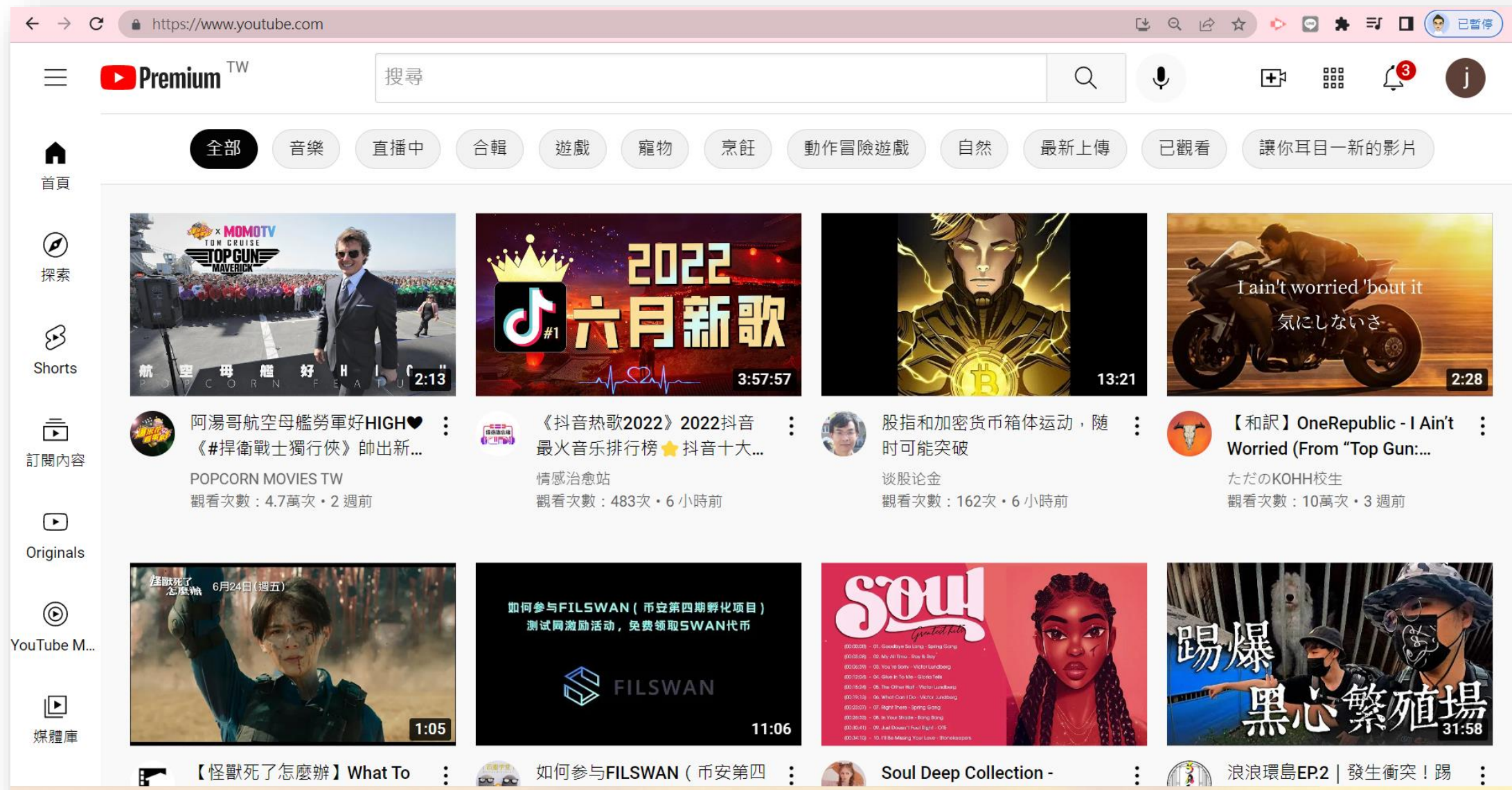





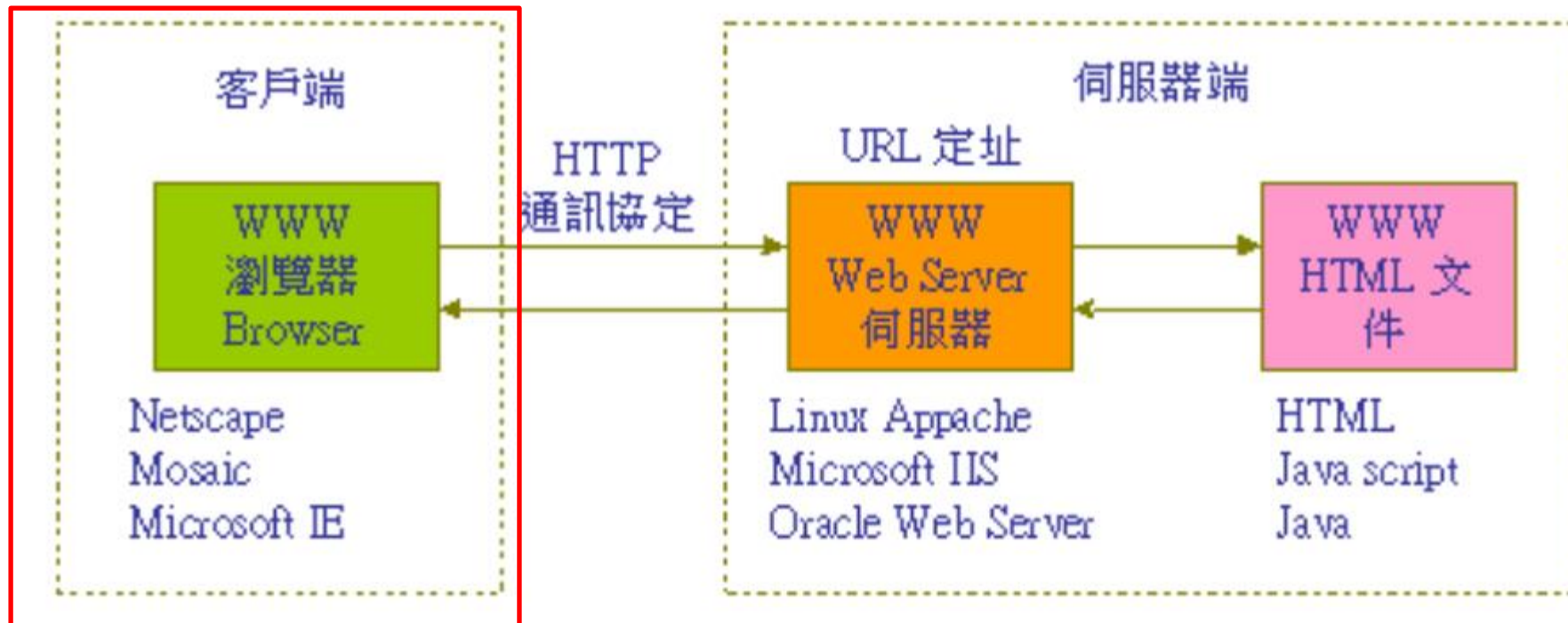
Python 程式教學 Chapter20

