[JSON] (JavaScript Object Notation)

-속성-값 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷.
-비동기 AJAX을 위해, 넓게는 XML(AJAX가 사용)을 대처하는 주요 데이터 포맷.
-본래는 자바스크립트 언어로부터 파생되어 자바스크립트의 구문 형식을 따르지만 언어 독립형 데이터 포맷. 즉, 프로그래밍 언어나 플랫폼에 독립적, C, C++, C#, JAVA, 자바스크립트, 펄, 파이썬 등 많은 프로그래밍 언어에서 쉽게 이용할 수 있음.1

JSON 객체 기본형태 { "속성명1": "value1" , "속성명2": "value2" }
JSON 객체 배열 기본형태

```
[   {         "name"     :  "김준서"
            , "addr"     :  "인천시"
       },
     {        "name"     :  "이승희"
            , "addr"     :  "서울"
       }
]
```

```
[   {    "속성명1":  "value1"
         ,"속성명2":  "value2"
     },
     {    "속성명1":  "value1"
         ,"속성명2":  "value2"
     }
]
```

JSON 예

```
{
    "response":
        {
          "header":
                   {"resultCode":"0000","resultMsg":"OK"},
          "body":
                   {"items":
                            {"item":
                                [
                                    {
                                      "addrCd":1111,
                                      "csForCnt":39559,
                                      "csNatCnt":52082,
                                      "gungu":"종로구",
                                      "resNm":"창덕궁",
                                      "rnum":1,
                                      "sido":"서울특별시",
                                      "ym":201612},
                                      {
                                      "addrCd":1111,
                                      "csForCnt":142110,
                                      "csNatCnt":151317,
```

```
                                    "gungu":"종로구",
                                    "resNm":"경복궁",
                                    "rnum":2,
                                    "sido":"서울특별시",
                                    "ym":201612
                                    }
                                ]
                            },
                            "numOfRows":100,
                            "pageNo":1,
                            "totalCount":12
                        }
        }
}
```

딕셔너리를 JSON으로(json.dumps()) **json으로 바꾸면--> 모두 문자열로 처리

ex01_to_json.py

```
import json

inst = {1:'guitar', 2:'violin', 3:'piano'}
to_json = json.dumps(inst)
print(type(to_json), to_json)
```

실행결과 :

```
<class 'str'> {"1": "guitar", "2": "violin", "3": "piano"}
```

ex01_to_json.py 계속

```
to_json = json.dumps(inst, indent=4, ensure_ascii=False)
                            #ndent=4 들여쓰기,  ensure_ascii=False 한글처리
print(to_json)
```

실행결과 :

```
{
    "1": "guitar",
    "2": "violin",
    "3": "piano"
}
```

딕셔너리 리스트를 JSON으로(json.dumps())

ex01_to_json.py 계속

```
import json

inst = [ { 1:'guitar', 2:'violin', 3:'piano'},
         {1: 'seoul', 2: 'busan', 3: 'incheon'} ]

to_json = json.dumps(inst)
print(type(to_json),  to_json)
```

실행결과 :

```
<class 'str'> [{"1": "guitar", "2": "violin", "3": "piano"}, {"1": "seoul", "2": "busan", "3":
```

"incheon"}]

ex01_to_json.py 계속

```
to_json = json.dumps(inst, indent=4, ensure_ascii=False)
                           #ndent=4 들여쓰기,  ensure_ascii=False 한글처리
print(to_json)
```

실행결과 :

```
[
    {
        "1": "guitar",
        "2": "violin",
        "3": "piano"
    },
    {
        "1": "seoul",
        "2": "busan",
        "3": "incheon"
    }
]
```

JSON을 딕셔너리로(json.loads())

ex02_from_json.py

```
import json
to_json =  '{"1": "guitar", "2": "violin", "3": "piano"}'  #json 문자열

from_json = json.loads(to_json)
print(type(from_json),   from_json)

print(from_json["3"]) # piano
```

실행결과 :

```
<class 'dict'> {'1': 'guitar', '2': 'violin', '3': 'piano'}
piano
```

JSON 배열을 딕셔너리 리스트로(json.loads())

ex02_from_json.py 계속

```
import json
to_json = '[{"1":"guitar","2":"violin","3":"piano"}, {"1":"seoul","2":"busan","3": "incheon"}]'

from_json = json.loads(to_json)
print(type(from_json),   from_json)


print(from_json[0]["2"])  # violin
```

실행결과 : <class 'list'> [{'1': 'guitar', '2': 'violin', '3': 'piano'}, {'1': 'seoul', '2': 'busan', '3': 'incheon'}]

Violin