Monitor your internet with python

Joseph · 9 mins read · Discuss on Reddit

python data-visualization

```
import speedtest
import datetime
import csv

s = speedtest.Speedtest()

with open('test.csv', mode='w') as speedcsv:
    csv_writer = csv.DictWriter(speedcsv, fieldnames=['time', 'downspeed', 'upspeed'])
    csv_writer.writeheader()
    while True:
        time = datetime.datetime.now().strftime("%H:%M:%S")
        downspeed = round((round(s.download()) / 1048576), 2)
        upspeed = round((round(s.upload()) / 1048576), 2)
        csv_writer.writerow({
            'time': time,
            'downspeed': downspeed,
            "upspeed": upspeed
        })
```

I live in Germany and Germany is known for its diversity. The people are diverse, the culture is diverse and the quality of internet connections are equally as diverse.

Having spoken to my internet provider who insisted I had no problems with my internet connection I decided to write a program to monitor my upload and download speed.

Getting started

We will be using two cool packages in python namely `speedtest` and `matplotlib`. `speedtest` will provide us with the information about our internet connection and `matplotlib` will allow us to generate graphs based on this information.

Whether you're in a virtual environment or not using one you will have to install speedtest and matplotlib.

```
pip install matplotlib
pip install speedtest-cli
```

We will also be using a few built in python packages `datetime` and `csv`. CSV (or comma-separated values) are a quick way of storing data. We will be using this to store the information about our internet speeds and then using matplotlib to make this information visual.

Gathering data

Create a python file called `monitor.py`

```
touch monitor.py
```

and let's put the following code on the inside

```
import speedtest
s = speedtest.Speedtest()
while True:
    print(s.download(), s.upload())
```

and we should be greeted by the following output before hitting `ctrl + C` to get out of this infinite loop.

```
75020309.22943133 24381170.373616524
105192450.00822952 40653433.153288215
```

So by creating a new instance of `speedtest` as `s` and testing the upload and download speed we are given the upload and download speed in bits per second. To convert this to megabits per second `(Mb/s)` we can do the following to include the time of the test too:

```
import speedtest
import datetime
import time
s = speedtest.Speedtest()

while True:
    time_now = datetime.datetime.now().strftime("%H:%M:%S")
    downspeed = round((round(s.download()) / 1048576), 2)
    upspeed = round((round(s.upload()) / 1048576), 2)
    print(f"time: {time_now}, downspeed: {downspeed} Mb/s, upspeed: {upspeed
    # 60 seconds sleep
    time.sleep(60)
```

Note: Added a delay due to feedback received by an awesome Reddit user.

which yields to:

```
time: 12:44:15, downspeed: 95.04 Mb/s, upspeed: 32.85 Mb/s
time: 12:44:35, downspeed: 99.46 Mb/s, upspeed: 38.76 Mb/s
time: 12:44:56, downspeed: 100.59 Mb/s, upspeed: 38.94 Mb/s
```

Now we will move on to recording this in a CSV file. CSVs are large text files which values separated by commas. Here is an example of one of mine:

```
time,downspeed,upspeed
12:17:01,100.05,38.28
12:17:21,100.53,37.85
12:17:42,74.87,25.92
```

We start with a header row then have the information split up. We can think about this like an excel sheet:

	time	downspeed	upspeed
	12:17:01	100.05	38.28
	12:17:21	100.53	37.85
	12:17:42	74.87	25.92

In order to record into a csv file in python we need to import the CSV package and ‘open’ a CSV file (if one doesn’t exist it will create one).

```
import speedtest
import datetime
import csv
import time

s = speedtest.Speedtest()

with open('test.csv', mode='w') as speedcsv:
    csv_writer = csv.DictWriter(speedcsv, fieldnames=['time', 'downspeed', '
    csv_writer.writeheader()
    while True:
        time_now = datetime.datetime.now().strftime("%H:%M:%S")
        downspeed = round((round(s.download()) / 1048576), 2)
        upspeed = round((round(s.upload()) / 1048576), 2)
        csv_writer.writerow({
            'time': time_now,
            'downspeed': downspeed,
            "upspeed": upspeed
        })
        # 60 seconds sleep
        time.sleep(60)
```

So while you let this code run for 4-5 minutes we can discuss what is going on. Line 7 `with open` essentiallly creates a csv file with the name `test.csv` with the headers `name`, `downspeed` and `upspeed` and writes them into the csv. Then the loop begins and every time a test is performed by speedtest it writes a new row

into the csv with the time, download speed and upload speed we specified before. So let's go and look at that now.

```
time,downspeed,upspeed
12:51:16,99.29,38.66
12:51:37,100.67,38.79
12:51:57,99.7,38.79
12:52:17,92.89,31.99
12:52:38,99.4,38.96
```

Cool now we are gathering information and storing it in a csv. You could do all sorts of clever stuff with the filename like name it today's date using `datetime` but for now I will keep it as test.

