中學生的Python趣學程
資料爬取與分析
Foundation of AI

跟著創新廣告和新創CEO
一起從創意發想, 程式素養, 創客修為到新創精神
玩轉程式趣學程

運算創意學園 http://www.941coding.com 編輯: 童教練

目次

封面改編圖像來源：攝圖網

Chapter 01 網頁資料擷取和轉換

之前我們已玩過 Python 的基本語法，選擇判斷，回圈，資料型態，檔案和異常處理… 等. 而這些都是接下來要玩網頁資料爬取和分析的基礎，而數據資料又是 AI 人工智慧的燃料，且網頁爬蟲資料分析其中重要環節就是相關 Pyhton 套件的運用，這些套件多數也應用於 AI. 可見它的重要性!

這篇網頁爬蟲資料分析主要以實戰應用為主，程式碼的數量和複雜度較之前的練習都提高許多，尤其對於國中生難度不小! 我們會以程式案例拆解的方式，降低難度，一步一步達到目標! 加油啦!

Python 網頁資料爬取過程一般如下:

1. 存取網站取得網頁內容(原始碼)
2. 解析網頁取得資料內容
3. 資料處理與展示

這章先來實作前兩項.

1-1 網址解析

我們擷取資料主要有取得 OpenData(開放資料)的檔案或抓取網頁中的內容資料. 不管哪一種，這些資料檔案和網頁內容都在網站的網頁中，都需要讓程式能早找到網站，那程式就需要知道網址解析網址.

要解析網址可用 Python 中 urllib 套件的 urlparse 函式，會傳回元組型態的 ParseResult 物件，物件屬性中可取得網址的各種參數.

我們以台灣銀行告牌匯率網頁為例,

網址為 => https://rate.bot.com.tw/xrt?Lang=zh-TW

(ch1-1_exchangeRateTable.py)

```
8 from urllib.parse import urlparse
9 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
10 xrt = urlparse(url)
11 print(xrt)
```

```
In [1]: runfile('G:/Course/Python_DataAnalysis/ch1-1_exchangeRateTable.py', wdir='G:/
Course/Python_DataAnalysis')
ParseResult(scheme='https', netloc='rate.bot.com.tw', path='/xrt', params='',
query='Lang=zh-TW', fragment='')
```

列	程式	說明
8	from urllib.parse import urlparse	由 urllib 套件的 parse 模組引入 urlparse 方法
9	url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'	將台灣銀行告牌匯率網頁的網址字串 'https://rate.bot.com.tw/xrt?Lang=zh-TW' 指派給變數 url
10	xrt = urlparse(url)	用引入的 urlparse 方法解析變數 url 的網址, 會傳回 ParseResult 的元祖物件, 元祖元素是網址的各項屬性
11	print(xrt)	輸出 ParseResult 的元祖物件: ParseResult(scheme='https', netloc='rate.bot.com.tw', path='/xrt', params='', query='Lang=zh-TW', fragment='') ParseResult 元祖物件屬性說明:

屬性	索引值	回傳值
scheme	0	scheme 通訊協定
netloc	1	網站域名
path	2	路徑
params	3	url 查詢參數 params 字串
query	4	query 查詢字串,GET 參數
fragment	5	框架名稱
port	無	通訊埠

繼續 ch1-1 輸出各屬性

```
8 from urllib.parse import urlparse
9 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
10 xrt = urlparse(url)
11 print(xrt)
12 print()
13 print('scheme =', xrt[0])
14 print('netloc =', xrt[1])
15 print('path =', xrt[2])
16 print('params =', xrt[3])
17 print('query =', xrt[4])
18 print('fragment =', xrt[5])
19 print('port =', xrt.port)
```

```
In [5]: runfile('G:/Course/Python_DataAnalysis/ch1-1_exchangeRateTable.py', wd
Course/Python_DataAnalysis')
ParseResult(scheme='https', netloc='rate.bot.com.tw', path='/xrt', params='',
query='Lang=zh-TW', fragment='')

scheme = https
netloc = rate.bot.com.tw
path = /xrt
params =
query = Lang=zh-TW
fragment =
port = None
```

