

別再問我Python打包成exe了！（終極版）

印象Python 今天

以下文章來源於凹凸數據



凹凸數據

一個不務正業的數據🐱！爬蟲、數據分析、可視化、方法論，一條龍服務！業務範圍：Python、SQL、Excel、Tableau……



文| 酷頭

來源：印象python [ID: python_logic]

嗨！大家好，我是酷頭

歡迎來到學習python的寶藏基地~~~



長按下方二維碼可以添加我為好友哦



今天就給大家全面總結一下：Python如何打包成exe，以及如何打得足夠小。

標準打包

目前比較常見的打包exe方法都是通過 如果對這塊已經很熟悉的小伙伴，可以直接下滑到本文下半部分。 [Pyinstaller](#)

為什麼要打包？

眾所周知，Python腳本不能在沒有安裝Python的機器上運行。

那我們如果寫了一個數據分析/自動化辦公的小腳本，想分享給同事小姐姐使用，可她電腦又沒有裝Python。

這個時候如果將腳本打包成exe文件，微信發送給她，即使她的電腦上沒有安裝Python解釋器，這個exe程序也能在上面運行。豈不美哉？



（當然，想通過幫安裝Python跟小姐姐建立感情的話，就當我沒說）

安裝Pyinstaller

首先我們要先安裝Pyinstaller，直接在cmd使用pip命令

```
pip install pyinstaller
```

如果網速太慢可以切換國內源來加速，本文其他部分直接使用國內源，不再重複註釋。

```

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple
tuna.tsinghua.edu.cn/simple
singhua.edu.cn/packages/79/87/8bb36bd4ebae147612c73d1
nd64.whl (8.7 MB)
| 3.4 MB 1.7 MB/s eta 0:00:04_

```

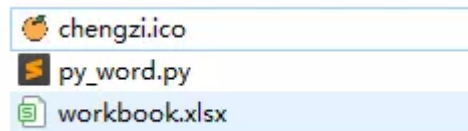
```

pip install -i https://pypi.douban.com/simple/ pyinstaller #豆瓣源
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pyinstaller #清华源

```

Pyinstaller打包步驟

这里我们拿之前《[Python自动化办公 | 同事要我帮忙补写178份Word日报！](#)》这篇的python代码，作为案例来演示。将其中脚本 `py_word.py`，待处理的表格文件 `workbook.xlsx`，以及准备好的软件图标图片 `chengzi.ico` 放在了我电脑的 `F:\py_word` 目录下（如果大家感兴趣的话，可以在文末下载获取）



1、cmd切换到我们刚刚放文件的目录

```

管理员: Anaconda Powershell Prompt (anaconda3)
(base) PS C:\Users\Administrator> cd F:\py_word
(base) PS F:\py_word>

```

2、执行命令 `Pyinstaller -F -w -i chengzi.ico py_word.py`，执行过程特别漫长，就没有录制动图。

执行完毕会发现当前目录多了几个文件夹，打开其中名为dist的文件夹。



已经生成了一个名为 `py_word` 的exe应用程序，并且图标也是我们设置的橙子图案，看来已经成功一半了。

要不运行一下，试试能否正常生成word日报？

名称	修改日期	类型	大小
 __pycache__	2021/1/12/周二 ...	文件夹	
 build	2021/1/12/周二 ...	文件夹	
 dist	2021/1/12/周二 ...	文件夹	
 chengzi.ico	2021/1/9/周六 2...	看图王 ICO 图片...	1 KB
 py_word.exe	2021/1/12/周二 ...	应用程序	347,818 KB
 py_word.py	2021/1/10/周日 ...	PY 文件	9 KB
 py_word.spec	2021/1/12/周二 ...	SPEC 文件	1 KB
 workbook.xlsx	2020/9/20/周日 ...	XLSX 工作表	13 KB

打包exe后执行

成功运行，可惜就是文件有点大（339M）



Pyinstaller参数详解

说回刚刚执行的命令

```
Pyinstaller -F -w -i chengzi.ico py_word.py
```

给大家解释一下其中Pyinstaller的参数，其中 `-F` 参数代表制作独立的可执行程序。

`-w` 是指程序启动的时候不会打开命令行。如果不加`-w`的参数，就会有黑洞洞的控制台窗口出来。比如在刚才的脚本里我加一行 `print('Hello World!')`，那么就不要放`-w`参数了，不然运行会报错，毕竟 `Hello World!` 需要在命令行里打印出来。此外，`-w` 参数在GUI界面时非常有用。

