



단국대학교
SW 중심대학

전공별 시활용

PART 2. 데이터 입출력



CONTENTS

- 1. 외부파일 읽기
 - 1-1. CSV 파일
 - 1-2. Excel 파일
 - 1-3. JSON 파일
- 2. 웹(web)에서 가져오기
 - 2-1. HTML 웹 페이지에서 표 속성 가져오기
- 3. API 활용하여 데이터 수집하기
- 4. 데이터 저장하기
 - 4-1. CSV 파일로 저장
 - 4-2. JSON 파일로 저장
 - 4-3. Excel 파일로 저장
 - 4-4. 여러 개의 데이터프레임을 하나의 Excel 파일로 저장

1. 외부파일 읽기



1. 외부파일 읽기

• 판다스 데이터 입출력 도구

- 판다스는 다양한 형태의 외부 파일을 읽어와서 데이터프레임으로 변환하는 함수를 제공.
- 어떤 파일이든 판다스 객체인 데이터프레임으로 변환되고 나면, 판다스의 모든 함수와 기능을 자유롭게 사용할 수 있음.
- 반대로, 데이터프레임을 다양한 유형의 파일로 저장할 수도 있음.

1. 외부파일 읽기

- 판다스 데이터 입출력 도구

File Format	Reader	Writer
CSV	read_csv	to_csv
JSON	read_json	to_json
HTML	read_html	to_html
Local clipboard	read_clipboard	to_clipboard
MS Excel	read_excel	to_excel
HDF5 Format	read_hdf	to_hdf
SQL	read_sql	to_sql

[표 2-1] 판다스 데이터 입출력 도구(출처: <http://pandas.pydata.org>)

1. 외부파일 읽기

▪ CSV 파일

- 데이터 값을 쉼표(,)로 구분하고 있다는 의미로 CSV(comma-separated values)라고 부르는 텍스트 파일
- 쉼표(,)로 열을 구분하고 줄바꿈으로 행을 구분함.
- read_csv() 함수에 확장자(.csv)를 포함하여 파일경로(파일명)을 입력하면, CSV 파일을 읽어와서 데이터프레임으로 변환함

CSV 파일 → 데이터프레임: `pandas.read_csv("파일 경로(이름)")`

- read_csv() 함수의 **header 옵션**은 데이터프레임의 열 이름으로 사용할 **행을 지정**함

CSV 파일

〈CSV 파일〉

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

* header 옵션

- '열 이름'이 되는 행을 지정
- `read_csv(file, header=?)`

❶ **header=0** (기본 값: 0행을 열 지정): `df = read_csv(file)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

❷ **header=1** (1행을 열 지정): `df = read_csv(file, header=1)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	0	1	4	7
0	1	2	5	8
1	2	3	6	9

❸ **header=None** (행을 열 지정하지 않음): `df = read_csv(file, header=None)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

[그림 2-1] CSV 파일 읽기 - header 옵션 비교

CSV 파일

〈CSV 파일〉

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

* index_col 옵션

- '행 주소'가 되는 열을 지정
- `read_csv(file, index_col=?)`

① `index_col=False` (인덱스 지정하지 않음)

```
: df = read_csv(file, index_col=False)
```

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

② `index_col='c0'` ('c0'열을 인덱스 지정)

```
: df = read_csv(file, index_col='c0')
```

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c1	c2	c3
0	1	4	7
1	2	5	8
2	3	6	9

[그림 2-2] CSV 파일 읽기 - index_col 옵션 비교

■ CSV 파일

옵션	설명
path	파일의 위치(파일명 포함), URL
sep(또는 delimiter)	텍스트 데이터를 필드별로 구분하는 문자
header	열 이름으로 사용될 행의 번호 (기본값은 0) header가 없고 첫 행부터 데이터가 있는 경우 None으로 지정 가능
index_col	행 인덱스로 사용할 열의 번호 또는 열 이름
names	열 이름으로 사용할 문자열의 리스트
skiprows	처음 몇 줄을 skip할 것인지 설정(숫자 입력) skip하려는 행의 번호를 담은 리스트로 설정 가능(예: [1, 3, 5])
parse_dates	날짜 텍스트를 datetime64로 변환할 것인지 설정(기본값은 False)
skip_footer	마지막 몇 줄을 skip할 것인지 설정(숫자 입력)
encoding	텍스트 인코딩 종류를 지정(예: 'utf-8')

[표 2-2] read_csv() 함수의 옵션

[예제 2-1] ① CSV 파일 미리보기

예제에서 불러올 CSV 파일의 내용을 확인하면, 데이터가 쉼표(,)와 행으로 구분된 것을 확인할 수 있다.

	A	B	C	D
1	c0	c1	c2	c3
2	0	1	4	7
3	1	2	5	8
4	2	3	6	9

〈CSV 파일〉 미리보기

```
1  c0,c1,c2,c3
2  0,1,4,7
3  1,2,5,8
4  2,3,6,9
```

[예제 2-1] ② CSV 파일 읽어오기

header 옵션이 없으면 CSV 파일의 첫 행의 데이터(c0,c1,c2,c3)가 열 이름이 된다.
한편, index_col 옵션을 지정하지 않으면, 행 인덱스는 정수 0, 1, 2가 자동으로 지정된다.
데이터프레임 df4의 경우, index_col='c0' 옵션을 사용하여 'c0' 열이 행 인덱스가 되는 것을 볼 수 있다.

```

3 # 라이브러리 불러오기
4 import pandas as pd
5
6 # 파일 경로(파이썬 파일과 같은 폴더)를 찾고, 변수 file_path에 저장
7 file_path = './read_csv_sample.csv'
8
9 # read_csv() 함수로 데이터프레임 변환. 변수 df1에 저장
10 df1 = pd.read_csv(file_path)
11 print(df1)
12 print('\n')
13
14 # read_csv() 함수로 데이터프레임 변환. 변수 df2에 저장. header=None 옵션
15 df2 = pd.read_csv(file_path, header=None)
16 print(df2)
17 print('\n')
18
19 # read_csv() 함수로 데이터프레임 변환. 변수 df3에 저장. index_col=None 옵션
20 df3 = pd.read_csv(file_path, index_col=None)
21 print(df3)
22 print('\n')
23
24 # read_csv() 함수로 데이터프레임 변환. 변수 df4에 저장. index_col='c0' 옵션
25 df4 = pd.read_csv(file_path, index_col='c0')
26 print(df4)
    
```