- 網頁格式介紹
- requests套件
- 初探bs4套件

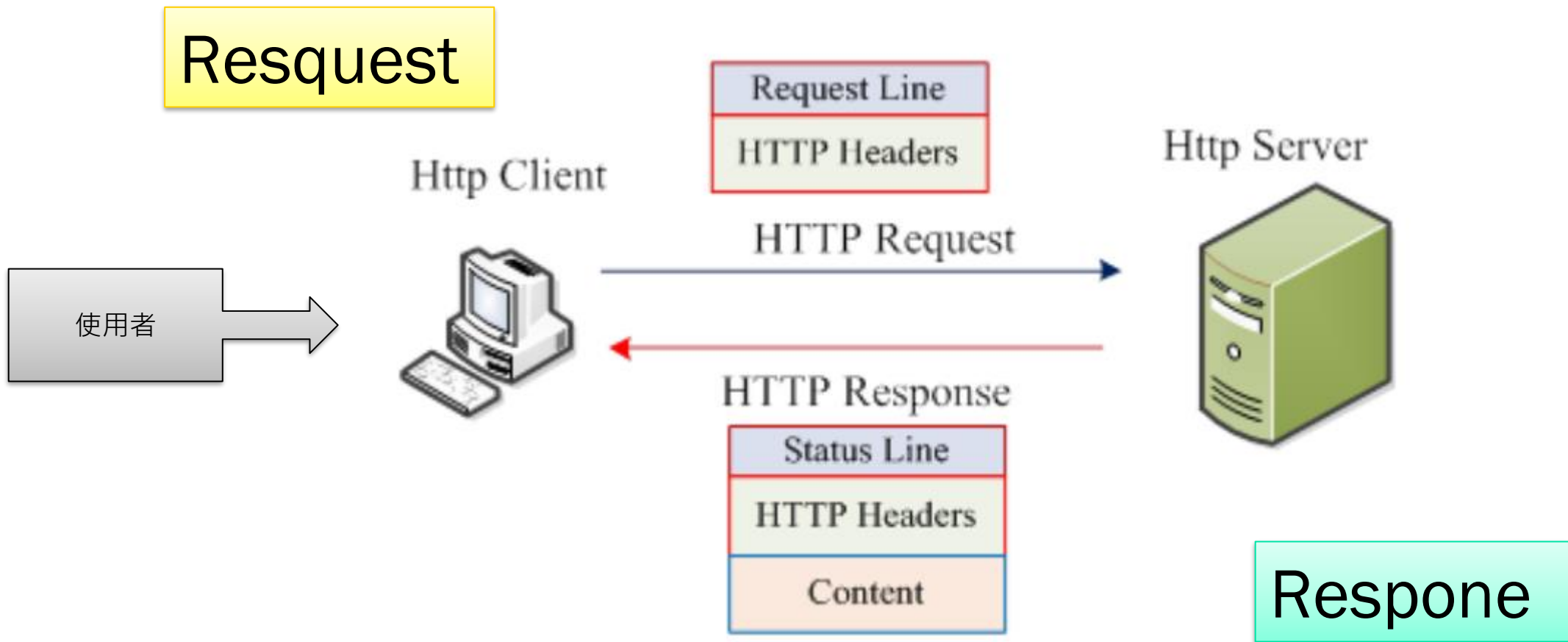
■ <https://www.youtube.com/>



