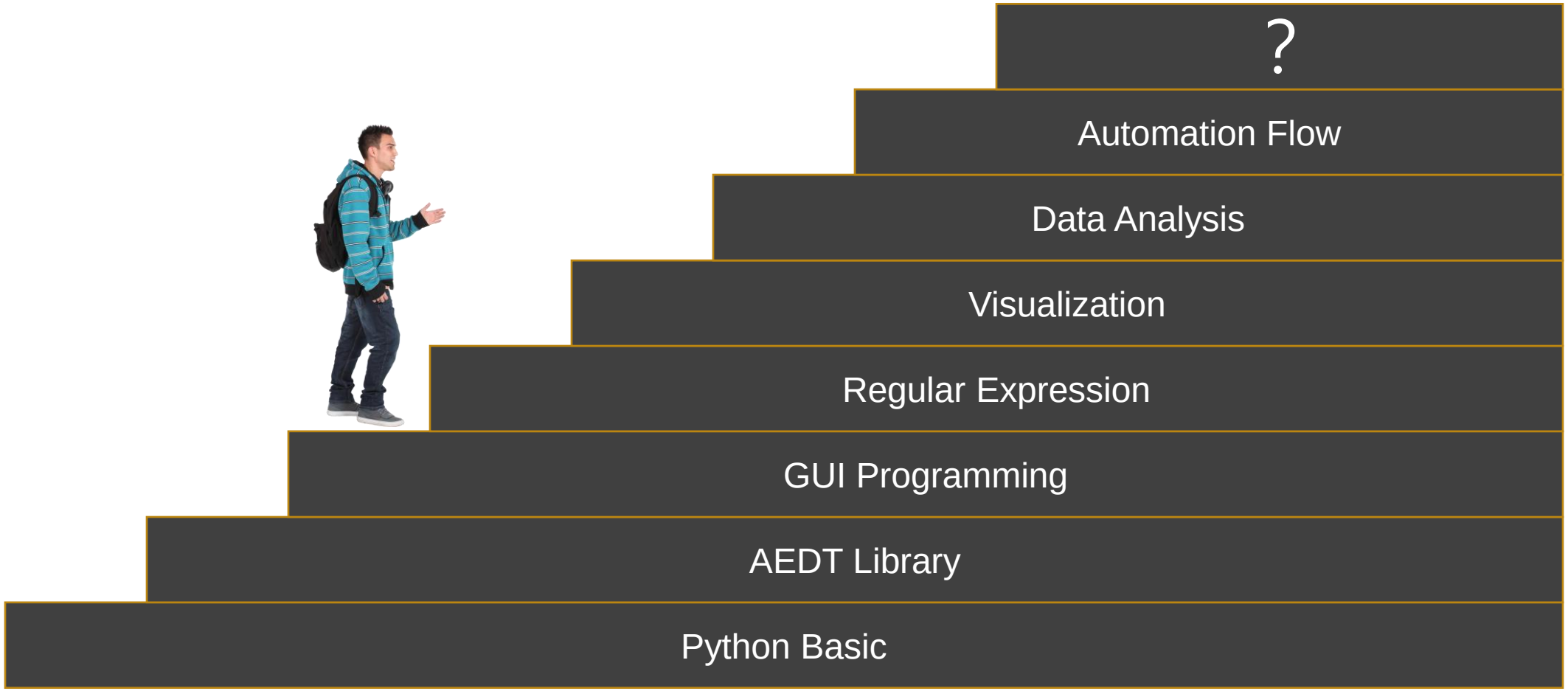
Week 8-模擬自動化進階

AEDT自動化學習路徑

Ansys

Simulation Automation Research Path



如何發展大程式碼？

個人經驗，對自動化程式而言，100行以內算是小程式。中型程式則是100-1000行，超過1000行可以算是大程式。這裡的行數不包含空白行，註解。開發小程式不太需要考慮結構的問題。中型程式就必須以函數，甚至是物件導向來組織程式碼。大型程式會需要模組化。程式規模越大，除錯困難度也越高，因此需要考慮單元測試。

- 使用者溝通
- 模組化
- 保護及加密
- 單元測試
- 版本管理

溝通的重要性

如果程式使用者跟程式開發者是同一個人，由於不需要溝通成本，開發速度是最有效率的。多數的時候，使用者與開發者為不同人，此時溝通的成本無可避免。很多開發者疏忽了溝通的重要性，等到程式開發完成才發現程式不符使用者期望，導致整個專案最後以失敗告終。

