## python实现操作excel

### 一、对比

要使用[Python](http://www.raincent.com/list-10-1.html)直接生成Excel文件，还要对已有的Excel文件进行读取。在这个过程中，研究并尝试了一些工具，也走了一些弯路。记录下来，下次再有类似需求的时候就不用搜索了。



#### 1.1 xlrd&xlwt

xlrd&xlwt主要是针对Office 2013或更早版本的XLS文件格式。

优点：

**（1）支持XLS格式**

XlsxWriter和OpenPyXL都不支持XLS格式，从这个角度看，xlrd&xlwt仍然有一定的不可替代性。

缺点：

**（1）对XLSX支持比较差**

目前xlrd已经可以读取XLSX文件了，有限地支持。至于xlwt我没有试验过，估计是够呛。

**（2）对修改的支持比较差**

xlrd和xlwt是两个相对独立的模块，虽然xlutils提供方法帮助你把xlrd.Book对象复制到xlwt.Workbook对象，但跟XlsxWriter类似，后者只是提供write方法，使得你无法很容易地获取当前已经写入的数据并进行有针对性的修改。如果非要这样做，你要不断地保存，然后再用新的xlrd.Book对象读取你要的信息，还是比较麻烦的。

**（3）功能很弱**

除了最基本的写入数据和公式，xlwt所提供的功能非常少(Excel 2013本身支持的功能也就很少)。对于读取也是一样的，很多信息在读入时就丢失掉了。

#### 1.2 OpenPyXL

OpenPyXL是比较综合的一个工具，能读能写能修改，功能还算可以但也有很大的缺陷。我在中间版本的时候是打算完全依赖它的，但后来发现一个严重的问题就放弃了。

优点：

**(1)能读能写能修改**

OpenPyXL的工作模式跟XlsxWriter和xlwt有很大的区别，它用的是getter/setter模式。你可以随时读取某个单元格的内容，并根据其内容进行相应的修改，OpenPyXL会帮你记住每个单元格的状态。

特别需要注意的一点：虽然它支持修改已有文件，但由于其所支持的功能有限，读入文件时会忽略掉它所不支持的内容，再写入时，这些内容就丢失了。因此使用时一定要慎重。比如下面的缺点中提到它无法读入公式，那如果你修改一个带有公式的文件，保存之后，所有的公式就都没有了。

**(2)功能还算可以**

整体来讲，它所支持的功能介于XlsxWriter和xlwt之间。

**缺点**

**(1)不支持XLS**

这件事情只能让xlrd和xlwt去做。

**(2)不支持读取公式**

这其实是个不太简单的事情，虽然我没尝试过，但相信xlrd也做不好这件事。

Excel的单元格如果是一个公式，它内部会同时保存公式本身和运算结果的缓存。用OpenPyXL读取单元格内容，它不会告诉你这个单元格的公式是什么，甚至不会告诉你这个单元格存的是公式，它只会拿到这个缓存的运算结果。我本来想利用它判别单元格是不是用了公式，然后做出不同的处理。结果遇到了这个问题，最后只好采取了其他变通的方式去做。

二、xlrd、xlwt操作excel表格详解

#### 2.1基本操作（读取、写入）

import os, time

import xlrd, xlwt

import sys

import chardet

import mysql.connector

reload(sys)

sys.setdefaultencoding('utf-8')

excel\_path = r'D:\desktop\test.xlsx'

# 打开excel文件

wb = xlrd.open\_workbook(excel\_path)

# 获取工作簿的所有sheet名

str\_sheets = wb.sheet\_names()

print('str\_sheets值：' + str(str\_sheets))

# 根据sheet索引或者名称获取sheet内容(两种方法)

ws01 = wb.sheet\_by\_index(0) # sheet索引从0开始

ws02 = wb.sheet\_by\_name('Sheet1')

# 获取表的名称、行列值

str\_sheet\_name = ws02.name

rows = ws02.nrows

cols = ws02.ncols

print('sheet表名称:' + str\_sheet\_name + ' 行数：' + str(rows) + ' 列数：'+ str(cols))

# 获取表的行列的内容,索引从0开始

first\_row = ws02.row\_values(0)

second\_col = ws02.col\_values(1)

print('第一行内容：' + str(first\_row) + '\n第二列内容: ' + str(second\_col))

# 获取单元格内容（三种方法）

str1 = ws02.cell(3, 1).value.encode('utf-8')

str2 = ws02.cell\_value(3, 1).encode('utf-8')

str3 = ws02.row(3)[1].value.encode('utf-8')

print('str1值：{} str2值：{} str3值：{}'.format(str1, str2, str3))

# 设置单元格样式

def set\_style(bold=False):

style = xlwt.XFStyle() # 初始化样式

font = xlwt.Font() # 为样式创建字体

font.name = u'黑体'

font.bold = bold

font.color\_index = 4

font.height = 220

style.font = font

return style

# 写入excel

def write\_excel():

f = xlwt.Workbook() # 创建工作簿

# 创建第一个sheet:

sheet2 = f.add\_sheet(u'sheet2\_write', cell\_overwrite\_ok=True) # 创建sheet

row0 = [u'业务', u'状态', u'北京', u'上海', u'广州', u'深圳', u'状态小计', u'合计']

# 生成第一行

for i in range(0, len(row0)):

sheet2.write(0, i, row0[i], set\_style())

f.save(r'd:\desktop\test1.xls') # 保存文件

write\_excel()

### 三、**OpenPyXL**操作excel表格详解

#### 3.1基本操作

import openpyxl, sys

import datetime, time

reload(sys)

sys.setdefaultencoding('utf-8')

excel\_path = r'D:\desktop\test.xlsx'

wb = openpyxl.load\_workbook(excel\_path)

# ws00 = wb.active

ws01 = wb['Sheet1']

# 索引从0开始

x = ws01.cell(4, 2).value

print(x)

ws02 = wb.create\_sheet('sheet3\_openpyxl')

ws02['A1'] = 42 # 写入数字

ws02['B1'] = u"你好" + "dxj728" # 写入文字（UTF-8格式）

ws02['A2'] = datetime.datetime.now() # 写入当前时间

ws02.cell(3, 3).hyperlink = 'https://www.dxj728.com' # 插入超链接（明文）

ws02.cell(4, 4, value=('=HYPERLINK("{}","{}")'.format('https://www.dxj728.com', 'dxj728'))) # 插入超链接字符

wb.save(excel\_path)