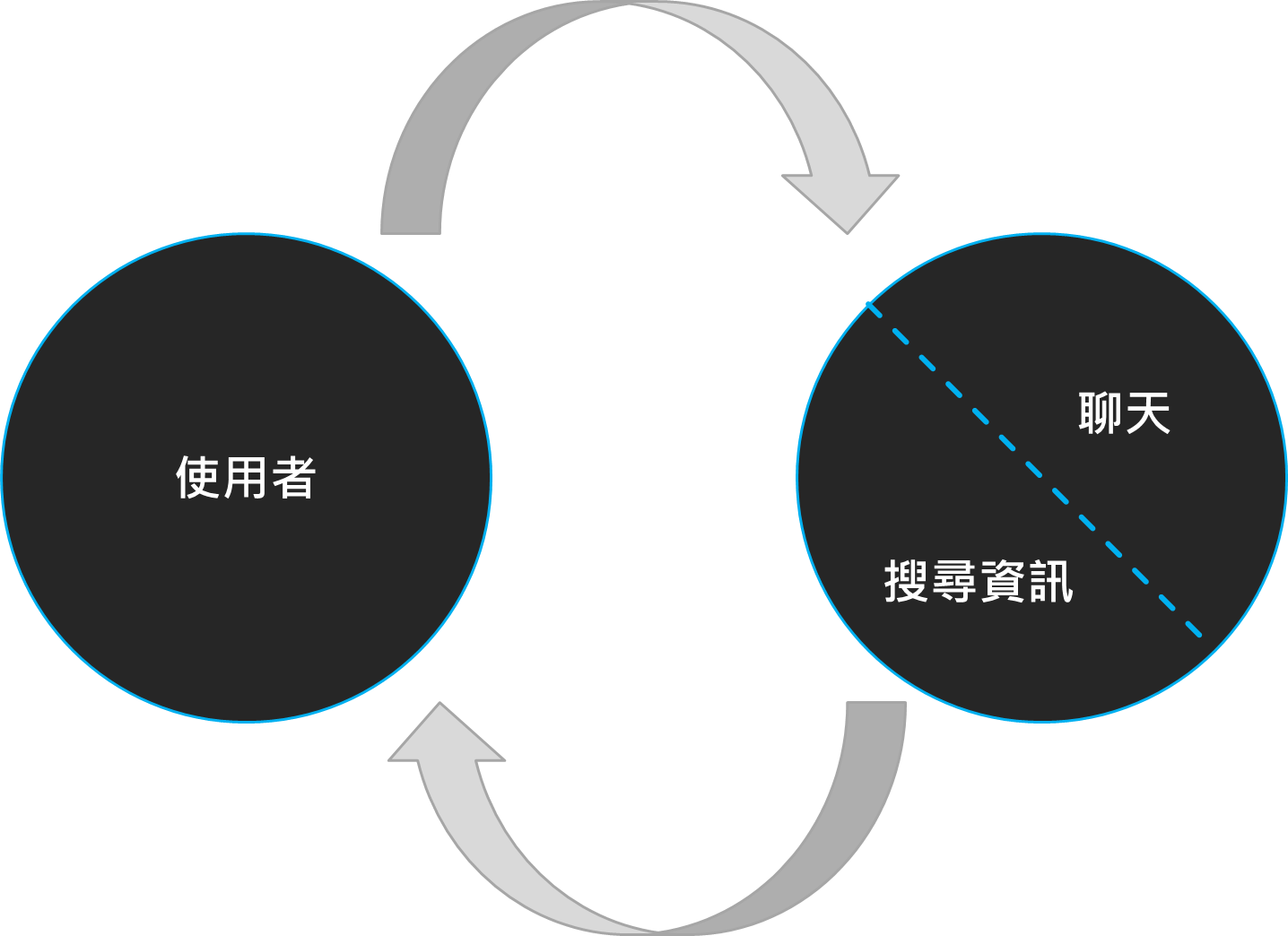
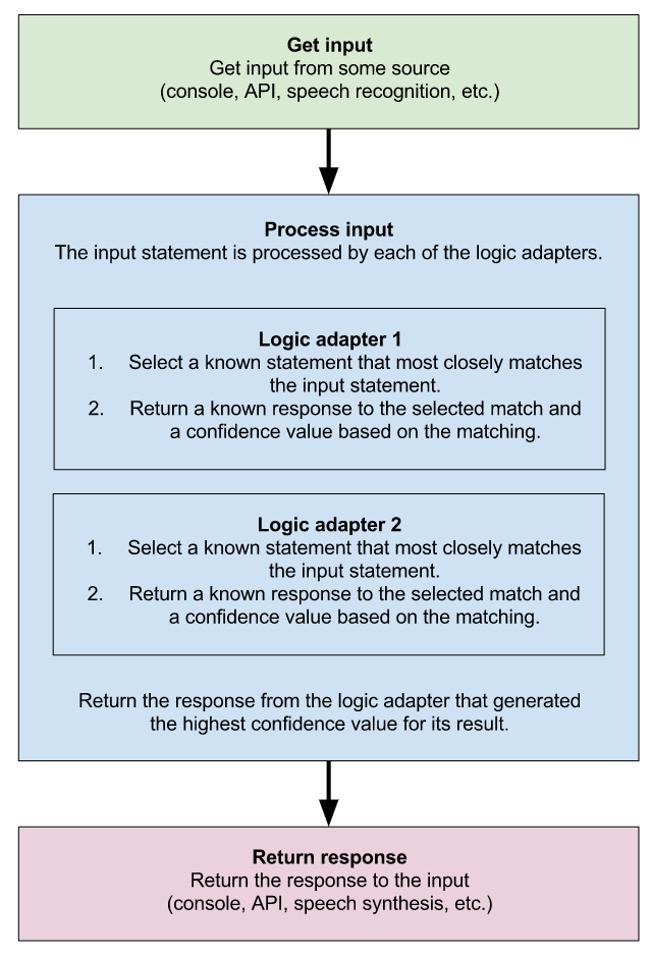
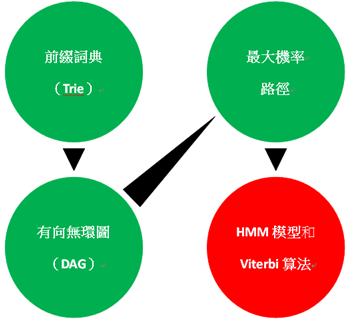
資訊檢索  
期末報告(word)

