How to read a text file into a string variable and strip newlines?

Ask Question

Asked 5 years, 11 months ago Active 3 months ago Viewed 1.3 M times

Home
Tags
Users
Jobs
Teams
Q&A for work
Learn more

Stack Overflow
Tags
Bookmarks
Jobs
Users
Ask
Teams
Create a Team
Free 30 Users Free

1668 votes

I use the following code segment to read a file in python:

```
open('myfile', 'r').read(read-size)
```

Input file is:

```
line1
line2
line3
```

and when I print data I get:

```
line1
line2
line3
```

But I need data in a `list`, form. How do I make it string? And also how do I remove the `'\n'`, `'\r'`, and `'\t'` characters from it?

edited

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

23 Answers

You could use

```
with open('myfile.txt', 'r') as file:
    data = file.read().replace('\n', ' ')
```

1177 votes

share · improve this answer

answered Dec 17 '08 at 17:38
Ciro Santilli
5.7K ↑ 1.1K ↓ 4.1K

Related: How to read a file line by line into a list or string in Python? — Jeff M Oct 17 '16 at 20:12

The file and the question are mentioned. Do you really need to get rid of the \n as well? — Julian Oct 17 '16 at 20:14

@ciro, note that using the keyword `file` for variable name is problematic, as are you just confused about the difference between read-mode characters in your print output and actually want to keep the newlines, but not have them display as "\n"? — Jeff M Oct 17 '16 at 20:16

To keep spacing from from a search engine you're probably looking for: @maxyo's answer. — Jonathan Sudiaman Apr 11 '18 at 20:24

edit a comment

23 Answers

You could use

```
with open('myfile.txt', 'r') as file:
    data = file.read().replace('\n', ' ')
```

1177 votes

share · improve this answer

answered Dec 17 '08 at 17:38
Ciro Santilli
5.7K ↑ 1.1K ↓ 4.1K

Is there a shorterer or just shiny `open('myfile.txt').read().replace('\n',' ')` instead? — numbwitch Apr 11 '14 at 19:32

Yes, you version does not explicitly close the file, that will then be delayed until the garbage collector runs in the program terminates. The with statement usually encapsulates proper resource handling. — Ciro Santilli Apr 11 '14 at 19:35

@numbwitch: For clarification, So, it seems that my version might be a bit small scripty. but CPOW it should perfectly do what's intended. (shorter is not always better) — Ciro Santilli Apr 11 '14 at 19:37

@ciro santilli: It is a large PITA in the tokenizing process, as it meant back up the open file handles if you have to terminate prematurely or hit a bad character. — Ignacio Vazquez-Abrams Apr 11 '14 at 19:39

`file.read().split()` will only remove the newline from the last line. `replace("\n", " ")` removes it everywhere, including the middle of the first line. — Ignacio Vazquez-Abrams Apr 11 '14 at 19:41

edit 7 more comments

use `read()`, not `readline()`.

```
with open('myfile.txt', 'r') as myfile:
    data = myfile.read().replace('\n', ' ')
```

581 votes

share · improve this answer

answered Dec 17 '08 at 17:38
Jason Baker
42.5K ↑ 1.2K ↓ 4.6K

Pehaps, look forward into readlines as the OP wanted. Also, the data read between the parenthesis is the number of bytes to read, which is optional. — Ciro Santilli Apr 11 '14 at 19:34

Works but not idiomatic file. To support stiff on python3, use an extra argument: `encoding='utf-8'`. — FastSolutions Jun 11 '17 at 19:11

edit a comment

You can read from a file in one line:

```
text = open('myfile.txt', 'r').read()
```

64 votes

share · improve this answer

answered Dec 17 '08 at 18:02
Ned Batchelder
167K ↑ 23K ↓ 19K

Please note that this does not close the file explicitly.

CPython will close the file when it exits as part of the garbage collection.

Alternative Python implementations don't. To write portable code, it is better to use `with` or close the file explicitly. Which is not always better. See http://stackoverflow.com/q/7395542/4279.

share · improve this answer

answered Dec 8 '14 at 9:00
Jakub Narębski
106K ↑ 41K ↓ 49K

This is anti-idiomatic and not recommended. `open` should be used either in `with` or `.close()` — Ciro Santilli Apr 11 '14 at 19:34

@JJ I am you explain the problem? Is the point a question of scope in data file `with ... as` statement ending, or flushing? — Ciro Santilli Apr 11 '14 at 19:36

@ciro: Yes, it is true that open(read) doesn't close the file so as other users `with ... as` or `.close()` are demonstrated in Python answer. More on the importance of closing files. — JJ Apr 11 '14 at 19:38

@JJ: Thanks. This clearly removes a source of error. Thanks! — Tilo May 15 '17 at 21:12

this is also bad because you've just shadowed `open`. Non-builtin — Chris_Rands Aug 17 '18 at 16:5

edit 2 more comments

To join all lines into a string and remove new lines I normally use:

```
with open('myfile.txt') as f:
    s = ''.join(line.rstrip() for x in f)
```

