# 心理與神經資訊學
# (Psychoinformatics & Neuroinformatics)

課號: Psy5261　　教室:綜合302

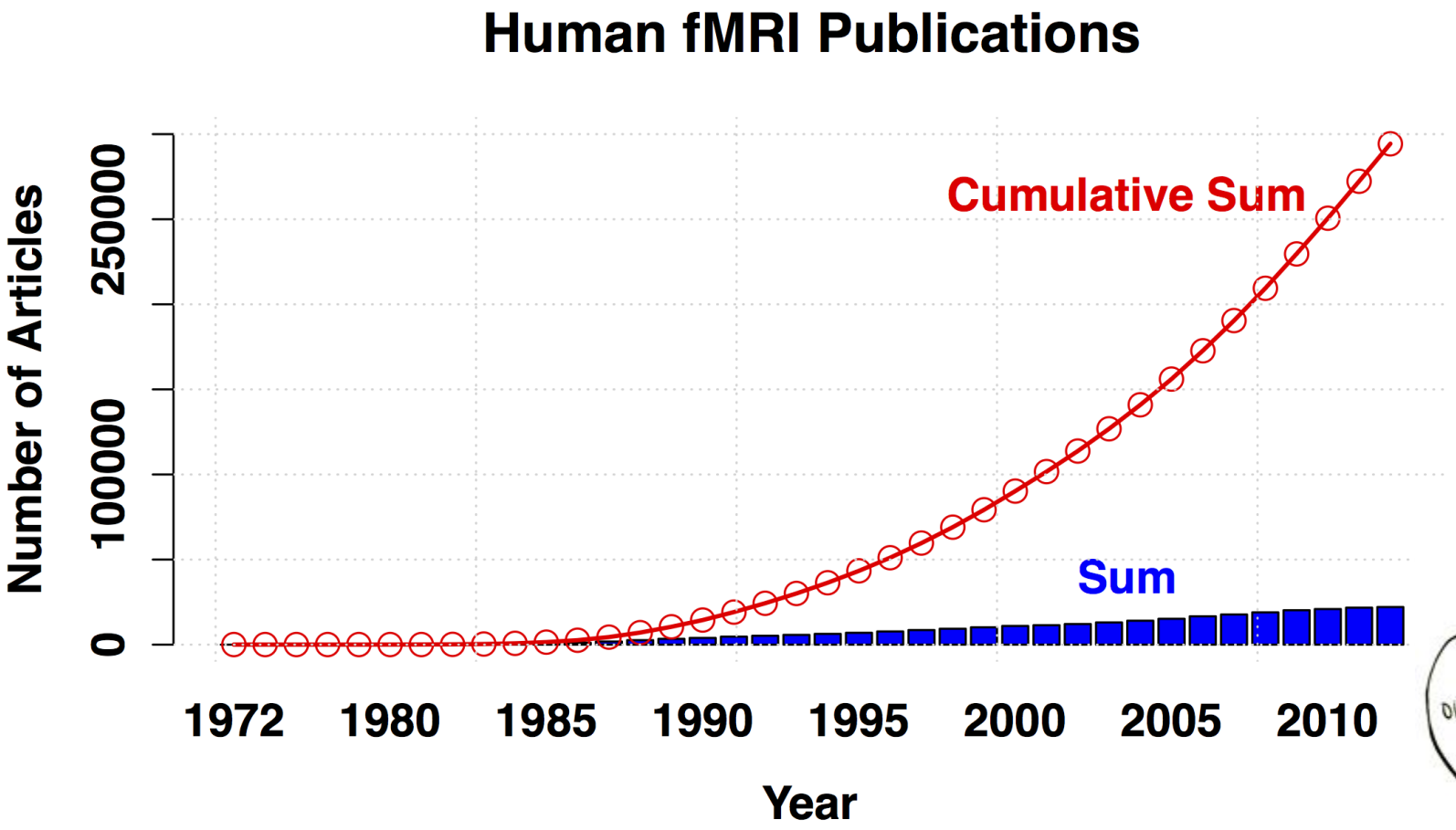
識別碼: 227U9340　　時間: 五234


作業可討論但請要抄襲喔
！

# 網頁前端技術
(HTML5, CSS, & JavaScript)

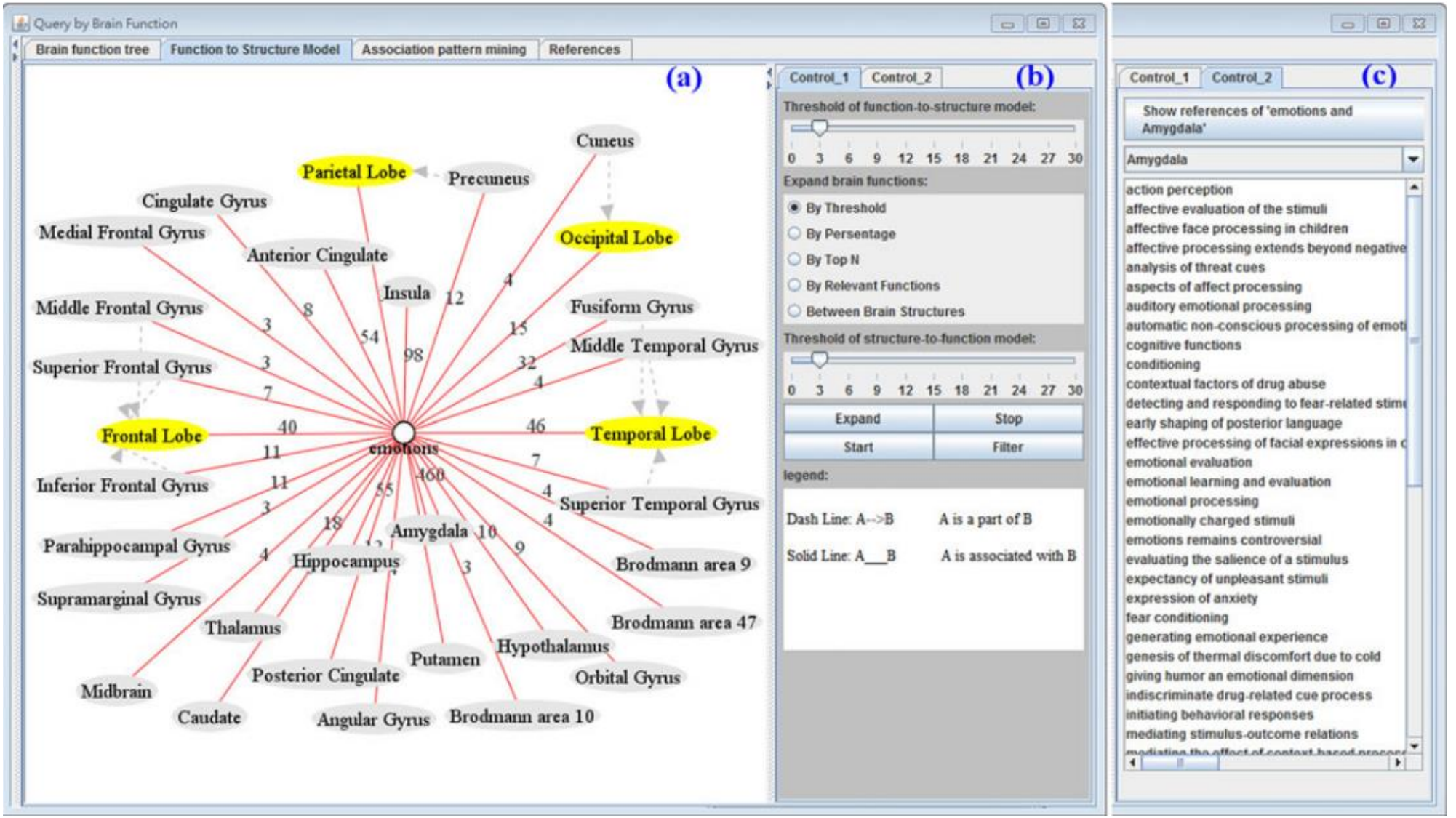
# 腦科學案例研究(1/3)