這種問題其實是可以避免的，最好的方法是開發過程中持續的與使用者討論，也可以透過原型展示來輔助討論的進行。也就是先完成操作介面的設計，如果符合客戶的期望，再進行核心代碼的開發。

溝通的困難

相信大家都有經驗在跟家人解釋工作的內容時，極其困難。但是跟大學同系所的同學解釋時就簡單的多，跟其他部門的同事只要幾句話就可以說明清楚。這個差異來自知識重疊的多寡。知識重疊越多，溝通成本越小。

因此在評估自動化程式開發的時候，除了必須考慮技術的掌握度，更重要的是跟使用者的契合度，幫自己同部門的同事開發軟體的困難度通常要比幫其他部門開發小的多，除了技術問題，彼此之間的默契與信賴讓溝通可以有更有效的進行也是很重要的原因。

解析Python模組(Module)和套件(Package)的概念

- 當我們在開發大型應用程式時，如果沒有適當的組織程式碼，除了會降低開發的效率外，也不易於維護，所以模組(Module)化就顯得相當的重要，讓程式碼能夠透過引用的方式來重複使用，提升重用性(Reusable)。

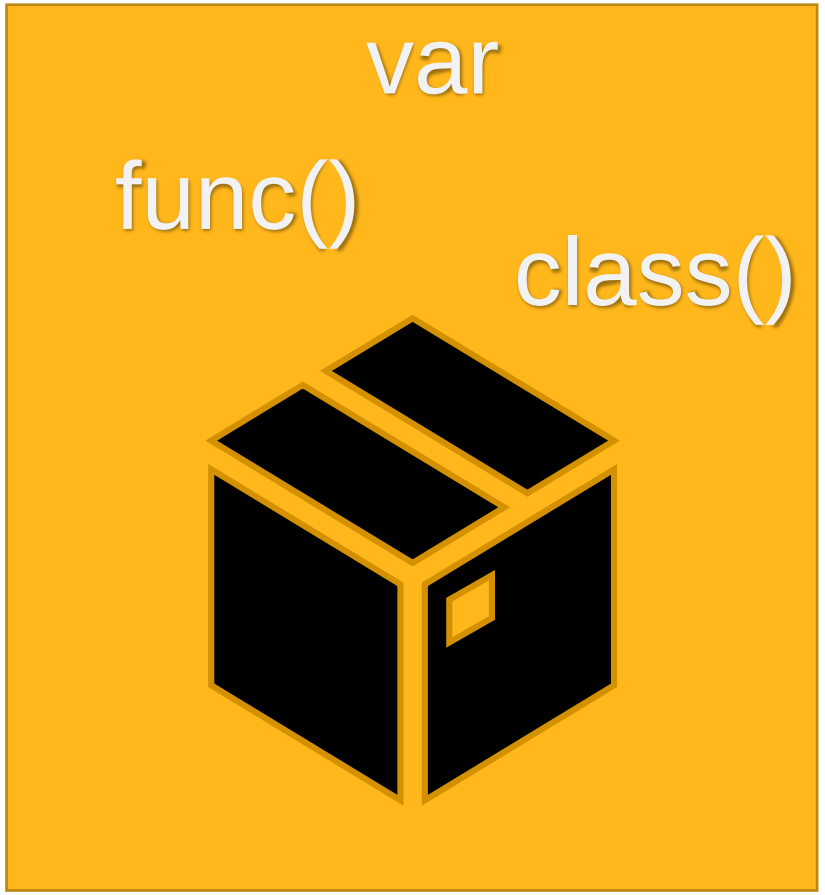
- 但是隨著專案模組(Module)的增加，將難以管理及問題的追蹤，這時候就能將模組(Module)打包成套件(Package)，利用其階層式的結構來彈性規劃模組(Module)。

自定義模組(Module)

只要你建立了一個原始碼檔案 modu.py，你就建立了一個模組 modu，原始碼主檔名就是模組名稱。

import modu 陳述句會在相同目錄下尋找 modu.py，如果沒找到，則會試著尋找在 sys.path 中遞迴地尋找 modu.py，如果還是沒有，則會引發 ImportError 例外。

模組提供了名稱空間。模組中的變數、函式與類別，基本上需透過模組的名稱空間來取得。在 Python 中，import、import as 與 from import 是陳述句，可以出現在程式中陳述句可出現的任何位置。



將IronPython模組編譯成.dll檔(僅限Windows)

