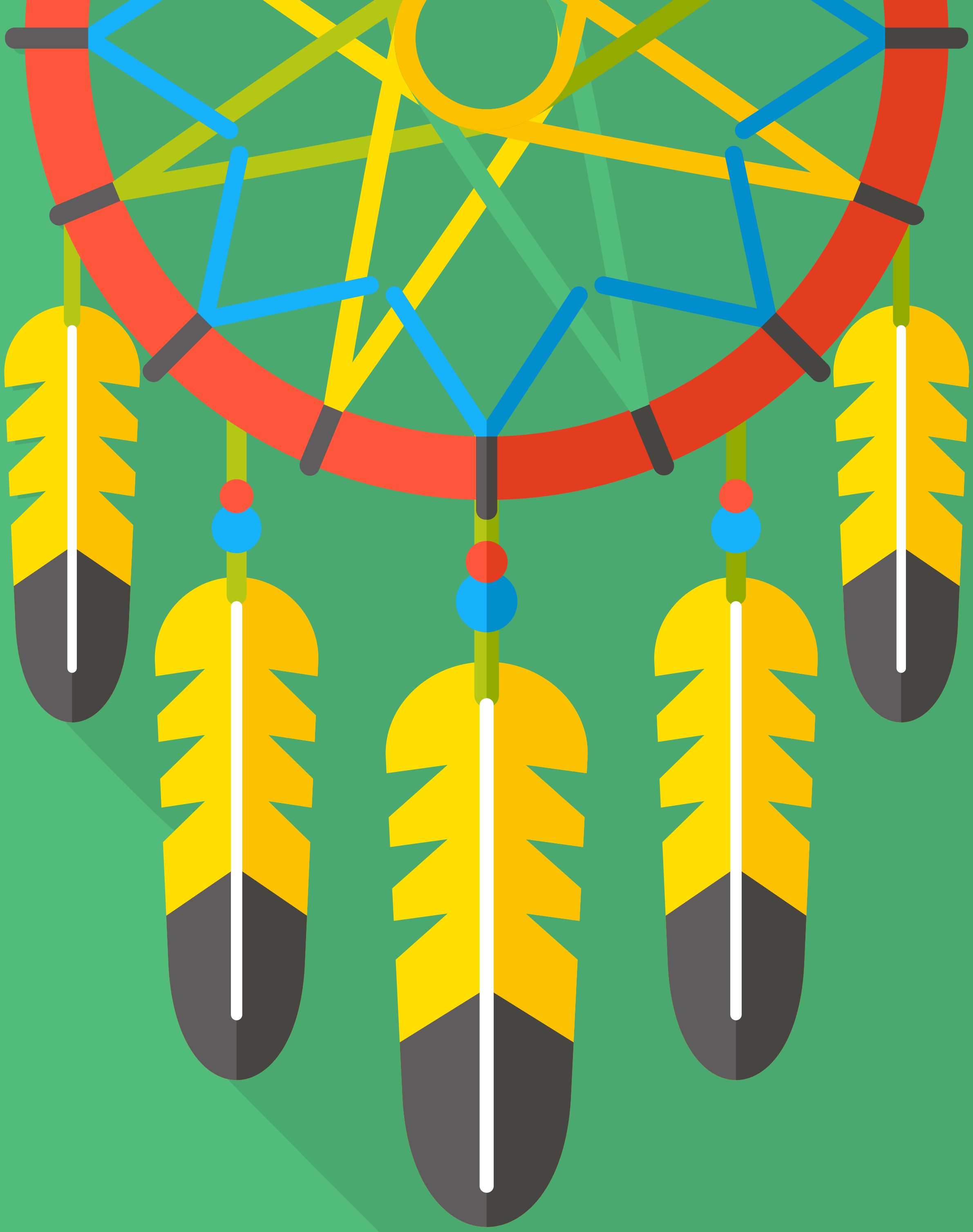


Everything you need to know about caching

Caching is a word that gets thrown around a lot. "Caching will increase your site speed!" "Caching is key to your site's performance!" But, what exactly is it? And why is it so important when you're designing a site?

This ebook will help you understand just that; what website caching is, when to use it, and how to use it effectively. Ready to cash in by caching your WordPress site? Let's dive in!

