ManPower計算系統

使用JIRA Restful API 解決方案

Python 讀書會

Jerry & Van

版本紀錄

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 內容 | 發佈日期 | 作者 |
| v0.0.1 | ManPower軟體架構簡介 | 2024/09/17 | Van |
|  |  |  |  |

目錄

[第1章 概論 3](#_Toc177569910)

[第1.1節 Man Power簡介 3](#_Toc177569911)

[第1.2節 軟體開發環境 4](#_Toc177569912)

[第1.3節 Http 和 Restful API 6](#_Toc177569913)

[第1.3.1節 HTTP協定 6](#_Toc177569914)

[第1.3.2節 RESTful API 8](#_Toc177569915)

[第1.3.3節 Man power tool 環境 11](#_Toc177569916)

[第2章 Man power Tool軟體需求 12](#_Toc177569917)

[第3章 Man power Tool軟體架構簡介 13](#_Toc177569918)

[第3.1軟體設計架構 13](#_Toc177569919)

[第3.2節 軟體套件簡介 15](#_Toc177569920)

[第3.2.1節 Python PyQt5套件 15](#_Toc177569921)

[第3.2.2節 Python json套件 15](#_Toc177569922)

[第3.2.3節 python Requests套件 15](#_Toc177569923)

[第3.2.4節 python ini-parser套件 15](#_Toc177569924)

[第3.2.5節 python JIra套件 16](#_Toc177569925)

[第3.2.6節 python openpyxl套件 16](#_Toc177569926)

[第3.3節Man Power Tool 初始化執行流程 17](#_Toc177569927)

# 第1章 概論

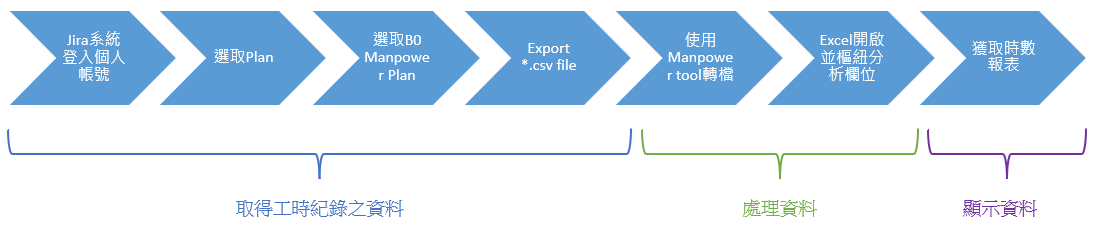
## 第1.1節 Man Power簡介

Man Power是一套AUO(友達光電有限公司)用來計算人力工時的簡易程式；主要方便觀察同仁每周工時，是否有**超過**或**不足**之情況，以便供同仁自行調整。該程式的運作流程如下：



從上面的流程架構圖來看，使用者必須**手動**進入AUO的JIRA系統，選擇自己的計畫(Plan)並且從JIRA中下載其CSV檔案(裡面有BU上所有成員在不同專案花費的時間)。下載完CSV檔案後，同仁便可以利用Man Power Tool，將下載的CSV檔案轉換成Excel文件。最後，同仁便可以**手動**改成較容易閱讀的形式(如**樞紐分析**)，並且得知自己與別人每周的工時。

因此，不難發現這一連串的動作中，很多動作仍然需要使用者使用「**手動**」的方式，個別完成。從這點出發，我們可以重新分析一下整個流程，如下圖：



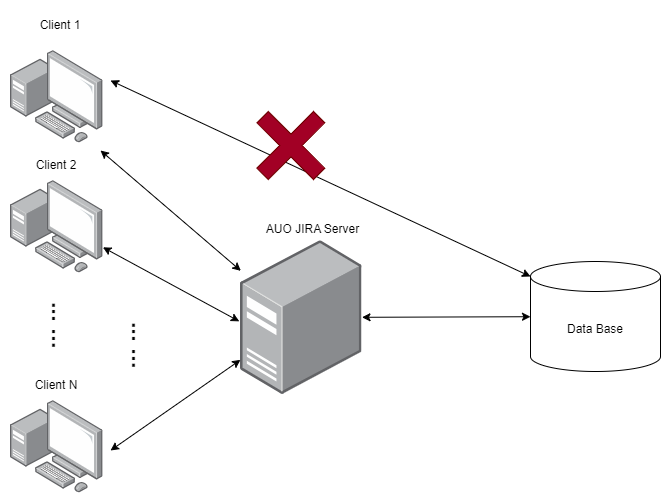
我們可以把整個流程分成三大部分，

1. 取得工時紀錄之資料；
2. 處理資料；
3. 顯示資料；

而本次Man Power Tool工時統計計算程式的「**重點核心**」也在於「**取得工時紀錄之資料**」，我們會採用網際網路常用的技術「**HTTP**」協定與「**RESTful API**」。

