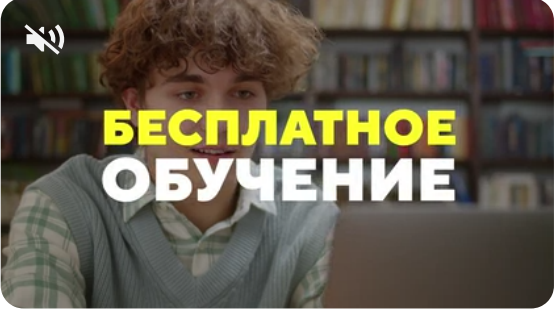


Модуль io, операции ввода/вывода в Python



synergyacademy.com

РЕКЛАМА

Бесплатные Курсы IT для 8-11 классов и Колледжистов
Бесплатные курсы программирования для 8-11 классов и студентов колледжей!
[Узнать больше](#)

[Стандартная библиотека Python3.](#) / Модуль io, операции ввода/вывода в Python

Ввод/вывод текста, двоичных и необработанных потоков

[Модуль io](#) предоставляет основные средства Python для работы с различными типами ввода-вывода. Существует три основных типа ввода-вывода: Text I/O - [текстовый](#) ввод-вывод, Binary I/O [двоичный](#) ввод-вывод и Raw I/O - необработанный ввод-вывод. Конкретный объект, принадлежащий любой из этих категорий, называется [файловым объектом](#).

Независимо от своей категории каждый конкретный объект потока также будет иметь различные возможности: он может быть доступен только для чтения, только для записи или для чтения и записи. Он также может разрешать произвольный произвольный доступ (поиск вперед или назад в любом месте) или только последовательный доступ в случае сокета или канала.

Все потоки тщательно следят за типом данных, которые им предоставляются. Например, передача [объекта str](#) методу [write\(\)](#) [двоичного потока](#) вызовет [исключение TypeError](#). То же самое будет, если передать [объект bytes](#) методу [write\(\)](#) [текстового потока](#).

Содержание:

- [Text I/O - текстовый ввод-вывод;](#)
- [Binary I/O - двоичный ввод-вывод;](#)
- [Raw I/O - необработанный ввод-вывод;](#)
- [Кодировка текста;](#)
 - [Включение EncodingWarning.](#)

Text I/O - текстовый ввод-вывод:

Текстовый ввод/вывод ожидает и производит [объекты типа str](#). Это означает, что каждый раз, когда хранилище изначально состоит из байтов, например в случае файла, кодирование и декодирование данных выполняется прозрачно, а также учитывается необязательный перевод специфичных для платформы символов новой строки.

Самый простой способ создать текстовый поток с помощью [встроенной функции open\(\)](#), при желании, указав кодировку:

```
fp = open("myfile.txt", "r", encoding="utf-8")
```

Текстовые потоки в памяти также доступны в виде [объектов io.StringIO](#):

```
fp = io.StringIO("some initial text data")
```

API текстового потока подробно описан в документации по [io.TextIOBase](#).

Binary I/O - двоичный ввод-вывод:

Двоичный ввод/вывод, также называемый буферизованным вводом/выводом ожидает объекты, похожие на байты, и создает [объекты типа bytes](#). Кодирование, декодирование или перевод новой строки не выполняется. Эта категория потоков может использоваться для всех видов нетекстовых данных, а также когда требуется ручное управление обработкой текстовых данных.

Самый простой способ создать двоичный поток - использовать [встроенную функцию open\(\)](#) в режиме mode='b':

```
fp = open("myfile.jpg", "rb")
```

Двоичные потоки в памяти также доступны в виде [объектов io.BytesIO](#):

```
fp = io.BytesIO(b"some initial binary data: \x00\x01")
```

API двоичного потока подробно описан в документации по [io.BufferedIOBase](#). Другие модули могут предоставлять дополнительные способы создания текстовых или двоичных потоков.