You could also set up a check to see if a certain amount of time has passed and to stop the application after that but I will leave that up to your own creativity.

Visualising the data

Let's make another python file to generate the graph of our internet connection. This is where we will use matplotlib.

```
touch make_graph.py
```

```
import matplotlib.pyplot as plt
import csv
import matplotlib.ticker as ticker
times = []
download = []
upload = []
```





The easiest way to generate a graph is by using an array. In order to populate our arrays we will have to iterate through our new csv file

```
with open('test.csv', 'r') as csvfile:
    plots = csv.reader(csvfile, delimiter=',')
    next(csvfile)
    for row in plots:
        times.append(str(row[0]))
        download.append(float(row[1]))
        upload.append(float(row[2]))

print(times, "\n", download, "\n", upload)
```

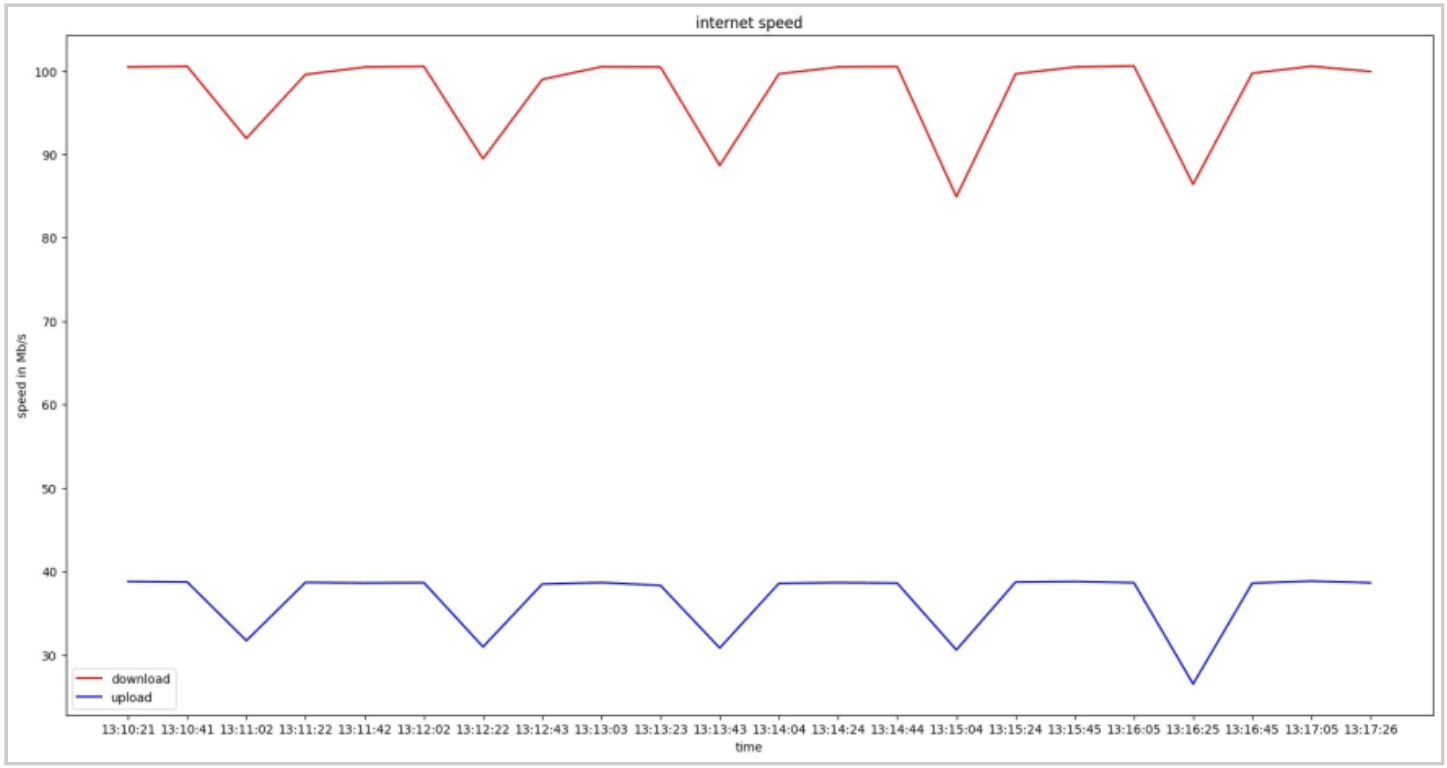
```
['12:51:16', '12:51:37', '12:51:57', '12:52:17', '12:52:38']
    [99.29, 100.67, 99.7, 92.89, 99.4]
    [38.66, 38.79, 38.79, 31.99, 38.96]
```

So now we are parsing our data! The `next(csvfile)` essentially skips the row of headers (that were for our benefit only, not python's). Now we come on to using `matplotlib` which I am by no standards an expert on. Their documentation is *extensive*.

```
plt.figure(30, 30)
plt.plot(times, download, label='download', color='r')
plt.plot(times, upload, label='upload', color='b')
plt.xlabel('time')
plt.ylabel('speed in Mb/s')
plt.title("internet speed")
plt.legend()
plt.savefig('test_graph.jpg', bbox_inches='tight')
```

This code tells matplotlib or `plt` object to create a figure with the lines labelled 'download' and 'upload' to be plotted against their respective arrays. So for

every item in the download and time array, a mark will be made on the graph against it. Finally the x and y-axis get labels and the graph gets saved.