## 第1.2節 軟體開發環境

要實現Man Power Tool，我們首先必須要克服一個難題；那便是如何存取AUO JIRA伺服器，儲存在資料庫(Database)的資料呢？因為我們客戶端是無法直接取得AUO JIRA伺服器背後的資料庫伺服器，如下圖：



由於我們無法直接存取AUO JIRA伺服器的資料庫，因此只能採取「**間接**」存取的方式，進一步對AUO JIRA伺服器的資料庫進行更新以及讀取；也就是說，我們必須透過AUO JIRA伺服器，間接地更新以及取得資料庫的資料。

常見的方式有二，一是採用爬蟲的方式，透過剖析伺服器回應的網頁取得我們需要的數據，以及錄製網頁對於伺服器的請求，進而更新資料庫的數據；這個方法聽起來就非常的麻煩吧！另一個方法則是透過**伺服器提供**的HTTP API或者是RESTful API來間接存取AUO JIRA伺服器資料庫；而HTTP API和RESTful API的差異，在下一節會介紹。第二個方式，聽上去雖然簡單，但還是需要伺服器本身有提供，才能讓客戶端有辦法透過API來更新與取得資料庫的資料。

必須強調的是，不管是方法一還是二，兩者使用的技術都是基於HTTP(HyperText Transfer Protocol)協定，或說兩種方式都是利用HTTP協定傳輸資料，

只是採用的技巧有所不同。我們可以這樣理解，當伺服器本身沒有提供HTTP API時，我們便只能採用爬蟲的方式以及錄製網頁對伺服器發動HTTP請求去存取資料庫的資料。

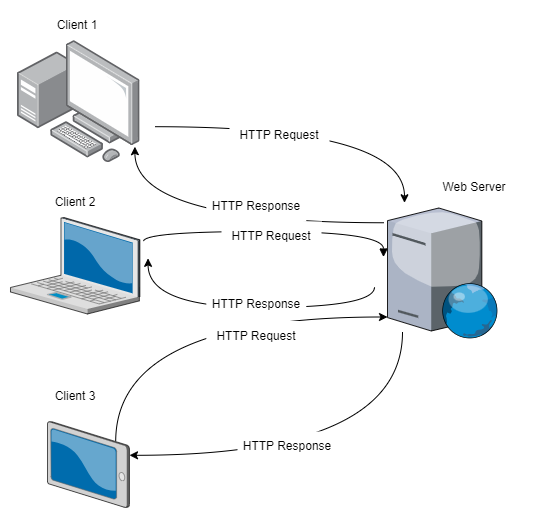
所幸，JIRA伺服器有提供HTTP RESTful API，我們可以利用HTTP RESTful API去存取AUO JIRA伺服器的資料庫。因此，在下一節，我們將簡介HTTP協定以及RESTful API，然後進一步勾勒出我們軟體開發的環境。

