**DD deck tool**

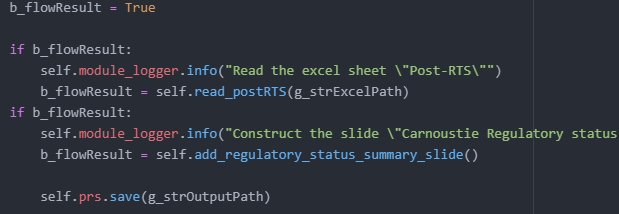
**Report.py:**

流程主控程式，由類別PPTXREPORT 來實現。Code內容多為呼叫函式，函式implement部分主要由另外兩隻程式負責PPTX\_FEATURE.py, EXCEL\_FEATURE.py。

**class PPTXREPORT:**

* **startFlow(self):**

主控流程函式，主要呼叫同個class內的函式。

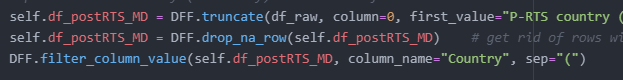


如 read\_postRTS跟add\_regulatory\_status\_slide。由函式名稱可以發現，函式內容多為一個大方向並具有針對性，不易泛化。像add\_regulatory\_status\_slide就是建立第一頁客戶的投影片。

* **read\_postRTS(self, excel\_path):**

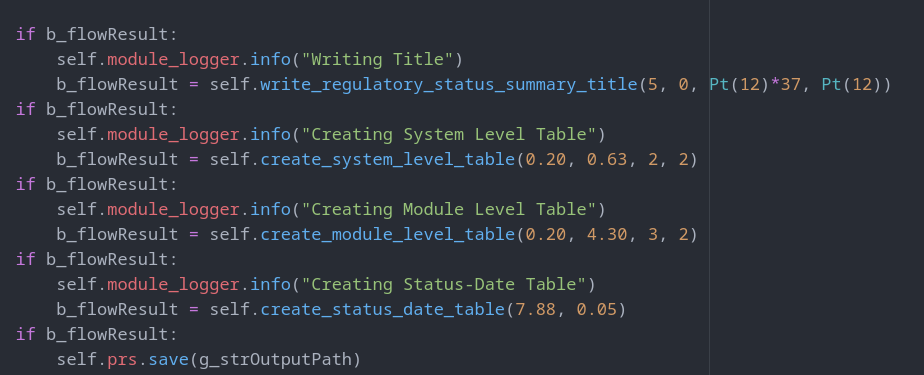
此函式目的為讀取客戶excel的”Post-RTS” sheet的內容並存為dataframe。

會呼叫自訂義靜態類別如下: 



* **add\_regulatory\_status\_slide(self):**

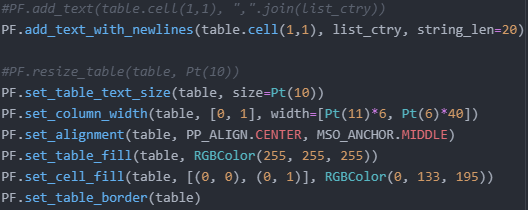
此函式主要實現客戶所提投影片第一頁內容。由於許多東西是要進行手刻，所以內部又細分幾個函式，並進行流程控管。如create\_system\_level\_table。



* **create\_system\_level\_table(row, col, left, top, slide\_idx=0):**

此函數主要為建立第一頁投影的”system level table”， 建立資料及調整style，slide\_idx預設為0表示第一頁。此函式多會呼叫自訂義靜態類別:





* **create\_module\_level\_table(row, col, left, top, slide\_idx=0):**

如上述函數，主要建立”module level table”，概念雷同。

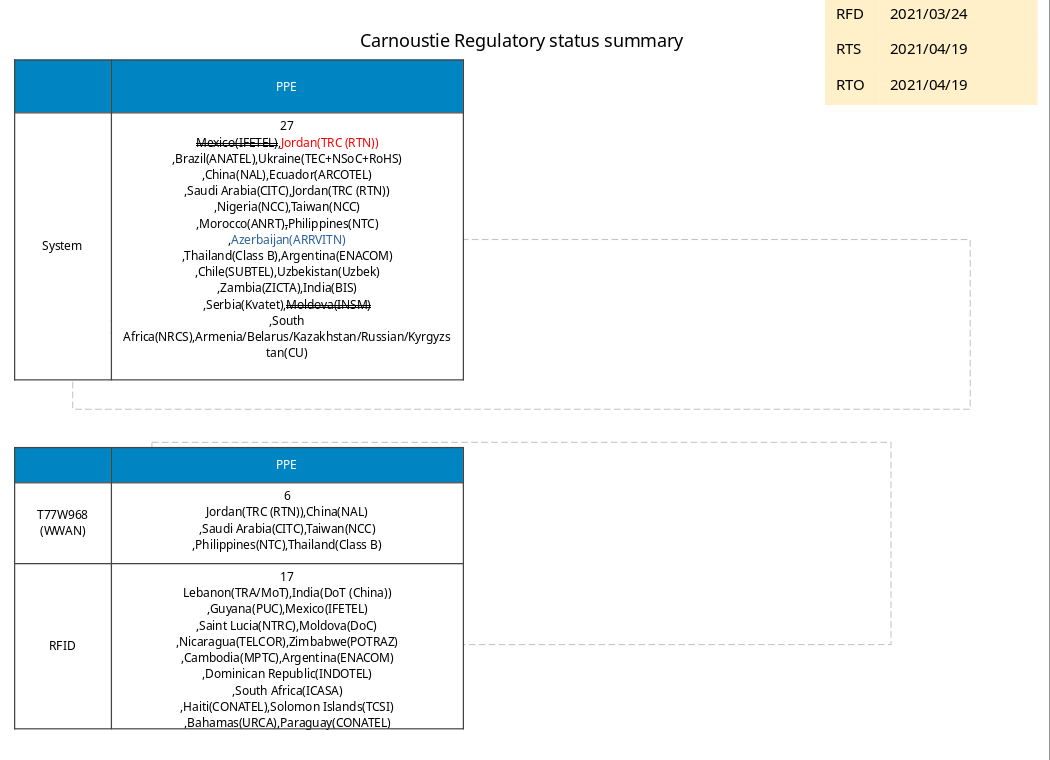
* **create\_status\_date\_table(self, left, top, slide\_idx=0):**

建立投影片右上角的日期狀態table.

* **write\_regulatory\_status\_summary\_title(self, left, top, width, height, slide\_idx=0):**

建立第一頁投影片標題並置中（投影片總長10inches)

上述4個函數執行完：



**EXCEL\_FEATURE.py**

主要處理excel資料的parsing，資料處離完畢丟回主控流程程式。

**Class DataFrameFeature:**

* **filter\_column\_value(df, \*, column\_name, sep):**

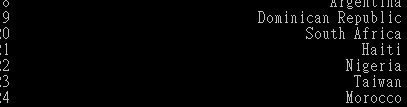
根據sep參數，對dataframe某一欄位的字串做處理(拿掉sep及sep後的字串)。

Ex. DFF.filter\_column\_value(self.df\_postRTS\_MD, column\_name="Country", sep="(")

執行前:



執行後:

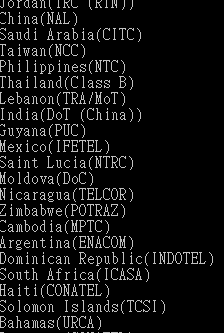


* **get\_country\_set(df, \*, category):**

依據category將Country欄位字串做分類並加上Certificate欄位的字串。最後回傳國家總數(不重複)及國家列表。

Ex. total\_ctry, list\_ctry = DFF.get\_country\_set(self.df\_postRTS\_MD, category="Host")

執行後得到的國家列表:

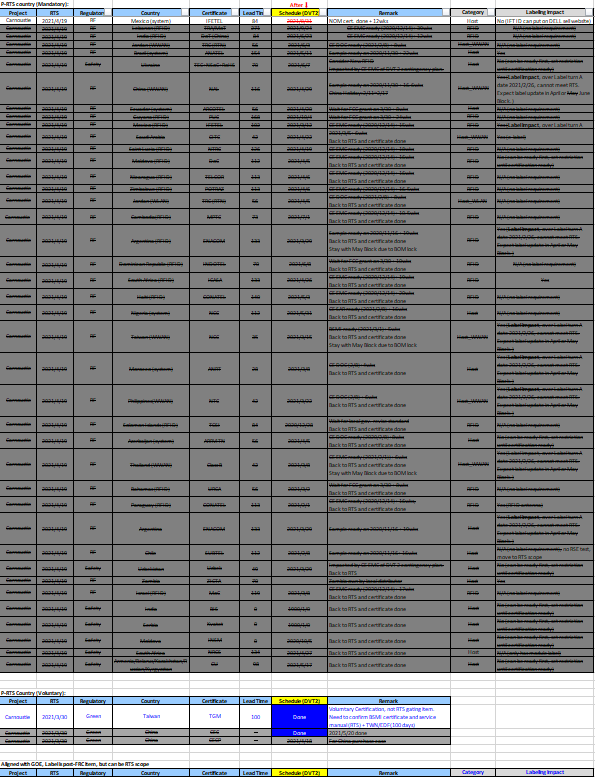
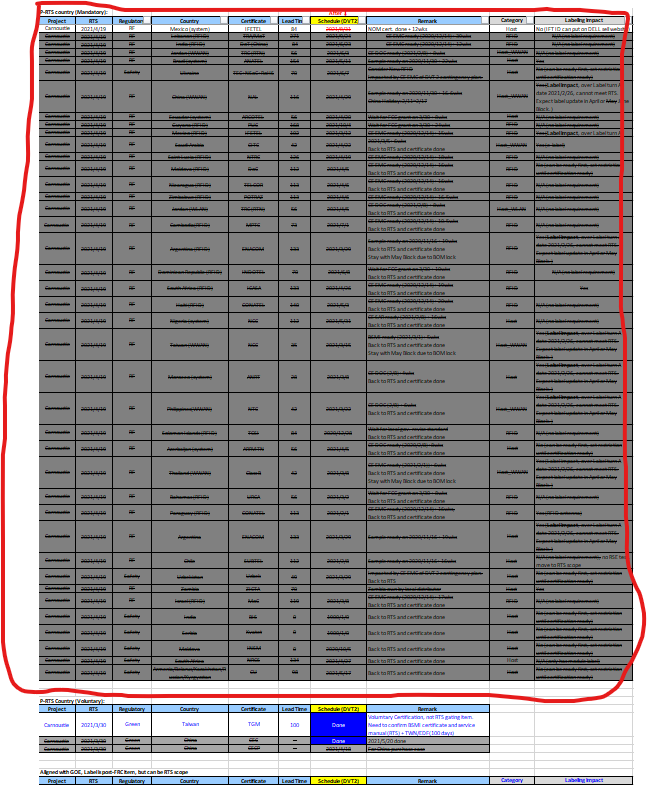


* **truncate(df, \*, column, first\_value, last\_value):**

根據first\_value跟last\_value來做分割dataframe的分割，擷取中間內容。

Ex. self.df\_postRTS\_MD = DFF.truncate(df\_raw, column=0, first\_value="P-RTS country (Mandatory):", last\_value="P-RTS Country (Voluntary):")

從左圖全部內容中抽取P-RTS country (Mandatory)的內容(如右圖所框)

* **drop\_na\_row(df):**

將dataframe中所有column階為Nan(空值)的row刪掉。

**PPTX\_FEATURE.py**

根據主控程式的需求，建立或處理PPT物件(如table value，table style)。

**class PresentationFeature:**

* **add\_text\_with\_newlines(cell, list\_ctry, \*, string\_len):**

加入文字到cell中並根據string\_len給定的值做換行。

Ex. PF.add\_text\_with\_newlines(table.cell(1,1), list\_ctry, string\_len=20)

