

Hướng dẫn đọc ghi file trong Python

Giáo viên: Nguyễn Hùng Cường

Python cung cấp các phương thức để có thể thao tác với các tập tin, ta có thể thực hiện việc thao tác với tập tin bằng cách sử dụng một đối tượng file.

Trước khi có thể đọc hoặc ghi file, ta cần mở file bằng cách sử dụng một hàm `open()` được cung cấp bởi Python. Hàm này cho phép tạo ra một đối tượng file.

Cú pháp:

```
file object = open(file_name [, access_mode][, buffering])
```

Ý nghĩa của các tham số:

- `file_name`: Là một chuỗi chứa tên tập tin mà ta muốn mở.
- `access_mode`: Là chế độ của file được mở, VD đọc, ghi, ghi nối v.v... Tham số này là tùy chọn và chế độ mặc định là đọc (`r`).
- `buffering`: Dùng để thiết lập bộ đệm. Nếu giá trị được thiết lập là 0 thì không sử dụng bộ nhớ đệm. Nếu giá trị là 1 thì thiết lập line buffering. Nếu giá trị là số <0 thì thiết lập kích thước bộ đệm mặc định.

Ý nghĩa của các chế độ mở file:

- `r`: Mở một file theo chế độ chỉ đọc. Con trỏ file (file pointer) được đặt ở đầu file. Đây là chế độ mặc định.
- `r`: Mở một file nhị phân theo chế độ chỉ đọc. Con trỏ file (file pointer) được đặt ở đầu file. Đây là chế độ mặc định.
- `r+`: Mở một file theo chế độ đọc và ghi. Con trỏ file (file pointer) được đặt ở đầu file.
- `r+`: Mở một file theo chế độ đọc và ghi trong định dạng nhị phân. Con trỏ file (file pointer) được đặt ở đầu file.
- `w`: Mở một file theo chế độ chỉ ghi. Ghi đè file nếu file đã tồn tại. Nếu file chưa tồn tại thì sẽ tạo mới một file để ghi.
- `wb`: Mở một file theo chế độ chỉ ghi trong định dạng nhị phân. Ghi đè file nếu file đã tồn tại. Nếu file chưa tồn tại thì sẽ tạo mới một file để ghi.
- `w+`: Mở một file để đọc và ghi. Ghi đè file nếu file đã tồn tại. Nếu file chưa tồn tại thì sẽ tạo mới một file để đọc và ghi.
- `wb+`: Mở một file để đọc và ghi trong định dạng nhị phân. Ghi đè file nếu file đã tồn tại. Nếu

file chưa tồn tại thì sẽ tạo mới một file để đọc và ghi.

- a: Mở một file để ghi nối. Con trỏ file sẽ được đặt ở cuối file nếu file đã tồn tại. Nếu file chưa tồn tại, tạo mới một file để ghi.

- ab: Mở một file để ghi nối trong định dạng nhị phân. Con trỏ file sẽ được đặt ở cuối file nếu file đã tồn tại. Nếu file chưa tồn tại, tạo mới một file để ghi.

- a+: Mở một file để ghi nối và đọc. Con trỏ file sẽ được đặt ở cuối file nếu file đã tồn tại. Nếu file chưa tồn tại, tạo mới một file để đọc và ghi.

- ab+: Mở một file để ghi nối và đọc trong định dạng nhị phân. Con trỏ file sẽ được đặt ở cuối file nếu file đã tồn tại. Nếu file chưa tồn tại, tạo mới một file để đọc và ghi.

Các thuộc tính của đối tượng file

Ngay khi một file đã được mở và một đối tượng file được tạo ra, ta có thể lấy về các thông tin liên quan đến file thông qua các thuộc tính của đối tượng file.

- file.closed: Trả lại true nếu file đã được đóng, nếu file vẫn còn mở thì trả lại false.

- file.mode: Trả lại chế độ truy nhập của file.

- file.name: Trả về tên của file.

Một số phương thức phổ biến khi làm việc với file trong Python:

Phương thức close(): Đóng tập tin đang được mở.

Chú ý: Python tự động đóng file khi đối tượng tham chiếu đến file được gán cho file khác.

Phương thức write()

Dùng để ghi một chuỗi vào tập tin đang được mở.

Chuỗi có thể có cả dữ liệu nhị phân, không chỉ có văn bản.

