공공 데이터 포탈의

[**한국문화관광연구원\_출입국관광통계서비스**] 실습 분석 및 정리

2017301009 김민지

**목차**

[코드]

openapi\_tour\_std.py -3

[코드 분석]

Import – 5

ServiceKey - 7

def getRequestUrl(url) – 8

def getTourismStatsItem(yyyymm, national\_code, ed\_cd) - 11

def getTourismStatsService(nat\_cd, ed\_cd, nStartYear, nEndYear) - 14

def main - 22

**[코드]**

**import** **os**

**import** **sys**

**import** **urllib.request**

**import** **datetime**

**import** **time**

**import** **json**

**import** **pandas** **as** **pd**

ServiceKey="Ody77GLuYeR**%2F**eFqbpduMN2Bi4Cka2fztbgnj6E2Eux1kUhy3e4epR28XKBUaObiqPoVzAizxXMBPXtMyuC9v9Q%3D%3D"

*#[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)

*#[CODE 3]*

**def** getTourismStatsService(nat\_cd, ed\_cd, nStartYear, nEndYear):

jsonResult = []

result = []

natName=''

dataEND = "**{0}{1:0>2}**".format(str(nEndYear), str(12)) *#데이터 끝 초기화*

isDataEnd = 0 *#데이터 끝 확인용 flag 초기화*

**for** year **in** range(nStartYear, nEndYear+1):

**for** month **in** range(1, 13):

**if**(isDataEnd == 1): **break** *#데이터 끝 flag 설정되어있으면 작업 중지.*

yyyymm = "**{0}{1:0>2}**".format(str(year), str(month))

jsonData = getTourismStatsItem(yyyymm, nat\_cd, ed\_cd) *#[CODE 2]*

**if** (jsonData['response']['header']['resultMsg'] == 'OK'):

*# 입력된 범위까지 수집하지 않았지만, 더이상 제공되는 데이터가 없는 마지막 항목인 경우 -------------------*

**if** jsonData['response']['body']['items'] == '':

isDataEnd = 1 *#데이터 끝 flag 설정*

dataEND = "**{0}{1:0>2}**".format(str(year), str(month-1))

print("데이터 없음.... **\n** 제공되는 통계 데이터는 **%s**년 **%s**월까지입니다."%(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, yyyymm, num))

print('----------------------------------------------------------------------')

jsonResult.append({'nat\_name': natName, 'nat\_cd': nat\_cd,'yyyymm': yyyymm, 'visit\_cnt': num})

result.append([natName, nat\_cd, yyyymm, num])

**return** (jsonResult, result, natName, ed, dataEND)

*#[CODE 0]*

**def** main():

jsonResult = []

result = []

natName=''

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? : '))

*#E : 방한외래관광객, D : 해외 출국*

ed\_cd = "E"

jsonResult, result, natName, ed, dataEND =getTourismStatsService(nat\_cd,ed\_cd, nStartYear, nEndYear) *#[CODE 3]*

**if** (natName=='') : *#URL 요청은 성공하였지만, 데이터 제공이 안된 경우*

print('데이터가 전달되지 않았습니다. 공공데이터포털의 서비스 상태를 확인하기 바랍니다.')

**else**:

*#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()

**[코드 분석 - Import]**

**import** **os**

**import** **sys**

**import** **urllib.request**

**import** **datetime**

**import** **time**

**import** **json**

**import** **pandas** **as** **pd**

**import는** 이미 만들어진 파이썬 프로그램 파일, 라이브러리안에 있는 파일등을 사용할 수 있게 해주는 명령어입니다.

**os 모듈은** Operating System의 약자로서 운영체제에서 제공되는 여러 기능을 파이썬에서 수행할 수 있게 해줍니다.

**sys 모듈은** 인터프리터에 의해 사용되거나 유지되는 일부 변수와 인터프리터와 강하게 상호 작용하는 함수에 대한 액세스를 제공합니다.

**urllib.request 모듈은** URL(대부분 HTTP)을 여는 데 도움이 되는 함수와 클래스를 정의합니다.

**datetime 모듈은** 날짜와 시간을 조작하는 클래스를 제공합니다.

**time 모듈은** 다양한 시간 관련 함수를 제공합니다. 특히 이 모듈에 정의된 대부분의 함수는 같은 이름의 플랫폼 C 라이브러리 함수를 호출합니다.

JSON(JavaScript Object Notation) 은 경량 데이터 교환 형식으로 사용되는 JavaScript 구문(ECMA-262 3rd Edition)의 하위 집합입니다. **json은** 표준 라이브러리 marshal과 pickle 모듈 사용자에게 익숙한 API를 제공합니다.

