

★ 00.常用代码【方便查询】：

2020年6月7日 1:19

库名	作用
xlrd	从excel中读取数据，支持xls,xlsx
xlwt	从excel进行修改操作，不支持对xlsx格式的修改
xlutils	在xlrd和xlwt中，对一个已存在的文件进行修改
openpyxl	主要针对xlsx格式的excel进行读取和编辑
pandas	可对csv进行操作，主要用于大数据分析

01.安装

2020年6月7日 1:36

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 openpyxl 回车

pip install openpyxl

或

pip3 install openpyxl

今后如果遇到库出问题解决方法：

(1) 卸载出问题的库

pip uninstall pandas

pip uninstall openpyxl

(2) 重新安装

python -m pip install pandas -i <https://pypi.tuna.tsinghua.edu.cn/simple>

python -m pip install openpyxl -i <https://pypi.tuna.tsinghua.edu.cn/simple>

02.工作簿

2020年6月7日 1:37

1、创建工作簿

工作簿=openpyxl.**W**orkbook('文件名称.xlsx')

工作簿.save(路径)

2、打开工作簿

工作簿 = openpyxl.load_workbook('文件名称.xlsx')

工作表 = 工作簿 ['工作表名']

属性	作用
<input checked="" type="checkbox"/> active	获取当前活跃的Worksheet
<input checked="" type="checkbox"/> worksheets	以列表的形式返回所有的Worksheet(表格)
<input type="checkbox"/> data_only	默认为False,为True时只读取数据不显示公式
<input type="checkbox"/> read_only	判断是否以read_only模式打开Excel文档
<input type="checkbox"/> encoding	获取文档的字符集编码
<input type="checkbox"/> properties	获取文档的元数据，如标题，创建者，创建日期等
<input type="checkbox"/> sheetnames	获取工作簿中的表（列表）

方法	作用
<input type="checkbox"/> 工作簿.sheetnames	获取所有表格的名称
<input type="checkbox"/> 工作簿['工作表名']	通过表格名称获取Worksheet对象
<input type="checkbox"/> 工作簿.active	获取活跃的表格
<input checked="" type="checkbox"/> remove	删除一个工作表对象【对象】
<input checked="" type="checkbox"/> create_sheet	创建一个空的表格【表名】
<input checked="" type="checkbox"/> copy_worksheet	在Workbook内拷贝表格【对象】

03.工作表

2020年6月7日 15:17

从工作簿中获取工作表

- (1) 获取工作表的名称
工作簿 .sheetnames
- (2) 指定工作表
工作簿 ['工作表名称']
- (3) 激活第一个工作表
工作簿.active
- (4) 获取工作表名称
工作表.title

属性	作用
<input checked="" type="checkbox"/> title	工作表的名称
<input type="checkbox"/> dimensions	表格的大小，这里的大小是指含有数据的表格的大小，即：左上角的坐标:右下角的坐标
<input type="checkbox"/> max_row	表格的最大行
<input type="checkbox"/> min_row	表格的最小行
<input type="checkbox"/> max_column	表格的最大列
<input type="checkbox"/> min_column	表格的最小列
<input type="checkbox"/> rows	按行获取单元格(Cell对象) - 生成器 工作表.rows
<input type="checkbox"/> columns	按列获取单元格(Cell对象) - 生成器 工作表.columns
<input type="checkbox"/> freeze_panes	冻结窗格 工作表.freeze_panes = "C3"
<input type="checkbox"/> values	按行获取表格的内容(数据) - 生成器

freeze_panes，参数比较特别，主要用于在表格较大时冻结顶部的行或左边的行。对于冻结的行，在用户滚动时，是始终可见的，可以设置为一个Cell对象或一个端元个坐标的字符串，单元格上面的行和左边的列将会冻结(单元格所在的行和列不会被冻结)。例如我们要冻结第一行那么设置A2为freeze_panes，如果要冻结第一列，freeze_panes取值为B1，如果要同时冻结第一行和第一列，那么需要设置B2为freeze_panes，freeze_panes值为none时 表示 不冻结任何列。

