



Reading Files

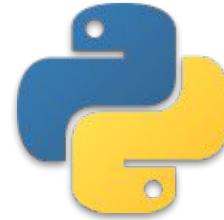




Table of Contents

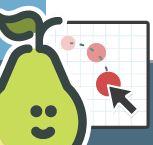
- ▶ `open()` Function
- ▶ `mode` Parameter
- ▶ Reading the Files with `.read()` Method
- ▶ Reading the Files with `.readline()` Method
- ▶ Reading the Files with `.readlines()` Method
- ▶ Reading the Files with Loops
- ▶ The Matter of Closing the Files



1

open() Function

How was your pre-class content. Did you understand the “Reading Files with Python”?



Students, drag the icon!



Pear Deck Interactive Slide
Do not remove this bar

open() Function (review)

- ▶ Parameters of the **open()** function :

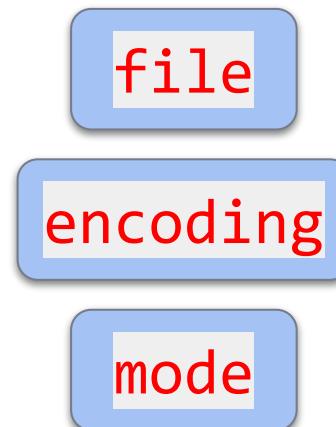
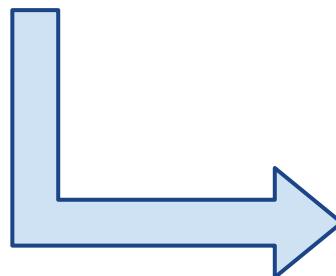
```
open (file, mode='r', buffering=-1, encoding=None,  
errors=None, newline=None, closefd=True, opener=None)
```

open() Function (review)



- We will mostly examine and work with some of the following parameters of the `open()` function :

```
open (file, mode='r', buffering=-1, encoding=None,  
errors=None, newline=None, closefd=True, opener=None)
```



open() Function (review)

file

- ▶ **file** is the only required parameter of the `open()` function. This parameter is a path-like object which is a `str` or `bytes` that represents a path in the file directory.

```
open ("file_name_and_path")
```

open() Function (review)

- ▶ You can see the current path of your Jupyter using **pwd** command.

```
In [4]: 1 pwd
```

```
Out[4]: 'C:\\\\Users\\\\YD'
```

open() Function (review)

file

- In the following example, you see that we open a .txt file object. We passed the required parameter (**file**) into the **open()** function.

```
1 my_file = open("first_file.txt") # this syntax opens a 'txt' file
2
3 print(type(my_file))
```

open() Function (review)

file

```
1 my_file = open("first_file.txt") # this syntax opens a 'txt' file  
2  
3 print(type(my_file))
```

```
1 <class '_io.TextIOWrapper'>  
2
```

open() Function (review)

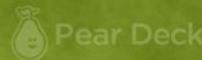
encoding

- ▶ We can open a file in the same way by using the following syntax :

```
1 my_file = open("first_file.txt", encoding="utf-8")
2 # we've used 'utf8' encoding format just the same as the previous one
```

I have learned enough about the encoding parameter so far.

True



False



Students choose an option



mode



2

mode Parameter

mode Parameter (review)



Character	What it's used for?
'r'	Open for reading (default). If the file doesn't exist, <code>FileNotFoundException</code> will raise.
'a'	Open for writing. It will append to the end of the file if it already exists. If there is no file, it will create it.
'w'	Open for writing. It will be overwritten if the file already exists. If there is no file, it will create it.
'x'	Open for exclusive creation, it will fail if the file already exists.
'b'	Open in binary mode.
't'	Open as a text file (default).
'+'	Open for updating (reading and writing).

mode Parameter (review)

- ▶ By default, `open()` function executes opening the file for reading ('`r`') as a text ('`t`'). In other words, it defaults to '`r`' or '`rt`' which means open for reading in text mode.



