



신문물

삼성 SW 청년 아카데미 서울캠퍼스 6기
GitLab 소스 클론 이후 빌드 및 배포

A406팀 뉴스빅

임재현, 김하영, 배용한, 손수연, 유민상, 허은아

목차

1. 프로젝트 기술 스택
2. 빌드 상세 내용
3. 배포
4. MySql 접속 정보

1. 프로젝트 기술 스택

가. 이슈 관리 : Jira

나. 형상 관리 : GitLab

다. 빌드/배포 관리 : Jenkins

라. 커뮤니케이션 : Mattermost, Webex, Notion

마. 개발 환경

1) 운영 체제 : Windows10

2) IDE

가) STS 3.9.14 RELEASE

나) Visual Studio Code 1.64.2

다) UI/UX : Figma

라) JetBrains PyCharm Community Edition 2018.2.3

3) 데이터베이스 : MySQL Workbench 8.0.27

4) 서버 : AWS EC2

가) Ubuntu 20.04.4 LTS

나) Docker 20.10.7

다) Nginx 1.18.0

5) 클러스터 서버 : AWS EC2

가) Ubuntu 20.04.4 LTS

나) Hadoop 3.3.1

바. 세부 사항

1) Baekend

가) Java(Open-JDK zulu 1.8)

나) Spring Boot 2.5.6

다) Gradle 7.4.2

라) Swagger2 2.9.2

마) JDBC

바) QueryDSL 1.0.10

사) Lombok 1.18.22

아) JPA 2.5.6

자) JWT 0.9.1

차) QLRM 1.7.1

카) Json Simple 1.1.1

타) FastAPI 0.75.0

파) uvicorn 0.17.6

하) KoNLPy 0.6.0

2) Frontend

가) Vue.js 3.0

나) Vuex 4.0.0

다) npm

라) Axios 0.26.1

마) Bootstrap 5.1.3

바) Chart.js 3.7.1

사) vue-d3-cloud 0.2.0

아) vue-router 4.0.13

자) vue-number-animation 1.1.2

2. 빌드 상세 내용

Jenkins 관리 페이지로 접속합니다. (<http://j6a406.p.ssafy.io:9090>)

관리 페이지에서 sinmunmul 아이템의 구성 탭으로 진입합니다.

General 소스 코드 관리 빌드 유발 빌드 환경 Build 빌드 후 조치

설명

[Plain text] [미리보기](#)

☒ GitHub project
Project url ?

[고급...](#)

☐ 사용자 빌드 경로 사용 ?

GitLab Connection

[고급...](#)

☐ Use alternative credential
☐ This build requires lockable resources
☐ Throttle builds ?
☐ 오래된 빌드 삭제 ?
☐ 이 빌드는 매개변수가 있습니다 ?
☐ 빌드 안함 ?
☐ 필요한 경우 concurrent 빌드 실행 ?
[고급...](#)

연동시킬 GitLab의 Repository URL을 입력하고 Credentials를 선택합니다.
master 브랜치와 연결합니다.

소스 코드 관리

☐ None
☒ Git ?

Repositories ?

Repository URL ?

Credentials ?
 [Add](#)

[고급...](#)

[Add Repository](#)

Branches to build ?

Branch Specifier (blank for 'any') ?

이후 빌드 유발 탭에서 GitLab Master 브랜치에 Push Event가 발생할 때, 빌드 유발이 되도록 선택합니다.



