



docker

2장

Docker Container 다루기

전체 내용

Docker Image를
다운로드하여
사용하기

Container를
실행하는 다양한
방법

Docker Image
및 Container
관리하기

쉬어 가는 코너

1-Docker Image를 다운로드하여 사용하기

Container 실행하기-docker run

container 중지하기-docker stop

Container에 접속하는 3가지 방법

docker images와 docker ps 명령어의 차이

1-Docker Image를 다운로드하여 사용하기

- Container 실행하기-**docker run**

- Host OS는 CentOS인데, docker는 최신의 Ubuntu 버전 이미지 다운로드하여 사용하기
 - whoami (##adminuser임을 알 수 있다)
 - **docker run -it ubuntu /bin/bash**
 - (##docker run은 image를 로컬 호스트나 docker hub에서 찾아서 Container를 실행하는 것이다.)
 - Docker 버전 확인하기
 - hostname -i
 - **cat /etc/*-release**
 - **apt update**
 - **apt install iputils-ping -y**
 - ping www.google.com -c 3
 - Docker에 접속한 상태에서 Host OS 버전 확인하기
 - **uname -a** (## 또는 **cat /proc/version**)
 - ## Host OS는 CentOS임을 알 수 있다
 - ## **uname -a**는 OS 커널 정보를 확인하는 것이므로 Host OS 정보를 가져온다

1-Docker Image를 다운로드하여 사용하기

- Container 실행하기-**docker run**

- 접속한 docker에서 빠져 나온 후 Container를 중지하기

- **exit**

- 다운로드한 docker 이미지 확인하기

- **docker --help**

- **docker images**

```
[root@centos1 docker]# docker images
REPOSITORY          TAG                 IMAGE ID
ubuntu              latest             ccc7a11d65b1
[root@centos1 docker]#
```

- Ubuntu 최신 이미지가 다운로드 되어 있음을 알 수 있다

- 실행 중인 Docker image인 Container가 있는지 확인하기

- **docker ps**

- 현재 실행중인 Container는 없다

- docker ps 다음에 사용할 수 있는 옵션을 확인하기 위해서는 **docker ps --help**

- 로컬 컴퓨터에 실행중인 것과 저장된 Container 모두 확인하기

- **docker ps -a**

1-Docker Image를 다운로드하여 사용하기

- Container 실행하기-**docker run**

- Host OS는 CentOS인데, docker는 최신의 CentOS 버전 이미지 다운로드하여 사용하기
 - whoami (##adminuser임을 알 수 있다)
 - **docker run -it centos /bin/bash**
 - 이 명령은 docker image를 로컬 repository 또는 Docker Hub에서 찾아서 Container를 생성한 후 실행중인 container에 접속하는 것이다
 - docker를 실행한다는 것은 docker container를 실행하는 것을 말한다
 - **-i**: interactive
 - **-t**: pseudo-tty
 - **/bin/bash**: 자동으로 실행할 파일
 - Docker container 버전 확인하기
 - hostname -i
 - **cat /etc/*-release**
 - ping www.google.com -c 3

1-Docker Image를 다운로드하여 사용하기

- Container 실행하기-**docker run**

- 접속한 docker container가 계속 실행 중인 상태에서 잠시 빠져 나오기
 - **Ctrl+p, Ctrl+q**
 - ## 접속한 container에서 잠시 빠져 나오는 것은 ctrl+p,ctrl+q이고, container에서 빠져 나온 후에 중지하는 것은 exit 또는 ctrl+d
- 실행중인 **container 목록** 확인하기
 - **docker ps**
- 실행중인 container와 중지된 container를 모두 확인하기
 - **docker ps -a**
- 실행중인 container에 **접속하기: docker exec**
 - docker ps
 - **docker exec -it** "container id" (또는 container name)

1-Docker Image를 다운로드하여 사용하기

- Container 중지하기-**docker stop**

- 다시 빠져 나와서 실행중인 **container** 중지하기: **docker stop**

- Ctrl+p, ctrl+q

- **docker info**

```
[root@centos1 ~]# docker info
Containers: 3
Running: 1
Paused: 0
Stopped: 2
Images: 2
```

- **docker images**

```
[root@centos1 ~]# docker images
REPOSITORY          TAG                 IMAGE ID
ubuntu              latest             ccc7a11d65b1
centos               latest             328edcd84f1b
```

