Renaming & Deleting File

Overview

- File handling in Python doesn't end with reading and writing files — you often need to **rename, delete, or organize** files and directories.
- Python does **not** provide any function/module to rename or delete file.
- So we use Python's built-in **os** and **shutil** modules to perform these operations.

Renaming File

Renaming File – os.rename()

- To rename a file, Python provides the **os.rename()** function.
- This function changes the name of an **existing file** to a **new one**.
- **Syntax:**

Syntax
`import os` `os.rename("oldfilename", "newfilename")`

Note: If the target file already exists, Python will overwrite it silently, so use carefully.

Renaming File – os.rename()

- **Example**

Python

```
import os
os.rename("oldData.txt", "newData.txt")
print("File renamed successfully!")
```

Tip:

You can use **os.path.exists("filename")** before renaming to check whether the file actually exists, avoiding runtime errors.

Deleting File

Deleting File – os.remove()

- Python provides **os.remove()** to delete files permanently.
- Before deleting, it's a good practice to **verify** the file's existence.

Syntax

```
import os
os.remove("fileName")
```

Deleting File – os.remove()

- Example

Python

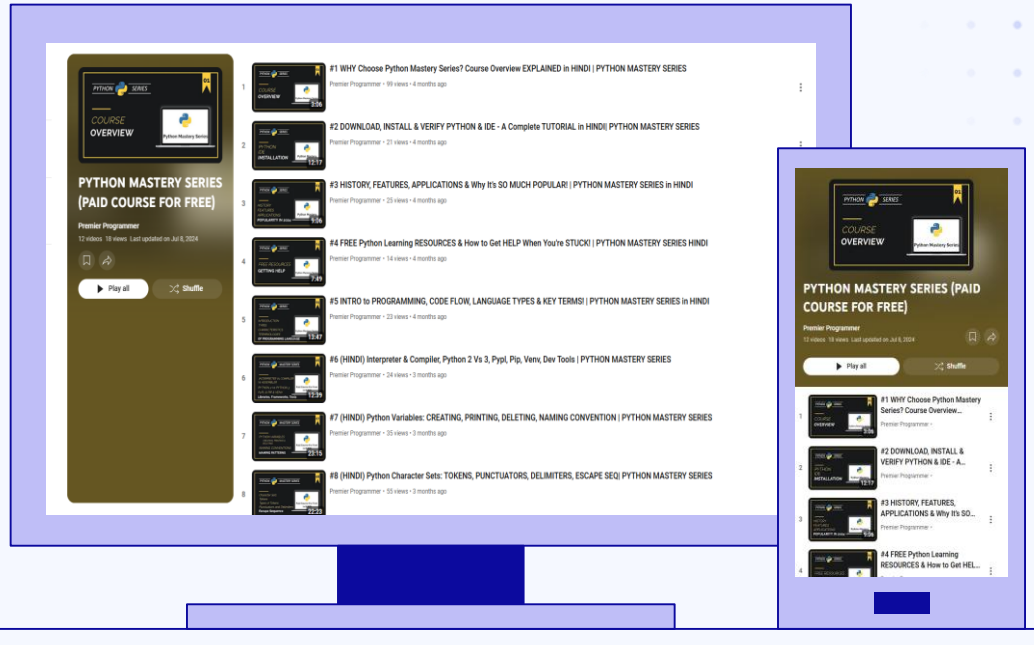
```python
import os
if os.path.exists("newData.txt"):
    os.remove("newData.txt")
    print("File deleted successfully!")
else:
    print("File does not exist.")
```

Important Notes:

- Once deleted, the file **cannot be recovered**.
- Attempting to delete a **non-existing** file will raise a **FileNotFoundError**.
- For deleting directories, you **cannot** use **os.remove()** — it's only for files.

WATCH

Level up your coding with each episode in this focused Python series.



Next Video!

Working with Directory