第二十四章 Python 操作 Excel 存储数据

24.1 openpyxl 模块

openpyxl 模块是一个读写 Excel 2010 文档的 Python 库，如果要处理更早格式的 Excel 文档，需要用到额外的库，openpyxl 是一个比较综合的工具，能够同时读取和修改 Excel 文档。

24.1.1 安装 openpyxl 模块

要想使用 openpyxl 模块，必须先安装此模块。直接使用 pip 就可以进行安装，命令如下：

```
pip install openpyxl
```

24.1.2 Excel 文件的三个对象

想要操作 Excel 首先要了解 Excel 基本概念，Excel 中列以字母命名，行以数字命名，比如左上角第一个单元格的坐标为 A1，下面的为 A2，右边的 B1。

openpyxl 中有三个不同层次的类，Workbook 是对工作簿的抽象，Worksheet 是对表格的抽象，Cell 是对单元格的抽象，每一个类都包含了许多属性和方法。

打开或者创建一个 Excel 需要创建一个 Workbook 对象。获取一个表则需要先创建一个 Workbook 对象，然后使用该对象的方法来得到一个 Worksheet 对象。如果要获取表中的数据，那么得到 Worksheet 对象以后再从中获取代表单元格的 Cell 对象。

1) Workbook 对象

一个 Workbook 对象代表一个 Excel 文档，因此在操作 Excel 之前，都应该先创建一个 Workbook 对象。对于创建一个新的 Excel 文档，直接进行 Workbook 类的调用即可，对于一个已经存在的 Excel 文档，可以使用 openpyxl 模块的 load_workbook 函数进行读取，该函数包涵多个参数，但只有 filename 参数为必传参数。filename 是一个文件名，也可以是一个打开的文件对象。

【示例 24-1】创建 Workbook 对象

```
import openpyxl
excel = openpyxl.Workbook()  # 创建本地工作簿
excel = openpyxl.load_workbook("abc.xlsx")  # 加载本地已存在的工作簿
# 操作工作簿完毕后需要保存工作簿
excel.save("workbook_test.xlsx")
```

Workbook 对象提供了很多属性和方法，其中，大部分方法都与 sheet 有关，常用属性和方法如表 24-1 所示：

表 24-1 Workbook 对象常用属性和方法

名称	说明
excel.active	获取当前活跃的 Worksheet 对象
excel.worksheets	以列表的形式返回所有的 Worksheet 对象
excel.sheetnames	获取工作簿中的表名[name1, name2, name3]

尚学堂·百战程序员

excel.encoding	获取文档的字符集编码
excel.properties	获取文档的元数据，如标题，创建者，创建日期等
excel_sheet = excel["Sheet1"]	通过工作表名获取到 worksheet 对象
excel.remove(excel_sheet)	删除一个表格，参数为 worksheet 对象
excel.create_sheet()	创建一个空的表格，参数为表名和 index

2) Worksheet 对象

通过 Worksheet 对象获取表格的属性，得到单元格中的数据，修改表格中的内容。openpyxl 提供了非常灵活的方式来访问表格中的单元格和数据，常用的 Worksheet 属性和方法如表 24-2 和表 24-3 所示：

表 24-2 Worksheet 对象常用属性

属性名	说明
excel_sheet.title	表格的标题(可读可写)
excel_sheet.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
excel_sheet.max_row	表格的最大行数
excel_sheet.min_row	表格的最小行数
excel_sheet.max_column	表格的最大列数
excel_sheet.min_column	表格的最小列数
excel_sheet.rows	按行获取单元格(Cell 对象)
excel_sheet.columns	按列获取单元格(Cell 对象)
excel_sheet.values	按行获取表格的内容(数据)

表 24-3 Worksheet 对象常用方法

属性名	说明
excel_sheet.iter_rows()	按行获取所有单元格，内置属性有(min_row,max_row,min_col,max_col)
excel_sheet.iter_cols()	按列获取所有的单元格
excel_sheet["A"]	返回 A 列中所有的单元格 cell 对象
excel_sheet["1"]	返回第一行中所有的单元格 cell 对象
excel_sheet["A1"]	返回该单元格对象
excel.cell()	设置单元格的值
excel.append(iterable)	逐行写

3) Cell

Cell 对象比较简单，常用的属性如表 24-4 所示：

表 24-4 Cell 对象常用属性

属性名	说明
row	单元格所在的行
column	单元格所在的列
value	单元格的值

尚学堂·百战程序员

coordinate	单元格的坐标

24.2 Python 操作 Excel

24.2.1 Python 操作 Excel 之读取

【示例 24-2】打开本地工作簿，获取所有工作表名称

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
#获取工作表的名称
print(wb.sheetnames)
#遍历获取工作表的名称
for sheet in wb:
    print(sheet.title)
```

执行结果如图 24-1 所示：

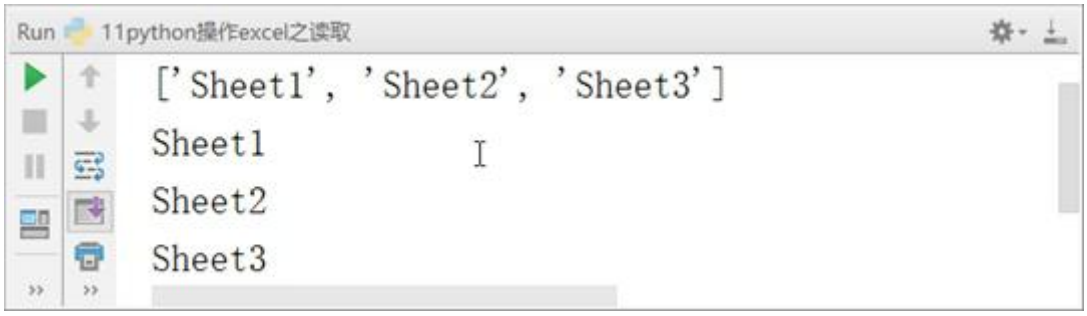


图 24-1 示例 24-2 运行效果图

【示例 24-3】创建工作表

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
#创建工作表
mySheet = wb.create_sheet('mySheet')
print(wb.sheetnames)
#遍历获取工作表的名称
for sheet in wb:
    print(sheet.title)
```

