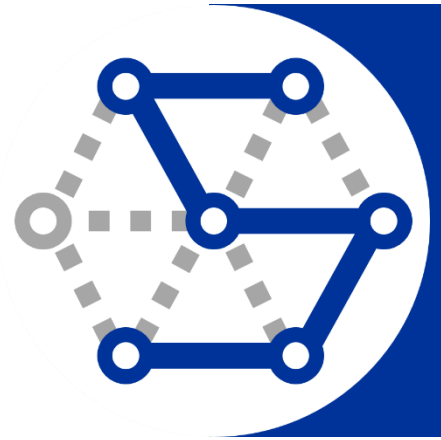

# 데이터 처리 프로그래밍

# Data Processing Programming

# 11
# 오픈 API & 웹 스크래핑

# 목차



# 1. 오픈 API

# Open API 이용 신청

**https://developers.naver.com/products/search/**

NAVER Developers  Products  Documents  Application  NAVER D2  Support  API 상태  Search Here

- API 이용 안내
- Clova
- 네이버 아이디로 로그인
- 지도
- 파파고
- 서비스 API
  - 데이터랩
  - 검색
  - 단축URL
  - 캡차
  - 네이버 공유하기
  - 모바일앱 연동
  - 네이버 오픈메인

## 네이버 검색 결과 컨텐츠

웹 서비스 또는 모바일 앱에 네이버 웹문서/블로그/뉴스/책/영화/카페글/지식iN/쇼핑/이미지/백과사전/전문자료 분야에 대한 검색 결과를 보여 줄 수 있습니다.

## 지역 검색

'OO역 맛집', 'OO동 술집'과 같은 검색 결과를 보여주고 싶을 때 사용하며, 네이버 지역 서비스에 등록된 각 지역별 업체 및 상호 검색결과를 보여줍니다.

## 검색 부가 기능

검색 부가 기능으로 특정 검색어에 대해 성인검색어 여부를 알려주는 기능과 검색창에 입력된 오타를 바로 잡아주는 오타변환 기능을 제공합니다.

* 처리한도 : 25,000/일

오픈 API 이용 신청  개발 가이드 보기

# 애플리케이션 등록

## 애플리케이션 등록 (API 이용신청)

애플리케이션의 기본 정보를 등록하면, 좌측 내 애플리케이션 메뉴의 서브 메뉴에 등록하신 애플리케이션 이름으로 서브 메뉴가 만들어집니다.

애플리케이션 이름

naverblog

- 네이버 아이디로 로그인할 때 사용자에게 표시되는 이름이므로 가급적 10자 이내의 간결한 이름을 사용해주세요.
- 40자 이내의 영문, 한글, 숫자, 공백문자, "-", "_"만 입력 가능합니다.

사용 API

선택하세요.

검색

비로그인 오픈 API 서비스 환경

환경 추가

Android 설정

안드로이드 앱 패키지 이름

com.example.naverblog

안드로이드 앱 패키지 이름을 입력하세요.
예) com.example.mynavermap

등록하기 취소

## naverblog

개요 | API 설정 | 멤버관리 | 로그인 통계 | API 통계 | Playground(Beta)

### 애플리케이션 정보

Client ID: 0GPGgUVKfYOrui4OXYL6

Client Secret: ••••••••••

보기

### API 호출 안내

지도 API 인증실패나 네이버 로그인 이용 제한이 걸렸다면 [API 설정] 탭에서 URL 관련 설정을 수정하시면 정상 이용 가능합니다 !!!

### 비로그인 오픈 API 당일 사용량

API호출량/일일허용량

검색

0/25000

# API 권한 설정

## naverblog

개요 | API 설정 | 멤버관리 | 로그인 통계 | API 통계 | Playground(Beta)

API 설정 메뉴에서는 사용하려는 API 종류와 API 서비스 환경을 설정할 수 있습니다.
네이버 로그인API를 사용하는 경우 애플리케이션 이름, 로고이미지, 개발상태도 수정할 수 있습니다.

1. 네이버 로그인 실패시 수정 방법

(1) 에러현상: '네이버는 등록되지 않은 임의의 앱(사이트)에서 네이버 아이디로 로그인을 제공하는 것을 제한하고 있습니다.' 에러 메시지 출력
(2) 원인: 하단의 '로그인 오픈 API 서비스 환경'에 설정된 1) 서비스 URL 값이 실제 서비스 URL과 일치하지 않거나 2) 네이버아이디로로그인 Callback URL 값이 잘못된 경우
(3) 조치 방법
- 서비스 URL: 실제 서비스하는 사이트 URL과 동일한지 확인 (www, 포트번호, http/https 구분 없이 도메인명만 정확히 입력)
- Callback URL: 네이버 로그인 후 이동하게되는 URL로서 5개까지 다른 Callback URL 등록 가능

2. 지도 API 인증 실패시 수정 방법

(1) 에러현상: 네이버 지도가 잠시 나타났다가, 'ClientID와 URL을 확인하세요'라는 인증 실패 alert 메세지가 출력
(2) 원인: 하단의 '비로그인 오픈 API 서비스 환경'에 설정된 서비스 URL 값이 실제 서비스 URL과 일치하지 않는 경우
(3) 조치 방법
- 실제 서비스 하는 사이트 URL과 동일한지 확인 (www, 포트번호, http/https 구분 없이 도메인명만 정확히 입력)
- 도메인은 최대 10개 까지 등록 가능하며 서브 도메인이 있을 경우는 대표 도메인만 www 없이 입력 (예: http://naver.com)
- 하이브리드앱이나 윈도우 애플리케이션은 location.href 객체 출력 값을 입력 (예: file://로컬URI)

# 네이버 API 종류

| 검색 API | 요청 URL |
|---|---|
| 블로그 | https://openapi.naver.com/v1/search/blog.json |
| 뉴스 | https://openapi.naver.com/v1/search/news.json |
| 책 | https://openapi.naver.com/v1/search/book.json |
| 성인 검색어 판별 | https://openapi.naver.com/v1/search/adult.json |
| 백과사전 | https://openapi.naver.com/v1/search/encyc.json |
| 영화 | https://openapi.naver.com/v1/search/movie.json |
| 카페글 | https://openapi.naver.com/v1/search/cafearticle.json |
| 지식iN | https://openapi.naver.com/v1/search/kin.json |
| 지역 | https://openapi.naver.com/v1/search/local.json |
| 오타변환 | https://openapi.naver.com/v1/search/errata.json |
| 웹문서 | https://openapi.naver.com/v1/search/webkr.json |
| 이미지 | https://openapi.naver.com/v1/search/image.json |
| 쇼핑 | https://openapi.naver.com/v1/search/shop.json |
| 전문자료 | https://openapi.naver.com/v1/search/doc.json |

# 네이버 블로그 서비스 API 예제


```
1   import os
2   import sys
3   import urllib.request
4
5   client_id = "[redacted]"
6   client_secret = "[redacted]"
7   encText = urllib.parse.quote("파이썬")
8   url = "https://openapi.naver.com/v1/search/blog?query=" + encText # json 결과
9   # url = "https://openapi.naver.com/v1/search/blog.xml?query=" + encText # xml 결과
10  request = urllib.request.Request(url)
11  request.add_header("X-Naver-Client-Id",client_id)
12  request.add_header("X-Naver-Client-Secret",client_secret)
13  response = urllib.request.urlopen(request)
14  rescode = response.getcode()
15  if(rescode==200):
16      response_body = response.read()
17      print(response_body.decode('utf-8'))
18  else:
19      print("Error Code:" + rescode)
```

# 네이버 블로그 서비스 API 예제 결과


{

"lastBuildDate": "Tue, 21 May 2019 14:49:40 +0900",

"total": 120380,

"start": 1,

"display": 10,

"items": [

{

"title": "Python] 몇줄의 코드만으로 Data Analysis - pandas... ",

"link": "https:\/\/blog.naver.com\/koys007?Redirect=Log&logNo=221541877701",

"description": "저는 제 경력동안 python과 data science에 종사하는 수많은 개발자들을 가르쳤습니다. python 캠프에서 제가 본 바로는 python을 배우는 크게 3가지 그룹으로 나눌수 있습니다: 웹, 과학, 기능 동작... ",

"bloggername": "koys007님의 블로그",

"bloggerlink": "https://blog.naver.com/koys007",

"postdate": "20190520"

},

…

# 네이버 뉴스 서비스 API 예제

```python
import os
import sys
import urllib.request

client_id = "[redacted]"
client_secret = "[redacted]"
encText = urllib.parse.quote("문재인")
url = "https://openapi.naver.com/v1/search/news?query=" + encText # json 결과
# url = "https://openapi.naver.com/v1/search/blog.xml?query=" + encText # xml 결과
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
response = urllib.request.urlopen(request)
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)
```

# 네이버 뉴스 서비스 API 예제 결과


{
"lastBuildDate": "Mon, 27 May 2019 17:42:17 +0900",
"total": 4277016,
"start": 1,
"display": 10,
"items": [
{
"title": "甲·乙 쏙 빼고…`갑을관계 개선` 토론회 연 공정위",
"originallink": "http://news.mk.co.kr/newsRead.php?year=2019&no=354860",
"link": "https://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=101&oid=009&aid=0004363552",
"description": "관료·정치인·학자들만 참여 토론회 패널 편향성도 논란 대부분 親정부·親노동 인사 박용만회장도 결국 불참 결정 문재인정부의 경제정책 3대 키워드 중 하나인 '공정경제'를 담당하는 공정거래위원회가 27일 문재인정부... ",
"pubDate": "Mon, 27 May 2019 17:38:00 +0900"

},
…

# 네이버 이미지 서비스 API 예제


```python
import os
import sys
import urllib.request

client_id = "[redacted]"
client_secret = "[redacted]"
encText = urllib.parse.quote("하와이")
url = "https://openapi.naver.com/v1/search/image?query=" + encText # json 결과
# url = "https://openapi.naver.com/v1/search/blog.xml?query=" + encText # xml 결과
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
response = urllib.request.urlopen(request)
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)
```

# 네이버 이미지 서비스 API 예제 결과


{

"lastBuildDate": "Mon, 27 May 2019 17:55:08 +0900",

"total": 267378,

"start": 1,

"display": 10,

"items": [

{

"title": "허니문여행지 BEST 5는 어디?",

"link": "http://post.phinf.naver.net/MjAxNzA1MDJfMTUy/MDAxNDkzNjk4OTMwMTc5.n6EwtnSo_4Xle-P-5Iu6H2x9NasKUNxwmJE30iXT_fUg.rtyKJqVzl2DTUqfU3z391RnwL_DZuefHpuI7AC_lslYg.JPEG/ILWBc5o5xveUgphJDgedSeQZ2JiU.jpg",

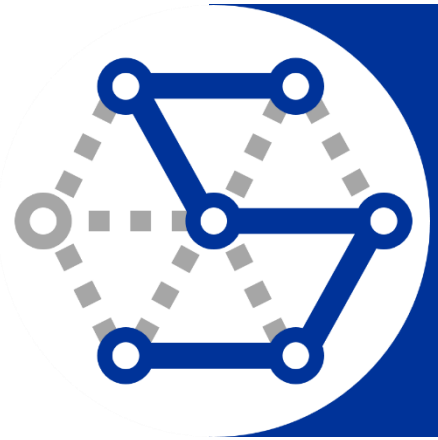
"thumbnail": "https://search.pstatic.net/common/?src=http://post.phinf.naver.net/MjAxNzA1MDJfMTUy/MDAxNDkzNjk4OTMwMTc5.n6EwtnSo_4Xle-P-5Iu6H2x9NasKUNxwmJE30iXT_fUg.rtyKJqVzl2DTUqfU3z391RnwL_DZuefHpuI7AC_lslYg.JPEG/ILWBc5o5xveUgphJDgedSeQZ2JiU.jpg&type=b150",

"sizeheight": "854",

"sizewidth": "1280"

},

…



# 2. 웹 스크래핑

# 웹 스크래핑Web Scraping

- 웹 스크래핑이란 HTTP를 통해 웹 사이트의 내용을 긁어다 원하는 형태로 가공하는 것
- 즉, 웹 사이트의 데이터를 수집하는 모든 작업을 의미함

