# Linux

**版本不确定**

## **[php 异步执行脚本](http://www.netingcn.com/php-asynchronous-execute-script.html)**

2013年8月1日[admin](http://www.netingcn.com/author/admin" \o "由admin发布)[没有评论](http://www.netingcn.com/php-asynchronous-execute-script.html" \l "respond" \o "《php 异步执行脚本》上的评论)

这里说的异步执行是让php脚本在后台挂起一个执行具体操作的脚本，主脚本退出后，挂起的脚本还能继续执行。比如执行某些耗时操作或可以并行执行的操作，可以采用php异步执行的方式。主脚本和子脚本的通讯可以采用外部文件或memcached的方式。原理就是通过exec或system来执行一个外部命令。注意：本文所述的是针对Linux环境。

在Linux下要让一个脚本挂在后台执行可以在命令的结尾加上一个 “&” 符号，有时候这还不够，需要借助nohup命令，关于nohup，可以参考[http://www.netingcn.com/linux-nohup.html](http://www.netingcn.com/linux-nohup.html" \o "http://www.netingcn.com/linux-nohup.html)。

CLI环境和Web环境执行的操作还不太一样。先来说CLI环境，这里需要用上nohup和&，同时还要把指定输出，如果不想要输出结果，可以把输出定向到/dev/null中。现在来做一个测试，假设在一个目录中有main.php、sub1.php和sub2.php，其中sub1和sub2内容一样都让sleep函数暂停一段时间。代码如下：

//main.php

<?php

$cmd = 'nohup php ./sub.php >./tmp.log &';

exec($cmd);

$cmd = 'nohup php ./sub1.php >/dev/null &';

exec($cmd);

?>

//sub1.php sub2.php

<?php

sleep(100000);

?>

上述文件中main.php是作为主脚本，在命令行中执行php main.php，可以看到main.php脚本很快就执行完并退出。在使用ps aux | grep sub命令搜索进程，应该可以在后台看到上述的两个子脚本，说明成功挂起了子脚本。

在Web环境下，执行php脚本都是Web服务器开启的cgi进程来处理，只要脚本不退出，就会一直占有该cgi进程，当启动的所有cgi进程都被占用完后就不能在处理新的请求。所以对那些可能会很费时的脚本，可以采用异步的方式。启动子脚本的方式和CLI差不多，必须要使用&和指定输出（只好是定向到/dev/null），但是不能使用nohup。例如：

<?php

$cmd = 'php PATH\_TO\_SUB1/sub1.php >/dev/null &';

exec($cmd);

$cmd = 'php PATH\_TO\_SUB1/sub2.php >/dev/null &';

exec($cmd);

?>

当在浏览器中访问该脚本文件，可以看到浏览器里面响应完成，同时使用ps命令查看后台可以看到sub1和sub2脚本。

注意上述例子中如果php命令不在PATH中，需要指定命令完整的路径。推荐使用完整路径，特别是在Web下。

# Windows cmd

No version check

一，国外佬大招

<http://nikic.github.io/2012/12/22/Cooperative-multitasking-using-coroutines-in-PHP.html>

我发现自己是php5.6，所以选用了这个协程的解决方案

试过后，发现协程，还是会阻塞

二、使用fsockopen的非阻塞模式请求另外的网址

[view plainprint?](http://www.justwinit.cn/)

1. $fp = fsockopen("www.example.com", 80, $errno, $errstr, 30);
2. if (!$fp) die('error fsockopen');
3. stream\_set\_blocking($fp,0);
4. $http = "GET /save.php  / HTTP/1.1\r\n";
5. $http .= "Host: www.example.com\r\n";
6. $http .= "Connection: Close\r\n\r\n";
7. fwrite($fp,$http);
8. fclose($fp);

三、使用Gearman  
Gearman是一个具有php扩展的分布式异步处理框架，能处理大批量异步任务。

四、pcntl\_fork();

试过，提示undefined function ,apache模式不能用，要在 cgi模式下用。

五，一哥们的方法，没验证过

<http://blog.csdn.net/linvo/article/details/5466046>

六，更详细的解析

<http://www.dadait.com/archives/Executing-background-processes-from-PHP-on-Windows.html>

# Windows Shell

no

在exec执行脚本的最后加上 “2>&1 &”