执行结果如图 24-2 所示：

尚学堂·百战程序员

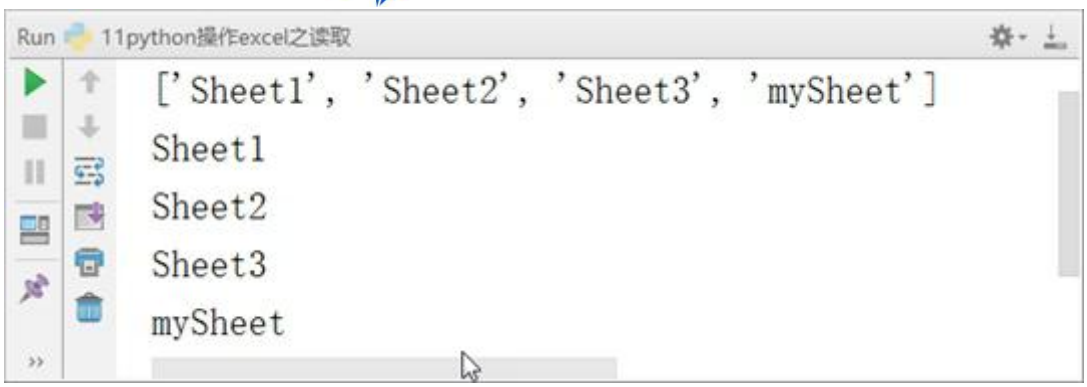


图 24-2 示例 24-3 运行效果图

【示例 24-4】根据工作表名称获取工作表

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
#根据名称获取表单
sheet3 = wb.get_sheet_by_name('Sheet2')
#或者
sheet4 = wb['mySheet']
```

【示例 24-5】获取单元格对象及单元格的值

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
#获取当前激活的工作表
ws = wb.active    #返回的是工作表对象
print(ws)    #<Worksheet "Sheet1">
#获取 Cell 对象及单元格的值
print(ws['A1'])    #<Cell 'Sheet1'.A1>
print(ws['A1'].value)
```

【示例 24-6】获取单元格的行、列及值

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
ws = wb.active
c = ws['B1']
print('Row:{} Column:{} is {}'.format(c.row,c.column,c.value))
print('cell:{}    is {}'.format(c.coordinate,c.value))
print('ws.cell(row=1,column=2).value：',ws.cell(row=1,column=2).value)
#for 循环获取
```

尚学堂·百战程序员

```
print('循环遍历获取'.center(20,'*'))
for r in ws.rows:
    for c in r:
        print(c.value,end='\t')
    print()
```

执行结果如图 24-3 所示：

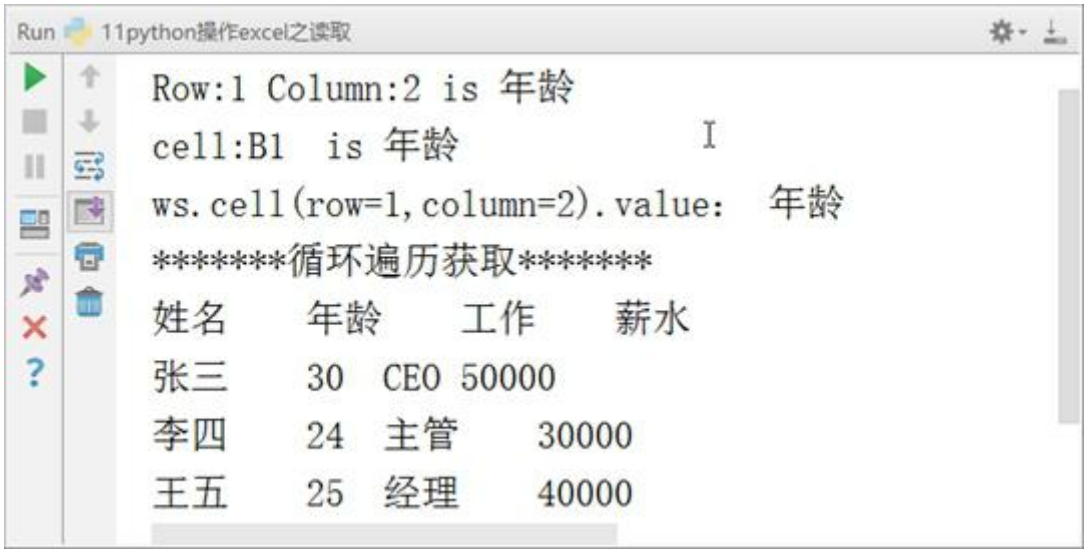


图 24-3 示例 24-6 运行效果图

【示例 24-7】读取整行、整列及部分几行

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
ws = wb.active
print('获取整列'.center(20,'*'))
colc = ws['C']  #获取到 C 列
print(colc)
print('遍历获取整列的数据'.center(20,'*'))
for c in colc:
    print(c.value,end='\t')
print()
print('获取整行数据'.center(20,'*'))
row6 = ws[6] #获取第 6 行
print(row6)
print('切片获取部分几行'.center(20,'*'))
row_range = ws[2:4]
print(row_range)
print('循环遍历部分几行数据'.center(20,'*'))
for row in row_range:
    for c in row:
```

尚学堂·百战程序员

```
            print(c.value,end='\t')
        print()
```

执行结果如图 24-4 所示：

图 24-4 示例 24-7 运行效果图

【示例 24-8】使用 iter_rows()逐行读取

```
    import openpyxl
    wb = openpyxl.load_workbook('excelTest.xlsx')
    ws = wb.active
    print('共{}行， 共{}列'.format(ws.max_row,ws.max_column))
    print('ws.iter_rows()设置最小列、最大列、最小行和最大行读取'.center(20,'*'))
    for row in ws.iter_rows(min_row=1,max_row=3,max_col=3,min_col=2):
        for cell in row:
            print(cell.value,end=',')
        print()
```

执行结果如图 24-5 所示：

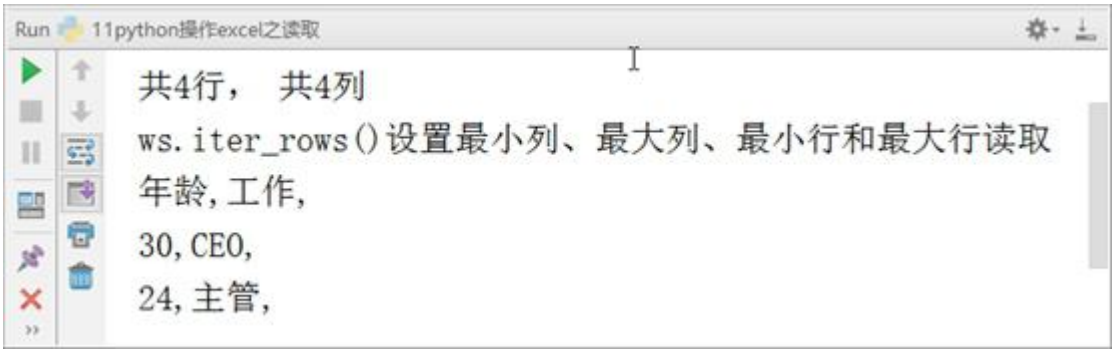


图 24-5 示例 24-8 运行效果图

尚学堂·百战程序员

【示例 24-9】部分行部分列切片读取

```
import openpyxl
wb = openpyxl.load_workbook('excelTest.xlsx')
ws = wb.active
print('共{}行，  共{}列'.format(ws.max_row,ws.max_column))
print('部分行部分列切片读取'.center(20,'*'))
cell_range = ws['A1:C3']
for rowObject in cell_range:
    for cellObject in rowObject:
        print(cellObject.coordinate,cellObject.value,end='\t')
    print()
```