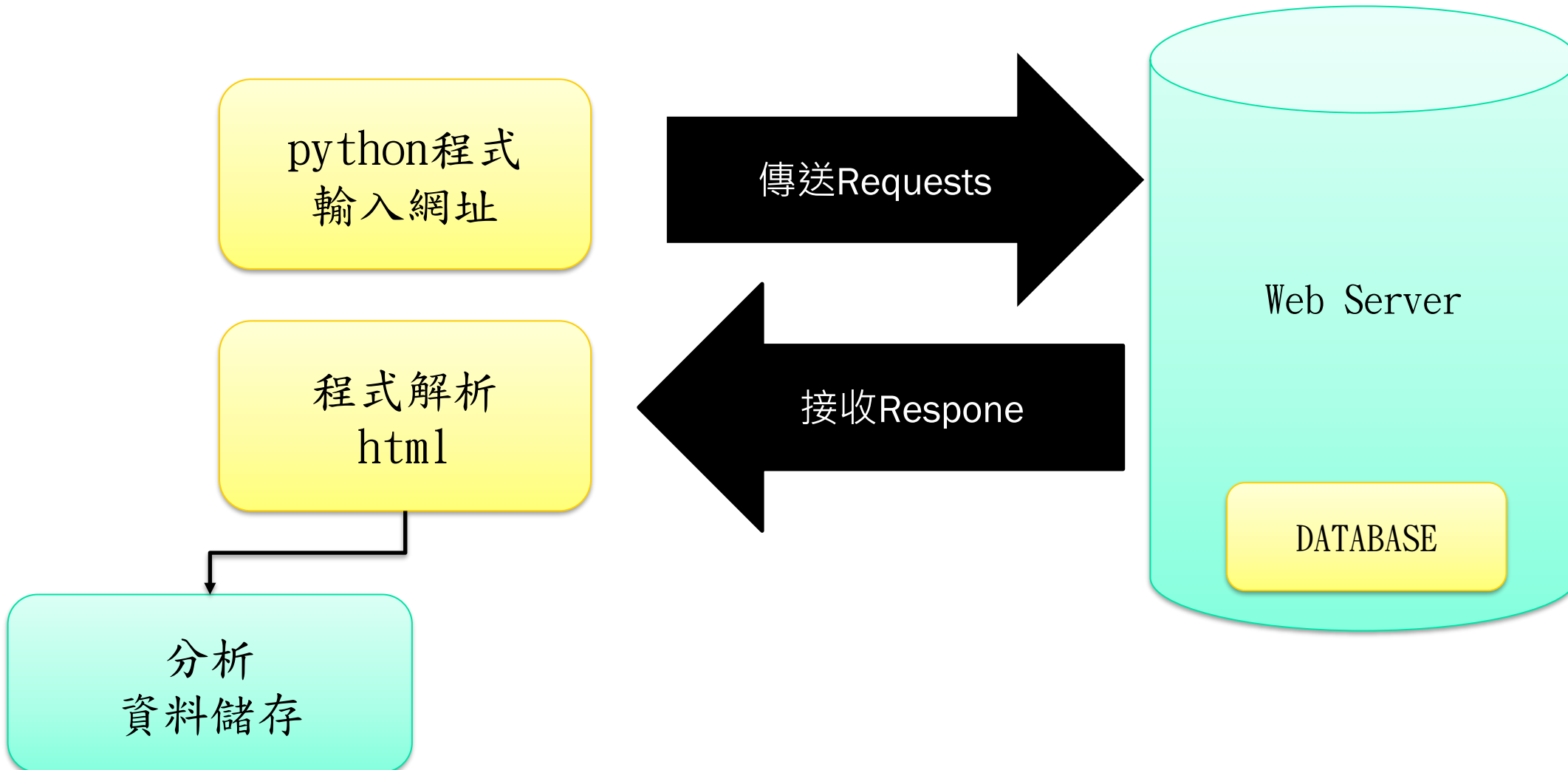
- http是種通訊協定，協助我們找到server並做溝通， python
以取得資料進而呈現在螢幕或網頁上。
- https(s→ssl→Secure)



