



# JSON檔案讀取

Data Processing JSON

# JSON是什麼？

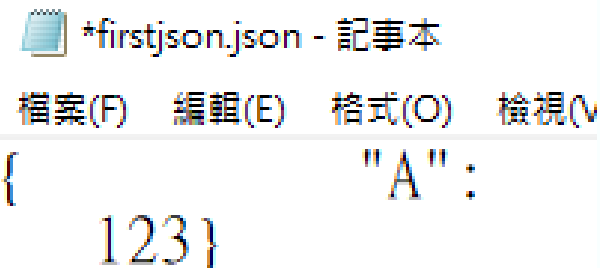
- ◆ JSON的全名是JavaScript Object Notation，是一種使用結構化方式呈現JavaScript物件的格式
- ◆ JSON最初是為了web應用而開發，不過目前JavaScript、JAVA、C#使用JSON的情況較多，PHP主要還是使用XML
- ◆ C、C++、JAVA、Python...等都支援JSON

{JSON}

{JSON}

# JSON檔案

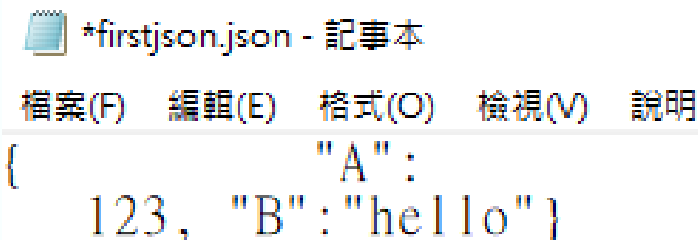
- ◆ JSON由「key(鍵)」和「value(值)」組成，資料型態不限(字串、整數、浮點數、布林、陣列...等)
- ◆ JSON不受換行符號限制，只要資料最外層有大括號包圍即可
- ◆ 若有多筆資料時，以逗號做區隔
- ◆ JSON檔案沒有註解



\*firstjson.json - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V)

```
{ "A":  
  123 }
```



\*firstjson.json - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
{ "A":  
  123, "B": "hello" }
```

# JSON結構

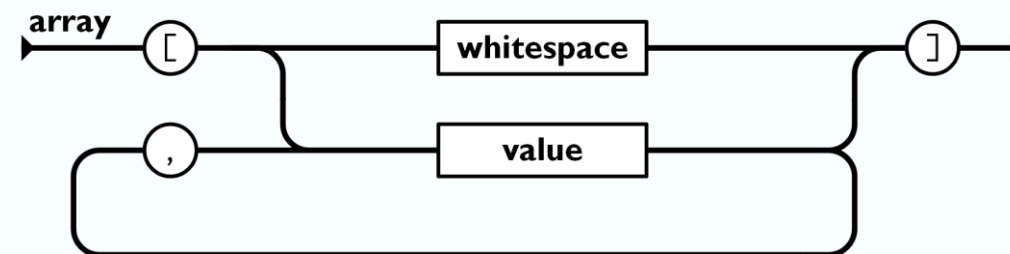
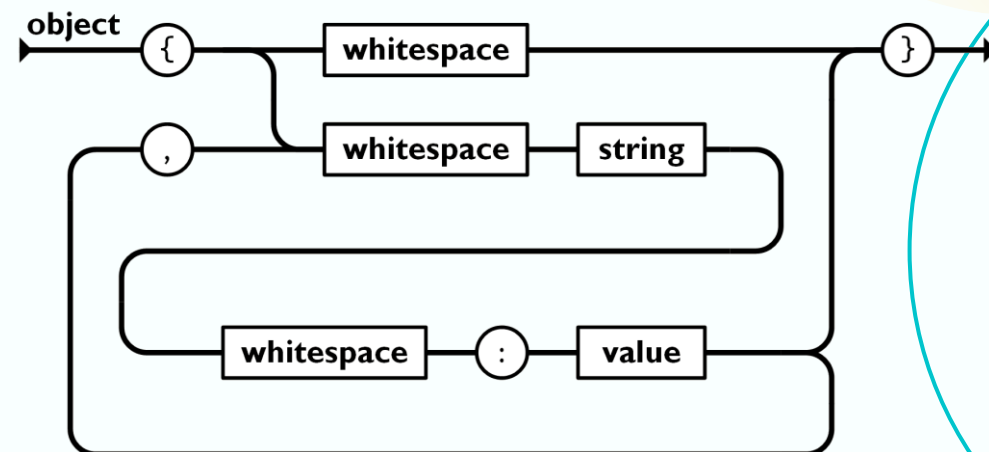
## ◆ JSON主要有兩種資料結構：

### ◆ JSON物件(object)

- ◆ 用大括號{ }表示
- ◆ 大括號內的對象是沒有順序的key:value集合
- ◆ 每個對象以逗號做區隔

### ◆ JSON陣列(array)

- ◆ 用中括號[ ]表示
- ◆ 中括號內的對象是有順序集合
- ◆ 每個對象以逗號做區隔



# JSON物件

- ◆ JSON物件是以「key:value」方式儲存
- ◆ 所有資料的最外層由大括號{ }所包覆
- ◆ 和python的dict不一樣的是，JSON的key一定是字串，且是雙引號；value可以是字串、數值、布林、陣列、物件或null值