〈실행 결과〉 코드 전부 실행

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

	c1	c2	c3
c0			
0	1	4	7
1	2	5	8
2	3	6	9

1. 외부파일 읽기

▪ Excel 파일

- Excel 파일(확장자: .xlsx)의 행과 열은 데이터프레임의 행, 열로 일대일 대응됨
- read_excel() 함수의 사용법은 앞에서 살펴본 read_csv() 함수와 거의 비슷함
- header, index_col 등 대부분의 옵션을 그대로 사용할 수 있음

Excel 파일 → 데이터프레임: `pandas.read_excel("파일 경로(이름) ")`

[예제 2-2] ① Excel 파일 미리보기

Excel 파일은 남북한의 발전량을 정리한 통계자료이며, 파이썬 파일과 같은 폴더에 저장

	A	B	C	D	E	F	AA	AB	AC	AD
1	전력량 (억kWh)	발전 전력별	1990	1991	1992	1993	2014	2015	2016	
2	남한	합계	1,077	1,186	1,310	1,444	5,220	5,281	5,404	
3		수력	64	51	49	60	78	58	66	
4		화력	484	573	696	803	3,427	3,402	3,523	
5		원자력	529	563	565	581	1,564	1,648	1,620	
6		신재생	-	-	-	-	151	173	195	
7	북한	합계	277	263	247	221	216	190	239	
8		수력	156	150	142	133	130	100	128	
9		화력	121	113	105	88	86	90	111	
10		원자력	-	-	-	-	-	-	-	
11										
12										

코드 실행 전 Excel 파일 데이터 추출을 지원하는 openpyxl 라이브러리를 설치해야 함

```
pip install openpyxl
```

[예제 2-2] ② Excel 파일 읽어오기

header 옵션을 추가하지 않은 경우에는 Excel 파일의 첫 행이 열 이름을 구성한다.
한편, header=None 옵션을 사용하면, 정수형 인덱스(0, 1, 2, ...)를 열 이름으로 자동 할당한다.

```
3 import pandas as pd
4
5 # read_excel() 함수로 데이터프레임 변환
6 df1 = pd.read_excel('./남북한발전전력량.xlsx')          # header=0 (default 옵션)
7 df2 = pd.read_excel('./남북한발전전력량.xlsx', header=None) # header = None 옵션
8
9 # 데이터프레임 출력
10 print(df1)
11 print('\n')
12 print(df2)
```

<실행 결과> 코드 전부 실행

전력량 (kWh)	발전	전력별	1990	1991	1992	...	2012	2013	2014	2015	2016
0	남한	합계	1077	1186	1310	...	5096	5171	5220	5281	5404
1	NaN	수력	64	51	49	...	77	84	78	58	66
2	NaN	화력	484	573	696	...	3430	3581	3427	3402	3523
3	NaN	원자력	529	563	565	...	1503	1388	1564	1648	1620
4	NaN	신재생	-	-	-	...	86	118	151	173	195
5	북한	합계	277	263	247	...	215	221	216	190	239
6	NaN	수력	156	150	142	...	135	139	130	100	128
7	NaN	화력	121	113	105	...	80	82	86	90	111
8	NaN	원자력	-	-	-	...	-	-	-	-	-

[9 rows x 29 columns]

	0	1	2	3	4	...	24	25	26	27	28	
0	전력량 (억kWh)	발전	전력별	1990	1991	1992	...	2012	2013	2014	2015	2016
1	남한	합계		1077	1186	1310	...	5096	5171	5220	5281	5404
2	NaN	수력		64	51	49	...	77	84	78	58	66
3	NaN	화력		484	573	696	...	3430	3581	3427	3402	3523
4	NaN	원자력		529	563	565	...	1503	1388	1564	1648	1620
5	NaN	신재생		-	-	-	...	86	118	151	173	195
6	북한	합계		277	263	247	...	215	221	216	190	239
7	NaN	수력		156	150	142	...	135	139	130	100	128
8	NaN	화력		121	113	105	...	80	82	86	90	111
9	NaN	원자력		-	-	-	...	-	-	-	-	-

[10 rows x 29 columns]

1. 외부파일 읽기

■ JSON 파일

- JSON 파일(확장자: .json)은 JavaScript에서 유래한 **데이터 공유를 목적**으로 개발된 특수한 파일형식
- 파이썬 딕셔너리와 비슷하게 'key : value' 구조를 가짐
- read_json() 함수를 사용하여, **JSON 파일**을 **데이터프레임**으로 변환함

	A1	B1
1	가	나
2	다	라

```
{
  "col_A1":{
    "row_1":"가",
    "row_2":"다"
  },
  "col_B1":{
    "row_1":"나",
    "row_2":"라"
  }
}
```

JSON 파일 → 데이터프레임: `pandas.read_json("파일 경로(이름) ")`

[예제 2-3] ① JSON 파일 미리보기

read_json_sample.json

JSON 파일에는 주요 파이썬 패키지의 출시년도, 개발자, 오픈소스 정보가 들어 있음

```
1  {
2      "name": {"pandas": "",
3              "NumPy": "",
4              "matplotlib": ""},
5
6      "year": {"pandas": 2008,
7              "NumPy": 2006,
8              "matplotlib": 2003},
9
10     "developer": {"pandas": "Wes McKinney",
11                  "NumPy": "Travis Oliphant",
12                  "matplotlib": "John D. Hunter"},
13
14     "opensource": {"pandas": "True",
15                   "NumPy": "True",
16                   "matplotlib": "True"}
17 }
```


