TOPSTechnologies

Opening and Closing Files

Presented for :

TOPs Technologies

Presented by :

Sumit B Yadav

Advance Python Programming

Que 1

In Python, when working with files, you can open them in different modes using the open() function.

1. Read Mode ('r')

- Opens the file for reading (default mode).
- If the file does not exist, it raises a FileNotFoundError.

example :

```
with open("example.txt", "r") as file:
content = file.read()
print(content)
```

2. Write Mode ('w')

- Opens the file for writing.
- If the file exists, it overwrites the content.
- If the file does not exist, it creates a new file.

example :

2. Write Mode ('w')

- Opens the file for writing.
- If the file exists, it overwrites the content.
- If the file does not exist, it creates a new file.

3. Append Mode ('a')

- Opens the file for appending data at the end.
- If the file does not exist, it creates a new file.
- Does not overwrite existing content.

example :

```
with open("example.txt", "a") as file:
file.write("\nAppending this line.")
```

4. Read and Write Mode ('r+')

- Opens the file for both reading and writing.
- The file must exist, otherwise, it raises a FileNotFoundError.

example :

```
with open("example.txt", "r+") as file:
content = file.read()
file.write("\nNew content added.")
```

5. Read and Write Mode ('w+')

- Opens the file for both reading and writing.
- If the file exists, it overwrites the content.
- If the file does not exist, it creates a new file.

example :

```
with open("example.txt", "w+") as file:
file.write("Overwriting content.")
file.seek(0)  # Move to the beginning of the file
print(file.read())  # Read the new content
```

Que. 2

The open() function in Python is used to create, open, and access files.

syntax :

```
file = open("filename", "mode")
```

- "filename": Name of the file (with the path if needed).
- "mode": Specifies how the file should be opened (read, write, append, etc.)

1. Creating a File

If the file does not exist, you can create it using 'w', 'a', or 'x' mode.

```
file = open("newfile.txt", "w")  # Creates a new file or overwrites if it exists
file.write("Hello, this is a new file.")
file.close()
```

2. Reading a File

To read an existing file, use 'r' mode.

ex :

```
with open("newfile.txt", "r") as file:
content = file.read()
print(content)
```

3. Writing to a File

Using 'w' mode will overwrite the file.

ex :

```
with open("newfile.txt", "w") as file:
file.write("This will replace previous content.")
```

4. Reading and Writing in the Same File (r+, w+)

Using 'r+' mode allows both reading and writing.

ex :

```
with open("newfile.txt", "r+") as file:
content = file.read()
print("Before Writing:", content)
file.write("\nAdding new content in read+write mode.")
```

5. Checking if a File Exists Before Opening

You can use os.path.exists() to avoid errors.

ex:

```
import os

if os.path.exists("newfile.txt"):
with open("newfile.txt", "r") as file:
print(file.read())
else:
print("File does not exist.")
```

Que. 3

In Python, after you've opened a file using the open() function, it is important to close the file using the close() method. Closing the file ensures that all changes made to the file are saved, and it frees up system resources.

- Memory Management: Closing a file releases the system resources associated with that file.
- Save Data: If you’re writing to a file, calling close() ensures that all data is properly written and saved.
- Avoid Errors: Leaving files open may lead to memory leaks or errors in your program

syntax :

```
# Open a file in write mode
file = open("example.txt", "w")

# Write data to the file
file.write("Hello, World!")

# Close the file
file.close()
```