Python标准库06 子进程 (subprocess包)

谢谢Tolbkni Kao帮我纠正错误

这里的内容以Linux进程基础和Linux文本流为基础。subprocess包主要功能是执行外部的命令和程序。比如说，我需要使用wget下载文件。我在Python中调用wget程序。从这个意义上来说，subprocess的功能与shell类似。

subprocess以及常用的封装函数

当我们运行python的时候，我们都是在创建并运行一个进程。正如我们在Linux进程基础中介绍的那样，一个进程可以fork一个子进程，并让这个子进程exec另外一个程序。在Python中，我们通过标准库中的subprocess包来fork一个子进程，并运行一个外部的程序(fork，exec见Linux进程基础)。

subprocess包中定义有数个创建子进程的函数，这些函数分别以不同的方式创建子进程，所以我们可以根据需要来从中选取一个使用。另外subprocess还提供了一些管理标准流(standard stream)和管道(pipe)的工具，从而在进程间使用文本通信。

使用subprocess包中的函数创建子进程的时候，要注意：

1） 在创建子进程之后，父进程是否暂停，并等待子进程运行。

2） 函数返回什么

3） 当returncode不为0时，父进程如何处理。

subprocess.call()

父进程等待子进程完成

返回退出信息(returncode，相当于exit code，见Linux进程基础)

subprocess.check_call()

父进程等待子进程完成

返回0

检查退出信息，如果returncode不为0，则举出错误subprocess.CalledProcessError，该对象包含有returncode属性，可用try...except...来检查(见Python错误处理)。

subprocess.check_output()

父进程等待子进程完成

返回子进程向标准输出的输出结果

检查退出信息，如果returncode不为0，则举出错误subprocess.CalledProcessError，该对象包含有returncode属性和output属性，output属性为标准输出的输出结果，可用try...except...来检查。

这三个函数的使用方法相类似，我们以subprocess.call()来说明：

```
import subprocess
rc = subprocess.call(["ls","-l"])
```

我们将程序名(ls)和所带的参数(-l)一起放在一个表中传递给subprocess.call()

可以通过一个shell来解释一整个字符串：

```
import subprocess
out = subprocess.call("ls -l", shell=True)
out = subprocess.call("cd ..", shell=True)
```

我们使用了shell=True这个参数。这个时候，我们使用一整个字符串，而不是一个表来运行子进程。Python将先运行一个shell，再用这个shell来解释这整个字符串。

shell命令中有一些是shell的内建命令，这些命令必须通过shell运行，$cd。shell=True允许我们运行这样一些命令。

Popen()

实际上，我们上面的三个函数都是基于Popen()的封装(wrapper)。这些封装的目的在于让我们容易使用子进程。当我们想要更个性化我们的需求的时候，就要转向Popen类，该类生成的对象用来代表子进程。

与上面的封装不同，Popen对象创建后，主程序不会自动等待子进程完成。我们必须调用对象的wait()方法，父进程才会等待 (也就是阻塞block)：

```
import subprocess
child = subprocess.Popen(["ping","-
c","5","www.google.com"])
print("parent process")
```

从运行结果中看到，父进程在开启子进程之后并没有等待child的完成，而是直接运行print。

对比等待的情况：

```
import subprocess
child = subprocess.Popen(["ping","-
c","5","www.google.com"])
child.wait()
print("parent process")
```

此外，你还可以在父进程中对子进程进行其它操作，比如我们上面例子中的child对象：

```
child.poll()            # 检查子进程状态
child.kill()            # 终止子进程
child.send_signal()     # 向子进程发送信号
child.terminate()       # 终止子进程
```

子进程的PID存储在child.pid

子进程的文本流控制

(沿用child子进程) 子进程的标准输入，标准输出和标准错误也可以通过如下属性表示：

```
child.stdin
child.stdout
child.stderr
```

我们可以在Popen()建立子进程的时候改变标准输入、标准输出和标准错误，并可以利用subprocess.PIPE将多个子进程的输入和输出连接在一起，构成管道(pipe)：

```
import subprocess
child1 = subprocess.Popen(["ls","-l"],
stdout=subprocess.PIPE)
child2 = subprocess.Popen(["wc"],
stdin=child1.stdout,stdout=subprocess.PIPE)
out = child2.communicate()
print(out)
```

subprocess.PIPE实际上为文本流提供一个缓存区。child1的stdout将文本输出到缓存区，随后child2的stdin从该PIPE中将文本读取走。child2的输出文本也被存放在PIPE中，直到communicate()方法从PIPE中读取出PIPE中的文本。

要注意的是，communicate()是Popen对象的一个方法，该方法会阻塞父进程，直到子进程完成。

我们还可以利用communicate()方法来使用PIPE给子进程输入：

```
import subprocess
child = subprocess.Popen(["cat"], stdin=subprocess.PIPE)
child.communicate("vamei")
```