列	程式	說明
13	print('scheme =', xrt[0])	輸出'shceme'字串和元祖索引值為 0 的第一個元素=>通訊協定: scheme = https
14	print('netloc =', xrt[1])	輸出'netloc'字串和元祖索引值為 1 的第二個元素=>網站域名: netloc = rate.bot.com.tw
15	print('path =', xrt[2])	輸出'path'字串和元祖索引值為 2 的第三個元素=>路徑: path = /xrt
16	print('params =', xrt[3])	輸出'params'字串和元祖索引值為 3 的第四個元素=>url 查詢參數: 不存在, 傳回空字串
17	print('query =', xrt[4])	輸出'query'字串和元祖索引值為 4 的第五個元素=>查詢字串: query = Lang=zh-TW
18	print('fragment =', xrt[5])	輸出'fragment'字串和元祖索引值為 5 的第六個元素=>框架名稱: 不存在, 傳回空字串
19	print('port =', xrt.port)	輸出'port'字串和元祖通訊埠, 因通訊埠在本機, 沒有索引值, 所以用 xrt.port 語法取得: 不存在, 傳回 None port = None

1-2 取得網頁內容

1-2-1 requests 套件取得網頁原始碼

requests 套件可以取得網頁原始碼. 流程如下:

1. 匯入 requests 套件. 使用 Anaconda 整合環境的好處是常用的數據分析和機器學習套件都已安裝, 直接匯入即可使用.
2. 用 requests.get()方法可模擬發出 HTTP GET 方法向伺服器送出請求(request).
3. 當伺服器接受請求後, 會回應(response)傳回網頁內容(原始碼).
4. 設定好編碼, 以 text 屬性取得網頁原始碼.

(ch1-2_requestsXR.py)

我們抓取上節的台灣銀行告牌匯率網頁的原始碼為例, 因該頁面原始碼有 2410 列程式敘述, 我取得這 2410 列後, 僅在螢幕輸出前 20 列.

首先看看網頁原始碼及要輸出的前 20 列:

1. 在網頁上按滑鼠右鍵，出現上圖選單
2. 在選單上選擇「檢視網頁原始碼(v)」按下
3. 即可出現該網頁的原始碼，如下圖:

這是網頁的 view-source

view-source:https://rate.bot.com.tw/xrt?Lang=zh-TW

```
<!DOCTYPE html>
<html lang="zh-TW" class="no-js">
<head>
    <meta charset="utf-8" />
    <title>臺灣銀行牌告匯率</title>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="description" content="臺灣銀行匯率利率黃金牌價查詢">
    <meta name="keywords" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no">
    <meta name="format-detection" content="telephone=no">
        <meta name="robots" content="index,follow" />

    <link rel="icon" type="image/x-icon" href="/favicon.ico">
    <link rel="stylesheet" href="/Content/css/font-awesome.min.css">


這是抓取 2410 列全部網頁原始碼後，要輸出的前 20 列


    <link rel="stylesheet" href="/Content/css/jquery-ui.min.css">
    <link rel="stylesheet" href="/Content/main-3.3.0.css">
    <style>
        h1.hero__header_small {
            font-size: 26px;
            margin: 12px 0;
        }

            h1.hero__header_eng {
                font-size: 20px;
                margin: -8px 0;
            }

        .chart-key-value .sp {
            position: relative;
            top: 2px;
        }

        select[name=month], select[name=day] {
            width: 5em;
        }

        :lang(en-US) {
```

```
8 import requests
9 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
10 html = requests.get(url)
11 html.encoding = 'utf-8'
12 htmllines = html.text.splitlines()
13 for i in range(20):
14     print(htmllines[i])
```

```
<!DOCTYPE html>
<html lang="zh-TW" class="no-js">
<head>
    <meta charset="utf-8" />
    <title>臺灣銀行牌告匯率</title>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="description" content="臺灣銀行匯率利率黃金牌價查詢">
    <meta name="keywords" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no">
    <meta name="format-detection" content="telephone=no">
        <meta name="robots" content="index,follow" />

    <link rel="icon" type="image/x-icon" href="/favicon.ico">
    <link rel="stylesheet" href="/Content/css/font-awesome.min.css">




    <link rel="stylesheet" href="/Content/css/jquery-ui.min.css">
```