- Client 透過 **Request** 跟Server要資料，
- Server 回傳 **Response** 給Client顯示資料。



爬蟲過程





requests模組

- `import requests`
- 使用 `requests.get` 傳送需求(網址)
- 等待回傳成功
- 回傳html格式
- 進行解析



```
import requests
```

```
url="http://www.yahoo.com.tw"
```

```
resp=requests.get(url)
```

```
print(resp. text)
```

```
remote: '/_td_remote',
xhr: '/_td_api'
};
```

```
YUI.namespace('Env.My.settings').context = {
    uh_js_file : '',
    videoAsyncEnabled: 0,
    videoplayerScriptElementId: 'videoplayerJs',
    videoplayerUrl: 'https://s.yimg.com/rx/builds/7.86.842.1557330173/zh-hant-tw/
videoplayer-nextgen-desktop-min.js',
    videoAutoplay: 1,
    videoLooping: 0,
    videoForcedError: 0,
    videoFullscreen: 1,
    videoHtml5: 1,
    videoMinControls: 1,
    videoCmsEnv: 'prod',
    videoMustWatch: 0,
    videoMWSticky: 0,
    videoMute: 1,
    videoQosRate: 1,
    videoBuffering: 0,
    videoRelated: 0,
    videoMaxRate: 0
};
```

偵測回傳是否成功

```
if resp.status_code == requests.codes.ok:  
    print("OK")
```

```
if resp.status_code == 200:  
    print("OK")
```

回傳200表示成功!
失敗代碼: 404



404

Page not found

The Page you are looking for doesn't exist or an other error occurred.
Go back, or head over to weebly.com to choose a new direction.

■ 網頁回傳代碼

200 OK

201 **Post** 指令被成功地執行

202 請求被接受

203 **Get**或**Head**請求被完成

204 請求被完成，但沒有內容傳回

300 資源可在許多地方被找到

301 資源永久的移除

302 資源暫時被移除

304 資源未被更改過

400 客戶端不正確的請求信息

401 未授權的請求訊息

402 完成請求信息必須有回應

403 禁止使用此資源

404 資源找不到

405 資源不允許使用此方法

406 不被接受的資源型別

410 無此資源

500 伺服器內部發生錯誤

501 沒有被實作的方法

502 不正確的閘道或伺服器負荷過重

503 無此服務或閘道逾時 200 秒

html 網頁

- test.html





html 基礎知識

1. `<!DOCTYPE html>`: 定義文件類型版本跟HTML語法版本
2. HTML 文件包含在`<html>` 和`</html>` 標籤之間
3. (meta) 和腳本 (script) 聲明包含在`<head>` 和`</head>` 標籤之間
4. 網站上可見的部分包含在`<body>` 和`</body>` 標籤之間
5. `<h1>` 和`<h6>` 標籤之間的部分為網站標題
6. `<p>` 用於定義段落的標籤
7. `<a>` 是超鏈接的標籤
8. `<table>` 是表格的標籤
9. `<tr>` 是表格行的標籤
10. `<td>` 是表格列的標籤

■ test.html



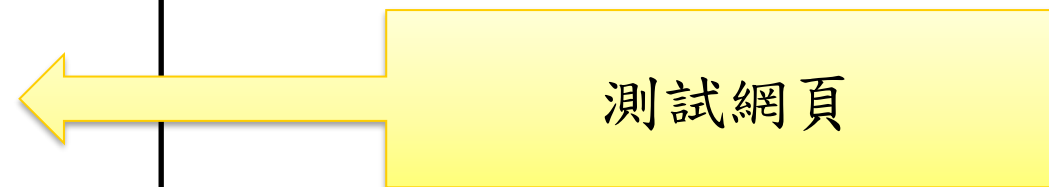
```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <title>測試網頁</title>
</head>

<body>
  <h2>網頁標題</h2>
  <p>測試爬蟲及連結</p>
  <a id="link1" href="https://www.yahoo.com.tw/">Link 1</a>
  <a id="link2" href="https://www.youtube.com/">Link 2</a>
  <a id="link3" href="https://www.sina.com.tw/">Link 3</a>
  <p>Hello, <b class="boldtext">Bold Text</b></p>
</body>

</html>
```