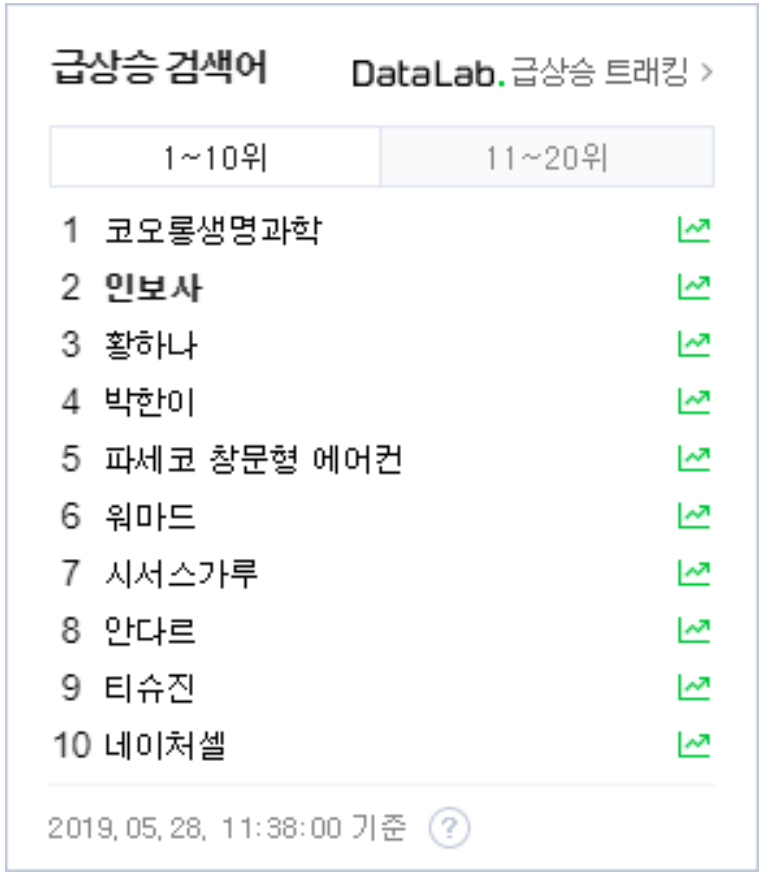
# 크롤링? 파싱?

- 크롤링
  - 웹 크롤러crawler라는 단어에서 유래되었으며 크롤러란 조직적, 자동화된 방법으로 월드와이드 웹을 탐색하는 컴퓨터 프로그램
  - 크롤링은 크롤러가 하는 작업을 부르는 말로, 여러 인터넷 사이트의 페이지(문서, html 등)를 수집해서 분류하는 것
  - 대체로 찾아낸 데이터를 저장한 후 쉽게 찾을 수 있게 인덱싱 수행
- 파싱
  - 파싱이란 어떤 페이지(문서, html 등)에서 내가 원하는 데이터를 특정 패턴이나 순서로 추출하여 정보를 가공하는 것
  - 파싱이란 일련의 문자열을 의미있는 토큰token으로 분해하고 이들로 이루어진 파스 트리parse tree를 만드는 과정
  - 입력 토큰에 내제된 자료 구조를 빌드하고 문법을 검사하는 역할을 함

# 필요한 모듈 설치

- pip install request
- pip install pandas

# 네이버 검색어 스크래핑




```python
import requests
from bs4 import BeautifulSoup

resp = requests.get('https://www.naver.com/')
soup = BeautifulSoup(resp.text, 'html.parser')
titles = soup.select('.ah_roll .ah_k')

for title in titles:
    print(title.get_text())
```

