Task 02 Python自动化之Excel

2.0 包的安装

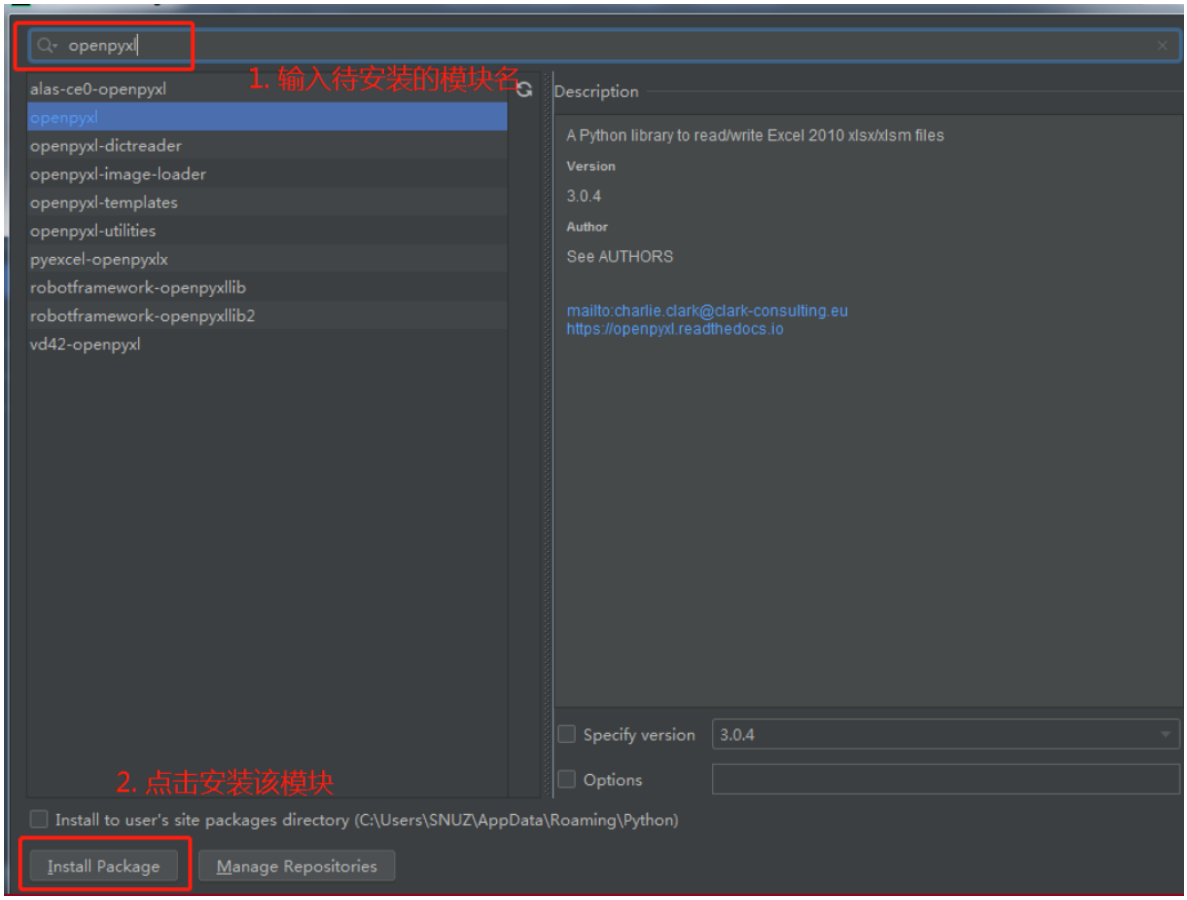
操作难度：☆

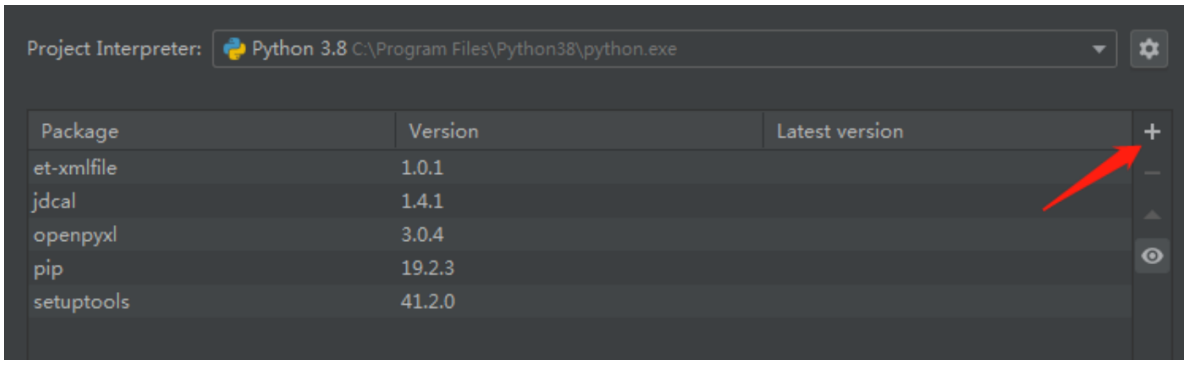
方法一：应用pip执行命令

安装**openpyxl**模块`pip install openpyxl`

注：openpyxl可以读取xlsx的格式，但是不可以去读xls格式；读取xls格式，可以安装**xlrd**模块，`pip install xlrd`，本章节以xlsx格式为主。

方法二：在Pycharm中：File->Setting->左侧Project Interpreter





2.1 Excel读取

项目难度：☆

- Excel全称为Microsoft Office Excel，2003年版本的是xls格式，2007和2007年之后的版本是xlsx格式。
- xlsx格式通过`openpyxl`模块打开； xls格式通过`xlwt`模块写，`xlrd`模块读取。
- 本文以xlsx模式为例

```
#多行内容显现
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

2.1.1 读取对应表格

```
#获取当前工作目录
import os
os.getcwd()

import warnings
warnings.filterwarnings('ignore')
```

关于路径：

文件应在当前工作目录才可引用，可导入`os`，使用函数`os.getcwd()`弄清楚当前工作目录是什么，可使用`os.chdir()`改变当前工作目录，具体可参考第一章节。（此处显现为相对路径）

1. 查看属性