將國家列表中的所有國家加入cell(1,1)，如果該行的字串長度以達到或超過20，換行。

如下(PPT table cell(1,1) 內容):



* **set\_table\_text\_size(table, \*, size):**

設定table中所有文字大小。

Ex. PF.set\_table\_text\_size(table, size=Pt(10))

* **set\_column\_width(table, list\_col\_idx, \*, width):**

可分別設定不同column的寬度。

Ex. PF.set\_column\_width(table, [0, 1], width=[Pt(11)\*6, Pt(6)\*40])

column 0寬度設為Pt(11)\*6, column 1 寬度設為Pt(6)\*40。可見下表兩欄位寬度不同。

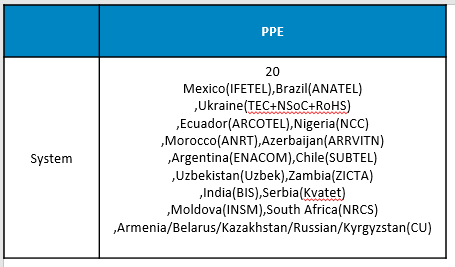


* **set\_alignment(table, horizen\_type, vertical\_type):**

設定對齊模式，分為水平及垂直。

Ex PF.set\_alignment(table, PP\_ALIGN.CENTER, MSO\_ANCHOR.MIDDLE)

設定為水平置中，垂直置中。如下表:



* **set\_table\_fill(table, RGBcolor):**

設定table背景顏色。

Ex. PF.set\_table\_fill(table, RGBColor(255, 255, 255))

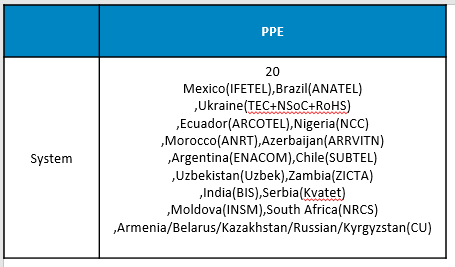
將所有cell背景設為白底(255,255,255)

* **set\_cell\_fill(table, list\_cell\_coord, RGBcolor):**

設定特定cell的背景顏色。

Ex. PF.set\_cell\_fill(table, [(0, 0), (0, 1)], RGBColor(0, 133, 195))

將cell(0,0),(0,1)的背景設為深藍，如下圖第一列



* **set\_table\_border(cls, table, border\_color="444444", border\_width='12700'):**

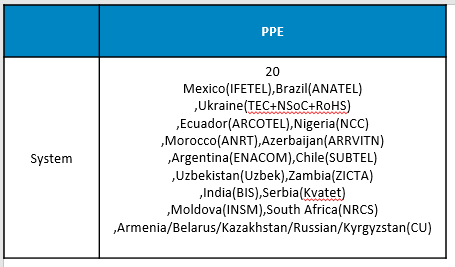
設定table border的style(顏色及寬度)，預設顏色為黑色，寬度為Pt(1)。此函式會呼叫該類別的SubElement函式。

Ex. PF.set\_table\_border(table)

執行前:



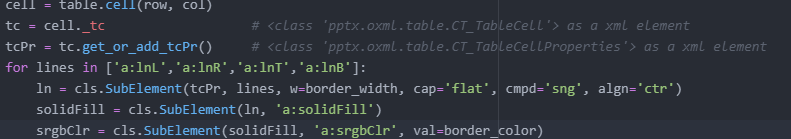
執行後:



* **SubElement(parent, tagname, \*\*kwargs):**

新增xml element以及其屬性到PPT的XML中。

Ex. set\_table\_border 函式中:

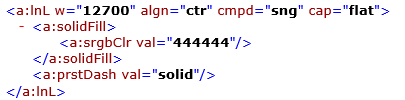


ln為一個element，tag名稱為a:lnL(其中的lines)，其父element為變數tcPr(也是一個element)。後方參數為我們此element設定的屬性。

solidFill同為一個element，tag名稱為a:solidFill，父element為ln，不帶屬性。

srgbClr，tag名稱為a:srgbClr，父solidFill，放方設定顏色屬性。

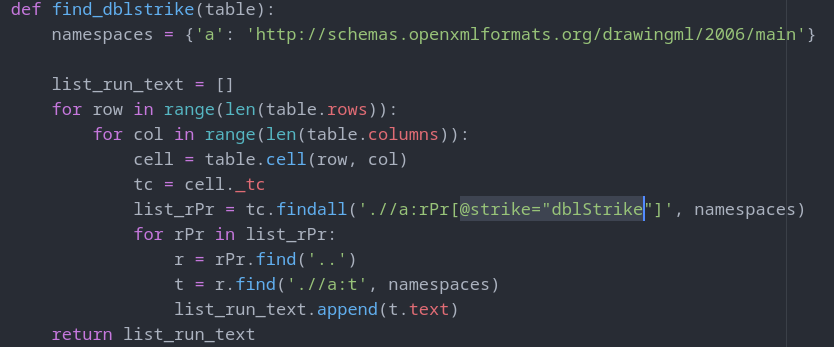
XML範例如下:



* **find\_dblstrike(table):**

從table的xml中去找有雙刪除線的文字

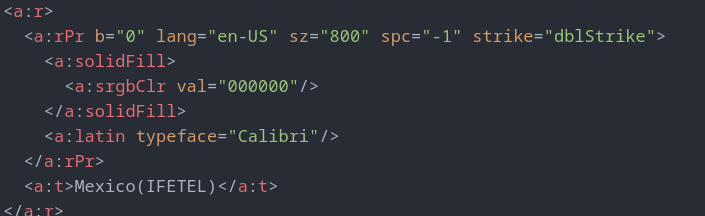
Ex. find\_dblstrike函式中：



從每個cell的xml element(cell.\_tc)中去尋找對應雙刪除的xml element(rPr),

找到之後找到對應的文字, rPr的父節點r, 將r的文字 append到list,回傳list.

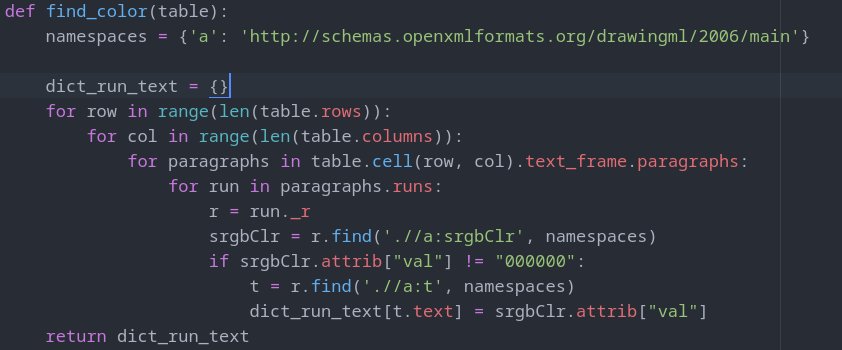
可參考下圖xml範例：



* **find\_color(table):**

從table的xml中去找除了黑色（hex:000000）以外的文字

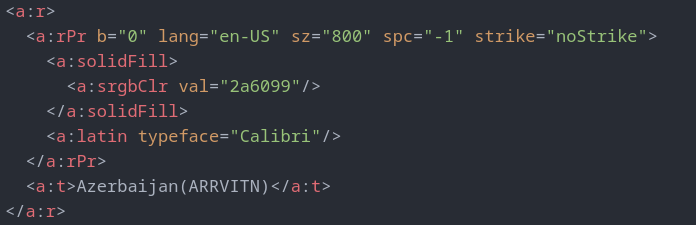
Ex.find\_color(table)函式內：



從每個cell的run element(run.\_r)中去找對應顏色的xml element(srgbClr),

只要不是黑色就挑出來,將文字及顏色update到dict,回傳dict.

