


Python 入门

姚开健

2017-03-16



前言

- 1、编程的背后是计算机基础
 - 2、授人以鱼不如授人以渔
 - 3、面向非程序员
- 

如何快速掌握一门语言的 50%

- 1. 首先了解该语言的基本数据类型，基本语法和主要语言构造
- 2. 其次掌握数组和其他集合类的使用
- 3. 简单字符串处理
- 4. 基本面向对象或者函数式编程的特征
- 5. 异常、错误处理、断言、日志和调试支持，对单元测试的支持
- 6. 程序代码和可执行代码的组织机制，运行时模块加载、符号查找机制
- 7. 基本输入输出和文件处理，输入输出流类的组织
- 8. 该语言如何进行 callback 方法调用，如何支持事件驱动编程模型
- 9. 研究 regex 和 XML 处理问题
- 10. 序列化和反序列化
- 11. 了解一下线程、并发和异步调用机制
-

大纲

- Python 介绍
- Python 开发环境
- Python 程序执行
- Python 程序组织
- Python 数据类型
- Python 函数与类
- Python 学习资源

Python 介绍

➤ 动态脚本语言

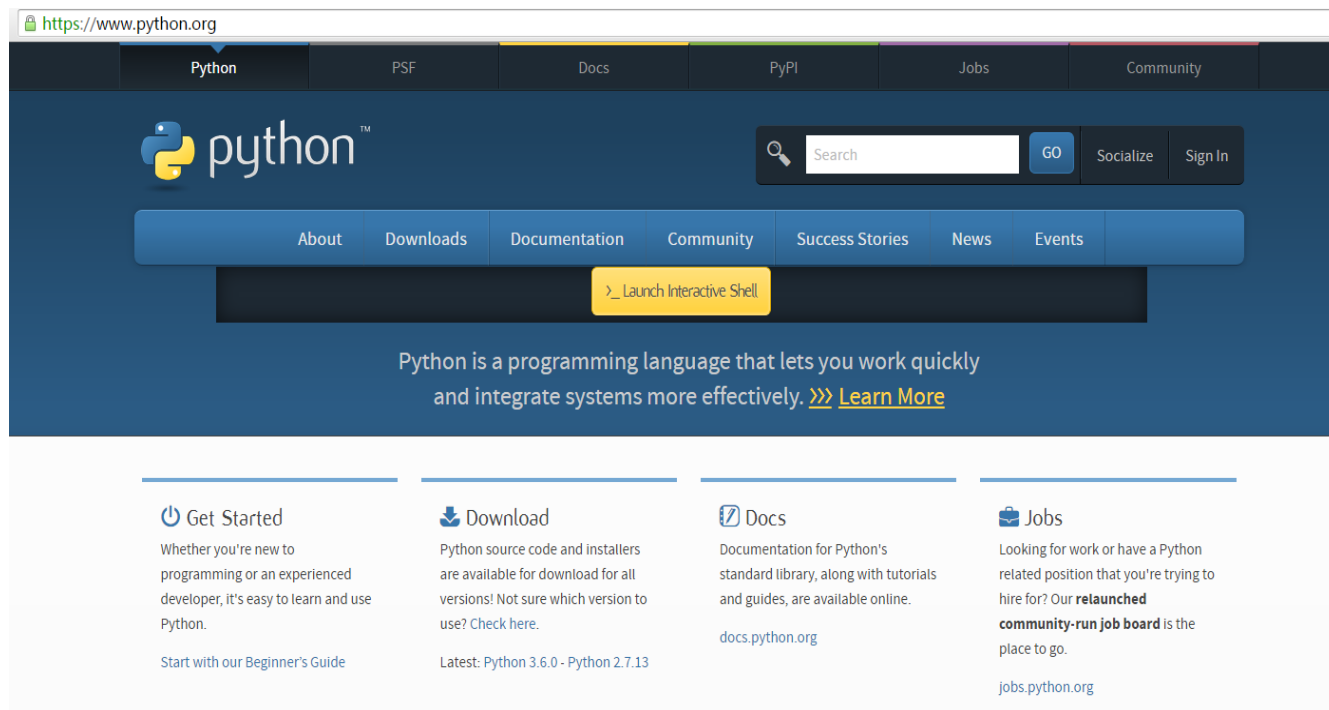
- 解释执行
- 跨平台

➤ 优点：

- 简单易学
- 开发速度快
- 丰富的库

➤ 缺点：

- 运行速度相对于编译型语言慢



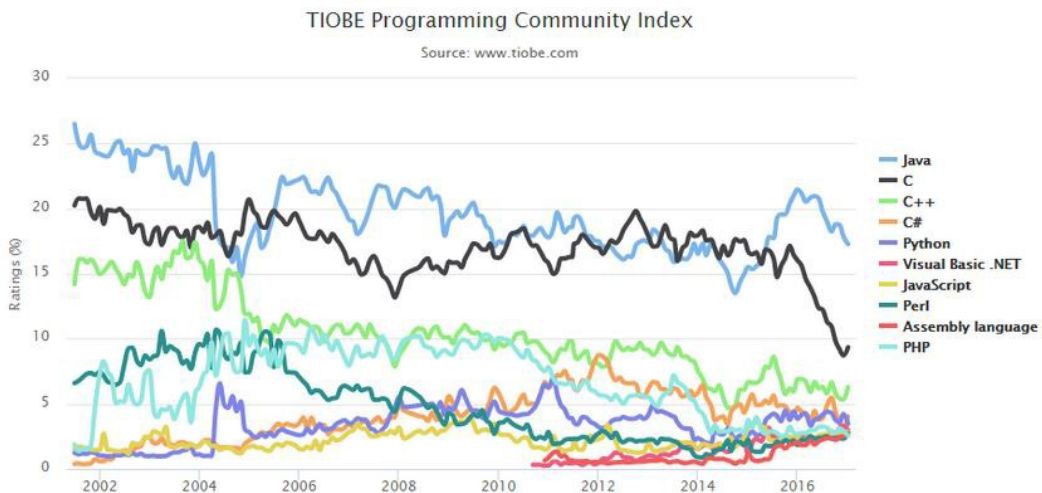
Python 介绍

➤ 应用领域

- 运维
- 测试
- 数据分析和数据挖掘
- 科研
- Web 开发
- 智能硬件

➤ 国内外使用 Python 开发的网站

- 知乎，豆瓣等
- Youtube，Dropbox 等



摘自 Tiobe 编程语言排行榜

<http://www.oschina.net/news/80812/go-is-tiobe-programming-language-of-2016>