```
#导入模块，查看属性
import openpyxl

wb = openpyxl.load_workbook('用户行为偏好.xlsx')
type(wb)
```

import * 和from...import...

`import *`和`from...import...`的区别

- `import`导入一个模块，相当于导入的是一个文件夹，相对路径。
- `from...import...`导入了一个模块中的一个函数，相当于文件夹中的文件，绝对路径。

2. 打开已经存在的Excel表格，查询对应sheet的名称

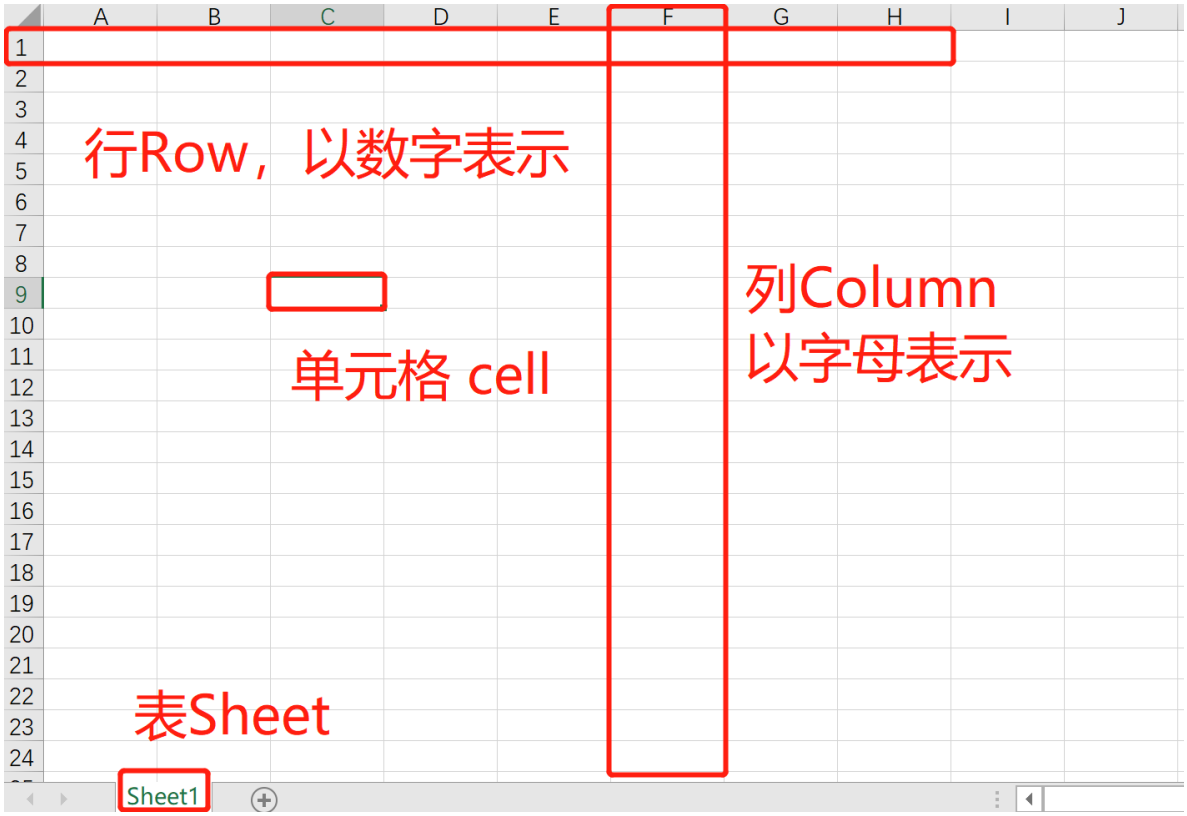
```
#导入模块中的函数，查询对应表的名称
from openpyxl import load_workbook

exl = load_workbook(filename = '用户行为偏好.xlsx')
print(exl.sheetnames)
```

```
'''通过传递表名字符串读取表、类型和名称 '''
sheet = exl.get_sheet_by_name('Sheet3')
sheet
type(sheet)
sheet.title
'''读取工作簿的活动表'''
#活动表是工作簿在Excel中打开时出现的工作表，再取得Worksheet对象后，可通过title属性取得它的名称。
anotherSheet = exl.active
anotherSheet
```

3. 获取Excel 内容占据的大小

```
sheet.dimensions
```

2.1.2 读取单元格



Cell

- Cell对象有一个value属性，包含这个单元格中保存的值。
- Cell对象也有row、column和coordinate属性，提供该单元格的位置信息。
- Excel用字母指定列，在Z列之后，列开始使用两个字母：AA、AB等，所以在调用的cell()方法时，可传入整数作为row和column关键字参数，也可以得到一个单元格。
- 注：第一行或第一列的整数取1，而不是0.

```
# 从表中取得单元格
## 获取表格名称
from openpyxl import load_workbook
exl = openpyxl.load_workbook('用户行为偏好.xlsx')
exl.get_sheet_names()

# 读取单元格
sheet = exl.get_sheet_by_name('用户次数偏好')
'''显现单元格格式'''
sheet['A1']
'''显现单元格文本内容'''
sheet['A1'].value
#另一种表达方式
a = sheet['A1']
a.value
'''行、列和数值显现'''
'Row' + str(a.row) + ', Column' + str(a.column) + ' is ' + a.value

'''显现单元格'''
'Cell ' + a.coordinate + ' is ' + a.value
```

```
# 顺B列打出前8行的奇数行单元格的值
for i in range(1,8,2):
    print(i,sheet.cell(row=i,column=2).value)
```

