

포팅 매뉴얼

☀ 상태	Not started
👤 담당자	
📁 분야	공통

Gitlab 정리 문서

사용 JVM :

Ubuntu

20.04 LTS

Java

openjdk version "1.8.0_342"

OpenJDK Runtime Environment (build 1.8.0_342-8u342-b07-0ubuntu1~20.04-b07)

OpenJDK 64-Bit Server VM (build 25.342-b07, mixed mode)

Docker

Docker version 20.10.18

1-3. 배포시 특이사항

▼ nginx with Docker

```
sudo docker container run --name webserver -d -p 80:80 nginx
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

```
ubuntu@ip-172-26-7-221:~$ sudo docker exec -it c0728b11a10b /bin/bash
root@c0728b11a10b:/# l
bash: l: command not found
root@c0728b11a10b:/# ls
bin  dev  docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot  docker-entrypoint.d  etc  lib  media  opt  root  sbin  sys  usr
root@c0728b11a10b:/# cd etc/
root@c0728b11a10b:/etc# ls
.pwd.lock  group  mke2fs.conf  rc6.d/
adduser.conf  group-  motd  rcS.d/
alternatives/  gshadow  mtab  resolv.conf
apt/  gshadow-  netconfig  rmt
bash.bashrc  gss/  nginx/  security/
bindresvport.blacklist  host.conf  nsswitch.conf  selinux/
ca-certificates/  hostname  opt/  shadow
ca-certificates.conf  hosts  os-release  shadow-
cron.d/  init.d/  pam.conf  shells
cron.daily/  inputrc  pam.d/  skel/
debconf.conf  issue  passwd  ssl/
debian_version  issue.net  passwd-  subgid
default/  kernel/  profile  subuid
deluser.conf  ld.so.cache  profile.d/  systemd/
dpkg/  ld.so.conf  rc0.d/  terminfo/
e2scrub.conf  ld.so.conf.d/  rc1.d/  timezone
environment  libaudit.conf  rc2.d/  ucf.conf
fonts/  localtime  rc3.d/  update-motd.d/
fstab  login.defs  rc4.d/  xattr.conf
gai.conf  logrotate.d/  rc5.d/
root@c0728b11a10b:/etc# cd etc/nginx/
root@c0728b11a10b:/etc/nginx# ls
conf.d/  fastcgi_params  mime.types  modules/  nginx.conf  scgi_params  uwsgi_params
root@c0728b11a10b:/etc/nginx# cd conf.d/
root@c0728b11a10b:/etc/nginx/conf.d# ls
default.conf
root@c0728b11a10b:/etc/nginx/conf.d#
```

Docker에서 NginX 설치

Docker로 Nginx를 설치하고 설정파일을 변경하는 것을 알아본다. 이미지를 다운로드 받는다.

ubuntu@DESKTOP-POHP5V7:~\$ sudo docker pull nginx [sudo] password for ubuntu: Using default tag:

latest latest: Pulling from library/nginx b4d181a07f80: Pull complete edb81c9bc1f5: Pull complete

https://velog.io/@latte_h/Docker%EC%97%90%EC%84%9C-NginX-%EC%84%A4%EC%B9%98

velog

docker container에 접속하기

IT/Docker 클송 2015. 7. 21. 00:43 일단 docker 라는 것을 처음 접하고, container를 실행해 본 후 가장 당황스러웠던 것은 "container에 어떻게 접속하지?" 일단 container 자체가 하나의 Process 였기에 container가 daemon으로 실행하고 나면 여기에 어떻게 접속해야할지 난감한 상황이.. root@~~# docker ps -a

<https://bluese05.tistory.com/21>



[Docker] Docker 컨테이너에 vim 설치하기

Docker 컨테이너 bin/bash에 vim 설치 Install Vim on Docker Container docker 컨테이너에서 vi 로 파일을 수정하려고 했으나 vi가 동작 하지 않고 아래와 같은 에러가 발생하였다. /bin/sh: 38: vi: not found..

<https://wecandev.tistory.com/64>



[Nginx] 심볼릭 링크 연결

Nginx는 sites-available에 파일을 작성하고, 실제 연결되어야 하는 파일만 심볼릭 링크를 통해 sites-enabled 폴더로 연결할 수 있습니다. conf 파일 만들기 test.conf를 sites-enabled에 연결하는 심볼릭 링크 만들기

<https://2vup.com/nginx-symbolic-link/>

안녕하세요.
새로운 것을 찾는데
Jamie입니다.



```

server {
    listen      80;
    listen  [::]:80;
    server_name localhost;

    #access_log /var/log/nginx/host.access.log main;

    location / {
        root    /usr/share/nginx/html;
        index   index.html index.htm;
    }

    #error_page 404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504 /50x.html;
    location = /50x.html {
        root    /usr/share/nginx/html;
    }

    # proxy the PHP scripts to Apache listening on 127.0.0.1:80
    #
    #location ~ \.php$ {
    #    proxy_pass http://127.0.0.1;
    #}

    # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
    #
    #location ~ \.php$ {
    #    root            html;
    #    fastcgi_pass    127.0.0.1:9000;
    #    fastcgi_index   index.php;
    #    fastcgi_param   SCRIPT_FILENAME /scripts$fastcgi_script_name;
    #    include          fastcgi_params;
    #}

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny all;
    #}
}

```

docker 컨테이너 접속

```
sudo docker exec -it [CONTAINER_ID] /bin/bash
```

```

ubuntu@ip-172-26-7-221:~/NGINX_Setting$ sudo vi conf.d/default.conf
ubuntu@ip-172-26-7-221:~/NGINX_Setting$ sudo docker cp ./conf.d/default.conf nginx-webserver:/etc/nginx/conf.d/
ubuntu@ip-172-26-7-221:~/NGINX_Setting$ sudo docker exec nginx-webserver nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-172-26-7-221:~/NGINX_Setting$

```

▼ nginx with docker-compose and ssl certbot

```

ubuntu@ip-172-26-7-221:~/NGINX_Setting$
conf.d data docker-compose.yml

```

data 는 아래 cmd 하니 자동 생성 됐었음.

```
sudo docker-compose up -d
```

docker ps 하니 잘 떴더라 (nginx, certbot)(내리지 말자!)

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED       STATUS       PORTS                               NAMES
c72ad9f9138   certbot/certbot  "/bin/sh -c 'python3 ..."  2 minutes ago  Up 2 minutes  0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp  nginx_settings_certbot_1
df9bee68218   nginx:latest    "/docker-entrypoint..."  2 minutes ago  Up 2 minutes  0.0.0.0:8080->8080/tcp, :::8080->8080/tcp  nginx_settings_nginx_1
b0dc15fede   springio/gs-spring-boot-docker  "java -jar /app.jar"      2 days ago    Up 2 days    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp  spring-boot-server
7f9fad2f12    mysql:latest    "docker-entrypoint.sh ..."  2 days ago    Up 2 days    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp  mysql-container
ubuntu@ip-172-26-7-221:~/NGINX_Settings$
```

cmd 좀 다르더라

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ curl -L https://raw.githubusercontent.com/wmnnd/nginx-certbot/master/init-letsencrypt.sh > init-letsencrypt.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 2491 100 2491  0     0  8114      0 --:--:-- --:--:-- --:--:-- 8114
```

init sh 파일 수정 (도메인, 경로, 이메일)