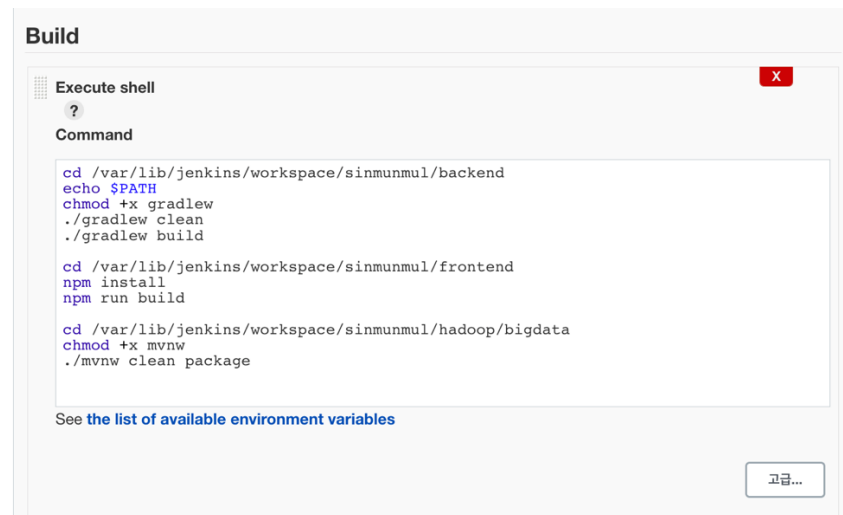
빌드 유발

- ☐ 빌드를 원격으로 유발 (예: 스크립트 사용) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ Build when a change is pushed to GitLab. GitLab webhook URL: <http://j6a406.p.ssafy.io:9090/project/sinmunmul> ?

Enabled GitLab triggers

- ☒ Push Events

다음은 빌드 과정입니다.



Build

Execute shell ?

Command

```
cd /var/lib/jenkins/workspace/sinmunmul/backend
echo $PATH
chmod +x gradlew
./gradlew clean
./gradlew build

cd /var/lib/jenkins/workspace/sinmunmul/frontend
npm install
npm run build

cd /var/lib/jenkins/workspace/sinmunmul/hadoop/bigdata
chmod +x mvnw
./mvnw clean package
```

See [the list of available environment variables](#)

고급...

가. SpringBoot 프로젝트 빌드

- 1) EC2 서버의 Jenkins가 받은 GitLab의 SpringBoot 프로젝트 위치로 이동합니다.
- 2) 새로운 환경에서 프로젝트를 설정할 때 java와 gradle을 설치하지 않고 바로 빌드할 수 있도록 gradlew를 사용하여 빌드합니다.
- 3) chmod +x gradle : 빌드 파일을 실행하기 위해 'chmod +x' 명령어로 파일 실행 권한을 줍니다.
- 4) ./gradlew clean : gradle의 캐시를 초기화합니다.
- 5) ./gradlew build : SpringBoot 프로젝트를 빌드하고 jar파일을 생성합니다.

나. Vue 프로젝트 빌드

- 1) EC2 서버의 Jenkins가 받은 GitLab의 Vue 프로젝트 위치로 이동합니다.
- 2) npm install : 필요한 node_modules를 설치합니다.
- 3) npm run build : Vue 프로젝트를 빌드합니다.

다. Hadoop MapReduce 빌드

- 1) EC2 서버의 Jenkins가 받은 GitLab의 MapReduce 프로젝트 위치로 이동합니다.
- 2) 별도로 Maven을 설치하지 않고 Maven을 이용할 수 있는 Mvnw 파일을 이용하여 빌드합니다.
- 3) chmod +x mvnw : 빌드 파일을 실행하기 위해 'chmod +x' 명령어로 파일 실행 권한을 줍니다.

- 4) ./mvnw clean package : Maven/target을 Clean 및 build합니다. 빌드를 성공적으로 마치면 ~/build/libs/sinmunmul-0.0.1-SNAPSHOT.jar가 생성됩니다.
- 5) 하둡 클러스터에서 하둡 프로젝트를 실행시키기 위해 jar파일을 서버에서 클러스터 서버로 전송하는 과정을 빌드 후 조치에서 수행합니다.

빌드 후 조치

Send build artifacts over SSH

SSH Publishers

SSH Server

Name ?

hadoopCluster

고급...

Transfers

Transfer Set

Source files ?

hadoop/bigdata/target/*.jar

Remove prefix ?

hadoop/bigdata/target/

Remote directory ?

/

Exec command ?