- <https://rbdkd67zcc92ou2hpwvxa-on.drv.tw/www/test.html>





使用requests讀取

```
1 url='https://rbdkd67zcc92ou2hpwvxa-on.driv.tw/www/test.html'  
2  
3 resp=requests.get(url)  
4 resp.encoding = 'utf-8'  
5  
6 resp.text
```

```
'<html>\r\n<head><title>測試網頁</title></head>\r\n<body>\r\n<h2>網頁標題</h2>\r\n<p>測試爬蟲及連結</p>\r\n<a id="link1" href  
="https://www.yahoo.com.tw/">Link 1</a>\r\n<a id="link2" href="https://www.youtube.com/">Link 2</a>\r\n<a id="link3" href="h  
ttps://www.sina.com.tw/">Link 2</a>\r\n<p>Hello, <b class="boldtext">Bold Text</b></p>\r\n</body>\r\n</html>'
```



BeautifulSoup

BeautifulSoup是一個Python套件，功能包括解析HTML、XML文件、修復標籤等錯誤的文件，以便提取其中的資料，在網路資料採集時非常有用。

安裝方式：

```
pip install bs4
```



```
1 from bs4 import BeautifulSoup
2
3 soup=BeautifulSoup(resp.text, 'lxml')
4
5 soup
```

```
<html>
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>
<title>測試網頁</title>
</head>
<body>
<h2>網頁標題</h2>
<p>測試爬蟲及連結</p>
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>
<a href="https://www.youtube.com/" id="link2">Link 2</a>
<a href="https://www.sina.com.tw/" id="link3">Link 3</a>
<p>Hello, <b class="boldtext">Bold Text</b></p>
<script async="" src="https://drv.tw/inc/wd.js"></script></body>
</html>
```

解析器

`lxml` 套件是用來作為 `BeautifulSoup` 的解析器 (Parser)，`BeautifulSoup` 可以支援的解析器其實不只一種，還有 `html.parser` (Python 內建) 與 `html5lib`，根據官方文件的推薦，我們使用解析速度最快的 `lxml`。

```
soup=BeautifulSoup(resp.text, 'lxml')
```

可改用其他解析器

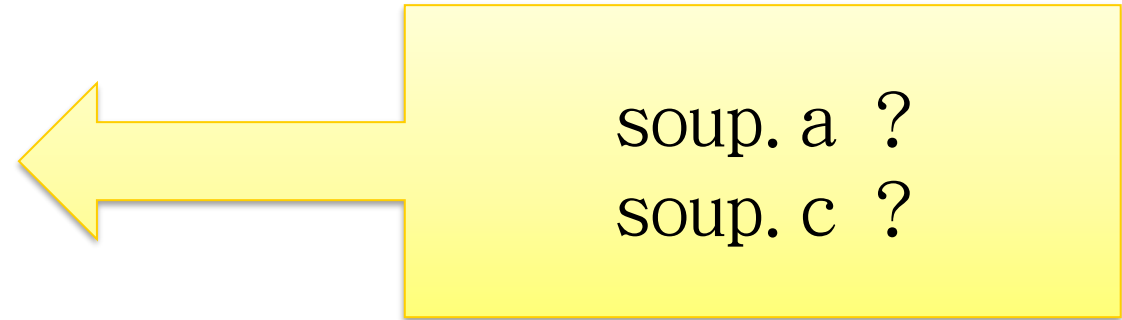
解析流程



取得標籤跟標籤內容

```
<html>
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>
<title>測試網頁</title>
</head>
<body>
<h2>網頁標題</h2>
<p>測試爬蟲及連結</p>
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>
<a href="https://www.youtube.com/" id="link2">Link 2</a>
<a href="https://www.sina.com.tw/" id="link3">Link 2</a>
<p>Hello, <b class="boldtext">Bold Text</b></p>
<script async="" src="https://drv.tw/inc/wd.js"></script></body>
</html>
```

```
print(soup.title)
print(soup.title.text)
print(soup.h2)
print(soup.h2.text)
print(type(soup.p))
print(type(soup.p.text))
```



```
<title>測試網頁</title>
測試網頁
<h2>網頁標題</h2>
網頁標題
<class 'bs4.element.Tag'>
<class 'str'>
```

```
import requests
from bs4 import BeautifulSoup
```

```
url='https://rbdkd67zcc92ou2hpwvxa-on.drv.tw/www/test.html'
resp=requests.get(url)
resp.encoding='utf-8'
```

```
if resp.status_code == requests.codes.ok:
```

```
    soup=BeautifulSoup(resp.text, 'lxml')
    print(soup)
    print(soup.title)
    print(soup.title.text)
    print(soup.h2)
    print(soup.h2.text)
    print(type(soup.p))
    print(type(soup.p.text))
    print(soup.c)
```

```
<title>測試網頁</title>
測試網頁
<h2>網頁標題</h2>
網頁標題
<class 'bs4.element.Tag'>
<class 'str'>
None
```

無此標籤則為None



find (單一搜尋)

```
link_tag=soup.find('a')  
print(link_tag)
```

```
1 link_tag=soup.find('a')  
2 link_tag
```

```
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>
```



```
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>
```

