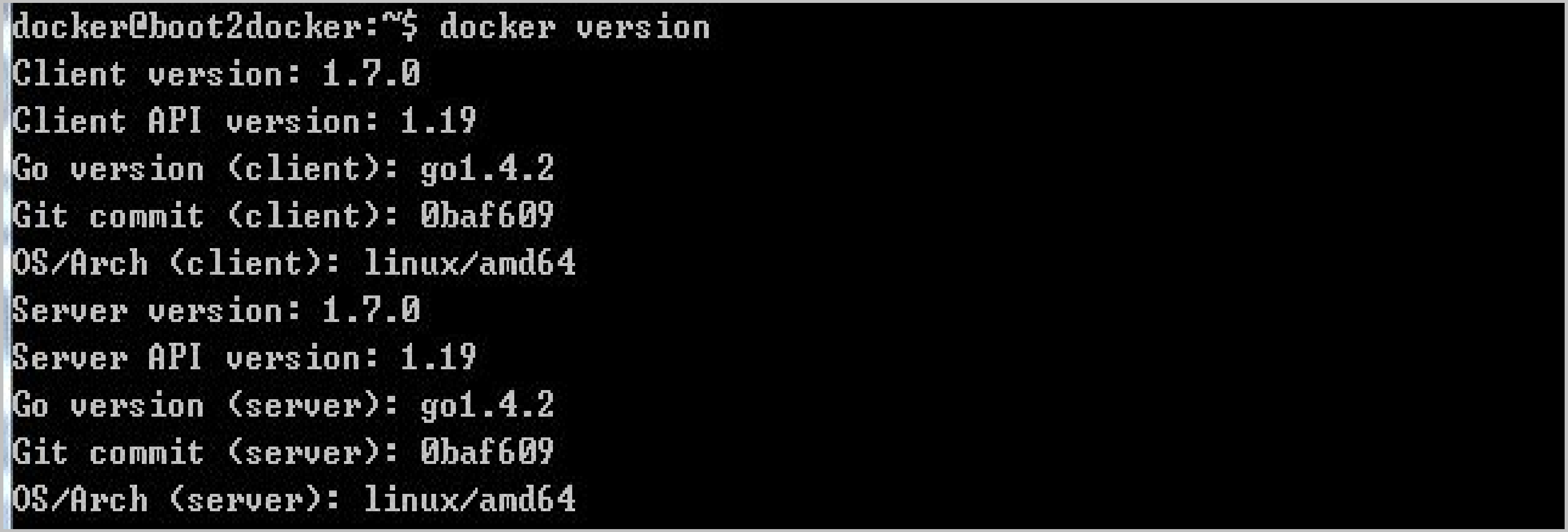
Docker Commands

https://docs.docker.com/reference/commandline/cli/

Docker Command

# Docker

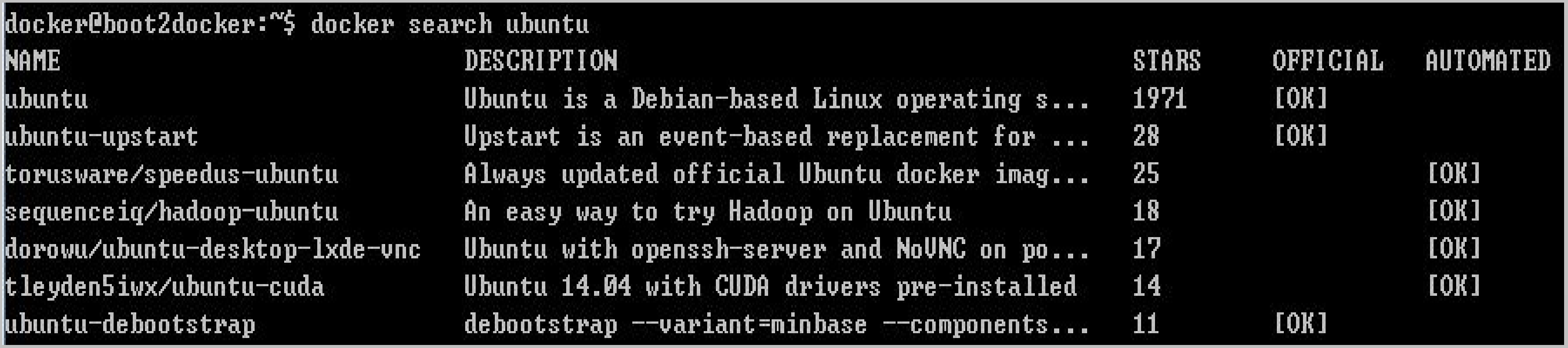
- version : docker 버전 확인
- info : docker 실행 시스템 정보
- events : 컨테이너들에 발생하는 이벤트 출력



```
docker@boot2docker:~$ docker version
Client version: 1.7.0
Client API version: 1.19
Go version (client): go1.4.2
Git commit (client): 0baf609
OS/Arch (client): linux/amd64
Server version: 1.7.0
Server API version: 1.19
Go version (server): go1.4.2
Git commit (server): 0baf609
OS/Arch (server): linux/amd64
```

Docker Command - Registry

- **search** : 이미지를 registry에서 조회
- pull : 이미지를 registry에서 받아옴
- push : 이미지를 registry에 올림
- login : registry에 로그인
- logout : registry에서 로그아웃



```
docker@boot2docker:~$ docker search ubuntu
NAME                          DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
ubuntu                        Ubuntu is a Debian-based Linux operating s...   1971      [OK]
ubuntu-upstart                Upstart is an event-based replacement for ...   28        [OK]
torusware/speedus-ubuntu      Always updated official Ubuntu docker imag...   25                   [OK]
sequenceiq/hadoop-ubuntu      An easy way to try Hadoop on Ubuntu             18                   [OK]
dorowu/ubuntu-desktop-lxde-vnc  Ubuntu with openssh-server and NoVNC on po...   17                   [OK]
tleyden5iwx/ubuntu-cuda       Ubuntu 14.04 with CUDA drivers pre-installed    14                   [OK]
ubuntu-debootstrap            debootstrap --variant=minbase --components...   11        [OK]
```

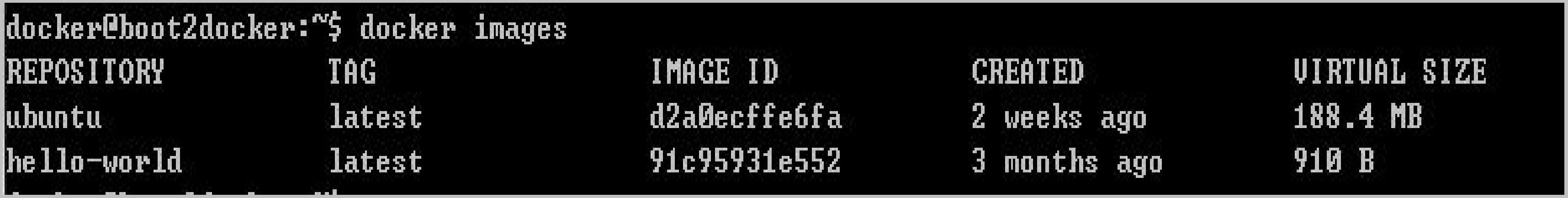
Docker Command - Image

# lifecycle

- **images** : 이미지 목록 조회
- **build** : dockerfile로 새 이미지 생성
- **commit** : 변경된 컨테이너를 커밋하여 새 이미지 생성
- import : 빈 이미지를 생성 후 tar파일을 import
- save : STDOUT을 통해 이미지(들)를 tar 아카이브로 저장
- load : STDIN을 통해 tar 아카이브에서 이미지(들)를 로드
  --> 부모 레이어, 태그, 버전 등 모두 복원
- **rmi** : 이미지 삭제
- tag : 저장소에 이미지 Tag 추가

# information

- history : 이미지의 변경이력 조회



```
docker@boot2docker:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
ubuntu              latest              d2a0ecffe6fa        2 weeks ago         188.4 MB
hello-world         latest              91c95931e552        3 months ago        910 B
```