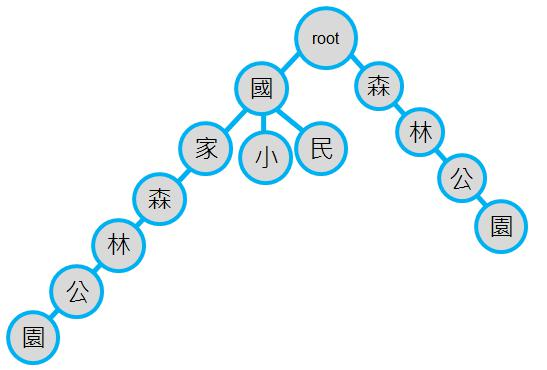
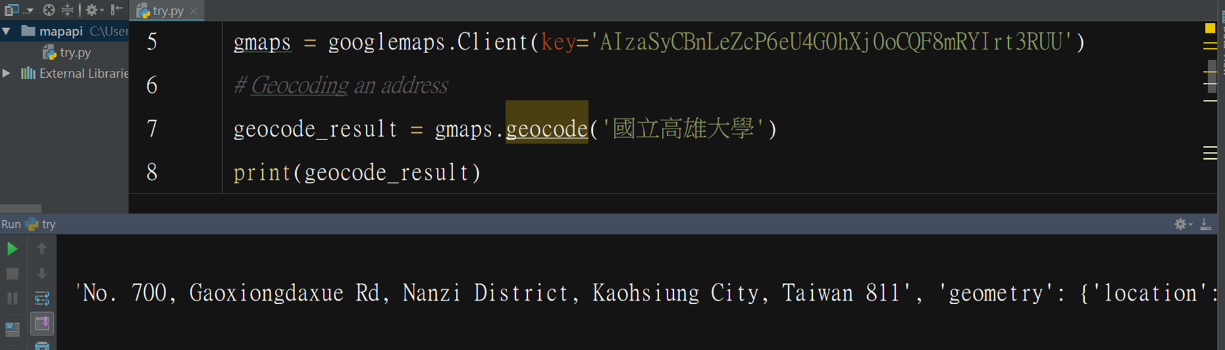
利用網路爬蟲實作簡易版Siri

