[HOL] Docker Container

1. Docker Hub에서 Container Image 검색하기
   1)Docker Version 확인
      $ sudo docker version

   2)Docker Service 확인하기
      $ su -
      # systemctl status docker

   3)Docker Hub에서 nginx 검색하기
      # docker search nginx

2. Container Image 다운로드 후 Image Layer 보기
   1)Docker Layer 확인하기
      # cd /var/lib/docker
      # ls -l
      # cd overlay2
      # ls -l
      # cd /home/{{계정}}
      # docker images

   2)Docker Hub에서 Nginx Pull
      # docker pull nginx:latest
      # docker images
      # ls -l /var/lib/docker/overlay2/   <---6개의 directory 확인

3. Container 실행하고 확인하기
   1)Docker Image 확인
      # docker image ls

   2)Docker Image 실행하기
      # docker run -d --name webserver -p 80:80 nginx:latest
      # curl localhost:80

   3)docker Container Stop
      # docker ps
      # docker stop werserver
      # docker ps -a

   4)docker Container remove
      # docker rm webserver
      # docker ps -a

   5)docker Image remove
      # docker image ls
      # docker rmi nginx
      # docker images
      # ls -l /var/lib/docker/overlay2/

4. Port Binding 하기
   1)Server 단에서 Nginx 실행하기
      # docker run -p 80:80 nginx
      log 대기

   2)Client 단에서
      $ curl localhost:80

      -Server 단의 logging
      172.17.0.1 - - [21/Jun/2021:06:02:26 +0000] "GET / HTTP/1.1" 200 612 "-" "curl/7.68.0" "-"

   3)Client 단에서 404 Not Found 페이지 호출
      $ curl localhost:80/aaa.html

-Server 단에서 에러 Logging
172.17.0.1 - - [21/Jun/2021:06:04:52 +0000] "GET /aaa.html HTTP/1.1" 404 153 "-" "curl/7.68.0" "-"
2021/06/21 06:04:52 [error] 31#31: *3 open() "/usr/share/nginx/html/aaa.html" failed (2: No such file or directory), client: 172.17.0.1, server: localhost, request: "GET /aaa.html HTTP/1.1", host: "localhost"

Ctrl + C <---- Server단에서 Service 중지

-Client 단에서 호출
$ curl localhost:80/aaa.html
curl : (7) Failed to connect to localhost port 80: Connection refused

4)Port binding 하기
-Server단에서 nginx 실행
docker run -p 8080:80 nginx
log 대기

-Client 단에서 접속
$ curl localhost:8080

-만일 $ curl localhost:80으로 연결하면
curl : (7) Failed to connect to localhost port 80: Connection refused

5)Tomcat 설치하기
-Tomcat Search
$ sudo docker search tomcat

-Tomcat Download
$ sudo docker pull consol/tomcat-8.0

-Tomcat Run
$ sudo docker run -d -p 8080:8080 --name tc consol/tomcat-8.0

-Tomcat Container check
$ sudo docker ps -a

-Web Browser에서 확인하기
http://container-ip:8080

-Tomcat Manager
http://container-ip:8080/manager/html

5. Docker Volume Mount하기
1)Server 단에서 MongoDB search
$ docker search mongodb

2)Server 단에서 MongoDB 실행하기
$ sudo docker run -v ${PWD}/data:/data/db mongo:4

3)Client 단에서 접속하기
$ ls -al
$ cd ./data
$ ls <----여러개의 파일과 디렉토리 확인
$ sudo docker ps <--MongoDB PID 확인

