Python中xlrd和xlwt模块使用方法

本文主要介绍可操作excel文件的xlrd、xlwt模块。其中xlrd模块实现对excel文件内容读取，xlwt模块实现对excel文件的写入。

安装xlrd和xlwt模块

xlrd和xlwt模块不是自带的，需要自行安装。模块安装建议使用pip自动安装。安装方法参考<Python自动安装第三方模块>

xlrd模块使用

测试使用excel文档名称为Student.xlsx，内容如下：

	A	B	C	D	E
1	姓名	年龄	出生日期	爱好	关系
2	xiaoming2	17	2015/5/5	basketball	friend
3	zhangsan2	18	2014/6/6	football	
4	lisi2	19	2013/7/7	basketball	friend2
5	wangwu2	20	2012/2/2	read book	
6	zhaoliu2	None			
7					
8					
9					
10					

Sheet1 Sheet2 Sheet3

(1) 打开excel文件并获取所有sheet

```
>>> import xlrd
>>> workbook = xlrd.open_workbook(r'D:\Program Files\Notepad++\Student.xlsx')
>>> print workbook.sheet_names()
[u'Sheet1', u'Sheet2', u'Sheet3']
```

(2) 根据下标获取sheet名称

```
>>> sheet2_name=workbook.sheet_names()[1]
>>> print sheet2_name
Sheet2
```

(3) 根据sheet索引或者名称获取sheet内容，同时获取sheet名称、行数、列数

```
>>> sheet2 = workbook.sheet_by_index(1)
>>> print sheet2.name,sheet2.nrows,sheet2.ncols
Sheet2 6 5
>>> sheet2 = workbook.sheet_by_name('Sheet2')
>>> print sheet2.name,sheet2.nrows,sheet2.ncols
Sheet2 6 5
```

(4) 根据sheet名称获取整行和整列的值

```
>>> sheet2 = workbook.sheet_by_name('Sheet2')
>>> rows = sheet2.row_values(3)
>>> cols = sheet2.col_values(2)
>>> print rows
[u'lisi2', 19.0, 41462.0, u'basketball', u'friend2'] #标红部分为日期2013/7/7，实际却显示为浮点数。后面有描述如何纠正
>>> print cols
[u'\u51fa\u751f\u65e5\u671f', 42129.0, 41796.0, 41462.0, 40941.0, u''] # 问题同上
>>>
```

(5）获取指定单元格的内容

```
>>> print sheet2.cell(1,0).value.encode('utf-8')
xiaoming2
>>> print sheet2.cell_value(1,0).encode('utf-8')
xiaoming2
>>> print sheet2.row(1)[0].value.encode('utf-8')
xiaoming2
```

(6）获取单元格内容的数据类型

```
>>> print sheet2.cell(1,0).ctype #第2行第1列:xiaoming2 为string类型
1
>>> print sheet2.cell(1,1).ctype #第2行第2列:12  为number类型
2
>>> print sheet2.cell(1,2).ctype #第2行第3列:2015/5/5 为date类型
3
```

说明：ctype：0 empty,1 string, 2 number, **3 date**, 4 boolean, 5 error

(7）获取单元内容为日期类型的方式

使用xlrd的xldate_as_tuple处理为date格式，先判断表格的ctype=3时xlrd才能执行操作，如下：

```
>>> from datetime import datetime,date
>>> sheet2.cell(1,2).ctype
3
>>> sheet2.cell(1,2).value
42129.0
>>> xlrd.xldate_as_tuple(sheet2.cell_value(1,2),workbook.datemode)
(2015, 5, 5, 0, 0, 0)
>>> date_value = xlrd.xldate_as_tuple(sheet2.cell_value(1,2),workbook.datemode)
>>> date(*date_value[:3])
datetime.date(2015, 5, 5)
```

```
>>> date(*date_value[:3]).strftime('%Y/%m/%d')
'2015/05/05'
```

