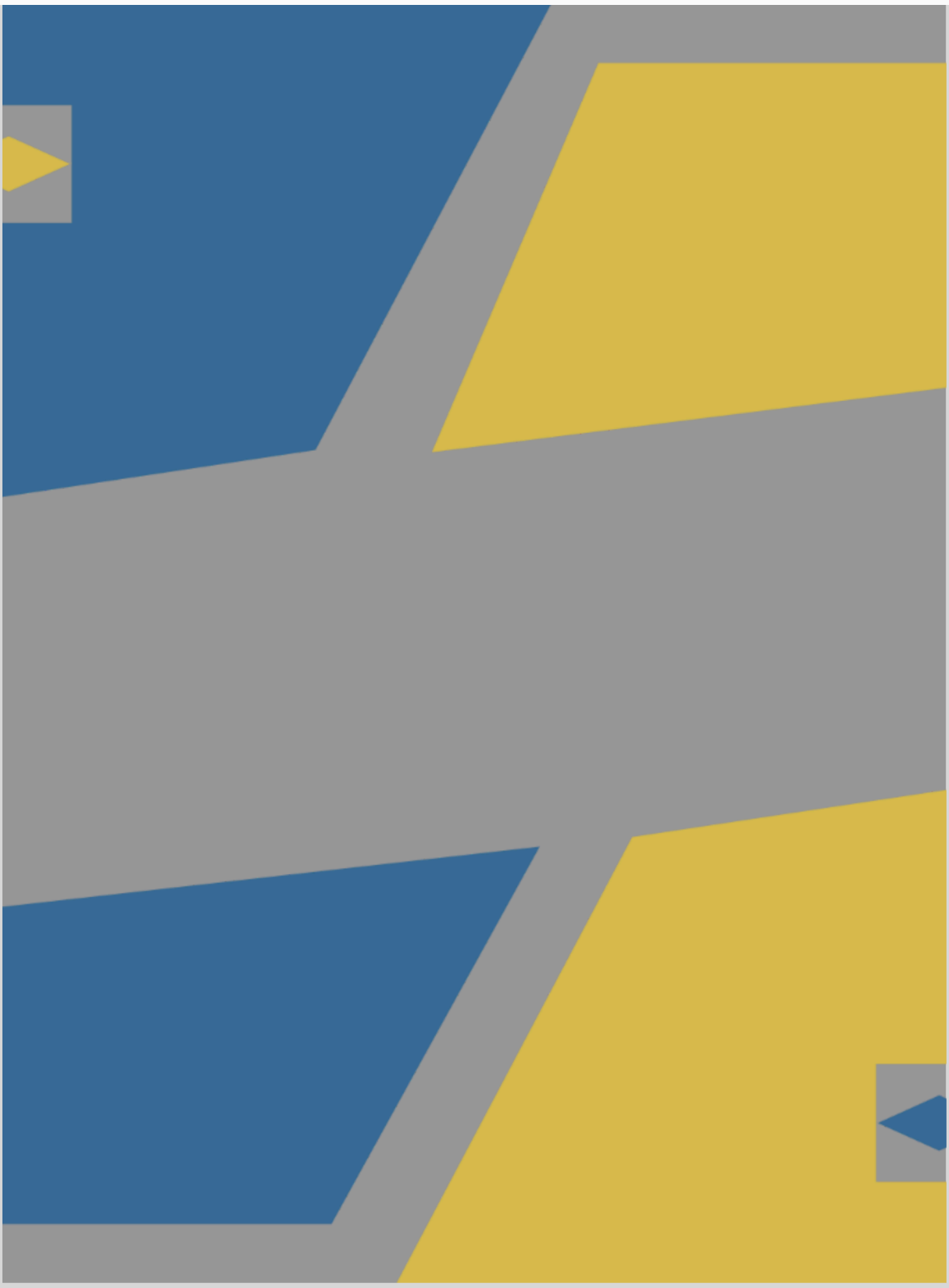


本篇文章讨论如何设置样式，以及如何利用Word的样式管理器。

使用样式——使用Python读写Office文档之四

Jerry Jho · 6 个月前

通常，在排版管理中，与字符内容无关的所有可视要素都可称之为『样式』，具体说来，包括字体、颜色、对齐、行间距等等。

在Word（以及带有文本编辑功能的Office软件）中，从样式应用的对象分，常用的样式类型有字符样式、段落样式、表格样式、枚举样式等。python-docx目前主要支持前三类，已经基本满足日常使用。

