



# Pandas & 爬蟲

賴璉錡

lclai.t11@o365.fcu.edu.tw

# Pandas

- Python的Pandas庫是一個功能強大的數據處理工具，可以讓使用者輕鬆地讀取、處理、操作和分析數據。
- Pandas庫提供了兩種核心的數據結構：
  - Series：類似於一維數組或列表的數據結構，每個元素都有一個標籤或索引。可以通過索引尋找和訪問數據。
  - DataFrame：類似於二維表格或關聯數據庫表的數據結構，由多個Series組成，每個Series都有一個列名稱，並且可以進行行和列的索引。

# 建立DataFrame

- 從列表創建DataFrame

```
import pandas as pd

data = [['John', 25], ['Jane', 30], ['Alice', 35]]
df = pd.DataFrame(data, columns=['Name', 'Age'])
print(df)
```

	Name	Age
0	John	25
1	Jane	30
2	Alice	35

- 從字典創建DataFrame

```
import pandas as pd

data = {'Name': ['John', 'Jane', 'Alice'], 'Age': [25, 30, 35]}
df = pd.DataFrame(data)
print(df)
```

	Name	Age
0	John	25
1	Jane	30
2	Alice	35

# 建立DataFrame

- 從CSV創建DataFrame

```
name,age  
John, 25  
Jane, 30  
Alice, 35
```

	A	B
1	name	age
2	John	25
3	Jane	30
4	Alice	35

```
import pandas as pd  
  
df = pd.read_csv('info.csv')  
print(df)
```

```
   name  age  
0  John   25  
1  Jane   30  
2  Alice  35
```

# DataFrame操作

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

# 選擇一列
col = df['Name']
print(col)

# 選擇多列
cols = df[['Name', 'Math']]
print(cols)
```

	Name	Math	English	Science
0	John	80	85	90
1	Jane	85	90	92
2	Alice	90	92	88
3	Bob	75	80	85
4	Tom	88	85	89
5	Jerry	92	90	85
6	Mary	90	92	88
7	Mike	85	80	86
8	Lucy	88	85	89
9	David	80	75	70
10	Lisa	85	80	82
11	Peter	90	92	88
12	Sophie	75	80	85
13	Ben	88	85	89
14	Maggie	55	60	45
15	Helen	90	92	88
16	Tim	50	60	65
17	Kelly	60	55	55
18	Frank	80	75	70
19	Joyce	45	50	60

# DataFrame操作

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

# 選擇一列
col = df['Name']
print(col)

# 選擇多列
cols = df[['Name', 'Math']]
print(cols)
```

```
0      John
1      Jane
2      Alice
3       Bob
4       Tom
5      Jerry
6      Mary
7      Mike
8      Lucy
9      David
10     Lisa
11     Peter
12    Sophie
13       Ben
14    Maggie
15     Helen
16      Tim
17     Kelly
18     Frank
19     Joyce
Name: Name, dtype: object
```

# DataFrame操作

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

# 選擇一列
col = df['Name']
print(col)

# 選擇多列
cols = df[['Name', 'Math']]
print(cols)
```

	Name	Math
0	John	80
1	Jane	85
2	Alice	90
3	Bob	75
4	Tom	88
5	Jerry	92
6	Mary	90
7	Mike	85
8	Lucy	88
9	David	80
10	Lisa	85
11	Peter	90
12	Sophie	75
13	Ben	88
14	Maggie	55
15	Helen	90
16	Tim	50
17	Kelly	60
18	Frank	80
19	Joyce	45

# DataFrame操作

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

# 選擇第一行
row = df.loc[0]
print(row)

# 選擇前3行
rows = df.loc[:2]
print(rows)
```

```
Name      John
Math       80
English    85
Science    90
Name: 0, dtype: object
```