那么如果是在脚本中需要获取并显示单元格内容为日期类型的，可以先做一个判断。判断ctype是否等于3，如果等于3，则用时间格式处理：

```
if (sheet.cell(row,col).ctype == 3):
  date_value = xlrd.xldate_as_tuple(sheet.cell_value(row,col),book.datemode)
  date_tmp = date(*date_value[:3]).strftime('%Y/%m/%d')
```

(8) 获取合并单元格的内容

```
>>> sheet2.cell(1,4).value #第4列的第2行和第3行是合并单元格
u'friend'
>>> sheet2.cell(2,4).value
u''
>>> sheet2.cell(5,1).value #第6行的第2和第3第4列是合并单元格，这里我们只获取到第6行第2列的值而第3列第4列获取的内容为空，如何处理?
u'None'
>>> sheet2.cell(5,2).value
u''
>>> sheet2.cell(5,3).value
u''
```

从实验结果可以看出来，第6行的第2和第3第4列是合并单元格，但这里我们只获取到第6行第2列的值而第3列第4列获取的内容为空，理论上来说合并的

单元格内容应该是一样的，但是现在只有合并的第一个单元格可以获取到值，其他为空，如何处理? 再用一种更直观的方式显示

```
>>> sheet2.row_values(5)
[u'zhaoliu2', u'None', u'', u'', u''] #标红的部分为合并单元格
>>> sheet2.col_values(4)
[u'\u5173\u7cfb', u'friend', u'', u'friend2', u'', u''] #标红的部分为合并单元格，注意这里是两个合并单元格
```

可以利用merged_cells方法进行处理，处理的方法是只能获取合并单元格的第一个cell的行列索引，才能读到值，读错了就是空值。即合并行单元格读取

行的第一个索引，合并列单元格读取列的第一个索引。这里，需要在读取文件的时候添加个参数，将formatting_info参数设置为True，默认是False，否

则可能调用merged_cells方法获取到的是空值。

```
>>> workbook = xlrd.open_workbook(r'D:\Program Files\Notepad++\Student.xlsx',formatting_info=True)

>>> sheet2 = workbook.sheet_by_name('sheet2')
```

```
>>> sheet2.merged_cells
[(1, 3, 4, 5), (3, 5, 4, 5), (5, 6, 1, 5)]
```

merged_cells返回的这四个参数的含义是：(row,row_range,col,col_range),其中[row,row_range)包括row,不包括row_range,col也是一样，下标从0开始。即(1, 3, 4, 5)的含义是：第2到3行（不包括第4行）合并，(5, 6, 1, 5)的含义是：第2到5列合并。利用这个，可以分别获取合并的三个单元格的内容：

```
>>> print sheet2.cell_value(1,4) #(1, 3, 4, 5)
friend
>>> print sheet2.cell_value(3,4) #(3, 5, 4, 5)
friend2
>>> print sheet2.cell_value(5,1) #(5, 6, 1, 5)
None
```

发现规律了没？是的，**获取merge_cells返回的row和col低位的索引即可**！ 于是可以这样一劳永逸：

```
>>> merge = []
>>> for (rlow,rhigh,clow,chigh) in sheet2.merged_cells:
...     merge.append([rlow,clow])
...
>>> merge
[[1, 4], [3, 4], [5, 1]]
>>> for index in merge:
...     print sheet2.cell_value(index[0],index[1])
...
friend
friend2
None
```

xlwt模块使用

假设要实现如下内容写入excel，如何实现

	A	B	C	D	E	F	G	H
1	业务	状态	北京	上海	广州	深圳	状态小计	合计
2		预订						
3		出票						
4		退票						
5	机票	业务小计						
6		预订						
7		出票						
8		退票						
9	船票	业务小计						
10		预订						
11		出票						
12		退票						
13	火车票	业务小计						
14		预订						
15		出票						
16		退票						
17	汽车票	业务小计						
18		预订						
19		出票						
20		退票						
21	其它	业务小计						
22	合计							