- **docker ps**

- **docker stop** "container id"

- **docker info**

- 모든 컨테이너가 중지됨

```
[root@centos1 ~]# docker ps
CONTAINER ID        IMAGE               COM
PORTS              NAMES
459375a75179       centos             "/b
nutes              musing_newt
[root@centos1 ~]# docker stop 459375a75179
459375a75179
[root@centos1 ~]# docker ps
CONTAINER ID        IMAGE               COM
PORTS              NAMES
[root@centos1 ~]#
```


1-Docker Image를 다운로드하여 사용하기

- Container에 접속하는 2가지 방법
 - 로컬 및 인터넷에서 docker image를 찾아서 container를 시작하면서 접속하기
 - **docker run --name hello -it centos /bin/bash**
 - **ctrl+p, ctrl+q**
 - 실행중인 container 접속하기
 - **docker exec -it hello /bin/bash**
 - **exit**
 - **docker ps -a**
 - 참고: **-it**를 사용하여 실행된 **Container만** docker attach로도 shell에 접속할 수 있다
 - **docker attach** hello (## "container name" 또는 "container id" 사용)
 - **ctrl+p, ctrl+q**

2-Container를 실행하는 다양한 방법

다양하게 이미지 실행하기

Docker Container에 접속하는 방법

MySQL에 접속하기

Docker Image 관리하기

2-Container를 실행하는 다양한 방법

- 다양하게 이미지 실행하기

- **docker run** --name mycon1 **centos** **cat /etc/hostname**

- mycon1는 container 이름이다
- **docker ps** (## 실행중인 container가 없음)
- **docker ps -a** (## 중지된 container는 있음)

- **docker run** --name mycon2 **centos** **ping www.google.com -c 3**

- **docker run** --name mycon3 **centos** **mkdir /imsidata**

- **docker run** --name mycon4 **-it centos** **mkdir /imsidata**

- **docker run** --name mycon5 **-it centos** **/bin/bash**

- **mkdir /imsidata**
- **exit**
- **docker start** mycon5 (##중지된 container를 실행만 하기)
- **docker attach** mycon5 **ls -l /imsidata** (##실행중에 container에 접속하기)
- **exit**

2-Container를 실행하는 다양한 방법

- 다양하게 이미지 실행하기
 - **docker run --name mycon6 -it -d centos /bin/bash**
 - **-d: --detach** (##백그라운드에서 container 실행함)
 - **docker ps** (##mycon6가 실행중임)
 - **docker attach mycon6**
 - **ls -l /imsidata** (## 이 디렉토리는 존재하지 않음. 다른 컨테이너임)
 - **exit**
 - **docker run --name myweb1 -d -p 80:80 httpd**
 - **-p: --publish** (##외부에서 80으로 접속하면 container의 80으로 보냄)
 - **docker ps** (##myweb1가 실행중임)
 - **curl localhost** (##Web service가 되고 있음)
 - mysql이나 httpd, nginx와 같은 서비스 이미지를 실행할 때는 반드시 -p 옵션을 사용해야 한다
 - docker root directory 용량 확인하기
 - **du -sh /var/lib/docker**

2-Container를 실행하는 다양한 방법

- Docker Container에 접속하는 방법
 - docker hub에서 docker image 다운로드만 하기
 - **docker images**
 - **docker pull centos**
 - **docker pull alpine**
 - **docker pull nginx**
 - **docker images**
 - 다운로드된 docker image를 실행하면서 Container에 접속하기
 - **docker run -i -t centos /bin/bash** (##container 이름 지정하지 않고 실행)
 - **ping www.google.com -c 3**
 - **exit**
 - **docker run --name mycentos -i -t centos /bin/bash**
 - **ping powershell.kr -c 3**
 - **exit**

2-Container를 실행하는 다양한 방법

- Docker Container에 접속하는 방법
 - 중지된 **Container**를 구동하여(start) 접속하기
 - **docker ps**
 - ##현재 실행중인 container 확인하기
 - **docker ps -a**
 - ##중지되었거나 실행중인 모든 **container** 확인하기
 - **docker ps -f status=exited**
 - ##실행이 중지된 container만 확인하기
 - **docker start** "container id" (또는 docker start "container name")
 - ## 중지된 container 실행하기