执行结果如图 24-6 所示：

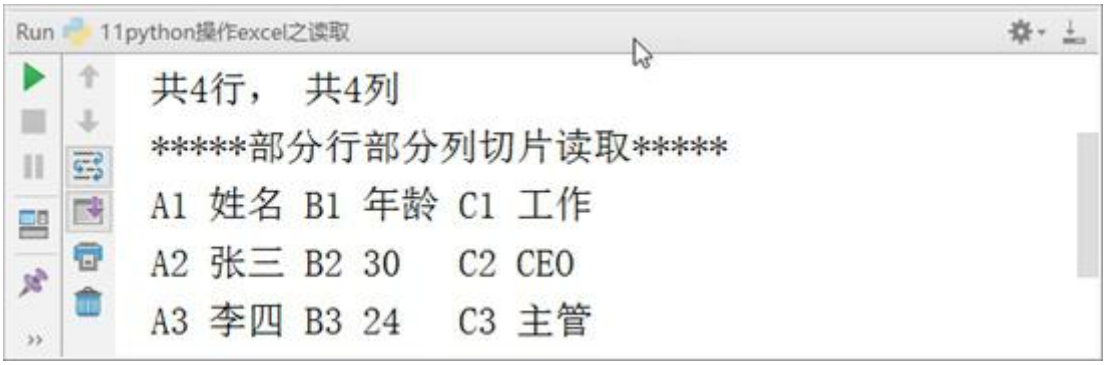


图 24-6 示例 24-9 运行效果图

【示例 24-10】列数字与字母的对应转换

```
from openpyxl.utils import get_column_letter,column_index_from_string
print('第 2 列对应的字母：',get_column_letter(2))
print('第 34 列对应的字母：',get_column_letter(34))
print('第 100 列对应的字母：',get_column_letter(100))
print('AAH 字母列对应的数字：',column_index_from_string('AAH'))
print('B 字母列对应的数字：',column_index_from_string('B'))
```

执行结果如图 24-7 所示：

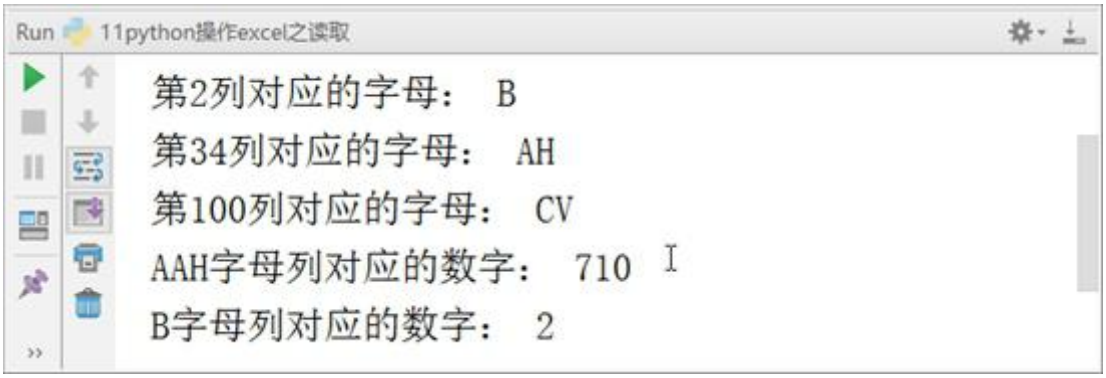


图 24-7 示例 24-10 运行效果图

尚学堂·百战程序员

24.2.2 Python 操作 Excel 之写

【示例 24-11】创建、删除工作表

```
import openpyxl
#python 操作 excel  之写
wb = openpyxl.Workbook()
sheet = wb.active
print(sheet.title)
sheet.title = 'PythonSheet'
print(wb.get_sheet_names())
wb.create_sheet(index=0,title='First Sheet')
wb.create_sheet(index=2,title='Three Sheet')
wb.create_sheet(index=1,title='Two Sheet')
print(wb.get_sheet_names())

#删除工作表
wb.remove_sheet(wb.get_sheet_by_name('Two Sheet'))
print(wb.get_sheet_names())
wb.save('example_copy.xlsx')
```

执行结果如图 24-8 所示：

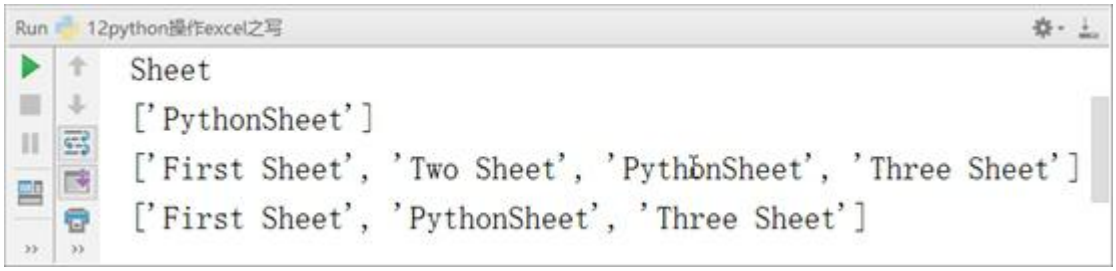


图 24-8 示例 24-11 运行效果图

【示例 24-12】使用 list 写入

```
import openpyxl
#向工作单元写内容
wb = openpyxl.Workbook()
sheet = wb.active
sheet['A1'] = 'Hello python'
print(sheet['A1'].value)

print('使用列表 list 写入'.center(20,'*'))
ws = wb.create_sheet('List sheet')
rows =[['Number','Batch 1','Batch2'],
```