```
domains=(example.org www.example.org)
rsa_key_size=4096
data_path="./data/certbot"
email="" # Adding a valid address is strongly recommended
staging=0 # Set to 1 if you're testing your setup to avoid hitting request limits

domains=(j7d209.p.ssafy.io)
rsa_key_size=4096
data_path="./data/certbot"
email="inseok9876@gmail.com" # Adding a valid address is strongly recommended
staging=0 # Set to 1 if you're testing your setup to avoid hitting request limits
```

[Docker] Docker 환경 Nginx에 SSL 인증서 적용하기(Let's Encrypt)

Nginx에 SSL 인증서를 적용한 내용을 공유합니다. 컨테이너 상의 Nginx에 인증서를 적용하다 보니 방법이 잘 이해가 가지 않아 생각보다 헤매었네요 😊 Let's Encrypt 유효성 검사를 완료하려면 Nginx가 필요하지만 인증서가 없으면 nginx가 시작되지 않습니다. 그래서 임시로 https를 사용하지 않는 척 nginx를 실행한 다음 인증서를 받습니다. <https://node-js.tistory.com/32>

이 사이트는 보안 연결(HTTPS)이 사용되었습니다

쿠키 36개 사용 중

사이트 설정

Docker의 'ERROR Couldn't connect to Docker daemon' 에러 해결 방법

우분투에서 Docker와 Docker-Compose를 사용하려고 했는데, 아래와 같이 "Couldn't connect to Docker daemon at http+docker://localunixsocket - is it running?" 에러가 발생하였습니다. 해결방법에 대해서 정리하였습니다. \$ docker-compose up -d ERROR: Couldn't connect to Docker daemon at <https://codechacha.com/ko/fix-couldnt-connect-to-docker-daemon/>



docker-compose up 이 안되길래 찾아봤었음

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo docker exec nginx_settings_nginx_1 nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo docker exec nginx_settings_nginx_1 nginx -s reload
```

Certbot 컨테이너는 이제 SSL 인증서를 새로 발급하거나 재발급할 때만 필요하고 다른 때에는 필요하지 않음

```
$ docker-compose up --build -d nginx
```

재발급

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo ./init-letsencrypt.sh
Existing data found for j7d209.p.ssafy.io. Continue and replace existing certificate? (y/N) y
### Downloading recommended TLS parameters ...
```

```
### Reloading nginx ...
2022/09/19 14:21:56 [notice] 28#28: signal process started
ubuntu@ip-172-26-7-221:~/NGINX_Settings$
```

sudo docker nginx reload 해도 되나?

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                                NAMES
7cb6eae8bda    nginx:latest                        "/docker-entrypoint..." 16 minutes ago Up 16 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp nginx_settings_nginx_1
8f83e929876a    certbot/certbot                    "certbot"                21 minutes ago Restarting (1) 40 seconds ago 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp nginx_settings_certbot_1
bdc6dc15feda    springio/gs-spring-boot-docker     "java -jar /app.jar"      5 days ago    Up 5 days    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp spring-boot-server
7f9fad8f132    mysql:latest                        "/docker-entrypoint..." 5 days ago    Up 5 days    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp mysql-container
c8728b1a10b    nginx                               "/docker-entrypoint..." 6 days ago    Exited (0) 5 days ago                                webserver
ubuntu@ip-172-26-7-221:~/NGINX_Settings$
```

아하.. 제대로 뜨지가 않았었네

443 server 에서 server_name 이 servername 으로 되어있더라? 고치니 잘 됨! (한참 걸렸네!)

https 설정.

▼ Springboot with docker using gradle

요약

그라이들 빌드

```
./gradlew build
```

도커 이미지 빌드

```
sudo docker build -f Dockerfile -t springieng .
```

도커 컨테이너 실행

```
sudo docker run -it -d -p 8080:8080 --name spring springieng
```

과정

./gradlew build 로 빌드(jar 파일 생성)

```
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo ./gradlew build
sudo: ./gradlew: command not found
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew
-bash: ./gradlew: Permission denied
```

안되서 접근 명령 설정 해주니 됨.

```

Dockerfile build.gradle gradle gradlew gradlew.bat settings.gradle src
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew build
-bash: ./gradlew: Permission denied
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo ./gradlew build
sudo: ./gradlew: command not found
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo gradlew build
sudo: gradlew: command not found
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ls
Dockerfile build.gradle gradle gradlew gradlew.bat settings.gradle src
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo ./gradlew build
sudo: ./gradlew: command not found
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew
-bash: ./gradlew: Permission denied
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew build
-bash: ./gradlew: Permission denied
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ chmod +x gradlew
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew build
-bash: ./gradlew: No such file or directory
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ ./gradlew build
Starting a Gradle Daemon (subsequent builds will be faster)

> Task :compileJava
Note: /home/ubuntu/gitLab/S07P22D209/backend/src/main/java/com/ieng/ieng/global/response/CommonResponse.java
uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

> Task :test
2022-09-24 08:04:12.341 INFO 952977 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closin
g JPA EntityManagerFactory for persistence unit 'default'
2022-09-24 08:04:12.344 INFO 952977 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : Hikari
Pool-1 - Shutdown initiated...
2022-09-24 08:04:12.354 INFO 952977 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : Hikari
Pool-1 - Shutdown completed.

BUILD SUCCESSFUL in 54s
7 actionable tasks: 7 executed
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ █

```

Dockerfile 생성

```

FROM openjdk:8-jdk-alpine
ARG JAR_FILE=build/libs/back-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
~
~
~

```

docker build (뒤에 온점 찍어야함)

```
sudo docker build -f Dockerfile -t springieng .
```

```

ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker build -f Dockerfile -t springieng .
Sending build context to Docker daemon 51.69MB
Step 1/4 : FROM openjdk:8-jdk-alpine
--> a3562aa0b991
Step 2/4 : ARG JAR_FILE=build/libs/ieng-0.0.1-SNAPSHOT.jar
--> Running in 4da5d1fa1199
Removing intermediate container 4da5d1fa1199
--> efb87949c2f5
Step 3/4 : COPY ${JAR_FILE} app.jar
--> 499385c451d1
Step 4/4 : ENTRYPOINT ["java","-jar","/app.jar"]
--> Running in 4aee23cd0073
Removing intermediate container 4aee23cd0073
--> 5d23227677c9
Successfully built 5d23227677c9
Successfully tagged springieng:latest
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ █

```

images 체크함

```

ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
springieng          latest             5d23227677c9       About a minute ago 155MB
springio/g5-spring-boot-docker  latest             4132d48b3f34       10 days ago        145MB
certbot/certbot     latest             100ff950389a       2 weeks ago        117MB
mysql               latest             ff3b5098b416       3 weeks ago        447MB
nginx               latest             2b7d6430f78d       4 weeks ago        142MB
openjdk             8-jdk-alpine      a3562aa0b991       3 years ago        105MB
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$