Pandas 모듈은 관계형 또는 레이블이 된 데이터로 쉽고 직관적으로 작업할 수 있도록 설계되었고 빠르고, 유연한 데이터 구조를 제공하는 Python 패키지입니다.

외부 데이터를 입력 받아 Pandas 자료구조로 저장 및 출력하는 기능을 제공합니다.

이번 프로젝트에서는 Pandas 의 DataFrame을 사용했으며 DataFrame이란 가로축과 세로축이 있는 2차원 데이터를 저장하는 자료구조를 의미합니다.

다른 모듈들과 다르게

**import** **pandas** **as** **pd**

as 란 옵션이 사용되었는데 이는 해당 모듈의 기능을 이용할 때

pd 라는 명칭으로 pandas 를 지칭하겠다는 의미입니다.

따라서 현재 코드에서는 pandas 의 기능을 호출할 시 pd 라는 짧은 단어로 호출할 수 있게 구현된 것을 볼 수 있습니다.

result\_df = pd.DataFrame(result, columns = columns)

**[코드 분석 - ServiceKey]**

ServiceKey="Ody77GLuYeR**%2F**eFqbpduMN2Bi4Cka2fztbgnj6E2Eux1kUhy3e4epR28XKBUaObiqPoVzAizxXMBPXtMyuC9v9Q%3D%3D"

ServiceKey 는 다른 변수들과 다르게 함수 내부가 아닌 바깥에 있습니다.

이는 파이썬에서 전역변수를 의미하는 것으로 이를 통해 여러 함수에서 공통적으로 해당 변수를 이용할 수 있습니다.

전역변수를 변경할 시 코드 전체에 영향을 주기 때문에 전역변수를 여러 개 할당하는 것은 지양됩니다.

그러나 이번 프로젝트에서 사용된 전역변수 ServiceKey 는 공공 데이터 포탈에서 주는 키로써, 해당 프로젝트에선 복잡성을 줄일 수 있다는 장점이 있기 때문에 전역변수로 선언한 것으로 분석됩니다.

**[코드 분석 - def getRequestUrl(url)]**

url 접속을 요청하고 응답을 받아서 반환

**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**

**def** getRequestUrl(url):

def getRequestUrl 함수는 매개 변수로 url 을 받습니다.

**try**:

**except** **Exception** **as** e:

try 블록 수행 중 오류가 발생하면 except 블록이 수행된다. 하지만 try 블록에서 오류가 발생하지 않는다면 except 블록은 수행되지 않습니다.

해당 구문이 쓰인 이유는 HTTP 송수신에서 발생 가능한 오류를 처리해주기 위해서입니다.

req = urllib.request.Request(url)

urllib.request는 모든 URL 스킴을 처리하기 위해 같은 Request 인터페이스를 사용합니다.

이 Request 객체로 urlopen을 호출하면 요청된 URL에 대한 응답 객체를 반환합니다.

response = urllib.request.urlopen(req)

urllib.request.urlopen 기능은 문자열 또는 객체일 수 있는 URL url 을 열도록 하는 기능입니다.

urllib.request.urlopen() 메서드로 url 을 열면 HTTPResponse 객체가 만들어집니다.

테이블이(가) 표시된 사진

자동 생성된 설명

**if** response.getcode() == 200:

200 은 옆의 사진과 같이 서버에 요청한 페이지가 성공적으로 처리된 경우를 의미합니다.

print ("[**%s**] Url Request Success" % datetime.datetime.now())

위의 코드는 response 가 200인 경우, 즉 처리가 성공적인 경우 출력하는 메시지 입니다.

[2021-09-25 12:15:09.759002] Url Request Success

위의 메시지는 실제로 프로젝트를 실행하여 얻은 메시지입니다.

