



# 網路爬蟲與資料分析 動態網頁解析

Instructor: 馬豪尚

# 常見的反爬蟲機制

## › 判斷瀏覽器 headers 資訊

- 判斷 User-Agent 是否像瀏覽器
- 檢查是否缺少一般瀏覽器會自動帶上的 headers

## › 判斷使用者行為

- 檢查使用者行為（如滑鼠移動、點擊、滾動等）來分辨是否為真人。  
例如，偵測頁面上的滑鼠移動或停留時間等行為，爬蟲若無法模擬這些操作，將難以通過此檢測。

## › Cookie 驗證與授權 Token

- 使用者登入後，將授權 token 存入瀏覽器的 Cookie，並要求後續請求帶上該 token 以確認合法性

# 常見的反爬蟲機制

## › 驗證碼機制

- 常見的驗證機制，用於阻擋自動化的請求
- 驗證碼（如 reCAPTCHA）可有效區分人類和機器請求

## › 請求頻率限制

- 限制單個 IP 或帳號的請求頻率，若達到特定次數則進行封鎖或設置冷卻時間

## › 封鎖代理伺服器

- 對使用代理伺服器或來自異常位置的 IP 進行封鎖。
- 某些網站會利用 IP 黑名單阻擋已知的爬蟲來源，並拒絕匿名代理的連接。

# 判斷瀏覽器 headers 資訊

- › 檢查http標頭資訊的user-agent
- › 應對方法
  - 將user-agent加入header資訊偽裝成各個瀏覽器送出請求

Request Example:

```
headers = {'user-agent': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36  
(KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36'}  
web = requests.get(url, headers=headers)
```

Selenium Example:

```
user_agent = "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/605.1.15  
(KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"  
opt = webdriver.ChromeOptions()  
opt.add_argument('--user-agent=%s' % user_agent)  
driver = webdriver.Chrome('./chromedriver', options=opt)
```

# 判斷瀏覽器 headers 資訊

- › 檢查是否缺少一般瀏覽器會自動帶上的 headers
  - 檢測瀏覽器的 `window.navigator` 是否包含 `webdriver` 屬性，在正常使用瀏覽器的情況下，`webdriver` 屬性是 `undefined`
- › 應對方法：
  - 使用 `selenium webdriver` 的 `execute_cdp_cmd`，將 `webdriver` 設定為 `undefined`

```
driver.execute_cdp_cmd("Page.addScriptToEvaluateOnNewDocument", {  
  "source": ""  
    Object.defineProperty(navigator, 'webdriver', {  
      get: () => undefined  
    })  
  ""  
})
```

# Python fake-useragent套件

## › 模組安裝

- `pip install fake_useragent`
- `From fake_useragent import UserAgent`

## › 宣告物件

- `ua = UserAgent()`

# Python fake-useragent套件

- › 產生各瀏覽器的假useragent
  - ua.ie
  - ua.google
  - ua.firefox
  - ua.safari
- › 將useragent加入header一併送出
  - headers={'user-agent': ua.google}
  - request.get(url, headers=headers)

# 判斷使用者行為

- › 判斷使用者刷新網頁/做出某種操作的時間
  - 應對方法: 加入(隨機)等待機制
    - › `from time import sleep`
    - › `sleep(number)`
- › 判斷是否是真的滑鼠點擊，而非機器送出點擊指令
  - 應對方法: 滑鼠需要真實移動到某個互動的元素之上
  - 使用Selenium的action chain
    - › `actions = ActionChains(driver)`
    - › `actions.move_to_element(submitBtn).click(submitBtn)`
    - › `actions.perform()`



# Cookie 驗證與授權 Token

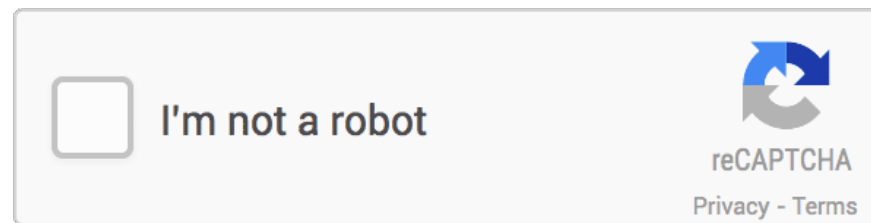
- › 檢查是否有授權 token 存入瀏覽器的 Cookie，並要求後續請求帶上該 token 以確認合法性
- › 應對方法: 加入授權token進cookie即可

