



데이터 과학 기반의 파이썬 빅데이터 분석

Chapter 05 파이썬 크롤링 - API 이용

목차

01 네이버 API를 이용한 크롤링

02 공공데이터 API 기반 크롤링

학습목표

분석할 데이터를 웹에서 수집하는 크롤링 방법을 안다.

개발자를 위해 제공하는 API로 웹 데이터 크롤링을 할 수 있다.

01. 네이버 API를 이용한 크롤링

■ 크롤링이란

■ 크롤링

- 웹에서 데이터를 수집하는 작업
- 크롤러 또는 스파이더라는 프로그램으로 웹 사이트에서 데이터를 추출

• 웹 API

- 웹 API는 일반적으로 HTTP 통신을 사용하는데 사용
- 지도, 검색, 추가, 환율 등 다양한 정보를 가지고 있는 웹 사이트의 기능을 외부에서 쉽게 사용할 수 있도록 사용 절차와 규약을 정의한 것

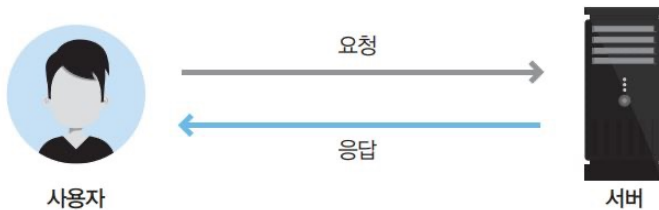


그림 5-1 웹 API를 이용한 HTTP 요청과 응답

표 5-1 웹 API 제공자

종류	주소
네이버 개발자 센터	https://developers.naver.com
카카오 앱 개발 플랫폼 서비스	https://developers.kakao.com
페이스북 개발자 센터	https://developers.facebook.com
트위터 개발자 센터	https://developer.twitter.com
공공데이터포털	https://www.data.go.kr
세계 날씨	http://openweathermap.org
유료/무료 API 스토어	http://mashup.or.kr http://www.apistore.co.kr/api/apiList.do

01. 네이버 API를 이용한 크롤링

■ 네이버 개발자 가입

1. 네이버 개발자 센터 접속하기

- 네이버 개발자 센터([https:// developers.naver.com](https://developers.naver.com))에 접속

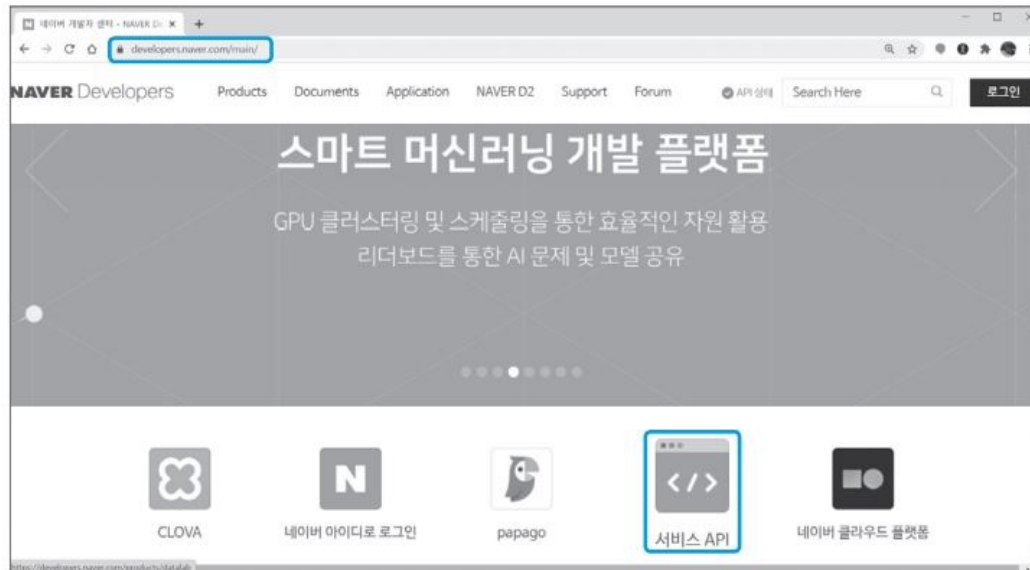


그림 5-2 네이버 개발자 센터

01. 네이버 API를 이용한 크롤링

■ 네이버 개발자 가입

2. 오픈 API 이용 신청하기

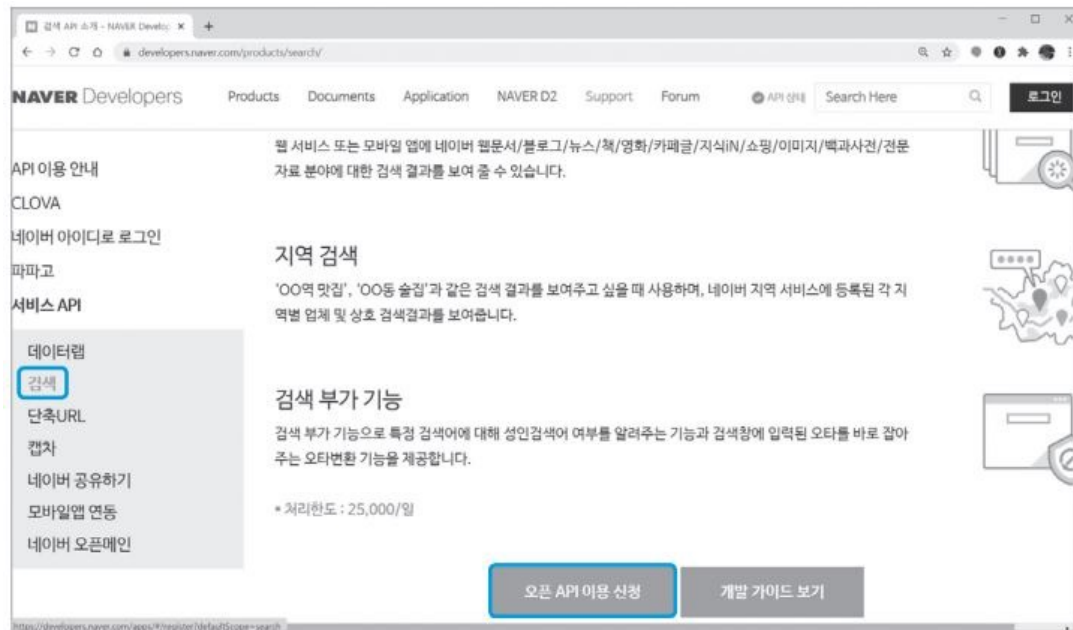


그림 5-3 오픈 API 이용 신청

01. 네이버 API를 이용한 크롤링

■ 네이버 개발자 가입

3. 애플리케이션 등록하기

내 애플리케이션

애플리케이션 등록

CLOVA Platform Console β

API 제휴 신청

계정 설정

애플리케이션 등록 (API 이용신청)

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

애플리케이션 이름 <small>◎</small>	<div>nvBig</div> <div>입력</div> <div>• 네이버 아이디로 로그인할 때 사용자에게 표시되는 이름이므로 가급적 10자 이내의 간결한 이름을 사용해주세요. • 40자 이하의 영문, 한글, 숫자, 공백문자, "-", "_", "."만 입력 가능합니다.</div>
사용 API <small>◎</small>	<div>선택하세요.</div> <div>검색</div>
비로그인 오픈 API 서비스 환경	<div>환경 추가</div> <div>WEB 선택</div> <div>웹 서비스 URL (최대 10개)</div> <div><div>http://localhost</div><div>입력</div><div>• 텍스트를 우측 끝의 "+" 버튼을 누르면 행이 추가되며, "-" 버튼을 누르면 행이 삭제됩니다. • http와 https는 구분하지 않습니다. • www는 빼고 입력해 주세요. (예) http://naver.com • 서브 도메인이 있으면 대표 도메인명만 입력해 주세요. (예: http://naver.com) • 파일저장도 않은 location.href 전체 줄의 값을 입력하면 됩니다. (예: file:///로컬 URL)</div></div> <div>등록하기 취소</div>

그림 5-4 애플리케이션 등록

01. 네이버 API를 이용한 크롤링

■ 네이버 개발자 가입

4. 애플리케이션 정보 확인하기

The screenshot displays the Naver Developer Console (nvBig) interface. On the left, a sidebar lists '내 애플리케이션' (My Applications) with four entries labeled 'nvBig', and a section for '애플리케이션 등록' (Application Registration) containing links for 'CLOVA Platform Console β', 'API 재휴 신청' (API Re-apply), and '계정 설정' (Account Settings). The main area is titled 'nvBig' and features a navigation bar with tabs: '개요' (Overview), 'API 설정' (API Settings), '멤버관리' (Member Management), '로그인 통계' (Login Statistics), 'API 통계' (API Statistics), and 'Playground (Beta)'. The 'API 설정' tab is active, showing '애플리케이션 정보' (Application Information). This section contains a table with two rows: 'Client ID' with the value '0LHQM4VX_MQM6JfkXofa' and 'Client Secret' with a masked value '*****'. A '보기' (View) button is located below the Client Secret field. At the bottom, there is a section titled 'API 호출 안내' (API Call Guide) with a note: '지도 API 인증실때나 네이버 로그인 이용 제한이 걸렸다면 [API 설정] 탭에서 URL 관련 설정을 수정하시면 정상 이용 가능합니다 !!!' (If you encounter restrictions when using the Map API authentication or Naver login, you can modify the URL-related settings in the [API Settings] tab to use normally !!!).