Python 开发环境

- IDLE
- 编辑器
 - PyCharm
 - Wing
 - Eclipse+PyDev
 - Vim
 -



Python 程序执行

交互式解释器和 IDE

```
root@dl580-5:~/gary# python
Python 2.7.6 (default, Jun 22 2015, 17:58:13)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print 'Hello guys'
Hello guys
>>> 
```

命令行启动脚本

```
root@hLinux:~/gary# python update app.py 
```


Python 程序组织

1，一个 .py 文件是一个模块，文件名就是模块名

2，含有 __init__.py 的文件夹是包，多个模块可以组成一个包

3，几种 import 方式：

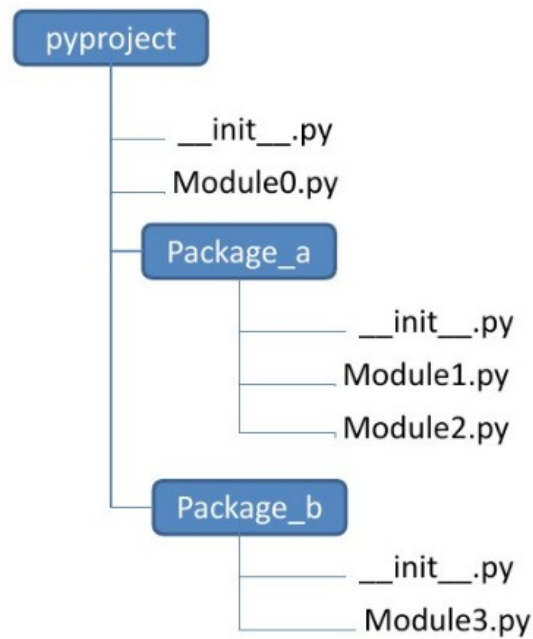
`import X`

`from X import *`

`from X import a, b, c`

`X = __import__('X')`

- 一个py程序结构



```
import Package_a.Module1
import Package_b.Module3
From Package_a.Module1 import *
```

```
import Module0
from Module0 import
```

```
Import Package_a.Module1 as m1
t = m1.Test()
....
```

```
from Package_a.Module1 import *
t = Test()
....
```

