安装：

2020年6月11日 10:05

1、找到pip3.exe所在的文件夹，复制路径

我的路径是： C:\Users\孙艺航\AppData\Local\Programs\Python\Python37\Scripts

2、按Win+R,输入CMD确定

3、进入后，先输入cd 路径 回车

4、输入 pip3 install python-pptx 回车

01.创建PPT写入标题与副标题

2020年6月11日 10:05

```
from pptx import Presentation
# 创建空白演示文稿
文件 = Presentation()
# 添加标题布局的幻灯片
# 文件.slide_layouts[0] 第一个母版中的第1个版式
# 添加新的一页，这个页面的版式就是括号里的
样式 = 文件.slide_layouts[0]
幻灯片 = 文件.slides.add_slide(母版)
# 设置标题和副标题
标题 = 幻灯片.shapes.title
副标题 = 幻灯片.placeholders[1]
标题.text = "跟着孙兴华学习Python办公自动化"
副标题.text = "PPT篇"
# 保存
文件.save('c:/1.pptx')
```

02.获取slide母版和shape样式

2020年6月24日 10:42

一、获取母版

```
from pptx import Presentation
文件 = Presentation("c:/练习1.pptx")  # "c:/1.pptx"
for 母版 in 文件.slides:
    print(母版)
```

二、获取样式

```
from pptx import Presentation
文件 = Presentation("c:/1.pptx")
for 母版 in 文件.slides:
    for 样式 in 母版.shapes:
        print(样式)
```

2.1 判断一个shape样式中是否存在文字

2020年6月24日 10:45

样式.has_text_frame 是否有文字

样式.text_frame 获取文字框

```
from pptx import Presentation
文件 = Presentation("c:/1.pptx")
for 母版 in 文件.slides:
    for 样式 in 母版.shapes:
        if 样式.has_text_frame:
            文字框 = 样式.text_frame
            print(文字框.text)
```

2.2 从shape样式中找paragraph段落

2020年6月24日 10:51

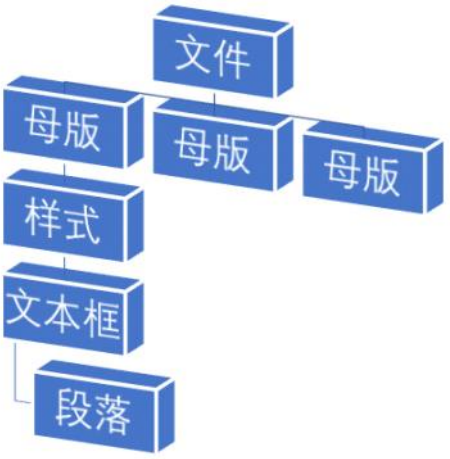
文本框.paragraphs 获取shape样式中的段落

```
from pptx import Presentation
文件 = Presentation("c:/1.pptx")
for 母版 in 文件.slides:
    for 样式 in 母版.shapes:
        if 样式.has_text_frame:
            文字框 = 样式.text_frame
            for 段落 in 文字框.paragraphs:
                print(段落.text)
```

2.3 将PPT中的文字部分转换成Word文档

2020年6月24日 10:59

```
from pptx import Presentation
from docx import Document
doc = Document()
文件 = Presentation("c:/练习3.pptx")
for 母版 in 文件.slides:
    for 样式 in 母版.shapes:
        if 样式.has_text_frame:  # 判断是否存在文字
            文字框 = 样式.text_frame
            for 段落 in 文字框.paragraphs:
                if 段落.text != "":  # 只导出不为空的段落
                    doc.add_paragraph(段落.text)
doc.save('c:/1.docx')
```



03. 获取母版slide中的信息【占位符的应用】

2020年6月24日 12:06

```
from pptx import Presentation
文件 = Presentation("c:/练习2.pptx")
# 文件.slide_layouts[0] 第一个母版中的第1个版式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
for 占位符 in 幻灯片.placeholders: # 遍历幻灯片中的占位符
    信息 = 占位符.placeholder_format
    print(f'索引：{信息.idx},名称:{占位符.name},类型:{信息.type}')
```