列	程式	說明
8	import requests	匯入 requests 套件
9	url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'	將台灣銀行告牌匯率網頁的網址字串 'https://rate.bot.com.tw/xrt?Lang=zh-TW' 指派給變數 url
10	html = requests.get(url)	用 requests.get(url)方法向 url 網址所在伺服器發出請求，伺服器接受請求後，傳回 url 網址網頁的原始碼，並指定給變數 html
11	html.encoding = 'utf-8'	設定 utf-8 編碼 到 11 列，使用 html.text 已取得該網頁 2410 列的所有原始碼，但原始碼過多，繼續處理以輸出前 20 列觀察結果
12	htmllines = html.text.splitlines()	html.text 取得的原始碼用 splitlines()方法，分割成以列為元素的串列．並指派給變數 htmllines
13, 14	for i in range(20): 　　print(htmllines[i])	用 for 回圈迭代 20 次, 14 列輸出 htmllines 串列索引值 0-19 的前 20 個元素，即原始碼前 20 列

我們也可以修改第 13 列程式敘述，輸出後 20 列原始碼．程式另存為 (ch1-2_requestsXR2.py)

```
8 import requests
9 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
10 html = requests.get(url)
11 html.encoding = 'utf-8'
12 htmllines = html.text.splitlines()
13 for i in range(len(htmllines)-20, len(htmllines)):
14     print(htmllines[i])
```

```
            //footable 關閉所有fsear表格內容
            $(".click_CollapseAll").click(function () {
                $('.footable').trigger('footable_collapse_all');
            })

            //取得最新xx
            $(".click_reload").on("click", function (e) {
                location.reload();
            });

        });

        $(window).load(function () {
            //最近一個營業日不可比較幣別，比較幣別為空時不顯示現金&即期(for chrome)
            search_range_check($('input:radio:checked[name="search_range"]'));
            //alert("load");
        })
    </script>
</body>
</html>
```

輸出結果和原始碼比較

```
2385
2386            //footable 展開所有表格內容
2387            $(".click_ExpandAll").click(function () {
2388                $('.footable').trigger('footable_expand_all');
2389            })
2390
2391            //footable 關閉所有fsear表格內容
2392            $(".click_CollapseAll").click(function () {
2393                $('.footable').trigger('footable_collapse_all');
2394            })
2395
2396            //取得最新xx
2397            $(".click_reload").on("click", function (e) {
2398                location.reload();
2399            });
2400                                    原始碼後 20 列
2401        });
2402
2403        $(window).load(function () {
2404            //最近一個營業日不可比較幣別，比較幣別為空時不顯示現金&即期(for chrome)
2405            search_range_check($('input:radio:checked[name="search_range"]'));
2406            //alert("load");
2407        })
2408    </script>
2409 </body>
2410 </html>
```

哇! 果然有 2410 列程式敘述, 超多的吧?!

列	程式	說明
13, 14	for i in range(len(htmllines)-20, len(htmllines)): print(htmllines[i])	用 for 回圈迭代 20 次, 14 列輸出 htmllines 串列索引值 len(htmllines)-20 到 len(htmllines)的最後 20 個元素, 即原始碼最後 20 列

如果進一步需了解「匯率」這個詞在網頁出現的次數, 一樣將程式另存為 (ch1-3_requestsXRFrequency.py)

```
 8 import requests
 9 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
10 html = requests.get(url)
11 html.encoding = 'utf-8'
12 htmllines = html.text.splitlines()
13 for i in range(len(htmllines)-13, len(htmllines)):
14     print(htmllines[i])
15
16 print('='*16)
17 count = 0
18 for line in htmllines:
19     if '匯率' in line:
20         count += 1
21 print('"匯率"共出現{}次'.format(count))
```

```
                location.reload();
            });

        });

        $(window).load(function () {
            //最近一個營業日不可比較幣別，比較幣別為空時不顯示現金&即期(for chrome)
            search_range_check($('input:radio:checked[name="search_range"]'));
            //alert("load");
        })
    </script>
</body>
</html>
================
"匯率"共出現60次
```

