Lickity Split (/blog)

Zoompf's Web Performance Blog

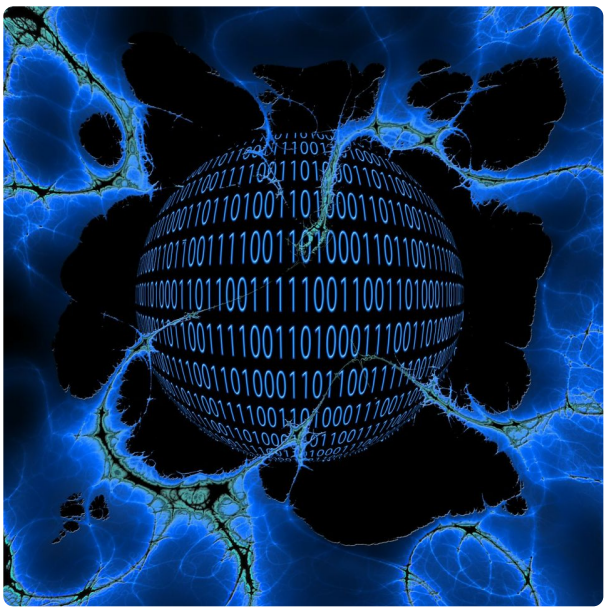
Note: Archived Content

This is the archived version of the Zoompf blog. Since our acquisition by Rigor (/blog/2015/10/zoompf-gets-acquired), all our new research and posts on web performance are being published on The Rigor Blog (http://rigor.com/blog)

Using real life collaboration to explain HTTP Caching

Zoompf Performance (/contact) on June 5, 2015. Category: Optimization (/blog/category/optimization)

Do you still “collaborate” with your team or project team using individual Microsoft Word and Excel files? Let’s hope not, but if you still do, or you remember when you did, then you already have a solid understanding of how HTTP caching – done correctly – can dramatically, and simply improve your website’s performance.



(/wp-content/uploads/2015/05/http_caching.jpg)
Image courtesy of Pixabay.

Here's how we used to collaborate on a document or spreadsheet:

- **Me** (project manager): Do you have the latest spreadsheet for this mornings meeting?
- **You** (team member): Uh, let me see. Mine's dated May 17th. Is that the latest one?
- **Me**: No, it's been updated since then.
- **You**: OK, thanks. Can you email me the latest copy?
- **Me**: Sure, is your email address still you@yourdomain.com (mailto:you@yourdomain.com)?
- **You**: Yes, that's correct.
- **Me**: OK, sending it over now.
- **You**: OK, thanks.
- **Me**: Did you get the file?
- **You**: No, not yet. Are you sure you sent it to the right email address? me@mydomain.com (mailto:me@mydomain.com)?
- **Me**: Yes, that's the address I used.
- **You**: I still don't have it. I wonder what's wrong.
- **Me**: Check your spam folder. Sometimes attachments get marked as spam.
- **You**: Nope, not in spam. Wait, here it comes. OK, I got the email. Now I'm downloading the file. Wow, that file is big, and my connection seems a little flaky at the moment. Says "5 minutes remaining."
- [five minutes later]
- **You**: Yup! There it is. OK, I've got it. It says May 18th. Can I still use that?
- **Me**: Yes!
- **Me**: Great, we can start the meeting now.
- [after the meeting] **You**: Hey, can I have the spreadsheet for the meeting this afternoon?
- **Me**: Yes! You already have it. It's the same one. Just keep using that one!

OMG. Ouch. So painful, but that's how we used to do it, before everything was "in the cloud" (**waves both hands around and speaks in a mystical voice**). Today, it's much different, since we can share access to a Google Doc that everyone can see in their browser, and that's *always* the latest version. The first conversation above simply does not have to happen.

Nor do we have to experience this pain on our websites if we implement proper HTTP caching (http://en.wikipedia.org/wiki/Web_cache). Here's a layman's version of how HTTP caching works, using a similar (but far less painful) conversation:

- **You**: Can I please get a copy of that file?
- **Me**: Yes, here's the file. It's valid until tomorrow at noon.
- **You**: Great, thanks. I'll see you at the morning and afternoon meetings, and I know I can just use the same one.
- **Me**: Correct.

