

Appendix

A

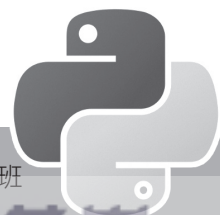
擴充實戰： Word 文件處理

Office 的文件是日常生活工作常用到的文件格式，其中 Word 格式的檔案更是重要。

Python 語言可透過 Win32com 模組對 Microsoft Office 文件進行存取，而 Python 已內含 Win32com 模組，不需另外安裝。若要使用 Win32com 模組處理 Microsoft Office 文件，電腦必須已安裝 Microsoft Office 軟體。

本章利用 Win32com 模組製作兩個實際應用：自動建立整個月份的營養午餐菜單 Word 文件，及自動取得指定目錄中所有 Word 文件 (包含子目錄)，並對所有 Word 檔案進行置換文字功能。

Python 初學特訓班





A.1 以 Win32com 模組處理 Word 文件

Python 語言可透過 Win32com 模組對 Microsoft Office 文件進行存取，而 Python 已內含 Win32com 模組，不需另外安裝。

要使用 Win32com 模組處理 Microsoft Office 文件，電腦必須已安裝 Microsoft Office 軟體。本章僅說明處理 Word 文件的方法。

A.1.1 建立新檔及儲存檔案

Win32com 模組不需安裝，直接匯入即可使用，此處匯入 Win32com 模組的 client 模組：

```
from win32com import client
```

處理 Word 文件需先建立 Word 應用。語法為：

```
應用變數 = client.gencache.EnsureDispatch('Word.Application')
```

例如建立 word 應用變數：

```
word = client.gencache.EnsureDispatch('Word.Application')
```

word 應用最常使用 Visible、DisplayAlerts 屬性及 Documents 方法：

- **Visible**：1 表示要顯示 Word 畫面，0 則不顯示 Word 畫面。
- **DisplayAlerts**：1 表示要顯示 Word 警告訊息，0 則不顯示 Word 警告訊息。
- **Documents**：操作 Word 文件，如開啟檔案、儲存檔案等。

建立新檔案

Win32com 模組建立新檔案是使用 Documents 的 Add 方法，語法為：

```
文件變數 = 應用變數.Documents.Add()
```

例如建立新文件的變數名稱為 doc：

```
doc = word.Documents.Add()
```

文件內容的位置可使用文件變數的 Range 方法設定，語法為：

```
範圍變數 = 文件變數.Range(起始位置, 結束位置)
```

起始位置及結束位置為整數，表示字元數目，例如設定文件前 10 個字元為 range1 變數：

```
range1 = doc.Range(0,9)
```

將文字加入 Word 文件有兩個方法：範圍變數的 InsertAfter 方法是將文字加在範圍變數結束位置的後方，語法為：

```
範圍變數.InsertAfter(文字)
```

使用 InsertAfter 方法插入文字後，會自動將範圍變數的結束位置移到插入文字最後方，再使用 InsertAfter 方法時會將文字加在原插入文字的後方。

範圍變數的 InsertBefore 方法是將文字加在範圍變數起始位置的前方，語法為：

```
範圍變數.InsertBefore(文字)
```

使用 InsertBefore 方法插入文字後不會改變範圍變數的起始位置，再使用 InsertBefore 方法時會將文字加在原插入文字的前方。

儲存檔案

本章範例 Word 檔案位於 media 資料夾中，由於 Win32com 模組存取檔案不能使用相對路徑，必須先取得 Python 程式檔所在路徑，語法為：

```
import os  
路徑變數 = os.path.dirname(__file__)
```

儲存 Word 檔案的語法為：

```
文件變數.SaveAs(檔案路徑)
```

例如將檔案存於 media 資料夾，檔名為 <test1.docx>：

```
cpath = os.path.dirname(__file__)  
doc.SaveAs(cpath + "\\media\\test1.docx")
```

