

```
1 1. Apache2 Installation
2   $ mkdir demo
3   $ cd demo
4   $ nano Dockerfile
5       FROM ubuntu:20.04
6
7       LABEL Author=Instructor
8       LABEL Email=javaexpert.com
9
10      ENV DEBIAN_FRONTEND=noninteractive
11
12      RUN apt-get update && apt-get -y install apache2
13
14      EXPOSE 80
15
16      CMD ["/usr/sbin/apachectl", "-DFOREGROUND"]
17
18  $ sudo docker build -t mywebserver .
19
20  $ sudo docker run -d -p 60000:80 mywebserver
21
22  -Web Browser 에서 확인할 것
23
24
25 2. Php 7.4 Installation
26   1) 모든 Process 종료하기
27       $ sudo docker rm -f `sudo docker ps -a -q`
28
29   2) Dockerfile 수정
30       $ nano Dockerfile
31           FROM ubuntu:20.04
32
33           LABEL Author=Instructor
34           LABEL Email=javaexpert.com
35
36           ENV DEBIAN_FRONTEND=noninteractive
37
38           RUN apt-get update && apt-get -y install apache2
39           RUN apt-get -y install software-properties-common
40           RUN add-apt-repository ppa:ondrej/php
41           RUN apt-get update && apt-get -y install php7.4
42
43           EXPOSE 80
44
45           CMD ["/usr/sbin/apachectl", "-DFOREGROUND"]
46
47   3) 재 빌드
48       $ sudo docker build -t mywebserver .
49
50   4) Docker Image Run
51       $ sudo docker run -d -p 60000:80 -v /home/ubuntu/demo/html:/var/www/html
52       mywebserver
53
54   5) index.php 파일 생성
55       $ cd html
56       $ sudo nano index.php
57           <?php
58               phpinfo();
59           ?>
```

60 6)Web Browser 에서 확인할 것

61 62 63 3. MySQL Installation

64 1)모든 Docker Process 삭제

65 \$ sudo docker ps -a

66 \$ sudo docker rm -f `sudo docker ps -a -q`

67
68 \$ sudo docker images

69 \$ sudo docker rmi -f `sudo docker images`

70 71 2)MySQL 실행하기

72 \$ sudo docker run -d -p 3306:3306 --name db -e
MYSQL_ROOT_PASSWORD=password mysql:5.7

73
74 \$ sudo docker inspect db

75 "Networks" > "IPAddress" 확인할 것

76
77 \$ sudo docker exec -it db bash

78 /# mysql -h 172.17.0.2 -u root --port 3306 -p

79 Enter password:password

80 mysql>

81 82 83 4. PHP7.4 + MySQL

84 1)모든 Docker Process 삭제

85 \$ sudo docker ps -a

86 \$ sudo docker rm -f `sudo docker ps -a -q`

87
88 \$ sudo docker images

89 \$ sudo docker rmi -f `sudo docker images`

90 91 2)Dockerfile 수정하기

92 FROM ubuntu:20.04

93
94 LABEL Author=Instructor

95 LABEL Email=javaexpert.com

96
97 ENV DEBIAN_FRONTEND=noninteractive

98
99 RUN apt-get update && apt-get -y install apache2

100 RUN apt-get -y install software-properties-common

101 RUN add-apt-repository ppa:ondrej/php

102 RUN apt-get update && apt-get -y install php7.4

103 RUN apt-get -y install php7.4-mysql <---추가

104
105 EXPOSE 80

106
107 CMD ["/usr/sbin/apachectl", "-DFOREGROUND"]

108 109 3)재 빌드

110 \$ sudo docker build -t mywebserver .

111 112 4)MySQL Run

113 \$ sudo docker run -d --name db -p 3306:3306 -v

/home/ubuntu/dbdata:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=password
mysql:5.7

114
115 \$ sudo docker exec -it db bash

116 /# mysql -h localhost -u root -p

```

117
118 mysql> CREATE DATABASE Employee CHARACTER SET utf8;
119 Query OK, 1 row affected (0.00 sec)
120
121 mysql> CREATE USER scott IDENTIFIED BY 'tiger';
122 Query OK, 0 rows affected (0.00 sec)
123
124 mysql> GRANT ALL PRIVILEGES ON *.* TO scott@'%';
125 Query OK, 0 rows affected (0.00 sec)
126
127 mysql> FLUSH PRIVILEGES;
128 Query OK, 0 rows affected (0.00 sec)
129
130 mysql>exit
131
132 5)Web Server Run
133 $ sudo docker run -d -p 60000:80 -v /home/ubuntu/demo/html:/var/www/html
  mywebserver
134
135 6)index.php 수정
136 <?php
137     $conn = mysqli_connect(
138         '172.17.0.2',
139         'scott',
140         'tiger',
141         'Employee',
142         '3306'
143     );
144     if(mysqli_connect_errno()){
145         echo "Failed to connect to MySQL: ".mysqli_connect_error();
146     }
147     $sql = "SELECT VERSION()";
148     $result = mysqli_query($conn, $sql);
149     $row = mysqli_fetch_array($result);
150     print_r($row["VERSION()"]);
151 ?>
152
153 7)Web Browser에서 확인
154 5.7.34
155
156
157 5. Git에 Docker Project 올리기
158 1)github에 login
159 https://github.com/gitinstructor
160
161 2)Private Repository 생성
162 -왜 Private?
163 --Database 인증정보처럼 민감한 데이터가 있기 때문...
164
165 3)현재 Cloud의 VM에는 Git이 기본적으로 설치되어 있음.
166 $ git --version으로 확인
167
168 4)방금 생성한 Repository를 Cloud VM에서 git clone 할 것
169 $ cd ~
170 $ git clone https://github.com/gitinstructor/docker-demo.git
171 Cloning into 'docker-demo'...
172 Username for 'https://github.com': gitinstructor <-- Repository를 Private로 생성했기
  때문
173 Password for 'https://github.com':
174 warning: You appear to have cloned an empty repository.