그림 5-5 애플리케이션 정보 확인

그림 5-6 검색 API 이용 안내 페이지

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

1. 전체 작업 설계하기

작업 설계	사용할 코드
1. 검색어 지정하기	srcText = '월드컵'
2. 네이버 뉴스 검색하기	getNaverSearch()
2.1 url 구성하기	url = base + node + srcText
2.2 url 접속과 검색 요청하기	urllib.request.urlopen()
2.3 요청 결과를 응답 JSON으로 받기	json.load()
3. 응답 데이터를 정리하여 리스트에 저장하기	getPostData()
4. 리스트를 JSON 파일로 저장하기	json.dumps()

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

2. 프로그램 구성 설계하기

[CODE 0]

```
def main()
```

1. 검색어 지정

2. 네이버 뉴스 검색

3. 응답 데이터 정리 후
리스트에 저장

4. 리스트를 JSON 파일로 저장

[CODE 2]

getNaverSearch()

json.load(responseDecode)

[CODE 1]

getRequestUrl()

Response.read()

[CODE 3]

getPostData()

jsonResult



01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

1. [CODE 0] 전체 작업 스토리를 설계

- 지역 변수

node: 네이버 검색 API에서 검색할 대상 노드([표 5-2] 참고)

srcText: 사용자 입력으로 받은 검색어 저장

cnt: 검색 결과 카운트

jsonResult: 검색 결과를 정리하여 저장할 리스트 객체

jsonResponse: 네이버 뉴스 검색에 대한 응답을 저장하는 객체

total: 전체 검색 결과 개수

post: 응답받은 검색 결과 중에서 한 개를 저장한 객체

items: 전체 응답 검색 결과로 내부 항목은 title, originallink, link, description, pubDate

jsonFile: JSON 파일에 저장할 데이터를 담은 객체

- 메서드

input('검색어를 입력하세요: '): 사용자로부터 입력을 받는다.

getNaverSearch(node, srcText, 1, 100): 1부터 100개의 검색 결과를 처리한다([CODE 2]).

getPostData(): 검색 결과 한 개를 처리한다([CODE 2]).

json.dumps(): 객체를 JSON 형식으로 변환

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

```
01 def main():
02     node = 'news' #크롤링할 대상
03     srcText = input('검색어를 입력하세요: ')
04     cnt = 0
05     jsonResult = []
06
07     jsonResponse = getNaverSearch(node, srcText, 1, 100) #[CODE 2]
08     total = jsonResponse['total']
09
10     while ((jsonResponse != None) and (jsonResponse['display'] != 0)):
11         for post in jsonResponse['items']:
12             cnt += 1
13             getPostData(post, jsonResult, cnt) #[CODE 3]
14
15             start = jsonResponse['start'] + jsonResponse['display']
16             jsonResponse = getNaverSearch(node, srcText, start, 100) #[CODE 2]
17
18     print('전체 검색 : %d 건' %total)
19
20     with open('%s_naver_%s.json' % (srcText, node), 'w', encoding = 'utf8')
        as outfile:
21         jsonFile = json.dumps(jsonResult, indent = 4, sort_keys = True,
                                ensure_ascii = False)
22
23         outfile.write(jsonFile)
24
25     print("가져온 데이터 : %d 건" %(cnt))
26     print('%s_naver_%s.json SAVED' % (srcText, node))
```

02행

네이버 뉴스 검색을 위해 검색 API 대상을 'news'로 설정

03행

파이썬 셸 창에서 검색어를 입력받아 srcText에 저장

07행

getNaverSearch() 함수를 호출하여 start = 1, display= 100에 대한 검색 결과를 반환받아 jsonResponse에 저장

10~16행

검색 결과 jsonResponse에 데이터가 있는 동안 for문(11~13행) 으로 검색 결과를 한 개씩 처리하는 작업 getPostData()을 반복 반복 작업이 끝나면 다음 검색 결과 100개를 가져오기 위해 start 위치를 변경

15행

getNaverSearch() 함수를 호출하여 새로운 검색 결과를 jsonResponse에 저장하고 16행 for문 11~13행을 다시 반복

20~23행

파일 객체를 생성하여 정리된 데이터를 JSON 파일에 저장

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

표 5-2 네이버 검색 API 개발자 가이드(<https://developers.naver.com/docs/search/news>)

구분	내용 및 설명	
URL	뉴스	https://openapi.naver.com/v1/search/news.json
	블로그	https://openapi.naver.com/v1/search/blog.json
	카페	https://openapi.naver.com/v1/search/cafearticle.json
	영화	https://openapi.naver.com/v1/search/movie.json
	쇼핑	https://openapi.naver.com/v1/search/shop.json
요청 변수	query	검색을 원하는 문자열이며 UTF-8로 인코딩한다.
	start	검색 시작 위치로 최대 1000까지 가능하다. 1(기본값)~1000(최대값)
	display	검색 결과 출력 건수를 지정한다. 10(기본값)~100(최대값)
응답 변수	items	검색 결과로 title, originallink, link, description, pubDate를 포함한다.
	title	검색 결과 문서의 제목이다.
	link	검색 결과 문서를 제공하는 네이버의 하이퍼텍스트 link다.
	originallink	검색 결과 문서를 제공하는 언론사의 하이퍼텍스트 link다.
	description	검색 결과 문서의 내용을 요약한 정보다.
	pubDate	검색 결과 문서가 네이버에 제공된 시간이다.

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

2. [CODE 1] url 접속을 요청하고 응답을 받아서 반환하는 부분을 작성

- 매개변수

url: 네이버 뉴스 검색('월드컵')에 대한 url

- 지역 변수

req: url 접속 요청(request) 객체

app_id: 네이버 개발자로 등록하고 받은 Client ID

app_secret: 네이버 개발자로 등록하고 받은 Client Secret

response: 네이버 서버에서 받은 응답을 저장하는 객체

- 메서드

urllib.request.Request(): urllib 패키지의 request 모듈에 있는 Request() 함수로

네이버 서버에 보낼 요청(request) 객체를 생성

Request.add_header(): 서버에 보내는 요청 객체에 헤더 정보를 추가

urllib.request.urlopen(): 서버에서 받은 응답을 변수에 저장하기 위해 메모리로 가져오는

urllib 패키지의 request 모듈에 있는 함수

response.getcode(): 요청 처리에 대한 응답 상태를 확인하는 response 객체의 멤버 함수로 상태 코드가 200이면
요청 처리 성공을 나타냄

datetime.datetime.now(): 현재 시간을 구하는 함수

response.read().decode('utf-8'): utf-8 형식으로 문자열을 디코딩

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

```
01 def getRequestUrl(url):
02     req = urllib.request.Request(url)
03     req.add_header("X-Naver-Client-Id", client_id)
04     req.add_header("X-Naver-Client-Secret", client_secret)
05
06     try:
07         response = urllib.request.urlopen(req)
08         if response.getcode() == 200:
09             print("[%s] Url Request Success" % datetime.datetime.now())
10             return response.read().decode('utf-8')
11     except Exception as e:
12         print(e)
13         print("[%s] Error for URL : %s" % (datetime.datetime.now(), url))
14     return None
```

02행

매개변수로 받은 url에 대한 요청을 보낼 객체를 생성

03~04행

API를 사용하기 위한 Client ID와 Client Secret 코드를 요청 객체 헤드에 추가

07행

요청 객체를 보내고 그에 대한 응답을 받아 response 객체에 저장

08~10행

getcode()로 response 객체에 저장된 코드를 확인
200이면 요청이 정상처리된 것이므로 성공 메시지를 파이썬 셸 창에 출력하고 응답을 utf-8 형식으로 디코딩하여 반환