# DataFrame操作

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

# 選擇第一行
row = df.loc[0]
print(row)

# 選擇前3行
rows = df.loc[:2]
print(rows)
```

	Name	Math	English	Science
0	John	80	85	90
1	Jane	85	90	92
2	Alice	90	92	88

# DataFrame 遞迴

- 使用 `iterrows()` 方法遍歷 DataFrame 的每一行

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

for index, row in df.iterrows():
    print(index, row)
```

```
0 Name      John
  Math      80
  English   85
  Science   90
Name: 0, dtype: object
1 Name      Jane
  Math      85
  English   90
  Science   92
Name: 1, dtype: object
2 Name      Alice
  Math      90
  English   92
  Science   88
Name: 2, dtype: object
3 Name      Bob
  Math      75
  English   80
  Science   85
Name: 3, dtype: object
```

# DataFrame 遞迴

- 使用 `iterrows()` 方法遍歷 DataFrame 的每一行

```
import pandas as pd

df = pd.read_csv('grade.csv')
print(df)

for index, row in df.iterrows():
    print(index, row['Name'], row['Math'])
```

```
0 John 80
1 Jane 85
2 Alice 90
3 Bob 75
4 Tom 88
5 Jerry 92
6 Mary 90
7 Mike 85
8 Lucy 88
9 David 80
10 Lisa 85
11 Peter 90
12 Sophie 75
13 Ben 88
14 Maggie 55
15 Helen 90
16 Tim 50
17 Kelly 60
18 Frank 80
19 Joyce 45
```

## Practice - DataFrame

- grade.csv
- 從grade.csv建立  
DataFrame
  - 1.取得Name, English的  
欄位資料
  - 2.計算Science的平均

	Name	English
0	John	85
1	Jane	90
2	Alice	92
3	Bob	80
4	Tom	85
5	Jerry	90
6	Mary	92
7	Mike	80
8	Lucy	85
9	David	75
10	Lisa	80
11	Peter	92
12	Sophie	80
13	Ben	85
14	Maggie	60
15	Helen	92
16	Tim	60
17	Kelly	55
18	Frank	75
19	Joyce	50

Science average is 79.45

# Pandas if else 條件式 (conditions)

- 在 pandas 中若我們要進行條件的篩選時，通常會直接對 DataFrame 設定篩選條件。

```
import pandas as pd

df = pd.read_csv("grade.csv")
tom = df[df["Name"] == "Tom"]

print(tom)
```

	Name	Math	English	Science
4	Tom	88	85	89

```
df = pd.read_csv("grade.csv")
eng = df[df["English"] < 80]

print(eng)
```

	Name	Math	English	Science
9	David	80	75	70
14	Maggie	55	60	45
16	Tim	50	60	65
17	Kelly	60	55	55
18	Frank	80	75	70
19	Joyce	45	50	60

# 儲存資料

- 透過 `df.to_csv()` 的方式可以將資料存成 `csv` 檔案。
  - `to_excel()` 方法可以存成 `excel` 的檔案(需安裝 `openpyxl` 套件)

```
import pandas as pd

data = {
    'name': ['John', 'Anna', 'Peter', 'Linda'],
    'location': ['New York', 'Paris', 'Berlin', 'London'],
    'age': [24, 13, 53, 33]
}

data_pandas = pd.DataFrame(data)
data_pandas.to_csv('data.csv')

data_pandas.to_excel('data.xlsx')
```

# HTML

# HTML(HyperText Markup Language)

- 是一種用於建立網頁的標準標記語言。
- 網頁是由一系列的標籤 (標記) 構成，用於描述網頁內容 (如文字、圖片、影片) 的結構和呈現方式。
- 瀏覽器會解析HTML的程式碼，並將其轉換成一般看到的網頁。
- HTML 標籤以尖括號 (<和>) 包圍，通常成對出現，用於表示標籤的開始和結束。
  - <p>這是一段文字</p>
  - <p> 是段落標籤的開始，</p> 是段落標籤的結束。標籤之間的內容是網頁上呈現的實際內容。



