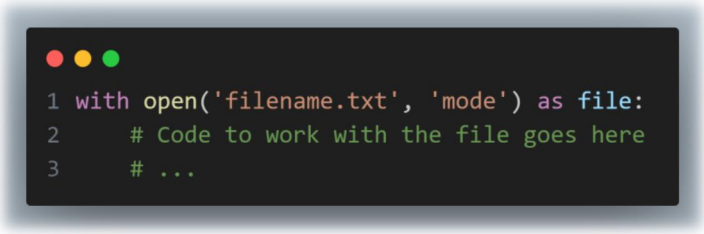
Python Files operations

With open:

❖ used to open a file in a way that ensures the file is properly closed after its suite finishes, even if an error occurs during the execution



```
with open('filename.txt', 'mode') as file:
    # Code to work with the file goes here
    # ...
```

File Modes for Text Files:

'r' (read):
- Opens an existing file for reading.
- Raises an error if the file doesn't exist.
- Default mode if none is specified.

'w' (write):
- Opens a file for writing.
- Creates a new file if it doesn't exist, or overwrites an existing file.

'a' (append):
- Opens a file for appending (adding content to the end).
- Creates a new file if it doesn't exist.

'x' (exclusive creation):
- Creates a new file for writing.
- Raises an error if the file already exists.

'r+' (read and write):
- Opens an existing file for both reading and writing.
- Raises an error if the file doesn't exist.

'w+' (write and read):
- Opens a file for both writing and reading.
- Creates a new file if it doesn't exist, or overwrites an existing file.

'a+' (append and read):
- Opens a file for both appending and reading.
- Creates a new file if it doesn't exist.

File Modes for Binary Files:

'rb' (read binary):
- Opens an existing binary file for reading.
- Raises an error if the file doesn't exist.

'wb' (write binary):
- Opens a binary file for writing.
- Creates a new file if it doesn't exist, or overwrites an existing file.

'ab' (append binary):
- Opens a binary file for appending.
- Creates a new file if it doesn't exist.

'rb+' (read and write binary):
- Opens an existing binary file for both reading and writing.
- Raises an error if the file doesn't exist.

'wb+' (write and read binary):
- Opens a binary file for both writing and reading.
- Creates a new file if it doesn't exist, or overwrites an existing file.

'ab+' (append and read binary):
- Opens a binary file for both appending and reading.
- Creates a new file if it doesn't exist.

readlines() method:

❖ This method reads all the lines from the file and returns them as a list of strings.

```python
with open('logfile.txt', 'r') as file:
    lines = file.readlines()
    for line in lines:
        print(line)
```

Use Cases:

Search for a word in lines then Replace this line in txt file

- We are using this to overwrite a line in the file
- But we can use it to edit the line too

```python
with open('logfile.txt', 'r') as file:
    lines = file.readlines()

for index, line in enumerate(lines):
    if "name" in line:
        print("Line index", index)
        print("Line Content=", line)
        lines[index] = "my name is Mohamed Hamed\n"  # Modify the line in the list

with open('logfile.txt', 'w') as file:
    file.writelines(lines)
```

Change line in binary file :

```python
with open('file.bin', 'rb+') as file:
    lines = file.readlines()

for index, line in enumerate(lines):
    if index == 2: # go to Specfic line
        line_inbytes = bytearray(line)  # Convert bytes to byteArray so we can modifiy it

        line_inbytes[1:10] = b'66'      # Change only the first 10 digits of the bytes
        line_inbytes = b'66'            # or we can overwrite the content of line to b'66'

        lines[index] = line_inbytes     # apply the changes to the specific line in all lines
        break


with open('file.bin', 'wb') as file:
    file.writelines(lines)              # write the lines back with the changed line
```

Remove Empty Lines:

```python
with open('logfile.txt', 'r') as file:       # Read from the File
    lines = file.readlines()

# Looping through the list `lines` in reverse order to efficiently remove empty lines
for index in range(len(lines) - 1, -1, -1):  # Loop in the list backward: range(Start=size-1, Stop=-1, Step=-1)
    if lines[index].strip() == "":           # Check if the line at index is empty after stripping whitespace
        del lines[index]                     # If empty, delete the line from the list

with open('logfile.txt', 'w') as file:       # Overwrite the file with the new changes
    file.writelines(lines)
```

Insert new Line Before a specific line:

We can insert the line after it by incrementing the index by 1 (index+1)

```python
with open('logfile.txt', 'r') as file:       # Read from the File
    lines = file.readlines()

# Iterate over a copy of the list `lines` using enumerate to access both index and line
line_to_insert = "insert This Line\n"
Required_word = "Mohamed"

for index, line in enumerate(lines[:]):
    if Required_word in line:
        print(f"found Required Line with Index = {index}")
        lines.insert(index, line_to_insert)

with open('logfile.txt', 'w') as file:       # Overwrite the file with the new changes
    file.writelines(lines)
```

We can use the same methods as above example to search for a line and delete it:

```python
del lines[Req_index]
```