**Engineer**

# Core Web Vitals

three measurements of how a page feels to use

## What it is

Google's web.dev documentation picks out three metrics from its wider Web Vitals programme and calls them core: Largest Contentful Paint for loading, Interaction to Next Paint for responsiveness and Cumulative Layout Shift for visual stability. Each describes something a visitor notices, applies to every page and can be measured in the field, in real visitors' browsers.

Largest Contentful Paint (LCP) is the time from the start of navigation to the moment the largest image, text block or video visible in the viewport has rendered. It is good at 2.5 seconds or less and poor above 4 seconds.

Interaction to Next Paint (INP) watches every click, tap and key press during a visit and takes the whole wait the visitor sees: the delay before the handlers start, the time they run and the time until the next frame is painted. The page's INP is the slowest of those, with one outlier dropped for every 50 interactions. Scrolling, hovering and zooming are left out. It is good at 200 milliseconds or less and poor above 500. INP replaced First Input Delay on 12 March 2024, having been introduced as an experimental metric in May 2022; First Input Delay measured only the delay before the first interaction's handlers began.

Cumulative Layout Shift (CLS) scores visible elements that move without being asked to. Each shift scores the share of the viewport the moving elements cover multiplied by how far they move. Shifts less than a second apart, up to five seconds in all, form a session window, and CLS is the total of the worst window. Shifts within 500 milliseconds of a tap, click or key press are excused; scrolling and dragging excuse nothing. It is good at 0.1 or less and poor above 0.25.

The verdict is read at the 75th percentile of page loads, separately for mobile and desktop, and a page passes only when all three are good there. A “good” line was accepted only if at least a tenth of origins already met it.

Underneath are browser interfaces with their own specifications, the Largest Contentful Paint, Layout Instability and Event Timing APIs, and a few lines show the raw material of CLS on any page:

```
new PerformanceObserver((list) => {
    for (const shift of list.getEntries()) {
        if (!shift.hadRecentInput) console.log(shift.value, shift.sources);
    }
}).observe({ type: "layout-shift", buffered: true });
```

That logs individual shifts, not the metric; for the three as Google’s tools report them, web.dev points to the `web-vitals` library.

The term is easy to confuse with the Lighthouse performance score, which comes from one simulated load in a lab, with no visitor in it; the Core Web Vitals assessment describes real visits at the 75th percentile, and a page can do well on one and badly on the other.

Use the concept in a decision

Put “What is Core Web Vitals?” to work in a real decision. We can help relate the concept to the system you are building or buying. Interfaces that load quickly, work on a phone and behave the same in every browser your customers happen to use.

**Explore our services** Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Directed full-stack GIS map development — full-stack mapping, from database to browser

**Directed full-stack GIS map development** Get in touch →

## Why it matters

The definitions leave open where the numbers come from, and that is the decision a site has to make: steer by lab estimates, or pay for field data.

Field data is what the thresholds are written for. The Chrome User Experience Report collects anonymised measurements from real Chrome users and feeds PageSpeed Insights and Search Console’s Core Web Vitals report without the site

adding anything, but only once a page or origin has enough visitors. The alternative is a script in the visitor's browser that sends every reading to a server: a privacy decision, and a cost carried on every page view.

Lab data costs neither. Lighthouse loads the page on a simulated device and network, before a change ships and as often as wanted. What it measures is a visit nobody made. It sees LCP and CLS during the load, but with nobody pressing anything it has no INP; web.dev offers Total Blocking Time as the closest lab measure and says it is no substitute.

The workable split is both, held apart: lab numbers as budgets that catch a change for the worse, field numbers, where they exist, as the only verdict on what visitors get.

## What it looks like in practice

A planning record shows a Core Web Vitals job was considered and declined: the Chrome UX Report needs an API key and enough traffic to report, and the pages are already held to size budgets in the quality gate. The site gathers no browser-side vitals itself: its credits page says it ships no executable JavaScript or browser analytics code. There is no site-supplied script to hold up a click or key press, although layout and rendering can still delay interaction. The same page points readers to PageSpeed Insights, which shows measurements from real visits once a page has enough of them.

What the site measures instead is the lab. Its Lighthouse configuration loads a fixed list of pages three times each under simulated mobile throttling and reports a failure when one of the English or Danish pages crosses any of these lines: LCP over 4,200 milliseconds, First Contentful Paint over 1,600, CLS over 0.02 and Total Blocking Time over 100, with slightly wider paint limits on the Ukrainian pages. These are budgets, not the field thresholds, written after the run-to-run spread had been measured at 299 milliseconds at worst and 80 at best, because a line inside that spread passes or fails by chance. The project's notes put most of the LCP down to render delay behind one render-blocking stylesheet. Weight that arrives before any content had already been cut: the stylesheet bundle weighed 87 KB with a base64 font inside it and 31 without. The runs take minutes, so they happen on demand, outside the gate.

The CLS budget, five times stricter than the 0.1 line, caught a case worth keeping. One category page failed at 0.107. The owner opened it, saw nothing wrong and said so. Lighthouse had named the parallax gears as the culprit, and the first written diagnosis believed it. A controlled test with a real wheel gesture scored 0.2477 whether the gears moved or not, while the heading and the last paragraph moved by zero pixels throughout. The cause was the footer: it is fixed to the viewport and changes height as its colophon opens, and the metric scores that box against the viewport although a fixed element cannot push anything. The budget stayed where it was; if the number is ever silenced, the plan is an exception for that one address.

## Where it goes wrong

**A lab report leaves INP out and the page is assumed responsive.** Lighthouse shows LCP and CLS, Total Blocking Time is low, and slow clicks in production come as a surprise. A simulated load has nobody interacting, and Total Blocking Time only counts main-thread blocking during that load. Read it as a warning light, time real interactions by hand in the browser's performance tools, and judge INP from field data where there is any.

**The element a tool blames turns out to be innocent.** The report names a node for a layout shift, changing that node moves the score by nothing, and the text on screen stays put. Attribution names nodes, not causes; the shift may belong to a fixed overlay that changes height, which counts as movement and is not excused by scrolling. Switch suspects off one at a time, measure whether content moves, and animate size with `transform`, which web.dev recommends because it causes no layout shift.

**Content jumps down as images and late arrivals load.** Readers lose their place and field CLS climbs past 0.1. web.dev lists the usual causes: images and video of unknown size, web fonts that render larger or smaller than the fallback, and widgets that resize themselves. Give every image `width` and `height` attributes so the browser reserves its box, as this site's image partials do, and reserve space for anything inserted later.

**A healthy average hides a failing assessment.** A dashboard shows a mean LCP well under 2.5 seconds, yet the page is rated as needing improvement. The assessment uses the 75th percentile, per device class, and a slow quarter of mobile visits disappears inside a mean that includes fast desktops. Report and target the 75th percentile, split by device.

# Further reading

- Web Vitals — the three metrics, their thresholds and the tools that report them.
- Defining the Core Web Vitals metrics thresholds — why the 75th percentile, and how each line was tested.
- Interaction to Next Paint — what counts as an interaction and how INP differs from First Input Delay.
- Largest Contentful Paint API — which elements are candidates and when their time is taken.
- Layout Instability API — the layout shift score, recent input and the sources of each shift.
- Event Timing API — how interaction latency is exposed to scripts.

Glossary

Choose for your actual constraints

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

**See what it costs** Get in touch →

## Make the terminology practical

Ask how “What is Core Web Vitals?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

**Explore our services**

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/core-web-vitals/