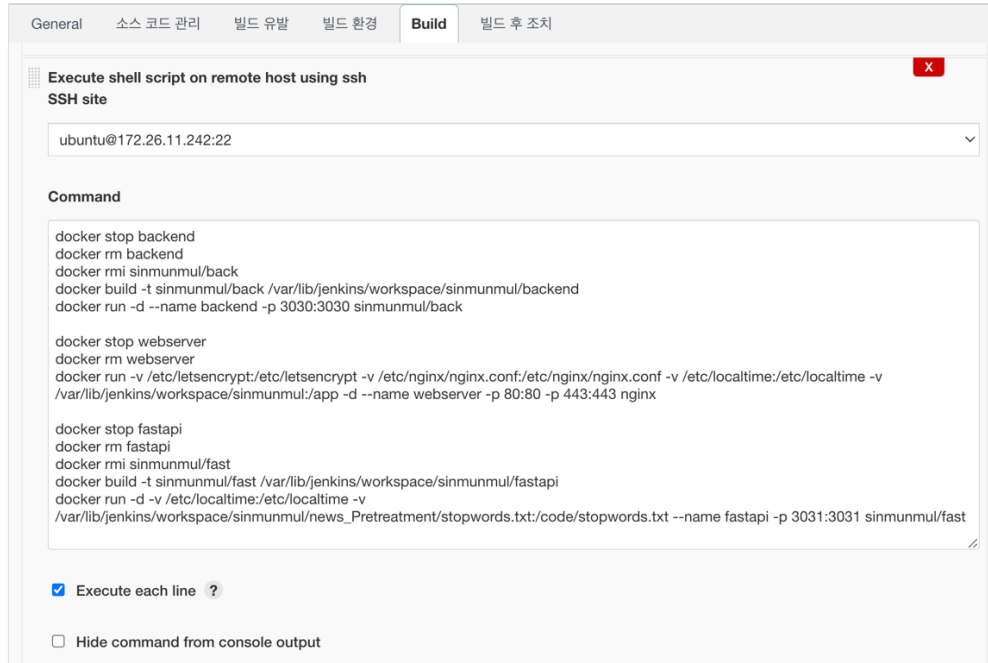
/home/j6a406/hadoop/test.sh

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

고급...

3. 배포

: remote host에 ssh를 통해 shell 명령으로 빌드와 배포를 진행하기 위해 Execute shell script on remote host using ssh 모듈을 사용합니다. 다양한 서버 환경에서도 배포될 수 있도록 Docker를 사용하여 배포하였습니다.



가. SpringBoot 프로젝트(백엔드) 배포

- 1) 현재 실행 중인 backend 도커 컨테이너를 중지시킵니다.
- 2) backend 도커 컨테이너를 삭제합니다.
- 3) backend 도커 이미지를 삭제합니다.
- 4) SpringBoot 프로젝트 위치에서 도커 이미지를 빌드합니다.

Dockfile

```
FROM openjdk:8-jdk-alpine

# SpringBoot 프로젝트 경로/build/~/jar
ARG JAR_FILE=build/libs/sinmunmul-0.0.1-SNAPSHOT.jar

# app.jar로 복사
COPY ${JAR_FILE} app.jar

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

- 5) 4)에서 빌드한 도커 이미지로 도커 컨테이너를 실행합니다. 서버 시간 동일한 도커 컨테이너 내의 타임존 설정을 위해 /etc/localtime 디렉토리를 도커 볼륨으로 설정하여 마운트하고 3030 포트를 할당합니다.

나. Nginx, Vue(프론트엔드) 배포

- 1) 현재 실행 중인 webserver 도커 컨테이너를 중지시킵니다.
- 2) webserver 도커 컨테이너를 삭제합니다.
- 3) Docker Hub에서 받은 Nginx 이미지로 도커 컨테이너를 실행합니다.

도커 볼륨 설정은 다음과 같습니다.

가) HTTPS 설정을 위한 SSL키가 위치한 디렉토리 /etc/letsencrypt

나) Reverse Proxy를 위한 Nginx 설정 파일이 위치한 디렉토리 /etc/nginx/nginx.conf

다) 타임존 설정을 위한 디렉토리 /etc/localtime

라) Vue 프로젝트 배포를 위한 Vue 프로젝트 /var/lib/Jenkins/workspace/sinmunmul

4) http 기본 포트 80과 https 기본 포트 443을 할당합니다.

다. FastAPI(백엔드) 배포

1) 현재 실행 중인 fastapi 도커 컨테이너를 중지시킵니다.

2) fastapi 도커 컨테이너를 삭제합니다.

3) fastapi 도커 이미지를 삭제합니다.

4) FastAPI 프로젝트 위치에서 도커 이미지를 빌드합니다

FastAPI에서 KoNLPy와 형태소 분석기 mecab를 사용하기 위해 ubuntu 이미지로 빌드합니다. ubuntu 환경에 Java와 Python을 설치하여 FastAPI를 실행시키기 위한 환경을 구성합니다. FastAPI 프로젝트와 패키지 관리를 위한 requirements.txt를 복사합니다. “pip install -r /code/requirements.txt” 명령어를 통해 FastAPI와 uvicorn을 설치합니다. 그런 다음 프로젝트 내에서 mecab을 설치하고 3031포트를 할당하여 FastAPI를 실행시킵니다.

