輕鬆學習 Python | 從基礎到應用，成為初級 Python 資料分析師

如何擷取網頁內容（網路爬蟲）

郭耀仁

課程綱要

- 網頁資料擷取的核心任務
- 由 Web API 擷取網頁資料
- 由 HTML 擷取網頁資料
- 操縱自動化瀏覽器擷取網頁資料

網頁資料擷取的核心任務

盤點核心任務

以 Python 豐富的模組、Chrome 瀏覽器外掛與開發者工具來進行兩項核心任務：

1. 獲得資料
2. 解析資料

獲得資料

- 使用 Quick JavaScript Switcher (https://chrome.google.com/webstore/detail/quick-javascript-switcher/geddoclleiomckbhadiaipdggiiccfje) 與 Chrome 開發者工具判斷網頁資料類型
- 以 `requests` 或 `selenium` 發送 HTTP 請求獲得網頁資料

解析資料

- 如果網頁資料是 JSON 格式：以 `requests` 獲取後可直接以 Python 資料結構解析
- 如果網頁資料是 XML 格式：以 `lxml` 搭配 XPath 解析
- 如果網頁資料是 HTML 格式：以 `pyquery` 或 `selenium` 搭配 CSS Selector 解析

由 Web API 擷取網頁資料

幫助瀏覽 Web API 資料外觀的 Chrome 外掛

JSON View
(https://chrome.google.com/webstore/detail/jsonview/chklaanhfefbnpoihckbnefhakgolnmc)

Web API 資料範例

- 空氣品質指標(AQI) (https://opendata.epa.gov.tw/ws/Data/AQI/?$format=json)
- data.nba (https://data.nba.com/)
- PChome (https://ecshweb.pchome.com.tw/search/v3.3/all/results?q=macbook&page=1&sort=sale/dc)

判斷網頁資料類型

- 以 https://ecshweb.pchome.com.tw/search/v3.3/?q=macbook (https://ecshweb.pchome.com.tw/search/v3.3/?q=macbook) 示範

獲得與解析 Web API 資料

- JSON 格式
- XML 格式

獲得與解析 Web API 資料：JSON 格式

以 https://opendata.epa.gov.tw/ws/Data/AQI/?$format=json (https://opendata.epa.gov.tw/ws/Data/AQI/?$format=json) 示範

In [1]:
```
import requests

aqi_url = "https://opendata.epa.gov.tw/ws/Data/AQI/?$format=json"
r = requests.get(aqi_url, verify=False)
aqi = r.json()
print(type(aqi))
```

```
<class 'list'>

/Users/kuoyaojen/anaconda3/lib/python3.7/site-packages/urllib3/connectionpool.
py:847: InsecureRequestWarning: Unverified HTTPS request is being made. Adding
certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/latest/advanced-usage.html#ssl-warnings
  InsecureRequestWarning)
```

獲得與解析 Web API 資料：XML 格式

以 https://opendata.epa.gov.tw/ws/Data/AQI/?$format=xml (https://opendata.epa.gov.tw/ws/Data/AQI/?$format=xml) 示範

In [2]:
```
import requests

aqi_url = "https://opendata.epa.gov.tw/ws/Data/AQI/?$format=xml"
r = requests.get(aqi_url, verify=False)
```

```
/Users/kuoyaojen/anaconda3/lib/python3.7/site-packages/urllib3/connectionpool.
py:847: InsecureRequestWarning: Unverified HTTPS request is being made. Adding
certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/latest/advanced-usage.html#ssl-warnings
  InsecureRequestWarning)
```