Docker Command - Container

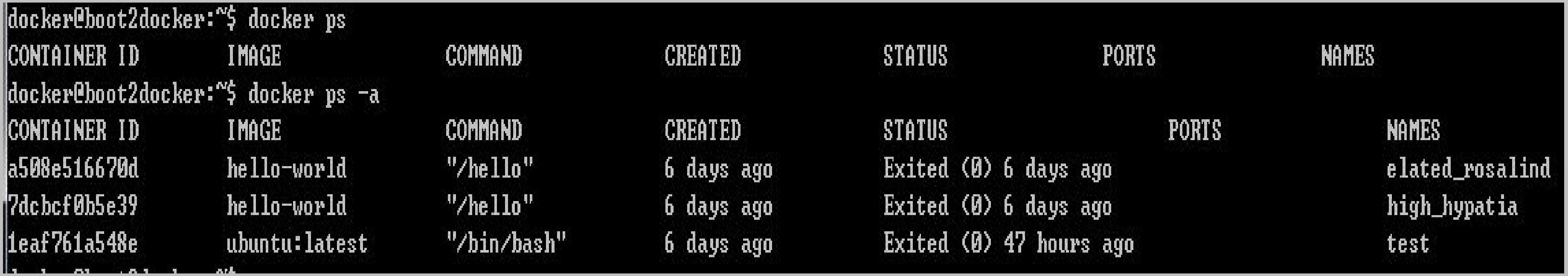
# lifecycle

- create : 이미지를 컨테이너로 생성
- **run** : 이미지를 컨테이너로 생성 후 실행
- **start** : 정지된 컨테이너를 실행
- **stop** : 실행중인 컨테이너를 종료
- restart : 실행중인 컨테이너를 재 시작
- kill : 실행중인 컨테이너를 강제 종료(SIGKILL)
- wait : 컨테이너가 종료될 때 까지 block.
- **rm** : 컨테이너 삭제
- export : 컨테이너 내 파일시스템을 tarball 형태로 출력
- **attach** : 실행중인 컨테이너에 접속
- **exec** : 실행중인 컨테이너 안의 command 실행
- pause : 컨테이너 내의 모든 프로세스 일시 중지(cgroups freezer)
- unpause : 컨테이너 내의 모든 프로세스 일시 중지 해제(cgroups freezer)
- renmae : 컨테이너 이름 변경
- cp : 컨테이너 안의 파일들을 호스트로 복사

Docker Command - Container

# information

- **ps** : 컨테이너 목록 조회
- **logs** : 실행중인 컨테이너의 로그를 출력
- stats : 실행중인 컨테이너의 리소스(cpu, memory..) 사용 통계 출력
- diff : 컨테이너의 파일시스템 변경정보 출력(A, D, C)
- **inspect** : 컨테이너 or 이미지의 상세 정보 출력
- top : 컨테이너 안에 실행중인 프로세스 조회
- port : 컨테이너의 포트 매핑 정보 조회



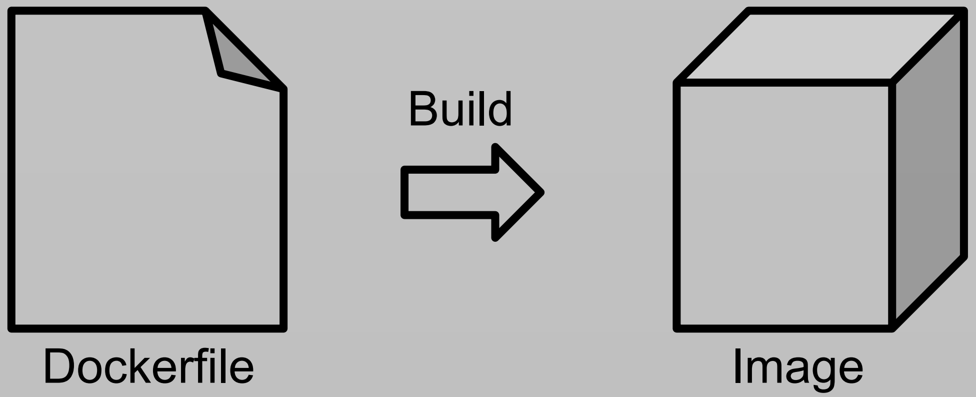
```
docker@boot2docker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
docker@boot2docker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                    PORTS               NAMES
a508e516670d        hello-world         "/hello"            6 days ago          Exited (0) 6 days ago                         elated_rosalind
7dcbcf0b5e39        hello-world         "/hello"            6 days ago          Exited (0) 6 days ago                         high_hypatia
1eaf761a548e        ubuntu:latest       "/bin/bash"         6 days ago          Exited (0) 47 hours ago                       test
```

dockerfile

https://docs.docker.com/reference/builder/

What is a Dockerfile?