論文怎麼讀都讀不完




# 腦科學案例研究(2/3)

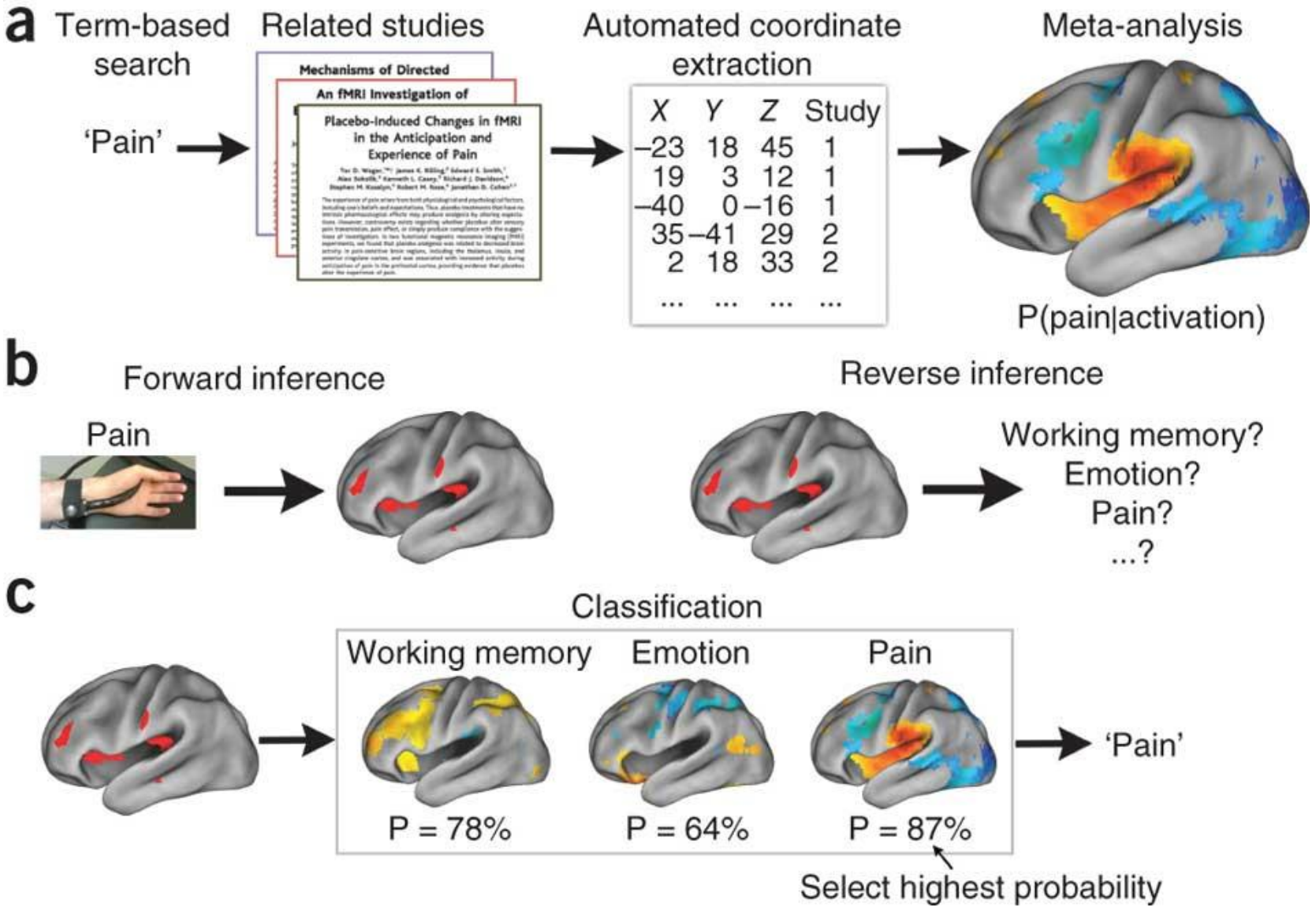
## 今天學完就可以讓程式可以幫我們讀論文




Hsiao et al., 2011, *Neuroinformatics*

# 腦科學案例研究(3/3)

## 用腦座標取代腦名稱可以更精確




Yarkoni et al., 2011, *Nature Methods*

作者 s512874690 (爪爪) 看板 PSY
標題 [疑問] 網路上的自己才是真正的自己嗎
時間 Fri Sep 25 17:22:05 2015

最近有點好奇，很多人在匿名的環境下發表的言論跟他們在現實上所表現的並不一樣

就好像以前用西斯匿名帳號發文的人，現實生活中可能是很避諱談到性這個方面的

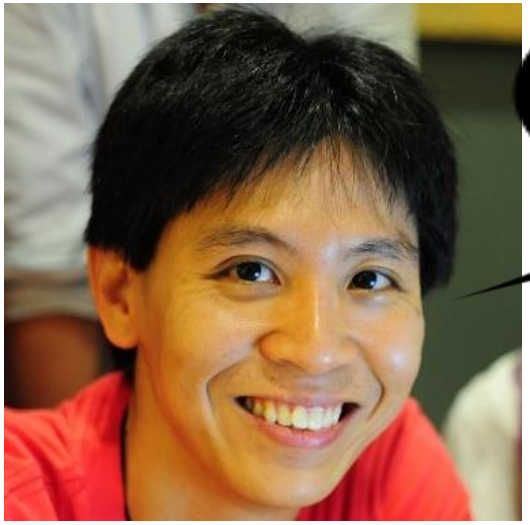
那在網路上的時候才是真正表現自己的個性嗎？

--
Sent from my Android

--
※ 發信站: 批踢踢實業坊(ptt.cc), 來自: 223.136.95.38
※ 文章網址: https://www.ptt.cc/bbs/PSY/M.1443172928.A.CEC.html
→ maoapple: 每個人的自我都有很多不同的面向，你可以理解成這是其 09/25 18:39
→ maoapple: 中一個面向。 09/25 18:39
推 twcandyman: 推樓上 也有人是盡可能表現一樣的 每個人的方式不同 09/25 20:29
推 winken2004: 現實中的你加網路中的你才是真正的你 09/27 01:52
→ lunenoir: 每個面向的自己都是你，看你願不願意接納而已 10/02 14:51

