# 3.2 csv 資料的讀取與寫入

csv 是許多資料編輯、讀取及儲存時很喜歡的格式，因為是純文字檔案，操作方便而且輕量。Python 可以使用 csv 模組輕鬆存取 .csv 檔案。

## 3.2.1 認識 CSV

CSV(Comma Separated Values) 是一種以符號分隔值的資料格式並以純文字的方式儲存為檔案，其中常用的符號為「,」。最廣泛的應用是在程式之間進行資料的交換，因為許多程式都有專屬的資料檔案，為了與其他的程式相通，就必須將資料內容轉換為通用格式方便其他程式使用，而 CSV 就是其中很受歡迎的選項。

csv 檔案是純文字的檔案，編輯時可以直接使用文字編輯器，如 Windows 內建的記事本，但是在閱讀上有時會較為不方便。Excel 也可以直接編輯、讀取及儲存 csv 檔案，以欄列的方式來顯示 csv 檔案的內容較易閱讀，因此有較多的人都會利用 Excel 來開啟編輯 csv 檔案。

## 3.2.2 csv 模組的使用

csv 模組是 Python 內建的模組，使用前並不需要安裝就能直接載入使用。在模組中常用來讀取與寫入的資料格式有二種，分別是 **串列** 及 **字典**，說明如下：

| 函數 | 說明 |
|---|---|
| csv.reader( 檔案 ) | 產生讀取 csv 資料的物件，讀取格式為串列。 |
| csv.writer( 檔案 ) | 產生寫入 csv 資料的物件，寫入格式為串列。 |
| csv.DictReader( 檔案 ) | 產生讀取 csv 資料的物件，讀取格式為字典。 |
| csv.DictWriter( 檔案 ) | 產生讀取 csv 資料的物件，寫入格式為串列。 |

## 3.2.3 csv 檔案讀取

若想將 csv 檔案中的資料以串列或字典格式進行讀取，可以使用以下的函數：

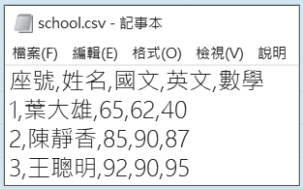
1. 讀取為串列格式：csv.reader()。
2. 讀取為字典格式：csv.DictReader()。

## 讀取 csv 檔案為串列資料

利用 csv.reader() 可以讀取 csv 檔案成為串列，例如：

程式碼：csv_read.py

```
import csv
# 開啟 csv 檔案
with open('school.csv', newline='') as csvfile:
    # 讀取 csv 檔案內容
    rows = csv.reader(csvfile)
    # 以迴圈顯示每一列
    for row in rows:
        print(row)
```




執行結果：

```
['座號', '姓名', '國文', '英文', '數學']
['1', '葉大雄', '65', '62', '40']
['2', '陳靜香', '85', '90', '87']
['3', '王聰明', '92', '90', '95']
```

## 讀取 csv 檔案為字典資料

利用 csv.DictReader() 可以讀取 csv 檔案內容轉為字典格式，例如：

程式碼：csv_read_dict.py

```
import csv
# 開啟 csv 檔案
with open('school.csv', newline='') as csvfile:
    # 讀取 csv 檔內容，將每一列轉成 dictionary
    rows = csv.DictReader(csvfile)
    # 以迴圈顯示每一列
    for row in rows:
        print(row['座號'],row['姓名'],row['國文'],row['英文'],
row['數學'])
```

執行結果：

```
1 葉大雄 65 62 40
2 陳靜香 85 90 87
3 王聰明 92 90 95
```

# 3.4 Excel 資料儲存與讀取

openpyxl 模組可以存取最新版的 Excel 文件格式，要特別注意的是它只支援 .xlsx 格式，並不支援 .xls 格式。Anaconda 預設安裝 openpyxl 模組，可以直接使用。

## 3.4.1 Excel 檔案新增及儲存

### openpyxl 模組新增及儲存的流程


### 用 openpyxl 模組儲存 xlsx 檔

將指定的資料儲存到 <test.xlsx> 檔。

程式碼：xlsx_write.py

```
import openpyxl
# 建立一個工作簿
workbook=openpyxl.Workbook()
# 取得第 1 個工作表
```

