



網路程式設計

Python資料程式設計

Instructor: 馬豪尚

資料交換格式

- › JSON和CSV是網路爬蟲常見的資料交換格式
- › JSON (JavaScript Object Notation) 是輕量級的資料交換語言，容易閱讀的文字為主，用來傳輸由屬性值或者序列性的值組成的資料物件。
- › CSV(Comma-Separated Values)，逗號分隔值(也稱為字元分隔值)，是一份純文字的檔案格式，以儲存表格資料。

Python 字典(Dictionary)

- › 字典是一個容器(集合)可以用來存放不同資料型態的資料，它的每一個元素是以鍵(Key)及值(Value)構成。
- › Dictionary(字典)有幾個特性
 - Iterable(可疊代的)：可以透過Python迴圈來進行元素的讀取。
 - Modifiable(可修改的)：和串列(List)一樣可以透過Python提供的方法(Method)來對Dictionary(字典)的值進行修改。
 - Key-Value pairs(鍵與值)：Dictionary(字典)的每一個元素由鍵(Key)及值(Value)構成。鍵(Key)的資料型態通常我們使用String(字串)或Integer(整數)，而值(Value)可以是任何資料型態。

Python 字典(Dictionary)

› 創建字典的方法

– 於 {} 符號中輸入每個元素的鍵(Key)與值(Value)

› fruits = {'西瓜':15, '香蕉':20, '水蜜桃':25}

– 使用dict()方法，傳入鍵(Key)的名稱，並且指派值(Value)給它

› fruits = dict(西瓜=15, 香蕉=20)

Python 字典(Dictionary)

- › 存取字典內元素的方法
 - 使用[], 傳入key的名稱
 - › `print(dict_name[key])`
 - 使用`get()`方法, 傳入要尋找的Key名稱, 它會回傳其Value
 - › `print(dict_name.get(key))`
 - 透過Python迴圈來存取字典中的每一個元素
 - › `for key in dict_name: print(key)` → 只能存取key
 - › `for tuple in dict_name.items(): print(tuple)` → 返回(key, value)

Python 字典操作

- › 新增或更新資料
 - `dict_name[key] = value`
- › 一次新增或更新多筆資料
 - `original_dict.update(new_dict)`
- › 字典複製
 - `new_dict = dict_name.copy()`

Python 字典操作

- › 刪除特定資料
 - `del dict_name['key']`
- › 刪除字典內所有元素
 - `dict_name.clear()`
- › 刪除特定資料並回傳刪除的資料值
 - `value = dict_name.pop(key)`
- › 後進先出方式刪除資料(刪除最後加入的資料)
 - `tuple = dict_name.popitem()`

Python 字典(Dictionary)

- › 查詢所有key
 - `dict_name.keys()`
- › 查詢所有value
 - `dict_name.values()`
- › 查詢所有(key, value)
 - `dict_name.items()`

排序字典OrderedDict

- › OrderedDict是dict的一個子類別
 - 會對key做排序，而排序的依據在於這個key被插入的先後順序
 - 更新了某個key的value值並不會影響他在OrderedDict的排序位置
 - 可以與sorted()結合使用，讓這個OrderedDict可以照我們所希望的依據來排序

排序字典OrderedDict

- › `import OrderedDict`
- › `d = OrderedDict()`
- › `d['first']=5`
- › `d['second']=4`
- › `d['third']=8`
- › `d['fourth']=7`

Python Sorted函數

- › 可對串列進行基本的排序(依照數字大小或依照英文字母順序)
 - `sorted(list, reverse=False)`
- › 可依照tuple中指定欄位來做排序
 - `sorted(tuple, key=lambda x:x[0])`

OrderedDict結合sorted函數

- › `d={'first'=5, 'second'=4, 'third'=8, 'fourth'=7 }`
- › 依照key排列
 - `sorted(d.items(),key=lambda x:x[0])`
 - `OrderedDict(sorted(d.items(),key=lambda x:x[0]))`
- › 依照value排列
 - `sorted(d.items(),key=lambda x:x[1])`
 - `OrderedDict(sorted(d.items(),key=lambda x:x[1]))`

JSON資料格式

- › 瀏覽器和網站伺服器之間交換資料，資料只能是文字形式，JSON就是一種文字資料格式，最初是為了JavaScript開發的
- › 這種資料格式常被應用在Web開發和大數據資料庫(NoSQL)，Python也有採用與支援這種格式，可以將資料以JSON的格式做儲存

JSON資料格式

› 物件

- 在json中的物件採用key : value的方式配對儲存
- 物件內容用左右大括弧{ }來包住
- key和value中間用冒號 ":" 來區隔
- 每一組key : value以逗號 "," 來區隔
- key必須是一個文字字串
- value可以是數值、字串、布林值、陣列、null

› 陣列

- 陣列的值可以是數值、字串、布林值、陣列、null

JSON資料樣式

```
{
  "id": 123,
  "Name": "wsrsw",
  "Email": "wsrsw@example.com",
  "contents": [
    {
      "subject": "Math",
      "score": 80
    },
    {
      "subject": "English",
      "score": 90
    }
  ]
}
```