```

그전에 8080 토피프로젝트 올려뒤서 failed

```

ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ docker run -p 8080:8080 --name spring springieng
docker: Error response from daemon: driver failed programming external connectivity on endpoint spring (d2805153a40ad61678013bb954dd018c7107abf3b916601cd17de2be134790fc): Bind for 0.0.0.0:8080 failed: port is already allocated.
ERROR[0000] error waiting for container: context canceled
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$

```

다시 돌리니 됨.

```

ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker container ls -a
CONTAINER ID   IMAGE      COMMAND                  CREATED    STATUS      PORTS
421c6bca3a44   certbot/certbot  "certbot"               4 days ago  Restarting (1) 13 seconds ago  0.0.0.0:80→80/tcp, :::80→80/tcp, 0.0.0.0:443→443/tcp,
3624d7519894   nginx:latest  "/docker-entrypoint..." 4 days ago  Up 4 days    0.0.0.0:3306→3306/tcp, :::3306→3306/tcp, 33060/tcp
7f9fadcf132    mysql:latest  "docker-entrypoint..." 10 days ago  Up 31 hours   0.0.0.0:3306→3306/tcp, :::3306→3306/tcp, 33060/tcp
c0728011a10b   nginx        "/docker-entrypoint..." 11 days ago  Exited (0) 10 days ago
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker run -p 8080:8080 --name spring springieng 5d23227677c9

```

로그인도 됨

근데 백그라운드 안돌아가더라 도커컨테이너 지우고 다시 런 명령어 침

```
sudo docker run -it -d -p 8080:8080 --name spring springieng
```

sudo docker run -it -d -p 8080:8080 -e TZ=Asia/Seoul --name spring springieng

-d : 백그라운드 실행

```

ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker run -it -d -p 8080:8080 --name spring springieng
a81c443cead76ca956eb23442c45ac4fa7411fe98d74ae4b95a288999fad5fba
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED    STATUS      PORTS
a81c443cead7   springieng  "java -jar /app.jar"     4 seconds ago  Up 4 seconds    0.0.0.0:8080→8080/tcp, :::8080→8080/tcp
421c6bca3a44   certbot/certbot  "certbot"               4 days ago  Restarting (1) 11 seconds ago
nginx_settings_certbot_1
3624d7519894   nginx:latest  "/docker-entrypoint..." 4 days ago  Up 4 days    0.0.0.0:80→80/tcp, :::80→80/tcp, 0.0.0.0:443→443/tcp, :::443→443/t
cp      nginx_settingsnginx_1
7f9fadcf132    mysql:latest  "docker-entrypoint..." 10 days ago  Up 31 hours   0.0.0.0:3306→3306/tcp, :::3306→3306/tcp, 33060/tcp
mysql-container
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED    STATUS      PORTS      NAMES
a81c443cead7   springieng  "java -jar /app.jar"     12 seconds ago  Up 12 seconds    0.0.0.0:8080→8080/tcp, :::8080→8080/tcp  spring
421c6bca3a44   certbot/certbot  "certbot"               4 days ago  Restarting (1) 18 seconds ago  0.0.0.0:80→80/tcp, :::80→80/tcp, 0.0.0.0:443→443/tcp, :::443→443/tcp  nginx_settings_certbot_1
3624d7519894   nginx:latest  "/docker-entrypoint..." 4 days ago  Up 4 days    0.0.0.0:80→80/tcp, :::80→80/tcp, 0.0.0.0:443→443/tcp, :::443→443/tcp  nginx_settingsnginx_1
7f9fadcf132    mysql:latest  "docker-entrypoint..." 10 days ago  Up 31 hours   0.0.0.0:3306→3306/tcp, :::3306→3306/tcp, 33060/tcp  mysql-container
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/backend$

```

gradlew Permission Denied

Mac , Android Studio 환경에서 터미널로 앱 빌드를 하고 싶으면 ./gradlew build 위와 같은 명령어로 빌드를 실행할 수 있다. bash: ./gradlew: Permission denied 그러나 위와 같이 Permission denied 가 발생하면 chmod +x gradlew 를 터미널에 입력 한 후, 다시 ./gradlew build 를 실행하면 빌드가 되는 것을 확인할 수 있다.

☞ <https://javalism.tistory.com/101>



gradle build permission denied

Docker #10. 스프링부트(spring-boot) Docker로 구동하기

스프링부트(spring-boot)프로젝트를 Docker로 구동해보기 앞선 포스팅에서 우리는 스프링부트 프로젝트를 jar로 만들었다. https://zunoxi.github.io/programming/2020/08/11/dev-web-spring_jar/ 포스트에 이어서 해당 jar파일을 이용해 도커 이미지를 만들어보자. 본문에 앞서, CI/CD pipeline이 잘 구축된 시스템에서는 jenkins상에서

☞ <https://zunoxi.tistory.com/69>



springboot 프로젝트 도커파일 실행 하는법

[Docker] - Spring Boot 애플리케이션을 Docker 이미지로 빌드후 실행해보기

저번장에 정리했던 Dockerfile에서 주로쓰이는 명령어 내용을 참고하여 Spring Boot 애플리케이션을 생성후 Docker 이미지로 빌드후 실행해보자 환경은 JAVA 11, Spring Boot 2.5.5 버전이고 Web 의존성만 추가하자 포트 번호를 8000으로하고 간단한 컨트롤러 클래스를 생성해보자. 근데 컨트롤러의 hello 메서드에는 docker-app 이라
:: <https://kim-jong-hyun.tistory.com/91>



백그라운드 실행...

nginx conf 파일 수정.

conf 먼저 수정하고 compose restart 하기

```
ubuntu@ip-172-26-7-221:~/NGINX_Settings$ sudo docker-compose restart
Restarting nginx_settings_certbot_1 ... done
Restarting nginx_settings_nginx_1 ... done
```

컨테이너 들어가서 conf 보니 정상적으로 바뀌어있었음

리로드 다시 해줌.

```
root@3624d7519894:~# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@3624d7519894:~# nginx -s reload
2022/09/24 14:24:52 [notice] 38#38: signal process started
root@3624d7519894:~#
```

▼ 컨테이너 시간 설정

처음에는 MySql 시간을 바꿔야하나, EC2 서버 시간을 바꿔야하나 이렇게 여러가지를 해보다가, 생각 해보니 Date 를 부르는 곳은 Spring 서버(Spring 컨테이너) 라는것을 인지했다.

Spring 서버의 시간을 바꾸려고

Docker run -d -p ... `-it -e TZ=Asia/Seoul` 로 추가를 하고 컨테이너상 date 를 찍어보니 안먹더라.

추후에 찾아보니 Dockerfile 에 설정한 Alpine 이라는 OS 는 너무 작아서 저 기능이 없다.

