при поддержке
{а}вижн

Лекция 12

Перевод и адаптация материалов Джона ДеНиро (John DeNero). Используется с разрешения автора.

Объекты

(Пример)

Объекты

Объект — это представление информации.

Он состоит из данных и алгоритмов, объединенных для создания абстракций.

Объекты могут представлять вещи, свойства, взаимодействия и процессы.

Тип объекта называют классом; в Python’е они рассматриваются как значения.

Объектно-ориентированное программирование:

— Метафора для организации больших программ

— Особый синтаксис может улучшить структуру программы

В Python’е каждое значение — это объект.

— У всех объектов есть атрибуты.

— Большая часть обработки данных происходит в методах объекта.

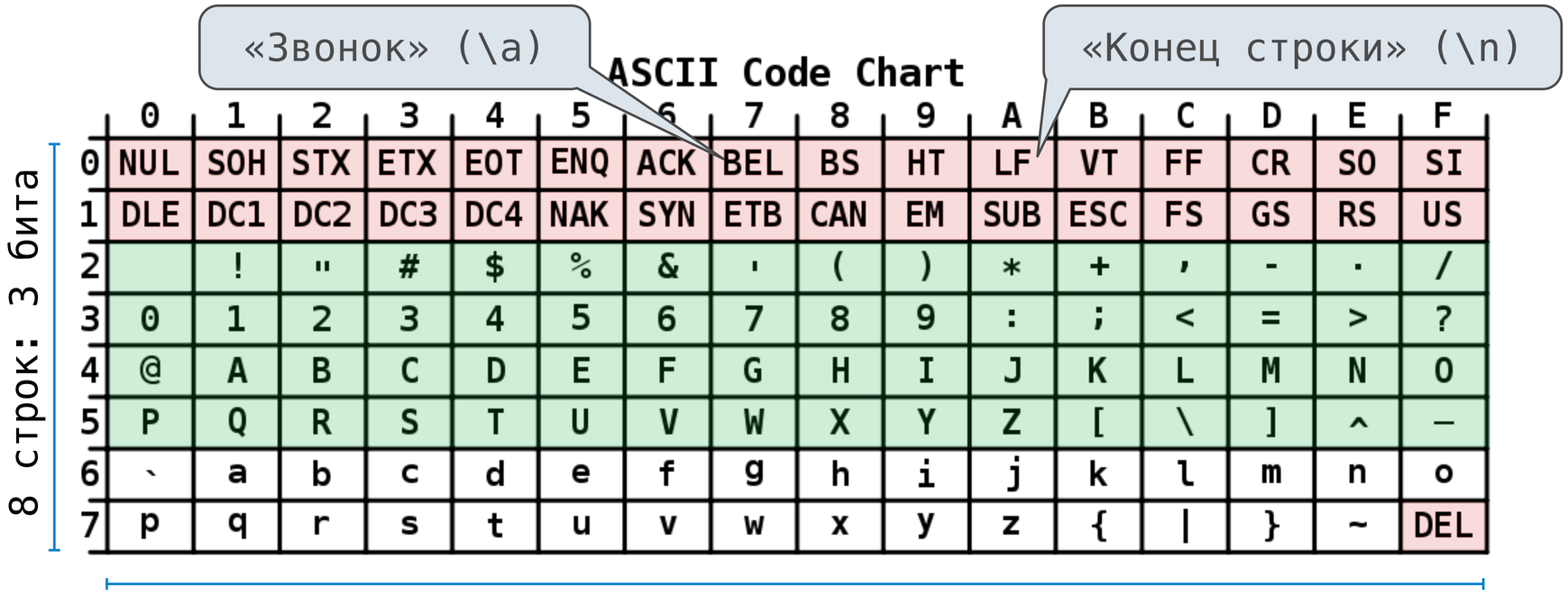
— Функции делают одну вещь; объекты делают множество связанных вещей.

Пример: Строки

(Пример)

Кодирование строк: стандарт ASCII

American Standard Code for Information Interchange



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

- Размещение было выбрано для поддержки сортировки по коду символа
- Строки 2–5 удобны для 6-битного подмножества (64 элемента)
- Управляющие символы были предусмотрены для передачи данных

(Пример)

Кодирование строк: стандарт Unicode

- 109,000 знаков
- 93 языка (упорядочены)
- Набор свойств знака (например, заглавность или строчность)
- Поддержка двунаправленного порядка вывода
- Каноническое имя для каждого знака

http://ian-albert.com/unicode_chart/unichart-chinese.jpg

U+0058 LATIN CAPITAL LETTER X

U+263a WHITE SMILING FACE

U+2639 WHITE FROWNING FACE

(Пример)

Кодирование строк: кодировка UTF-8

UTF (UCS (Universal Character Set) Transformation Format)

Unicode: Связь между знаками и целыми числами

UTF-8: Связь между этими целыми и байтами

Байт — это 8 бит. Байт может кодировать любое целое в интервале 0–255.

