Робота з файлами

№ уроку: 8 **Курс:** Python Essential

Засоби навчання: PyCharm

Огляд, ціль та призначення уроку

Після завершення уроку учні матимуть уявлення про файли та потоки, зможуть записувати та зчитувати дані з файлів, мати уявлення про роботу менеджерів контексту.

Вивчивши матеріал даного заняття, учень зможе:

- Мати уявлення про роботу з файлами
- Записувати та зчитувати дані у текстовому та бінарному форматах
- Мати уявлення про оператор with та менеджерів контексту

Зміст уроку

1. Навіщо працювати з файлами?
2. Файли та файлова система
3. Відкриття файлів
4. Закриття файлу
5. Запис до файлу
6. Читання з файлу
7. Методи об'єкта файлу

Резюме

Файли використовуються програмами для **довготривалого** зберігання інформації, як необхідної для власної роботи (наприклад, налаштування), так і отриманої під час виконання (результати обчислень тощо). Більшість програм сьогодні у тому чи іншому вигляді використовують файли, зберігаючи результати роботи між сеансами запуску.

Файли та файлова система
Файл - іменована область даних на носії інформації.
Оскільки оперативна пам'ять (ОЗП) є енергозалежною (яка втрачає свої дані при вимкненні комп'ютера), ми використовуємо файли для майбутнього використання даних, постійно зберігаючи їх.
Коли ми хочемо читати або записувати до файлу, нам потрібно спочатку відкрити його. Коли ми закінчимо роботу з файлом, його потрібно закрити, щоб звільнити ресурси, пов'язані з файлом.

Отже у Python файлова операція виконується в наступному порядку:
1. Відкрити файл.
2. Читання чи запис (виконання операції).
3. Закрити файл.

Відкриття файлів
Python має вбудовану функцію open() для відкриття файлу. Ця функція повертає файловий об'єкт, також званий **дескриптором**, оскільки він використовується для читання або зміни файлу відповідно.

```
>>> f = open("test.txt")    # open file in current directory
>>> f = open("C:/Python38/README.txt")  # specifying full path
```

Ми можемо вказати режим під час відкриття файлу. У режимі ми вказуємо, чи хочемо прочитати r, записати w або додати до файлу. Ми також можемо вказати, чи хочемо відкрити файл у текстовому або двійковому режимі.

CyberBionic systematics
CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

За замовчуванням читання у текстовому режимі. У цьому режимі ми отримуємо рядки під час читання з файлу. З іншого боку, двійковий режим повертає байти, і це режим, який слід використовувати при роботі з не текстовими файлами, такими як зображення або виконувані файли.

Режим роботи з файлами

Режим	Опис
r	Відкриває файл для читання. (за замовчуванням)
w	"Відкриває файл для запису. Створює новий файл, якщо він не існує, або обрізає файл, якщо він існує."
x	"Відкриває файл для ексклюзивного створення. Якщо файл вже існує, операція не виконується."
a	" Відкриває файл для додавання в кінець файлу без його усічення. Створює новий файл, якщо він не існує."
t	Відкривається у текстовому режимі. (за замовчуванням)
b	Відкривається у бінарному режимі.
+	Відкриває файл для оновлення (читання та запису)

```
f = open("test.txt")      # еквівалентно 'r' або 'rt'
f = open("test.txt",'w')  # запис у текстовому режимі
f = open("img.bmp",'r+b') # читання та запис у двійковому режимі
```

На відміну від інших мов, символ **a** не має на увазі число 97, доки він не закодований з використанням ASCII (або інших еквівалентних кодувань).

```
f = open("test.txt", mode='r', encoding='utf-8')
```

Закриття файлу

Коли ми закінчили виконання операцій із файлом, нам потрібно правильно закрити файл.
Закриття файлу звільнить ресурси, пов'язані з файлом. Це робиться за допомогою методу close ().
У Python є збирач сміття для очищення об'єктів, на які немає посилань, але ми не повинні покладатися на нього під час закриття файлу.

```
f = open("test.txt", encoding = 'utf-8')
# якісь операції з файлом...
f.close()
```

Цей спосіб **не** зовсім **безпечний** (читати: небезпечний). Якщо під час виконання будь-якої операції з файлом виникає виняток, код завершується без закриття файлу.
Більш безпечний спосіб – використовувати блок try...finallly.

```
try:
   f = open("test.txt", encoding = 'utf-8')
   # якісь операції з файлом...
finally:
   f.close()
```

Таким чином, ми гарантуємо, що файл буде правильно закритий, навіть якщо виникне виняток, який призводить до припинення виконання програми.
Найкращий спосіб закрити файл - використовувати оператор with. Це гарантує, що файл буде закрито під час виходу з блоку всередині оператора with.
Нам не слід явно викликати метод close(). Це робиться зсередини:

```
with open("test.txt", encoding = 'utf-8') as f:
   pass
   # якісь операції з файлом...Запись в файл
```

Щоб записати дані до файлу, потрібно відкрити його в режимі запису w, додавання a або ексклюзивного створення (**exclusive creation**) x.

CyberBionic systematics

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Нам потрібно бути обережними з режимом w, тому що файл буде повністю перезаписано, якщо він вже існує. Через це стираються всі попередні дані.
Запис рядка чи послідовності байтів (для двійкових файлів) виконується за допомогою методу write(). Цей метод повертає кількість символів, записаних до файлу.

```
with open("test.txt",'w',encoding = 'utf-8') as f:
   f.write("my first file\n")
   f.write("This file\n\n")
   f.write("contains three lines\n")
```

Ця програма створить новий файл із ім'ям test.txt у поточному каталозі, якщо він не існує. Якщо він існує, він перезаписується.
Ми повинні самі включати символи нового рядка, щоб розрізняти різні рядки або використовувати метод f.writeline().

