IT CAREER HUB

Урок 26.

Работа с файловой системой

IT CAREER HUB

Введение в файловую систему

Файловая система – это способ хранения и организации данных на диске компьютера. Она представляет собой структуру, состоящую из файлов и папок (директорий), которые можно читать, изменять, удалять или перемещать.

Пример структуры файловой системы

```
/home/user/
├── PycharmProjects/
│   ├── project1/
│   │   ├── data
│   │   │   └── file.txt
│   │   └── main.py
│   └── project2/
│       └── main.py
├── documents/
└── downloads/
```

IT CAREER HUB

Абсолютные и относительные пути

Абсолютный путь

Абсолютный путь – это полный путь от корневой директории до файла или папки. Он указывает точное местоположение объекта в файловой системе.

Пример

Абсолютный путь в Linux/macOS начинается с / (корень файловой системы)

```python
absolute_path = "/home/user/documents/file.txt"
```

Абсолютный путь в Windows начинается с буквы диска (например, C:\\)

```python
absolute_path = "C:\\Users\\User\\Documents\\file.txt"
```

Относительный путь

Относительный путь – это путь к файлу или папке относительно текущей рабочей директории. Он не содержит полного пути до корневой директории.

- Для текущей директории используется . (точка).
- Для родительской директории используется .. (две точки).

Пример

Если текущая директория `/home/user/PycharmProjects/project1/data`:

```python
./file.txt

#указывает на файл /home/user/PycharmProjects/project1/data/file.txt

../file.txt

#указывает на файл /home/user/PycharmProjects/project1/file.txt
```

Когда использовать

Абсолютный путь	**Относительный путь**
Когда нужно работать с файлами независимо от текущей директории. Когда путь к файлу фиксирован (например, конфигурационные файлы).	Когда работа с файлами локализована в пределах текущей директории. Для перемещаемых скриптов.

IT CAREER HUB

Специальные обозначения для пути

При работе с файловой системой в относительных путях используются специальные обозначения, упрощающие навигацию.

Основные обозначения

Знак	Значение	Описание	Пример
.	Текущая директория	Указывает на текущую директорию, где находится программа во время выполнения. Используется для построения относительных путей	./file.txt файл file.txt в текущей директории
..	Родительская директория	Указывает на директорию на уровень выше (родительскую)	../file.txt файл file.txt в родительской директории ../../file.txt файл в директории на два уровня выше
/	Разделитель в пути	В Linux и macOS используется для указания вложенных директорий	/home/user/file.txt абсолютный путь
\\	Разделитель в пути	Только в Windows	C:\\Users\\User\\Documents\\file.txt абсолютный путь
~	Домашняя директория	Указывает на домашнюю директорию пользователя (Linux и macOS)	~/documents/file.txt путь к файлу в директории documents домашней папки

IT CAREER HUB

Модуль os

Модуль os предоставляет инструменты для работы с операционной системой, включая файловую систему.

Для работы с функциями модуля os необходимо его импортировать:

Синтаксис

```python
import os
```

IT CAREER HUB

Навигация по файловой системе

Получение текущей директории

Для определения текущей рабочей директории используется функция `os.getcwd()`.

Пример

Python

```python
import os

# Получаем текущую рабочую директорию
current_dir = os.getcwd()
print(f"Текущая директория: {current_dir}")
```

IT CAREER HUB

Изменение текущей директории

Для перехода в другую директорию используется функция `os.chdir(path)`.

Пример

Python

```python
import os

# Переход в директорию с использованием относительного пути
os.chdir("./subdirectory")  # Ошибка, если директории нет!
print(f"Текущая директория: {os.getcwd()}")

# Переход в родительскую директорию
os.chdir("..")
print(f"Текущая директория после перехода: {os.getcwd()}")

# Переход в директорию с использованием абсолютного пути
os.chdir("/home/user/documents")  # Замените путь на существующий
print(f"Текущая директория: {os.getcwd()}")
```

IT CAREER HUB

Работа с файлами и директориями

Проверка существования

Для проверки существования файла или директории используется функция `os.path.exists(path)`.

