Emmanuel Salako

About Follow

Python File Handling

 Emmanuel Salako Nov 5, 2020 · 2 min read

Definition and Usage of readline

The readline() method returns one line from the file. You can also specified how many bytes from the line to return, by using the size parameter.

```
f = open("data.txt", "r")
print(f.readline())
```

Definition and Usage of readlines

The `readlines()` method returns a list containing each line in the file as a list item. Use the hint parameter to limit the number of lines returned. If the total number of bytes returned exceeds the specified number, no more lines are returned.

```
f = open("data.txt", "r")
print(f.readline(33))
```

Definition and Usage of tell

The `tell()` method returns the current file position in a file stream.

```
f = open("demofile.txt", "r")
print(f.readline())
print(f.tell())
```

Definition and Usage of truncate

The `truncate()` method resizes the file to the given number of bytes. If the size is not specified, the current position will be used.

```
f.close()

#open and read the file after the truncate:
f = open("data.txt", "r")
print(f.read())
```

Definition and Usage of write

The `write()` method writes a specified text to the file. Where the specified text will be inserted depends on the file mode and stream position.

`"a"`: The text will be inserted at the current file stream position, default at the end of the file.

`"w"`: The file will be emptied before the text will be inserted at the current file stream position, default 0.

```
f = open("data.txt", "a")
f.write("See you soon!")
f.close()

#open and read the file after the appending:
f = open("data.txt", "r")
print(f.read())
```

```
f = open("data.txt", "a")
f.write("\nSee you soon!")
f.close()

#open and read the file after the appending:
f = open("data.txt", "r")
print(f.read())
```

Definition and Usage

The `writelines()` method writes the items of a list to the file. Where the texts will be inserted depends on the file mode and stream position.

`"a"`: The texts will be inserted at the current file stream position, default at the end of the file.

```
f = open("demofile3.txt", "a")
f.writelines(["See you soon!", "Over and out."])
f.close()

#open and read the file after the appending:
f = open("demofile3.txt", "r")
print(f.read())

f = open("demofile3.txt", "a")
f.writelines(["\nSee you soon!", "\nOver and out."])
f.close()

#open and read the file after the appending:
f = open("demofile3.txt", "r")
print(f.read())
```

Definition and Usage

The `flush()` method cleans out the internal buffer.

```
f = open("myfile.txt", "a")
f.write("Now the file has one more line!")
f.flush()
f.write("...and another one!")
```

Definition and Usage

The `fileno()` method returns the file descriptor of the stream, as a number. An error will occur if the operator system does not use a file descriptor.

```
f = open("demofile.txt", "r")
print(f.fileno())
```

Definition and Usage

The `isatty()` method returns `True` if the file stream is interactive, example: connected to a terminal device.

```
f = open("demofile.txt", "r")
print(f.isatty())
```

which means the whole file.

```
f = open("demofile.txt", "r")
print(f.read())
```

```
f = open("demofile.txt", "r")
print(f.read(33))
```