JSON字串轉Python dict

- › Import json
- › loads(json_str)
 - 回傳一個python字典物件

json與python資料類型轉換關係表

Python	JSON
dict	object
list, tuple	array
str, unicode	string
int, long, float	number
True	true
False	false
None	null

Python dict轉JSON字串

- › `Json_str = json.dumps(dict)`
- › `dumps`中的`sort_keys`參數
 - Python字典是無序的資料，使用`sort_keys=True`可以將轉成的json進行排序
 - `json.dumps(dict, sort_keys=True)`
- › `dumps`中的`indent`參數
 - `indent`可以讓轉成json格式進行縮排排版，讓json格式比較好閱讀
 - `json.dumps(dict, indent=4)`

將Python資料直接輸出成json檔

› dump()函數

- json.dump(data, jsonfile)
- 第一個參數為python的資料，想要序列化的目標
 - › dictObj = {'b':80, 'a':25, 'c':60}
- 第二個參數為開啟的json寫入物件
 - › with open('mock_data.json', 'w', newline='') as **jsonfile**:

練習1

- › 參考populations.json檔案，該檔案為世界各國人口數的資料
 - 將2000年的相關資料取出並存入populations_2000.json檔案

練習2

- › 參考aqi.json檔案，該檔案為環保署空氣品質資料
 - 列出隸屬於“台中市”底下的所有站台名稱、站台ID、pm2.5的值
 - 台中市= \u81fa\u4e2d\u5e02

CSV檔案

› 用逗號隔開的文字檔案

```
first_name,last_name,city  
Eli,Manning,New York  
Kevin ,James,Cleveland  
Mike,Jordon,Chicago
```

first_name	last_name	city
Eli	Manning	New York
Kevin	James	Cleveland
Mike	Jordon	Chicago

Python 讀取 CSV

- › Import csv
- › 開啟csv檔案
 - csvFile = open(檔案名稱, encoding='utf-8')
 - with open(檔案名稱, encoding='utf-8') as csvFile :
 - csvFile可以自行命名
- › 建立csv的reader物件
 - csvReader = csv.reader(csvFile)
物件變數 csv模組讀取函數
- › 將資料讀取成串列形式
 - data_list=list(csvReader)

Python 讀取 CSV

- › 使用for迴圈讀取串列內容
 - for row in data_list:
- › 使用串列索引讀取csv的內容
 - data_list[0][1], data_list[0][2]

Python 讀取 CSV - 字典

- › 將CSV讀成字典物件輸入
 - `csvDictReader = csv.DictReader(csvFile)`
 - 傳回的是 ****OrderedDict(排序字典)**** 類型，所以可以用欄位名稱當索引取得資料

`{'first_name': 'Eli', 'last_name': 'Manning', 'city': 'New York'}`

`{'first_name': 'Kevin', 'last_name': 'James', 'city': 'Cleveland'}`

`{'first_name': 'Mike', 'last_name': 'Jordon', 'city': 'Chicago'}`

first_name	last_name	city
Eli	Manning	New York
Kevin	James	Cleveland
Mike	Jordon	Chicago

Python 寫入 CSV

- › Import csv
- › 開啟寫入csv檔案
 - csvFile = open(檔案名稱, 'w', newline = '')
 - with open(檔案名稱, 'w', newline = '') as csvFile :
 - csvFile可以自行命名
- › 建立csv寫入物件
 - writer = csv.writer(csvFile)
- › 寫入一系列資料到csv
 - writer.writerow(['Lisa', 'Liao', 'Seattle'])

Python 寫入 CSV

› 寫入欄位標題(header)

– 寫入一列資料代表欄位標題

› `writer.writerow(['first_name', 'last_name', 'city'])`

› delimiter: 分隔符號

– `writer = csv.writer(csvfile, delimiter='設定的分隔符號')`

– `writer = csv.writer(csvfile, delimiter='\t')`

– `writer = csv.writer(csvfile, delimiter=' ')`

```
Name    Age    City
Hung    35     Taipei
James   40     Chicago
```

Python 寫入 CSV -字典

- › Import csv
- › 開啟寫入csv檔案
 - csvFile = open(檔案名稱, 'w', newline = '')
 - with open(檔案名稱, 'w', newline = '') as csvFile :
 - csvFile可以自行命名
- › 建立csv寫入物件
 - dicWriter = csv.DictWriter(csvfile, fieldnames=fields)
 - 在完成 dictWiter之前，必須先設定fields串列，這個串列包含未來字典中的鍵(key)。

Python 寫入 CSV -字典

- › 寫入欄位標題(字典的key)
 - `dictWriter.writeheader()`
- › 寫入字典資料
 - `dictWriter.writerow({'first_name':'Lisa','last_name' : 'Liao', 'city' : 'Seattle'})`

練習3

- › 參考csvReport.csv，該檔案為一間茶葉公司的業務成績，
 - Name: 業務員名
 - Year: 年分
 - Product: 銷售產品
 - Price: 產品單價
 - Quantity: 銷售數量
 - Revenue: 銷售總額
 - Location: 地區
- › 請統計以下的資料
 - Steve在2015-2017年總共銷售的業績為多少
 - Steve在2017年銷售業績數字尚未填入，請計算完之後填入該檔案