尚学堂·百战程序员

```
            [2,40,30],
            [3,40,25],
            [4,30,24],
            [5,26,28],
            [6,32,21],
            [7,28,31]]
for row in rows:
    ws.append(row)
```

【示例 24-13】使用 range 方式写入

```
import openpyxl
#向工作单元写内容
wb = openpyxl.Workbook()
ws2 = wb.create_sheet('range names')
for row in range(1,40):
    ws2.append(range(17))
```

【示例 24-14】使用 cell()方法写入

```
import openpyxl
from openpyxl.utils import get_column_letter
#向工作单元写内容
wb = openpyxl.Workbook()
ws3 = wb.create_sheet(title='Data')
for row in range(5,30):
    for col in range(15,54):
        ws3.cell(column=col,row=row,value=get_column_letter(col))
wb.save('example_copy.xlsx')
```

24.2.3 Python 操作 Excel 之修改样式

【示例 24-15】修改字体样式

```
import openpyxl
from openpyxl.styles import Font
from openpyxl.styles import colors
wb = openpyxl.Workbook()
#对字体对象的修改
ws = wb.active
ws.title = 'Font'
```

尚学堂·百战程序员

```
#创建字体对象
font = Font(size=24,italic=True)
ws['B3'].font = font
ws['B3'] = '24pt Italic'

boldRedFont = Font(name='微软雅黑',bold=True,color=colors.RED)
ws['A1'].font = boldRedFont
ws['A1'] = 'Bold Red  微软雅黑'
wb.save('style_excelTest.xlsx')
```

执行结果如图 24-9 所示：

图 24-9 示例 24-15 运行效果图

【示例 24-16】设置单元格公式

```
import openpyxl
from openpyxl.styles import Font
from openpyxl.styles import colors
wb = openpyxl.Workbook()
ws = wb.active
#设置单元格公式
ws = wb.create_sheet('Formula')
ws['A1'] = 20
ws['A2'] = 30
ws['A3'] = '=SUM(A1:A2)'
wb.save('style_excelTest.xlsx')
```

执行结果如图 24-10 所示：

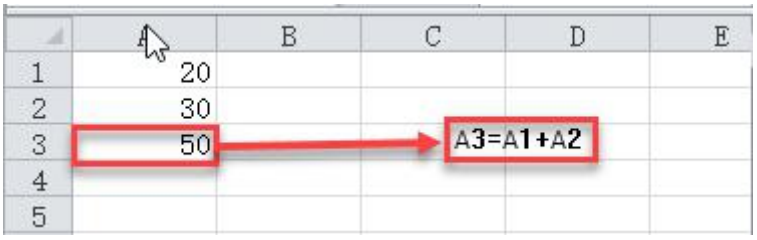


图 24-10 示例 24-16 运行效果图

尚学堂·百战程序员

【示例 24-17】设置行高和列宽

```
import openpyxl
from openpyxl.styles import Font
from openpyxl.styles import colors
wb = openpyxl.Workbook()
ws = wb.active
#设置行高和列宽
ws = wb.create_sheet('dimesions')
ws['A1'] = 'Tall row'
ws.row_dimensions[1].height = 70
ws['B2'] = 'Wide column'
ws.column_dimensions['B'].width = 20
wb.save('style_excelTest.xlsx')
```

执行结果如图 24-11 所示：

图 24-11 示例 24-17 运行效果图

【示例 24-18】合并单元格

```
import openpyxl
from openpyxl.styles import Font
from openpyxl.styles import colors
wb = openpyxl.Workbook()
ws = wb.active
#合并单元格
ws = wb.create_sheet('merged')
ws.merge_cells('A1:D3')
ws['A1'] = 'Twelve cells merged together'
ws.merge_cells('C5:D5')
ws['C5'] ='Two merged cells'
```

尚学堂·互联网程序员

```
wb.save('style_excelTest.xlsx')
```

执行结果如图 24-12 所示：

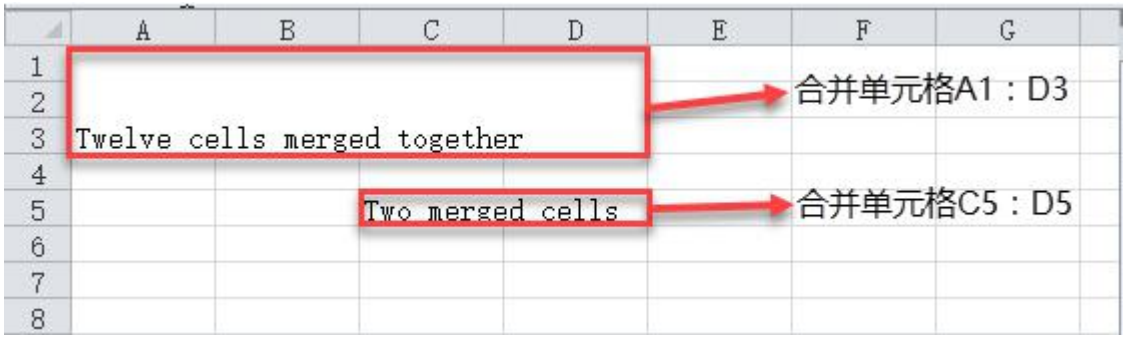


图 24-12 示例 24-18 运行效果图

【示例 24-19】拆分单元格

```
import openpyxl
from openpyxl.styles import Font
from openpyxl.styles import colors
wb = openpyxl.Workbook()
ws = wb.active
#拆分单元格
ws = wb.copy_worksheet(wb.get_sheet_by_name('merged'))
ws.title = 'unmerged'
ws.unmerge_cells('A1:D3')
ws.unmerge_cells('C5:D5')
wb.save('style_excelTest.xlsx')
```

24.2.4 Python 操作 Excel 之图表

1) 饼图

饼图将数据绘制为一个圆的切片，每个切片代表整个百分比。切片按顺时针方向绘制，圆的顶部为 0°。

【示例 24-20】饼图

```
import openpyxl
from openpyxl.chart import (
PieChart,
Reference,
)
```

尚学堂·瓦战程序员

```
#绘制饼状图
wb = openpyxl.Workbook()
ws = wb.active
ws.title = 'pieChart'
data = [
    ['Pie','Sold'],
    ['Apple',50],
    ['Cherry',30],
    ['Pumpkin',10],
    ['Chocolate',40],
]
#单元格写入数据
for row in data:
    ws.append(row)
#创建饼图对象
pie = PieChart()
labels = Reference(ws,min_col=1,min_row=2,max_row=5)
data = Reference(ws,min_col=2,min_row=2,max_row=5)
pie.add_data(data)
pie.set_categories(labels)
pie.title = 'Pies sold by category'
ws.add_chart(pie,'D1')
wb.save('char_excel_text.xlsx')
```

