22.FileHandling

June 4, 2020

File Handling

[1]:
```
import time
```

[7]:
```
time.ctime()[11:16]
```

[7]:
```
'19:07'
```

[21]:
```
class Person:
    def __init__(self, name):
        self.name = name
    def __str__(self):
        return self.name
    def awake(self):
        if time.ctime()[11:16] != '19:11':
            return True
        print("Sleep Now")
        return False
    def code(self):
        print(f"!!mr. {self.name} !!! do code do code do code!!")
        time.sleep(2)
```

[22]:
```
me = Person('Sachin Yadav')
```

[23]:
```
print(me)
```

```
Sachin Yadav
```

[24]:
```
while me.awake():
    me.code()
```

```
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
```

```
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
!!mr. Sachin Yadav !!! do code do code do code!!
Sleep Now
```

[26]:

```python
print("Hello World!")
```

```
Hello World!
```

0.1 Os Module to use utility of your operating system

```
Path -> Absolute and relative, location of files
```

0.1.1 File Handling

Type of Files

```
Regular files, text files, human readble files
```

```
Byte files, Binary files, machine or program readable files
```

syntax

```
fp = open(path, mode, encoding)
```

```
fp.read()
```

```
fp.write(strings/bytes)
```

```
fp.readline()
```

```
fp.seek(12)
```

```
fp.tell()
```

```
fp.close()
```

path -> absolute path that start from root directory or / , relative path which starts from cu

Mode of Operation on a File

r - read only file, set by default

w - write only file, over-write existing file if exists

x - write only file, always creates a new file, through an error if file exists

a - append only mode, always add content at the end of file

t - regular file, text file, set by default

b - byte file

+ - give you permission to read and write into file at same time

if you do not pass any mode it bydefault is rt

rt+ - read and write a text file

ab+ - append and read a binary file

wt - write only text file, over-write content if exists

xb+ - new file with read write access

a   - append only text file

buffering ?

Path, mode, encoding

[27]:
```python
path = "C:\\users\\sachin\\Desktop\\hadoop.py"
```

[28]:
```python
import os
```

[29]:
```python
os.path.exists(path)
```

[29]:
```
True
```

[30]:
```python
os.path.isfile(path)
```

[30]:
```
True
```

[31]:
```python
os.access(path, os.R_OK)
```

```
[31]: True
```

```
[32]: fp = open('aljfsjdkfjdslfjas')
      # Exceptions or runtime errors or logical errors
```

```
              ␣
      ↪---------------------------------------------------------------------------

             FileNotFoundError                         Traceback (most recent call␣
      ↪last)

             <ipython-input-32-dc925fea5a1e> in <module>
         ----> 1 fp = open('aljfsjdkfjdslfjas')


             FileNotFoundError: [Errno 2] No such file or directory:␣
      ↪'aljfsjdkfjdslfjas'
```

```
[33]: path = "C:\\users\\sachin\\Desktop\\hadoop.py"
```

```
[34]: mode = 'rt'
```

```
[35]: encoding = 'utf-8'
```

```
[36]: fp = open(path, mode, -1, encoding)
```

```
[38]: print("mode: ", fp.mode)
```

```
      mode:  rt
```

```
[39]: print("name: ", fp.name)
```

```
      name:  C:\users\sachin\Desktop\hadoop.py
```

```
[40]: print("encoding: ", fp.encoding)
```

```
      encoding:  utf-8
```

```
[41]: print("closed?: ", fp.closed)
```

```
      closed?:  False
```

```
[42]: print("readable ?: ", fp.readable())
```

```
      readable ?:  True
```

[43]:

```
print("writeable ?: ", fp.writable())
```

```
writeable ?:  False
```

[44]:

```
print("seekable ?: ", fp.seekable())
```

```
seekable ?:  True
```

[45]:

```
print( *[ func for func in dir(fp) if func[0].islower() ], sep='\n' )
```

```
buffer
close
closed
detach
encoding
errors
fileno
flush
isatty
line_buffering
mode
name
newlines
read
readable
readline
readlines
reconfigure
seek
seekable
tell
truncate
writable
write
write_through
writelines
```

