Core Web Vitals report: 28 Ways to supercharge your site

30-second summary:

- Google plans to roll out the new Core Web Vitals update in early 2021.
- The overall size, dimensions, load order, and format of your images will drastically affect your PageSpeed score.
- Loading critical CSS and JS inline can improve the perceived load time of your site.
- Above-the-fold videos and large background images can be particularly damaging to your Largest Contentful Paint time.
- A server upgrade and a CDN can improve your server response time and your contentful paint score.
- Founder of Content Powered, James Parsons, shares an exhaustive list of 28 elements that will supercharge your site for Google's Core Web Vitals update and Google PageSpeed Insights.

Announced in early 2020, the Core Web Vitals are a set of metrics Google is developing and plans to roll into their overall search algorithm in May of 2021. Given that it's almost 2021 now, anyone who wants to get ahead on optimizing their site for this new algorithm update can get to work now. Thankfully, Google has been very good about publicly disclosing what these new metrics are and how they work.

Armed with that information, it's possible to build a checklist of action items to check and optimize on your site to ready yourself for the inevitable rollout of these new ranking factors. Here are 28 such items for that checklist.

A. Image optimization

Images are one of the largest influencing factors in the core web vitals. All of the web vitals measure the time until some initial rendering, and loading images is the largest source of delay before a page is initially fully loaded. Thus, optimizing images tends to be the most powerful tool for improving core web vitals.

1. Reduce the Dimensions of Background Images

Background images are rarely fully necessary to a site design and can be a large source of delay in loading a page for the first time.

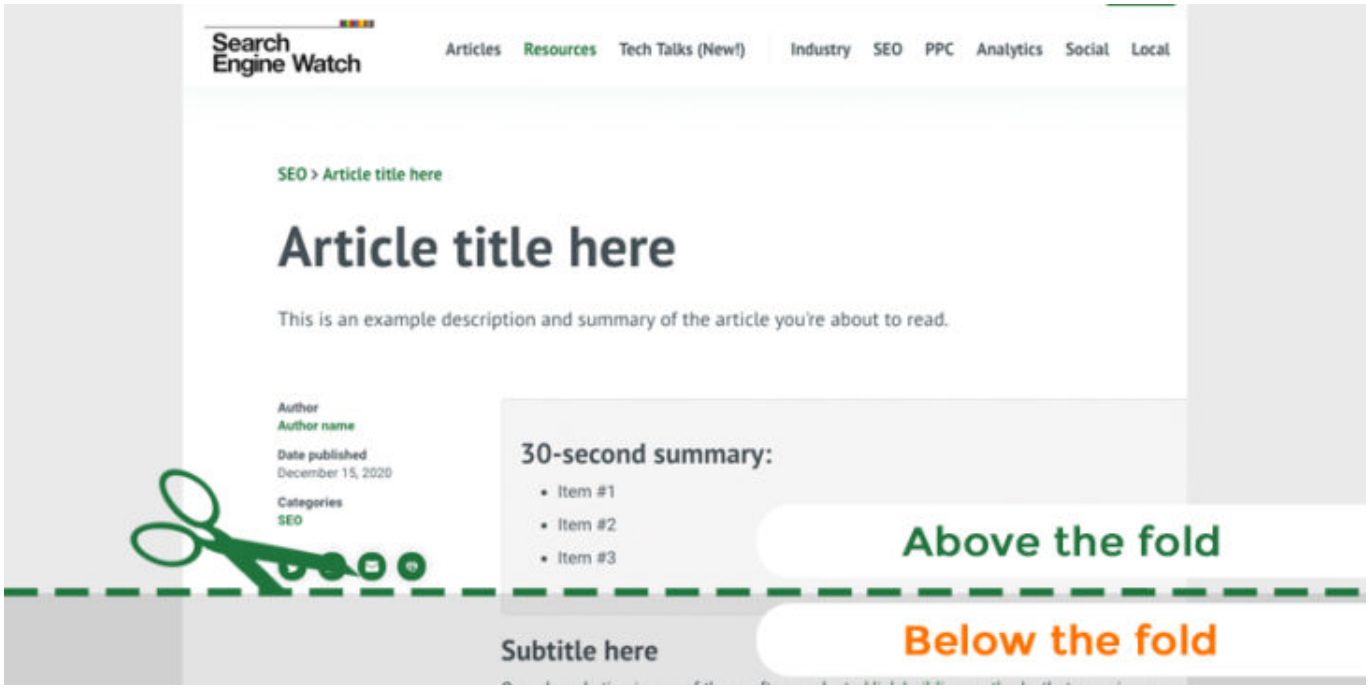
If you use a background image, reduce how large that image is and optimize it so it loads as close to instantaneously as possible.