最后的 `-i chengzi.ico` 就是指设置自己的图标图案，因为默认打包图片是下图这样的。这个参数也可以写成 `--icon=chengzi.ico`



最后稍微总结一下：

```
Pyinstaller -F py_word.py 打包exe
```

```
Pyinstaller -F -w py_word.py 不帶控制台的打包
```

```
Pyinstaller -F -w -i chengzi.ico py_word.py 打包指定exe图标打包
```

以上三个是比较常用的参数，其他参数详见下表

-F, -onefile	产生一个文件用于部署 (参见XXXXX).
-D, -onedir	产生一个目录用于部署 (默认)
-K, -tk	在部署时包含 TCL/TK
-a, -ascii	不包含编码. 在支持Unicode的python版本上默认包含所有的编码.
-d, -debug	产生debug版本的可执行文件
-w, -windowed, -noconsole	使用Windows子系统执行. 当程序启动的时候不会打开命令行(只对Windows有效)
-c, -nowindowed, -console	使用控制台子系统执行(默认)(只对Windows有效)
-s, -strip	可执行文件和共享库将run through strip. 注意Cygwin的strip往往使普通的win32 DLL无法使用.
-X, -upx	如果有UPX安装(执行Configure.py时检测), 会压缩执行文件(Windows系统中的DLL也会)(参见note)
-o DIR, -out=DIR	指定spec文件的生成目录, 如果没有指定, 而且当前目录是PyInstaller的根目录, 会自动创建一个用于输出(spec和生成的可执行文件)的目录. 如果没有指定, 而当前目录不是PyInstaller的根目录, 则会输出到当前的目录下.
-p DIR, -path=DIR	设置导入路径(和使用PYTHONPATH效果相似). 可以用路径分割符(Windows使用分号, Linux使用冒号)分割, 指定多个目录. 也可以使用多个-p参数来设置多个导入路径
-icon=<FILE.ICO>	将file.ico添加为可执行文件的资源(只对Windows系统有效)
-icon=<FILE.EXE,N>	将file.exe的第n个图标添加为可执行文件的资源(只对Windows系统有效)
-v FILE, - version=FILE	将verfile作为可执行文件的版本资源(只对Windows系统有效)
-n NAME, - name=NAME	可选的项目(产生的spec的)名字. 如果省略, 第一个脚本的主文件名将作为spec的名字

Pyinstaller参数大全

自己做的软件都喜欢放上自己的图标，不过哪来那么多ico图片呢？

一个是找专门的ico图片网站，不过都很小众，图片库也很小。

另一个是可以自己生成，这里就给大家分享一个网站，可以把其他格式图片转成ico格式：<https://app.xunjiepdf.com/img2icon/>



ico图片格式转换



好了，小伙伴们

来到了最激动人心的时刻，刚刚生成的exe实在太大了，300多M的软件程序想用微信传一下都费劲。

我也试过很多方法，比如：修改spec文件自定义打包、pipenv 虚拟环境、使用开源的upx压缩等等，但是往往要么过程比较麻烦，要么成功率不高（压缩成不成功全看脸）。

而我要分享的，是自己一直在用的，最简单且成功率极高的方法——conda创建虚拟环境。



Python打包为什么大？

在压缩打包之前，先简单说一下为什么Python打包过大？

Python打包exe，不但体积大而且运行奇慢。解释型语言大都是这个样子，只不过Python尤其突出。要解决大而慢，只能用编译型语言，如C，C++，甚至VB都好很多，体积最小的是汇编。^[1]

此外，还有知乎大佬说是因为“Anaconda里内置了很多库，打包的时候打包了很多不必要的模块进去，要用纯净的Python来打包。”

所以我们可以模拟一个新环境，其中只安装我们本次打包所必要的工具包即可。

那最适合的就是——虚拟环境了！

虚拟环境

Python创建虚拟环境的方法有很多，而我是个Anaconda忠实用户，如果你跟我一样，那就简单了。（大家也可以使用Virtualenv、Pipenv来设置虚拟环境，善用搜索，方法大同小异）

