OS Module in Python with Examples

Difficulty Level : Easy • Last Updated : 16 Jun, 2022

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system.

Handling the Current Working Directory

Consider **Current Working Directory(CWD)** as a folder, where the Python is operating. Whenever the files are called only by their name, Python assumes that it starts in the CWD which means that name-only reference will be successful only if the file is in the Python's CWD.

Note: The folder where the Python script is running is known as the Current Directory. This is not the path where the Python script is located.

Getting the Current working directory

To get the location of the current working directory os.getcwd() is used.

Example:

Python3

```
# Python program to explain os.getcwd() method

# importing os module
import os

# Get the current working
# directory (CWD)
cwd = os.getcwd()

# Print the current working
# directory (CWD)
print("Current working directory:", cwd)
```

Output:

```
Current working directory: /home/nikhil/Desktop/gfg
```

Changing the Current working directory

To change the current working directory(CWD) os.chdir() method is used. This method changes the CWD to a specified path. It only takes a single argument as a new directory path.

Note: The current working directory is the folder in which the Python script is operating.

Example:

Python3

```
# Python program to change the
# current working directory


import os

  Function to Get the current
```

```
    print(os.getcwd())
    print()


# Driver's code
# Printing CWD before
current_path()

# Changing the CWD
os.chdir('../')

# Printing CWD after
current_path()
```

Output:

```
Current working directory before
C:\Users\Nikhil Aggarwal\Desktop\gfg

Current working directory after
C:\Users\Nikhil Aggarwal\Desktop
```

Creating a Directory

There are different methods available in the OS module for creating a directory. These are –

- os.mkdir()
- os.makedirs()

Using os.mkdir()

os.mkdir() method in Python is used to create a directory named path with the specified numeric mode. This method raises FileExistsError if the directory to be created already exists.

Example:

ython3

```
# importing os module
import os

# Directory
directory = "GeeksforGeeks"

# Parent Directory path
parent_dir = "D:/Pycharm projects/"

# Path
path = os.path.join(parent_dir, directory)

# Create the directory
# 'GeeksForGeeks' in
# '/home / User / Documents'
os.mkdir(path)
print("Directory '% s' created" % directory)

# Directory
directory = "Geeks"

# Parent Directory path
parent_dir = "D:/Pycharm projects"

# mode
mode = 0o666

# Path
path = os.path.join(parent_dir, directory)

# Create the directory
# 'GeeksForGeeks' in
# '/home / User / Documents'
# with mode 0o666
os.mkdir(path, mode)
print("Directory '% s' created" % directory)
```

Output:

```
Directory 'GeeksforGeeks' created
Directory 'Geeks' created
```

Using os.makedirs()

os.makedirs() method in Python is used to create a directory recursively. That means while making leaf directory if any intermediate-level directory is missing,

Example:

Python3

```
# Python program to explain os.makedirs() method

# importing os module
import os

# Leaf directory
directory = "Nikhil"

# Parent Directories
parent_dir = "D:/Pycharm projects/GeeksForGeeks/Authors"

# Path
path = os.path.join(parent_dir, directory)

# Create the directory
# 'Nikhil'
os.makedirs(path)
print("Directory '% s' created" % directory)

# Directory 'GeeksForGeeks' and 'Authors' will
# be created too
# if it does not exists



# Leaf directory
directory = "c"

# Parent Directories
parent_dir = "D:/Pycharm projects/GeeksforGeeks/a/b"

# mode
mode = 0o666

path = os.path.join(parent_dir, directory)

# Create the directory 'c'

os.makedirs(path, mode)
print("Directory '% s' created" % directory)


  'GeeksForGeeks', 'a', and 'b'
  will also be created if
# it does not exists
# os.makedirs() method will
# create them

# os.makedirs() method can be
# used to create a directory tree
```

Output:

```
Directory 'Nikhil' created
Directory 'c' created
```

Listing out Files and Directories with Python

os.listdir() method in Python is used to get the list of all files and directories in the specified directory. If we don’t specify any directory, then the list of files and directories in the current working directory will be returned.

Example:

Python3

```
# Python program to explain os.listdir() method

# importing os module
import os

# Get the list of all files and directories
# in the root directory
path = "/"
dir_list = os.listdir(path)

print("Files and directories in '", path, "' :")

# print the list
print(dir_list)
```

Output:

```
Files and directories in ' / ' :
['sys', 'run', 'tmp', 'boot', 'mnt', 'dev', 'proc', 'var', 'bin',
```

Deleting Directory or Files using Python

OS module proves different methods for removing directories and files in Python. These are –

- Using os.remove()
- Using os.rmdir()

Using os.remove()

os.remove() method in Python is used to remove or delete a file path. This method can not remove or delete a directory. If the specified path is a directory then OSError will be raised by the method.

Example: Suppose the file contained in the folder are:

Python3

```
# Python program to explain os.remove() method

# importing os module
import os
```

```
# File location
location = "D:/Pycharm projects/GeeksforGeeks/Authors/Nikhil/"

# Path
path = os.path.join(location, file)

# Remove the file
# 'file.txt'
os.remove(path)
```

Output:

Using os.rmdir()

os.rmdir() method in Python is used to remove or delete an empty directory. OSError will be raised if the specified path is not an empty directory.

Example: Suppose the directories are

Python3

```python
# Python program to explain os.rmdir() method

# importing os module
import os

# Directory name
directory = "Geeks"

# Parent Directory
parent = "D:/Pycharm projects/"

# Path
path = os.path.join(parent, directory)

# Remove the Directory
# "Geeks"
os.rmdir(path)
```

Output:

Commonly Used Functions

1. os.name: This function gives the name of the operating system dependent module imported. The following names have currently been registered: ‘posix’, ‘nt’, ‘os2’, ‘ce’, ‘java’ and ‘riscos’.