- A Dockerfile is a **text document** that contains all the **commands** you would normally **execute manually** in order to build a Docker image.



dockerfile sample

```
$vi dockerfile
----------------------------------------------------------
FROM java:openjdk-8u45-jdk
MAINTAINER chris@chrisrichardson.net
ADD build/spring-boot-restful-service.jar .
CMD java -jar spring-boot-restful-service.jar
EXPOSE 8080
----------------------------------------------------------
```

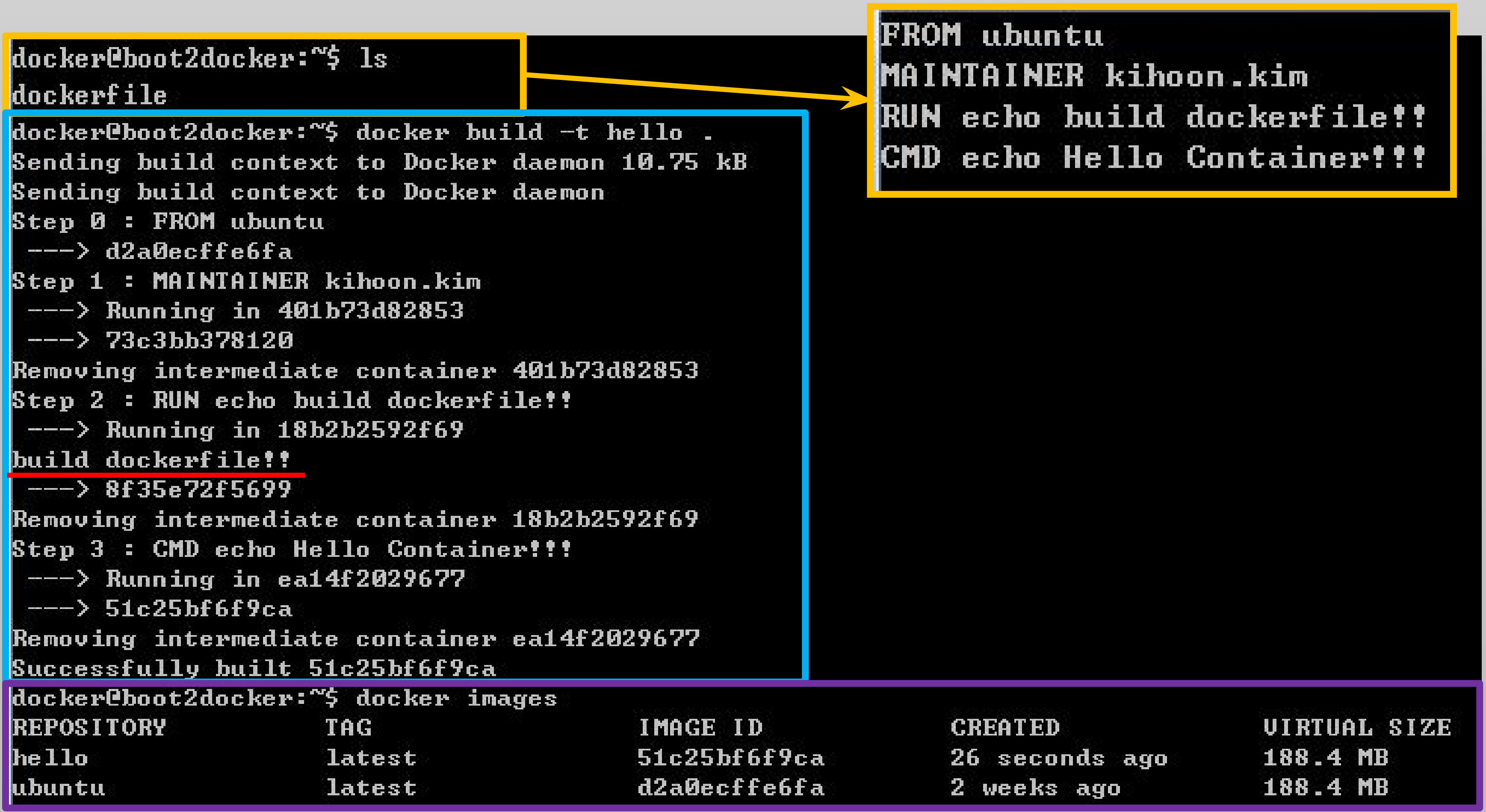
https://github.com/cer/microservices-examples/blob/master/spring-boot-restful-service/docker/Dockerfile