# 心理學案例研究(1/2)

當本門課魯師還在唸博士班時，
有一天系上的印度學長跑來問網路擷取資料的問題。



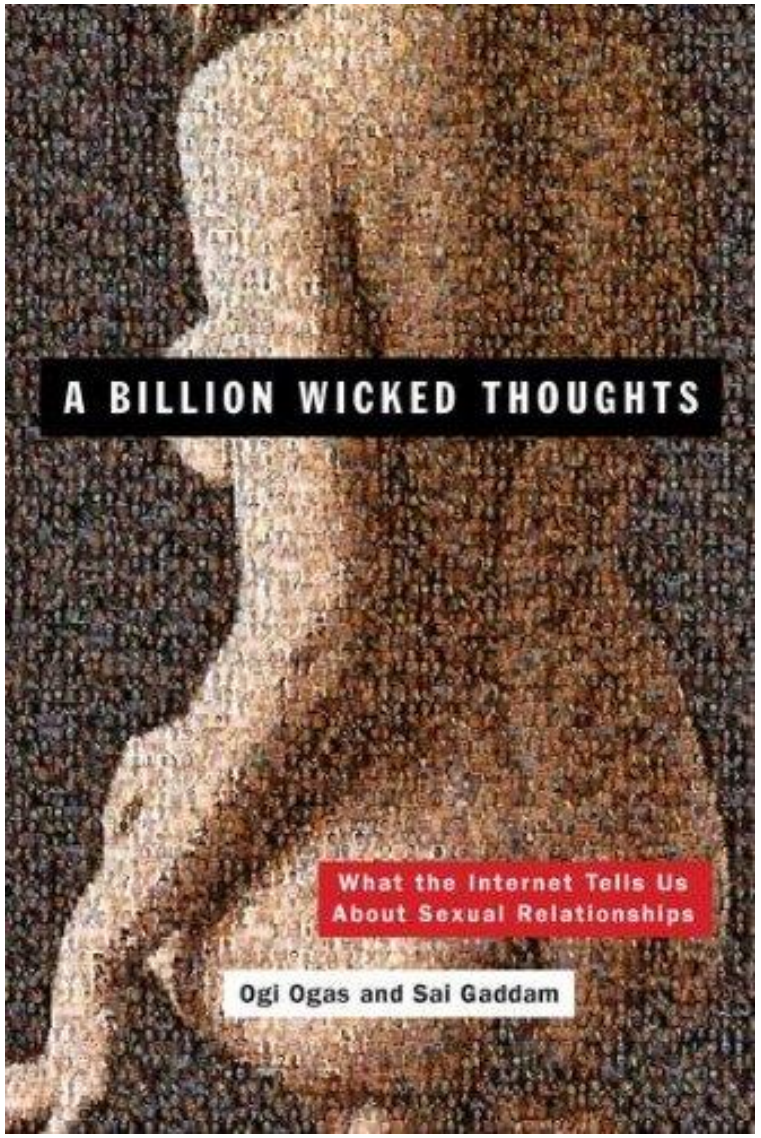
**Tren: How to get these data from dogpile?**

**Sai: That's easy! You can ...**

**感謝大大無私分享 Orz**

# 心理學案例研究(2/2)

結果印度學長後來和另一位學長出了這本書：




小故事大啟示：

師父領進門

修行在個人

# 今天實作範例:愛情心理學研究

假設我們要到PTT網頁版的Boy-Girl版搜集資料



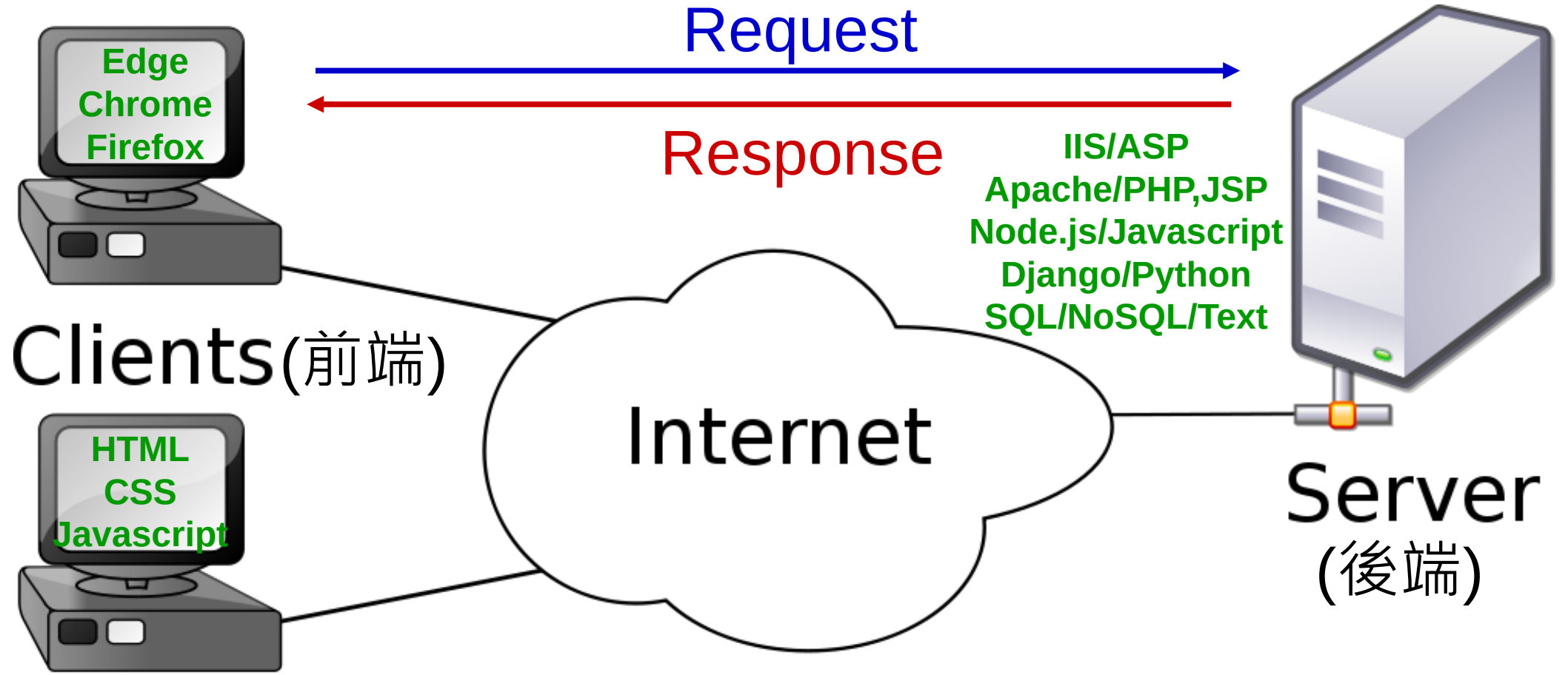
```
import urllib.request
u='http://www.ptt.cc/bbs/Boy-Girl/'
r=urllib.request.Request(u,headers={'User-Agent':''})
data=urllib.request.urlopen(r).read()
print(data.decode('utf-8'))
```

# 這些是什麼東東?

```
<div class="r-ent">
<div class="nrec"><span class="hl f2">1</span></div>
<div class="mark"></div>
<div class="title">
<a href="/bbs/Boy-Girl/M.1394964006.A.850.html">Re: [心情] 我要怎麼原諒那個人</a>
</div>
<div class="meta">
<div class="date">3/16</div>
<div class="author">wbson</div>
</div>
</div>
```

# 網頁前端(frontend)和後端(backend)

前端資料由瀏覽器來處理，
主要是頁面的顯示和小資料的儲存/查詢




後端資料由伺服器來處理，主要是大資料的儲存/查詢

# 基本的HTML語法

```
<h1>標1</h1><hr> <h2>標2</hr><a href=http://ptt.cc>連接</a>
這樣<br>可以斷行 <img src=http://bit.ly/1p2za46>
<center>這樣可以置中和<font color=red>換色</font></center>
<ul>
<li><b>粗體</b></li>
<li><i>斜體</i></li>    <li><u>底線</u></li>
</ul>
<table border=1>
<tr><td>11</td>    <td>12</td></tr>
<tr><td>21</td>                    <td>22</td></tr>
</table>
```


更多的HTML語法可來這裡學

# CSS: One style fits all

```
<style>
body {color:white; background-color:black;}
h1 {color:red; font-size:20pt}
.yy {color:yellow}
span#gg {color:green}
div#bb {color:blue}
</style>
```

更多的CSS語法可來這裡學

```
<h1>Hi!</h1>
This is <span class=yy>test1</span><hr>
This is <div class=yy>test2</div><hr>
This is <span id=gg>test3</span><hr>
This is <div id=gg>test4</div><hr>
This is <div id=bb>test5</div><hr>
```

# 這些是什麼東東?

```
<div class="r-ent">
 <div class="nrec"><span class="hl f2">1</span></div>
 <div class="mark"></div>
 <div class="title">
  <a href="/bbs/Boy-Girl/M.1394964006.A.850.html">Re: [心情] 我要怎麼原諒那個人</a>
 </div>
 <div class="meta">
  <div class="date"> 3/16</div>
  <div class="author">wbson</div>
 </div>
</div>
```

# 工欲善其事必先利其器(1/2)

## 使用Chrome的Developer Tools幫忙理解

# 工欲善其事必先利其器(2/2)

## FireFox也有對應的Developer Tools (原Firebug)

# 網頁資料的搜集

(LXML, Scrapy, & Selenium)

