Porting Manual

태그

개발환경

✔️ Deploy Server

- Ubuntu 20.04 LTS
- Window Server 2019

✔️ Front-end

- Vue : 2.6.11

- vue/cli: 4.5.0

✔️ DB

- Mysql : 8.0.25
- HBase : 2.3.6

✔️ Back-end

- Python : 3.6.8
- Java : openjdk-1.8.0_262
- Fast API: 0.68.2

- Gunicorn

- npm: 6.14.13
- docker : 20.10.8
- docker-compose : 1.28.0-rc2

✔️ IDE

- Pycharm Community : 2020.3
- Visual Studio Code : 1.59.0
- Mysql Workbench : 8.0.25
- Anaconda 3, Miniconda

배포방법

▼ 배포 서버에 접속 및 AWS EC2 Port 세팅을 가정하고 진행

1) Window Server 2019 환경

2) Ubuntu 20.04 LTS 환경

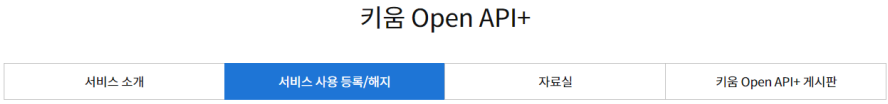
✔️ AWS Window Server 2019

Kiwoom API 설치

```
https://www.kiwoom.com/h/customer/download/VOpenApiInfoView   브라우저 접속


- 키움 Open API+ 모듈 다운로드                          설치
- KOA Studio 다운로드                                    설치


OpenAPI 폴더 내에 KOA Studio 번들 File 이동

- KOAStudioSA -> OpenAPI 폴더
```

Kiwoom API 등록

- API 사용에 사용자 인증이 필요합니다.

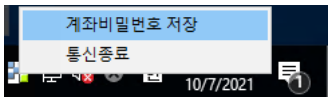


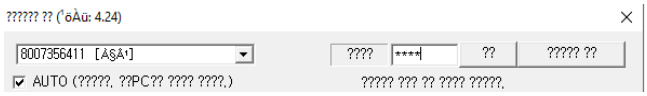
서비스 사용 등록/해지

```
체험 투자 계정 등록
```

KOAStudioSA 프로그램 실행 & 버전 업데이트 & 로그인

자동 로그인 설정

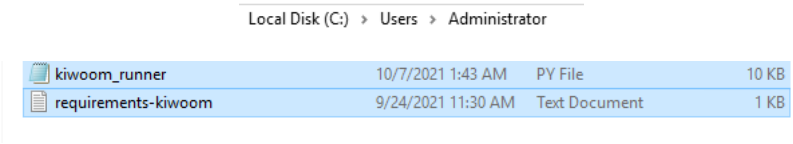





```
- 체험 투자 계정 기본 비밀번호 : 1234
- AUTO 체크 설정
- ?? 버튼 클릭 -> 자동 로그인 등록
```

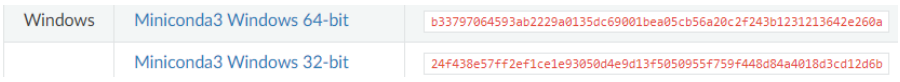
프로그램 File 준비

```
C://Users/Administrator 경로에 2개의 파일 준비
```



Miniconda 설치

```
https://docs.conda.io/en/latest/miniconda.html   브라우저 접속

- Miniconda3 Windows 32-bit                      설치
```



Miniconda 가상 환경 세팅

```
conda create --name py36_32 python=3.6.8

conda activate py36_32

pip install --upgrade pip

pip install -r requirements-kiwoom.txt
```

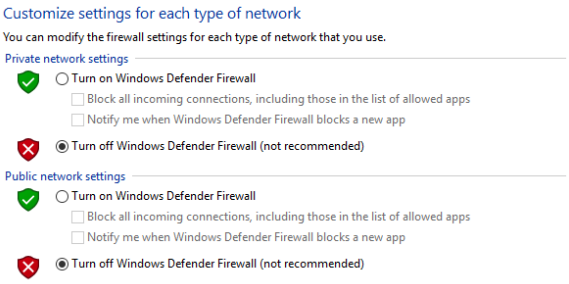
Kiwoom API 프로그램 구동

```
cd C:\Users\Administrator

python kiwoom_runner.py
```

방화벽 해제 (AWS 인바운드 규칙 설정도 필요)

- 외부 IP 접근을 허용하기 위해 방화벽을 해제합니다.

```
제어판 > System and Security > Windows Defender Firewall > Customize Settings > Turn off
```



- WebSocket을 통해 Fast API 서버와 통신하므로 상시 구동 시켜둡니다.

