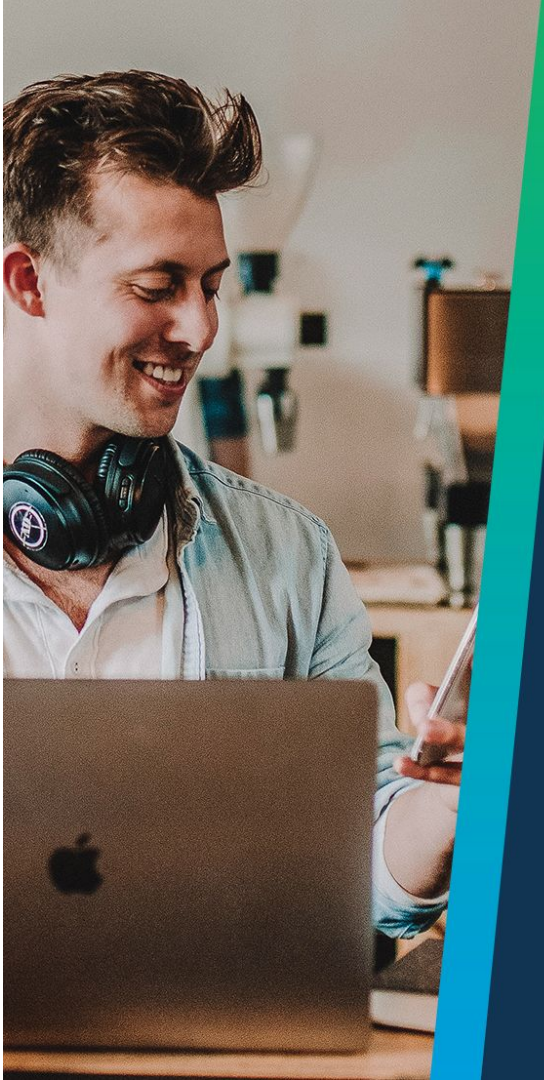


Hyperiondev

Working with External Data Sources - Input

Lecture – Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
- No question is daft or silly - **ask them!**
- There are Q/A sessions midway and at the end of the session, should you wish to ask any follow-up questions.
- You can also submit questions here: http://hyperiondev.com/sbc4-se-questions
- For all non-academic questions, please submit a query: www.hyperiondev.com/support
- Report a safeguarding incident: http://hyperiondev.com/safeguardreporting
- We would love your feedback on lectures: https://hyperionde.wufoo.com/forms/zsgv4m40ui4i0g/

Objectives

- Create smarter programs by learning how to read data from text files

File I/O

- ★ File I/O stands for file input/output
- ★ It is a process that reads data from an external file on the computer or outputs to another file.
- ★ Python has a built-in file type, which is the complex data type.
- ★ This means that Python can create variables of type “file”.

Opening a File

- ★ To read from a file, we must first open it.
- ★ To open a file, we use Python's built-in open() function, which creates what is known as a file object.

- ★ To utilize the file object's data, we store the file object in a variable.

- ★ Once we are done, we then close the file.

Opening Files

- To use a file in our program, we store the file object in a variable as such :
  - **file = open(file_name , access_mode)**

- **Access mode** : what the user can do when the file has been opened, such as reading (r), writing (w), or reading and writing (r+).

Opening Example

```
# To make opening the file easier,
#   best to keep the text file in the
#       same location as your Python file.

file_name = "input.txt"

file = open(file_name, "r")

# File is now being read by Python.
```

Reading Files

- ★ Files are opened in Python with the open() function. We know that open() will return a file object.
- ★ To then properly read the object, we will need to use the read method.
- ★ There are three methods : .read() , .readline(), .readlines()

Read Example

```
file_name = "input.txt"

file = open(file_name, "r")

# Read will simply read over all lines in
#   our text file. We can display them by printing.
lines = file.read()

print(lines)
```

Readline Example

```
file_name = "input.txt"

file = open(file_name, "r")

# Readline will simply read over the first line in
#   our text file. We can display them by printing.
line = file.readline()

print(line)
```

Readlines Example

```
file_name = "input.txt"

file = open(file_name, "r")

# Readlines will simply read over each line individually
#   within the text file. We can display them by printing.
line = file.readlines()

print(line)
# Keep in mind that the output is actually a list.
```

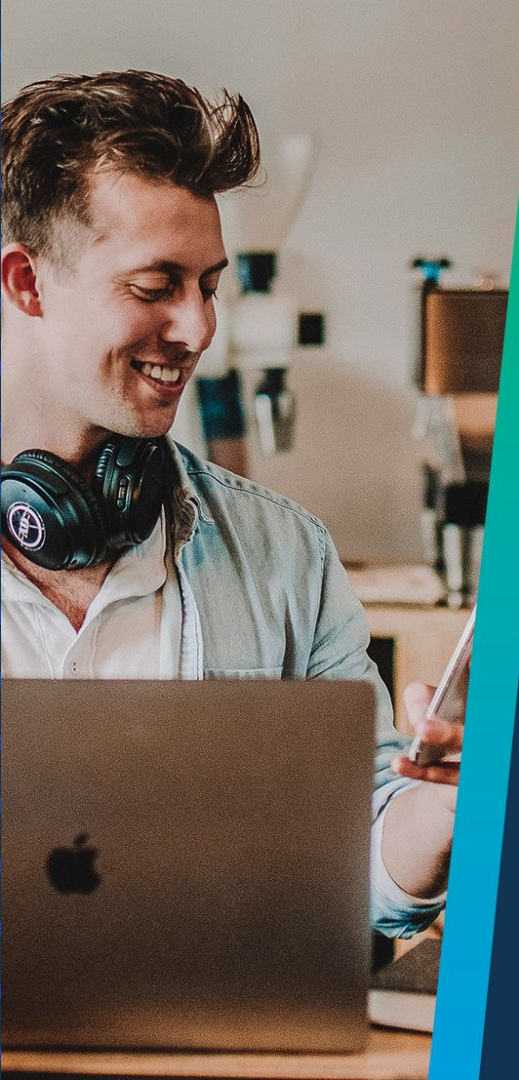
Closing a File

- ★ The close() method ensures system resources are not wasted in our programs.
- ★ It is always best practice to close files when you are finished working with them.
- ★ Remember that once a file is closed, it cannot be read again until is is re-opened.

Hyperiondev

Q & A Section

Please use this time to ask any questions relating to the topic explained, should you have any



Hyperiondev

Thank you for joining us