

- 目录
- Python教程

Python简介

安装Python

第一个Python程序

Python基础

函数

高级特性

函数式编程

模块

面向对象编程

面向对象高级编程

错误、调试和测试

IO编程

进程和线程

正则表达式

常用内建模块

常用第三方模块

Pillow

requests

chardet

psutil

virtualenv

图形界面

网络编程

电子邮件

访问数据库

Web开发

异步IO

实战

FAQ

期末总结

关于作者

 廖雪峰 北京 朝阳区

➕ 加关注

廖雪峰

自己的Python课程

Python商业爬虫全解码

让天下没有爬不到的数据！

Python爬虫

+

数据分析

Python机器学习

+

深度学习

.....

找廖雪峰老师

廖雪峰老师

自己的Java课程

Java高级架构师

更专业

更权威

源码分析专题

+

微服务架构专题

高并发分布式专题

+

性能优化专题

.....

找廖雪峰老师

腾讯云

学生服务器体验套餐

10元/月

1核2G · 1M带宽 · 50GB流量

校园专享服务器套餐10元/月

 腾讯云

腾讯云服务器高性能计算能力,更有学生优惠套餐云服务器/域名/存储等服务.

➔

1 | python免费公开课

授课模式：在线直播+课后视频，从零基础到中级开发工程师

编程学习网

2 | 视频通话SDK

声网Agora.io，API接口，4行代码接入。每月1万分钟免费。

www.agora.io

psutil

阅读 65082

用Python来编写脚本简化日常的运维工作是Python的一个重要用途。在Linux下，有许多系统命令可以让我们时刻监控系统运行的状态，如 `ps`，`top`，`free` 等等。要获取这些系统信息，Python可以通过 `subprocess` 模块调用并获取结果。但这样做显得很麻烦，尤其是要写很多解析代码。

在Python中获取系统信息的另一个好办法是使用 `psutil` 这个第三方模块。顾名思义，`psutil` = process and system utilities，它不仅可以通过一两行代码实现系统监控，还可以跨平台使用，支持Linux / UNIX / OSX / Windows等，是系统管理员和运维小伙伴不可或缺的必备模块。

安装psutil

如果安装了Anaconda，`psutil`就已经可用了。否则，需要在命令行下通过pip安装：

```
$ pip install psutil
```

如果遇到Permission denied安装失败，请加上sudo重试。

获取CPU信息

我们先来获取CPU的信息：

```
>>> import psutil
>>> psutil.cpu_count() # CPU逻辑数量
4
>>> psutil.cpu_count(logical=False) # CPU物理核心
2
# 2说明是双核超线程。4则是4核非超线程
```

统计CPU的用户 / 系统 / 空闲时间：

```
>>> psutil.cpu_times()
scputimes(user=10963.31, nice=0.0, system=5138.67, idle=366102.45)
```

再实现类似 `top` 命令的CPU使用率，每秒刷新一次，累计10次：

```
>>> for x in range(10):
...     psutil.cpu_percent(interval=1, percpu=True)
...
[[4.0, 4.0, 4.0, 4.0]
 [2.0, 3.0, 4.0, 3.0]
 [8.0, 4.0, 3.0, 4.0]
 [12.0, 3.0, 3.0, 3.0]
 [18.8, 5.1, 5.9, 5.0]
 [10.9, 5.0, 4.0, 3.0]
 [12.0, 5.0, 4.0, 5.0]
 [15.0, 5.0, 4.0, 4.0]
 [19.0, 5.0, 5.0, 4.0]
 [9.0, 3.0, 2.0, 3.0]]
```

获取内存信息

使用`psutil`获取物理内存和交换内存信息，分别使用：

```
>>> psutil.virtual_memory()
system(total=8589934592, available=286520064, percent=66.6, used=7201386496, free=216178688, active=3342192640, inactive=2850341376, wired=1208652480)
>>> psutil.swap_memory()
swapp(total=1073741824, used=150732800, free=923009024, percent=14.0, sin=10705981440, sout=40353792)
```

返回的是字节为单位的整数，可以看到，总内存大小是8589934592 = 8 GB，已用7201386496 = 6.7 GB，使用了66.6%。

而交换区大小是1073741824 = 1 GB。

获取磁盘信息

可以通过`psutil`获取磁盘分区、磁盘使用率和磁盘IO信息：

```
>>> psutil.disk_partitions() # 磁盘分区信息
[('sda1', device='\\dev\\sda1', mountpoint='/', fstype='hfs', opts='rw,local,rootfs,dovfs,journaled,multilabel')]
>>> psutil.disk_usage('/') # 磁盘使用情况
diskusage(total=989982549504, used=390880133120, free=607840272384, percent=39.1)
>>> psutil.disk_io_counters() # 磁盘IO
diskio(read_count=988513, write_count=274457, read_bytes=14856830464, write_bytes=17509420032, read_time=2228966, write_time=1618405)
```

