Manage files and folders in Python

Python also provides a variety of methods to handle various directory-related operations. In this article, we will learn about managing files and directories in Python, namely creating folders, renaming folders, listing folders and working with them.

In this article, Quantum will work with you on managing files and directories in Python, namely creating folders, renaming folders, listing folders and working with them.

Directory in Python

In the previous article, we became familiar with the file and manipulate files in Python. Here you can simply understand that the directory is where all the files are. Sort the code as well as the file in different folders to easily manage everything.

Python also provides a variety of methods to handle various directory-related operations. The module os has been built to provide methods to help you create, delete, and change folders.

Show current directory

The *getcwd () method* displays the current working directory, returning the result as a string.

We can also use the *getcwdb () method* to get the result in bytes.

```
>>> import os

>>> os.getcwd()
'C:Program FilesPyScripter'
```

```
>>> os.getcwdb()
b'C:Program FilesPyScripter'
```

Change the current directory

The current working directory can be changed by the *chdir () method.*

chdir () receives a parameter as the name of the directory you want to go from the current directory. It is possible to use either a slash (/) or a backslash () to separate elements in the path, but it is still best to use a backslash ().

```
>>> os.chdir('C:Python33')

>>> print(os.getcwd())
C:Python33
```

List of folders and files

You can list all files and subdirectories inside a directory using the *listdir () method.*

This method takes a path and returns a list of subdirectories and files in that path.

If no path is specified, the returned result will be retrieved from the current working directory.

```
>>> print(os.getcwd())
C:Python33

>>> os.listdir()
['DLLs',
'Doc',
'include',
'Lib',
'libs',
'LICENSE.txt',
'NEWS.txt',
'python.exe',
'pythonw.exe',
'README.txt',
'Scripts',
'tcl',
'Tools']

>>> os.listdir('G:')
['$RECYCLE.BIN',
'Movies',
'Music',
'Photos',
'Series',
'System Volume Information']
```

Create a new folder

To create new folders, use the *mkdir () method* of Module os.

You can choose where to store the new folder by writing the full path to the place you want to create. If the full path is not specified, the new directory will be created in the current working directory.

```
>>> os.mkdir('test')

>>> os.listdir()
['test']
```

Rename the folder or file name

You use the *rename ()* method to *rename* a folder or a file.

```
>>> os.listdir()
['test']

>>> os.rename('test','new_one')

>>> os.listdir()
['new_one']
```

Delete the folder or file

To remove a file, you use the *remove () method.*

Similar to deleting the entire directory, use *rmdir ()*

```
>>> os.listdir()
['new_one', 'old.txt']

>>> os.remove('old.txt')
>>> os.listdir()
['new_one']

>>> os.rmdir('new_one')
>>> os.listdir()
[]
```

Note that the *rmdir () method* can only delete empty directories.

So to remove a non-empty directory, we can use the *rmtree* () method inside the *shutil* module .

```
>>> os.listdir()
['test']

>>> os.rmdir('test')
Traceback (most recent call last):
.
```

```
OSError: [WinError 145] The directory is not empty: 'test'

>>> import shutil

>>> shutil.rmtree('test')
>>> os.listdir()
[]
```

So you are familiar with the most basic operations with the directory. The os module in Python still provides a lot of other useful methods to perform operations with files and folders. Stay tuned for the next lessons of Quantrimang.

Wish you learn Python very fun!

See more:

1. Matrix in Python
2. Module in Python
3. The Python exercise has a solution

Previous article: Working with File in Python

Next lesson: Error and Exception in Python