



Python 爬蟲實戰

楊証琨, 楊鎮銘

中央研究院資訊科學研究所資料洞察實驗室



Lecturers



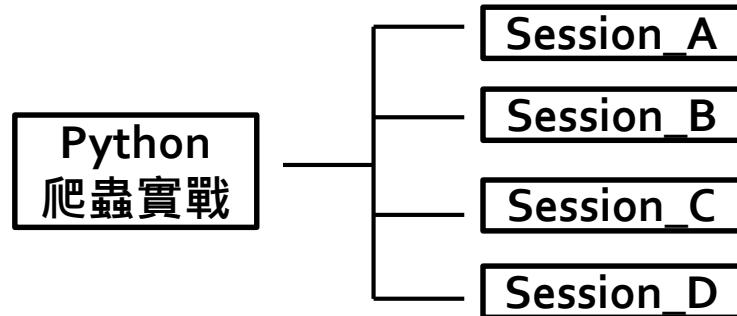
- 台大土木所畢
- 中研院資訊所研究助理
- 研究領域
 - 社群媒體資料處理分析
 - 製造工業影像資料分析

- 中研院資訊所研究助理
- 研究領域
 - pattern recognition
 - 資料分析



課程進行方式

- 今日的課程共分為四個 Session，課程會用到的 code 及 data 都放在相對應的資料夾內



- 課程會先由講師講解範例，在看到這個符號出現時，就代表是大家的練習時間囉！練習時有任何問題歡迎隨時舉手詢問旁邊的助教們



Outline – 上午

A. 爬蟲基本介紹

- HTML 初次見面
- 第一支爬蟲程式
- BeautifulSoup 的常用函數
- 正則表達式 (regular expression)

B. 爬蟲實戰與資料分析

- BeautifulSoup + regular expression
- GET vs. POST
- 使用 pandas 儲存資料
- 文字探勘與資料分析

Outline – 下午

C. 靜態網頁以外的爬蟲

- 圖片爬蟲
- 檔案爬蟲
- 網站爬蟲

D. 現實世界的爬蟲

- 現代網站爬蟲衍生的問題
- 動態網頁爬蟲

上午你將會學到...

- 運用 requests 發送 GET, POST 請求
- 運用 BeautifulSoup 解析 HTML 網頁
- 運用 regular expression 尋找目標資訊
- 運用 pandas 將抓到的資訊儲存為表格
- 運用 sklearn, jieba, wordcloud 做簡單的資料分析



實際 coding 把網頁資料爬下來!



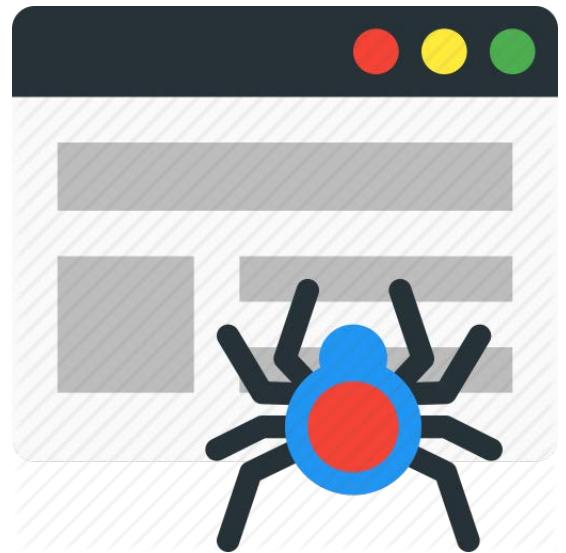


Outline

A. 爬蟲基本介紹

- 爬蟲介紹與 HTML 初次見面
- 第一支爬蟲程式
- BeautifulSoup 的常用函數
- 正則表達式 (regular expression)

爬蟲是什麼？有毒嗎？



網路爬蟲 (Web Crawler)

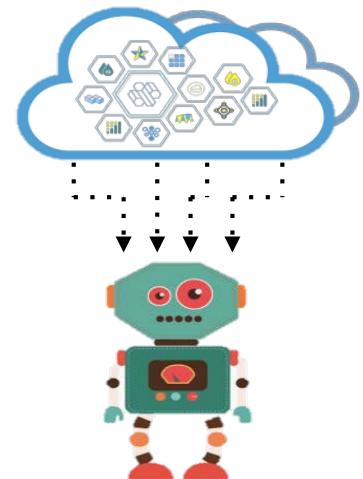
□ 網路爬蟲是一種機器人，可以幫您自動地瀏覽網際網路並擷取目標資訊

- 自動化爬取目標網站，並按照您的需求蒐集目標資料

□ 無所不在的爬蟲 - **Google**

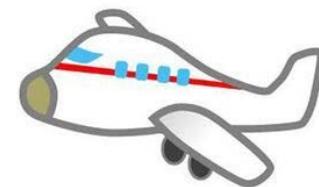


- 爬遍世界上的網站，建立索引來成立搜尋引擎



爬蟲可以做甚麼?

- 爬遍所有航空公司的網站，讓使用者只要輸入起訖點就可以知道最便宜的票價在哪裡
- 爬遍 Ptt、新聞評論，建立輿情分析系統，測風向!
- 你還能做甚麼?



網路爬蟲三步驟

1. 取得指定網域下的 HTML 資料

- 透過 requests 取得 HTML

2. 解析這些資料以取得目標資訊

- 透過開發者工具，觀察目標資訊的位置
- 透過 BeautifulSoup 解析 HTML

3. Loop!



爬蟲眼中的世界

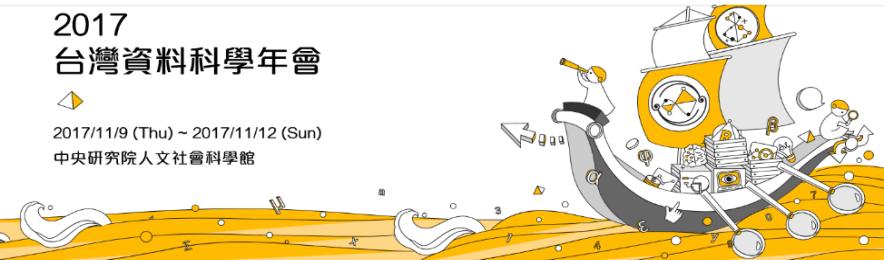


```
CSS/JS TYPE: REVEAL
<html lang="zh-TW" class="js js flexbox canvas canvastext webgl no-touch geolocation postmessage websqldatabase indexeddb hashchange history draganddrop websockets rgba hsla multiplebgs backgroundsize borderimage borderradius boxshadow textshadow opacity cssanimations csscolumns webworkers applicationcache smil svg inlinesvg smil svgclippaths background-fixed" style="background-color: #f0f0f0; font-family: sans-serif; font-size: 14px; margin: 0; padding: 0; color: black; position: relative; height: 100%; width: 100%;">
  <head>...</head>
  <body>...
    <script>var $=jQuery;</script>
    <div id="box_wrapper">
      <header id="header" class="include affix" data-include="header">...</header>
      <section id="mainSlider">...</section>
      <section class="intro">
        <div id="banner">
          
        </div>
        <div class="container">
          <div class="row">
            <div class="col-md-12" style="text-align: center; margin-bottom: 20px;">
              <img alt="台灣資料科學年會 logo" style="width: 150px; height: auto; margin-bottom: 10px;"/>
              <p>台灣資料科學年會由一群愛好資料科學的同好們共同舉辦，宗旨為推廣資料科學的認知、技術及應用。讓聽包含資料科學的各個層面，例如數理統計、資料視覺化、資料處理及計算、資料分析應用等，期待能透過演講、資料分析上手教學課程、心得分享、資料交流等各種形式，將我們對於資料科學的熱情傳達給大眾，幫助與會聽眾瞭解資料科學的魅力，進入資料科學的領域，進而一起探索與開發資料科學的潛力，更重要的是能促進將資料科學引入每個個人的專業領域之中，帶來新的創新及價值。</p>
            </div>
            <div class="col-md-5" style="margin-bottom: 20px;">
              <img alt="Google Chrome icon" style="width: 100px; height: auto; margin-bottom: 10px;"/>
              <img alt="Internet Explorer icon" style="width: 100px; height: auto; margin-bottom: 10px;"/>
            </div>
          </div>
        </div>
      </section>
      <section id="services" class="about grey_section" data-pg-collapsed="true">...</section>
      <section id="copyright" class="include color_section" data-include="copyright">...</section>
      <script src="/js/all.js?20170971212"></script>
      <script type="text/javascript">...</script>
      <script type="text/javascript" src="/wp-content/themes/datasciutu/assets/js/skip-link-focus-fix.js?v=1.0"></script>
      <script type="text/javascript" src="/wp-content/themes/datasciutu/assets/js/global.js?v=enc1.0"></script>
      <script type="text/javascript" src="/wp-content/themes/datasciutu/assets/js/jquery.scrollTo.js?v=2.1.2"></script>
      <script type="text/javascript" src="/wp-includes/js/wp-embed.min.js?v=4.7.4"></script>
      <script>$(function() { ... })</script>
    </div>
  </body>
</html>
```

台灣資料科學年會

2017 台灣資料科學年會

2017/11/9 (Thu) ~ 2017/11/12 (Sun)
中央研究院人文社會科學館

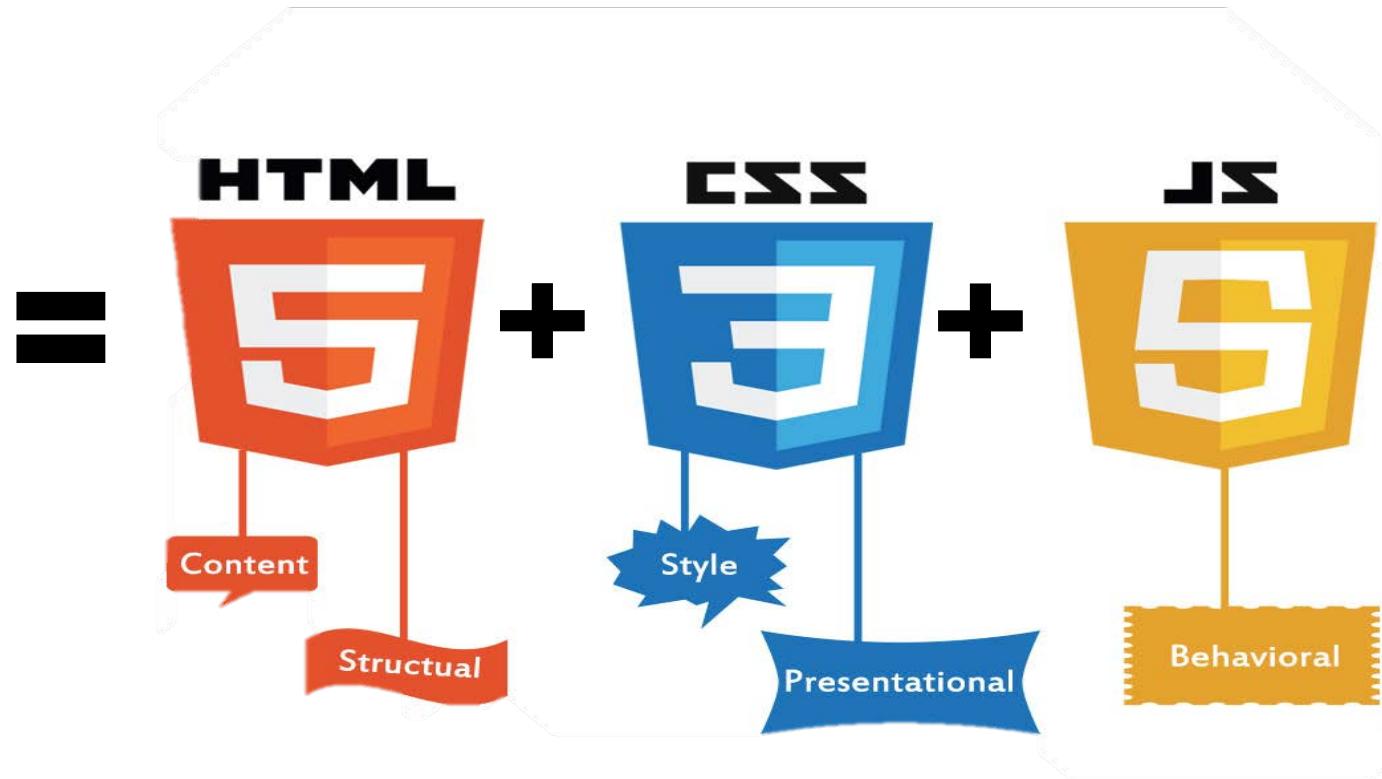


關於本年會

台灣資料科學年會由一群愛好資料科學的同好們共同舉辦，宗旨為推廣資料科學的認知、技術及應用。讓聽包含資料科學的各個層面，例如數理統計、資料視覺化、資料處理及計算、資料儲存以及各領域的資料分析應用等，期待能透過演講、資料分析上手教學課程、心得分享、資料交流等各種形式，將我們對於資料科學的熱情傳達給大眾，幫助與會聽眾瞭解資料科學的魅力，進入資料科學的領域，進而一起探索與開發資料科學的潛力，更重要的是能促進將資料科學引入每個個人的專業領域之中，帶來新的創新及價值。

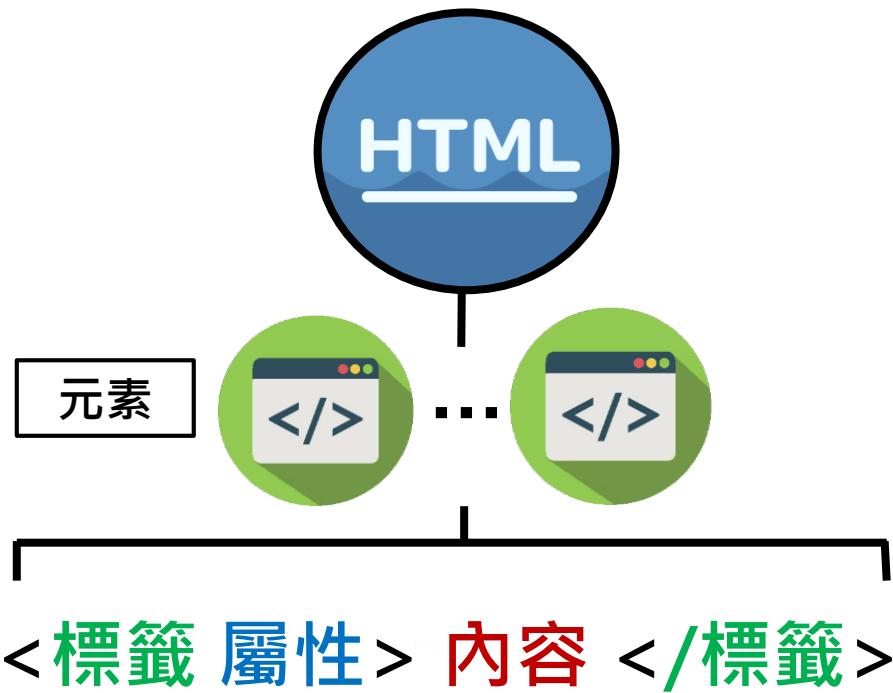


網頁的組成



什麼是 HTML?

- HyperText Markup Language (超文件標示語言)
- 由一群元素 (Elements) 所組成的階層式文件
- 一個元素包含開始標籤、結束標籤、屬性以及內容



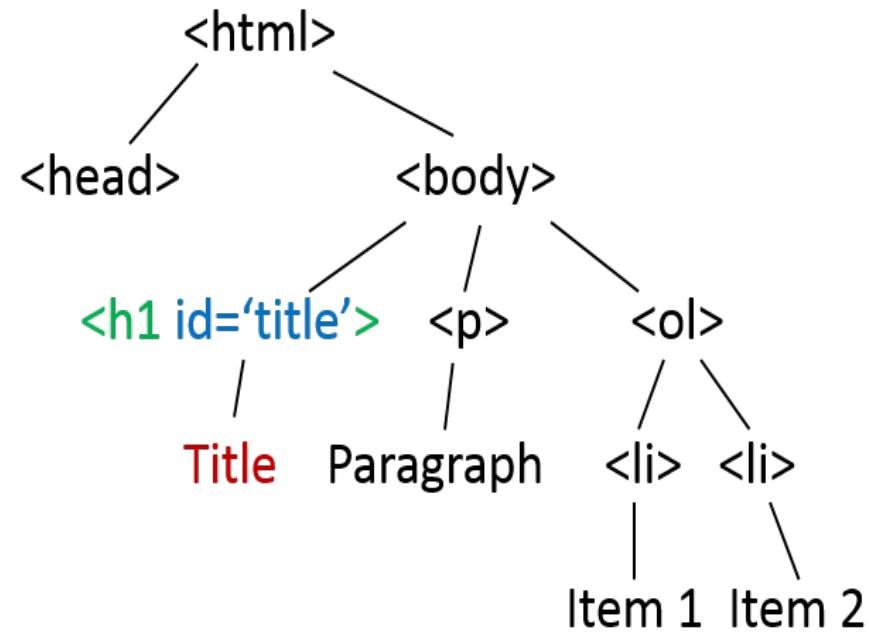
HTML 的結構

- 以樹狀結構，階層式呈現元素之間的關係

<標籤 屬性> 內容 </標籤>

Document Object Model

```
<html>
  <head>
    ...
  </head>
  <body>
    <h1 id="title"> Title </h1>
    <p> Paragraph </p>
    <ol>
      <li> Item 1 </li>
      <li> Item 2 </li>
    </ol>
  </body>
</html>
```



HTML 元素的組成 - 標籤 (Tags)

- 標籤通常是成對出現，有一個**開始標籤 (start tag)** 與**結束標籤 (end tag)**，例如此處的 `<h1>`, `</h1>`

開始標籤 → `<h1>I am heading!</h1>` ← 結束標籤

- 標籤內可以有**屬性 (attributes)** 來說明這個標籤的性質

`<p class="Hi!我是 attributes">Hi!我是 contents</p>`

- 開始標籤與結束標籤所包含的即為**內容 (contents)**，通常為顯示在網頁上的文字

`<title>My Coding Journal</title>`



常見的標籤 (Tags)

標籤名稱	用途
<h1> - <h6>	標題
<p>	段落
<a>	超連結
<table>	表格
<tr>	表格內的 row
<td>	表格內的 cell
 	換行 (無結束標籤)

標籤內的屬性 (Attributes)

- 標籤內可以有**屬性 (attributes)** 來說明這個標籤的性質，通常以 name = "value" 的方式呈現

```
<p class="Hi!我是 attributes">Hi!我是 contents</p>
```

- 一個標籤內也可同時存在多個屬性
 1. class = "value"
 2. id = "id1"

```
<div class="value" id="id1">Hi!我有兩個屬性</div>
```

- 超連結通常都是屬性 href 的值

```
<a href="http://foundation.datasci.tw/">資料協會網站</a>
```



常見的屬性 (Attributes)

屬性名稱	意義
class	標籤的類別 (可重複)
id	標籤的 id (不可重複)
title	標籤的顯示資訊
style	標籤的樣式
data-*	自行定義新的屬性

與 HTML 的初次見面 (1/3)

□ Step 1. 請先用 Chrome 打開 example_page.html



Python 爬蟲實戰

這是 h2 標籤的內容

這是 h3 標籤的內容

這是 p 標籤的內容

這是 div 標籤的內容，即使換行寫，網頁顯示出的文字一樣是不會換行

但如果用了 br 標籤
就可以順利斷行了

我是有著屬性 class="zzz" 的標籤內容

我是有著屬性 id="value-of-attr" 的標籤內容

標頭 1 (table-header)	標頭 2 (table-header)	標頭 3 (table-header)	標頭 4 (table-header)
列2 欄1	列2 欄2 (我的屬性 class="zzz")	列2 欄3 (我是 a 標籤，屬性 href=網址)	列2 欄4 (資料協會)
列3 欄1	列3 欄2 • 我是 li 標籤 (列表)，屬性 class="zz" • 第二個 li 標籤	列3 欄3 (資料協會-python 爬蟲實戰)	列3 欄4 (我的屬性 class="zzzz")

與 HTML 的初次見面 (2/3)

- Step 2. 請再用文字編輯器 (sublime text, 記事本...) 打開 example_page.html

```
26 ▼ <body>
27     <h1>Python 爬蟲實戰</h1>
28     <h2>這是 h2 標籤的內容</h2>
29     <h3>這是 h3 標籤的內容</h3>
30
31     <p title="i-am-title">這是 p 標籤的內容</p>
32
33     <div>
34         這是 div 標籤的內容,
35         即使換行寫，網頁顯示出的文字一樣是不會換行
36     </div>
37
38     <p>但如果用了 br 標籤 <br/> 就可以順利斷行了</p>
39
40     <div class = "zzz" id = "id1">
41         我是有著屬性 class="zzz" 的標籤內容
42     </div>
43     <p hidden>python_crawler</p>
44     <div id = "value-of-attr">
45         我是有著屬性 id="value-of-attr" 的標籤內容
46
```

與 HTML 的初次見面 (3/3)

</ CODE >

- Step 3. 用文字編輯器 (sublime text, notepad...) 打開 example_page.html 後
 - 在 h3 標籤的下一行加入 **<h4>我是 h4</h4>**
 - 將 p 標籤，屬性 title = "i-am-title" 的值改成 "**i-am-new-title**"，存檔後將滑鼠移到 "這是 p 標籤的內容" 上看看
 - 將 "列 2 欄 3" 的 a 標籤，屬性 href 的值改成 "**http://www.google.com.tw**"



CSS 的基本介紹

你有 freestyle 吗？

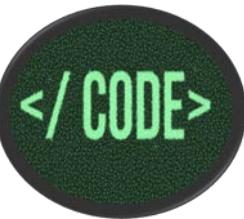


- Cascading Style Sheets (串接樣式表)
- 一種用來替 HTML 增加 style 的語言，舉凡修改顏色、字體大小、字體類型等等，皆由 CSS 完成

```
<html>
<head>
    <style>
        #i-am-id {
            background-color: LightCyan;
        }
    </style>
</head>
</html>
```



CSS 小試



- 請用文字編輯器 (sublime text, notepad...) 打開 example_page.html 後
 - 將 `#i-am-id {background-color: LightCyan;}` 的 **LightCyan** 改成 **Royalblue**，存檔後重新整理剛剛打開的網頁

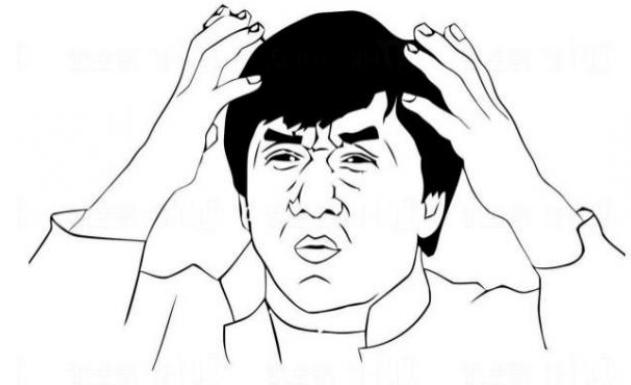
```
.abc {
    color: blue;
    font-size: 40px;
}

#i-am-id {
    background-color: Royalblue;
}

table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
```

小結

- 甚麼元素、標籤、屬性都聽不懂，有沒有懶人包？！



```
<p class="value"> target </p>
```

目標標籤

輔助資訊

目標資訊



在動手寫爬蟲之前...

□ 前人種樹，後人乘涼

- ▣ 許多爬蟲程式在 GitHub 可以找得到
e.g. PTT Crawler, 漫畫下載器



□ 想爬的網站是否有 API?