[46]:

```
print(type(fp))
```

```
<class '_io.TextIOWrapper'>
```

[47]:

```
fp.newlines
```

```
new line -> line break or new line character depends system specific

window -> \r\n

linux / unix -> \n
```

```
[48]: fp.tell() # at which pos we are at current time
```

```
[48]: 0
```

```
[49]: size = 10
      chars_10 = fp.read(size)
```

```
[50]: print(chars_10)
```

```
#!/usr/loc
```

```
[51]: fp.tell()
```

```
[51]: 10
```

```
[52]: next_10 = fp.read(size)
```

```
[53]: print(next_10)
```

```
al/anacond
```

```
[54]: next_10 = fp.read(size)
      print(next_10)
```

```
a3/bin/pyt
```

```
[55]: next_10 = fp.read(size)
      print(next_10)
```

```
hon

from
```

```
[56]: fp.tell()
```

```
[56]: 40
```

```
[57]: fp.seek(0)
```

```
[57]: 0
```

```
[58]: fp.tell()
```

```
[58]: 0
```

```
[59]: line = fp.read(35)
```

```
[60]: print(line)
```

```
#!/usr/local/anaconda3/bin/python
```

```
[61]: fp.read(5)
```

```
[61]: 'from '
```

```
[62]: fp.seek(0)
```

```
[62]: 0
```

```
[63]: line = fp.readline()
```

```
[64]: print(line)
```

```
#!/usr/local/anaconda3/bin/python
```

```
[65]: print(fp.readline())
```

```
[66]: print(fp.readline())
```

```
from flask import Flask, request
```

```
[67]: print(fp.readline())
```

```
import subprocess as sp
```

```
[68]: fp.seek(0)
```

```
[68]: 0
```

```
[69]: all_data = fp.read()
      print(all_data)
```

```
#!/usr/local/anaconda3/bin/python

from flask import Flask, request
import subprocess as sp

app = Flask(__name__)
```

```
@app.route('/')
def index():
    html=f"""
    <!Doctype html>
        <html>
            <body>
                <h1 style='color:red'>Welcome to Hadoop Commnad Service </h1>
                <form method='POST' action='/run'>
                <p><label>Command: </label><input type='text' name='cmd'></p>
                <p><input type='submit' value='Run Code'>
                </form>
            </body>
        </html>
    """
    return html

@app.route('/run', methods=['POST'])
def run():
    html = "<h1 style='color:#123456'>Here is Your Output <a
href='/'>Home</a></h1></br></br></br>"
    cmd = request.form.get('cmd')
    output = sp.getoutput('hdfs dfs -'+cmd)
    output = output.replace('\n', '</br>')
    output = output.replace('\t', '&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;')
    return html+output


if __name__ == '__main__':

    app.run('172.25.5.10', 80, debug=True)
```

[70]:
```
print(fp.read())
```

[71]:
```

```

[71]: 0

[78]:
```
fp.seek(0)
for line in fp:
    print(line.upper(), end='')
```

```
#!/USR/LOCAL/ANACONDA3/BIN/PYTHON
```

```
FROM FLASK IMPORT FLASK, REQUEST
IMPORT SUBPROCESS AS SP

APP = FLASK(__NAME__)

@APP.ROUTE('/')
DEF INDEX():
    HTML=F"""
    <!DOCTYPE HTML>
        <HTML>
            <BODY>
                <H1 STYLE='COLOR:RED'>WELCOME TO HADOOP COMMNAD SERVICE </H1>
                <FORM METHOD='POST' ACTION='/RUN'>
                <P><LABEL>COMMAND: </LABEL><INPUT TYPE='TEXT' NAME='CMD'></P>
                <P><INPUT TYPE='SUBMIT' VALUE='RUN CODE'>
                </FORM>
            </BODY>
        </HTML>
    """
    RETURN HTML

@APP.ROUTE('/RUN', METHODS=['POST'])
DEF RUN():
    HTML = "<H1 STYLE='COLOR:#123456'>HERE IS YOUR OUTPUT <A
HREF='/'>HOME</A></H1></BR></BR></BR>"
    CMD = REQUEST.FORM.GET('CMD')
    OUTPUT = SP.GETOUTPUT('HDFS DFS -'+CMD)
    OUTPUT = OUTPUT.REPLACE('\N', '</BR>')
    OUTPUT = OUTPUT.REPLACE('\T', '&NBSP;&NBSP;&NBSP;
&NBSP;&NBSP;&NBSP;&NBSP;&NBSP;')
    RETURN HTML+OUTPUT


IF __NAME__ == '__MAIN__':

    APP.RUN('172.25.5.10', 80, DEBUG=TRUE)
```