2-Container를 실행하는 다양한 방법

- Docker Container에 접속하는 방법
 - Container를 **background**에서 실행만 하고 접속을 하지 않기
 - **docker run --name my_daemonized -d ubuntu /bin/sh -c "while true; do echo my daemonized container; sleep 1; done "**
 - ## -d: detach
 - **docker logs** my_daemonized
 - **docker logs -f** my_daemonized
 - ctrl+c
 - **docker top** my_daemonized

2-Container를 실행하는 다양한 방법

- Docker Container에 접속하는 방법
 - **background에서 실행중인 Container를 외부에서 명령어 실행하기**
 - **docker exec -d my_daemonized mkdir /tmp/test**
 - ##docker exec 다음에 container 이름을 사용
 - **docker exec -i -t my_daemonized /bin/bash**
 - **ls -l /tmp**
 - **exit**
 - **docker ps**
 - **docker rm \$(docker ps -aq) -f**
 - ##실행중이거나 중지된 모든 container 삭제하기

2-Container를 실행하는 다양한 방법

- Docker Container에 접속하는 방법
 - docker host가 갑자기 restart 할 때 실행중이던 container들을 자동으로 실행하기: **--restart=always**
 - **docker run --name my_daemonized -d --restart=always ubuntu /bin/sh -c "while true; do echo my daemonized container; sleep 1; done"**
 - **reboot**
 - 다시 docker host에 접속하여 container가 자동으로 실행되었는지 확인
 - **docker ps**
 - **docker rm \$(docker ps -aq) -f**
 - 외부에서 접속하기: Port를 Publish하기
 - **docker run -d -p 8080:8080 glassfish**
 - **docker run --name mynginx -d -p 80:80 nginx**
 - 외부에서 Web Browser로 접속하기

3-Docker Image 및 Container 관리하기

Docker 구성 요소

Docker Image란?

Docker Container란?

Docker Repository란?

Docker Image 관리하기

3-Docker Image 및 Container 관리하기

- Docker 구성 요소

- Docker Engine(=Docker Daemon, Docker Runtime)

- Shipping Yard(적하 야적장): 물건을 배송하려면 옮기는 장소 및 환경 필요
 - **systemctl status docker.service**

- Docker Images

- Shipping Manifest(적하 목록): 배송할 물건들(피아노, 쿠키, 자동차,쌀)
 - **docker images**

- Docker Containers

- Shipping Containers(적하 컨테이너): 물건들을 컨테이너에 넣어서 배송
 - **docker ps -a**

Docker Engine
(Shipping Yard)



Docker Images
(Manifests)

SHIPPING MANIFEST									
Date Entered : 01/11/2010		Blk# : 008		Manifest Number : 2					
User ID : [REDACTED]		Route : T3		Driver Name : [REDACTED]					
Truck ID : [REDACTED]		Schedule Date of Departure : 01/11/2010		Number of Stops : 4					
Actual Date of Departure : 01/12/2010		Schedule Time of Departure : 0800		Actual Time of Departure : 1400					
Date of Return : 01/11/2010		Time of Return : 1400		Image No : 10000					
Image Code : 10000		Driver Comments : Actual of company pickup							
Delivery Totals									
Customer Name	Trailer #	Boxes	Code	Pallets	Units	Cartons	Trucks	Bags	Other
TEST ACCOUNT	0001	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0002	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0003	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0004	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0005	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0006	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0007	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0008	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0009	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0010	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0011	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0012	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0013	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0014	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0015	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0016	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0017	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0018	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0019	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0020	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0021	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0022	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0023	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0024	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0025	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0026	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0027	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0028	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0029	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0030	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0031	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0032	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0033	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0034	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0035	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0036	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0037	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0038	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0039	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0040	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0041	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0042	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0043	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0044	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0045	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0046	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0047	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0048	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0049	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0050	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0051	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0052	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0053	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0054	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0055	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0056	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0057	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0058	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0059	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0060	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0061	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0062	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0063	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0064	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0065	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0066	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0067	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0068	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0069	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0070	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0071	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0072	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0073	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0074	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0075	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0076	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0077	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0078	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0079	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0080	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0081	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0082	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0083	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0084	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0085	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0086	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0087	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0088	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0089	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0090	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0091	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0092	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0093	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0094	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0095	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0096	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0097	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0098	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0099	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0100	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0101	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0102	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0103	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0104	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0105	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0106	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0107	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0108	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0109	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0110	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0111	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0112	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0113	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0114	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0115	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0116	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0117	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0118	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0119	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0120	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0121	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0122	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0123	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0124	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0125	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0126	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0127	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0128	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0129	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0130	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0131	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0132	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0133	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0134	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0135	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0136	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0137	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0138	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0139	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0140	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0141	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0142	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0143	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0144	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0145	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0146	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0147	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0148	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0149	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0150	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0151	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0152	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0153	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0154	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0155	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0156	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0157	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0158	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0159	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0160	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0161	1	1	1	1	1	1	1	1
INTEGRATED MATERIALS, L.P.	0162	1							

