

A COMPLETE GUIDE

CORE WEB VITALS



PUBLISHED BY **Search Engine Journal**®

Boost your SEO strategy with interactive content experiences

84% of consumers expect brands to produce content that entertains, provides solutions and produces experiences. Is your company meeting their expectations?

Your audience demands premium content experiences.
Rock Content helps you deliver.



More engagement, more traffic

Users will engage and spend more time on your page, attracting backlinks



Get valuable data from your prospects

Interactive content provides you with valuable data so you can make improvements on your SEO On Page



Improve your mobile experience

Improve your mobile experience and you can get valuable points from the search engines



Increase your conversion rate

Interactive content converts 2x more than static content

2000+ brands have already selected Rock Content

VISA

Johnson's

KAYAK

Genentech

The Economist

LinkedIn

HIRED

THE HUFFINGTON POST

CVS Health

castlight

citi



verizon

CISCO

salesforce

Ford

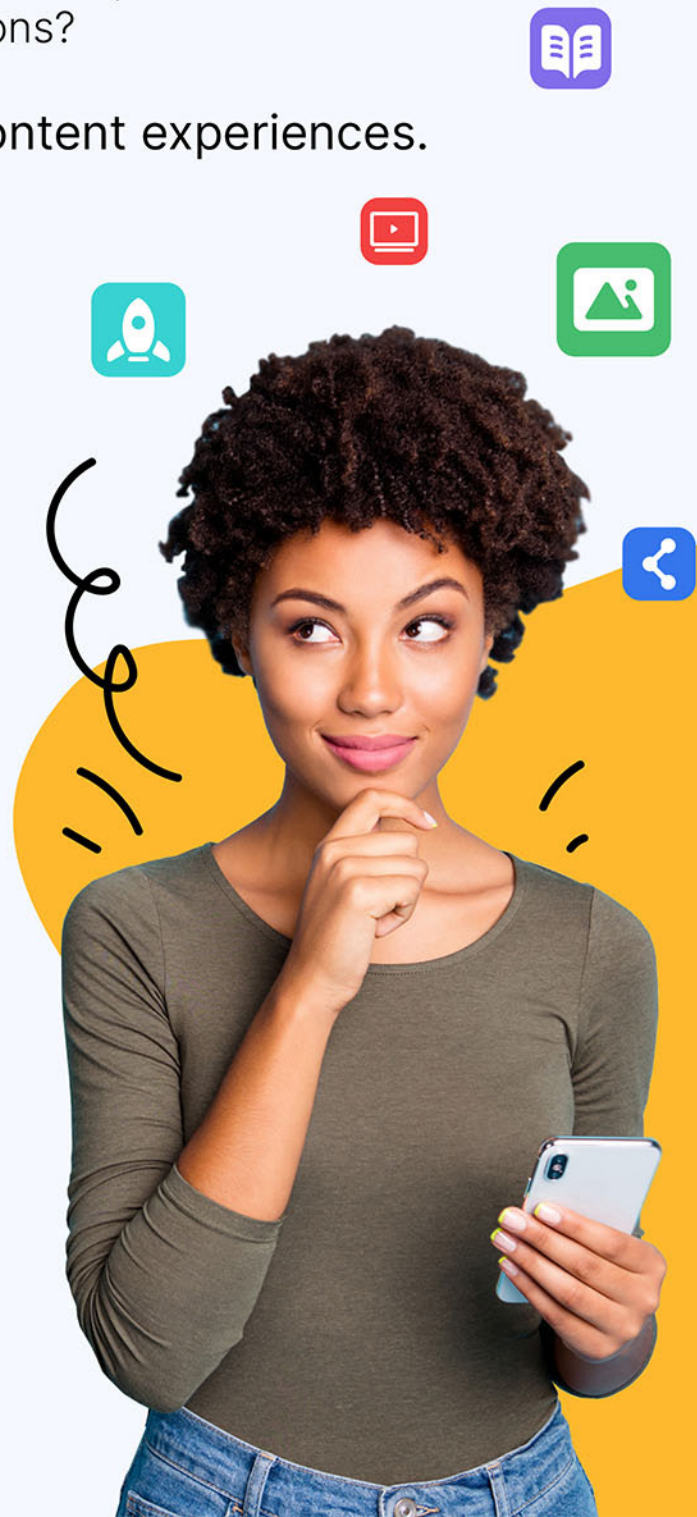
McCANN-ERICKSON

NATIONAL GEOGRAPHIC

Spotify

ESTÉE LAUDER

[Request a Demo →](#)



A COMPLETE GUIDE

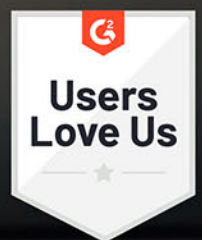
CORE WEB VITALS

Brought to you by
Search Engine Journal



Conductor is the #1 rated SEO technology platform by customers and industry experts.

Take a demo to see how
Conductor gets you found.



Content

08

Introduction

Google's New Ranking Factor: Core Web Vitals — Are You Ready?

Anna Lea Crowe

11

Chapter 1

How You Can Measure Core Web Vitals

Matt Southern

28

Chapter 2

Core Web Vitals FAQ: Advice & Insights from Google

Roger Montti

42

Chapter 3

First Input Delay - A Simple Explanation

Roger Montti

No code visual media optimization



Media Optimizer from Cloudinary allows businesses to automatically deliver images and video in smaller file sizes with high visual fidelity—without the need to manually create format, quality, and size variants, or write URL parameters.

That means better web performance, improved Core Web Vitals, increased SEO rankings, and an enhanced user experience with no code required.

MEET MEDIA OPTIMIZER

Content

60

Chapter 4

Cumulative Layout Shift — Overview of 2021 Google Ranking Factor

Roger Montti

69

SEJ Partner

How To Optimize Media and Reduce Largest Contentful Paint

Nathan Kelley, Managing Director,
Media Optimizer, Cloudinary

83

Chapter 5

What Is Largest Contentful Paint: An Easy Explanation

Roger Montti

94

Chapter 6

A Technical SEO Guide to Lighthouse Performance Metrics

Jamie Indigo & Rachel Anderson



Introduction

Google's New Ranking Factor: Core Web Vitals — Are You Ready?



Anna Lea Crowe

Assistant Editor at Search Engine Journal

On May 28, 2020, [Google introduced](#) something they called Core Web Vitals.

And, now Core Web Vitals is becoming Google's new ranking factor. Core Web Vitals, presents an opportunity for a rankings boost — but only for those who get it right. This guide will help you learn everything you need to know to take advantage of Core Web Vitals.

Google brought us Panda and Penguin.

In fact, the world's largest search engine brings us hundreds of updates each year.

Some are more impactful than others, and one of those is the impending rollout of Core Web Vitals in June 2021.

If you're an SEO professional or digital marketer, you need to read this.

Core Web Vitals is a specific set of metrics site owners should focus on when optimizing for user experience.

And when Google tells us exactly what it wants in this much detail, we'd best listen.

By defining these Core Web Vitals, Google aims to provide clear guidance on the quality signals it says are “essential to delivering a great user experience on the web.”

Web page performance matters to publishers because fast pages generate more leads, more sales, and more advertising revenue.

Are you positioned to take advantage of Core Web Vitals and enjoy the rankings boost that achieving good scores in these metrics provides?

Measuring your Core Web Vitals is so easy you could probably do it in your sleep (said no one, ever).

Keep scrolling through our guide to get to the heart of the issues of performance on your website.

By the end of this guide, you'll have a clearer understanding of what Core Web Vitals mean and how to diagnose issues.

Ready to get started?



Chapter 1

How You Can Measure Core Web Vitals



Matt Southern

Lead News Writer at Search Engine Journal

Google has defined a set of metrics site owners should focus on when optimizing for user experience.

By defining these Core Web Vitals, Google aims to provide unified guidance for quality signals that Google says are “essential to delivering a great user experience on the web.”

“Optimizing for quality of user experience is key to the long-term success of any site on the web.

Whether you’re a business owner, marketer, or developer, Web Vitals can help you quantify the experience of your site and identify opportunities to improve.”

Google emphasizes the importance of Core Web Vitals over other metrics as they’re critical to all web experiences.

Users' expectations for web experiences can vary according to site and context, but some remain consistent regardless of where they are on the web.

Core Web Vitals are the user experience needs that all websites should be striving to meet.

Specifically, Google identifies the core user experience needs as: loading, interactivity, and visual stability.

Here's how those user experience needs are measured.

Measuring User Experience With Core Web Vitals

Google says site owners can measure the quality of their site's user experience with these metrics:

- **Largest Contentful Paint:** The time it takes for a page's main content to load. An ideal LCP measurement is less than or equal to 2.5 seconds.
- **First Input Delay:** The time it takes for a page to become interactive. An ideal measurement is less than or equal to 100 ms.
- **Cumulative Layout Shift:** The amount of unexpected layout shift of visual page content. An ideal measurement is less than or equal to 0.1.



Google explains why these three metrics, in particular, are so important:



“All of these metrics capture important user-centric outcomes, are field measurable, and have supporting lab diagnostic metric equivalents and tooling.

For example, while Largest Contentful Paint is the topline loading metric, it is also highly dependent on First Contentful Paint (FCP) and Time to First Byte (TTFB), which remain critical to monitor and improve.”

Related: [Googler Explains Usability and User Experience Ranking Factors](#)

How to Measure Core Web Vitals

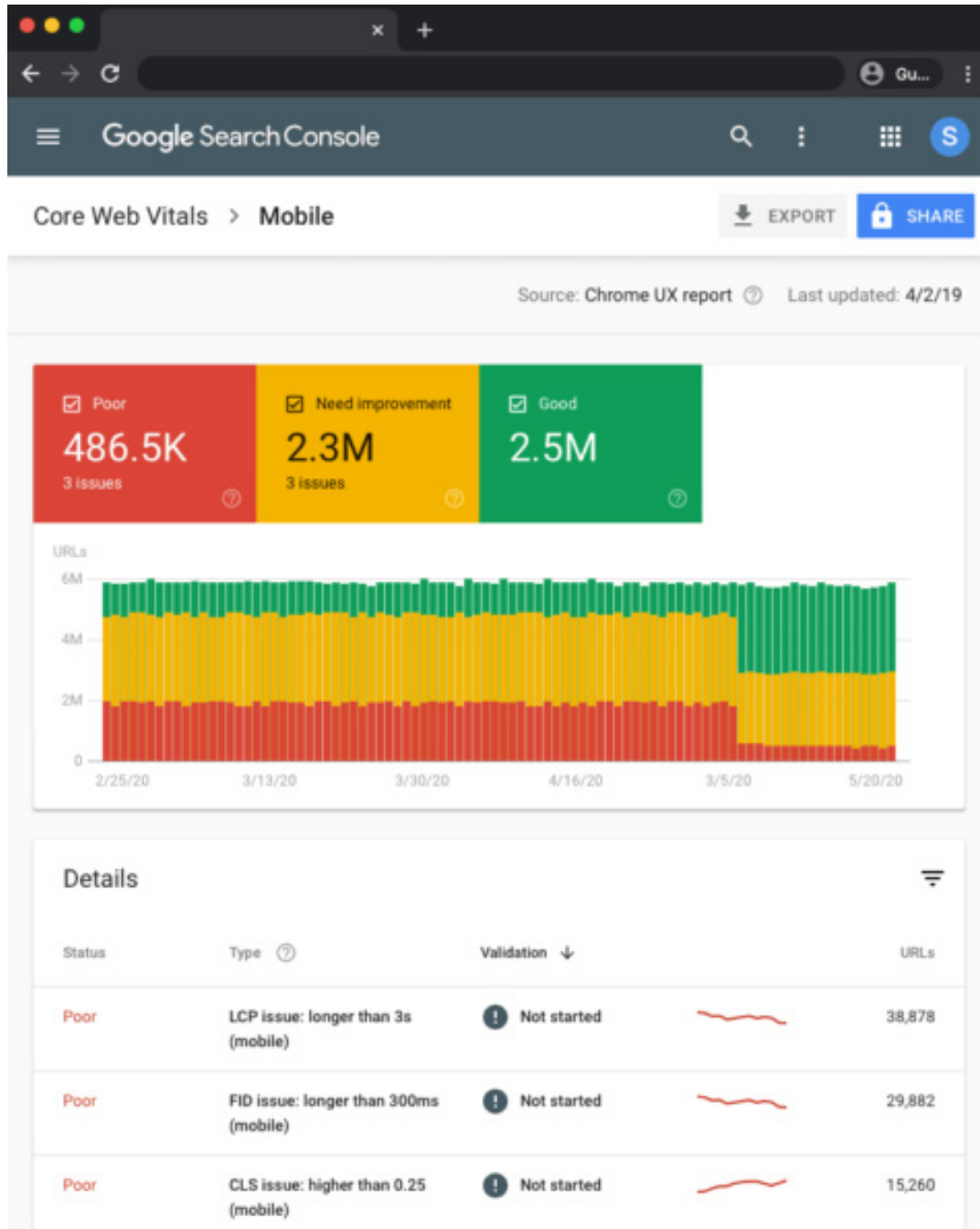
Google incorporates Core Web Vitals measurement capabilities into many of its existing tools.