그래서 Dockerfile 에서 다운을 받아서 EC2서버의 시간방식을 copy 해서 설정을해주는 방식으로 진행 했다.

```
# Dockerfile
ENV TZ=Asia/Seoul

RUN apk --no-cache add tzdata && \
    cp /usr/share/zoneinfo/$TZ /etc/localtime && \
    echo $TZ > /etc/timezone \
    apk del tzdata
```

변경된 Spring 의 Dockerfile

```
FROM openjdk:8-jdk-alpine
ARG JAR_FILE=build/libs/ieng-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar

#time Setting for alpine
ENV TZ=Asia/Seoul
RUN apk --no-cache add tzdata && \
    cp /usr/share/zoneinfo/$TZ /etc/localtime && \
    echo $TZ > /etc/timezone \
    apk del tzdata

ENTRYPOINT ["java", "-jar", "/app.jar"]
```


[docker] alpine 리눅스에서 timezone 설정하기(docker container)

alpine 리눅스에서 timezone 설정하기(docker container) alpine 리눅스는 워낙 경량 이미지다보니 TZ 환경변수를 설정해주는 것만으로는 timezone이 정상적으로 변경되지 않는다. KST 로 timezone을 설정하는 방..

🔗 <https://young-cow.tistory.com/85>

[docker] alpine 리눅스에서 timezone 설정하기 (docker container)

▼ S3 버킷 생성

일반 구성

버킷 이름

ieng-bucket

버킷 이름은 전역에서 고유해야 하며 공백 또는 대문자를 포함할 수 없습니다. [버킷 이름 지정 규칙 참조](#)

AWS 리전

아시아 태평양(서울) ap-northeast-2 ▼

기존 버킷에서 설정 복사 - 선택 사항
다음 구성의 버킷 설정만 복사됩니다.

버킷 선택

객체 소유권 [Info](#)

다른 AWS 계정에서 이 버킷에 작성한 객체의 소유권 및 액세스 제어 목록(ACL)의 사용을 제어합니다. 객체 소유권은 객체에 대한 액세스를 지정할 수 있는 사용자를 결정합니다.

- ☐ ACL 비활성화됨(권장)
이 버킷의 모든 객체는 이 계정이 소유합니다. 이 버킷과 그 객체에 대한 액세스는 정책을 통해서만 지정됩니다.

- ☒ ACL 활성화됨
이 버킷의 객체는 다른 AWS 계정에서 소유할 수 있습니다. 이 버킷 및 객체에 대한 액세스는 ACL을 사용하여 지정할 수 있습니다.

객체 소유권

- ☒ 버킷 소유자 선호
이 버킷에 작성된 새 객체가 bucket-owner-full-control 삽입 ACL을 지정하는 경우 새 객체는 버킷 소유자가 소유합니다. 그렇지 않은 경우 객체 라이터가 소유합니다.
- ☐ 객체 라이터
객체 라이터는 객체 소유자로 유지됩니다.

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

☐ 모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

- ☐ 새 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하며, 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.
- ☐ 임의의 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.
- ☐ 새 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지점 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.
- ☐ 임의의 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 교차 계정 액세스 차단
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지점에 대한 퍼블릭 및 교차 계정 액세스를 무시합니다.



모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.

정적 웹 사이트 호스팅과 같은 구체적으로 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.

- ☒ 현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.

버킷 버전 관리

버전 관리는 객체의 여러 버전을 동일한 버킷에서 관리하기 위한 수단입니다. 버전 관리를 사용하여 Amazon S3 버킷에 저장된 모든 객체의 각 버전을 보존, 검색 및 복원할 수 있습니다. 버전 관리를 통해 의도치 않은 사용자 작업과 애플리케이션 장애를 모두 복구할 수 있습니다. [자세히 알아보기](#)

버킷 버전 관리

- ☒ 비활성화

태그 (0) - 선택 사항

버킷에 태그를 지정하여 스토리지 비용 또는 기타 기준을 추적합니다. [자세히 알아보기](#)

이 버킷과 연결된 태그가 없습니다.

태그 추가

기본 암호화

이 버킷에 저장된 새 객체를 자동으로 암호화합니다. [자세히 알아보기](#)

서버 측 암호화

- ☒ 비활성화
☐ 활성화

이름 ▲	AWS 리전 ▼	액세스 ▼	생성 날짜 ▼
ieng-bucket	아시아 태평양(서울) ap-northeast-2	객체를 퍼블릭으로 설정할 수 있음	2022. 9. 28. am 12:55:33 AM KST

ieng-bucket Info

객체 | 속성 | **권한** | 지표 | 관리 | 액세스 지점

권한 개요

액세스
객체를 퍼블릭으로 설정할 수 있음

퍼블릭 액세스 차단(버킷 설정)

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 모든 S3 버킷 및 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 [모든 퍼블릭 액세스 차단]을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 [모든 퍼블릭 액세스 차단]을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 버킷 또는 내부 객체에 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

편집

모든 퍼블릭 액세스 차단

비활성

▶ 이 버킷의 개별 퍼블릭 액세스 차단 설정

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

편집

삭제

버킷 정책 편집 Info

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. 자세히 알아보기 [↗](#)


정책 예제 [↗](#)

정책 생성기 [↗](#)

✓2

QV

버킷 ARN

 arn:aws:s3:::ieng-bucket

정책

1

문 편집

문 선택

정책에서 기존 문을 선택하거나 새 문을 추가합니다.

[+ 새 문 추가](#)

정책 생성기로 가서 생성 해준듯



AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to [Amazon Web Services \(AWS\)](#) products and resources. For more information about creating policies, see [key concepts in Using AWS Identity and Access Management](#). Here are [sample policies](#).

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), an [SNS Topic Policy](#), a [VPC Endpoint Policy](#), and an [SQS Queue Policy](#).

Select Type of Policy S3 Bucket Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect ☒ Allow ☐ Deny

Principal
Use a comma to separate multiple values.

AWS Service Amazon S3 ☐ All Services ('*')
Use multiple statements to add permissions for more than one service.

Actions -- Select Actions -- ☒ All Actions ('*') ?

Amazon Resource Name (ARN)
ARN should follow the following format: arn:aws:s3:::\${BucketName}/\${KeyName}.
Use a comma to separate multiple values.

[Add Conditions \(Optional\)](#)

이후 generate policy 한다.

버킷 정책 편집 [Info](#)

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

버킷 ARN

정책

```
1 {
2   "Id": "Policy1664294542262",
3   "Version": "2012-10-17",
4   "Statement": [
5     {
6       "Sid": "Stmt1664294520436",
7       "Action": "s3:*",
8       "Effect": "Allow",
9       "Resource": "arn:aws:s3:::ieng-bucket",
10      "Principal": "*"
11    }
12  ]
13 }
```

복붙해준다.

🔒 버킷 정책을 편집합니다.

Amazon S3 > 버킷 > ieng-bucket

ieng-bucket info

퍼블릭 액세스 가능

개별 | 속성 | **권한** | 지표 | 관리 | 액세스 지정

권한 개요

액세스

⚠️ **퍼블릭**

퍼블릭 액세스 차단(버킷 설정)

퍼블릭 액세스는 ACL, 액세스 제어 목록, 버킷 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 모든 S3 버킷 및 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 [퍼블릭 액세스 차단]을 활성화합니다. 이 설정은 이 버킷 및 해당 객체에만 적용됩니다. AWS에서는 [퍼블릭 액세스 차단]을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 버킷 또는 내부 객체에 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 해당 개별 설정을 사용자 지정할 수 있습니다. 자세히 알아보기

모든 퍼블릭 액세스 차단

⚠️ **비활성**

▶ 이 버킷의 개별 퍼블릭 액세스 차단 설정

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계층의 소유권 객체에는 적용되지 않습니다. 자세히 알아보기

```

{
  "Version": "2012-10-17",
  "Id": "Policy1664294542262",
  "Statement": [
    {
      "Sid": "Stmt1664294520436",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3::ieng-bucket"
    }
  ]
}

```

Minicrowd 정품 인증

업로드 Info

S3에 업로드할 파일 및 폴더를 추가합니다. 160GB보다 큰 파일을 업로드하려면 AWS CLI, AWS SDK 또는 Amazon S3 REST API를 사용합니다. 자세히 알아보기

여기에 업로드할 파일과 폴더를 끌어서 놓거나, [파일 추가] 또는 [폴더 추가]를 선택합니다.

