

# приостановить выполнение команды оболочки без pid

Спрашивалось 7 лет, 8 месяцев назад · Изменено 10 месяцев назад · Просмотрено 761 раз



1

Мне нужно что-то вроде `$command & stop` Это должно выполнить команду и приостановить ее. Позже приложение возобновит выполнение команды для получения полных результатов.



Я понимаю, что задание может быть приостановлено с помощью сигнала `stop` для соответствующего `pid`.



```
$kill -SIGSTOP 12753
```

Когда мы выполняем команду, мы едва знаем ее `pid`. Для получения `pid` и выполнения требуемого требуется дополнительная команда. Я хочу избежать дополнительной команды и временного интервала.

В основном приложение предназначено для измерения производительности сети. Запуск всех команд переводит их в режим остановки. Остановленные команды возобновляются в соответствии с типом необходимого трафика.

[linux](#) [оболочка](#)

Поделиться · Редактировать

Подписаться

отредактировано 11 июня 2016 г. в 3:59

спрошено 11 июня 2016 г. в 3:17



[sandy\\_1111](#)

373 3 13

Арджун, это слишком вручную. Мне это нужно как скрипт, поскольку в них тысячи команд, которые необходимо перевести в остановленное состояние сразу после выполнения.

– [sandy\\_1111](#) 11 июня 2016 г., 3:30

## 6 Ответов

Отсортировано по:

Highest score (default)



Не нашли ответ? [Задайте вопрос на Stack Overflow на русском.](#)



2

Идентификатор процесса последней запущенной фоновой команды доступен в параметре оболочки `$!`:

```
$ command & kill -SIGSTOP $!
```



(Проверьте документацию по реализации вашей командной строки `kill` на наличие правильного формата.)



Поделиться Редактировать

Подписаться

отредактировано 11 июня  
2016 г. в 12:44

ответил 11 июня 2016 г. в 4:16



**чепнер**

**509 тыс.** 73 548

699

спасибо, чепнер, но, похоже, это не работает, когда мы вводим скрипт. сначала `-SIGSTOP` выдал ошибку **run.sh: строка 2: kill: SIGSTOP: неверная спецификация сигнала** использовалось числовое значение signal (19) скрипт работал нормально, но остановленные процессы не перечислены в `ps -eaf - санди_1111` 11 июня 2016 в 5:07 ✎

@WarrenYoung это тоже неправильно. Фоновой команды больше нет; `kill` не будет выполняться, пока команда не завершится, и кто знает, на что `$!` установлено значение. **-чепнер** 11 июня 2016 в 12:09

@chepner: Вы правы в `&`, конечно. Что касается `-STOP` вместо `-SIGSTOP`, оба работают в Linux или OS X, поскольку они работают под управлением Bash, и вы используете встроенный Bash в этом случае. Вы также можете использовать `-s STOP` или `-s SIGSTOP` в этих системах. Все это, кажется, даже работает с `/bin/kill` тоже. **-Уоррен Янг** 11 июня 2016 в 12:50

`dash` по крайней мере, возвращает значение `Illegal option -S`, поэтому OP может использовать какую-то другую оболочку, использующую другой формат. **-чепнер** 11 июня 2016 в 13:16



**2**



Пожалуйста, подтвердите, хотите ли вы остановить свою команду или выполнить ее в фоновом режиме (добавьте `&` к своей команде)?

Если ожидается, что ваше приложение позже запустит остановленную команду, то почему бы вам не запустить свою команду (для остановки) в самом этом приложении.

Это помогает :

```
sleep 5 & kill -SIGSTOP $!
```

Выше мы выполнили `sleep` (демонстрационную команду) в течение 5 секунд в фоновом режиме. Затем отправим на `kill` для остановки, используя его PID, полученный с помощью `$!`.

Поделиться Редактировать

Подписаться

отредактировано 11 июня  
2016 г. в 4:33

ответил 11 июня 2016 г. в 3:47





**Рохит Верма**

**457** 2 13

[meta.stackoverflow.com/questions/265783/...](https://meta.stackoverflow.com/questions/265783/...) **-Арьян** 11 июня 2016 в 3:49

Арьян, 2-я строка - это ответ :) **-Рохит Верма** 11 июня 2016 в 3:52

@ Rohit Verma, запуск каждой новой команды как дочерней занимает много времени. Поэтому я сначала запускаю все необходимые команды как процесс, а позже возвращаюсь, чтобы активировать его. – [sandy\\_1111](#) 11 июня 2016 в 4:06 

привет, sandy\_1111, у тебя был ответ. `command & kill -SIGSTOP $!` – [Рохит Верма](#) 11 июня 2016 в 4:27 

@RohitVerma хотя это и привлекательно, но ошибочно. что, если бы команда была `echo "привет"` ? тогда `$!` было бы `echo "привет"`. для `kill` требуется имя команды, которое редко является командой. – [Баруди Сафвен](#) 11 июня 2016 в 8:49



1

Demo & kludge использует **тайм-аут** (по какой-то причине **тайм-аут** интерпретирует длительность "0 секунд" как "выполняться вечно"), чтобы остановить **yes** до того, как он что-либо выведет:



```
# run 'yes' command, let it print 5 numbered lines, but stop it immediately
timeout -s SIGSTOP .000000001s yes | head -n 5 | cat -n
```



Вывод (в STDERR):

```
[1]+  Stopped      timeout -s SIGSTOP .000000001s yes | head -n 5 | cat -n
```

Теперь перезапустите его:

```
fg > /dev/null
```

Вывод:

```
1  y
2  y
3  y
4  y
5  y
```

Метод для пользователей, застрявших на версии 8.12 или более ранней **coreutils** (до 2011 года), в котором в **таймауте** отсутствуют *субсекундные интервалы*. Требуется секундное ожидание.

