

Python OS (Most used methods)

os.listdir() :

❖ it returns the whole contents of a folder (folders + files)

Use case:

- we can use it to find a file in a specific folder

```
1 def iterate_files(path):
2     # Loop through all the items in the path
3     for item in os.listdir(path):
4         # Join the path and the item name
5         item_path = os.path.join(path, item)
6         # Check if the item is a file
7         if os.path.isfile(item_path):
8             # Do something with the file
9             print(item_path)
```

os.walk() :

- ❖ it returns a tuple of (root, dirs, files) for each subdirectory
- ❖ iterate through files and directories recursively in a given directory

Use case:

- we can use it to find a file in a specific folder recursively

Example:

In this folder , the python script will

- 1 - iterate at the given folder first and find (sub-folders + files)
- 2 - will enter each sub-folders and walk inside it at each (sub-folders + files)
- 3 - It will return 3 tuples (root, dirs, files) as shown in the picture

Name	Date modified	Type
folder1	1/16/2024 1:40 AM	File folder
folder2	1/16/2024 1:40 AM	File folder
folder3	1/16/2024 1:40 AM	File folder
file 1	1/16/2024 1:41 AM	Text Document
file 2	1/16/2024 1:41 AM	Text Document
file 3	1/16/2024 1:41 AM	Text Document

root	directory	files
'D:\\1_Embedded\\GitHub atmega32 driver\\Folder0'	['folder1', 'folder2', 'folder3']	['file 1.txt', 'file 2.txt', 'file 3.txt']
'D:\\1_Embedded\\GitHub atmega32 driver\\Folder0\\folder1'	[], []	
'D:\\1_Embedded\\GitHub atmega32 driver\\Folder0\\folder2'	[], []	
'D:\\1_Embedded\\GitHub atmega32 driver\\Folder0\\folder3'	[], []	

Example :

Search for a file (in 2 ways) using :

- 1- os
- 2- pathlib

```
1 from pathlib import Path
2 import os
3
4
5 def search_file(path, name):
6     # Loop through all subdirectories in the path
7     for root, dirs, files in os.walk(path):
8         # Loop through all files in the current directory
9         for file in files:
10             # Check if the file name matches the name
11             if file == name:
12                 # Return the absolute path of the file
13                 return os.path.join(root, file)
14     # Return None if no file is found
15     return None
16
17 def search_file2(path, name):
18
19     files_pathes = [files for files in path.rglob(name)]
20     for file_path in files_pathes:
21         return file_path
22     return None
23
24
25 # Get the parent directory of this script
26 script_directory = Path(__file__).parent
27 # create object path
28 another_directory = Path(r"D:\1_Embedded\GitHup atmega32 driver\Folder0")
29
30
31 # join pathes
32 log_file_path = script_directory / "logfile.txt"
33
34 # search for the file
35 myfile_path1 = search_file(script_directory,"logfile.txt")
36 myfile_path2 = search_file2(script_directory,"logfile.txt")
37
38 print(type(myfile_path1))
39 print(type(myfile_path2))
40
41 with open(myfile_path1, 'r') as file2:
42     # Process the contents of file2
43     contents2 = file2.read()
44     print(contents2)
45
46 with open(myfile_path2, 'r') as file2:
47     # Process the contents of file2
48     contents2 = file2.read()
49     print(contents2)
50
51
```

Find Current Python_File Location :

Using OS:

```
1 import os
2
3
4 myFile = os.path.realpath(__file__)
5 myPath = os.path.dirname(os.path.realpath(__file__))
6
7 print(myFile) # out : D:\1_Embedded\My Repo's\Studying Python\main.py
8 print(myPath) # out : D:\1_Embedded\My Repo's\Studying Python
```

Using Pathlib:

```
1 from pathlib import Path
2 script_path = Path(__file__).absolute()
3 script_directory = script_path.parent
4
5 print("Script path      :", script_path)      # Script path      : d:\1_Embedded\My Repo's\Studying Python\main.py
6 print("Script directory:", script_directory)  # Script directory: d:\1_Embedded\My Repo's\Studying Python
```

Difference between getcwd() and abspath() :

```
1 # Get the current directory path of the script
2 current_directory = os.path.dirname(os.path.abspath(__file__))
3 print(current_directory) # d:\1_Embedded\My Repo's\Studying Python
4
5
6 # Get the current Working directory path
7 # Might not always be the same as the script's directory, especially if the script was executed from a different location.
8 current_directory = os.getcwd()
9 print(current_directory) # D:\1_Embedded\My Repo's
```