Core Web Vitals can now be measured using:

- Search Console.
- PageSpeed Insights.
- Lighthouse.
- Chrome DevTools.
- Chrome UX Report.
- Web Vitals Extension.

Here's more about using each of these tools to measure Core Web Vitals.

Search Console



There's a new Core Web Vitals report in Search Console to help site owners to evaluate pages across an entire site.

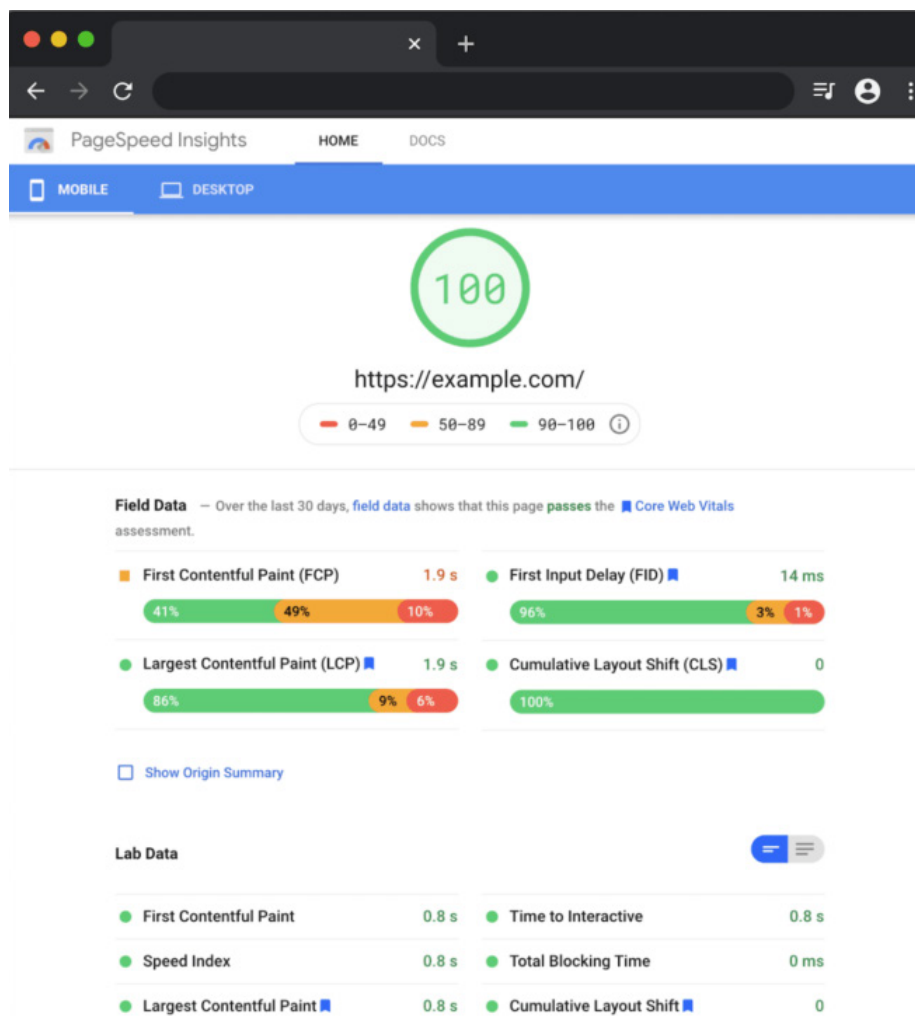
The report identifies groups of pages that require attention based on real-world data from the Chrome UX report.

With this report, be aware that URLs will be omitted if they do not have a minimum amount of reporting data.

PageSpeed Insights

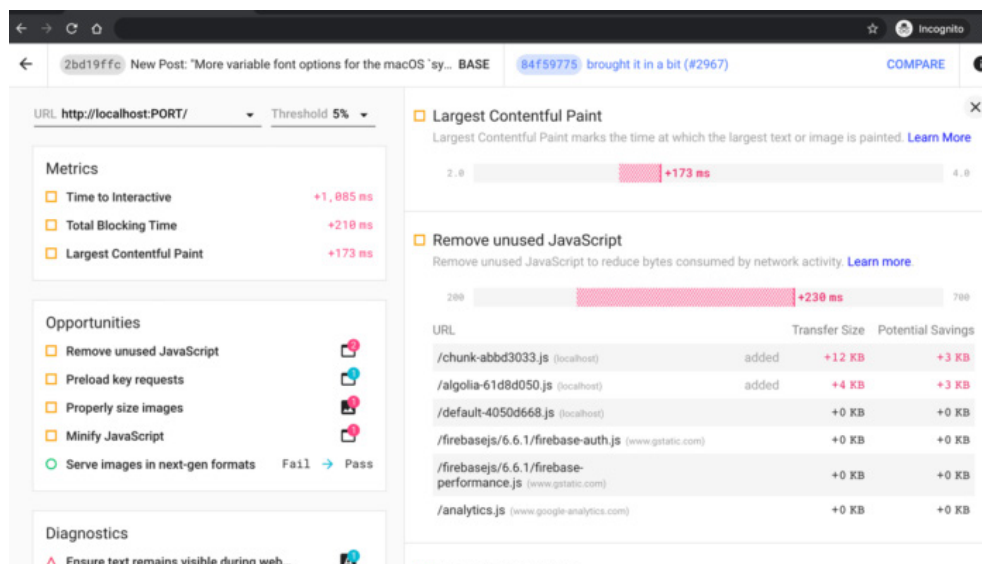
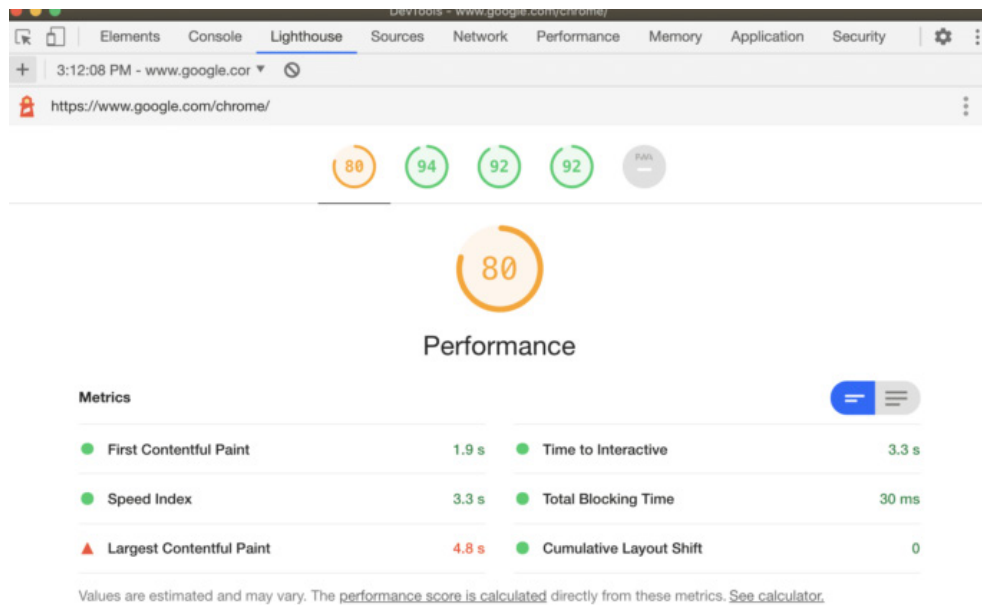
PageSpeed Insights has been upgraded to use Lighthouse 6.0, which makes it capable of measuring Core Web Vitals in both the lab and field sections of the report.

Core Web Vitals are annotated with a blue ribbon, as shown below.



Lighthouse

Lighthouse was recently upgraded to version 6.0, including additional audits, new metrics, and a newly composed performance score.



Two of these new metrics added are Largest Contentful Paint (LCP) and Cumulative Layout Shift (CLS).

These metrics are lab implementations of Core Web Vitals and provide diagnostic information for optimizing user experience.

The third new metric – Total Blocking Time (TBT) – is said to correlate well with First Input Delay (FID), which is another Core Web Vitals metric.

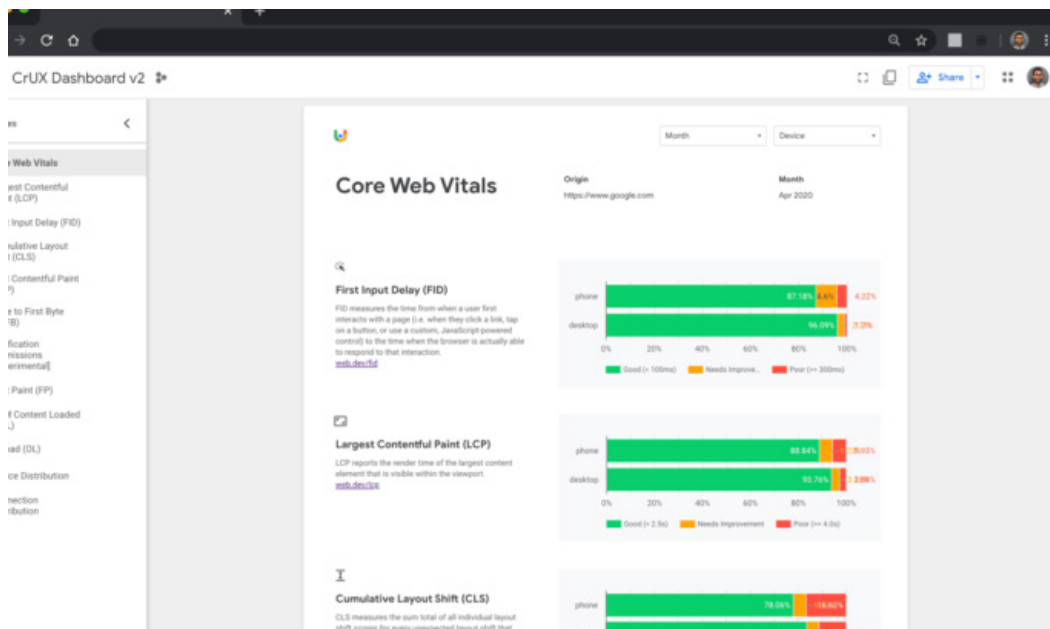
All of the products that Lighthouse powers are updated to reflect the latest version.

Chrome UX Report

Also referred to as CrUX, this report is a public dataset of real user experience data on millions of websites.

The Chrome UX report measures field versions of all the Core Web Vitals, which means it reports real-world data rather than lab data.

