Engineer

Lighthouse

Chrome's own audit of speed, markup and good practice

What it is

Lighthouse, often called Google Lighthouse, is an open-source program from the Chrome team that loads a single web page under controlled conditions, records what the browser did, and grades the result out of 100 in four categories: performance, accessibility, best practices and SEO. It works on any page it can reach, public or behind a login, and every check in its report links to a short document explaining why the check exists and how to pass it.

The same engine runs in four places:

- in Chrome DevTools, from the Lighthouse panel;
- on the command line, as `lighthouse <url>` after a global npm install;
- as a Node module, inside a script of your own;
- behind PageSpeed Insights, a Google service that runs it on Google's machines.

The performance score comes only from metrics measured during the load: First Contentful Paint, Speed Index, Largest Contentful Paint, Total Blocking Time and Cumulative Layout Shift. Each metric is placed on a curve drawn from real websites in the HTTP Archive, and the category score is a weighted average of the five. In Lighthouse 10 the weights are 30% for Total Blocking Time, 25% each for Largest Contentful Paint and Cumulative Layout Shift, and 10% each for the other two. They have changed between versions, because the team keeps researching what most affects how fast a page feels.

The curve has two fixed points. A metric value at the 25th percentile of the archive scores 50, and one at the 8th percentile scores 90. Lower is better for every one of these metrics, so a page that scores 50 on one of them is already

ahead of three quarters of the sites measured; 50 is not average. Near the top the curve flattens: going from 99 to 100 takes about as much improvement as going from 90 to 94.

The other three categories are scored from audits instead. In accessibility, each audit passes or fails with no credit for a partial pass, each is weighted by the axe rule engine's assessment of how hard a failure hits users, and the checks that need a person are left out of the score and listed for manual review.

Use the concept in a decision

Put "What is Lighthouse?" 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 decision Lighthouse serves is whether a change made a page slower, harder to use or harder to find, and whether that should stop it shipping. It answers in a form a machine can act on: a number, the same audits every time and a report a script can parse. That makes it cheap enough to run on every change rather than once before a launch.

The number is a laboratory measurement: one device profile, one simulated network, one load. PageSpeed Insights shows the difference by putting two kinds of data side by side. Its lab section is a Lighthouse run. Its field section comes from the Chrome UX Report: what real Chrome users experienced over the previous 28 days, given at the 75th percentile so that the figure describes most visitors rather than the luckiest. A page with too few visits of its own is shown with figures for its whole origin. The documentation says to expect the two sections to differ.

That split decides what each is for. Field data says whether users have a problem, but it arrives weeks after a change and says nothing about a branch that has not shipped. Lab data can be collected before release and as often as needed, which makes it a guard, but it describes one simulated visit.

The other limit is noise. The scoring guide suggests reading performance “as a distribution of scores, rather than a single number”, and lists causes outside the page: A/B tests and rotating adverts, changes in internet routing, a fast desktop against a slow laptop, browser extensions and antivirus software. A threshold checked against one run will fail at random. The project’s variability guide gives the remedy a figure: the median of five runs is twice as stable as a single run.

Lighthouse CI turns that into a rule. It runs Lighthouse several times per address, three by default, combines the runs and checks the result against assertions for a whole category or a single audit. An assertion at `warn` only reports; one at `error` fails the build. Its threshold is a minimum score, a maximum value for a metric or a maximum number of offending items:

```
{
    "ci": {
        "collect": { "numberOfRuns": 5 },
        "assert": {
            "assertions": {
                "categories:accessibility": ["error", { "minScore": 1 }],
                "largest-contentful-paint": [
                    "error",
                    { "maxNumericValue": 2500, "aggregationMethod": "median" }
                ]
            }
        }
    }
}
```

What it looks like in practice

This site’s credits page lists Lighthouse among the tools the site runs against itself, on the machine it is written on, and gives it one job: it holds the performance budgets. A budget is a limit agreed in advance, so that a page that gets slower fails a check instead of starting an argument about whether a few hundred milliseconds matter.

A later section of the same page links PageSpeed Insights among checks readers can run for themselves, and says it runs Lighthouse from Google’s machines rather than the site’s, with real-world data where enough people have visited. The page keeps the two apart on purpose. The first section is the site marking its own homework; the second hands the same questions to tools that are not the site’s own and take nothing on its word.

The habit that makes a budget worth holding also shows in work here that has nothing to do with a score. The account of how the stylesheet bundle was cut from 87 KB to 31 opens its method with “Measurement first.” Two thirds of the largest stylesheet turned out to be one base64-encoded font; a third font weight was preloaded only when field data showed it closing the longest critical chain; and a modern image format for the brand mark was measured and declined, because it came out larger. Every decision there has a measurement attached, including the one that went the other way. Lighthouse’s suggestions need the same treatment: each is a hypothesis until a measurement says it helped.

Where it goes wrong

Most Lighthouse mistakes come from reading the number as more than it is.

A single run is read as the truth. One report shows 94, the next 87, and someone hunts a regression that never happened. Compare medians of several runs, and set the aggregation in Lighthouse CI on purpose: an assertion written without options takes the value most likely to pass across all runs, so a regression gets through as long as one run escapes it.

Runs are collected side by side. Auditing ten addresses at once on one machine makes them compete for processor, memory and network, and the variability guide says plainly not to do it. Run them in turn, or across machines; for Lighthouse, more small machines beat one large one.

The lab score is read as what visitors get. A 98 on a developer’s laptop says the page can be fast; only field data says whether it was. Read the field section in PageSpeed Insights, and expect a page with few visits to show its origin’s figures.

The suggestions are treated as the score. Opportunities and Diagnostics do not count towards the performance score; they move it only through the metrics they improve. Pick the one that should move a metric that matters for this page, make the change, and measure again.

A perfect score becomes the target. The scoring guide calls 100 extremely challenging and says it is not expected. Once a page is in the green band, the last points cost more than they return.

A full accessibility score is read as an accessible page. A 100 there means the automated audits passed. The manual checks are listed beside the score and never counted, so a page can score 100 and still be impossible to use with a keyboard.

Further reading

- Introduction to Lighthouse — the ways to run it, the categories it reports and where each audit's documentation lives.
- Lighthouse performance scoring — the metric weights by version, the scoring curves and the colour bands.
- Lighthouse accessibility score — how audits are weighted and why manual checks stay out of the score.
- Score Variability — the sources of noise, what reduces each one, and the case for running several times.
- About PageSpeed Insights — lab against field data, the 75th percentile and the Core Web Vitals assessment.
- Lighthouse CI configuration — collecting several runs, writing assertions and choosing how runs are combined.

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 Lighthouse?” 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/lighthouse/