Tips:

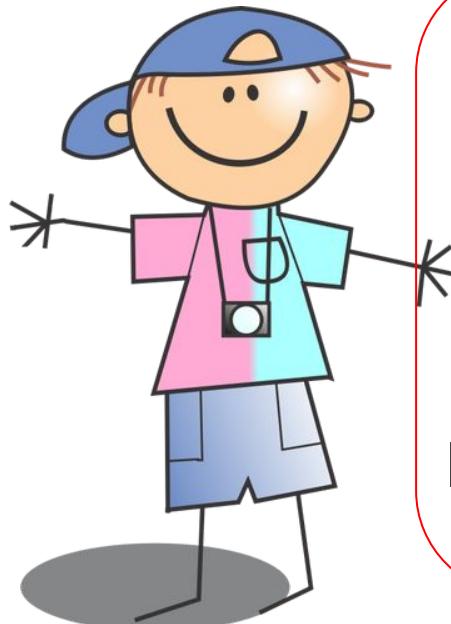
- Note that, as you can see above, depending on what you are going to do with the file, you can combine the modes according to your needs.
- If you want to *open* and be able to *read*, *modify*, and *update* the existing file you should set the mode as '`r+`'.

mode Parameter (review)

- ▶ Another **important point** we should cover about **mode** is that you can choose the format in which you want to open the file as text ('**t**') or binary ('**b**').

'**b**'

Returns contents as
bytes objects
without any
decoding.



'**t**'

The contents of
the file are
returned as **str**.
It's **default** for mode
parameter.

most common use



Reading Methods

- ▶ You can use the following methods for reading a file :

`.read(size)`

`.readline(size)`

`.readlines()`

`using loops`



3

Reading the Files with `.read()` Method

Reading Methods



- ▶ Create a text file manually named **fishes.txt**:

fishes.txt

Orca is a kind of Dolphin.

Blue Whale is the largest animal known on earth.

Sharks are the sister group to the Rays (batoids).

The Tuna Fish can weigh up to 260 kg.

Squid and Octopus are in the same class.

Reading Methods



Keep in your mind!

You can pass **full path** of the file into `open()` function. So that, when the file path is entered, the location where the file is saved does not matter.

```
file = open("C:/users/desktop/fishes.txt", "r")
```

Basic Structure of Working with Files

```
file = open("file_name", "r")
```

```
...
```

```
body
```

```
...
```

```
file.close()
```

.read() Method (review)

- ▶ Reading all content at once :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read()) # displays the entire text content
4
5 sea.close() # be sure to close the file
```

.read() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read()) # displays the entire text content
4
5 sea.close() # be sure to close the file
```

- 1 Orca is a kind of Dolphin.
- 2 Blue Whale is the largest animal known on earth.
- 3 Sharks are the sister group to the Rays (batoids).
- 4 The Tuna Fish can weigh up to 260 kg.
- 5 Squid and Octopus are in the same class.
- 6

.read() Method (review)

- ▶ Reading limited number of characters :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 sea.close()
6
```

What is the output? Try to figure out in your mind...

.read() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 sea.close()
6
```

```
1 Orca is a kind of Dolphin.
2 Blue W
```

.read() Method (review)

- Now let's continue to read using the **size** parameter.

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.close()
8
9
```

What is the output? Try to figure out in your mind...

.read() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.close()
8
9
```

```
3
4 hale is the largest anima
5 |
```

.read() Method (review)

- Now let's continue to read using the **size** parameter.

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.seek(0) # changes the stream (cursor) position to zero
8 print(sea.read(33)) # displays the first 33 chars again
9
10 sea.close()
11
```

What is the output? Try to figure out in your mind...

.read() Method (review)

► The output:

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.seek(0) # changes the stream (cursor) position to zero
8 print(sea.read(33)) # displays the first 33 chars again
9
10 sea.close()
11
```

```
5
6 Orca is a kind of Dolphin.
7 Blue W
8
```



.read() Method (review)

- ▶ Find the cursor :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.seek(0) # changes the stream (cursor) position to zero
8 print(sea.read(33)) # displays the first 33 chars again
9
10 print(sea.tell()) # returns the current stream (cursor) position
11
12 sea.close()
13
```

What is the output? Try to figure out in your mind...



.read() Method (review)

► The output:

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.read(33)) # displays the first 33 chars of the text
4
5 print(sea.read(25)) # displays the next 25 chars of the text
6
7 sea.seek(0) # changes the stream (cursor) position to zero
8 print(sea.read(33)) # displays the first 33 chars again
9
10 print(sea.tell()) # returns the current stream (cursor) position
11
12 sea.close()
13
```

```
8
9 34
10
```