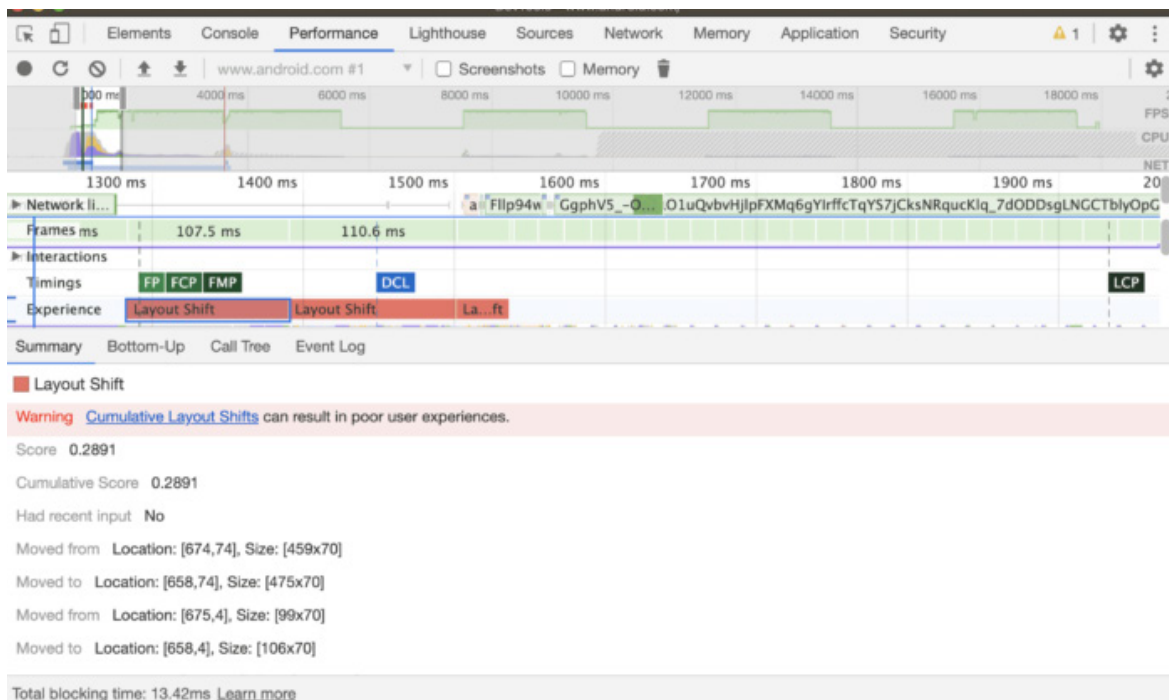
Google has recently updated the report with a new Core Web Vitals landing page.



The report can be accessed [here](#).

Chrome DevTools

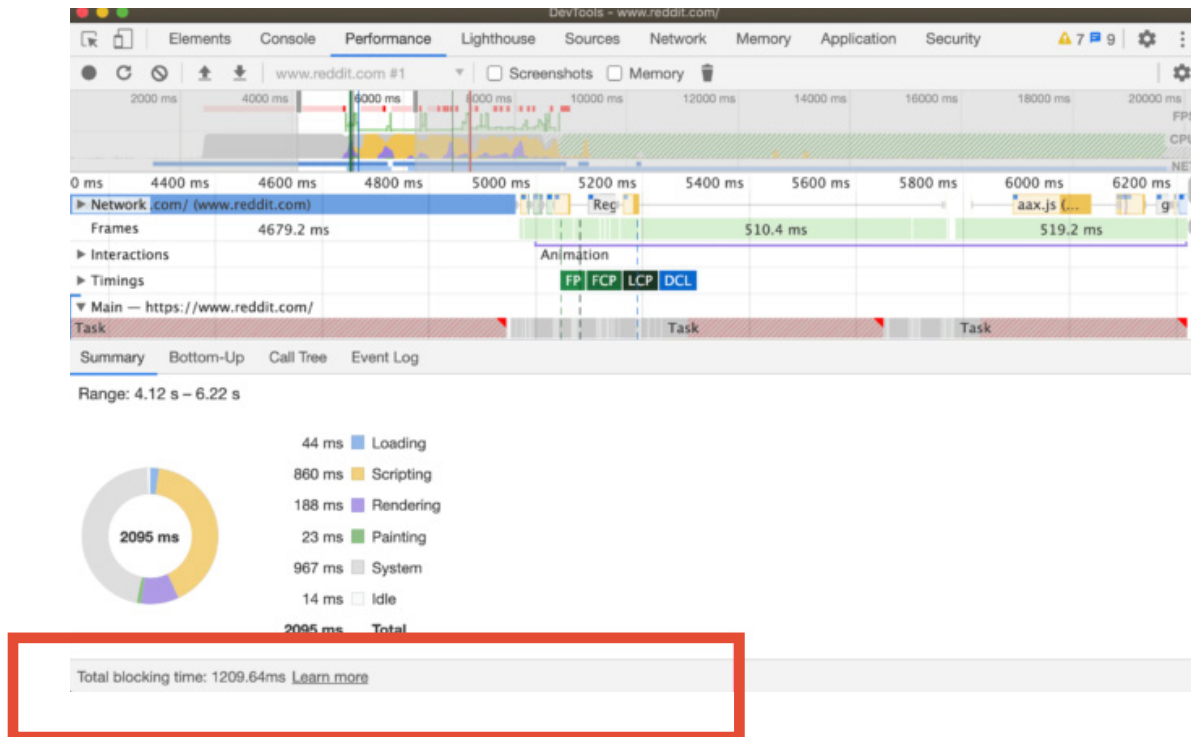
Chrome DevTools has been updated to help site owners find and fix visual instability issues on a page that can contribute to Cumulative Layout Shift (CLS).



Select a Layout Shift to view its details in the Summary tab. To visualize where the shift itself occurred, hover over the Moved from and Moved to fields.

Chrome DevTools also measures Total Blocking Time (TBT), which is useful for improving First Input Delay (FID).

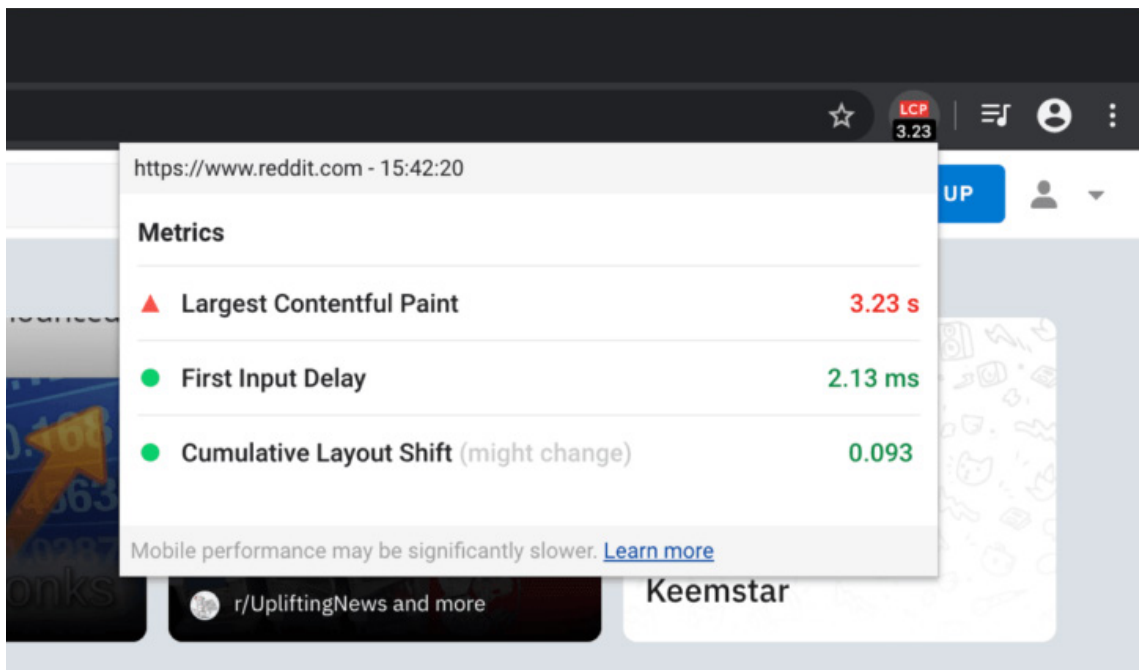
TBT is now shown in the footer of the Chrome DevTools Performance panel when you measure page performance.



Performance optimizations that improve TBT in the lab should also improve FID.

Web Vitals Extension

A new extension, now available to install from the Chrome Web Store, measures the three Core Web Vitals metrics in real-time.



You can download and install the extension [here](#).

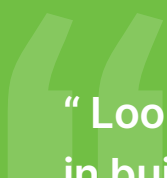
What About Other Valuable Metrics?

As important as the Core Web Vitals are, they're not the only user experience metrics to focus on.

As Google improves its understanding of user experience going forward, it will update the Core Web Vitals annually.

Google will also provide updates on the future Web Vitals candidates, the motivation behind choosing them, and the implementation status.

For now, at least, the company is heavily invested in improving its understanding of page speed:



“ Looking ahead towards 2021, we are investing in building better understanding and ability to measure page speed and other critical user experience characteristics.



For example, extending the ability to measure input latency across all interactions, not just the first; new metrics to measure and quantify smoothness; primitives and supporting metrics that will enable delivery of instant and privacy preserving experiences on the web; and more.”



Chapter 2

Core Web Vitals FAQ: Advice & Insights from Google



Roger Montti

News Writer at SEJ and Owner at Martinibuster.com

Google published a document that provides insights into how Core Web Vitals (CWV) works and the value for ranking purposes. This article discusses it.

Core Web Vitals (CWV) is a set of metrics developed by Google to help website publishers improve page performance for the benefit of site visitors. Web page performance matters to publishers because fast pages generate more leads, more sales, and more advertising revenue.

Page performance is important to site visitors because it reduces the time it takes for them to get what they want.

Beginning in mid June 2021 (previously May 2021) Core Web Vitals becomes a minor ranking factor. Some articles have overstated the importance of CWV as a being a critical ranking factor. But that's not accurate.

Relevance has always been the most important ranking factor, even more important than page speed.

[Statements from Google's John Mueller](#) assure that relevance will continue to be the stronger influence when Core Web Vitals becomes a ranking factor.



According to Mueller:

All of these metrics capture important user-centric outcomes, are field measurable, and have supporting lab diagnostic metric equivalents and tooling.

For example, while Largest Contentful Paint is the topline loading metric, it is also highly dependent on First Contentful Paint (FCP) and Time to First Byte (TTFB), which remain critical to monitor and improve.

While Core Web Vitals may not necessarily have a noticeable impact on rankings, it remains inadvisable to ignore the Core Web Vitals metric. A poor performing web page causes disadvantages in other ways such as lower earnings and possibly less popularity.

Popularity is a key to important ranking factors like links. So it can be asserted that ranking better for Core Web Vitals could help rankings in an indirect manner in addition to the direct ranking boost given by Google's algorithm.

CWV Intended to Encourage a Healthy Web Experience

The goal for Core Web Vitals is to have a shared metric for all sites in order to improve user experience across the web.

Q: Is Google recommending that all my pages hit these thresholds? What's the benefit?

A: We recommend that websites use these three thresholds as a guidepost for optimal user experience across all pages.

Core Web Vitals thresholds are assessed at the per-page level, and you might find that some pages are above and others below these thresholds.

The immediate benefit will be a better experience for users that visit your site, but in the long-term we believe that working towards a shared set of user experience metrics and thresholds across all websites, will be critical in order to sustain a healthy web ecosystem.

AMP Is a Fairly Reliable Way to Score Well

AMP is an acronym for Accelerated Mobile Pages. It's an HTML framework for delivering to mobile devices web pages that are slimmed down, load fast, and are attractive.

