

## 1 Lab6. Django Project Automatic Deployment with Jenkins II

### 1. Django Project Dockerizing Version 1

```
1)demo 디렉토리 생성
$ mkdir demo
$ cd demo

2)dockerfile 작성하기
$ nano dockerfile
FROM python:latest
RUN apt-get update
RUN apt-get -y upgrade
RUN apt-get -y dist-upgrade
RUN apt install -y nginx
RUN git clone https://github.com/gitinstructor/DjangoHome.git
WORKDIR /DjangoHome/myproject/
RUN pip install --upgrade pip
RUN pip install -r requirements.txt
RUN pip install gunicorn uwsgi

EXPOSE 80

CMD ["bash", "-c", "python manage.py runserver 0:80"]

3)build
$ docker build -t myweb:v1 .

4)이미지 확인
$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
myweb v1 dkfdkddjd 33 seconds ago 1.03GB

5)이미지 Container로 실행하기
$ docker run -dp 80:80 myweb:v1

6)Container 확인하기
$ docker ps -a

7)Web Browser에서 확인하기
```

### 2. Django Project Dockerizing Version 2

```
1)uwsgi 사용하기
2)dockerfile 수정하기
FROM python:latest

RUN apt-get update
RUN apt-get -y upgrade
RUN apt-get -y dist-upgrade
RUN apt install -y nginx
RUN git clone https://github.com/gitinstructor/DjangoHome.git

WORKDIR /DjangoHome/myproject/
RUN pip install --upgrade pip
RUN pip install -r requirements.txt
RUN pip install gunicorn uwsgi
RUN mkdir -p /var/log/uwsgi/myproject/
COPY ./myproject.ini /DjangoHome/myproject/.config/uwsgi/myproject.ini
CMD ["uwsgi", "--ini", "./.config/uwsgi/myproject.ini"]

EXPOSE 80

3)이미지에서 사용할 myproject.ini 새로 생성하기

[uwsgi]
chdir = /DjangoHome/myproject/

module = myproject.wsgi:application

uid = root
gid = root

http = :80

enable-threads = true
master = true
vacuum = true
pidfile = /tmp/myproject.pid
logto = /var/log/uwsgi/myproject/$(date +%Y-%m-%d).log
log-reopen = true

4)Container, Image 모두 지우기
$ docker rm -f `docker ps -a -q`
$ docker rmi -f `docker images -q`
```

```

85     $ docker volume prune
86     $ docker system prune -a
87
88 5)build
89     $ docker build -t myweb:v2 .
90
91 6)이미지 확인
92     $ docker images
93     REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
94     myweb                v2          dkfdfkddjd       33 seconds ago  1.03GB
95
96 7)이미지 Container로 실행하기
97     $ docker run -dp 80:80 myweb:v2
98
99 8)Container 확인하기
100    $ docker ps -a
101
102 9)Web Browser에서 확인하기
103
104
105 3. Django Project Dockerizing Version 3
106 1)Nginx와 연동하기
107 2)nginx.conf 파일 생성하기
108     $ nano nginx.conf
109         user root;                <---여기가 중요
110         worker_processes auto;
111         pid /run/nginx.pid;
112         include /etc/nginx/modules-enabled/*.conf;
113
114     events {
115         worker_connections 768;
116         # multi_accept on;
117     }
118
119     http {
120
121         ##
122         # Basic Settings
123         ##
124
125         sendfile on;
126         tcp_nopush on;
127         types_hash_max_size 2048;
128         # server_tokens off;
129
130         # server_names_hash_bucket_size 64;
131         # server_name_in_redirect off;
132
133         include /etc/nginx/mime.types;
134         default_type application/octet-stream;
135
136         ##
137         # SSL Settings
138         ##
139
140         ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
141         ssl_prefer_server_ciphers on;
142
143         ##
144         # Logging Settings
145         ##
146
147         access_log /var/log/nginx/access.log;
148         error_log /var/log/nginx/error.log;
149
150         ##
151         # Gzip Settings
152         ##
153
154         gzip on;
155
156         # gzip_vary on;
157         # gzip_proxied any;
158         # gzip_comp_level 6;
159         # gzip_buffers 16 8k;
160         # gzip_http_version 1.1;
161         # gzip_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss
162         text/javascript;
163
164         ##
165         # Virtual Host Configs
166         ##
167
168         include /etc/nginx/conf.d/*.conf;

```

```

168         include /etc/nginx/sites-enabled/*;
169     }
170
171 3)dockerfile 수정하기
172     FROM      python:latest
173
174     RUN        apt-get update
175     RUN        apt-get -y upgrade
176     RUN        apt-get -y dist-upgrade
177     RUN        apt install -y nginx
178     COPY       ./nginx.conf /etc/nginx/nginx.conf
179
180     RUN        git clone https://github.com/gitinstructor/DjangoHome.git
181     WORKDIR    /DjangoHome/myproject/
182     RUN        cp .config/nginx/myproject.conf /etc/nginx/sites-enabled/
183     CMD        ["nginx", "-g", "daemon on;"]
184
185     RUN        pip install --upgrade pip
186     RUN        pip install -r requirements.txt
187     RUN        pip install gunicorn uwsgi
188     RUN        mkdir -p /var/log/uwsgi/myproject
189     COPY       ./myproject.ini /DjangoHome/myproject/.config/uwsgi/myproject.ini
190
191     RUN        service nginx restart
192     CMD        ["uwsgi", "--ini", "./.config/uwsgi/myproject.ini"]
193
194     EXPOSE     80
195
196 4)myproject.ini 수정하기
197     [uwsgi]
198     chdir = /DjangoHome/myproject/
199
200     module = myproject.wsgi:application
201
202     uid = root
203     gid = root
204
205     http = :80                <---- 삭제할 것
206     socket = /tmp/myproject.sock <--- 새로 추가
207     chmod-socket = 666        <--- 새로 추가
208     chown-socket = root:root   <--- 새로 추가
209
210     enable-threads = true
211     master = true
212     vacuum = true
213     pidfile = /tmp/myproject.pid
214     logto = /var/log/uwsgi/myproject/$(exec://date +%Y-%m-%d).log
215     log-reopen = true
216
217 5)Container, Image 모두 지우기
218     $ docker rm -f `docker ps -a -q`
219     $ docker rmi -f `docker images -q`
220     $ docker volume prune
221     $ docker system prune -a
222
223 6)build
224     $ docker build -t myweb:v3 .
225
226 7)이미지 확인
227     $ docker images
228     REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
229     myweb            v3          dkfdkddjd    33 seconds ago 1.03GB
230
231 8)이미지 Container로 실행하기
232     $ docker run -dp 80:80 myweb:v3
233
234 9)Container 확인하기
235     $ docker ps -a
236
237 10)Web Browser에서 확인하기
238
239 11)하지만, 정상적으로 서비스가 진행되고 있지 않다. 그 이유는 nginx 서비스가 restart가 되지 않았기 때문이다.
240 12)그래서, Container ID로 들어가서 다음과 같이 명령을 실행하면 정상적으로 서비스가 진행된다.
241     $ docker exec -it dk(Container ID) bash
242     root@...:/DjangoHome/myproject# service nginx restart
243     Restarting nginx: nginx.
244     root@...:/DjangoHome/myproject# exit
245
246 13)Web Browser에서 확인하기
247
248
249 4. Django Project Dockerizing Version 4
250 1)dockerfile 수정하기
251     FROM      python:latest

```

```

252
253 RUN apt-get update
254 RUN apt-get -y upgrade
255 RUN apt-get -y dist-upgrade
256
257 RUN git clone https://github.com/gitinstructor/DjangoHome.git
258 WORKDIR /DjangoHome/myproject/
259
260 RUN pip install --upgrade pip
261 RUN pip install -r requirements.txt
262 RUN pip install gunicorn uwsgi
263 RUN mkdir -p /var/log/uwsgi/myproject
264 COPY ./myproject.ini /DjangoHome/myproject/.config/uwsgi/myproject.ini
265
266 CMD ["gunicorn", "--bind", "0:8000", "myproject.wsgi:application"]
267
268 EXPOSE 8000
269
270 2)docker-compose.yml 생성하기
271 $ nano docker-compose.yml
272     version: '3'
273     services:
274
275         django:
276             container_name: django
277             build:
278                 context: .
279                 dockerfile: ./dockerfile
280             restart: always
281             ports:
282                 - 80:8000
283
284 3)docker-compose로 build하기
285 $ docker-compose up -d
286
287 4)이미지 확인
288 $ docker images
289 REPOSITORY TAG IMAGE ID CREATED SIZE
290 demo-django latest ac378c01d36b About a minute ago 1.02GB
291
292 5)Web Browser에서 확인하기

```