每种样式包含的内容主要有字体（包括字体名称、颜色、倾斜、加粗）和段落格式（缩进、分页模式）等。根据样式应用对象的不同，每类样式不一定包含所有的内容。例如，字符样式就没有段落格式这个内容。

Word有样式管理器，负责管理内建样式和用户自定义样式。python-docx可以操作这个管理器，把自定义的样式添加到管理器里，或者从管理器选取一个样式加以应用。

由于python-docx的样式设置并不能包含Word的所有样式设置。因此个人的建议是：

1. 当你的操作只是涉及Word本身（例如批量整理文件格式），可以利用Word的图形界面设置好自定义样式，然后利用python-docx操作样式管理器，给特定的对象特定的样式。
2. 当你的操作涉及Word之外的东西，而且对样式设置的要求不高（例如，HTML转Word，或者格式比较简单的文档，例如公文），可以使用python-docx直接设置样式。
3. 如果你既不能用Word设置自定义样式（例如需要几百种样式类型），又对样式类型的设置要求很高，则不要用python-docx，考虑VBScript/VBA，或者win32com等功能完备的工具。

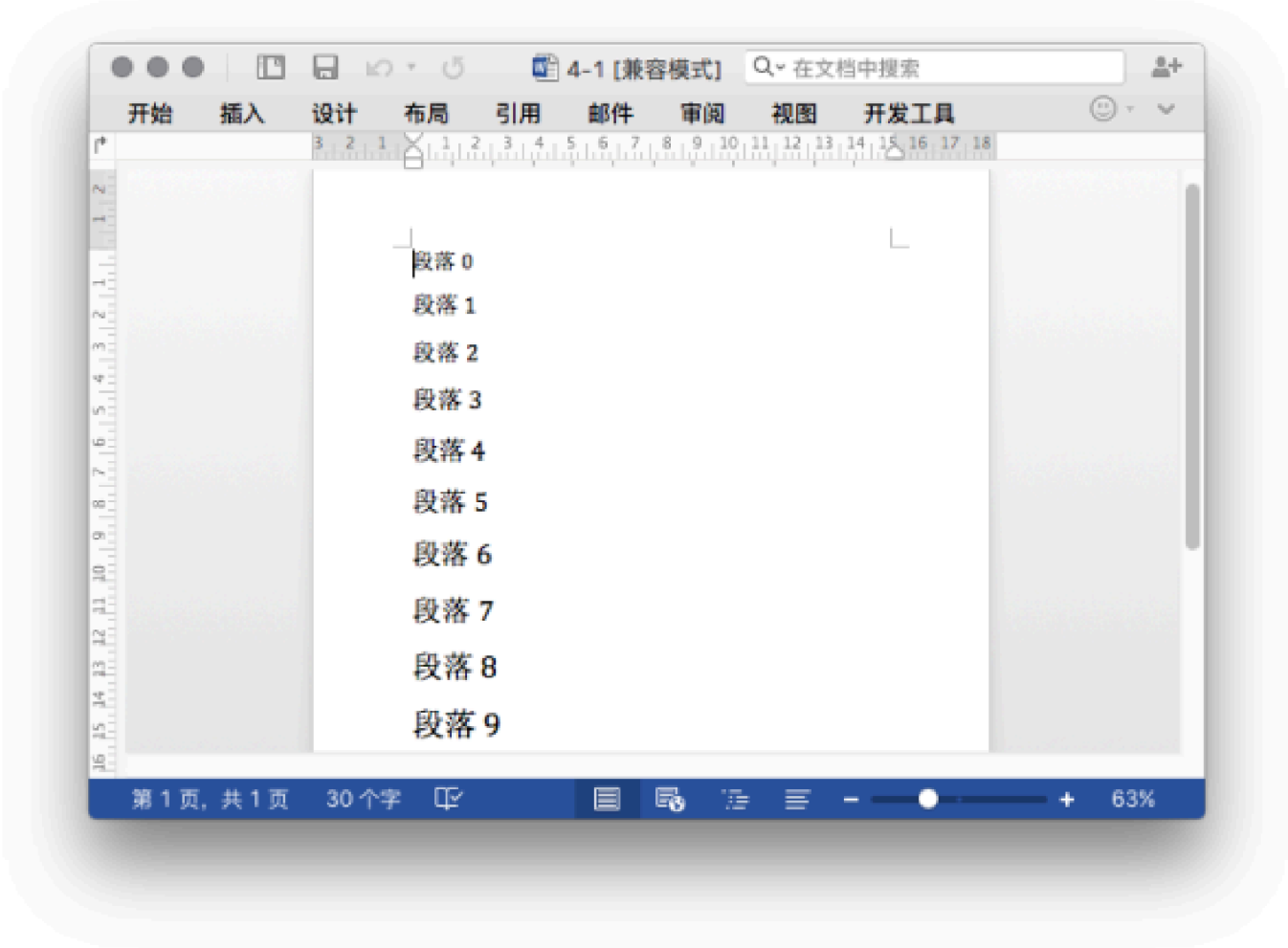
1. 创建样式和设置字体

游程对象、段落对象和表格对象都有一个成员Style，而这个Style有一个成员Font，包含了字体的所有设置。

如果凭空的创建一个Style，要使用Document对象的Styles集合的add_style方法。下面的例子里添加了10个段落，为每个段落创建一个新的Style。这些Style的区别是字号依次增加。

```
# -*- coding: utf-8 -*-
from docx import Document
from docx.shared import Pt
from docx.enum.style import WD_STYLE_TYPE

doc = Document()
for i in range(10):
    p = doc.add_paragraph(u'段落 %d' % i)
    style = doc.styles.add_style('UserStyle%d' %i,WD_STYLE_TYPE.PARAGRAPH)
    style.font.size = Pt(i+20)
```

```
    p.style = style

doc.save('4-1.docx')
```