执行结果如图 24-13 所示：

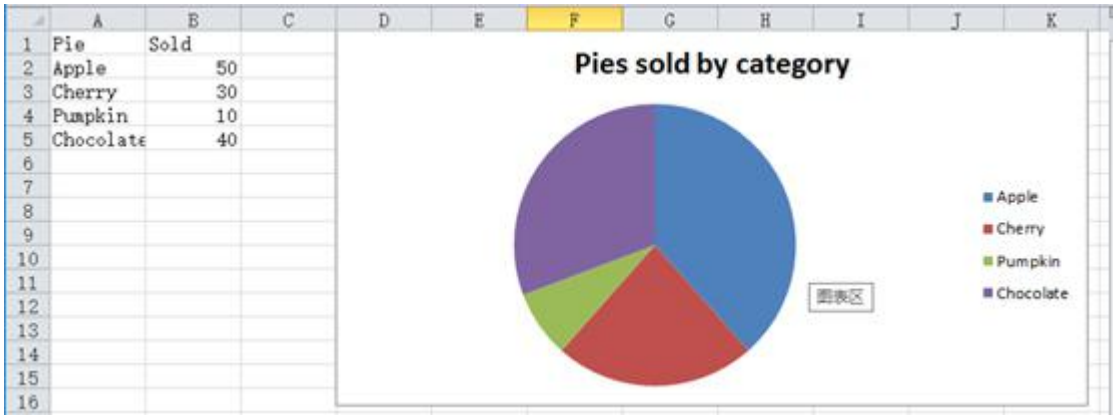


图 24-13 示例 24-20 运行效果图

2) **条形图和柱形图**

尚学堂·百战程序员

在条形图中，值被绘制为水平条或垂直列。可以通过 type 属性来设置。绘制垂直的条形图则使用如下：

```
chart1.type = 'col'
```

绘制成水平条形图示例如下：

```
chart1.type = 'bar'
```

【示例 24-21】条形图

```
import openpyxl
from openpyxl.chart import (
BarChart,
Reference,
)
rows =[['Number','Batch1','Batch2'],
        [2,40,30],
        [3,40,25],
        [4,30,24],
        [5,26,28],
        [6,32,21],
        [7,28,31]]
ws = wb.create_sheet('barChart')
for row in rows:
    ws.append(row)

chart1 = BarChart()
chart1.type = 'col'    #bar    垂直或水平
chart1.style = 15
chart1.title = 'Bar Chart'
chart1.y_axis.title = 'Sample length(mm)'
chart1.x_axis.title = 'Test number'
cats = Reference(ws,min_col=1,min_row=2,max_row=7)
data = Reference(ws,min_col=2,max_col=3,min_row=1,max_row=7)
chart1.add_data(data,titles_from_data=True)
chart1.set_categories(cats)
ws.add_chart(chart1,'E1')
```

尚学堂·互战程序员

```
wb.save('char_excel_text.xlsx')
```

执行结果如图 24-14 所示：

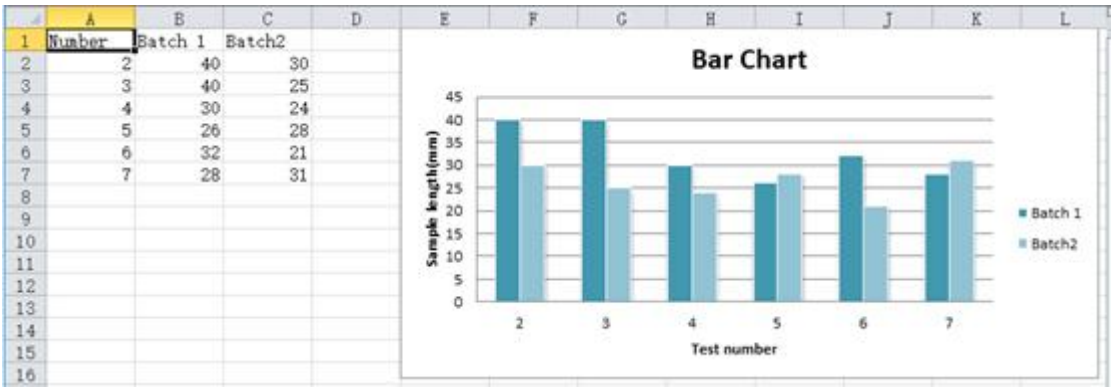


图 24-14 示例 24-21 运行效果图

3) 气泡图

气泡图类似于散点图，但使用第三维来确定气泡的大小。图表可以包括多个系列。

【示例 24-22】气泡图

```
import openpyxl
#绘制气泡图
from openpyxl.chart import (
Series,
BubbleChart,
Reference,
)
rows =  [
    ('商品数量','在 USD 销售','市场份额'),
    (14,12200,15),
    (20,60000,33),
    (18,24400,10),
    (22,32000,42),
    (),
    (12,8200,18),
    (15,50000,30),
    (19,22400,15),
    (25,25000,50)
]
ws = wb.create_sheet('BubbleChart')
```

尚学堂·百战程序员

```
    for row in rows:
        ws.append(row)
    #创建气泡图
    chart = BubbleChart()
    chart.style = 18
    #添加第一组数据
    xvalues = Reference(ws,min_col=1,min_row=2,max_row=5)
    yvalues = Reference(ws,min_col=2,min_row=2,max_row=5)
    size = Reference(ws,min_col=3,min_row=2,max_row=5)
    series = Series(values=yvalues,xvalues=xvalues,zvalues=size,title='2013')
    chart.series.append(series)

    #添加第二组数据
    xvalues = Reference(ws,min_col=1,min_row=7,max_row=10)
    yvalues = Reference(ws,min_col=2,min_row=7,max_row=10)
    size = Reference(ws,min_col=3,min_row=7,max_row=10)
    series = Series(values=yvalues,xvalues=xvalues,zvalues=size,title='2014')
    chart.series.append(series)
    ws.add_chart(chart,'E1')
```