11~14행

요청이 처리되지 않은 예외 사항exception이 발생하면 에러 메시지를 파이썬 셸 창에 출력

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

3. [CODE 2] 네이버 뉴스 검색 url을 만들고 [CODE 1]의 `getRequestUrl(url)`을 호출하여 반환받은 응답 데이터를 파이썬 json 형식으로 반환하는 부분

- 매개변수

node: 네이버 검색 API를 이용하여 검색할 대상 노드(news, blog, cafearticle, movie, shop 등 [표 5-2] 참고)

srcText: 검색어

page_start: 검색 시작 위치(1~1000)

display: 출력 건수(10~100)

- 지역 변수

base: 검색 url의 기본 주소

node: 검색 대상에 따른 json 파일 이름

parameter: url에 추가할 검색어와 검색 시작 위치, 출력 건수 등의 매개변수

responseDecode: `getRequestUrl(url)`을 호출하여 반환받은 응답 객체(utf-8로 디코드)

- 메서드

`getRequestUrl(url)`: [CODE1]을 호출하여 url 요청에 대한 응답을 받음

`json.loads(responseDecode)`: 응답 객체를 파이썬이 처리할 수 있는 JSON 형식으로 변환

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

```
01 def getNaverSearch(node, srcText, page_start, display):
02     base = "https://openapi.naver.com/v1/search"
03     node = "/%s.json" % node
04     parameters = "?query=%s&start=%s&display=%s" % (urllib.parse.
        quote(srcText), start, display)
05
06     url = base + node + parameters
07     responseDecode = getReqeustUrl(url) #[CODE 1]
08
09     if (responseDecode == None):
10         return None
11     else:
12         return json.loads(responseDecode)
```

02~06행

[표 5-2]의 네이버 검색 API 정보에 따라 요청 URL을 구성

07행

완성한 url을 이용하여 getReqeustUrl() 함수를 호출하여 받은 utf-8 디코드 응답을 responseDecode에 저장

12행

서버에서 받은 JSON 형태의 응답 객체를 파이썬 객체로 로드하여 반환

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

4. [CODE 3] JSON 형식의 응답 데이터를 필요한 항목만 정리하여 딕셔너리 리스트인 `jsonResult`를 구성하고 반환하도록 작성

- 매개변수

`post`: 응답으로 받은 검색 결과 데이터 중에서 결과 한 개를 저장한 객체

`jsonResult`: 필요한 부분만 저장하여 반환할 리스트 객체

`cnt`: 현재 작업 중인 검색 결과의 번호

- 지역 변수

`post['title']`: `post` 객체의 `title` 항목에 저장된 값

`post['description']`: `post` 객체의 `description` 항목에 저장된 값

`post['originallink']`: `post` 객체의 `originallink` 항목에 저장된 값

`post['link']`: `post` 객체의 `link` 항목에 저장된 값

- 메서드

`datetime.datetime.strptime()`: 문자열을 날짜 객체 형식으로 변환

`pDate.strftime()`: 날짜 객체의 표시 형식을 지정

`jsonResult.append()`: 리스트 객체인 `jsonResult`에 원소를 추가

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

3. 함수 설계하기

```
01 def getPostData(post, jsonResult, cnt):
02     title = post['title']
03     description = post['description']
04     org_link = post['originallink']
05     link = post['link']
06
07     pDate = datetime.datetime.strptime(post['pubDate'], '%a,
           %d %b %Y %H:%M:%S +0900')
08     pDate = pDate.strftime('%Y-%m-%d %H:%M:%S')
09
10     jsonResult.append({'cnt':cnt, 'title':title, 'description': description,
           'org_link':org_link, 'link': org_link, 'pDate':pDate})
11     return
```

02~05행

검색 결과가 들어 있는 post 객체에서 필요한 데이터 항목을 추출하여 변수에 저장

07행

네이버에서 제공하는 시간인 pubDate는 문자열 형태이므로 날짜 객체로 변환

pubDate는 그리니치 평균시 형식을 사용하는데 한국 표준시보다 9시간 느리므로 +0900 을 사용해 한국 표준시로 맞춤

08행

수정된 날짜를 '연-월-일 시:분:초' 형식으로 나타냄

10행

2~5행에서 저장한 데이터를 딕셔너리 형태인 {'키':값}으로 구성하여 리스트 객체인 jsonResult에 추가

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

4. 전체 프로그램 작성하기

- 파이썬 셸 창에서 [File]-[New File]을 클릭해 새 파일 창을 열고 다음의 파이썬 프로그램을 작성

[프로그램 5-1] nvCrawler.py

```
import os
import sys
import urllib.request
import datetime
import time
import json

client_id = '본인이 발급받은 네이버 Client ID'
client_secret = '본인이 발급받은 네이버 Client Secret'

#[CODE 1]
def getRequestUrl(url):
    req = urllib.request.Request(url)
    req.add_header("X-Naver-Client-Id", client_id)
    req.add_header("X-Naver-Client-Secret", client_secret)

    try:
        response = urllib.request.urlopen(req)
        if response.getcode() == 200:
            print("[%s] Url Request Success" % datetime.datetime.now())
            return response.read().decode('utf-8')
```

```
    except Exception as e:
        print(e)
        print("[%s] Error for URL : %s" % (datetime.datetime.now(), url))
        return None
```

#[CODE 2]

def getNaverSearch(node, srcText, start, display):

```
    base = "https://openapi.naver.com/v1/search"
    node = "/%s.json" % node
    parameters = "?query = %s&start = %s&display = %s" %
        (urllib.parse.quote(srcText), start, display)
```

```
    url = base + node + parameters
    responseDecode = getRequestUrl(url) #[CODE 1]
```

```
    if (responseDecode == None):
        return None
    else:
        return json.loads(responseDecode)
```

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

4. 전체 프로그램 작성하기

```
#[CODE 3]
def getPostData(post, jsonResult, cnt):
    title = post['title']
    description = post['description']
    org_link = post['originallink']
    link = post['link']

    pDate = datetime.datetime.strptime(post['pubDate'], '%a,
                                         %d %b %Y %H:%M:%S+0900')
    pDate = pDate.strftime('%Y-%m-%d %H:%M:%S')

    jsonResult.append({'cnt':cnt, 'title':title, 'description': description,
                      'org_link':org_link, 'link': org_link, 'pDate':pDate})

    return

#[CODE 0]
def main():
    node = 'news' #크롤링할 대상
    srcText = input('검색어를 입력하세요: ')
    cnt = 0
    jsonResult = []
    jsonResponse = getNaverSearch(node, srcText, 1, 100) #[CODE 2]
    total = jsonResponse['total']
```

```
while ((jsonResponse != None) and (jsonResponse['display'] != 0)):
    for post in jsonResponse['items']:
        cnt += 1
        getPostData(post, jsonResult, cnt) #[CODE 3]

    start = jsonResponse['start'] + jsonResponse['display']
    jsonResponse = getNaverSearch(node, srcText, start, 100) #[CODE 2]

print('전체 검색 : %d 건' %total)

with open('%s_naver_%s.json' % (srcText, node), 'w', encoding='utf8')
    as outfile:
        jsonFile = json.dumps(jsonResult, indent = 4, sort_keys = True,
                               ensure_ascii = False)

        outfile.write(jsonFile)

print("가져온 데이터 : %d 건" %(cnt))
print('%s_naver_%s.json SAVED' % (srcText, node))

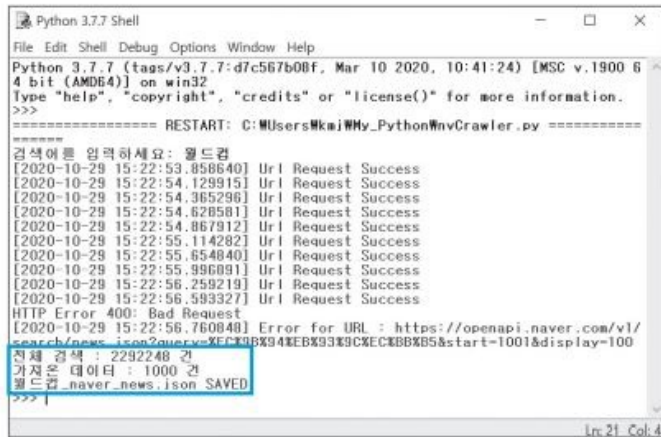
if __name__ == '__main__':
    main()
```

