Python Read Text File

Summary: in this tutorial, you learn various ways to read text files in Python.

TL;DR

The following shows how to read all texts from the `readme.txt` file into a string:

```
with open('readme.txt') as f:
    lines = f.readlines()
```

Steps for reading a text file in Python

To read a text file in Python, you follow these steps:

- First, open a text file for reading by using the `open()` function.
- Second, read text from the text file using the file `read()` , `readline()` , or `readlines()` method of the file object.
- Third, close the file using the file `close()` method.

1) open() function

The `open()` function has many parameters but you'll be focusing on the first two:

```
open(path_to_file, mode)
```

The `path_to_file` parameter specifies the path to the text file.

If the program and file are in the same folder, you need to specify only the filename of the file. Otherwise, you need to include the path to the file as well as the filename.

To specify the path to the file, you use the forward-slash (`'/'`) even if you're working on Windows.

For example, if the file `readme.txt` is stored in the `sample` folder as the program, you need to specify the path to the file as `c:/sample/readme.txt`

The `mode` is an optional parameter. It's a string that specifies the mode in which you want to open the file. The following table shows available modes for opening a text file:

Mode	**Description**
`'r'`	Open for text file for reading text
`'w'`	Open a text file for writing text
`'a'`	Open a text file for appending text

For example, to open a file whose name is `the-zen-of-python.txt` stored in the same folder as the program, you use the following code:

```
f = open('the-zen-of-python.txt','r')
```

The `open()` function returns a file object which you will use to read text from a text file.

2) Reading text methods

The file object provides you with three methods for reading text from a text file:

- `read(size)` – read some contents of a file based on the optional size and return the contents as a string. If you omit the size, the `read()` method reads from where it left off till the end of the file. If the end of a file has been reached, the `read()` method returns an empty string.
- `readline()` – read a single line from a text file and return the line as a string. If the end of a file has been reached, the `readline()` returns an empty string.
- `readlines()` – read all the lines of the text file into a list of strings. This method is useful if you have a small file and you want to manipulate the whole text of that file.

3) close() method

The file that you open will remain open until you close it using the `close()` method.

It's important to close the file that is no longer in use for the following reasons:

- First, when you open a file in your script, the file system usually locks it down so no other programs or scripts can use it until you close it.
- Second, your file system has a limited number of file descriptors that you can create before it runs out of them. Although this number might be high, it's possible to open a lot of files and deplete your file system resources.
- Third, leaving many files open may lead to race conditions which occur when multiple processes attempt to modify one file at the same time and can cause all kinds of unexpected behaviors.

The following shows how to call the `close()` method to close the file:

```
f.close()
```

To close the file automatically without calling the `close()` method, you use the `with` statement like this:

```
with open(path_to_file) as f:
    contents = f.readlines()
```

In practice, you'll use the `with` statement to close the file automatically.

Reading a text file examples

We'll use the-zen-of-python.txt file for the demonstration.

The following example illustrates how to use the `read()` method to read all the contents of the `the-zen-of-python.txt` file into a string:

```
with open('the-zen-of-python.txt') as f:
    contents = f.read()
    print(contents)
```

Output:

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
...
```

The following example uses the `readlines()` method to read the text file and returns the file contents as a list of strings:

```
with open('the-zen-of-python.txt') as f:
    [print(line) for line in f.readlines()]
```

Output:

```
Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

...
```

The reason you see a blank line after each line from a file is that each line in the text file has a newline character (\n). To remove the blank line, you can use the `strip()` method. For example:

```
with open('the-zen-of-python.txt') as f:
    [print(line.strip()) for line in f.readlines()]
```

The following example shows how to use the `readline()` to read the text file line by line:

```
with open('the-zen-of-python.txt') as f:
    while True:
        line = f.readline()
        if not line:
            break
        print(line.strip())
```

Output:

```
Explicit is better than implicit.
Complex is better than complicated.
Flat is better than nested.
...
```

A more concise way to read a text file line by line

The `open()` function returns a file object which is an iterable object. Therefore, you can use a `for` loop to iterate over the lines of a text file as follows:

```
with open('the-zen-of-python.txt') as f:
    for line in f:
        print(line.strip())
```

This is a more concise way to read a text file line by line.

Read UTF-8 text files

The code in the previous examples works fine with ASCII text files. However, if you're dealing with other languages such as Japanese, Chinese, and Korean, the text file is not a simple ASCII text file. And it's likely a UTF-8 file that uses more than just the standard ASCII text characters.

To open a UTF-8 text file, you need to pass the `encoding='utf-8'` to the `open()` function to instruct it to expect UTF-8 characters from the file.

For the demonstration, you'll use the following `quotes.txt` file that contains some quotes in Japanese.

The following shows how to loop through the `quotes.txt` file:

```
with open('quotes.txt', encoding='utf8') as f:
    for line in f:
        print(line.strip())
```

Output:

Summary

- Use the `open()` function with the `'r'` mode to open a text file for reading.
- Use the `read()`, `readline()`, or `readlines()` method to read a text file.
- Always close a file after completing reading it using the `close()` method or the `with` statement.
- Use the `encoding='utf-8'` to read the UTF-8 text file.