先记几个命令，很简单

```
conda create -n 虚拟环境名字 python==3.6 #创建虚拟环境
```

```
conda activate 虚拟环境名字 #激活虚拟环境
```

```
conda deactivate #退出虚拟环境
```

从开始菜单运行“Anaconda Prompt”，出现的界面输入创建虚拟环境的指令。成功创建了一个名字为 `aotu`，且基于python版本3.6的虚拟环境。

```
管理员: Anaconda Powershell Prompt (anaconda3)
(base) PS C:\Users\Administrator> conda create -n aotu python=3.6
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

在创建过程中需要回复（y/n），Yes，再激活虚拟环境

```
(base) PS C:\Users\Administrator> conda activate aotu
(aotu) PS C:\Users\Administrator> _
```

conda安装的虚拟环境，会把虚拟环境的目录生成在anaconda安装目录下的env目录下。

本地磁盘 (C:) > Users > Administrator > anaconda3 > envs >

名称

修改日期

类型



2021/1/12/周二 ...	文件夹
2020/10/8/周四 ...	文件夹
2020/10/8/周四 ...	文件夹
2021/1/9/周六 1...	文件夹

当然我们也可以在刚刚的窗口，使用命令 `conda info --envs`，来查看conda环境下所有的虚拟环境

```
(aotu) PS C:\Users\Administrator> conda info --envs
# conda environments:
#
base                  C:\Users\Administrator\anaconda3
aotu                  * C:\Users\Administrator\anaconda3\envs\aotu
env5                  C:\Users\Administrator\anaconda3\envs\env5
env66                 C:\Users\Administrator\anaconda3\envs\env66
gui                   C:\Users\Administrator\anaconda3\envs\gui
```

安装所需的库

上面我们已经创建并激活了名为 `aotu` 虚拟环境，输入 `conda list` 可以查看当前虚拟环境里已经安装的库。

```
(aotu) PS C:\Users\Administrator> conda list
# packages in environment at C:\Users\Administrator\anaconda3\envs\aotu:
#
```

# Name	Version	Build	Channel
certifi	2020.12.5	py36haa95532_0	
pip	20.3.3	py36haa95532_0	
python	3.6.12	h5500b2f_2	
setuptools	51.1.2	py36haa95532_3	
sqlite	3.33.0	h2a8f88b_0	
vc	14.2	h21ff451_1	
vs2015_runtime	14.27.29016	h5e58377_2	
wheel	0.36.2	pyhd3eb1b0_0	
wincertstore	0.2	py36h7fe50ca_0	
zlib	1.2.11	h62dcd97_4	

我们打开所要打包的Python脚本，对比上图，发现 `pandas`，`docx` 这两个库还需要额外安装。当然，也不能少了打包必不可少的 `pyinstaller` 库。

```
import pandas as pd
from docx import Document
from docx.shared import Pt
from docx.shared import Inches
from docx.xml.ns import qn
from docx.enum.text import WD_PARAGRAPH_ALIGNMENT
from docx.enum.section import WD_ORIENTATION
```

'''----- 读取excel表数据整理后写入word （纯文本） -----'''

```
def wu_to_word(filepath):
    df = pd.read_excel(filepath, sheet_name="无")
    date_list = list(df['日期'])
    for d in date_list:
        filename = wordname+str(d)+"_docx" # 输出的word文件名
```

```

filename = "notname" + str(d[4:6]) + ".docx"
title = "(" + str(d)[:4] + "." + str(d)[4:6] + "." + str(d)[6:8] + ")" # 副标题日期XXXX.XX.XX
word = str(d)[:4] + "年" + str(d)[4:6] + "月" + str(d)[6:8] + "日" # 开头、落款日期XXXX年XX月XX日
wu_doc(title, word, filename)
print(f"文件: {filename},{title},{word} 已保存")

```

待打包脚本

安装库的过程不再赘述

```

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pandas

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple python-docx

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple pyinstaller

```

安装后再看

```

(aotu) PS C:\Users\Administrator> conda list
# packages in environment at C:\Users\Administrator\anaconda3\envs\autu:
#
# Name                      Version           Build    Channel
altgraph                    0.17              pypi_0   pypi
certifi                     2020.12.5         py36haa95532_0
future                      0.18.2            pypi_0   pypi
lxml                        4.6.2             pypi_0   pypi
numpy                       1.19.5            pypi_0   pypi
pandas                      1.1.5             pypi_0   pypi
pefile                      2019.4.18         pypi_0   pypi

```