[예제 2-4] ② JSON 파일 읽어오기

JSON 파일의 "name" 데이터("pandas", "NumPy", "matplotlib")가 인덱스로 지정된다.


```
3 import pandas as pd
4
5 # read_json() 함수로 데이터프레임 변환
6 df = pd.read_json('./read_json_sample.json')
7 print(df)
8 print('\n')
9 print(df.index)
```

〈실행 결과〉 코드 전부 실행

name	year	developer	opensource
NumPy	2006	Travis Oliphant	True
matplotlib	2003	John D. Hunter	True
pandas	2008	Wes Mckinneye	True

Index(['NumPy', 'matplotlib', 'pandas'], dtype='object')

2. 웹(WEB)에서 가져오기



2. 웹(web)에서 가져오기

▪ HTML 웹 페이지에서 표 속성 가져오기

- read_html() 함수는 HTML 웹 페이지에 있는 <table> 태그에서 표 형식의 데이터를 모두 찾아서 데이터프레임으로 변환함

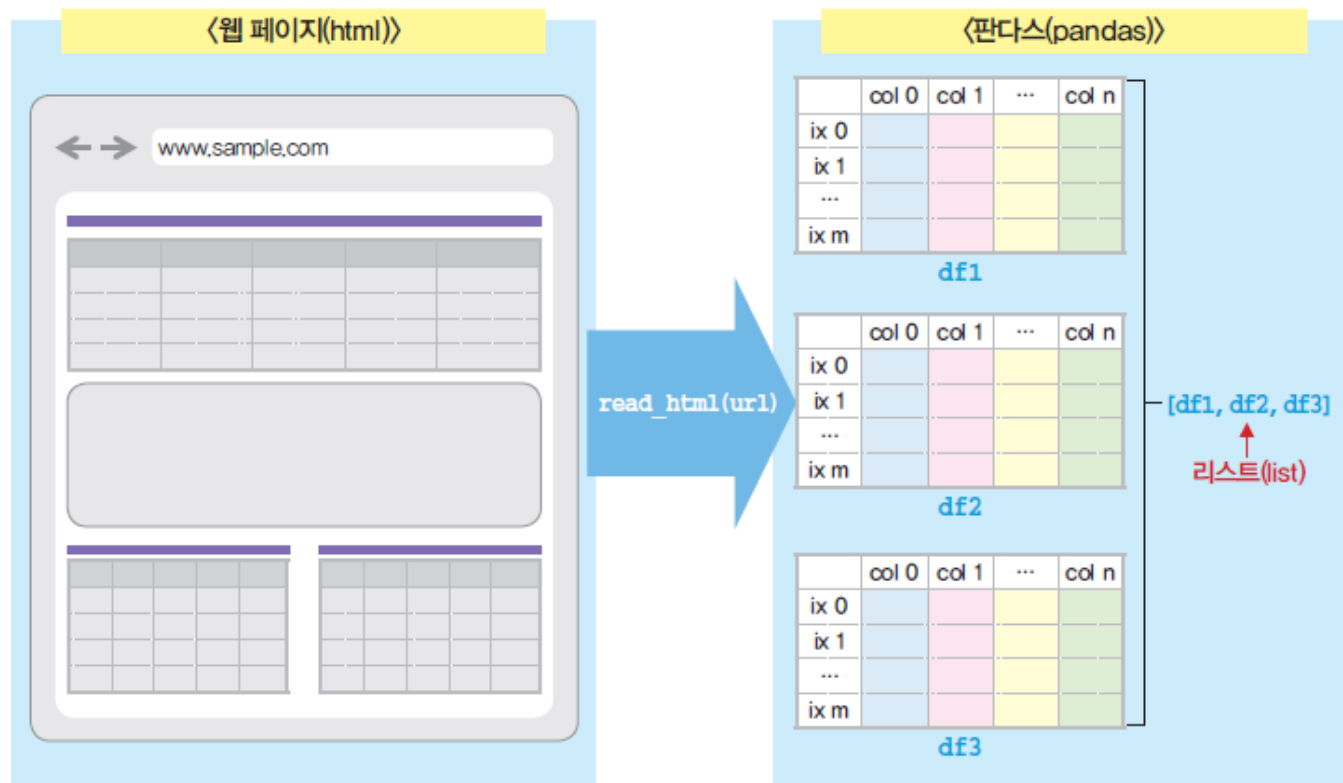
HTML 표 속성 읽기 : `pandas.read_html("웹 주소(URL)" 또는 "HTML 파일 경로(이름)")`

코드 실행 전 지원하는 lxml 라이브러리를 설치해야 함

```
pip install lxml
```

2. 웹(web)에서 가져오기

▪ HTML 웹 페이지에서 표 속성 가져오기



[그림 2-3] HTML 페이지의 표 가져오기

[예제 2-4] 웹에서 표 정보 읽기

표 데이터들은 각각 별도의 데이터프레임으로 변환되기 때문에, 여러 개의 데이터프레임(표)을 원소로 갖는 리스트가 반환된다.