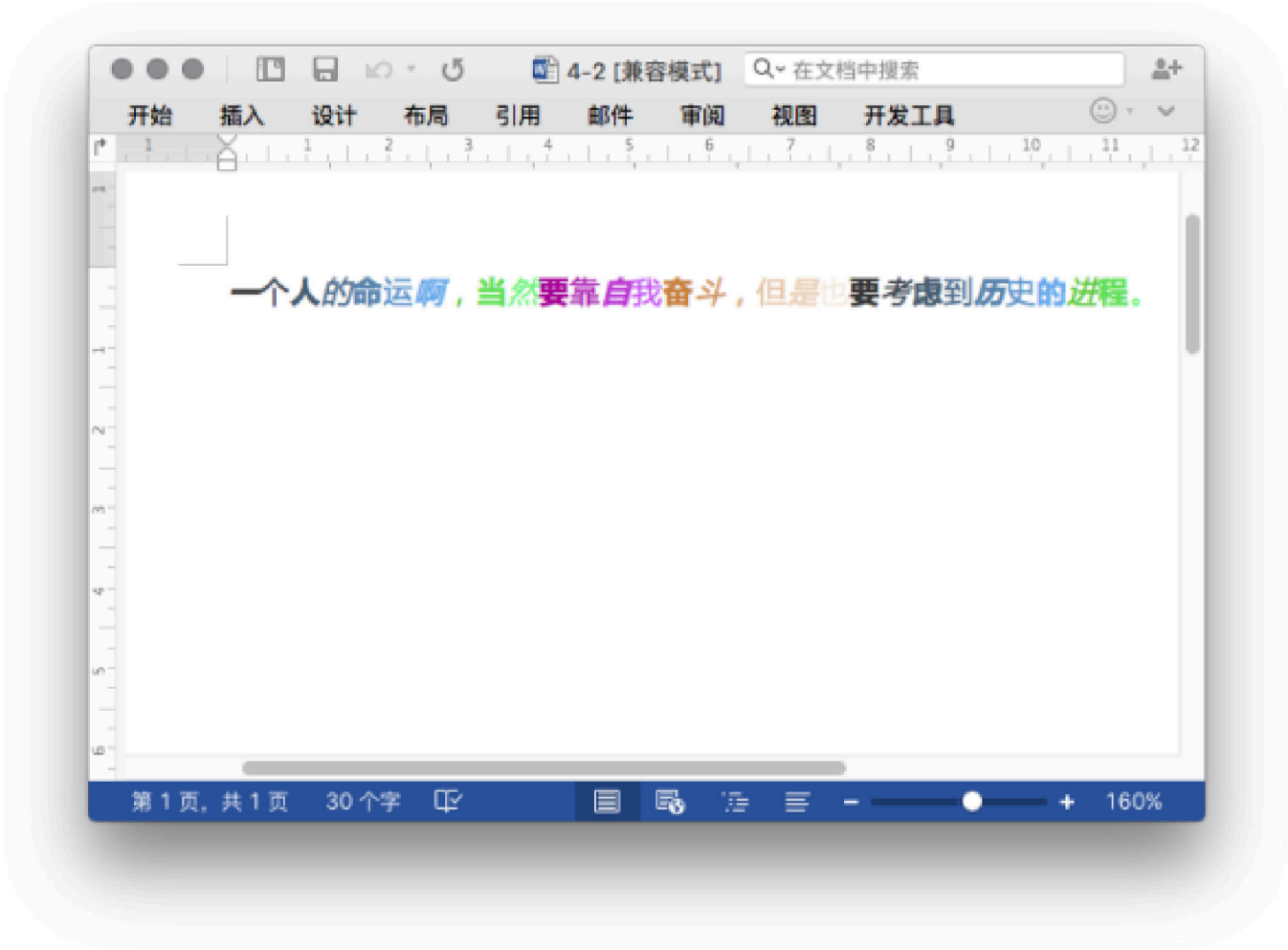


注意到python-docx提供了docx.shared模块，供以Pt（磅）为单位设置字体。类似的单位有Mm、Cm、Inches、Emu（物理尺寸）以及Twips（缇，微软喜好的一种蜜汁尺寸）

对于游程对象，其字体操作可以直接使用Run.font成员。下面给一个字体变化的一句话：

```
# -*- coding: utf-8 -*-
from docx import Document
from docx.shared import RGBColor
from docx.oxml.ns import qn

doc = Document()
p = doc.add_paragraph()
text_str = u'一个人的命运啊，当然要靠自我奋斗，但是也要考虑到历史的进程。'
for i,ch in enumerate(text_str):
