help(Python)

小石头

提纲

inspect.getmembers(slides)

- Python本身
  - 什么是Python?
  - 为什么用Python?
- Python Community
- Python本地开发
- 全栈Python?
- Python 2 vs Python 3
- Pypi 库分析

什么是Python?

def(Python)

Python is a

widely used
high-level,
general-purpose,
interpreted,
dynamic

programming language.

Ref: https://en.wikipedia.org/wiki/Python_(programming_language)

Python is a

widely used

high-level,

general-purpose,

interpreted,

dynamic

programming language.

Ref: https://en.wikipedia.org/wiki/Python_(programming_language)


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION
IV
Disney · PIXAR
Instagram


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION
IV
Disney · PIXAR
Instagram


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION IV
Instagram
Disney · PIXAR


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION IV
Instagram
Disney · PIXAR

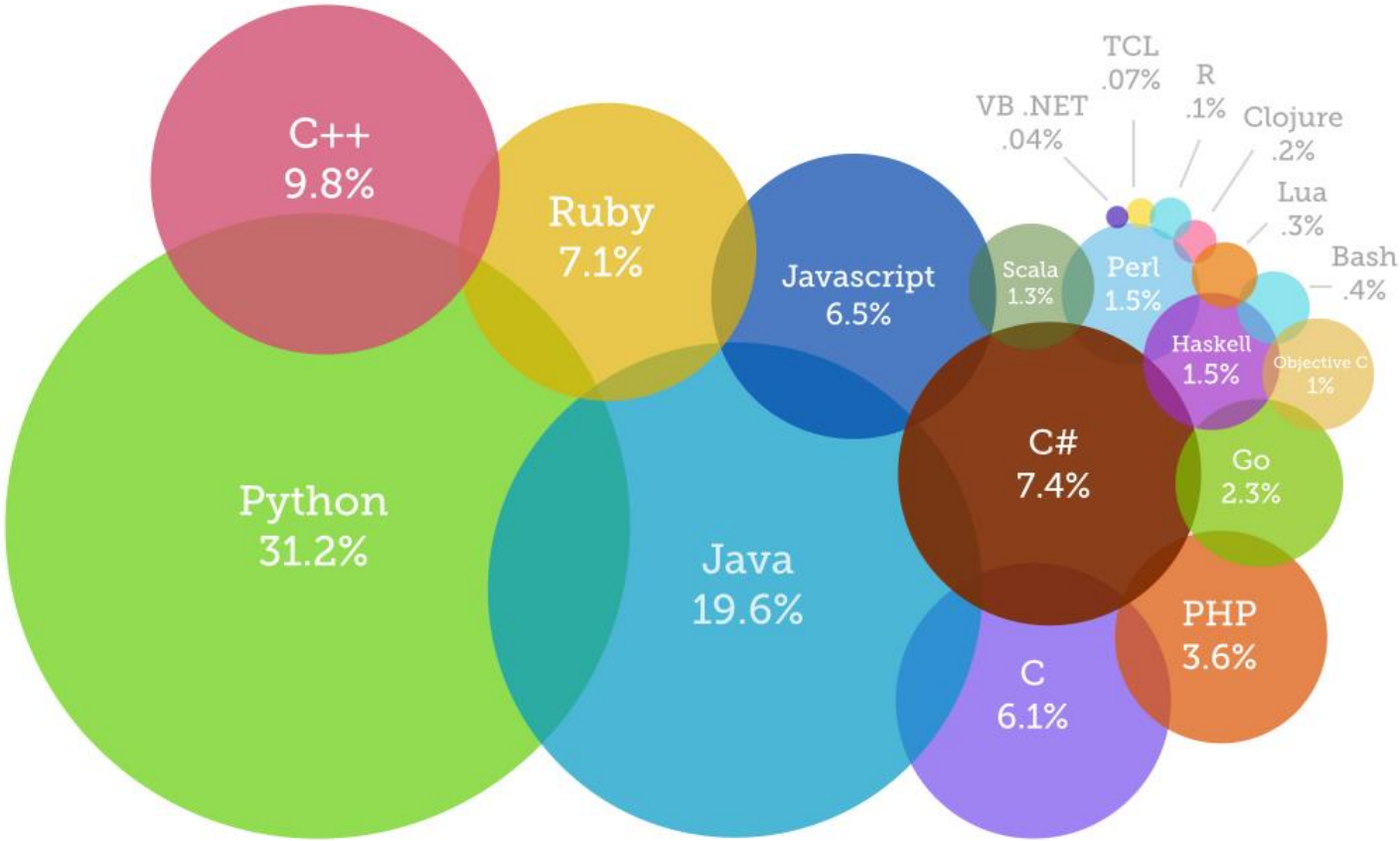

NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION IV
Disney · PIXAR
Instagram