Python3

```
import os

print(os.name)
```

Output:

```
posix
```

Note: It may give different output on different interpreters, such as ‘posix’ when you run the code here.

2. os.error: All functions in this module raise OSError in the case of invalid or inaccessible file names and paths, or other arguments that have the correct type, but are not accepted by the operating system. os.error is an alias for built-in OSError exception.

ython3

```
try:
    # If the file does not exist,
    # then it would throw an IOError
    filename = 'GFG.txt'
    f = open(filename, 'rU')
    text = f.read()
    f.close()

# Control jumps directly to here if
# any of the above lines throws IOError.
except IOError:

    # print(os.error) will <class 'OSError'>
    print('Problem reading: ' + filename)

# In any case, the code then continues with
# the line after the try/except
```

Output:

```
Problem reading: GFG.txt
```

3. os.popen(): This method opens a pipe to or from command. The return value can be read or written depending on whether the mode is ‘r’ or ‘w’.

Syntax:

```
os.popen(command[, mode[, bufsize]])
```

Parameters mode & bufsize are not necessary parameters, if not provided, default ‘r’ is taken for mode.

Python3

```
import os
fd = "GFG.txt"

# popen() is similar to open()
file = open(fd, 'w')
file.write("Hello")
file.close()
file = open(fd, 'r')
text = file.read()
print(text)
```

```
file.write("Hello")
# File not closed, shown in next function.
```

Output:

```
Hello
```

Note: Output for popen() will not be shown, there would be direct changes into the file.

4. os.close(): Close file descriptor fd. A file opened using open(), can be closed by close()only. But file opened through os.popen(), can be closed with close() or os.close(). If we try closing a file opened with open(), using os.close(), Python would throw TypeError.

Python3

```
import os


fd = "GFG.txt"
file = open(fd, 'r')
text = file.read()
print(text)
os.close(file)
```

Output:

```
Traceback (most recent call last):
  File "C:\Users\GFG\Desktop\GeeksForGeeksOSFile.py", line 6, in
    os.close(file)
TypeError: an integer is required (got type _io.TextIOWrapper)
```

Note: The same error may not be thrown, due to the non-existent file or permission privilege.

sufficient privilege permission to change the file.

Python

```
import os


fd = "GFG.txt"
os.rename(fd,'New.txt')
os.rename(fd,'New.txt')
```

Output:

```
Traceback (most recent call last):
  File "C:\Users\GFG\Desktop\ModuleOS\GeeksForGeeksOSFile.py", line 3,
in
    os.rename(fd,'New.txt')
FileNotFoundError: [WinError 2] The system cannot find the
file specified: 'GFG.txt' -> 'New.txt'
```

Understanding the Output: A file name “GFG.txt” exists, thus when os.rename() is used the first time, the file gets renamed. Upon calling the function os.rename() second time, file “New.txt” exists and not “GFG.txt”
thus Python throws FileNotFoundError.

6. os.remove(): Using the Os module we can remove a file in our system using the remove() method. To remove a file we need to pass the name of the file as a parameter.

Python3

```
import os #importing os module.

os.remove("file_name.txt") #removing the file.
```

The OS module provides us a layer of abstraction between us and the operating system. When we are working with os module always specify the absolute path pending upon the operating system the code can run on any os but we need to

7. os.path.exists(): This method will check whether a file exists or not by passing the name of the file as a parameter. OS module has a sub-module named PATH by using which we can perform many more functions.

Python3

```
import os
#importing os module

result = os.path.exists("file_name") #giving the name of the file as a parameter

print(result)
```

Output

```
False
```

As in the above code, the file does not exist it will give output False. If the file exists it will give us output True.

8. os.path.getsize(): In this method, python will give us the size of the file in bytes. To use this method we need to pass the name of the file as a parameter.

Python3

```
import os #importing os module

size = os.path.getsize("filename")

print("Size of the file is", size," bytes.")
```

Output:

```
Size of the file is 192 bytes.
```

This article is contributed by **Piyush Doorwar**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the eeksforGeeks main page and help other Geeks.

Like

Previous

Next

Related Articles

1. colorsys module in Python with examples
2. MySQL-Connector-Python module in Python
3. twitter-text-python (ttp) module - Python
4. Secrets | Python module to Generate secure random numbers
5. Python calendar module : formatmonth() method
6. Python | Writing to an excel file using openpyxl module
7. median() function in Python statistics module
8. OS Path module in Python

10. uniform() method in Python Random module

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Easy

Easy Normal Medium Hard Expert

Improved By : Akanksha_Rai, mejustmyself9, ddeevviissaavviittaa, harshkes, amartyaghoshgfg, geekygirl2001

Article Tags : python-modules, python-os-module, Python

Practice Tags : python

Report Issue

A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Start Your Coding Journey Now!

Login

Register

Contact Us

Privacy Policy

Copyright Policy

Advertise with us

SDE Cheat Sheet

Machine learning

CS Subjects

Video Tutorials

Courses

News

Top News

Technology

Work & Career

Business

Finance

Lifestyle

Knowledge

Languages

Python

Java

CPP

Golang

C#

SQL

Kotlin

Web Development

Web Tutorials

Django Tutorial

HTML

JavaScript

Bootstrap

ReactJS

NodeJS

Contribute

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship