

```

1 Lab11. Docker Compose를 사용하는 이유
2
3 1. Docker 실행 명령어를 일일이 입력하기가 복잡해서
4     1)Nginx 실행하기
5         $ sudo docker run -it nginx
6         <--- 실행은 되지만 웹브라우저로 연결 불능, 바로 shell로 연결됨.
7
8     2)Nginx Container 실행 + Host 8080 Port 연결하기
9         $ sudo docker run -it -p 8080:80 nginx
10        <---실행도 되고 웹브라우저로 연결 가능하지만, 바로 shell로 연결됨.
11
12    3)Nginx Container 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제
13        $ sudo docker run -it -p 8080:80 --rm nginx
14        <---실행도 되고 웹브라우저로 연결 가능하지만, 바로 shell로 연결됨, Ctrl + C로 bash에 나오면 바로 Container
        종료됨.
15
16    4)Nginx 컨테이너 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제 + Host의 Directory를 Container
        안에서 링크하기
17        $ vi index.html
18        <h1>Hello, Docker Compose World!!!</h1>
19
20        $ sudo docker run -it -p 8080:80 --rm -v ${PWD}:/usr/share/nginx/html/ nginx
21        <---실행은 되지만, 웹브라우저로 접속시 403 (13: Permission denied) Error 발생
22        <---오류를 해결하려면 nginx Container의 /etc/nginx/nginx.conf파일 첫 줄에 user root;를 넣어야 함.
23        <---어쨌든 복잡함.
24
25
26 2. 컨테이너끼리 연결하기 편해서 --1) 후 바로 --3)실행
27     1)준비 : django-sample 이미지를 빌드
28         $ git clone https://github.com/raccoonny/django-sample-for-docker-compose.git django-sample
29         $ cd django-sample
30         $ docker build -t django-sample .
31
32     2)django 컨테이너 실행 + postgres 컨테이너 실행
33         $ docker run --rm -d --name django -p 8000:8000 django-sample
34
35         $ docker ps -a
36
37         -Web Browser를 열고 http://ip:8000
38         --django 잘 실행되고 있음을 확인
39
40         $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
41         > -e POSTGRES_USER=sampleuser \
42         > -e POSTGRES_PASSWORD=samplesecret \
43         > postgres
44
45         -Web Browser를 열고 http://ip:8000
46         --그냥 django만 잘 실행되고 있음.
47
48         -다음 3) 실행을 위해 모든 Docker Image와 Container 삭제하기
49         $ docker rm -f `docker ps -a -q`
50         $ docker rmi -f `docker images -q`
51
52         -django-sample image 다시 build
53         $ docker build -t django-sample .
54
55     3)postgres 컨테이너 실행 + django 컨테이너 실행 + 서로 연결하기
56         $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
57         > -e POSTGRES_USER=sampleuser \
58         > -e POSTGRES_PASSWORD=samplesecret \
59         > postgres

```

```

60
61 $ docker run -d --rm -p 8000:8000 -e DJANGO_DB_HOST=db ₩
62 > --link postgres:db ₩
63 > django-sample
64
65
66 3. 특정 컨테이너끼리만 통신할 수 있는 가상 네트워크 환경을 편리하게 관리하고 싶어서
67 1)postgres 컨테이너 실행 + django1 컨테이너 연결
68 $ docker run --rm -d --name postgres ₩
69 > -e POSTGRES_DB=djangosample ₩
70 > -e POSTGRES_USER=sampleuser ₩
71 > -e POSTGRES_PASSWORD=samplesecret ₩
72 > postgres
73
74 $ docker run -d --rm --name django1 ₩
75 > -p 8000:8000 ₩
76 > -e DJANGO_DB_HOST=db ₩
77 > --link postgres:db ₩
78 > django-sample
79
80 2)postgres 컨테이너는 호스트의 다른 컨테이너들이 모두 접근할 수 있음
81 $ docker run -d --rm --name django2 ₩
82 > -p 8001:8000 ₩
83 > -e DJANGO_DB_HOST=db ₩
84 > --link postgres:db ₩
85 > django-sample
86
87 3)postgres 컨테이너 + django1 컨테이너만 통신할 수 있는 가상 네트워크 만들기
88 -도커 네트워크 살펴보기
89 $docker network ls
90
91 -도커 네트워크 생성하기
92 $ docker network create --driver bridge web-service
93 $ docker network ls
94
95 -컨테이너 실행하기
96 $ docker run --rm -d --name postgres ₩
97 > --network web-service ₩
98 > -e POSTGRES_DB=djangosample ₩
99 > -e POSTGRES_USER=sampleuser ₩
100 > -e POSTGRES_PASSWORD=samplesecret ₩
101 > postgres
102
103 $ docker run -d --rm --name django1 ₩
104 > --network web-service ₩
105 > -p 8000:8000 ₩
106 > -e DJANGO_DB_HOST=db ₩
107 > --link postgres:db ₩
108 > django-sample
109
110 $ docker run -d --rm --name django2 ₩
111 > -p 8001:8000 ₩
112 > -e DJANGO_DB_HOST=db ₩
113 > --link postgres:db ₩
114 > django-sample
115 docker: Error response from daemon: Cannot link to /postgres, as it does not belong to the default
network.
116
117
118
119 4. 이 모든 것을 간단한 명령어로 관리하고 싶어서

```

1) 실행 명령어와 종료 명령어

```
$ docker network create --driver bridge web-service
```

```
$ docker run --rm -d --name postgres ₩  
> --network web-service ₩  
> -p 5432:5432 ₩  
> -e POSTGRES_DB=djangosample ₩  
> -e POSTGRES_USER=sampleuser ₩  
> -e POSTGRES_PASSWORD=samplesecret ₩  
> postgres
```

```
$ docker run -d --rm --name django1 ₩  
> --network web-service ₩  
> -p 8000:8000 ₩  
> -e DJANGO_DB_HOST=db ₩  
> --link postgres:db ₩  
> django-sample
```

```
$ docker kill django1 postgres
```

```
$ docker network rm web-service
```

2) docker-compose.yml 생성하기

```
$ cd django-sample
```

```
$ nano docker-compose.yml
```

```
version: '3'
```

```
volumes:
```

```
  postgres_data: {}
```

```
services:
```

```
  db:
```

```
    image: postgres
```

```
    volumes:
```

```
      - postgres_data:/var/lib/postgres/data <--- ':'과 '/'는 반드시 붙인다.
```

```
    environment:
```

```
      - POSTGRES_DB=djangosample
```

```
      - POSTGRES_USER=sampleuser
```

```
      - POSTGRES_PASSWORD=samplesecret
```

```
  django:
```

```
    build:
```

```
      context: .
```

```
      dockerfile: ./compose/django/Dockerfile-dev
```

```
      volumes:
```

```
        - ./app/
```

```
      command: ["/manage.py", "runserver", "0:8000"]
```

```
      environment:
```

```
        - DJANGO_DB_HOST=db
```

```
      depends_on:
```

```
        - db
```

```
      restart: always
```

```
      ports:
```

```
        - 8000:8000
```

3) 도커 컴포즈로 실행하고 종료하기

```
- 모든 docker process 중지
```

```
- 모든 docker images 삭제
```

```
- docker network도 기본 3개만 남기고 모두 삭제
```

```
$ docker-compose up -d <--- 반드시 django-sample directory 안에서 실행할 것
```

181 \$ docker-compose down
182
183 -웹 브라우저로 확인할 것