3.1 向占位符内填写内容

2020年6月24日 12:09

占位符.text = 字符串

文件.save(文件路径)

```
from pptx import Presentation
文件 = Presentation("c:/练习2.pptx")
# 文件.slide_layouts[0] 第一个母版中的第1个版式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
for 占位符 in 幻灯片.placeholders: # 遍历母版中的占位符
    信息 = 占位符.placeholder_format
    # print(f'索引：{信息.idx},名称:{占位符.name},类型:{信息.type}')
    占位符.text = f'{信息.idx}-{信息.type}'
文件.save('c:/1.pptx')
```

3.2 根据占位符ID确定要填哪里

2020年6月24日 12:35

母版.placeholders[占位符ID]

```
from pptx import Presentation
文件 = Presentation("c:/练习2.pptx")
# 文件.slide_layouts[0] 第一个母版中的第1个版式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
for 占位符 in 幻灯片.placeholders: # 遍历幻灯片中的占位符
    信息 = 占位符.placeholder_format
    # print(f'索引：{信息.idx},名称:{占位符.name},类型:{信息.type}')
    占位符.text = f'{信息.idx}-{信息.type}'
标题 = 幻灯片.placeholders[0]
副标题 = 幻灯片.placeholders[1]
文件.save('c:/1.pptx')
```

3.3 修改占位符里的内容

2020年6月24日 12:45

占位符.text = 字符串

```
from pptx import Presentation
文件 = Presentation("c:/练习2.pptx")
# 文件.slide_layouts[0] 第一个母版中的第1个版式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
for 占位符 in 幻灯片.placeholders: # 遍历幻灯片中的占位符
    信息 = 占位符.placeholder_format
    # print(f'索引：{信息.idx},名称:{占位符.name},类型:{信息.type}')
    占位符.text = f'{信息.idx}-{信息.type}'
标题 = 幻灯片.placeholders[0]
副标题 = 幻灯片.placeholders[1]
标题.text = '跟着孙兴华学习Python自动化'
副标题.text = 'PPT篇'
文件.save('c:/1.pptx')
```

04.添加段落

2020年6月24日 12:50

```
from pptx import Presentation
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[1])
样式 = 幻灯片.shapes
标题 = 样式.title
正文 = 样式.placeholders[1]
标题.text = '疫情期间如何不给国家添乱？'
文本框 = 正文.text_frame
文本框.text = '减少非必要的外出'
段落 = 文本框.add_paragraph()
段落.text = '带口罩，勤洗手'
段落 = 文本框.add_paragraph()
段落.text = '跟着孙兴华一起学习'
文件.save('c:/1.pptx')
```

4.1 对段落设置层级的关系

2020年6月24日 14:45

```
from pptx import Presentation
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[1])
样式 = 幻灯片.shapes
标题 = 样式.title
正文 = 样式.placeholders[1]
标题.text = '疫情期间如何不给国家添乱？'
文本框 = 正文.text_frame
文本框.text = '减少非必要的外出'
段落 = 文本框.add_paragraph()
段落.text = '带口罩，勤洗手'
段落.level = 1
段落 = 文本框.add_paragraph()
段落.text = '跟着孙兴华一起学习'
段落.level = 2
文件.save('c:/1.pptx')
```

05.添加文本框

2020年6月24日 14:47

母版.shapes.add_textbox(left,top,width,height)

```
from pptx import Presentation
from pptx.util import Cm,Pt # 单位
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # 这是样式是空白
left = top = width = height = Cm(4) # 左、顶、宽、高
文本框对象 = 幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华一起学习Python办公自动化'
段落 = 文本框.add_paragraph()
段落.text = 'PPT篇'
段落.font.bold = True
段落.font.size = Pt(40)
段落.font.name = '微软雅黑'
文件.save('c:/1.pptx')
```

5.1 文本框中文本的对齐

2020年6月26日 8:09

```
from pptx import Presentation
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE # 对齐文本和自动调整
from pptx.util import Cm

文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华学习Python办公自动化'

# 文本框.margin_bottom = Cm(0.1) #下边距
# 文本框.margin_left = Cm(4) #左边距
文本框.vertical_anchor = MSO_ANCHOR.MIDDLE #对齐文本方式：中部对齐
文本框.word_wrap = True #框中文字自动换行
文本框.auto_size = MSO_AUTO_SIZE.NONE  # 不自动调整

文件.save('c:/1.pptx')
```

```
from pptx.enum.text import MSO_ANCHOR  # 文本框中的文本对齐方式
```

