Реферат

Тема: Робота з файлами

Файл (англ. file — шухляда, тека, папка) — інформаційний об'єкт, що містить дані або програми і розміщується на поіменованій ділянці носія даних, сутність, елемент (одиниця носія інформації; англ. media unit), що дозволяє отримати доступ до певного ресурсу обчислювальної системи і має такі ознаки:

* фіксована назва (назва файлу — послідовність символів (англ. string), число чи щось інше, що однозначно характеризує файл);
* певну логічну будову (структуру) і відповідні йому операції читання/запису.

На практиці це іменований блок інформації, який зберігається на носії інформації.

Файл обов'язково має назву і може мати будь-який розмір інформації (максимальна довжина назви та розміру файлу обмежується властивостями конкретної файлової системи). Файл може мати набір атрибутів. Треба зазначити, що файл може мати більш ніж одну назву (або синоніми назви, наприклад в Unix можна створювати жорсткі посилання на файл, які стають повним аналогом початкової назви файлу).

В інформатиці використовується наступне визначення: файл — це впорядкована сукупність даних, що зберігається на диску і займає іменовану область зовнішньої пам'яті. Згідно з термінологією, прийнятою в програмуванні, файл — це довільний блок інформації, або пристрій вводу-виводу, асоційований із ним (як середовище, засіб передачі "довільних блоків інформації"). Кожна комп'ютерна програма відкриває принаймні три файли стандартних потоків: вхідний файл (stdin), вихідний файл (stdout), файл виводу повідомлень про помилки (stderr). Вхідний файл зазвичай асоціюється із клавіатурою, вихідний файл та файл виводу помилок — із дисплеєм терміналу, проте кожен із цих файлів може бути перенаправлений на інший носій інформації, наприклад, на дисковий файл, про який мовилося вище, на мережу, яка також є файлом, на контролер будь-якого периферійного пристрою комп'ютера, зчитувача інформації тощо. Файлом може бути також ділянка оперативної пам'яті програми.

Python підтримує безліч різних типів файлів, але умовно їх можна розділити на два види: текстові та бінарні. Текстові файли - це наприклад файли з розширенням cvs, txt, html, загалом будь-які файли, які зберігають інформацію у текстовому вигляді. Бінарні файли - це зображення, аудіо та відеофайли і т.д. Залежно від типу файлу, робота з ним може трохи відрізнятися.

При роботі з файлами необхідно дотримуватися певної послідовності операцій:

* Відкриття файлу за допомогою методу open()
* Читання файлу за допомогою методу read() або запис файл за допомогою методу write()
* Закриття файлу методом close()

Щоб розпочати роботу з файлом, його треба відкрити за допомогою функції open(), яка має таке формальне визначення:

open(file, mode)

Перший параметр функції - шлях до файлу. Шлях файлу може бути абсолютним, тобто починатися з літери диска, наприклад C://somedir/somefile.txt. Або можна бути відносним, наприклад, somedir/somefile.txt - у цьому випадку пошук файлу йтиме щодо розташування запущеного скрипта Python.

Другий аргумент - mode встановлює режим відкриття файлу в залежності від того, що ми збираємося з ним робити. Існує 4 загальні режими:

* r (Read). Файл відкривається для читання. Якщо файл не знайдено, то генерується виняток FileNotFoundError
* w (Write). Файл відкривається для запису. Якщо файл відсутній, він створюється. Якщо подібний файл вже є, він створюється заново, і відповідно старі дані у ньому стираються.
* a (Append). Файл відкривається для запису. Якщо файл відсутній, він створюється. Якщо подібний файл вже є, дані записуються в його кінець.
* b (Binary). Використовується для роботи з бінарними файлами. Застосовується разом з іншими режимами – w або r.

try:

somefile = open("hello.txt", "w")

try:

somefile.write("hello world")

except Exception as e:

print(e)

finally:

somefile.close()

except Exception as ex:

print(ex)

Однак є і зручніша конструкція - конструкція with:

with open(file, mode) as file\_obj:

Запис до текстового файлу

Щоб відкрити текстовий файл на запис, необхідно застосувати режим w (перезапис) або a (дозапис). Потім для запису застосовується метод write(str), в який передається рядок, що записується. Варто зазначити, що записується саме рядок, тому, якщо потрібно записати числа, дані інших типів, їх попередньо потрібно конвертувати в рядок.

with open("hello.txt", "w") as file:

file.write("hello world")

Запишемо деяку інформацію у файл "hello.txt":

Якщо ми відкриємо папку, де знаходиться поточний скрипт Python, то побачимо там файл hello.txt. Цей файл можна відкрити в будь-якому текстовому редакторі та за бажанням змінити.