Python 程序组织

1，使用缩进表达代码逻辑

2，使用 `__name__` 系统变量确定模

块是被导入还是直接执行

3，顶级代码块在模块导入时自动执行

```
#!/usr/bin/env python
```

(1) 起始行

```
"this is a test module"
```

(2) 模块文档（文档字符串）

```
import sys  
import os
```

(3) 模块导入

```
debug = True
```

(4) (全局) 变量定义

```
class FooClass (object):  
    "Foo class"  
    pass
```

(5) 类定义（若有）

```
def test():  
    "test function"  
    foo = FooClass()  
    if debug:  
        print 'ran test()'
```

(6) 函数定义（若有）

```
if __name__ == '__main__':  
    test()
```

(7) 主程序

Python 数据类型

数字

int

boolean

float

complex number

string

list

tuple

dictionary

数据类型	存储模型	更新模型	访问模型
数字	Scalar	不可更改	直接访问
字符串	Scalar	不可更改	顺序访问
列表	Container	可更改	顺序访问
元组	Container	不可更改	顺序访问
字典	Container	可更改	映射访问

Python 数据类型

List

- 1 , 动态数组, 元素可为任意类型
- 2 , 索引访问, 切片截取
- 3 , 可变对象, 序列对象
- 4 , 拥有专有的内建函数

List Method

`list.append(obj)`

`list.count(obj)`

`list.extend(seq)*`

`list.index(obj, i=0, j=len(list))`

`list.insert(index, obj)`

`list.pop(index=-1)*`

`list.remove(obj)`

`list.reverse()`

Operation

向列表中添加一个对象 `obj`

返回一个对象 `obj` 在列表中出现的次数

把序列 `seq` 的内容添加到列表中

返回 `list[k] == obj` 的 `k` 值, 并且 `k` 的范围在 `i<=k<j`; 否则引发 `ValueError` 异常.

在索引量为 `index` 的位置插入对象 `obj`.

删除并返回指定位置的元素, 默认是最后一个元素

从列表中删除对象 `obj`

原地翻转列表

Python 数据类型

Tuple

- 1，不可变的 List
- 2，索引访问，切片截取
- 3，不可变对象，序列对象
- 4，没有专有的内建函数和操作符

```
>>> li = ('a', 'b', 'c')
>>> li
('a','b','c')
>>> li[0]
'a'
>>> li[2]
'c'
>>> li[-1]
'c'
>>> li[-2]
'b'
```

Python 函数与类

- 1，可变类型参数与不可变类型参数
- 2，必备参数，默认参数
- 3，关键字参数，不定长参数（*args, **kwargs）
- 4，高阶函数与匿名函数

单击图标添加图片

普通函数：

```
def functionname( parameters ):
    " 函数_文档字符串 "
    function_suite
    return [expression]
```

匿名函数：

```
print map(lambda x:x*x,[1,2,3,4,5,6,7])
```

高阶函数

```
def f(x):
    return x* x
print map(f,[1,2,3,4,5,6,7])
```

Python 函数与类

- 1，类数据属性和实例数据属性
- 2，类方法，实例方法和静态方法
- 3，使用下划线进行访问控制

```
>>> class MyClass:
...     """A simple example class"""
...     i = 12345
...     def f(self):
...         return 'hello world'
...
>>> x = MyClass()
>>> x.f()
'hello world'
>>> xf = x.f
>>> xf()
'hello world'
```

