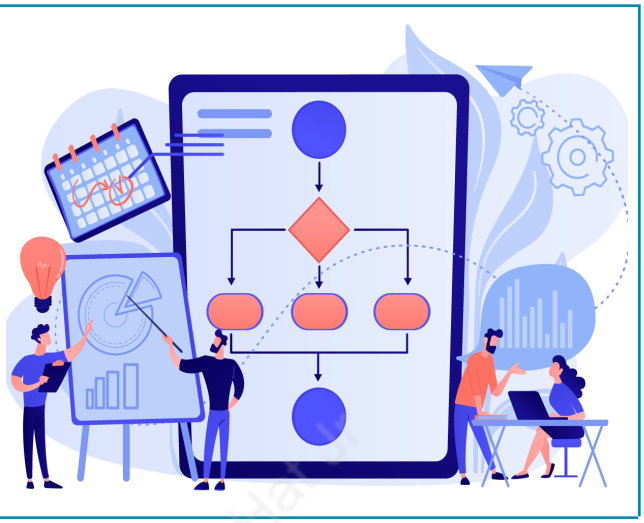
AUTOMATE FILE SEGREGATION



What is our GOAL for this MODULE?

In this class, we learned about the **os** and **shutil** modules of **Python** and also built a python program to move image files from one folder to another folder.

What did we ACHIEVE in the class TODAY?

- Understood the importance of **os** and **shutil** modules
- Use the **os** and **shutil** module to create a folder organizer program.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- The **pip install**
- The **os, shutil** modules
- The **while** loop

How did we DO the activities?

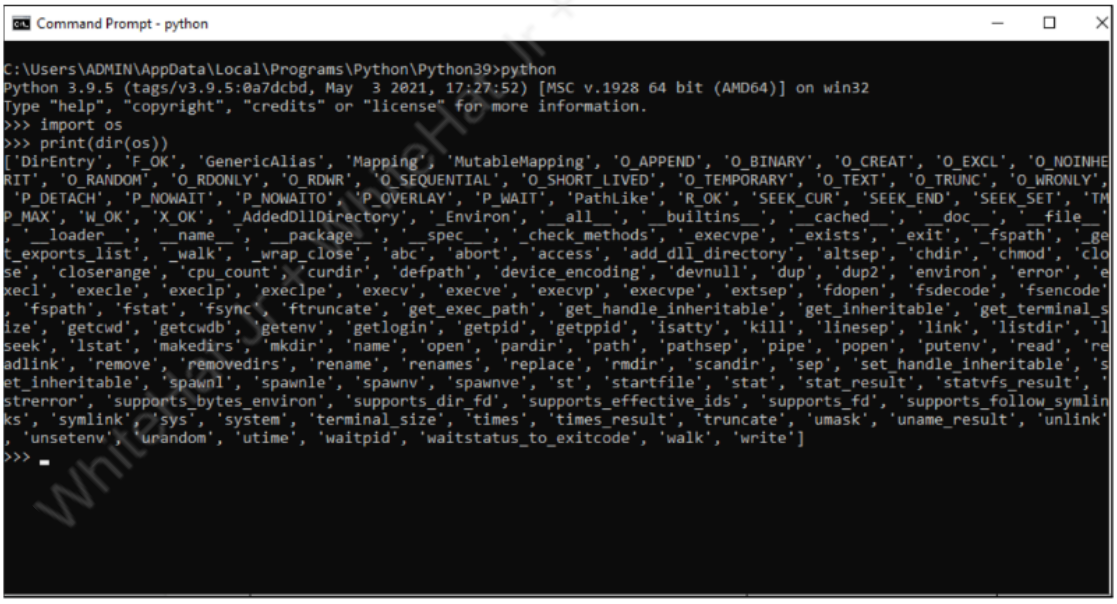
1. **OS module**: Our operating system provides us with different functionalities like moving files, copying files, and so on. The **os** module in Python provides us a way of using these operating system-dependent functionalities.

