

简介

python-pptx是一个用于创建、读取和更新PowerPoint (.pptx) 文件的 python库 。

典型的用途是根据动态内容（如数据库 查询、分析数据等），将这些内容自动化生成PowerPoint演示文稿，将数据可视化 ，方便查看
我们也可以用它做办公自动化，定义一套模板，然后根据给出的内容批量生成PPT文件，大大提高我们的办公效率

Python-pptx的Github 地址

```
https://github.com/scanny/python-pptx
```

开发文档

```
https://python-pptx.readthedocs.io/en/latest/
```

安装python-pptx

```
pip install python-pptx
```

本文使用的版本为0.6.21

使用python-pptx创建新的PPT

生成一个全新的PPT文件，这种方式适用于所有样式都是由代码来控制来控制的场景
幻灯片效果

Hello, World!

python-pptx create it

CSDN @ZHU_文涛

实现以上效果的代码

python

```
1  from pptx import Presentation
2
3  # 创建一个新的 Presentation 对象
4  prs = Presentation()
5  # 获取一个包含主标题和副标题的幻灯片版式
6  title_slide_layout = prs.slide_layouts[0]
7  # 将幻灯片加入到PPT中
8  slide = prs.slides.add_slide(title_slide_layout)
9  # 获取主标题
10 title = slide.placeholders[0]
```

```

11 | # 获取副标题
12 | subtitle = slide.placeholders[1]
13 |
14 | title.text = "Hello, World!"
15 | subtitle.text = "python-pptx create it"
16 | # 保存创建的PPT文件
17 | prs.save('G:/simple_ppt/test/test1.pptx')

```

上例中的`prs.slide_layouts[0]` 获取幻灯片的版式，幻灯片的版式共有11个，如下所示



CSDN @ZHU_文涛

从左到右依次是`slide_layouts[0]`、`slide_layouts[1]`一直到 `slide_layouts[10]`，通过对应的下标即可获取对应的幻灯片版式

Tips:

上面代码中的slide = prs.slides.add_slide(title_slide_layout)

即prs.slides 代表的是当前PPT中所有幻灯片的集合，通过add_slide 添加一张幻灯片后拿到的slide，后续针对这个slide 的各种操作也就是单张幻灯片的操作

修改幻灯片大小

1、直接通过slide_width和slide_height指定

python

```
1 | from pptx.util import Cm
2 |
3 | prs = Presentation()
4 | prs.slide_width = Cm(33.85)
5 | prs.slide_height = Cm(19.02)
```

2、通过模板指定

可以先自定义一个指定了宽高的空白页PPT模板，创建Presentation对象时引用它，后续创建的幻灯片就能继承到对应的宽高大小

python

```
1 | prs = Presentation("G:/simple_ppt/test/template.pptx")
```

创建文本

段落创建

想要在幻灯片中添加文本，先要通过add_textbox创建一个文本框，然后取得text_frame来进行操作

python

```
1  from pptx import Presentation
2  from pptx.util import Cm
3
4  def test_blog_text_add():
5      prs = Presentation()
6      prs.slide_width = Cm(33.85)
7      prs.slide_height = Cm(19.02)
8
9      bullet_slide_layout = prs.slide_layouts[6]    # 空白版式
10     slide = prs.slides.add_slide(bullet_slide_layout)
11
12     # 添加文本框
13     tx_box = slide.shapes.add_textbox(left=Cm(2.58), top=Cm(1.16), width=Cm(28), height=Cm(2.36))
14     tf = tx_box.text_frame
15
16     p0 = tf.paragraphs[0]
17     p0.text = '这是第一行段落'
18
19     p1 = tf.add_paragraph()
20     p1.text = '这是新增的第二行段落'
21
22     run = p1.add_run()
23     run.text = '。第二行结尾直接添加文字'
24
25     prs.save("G:/simple_ppt/test/blog_test.pptx")
```

生成效果如下



Tips:

上面代码中的: `tx_box = slide.shapes.add_textbox(...`

`slide.shapes`代表的是当前幻灯片中所有元素的集合, 如文本框、图片、图标、视频等等可框选的东西, 都是`shapes`, 所以若要添加什么东西, 也是通过`shapes.add_xxx`的方式来实现

文本样式添加

一、自动换行

如果我们输入的文本大于文本框的长度时，默认是不会换行的，可以使用`tf.word_wrap`来指定自动换行

python

```
1 | tf = tx_box.text_frame
2 | tf.word_wrap = True
```

二、文本布局样式

文本框中的文本默认是上方对齐，可以使用`tf.vertical_anchor` 来指定文本的布局方式

python

```
1 | from pptx.enum.text import MSO_ANCHOR
2 |
3 | tf = tx_box.text_frame
4 | tf.vertical_anchor = MSO_ANCHOR.MIDDLE
```

- TOP: 将文本与文本框顶部对齐
- MIDDLE: 垂直居中文本
- BOTTOM: 将文本与文本框底部对齐

参考: <https://python-pptx.readthedocs.io/en/latest/api/enum/MsoVerticalAnchor.html>

注意，这个只是指定了文本垂直方向上的移动，如想文本基于整个文本框居中需要指定段落的布局方式
设置文本段落布局可以通过设置`p.alignment`的方式

python

```
1 from pptx.enum.text import MSO_ANCHOR, PP_ALIGN
2
3 tf = tx_box.text_frame
4 tf.vertical_anchor = MSO_ANCHOR.MIDDLE
5 p0 = tf.paragraphs[0]
6 p0.text = '这是第一行段落'
7 p0.alignment = PP_ALIGN.CENTER
```


效果如图



PP_ALIGN的参数有以下几个

- CENTER: 居中对齐
- DISTRIBUTE: 在一行中从左到右均匀分布
- JUSTIFY: 每行都在页边空白处开始和结束，并调整单词之间的间距，使该行正好填满段落的宽度
- JUSTIFY_LOW: 在单词之间使用少量空格进行对齐
- LEFT: 默认的，左对齐

- RIGHT: 右对齐
- THAI_DISTRIBUTE: 泰语分散对齐, 输入泰语时候指定

以上效果就不一一演示了, 自己尝试下选择合适的就行

参考:

<https://python-pptx.readthedocs.io/en/latest/api/enum/PpParagraphAlignment.html>

三、文字样式修改

文字的字体、字号、加粗、斜体、下划线、颜色、超链接 等, 这些样式通过font 来设置
参数较多, 直接上代码

python

```
1  from pptx import Presentation
2  from pptx.util import Cm, Pt
3  from pptx.dml.color import RGBColor
4  from pptx.enum.text import MSO_ANCHOR, PP_ALIGN
5
6  def test_blog_text_add():
7      prs = Presentation()
8      prs.slide_width = Cm(33.85)
9      prs.slide_height = Cm(19.02)
10
11     bullet_slide_layout = prs.slide_layouts[6]
12     slide = prs.slides.add_slide(bullet_slide_layout)
13
14     # 添加文本框
15
```

```
16 tx_box = slide.shapes.add_textbox(left=Cm(2.58), top=Cm(1.16), width=Cm(28.47), height=Cm(5))
17 tf = tx_box.text_frame
18 tf.word_wrap = True      # 自动换行
19 tf.vertical_anchor = MSO_ANCHOR.MIDDLE # 垂直居中
20
21 p0 = tf.paragraphs[0]    # 第一行段落
22 p0.alignment = PP_ALIGN.CENTER # 设置段落文字居中
23 p0.line_spacing = 1.3    # 间距
24 p0.font.name = 'Arial Black' # 字体
25 p0.font.size = Pt(40)    # 字号
26 p0.font.italic = True    # 斜体
27 p0.font.bold = True      # 粗体
28 p0.font.underline = True # 显示下划线
29 p0.font.color.rgb = RGBColor(255, 0, 0) # 设置红色
30 p0.text = 'Hello World!'
31
32 p1 = tf.add_paragraph() # 添加新段落
33 p1.text = '这是第二行段落'
34
35 run = p1.add_run()
36 run.text = "。第二行结尾直接添加文字"
37 run.hyperlink.address = 'https://www.baidu.com' # 添加超链接
38
39 prs.save("G:/simple_ppt/test/blog_test.pptx")
```