2. Minimize or Replace Background Images with Patterns

If you’re not tied to a specific background image, either replace the image with flat colors, a gradient, or even a simple tiled pattern. Again, the goal is to minimize how many assets need to load before the initial load of the website is complete. Since background images don’t make a huge impact (and are even less necessary on mobile), minimize or remove them as much as possible.

3. Remove Images on Mobile Above the Fold

Speaking of mobile, the mobile browsing experience is often slower than desktop browsing due to the quality of cell and wireless signals. Mobile devices are especially susceptible to delays in the first input and on the content shift.



To help avoid that, strive to make as much of your above-the-fold content as possible based on text and other simple elements. Large images and slideshows above the fold are particularly rough on your score, so remove or move them as much as possible.

4. Implement Lazy Loading

Lazy loading is a common technique for speeding up the initial load of any given page. With Google's new metrics on the horizon, it's no surprise that support for it is quickly becoming a default feature. WordPress, for example, added native default lazy loading in version 5.5 earlier this year. Make use of lazy loading for any content, particularly images, that doesn't need to load above the fold initially.

5. Use WebP Images

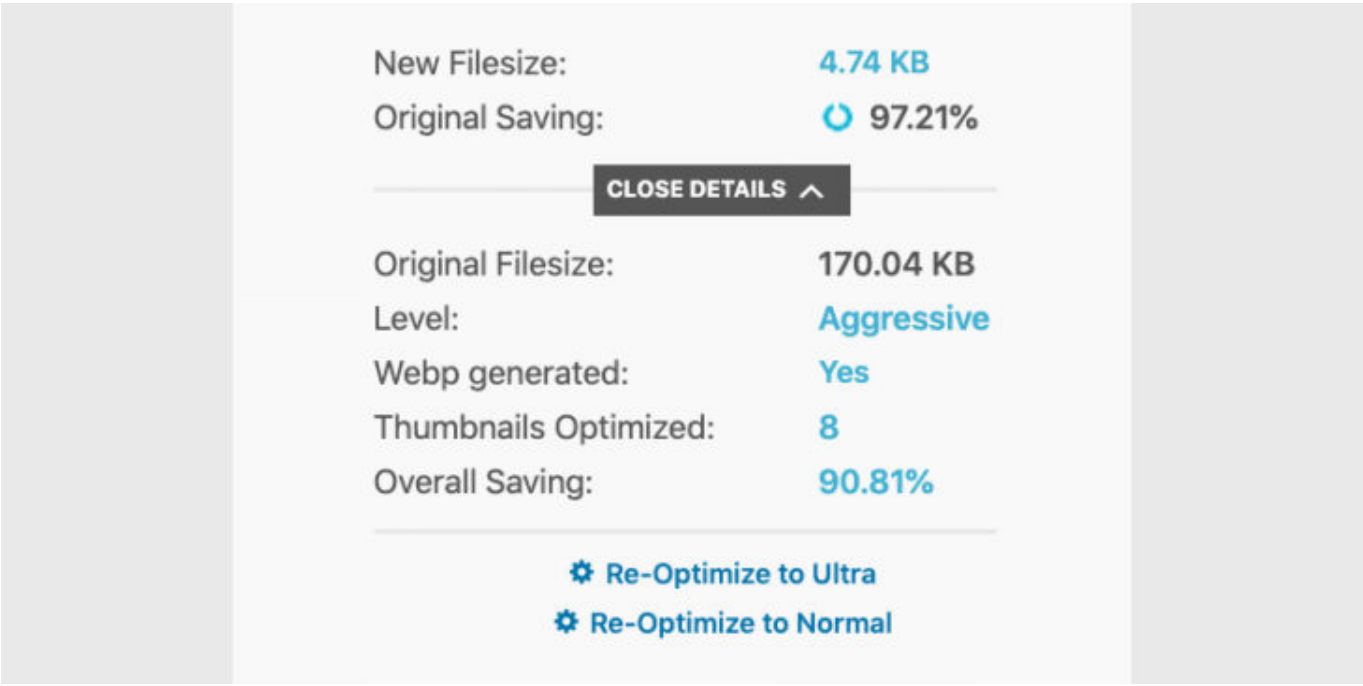
Another Google initiative, WebP is a new image format developed back in 2010. It's a smaller image format with better compression algorithms than your traditional image formats like PNG.

