



Основы языка Python. Интерактивный
курс

file

План

- Функция open. Ее параметры.
- Запись текста в файл.
- Чтение текста из файла.
- Заккрытие файла.
- Работа через with.



Функция open

- открывает файл в указанном режиме
- `f = open('my.txt', 'w')`
- `file` — имя файла
- `mode` — режим
- `encoding` — кодировка



Режимы открытия mode

- r — чтение
- w — запись, если файла нет, создается новый
- x — запись, если файла нет, ошибка
- a — дозапись
- b — двоичный режим
- + — открытие на чтение и запись



Запись текста в файл

- `write` — записать строку в файл
- `writelines` — записать список строк в файл
- `\n` — символ конца строки



Чтение из файла

- `read` — чтение всего файла
- `for line in f:` — чтение файла построчно
- `readlines` — чтение файла в список строк



Заккрытие файла

- После работы с файлом его необходимо закрывать.
- Открытые файлы тратят ресурсы системы.
- `f.close()`.
- Если до `close` произойдет исключительная ситуация, файл не будет закрыт.
- Удобным вместо `close()` является использование `with`.



Python



Строки байт. Кодировки

План

- Строки байт.
- Пример работы со строками байт.
- Перевод строки `str` в байты.
- Перевод байт в строку `str`.
- Кодировки.



Типы строк в Python

- `str` — обычные строки
- `bytes` — строки байт
- `bytearray` — изменяемая строка байт



Действия со строками байт

- индекс `sb[0]`
- срез `sb[1:3]`
- ...



Как строка хранится в памяти?

- Любая информация хранится в памяти как набор 0 и 1.
- Строки не являются исключением.
- Каждому символу ставится в соответствие определенный код (число).
- Коды могут быть разные и зависят от кодировки.



Основные кодировки

- `ascii` — американские символы.
- `latin-1` — европейские символы.
- `utf-8` — универсальная кодировка для большинства языков.
- Чем универсальнее кодировка, тем больше байт требуется для кодирования одного символа.



ascii — таблица

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	\$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(72	48	110	H	104	68	150	h
9	9	11		41	29	51)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	
29	1D	35		61	3D	75	=	93	5D	135]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	



Перевод строки в байты (кодирование)

- `'Hello world'.encode('utf-8')`.
- При переводе строки `str` в байты `bytes` указываем кодировку.
- Кодировка должна поддерживать символы нужного нам алфавита.



Перевод байт в строку (декодирование)

- `sb.decode('utf-8')`.
- Указываем кодировку, которой мы кодировали строку.



Python



Запись и чтение байтов

План

- Работа с файлом в режиме байтов.
- Параметр `encoding`.
- Запись байтов в файл.
- Чтение байтов из файла.
- Примеры использования.



Работа с файлом в режиме байтов

- `open('filename', 'wb')` — режим записи байтов
- `open('filename', 'rb')` — режим чтения байтов
- параметр `encoding` определяет кодировку
- `open('filename', 'w', encoding='utf-8')`



Запись байтов в файл

- `f.write(b'some bytes')` — файл открыт в режиме `wb`
- `f.write('some str')` — файл открыт в режиме `w`
- в любом случае информация хранится в виде нулей и единиц



Чтение байтов из файла

- `f.read()` - файл открыт в режиме `rb` — читаем байты
- `f.read()` - файл открыт в режиме `r` — читаем строки





Python

pickle

План

- Сериализация.
- Запись сложного объекта в файл.
- Модуль pickle.
- Основные функции.
- Примеры применения.



Сериализация

- Процесс преобразования объекта в поток байтов для сохранения или передачи в память, базу данных или файл.
- Обратный процесс — десериализация.



Применение сериализации

- сохранение объекта в файл
- сохранение объекта в базу данных
- передача объекта по сети
- ...



Способы записи объекта в файл

- ручной (создание велосипеда)
- универсальный pickle



Ручной способ

```
{'name': 'Max', 'phones': [123, 345]}
```

Как перевести такой объект (dict) в байты для сохранения в файл?

- Придумать способ приведения объекта к более простым.
- Придумать свой формат хранения.



Ручной способ (Недостатки)

- Не универсальный.
- При небольшом изменении объекта изменится весь алгоритм.
- Надо помнить, как мы делали сохранение, чтобы потом прочитать.
- Трудоемкий.



Модуль pickle

- Сохраняет сложные объекты в файл.
- Преобразует сложные объекты в байты.
- Встроен в Python.



pickle. Основные функции

- dump — сохранение объекта в файл
- dumps — преобразование объекта в байты
- load — загрузка объекта из файла
- loads — загрузка объекта из набора байт





Python

json

План

- Формат json. Применение.
- Модуль json в Python. Применение.
- Основные функции.
- Примеры.



Формат json

- JavaScript Object Notation.
- Текстовый формат обмена данными, основанный на JavaScript.
- Аналогичен набору словарей, списков, простых типов данных в Python.
- Является просто текстом (строкой).



Применение

- хранение данных
- передача данных
- чаще всего используется в web-разработке для передачи данных по протоколу http



json в Python

- Основные структуры Python схожи с форматом.
- Требуется только преобразование в строку и обратно.
- Этим занимается модуль json.
- `import json.`



json. Основные функции

- `dump` — сохранение объекта в формате json в файл
- `dumps` — преобразование объекта в json (в текст)
- `load` — загрузка объекта из файла
- `loads` — загрузка объекта из формата json (строки)



Задача

- Передать список любимых песен и их исполнителей своему другу, разработчику C#.