# 네이버 블로그 스크래핑


NAVER 블로그
글
갤럭시
통합검색
블로그 홈
주제별 보기
이달의 블로그
공식블로그
파워블로그
챌린지 프로그램
글
블로그
별명·아이디
갤럭시에 대한 검색결과 입니다. 1,979,474건
정확도
최신순
기간 전체
갤럭시S10 5G 후기! 스펙 카메라 기능까지 2019. 5. 26.
그럼 오늘은 5G 전용 스마트폰, 삼성전자의 갤럭시 S10 5G에 대해서 알아볼게요! 스펙 살펴보기 갤럭시 S10 5G 스펙을 살펴보면 디스플레이는 162.9mm (6.7")로 19:9 대화면비의 다이내믹 아몰레드를 지원하고 있고요. 램 8GB, 메모리는 내장 메모리 256GB와 512GB 두 가지입니다. 아쉬운 점은 외장...
블링유에 | 블링유에 텔레파시푸시
갤럭시S10 5G
갤럭시A30 삼성효도폰 학생폰으로 딱이네요! 2019. 5. 16.
그러던 중 알게된 것이 제가 이번에 아들에게 사준 갤럭시A30입니다만, 아들에게 어떤 걸 가지고 싶냐고 물어보니까 신나는 말투로 갤럭시를 쓰고 싶다길래, 그런 바램은 들어주고 싶은 것이 역시 부모의 마음 아니겠습니까. 직원분에게 고르러 왔다고 한 뒤에 이런저런 요구사항을 말했더니...
프리오스 | 프리오스토리
A30

# 라이브러리 선언 및 네이버 개발자 ID/SECRET 선언

```python
# -*- coding: utf-8 -*-
import re
import json
import math
import requests
import urllib.request
import urllib.error
import urllib.parse
from bs4 import BeautifulSoup
import pandas as pd

# https://developers.naver.com/main/ 사이트에서 어플리케이션 등록
# Naver Development ID/SECRET 필요
naver_client_id = "[redacted]"
naver_client_secret = "[redacted]"
```

# get_blog_count()


```
18  def get_blog_count(query, display):
19      encode_query = urllib.parse.quote(query)
20      search_url = "https://openapi.naver.com/v1/search/blog?query=" + encode_query
21      request = urllib.request.Request(search_url)
22
23      request.add_header("X-Naver-Client-Id", naver_client_id)
24      request.add_header("X-Naver-Client-Secret", naver_client_secret)
25
26      response = urllib.request.urlopen(request)
27      response_code = response.getcode()
28
```

# get_blog_count()


```
29      if response_code is 200:
30          response_body = response.read()
31          response_body_dict = json.loads(response_body.decode('utf-8'))
32
33          print("Last build date: " + str(response_body_dict['lastBuildDate']))
34          print("Total: " + str(response_body_dict['total']))
35          print("Start: " + str(response_body_dict['start']))
36          print("Display: " + str(response_body_dict['display']))
37
38          if response_body_dict['total'] == 0:
39              blog_count = 0
40          else:
41              blog_total = math.ceil(response_body_dict['total'] / int(display))
42
43              if blog_total >= 1000:
44                  blog_count = 1000
45              else:
46                  blog_count = blog_total
47
48              print("Blog total: " + str(blog_total))
49              print("Blog count: " + str(blog_count))
50
51          return blog_count
```

# get_blog_post()


```
54  def get_blog_post(query, display, start_index, sort):
55      global no, df
56
57      encode_query = urllib.parse.quote(query)
58      search_url = "https://openapi.naver.com/v1/search/blog?query=" + encode_query + "&display=" + str(
59          display) + "&start=" + str(start_index) + "&sort=" + sort
60
61      request = urllib.request.Request(search_url)
62
63      request.add_header("X-Naver-Client-Id", naver_client_id)
64      request.add_header("X-Naver-Client-Secret", naver_client_secret)
65
66      response = urllib.request.urlopen(request)
67      response_code = response.getcode()
68
```

# get_blog_post()

```
69      if response_code is 200:
70          response_body = response.read()
71          response_body_dict = json.loads(response_body.decode('utf-8'))
72          for item_index in range(0, len(response_body_dict['items'])):
73              try:
74                  remove_html_tag = re.compile('<.*?>')
75                  title = re.sub(remove_html_tag, '', response_body_dict['items'][item_index]['title'])
76                  link = response_body_dict['items'][item_index]['link'].replace("amp;", "")
77                  description = re.sub(remove_html_tag, '', response_body_dict['items'][item_index]['description'])
78                  blogger_name = response_body_dict['items'][item_index]['bloggername']
79                  blogger_link = response_body_dict['items'][item_index]['bloggerlink']
80                  post_date = response_body_dict['items'][item_index]['postdate']
81
82                  no += 1
83                  post_code = requests.get(link)
84                  post_text = post_code.text
85                  post_soup = BeautifulSoup(post_text, 'html.parser')
86
```