列	程式	說明
13	for i in range(len(htmllines)-13, len(htmllines)):	用 for 回圈迭代 13 次, 14 列輸出 htmllines 串列索引值 len(htmllines)-13 到 len(htmllines)的最後 13 個元素, 即原始碼最後 13 列, 輸出較少資料, 方便後續程式輸

		出觀察結果.
16	print('='*16)	輸出 16 個等號 '=', 用以區隔兩類輸出.
17	count = 0	定義變數 count, 設初始值為 0, 作為匯率出現次數的計數器.
18	for line in htmllines:	For 回圈以變數 line 作為迭代串列 htmllines 的元素.
19, 20	if '匯率' in line: count += 1	If 判斷式, 如果'匯率'有在 line(原始碼的 2410 列,迭代 2410 次)中,則變數 count 加一
21	print('" 匯 率 " 共 出 現 {} 次 '.format(count))	For 回圈迭代完成, 輸出出現總次數 ========== "匯率"共出現 60 次

1-2-2 BeautifulSoup 套件解析網頁

我們爬取網頁內容, 一般不是需要整個網站或網頁資料, 而是其中特定資料. 例如, 台灣銀行告牌匯率網頁假設只需台銀各幣別現金買入匯率, 如下紅框標示的 19 種幣別和台幣的匯率.

這時就需要功能更強的網頁解析工具 BeautifulSoup 套件. 流程如下:

1. 從 bs4 套件匯入 BeautifulSoup 模組. 使用 Anaconda 整合環境的好處是常用的數據分析和機器學習套件都已安裝, 直接匯入即可使用.
2. 用 html.parser 解析 requests 取得的原使碼. 語發如下:
   sp = BeautifulSoup(原使碼, 'html.parser')
   sp 為 BeautifulSoup 的物件
3. 接下來就可以始用 BeautifulSoup 物件 sp 的方法和屬性解析網頁. BeautifulSoup 物件的常用方法和屬性:

方法或屬性	說明
title	傳回網頁標題; 例: sp.title
text	傳回不含 HTML tag 的網頁文字內容
find()	傳回第一個符合條件的標籤(HTML tag); 例: sp.find('tr') 找到後傳回一個字串, 找不到傳回 None
find_all()	傳回所有符合條件的標籤(HTML tag); 例: sp.find_all('a') 找到後傳回一個串列(list), 找不到傳回空串列([])
select()	傳回 CSS 選擇器(id 或 class)或標籤(HTML tag); 例: sp.select('#id'), sp.select('.class'), sp.select('td') 找到後傳回一個串列(list), 找不到傳回空串列([])

(ch1-4_BeautifulSoup1.py)