메시지 속 날짜 [2021-09-25 12:15:09.759002]는 datetime 모듈의 datetime.now() 함수의 기능으로 현재 날짜와 시간을 문자열로 받아온 것입니다.

**return** response.read().decode('utf-8')

Request 객체로 urlopen을 호출하면 요청된 URL에 대한 응답 객체를 반환하는데 이 응답은 파일류 객체입니다, 따라서 응답에서 .read()를 호출할 수 있습니다.

또한 decode 함수를 이용하여 utf-8 형식으로 디코딩을 해줍니다.

디코딩이란 부호화된 정보를 부호화되기 전 상태로 돌리는 것입니다.

따라서 해당 코드는 utf-8 형식으로 디코딩된 response 를 def getRequestUrl(url) 을 호출한 곳으로 반환해준다는 것을 의미합니다.

print(e)

print("[**%s**] Error for URL : **%s**" % (datetime.datetime.now(), url))

**return** **None**

except 구문 안에 존재하는 코드 입니다.

이는 200이 아닌 코드, 즉 서버에 요청한 페이지가 성공적으로 처리되지 못한 경우를 의미합니다.

때문에

print(e)

e 라는 변수는

**except** **Exception** **as** e:

except 블록이 실행되고 변수 e에 담기는 오류 메시지를 출력하라는 것입니다.

이후 datetime.now() 함수를 이용하여 해당 시간과 함께 에러가 생겼음을 출력해줍니다.

또한 반환되는 값은 None 입니다.

None이란 아무런 값이 존재하지 않는 것을 의미하며, 즉 def getRequestUrl(url) 을 호출한 곳으로 그 어떤 것도 반환하지 않는 것을 의미합니다.

**[코드 분석 - def getTourismStatsItem(yyyymm, national\_code, ed\_cd)]**

출입국관광통계서비스의 오픈 API를 사용하여 데이터 요청 url을 만들고 getRequestUrl(url)을 호출해서 받은 응답 데이터를 반환

**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)

**def** getTourismStatsItem(yyyymm, national\_code, ed\_cd):

getTourismStatsItem 은 매개변수로 yyyymm , national\_code, en\_cd 를 받습니다.

각각의 매개변수는 아래와 같은 이유로 사용됩니다.

yyyymm: 수집할 연월(예: 202003)

nat\_cd: 수집 대상 국가의 코드(예: 중국 = 112)

ed\_cd: 수집할 데이터 종류(방한외래관광객 = "E")

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

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

Parameters 를 += 함수를 이용하여 매개변수로 받은 값들을 추가합니다.

해당 방식을 이용한 이유는 한국문화관광연구원\_출입국관광통계서비스에서 제공하는 가이드의 요청 메시지 예제의 방식을 보면 알 수 있습니다.[[1]](#footnote-1)

테이블이(가) 표시된 사진

자동 생성된 설명

공통 부분인 service\_url 을 제외하면 필수적으로 사용자가 입력해야하는 구간이 존재합니다.

이를 위해 parameters 를 += 함수를 이용해 사용자에게 입력받은 값들을 넣어줍니다.

url 은 REST(URL) 과 같이 만들기 위하여 service\_url 과 parameter 를 합치는 것을 의미하는 구문입니다.

프로젝트에서 실행하면 url 은

<http://openapi.tour.go.kr/openapi/service/EdrcntTourismStatsService/getEdrcntTourismStatsList?_type=json&serviceKey=Ody77GLuYeR%2FeFqbpduMN2Bi4Cka2fztbgnj6E2Eux1kUhy3e4epR28XKBUaObiqPoVzAizxXMBPXtMyuC9v9Q%3D%3D&YM=201701&NAT_CD=112&ED_CD=E>

과 같이 출력됩니다.

retData = getRequestUrl(url)

retData 는 getRequestUrl() 함수를 호출해서 받은 응답을 저장하는 변수입니다.

**if** (retData == **None**):

**return** **None**

**else**:

**return** json.loads(retData)

def getRequestUrl 함수는 성공하면 utf-8 로 디코딩된 응답을 반환하지만 실패시 None, 즉 어떤 것도 반환하지 않습니다.

이를 이용해 retData 가 None 일 경우 def getTourismStatsItem 함수 또한 마찬가지로 그 어떤 것도 들어있지 않은 것을 반환합니다.