```

5)앞에서 생성한 Dockerfile 과 index.php를 clone한 docker-demo에 같이 복사해서 넣을 것
-일반적으로 Dockerfile과 index.php같은 소스코드를 같이 넣는다.

```
$ cd docker-demo/  
$ cp ~/demo/Dockerfile .  
$ cp ~/demo/html/index.php .  
$ ls  
Dockerfile index.php
```

6)Github에 Push하기

```
$ git add .  
$ git commit -m "Project Initialization"  
[master (root-commit) f49a009] Project Initialization  
Committer: ubuntu <ubuntu@localhost.localdomain>  
Your name and email address were configured automatically based  
on your username and hostname. Please check that they are accurate.  
You can suppress this message by setting them explicitly. Run the  
following command and follow the instructions in your editor to edit  
your configuration file:
```

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

```
2 files changed, 32 insertions(+)  
create mode 100644 Dockerfile  
create mode 100644 index.php
```

```
$ git push  
Username for 'https://github.com': gitinstructor  
Password for 'https://gitinstructor@github.com':  
Enumerating objects: 4, done.  
Counting objects: 100% (4/4), done.  
Compressing objects: 100% (4/4), done.  
Writing objects: 100% (4/4), 743 bytes | 743.00 KiB/s, done.  
Total 4 (delta 0), reused 0 (delta 0)  
To https://github.com/gitinstructor/docker-demo.git  
* [new branch] master -> master
```

7)Github에서 Refresh해서 확인할 것

6. DockerHub와 GitHub 연동하기

1)DockerHub에 Login

2)DockerHub에 Private Repository 생성하기

- 이름 : docker-demo
- Visibility : Private
- Build Settings : GitHub Connected(이미 GitHub와 DockerHub가 연동되어 있을 경우) 클릭
- Select organization : git 계정 선택
- Select repository : docker-demo
- Create & Build button click
- 그럼 DockerHub에서 Build가 진행됨.
- 화면 우측의 Recent builds에 목록에 보면 github계정/repository가 있고 그 아래에 Build in 'master'의 아이콘이 노란색으로 현재 빌드중임을 알려준다.
- github계정/repository 링크를 클릭하면 In Progress가 현재 빌드 진행중임을, 그리고 빌드가 마치고 성공했다면 SUCCESS 초록색 글자가 나타난다.
- 그 아래에 Build Log도 보이고 Dockerfile도 볼 수 있고 Readme도 볼 수 있다.
- 이 Build Log는 우리가 수동으로 빌드했을 때의 로그와 동일하다.

-다시 General에 가면 노란색 버튼이 초록색 체크 아이콘으로 변경된 것을 볼 수 있다.
-그래서 이제까지 우리가 Cloud에서 생성했던 모든 Docker 이미지를 삭제해도 된다는 의미이다.

3) Cloud에 있는 모든 Docker Image 삭제

```
$ sudo docker rm -f `sudo docker ps -a -q`  
$ sudo docker rmi -f `sudo docker images`  
$ sudo docker ps -a  
$ sudo docker images
```

4) Github docker-demo repository의 README 파일 생성하기

-docker-demo repository에서 [Add a README] 클릭

```
# Docker를 사용하는 진짜 이유 : CI/CD  
### Installation  
<pre>      <--일반적으로 pre tag로 작성  
cd /home  
sudo git clone https://github.com/gitinstructor/docker-demo <---github  
repository 경로  
cd docker-demo  
</pre>  
  
### Run  
<pre>  
# Login for Private Docker Repository  
sudo docker login  
sudo docker pull pythonexpert/docker-demo  
sudo docker run -d --name db -p 3306:3306 -v  
/home/ubuntu/dbdata:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=password  
mysql:5.7  
sudo docker run -p 60000:80 -v /home/docker-demo/Project:/var/www/html  
pythonexpert/docker-demo
```

-README 파일 COMMIT

-이렇게 하면 local machine에서 코드를 수정하면 그 변경된 내용이 commit만 하면 자동으로 dockerhub에서 build를 한다.