    - Import **os** module.

```
>>> import os
```

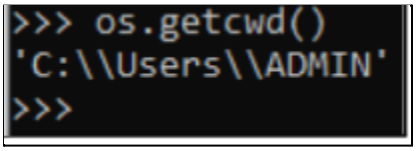
    - We can use **dir()** on the os module to see the methods of the os module.

```
>>> print(dir(os))
```



    - Check our current working directory by using **getcwd()** in the **os** module.

```
>>> os.getcwd()
```



- The **mkdir()** method in the os module is used to create new files or folders.

```
os.mkdir("name of the file/folder to create")
```

```
>>> os.mkdir("102")
```

- To get all the files and folders of the current directory, we can use the **listdir()** method in the **os** module.

```
>>> os.listdir()
```

```
>>> os.mkdir("102")
>>> os.listdir()
['.android', '.config', '.expo', '.idlerc', '.vscode', '102', '3D Objects',
Cookies', 'Desktop', 'Documents', 'Downloads', 'e-library', 'Favorites', 'Ir
gs', 'Music', 'My Documents', 'NetHood', 'node_modules', 'NTUSER.DAT', 'ntus
{53b39e88-18c4-11ea-a811-000d3aa4692b}.TM.blf', 'NTUSER.DAT{53b39e88-18c4-11
000000001.regtrans-ms', 'NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TM
'ntuser.ini', 'OneDrive', 'package-lock.json', 'Pictures', 'PrintHood', 'Rec
SlackSetup.exe', 'Start Menu', 'Templates', 'Videos']
>>>
```

- To check if a particular file or folder is present in a given folder path, we can use the **exists()** method in the **os.path** module.

```
os.path.exists("path of a folder/file)
```

```
>>> os.getcwd()
'C:\\Users\\ADMIN\\AppData\\Local\\Programs\\Python\\Python39'
>>> path = '/usr/local/bin/'
>>> isExist = os.path.exists(path)
>>> print(isExist)
False
>>> path = 'C:\\Users\\ADMIN\\AppData\\Local\\Programs\\Python\\Python39'
>>> isExist = os.path.exists(path)
>>> print(isExist)
True
>>>
```

- The **os.path.splitext()** method in **Python** is used to split the path name into the **(root, ext)** pair where **root** is the whole path except the extension and **ext** is the file extension with a dot.

```
root, ext = os.path.splitext("path of a folder/file)
```

```
>>> import os
>>> path = "Downloads/feather.jfif"
>>> root, extension = os.path.splitext(path)
>>> print("Root of the path : ", root)
Root of the path :  Downloads/feather
>>> print("Extension of the path : ", extension)
Extension of the path :  .jfif
>>>
```

2. The **shutil** module: This helps us in automating the process of copying and removal of files and directories.
   Steps to **copy** a file:

   - Take a **source** variable and assign a file path.

```
>>> source = "/Downloads/feather.jfif"
```

   - Take a **destination** variable, assign a path where we want to create a copy.

```
>>> destination = "/Downloads/copyfeature.jfif"
```

   - Use the **copy()** method in the **shutil** module to copy the file from source to destination.

```
shutil.copy(source, destination)
```

```
>>> dest =shutil.copy(source, destination)
```

- Print the content of the folder to see if the copy is created.

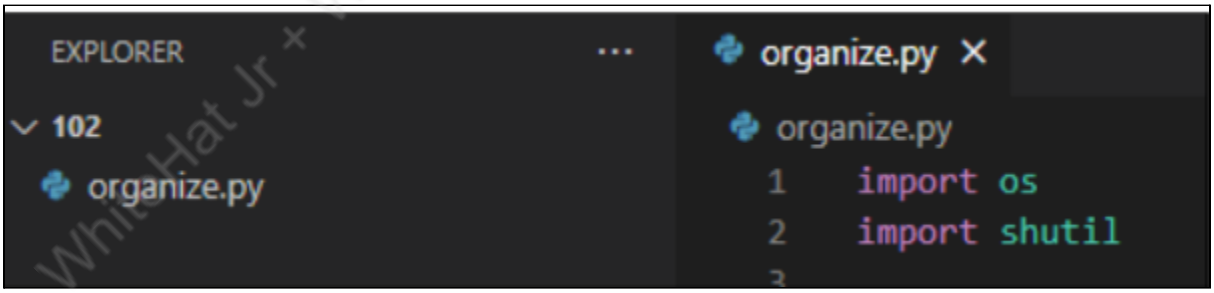
```
>>> print("After copying file:")

>>> print(os.listdir(path))
```

```
>>> source = "Downloads/feather.jfif"
>>> destination = "Downloads/copyfeature.jfif"
>>> dest = shutil.copy(source, destination)
>>> print("After copying file:")
After copying file:
>>> print(os.listdir(path))
['ChromeSetup.exe', 'copyfeature.jfif', 'desktop.ini', 'feather.jfif', 'flower.jfif', 'hunte
er-main.zip', 'PiratesInvasionStage-2-main', 'PiratesInvasionStage-2-main.zip', 'PRO-C175.do
iz-20.docx', 'Pro-Quiz-21.docx', 'Pro-Quiz-C22 (1).docx', 'Pro-Quiz-C22.docx', 'PRO_V3_C40_L
tup-x64-1.55.2.exe']
>>> _
```

- There are other methods of **shutil()** to move the files or remove files altogether.
- The **shutil.move()** to move a file from source to destination.