    run = p.add_run(ch)
    font = run.font
    font.name = u'微软雅黑'
    # bug of python-docx
```

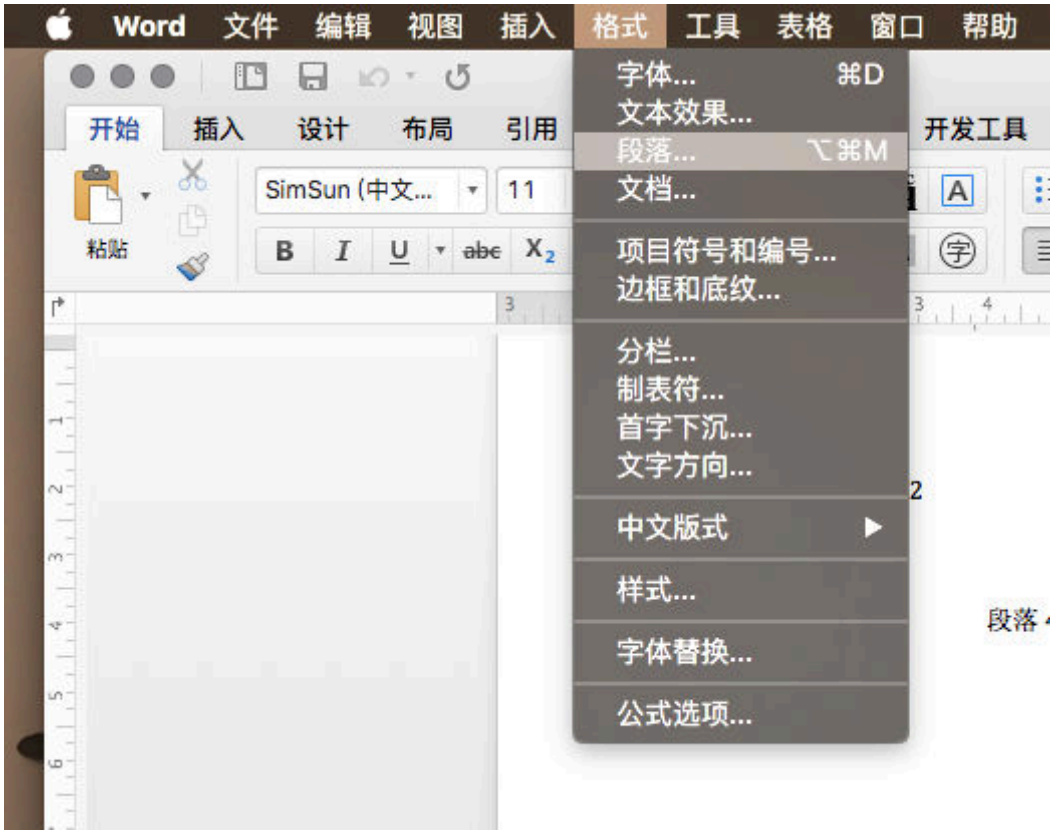
```
        run._element.rPr.rFonts.set(qn('w:eastAsia'), u'微软雅黑')
        font.bold = (i % 2 == 0)
        font.italic = (i % 3 == 0)
        color = font.color
        color.rgb = RGBColor(i*10 % 200 + 55,i*20 % 200 + 55,i*30 % 200 + 55)

doc.save('4-2.docx')
```