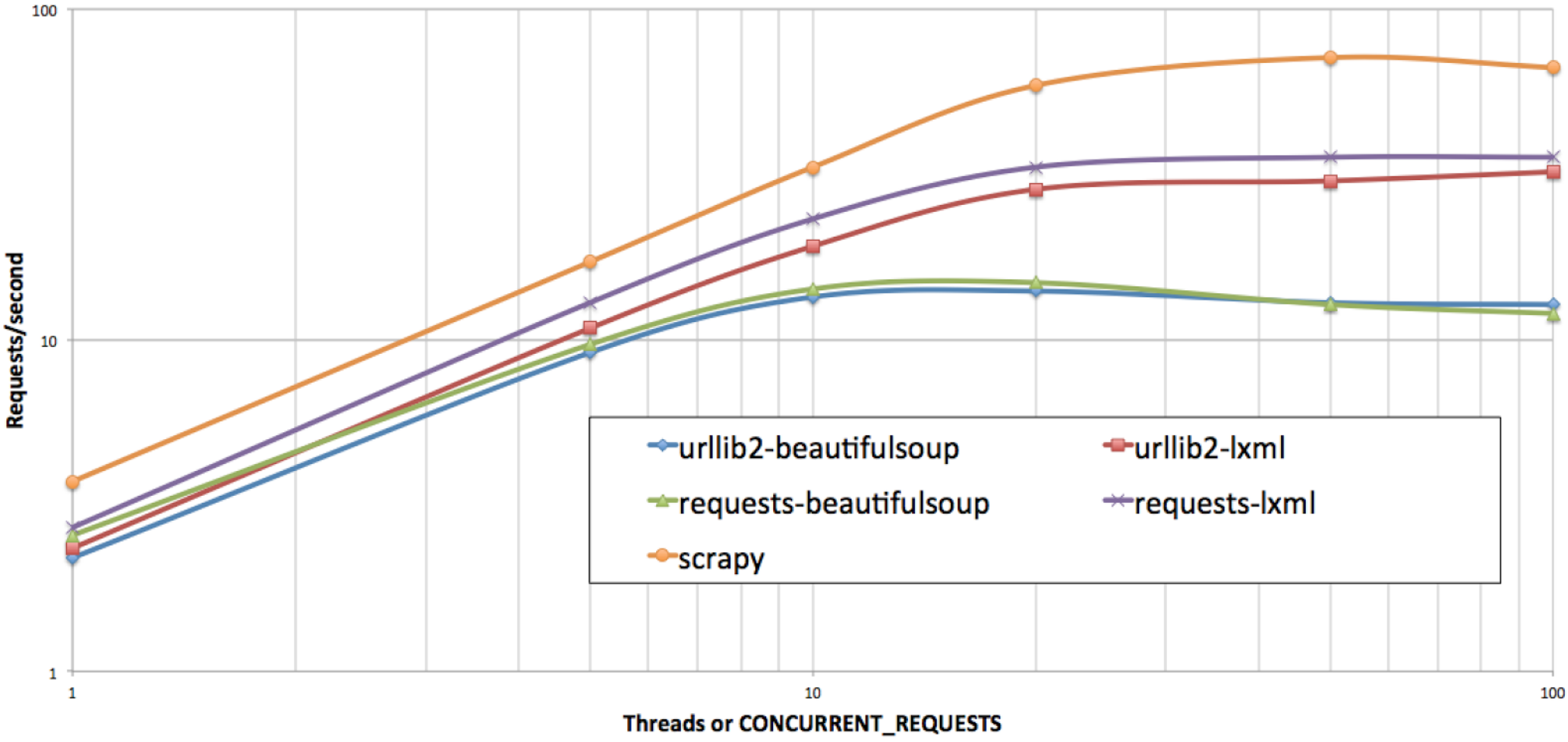
# 不同"爬蟲"的比較(1/3)

Beautiful Soup是最好學/寫的

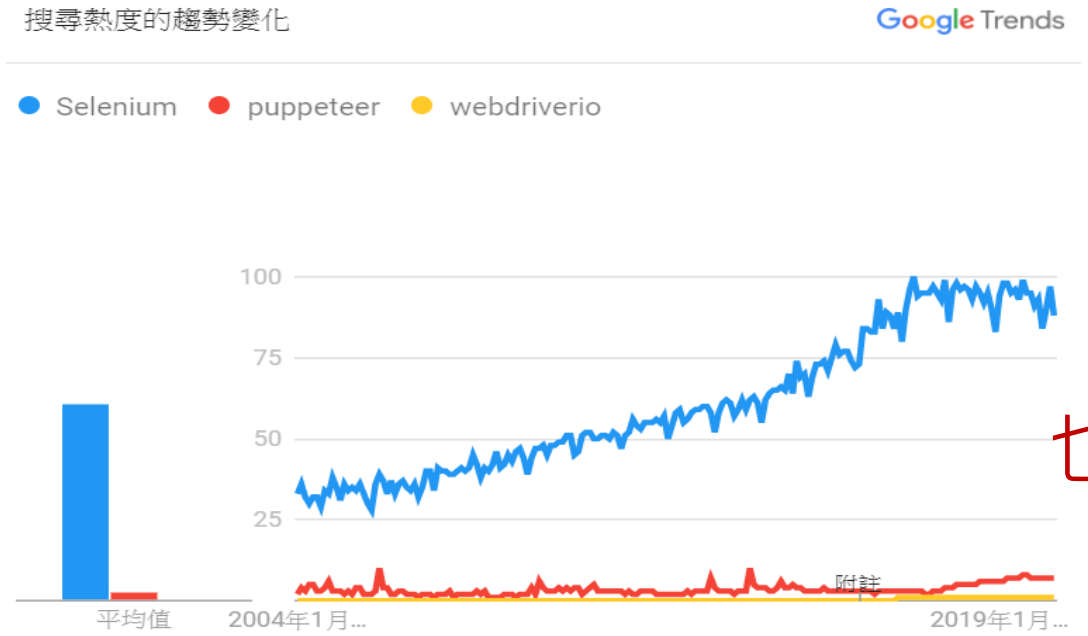
| Aspect | requests + lxml | requests + Beautiful Soup | Scrapy |
|---|---|---|---|
| *Performance* | ✔ | ✗ | ✔✔ |
| *Ease of installation* | ✔ | ✔ | ✔ |
| *Development experience* | ✗ | ✔✔ | ✔ |
| Memory usage | ✔ | ✔ | ✔ |
| *Output files and formats* | ✗ | ✗ | ✔ |
| *Javascript support* | ✗ | ✗ | ✗ |