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION IV
Disney · PIXAR
Instagram


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION
IV
Disney · PIXAR
Instagram


NumPy
SciPy
reddit
yum
yellowdog updater modified
YouTube
Dropbox
Quora
Bitbucket
BitTorrent
mercurial
Google
yelp
blender
YAHOO!
unity
EVE
ONLINE
Spotify
Sid Meier's
CIVILIZATION
IV
Disney · PIXAR
Instagram

Most Popular Coding Languages of 2015



Python is a

widely used

high-level,

general-purpose,

interpreted,

dynamic

programming language.

Ref: https://en.wikipedia.org/wiki/Python_(programming_language)

A **high-level** programming language

is a programming language with

strong abstraction

from the details of the computer.

Reference: https://en.wikipedia.org/wiki/High-level_programming_language

The essence of abstractions is preserving information that is relevant in a given context, and forgetting information that is irrelevant in that context.

– John V. Guttag

Reference: Guttag, John V. (2013-01-18). *Introduction to Computation and Programming Using Python* (Spring 2013 ed.). Cambridge, Massachusetts: The MIT Press. ISBN 9780262519632

私货警告

BB Warning

以下内容为小石头个人私货和YY, 脑袋弄坏了, 小石头不负责哈.

Not Responsible For Any Brain Damages

编程 == 抽象化 + 具象化

现实问题 → 抽象化 → 人脑中的问题

需求

人脑中的问题 → 咖啡 → 人脑中解决方法

设计

人脑中解决方法 → 具象化 → 计算机程序

实现

Python 极为接近自然语言

→ 简化了实现的过程
→ 学习曲线极低
→ 代码更加可读

更多时间解决问题, 更少时间解决bug

可读性

写码一时爽, 看码愁断肠.

Python的设计哲学就是让代码非常可读.

Python is a

widely used
high-level,
general-purpose,
interpreted,
dynamic

programming language.

Ref: https://en.wikipedia.org/wiki/Python_(programming_language)

Python的实现

误解	正解
Python是一门直译语言, 所以Python很慢	Python是一门语言, 这种语言有多种实现
Python底层是用C写的	CPython的底层是用C写的.

Class Python(object):

- CPython
  - 官方, 作为参照的Python
  - C
  - 最高兼容, 最广泛使用
  - 要用C extensions? 用CPython
- PyPy
  - 用Python写的Python
  - JIT compiler (即时编译)
  - 支持多种Backend(C, CLI, JVM)

Reference: http://docs.python-guide.org/en/latest/starting/which-python/#implementations

Class Python(object):

- JPython/Jython
  - Python代码 → Java bytecode → JVM
  - 像用Python一样用Java class
- IronPython
  - Python for .NET
  - 同时使用Python 和 .NET 库
- PythonNet

Reference: http://docs.python-guide.org/en/latest/starting/which-python/#implementations

Python is a

widely used
high-level,
general-purpose,
interpreted,

dynamic

programming language.

Ref: https://en.wikipedia.org/wiki/Python_(programming_language)

语言特征

Java

静态语言
改变代码要重新compile
Java with Reflection = dynamic

静态类型语言
```
int myCounter = 0;
String myString =
String.valueOf(myCounter);
```

强类型语言
```
myString + myCounter  // raise..
```

Python

动态语言
CI, 边跑边更新code, 支持eval (不推荐)

动态类型语言
```
counter = 0
string = str(counter)
```

强类型语言
```
string + counter  # TypeError
```

语言特征 - 其他

Python

支持指令式编程
(Interactive shell demo)

快速开发
(此处应有Flask demo)

库多
Github 第三大语言

Python 2 vs Python 3

重要: Python 3 is backwards incompatible

- Strings, Lazy Evaluation (aka Generators), etc.
- Dive Into Python 3
- The key differences between Python 2.7.x and Python 3.x with examples

我应该用Python 3 吗?
- 你有legacy code吗?
- 你用的library都有port to python 3吗?
  - Can I use Python 3
- The state of Python (2 vs 3)

Porting to Python 3
- `six`
- `2to3`

当我们在说编程语言的时候
我们在说什么？

Python Community

菜鸟警告

N00bs Warning

小石头是菜鸟, 请有批判性地观赏, 不对的地方请指正.