파일 및 폴더 (1 합계, 30.4KB)

이 테이블의 모든 파일과 폴더가 업로드됩니다.

< 1 >

<input type="checkbox"/>	이름	폴더	유형	크기
<input type="checkbox"/>	profile_default.png	-	image/png	30.4KB

대상

대상

s3://ieng-bucket

▶ **대상 세부 정보**

지정된 대상에 저장된 새 객체에 영향을 미치는 버킷 설정.

▶ **권한**

다른 AWS 계정에 퍼블릭 액세스 및 액세스 권한을 부여합니다.

▶ **속성**

스토리지 클래스, 암호화 설정, 태그 등을 지정합니다.

취소
업로드

이미지 업로드 해보기(profile_default.png)

업로드 성공

아래에서 세부 정보를 확인합니다.

업로드: 상태

닫기

이 페이지에서 나가면 아래의 정보를 더 이상 확인할 수 없습니다.

요약

대상

s3://ieng-bucket

성공

1개 파일, 30.4KB (100.00%)

실패

0개 파일, 0B (0%)

파일 및 폴더

구성

파일 및 폴더 (1 합계, 30.4KB)

이름으로 찾기

< 1 >

이름	폴더	유형	크기	상태	오류
profile_default.png	-	image/png	30.4KB	성공	-

Amazon S3 > 버킷 > ieng-bucket > profile_default.png

profile_default.png

Info

S3 URI 복사

다운로드

열기

객체 작업

속성

권한

버전

객체 개요

소유자

3a5b0001151244959ff2c6b6b81b8dd0a125696e357987c2d99658fc48f494dd

AWS 리전

아시아 태평양(서울) ap-northeast-2

마지막 수정

2022. 9. 28. am 1:11:44 AM KST

크기

30.4KB

유형

png

키

profile_default.png

S3 URI

s3://ieng-bucket/profile_default.png

Amazon 리소스 이름(ARN)

arn:aws:s3:::ieng-bucket/profile_default.png

엔티티 태그(Etag)

35e5e6c4218f76fa4d3bb2110e6cab54

객체 URL

https://ieng-bucket.s3.ap-northeast-2.amazonaws.com/profile_default.png

이미지 이름 클릭시 나오는 정보

https://ieng-bucket.s3.ap-northeast-2.amazonaws.com/profile_default.png

S3의 퍼블릭 액세스 권한을 설정합니다.

※ 외부에 S3을 공개할 경우 모든 퍼블릭 액세스 차단을 체크 해제하고, 공개하지 않는다면 체크를 해주시면 됩니다.

퍼블릭 액세스를 차단할 경우, IAM에서 AWSSecretKey를 발급받고 키를 이용해서 S3 객체에 접근할 수 있습니다.

저는 외부에 S3을 공개할 예정이므로, 체크를 해제합니다.

(체크하고 정책 설정하느라 삼질을 좀 했습니다 🤔🤔)

버킷 권한 풀고 이미지 권한도 풀어주니 https:// 로 그냥 접근이 되긴 하는데 그렇게 하면 안될거 같음. IAM 통해서 하는게 안전하다고 생각함.

IAM

iam 도 설정함

▼ S3 with springboot

api/members/sign-up/default

회원가입 시 requestDto 로 파일을 받아야 함

일단 하기전에 RequestPart로 받도록 설정 했음(request 를 json 이 아니라 form/data로 받아야해서)

```

@PostMapping("/sign-up")
public ResponseEntity<?> createMember(@RequestPart("data") MemberRequestDto memberRequestDto){
    try{
        logger.debug("api/sign-up");
        String email = memberRequestDto.getEmail();

        String accessToken = jwtService.createAccessToken(email);
        String refreshToken = jwtService.createRefreshToken();

        HttpHeaders headers = loginService.createTokenHeader(accessToken, refreshToken);

        MemberResponseDto memberResponseDto = memberService.createMember(memberRequestDto, refreshToken);
        return ResponseEntity.status(HttpStatus.CREATED).headers(headers).body(CommonResponse.createSuccess( message: "회원
    })
    catch(DuplicateNicknameException e){
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body(CommonResponse.createError("닉네임 중복 회원 가입 불가."))
    }
}

```

postman 이 계속 안됐는데 contentType 을 넣어줬어야했다.

Resolved [org.springframework.web.HttpMediaTypeNotSupportedException: Content type 'application/octet-stream' not supported]

KEY	VALUE	CONTENT TYPE	DESCR
<input checked="" type="checkbox"/> data	{ "provider": "leng", "email": "inseinsein@adm	application/json	
<input checked="" type="checkbox"/> imgfile	~~sdafjsdafjzildxcjfilz	multipart/form-data	
<input type="checkbox"/> email	go@dfij.com	Auto	
<input type="checkbox"/> password	12345	Auto	
<input type="checkbox"/> nickname	ggttasawe	Auto	
<input type="checkbox"/> memberYMD	1993-11-16	Auto	
Key	Value	Auto	Descrip
Response			

[local] leng / [member] local 회원가입(form-data + S3)

POST http://localhost:8080/api/members/sign-up

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION
data	{"provider": "leng", "username": "inseinsein@adminaa.com", "p	
profile_image	testimage.jpg X	
email	go@dfj.com	

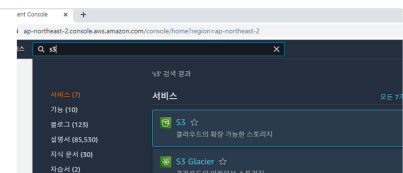
하니까 되더라(application.properties 에 내가 ; 찍어서 안된듯)

이름이 이상하게 들어가던데..

[AWS] 이미지 저장을 위한 S3 버킷 생성하기

안녕하세요. J4J입니다. 이번 포스팅은 이미지 저장을 위한 S3 버킷 생성하는 방법에 대해 적어보는 시간을 가져 보려고 합니다. [1. S3 서비스 접근] [2. 버킷 만들기 클릭] [3. 버킷 생성 정보 입력 - 일반 구성] 일반 구성에서 는 버킷 이름과 AWS 리전을 선택해주면 됩니다.