可以看到，磁盘 `'/'` 的总容量是989982549504 = 930 GB，使用了39.1%。文件格式是HFS，`opts` 中包含 `rw` 表示可读写，`journalled` 表示支持日志。

获取网络信息

`psutil`可以获取网络接口和网络连接信息：

```
>>> psutil.net_io_counters() # 获取网络读写字节/包的个数
netio(bytes_sent=3885744870, bytes_recv=10357676702, packets_sent=10613069, packets_recv=10423357, errin=0, errout=0, dropin=0, dropout=0)
>>> psutil.net_if_addrs() # 获取网络接口信息
{
  'lo0': [unic(family=AddressFamily.AF_INET, 2), address='127.0.0.1', netmask='255.0.0.0', ...],
  'en1': [unic(family=AddressFamily.AF_INET, 2), address='10.0.1.80', netmask='255.255.255.0', ...],
  'en0': [...],
  'bridge0': [...]
}
>>> psutil.net_if_stats() # 获取网络接口状态
{
  'lo0': snicstats(isup=True, duplex=Onduplex.NIC_DUPLEX_UNKNOWN, 0), speed=0, mtu=16384),
  'en0': snicstats(isup=True, duplex=Onduplex.NIC_DUPLEX_UNKNOWN, 0), speed=0, mtu=1500),
  'en1': snicstats(...),
  'en2': snicstats(...),
  'bridge0': snicstats(...)
}
```

要获取当前网络连接信息，使用 `net_connections()`：

```
>>> psutil.net_connections()
Traceback (most recent call last):
PermissionError: [Errno 1] Operation not permitted
During handling of the above exception, another exception occurred:
Traceback (most recent call last):
...
psutil.AccessDenied: psutil.AccessDenied (pid=3847)
```

你可能会得到一个`AccessDenied` 错误，原因是`psutil`获取信息也是要走系统接口，而获取网络连接信息需要root权限，这种情况下，可以退出Python交互环境，用 `sudo` 重新启动：

```
$ sudo python3
Password: *****
Python 3.6.3 ... on darwin
Type "help", ... for more information.
>>> import psutil
>>> psutil.net_connections()
[
  scconn(fd=83, family=AddressFamily.AF_INET6, 30), type=1, laddr=addr(ip=':::127.0.0.1', port=62911), raddr=addr(ip=':::127.0.0.1', port=3306), status='ESTABLISHED', pid=3725),
  scconn(fd=84, family=AddressFamily.AF_INET6, 30), type=1, laddr=addr(ip=':::127.0.0.1', port=62905), raddr=addr(ip=':::127.0.0.1', port=3306), status='ESTABLISHED', pid=3725),
  scconn(fd=89, family=AddressFamily.AF_INET6, 30), type=1, laddr=addr(ip=':::', port=6080), raddr=0, status='LISTEN', pid=3725),
  scconn(fd=103, family=AddressFamily.AF_INET6, 30), type=1, laddr=addr(ip=':::127.0.0.1', port=62918), raddr=addr(ip=':::127.0.0.1', port=3306), status='ESTABLISHED', pid=3725),
  scconn(fd=105, family=AddressFamily.AF_INET6, 30), type=1, ..., pid=3725),
  scconn(fd=106, family=AddressFamily.AF_INET6, 30), type=1, ..., pid=3725),
  scconn(fd=107, family=AddressFamily.AF_INET6, 30), type=1, ..., pid=3725),
  ...
  scconn(fd=27, family=AddressFamily.AF_INET, 2), type=2, ..., pid=1)
]
```