49 votes

share · improve this answer

edited Oct 8 at 18:36
Community ♦
1 ↑ 1 ↓ 1

answered Dec 17 '08 at 17:45
Ivan Vinogradov
1.4K ↑ 1.1K ↓ 1.1K

It is going (miss)(miss)(miss)(miss) in my code line dropbox/hydra-core/1398/6/1398/2/2 — Hoque Singh Apr 11 '14 at 19:40

you may want to specify the character encoding — Pedro Lobito Feb 17 '14 at 19:36

edit a comment

In Python 3.5 or later, using `pathlib` you can copy text file contents into a variable and close the file in one line:

```
from pathlib import Path
txt = Path('myfile.txt').read_text()
```

33 votes

share · improve this answer

answered Dec 28 '15 at 17:48
Dan Stevens
5.7K ↑ 1.1K ↓ 4.1K

and then you can use `str.replace` to remove the newlines:

```
txt = txt.replace('\n', ' ')
```

share · improve this answer

answered Dec 28 '15 at 17:49
Dan Stevens
5.7K ↑ 1.1K ↓ 4.1K

edit a comment

This can be done using the `read()` method:

```
text_as_string = open('hour_hour_file.txt', 'r').read()
```

29 votes

share · improve this answer

answered Dec 18 '08 at 7:58
Jeremy
4.5K ↑ 1.2K ↓ 4.6K

Or as the default mode itself is 'r' (read) so simply use:

```
text_as_string = open('hour_hour_file.txt').read()
```

share · improve this answer

answered Dec 18 '08 at 7:58
Jeremy
4.5K ↑ 1.2K ↓ 4.6K

edit a comment

```
with open('myfile.txt') as myfile:
    data=''.join(line.rstrip() for line in myfile)
```

29 votes

share · improve this answer

answered Dec 3 '11 at 18:47
Haggai Nuchi
4.5K ↑ 1.2K ↓ 4.6K

edit a comment

I have folded around with this for a while and seems I have yet to use `.join` in combination with `.rstrip()`. Without `.rstrip()`, Python adds a newline to the end of the string, which in most cases is not very useful.

```
with open('myfile.txt') as f:
    s = ''.join(line.rstrip() for line in f)
    print file_content
```

6 votes

share · improve this answer

answered Dec 17 '08 at 17:48
Jason Baker
42.5K ↑ 1.2K ↓ 4.6K

edit 2 more comments

It's hard to tell exactly what you're after, but something like this should get you started:

```
with open('myfile.txt') as f:
    data = ''.join(line.replace('\n', ' ') for line in f.readlines())
```

4 votes

share · improve this answer

answered Dec 17 '08 at 18:01
Chris Dixon
9.1K ↑ 1.2K ↓ 4.6K

replace(data,e,'') - >>> data=f.read() Traceback (most recent call last): File "<stdin>", line 1, in <module> AttributeError: 'file' object has no attribute 'read' >>> data=f.readline() >>> data 'line1\n' >>> data=data.replace('\n','') >>> data 'line1' >>> — Cenzar Dec 17 '08 at 21:2

edit a comment

This is a one line, copy-pasteable solution that also closes the file object:

```
s = open('data.txt').read().replace('\n', ' ')
```

4 votes

share · improve this answer

answered Aug 23 '18 at 15:50
Ryan Saxe
2.3K ↑ 1.1K ↓ 1.1K

edit a comment

You can also strip each line and concatenate into a final string:

```
myfile = open('data.txt', 'r')
data = '
'
for line in myfile.readlines:
    data = data + line.strip()
```

3 votes

share · improve this answer

answered Dec 18 '08 at 8:02
Ryan Ahmed
1.1K ↑ 1.1K ↓ 1.1K

This should also work just fine. — Ryan Ahmed Aug 21 '18 at 19:5

edit a comment

you can compress this into one into two lines of code!!!

```
lookup = open('filename', 'r').read().replace('\n', ' ')
print(lookup)
```

3 votes

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

If your file reads:

```
hello line one you
hello line two you
blank line
```

python output

```
hello line one you and are your blank line
```

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

```
f = open('data.txt', 'r')
data = ''
while 1:
    line = f.readline()
    if not line:
        break
    strip = line
    data = data + strip
f.close()

print data
```

2 votes

share · improve this answer

answered Dec 17 '08 at 18:01
Ryan Ahmed
1.1K ↑ 1.1K ↓ 1.1K

Loses whole lines or strips \n. Line should be entered. Some versions of Python may manage to avoid this, but it's not guaranteed. Also, it's inefficient and non-idiomatic. Also, it leaks the file handle. Also, it's not obvious the machinery that would be required to port code to just one or two way of doing things. — Ignacio Vazquez-Abrams Dec 17 '08 at 18:04

Thank for correcting me. But one small thing is that I have not to remove the new line, because when limited, it dont'work "on" an. @Ignacio — Ryan Ahmed Dec 17 '08 at 18:1

edit a comment

python3. Google "list comprehension" if the square bracket syntax is new to you.

```
with open('data.txt', 'r') as f:
    lines = [line.strip() for line in list(f)]
```

2 votes

share · improve this answer

answered Dec 17 '08 at 17:48
Jason Baker
42.5K ↑ 1.2K ↓ 4.6K

edit a comment

I'm surprised nobody mentioned `splitlines()`. yet:

```
with open('myfile.txt', 'r') as myfile:
    data = myfile.read().splitlines()
```

2 votes

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

Variable `data` is now a list that looks like this when printed:

```
['line1', 'line2', 'line3']
```

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

Note there are no newlines ('\n').