```
{  
  "result": 1,  
  "message": {  
    "78": {  
      "id": "78",  
      "username": "Jenn",  
      "status": "0"  
    },  
    "40": {  
      "id": "40",  
      "username": "jhondoe",  
      "status": "1"  
    },  
    "121": {  
      "id": "121",  
      "username": "April",  
      "status": "1",  
    },  
    "132": {  
      "id": "132",  
      "username": "jhondoe",  
      "status": "1",  
    }  
  }  
}
```

# JSON陣列

- ◆ JSON陣列是由一系列的value所組成
- ◆ 所有資料的最外層由中括號[ ] 所包覆
- ◆ 陣列的值可以是字串、數值、布林、陣列或null值

stations.json - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
[
  {
    "Station": "土城站",
    "Destination": "頂埔站",
    "UpdateTime": "2022-07-16T15:39:58.897"
  },
  {
    "Station": "大安站",
    "Destination": "象山站",
    "UpdateTime": "2022-07-16T15:39:33.37"
  },
  {
    "Station": "中山站",
    "Destination": "松山站",
    "UpdateTime": "2022-07-16T15:40:11.093"
  },
  {
    "Station": "中山國中站",
    "Destination": "南港展覽館站",
    "UpdateTime": "2022-07-16T15:40:05"
  },
  {
    "Station": "內湖站",
    "Destination": "南港展覽館站",
    "UpdateTime": "2022-07-16T15:40:05"
  },
  {
    "Station": "木柵站",
    "Destination": "南港展覽館站",
    "UpdateTime": "2022-07-16T15:40:05"
  }
]
```

# JSON常用方法

方法	說明
load()	針對JSON檔案的操作，使用load()方法讀取檔案做成物件，並回傳成dict型態
loads()	針對JSON字串的操作，讀取JSON檔案後做成字串，並回傳成dict型態
dump()	針對JSON檔案的操作，使用dump()方法將dict型態資料寫成JSON檔案
dumps()	針對JSON字串的操作，將dict型態資料寫成JSON格式的檔案

# JSON物件讀取 – load()

- ◆ 使用load()函數所讀出的JSON檔案格式為dict

```
1  import json
2
3  # 無中文的JSON檔案
4  fp = "../JSON/jsonobject_en.json"
5
6  ▼ with open(fp) as file_en:
7      data = json.load(file_en)
8
9  print(data)          # {'A': 'apple', 'B': 'book', 'C': 'cat'}
10 print(type(data))    # <class 'dict'>
11
12
13 # 有中文的JSON檔案
14 fp = "../JSON/jsonobject_ch.json"
15
16 ▼ with open(fp, encoding="utf8") as file_en:
17     data = json.load(file_en)
18
19 print(data)          # {'A': '蘋果', 'B': '書', 'C': '貓'}
20 print(type(data))    # <class 'dict'>
```



# JSON物件讀取 – loads()

- ◆ 和load()不同之處在於，要使用loads()前需要先用read()函式讀取JSON檔案，讀出後的格式為字串，然後再用loads()將字串轉成dict

```
1 import json
2
3 # 無中文的JSON檔案
4 fp = "../JSON/jsonobject_en.json"
5
6 ▼ with open(fp) as file_en:
7     # 使用read()函數讀取JSON檔案並轉成字串
8     data = file_en.read()
9
10 # 將型態為字串的data用loads()函數轉成dict型態
11 file = json.loads(data)
12 print(file)          # {'A': 'apple', 'B': 'book', 'C': 'cat'}
13 print(type(file))    # <class 'dict'>
```

```
16 # 有中文的JSON檔案
17 fp = "../JSON/jsonobject_ch.json"
18
19 ▼ with open(fp, encoding="utf8") as file_ch:
20     # 使用read()函數讀取JSON檔案並轉成字串
21     data = file_ch.read()
22
23 # 將型態為字串的data用loads()函數轉成dict型態
24 file = json.loads(data)
25 print(file)          # {'A': '蘋果', 'B': '書', 'C': '貓'}
26 print(type(file))    # <class 'dict'>
```