dockerfile Command

- **FROM** : Base Image
- **MAINTAINER** : 작성자
- **RUN** : 빌드 시 쉘 명령어 실행. Image에 commit 됨. (/bin/sh –c 명령어)
- **CMD** : 컨테이너 시작 시 실행할 명령어 or ENTRYPOINT의 arguments)
- **ENTRYPOINT** : 컨테이너 시작 시 실행할 명령어
- **LABEL** : 이미지의 메타데이터(version, description..) 기술. key-value
- **EXPOSE** : listen on the specified network ports at runtime
- **ENV** : 환경변수. 생성된 이미지 결과에도 반영 됨.
- **ADD** : 호스트의 파일, 디렉토리 및 원격 URL 을 컨테이너로 복사
- **COPY** : 호스트의 파일, 디렉토리를 컨테이너로 복사
- **VOLUME** : 호스트와 컨테이너들 간 호스트의 디렉토리 공유.
- **USER** : RUN, CMD, ENTRYPOINT 명령을 실행할 사용자 계정 설정
- **WORKDIR** : change directory
- **ONBUILD** : 이미지가 다른 이미지의 Base Image로 사용되어 빌드 될 때 실행할 명령어

Build Dockerfile

$ docker **build** **-t** {image_name} {dockerfile_path}



```
docker@boot2docker:~$ ls
dockerfile
docker@boot2docker:~$ docker build -t hello .
Sending build context to Docker daemon 10.75 kB
Sending build context to Docker daemon
Step 0 : FROM ubuntu
 ---> d2a0ecffe6fa
Step 1 : MAINTAINER kihoon.kim
 ---> Running in 401b73d82853
 ---> 73c3bb378120
Removing intermediate container 401b73d82853
Step 2 : RUN echo build dockerfile!!
 ---> Running in 18b2b2592f69
build dockerfile!!
 ---> 8f35e72f5699
Removing intermediate container 18b2b2592f69
Step 3 : CMD echo Hello Container!!!
 ---> Running in ea14f2029677
 ---> 51c25bf6f9ca
Removing intermediate container ea14f2029677
Successfully built 51c25bf6f9ca
docker@boot2docker:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
hello               latest              51c25bf6f9ca        26 seconds ago      188.4 MB
ubuntu              latest              d2a0ecffe6fa        2 weeks ago         188.4 MB
```

```
FROM ubuntu
MAINTAINER kihoon.kim
RUN echo build dockerfile!!
CMD echo Hello Container!!!
```