# 不同"爬蟲"的比較(2/3)

Beautiful Soup卻是處理速度最慢的

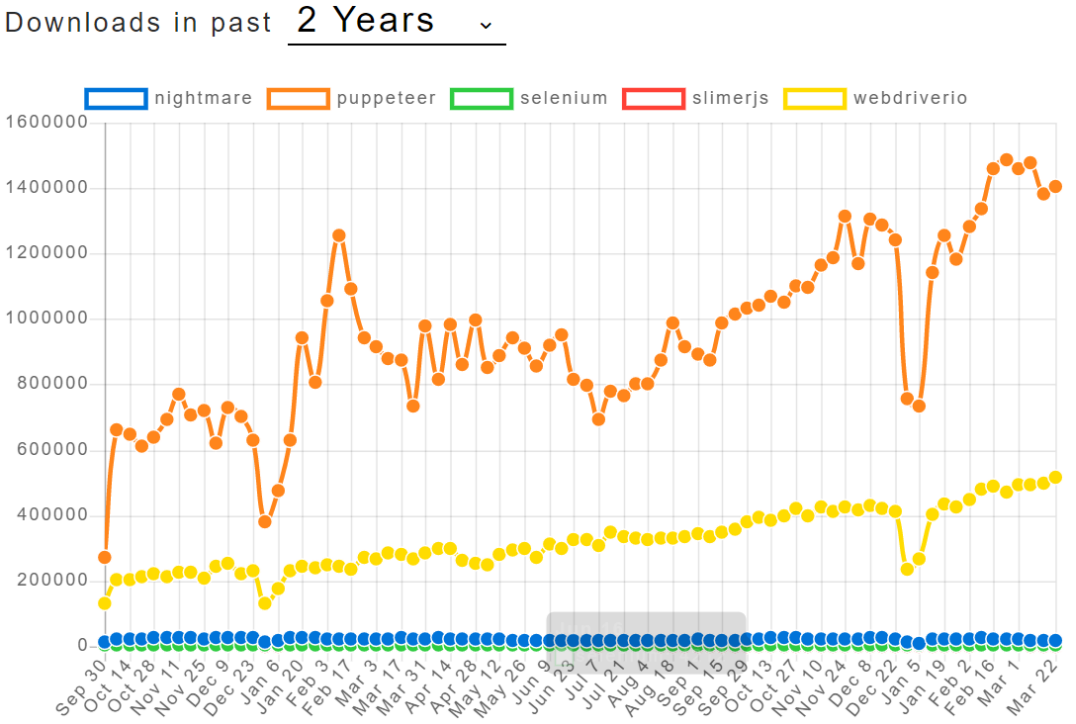


# 不同"爬蟲"的比較(3/3)



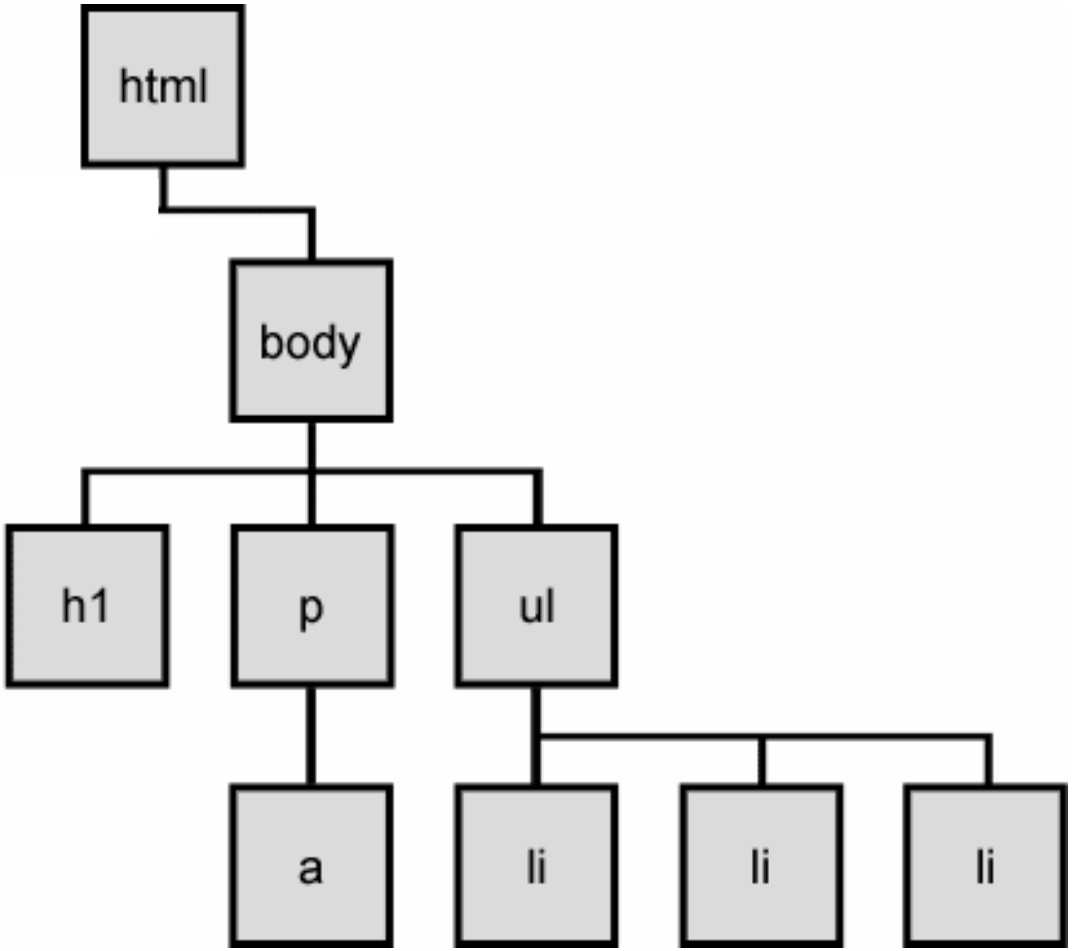

也有用JS寫的瀏覽器控制器

最夯的是puppeteer

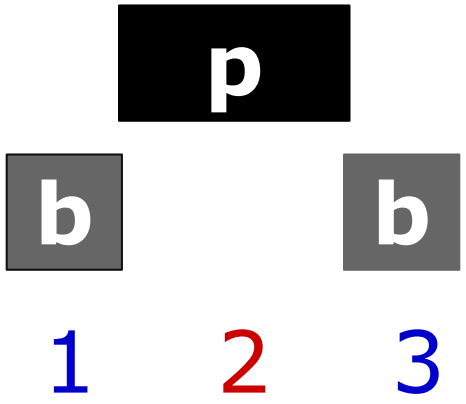


# LXML: HTML Parser

讓我們可悠遊在HTML文件的樹狀結構中



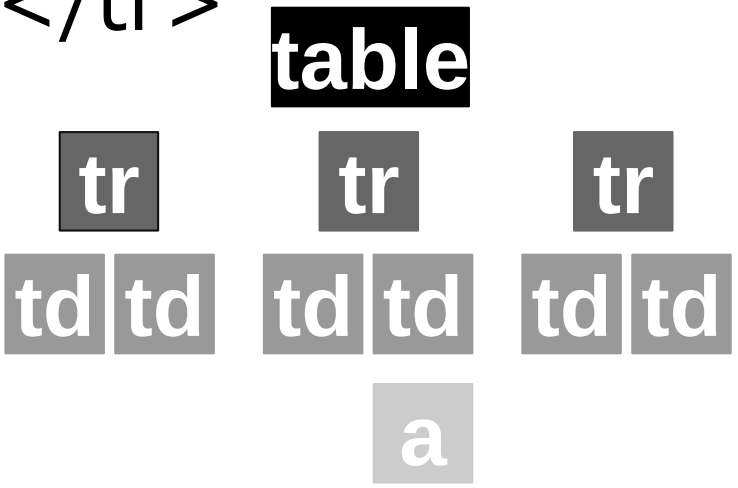
# Scrapy Selector基本語法(1/2)




```
from scrapy.selector import Selector
data='<p><b>1</b>2<b>3</b></p>'
t=Selector(text=data)
print(t.xpath('//p').extract())
print(t.xpath('//p/text()').extract()) #2
print(t.xpath('//p/*').extract()) #1,3
print(t.xpath('//b/text()')) #1,3
print(t.xpath('//p').re('\d')) #2
```

# Scrapy Selector基本語法(2/2)

```
from scrapy.selector import Selector
data='''<table>
<tr><td>11</td><td>12</td></tr>
<tr><td>21</td><td><a
href="http://ptt.cc">22</a></td></tr>
<tr><td>31</td><td>32</td></tr>
</table>'''
```




```
t=Selector(text=data)
print(t.xpath('//td'))
print(t.xpath('//td/..')[1].xpath('*/text()'))
print(t.xpath('//a/@href'))
print(t.xpath('//a/text()'))
```

