File and Directory Related Methods:

There are three important sources, which provides a wide range of utility methods to handle and manipulate files and directories on Windows and Unix operating systems. They are as follows.

File Object Methods: The file object provides functions to manipulate files.

OS objects Methods: This provides methods to process files as well as directories.

File Methods:

A file object is created using open function and here is a list of functions which can be called on this object .

file.close() : Close the file. A closed file cannot be read or written any more.

file.flush() : Flush the internal buffer, like stdio’s flush. This may be no-op on some file-like objects.

file.fileno(): Returns the integer file descriptor that is used by the underlying implementation to request I/O operations from the operating system.

The are some more method apart for this

File close() Method: The method close() closes the opened file. A closed file cannot be read or written any more. Any operation, which requires that the file be opened will raise a ValueError after the file has been closed. Calling close() more than once is allowed .

Python automatically closes a file when the reference object of a file is reassigned to another file. It is a good practice to use the close() method to close a file.

Syntax:

Following is the syntax for close() method

fileObject.close()

Parameters: NA

Return Value:

This method does not return value.

Example:

fo=open("foo.txt","wb")

print("Name of the file:",fo.name)

fo.close()

File flush() Method: The method Flush() the internal buffer, like stdio’s flush. This may be no-op on some file-like objects.

Python automatically flushes the when closing them. But you may want to flush the data before closing any file.

Syntax:

Following is the syntax for flush() method

fileObject.flush()

Parameters: NA

Return Value:

This method does not return value.

Example:

fo=open("foo.txt","wb")

print("Name of the file:",fo.name)

#Here it does nothing, but you can call it with read operation

fo.flush()

# Close opend file

fo.close()

File fileno() Method: The method fileno() returns the integer file descriptor that is used by the underlying implementation to request I/O operations from the operating system.

Syntax:

Following is the syntax for fileno() method

fileObject.fileno()

Parameters: NA

Return Value:

This method returns the integer file descriptor.

Example:

fo=open("foo.txt","wb")

print("Name of the file:",fo.name)

fo.close()

File isatty() Method: The method isatty() returns True if the file is connected (is associated with a terminal device) to a tty(-like) device, else False.

Syntax

fileObject.isatty()

Parameters: NA

Return Value: This method returns true if the file is connected (is associated with a terminal device) to a tty(-like) device, else false.

Example :

fo=open("abcd.txt","w")

print("Name of the file:",fo.name)

ret=fo.isatty()

print("Return value:",ret)

fo.close()

File next() Method: Python3 has a build in function next() which retrieves the next item from the iterator by calling its \_next\_() method. If default is given, it is returned if the iterator is exhausted , otherwise StopIteration is raised. This method can be used to read the next input line, from the file object.

Syntax:

next(iterator[,default])

Parameters:

Iterator: file object from which lines are to be read

Default: returned if iterator exhausted. If not given, StopIteration is raised

Return Value: This method returns the next input line.

Example:

fo=open("foo.txt", "r")

print("Name of the file:",fo.name)

for index in range(5):

line=next(fo)

print("Line No %d - %s" %(index,line))

fo.close()

File readline() Method: The method readline() reads one either line from the file. A trailing newline character is kept in the string. If the size argument is present and non-negative , it is a maximum byte count including the trailing newline and an incomplete line may be returned.

An empty string is returned only when EOF is encountered immediately.

Syntax: fileObject.readline(size);

Parameters:

Size : This is the number of bytes to be read from the file.

Return Value:

The method returns the line read from the file.

Example : fo=open("foo.txt","rt")

print("Name of the file:",fo.name)

line= fo.readline()

print("Read Line:%s" %(line))

line=fo.readline(5)

print("Read Line: %s" %(line))

fo.close()

File readlines() Method: The method readlines() reads until EOF using readline() and returns a list containing the lines. If the optional sizehint argument is present, instead of reading up to EOF, whole lines totalling approximately sizehint bytes (possibly after rounding up to an internal buffer size) are read.

An empty string is returned only when EOF is encountered immediately.

Syntax: fileObject.readlines( sizehint );

Parameters sizehint - This is the number of bytes to be read from the file.

Return Value This method returns a list containing the lines.