Potential improvements

This is a short form of a project I recently finished which contains automated uploads to a cloud platform, schedules itself daily does some annotations of the lowest download speed. I initially made it to complain to my internet provider however I know that it will be like showing a graph to a brick wall.

Here is part II of the post where we send a tweet to the ISP telling them about our low internet speed.

Tweeting about your low internet speed and tagging your ISP

#python #imageprocessing

October 25, 2020 2 mins read

If you have any questions about this tutorial please don't hesitate to reach out to me.

@jsphwllng

About Author

Joseph

Developer fluent in Rails, MERN and Python. Now learning Go and looking for a position in Berlin.

You May Also Enjoy

Please share your Feedback:

Did you enjoy reading or think it can be improved? Don’t forget to leave your thoughts in the comments section below! If you liked this article, please share it with your friends, and read a few more!





Load Comments...

Subscribe

Email Address *

Subscribe

Read old issues

Python

Django

Newsletter

Data Science

Best: A GitHub Pages Theme

Convert Python Script to .exe File

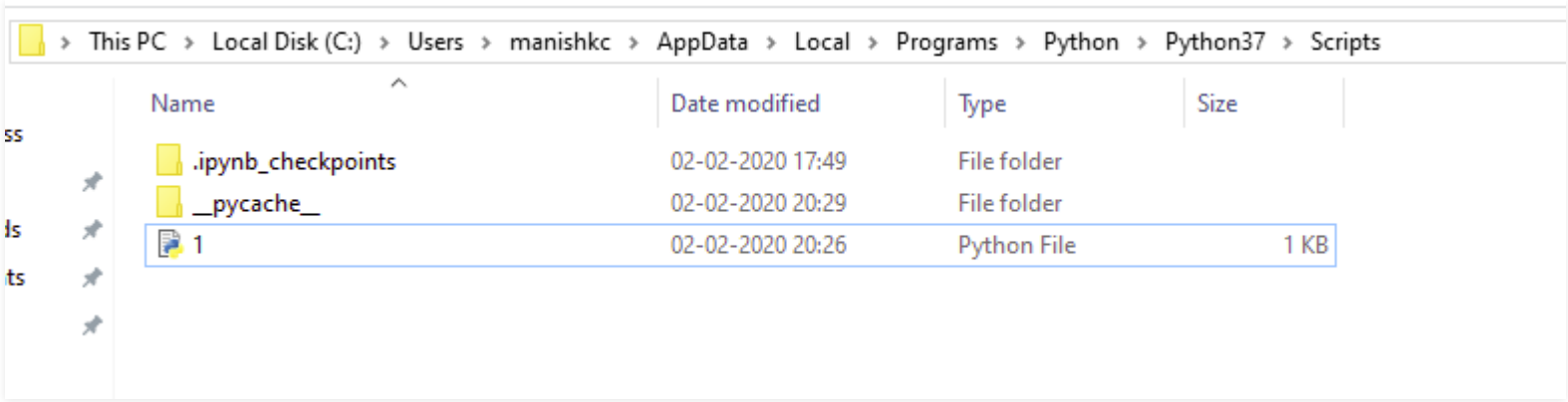
Difficulty Level : Hard • Last Updated : 01 Feb, 2021

We create lots of Python programs per day and want to share them with the world. It is not that you share that Python program with everyone, and they will run this script in some IDLE shell. But you want everyone to run your Python script without the installation of Python. So for this work, you can convert the .py file to .exe file. In this article, you will learn how you can convert .py file to .exe file. Follow the below steps for the same.

Step 1:
Install the library pyinstaller.
Type below command in the command prompt.

