




Week 11



檔案處理

- Open file 時 import os
 - 方便透過path功能查詢檔案是否存在
 - 到指定路徑下找檔案
- 另一作法: import csv
 - 讀出csv檔
 - 寫入csv檔

比較兩種做法

```
import os
#若檔案存在，建立檔案物件並開啟
if os.path.isfile('test.csv'):
    infile=open('test.csv','r')
    outfile=open('result.txt','w+')
    for record in infile:
        data=record.split(',')
        if(data[0].find('2013') > -1 & data[2].find('國小')
> -1):
            if (data[3]=='6年級'):
                str=data[1]+' '+data[2]+' '+data[3]
                outfile.write(str+'\n')
    infile.seek(0)
```

```
import os
import csv
if os.path.isfile('test.csv'):
    infile=open('test.csv','r')
    outfile=open('result.csv','w+')
    csvreader=csv.reader(infile,delimiter=',')
    csvwriter=csv.writer(outfile)#,delimiter=',')
    header=next(csvreader)
    for data in csvreader:
        if(data[0].find('2013') > -1 & data[2].find('
國小') > -1):
            if (data[3]=='6年級'):
                data[0]="
                data[4]="
                csvwriter.writerow(data)
    infile.seek(0)
```

你可以做的練習

- 將日期字串中的 '/' 改成 '-' 存成另一個檔
- 可使用replace的功能
如: data[0].replace('/' , '-')

Python 存取網站方式

- 靜態網頁擷取
- 動態網頁擷取
- Logging 模組

以維基百科為練習。

<https://en.wikipedia.org/>

```
>>> import requests
>>> r = requests.get('https://en.wikipedia.org/')
>>> r
<Response [200]>
>>> r.ok
True
>>> type(r)
<class 'requests.models.Response'>
>>> len(r.text)
75579
>>> |
```

html的狀態碼: 2xx成功、
3xx重新導向
4xx用戶端錯誤
5xx伺服器端錯誤

```

>>> r.encoding      編碼方式
'UTF-8'
>>> r.headers
{'Date': 'Mon, 11 May 2020 02:36:19 GMT', 'Content-Type': 'text/html; charset=UTF-8', 'Server': 'mw1326.eqiad.wmnet', 'X-Content-Type-Options': 'nosniff', 'P3P': 'CP="See https://en.wikipedia.org/wiki/Special:CentralAutoLogin/P3P for more info."', 'Content-language': 'en', 'Vary': 'Accept-Encoding, Cookie, Authorization', 'Last-Modified': 'Mon, 11 May 2020 02:36:18 GMT', 'Content-Encoding': 'gzip', 'Age': '239', 'X-Cache': 'cp5009 hit, cp5010 hit/500', 'X-Cache-Status': 'hit-front', 'Server-Timing': 'cache;desc="hit-front"', 'Strict-Transport-Security': 'max-age=106384710; includeSubDomains; preload', 'X-Client-IP': '36.230.155.158', 'Cache-Control': 'private, s-maxage=0, max-age=0, must-revalidate', 'Accept-Ranges': 'bytes', 'Content-Length': '19384', 'Connection': 'keep-alive'}
>>> type(r.headers)
<class 'requests.structures.CaseInsensitiveDict'>
>>> r.headers['Content-Type']
'text/html; charset=UTF-8'
>>> |

```

直接以當中的設定去判別應如何處理該網頁

靜態網頁

- 靜態網頁中不包含任何.js檔
- 伺服器回傳的時候就是完整的網頁
- 此時網路爬蟲程式中最重要的部分就是如何解析網頁的HTML檔案。
- HTML定義元素
 - Tag(標籤)就是元素的名字
 - Attribute(屬性)描述元素的屬性
 - Content(內容)則是元素的內容

靜態網頁擷取

➤HTML常用標籤

- <!--註解文字-->
- 粗體文字、<i>斜體文字</i>、<u>底線文字</u>
- <head>網站的開頭</head>
- <body>網頁檔案之主體</body>
- <div>網頁檔案的一個區塊，裡面可以包含很多元素</div>
- <title>網頁標題名稱（顯示於視窗標題和分頁之名稱）</title>
- <h1>HTML內文標題1(最高級)標題，通常也是標題中最重要的</h1>
- <a href>超連結，跟著href屬性一起合用
- <form>使用者輸入之HTML表單</from>
- <tr> / <td>：定義表格時最常用的兩個標籤，<tr> 是列，<td> 則是欄。