Read Critically, Correct When Necessary

Python Community

我见过最好的community

- Pypi
- PEPs: Simple and Well Defined Best Practices
- We are all responsible users
  - 与其安防层层防火墙, 不如建立好的convention
  - Python不能阻止你写出坏代码, 但用Python写出坏代码比较难.

Python程序员活的更开心 :-)

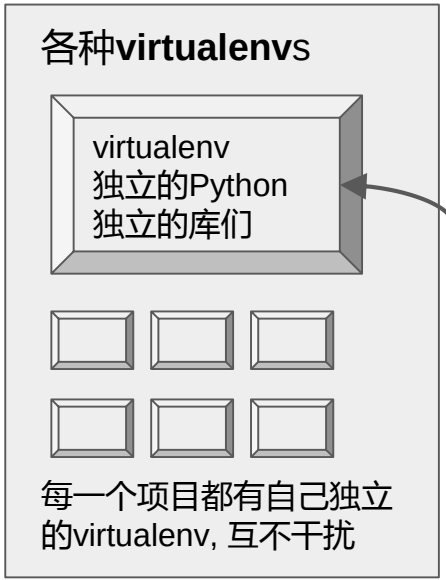
如何使用Python?

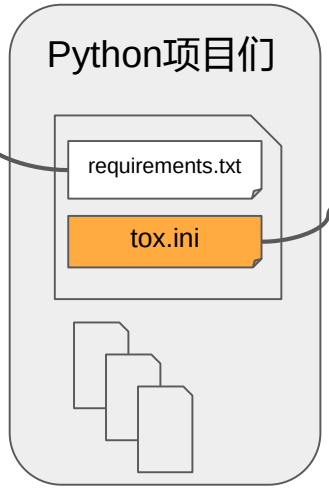
小石头的本地开发

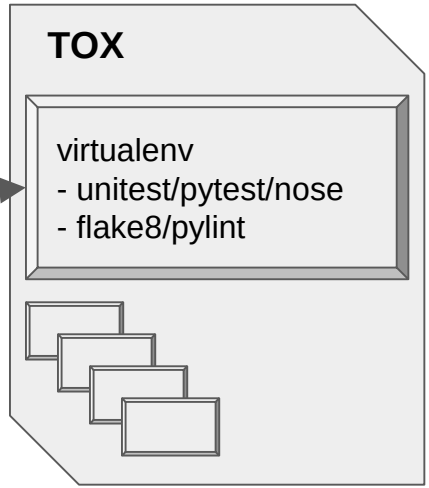
use virtualenv!

virtualenvwrapper
virtualenvs的管理工具

```
$ cd ~/workspace
$ mkvirtualenv requests
(requests)$ git clone https://github.com/kennethreitz/requests
(requests)$ pip install -r requests/requirements.txt
(requests)$ tox -e py34
```