# 4.2 繪製長條圖：bar、barh

## 4.2.1 繪製直條圖

**直條圖** 是以 **bar** 函數繪製，語法為：

```
plt.bar(x 座標串列 , y 座標串列 , width=0.8, bottom=0[, 其他參數 ])
```

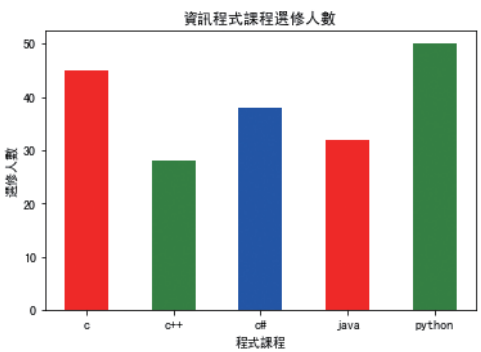
繪圖時除了 x、y 座標串列參數之外，呈現每個項目的矩形是重點，常用參數有：

- **width**：設定每個項目矩形的寬度。以二個刻度之間的距離為基準，用百分比為單位來設定。不設定時預設值為 0.8。
- **bottom**：設定每個項目矩形 y 座標的起始位置，不設定時預設值為 0。
- **color**：設定每個項目矩形的顏色，設定值與折線圖相同，預設為藍色。例如設定紅色可以為 "r" 或 "red"。如果設定值為 "rgb"，代表會以紅、綠、藍依序循環顯示每個項目矩形的顏色。
- **label**：設定每個項目圖例名稱，此屬性需搭配 **legend** 函數才有效果。

本範例將要用直條圖呈現每個課程的選修人數：

**程式碼：bar1.py**

```
...
listx = ['c','c++','c#','java','python']
listy = [45,28,38,32,50]
plt.bar(listx, listy, width=0.5, color='rgb')
plt.title(" 資訊程式課程選修人數 ")
plt.xlabel(" 程式課程 ")
plt.ylabel(" 選修人數 ")
...
```

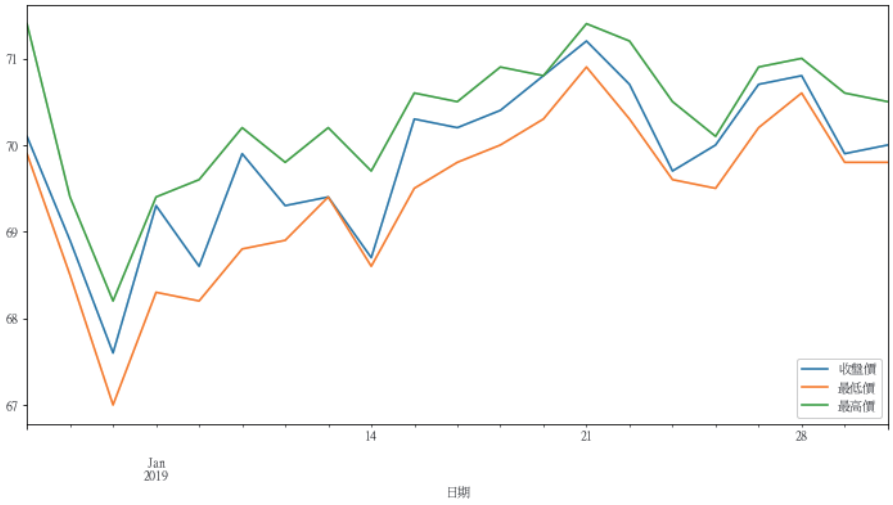


# 10.3 實戰：個股單月與年度統計圖

股票市場強調公開、透明，幾乎所有股票資訊皆可在台灣證券交易所取得。本專題擷取台灣證券交易所日成交資料，先以單月繪製模組統計圖，再集合全年 12 個月資料繪製全年統計圖。

## 10.3.1 單月個股統計圖

為了不必每次執行都到台灣證券交易所讀取資料，程式第一次執行會將資料存於 CSV 檔，第二次以後執行程式就由 CSV 檔讀取資料，不但節省網路流量，也加快執行速度。本範例以收盤價、最高價及最低價繪製線形圖，使用者可由三者推估股價走勢及當日股價震盪情形。