Raw I/O – необработанный ввод-вывод:

Необработанный ввод/вывод, также называемый небуферизованным вводом/выводом обычно используется как низкоуровневый строительный блок для двоичных и текстовых потоков. Очень редко встречается необходимость напрямую манипулировать необработанным потоком из пользовательского кода. Тем не менее, вы можете создать необработанный поток, открыв файл в двоичном режиме `mode='rb'` с отключенной буферизацией `buffering=0`:

```
fp = open("myfile.jpg", "rb", buffering=0)
```

API необработанного потока подробно описан в документации по [io.RawIOBase](#).

Кодировка текста.

Кодировка по умолчанию для [io.TextIOWrapper](#) и встроенной [функции open\(\)](#) зависит от локали ([locale.getpreferredencoding\(False\)](#)).

Многие разработчики не указывают кодировку при открытии текстовых файлов, закодированных в UTF-8 (например, JSON, TOML, Markdown и т. д.), так как в Unix по умолчанию используют локаль UTF-8. Но, для большинства пользователей Windows, это вызывает ошибки, так как кодировка локали не является UTF-8. Например:

```
# Может не работать в Windows, если в
# файле есть символы, отличные от ASCII.
with open("README.md") as f:
    long_description = f.read()
```

Кроме того, хотя пока нет конкретного плана, Python может изменить кодировку текстового файла по умолчанию на UTF-8 в будущем.

Соответственно, настоятельно рекомендуется явно указывать кодировку при открытии текстовых файлов. Если необходимо использовать UTF-8, то нужно передавать `encoding='utf-8'`. Чтобы использовать текущую кодировку языкового стандарта, то **Python 3.10 поддерживает** передачу аргумента `encoding='locale'`.

Если нужно запустить существующий код в Windows, который пытается открыть файлы UTF-8 с использованием кодировки локали по умолчанию, то можно включить режим UTF-8.

Включение EncodingWarning.

Новое в Python 3.10

Чтобы найти, где в коде используется кодировка локали по умолчанию, можно включить [параметр командной строки](#) `-X warn_default_encoding` или установить [переменную среды](#) `PYTHONWARNDEFAULTENCODING`, которая будет выдавать предупреждение [EncodingWarning](#) при использовании кодировки по умолчанию.

Если код предоставляет API, который использует функцию `open()` или `io.TextIOWrapper` и при этом передает `encoding=None` в качестве аргумента, то можно использовать функцию [io.text_encoding\(\)](#), чтобы вызывающие API генерировали `EncodingWarning`, если они не передают кодировку.

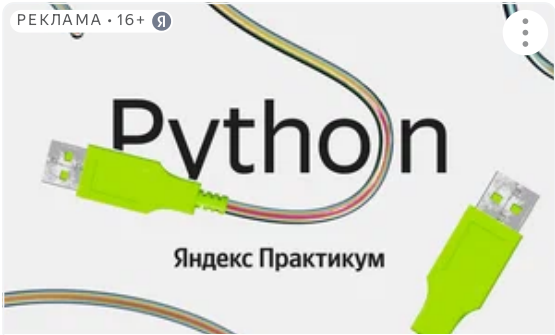
Для новых API, необходимо рассматривать возможность использования UTF-8 по умолчанию (т.е. `encoding='utf-8'`).

Содержание раздела:
<ul style="list-style-type: none">КРАТКИЙ ОБЗОР МАТЕРИАЛА.Интерфейс высокого уровня модуля ioИерархия классов модуля ioКласс io.IOBase() модуля ioКласс io.RawIOBase() модуля ioКласс io.BufferedIOBase() модуля ioКласс io.FileIO модуля ioКласс io.BytesIO() модуля ioЧтение и запись буферизованных потоковКласс io.TextIOBase() модуля ioКласс io.TextIOWrapper модуля ioФункция io.StringIO() модуля ioПример использования io.StringIO()

- [Пример использования io.BytesIO\(\)](#)
- [Пример использования io.TextIOWrapper\(\)](#)


ХОЧУ ПОМОЧЬ
ПРОЕКТУ

РЕКЛАМА · 16+ ⓘ



Python

Яндекс Практикум

practicum.yandex.ru

Профессия: Python-разработчик. Курс от Яндекса.

5,0

★

Рейтинг организации ⓘ

Поможем освоить новую профессию с нуля за 9 месяцев. Начните учиться бесплатно!

Курсы программирования

>

Трудоустройство

>

Программа

>

Узнать больше