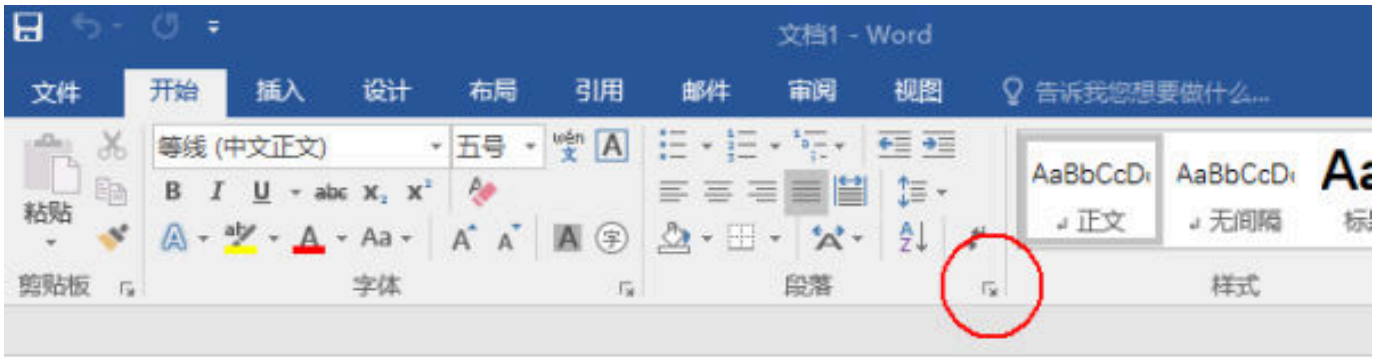


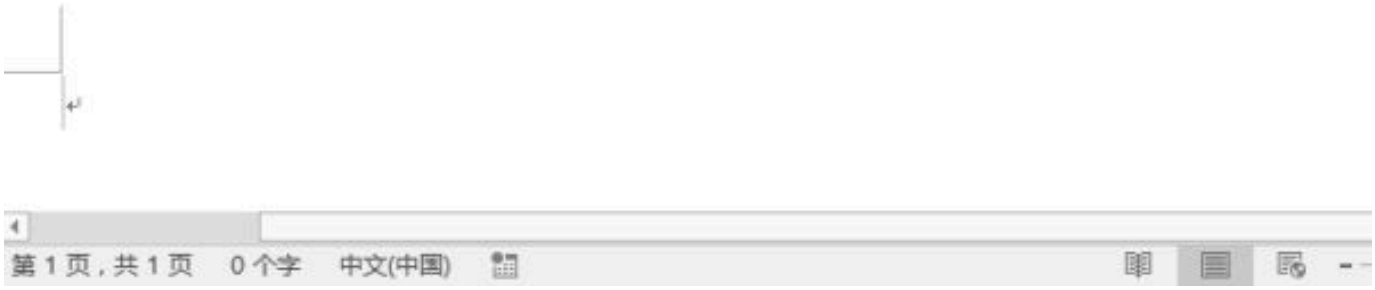
这里使用了docx.shared模块提供的RGBColor类。

2. 段落格式