```
Dockerfile 979 Bytes [Edit] [Web IDE] [Repl]

1 # KoNLPy는 자바가 필요해서 우분투 환경에 자바랑 파이썬 설치하기
2 FROM ubuntu:latest
3
4 ENV LANG=C.UTF-8
5 ENV DEBIAN_FRONTEND=noninteractive
6 RUN apt-get update && \
7     apt-get install -y --no-install-recommends tzdata g++ curl
8
9 # install java
10 RUN apt-get install -y openjdk-8-jdk
11 ENV JAVA_HOME="/usr/lib/jvm/java-1.8-openjdk"
12
13 # install python
14 RUN apt-get install -y python3-pip python3-dev
15 RUN cd /usr/local/bin && \
16     ln -s /usr/bin/python3 python && \
17     ln -s /usr/bin/pip3 pip && \
18     pip install --upgrade pip
19
20 # apt clean
21 RUN apt-get clean && \
22     rm -rf /var/lib/apt/lists/*
23
24 #
25 WORKDIR /code
26
27 #
28 COPY ./ /code
29 COPY ./requirements.txt /code/requirements.txt
30
31 #
32 RUN apt update
33 RUN apt-get install -y curl git
34 RUN pip install --no-cache-dir --upgrade -r /code/requirements.txt
35 RUN curl -L https://raw.githubusercontent.com/konlpy/konlpy/master/scripts/mecab.sh | bash
36
37 #
38 COPY ./app /code/app
39
40 CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "3031"]
```

5) 4)에서 생성한 이미지로 도커 컨테이너를 실행합니다.

도커 볼륨 설정은 다음과 같습니다.

가) 타임존 설정을 위한 디렉토리 /etc/localtime

나) 뉴스 데이터 전처리 시 사용할 불용어 파일 /stopwords.txt

reverse proxy를 위한 nginx.conf의 내용은 다음과 같습니다.

```
server {
    server_name j6a406.p.ssafy.io;

    location / {
        root /app/frontend/dist;
        index index.html index.htm;
        try_files $uri $uri/ /index.html?q=$uri=$args;
    }

    listen 443 ssl default_server; # managed by Certbot
    listen [::]:443 ssl default_server;

    ssl_certificate /etc/letsencrypt/live/j6a406.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/j6a406.p.ssafy.io/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

    location /api/ {
        proxy_pass http://j6a406.p.ssafy.io:3030;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Host $http_host;
        proxy_ssl_server_name on;
    }

    location /fapi/ {
        proxy_pass http://j6a406.p.ssafy.io:3031;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Host $http_host;
        proxy_ssl_server_name on;
    }
}

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

    server_name j6a406.p.ssafy.io;

    location / {
        return 301 https://$server_name$request_uri;
    }
}
```

- http 기본 포트인 80포트로 들어오는 경우 https의 기본 포트인 443포트로 리다이렉트 시킵니다.
- 443포트로 접속하면, /etc/letsencrypt에 위치한 SSL 키 값들을 통해 인증을 진행합니다.
- 인증절차를 통과하면 /로 접근했을 때, Vue 프로젝트가 빌드된 디렉터리에서 index.html을 Mapping합니다.
- 도메인주소/api로 접속 시 SpringBoot 프로젝트가 실행 중인 3030포트로 연결됩니다.
- 도메인주소/fapi로 접속 시 FastAPI 프로젝트가 실행 중인 3031포트로 연결됩니다.

4. MySQL DataBase 접속 정보

mysql -u root -p 명령어로 root로 mysql에 접속합니다. 이때, 초기 접속 시 설정한 비밀번호를 입력하여 DB에 접속합니다.

```
ubuntu@ip-172-26-11-242:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4693
Server version: 8.0.28-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> █
```