While it hasn't really picked up widespread traction until

recently, it’s becoming more and more valuable as both users and search engines are increasingly concerned with speed and load times. Support is widespread, even if usage isn’t, so you can more-or-less safely use WebP images as your primary image files.

6. Optimize Image File Sizes

Using a tool to crunch or smush image files to be smaller in file size should be a default part of optimizing images for the web by this point.



If you don’t do it already, make sure you implement a way to process images as part of your blogging workflow moving forward. You’ll also want to make sure you’ve defined the height and width of images to prevent layout shift.

B. CSS optimization

CSS has become an increasingly critical part of many site designs, so much so that blocking it makes the web almost unrecognizable. With so much of a site reliant on CSS for everything from colors to positioning, making sure your code is optimized is more important than ever.

7. Inline Critical CSS

You don’t need to inline every bit of your CSS, though that works as well. In particular, you want to inline CSS that is critical to the overall design and layout of your theme.



This minimizes the number of individual files a browser needs to call from your server just to load the initial layout and paint the initial content on your site.

8. Minify CSS

CSS is by default a very minimalist language and can operate perfectly well without spaces, indentation, comments, and other text that makes it more user-friendly and easier to develop. Before uploading new code to your site, run it through a tool to minify it and remove all of that excess cruft that has a microscopic-yet-tangible effect on page loading.

9. Consolidate CSS Files and Code

It can be tempting to store CSS in a variety of files and scattered throughout your code, placing it where it seems like it should be rather than where it makes sense to put it. Remember; what is easiest as a developer is not necessarily the fastest for a user. Consolidate your CSS, whether it's inline or in separate files, and only execute specific elements as necessary.

10. Optimize CSS Delivery

CSS is often a late-loading element of site code. Traditional site design loads the framework for the site, then the content, then the CSS to format it all. Particularly when CSS is stored in an external file, this delays loading significantly. Preloading your CSS is a strategy recommended by Google to force the browser to load the CSS and have it ready when it’s needed.

C. JavaScript optimization

JavaScript is one of the biggest sources of code bloat and delay in loading websites. Optimizing your site’s JS can help speed it up tremendously, even when it doesn’t seem like it would have much of an effect based on what you’re doing to it.

11. Minify JS Scripts

Like CSS, JavaScript doesn’t need extraneous spaces and breaks to function. It also doesn’t need verbose variable names, which are useful for development but can increase the size of scripts by a significant amount.

Run your scripts through a minifier before adding them to your site.