代码如下：

```
'''
设置单元格样式
'''

def set_style(name,height,bold=False):
  style = xlwt.XFStyle() # 初始化样式

  font = xlwt.Font() # 为样式创建字体
  font.name = name # 'Times New Roman'
  font.bold = bold
  font.color_index = 4
  font.height = height

  # borders= xlwt.Borders()
  # borders.left= 6
  # borders.right= 6
  # borders.top= 6
  # borders.bottom= 6

  style.font = font
  # style.borders = borders

  return style
```

```
#写excel
def write_excel():
  f = xlwt.Workbook() #创建工作簿

  '''
  创建第一个sheet:
    sheet1
  '''
  sheet1 = f.add_sheet(u'sheet1',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']
  column0 = [u'机票',u'船票',u'火车票',u'汽车票',u'其它']
  status = [u'预订',u'出票',u'退票',u'业务小计']

  #生成第一行
  for i in range(0,len(row0)):
    sheet1.write(0,i,row0[i],set_style('Times New Roman',220,True))

  #生成第一列和最后一列(合并4行)
  i, j = 1, 0
  while i < 4*len(column0) and j < len(column0):
    sheet1.write_merge(i,i+3,0,0,column0[j],set_style('Arial',220,True)) #第一列
    sheet1.write_merge(i,i+3,7,7) #最后一列"合计"
    i += 4
    j += 1

  sheet1.write_merge(21,21,0,1,u'合计',set_style('Times New Roman',220,True))

  #生成第二列
  i = 0
  while i < 4*len(column0):
    for j in range(0,len(status)):
      sheet1.write(j+i+1,1,status[j])
    i += 4

  f.save('demo1.xlsx') #保存文件

if __name__ == '__main__':
  #generate_workbook()
  #read_excel()
  write_excel()
```

需要稍作解释的就是write_merge方法：

```
write_merge(x, x + m, y, w + n, string, sytle)
```

x表示行，y表示列，m表示跨行个数，n表示跨列个数，string表示要写入的单元格内容，style表示单元格样式。其中，x，y，w，h，都是以0开始计算
的。

这个和xlrd中的读合并单元格的不太一样。

如上述：sheet1.write_merge(21,21,0,1,u'合计',set_style('Times New Roman',220,True))

即在22行合并第1,2列，合并后的单元格内容为"合计"，并设置了style。

如果需要创建多个sheet，则只要f.add_sheet即可。

如在上述write_excel函数里f.save('demo1.xlsx') 这句之前再创建一个sheet2，效果如下：

	A	B	C	D	E
	姓名	年龄	出生日期	爱好	关系
2	小杰		1991/11/11		
3	小胖				好朋友
4	小明				
5	大神				
6	大仙				
7	小敏				
8	无名		暂无		
9					
10					

sheet1 sheet2

代码也是真真的easy的了：

```
'''
  创建第二个sheet:
    sheet2
  '''
  sheet2 = f.add_sheet(u'sheet2',cell_overwrite_ok=True) #创建sheet2
  row0 = [u'姓名',u'年龄',u'出生日期',u'爱好',u'关系']
  column0 = [u'小杰',u'小胖',u'小明',u'大神',u'大仙',u'小敏',u'无名']

  #生成第一行
  for i in range(0,len(row0)):
    sheet2.write(0,i,row0[i],set_style('Times New Roman',220,True))

  #生成第一列
  for i in range(0,len(column0)):
    sheet2.write(i+1,0,column0[i],set_style('Times New Roman',220))

  sheet2.write(1,2,'1991/11/11')
  sheet2.write_merge(7,7,2,4,u'暂无') #合并列单元格
  sheet2.write_merge(1,2,4,4,u'好朋友') #合并行单元格

  f.save('demo1.xlsx') #保存文件
```