使用 Python 来控制 MS Word

Description

Python 可以透过 win32com 这个 module 来控制 Office 系列的软件，本文将会针对如何运用 Python 来控制 MS Word 做一个简单的介绍。

Text

要使用Python控制MS Word，您需要先安装win32com套件，这个套件可以到 http://sourceforge.net/projects/pywin32/ 找到。本文假设您已经正确安装win32com及相关套件，所以将不再针对此部分多做说明。

毫无疑问的，您需要先 import win32com 模块才能进行 Word 的控制。

```
import win32com
from win32com.client import Dispatch, constants
```

接着，要让我们的 Python 程序和 MS Word 建立起连接。

```
msword = Dispatch('Word.Application')
```

用 Dispatch()的方式将会启动 MS Word。不过，如果您已经有执行 MS Word，则此方式只会与现存的 MS Word 建立连接。如果您强烈希望能够有一个新的 MS Word 程序出现，可用下面的方式：

```
msword = DispatchEx('Word.Application')
```

此时您会发现画面上没有任何 MS Word 出现，其实他已经在背后执行了。您可以透过工作管理员来查看是否有一个名为"WINWORD.EXE"的 Process。不产生画面的好处是您可以在背景来处理要进行的工作。如果您想要看看到底是不是真的成功的启动 MS Word，请设定 Visible 属性。

```
msword.Visible = 1  # 1 表示要显示画面，若为 0 则不显示画面。您可以随时更改此属性。
```

除了不显示画面外，您也许还会希望不要显示一些警告信息。此时请设定 DisplayAlerts 属性：

```
msword.DisplayAlerts = 0    # 1 表示要显示讯息，0 则会隐藏讯息。
```

若您真的有把画面叫出来，您或许会发现此时的画面上是没有任何文件开启的。没错！前面的动作只是帮助我们启动 Word 并且建立连接，接着我们就要来开启文件了。我们可以开启已经存在的文件，或者是新增一个空白文件。

```
doc = msword.Documents.Open(FileName="xxx.doc") # 开启一个名为 xxx.doc 的文件。
newDoc = msword.Documents.Add()     # 开启一个新的文件。
msword.Quit()                       # 关闭 MS Word。
```

当然，除了开启档案或新建文件，您也可以存盘或者控制这些文件。

```
docCnt = msword.Documents.Count     # 取得目前开启檔的数量。
doc = msword.Documents[n]           # 取得第 n 个文件的对象，以便后面的控制。
doc.Activate()                      # 将檔设定为主要工作檔。
doc.PrintOut()                      # 打印文件
doc.Save()                          # 存档
doc.SaveAs('xxx.doc')               # 另存新檔
doc.Undo(n)                         # 回复前 n 次的动作
```

取得与文件的联系，接着我们可以对它进行编辑。不过，我们要能够先取得编辑的控制权。透过 Document 的 Range()函式，我们可以取得 MS Word 的 Range 对象。

```
    range = doc.Range(0, 0) # 取得 Range 对象，范围为文件的最开头。
    range = doc.Range()     # 取得 Range 对象，范围为文件的最尾端。
    range = doc.Range(doc.Content.Start, doc.Content.End)  # 取得 Range 对象，范围整份文件。
```

有了 Range 对象，我们就可以开始进行编辑了。

```
    range.InsertBefore('在 range 前面插入的文字')
    range.InsertAfter('在 range 后面插入的文字')
    select = range.Select() # 将 range 的范围全部选取。并且取得 Selection 对象。
```