# get_blog_post()

```
87                  blog_post_content_text = ""
88                  for mainFrame in post_soup.select('iframe#mainFrame'):
89                      blog_post_url = "http://blog.naver.com" + mainFrame.get('src')
90                      blog_post_code = requests.get(blog_post_url)
91                      blog_post_text = blog_post_code.text
92                      blog_post_soup = BeautifulSoup(blog_post_text, 'html.parser')
93
94                      for blog_post_content in blog_post_soup.find_all('div', class_='se-viewer'):
95                          blog_post_content_text = blog_post_content.get_text()
96
97                      for blog_post_content in blog_post_soup.find_all('div', class_='se_doc_viewer'):
98                          blog_post_content_text = blog_post_content.get_text()
99
100                     for blog_post_content in blog_post_soup.select('div#postViewArea'):
101                         blog_post_content_text = blog_post_content.get_text()
102
103                 df.loc[no] = [title, link, description, blogger_name, blogger_link, post_date, blog_post_content_text]
104                 print("#", end='')
105
106             except:
107                 print("X", end='')
108                 item_index += 1
```

# Main


```
111  no = 0                  # 몇개의 포스트를 저장하였는지 세기 위한 index
112  query = "BTS"           # 검색을 원하는 문자열로서 UTF-8로 인코딩한다.
113  display = 10            # 검색 결과 출력 건수 지정, 10(기본값),100(최대)
114  start = 1               # 검색 시작 위치로 최대 1000까지 가능
115  sort = "sim"            # 정렬 옵션: sim(유사도순, 기본값), date(날짜순)
116
117  # 블로그를 DataFrame에 저장
118  df = pd.DataFrame(columns=("Title", "Link", "Description", "Blogger Name", "Blogger Link", "Post Date", "Post Contents"))
119
120  blog_count = get_blog_count(query, display)
121  for start_index in range(start, blog_count + 1, display):
122      get_blog_post(query, display, start_index, sort)
123
124  df.to_csv("./data/" + query + ".csv", header=True, index=False)
125  print(str(no) + "개 블로그 저장")
```

# 스크래핑 수행

Last build date: Tue, 21 May 2019 14:56:36 +0900

Total: 747682

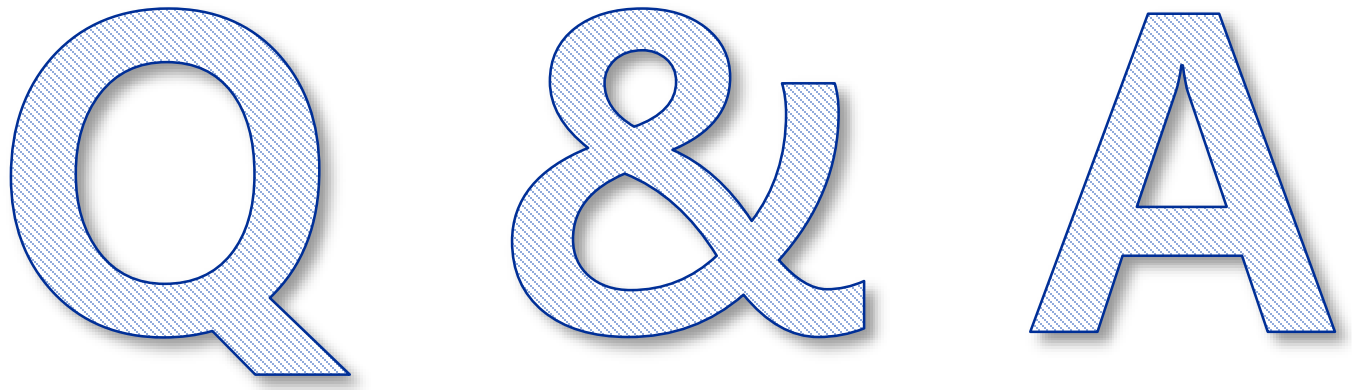
Start: 1

Display: 10

Blog total: 74769

Blog count: 1000

- BTS.csv


Q & A