.read() Method



► Task :

- ▶ Create a **txt** file named **rumi.txt** in your current working directory consisting of the following quotes of Rumi.

rumi.txt

```
I want to sing
Like the birds sing,
Not worrying about
Who hears or
What they think.
```

.read() Method

► Task (continued):

- ▶ Create a **txt** file named **rumi.txt** in your current directory consisting of the following quotes of Rumi.
- ▶ Read and display the entire contents of this file at once.
- ▶ Display the *first two lines* using `read()` method.

.read() Method (review)



► Task (continued):

- ▶ Display the next **13** chars of the content,
- ▶ Display the current location of the cursor,
- ▶ Bring the cursor onto beginning of the second line and display the second line again.
- ▶ Close the file.

.read() Method (review)

- ▶ The entire code block can be as follows (suppose you've created the **rumi.txt** manually into your current directory):

```
1 f = open("rumi.txt", "r", encoding="utf-8")
2
3 print(f.read(35))
4 print(f.read(13))
5 print(f.tell())
6 f.seek(15)
7 print(f.read(20))
8
9 f.close()
10
```



.read() Method (review)

- ▶ The output :

Output

First 35 chars
(first two lines)

```
I want to sing  
Like the birds sing,
```

The next 13 chars

```
Not worrying
```

```
48
```

```
Like the birds sing,
```

The current location
of the cursor



4

Reading the Files with `.readline()` Method

.readline() Method (review)

- ▶ Reading text files line by line :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readline()) # displays the first line of the text
4 print(sea.readline()) # displays the second line
5 print(sea.readline()) # each time it goes to the new line
6
7 sea.close()
```

What is the output? Try to figure out in your mind...

.readline() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readline()) # displays the first line of the text
4 print(sea.readline()) # displays the second line
5 print(sea.readline()) # each time it goes to the new line
6
7 sea.close()
```

```
1 Orca is a kind of Dolphin.
2
3 Blue Whale is the largest animal known on earth.
4
5 Sharks are the sister group to the Rays (batoids).
6
```

.readline() Method (review)

- ▶ So that, first line of text consists of 27 *characters*. Let's choose 13 as the **size** parameter and see what happens :

`fishes.txt`

```
Orca is a kind of Dolphin.  
Blue Whale is the largest animal known on earth.  
Sharks are the sister group to the Rays (batoids).  
The Tuna Fish can weigh up to 260 kg.  
Squid and Octopus are in the same class.
```

.readline() Method (review)



fishes.txt

Orca is a kind of Dolphin.\n



=27

same class. But the same things are not occupied by the same class.

.readline() Method (review)

```
1 sea = open("fishes.txt", 'r')  
2  
3 print(sea.readline(13))  
4 print(sea.readline(13))  
5 print(sea.readline(13))  
6 print(sea.readline(13))  
7  
8 sea.close()
```

What is the output? Try to figure out in your mind...



USAY®
Students, write your response!
REINVENT YOURSELF

.readline() Method (review)

```
1 sea = open("fishes.txt", 'r')  
2  
3 print(sea.readline(13))  
4 print(sea.readline(13))  
5 print(sea.readline(13))  
6 print(sea.readline(13))  
7  
8 sea.close()
```

```
1 Orca is a kin  
2 d of Dolphin.
```

\n is the last char of
the first line

when a line ends, whatever
the **size** parameter is, it goes
ahead to the next line

```
4  
5 Blue Whale is  
6
```

default empty
line

.readline() Method

► Task :

- Consider the **rumi.txt** file you created before.

rumi.txt

```
I want to sing
Like the birds sing,
Not worrying about
Who hears or
What they think.
```

.readline() Method

► Task (continued) :

- ▶ Read and display the first line of this file,
- ▶ Read and display the second line of this file,
- ▶ Read and display the third line of this file using **size** parameter in the method,
- ▶ Close the file.