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

4. 전체 프로그램 작성하기

- [F5]를 눌러 실행하면 파이썬 셸 창에 print 명령의 실행 결과인 '검색어를 입력하세요:'가 출력
- '월드컵'을 입력하면 nvCrawler.py 파일이 저장된 위치에 JSON 파일이 생성



```
Python 3.7.7 Shell
File Edit Shell Debug Options Window Help
Python 3.7.7 (tags/v3.7.7:d7c567b0bf, Mar 10 2020, 10:41:24) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\kaj\My_Python\nvCrawler.py =====
검색어를 입력하세요: 월드컵
[2020-10-29 15:22:53.858640] Url Request Success
[2020-10-29 15:22:54.129915] Url Request Success
[2020-10-29 15:22:54.365296] Url Request Success
[2020-10-29 15:22:54.628581] Url Request Success
[2020-10-29 15:22:54.867912] Url Request Success
[2020-10-29 15:22:55.114282] Url Request Success
[2020-10-29 15:22:55.654840] Url Request Success
[2020-10-29 15:22:55.996091] Url Request Success
[2020-10-29 15:22:56.259219] Url Request Success
[2020-10-29 15:22:56.593327] Url Request Success
HTTP Error 400: Bad Request
[2020-10-29 15:22:56.760848] Error for URL : https://openapi.naver.com/v1/search/news.json?query=월드컵&start=1001&display=100
현재 검색 : 2292248 건
가져온 데이터 : 1000 건
월드컵_naver_news.json SAVED
>>>
```

(a) 파이썬 셸 창에 출력된 print 명령 실행 결과



(b) JSON 파일 생성

그림 5-7 실행 결과 확인 1

01. 네이버 API를 이용한 크롤링

■ 네이버 뉴스 크롤링

4. 전체 프로그램 작성하기

- 결과 확인하기



(a) JSON 파일을 메모장에서 열어 확인하기



(b) 네이버 웹 페이지에서 '월드컵'으로 검색한 결과

그림 5-8 실행 결과 확인 2

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

1. 공공데이터포털 회원가입하기

회원가입 | 공공데이터포털

data.go.kr/aim/mss/mberSubscribeConfirmFormView.do

국민과 함께 하는 공공데이터포털에 오신 것을 환영합니다.

회원가입을 하시면 공공데이터포털을 통해 국가가 보유하고 있는 유용한 공공데이터와 오픈API 등 모든 정보를 자유롭게 이용하실 수 있습니다.

STEP1 가입확인 STEP2 약관동의 STEP3 정보입력

회원가입여부 확인을 위해 이름, 이메일 주소를 입력해주세요.

이름

이메일

가입확인

그림 5-9 공공데이터포털의 회원가입 화면

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

2. 출입국관광통계서비스 검색하기

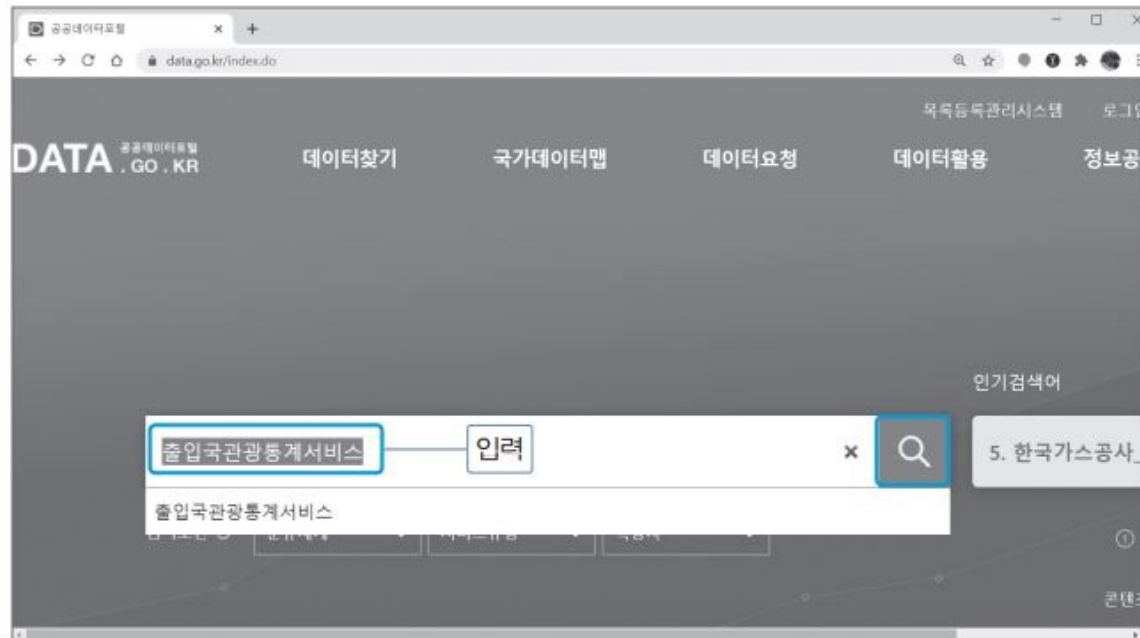


그림 5-10 로그인 후 데이터 검색

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

3. [오픈 API] 탭을 선택한 뒤 API 목록에서 [출입국관광통계서비스]를 클릭

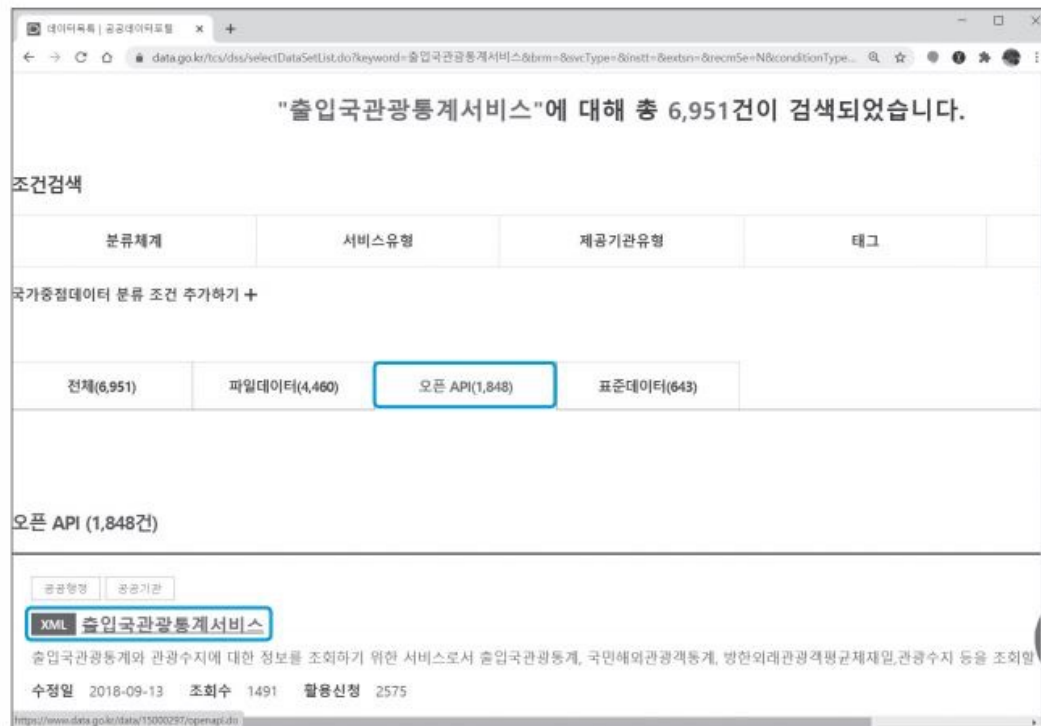


그림 5-11 원하는 API 선택

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

4. OpenAPI 개발계정 신청하기



그림 5-12 오픈 API 활용 신청하기

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

5. [OpenAPI 개발계정 신청] 페이지의 [활용목적]에서 [연구(논문 등)]을 선택한 뒤 아래 텍스트 박스에 '공공데이터 활용 학습'을 입력

The screenshot shows the 'OpenAPI 개발계정 신청' (OpenAPI Development Account Application) page. The left sidebar contains navigation links: '개발계정' (Development Account), '활용현황' (Usage Status), '운영계정' (Operation Account), and '인증키 발급현황' (Authentication Key Issuance Status). The main content area is titled '출입국관광동계서비스' (Immigration, Customs and Border Service) and displays a table with application details.

