webbrowser：是 Python 自帶的，打開瀏覽器獲取指定頁面。

requests：從網際網路上下載檔案和網頁。

Beautiful Soup：解析 HTML，即網頁編寫的格式。

mapIt.py

命令列> python mapIt.py 870 Valencia St, San Francisco, CA 94110

在程式的#!行之後，需要導入 webbrowser 模組，用於載入流覽器；導入 sys 模塊，用於讀入可能的命令列參數。

不希望程式的名稱出現在這個字串中，所以不是使用 sys.argv，而是使用 sys.argv[1:]

如果沒有命令列參數，程式將假定位址保存在剪貼板中。可以用 pyperclip.paste()取得剪貼板的內容，並將它保存在名為 address 的變數中，最後，啟動外部流覽器訪問 Google 地圖的 URL，調用 webbrowser.open()。

用 requests.get()函數下載一個網頁

requests.get()函數接受一個要下載的 URL 字串，通過在 requests.get()的返回值上調用 type()，你可以看到它返回一個 Response 物件

```
>>> import requests
>>> res =
requests.get('http://www.gutenberg.org/cache/epub/1112/pg1112.txt')
>>> type(res)
<class 'requests.models.Response'>
>>> res.status_code == requests.codes.ok
```

True

>>> len(res.text)

178981

>>> print(res.text[:250])

Response 物件有一個 status_code 屬性，可以檢查它是否等於 requests.codes.ok，瞭解下載是否成功。有另一種簡單的”檢查成功”方法，是在 Response物件上調用 raise_for_status()方法。如果下載檔案出錯，這將拋出異常。如果下載成功，則什麼也不做。

```
In [10]: res = requests.get('http://inventwithpython.com/
page_that_does_not_exist')

In [11]: res.raise_for_status()
Traceback (most recent call last):

  File "<ipython-input-11-cd6be6b74546>", line 1, in <module>
    res.raise_for_status()

  File "C:\Users\polaris\Anaconda3\envs\Ruby\lib\site-packages
\requests\models.py", line 935, in raise_for_status
    raise HTTPError(http_error_msg, response=self)

HTTPError: 404 Client Error: Not Found for url: http://
inventwithpython.com/page_that_does_not_exist
```

201901228a.py

```
import requests
res = requests.get('http://inventwithpython.com/page_that_does_not_exist')
try:
    res.raise_for_status()
except Exception as exc:
    print('There was a problem: %s' % (exc))
```

將下載的檔保存到硬碟

可以用標準的 open()函數和 write()方法，將 Web 頁面保存到硬碟中的一個文件。但是，這裡稍稍有一點不同。

首先，必須用“寫二進位”模式打開該檔，即向函數傳入字串'wb'，作為 open()的第二參數。即使該頁面是純文字的（例如前面下載的羅密歐與茱麗葉的文本），也需要寫入二進位資料，而不是文本資料，目的是為了保存該文本中的”Unicode 編碼”。

201901228b.py

下載並保存到檔的完整過程如下：

1．調用 requests.get()下載該文件。

2．用'wb'調用 open()，以寫二進位的方式打開一個新檔。

3．利用 Respose 物件的 iter_content()方法做迴圈。

4．在每次反覆運算中調用 write()，將內容寫入該檔。

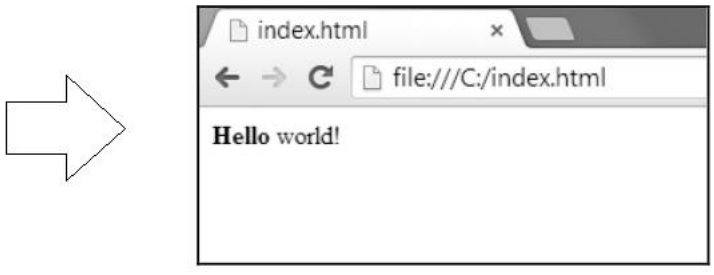
5．調用 close()關閉該文件。

for 迴圈和 iter_content()的部分可能看起來比較複雜，這是為了確保 requests 模組即使在下載巨大的檔時也不會消耗太多記憶體。

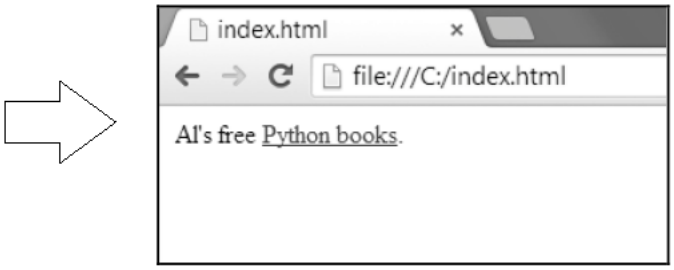
學習 HTML 的資源 https://www.w3schools.com/html/