✔️ AWS Ubuntu 20.04 LTS

Java 8 버전 설치

1. Azul의 public key 추가

   ```
   sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 0x219BD9C9
   ```

2. Azul Repository 추가

   ```
   sudo apt-add-repository 'deb http://repos.azulsystems.com/ubuntu stable main'
   ```

3. apt-get 업데이트

   ```
   apt-get update
   ```

4. zulu-8 설치

   ```
   apt-get install zulu-8
   ```

5. 환경변수 설정

```
# /etc/profile

export JAVA_HOME=/usr/lib/jvm/zulu-8-amd64
```

6. 자바 설치확인

```
java -version
```

Docker-CE설치

1. apt 라이브러리 update 하기

```
sudo apt update
```

2. 다운로드를 위한 util 설치

```
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
```

3. Docker 공식 GPG 키 추가하기

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4. apt 리스트에 도커 다운경로 추가하기

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
```

5. 도커 다운로드

```
apt-cache policy docker-ce
```

6. 도커 설치하기

```
sudo apt install docker-ce

# apt 패키지 인덱스 업데이트하기
sudo apt-get update
```

7. sudo 없이 도커를 사용하기 위해 사용자를 docker 그룹에 등록

```
sudo usermod -aG docker [현재 사용자]

# 이를 반영하기 위한 시스템 재부팅
sudo reboot
```

8. docker-ce를 설치했다면 이제 docker-compose도 설치해주자

```
sudo curl -L https://github.com/docker/compose/releases/download/1.28.0-rc2/docker-compose-`uname -s`-`uname -m` -o /usr/local/bir
```

9. docker-compose도 실행권한을 주자 (다른 방식 연습!)

```
sudo chmod +x /usr/local/bin/docker-compose
```

Mysql 설치

1. Mysql 설치

```
sudo apt-get update
sudo apt-get install mysql-server
```

2. ubunut mysql은 대소문자 구별이 default로 설정되어 있음.
   - 이를 해결하려면 아래의 과정을 수행한다.

   1. Mysql 서비스 중지

   ```
   sudo service mysql stop
   ```

   2. Mysql 데이터 디렉토리 삭제

   ```
   sudo rm -rf /var/lib/mysql
   ```

   3. Mysql 데이터 디렉토리 재생성

   ```
   sudo mkdir /var/lib/mysql
   sudo chown mysql:mysql /var/lib/mysql
   sudo chmod 700 /var/lib/mysql
   ```

   4. lower_case_table_names = 1를 `[mysqld]` 섹션에 추가

   ```
   # /etc/mysql/mysql.conf.d/mysqld.conf
   [mysqld]
   ...

   lower_case_table_names = 1
   ```

   5. Mysql 초기화

   ```
   sudo mysqld --defaults-file=/etc/mysql/my.cnf --initialize --lower_case_table_names=1 --user=mysql --console
   ```

   6. Mysql 서비스 시작

   ```
   sudo service mysql start
   ```

   7. 새로 생성된 root의 비밀번호 검색

   ```
   sudo grep 'temporary password' /var/log/mysql/error.log
   ```

   8. root로 접속

   ```
   sudo mysql -u root -p
   ```

   9. 비밀번호 변경

   ```
   ALTER USER 'root'@'localhost' IDENTIFIED BY '1234';
   ```

   10. lower_case_table_names을 해서 1 출력 확인

```
SHOW VARIABLES LIKE 'lower_case_%';
```

3. AWS 설정에서 보안그룹 설정을 통해 mysql 기본 포트(3306)을 열어주어야 한다.
4. 외부접속 허용하기 ⇒ mysqld.cnf 수정

```
# /etc/mysql/mysql.conf.d/mysqld.cnf
sudo vi /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
# bind-address를 찾아서 0.0.0.0으로 수정
bind-address = 0.0.0.0
```

5. mysql 재시작 및 접속

```
sudo service mysql restart

sudo mysql -u root -p
```

6. 외부접속 권한을 가진 유저생성

```
CREATE USER 'root'@'%' IDENTIFIED BY '1234';

GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;

FLUSH PRIVILEGES;
```

- MYSQL 8.0부터 Grant 명령어를 사용하여 묵시적으로 사용자를 생성할 수 없다.
  따라서 '%' : 원격접속을 할 수 있는 사용자(여기서는 root)를 생성해 주어야한다.

소스코드 Git clone 받기

```
git clone https://lab.ssafy.com/s05-bigdata-dist/S05P21A602.git
```

Docker 이미지 빌드 & 실행

```
cd S05P21A602

docker-compose up -d
```

Docker

- docker-compose.yml

```
version: '3'
services:
        server:
                container_name: server
                build:
                        context: ./backend
                ports:
                        - "8080:8080"
        client:
                container_name: client
```

```
                build:
                        context: ./frontend
                depends_on:
                        - server
                ports:
                        - "80:80"
```

- frontend/Dockerfile

```
FROM node:lts-alpine as build-stage

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

RUN npm run build

FROM nginx:stable-alpine as production-stage
COPY ./.nginx/nginx.conf /etc/nginx/conf.d/default.conf
RUN rm -rf /usr/share/nginx/html/*

COPY --from=build-stage /app/dist /usr/share/nginx/html
EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

- backend/Dockerfile

```
FROM python:3.6.8

WORKDIR /code/root

RUN pip install --upgrade pip

COPY ./requirements.txt /code/requirements.txt

RUN pip install --no-cache-dir --upgrade -r /code/requirements.txt

RUN apt-get update

RUN apt-get install -y openjdk-8-jdk

COPY ./root /code/root

EXPOSE 8080

CMD ["gunicorn", "main:app", "-k", "uvicorn.workers.UvicornWorker", "--bind", "0.0.0.0:8080", "-w", "4"]
```

- .nginx/nginx.conf

```
server {
  listen 80;
  client_max_body_size 5M;
  location /api {
      proxy_set_header Host $host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_pass http://server:8080;
  }


  location / {
    alias /usr/share/nginx/html/;
    try_files $uri $uri/ /index.html;
  }
}
```

데이터베이스 접속정보

Mysql

- host: j5a602.p.ssafy.io
- port: 3306
- user: root
- password: 1234

접속 URL

: j5a602.p.ssafy.io