Phương thức write() sẽ không tự thêm ký tự xuống dòng vào cuối chuỗi.

Cú pháp:

```
fileObject.write(string);
```

Phương thức read()

Cho phép đọc một chuỗi từ một file đang mở. Khi gọi phương thức này, ta cần truyền vào tham số là số các byte được đọc từ file.

Cú pháp:

```
fileObject.read([count]);
```

Phương thức `tell()`: Lấy về vị trí hiện tại của con trỏ file.

Phương thức `seek()`: Thay đổi vị trí hiện tại của con trỏ file.

Phương thức `rename()`: Dùng để đổi tên file. Phương thức này nhận 2 tham số là tên file hiện tại và tên mới.

Cú pháp:

```
os.rename(current_file_name, new_file_name)
```

Phương thức `remove()`: Dùng để xóa file. Ta cần truyền vào tên file muốn xóa.

Cú pháp:

```
os.remove(file_name)
```

Thao tác với thư mục trong Python

Python cung cấp module `os` có các phương thức cho phép tạo, xóa, và thay đổi các thư mục.

Một số phương thức của module `os`:

`makedirs()`: Tạo thư mục mới. Phương thức này nhận một tham số là tên thư mục ta muốn tạo.

`chdir()`: Thay đổi thư mục hiện tại. Phương thức này nhận một tham số là tên thư mục mà ta muốn chuyển đến.

`getcwd()`: Hiển thị thư mục làm việc hiện tại.

`rmdir()`: Xóa thư mục.

Các bước để thực hiện đọc ghi file

Bước 1: Tạo ứng dụng

Mở một trình soạn thảo, ở đây ta sử dụng PyCharm. Sau đó ta tạo mới một Project Python, đặt tên và chọn đường dẫn chứa đồ án vừa tạo.

Bước 2: Viết mã để ghi file

Sau khi đã tạo project, hãy tạo file mã nguồn python, sau đó viết mã để thực hiện việc ghi dữ liệu lên file như hình sau:

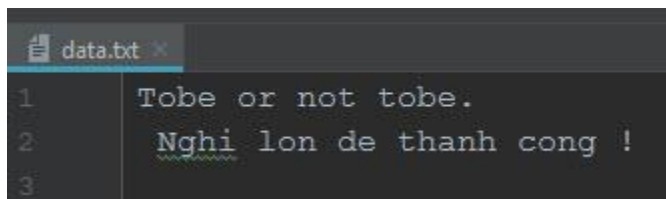
```
#mo file de ghi
fo = open("data.txt", "w")
#ghi du lieu len file
fo.write("Tobe or not tobe.\n Nghi lon de thanh cong !\n");

# Close opened file
fo.close()

print("Ghi file thanh cong !")
```

Trong ví dụ này, ta đã tạo và mở một file có tên là data.txt, sau đó ta ghi dữ liệu lên file, rồi ta đóng file vừa mở.

Sau khi đã thực thi file mã nguồn, file văn bản sẽ được tạo ra. Ta có thể mở file dữ liệu ra để xem dữ liệu đã được ghi.



```
data.txt x
1 Tobe or not tobe.
2 Nghi lon de thanh cong !
3
```

Bước 3: Viết mã để đọc file

Tiếp theo, ta tạo một file mã nguồn Python để viết mã đọc file dữ liệu vừa được tạo ra ở bước trên.

Trong bài này, ta đã mở file data.txt ở bước trên, rồi đọc một chuỗi bao gồm 20 ký tự, sau đó ta in ra chuỗi vừa đọc. Cuối cùng ta đóng file được mở.

```
DemoDocFile.py x
1  #mo file de doc du lieu
2  fo = open("data.txt", "r+")
3
4  #doc mot string trong file
5  str = fo.read(20);
6
7  #in ra chuoi duoc doc
8  print("Chuoi duoc doc la : ", str)
9
0  # Close opened file
1  fo.close()
2
```

Bước 4: Thực thi chương trình và xem kết quả

Sau khi đã viết mã xong, ta thực thi chương trình Python, ta thấy nội dung đã được hiển thị như hình bên dưới.

```
E:\DemoPythonBasics\venv\Scripts\python.exe E:/DemoPythonBasics/DemoDocFile.py
Chuoi duoc doc la :  Tobe or not tobe.
```