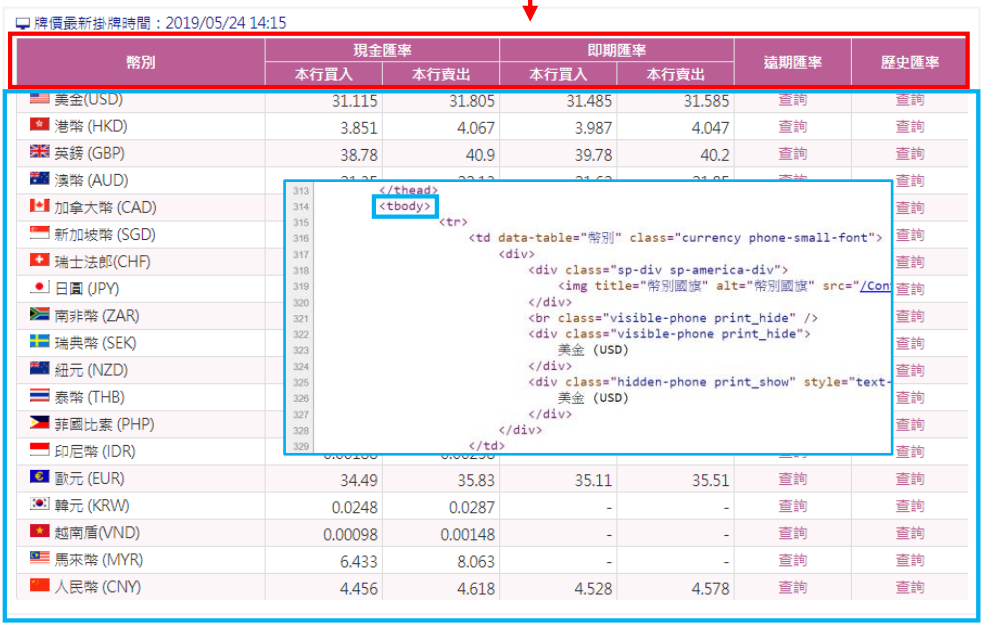
這個案例就來實作抓取 19 種幣別和台幣的匯率:

```
 8 import requests
 9 from bs4 import BeautifulSoup
10 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
11 html = requests.get(url)
12 html.encoding = 'utf-8'
13 sp = BeautifulSoup(html.text, 'html.parser')
14 data1 = sp.find('tbody').find_all('tr')
15 print(len(data1))
16 print(data1)
```

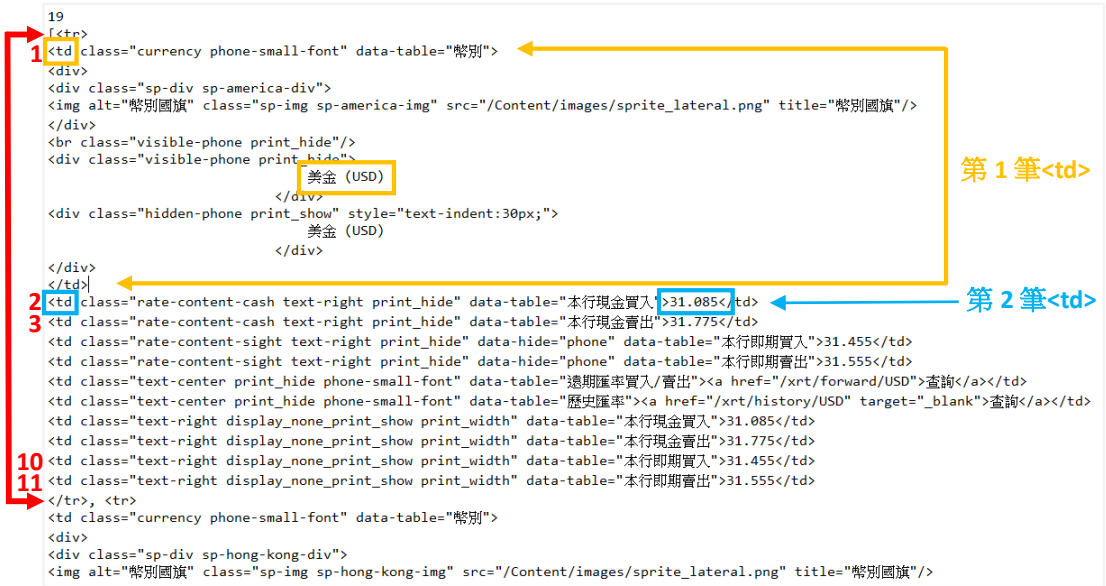
```
19
[<tr>
<td class="currency phone-small-font" data-table="幣別">
<div>
<div class="sp-div sp-america-div">
<img alt="幣別國旗" class="sp-img sp-america-img" src="/Content/images/sprite_lateral.png" title="幣
別國旗"/>
</div>
<br class="visible-phone print_hide"/>
<div class="visible-phone print_hide">
                                美金 (USD)
                            </div>
<div class="hidden-phone print_show" style="text-indent:30px;">
                                美金 (USD)
                            </div>
</div>
</td>
<td class="rate-content-cash text-right print_hide" data-table="本行現金買入">31.085</td>
<td class="rate-content-cash text-right print_hide" data-table="本行現金賣出">31.775</td>
<td class="rate-content-sight text-right print_hide" data-hide="phone" data-table="本行即期買
入">31.455</td>
<td class="rate-content-sight text-right print_hide" data-hide="phone" data-table="本行即期賣
出">31.555</td>
```