方法	作用
<input type="checkbox"/> iter_rows	按行获取所有单元格，内置属性有(min_row,max_row,min_col,max_col)
<input type="checkbox"/> iter_columns	按列获取所有的单元格
<input type="checkbox"/> append	在表格末尾添加数据
<input type="checkbox"/> merged_cells	合并多个单元格
<input type="checkbox"/> unmerged_cells	移除合并的单元格

04.单元格

2020年6月7日 13:44

属性	作用
row	单元格所在的行
column	单元格所在的列
value	单元格的值
coordinate	单元格的坐标

01.工作簿新建、保存和打开

2020年6月7日 16:16

一、新建和保存

```
import openpyxl as vb
```

```
路径 = r'c:/孙兴华.xlsx'
```

```
工作簿 = vb.Workbook(路径)
```

```
工作簿.save(路径1)
```

注意:

1. 使用Workbook新建, 首字母大写
2. 别忘记保存

二、打开工作簿

```
import openpyxl as vb
```

```
路径 = r'c:/001.xlsx'
```

```
工作簿 = vb.load_workbook(路径)
```

```
工作表 = 工作簿['1月']
```

```
print(工作表)
```

注意:

1. 指定工作簿
2. 指定工作表

关于工作表的相关属性, 查阅常用代码04

三、激活工作表, 默认第0个

```
工作表 = 工作簿.active
```

02.工作表的创建、删除、复制

2020年6月7日 17:23

一、显示工作簿中的所有工作表和表名

```
import openpyxl as vb
路径 = r'c:/001.xlsx'
工作簿 = vb.load_workbook(路径)
显示所有工作表 = 工作簿.worksheets
for 工作表 in 显示所有工作表:
    print(工作表.title)
```

步骤:

1. 给工作表指定对象
2. 用remove删除这个对象

二、删除指定工作表

```
import openpyxl as vb
路径 = r'c:/001.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['4月']
工作簿.remove(工作表)
工作簿.save(路径)
```

三、新建指定工作表

```
import openpyxl as vb
路径 = r'c:/001.xlsx'
工作簿 = vb.load_workbook(路径)
工作簿.create_sheet('4月')
工作簿.save(路径)
```

四、复制指定工作表

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
```

```
工作簿 = vb.load_workbook(路径)
复制表 = 工作簿.copy_worksheet(工作簿['9月']) # 这里是工作表对象
复制表.title = '我是刚复制的表'
工作簿.save(路径)
```


练习：

2020年6月7日 19:41

练习1：新建100张工作表


```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.Workbook(路径) # 打开工作簿
for i in range(1,101):
    工作簿.create_sheet(str(i) + '月')
工作簿.save(路径)
```

练习2：除了9月份的工作表以外都删除

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径) # 打开工作簿
显示所有工作表 = 工作簿.worksheets
for 工作表 in 显示所有工作表:
    if 工作表.title != '9月':
        工作表 = 工作簿[工作表.title]
        工作簿.remove(工作表)
工作簿.save(路径)
```

如果表名是：

北京-01, 北京-02, 上海-01, 上海-02
if 工作表.title.split("-")[0] != '北京':



练习3：批量修改工作表的名称

```
import openpyxl as vb
路径 = r'c:/模板.xlsx'
工作簿 = vb.load_workbook(路径)
显示所有工作表 = 工作簿.worksheets
for 工作表 in 显示所有工作表:
    工作表.title = '北京' + 工作表.title
工作簿.save(路径)
```

练习4：将模板批量复制


```
import openpyxl as vb
路径 = r'c:/模板.xlsx'
工作簿 = vb.load_workbook(路径)
for i in range(1,32):
    复制表 = 工作簿.copy_worksheet(工作簿['Sheet1'])
    复制表.title = '7月' + str(i) + '日'
工作簿.remove(工作簿['Sheet1']) # 把模板删除
工作簿.save(路径)
```