靜態網頁擷取

➤網路爬蟲常用的屬性

- id：獨一無二的代表網頁。
- class：描述類似的元素的歸類。
- href：超連結，有超連結我們就可以繼續深入下一個連結。

➤靜態網頁網路爬蟲步驟

- 獲取網站
- 分析網站
- 儲存結果

```
>>> dir(bs4.BeautifulSoup)
['_ASCI_SPACES', 'DEFAULT_BUILDER_FEATURES', 'NO_PARSER_SPECIFIED_WARNING', 'ROOT_TAG_NAME', '_
bool', '_call', '_class', '_contains', '_copy', '_delattr', '_delitem', '_
dict', '_dir', '_doc', '_eq', '_format', '_ge', '_getattr', '_getattribut
e', '_getitem', '_getstate', '_gt', '_hash', '_init', '_init_subclass', '_
iter', '_le', '_len', '_lt', '_module', '_ne', '_new', '_reduce', '_red
uce_ex', '_repr', '_setattr', '_setitem', '_sizeof', '_str', '_subclasshook',
'_unicode', '_weakref', 'all_strings', 'check_markup_is_url', 'decode_markup', 'fe
ed', 'find_all', 'find_one', 'is_xml', 'lastRecursiveChild', 'last_descendant', 'linkage
fixer', 'popToTag', 'should_pretty_print', 'append', 'childGenerator', 'children', 'clear', '
decode', 'decode_contents', 'decompose', 'decomposed', 'descendants', 'encode', 'encode_content
s', 'endData', 'extend', 'extract', 'fetchNextSiblings', 'fetchParents', 'fetchPrevious', 'fetc
hPreviousSiblings', 'find', 'findAll', 'findAllNext', 'findAllPrevious', 'findChild', 'findChil
dren', 'findNext', 'findNextSibling', 'findNextSiblings', 'findParent', 'findParents', 'findPre
vious', 'findPreviousSibling', 'findPreviousSiblings', 'find_all', 'find_all_next', 'find_all_p
revious', 'find_next', 'find_next_sibling', 'find_next_siblings', 'find_parent', 'find_parents',
'find_previous', 'find_previous_sibling', 'find_previous_siblings', 'format_string', 'formatt
er_for_name', 'get', 'getText', 'get_attribute_list', 'get_text', 'handle_data', 'handle_endtag',
'handle_starttag', 'has_attr', 'has_key', 'index', 'insert', 'insert_after', 'insert_before',
'isSelfClosing', 'is_empty_element', 'new_string', 'new_tag', 'next', 'nextGenerator', 'nextS
ibling', 'nextSiblingGenerator', 'next_elements', 'next_siblings', 'object_was_parsed', 'parent
Generator', 'parents', 'parserClass', 'popTag', 'prettify', 'previous', 'previousGenerator', 'p
reviousSibling', 'previousSiblingGenerator', 'previous_elements', 'previous_siblings', 'pushTag',
'recursiveChildGenerator', 'renderContents', 'replaceWith', 'replaceWithChildren', 'replace
with', 'replace_with_children', 'reset', 'select', 'select_one', 'setup', 'smooth', 'string', '
string_container', 'strings', 'stripped_strings', 'text', 'unwrap', 'wrap']
```

BeautifulSoup

解析器	使用方法
Python's html.parser	BeautifulSoup(markup, "html.parser")
lxml's HTML parser	BeautifulSoup(markup, "lxml")
lxml's XML parser	BeautifulSoup(markup, "lxml-xml") BeautifulSoup(markup, "xml")
html5lib	BeautifulSoup(markup, "html5lib")


```

>>> from bs4 import BeautifulSoup
>>> soup = BeautifulSoup(r.text, 'html.parser')
>>> print(soup.title)
<title>Wikipedia, the free encyclopedia</title>
>>> soup.find_all('div', class_='otd-footer')
[<div class="otd-footer hlist noprint" style="text-align: right;">
<ul><li><b>a href="/wiki/Wikipedia:Selected_anniversaries/April" title="Wikipedia:Selected ann
iversaries/April">Archive</a></b></li>
<li><b>a class="extiw" href="https://lists.wikimedia.org/mailman/listinfo/daily-article-1" tit
le="mail:daily-article-1">By email</a></b></li>
<li><b>a href="/wiki/List_of_historical_anniversaries" title="List of historical anniversaries
">List of historical anniversaries</a></b></li></ul>
</div>]

```

也可以寫成

```

soup.find_all('div', 'otd-footer')
soup.find_all('div', {'class': 'otd-footer'})