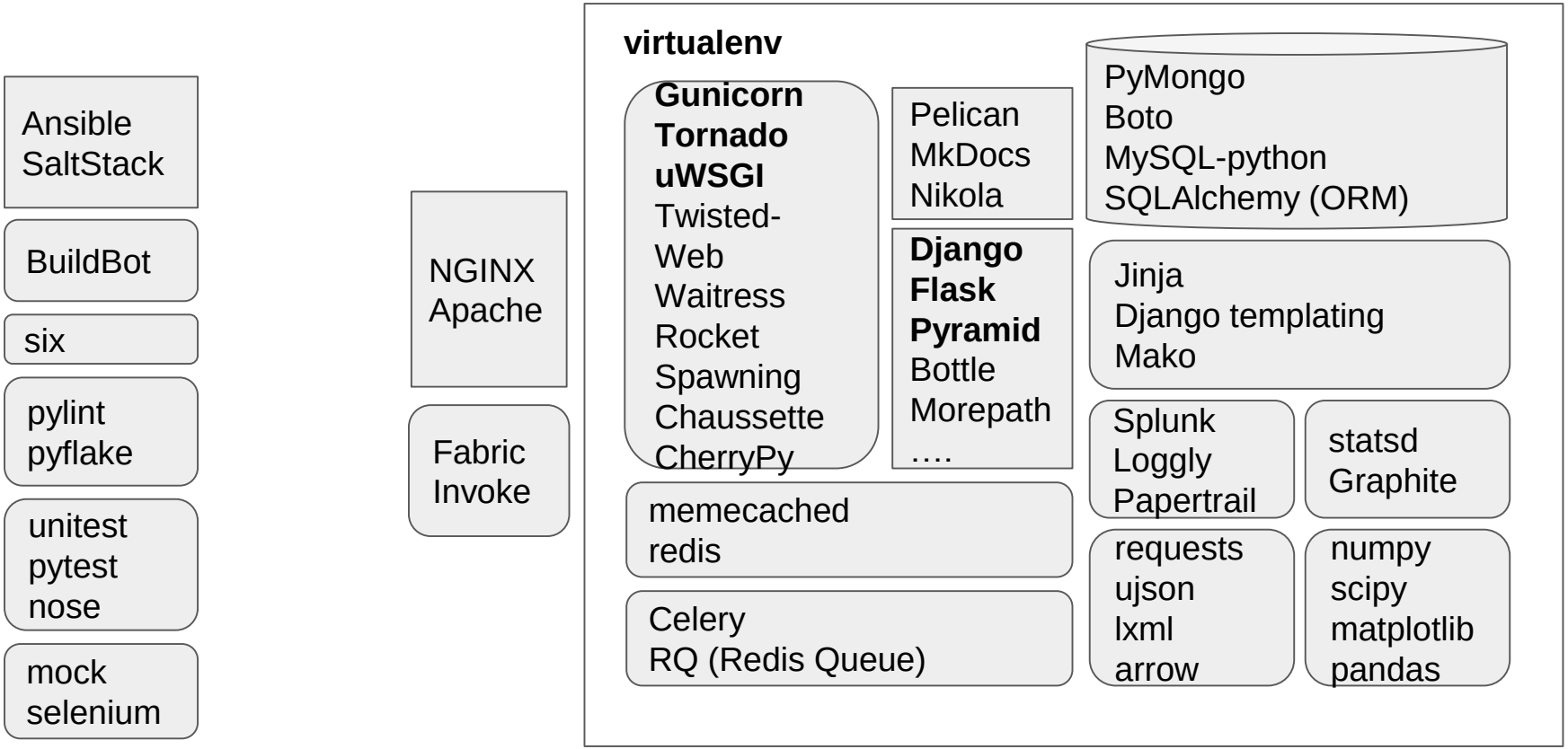


其他工具和库

- **awscli**
- **click**
  (better argparse)
- **mock** (py3 自带)
- Qt (非python)

全栈Python?