5) 소스쪽에서 수정하고 다시 github에 push 하기

```
$ cd ~/docker-demo  
$ mkdir Project  
$ mv index.php ./Project/index.php  
$ ls ./Project  
index.php  
$ git add .  
$ git commit -m "index.php path changed"  
[master 066cc46] index.php path changed  
Committer: ubuntu <ubuntu@localhost.localdomain>  
Your name and email address were configured automatically based  
on your username and hostname. Please check that they are accurate.  
You can suppress this message by setting them explicitly. Run the  
following command and follow the instructions in your editor to edit  
your configuration file:
```

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

```
1 file changed, 0 insertions(+), 0 deletions(-)  
rename index.php => Project/index.php (100%)
```

```
288 $ git push
289 Username for 'https://github.com': gitinstructor
290 Password for 'https://gitinstructor@github.com':
291 To https://github.com/gitinstructor/docker-demo.git
292 ! [rejected]      master -> master (fetch first)
293 error: failed to push some refs to 'https://github.com/gitinstructor/docker-demo.git'
294 hint: Updates were rejected because the remote contains work that you do
295 hint: not have locally. This is usually caused by another repository pushing
296 hint: to the same ref. You may want to first integrate the remote changes
297 hint: (e.g., 'git pull ...') before pushing again.
298 hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

-오류발생...그 이유는 README 파일을 수정했기 때문, 그러면 pull을 하면 됨.

```
303 $ git pull
304 Username for 'https://github.com': gitinstructor
305 Password for 'https://gitinstructor@github.com':
```

-그러면 README 파일을 불러옴
-바로 Ctrl + X로 빠져 나옴.
-다시 push 하면 반영됨.

```
311 $ git push
312 Username for 'https://github.com': gitinstructor
313 Password for 'https://gitinstructor@github.com':
314 Enumerating objects: 7, done.
315 Counting objects: 100% (7/7), done.
316 Compressing objects: 100% (4/4), done.
317 Writing objects: 100% (5/5), 596 bytes | 298.00 KiB/s, done.
318 Total 5 (delta 1), reused 0 (delta 0)
319 remote: Resolving deltas: 100% (1/1), done.
320 To https://github.com/gitinstructor/docker-demo.git
321    ed26cc5..f6369c0  master -> master
```

6)Github에서 Refresh 하면 docker-demo Repository에 Project 폴더가 생겼고, 그 폴더 안에 index.php 파일이 있는 것을 확인할 수 있다.

7)GitHub에서 변경된 것을 DockerHub가 감지하면 바로 다시 Build 한다.

8)만일 성공적으로 빌드가 되면, 화면 Refresh했을 때, 방금 SUCCESS 빌드한 빌드 상세 페이지의 README 파일이 위에서 새로 생성한 README 파일로 변경된 것을 확인할 수 있다.

7. DockerHub의 READMD 파일대로 수행해 보기

1)현재 모든 Docker Image혹은 Docker Container가 모두 없다는 것을 확인한다 .

```
$ sudo docker ps -a
$ sudo docker images
```

2)DockerHub의 README 파일의 내용대로 그대로 해 본다.

3)Web Browser에서 확인하면
5.7.34

8. Jenkins를 이용해서 Docker Project Build 하기

1)현재 모든 Docker Image혹은 Docker Container가 모두 없다는 것을 확인한다 .

```
$ sudo docker ps -a
$ sudo docker images
```

2)Jenkins Container Image Pull

```
$ sudo docker pull jenkins/jenkins
```

```
346 $ sudo docker run -dp 60000:8080 -v /home/jenkins:/var/jenkins_home -v  
/var/run/docker.sock:/var/run/docker.sock -u root jenkins/jenkins
```

347

348 3)Web Browser에서

349 -http://{EIP}:8080

350 -Administrator password :

351 -어드민 패스워드는 \$ sudo docker logs jenkins's pid 에서 비밀번호 확인할 것

352 -그리고 Continue 버튼 클릭...

353 -Install suggested plugins 버튼 클릭

354 -혹 설치 에러가 나도 [continue] 링크 클릭

355 -관리자 계정 생성하기

356 -아이디 : admin/P@\$W0rd

357 > Save and Continue

358

359

360 4)Jenkins page에서

361 -왼쪽 프레임에서 새로운 item 클릭

362 -Enter an item name : example

363 -Freesstyle project 선택

364 -Build > Add build step > Execute shell

365 -Command : DockerHub의 README에서

366 sudo docker pull pythonexpert/docker-demo

367 sudo docker run -d --name db -p 3306:3306 -v

/home/ubuntu/dbdata:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=password
mysql:5.7

368 sudo docker run -p 60000:80 -v /home/docker-demo/Project:/var/www/html
pythonexpert/docker-demo

369 -저장

370 -다시 젠킨스 페이지에서 좌측 프레임의 [Build Now] 클릭

371 -Console Output를 클릭하여 상세 내용 볼 수 있음.

372 -jenkins Container 안에는 docker가 설치안되어 있으면 에러 발생함.

373

374 \$ sudo docker exec -it jenkins' pid bash

375 /# docker 설치할 것