IT Scripting and Automation

Read / Write Files in Python

Lecturer: Art Ó Coileáin

File Types

- There are two types of files:
  - Text files.
  - Binary files.
- A text file is simply a file which stores sequences of characters, whereas in the case of binary file data is stored in the same format as in Computer memory.

Examples:

- Text files: Python source code, HTML file, text file, markdown file etc.
- Binary files: executable files, images, audio etc.

NB:

- It is important to note that inside the disk both types of files are stored as a sequence of 1s and 0s. The only difference is that when a text file is opened the data is decoded back using the same encoding scheme they were encoded in. However, in the case of binary files no such thing happens.

Python File Objects

- Python has in built functions to create and manipulate files.
- The **io** module is the default module for accessing files and you don't need to import it.
- The module consists of **open(filename, access_mode)** that returns a file object, which is called "handle".
- You can use this handle to read from or write to a file.
- Python treats the file as an object, which has its own attributes and methods
- As you might have expected from reading the previous section, text files have an **End Of Line** (EOL) character to indicate each line's termination.
- In Python, the **new line character (\n)** is **default EOL** terminator.

OPEN() Function

- The built in Python function **open()** has the following arguments:

open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd =True, opener=None)

- The above **open()** tells also the default values for each argument
- ***file*** is an argument that you have to provide to the open function. This argument is basically the path where your file resides
  - All other arguments are optional and have default values.

Example:

```
f_handle = open ("mytext.txt")
```

OR

```
f_handle = open ("\\directory\\file.txt")
```

Catch the exception

- Make sure file name and path given is correct, otherwise you'll get a **FileNotFoundError**

Example:

```
try:
        f_handle =open("myfile.txt", “r”)
except IOError
        print("File not found or path is incorrect")
else:
        print(“Correct file opening”)
        #Actions after opening a file
```

Access Modes

- Access modes define in which way you want to open a file:
  - You may want to open a file for read only, write only or for both.
  - It specifies from where you want to start reading or writing in the file.
- You specify the access mode of a file through the mode argument.
  - You use 'r', the default mode, to read the file.
  - In other cases where you want to write or append, you use ‘w’ or 'a', respectively.

Character	Function
r	Open file for reading only. Starts reading from beginning of file. This default mode.
rb	Open a file for reading only in binary format. Starts reading from beginning of file.
r+	Open file for reading and writing. File pointer placed at beginning of the file.
w	Open file for writing only. File pointer placed at beginning of the file. Overwrites existing file and creates a new one if it does not exists.
wb	Same as **w** but opens in binary mode.
w+	Same as **w** but also alows to read from file.
wb+	Same as **wb** but also alows to read from file.
a	Open a file for appending. Starts writing at the end of file. Creates a new file if file does not exist.
ab	Same as **a** but in binary format. Creates a new file if file does not exist.
a+	Same a **a** but also open for reading.
ab+	Same a **ab** but also open for reading.

How to read a file in Python?

Example:

```
f_handle=open("myfile.txt", "r") # open the file in read mode

f_handle.read()          # read the entire file
f_handle.read(2)         # read the fist two characters of the file
f_handle.readline()      # read a full line
f_handle.readline(3)     # read the first three bytes of a line. It does not read
more than a line

f_handle.close()         #close the file (clear buffer)

print(f_handle.readline()) #print the read line to the standard output (screen)
```

How to read a file in Python?

```
#!/usr/bin/env python3

import sys
f=sys.argv[1]
try:
    my_file_handle=open(f,'r')
except IOError:
    print("File not found or path incorrect")
else:
    print("Reading file " + sys.argv[1] )
    c=1
    for line in my_file_handle:
      print("Reading line %d: "%(c) , line,end='')
      c+=1
    my_file_handle.close()
```

How to write files in Python?

Example:

new_file = open(“newfile.txt”,mode=“w”,encoding ="utf-8")

new_file.write("Writing to a new file\n")

new_file.close()

```
#!/usr/bin/env python3

import sys

f=sys.argv[1]
print("Writing to file:"+f)
try:
    my_file=open(f,'w',encoding="utf-8")
except IOError:
    print("File not found or path incorrect")
else:
    print("Writing started")
    for c in range(5):
      my_file.write("Writing line %d to a file \n" %(c))
    print("Writing to file has finished")
    my_file.close()
~
```

Exercise

- Improve the Scripts in the previous slides:
  - Validate the number of parameters - Only accept one parameter.