[79]:

```
fp.close() # to free buffer space and resource lock
```

Context Managers

they resource automatically such that if you forgot to close file after processing they will cl

[80]:

```
path = 'C:\\users\\sachin\\Desktop\\hadoop.py'
mode = 'rt'
encoding = 'utf-8'
```

[81]:

```
fp = open(path, mode, -1, encoding)
data = fp.read()
print(data)
```

```
#!/usr/local/anaconda3/bin/python

from flask import Flask, request
import subprocess as sp

app = Flask(__name__)

@app.route('/')
def index():
    html=f"""
    <!Doctype html>
        <html>
            <body>
                <h1 style='color:red'>Welcome to Hadoop Commnad Service </h1>
                <form method='POST' action='/run'>
                <p><label>Command: </label><input type='text' name='cmd'></p>
                <p><input type='submit' value='Run Code'>
                </form>
            </body>
        </html>
    """
    return html

@app.route('/run', methods=['POST'])
def run():
    html = "<h1 style='color:#123456'>Here is Your Output <a
href='/'>Home</a></h1></br></br></br>"
    cmd = request.form.get('cmd')
    output = sp.getoutput('hdfs dfs -'+cmd)
    output = output.replace('\n', '</br>')
    output = output.replace('\t', '&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;')
    return html+output


if __name__ == '__main__':

    app.run('172.25.5.10', 80, debug=True)
```

[83]:

```
fp.closed
```

[83]:
```
False
```

[84]:
```
fp.close() # we can forget it so what to do to prevent it
```

use context managers

[85]:
```
path = 'C:\\users\\sachin\\Desktop\\hadoop.py'
mode = 'rt'
encoding = 'utf-8'
# fp = open(...)
with open(path, mode, encoding=encoding) as fp:
    data = fp.read()
    print(data)
```

```
#!/usr/local/anaconda3/bin/python

from flask import Flask, request
import subprocess as sp

app = Flask(__name__)

@app.route('/')
def index():
    html=f"""
    <!Doctype html>
        <html>
            <body>
                <h1 style='color:red'>Welcome to Hadoop Commnad Service </h1>
                <form method='POST' action='/run'>
                <p><label>Command: </label><input type='text' name='cmd'></p>
                <p><input type='submit' value='Run Code'>
                </form>
            </body>
        </html>
    """
    return html

@app.route('/run', methods=['POST'])
def run():
    html = "<h1 style='color:#123456'>Here is Your Output <a
href='/'>Home</a></h1></br></br></br>"
    cmd = request.form.get('cmd')
    output = sp.getoutput('hdfs dfs -'+cmd)
    output = output.replace('\n', '</br>')
    output = output.replace('\t', '&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;')
    return html+output
```

```
if __name__ == '__main__':

    app.run('172.25.5.10', 80, debug=True)
```

[86]:

```
fp.closed
```

[86]:

```
True
```

[87]:

```
def read_file(path, mode='rt', encoding='utf-8'):
    if os.path.exists(path):
        if os.path.isfile(path):
            if os.access(path, os.R_OK):
                with open(path, mode, encoding=encoding) as file:
                    data = file.read()
                    fp.close()
                return data
            else:
                print("!!!Hey!!! you don't have permission to open this file...
 ↪don't try something nesty")
        else:
            print("!!Error!! you fool, idiot give a file path not directory␣
 ↪path")
    else:
        print("!!Error!! Given Path Does not Exists please Double check it and␣
 ↪retry")
```

[89]:

```
data = read_file('aldksjfds')
```

```
!!Error!! Given Path Does not Exists please Double check it and retry
```