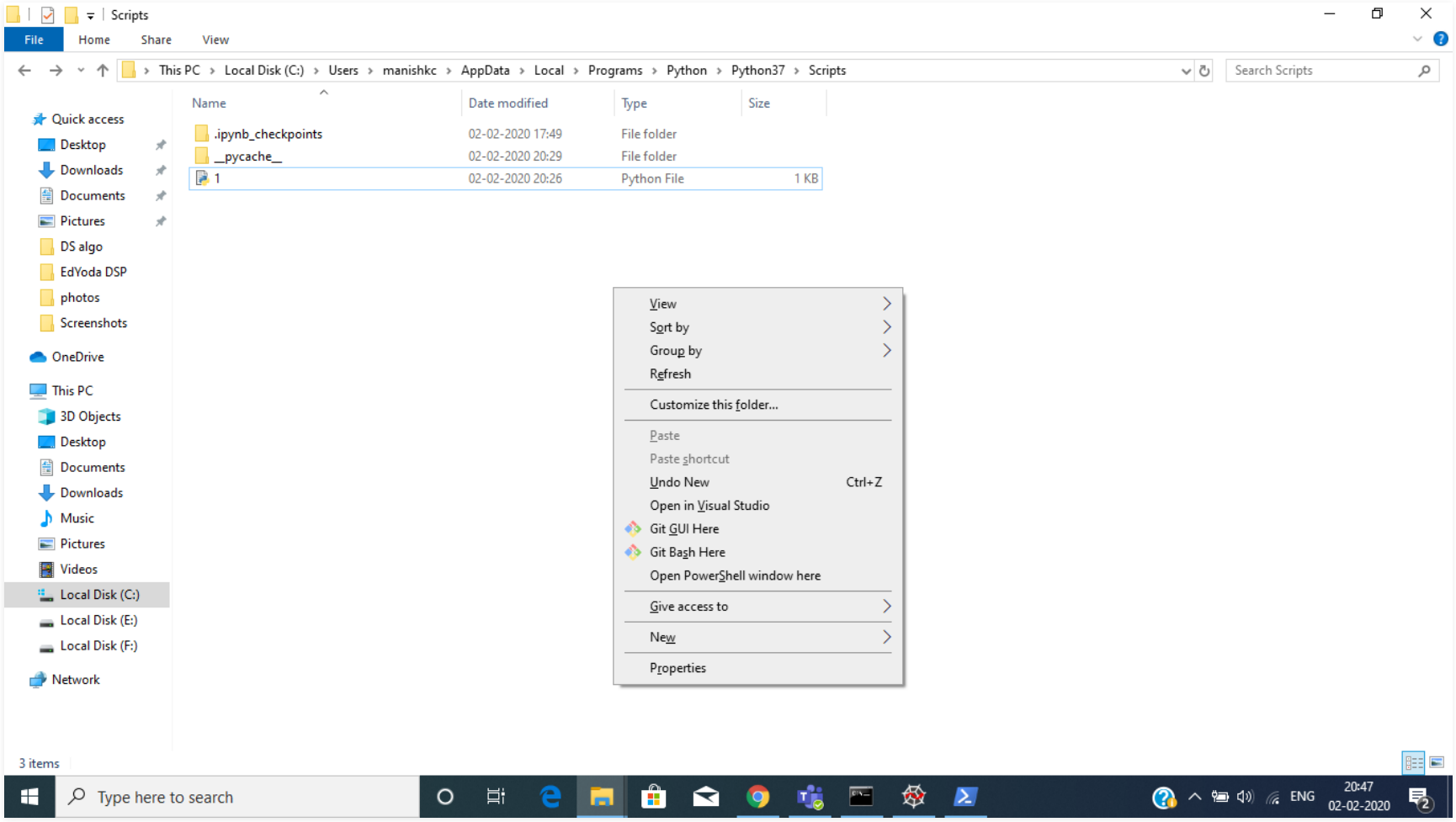
```
pip install pyinstaller
```

Step 2:



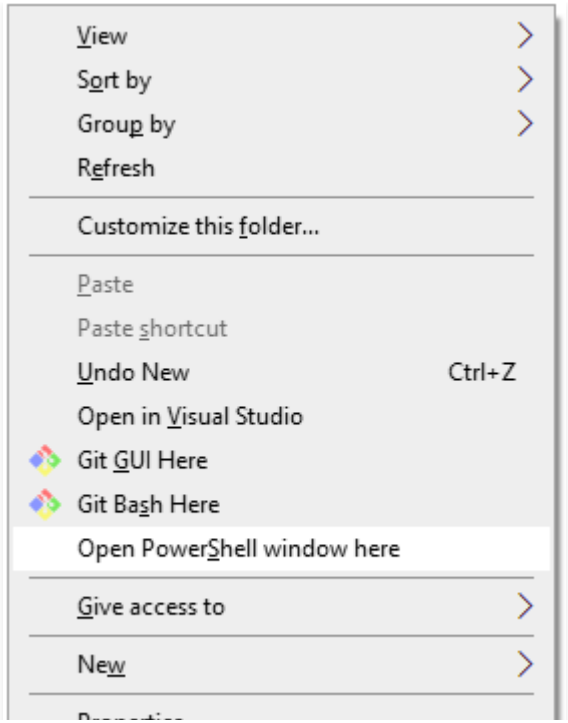
Step 3:

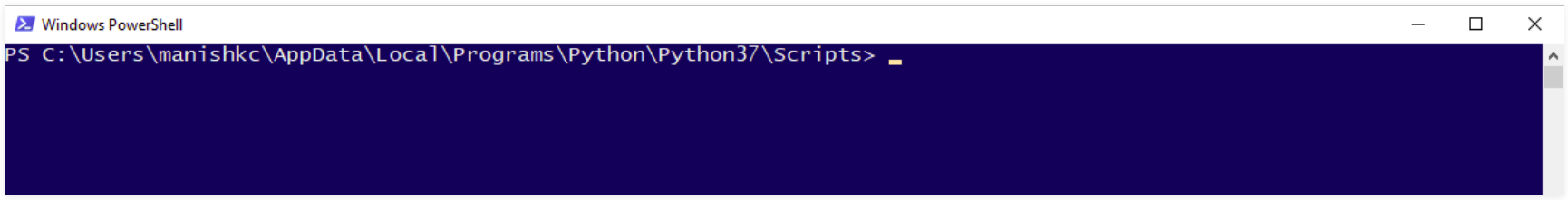
box.



Step 4:

Click on 'Open PowerShell window here'.





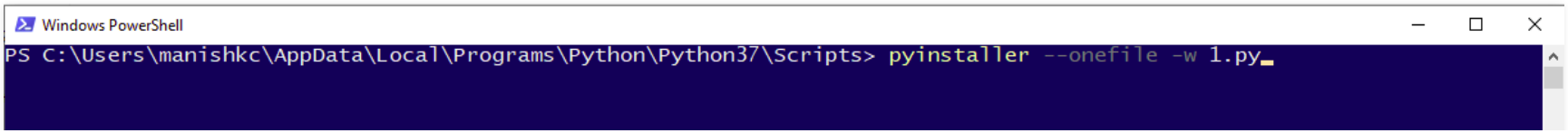
Step 5:

Type the command given below in that PowerShell window.

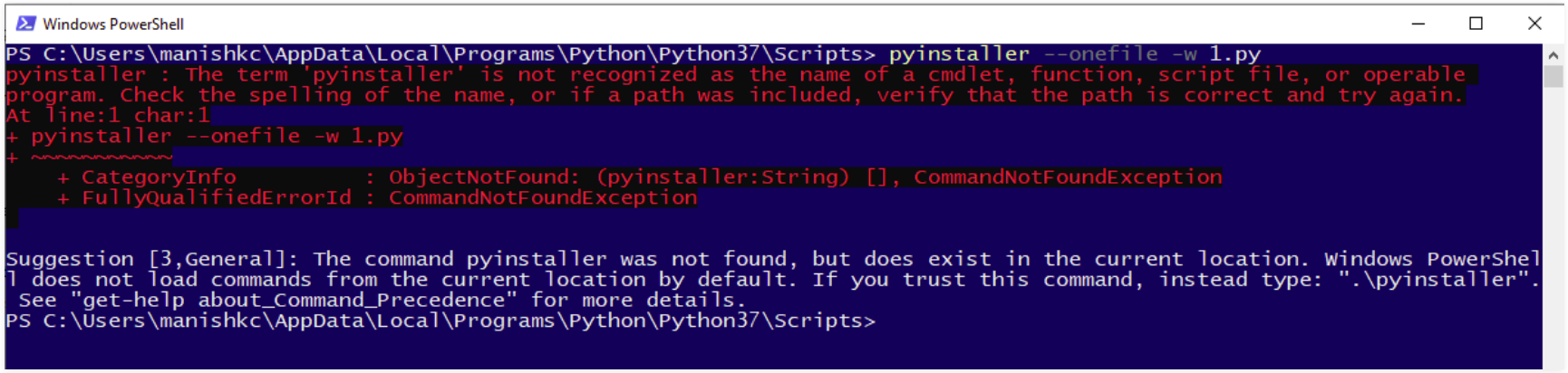
```
pyinstaller --onefile -w 'filename.py'
```

Here the '.py' file name is '1'.

See below:



In case you get an error at this point in the PowerShell window like this:



The correction while typing the above command:

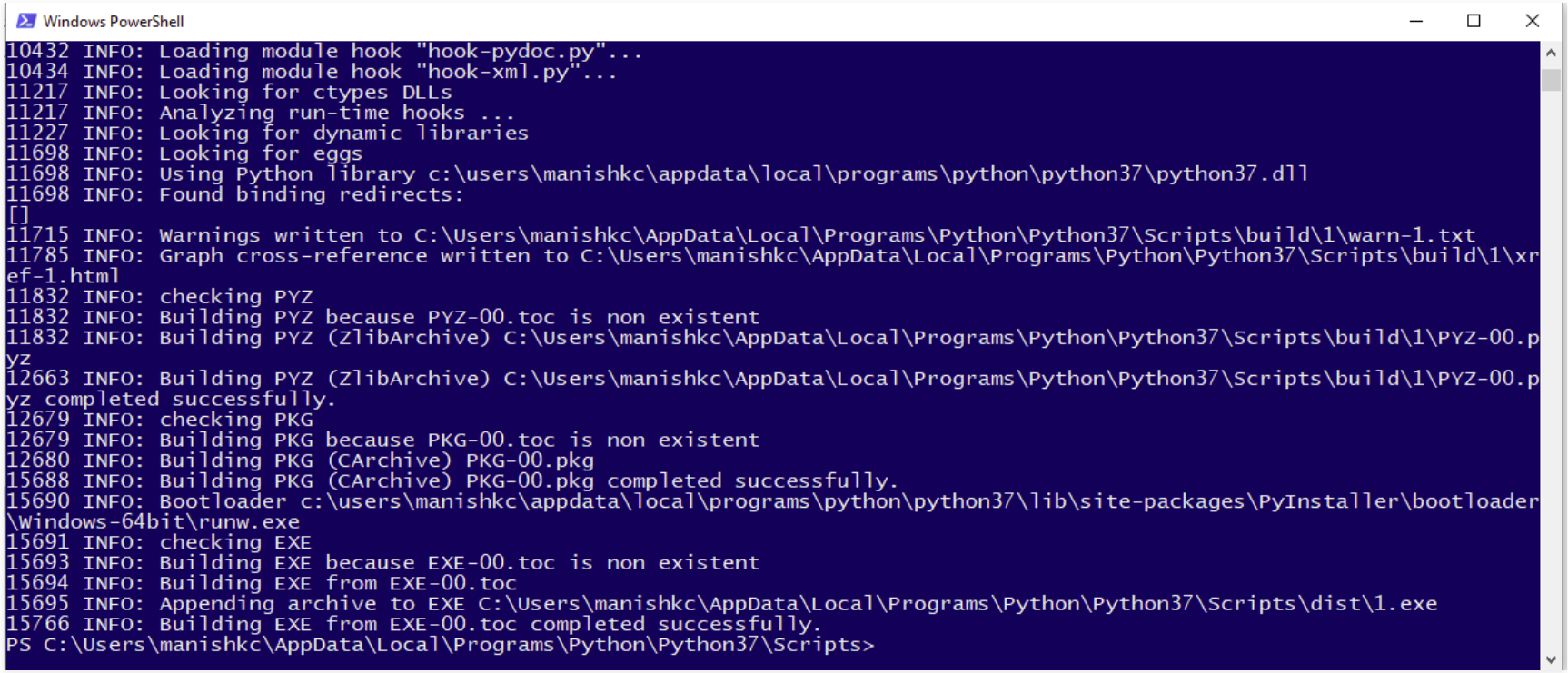
```
.\pyinstaller --onefile -w 'filename.py'

For any missing package:
pyinstaller --hidden-import 'package_name' --onefile 'filename.py'
```

Step 6:

After typing the command 'Hit the Enter'.

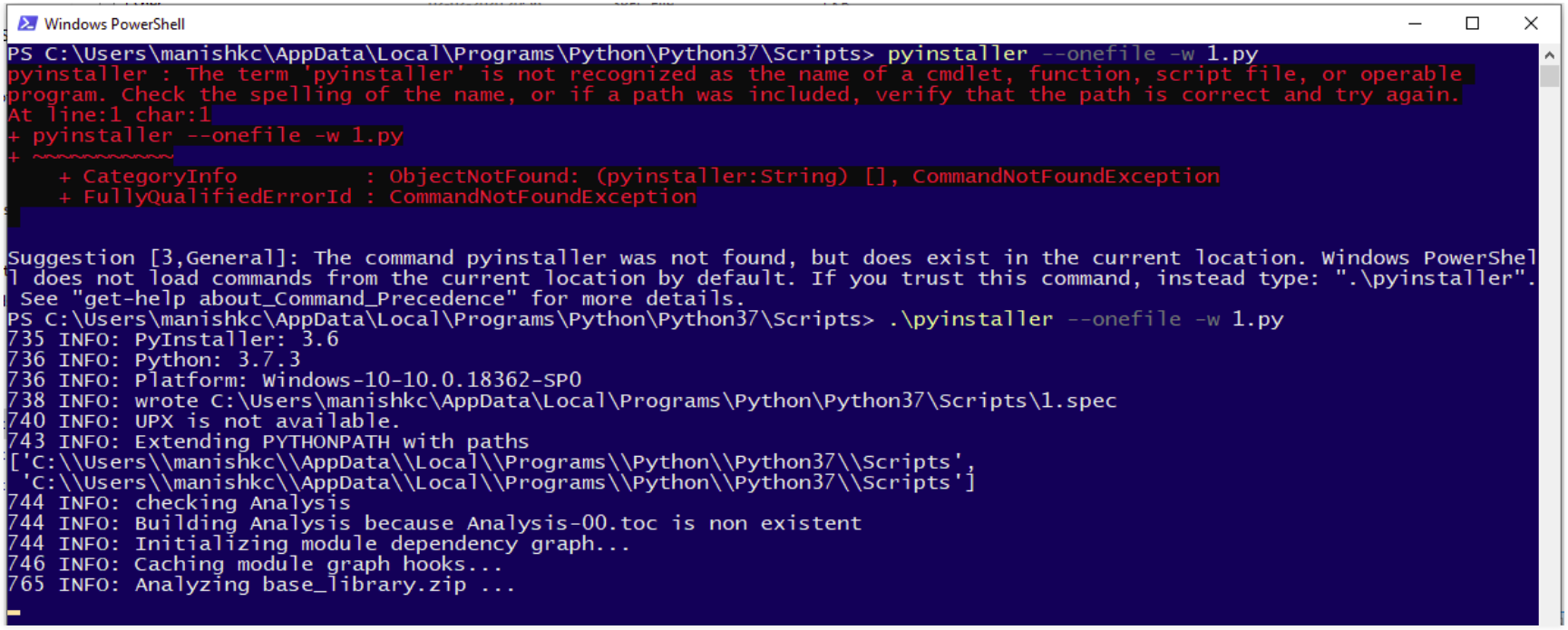
It will take some time to finish the process depending on the size of the file and how big is your project.