A1045528 王彥翔

A1045503 趙普

* 環境和架構  
  以Python實作，整體架構如Fig.1  
    
  ▲Fig.1  
  首先，聊天和分辨使用者是否要搜索資訊的功能以「ChatterBot」實現
  1. **ChatterBot**整體架構和原理如Fig.2  
       
     ▲Fig.2 (資料來源：https://chatterbot.readthedocs.io/en/stable/)
     1. Get input  
        包含其他程式的API都可，日後甚至可結合其他程式使用。
     2. Process input  
        此節分為前、後兩部分：
        1. Preprocessors  
           Preprocessors會先將輸入處理為可辨識的統一格式，如去除空白、將unicode字元轉換為ASCII等。
        2. Logic adaper  
           主要處理和回話部分，會有複數個logic adaper同時處理，並回傳值，最後則選擇最高值回復，若值相同則選最快處理完者。
     3. Return response  
        將剛選擇出得回復傳出，和input相同，亦之前他程式的API。
  2. **資訊檢索**若ChatterBot處理時發現使用者要搜尋資料時便會使用網路爬蟲統整資料作為回復；為了達到此目的則需要Requests和BeautifulSoup。
     1. Requests  
        使用瀏覽器瀏覽時使用者端會向伺服器請求資料，伺服器再回傳，並藉由瀏覽器呈現；Requests套件即是代向伺服器發出請求的工具，收到資料後再由BeautifulSoup處理。
     2. BeautifulSoup  
        其主要功能為解析HTML、XML檔，並將待解析資訊建立為一棵樹，固可使用遍歷樹、搜尋樹相關的Function。
  3. **Jieba(結巴)**結巴主要功能為斷詞，同時也具有詞性標註、提取關鍵字等功能。其運作方式如Fig.3。