- `<a>` → 標籤

- `href, id` → 屬性

`href` → "http://www.yahoo.com.tw/" → (內容)

`id` → "link1" → (內容)

- `Link 1` → 標籤 `<a>` 的連結文字(text)



使用get方法取得屬性內容

```
print(link_tag)
print(link_tag.text)
print(link_tag.get('href'))
print(link_tag.get('id'))
```

```
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>
Link 1
https://www.yahoo.com.tw/
link1|
```

find_all (重複搜尋)

```
link_tags=soup.find_all('a')  
print(link_tags)
```



透過find_all查詢如
果回傳值為
`bs4.element.ResultSet`

```
[<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>,  
  <a href="https://www.youtube.com/" id="link2">Link 2</a>,  
  <a href="https://www.sina.com.tw/" id="link3">Link 3</a>]
```

```
link_tags=soup.find_all('a')  
print(link_tags)
```

```
for link in link_tags:  
    print(link.text)
```

```
for link in link_tags:  
    print(link.get('href'))  
    print(link.get('id'))
```

Link 1

Link 2

Link 3

<https://www.yahoo.com.tw/>

link1

<https://www.youtube.com/>


link2

<https://www.sina.com.tw/>

link3

- 也可以使用這種方式取屬性的值

```
for link in link_tags:  
    print(link['href'])  
    print(link['id'])
```



```
for link in link_tags:  
    print(link.get('href'))  
    print(link.get('id'))
```



同時搜尋多種標籤

```
tags = soup.find_all(["a", "b"])\nprint(tags)
```

```
[<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>,\n <a href="https://www.youtube.com/" id="link2">Link 2</a>, <a\n href="https://www.sina.com.tw/" id="link3">Link 3</a>, <b cl\n ass="boldtext">Bold Text</b>]
```



使用id查詢

- 使用id查詢

```
link1_tag = soup.find(id="link1")  
print(link1_tag)
```

```
<a href="http://www.yahoo.com.tw/" id="link1">Link 1</a>
```




使用標籤跟屬性查詢

- 更精準的查詢

```
link_tags=soup.find("a",href="http://www.yahoo.com.tw/")  
print(link_tags)
```

```
<a href="http://www.yahoo.com.tw/" id="link1">Link 1</a>
```



練習

- 請查找出第二個link的連結跟文字內容，並輸出以下內容
- 使用id方式查詢➡link2

Link 2

<https://www.youtube.com/>



程式碼

```
import requests
import webbrowser
from bs4 import BeautifulSoup

url='https://rbdkd67zcc92ou2hpwvxa-on.drv.tw/www/test.html'
resp=requests.get(url)
resp.encoding='utf-8'

if resp.status_code==requests.codes.ok:

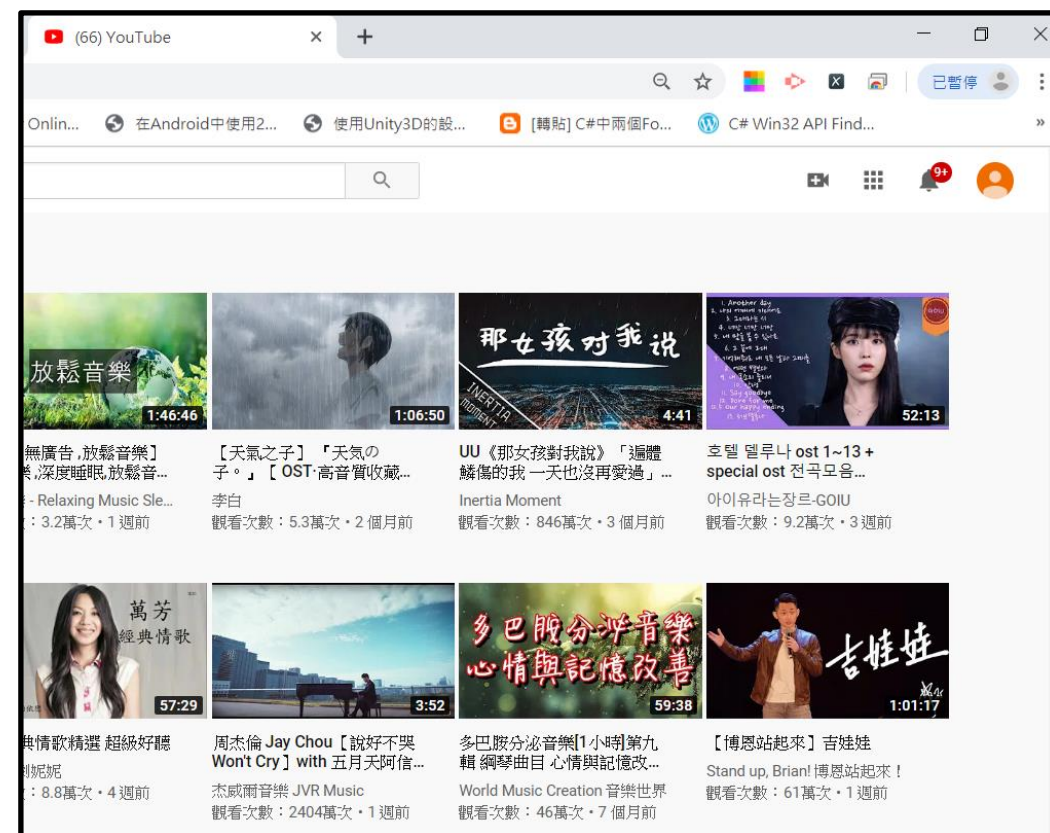
    soup=BeautifulSoup(resp.text,'lxml')
    link=soup.find(id='link2')
    print(link.text)
    print(link.get('href'))
```