하지만 만약 None 이 아닌 경우에는 utf-8 로 디코딩된 응답은 json.loads() 메서드를 사용하여 문자열을 Python 타입으로 변경하여 def getTourismStatsItem 를 호출한 곳으로 반환합니다.

**[코드 분석 - def getTourismStatsService(nat\_cd, ed\_cd, nStartYear, nEndYear)**

수집 기간 동안 월 단위로 getTourismStatsItem()을 호출해 받은 데이터를 리스트로 묶어 반환

**def** getTourismStatsService(nat\_cd, ed\_cd, nStartYear, nEndYear):

jsonResult = []

result = []

natName=''

dataEND = "**{0}{1:0>2}**".format(str(nEndYear), str(12)) *#데이터 끝 초기화*

isDataEnd = 0 *#데이터 끝 확인용 flag 초기화*

**for** year **in** range(nStartYear, nEndYear+1):

**for** month **in** range(1, 13):

**if**(isDataEnd == 1): **break** *#데이터 끝 flag 설정되어있으면 작업 중지.*

yyyymm = "**{0}{1:0>2}**".format(str(year), str(month))

jsonData = getTourismStatsItem(yyyymm, nat\_cd, ed\_cd) *#[CODE 2]*

**if** (jsonData['response']['header']['resultMsg'] == 'OK'):

*# 입력된 범위까지 수집하지 않았지만, 더이상 제공되는 데이터가 없는 마지막 항목인 경우 -------------------*

**if** jsonData['response']['body']['items'] == '':

isDataEnd = 1 *#데이터 끝 flag 설정*

dataEND = "**{0}{1:0>2}**".format(str(year), str(month-1))

print("데이터 없음.... **\n** 제공되는 통계 데이터는 **%s**년 **%s**월까지입니다."%(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, yyyymm, num))

print('----------------------------------------------------------------------')

jsonResult.append({'nat\_name': natName, 'nat\_cd': nat\_cd,'yyyymm': yyyymm, 'visit\_cnt': num})

result.append([natName, nat\_cd, yyyymm, num])

**return** (jsonResult, result, natName, ed, dataEND)

**def** getTourismStatsService(nat\_cd, ed\_cd, nStartYear, nEndYear):

def getTourismStatsService 은 매개변수로 nat\_cd, en\_cd, nStartYear, nEndYear 을 매개변수로 받습니다.

각 매개변수는 아래와 같은 의미를 지닙니다.

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

jsonResult = []

result = []

natName=''

dataEND = "**{0}{1:0>2}**".format(str(nEndYear), str(12)) *#데이터 끝 초기화*

isDataEnd = 0 *#데이터 끝 확인용 flag 초기화*

각 변수들을 초기화하여 선언하는 코드들입니다.

Format 함수를 사용했는데 {0}{1:0>2} 를 분석하면

{0} 에는 str(nEndYear) 이 들어가며

{1:0>2} 에는 str(12)가 들어가는 것을 의미합니다.

이 때 ‘>’ 는 오른쪽 정렬을 의미하며 2 는 두자리로 표현할 것임을 의미합니다.

또한 0 은 전체자리에서 빈공간이 발생할 경우 0으로 채운다는 의미입니다.

단 dataEND 에서는 12 가 두자리 이므로 0 이 채워지지 않았습니다.

예를 들어 아래와 같이 사용된다면

dataEND = "**{0}{1:5>5}**".format(str(nEndYear), str(12))

202055512

위와 같이 총 5글자에서 12가 들어가고 남은 세자리를 5로 채워줍니다

**for** year **in** range(nStartYear, nEndYear+1):

반복문 for 이 사용되었습니다.

해당 코드는 nStartYear 에서 (nEndYear+1)-1 까지 반복할 것을 의미합니다.

year 은 nStartYear 부터 시작합니다.

**for** month **in** range(1, 13):

반복문 for 이 사용되었습니다.

month 는 1월부터 12월까지만 존재하기에 따로 변수를 할당하지 않고 상수 1과 13을 이용한 것으로 보입니다.

따라서 month 는 1부터 시작하여 12까지 반복하게 됩니다.

