Aaron T. Grogg (https://aarontgrogg.com/)

Thoughts, Findings, and Rants…

Home (https://aarontgrogg.com/)

Projects (https://aarontgrogg.com/projects/)

About (https://aarontgrogg.com/about/)

Contact (https://aarontgrogg.com/contact/)

Search

Web Performance Audits: What's in your Checklist?

(https://bsky.app/intent/compose?
text=https%3A%2F%2Faarontgrogg.com%2Fblog%2F2023%2F06%2F09%2Fperformance-
audits-whats-in-your-checklist%2F)

(http
mini=true&url=https%3A%2F%2Faa

checklist%2F&title=Web+Perform

June (https://aarontgrogg.com/blog/2023/06/) 9 (https://aarontgrogg.com/blog/2023/06/09/), 2023 (https://aarontgrogg.com/blog/2023/) by Aaron T. Grogg (https://aarontgrogg.com/)

Performance audits are an invaluable step in fine-tuning a site's performance. But man, there is a lot to look at… Which makes it really easy to miss something (or some*things*) on any given audit…

So, to help forgetful people like me, below is a collection of items that I typically check during an audit.

This list is intended to be organic, so please share any additions, alterations or deletions that you recommend.

Item	Description	References
Request Headers	Incorrrect HTTP headers can harm page caching, use wrong protocols, and even hurt security.	• https://www.corewebvitals.io/tools/pagespeed-header-analyzer (https://www.corewebvitals.io/tools/pagespeed-header-analyzer)
Redirects	`301` or `302` redirects incur unnecessary latency. Try to remove any you encounter.	• https://www.matthewedgar.net/do-redirects-add-to-website-speed/ (https://www.matthewedgar.net/do-redirects-add-to-website-speed/)

Item	Description	References
`4xx` errors	`4xx` errors incur latency and block requests while the browser waits for the URL to resolve. There should be no `4xx` errors. This includes links to other pages.	• https://developers.google.com/search/blog/2011/05/do-404s-hurt-my-site (https://developers.google.com/search/blog/2011/05/do-404s-hurt-my-site)
Viewport `meta`	Make sure "viewport" `meta` is `<meta name="viewport" content="width=device-width, initial-scale=1" />`. This tells the browser to resize to the current screen width and allows the user to pinch to zoom in and out. Definitely no "maximum-scale=1" or "user-scalable=no".	• https://developer.mozilla.org/en-US/docs/Web/HTML/Viewport_meta_tag (https://developer.mozilla.org/en-US/docs/Web/HTML/Viewport_meta_tag)
`preconnect` links	There should be `preconnect` links for any important domains that will be needed later, such as a CDN, Cookie Consent assets, etc. This will help these assets fetch faster.	• https://www.igvita.com/2015/08/17/eliminating-roundtrips-with-preconnect/ (https://www.igvita.com/2015/08/17/eliminating-roundtrips-with-preconnect/)
`preload` links	There should be `preload` links for all above-the-fold fonts. Also add one for the main hero asset (image, video, etc.) if it is hard for the browser to find in the HTML. This will help these assets fetch faster.	• https://3perf.com/blog/link-rels/ (https://3perf.com/blog/link-rels/)
Font `preload crossorigin`	If preloading any font, it should have a `crossorigin` attribute set to `"anonymous"`.	• https://web.dev/codelab-preload-web-fonts/ (https://web.dev/codelab-preload-web-fonts/)
Validate Resource Hints	Once you think all the preconnect and preload hints are setup, validate to make sure.	• https://www.debugbear.com/resource-hint-validator (https://www.debugbear.com/resource-hint-validator)
`fetchpriority` for hero asset	The hero asset should have `fetchpriority="high"`. This could be on the asset's `preload` link if it is hard for the browser to find in the HTML.	• https://web.dev/fetch-priority/ (https://web.dev/fetch-priority/)