PATH=%PATH%;C:\Program Files\AnsysEM\AnsysEM20.2\Win64\common\IronPython

ipy64 "C:/Program Files/AnsysEM/AnsysEM20.2/Win64/common/IronPython/Tools/Scripts/pyc.py" "D:/demo/mymodule.py"

pause

```
D:\demo>ipy64 "C:/Program Files/AnsysEM/AnsysEM20.2/Win64/common/IronPython/Tools/Scripts/pyc.py" "D:/demo/mymodule.py"
Input Files:
        D:/demo/mymodule.py
Output:
        mymodule
Target:
        Dll
Platform:
        ILOnly
Machine:
        I386

Compiling...
Saved to mymodule

D:\demo>pause
請按任意鍵繼續 . . .
```

連結.dll檔，匯入模組並測試模組函式

```
import os
os.chdir(os.path.dirname(__file__))

import clr
clr.AddReference("mymodule.dll")

import mymodule

x = mymodule.myadd(1,3)
AddWarningMessage(str(x))
```

Cython[WIKI]

- **Cython**是結合了Python和C的語法的一種語言，可以簡單的認為就是給Python加上了靜態類型後的語法，使用者可以維持大部分的Python語法，而不需要大幅度調整主要的程式邏輯與演算法。但由於會直接編譯為二進位程序，所以性能較Python會有很大提升。
- Cython被大量運用在CPython函式庫的撰寫，以取得較高的執行效能。Cython將CPython代碼轉譯成 C 或 C++ 語法後，自動包裝上函式呼叫界面生成 .pyx 後綴的執行檔，即可當成普通的函式庫。其性能一般遜於原生的 C/C++ 函式庫，但由於 CPython 語法的易用性可以縮短開發時間。Cython 也可以用於編譯以 C/C++ 為 CPython 撰寫的函式庫。
- 目前 Cython 可以在 Windows, MacOS 與 Linux 上使用，可以編譯 2.6, 2.7 與 3.3 至 3.7 版本的 CPython 語法。

Cython執行步驟

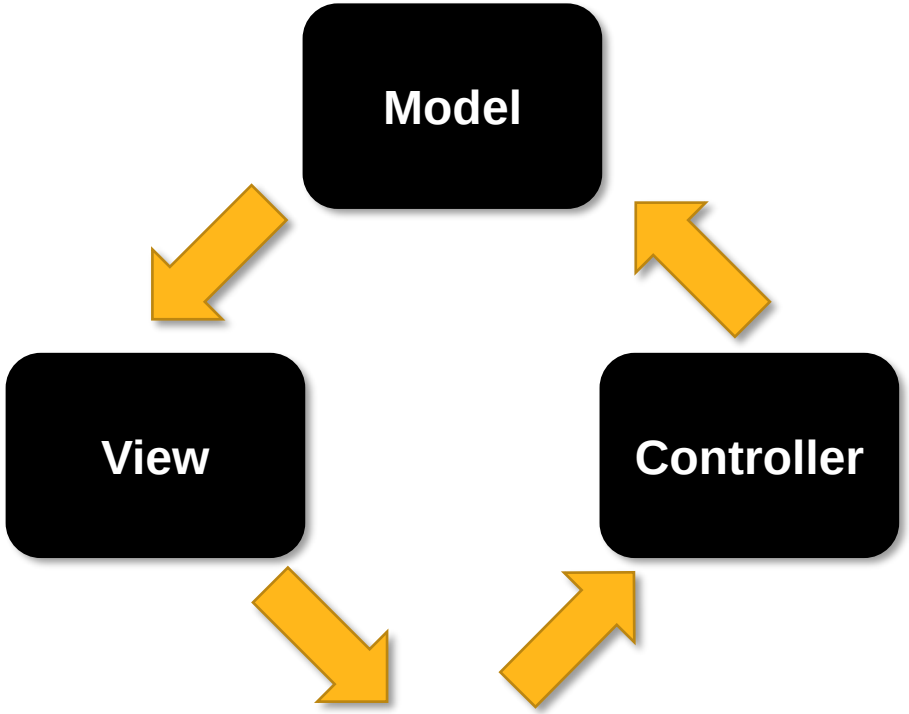
電腦需安裝C++編譯器(Windows: Visual Studio)

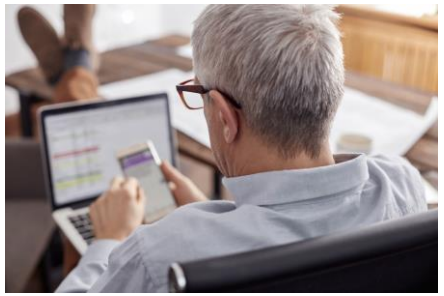
1. 準備main.py, module.pyx, setup.py, compile.bat
2. 執行compile.bat
3. 輸出 module.c 及.pyd
4. 測試.pyd