- ▣ 許多公司會透過 API 來提供乾淨、整齊的資料，並訂定爬行的規則
- ▣ e.g. [Facebook API](#), [Google Maps API](#)

□ 當一隻有禮貌的爬蟲

- ▣ 過於頻繁、大量的送出 requests 會造成伺服器的負擔



Outline

A. 爬蟲基本介紹

- 爬蟲介紹與 HTML 初次見面
- 第一支爬蟲程式
- BeautifulSoup 的常用函數
- 正則表達式 (regular expression)

情境

- 又是難受的星期一，慣老闆一進到公司後，就要你把上周所有的新聞標題抓下來，要好好關心一下國家大事



- 剛上完 Python 爬蟲實戰的你，開始回想起上課的內容



標題文字在哪個標籤裡？

網頁名稱-python crawler x

file:///C:/Users/home/Dropbox/python_crawler0726/example01.html

應用程式 YouTube Yahoo奇摩 Facebook Udacity - Free Online Outlook Web App IIS, Academia Sinica Google

Python 爬蟲實戰

這是一段標籤的內容

這是 h3 標籤的內容

這是 p 標籤的內容

這是 div 標籤的內容，即使換行

但如果用了 br 標籤
就可以順利斷行了

我是有著屬性 class="zzz" 的標籤內容
我是有著屬性 id="value-of-attr" 的標籤內容

標頭 1 (table-header)	標頭 2 (table-header)	標頭 3 (table-header)	標頭 4 (table-header)
列2 欄1	列2 欄2 (我的屬性 class="zzz")	列2 欄3 (我是 a 標籤，屬性 href=網址)	列2 欄4 (資料協會)
列3 欄1	列3 欄2 <ul style="list-style-type: none">我是 li 標籤(列表)，屬性 class="zz"第二個 li 標籤	列3 欄3 (資料協會-python 爬蟲實戰)	列3 欄4 (我的屬性 class="zzzz")

上一頁(B) Alt+向左鍵
下一頁(F) Alt+向右鍵
重新載入(R) Ctrl+R
另存新檔(A)... Ctrl+S
列印(P)... Ctrl+P
投放(C)...
翻譯成中文 (繁體) (T)
檢視網頁原始碼(V) Ctrl+U
檢查(N) Ctrl+Shift+I

Elements Console Sources Network Performance »

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <h1>Python 爬蟲實戰</h1> == $0
    <h2>這是 h2 標籤的內容</h2>
    <h3>這是 h3 標籤的內容</h3>
    <p title="i-am-title">這是 p 標籤的內容</p>
    <div>
      這是 div 標籤的內容,
      html body h1
    
```

Styles Event Listeners DOM Breakpoints Properties

Filter :hover .cls +

element.style {

h1 { user agent stylesheet

display: block;
font-size: 2em;
-webkit-margin-before: 0.67em;
-webkit-margin-after: 0.67em;
-webkit-margin-start: 0px;
-webkit-margin-end: 0px;
font-weight: bold;

margin 21.440
border -
padding -
967.200 x 39.200
21.440

Filter Show all

display block
font-size 32px
font-weight bold
height 39.2px
width 967.2px

Console What's New x

Highlights from Chrome 59 update

範例 oo: 第一支爬蟲程式

- 如何透過程式，取出範例網頁中的大標題
「Python 爬蟲實戰」？

```
# import 套件
import requests
from bs4 import BeautifulSoup
# 用 requests 抓取網頁並存在 response
response =
requests.get("https://jimmy15923.github.io/
example_page")
# 用 BS4 解析 HTML 並把結果回傳 soup
soup = BeautifulSoup(response.text, "lxml")
# 印出 h1 標籤
print(soup.find("h1"))
```

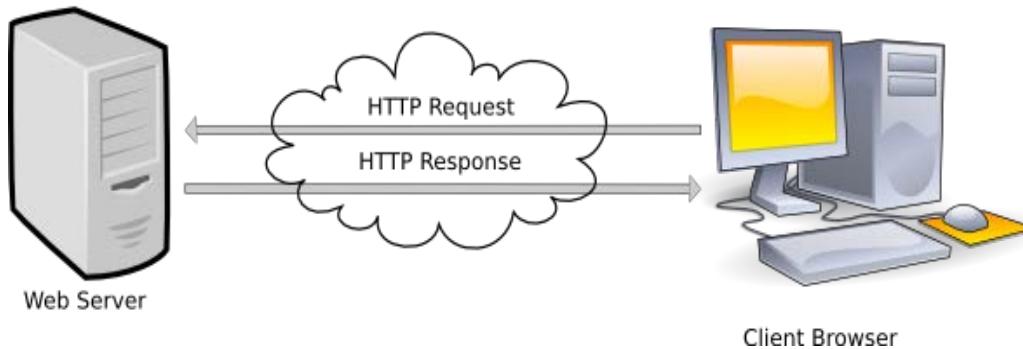
Requests 的功能

□ Requests: HTTP for Humans

```
response = requests.get("www.yahoo.com.tw")
```



www.yahoo.com.tw



□ 最常用的兩個方法

- GET
- POST



Requests 的常用函數

□ response.status_code



- 403 Forbidden

- 404 Not Found

□ response.encoding

- 如果是中文網站要特別注意編碼的問題

□ response.text

- 目標網頁的 HTML 文字



拿到目標網頁的 HTML 了！

```
In [13]: # 目標網站的 HTML  
response.text
```

```
Out[13]: '<!DOCTYPE html>\r\n<html>\r\n<head>\r\n<style>\r\n.abc {color: blue; font-size: 40px;}\r\n</style>\r\n</head>\r\n<body>\r\n<title>Page Title</title>\r\n<h1>This is a Heading</h1>\r\n<p>This is a paragraph.</p>\r\n<div> This is embeded in a div tag\r\nno matter I break it or not.\r\n</div>\r\n<p> But this will have a break </p>\r\n<p> Break</p>\r\n<p>python_crawler</p>\r\n<div class = "zzz" >\r\nHello! I am div with "class" attr!\r\n</div>\r\n<div id = "i-am-id">\r\nI am div with id attrs\r\n<table>\r\n<tr>\r\n<th>標頭 1 (table-header)</th>\r\n<th>標頭 2 (table-header)</th>\r\n<th>標頭 3 (table-header)</th>\r\n<th>標頭 4 (table-header)</th>\r\n</tr>\r\n<tr>\r\n<td> 列2 欄1 </td>\r\n<td class = "zzz"> 列2 欄2 my attributes class is zzz </td>\r\n<td> 列2 欄3 <a href = "http://www.google.com.tw"> 列2 欄4 (資料協會) </a> </td>\r\n<td value = "5566">列3 欄1 </td>\r\n<td>列3 欄2<br>\r\n<td>列3 欄3 (資料協會-python 跳蟲實戰)</td>\r\n<td class = "zzzz"> 列3 欄4 (my attributes class is zzzz) </td>\r\n</tr>\r\n</table>\r\n</div>\r\n<p>python_crawler</p>\r\n</body>\r\n</html>'
```

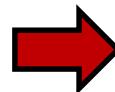


BeautifulSoup 登場!

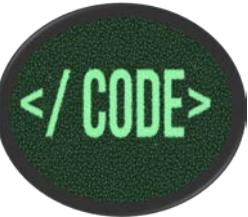
- 強大且簡單易學的 HTML 解析器
- 將 HTML 轉變成 BeautifulSoup 物件
- 再用 BeautifulSoup 的函數取得想要的標籤資訊

```
In [13]: # 目標網站的 HTML  
response.text
```

```
Out[13]: '<!DOCTYPE html>\\r\\n<html>\\r\\n<head>\\r\\n<style>\\r\\n    .abc {\\r\\n        color: blue;\\r\\n        font-size: 40px;\\r\\n    }\\r\\n</style>\\r\\n<body>\\r\\n<title>Page Title</title>\\r\\n<h1>This is a Head\\ng</h1>\\r\\n<p>This is a paragraph.</p>\\r\\n<div>\\r\\n    This is embedded in a div tag!\\r\\n    no matter I break it or not.\\r\\n</div>\\r\\n<div>\\r\\n    But this will have a break </p>\\r\\n    Break</p>\\r\\n<p>python_crawler</p>\\r\\n<div class = "zzz" >\\r\\n    Hello! I\\r\\n    am div with "class" attr!\\r\\n</div>\\r\\n<div id = "i-am-id">\\r\\n    I am div with id attrs\\r\\n<table>\\r\\n    <tr>\\r\\n        <th>標頭 1 (table-header)</th>\\r\\n        <th>標頭 2 (table-header)</th>\\r\\n        <th>標頭 3 (table-header)</th>\\r\\n        <th>標頭 4 (table-h\\r\\n            <th>\\r\\n                <td>\\r\\n                    <tr>\\r\\n                        <td>\\r\\n                            <td class = "zzz" >\\r\\n                                <td> my attributes class is zzz </td>\\r\\n                            <td>\\r\\n                                <a href = "http://www.google.com.tw" >\\r\\n                                    <a href = "http://foundation.datasci.tw/">\\r\\n                                        </a>\\r\\n                                </td>\\r\\n                            </td>\\r\\n                        </tr>\\r\\n                    </td>\\r\\n                </td>\\r\\n            </tr>\\r\\n        </td>\\r\\n    </tr>\\r\\n</table>\\r\\n<div>\\r\\n    <p>python_crawler</p>\\r\\n</div>'
```



練習 oo: 淺嘗 BeautifulSoup



請輸入以下的 codes 並 print 出結果

```
print(soup.find("h1"))
print(soup.h1)
print(soup.html.h1)
print(soup.body.h1)
print(soup.html.body.h1)
```

- BeautifulSoup 可以直接找出你想要的 tags
而不需告訴他路徑



Outline

A. 爬蟲基本介紹

- 爬蟲介紹與 HTML 初次見面
- 第一支爬蟲程式
- BeautifulSoup 的常用函數
- 正則表達式 (regular expression)

讓 BeautifulSoup 更好喝的函數 - find()

- `find(tag, attribute, recursive, text, keywords)`

Signature: `soup.find(name=None, attrs={}, recursive=True, text=None, **kwargs)`

Docstring:

Return only the first child of this Tag matching the given

- 當你想要找一個標籤

`soup.find("tag")`

≡

`soup.tag`

- 當你想要找一個標籤且屬性為特定值

`soup.find("tag", {"attr": "value"})` ≡ `soup.find("tag", attr="value")`



讓 BeautifulSoup 更好喝的函數 - find_all()

- 一個 tag 不夠，想找多個怎麼辦？
- find_all() 可以幫上忙



Signature: soup.find_all(name=None, attrs={}, recursive=True, text=None, limit=None, **kwargs)

Docstring:

Extracts a list of Tag objects that match the given criteria. You can specify the name of the Tag and any attributes you want the Tag to have.

- 用法跟 find() 一樣，但是回傳的是 Python 的 list



找到標籤了！

- ❑ 其實我們不在乎標籤本身，我們要的是那個躲在標籤裡面的內容或是屬性！

```
In [753]: 1 print(soup.find("h1"))
```

```
<h1>Python 爬蟲實戰</h1>
```

- ❑ 加上 .text 後，成功拿到這個標籤的內容

```
In [754]: 1 print(soup.find("h1").text)
```

```
Python 爬蟲實戰
```

範例 01: BeautifulSoup 的常用函數

```
# 找出第一個 td 的標籤  
print(soup.find("td"))
```

```
<td> 列2 欄1 </td>
```

```
# 找出第一個 td 的標籤並印出其文字內容  
print(soup.find("td").text)
```

```
列2 欄1
```

```
# 找出所有 td 的標籤
```

```
print(soup.find_all("td"))
```

```
[<td> 列2 欄1 </td>, <td class="zzz"> 列2 欄2 my attributes class is zzz </td>, <td>  
<a href="http://www.google.com.tw"> 列2 欄3 </a>  
</td>, <td>  
<a href="http://foundation.datasci.tw/"> 列2 欄4 (資料協會) </a>  
</td>, <td value="5566">列3 欄1 </td>, <td>列3 欄2  
    <oi>  
<li class="zz"> my attributes class is zz </li>  
<li> new cell 2 </li>  
</oi></td>, <td>  
<a href="http://foundation.datasci.tw/python-ml-170812/"> 列3 欄3 (資料協會-python 蟑蟲實戰) </a>  
</td>, <td class="zzzz"> 列3 欄4 (my attributes class is zzzz) </td>]
```

範例 01: BeautifulSoup 的常用函數

```
# 不指定標籤，但找出所有屬性 class = "zzz" 的標籤  
print(soup.find_all("", {"class": "zzz"}))
```

```
[<div class="zzz">  
Hello! I am div with "class" attr!  
</div>, <td class="zzz"> 列2 欄2 my attributes class is zzz </td>]
```

```
# 找出所有 td 標籤的第三個並找出其中的 a 標籤  
print(soup.find_all("td")[2].find("a"))
```

```
<a href="http://www.google.com.tw"> 列2 欄3 </a>
```

範例 01: BeautifulSoup 的常用函數

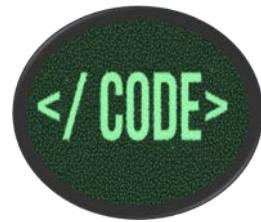
```
# 找出所有內容等於 python_crawler 的文字  
print(soup.find_all(text="python_crawler"))
```

```
['python_crawler', 'python_crawler']
```

```
# 找出第一個 a 標籤並印出屬性  
print(soup.find("a").attrs)  
print(soup.find("a")["href"])
```

```
{'href': 'http://www.google.com.tw'}  
http://www.google.com.tw
```

練習 01: BeautifulSoup 的常用函數 (8 mins)



□ 請觀察範例網頁後，嘗試回答以下的問題

- 請計算範例網頁中，共含有幾個 "td" 的標籤 (tags)?
- 請找出 "div" 標籤，且屬性 (attributes) id = "i-am-id" 的文字內容？
- 請找出列 3 欄 3 背後的超連結網址？(在網頁上面按右鍵 → 檢查，有哪些輔助資訊可以幫我們找到這個標籤?)



練習 01: 答案

```
# 找出所有 td 標籤，並用 len 計算長度  
print(len(soup.find_all("td")))
```

8

```
# 找到 div 標籤，屬性 id = "id1"，再印出其內容  
print(soup.find("div", id="id1").text)
```

我是有著屬性 class="zzz" 的標籤內容

```
# 透過觀察網頁可以發現 列3欄3 有個 id = hyperlink 可以幫助我們定位這個 tag，再把 tag 的 href 找出來  
print(soup.find("a", {"id": "hyperlink"})["href"])
```

<http://foundation.datasci.tw/python-crawling-170813/>

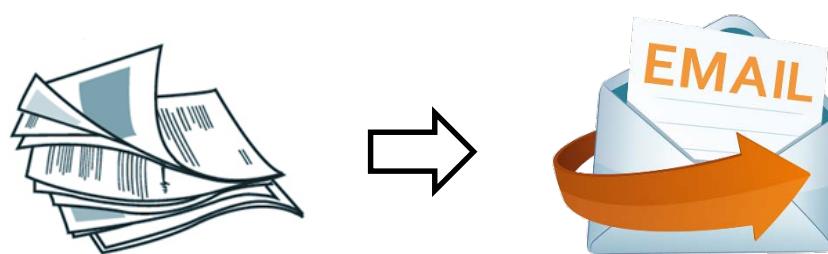
Outline

A. 爬蟲基本介紹

- 爬蟲介紹與 HTML 初次見面
- 第一支爬蟲程式
- BeautifulSoup 的常用函數
- 正則表達式 (regular expression)

情境

- 慣老闆一大清早就跑過來，交給你上百份文件，要你從這些文件裡面，整理出所有客戶的 Email 清單，怎麼辦？



- 而這對剛上完 Python 爬蟲實戰的你，毫無困難



BeautifulSoup 的調味料: 正則表達式

- 甚麼是 regular expression?

[A-Za-zo-9._]+@[A-Za-z.]+\.(com|edu)+\.\tw



- 別害怕! 我們一步一步來



regular expression 的威力

REGULAR EXPRESSION

3 matches, 1967 steps (~4ms)

/ [A-Za-z0-9._+]+@[A-Za-z\.\.]+\.(com|edu)+\.\.tw / g

TEST STRING

SWITCH TO UNIT TESTS ▶

```
"""
swc@iis.sinica.edu.tw02-27883799#1712Big Data Analytics/ Deep
LearningSocial Computing / Computational Social Science /
Crowdsourcing
Multimediaand Network SystemsQuality of ExperienceInformation
SecurityPh.D. candidate at NTU EEchihfan02-27883799#1602Camera
CalibrationComputer VisionData
Analysisicmchang02-27883799#1671System OptimizationMachine
LearningyusraBig data
analysisiccclin02-27883799#1668Data Analysisrusi02-
27883799#1668Government Procurement ActFinancial|
AnalysisPsychology & NeuroscienceMarketingxinchinchenEmbedded
Systemkyoyachuan062602-27883799
#1601FinTechActuarial ScienceData Analysiskai0604602-
27883799#1601Data AnalysisMachine Learningchloe02-
27839427Accountingafun02-27883799 afun@iis.sinica.edu.tw
#1673Data AnalysisWeb developmentyunhsu198902-
27883799#1668MarketingTIGP Ph.D. Fellow at Academia Sinica &
NCCUBaowalyMachine LearningData AnalysisSocial
Computingchangyc1427883799#1678
Data Analysisjimmy1592302-2788379
jimmy15923@iis.sinica.com.tw#1688Data
AnalysisjasontangAnalysisMachine Learninguchen02-27883799#1668Deep
Learningpjwu02-27883799#1604Computational PhotographyData Analysis
....
```

regular expression 常用符號

□ regular expression 使用許多符號來訂定搜尋規則，要學會使用就必須知道符號的意義

符號	意義	範例	符合範例的字串
*	前一字元或括號內字元出現 0次或多次	a^*b^*	aaaab 、 aabb 、 bbb
+	前一字元或括號內字元出現 1次或多次	$a+b^+$	aaabb 、 abbbb 、 abbbbb
{m,n}	前一字元或括號內字元出現 m次到 n 次 (包含 m, n)	$a\{1,2\}b\{3,4\}$	abbb 、 aabbbb 、 aabbb
[]	符合括號內的任一字元	$[A-Z]^+$	APPLE 、 QWER
\	跳脫字元	$\backslash\backslash\backslash\backslash$. \\"
.	符合任何單一字元 (符號, 數字, 空格)	$a.c$	auc 、 abc 、 a c

Python 的 re

- Python 有內建的 regular expression 函數
- 推薦使用 re.findall()

Signature: `re.findall(pattern, string, flags=0)`

Docstring:

Return a list of all non-overlapping matches in the string.

If one or more capturing groups are present in the pattern, return a list of groups; this will be a list of tuples if the pattern has more than one group.

```
# 找出所有內容等於 python_crawler 的文字
pattern = "我寫好的 regular expression"
string = "我想要找的字串"
re.findall(pattern, string)
```

範例 02-1: *, +, {} 的用法

- * 代表前面的字元可出現零次以上
- + 代表前面的字元至少要出現一次以上
- {m,n} 代表前面的字元可出現 m 次 ~ n 次 (包含)

In [759]:

```
1 import re
2 pattern = "a+b*c"
3 test_string = 'find aabc, ac, skip abb, dd'
4 re.findall(pattern, test_string)
```

Out[759]: ['aabc', 'ac']

練習 02-1: 答案

□ a^*b+c

- a 出現零次以上
- b 出現一次以上
- c 出現一次

```
In [17]: ### your codes
pattern = "a*b+c"
test_string = 'find abbbbc, bc, skip c, acc'
re.findall(pattern, test_string)
```

```
Out[17]: ['abbbbc', 'bc']
```

範例 02-1: 找到數字

- [] 代表的意思是這個字元可以是括號內的任何一個
- [0-9] 代表可以是 0~9 之間的任意數字
- [a-z] 代表可以是 a~z 之間的任意文字

In [760]:

```
1 pattern = "[0-9]+"
2 test_string = '12 drummers drumming, 11 pipers piping, 10 lords a-leaping'
3 re.findall(pattern, test_string)
```

Out[760]: ['12', '11', '10']

練習 02-2: 答案

□ [0-9]+

□ 符合 0-9 任一數字，且須出現一次以上

□ [1-3]+

□ 符合 1-3 任一數字，且須出現一次以上

□ 123

□ 符合 123 的字元

```
In [76]: ## TODO
    pattern = "123"
    test_string = 'abc123xyz, de123fine"123", test = 123'
    re.findall(pattern, test_string)
```

```
Out[76]: ['123', '123', '123', '123']
```

範例 02-3: 找到文字

- 當有指定的文字需要搜尋，可透過 [] 搭配 *, +, {} 進行搜尋

```
In [761]: 1 pattern = "[cmf]an"
            2 test_string = 'find: can, man, fan, skip: dan, ran, pan'
            3 re.findall(pattern, test_string)
```

```
Out[761]: ['can', 'man', 'fan']
```

```
In [762]: 1 pattern = "jim{2,5}y"
            2 test_string = 'find: jimmy, jimmmy, jimmmyy, skip: jimy'
            3 re.findall(pattern, test_string)
```

```
Out[762]: ['jimmy', 'jimmmy', 'jimmmmy']
```

練習 02-3: 答案

□ [A-Z]+[a-z]

- 前兩個字元要是 A-Z 其中一個，且出現一次以上
- 最後一個字元要是 a-z 其中一個

```
In [82]: # TODO
pattern = "[A-Z]+[a|z]"
test_string = 'find: ABi, BBC, CNn, skip: ai, be, cd'
re.findall(pattern, test_string)

Out[82]: ['ABi', 'BBC', 'CNn']
```

範例 02-4: 跳脫符號

- 萬一我今天想要找到的就是 + 這個符號怎麼辦？
- 在前面加上一個跳脫符號 \
- . 代表任何字元 (符號、數字、空格)

In [763]:

```
1 pattern = ".{3}\."
2 test_string = 'find: 591., dot., yes., skip: non!'
3 re.findall(pattern, test_string)
```

Out[763]: ['591.', 'dot.', 'yes. ']

練習 02-4: 答案

□ [A-Z]\+[a-z]

- 第一個字元要是 A-Z 其中一個
- 第二個字元要是 + 號
- 第三個字元要是 a-z 其中一個

```
In [85]: # TODO
pattern = "[A-Z]\+[a-z]"
test_string = 'find: A+c, B+d, C+x'
re.findall(pattern, test_string)
```

```
Out[85]: ['A+c', 'B+d', 'C+x']
```

範例 02-5: 條件式搜尋

□ | 代表左右邊只要任一符合條件即可

```
In [766]: 1 pattern = "I love cats|I love dogs"
           2 test_string = 'find: I love cats, I love dogs, skip: I love logs, I love cogs'
           3 re.findall(pattern, test_string)
```

```
Out[766]: ['I love cats', 'I love dogs']
```

練習 02-5: 答案

□ `jimy|jim{3}y`

- 符合 `jimy`
- 或是符合 `jimmmy`

```
In [89]: pattern = "jimy|jim{3}y"
test_string = 'find: jimy, jimmmy, skip: jimmy, jimmmy'
re.findall(pattern, test_string)
```

```
Out[89]: ['jimy', 'jimmmy']
```

Email 的 regular expression (1/2)

□ [A-Za-zo-9._]+@[A-Za-z.]+(com|edu)\.tw

□ [A-Za-zo-9._]+@

□ 大小寫英文、數字、點、底線，可出現一次以上，
且加上 @

REGULAR EXPRESSION

4 matches, 105 steps (~0ms)

// [A-Za-z0-9._]+@ / g

TEST STRING

SWITCH TO UNIT TESTS ▾

```
jimmy15923@iis.sinica.edu.tw
jimmy15923.tem98@test.com.tw
afun@iis.sinica.edu.tw
afun@test.xyz.tw
```

Email 的 regular expression (2/2)

□ [A-Za-zo-9._]+@[A-Za-z.]+(com|edu)\.tw

□ [A-Za-z.]+(com|edu)\.tw

□ 大小寫英文、點，可出現一次以上，且結尾須為 com 或 edu 再加上 .tw，

REGULAR EXPRESSION 3 matches, 125 steps (~1ms)