MSO_ANCHOR.MIDDLE	中间的
MSO_ANCHOR.BOTTOM	底部对齐
MSO_ANCHOR.TOP	顶部对齐

```
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE  # 自动调整
```

MSO_AUTO_SIZE.NONE	不自动调整
MSO_AUTO_SIZE.TEXT_TO_FIT_SHAPE	溢出时缩排文字
MSO_AUTO_SIZE.SHAPE_TO_FIT_TEXT	根据文字调整形状大小

5.2 文本框的颜色

2020年6月26日 8:10

```
from pptx import Presentation
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE # 对齐文本和自动调整
from pptx.util import Cm
from pptx.dml.color import RGBColor
文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华学习Python办公自动化'

文本框.margin_bottom = Cm(0.1) #下边距
文本框.margin_left = 3 #左边距
文本框.vertical_anchor = MSO_ANCHOR.MIDDLE #对齐文本方式：中部对齐
文本框.word_wrap = True #框中文字自动换行
文本框.auto_size = MSO_AUTO_SIZE.NONE  # 不自动调整

# 填充颜色调整
填充 = 文本框对象.fill
填充.solid() # 纯色填充
填充.fore_color.rgb = RGBColor(255,255,0)

文件.save('c:/1.pptx')
```

5.3 边框调整

2020年6月26日 8:28

```
from pptx import Presentation
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE # 对齐文本和自动调整
from pptx.util import Cm
from pptx.dml.color import RGBColor
文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华学习Python办公自动化'

文本框.margin_bottom = Cm(0.1) #下边距
文本框.margin_left = Cm(3) #左边距
文本框.vertical_anchor = MSO_ANCHOR.MIDDLE #对齐文本方式：中部对齐
文本框.word_wrap = True #框中文字自动换行
文本框.auto_size = MSO_AUTO_SIZE.NONE  # 不自动调整

# 填充颜色调整
填充 = 文本框对象.fill
填充.solid() # 纯色填充
填充.fore_color.rgb = RGBColor(255,255,0)

# 文本框边框调整
边框 = 文本框对象.line
边框.color.rgb = RGBColor(255,0,0)
边框.width = Cm(0.5) # 边框有多粗，术语叫宽度

文件.save('c:/1.pptx')
```

5.4 段落对齐调整

2020年6月26日 8:43

```
from pptx import Presentation
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE # 对齐文本和自动调整
from pptx.util import Cm
from pptx.dml.color import RGBColor
from pptx.enum.text import PP_ALIGN # 段落对齐调整
文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华学习Python办公自动化'

文本框.margin_bottom = Cm(0.1) #下边距
文本框.margin_left = Cm(3) #左边距
文本框.vertical_anchor = MSO_ANCHOR.MIDDLE #对齐文本方式：中部对齐
文本框.word_wrap = True #框中文字自动换行
文本框.auto_size = MSO_AUTO_SIZE.NONE  # 不自动调整

# 填充颜色调整
填充 = 文本框对象.fill
填充.solid() # 纯色填充
填充.fore_color.rgb = RGBColor(255,255,0)

# 文本框边框调整
边框 = 文本框对象.line
边框.color.rgb = RGBColor(255,0,0)
边框.width = Cm(0.5) # 边框有多粗，术语叫宽度

# 段落对齐调整
段落 = 文本框.add_paragraph()
段落.text = 'PPT篇 python-pptx'
段落.alignment = PP_ALIGN.RIGHT
```

CENTER	居中
DISTRIBUTE	分散对齐
JUSTIFY	两端对齐
LEFT	靠左
RIGHT	靠右

文件.save('c:/1.pptx')