03.工作表中单元格的操作

2020年6月7日 16:32

一、获取一个单元格的值

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
单元格 = 工作表['A1'].value
print(单元格)
```



VBA中表示单元格:

range('A1')

cells(行, 列)

单元格 = 工作簿['1月']['A1'].value

单元格 = 工作表.cell(row=1,column=1).value

工作表 = 工作簿.worksheets[0]

二、获取第2列1、3、5、7行的数据

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for i in range(1,8,2):
    print(i,工作表.cell(row=i,column=2).value)
```

3.1 获取一个区域的单元格

2020年6月9日 20:20

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
单元格区域 = 工作表['A1:C10']
```

按行的优势：这一行所有使用的单元格都会获取到，新增列也可以获取到

```
# 单元格区域 = 工作表['1:10']
# 单元格区域 = 工作表['A:C']
```

按列的优势：取完一列的数据之后再取下一列。【这是唯一的按列取数据】

```
for 数据 in 单元格区域: # 循环每行
    for 单元格 in 数据: # 循环每个单元格
        print(单元格.value)
```

★ 例1：使用list(工作表.values)

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
print(list(工作表.values))
```

只能对整个工作表操作，不能对单独区域操作
print(list(工作表.values)[1:3])
可以用切片的方式截取某一段数据

★ 例2：.iter_rows(min_row=最低行数, max_row=最高行数, min_col=最低列数, max_col=最高列数)

[一般情况下只需要定位起点]

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
# 行和列的范围 iter_cols按列
for 行 in 工作表.iter_rows(min_row=1,max_row=10,min_col=1,max_col=3):
    for 单元格 in 行:
        print(单元格.value)
```

或：

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for 列 in 工作表.iter_cols(min_row=1,max_row=10,min_col=1,max_col=3):
    for 单元格 in 列:
        print(单元格.value)
```

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
# 行和列的范围 iter_cols按列
for 行 in 工作表['A1:C10']:
    for 单元格 in 行:
        print(单元格.value)
```

3.1.1 获取每一行、每一列

2020年6月10日 16:14

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
# 因为按行, 所以返回A1, B1, C1这样的顺序
for 行 in 工作表.rows:
    for 单元格 in 行:
        print(单元格.value)
# A1, A2, A3这样的顺序
for 列 in 工作表.columns:
    for 单元格 in 列:
        print(单元格.value)
```

3.1.2 列字母和数字之间的转换

2020年6月10日 16:18

```
import openpyxl as vb
```

```
# 根据列的数字返回字母
```

```
数字转字母 = vb.utils.get_column_letter(2)
```

```
print(数字转字母)
```

```
# 根据字母返回列的数字
```

```
字母转数字 = vb.utils.column_index_from_string('D')
```

```
print(字母转数字)
```

→ 使用这些函数时，不必加载一个工作簿

3.2 读取数据

2020年6月7日 12:47

```
import openpyxl as vb
路径 = r'c:/测试2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
最大行 = 工作表.max_row
最大列 = 工作表.max_column
A1单元格的值 = 工作表['A1'].value
A1单元格的值2 = 工作表.cell(1,1).value
A1单元格的列 = 工作表['A1'].column
A1单元格的行 = 工作表['A1'].row
# 获取C列所有数据
列表1 = []
for i in 工作表['C']:
    列表1.append(i.value)
# 获取第1行所有数据
列表2 = []
for i in 工作表[1]:
    列表2.append(i.value)
# 获取所有数据
列表3 = []
for 行 in 工作表.rows:
    for 单元格 in 行:
        列表3.append(单元格.value)
print(列表3)
```

3.3 写入数据

2020年6月10日 15:59

一、向一个单元格写入数据

```
import openpyxl as vb
路径 = r'c:/测试2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
工作表.cell(1,5,value='孙兴华')
工作表['E2']='兴华'
工作簿.save(路径)
```

二、在最后一列写入数据

```
工作表.append(列表)
```

三、向一个区域内写入数据

```
for 行 in 工作表['A1:B4']:
    for 单元格 in 行:
        单元格.value = 520
```