# 搜集連結(links)

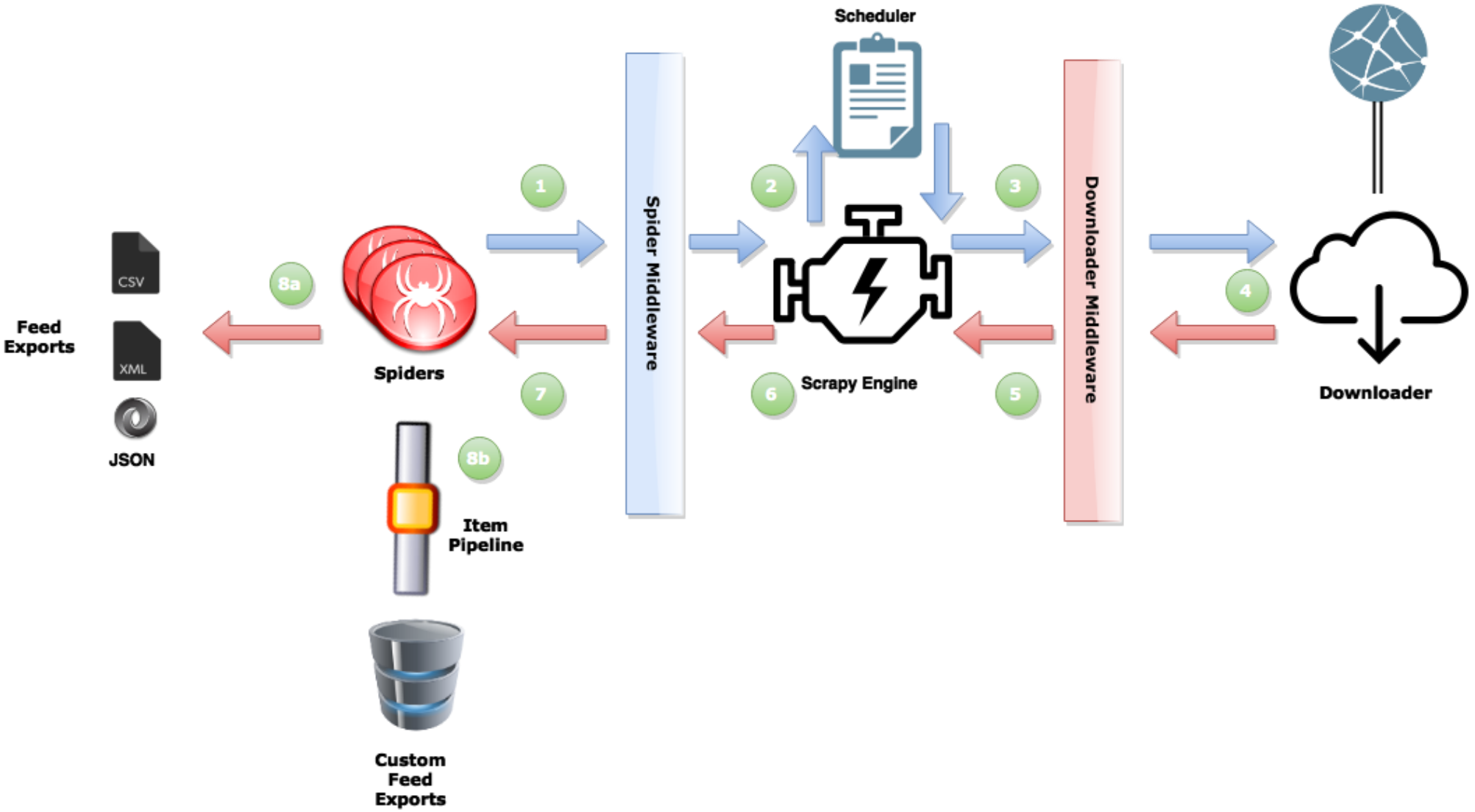
可在Terminal裡或是Jupyter Notebook跑

(Scrapy裡其實有個LinkExtractor)

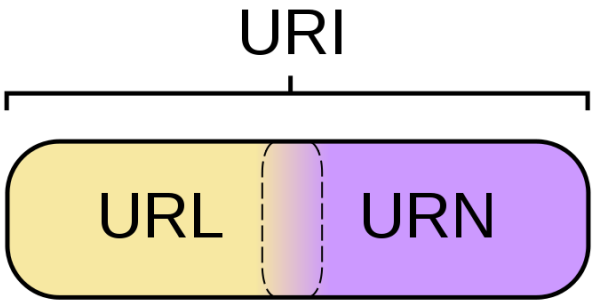

```
import scrapy
class Spider(scrapy.Spider):
  name="ptt"
  start_urls=["http://www.ptt.cc/bbs/Boy-Girl/"]
  def parse(self, response):
    for link in response.xpath('//a'):
      print(link.xpath('text()').extract())
      print(link.xpath('@href').extract())
```

# Scrapy架構

殺雞用牛刀?



# LXML搜集文章資訊(1/2)



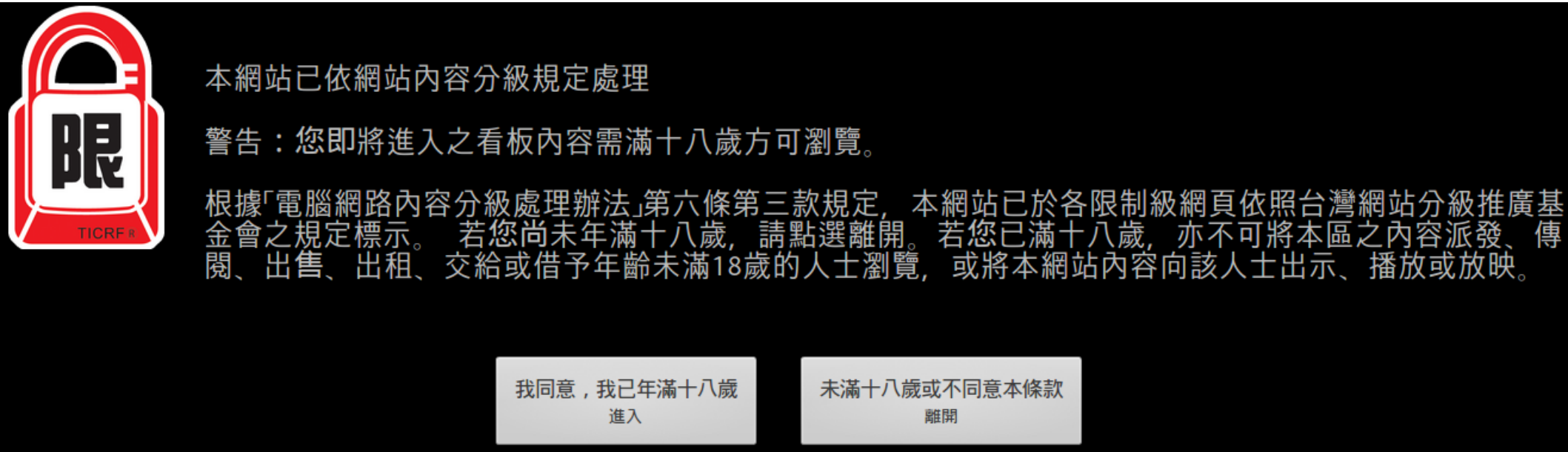

```
import urllib, lxml.html
URL='http://www.ptt.cc'
URN='/bbs/Boy-Girl/M.1394964006.A.850.html'
h={'User-Agent':'Mozilla/5.0'}
r=urllib.request.Request(URL+URN,headers=h)
data=urllib.request.urlopen(r).read()
t=lxml.html.fromstring(data.decode('utf-8'))
print(t.text_content()) #整頁
```

# LXML搜集文章資訊(2/2)

```
x=t.xpath('//div[@id="main-content"]')[0]
print(x.text_content()) #主文
y=t.xpath('//div[@id="main-content"]/text()')
print(''.join(y)) #新文
z=t.xpath('//span[@class="f6"]')
for i in z:
 print(i.text) #引言
H=t.xpath('//*[contains(text(),"恨")]')[0]
print(H.text) #含恨
```