```

** 這裡的語法是用 "class_"，因為class是python內建的關鍵字之一，所以beautifulsoup使用class_替代 **

```

>>> soup.find("h1")
<h1 class="firstHeading" id="firstHeading" lang="en">Main Page</h1>
>>> soup.find("h1").contents
['Main Page']
>>> soup.find('h1').contents
['Main Page']
>>> soup.find_all('h1').contents
Traceback (most recent call last):
  File "<pyshell#28>", line 1, in <module>
    soup.find_all('h1').contents
  File "C:\Users\Li-Ling\AppData\Local\Programs\Python\Python38-32\lib\site-packages\beautifulsoup4-4.9.0-py3.8.egg\bs4\element.py", line 2127, in __getattr__
    raise AttributeError(
AttributeError: ResultSet object has no attribute 'contents'. You're probably treating a list of elements like a single element. Did you call find_all() when you meant to call find()?

```

```

>>> for title in soup.find_all('h1'):
>>>     print(title.contents)

```

```

['Main Page']
>>>

```

```
>>> soup.find("h2")
<h2 id="mp-tfa-h2" style="margin:0.5em; background:#cef2e0; font-family:inherit; font-size:120%; font-weight:bold; border:1px solid #a3bfb1; color:#000; padding:0.2em 0.4em;"><span id="From_today.27s_featured_article"></span><span class="mw-headline" id="From_today's_featured_article">From today's featured article</span></h2>
```

```
>>> test=soup.find_all("h2")
>>> for t in test:
    print(t.span)
```

```
<span id="From_today.27s_featured_article"></span>
<span class="mw-headline" id="Did_you_know...">Did you know ...</span>
<span class="mw-headline" id="In_the_news">In the news</span>
<span class="mw-headline" id="On_this_day">On this day</span>
<span id="Today.27s_featured_picture"></span>
<span class="mw-headline" id="Other_areas_of_Wikipedia">Other areas of Wikipedia</span>
<span id="Wikipedia.27s_sister_projects"></span>
<span class="mw-headline" id="Wikipedia_languages">Wikipedia languages</span>
None
>>> for t in test:
    print(t.span.contents)
```

```
[]
['Did you know\xa0...']
```

靜態網頁網路爬蟲概念

■ 靜態網頁網路爬蟲實作

- 獲取網頁：以 <http://rate.bot.com.tw/xrt?Lang=zh-TW> (台銀牌告匯率)舉例

臺灣銀行
BANK OF TAIWAN

2017/07/31 本行營業時間牌告匯率

請注意：1. 本表資料僅供參考，不代表實際交易匯率。
2. 「網路銀行」及「EasyWeb線上申請匯款附加費」之實際交易匯率，以交易時顯示之匯率為準。
3. 櫃檯實際交易匯率以交易時本行匯率為準。
4. 本網頁牌告匯率資訊為靜態顯示，顯示之牌告匯率資訊不會隨後續資訊而自動更新資訊，欲得知本行最新牌告匯率資訊請至「取得最新報價」處。

取得最新報價 線上申請及辦理外匯業務

牌價最新換牌時間：2017/07/31 12:01

幣別	現金匯率		即期匯率		遠期匯率	歷史匯率
	本行買入	本行賣出	本行買入	本行賣出		
美金 (USD)	29.905	30.447	30.205	30.305	查詢	查詢
港幣 (HKD)	3.724	3.919	3.844	3.904	查詢	查詢
英鎊 (GBP)	38.64	40.57	39.51	39.93	查詢	查詢
澳幣 (AUD)	23.82	24.48	24.01	24.24	查詢	查詢
加拿大幣 (CAD)	23.87	24.61	24.14	24.36	查詢	查詢
新加坡幣 (SGD)	21.77	22.55	22.19	22.37	查詢	查詢
瑞士法郎 (CHF)	30.55	31.61	31.08	31.37	查詢	查詢
日圓 (JPY)	0.2653	0.2763	0.2717	0.2757	查詢	查詢
南非幣 (ZAR)	-	-	2.28	2.36	查詢	查詢

靜態網頁網路爬蟲概念

靜態網頁網路爬蟲實作

- 分析網頁：從前面網頁中按滑鼠右鍵，出現快顯功能表後，點選檢視網頁原始碼按鈕，接著會出現我們想分析的內容



2017/07/31 本行營業時間牌告匯率

請注意：1. 本表資料僅供參考，不代表實際交易匯率。
2. 「網路銀行」及「Easy線上申請現鈔或匯兌」之實際交易匯率，以交易時顯示之匯率為準。
3. 除匯兌交易匯率以交易時本行匯率為準。
4. 本網頁牌告匯率資訊為靜態顯示，顯示之牌告匯率資訊不會隨後續變動而自動更新資訊，欲得知本行最新牌告匯率資訊請按「取得最新報價」或「線上申請外幣現鈔或匯兌」。