可參考下圖xml：

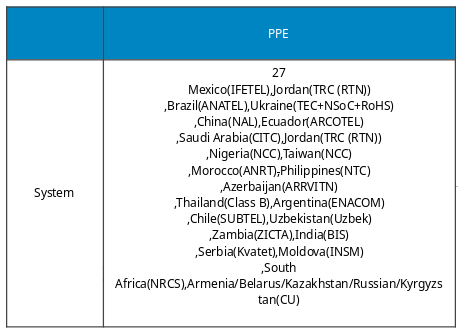


* **set\_dblstrike(cls, table, list\_run\_text):**

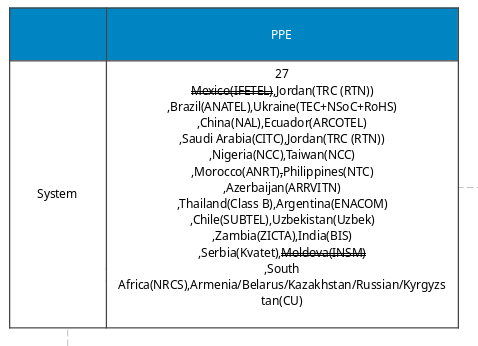
給定一個文字list,透過xml將table中對應的文字劃上雙刪除線

Ex.PF.set\_dblstrike(table, list\_dblstrike\_run), list內容為”Mexico(IFETEL)”, “Moldova(INSM)”

執行前:



執行後：

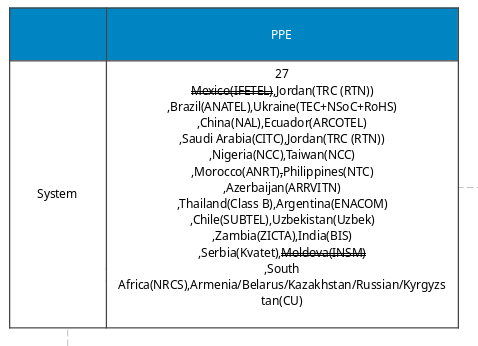


* **set\_color(cls, table, dict\_run\_text):**

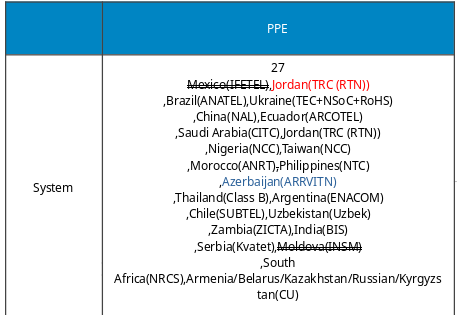
給定一個dict,內容需為{文字：色碼} ,透過xml將對應的文字填上顏色.

Ex.PF.set\_color(table, dict\_color\_text), dict內容為{“Jordan(TRC (RTN))”:”ff0000", “Azerbaijan(ARRVITN)”:”2a6099"}

執行前：



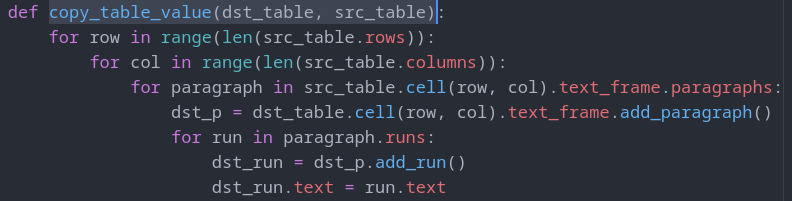
執行後：



* **copy\_table\_value(dst\_table, src\_table)：**

複製來源table內的值到目的table.

Ex.copy\_table\_value(dst\_table, src\_table)函式內：



由於諸多數值的style屬性都存於xml的run element中,因此複製時以run為單位做數值複製,以利之後設定或複製style屬性時可對應到一致的xml run element.

* **print\_table\_xml(cls, table, table\_name, path=os.getcwd()):**

將table的xml內容輸出為檔案,方便做xml屬性的查看.

Ex.PF.print\_table\_xml(dict\_table["System"], table\_name="System"):

將table: dict\_table[“System”]的xml內容輸出為System\_table.xml