5.5 段落中增加块

2020年6月26日 8:46

```
from pptx import Presentation
from pptx.enum.text import MSO_ANCHOR,MSO_AUTO_SIZE # 对齐文本和自动调整
from pptx.util import Cm,Pt
from pptx.dml.color import RGBColor
from pptx.enum.text import PP_ALIGN # 段落对齐调整
文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
文本框.text = '跟着孙兴华学习Python办公自动化'

文本框.margin_bottom = Cm(0.1) #下边距
文本框.margin_left = Cm(3) #左边距
文本框.vertical_anchor = MSO_ANCHOR.MIDDLE #对齐文本方式：中部对齐
文本框.word_wrap = True #框中文字自动换行
文本框.auto_size = MSO_AUTO_SIZE.NONE  # 不自动调整

# 填充颜色调整
填充 = 文本框对象.fill
填充.solid() # 纯色填充
填充.fore_color.rgb = RGBColor(255,255,0)

# 文本框边框调整
边框 = 文本框对象.line
边框.color.rgb = RGBColor(255,0,0)
边框.width = Cm(0.5) # 边框有多粗，术语叫宽度

# 段落对齐调整
段落 = 文本框.add_paragraph()
段落.text = 'PPT篇 python-pptx'
段落.alignment = PP_ALIGN.RIGHT

# 增加文字块
段落1 = 文本框.add_paragraph()
```

```
块1 = 段落1.add_run()
块1.text = '孙兴华'
块1.font.size = Pt(48)
块2 = 段落1.add_run()
块2.text = '20岁'
块2.font.size = Pt(12)

文件.save('c:/1.pptx')
```

段落.level	段落缩进层级
段落.line_spacing	段落行间距
段落.runs	段落内的文字块
段落.space_after	段后距
段落.space_before	段前距

5.6 文字设置

2020年6月26日 9:10

记住一个层级关系，文字是在文本框的段落里写的，就可以了

```
from pptx import Presentation
from pptx.util import Cm,Pt
from pptx.dml.color import RGBColor

文件 = Presentation()
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])

left = top =  Cm(5)
height = Cm(3)
width = Cm(15)
文本框对象 =幻灯片.shapes.add_textbox(left,top,width,height)
文本框 = 文本框对象.text_frame
# 文本框.text = '跟着孙兴华学习Python办公自动化'   # 不在这里添加文字

段落 = 文本框.add_paragraph()
段落.text = '跟着孙兴华学习Python办公自动化'
段落.font.name = '微软雅黑'
段落.font.bold = True
段落.font.color.rgb = RGBColor(255,0,0)
段落.font.size = Pt(24)

文件.save('c:/1.pptx')
```

可以对段落或者块设置字体

.font.name	字体名称
.font.bold	是否加粗
.font.italic	是否斜体
.font.color	字体颜色
.font.size	字体大小

附：文件调整方法

2020年6月24日 15:04

方法	作用
all_caps	全部大写字母
bold	加粗
color	字体颜色
complex_script	是否为“复杂代码”
cs_bold	“复杂代码”加粗
cs_italic	“复杂代码”斜体
double_strike	双删除线
emboss	文本以凸出页面的方式出现
hidden	隐藏
imprint	印记
italic	斜体
name	字体
no_proof	不验证语法错误
outline	显示字符的轮廓
shadow	阴影
small_caps	小型大写字母
snap_to_grid	定义文档网格时对齐网络
strike	删除线
subscript	下标
superscript	上标
underline	下划线

06.添加图片

2020年6月24日 15:09

母版.shapes.add_picture(图片路径，距离左边，距离顶端，宽度，高度)

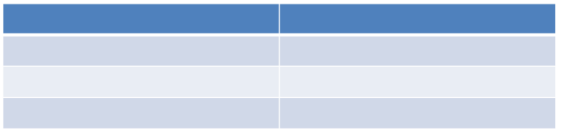
```
from pptx import Presentation
from pptx.util import Cm
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # 这是样式是空白
left = top = Cm(3)
height = Cm(12)
幻灯片.shapes.add_picture("c:/赵丽颖.jpg",left,top,height = height)
文件.save('c:/1.pptx')
```

07.添加表格

2020年6月24日 15:15

母版.add_tanle(rows,cols,left,top,width,height)

```
from pptx import Presentation
from pptx.util import Cm
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # 这是样式是空白
样式 = 幻灯片.shapes
rows,cols = 3,3          # 3行3列
left = top = Cm(5)       # 左和顶部
height = Cm(3)           # 高
width = Cm(18)           # 宽
表 = 样式.add_table(rows,cols,left,top,width,height).table
文件.save('c:/1.pptx')
```



7.1 调整表格大小

2020年6月24日 16:49

```
from pptx import Presentation
from pptx.util import Cm
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # 这是样式是空白
样式 = 幻灯片.shapes
rows,cols = 4,2
left = top = Cm(5)
height = Cm(3)
width = Cm(18)

表 = 样式.add_table(rows,cols,left,top,width,height).table
表.columns[0].width = Cm(5)
表.columns[1].width = Cm(3)
表.rows[0].height = Cm(1)

文件.save('c:/1.pptx')
```