AMP was originally developed by Google but is open source. AMP can accommodate ecommerce sites as well as informational sites. There are, for example, apps for the Shopify ecommerce platform as well as plugins for WordPress sites that make it easy to add AMP functionality to a website.

Google will show preference to a website's AMP version for the purposes of calculating Core Web Vitals score. So if a site is having a difficult time optimizing for core web vitals, using AMP is a fast and easy way to gain a high Core Web Vitals score.

Nevertheless, Google warned that there are factors like a slow server or poorly optimized images that can still negatively impact the core web vitals score.

Q: If I built AMP pages, do they meet the recommended thresholds?

A: There is a high likelihood that AMP pages will meet the thresholds. AMP is about delivering high-quality, user-first experiences; its initial design goals are closely aligned with what Core Web Vitals measure today.

This means that sites built using AMP likely can easily meet Web Vitals thresholds.

Furthermore, AMP's evergreen release enables site owners to get these performance improvements without having to change their codebase or invest in additional resources.

It is important to note that there are things outside of AMP's control which can result in pages not meeting the thresholds, such as slow server response times and un-optimized images.

First Input Delay Does Not Consider Scrolling or Bounce/Abandon

First Input Delay (FID) is a metric that measures the time it takes from when a site visitor interacts with a site to when the browser responds to that interaction.

Once a site appears to be downloaded and interactive elements appear to be ready to be interacted with, a user should ideally be able to start clicking around without delay.

A bounce is when a visitor visits a site but then soon after abandons the page, presumably returning back to the search page.

The question is about bounced sessions but the answer incorporates scrolling as well.

Google answers that bounce and abandonment are not a part of the FID metric, presumably because there was no interaction.

Q: Can sessions that don't report FID be considered "bounced" sessions?

A: No, FID excludes scrolls, and there are legitimate sessions with no non-scroll input. Bounce Rate and Abandonment Rate may be defined as part of your analytics suite of choice and are not considered in the design of CWV metric.

Core Web Vitals Impacts Ranking

This section reiterates and confirms that Core Web Vitals will become a ranking signal in June 2021.

Starting June 2021, Core Web vitals will be included in page experience signals together with existing search signals including mobile-friendliness, safe-browsing, HTTPS-security, and intrusive interstitial guidelines.

Importance of Core Web Vitals Ranking Signal For Ranking

Ranking signals are said to have different weights. That's a reflection that some ranking signals have more importance than other ranking signals.

So when it's said that a ranking signal is weighted more than another ranking signal, that means that it's more important.

This is an interesting section of the FAQ because it deals with how much weight the Core Web Vitals ranking signal has compared to other ranking signals.

Google appears to say that the Core Web Vitals ranking signal is weaker than other ranking signals that are directly related to satisfying a user query.

So it's almost like there is a hierarchy of signals, with intent-related signals given more importance than user experience signals.

Here's how Google explains it:

Q: How does Google determine which pages are affected by the assessment of Page Experience and usage as a ranking signal?

A: Page experience is just one of many signals that are used to rank pages. Keep in mind that intent of the search query is still a very strong signal, so a page with a subpar page experience may still rank highly if it has great, relevant content.

Q: What can site owners expect to happen to their traffic if they don't hit Core Web Vitals performance metrics?

A: It's difficult to make any kind of general prediction. We may have more to share in the future when we formally announce the changes are coming into effect. Keep in mind that the content itself and its match to the kind of information a user is seeking remains a very strong signal as well.

Field Data in Search Console Core Web Vitals Reporting

This next section explains possible discrepancies between what a publisher experiences in terms of download speed and what users on different devices and Internet connections might experience.

That's why Google Search Console may report that a site scores low on Core Web Vitals despite the site being perceived as fast by the publisher.

More importantly, the Core Web Vitals metric is concerned with more than just speed.

Furthermore, the Search Console report is based on real-world data whereas Lighthouse data is based on simulated users on simulated devices and simulated internet connections.

Real-world data is called Field Data while the testing based on simulations is called Lab Data.

Q: My page is fast. Why do I see warnings on the Search Console Core Web Vitals report?

A: Different devices, network connections, geography, and other factors may contribute to how a page loads and is experienced by a particular user. While some users, in certain conditions, can observe a good experience, this may not be indicative of other user's experience.

Core Web Vitals look at the full body of user visits and its thresholds are assessed at the 75th percentile across the body of users. The SC CWV report helps report on this data.

...remember that Core Web Vitals is looking at more than speed. For instance, Cumulative Layout Shift describes users annoyances like content moving around...

Q: When I look at Lighthouse, I see no errors. Why do I see errors on the Search Console report?

A: The Search Console Core Web Vitals report shows how your pages are performing based on real world usage data from the CrUX report (sometimes called “field data”). Lighthouse, on the other hand, shows data based on what is called “lab data”. Lab data is useful for debugging performance issues while developing a website, as it is collected in a controlled environment. However, it may not capture real-world bottlenecks.

Google published a Frequently Asked Questions section about Core Web Vitals that answers many questions. While the above questions were the ones I thought were particularly interesting, do take a moment to review the rest of the FAQ as there is much more information there.



Chapter 3

First Input Delay - A Simple Explanation



Roger Montti

News Writer at SEJ and Owner at Martinibuster.com

First Input Delay (FID) is a user experience metric that Google introduced and will soon use as a small ranking factor. This article offers an easy-to-understand overview of FID to help make sense of the topic.

First input delay is more than trying to please Google. Improvements to site performance generally lead to increased sales, ad revenue, and leads.

Definition of First Input Delay

FID is the measurement of the time it takes for a browser to respond to a site visitor's first interaction with the site while the site is loading. This is sometimes called Input Latency.

An interaction can be tapping a button, a link, or a keypress, and the response given in response. Text input areas, dropdowns, and checkboxes are other kinds of interaction points that FID will measure.

Scrolling or zooming do not count as interactions because there's no response expected from the site itself.

The goal for FID is to measure how responsive a site is while it's loading.

The Cause of First Input Delay

First Input Delay is generally caused by images and scripts that download in a non-orderly manner. This disordered coding causes the web page download to excessively pause, then start, then pause. This causes unresponsive behavior for site visitors attempting to interact with the web page.

It's like a traffic jam caused by a free for all where there are no traffic signals, which causes accidents and slowdowns. Fixing it is about bringing order to the traffic.

Google describes the [cause of input latency](#) like this:

“In general, input delay (a.k.a. input latency) happens because the browser’s main thread is busy doing something else, so it can’t (yet) respond to the user. One common reason this might happen is the browser is busy parsing and executing a large JavaScript file loaded by your app. While it’s doing that, it can’t run any event listeners because the JavaScript it’s loading might tell it to do something else.”

How to Fix Input Latency

Since the root cause of First Input Delay is the disorganized download of scripts and images, the way to fix the problem is to thoughtfully bring order to how those scripts and images are presented to the browser for download.

Solving the problem of FID generally consists of using HTML attributes to control how scripts download, optimizing images (the HTML and the images), and thoughtfully omitting unnecessary scripts. The goal is to optimize what is downloaded to eliminate the typical pause and start download of unorganized web pages.

Why Browsers Become Unresponsive

Browsers are software that complete tasks to show a web page. The tasks consist of downloading code, images, fonts, style information and scripts, and then running (executing) the scripts and building the web page according to the HTML instructions.

This process is called rendering. The word render means “to make” and that’s what a browser does by assembling the code and images to render a web page.

The individual rendering tasks are called threads. The word thread is short for “thread of execution” which means an individual sequence of action (in this case, the many individual tasks done to render a web page).

In a browser, there is one thread that’s called the Main Thread. The main thread is responsible for creating (rendering) the web page that a site visitor sees.

The main thread can be visualized as a highway in which cars are symbolic of the images and scripts that are downloading and executing when a person visits a website.

Some code is large and slow. This causes the other tasks to stop and wait for the big and slow code to get off the highway (finish downloading and executing).

The goal is to code the web page in a manner that optimizes which code is downloaded first, when the code is executed, in an orderly manner so that the web page downloads in the fastest possible manner.

Don't Lose Sleep Over Third-Party Code

When it comes to Core Web Vitals and especially with First Input Delay, there is some code over which there is little that can be done. However, this is likely to be the case with your competitors as well.

For example, if your business depends on Google AdSense (a big render blocking script) the problem is going to be the same for your competitor. There are solutions like [lazy loading using Google Ad Manager](#), however.

In some cases, it may be enough to do the best you can because your competitors may not do any better either.

So in those cases, it's best to take your wins where you can find them and don't sweat the losses where you can't make a change.

JavaScript Impact on First Input Delay

JavaScript is like a little engine that makes things happen. When a name is entered on a form, a JavaScript might be there to make sure both the first and last name is entered. When a button is pressed, a JavaScript may be there to tell the browser to spawn a thank you message in a popup.

The problem with JavaScript is that it not only has to download but it also has to run (execute). So those are two things that contribute to input latency.

If a big JavaScript file is located near the top of the page, then that file is going to block the rest of the page beneath it from rendering (becoming visible and interactive) until that script is finished downloading and executing.

This is called blocking the page.

The obvious solution is to relocate these kinds of scripts from the top of the page and put them at the bottom of the page so that they don't interfere with all the other page elements that are waiting to render.

But this can be a problem if, for example, it's placed at the end of a very long web page. The reason is because once the large page is loaded and the user is ready to interact with it, the browser will still be signaling that it is downloading (because the big JavaScript file is lagging at the end). The page may download faster but then stall while waiting for the JavaScript to execute.

There's a solution for that!

Defer and Async Attributes

The Defer and Async HTML attributes are like traffic signals that control the start and stop of how JavaScript downloads and executes.

An HTML attribute is something that transforms an HTML element, kind of like extending the purpose or behavior of the element.

It's like if you learn a skill, that skill becomes an attribute of who you are.

In this case, the Defer and Async attributes tell the browser to not block HTML parsing while downloading. These attributes tell the browser to keep the main thread going while the JavaScript is downloading.

Async Attribute

JavaScript files with the Async attribute will download and then execute as soon as it is downloaded. When it begins to execute is the point at which the JavaScript file blocks the main thread.

Normally the file would block the main thread when it begins to download. But not with the async (or defer) attribute.

This is called an asynchronous download, where it downloads independently of the main thread and in parallel with it.

The async attribute is useful for third-party JavaScript files like advertising and social sharing, files in which it doesn't matter in what order they are executed.

Defer Attribute

JavaScript files with the “defer” attribute will also download asynchronously.

But the deferred JavaScript file will not execute until the entire page is downloaded and rendered. Deferred scripts also execute in the order in which they are located on a web page.

Scripts with the defer attribute are useful for JavaScript files that depend on page elements being loaded and when the order they are executed matter.

In general, use the defer attribute for scripts that aren't essential to the rendering of the page itself.

Input Latency Is Different for All Users

It's important to be aware that First Input Delay scores are variable and inconsistent. The scores vary from visitor to visitor.

The variance in scores is unavoidable because the score depends on interactions that are particular to the individual visiting a site.

Some visitors might be distracted and not interact until a moment where all the assets are loaded and ready to be interacted with.

This is how Google describe it:

“Not all users will interact with your site every time they visit. And not all interactions are relevant to FID...

In addition, some user's first interactions will be at bad times (when the main thread is busy for an extended period of time), and some user's first interactions will be at good times (when the main thread is completely idle).

This means some users will have no FID values, some users will have low FID values, and some users will probably have high FID values.

Why Most Sites Fail FID

Unfortunately, many content management systems, themes, and plugins were not built to comply with this relatively new metric.

This is the reason why so many publishers are dismayed to discover that their sites don't pass the First Input Delay test.

But that's changing as the web software development community responds to demands for different coding standards from the publishing community.

And it's not that the software developers making content management systems are at fault for producing products that don't measure up against these metrics.

