# Working with Zip Files in Python

Python allows you to work with zip files using the zipfile library. You check a list of files in a zip file, write files to a file zip file, create files, and a host of other things with the library. In this tutorial, you will learn how to work with zip files in Python using the zipfile library and os library. Let’s get into it.

## What is a Zip file?

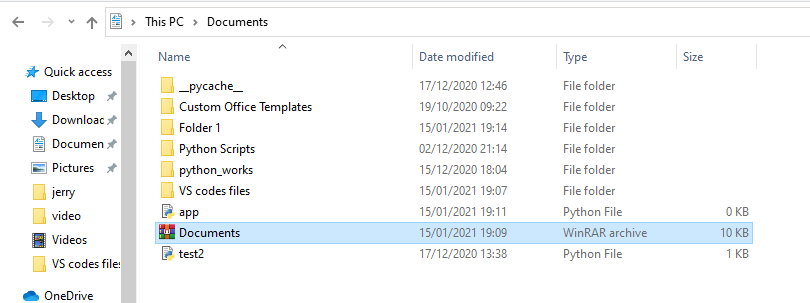
A zip file is an archive file that allows lossless data compression. A zip file can contain one or more files that have been compressed. This allows the files to be transferred as a whole without losing any fragment. Additionally, a compressed file has a smaller file size than the cumulative size of the individual files. So if you ask why we need zip files? Two reasons:

* To enhance transfer of files at maximum speed without losing any file fragment
* To reduce the overall file size.

Now lets extract a zip file in python.

## Extracting Documents in a Zip File with Python

Lets say we need to extract the zip file named ‘Documents’ on our local machine.



We run the following line of codes

#import the ZipFile class from the zipclass library

**from** **zipfile** **import** ZipFile

**with** ZipFile(r"C:\Users\DELL\Documents\Documents.zip", 'r') **as** zip:

#check the file names in the zip file with the printdir() method

**print**('These are the files in the zip file')

zip.printdir()

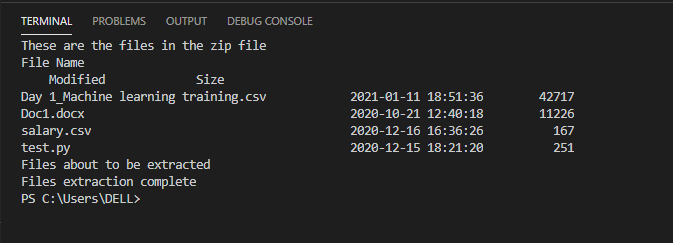
#extract the files in the zip file

**print**('Files about to be extracted')

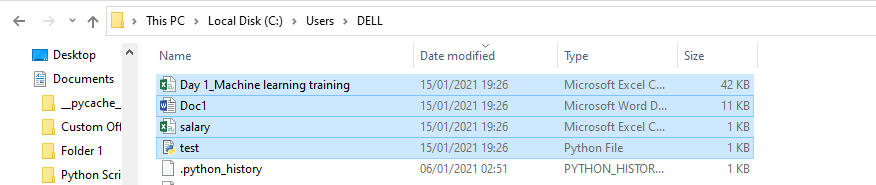
zip.extractall()

**print**('Files extraction complete')

Output:



As seen from the figure above, the code ran successfully. Let’s see if the files were truly extracted.



All the files were extracted to our working directory. Let’s break down the earlier line of code that performed the extraction.

We started by importing the necessary module. Then the ZipFile module was instantiated as zip. The ZipFile class takes two arguments, the zip file path and the mode which could be write ‘w’, read ‘r’, append ‘a’ or exclusive create ‘x’. Since we only wanted to parse the file, we used the read mode.

The printdir() method was used to check the file names and sizes of the zip file. Finally, the files were extracted using the extractall() method.

## Creating/Writing to a Zip File with Python

We have seen how to extract files in a zip file. We can also write to a zip file using the write() method. The block of code below would write the 4 files we earlier extracted to an empty zip file.

#import the ZipFile class from the zipclass library

**from** **zipfile** **import** ZipFile

#list the files to compress

files\_to\_compress = ['Day 1\_Machine learning training.csv', 'Doc1.docx', 'salary.csv', 'test.py']

**with** ZipFile('newzipfile.zip', 'w') **as** zip:

#write the each file in a zip file called newzipfile.zip

**for** each\_file **in** files\_to\_compress:

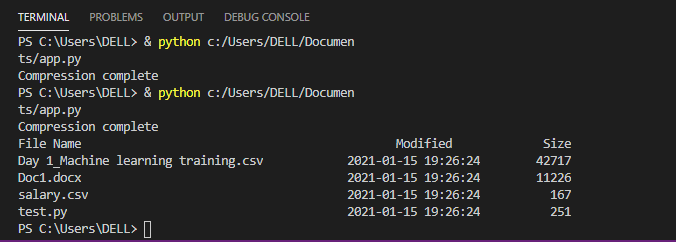
zip.write(each\_file)

**print**('Compression complete')

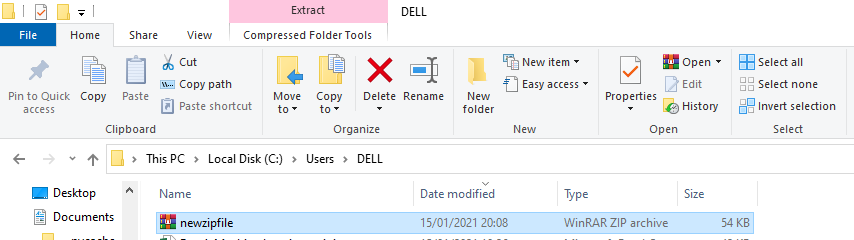
#print the files in the in the zip file

zip.printdir()

Output:



Let’s see if the zip file was created.



As seen, the zip file has been created.

If you wish to write all files in a folder to a zip file, you can use the code below.

#import necessary libraries

**import** **os**

**from** **os.path** **import** join

**from** **zipfile** **import** ZipFile

# create an empty list to store the files

files\_list = []

#use the walk method extract directory path, directory name and file name

**for** dir\_path, directory, files **in** os.walk('{directory of folders}'):

**for** each\_file **in** files:

# create a full path by joining the parent path to each file name

each\_filepath = join(dir\_path, each\_file)

#add the each file path to the empty list

files\_list.append(each\_filepath)

**with** ZipFile('{create path to store the zipped file}', 'w') **as** zip:

**for** file **in** files\_list:

zip.write(file)

In conclusion, you have learnt how to extract files in a zip file and how to write a file(s) to a zip file. If you have any questions, feel free to leave them in the comment section, I’d make sure to respond to them.