.readline() Method

- ▶ The entire code block can be as follows (suppose you've created the **rumi.txt** manually into your current directory):

```
1 f = open("rumi.txt", "r", encoding="utf-8")
2
3 print(f.readline())
4 print(f.readline())
5 print(f.readline(18))
6
7 f.close()
8
```

.readline() Method

- ▶ The output :

Output

```
I want to sing
```

```
Like the birds sing,
```

```
Not worrying about
```

reading of 18 chars
displays the third line



5

Reading the Files with `.readlines()` Method

.readlines() Method (review)

- ▶ Using our same text file, let's look at the following example on how the output looks like :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readlines())
4
5 sea.close()
```

What is the output? Try to figure out in your mind...

.readlines() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readlines())
4
5 sea.close()
```

```
1 ['Orca is a kind of Dolphin.\n', 'Blue Whale is the largest animal known on
2 earth.\n', 'Sharks are the sister group to the Rays (batoids).\n', 'The Tuna
3 Fish can weigh up to 260 kg.\n', 'Squid and Octopus are in the same class.']
4
```

.readlines() Method (review)

- Let's take a look at the following example where .readline() and .readlines() are used together.

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readline()) # first line
4 print(sea.readlines()) # the rest of the lines
5
6 sea.close()
```

What is the output? Try to figure out in your mind...



.readlines() Method (review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 print(sea.readline()) # first line
4 print(sea.readlines()) # the rest of the lines
5
6 sea.close()
```

```
1 Orca is a kind of Dolphin.
2
3 ['Blue Whale is the largest animal known on earth.\n', 'Sharks are the sister
4 group to the Rays (batoids).\n', 'The Tuna Fish can weigh up to 260 kg.\n',
5 'Squid and Octopus are in the same class.']
6 |
```

.readlines() Method (review)

- You may have noticed that the type of the `sea.readlines()` object is the **list**. Let's see it :

```
1 sea = open("fishes.txt", 'r')
2
3 print(type(sea.readlines()))
4
5 sea.close()
```

```
1 <class 'list'>
2
```

⚠ Attention ! :

- Although it is very good to read relatively small files with this method, but when large files are involved, this method can be inefficient.

.readlines() Method



► Task :

- ▶ Consider the **rumi.txt** file you created before.

rumi.txt

```
I want to sing
Like the birds sing,
Not worrying about
Who hears or
What they think.
```

.readlines() Method

► Task (continued) :

- ▷ Read and display the entire file in a **list** form,
- ▷ Close the file.

.readlines() Method

- ▶ The entire code block can be as follows (suppose you've created the **rumi.txt** manually into your current directory):

```
1 f = open("rumi.txt", "r", encoding="utf-8")
2
3 print(f.readlines())
4
5 f.close()
6
```

.readlines() Method

- ▶ The output :

Output

```
['I want to sing\n', 'Like the birds sing,\n', 'Not worrying about\n', 'Who hears or\n', 'What they think']
```



6

Reading the Files with loops

Reading the Files with Loops(review)

- The most efficient way to read the contents of a file is using the for loop. Let's take a look at an example of how we do this.

```
1 sea = open("fishes.txt", 'r')
2
3 for line in sea:
4     print(line)
5
6 sea.close()
```

What is the output? Try to figure out in your mind...

Reading the Files with Loops(review)

- ▶ The output :

```
1 sea = open("fishes.txt", 'r')
2
3 for line in sea:
4     print(line)
5
6 sea.close()
```

```
1 Orca is a kind of Dolphin.
2
3 Blue Whale is the largest animal known on earth.
4
5 Sharks are the sister group to the Rays (batoids).
6
7 The Tuna Fish can weigh up to 260 kg.
8
9 Squid and Octopus are in the same class.
10
```

Reading the Files with Loops(review)

- Since the type of `sea.readlines()` object is a `list`, we can use it as an iterator and read the whole file in the same way.

```
1 sea = open("fishes.txt", 'r')
2
3 for line in sea.readlines():
4     print(line)
5
6 sea.close()
```

```
1 Orca is a kind of Dolphin.
2
3 Blue Whale is the largest animal known on earth.
4
5 Sharks are the sister group to the Rays (batoids).
6
7 The Tuna Fish can weigh up to 260 kg.
8
9 Squid and Octopus are in the same class.
10
```

Reading the Files with Loops

► Task :

- Consider the **rumi.txt** file you created before.

rumi.txt

```
I want to sing
Like the birds sing,
Not worrying about
Who hears or
What they think.
```

Reading the Files with Loops

► **Task (continued) :**

- ▶ Read and display the entire file with loops using file-like object as an iterator,
- ▶ Read and display the entire file with loops using `.readlines()`-produced `list` as an iterator,
- ▶ Close the file.