## 第1.3節 Http 和 Restful API

本節又分三小節，分別介紹HTTP協定、RESTful API與Man Power Tool開發環境的介紹。

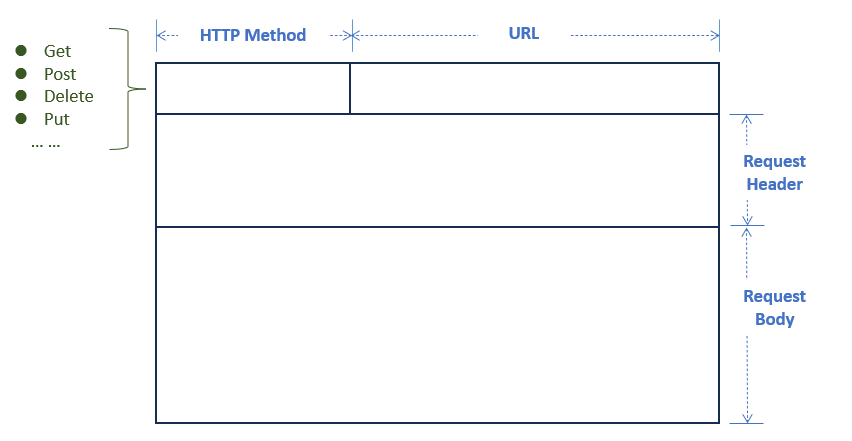
### 第1.3.1節 HTTP協定

在網際網路的世界中，網站伺服器(Web Server)主要使用的通訊協定便是應用非常廣泛的**HTTP**(**H**yper**T**ext **T**ransfer **P**rotocol)協定，**HTTP**協定採用**請求**(**Request**)/**回應**(**Response**)機制；**請求**的意思便是對網站伺服器請求「**資源**(**Resource**)」，而網站伺服器接收到該請求以後，如果該請求是伺服器允許的，便會給予對應的**回應**。如下圖：



HTTP協定的誕生並不難以理解，畢竟網站伺服器需要與各式各樣不同地客戶端(Client)**溝通**，交換資源；為了避免溝通發生歧異，便需要統一的規範與通訊格式，HTTP協定便定義了這一套客戶端與伺服器端之間溝通的方法。

既然HTTP協定如此重要，接下來，我們便需要簡單地了解一下HTTP請求與回應封包的格式；首先，我們先來認識一下HTTP請求封包，如下圖：



其中HTTP Method欄位常見的為Get、Post、Delete以及Put，用來決定對網站伺服器進行的操作。URL欄位則是填寫網站伺服器的位址。URL的格式如下：

**使用的協定**://**域名**(or **IP**)[**:port**]/**路徑**(**path**)

例如：

<http://192.168.1.223:8080/>test

<http://localhost:8080/test>

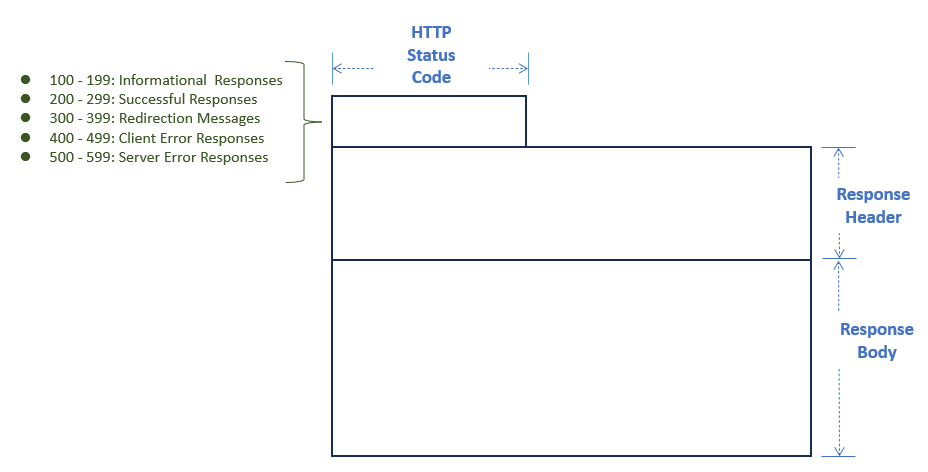
若沒有填寫埠號(Port)，預設為80。Request Header欄位可以讓使用者添加其額外的設定給予後端伺服器，如Cookie。當HTTP Method為Get時，參數是放置在符號，”?”後面，例如：

<http://localhost:8080/>[test?id=124&name=Judy](http://localhost:8080/test?id=124&name=Judy)

參數的格式為Key=Value；多個參數時用”&”符號連接。若HTTP Method為Put或Post時，參數不會放置在URL，而是放置在Request Body欄位之中，格式可以是Key=Value格式，