对于段落样式和表格样式，有一个段落格式的成员paragraph_format。它相当于Word的这个选项：

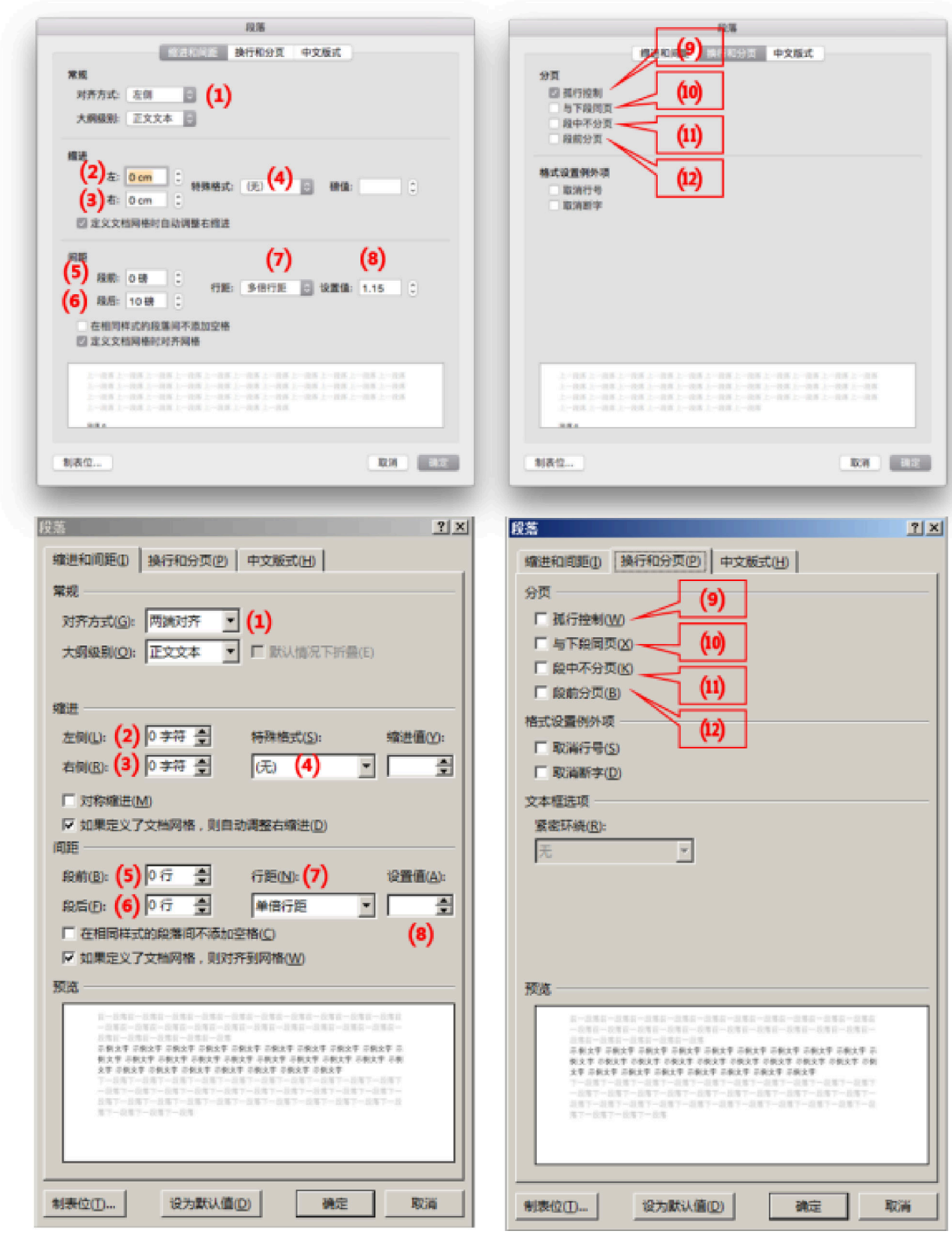






相应的，段落格式对象的各种成员，相当于对话框里的这些设置要素：

1.