列	程式	說明
8	import requests	匯入 requests 套件
9	from bs4 import BeautifulSoup	從 bs4 套件匯入 BeautifulSoup 模組
10	url = 'https://rate.bot.com.tw/	將台灣銀行告牌匯率網頁的網址字串 'https://rate.bot.com.tw/xrt?Lang=zh-TW'

	xrt?Lang=zh-TW'	指派給變數 url
11	html = requests.get(url)	用 requests.get(url)方法向 url 網址所在伺服器發出請求，伺服器接受請求後，傳回 url 網址網頁的原始碼，並指定給變數 html
12	html.encoding = 'utf-8'	設定 utf-8 編碼
13	sp = BeautifulSoup(html.text, 'html.parser')	用 html.parser 解析 requests 取得的原始碼 html.text，傳回 BeautifulSoup 物件，並指給變數 sp；就可以使用 sp BeautifulSoup 物件的方法和屬性解析 requests 取得的原始碼 html.text
	上節已看過網頁原始碼，包括對網頁按右鍵，選擇「檢視網頁原始碼(v)」或用 requests 取得的原始碼. 而現在需研究網頁結構(原始碼內容)，才可用 sp BeautifulSoup 物件的方法和屬性找出資料的 html 標籤或 css 選擇器，取出特定資料. 看下圖網頁分析.	原始碼的第 264 列 <table> 標籤內建構了各幣別的匯率表格 原始碼的第 265 列 <thead> 標籤內建構表格表頭，但我們要的匯率資料不在此 原始碼的第 314 列 <tbody> 標籤內建構表格匯率資料，我們要抓取此處匯率資料



列	程式	說明
14	data1 = sp.find('tbody').find_all('tr')	網頁原始碼再深入分析，可觀察到每一筆幣別匯率都在一<tr>標籤內，因此 1. 使用 BeautifulSoup 的物件 sp 的 find()方法; sp.find('tbody')找到表格內文標籤 <tbody> 2. 用 find_all()方法, sp.find('tbody').find_all('tr')再找到 <tbody>內的所有<tr>標籤，每一<tr>標籤內有一筆幣別匯率資料 3. 將上一步驟找到由<tr>標籤為元素所組成的串列，指定給變數 data1
15	print(len(data1))	輸出串列的長度 19; 即網頁有 19 種幣別匯率
16	print(data1)	輸出串列 data1，繼續分析內容 html 結構
	看下圖是由第 16 列輸出 data1 的結果，紅色箭頭所指為第一筆<tr>資料，即美金匯率資料; 而每一筆<tr>內有 11 筆<td>標籤	而我們要抓取的資料「美金 (USD)」是在第一筆<td>, class="visible-phone print_hide" 的 標籤內 資料「31.085」是在第二筆<td>標籤內



(ch1-4_BeautifulSoup2.py)

