

File Modes

Method/Keyword	Meaning
r	Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode .
w	Opens a file for writing only, truncating the file first. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
x	Opens for exclusive creation, failing if the file already exists
a	Opens a file for writing/append. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
+	Open for updating (reading and writing)
r+	Opens a file for both reading and writing. The file pointer placed at the beginning of the file.
a+	Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing
w+	Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
b	Binary mode
sdffds	kjhljkh
kjhjkh	hjkjkh
hjkjkh	hjkjh
hkjjkh	jhkjkhjk
ghjkgjh	hgjhghjg
hjghjghjhghgj	jhgghjghjhghj
ghghjhjg	jghghj
jghjhghjg	hgjjhgghj
ghjghjghj	hgghjghj
jhgghjghj	hgjhghghjh
ghjhghjghj	hgjhjg
jhgghjhghj	hgjjhgjhghj
gjhjhghjg	hgjhjghjhghjg
ghjhghjghj	ghjhjhghjg
jghjhghghj	ghjhjghjhghj
ghghjghjh	ghjjhgjhghj
ghjghghjhjg	hgjjghghjghj
ghjjghghjghj	hgjjghjghghj
hjghjghjhghjg	jjhgghjhghj
jghjhghjghghj	hghghjjghjhghj



File Modes (cont)

gjhghgjh	hgjjghjg
hjghgjjghj	hgjjghjgghj
hgjhjgghjghj	jhghgjhjghj
hgjjghjghjg	hgjjghjghjg
jhghghjghjghjg	ghjghjghjghjghj
jhghghjghjghjg	hgjjghjghjghjg

Functions

Method/Keyword	Parameters	Uses
open()	filename and open mode (optional)	create a file object by opening/creating the file in the specified read/write mode
with	-	use it together with open(); closes the file after the suite completes
read()	size (optional)	read the file up to the size specified if set
readline()	size (optional)	read a single line with a size limit if set
readlines()	size (optional)	create a list of the read lines with an optional size hint
write()	the string	write the string to the file that is opened with a certain writeable mode
writelines()	the list of strings	write the list of strings to the file that is opened with a certain writeable mode
seek()	offset, whence=SEEK_SET	Change the stream position to the given byte offset
truncate()	size=None	Resize the stream to the given size in bytes (or the current position if size is not specified).

File Modes

Start	Read	Write	Create	Truncate	Cursor
r	*				Start
w		*	*	*	Start
a		*	*		End
r+	*	*			Start
w+	*	*	*	*	Start
a+	*	*	*		End
x			*		Start

readline()

```
with open('test.txt') as file:
    file.readline()
    file.readline(2) # 2 characters from
line 2
    file.readline(5) # 3 characters from
line 3
    file.readline()
    file.readline()
'0 Start Line 0 - Line 0 End 0\n'
'1 '
'Start'
' Line 1 - Line 1 End 1\n'
''
```

Writing a CSV File

```
big_list = [{'name': 'Fredrick Stein', 'userid':
6712359021, 'is_admin': False}]

import csv

with open('output.csv', 'w') as output_csv:
    fields = ['name', 'userid', 'is_admin']
    output_writer = csv.DictWriter(output_csv, fieldnames=fields)

    output_writer.writerow()
    for item in big_list:
        output_writer.writerow(item)
```

Open CSV in write mode, define **fields**, instantiate CSV writer object and pass two arguments.

.writeheader() writes headers from fieldnames.

Read File

```
with open('test.txt') as file:
    file.read(6) # Size smaller than file
size
'0 Star'
```

When the size argument is omitted or set as negative, or set as an

If you call the `readline()` method multiple times, the reading will be continued at where it was read last time.

`readline()` method can also take in a size parameter, which will be used as a limit for reading the line. Again, reading is continued at where it was read last time.

Write a File

```
with open('test.txt', 'w') as file:
    file.write('This is a writing method.')
25
```

The **w** mode will truncate the file, and thus the file will only contain the new values. The **a** mode will allow you to append new values to the existing file.

Printing the number of characters written can be suppressed by assigning the returned value to an underscore.

Reading Different Types of CSV

```
import csv

with open('addresses.csv', newline='') as
    addresses_csv:
        address_reader = csv.DictReader(
            addresses_csv, delimiter=';')
        for row in address_reader:
            print(row['Address'])
```

We change the delimiters to indicate where the different values start and stop. We pass **delimiter** parameter, which is used to delineate separate fields in the CSV.

integer greater than the file size, all the file contents will be read.

Reading a file for the second time will return an empty string.

readlines()

```
with open('test.txt') as file:
    file.readlines()
    # file.readlines(29) reads 29 characters,
    # 30 reads whole of next line
['0 Start Line 0 - Line 0 End 0\n', '1 Start Line
1 - Line 1 End 1\n']
```

When size is set as a positive integer, it will read that many characters (or bytes in the binary mode) from the file and enough to complete that line.

writelines()

```
with open('test.txt', 'w') as file:
    file.writelines(['Line 0\n', 'Line
1'])
with open('test.txt') as file:
    file.read()
'Line 0\nLine 1'
```

This method will take in a list of strings as the parameter. How the lines are written (e.g., overwriting or appending) using this method is similar to the implementation of the `write()` method.

What is a CSV File?

Comma-Separated Values are usually the way that data from spreadsheet is exported into a portable format.



Reading a CSV File

```
import csv
list_of_email_addresses = []
with open('users.csv', newline='') as users_csv:
    user_reader = csv.DictReader(users_csv)
    for row in user_reader:
        list_of_email_addresses.append(
            row['Email'])
```

We can convert data into a dictionary using ***csv** and its **DictReader** object.

newline="" ensures that we don't mistake a line break in one of our data fields as a new row in CSV.



By [exotic](#)
cheatography.com/exotic/

Not published yet.
Last updated 22nd April, 2022.
Page 4 of 4.

Sponsored by [Readable.com](#)
Measure your website readability!
<https://readable.com>