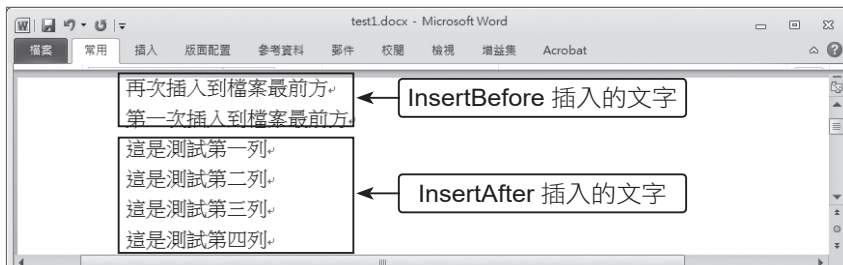
處理完 Word 文件，通常會在程式最後關閉 Word 檔案及應用，以免佔用系統資源，語法為：

```
文件變數.Close()  
應用變數.Quit()
```



範例：建立 Word 檔案並存檔

以 Win32com 模組建立 Word 檔案，加入內容後存檔。



程式碼：chA\newdocx1.py

```
1 import os
2 from win32com import client
3 word = client.gencache.EnsureDispatch('Word.Application')
4 word.Visible = 1
5 word.DisplayAlerts = 0
6 doc = word.Documents.Add()
7 range1 = doc.Range(0,0) # 檔案起始處
8 range1.InsertAfter(" 這是測試第一列 \n 這是測試第二列 \n")
9 range1.InsertAfter(" 這是測試第三列 \n 這是測試第四列 \n")
10 range1.InsertBefore(" 第一次插入到檔案最前方 \n")
11 range1.InsertBefore(" 再次插入到檔案最前方 \n")
12 cpath = os.path.dirname(__file__)
13 doc.SaveAs(cpath + "\\media\\test1.docx")
14 #doc.Close()
15 #word.Quit()
```

程式說明

- 4-5 顯示 Word 畫面但不顯示警告訊息。
- 6 建立 Word 檔案。
- 7 新檔案沒有內容，將範圍設為檔案起始處。
- 8-9 使用 **InsertAfter** 加入文字兩次，注意第二次加入的文字在第一次加入文字的後面。
- 10-11 使用 **InsertBefore** 加入文字兩次，注意第二次加入的文字在第一次加入文字的前面。
- 12-13 先取得 Python 檔案路徑後存檔。
- 14-15 此兩列註解可讓使用者觀看 Word 檔案內容，請手動關閉 Word。

A.1.2 開啟檔案及顯示檔案內容

建立檔案後，接著要開啟檔案並且讀取檔案內容。

Win32com 模組開啟檔案是使用 Documents 的 Open 方法，語法為：

```
文件變數 = 應用變數.Documents.Open(檔案路徑)
```

例如開啟上一節建立 <test1.docx>，文件變數名稱為 doc：

```
doc = word.Documents.Open(cpath + "\\media\\test1.docx")
```

取得文件內容的方法是先取得所有段落，再以迴圈顯示段落內容。

取得所有段落的語法為：

```
段落集合變數 = 文件變數.Paragraphs
```

例如取得所有段落存於 paragraphs 變數中：

```
paragraphs = doc.Paragraphs
```

取得段落內容的語法為：

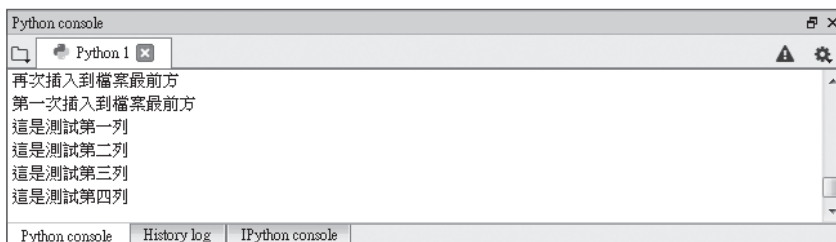
```
段落集合元素.Range.Text
```

使用迴圈顯示段落內容的程式為：

```
for p in paragraphs:  
    text = p.Range.Text.strip()  
    print(text)
```

第 2 列程式「p.Range.Text」為取得段落內容，「strip()」方法移除段落頭尾的空白字元。

<readdocx2.py> 執行結果為：



```
Python console  
再次插入到檔案最前方  
第一次插入到檔案最前方  
這是測試第一列  
這是測試第二列  
這是測試第三列  
這是測試第四列  
Python console History log IPython console
```