12. Consolidate Scripts and Minimize Usage

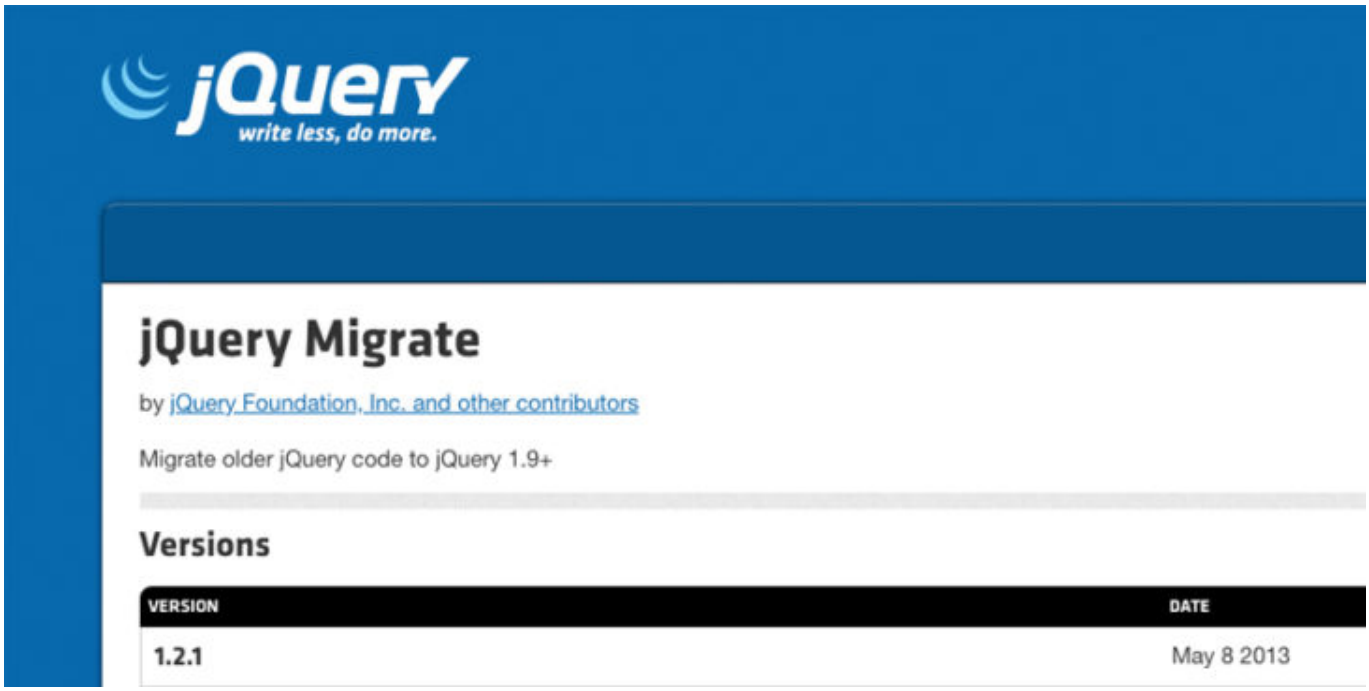
Many of the purposes web designers use JavaScript for have been available as features in HTML5 and CSS3 for years now. Particularly in older websites, a revamp or review of scripts can find alternative, faster ways to do the same things. Review and optimize, minimize, consolidate, and strip as much JavaScript as you can from your site.

13. Defer or Async Scripts Whenever Possible

Scripts are roadblocks in rendering a website. When a browser has to render a JS script, it has to process through that script before it can continue loading the page. Since many developers put scripts in their headers, this delays page loading significantly. Using Defer allows the browser to continue loading the page before executing the script, while Async allows them to load simultaneously. Using these two features allows you to offset the delay inherent in using scripts and speed up your initial page loads.

14. Remove jQuery Migrate

A recent update to jQuery has led to a lot of old plugins and scripts no longer working. To buy time and allow webmasters to update their sites, the Migrate module was introduced. This is essentially a translation module that allows old jQuery to function on sites that utilize a newer version of jQuery.



Perform an audit of your site to see if anything you're using – particularly old plugins and apps – uses jQuery Migrate. If so, consider updating or replace those plugins. Your goal is to remove usage of the Migrate module entirely because it's rather bulky and can slow down websites dramatically.