执行结果如图 24-15 所示：

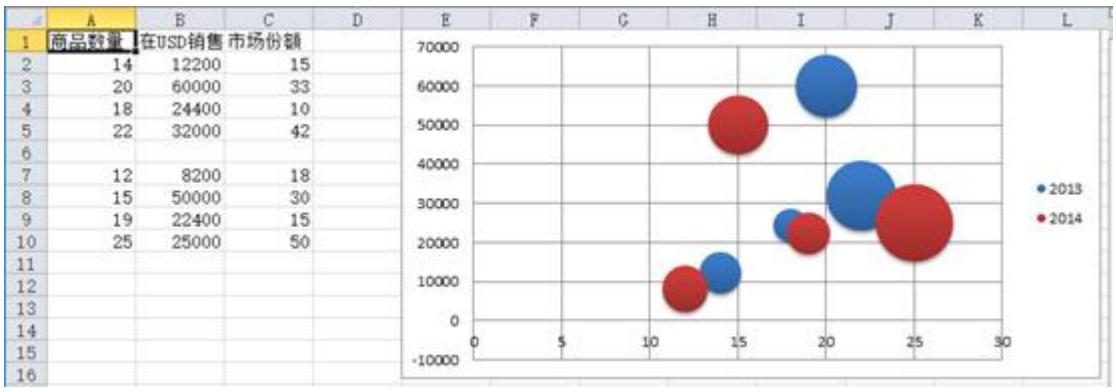


图 24-15 示例 24-22 运行效果图

4) 散点图

散点图或 xy 图类似于某些折线图。主要的区别是一个系列的值相对于另一个系列。当值无序时，这很有用。

【示例 24-23】散点图

```
    import openpyxl
```

尚学堂·百战程序员

```
from openpyxl.chart import (
BarChart,
Reference,
ScatterChart
)
#绘制散点图
rows = [
    ['Size', 'Batch 1', 'Batch 2'],
    [2, 40, 30],
    [3, 40, 25],
    [4, 50, 30],
    [5, 30, 25],
    [6, 25, 35],
    [7, 20, 40],
]

ws = wb.create_sheet('Scatter Chart')
#添加数据
for row in rows:
    ws.append(row)
chart = ScatterChart()
chart.title = '散点图'
chart.style = 13
chart.x_axis.title = 'Size'
chart.y_axis.title = 'Percentage'
xvalues = Reference(ws, min_col=1, min_row=2, max_row=7)
for c in range(2,4):
    yvalues = Reference(ws,min_col=c,min_row=1,max_row=7)
    series = Series(values=yvalues,xvalues=xvalues,title_from_data=True)
    chart.series.append(series)
ws.add_chart(chart,'E1')
wb.save('char_excel_text.xlsx')
```

执行结果如图 24-16 所示：

尚学堂·互战程序员

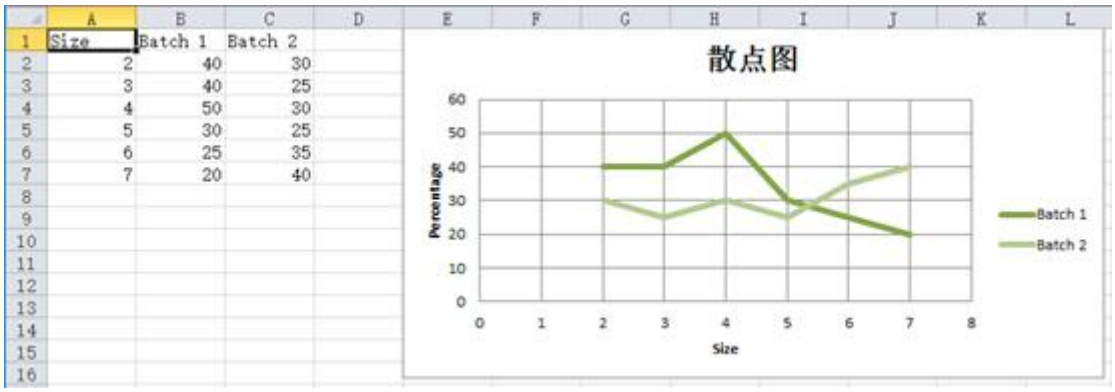


图 24-16 示例 24-23 运行效果图

24.3 数据存储

数据的存储是爬虫开发中一个很重要的环节，而存储的形式也分为很多种，大体来说分为两种。一种是将爬取到的数据储存到文件中，另一种就是将数据直接存储到数据库中。

本节讲解如何将爬取到的数据存储到 TXT，Excel 文件中。其实这些方法大同小异，原理都差不多，只不过是分别运用了不同的库函数，使用的不同的方法而已。

24.3.1 存储至 TXT

存到 TXT 文件是最简单也是最好理解的，主要是用到了 open 函数。我们爬取 http://seputu.com/，获取每个标题下的章节内容、链接。

首先使用 Requests 访问 http://seputu.com/，获取 HTML 文档内容。接着分析 http://seputu.com/首页的 HTML 结构，确定要抽取标记的位置，分析如下：

标题和章节都被包围在 标记下，标题位于其中的 下的<h2>中，章节位于其中的 下的<a>中，代码如下所示。

```
    <div class="bg">
        <div class="mulu">
            <div class="mulu-title">
                <center>
                    <h2>盗墓笔记 1 七星鲁王宫</h2>
                </center>
            </div>
            <div class="box">
                <ul>
                    <li><a href="http://seputu.com/biji1/1.html" title="[2012-5-19 3:3:52] 七星鲁王 第一章 血尸">七星鲁王 第一章 血尸</a></li>
```

尚学堂·百战程序员

【示例 24-24】爬取数据存储至 txt

```
import requests
from bs4 import BeautifulSoup
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.139 Safari/537.36'
}
url = 'http://seputu.com/'
r = requests.get(url,headers=headers)
soup = BeautifulSoup(r.text,'html.parser')
for mulu in soup.find_all(class_='mulu'):
    h2 = mulu.find('h2')
    if h2 != None:
        h2_title = h2.string   #获取标题
        #每个标题的章节内容和 url 保存到该标题的 txt 文件中
        #创建 txt 文件
        with open('e:/txt_data/'+h2_title+'.txt','a',encoding='utf8') as f:
            #获取所有的 a 标记中的 url 和章节内容
            for a in mulu.find(class_='box').find_all('a'):
                href = a.get('href')
                box_title = a.get('title')
                f.write(box_title+'\t'+href+'\n')
```