Item	Description	References
Mind Critical Rendering Path	Only resources that are critical for the initial page load should be allowed to block the critical rendering path.	- https://developer.mozilla.org/en-US/docs/Web/Performance/Critical_rendering_path (https://developer.mozilla.org/en-US/docs/Web/Performance/Critical_rendering_path) - https://addyosmani.com/blog/script-priorities/ (https://addyosmani.com/blog/script-priorities/) - https://github.com/addyosmani/critical (https://github.com/addyosmani/critical)
JS Aggregation	The essence of aggregation is to group the JavaScript and CSS files in bundles in order to reduce the amount of HTTP requests necessary for the page to load. Instead of a couple dozens files, a page will load with just a handful of aggregates. This *can* lead to significant website speed gains if done properly, but only assets that are needed for each page should be loaded by that page. - Reduce HTTP requests and minimize CSS and JavaScript - Load visible content first and defer everything else - Only load necessary JavaScript and CSS files that are helpful, especially for mobile users - Too much JavaScript or CSS cause the pages to slow down - Strike a balance between making fewer HTTP requests and leveraging browser caching	- https://www.lullabot.com/articles/javascript-aggregation-in-drupal-7 (https://www.lullabot.com/articles/javascript-aggregation-in-drupal-7) - https://www.metaltoad.com/blog/drupal-7-taking-control-css-and-js-aggregation (https://www.metaltoad.com/blog/drupal-7-taking-control-css-and-js-aggregation)
Minify / Compress	Minify / compress your code where possible and consider reorganizing your CSS altogether.	https://github.com/kangax/html-minifier (https://github.com/kangax/html-minifier)
Check cache controls	Make sure assets that don't undergo frequent changes (logos, icons, etc.) are being cached by the browser so that the visitors don't have to re-fetch them on every page load.	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control)

Item	Description	References
Asset priority	Any CSS or JS that is not needed for the initial page view should be delayed (`defer`, `async`, lazy load, etc.), allowing for critical assets to load first.	• [https://addyosmani.com/blog/script-priorities/ (https://addyosmani.com/blog/script-priorities/)](https://addyosmani.com/blog/script-priorities/) • [Scheduling Scripts Intuitively and Performantly (https://bit.ly/script-scheduling)](https://bit.ly/script-scheduling) • [Chrome Resource Priorities and Scheduling (https://docs.google.com/document/d/1bCDuq9H1ih9iNjgzyAL0gpwNFiEP4TZS-YLRp_RuMlc/)](https://docs.google.com/document/d/1bCDuq9H1ih9iNjgzyAL0gpwNFiEP4TZS-YLRp_RuMlc/) • [Preload, Prefetch And Priorities in Chrome (https://medium.com/reloading/preload-prefetch-and-priorities-in-chrome-776165961bbf)](https://medium.com/reloading/preload-prefetch-and-priorities-in-chrome-776165961bbf) • [Priorities for rel=preload in Chrome and WebKit (https://twitter.com/yoavweiss/status/1096075414697639936)](https://twitter.com/yoavweiss/status/1096075414697639936)
WOFF2 priority	Make sure WOFF2 is the first font format in your CSS font-family declaration (before WOFF).	• [https://medium.com/geekculture/one-font-format-to-rule-them-all-woff2-d053e2e5165a (https://medium.com/geekculture/one-font-format-to-rule-them-all-woff2-d053e2e5165a)](https://medium.com/geekculture/one-font-format-to-rule-them-all-woff2-d053e2e5165a)
CSS `@import`	There should be no `@import` statements in any CSS.	• [https://www.stevesouders.com/blog/2009/04/09/dont-use-import/ (https://www.stevesouders.com/blog/2009/04/09/dont-use-import/)](https://www.stevesouders.com/blog/2009/04/09/dont-use-import/)
Background-image	There should be minimum of cases where images served via CSS `background-image`.	• [https://www.successlocal.co.uk/how-do-background-images-affect-your-website (https://www.successlocal.co.uk/how-do-background-images-affect-your-website)](https://www.successlocal.co.uk/how-do-background-images-affect-your-website) • [https://web.dev/optimize-lcp/ (https://web.dev/optimize-lcp/)](https://web.dev/optimize-lcp/)

