

# Crawler API 使用教學

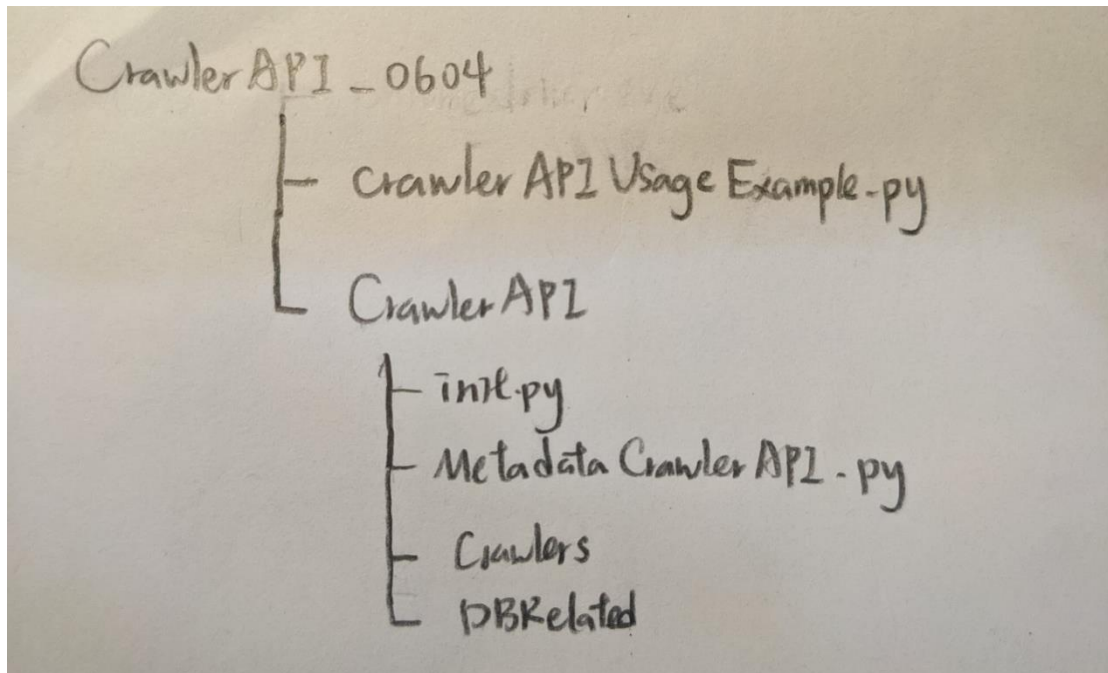
更新時間: 2022/06/04

- API 簡介

使用者會在各個史料平台上挑選資料，並把這些資料在平台上匯集下載。此 API 會以這些資料為目標進行爬蟲，以用來取得更多 `metadata` 的細節。此 API 會接受從各個史料平台上下載的、未經修改的檔案(通常為 `csv` 檔)，並根據裡頭的典藏號在各平台爬蟲，最後把爬來的內容更新到我們的 OD 資料庫。

- API 使用教學

API 的資料夾架構如圖一，在使用時，請讓主程式和 `CrawlerAPIUsageExample.py` 在同一個資料夾。



(圖一，API 資料夾架構圖)

此 API 以 `function call` 的方式使用。分成三個部分，分別是「判斷史料平台」、「提取典藏號」與「由典藏號爬蟲」，以下就分這三部分介紹。

- 判斷史料平台

使用者會從從史料平台下載需要的資料(通常為 `csv` 檔案)，此部分就是從此 `csv` 檔案判斷是由哪一個爬蟲來處理。

```
WebAbbr = IdentifyWebAbbr(FilePath)
```

- ◆ Input: `FilePath` 使用者下載下來的檔案路徑，資料型態為 `string`。
- ◆ Output: `WebAbbr` 網站的英文縮寫，資料型態為 `string`。若失敗，

會回傳 False。

#### ■ 提取典藏號

延續上個部分，從此 csv 檔案提取需要的全部典藏號，並一起輸出。

```
ids = InputToIDs(WebAbbr, FilePath, IsReturnJson)
```

- ◆ Input: WebAbbr 網站的英文縮寫，資料型態為 string。
- ◆ Input: FilePath 使用者下載下來的檔案路徑，資料型態為 string。
- ◆ Input: IsReturnJson 預設值為 True。填是，回傳的資料型態會是 json，否則為 dictionary。
- ◆ Output: ids 複數個典藏號，資料型態根據最後一個 input 有所不同。填是，為 json，否則為 dictionary。兩者的 key 皆為 0~長度-1，value 則為典藏號(string)。若失敗，會回傳 False。