快速複習

```
<strong>Hello</strong> world!
```



```
Al's free <a href="http://inventwithpython.com">Python books</a>.
```



某些元素具有 id 屬性，可以用來在網頁原始碼上唯一地確定該元素。程式根據元素的 id 屬性來尋找它，所以利用瀏覽器的開發者工具，弄清楚元素的 id 屬性，這是編寫 Web 抓取程式常見的任務。

在 Windows 版的 Chrome 和 IE 中，開發者工具已經安裝了。可以按下 F12，讓它們出現。再次按下 F12，可以讓開發者工具消失。

使用開發者工具來尋找 HTML 元素

需要弄清楚，這段 HTML 的哪個部分對應於網頁上你感興趣的資訊。

這就是可以利用開發者工具的地方。假定需要編寫一個程式，從 http://weather.gov/ 獲取天氣預報資料。在寫程式之前，先做一點調查。如果訪問該網站，並查找郵遞區號 94105，該網站將打開一個頁面，顯示該地區的天氣預報。

如果想抓取那個郵遞區號對應的氣溫資訊，怎麼辦？右鍵點擊它在頁面的位置（或在 OS X 上用 Control-點擊），在彈出的功能表中選擇 Inspect Element。這將打開開發者工具視窗，其中顯示產生這部分網頁的 HTML。

看起來氣溫資訊包含在一個<p>元素中，帶有 myforecast-current-lrg 的類。之後，BeautifulSoup 模組可以說明如何在這個字串中找到它。

BeautifulSoup 模組的名稱是 bs4（表示 Beautiful Soup，第 4 版）。要安裝它，需要在命令列中運行 pip install beautifulsoup4

雖然安裝時使用的名字是 beautifulsoup4，但要導入它，則使用 import bs4。

透過 select()方法尋找元素

針對要尋找的元素，調用 method()方法，傳入一個字串作為 CSS ”選擇器”，即可以取得 Web 頁面元素。

表 11-2 CSS 选择器的例子

传递给 select()方法的选择器	将匹配…
soup.select('div')	所有名为 的元素
soup.select('#author')	带有 id 属性为 author 的元素
soup.select('.notice')	所有使用 CSS class 属性名为 notice 的元素
soup.select('div span')	所有在 元素之内的元素
soup.select('div > span')	所有直接在 元素之内的元素，中间没有其他元素
soup.select('input[name]')	所有名为<input>，并有一个 name 属性，其值无所谓的元素
soup.select('input[type="button"]')	所有名为<input>，并有一个 type 属性，其值为 button 的元素

不同的選擇器模式可以組合起來，形成複雜的匹配。例如，soup.select('p #author') 將匹配所有 id 屬性為 author 的元素，只要它也在一個<p>元素之內。

```
In [3]: import bs4

In [4]: exampleFile = open('example.html')

In [5]: exampleSoup = bs4.BeautifulSoup(exampleFile.read())
C:\Users\polaris\Anaconda3\envs\Ruby\lib\site-packages
\bs4\__init__.py:181: UserWarning: No parser was explicitly
specified, so I'm using the best available HTML parser for this
system ("lxml"). This usually isn't a problem, but if you run this
code on another system, or in a different virtual environment, it may
use a different parser and behave differently.

The code that caused this warning is on line 269 of the file C:\Users
\polaris\Anaconda3\envs\Ruby\lib\site-packages\spyder\utils\ipython
\start_kernel.py. To get rid of this warning, change code that looks
like this:

 BeautifulSoup(YOUR_MARKUP})

to this:

 BeautifulSoup(YOUR_MARKUP, "lxml")

  markup_type=markup_type))
```

```
In [5]: exampleSoup = bs4.BeautifulSoup(exampleFile.read(), "lxml")
```

```
In [6]: elems = exampleSoup.select('#author')

In [7]: type(elems)
Out[7]: list

In [8]: len(elems)
Out[8]: 1

In [9]: type(elems[0])
Out[9]: bs4.element.Tag

In [10]: elems[0].getText()
Out[10]: 'Al Sweigart'

In [11]: str(elems[0])
Out[11]: '<span id="author">Al Sweigart</span>'

In [12]: elems[0]
Out[12]: <span id="author">Al Sweigart</span>

In [13]: elems[0].attrs
Out[13]: {'id': 'author'}

In [14]: type(elems[0].attrs)
Out[14]: dict
```