```
/ @[A-Za-z.]+(com|edu)\.tw / g
```

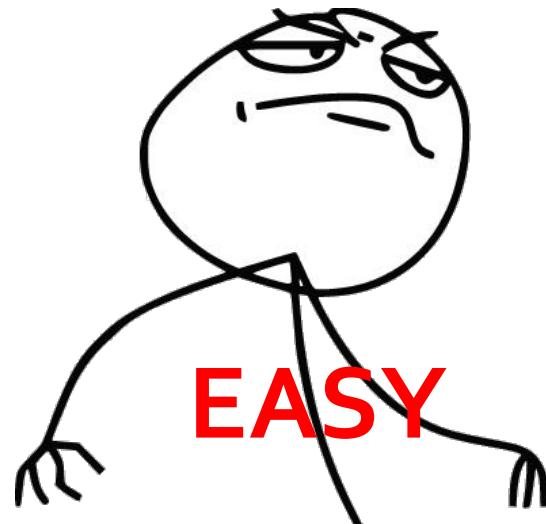
TEST STRING SWITCH TO UNIT TESTS ▾

```
jimmy15923@iis.sinica.edu.tw
jimmy15923.tem98@test.com.tw
afun@iis.sinica.edu.tw
afun@test.xyz.tw
```

甚麼是 regular expression?

- 再看一次!

[A-Za-zo-9._]+@[A-Za-z.]+\.(com|edu)+\.\tw



練習 02: regular expression (10 mins) </ CODE >

- 請觀察 518 黃頁網，並找出所有店家的電話號碼
(包含分機)
 - 請透過開發者工具，觀察目標資訊藏在那些標籤底下？
 - 標籤內有甚麼特別的屬性可以讓我們找到嗎？
 - 怎麼把包含電話號碼的 list 合併成一個大的 test_string？

```
# 非常實用的 for loop 寫法，當你使用 find_all 後，  
想一口氣把 list 裡面所有 tags 的內容文字取出時，  
可以這樣寫
```

```
[tag.text for tag in soup.find_all("tag")]
```



練習 02: 答案

```
# 用 requests 抓取網頁並存在 response
# 用 BeautifulSoup 解析 HTML 並把結果回傳 soup
response = requests.get("http://yp.518.com.tw/service-life.html?ctf=10")
soup = BeautifulSoup(response.text,"lxml")

# 抓到所有 li 標籤，且屬性 class=comp_tel，並存在 list 裡
all_phone_text = [tag.text for tag in
soup.find_all("li", {"class":"comp_tel"})]

# 將 list 的所有文字，存為一個 string
all_phone_text = "".join(all_phone_text)

# 用 regular expression 找出所有電話號碼
phone_number = re.findall("0[1-9]+-[0-9]+",
all_phone_text)
```



Session B 即將在 10:50 開始...

- 請先使用 jupyter notebook 打開 Session_B 資料夾裡面的 **Session_B.ipynb** 檔案，並執行第一段 code，確定套件都能順利 import
- 還沒有下載 code 的同學，請到以下網址下載或是教室前方有隨身碟可以使用

<https://goo.gl/e5csuH>





爬蟲實戰練習與資料分析



Outline

B. 爬蟲實戰練習與資料分析

- BeautifulSoup + regular expression
- GET vs. POST
- 使用 pandas 儲存資料
- 文字探勘與資料分析

情境

- 自從慣老闆發現你很好用之後，某天又突然跑過來，這次是要請你把美食網站上面，所有店家的地址都抓下來，而且老闆限定期是新北市的喔！



- 同樣的，這對剛上完 Python 爬蟲實戰的你，也是小菜一碟



BeautifulSoup + regular expression

- BeautifulSoup 可以幫您解析 HTML
- regular expression 可以按照您的規則回傳字串



- 兩個加在一起，就可以按照想要的規則取出目標標籤



範例 03: BeautifulSoup + regular expression

```
# import 套件
import re

# 用 regular expression 找出所有 td 或 tr 標籤
soup.find_all(re.compile("t(d|r)"))
```

- 運用 BeautifulSoup 加上 regular expression，可以抓到任何給定條件的資訊
- 在 BeautifulSoup 裡必須使用 re.compile 來寫 pattern
- re.compile 可以用來尋找 tags, attrs 及 contents

範例 03: BeautifulSoup + regular expression

```
# 找出所有屬性為 class 且值包含至少一個 z 以上的標籤  
soup.find_all("", {"class":re.compile("z+")})
```

```
In [15]: soup.find_all("", {"class":re.compile("z+")})
```

```
Out[15]: [<div class="zzz">  
    Hello! I am div with "class" attr!  
    </div>,  
    <td class="zzz"> 列2 欄2 my attributes class is zzz </td>,  
    <li class="zz"> my attributes class is zz </li>,  
    <td class="zzzz"> 列3 欄4 (my attributes class is zzzz) </td>]
```

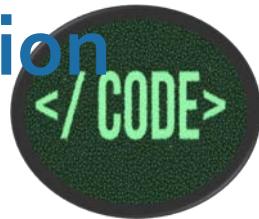
```
# 用 regular expression 找出所有包含 python 的  
contents
```

```
soup.find_all("", text=re.compile("python"))
```

```
In [16]: print(soup.find_all("", text = re.compile("python")))
```

```
['python_crawler', '列3 欄3 (資料協會-python 爬蟲實戰)', 'python_crawler']
```

練習 03: BeautifulSoup + regular expression (10 mins)



1. 請找出範例網頁中所有標籤，其屬性 href 的值包含資料科學協會的網址 ("http://foundation.datasci.tw/...")
 - 要怎麼找到屬性是 href 的所有標籤？
 - 要如何把 regular expression 加進 BeautifulSoup ?
2. 請觀察518 黃頁網，並找出所有位在新北市的店家地址
 - 地址的資訊都藏在哪些 tags 底下？
 - 怎麼把 regular expression 用在 contents 上面？



練習 03-1: 答案

```
# 萬年起手式，先 requests 再用 BeautifulSoup 解析
response = requests.get(
    "https://jimmy15923.github.io/example_page")
soup = BeautifulSoup(response.text, "lxml")

# 不指定標籤，找到屬性 href，值為特定網址
print(soup.find_all("", {"href":re.compile(
    "http://foundation.datasci.tw")}))
```

```
[<a href="http://foundation.datasci.tw/"> 列2 欄4 (資料協會) </a>,
 <a href="http://foundation.datasci.tw/python-ml-170812/" id="hyperlink"> 列3 欄3 (資料協會-pyton 爬蟲實戰)</a>]
```

練習 03-2: 答案

```
# 萬年起手式，先 requests 再用 BeautifulSoup 解析
response =
requests.get("http://yp.518.com.tw/service-
life.html?ctf=10")
soup = BeautifulSoup(response.text, "lxml")
```

```
# 找到標籤 li，且屬性 class=comp_loca，內容包含
"新北"
```

```
print(soup.find_all("li", {"class": "comp_loca"},  
text=re.compile("新北")))
```

```
[<li class="comp_loca">新北市 / 永和區 永亨路49號</li>,
<li class="comp_loca">新北市 / 板橋區 僑中一街124巷62-20號</li>,
<li class="comp_loca">新北市 / 板橋區 僑中一街124巷62-20號</li>,
<li class="comp_loca">新北市 / 板橋區 國光路39號</li>]
```

Outline

B. 爬蟲實戰與資料分析

- Beautiful Soup + regular expression
- GET vs. POST
- 使用 pandas 儲存資料
- 文字探勘與資料分析

情境

- 慣老闆又出現啦!這次想要請你把某個時段的高鐵時刻表全部找出來，方便以後出差的時候參考
- 你找到高鐵時刻表網站，開心的想起萬年起手式，開始 import requests，request.get，這時...



不會改變的網址

□ 上網時常發現，許多操作像是搜尋、點選等均都是在同一個網址底下完成，這種網頁該如何用我們之前學的 requests 來爬取？

<http://www.thsrc.com.tw/tw/TimeTable/SearchResult>

時刻表與票價查詢

請選擇查詢條件

出發站 : <input type="text" value="請選擇"/>	日期 : <input type="text" value="2017/06/25"/>	<input type="button" value="立即查詢"/>
到達站 : <input type="text" value="請選擇"/>	時間 : <input type="text" value="16:00"/> 出發	
適用優惠 : <input type="checkbox"/> 早鳥 <input type="checkbox"/> 大學生 <input type="checkbox"/> 25人團體 <input type="checkbox"/> 校外教學 <input type="checkbox"/> 平日離峰 <input type="checkbox"/> 學生暑期	時刻表及票價資訊下載	
<ul style="list-style-type: none">• 2017/7/1 起適用時刻表• 2017/4/28~6/30 通用時刻表		

公告事項說明

- 南港-台北、南港-板橋、台北-板橋間恕不發售商務車廂車票及團體票。
- 欲得知列車停靠站詳細說明，請點選上方之車次連結。
- 本系統僅提供時刻表及票價查詢功能，如欲直接訂位，請使用台灣高鐵網路訂位系統進行線上訂位及付款。
- 本時刻表所列各次列車為開車時刻，唯於終點站為到達時刻。

票價優惠說明

- 早鳥優惠車票，自祭車日(含)前28日起開始限量發售，最晚發售至祭車日(含)前5日截止。
- 本首頁顯示之早鳥適用優惠為該車次最低車票折扣，愈早訂位者，愈有機會訂購早鳥 65 折優惠車票。早鳥 65 折銷售完畢即改發售早鳥 8 折，早鳥 8 折銷售完畢即改發售早鳥 9 折，早鳥 9 折銷售完畢即提前截止並改發售原價車票。
- 以上票價為折扣後之價格。
- 以上列車皆為低價票種性車票，資訊若有更動，以各車站相場小票為準。

<http://www.thsrc.com.tw/tw/TimeTable/SearchResult>

您的查詢結果

南港站 ➤ 台南站 2017/06/25(周一) 16:00 出發	<input type="button" value="檢視 2017/06/25 時刻表 (含南下/北上車次)"/>
-----------------------------------	---

車次資訊				適用優惠	備註
車次	出發時間	抵達時間	行車時間	早鳥	
0841	16:00	18:11	02:11		
0661	16:10	18:06	01:56		
0663	16:35	18:32	01:57		
1241	16:40	18:17	01:37		
0845	17:00	19:11	02:11		
0667	17:10	19:06	01:56		
0669	17:35	19:32	01:57		
1245	17:40	19:17	01:37		

為何網址不會改變?

□ 兩種常見原因

- 網頁透過 POST 的方式取得資料，先由瀏覽器在背景送一些資料給 Server，Server 收到 POST 請求後，回傳相對應的資料
- 現代的網頁為了提升使用者體驗，會運用 JavaScript, AJAX 等技術，來動態載入資料而不需重新整理網頁

GET vs. POST

□ GET: 發送 requests，Server 回傳資料

□ URL 會隨著不同的網頁改變

1. <http://www.taipeibo.com/yearly/2017>

2. <http://www.taipeibo.com/yearly/2016>

□ POST: 發送 requests 並附帶資料，Server 回傳資料

□ 網址不會改變，但是網頁資料會隨著使用者不同的 requests 改變

如何知道 request methods? (1/4)

1. 右鍵→檢查

The screenshot shows the Taiwan High-Speed Rail website's search interface. A context menu is open over the search input field, with the '檢查(N)' option highlighted. The search form includes fields for date (2017/08/07), time (16:30), departure (出發), and arrival (抵達). Below the search form, a table displays travel information for a specific route and time.

網站導覽 | 繁體中文 | 日本語 | English | 請輸入查詢關鍵字 | | [粉絲專區](#)

臺灣高鐵 TAIWAN HIGH SPEED RAIL

優惠活動 | 購票資訊 | 乘車指南 | 關於高鐵 | 投資人關係 | [搭高鐵遊台灣](#) | [24h 網路訂票](#)

首頁 > 購票資訊 > 快速查詢 > 時刻表與票價查詢 | [列印本頁](#)

時刻表與票價查詢

請選擇查詢條件

日期 : 2017/08/07 | 時間 : 16:30 | 出發 | 立即查詢

25人團體 | 校外教學 | 平日離峰 | 學生暑期

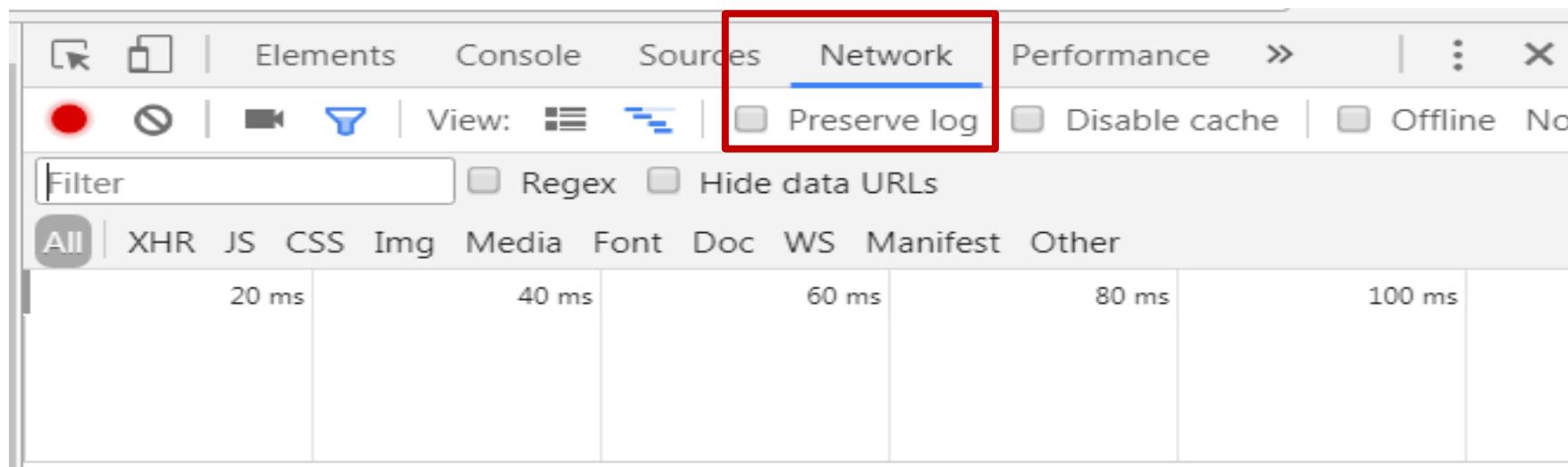
時刻表及票價資訊下載
▪ 2017/8/1起適用時刻表

7/08/07(周一) 16:30 出發 | 檢視 2017/08/07 時刻表 (含南下/北上車次)

資訊	適用優惠	備註
抵達時間	行車時間	早鳥
0845	17:00	18:27
18:27	01:27	

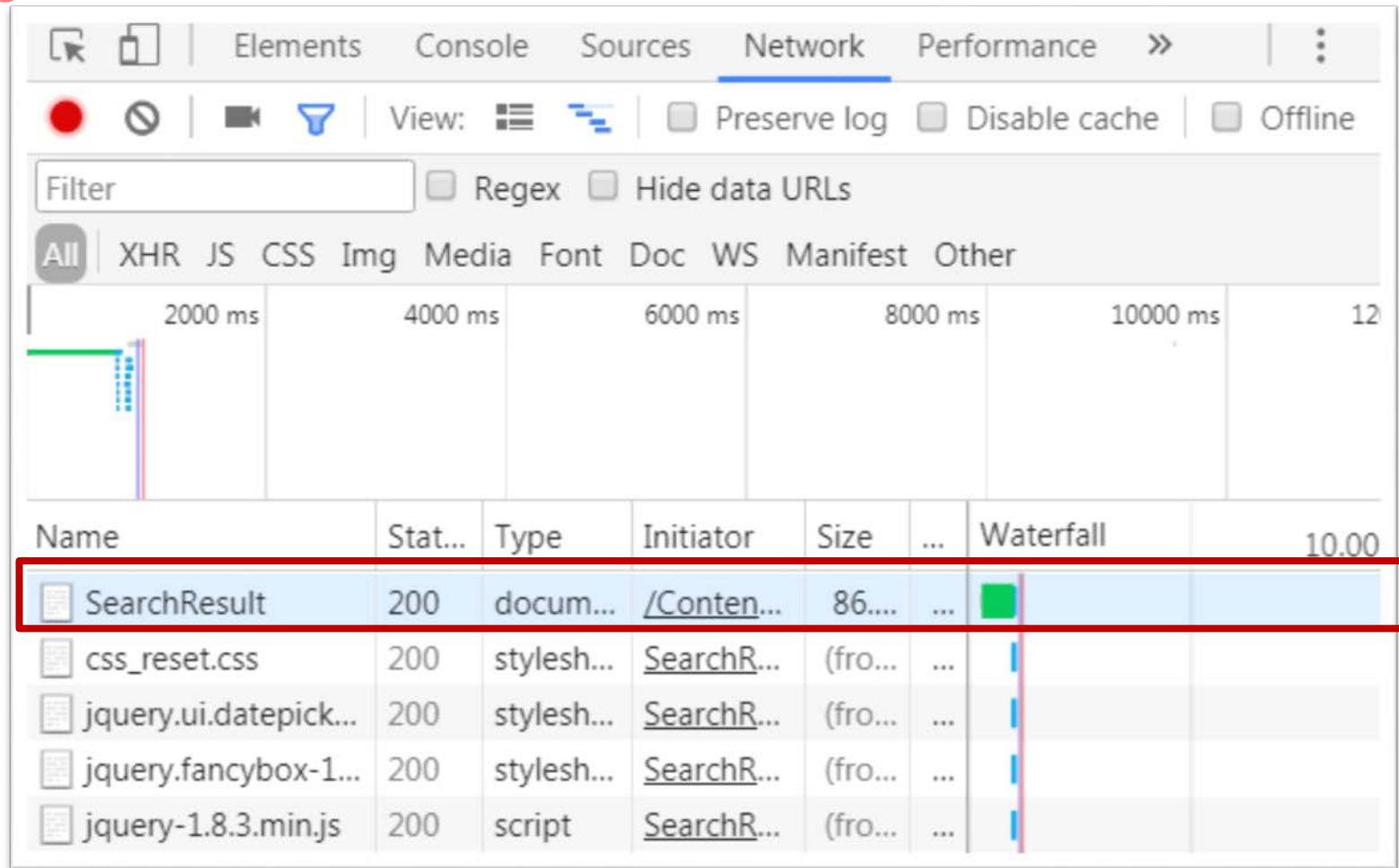
如何知道 request methods? (2/4)

- 選擇 Network 並勾選底下的 Preserve log



如何知道 request methods? (3/4)

3. 設定要查詢的資料後送出



The screenshot shows the Network tab of the Google Chrome DevTools. The timeline at the top indicates a total duration of 12 seconds. Below the timeline is a table of network requests:

Name	Status	Type	Initiator	Size	Waterfall
SearchResult	200	document	/Content...	86...	10.00
css_reset.css	200	stylesheet	SearchR...	(from ...)	
jquery.ui.datepick...	200	stylesheet	SearchR...	(from ...)	
jquery.fancybox-1...	200	stylesheet	SearchR...	(from ...)	
jquery-1.8.3.min.js	200	script	SearchR...	(from ...)	

如何知道 request methods? (4/4)

4. 上方的標籤選 Headers, 可以看到 Request Method 是 POST!

X Headers Preview Response Cookies Timing

▼ General

Request URL: <http://www.thsrc.com.tw/tw/TimeTable/SearchResult>

Request Method: POST

Status Code: 200 OK

Remote Address: 61.31.57.189:80

Referrer Policy: no-referrer-when-downgrade



Form Data

- Form Data 是我們剛剛在做 POST 時，傳送給 Server 的資料

```
▼ Form Data      view source      view URL encoded
```

```
StartStation: 2f940836-cedc-41ef-8e28-c2336ac8fe68
```

```
EndStation: a7a04c89-900b-4798-95a3-c01c455622f4
```

```
SearchDate: 2017/07/27
```

```
SearchTime: 14:30
```

```
SearchWay: DepartureInMandarin
```

```
RestTime:
```

```
EarlyOrLater:
```

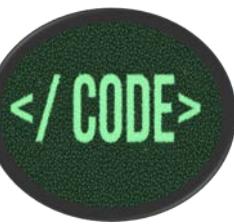
- Server 看到你的 POST，且透過你附帶的 Form data，回傳你想要查詢的 HTML 結果

範例 04: 如何使用 POST

```
# 觀察網頁後，找到 option 的值
form_data = {
    "StartStation": "2f940836-cedc-41ef-8e28-c2336ac8fe68",
    "EndStation": "e6e26e66-7dc1-458f-b2f3-71ce65fdc95f",
    "SearchDate": "2017/08/13",
    "SearchTime": "20:30",
    "SearchWay": "DepartureInMandarin"}

# requests 改用 POST，並放入 form_data
response_post = requests.post("https://www.thsrc.com.tw/TimeTable/SearchResult", data = form_data)
soup_post = BeautifulSoup(response_post.text, "lxml")
```

練習 04: 如何使用 POST (8 mins)



- 請運用 POST 方式，找出 2017 年 8 月 14 日 21:30，南港站到台南站共有幾個班次？
 - 觀察南港、台南站的 option value 是甚麼？
 - 查看班次的資訊都藏在哪些標籤內？



練習 04: 答案

```
# 將要查詢的資料寫成 dictionary
form_data = {
    "StartStation": "2f940836-cedc-41ef-8e28-c2336ac8fe68",
    "EndStation": "9c5ac6ca-ec89-48f8-aab0-41b738cb1814",
    "SearchDate": "2017/08/14",
    "SearchTime": "21:30",
    "SearchWay": "DepartureInMandarin"}
```



```
# requests 改用 POST，並放入 form_data
response_post = requests.post("https://www.thsrc.com.tw/tw/TimeTable/SearchResult", data = form_data)
soup_post = BeautifulSoup(response_post.text, "lxml")
```



```
# 找出有幾個 td 標籤，屬性為 class=column1
print(len(soup_post.find_all("td", class_="column1")))
```

Outline

B. 爬蟲實戰與資料分析

- BeautifulSoup + regular expression
- GET vs. POST
- ▣ 使用 pandas 儲存資料
- 文字探勘與資料分析

情境

- 辛苦抓完所有東西後，慣老闆卻要求說，要把資料全部都存成表格，這樣我才能方便用 excel 打開還有編輯啊！



- 所幸，Python 爬蟲實戰還是有教過怎麼做



pandas 登場

- ❑ pandas 是一個基於 numpy 的函式庫，不論是用來
讀取、處理數據都非常的簡單方便



- ❑ 處理結構化的數據非常好用!



pandas 的核心 - DataFrame

In [37]: df1

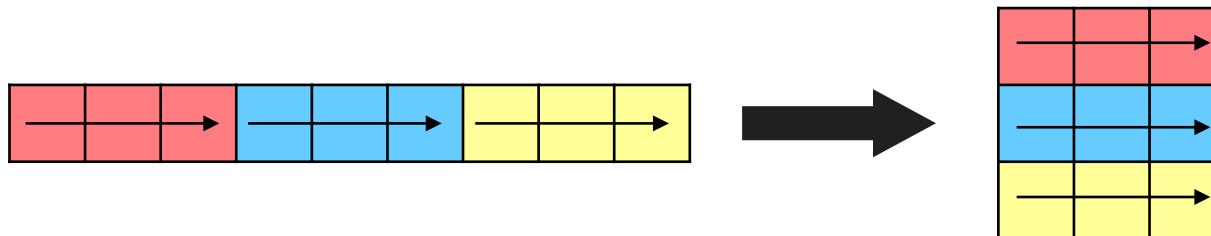
Out[37]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	47215	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

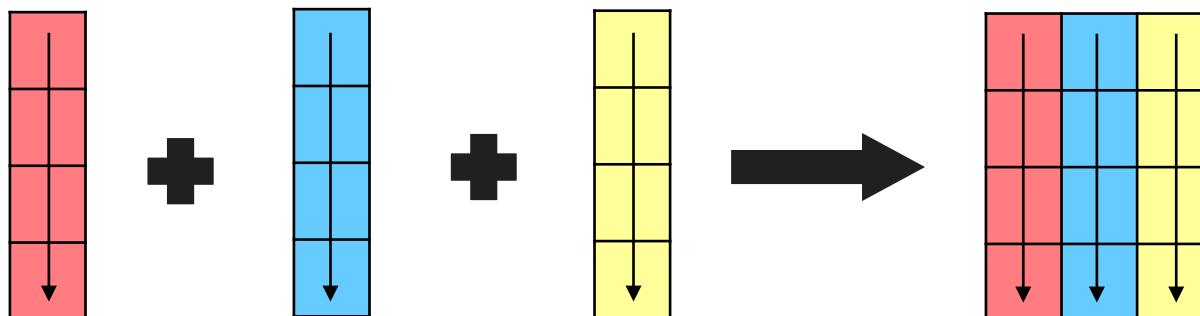
- DataFrame 就是由 rows 跟 columns 所組成的一個大表格

DataFrame 的組成

- 一個 DataFrame 的組成可以分成兩種情形來看
 - 由 rows 組成



- 由 columns 組成



如何用 pandas 儲存爬下來的資料？

- 如果爬下來的資料是以 row 為單位

