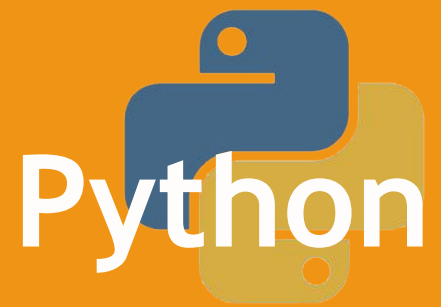
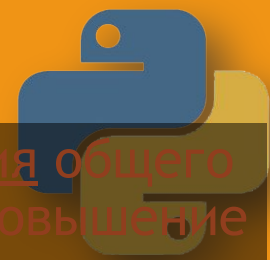


Язык
Программирования





Описание

- Высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объем полезных функций.

```
31 self.file = None
32 self.fingerprints = set()
33
34 self.logdupes = True
35 self.debug = debug
36 self.logger = logging.getLogger(__name__)
37 if path:
38     self.file = open(os.path.join(path, "requests.log"),
39                     "a")
40     self.fingerprints.update([os.path.join(path, "requests.log")])
41
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.getboolean("requests.log.debug")
45     return cls(job_dir(settings), debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + os.linesep)
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```

```
import random
s = []
e = []

for x in range(0, random.randint(0, 10)):
    s.append(random.randint(0, 1000))
    e.append(random.randint(0, 1000))

print(s, '- первый список')
print(e, '- второй список')
print(sum(s), '-', 'сумма элементов первого списка')
print(sum(e), '-', 'сумма элементов второго списка')

if sum(s) > sum(e):
    print('left - рычаг наклонился влево')
    print(sum(s) - sum(e), 'разность первого и второго списка')
elif sum(s) < sum(e):
    print('right - рычаг наклонился вправо')
    print(sum(e) - sum(s), 'разность второго и первого списка')
elif sum(s) == sum(e):
    print('balance - рычаг остается в балансе')
    print(sum(s) - sum(e), 'разность одинаковых по длине списков')
```

Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.



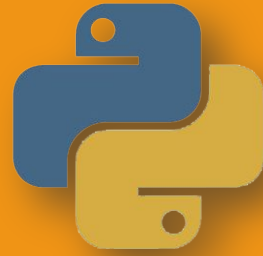
Преимущества Python



1. открытая разработка;
2. довольно прост в изучении, особенно на начальном этапе;
3. особенности синтаксиса стимулируют программиста писать хорошо читаемый код;
4. предоставляет средства быстрого прототипирования и динамической семантики;
5. имеет большое сообщество, позитивно настроенное по отношению к новичкам;
6. множество полезных библиотек и расширений языка можно легко использовать в своих проектах благодаря предельно унифицированному механизму импорта и программным интерфейсам;
7. механизмы модульности хорошо продуманы и могут быть легко использованы;
8. абсолютно всё в Python является объектами в смысле ООП, но при этом объектный подход не навязывается программисту.



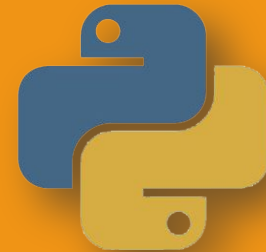
Преимущества Python



Многие разработчики считают Python универсальным вариантом для программирования, чего только душе программиста будет угодно. Он достаточно простой и читабельный, а при всей своей простоте Python позволяет реализовать практически любую идею.



Преимущества Python

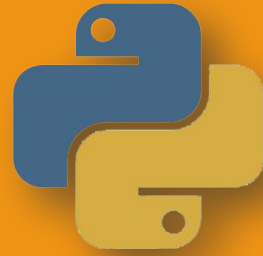


Пример кода, написанного на Python