15. Use Google Hosted JS Whenever Possible

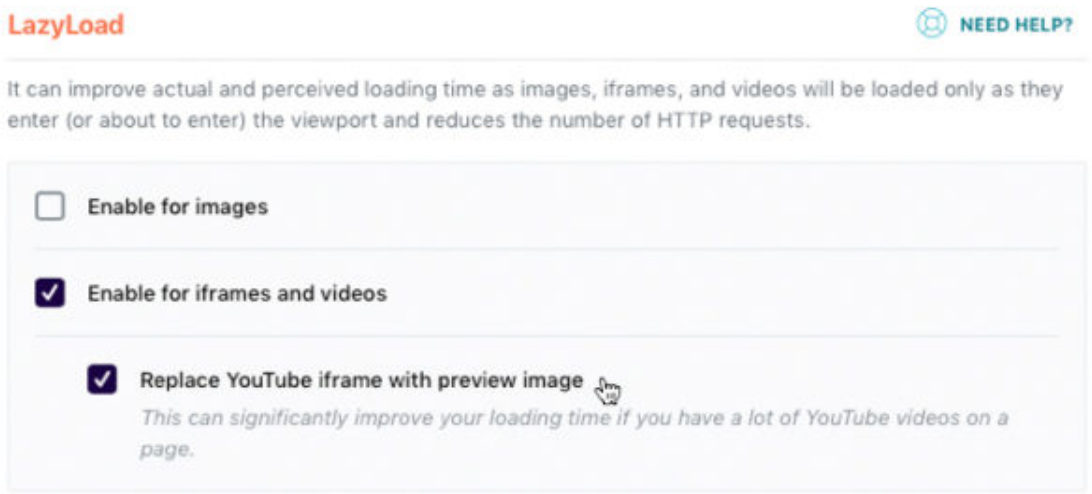
Google offers a range of standard libraries hosted on their servers for use on your website. Rather than relying on a third party for those libraries or hosting them yourself, use Google's versions for the fastest possible load times.

D. Video optimization

Videos are increasingly popular as part of the average website, from core elements of content to video-based advertising and everything in between. They’re also extremely large files, even with partial loading and modern video buffering. Optimize your use of video as much as possible.

16. Use Image Placeholders for Video Thumbnails

There are plenty of users who browse the web with no desire to watch videos, so forcing videos to load in the background for them is completely unnecessary. A good workaround is to use an image placeholder where the video would normally load.



The image loads faster and looks like the video player with a loaded thumbnail. When a user clicks it to start the video, it begins the video load but doesn’t require loading any of the video file or player until that point.

17. Minimize Videos Above the Fold

As with images, video files are extremely heavy, so loading them above the fold is a guaranteed delay on your first content paint. Push them below the fold; most people want to read a title and introduction before they get to the video anyway.

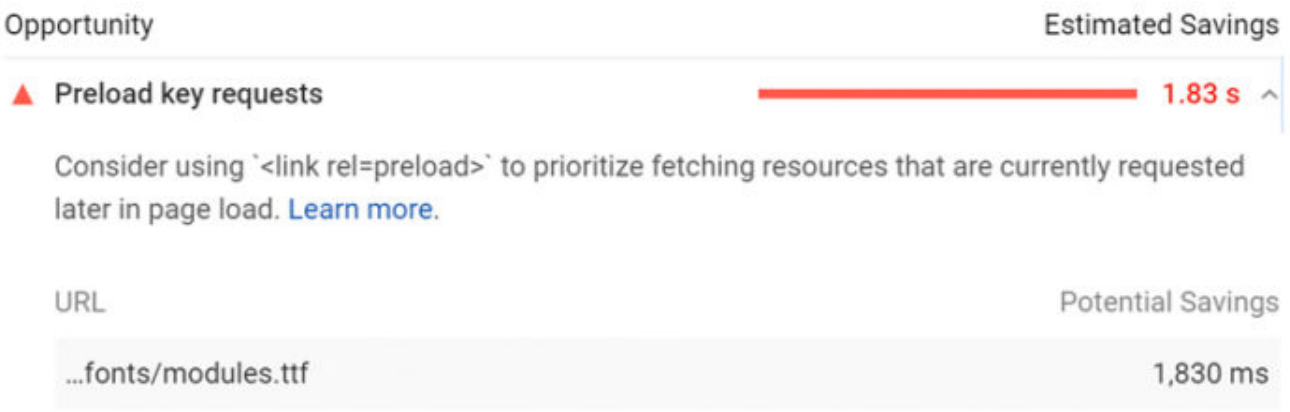
E. Font and icon optimization

Fonts and icon usage can be a lot heavier on a site’s load times than you might expect. Optimizing them might seem like minuscule detail work, but when you see the impact it can have, you’ll wonder why you never made these minor-yet-impactful optimizations before.