牌價最新掛牌時間：2017/07/31 12:01

幣別	本行買入	本行賣出	定期匯率	活期匯率	歷史匯率
美金 (USD)	29.915	30.305	查詢	查詢	查詢
港幣 (HKD)	3.72	3.904	查詢	查詢	查詢
英鎊 (GBP)	38.6	39.93	查詢	查詢	查詢
澳幣 (AUD)	23.6	24.24	查詢	查詢	查詢
加拿大幣 (CAD)	23.6	24.14	查詢	查詢	查詢
新加坡幣 (SGD)	21.77	22.19	查詢	查詢	查詢
瑞士法郎 (CHF)	30.55	31.08	查詢	查詢	查詢
日圓 (JPY)	0.2653	0.2717	查詢	查詢	查詢
南非幣 (ZAR)	-	2.28	查詢	查詢	查詢

靜態網頁網路爬蟲概念

- 分析網頁：從原始碼可以看出，這個網頁的表格包括由tr(表格的列標籤)分割的各個幣別，各幣別內還有td(表格的行標籤) 搭配div class和visible-phone描述的幣別資訊(如美金(USD))與由td標籤描述的各種匯率資料(如上圖中本行現金買入匯率為29.915)所組成

```
<html lang="zh-TW" class="no-js">
<head>
  <meta charset="utf-8" />
  <title>臺灣銀行牌告匯率</title>
</head>
<tr>
  <td data-table="幣別" class="currency phone-small-font">
    <div>
      <div class="sp-div sp-america-div">
        
      </div>
      <div class="visible-phone print_hide" />
      <div class="visible-phone print_hide">
        美金 (USD)
      </div>
      <div class="hidden-phone print_show" style="text-indent:30px;">
        美金 (USD)
      </div>
    </div>
  </td>
  <td data-table="本行現金買入" class="rate-content-cash text-right print_hide">29.915</td>
  <td data-table="本行現金賣出" class="rate-content-cash text-right print_hide">30.457</td>
  <td data-table="本行即期買入" class="rate-content-sight text-right print_hide" data-hide="phone">30.215</td>
  <td data-table="本行即期賣出" class="rate-content-sight text-right print_hide" data-hide="phone">30.315</td>
</tr>
```

要找的資訊藏在哪段原始碼?