```
1 from functools import partial, wraps
2 from inspect import getargspec
3
4 from inspect import getargspec
5
6 class F(object):
7     """Provide simple syntax for functions composition
8     (through << and >> operators) and partial function
9     application (through simple tuple syntax).
10    Usage example:
11    >>> func = F() << (_ + 10) << (_ + 5)
12    >>> print(func(10))
13    25
14    >>> func = F() >> (filter, _ < 6) >> sum
15    >>> print(func(range(10)))
16    15
17    """
18
19    __slots__ = "f",
20
21    def __init__(self, f = identity, *args, **kwargs):
22        self.f = partial(f, *args, **kwargs) if any([args, kwargs]) else f
23
24    @classmethod
25    def __compose(cls, f, g):
26        """Produces new class instance that will
27        execute given functions one by one. Internal
28        method that was added to avoid code duplication
29        in other methods.
30        """
31        return cls(lambda *args, **kwargs: f(g(*args, **kwargs)))
```



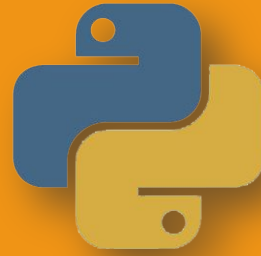
Недостатки Python



- не слишком удачная поддержка многопоточности;
- на Python создано не так уж много качественных программных проектов по сравнению с другими универсальными языками программирования, например, с Java;
- отсутствие коммерческой поддержки средств разработки;
- изначальная ограниченность средств для работы с базами данных;
- бенчмарки показывают меньшую производительность Python по сравнению с основными Java VM, что создаёт этому языку репутацию медленного.



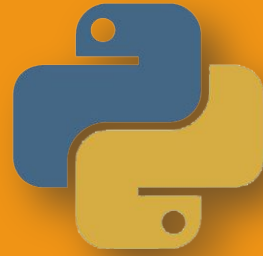
Недостатки Python



В сентябрьском рейтинге популярности языков программирования наблюдается рост популярности языка Python (7.653%), который переместился на третье место, вытеснив с него язык C++ (7.396%). Первое и второе места как и раньше удерживают языки Java (17.436%) и C (15.447%). По сравнению с сентябрём прошлого года.

		Сентябрь 2018		Сентябрь 2017	
1		1	Java	17.436%	+4.75%
2		2	C	15.447%	+8.06%
3	5	▲	Python	7.653%	+4.67%
4	3	▼	C++	7.394%	+1.83%
5	8	▲	Visual Basic .NET	5.308%	+3.33%
6	4	▼	C#	3.295%	-1.48%
7	6	▼	PHP	2.775%	+0.57%
8	7	▼	JavaScript	2.131%	+0.11%
9	-	▲	SQL	2.062%	+2.06%
10	18	▲	Objective-C	1.509%	+0.00%
10	18	▼	Objective-C	1.509%	+0.00%

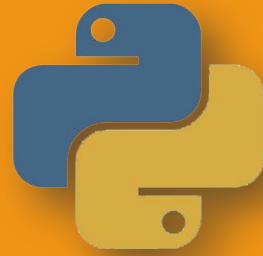
Структура данных



Python поддерживает динамическую типизацию, то есть тип переменной определяется только во время исполнения. Поэтому вместо «присваивания значения переменной» лучше говорить о «связывании значения с некоторым именем». В Python имеются встроенные типы: булевый, строка, Unicode-строка, целое число произвольной точности, число с плавающей запятой, комплексное число и некоторые другие. Из коллекций в Python встроены: список, кортеж (*неизменяемый список*), словарь, множество и другие. Все значения являются объектами, в том числе функции, методы, модули, классы.



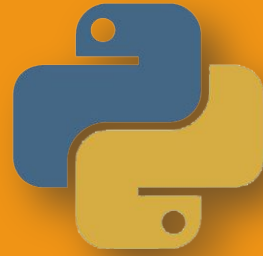
Структура данных



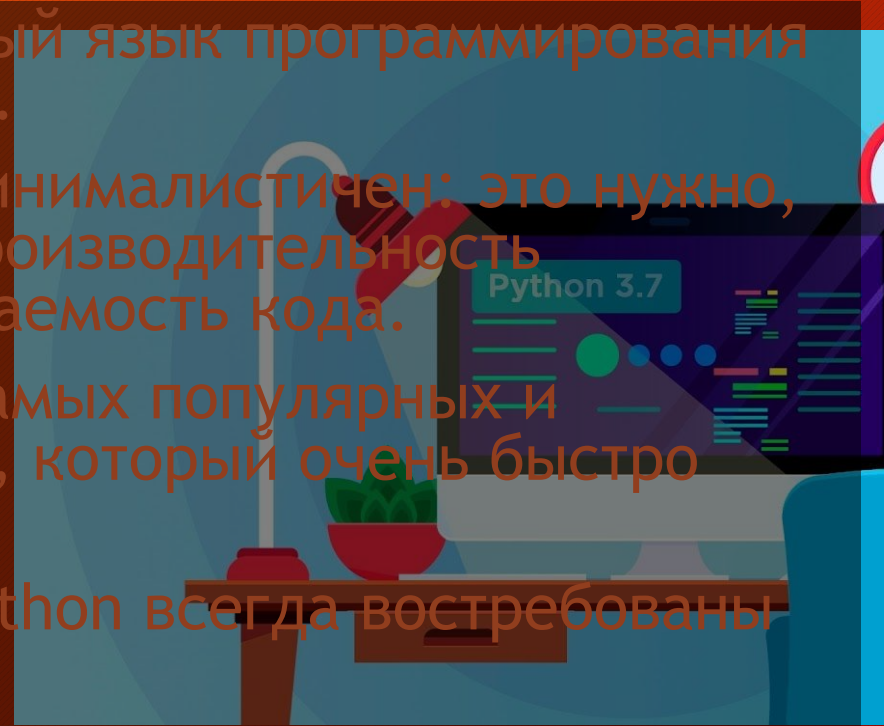
Все объекты делятся на ссылочные и атомарные. К атомарным относятся `int`, `long` (в версии 3 любое число `int`, так как в версии 3 нет ограничения на размер), `complex` и некоторые другие. При присваивании атомарных объектов копируется их значение, в то время как для ссылочных копируется только указатель на объект, таким образом, обе переменные после присваивания используют одно и то же значение. Ссылочные объекты бывают изменяемые и неизменяемые. Например, строки и кортежи являются неизменяемыми, а списки, словари и многие другие объекты — изменяемыми. Кортеж в Python является, по сути, неизменяемым списком. Во многих случаях кортежи работают быстрее списков, поэтому если не планируется изменять последовательность, то лучше использовать именно их.

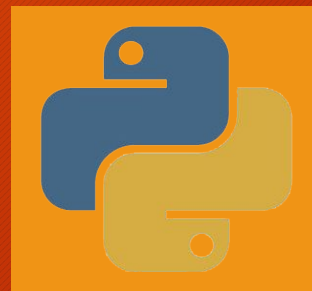


Зачем учить Python?



- Это высокоуровневый язык программирования общего назначения.
- Синтаксис языка минималистичен: это нужно, чтобы увеличить производительность разработчика и читаемость кода.
- Python – один из самых популярных и адаптивных языков, который очень быстро развивается.
- Разработчики на Python всегда востребованы на рынке.





Презентация окончена

Спасибо за внимание