Тепер дозапишемо в цей файл ще один рядок:

with open("hello.txt", "a") as file:

    file.write("\ngood bye, world")

Ще один спосіб запису в файл є стандартним методом print(), який застосовується для виведення даних на консоль:

with open("hello.txt", "a") as hello\_file:

print("Hello, world", file=hello\_file)

Читання файлу

Для читання файлу він відкривається з режимом r (Read), а потім ми можемо вважати його вміст різними методами:

* readline(): зчитує один рядок із файлу
* read(): зчитує весь вміст файлу в один рядок
* readlines(): зчитує всі рядки файлу до списку

Наприклад, вважаємо вище записаний файл рядково:

with open("hello.txt", "r") as file:

    for line in file:

        print(line, end="")

Незважаючи на те, що ми явно не застосовуємо метод readline() для читання кожного рядка, але при переборі файлу цей метод автоматично викликається для отримання кожного нового рядка. Тому в циклі вручну немає сенсу викликати метод readline. І оскільки рядки поділяються символом перекладу рядка \n, то щоб виключити зайвого перенесення на інший рядок у функцію print передається значення end="".

Тепер явно викличемо метод readline() для читання окремих рядків:

with open("hello.txt", "r") as file:

    str1 = file.readline()

    print(str1, end="")

    str2 = file.readline()

    print(str2)

Питання:

1. Що таке файл?
2. Як файл перекладається з англійської?
3. Які ознаки має файл?
4. Що з себе файл представляє на практиці?
5. Чи обов’язково файл повинен мати назву?
6. Як відкрити файл на Пайтоні?
7. Які найвідоміші формати файлів?
8. Які у Пайтоні є модифікатори для відкриття файлу?
9. Як відкрити файл для читання?
10. Як відкрити файл для запису?
11. Як відкрити файл для дозапису?
12. Як відкрити файл бінарний?
13. З якими модифікаторами комбінується модифікатор b?
14. Чи можна поєднати модифікатор w з модифікатором b?
15. Чи можна поєднати модифікатор r з модифікатором b?
16. Чи можна поєднувати модифікатори?
17. Чи може викинути Екзепшен функція роботи з файлами?
18. Чи можна отримати виключення при роботі з файлами?
19. Як обробити виключення при роботі з файлами?
20. Скільки існує найпростіших методів обробки виключень?
21. Що краще, використати try catch чи with?
22. Яка форма запису with?
23. Яка форма запису try catch?
24. Що таке повний шлях?
25. Що таке відносний шлях?
26. Який метод для закриття файлу?
27. Який метод для відкриття файлу?
28. Який метод для читання з файлу?
29. Який метод для запису у файл?
30. Чи потрібно закривати файли?
31. Чи закривається файл автоматично при роботі with?
32. Чи може with перехоплювати виключення?
33. Чи можна використовувати методи роботи з файлами у try catch?
34. Чи можна відкрити бінарний файл?
35. У якому методі задається ім’я файлу, що відкривають?
36. Що позначає модифікатор «a»?
37. Як розшифрувати модифікатор «a»?
38. Що позначає модифікатор «b»?
39. Як розшифрувати модифікатор «b»?
40. Що позначає модифікатор «r»?
41. Як розшифрувати модифікатор «r»?
42. Що позначає модифікатор «w»?
43. Як розшифрувати модифікатор «w»?
44. Чи можна використати метод для читання файлу?
45. Чи можна використати метод readline() для читання файлу?
46. Чи можна використати метод read() для читання файлу?
47. Чи можна використати метод readlines() для читання файлу?
48. Метод read() зчитує весь файл?
49. Чи зчитає метод readlines() усі рядки файлу?
50. Чи можна перебрати результат readlines() у for?
51. Чи можна перебрати результат readlines() у while?
52. Чи варто використовувати with при роботі з файлами?

Джерела:

https://metanit.com/python/tutorial/4.1.php

https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB