

# 第八讲 - 文件读写

张建章

阿里巴巴商学院

杭州师范大学

2022-09



1 open 函数

2 文件内容读取

3 文件内容写入

4 文件内移动游标

5 文件读写实例

6 文件读写工具包

本讲主要学习文本文件的打开和读写操作，暂不涉及操作其他类型的文件（如，图像、音视频）。使用内置函数 `open` 打开文本文件，其用法如下：

```
file_object = open(file_path, open_mode)
```

其中，参数 `file_path` 表示文件路径，没有默认值，必须提供；参数 `open_mode` 表示打开模式，默认值为 `rt` 表示文本只读模式。

```
f = open('./stopwords.txt') # 打开存储停用词表的文件
stopwords_str = f.read() # 读取文件中的全部内容
f.close() # 关闭文件
# 将文件中的内容stopwords_str先去掉一层引号，再作为Python语句运行
stopwords_list = eval(stopwords_str) # 将文件中的内容转化为列表

# with关键字和open配合使用可以避免手动关闭文件
with open('./stopwords.txt') as f:
    stopwords_str = f.read()
stopwords_list = eval(stopwords_str)
```

注意：推荐 `open` 函数和关键字 `with` 配合使用，避免手动关闭文件。

## 文件路径

**绝对路径:** 从根目录开始到文件的路径, 如:

`/home/zjz/Desktop/stopwords.txt` (linux)

`C:\Users\zjz\Desktop\stopwords.txt` (Windows)

**相对路径:** 从当前目录到文件的路径, 如:

`./stopwords.txt` (linux/Windows, 假定当前目录为桌面)

**注意:** 推荐使用相对路径, 提高代码的可移植性。如果四六级高频词作业我使用绝对路径, 每个同学拿到作业题干后都需要对绝对路径进行修改后才能运行, 这样的代码可移植性太差。

## 打开模式

打开模式参数包含三部分，格式如下：

```
r/w/a t/b [+]
```

**第一部分**为读写模式，有如下三种选择 (三选一):

r: 只读 (only read), 是读写模式的默认值;

w: 只写, 文件不存在则创建, 文件存在, 则清空已有内容, 写入新内容;

a: 只追加写入, 文件不存在则创建, 文件存在, 则在已有内容之后追加写入新内容。

**第二部分**为文件格式, 有如下两种选择 (二选一):

t: 文本格式, 是文件格式的默认值;

b: 二进制格式, 一般用于打开图片、视频等非文本文件。

**第三部分**的加号表示在原有基础上可读可写 (可选)

常用的打开模式参数有: rt, wt, at, t 是默认文件格式, 可以省略, 即, r, w, a。

```
# 读取stopwords.txt中保存的通用英语停用词
with open('./stopwords.txt') as f:
    stopwords_str = f.read()

# 新建领域停用词表，写入三个停用词
with open('./domain_stopwords.txt', 'w') as f:
    f.write('part\nquestions\nexam') # \n表示换行

# 在新建的领域停用词表中，追加写入两个停用词
with open('./domain_stopwords.txt', 'a') as f:
    f.write('\nminutes\ncollege') # \n表示换行
```

除上例中常用的三种打开模式参数外，我们看几个复杂例子 (不常用)。rb+ 表示以二进制格式打开一个文件，可读可写，若文件不存在不会创建，wb+，表示以二进制格式打开一个文件，可读可写，若文件不存在则创建一个。

使用 `open` 函数打开文件后，可用下面三个方法读取文件中的内容：

① `read`，它有一个可选非关键字参数 `size`，指定读取的内容长度，该参数默认值为-1，表示读取文件中的全部内容。

```
# 读取stopwords.txt中的前7个字符，空格和换行符都要计入
# ['i', \n
with open('./stopwords.txt') as f:
    stopwords_str = f.read(7) # f.read(size=7)是错误写法，
    ↪ 因为size是非关键字参数

# 读取stopwords.txt中的全部内容
with open('./stopwords.txt') as f:
    stopwords_str = f.read()
```

② **readline**，读取文件中一行的内容，包括换行符在内，它有一个可选非关键字参数 **size**，指定读取的内容长度，该参数默认值为-1，表示读取一整行内容。

```
# 读取一行内容，['i',\n
with open('./stopwords.txt') as f:
    line = f.readline()
```

```
# 读取一行内容的前2个字符，['
with open('./stopwords.txt') as f:
    line = f.readline(2)
```

```
# 指定的长度超过一行的长度，只返回一行内容，['i',\n
with open('./stopwords.txt') as f:
    line = f.readline(20)
```