$ sudo docker exec -it PID(앞 2자리도 가능) mongo
>
> show dbs;
admin
config
local

>use example
switched to db example
>db.example.insert({"name":"Henry Instructor"})

```
    WriteResult({"nInserted" : 1})

    >db.example.find({})
    ....
    >exit

    $ Server 단에서 Ctrl + C 로 서비스 정지

  4)다시 Docker Run을 했을 때 Data가 남아 있을 것인가?
    -Server단에서 MongoDB 실행
      $ sudo docker run -v ${PWD}/data:/data/db mongo:4

    -Client 단에서 접속
      $ sudo docker ps   <--- PID확인

      $ sudo docker exec -it PID(앞 2자리도 가능) mongo
      >show dbs
        <--- example db 확인

      >use example
      >db.example.find({})
        <-- 앞에서 저장한 데이터 확인

  5)MongoDB Image 모두 삭제
  6)다시 Server 단에서 MongoDB Image Run

    $ sudo docker run mongo:4

  7)Client 단에서 접속
    $ sudo rm -rf ./data
    $ sudo docker exec -it PID mongo
    >show dbs
    >use example
    >db.example.insert({"name" : "Henry Instructor"})
    >db.example.find({})
    >exit

    -MongoDB Server도 Ctrl + C로 서비스 정지

  8)다시 MongoDB Server Start
    $ sudo docker run mongo:4

  9)Client 단에서 접속
    $ sudo docker exec -it PID mongo
    >
    >show dbs
      <---example db 없음.


6. Container Image 삭제하기
  1)Server 단에서 redis 실행하기
    # docker run -p 6379:6379 redis

  2)클라이언트 단에서
    $ sudo apt install redis-tools
    $ redis-cli
    127.0.0.1:6379>set name "Henry Instructor"
    OK
    127.0.0.1:6379>get name
    "Henry Instructor"
    127.0.0.1:6379>exit

    $ sudo docker ps -a   <-- PID 확인
    $ sudo docker rm PID --> 실패, 이유는 현재 Docker Container 실행 중
    $ sudo docker stop PID  <---클라이언트 세션에서 서버 서비스 중지시킴.

  3)Container 삭제하기
```

$ sudo docker ps -a <--- PID확인
$ sudo docker rm PID

4)Container Image 삭제하기
docker images <--- PID 확인
docker rmi PID

7. MySQL 사용하기
1)Docker로 MySQL Run
$ mkdir mysql
$ cd mysql
$ su -
# docker pull mysql:5.7.34
# docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=password -d -p 3306:3306 mysql:5.7.34
# docker ps -a

2)MySQL Workbench 설치하기
-https://dev.mysql.com/downloads/workbench/
-Windows (x86, 64-bit), MSI Installer 다운로드 후 설치

3)MySQL Workbench에서 Docker의 MySQL 연결하기
-MySQL Connection 추가
--Connection Name : docker-mysql
--Hostname : 192.168.56.101
--Port : 3306
--Usrname : root
--Password : Store in Vault ... 클릭 > Password : password > OK
--Test Connection Click
--OK
-docker-mysql double-click

4)Terminal 에서 연결하기
docker exec -it mysql-container bash
mysql -u root -p
Enter password : password
mysql > show databases;

mysql>exit
exit
docker rm -f mysql-container

8. Web Server를 만들어보기
1)Docker Image Pull
$ sudo docker pull httpd

$ sudo docker images

2)Docker Container 구동하기
-docker run 명령을 통해 Container 를 시작하고 Web 서비스를 구성 할 수 있다
$ sudo docker run httpd

-하지만, Container 가 Foreground 로 작동하면서 Shell 을 사용을 못할 뿐더러, Shell이 종료가 되면 httpd Container도 중지된다.
-위와 같이 되면, 전혀 서비스에 적용 할 수가 없다.
-그리하여 아래와 같이 background 로 container 를 실행하면 된다.

$ sudo docker run -d httpd
$ sudo docker ps -a

-Shell 에서 다른 명령도 가능하고 서비스가 계속 실행되는 것을 확인 할 수 있다.
-그럼 실제로 서비스가 작동하는지 확인해 본다.

$ curl http://127.0.0.1
curl: (7) Failed connect to 127.0.0.1; 연결이 거부됨.

-기존에 실행중이던 Container 중지

```
$ sudo docker stop [container ID]
$ sudo docker ps -a
```

-Port Binding

```
$ sudo docker run -d -p 80:80 httpd
$ sudo docker ps -a
```

-Service 확인하기

```
$ curl http://127.0.0.1
<html><body><h1>It works!</h1></body></html>
```

-Web Browser에서 확인할 것

3)index.html 수정하기

-Container 내부로 들어가서 index.html 수정하기

```
$ sudo docker -it [container ID] bash
root@419c02446fed:/# cd /usr/local/apache2/htdocs
root@419c02446fed:/usr/local/apache2/htdocs# cat index.html
<html><body><h1>It works!</h1></body></html>
root@419c02446fed:/usr/local/apache2/htdocs# echo "<html><body><h1>Docker Test Page</h1></body></html>" > index.html
root@419c02446fed:/usr/local/apache2/htdocs# exit
exit
$ curl http://127.0.0.1
<html><body><h1>Docker Test Page</h1></body></html>
```

-Web Browser에서 확인할 것