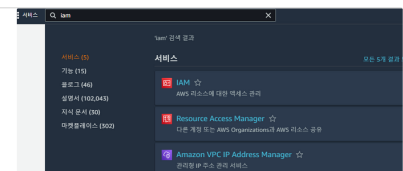
🔗 <https://forj.tistory.com/260>



[SpringBoot] AWS S3에 파일 업로드하기

안녕하세요. J4J입니다. 이번 포스팅은 AWS S3에 파일 업로드하는 방법에 대해 적어보는 시간을 가져보려고 합니다. 들어가기에 앞서 Controller에서 파일 데이터를 받을 수 있기 위해 multipartfile을 사용할 예정입니다. 또한 AWS S3 버킷 생성하는 방법은 [AWS] 이미지 저장을 위한 S3 버킷 생성하기 를 참고해주시길 바랍니다. 가장 먼

🔗 <https://forj.tistory.com/261>



IAM 과 S3

[Spring] Spring Boot AWS S3 사진 업로드 하는 법

이번 글에서는 Spring으로 AWS S3에 파일 업로드 하는 글을 정리해보겠습니다.(기존에 썼던 글은 잘못되거나 부족한 점이 많아서 이번 기회에 다시 써서 정리해보려 합니다.) 먼저 S3 버킷을 생성합니다. 버킷 이름만 설정한 후에 나머지 설정은 Default 설정 그대로 두고 생성하겠습니다. 먼저 IAM 사용자를 생성하겠습니다. 그리고 위와 같

🔗 <https://devlog-wjdrbs96.tistory.com/323>



s3사이트 여는법

Springboot로 S3 파일 업로드하기

이번 포스팅은 스프링에서 AWS S3 파일 업로드하는 방법입니다. 주로 이미지 파일을 올릴 때 많이 사용되곤 합니다. 1. 의존성 추가하기 build.gradle aws-spring-cloud-aws Spring-Cloud-AWS 의존성을 추가합니다. 2. S3 업로드 환경변수 설정 EC2에서 Spring Cloud 프로젝트를 실행시키면 기본으로 CloudFormation 구성을 시작

🔗 <https://www.sunny-son.space/spring/Springboot%EB%A1%9C%20S3%20%ED%8C%8C%EC%9D%BC%20%EC%97%85%EB%A1%9C%EB%93%9C/>



S3 업로드 방법

[Spring boot - 에러 해결] org.springframework.web.HttpMediaTypeNotSupportedException: Content type 'application/octet-stream' not supported

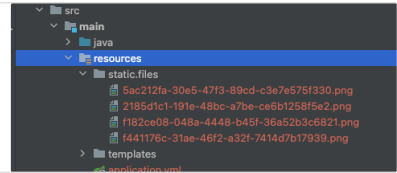
Resolved [org.springframework.web.HttpMediaTypeNotSupportedException: Content type 'application/octet-stream' not supported] public ResponseEntity saveContents(@RequestPart(required = false) AllContentsRequest contentsRequest) throws IOException { log.info("Contents controller: api/contents/new -----"); contentsRequestLinkedList.addAll(contentsRequest.getContents()); Book book = bookService.saveBook(contentsRequest.getIsbn());

🔗 <https://chea-young.tistory.com/33>

[AWS] Springboot에 AWS S3 연동 (이미지, 동영상 업로드)

따라 하시지만 해도 로컬에서 이미지, 동영상 파일 업로드가 가능하고, EC2에 배포한 환경에서도 파일 업로드가 가능합니다. 코드는 Github 에 있고, 함께 보시면 더 이해하기 쉬우실 것 같습니다. 1. AWS S3 버킷 설정 2. AWS IAM User 생성 4. 이미지, 동영상 업로드 결과 확인 5. 배포 환경에서 업로드 확인 S3 버킷을 생성할 때 원하는 이

🔗 <https://develop-writing.tistory.com/128>



▼ Redis on ubuntu

레디스 이미지 가져오기

```
ubuntu@ip-172-26-7-221:~$ sudo docker image pull redis
```

```
ubuntu@ip-172-26-7-221:~$ sudo docker network create redis-net
```

```
ubuntu@ip-172-26-7-221:~$ sudo docker run --name redisnet -p 6379:6379 --network redis-net -d redis redis-server --appendonly yes
```

```
ubuntu@ip-172-26-7-221:~$ sudo docker run -it --network redis-net --rm redis redis-cli -h redisnet
redisnet:6379>
```

성공

[성공] Ubuntu 20.04 : Redis 설치하는 방법, 예제, 명령어

Redis 패키지는 기본 Ubuntu 20.04 저장소에 포함되어 있습니다. 설치하는 매우 간단합니다. 아래 단계를 따르십시오. SSH 터미널에서 다음 명령을 실행하여 apt 패키지 목록을 업데이트하는 것으로 시작하십시오. sudo apt update sudo apt install redis-server 설치가 완료되면 Redis 서비스가 자동으로 시작됩니다. 서비스 상태를 확

🔗 <https://cloud-oky.tistory.com/395>

```
Unpacking lua-cjson:amd64 (2.1.0dfsg-2.1) ...
Selecting previously unselected package redis-tools.
Preparing to unpack .../6-redis-tools_5.0.14-0ubuntu1.1.ddeb ...
Unpacking redis-tools (5.0.14-0ubuntu1.1) ...
Selecting previously unselected package redis-server.
Preparing to unpack .../7-redis-server_5.0.14-0ubuntu1.1.ddeb ...
Unpacking redis-server (5.0.14-0ubuntu1.1) ...
Setting up lua-cjson:amd64 (2.1.0dfsg-2.1) ...
Setting up liblua5.3-0:amd64 (5.3.6-2ubuntu1) ...
Setting up lua-bitop:amd64 (1.0.2-5) ...
Setting up liblua5.3-dev:amd64 (5.3.6-2ubuntu1) ...
Setting up liblualib5:amd64 (5.3.6-2ubuntu1) ...
Setting up redis-tools (5.0.14-0ubuntu1.1) ...
Created symlink /etc/systemd/system/redis.service → /lib/systemd/system/redis-server.service.
Created symlink /etc/systemd/system/multi-user.target.wants/redis-server.service → /lib/systemd/system/redis-server.service.
Processing triggers for systemd (245.4-4ubuntu3.6) ...
```

[OS/UBUNTU] Redis 설치하기 (Docker 이미지)

1. CTRL + ALT + T 키를 눌러서 [터미널]을 실행한다. 2. Redis 컨테이너 설치에 앞서 컨테이너와 호스트간 볼륨 매핑을 위한 디렉토리를 생성하기 위해 [터미널]에서 아래 명령을 실행한다. sudo mkdir -p /data/redis 3. Redis 도커 이미지를 다운로드하기 위해 [터미널]에서 아래 명령을 실행한다. sudo docker image pull redis 4.

🔗 <https://icodetalker.tistory.com/9067>



[Redis] 도커(Docker)로 Redis 설치하기 | 블로그 | 딩그르

Django에서 Cache Backend로 사용하고 Celery의 메세징 큐로 사용하기 위한 Redis를 설치해 보겠습니다. 기본 워크플로우는 Celerybeat가 또는 직접 작업지시를 했을 경우, 작업을 Redis 큐에 넣고 Celery worker 가 Django 안에 정의된 대로 일을 수행하는 플로우 입니다. 아.. 없었으면 큰일이었을꺼예요. 따로 뒤에서 데몬을 계속 돌려야

🔗 <https://dingrr.com/blog/post/redis-%EB%8F%84%EC%BB%A4docker%EB%A1%9C-redis-%EC%84%A4%EC%B9%98%ED%95%98%EA%B8%B0>

with redis-cli



▼ react on docker

구조

```
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/frontend$ ls
Dockerfile default.conf docker-compose.yml package-lock.json package.json public src
```

default.conf 는 docker 내부에 깔릴 nginx 의 conf 파일.