Alright, so what exactly does caching do?

According to the dictionary, a cache is a collection of items stored in a hidden place. When it comes to your site, a cache is really no different: it's a temporary location on a server and/or your computer that holds a website's assets.

There are two types of caching that we'll be talking about today: server-side caching and browser-side caching. Let's start with the browser side of things.

> A cache is...a temporary location on a server and/or your computer that holds a website's assets.

Basically, when a user loads your site for the first time, the browser will store, or cache, the contents of your site locally. This includes HTML files, CSS stylesheets, images, and any other assets your site may contain.

When your site's content is cached by a user's browser, that means the browser already has all of the assets it needs to load your site. This is beneficial because the next time that same user visits your site, the browser will be able to load the content without having to retrieve everything from the server again, improving your site's performance.

Server-side caching is basically the same idea, but the cache takes place on the server level instead of the browser. This can save a lot of time loading content because the server doesn't have to use PHP to communicate to the database every time a page needs to load.

Once your site's content is cached, whether it's on the browser, the server-side, or both, your site will load much faster for your users.This makes caching incredibly important when it comes to your site's speed and performance.

Ok, so how does all of this work?

When you visit a WordPress website, a large number of things happen all at once. A single request from your computer or phone to an uncached WordPress site typically looks like this:

1. A user wants to load a web page, so they enter the URL.
2. The browser sends an HTTP request to the server.
3. The server gets the request.
4. The server passes the request to PHP to be executed.
5. PHP makes a request to the database for the site's content.
6. The database responds with the information that PHP needs.
7. PHP generates all the static HTML.
8. PHP hands off all information back to the server.
9. The server sends all information to the browser.
10. The web page is loaded for the user.

If a user has already visited your site, browser caching can help speed things up by eliminating the need to talk to the server. The only time caching doesn't improve performance is if it's a user's first time visiting your site, or if you have new, updated content, because then the browser still has to communicate with the server and complete all of those tasks.

If your site is experiencing a low amount of traffic, these requests can happen fairly quickly. However, the moment traffic begins to scale up, this process quickly turns into a series of bottlenecks that have to be navigated by the server over and over, consuming resources until page loads are crawling or, in some circumstances, the website crashes.

Here's where a server-side caching engine comes into play. A caching engine is a software process that sits between the web server and WordPress and its database.

With a caching engine, the server actually saves the output of the long process we talked about above for a set amount of time to automatically re-serve the same content to new users. That way, instead of executing the same code and talking to the database 1,000 times for 1,000 users visiting the same page, the server executes the code one time, caches the result, and serves it back up 999 times.

In other words, server-side caching is where you can really make a difference on performance. Not only will server-side caching reduce the load on your server, but pages will load significantly faster because the server just has to grab and serve the saved copy, instead of executing code, querying the database, and generating the output.

But, how do users see updated content then?

Don't fret! You'll always be able to serve up your newest content for your users. You just have to define the caching rules that are best for your site's content.

> "Static assets that don't change often are usually pretty safe to cache for longer periods of time.

Static assets that don't change often are usually pretty safe to cache for longer periods of time. These would be assets such as your logo, any main imagery, CSS stylesheets, etc. Basically anything that you don't foresee yourself updating for a while, you can go ahead and plan to cache that content for a greater period of time.

On the flip side, if you have content that changes frequently, you'll want to consider caching it for less time to make sure your users are always getting the latest content. These would be aspects of your site such as a blog or eCommerce section where content is frequently changing and being updated.

Don't worry though – even if "outdated" cached content is being loaded, that usually doesn't last long before the cache resets (depending on your settings, of course). Even on massive websites with frequently updating and changing content, caching is still used extensively. Setting server-side caching to save pages for ten minutes, for example, will result in a massive speed increase and generally won't affect content delivery.

If you're concerned, though, you always have the option to manually flush your site's cache.

Awesome! So how does browser caching work?

It's easy, I promise! But first, let's quickly go over some of the technical aspects of caching.