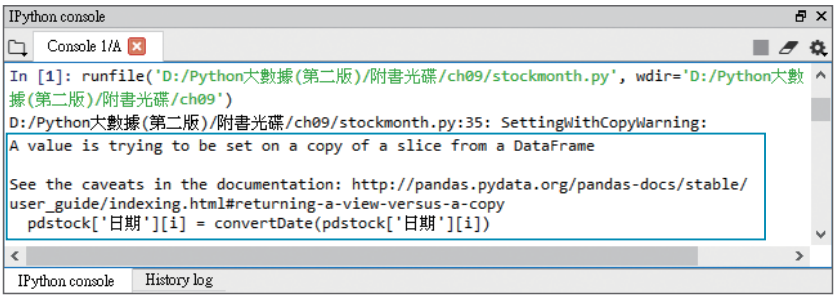

程式碼：**stockmonth.py**

```
1 def convertDate(date):  # 轉換民國日期為西元 :108/01/01->20190101
2     str1 = str(date)
3     yearstr = str1[:3]  # 取出民國年
4     realyear = str(int(yearstr) + 1911)  # 轉為西元年
5     realdate = realyear + str1[4:6] + str1[7:9]  # 組合日期
6     return realdate
7
8 import requests
```

```
 9 import json, csv
10 import pandas as pd
11 import os
12 import matplotlib.pyplot as plt
13
14 plt.rcParams["font.sans-serif"] = "mingliu"  # 設定中文字型
15 plt.rcParams["axes.unicode_minus"] = False
16
17 pd.options.mode.chained_assignment = None  #取消顯示pandas資料重設警告
18
19 filepath = 'stockmonth01.csv'
20
21 if not os.path.isfile(filepath):  # 如果檔案不存在就建立檔案
22     url_twse = 'http://www.twse.com.tw/exchangeReport/STOCK_DAY?
        response=json&date=20190101&stockNo=2317&_=1521363562193'
23     res = requests.get(url_twse)  # 回傳為json資料
24     jdata = json.loads(res.text)  #json解析
25
26     outputfile = open(filepath, 'w', newline='', encoding='utf-8')
        # 開啟儲存檔案
27     outputwriter = csv.writer(outputfile)  # 以csv格式寫入檔案
28     outputwriter.writerow(jdata['fields'])
29     for dataline in (jdata['data']):  # 寫入資料
30         outputwriter.writerow(dataline)
31     outputfile.close()  # 關閉檔案
32
33 pdstock = pd.read_csv(filepath, encoding='utf-8')  # 以pandas讀取檔案
34 for i in range(len(pdstock['日期'])):  # 轉換日期式為西元年格
35     pdstock['日期'][i] = convertDate(pdstock['日期'][i])
36 pdstock['日期'] = pd.to_datetime(pdstock['日期'])
    # 轉換日期欄位為日期格式
37 pdstock.plot(kind='line', figsize=(12, 6), x='日期',
    y=['收盤價', '最低價', '最高價'])  # 繪製統計圖
```

**程式說明**

- 1-6　將以民國為年份的日期字串轉換成以西元為年份的日期字串之自訂函式。
- 8-12　含入模組。
- 14-15　設定繪圖的中文字型。
- 17　35 列程式轉換日期格式時，Pandas 會顯示資料被重設的警告訊息。本列程式可將上述警告訊息移除不顯示。




- 19　設定 CSV 檔案名稱為「stockmonth01.csv」。
- 21-31　檢查 CSV 檔案是否存在，如果不存在就建立 CSV 檔案。
- 21　檢查 CSV 檔案是否存在，若不存在才執行 22-31 列程式。
- 22-23　程式讀取網頁資料。
- 24　將 JSON 資料的資料型態由字串轉換為字典，如此在 28 及 29 列程式才能以欄位名稱取得資料。
- 26-27　建立檔案，並設定以 CSV 格式寫入資料。由網頁讀取的資料為：

```
{"stat":"OK","date":"20190101","title":"108 年 01 月 2317 鴻海
    各日成交資訊",
"fields":["日期","成交股數","成交金額","開盤價","最高價","最低價",
    "收盤價","漲跌價差","成交筆數"],
"data":[
  ["108/01/02","16,775,306","1,182,131,473","71.40","71.40",
    "69.90","70.10","-0.70","7,968"],
  [108/01/03","36,659,461","2,526,323,765","69.00","69.40",
    "68.50","68.90","-1.20","17,345"],
  … (略)
```