In [3]:

```
from lxml import etree
from io import BytesIO

f = BytesIO(r.content)
tree = etree.parse(f)
site_names = [t.text for t in tree.xpath("/AQI/Data/SiteName")]
print(site_names)
```

```
['屏東(琉球)', '苗栗(後龍)', '彰化(大城)', '臺南(北門)', '富貴角', '麥寮', '關山', '馬公', '金門', '馬祖', '埔里', '復興', '永和', '竹山', '中壢', '三重', '冬山', '宜蘭', '陽明', '花蓮', '臺東', '恆春', '潮州', '屏東', '小港', '前鎮', '前金', '左營', '楠梓', '林園', '大寮', '鳳山', '仁武', '橋頭', '美濃', '臺南', '安南', '善化', '新營', '嘉義', '臺西', '朴子', '新港', '崙背', '斗六', '南投', '二林', '線西', '彰化', '西屯', '忠明', '大里', '沙鹿', '豐原', '三義', '苗栗', '頭份', '新竹', '竹東', '湖口', '龍潭', '平鎮', '觀音', '大園', '桃園', '大同', '松山', '古亭', '萬華', '中山', '士林', '淡水', '林口', '菜寮', '新莊', '板橋', '土城', '新店', '萬里', '汐止', '基隆']
```

由 HTML 擷取網頁資料

常見用來標示 HTML 資料的方法

- HTML 的標籤名稱
- HTML 標籤中給予的 id
- HTML 標籤中給予的 class
- **資料所在的 CSS 選擇器（CSS Selector）**
- 資料所在的 XPath

幫助 CSS 選擇的 Chrome 外掛

SelectorGadget
(https://chrome.google.com/webstore/detail/selectorgadget/mhjhnkcfbdhnjickkkdbjoemdmb

SelectorGadget (https://chrome.google.com/webstore/detail/selectorgadget/mhjhnkcfbdhnjickkkdbjoemdmbfginb) 的使用方法

1. 點選 SelectorGadget 的外掛圖示
2. 留意 SelectorGadget 的 CSS 選擇器
3. 移動滑鼠到想要定位的元素
4. 在想要定位的資料上面點選左鍵，留意 Clear 後面數字表示有多少個元素被選擇到
5. 移動滑鼠點選不要選擇的元素（改以紅底標記），並同時注意 CSS 選擇器位址與 Clear 後面數字

以 Captain Marvel (2019) (https://www.imdb.com/title/tt4154664) 示範 SelectorGadget (https://chrome.google.com/webstore/detail/selectorgadget/mhjhnkcfbdhnjickkkdbjoemdmbfginb) 的使用方法

- 評分
- 劇情類型
- 演員名單

使用 PyQuery 擷取網頁資料

- 使用 pyquery 模組的 PyQuery() 函數取得 HTML 文件
- 指派 CSS 選擇器或 XPath 解析出所需部分資料

In [4]:

```
#!pip install pyquery
from pyquery import PyQuery as pq

page_url = "https://www.imdb.com/title/tt4154664"
html_doc = pq(page_url)
```

In [5]:
```
rating_css = "strong span"
rating = [float(i.text()) for i in html_doc.items(rating_css)]
print(rating[0])
```

```
7.2
```

In [6]:

```
genre_css = ".subtext a"
genre = [i.text() for i in html_doc.items(genre_css)]
release_date = genre.pop()
print(genre)
print(release_date)
```

```
['Action', 'Adventure', 'Sci-Fi']
6 March 2019 (Taiwan)
```

In [7]:
```
cast_css = ".primary_photo+ td a"
cast = [i.text() for i in html_doc.items(cast_css)]
print(cast)
```

```
['Brie Larson', 'Samuel L. Jackson', 'Ben Mendelsohn', 'Jude Law', 'Annette Be
ning', 'Lashana Lynch', 'Clark Gregg', 'Rune Temte', 'Gemma Chan', 'Algenis Pe
rez Soto', 'Djimon Hounsou', 'Lee Pace', 'Chuku Modu', 'Matthew Maher', 'Akira
Akbar']
```

In [8]:

```
def get_movie_data(movie_url):
    """Getting movie data from IMDB.com"""
    page_url = "https://www.imdb.com/title/tt4154664"
    rating_css = "strong span"
    genre_css = ".subtext a"
    cast_css = ".primary_photo+ td a"
    html_doc = pq(page_url)
    rating = [float(i.text()) for i in html_doc.items(rating_css)][0]
    genre = [i.text() for i in html_doc.items(genre_css)]
    release_date = genre.pop()
    cast = [i.text() for i in html_doc.items(cast_css)]
    movie_data = {
        "movieRating": rating,
        "movieGenre": genre,
        "movieReleaseDate": release_date,
        "movieCast": cast
    }
    return movie_data
```