生成的效果



注意，当给一个文本添加了超链接后，文字的颜色就无法指定了，会变成图中这种蓝色加下划线的样式

段落间距设置

可通过 `line_spacing` 指定

python

```
1 | p0.line_spacing = 1.3
```

文字大小自动改变

有时候我们要输入的文本太长，而文本框区域有限，此时可以指定文字的大小根据文本框的大小自动调整文字的大小

python

```
1 | from pptx.enum.text import MSO_AUTO_SIZE
2 |
3 | tf = tx_box.text_frame
4 | tf.auto_size = MSO_AUTO_SIZE.TEXT_TO_FIT_SHAPE
```

MSO_AUTO_SIZE还有其它三个参数

- NONE：不进行任何自动调整，文字可以超出文本框的边界
- SHAPE_TO_FIT_TEXT：根据文字的内容自动调整文本框的宽度和高度，这样可以保持文字的大小不变
- TEXT_TO_FIT_SHAPE：根据文本框的大小自动调整文字的大小，这样可以让文字完全填充文本框

参考：

<https://python-pptx.readthedocs.io/en/latest/api/enum/MsoAutoSize.html>

文本层级设置

文字层级一般用来处理段落的缩进，对内容进行层级管理，通过level来指定，每个paragraph的level默认就是0

python

```
1  p2 = tf.add_paragraph()  
2  p2.text = '第一层'  
3  p2.level = 0  
4  
5  p3 = tf.add_paragraph()  
6  p3.text = '第二层'  
7  p3.level = 1  
8  
9  p4 = tf.add_paragraph()  
10 p4.text = '第三层'  
11 p4.level = 2
```

效果如下



创建图片

使用add_picture 可以添加图片，指定对应的坐标即可

python

```
1 | img_path = 'G:/simple_ppt/res/picture.png'
2 |
```

```
slide.shapes.add_picture(img_path, left=Cm(2.58), top=Cm(6.16), width=Cm(8.3), height=Cm(5.13))
```

left和top表示图片左上角顶点分别距离幻灯片左边框和上边框的距离，width和height表示图片的宽和高

创建视频或音频

添加视频使用add_movie

python

```
1 video_path = 'G:/simple_ppt/res/movie.mp4'
2 slide.shapes.add_movie(video_path, Cm(11.66), Cm(6.22), Cm(8.11), Cm(5.07), mime_type='video/mp4')
```

视频显示的时候不会自动获取视频里的画面作为预览图，只会显示一个默认的喇叭图标，若想要根据视频的画面来生成预览图，可以借助OpenCV 工具来获取视频帧存为图片，然后通过poster_frame_image参数来指定

python

```
1 import cv2
2
3 video_path = 'G:/simple_ppt/res/movie.mp4'
4 cap = cv2.VideoCapture(video_path)
5 cap.set(cv2.CAP_PROP_POS_FRAMES, 0) # 设置要获取的帧
6 ret, frame = cap.read()
7 cv2.imwrite(save_poster_temp, frame)
8 cap.release()
9
10 slide.shapes.add_movie(video_path, Cm(11.66), Cm(6.22), Cm(8.11), Cm(5.07), mime_type='video/mp4', poster_
```


Python-pptx中并没有直接提供添加音频的方法，不过其实音频也可以通过add_movie来指定，只需要修改ime_type参数为audio/mp3

python

```
1 audio_path = 'G:/simple_ppt/res/audio.mp3'  
2 slide.shapes.add_movie(audio_path, Cm(19.77), Cm(6.22), Cm(8.11), Cm(5.07), mime_type='audio/mp3')
```

效果如下

Hello World!

这是第二行段落。第二行结尾直接添加文字

第一层

第二层

第三层



CSDN @ZHU_文涛

创建形状图形

Python-pptx中支持添加形状图形，也就是下面这些



可通过add_shape来添加

python

```
1 | from pptx.enum.shapes import MSO_SHAPE
2 |
3 |
```

```
slide.shapes.add_shape(MSO_SHAPE.STAR_5_POINT, left=Cm(28.83), top=Cm(6.87), width=Cm(3.7), height=Cm(3.7))
```

STAR_5_POINT表示一个五角星，更多形状参数可查看以下链接

<https://python-pptx.readthedocs.io/en/latest/api/enum/MsoAutoShapeType.html#msoautoshapetype>

形状图形的一些属性设置

python

```
1 shape = slide.shapes.add_shape(MSO_SHAPE.STAR_5_POINT, left=Cm(28.83), top=Cm(6.87), width=Cm(3.7), height=Cm(3.7))
2 shape.rotation = 45 # 旋转图标45°
3 shape.shadow.inherit = True # 是否取消倒影显示
4 shape.fill.solid() # 设置这个后才能通过下面的fore_color来设置颜色
5 shape.fill.fore_color.rgb = RGBColor(255, 255, 0) # 修改填充颜色
6 shape.line.color.rgb = RGBColor(255, 0, 0) # 修改边框颜色
7 shape.line.width = Cm(0.1) # 修改边框宽度
```

还可以通过dash_style来指定边框的线条样式

python

```
1 from pptx.enum.dml import MSO_LINE
2
3 shape.line.dash_style = MSO_LINE.DASH # 设置边框为虚线
```

MSO_LINE其它参数:

- MSO_LINE.SOLID: 实线
- MSO_LINE.DASH: 短划线
- MSO_LINE.DASH_DOT: 点划线
- MSO_LINE.DASH_DOT_DOT: 双点划线
- MSO_LINE.LONG_DASH: 长划线
- MSO_LINE.LONG_DASH_DOT: 长点划线
- MSO_LINE.ROUND_DOT: 圆点线
- MSO_LINE.SQUARE_DOT: 方点线

Hello World!

这是第二行段落。第二行结尾直接添加文字

第一层

第二层

第三层



想要图形的边框，可以使用以下方法将边框指定为透明

python

```
1 | shape.line.fill.background()
```

创建幻灯片背景

可以通过slide的background来指定纯色背景

python

```
1 | bg = slide.background
2 | bg.fill.solid()
3 | bg.fill.fore_color.rgb = RGBColor(219, 238, 244)
```

python-pptx库并没有直接提供设置图片作为幻灯片背景的方法，但可以通过将图片设置为铺满整个幻灯片来达到同样的效果

python

```
1 | img_path = "G:/bg_image.png"
2 | slide.shapes.add_picture(img_path, Cm(0), Cm(0), width=prs.slide_width, height=prs.slide_height)
```

slide_width和slide_height获取的分别是整张幻灯片的宽和高

这里要注意图片的层级问题，由于没有提供设置图片层级的方法，所以作为背景的图片应该放在构建幻灯片的第一位

Hello World!

这是第二行段落。第二行结尾直接添加文字

第一层

第二层

第三层



创建幻灯片备注信息

幻灯片底部的备注信息，在分屏预览时可用于提示演讲人更详细的幻灯片内容细节

通过has_notes_slide来判断幻灯片是否有备注，通过以下代码可以获得幻灯片的备注信息

python

```
1 | if slide.has_notes_slide:  
2 |     text_frame = slide.notes_slide.notes_text_frame  
3 |     print("备注文本: ", text_frame.text)
```

若想修改备注信息，直接通过text指定即可

python

```
1 | text_frame = slide.notes_slide.notes_text_frame  
2 | text_frame.text = "被修改的备注信息"
```

备注修改效果



总结

本篇文章介绍了如何使用python-pptx框架生成PPTX文件的方法，通过以上学习，可以掌握如何通过Python-pptx框架控制幻灯片大小、创建各种样式的文本、在幻灯片中添加图片、视频和音频，添加形状图形及如何修改它的样式，如何设置幻灯片的背景和备注信息等技能