When a user loads a web page, the browser sends an HTTP request.
That request looks something like this:
The first line, GET, is the request the browser is sending to the server.

```
GET /design-and-wordpress-resources/ebooks HTTP/1.1
Host: www.getflywheel.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_1) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/48.0.2564.103 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Cache-Control: public
```

The rest of the lines are all examples of HTTP headers, which is how your site can communicate additional information. Notice the very last line, Cache-Control: public. This is one of the many caching headers you can use to tell the browser how to cache your site.

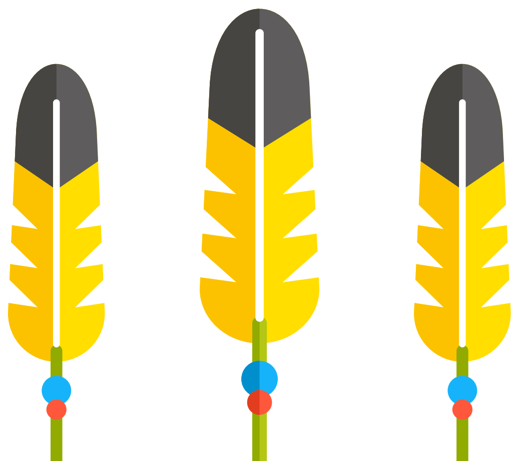
Some popular headers you can use to give the browser caching instructions are:

- Cache-Control
- Date
- ETag
- Expires
- If-Modified-Since
- If-None-Match
- Last-Modified

We'll do a quick overview of each type -- but don't worry, there's no test afterwards!

Cache-control

One of the most popular options, cache-control is simply a powerful header for defining multiple caching rules. With five values that can be comma-separated, you can mix and match to create the perfect caching scenario for your content.



Values:

- Public: Allows the browser and any other network (such as CDNs) to cache the content.
- Private: Restricts caching to ONLY the browser.
- No-cache: The file may be cached, but the browser should still check the server to see if the content has changed.
- No-store: NOTHING should be cached.
- Max-age: The maximum number of seconds a browser should use a cached file before checking the server for new content.

ETag / If-None-Match

Entity tags, or ETags, are basically like a version number that's associated with the file. These are super useful for cache validations, when the browser checks with the server to see if the cached content is out of date or still current. The browser does that by using the If-None-Match header, to compare the ETags to see if the new content matches the old content, or if it should take a new cache.

Expires / Date

The Expires header is very similar to the max-age value of the Cache-Control header, but instead of defining how long a cache is good for, you set the exact date and time that a cache expires. It looks something like this:

```
Fri, 08 July 2016 23:59:59 GMT
```

One important thing to note: If you use an Expires header, it's also a good idea to use a Date header, to make sure the server knows the correct date right now.

If-Modified-Since / Last-Modified

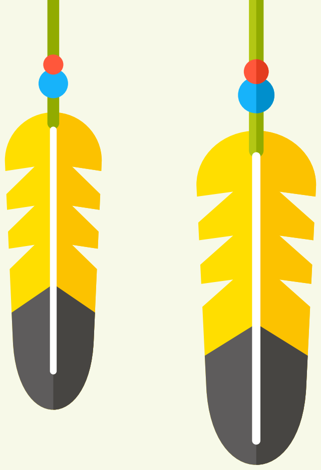
Using If-Modified-Since and Last-Modified headers is an alternative to using ETags and If-None-Match headers. In this case, instead of the ETag, the validation is based upon the date a file was last modified. The browser then uses the If-Modified-Since header to compare the date of the cached content with the date of the server content, and determines if it should download a new copy of content.

Sweet! How about server-side caching?

In true WordPress form, all it takes is a plugin to tell the browser your server-side caching settings: W3 Total Cache. Of course, there are other plugins that would also work, but this is one of the most common options out there. Plugins like this will allow you to control server-side caching on your site, which will help you deliver content to your users super quickly.

One thing to note: while using plugins like this to improve server-side caching can be effective, they can take a lot of configuration and aren't the fastest server-side solution out there. Because WordPress plugins are PHP-based, it still takes time for the server to execute PHP-based caching code.

A much better solution is to use an HTTP accelerator such as Varnish, which sits outside of WordPress, instead of living inside WordPress like W3TotalCache does. Varnish caches pages that have been rendered by WordPress in memory and can serve them up at lightning speeds. A server-side solution like Varnish is often the fastest possible caching system because it serves pages directly from the server's memory, but also because it never has to execute PHP (or any other part of WordPress) in order to work.