3-Docker Image 및 Container 관리하기

- Docker 구성 요소

- Docker Registry

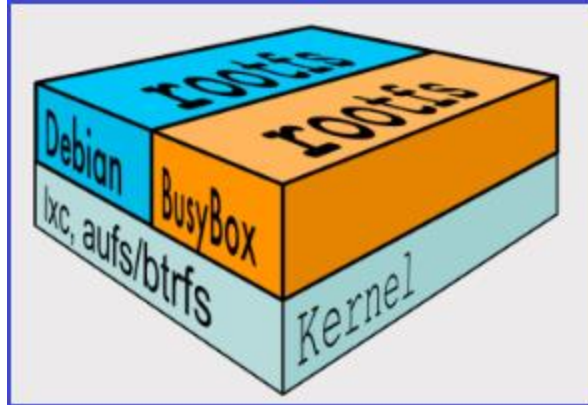
- Docker Repository를 운영하는 Server
 - 예: ktds12.azurecr.io (##azure Cloud)
 - Docker Repository를 생성하려면 먼저 Docker Registry를 구축해야 하는데, 특별하게 지정하지 않고 docker image를 검색할 때는 **http://hub.docker.com**에서 진행한다

- Docker Repository

- Docker Image들을 저장해 둔 장소를 말한다
 - ktds12.azurecr.io의 nginx가 docker repository이다
 - hub.docker.com의 **ezra1044/centosyslee**가 docker repository이다
 - docker image는 docker registry의 repository에 저장되어 있다
 - 이러한 이미지를 로컬 컴퓨터로 다운로드하여 실행하여 container로서 사용한다
 - **docker search apache**

3-Docker Image 및 Container 관리하기

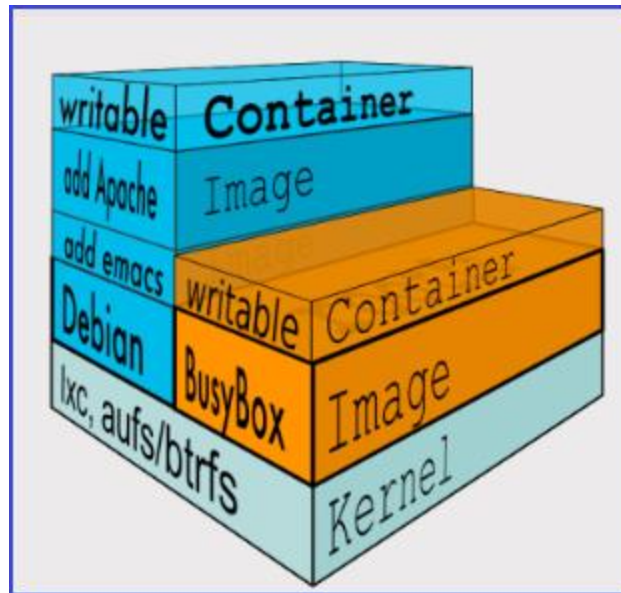
- Docker Image란?
 - A template of container
 - 사전에 Application과 그것의 실행환경이 설치되어 있다
 - Dockerfile이나 container를 사용하여 Image를 생성할 수 있다
 - Image가 생성된 후에는 Read-Only가 된다



3-Docker Image 및 Container 관리하기

- Docker Container란?