# 驗證碼機制

- › 加入驗證碼技術驗證是否為真實使用者
  - 例如 reCAPTCHA
- › 應對方法:
  - 破解驗證碼的難度通常較高，尤其是圖片或行為型驗證碼，需搭配一些 AI 來處理圖形、數字、文字的識別，通常只要能識別驗證碼就能破解
  - reCAPTCHA V2 Invisible
    - › <https://github.com/2captcha/2captcha-python?tab=readme-ov-file#recaptcha-v2>

# reCAPTCHA V2 Invisible

- › 安裝package
  - pip3 install 2captcha-python
  - from twocaptcha import TwoCaptcha
- › 使用TwoCaptcha函數輸入你的2captcha API Key宣告物件
  - solver = TwoCaptcha('YOUR\_API\_KEY')



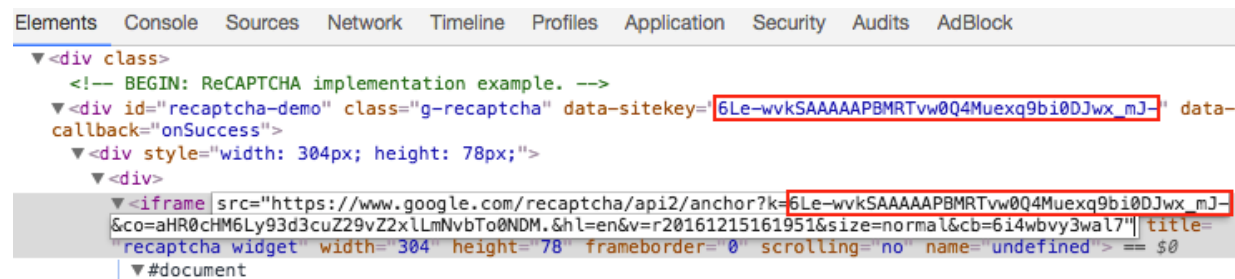
# reCAPTCHA V2 Invisible

## › Example

– <https://www.google.com/recaptcha/api2/demo>

## › 在該網頁頁面中找到data-sitekey

– data-sitekey="6Le-wvkSAAAAAPBMRTvw0Q4Muexq9bi0DJwx\_mJ-"



```
Elements Console Sources Network Timeline Profiles Application Security Audits AdBlock
▼ <div class=
  <!-- BEGIN: ReCAPTCHA implementation example. -->
  ▼ <div id="recaptcha-demo" class="g-recaptcha" data-sitekey="6Le-wvkSAAAAAPBMRTvw0Q4Muexq9bi0DJwx_mJ-" data-
    callback="onSuccess">
      ▼ <div style="width: 304px; height: 78px;">
        ▼ <div>
          ▼ <iframe src="https://www.google.com/recaptcha/api2/anchor?k=6Le-wvkSAAAAAPBMRTvw0Q4Muexq9bi0DJwx_mJ-
            &sco=aHR0cHM6Ly93d3cuZ29vZ2x1LnVbTo0NDM.&hl=en&v=r20161215161951&size=normal&cb=6i4wbvy3wal7" title=
              "recaptcha widget" width=304 height=78 frameborder=0 scrolling=no name=undefined"> == $0
          ▼ #document
```

# reCAPTCHA V2 Invisible

- › 使用solver物件呼叫recaptcha函式
  - result =  
solver.recaptcha( sitekey='6LdO5\_IbAAAAAAeVBL9TCIS19NUT  
t5wswEb3Q7C5',  
url='https://2captcha.com/demo/recaptcha-v2-invisible')

# Cloudflare 5秒挑戰

## › Cloudflare採取多層次的防護機制

- IP封鎖和速率限制：監控訪問頻率和模式，識別異常行為，對可疑IP地址進行速率限制或封鎖。
- JavaScript挑戰：要求訪問者執行特定的JavaScript代碼，以驗證其為真實使用者。
- 設備指紋識別：收集並分析訪問設備的特徵資訊，區分自動化爬蟲和真實使用者。
- CAPTCHA驗證：當檢測到可疑行為時，觸發CAPTCHA驗證，阻止自動化腳本的操作。

# 練習

- › 爬取內政部不動產交易實價查詢網站
  - <https://lvr.land.moi.gov.tw/>
- › 使用Selenium模擬瀏覽器和網頁互動
  - 輸入縣市和鄉鎮市區(任意)
  - 爬取顯示出的前15筆資料
  - 將資料存成csv檔