

```

1 1. Docker 실행 명령어를 일일이 입력하기가 복잡해서
2   1)Nginx 실행하기
3     $ docker run -it nginx
4   2)Nginx Container 실행 + Host 8080 Port 연결하기
5     $ docker run -it -p 8080:80 nginx
6   3)Nginx Container 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제
7     $ docker run -it -p 8080:80 --rm nginx
8   4)Nginx 컨테이너 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제 + Host의 Directory를 Container
9     안에서 링크하기
10    $ vi index.html
11    <h1>Hello, Docker Compose</h1>
12
13    $ docker run -it -p 8080:80 --rm -v ${PWD}:/usr/share/nginx/html/ nginx
14
15 2. 컨테이너끼리 연결하기 편해서 --1) 후 바로 --3)실행
16   1)준비 : django-sample 이미지를 빌드
17     $ git clone https://github.com/raccoonyy/django-sample-for-docker-compose.git django-sample
18     $ cd django-sample
19     $ docker build -t django-sample .
20
21   2)django 컨테이너 실행 + postgres 컨테이너 실행
22     $ docker run --rm -d --name django -p 8000:8000 django-sample
23
24     $ docker ps -a
25
26     -Web Browser를 열고 http://ip:8000
27     --django 잘 실행되고 있음을 확인
28
29     $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
30     > -e POSTGRES_USER=sampleuser \
31     > -e POSTGRES_PASSWORD=samplesecret \
32     > postgres
33
34     -Web Browser를 열고 http://ip:8000
35     --그냥 django만 잘 실행되고 있음.
36
37   3)postgres 컨테이너 실행 + django 컨테이너 실행 + 서로 연결하기
38     $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
39     > -e POSTGRES_USER=sampleuser \
40     > -e POSTGRES_PASSWORD=samplesecret \
41     > postgres
42
43     $ docker run -d --rm -p 8000:8000 -e DJANGO_DB_HOST=db \
44     > --link postgres:db \
45     > django-sample
46
47
48 3. 특정 컨테이너끼리만 통신할 수 있는 가상 네트워크 환경을 편리하게 관리하고 싶어서
49   1)postgres 컨테이너 실행 + django1 컨테이너 연결
50     $ docker run --rm -d --name postgres \
51     > -e POSTGRES_DB=djangosample \
52     > -e POSTGRES_USER=sampleuser \
53     > -e POSTGRES_PASSWORD=samplesecret \
54     > postgres
55
56     $ docker run -d --rm --name django1 \
57     > -p 8000:8000 \
58     > -e DJANGO_DB_HOST=db \
59     > --link postgres:db \
60     > django-sample
61
62   2)postgres 컨테이너는 호스트의 다른 컨테이너들이 모두 접근할 수 있음
63     $ docker run -d --rm --name django2 \
64     > -p 8001:8000 \
65     > -e DJANGO_DB_HOST=db \
66     > --link postgres:db \

```

```

67 > django-sample
68
69 3)postgres 컨테이너 + django1 컨테이너만 통신할 수 있는 가상 네트워크 만들기
70 -도커 네트워크 살펴보기
71 $docker network ls
72
73 -도커 네트워크 생성하기
74 $ docker network create --driver bridge web-service
75 $ docker network ls
76
77 -컨테이너 실행하기
78 $ docker run --rm -d --name postgres \
79 > --network web-service \
80 > -e POSTGRES_DB=djangosample \
81 > -e POSTGRES_USER=sampleuser \
82 > -e POSTGRES_PASSWORD=samplesecret \
83 > postgres
84
85 $ docker run -d --rm --name django1 \
86 > --network web-service \
87 > -p 8000:8000 \
88 > -e DJANGO_DB_HOST=db \
89 > --link postgres:db \
90 > django-sample
91
92 $ docker run -d --rm --name django2 \
93 > -p 8001:8000 \
94 > -e DJANGO_DB_HOST=db \
95 > --link postgres:db \
96 > django-sample
97
98
99 4. 이 모든 것을 간단한 명령어로 관리하고 싶어서
100 1)실행 명령어와 종료 명령어
101 $ docker network create --driver bridge web-service
102
103 $ docker run --rm -d --name postgres \
104 > --network web-service \
105 > -p 5432:5432 \
106 > -e POSTGRES_DB=djangosample \
107 > -e POSTGRES_USER=sampleuser \
108 > -e POSTGRES_PASSWORD=samplesecret \
109 > postgres
110
111 $ docker run -d --rm --name django1 \
112 > --network web-service \
113 > -p 8000:8000 \
114 > -e DJANGO_DB_HOST=db \
115 > --link postgres:db \
116 > django-sample
117
118 $ docker kill django1 postgres
119 $ docker network rm web-service
120
121 2)docker-compose.yml
122 version: '3'
123
124 volumes:
125     postgres_data: {}
126
127 services:
128     db:
129         image: postgres
130         volumes:
131             - postgres_data:/var/lib/postgres/data
132         environment:
133             - POSTGRES_DB=djangosample

```

```
134     - POSTGRES_USER=sampleuser
135     - POSTGRES_PASSWORD=samplesecret
136 django:
137   build:
138     context: .
139     dockerfile: ./compose/django/Dockerfile-dev
140     volumes:
141       - ./:/app/
142     command: ["/manage.py", "runserver", "0:8000"]
143     environment:
144       - DJANGO_DB_HOST=db
145     depends_on:
146       - db
147     restart: always
148     ports:
149       - 8000:8000
```

- 150
- 151 3)도커 컴포즈로 실행하고 종료하기
- 152 -모든 docker process 중지
  - 153 -모든 docker images 삭제
  - 154 -django-sample folder 삭제
- 155
- ```
156 $ docker-compose up -d
157 $ docker-compose down
```
- 158
- 159 -웹 브라우저로 확인할 것