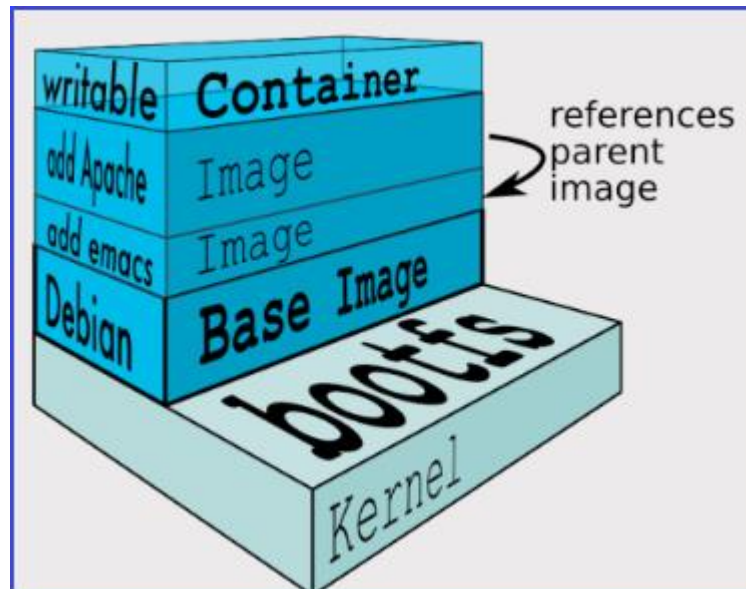
- Application을 실행하고 서비스하는 모든 환경을 갖춘 소프트웨어
- Image에서 Container를 생성한다
- 실행중인 프로세스를 말한다
- 컨테이너를 실행한 후에 수정도 가능하고 Access도 가능하다
- Container에서 수정된 것을 적용하여 새로운 Image를 생성할 수 있다



3-Docker Image 및 Container 관리하기

- Image vs Container

- Object Oriented Programming 개념에 견주어 보면, Image는 Class, Container는 Object에 해당된다
- Image는 수정할 수 **없고**, Container를 수정할 수 **있다**
- Image는 확장할 수 있다
- Image가 실행 중일 때 Container로 변형된다



3-Docker Image 및 Container 관리하기

- Docker Repository란?

- Docker Image를 저장해 둔 장소
- <https://hub.docker.com>에 저장함
- Docker hub가 제공하는 저장소에는 공개/비공개 2가지가 있는데 공개 저장소는 사용 개수에 제한이 없이 무제한으로 사용할 수 있다.
- 비공개 저장소는 기본적으로 1개는 무료로 제공해 주지만, 그 이상의 비공개 저장소를 사용하기 위해서는 비용을 지불해야한다.
 - 결제 방식은 월 결제 방식이며 <https://hub.docker.com/account/billing-plans/>에서 가격 정보를 확인할 수 있다.

The screenshot shows the Docker Hub web interface. At the top, there's a navigation bar with 'Dashboard', 'Explore', 'Organizations', 'Create', and a user profile for 'jesuswithme'. Below this, a search bar and filters for 'Repositories', 'Stars', and 'Contributed' are visible. The main section is titled 'Repositories' and includes a 'Create Repository +' button. A search filter 'Type to filter repositories by name' is present. The first repository listed is 'jesuswithme/ubuntu16' with '0 STARS' and '44 PULLS'. A 'DETAILS' button is next to it. On the right, a 'Docker Security Scanning' panel offers to 'Protect your repositories from vulnerabilities' with a 'Try it free' link.

Repository	Stars	Pulls	Action
jesuswithme/ubuntu16 public	0	44	DETAILS

3-Docker Image 및 Container 관리하기

- Docker Image 관리하기

- 사용하고자 하는 Docker Image 검색하기: **docker search**

- Docker Image는 Docker Hub(<https://hub.docker.com/>)에 저장되어 있다
 - Docker search 명령으로 원하는 image을 찾을 수 있다

- **docker help**

- **docker search --help**

- **docker search** fedora

- **docker search** nginx

- **docker search** apache

- docker search --**filter**=**stars=3** apache (##별표 3개 이상이 것만 검색)

- docker search --**filter**=**stars=3 --no-trunc** apache (##설명란 모두 보기)

- docker search --**filter**=**is-official=true** apache

- docker search --**filter**=**is-official=true --no-trunc** apache

- **docker search ubuntu:14.04** (##특정한 버전 지정하여 검색하기)

3-Docker Image 및 Container 관리하기

- Docker Image 관리하기

- 사용하고자 하는 Docker Image 검색하기: **docker search**

- **docker search** apache

- **NAME** : 이미지 이름. 이름에서 /로 구분되어 있는 것들은 개별 사용자들의 공개저장소에 있는 이미지임을 알려준다
 - **DESCRIPTION**: 이미지에 대한 상세 설명
 - **STARS**: 이미지가 사용자들로부터 몇 개의 별을 받았는지 보여준다
 - **OFFICIAL**: 이 이미지가 공식 이미지 여부 확인
 - **AUTOMATED**: 이 이미지가 github나 bitbucket같은 외부 소스 저장소에 있는 소스를 dockerhub에서 자동으로 빌드하고 업데이트한 것인지 여부

```
[root@centos1 ~]# docker search apache
```