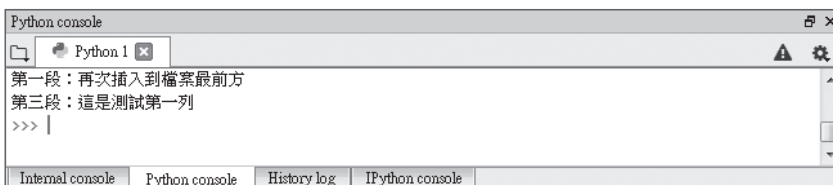
此種方法可以取得任何一個段落的内容，所以可依使用者需求顯示檔案部分內容。取得一個段落内容的語法為：

```
段落集合變數 (n).Range.Text
```

注意「n」的值是由 1 開始編號，即 1 表示第一段，2 表示第二段，依此類推。例如下面程式會顯示第一段及第三段的内容：(<readdocx3.py>)

```
paragraphs = doc.Paragraphs
print(" 第一段：" + paragraphs(1).Range.Text.strip())
print(" 第三段：" + paragraphs(3).Range.Text.strip())
```

執行結果為：



A.1.3 範圍格式設定

Win32com 模組可為特定範圍内容指定格式，較常用的格式有標題、排列方式及字型格式。許多格式是使用常數表示，需先匯入 constants 模組：

```
from win32com.client import constants
```

設定標題格式的語法為：

```
範圍變數.Style = constants.標題常數
```

標題常數 的常數值有 wdStyleHeading1 到 wdStyleHeading9，包括字體尺寸、粗體等設定，wdStyleHeading1 字體最大，wdStyleHeading9 字體最小。

設定排列方式格式的語法為：

```
範圍變數.ParagraphFormat.Alignment = constants.排列常數
```

排列常數 的常數值有：

- **wdAlignParagraphRight**：靠右對齊。
- **wdAlignParagraphLeft**：靠左對齊。
- **wdAlignParagraphCenter**：置中對齊。

- **wdAlignParagraphJustify**：左右散開對齊。

設定字型格式的語法為：

範圍變數 .Style.Font. 字型屬性 = 設定值

字型屬性 常用者為：

- **Name**：字型名稱，如 Arial、新細明體等。
- **Color**：字型顏色，以六個十六進位數值組成，如「0xFF0000」為藍色、「0x00FF0000」為綠色、「0x0000FF」為紅色等。
- **Bold**：「1」表示粗體，「0」表示正常。
- **Italic**：「1」表示斜體，「0」表示正常。
- **Underline**：「1」表示加底線，「0」表示正常。
- **Shadow**：「1」表示加陰影，「0」表示正常。
- **Outline**：「1」表示加外框，「0」表示正常。
- **Size**：字體尺寸。注意字體尺寸是設定整個文件字體尺寸，只要文件中沒有個別設定過字體尺寸的內容都會變為此設定尺寸（如 wdStyleHeading1 到 wdStyleHeading9 包含字體尺寸設定，不受此設定影響）。

範例：Word 文件格式設定

對 Word 文件做各種格式設定，注意段落三設定字體尺寸為 10，除了段落一及段落二外，其餘段落字體尺寸皆為 10。由於 Word 文件內容有修改，關閉 Word 軟體時會顯示詢問是否存檔的對話方塊。





程式碼：chA\styledocx1.py

```
1 import os
2 from win32com import client
3 from win32com.client import constants
4 .....略
5
6 8 doc = word.Documents.Open(cpath + "\\media\\clipgraph.docx")
7 9 paragraphs = doc.Paragraphs
8 10 range1 = paragraphs(1).Range # 第一段落
9 11 range1.Style = constants.wdStyleHeading1
10 12 range1.Style.Font.Name = "標楷體"
11 13 range1.Style.Font.Color = 0xFF0000 # 藍色
12 14 range1.Style.Font.Bold = 1
13 15
14 16 range2 = paragraphs(2).Range # 第二段落
15 17 range2.Style = constants.wdStyleHeading3
16 18 range2.ParagraphFormat.Alignment = constants.wdAlignParagraphRight
17 19
18 20 range3 = paragraphs(3).Range # 第三段落
19 21 range3.Style.Font.Size = "10"
20 22 #doc.Close()
21 23 #word.Quit()
```

程式說明

- 3 匯入 **constants** 模組。
- 8 開啟 <media> 資料夾的 <clipgraph.docx> 檔。
- 9 取得所有段落。
- 10-14 設定段落一為第一級標題、標楷體、藍色、粗體。
- 16-17 設定段落二為第三級標題、靠右對齊。
- 20-21 設定段落三字體尺寸為 10，除了段落一及二設定過標題格式外，文件中其餘內容字體尺寸都會設定為 10。