Оберните командную строку в вызов командной строки, которому предшествует ожидание в 1 секунду - таким образом, **тайм-аут** ожидает 1 секунду, и одновременно то же самое происходит с командной строкой. Общее время ожидания 1 секунда:

```
timeout -s SIGSTOP 1s sh -c "sleep 1s; yes | head -n 5 | cat -n"
```

Вывод такой же, как и раньше, `fg` тоже такой же.

```
timeout -s SIGSTOP 1s sh -c "sleep 1s; yes | head -n 5 | cat -n" &
```

[1] 14601

8191 2 30 50

в RHEL она принимает ее только в виде целых чисел, а не с плавающей точкой. таким образом, минимум, который я могу подождать, составляет 1 секунду ..! – [sandy\\_1111](#) 13 июня 2016 в 6:56



1



417

затем возобновить ее:

```
kill -SIGCONT $(pidof command_name)
```

если имя команды не является постоянным, но есть шаблон, вы можете создать подобный скрипт, вы можете вызвать его `pof.sh`:

```
ps -ely | grep $1 | tr -s ' ' | cut -d" " -f3
```

```
command & kill -SIGSTOP $(bash pof.sh pattern)
```


Одним из недостатков этого скрипта является то, что в случае, если многие строки соответствуют шаблону, он вернет все их `pid`, если это проблема, вы можете поместить выходные данные в массив и продолжить оттуда.

Поделиться Редактировать

отредактировано 13 июня  
2016 г. в 18:00

ответил 11 июня 2016 г. в 4:27

Подписаться

 **Ав Баруди Сафвен**  
поль 812 7 17

что, если предполагается, что команда выполняется в фоновом режиме ..? это **command && kill -SIGSTOP \$(pidof command\_name)!!!**. Это приведет к ошибке **-bash: синтаксическая ошибка рядом с неожиданным токеном `& &'** – [sandy\\_1111](#) 13 июня 2016 г., 5:01

Решением проблемы было бы `command & kill -SIGSTOP $(pidof имя_команды)`  
– [Баруди Сафвен](#) 13 июня 2016 г. в 17:55 ✎



1



Попробуйте `killall` использовать опцию **--signal**, где вы можете указать имя процесса.

```
linux:~ # killall
Usage: killall [OPTION]... [--] NAME...
    killall -l, --list
    killall -V, --version
```

<code>-e,--exact</code>	require exact match for very long names
<code>-I,--ignore-case</code>	case insensitive process name match
<code>-g,--process-group</code>	kill process group instead of process
<code>-i,--interactive</code>	ask for confirmation before killing
<code>-l,--list</code>	list all known signal names
<code>-q,--quiet</code>	don't print complaints
<code>-r,--regexp</code>	interpret NAME as an extended regular expression
<code>-s,--signal SIGNAL</code>	send this signal instead of SIGTERM
<code>-u,--user USER</code>	kill only process(es) running as USER
<code>-v,--verbose</code>	report if the signal was successfully sent
<code>-V,--version</code>	display version information
<code>-w,--wait</code>	wait for processes to die

Проверяется запуском `md5sum` в сеансе командной строки:

```
linux$ md5sum
```

и в другом сеансе запустил:

```
killall -s SIGSTOP md5sum
```

В md5sum сеансе это приводит к следующему:

```
[1]+  Stopped                  md5sum
```

Поделиться Редактировать

Подписаться

отредактировано 11 июня  
2016 г. в 3:30

ответил 11 июня 2016 г. в 3:23



bvj

3324

32

31

thanks bvj, but does `killall` command applies to all the relevant commands..? **`killall -s SIGSTOP md5sum`** would stop all the running `md5sum`. Should extra care be taken to match a exact command using regular expression..? – [sandy\\_1111](#) Jun 11, 2016 at 3:36

@sandy\_1111 It matches exact names, so `killall -s SIGSTOP md5` will not stop `md5sum`. – [Arjan](#) Jun 11, 2016 at 3:53

@sandy\_1111 Well, you could have processes with identical names but might be different binaries distinguished by their respective paths. If you know where your commands live, you can parse the results of `ps -Af` to determine which processes you wish to suspend. – [bvj](#) Jun 11, 2016 at 5:32



Plainly answering the title.

0

Pausing a running command, e.g. FFmpeg:



```
kill -17 `ps aux | grep -E "\s[f]mpeg\s" | awk '{print $2}'`
```



Or, even simpler:



```
kill -17 `pidof ffmpeg`
```

Use `-17` or `-18` —read [here](#) to know the difference.

I prefer to use `pgrep` as it seems a bit more reliable than `pidof`:

```
kill -17 `pgrep ffmpeg`
```

Or, create your own “command”:

```
pid () { ps aux | grep -E "\s$(echo $1 | sed -r 's/(\S)(\S*)/\[\1\]\2/' )\s" | awk '{print $2}' }
```

Thus, pausing/resuming becomes simpler:

```
$ kill -17 `pid ffmpeg`  
$ kill -19 `pid ffmpeg`
```

There's also the `pkill` command:

```
pkill -17 -f ffmpeg
```

But, it's less reliable than combining `kill` with `pgrep` or `pid`.

The reason why it's less reliable: the `pkill` command will target **any** process containing *ffmpeg* word—and not only `ffmpeg` processes.

Share Edit Follow

edited Apr 16, 2023 at 9:42

answered Apr 15, 2023 at 11:21



Faxopita

83 1 9