- 28　資料標題在「fields」欄位，此列程式將標題寫入檔案。
- 29-30　每日的股票資料在「data」欄位，「data」欄位是一個二維串列，這兩列程式逐一將每日的股票資料寫入檔案。
- 31　關閉檔案。

完成的 CSV 檔如下：

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 日期 | 成交股數 | 成交金額 | 開盤價 | 最高價 | 最低價 | 收盤價 | 漲跌價差 | 成交筆數 | |
| 2 | 108/01/02 | 16,775,306 | 1,182,131,473 | 71.4 | 71.4 | 69.9 | 70.1 | -0.7 | 7,968 | |
| 3 | 108/01/03 | 36,659,461 | 2,526,323,765 | 69 | 69.4 | 68.5 | 68.9 | -1.2 | 17,345 | |
| 4 | 108/01/04 | 37,313,571 | 2,520,135,828 | 68.2 | 68.2 | 67 | 67.6 | -1.3 | 18,110 | |
| 5 | 108/01/07 | 24,084,557 | 1,661,796,863 | 68.7 | 69.4 | 68.3 | 69.3 | 1.7 | 11,167 | |

標題

每日資料

- 33　使用 Pandas 的 read_csv 方法由 CSV 檔讀取資料。
- 34-35　轉換日期為西元年格式。
- 36　以 Pandas 的 to_datetime 方法將日期資料型態由字串轉換為日期格式。
- 37　繪出圖形：「kind='line'」為線形圖，「figsize=(12,6)」設定圖形長度及寬度，「x=' 日期 '」設定以日期欄位做為橫軸，「y=[' 收盤價 ',' 最低價 ',' 最高價 ']」表示同時繪出收盤價、最低價、最高價三條線形圖。

## 10.3.2 全年個股統計圖

有了單月個股資料，可以使用迴圈結合全年十二個月個股資料，就能繪製全年個股統計圖了！


程式碼：stockyear.py

```
1 def twodigit(n):  # 將數值轉為二位數字串
2     if(n < 10):
3         retstr = '0' + str(n)
4     else:
5         retstr = str(n)
6     return retstr
7
8 def convertDate(date):  # 轉換民國日期為西元 :106/03/02->20170302
```

Chapter

15

# 即時網路聲量輿情收集器

* **專題方向**
* **關鍵技術**

  擷取及分析非同步載入資料

  下載指定日期的資料

  將資料儲存在 txt 檔案中
* **實戰：即時網路聲量輿情資料下載**

  擷取即時熱門關鍵字及資訊

  依日期儲存收集結果

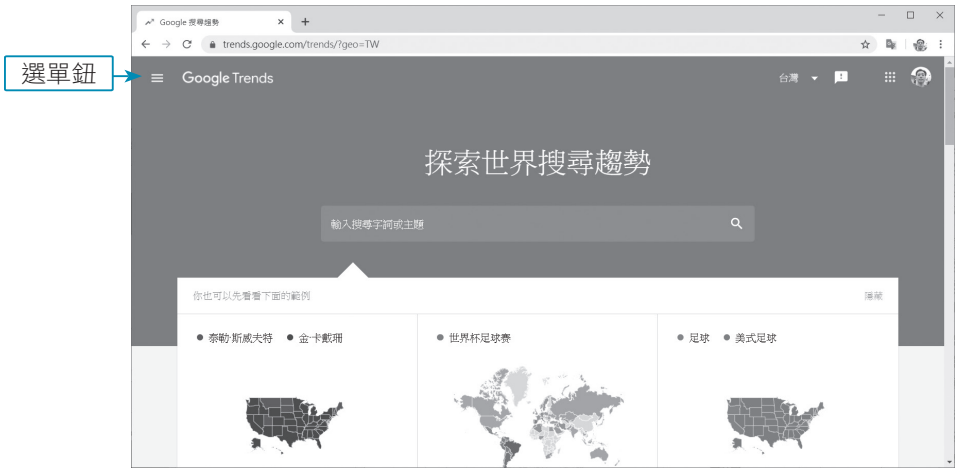
  延伸應用

# 15.1 專題方向

**Google 搜尋趨勢** (https://trends.google.com/trends) 可根據設定的地區中搜尋的熱門關鍵字進行分析並排名，並依關鍵字進行相關的網路資源收集，讓我們即時了解時勢、掌握時代的脈動，匯集即時網路聲量、情蒐輿論最新發展。

## 專題檢視

在這個專題中，我們將由 Google 搜尋趨勢網站搜尋台灣最近兩天最熱門的新聞話題關鍵字，並將這些新聞資訊全部下載並存檔，供爾後的參考。




▲ Google 搜尋趨勢網站：https://trends.google.com/trends

請按左上角的選單鈕，選擇 **搜尋趨勢**，預設是以 **每日搜尋趨勢** 這個分類來顯示台灣目前最熱門的搜尋關鍵字排行。

點選每一個熱門的關鍵字，就可以看到和這關鍵字相關的新聞。


點選每一則新聞就會顯示該新聞的詳細內容。


## 專題重點

本專題中我們以 Google 熱門的關鍵字下載相關的新聞，有幾個需要關注的重點：

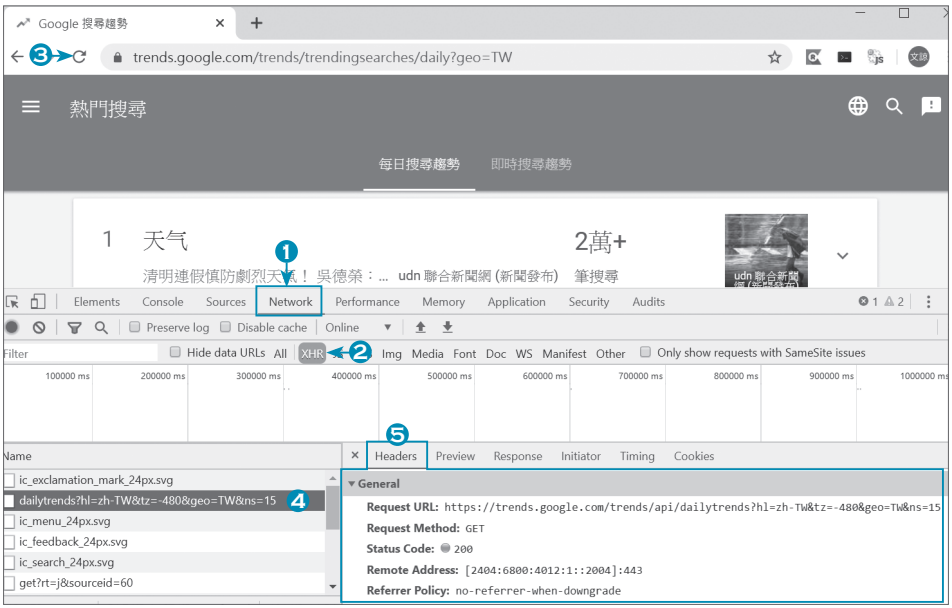
1. 從主頁面解析如何以參數，設定下載日期下載相關的新聞。
2. 將回傳的字串資料轉換為字典，並使用 JSON Viewer 分析字典的結構。
3. 將回傳的資料依日期存成文字檔。

# 15.2 關鍵技術

本專題首先在頁面找出資料載入的來源，再利用日期的設定，下載指定日期的資料。

## 15.2.1 擷取及分析非同步載入資料

### 觀察 XHR 請求方式




❶ 開啟 **開發人員工具** 介面並調整其位置，然後按 **Network** 切換到 Network 頁籤。

❷ 按 **XHR** 頁籤觀察 Ajax 動態網頁產生的結果。

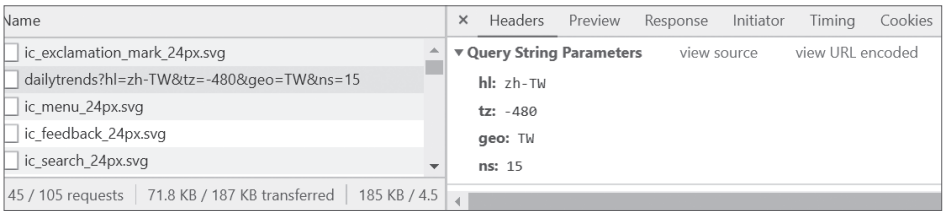
❸ 再按一次重整按鈕。

❹ 點選 dailytrends?hl=zh-TW&tz=-480&geo=TW&ns=15 加以分析。

❺ 點選 **Headers** 頁籤並向下捲動，觀察各項參數。

在 **General** 項目中取得 **RequestURL** 和 **RequestMethod** 表示是向「https://trends.google.com/trends/api/dailytrends?hl=zh-TW&tz=-480&geo=TW&ns=15」 以 GET 方法進行請求。