預設值為儲存檔案

上面範例如果將 22 及 23 列程式前「#」移除，程式會自動關閉文件及 Word 軟體，不會顯示詢問是否存檔的對話方塊，而是直接儲存修改後的內容，原始文件將會被覆蓋

A.1.4 表格處理

表格是 Word 文件常用的項目，Win32com 提供許多表格處理命令。

建立新表格的語法為：

```
表格變數 = 文件變數.Tables.Add(範圍變數, 列數, 行數)
```

例如在 range1 範圍變數處新增一個 3 列 4 行的表格 table：

```
table = doc.Tables.Add(range1, 3, 4)
```

設定儲存格內容的語法有兩種，第一種是使用 Cell 方法，語法為：

```
表格變數.Cell(列, 行).Range.Text = 設定值
```

例如設定第 2 列、第 3 行的內容為「姓名」：

```
table.Cell(2, 3).Range.Text = "姓名"
```

注意列、行的編號從「1」開始。

若要在表格右方新增一列，或在表格下方新增一行的語法為：

```
表格變數.Rows.Add() # 新增一列  
表格變數.Columns.Add() # 新增一行
```

有新增列、行的方法，當然也有刪除列、行的方法，語法為：

```
表格變數.Rows(n).Delete() # 刪除第 n 列  
表格變數.Columns(n).Delete() # 刪除第 n 行
```

「n」為要刪除的列、行編號，例如：

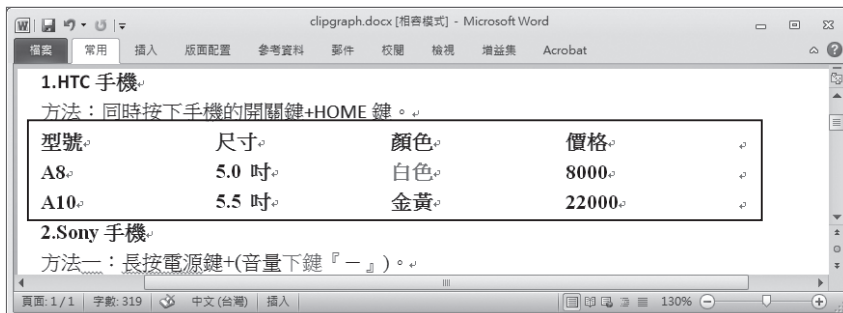
```
table.Rows(2).Delete() # 刪除第 2 列  
table.Columns(3).Delete() # 刪除第 3 行
```

Win32com 模組也提供取得表格列、行數量的方法，語法為：

```
變數名稱 = 表格變數.Rows.Count # 取得列數量  
變數名稱 = 表格變數.Columns.Count ## 取得行數量
```

**範例：Word 文件新增表格及儲存格內容**

在 Word 文件中新增一個 3 列 4 行的表格，再使用迴圈設定儲存格內容。



程式碼：chA\tabledocx1.py

```
.....略
7 doc = word.Documents.Open(cpath + "\\media\\clipgraph.docx")
8 data = [ ["型號", "尺寸", "顏色", "價格"],
           ["A8", "5.0 吋", "白色", "8000"], \
9           ["A10", "5.5 吋", "金黃", "22000"] ]
10 paragraphs = doc.Paragraphs
11 range1 = paragraphs(4).Range
12 table = doc.Tables.Add(range1, 3, 4)
13 for i in range(1,table.Rows.Count+1):
14     for j in range(1,table.Columns.Count+1):
15         table.Cell(i,j).Range.Text = data[i-1][j-1]
16 table.Cell(2,3).Range.Font.Color = 0x0000FF
17 #doc.Close()
18 #word.Quit()
```

程式說明

- 8-9 建立 3x4 的二維串列做為表格儲存格的內容。
- 10-11 因段落 4 只有一個換行符號，所以用段落 4 做為插入表格的位置。
- 12 建立 3 列 4 行的表格。
- 13-14 使用迴圈逐一加入儲存格內容：注意儲存格編號是由 1 開始，所以 range 函式範圍要由 1 到列或行數量加 1。
- 15 儲存格編號是由 1 開始，串列索引值則由 0 開始，所以串列索引要使用「data[i-1][j-1]」。
- 16 設定第 2 列第 3 行儲存格內容顏色為紅色。