If you're hosted on Flywheel, you don't need to worry about caching plugins at all – we automatically take care of server-side caching for you using Varnish. This means your site will be equipped with one of the fastest caching techniques available, and you don't even have to worry about setting any of it up!

Plus, Flywheel servers are configured to automatically flush the cache anytime files change and anytime a post or page is updated. This way, you never have to worry about your users seeing outdated content. You can just focus on building beautiful sites.

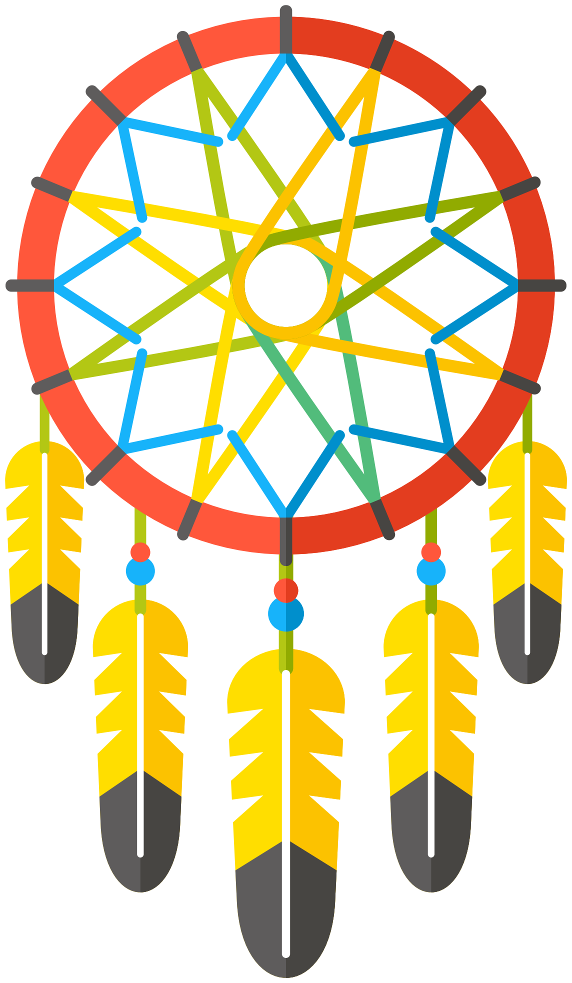
While in most cases, caching is incredibly beneficial to your site, there are a few scenarios where you'll want to pay special attention to what's going on.

The first situation, and probably most common, is whenever you install a new theme or plugin on your WordPress site. Remember those headers that are communicated to the browser? Sometimes theme or plugin developers will set their own that can override your current settings. Instead of increasing performance, that can actually have the opposite effect and cancel out whatever performance boosts you had in place.

The second scenario is whenever you're making updates to your site. While caching is important, I'm guessing it won't constantly be at the front of your mind for the rest of your design career. If something doesn't look right immediately following an update, always remember to refresh the page (and then maybe flush the cache). Then, depending on what content you've added to the site, consider how it affects your users, and if you need to redefine any caching settings.

Luckily, with some help from your trusted plugins, if something gets a little messed up, you can usually get everything back on track pretty quickly. (Or if your site's hosted on Flywheel, our handy-dandy support team is always around to help answer further questions!)

Once you've optimized your site's caching settings, your site's loading times will be super fast. If you want it to be even faster, however, you might want to look into other ways to improve site performance, such as auditing plugins or using a CDN. Hop over to another one of our ebooks, "How to make your site's performance out of this world," to learn how to make your site even faster.



What is Flywheel?

Flywheel is a delightful platform that empowers designers, developers, and digital agencies to focus on what they do best — building beautiful, functional sites for their clients. We make it a breeze to create and develop WordPress sites, handle hosting, manage projects, and ultimately scale your business.

Stop wasting time on server management, security plugins, caching, and all those other boring repetitive tasks that take your focus away from growing your business and jeopardize your relationship with clients. Get Flywheel and get back to doing what you love.

LEARN MORE

FLYWHEEL