# 觀察網頁頁面

- 開啟逢甲大學首頁
- 畫面空白處按右鍵並選取“檢視網頁原始碼”

```
436 <div class="m-header-tour__item a-lang-switcher" >
437   <div class="a-lang-switcher__display">
438                                     <a class="a-lang-switcher__link" href="https://www.fcu.edu.tw/en/">English</a>
439   </div>
440 </div>
441   <a href="https://www.fcu.edu.tw/?s=" class="m-header-tour__item m-header-tour__search">
442     <icon svg-class="icon" name="search"></icon>
443   </a>
444 </div>
445 <div class="o-header__main">
446   <nav class="o-header__nav" ref="header">
447     <a class="o-header__logo pic" href="https://www.fcu.edu.tw/">
448       
449     </a>
450     <div class="o-header__nav-content">
451       <ul class="o-header__list" v-if="isDesktop">
452         <li>
453           <a
454             class="o-header__item "
455             data-post-id="2"
456             data-link-id="70"
457             data-first="f"
458             data-second="f"
459             data-result=""
460             target=""
461             href="https://www.fcu.edu.tw/apply/"
462           >招生</a>
463         </li>
464       </ul>
465     </div>
466   </nav>
467 </div>
```

# 常見HTML標籤

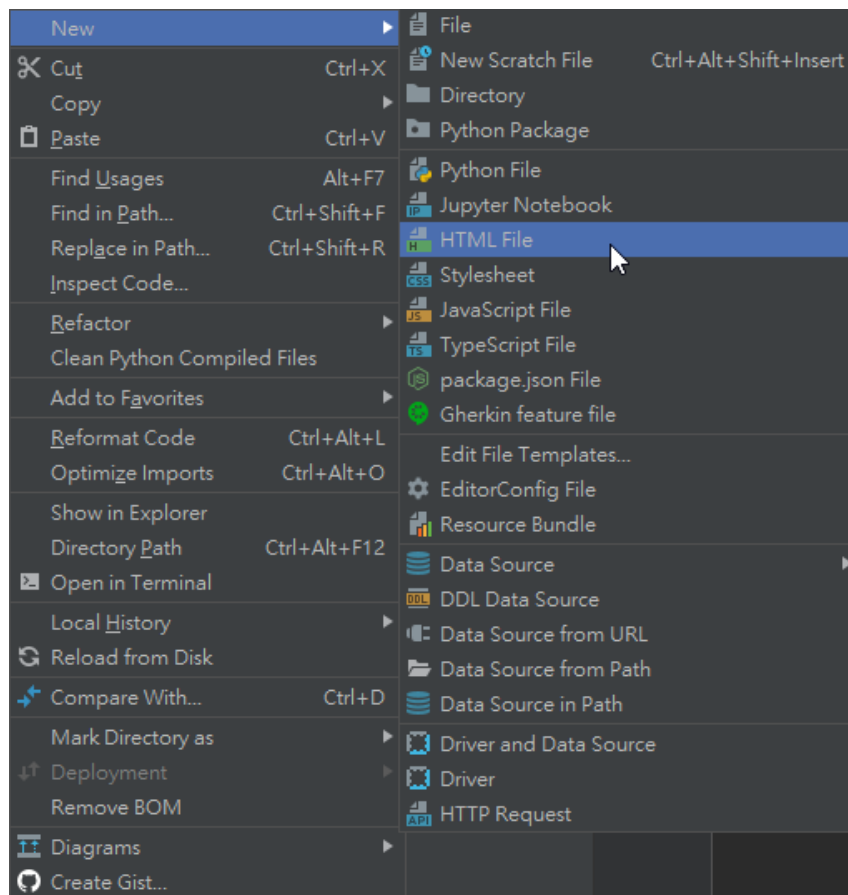
標籤	用途
<html>	標記 HTML 文件的開始和結束。
<head>	包含網頁的元數據，如標題、字符集和引入的外部檔案（CSS、JavaScript 等）。
<body>	包含網頁的主要內容，如文本、圖片和連結等。
<div>	區塊級元素，它是 "division"（劃分、區段）的縮寫。沒有特定的語義
<h1> ~ <h6>	標記網頁的標題，從 <h1>（最大）到 <h6>（最小）。
<p>	標記段落。
<a>	標記超連結，通常使用 href 屬性指定連結目標。
<img>	標記圖片，使用 src 屬性指定圖片的來源，alt 屬性為替代文本。
<ul> 和 <li>	標記無序列表（<ul>）和列表項目（<li>）。
<ol> 和 <li>	標記有序列表（<ol>）和列表項目（<li>）。
<table>、<tr>、<th> 和 <td>	標記表格（<table>）、表格行（<tr>）、表格標題（<th>）和表格數據（<td>）。