18. Preload Fonts

Similar to scripts, when your website calls for a font that it needs to load, loading that font takes precedence and stops the rest of the code from rendering.

Using a preload command to load the font earlier than necessary helps speed up page loading, as well as preventing the “flash of unstyled text” effect that happens for a brief instant between the text loading and the font styling appearing.



19. Only Use Fonts You Need

Many web fonts and font families load their entire character sets and stylesheets when called, even if your page doesn't utilize 90% of that content. Often, you can limit how much you load, though you may need to pay for premium font access. It can be quite worthwhile if you're using limited amounts of a given font, or a font that has a particularly large character set included.

20. Use SVG Whenever Possible

SVGs are Scalable Vector Graphics and are a way to create extremely small elements of a page that can nevertheless scale indefinitely, as well as be manipulated individually, to a much greater degree than traditional fonts and icons. If possible, switch to using SVGs instead of your usual icons.

F. Server optimization

No matter how many optimizations you make to the code of your website, to your images, or to other elements of your site, none of it matters if your server is slow. The proliferation

of web hosting companies, the ongoing development of faster and stronger tech, all means that web hosting shows its age very quickly. Every few years, it can be worthwhile to change or upgrade hosting to faster infrastructure.

21. Upgrade to a Faster Server

You don’t necessarily need to upgrade from a shared host to a dedicated host, though this can help with some of the speed issues inherent in shared hosting. Even simply upgrading from a slower package to a faster one can be a good use of a budget.