파일

```
#Dockerfile

# nginx가 제공할 해줄 빌드 파일들을 생성하는 코드
# node 12 version, build file
FROM node:12 as builder
#working dir as /app
WORKDIR /app
# copy package.json/ package.lock.json
COPY ./package*.json ./
# cmd: npm install
RUN npm install
#copy local file
COPY . .
# build
RUN npm run build

FROM nginx
EXPOSE 3001
COPY ./default.conf /etc/nginx/conf.d/default.conf # 로컬에 있는 default.conf 파일을 도커 /etc/nginx/conf.d/default.conf
로 복사
COPY --from=builder /app/build /usr/share/nginx/html # 위에서 생성한 build 파일을 /usr/share/nginx/html로 복사
```

Dockerfile

Dockerfile 은 주석을 코드 옆에 쓰면 안되나봐

```
# docker-compose.yml
version: "3"
services:
  frontend: # 이름은 어떤걸로 지정해도 상관없음
    build:
      dockerfile: Dockerfile # dockerfile이름
      context: ./ # 도커 파일 위치 명시
    volumes:
      - /app/node_modules #도커 /app/node_modules는 맵핑을 따로 안해주겠다.
      - ./:/app # 로컬에 있는 모든 파일을 맵핑
    ports:
      - "3001:3001" # port 맵핑
    stdin_open: true
```

docker-compose.yml

```
#nginx.conf inside react docker
server {
    listen 3001; # 3001 port for react

    location / {

        root /usr/share/nginx/html; # HTML파일이 위치할 경로 설정 (위에 docker 파일을 참조하면 됩니다)

        index index.html index.htm; # 사이트의 index페이지로 설정할 파일명 설정

        try_files $uri $uri/ /index.html; # 리액트에서 페이지 라우팅을 제대로 하기 위해 적어줘야 하는 코드
    }
}
```

default.conf(폴더안에 넣고싶었는데 Dockerfile 빌드가 그렇게 하니 못찾더라..)

빌드 방법

```
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/frontend$ sudo docker-compose up -d
Creating network "frontend_default" with the default driver
Building frontend
```

docker-compose 를 통해 빌드(컨테이너 런까지 다되네)

```
sudo docker-compose up -d
```

빌드 하니 한참 걸린다 (step 이 10개나 있어서)

```
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/frontend$ sudo docker-compose up -d
Creating network "frontend_default" with the default driver
Building frontend
Step 1/10 : FROM node:12 as builder
12: Pulling from library/node
f5196cdf2518: Pull complete
9bed1e86f01e: Pull complete
f44e4bdb3a6c: Pull complete
2f75d131f406: Pull complete
07dff4ad21eb: Pull complete
e0ac4f13b766: Pull complete
df2c3b2eb7cc: Pull complete
efe636eac583: Pull complete
fe17849545bb: Pull complete
Digest: sha256:01627afeb110b3054ba4a1405541ca095c8bfca1cb6f2be9479c767a2711879e
Status: Downloaded newer image for node:12
--> 6c8de432fc7f
Step 2/10 : WORKDIR /app
--> Running in 9667756dbd97
Removing intermediate container 9667756dbd97
--> 022c71f6c4e8
Step 3/10 : COPY ./package*.json ./
--> 2cda30aa396d
Step 4/10 : RUN npm install
--> Running in 7e628e01c7bd
npm WARN read-shrinkwrap This version of npm is compatible with lockfileVersion@1, but package-lock.json was generated
or lockfileVersion@2. I'll try to do my best with it!

> core-js@3.25.2 postinstall /app/node_modules/core-js
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js:
> https://opencollective.com/core-js
> https://patreon.com/zloirock
> bitcoin: bc1qlea7544qtsmj2rayg0lthvza9fau63ux0fstcz

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)

> core-js-pure@3.25.2 postinstall /app/node_modules/core-js-pure
> node -e "try{require('./postinstall')}catch(e){}"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!

The project needs your help! Please consider supporting of core-js:
> https://opencollective.com/core-js
> https://patreon.com/zloirock
> bitcoin: bc1qlea7544qtsmj2rayg0lthvza9fau63ux0fstcz

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)
```

아래도 더있더라..

성공적으로 image 뒀다.

```
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/frontend$ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
eb6d1700769f   frontend_frontend                    "/docker-entrypoint..." 6 seconds ago   Up 5 seconds   80/tcp, 0.0.0.0:3001->3001/tcp, :::3001->3001/tcp   frontend_frontend_1
d2baa75822b3   springeng                            "java -jar /app.jar"      13 hours ago   Up 13 hours    0.0.0.0:8080->8080/tcp, :::8080->8080/tcp           spring
9e77fe214a3    redis                               "docker-entrypoint.s..." 39 hours ago   Up 39 hours    0.0.0.0:6379->6379/tcp, :::6379->6379/tcp           redisnet
7f9fad8f132    mysql:latest                         "docker-entrypoint.s..." 2 weeks ago    Up 7 days      0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql-container
```

<http://j7d209.p.ssafy.io:3001> 은 들어가짐

<https://j7d209.p.ssafy.io>

nginx reverse proxy 해주니 잘들어가짐.

docker를 이용한 react 앱 배포(무중단배포)

오늘은 docker를 이용하여 AWS EC2에 react앱을 배포해 보겠습니다. nginx 또한 사용할 예정입니다.(AWS EC2에 관해서는 따로 다루지 않을 예정) 시작하기전 docker의 간단한 명령어와 docker가 설치가 되어 있어야 합니다.(docker-compose 포함) 터미널창에 `npx create-react-app ./` 커맨드를 입력한 후 react 앱을 실행시키는 방법 <https://velog.io/@wmc1415/docker%EB%A5%BC-%EC%9D%B4%EC%9A%A9%ED%95%9C-react-%EC%95%B1-%EB%B0%B0%ED%8F%AC>



Can't find App.js and were to reload
Learn More

1

<https://velog.io/@bsjp400/Docker-Working-Directory%EB%A5%BC-%EB%AA%85%EC%8B%9C%ED%95%B4%EC%A4%98%EC%95%BC-%EB%90%98%EB%8A%94-%EC%9D%B4%EC%9C%A0>

▼ django on docker

8081 포트 사용할 예정.

장고 버전?