# HTML tags 屬性

- 在HTML中，每個標籤 (tag) 都可以賦予他們一些屬性，這些屬性可以用來分組或更改他們的樣式等。
- 常見屬性：
  - id: 每個標籤可以賦予一個id，javascript或css就可以透過id來指定這個標籤元素，須注意在同個頁面id不得重複。
  - class: 作用與id相似，但是可以將相同類別的元素設定為同個class名稱，也可以宣告多個class來疊加不同的樣式。
  - style: 可以用css語法來改變該元素的外觀。

# Practice: HTML

- 與新增py檔相同，新增一個HTML檔案



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
</body>
</html>
```

# Practice: HTML

- 打開專案資料夾
- 用瀏覽器開啟新增的HTML檔案
- 回到PyCharm編輯器
- 更改<title>標籤內的內容
- 回到瀏覽器按下F5或重新整理後觀察分頁名稱

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>abc</title>
</head>
<body>
</body>
</html>
```

abc

# Practice: HTML標籤練習

- 在<body>區塊中新增標籤，並觀察其作用

```
<h1>h1標題</h1>
<h2>h2標題</h2>
...
<h5>h5標題</h5>
<h6>h6標題</h6>
```

## h1標題

## h2標題

...

## h5標題

## h6標題

這是一段文字

```
<p>這是一段文字</p>
<a href="https://fcu.edu.tw">逢甲大學</a>

```



[逢甲大學](https://fcu.edu.tw)

```
<ul>
  <li>蘋果</li>
  <li>香蕉</li>
  <li>橘子</li>
</ul>
<ol>
  <li>第一項</li>
  <li>第二項</li>
  <li>第三項</li>
</ol>
```

- 蘋果
  - 香蕉
  - 橘子
- 第一項
  - 第二項
  - 第三項

```
<table>
  <tr>
    <th>股票名稱</th>
    <th>股價</th>
    <th>漲跌幅</th>
  </tr>
  <tr>
    <td>漢翔</td>
    <td>57.2</td>
    <td>2.33%</td>
  </tr>
  <tr>
    <td>華航</td>
    <td>18.85</td>
    <td>1.89%</td>
  </tr>
</table>
```

股票名稱	股價	漲跌幅
漢翔	57.2	2.33%
華航	18.85	1.89%

# 網頁結構

- HTML標籤內包含整個網頁的內容，通常分為以下3個部分：
  - `<head>`: 包含網頁的元數據，如標題、字符集和引入的外部檔案（CSS、JavaScript 等）。
  - `<body>`: 網頁中主要的內容區塊
  - `<footer>`: 放在網頁中最底部的區塊，通常包含版權、聯繫方式、網站地圖或其他一般性的訊息。
- `<body>`中會使用`<div>`來將頁面中的內容分為不同區塊。

# Practice: 觀察網站

- 開啟 ilearn
- 按下 F12 開啟開發者模式

```
<!DOCTYPE html>
<html dir="ltr" lang="zh-tw" xml:lang="zh-tw" class="no-js yui
i3-js-enabled">
  <head> </head>
  <body id="page-site-index" class="format-site course path-s
ite safari dir-ltr lang-zh_tw yui-skin-sam yui3-skin-sam il
earn2-fcu-edu-tw pagelayout-frontpage course-1 context-2 lo
ggedin desktopdevice pagewidthnormal custommenuitems hasbor
inglayout floatingsubmit has-region-side-pre used-region-si
de-pre has-region-footer-left empty-region-footer-left has-
region-footer-middle empty-region-footer-middle has-region-
footer-right empty-region-footer-right has-region-hidden-do
ck used-region-hidden-dock has-region-page-top empty-region
-page-top has-region-header empty-region-header jsenabled">
  </body> == $0
</html>
<!-- Essential theme version: 2018051908 is developed by
Gareth J Barnard: about.me/gjbarnard -->
```