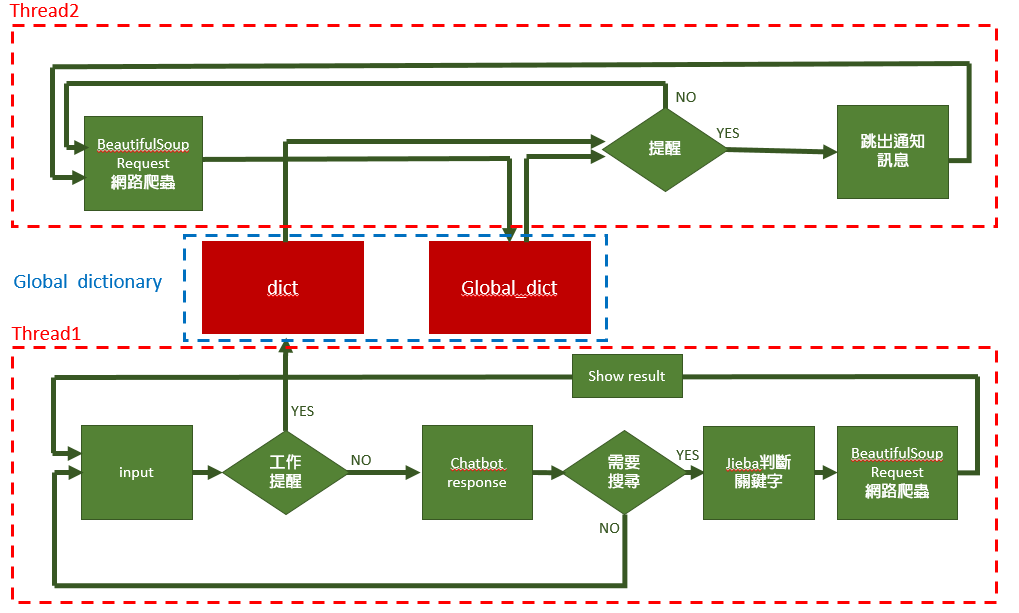
  
▲Fig.3

* + 1. 前綴詞典（Trie）  
       將文本中的「所有」分詞可能建成前綴詞典（Trie），例如「國家森林公園、國民、國小」時就會形成如Fig.4的前綴詞典。  
       ▲Fig.4  
       如此一來不論是節省空間或搜尋便利都佳。
    2. 有向無環圖（DAG）  
       以「國家森林公園」為例，若以「國」為頭可能是「國」、「國家」、「國家森林」、「國家森林公園」但若以「家」為起始則不可能為「家森林」之類的，將這種斷詞的結果以矩陣儲存即是有向無環圖。
    3. 最大機率路徑  
       自剛剛所得的有向無環圖中最大機率路徑者即為最佳的斷詞解。
    4. HMM模型和Viterbi算法  
       但那如果輸入的字詞字典裡不存在就改以HMM模型和Viterbi算法試著劃分，得到最終結果。
  1. Thread  
     Thread即為執行續，是電腦進行任務排程時的最小單位。主要是執行時可能執行續間彼此資源互相衝突，需要適當調配才能避免卡死，甚至增加效率。
     1. Single threading  
        單執行續並不會有排程問題，工作必須是一個做完才能接下一個，不能中途切換，例如DOS系統即是如此。
     2. Multithreading  
        多執行續即可在不同工作間互相切換(例如Windows系統)，為了達成此目的須將工作分為數階段，方便管理。
        1. 執行中（Running）
        2. 暫停（Suspended）  
           暫停者可以繼續執行（resume）
        3. 被阻擋（Blocked）
        4. 終止（Terminated）  
           無法再繼續執行
  2. Google Map API  
     Google Map API以平台分為四大類，Android、iOS、Web API和Web 服務 API，而我們採用的則是Web 服務 API中的Geocoding API。
     1. Geocoding API  
        Geocoding API主要功能有地理編碼及反向地理編碼，所謂的地理編碼即是將地址轉換為地理座標(經緯度)的程序，而反向則是將地理座標反轉為人看得懂的地址。其輸出會是一含地點ID的固定格式JSON檔。
     2. 地點ID與TAG  
        在GoogleMap的設計中，每個地點都會有一獨一無二的ID，就算是2家7-11便利商店亦不同；而回應的JSON中會包含對這個地點的屬性，例如7-11就會有store、food等。  
        (Fig.5為搜尋「國立高雄大學」的範例)  
        ▲Fig.5  
        節錄結果：” 'place\_id': 'ChIJO9VRShEPbjQR8lX-r-yB5uY’  
         'point\_of\_interest', 'university'”
     3. 理想與實際困境  
        理論上我們可以藉由Geocoding API中對搜尋結果的優先篩選機制挑出同性質距離最近、佳者，但電腦並無原生GPS，僅依靠IP位置只能確定到「城市」等級，當要搜尋「加油站、餐廳」等同質性高或重複的結果時不夠精確，但可以靠著登入同時登入於具GPS的裝置取得資料，在此情況下確實可達成與直接使用GoogleMap相同的成效。
* 實作

2.1程式架構

整體架構和原理如Fig.6，總共分成兩個Thread，中間以Global dictionary做為橋樑，使兩個Thread能藉由Global dictionary傳送參數。

1. Thread1：處理chatbot對話功能、搜尋功能與安排提醒事項至dict
2. Thread2:定時向國家時間與頻率標準實驗室獲取當前時間，並且將其與dict中的時間做比較，判斷是否該跳出提醒。
3. Dict:以要提醒的事項(Job)為key，且以提醒時間為value。
4. Global\_dict:以 value存取目前的標準時間。