For example, WordPress addressed a shortcoming in the Gutenberg website editor that was causing it to score less well than it could.

Gutenberg is a visual way to build sites using the interface or metaphor of blocks. There's a widgets block, a contact form block, and a footer block, etc.

So the process of creating a web page is more visual and done through the metaphor of building blocks, literally building a page with different blocks.

There are different kinds of blocks that look and behave in different ways. Each individual block has a corresponding style code (CSS), with much of it being specific and unique to that individual block.

The standard way of coding these styles is to create one style sheet containing the styles that are unique to each block. It makes sense to do it this way because you have a central location where all the code specific to blocks exists.

The result is that on a page that might consist of (let's say) twenty blocks, WordPress would load the styles for those blocks plus all the other blocks that aren't being used.

This is before Core Web Vitals (CWV) that was considered as the standard way to package up CSS.

After the introduction of Core Web Vitals that practice is considered code bloat.

This is not meant as a slight against the WordPress developers. They did a fantastic job.

This is just a reflection of how rapidly changing standards can hit a bottleneck at the software development stage before being integrated into the coding ecosystem.

We went through the same thing with the transition to mobile-first web design.

Gutenberg 10.1 Improved Performance

WordPress [Gutenberg 10.1 introduced an improved way to load the styles](#) by only loading the styles that were needed and not loading the block styles that weren't going to be used.

This is a huge win for WordPress, the publishers who rely on WordPress, and of course, the users who visit sites created with WordPress.

Time to Fix First Input Delay is Now

I believe that moving forward more and more software developers responsible for the CMS, themes, and plugins will transition to First Input Delay-friendly coding practices.

But until that happens, the burden is on the publisher to take steps to improve First Input Delay. Understanding it is the first step.



Chapter 4

Cumulative Layout Shift — Overview of 2021 Google Ranking Factor



Roger Montti

News Writer at SEJ and Owner at Martinibuster.com

Cumulative Layout Shift (CLS) is a Google metric that measures a user experience event. This metric is said to become a ranking factor in 2021. That means it's important to understand what CLS is and how to optimize for it.

Definition of Cumulative Layout Shift

What Is Cumulative Layout Shift?

CLS is the unexpected shifting of web page elements while the page is still downloading. The kinds of elements that tend to cause shift are fonts, images, videos, contact forms, buttons, and other kinds of content.

Minimizing CLS is important because pages that shift around can cause a poor user experience.

A poor CLS score is indicative of coding issues that can be solved.

Why CLS Happens

According to Google, there are five reasons why Cumulative Layout Shift happens:

- 1.** “Images without dimensions.
- 2.** Ads, embeds, and iframes without dimensions.
- 3.** Dynamically injected content.
- 4.** Web Fonts causing FOIT/FOUT.
- 5.** Actions waiting for a network response before updating DOM.”

Images and videos need to have the height and width dimensions declared in the HTML. With regard to responsive images, make sure that the different images sizes for the different viewports use the same aspect ratio.

Google recommends using [AspectRatioCalculator.com](https://aspectratiocalculator.com) to calculate the aspect ratios. It's a good resource.

Ads Can Cause CLS

This one is a little tricky to deal with. One way to deal with ads that cause CLS is to style the element where the ad is going to show.

For example, if you style the div to have a specific height and width then the ad will be constrained to those.

There are two solutions if there's a lack of inventory and an ad doesn't show up.

If an element containing an ad does not show an ad, you can set it so that an alternative banner ad or placeholder image is used to fill the space.

Alternatively, for some layouts where an ad fills an entire row on the top of perhaps a column on the right or left gutter of a web page, if the page does not show up there won't be a shift. It won't make a difference either on mobile or desktop. But that depends on the theme layout.

You'll have to test that out if ad inventory is an issue.

Dynamically Injected Content

This is content that is injected into the web page. For example, in WordPress, you can link to a YouTube video or a Tweet and WordPress will display the video or tweet as an embedded object.

Web Based Fonts

Downloaded web fonts can cause what's known as Flash of invisible text (FOIT) and Flash of Unstyled Text (FOUT).

A way to prevent that is to use `rel="preload"` in the link for downloading that web font.

Lighthouse can help you diagnose what is causing CLS.

CLS Can Sneak in During Development

Cumulative layout shift can slip through during the development stage. What can happen is that many of the assets needed to render the page are loaded onto a browser's cache.

The next time a developer or publisher visits the page under development, they won't notice a layout shift because the page elements are already downloaded.

That's why it's useful to have a measurement in the lab or in the field.

How Cumulative Layout Shift is Calculated

The calculation involves two metrics/events. The first is called Impact Fraction.

Impact Fraction

Impact fraction is a measurement of how much space an unstable element takes up in the viewport.

A viewport is what you see on the mobile screen.

When an element downloads and then shifts, the total space that the element occupied, from the location that it occupied in the viewport when it's first rendered to the final location when the page is rendered.

The example that Google uses is an element that occupies 50% of the viewport and then drops down by another 25%.

When added together, the 75% value is called the Impact Fraction and it's expressed as a score of 0.75.

Distance Fraction

The second measurement is called the Distance Fraction. The distance fraction is the amount of space that the page element has moved from the original position to the final position.

In the above example, the page element moved 25%.

So now the Cumulative Layout Score is calculated by multiplying the Impact Fraction by the Distance Fraction:

$$0.75 \times 0.25 = 0.1875$$

There's some more math and other considerations that go into the calculation. What's important to take away from this is that the score is one way to measure an important user experience factor.

How to Measure CLS

There are two ways to measure CLS. Google calls the first way in the Lab. The second way is called in the Field.

In the lab means simulating an actual user downloading a web page. Google uses a [simulated Moto G4 for generating the CLS score](#) within the lab environment.

Lab tools are best for understanding how a layout may perform before pushing it live to users. It gives publishers the opportunity to [test a layout for issues](#).

[Lab tools](#) consist of [Chrome Dev Tools](#) and [Lighthouse](#).

Understand Cumulative Layout Shift

It's important to understand Cumulative Layout Shift. It's not necessary to understand how to do the calculations yourself. But just knowing about it and what it is important because this metric is scheduled to become a ranking factor some time in 2021.



SEJ Partner

How To Optimize Media and Reduce Largest Contentful Paint



Nathan Kelley

Managing Director, Media Optimizer, Cloudinary

Though conventional factors will still apply to a company's optimization efforts, the anticipated rollout of [Google's new Core Web Vital metrics](#) has been a hot topic in SEO, and for a good reason.

Google gave brands and developers a 12-month head start to make preparations, knowing the prize to be won – the appeal of more prominent search rankings – was something worth optimizing for.

Largest Contentful Paint (LCP), First Input Delay (FID), and Cumulative Layout Shift (CLS) measure how well a website meets a user's expectations of an online experience – taking into account a page's load time, interactivity, and stability, respectively.

With several best practices swirling for how to meet the thresholds Google has set for each, it's essential to understand that the increasingly visual nature of our world requires us to rethink how we leverage rich media online.

Our websites will be more likely to be found in search, and then enjoyed by site visitors, if we address the ways that media is hindering web performance.

The Digital – and Therefore, Visual – Nature of Our World Is Accelerating Rapidly

Almost every aspect of our daily lives got pushed online in 2020.

Work, school, commerce, entertainment – the usual mechanics of our day-to-day suddenly went virtual, and all of us spent more time online than perhaps we ever had.

Data from McKinsey shows that the U.S. leaped [10 years forward in 90 days' time](#).

Analyzing consumer behavior in 2020 paints the digital picture quite clearly.

- During peak times, Zoom reported [300 million](#) daily participants in virtual meetings on its platform.
- Broadband data usage increased [47%](#) in the early days of the COVID-19 pandemic.
- Consumer ecommerce spending was up [44%](#) in 2020 compared to the year prior.

All of this sets the scene for a conversation around web performance and online visual media, especially as it relates to Google's new Core Web Vital metrics, and specifically LCP.

As consumers, we crave more (and better) visual experiences online. Plus, we expect our digital interactions to load without delay.

Then, as brand marketers or teams that support online experiences ourselves, we're inclined to give the people what they want.

But, as we beef up the visual appeal of websites, the sheer volume of these assets adds up.

Over the last few years, HTTP Archive has noted an 85% increase in online page weight. For websites in the 90th percentile, [75%](#) of their page weight is dominated by images and video.

Visual media helps to tell compelling stories on behalf of brands, but the sum of unoptimized media files can detract significantly from user experience.

Page Load Time Matters and Could Be Working Against Your Profitability

Looking back to [Google I/O 2019](#), performance engineer Paul Irish made it clear that page load time, as Google sees it, isn't just "speed for the sake of speed."

Instead, speed has real business implications.

Google's benchmark data demonstrates how longer page load times have a negative impact on bounce rates, making conversion efforts even more challenging.

Speaking to the [critical nature of performance](#) at the 2019 event, Google product manager Elizabeth Sweeney pointed to these interesting [stats](#):

- The probability of a user bouncing from a site increases by 32% when the overall page load time goes from 1 to 3 seconds.
- 53% of mobile site visits are abandoned if a page takes longer than 3 seconds to load.
- At 10 seconds, the chance of bounce is increased by 123%.

Sweeney went on to say:

“The impact on user experience is not minimal. In fact, the speed that it takes for a page to load is revealed to be the most important factor in a user’s mobile experience. It’s more important than how easy it is to find what they want, it’s more important than the simplicity of using the site, and interestingly enough, it is three times more important than what a site looks like.”

Did you catch that?

Page load time is the most important factor in a user’s mobile experience. That’s huge.

So, Let's Get Laser-Focused on LCP

As stated, Largest Contentful Paint measures the render time of the largest visible content element, which is usually a hero image or carousel of photos.

Or, as explained by Ryan Shelley in a [recent Hack My Growth episode](#):

“Load time is not how fast the entire page loads, but how fast most of the content, the most important part of the page, loads. Can a user see what they want to see quickly?”

Specific to LCP, Google recommends a target page load time of ≤ 2.5 seconds for the largest content to render.

What's concerning is that in 2020, it was determined that [47%](#) of websites have an LCP score of greater than 2.5 seconds.

There's a good chance your site could fall into that nearly-half "Needs Improvement" segment, so there's certainly work to be done.

It's important to note that LCP can mean different things for different industries or verticals.

For example, an e-commerce site will naturally have more visual, rich-media content weighing a page down than, let's say, a financial services company.

We're about to focus on best practices for media optimization, but as you consider the LCP metric and work to reduce the page load time accordingly, keep in mind that the largest "content" may not be obvious at first.

How can you identify the element that dictates your LCP score?

Check out your dashboard in [Google's DevTools performance panel](#), and leverage [Lighthouse's audit](#), too.

Best Practices for Optimizing Media for Improved LCP

Optimizing media is the right next step for improving a site's LCP score.

So, what action can you take to start chipping away at that detrimental lag time?

- Convert your photos to more efficient, modern formats. These could include AVIF, JPEG 2000, JPEG XL, or WebP.
- Compress images to reduce their bandwidth, but ensure that they still display in high quality. In addition, you can lazy load when possible.
- Use responsive images to ensure that visuals look awesome, no matter the size or orientation of the device they're viewed on.
- GIFs are prevalent in social media and on the web, but can drag significantly on page load time. Use video as an alternative to overcome this.

- For further speed improvements, deliver the largest content elements (those that directly impact an LCP score) from the cache instead of the original asset location.

Addy Osmani, the technical lead for Web Performance Tooling at Google, confesses that “even us who are web perf enthusiasts sometimes have a hard time staying on top of everything in the image optimization world.”

But [one recommendation he has](#) is the use of an image Content Delivery Network (CDN), or better yet, a multi-CDN approach to be more reliable and scalable.

“I’m seeing an increasing number of sites leveraging image CDNs to get control, tweak parameters in a URL for an image, and change what format and quality get served down to a user,” Osmani says.

Consider Using an Automated Image Optimization Tool

To quote Google's Osmani one more time:

“For many sites, images are the largest element in view when the page has finished loading. Optimize them.”

As a first step, companies can utilize tools such as [Website Speed Test](#) to perform an image analysis. This tool provides measurable and actionable information about how to go beyond simple compression.

In addition to the aforementioned methodical steps toward improved page load times, businesses must embrace enterprise-wide optimization capabilities.

This is because optimizing media has to be done in a way that doesn't have the opposite outcome as intended, serving up compressed imagery that loads quickly but disappoints in quality.

At the scale that digital businesses have to create and distribute new content, a tool that provides automation for optimization efforts is key.

Instead of leveraging multiple discrete tools, you could explore a product like Cloudinary's [Media Optimizer](#), which effectively and efficiently optimizes media, delivering the right format and quality through multi-CDN edge nodes.

In other words, Media Optimizer optimizes both quality and size, serving high visual fidelity in small files.

If automation is introduced to the image workflow, the correct file size needs to be correctly applied while maintaining the visual fidelity of that asset.

In addition to maintaining visual quality, automation will free up team members from doing manual transformations that zap their time and creative energy.

Cropping, reformatting, resizing and other tasks are better suited for a tool that can knock out the needed optimizations in no time.

In looking to a media optimization tool, organizations will benefit from one that provides a real-time dashboard view of insights that speak to web performance.

By having an accurate understanding of how media is performing, teams can know in an instant where improvements can be made, further boosting SEO in the process.

Optimized Media Is Key to a 'Good' LCP

With urgency, digitally minded companies are rising to the occasion to improve the experience visitors have on their website.

In fact, according to Google, there has been “a median [70% increase](#) in the number of users engaging with Lighthouse and PageSpeed Insights” DevTools since the announcement of Core Web Vitals in May of 2020. These tools aren't media optimization solutions – rather, they're open-source, readily available sites for analyzing the content of a webpage.

But this statistic shows that if you're looking to make adjustments and improvements for these new Core Web Vital metrics, then you're in good company.

The user experience is important for a variety of reasons, but with the determination of Core Web Vitals as a definite SEO ranking signal, it makes sense that there's an increasing interest in improving web performance.

With the visual economy taking precedence, optimized media will only grow in importance.

The background of the top half of the page is a white canvas filled with various colorful geometric shapes. These include circles, squares, triangles, and rounded rectangles in shades of red, orange, green, blue, and light gray. Some shapes are solid, while others have a gradient effect, creating a modern and abstract design.

Chapter 5

What Is Largest Contentful Paint: An Easy Explanation



Roger Montti

News Writer at SEJ and Owner at Martinibuster.com

Largest Contentful Paint (LCP) is a Google user experience metric. It is set to become a ranking factor in 2021. This guide explains what LCP is and how to achieve the best scores.

Definition of Largest Contentful Paint

What Is Largest Contentful Paint?

LCP is a measurement of how long it takes for the main content of a page to download and be ready to be interacted with. What is measured is the largest image or block of context within the user viewport. Anything that extends beyond the screen does not count.

Typical elements measured are images, video poster images, background images, and block-level text elements like paragraph tags.

Why Is LCP Measured?

LCP was chosen as a key metric for the Web Vitals score because it accurately measures how fast a web page can be used. Additionally, it is easy to measure and optimize for.

Block-level Elements Used to Calculate the LCP Score

Block-level elements used for calculating the Largest Contentful Paint score can be the `<main>` and `<section>` elements, as well as the heading, div, form elements.

Any block-level HTML element that contains text elements can be used, as long as it's the largest one. Not all elements are used. For example, the SVG and VIDEO elements are not currently used for calculating the Largest Contentful Paint.

LCP is an easy metric to understand because all you have to do is look at your web page and determine what the largest text block or image is and then optimize it by making it smaller or removing anything that would prevent it from downloading quickly.

Because Google includes most sites in the mobile first index, it's best to optimize the mobile viewport first, then the desktop.

Delaying Large Elements Might Not Help

Sometimes a web page will render in parts. A large featured image might take longer to download than the largest text block-level element.

What happens, in this case, is that a *Performance Entry* is logged for the largest text block-level element. But when the featured image at the top of the screen loads, if that element takes up more of the user's screen (their viewport), then another *PerformanceEntry* object will be reported for that image.

Images Can Be Tricky for LCP Scores

Web publishers commonly upload images at their original size and then use HTML or CSS to resize the image to display at a smaller size.

The original size is what Google refers to as the “intrinsic” size of the image.

If a publisher uploads an image that’s 2048 pixels wide and 1152 pixels in height, that 2048 × 1152 height and width are considered the “intrinsic” size.

Now, if the publisher resizes the 2048 × 1152 pixel image to 640 × 360 pixels, the 640×360 size image is called the visible size.

For the purposes of calculating the image size, Google uses whichever size is smaller between the intrinsic and visible size images.

Note About Image Sizes

It's possible to achieve a high Largest Contentful Paint score with a large intrinsic size image that is resized with HTML or CSS to be smaller.

But it's a best practice to make the intrinsic size of the image match the visible size.

The image will download faster and your Largest Contentful Paint score will go up.

How LCP Handles Images Served from Another Domain

Images served from another domain, like from a CDN, are generally not counted in the Largest Contentful Paint calculation.

Publishers who want to have those resources be a part of the calculation need to set what's called a Timing-Allow-Origin header. Adding this header to your site can be tricky because if you use a wildcard (*) in the configuration, then it could open your site up to hacking events.

In order to do it properly, you would have to add a domain that's specific to Google's crawler in order to whitelist it so that it can see the timing information from your CDN.

So at this point, resources (like images) that are loaded from another domain (like from a CDN) will not be counted as part of the LCP calculation.

Scoring Gotchas to Be Aware Of

All elements that are in the user's screen (the viewport) are used to calculate LCP. That means that images that are rendered off-screen and then shift into the layout once they are rendered may not count as part of the Largest Contentful Paint score.

On the opposite end, elements that start out in the user viewport and then get pushed off-screen may be counted as part of the LCP calculation.

How to Get the LCP Score

There are two kinds of scoring tools. The first one is called **Field Tools**, and the second one is called **Lab Tools**.

Field tools are actual measurements of a site. Lab tools give a virtual score based on a simulated crawl using algorithms that approximate Internet conditions that a typical user on a mobile phone might encounter.

How to Optimize for Largest Contentful Paint

There are three main areas to optimize (plus one more for JavaScript Frameworks):

1. Slow servers.
2. Render-blocking JavaScript and CSS.
3. Slow resource load times.

A slow server can be an issue with DDOS levels of hacking and scraper traffic on a shared or VPS host. You may find relief by installing a WordPress plugin like WordFence to find out if you're experiencing a massive onslaught and then block it.

Other issues could be misconfiguration of a dedicated server or VPS. A typical issue can be the amount of memory allotted to PHP.

Other issues could be outdated software like an old

PHP version or CMS software that is outdated.
Worst case scenario is a shared server with multiple users that are slowing down your box. In that case, moving to a better host is the answer.

Typically, applying fixes like adding caching, optimizing images, fixing render blocking CSS and JavaScript, and pre-loading certain assets can help.

Google has a neat tip for dealing with CSS that's not essential for rendering what the user sees:

“Remove any unused CSS entirely or move it to another stylesheet if used on a separate page of your site.

For any CSS not needed for initial rendering, use loadCSS to load files asynchronously, which leverages rel="preload" and onload.

```
<link rel="preload" href="stylesheet.css"  
as="style" onload="this.rel='stylesheet'">
```


Field Tools for LCP Score

Google lists three field tools:

- [PageSpeed Insights](#).
- [Search Console](#) (Core Web Vitals report).
- [Chrome User Experience Report](#).

The last one – Chrome User Experience Report – requires a Google account and a Google Cloud Project. The first two are more straightforward.

Lab Tools for LCP Score

Lab measurements are simulated scores.

- [Chrome DevTools](#).
- [Lighthouse](#).
- [Chrome User Experience Report](#).

The first two tools are provided by Google. The third tool is provided by a third party.



Chapter 6

A Technical SEO Guide to Lighthouse Performance Metrics



Jamie Indigo

Technical SEO Consultant at Not a Robot and DeepCrawl



Rachel Anderson

Director of Web Intelligence at Local SEO Guide

Maybe you're here because you're a die-hard fan of performance metrics. Or maybe you don't know what Lighthouse is and are too afraid to ask.

Either is an excellent option. Welcome!

For this merry adventure into demystifying developer documentation, I've recruited Technical SEO and Google Data Studio nerd, [Rachel Anderson](#).

Together, we're hoping to take your performance improvement efforts from "make all the numbers green" to some clear and meaningful action items.

We're going to look at:

- What the heck is Lighthouse?
(In case you didn't know and were afraid to ask.)
- Updates to how Performance Score is calculated (versions 6 and 7).
- How to test performance using Lighthouse.
- What metrics comprise Lighthouse's Performance score.
- What those metrics mean.
- How to improve them.

What Is Lighthouse?

Lighthouse is an open-source [auditing tool](#) that provides standardized scores across five areas:

- Performance.
- Progressive Web App.
- Best Practices.
- Accessibility.
- SEO.

For the purposes of this article, we're going to use the name Lighthouse to refer to the series of tests executed by the shared Github repo, regardless of the execution method.

Lighthouse runs performance tests using emulated data, also known as lab data.

This is performance data collected within a controlled environment with predefined device and network settings.

Lab data is helpful for debugging performance issues. It does not mean that the experience on your local machine in a controlled environment represents the experiences of real humans in the wild.

Updates to Lighthouse: Web Core Vitals

On May 5, 2020, the Chromium project [announced](#) a set of three metrics with which the Google-backed open-source browser would measure performance.

The metrics, known as Web Vitals, are part of a Google initiative designed to provide unified guidance for quality signals.

The goal of these metrics is to measure web performance in a user-centric manner.

Within two weeks, Lighthouse v6 rolled out with a modified version of [Web Core Vitals](#) at the heart of the update.

July 2020 saw Lighthouse v6's unified metrics adopted across Google products with the release of Chrome 84.

Chrome DevTools Audits panel was renamed to Lighthouse. Pagespeed insights and Google Search Console also reference these unified metrics.

Web Core Vitals comprise 55% of Lighthouse's weighted performance score. This change in focus sets new, more refined goals.

Overall, most pages saw minimal impact with [83.32% of tests](#) shifting ten points or less on the shift to v6.

Version 7 is currently out on [Github](#) and slated for large scale rollout with the stable Chrome 89 release in March 2021.

How to Test Performance Using Lighthouse

Methodology Matters

Out of the box, Lighthouse audits a single page at a time.

A single page score doesn't represent your site, and a fast homepage doesn't mean a fast site.

Test multiple page types within your site.

Identify your major page types, templates, and goal conversion points (signup, subscribe, and checkout pages).

Example Page Testing Inventory

URL	Page Type
/	Homepage
/tools	Category Template
/tools/screwdrivers	Product Listing Page Template
/acme/deluxe-anvil	Product Detail Page Template
/cart	Cart
/checkout	Checkout
/order-confirmation	Order confirmation
/blog	Blog Root
/blog/roadrunners-101	Blog Template

Before you begin optimizing, run Lighthouse on each of your sample pages and save the report data.

Record your scores and the to-do list of improvements.

Prevent data loss by saving the JSON results and utilizing [Lighthouse Viewer](#) when detailed result information is needed.

Get Your Backlog to Bite Back Using ROI

Getting development resources to action SEO recommendations is hard.

An in-house SEO professional could destroy their pancreas by having a birthday cake for every backlogged ticket's birthday. Or at least learn to hate cake.

In my experience as an in-house enterprise SEO pro, the trick to getting performance initiatives prioritized is having the numbers to back the investment.