**if**(isDataEnd == 1): **break**

만약 jsonData 의 마지막에 도달하게 되면 1로 isDataEnd 변수의 값을 변경하여 반복문을 중단하기 위해 사용된 것으로 보입니다.

yyyymm = "**{0}{1:0>2}**".format(str(year), str(month))

jsonData = getTourismStatsItem(yyyymm, nat\_cd, ed\_cd)

yyyymm 또한 dataEND 와 같이 format 함수를 이용하여 값을 저장합니다.

yyyymm 이라 지은 이유는 {0} 에 들어가는 str(year) 이 4글자인 것과 {1:0>2} 에 들어갈 달이 최대 두글자이기 때문이라 봅니다.

특히 1달부터 9달은 두글자가 되기 위해 0이 앞에 들어가야 합니다.

따라서 {1:0>2} 은 공백 시 0을 넣어주는 오른쪽 정렬의 두글자를 생성하기 위해 사용한 코드입니다.

jsonData 는 getTourismStatsItem 을 년도날짜, 수집한 국가 이름 데이터, 수집 데이터의 종류라는 값을 인자로 주고 호출하여 반환 받은 월 데이터 값을 저장하는 변수입니다.

**if** (jsonData['response']['header']['resultMsg'] == 'OK'):

해당 코드를 이해하려면 jsonData 의 형식을 보는 것이 필요합니다.

{

**"response"**: {

**"body"**: {

**"items"**: {

**"item"**: {

**"ed"**: "방한외래관광객",

**"edCd"**: "E",

**"natCd"**: 112,

**"natKorNm"**: "중 국",

**"num"**: 565243,

**"rnum"**: 1,

**"ym"**: 201701

}

},

**"numOfRows"**: 10,

**"pageNo"**: 1,

**"totalCount"**: 1

},

**"header"**: {

**"resultCode"**: "0000",

**"resultMsg"**: "OK"

}

}

}

프로젝트를 실제로 실행하고 얻은 jsonData 의 모습입니다.

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명[[2]](#footnote-2)

**"resultMsg"**: "OK" 처럼 jsonData['response']['header']['resultMsg'] 가 OK 인 경우에만 조건문을 실행한다는 의미입니다.

**if** jsonData['response']['body']['items'] == '':

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

해당 조건문에서는 아래의 코드를 실행합니다.

isDataEnd = 1

dataEND = "**{0}{1:0>2}**".format(str(year), str(month-1))

print("데이터 없음.... **\n** 제공되는 통계 데이터는 **%s**년 **%s**월까지입니다."%(str(year), str(month-1)))

**break**

데이터의 끝임을 알리기 위해 플래그의 의미를 지닌 isDataEnd 를 1로 변경합니다.

또한 dataEND 를 format 함수를 이용해 값을 저장합니다.

또한 해당 년도와 달-1 을 출력하며 반복문을 빠져나옵니다.

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, yyyymm, num))

print('----------------------------------------------------------------------')

jsonResult.append({'nat\_name': natName, 'nat\_cd': nat\_cd,'yyyymm': yyyymm, 'visit\_cnt': num})

result.append([natName, nat\_cd, yyyymm, num])

위의 코드는

jsonData['response']['header']['resultMsg'] == 'OK' 이면서

jsonData['response']['body']['items'] == '' 가 아닌 코드 인 경우 실행됩니다.

print (json.dumps(jsonData, indent=4, sort\_keys=**True**, ensure\_ascii=**False**))

json.dumps() 메서드를 써서 Python Object를 문자열로 변환하여 출력합니다.

양의 정수 indent를 사용하면, 해당 값만큼의 스페이스로 들여쓰기합니다.

sort\_keys가 참이면 (기본값: False), 딕셔너리의 출력이 키로 정렬됩니다.

ensure\_ascii가 참(기본값)이면, 출력에서 모든 비 ASCII 문자가 이스케이프 됩니다.

ensure\_ascii가 거짓이면, 그 문자들은 있는 그대로 출력됩니다.

natName = jsonData['response']['body']['items']['item']['natKorNm']

natName = natName.replace(' ', '')

num = jsonData['response']['body']['items']['item']['num']

ed = jsonData['response']['body']['items']['item']['ed']

수집한 국가 이름인 ['natKorNm'] 항목의 값에서 띄어쓰기를 제거하고 natName 에 저장합니다.