名稱	狀態	修改日期	類型	大小
build	✓	8/17/2020 7:08 A...	檔案資料夾	
module.pyx	✓	8/17/2020 7:05 A...	PYX 檔案	1 KB
setup.py	✓	8/17/2020 7:07 A...	PY 檔案	1 KB
compile.bat	✓	8/17/2020 7:08 A...	Windows 批次檔案	1 KB
module.c	✓	8/17/2020 7:08 A...	C Source	103 KB
module.cp37-win_amd64.pyd	✓	8/17/2020 7:08 A...	Python Extension...	21 KB
main.py	✓	8/17/2020 7:09 A...	PY 檔案	1 KB

GUI 程式架構

- MVC模式（Model–view–controller）是軟體工程中的一種軟體架構模式，把軟體系統分為三個基本部分：模型、視圖和控制器。
  - 模型（Model） - 程式設計師編寫程式應有的功能（實現演算法等等）、資料庫專家進行資料管理和資料庫設計(可以實現具體的功能)。
  - 視圖（View） - 介面設計人員進行圖形介面設計。
  - 控制器（Controller）- 負責轉發請求，對請求進行處理。





單元測試(Unit Testing)[WIKI]

在電腦編程中，單元測試又稱為模組測試，是針對程式模組（軟體設計的最小單位）來進行正確性檢驗的測試工作。程式單元是應用的最小可測試部件。在程序化編程中，一個單元就是單個程式、函式、過程等；對於物件導向程式設計，最小單元就是方法，包括基礎類別（超類）、抽象類、或者衍生類別（子類）中的方法。

通常來說，程式設計師每修改一次程式就會進行最少一次單元測試，在編寫程式的過程中前後很可能要進行多次單元測試，以證實程式達到軟體規格書要求的工作目標，沒有程式錯誤；雖然單元測試不是必須的，但也不壞，這牽涉到專案管理的政策決定。

善用if __name__ == '__main__':敘述

軟體版本控制

軟體設計師常會利用版本控制來追蹤、維護原始碼、檔案以及設定檔等等的改動，並且提供控制這些改動控制權的程式。在最簡單的情況下，軟體設計師可以自己保留一個程式的許多不同版本，並且為它們做適當的編號。這種簡單的方法已被用在很多大型的軟體專案中。該方法雖然可行，但不夠有效率。除了必須同時維護很多幾乎一樣的原始碼備份外；而且極度依賴軟體設計師的自我修養與開發紀律，但這卻常是導致錯誤發生的原因。 [WIKI]

最受歡迎的版本控制技術與網站為git及GitHub，參考：

Git與GitHub介紹，軟體版本控制基本教學

進階研究課題

Python

- Object Oriented Programming
- Itertools, collections,…
- Module and Package
- Numpy & Scipy
- Unit Test
- Matplotlib
- Cython
- …

AEDT

- Cross Platform
- SIwave
- WPF
- ODB
- ACT
- UDP (User Defined Primitive)
- UDO (User Defined Output)
- …

NumPy[WIKI]

NumPy引入了多維陣列以及可以直接有效率地操作多維陣列的函式與運算子。因此在NumPy上只要能被表示為針對陣列或矩陣運算的演算法，其執行效率幾乎都可以與編譯過的等效C語言程式碼一樣快。

NumPy提供了與MATLAB相似的功能與操作方式，因為兩者皆為直譯語言，並且都可以讓使用者在針對陣列或矩陣運算時提供較純量運算更快的效能。兩者相較之下，MATLAB提供了大量的擴充工具箱（例如Simulink）；而NumPy則是根基於Python這個更現代、完整並且開放原始碼的程式語言之上。

SciPy [WIKI]

SciPy是一個開源的Python演算法庫和數學工具包。

SciPy包含的模組有最佳化、線性代數、積分、插值、特殊函數、快速傅立葉變換、訊號處理和圖像處理、常微分方程式求解和其他科學與工程中常用的計算。與其功能相類似的軟體還有MATLAB、GNU Octave和Scilab。

成果以及開發經驗分享

分享原則

- 內容可以包括
  - 自動化程式展示
  - 學習及開發心得
  - 未來可能開發方向
  - 課程建議


Ansys