# 18禁的八卦版

```
import urllib.request
u='http://www.ptt.cc/bbs/Gossiping/'
h={'User-Agent':'Mozilla/5.0'}
r=urllib.request.Request(u,headers=h)
data=urllib.request.urlopen(r).read()
print(data.decode('utf-8'))
```

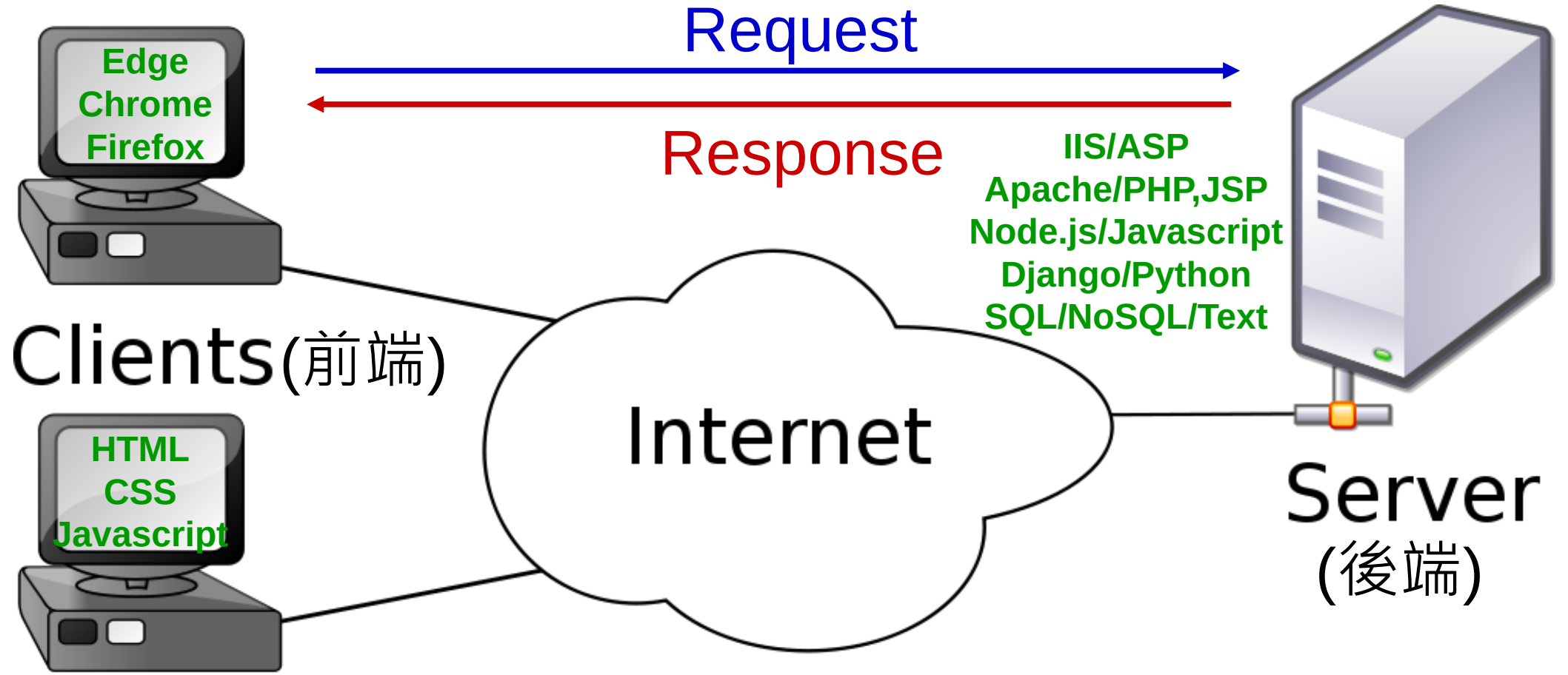
# 花生鼠模式?

藍字為前端請求 紅字為後端回應者

```
<div class="bbs-screen bbs-content center clear">
    <form action="/ask/over18" method="post">
        <input type="hidden" name="from"
value="/bbs/Gossiping/index.html">
        <button class="btn-big" type="submit" name="yes"
value="yes">我同意，我已年滿十八歲<br><small>進入
</small></button>
        <button class="btn-big" type="submit" name="no"
value="no">未滿十八歲或不同意本條款<br><small>離開
</small></button>
    </form>
</div>
```