执行结果如图 24-17 所示：

尚学堂·互战程序员

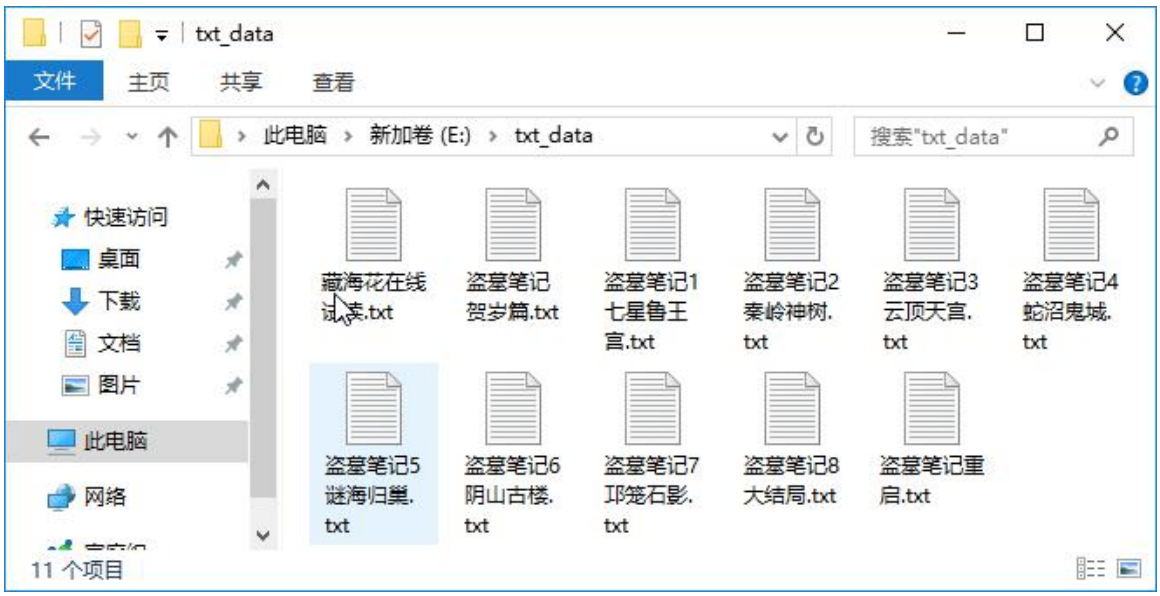


图 24-17 示例 24-24 运行效果图

24.3.2 存储至 excel

上述示例将获取到的章节内容和链接保存到 txt 中，下面我们将数据保存到 excel。

【示例 24-25】爬取数据保存到 excel

```
import requests
from bs4 import BeautifulSoup
import openpyxl
user_agent = 'Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.113 Mobile Safari/537.36'
headers = {'User-Agent':user_agent}
r = requests.get('http://seputu.com/',headers=headers)
soup = BeautifulSoup(r.text,'html.parser')
#创建工作簿
wb = openpyxl.Workbook()
for mulu in soup.find_all(class_='mulu'):
    h2 = mulu.find('h2')
    if h2!= None :
        h2_title = h2.string
        #每个标题创建一个 sheet
        #创建工作表 sheet
        ws = wb.create_sheet(h2_title)
```

尚学堂·百战程序员

```
            ws.append(['txt','href','title'])
            for a in mulu.find(class_='box').find_all('a'):
                href = a.get('href')
                title = a.get('title')
                txt = a.text
                ws.append([txt,href,title])
    wb.save('fiction.xlsx')
```

执行结果如图 24-18 所示：

	A	B	C
1	txt	href	title
2	七星鲁王	http://se	[2012-5-19 3:3:52] 七星鲁王 第一章 血尸
3	七星鲁王	http://se	[2012-5-19 3:5:22] 七星鲁王 第二章 五十年后
4	七星鲁王	http://se	[2012-5-19 3:6:15] 七星鲁王 第三章 瓜子庙
5	七星鲁王	http://se	[2012-5-19 3:30:57] 七星鲁王 第四章 尸洞
6	七星鲁王	http://se	[2012-5-19 4:52:12] 七星鲁王 第五章 水影
7	七星鲁王	http://se	[2012-5-19 4:53:14] 七星鲁王 第六章 积尸地
8	七星鲁王	http://se	[2012-5-22 14:42:52] 七星鲁王 第七章 一百多个人头
9	七星鲁王	http://se	[2012-5-22 14:43:24] 七星鲁王 第八章 山谷
10	七星鲁王	http://se	[2012-5-22 14:44:0] 七星鲁王 第九章 古墓
11	七星鲁王	http://se	[2012-5-22 14:44:35] 七星鲁王 第十章 影子
12	七星鲁王	http://se	[2012-5-22 14:45:8] 七星鲁王 第十一章 七星棺
13	七星鲁王	http://se	[2012-5-22 14:45:44] 七星鲁王 第十二章 门
14	七星鲁王	http://se	[2012-5-22 14:46:15] 七星鲁王 第十三章 02200059

Sheet 盗墓笔记1七星鲁王宫 盗墓笔记2秦岭神树 盗墓笔记3云顶天宫 盗墓笔记4蛇沼鬼城 盗墓笔记5谜海归

图 24-18 示例 24-25 运行效果图

保存零点书屋热门小说精选中的书名和链接到 excel。

首先访问零点书屋网址 https://www.lingdianshuwu.com。接着分析 HTML 文档内容，确定要抽取标记的位置。热门小说精选的书名及链接都被包含在<li class=”new_2”>标记下的<a>中，如图 24-19 所示。

```
▼<ul>
    <li class="new_1">热门小说精选</li>
  ▼<li class="new_2">
      <a href="html/books/3572517.html" target="_blank">天神诀</a>
    </li>
  ▶<li class="new_2">…</li>
  ▶<li class="new_2">…</li>
  ▶<li class="new_2">…</li>
  ▶<li class="new_2">…</li>
```

图 24-19 热门小说精选 HTML 内容

【示例 24-26】使用 bs4 爬取数据保存到 excel

```
import requests
import openpyxl
from bs4 import BeautifulSoup
```