제공기관	한국문화관광연구원	서비스유형	REST
심의여부	자동승인	신청유형	개발계정 활용신청
처리상태	신청	활용기간	승인일로부터 24개월

Below the table, there is a section titled '공공데이터 제공제도' (Public Data Provision System) with three bullet points explaining the terms of use. The '활용목적 선택' (Purpose of Use Selection) section shows a list of radio buttons: '웹 사이트 개발', '연계발 (모바일, 솔루션 등)', '기타', '참고자료', and '연구(논문 등)'. The '연구(논문 등)' option is selected. Below this, there is a text input field with the placeholder '공공데이터 활용 학습' and a button labeled '입력' (Input).

그림 5-13 활용 목적 입력

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

6. 상세기능정보 선택]에서 [출입국관광통계조회]를 선택하고 [라이선스 표시]에서 [동의합니다]에 체크한 뒤 버튼을 클릭

The screenshot shows a web browser window with the URL `data.go.kr/ins/api/selectDevAccountRequestForm.do?publicDataDetailPk=c0dd79af556-cbf-8881-a80b-c205d511078b`. The page is titled '상세기능정보 선택' (Detailed Function Information Selection). It contains a table with the following data:

<input type="checkbox"/>	상세기능	설명
<input checked="" type="checkbox"/>	출입국관광통계조회	기간, 국가의 검색조건에 따라 관광출입국자수를 제공하는 기능
<input type="checkbox"/>	국민해외관광객통계조회	월별, 성별, 연령대, 출국항의 검색 조건에 따라 국민해외관광객수를 제공하는 기능
<input type="checkbox"/>	방한외래관광객통계조회	월별, 국적, 성별, 연령대, 여행목적, 입국항의 검색 조건에 따라 방한외래관광객수를 제공하는 기능
<input type="checkbox"/>	방한외래관광객평균체재일수	월별, 국적의 검색 조건에 따라 방한외래관광객 평균체재일수를 제공하는 기능
<input type="checkbox"/>	관광수지조회	월별 관광 수입과 지출, 1인당 평균 소비액을 제공하는 기능

Below the table is the '라이선스 표시' (License Display) section. It includes a checkbox for '이용허락범위' (Scope of Use) and a checkbox for '저작자표시' (Attribution), both of which are checked. The '동의합니다' (I agree) checkbox is also checked. At the bottom right, there are two buttons: '취소' (Cancel) and '활용신청' (Apply for Use).

그림 5-14 활용 신청 완료

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

7. OpenAPI 개발계정 발급받기

DATA .GO . KR

데이터찾기 국가데이터맵 데이터요청 데이터활용 정보공유

마이페이지

오픈API

개발계정

활용현황

운영계정

인증키 발급현황

DATA

나의 문의

나의 관심

나의 제공신청

나의 분쟁조정

회원정보 수정

개발계정 상세보기

기본정보

데이터명	출입국관광통계서비스	상세설명
서비스유형	REST	심의여부 자동승인
신청유형	개발계정 활용신청	처리상태 승인
활용기간	2020-10-29 ~ 2022-10-29	

서비스정보

일반 인증키 (UTF-8)	Ody77GLuYeR%2FeFqbpduMN2Bi4Cka2ftbtgnj6E2Eux1kUhy3e4epR28XKBuaObiqPoVzAizxXMBPX D
End Point	http://openapi.tour.go.kr/openapi/service
데이터포맷	XML
참고문서	한국문화관광연구원 출입국관광통계서비스 활용가이드 v1.1.docx

[처리상태]가 [승인]이므로
바로 사용할 수 있다.

[활용기간]은 신청일로부터
24개월이며 연장 신청이 가능하다.

발급된 인증키 정보다.
크롤링 작업에 필요하므로
복사해둔다.

링크를 클릭하면 오픈 API를
이용하는 자세한 방법을 알 수 있다.

그림 5-15 발급받은 개발계정 확인

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

8. 오픈 API 사용 방법 확인하기

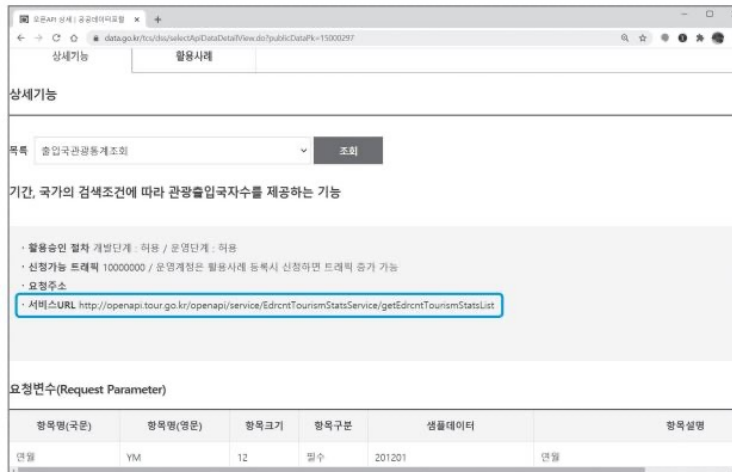


그림 5-16 오픈 API 사용 방법 확인

출력결과(Response Element)

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
결과코드	resultCode	4	필수	0000	결과코드
결과메시지	resultMsg	50	필수	OK	결과메시지
한 페이지 결과 수	numOfRows	2	옵션	10	한 페이지 결과 수
페이지 번호	pageNo	5	옵션	1	페이지 번호
전체 결과 수	totalCount	7	옵션	12334	전체 결과 수
출입국 구분	ed	14	필수	방한외래관광객	출입국구분

그림 5-17 크롤링 결과로 받을 데이터 항목

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

8. 오픈 API 사용 방법 확인하기

- [요청변수]: 서비스 URL 뒤에 추가할 매개변수 항목
- [출력결과]: 크롤링 결과로 받을 데이터 항목

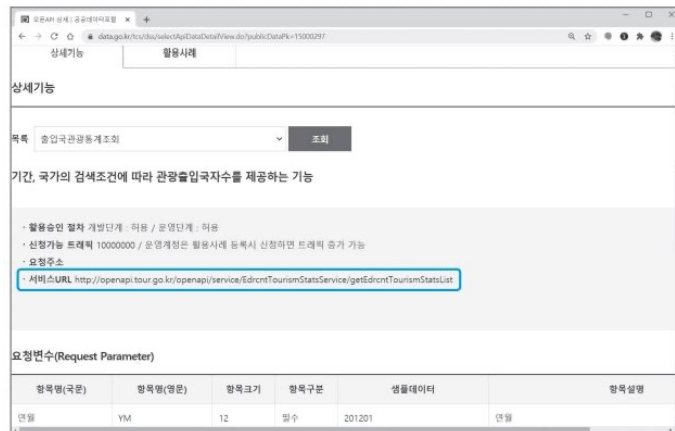


그림 5-16 오픈 API 사용 방법 확인

The screenshot shows the '출력결과(Response Element)' table, which details the structure of the data returned by the API. It includes fields like resultCode, resultMsg, numOfRows, pageNo, totalCount, and ed.

항목명(국문)	항목명(영문)	항목크기	항목구분	샘플데이터	항목설명
결과코드	resultCode	4	필수	0000	결과코드
결과메시지	resultMsg	50	필수	OK	결과메시지
한 페이지 결과 수	numOfRows	2	옵션	10	한 페이지 결과 수
페이지 번호	pageNo	5	옵션	1	페이지 번호
전체 결과 수	totalCount	7	옵션	12334	전체 결과 수
출입국 구분	ed	14	필수	방한외래관광객	출입국구분

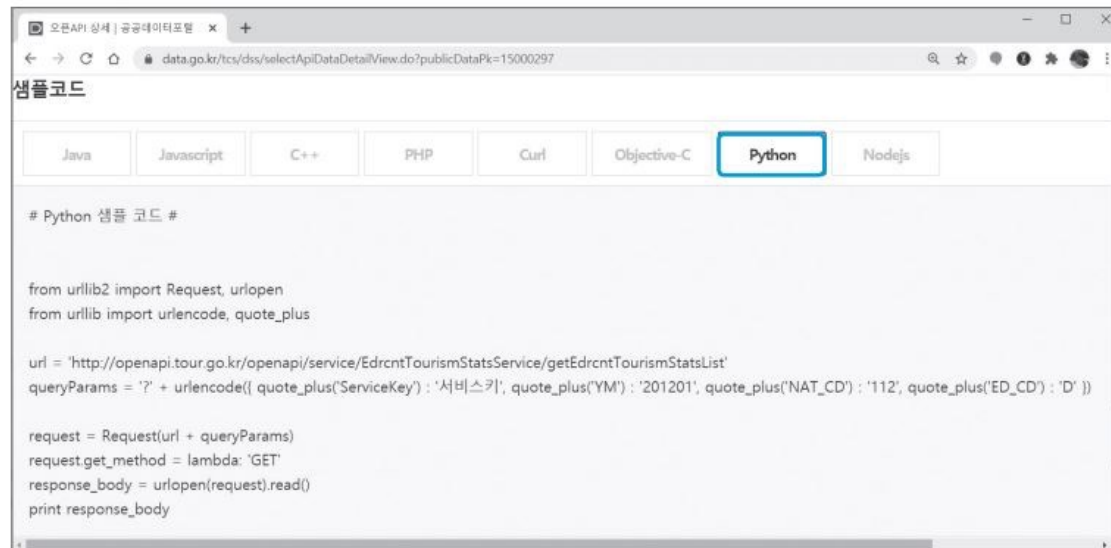
그림 5-17 크롤링 결과로 받을 데이터 항목

02. 공공데이터 API 기반 크롤링

■ 공공데이터 활용신청

8. 오픈 API 사용 방법 확인하기

- [샘플코드]: 서비스 URL과 매개변수를 연결해서 만든 url에 HTTP 요청을 보내고 응답을 받는 작업을 프로그래밍 언어로 구현한 코드를 보임



```
# Python 샘플 코드 #

from urllib2 import Request, urlopen
from urllib import urlencode, quote_plus

url = 'http://openapi.tour.go.kr/openapi/service/EdrcntTourismStatsService/getEdrcntTourismStatsList'
queryParams = '?' + urlencode([ quote_plus('ServiceKey') : '서비스키', quote_plus('YM') : '201201', quote_plus('NAT_CD') : '112', quote_plus('ED_CD') : 'D' ])

request = Request(url + queryParams)
request.get_method = lambda: 'GET'
response_body = urlopen(request).read()
print response_body
```

그림 5-18 파이썬으로 작성된 샘플코드 확인

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

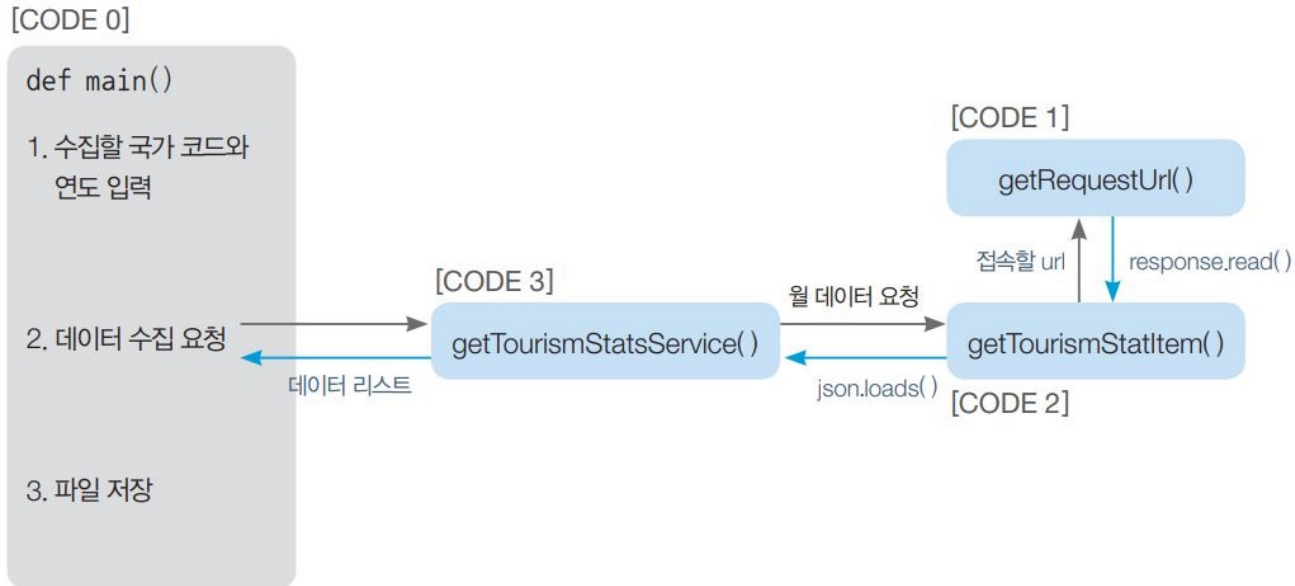
1. 전체 작업 설계하기

작업 설계	사용할 코드
1. 데이터를 수집할 국가코드와 연도 입력하기	<code>national_code, nStartYear, nEndYear</code>
2. 데이터 수집 요청하기	<code>getTourismStatsService()</code>
2.1 url 구성하여 데이터 요청하기	<code>getTourismStatsItem()</code>
2.2 url 접속하고 요청하기	<code>getRequestUrl()</code>
2.3 응답 데이터를 리스트로 구성하기	<code>jsonResult, result</code>
3. 데이터를 JSON 파일과 CSV 파일로 저장하기	<code>json.dumps(), to_csv()</code>

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

2. 프로그램 구성 설계하기



02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

1. [CODE 0] 전체 작업 스토리를 구성

- 지역 변수

- jsonResult: 수집한 데이터를 저장할 리스트 객체로 JSON 파일 저장용
- result: 수집한 데이터를 저장할 리스트 객체로 CSV 파일 저장용
- nat_cd: 데이터를 수집할 국가 코드
- natName: 데이터를 수집할 국가 이름
- ed_cd: 입국/출국 코드('E' 또는 'D')
- nStartYear: 데이터 수집 시작 연도
- nEndYear: 데이터 수집 끝 연도
- dataEND: 마지막 데이터의 연월
- jsonFile: JSON 파일에 저장할 데이터를 담은 객체

- 메서드

- input(): 사용자로부터 입력을 받는다.
- getTourismStatsList(): 방한외래관광객 데이터를 요청 ([CODE 3])
- json.dumps(): 객체를 JSON 형식으로 변환
- pd.DataFrame(): 리스트를 데이터프레임 형식으로 변환
- to_csv(): 데이터프레임을 CSV 파일로 저장

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

```
01 def main():
02     jsonResult = []
03     result = []
04
05     print("<< 국내 입국한 외국인의 통계 데이터를 수집합니다. >>")
06     nat_cd = input('국가 코드를 입력하세요(중국: 112 / 일본: 130 / 미국: 275) :')
07     nStartYear = int(input('데이터를 몇 년부터 수집할까요? : '))
08     nEndYear = int(input('데이터를 몇 년까지 수집할까요? : '))
09     ed_cd = "E" #E : 방한외래관광객, D : 해외 출국
10
11     jsonResult, result, natName, dataEND = getTourismStatsService(nat_
                                cd, ed_cd, nStartYear, nEndYear) #[CODE 3]
12
13     #파일저장 1 : json 파일
14     with open('./%s_%s_%d_%s.json' % (natName, ed, nStartYear, dataEND),
                'w',encoding='utf8') as outfile:
15         jsonFile = json.dumps(jsonResult, indent = 4, sort_keys = True,
                                ensure_ascii = False)
16         outfile.write(jsonFile)
17     #파일저장 2 : csv 파일
18     columns = ["입국자국가", "국가코드", "입국연월", "입국자 수"]
19     result_df = pd.DataFrame(result, columns = columns)
20     result_df.to_csv('./%s_%s_%d_%s.csv' % (natName, ed, nStartYear, dataEND),
                        index = False, encoding = 'cp949')
```

06행

데이터를 수집할 국가 코드를 입력

07행

데이터를 수집할 시작 연도를 입력

08행

데이터를 수집할 마지막 연도를 입력

11행

getTourismStatsService() 함수를 호출하여 반환받은
수집 데이터를 jsonResult,result, natName, dataEND에 저장

14~16행

수집 데이터를 딕셔너리의 리스트로 저장한 jsonResult를
json.dumps()를 통해json 객체로 변환한 후 JSON 파일에 저장

18행

데이터프레임에 만들 컬럼명을 리스트로 만들

19행

수집 데이터를 리스트로 저장한 result를 데이터프레임으로 변환

20행

데이터프레임 객체인 result_df를 CSV 파일로 저장

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

2. [CODE 1] url 접속을 요청하고 응답을 받아서 반환

- 매개변수

- url: 출입국관광통계서비스의 오픈 API를 사용하는 데이터를 요청하는 url

- 지역 변수

- req: url 접속을 요청하는 객체
 - response: 서버에서 받은 응답을 저장하는 객체

- 메서드

- urllib.request.Request(): urllib 패키지의 request 모듈에 있는 Request() 함수로 요청 객체를 생성
 - urllib.request.urlopen(): 서버에 요청을 보내고 받은 응답을 객체로 반환
 - response.getcode(): 요청 처리에 대한 응답 상태를 확인하는 response 객체의 멤버 함수, 상태 코드가 200이면 요청 처리 성공을 나타냄
 - datetime.datetime.now(): 현재 시간을 구함
 - response.read().decode('utf-8'): 문자열을 utf-8 형식으로 디코딩