③ `readlines`，读取文件中的全部内容，返回一个字符串列表，文件中的每一行 (包括换行符) 是列表中的一个元素。

```
# ['part\n', 'question\n', 'exam\n', 'minutes\n', 'college']  
with open('./domain_stopwords.txt') as f:  
    lines = f.readlines()
```

```
# 使用for循环逐行读取内容 (包括每行末尾的换行符)并处理  
with open('./domain_stopwords.txt') as f:  
    for line in f:  
        print(len(line.strip()))
```

注意: `read` 和 `readline` 方法返回的是字符串，`readlines` 方法返回的是字符串列表。

使用 `open` 函数打开文件后，可用下面两个方法把字符串写入文件中：

① `write`，把字符串写入到文件中。

```
# 新建领域停用词表，写入三个停用词
content = 'part\nquestions\nexam'
with open('./domain_stopwords.txt', 'w') as f:
    f.write(content) # \n表示换行

# 在新建的领域停用词表中，追加写入两个停用词
with open('./domain_stopwords.txt', 'a') as f:
    f.write('\nminutes\ncollege') # \n表示换行
```

② `writelines`，将字符串列表写入文件，即，使用空字符作为连接符，将一个字符串列表拼接为一个字符串，再将字符串写入文件。

```
lines = ['part\n', 'question\n', 'exam']
with open('./domain_stopwords.txt', 'w') as f:
    f.writelines(lines)

with open('./domain_stopwords.txt', 'w') as f:
    f.write(''.join(lines))

lines = ['part', 'question', 'exam']
with open('./domain_stopwords.txt', 'w') as f:
    f.write('\n'.join(lines))
```

注意: 没有 `writeline` 方法。

打开文件后可以使用 `tell` 方法查看游标位置，用 `seek` 方法移动游标 (cursor)，这两个方法的单位均为字节 (byte)，一个英文字符占一个字节，一个 UTF 编码的中文字符占 3 个字节。

`seek` 方法有两个非关键字参数，`offset` 指定移动的字节数，没有默认值，不能为负，`whence` 指定游标从哪个位置开始移动，默认值为 0，表示从文件头开始移动，另外两个可选值为 1 和 2，分别表示从当前位置和文件末尾开始移动。

```
lines = ['大学', 'exam', '英语考试']
with open('./domain_stopwords.txt', 'w', encoding = 'utf-8') as f:
    f.write('\n'.join(lines))

f = open('./domain_stopwords.txt', 'rb')
f.tell() # 0
f.seek(3) # 3
f.readline() # b'\xe5\xad\xa6\n'
type(b'\xe5\xad\xa6\n') # bytes
b'\xe5\xad\xa6\n'.decode('utf-8') # 学\n
```

```
f.tell() # 7
f.seek(3,1) # 10
f.readline() # m\n
f.tell() # 12
f.seek(2,2) # 26
f.seek(6,0) # 6
f.readline() # \n
f.seek(-3,1) # 3
f.readline().decode('UTF-8') # 学\n
```

# 下面查看文件对象 *f* 的一些属性

```
f.closed # False, 文件处于未关闭状态
f.encoding # utf-8, 文件编码方式为utf-8
f.mode # rb, 文件打开模式为二进制只读
f.name # ./domain_stopwords.txt, 文件路径
f.close() # 关闭文件, 一定要记得关闭文件
f.closed # True, 文件处于关闭状态
```

**注意:** 使用 `open` 函数打开文件, 文件读写操作完成后, 一定要记得关闭文件, 如果你怕忘记关闭, 推荐 `open` 函数和 `with` 关键字搭配使用。

内置函数 `print` 不仅能够在屏幕打印输出，还可以将内容写入到文件，如下示例：

```
lines = ['part', 'question', 'exam']
with open('./domain_stopwords.txt', 'w') as f:
    for line in lines:
        print(line, end='\n', file = f)

with open('./domain_stopwords.txt', 'r') as f:
    content = f.read() # part\nquestion\nexam\n
```