3.4 行、列的插入与删除

2020年6月10日 16:27

插入列：工作表.insert_cols(位置,列数)，其中位置是指在工作表的第几列前插入多少列。
插入行：工作表.insert_rows(位置,行数)，其中位置是指在工作表的第几行前插入多少行。
删除列：工作表.delete_cols(位置,列数)，从指定位置开始向后删除指定的列数。
删除行：工作表.delete_rows(位置,行数)，从指定位置开始向下删除指定的行数。

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
工作表.insert_cols(idx=2,amount=5)
工作表.insert_rows(idx=2,amount=5)
工作表.delete_cols(idx=2,amount=5)
工作表.delete_rows(idx=2,amount=5)
工作簿.save(路径)
```


3.5 移动单元格

2020年6月10日 16:46

```
import openpyxl as vb
路径 = r'c:/测试.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
# rows和cols正数为向下或向右、负数为向左或向上
工作表.move_range("A1:C3",rows=10,cols=10)
工作簿.save(路径)
```

3.6 冻结单元格

2020年6月10日 16:50

```
import openpyxl as vb  
路径 = r'c:/测试.xlsx'  
工作簿 = vb.load_workbook(路径)  
工作表 = 工作簿['Sheet1']  
工作表.freeze_panes = "C3"  
工作簿.save('c:/1.xlsx')
```

练习1：每张工作表的固定单元格求和

2020年6月10日 17:21

例1：将每张工作表中指定单元格的值汇总

```
import openpyxl as vb
路径 = r'c:/练习1.xlsx'
工作簿 = vb.load_workbook(路径)
列表 = []
for 工作表 in 工作簿.worksheets:
    列表.append(工作表['D6'].value)
print(sum(列表))

求和 = sum([工作表['D6'].value for 工作表 in 工作簿.worksheets])
print(求和)
```

第06课 for循环、公共操作与推导式

什么是推导式：就是**简化代码**的（如果你不会，你的代码写的多一些，如果你学会了，你的代码写的少一些）

创建一个由0-10组成的列表

列表名 = [] # 创建一个空列表

```
for 变量名 in range(0,11):
    列表名.append(变量名)
print(列表名)
```

一、列表推导式

列表名 = [变量名 for 变量名 in range(0,11)]



将遍历好的0-10这11个数，每次保存一次，就生成了[0,1,2,...,10]

主讲：孙兴华

P71

练习2：按行或列求和

2020年6月10日 17:22

准备：将每一行数据以一个列表的方式显示

```
import openpyxl as vb
路径 = r'c:/练习2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
最大行号 = 工作表.max_row
最大列号 = 工作表.max_column
for 行 in 工作表.iter_rows(1,最大行号,1,最大列号):
    print([单元格.value for 单元格 in 行])
```

for 行 in 工作表.rows:

练习2:

```
import openpyxl as vb
路径 = r'c:/练习2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for 行 in 工作表.rows:
    print([单元格.value for 单元格 in 行][1:])
```

使用切片把非数值切出去

例1：按行求和

```
import openpyxl as vb
路径 = r'c:/练习2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for 行 in list(工作表.rows)[1:]:
    数据 = [单元格.value for 单元格 in 行]
    print(数据[0],sum(数据[1:]))
```

例2：按列求和


```
import openpyxl as vb
路径 = r'c:/练习2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for 列 in list(工作表.columns)[1:]:
    数据 = [单元格.value for 单元格 in 列]
    print(数据[0],sum(数据[1:]))
```

练习3：成绩为空的标记缺考

2020年6月10日 17:22

```
import openpyxl as vb
路径 = r'c:/练习3.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
for 行 in 工作表.iter_rows(min_row=2,min_col=2):
    for 单元格 in 行:
        if 单元格.value == None:
            单元格.value = '缺考'
工作簿.save('c:/1.xlsx')
```

这里的空值用None表示



练习4：总分大于270分是优秀

2020年6月10日 17:22

