

Downloading a Remote File With cURL and PHP

By Quentin Zervaas, 24 March 2010, tagged in [cURL](#), [PHP](#)

cURL is a great tool to help you connect to remote web sites, making it easy to post forms, retrieve web pages, or even to download files. In this PhpRiot Snippet I'll show you how you can download a file straight to disk using cURL.

Note: To simplify our key points in this article we don't perform any error checking on the cURL requests. You should always do so; the `curl_getinfo` function is extremely useful for this.

If you use basic options when executing a cURL request, the returned data will be output directly. If instead you want to assign the response to a PHP variable you would specify the `CURLOPT_RETURNTRANSFER` option.

You can then read the response and write it to disk, as shown in the following listing. The `$path` variable is the location on the server where you want to write the file.

Listing 1. Downloading a file and saving it with `file_put_contents()` (`listing-1.php`)

```
<?php
$url = 'http://www.example.com/a-large-file.zip';
$path = '/path/to/a-large-file.zip';

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$data = curl_exec($ch);

curl_close($ch);

file_put_contents($path, $data);
?>
```

There is however a problem with this code. If the file you're downloading is quite large, the entire contents must be read into memory before being written to disk. Doing so can result in your script breaking down due to exceeding the memory limit.

Note: Even if your memory limit is set extremely high, you would be putting unnecessary strain on your server by reading in a large file straight to memory.

Therefore, you should let cURL do the hard work by passing to it a writable file stream. You can do this using the `CURLOPT_FILE` option.

Note: Because we'll be writing to a file you no longer specify the `CURLOPT_RETURNTRANSFER` option.

To do this you must first create a new file pointer using `fopen()`. Next we pass this file pointer to cURL and perform the request. Finally, we close the file pointer.

Listing 2. Using cURL to download straight to a file stream (`listing-2.php`)

```
<?php
$url = 'http://www.example.com/a-large-file.zip';
$path = '/path/to/a-large-file.zip';

$fp = fopen($path, 'w');

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_FILE, $fp);

$data = curl_exec($ch);

curl_close($ch);
fclose($fp);
?>
```

That's all there is to it. You can now download files without worrying about exceeding PHP's memory limit.