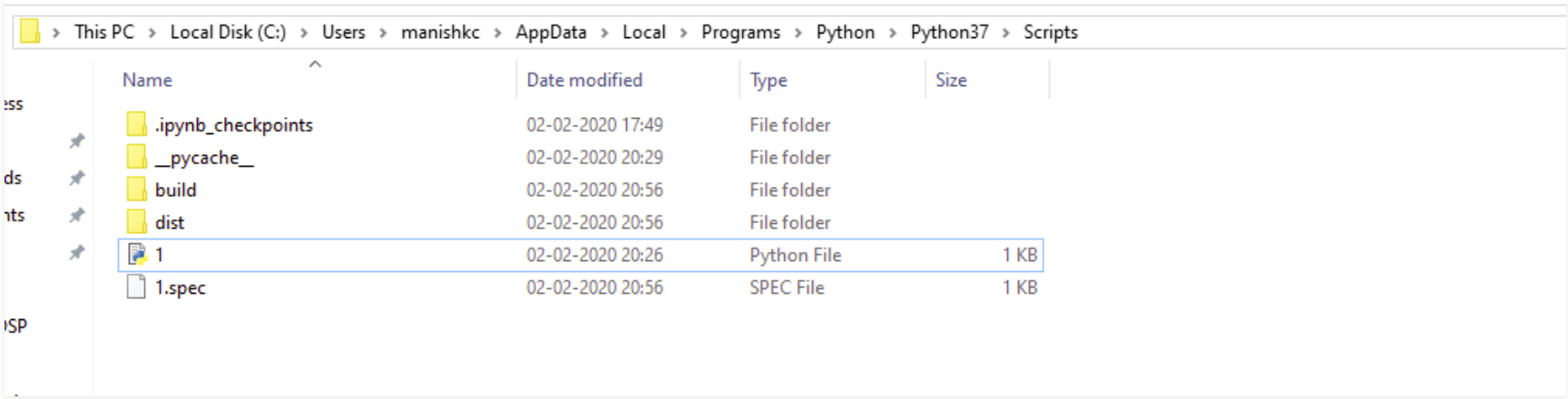
After the processing has been finished, the window will look as below:



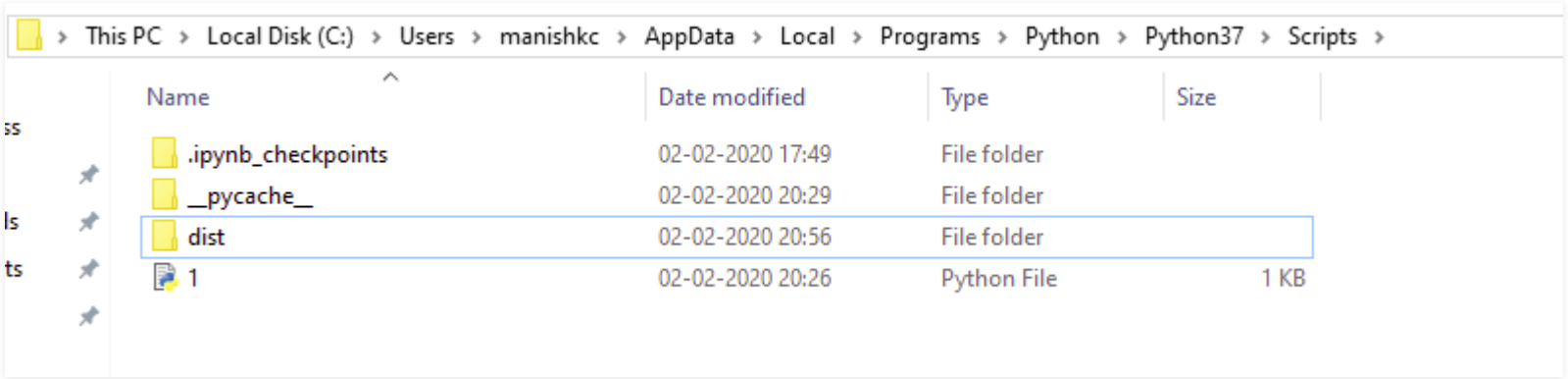
Step 7:

See the directory it should look like this:



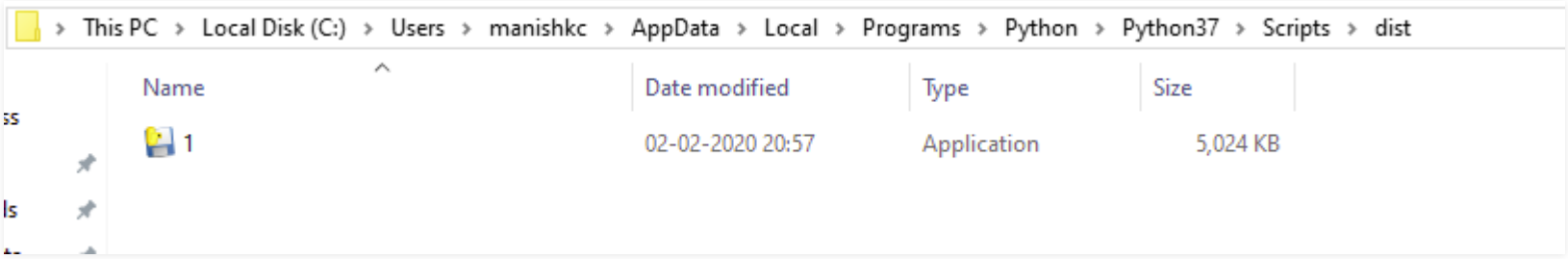


'build' folder and '1.spec' is of no use. You can delete these if you want, it will not affect your '.exe' file.

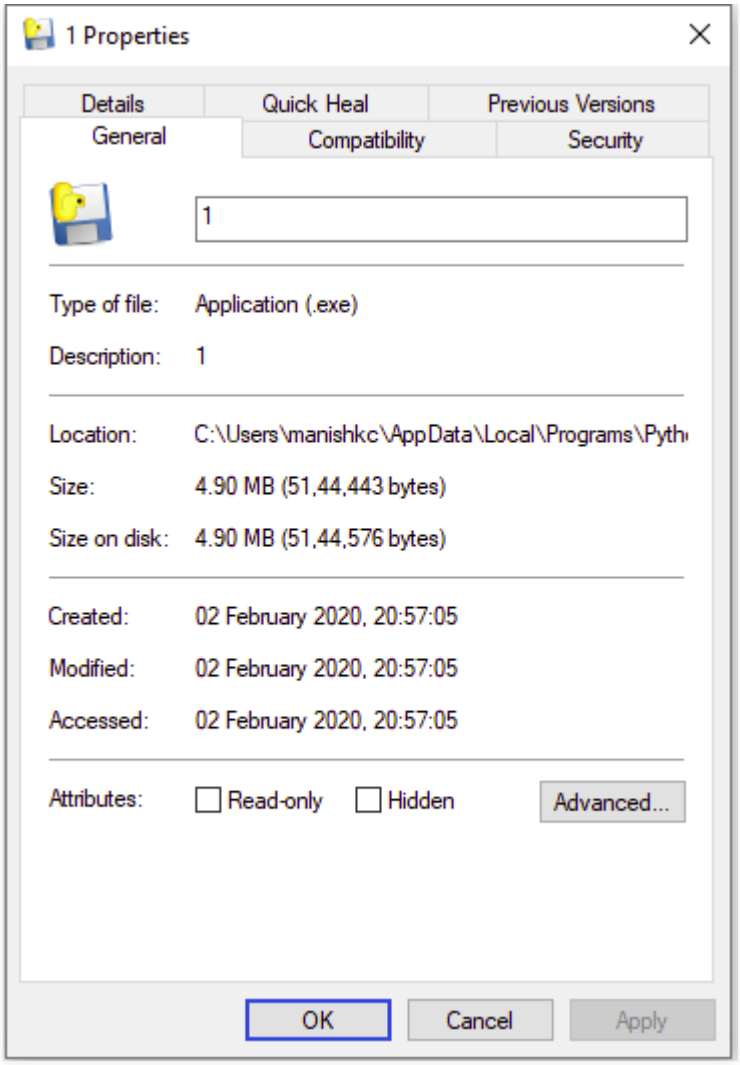


Step 8:

Open 'dist' folder above. Here you will get your '.exe' file.



Right-click on the file and check the properties.



Like 0

Next >|

Using CX_Freeze in Python

RECOMMENDED ARTICLES

Page :

Article Contributed By :

mkumarchaudhary06
@mkumarchaudhary06

Vote for difficulty

Current difficulty : Hard

Improved By : shanmalan

Article Tags : python-utility , Technical Scripter 2019 , Python , Technical Scripter

Improve Article | Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org