```
import openpyxl as vb
路径 = r'c:/练习2.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
工作表['E1'] = '评价'
范围 = 工作表.iter_rows(min_row=2,min_col=2)
for 行 in 范围:
    总分 = sum([单元格.value for 单元格 in 行][: -1])
    if 总分 >= 270:
        行[-1].value = '优秀'
工作簿.save('c:/1.xlsx')
```

练习5：筛选总分大于270分

2020年6月11日 7:25

```
import openpyxl as vb
路径 = 'c:/练习5.xlsx'
工作簿=vb.load_workbook(路径)
工作表=工作簿['Sheet1']
for 行号 in range(工作表.max_row,1,-1):
    总分 = sum([单元格.value for 单元格 in 工作表[行号]
[1:]])
    if 总分 <= 270:
        工作表.delete_rows(行号)
工作簿.save('c:/1.xlsx')
```

练习6：按指定列拆分成多个工作表


2020年6月10日 21:40

```
import openpyxl as vb
路径 = 'c:/练习6.xlsx'
工作簿=vb.load_workbook(路径)
工作表=工作簿['Sheet1']
范围=工作表.iter_rows(min_row=2)
字典={}
for 行 in 范围:
    每行数据=[单元格.value for 单元格 in 行]
    if 每行数据[1] in 字典.keys():
        字典[每行数据[1]] += [每行数据]
    else:
        字典[每行数据[1]] = [每行数据]
for 键,值 in sorted(字典.items()):
    nws = 工作簿.create_sheet(键)
    nws.append(['姓名','班级','分数'])
    for 数据 in 值:
        nws.append(数据)
工作簿.remove(工作簿['Sheet1'])
工作簿.save('c:/1.xlsx')
```

推荐使用Pandas

Pandas笔记

19.Excel文件的拆分与合并



每行的单元格数据放到一个列表里

3.7 合并与取消合并单元格

2020年6月11日 7:19

```
import openpyxl as vb
路径 = 'c:/测试3.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
工作表.merge_cells('B3:F5') # 合并
工作表.unmerge_cells('B3:F5') # 取消合并
工作簿.save(路径)
```

工作表.merge_cells(start_row=起始行号, start_column=起始列号,end_row=结束行号,end_column=结束列号)

3.8 使用公式及注意事项

2020年6月11日 7:25

```
import openpyxl as vb
路径 = 'c:/测试3.xlsx'
工作簿 = vb.load_workbook(路径)
工作表 = 工作簿['Sheet1']
工作表['F1'] = '=sum(C1:E1)'
工作簿.save(路径)
```

注意读取时会直接读取公式，所以要进行如下设置

```
import openpyxl as vb
路径 = 'c:/测试3.xlsx'
工作簿 = vb.load_workbook(路径, data_only=True)
工作表 = 工作簿['Sheet1']
工作簿.save(路径)
print(工作表['F1'].value)
```

3.9 对行和列进行分组

2020年6月11日 7:52

```
import openpyxl as vb  
路径 = 'c:/测试3.xlsx'  
工作簿 = vb.load_workbook(路径,data_only=True)  
工作表 = 工作簿['Sheet1']  
工作表.column_dimensions.group('A','D', hidden=True)  
工作簿.save(路径)
```

```
import openpyxl as vb  
路径 = 'c:/测试3.xlsx'  
工作簿 = vb.load_workbook(路径,data_only=True)  
工作表 = 工作簿['Sheet1']  
工作表.row_dimensions.group(1,5, hidden=True)  
工作簿.save(路径)
```

3.10 给单元格添加批注

2020年6月11日 8:31

```
import openpyxl as vb
路径 = 'c:/测试3.xlsx'
工作簿 = vb.load_workbook(路径,data_only=True)
工作表 = 工作簿['Sheet1']
批注 = vb.comments.Comment('这里写批注','孙兴华')
工作表['F20'].comment = 批注
工作表['F21'].comment = 批注
工作簿.save(路径)
```

04.装饰部分

2020年6月11日 7:23

4.1 字体 Font

2020年6月11日 2:07

Font(name='Calibri', size=11, bold=False, italic=False, vertAlign=None, underline='none', strike=False, color='FF000000')

参数解读:

name: 字体名称, 注意中文字体前面加u

size: 字号大小

bold: True (加粗) / False (不加粗)

italic: True (倾斜) / False (不倾斜)

vertAlign: 'None' (默认) / 'superscript' (上标) / 'subscript' (下标)

underline: 'None' (默认) / 'single' (单下划线) / 'double' (双下划线) / 'singleAccounting' (会计用单下划线) / 'doubleAccounting' (会计用双下划线)

strike: 'True' (显示删除线) / 'False' (不显示删除线)

color: 字体的颜色 [RGB转HEX](#)