〈실행 결과〉 코드 전부 실행

```
3 import pandas as pd
4
5 # HTML 파일 경로 or 웹 페이지 주소를 url 변수에 저장
6 url = './sample.html'
7
8 # HTML 웹페이지의 표(table)를 가져와서 데이터프레임으로 변환
9 tables = pd.read_html(url)
10
11 # 표(table)의 개수 확인
12 print(len(tables))
13
14 # tables 리스트의 원소를 iteration하면서 각각 화면 출력
15 for i in range(len(tables)):
16     print("tables[%s]" % i)
17     print(tables[i])
18     print('\n')
19
20
21 # 파이썬 패키지 정보가 들어 있는 두 번째 데이터프레임을 선택하여 df 변수에 저장
22 df = tables[1]
23
24 # 'name' 열을 인덱스로 지정
25 df.set_index(['name'], inplace=True)
26 print(df)
```

2

tables[0]

Unnamed: 0	c0	c1	c2	c3
0	0	0	1	4
1	1	1	2	5
2	2	2	3	6

tables[1]

	name	year	developer	opensource
0	NumPy	2006	Travis Oliphant	True
1	matplotlib	2003	John D. Hunter	True
2	pandas	2008	Wes McKinney	True

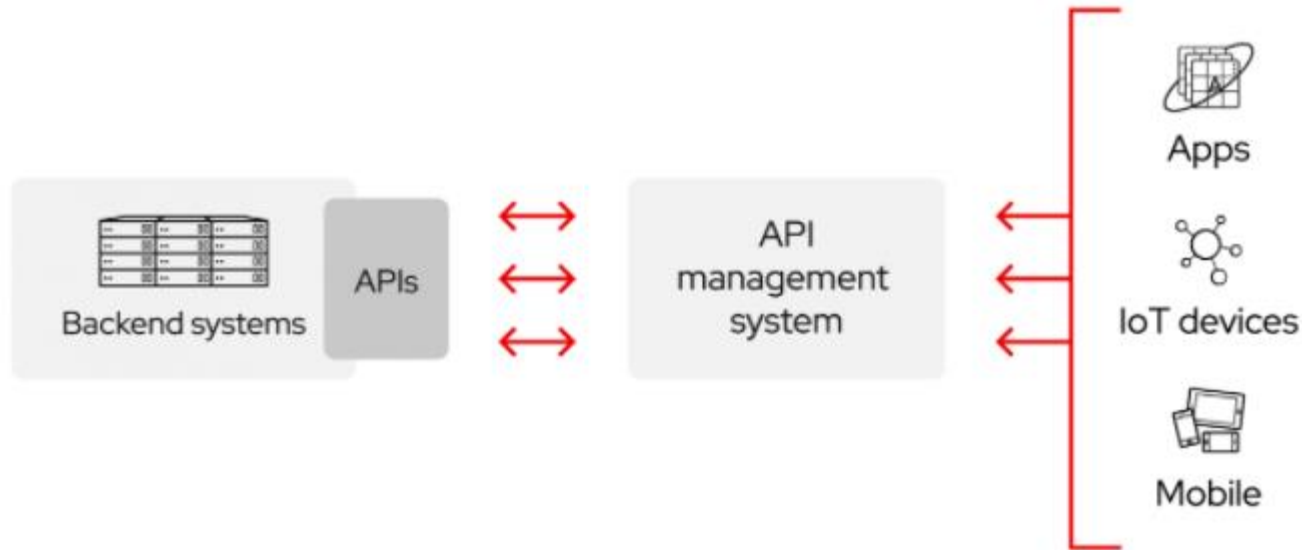
	year	developer	opensource
name			
NumPy	2006	Travis Oliphant	True
matplotlib	2003	John D. Hunter	True
pandas	2008	Wes McKinney	True

3. API 활용하여 데이터 수집하기



■ API(Application Programming Interface)

– 애플리케이션 소프트웨어를 구축하고 통합하기 위한 정의 및 프로토콜 세트

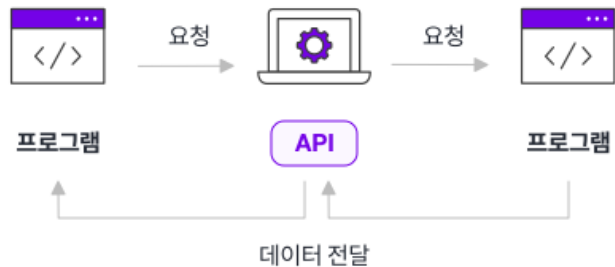


▪ API(Application Programming Interface)

• 점원의 역할



• API의 역할



- 서버와 데이터베이스에 대한 출입구 역할
- 애플리케이션과 기기가 원활하게 통신할 수 있도록 함
- 모든 접속을 표준화

3. API 활용하여 데이터 수집하기

- API를 통해서 수집한 데이터를, 판다스 자료구조로 변환하는 방법 알기
- Google 지오코딩 API 사례
 - 구글 지오코딩 : 장소 이름 또는 주소를 입력하면, 위도와 경도 좌표 정보를 변환해 주는 서비스
 - 서비스를 이용하려면, 사용자 인증 후에 API 키를 발급받아야 함

