File IO

Python Open Lab

Why we need to do File IO?

- We have learned how to use user input.
  - myInput = input(“please input a number:”)
  - myNumber = str(myinput)
- Sometimes it is time-consuming to input data one by one. Using file is efficient to read data.
- We can also store the result to files.

Open Files

- Using python built-in “open” function, return a file object
  - afile = open(*filename*, *mode*)
  - afile.*method*()
- filename: absolute path, relative path
- mode: ‘r’, ‘w’, ‘a’
- afile is the object which represents a file
  - Just like list, dictionary… We can treat them in a similar way

Filename

- Filename includes the name of the file and its directory path
- Name is easy to get, the path is more difficult
- Absolute path and relative path
- For example,
  - /home/myname/a.txt
  - ./a.txt

File system

- /home
  - /Mike
    - /OpenLab
      - a.txt
      - d.txt
    - e.txt
  - /Jason
    - /OpenLab
      - b.txt
  - /James
    - /OpenLab
      - c.txt

- Present directory path:
  - MacOS: pwd
  - Windows: cd
- Use your terminal to input the command

Absolute path and relative path

- Absolute path
  - See files from the global view of the file system
  - Need to write the full path of the file
  - MacOs: /home/myname/a.txt
  - Windows: C:/Users/myname/a.txt

- Relative path
  - See files from your present directory (the position of your python script)
  - Do not need to write the full path of the file
  - MacOS/Windows ： ./d.txt , ../e.txt

Something to know about relative path

- Present directory and parent directory
  - Result of pwd: /home/Michael/lab
  - ‘.’ represents the present directory, which means directory ‘lab’
  - ‘..’ represents the parent directory, which means directory ‘Michael’
- For example, if I have a script ‘file.py’ in this directory
  - My present position: /home/Michael/lab/file.py
  - /home/Michael/lab/data.txt
  - /home/Michael/data2.txt

  - To read data.txt, I need to use “./data.txt” as the relative path in script.py
  - To read data2.txt, I need to use “../data2.txt” as the relative path in script.py

Mode

- afile = open(*filename*, ***mode***)
- ‘r’
  - Read only
- ‘w’
  - Write only
- ‘a’
  - Append text to the end
- ‘r+’
  - Both read and write

Before we try examples

- Create a file named ‘a.txt’ at the same directory of your python script
- Add this content to your file:
  - Line1: Hello world
  - Line2: I love NY
  - Line3: I like Friday!
- Close the file
- We are all set!

Read file

```
afile = open('a.txt', 'r')
content = afile.read()
print(content)
afile.close()
```

What we get?

Guess what does the function read do?

Read file

- Want to read line by line?

```
afile = open('a.txt', 'r')
content = afile.readline()
print(content)
content = afile.readline()
print(content)
afile.close()
```

Read file

- Use a for loop to read files line by line

```
afile = open('a.txt')
for line in afile:
        print(line)
afile.close()
```

Read file

- Read all lines at one time
- file.readlines(), return a list

```
afile = open('a.txt', 'r')
lines= afile.readlines()
for line in lines:
    print(line)
afile.close()
```

Read file

- Let the program close the file automatically

```
with open('a.txt') as fp:
        line = fp.readline()
        while line:
                print(line)
```

Write file

- output.write(aString)
- Write a string of characters (or bytes) into file

```
afile = open('a.txt', 'r+')
afile.write("Friday is great!")
print(afile.readlines())
afile.close()
```

Write file

- Append to a file to avoid overwrite
- Change mode to ‘a’

```
afile = open(‘a.txt’, ‘a’)
afile.write(“Friday is great!”)
afile.close()
afile = open(‘a.txt’, ‘r’)
print(afile.readlines())
afile.close()
```

Write file

- We can to change line
- Use ‘\n’

```
#do some clean up, remove everything in the a.txt
afile = open(‘a.txt’, ‘r+’)
afile.write(“”)
afile.close()
```

Write file

```
List = [‘I love NY’, ‘Python lab’, ‘Today is Friday’]
afile = open(‘a.txt’, ‘w’)
for sentence in list:
        afile.write(sentence+ “\n”)
afile.close()
print(open(‘a.txt’, ‘r’).read())
```

File existence

- afile = open(‘a.txt’, ‘w’)
- What if a.txt does not exist?
- Mode ‘w’ will create an empty file named ‘a.txt’
- Only the mode ‘w’ will do the creation, other modes will show an error

File existence

```
afile = open('g.txt', 'w')
afile.write("Friday is great!")
afile.close()
```

The code will look at the directory, if there is no 'g.txt', it creates one and write content to it.

Exercise

- Copy file Copy the content of a.txt to b.txt
- Modify file Output the content of a.txt to b.txt, all in uppercase
- Keyword spot We have a file ‘alice.txt’, output all lines that contains ‘alice’ to ‘result.txt’

Reference

- Learning python(5th edition),
  - Chapter 9, pp.282-294