There is ~~an app~~ library for it



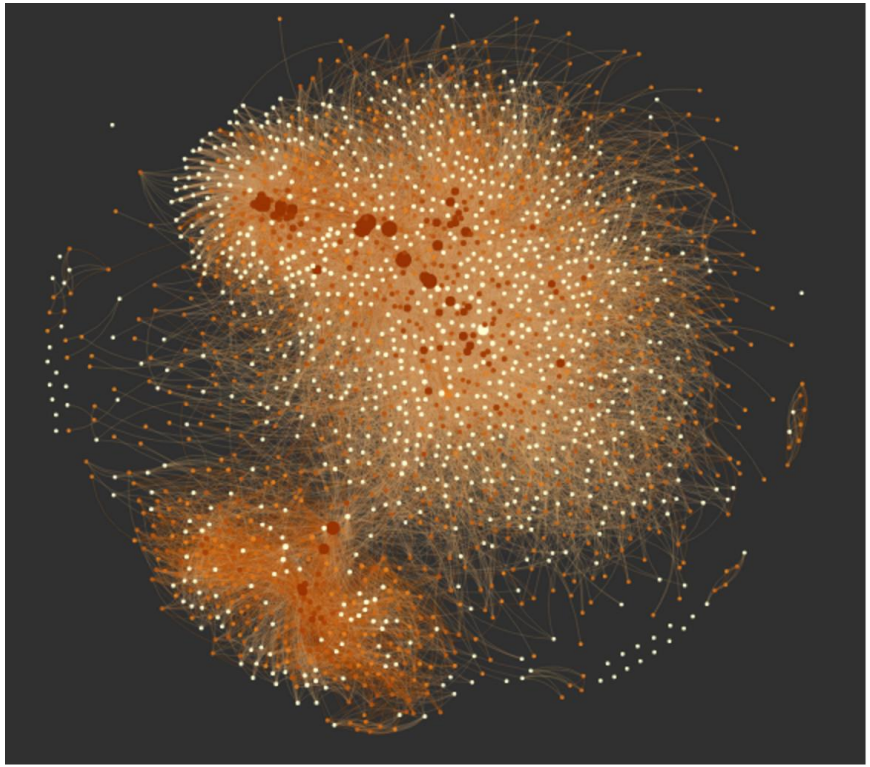
PyPI

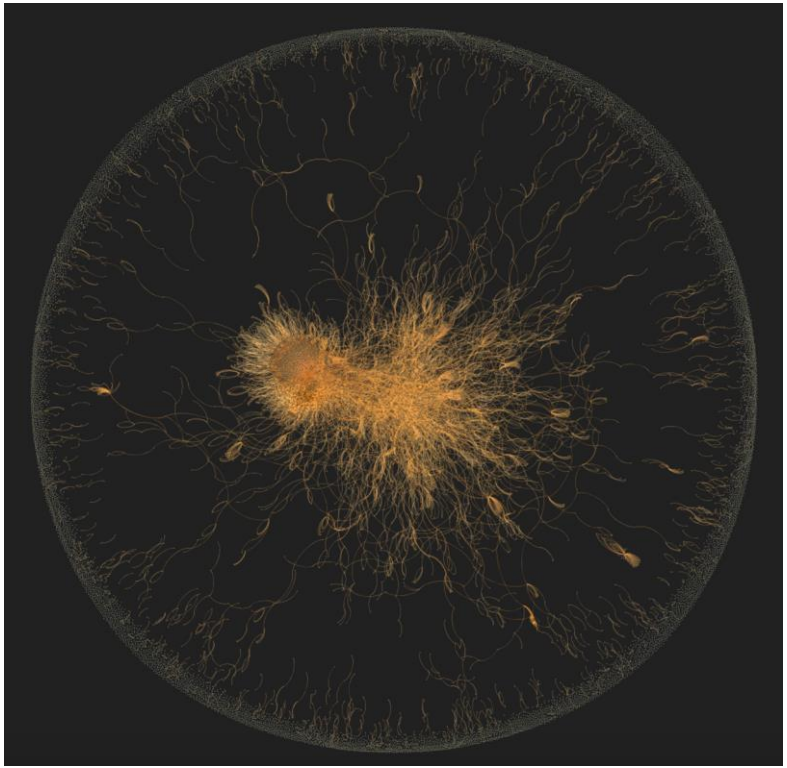
pip install -r requirements.txt

- the Python Package Index
- 开源, Python默认配置, 79k+
- 简单无痛, requirements 文件哭瞎Java程序员
    - 个人在mac为主windows为辅上的体验. linux (debian) 上的体验比较糟糕, 但也可能是因为我是

```
pip install requests
pip install -r requirements.txt
pip install git+git://github.com/myuser/foo.git@v123
```