3. Creating file organizer with the **Python**
   - Create a folder named **C102** in your local system.
   - Open that folder in VSC.
   - Create a new file named **organize.py**.
   - Import **os** and **shutil** modules



- Take two variables, **from_dir** and **to_dir** to store **source** path and **destination** path, respectively.

```
from_dir = "C:/Users/ADMIN/Downloads"
to_dir = "C:/WhiteHatJr/dowanloadedimages"
```

- Take a variable, **list_of_files**, to store the names of all the files. Add **print(list_of_files)** and run the code.

```
from_dir = "C:/Users/ADMIN/Downloads"
to_dir = "C:/WhiteHatJr/dowanloadedimages"

list_of_files = os.listdir(from_dir)
print(list_of_files)
```

```
PS C:\WhiteHatJr\Python\102> py organise.py
['ChromeSetup.exe', 'desktop.ini', 'hunter-archer-main', 'hunter-archer-main.zip', 'PiratesInvasionStage-2-main', 'PiratesInvasionStage-2-main.zip', 'PRO-C175.docx', 'PRO-C23.docx', 'Pro-Quiz-20.docx', 'Pro-Quiz-21.docx', 'Pro-Quiz-C22 (1).docx', 'Pro-Quiz-C22.docx', 'PRO_V3_C40_LITE.docx.pdf', 'VSCodeUserSetup-x64-1.55.2.exe']
PS C:\WhiteHatJr\Python\102>
```

- Create a **for-in** loop to traverse through the **list_of_files**: Use **os.path.splitext()** on each file name to find the extension.

```
list_of_files = os.listdir(from_dir)
#print(list_of_files)


# Move All Image files from Downloads Folder to Another Folder
for file_name in list_of_files:

    name, extension = os.path.splitext(file_name)
    print(name)
    print(extension)
```

```
.zip
PRO-C175
.docx
PRO-C23
.docx
Pro-Quiz-20
.docx
Pro-Quiz-21
.docx
Pro-Quiz-C22 (1)
.docx
Pro-Quiz-C22
.docx
PRO_V3_C40_LITE.docx
.pdf
smile
.jfif
```

- Write an if condition to check if the extension is blank, if the condition is true then continue. If the extension is blank, it will jump to the next file and check for the extension of the next file.

```
if extension == '':
    continue
```

- Write another if condition to check if the extension of the image file in the path is one of the extensions in a list **['.gif', '.png', '.jpg', '.jpeg', '.jfif']**

```
if extension == '':
    continue
if extension in ['.gif', '.png', '.jpg', '.jpeg','.jfif']:
```

- If the extension is in the list **['.gif', '.png', '.jpg', '.jpeg','.jfif']**, then:
    - Create 3 variables for the name of the directory paths:
        1. Create **path1** as the name of the source path. Use string concatenation to merge from_dir+'/'+file_name
           Example **path1**: Downloads/ImageName1.jpg
        2. Create **path2** as we want to create a new folder with that extension name and move the files to that folder. Use string concatenation to merge to_dir+ '/'+ "Image_Files"
           Example **path2**: D:/My Files/Image_Files
        3. Create **path3** to assign the destination path with the same file name as the source. Use string concatenation to merge to_dir + '/' +"Image_Files" + '/' + file_name
           Example **path3**: D:/MyFiles/Image_Files/ImageName1.jpg

```
path1 = from_dir + '/' + file_name
path2 = to_dir + '/' + "Image_Files"
path3 = to_dir + '/' + "Image_Files" + '/' + file_name
print("path1 " , path1)
print("path3 ", path3)
```

```
PROBLEMS   OUTPUT   TERMINAL   DEBUG CONSOLE

PS C:\WhiteHatJr\Python\102> py organise.py
path1  C:/Users/ADMIN/Downloads/bus.jfif
path3  C:/WhiteHatJr/dowanloadedimages/Image_Files/bus.jfif
path1  C:/Users/ADMIN/Downloads/copyfeature.jfif
path3  C:/WhiteHatJr/dowanloadedimages/Image_Files/copyfeature.jfif
path1  C:/Users/ADMIN/Downloads/feather.jfif
path3  C:/WhiteHatJr/dowanloadedimages/Image_Files/feather.jfif
path1  C:/Users/ADMIN/Downloads/flower.jfif
path3  C:/WhiteHatJr/dowanloadedimages/Image_Files/flower.jfif
path1  C:/Users/ADMIN/Downloads/mushroom-house.jpg
path3  C:/WhiteHatJr/dowanloadedimages/Image_Files/mushroom-house.jpg
```