Пример

Python

```python
import os

# Проверка существования файла
file_path = "example.txt"
# или
# file_path = "example_folder/example.txt"
if os.path.exists(file_path):
    print(f"Файл '{file_path}' существует.")
else:
    print(f"Файл '{file_path}' не найден.")

# Проверка существования директории
directory_path = "example_folder"
# или
# directory_path = "parent_folder/child_folder"
if os.path.exists(directory_path):
    print(f"Директория '{directory_path}' существует.")
else:
    print(f"Директория '{directory_path}' не найдена.")
```

Получение списка файлов и папок в директории

Функция `os.listdir(path)` возвращает список файлов и папок в указанной директории.

Пример

```python
import os

# Список содержимого текущей директории
contents = os.listdir(".")
print("Содержимое текущей директории:", contents)

# Список содержимого указанной директории
specific_dir = "parent_folder"
if os.path.exists(specific_dir):
    print(f"Содержимое директории '{specific_dir}':", os.listdir(specific_dir))
```

Создание новой директории

`os.mkdir(path)`	`os.makedirs(path)`
Создаёт одну директорию. Вызывает ошибку, если директория уже существует.	Создаёт директорию вместе с родительскими, если их ещё нет. Вызывает ошибку, если директория уже существует, но с использованием параметра exist_ok=True ошибка не возникает.

Пример

```python
import os

dir_path = "example_folder"
if not os.path.exists(dir_path):
    # Создание одной директории
    os.mkdir("example_folder")
    print(f"Директория '{dir_path}' создана.")
else:
    print(f"Директория '{dir_path}' существует.")

# Создание вложенных директорий
os.makedirs("parent_folder/child_folder", exist_ok=True)
print("Вложенные директории 'parent_folder/child_folder' созданы.")
```

Создание файла

Создание файла осуществляется с использованием функции `open(path, mode)`, которая не является частью модуля `os`. Подробности работы с файлами будут рассмотрены позже.

Пример

```python
# Создание нового пустого файла
file_name = "example.txt"
open(file_name, "w").close()
print(f"Файл '{file_name}' создан или перезаписан.")
```

Определение типа объекта: файл или нет

Функция `os.path.isfile(path)` используется для проверки, является ли указанный путь файлом.

IT CAREER HUB

Пример

Python

```python
import os


file_path = "example.txt"
# Проверяем, существует ли объект
if os.path.exists(file_path):
    # Проверяем, является ли файлом
    if os.path.isfile(file_path):
        print(f"'{file_path}' существует и это файл.")
    else:
        print(f"'{file_path}' существует, но это не файл.")
else:
    print(f"'{file_path}' не существует.")
```

IT CAREER HUB

Проверка типа объекта: файл или директория

Помимо `os.path.isfile` есть функция `os.path.isdir`, которая позволяет проверить, является ли путь директорией.

Пример

Python

```python
import os

path = "example.txt"
# path = "parent_folder"

# Проверяем, является ли файлом
if os.path.isfile(path):
    print(f"'{path}' существует и это файл.")
# Проверяем, является ли директорией
elif os.path.isdir(path):
    print(f"'{path}' существует и это директория.")
```

IT CAREER HUB

Переименование

Для переименования файлов или директорий используется функция `os.rename(src, dst)`.

Пример

Python

```python
import os


# Переименование файла
old_file_name = "example.txt"
new_file_name = "renamed.txt"
if os.path.exists(old_file_name):
    os.rename(old_file_name, new_file_name)
    print(f"Файл переименован в '{new_file_name}'.")
else:
    print(f"Файл '{old_file_name}' не найден.")

# Переименование директории
old_dir_name = "example_folder"
new_dir_name = "renamed_folder"
if os.path.exists(old_dir_name):
    os.rename(old_dir_name, new_dir_name)
    print(f"Директория переименована в '{new_dir_name}'.")
else:
    print(f"Директория '{old_dir_name}' не найдена.")
```

Удаление

Для удаления файлов используется функция `os.remove(path)`, а для удаления пустых директорий – `os.rmdir(path)`.