байты	целые
00000000	0
00000001	1
00000010	2
00000011	3

Код переменной длины: представление чисел различается количеством битов необходимых для их кодирования.

В Python’e: длина строки измеряется в знаках, длина представления в байтах.

(Пример)

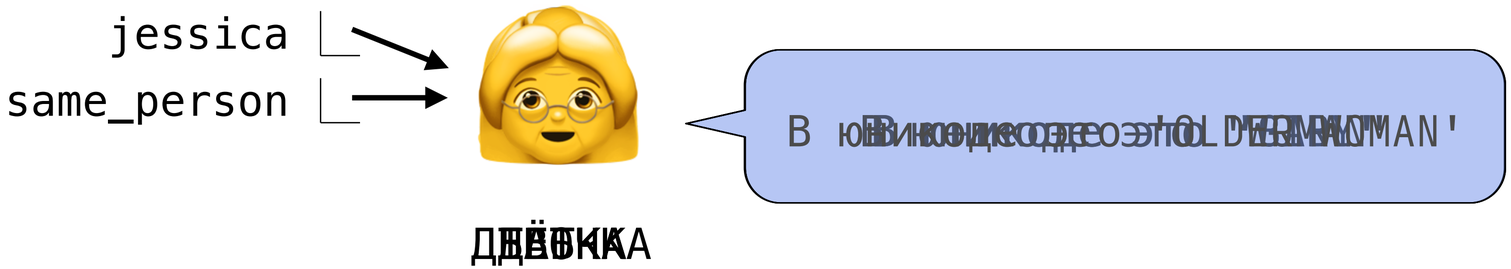
Изменение состояния объекта

Некоторые объекты могут меняться

[Пример]

Первый пример курса об изменении состояния объекта.

Объект может существенно изменяться в процессе вычисления:



Все имена ссылающиеся на один объект подвержены изменениям.

Объекты только *изменяемых* типов могут меняться: списки и словари.

{Пример}

Изменения могут произойти внутри вызова функции

Функция может изменить любой объект в области видимости.

```
>>> four = [1, 2, 3, 4]
>>> len(four)
4
>>> mystery(four)
>>> len(four)
2
```

```
def mystery(s):
    s.pop()
    s.pop()
```

или

```
def mystery(s):
    s[2:] = []
```

```
>>> four = [1, 2, 3, 4]
>>> len(four)
4
>>> another_mystery() # Без аргументов!
>>> len(four)
2
```

```
def another_mystery():
    four.pop()
    four.pop()
```

Интерактивная диаграмма

Таплы

(Пример)

Таплы — это неизменяемые последовательности

Неизменяемые значения не могут изменяться!

```
>>> turtle = (1, 2, 3)
>>> ooze()
>>> turtle
(1, 2, 3)
```

Следующая лекция: ooze может изменить связь имени turtle

```
>>> turtle = [1, 2, 3]
>>> ooze()
>>> turtle
['Здесь может быть что угодно!']
```

Значение выражения может измениться из-за изменений имён или объектов

Изменение имени:

```
>>> x = 2
>>> x + x
4
>>> x = 3
>>> x + x
6
```

Изменение объекта:

```
>>> x = [1, 2]
>>> x + x
[1, 2, 1, 2]
>>> x.append(3)
>>> x + x
[1, 2, 3, 1, 2, 3]
```

Неизменяемая последовательность всё же может измениться если *содержит* изменяемое значение.

```
>>> s = ([1, 2], 3)
>>> s[0] = 4
ОШИБКА
```

```
>>> s = ([1, 2], 3)
>>> s[0][0] = 4
>>> s
([4, 2], 3)
```

Изменение

Одинаковость и изменяемость

- Если отбросить изменяемость объектов, составной объект будет просто объединением своих частей.
- Рациональное число — это просто числитель и знаменатель.
- При учёте изменяемости появляется дополнительное свойство.
- Составной объект приобретает «идентичность» в дополнение к частям, из которых он состоит.
- Список остается «тем же» списком, даже если его содержимое полностью поменялось.
- Противоположно: два списка с одинаковым содержанием, остаются различными.

```
>>> a = [10]
>>> b = a
>>> a == b
True
>>> a.append(20)
>>> a == b
True
>>> a
[10, 20]
>>> b
[10, 20]
```

```
>>> a = [10]
>>> b = [10]
>>> a == b
True
>>> b.append(20)
>>> a
[10]
>>> b
[10, 20]
>>> a == b
False
```