我们启动子进程之后，cat会等待输入，直到我们用communicate()输入"vamei"。

通过使用subprocess包，我们可以运行外部程序。这极大的拓展了Python的功能。如果你已经了解了操作系统的某些应用，你可以从Python中直接调用该应用(而不是完全依赖Python)，并将应用的结果输出给Python，并让Python继续处理。shell的功能(比如利用文本流连接各个应用)，就可以在Python中实现。

总结

subprocess.call, subprocess.check_call(),
subprocess.check_output()

subprocess.Popen(), subprocess.PIPE

Popen.wait(), Popen.communicate()

标签: Python

Vamei
关注 - 26
粉丝 - 4985

荣誉：推荐博客

« 上一篇：Linux进程基础

» 下一篇：Linux信号基础

评论列表

#1楼 2012-09-23 21:32 hollyspirit

总结的相当不错。

支持(0) 反对(0)

#2楼 2012-09-23 23:32 Tolbkni Kao

博主，单词拼错了，不是 pearl 而是 perl。

支持(0) 反对(0)

#3楼[楼主] 2012-09-24 09:28 Vamei

@ Tolbkni Kao

:-)，没错，我写错了，谢谢你提醒。

支持(0) 反对(0)

#4楼 2012-12-17 15:58 Honghe

Python对Linux的支持还是高于win32

支持(0) 反对(0)

#5楼 2014-02-10 19:45 M2014

写的真好，比Python编程手册有乐趣，看了几天了。vamei帅！感谢您的辛苦付出：）

关于最后一个例子：

```
import subprocess
child = subprocess.Popen(["cat"], stdin=subprocess.PIPE)
child.communicate("vamei")
```

python2.* 运行没有问题，在 python3.* 下报 TypeError: 'str' does not support the buffer interface 错误，研究了一番，在3.*下改为：

```
import subprocess
```

```
child = subprocess.Popen(["cat"], stdin=subprocess.PIPE)
child.communicate("vamei".encode())
```

即可~

支持(0) 反对(0)

#6楼[楼主] 2014-02-10 23:15 Vamei

@ M2014

谢谢你的热心实验。

准备搞一个开放写书的计划。那样的话，如果你有python 3的代码，就可以加进来。

支持(0) 反对(0)

#7楼 2014-02-11 18:47 M2014

@Vamei

你写的代码细节和文章条理都很好，如果你有开放写书的计划热心读者当然可以多多参与，但多人写一本书也会有好多麻烦，整理不好阅读起来可能会有堆砌的感觉。

我倒希望学好了之后能够为你的作品出一份力呢：）

支持(0) 反对(0)

#8楼 2014-02-11 18:49 M2014

@ Vamei

看了你的博客才注册的ID，:P

支持(0) 反对(0)

#9楼[楼主] 2014-02-11 22:58 Vamei

@ M2014

:-)，谢谢你的捧场。我觉得更多人参与，可以更有效率。关键是有人主导并引导好整个项目。

支持(0) 反对(0)

#10楼 2014-02-21 16:45 特务小强

楼主，目前rhel5.7老版本的操作系统python版本还是停留在2.4.3，升级又不能随便升级，怕影响服务器稳定性，针对这种情况，如果做python2.7开发？

支持(0) 反对(0)

#11楼[楼主] 2014-02-22 03:11 Vamei

@ 特务小强

你可以在home路径下自己装一个2.7

支持(0) 反对(0)

#12楼 2014-04-23 20:38 micfan

nice

支持(0) 反对(0)

#13楼 2014-05-26 10:23 zero_learner

总结的很好，终于懂了

支持(0) 反对(0)

#14楼 2014-05-31 23:52 滚出碗里

关于下面的这个例子

```
import subprocess
child = subprocess.Popen(["cat"], stdin=subprocess.PIPE)
child.communicate("test1.py")
```

我怎么输出不了 cat test1.py中的内容

反而原样输出 [test.py]

想问下大家有没有注意

支持(0) 反对(0)

#15楼[楼主] 2014-06-01 02:09 Vamei

@ 滚出碗里

你在命令行下用cat行吗？

支持(0) 反对(0)

#16楼 2014-06-01 02:21 滚出碗里

@ Vamei

直接cat OK 出来的是内容

支持(0) 反对(0)

#17楼 2014-08-03 21:07 陈不二

```
>>> subprocess.call("pwd",shell=True)
/home/czy/temp
0
>>> subprocess.call("cd ..",shell=True)
0
>>> subprocess.call("pwd",shell=True)
/home/czy/temp
0
>>>
```

为什么了调用了却没有改变pwd？哦哦

支持(0) 反对(0)

#18楼[楼主] 2014-08-03 22:12 Vamei

@ 陈不二

应该每次subprocess是fork出来一个新的进程吧？所以cd不会对pwd产生效果。

支持(0) 反对(0)