尚学堂·百战程序员

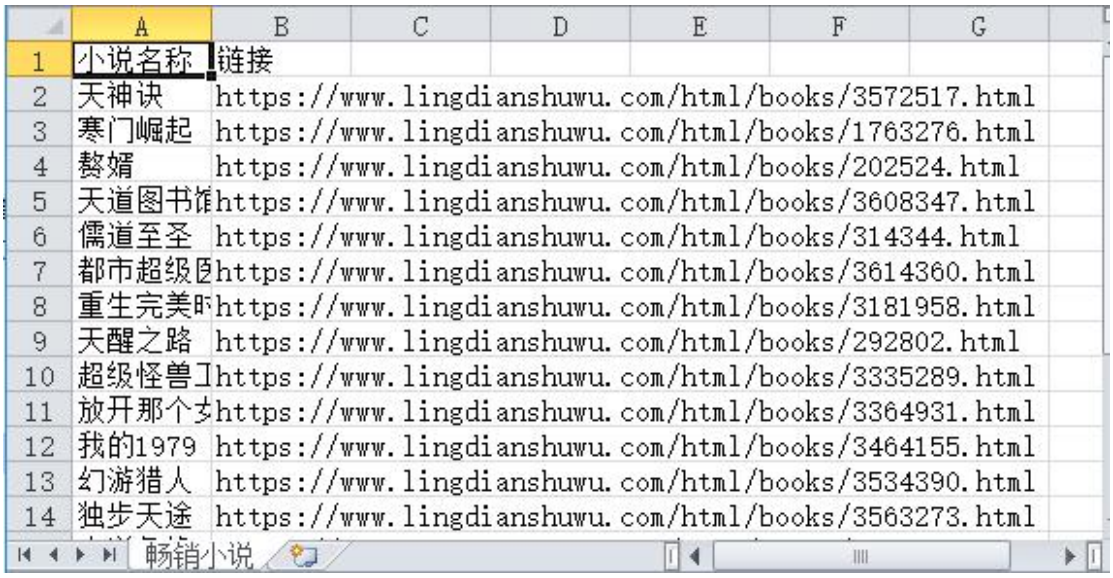
```
#1.获取请求
resp = requests.get('https://www.lingdianshuwu.com')
resp.encoding='gb2312'
#2.解析数据
bs = BeautifulSoup(resp.text,'html.parser')
#3.获取数据
li_tags = bs.find_all('li',class_='new_2')

#4.提取数据
li_a =[['小说名称','链接']]
for item in li_tags:
    a_name = item.text    #  名称
    a_url = 'https://www.lingdianshuwu.com/'+item.find('a')['href']
    li_a.append([a_name,a_url])

#将数据保存到 excel 中
wb = openpyxl.Workbook()
ws = wb.active
ws.title = '畅销小说'
for row in li_a:
    ws.append(row)
wb.save('story.xlsx')
```

执行结果如图 24-20 所示：

尚学堂·百战程序员



	A	B	C	D	E	F	G
1	小说名称	链接					
2	天神诀	https://www.lingdianshuwu.com/html/books/3572517.html					
3	寒门崛起	https://www.lingdianshuwu.com/html/books/1763276.html					
4	赘婿	https://www.lingdianshuwu.com/html/books/202524.html					
5	天道图书馆	https://www.lingdianshuwu.com/html/books/3608347.html					
6	儒道至圣	https://www.lingdianshuwu.com/html/books/314344.html					
7	都市超级医	https://www.lingdianshuwu.com/html/books/3614360.html					
8	重生完美时	https://www.lingdianshuwu.com/html/books/3181958.html					
9	天醒之路	https://www.lingdianshuwu.com/html/books/292802.html					
10	超级怪兽卫	https://www.lingdianshuwu.com/html/books/3335289.html					
11	放开那个女	https://www.lingdianshuwu.com/html/books/3364931.html					
12	我的1979	https://www.lingdianshuwu.com/html/books/3464155.html					
13	幻游猎人	https://www.lingdianshuwu.com/html/books/3534390.html					
14	独步天途	https://www.lingdianshuwu.com/html/books/3563273.html					

畅销小说

图 24-20 示例 24-26 运行效果图

保存起点月票榜的书名、作者到 excel 中。

首先访问起点月票网址 https://www.qidian.com/rank/yuepiao。接着分析 HTML 文档内容，确定要抽取标记的位置。起点月票榜的小说名称在 下的<h4>下的<a>标记中，作者在<p class='author'>下的第一个<a>中，如图 24-21 所示。

```
<div class="book-mid-info">
  <h4>
    <a href="//book.qidian.com/info/1013562540" target="_blank" data-eid="qd_C40" data-bid="1013562540">第一序列</a>
  </h4>
  <p class="author">
    <img src="//qidian.gtimg.com/qd/images/ico/user.f22d3.png">
    <a class="name" href="//me.qidian.com/authorIndex.aspx?id=5175790" target="_blank" data-eid="qd_C41">会说话的肘子</a>
    <em>|</em>
    <a href="//www.qidian.com/dushi" target="_blank" data-eid="qd_C42">都市</a>
    <em>|</em>
    <span>连载</span>
  </p>
  <p class="intro">…</p>
  <p class="update">…</p>
</div>
```

图 24-21 起点月票榜 HTML 内容

【示例 24-27】使用 XPath 爬取数据保存到 excel

```
import  requests
from lxml import etree
import openpyxl
#请求 url
url = 'https://www.qidian.com/rank/yuepiao'
header = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36'}
resp = requests.get(url,headers=header)
```

尚学堂·百战程序员

```
e = etree.HTML(resp.text)
names = e.xpath('//div[@class="book-mid-info"]/h4/a/text()')
authors = e.xpath('//p[@class="author"]/a[1]/text()')
# 提取数据
li_a =[['名称','作者']]
for name,author in zip(names,authors):
    li_a.append([name,author])
#将数据保存到 excel 中
wb = openpyxl.Workbook()
ws = wb.active
ws.title = '起点月票榜'
for row in li_a:
    ws.append(row)
wb.save('qidian.xlsx')
```

执行结果如图 24-22 所示：

	A	B	C	D	E
1	名称	作者			
2	第一序列	会说话的肘子			
3	万族之劫	老鹰吃小鸡			
4	诡秘之主	爱潜水的乌贼			
5	我师兄实在太稳健了	言归正传			
6	御九天	骷髅精灵			
7	这号有毒	幼儿园一把手			
8	当医生开了外挂	手握寸关尺			
9	明天下	孑与2			
10	大奉打更人	卖报小郎君			
11	平平无奇大师兄	黑夜弥天			
12	亏成首富从游戏开始	青衫取醉			
13	东晋北府一丘八	指云笑天道1			
14	圣光	通吃道人.QD			

起点月票榜

图 24-22 示例 24-27 运行效果图

尚学堂·百战程序员

习题

一、 编码题

1. 纯文本文件 student.txt 为学生信息， 里面的内容（包括花括号）如下所示：

```
{ "1":["张三",150,120,100], "2":["李四",90,99,95], "3":["王五",60,66,68] }
```

请将上述内容写到 student.xls 文件中。

2. Python 操作 Excel，读取 student.xls 文件中的内容。
3. 使用 bs4 爬取零点书屋热门小说精选中的书名和链接保存到 bs4_data.xls 中。
4. Python 操作 Excel，读取 bs4_data.xls 文件中的内容。
5. 使用 XPath 爬取起点月票榜的书名、作者及链接到 xpath_data.xls 中。
6. Python 操作 Excel，读取 xpath_data.xls 文件中的内容。

尚学堂·百战程序员