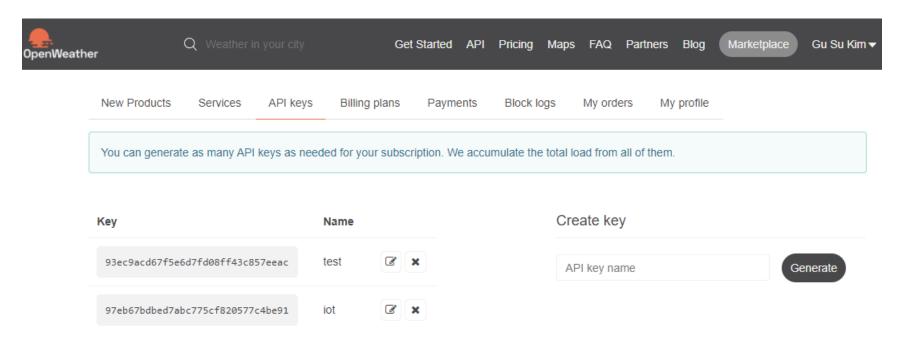
openweather

- o https://openweathermap.org/
- o 회원가입 -> 로그인
- o API키 발급
 - 회원 가입시 자동 발급



❖ api 호출

o http://api.openweathermap.org/data/2.5/weather?q={city}&APPID={API_KEY}&
 lang=kr

❖ 결과 json

```
"coord": { "lon": 126.9778, "lat": 37.5683 },
"weather": [
 { "id": 800, "main": "Clear", "description": "맑음", "icon": "01d" }
"base": "stations",
"main": {
 "temp": 272.82,
  "feels_like": 268.77,
  "temp min": 272.15,
  "temp_max": 273.15,
  "pressure": 1028,
  "humidity": 34
"visibility": 10000,
"wind": { "speed": 1.03, "deg": 120 },
"clouds": { "all": 0 },
"dt": 1612420804,
```

❖ 결과 json

```
"sys": {
    "type": 1,
    "id": 8105,
    "country": "KR",
    "sunrise": 1612391603,
    "sunset": 1612429114
},
    "timezone": 32400,
    "id": 1835848,
    "name": "Seoul",
    "cod": 200
}
```

requests 모듈

❖ 파이썬 requets 모듈

- o http 통신을 지원하는 모듈
 - Web openApi 호출에 사용
- o 설치
 - pip install requests
- ㅇ 사용법
 - res = requests.HTTP_메소드(URL, [headers=], [data=])
 - res = requests.get(url)
 - res = requests.post(url, data=obj_dict)
 - res = requests.put(url, data=obj_dict)
 - res = requests.delete(url)

requests 모듈

❖ Response 객체

o requests 요청의 리턴값

```
■ status_code 응답 상태 코드(200, 404, 501...)
```

■ headers 응답 헤더

■ cookies 쿠키 목록

■ encoding 응답 데이터(body) 인코딩 방식

■ text 텍스트 응답 데이터(html, txt ...)

■ content 바이너리 응답 데이터(images, audio, video…)

■ .json() json 문자열을 해석해서 dict 타입 리턴

GET 요청 보내기

```
import requests

url = "https://www.naver.com"
response = requests.get(url)
print("status code :", response.status_code)
print(response.text)
```

GET 요청 보내기

❖ 파일 다운로드

- o url에서 파일명 추출하기
 - file_name = url.split('/')[-1]

JSON

❖ JSON

- o JavaScript Object Notation
- ㅇ 문자열로 정보의 구조화를 표현
 - 자바스크립트 객체의 리터럴 표기방식과 동일
- ㅇ 파이썬에서는 사전과 유사
- o 규칙
 - 키이름은 반드시 큰 따움표로 표기
 - 값의 표기
 - 문자열: 큰 따움표로 표기
 - 숫자: 정수/실수 그대로 표기
 - 부울린: true/false
 - 배열(리스트): []로 표기

JSON

❖ 파이썬 json 모듈

- ㅇ 파이썬의 표준 모듈
 - import json
- o json 해석 처리
 - dict 타입 ←→ json 문자열 상호 변환
- o dict 타입 객체 → json 문자열 변환
 - dumps(dict_obj) : json 문자열 리턴
 - dump(file, dict_obj) : json 문자열을 file에 저장
- o json 문자열 → dict 타입 객체 변환
 - loads(json_str) : json 문자열을 분석하여 dict 타입 객체 반환
 - load(file): json file을 분석하여 dict 타입 객체 반환

weather.py

```
import requests as req
import json
API KEY = '93ec9acd67f5e6d7fd08ff43c857eeac'
def get weather(city='Seoul'):
    URL = f'http://api.openweathermap.org/data/2.5/weather?q={city}&APPID={API_KEY}&lang=kr'
    print(URL)
    weather = {}
    res = req.get(URL)
    if res.status code == 200:
        result = res.json()
        weather['main'] = result['weather'][0]['main']
        weather['description'] = result['weather'][0]['description']
        print(result['weather'][0]['description'])
        icon = result['weather'][0]['icon']
        weather['icon'] = f'http://openweathermap.org/img/w/{icon}.png'
        weather['etc'] = result['main']
    else:
        print('error', res.status_code)
    return weather
```

weather.py

```
weather = get_weather()
print(json.dumps(weather, indent=4, ensure_ascii=False))
```