```
#确定表格的最大行数和最大列数，即表的大小
sheet.max_row
sheet.max_column
```

2.1.3 读取多个格子的值

```
#A1到C8区域的值
cells = sheet['A1:C8']
type(cells)
#用enumerate包装一个可迭代对象，同时使用索引和迭代项
for index, item in enumerate(sheet['A1:C8']):
    if index >= 1:
        print("\n")
    for cell in item:
        print(cell.value,end=" ")
```

```
# 指定范围的值
# 行获取
for row in sheet.iter_rows(min_row = 1, max_row = 5,
                           min_col = 2, max_col = 6):
    print(row)
    # 一列由多个单元格组成，若需要获取每个单元格的值则循环获取即可
    for cell in row:
        print(cell.value)

# 列获取
for col in sheet.iter_cols(min_row = 1, max_row = 5,
                           min_col = 2, max_col = 6):
    print(col)

    for cell in col:
        print(cell.value)
```

2.1.4 练习题

找出用户行为偏好.xlsx中sheet1表中空着的格子，并输出这些格子的坐标

```
from openpyxl import load_workbook

exl = load_workbook('用户行为偏好.xlsx')
sheet = exl.active

for row in sheet.iter_rows(min_row = 1, max_row = 29972,
                           min_col = 1, max_col = 10):
                           #具体查看对应表格的行列数
    for cell in row:
        if not cell.value:
            print(cell.coordinate)
```

2.2 Excel写入

项目难度：☆

2.2.1 写入数据并保存

1. 原有工作簿中写入数据并保存

```
# 已有的表格赋值保存
from openpyxl import load_workbook

exl = load_workbook(filename = '用户行为偏好.xlsx')
sheet = exl.active
sheet['A1'] = 'hello world'
#或者cell = sheet['A1']
#cell.value = 'hello world'
exl.save(filename = '用户行为偏好.xlsx') #存入原Excel表中，若创建新文件则可命名为不同名称
```

2. 创建新的表格写入数据并保存

```
# openpyxl 写入xsxl
from openpyxl import load_workbook
wb = openpyxl.Workbook()
# 创建一个sheet
sh = wb.active
sh.title = 'My Worksheet' #注：此处在工作簿内的表格名称没有变。

# 写入excel
# 参数对应 行, 列, 值
sh.cell(1,1).value = 'this is test'

# 保存
wb.save('new_test.xlsx')
```

2.2.2 将公式写入单元格保存

```
# 公式写入单元格保存
from openpyxl import load_workbook

exl = load_workbook(filename = '用户行为偏好.xlsx')
sheet = exl.get_sheet_by_name('Sheet3')
sheet.dimensions        #先查看原有表格的单元格范围，防止替代原有数据

sheet['A30'] = '=SUM(A1:D1)'
exl.save(filename='用户行为偏好.xlsx')
```

2.2.3 插入数据

```
#插入列数据
'''idx=2第2列，第2列前插入一列'''
sheet.insert_cols(idx=2)
'''第2列前插入5列作为举例'''
sheet.insert_cols(idx=2, amount=5)

#插入行数据
'''插入一行'''
sheet.insert_rows(idx=2)
'''插入多行'''
sheet.insert_rows(idx=2, amount=5)

exl.save(filename='用户行为偏好.xlsx')
```

2.2.4 删除

```
# 删除多列
sheet.delete_cols(idx=5, amount=2)
# 删除多行
sheet.delete_rows(idx=2, amount=5)

exl.save(filename='用户行为偏好.xlsx')
```

2.2.5 移动

当数字为正即向下或向右，为负即为向上或向左

```
#移动
'''当数字为正即向下或向右，为负即为向上或向左'''
sheet.move_range('B3:E16',rows=1,cols=-1)
```

2.2.6 Sheet表操作

1. 创建新的sheet