Пример

Python

```python
import os


# Удаление файла
file_to_delete = "renamed.txt"
if os.path.exists(file_to_delete):
    os.remove(file_to_delete)
    print(f"Файл '{file_to_delete}' удалён.")
else:
    print(f"Файл '{file_to_delete}' не найден, удаление невозможно.")

# Удаление пустой директории
empty_dir_to_delete = "renamed_folder"
if os.path.exists(empty_dir_to_delete):
    os.rmdir(empty_dir_to_delete)
    print(f"Пустая директория '{empty_dir_to_delete}' удалена.")
else:
    print(f"Директория '{empty_dir_to_delete}' не найдена или не пуста.")
```

IT CAREER HUB

Задания для закрепления 1

1. **Что делает `os.getcwd()`?**

a) Задает текущую рабочую директорию
b) Получает текущую рабочую директорию
c) Создаёт новую директорию
d) Удаляет текущую директорию

2. **Какой путь является абсолютным?**

a) `C:\\Users\\Admin\\Documents\\file.txt`
b) `./Documents/file.txt`
c) `../file.txt`
d) `Documents/file.txt`

Посмотреть ответы

IT CAREER HUB

Работа с путями

Разделение пути на компоненты

Функция	Назначение
`os.path.split(path)`	Разделяет путь на две части: директорию и имя файла
`os.path.basename(path)`	Возвращает только имя файла или папки
`os.path.dirname(path)`	Возвращает только директорию без имени файла

Пример

Python

```python
import os

path = "/subdirectory/example.txt"

# Разделение пути на директорию и файл
directory, file = os.path.split(path)
print(f"Директория: {directory}, Файл: {file}")

# Получение только имени файла
print(f"Имя файла: {os.path.basename(path)}")

# Получение только директории
print(f"Директория: {os.path.dirname(path)}")
```

Получение абсолютного пути

Функция `os.path.abspath(path)` преобразует относительный путь в абсолютный.

Пример

Python

```python
import os


# Относительный путь
relative_path = "example.txt"

# Преобразование в абсолютный путь
absolute_path = os.path.abspath(relative_path)
print(f"Абсолютный путь: {absolute_path}")
```

Соединение путей

Для объединения нескольких компонентов пути к файлам и директориям используется `os.path.join()`.

Пример

Python

```python
import os


# Объединение нескольких компонентов
current_dir = os.getcwd()
sub_dir = "docs"
file_name = "data.txt"
full_path = os.path.join(current_dir, sub_dir, file_name)
print(f"Путь: {full_path}")
```

Обход директории и её содержимого

Функция `os.walk` позволяет рекурсивно обойти директорию и её содержимое, включая файлы и поддиректории. Это удобный инструмент для работы со сложными структурами файловой системы.

Пример

Python

```python
import os

# Рекурсивный обход директории
for root, dirs, files in os.walk("."):
    print(f"Текущая директория: {root}")
    print(f"Поддиректории: {dirs}")
    print(f"Файлы: {files}")
    print("-" * 50)
```

Пример использования: Поиск файлов с определённым расширением

Python

```python
import os


# Поиск всех .txt файлов
for root, dirs, files in os.walk("."):
    for file in files:
        if file.endswith(".txt"):
            print(f"Найден файл: {os.path.join(root, file)}")
```

IT CAREER HUB

Задание для закрепления 2

1. **Какой результат будет выведен при выполнении следующего кода?**

```python
import os


path = "/home/user/docs/file.txt"
print(os.path.dirname(path))
```

a) `/home/user/docs/file.txt`
b) `/home/user/docs`
c) `file.txt`
d) `docs/file.txt`