**Headers** 頁籤再向下捲動到最後，觀察 **Query String Parameters** 參數，主要的是 hl、tz、geo 和 ns 參數。




依照觀察結果，我們可以使用 requests 模組，向「https://trends.google.com/trends/api/dailytrends」加上參數發出 GET 請求，即可得到回傳資料。程式碼如下：

```
import requests
url = 'https://trends.google.com/trends/api/dailytrends'
payload = {
            "hl": "zh-TW",
            "tz": "-480",
            "geo": "TW",
            "ns": "15",
          }
html = requests.get(url,params=payload)
```

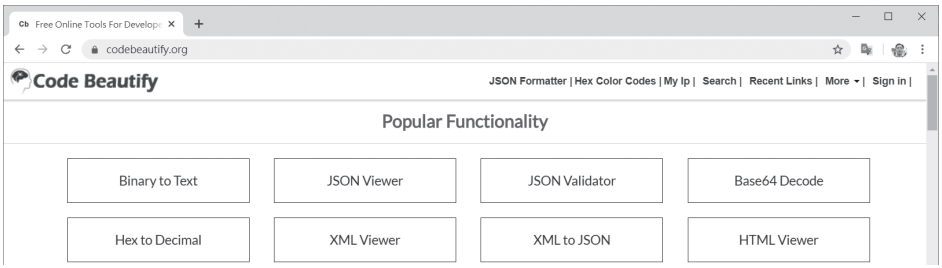
## 下載 json 文字檔

其實 GET 請求的方式，即是將頁面網址加上參數。所以請將剛才的網址：「https://trends.google.com/trends/api/dailytrends?hl=zh-TW&tz=-480&geo=TW&ns=15」直接貼到瀏覽器執行，如下圖該網站會回傳相關資料，化為下載的檔案：<json.txt>。

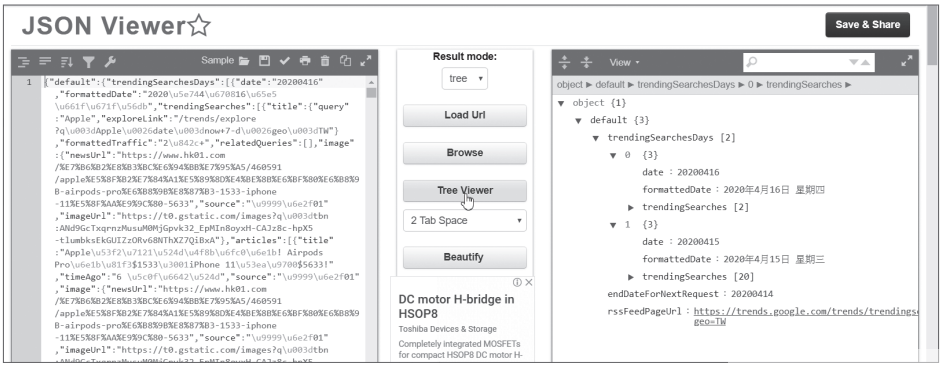
這個下載的 <json.txt> 檔就包含了搜尋關鍵字及相關網頁的資料，用記事本打開可看出資料是字典格式的字串，但第一列文字是多餘的。仔細觀察文字是用 utf-8 編碼。


## 以 JSON Viewer 解析

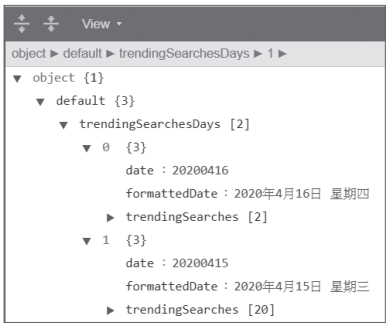
如果想進一步解析，可以使用相關工具來幫忙。**CodeBeautify**(https://codebeautify.org/) 是一個提供多種線上編碼、解碼工具的網站。在這裡我們主要是利用 JSON Viewer 作解碼，請按 **JSON Viewer** 鈕。