A.1.5 加入圖片

在 Word 文件加入圖片的語法為：

範圍變數 `.InlineShapes.AddPicture(檔案路徑, 連結, 儲存)`

- **連結**：是布林值，`True` 表示連結到原始圖片檔，`False` 表示不連結到圖片檔。
- **儲存**：是布林值，`True` 表示將原始圖片檔存於 Word 檔中，`False` 表示 Word 檔中不儲存圖片檔。

例如在 `<clipgraph.docx>` 段落 4 插入 `<cell.jpg>` 圖片，並將圖片檔儲存於 Word 檔內：`(<imagedocx1.py>)`

```
.....略  
doc = word.Documents.Open(cpath + "\\media\\clipgraph.docx")  
paragraphs = doc.Paragraphs  
range1 = paragraphs(4).Range  
range1.InlineShapes.AddPicture(cpath + "\\media\\cell.jpg", False, True)
```

執行結果為：





A.1.6 取代文字

Win32com 模組提供自動取代文件中指定文字的功能。通常使用取代文字前會先清除搜尋文字及取代文字的格式，以免因格式影響取代效果，語法為：

```
Word 應用變數.Selection.Find.ClearFormatting()  
Word 應用變數.Selection.Find.Replacement.ClearFormatting()
```

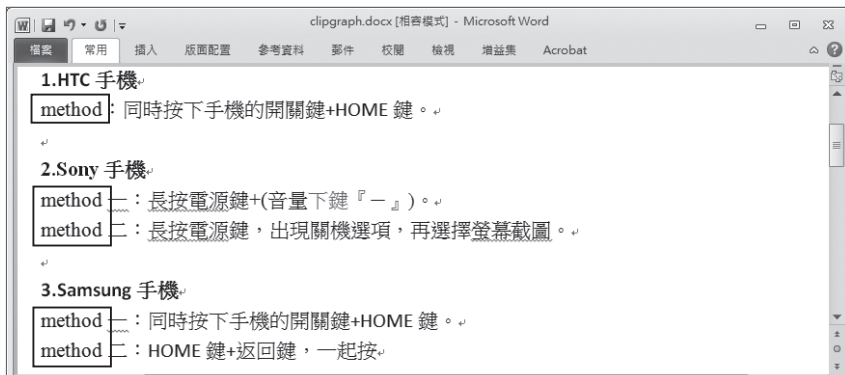
取代 Word 文件特定文字的語法為：

```
Word 應用變數.Selection.Find.Execute( 被取代文字, 比對 1, 比對 2,  
    比對 3, 比對 4, 比對 5, 方向, 動作, 格式, 取代文字, 取代次數 )
```

- **比對 1**：布林值，True 為區分大小寫字母，False 為不分大小寫字母。
- **比對 2**：布林值，True 為完整文字，False 為部分文字也取代。
- **比對 3**：布林值，True 為可使用萬用字元，False 為不可使用萬用字元。
- **比對 4**：布林值，True 為可使用 Like 比對，False 為不可使用 Like 比對。
- **比對 5**：布林值，True 為完全符合時態，False 為不分時態（例如過去式、過去分詞也可以）。
- **方向**：布林值，True 表示向前搜尋，False 表示向後搜尋。
- **動作**：搜尋到被取代文字後所執行的動作，可能的常數值有：
 - constants.wdFindAsk**：顯示對話方塊詢問使用者是否繼續搜尋。
 - constants.wdFindContinue**：取代後繼續搜尋。
 - constants.wdFindStop**：取代後停止搜尋。
- **格式**：布林值，True 表示需符合格式，False 表示不需符合格式。
- **取代次數**：設定取代的次數，可能的常數值有：
 - constants.wdReplaceNone**：不取代。
 - constants.wdReplaceOne**：只取代一次。
 - constants.wdReplaceAll**：取代全部。

範例：Word 文件取代文字

將 Word 文件中所有「方法」都取代為「method」。



程式碼：chA\replace1.py

```
1 import os
2 from win32com import client as client
3 from win32com.client import constants
4 .....略
5
6 8 doc = word.Documents.Open(cpath + "\\media\\clipgraph.docx")
7 9 word.Selection.Find.ClearFormatting()
8 10 word.Selection.Find.Replacement.ClearFormatting()
9 11 word.Selection.Find.Execute("方法", False, False, False, False, False,
10 True, constants.wdFindContinue, False, "method", constants.wdReplaceAll)
11 12 #doc.Close()
12 13 #word.Quit()
```