但若為RESTful風格，則為XML格式或是JSON格式。

HTTP Response的封包格式則如下圖：



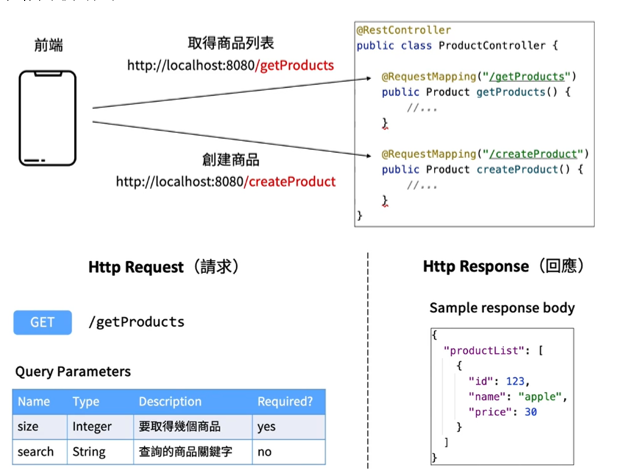
網頁伺服器收到請求後，會回應「請求的結果」，最直接明瞭的就是透過HTTP Status Code回應。我們不詳細介紹這個HTTP Response，因為畢竟這些回應與實際的伺服器實作有很大的不同。更多有關HTTP的知識可以自行去網路搜索，我們就只將HTTP協定的基礎介紹到這裡。

### 第1.3.2節 RESTful API

在介紹**RESTful API**之前，我們先得解釋一下什麼是**API**，所謂的API指的是：

* 1. API是指用工程師的方式去說明某個功能的使用方法；
  2. 目的：寫清楚這個功能要怎麼用；

而**HTTP API**，自然就是要告訴開發前端或客戶端的工程師，要如何發送HTTP請求才可以得到或說達到伺服器提供預期的服務或是資源。如下頁的圖，



上圖提供GET /getProducts，便代表前端或客戶端只需要發送HTTP Method為GET，URL路徑為getProducts，且必須在”?”後面帶size參數的HTTP請求，便可以取得指定size的產品資訊。而所謂的RESTful API便是符合REST風格的HTTP API。

何謂**REST**(**Representational State Transfer**)？按照英文的翻譯，即**表現層狀態轉換**；這是甚麼意思呢？在網際網路的世界中，所有的一切統稱叫作「**資源**」(Resource)，而所謂的表現層狀態，指的便是「資源」的狀態。那麼如何表現出資源的狀態呢? 一種直觀的想法便是利用HTTP Method去表達、表現現在想要如何處理「資源」。如下表：

|  |  |  |
| --- | --- | --- |
| HTTP Method | 對應的資料庫操作 | 說明 |
| POST | **C**reate (新增) | 新增一個資源 |
| GET | **R**ead (查詢) | 取得一個資源 |
| PUT | **U**pdate (修改) | 更新一個已經存在的資源 |
| DELETE | **D**elete (刪除) | 刪除一個資源 |

如上表，我們便可以使用HTTP Method的POST、GET、PUT以及DELETE去**表現**請求資源的**狀態**，要新增還是要刪除等等之**轉換**。

但是還有個問題，我們要對什麼「資源」進行處理呢？為了讓HTTP請求可以**表現**出來這個狀態，**REST**風格採用了URL路徑來描述資源之間的關係，如下表範例：

|  |  |
| --- | --- |
| Http method + URL路徑 | 說明 |
| GET/users | 取得所有user |
| GET/users/123 | 取得user id為123的user |
| GET/users/123/articles | 取得 user id為123的user所寫的所有文章 |
| GET/users/123/articles/456 | 取得user id為123的user所寫的、article id為456的文章 |
| GET/users/123/videos | 取得user id為123的user所錄的所有影片 |
| GET/users/123/videos/789 | 取得user id為123的user所錄的、video id為789的影片 |
| GET/users/100 | 取得user id為100的user |

最後，則是傳輸資料之間的格式能夠統一，因此REST風格要求資料必須以XML格式或是JSON格式。

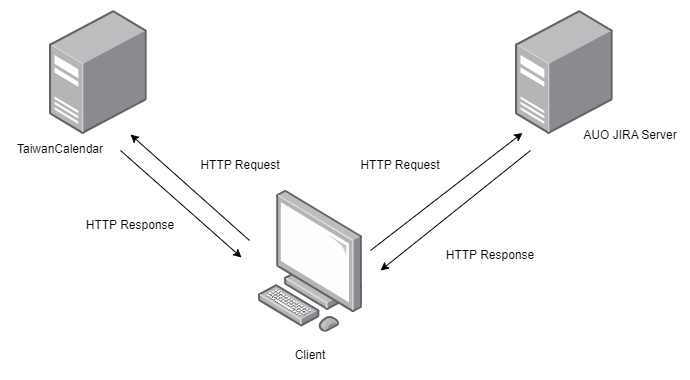
總結一下RESTful API的準則如下：

1. 使用**HTTP** method表示動作；
2. 使用**URL**路徑描述資源之間的階層關係；
3. response body返回json或是xml格式；