▪ API Key 받기

- 이전에 큰 개념 먼저 보기



1. API Key는?

API에 접근할 수 있는 사용자 인증 정보

API 제공업체별로 Key를 얻는 방법 유사





이번 시간에는 구글의 수많은 API 중, 지오키링 API 활용법을 알아보는 것



아래 계층 구조를 이해하는 것이 중요



구글 클라우드 플랫폼

내 프로젝트

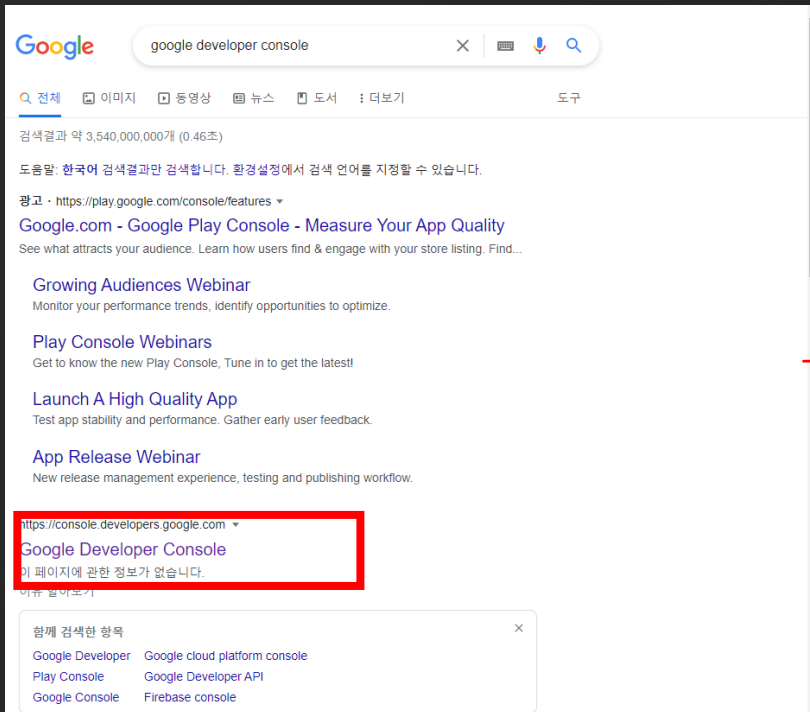
(설정 요소 중 하나가 API 키생성: 내 프로젝트에 접근하기 위한 자격증명)

구글은 훌륭한 API를 제공하고 난 이 중 필요한 API를 내 프로젝트에 붙여 쓰면 된다 그 중 하나가 지오코딩API.

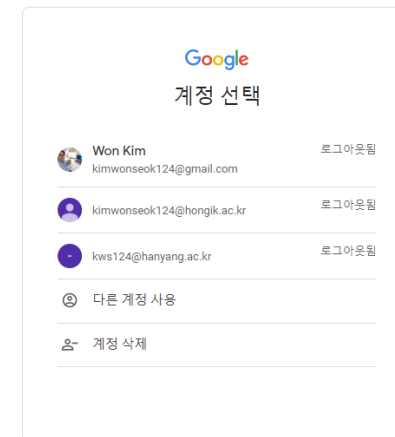




API Key 만들기_Google Developer Console



구글 계정 필요. 없으면 생성

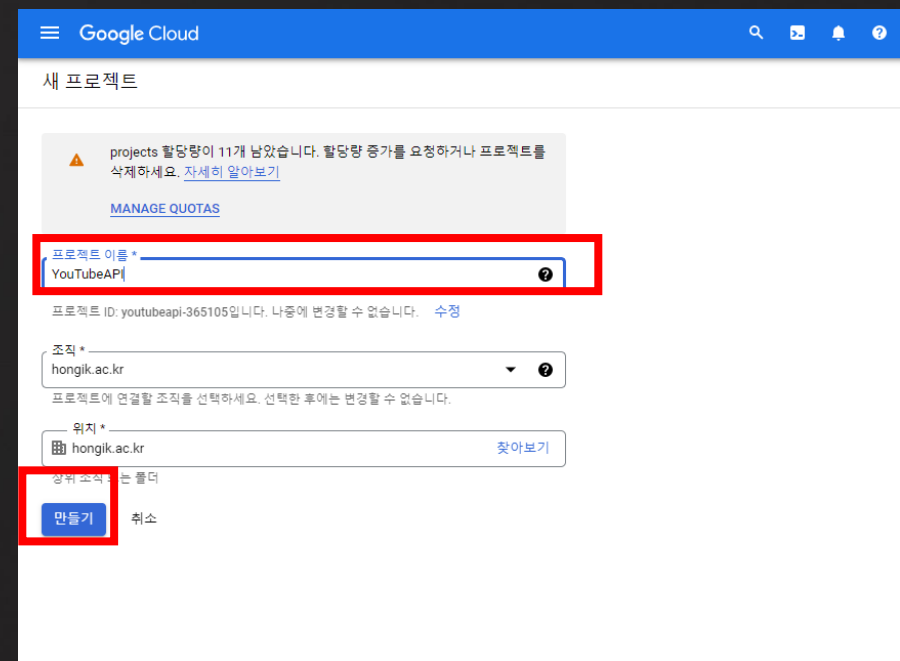
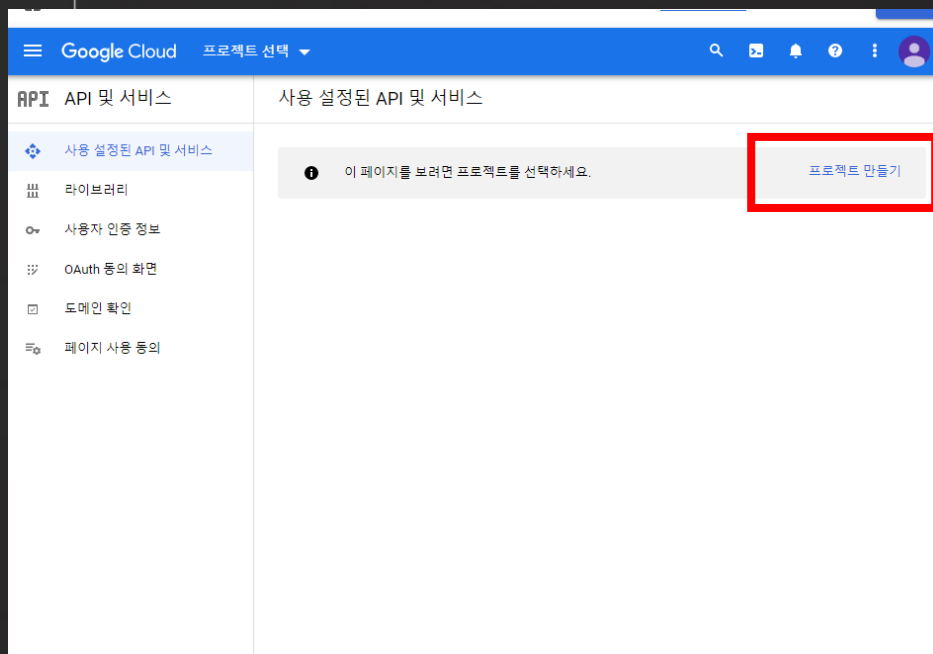


