**python之shutil模块使用方法(三分钟了解)**

[shutil](https://so.csdn.net/so/search?q=shutil&spm=1001.2101.3001.7020" \t "_blank)**模块**

**1.1简介**

shutil模块提供了一些针对文件和目录，压缩包的高级操作，主要是拷贝、移动。对于单个文件的操作，可参考os模块的用法。

**1.2 作用及作用范围**

**作用**：

1. 快速的文件复制
2. 快速的文件压缩
3. 快速的文件目录递归复制

**作用范围：**  
shutil模块是高级的文件，文件夹，压缩包的处理模块，能处理拷贝大多数文件数据，但是在Windows和 POSIX平台上，文件所有者，ACL以及ADS(供选数据流)不会被复制。

**1.3文件拷贝示例**

**1.3.1、shutil.copyfileobj() ==> 将文件的内容拷贝到另一个文件(可以指定length拷贝)**

import shutil

# shutil.copyfileobj(fsrc, fdst[, length])，将文件内容拷贝到另一个文件中

shutil.copyfileobj(open('shutil模块.py', 'r', encoding='utf-8'), open('测试文件15.py', 'w'))

# 在 python3.8版本中，写入文件无需存在,读取文件得指定编码格式，不然写入会报错

**1.3.2、shutil.copyfile(src, dst) ==> 拷贝文件**

# shutil.copyfile(src, dst)，拷贝文件

shutil.copyfile('shutil模块.py', 'f2.log') # 目标文件无需存在

**1.3.3、shutil.copymode(src, dst) ==> 仅拷贝权限，内容、组、用户均不变**

# shutil.copymode(src, dst)，仅拷贝权限。内容、组、用户均不变

shutil.copymode('f1.log', 'f2.log') # 目标文件必须存在

**1.3.4、shutil.copystat(src, dst) ==> 仅拷贝状态的信息，包括：mode bits, atime, mtime, flags**

# shutil.copystat(src, dst)，仅拷贝状态的信息，包括：mode bits, atime, mtime, flags

shutil.copystat('f2.log', 'f1.log') # 目标文件必须存在

**1.3.5、shutil.copy(src, dst) ==> 拷贝文件和权限**

# shutil.copy(src, dst)，拷贝文件和权限

shutil.copy('f1.log', 'f2.log')

**1.3.6、shutil.copy2(src, dst) ==> 拷贝文件和状态信息**

# shutil.copy2(src, dst)，拷贝文件和状态信息

shutil.copy2('f1.log', 'f2.log')

**1.3.7、shutil.copytree(src,det,symlinks=False,ignore=None)==>递归的去拷贝文件**

# shutil.ignore\_patterns(\*patterns)

# shutil.copytree(src, dst, symlinks=False, ignore=None)，递归的去拷贝文件夹

# 目标目录不能存在，注意对folder2目录父级目录要有可写权限，ignore的意思是排除

shutil.copytree('folder1', 'folder2', ignore=shutil.ignore\_patterns('\*.pyc', 'tmp\*'))

**1.4 文件、文件夹打包**

**1.4.1shutil.make\_archive()方法**

# =====================================================================

# shutil.make\_archive(base\_name, format, ...)，创建压缩包并返回文件路径，例如：zip、tar

'''

base\_name： 压缩包的文件名，也可以是压缩包的路径。只是文件名时，则保存至当前目录，否则保存至指定路径

(例子： Presley=>保存至当前路径，/User/Presley =>保存至/Users/路径下)

format：压缩包种类，“zip”, “tar”, “bztar”，“gztar”

root\_dir：要压缩的文件夹路径（默认当前目录）

owner：用户，默认当前用户

group：组，默认当前组

logger：用于记录日志，通常是logging.Logger对象

'''

#将 /data 下的文件打包放置当前程序目录

ret = shutil.make\_archive("data\_bak", 'gztar', root\_dir='/data')

#将 /data下的文件打包放置 /tmp/目录

ret = shutil.make\_archive("/tmp/data\_bak", 'gztar', root\_dir='/data')

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreBlack.png

**1.5递归移动文件和删除文件**

**1.5.1、shutil.rmtree(path[, ignore\_errors[, onerror]]) ==> 递归的去删除文件**

#shutil.rmtree(path[, ignore\_errors[, onerror]])，递归的去删除文件

shutil.rmtree('folder1')

**1.5.2、shutil.move(src, dst) ==> 递归的去移动文件**

# shutil.move(src, dst)，递归的去移动文件，它类似mv命令，其实就是重命名

shutil.move('测试文件6.py', '测试文件1.py')

**1.6 压缩和解压**

|  |
| --- |
|  |

**实例**

**# 制作压缩文件：参数1：压缩文件名称，参数2：压缩文件类型，参数3：需要压缩的目录路径  
shutil.make\_archive('dir1\_arch','zip',r'./dir1/') # 将当前目录下面的dir1文件夹压缩为dir1\_arch.zip文件保存在当前目录下面**

**1.6.1、zipfile压缩解压缩**

# shutil 对压缩包的处理是调用 ZipFile 和 TarFile 两个模块来进行的，详细：

import zipfile

# 压缩

z = zipfile.ZipFile('测试.zip', 'w')

z.write('f1.log')

z.write('f2.log')

z.close()

# 解压到 tmp目录下

z = zipfile.ZipFile('测试.zip', 'r')

z.extractall(path='./tmp')

z.close()

**1.6.2、tarfile压缩解压缩**

# 1.2 tarfile压缩解压缩

import tarfile

# 压缩

t = tarfile.open('./tmp/laxi.zip','w')

t.add('./测试文件1.py',arcname='1.py')

t.add('./测试文件2.py',arcname='2.py')

t.close()

# tmp 解压

t = tarfile.open('./tmp/laxi.zip','r')

t.extractall('./tmp')

t.close()