RESTful API沒有標準答案，永遠記住RESTful API實現的目的便是為了簡化團隊溝通的成本，我們本小節的介紹旨在助於我們能更好理解JIRA API設計，更多RESTful API的知識還是可以透過各式各樣的網路文件得到更進一步的認識。

### 第1.3.3節 Man power tool 環境

Man Power Tool的環境如下圖：



由於我們需要取得台灣的行事曆，所以Man Power Tool所在的客戶端，除了需要向AUO JIRA伺服器取得與更新資料外，也需要向Taiwan Calendar的伺服器取得目前年份的行事曆。

# 第2章 Man power Tool軟體需求

# 第3章 Man power Tool軟體架構簡介

根據前兩章的介紹，我們將在本章中架構一個軟體結構去完成軟體需求。在第1章中，我們分析了整個Man Power Tool系統，可以分為：

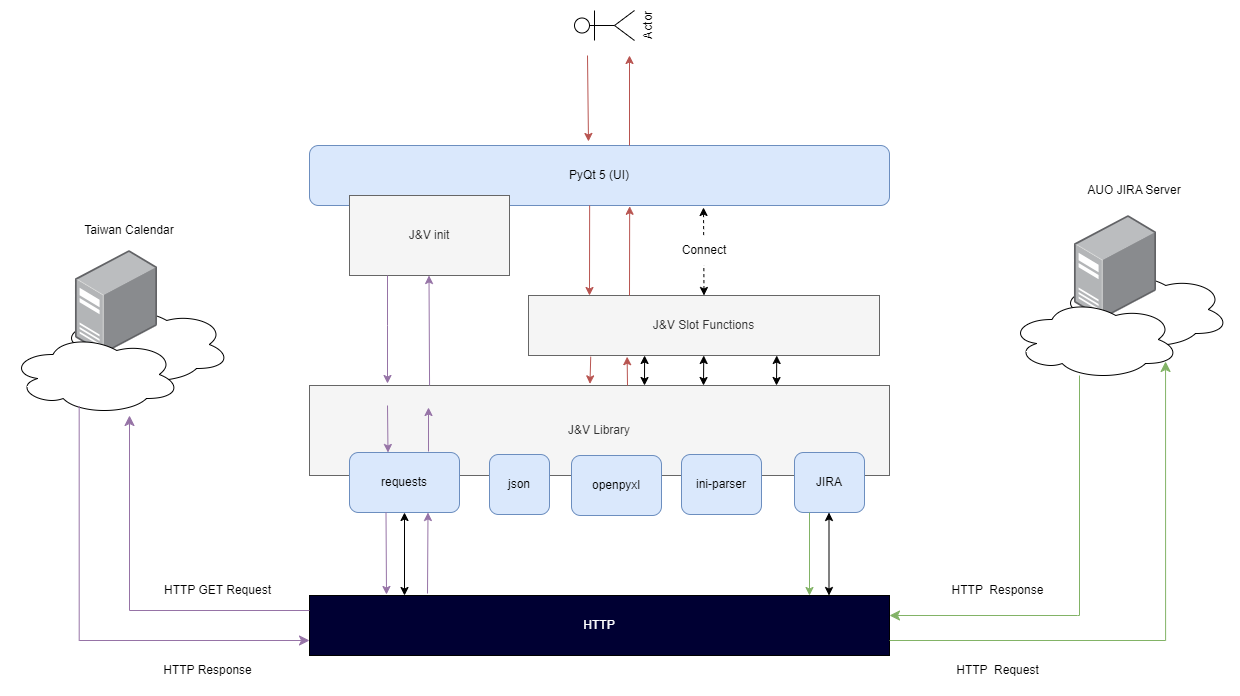
1. 取得工時紀錄之資料；
2. 處理資料；
3. 顯示資料；

同時我們也介紹了我們如何使用RESTful API去「取得工時紀錄之資料」。而在第2章，我們也介紹我們「處理資料」和「顯示資料」的需求。而在本章中，我們會把這些需求以具體的方式呈現出來。

## 第3.1軟體設計架構

由於空間不足，我們將Man Power Tool的軟體設計架構圖放置在下一頁，如下頁圖，不難發現Man Power Tool的軟體架構是由一些淺藍色方塊、一些淺灰白色方塊以及一個黑色方塊「HTTP」所構成；這裡，值得我們注意的只有圖中的淺藍色方塊以及淺灰白色方塊；淺藍色方塊為Python開放原始碼的模組，而淺灰白色方塊為我們自行封裝的模組，主要的目的是橋接使用者介面、Man Power Tool的運作演算法，以及封裝Python開放原始碼模組。