#### ■ 由典藏號爬蟲

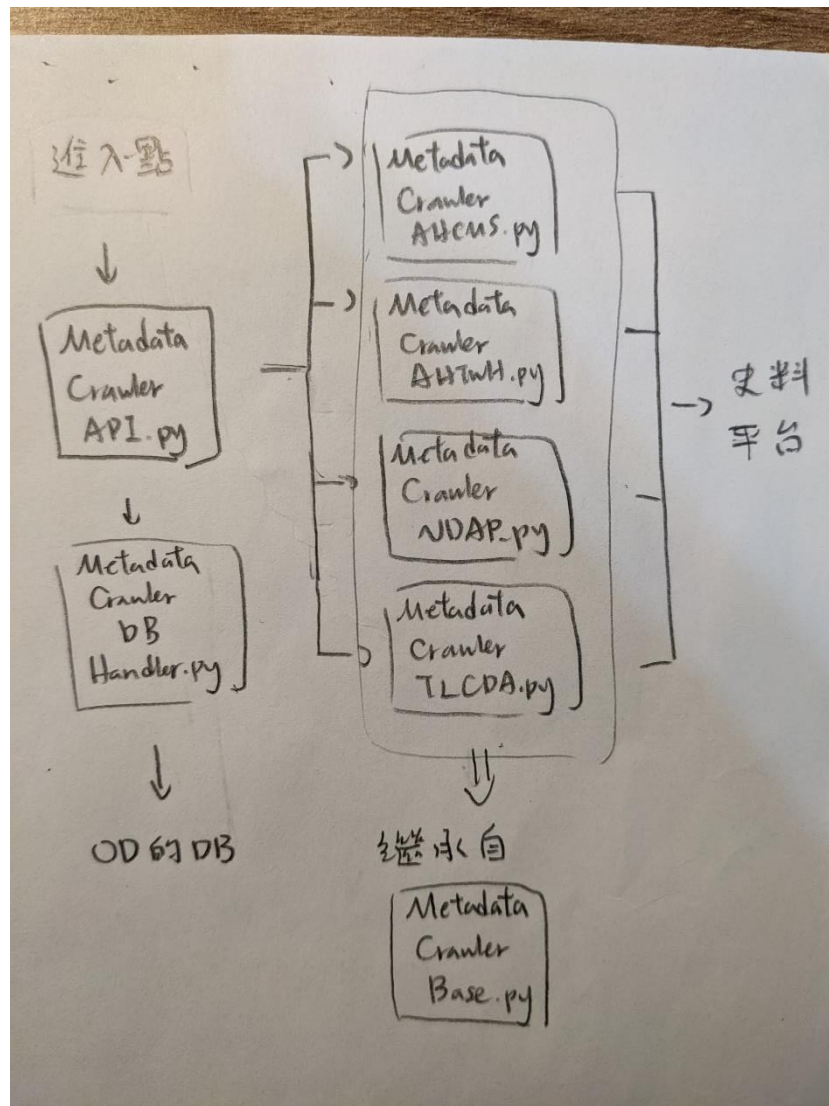
延續上個部分，把典藏號一個個放到此部分，就會執行爬蟲，並把所得資料存到資料庫。

```
isSuccess = Crawling(WebAbbr, id)
```

- ◆ Input: WebAbbr 網站的英文縮寫，資料型態為 string。
- ◆ Input: id 單一典藏號，資料型態為 string。
- ◆ Output: 是否成功，資料型態為 bool。

在 CrawlerAPIUsageExample.py 此檔案中，也有使用此 API 的方式可以參考。

- API 架構介紹  
架構圖如圖二。



(圖二，API 架構圖)

首先是使用 API 的進入點，也就是引用的 MetadataCrawlerAPI.py，他會呼叫 MetadataCrawlerDBHandler.py 來間接操作 OD 的資料庫，也會使用不同的 MetadataCrawlerXXX 來判斷史料平台、提取典藏號與爬蟲。而這些 MetadataCrawlerXXX 都是繼承自 MetadataCrawlerBase。

- API 新增「支援的史料系統」教學  
若想要新增支援的史料平台，就必須為了此平台寫客製化的程式，再把此程式合成到此 API 中。步驟如下
  - 新增一個 python 檔案到 CrawlerAPI/Crawlers/底下，並宣告一個 class 繼承 MetadataCrawlerBase。
  - 根據該史料平台覆寫 MetadataCrawlerBase 中的函式。

- ◆ **IdentifyCSV**: 判斷使用者下載下來的資料是否為此 **Crawler** 所處理的網站。回傳 **bool**。
- ◆ **InputToIDs**: 從使用者下載下來的資料，提取其中複數個典藏號。回傳 **list of string**。
- ◆ **IDCrawling**: 根據典藏號來爬蟲。回傳 **dictionary** 來存放資料。
- ◆ **DataCleaning**: 從網站直接爬取的資料，需要的資料清洗動作，像是去除空白鍵等等。回傳 **string**。
- ◆ **DataLinking**: 把上述的 **dictionary** 依照規範轉換成 **list**。此規範與 **OD** 資料庫的儲存方式有關，可以參考 **CrawlerAPI/DBRelated/MetadataFormat**。(舉例: 若爬下來的資料 **A** 屬於「原始時間記錄」欄位，則要把資料 **A** 擺放到 **list** 的第 7 格，也就是 **list[6]** 中。)

另外，**OD** 資料庫有些規定的事項要注意:

- 第 12 欄位(文件原系統頁面 **URL**)，要填入爬蟲的網頁 **URL**。
- 第 14 欄位(爬蟲 **Original**)，要填入爬到的原始資料(以 **string of json** 呈現)。
- 15~18 欄位(相關人員、相關地點、相關組織、關鍵詞)要以半形分號(;)來做分隔。

- 在 **CrawlerAPI/Crawlers/MetadataCrawlers** 中，添加新增的 **class**。

完成後即可，可以參考現有的四個 **class** 來寫，特別注意地方議會議事總庫 (**TLCDA**)，由於網站上能爬到的資料比使用者能下載下來的還要少，所以相關的 **crawler** 並沒有實際的爬蟲作用。