## [关于python文件操作](http://www.cnblogs.com/rollenholt/archive/2012/04/23/2466179.html)

2012-04-23 11:28 by Rollen Holt, 144885 阅读, 8 评论, [收藏](http://www.cnblogs.com/rollenholt/archive/2012/04/23/2466179.html), [编辑](http://i.cnblogs.com/EditPosts.aspx?postid=2466179)

总是记不住API。昨晚写的时候用到了这些，但是没记住，于是就索性整理一下吧：

**python中对文件、文件夹（文件操作函数）的操作需要涉及到os模块和shutil模块。**

得到当前工作目录，即当前Python脚本工作的目录路径: **os.getcwd()**

返回指定目录下的所有文件和目录名:**os.listdir()**

函数用来删除一个文件:**os.remove()**

删除多个目录：**os.removedirs（r“c：\python”）**

检验给出的路径是否是一个文件：**os.path.isfile()**

检验给出的路径是否是一个目录：**os.path.isdir()**

判断是否是绝对路径：**os.path.isabs()**

检验给出的路径是否真地存:**os.path.exists()**

返回一个路径的目录名和文件名:**os.path.split()**   eg os.path.split('/home/swaroop/byte/code/poem.txt') 结果：('/home/swaroop/byte/code', 'poem.txt')

分离扩展名：**os.path.splitext()**

获取路径名：**os.path.dirname()**

获取文件名：**os.path.basename()**

运行shell命令: **os.system()**

读取和设置环境变量:**os.getenv() 与os.putenv()**

给出当前平台使用的行终止符:**os.linesep**Windows使用'\r\n'，Linux使用'\n'而Mac使用'\r'

指示你正在使用的平台：**os.name**      对于Windows，它是'nt'，而对于Linux/Unix用户，它是'posix'

重命名：**os.rename（old， new）**

创建多级目录：**os.makedirs（r“c：\python\test”）**

创建单个目录：**os.mkdir（“test”）**

获取文件属性：**os.stat（file）**

修改文件权限与时间戳：**os.chmod（file）**

终止当前进程：**os.exit（）**

获取文件大小：**os.path.getsize（filename）**

**文件操作：**  
**os.mknod("test.txt")**       创建空文件  
**fp = open("test.txt",w)**   直接打开一个文件，如果文件不存在则创建文件

关于open 模式：

w     以写方式打开，  
a     以追加模式打开 (从 EOF 开始, 必要时创建新文件)  
r+     以读写模式打开  
w+     以读写模式打开 (参见 w )  
a+     以读写模式打开 (参见 a )  
rb     以二进制读模式打开  
wb     以二进制写模式打开 (参见 w )  
ab     以二进制追加模式打开 (参见 a )  
rb+    以二进制读写模式打开 (参见 r+ )  
wb+    以二进制读写模式打开 (参见 w+ )  
ab+    以二进制读写模式打开 (参见 a+ )

**fp.read([size])**                   #size为读取的长度，以byte为单位

**fp.readline([size])**               #读一行，如果定义了size，有可能返回的只是一行的一部分

**fp.readlines([size])**             #把文件每一行作为一个list的一个成员，并返回这个list。其实它的内部是通过循环调用readline()来实现的。如果提供size参数，size是表示读取内容的总长，也就是说可能只读到文件的一部分。

**fp.write(str)**                   #把str写到文件中，write()并不会在str后加上一个换行符

**fp.writelines(seq)**        #把seq的内容全部写到文件中(多行一次性写入)。这个函数也只是忠实地写入，不会在每行后面加上任何东西。

**fp.close()**                     #关闭文件。python会在一个文件不用后自动关闭文件，不过这一功能没有保证，最好还是养成自己关闭的习惯。  如果一个文件在关闭后还对其进行操作会产生ValueError

**fp.flush()**                                   #把缓冲区的内容写入硬盘

**fp.fileno()**                                  #返回一个长整型的”文件标签“

**fp.isatty()**                                  #文件是否是一个终端设备文件（unix系统中的）

**fp.tell()**                                         #返回文件操作标记的当前位置，以文件的开头为原点

**fp.next()**                                   #返回下一行，并将文件操作标记位移到下一行。把一个file用于for … in file这样的语句时，就是调用next()函数来实现遍历的。

**fp.seek(offset[,whence])**            #将文件打操作标记移到offset的位置。这个offset一般是相对于文件的开头来计算的，一般为正数。但如果提供了whence参数就不一定了，whence可以为0表示从头开始计算，1表示以当前位置为原点计算。2表示以文件末尾为原点进行计算。需要注意，如果文件以a或a+的模式打开，每次进行写操作时，文件操作标记会自动返回到文件末尾。

**fp.truncate([size])**                    #把文件裁成规定的大小，默认的是裁到当前文件操作标记的位置。如果size比文件的大小还要大，依据系统的不同可能是不改变文件，也可能是用0把文件补到相应的大小，也可能是以一些随机的内容加上去。

**目录操作：**  
**os.mkdir("file")**              创建目录  
复制文件：  
**shutil.copyfile("oldfile","newfile")**      oldfile和newfile都只能是文件  
**shutil.copy("oldfile","newfile")**         oldfile只能是文件夹，newfile可以是文件，也可以是目标目录  
复制文件夹：  
**shutil.copytree("olddir","newdir")**       olddir和newdir都只能是目录，且newdir必须不存在  
重命名文件（目录）  
**os.rename("oldname","newname")**      文件或目录都是使用这条命令  
移动文件（目录）  
**shutil.move("oldpos","newpos")**  
删除文件  
**os.remove("file")**  
删除目录  
**os.rmdir("dir")**只能删除空目录  
**shutil.rmtree("dir")**  空目录、有内容的目录都可以删  
转换目录  
**os.chdir("path")**换路径

**相关例子**

**1**将文件夹下所有图片名称加上'\_fc'

**python代码:**

# -\*- coding:utf-8 -\*-  
import re  
import os  
import time  
#str.split(string)分割字符串  
#'连接符'.join(list) 将列表组成字符串  
def change\_name(path):  
    global i  
    if not os.path.isdir(path) and not os.path.isfile(path):  
        return False  
    if os.path.isfile(path):  
        file\_path = os.path.split(path) #分割出目录与文件  
        lists = file\_path[1].split('.') #分割出文件与文件扩展名  
        file\_ext = lists[-1] #取出后缀名(列表切片操作)  
        img\_ext = ['bmp','jpeg','gif','psd','png','jpg']  
        if file\_ext in img\_ext:  
            os.rename(path,file\_path[0]+'/'+lists[0]+'\_fc.'+file\_ext)  
            i+=1 #注意这里的i是一个陷阱  
        #或者  
        #img\_ext = 'bmp|jpeg|gif|psd|png|jpg'  
        #if file\_ext in img\_ext:  
        #    print('ok---'+file\_ext)  
    elif os.path.isdir(path):  
        for x in os.listdir(path):  
            change\_name(os.path.join(path,x)) #os.path.join()在路径处理上很有用

img\_dir = 'D:\\xx\\xx\\images'  
img\_dir = img\_dir.replace('\\','/')  
start = time.time()  
i = 0  
change\_name(img\_dir)  
c = time.time() - start  
print('程序运行耗时:%0.2f'%(c))  
print('总共处理了 %s 张图片'%(i))

输出结果：

程序运行耗时:0.11  
总共处理了 109 张图片