# JSON陣列讀取 – load()

- ◆ 使用load()函數所讀出的JSON檔案格式為list

```
1 import json
2
3 # 無中文的JSON檔案
4 fp = "../JSON/jsonarray_en.json"
5
6 ▼ with open(fp) as file_en:
7     data = json.load(file_en)
8
9 print(data)      # ['A', 'apple', 'B', 'book', 'C', 'cat']
10 print(type(data)) # <class 'list'>
11
12
13 # 有中文的JSON檔案
14 fp = "../JSON/jsonarray_ch.json"
15
16 ▼ with open(fp, encoding="utf8") as file_ch:
17     data = json.load(file_ch)
18
19 print(data)      # ['A', '蘋果', 'B', '書', 'C', '貓']
20 print(type(data)) # <class 'list'>
```

# JSON陣列讀取 – loads()

- ◆ 先用read()讀檔並得到字串型態資料，再用loads()將字串轉成list

```
1  import json
2
3  # 無中文的JSON檔案
4  fp = "../JSON/jsonarray_en.json"
5
6  ▼ with open(fp) as file_en:
7      # 使用read()函數讀取JSON檔案並轉成字串
8      data = file_en.read()
9
10 # 將型態為字串的data用loads()轉成list型態
11 file = json.loads(data)
12 print(file)           # ['A', 'apple', 'B', 'book', 'C', 'cat']
13 print(type(file))     # <class 'list'>
14
15 # 有中文的JSON檔案
16 fp = "../JSON/jsonarray_ch.json"
17
18 ▼ with open(fp, encoding="utf8") as file_ch:
19     # 使用read()函數讀取JSON檔案並轉成字串
20     data = file_ch.read()
21
22 # 將型態為字串的data用loads()轉成list型態
23 file = json.loads(data)
24 print(file)           # ['A', '蘋果', 'B', '書', 'C', '貓']
25 print(type(file))     # <class 'list'>
```

# JSON物件寫入 – dump()、dumps()

- ◆ dump()：將dict型態的資料直接寫成JSON檔案
- ◆ dumps()：先將dict型態的資料轉成字串型態，再用write()寫入檔案

```
1  import json
2
3  dictobject = {"A":"apple","B":"book","C":"cat"}
4
5  fp1 = "jsonobject1.json"
6  ▼ with open(fp1, "w") as outfile:
7      # 使用dump()將dict資料寫到指定檔案中
8      json.dump(dictobject, outfile)
9
10
11  fp2 = "jsonobject2.json"
12  # 先將dict資料用dumps()轉成字串
13  file = json.dumps(dictobject)
14  ▼ with open(fp2, "w") as outfile:
15      # 再使用write()寫入檔案
16      outfile.write(file)
```

# JSON陣列寫入 – dump()、dumps()

- ◆ 寫入方式和JSON物件一樣，唯一要注意的就是編碼問題
- ◆ 若寫入的資料有中文，寫入時即便有設定編碼是utf8，但打開檔案看中文字會呈現「\uXXX」的樣子，其實這不是亂碼，這是ASCII編碼，若想要排除這狀況，可以增加一參數「ensure\_ascii=False」

jsonarray1.json - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

["A", "\u860b\u679c", "B", "\u66f8", "C", "\u8c93"]

```
1 import json
2
3 dictarray = ["A", "蘋果", "B", "書", "C", "貓"]
4
5 fp1 = "jsonarray1.json"
6 ▼ with open(fp1, "w") as outfile:
7     # 使用dump()將dict資料寫到指定檔案中
8     json.dump(dictarray, outfile, ensure_ascii=False)
9
10
11 fp2 = "jsonarray2.json"
12 # 先將dict資料用dumps()轉成字串
13 file = json.dumps(dictarray, ensure_ascii=False)
14 ▼ with open(fp2, "w") as outfile:
15     # 再使用write()寫入檔案
16     outfile.write(file)
```

# JSON和Python資料型態對照

## load()

JSON	Python
object	dict
array	list
string	str
number(int)	int
number(real)	float
true	True
false	False
null	None

## dump()

Python	JSON
dict	object
list	array
str	string
int	number(int)
float	number(real)
True	true
False	false
None	null