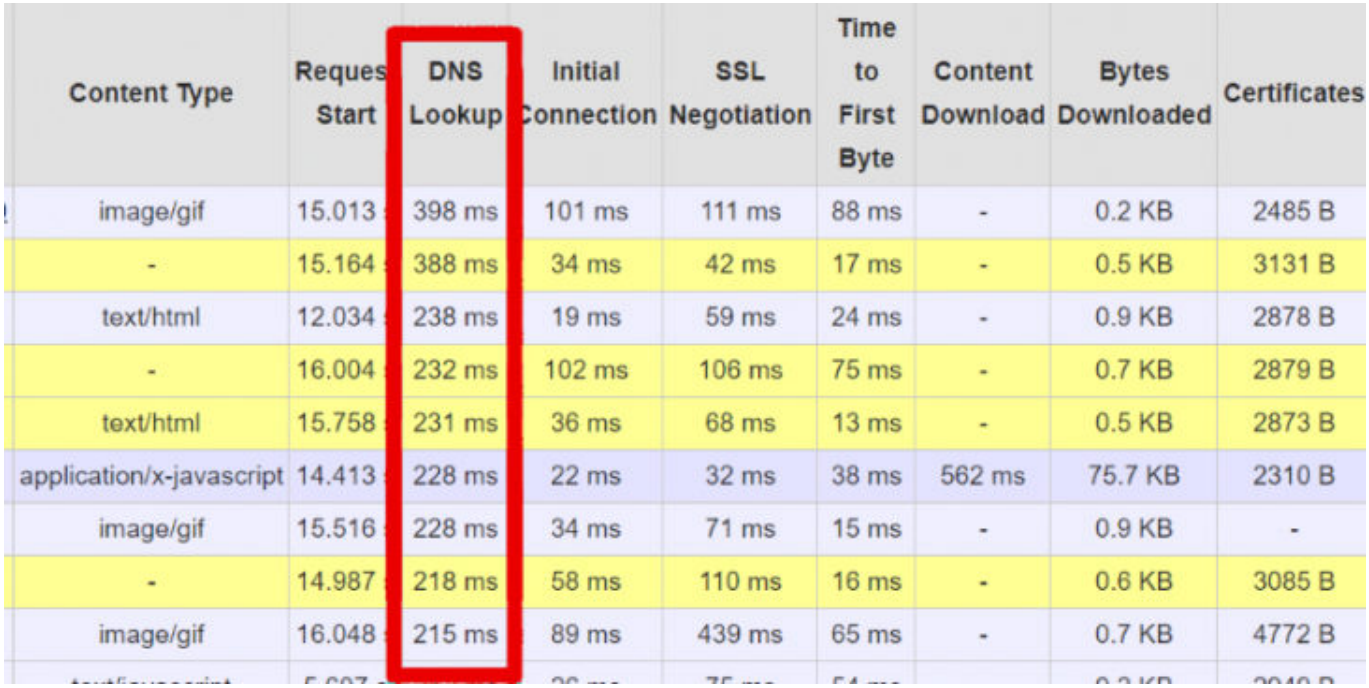
22. Use a CDN

Modern content delivery networks can handle most of the elements of your site faster than your typical web host can in almost every circumstance. At a minimum, consider using a CDN for your images, videos, and other multimedia. You can also consider offloading stand-alone script files as well.

23. Preload DNS Queries

Preloading or prefetching DNS queries helps minimize the delay between an asset being requested by the visitor and the display of that asset.

This couples with using a CDN to store assets by loading and resolving the CDN’s domain before it’s called for the first time, further speeding up page load times.



Content Type	Reques Start	DNS Lookup	Initial Connection	SSL Negotiation	Time to First Byte	Content Download	Bytes Downloaded	Certificates
image/gif	15.013	398 ms	101 ms	111 ms	88 ms	-	0.2 KB	2485 B
-	15.164	388 ms	34 ms	42 ms	17 ms	-	0.5 KB	3131 B
text/html	12.034	238 ms	19 ms	59 ms	24 ms	-	0.9 KB	2878 B
-	16.004	232 ms	102 ms	106 ms	75 ms	-	0.7 KB	2879 B
text/html	15.758	231 ms	36 ms	68 ms	13 ms	-	0.5 KB	2873 B
application/x-javascript	14.413	228 ms	22 ms	32 ms	38 ms	562 ms	75.7 KB	2310 B
image/gif	15.516	228 ms	34 ms	71 ms	15 ms	-	0.9 KB	-
-	14.987	218 ms	58 ms	110 ms	16 ms	-	0.6 KB	3085 B
image/gif	16.048	215 ms	89 ms	439 ms	65 ms	-	0.7 KB	4772 B

24. Preload Your Cache

Often, a cache plugin or script used on a website triggers when the first visitor arrives to view the page. That first visitor has a slower experience, but their loads cache the page for future visitors until the cache expires. Unfortunately, the first visit is often a Google bot crawling your page from your XML sitemap or an internal link, and that means that Google is the first one to experience the slow version of your site. You can get around this by preloading the cache on your website so Google's next visit is a guaranteed fast-loading web page.

25. Consider a Server-Side Cache

Software such as Varnish Cache acts as a server-side cache to further speed up the generation and serving of a cached version of your page, making it as fast as possible with as few server calls as possible.