Item	Description	References
Images on Mobile	Don't make mobile users download images designed for larger screens. Make sure mobile has reduced size images & videos. And if using a hero asset `preload`, make sure you also have one for the mobile-sized asset.	• https://web.dev/preload-respon-sive-images/ (https://web.dev/preload-respon-sive-images/)
Animated GIFs	Any animated GIFs should be converted to MP4 videos, then compressed for the web.	• https://www.smashingmagazine.com/2018/11/gif-to-video/ (https://www.smashingmagazine.com/2018/11/gif-to-video/) • https://ezgif.com/gif-to-mp4 (https://ezgif.com/gif-to-mp4) • https://www.videosmaller.com/ (https://www.videosmaller.com/)
JPG/PNG > WEBP/AVIF	Any JPG, PNG or GIF images should be served to the browser as WEBP or AVIF, if possible. This can often be handled auto-matically for images served via a CDN.	• https://web.dev/optimize-lcp/ (https://web.dev/optimize-lcp/)
Video compression	Check whether videos are compressed and whether they need their audio track.	• https://www.movavi.com/videoconverter/ (https://www.movavi.com/videoconverter/) • https://www.freeconvert.com/video-compressor (https://www.freeconvert.com/video-compressor) • https://www.videosmaller.com/ (https://www.videosmaller.com/)
Video `preload`	The `preload` attribute should be set to `"none"` for any video that is below-the-fold. If the video is above-the-fold, the `preload` should be set to `"auto"`.	• https://web.dev/lazy-loading-video/ (https://web.dev/lazy-load-ing-video/)
Video `autoplay`	The `autoplay` attribute should be set to `"no"`, unless it is the hero asset. For below-the-fold videos, the `autoplay` attribute should be set to `"no"`.	• https://web.dev/lazy-loading-video/ (https://web.dev/lazy-load-ing-video/)
Video `poster`	Ideally video elements should have a `poster` attribute that can display while the video is downloading. A video `poster` image can qualify as an LCP (https://web.dev/lcp/#what-elements-are-considered).	• https://web.dev/lazy-loading-video/ (https://web.dev/lazy-load-ing-video/)

Item	Description	References
Lazy load settings	- Any asset that is initially **on**screen should **not** be lazy loaded - Any asset that is initially **off**screen **should** be lazy loaded - Any asset that **should** be lazy loaded should have the `loading="lazy"` attribute and value added to it	- https://addyosmani.com/blog/lazy-loading/ (https://addyosmani.com/blog/lazy-loading/) - https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img#attr-loading (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img#attr-loading)
Text in images	Ideally all text would be rendered as HTML, not within images or videos.	https://developers.google.com/search/docs/advanced/guidelines/google-images (https://developers.google.com/search/docs/advanced/guidelines/google-images)
Text readability	Ideally all text should be readable at all times. This includes anything that renders over images or videos, both while the asset is still loading (white text should have some kind of background color), and after the asset has loaded (make sure text does not overlap a color that makes it unreadable).	https://w3c.github.io/low-vision-a11y-tf/requirements.html#text-contrast (https://w3c.github.io/low-vision-a11y-tf/requirements.html#text-contrast)
Image `alt` attributes	All `img` elements should have an `alt` attribute that describes the **content** of the image.	https://ahrefs.com/blog/alt-text/ (https://ahrefs.com/blog/alt-text/)
In-page CSS	Typically CSS should be served via an external file, so it can be cached and re-used. But there are cases where in-page (or embedded) CSS makes sense. In these cases, check the size of the code and see if it is possible to reduce the code by minifying or removing unused code.	https://web.dev/remove-unused-code/ (https://web.dev/remove-unused-code/)
In-page JS	Typically JS should be served via an external file, so it can be cached and re-used. But there are cases where in-page (or embedded) JS makes sense. In these cases, check the size of the code and see if it is possible to reduce the code by minifying or removing unused code.	https://web.dev/remove-unused-code/ (https://web.dev/remove-unused-code/)
In-page JSON	Typically JSON should be served via an external file, so it can be cached and re-used. But there are cases where in-page (or embedded) JSON makes sense. In these cases, check the size of the code and see if it is possible to reduce the code by minifying or removing unused code.	https://web.dev/remove-unused-code/ (https://web.dev/remove-unused-code/)

Item	Description	References
Third-party scripts	Check the purpose of each third-party script to make sure it is still needed. If so, see if the files can be delayed or optimized.	- https://web.dev/efficiently-load-third-party-javascript/#use-async-or-defer (https://web.dev/efficiently-load-third-party-javascript/#use-async-or-defer)
KPIs	Check that the page is within any Performance KPIs thresholds you have previously established.	- https://web.dev/vitals/ (https://web.dev/vitals/)
Performance Budget	Check that all page assets are within the any Performance Budget thresholds you have previously established.	- https://web.dev/performance-budgets-101/ (https://web.dev/performance-budgets-101/) - https://developer.mozilla.org/en-US/docs/Web/Performance/Performance_budgets (https://developer.mozilla.org/en-US/docs/Web/Performance/Performance_budgets) - https://addyosmani.com/blog/performance-budgets/ (https://addyosmani.com/blog/performance-budgets/) - https://timkadlec.com/remembers/2019-03-07-performance-budgets-that-stick/ (https://timkadlec.com/remembers/2019-03-07-performance-budgets-that-stick/)
Page validation	Page errors can also tax the browser as it tries to build a page and has to make corrections.	- https://validator.w3.org/ (https://validator.w3.org/)
Accessibility validation	Make pages accessible to **everyone**. 15% of the world's population has some sort of disability (WHO (https://apps.who.int/iris/handle/10665/44575)). 90% of websites are inaccessible to people with disabilities who use assistive technologies. Shopping sites average 72.9 errors per page. Between 2017 and 2021, the number of ADA-related lawsuits and legal actions increased by 400% (DDIY (https://ddiy.co/web-accessibility-statistics/)).	- https://wave.webaim.org/ (https://wave.webaim.org/)