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

```
01 def getRequestUrl(url):
02     req = urllib.request.Request(url)
03     try:
04         response = urllib.request.urlopen(req)
05         if response.getcode() == 200:
06             print("[%s] Url Request Success" % datetime.datetime.now())
07             return response.read().decode('utf-8')
08     except Exception as e:
09         print(e)
10         print("[%s] Error for URL : %s" % (datetime.datetime.now(), url))
11         return None
```

02행

매개변수로 받은 url에 대한 요청을 보낼 객체를 생성

04행

요청 객체를 보내서 받은 응답 데이터를 response 객체에 저장

05~07행

response 객체에 저장된 코드를 확인

코드가 200이면 요청을 정상 처리한 것이므로 성공

메시지와 현재 시간을 파이썬 셸 창에 출력하고 응답을 utf-8
형식으로 디코딩하여 반환

08~11행

요청이 처리되지 않은 예외 사항이 발생하면 에러 메시지를
파이썬 셸 창에 출력

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

3. [CODE 2] 출입국관광통계서비스의 오픈 API를 사용하여 데이터 요청 url을 만들고 [CODE 1]의 `getRequestUrl(url)`을 호출해서 받은 응답 데이터를 반환

- 매개변수

- `yyyymm`: 수집할 연월(예: 202003)
- `nat_cd`: 수집 대상 국가의 코드(예: 중국 = 112)
- `ed_cd`: 수집할 데이터 종류(방한외래관광객 = "E")

- 지역 변수

- `service_url`: 출입국관광통계서비스의 공공데이터에 접속할 앤드 포인트 주소
- `parameters`: url에 추가할 매개변수
- `url`: `service_url`과 `parameters`를 연결하여 완성한 url
- `responseDecode`: [CODE1]의 `getRequestUrl(url)`을 호출하여 반환받은 응답 객체

- 메서드

- `getRequestUrl()`: [CODE1]을 호출하여 url 요청에 대한 응답 데이터를 받음
- `json.loads()`: json 형식으로 받은 응답 데이터인 `responseDecode`를 파이썬 객체로 읽음

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

```
01 def getTourismStatsItem(yyyyymm, nat_cd, ed_cd):
02     service_url = "http://openapi.tour.go.kr/openapi/service/
           EdrcntTourismStatsService/getEdrcntTourismStatsList"
03     parameters = "?_type=json&serviceKey=" + ServiceKey #인증키
04     parameters += "&YM=" + yyyyymm
05     parameters += "&NAT_CD=" + nat_cd
06     parameters += "&ED_CD=" + ed_cd
07
08     url = service_url + parameters
09
10     responseDecode = getRequestUrl(url) #[CODE 1]
11
12     if (responseDecode == None):
13         return None
14     else:
15         return json.loads(responseDecode)
```

02~08행

출입국관광통계서비스의 오픈 API 상세정보 페이지에서 찾은 서비스 URL, 요청매개변수 정보, 발급받은 인증키를 사용하여 데이터 요청 URL을 구성

10행

구성한 url로 getRequestUrl() 함수를 호출해서 받은 응답(utf-8로 디코드됨)을 responseDecode에 저장

15행

서버에서 받은 JSON 형태의 응답 객체를 파이썬 객체로 로드하여 반환

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

4. [CODE 3] 수집 기간 동안 월 단위로 [CODE 2]의 getTourismStatsItem()을 호출해 받은 데이터를 리스트로 묶어 반환

- 매개변수

- nat_cd: 수집 대상 국가의 코드(예: 중국 = 112)
- ed_cd: 수집 데이터의 종류(방한외래관광객 = "E")
- nStartYear: 데이터 수집 시작 연도
- nEndYear: 데이터 수집 끝 연도

- 지역 변수

- jsonResult: 수집한 데이터를 JSON 저장용으로 구성할 딕셔너리의 리스트 객체
- result: 수집한 데이터를 CSV 저장용으로 구성할 리스트 객체
- jsonData: [CODE2]의 getTourismStatsItem()을 호출하여 반환받은 응답 객체
- dataEND: 마지막 데이터의 연월
- natName: 수집한 국가 이름 데이터
- num: 수집한 방문객 수 데이터
- ed: 수집한 출입국 구분 데이터

- 메서드

- getTourismStatsItem(): [CODE2]를 호출하여 응답으로 받은 월 데이터를 반환
- json.dumps(): 객체를 JSON 형식으로 변환

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

3. 함수 설계하기

```
01 def getTourismStatsService(nat_cd, ed_cd, nStartYear, nEndYear):
02     jsonResult = []
03     result = []
04     for year in range(nStartYear, nEndYear+1):
05         for month in range(1, 13):
06             yyyyymm = "{0}{1:0>2}".format(str(year), str(month))
07             jsonData = getTourismStatsItem(yyyyymm, nat_cd, ed_cd) #[CODE 2]
08             if (jsonData['response']['header']['resultMsg'] == 'OK'):
09                 #데이터가 없는 마지막 항목인 경우 -----
10                 if jsonData['response']['body']['items'] == '':
11                     dataEND = "{0}{1:0>2}".format(str(year), str(month-1))
12                     print("데이터 없음.... \n제공되는 통계 데이터는 %s년 %s월까지입니다."
13                           %(str(year), str(month-1)))
14                     break
15                 #jsonData를 출력하여 확인.....
16                 print(json.dumps(jsonData, indent = 4,
17                                   sort_keys = True, ensure_ascii = False))
18
19                 natName = jsonData['response']['body']['items']['item']
20                             ['natKorNm']
21                 natName = natName.replace(' ', '')
22                 num = jsonData['response']['body']['items']['item']['num']
23                 ed = jsonData['response']['body']['items']['item']['ed']
24                 print('[ %s : %s ]' % (natName, yyyyymm, num))
25                 print('-----')
26                 jsonResult.append({'nat_name': natName, 'nat_cd': nat_cd,
27                                   'yyyyymm': yyyyymm, 'visit_cnt': num})
28                 result.append([natName, nat_cd, yyyyymm, num])
29     return (jsonResult, result, natName, ed, dataEND)
```

06행

수집할 연도와 월을 여섯 자리로 맞추어 yyyyymm에 저장

07행

getTourismStatsItem()을 호출해 받은 월 데이터를 jsonData에 저장

08행

응답 데이터가 정상인지 확인

10~13행

['items'] 항목에 값이 없으면 출입국관광통계 데이터가 아직 들어가지 않은 마지막 월이므로 날짜를 dataEND에 저장하고 데이터 수집 작업을 중단

15행

수집한 월 데이터인 jsonData 내용을 확인할 수 있게 파이썬 셸 창에 출력

17~18행

수집한 국가 이름인 ['natKorNm'] 항목의 값에서 띄어쓰기를 제거하고 natName에 저장

19행

수집한 월의 데이터 수인 ['num'] 항목의 값을 num에 저장

20행

수집한 출입국 구분 데이터인 ['ed'] 항목의 값을 ed에 저장

23행

수집한 국가 이름(natName), 국가 코드(nat_cd), 날짜(yyyyymm), 데이터 수(num)를 딕셔너리 자료형으로 구성하여 jsonResult 리스트에 원소로 추가

24행

수집한 국가 이름(natName), 국가 코드(nat_cd), 날짜(yyyyymm), 데이터 수(num)를 result 리스트에 원소로 추가

25행

수집하여 정리한 데이터를 반환

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

4. 전체 프로그램 작성하기

- 파이썬 셸 창에서 [File]-[New File]을 선택해 새 파일 창을 열고 다음의 파이썬 프로그램을 작성

```
import os
import sys
import urllib.request
import datetime
import time
import json
import pandas as pd
```

ServiceKey = "인증키 "

#[CODE 1]

def getRequestUrl(url):

```
    req = urllib.request.Request(url)
    try:
        response = urllib.request.urlopen(req)
        if response.getcode() == 200:
            print("[%s] Url Request Success" % datetime.datetime.now())
            return response.read().decode('utf-8')
    except Exception as e:
        print(e)
        print("[%s] Error for URL : %s" % (datetime.datetime.now(), url))
        return None
```