```
In [42]: df = pd.DataFrame(columns= ["column1", "column2", "column3"])
df
```

```
Out[42]:
```

	column1	column2	column3
--	---------	---------	---------

```
In [46]: df.loc[0] = [1, 2, 3]
df
```

```
Out[46]:
```

	column1	column2	column3
0	1.0	2.0	3.0

```
In [47]: df.loc[1] = [4, 5, 6]
df
```

```
Out[47]:
```

	column1	column2	column3
0	1.0	2.0	3.0
1	4.0	5.0	6.0

把爬下來的資料變成表格

□ 如果爬下來的資料是以 column 為單位

```
In [42]: col_1 = [1.0, 4.0]
          col_2 = [2.0, 5.0]
          col_3 = [3.0, 6.0]
```

```
In [43]: df = pd.DataFrame({ "column1": col_1,
                           "column2": col_2,
                           "column3": col_3})
```

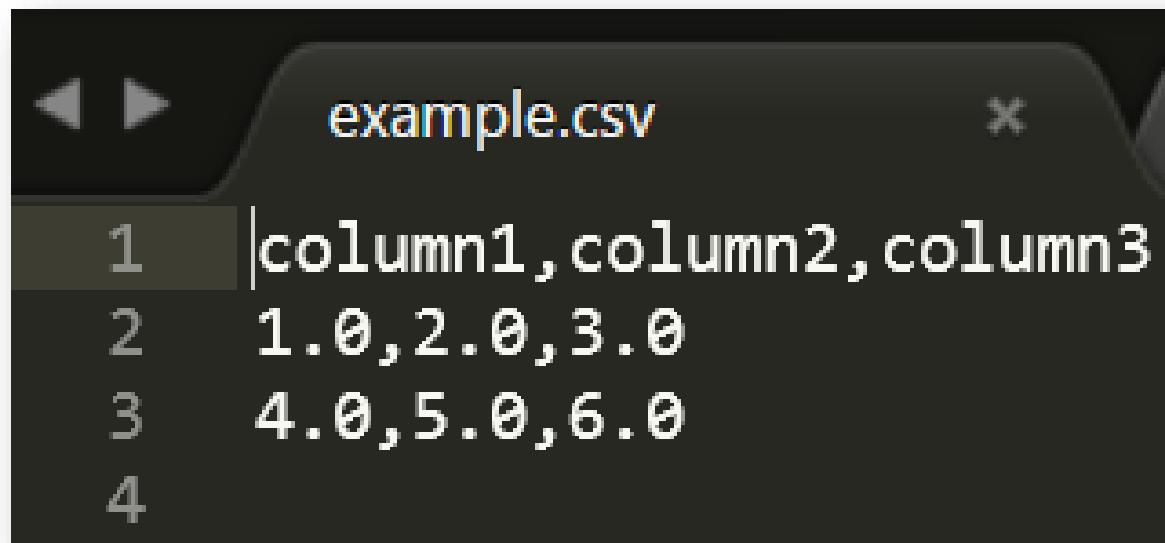
```
In [44]: df
```

Out[44]:

	column1	column2	column3
0	1.0	2.0	3.0
1	4.0	5.0	6.0

常用的資料格式 - CSV

□ Comma-Separated Values，是一種常見的資料格式，本身是一個純文字檔，用來記錄欄位的資料，欄位之間的資料常用逗號作分隔



將 DataFrame 存成 CSV

```
# 將剛剛建立好的 DataFrame 存成 csv  
df.to_csv("filename.csv", index=False,  
encoding="cp950")
```

- 不須額外增加 index 這個欄位
- windows 系統，encoding 使用 cp950 (有中文的話用 excel 打開才不會亂碼)，若為 linux 則用 utf-8

	A	B	C
1	column1	column2	column3
2		1	2
3		4	5

範例 05: 將高鐵時刻表的結果存成 CSV

```
# 將要查詢的資料寫成 dictionary
form_data = {
    "StartStation": "2f940836-cedc-41ef-8e28-c2336ac8fe68",
    "EndStation": "9c5ac6ca-ec89-48f8-aab0-41b738cb1814",
    "SearchDate": "2017/08/14",
    "SearchTime": "21:30",
    "SearchWay": "DepartureInMandarin"}

# requests 改用 POST，並放入 form_data
response_post = requests.post("https://www.thsrc.com.tw/tw/TimeTable/SearchResult", data = form_data)
soup_post = BeautifulSoup(response_post.text, "lxml")
```

範例 05: 將高鐵時刻表的結果存成 CSV - by columns

```
# 找出所有 td 標籤 屬性 class=column1 的內容，並存成 list
train_number = [tag.text for tag in
soup_post.find_all("td", class_="column1")]

# 找出所有 td 標籤 屬性 class=column3 的內容，並存成 list
departure = [tag.text for tag in
soup_post.find_all("td", class_="column3")]

# 找出所有 td 標籤 屬性 class=column4 的內容，並存成 list
arrival = [tag.text for tag in
soup_post.find_all("td", class_="column4")]

# 找出所有 td 標籤 屬性 class=column2 的內容，並存成 list
travel_time = [tag.text for tag in
soup_post.find_all("td", class_="column2")]
```

範例 05: 將高鐵時刻表的結果存成 CSV - by columns

```
# 建立 DataFrame，把 dictionary 放入，並指定 columns 順序
highway_df = pd.DataFrame(
{"車次":train_number,
 "出發時間":departure,
 "抵達時間":arrival,
 "行車時間":travel_time},
columns = ["車次", "出發時間", "抵達時間", "行車時間"])
```

```
In [24]: 1 highway_df
```

```
Out[24]:
```

	車次	出發時間	抵達時間	行車時間
0	0693	21:30	23:27	01:57
1	0333	21:45	23:41	01:56
2	0295	22:05	23:47	01:42

範例 05: 將高鐵時刻表的結果存成 CSV - by rows

```
# 先建立 DataFrame · 確定 columns
highway_df = pd.DataFrame(columns = ["車次", "出發時間",
                                       "抵達時間", "行車時間"])

# loop 3 次 · 每次取出一個 row 的 3 個值並存入 DataFrame
for i in range(3):
    row = soup_post.find_all("table", class_=
                           "touch_table")[i]

    row_contents = [tag.text for tag in row.find_all("td",
                                                       class_= re.compile("column"))]

    highway_df.loc[i] = row_contents
```

範例 05: 將高鐵時刻表的結果存成 CSV

```
# 把建立好的 DataFrame 存成 CSV
```

```
# for windows
```

```
highway_df.to_csv("data/demo05_highway_schedule.csv", index = False, encoding = "cp950")
```

```
# for linux
```

```
highway_df.to_csv("data/demo05_highway_schedule.csv", index = False, encoding = "utf-8")
```

	A	B	C	D
1	車次	出發時間	抵達時間	行車時間
2	693	21:30	23:27	01:57
3	333	21:45	23:41	01:56
4	295	22:05	23:47	01:42

練習 05: 將抓下來的資訊儲存成表格 (8 mins)

- 請觀察 [518 黃頁網](#)，並將店名、地址及電話三個欄位抓下來，並存成表格如下

店家	地址	電話
藍柚小廚	新北市/永和區...	02-2924...
果蔗新鮮	桃園縣/龍潭鄉...	03-470...
...

- 觀察店名、地址及電話都藏在哪些標籤底下？有共通的屬性嗎？
- 選擇要用 rows 或 columns 來組成 DataFrame

練習 05: 答案

```
# 萬年起手式
response = requests.get("http://yp.518.com.tw/service-life.html?ctf=10")
soup = BeautifulSoup(response.text, "lxml")

# 店家名稱與電話存在同一標籤，地址則在另一標籤
name_phone = [tag.text for tag in soup.find_all("li",
class_="comp_tel")]
address = [tag.text for tag in soup.find_all("li",
class_="comp_loca")]

# 運用 re 找到所有的電話號碼，運用 split，抓到店家名稱
name_phone_str = "".join(name_phone)
phone = re.findall("[0-9]{2}-[0-9]+", name_phone_str)
name = [x.split("/")[0].strip() for x in name_phone]
```

練習 05: 答案

```
# 將剛剛處理完的 list，以 dictionary 放進 pd.DataFrame 中
```

```
df = pd.DataFrame({"店名":name,  
                   "地址": address,  
                   "電話": phone},columns = ["店名","地址","電話"])  
df.to_csv("csv_results/practice05.csv", index =False,  
encoding="cp950")
```

	店名	地址	電話
0	藍柚小廚	新北市 / 永和區 永亨路49號	02-29242789
1	果蔗新鮮	桃園縣 / 龍潭鄉 龍元路33號	03-4709933
2	沁采美食館	台中市 / 北區 進德北路38號	04-23601719
3	御私藏鮮奶茶專賣	臺南市 / 東區 育樂街17號	06-2092929
4	天沅茶舖	嘉義市 / 西區 仁愛路449號	05-2238686

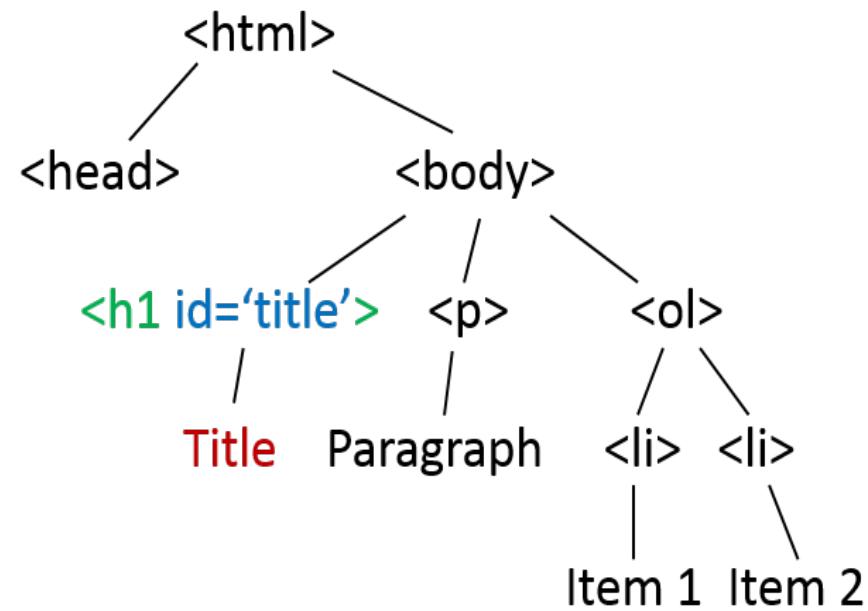
小結

□ HTML

- 元素所組成的階層式文件
- 標籤中含有屬性 (attributes) 以及內容

<標籤 屬性> 內容 </標籤>

```
<html>
  <head>
  </head>
  <body>
    <h1 id='title'> Title </h1>
    <p> Paragraph </p>
    <ol>
      <li> Item 1 </li>
      <li> Item 2 </li>
    </ol>
  </body>
</html>
```





小結

- requests
 - GET, POST
- BeautifulSoup
 - find_all(tags, attributes)
- regular expression
 - find_all(tags, {attributes: "your_re"})



爬蟲實戰練習

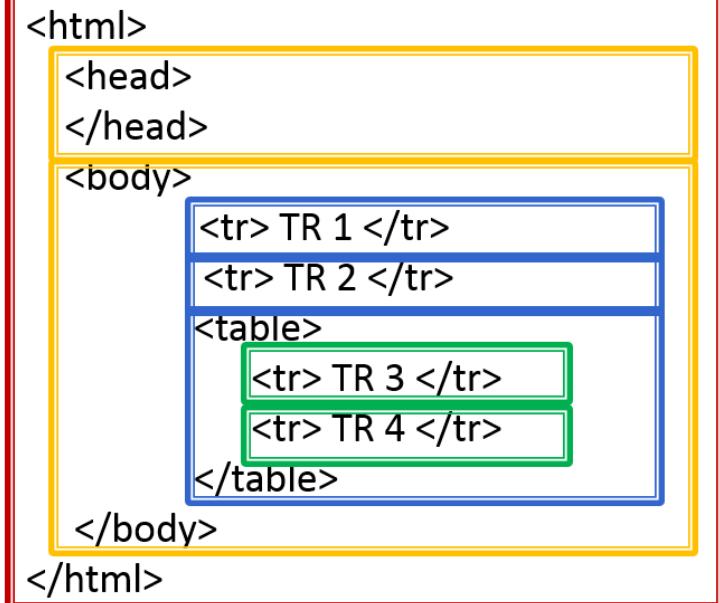
爬蟲實戰攻略 - 掌握階層式的架構

- 当你使用 `find_all()` 時，會抓出所有符合條件的標籤，但當你先找到一個特定標籤，在使用 `find_all()` 時，找到的就是在那特定標籤底下所有符合條件的標籤。

```
# 找出所有 tr 標籤
soup.find_all("tr")
```

```
# 找到 table 標籤底下的所有 tr 標籤
soup.find("table").find_all("tr")
```

```
# Warning! 不可以這樣寫! Why?
soup.find_all("table").find("tr")
```



爬蟲實戰小技巧

□ 找到目標資訊藏在哪些標籤底下？

- `stripped_strings`: 找出 tag 底下所有的文字，且幫你去除所有空格、換行符號等等，需要用 `iterate` 的方式取值。

```
In [24]: example
```

```
Out[24]: <tr class="tb-top">
    <th>週次</th>
    <th>日期</th>
    <th>週末票房總和</th>
    <th>漲跌幅</th>
    <th>冠軍片名</th>
    <th>英文片名</th>
    <th class="import">週末票房冠軍</th>
    <th>冠軍比例*</th>
</tr>
```

```
In [21]: [text for text in example.stripped_strings]
```

```
Out[21]: ['週次', '日期', '週末票房總和', '漲跌幅', '冠軍片名', '英文片名', '週末票房冠軍', '冠軍比例*']
```

爬蟲實戰小技巧

□ 如何找出沒有任何屬性的標籤?

```
In [195]: print(soup.find_all("li"))
```

```
[<li class="zz"> my attributes class is zz </li>, <li> new cell 2 </li>]
```

```
In [196]: print(soup.find_all("li", {"class":None}))
```

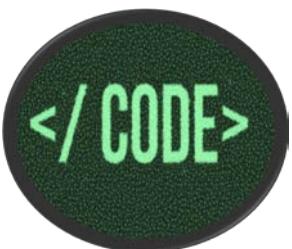
```
[<li> new cell 2 </li>]
```

□ 如何對 list 做 loop，同時增加 index 值?

```
In [1200]: 1 test_list = [4, 5, 6]
2 for i, num in enumerate(test_list):
3     print(i, num)
```

```
0 4
1 5
2 6
```

爬蟲實戰練習



- 請運用所學過的方法，嘗試爬取以下這兩個網頁的資訊，並儲存成 CSV
 - 台北票房觀測站-年度排名，請爬取 2016、2017 年度排名 (15 ~ 20 mins)
 - 如果卡住的話，歡迎參考範例 o6 的 code
 - yahoo 奇摩電影評論，請選擇在票房排行榜上任何一部您喜歡的電影，並將其所有的評論文字、評論星等以及該電影的名稱抓下來 (20 ~ 25 mins)
 - 如果卡住的話，歡迎參考練習 o6-2 的提示



終於把資料都爬下來了

- 恭喜你成功養了一隻會實際爬網頁的爬蟲



- 隨著不同網頁的變化，你的爬蟲要學會變的更強大堅韌，才能成功完成任務



Outline

B. 爬蟲實戰與資料分析

- BeautifulSoup + regular expression
- GET vs. POST
- 使用 pandas 儲存資料
- 文字探勘與資料分析

情境

- 辛苦把資料都抓下來，也把成功資料儲存成結構化的數據，慣老闆突然神來一筆，最近大數據還有甚麼 AI 不是非常紅嗎？為什麼不用這些資料來做一些大數據分析阿！



- 已經受不了在準備辭呈的你突然想起，Python 爬蟲實戰好像也有玩過一點資料分析耶!



Data Summary

- 電影票房資料 - 排名、片名、院數、映期、上映日期、平均票房、累積票房
 - 2016, 2017 年各 100 部

排名	中文片名	英文片名	院數	映期	上映日期	平均票房	累積票房	年度
1	美國隊長3：英雄內戰	Captain America 3	24	86	2016/04/27	7,652,852	183,668,450	2016
2	惡棍英雄：死侍	Deadpool	23	80	2016/02/09	6,200,975	142,622,425	2016
3	屍速列車	Train to Busan	22	78	2016/09/02	6,047,077	133,035,683	2016

- 電影評論資料 - 片名、評論文字、評論星等
 - 200 部電影的所有評論及星等

comments	movie	star
爽片，劇情符合邏輯但不合常理，爽就好~傑森在這片淪為小配角了~ 玩命關頭8\n Fast & Furious 8\n		5
爛戲一部,\n就只有靠傑森史塔森的橋段加分 玩命關頭8\n Fast & Furious 8\n		2
我覺得這集還不錯呀，最後一刻最感人 玩命關頭8\n Fast & Furious 8\n		5



資料清理

- 在資料分析開始前，資料清理永遠都是最重要的工作，資料的乾淨與否會影響分析的結果
 - 檢查是不是抓下來的資訊是否正確？

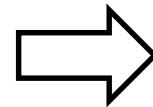
movie	star
美國隊長3：英雄內戰\n Captain America: Civil War\n	1
美國隊長3：英雄內戰\n Captain America: Civil War\n	3
美國隊長3：英雄內戰\n Captain America: Civil War\n	5

- 格式是不是可以使用？

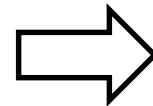
平均票房	累積票房	年度
7,652,852	183,668,450	2016
6,200,975	142,622,425	2016
6,047,077	133,035,683	2016
4,942,212	118,613,078	2016
5,049,736	116,143,931	2016

可以玩甚麼分析呢？

200 部電影的評論文字
且有評論星等



200 部電影的票房資料
有評論跟星等



文字探勘與文字雲

- 探討評論文字與星等之間的關係
 - 級五顆星的人都寫了甚麼？
 - 級一顆星的人又都寫了甚麼？
 - 文字雲可以美觀地呈現出文字的重要程度



中文字是需要斷詞的

□ 甚麼是斷詞？

- 例如：我覺得不行
- 我 / 覺得 / 不行 ○
- 我覺 / 得不行 X

我覺得不行



□ 英文本身就已經用空格做好斷詞這件事，但是中文真的不行

- 乖乖的做斷詞吧！



jieba 登場!

```
# 載入套件
import jieba

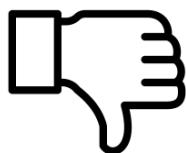
# 把"我覺得可以"作斷詞後
print([x for x in jieba.cut("我覺得不行")])

['我', '覺得', '不行']
```

- 把每一條評論斷完詞後，我們就可以得到每條評論都用了哪些詞 (e.g., 好看、爛片、無聊)
- 再運用統計的方式，計算一下五星評論與一星評論之間的用詞頻率是否有所不同

文字雲的應用

- 經過斷詞後，我們得到了詞彙在評論中出現的頻率，透過 WordCloud 這個套件可以呈現出這樣的結果



一顆星評論



五顆星評論



TF-IDF

- 直接統計頻率好像太 Low 了，有沒有潮一點的方法？

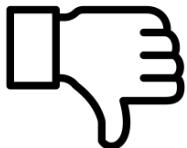


□ Term Frequency - Inverse Document Frequency

- TF: 詞頻，該詞在某一文件中出現的次數
- IDF: 逆向文件頻率，該詞在所有文件中出現的次數

TF-IDF 的文字雲

- 透過 TF-IDF 的分析過後，較有意義的詞會更容易得到更高的權重



一顆星評論



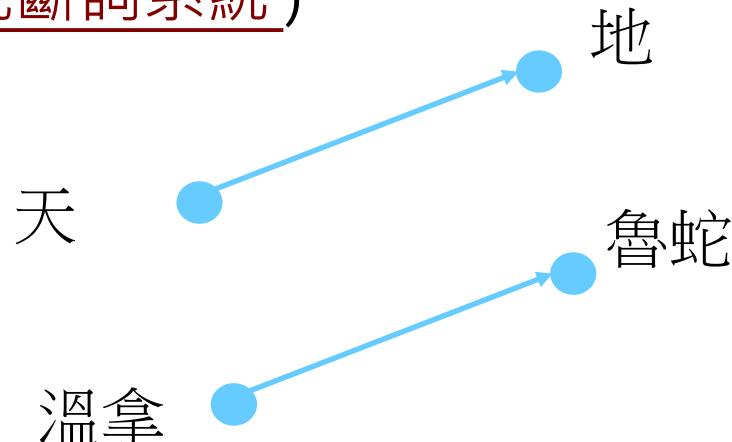
五顆星評論



文字探勘

只要有 Data，不怕沒得玩

- 換套斷詞方式 ([中研院斷詞系統](#))
- word2vec (詞向量)



工商服務時間

- 09/03(日) [無所不在的自然語言處理—基礎概念、技術與工具介紹](#)
- 本課程為自然語言處理的基礎課程，具程式背景，有撰寫爬蟲經驗者較能進入狀況

票房資料分析

叫好又叫座



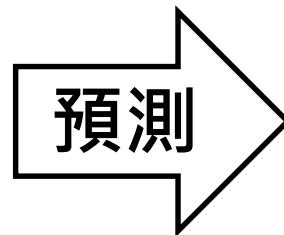
&



- 叫好跟叫座有關係嗎?
- 讓我們用資料找真相



用平均星等來預測票房



- 將電影所有的評論星等作平均，即可獲得該電影的平均評價
- 透過 Linear Regression，來探討平均星等與票房之間的關係



Linear Regression on Scikit-Learn

```
# 切 training data 跟 testing data
X_train,X_test,y_train,y_test = train_test_split(
movie_box["平均星等"], np.log10(movie_box["平均票房"]))