G. Additional optimization

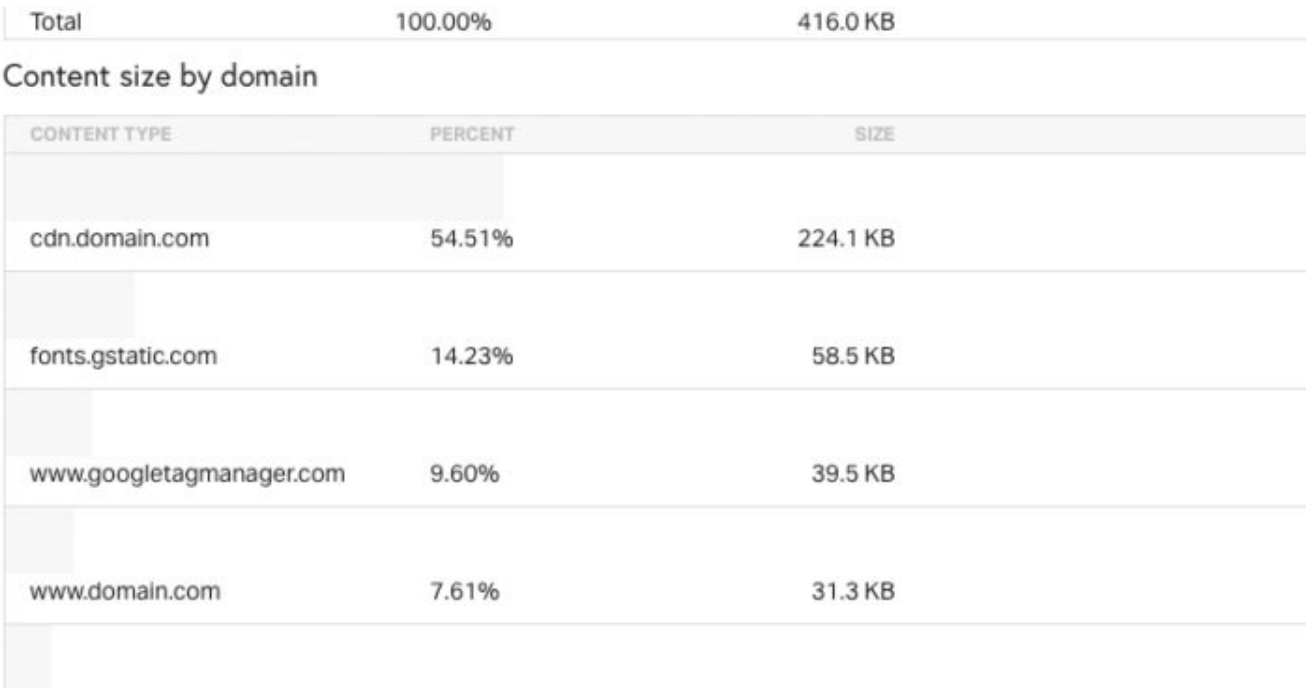
Anything that didn't fit in another category has been added

here. These additional optimizations might not apply to your site design, but if they do, taking care of them can be a great boon.

26. Minimize Third-Party Scripts

Webmasters in 2021 will need to strike a balance between site speed optimizations and user engagement tools.

Many plugins, such as social sharing buttons, third-party comment systems, and media embeds all need to execute third-party scripts in order to work, but those scripts slow down the site. Minimize them as much as possible, and try to find the fastest versions of each.



Total	100.00%	416.0 KB

Content size by domain

CONTENT TYPE	PERCENT	SIZE
cdn.domain.com	54.51%	224.1 KB
fonts.gstatic.com	14.23%	58.5 KB
www.googletagmanager.com	9.60%	39.5 KB
www.domain.com	7.61%	31.3 KB

27. Avoid Pre-Load Filler

A common technique for sites with slower load times is to add a spinner, a loading icon, an animation, or another form of content that loads and displays to indicate to a user that the site is, in fact, loading. While this can help minimize bounces, it’s a huge hit to the initial loads measured by the core web vitals. Remove these and work to speed up your site such that you don’t need them.

28. Consider a Site Redesign

When all is said and done, sometimes you need to make so many changes to so many foundational elements of your site that it’s easier to simply scrap your current design and engineer a new one with speed in mind. Consider it a possibility, and analyze the benefits you’ll get from optimized core web vitals. No one knows yet how influential those metrics will be on the overall algorithm, but it certainly can’t hurt to optimize for them.

James Parsons is the founder of Content Powered, a blog management & content marketing company. He’s worked as a senior-level content marketer for over a decade and writes for Forbes, Entrepreneur, Inc, and Business Insider.

More about:

Core Web Vitals report: 28 Ways to supercharge your site