Python Pathlib(Most used methods)

Get Current Script Path :

```
1 from pathlib import Path
2
3 script_name = Path(__file__).name
4 script_path = Path(__file__).absolute()      # Get Absolute Path of Current Script
5
6 script_directory_1 = script_path.parent       # Move one step back from the script directory
7 script_directory_2 = Path(__file__).parent    # same as the previous line
8
9 parent_directory = script_path.parent.parent  # Move two steps back from the script directory
10
11 print("Script name      :", script_name)      # Script name      : main.py
12 print("Script path      :", script_path)      # Script path       : d:\1_Embedded\My Repo's\Studying Python\main.py
13 print("script_directory_1:", script_directory_1) # Script directory: d:\1_Embedded\My Repo's\Studying Python
14 print("script_directory_2:", script_directory_2) # Script directory: d:\1_Embedded\My Repo's\Studying Python
15 print("parent_directory :", parent_directory)  # parent directory: d:\1_Embedded\My Repo's\
```

Make New Directory :

```
1 """ Make New Directories inside new Directory """
2
3 new_directory = script_directory / 'python-file-paths' / 'foo' / 'bar' / 'baz.file'
4 new_directory.mkdir(parents=True, exist_ok=True)
5
6 # exist_ok=True : Fix error when trying to make a directory that already exists
7 # parents=True   : Fix error when trying to make a directory that doesn't exist like 'bar'
```

Searching For Files or directories:

```
1 """ Searching for files Using rglob """
2
3 # Search for Specific file Extension
4 found_files = script_directory.rglob('*.py')
5
6 # The asterisk (*) is used to search for all files and folder in the current directory.
7 found_files = script_directory.rglob('*')
8 # Search with Partial Name
9 found_files = script_directory.rglob('**/*{file_name_partial}*')
10
11 # Search for Files in a Specific Subdirectory .. its goes to script_directory and find folder mypackages then start searching in it
12 specific_subdirectory_files = [file for file in script_directory.rglob('mypackages/*') if file.is_file()]
13
14 # Search with Multiple Criteria
15 specific_subdirectory_python_files = [file for file in script_directory.rglob('mypackages/*.py') if file.is_file()]
```

Searching For Files only or directories only in specific path :

```
1 """ Looping on the files or directories after getting it """
2
3 found_files = script_directory.rglob('*')
4 MyFiles = [file for file in found_files if file.is_file()]      # Filter the results to include only files
5
6 found_files = script_directory.rglob('*')
7 MyFolders = [folder for folder in found_files if folder.is_dir()] # Filter the results to include only Folders
```

Search Recursively using rglob

```
1 Syntax :  
2 Path.rglob(pattern)
```

Use Folder.parts To Exclude specific folder and its content from the search

```
1 """  
2     folder.parts : ('path', 'to', 'start', 'directory')  
3     is an attribute that returns a tuple representing the individual components of the path.  
4     Each element in the tuple corresponds to a part of the path, such as directories and the final file or directory.  
5 """  
6 print(script_directory.parts) # ('d:\\', '1_Embedded', 'My Repo's', 'Studying Python')  
7  
8 Example :  
9 # find all folder and sub folder and exclude (.git) and its content  
10 all_directories = [folder for folder in script_directory.rglob('*') if folder.is_dir() and folder.name != '.git' and not any(part == '.git' for part in folder.parts)]
```

Save File paths in 2 ways:

```
1 """ Search For Folder and Access it using 2 Ways """  
2  
3 # 1- Converting result to a List  
4 found_path = list(script_directory.rglob('mypackages'))  
5 print(found_path) # [WindowsPath("d:/1_Embedded/My Repo's/Studying Python/mypackages")]  
6  
7 # 2- Using list comprehension to Loop on files and return them in list  
8 found_path = script_directory.rglob('mypackages')  
9 MyFolders = [folder for folder in found_path]  
10 print(MyFolders) # [WindowsPath("d:/1_Embedded/My Repo's/Studying Python/mypackages")]
```

Note:

```
1 rglob("*.c") = glob("**/*.c"):  
2 This is like calling Path.glob() with "**/" added in front of the given relative pattern
```