7.2 导入表格

2020年6月24日 16:57

```
from pptx import Presentation
from pptx.util import Cm
import pandas as pd
文件 = Presentation()
# 文件.slide_layouts[0] 第一个母版中的第1个样式
# 添加新的一页，这个页面的版式就是括号里的
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # 这是样式是空白
样式 = 幻灯片.shapes
rows,cols = 4,2
left = top = Cm(5)
height = Cm(3)
width = Cm(18)

表 = 样式.add_table(rows,cols,left,top,width,height).table
表.columns[0].width = Cm(5)
表.columns[1].width = Cm(3)
表.rows[0].height = Cm(1)

数据 = pd.read_excel('c:/表格.xlsx',header=None)
# print(数据.iloc[0,0])
for 行 in range(rows):
    for 列 in range(cols):
        表.cell(行,列).text = str(数据.iloc[行,列])
文件.save('c:/1.pptx')
```

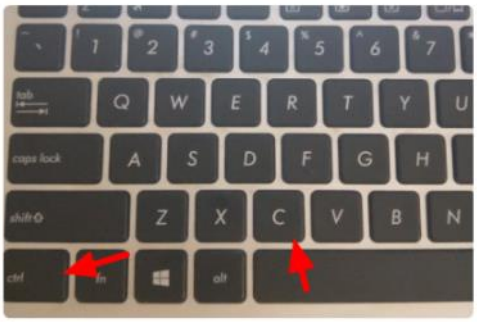
附：幻灯片做母版的方法

2020年6月25日 13:43

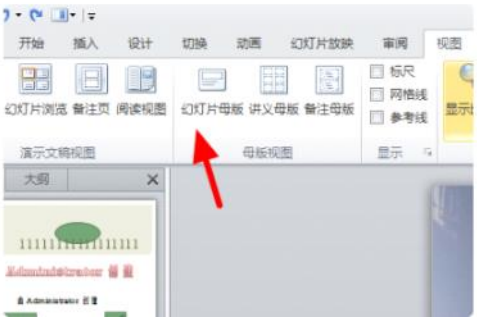
把ppt中某一页变成幻灯片母版的具体步骤如下：

需要准备的材料分别是：电脑、PPT。

1、首先打开需要编辑的PPT，按“Ctrl+C”复制想要变成幻灯片母版的那一页。

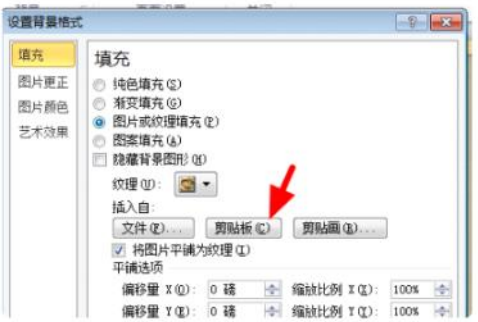


2、然后点击打开视图中的“幻灯片母版”。



3、然后在弹出来的窗口中点击打开背景样式中的“设置背景格式”。

4、然后在弹出来的窗口中点击选择图片或纹理填充中的“剪贴板”即可。



练习1：自动填充模版生成多个PPT文件

2020年6月24日　　17:08

```
from pptx import Presentation
import pandas as pd
from pptx.util import Cm
文件 = Presentation("c:/个人简历.pptx")
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
# for 占位符 in 母版.placeholders: # 遍历母版中的占位符
#     信息 = 占位符.placeholder_format
#     占位符.text = f'{信息.idx}'
p姓名 = 幻灯片.placeholders[11]
p性别 = 幻灯片.placeholders[12]
p年龄 = 幻灯片.placeholders[13]

数据 = pd.read_excel('c:/表格2.xlsx')
for 行号 in 数据['序号']:
    姓名 = 数据.iloc[行号-1,1]
    性别 = 数据.iloc[行号-1,2]
    年龄 = 数据.iloc[行号-1,3]
    p姓名.text = 姓名
    p性别.text = 性别
    p年龄.text = str(年龄)
    left = Cm(6.87)
    top = Cm(6.28)
    height = Cm(4)
    幻灯片.shapes.add_picture(f"c:/图片/{姓名}.png", left, top, height=height)
    文件.save(f'c:/{行号}.pptx')
```