- 可善用檢查功能

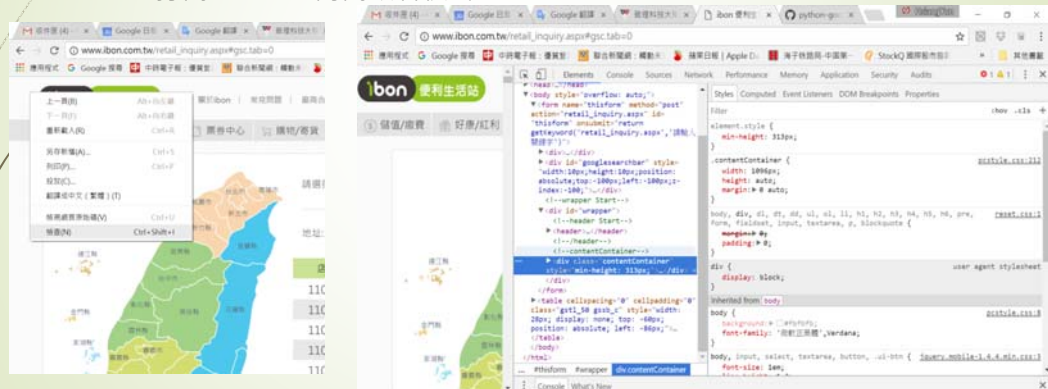
動態網頁擷取

- 爬取用戶與伺服器互動過程
- 範例程式E2-1-2-1.py-查詢ibon中7/11門市資訊

爬取用戶與伺服器互動過程

以ibon網站查詢各門市資訊為例，分成三個步驟

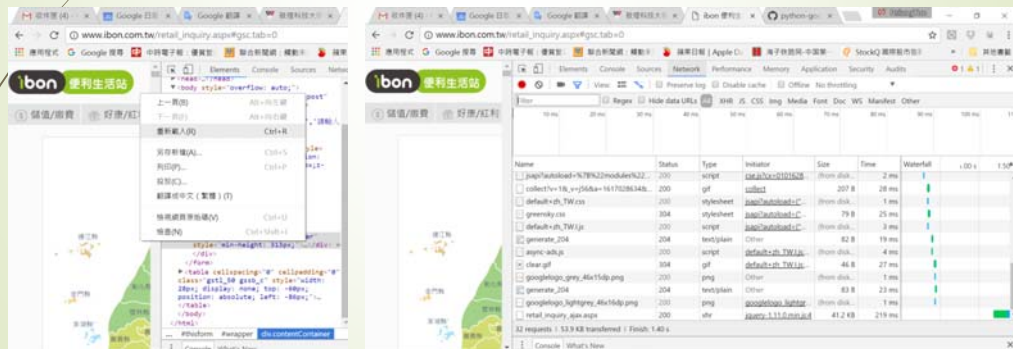
- 在原來頁面下按滑鼠右鍵，出現快顯功能表後，點選【檢查】按鈕，打開Chrome的開發者模式。



爬取用戶與伺服器互動過程

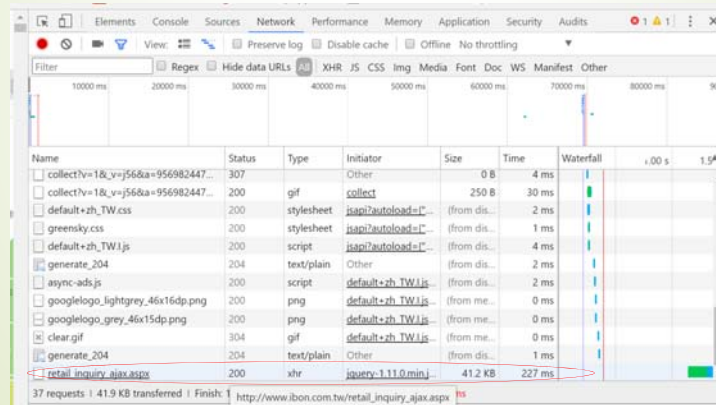
以ibon網站查詢各門市資訊為例，分成三個步驟

- 重新整理網頁頁面：在原來頁面下按滑鼠右鍵，出現快顯功能表後，點選【重新載入】重整網頁頁面。
- 接著點選【Network】按鈕，開始觀察用戶與伺服器間的互動。



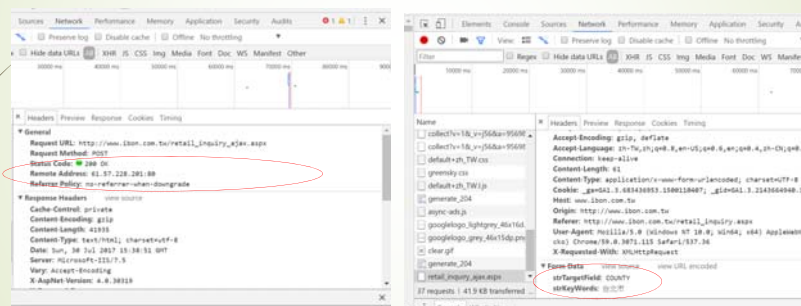
爬取用戶與伺服器互動過程

- 以ibon網站查詢各門市資訊為例，分成三個步驟
- 觀察資料獲取的方式：從右邊各指令花費時間發現 retail_inquiry_ajax.aspx 耗時最長，可以推論是進行資料傳接造成。



爬取用戶與伺服器互動過程

- 以ibon網站查詢各門市資訊為例，分成三個步驟
- 以滑鼠點擊左圖retail_inquiry_ajax.aspx，可得到右圖，瞭解此時請求的網址是http://www.ibon.com.tw/retail_inquiry_ajax.aspx，請求的方式是post。

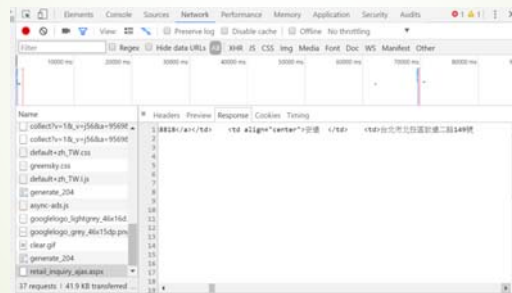


- 往下捲動頁面到最底，可得圖2-1-2-8，瞭解使用Post方式請求時，傳出的資訊與相關變數名稱，分別是'strTargetField':'COUNTY'、'strKeywords':'縣市別'。

爬取用戶與伺服器互動過程

➡ 以ibon網站查詢各門市資訊為例，分成三個步驟

- 接著點選【Response】按鈕確認回傳資訊是否存在，能發現從中可得所要的店名、地址等資訊。



- 觀察完就可以開始用Python爬取資訊

你可以做的練習

- ➡ 試試查找你常上的店家資訊查詢，利用程式把資料找回來