### 文件操作

文件对象的介绍（分类）

原始二进制文件

缓冲二进制文件

文本文件

流（文件对象）

文件对象file object 对外提供面向下层资源的对象，带有read()和write()这样的方法 文件对象也被称为文件类对象或流(stream)

接口：代码，提供了操作硬件方法， io的模块(输入可输出)

##### python基本文件操作

创建文件对象

python使用文件对象能做大部分的文件操作

In [4]: f = open('test1')

---------------------------------------------------------------------------

FileNotFoundError Traceback (most recent call last)

<ipython-input-4-61a913681263> in <module>

----> 1 f = open('test1')

FileNotFoundError: [Errno 2] No such file or directory: 'test1'

In [8]: f = open('test.py')

In [9]: f

Out[9]: <\_io.TextIOWrapper name='test.py' mode='r' encoding='UTF-8'>

打开文件以什么方式：open(‘file’,’mode’) 默认是读

文件的访问模式 注意：写的打开执行写就清空了文件

|  |  |  |  |
| --- | --- | --- | --- |
| 模式 | 描述 | 模式 | 描述 |
| r | 以只读的方式，  这是默认模式 | r+ = r+a | 以可读可写方式打开 |
| w | 以只写的方式  没有文件创建 | w+ = w+r | 以可读可写方式打开 |
| a | 以追加方式打开一个文件 | a+ = a+r | 打开一个文件用于读写，文件存在，放才结尾，不存在新建 |

操作文件的内置方法：

|  |  |  |  |
| --- | --- | --- | --- |
| 方法 | 描述 | 参数 | 案例 |
| open() | 打开文件 | w,a  w w+ r+ 覆盖写入  a a+ 追加写入 | f = open(‘test.py’)  f = open(‘test.py’,’w’)  f = open(‘test.py’,’a’) |
| f.read() | 读取文件 默认整个读取 | size读取几个字符 | In [51]: f.read(10)  Out[51]: '#!/usr/bin' |
| f.write() | 写文件 | write(string) | In [70]: f.write('hello')  Out[70]: 5  In [65]: f.close()  In [66]: cat test.py  hello |
| f.close() | 关闭文件,也是保存 |  | f.close() |
| f.readline() | 读文件一行一行的读 |  | In [9]: f.readline()  Out[9]: '#!/usr/bin/env python3\n' |
| f.readlines() | 列表形式返回文件内容一行为一个元素 |  | In [3]: f.readlines()  Out[3]: ['a "\n', 'bc\n', "'\n"] |
| f.writelines() | 把一个文件写入另一个文件的方法 |  | 第六个案例 |

1.文件存在：  
 覆盖写入：把内容写入到文件

追加写入

2.文件不存在

创建文件

3.读取文件的完整过程：

1. 打开文件
2. f.read()读取文件
3. f.close()关闭文件

4.写入文件的完整过程：

1.创建文件像

2.f.write()把内容写到缓存

3.关闭文件，保存文件到硬盘

5.写入列表的全过程：

1. 打开文件
2. f.write(str([1,2,3]))
3. 关闭文件

In [112]: f = open('test.py','a')

In [113]: f.write(str({'name':'wwine'}))

Out[113]: 17

In [114]: f.close()

In [115]: cat test.py

worldhellohello[1, 2, 3]{'name': 'wwine'}

6.把一个文件里的所有内容写入另一个文件

In [11]: f = open('test01.py','w')

In [12]: f1 = open('test.py')

In [13]: f.writelines(f1.read())

In [15]: f.close()

In [16]: f1.close()

In [10]: with open('test.py') as f:

...: for i in f:

...: print(i)

这样就不用关闭文件了不然要close一下

##### python文件的进阶操作：

import os 导入os模块

|  |  |  |
| --- | --- | --- |
| 操作 | 解释 | 说明 |
| os.mkdir(‘/dir’) | 根下边创建dir目录 | 创建目录 |
| os.makedirs('dir/a/b') | mkdir -p | 创建目录 |
| os.rename(‘dir’,’b’) | dir目录重命名为b | 重命名 |
| os.listdir('.') | 查看当前目录的文件及目录 | 遍历目录 |
| os.walk(path) | 迭代这个目录，遍历后返回目录，一层一层返回，是一个三元组  格式：(根目录，[目录],[文件]) | 迭代 |
| os.remove('a.txt') | 删除当前目录下a.txt | 删除文件 |
| os.mknod(‘a.txt’) | 创建a.txt | 创建文件 |
| os.stat('c.txt') | 查看时间。后面跟.st\_mtime就会返回时间 | 查看一些参数时间啥的 |

手动迭代：

In [72]: def f():

...: for i in os.walk('touch'):

...: yield i

...:

In [73]: d = f()

In [74]: d.\_\_next\_\_()

自动迭代

In [71]: for paths,dirs,files in os.walk('touch'):

...: print('路径',paths,'下的文件及目录')

...: for i in files:

...: print(i)

...: for i in dirs:

...: print(i)

##### python高级文件操作

import shutil

|  |  |  |
| --- | --- | --- |
| 方法 | 作用 | 说明 |
| shutil.copy(‘src’,’dst’) | 拷贝文件 | 复制 |
| shutil.copy2(‘src’,’dst’) | 拷贝文件，文件任何数据不变（元数据） | 复制 |
| shutil.copytree(‘src’,’dst’) | 拷贝目录 | 复制 |
| shutil.move(‘src’,’dst’) | 移动文件或目录 | 移动，相当于mv |
| shutil.rmtree(path) | 导入模块 | 删除目录 |

案例：

file = shutil.ignore\_patterns('nginx.conf','a,txt.bak','b.txt')

shutil.copytree('nginx/','nginx.bak',ignore=file)



##### python 的常用路径操作

判断文件目录，存在不存在

import os

|  |  |  |  |
| --- | --- | --- | --- |
| 路径操作 |  | 案例 | 结果 |
| os.path.join(path) | 拼接路径 | os.path.join('usr','local','src')  os.path.join('/usr','local','src') | 'usr/local/src'  /usr/local/src' |
| os.path.isdir(path) | 判断是否为目录 | os.path.isdir('nginxx/')  os.path.isdir('c.txt') | True  False |
| os.path.isfile(file) | 判断是否为文件 | os.path.isfile('c.txt')  os.path.isfile('nginxx/') | True  False |
| os.path.exists(path) | 判断路径是否存在 | os.path.exists('/a')  os.path.exists('/usr') | False  True |