開啟網頁

```
import webbrowser  
webbrowser.open(link.get('href'))
```

Link 2

<https://www.youtube.com/>



```
import requests
import webbrowser
from bs4 import BeautifulSoup

url = 'http://www.17app.url.tw/test.html'
resp = requests.get(url)
resp.encoding = 'utf-8'

if resp.status_code == requests.codes.ok:

    soup = BeautifulSoup(resp.text, 'lxml')
    link = soup.find(id='link2')
    print(link.text)
    print(link.get('href'))
    webbrowser.open(link.get('href'))
```

利用class來定位屬性

- class是python 保留字故不能使用class ="boldtext" ，只能使用class_

```
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>  
<a href="https://www.youtube.com/" id="link2">Link 2</a>  
<a href="https://www.sina.com.tw/" id="link3">Link 2</a>  
<p>Hello, <b class="boldtext">Bold Text</b></p>  
<script async="" src="https://drv.tw/inc/wd.js"></script></body>  
</html>
```

```
b_tag = soup.find_all("b", class_="boldtext")  
print(b_tag)  
print(b_tag.text)
```

透過本文文字去尋找

```
<a href="https://www.yahoo.com.tw/" id="link1">Link 1</a>  
<a href="https://www.youtube.com/" id="link2">Link 2</a>  
<a href="https://www.sina.com.tw/" id="link3">Link 3</a>  
<p>Hello, <b class="boldtext">Bold Text</b></p>  
<script async="" src="https://drv.tw/inc/wd.js"></script></body>  
</html>
```

```
print(soup.find("a", string="Link 3" ))
```



```
import requests
from bs4 import BeautifulSoup

url = 'http://www.17app.url.tw/test.html'
resp = requests.get(url)
resp.encoding = 'utf-8'

if resp.status_code == requests.codes.ok:

    soup = BeautifulSoup(resp.text, 'lxml')
    link = soup.find(id='link2')
    print(link.text)
    print(link.get('href'))
    #webbrowser.open(link.get('href'))

    print(soup.find("a", string="Link 3"))
    print(soup.find("a", string="Link 1"))
    print(soup.find("a", string="Link 2"))
```

Link 2

<https://www.youtube.com/>

[Link 3](https://www.sina.com.tw/)

[Link 1](https://www.yahoo.com.tw/)

[Link 2](https://www.youtube.com/)

