Python 备份文件夹结构

作者：HaujetZhao
原文链接：https://ld246.com/article/1611458803841
来源网站：链滴

```
import sys
import os
import pathlib
import shutil
import filecmp
import fnmatch
import tqdm
from pprint import pprint

r'''
作者：HaujetZhao
日期：2021 年 1 月 24 日

脚本功能：
    将 “源目录” 的文件夹、文件结构复制到 “目标文件夹”，小于指定大小的文件复制过去，大于定大小的文件，就不复制了，而是在目标文件夹中新建一个同名的空白文件。

解决的问题：

    硬盘上有许多珍贵的资料，是辛苦收集来的，它们有这些特征：

    * 它们不是太珍贵，因为在网上花一些力气可以找到
    * 它们也珍贵，因为的确有用
    * 某些时候可能需要其中一些文件
    * 其中有一些文件体积特别大，多处备份会很费空间、成本
    * 丢掉这些文件可能会有些 trouble
    * 一旦丢失，因为不记得都有哪些文件，即使网上有资源，也很难一下子收集起来

    比如：

    * 一些电影、记录片（动辄好几十 GB）
    * 为了防止和谐、方便观看而缓存的一些油管、B站的优秀视频（动辄好几百 MB）
    * 一些百科文件（动辄好几十 GB）

    它们不值得花大精力进行冗余备份，但是一旦丢掉也挺可惜。

    所以，我就需要将这些文件夹、文件名的结构备份下来，备份占用的体积小，同时，在源文件丢失，我可以清楚地知道有哪些文件丢了，再去网上下一份。

    另外，还加了一个筛选功能，比如，小于 1000kB 的文件就备份上，这些文件一般是文本文件，里的信息一般比较重要。

    比如，可以在一个文本文件中中记录下，某些视频是从哪里下载来的，链接是什么，UP 主是谁。样，文件丢失后，还可以方便地再下载一份，如果和谐了，也知道是谁的什么作品，可以去找作者联。

用法：
    在下方设定参数，再运行此 python 脚本

'''

# ================================================================
# 在这里设定参数
```

```
源根目录 = r'E:/'
目标根目录 = r'F:/仓库盘备份'

复制文件体积阈值 = 1000  # 单位是 kB，大于这个大小的文件不会被复制，而是会创建一个同名的
白文件
文件名匹配规则 = r'*'

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

def 检查路径(路径):
    if not os.path.exists(路径):
        try:
            os.makedirs(路径)
            return True
        except:
            return False
    else:
        return True

def 遍历得全部文件夹(父文件夹):
    子文件夹列表 = []
    print(f'\n正在获取所有子文件夹：{父文件夹}')
    for root, dirs, files in os.walk(父文件夹):
        子文件夹列表.append(root)
    return 子文件夹列表

def 遍历得全部文件(父文件夹):
    子文件列表 = []
    print(f'\n正在获取所有子文件：{父文件夹}')
    for root, dirs, files in os.walk(父文件夹):
        if len(files) == 0: next
        for file_ in files:
            子文件列表.append(os.path.join(root, file_))
    return 子文件列表

def 源列表转目标列表(源路径列表, 源根目录, 目标根目录):
    路径列表 = []
    for 源路径 in 源路径列表:
        目标路径 = str(pathlib.Path(目标根目录) / pathlib.Path(源路径).relative_to(源根目录))
        路径列表.append(目标路径)
    return 路径列表

def 清理废弃路径(实际路径列表, 目标路径列表):
    # 实际路径列表 表示 目前有哪些文件
    # 目标路径列表 表示 清理后要留下哪些文件
    要清理的路径集合 = set(实际路径列表) - set(目标路径列表)
    for 清理目标 in 要清理的路径集合:
        try:
            if os.path.isdir(清理目标):
                shutil.rmtree(清理目标)
            elif os.path.isfile(清理目标):
                os.remove(清理目标)
```

```
        except Exception as e:
            print(f'一个文件清理失败\n    路径：{清理目标}\n    原因：{e}')


def 得到路径Pair列表(源路径列表, 源根目录, 目标根目录):
    路径Pair列表 = []
    for 源路径 in 源路径列表:
        目标路径 = str(pathlib.Path(目标根目录) / pathlib.Path(源路径).relative_to(源根目录))
        路径Pair列表.append([源路径, 目标路径])
    return 路径Pair列表

def main():
    print('\n开始备份\n')
    global 源根目录, 目标根目录, 复制文件体积阈值, 文件名匹配规则

    if len(sys.argv) > 1:
        源根目录 = sys.argv[1]
        目标根目录 = sys.argv[2]

    if not 检查路径(源根目录): print(f'源目录不存在')
    if not 检查路径(目标根目录): print(f'目标目录不存在')

    # 得到源目录实际的文件夹、文件
    源目录列表 = 遍历得全部文件夹(源根目录)
    源文件列表 = 遍历得全部文件(源根目录)


    # 将源目录实际的文件夹、文件转为目标路径
    目标目录列表 = 源列表转目标列表(源目录列表, 源根目录, 目标根目录)
    目标文件列表 = 源列表转目标列表(源文件列表, 源根目录, 目标根目录)

    # 得到目标中实际的文件和文件夹
    目标实际目录列表 = 遍历得全部文件夹(目标根目录)
    目标实际文件列表 = 遍历得全部文件(目标根目录)

    清理废弃路径(目标实际目录列表, 目标目录列表)
    清理废弃路径(目标实际文件列表, 目标文件列表)

    目录Pair列表 = 得到路径Pair列表(源目录列表, 源根目录, 目标根目录)
    文件Pair列表 = 得到路径Pair列表(源文件列表, 源根目录, 目标根目录)

    # pprint(目录Pair列表)

    # 复制文件夹结构
    print(f'\n开始备份文件夹结构')
    for 目录Pair in tqdm.tqdm(目录Pair列表):
        检查路径(目录Pair[1])

    # 复制文件结构
    print(f'\n开始备份文件')
    for 文件Pair in tqdm.tqdm(文件Pair列表):
        源文件kB大小 = os.path.getsize(文件Pair[0]) / 1024
        if 源文件kB大小 <= 复制文件体积阈值 and fnmatch.fnmatch(文件Pair[0], 文件名匹配规则):
            try:
```

```
                if os.path.exists(文件Pair[1]):
                    if filecmp.cmp(文件Pair[0], 文件Pair[1]):
                        next
                shutil.copy(文件Pair[0], 文件Pair[1])
            except Exception as e:
                print(f'一个文件复制失败\n    源路径：{文件Pair[0]}\n    目标路径：{文件Pair[1]}\n    原
：{e}')
        else:
            if os.path.exists(文件Pair[1]):
                pass
            else:
                try:
                    f = open(文件Pair[1], 'wb')
                    f.close
                except Exception as e:
                    print(f'一个空白文件创建失败\n    源路径：{文件Pair[0]}\n    目标路径：{文件Pair[1]}\n
  原因：{e}')
    print(f'\n完成\n')

if __name__ == '__main__':
    main()
```