# CSS 選擇器 Selector

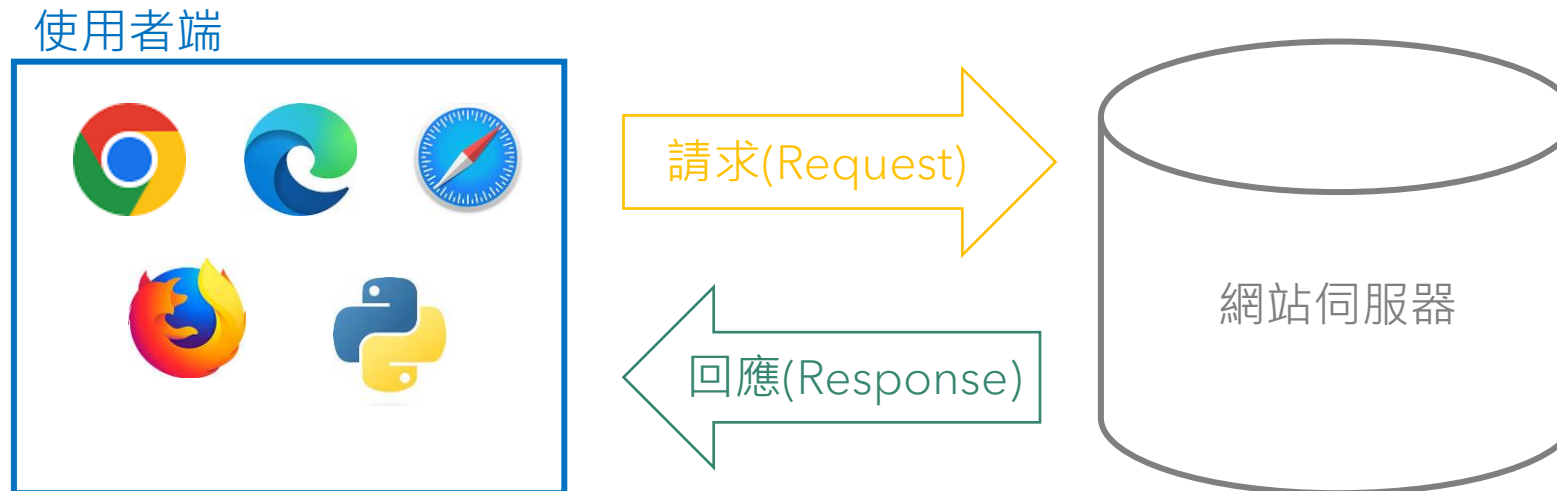
為了獲取特定元素，我們會使用選擇器 (selector)，選擇器挑選出來的元素可能有一個或多個。

- Example:
  - `<p class='comment' id='cmt_1'>實用又好看</p>`
  - 透過標籤選擇: `p`
  - 透過id選擇: `#cmt_1`
  - 透過class選擇: `.comment`
- 練習遊戲: [CSS Diner](#)

# 爬蟲

# 爬蟲

- 網頁爬蟲能依特定模式或規則自動擷取網頁程式碼以取得資料。
- 不只能夠透過Python來爬蟲，如Java、PHP等程式語言皆能進行網頁爬蟲的程式開發。
- 網頁爬蟲實際上就是模擬使用者在瀏覽網頁。



# 請求方法(Request Method)

- http的請求基本上分為 4 種：
  - GET(查詢)、POST(新增)、PUT(修改)及DELETE(刪除)
- 大部分爬取資料都是向伺服器要求資料，所以使用GET居多。
  - requests.get()會返回一個Response的物件
  - 這個Response的物件有個屬性text，這個text是這個網頁的html檔案。

```
import requests

response = requests.get("https://ilearn2.fcu.edu.tw/")
print(type(response))
print(response.text)
```

```
<class 'requests.models.Response'>
```

```
<!DOCTYPE html>
<html dir="ltr" lang="zh-tw" xml:lang="zh-tw" class="no-js">
<head>
  <title>網路教室 iLearn</title>
  <link rel="shortcut icon" href="https://ilearn2.fcu.edu.tw/theme/image
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="keywords" content="moodle, 網路教室 iLearn" />
<link rel="stylesheet" type="text/css" href="https://ilearn2.fcu.edu.tw/th
<script type="text/javascript">
//<![CDATA[
var M = {}; M.yui = {};
```