程式練習

■ 查詢網址

- https://0pyvhrv8owyldzd42nlz4q-on.driv.tw/html/test_link.html

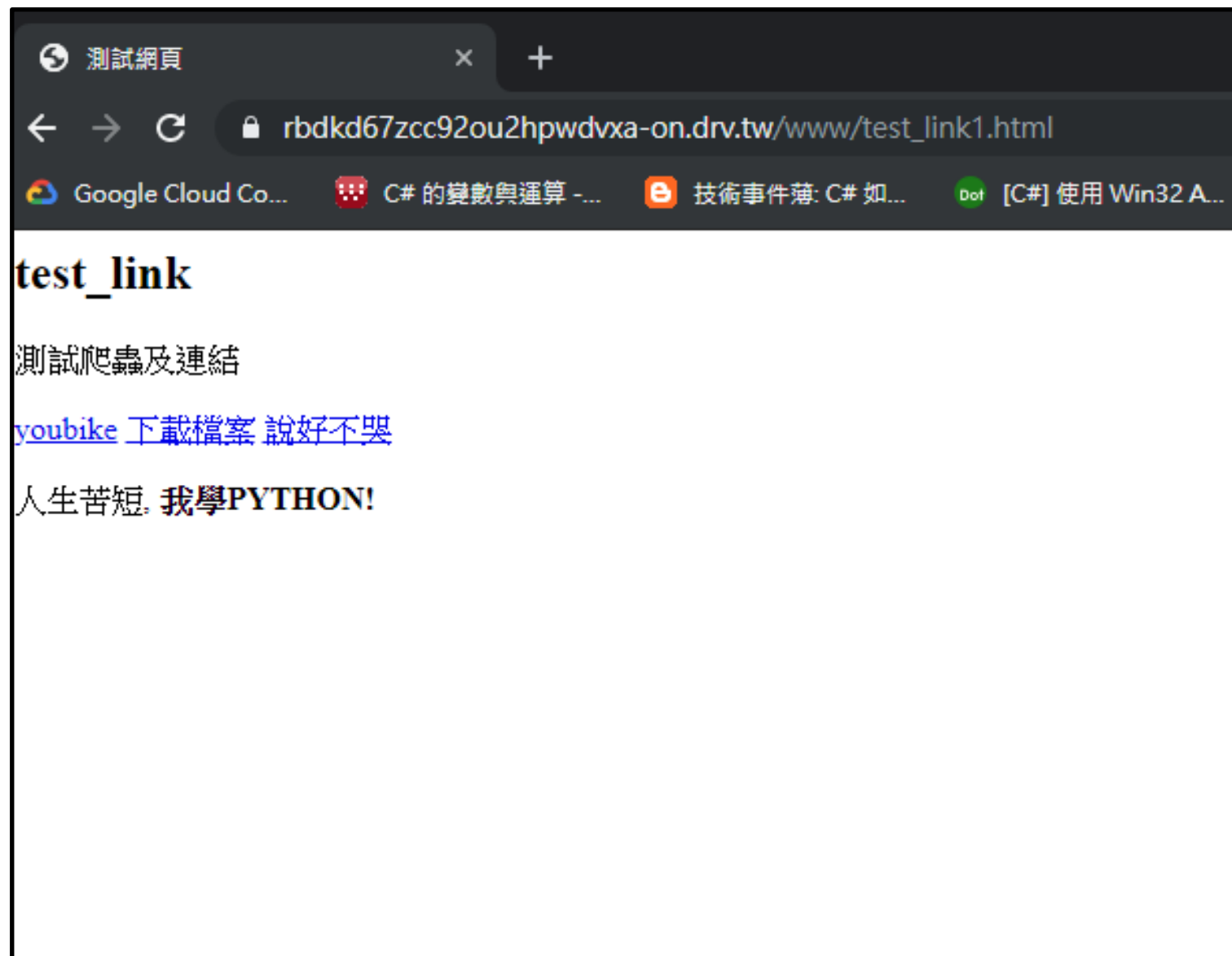
輸出以下資料

- 輸出title, p 標籤內容(text)
- 將連結跟本文內容用二維串列儲存
- 使用webbrowser 套件開啟最後一個link

測試爬蟲及連結

人生苦短，我學PYTHON!

```
[[ 'youbike', 'http://opendata.hccg.gov.tw/dataset/1f334249-9b55-4c42-aec1-5a8a8b5e07ca/  
resource/805975a5-3549-43a7-be9c-898f39117b3a/download/20190704113116284.json'], ['下載檔  
案', 'https://drive.google.com/open?id=1v3EMqCAqKMtinpxNbry6fWoYYtAGAlVm'], ['說好不哭',  
'https://www.youtube.com/watch?v=HK7SPnGSxLM']]
```





備註

標 記	描 述
<HTML> </HTML>	宣告網頁將以 HTML 編寫。
<HEAD> </HEAD>	定義網頁的檔頭。
<TITLE> ... </TITLE>	定義標題（並不在網頁上顯示）。
<BODY> ... </BODY>	框註內為網頁主體。
<Hn> ... </Hn>	n=1~6，框註內六個階層的標題字大小。
 ... 	設定框註內文字為粗體。
<I> </I>	設定框註內文字為斜體字。
 ... 	框註內為無序串列（註標式）。
 ... 	框註內為編號串列。
<MENU> ... </MENU>	框註內項目行成選單。
	串列項目的開始（並無 ）。
 	強迫分離。
<P>	區段開始。
<HR>	水平線。
<PRE> ... </PRE>	已格式化文字。
	在此載入影像圖形。
 ... 	定義超連結。

```
import requests
from bs4 import BeautifulSoup
import webbrowser
url = 'http://www.17app.url.tw/test_link.html'
resp = requests.get(url)
resp.encoding = 'utf-8'

datas = []
if resp.status_code == requests.codes.ok:
    ....
    ....#print(resp.text)
    ....soup = BeautifulSoup(resp.text, 'lxml')
    ....
    ....for p in soup.find_all('p'):
    ....    print(p.text)
    ....
    ....links = soup.find_all('a')
    ....
    ....for link in links:
    ....    #print(link['href'])
    ....    #print(link.text)
    ....    datas.append([link.text, link['href']])
    ....
print(datas)
```

```
webbrowser.open(datas[-1][-1])
```