Оператор идентичности

Идентичность

`<exp0> is <exp1>`

вернёт `True` если результаты обоих выражений `<exp0>` и `<exp1>` укажут на один объект

Равенство

`<exp0> == <exp1>`

вернет `True` если результаты обоих выражений `<exp0>` и `<exp1>` будут равными значениями

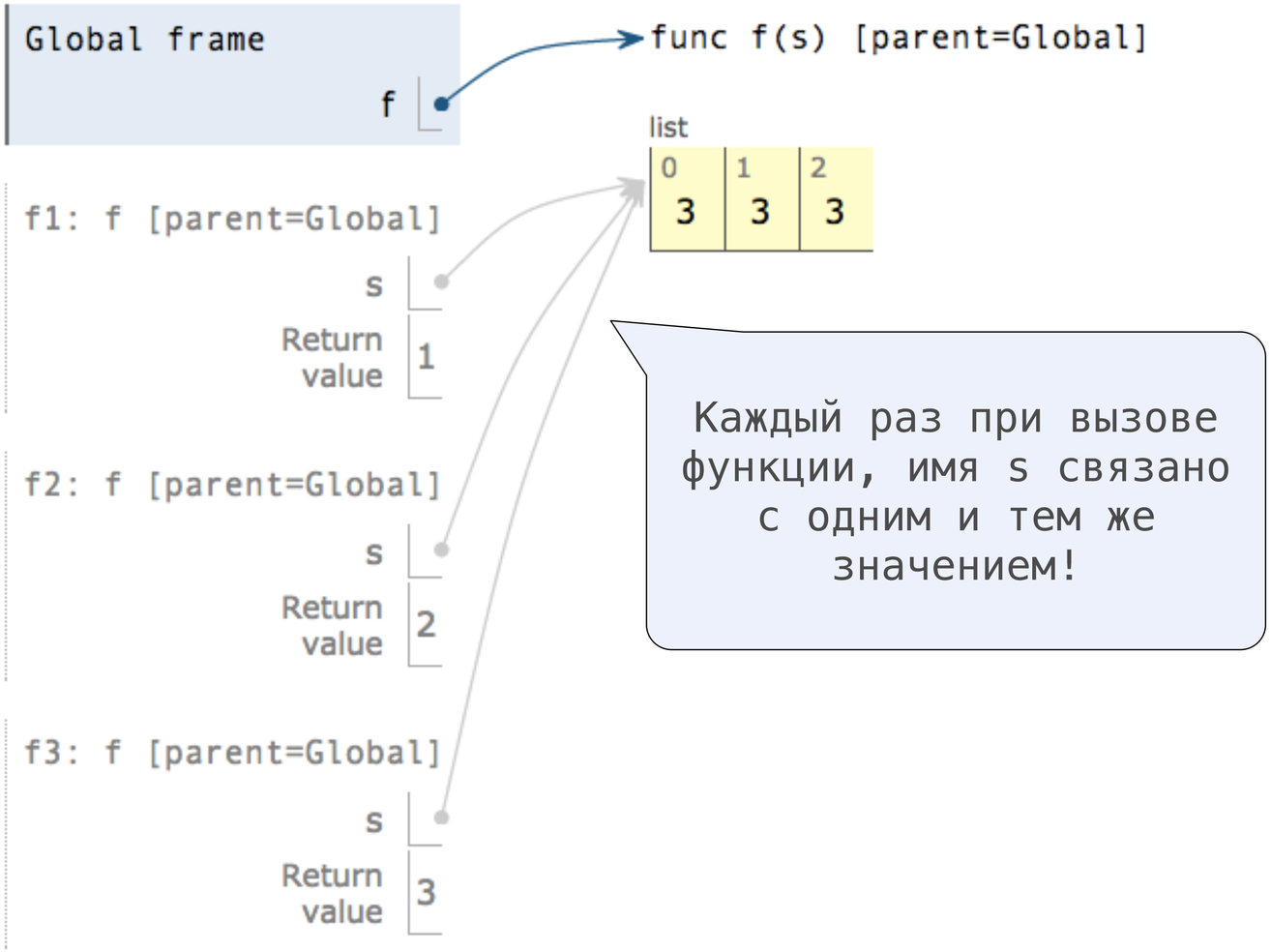
Идентичные объекты всегда равны

(Пример)

Изменения аргументов по-умолчанию опасны!

Значение аргумента по-умолчанию является частью функции и не создается при вызове.

```
>>> def f(s=[]):
...     s.append(3)
...     return len(s)
...
>>> f()
1
>>> f()
2
>>> f()
3
```



Интерактивная диаграмма