한국어 도움말 개인정보처리방침 약관





API Key 만들기_프로젝트 만들기



API Key 만들기_라이브러리內 API 선택

Google Cloud YouTubeAPI

확인

API 및 서비스

사용 설정된 API 및 서비스

라이브러리

라이브러리 선택

Google Cloud DKU-Google-API

API 라이브러리

geocoding

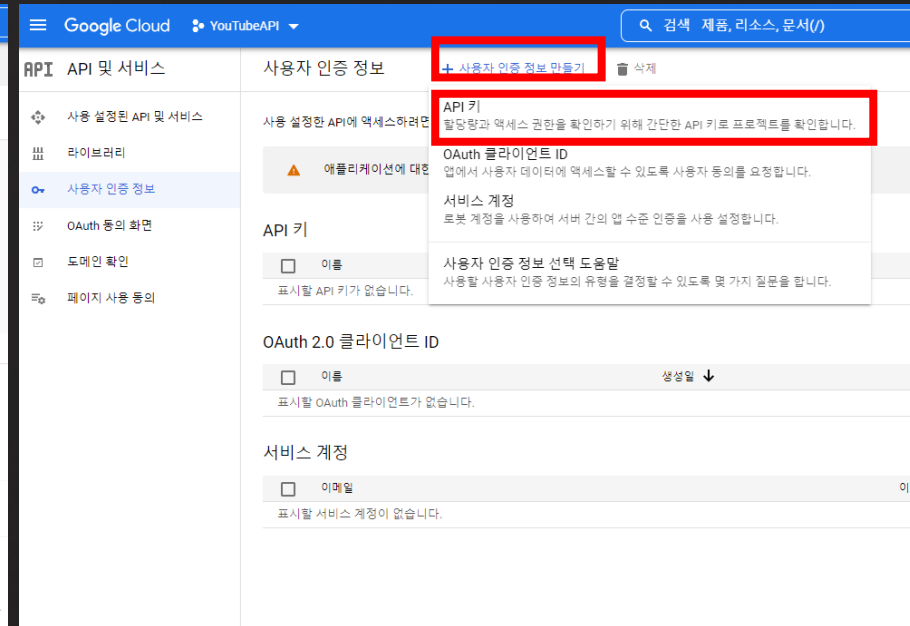
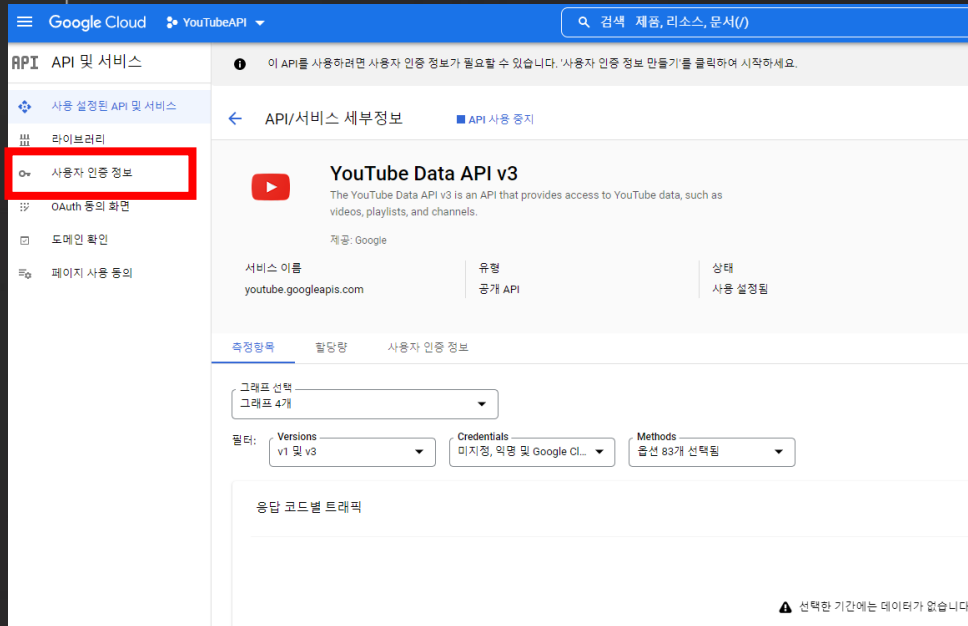
Geocoding API

구글이 제공하는 리소스 중,
Geocoding API 선택,
이후 로드되는 웹페이지에서 '사용' 클릭





API Key 만들기_사용자 인증 정보



이제 특정 프로젝트에서 사용할
특정 API를 매칭 시켰다.

사용자 인증 정보 중
API 키 선택





API Key 만들기_API Key 완성

Google Cloud YouTubeAPI

API 및 서비스 사용자 인증 정보 + 사용자 인증 정보 만들기 삭제

사용 설정된 API 및 서비스

라이브러리

사용자 인증 정보

OAuth 동의 화면

도메인 확인

페이지 사용 동의

사용 설정된 API에 액세스하려면 사용자 인증 정보를 만드세요. 자세히 알아보기

애플리케이션에 대한 정보를 포함하여 OAuth 동의 화면을 구성해야 합니다. 동의 화면 구성

API 키

API 키 생성됨

애플리케이션에서 이 키를 사용하려면 키를 key=API_KEY 매개변수로 전달하세요.

