# Python学习笔记-文件操作

Python提供了默认操作文件所必需的基本功能和方法，可以使用文件对象执行大部分文件操作。

## 创建文件对象模式

file = open(file\_name [, access\_mode] [, buffering])

* file\_name : 参数是一个字符串值，指定要访问的文件的名称。
* access\_mode : 确定文件打开的模式，即读取，写入，追加等。可能的值的完整列表如下表所示。 这是一个可选参数，默认文件访问模式为(r - 也就是只读)。
* buffering : 如果buffering值设置为0，则不会发生缓冲。 如果缓冲值buffering为1，则在访问文件时执行行缓冲。如果将缓冲值buffering指定为大于1的整数，则使用指定的缓冲区大小执行缓冲操作。如果为负，则缓冲区大小为系统默认值(默认行为)。

## 几种常用的文件使用模式

* r:打开的文件为只读模式,文件指针位于文件的开头,这是默认模式
* r+:打开读写文件,文件指针放在文件的开头
* w:打开仅供写入的文件,如果文件存在,则覆盖该文件, 如果文件不存在,则创建一个新文件进行写入
* w+:打开写入和取读的文件,如果文件存在,则覆盖现有文件,如果文件不存在,创建一个新文件进行阅读和写入
* a:打开一个文件进行追加.如果文件存在,则文件指针位于文件末尾,也就是说,文件处于追加模式,如果文件不存在,它将创建一个新文件进行写入
* a+:打开一个文件,用于追加和阅读,如果文件存在,则文件指针位于文件末尾,文件以附加模式打开, 如果文件不存在,它将创建一个新文件进行阅读和写入

## 文件对象属性

* file.closed:如果文件关闭则返回true,否则返回false
* file.mode:返回打开文件的访问模式
* file.name:返回文件的名称

## 文件对象方法

* file.close():关闭文件,无法读取或写入关闭的文件
* file.flush():清空内部缓冲区,类似于stdio的fflush
* file.fileno():返回底层实现使用的整数文件描述符,以从操作系统请求I/O操作
* file.isatty():如果文件连接到tty(-like)设备,则返回True,否则返回False
* next(file):每次调用时返回文件的下一行
* file.read([size]):从文件中读取最多为size个字节(如果在获取size字节之前读取命中EOF,则读取更少字节的数据)
* file.readline([size]):从文件中读取一行,字符串中保留一个尾随的换行字符
* file.readlines([sizeint]):使用readline()读取并返回一个包含行的列表直到EOF,如果可选的sizehint参数存在,而不是读取到EOF,则读取总共大约为sizehint字节的字符串(可能在舍入到内部缓冲区大小之后)的整行
* file.seek(offset[, whence]):设置文件的当前位置
* file.tell():返回文件的当前位置
* file.truncate([size]):截断文件大小,如果可选的size参数存在,则该文件将被截断为size(最多)大小
* file.write(str):将一个字符串写入文件,无返回值
* file.writelines(sequence):将一串字符串写入文件,该序列可以是生成字符串的任何可迭代对象,通常是字符串列表