Happy auditing,
Atg

This entry was posted in accessibility (https://aarontgrogg.com/blog/category/accessibility/), aria (https://aarontgrogg.com/blog/category/aria/), best practices (https://aarontgrogg.com/blog/category/best-practices/), performance (https://aarontgrogg.com/blog/category/performance/), testing (https://aarontgrogg.com/blog/category/testing/), web development (https://aarontgrogg.com/blog/category/web-development/). Bookmark the permalink (https://aarontgrogg.com/blog/2023/06/09/performance-audits-whats-in-your-checklist/).

2 Responses to *Web Performance Audits: What's in your Checklist?*

1. Aaron T. Grogg (https://aarontgrogg.com/) says:

   June 9, 2023 at 7:35 pm (https://aarontgrogg.com/blog/2023/06/09/performance-audits-whats-in-your-checklist/#comment-300416)

   Vitaly Friedman of Smashing Magazine fame compiled an awesome list too: https://paper.dropbox.com/doc/Performance-Optimization-Strategy-in-2023-qWcr7orx2cEWHpLqoLeTC (https://paper.dropbox.com/doc/Performance-Optimization-Strategy-in-2023-qWcr7orx2cEWHpLqoLeTC) Gonna need to pilfer… ;-)

   Reply (https://aarontgrogg.com/blog/2023/06/09/performance-audits-whats-in-your-checklist/?replytocom=300416#respond)

2. Aaron T. Grogg (https://aarontgrogg.com/) says:

   September 22, 2025 at 4:53 pm (https://aarontgrogg.com/blog/2023/06/09/performance-audits-whats-in-your-checklist/#comment-323349)

   Added new item for HTTP Headers, including a link to Arjen Karel's Analyzer (https://www.corewebvitals.io/tools/pagespeed-header-analyzer)!

   Reply (https://aarontgrogg.com/blog/2023/06/09/performance-audits-whats-in-your-checklist/?replytocom=323349#respond)

Leave a Reply

Your email address will not be published. Required fields are marked *

> Your Comment

Name *

> Your Name

Email *

> contact@example.com

Website

> http://example.com/

Save my name, email, and website in this browser for the next time I comment.

Notify me of follow-up comments by email.

Notify me of new posts by email.

Post Comment

To respond on your own website, enter the URL of your response which should contain a link to this post's permalink URL. Your response will then appear (possibly after moderation) on this page. Want to update or remove your response? Update or delete your post and re-enter your post's URL again. (Find out more about Webmentions. (https://indieweb.org/webmention))

URL/Permalink of your article

Ping me!

Contact

(https://bsky.app/profile/aarontgrogg.com)(https://www.linkedin.com/in/aarontgrogg)(https://github.com/aarontgrogg)(https://aarontg

Top Posts

Getting to Know… WPO (Web Performance Optimization) (/blog/2022/01/24/getting-to-know-wpo-web-performance-optimization/)

Web Performance Audits: What's in your Checklist? (/blog/2023/06/09/performance-audits-whats-in-your-checklist/)

Replace JS with no-JS (or lo-JS) Options (/blog/2023/05/31/replace-js-with-no-js-or-lo-js-options/)

Replacing jQuery with Vanilla JS / ES6 (/blog/2021/09/29/replacing-jquery-with-vanilla-es6/)

How to replace remote files with local files when debugging (/blog/2015/03/24/how-to-replace-remote-files-with-local-files-when-debugging/)

The difference between Service Workers, Web Workers and WebSockets (/blog/2015/07/20/the-difference-between-service-workers-web-workers-and-websockets/)

Series Part 1: A beginner's guide to comparing, and getting started with, MVC frameworks: Intro (/blog/2015/11/02/series-part-1-a-beginners-guide-to-comparing-and-getting-started-with-mvc-frameworks-intro/)

Meta

Categories (/categories/)

Archives (/archives/)

Posts RSS (/feed/)

Comments RSS (/comments/feed/)

WordPress.org (https://wordpress.org/)

Aaron T. Grogg (https://aarontgrogg.com/)