## Practice: 網頁擷取

- 請用`requests.get`的方法取得以下網站的html檔案內容
  - <https://ilearn2.fcu.edu.tw/>
  - <https://www.fcu.edu.tw/>
  - <https://tw.stock.yahoo.com/>

```
<!DOCTYPE html>
<html class="no-js no-svg -lang-zh" lang="zh-TW">
  <head>
    <meta charset="UTF-8" />
    <meta name="description" content="逢甲大學以學生為本，以
    <title>逢甲大學</title>
    <meta http-equiv="Content-Type" content="text/html;
```

```
<!DOCTYPE html>
<html dir="ltr" lang="zh-tw" xml:lang="zh-tw" cla
<head>
  <title>網路教室 iLearn</title>
  <link rel="shortcut icon" href="https://ilearn
  <meta http-equiv="Content-Type" content="text/
```

```
<!DOCTYPE html><html id="atomic" class="NoJs desktop" lang="zh
.mark && window.performance.mark('PageStart');</script><meta
content="Yahoo奇摩股市提供國內外財經新聞，台股、期貨、選擇權、國際指數、
property="og:image" content="https://s.yimg.com/cv/apiv2/soci
property="og:url" content="https://tw.stock.yahoo.com/" /><me
content="Yahoo奇摩股市提供國內外財經新聞，台股、期貨、選擇權、國際指數、
```

# BeautifulSoup

- BeautifulSoup是一個用來**解析HTML結構**的Python套件 (Package) ，將取回的網頁HTML結構，透過其提供的方法 (Method) ，能夠輕鬆的搜尋及擷取網頁上所需的資料，因此廣泛的應用在網頁爬蟲的開發上。
- response.text的型態是**字串**，所以必須透過BeautifulSoup來轉換成**可以擷取html節點資料的型態**。

```
import requests
from bs4 import BeautifulSoup

response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
print(fcu_undergraduate.prettify()) # 輸出排版後的HTML內容
```

# BeautifulSoup 以HTML標籤及屬性搜尋節點

- `find()` : 只搜尋**第一個**符合條件的html節點，傳入要搜尋的標籤名稱。

```
import requests
from bs4 import BeautifulSoup

response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
fcu_title = fcu_undergraduate.find("title")
print(fcu_title)
```

```
<title>學士班 | 逢甲大學</title>
```

# BeautifulSoup 以HTML標籤及屬性搜尋節點

- `find_all()` : 搜尋網頁中所有符合條件的HTML節點，傳入要搜尋的HTML標籤名稱，可輸入多個標籤。

```
import requests
from bs4 import BeautifulSoup

response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
fcu_a = fcu_undergraduate.find_all("a")
print(fcu_a)
```

```
[<a class="m-header-tour__item m-header-tour__link -no-border" href="#">
</a>, <a class="o-header__logo pic" href="https://www.fcu.edu.tw/">

</a>, <a class="o-header__item" data-first="f" data-link-id="70" href="#">
    招生總覽
  </a>, <a class="a-nav-pop__text" href="https://www.fcu.edu.tw/undergraduate/apply/">
    本國生招生
  </a>, <a class="a-nav-pop__text" href="https://www.fcu.edu.tw/undergraduate/apply/">
    推廣教育
  </a>, <a class="a-nav-pop__text" href="https://www.fcu.edu.tw/undergraduate/apply/">
    海外生招生
  </a>]
```



# BeautifulSoup 以HTML標籤及屬性搜尋節點

- `select()` : 可以透過 **class** 的名稱來搜尋所有節點。

纖維與複合材料學系

```
list"> flex
::before
<p class="a-default-list__title">纖維與複合材料學系</p> = $0
<button class="a-default-list__btn">...</button> flex
::after
</a>
...
```

```
response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
deps = fcu_undergraduate.select(".a-default-list__title")
print(deps)
```

```
[<p class="a-default-list__title">纖維與複合材料學系</p>, <p class="a-default-list__title">機械與電腦輔助工程學系</p>,
```