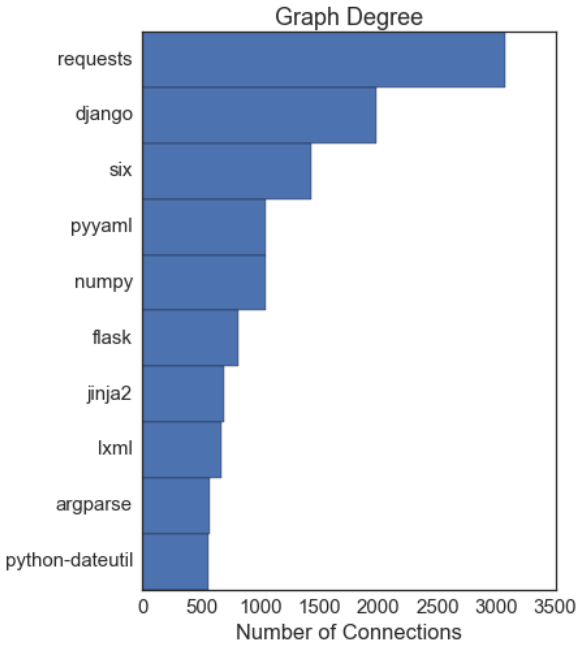
PyPI Dependency Analysis

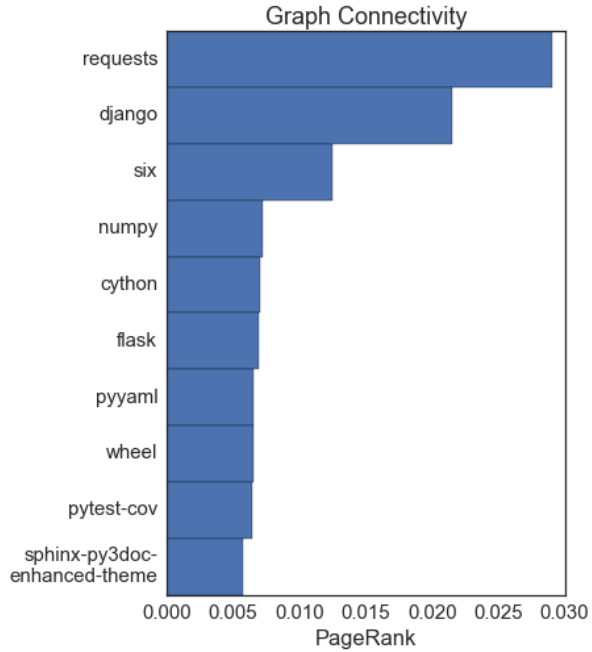


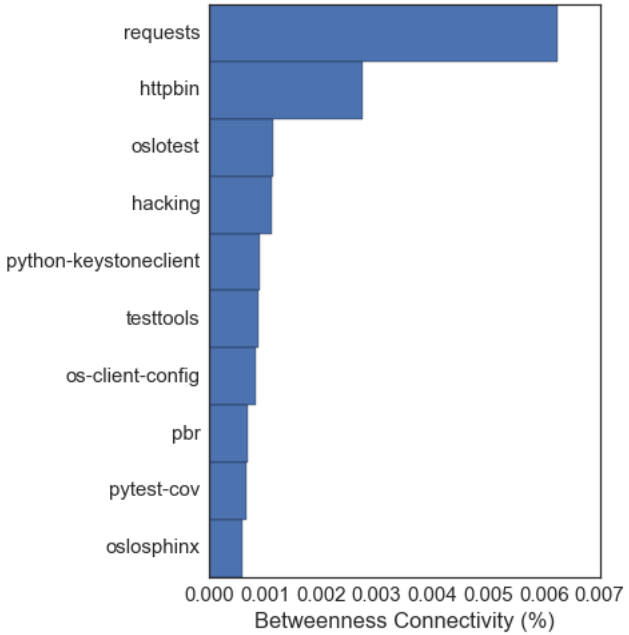


Source: http://kgullikson88.github.io/blog/pypi-analysis.html, https://ogirardot.wordpress.com/2013/01/05/state-of-the-pythonpypi-dependency-graph/