- Before moving any files, we should check if the directory/folder is present or not.

1. Check if the folder/directory path exists before moving using an if condition:
    a. Create a condition to check if the destination path exists at **path2**.
    b. If true, use **print()** method to print a message moving with the file name which is being moved.
    c. Use **shutil.move(path1, path3)**.In this case, **path1** is the source path and **path3** is the destination path.
2. Else make a new folder/directory then move:
    a. Use **os.makedirs()** to create **path2**.
    b. Use **print()** method to print a message moving with the file name.
    c. Use **shutil.move(path1, path3)**

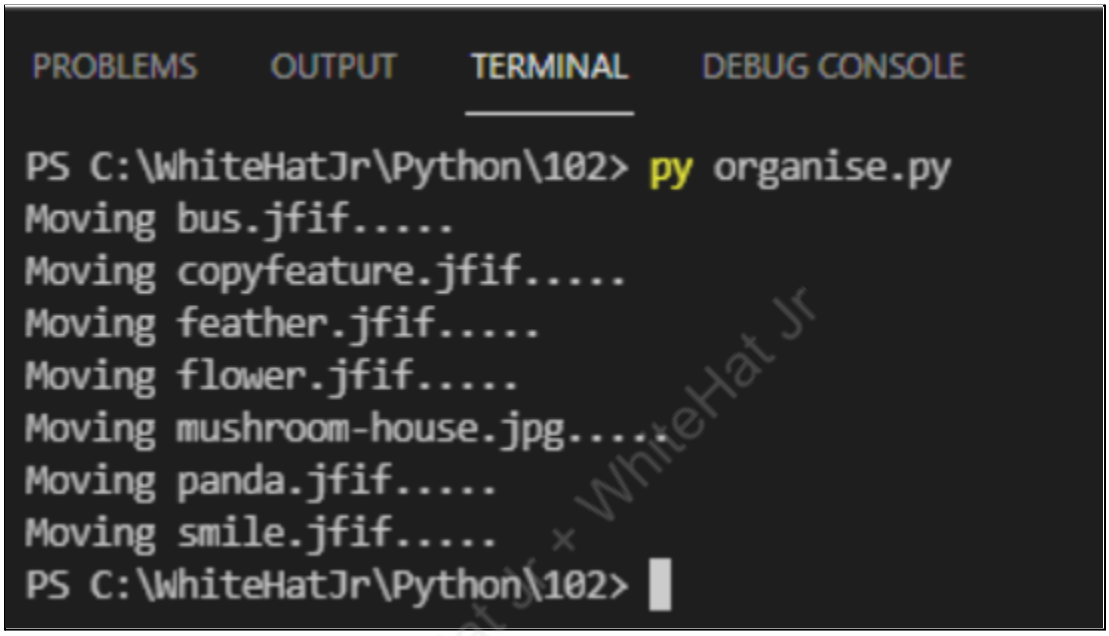
```
# Check if Folder/Directory Path Exists Before Moving
# Else make a NEW Folder/Directory Then Move
if os.path.exists(path2):
    print("Moving " + file_name + ".....")

    # Move from path1 ---> path3
    shutil.move(path1, path3)

else:
    os.makedirs(path2)
    print("Moving " + file_name + ".....")
    shutil.move(path1, path3)
```

4. Run the code to check the output.

OUTPUT:



What's next?

In the next class, you will be introduced to the **watchdog** module to manage file system creation events. We will automate "Download" folder/files movement as soon as the file is downloaded, and segregate it into different directories based on the type of the file.

EXTEND YOUR KNOWLEDGE:

Learn more about the **os** module here