#[CODE 2]

def getTourismStatsItem(yyyymm, national_code, ed_cd):

```
    service_url = "http://openapi.tour.go.kr/openapi/service/
        EdrcntTourismStatsService/getEdrcntTourismStatsList"
```

```
    parameters = "?_type=json&serviceKey=" + ServiceKey #인증키
    parameters += "&YM=" + yyyymm
    parameters += "&NAT_CD=" + national_code
    parameters += "&ED_CD=" + ed_cd
```

```
    url = service_url + parameters
```

```
    retData = getRequestUrl(url) #[CODE 1]
```

```
    if (retData == None):
```

```
        return None
```

```
    else:
```

```
        return json.loads(retData)
```

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

4. 전체 프로그램 작성하기

```
#[CODE 3]
def getTourismStatsService(nat_cd, ed_cd, nStartYear, nEndYear):
    jsonResult = []
    result = []
    for year in range(nStartYear, nEndYear+1):
        for month in range(1, 13):
            yyyyymm = "{0}{1:0>2}".format(str(year), str(month))
            jsonData = getTourismStatsItem(yyyyymm, nat_cd, ed_cd) #[CODE 2]
            if (jsonData['response']['header']['resultMsg'] == 'OK'):
                #데이터가 없는 마지막 항목인 경우 -----
                if jsonData['response']['body']['items'] == "":
                    dataEND = "{0}{1:0>2}".format(str(year), str(month-1))
                    print("데이터 없음.... Wn 제공되는 통계 데이터는 %s년 %s월까지
                        입니다." %(str(year), str(month-1)))
                    break
            #jsonData를 출력하여 확인.....
            print(json.dumps(jsonData, indent = 4, sort_keys = True, ensure_
                ascii = False))
            natName = jsonData['response']['body']['items']['item']['natKorNm']
            natName = natName.replace(' ', '')
            num = jsonData['response']['body']['items']['item']['num']
            ed = jsonData['response']['body']['items']['item']['ed']
            print(' %s_%s : %s ' %(natName, yyyyymm, num))
            print('-----')
            jsonResult.append({'nat_name': natName, 'nat_cd': nat_cd,
                'yyyyymm': yyyyymm, 'visit_cnt': num})
            result.append([natName, nat_cd, yyyyymm, num])
    return (jsonResult, result, natName, ed, dataEND)
```

```
#[CODE 0]
def main():
    jsonResult = []
    result = []

    print("<< 국내 입국한 외국인의 통계 데이터를 수집합니다. >>")
    nat_cd = input('국가 코드를 입력하세요(중국: 112 / 일본: 130 / 미국: 275)')
    nStartYear = int(input('데이터를 몇 년부터 수집할까요? : '))
    nEndYear = int(input('데이터를 몇 년까지 수집할까요? : '))
    ed_cd = "E" #E : 방한외래관광객, D : 해외 출국
    jsonResult, result, natName, ed, dataEND = getTourismStatsService(nat_
        cd, ed_cd, nStartYear, nEndYear) #[CODE 3]

    #파일저장 1 : json 파일
    with open('./%s_%s_%d_%s.json' % (natName, ed, nStartYear, dataEND),
        'w', encoding = 'utf8') as outfile:
        jsonFile = json.dumps(jsonResult, indent = 4, sort_keys = True,
            ensure_ascii = False)
        outfile.write(jsonFile)

    #파일저장 2 : csv 파일
    columns = ["입국자국가", "국가코드", "입국연월", "입국자 수"]
    result_df = pd.DataFrame(result, columns = columns)
    result_df.to_csv('./%s_%s_%d_%s.csv' % (natName, ed, nStartYear,
        dataEND), index=False, encoding='cp949')

    if __name__ == '__main__':
        main()
```

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

4. 전체 프로그램 작성하기

- 작성한 파이썬 파일은 앞에서 생성한 My_Python 폴더에 openapi_tour.py로 저장
- F5 를 눌러 실행하면 파이썬 셸 창에 값을 입력하라는 메시지가 나타남
- 국가 코드는 '112', 데이터 수집 시작 연도는 '2017', 데이터 수집 마지막 연도는 '2020'을 각각 입력
- 수집된 데이터가 파이썬 셸 창에 출력되는 것을 확인
- 실행이 끝나면 openapi_tour.py 파일이 저장된 위치에 JSON 파일과 CSV 파일이 생성



```
Python 3.7.7 Shell
File Edit Shell Debug Options Window Help
Python 3.7.7 (tags/v3.7.7:d7c567b08f, Mar 10 2020, 10:41:24) [M
SC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more inf
ormation.
>>>
===== RESTART: C:\Users\kmj\My_Python\openapi_tour.p
y =====
<< 국내 입국한 외국인의 통계 데이터를 수집합니다. >>
국가 코드를 입력하세요(중국: 112 / 일본: 130 / 미국: 275) : 112
데이터를 몇 년부터 수집할까요? : 2017
데이터를 몇 년까지 수집할까요? : 2020
[2020-10-29 16:47:21.353606] Url Request Success
{
  "response": {
    "body": {
      "items": {
        "item": {
          "ed": "방한외래관광객",
          "edCd": "E"
        }
      },
      "numOfRows": 10,
      "pageNo": 1,
      "totalCount": 1
    },
    "header": {
      "resultCode": "0000",
      "resultMsg": "OK"
    }
  }
}
[ 중국_202008 : 16275 ]

[2020-10-29 16:47:24.152464] Url Request Success
데이터 없음...
제공되는 통계 데이터는 2020년 8월까지입니다.
>>>
```

그림 5-19 파이썬 셸 창에서 실행 결과 확인

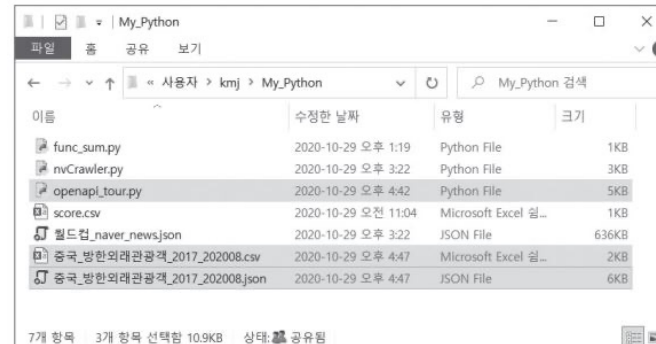


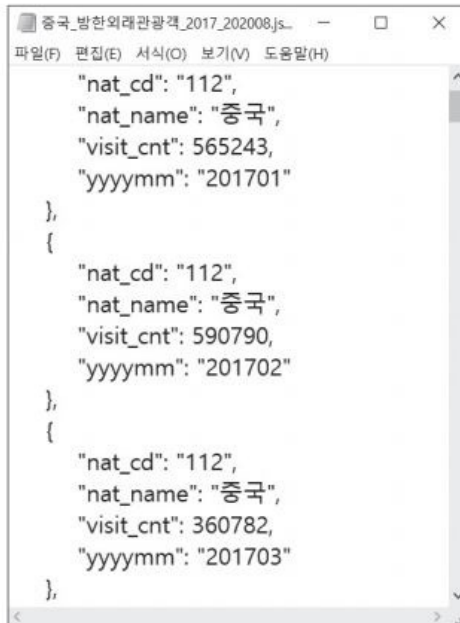
그림 5-20 JSON 파일과 CSV 파일이 생성된 모습

02. 공공데이터 API 기반 크롤링

■ 공공데이터 크롤링

4. 전체 프로그램 작성하기

- 결과 확인하기



```

{
  "nat_cd": "112",
  "nat_name": "중국",
  "visit_cnt": 565243,
  "yyyymm": "201701"
},
{
  "nat_cd": "112",
  "nat_name": "중국",
  "visit_cnt": 590790,
  "yyyymm": "201702"
},
{
  "nat_cd": "112",
  "nat_name": "중국",
  "visit_cnt": 360782,
  "yyyymm": "201703"
}

```

(a) JSON 파일 생성



	A	B	C	D
1	입국자국가	국가코드	입국연월	입국자 수
2	중국	112	201701	565243
3	중국	112	201702	590790
4	중국	112	201703	360782
5	중국	112	201704	227811
6	중국	112	201705	253359
7	중국	112	201706	254930
8	중국	112	201707	281263
9	중국	112	201708	339388
10	중국	112	201709	318682
11	중국	112	201710	345384

(b) CSV 파일 생성

그림 5-21 생성 파일로 실행 결과 확인



데이터 과학 기반의 파이썬 빅데이터 분석

감사합니다.