程式說明

- 9-10 清除搜尋及取代文字格式。
- 11 將「方法」文字全部取代為「method」。



A.2 實戰：菜單自動產生器及批次置換文字

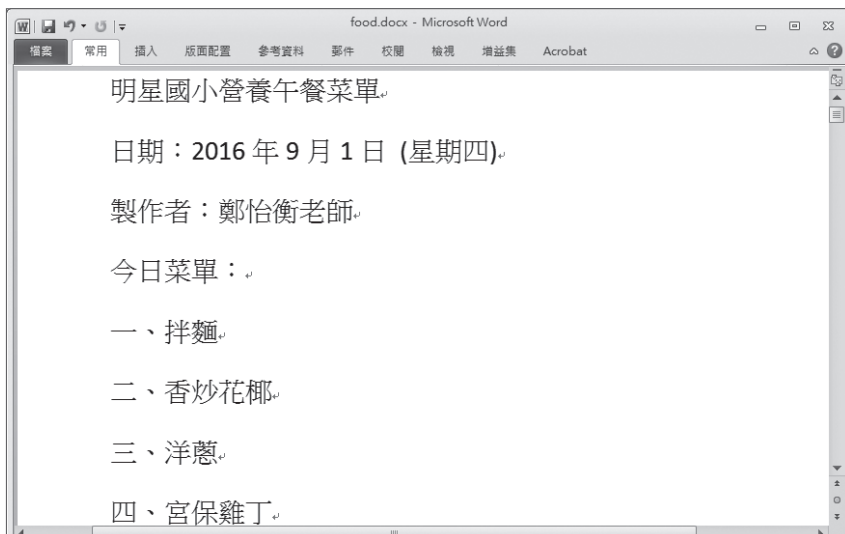
本節利用 Win32com 模組製作兩個實際應用：自動建立整個月份的營養午餐菜單 Word 文件，及自動取得指定目錄中所有 Word 文件 (包含子目錄)，並對所有 Word 檔案進行置換文字功能。

A.2.1 應用一：自動建立菜單 Word 文件

許多學校營養午餐的菜單是由教師輪流製作，讓教師在教學之外增加不少負擔，本例將自動產生教師最熟悉的 Word 菜單文件，若教師對產生的菜單不滿意，只要略做修改即可。

應用程式總覽

菜單會由 3 種主食隨機選取一種、20 種蔬菜及 20 種魚肉各隨機選取兩種、10 種湯類隨機選取一種組合成當日菜單。每天菜單自成一頁，週六及週日會自動跳過 (範例以 2016 年 9 月份菜單為例)。(<randfood.py>)



蔬菜及魚肉要由串列隨機選取兩種，且兩個元素不可重複，因此撰寫取得指定範圍中 2 個不重複亂數的函式。

應用程式內容

程式碼：chA\randfood.py

```
1 def getrandom2(n1, n2): # 取得 2 個不重複的亂數
2     while True:
3         r1 = random.randint(n1, n2)
4         r2 = random.randint(n1, n2)
5         if r1 != r2: # 如果兩數不相等就跳出迴圈
6             break
7     return r1, r2
```

程式說明

- 2-6 使用無窮迴圈取得 2 個不重複的亂數。
- 3-4 取得 2 個亂數。
- 5-6 如果 2 個亂數不相等就跳出無窮迴圈。
- 7 傳回 2 個亂數。