▲Fig.6

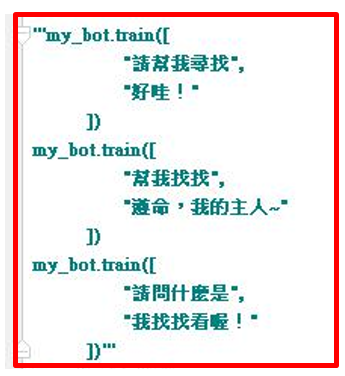
2.2程式碼

* + Thread1：

第一步先import套件，如下。



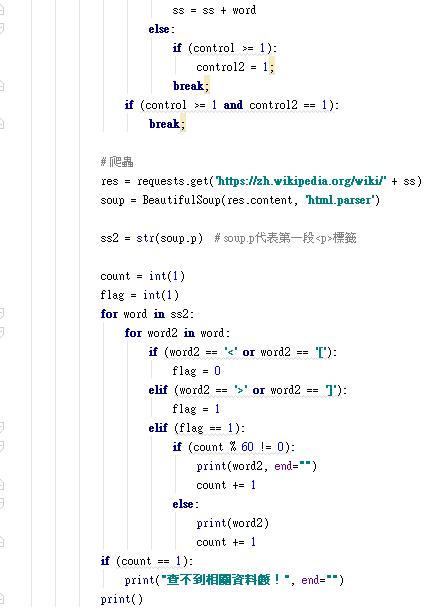
在開始前必須先訓練Chatbot特定對話，如Fig.7。



▲Fig.7

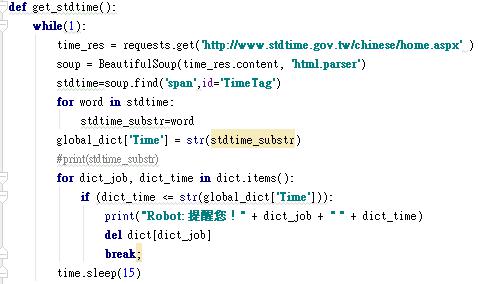
接著便進入thread1主要chatbot\_func()的部分：





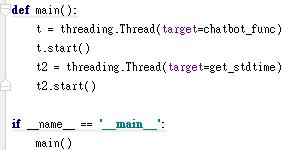
* + Thread2：

Thread2主要function get\_stdtime()，主要是做時間爬取與比對。



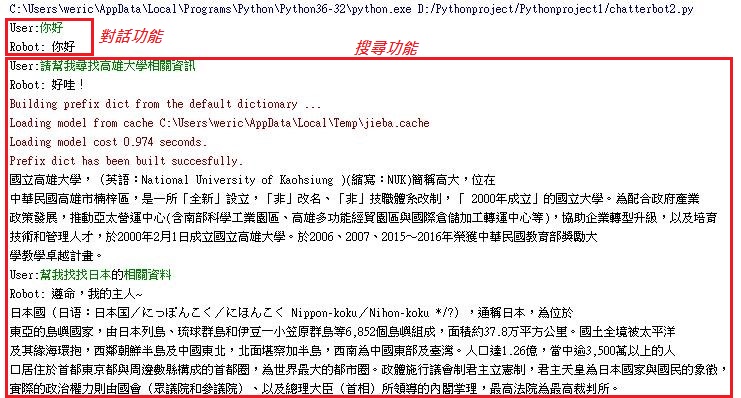
* + Main：

main建立兩個Threads執行上述提及的兩個function。

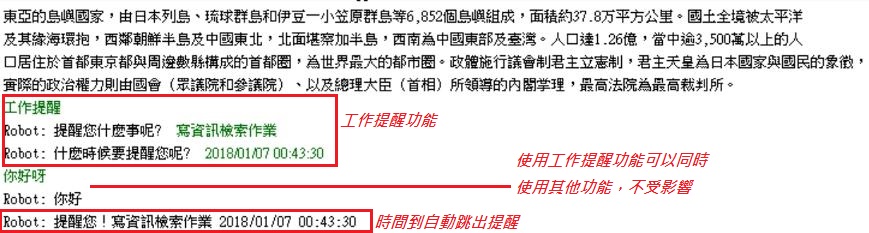


2.3執行結果

* + Result 1：對話功能、搜尋功能。



* + Result 2：工作提醒功能、對話功能。



* 資料參考：
* **ChatterBot：**

https://github.com/gunthercox/ChatterBot

* **Jieba ：**

https://github.com/fxsjy/jieba

* **BeautifulSoup：**

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

* **Thread-Multithreading：**https://en.wikipedia.org/wiki/Thread\_(computing)
* **Multithreading：**https://litotom.com/2017/01/12/java-threading-intro/
* **Thread States：**http://www.informit.com/articles/article.aspx?p=31093&seqNum=
* **Google Map API：**  
  http://maplink.global/en/google/
* **Google Map API：**https://developers.google.com/maps/?hl=zh-tw
* **Google Map API：**https://digit.hbs.org/submission/google-maps-the-most-expansive-data-machine/
* **Geolocation API：**https://developers.google.com/maps/documentation/geolocation/intro?hl=zh-tw
* **Geocoding API：**https://developers.google.com/maps/documentation/geocoding/start?hl=zh-tw
* **googlemaps/google-maps-services-python：**https://github.com/googlemaps/google-maps-services-python