```
import openpyxl as vb
```

```
路径 = r'c:/测试3.xlsx'
```

```
工作簿 = vb.load_workbook(路径)
```

```
工作表 = 工作簿['Sheet1']
```

```
字体对象 = vb.styles.Font(name=u'微软雅黑', bold=True, italic=True, size=72)
```

```
工作表['A1'].font = 字体对象
```

```
字体对象2 = vb.styles.Font(name=u'隶书', bold=False, italic=False, size=48)
```

```
工作表['A2'].font = 字体对象2
```

4.2 对齐 Alignment

2020年6月11日 3:16

Alignment(horizontal='general',vertical='bottom', text_rotation=0, wrap_text=False, shrink_to_fit=False, indent=0)

horizontal: 'general' (常规) / 'justify' (两端对齐) / 'right' (靠右) / 'centerContinuous' (跨列居中) / 'distributed' (分散对齐) / 'fill' (填充) / 'center' (居中) / 'left' (靠左)

vertical: 'center' (垂直居中) / 'top' (靠上) / 'bottom' (靠下) / 'justify' (两端对齐) / 'distributed' (分散对齐)

text_rotation: 指定文本旋转角度

wrap_text: 是否自动换行

shrink_to_fit: 是否缩小字体填充

indent: 指定缩进

4.3 边框 Side

2020年6月11日 3:33

Side(style=连线样式,color=边线颜色)

Border(left=左边线样式,right=右连线样式,top=上边线样式,bottom=下边线样式)

style参数的种类: 'double', 'mediumDashDotDot', 'slantDashDot', 'dashDotDot', 'dotted', 'hair', 'mediumDashed', 'dashed', 'dashDot', 'thin', 'mediumDashDot', 'medium', 'thick'