练习2：生成一个PPT文件多页PPT

2020年6月25日 9:59

```
from pptx import Presentation
import pandas as pd
from pptx.util import Cm
文件 = Presentation("c:/个人简历.pptx")

数据 = pd.read_excel('c:/表格2.xlsx')
for 行号 in 数据['序号']:
    姓名 = 数据.iloc[行号-1,1]
    性别 = 数据.iloc[行号-1,2]
    年龄 = 数据.iloc[行号-1,3]
    幻灯片 = 文件.slides.add_slide(文件.slide_layouts[0])
    p姓名 = 幻灯片.placeholders[11]
    p性别 = 幻灯片.placeholders[12]
    p年龄 = 幻灯片.placeholders[13]
    p姓名.text = 姓名
    p性别.text = 性别
    p年龄.text = str(年龄)
    left = Cm(6.87)
    top = Cm(6.28)
    height = Cm(4)
    幻灯片.shapes.add_picture(f"c:/图片/{姓名}.png", left, top, height=height)
文件.save(f'c:/1.pptx')
```

08.四种基本图形

2020年6月26日 9:34

（柱形图、折线图、饼图、条形图）

8.1 柱形图

2020年6月26日 9:35

```
from pptx import Presentation
from pptx.util import Inches
from pptx.chart.data import ChartData
from pptx.enum.chart import XL_CHART_TYPE, XL_LABEL_POSITION

# 创建幻灯片----------------------------------------------
文件 = Presentation()  # 初始化 ppt 文档
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6])  # slide幻灯片
图表 = 幻灯片.shapes

# 定义图表数据----------------------------------------------
x = ['1季度', '2季度', '3季度', '4季度']
y = [8848, 5273, 8989, 8999]
z = [3568, 2572, 3948, 4105]

图表数据 = ChartData()
图表数据.categories = x  # 设置x轴
图表数据.add_series(name='销量', values=y)
图表数据.add_series(name='金额', values=z)

# 添加图表----------------------------------------------
left, top, width, height = Inches(0.5), Inches(1.5), Inches(9), Inches(6)
创建图表 = 图表.add_chart(chart_type=XL_CHART_TYPE.COLUMN_CLUSTERED,  # 簇状柱形图
                          x=left, y=top,  # 图表区的位置
                          cx=width, cy=height,  # 图表的宽和高
                          chart_data=图表数据)

文件.save('c:/1.pptx')
```

8.2 折线图

2020年6月26日 9:35

```
from pptx import Presentation
from pptx.util import Inches
from pptx.chart.data import ChartData
from pptx.enum.chart import XL_CHART_TYPE, XL_LABEL_POSITION

# 创建幻灯片----------------------------------------------
文件 = Presentation()  # 初始化 ppt 文档
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # slide幻灯片
图表 = 幻灯片.shapes


# 定义图表数据----------------------------------------------
x = ['1季度', '2季度', '3季度', '4季度']
y = [8848, 5273, 8989, 8999]
z = [3568, 2572, 3948, 4105]

图表数据 = ChartData()
图表数据.categories = x  # 设置x轴
图表数据.add_series(name='销量', values=y)
图表数据.add_series(name='金额', values=z)


# 添加图表----------------------------------------------
left, top, width, height = Inches(0.5), Inches(1.5), Inches(9), Inches(6)
创建图表 = 图表.add_chart(chart_type=XL_CHART_TYPE.LINE, # 图表类型
                          x=left, y=top,  # 图表区的位置
                          cx=width, cy=height,  # 图表的宽和高
                          chart_data=图表数据)
```

文件.save('c:/1.pptx')

