Работа с файлами в Python с помощью модуля OS

Обработка файлов в Python с помощью модуля os включает создание, переименование, перемещение, удаление файлов и папок, а также получение списка всех файлов и каталогов и многое другое.

Модуль встроенный, поэтому для работы с ним не нужно ничего устанавливать.

Вывод текущей директории

Для получения текущего рабочего каталога используется os.getcwd():

```
import os

# вывести текущую директорию
print("Текущая директория:", os.getcwd())
```

os.getcwd() возвращает строку в Юникоде, представляющую текущий рабочий каталог.

Создание папки

Для создания папки/каталога в любой операционной системе нужна следующая команда:

```
# создать пустой каталог (папку)
os.mkdir("folder")
```

После ее выполнения в текущем рабочем каталоге тут же появится новая папка с названием «folder».

Если запустить ее еще раз, будет вызвана ошибка FileExistsError, потому что такая папка уже есть. Для решения проблемы нужно запускать команду только в том случае, если каталога с таким же именем нет. Этого можно добиться следующим образом:

```
# повторный запуск mkdir с тем же именем вызывает FileExistsError,
# вместо этого запустите:
if not os.path.isdir("folder"):
     os.mkdir("folder")
```

Функция os.path.isdir() вернет True, если переданное имя ссылается на существующий каталог.

Изменение директории

```
# изменение текущего каталога на 'folder'
os.chdir("folder")

Еще раз выведем рабочий каталог:

# вывод текущей папки
print("Текущая директория изменилась на folder:", os.getcwd())
```

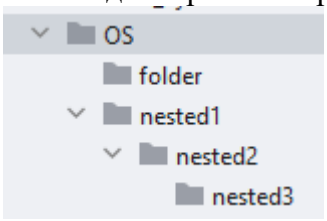
Создание вложенных папок

Предположим, вы хотите создать не только одну папку, но и несколько вложенных:

```
# вернуться в предыдущую директорию
os.chdir("..")
```

```
# сделать несколько вложенных папок
os.makedirs("nested1/nested2/nested3")
```
!!! Внимание!!! После выполнения – закомментировать этот код.

Это создаст три папки рекурсивно, как показано на следующем изображении:



Для дальнейшей работы создадим файл

```
# создать новый текстовый файл
text_file = open("text.txt", "w")
# запись текста в этот файл
text_file.write("Это текстовый файл")
text_file.close()
```
!!! Внимание!!! После выполнения – закомментировать этот код.

Переименование файлов

```
# переименовать text.txt на renamed-text.txt
os.rename("text.txt", "renamed-text.txt")
```
!!! Внимание!!! После выполнения – закомментировать этот код.

Функция os.rename() принимает 2 аргумента: имя файла или папки, которые нужно переименовать и новое имя.

Перемещение файлов

Функцию os.replace() можно использовать для перемещения файлов или каталогов:

```
# заменить (переместить) этот файл в другой каталог
os.replace("renamed-text.txt", "folder/renamed-text.txt")
```
!!! Внимание!!! После выполнения – закомментировать этот код.

Стоит обратить внимание, что это перезапишет путь, поэтому если в папке folder уже есть файл с таким же именем (renamed-text.txt), он будет перезаписан.

Список файлов и директорий

```
# распечатать все файлы и папки в текущем каталоге
print("Все папки и файлы:", os.listdir())
```

Функция os.listdir() возвращает список, который содержит имена файлов в папке. Если в качестве аргумента не указывать ничего, вернется список файлов и папок текущего рабочего каталога.

Если нужно узнать состав папок, то используем функцию os.walk():

```
# распечатать все файлы и папки рекурсивно
for dirpath, dirnames, filenames in os.walk("."):
    # перебрать каталоги
    for dirname in dirnames:
```

```
        print("Каталог:", os.path.join(dirpath, dirname))
    # перебрать файлы
    for filename in filenames:
        print("Файл:", os.path.join(dirpath, filename))
```

os.walk() — это генератор дерева каталогов. Он будет перебирать все переданные составляющие. Здесь в качестве аргумента передано значение «.», которое обозначает верхушку дерева.

Метод os.path.join() был использован для объединения текущего пути с именем файла/папки.

Удаление файлов

```
# удалить созданный файл
os.remove("folder/renamed-text.txt")
```

!!! Внимание!!! После выполнения – закомментировать этот код.

os.remove() удалит файл с указанным именем (не каталог).

Удаление директорий

```
# удалить папку
os.rmdir("folder")
```

!!! Внимание!!! После выполнения – закомментировать этот код.

Для удаления каталогов рекурсивно необходимо использовать os.removedirs(). Это удалит только пустые каталоги.

```
# удалить вложенные папки
os.removedirs("nested1/nested2/nested3")
```

!!! Внимание!!! После выполнения – закомментировать этот код.

Получение информации о файлах

Для получения информации о файле в ОС используется функция os.stat(), которая выполняет системный вызов stat() по выбранному пути:

```
# вывести некоторые данные о файле
print(os.stat("text.txt"))
```

Вернется кортеж с отдельными метриками. В их числе есть следующие:

st_size — размер файла в байтах
st_atime — время последнего доступа в секундах (временная метка)
st_mtime — время последнего изменения
st_ctime — в Windows это время создания файла, а в Linux — последнего изменения метаданных

Для получения конкретного атрибута нужно писать следующим образом:

```
# например, получить размер файла
print("Размер файла:", os.stat("text.txt").st_size)
```