接著宣告各變數及串列。

程式碼：chA\randfood.py (續)

```
9 import os, random
10 from win32com import client
11 from win32com.client import constants
12 word = client.gencache.EnsureDispatch('Word.Application')
13 word.Visible = 1
14 word.DisplayAlerts = 0 # 不顯示警告
15 doc = word.Documents.Add()
16 range1 = doc.Range(0,0) # 檔案開頭
17 range1.Style.Font.Size = "16" # 字體尺寸
18 title = "明星國小營養午餐菜單"
19 year1 = "2016 年 9 月"
20 week = ["一", "二", "三", "四", "五"]
21 teacher = ["歐陽怡", "翟定國", "陳碧山", "陳麗娟", "鄭怡衡", "林鄧超",
22            "朱健政", "劉偉明", "劉維琪", "梁銀燕"]
23 rice = ["糙米飯", "白米飯", "拌麵"]
24 vegetable = ["毛豆白菜", "豆芽菜", "蛋香時瓜", "高麗菜", "佛手瓜",
25              "酸菜豆包", "冬瓜", "蘿蔔海結", "茄汁洋芋", "家常豆腐", "鮮菇花椰",
26              "豆皮三絲", "伍彩雪蓮", "干香根絲", "茄汁豆腐", "香炒花椰", "芹香粉絲",
27              "紅蘿蔔", "洋葱", "青椒"]
28 meat = ["糖醋排骨", "美味大雞腿", "椒鹽魚條", "香菇肉燥", "宮保雞丁",
29          "香滷腿排", "梅干絞肉", "香酥魚丁", "條瓜燒雞", "時瓜肉絲", "海結滷肉",
```



```

        "蔥燒雞", "柳葉魚", "咖哩絞肉", "筍香雞", "沙茶豬柳", "五香棒腿",
        "三杯雞丁", "海結豬柳", "茄汁雞丁"]
25 soup = ["蛋香木須湯", "味噌海芽湯", "綠豆湯", "榨菜肉絲湯", "薑絲海芽湯",
        "枸杞愛玉湯", "冬菜蛋花湯", "冬瓜西米露", "紫菜蛋花湯", "蛋香木須湯"]
26 date1= 1    # 開始日期為 1 日
27 weekday = 4    # 開始日期為星期四

```

程式說明

- 15 建立新的 Word 檔案。
- 16 將範圍設於檔案起始處，由檔案起始處開始加入內容。
- 17 設定字體大小為 16。
- 20 week 串列儲存星期一到星期五。
- 21 teacher 串列存 10 個老師的姓名。
- 22 rice 串列存 3 個主食。
- 23 vegetable 串列存 20 道蔬菜菜名。
- 24 meat 串列存 20 道魚肉菜名。
- 25 soup 串列存 10 道湯的名稱。
- 26 date1 為每個月 1 日開始開菜單。
- 27 weekday 存菜單是星期幾。

最後是產生整月份菜單程式碼。

程式碼：chA\randfood.py (續)

```

29 while weekday < 6 and date1 < 31:    # 週一到週五及 30 日前才製作菜單
30     range1.InsertAfter(title + "\n")
31     range1.InsertAfter("日期：" + year1 + str(date1) + " 日
        ( 星期 " + week[weekday-1] + ") \n")
32     range1.InsertAfter("製作者：" + teacher[random.randint(0,9)] + " 老師 \n")
33     range1.InsertAfter("今日菜單：\n")
34     range1.InsertAfter("一、" + rice[random.randint(0,2)] + "\n")
35     rand1, rand2 = getrandom2(0,19)    # 取得兩個亂數
36     range1.InsertAfter("二、" + vegetable[rand1] + "\n")
37     range1.InsertAfter("三、" + vegetable[rand2] + "\n")
38     rand1, rand2 = getrandom2(0,19)
39     range1.InsertAfter("四、" + meat[rand1] + "\n")
40     range1.InsertAfter("五、" + meat[rand2] + "\n")
41     range1.InsertAfter("六、" + soup[random.randint(0,9)] + "\n")
42     range1.Collapse(constants.wdCollapseEnd)    # 移到 range 尾

```



```

43     range1.InsertBreak(constants.wdSectionBreakNextPage) # 換頁
44     weekday += 1 # 星期加 1
45     date1 += 1 # 日期加 1
46     if weekday == 6: # 如果是星期六
47         weekday = 1 # 設為星期一
48         date1 += 2 # 日期加 2 (星期六及星期日)
49
50 cpath=os.path.dirname(__file__)
51 doc.SaveAs(cpath + "\\media\\food.docx") # 存為 <food.docx>
52 #doc.Close()
53 #word.Quit()