2. **Какой метод используется для рекурсивного обхода файлов и папок в директории?**

a) `os.listdir()`
b) `os.walk()`
c) `os.recursion_walk()`
d) `os.scan()`

Посмотреть ответы

IT CAREER HUB

Модуль sys

Модуль sys предоставляет доступ к функциям и переменным, которые используются и поддерживаются интерпретатором Python. Этот модуль позволяет взаимодействовать с окружением Python, управлять аргументами командной строки, путями поиска модулей, стандартным вводом и выводом.

Для работы с функциями модуля sys необходимо его импортировать:

Синтаксис

```python
import sys
```

IT CAREER HUB

Аргументы командной строки

Скрипт – исполняемый файл с кодом.

Аргументы командной строки

sys.argv — это список, который содержит аргументы, переданные скрипту при его запуске из командной строки.

- Первый элемент списка (`sys.argv[0]`) — это имя скрипта.
- Остальные элементы — это аргументы, переданные после имени скрипта.

Синтаксис

Python

```python
import sys

# Доступ к аргументам командной строки
arguments = sys.argv
```

Пример

Python

```python
import sys

# Вывод всех аргументов командной строки
print("Все аргументы:", sys.argv)
# Работа с первым аргументом (если он передан)
if len(sys.argv) > 1:
    print(f"Переданный аргумент: {sys.argv[1]}")
else:
    print("Аргументы не переданы.")
```

Запуск python файла через консоль

Python

```
# Замените script.py на название вашего файла
python script.py argument1
```

Вывод

```
(.venv) tanya@tanya-nitro:~/PycharmProjects/pythonProgramItch$ python script.py argument1
Все аргументы: ['script.py', 'argument1']
Переданный аргумент: argument1
```

Пример: Обработка нескольких аргументов

Python

```
import sys

if len(sys.argv) > 1:
    for i, arg in enumerate(sys.argv[1:], start=1):
        print(f"Аргумент {i}: {arg}")
else:
    print("Аргументы не переданы.")
```

Запуск python файла через консоль

Python

```
python script.py arg1 arg2 arg3
```

IT CAREER HUB

Практическое применение sys.argv

Аргументы командной строки часто используются для передачи данных в скрипт, делая программу более гибкой.

Пример: Удаление файлов с определённым расширением

Скрипт принимает расширение и удаляет все файлы с этим расширением в текущей директории.

Python

```python
import sys
import os

# Проверяем, передан ли аргумент
if len(sys.argv) != 2:
    print("Использование: python script.py <расширение>")
    sys.exit(1)

extension = sys.argv[1]

# Удаляем файлы с указанным расширением
for file_name in os.listdir("."):
    if file_name.endswith(extension):
        os.remove(file_name)
        print(f"Удалён файл: {file_name}")
```

Запуск python файла через консоль

Python

```
python script.py .txt
```

IT CAREER HUB

Пример: Вывод содержимого директории

Скрипт принимает директорию как аргумент и выводит её содержимое.

Python

```python
import sys
import os

# Проверяем, передан ли аргумент
if len(sys.argv) != 2:
    print("Использование: python script.py <директория>")
    sys.exit(1)

directory = sys.argv[1]

# Проверяем существование директории
if not os.path.isdir(directory):
    print(f"Директория '{directory}' не существует.")
    sys.exit(1)

# Выводим содержимое директории
print(f"Содержимое директории '{directory}':")
for item in os.listdir(directory):
    print(f"- {item}")
```

Запуск python файла через консоль

Python

```
python script.py /home/user/documents
```

IT CAREER HUB

Пример: Разная конфигурация логирования

Этот скрипт демонстрирует различное поведение в зависимости от режима (debug или release): в режиме отладки программа выводит дополнительную информацию, а в рабочем режиме — только ключевые события.

