


- [Home](#)
-  [Subscribe](#)

Mike Nott

SEO, Music, Photography & Other Stuff

PHP Crawler

Posted by Mike

For crawling in [PHP](#) I have always used the fantastic [cURL](#) .

My curl single-threaded function:

```
[code lang="php"]
```

```
function singlethread_crawl($url)
{
    $agent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)";

    $ch = curl_init();

    curl_setopt($ch, CURLOPT_NOSIGNAL, 1);
    curl_setopt($ch, CURLOPT_NOPROGRESS, 1);
    curl_setopt($ch, CURLOPT_FAILONERROR, 1);
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_USERAGENT, $agent);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
    curl_setopt($ch, CURLOPT_MAXREDIRS, 1);
    curl_setopt($ch, CURLOPT_TIMEOUT, 5);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

    $html = curl_exec($ch);

    curl_close ($ch);

    return $html;
}
```

```
[/code]
```

My curl multi-threaded function:

```
[code lang="php"]
```

```
function multithread_crawl($urls, $timeout, $verbose)
{
    $agent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)";

    $mh = curl_multi_init();

    foreach ($urls as $i => $url)
    {
        $conn[$i] = curl_init($url);
        curl_setopt($conn[$i], CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($conn[$i], CURLOPT_NOSIGNAL, 1);
        curl_setopt($conn[$i], CURLOPT_NOPROGRESS, 1);
        curl_setopt($conn[$i], CURLOPT_FAILONERROR, 1);
        curl_setopt($conn[$i], CURLOPT_URL, $url);
```

```
curl_setopt($conn[$i], CURLOPT_USERAGENT, $agent);
curl_setopt($conn[$i], CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($conn[$i], CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($conn[$i], CURLOPT_MAXREDIRS, 1);
curl_setopt($conn[$i], CURLOPT_TIMEOUT, $timeout);

curl_multi_add_handle ($mh, $conn[$i]);
}

do
{
    $mrc = curl_multi_exec($mh, $active);
}
while ($mrc == CURLM_CALL_MULTI_PERFORM);

while ($active and $mrc == CURLM_OK)
{
    if (curl_multi_select($mh) != -1)
    {
        do
        {
            $mrc = curl_multi_exec($mh, $active);
        }
        while ($mrc == CURLM_CALL_MULTI_PERFORM);
    }
}

if ($mrc != CURLM_OK)
{
    print "Curl multi read error $mrc\n";
}

$res = array();
$e = 0;

foreach ($urls as $i=> $url)
{
    if (($err = curl_error($conn[$i])) == "")
    {
        $res[$i]=curl_multi_getcontent($conn[$i]);
    }
    else
    {
        if ($verbose == "yes"){
            echo "error: ".$url." (".$err.")\n";
        } else {
            $e++;
        }
    }
}

curl_multi_remove_handle($mh,$conn[$i]);
curl_close($conn[$i]);
}

curl_multi_close($mh);

$s = count($urls)-$e;

if ($verbose == "no"){
    echo "errors ".$e." | success ".$s."\n";
}

return $res;
```

```
}
```

```
[/code]
```

However there are some annoyances in curl – the main one for me being that you can't pass variables to the `write_function`,

```
[code lang="php"]
curl_setopt($conn[$i], CURLOPT_WRITEFUNCTION, myfunction);
[/code]
```

which makes it useless for updating rows etc in a db (you can use [curl_getinfo](#) to get the url so do a lookup – but that is pretty backwards). This means that the crawling is not even close to being truly multithreaded as you have to wait for all urls to finish before working with the data.

So I thought I'd have a go at writing the raw crawler myself using [fsockopen](#). Is not perfect as the multithread function does require the single thread one to follow any redirects.

My own single-threaded function:

```
[code lang="php"]

function mycrawler_single($url, $timeout=10, $maxredirs=1)
{
    $urlinfo = parse_url($url);

    if (empty($urlinfo['scheme'])) {$urlinfo = parse_url('http://'.$url);}
    if (empty($urlinfo['path'])) {$urlinfo['path']="/";}

    if (empty($urlinfo['port']))
    {
        switch($urlinfo['scheme'])
        {
            case "http":
                $urlinfo['port'] = 80;
                break;
            case "https":
                $urlinfo['port'] = 443;
                break;
        }
    }

    if (isset($urlinfo['query']))
    {
        $request = "GET ".$urlinfo['path']."?". $urlinfo['query']." ";
    } else {
        $request = "GET ".$urlinfo['path']." ";
    }

    $request .= "HTTP/1.0\r\n";
    $request .= "Host: ".$urlinfo['host']."\r\n";
    $request .= "User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)\r\n";
    $request .= "Connection: close\r\n\r\n";

    $fp = fsockopen($urlinfo['host'], $urlinfo['port'], $errno, $errstr, $timeout);

    if (!$fp)
    {
        echo "(".$errno.")". $errstr. "\n";
    }
    else
    {
        fwrite($fp, $request);
    }
}
```

```

while (!feof($fp))
{
    $data .= fgets($fp, 4096);
}

fclose($fp);

$tmp = explode("\r\n\r\n", $data, 2);

$urlinfo['header'] = $tmp[0];
$urlinfo['html'] = $tmp[1];

if ((strpos($urlinfo['header'], "location:") && ($maxredirects > 0))
{
    preg_match("/\r\nlocation:(.*)/i", $urlinfo['header'], $match);

    if ($match)
    {
        $redirect = trim($match[1]);

        echo "Redirecting to ".$redirect."";

        $maxredirects--;

        return mycrawler_single($redirect, $timeout, $maxredirects);
    }
}

return $urlinfo;
}
}

```

[/code]

My own multi-threaded function:

[code lang="php"]

```

function mycrawler_multi($urls, $timeout=10, $maxredirects=1)
{

    for ($i=0; $i {
        $urlinfo[$i] = parse_url($urls[$i]);
        $maxredirects[$i] = $maxredirects;

        if (empty($urlinfo[$i]['scheme'])) {$urlinfo[$i] = parse_url('http://'.$url);}
        if (empty($urlinfo[$i]['path'])) {$urlinfo[$i]['path']="";}

        if (empty($urlinfo[$i]['port']))
        {
            switch($urlinfo[$i]['scheme'])
            {
                case "http":
                    $urlinfo[$i]['port'] = 80;
                    break;
                case "https":
                    $urlinfo[$i]['port'] = 443;
                    break;
            }
        }

        if (isset($urlinfo[$i]['query']))
        {
            $request[$i] = "GET ".$urlinfo[$i]['path']."?".urlinfo[$i]['query']." ";

```

```

    } else {
$request[$i] = "GET ".$urlinfo[$i]['path']." ";
    }

$request[$i] .= "HTTP/1.0\r\n";
$request[$i] .= "Host: ".$urlinfo[$i]['host']."\r\n";
$request[$i] .= "User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)\r\n";
$request[$i] .= "Connection: close\r\n\r\n";

$fp[$i] = fsockopen($urlinfo[$i]['host'], $urlinfo[$i]['port'], $urlinfo[$i]['errno'], $urlinfo[$i]['errstr'], $timeout);

socket_set_blocking($fp[$i], false);

if (!$fp[$i])
{
echo "(".$urlinfo[$i]['errno'].")".$urlinfo[$i]['errstr']."\n";
}
else
{
fwrite($fp[$i], $request[$i]);
}
}

$done = false;
$numdone = array();

while (!$done)
{
for ($i=0; $i {
if (!feof($fp[$i]))
{
$data[$i] .= fgets($fp[$i], 4096);
}
elseif (empty($numdone[$i]))
{
$numdone[$i] = 1;

$tmp[$i] = explode("\r\n\r\n", $data[$i], 2);

$urlinfo[$i]['header'] = $tmp[$i][0];
$urlinfo[$i]['html'] = $tmp[$i][1];

if ((strpos($urlinfo[$i]['header'], "location:")) && ($maxredirects[$i] > 0))
{
preg_match("/\r\nlocation:(.*)/i", $urlinfo[$i]['header'], $match[$i]);

if ($match[$i])
{
$redirect[$i] = trim($match[$i][1]);

echo "Redirecting to ".$redirect[$i]."\n";

$maxredirects[$i]--;

$urlinfo[$i] = mycrawler_single($redirect[$i], $timeout, $maxredirects[$i]);
}
}
}
}

$done = (array_sum($numdone) == count($urls));
}

```

```
for ($i=0; $i {  
    fclose($fp[$i]);  
}
```

```
return $urlinfo;  
}
```

```
[/code]
```

All require PHP5.

Post Info and Links

Get more out of this post by commenting and keeping track of it using these online tools.

Posted on: December 31, 2005

Comments: [9 Responses](#)

Tags: [Code](#), [PHP](#)

 [Trackback](#)

Link using:  [Delicio.us](#)

 [Digg](#)

 [Technorati](#)

Leave a Comment

Name (Required)

Mail (Not published) (Required)

Website

Comments

1

Written by: [Jeff Beck](#)

Posted on: January 18, 2006 at 7:24 pm

I've been trying to do much of the same thing on a site I have been working on. The problem I have been running up against has been fSockOpen; it waits for a connection to the server before returning. As such, when you loop through an array of URLs, it can take maybe twice as long to connect and retrieve data as with cURL multi. The fastest solution I have seen involved fSockOpen connecting to the local machine (aka PHP multithread HACK), and then letting each PHplet file do the crawling independantly. This cuts even the cURL multi in half, but is not very pratical on the server load (1 page = 30+ httpd files running). Speed wise, have you compared your PHP function with the cURL function?

2

Written by: [HM2K](#)

Posted on: June 1, 2006 at 2:26 pm

You never once actually said what this function does, "PHP Crawler" is VERY generic.

Also you have not shown any examples of output.

3

Written by: [Mike](#)

Posted on: September 13, 2006 at 9:40 am

If you don't know what to do with the code, then it isn't for you 😊

4

Written by: [Rishiraj](#)

Posted on: October 15, 2007 at 4:19 pm

I have freeernti hosting server with php 4.7.

Is there any alternative for me if i don't want to use curl?

5

Written by: [Mike](#)

Posted on: October 15, 2007 at 4:25 pm

Rishiraj – use the 'mycrawler_single' function described in the post above. It should only need minimal tweaking to work in php 4.

6

Written by: [Rays](#)

Posted on: November 11, 2007 at 7:40 am

I'm confuse..

Somebody help me please?

Maybe, i must create auto crawler engine.

Hemm.. i need team work for this time!

7

Written by: [David Arakelian](#)

Posted on: February 17, 2008 at 6:05 pm

Thanks for sharing your code on the use of the cURL multi functions. I run a lot of scripts that take a long time to execute (4 – 5 days in some cases) because I am using a single stream. Using your code and some proxies I can probably get this down to a few hours 😊

8

Written by: [Tuvok](#)

Posted on: June 10, 2009 at 4:01 pm

Thanks. It's nice to see examples of very useful things.

Which option is best for a 100,000+ url crawl project, with hardware not being an obstacle ?

9

Written by: [Air Force 1](#)

Posted on: September 6, 2009 at 11:06 am

This is a wonderful script. But I used snoop

Friends & Peers

- [Andre Chaperon](#)
- [Ben Lovell](#)
- [Dean Chew](#)
- [Jane Copland](#)
- [Julie Joyce](#)
- [LondonSEO](#)
- [Lost Twenty](#)
- [Paul Humphreys](#)
- [Paul Madden](#)

- [Rob Kerry](#)
- [SEO Philippines](#)
- [Steve Hill](#)
- [Tony Spencer](#)

Find More Posts

Search for posts using the search form or by selecting a keyword from the list below

Search 

- [Ayima](#)
- [Blog](#)
- [Code](#)
- [Computers](#)
- [Music](#)
- [Other Stuff](#)
- [PHP](#)
- [Search](#)
- [Twitter](#)
- [Web/Net](#)
- [Wordpress](#)

Archives

- [June 2012](#)
- [May 2012](#)
- [March 2012](#)
- [February 2012](#)
- [January 2012](#)
- [December 2011](#)
- [November 2011](#)
- [October 2011](#)
- [August 2011](#)
- [May 2011](#)
- [April 2011](#)
- [March 2011](#)
- [February 2011](#)
- [January 2011](#)
- [December 2010](#)
- [August 2010](#)
- [July 2010](#)
- [June 2010](#)
- [May 2010](#)
- [April 2010](#)
- [March 2010](#)
- [February 2010](#)
- [January 2010](#)
- [December 2009](#)
- [November 2009](#)
- [October 2009](#)
- [September 2009](#)
- [August 2009](#)
- [July 2009](#)
- [June 2009](#)
- [May 2009](#)
- [April 2009](#)
- [March 2009](#)
- [February 2009](#)
- [September 2008](#)

- [June 2008](#)
 - [March 2008](#)
 - [December 2007](#)
 - [October 2007](#)
 - [September 2007](#)
 - [January 2006](#)
 - [December 2005](#)
 - [November 2005](#)
-

©2007 Mike Nott