```

程式說明

- 29-48 建立 Word 檔案內容：如果是星期一到星期五且日期小於 30 日就執行 29-47 列。
- 30 加入標題。
- 31 加入日期，因為串列索引是由 0 開始，最後的星期使用「week[weekday-1]」取得中文星期文字。
- 32 由教師姓名串列隨機取得一位姓名輸出。
- 34 由主食串列隨機取得一道主食輸出。
- 35-37 先由 getrandom2(0,19) 函式取得 0 到 19 間兩個不重複亂數，再從蔬菜串列取出兩道菜名輸出。
- 38-40 與 34-46 相同，從魚肉串列取出兩道菜名輸出。
- 41 由湯名串列隨機取得一道湯輸出。
- 42 將插入點移到文件最後。
- 43 插入換列，使下一個菜單由新頁開始。
- 44-45 星期及日期都加 1。
- 46-48 如果是星期六，就將星期設為星期一，並將日期加 2 以跳過星期六及星期日。
- 50-51 將菜單存於 <media> 資料夾的 <food.docx> 檔案。

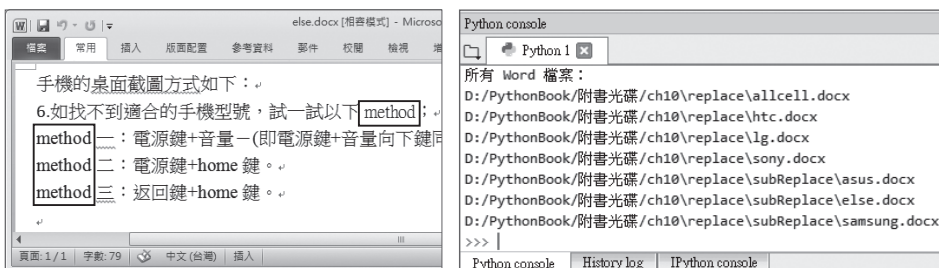


A.2.2 應用二：批次置換 Word 檔案的文字

製作 Word 文件時，常有許多文件需做相同文字置換，若是一個一個檔案操作，會浪費大量時間。本應用會找出指定目錄中所有 Word 檔案（包含子目錄），並對每一個檔案進行文字置換。

應用程式總覽

程式執行後會將 <replace> 目錄（包含子目錄）所有 Word 文件中的「方法」都置換為「method」。下左圖為 <replace\subReplace\else.docx> 檔置換後的結果，下右圖是在命令視窗顯示所有 Word 檔案。（<replaceall.py>）



應用程式內容

程式碼：chA\replaceall.py

```
1 import os
2 from win32com import client
3 from win32com.client import constants
4 word = client.gencache.EnsureDispatch('Word.Application')
5 word.Visible = 0
6 word.DisplayAlerts = 0
7 runpath = os.path.dirname(__file__) + "\\replace" #處理<replace>資料夾
8 tree = os.walk(runpath) #取得目錄樹
9 print("所有 Word 檔案:")
10 for dirname, subdir, files in tree:
11     allfiles = []
12     for file in files: #取得所有.docx .doc檔，存入allfiles串列中
13         ext = file.split(".")[-1] #取得附加檔名
14         if ext=="docx" or ext=="doc": #取得所有.docx .doc檔
15             allfiles.append(dirname + '\\'+ file) #加入allfiles串列
16
```

```
17     if len(allfiles) > 0: # 如果有符合條件的檔案
18         for dfile in allfiles:
19             print(dfile)
20             doc = word.Documents.Open(dfile) # 開啟檔案
21             word.Selection.Find.ClearFormatting()
22             word.Selection.Find.Replacement.ClearFormatting()
23             word.Selection.Find.Execute("方法", False, False,
                False, False, False, True, constants.wdFindContinue,
                False, "method", constants.wdReplaceAll)
24             doc.Close()
25 word.Quit()
```

程式說明

- 5 及 25 本應用會處理多個 Word 檔，因此不顯示 Word 軟體。
- 7 處理的資料夾為 <replace>。
- 8 使用「os.walk」取得包括子目錄的檔案結構。
- 10 依次處理各資料夾中的檔案。
- 11 allfiles 串列儲存資料中所有檔案。
- 12 「os.walk」取得的檔案傳回值存於 files 變數中。
- 13 取得附加檔名。
- 14-15 如果是 Word 檔案就將檔案路徑加入 allfiles 串列。
- 17-24 如果有 Word 檔案才處理。
- 18 逐一處理 Word 檔案。
- 20 開啟檔案。
- 21-22 清除搜尋及取代文字格式。
- 23 將檔案中「方法」文字都置換為「method」。
- 24 關閉檔案。

[illegible]