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 file is in the same folder as the program, you just need to specify the name of the file. Otherwise, you need to specify the path to the file.

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

For example, if the file is `readme.txt` 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()` – read all text from a file into a string. This method is useful if you have a small file and you want to manipulate the whole text of that file.
- `readline()` – read the text file line by line and return all the lines as strings.
- `readlines()` – read all the lines of the text file and return them as a list of strings.

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. If you don't close the file, the program may crash or the file would be corrupted.

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 (https://www.pythontutorial.net/wp-content/uploads/2020/10/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:

```
lines = []
with open('the-zen-of-python.txt') as f:
    lines = f.readlines()

count = 0
for line in lines:
    count += 1
    print(f'line {count}: {line}')
```

Output:

```
line 1: Beautiful is better than ugly.

line 2: Explicit is better than implicit.

line 3: Simple is better than complex.
...
```

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:
    line = f.readline()
    while line:
        line = f.readline()
        print(line)
```

Output:

```
Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.
...
```

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

The `open()` function returns a file object which is an iterable (https://www.pythontutorial.net/python-basics/python-iterables/) 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)
```

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` (https://www.pythontutorial.net/wp-content/uploads/2020/10/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.