```
import openpyxl as vb
```

```
路径 = r'c:/测试3.xlsx'
```

```
工作簿 = vb.load_workbook(路径)
```

```
工作表 = 工作簿['Sheet1']
```

```
side = vb.styles.Side(style='thin',color='FF000000')
```

```
border = vb.styles.Border(left=side,right=side,top=side,bottom=side)
```

```
工作表['A1'].border = border
```

```
工作簿.save(路径)
```


4.4 填充 PatternFill

2020年6月11日 8:02

`PatternFill(fill_type=None, start_color='FFFFFFF', end_color='FF00000')`

fill_type: 'None' (不填充) / 'solid' (实心填充) / 'darkGray' (75%灰色) / 'mediumGray' (50%灰色) / 'lightGray' (25%灰色) / 'gray125' (12.5%灰色) / 'gray0625' (6.25%灰色) / 'darkHorizontal' (水平条纹) / 'darkVertical' (垂直条纹) / 'darkDown' (逆对角线条纹) / 'darkUp' (对角线条纹) / 'darkGrid' (对角线剖面线) / 'darkTrellis' (粗对角线剖面线) / 'lightHorizontal' (细水平条纹) / 'lightVertical' (细垂直条纹) / 'lightDown' (细逆对角线条纹) / 'lightUp' (细对角线条纹) / 'lightGrid' (细水平剖面线) / 'lightTrellis' (细对角线剖面线)

start_color / fgColor: 背景颜色 [RGB转HEX](#)

end_color / bgColor: 图案颜色 [RGB转HEX](#)

4.5 渐变填充 GradientFill

2020年6月11日 8:02

GradientFill (stop=(渐变颜色1, 渐变颜色2,))

fill_type: 'linear' (线性渐变) / 'path' (中心扩散)

degree: 旋转角度

stop: 一个元组 (OO, XX), OO 为起始颜色, XX 为结束颜色



建议使用元组保存

4.6 锁定单元格和隐藏公式

2020年6月11日 8:07

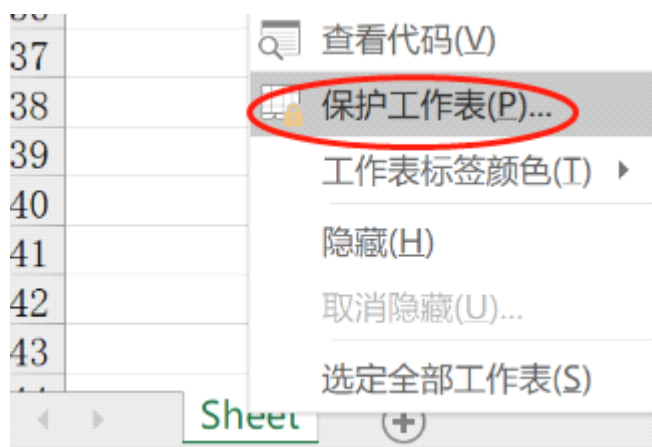
Protection(locked=True, hidden=False)

locked: 指定是否锁定单元格

hidden: 指定是否隐藏公式

只有在开启 “保护工作表” 之后，“锁定单元格” 和 “隐藏公式” 才生效。

在工作表处点击右键，即可开启 “保护工作表”：



4.7 行高和列宽

2020年6月11日 8:24

```
import openpyxl as vb
```

```
路径 = 'c:/测试3.xlsx'
```

```
工作簿 = vb.load_workbook(路径,data_only=True)
```

```
工作表 = 工作簿['Sheet1']
```

```
工作表.row_dimensions[1].height = 200
```

```
工作表.column_dimensions['B'].width = 100
```

```
工作簿.save(路径)
```

05.图表操作

2020年6月11日 8:46

5.1 插入图片

2020年6月11日 8:46

```
import openpyxl as vb
路径 = 'c:/测试3.xlsx'
工作簿 = vb.load_workbook(路径,data_only=True)
工作表 = 工作簿['Sheet1']
图片 = vb.drawing.image.Image('c:/童虎.png')
图片.height = 100
图片.width = 70
工作表.add_image(图片,"F19")
工作簿.save(路径)
```

5.2 柱状图

2020年6月11日 8:52

```
import openpyxl as vb
路径 = 'c:/柱状图.xlsx'
工作簿 = vb.load_workbook(路径,data_only=True)
工作表 = 工作簿['Sheet1']
# 新建一个柱状图
chart = vb.chart.BarChart()
# 设定数据引用范围
数据 = vb.chart.Reference(工作表,min_row=1,max_row=5,min_col=2,max_col=3)
# X轴项目名称
项目 = vb.chart.Reference(工作表,min_row=2,max_row=5,min_col=1)
# 给柱状图添加数据, 数据源中有标题, 因为数据中有标题行, 这里为True
chart.add_data(数据,titles_from_data=True)
# 设定X轴的项目
chart.set_categories(项目)
工作表.add_chart(chart,"F1")
工作簿.save(路径)
```

5.3 折线图

2020年6月11日 8:53

```
import openpyxl as vb
路径 = 'c:/折线图.xlsx'
工作簿 = vb.load_workbook(路径,data_only=True)
工作表 = 工作簿['Sheet1']
# 新建一个折线图
chart = vb.chart.LineChart()
# 设定数据引用范围
数据 = vb.chart.Reference(工作表,min_row=2,max_row=3,min_col=1,max_col=13)
# 分类的项目
项目 = vb.chart.Reference(工作表,min_row=1,min_col=2,max_col=13)
# 给柱状图添加数据, 数据源中有标题, from_rows=True因为数据是横项
chart.add_data(数据,from_rows=True,titles_from_data=True)
# 设定X轴的项目
chart.set_categories(项目)
工作表.add_chart(chart,"A8")
工作簿.save(路径)
```