获取进程信息

通过`psutil`可以获取到所有进程的详细信息：

```
>>> psutil.pids() # 所有进程ID
[3865, 3864, 3863, 3866, 3855, 3853, 3776, ..., 45, 44, 1, 0]
>>> p = psutil.Process(3776) # 获取指定进程ID=3776，其实就是当前Python交互环境
>>> p.name() # 进程名称
'python3.6'
>>> p.exe() # 进程可执行文件
'/Users/michael/anaconda3/bin/python3.6'
>>> p.cwd() # 进程工作目录
'/Users/michael'
>>> p.cmdline() # 进程启动的命令行
['python3']
>>> p.ppid() # 父进程ID
3765
>>> p.parent() # 父进程
<psutil.Process(pid=3765, name='bash') at 4503144040>
>>> p.children() # 子进程列表
[]
>>> p.status() # 进程状态
'running'
>>> p.username() # 进程用户名
'michael'
>>> p.create_time() # 进程创建时间
1511062731.120333
>>> p.terminal() # 进程终端
'/dev/tty002'
>>> p.cpu_times() # 进程使用的CPU时间
pcputimes(user=0.081150144, system=0.053269812, children_user=0.0, children_system=0.0)
>>> p.memory_info() # 进程使用的内存
pmem(rss=8310784, vms=2481725440, pfaunts=3207, pages=18)
>>> p.open_files() # 进程打开的文件
[]
>>> p.connections() # 进程相关网络连接
[]
>>> p.num_threads() # 进程的线程数量
1
>>> p.threads() # 所有线程信息
[thread(id=1, user_time=0.090318, system_time=0.062736)]
>>> p.environ() # 进程环境变量
{'SHELL': '/bin/bash', 'PATH': '/usr/local/bin:/usr/bin:/usr/sbin:/sbin/...', 'PWD': '/Users/michael', 'LANG': 'zh_CN.UTF-8', ...}
>>> p.terminate() # 结束进程
Terminated: 15 <- 自己把自己结束了
```

和获取网络连接类似，获取一个root用户的进程需要root权限，启动Python交互环境或新 `.py` 文件时，需要 `sudo` 权限。

`psutil`还提供了一个 `test()` 函数，可以模拟出 `ps` 命令的效果：

```
$ sudo python3
Password: *****
Python 3.6.3 ... on darwin
Type "help", ... for more information.
>>> import psutil
>>> psutil.test()
USER      PID  MMEN  VSZ    RSS  TTY      START  TIME  COMMAND
root        0  24.0  74270628 2016380 ?        Nov18   40:51  kernel_task
root         1   0.1  2494140   9484 ?        Nov18   01:39  launchd
root        44   0.4  2519872   36404 ?        Nov18   02:02  UserEventAgent
root        45    0.5  2474032   1516 ?        Nov18   00:14  syslogd
root        47   0.1  2504768   8912 ?        Nov18   00:03  kextd
root        48   0.1  2503544   4720 ?        Nov18   00:19  fsaverd
appleveeh  52   0.1  2469748   5024 ?        Nov18   00:00  appleveevad
root        53   0.1  2500592   6132 ?        Nov18   00:02  configd
...
```

小结

`psutil`使得Python程序获取系统信息变得易如反掌。

`psutil`还可以获取用户信息、Windows服务等很多有用的系统信息，具体请参考`psutil`的官网：<https://github.com/giampaolo/psutil>

读后有收获可以请作者喝咖啡，读后有疑问请加群讨论：



还可以分享给朋友：

分享到微博

廖雪峰

官方 独家

Python

商业爬虫全解码

找廖雪峰老师

ACM金牌得主

全球顶尖名企一线数据科学家倾力指导

人工智能与自然语言/计算机视觉课程培训

Artificial Intelligence For NLP/CV Courses

无offer退全款

廖雪峰推荐

JAVA进阶教程

原价1699元

0元领取

阿里云

高性能云服务器首台5折

企业级上云最早动值，最大20Gbps内网带宽，450万PPS

立即购买

5折

E-LEARNING

Python全栈实战课程限时免费领取

120天腾讯课堂老师带你从零基础到项目实战，全系统学习爬虫、数据分析、Web开发等Python开发技术

python大型免费高级进阶公开课

课程模式：在线直播+课后视频，从零基础到中级开发工程师

评论


 upbeat_pesch created at May 18, 2019 11:34 AM, Last updated at May 18, 2019 10:03 PM

```
print(psutil.pids())#获取所有进程IDp = psutil.Process(1000)print(p.name)print(p.exe)
```

上述代码，p.name和p.exe教程中都有（），但是我如果加了（）就会报错，不加还能跑出来下面的内容

<bound method Process.name of psutil.Process(pid=1000, name='lbmpmsvc.exe', started='10:16:59')>

<bound method Process.exe of psutil.Process(pid=1000, name='lbmpmsvc.exe', started='10:16:59')>

 upbeat_pesch

Created at May 18, 2019 12:08 PM