如果要设定 Style，可以透过 range 对象的 Style 属性来设定。

```
    range.Style = constants.wdStyleHeading1     # 设定 style 为 Heading 1
    range.Style = constants.wdStyleHeading2     # 设定 style 为 Heading 2
    range.Style = constants.wdStyleHeading3     # 设定 style 为 Heading 3
    range.Style = constants.wdStyleHeading4     # 设定 style 为 Heading 4
    range.Style = constants.wdStyleHeading5     # 设定 style 为 Heading 5
    range.Style = constants.wdStyleHeading6     # 设定 style 为 Heading 6
    range.Style = constants.wdStyleHeading7     # 设定 style 为 Heading 7
    range.Style = constants.wdStyleHeading8     # 设定 style 为 Heading 8
    range.Style = constants.wdStyleHeading9     # 设定 style 为 Heading 9
    range.ParagraphFormat.Alignment = constants.wdAlignParagraphLeft   # 设定段落为靠左
    range.ParagraphFormat.Alignment = constants.wdAlignParagraphRight  # 设定段落为靠右
    range.ParagraphFormat.Alignment = constants.wdAlignParagraphCenter # 设定段落为置中
    range.ParagraphFormat.Alignment = constants.wdAlignParagraphJustify # 设定段落为左右对齐
    range.Style.Font.Name = "Arial"             # 设定字型为 Arial
    range.Style.Font.Name = "Time New Roman"    # 设定字型为 Time New Roman
    range.Style.Font.Name = "标楷体"             # 设定字型为标楷体
    range.Style.Font.Color = 0xFF0000           # 设定字型的颜色为 Blue
    range.Style.Font.Color = 0x00FF00           # 设定字型的颜色为 Green
    range.Style.Font.Color = 0x0000FF           # 设定字型的颜色为 Red
    range.Style.Font.Bold = 1                   # 设定字型为粗体字
    range.Style.Font.Italic = 1                 # 设定字型为斜体字
    range.Style.Font.Underline = 1              # 为字型加底线
    range.Style.Font.Shadow = 1                 # 为字型加阴影
    range.Style.Font.Outline = 1                # 为字型加外框
```

如果要插入一个表格，可以用下面的方式来做。

```
    table = doc.Tables.Add(range, 3, 4)             # 新增一个 3x4 表格
    table.Cell(1,1).Range.InsertAfter('Some text') # 新增文字到 cell(1,1)
    table.Cell(1,1).Range.Font.Name = "Arial"       # 设定字型为 Arial
    table.Cell(1,1).Range.Font.Color = 0xFF0000     # 设定字型为 blue
    table.Rows.Add()                                # 新增一个 Row
    table.Columns.Add()                             # 新增一个 Column
```

通过Python可以很方便的操作Word

http://czug.org/blog/panjy/python-word

```
    import win32com
    from win32com.client import Dispatch, constants

    w = win32com.client.Dispatch('Word.Application')
    # 或者使用下面的方法，使用启动独立的进程
    # w = win32com.client.DispatchEx('Word.Application')

    # 后台运行，不显示，不警告
    w.Visible = 0
    w.DisplayAlerts = 0

    # 打开新的文件
    doc = w.Documents.Open( FileName = filenamein )
    # worddoc = w.Documents.Add() # 创建新的文档

    # 插入文字
    myRange = doc.Range(0,0)
    myRange.InsertBefore('Hello from Python!')

    # 使用样式
    wordSel = myRange.Select()
    wordSel.Style = constants.wdStyleHeading1

    # 正文文字替换
    w.Selection.Find.ClearFormatting()
    w.Selection.Find.Replacement.ClearFormatting()
    w.Selection.Find.Execute(OldStr, False, False, False, False, False, True, 1,
True, NewStr, 2)

    # 页眉文字替换
    w.ActiveDocument.Sections[0].Headers[0].Range.Find.ClearFormatting()
    w.ActiveDocument.Sections[0].Headers[0].Range.Find.Replacement.ClearForma
tting()
    w.ActiveDocument.Sections[0].Headers[0].Range.Find.Execute(OldStr,  False,
False, False, False, False, True, 1, False, NewStr, 2)

    # 表格操作
    doc.Tables[0].Rows[0].Cells[0].Range.Text ='123123'
    worddoc.Tables[0].Rows.Add() # 增加一行

    # 转换为 html
    wc = win32com.client.constants
    w.ActiveDocument.WebOptions.RelyOnCSS = 1
    w.ActiveDocument.WebOptions.OptimizeForBrowser = 1
    w.ActiveDocument.WebOptions.BrowserLevel = 0 # constants.wdBrowserLevelV4
    w.ActiveDocument.WebOptions.OrganizeInFolder = 0
    w.ActiveDocument.WebOptions.UseLongFileNames = 1
    w.ActiveDocument.WebOptions.RelyOnVML = 0
    w.ActiveDocument.WebOptions.AllowPNG = 1
    w.ActiveDocument.SaveAs(FileName    =    filenameout,    FileFormat    =
wc.wdFormatHTML)

    # 打印
    doc.PrintOut()

    # 关闭
    # doc.Close()
    w.Documents.Close(wc.wdDoNotSaveChanges)
    w.Quit()
```