Читання з файлу

Щоб прочитати файл, ми повинні відкрити файл у режимі читання r.
Для цього доступні різні методи. Ми можемо використовувати метод read(size) для читання кількості даних розміру. Якщо параметр розміру не вказано, читання виконується до кінця файлу.
Ми можемо прочитати файл text.txt, який ми написали в попередньому розділі, наступним чином:

```
>>> f = open("test.txt",'r',encoding = 'utf-8')
>>> f.read(4)
'This'

>>> f.read(4)
' is '

>>> f.read()
'my first file\nThis file\ncontains three lines\n'

>>> f.read()
```

Метод read() повертає новий рядок як '\n'. Після досягнення кінця файлу при подальшому читанні ми отримуємо порожній рядок.
Ми можемо змінити поточний курсор у файлі (**позицію**) за допомогою методу seek(). Так само метод tell() повертає нашу поточну позицію (в байтах).

```
>>> f.tell()
56

>>> f.seek(0)
0

>>> print(f.read())
This is my first file
This file
contains three lines
```

Ми можемо читати файл по рядку, використовуючи цикл for. Це водночас і ефективно, і швидко.

```
>>> for line in f:
...     print(line, end = '')
...
This is my first file
This file
contains three lines
```

У цій програмі рядки вже у самому файлі містять символ нового рядку \n. Отже ми використовуємо параметр end="" функції print(), щоб уникнути появи двох символів нового рядка під час друку.
У якості альтернативи ми можемо використовувати метод readline() для читання окремих рядків файлу. Цей метод читає файл до нового рядка, включаючи символ нового рядка.

CyberBionic systematics

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

```
>>> f.readline()
'This is my first file\n'

>>> f.readline()
'This file\n'

>>> f.readline()
'contains three lines\n'

>>> f.readline()
''
```

Нарешті, метод readlines() повертає список рядків всього файлу, котрі залишилися. Всі ці методи читання повертають порожні значення при досягненні кінця файлу (**EOF**).

```
>>> f.readlines()
['This is my first file\n', 'This file\n', 'contains three lines\n']
```

Методи об'єкта файлу

Файловий об'єкт має різні методи. Деякі з них були використані у наведених вище прикладах.
Ось повний список методів у текстовому режимі з коротким описом:

Методи файлового дескриптору

Метод	Опис
close()	Закриває відкритий файл. Не діє, якщо файл закрито.
detach()	Відокремлює базовий двійковий буфер від TextIOBase та повертає його.
fileno()	Повертає ціле число (дескриптор файлу) файлу.
flush()	Очищає буфер запису файлового потоку.
isatty()	Повертає True, якщо файловий потік є інтерактивним.
read(n)	Читає не більше n символів із файлу. Читає до кінця файлу, якщо він є негативним або None.
readable()	Повертає True якщо файловий потік можна читати.
readline(n=-1)	Читає та повертає один рядок із файлу. Зчитує не більше n байтів, якщо зазначено.
readlines(n=-1)	Читає та повертає список рядків із файлу. Зчитує не більше n байтів/символів, якщо вказано.
seek(offset,from=SEEK_SET)	Змінює позицію файлу на зміщення у байтах відносно from (початок, поточне, кінець).
seekable()	Повертає True, якщо файловий потік підтримує довільний доступ.
tell()	Повертає поточне розташування файлу.
truncate(size=None)	Змінює розмір файлового потоку до байтів. Якщо розмір не вказано, змінюється до поточного розташування.
writable()	Повертає True, якщо файловий потік може бути записано.
write(s)	Записує рядок s у файл та повертає кількість записаних символів.
writelines(lines)	Записує список рядків у файл.

Закріплення матеріалу

- Що таке файл?
- Що таке файловий об'єкт (потік)?
- За допомогою якої вбудованої функції можна відкрити файл?
- Для чого потрібно закривати файли?
- За допомогою якої конструкції Python можна автоматично закривати файли після того, як вони більше не потрібні?
- Які є режими відкриття файлів?
- Як прочитати дані з файлу?
- Як записати дані до файлу?

CyberBionic systematics

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Додаткове завдання

Завдання
Створіть список товарів в інтернет-магазині. Серіалізуйте його за допомогою pickle та збережіть у JSON

Самостійна діяльність учня

Завдання 1
Напишіть скрипт, який створює текстовий файл і записує до нього 10000 випадкових дійсних чисел. Створіть ще один скрипт, який читає числа з файлу та виводить на екран їхню суму.

Завдання 2
Модифікуйте вихідний код сервісу зі скорочення посилань із попередніх двох уроків так, щоб він зберігав базу посилань на диску і не «забув» при перезапуску. За бажанням можете ознайомитися з модулем shelve (https://docs.python.org/3/library/shelve.html), який у даному випадку буде дуже зручним та спростить виконання завдання.

Рекомендовані ресурси

https://docs.python.org/3/tutorial/inputoutput.html
https://docs.python.org/3/reference/datamodel.html#context-manager
https://docs.python.org/3/library/functions.html#open

Статті у Вікіпедії про ключові поняття, розглянуті на цьому уроці
https://uk.wikipedia.org/wiki/Файл

CyberBionic systematics

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com