PYTHON TUTORIAL FOR BEGINNERS

Source: www.youtube.com/@RishabhMishraOfficial

Chapter - 22

File Handling in Python

- What is File Handling
- Open & Read a File
- Write & Append to a File
- Create a File
- Close a file
- Work on txt, csv, excel, pdf files

File handling in Python

File handling in Python allows you to read from and write to files. This is important when you want to store data permanently or work with large datasets.

Python provides built-in functions and methods to interact with files.

Steps for File Handling in Python:

- Opening a file
- Reading from a file
- Writing to a file
- Closing the file

Open a File

To perform any operation (read/write) on a file, you first need to open the file using Python's open() function.

Syntax: `file_object = open('filename', 'mode')`

- ***'filename'***: Name of the file (can be relative or absolute path).
- ***'mode'***: Mode in which the file is opened (read, write, append, etc.).

File Modes:

- 'r': Read (default mode). Opens the file for reading.
- 'w': Write. Opens the file for writing (if file doesn't exist, it creates one).
- 'a': Append. Opens the file for appending (if file doesn't exist, it creates one).
- 'rb'/'wb': Read/Write in binary mode.

Example: Opening a file for reading

```
file = open('example.txt', 'r')
```

Read from a File

Once a file is open, you can read from it using the following methods:

- read(): Reads the entire content of the file.
- readline(): Reads one line from the file at a time.
- readlines(): Reads all lines into a list.

```
# Example: Reading the entire file
file = open('example.txt', 'r')
content = file.read()
print(content)
file.close()
```

```
# Example: Reading one line at a time
file = open('example.txt', 'r')
line = file.readline()
print(line)
file.close()
```

Write to a File

To write to a file, you can use the write() or writelines() method:

- write(): Writes a string to the file.
- writelines(): Writes a list of strings.

```
# Example: Writing to a file (overwrites existing content)
file = open('example.txt', 'w')
file.write("Hello, world!")
file.close()
```

*# **Example**: Appending to a file (add line to the end)*

```
file = open('example.txt', 'a')
file.write("\nThis is an appended line.")
file.close()
```

*# **Close a file:***

```
file.close()
```

Close a File

Instead of manually opening and closing a file, you can use the **with** statement, which automatically handles closing the file when the block of code is done.

*# **Example**: Reading with `with` statement*

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

In this case, you don't need to call file.close(), because Python automatically closes the file when the block is finished.

*# **Example**: Using **exception handling** to close a file*

```
try:
    open('example.txt', 'r') as file:
    content = file.read()
    print(content)
finally:
    file.close()
```

Working with Difft Format Files

*# **csv** - Using csv module*

```
import csv
file = open('file.csv', mode='r')
reader = csv.reader(file)
```

 # csv - Using pandas library

```
import pandas as pd
df = pd.read_csv('file.csv')
```

 # excel - Using pandas library

```
import pandas as pd
df = pd.read_excel('file.xlsx')
```

 # PDF Using PyPDF2 library:

```
import PyPDF2
file = open('file.pdf', 'b')
pdf_reader = PyPDF2.PdfReader(file)
```

Python Tutorial Playlist: **Click Here**

https://www.youtube.com/playlist?list=PLdOKnrf8EcP384Ilxra4UlK9BDJGwawg9