# 建立 Regression 模型
reg = LinearRegression()

# 放入 training data 進行訓練
reg.fit(X_train, y_train)

# 用 testing data 做預測
y_pred = reg.predict(X_test)

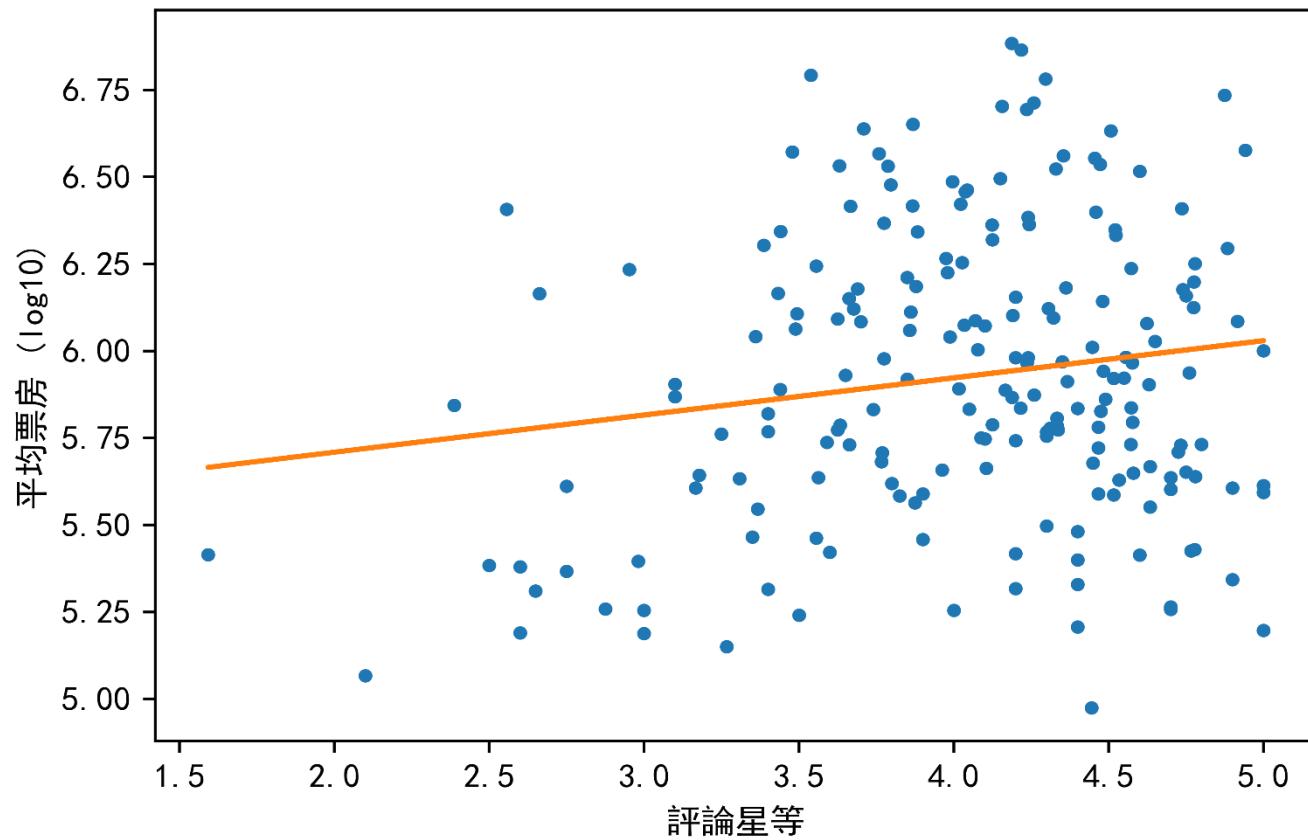
# 算出 testing data 與預測結果的相關係數
print(pearsonr(y_pred, y_test))
```

0.21



為什麼這麼差? (1/2)

- 評論星等跟票房沒有很強烈的線性關係



為什麼這麼差? (2/2)

□ Features 用的太少

- 還有很多變數可以解釋票房的變異



□ 回歸太難了

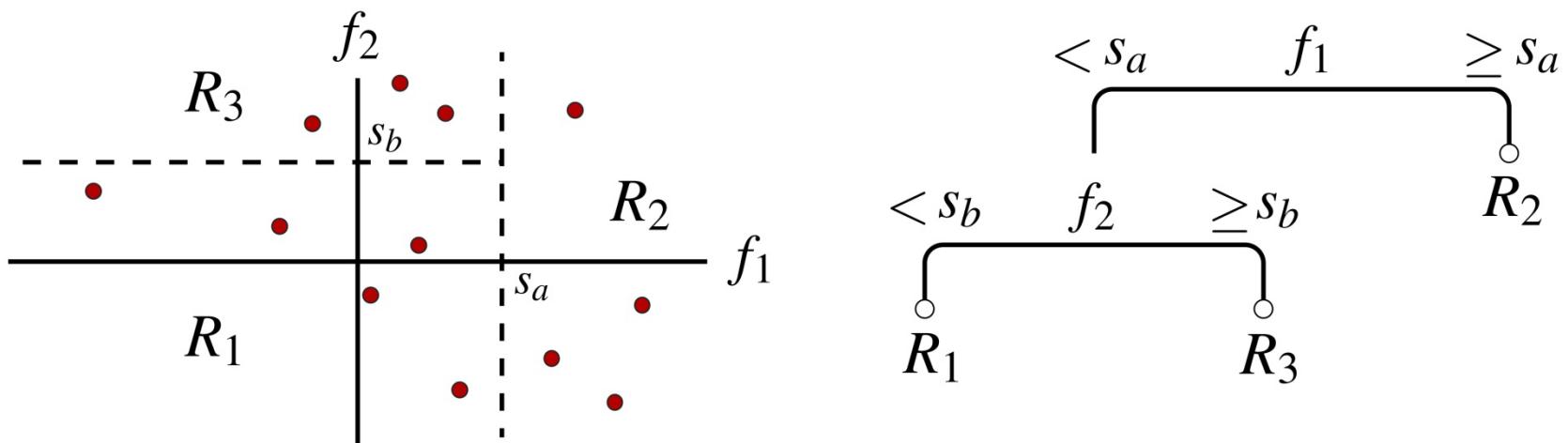
- 試試用分類吧!

不叫座 < 票房中位數 < 叫座



分類問題的好幫手 - 決策樹 (Decision Tree)

- 在空間中找到一些線 (rule) 讓這些點可以被分開
- 每個分支就好像在做決策 (Decision)



Decision Tree on Scikit-Learn

```
# 切 training data 跟 testing data
X_train,X_test,y_train,y_test = train_test_split(
movie_box[["評論數量", "平均星等", "映期"]], movie_box["平均
票房"])

# 建立 Decision Tree 模型
clf = DecisionTreeClassifier(max_depth=3)

# 放入 training data 進行訓練
clf.fit(X_train, y_train)

# 用 testing data 做預測
y_pred = reg.predict(X_test)

# 算出 testing data 與預測結果的準確率
print(accuracy_score(y_pred, y_test))
```

0.84

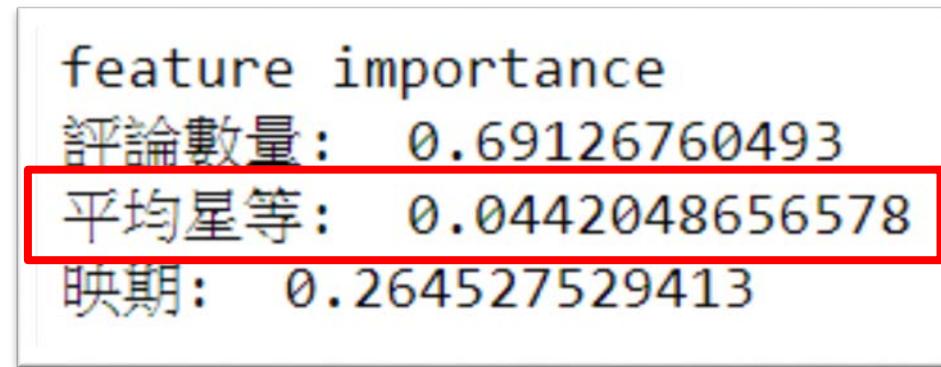


生成的決策樹

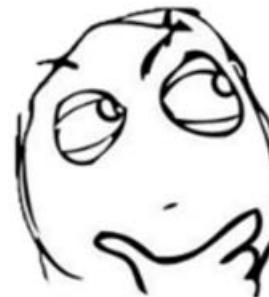


小結

- 透過 Decision Tree 的 features importance，我們可以發現，平均星等根本就不太重要嘛！



- 你還能想出其他的 features 嗎？





上午我們學到了...



運用 BeautifulSoup 解析 HTML 網頁

```
requests.get("website"), requests.post("website", form_data)
```



運用 requests 發送 GET, POST 請求

```
soup = BeautifulSoup(response.text), soup.find_all()
```



運用 regular expression 尋找目標資訊

```
re.findall(pattern, test_string)
```



運用 pandas 將抓到的資訊儲存為表格

```
DataFrame.to_csv(file_path)
```



運用 sklearn, matplotlib, wordcloud 做簡單的資料分析



下午你會學到更多!

- 靜態網頁以外的爬蟲
 - 圖片爬蟲
 - 檔案爬蟲
 - 網站爬蟲
- 現實世界的爬蟲
 - 現代網站爬蟲衍生的問題
 - 動態網頁爬蟲

Outline

C. 靜態網頁以外的爬蟲

- 圖片爬蟲
- 檔案爬蟲
- 網站爬蟲

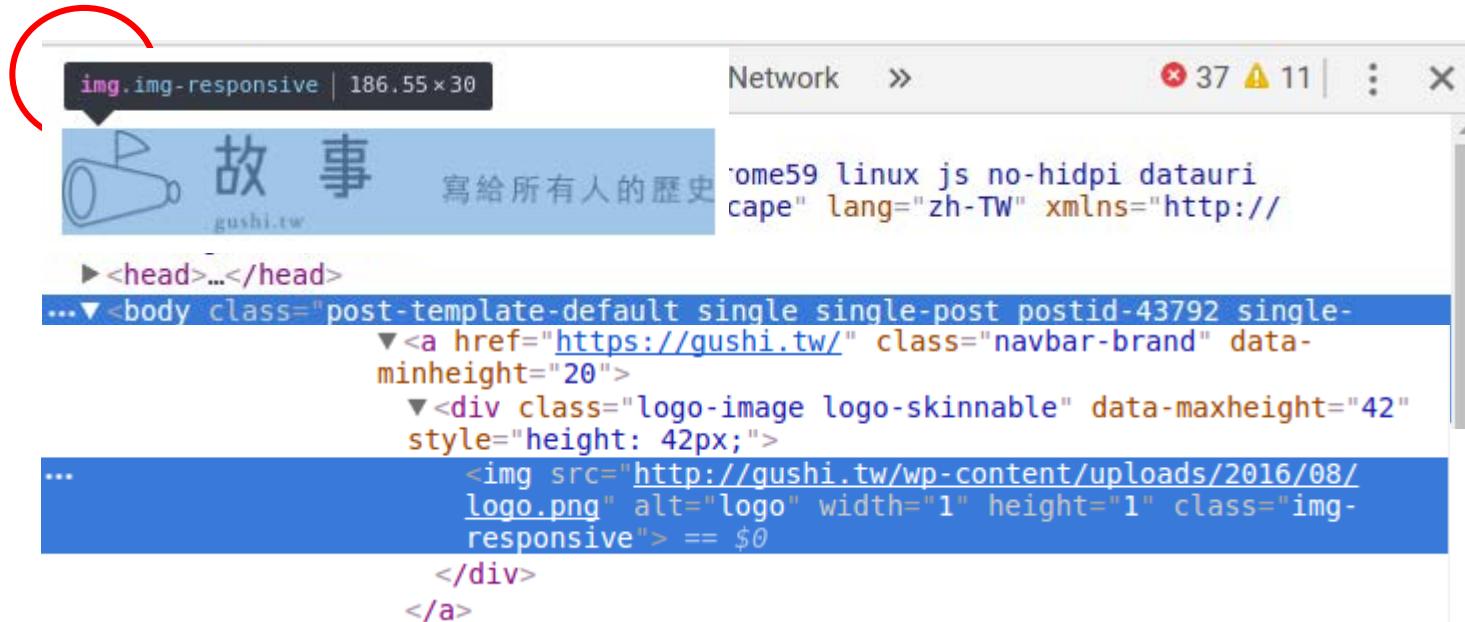
D. 現實世界的爬蟲

- 現代網站爬蟲衍生的問題
- 動態網頁爬蟲



圖片爬蟲 - 圖片的 tag ?

- 打開瀏覽器的開發者工具
- 選擇 inspect 工具
- 滑鼠移到你想選擇的圖片



圖片爬蟲 - 與文字爬蟲的差異

□ 文字爬蟲

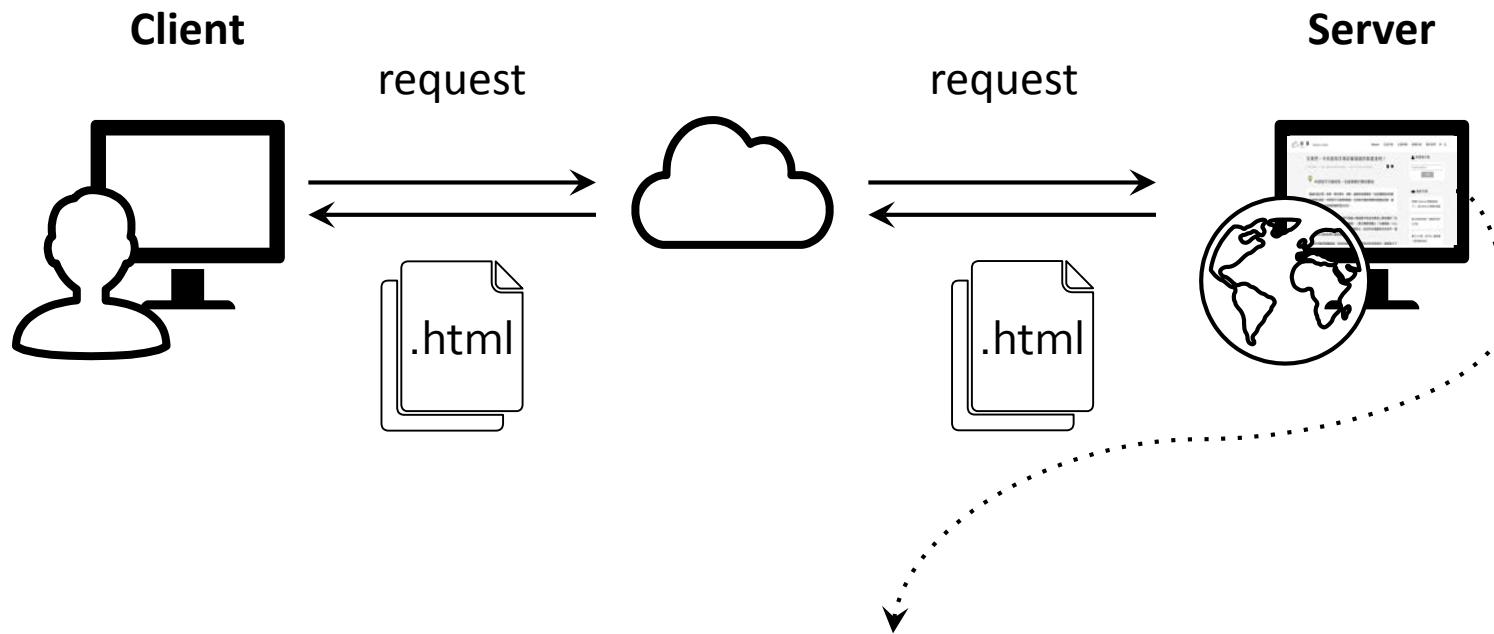
- 送 request 到目標網頁
- 透過 BeautifulSoup 取得目標文字 tag
- 透過 .text 可以直接拿到文字資訊

□ 圖片爬蟲透過解析網頁只能拿到圖片位置資訊

- 送 request 到目標網頁
- 透過 BeautifulSoup 取得目標圖片 tag
- 透過 src 屬性拿到圖片位置
- 再送 request 到圖片真正的位置
- 取得圖片並儲存

爬文字的過程 - 取得頁面資訊

□ 爬文字的過程

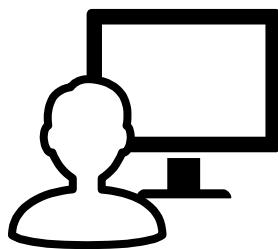


<https://gushi.tw/hu-shih-memorial-hall/>

爬文字的過程 - 解析頁面並取得文字

□ 爬文字的過程

Client



Beautifulsoup()



尋找第 4 個 < p > tag



透過 .text 取得文字資訊

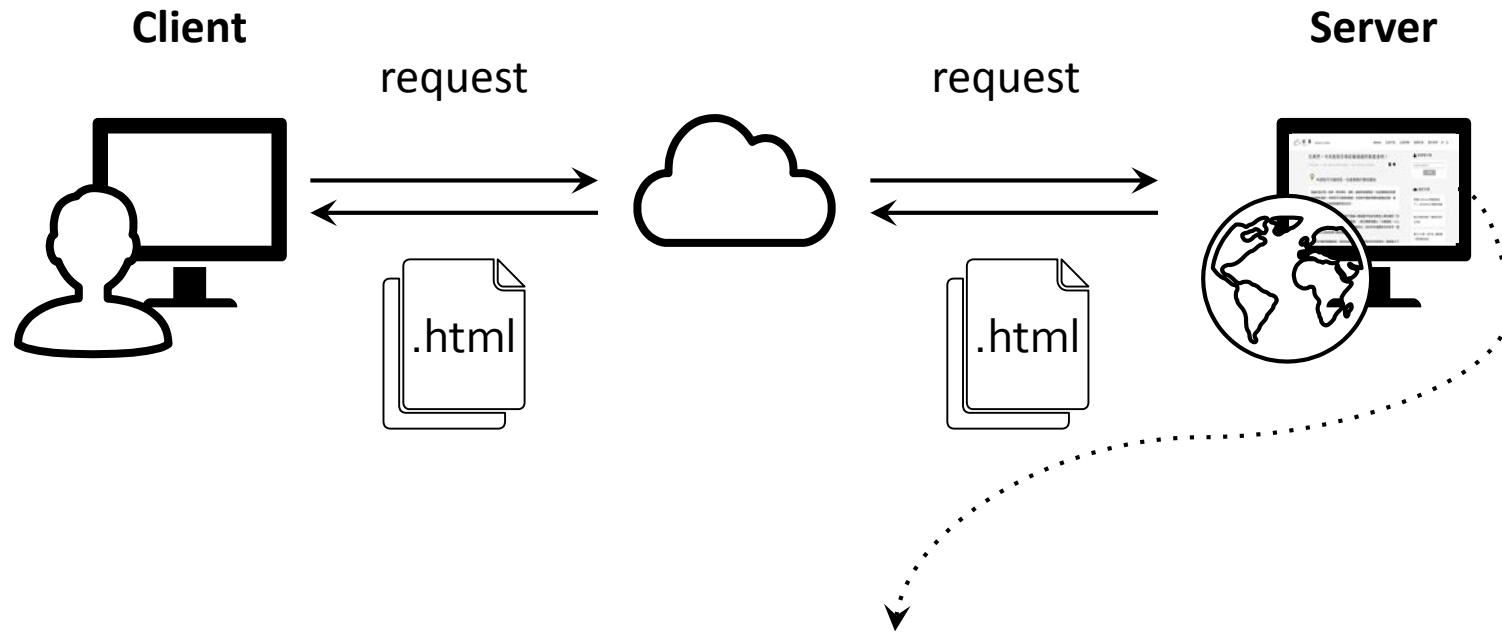
```
In [13]: text = soup.find_all('p')  
print(text[3].text)
```

這樣資深的硬派文青，你難道不好奇他的書房長怎樣？用什麼筆寫作？又有什麼寫作習慣？答案都在胡適紀念館等你發掘！

</p>



爬圖片的過程 - 取得頁面資訊



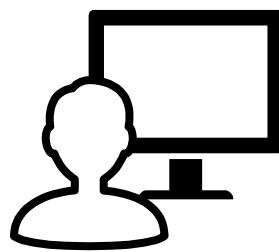
<https://gushi.tw/hu-shih-memorial-hall/>



爬圖片的過程 - 取得位置資訊

□ 爬圖片的過程

Client



Beautifulsoup()

尋找第 1 個 tag

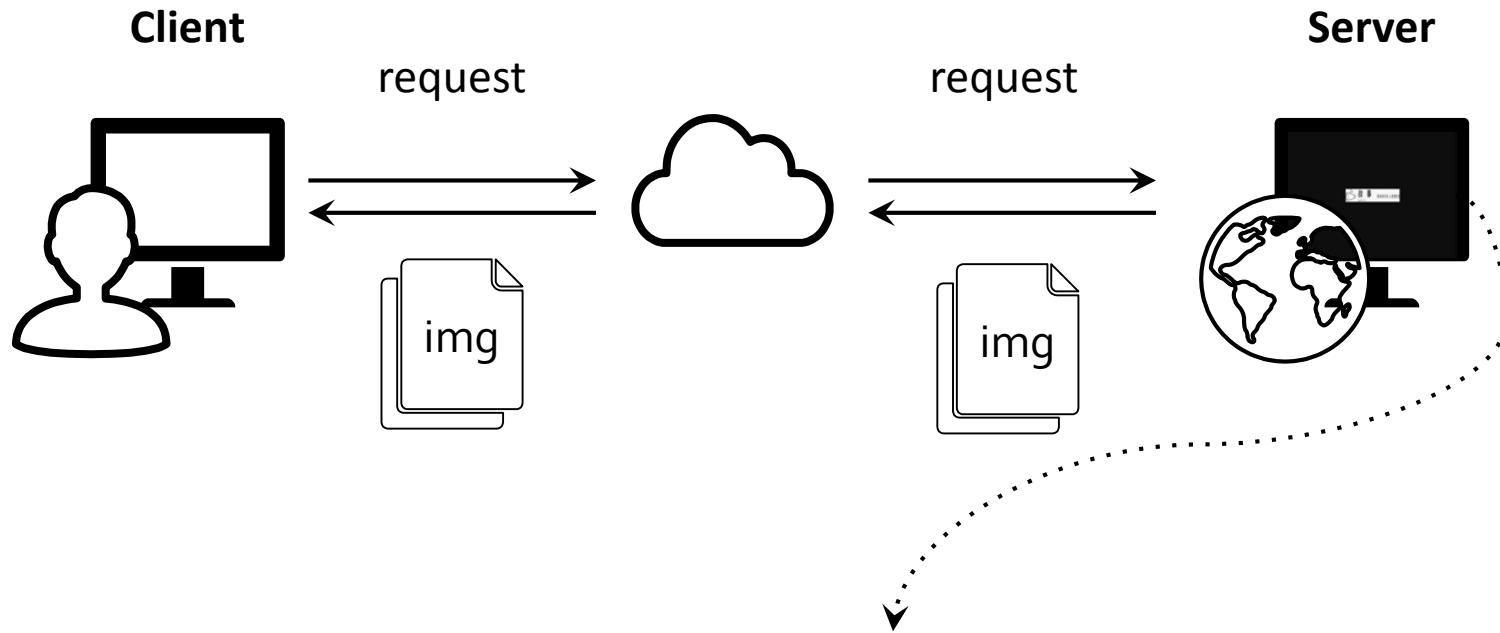
透過 src 取得圖片位置資訊

```
In [16]: image = soup.find_all('img')
print(image[0]['src'])
```

```
http://gushi.tw/wp-content/uploads/2016/08/logo.png
wp-content/uploads/2016/08/logo.png 从 url - 1 /
```

爬圖片的過程 - 取得圖片

□ 爬圖片的過程



<http://gushi.tw/wp-content/uploads/2016/08/logo.png>

爬圖片的過程 - 下載圖片

