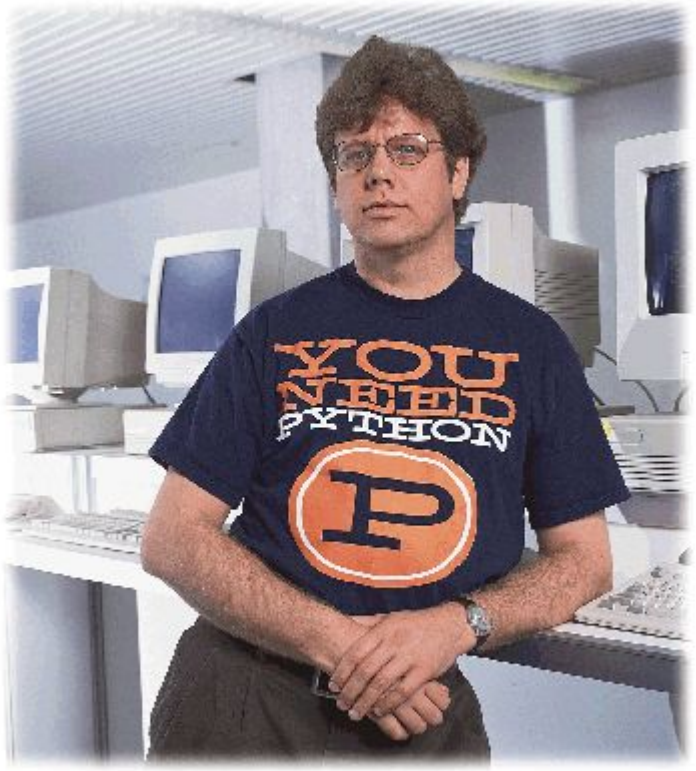
Advanced Python. Introduction.

Цели курса

Обзор курса, блок 1: разработка на Python

- Язык программирования Python
- Стандартная библиотека
- Структуры данных
- Экосистема Python

Python



- 1991 - Гвидо ван Россум опубликовал исходники первой версии
- 1994 - 1.0
- 2000 - 2.0
- 2001 - Python Software Foundation
- 2008 - 3.0

Python

- динамический
- интерпретируемый (?)
- объектно - ориентированный (??)
- функциональный (???)
- **мульти-парадигменный** (multiparadigm)

Аналоги

- Perl
- Ruby
- PHP

Почему Python

- Скорость разработки
- Переносимость
- Универсальность
- Расширяемость

Области применения

- Системное программирование, администрирование
- Научные вычисления
- Веб/интернет
- Образование
- Десктоп приложения
- Встраиваемые системы скриптования

Крупные пользователи

- Google
- Facebook
- Instagram
- Dropbox
- Mozilla foundation
- Disqus
- Microsoft

Реализации

- CPython
- PyPy
- IronPython
- Jython

Python 2 vs Python 3

- Проблемы Python 2

```
UnicodeDecodeError: 'ascii' codec can't decode
byte 0xff in position 6: ordinal not in range(128)
```

- Решение: Python 3
- Реальность: Python 2/3

Python 2 vs Python 3

- Unicode: str, byte, bytearray
- print/print()
- range/xrange
- Целочисленное деление
- Синтаксис raise

Интерпретатор

- Интерактивная разработка
- REPL: read, eval, print, loop.
- Hello world:

```
>>> print(“Hello, world!”)
Hello, world!
```

- help(), dir()
- Бонус: философия Python

```
>>> import this
```

Исполнение программ

```
$ cat hello.py:
#!/usr/bin/env python
print(“Hello, world!”)

$ python hello.py
Hello, world!
```

Ключевые слова

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	break
except	in	raise		

Объекты, типы

- В Python все - объекты
- Типы, классы. type()
- Идентичность (identity). Функция id()
- Неизменяемые типы (immutable): String, Integer, Tuple
- Изменяемые (mutable) типы: Dictionary, List, Set. Контейнерные типы.

Литералы

```
>>> 42
42
>>> 'Hello, world!'
'Hello, world!'
>>> True
True
>>> False
False
>>> None
>>>
```

Литералы

```
>>> [1, 2, 3]
[1, 2, 3]
>>> {'a', 'b', 'c', 'a'}
{'b', 'c', 'a'}
>>> ('a', 1, True)
('a', 1, True)
>>> {'a': 1, 'b': 2}
{'a': 1, 'b': 2}
```

Переменные, присваивание

```
answer = 42
greeting = 'Hello, world!'
cond = True
somevar = None

person = {'name': 'Random J. Hacker',
          'age': 25}

result = math.pi * r ** 2
a, b = b, a + b
first, *rest = range(10)
```

Операторы

- Перегрузка операторов
- Логические: and, or, not
- Identity: is, is not
- Арифметические операции: +, -, *, /, %, **, +=, -=, *=
- Арифметическое сравение: <, >, ==, !=, >=, <=
- Строковые: +, +=

Операторы (коллекции)

- in, not in
- +, *
- [i], [i:j], [i:j:k]
- len, min, max
- .index, .count

Порядок операций

В Python применяются обычные математические правила:

- скобки - ()
- возведение в степень - **
- умножение, деление
- сложение, вычитание

Все остальные операции с одинаковым приоритетом выполняются слева направо.

Комментарии

Однострочный комментарий:

```
# TODO: write test
```

Многострочный комментарий:

```
# The code below does some magic,
# which I don’t understand. You
# should not touch it.
```

Блоки, структура

- В Python для создания блочной структуры программы используются отступы - табуляция или пробелы
- Не смешивайте их никогда!
- В любой непонятной ситуации читайте PEP8

Условный оператор if

Общий вид:

```
if expression1:
    statement
elif expression2:
    statement
else:
    statement
```

Условный оператор if

Пример:

```
distance = 150
if distance < 100:
    action = 'melee'
elif distance >= 100 and distance < 200:
    action = 'shooting'
else:
    action = 'wait'
```

Циклы: while, for

Общий вид:

```
while expression:
  statement

for variable in sequence:
  statement
```

Циклы: while, for

Пример:

```
x = 0
while x < 10:
  print(x)
  x += 1

for x in range(10):
  print(x)
```

Циклы: while, for

- **break -** прерывает исполнение цикла
- **continue -** переход к следующей итерации
- **else**