第九章 文件和输入输出

- 文件对象
- 文件内建函数-`open()`和`file()`
- 文件内建方法
- 文件内建属性
- 标准文件
- 命令行参数
- 文件系统
- 文件执行
- 永久存储模块
- 相关模块
- 练习

文件对象

打开一个URL页面来读取Web页面，其实也类似于发开一个网络上的文件而已，只不过这个文件需要通过网络才能访问到，其他的操作与本地文件没有区别。

文件只是连续的字节序列。数据的传输会用到字节流，无路怒字节流是由单个字节还是大块数据组成的

文件内建函数[`open()`和`file()`]

`file_object = open(file_name, access_mode='r', buffering=-1)`
参数buffering表示访问文件时采用的缓冲方式。
0–不缓冲
1–缓冲1行数据
N–标识缓冲区的大小
不提供或者负值–默认缓冲机制(行缓冲)

工厂函数 `file()`

内建函数`dict(), bool(), file(),list(), str(),`等这些用来创建对象的实例。

Universal NEWLINE Support(UNS)

在读取文件是若使用'U'标志，所有的行分隔符(无论原来是什么)通过Python的输入方法都会被替换为NEWLINE(\n)

文件内建方法

文件的方法分为四类:输入、输出、文件内移动以及杂项操作。

输入

`file.read([size])`
将读取字节到字符串中，最多读取给定size的字节数，如果没有给size参数或者为负值(默认为-1)，则将读取至末尾。

`file.readline([size])`
读取打开文件的一行(包含行结束符)，整行作为字符串返回。默认为-1表示读至行结束符，如果超过size个字节将会返回不完整的行。

`file.readlines([sizehint])`
读取所有(剩余的)行作为一个字符串列表返回。sizehint 代表返回的最大字节大小，缓冲区大小

可以使用`iter(file)`去进行文件的迭代

```
with open('mydata.txt') as fp:
    for line in iter(fp.readline, ''):
        process_line(line)
```

输出

`file.write(str)`–把含有文本数据或二进制的数据块的字符串写入到文件中去。

`file.writelines(sequence)`–将字符串列表写入文件中

以上者两个方法并不会加入换行符。

文件内移动

`file.seek(offset[, whence])`
`os.SEEK_SET` or 0 (absolute file positioning); other values are `os.SEEK_CUR` or 1 (seek relative to the current position) and `os.SEEK_END` or 2 (seek relative to the file's end).

`text()`–返回当前指针在文件中的位置(单位为字节

文件迭代

```
#eachLine表示文件的一行，包括换行符
for eachLine in f
    print eachLine
```

文件迭代相比其他方法(`readline()等`)更为高效，`iter()`和`next()`方法结合使用来迭代文件。

其他

`close()`
`fileno()`–返回打开文件的文件描述符(fd)
`flush()`
`file.truncate([size])`–Truncate the file's size.

文件方法杂项

使用文件迭代器

```
filename = raw_input('Enter file name: ')
f = open(filename, 'r')
for eachLine in f:
    print eachLine, f.close()
```

关于换行符

系统	使用的换行符
POSIX(Unix系列)或者Mac OS X	NEWLINE(\N)
Dos/Win32	\r\n

路径分隔符

系统	使用的换行符
POSIX(Unix系列)	/
Dos/Win32	\

Python 的os模块屏蔽了底层的多样性，提供了统一的接口。

跨平台的os模块

os模块属性	描述
linesep	文件中分隔行的字符串
sep	分隔文件路径名的字符串
pathsep	分隔文件路径的字符串
curdir	当前工作目录的字符串名称
pardir	父目录字符串名称

print 语句默认在输出内容末尾后加一个换行符, 而在语句后加一个逗号就可以避免这个行为

文件内建属性

`file.closed`
`file.encoding`
`file.mode`
`file.name`
`file.newlines`
`file.softspace` –Boolean that indicates whether a space character needs to be printed before another value when using the print statement.为 0 表示在输出一数据后，要加上一个空格符，1 表示不加

标准文件

`stdin`, `stdout`, `stderr` 三个标准输入，标准输出，标准错误文件。
模块sys可以访问这些文件
`sys.stdin` – `raw_input()` 则通常从 `stdin` 接受输入
`sys.stdout` – `print` 语句通常输入到 `sys.stdout`

命令行参数

`sys.argv`的第一项`sys.argv[0]`代表程序的名称
`len(sys.argv)`就是参数个数argc了

模块getopt和optparse用来辅助处理命令行参数。

文件系统

os模块可以实现对文件系统的访问。
os 模块负责处理大部分的文件系统操作和对进程和进程运行环境的管理

os的子模块 os.path 可以完成一些针对文件路径名的处理.

这两个模块提供了与平台无关的文件系统访问。

永久存储模块

模块`marshal`和`pickle`用来转换并存储Python对象。

`marshal` 只能处理简单的 Python 对象
`pickle` 还可以处理递归对象, 被不同地方多次引用的对象, 以及用户定义的类和实例

Python 的另一些模块`dbhash/bsddb, dbm, gdbm, dumbdbm 等`以及它们的"管理器"(`anydbm`)只提供了 Python 字符串的永久性储存。

数据的序列化/数据的扁平化/数 据 的 顺 序 化:
将复杂的对象转换为一个二进制数据集合，可以保存或者通过网络发送，然后在重新把数据集合恢复为原来的格式

DBM 风格的模块

最好使用 `anydbm` 模块, 它会自动检测系统上已安装的 DBM 兼容模块

shelve 模块

shelve 模块允许对数据库文件进行并发的读访问, 但不允许共享读/写访问.

核心模块: pickle 和 cPickle
pickle 模块会创建一个 Python 语言专用的二进制格式, 你不需要考虑任何文件细节, 它会帮你干净利索地完成读写对象操作, 唯一需要的只是一个合法的文件句柄.

pickle 模块中的两个主要函数是 dump() 和 load()

pickle 模块中的两个主要函数是 `dump()` 和 `load()`

cPickle 是 pickle 的一个更快的 C 语言编译版本.

相关模块

与文件和输入输出相关的模块
`base64,binascii,bz2,csv,filecmp,fileinput,getopt/optparse,glob/fnmatch,gzip/zlib,shutil,c/StringIO,tarfile,tempfile,uu,zipfile`

`socket,popen()2,fdopen(),urllib`

练习