8.3 饼图

2020年6月26日 9:35

```
from pptx import Presentation
from pptx.util import Inches, Pt
from pptx.chart.data import ChartData
from pptx.enum.chart import XL_CHART_TYPE, XL_LABEL_POSITION

# 创建幻灯片----------------------------------------------
文件 = Presentation()  # 初始化 ppt 文档
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # slide幻灯片
图表 = 幻灯片.shapes


# 定义图表数据----------------------------------------------
x = ['1季度', '2季度', '3季度', '4季度']
y = [8848, 5273, 8989, 8999]
z = [3568, 2572, 3948, 4105]

图表数据 = ChartData()
图表数据.categories = x  # 设置x轴
图表数据.add_series(name='销量', values=y)
图表数据.add_series(name='金额', values=z)


# 添加图表----------------------------------------------
left, top, width, height = Inches(0.5), Inches(1.5), Inches(9), Inches(6)
创建图表 = 图表.add_chart(chart_type=XL_CHART_TYPE.PIE, # 图表类型
                          x=left, y=top,  # 图表区的位置
                          cx=width, cy=height,  # 图表的宽和高
```

```
chart_data=图表数据)

设置图表 = 创建图表.chart
布局 = 设置图表.plots[0]
# 设置数据标签
布局.has_data_labels = True    # 显示数据标签
数据标题 = 布局.data_labels    # 获取数据标签控制类
数据标题.show_category_name = True    # 是否显示类别名称
数据标题.show_value = False    # 是否显示值
数据标题.show_percentage = True    # 是否显示百分比
数据标题.number_format = '0.0%'    # 标签的数字格式
数据标题.position = XL_LABEL_POSITION.INSIDE_END    # 标签位置
数据标题.font.name = 'Arial'
数据标题.font.size = Pt(14)

# 设置图表标题
设置图表.has_title = True    # 显示标题
标题 = 设置图表.chart_title.text_frame.add_paragraph()
标题.text = '销量占比'    # 标题内容
标题.font.size = Pt(16)    # 字体大小

文件.save('c:/1.pptx')
```

8.4 条形图

2020年6月26日 9:35

```
from pptx import Presentation
from pptx.util import Inches, Pt
from pptx.chart.data import ChartData
from pptx.enum.chart import XL_CHART_TYPE, XL_LABEL_POSITION

# 创建幻灯片----------------------------------------------
文件 = Presentation()  # 初始化 ppt 文档
幻灯片 = 文件.slides.add_slide(文件.slide_layouts[6]) # slide幻灯片
图表 = 幻灯片.shapes


# 定义图表数据----------------------------------------------
x = ['1季度', '2季度', '3季度', '4季度']
y = [8848, 5273, 8989, 8999]
z = [3568, 2572, 3948, 4105]

图表数据 = ChartData()
图表数据.categories = x  # 设置x轴
图表数据.add_series(name='销量', values=y)
图表数据.add_series(name='金额', values=z)


# 添加图表----------------------------------------------
left, top, width, height = Inches(0.5), Inches(1.5), Inches(9), Inches(6)
创建图表 = 图表.add_chart(chart_type=XL_CHART_TYPE.BAR_CLUSTERED, # 图表类型
                          x=left, y=top,  # 图表区的位置
                          cx=width, cy=height,  # 图表的宽和高
                          chart_data=图表数据)

设置图表 = 创建图表.chart
布局 = 设置图表.plots[0]
# 设置数据标签
```

```
布局.has_data_labels = True    # 显示数据标签
数据标题 = 布局.data_labels    # 获取数据标签控制类
数据标题.number_format = '#,#'    # 标签的数字格式
数据标题.position = XL_LABEL_POSITION.OUTSIDE_END    # 标签位置
数据标题.font.name = 'Arial'
数据标题.font.size = Pt(14)

文件.save('c:/1.pptx')
```

09.删除指定页面

2020年6月26日 11:38

一、删除一张PPT指定页面

```
from pptx import Presentation
文件 = Presentation('c:/练习1.pptx')
页 = list(文件.slides._sldIdLst) # 获取页的列表
# print(len(页))
文件.slides._sldIdLst.remove(页[0]) # 指定删除页
# 文件.slides._sldIdLst.remove(页[len(页)-1]) # 删除最后一页
文件.save('c:/2.pptx')
```

二、删除指定目录下所有的PPT指定页面

```
from pptx import Presentation
import os
for 文件夹路径,子文件夹列表,文件列表 in os.walk('c:/ppt'):
    for i in 文件列表:
        文件 = Presentation(文件夹路径+'/'+i)
        页 = list(文件.slides._sldIdLst)  # 获取页的列表
        文件.slides._sldIdLst.remove(页[len(页)-1]) # 删除最后一页
        文件.save(f'c:/孙兴华/{i}')
```