파이썬 버전?

```
# ./Dockerfile for Django
FROM python:3.6

#RUN apt-get update \
# && apt-get install -y --no-install-recommends \
# postgresql-client \
# && rm -rf /var/lib/apt/lists/*
# 나의 Django 코드를 컨테이너에 복사합니다 .
COPY . /app
# requirements.txt에 적혀 있는 pip 패키지들을 설치합니다 .
RUN pip install -r /app/requirements.txt
# start 파일을 실행 가능하게 합니다 .
RUN chmod 755 /app/start
# 워킹 디렉토리를 /app으로 합니다 .
WORKDIR /app
# 8000번 포트를 expose합니다 .
EXPOSE 8081
# /app/start 파일을 실행시킵니다 .
ENTRYPOINT ["/app/start"]
```

Dockerfile

```
$ vi Dockerfile
$ sudo docker build -t dockerdjango .
```

이미지 빌드 커맨드

```
sudo docker build -t dockerdjango .
```

이미지 빌드

```
ERROR: Could not find a version that satisfies the requirement asgiref==3.5.1 (from versions: 0.8, 0.9, 0.9.1, 0.10.0, 0.11.0, 0.11.1, 0.11.2, 0.12.0, 0.12.1, 0.13.0, 0.13.2, 0.13.3, 0.14.0, 1.0.0, 1.0.1, 1.1.0, 1.1.1, 1.1.2, 2.0.0, 2.0.1, 2.1.0, 2.1.1, 2.1.2, 2.1.3, 2.1.4, 2.1.5, 2.1.6, 2.2.0, 2.3.0, 2.3.1, 2.3.2, 3.0.0, 3.1.0, 3.1.1, 3.1.2, 3.1.3, 3.1.4, 3.2.0, 3.2.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.2.6, 3.2.7, 3.2.8, 3.2.9, 3.2.10, 3.3.0, 3.3.1, 3.3.2, 3.3.3, 3.3.4, 3.4.0, 3.4.1)
ERROR: No matching distribution found for asgiref==3.5.1
WARNING: You are using pip version 21.2.4; however, version 21.3.1 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
```

빌드중 오류

```
RUN pip install -r /app/requirements.txt 에서 오류
```

asgiref 3.5.1 을 찾아보니 python dependency 가 3.7 이상이라서 그렇게 했음.

requirements.txt 내 conflict (민철 얘기 하기.)

certifi 두개있고

charset-normalizer 두개있고

idna 두개있고

requests 두개 있고

urllib 두개 있고

3.9 로 파이썬 버전 바꾸고

requirements2.txt 로 install 하게끔 한다음에

```
Stored in directory: /root/.cache/pip/wheels/8e/70/28/3d6ccdb6e315f65f245da085402a2e1c7d14b90b30f239e2cf4
Successfully built clarifai future
Installing collected packages: pytz, jsonschema, urllib3, tzdata, sqlparse, six, protobuf, Pillow, jmespath, idna, future, configparser, charset-normalizer, chardet, certifi, backports.zoneinfo, asgiref, requests, python-dateutil, mysql-connector-python, grpcio, googleapis-common-protos, Django, django-rest-framework, django-storages, django-extensions, clarifai-grpc, clarifai, botocore, s3transfer, boto3
Successfully installed Django-4.1.1 Pillow-9.2.0 asgiref-3.5.2 backports.zoneinfo-0.2.1 boto3-1.24.82 botocore-1.27.82 certifi-2022.6.15 chardet-4.0.0 charset-normalizer-2.1.1 clarifai-2.6.2 clarifai-grpc-8.8.0 configparser-3.8.1 django-extensions-3.2.1 django-storages-1.13.1 django-rest-framework-3.14.0 future-0.18.2 googleapis-common-protos-1.53.0 grpcio-1.44.0 idna-2.10 jmespath-1.0.1 jsonschema-2.6.0 mysql-connector-python-8.0.30 protobuf-3.19.3 python-dateutil-2.8.2 pytz-2022.2.1 requests-2.25.1 s3transfer-0.6.0 six-1.16.0 sqlparse-0.4.3 tzdata-2022.4 urllib3-1.26.12
```

install requirements은 된듯

```
common-protos-1.53.0 grpcio-1.44.0 idna-2.10 jmespath-1.0.1 jsonschema-2.6.0 mysql-connector-python-8.0.30 protobuf-3.19.3 python-dateutil-2.8.2 pytz-2022.2.1 requests-2.25.1 s3transfer-0.6.0 six-1.16.0 sqlparse-0.4.3 tzdata-2022.4 urllib3-1.26.12
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.2.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 5ab902877177
--> 1875f30b554b
Step 5/8 : RUN chmod 755 /app/start
--> Running in 23efcd42e87
chmod: cannot access '/app/start': No such file or directory
The command '/bin/sh -c chmod 755 /app/start' returned a non-zero code: 1
```

start 파일이 없대서 만들어줬음

```
openjdk_8-jdk-alpine_85562aa0b991_3_years_ago_105MB
ubuntu@ip-172-26-7-221:~/gitLab/S07P22D209/ai/ieng_ai$ sudo docker run -d -p 8081:8081 --name django dockerdjango
e50edd4f3458edd12159d77ab4f8dda8d182722832d039b99383c99c7e1395fd
```

이미지 run 커맨드

sudo docker run -d -p 8081:8081: --name {생길 컨테이너의 이름 명명} {이미지 이름}

```
sudo docker run -d -p 8081:8081 --name django dockerdjango
```

nginx 리버스 프록시 해줬고

장고는 올라갔는데 ai-api 가 안되더라

DisallowedHost at / 라고 뜨던데 settings.py 에서 '*' 로 수정 해줬음

▲ For development, you can use the * wildcard to allow all hosts in `settings.py`:

12 **ALLOWED_HOSTS = ['*']**

▼ Important

🕒 Modify this configuration when you deploy your app in production environment.

Share Improve this answer Follow

edited Aug 8, 2020 at 0:03

answered Jan 14, 2020 at 22:12



Rob Bednark

23.8k ● 20 ● 78 ● 117



kapoc

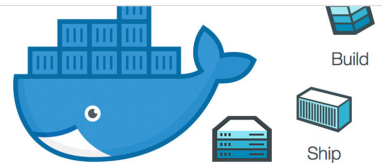
139 ● 1 ● 4

Add a comment

나만의 도커(Docker) 이미지를 만들어서 장고(Django) 서비스 배포하기

이번 장에서는 Django 공식 도커 이미지를 확인해보았고, Deprecated되었음을 확인했습니다. 이번 장에서는 나만의 Django 이미지를 직접 만들어서 배포해보고, docker hub에 업로드까지 해보겠습니다. 본 포스팅의 프로젝트 소스는 아래의 링크를 통해 얻을 수 있습니다. github.com/siner308/django-with-custom-image-docker

<https://blog.siner.io/2019/02/25/django-docker-custom-image/>



Django : Docker로 배포하기

팀프로젝트를 하면서 개발용 서버와 프론트와 통신하는 서버(EC2)를 분리하여 작업을 하였다. EC2의 프리티어 용량을 8기가로 적게만들기도 했고, 통신도 많이하다보니 쉽게 로그파일이 많이 쌓였다. 또 내가 실수로 쌓여져 있는 로그파일들을 관리하려다가 EC2를 잘못 건드리기도 하여 EC2 삭제와 종료를 수없이 반복하였다. EC2를 생

📄 <https://velog.io/@wind1992/Django-Docker%EB%A1%B0%B0%ED%8F%AC%ED%95%98%EA%B8%B0>



1-4. DB 접속 정보

- spring 컨테이너 에 `backend\src\main\resources\application.properties`