# 取得節點資料

- <節點物件>.getText() : 取得節點文字內容。

```
response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
deps = fcu_undergraduate.select(".a-default-list__title")

for d in deps:
    print(d.getText())
```

纖維與複合材料學系  
機械與電腦輔助工程學系  
工業工程與系統管理學系  
化學工程學系

- <節點物件>.get("<要取得的屬性>") : 取得節點屬性值。

```
response = requests.get('https://www.fcu.edu.tw/undergraduate/')
fcu_undergraduate = BeautifulSoup(response.text, "html.parser")
deps = fcu_undergraduate.select(".a-default-list__title")

for d in deps:
    print(d.get("class"))
```

```
['a-default-list__title']
['a-default-list__title']
['a-default-list__title']
['a-default-list__title']
```

# Practice: ptt 股票版聊天資料

從ptt網站取得資料，並取得所有推文資料，然後利用 pandas 存成 csv。

[閒聊] 2024/11/20 盤中間聊 - 看板 Stock - 批踢踢實業坊

→ zzzzzzzzzzy:	岸花早小恩早阿雯早阿文早大家好	11/20 08:31
→ River79835 :	好久沒看到長榮試挫漲停	11/20 08:31
→ dennistao :	好的空一口	11/20 08:31
推 intointo :	大家安安 彈不彈?	11/20 08:31
推 swat11239 :	沒人開 丸子 人走茶涼 995	11/20 08:31
推 phoenixcx :	早 賺錢	11/20 08:31
推 a069275235 :	馬來大你遲到!	11/20 08:31
推 s155260 :	多蛙邊吃布丁邊數錢	11/20 08:31
推 ethan0419 :	開嘎	11/20 08:31
噓 a83006 :	99公公窩套在山頂219洗碗風好大	11/20 08:32
推 yillusionwei:	早安 美超微好噴	11/20 08:32
推 Kobe5210 :	颶風再起	11/20 08:32
推 maxman77 :	阿榮!	11/20 08:32
推 c1951 :	<a href="https://reurl.cc/eGx30W">https://reurl.cc/eGx30W</a> 股市高點歐印 賺更多啊?	11/20 08:32
→ wings0610 :	99大象啊啊啊啊啊 一人買一象明年生小象	11/20 08:32
推 jeff0025 :	丸子	11/20 08:33
推 gto9987 :	等等開盤應該直接漲100點以上	11/20 08:33
推 eeeee118 :	今天神達繼續噴噴嗎	11/20 08:33
推 CloudyWing :	台指期紅，微台指綠?	11/20 08:33

# 透過 **API** 取得資料

- API (應用程式介面, `Application Programming Interface`)
  - 告訴伺服器你要什麼資料，遵循他的規則就會回傳你需要的 `json` 格式資料。

# 透過 API 取得資料

- 使用 requests 透過 api 取得資料，資料的text一樣是字串型態。
- 可以透過 json 模組的 loads() 方法來將字串轉換成 Python 的 dict 型態。


## 臺中市111年11月份十大易肇事路段(口)

臺中市111年11月份十大高肇事路口交通事故資料檔案

### 資料與資源

↓ 一鍵下載



臺中市111年11月份10大易肇事路口統計表   
臺中市111年11月份10大易肇事路口統計表.XML

↪ 探索



臺中市111年11月份10大易肇事路口統計表  
臺中市111年11月份10大易肇事路口統計表.JSON

↪ 探索



臺中市111年11月份10大易肇事路口統計表  
臺中市111年11月份10大易肇事路口統計表.CSV

↪ 探索

111年

交通事故

高肇事路口

# 透過 **API** 取得資料

```
import requests, json

url="https://datacenter.taichung.gov.tw/swagger/OpenData/1e510da0-a5e8-4577-9797-fc1c7e9e4734"
response = requests.get(url)
print(type(response.text))
print(response.text)
traffic_json = json.loads(response.text)
print(type(traffic_json))
print(traffic_json)
```

```
<class 'str'>
[{"編號": "1" , "轄區分局": "第四分局" , "路口名稱": "(南屯區)五權西路與環中路口"}]
<class 'list'>
[{'編號': '1', '轄區分局': '第四分局', '路口名稱': '(南屯區)五權西路與環中路口',
```