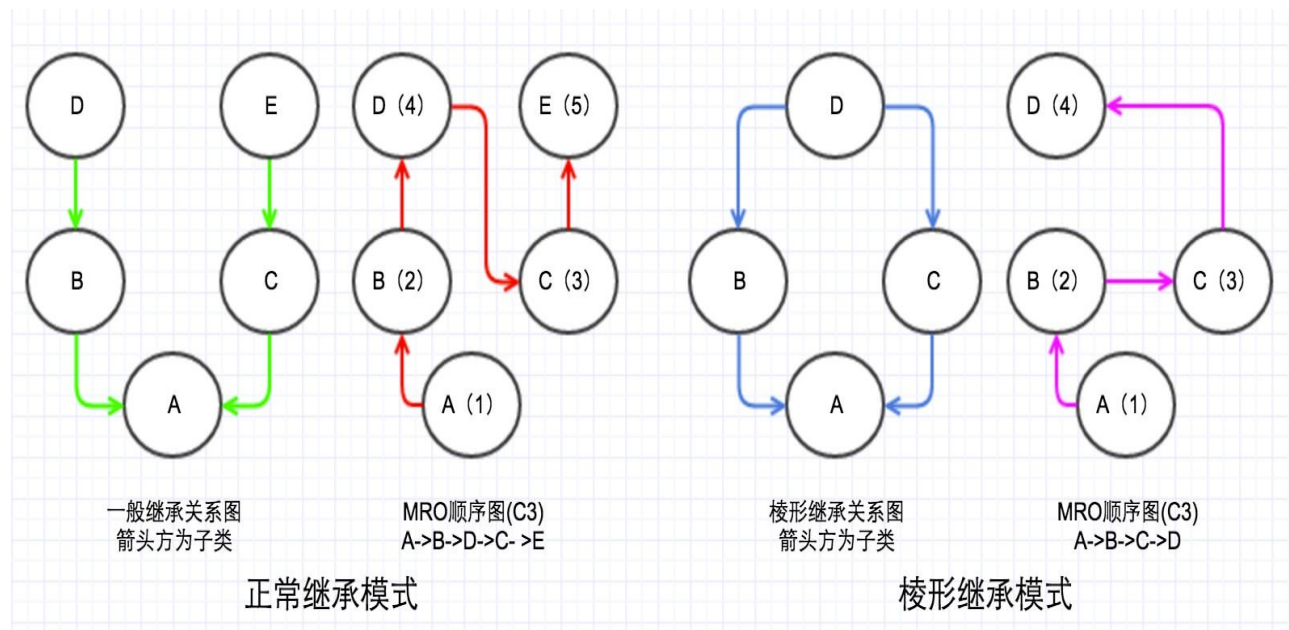
Python 函数与类

所有的类自带特殊方法
可自定义

类属性	含义
<code>__name__</code>	类的名字 (字符串)
<code>__doc__</code>	类的文档字符串
<code>__bases__</code>	类的所有父类组成的元组
<code>__dict__</code>	类的属性组成的字典
<code>__module__</code>	类所属的模块
<code>__class__</code>	类对象的类型

Python 函数与类

支持多重继承，方法顺序 MRO
深度优先



单击图标添加图片

Python 的访问控制

- 1，使用单下划线进行模块级私有化
- 2，使用双下划线进行类级私有化
- 3，LEGB-rule 作用域：Local -> Enclosing -> Global -> Built-in

LEGB 含义解释：

L-Local(function)；函数内的名字空间

E-Enclosing function locals；外部嵌套函数的名字空间（例如 closure）

G-Global(module)；函数定义所在模块（文件）的名字空间

B-Builtin(Python)；Python 内置模块的名字空间

Python 异常处理

- 1 , 默认异常处理器
- 2 , try except
- 3 , try finally
- 4 , assert False “错误提示”
- 5 , with ... as ...

单击图标添加图片

```
try except:
```

```
s = 'Hello girl!'
```

```
try:
```

```
    print s[100]
```

```
except IndexError:
```

```
    print 'error...'
```

```
print 'continue'
```

```
assert False :
```

```
assert False,'error...'
```

```
print 'continue'
```

```
with ... as ...
```

```
with open('nothing.txt','r') as f:
```

```
    f.read()
```

```
print 2/0
```

```
print 'continue'
```

单击图标添加图片

Python 学习资源

1，官方文档

<https://www.python.org/doc/>

2，标准库

3，iPython

4，书籍

5，知乎，讨论组等

6，Google，百度，stackoverflow 等

推荐书籍

入门：

《Python 核心编程》

《Python 学习手册》

进阶：

《Python 高级编程》

《Python 源码剖析》

《Python 标准库》

《Fluent Python》