```
from urllib.request import urlretrieve
```

```
# 透過 urlretrieve 下載圖片  
# url: 你要下載的圖片位置  
# file: 你要儲存的文件名稱  
urlretrieve(url, file)
```

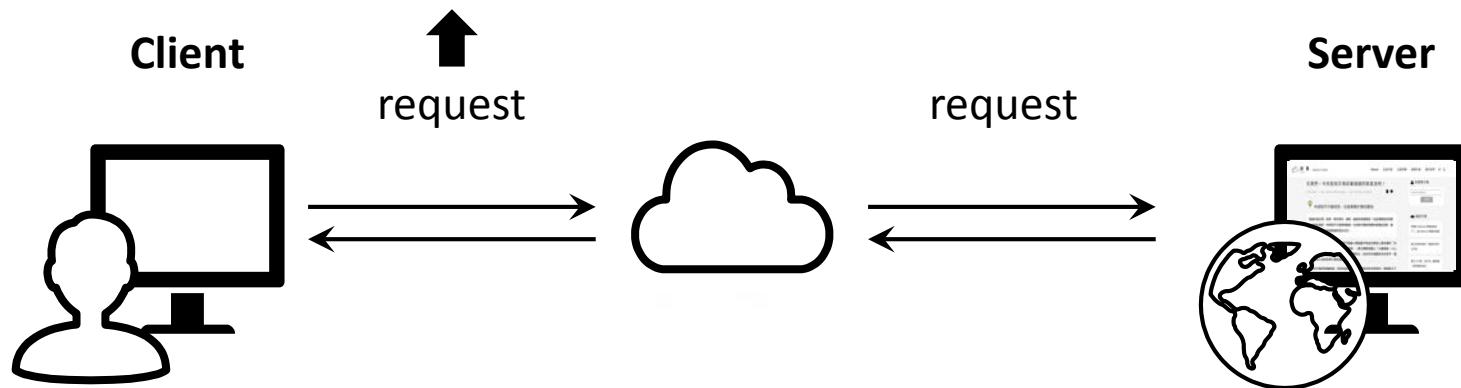
```
/usr/lib/python3.5/urllib/request.py in http_error_default(self, req, fp, code, msg, hdrs)  
 588 class HTTPDefaultErrorHandler(BaseHandler):  
 589     def http_error_default(self, req, fp, code, msg, hdrs):  
--> 590         raise HTTPError(req.full_url, code, msg, hdrs, fp)  
 591  
 592 class HTTPRedirectHandler(BaseHandler):
```

HTTPError: HTTP Error 403: Forbidden

爬圖片的過程 - 偽裝成瀏覽器發送請求

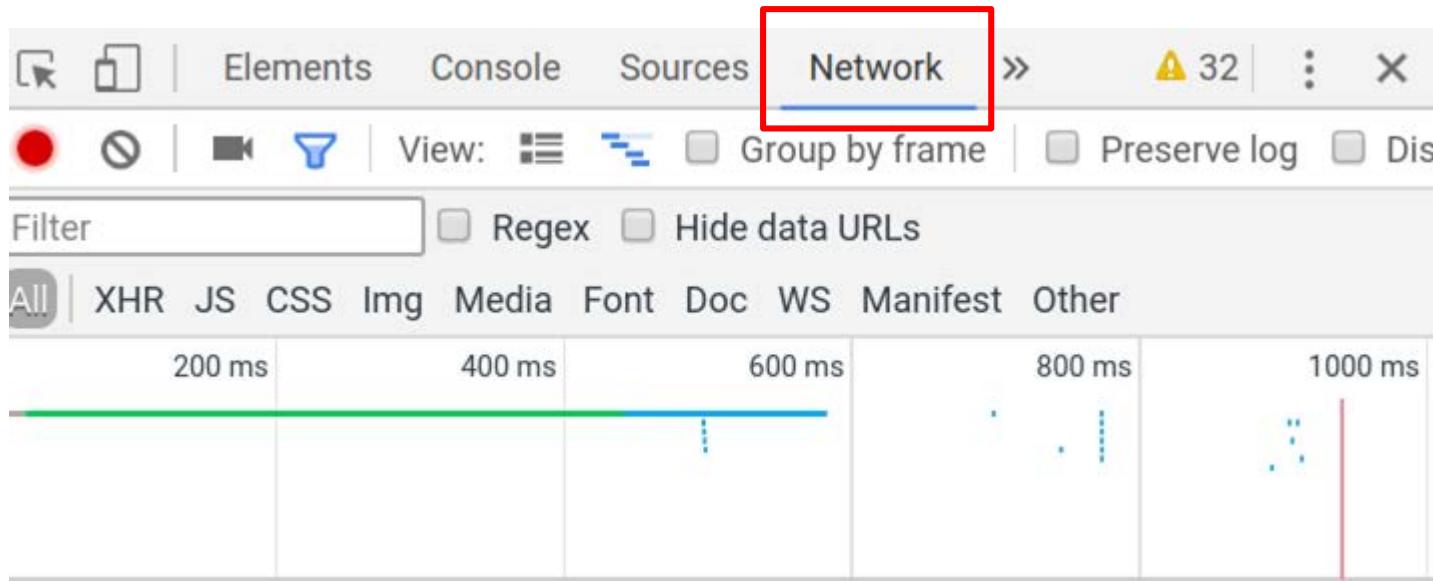
- 部份網站會判斷你是否為爬蟲程式
- 加上身份識別偽裝成瀏覽器送出請求

```
{'Cache-Control': 'max-age=0, no-cache, no-store, must-revalidate, public',
'Connection': 'keep-alive',
'Content-Encoding': 'gzip',
'Content-Type': 'text/html; charset=UTF-8',
'Date': 'Wed, 02 Aug 2017 17:19:34 GMT',
'Expires': 'Mon, 29 Oct 1923 20:30:00 GMT',
'Last-Modified': 'Tue, 01 Aug 2017 11:55:50 GMT',
'Pragma': 'public',
'Server': 'cloudflare-nginx',
'Transfer-Encoding': 'chunked',
'Vary': 'Accept-Encoding,User-Agent',
'X-Powered-By': 'W3 Total Cache/0.9.4.1'}
```



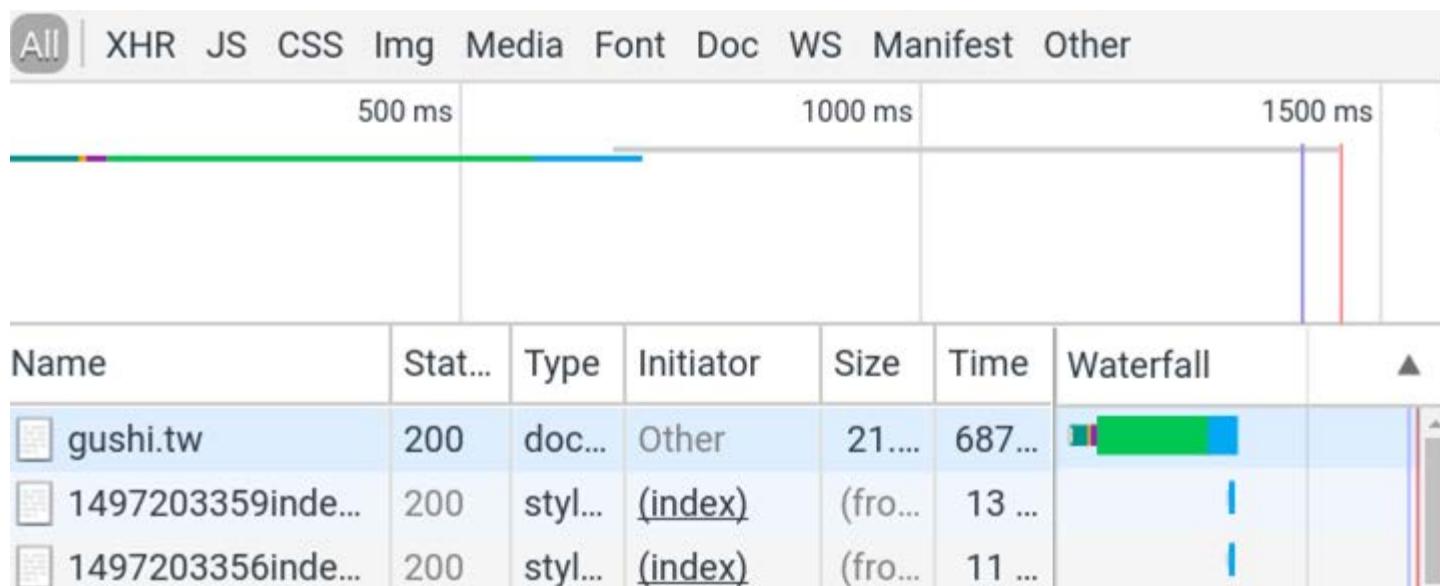
身份識別 User-Agent

- 敘述瀏覽器使用的系統, 平台, 版本等資訊的字串
- 瀏覽器開發者工具 > Network



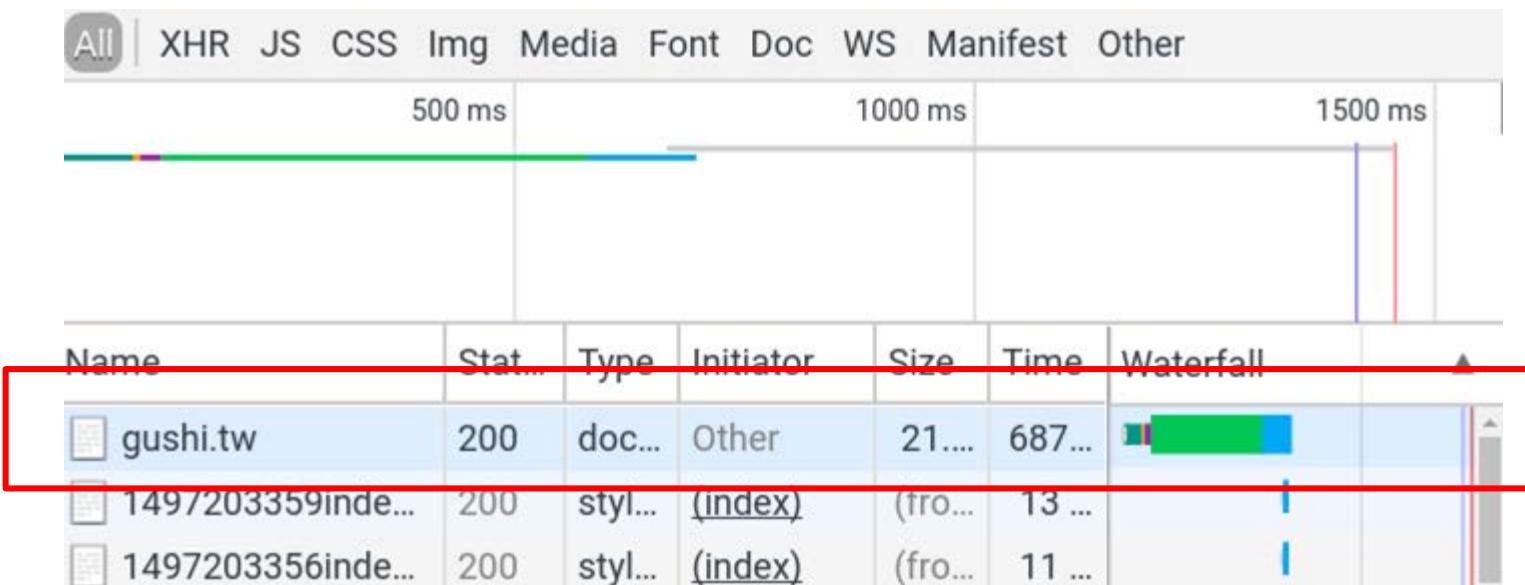
身份識別 User-Agent

- 敘述瀏覽器使用的系統, 平台, 版本等資訊的字串
- 瀏覽器開發者工具 > Network > 重新整理網頁



身份識別 User-Agent

- 重新整理網頁之後選擇對網頁送出的 request



身份識別 User-Agent

□ 檢查 Headers 欄位中的 Request Headers

The screenshot shows the Network tab of a browser developer tools interface. A red box highlights the 'Headers' tab in the top navigation bar. Another red box highlights the 'Request Headers' section under the 'Response Headers' heading. The 'Request Headers' list includes:

- :authority: gushi.tw
- :method: GET
- :path: /
- :scheme: https
- accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
- accept-encoding: gzip, deflate, br
- accept-language: zh-TW,zh;q=0.8,en-US;q=0.6,en;q=0.4
- cache-control: max-age=0
- cookie: [REDACTED]
- upgrade-insecure-requests: 1
- user-agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.78 Safari/537.36

爬圖片的過程 - 偽裝成瀏覽器發送請求

- 部份網站會判斷你是否為爬蟲程式
- 假裝成瀏覽器發送請求預防被檔

```
from urllib.request import build_opener  
from urllib.request import install_opener  
opener = build_opener()  
opener.addheaders = [('User-Agent', 'Mozilla/5.0  
(X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like  
Gecko) Chrome/60.0.3112.78 Safari/537.36')]  
install_opener(opener)
```

練習 oo: 下載圖片 (5 ~ 8 mins)

目標程式 : oo_download_image.py

目標網站 : <https://gushi.tw/hu-shih-memorial-hall/>

目標 : 下載該頁面的第一個圖片

- 取得圖片的位置
- 透過開發者工具找到 User-Agent 的值
- 假裝成瀏覽器送出請求
- 下載圖片

Bonus: 檢查圖片下載百分比

- 透過 urlretrieve 的 callback 實作
- callback 代表做完功能 A 時要做的功能 B

```
from urllib.request import urlretrieve
# 設計 callback
def check_percentage(chunk, size, remote):
    percent = 100.0 * chunk * size / remote
    if percent > 100.0:
        percent = 100.0
    print('Download...{:.2f}%'.format(percent))
urlretrieve(url, 'logo.png', check_percentage)
```

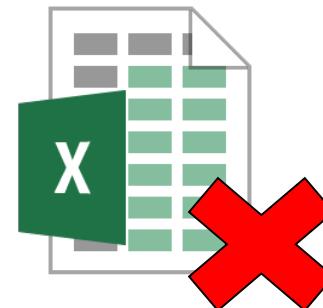
圖片存檔 - 副檔名的重要性

- 副檔名只是讓電腦知道要用甚麼方式讀取的提示
- 副檔名錯誤就無法正確開啟檔案
- 直接更改副檔名並不等於轉檔



test.docx

rename



test.xlsx



用 Office Word
開啟的文件

用 Office Excel
開啟的文件



站外資源

批踢踢實業坊 > 看板 Beauty

聯絡資訊 關於我們

作者 fantacy5566 (fantacy5566)
標題 [帥哥] 說我帥太沈重
時間 Sat Dec 17 18:15:57 2016

看板 Beauty

<http://i.imgur.com/q6fMyz9.jpg>



眼見不為憑

□ <http://i.imgur.com/q6fMyz9.jpg>



眼見不為憑

□ <http://i.imgur.com/q6fMyz9.png>



眼見不為憑

□ <http://i.imgur.com/q6fMyz9.gif>



GQ



獲取真實的圖片格式

```
import requests  
from bs4 import BeautifulSoup  
from PIL import Image  
  
url = 'http://i.imgur.com/q6fMyz9.jpg'  
response = requests.get(url, stream=True)  
  
# 讓 PIL.Image 讀進圖片幫我們了解圖片格式  
image = Image.open(response.raw)  
print(image.format) # JPEG
```

圖片存檔 - 原始檔名與正確的副檔名

- 透過圖片 URL 提取檔名
- 透過檢查過後的圖片格式當作副檔名

```
# url = 'http://i.imgur.com/q6fMyz9.jpg'
filename = url.split('/')[-1] # q6fMyz9.jpg
filename = filename.split('.')[0] # q6fMyz9
ext = image.format.lower() # JPEG -> jpeg
download_filename = '{}.{}'.format(filename,
ext) # q6fMyz9.jpeg
```

練習 01: 判斷格式並下載圖片 (5 ~ 8 mins)

目標程式：o1_download_image_and_check_format.py

目標：下載圖片並以正確格式儲存

- 透過 stream 的方式送出請求
- 透過 PIL.Image 檢查圖片格式
- 透過字串處理取得適合的檔案名稱
- 下載圖片

```
url = 'http://imgur.com/rqCqA.png'  
url = 'http://imgur.com/rqCqA.jpg'  
url = 'http://imgur.com/rqCqA.gif'
```

練習 01: 延伸思考

- 是否可以不透過 stream 的方式來判斷圖片？
 - 透過 BytesIO 將圖片轉成 PIL.Image 需要的格式
- 是否可以只傳一次 request 就做判斷格式與存檔
 - bonus_one_requests.py
 - 直接使用 PIL.Image 存檔而不透過 urlretrieve

Outline

- 靜態網頁以外的爬蟲
 - 圖片爬蟲
 - 檔案爬蟲
 - 網站爬蟲
- 現實世界的爬蟲
 - 現代網站爬蟲衍生的問題
 - 動態網頁爬蟲

檔案爬蟲 - 超連結檔案

- 透過開發者工具查看檔案的 tag

年度▼	系所組	科目名稱	試題檔案
106	中國文學研究所	文字學	
106	中國文學研究所	聲韻學	
106	中國文學研究所	中國語文能力測驗	
106	中國文學研究所	中國文學史	
106	中國文學研究所	中國思想史	

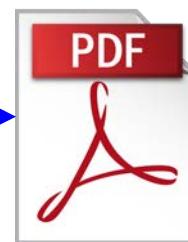
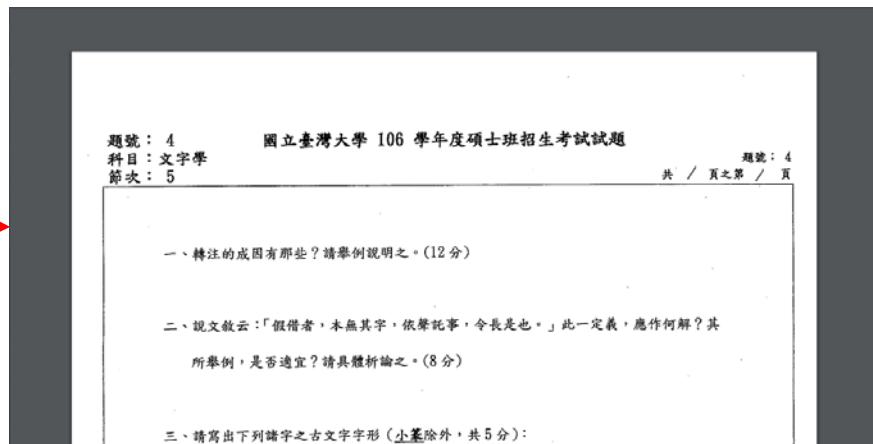
```
<a href="/exam/sites/all/modules/pubdlcnt/pubdlcnt.php?file=http://140.112.115.12/exam/sites/default/files/exam/graduate/106g/106_graduate_4.pdf&nid=5814">

</a>
```



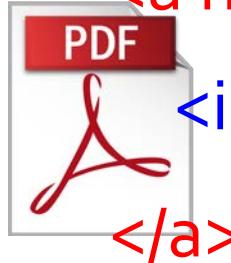
檔案爬蟲 - 超連結的本體

- 外表看似圖片，但其實本體是超連結



檔案爬蟲 - 定位節點

- 尋找所有 a tag 再用 regular expression 過濾 href
- 尋找所有裏面包含 img tag 的 a tag
- 尋找相同圖片而且上層是 a tag 的 img tag



```
<a href="...">
    src =
        
</a>
```

檔案爬蟲 - 定位節點

- 尋找相同圖片而且上層是 a tag 的 img tag

```
# 透過 regular expression 找到相同圖片的 img tag
images = soup.find_all('img',
{'src': re.compile('application-pdf\.png')})

for image in images:
    # 透過 parent 函數尋訪 img tag 的上一層 tag
    print(image.parent['href'])
```

href 的絕對路徑與相對路徑

- 透過給予檔案位置 (URL) 讓網頁可以參考並顯示
e.g. ,
- 敘述檔案位置的方式分為
 - 絕對路徑 (e.g.
<http://140.112.115.12/exam/sites/all/modules/filefield/icons/application-pdf.png>)
 - 相對路徑 (e.g.
/exam/sites/all/modules/pubdlcnt/pubdlcnt.php)

檔案爬蟲 - 相對位置

- URL 代表檔案在網路上的位置
- 無法直接對相對路徑送 requests



請問可以外送十杯咖啡嗎？



可以阿，請問要送到哪裡？



門口進來左轉的樓梯到三樓右手邊的櫃台



... 是奧客嗎



檔案爬蟲 - 相對位置

- URL 代表檔案在網路上的位置
- 無法直接對相對路徑送 requests

```
url =  
'/exam/sites/all/modules/pubdlcnt/pubdlcnt.php?file=http://140.  
112.115.12/exam/sites/default/files/exam/graduate/106g/106_ gr  
aduate_4.pdf&nid=5814'  
  
response = requests.get(image_url)
```

InvalidSchema: No connection adapters were found for '/exam/sites/all/m
odules/pubdlcnt/pubdlcnt.php?file=http://140.112.115.12/exam/sites/defa
ult/files/exam/graduate/106g/106_graduate_4.pdf&nid=5814'

檔案爬蟲 - 將相對路徑轉為絕對路徑

- 必須把相對路徑轉為絕對路徑才能送出 request
- URL 有一定的格式，很難單純做字串處理轉換
- 透過參考用的絕對路徑就可以轉換相對路徑

絕對路徑 <http://exam.lib.ntu.edu.tw/graduate>

相對路徑 /exam/sites/all/modules/pubdlcnt/pubdlcnt.php

組合

→ [http://exam.lib.ntu.edu.tw/exam/sites/all/modules
/pubdlcnt/pubdlcnt.php](http://exam.lib.ntu.edu.tw/exam/sites/all/modules/pubdlcnt/pubdlcnt.php)



檔案爬蟲 - 取得絕對位置

- 透過 urllib.parse.urljoin 取得絕對位置
- 當前網頁的 URL 最適合拿來做參考用的絕對路徑

```
from urllib.parse import urljoin  
  
print(urljoin(絕對路徑, 相對路徑))
```

```
print(urljoin(response.url, images[0].parent['href']))
```

```
http://140.112.115.12/exam/sites/all/modules/pubdlcnt/pubdlcnt.php?file  
=http://140.112.115.12/exam/sites/default/files/exam/graduate/106g/106_  
graduate_4.pdf&nid=5814
```

檔案爬蟲 - 解析 URL

- ❑ urljoin 其實是透過 urllib.parse.urlparse 將兩組 URL 拆解成數個片段再去組合出新的絕對路徑

絕對路徑

```
print(urlparse('http://exam.lib.ntu.edu.tw/graduate'))
```

```
ParseResult [scheme='http', netloc='exam.lib.ntu.edu.tw', path='/graduate', params='', query='', fragment='']
```

相對路徑

```
print(urlparse('/exam/sites/all/modules/pubdlcnt/pubdlcnt.php?file=http:'))
```

```
ParseResult [scheme='', netloc='', path='/exam/sites/all/modules/pubdlcnt/pubdlcnt.php', params='', query='file=http://140.112.115.12/exam/sites/default/files/exam/graduate/106g/106_graduate_4.pdf&nid=5814', fragment='']
```

練習 02-1: 檔案下載與 URL 轉換 (5 ~ 8 mins)

目標程式：o2_1_observe_urljoin.py

目標：觀察不同的情況透過 urljoin 的結果

- 開頭有無斜線與兩條斜線的差別
- 不斷回到上一層的結果