首先，我們會利用Python的PyQt5套件打造Man Power Tool使用者介面(User Interface, UI)，然後在UI與Python開放原始碼模組之前，隔一層由我們所撰寫的程式模組去銜接這兩層。如此，Man Power Tool的軟體架構便組建完成。更多的內容，我們會在之後的章節做更完整的介紹。



## 第3.2節 軟體套件簡介

本節延續上節，針對架構圖中淺藍色方塊的Python開放原始碼套件做一個簡介。

### 第3.2.1節 Python PyQt5套件

PyQt5套件是基於Qt 5的Python開發套件；換言之，我們可以利用該套件的Qt Designer開發使用者介面，而後轉成Python程式碼。轉換後，便可以將之整合至我們的Man Power Tool之中。

### 第3.2.2節 Python json套件

由於大多數網站伺服器，特別是符合RESTful API的網站伺服器，伺服器回應的HTTP Response的資料格式大多都是JSON格式。因此為了方便開發，我們引入了這個Json套件，讓我們方便處理JSON字串。

### 第3.2.3節 python Requests套件

由於我們必須對網站伺服器發起HTTP請求，因此需要這個Requests套件去發送HTTP API或是HTTP RESTful API對網站伺服器進行增(Create)、查(Read)、改(Update)、刪(Delete) ，即CRUD。

### 第3.2.4節 python ini-parser套件

關於程式的一些預設值，或是設定值，為了能夠實現更佳彈性的設計，我們通常會把這些設定值獨立於程式之外，成為設定檔案。最常見的設定檔案便為ini設定檔案，而ini-parser套件便是用來剖析這類設定檔案的套件。

### 第3.2.5節 python JIra套件

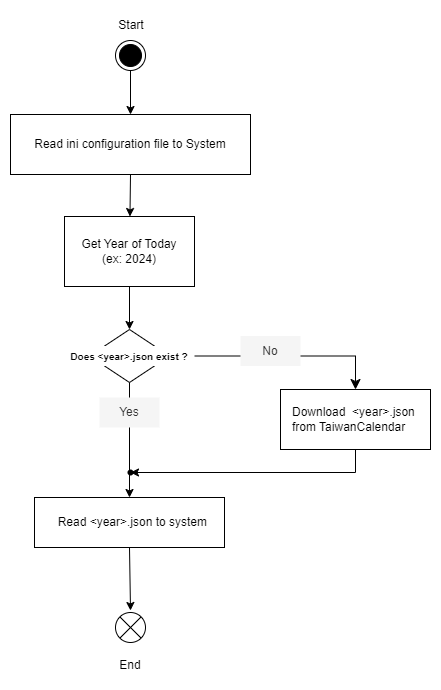
除了使用Requests套件可以呼叫HTTP API以及HTTP RESTful API，針對JIRA的伺服器，Python也有專門的jira套件可以對JIRA伺服器進行操作。

### 第3.2.6節 python openpyxl套件

除了取得以及處理資料外，Man Power Tool還需要將其整理成EXCEL，openpyxl便是方便我們操作EXCEL的套件。

## 第3.3節Man Power Tool 初始化執行流程

這一節將介紹Man Power Tool初始化執行流程；當Man Power Tool程式被使用者點擊後，程式便會開始運作，而在運作一開始便會執行初始化之動作。如下圖：



首先程式一開始會在指定目錄下讀取ini設定檔案，並且把這些設定從檔案載入至記憶體；而後系統需要取得今年台灣的行事曆，該行事曆可以至如下網址下載：

<https://github.com/ruyut/TaiwanCalendar>

因此我們必須取得今天日期的年份，並且根據年份至該網站取得以年份命名的JSON檔案，以今年2024年為例，便需要取得2024.json檔案。但問題來了，這個檔案理論上應該只需要下載一次便可，而不需要每次Man Power Tool被點擊都被執行一次才對；於是我們在取得今年的年份後，便可以至指定的位置去判斷<year>.json的檔案是否存在？

如果該檔案不存在，便要至TaiwanCalendar網站下載<year>.json檔案；而如果檔案存在，那便讀取<year>.json檔案，並且選擇一個適當的資料構將之載入記憶體，以便後續程式演算法的使用與呈現。