```
from openpyxl import Workbook

workbook=Workbook()
sheet=workbook.active
workbook.save(filename='new_test.xlsx')

exl.create_sheet('new_sheet')
```

2. 修改sheet表名

```
sheet = exl.active
sheet.title = 'newname'
```

2.3 Excel 样式

项目难度：☆☆

2.3.1设置字体样式

1. 设置字体样式

`Font(name字体名称,size大小,bold粗体,italic斜体,color颜色)`

```
from openpyxl import Workbook
from openpyxl.styles import Font

workbook = Workbook()
sheet = workbook.active
cell = sheet['A1']
font = Font(name='字体', size=10, bold=True, italic=True, color='FF0000')
cell.font = font
workbook.save(filename='new_test')
```

2. 设置多个格子的字体样式

```
from openpyxl import Workbook
from openpyxl.styles import Font

workbook = Workbook()
sheet = workbook.active
cells = sheet[2]
font = Font(name='字体', size=10, bold=True, italic=True, color='FF000000')
for cell in cells:
    cell.font = font
workbook.save(filename='new_test')
```

2.3.2 设置对齐样式

水平对齐：`distributed, justify, center, left, fill, centerContinuous, right, general`

垂直对齐：`bottom, distributed, justify, center, top`

1. 设置单元格边框样式

`Side`：变现样式，边线颜色等

`Border`：左右上下边线

```
from openpyxl import Workbook
from openpyxl.styles import Font

workbook = Workbook()
sheet = workbook.active
cell = sheet['A1']
side = Side(border_style='thin', color='FF000000')
#先定好side的格式
border = Border(left=side, right=side, top=side, bottom=side)
#代入边线中
cell.border = border
workbook.save(filename='new_test')
```

2. 设置单元格边框样式

变现样式：`double, mediumDashDotDot, slantDashDot, dashDotDot, dotted, hair, mediumDashed, dashed, dashDot, thin, mediumDashDot, medium, thick`

```
from openpyxl import Workbook
from openpyxl.styles import PatternFill, Border, Side, Alignment, Font,
GradientFill

workbook = Workbook()
sheet = workbook.active
cell = sheet['A1']
pattern_fill = PatternFill(fill_type='solid',fgColor="DDDDDD")
cell.fill = pattern_fill
#单色填充
cell2 = sheet['A3']
gradient_fill = GradientFill(stop=('FFFFFF', '99ccff','000000'))
cell2.fill = gradient_fill
#渐变填充
workbook.save(filename='new_test')
```

2.3.3 设置行高与列宽

```
from openpyxl import Workbook

workbook = Workbook()
sheet = workbook.active
sheet.row_dimensions[1].height = 50
sheet.column_dimensions['C'].width = 20
workbook.save(filename='new_test')
```

2.3.4 合并、取消合并单元格

```
sheet.merge_cells('A1:B2')
sheet.merge_cells(start_row=1, start_column=3,
                  end_row=2, end_column=4)

sheet.unmerge_cells('A1:B2')
sheet.unmerge_cells(start_row=1, start_column=3,
                    end_row=2, end_column=4)
```

2.3.5 练习题

打开test文件，找出文件中购买数量`buy_mount`超过5的行，并对其标红、加粗、附上边框。

```
from openpyxl import load_workbook
from openpyxl.styles import Font, Side, Border

workbook = load_workbook('./test.xlsx')
sheet = workbook.active
buy_mount = sheet['F']
row_lst = []

for cell in buy_mount:
    if isinstance(cell.value, int) and cell.value > 5:
        print(cell.row)
```

```
        row_lst.append(cell.row)

side = Side(style='thin', color='FF000000')
border = Border(left=side, right=side, top=side, bottom=side)
font = Font(bold=True, color='FF0000')
for row in row_lst:
    for cell in sheet[row]:
        cell.font = font
        cell.border = border
workbook.save('new_test.xlsx')
```

2.4 后记

- Python与Excel的自动化内容较多，此篇重在介绍基础，起到抛砖引玉的学习效果。
- Excel文档参考 用户行为偏好.xlsx