[91]:

```
print(data)
```

```
None
```

[92]:

```
path = "C:\\Windows\\System32\\config\\SAM"
```

[93]:

```
data = read_file(path)
```

```
        ␣
 ↪---------------------------------------------------------------------------

        PermissionError                           Traceback (most recent call␣
 ↪last)
```

```
        <ipython-input-93-396c6cc84e9f> in <module>
    ----> 1 data = read_file(path)


        <ipython-input-87-dfd735d7455d> in read_file(path, mode, encoding)
          3         if os.path.isfile(path):
          4             if os.access(path, os.R_OK):
    ----> 5                 with open(path, mode, encoding=encoding) as file:
          6                     data = file.read()
          7                     fp.close()


        PermissionError: [Errno 13] Permission denied: 'C:
   ↪\\Windows\\System32\\config\\SAM'
```

[94]:
```
data = read_file('C:\\users\\sachin\\desktop\\hadoop.py')
```

[95]:
```
print(data)
```

```
#!/usr/local/anaconda3/bin/python

from flask import Flask, request
import subprocess as sp

app = Flask(__name__)

@app.route('/')
def index():
    html=f"""
    <!Doctype html>
        <html>
            <body>
                <h1 style='color:red'>Welcome to Hadoop Commnad Service </h1>
                <form method='POST' action='/run'>
                <p><label>Command: </label><input type='text' name='cmd'></p>
                <p><input type='submit' value='Run Code'>
                </form>
            </body>
        </html>
    """
    return html

@app.route('/run', methods=['POST'])
def run():
    html = "<h1 style='color:#123456'>Here is Your Output <a
href='/'>Home</a></h1></br></br></br>"
```

```
    cmd = request.form.get('cmd')
    output = sp.getoutput('hdfs dfs -'+cmd)
    output = output.replace('\n', '</br>')
    output = output.replace('\t', '&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;')
    return html+output

if __name__ == '__main__':

    app.run('172.25.5.10', 80, debug=True)
```

[104]:

```
# student.txt -> relative path -> current dir -> os.getcwd()
print(os.getcwd())
fp = open('student.txt', 'w')
```

```
C:\Batches\Batch_7pm_online
```

[105]:

```
# write stirngs because it's regular file
fp.write("Name,Math,Phy,Chem\n")
```

[105]:

```
19
```

pwd -> works in jupyter not in general

[106]:

```
fp.write("Sachin,100,95,85\n")
```

[106]:

```
17
```

[107]:

```
fp.write("Rajat,90,78,95\n")
```

[107]:

```
15
```

[108]:

```
fp.write('Nidhi,90,90,90\n')
# no of bytes written to file
```

[108]:

```
15
```

[109]:

```
fp.write('Simran,90,95,80\n')
```

[109]:

```
16
```

[110]:

```
fp.write("Vansu,67,76,89\n")
```

[110]:

```
15
```

[111]:

```
fp.close()
```

[113]:
```
fp = open('student.txt', 'rt')
```

[114]:
```
data = fp.read()
```

[115]:
```
print(data)
```

```
Name,Math,Phy,Chem
Sachin,100,95,85
Rajat,90,78,95
Nidhi,90,90,90
Simran,90,95,80
Vansu,67,76,89
```

[116]:
```
fp.close()
```

Assignment -> open above student.txt file and do following

```
write a function student_result(path)

find percentage marks gain by each student and create a file student_result.txt and write resul

Sachin,95.00%
Rajat,89.00%
Nidhi,99.00%
...

write a function subject_result(path)

find overall result subject wise and write into subject_result.txt file as below format

Math,89.00%
Phy,76.00%
...
```

[117]:
```
fp = open("student.txt", 'a')
```

[118]:
```
fp.write("Neha,98,87,79\n")
```

[118]:
```
14
```

[119]:
```
fp.close()
```

[120]:
```
fp = open('student.txt')
```

[121]:
```
data = fp.read()
```

[122]:
```
print(data)
```

```
Name,Math,Phy,Chem
Sachin,100,95,85
Rajat,90,78,95
Nidhi,90,90,90
Simran,90,95,80
Vansu,67,76,89
Neha,98,87,79
```

[123]:
```
fp.close()
```

JSON and PICKLE

`Serialization & Deserialization`

OOPS

[ ]:
```
```