```
print(urljoin(response.url, '105g/'))
print(urljoin(response.url, '/105g/'))
print(urljoin(response.url, '//facebook.com'))
print(urljoin(response.url, '../'))
print(urljoin(response.url, '../../'))
```

練習 02-2: 檔案下載與 URL 轉換 (5 ~ 8 mins)

目標程式：o2_2_download_history_exam.py

目標網站：<http://140.112.115.12/exam/graduate>

目標：下載頁面上的所有考古題 (25 份)

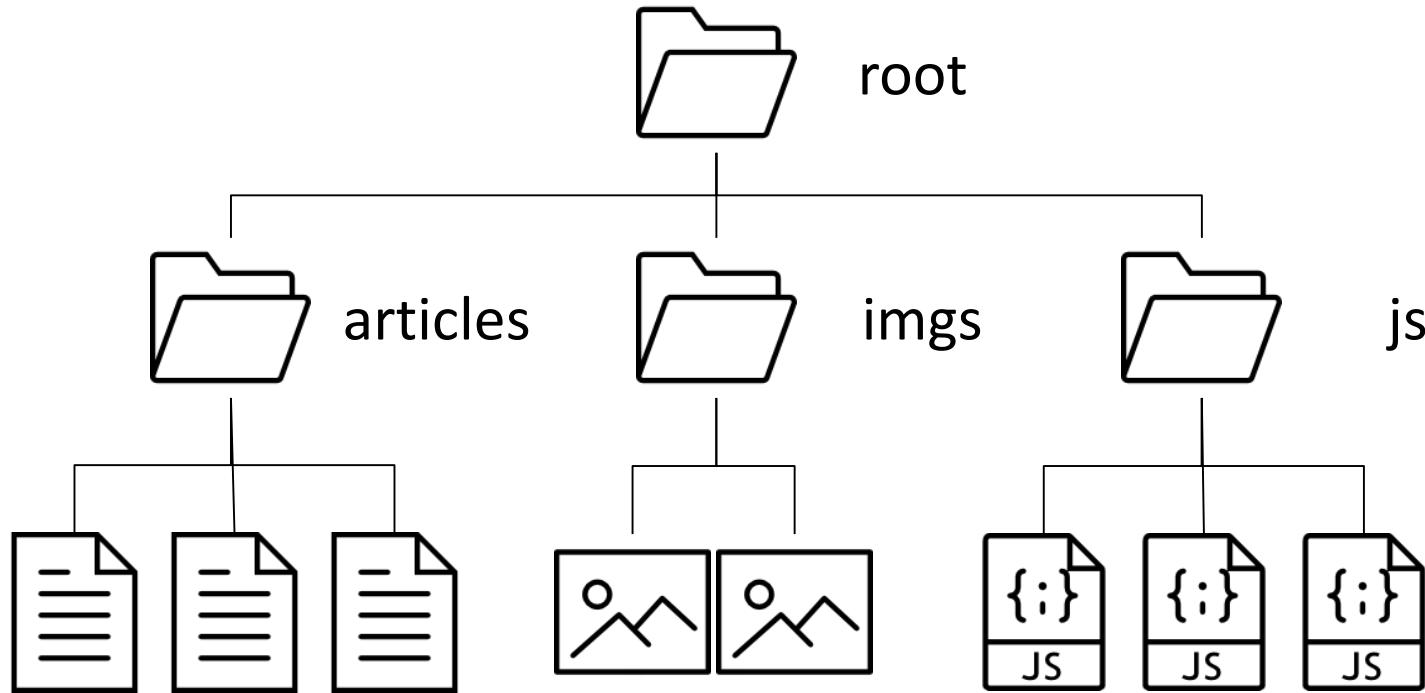
- 定位超連結的 tag
- 將超連結的相對路徑轉換成絕對路徑
- 字串處理取得要下載的檔案名稱
- 下載檔案

Outline

- 靜態網頁以外的爬蟲
 - 圖片爬蟲
 - 檔案爬蟲
 - 網站爬蟲
- 現實世界的爬蟲
 - 現代網站爬蟲衍生的問題
 - 動態網頁爬蟲

網站結構

- 網頁是一份 HTML 檔案
- 網站是一堆網頁以階層式的方式組成的集合



網站瀏覽行為

- 網站存放在遠端電腦中，我們稱該電腦為主機
- 網站瀏覽即是查看目標主機的不同檔案而已

故 事 寫給所有人的歷史

Home 全部文章 主題特輯 專欄作者 關於我們 f q

關鍵年代
人類如何走向了戰爭？

關鍵年代：二十世紀人類如何走向戰爭

關於主題特輯

你的、我的、他們的故事，即便距離再遠，都可能彼此關聯。「主題特輯」試著牽繫起古往今來的歷史線索，讓武則天與凱薩琳互訴衷曲，讓吸血鬼與閻羅王談論生命與死亡……每一個專輯，都是愛麗絲的樹洞。這次，你打算跌進哪一個王國呢？

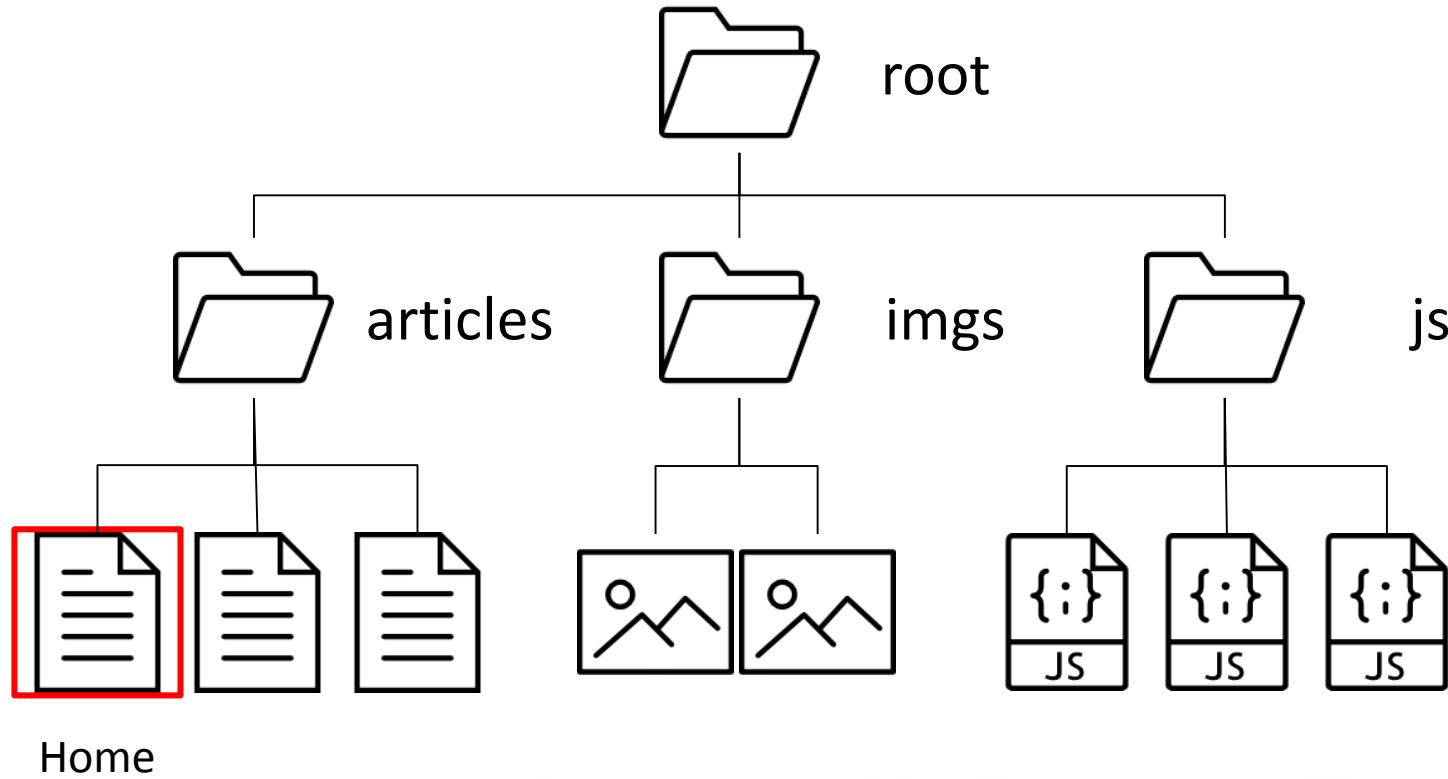
特輯列表

研之有物

雞年談雞

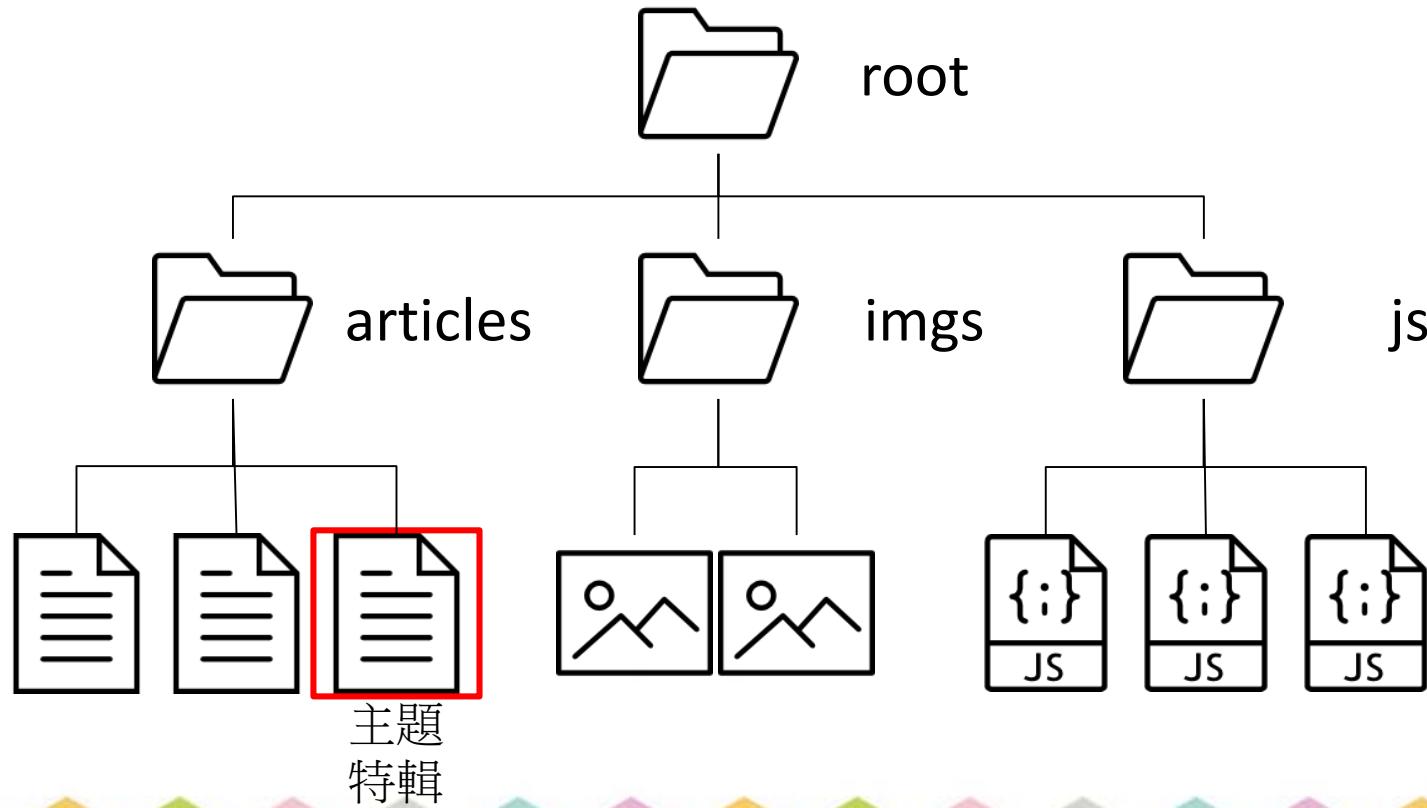
網站瀏覽行為

- 網站存放在遠端電腦中，我們稱該電腦為主機
- 網站瀏覽即是查看目標主機的不同檔案而已



網站瀏覽行為

- 網站存放在遠端電腦中，我們稱該電腦為主機
- 網站瀏覽即是查看目標主機的不同檔案而已



遍歷網站 - 從網頁連結到其他網頁

- 透過開發者工具查看
- 超連結即是 tag



Home



主題
特輯

```
<a href="https://gushi.tw/featurette/" title="主題特輯">主題特輯<i class="fa fa-angle-right fa-dropdown"></i></a>
```



檢查 href 路徑之後送 request
(urljoin, requests)

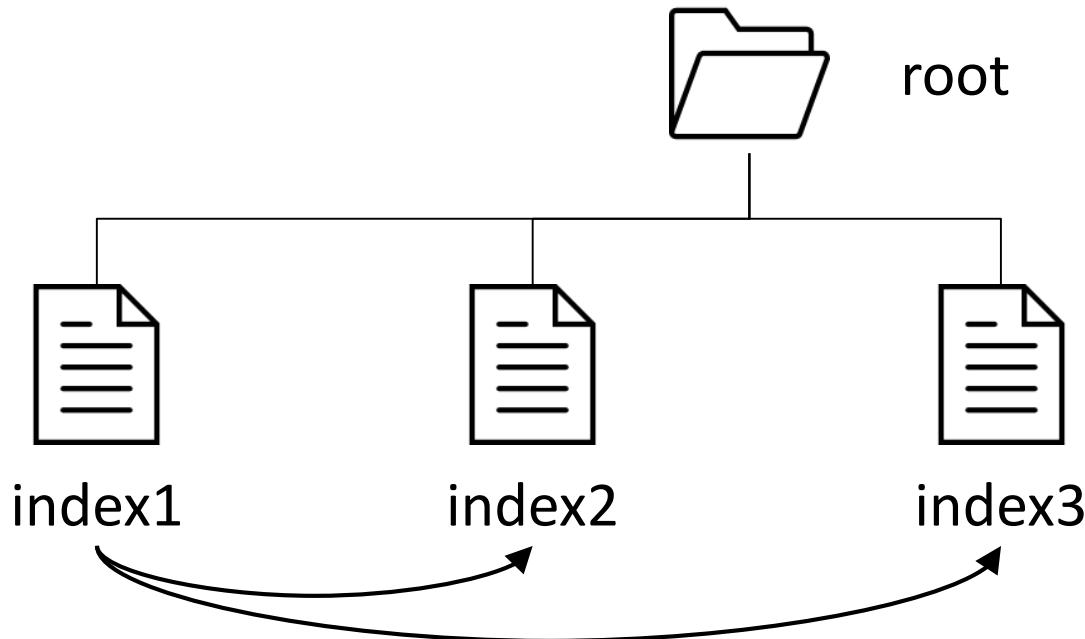


取得主題特輯頁面



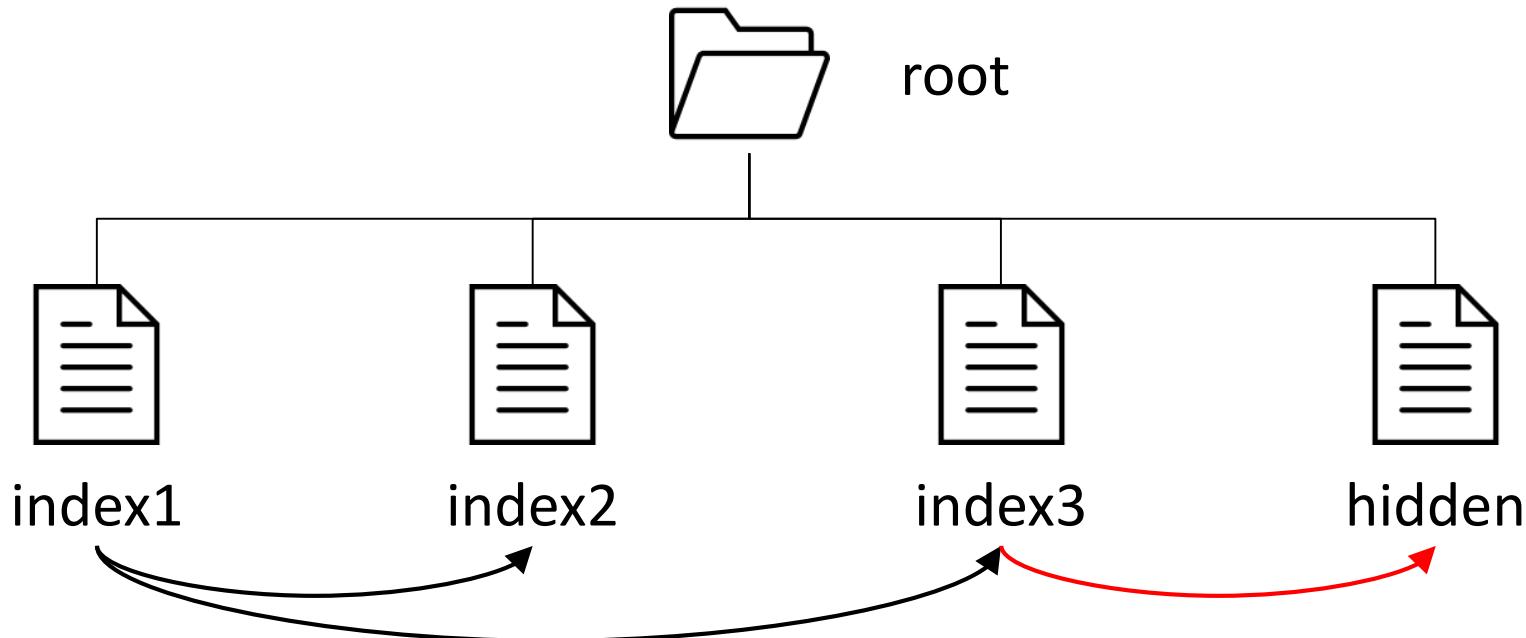
透過迴圈尋訪網站

- 透過迴圈對所有網址超連結都送出 request



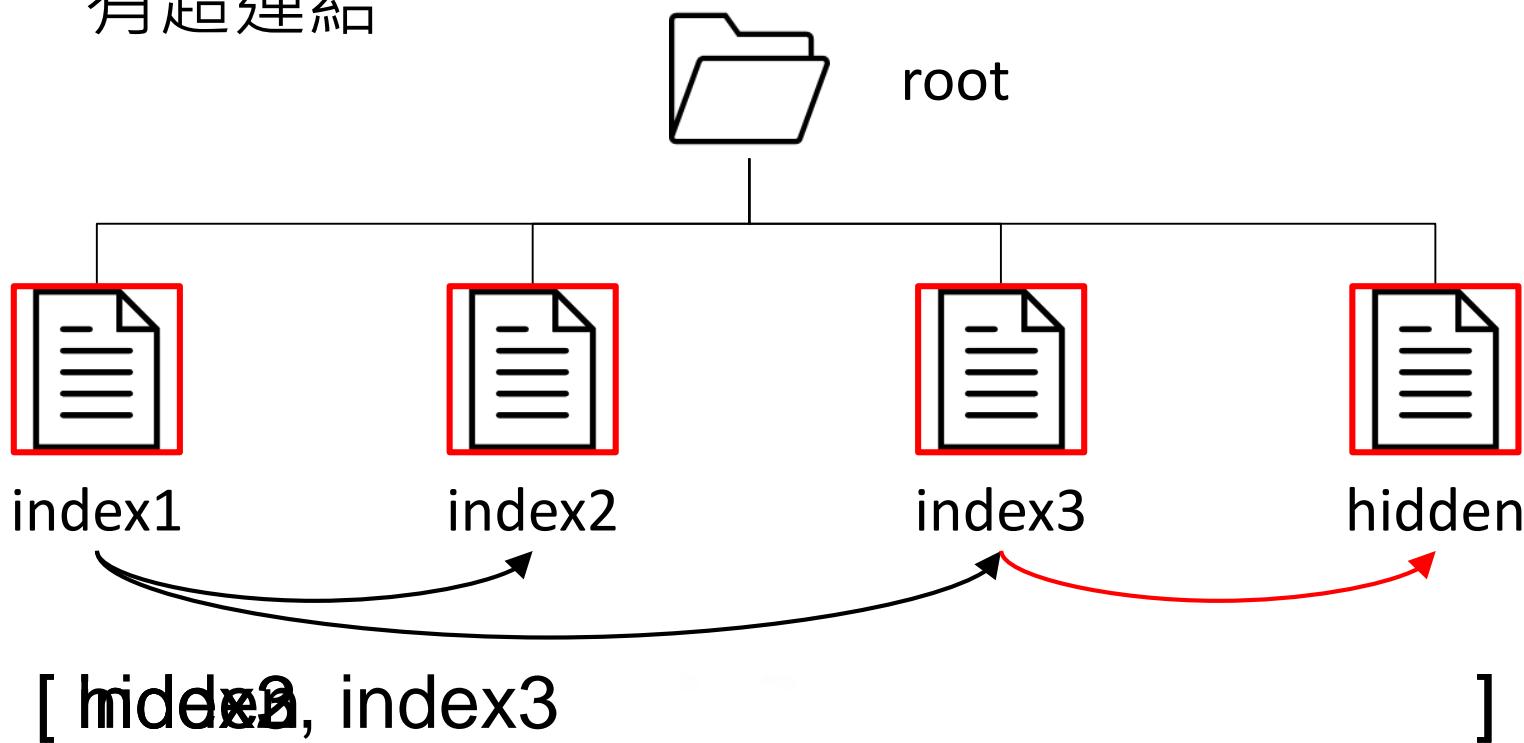
透過迴圈尋訪網站

- 並不是所有網頁的超連結都會出現在首頁
- 只做一次迴圈無法發現其他網頁裡的超連結



遍歷網站

- 看過網站所有超連結 = 看過所有網頁所有超連結
- 紀錄所有需要送 requests 的超連結，直到送過所有超連結



遍歷網站 - 直到看過所有連結

- 宣告一個 list 儲存即將要送 request 的網址
- 決定送 request 的中止條件

```
# 儲存即將要送 request 的網址  
wait_list = ['https://afuntw.github.io/demo-  
crawling/demo-page/ex1/index1.html']
```

```
# 當 wait list 裡還有網址的時候...  
while wait_list != []:  
    # 送 request 的流程
```

遍歷網站 - 更新 wait_list 清單

- 從 wait_list 中取出網址
- 從 wait_list 中刪除已經取出的網址
- 從 wait_list 中放入新的網址

```
# 從 wait_list 中取出第一個網址並更新  
url = wait_list.pop(0)
```

```
# 從 wait_list 中放入新的網址  
wait_list.append(new_url)
```

練習 03: 取得真正的所有標題 (5~8 mins)

目標程式 : o3_crawling_demo1_hidden.py

目標網站 : [demo_website_1](#)

目標 : 透過超連結不斷爬取多個網頁的 h1 tag

- 將需要送 request 的超連結存入等待清單
- 不斷的拿等待清單的超連結送 request

Python Crawling Home About Contact

頁面 4 - Hidden page

恭喜你！這裡是隱藏網頁, 看到這個訊息就代表你會用 recursive 的方法遍歷網站了



現實中網站的超連結設計

- 網頁間可以互相超連結
- 導覽列



遍歷網站的迴圈問題

Python Crawling Home **About** **Contact**

頁面 1 - Home

This is a template for a simple marketing or informational website. It includes a header with a logo and supporting pieces of content. Use it as a starting point to create your own unique website.

[Next Page »](#)

Python Crawling **Home** About **Contact**

頁面 2 - About

This is a template for a simple marketing or informational website. It includes a header with a logo and supporting pieces of content. Use it as a starting point to create your own unique website.

[Next Page »](#)

wait list

- About**
- Contact**
- Contact

1. wait list 有重複 URL
2. 打算對 Home 重複存取

解決遍歷網站的迴圈問題

- 建立一個已經送過 request 的清單 viewed_list
- wait_list 裏面也不能有重複的 URL

```
viewed_list = []
# 將送過 request 的網址存入 viewed_list
viewed_list.append(url)

# 檢查新網址沒有出現在 wait_list 與 viewed_list
if new_url not in wait_list and new_url not in
viewed_list:
    wait_list.append(new_url)
```

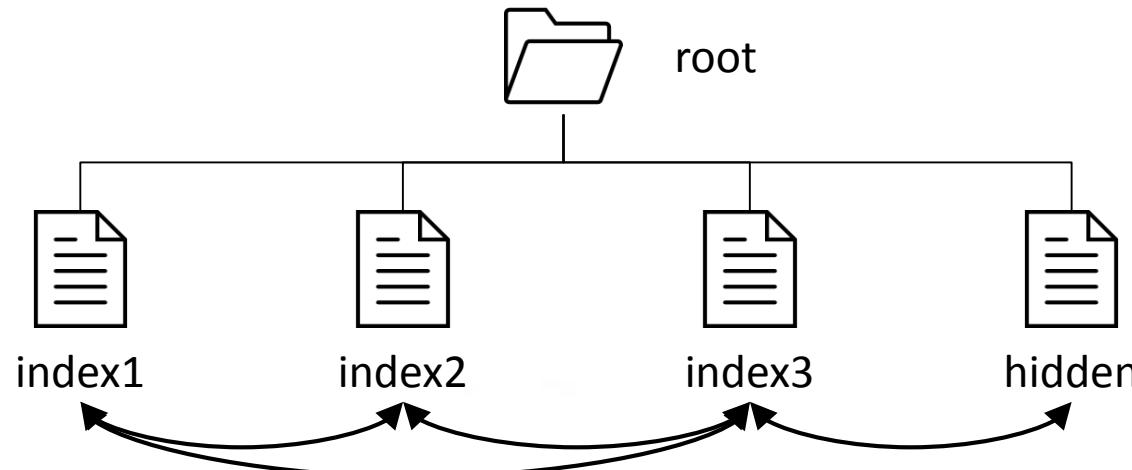
練習 04: 避免迴圈問題 (5~8 mins)

目標程式 : o4_crawling_demo2_no_infinite.py

目標網站 : demo_website_2

目標 : 避免無窮迴圈的爬取網站的 h1 tag

- 將需要送 request 的超連結存入等待清單
- 紀錄送過的 request



Summary

- 圖片爬蟲 & 檔案爬蟲
 - 如果下載檔案被拒絕可以嘗試加上 User-Agent
 - 必要時下載圖片要檢查格式
 - 送 request 之前記得確認路徑是否為絕對路徑
- 網站爬蟲
 - 超連結網頁裡的超連結網頁也要送 request
 - 紀錄存取過的網頁

Outline

- 靜態網頁以外的爬蟲
 - 圖片爬蟲
 - 檔案爬蟲
 - 網站爬蟲
- 現實世界的爬蟲
 - 現代網站爬蟲衍生的問題
 - 動態網頁爬蟲

href 不全然是你想要的網址

- 並非定義一串網址，而是超連結
- urljoin 回傳的值並非全部都可以送 request

聯絡我們

服務時間: 週一至週五 09:00~18:00

聯絡電話: (02) 2783-9427

聯絡信箱: secretary@datasci.tw