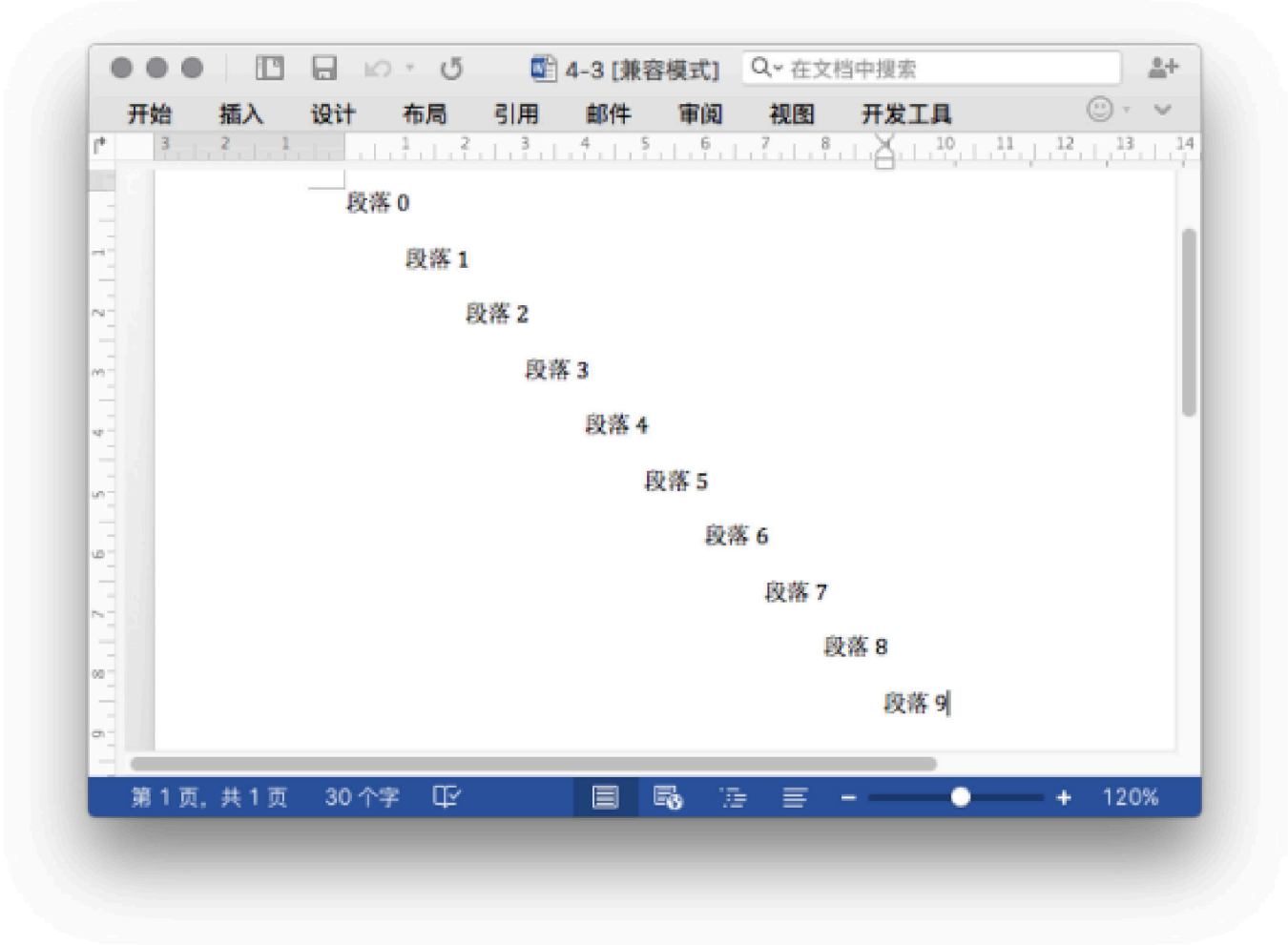
ParagraphFormat.alignment（选择一个WD_PARAGRAPH_ALIGNMENT）

2. ParagraphFormat.left_indent（长度）
3. ParagraphFormat.right_indent（长度）
4. ParagraphFormat.first_line_indent（长度）
5. ParagraphFormat.space_before（长度）

6. ParagraphFormat.space_after（长度）
7. ParagraphFormat.line_spacing_rule（选择一个WD_LINE_SPACING）
8. ParagraphFormat.line_spacing（长度）
9. ParagraphFormat.widow_control（True：设置，None：继承Style设置）
10. ParagraphFormat.keep_with_next（True：设置，None：继承Style设置）
11. ParagraphFormat.keep_together（True：设置，None：继承Style设置）
12. ParagraphFormat.page_break_before（True：设置，None：继承Style设置）

比如可以这样设置递进的左对齐：

```
 # -*- coding: utf-8 -*-
from docx import Document
from docx.shared import Cm
from docx.enum.style import WD_STYLE_TYPE

doc = Document()
for i in range(10):
    p = doc.add_paragraph(u'段落 %d' % i)
    style = doc.styles.add_style('UserStyle%d' %i,WD_STYLE_TYPE.PARAGRAPH)
    style.paragraph_format.left_indent = Cm(i)
    p.style = style

doc.save('4-3.docx')
```



