

PHP Speedy

In [my previous article](#) I discussed various ways to improve website performance. I mentioned [PHP Speedy](#), a script from [Aciddrop](#).

PHP Speedy has 3 main functions...

1. “Minify” your code, JavaScript and CSS.
2. GZIP code, JavaScript and CSS.
3. Add Far_future_expires for JavaScript and CSS

Now, there’s not a lot of documentation on what exactly it does, just how to use it. However, PHP Speedy is based on a number of utilities that the author had been working on, so his blog comments on those can give us a clue.

For a start, PHP Speedy will combine JavaScript and CSS scripts – so if your code references a number of these, PHP Speedy will change it to just the one, reducing the number of server calls.

Here are the options in more detail...

Minify

This is the most unknown of the options, other than the fact that the JavaScript part of this is known to be a version of [JSMIn](#).

What we do know is that it reduces your code size by removing redundant code. I contacted the author of PHP Speedy, Leon Chevalier, and he confirmed more of the detail. For CSS files whitespace and comments are removed. For code all leading spaces, tabs and carriage returns NOT preceded by a PHP close tag are removed.

I’m sure this could be more robust – for example why not remove tabs and carriage returns for CSS – but is probably nearly as good as minifying your code beforehand.

So, the big question is, does minifying your code before using PHP Speedy make a difference? Well, I ran one of my sites CSS through the [online compressor at CSSdrive](#) – this reduced it by 25%. However, according to [Yslow](#), it made little, or no, difference in speed. The conclusion must therefore be that any compression before using PHP Speedy is a waste of time. This will save a lot of effort long term.

GZIP

GZIP can be activated via your .htaccess file but some people, myself included, may find even this impossible with their current host. There are script options, and I’m assuming this is what PHP Speedy is making use of – it’s certainly performing a GZIP on the code, CSS and JavaScript, as stated.

PHP Speedy

Far Future Expires

Now, this is the bit I've always been afraid of. If used manually, you have to remember to provide files with different filenames whenever they change. Otherwise the previous cached version will be used.

PHP Speedy is clever though – if you remember, it will combine CSS and JavaScript's (even if there's just one) and give it its own name. This remains the same UNTIL you change any of the files, then the filename is modified. This means YOU don't have to worry about changing the filename whenever the code changes – PHP Speedy does it for you.

What does this mean? If you haven't changed any of the JS or CSS files, then it will be cached at the users end.

Conclusion

PHP Speedy is an incredibly well thought out and useful utility – it puts together a lot of the principles from my original article, and does so in a way that's easy to implement. It does a lot of stuff on the fly saving a lot of time and effort.

It would be nice if there was more information available on what's it doing, rather than how good it is.

But, the proof is in the pudding, and this very blog is now working with PHP Speedy (all options turned on).

Which reminds me; during the install there are instructions on how to add PHP Speedy to WordPress. This involved adding code to index.php. I found this only worked on the main blog page and no others, but have found that adding the code to the header.php and footer.php instead works a treat.