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
tomcat	Apache Tomcat is an open source implementa...	1448	[OK]	
httpd	The Apache HTTP Server Project	1197	[OK]	
cassandra	Apache Cassandra is an open-source distrib...	628	[OK]	
maven	Apache Maven is a software project managem...	448	[OK]	
solr	Solr is the popular, blazing-fast, open so...	420	[OK]	
zookeeper	Apache ZooKeeper is an open-source server ...	201	[OK]	
eboraas/apache-php	PHP5 on Apache (with SSL support), built o...	132		[OK]
eboraas/apache	Apache (with SSL support), built on Debian	81		[OK]

3-Docker Image 및 Container 관리하기

- Docker Image 관리하기

- 찾은 이미지를 로컬로 다운로드만 하기: **docker pull**

- 필요한 docker image(build-time)를 로컬 컴퓨터에 저장해 놓고 필요할 때마다 실행하여(container(=runtime)) 사용한다
 - docker search fedora
 - **docker pull --help**
 - **docker pull** fedora
 - **docker images**
 - **docker images** **fedora**

3-Docker Image 및 Container 관리하기

- Docker Image 관리하기

- 실행이 중지된 container 실행하기: **docker start**

- Docker 이미지를 실행하면 Container가 되며, 내용을 편집하여 다시 저장할 수 있다
 - 다시 저장된 것은 **Docker Image가 아니고 Container**라고 한다
 - Docker를 실행하여 사용하는 방법은 1) 중지된 **container**를 시작하거나 2) **docker 원본 Image를 실행**할 수 있다

- **docker ps -a**

- 중지된 container를 다시 실행하기 위해 실행되고 있거나 실행이 중지된 Container 목록을 확인한다

- **docker start "container id"**

- Status를 확인한 후 중지된 Container를 실행한다

- **docker ps**

- **docker attach "container id"**

- **mkdir /lab**

- **exit**

- **docker ps**

3-Docker Image 및 Container 관리하기

- Docker Image 관리하기
 - 실행 중인 container 중지하기: **docker stop**
 - **docker ps -a**
 - **docker start myfedora**
 - **docker ps**
 - 실행 중인 container 확인하기
 - **docker stop** "container name"
 - **docker ps**
 - 서비스 중인 container가 중지되었기 때문에 여기에 보이지 않는다

4-쉬어 가는 코너

가장 많이 사용하는 docker 명령어

가장 많이 사용하는 docker 명령어

- 가장 많이 사용하는 docker 명령어
 - 각 명령어에 대한 도움말(Help) 활용하기
 - `docker image --help`
 - OS Container 실행하기
 - `docker run -itd --name myubuntu ubuntu`
 - Application 실행하기
 - `docker run -d --name myweb -p 8080:80 nginx`
 - Image를 생성한 절차 확인하기
 - `docker image history centos`
 - Docker Volume 및 Network을 지정하여 사용하기
 - `docker volume create convol`
 - `docker network create connet`
 - `docker run -itd --name examplecon -v convol:/data --net connet centos`

가장 많이 사용하는 docker 명령어

- Docker Volume 및 Network을 지정하여 사용하기(계속)
 - **docker volume ls**
 - **docker volume create** **apacheroot**
 - **docker run -d --name myapache -p 8888:80 -v**
apacheroot:/usr/local/apache2/htdocs/ --net connect httpd
 - **docker volume inspect** apacheroot
 - **cd /var/lib/docker/volumes/apacheroot/_data**
 - **ls -l**
 - **echo "Hi, Everybody" > /var/lib/docker/volumes/apacheroot/_data/index.html**
 - **curl http://localhost:8888**
 - **docker network inspect connect | grep IPv4Address**

```
[root@vm1 _data]# docker network inspect connet | grep IPv4Address
    "IPv4Address": "172.18.0.3/16",
    "IPv4Address": "172.18.0.2/16",
```
 - **docker exec -it example /bin/bash**
 - ping 172.18.0.3 (##성공.. 같은 network를 사용하기 때문에 통신 가능)