```

pip 20.3.3 py36haa95532_0
pyinstaller 4.1 pypi_0 pypi
pyinstaller-hooks-contrib 2020.11 pypi_0 pypi
python 3.6.12 h5500b2f_2
python-dateutil 2.8.1 pypi_0 pypi
python-docx 0.8.10 pypi_0 pypi
pytz 2020.5 pypi_0 pypi
pywin32-ctypes 0.2.0 pypi_0 pypi
setuptools 51.1.2 py36haa95532_3
six 1.15.0 pypi_0 pypi
sqlite 3.33.0 h2a8f88b_0
vc 14.2 h21ff451_1
vs2015_runtime 14.27.29016 h5e58377_2
wheel 0.36.2 pyhd3eb1b0_0
wincertstore 0.2 py36h7fe50ca_0
zlib 1.2.11 h62dcd97_4

```

已成功安装，还有一些因为这几个库所附带安装的，就不管他们了。

Pyinstaller打包步骤

这里就不再重复了，只换个苹果图标试试

```
Pyinstaller -F -w -i apple.ico py_word.py
```

```

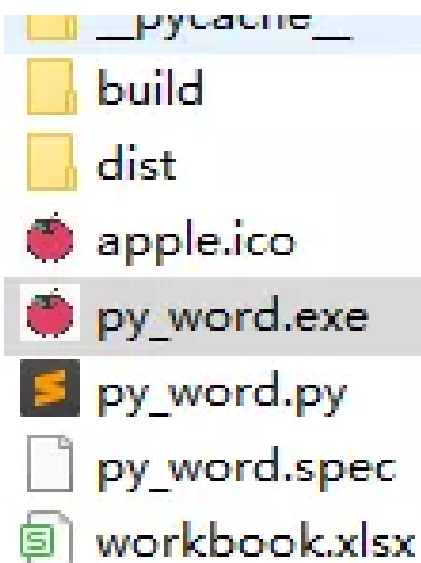
(aotu) PS C:\Users\Administrator> cd F:\py_word
(aotu) PS F:\py word> Pyinstaller -F -w -i pingguo.ico py word.py

```



```
105 INFO: PyInstaller: 4.1
105 INFO: Python: 3.6.12 (conda)
106 INFO: Platform: Windows-10-10.0.14393-SP0
107 INFO: wrote F:\py_word\py_word.spec
109 INFO: UPX is not available.
112 INFO: Extending PYTHONPATH with paths
['F:\\py_word', 'F:\\py_word']
126 INFO: checking Analysis
```

生成



- __pycache__
- build
- dist
- apple.ico
- py_word.exe
- py_word.py
- py_word.spec
- workbook.xlsx

大小: 29.8 MB (31,319,558 字节)

占用空间: 29.8 MB (31,322,112 字节)

创建时间: 2021年1月12日, 今天, 3 分钟之前

修改时间: 2021年1月12日, 今天, 3 分钟之前

成功压缩到29.8M，如果不导入pandas这位大神，应该就可以10多M了

运行一下，毫无问题

 __pycache__	2021/1/12/周二 ...	文件夹	
 build	2021/1/12/周二 ...	文件夹	
 dist	2021/1/12/周二 ...	文件夹	
 apple.ico	2021/1/12/周二 ...	看图王 ICO 图片...	13 KB
 py_word.exe	2021/1/12/周二 ...	应用程序	30,586 KB
 py_word.py	2021/1/12/周二 ...	PY 文件	9 KB
 py_word.spec	2021/1/12/周二 ...	SPEC 文件	1 KB
 workbook.xlsx	2020/9/20/周日 ...	XLSX 工作表	13 KB

最后再简单总结一下虚拟环境+打包的全过程（只三步）：

