Файловый ввод и вывод

Файл — это всего лишь набор данных, сохраненный в виде последовательности битов на компьютере. Информация хранится в куче данных (структура данных) и имеет название «имя файла» (filename).

В Python существует два типа файлов:

- Текстовые
- Бинарные

Текстовые файлы

В них хранятся последовательности символов, которые понимает человек.

Текст может храниться в двух форматах: (.txt) — простой текст и (.rtf) — «формат обогащенного текста».

Бинарные файлы

В бинарных файлах данные отображаются в закодированной форме (с использованием только нулей (0) и единиц (1) вместо простых символов). В большинстве случаев это просто последовательности битов.

Они хранятся в формате .bin.

Открытие файла

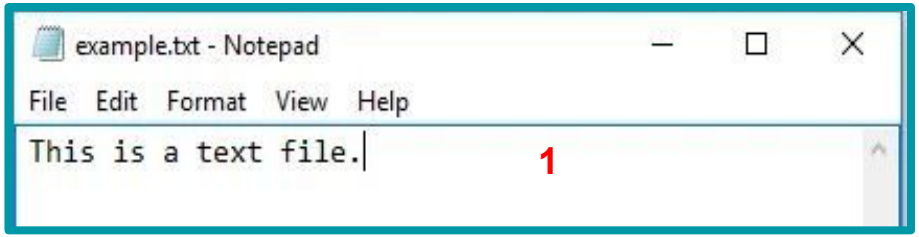
Метод open()

```
f = open(file_name, access_mode)
```

С ее помощью можно открыть любой файл на компьютере. Технически Python создает на его основе объект.

Режим	Описание
r	Только для чтения.
w	Только для записи. Создаст новый файл, если не найдет с указанным именем.
rb	Только для чтения (бинарный).
wb	Только для записи (бинарный). Создаст новый файл, если не найдет с указанным именем.
r+	Для чтения и записи.
rb+	Для чтения и записи (бинарный).
w+	Для чтения и записи. Создаст новый файл для записи, если не найдет с указанным именем.
wb+	Для чтения и записи (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.
a	Откроет для добавления нового содержимого. Создаст новый файл для записи, если не найдет с указанным именем.
a+	Откроет для добавления нового содержимого. **Создаст новый файл** для чтения записи, если не найдет с указанным именем.
ab	Откроет для добавления нового содержимого (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.
ab+	Откроет для добавления нового содержимого (бинарный). Создаст новый файл для чтения записи, если не найдет с указанным именем.

Пример




```
f = open('example.txt','r')  # открыть файл из рабочей директории в режиме чтения

fp = open('C:/xyz.txt','r')  # открыть файл из любого каталога
```

2

```
>>> print(*f) # выводим содержимое файла

This is a text file.

>>> print(f) # выводим объект

<_io.TextIOWrapper name='example.txt' mode='r' encoding='cp1252'>
```

3

Закрытие файла

Метод close()

После открытия файла в Python его нужно закрыть. Таким образом освобождаются ресурсы и убирается мусор. Python автоматически закрывает файл, когда объект присваивается другому файлу.

Способ №1

```
f = open('example.txt','r')
# работа с файлом
f.close()
```

Способ №2

Вот как сделать это исключение:

```
f = open('example.txt','r')
try:
   # работа с файлом
finally:
   f.close()
```

Способ №3

Инструкция with

Вот как это реализовать в коде.

```
with open('example.txt') as f:
     # работа с файлом
```

Чтение и запись файлов в Python

Функция read()

Функция read() используется для чтения содержимого файла после открытия его в режиме чтения (r).

Синтаксис

```
file.read(size)
```

Пример

```
>>> f = open('example.txt','r')
>>> f.read(7)  # чтение 7 символов из example.txt
'This is '
```

Функция readline()

Функция readline() используется для построчного чтения содержимого файла.

```
>>> x = open('test.txt','r')
>>> x.readline()  # прочитать первую строку
This is line1.
>>> x.readline(2)  # прочитать вторую строку
This is line2.
>>> x.readlines()  # прочитать все строки
['This is line1.','This is line2.','This is line3.']
```

Функция write()

Функция write() используется для записи в файлы Python, открытые в режиме записи.

Синтаксис

```
file.write(string)
```

Пример

```
>>> f = open('xyz.txt','w')  # открытие в режиме записи
>>> f.write('Hello \n World')  # запись Hello World в файл
Hello
World
 >>> f.close()  # закрытие файла
```

Переименование файлов в Python

Функция rename()

Используется для переименования файлов в Python. Для ее использования сперва нужно импортировать модуль os.

Синтаксис

```
import os
os.rename(src,dest)
```

Пример

```
>>> import os
>>> # переименование xyz.txt в abc.txt
 >>> os.rename("xyz.txt","abc.txt")
```

Текущая позиция в файлах Python

В Python возможно узнать текущую позицию в файле с помощью функции tell(). Таким же образом можно изменить текущую позицию командой seek().

Пример

```
>>> f = open('example.txt')  # example.txt, который мы создали ранее
>>> f.read(4)  # давайте сначала перейдем к 4-й позиции
This
>>> f.tell()  # возвращает текущую позицию
4
 >>> f.seek(0,0)  # вернем положение на 0 снова
```

Методы файла в Python

`file.close()`	закрывает открытый файл
`file.fileno()`	возвращает целочисленный дескриптор файла
`file.flush()`	очищает внутренний буфер
`file.isatty()`	возвращает True, если файл привязан к терминалу
`file.next()`	возвращает следующую строку файла
`file.read(n)`	чтение первых n символов файла
`file.readline()`	читает одну строчку строки или файла
`file.readlines()`	читает и возвращает список всех строк в файле
file.seek(offset[,whene])	устанавливает текущую позицию в файле
`file.seekable()`	проверяет, поддерживает ли файл случайный доступ. Возвращает `True`, если да
`file.tell()`	возвращает текущую позицию в файле
`file.truncate(n)`	уменьшает размер файл. Если n указала, то файл обрезается до n байт, если нет — до текущей позиции
`file.write(str)`	добавляет строку `str` в файл
`file.writelines(sequence)`	добавляет последовательность строк в файл

Спасибо за внимание!