Python

```python
import sys
import logging

# Проверяем количество аргументов
if len(sys.argv) != 2:
    print("Использование: python script.py <режим>")
    print("<режим> - debug или release")
    sys.exit(1)

# Получаем режим работы
mode = sys.argv[1]

# Проверяем корректность режима
if mode not in ["debug", "release"]:
    print("Ошибка: режим должен быть 'debug' или 'release'.")
    sys.exit(1)

# Настройка логирования
log_level = logging.DEBUG if mode == "debug" else logging.INFO
logging.basicConfig(
    format="%(asctime)s - %(levelname)s - %(message)s",
    level=log_level
)

# Пример событий во время выполнения программы
logging.info("Инициализация программы")
logging.debug("Чтение конфигурации")
logging.debug("Подключение к базе данных")
logging.info("Обработка данных")
logging.critical("Критическая ошибка: соединение с базой данных потеряно!")
logging.info("Завершение работы программы")
```

IT CAREER HUB

Запуск в режиме отладки

Python

```
python script.py debug
```

Запуск в рабочем режиме

Python

```
python script.py release
```

IT CAREER HUB

Задание для закрепления 3

Какие утверждения о sys.argv верны?

a) `sys.argv` — это список строк
b) `sys.argv` содержит только числовые значения
c) `sys.argv` всегда содержит ровно два элемента
d) Первый элемент `sys.argv[0]` — это имя запущенного скрипта

Посмотреть ответ

IT CAREER HUB

Ответы на задания

Задания на закрепление 1	Вернуться к заданиям
1. Метод `os.getcwd()`	Ответ: b
2. Абсолютный путь	Ответ: a
Задания на закрепление 2	Вернуться к заданиям
1. Результат выполнения кода	Ответ: b
2. Метод для рекурсивного обхода	Ответ: b
Задания на закрепление 3	Вернуться к заданиям
утверждения о `sys.argv`	Ответ: a, d

IT CAREER HUB

🔎 Практическая работа

1. Определение типа объекта

Напишите программу, которая:

- Запрашивает у пользователя путь.
- Определяет, является ли путь файлом, папкой или он не существует.

Пример ввода

```python
Введите путь: example.txt
```

Пример вывода

```python
example.txt — это файл.
```

IT CAREER HUB

Решение

Python

```python
import os

file_name = input("Введите имя файла: ")

found = False
for root, _, files in os.walk("."):
    if file_name in files:
        print(f"Файл найден: {os.path.join(root, file_name)}")
        found = True
        break

if not found:
    print(f"Файл {file_name} не найден.")
```

IT CAREER HUB

2. Файлы с заданным расширением

Напишите программу, которая:

- Принимает путь к директории и расширение файлов через аргументы командной строки.
- Ищет файлы с указанным расширением в указанной директории.
- Выводит список найденных файлов.

Пример запуска

Python

```
python script.py /home/user/projects .py
```

Пример вывода

Python

```
Найденные файлы в директории '/home/user/projects': script.py, test.py, main.py
```

IT CAREER HUB

Решение

Python

```
import os
import sys

# Проверяем переданные аргументы
if len(sys.argv) != 2:
    print("Использование: python script.py <путь_к_директории>")
    sys.exit(1)

directory = sys.argv[1]

if os.path.exists(directory) and os.path.isdir(directory):
    empty_dirs = []
    for d in os.listdir(directory):
        if os.path.isdir(os.path.join(directory, d)) and not
os.listdir(os.path.join(directory, d)):
            empty_dirs.append(d)

    for d in empty_dirs:
        os.rmdir(os.path.join(directory, d))

    print(f"Удалены пустые папки: {', '.join(os.path.join(directory, d)
for d in empty_dirs)}" if empty_dirs else "Пустых папок не найдено.")
else:

    print(f"Директория {directory} не найдена.")
```