# 透過 API 取得資料

- 將 dict 資料轉成 Dataframe 格式

```
import requests, json
import pandas as pd

url="https://datacenter.taichung.gov.tw/swagger/OpenData/1e510da0-a5e8-4577-9797-fc1c7e9e4734"
response = requests.get(url)
traffic_json = json.loads(response.text)
print(traffic_json)

df = pd.DataFrame(traffic_json)
print(df)
```

	編號	轄區分局	路口名稱	A1	A2件數	A2受傷	A3	總件數	發生時間	主要肇因
0	1	第四分局	(南屯區)五權西路與環中路口	0	5	6	17	22	16-18	未注意車前狀態
1	2	第六分局	(西屯區)環中路與市政路口	0	3	3	13	16	14-16	未注意車前狀態
2	3	第六分局	(西屯區)文心路與臺灣大道口	0	7	11	8	15	14-16	未注意車前狀態
3	4	第六分局	(西屯區)臺灣大道與安和路口	0	7	11	8	15	08-10	未注意車前狀態
4	5	第六分局	(西屯區)惠來路與臺灣大道口	0	9	11	3	12	08-10	未注意車前狀態
5	6	第二分局	(北區)崇德路與三民路口	0	9	10	3	12	10-12	違反特定標誌(線)禁制
6	7	霧峰分局	(大里區)德芳南路與環中東路口	0	4	4	8	12	06-08	變換車道或方向不當
7	8	第六分局	(西屯區)臺灣大道與惠中路口	0	4	7	7	11	10-12	未注意車前狀態
8	9	第六分局	(西屯區)逢甲路與福星路口	0	5	6	5	10	20-22	未保持行車安全距離
9	10	第六分局	(西屯區)西屯路與環中路口	0	5	7	5	10	06-08	未注意車前狀態

# 將資料整理並匯出

- 目標：將臺中市111年11月份十大易肇事路段(口)的資料儲存成excel的格式，並依據不同的分局存成不同的工作簿。
- Step1: 要先知道有哪些分局，可以用 `df['轄區分局'].unique()` 的方法得知有哪些分局。
- Step2: 透過for迴圈找出dataframe中屬於這些分局的資料，並且匯出到不同工作簿。

```
excel_writer = pd.ExcelWriter('臺中市111年11月份十大易肇事路段.xlsx')
for station in stations:
    station_df = df[df['轄區分局'] == station]
    station_df.to_excel(excel_writer, sheet_name=station)

excel_writer.close()
```



# HW - 112年11月交通事故資料

- <https://opendata.taichung.gov.tw/>
- 請透過 api 取得112年11月台中市交通事故資料，並且依照區  
的欄位資料分別儲存至不同的工作簿
  - <https://opendata.taichung.gov.tw/search/c8c10867-0ff6-4adb-b8af-bc6ea19357bb>

	A	B	C	D	E	F	G	H	I	J	K	L	M	
1		年	月	日	時	分	縣市	區	死	受傷	2-30	天候	光線	道路
2	0	2022	11	1	0	27	臺中市	西屯區	0	0	0	8		
3	14	2022	11	1	6	51	臺中市	西屯區	0	1	0	8	1	5
4	22	2022	11	1	7	24	臺中市	西屯區	0	0	0	8		
5	25	2022	11	1	7	34	臺中市	西屯區	0	1	0	8	1	5
6	30	2022	11	1	7	43	臺中市	西屯區	0	0	0	8		
7	39	2022	11	1	7	48	臺中市	西屯區	0	1	0	8	1	5
8	48	2022	11	1	8	5	臺中市	西屯區	0	0	0	8		
9	61	2022	11	1	8	39	臺中市	西屯區	0	0	0	8		
10	63	2022	11	1	8	40	臺中市	西屯區	0	1	0	8	1	5
11	66	2022	11	1	8	47	臺中市	西屯區	0	0	0	8		
12	72	2022	11	1	8	14	臺中市	西屯區	0	1	0	8	1	5