Package Importance

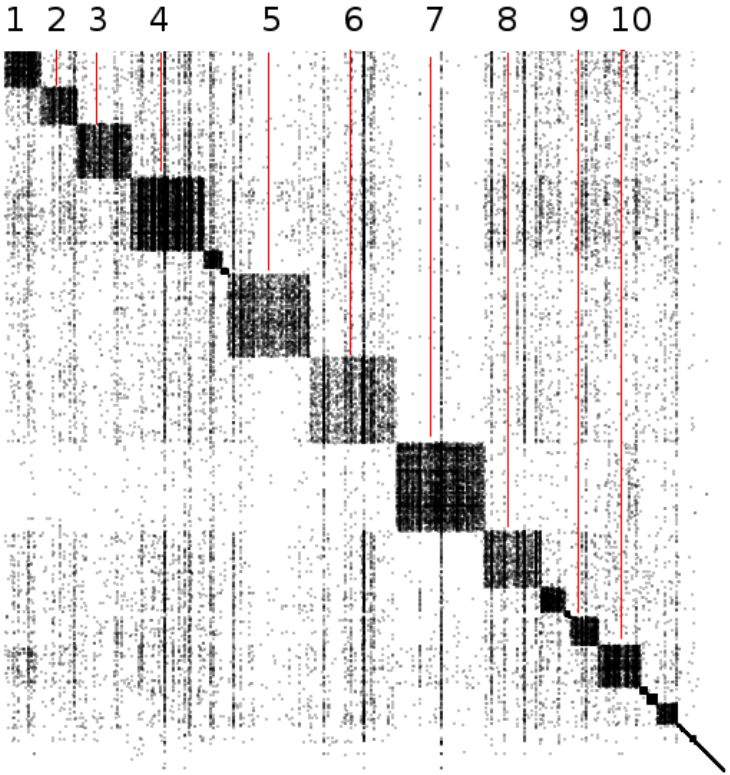






Source: http://kgullikson88.github.io/blog/pypi-analysis.html,

Development Communities



Source: http://kgullikson88.github.io/blog/pypi-analysis.html,

1. `flask` & `bottle` packages: web server frameworks
2. `redis`, `tornado`, & `pyzmq`.
3. the pydata community - `numpy`, `scipy`, `matplotlib`, & `pandas`.
4. Testing & documentation packages.
5. `django` packages
6. `requests` + other tools for web interface/scraping.
7. the `zope` community
8. Static website development and configuration: `pyyamml` & `jinja2` packages.
9. Small & useful utilities: `argparse`, `decorator`, `pyparsing`..
10. Large variety of packages - `sqlalchemy` etc.

怎样用好Python?

什么是Pythonic?

```
# 这还差不多
猫 = [名字 for 名字 in 动物们 if '猫' in 名字]
```

Pythonic == Idiomatic

- Python社区 认为 “好代码” 很重要
- 什么是好代码? Best Practices有哪些? 我们应该怎么写代码?
- Python Enhancement Proposals (PEP)
  - PEP 8 -- Style Guide for Python Code

```
动物们 = ['大花猫', '大花狗', '大花生']
```

```
# 不pythonic
猫 = []
for 索引 in range(len(动物们)):
        if 动物们[索引].find('猫') != -1:
                猫.append(动物们[索引])
```

Google Python Style Guide

读好的代码

Repo

- HowDoI
- Flask
- Diamond
- Werkzeug
- Requests
- TabLib

程序员

- Guido van Rossum
- Tim Peters
- Raymond hettinger
- David Beazley
- Armin Ronacher
- Glyph Lefkowitz(Twisted)
- Benoit Chesneau(Gunicorn)

摘自: http://docs.python-guide.org/en/latest/writing/reading/

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
 Namespaces are one honking great idea -- let's do more of those!
```

感谢Python

Thanks to Python

(此处应有鸡汤)

There should be chicken soup

小伙伴们的采访:
你喜欢和不喜欢Python在哪里?

喜欢

- 开发效率高, 容易prototype 5/5
- 库多 4/5
- interactive shell 特别方便 3/5
- 返回值好, 对函数式编程友好 1/5
- Spark上面可以用 1/5
- 各种数据类型的转换都支持 1/5

不喜欢

- 没有类型核查, 影响程序正确率 3/5
- 速度不够快 2/5
- 没有, 很喜欢python 2/5
- 基于纯缩进, 长函数很难维护 1/5
- Perl对regex的支持更好 1/5

小伙伴们的采访:
在什么情况下是你绝对不会用Python的, 或者反之?

会用Python

- 测试自己的想法, Prototyping 4/5

不会用Python

- 强调运行效率的时候 4/5

小伙伴们的采访

使用的IDE?

- Vim 3/5
- PyCharm 2/5

使用Python的场景?

- Web App 2/5
- 大数据 1/5
- 后端异步 1/5
- 测试框架 1/5

小伙伴们的采访 - 数据科学家们

使用的库们?

Pandas, scikit-learn, keras, theano…

“画图的, matplotlib万能屌丝包, seaborn一般增强包, bokeh高端逼格包, plotly联网装逼包..”

Spark上面可以用

Pandas太牛逼

还有其他的吗?

新手选PyCharm做IDE吧

Windows的小朋友可以直接搜winpython的库, 一堆

stack overflow真好啊(拍大腿)

References

- 敬爱的Wikipedia
  - Python, high-level programming language, abstraction, dynamic programming language
  - Type System, interpreted language, JIT
- Full Stack Python
- The Hitchhiker’s Guide to Python!
- Python Dependency Analysis
- Useful links: please check slides =)

问题们?

input('Questions?')