上述程式碼將帶有 id="author"的元素，從示例 HTML 中找出來。使用 select('#author')返回一個列表，其中包含所有帶有 id="author"的元素。我們將這個Tag 物件的清單保存在elems變數中，len(elems)得知清單中只有一個 Tag 物件，只有一次匹

配。在該元素上調用 getText()方法，返回該元素的文本，或內部的 HTML。一個元素的文本是在開始和結束標籤之間的內容：在這個例子中，就是'Al Sweigart'。 將該元素傳遞給 str()，這將返回一個字串，其中包含開始和結束標籤，以及該元素的文本。最後，attrs 給了我們一個字典，包含該元素的屬性'id'，以及 id 屬性的值'author'。

也可以從 BeautifulSoup 物件中找出<p>元素。在互動式環境中輸入以下程式碼：

```
<!-- This is the example.html file. -->

<html><head><title>The Website Title</title></head>
<body>
<p>Download my <strong>Python</strong> book from <a href="http://inventwithpython.com">my website</a>.</p>
<p class="slogan">Learn Python the easy way!</p>
<p>By <span id="author">Al Sweigart</span></p>
</body></html>
```

2 1 0

```
>>> pElems = exampleSoup.select('p')
>>> str(pElems[0])
'<p>Download my <strong>Python</strong> book from <a href="http://
inventwithpython.com">my website</a>.</p>'
>>> pElems[0].getText()
'Download my Python book from my website.'
>>> str(pElems[1])
'<p class="slogan">Learn Python the easy way!</p>'
>>> pElems[1].getText()
'Learn Python the easy way!'
>>> str(pElems[2])
'<p>By <span id="author">Al Sweigart</span></p>'
>>> pElems[2].getText()
'By Al Sweigart'
```

結果顯示select()給一個列表 包含 3 次匹配 我們將它保存在 pElems 中。在 pElems[0]、pElems[1]和 pElems[2]上使用 str()，將每個元素顯示為一個字串，並在每個元素上使用 getText()，顯示它的文本。

透過元素的屬性獲取資料

Tag 物件的 get()方法讓我們很容易從元素中獲取屬性值。向該方法傳入一個屬性名稱的字串，它將返回該屬性的值。

```
>>> import bs4
>>> soup = bs4.BeautifulSoup(open('example.html'))
>>> spanElem = soup.select('span')[0]
>>> str(spanElem)
'<span id="author">Al Sweigart</span>'
>>> spanElem.get('id')
'author'
>>> spanElem.get('some_nonexistent_addr') == None
True
>>> spanElem.attrs
{'id': 'author'}
```

這裡使用 select()來尋找所有元素，然後將第一個匹配的元素保存在 spanElem 中，get('id')，返回屬性名'id'的屬性值'author'。

實作: 20191228bb.py

實作: 用Google 搜尋 “Beautiful Soup” 並自動開啟瀏覽器連結搜尋結果的至多前五筆網頁

lucky.py

執行畫面:

```
\雲端運算程式開發>python lucky.py Beautiful Soup
```

- 從命令列參數中獲取查詢關鍵字。→從 sys.argv 中讀取命令列參數。
- 取得查詢結果頁面。 →用 requests 模組取得查詢結果頁面。
- 為每個結果打開一個瀏覽器分頁。 →找到每個查詢結果的連結。

  & 調用 webbrowser.open()打開瀏覽器。

```
In [20]: linkElems[2]
Out[20]: <a href="/url?q=https://ithelp.ithome.com.tw/articles/
10196817&amp;sa=U&amp;ved=0ahUKEwjv-
bXK5LHhAhXGyLwKHZXJCFUQFggiMAI&amp;usg=AOvVaw1TzbqsojHZI8oPLrX8aV3e">
[Day23]<b>Beautiful Soup</b>網頁解析！ - iT 邦幫忙::一起幫忙解決難題
...</a>

In [21]: linkElems[1]
Out[21]: <a href="/url?q=https://www.crummy.com/software/
BeautifulSoup/bs4/doc/&amp;sa=U&amp;ved=0ahUKEwjv-
bXK5LHhAhXGyLwKHZXJCFUQFggZMAE&amp;usg=AOvVaw3fojGb2ELZNgZ4zli_-4L_">
<b>Beautiful Soup</b> Documentation — <b>Beautiful Soup</b> 4.4.0
documentation</a>
```

其它範例 lucky1.py