File Handling Basics

Overview

File handling means interacting with files on your computer to **read data** from them or **write data** to them. It allows you to **store data permanently** on disk — unlike variables which are temporary (stored in RAM).

There are mainly **two types** of files:

Type	Description	Example
Text files	Store data in human-readable form (characters)	.txt, .csv, .json
Binary files	Store data in machine-readable form (0s and 1s)	.exe, .dat, .png, .mp3

Opening File

Opening File – open()

- Python provides built-in functions for file handling — mainly using the **open()** function

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

Opening File – open()

- Python provides built-in functions for file handling — mainly using the **open()** function

Mode	Meaning	Description
'r'	read (default)	Opens file for reading (default mode)
'w'	Write	Creates new file or overwrites existing
'a'	Append	Adds new data to end of file
'x'	Exclusive Create	Creates new file; error if file exists
'b'	Binary	Used with binary files (e.g., 'rb', 'wb')
't'	Text	Default mode (e.g., 'rt', 'wt')

Opening File – open()

- Python provides built-in functions for file handling — mainly using the **open()** function

Syntax
`file = open("filename", "mode")`
Examples
`file = open("file.txt", “r") # Opening a file in read mode`
`file = open("file.txt", “w") # Opening a file in write mode`
`file = open("file.txt", “a") # Opening a file in append mode`
`file = open("file.txt", “b") # Opening a file in binary mode`

Closing File

Closing File – close()

- Close a file in Python using the **close()** method.

Why close file?
- Frees up system resources
- Ensures data is properly written to disk.
- Prevents file corruption or data loss.

Syntax

```
file = open("filename", “mode")
File.close()
```

BEST PRACTICE (OPENING & CLOSING)

- Use **with** keyword.

Why?

- Automatically handles closing the file – even if an error occurs.

Syntax

```
with open("filename", “mode") as file:
    # content  here

# file automatically closed here
```

Reading File

Reading File

- Means opening the file in reading mode, and apply various methods to extract data.
- ‘r’ is default mode for open() function in read mode, but specifying it explicitly is a good practice.

Reading File – 3 methods

Python provide **three** ways:

Methods	Description
`f.read()`	Reads entire file
`f.readline()`	Reads one line at a time
`f.readlines()`	Returns a list of all lines

Reading File – read()

- It reads **entire file as a single string**
- Useful when you need to **process the whole file at once**

Python

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

# file automatically closed here
```

Reading File – read(x)

- You can specify **how many characters** you want to read.
- Below code **return first 5 characters** from the file

Python

```
with open(“data.txt", “r") as file:
    content = file.read(5)
    print(content)

# file automatically closed here
```

Reading File – readline()

- It read **one line at a time**.
- Useful for reading **line-by-line** (loops for multiple lines), especially for large files.

Python

```
with open(“data.txt", “r") as file:
    content = file.readline()
    print(content)

# file automatically closed here
```

Reading File – readlines()

- It read **all the lines** and returns them as **list of all lines**.
- Each string is a **single line** from the file, including new line character **(\n)**
- Useful for **processing** and **analyzing** all lines.

Python

```
with open(“data.txt", “r") as file:
    content = file.readlines()
    print(content)

# file automatically closed here
```

Writing to File

Writing to File – 2 methods

- Means opening the file in writing mode, and apply various methods to add content to the file.

Python provide **two** methods for writing into a file:

Methods	Description
`f.write()`	Writes a single string to the file
`f.writelines()`	Writes multiple lines from a list of strings

Writing to File – 5 Modes

Mode	Meaning	Description
`'w'`	Write Mode	Creates new file or overwrite existing
`'a'`	Append Mode	Adds data to end of file
`'x'`	Exclusive Creation Mode	Creates new file, error if exists
`'b'`	Binary Mode	Used with other modes for binary data
`'+'`	Update Mode	Opens file for reading and writing

Writing to File – write() | ‘w’

- Writes a single string to the file.
- Overwrites existing content

Python

```
with open(“data.txt", “w") as file:
    file.write(“Hey Everyone”)

# file automatically closed here
```

Writing to File – write() | ‘a’

- Appends data at the end of the file.
- Old content remains, new content added.
- New line character (\n) is not included automatically.

Python

```
with open(“data.txt", “a") as file:
    file.write(“Hey Everyone”)

# file automatically closed here
```

Writing to File – writelines() | ‘w’

- Writes multiple lines from a list.
- Overwrites existing content

Python

```
lines = [‘Line 1\n’, ‘Line 2\n’, ‘Line 3\n’]
with open(“data.txt", “w") as file:
    file.writelines(lines)

# file automatically closed here
```

Writing to File – writelines() | ‘a’

- Appends multiple lines from a list.
- Old content remains, new content added.
- New line character (\n) is not included automatically.

Python

```
lines = [‘Line 4\n’, ‘Line 5\n’, ‘Line 6\n’]
with open(“data.txt", “a") as file:
    file.writelines(lines)

# file automatically closed here
```

Writing to File at specific position

- The **seek()** method is used to **change the current position** of the file pointer (cursor) within an open file.

Syntax
`file_object.seek(offset, whence)`

Parameters:

- **offset** → The number of bytes (or characters) to move.
- **whence** → Optional parameter that defines the reference point for the movement.

Writing to File at specific position

Possible values of whence:

Value	Meaning	Description
0	Beginning of file (default)	Moves pointer to offset bytes from start
1	Current position	Moves pointer relative to its current place
2	End of file	Moves pointer relative to end (often negative offset)

Syntax
`file_object.seek(offset, whence)`

Writing to File at specific position

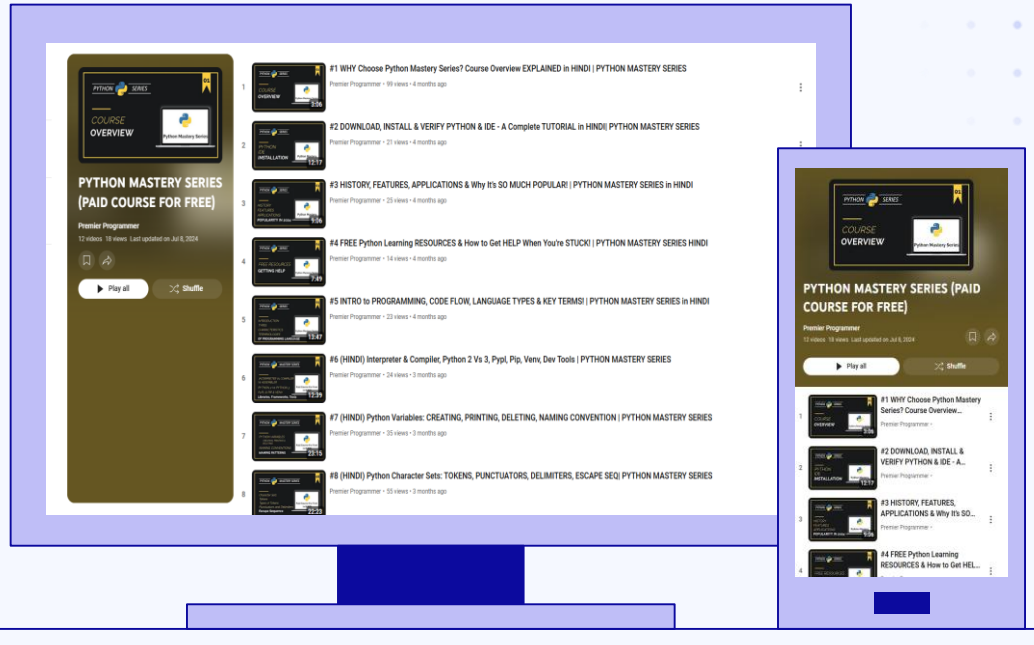
Example

Python

```python
with open("data.txt", "r") as f:
    f.seek(6)
    print(f.read())
```

WATCH

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



Next Video!

File Renaming
File Deleting