This starting data will become dollar signs that serve to justify and reward development efforts.

Chances are you're going to have more than one area flagged during tests. That's okay!

If you're wondering which changes will have the most bang for the buck, check out the [Lighthouse Scoring Calculator](#).

How to Run Lighthouse Tests

This is a case of many roads leading to Oz. Sure, some scarecrow might be particularly loud about a certain shade of brick but it's about your goals.

Looking to integrate SEO tests into the release process? Time to learn some NPM.

Have less than five minutes to prep for a prospective client meeting? A couple of one-off reports should do the trick.

Whichever way you execute, default to mobile unless you have a special use-case for desktop.

For One-Off Reports: Chrome Devtools

Test one page at a time with the Lighthouse panel in [Chrome DevTools](#). Because the report will be emulating a user's experience using your browser, use an incognito instance with all extensions, and the browser's cache disabled.

Pro tip: Create a Chrome profile for testing. Keep it local (no sync enabled, password saving, or association to an existing Google account) and don't install extensions for the user.

How to Run a Test Lighthouse Using Chrome DevTools

1. Open an incognito instance of Chrome.
2. Navigate to the Network panel of Chrome Dev Tools (Command + Option + I on Mac or Control + Shift + I on Windows and Linux).
3. Tick the box to disable cache.
4. Navigate to the Lighthouse panel.
5. Click Generate Report.
6. Save the file.

Pros of Running Lighthouse From DevTools

- You can test local builds and authenticated pages.
- Saved reports can be compared using the [Lighthouse CI Diff tool](#).

Cons of Running Lighthouse From DevTools

- One report at a time.
- Requires you to manually save results.

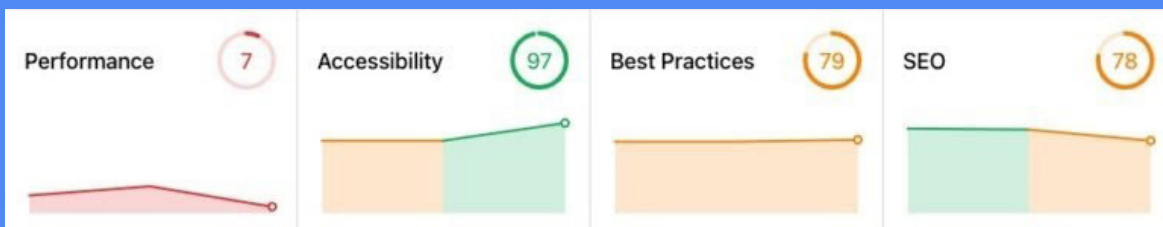
For Testing the Same Pages Frequently: web.dev

It's just like DevTools but you don't have to remember to disable all those pesky extensions!

1. Visit web.dev/measure/.
2. Sign in using your Google account.
3. Enter your URL.
4. Click Run Audit.

Pros of Running Lighthouse From web.dev

- Captures a nifty timeline of results! (As long as you're logged in).



- Quick links to guides for improving issues.
- Saved reports can be compared using the [Lighthouse CI Diff tool](#).

Cons of Running Lighthouse From web.dev

- One report at a time.
- You'll need to remember which URLs you're tracking over time.

For Testing at Scale (and Sanity): Node Command Line

1. [Install npm](#). (**Mac Pro tip:** Use homebrew to avoid obnoxious dependency issues.)
2. Install the [Lighthouse node module](#) with `npm install -g`
3. Run a single test with `lighthouse <url>`
4. Run tests on lists of URLs by running tests [programmatically](#).

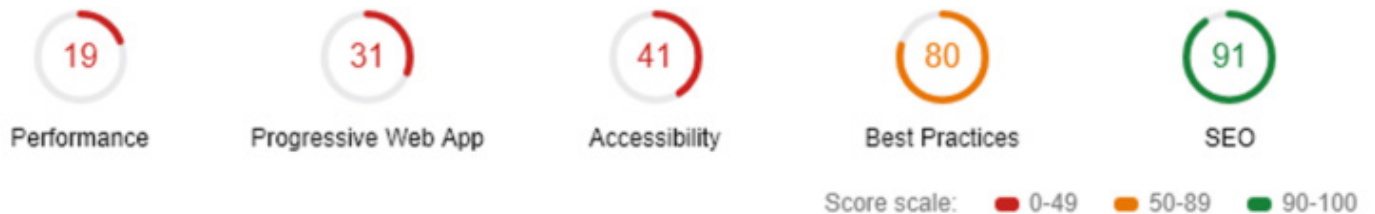
Pros of Running Lighthouse From Node

- Many reports can be run at once.
- Can be set to run automatically to track change over time.

Cons of Running Lighthouse From Node

- Requires some coding knowledge.
- More time-intensive setup.

Lighthouse Performance Metrics Explained



In versions 6 and 7, Lighthouse's performance score made of seven metrics with each contributing a percentage of the total performance score.

Metric Name	Weight
Largest Contentful Paint (LCP)	25%
Total Blocking Time (TBT)	25%
First Contentful Paint (FCP)	15%
Speed Index (SI)	15%
Time To Interactive (TTI)	15%
Cumulative Layout Shift (CLS)	5%

Largest Contentful Paint (LCP)

What it represents: A user's perception of loading experience.

Lighthouse Performance score weighting: 25%

What it measures: The point in the page load timeline when the page's largest image or text block is visible within the viewport.

How it's measured: Lighthouse extracts LCP data from [Chrome's tracing tool](#).

Is Largest Contentful Paint a Web Core Vital? Yes!

LCP Scoring

Goal: Achieve LCP in < 2.5 seconds.

LCP time (in milliseconds)	Color-coding
0–2,500	Green (fast)
2,501–4,000	Orange (moderate)
Over 4,000	Red (slow)

What Elements Can Be Part of LCP?

- Text.
- Images.
- Videos.
- Background images.

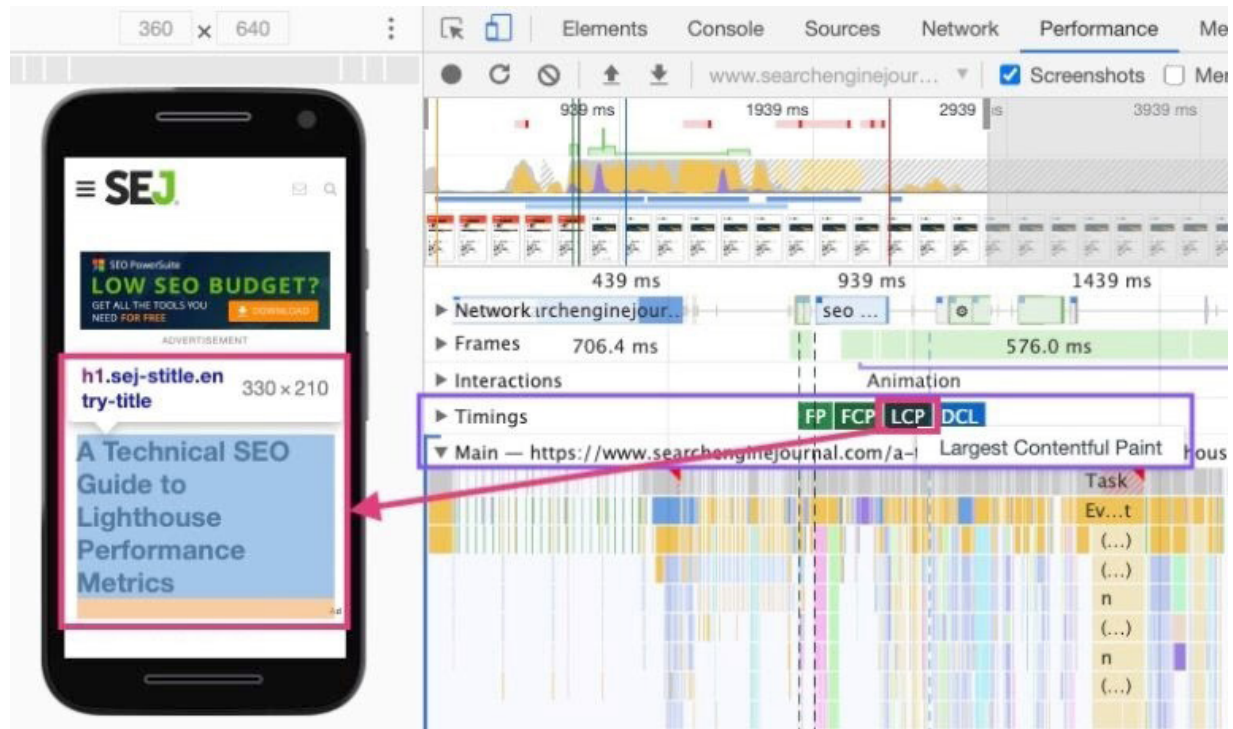
What Counts as LCP on Your Page?

It depends! LCP typically varies by page template.

This means that you can measure a handful of pages using the same template and define LCP.

How to Define LCP Using Chrome Devtools

- 1.** Open the page in Chrome.
- 2.** Navigate to the Performance panel of Dev Tools
(Command + Option + I on Mac or Control + Shift + I on Windows and Linux).
- 3.** Hover over the LCP marker in the Timings section.
- 4.** The element(s) that correspond to LCP is detailed in the Related Node field.



What Causes Poor LCP?

Poor LCP typically comes from four issues:

1. Slow server response times.
2. Render-blocking JavaScript and CSS.
3. Resource load times.
4. Client-side rendering.

How to Fix Poor LCP

If the cause is slow server response time:

- Optimize your server.
- Route users to a nearby CDN.
- Cache assets.
- Serve HTML pages cache-first.
- Establish third-party connections early.

If the cause is render-blocking JavaScript and CSS:

- Minify CSS.
- Defer non-critical CSS.
- Inline critical CSS.
- Minify and compress JavaScript files.
- Defer unused JavaScript.
- Minimize unused polyfills.

If the cause is resource load times:

- Optimize and compress images.
- Preload important resources.
- Compress text files.
- Deliver different assets based on network connection (adaptive serving).
- Cache assets using a service worker.

If the cause is client-side rendering:

- Minimize critical JavaScript.
- Use another rendering strategy (Check out the breakdown of rendering options in the [Guide to Angular](#)).

Resources For Improving LCP

- [Largest Contentful Paint \(LCP\) web.dev.](#)
- [Optimize Largest Contentful Paint web.dev.](#)
- [Lighthouse Largest Contentful Paint web.dev.](#)

Total Blocking Time (TBT)

What it represents: Responsiveness to user input.

Lighthouse Performance score weighting: 25%

What it measures: TBT measures the time between First Contentful Paint and Time to Interactive. TBT is the lab equivalent of First Input Delay (FID) – the field data used in the Chrome User Experience Report and Google’s upcoming Page Experience ranking signal.

How it’s measured: The total time in which the main thread is occupied by tasks taking more than 50ms to complete. If a task takes 80ms to run, 30ms of that time will be counted toward TBT. If a task takes 45ms to run, 0ms will be added to TBT.

Is Total Blocking Time a Web Core Vital? Yes! It’s the lab data equivalent of First Input Delay (FID).

TBT Scoring

Goal: Achieve TBT score of less than 300 milliseconds.

TBT time (in milliseconds)	Color-coding
0–300	Green (fast)
301–600	Orange (moderate)
Over 600	Red (slow)

First Input Delay, the field data equivalent to TBT, has different thresholds.

FID time (in milliseconds)	Color-coding
0–100	Green (fast)
101–300	Orange (moderate)
Over 300	Red (slow)

Long Tasks & Total Blocking Time

TBT measures long tasks—those taking longer than 50ms.