Much better! Now let's apply this conversation to HTTP caching. First, it's the user's (client) browser that does the act of caching, and it's the server that determines (a) if it will allow caching and (b) how that caching will work.

Assuming the server *is* allowing HTTP caching, then the user's browser will store a copies of certain files (like a logo, JavaScript file, or even a PDF) in its cache, so that the browser does not have to re-download those files every single time the user visits that website. Just like the two conversations above, there is still an initial conversation, and then a conversation to determine if the file is up to date. But, with HTTP caching, there's no need to take 5 minutes (obviously an exaggeration, since nobody's going to wait 5 minutes for a web page to load) to download a file.

Instead, the server sends over what we call an `Expires` or `Cache-Control` header, which tells the user's browser "*here's how long that file is valid. You don't need to download it again until after that date & time"*. Then the user's browser can just quickly check the header when it needs that file. If the file is expired, then the browser asks "Is this the latest file version?" If the server's answer is "yes", then the browser still doesn't have to download the file. If the server's answer is "no", then the browser needs to download the file again.

The beautify of caching is that, as long as a file has not expired, the browser never has to check with the server to make sure it is still OK to use. This "conversation" never needs to happen! This means that HTTP caching can eliminate the need to request and download files from your web server, speeding up the user's experience on your website.

And that's what web performance is all about: making the user experience (aka UX) better through faster performance (/blog/2015/06/webperf-is-part-of-ux). However, just "turning on" HTTP caching to cause problems where client use stale content. Ideally, HTTP caching has to be designed fin rom the beginning, and written into the software code. For example, your code could automatically embed time-date stamps in the URL for a resource, allowing you to use HTTP caching without worrying about client using stale content. Performance, including the use of HTTP caching, is at its best when performance is part of the design process.

For an in-depth technical explanation of HTTP caching, Heroku (The Salesforce mobile app development platform) has a solid explanation here (https://devcenter.heroku.com/articles/increasing-application-performance-with-http-cache-headers). I also suggest Mark Notthingham's Caching Tutorial (https://www.mnot.net/cache_docs/).

Are you interested in web performance optimizations like using HTTP caching? You will love our free Zoompf Alerts (/alerts) beta. Zoompf Alerts monitors your site throughout the day, notifying you when performance problems get introduced with your CSS, JavaScript, HTML or Image and more. Make sure nothing changes in a manner that hurts your website performance. It's free and you can opt-out at any time so sign up for Zoompf Alerts beta now (/alerts)

Next Post (/blog/2015/06/what-orange-juice-can-teach-you-about-web-performance)

Earlier Post (/blog/2015/06/webperf-is-part-of-ux)

Comments

Have some thoughts, a comment, or some feedback? Talk to us on Twitter @zoompf (https://twitter.com/zoompf) or use our contact us form (/contact).

Zoompf Becomes Rigor Optimization!

Zoompf's web performance product is now Rigor Optimization. Learn more (http://rigor.com/features).

Get Your Free Report

Get a free detailed performance analysis of your website right now.

Free Performance Report (http://rigor.com/free)

Blog Topics

announcement (/blog/tag/announcement) apache (/blog/tag/apache) asp.net (/blog/tag/asp-net) best practice (/blog/tag/best-practice) bug (/blog/tag/bug) caching (/blog/tag/caching) CDN (/blog/tag/cdn) Chrome (/blog/tag/chrome) compression (/blog/tag/compression) conference (/blog/tag/conference) CSS (/blog/tag/css) DevOps (/blog/tag/devops) Free (/blog/tag/free) Google (/blog/tag/google) How Fast Is? (/blog/tag/how-fast-is) HTML (/blog/tag/html) HTTP (/blog/tag/http) humor (/blog/tag/humor) IE (/blog/tag/ie) images (/blog/tag/images) JavaScript (/blog/tag/javascript) Lose The Wait (/blog/tag/lose-the-wait) Lossless Optimization (/blog/tag/lossless-optimization) Microsoft (/blog/tag/microsoft) performance (/blog/tag/performance) PHP (/blog/tag/php) PNG (/blog/tag/png) Presentations (/blog/tag/presentations) recommendation (/blog/tag/recommendation) report