수집한 월의 데이터 수인 ['num'] 항목의 값을 num에 저장합니다.

수집한 출입국 구분 데이터인 ['ed'] 항목의 값을 ed에 저장합니다.

print('[ **%s**\_**%s** : **%s** ]' %(natName, yyyymm, num))

print('----------------------------------------------------------------------')

natName, yyyymm,num 을 출력합니다.

각각 수집한 국가 이름 데이터,년도날짜,수집한 방문객 수 데이터를 의미합니다.

jsonResult.append({'nat\_name': natName, 'nat\_cd': nat\_cd,'yyyymm': yyyymm, 'visit\_cnt': num})

result.append([natName, nat\_cd, yyyymm, num])

수집한 국가 이름(natName), 국가 코드(nat\_cd), 날짜(yyyymm), 데이터 수(num) 를 딕셔너리 자료형으로 구성하여 jsonResult 리스트에 원소로 추가하는 코드입니다.

또한 수집한 국가 이름(natName), 국가 코드(nat\_cd), 날짜(yyyymm), 데이터 수(num)를 result 리스트에 원소로 추가합니다.

**return** (jsonResult, result, natName, ed, dataEND)

이제 getTourismStatsService 를 호출한 곳으로 jsonResult, result, natName,ed,dataEND 를 반환합니다.

**[코드 분석 - def main]**

**def** main():

jsonResult = []

result = []

natName=''

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? : '))

*#E : 방한외래관광객, D : 해외 출국*

ed\_cd = "E"

jsonResult, result, natName, ed, dataEND =getTourismStatsService(nat\_cd,ed\_cd, nStartYear, nEndYear) *#[CODE 3]*

**if** (natName=='') : *#URL 요청은 성공하였지만, 데이터 제공이 안된 경우*

print('데이터가 전달되지 않았습니다. 공공데이터포털의 서비스 상태를 확인하기 바랍니다.')

**else**:

*#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')

jsonResult = []

result = []

natName=''

각 변수를 초기화해줍니다.

각 변수는 아래와 같은 값들을 저장하기 위해 선언합니다.

jsonResult: 수집한 데이터를 JSON 저장용으로 구성할 딕셔너리의 리스트 객체

result: 수집한 데이터를 CSV 저장용으로 구성할 리스트 객체

natName: 수집한 국가 이름 데이터

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? : '))

*#E : 방한외래관광객, D : 해외 출국*

ed\_cd = "E"

각각 사용자의 입력을 받아 값을 저장하는 변수입니다.

jsonResult, result, natName, ed, dataEND =getTourismStatsService(nat\_cd,ed\_cd, nStartYear, nEndYear)

getTourismStatsService 에 국가코드, 방한외래관광객,시작년도,종료년도를 인자로 제공하여 반환 받은 값들을 각각 저장합니다.

**if** (natName=='') :

print('데이터가 전달되지 않았습니다. 공공데이터포털의 서비스 상태를 확인하기 바랍니다.')

해당 코드는 URL 요청은 성공하였지만, 데이터 제공이 안된 경우 실행되는 조건문입니다.

에러 상황에 대한 예외처리를 위해 사용된 것으로 분석됩니다.

**else**:

*#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')

위의 코드는 URL 요청이 성공하였으며 데이터 제공이 성공적인 경우 실행됩니다.

**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)

.json 이라고 적혀있듯 현재까지 얻은 값들을 저장한다는 뜻입니다.

이 때 저장되는 파일의 이름을 검색한 나라,방한외래관광객,시작년도,종료년도달 을 합쳐 저장합니다.



위의 사진과 같이 저장됩니다.

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')

result\_df 는 columns 를 입국자 국가, 국가코드, 입국연월, 입국자 수를 컬럼으로 가진 딕셔너리가 됩니다.

이를 json 에서 했듯이 이름을 저장하며 to\_csv() 함수를 이용하여 아래의 사진과 같이 csv 파일로 저장합니다.



1. 한국문화관광연구원 출입국관광통계서비스 활용가이드\_v1.1.docx [↑](#footnote-ref-1)
2. 한국문화관광연구원 출입국관광통계서비스 활용가이드\_v1.1.docx [↑](#footnote-ref-2)