Python: 使用zipfile+io模块在内存中进行zip操作

1. 网上参考代码

转自原文：https://www.cnblogs.com/yaoyu126/p/3700593.html

```
#!/usr/bin/env python
#coding=utf-8
'''
  版权所有 (c) 2014 yao_yu (http://blog.csdn.net/yao_yu_126)
  本代码采用MIT许可
  使用io.BytesIO()在内存中压缩,解压缩zip文件
     2014-04-30       yaoyu 创建
'''

import zipfile
import os
import io
from base64     import standard_b64decode
try:
    from .yy_file       import *
except:
    from yy_file        import *

__all__ = ['yy_zip_files2buffer', 'yy_zip_files', 'yy_unzip_buffer2files',
'yy_unzip_base64zipfile']

def yy_zip_files2buffer(files):
    '''文件压缩到内存中'''
    buffer = io.BytesIO()
    zfile = zipfile.ZipFile(buffer, 'w', zipfile.ZIP_DEFLATED,
allowZip64=False)
    for i in files:
        if os.path.isfile(i):
            zfile.write(i, os.path.basename(i))
    zfile.close()
    buffer.seek(0)
    return buffer

def yy_zip_files(zipfilename, files):
    '''压缩文件列表到zip文件'''
    yy_file_write_bytes(zipfilename, yy_zip_files2buffer(files).read())

def yy_unzip_buffer2files(buffer):
    '''从内存压缩文件中读取文件信息'''
    zfile = zipfile.ZipFile(buffer, 'r')
    files = []
    for i in zfile.filelist:
```

```
        files.append((i.filename, zfile.read(i.filename)))
    return files

def yy_unzip_base64zipfile(szFile):
    '''从base64压缩的zip文件中读取第一个文件二进制内容'''
    bytescontent = yy_file_read_bytes(szFile)
    for (find, replace) in ((b'&#x0D;', b'\n'),
                            (b'&#x0A;', b'\r')):
        if find in bytescontent:
            bytescontent = bytescontent.replace(find, replace)
    filebytes = standard_b64decode(bytescontent)
    fileinfos = yy_unzip_buffer2files(io.BytesIO(filebytes))
    if fileinfos:
        return fileinfos[0][1]

if __name__ == '__main__':
    sourcedir = '/Volumes/Data/Document/百度云同步
盘/Develop/Python/Calculator'
    files = (os.path.join(sourcedir, i) for i in os.listdir(sourcedir))
    yy_zip_files(os.path.join(sourcedir, 'dist', 'r.zip'), files)
    #print(yy_unzip_buffer2files(yy_zip_files2buffer(files)))


    filename = '/Volumes/Data/Download/aaa'
    filebytes = yy_unzip_base64zipfile(filename)
    yy_file_write_bytes(filename + '.xls', filebytes)
```

2. 读zip文件到内存，然后解压到内存

```
#!/usr/bin/python3

import zipfile
import io


fh = open('ab.zip', 'rb')
fbuf = fh.read()    #读取zip二进制文件内容到buffer□注意buffer溢出。
fh.close()

fio = io.BytesIO()    #创建一个文件io

fio.write(fbuf)      #将zip二进制内容写入文件io



# zfile = zipfile.ZipFile('ab.zip', 'r')
zfile = zipfile.ZipFile(fio, 'r')    #直接从文件io处操作，而不是指定硬盘上的文件。
files = []
for i in zfile.filelist:
```

```
    files.append((i.filename, zfile.read(i.filename)))

for tmp in files:
    print(f'{tmp}')    # output: ('a.txt', b' a\n')
    fname = tmp[0]
    fh = open(fname)
    _buff = fh.read()
    print(f'from file {fname}, read all = {_buff}')
```

3. 读zip文件然后AES加密写到文件，读加密后的文件解密到内存，然后在内存解压zip文件

```
#!/usr/bin/python3

import zipfile
import io

#coding: utf8
from Crypto import Random

import sys
from Crypto.Cipher import AES
from binascii import b2a_hex, a2b_hex

class prpcrypt():
    def __init__(self, key):
        self.key = key
        self.mode = AES.MODE_CBC

    #加密函数，如果text不是16的倍数【加密文本text必须为16的倍数！】，那就补足为16的倍数
    def encrypt(self, text):
        cryptor = AES.new(self.key, self.mode, self.key)
        #这里密钥key 长度必须为16（AES-128）、24（AES-192）或32（AES-256）Bytes 长度.目前AES-128足够用
        length = 16
        count = len(text)
        # print (f'count = {count}')

        if(count % length != 0) :
                add = length - (count % length)
        else:
            add = 0

        text = text + (b'\0' * add)
        self.ciphertext = cryptor.encrypt(text)
        #因为AES加密时候得到的字符串不一定是ascii字符集的，输出到终端或者保存时候可能存在问题
        #所以这里统一把加密后的字符串转化为16进制字符串
        return b2a_hex(self.ciphertext)
```

```
    #解密后，去掉补足的空格用strip() 去掉
    def decrypt(self, text):
        cryptor = AES.new(self.key, self.mode, self.key)
        plain_text = cryptor.decrypt(a2b_hex(text))
        return plain_text.rstrip(b'\0')

# if __name__ == '__main__':
#     pc = prpcrypt('keyskeyskeyskeys')       #初始化密钥
#     e = pc.encrypt("0123456789ABCDEF")
#     d = pc.decrypt(e)
#     print (f'e,d = {e}, {d}')
#     e = pc.encrypt("00000000000000000000000000")
#     d = pc.decrypt(e)
#     # print (e, d)
#     print (f'e,d = {e}, {d}')
#
# exit(0)


pc = prpcrypt('keyskeyskeyskeys')       #初始化密钥

################ read file, then encrypt #################
fh = open('ab.zip', 'rb')
fbuf = fh.read()
fh.close()

fbuf_str = b2a_hex(fbuf)
print(f'fbuf_str = {fbuf_str}')

e = pc.encrypt(fbuf_str)
print(f'e = {e}')

fh = open('_ab.e', 'wb')
fh.write(a2b_hex(e))
fh.close()



################ read the entrypted file, decrypt ################
fh = open('_ab.e', 'rb')
e = b2a_hex(fh.read())
fh.close()

d = pc.decrypt(e)
print(f'd = {d}')



fio = io.BytesIO()
# fio.write(fbuf)
```

```
fio.write(a2b_hex(d))



# zfile = zipfile.ZipFile('ab.zip', 'r')
zfile = zipfile.ZipFile(fio, 'r')
files = []
for i in zfile.filelist:
    files.append((i.filename, zfile.read(i.filename)))

for tmp in files:
    print(f'{tmp}')
    fname = tmp[0]
    fh = open(fname)
    _buff = fh.read()
    print(f'from file {fname}, read all = {_buff}')
```