請以記事本開啟 <json.txt> 檔內容，先刪除第一列文字後，然後將所有文字貼在左邊的視窗中，按下中間 **Tree Viewer** 鈕即可在右邊的視窗中顯示解碼的結果。



仔細分析後發現，需要的資訊都放在 default 鍵的 trendingSearchesDays 鍵值對中，其中有兩筆資料，第一筆是當天的資料，第二筆則是前一天的資料。




在 Python 中可以利用 json 模組接收回傳的資料，下列程式會以「,」字元將下載的字串資料分割為串列，並將串列元素分別儲存在「_,datas」變數中，其中 datas 變數才是真正的資料。接著以 json.loads(datas) 轉換為字典，再以 jsondata['default']['trendingSearchesDays'] 取得資訊並存在 trendingSearchesDays 串列中。

```
_,datas=html.text.split(',',1)
jsondata=json.loads(datas) # 將下載資料轉換為字典
trendingSearchesDays=jsondata['default']['trendingSearchesDays']
```

接著利用迴圈就可以處理所有 trendingSearchesDays 串列，然後以 formattedDate 鍵讀取日期。

```
for trendingSearchesDay in trendingSearchesDays:
    formattedDate=trendingSearchesDay['formattedDate']
```

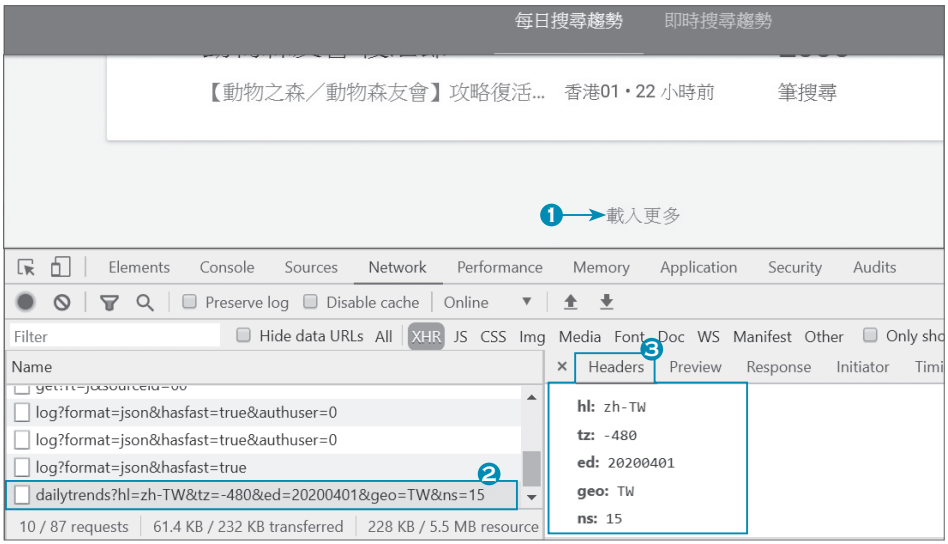
每篇文章的詳細資訊都包括在 trendingSearches 鍵的 articles 鍵值對中，articles 是一個串列，每一個串列元素就是一則新聞，每則新聞中則可以 title、source、timeAgo、snippet 和 url 鍵取得標題、媒體、發佈時間、內容和相關連結的詳細資訊。

可以下列程式讀取每篇文章的詳細資訊。

```
for trendingSearchesDay in trendingSearchesDays:
    formattedDate=trendingSearchesDay['formattedDate']
    print('日期:' + formattedDate)
    print()
    for data in trendingSearchesDay['trendingSearches']:
        print('【主題關鍵字:' + data['title']['query'] + '】')
        print()
        for content in data['articles']:
            print('標題:', content['title'])
            print('媒體:', content['source'])
            print('發佈時間:', content['timeAgo'])
            print('內容:', content['snippet'])
            print('相關連結:', content['url'])
            print()
        print('-'*50)
```

## 15.2.2 下載指定日期的資料

利用 ed 參數可以設定下載的日期，請依如下的操作。




❶ 網頁往下捲動到最下方，按 **載入更多** 鈕。

❷ 選取新產生載入網址。