In [9]:
```
get_movie_data("https://www.imdb.com/title/tt4154664")
```

Out[9]:
```
{'movieRating': 7.2,
 'movieGenre': ['Action', 'Adventure', 'Sci-Fi'],
 'movieReleaseDate': '6 March 2019 (Taiwan)',
 'movieCast': ['Brie Larson',
  'Samuel L. Jackson',
  'Ben Mendelsohn',
  'Jude Law',
  'Annette Bening',
  'Lashana Lynch',
  'Clark Gregg',
  'Rune Temte',
  'Gemma Chan',
  'Algenis Perez Soto',
  'Djimon Hounsou',
  'Lee Pace',
  'Chuku Modu',
  'Matthew Maher',
  'Akira Akbar']}
```

實作練習：回傳資料加入片長 `movieLength`

實作練習：讓 `get_movie_data()` 更方便使用

- 可以輸入電影名稱，而非 URL！
- 以 `urllib.parse.quote_plus()` 製作 query string
- 以 `.attr('href')` 獲得連結

實作練習：擷取華航電影清單再前往 IMDB 查詢平等

- 華航電影清單 (http://www.fantasy-sky.com/ContentList.aspx?section=002)

幫助檢視 Cookies 的 Chrome 外掛

EditThisCookie
(https://chrome.google.com/webstore/detail/editthiscookie/fngmhnnpilhplaeedifhccceomclg

操縱自動化瀏覽器擷取網頁資料

在研究如何使 `get_movie_data()` 更方便的過程中我們做了幾個動作

1. 前往 https://www.imdb.com/ (https://www.imdb.com/) 首頁
2. 輸入電影名稱
3. 點選搜尋
4. 點選 Movie 分類標籤
5. 點選相似度最高的搜尋結果

這些操作可以利用 selenium 模組來自動化!

什麼是 Selenium

- Selenium 是瀏覽器自動化測試的解決方案
- Python 透過 Selenium WebDriver 呼叫瀏覽器驅動程式，再由瀏覽器驅動程式去呼叫瀏覽器
- 對 **Google Chrome** 與 Mozilla Firefox 兩個主流瀏覽器的支援最好

Selenium 環境設定

- 前往官方網站 (https://www.google.com/chrome/)下載最新版的瀏覽器
- 下載最新版的瀏覽器驅動程式 ChromeDriver (http://chromedriver.chromium.org/)
- 下載完成以後解壓縮在熟悉路徑讓後續指派較為方便

測試是否設定完成

用程式碼透過 ChromeDriver 操控 Chrome 瀏覽器前往 IMDB 首頁並將首頁的網址印出再關閉瀏覽器

In [10]:
```
#!pip install selenium
from selenium import webdriver

driver_path = "/Users/kuoyaojen/Downloads/chromedriver"
imdb_home = "https://www.imdb.com/"
driver = webdriver.Chrome(executable_path=driver_path) # Use Chrome
driver.get(imdb_home)
print(driver.current_url)
driver.close()
```

```
https://www.imdb.com/
```

要使用的方法

- `driver.get()`：前往 IMDB 首頁
- `driver.find_element_by_css_selector()`：定位搜尋欄位、搜尋按鈕與搜尋結果連結
- `driver.current_url`：取得當下瀏覽器的網址
- `elem.send_keys()`：輸入電影名稱
- `elem.click()`：按下搜尋按鈕與連結

實作練習：以 selenium 實作 get_movie_data()

延伸閱讀

- Requests: HTTP for Humans (http://docs.python-requests.org/en/master/)
- pyquery: a jquery-like library for python (https://pythonhosted.org/pyquery/)
- Selenium with Python (https://selenium-python.readthedocs.io/)
- 靜態擷取網頁內容 (https://www.datainpoint.com/data-science-in-action/04-scraping-web-statically.html)
- 動態擷取網頁內容 (https://www.datainpoint.com/data-science-in-action/05-scraping-web-dynamically.html)