Dockerfile - General guidelines

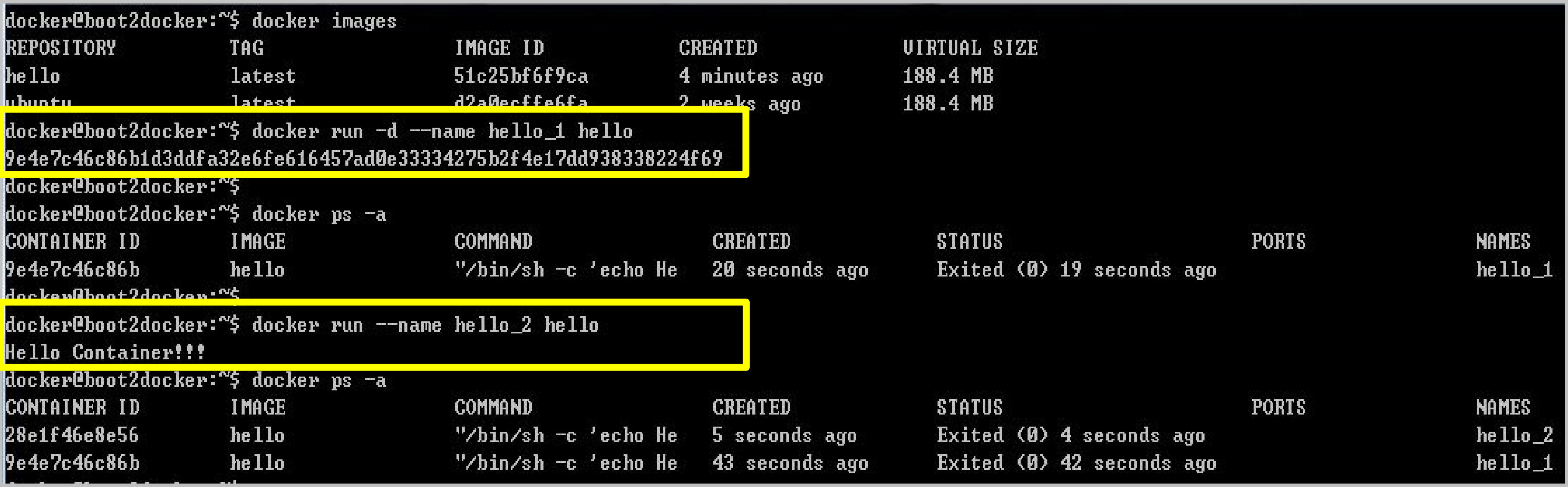
- Containers should be ephemeral
- Use a .dockerignore file
- Avoid installing unnecessary packages
- Run only one process per container
- Minimize the number of layers
- Sort multi-line arguments
- Build cache

https://docs.docker.com/articles/dockerfile_best-practices/

Docker run

$ docker **run** --name {container name}

- **Detached**(background) vs **foreground**
  - detached : '-d' option
  - foreground : default when -d is not specified
- **Container Name**
  - --name {이름}



```
docker@boot2docker:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
hello               latest              51c25bf6f9ca        4 minutes ago       188.4 MB
ubuntu              latest              d2a0ecffe6fa        2 weeks ago         188.4 MB
docker@boot2docker:~$ docker run -d --name hello_1 hello
9e4e7c46c86b1d3ddfa32e6fe616457ad0e33334275b2f4e17dd938338224f69
docker@boot2docker:~$
docker@boot2docker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                CREATED             STATUS                       PORTS               NAMES
9e4e7c46c86b        hello               "/bin/sh -c 'echo He   20 seconds ago      Exited (0) 19 seconds ago                        hello_1
docker@boot2docker:~$
docker@boot2docker:~$ docker run --name hello_2 hello
Hello Container!!!
docker@boot2docker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                CREATED             STATUS                       PORTS               NAMES
28e1f46e8e56        hello               "/bin/sh -c 'echo He   5 seconds ago       Exited (0) 4 seconds ago                         hello_2
9e4e7c46c86b        hello               "/bin/sh -c 'echo He   43 seconds ago      Exited (0) 42 seconds ago                        hello_1
```

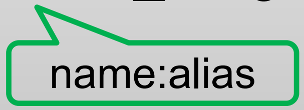
https://docs.docker.com/reference/run/

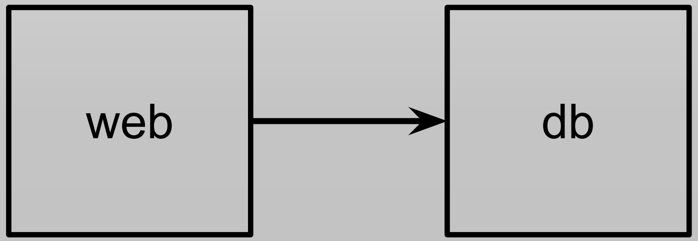
Container Link

https://docs.docker.com/userguide/dockerlinks/
https://docs.docker.com/articles/ambassador_pattern_linking

--link option

```
$ sudo docker run -d -P --name web  --link db:db  web_image
```





# 두가지 방법으로 Target Container 연결 정보 전달

- Environment variables : $ env
- /etc/hosts : $ cat /etc/hosts

Docker Compose

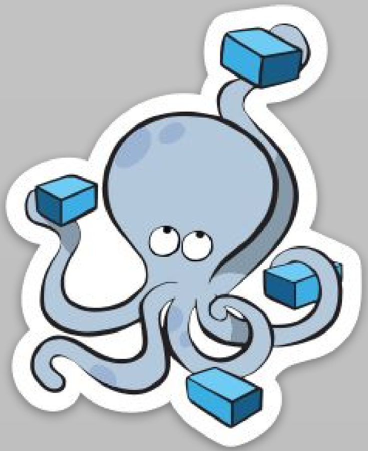
http://docs.docker.com/compose/

Docker Compose

- Compose is a **tool** for **defining** and **running** **multi-container applications** with **Docker**.