```
<a href="mailto:secretary@datasci.tw">secretary@datasci.tw</a>
```

你對 href 夠了解嗎？

href 可能的值	敘述	範例
absolute URL	絕對路徑	https://gushi.tw/
relative URL	相對路徑	/ex1/html1.html
anchor	同一頁面的其他 tag	#top
other protocols	其他協定	mailto://example@gmail.com
JavaScript	程式碼	javascript:console.log("Hello")

- 對 anchor 送出 request 會拿到同樣的頁面
- 網頁一般只會用 HTTP 或是 HTTPS 協定
- 無法對程式碼送出 request



過濾 href

```
import re
# 過濾锚點, e.g. #top
check_url_1 = re.match('#.*', url) # True/False

# 過濾其他協定, 只接受 http/https
from urllib.parse import urlparse
check_url_2 = urlparse(url).scheme not in
['https', 'http']

# 過濾程式碼, e.g. javascript:alert();
check_url_3 = re.match('^\wjavascript.*', url)
```

練習 05: 過濾 href (5~10 mins)

目標程式 : o5_crawling_demo3_filter_href.py

目標網站 : [demo_website_3](#)

目標 : 過濾不必要的超連結並取得網站的所有 h2 tag

- 判斷過濾不必要的超連結的時機
 - urljoin 前還是後？

```
# 觀察下列的值並嘗試送出 request
anchor = urljoin(url, '#top')
protocol = urljoin(url,
'mailto:example@gmail.com')
code = urljoin(url, 'javascript:alert("Hi");')
```

符合絕對路徑的 url 一定沒問題？

- 符合絕對路徑的 URL 可以送 request
- 可以送 request 的 URL 不代表是你需要的

關於我們

課程資訊

推廣活動

人才媒合

學習資源

參與我們



台灣資料科學年會
@twdsconf

Home
Posts
Services

2017 台灣資料科學年會

2017/11/9 (Thu) ~ 2017/11/12 (Sun)
中央研究院人文社會科學館



Liked ▾ Following ▾ Share ...

Book Now

Message

使用 urlparse 的極限

- ❑ urlparse 可分析的 URL 片段

shceme://netloc/path;params?query#fragment

e.g. http://www.facebook.com/twdsconf

- ❑ 完全不一樣的網站仍然可透過判斷 netloc 決定
- ❑ 若是網站有子網站就會被忽略

```
url = 'http://tdsf.iis.sinica.edu.tw/'  
fb = 'https://www.facebook.com/twdsconf/'
```

```
print(urlparse(url).netloc)  
print(urlparse(fb).netloc)
```

tdsf.iis.sinica.edu.tw

www.facebook.com

子網域, 網域與後綴

- 透過 urlparse 取得的 netloc 可以再拆解
- netloc = 子網域.網域.後綴

e.g. www.facebook.com

子網域 網域 後綴

台大首頁 http://www.ntu.edu.tw

台大中文 http://www.cl.ntu.edu.tw

台大法律 http://www.law.ntu.edu.tw

台大化工 http://www.che.ntu.edu.tw

...

透過 tldextract 分析網域

- 將 netloc 分段成子網域.網域.後綴並不是單純透過 “.” 來切割字串就好
- 不同的網站可以用 netloc 或是 domain 判斷
- 子網站用 netloc 無法判斷

```
from tldextract import extract
print(extract('http://tdsf.iis.sinica.edu.tw'))
print(extract('http://www.ntu.edu.tw'))
print(extract('http://www.ntu.edu.tw/'))
print(extract('http://www.cl.ntu.edu.tw'))
```

```
ExtractResult(subdomain='tdsf.iis', domain='sinica', suffix='edu.tw')
ExtractResult(subdomain='www', domain='ntu', suffix='edu.tw')
ExtractResult(subdomain='www', domain='ntu', suffix='edu.tw')
ExtractResult(subdomain='www.cl', domain='ntu', suffix='edu.tw')
```

練習 o6: 檢查域名 (5~8 mins)

目標程式 : o6_crawling_demo4_extract_domain.py

目標網站 : demo_website_4

目標 : 分析網域 , 只對 www 或是跟原本網址一樣的
sub domain 送出 request

- 過濾不必要的超連結後再分析域名

```
# 延伸思考：短網址或是路徑為 ip 拆解的結果？  
from tldextract import extract  
print(extract('https://goo.gl/z3z1G7'))  
print(extract('https://127.0.0.1:80'))
```

Bonus: Google 短網址服務

- 網站轉址無法透過 URL 判斷是否屬於相同的網域
- 需要送一次 request 再從回傳的 response 中取得原始 URL

```
from tldextract import extract
extract_url = extract(url)
# 判斷是否為 google 短網址
if extract_url.domain == 'goo' or
extract_url.suffix == 'gl':
    response = requests.get(extract_url)
    print(response.url)
```

爬網站的重點回顧

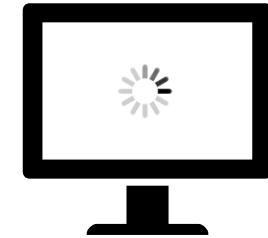
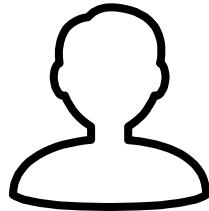
- 若是要尋訪所有網頁，要不斷對找到的網址送 request，並且紀錄尋訪過的網址
- href 的值要經過過濾，取出符合網址格式
- 分析網址格式與域名確認識自己想要爬的網頁

Outline

- 靜態網頁以外的爬蟲
 - 圖片爬蟲
 - 檔案爬蟲
 - 網站爬蟲
- 現實世界的爬蟲
 - 現代網站爬蟲衍生的問題
 - 動態網頁爬蟲

甚麼是動態網頁？

- 透過程式變動網頁架構
- 需要時間載入資料或是不同使用行為等都會透過動態網頁的方式呈現



故事募資計劃
打造臺灣的人文知識新媒體

2015 年的 9 月，「故事」就踏上了它的歷史」正式登場。從那時開始，一直到今天，一千多個日子以來，感謝許多讀者朋友的支持。我們一步一步前進，而目標還在前方，所以我們希望和你一起，把故事繼續說下去！

最新文章

專欄作者

- 老尸上譯了
- 山林碎語 NEW
- 冷知識週刊 NEW
- 謝金魚的金魚缸
- Emery 的歷史角落
- 胡川安的 Life Circus
- 有問題要跟海鷺說
- 尋「臺」啟事
- 馬雅圖史館 NEW
- 所有專欄

故事
打造臺灣的人文知識新媒體

故事募資計劃
打造臺灣的人文知識新媒體

最新文章

專欄作者

- 老尸上譯了
- 山林碎語 NEW
- 冷知識週刊 NEW
- 謝金魚的金魚缸
- Emery 的歷史角落
- 胡川安的 Life Circus
- 有問題要跟海鷺說
- 尋「臺」啟事
- 馬雅圖史館 NEW
- 所有專欄

靜態網頁 vs. 動態網頁

- 透過 requests.get 拿到的是靜態網頁
- 檢視網頁原始碼看到的是靜態網頁
- 開發者工具 (inspect) 看到的是動態網頁

靜態網頁

The screenshot shows a static website for a crowdfunding project. The header features a cartoon illustration of a boat on water. Below it, the text reads '故事募資計劃 打造臺灣的人文知識新媒體'. The main content area contains a message from the founder: '2015 年的 9 月，『故事』誠招所有人的歷史」正式登場。從那時開始，一直到今天，一千多個日子以來，感謝許許多多讀者朋友的支持。我們一步一步前進，而目標還在前方，所以我們希望和你一起，把故事繼續說下去！」. On the right, there's a sidebar with a '最新文章' section and a '專欄作者' section.

動態網頁

The screenshot shows a dynamic website for the same crowdfunding project. The layout is identical to the static version, but the content is generated dynamically. The main message from the founder is displayed in a larger, bold font. The sidebar on the right also contains dynamic content, such as '最新文章' and '專欄作者' sections, which are populated with real-time data from a database.



檢查網頁是靜態還是動態

- Chrome extension (Quick Javascript Switcher)
- 透過工具的開關檢查頁面是否有變化

關掉 JavaScript

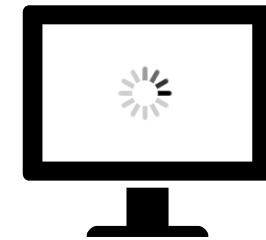
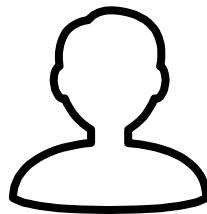


打開 JavaScript



取得程式更新後的網頁

- 送出 request 之後等到網頁 loading 完成再要求回傳網頁



故事 真地所有人的歷史

故事募資計劃
打造臺灣的人文知識新媒體

方，所以我們希望和你一起，把故事繼續說下去！

Home 全部文章 主題特輯 專欄作者 關於我們

最新文章

-
-

專欄作者

- 老尸上課了 山林碎語 NEW
- 冷知識週刊 謝金魚的金魚缸 Emery 的歷史角落 胡川安的 Life Circus
- 有問題要跟海鵠說 寶「臺」啟事 馬雅歷史館
- 所有專欄

對藝術家來說，世界不僅僅是由幾個一百年的圖樣

故事 真地所有人的歷史

故事募資計劃
打造臺灣的人文知識新媒體

方，所以我們希望和你一起，把故事繼續說下去！

Home 全部文章 主題特輯 專欄作者 關於我們

最新文章

-
-

專欄作者

- 老尸上課了 山林碎語 NEW
- 冷知識週刊 謝金魚的金魚缸 Emery 的歷史角落 胡川安的 Life Circus
- 有問題要跟海鵠說 寶「臺」啟事 馬雅歷史館
- 所有專欄

對藝術家來說，世界不僅僅是由幾個一百年的圖樣

模擬使用者操作瀏覽器的行為

- Selenium 本來是網頁自動測試工具
- Selenium 經常被拿來處理動態網頁爬蟲
- 因為是模擬操作瀏覽器，速度上會比靜態網頁慢

```
from selenium import webdriver
# 透過指定的瀏覽器 driver 打開 Chrome
driver =
webdriver.Chrome('../webdriver/chromedriver')
# 透過瀏覽器取得網頁
driver.get('https://afuntw.github.io/demo-
crawling/demo-page/ex4/index1.html')
```

瀏覽器的大小也會影響網頁結構

- 現代網站根據使用者裝置的大小會有不同的呈現



故事 寫給所有人的歷史

Home

全部文章

主題特輯

專欄作者

關於我們

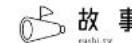


故事募資計劃
打造臺灣的人文知識新媒體

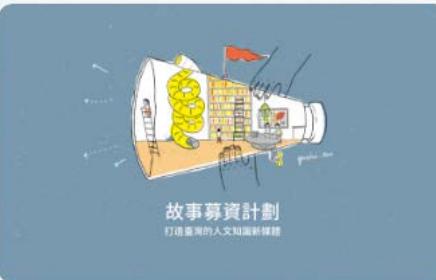


故事會員招募中

2015 年的 9 月，「故事：寫給所有人的歷史」正式登場。從那時開始，一直到今天，一千多個日子以來，感謝許許多多讀者朋友的支持。我們一步一步前進，而目標還在前方，所以我們希望和你一起，把故事繼續說下去！



故事
gushi.tw



故事募資計劃
打造臺灣的人文知識新媒體



故事會員招募中

瀏覽器視窗最大化

- 一開始打開瀏覽器的時候並非視窗最大化
- 建議一般使用情況都先將瀏覽器視窗最大化

```
from selenium import webdriver

# 透過指定的瀏覽器 driver 打開 Chrome
driver =
webdriver.Chrome('..../webdriver/chromedriver')
# 將瀏覽器視窗最大化
driver.maximize_window()
```

Selenium 定位 tag

- 大致上與 BeautifulSoup 定位 tag 的方法相似
- `find_element_by_id()`
- `find_element_by_tag_name()`
- `find_element_by_class_name()`
- ...

```
# e.g. 取得 id='first' 的 tag
id_tag = drver.find_element_by_id('first')
print(id_tag)
```

```
<selenium.webdriver.remote.webelement.WebElement (session="d348c1e5b8f4
b246f6e9a7d5a69e21b7", element="0.038819750219358795-1")>
```

練習 07: 靜態與動態爬蟲的差異 (5 mins)

目標程式：07_crawling_demo4_selenium.py

目標網站：[demo_website_4](#)

目標：分別透過 requests 與 Selenium 爬網站上
id = 'first' 的 tag

- 開啟瀏覽器並且將視窗最大化
- 透過 Selenium 定位 tag 並取得文字資料

```
>>> requests: First featurette heading. It'll  
blow your mind.
```

```
>>> selenium: Rendered by Javascript
```

了解 tag 之間的關係

```
<bookstore>
  <book>
    <title>Harry Potter</title>
    <author>K. Rowling</author>
  </book>
</bookstore>
```

- ❑ book 是 title 與 author 的 parent
- ❑ title 與 author 都是 book 的 child

了解 tag 之間的關係

```
<bookstore>
  <book>
    <title>Harry Potter</title>
    <author>K. Rowling</author>
  </book>
</bookstore>
```

- title 與 author 各自為對方的 sibling
- bookstore, book 都是 title 與 author 的 ancestors
- book, title, author 都是 bookstore 的 descendant

Selenium 定位 tag - XPath

- Selenium 還可透過類似路徑寫法的 XPath 定位 tag

語法	意義
/	從 root 開始選擇
//	從任何地方開始選擇
.	選擇當下這個 node
..	選擇當下這個 node 的 parent node
@	選擇 attribute
*	選擇任何 node
	OR

XPath 範例

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title></title>
5 </head>
6 <body>
7     <div></div>
8     <div></div>
9     <div>
10        <div></div>
11        <div></div>
12        <div></div>
13    </div>
14 </body>
15 </html>
```

html



body



第三個 div



第一個 div

XPath 範例

- BeautifulSoup 寫法
 - soup.find_all('div')[2].find_all('div')[0]

```
from selenium import webdriver

# 打開瀏覽器，視窗最大化，對目標網址送 request...
# 尋找一個 html > body > div[2] > div[0]
h2 = driver.find_element_by_xpath(
    '/html/body/div[2]/div[0]')
```

XPath 範例

- 透過 By 可以更簡單更換定位方式

```
from selenium import webdriver
from selenium.webdriver.common.by import By

# 打開瀏覽器，視窗最大化，對目標網址送 request...
# 尋找網頁中所有的 p tag
p = driver.find_elements(By.XPATH, '//p')
```

XPath 範例

```
from selenium import webdriver
from selenium.webdriver.common.by import By

# 尋找任何一個 id = 'first' 的 tag
h2 = driver.find_element(By.XPATH,
    '//*[@id="first"]')

# 尋找網頁中 id = 'second' 或 'third' 的 h2 tag
p = driver.find_elements(By.XPATH,
    '//h2[@id="second"] | //h2[@id="third"]')
```

XPath helper

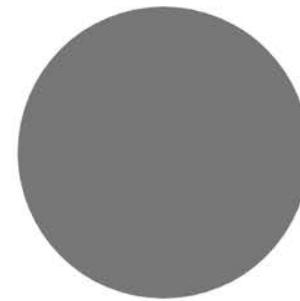
□ Chrome extension (XPath helper)

QUERY

```
//h2
```

RESULTS (6)

```
Home Heading 1
Home Heading 2
Home Heading 3
Rendered by Javascript
Oh yeah, it's that good. See for yourself.
```



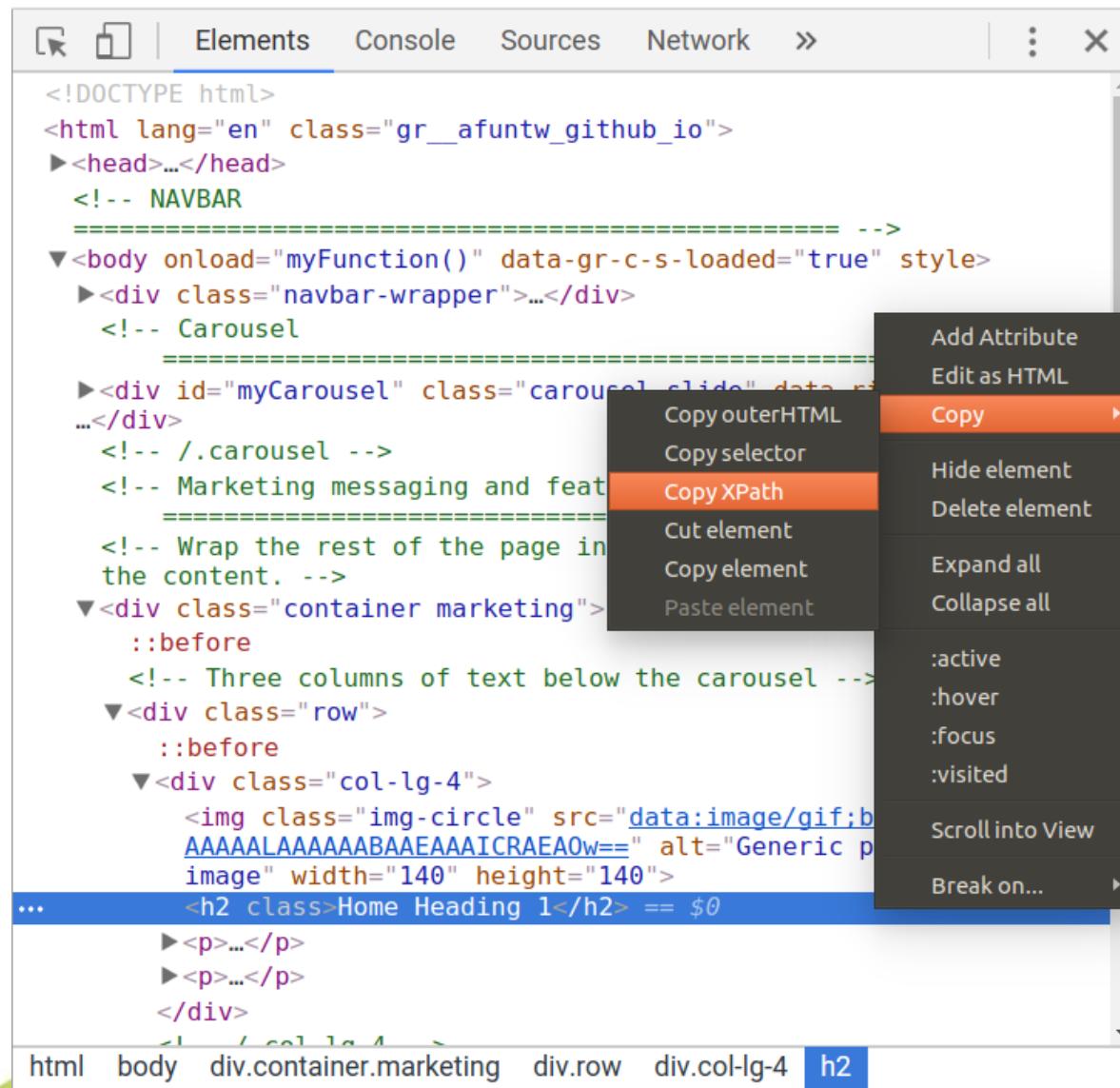
Home Heading 1

Donec sed odio dui. Etiam porta sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Praesent commodo cursus magna.

[View first »](#)



透過開發者工具取得 XPath



練習 08: 透過 XPath 做動態爬蟲 (10 mins)

目標程式：o8_crawling_pchome_selenium.py

目標網站：<http://24h.pchome.com.tw/region/DHBE>

目標：取得頁面上調列商品的名稱與價格 (30 項)

 窄邊框，一年保 ASUS B9440UA-0251A7 200U (i5-7200U/8G/256G) 7代Core i5 // SSD // 輕1.05 網路價\$32900詳	 14.0'' i5-7200U 940MX 2G 1TB FHD 青光霧面 (商)Lenovo ThinkPad T 470 20HRA00STW(i5-72 Lenovo ThinkPad T470 20 網路價\$39900詳	 i7-7600U SSD W10P (商)HP X360 1030 G2(i7-7600U/8G*2/512GB SS ▼7代i7輕薄商務★軍規認 網路價\$68900詳	 i5-7200U 500GB 4G DDR4 1.85Kg (商)HP 240 G6 (i5-7200 U/14G/500GB/Win10) ▼7代Core i5★商務必備▼ 網路價\$18900詳	 15.6'' i7-7820HQ M2200M FHD 1TB+256GB DELL M7520(i7-7820H Q/Nvidia Quadro M2200 【酷炫狂潮 戴爾再現】 網路價\$89990詳
 Win 10 Pro，一年保 ASUS P2430UJ-0321A6 200U (i5-6200U/4G/500G) ★6代商務效能獨顯機★ 網路價\$22900詳	 14.0'' i7-7500U Win10 Pro 256G SSD 1.13Kg (商)Lenovo ThinkPad X 1c 20HRA010TW(i7-750 超輕巧薄型—14吋小黑機· 網路價\$59900詳	 Core i5-7200U 1.59kg Win10Home 500GB (商) HP Probook 430 G4 (i5-7200U/4G DDR4/500 ▼7代i5★商務必備▼搶！ 網路價\$25900詳	 i5-7200U 500GB 4G DDR4 1.85Kg (商)HP 240 G6 (i5-7200 U/14G/500GB/Win10pr ▼7代Core i5★商務必備▼ 網路價\$23900詳	 15.6'' i7-7820HQ Win10Pro 512GB Nvidia M1200 DELL M5520(i7-7820H Q/M1200M/512GB SSD/ 【酷炫狂潮 戴爾再現】 網路價\$98990詳
 Win 10 Pro，一年保固 ASUS P2530UJ-0461A6	 14.0'' i7-7600U 512G SSD Win10P 三年全球保固 (商)Lenovo ThinkPad T	 15.6'' i7 SSD Win10Pro (商)HP Probook 650 G3	 Core i3-6006U DVD W10 500GB (商)HP 240 G6 (i3-6006	 窄邊框，一年保 ASUS B9440UA-0251A7

Summary

- 現代網站爬蟲衍生的問題
 - 過濾非必要的 href
 - 解析網域判斷是否要存取該網頁
- 動態網頁
 - 透過 Selenium 取得程式改變後的網頁結構
 - 透過 XPath 定位 tag

爬蟲不一定要爬網頁

- 有 API (Application Programming Interface) 的話就透過 API 做爬蟲 (e.g. Facebook Graph API)

Graph API Explorer Application: [?] Graph API Explorer ▾

Access Token:  [REDACTED] 

 GET ▾ → /v2.10/twdsconf?fields=about  

Learn more about the Graph API syntax

Node: twdsconf

about
+ Search for a field

{
 "about": "台灣資料科學年會 | http://datasci.tw/
台灣資料科學協會 | http://foundation.datasci.tw/

台灣資料科學愛好者交流區
<https://www.facebook.com/groups/datasci.tw/>,
 "id": "1502894619932654"
}

Reference

- 維基 User Agent 詳細格式說明
- href 定義
- XPath cheat sheet
- Selenium 文件