D19：動態網頁爬蟲 - 使用Selenium + BeautifulSoup 模擬瀏覽器執行

簡報閱讀　範例與作業　問題討論

動態網頁爬蟲 - 使用 Selenium

本日知識點目標

第一種動態網頁爬蟲策略

利用 Selenium 模擬操作瀏覽器

準備 Selenium 環境

範例：使用 Selenium 進行爬蟲

重要知識點複習

動態網頁爬蟲 - 使用 Selenium

本日知識點目標

解題時間

- 能夠使用 Selenium 撰寫動態網頁爬蟲

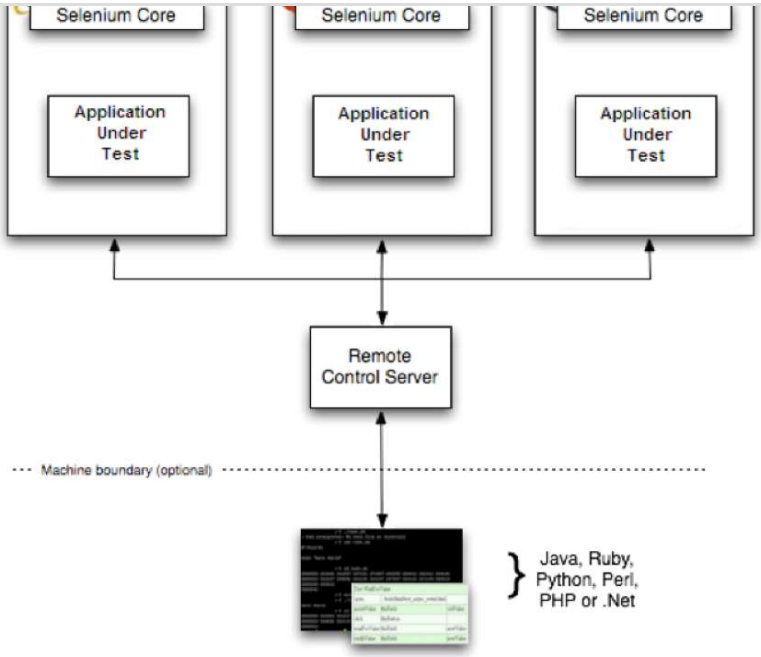
第一種動態網頁爬蟲策略

關於這種利用到 JavaScript 的非同步特性載入更多資料的網頁稱為動態網頁。而爬蟲程式也會因為沒有執行到 JavaScript 導致資料不完全的現象。

第一種解法會採用 selenium 這樣的瀏覽器模擬工作，從模擬使用者打開瀏覽器的行為，到模擬器執行JavaScript 動態載資料之後。

利用 Selenium 模擬操作瀏覽器

Selenium 是一個瀏覽器自動化（Browser Automation）工具，讓程式可以直接驅動瀏覽器進行各種網站操作。最早的目的是用來進行網頁測試使用，這邊我們藉由特性來運行 JavaScript 作為爬蟲用。



準備 Selenium 環境

1. 安裝 selenium 套件

```
$ pip install selenium
```

2. 下載 Chrome 驅動程式

上面第一步驟只是安裝 Selenium 模組而已，必須要下載對應的瀏覽器 Chrome 的驅動程式（建議放在程式相同目錄下）：

http://chromedriver.chromium.org/downloads

範例：使用 Selenium 進行爬蟲

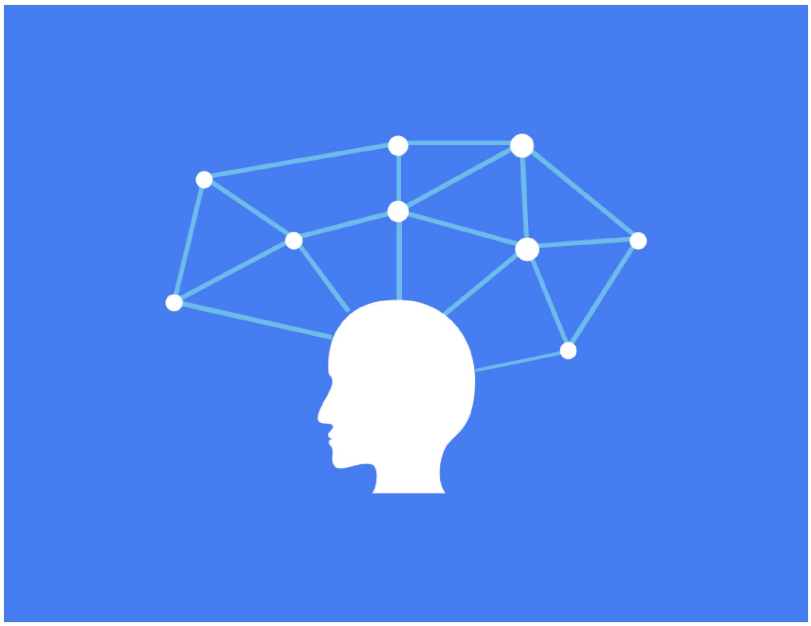
```
from selenium import webdriver

browser = webdriver.Chrome(executable_path='./chromedriver')
browser.get("http://www.google.com")
browser.close()

browser.page_source
```

這邊設定會去讀取我們放在相同目錄下的 driver 檔案

跳轉到設定的網址上！

透過 **browser.page_**source 可以取出，目前網頁上當下的 HTML，不過這是一個 HTML 格式的字串，此時就可以再利用 BeautifulSoup 進行解析。

重要知識點複習



- 了解 Selenium 用於動態網頁爬蟲的原理
- 能夠使用 Selenium 撰寫動態網頁爬蟲

解題時間

下一步：閱讀範例與完成作業