# a three-step process

1. Define your app’s environment with a **Dockerfile**
2. Define the services that make up your app in **docker-compose.yml**
3. run **docker-compose up**



Install Compose

```
$ curl -L
https://github.com/docker/compose/releases/download/1.3.2/docker-compose-`
uname -s`-`uname -m` > /usr/local/bin/docker-compose
$ chmod +x /usr/local/bin/docker-compose
```

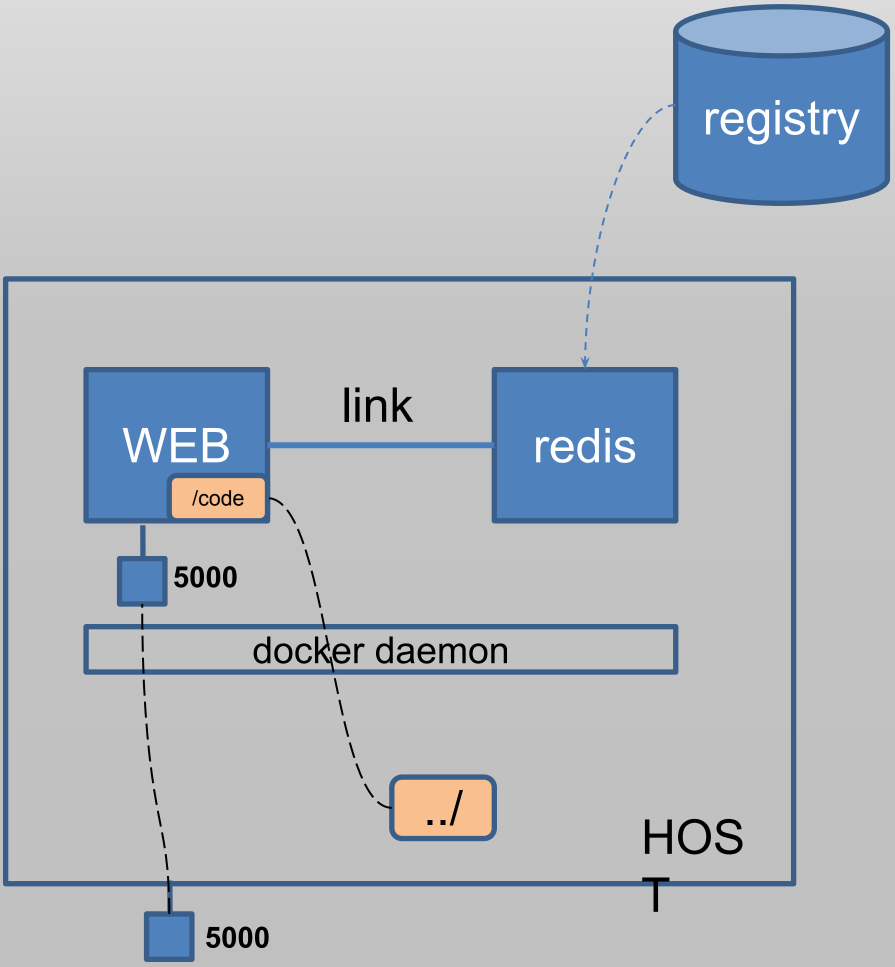
OR

```
$ apt-get install python-pip
$ pip install -U docker-compose
```

docker-compose.yml

```
web:
  build: .
  ports:
    - "5000:5000“
  volumes:
    - .:/code
  links:
    - redis
redis:
  image: redis
------------------------
$ docker-compose up
$ docker-compose stop
$ docker-compose start
```

```
$ docker run –name redis redis:latest
$ docker build –t web .
$ docker run –p 5000:50000 –link redis:redis \
              --name web web
$ docker stop redis
$ docker stop web
$ docker start redis
$ docker start web
………
```



references

- https://docs.docker.com/
- https://github.com/docker/docker

감사합니다.