將 ch1-4_BeautifulSoup1.py 另存為 ch1-4_BeautifulSoup2.py 繼續用 BeautifulSoup 的物件 sp 解析網頁，取得目標資料

```
 8 import requests
 9 from bs4 import BeautifulSoup
10 url = 'https://rate.bot.com.tw/xrt?Lang=zh-TW'
11 html = requests.get(url)
12 html.encoding = 'utf-8'
13 sp = BeautifulSoup(html.text, 'html.parser')
14 data1 = sp.find('tbody').find_all('tr')
15 #print(len(data1))
16 #print(data1)
17 for data2 in data1:
18     data3 = data2.find_all('td')
19     usd = data3[0].find('div', {'class':'visible-phone print_hide'})
20     usd_text = usd.text.strip()
21     print(usd_text, ':', end='')
22     print(data3[1].text)
```

```
美金 (USD) :31.085
港幣 (HKD) :3.848
英鎊 (GBP) :38.74
澳幣 (AUD) :21.36
加拿大幣 (CAD) :22.93
新加坡幣 (SGD) :22.31
瑞士法郎 (CHF) :30.62
日圓 (JPY) :0.2782
南非幣 (ZAR) :-
瑞典幣 (SEK) :2.9
紐元 (NZD) :20.1
泰幣 (THB) :0.8652
菲國比索 (PHP) :0.5299
印尼幣 (IDR) :0.00188
歐元 (EUR) :34.44
韓元 (KRW) :0.0248
越南盾 (VND) :0.00098
馬來幣 (MYR) :6.429
人民幣 (CNY) :4.457
```

列	程式	說明
15, 16	#print(len(data1)) #print(data1)	將 15, 16 列改為註解 原輸出只是要分析已抓取的資料，並非目標資料
17	for data2 in data1:	以 for 回圈的變數 data2 迭代 data1 串列中的元素, data2 即為 data1 串列中的每一筆<tr>標籤(共 19 筆)
18	data3 = data2.find_all('td')	以之前的分析，每一筆<tr>標籤中有 11 筆<td>標籤，以 data2.find_all('td') 找出所有<td>標籤，傳回的串列指派給變數 data3
19	usd = data3[0].find('div', {'class':'visible-phone print_hide'})	之前已分析過要抓取的幣別資料「美金(USD)」是在第一筆<td> (索引值為 0, 即 data3[0]), class="visible-phone print_hide" 的 標籤內; 以(data3[0].find('div', {'class':'visible-phone print_hide'})的語法表示, 找到 class 為 visible-phone print_hide 的 div. 其中注意{'class':'visible-phone print_hide'}語法說明如下: BeautifulSoup 物件方法(find or find_all)(tag, { 屬性名稱 : 屬性內容 }), 可取得標籤 tag 中符合的屬性. 傳回的字串指定給變數 usd

20	usd_text = usd.text.strip()	再次觀察原始碼如下圖第 323 列要抓取的資料「美金 (USD)」，有換行或空白等字符，因此需去除這些多餘的字符，才不會影響輸出格式. usd.text 取得字串，用 strip()方法去除字串左右多餘的字符，並將結果指定給變數 usd_text 代表各幣別
21	print(usd_text, ':', end='')	輸出變數 usd_text 幣別，加上冒號「:」且不換行(end='')
22	print(data3[1].text)	之前已分析過要抓取的匯率資料「31.085」是在第二筆<td>標籤內(索引值為 1，即 data3[1]), data3[1].text 取得資料輸出

```
314          <tbody>
315              <tr>
316                  <td data-table="幣別" class="currency phone-small-font">
317                      <div>
318                          <div class="sp-div sp-america-div">
319                              <img title="幣別國旗" alt="幣別國旗" src="/Content/images/spr
320                          </div>
321                          <br class="visible-phone print_hide" />
322                          <div class="visible-phone print_hide">
323                              美金 (USD)
324                          </div>
325                          <div class="hidden-phone print_show" style="text-indent:30px;">
326                              美金 (USD)
327                          </div>
328                      </div>
329                  </td>
```

從呈現內容可知有換行和空白

哇! 我們輕鬆就從台灣銀行網頁抓取需要的 19 種幣別的匯率，是不是很酷? 接下來我們要玩更酷的實作喔!

註: 每個網頁的 HTML 結構都不同，因此要有 HTML 網頁標籤語言基礎知識，在爬取網頁資料時才能應付不同的網頁結構. 而學員孩子在學習 Python 程式語言之前，也已學習網站設計. 對於爬取網頁資料就游刃有餘了! 因此對 HTML 網頁標籤語言和 CSS 階層樣式表不熟悉的同學，建議先上 w3c 網站熟悉喔!