API 키

This key is unrestricted. To prevent unauthorized use, we recommend restricting where and for which APIs it can be used. [Edit API key](#) [Learn more](#)

닫기

Google Cloud YouTubeAPI

API 및 서비스 사용자 인증 정보 + 사용자 인증 정보 만들기 삭제

사용 설정된 API 및 서비스

라이브러리

사용자 인증 정보

OAuth 동의 화면

도메인 확인

페이지 사용 동의

사용 설정된 API에 액세스하려면 사용자 인증 정보를 만드세요. 자세히 알아보기

애플리케이션에 대한 정보를 포함하여 OAuth 동의 화면을 구성해야 합니다. 동의 화면 구성

API 키

이름	생성일	제한사항	작업
API 키 1개	2022. 10. 10.	없음	키 표시

OAuth 2.0 클라이언트 ID

이름	생성일	유형	클라이언트 ID	작업
표시할 OAuth 클라이언트가 없습니다.				

서비스 계정

이메일	이름	작업
표시할 서비스 계정이 없습니다.		

내 프로젝트에 접근하는
코드작성시 필요

필요할 때 '키표시' 클릭 후,
API Key를 복사하여 사용.



■ 기타

- 위 과정 중, 개인정보 입력, 결제 정보 입력, 전화번호 인증 등 필요
- 무료 요금제 이용 가능(할당량 존재)
- 발급한 API키는 언제든지 개발자 콘솔에서 확인가능하므로 따로 저장할 필요 없음. 유출되지 않도록 주의

3. API 활용하여 데이터 수집하기

- googlemaps 라이브러리 설치

```
pip install googlemaps
```

- 코랩의 코드창에서 라이브러리 설치할 때는 !pip

[예제 2-6] ① 지오코딩 API 호출 결과 미리보기

구글 지오코딩 API를 사용하여 “해운대해수욕장” 위치 정보를 확인하면, 딕셔너리 형태를 갖는다.

예를 들면, 'location'을 키로 하는 데이터 중에서 'lat'와 매칭되는 숫자가 “해운대해수욕장”의 위도를 나타낸다. 경도는 'lng' 값과 매칭되는 숫자다.

〈참고〉 지오코딩 API 호출 결과 미리보기(해운대 해수욕장)

```
1 {'location': {'lat': 35.1586975, 'lng': 129.1603842},  
2  'location_type': 'APPROXIMATE',  
3  'viewport': {'northeast': {'lat': 35.1678193, 'lng': 129.1763916},  
4  'southwest': {'lat': 35.1495747, 'lng': 129.1443768}}}
```

[예제 2-6] ② 지오코딩으로 위도, 경도 정보 가져오기

예제코드 9번 라인에 직접 발급받은 API 키를 입력한다. 3개의 장소("서울시청", "국립국악원", "해운대 해수욕장")에 대한 GPS(위도, 경도) 데이터를 2개의 리스트(lat, lng)에 저장한다. DataFrame() 메소드로 데이터프레임을 만들 때, '위도' 열에 lat 리스트를 매칭하고 '경도' 열에는 lng 리스트를 매칭한다. 장소명이 들어 있는 리스트(places)를 행 인덱스로 설정한다.

```
3  ## google 지오코딩 API를 통해 위도, 경도 데이터 가져오기
4
5  # 라이브러리 가져오기
6  import googlemaps
7  import pandas as pd
8
9  my_key = "----발급받은 API 키 입력-----"
10
11 # 구글맵스 객체 생성하기
12 maps = googlemaps.Client(key=my_key) # my key값 입력
13
14 lat = [] # 위도
15 lng = [] # 경도
16
17 # 장소(또는 주소) 리스트
18 places = ["서울시청", "국립국악원", "해운대해수욕장"]
19
20 i=0
21 for place in places:
22     i = i + 1
23     try:
24         print(i, place)
25         # 지오코딩 API 결과값 호출하여 geo_location 변수에 저장
26         geo_location = maps.geocode(place)[0].get('geometry')
27         lat.append(geo_location['location']['lat'])
28         lng.append(geo_location['location']['lng'])
29
30     except:
31         lat.append('')
32         lng.append('')
33         print(i)
```

발급받은 API 키 입력

```
35 # 데이터프레임으로 변환하기
36 df = pd.DataFrame({'위도':lat, '경도':lng}, index=places)
37 print('\n')
38 print(df)
```

〈실행 결과〉 코드 전부 실행

- 1 서울시청
- 2 국립국악원
- 3 해운대해수욕장

	위도	경도
서울시청	37.566295	126.977945
국립국악원	37.477759	127.008304
해운대해수욕장	35.158698	129.160384

3. API 활용하여 데이터 수집하기

■ 머신러닝에 유용한 데이터셋 소스

1. 사이킷런(scikit-learn), 시본(seaborn) 등 파이썬 라이브러리 제공 데이터셋
2. 캐글(Kaggle) : <https://www.kaggle.com/>
3. UCI 머신러닝 저장소 : <https://archive.ics.uci.edu/ml/datasets.php>
4. 공공 데이터
 - (해외) WorldBank, WTO 등 국제기구
 - (국내) 공공데이터 포털, 국가통계포털 등

4. 데이터 저장하기



4. 데이터 저장하기

▪ CSV 파일로 저장

- 판다스 데이터프레임은 2차원 배열로 구조화된 데이터이기 때문에 2차원 구조를 갖는 CSV 파일로 변환할 수 있음.
- 데이터프레임을 CSV 파일로 저장하려면 `to_csv()` 메소드를 적용함
- CSV 파일을 저장할 파일 경로와 파일명(확장자 포함)을 따옴표(" " 또는 ' ') 안에 입력함

CSV 파일로 저장: `DataFrame` 객체.`to_csv("파일 이름 (경로) ")`

[예제 2-7] CSV 파일로 저장

14라인의 `print(df)` 명령에 의해 데이터프레임의 내용이 Ipython 콘솔에 출력된다.