`print` 函数的 `end` 参数指定结尾字符，默认值为换行符 `\n`。

使用 `input` 函数接收键盘输入，并写入文件。

```
# 给出一个上联，使用input函数接收键盘输入下联，
↪ 并将上下联写入文件duilian.txt
left_text = '南通州，北通州，南北通州通南北' # 东当铺，西当铺，
↪ 东西当铺当东西
right_text = input('上联是： {}, 请您赐下联： '.format(left_text))
with open('./duilian.txt', 'w') as f:
    print('我的上联是： {}'.format(left_text))
    print(left_text, file = f)
    print('您的下联是： {}'.format(right_text))
    print(right_text, file = f)
```

思考：修改上述代码，使对联在文件中竖排显示。

文件 `grades.txt` 中保存的内容如下图所示，请你编程：

- ① 读取 `grades.txt` 文件中的内容；
- ② 解析读取的内容，计算每位同学的平均分；
- ③ 将计算结果写入 `average.txt` 文件中，第一列为姓名，第二列为平均分。

| 姓名 | 语文 | 数学 |
|----|----|----|
| 张三 | 88 | 64 |
| 李四 | 90 | 78 |
| 王五 | 65 | 45 |
| 赵六 | 43 | 66 |
| 冯七 | 62 | 56 |
| 刘八 | 90 | 42 |
| 阮九 | 68 | 36 |
| 陆十 | 86 | 46 |

点击下载 `grades.txt` 文件 (右键-另存为即可保存到本地)。



```
# 打开文件，读取内容
with open('./grades.txt') as f:
    lines = f.readlines()

# 解析内容，计算平均分
new_lines = ['姓名\t平均分']
for line in lines[1:]:
    l = line.strip().split()
    name = l[0]
    avg = sum(map(float, l[1:]))/len(l[1:])
    avg_str = str(avg)
    new_lines.append(name + '\t' + avg_str)

# 将结果写入文件
with open('./average.txt', 'w') as f:
    f.write('\n'.join(new_lines))
```

为便于各类文本文件读写，参照上一讲模块与包的内容，我自己编写了一个文件读写 `package`，可用于读写多种文本文件，如 `txt`, `xlsx`, `csv` 等，安装使用方法如下：

① 点我下载 `utils.zip`;

② 解压 `utils.zip` 得到 `utils` 文件夹，将该文件夹放置到 `anaconda3\Lib\` 中；

③ 开始使用，示例如下

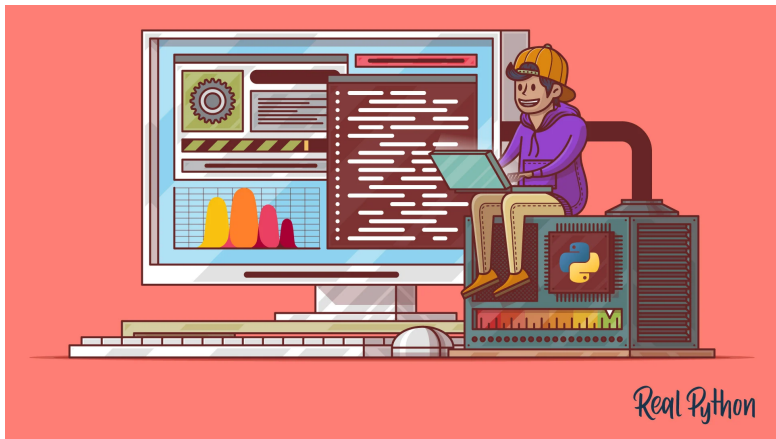
```
# 指定windows平台下Python运行时的默认编码类型为UTF-8
import _locale
_locale._getdefaultlocale = (lambda *args: ['zh_CN', 'utf8'])

from utils.fileUtils import *

lines1 = readExcelToList('./账面.xlsx')
len(lines1) # 查看行数是否与excel文件中的行数相符
lines1[:3] # 查看内容和格式是否与excel中的相符
help(readExcelToList) # 查看readExcelToList函数用法
```

## 写在课程最后的话

学习这门课程的目的是为了培养**计算思维**，为今后的学习、工作提供一种高效的工具。



这个任务能用 Python 更高效解决吗，如果能，应该如何分解问题，对于分解后的问题，通过查阅互联网寻找解决方法，整合解决方案。

THE END