Reading the Files with Loops

- The entire code block can be as follows (suppose you've created the **rumi.txt** manually into your current directory):

```
1 f = open("rumi.txt")
2
3 for line in f:
4     print(line)
5
6 f.close()
7
```

f is **file-like object**
and it's used as an
iterator

Reading the Files with Loops

► The output :

Output

```
I want to sing
```

```
Like the birds sing,
```

```
Not worrying about
```

```
Who hears or
```

```
What they think
```

Reading the Files with Loops

- The entire code block can be as follows (suppose you've created the **rumi.txt** manually into your current directory):

```
1 f = open("rumi.txt", "r", encoding="utf-8")
2
3 for line in f.readlines():
4     print(line)
5
6 f.close()
7
```

f.readlines() is a
list type and it's used
as an iterator

Reading the Files with Loops

- ▶ It gives the same output :

Output

```
I want to sing
```

```
Like the birds sing,
```

```
Not worrying about
```

```
Who hears or
```

```
What they think
```

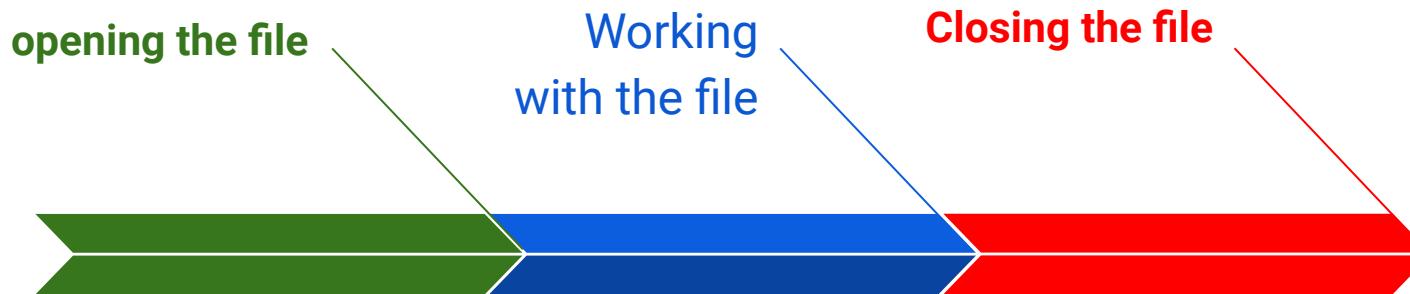


7

The Matter of Closing Files

The Matter of Closing the Files(review)

Workflow of working with files to be followed



The Matter of Closing the Files(review)

- Syntax of working with files :

```
with open("my_file.txt", "r", encoding="utf-8") as f:  
    code block
```

The Matter of Closing the Files(review)

- Take a look at the way we use it :

```
1 with open("fishes.txt", "r") as sea:  
2     print(sea.read()) # needs indented code block
```

- 1 Orca is a kind of Dolphin.
- 2 Blue Whale is the largest animal known on earth.
- 3 Sharks are the sister group to the Rays (batoids).
- 4 The Tuna Fish can weigh up to 260 kg.
- 5 Squid and Octopus are in the same class.
- 6

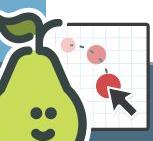
Using the **with ... as ...** block

- Create another txt file,
- Implement the same steps (reading methods) in the tasks shown in this presentation.

You can complete this assignment by yourself or in a group session later.



How well did you like this lesson?



Students, drag the icon!





THANKS!

End of the Lesson

(Reading Files)

next Lesson

Writing Files

click above