#19楼 2015-03-06 16:11 星空刺

@ 滚出碗里

引用

关于下面的这个例子

```
import subprocess
child = subprocess.Popen(["cat"], stdin=subprocess.PIPE)
child.communicate("test1.py")
```

我怎么输出不了 cat test1.py中的内容

反而原样输出 [test.py]

想问下大家有没有注意

和你一样，不晓得为什么

已经知晓了。。。cat后面不是标准输入

支持(0) 反对(0)

#20楼 2015-07-28 16:17 focky

@ 星空刺

那么具体该怎么修改呢？

支持(0) 反对(0)

#21楼 2015-07-28 16:22 focky

Python小白请教楼主几个问题：

1.subprocess.check_output()在我的环境里面实现不了，我的python版本是2.6.6的

2.child.kill()括号里面是输入进程id吗？

3.就是大家都反馈的一个问题，cat原样输出了文件名称，不知道怎么解决。

谢谢楼主！

支持(0) 反对(0)

#22楼 2016-01-18 10:41 lsjwg

总结的太清楚了,谢谢了啊.收藏

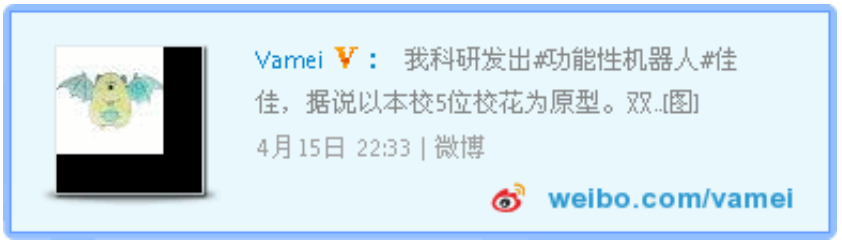
支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。

公告

你好，这里是Vamei，一名编程爱好者。我在博客里写了**Python/Linux/**网络协议**/**算法**/Java/**数据科学系列文章，从这里开始阅读。非常期待和你的交流。

我的微博 

下列教程已经做成电子出版物，内容经过修订，也方便离线阅读：

协议森林

欢迎阅读我写的其他书籍：

现代小城的考古学家

天气与历史的相爱相杀

随手拍光影

昵称：Vamei
园龄：4年1个月
荣誉：推荐博客
粉丝：4985
关注：26
+加关注

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

Python(61)

Java(42)

大数据(22)

Linux(17)

网络(16)

算法(15)

文青(14)

技普(9)

系列索引(6)

开发工具(4)

更多

系列文章

Java快速教程

Linux的概念与体系

Python快速教程

数据科学

协议森林

纸上谈兵: 算法与数据结构

积分与排名

积分 - 659668

排名 - 122

最新评论

1. Re:Java基础11 对象引用

受教!

--MisslLost

2. Re:Python快速教程

看评论区一片喝彩！看来我得在此扎营了！

--测试小蚂蚁

3. Re:Python进阶06 循环对象

好好地列表解析变成了表推导

--ashic

4. Re:“不给力啊，老湿！”：RSA加密与破解

感谢楼主精彩分享

--worldball

5. Re:概率论04 随机变量

你写的这一系列太棒了，刚加入博客园就在你这里学到了，我要转载过去学习一下

--yixius

6. Re:Python基础03 序列

挺好的教程、、、、

--王小拽的号

7. Re:Python进阶07 函数对象

def func(x,y): print x**ydef test(f,a,b): print 'test' print f(a,b)test (func,3,2)
输出的内容:tes.....

--M-edea

8. Re:Python进阶02 文本文件的输入输出

@coderXT换行符：\n...

--行者之印

9. Re:数据科学

博主啊，这里是一枚即将大二的计算机新人，大一学了python，java，还有一些算法，数据结构，图论了，感觉我对数学又一些反感，但是听说离散数学对计算机专业的很重要，不知道怎么去学比较好呢，我想像您写.....

--Acokil

10. Re:为什么要写技术博

楼主是用自己自定义的模板吗？在博客园里找不到这种风格的blog模板？

--行者之印

11. Re:来玩Play框架01 简介

挖煤哥,我补充了一下Windows下的搭建play框架,希望有点帮助,谢谢!

--Sungeek

12. Re:来玩Play框架07 静态文件

@helper.form(action = routes.Application.upload, 'enctype ->
"multipart/form-data") {--action = rout.....

--quxiaozha

13. Re:来玩Play框架07 静态文件

该记录将/assets/下的URL，对应到项目的/public文件夹内的文件。比如在项目的/public/images/test.jpg，就可以通过/assests/images/test.jpg这一.....

--quxiaozha

14. Re:来玩Play框架06 用户验证

支持挖煤哥~~~

--quxiaozha

15. Re:“不给力啊，老湿！”：RSA加密与破解

@maanshancss请你仔细阅读了这个文章再来评价。...

--Vamei

推荐排行榜