**Reporter测试记录文档**

这个方法可以用来生成完整的回测报告。以文字、图表为主，统一格式

# 一、研究背景介绍

python-docx 是用于创建和更新Microsoft Word（.docx）文件的Python库。

## 1) 无序项目

* 无序项目1
* 无序项目2
* 无序项目3

## 2) 有序项目

1. 有序项目1
2. 有序项目2
3. 有序项目3

# 二、主要研究结果

Python中可以用docx来生成word文档，docx中可以自定义文字的大小和字体等。

段落是Word中的一个块级对象，在其所在容器的左右边界内显示文本，当文本超过右边界时自动换行。段落的边界通常是页边界，也可以是分栏排版时的栏边界，或者表格单元格中的边界。段落格式用于控制段落在其容器（例如页、栏、单元格）中的布局，例如对齐方式、左缩进、右缩进、首行缩进、行距、段前距离、段后距离、换页方式、Tab键字符格式等。

newfile = docx.Document()  
 newfile.styles['Normal'].font.name = 'Times New Roman'  
 newfile.styles['Normal'].\_element.rPr.rFonts.set(qn('w:eastAsia'), u'宋体')

## 1) 无序项目

* 无序项目1
* 无序项目2
* 无序项目3

## 2) 有序项目

1. 有序项目1
2. 有序项目2
3. 有序项目3

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

|  |  |  |
| --- | --- | --- |
| x | y | z |
| 1 | 2 | 3 |
| 1 | 2 | 3 |
| 1 | 2 | 3 |

# 三、讨论

Python中可以用docx来生成word文档，docx中可以自定义文字的大小和字体等。

段落是Word中的一个块级对象，在其所在容器的左右边界内显示文本，当文本超过 右边界时自动换行。段落的边界通常是页边界，也可以是分栏排版时的栏边界，或者表格单元格中的边界。段落格式 用于控制段落在其容器（例如页、栏、单元格）中的布局，例如对齐方式、左缩进、右缩进、首行缩进、行距、段前 距离、段后距离、换页方式、Tab键字符格式等。

# 四、源码

class WordWriter:  
 """用 Word 文档记录信息"""  
  
 def \_\_init\_\_(self, file\_docx=None):  
 self.file\_docx = file\_docx  
 if file\_docx and os.path.exists(file\_docx):  
 self.document = docx.Document(file\_docx)  
 else:  
 self.document = docx.Document()  
 self.document.core\_properties.author = "Reporter"  
  
 # 设置正文样式  
 self.document.styles["Normal"].font.name = 'Times New Roman'  
 self.document.styles["Normal"].element.rPr.rFonts.set(qn('w:eastAsia'), '宋体')  
  
 def add\_title(self, text):  
 self.document.core\_properties.title = text  
 title\_ = self.document.add\_heading(level=0)  
 title\_.alignment = WD\_PARAGRAPH\_ALIGNMENT.CENTER  
 title\_run = title\_.add\_run(text)  
 title\_run.font.size = Pt(22)  
 title\_run.font.bold = True  
 title\_run.font.name = 'Times New Roman'  
 title\_run.element.rPr.rFonts.set(qn('w:eastAsia'), '微软雅黑')  
  
 def add\_heading(self, text, level=1):  
 if level == 1:  
 size = 18  
 elif level == 2:  
 size = 16  
 else:  
 size = 14  
 title\_ = self.document.add\_heading(level=level)  
 title\_run = title\_.add\_run(text)  
 title\_run.font.size = Pt(size)  
 title\_run.font.name = 'Times New Roman'  
 title\_run.element.rPr.rFonts.set(qn('w:eastAsia'), '微软雅黑')  
 title\_run.font.color.rgb = RGBColor(0, 0, 0)  
  
 def add\_paragraph(self, text, style=None, bold=False):  
 p = self.document.add\_paragraph(style=style)  
 p.paragraph\_format.left\_indent = Cm(0)  
 p.paragraph\_format.first\_line\_indent = Cm(0.74)  
 p.paragraph\_format.line\_spacing = 1.25  
 p.paragraph\_format.space\_before = Pt(8)  
 p.paragraph\_format.space\_after = Pt(8)  
  
 text = p.add\_run(text)  
 text.bold = bold  
 text.font.name = 'Times New Roman'  
 text.element.rPr.rFonts.set(qn('w:eastAsia'), '宋体')  
 text.font.size = Pt(12)  
  
 def add\_df\_table(self, df: pd.DataFrame, style='Table Grid'):  
 """添加数据表  
  
 https://www.jianshu.com/p/93e0df92cf16  
 :param df: 数据表  
 :param style: 表格样式  
 :return:  
 """  
 if df.empty:  
 print(f"add\_df\_table error: 传入的数据表是空的")  
 return  
  
 table = self.document.add\_table(rows=1, cols=df.shape[1], style=style)  
 # 设置整个表格字体属性  
 table.style.font.size = Pt(12)  
 table.style.font.color.rgb = RGBColor(0, 0, 0)  
 table.style.paragraph\_format.alignment = WD\_PARAGRAPH\_ALIGNMENT.CENTER  
  
 records = df.to\_dict("records")  
 columns = df.columns.to\_list()  
  
 hdr\_cells = table.rows[0].cells  
 for i, c in enumerate(columns):  
 hdr\_cells[i].text = c  
  
 for row in records:  
 row\_cells = table.add\_row().cells  
 for i, c in enumerate(columns):  
 row\_cells[i].text = str(row[c])  
  
 def add\_picture(self, file, width=None, height=None, alignment='center') -> None:  
 """写入图片到文档中  
  
 :param file: 图片文件路径  
 :param width: 图片宽度，默认单位 cm  
 :param height: 图片高度，默认单位 cm  
 :param alignment: 图片对齐，默认 center  
 :return:  
 """  
 alignment\_map = {  
 'center': WD\_PARAGRAPH\_ALIGNMENT.CENTER,  
 'left': WD\_PARAGRAPH\_ALIGNMENT.LEFT,  
 'right': WD\_PARAGRAPH\_ALIGNMENT.RIGHT,  
 }  
 if isinstance(width, int):  
 width = Cm(width)  
  
 if isinstance(height, int):  
 height = Cm(height)  
  
 paragraph = self.document.add\_paragraph()  
 paragraph.alignment = alignment\_map[alignment]  
 run = paragraph.add\_run("")  
 run.add\_picture(file, width, height)  
  
 def add\_page\_break(self):  
 """添加分页符"""  
 self.document.add\_page\_break()  
  
 def save(self, file\_docx=None):  
 """保存结果到文件"""  
 if not file\_docx:  
 file\_docx = self.file\_docx  
 self.document.save(file\_docx)