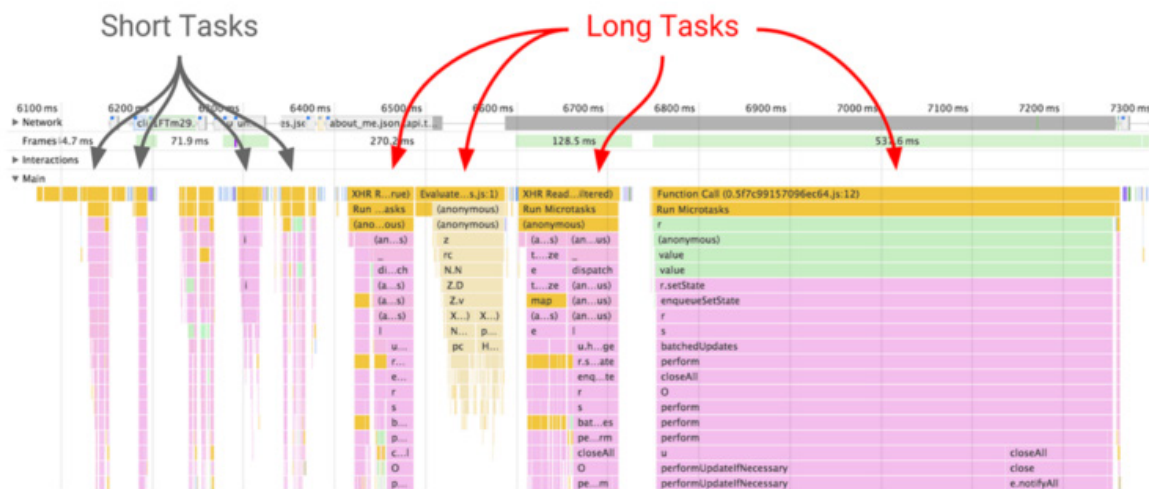
When a browser loads your site, there is essentially a single line queue of scripts waiting to be executing.

Any input from the user has to go into that same queue.

When the browser can't respond to user input because other tasks are executing, the user perceives this as lag.

Essentially, long tasks are like that person at your favorite coffee shop who takes far too long to order a drink.

Like someone ordering a 2% venti four-pump vanilla, five-pump mocha whole-fat froth, long tasks are a major source of bad experiences.

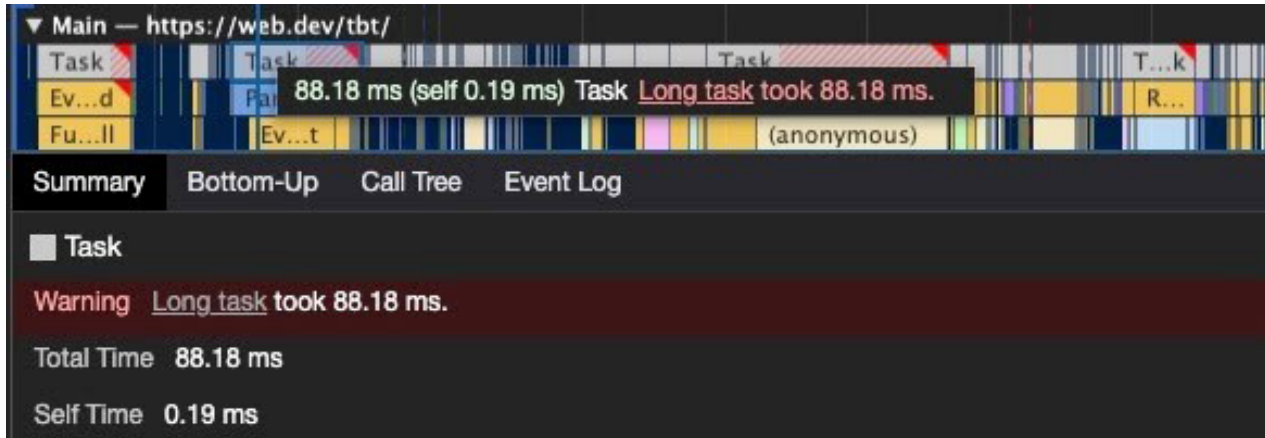


What Causes a High TBT on Your Page?

Heavy JavaScript.

That's it.

How to See TBT Using Chrome Devtools



How to Fix Poor TBT

- Break up Long Tasks.
- Optimize your page for interaction readiness.
- Use a web worker.
- Reduce JavaScript execution time.

Resources For Improving TBT

- [First Input Delay \(FID\) web.dev](https://web.dev/first-input-delay/)
- [Total Blocking Time \(TBT\) web.dev](https://web.dev/total-blocking-time/)
- [Optimize First Input Delay web.dev](https://web.dev/optimize-first-input-delay/)
- [Lighthouse: Total Blocking Time web.dev](https://web.dev/lighthouse-total-blocking-time/)

First Contentful Paint (FCP)

What it represents: FCP marks the time at which the first text or image is painted (visible).

Lighthouse Performance score weighting: 15%

What it measures: The time when I can see the page I requested is responding. My thumb can stop hovering over the back button.

How it's measured: Your FCP score in Lighthouse is measured by comparing your page's FCP to FCP times for real website data [stored by the HTTP Archive](#). Your FCP increases if it is faster than other pages in the HTTP Archive.

Is First Contentful Paint a Web Core Vital? No

FCP Scoring

Goal: Achieve FCP in < 2 seconds.

FCP time (in seconds)	Color-coding	FCP score (HTTP Archive percentile)
0-2	Green (fast)	75-100
2-4	Orange (moderate)	50-74
4	Red (slow)	0-49

What Elements Can Be Part of FCP?

The time it takes to render the first visible element to the DOM is the FCP. Anything that happens before an element that renders non-white content to the page (excluding iframes) is counted toward FCP.

Since iframes are not considered part of FCP, if they are the first content to render, FCP will continue counting until the first non-iframe content loads, but the iframe load time isn't counted toward the FCP.

The documentation around FCP also calls out that is often impacted by font load time and there are [tips for improving font loads](#).

FCP Using Chrome Devtools

1. Open the page in Chrome.
2. Navigate to the Performance panel of Dev Tools
(Command + Option + I on Mac or Control + Shift + I on Windows and Linux).
3. Click on the FCP marker in the Timings section.
4. The summary tab has a timestamp with the FCP in ms.

How to Improve FCP

In order for content to be displayed to the user, the browser must first download, parse, and process all external stylesheets it encounters before it can display or render any content to a user's screen.

The fastest way to bypass the delay of external resources is to use in-line styles for above-the-fold content.

To keep your site sustainably scalable, use an [automated tool](#) like penthouse and Apache's mod_pagespeed. These solutions will come with some restrictions to functionalities, require testing, and may not be for everyone.

Universally, we can all improve our site's time to First Contentful Paint by [reducing the scope and complexity of style calculations](#).

If a style isn't being used, remove it. You can identify unused CSS with [Chrome Dev Tool's built-in Code Coverage functionality](#).

Use better data to make better decisions.
Similar to TTI, you can [capture real user metrics for FCP using Google Analytics](#) to correlate improvements with KPIs.

Speed Index

What it represents: How much is visible at a time during load.

Lighthouse Performance score weighting: 15%

What it measures: The Speed Index is the average time at which visible parts of the page are displayed.

You'll have to ask the kindly wizards at webpagetest.org for the specifics but roughly, Speedline scores vary by the size of the viewport (read as device screen) and has an algorithm for calculating the completeness of each frame.



Is Speed Index a Web Core Vital? No

SI Scoring

Goal: achieve SI in < 4.3 seconds.

SI time (in seconds)	Color-coding	FCP score (HTTP Archive percentile)
0–4.3	Green (fast)	75–100
4.4–5.8	Orange (moderate)	50–74
5.8+	Red (slow)	0–49

How to Improve SI

Speed score reflects your site's Critical Rendering Path. A "critical" resource means that the resource is required for the first paint or is crucial to the page's core functionality.

The longer and denser the path, the slower your site will be to provide a visual page. If your path is optimized, you'll give users content faster and score higher on Speed Index.

How the Critical Path Affects Rendering



Lighthouse recommendations commonly associated with a slow Critical Rendering Path include:

- Minimize main-thread work.
- Reduce JavaScript execution time.
- Minimize Critical Requests Depth.
- Eliminate Render-Blocking Resources.
- Defer offscreen images.

Time to Interactive

What it represents: Load responsiveness; identifying where a page looks responsive but isn't yet.

Lighthouse Performance score weighting: 15%

What it measures: The time from when the page begins loading to when its main resources have loaded and are able to respond to user input.

How it's measured: TTI measures how long it takes a page to become fully interactive. A page is considered fully interactive when:

- The page displays useful content, which is measured by the [First Contentful Paint](#).
- Event handlers are registered for most visible page elements.
- The page responds to user interactions within 50 milliseconds.

Is Time to Interactive a Web Core Vital? No

TTI Scoring

Goal: achieve TTI score of less than 3.8 seconds.

TTI Score (in milliseconds)	Color-coding
0–3.8	Green (fast)
3.8 - 7.3	Orange (moderate)
7.3+	Red (slow)

Cumulative Layout Shift (CLS)

What it represents: A user's perception of a page's visual stability.

Lighthouse Performance score weighting: 5%

*Expect CLS to increase in weighting as they work the bugs out. Smart bet says Q4 2021.

What it measures: It quantifies shifting page elements through the end of page load.

How it's measured: Unlike other metrics, CLS isn't measured in time. Instead, it's a calculated metric based on the number of frames in which elements move and the total distance in pixels the elements moved.

CLS Layout Score = impact fraction * distance fraction



CLS Score (in milliseconds)	Color-coding
0–0.01	Green (good)
0.1–0.25	Orange (needs improvement)
0.25+	Red (slow)

What Elements Can Be Part of CLS?

Any visual element that appears above the fold at some point in the load.

That's right – if you're loading your footer first and then the hero content of the page, your CLS is going to hurt.

Causes of Poor CLS

- Images without dimensions.
- Ads, embeds, and iframes without dimensions.
- Dynamically injected content.
- Web Fonts causing FOIT/FOUT.
- Actions waiting for a network response before updating DOM.

How to Define CLS Using Chrome Devtools

1. Open the page in Chrome.
2. Navigate to the Performance panel of Dev Tools
(Command + Option + I on Mac or Control + Shift + I on Windows and Linux).
3. Hover and move from left to right over the screenshots of the load (make sure the screenshots checkbox is checked).
4. Watch for elements bouncing around after the first paint to identify elements causing CLS.

How to Improve CLS

Once you identify the element(s) at fault, you'll need to update them to be stable during the page load.

For example, if slow-loading ads are causing the high CLS score, you may want to use placeholder images of the same size to fill that space as the ad loads to prevent the page shifting.

Some common ways to improve CLS include:

- Always include width and height size attributes on images and video elements.
- Reserve space for ad slots (and don't collapse it).
- Avoid inserting new content above existing content.
- Take care when placing non-sticky ads near the top of the viewport.
- Preload fonts.

CLS Resources

- [Optimize Cumulative Layout Shift web.dev](#)
- [Cumulative Layout Shift \(CLS\) web.dev](#)
- [Cumulative Layout Shift \(CLS\) in AMP – The AMP Blog](#)
- [Cumulative Layout Shift \(CLS\) Calculator](#)

Conclusion

The complexity of performance metrics reflects the challenges facing all sites.

We use performance metrics as a proxy for user experience – that means factoring in some unicorns.

Tools like Google's [Test My Site](#) and [What Does My Site Cost?](#) can help you make the conversion and customer-focused arguments for why performance matters.

Hopefully, once your project has traction, these definitions will help you translate Lighthouse's single performance metric into action tickets for a skilled and collaborative engineering team.

Track your data and shout it from the rooftops. As much as Google struggles to quantify qualitative experiences, SEO professionals and devs must decode how to translate a concept into code.

Test, iterate, and share what you learn! I look forward to seeing what you're capable of, you beautiful unicorn.



Need More Insightful SEO Resources?

Visit [SearchEngineJournal.com](https://www.searchenginejournal.com)