15라인은 `to_csv()` 메소드를 적용하여 파이썬 실행 파일이 위치하고 있는 현재 디렉토리에 파일명을 "df_sample.csv" 저장하는 명령이다. csv 파일을 열어보면 쉼표와 줄바꿈으로 구분되는 2차원 구조가 확인된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
11
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_csv() 메소드를 사용하여 CSV 파일로 내보내기. 파일명은 df_sample.csv로 저장
17 df.to_csv("./df_sample.csv")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - CSV 파일 내용 보기(File: example/part2/df_sample.csv)

```
name,algol,basic,c++
Jerry,A,C,B+
Riah,A+,B,C
Paul,B,B+,C+
```

4. 데이터 저장하기

▪ JSON 파일로 저장

- 데이터프레임을 JSON 파일로 저장하려면 `to_json()` 메소드를 이용
- JSON 파일의 이름(확장자 포함)을 저장하려는 파일 경로와 함께 따옴표 (" " 또는 ' ')안에 입력함

JSON 파일로 저장: `DataFrame 객체.to_json("파일 이름 (경로) ")`

[예제 2-8] JSON 파일로 저장

print(df) 명령에 의해 데이터프레임의 내용이 출력된다.

to_json() 메소드를 이용하여 데이터프레임을 현재 디렉터리에 JSON 파일로 변환하여 저장한다.

JSON 파일을 열어보면 데이터프레임의 행, 열이 JSON 파일의 형식에 맞춰 정리된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_json() 메소드를 사용하여 JSON 파일로 내보내기. 파일명은 df_sample.json로 저장
17 df.to_json("./df_sample.json")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - JSON 파일 내용 보기(File: example/part2/df_sample.json)

```
{
  "algol":{"Jerry":"A","Riah":"A+","Paul":"B"},
  "basic":{"Jerry":"C","Riah":"B","Paul":"B+"},
  "c++":{"Jerry":"B+","Riah":"C","Paul":"C+"}
}
```

4. 데이터 저장하기

▪ Excel 파일로 저장

- 데이터프레임은 Excel 파일과 아주 유사한 구조를 가짐
- 데이터프레임의 행과 열은 Excel 파일의 행과 열로 일대일로 대응됨
- 데이터프레임을 Excel 파일로 저장할 때는 `to_excel()` 메소드를 적용함

Excel 파일로 저장: `DataFrame 객체.to_excel("파일 이름 (경로) ")`

- `to_excel()` 메소드를 사용하려면 `openpyxl` 라이브러리를 사전에 설치해야 함

[예제 2-9] Excel 파일로 저장

IPython 콘솔에 데이터프레임의 내용이 출력된다.

데이터프레임을 판다스 `to_excel()` 메소드를 이용하여 현재 디렉터리에 Excel 파일로 저장한다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
11
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_excel() 메소드를 사용하여 Excel 파일로 내보내기. 파일명은 df_sample.xlsx로 저장
17 df.to_excel("./df_sample.xlsx")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - Excel 파일 내용 보기

	A	B	C	D	E	F
1	name	algol	basic	c++		
2	Jerry	A	C	B+		
3	Riah	A+	B	C		
4	Paul	B	B+	C+		
5						
6						
7						
8						
9						
10						

4. 데이터 저장하기

■ 여러 개의 데이터프레임을 하나의 Excel 파일로 저장

- 판다스 `ExcelWriter()` 함수는 Excel 워크북 객체를 생성함
- 데이터프레임에 `to_excel()` 메소드를 적용할 때, 삽입하려는 워크북 객체 (Excel 파일)를 인자로 전달함
- `sheet_name` 옵션에 Excel 파일의 시트 이름을 입력하여 삽입되는 시트 위치를 지정할 수 있음
- 데이터프레임을 삽입하는 시트 이름을 다르게 설정하면, 같은 Excel 파일의 서로 다른 시트에 여러 데이터프레임을 구분하여 저장함

데이터프레임 여러 개를 Excel 파일로 저장: `pandas.ExcelWriter("파일 이름(경로)")`

[예제 2-10] ExcelWriter() 활용

Print() 명령을 사용하여 두 데이터프레임 df1, df2 의 내용이 Ipython 콘솔에 출력된다.

ExcelWriter() 함수로 생성한 워크북 객체를 writer 변수에 저장하고, "./df_excelwriter.xlsx" 파일 경로에 Excel 파일로 저장된다.

폴더에 저장된 Excel 파일을 실행하여 열어보면 df1 (sheet1에 저장), df2 (sheet2에 저장)에 삽입된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df1, df2에 저장
6 data1 = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7          'algol' : [ "A", "A+", "B"],
8          'basic' : [ "C", "B", "B+"],
9          'c++' : [ "B+", "C", "C+" ]}
10
11 data2 = {'c0':[1,2,3],
12          'c1':[4,5,6],
13          'c2':[7,8,9],
14          'c3':[10,11,12],
15          'c4':[13,14,15]}
16
17 df1 = pd.DataFrame(data1)
18 df1.set_index('name', inplace=True) # name 열을 인덱스로 지정
19 print(df1)
20 print('\n')
21
22 df2 = pd.DataFrame(data2)
23 df2.set_index('c0', inplace=True) # c0 열을 인덱스로 지정
24 print(df2)
25
26 # df1을 'sheet1'으로, df2를 'sheet2'로 저장(Excel 파일명은 "df_excelwriter.xlsx")
27 writer = pd.ExcelWriter("./df_excelwriter.xlsx")
28 df1.to_excel(writer, sheet_name="sheet1")
29 df2.to_excel(writer, sheet_name="sheet2")
30 writer.save()
```

<실행 결과> 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

	c1	c2	c3	c4
c0				
1	4	7	10	13
2	5	8	11	14
3	6	9	12	15

<실행 결과> 코드 전부 실행 ② - Excel 파일 내용 보기

(File: example/part2/df_excelwriter.xlsx)