3. 样式管理器

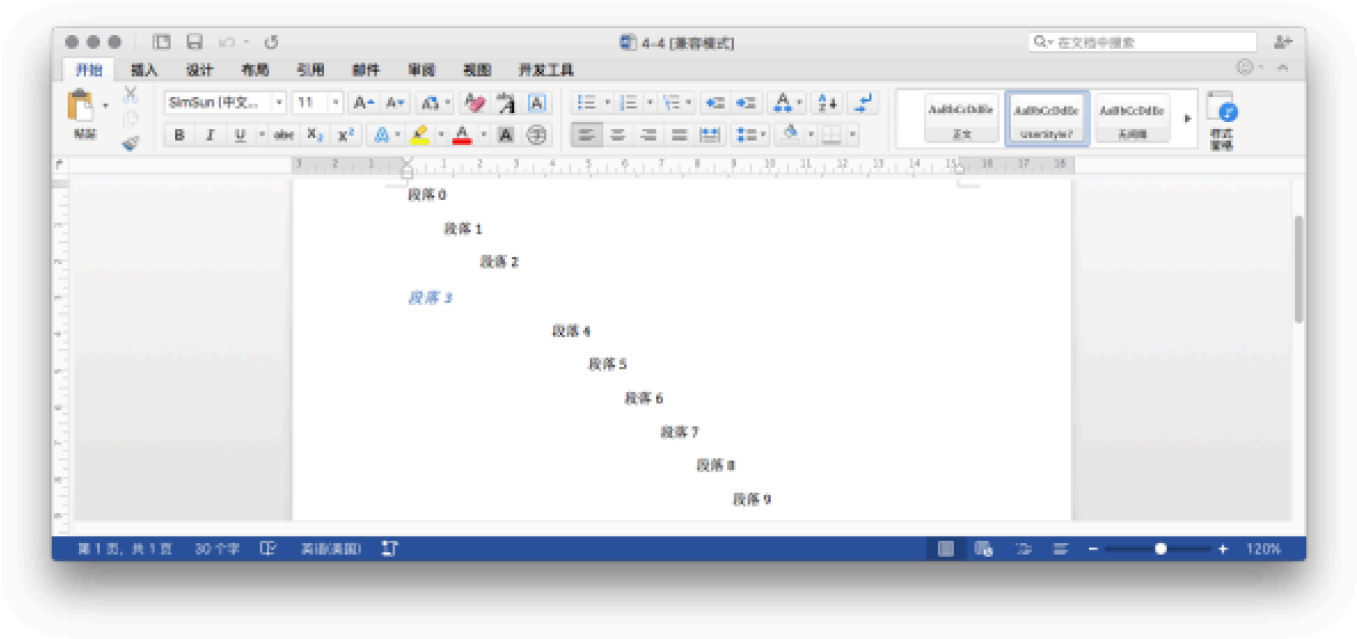
Word自带很多样式。可以利用Document.styles集合来进行访问builtin属性为True的自带样式。当然，你自己通过add_style增加的样式，也会放在styles集合里。

对于你自己创建的样式，若将其hidden和quick_style属性分别设置为False和True，则可将这个自创建样式添加到Word的快速样式管理器里。

下面举一个综合的例子。

```
# -*- coding: utf-8 -*-
from docx import Document
from docx.shared import Cm
from docx.enum.style import WD_STYLE_TYPE

doc = Document()
for i in range(10):
    p = doc.add_paragraph(u'段落 %d' % i)
    style = doc.styles.add_style('UserStyle%d' %i,WD_STYLE_TYPE.PARAGRAPH)
    style.paragraph_format.left_indent = Cm(i)
    p.style = style
```

```
    if i == 7:
        style.hidden = False
        style.quick_style = True

for style in doc.styles:
    print style.name,style.builtin

doc.paragraphs[3].style = doc.styles['Subtitle']

doc.save('4-4.docx')
```



注意样式面板的第二个快速样式就是我们刚刚添加的『UserStyle7』

==========

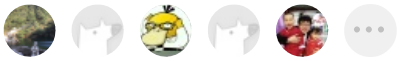
到此，这一系列的Word部分就告一段落了。接下来，本系列文章将聊一聊用python操作PowerPoint。

参考：

python-docx.readthedocs.io...

Python　Microsoft Office

58



6 条评论

写下你的评论

Shawn Zhang

求怎样编辑numbering style！
拜谢

5 个月前

伊斯特伍德

run里设置中文字体的方式已经给出，不知道table中如何设置中文字体呢？

5 个月前

kaiiak

最近在用golang写，头都大了

5 个月前

Jerry Jho（作者） 回复 **伊斯特伍德**

查看对话

道理是相同的

5 个月前

Jerry Jho（作者） 回复 **Shawn Zhang**

查看对话

python-docx暂不直接支持。

5 个月前

伊斯特伍德 回复 **Jerry Jho**（作者）

查看对话

table._element.rPr.rFonts.set(qn('w:eastAsia'), u'微软雅黑')
试了一下，系统报错了呢

5 个月前

5 个月前

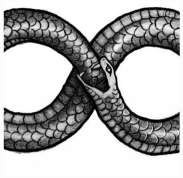
文章被以下专栏收录

蛇之魅惑

有趣的Python小知识。

进入专栏

推荐阅读



Python 的迭代器和生成器

Jerry Jho · 4 个月前　　发表于 蛇之魅惑

Python与Tcl混合编程 - 在Python中调用Tcl

Jerry Jho · 6 个月前　　发表于 蛇之魅惑

加拿大威士忌「比你想象的好」

dizzarz · 1 个月前 · 编辑精选　　发表于 猎酒党

江南春天的野菜，比肉还好吃！

丁小穗 · 1 个月前 · 编辑精选　　发表于 企鹅吃喝指南