At that point, assume the you want to print back the lines to console, which you can achieve with a for loop:

```
for line in data:
    print line
```

share · improve this answer

answered Feb 17 '14 at 1:06
Tyler
1.1K ↑ 1.1K ↓ 1.1K

This is wrong. You won't p rapidly. — Kasem Khalil Dec 29 '15 at 21:4

edit 1 more comment

I don't feel that anyone addressed the [] part of your question. When you read each line into your variable, because there were multiple lines before you replaced the '\n' with, you ended up creating a list. If you have a variable s1 and print it out you'll

```
s1 = open('myfile.txt', 'r').read().replace('\n', ' ')
print s1
```

or

```
s1 = open('myfile.txt', 'r').read().replace('\n', ' ')
print s1[0]
```

You will see the entire first line with the brackets. If you call each element of the (array of sorts) s1[0] then it omits the brackets. If you use the str() function you will see just the data and not the " either. str(s1[0])

share · improve this answer

answered Feb 17 '14 at 20:19
Kevin Fegan
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

Maybe you could try this? I use this in my programs.

```
f = open('data.txt', 'r')
data = f.readlines()
for line in data:
    data2 = data2.strip()
    data = f.readline().strip()
```

share · improve this answer

answered Jul 14 '20 at 20:22
user11755347

edit a comment

This works. Change your file to:

```
line1
line2
line3
```

0 votes

share · improve this answer

Then:

```
file = open("file.txt")
line = file.read()
words = line.split()
```

This creates a list named `words` that equals:

```
['line1', 'line2', 'line3']
```

Then grab all of the "in". To answer the part about the newlines getting in your way, just do this:

```
for word in words:
    print word + "looking words in the list above"
```

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

```
print name[0:n] + " Hello Kim Jong-Il" sqlite3 indicates no spaces
file names can be alphanumeric and have ...
```

This returns:

share · improve this answer

answered Dec 25 '10 at 20:54
Ashley
1.1K ↑ 1.1K ↓ 1.1K

Changing the file might work in a one off situation but if you have hundreds of files this just ain't a workable solution. — Jonathan Dec 17 '08 at 17:48

edit a comment

```
with open(filename, mode, "r") as myfile:
    for line in myfile:
    lineNum = lineNum + 1
    if lineNum == 2:
        print line
```

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

This code will help you to read the first line and then using the list and split option you can convert the first line read into words and access the second in a list.

Than you can easily access any word, or even store it in a string.

You can also do the same thing with using a for loop.

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

```
file = open('myfile.txt', 'r')
lines = file.readlines()

                    #string declaration
for l in range(len(lines)):
    lines[l] = lines[l].strip()

print str
```

share · improve this answer

answered Dec 17 '08 at 18:03
Ryan Ahmed
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

Regular expression works too:

```
import re
with open('myfile.txt', 'r') as f:
    data = f.read()
    print re.split(r'\n+', data)[0], re.split(r'\n+', data)[1]

[]

['Test', 'array', 'and', 'Newl', 'Header']
```

share · improve this answer

answered Aug 14 '09 at 15:08
Ryan Ahmed
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

To remove line breaks using Python you can use `rstrip()` function of a string.

This example removes all 3 types of line breaks:

```
my_string = open('data.txt').read()
print my_string

my_string = my_string.replace("\r","").replace("\n","")
print my_string
```

Example file is:

```
line1
"data", "data",
"data"
```

share · improve this answer

answered Dec 17 '08 at 17:48
Michael Kent
1.1K ↑ 1.1K ↓ 1.1K

edit a comment

Your Answer

Sign up or log in

Sign up using Google

Sign up using Facebook

Sign up using Email and Password

Post as a guest

Name

Email

Required, but never shown

Post Your Answer

By clicking "Post Your Answer", you agree to our terms of service, privacy policy and cookie policy

Not the answer you're looking for? Browse other questions tagged: python or ask your own question.

STACK OVERFLOW Questions Jobs Developer Jobs Directory Salary Calculator Help Mobile Stack Exchange Responsibles

PRODUCTS Teams Talent Advertising Enterprise

COMPANY About Press Work Here Legal Privacy Policy Contact Us

STACK EXCHANGE NETWORK Technology Life / Arts Culture / Recreation Science Other

Blog Facebook Twitter LinkedIn

site design / logo © 2021 Stack Exchange Inc; user contributions licensed under cc by-sa 4.0 with attribution required.
rev 2021.9.24.40143