# 網頁前端(frontend)和後端(backend)

前端資料由瀏覽器來處理，
主要是頁面的顯示和小資料的儲存/查詢

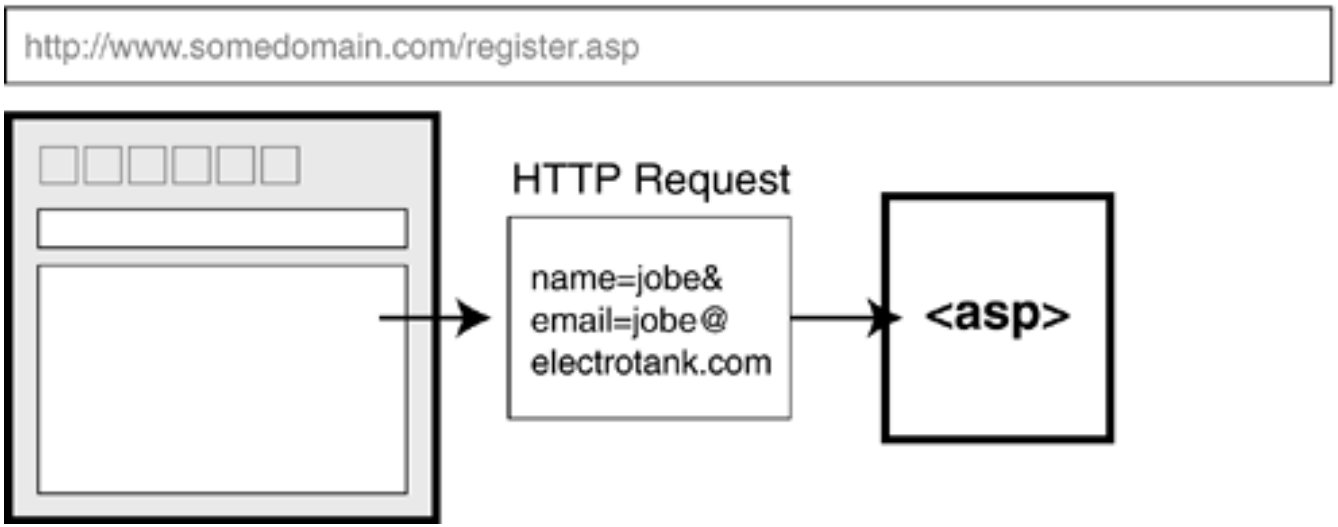



後端資料由伺服器來處理，主要是大資料的儲存/查詢

# Request傳送資料方式: Get vs. Post

## Using GET


## Using POST 比較安全



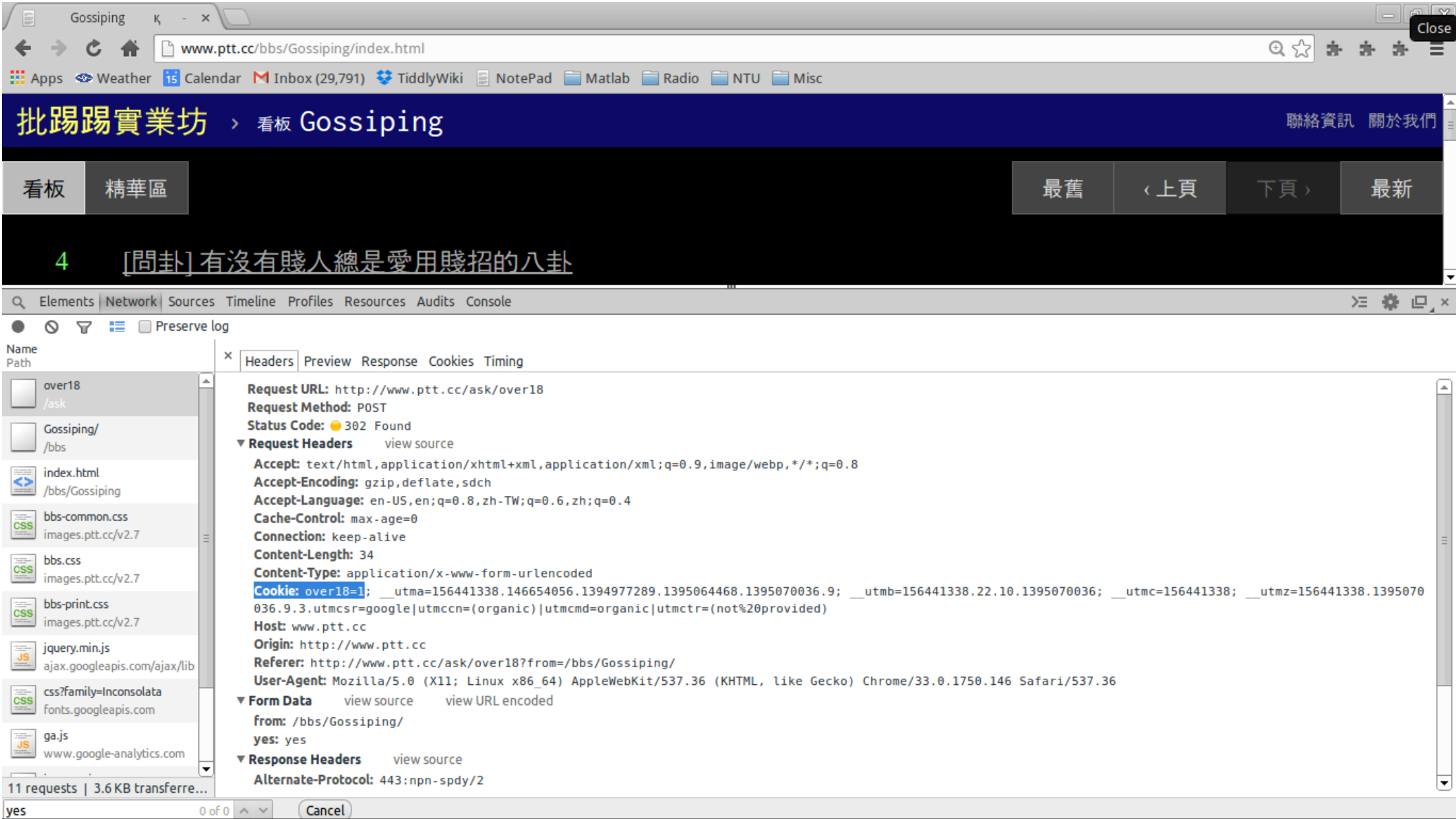
# 破解18禁的八卦版(1/3)

```
from urllib import parse,request
URL='https://www.ptt.cc'
URN='/ask/over18'
q=parse.urlencode({'yes':'yes','from':'/bbs/Gossiping/'})
q=q.encode('utf-8')
h={'User-Agent':'Mozilla/5.0'}
req=request.Request(URL+URN,q,h)
response=request.urlopen(req)
data=response.read()
print(data.decode('utf-8'))
```

為何還是不行?

# 破解18禁的八卦版(2/3)

## 看看瀏覽器到底做了什麼?



# 破解18禁的八卦版(3/3)

```
from urllib import parse,request
URL='https://www.ptt.cc'
URN='/ask/over18'
q=parse.urlencode({'yes':'yes','from':'/bbs/Gossiping/'})
q=q.encode('utf-8')
h={'Cookie':'over18=1','User-Agent':'Mozilla/5.0'}
req=request.Request(URL+URN,q,h)
response=request.urlopen(req)
data=response.read()
print(data.decode('utf-8'))
```

# 更進階的爬蟲議題

師父領進門,修行在個人

**驗證碼**: pytesseract, selenium, touclick

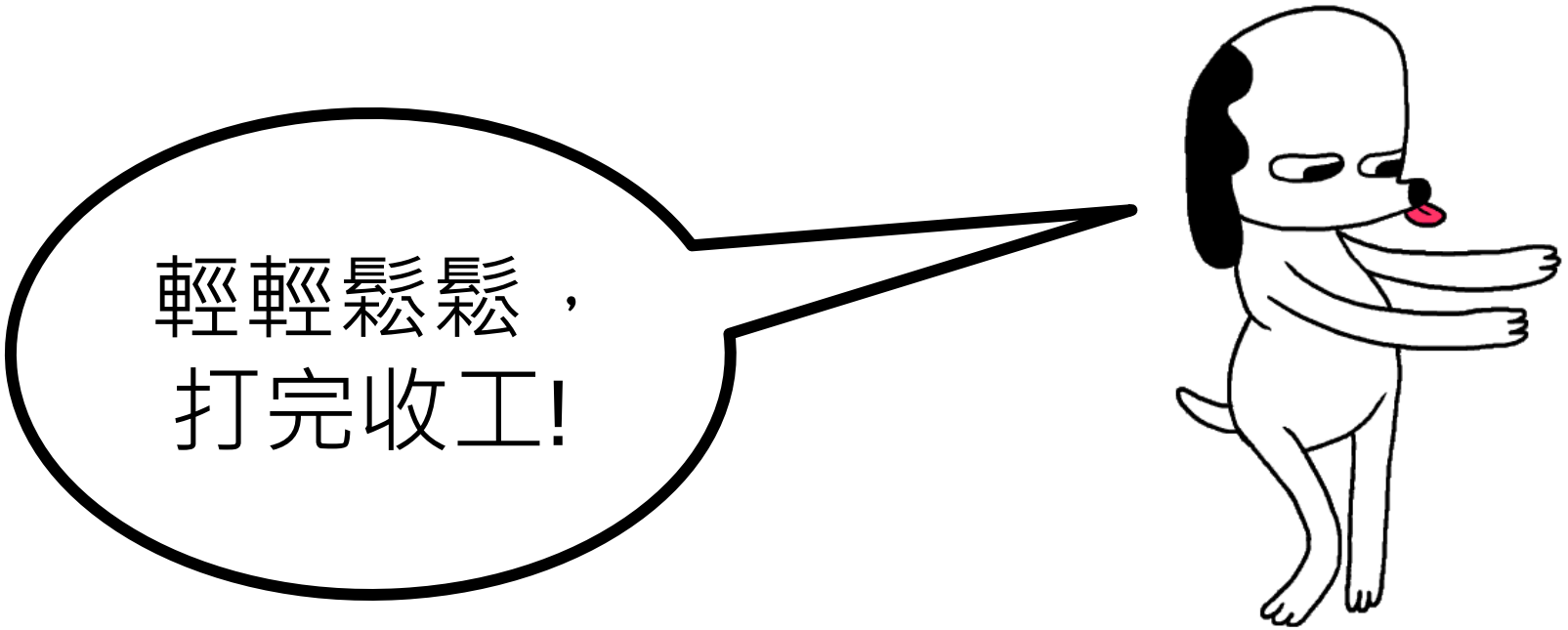
**JS產生的動態資料**: scrapy-splash

**分散式爬取**: scrapy-redis

# 神 Selenium 神

## Selenium可自動化一切瀏覽動作

```
from selenium import webdriver
URI='https://www.ptt.cc/bbs/Gossiping/'
driver=webdriver.Chrome() # try Firefox()
driver.get(URI)
btn=driver.find_element_by_name('yes')
driver.save_screenshot('before_click.png')
btn.click()
driver.save_screenshot('after_click.png')
print(driver.page_source)
```


輕輕鬆鬆，
打完收工!

# 本週作業

進一步搜尋Boy-Girl版資訊

1. index.html右上角[<上頁]中包含了總頁數資訊，請用LXML抓出此經常變動的數字。

2. 請用LXML找出距離現在時間最近的一篇[爆]文標題與URN。

3. 請用Selenium在index.html往前翻三頁並拍照。

# Game Over