```
#创建虚拟环境
conda create -n aotu python=3.6

#激活虚拟环境
conda activate aotu

#Pyinstaller打包
Pyinstaller -F -w -i apple.ico py_word.py
```

总结一些小坑

- 1、说起来还是有点玄学，上文中一模一样的过程我在两个电脑都执行过一遍。在其中一个上显示缺少 `xlrd` 这个库，安装后成功打包，也是一样的大小。小伙伴们可以在文末获取文件，也试试。
- 2、在安装库是要注意一些库名，比如docx这个库不要 `pip install docx`，而是需要

```
pip install python-docx
```

还有一些库可能因为版本不同导致不能使用，多碰碰壁就好了。

- 3、为了防止打包时候有些库没安装好，可以先在虚拟环境中执行一下Python脚本。运行无误的话再打包，比较保险。

```
(aotu) PS F:\py_word> python F:\py_word\py_word.py
文件: XX公司业务数据表 (日报20191120) .docx, (2019. 11. 20), 2019年11月20日 已保存
文件: XX公司业务数据表 (日报20191121) .docx, (2019. 11. 21), 2019年11月21日 已保存
文件: XX公司业务数据表 (日报20191122) .docx, (2019. 11. 22), 2019年11月22日 已保存
文件: XX公司业务数据表 (日报20191126) .docx, (2019. 11. 26), 2019年11月26日 已保存
文件: XX公司业务数据表 (日报20191127) .docx, (2019. 11. 27), 2019年11月27日 已保存
文件: XX公司业务数据表 (日报20191204) .docx, (2019. 12. 04), 2019年12月04日 已保存
文件: XX公司业务数据表 (日报20191209) .docx, (2019. 12. 09), 2019年12月09日 已保存
```

```
文件: XX公司业务数据表 (日报20191212) .docx, (2019. 12. 12), 2019年12月12日 已保存
文件: XX公司业务数据表 (日报20191217) .docx, (2019. 12. 17), 2019年12月17日 已保存
文件: XX公司业务数据表 (日报20191219) .docx, (2019. 12. 19), 2019年12月19日 已保存
文件: XX公司业务数据表 (日报20191220) .docx, (2019. 12. 20), 2019年12月20日 已保存
文件: XX公司业务数据表 (日报20191223) .docx, (2019. 12. 23), 2019年12月23日 已保存
文件: XX公司业务数据表 (日报20191224) .docx, (2019. 12. 24), 2019年12月24日 已保存
文件: XX公司业务数据表 (日报20191225) .docx, (2019. 12. 25), 2019年12月25日 已保存
文件: XX公司业务数据表 (日报20191226) .docx, (2019. 12. 26), 2019年12月26日 已保存
文件: XX公司业务数据表 (日报20200101) .docx, (2020. 01. 01), 2020年01月01日 已保存
文件: XX公司业务数据表 (日报20200103) .docx, (2020. 01. 03), 2020年01月03日 已保存
文件: XX公司业务数据表 (日报20200107) .docx, (2020. 01. 07), 2020年01月07日 已保存
文件: XX公司业务数据表 (日报20200108) .docx, (2020. 01. 08), 2020年01月08日 已保存
文件: XX公司业务数据表 (日报20191119) .docx 已保存
文件: XX公司业务数据表 (日报20191125) .docx 已保存
```

4、最后，如果想要删除虚拟环境的话，可执行下方命令

```
conda remove -n aotu--all
```

5、打包为exe的话，版本尽量选择python3.6+32位版本，因为win64位系统向下兼容32位程序，但是如果不考虑32位系统的话无所谓，直接python64位版本直接打包就可以，只是只能在win64位系统上跑。^[2]

下载链接

如果大家想测试Python打包，手头又没有合适的文件，可以在「[凹凸数据](#)」后台回复“**打包**”即可获得以下文件：



参考资料

- [1] 弗拉基米尔: <https://www.zhihu.com/question/281858271/answer/613147412>
- [2] 《别再问我怎么Python打包成exe了！》: https://mp.weixin.qq.com/s/zilDeFunWLG0mBS_x0vNnA

【印象python】交流群正式开放啦！
欢迎扫码添加老邓好友，通过后拉你入群。



[阅读原文](#)

喜欢此内容的人还喜欢

肯德基竟无鸡可炸？Scrapy爬虫知乎评论，看看网友们怎么说？

印象Python



办实事 | 海南海事“五个强化”全力护航“十一”黄金周水路出行安全

中国海事



00后都已经要结婚了？？

摩登人類