附送：word\Excel\ppt 批量转pdf工具

2020年6月26日 12:03

```
import os
from win32com.client import Dispatch, constants, gencache, DispatchEx

class PDFConverter:
    def __init__(self, pathname, output, export='.'):
        self._handle_postfix = ['doc', 'docx', 'ppt', 'pptx', 'xls', 'xlsx']
        self._filename_list = list()
        self._export_folder = os.path.join(os.path.abspath('.'), output)
        if not os.path.exists(self._export_folder):
            os.mkdir(self._export_folder)
        self._enumerate_filename(pathname)

    def _enumerate_filename(self, pathname):
        '''
        读取所有文件名
        '''
        full_pathname = os.path.abspath(pathname)
        if os.path.isfile(full_pathname):
            if self._is_legal_postfix(full_pathname):
                self._filename_list.append(full_pathname)
            else:
                raise TypeError('文件 {} 后缀名不合法！仅支持如下文件类型：{}。'.format(pathname, '、'.join(self._handle_postfix)))
        elif os.path.isdir(full_pathname):
            for relpath, _, files in os.walk(full_pathname):
                for name in files:
                    filename = os.path.join(full_pathname, relpath, name)
                    if self._is_legal_postfix(filename):
                        self._filename_list.append(os.path.join(filename))
        else:
            raise TypeError('文件/文件夹 {} 不存在或不合法！'.format(pathname))

    def _is_legal_postfix(self, filename):
        return filename.split('.')[-1].lower() in self._handle_postfix and not os.path.basename(filename).startswith('~')

    def run_conver(self):
        '''
        进行批量处理，根据后缀名调用函数执行转换
        '''
        print('需要转换的文件数：', len(self._filename_list))
        for filename in self._filename_list:
            postfix = filename.split('.')[-1].lower()
            funcCall = getattr(self, postfix)
            print('原文件：', filename)
            funcCall(filename)
        print('转换完成！')

    def doc(self, filename):
```

```
    '''
    doc 和 docx 文件转换
    '''
    name = os.path.basename(filename).split('.')[0] + '.pdf'
    exportfile = os.path.join(self._export_folder, name)
    print('保存 PDF 文件：', exportfile)
    gencache.EnsureModule('{00020905-0000-0000-C000-000000000046}', 0, 8, 4)
    w = Dispatch("Word.Application")
    doc = w.Documents.Open(filename)
    doc.ExportAsFixedFormat(exportfile, constants.wdExportFormatPDF,
            Item=constants.wdExportDocumentWithMarkup,
            CreateBookmarks=constants.wdExportCreateHeadingBookmarks)

    w.Quit(constants.wdDoNotSaveChanges)

def docx(self, filename):
    self.doc(filename)

def xls(self, filename):
    '''
    xls 和 xlsx 文件转换
    '''
    name = os.path.basename(filename).split('.')[0] + '.pdf'
    exportfile = os.path.join(self._export_folder, name)
    xlApp = DispatchEx("Excel.Application")
    xlApp.Visible = False
    xlApp.DisplayAlerts = 0
    books = xlApp.Workbooks.Open(filename,False)
    books.ExportAsFixedFormat(0, exportfile)
    books.Close(False)
    print('保存 PDF 文件：', exportfile)
    xlApp.Quit()

def xlsx(self, filename):
    self.xls(filename)

def ppt(self, filename):
    '''
    ppt 和 pptx 文件转换
    '''
    name = os.path.basename(filename).split('.')[0] + '.pdf'
    exportfile = os.path.join(self._export_folder, name)
    gencache.EnsureModule('{00020905-0000-0000-C000-000000000046}', 0, 8, 4)
    p = Dispatch("PowerPoint.Application")
    ppt = p.Presentations.Open(filename, False, False, False)
    ppt.ExportAsFixedFormat(exportfile, 2, PrintRange=None)
    print('保存 PDF 文件：', exportfile)
    p.Quit()
```

```
    def pptx(self, filename):
        self.ppt(filename)

if __name__ == "__main__":
    input_folder = 'c:/office'    # 你的文件放在哪里了?
    output_folder = "c:/pdf"   # 转换后想放在哪里?
    pathname = os.path.join(os.path.abspath('.'), input_folder)
    pdfConverter = PDFConverter(pathname,output_folder)
    pdfConverter.run_conver()
```