Page Speed

Refresher and Update

adtrak.

2017	Page speed training and initial Resource Centre guides
2018	Major PageSpeed Insights update in Nov - now powered by Lighthouse
2019	Detailed Adtrak blog post in Jan and new RC guides based on PSI update
2020	Another big PSI update in May - new metrics and Core Web Vitals

Why is it so important?

Mobile web usage is ever increasing

Mobile connections are less reliable and depend on your location, which can often be rapidly changing if on public transport etc.

Mobile hardware is also much less powerful than desktop hardware, meaning they can’t process your site’s resources as quickly

It’s a ranking factor

Page speed can now have a direct effect on Google rankings

Refresher

Gzip Compression

Browser Caching

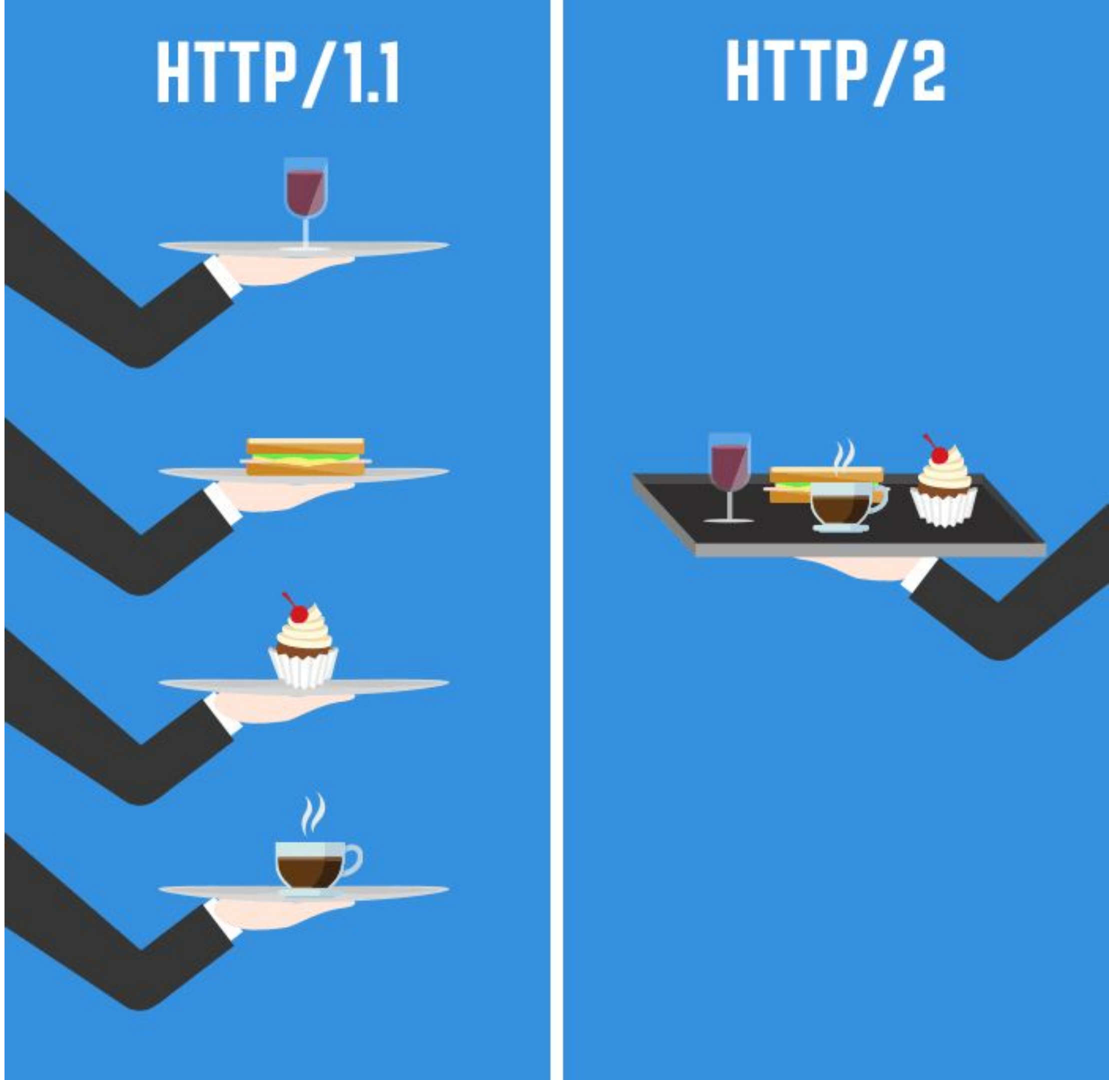
Enabled automatically on our servers

Minify CSS & JS

Concatenate CSS and JS?

Not something we need to worry too much about anymore

Combining small files as we generally do is fine, but probably better to keep big stuff separate



If you have got big stuff… do you need it?

mmenu.js

carousel plugins

modal popup plugins

slideshow plugins

font awesome

Streamline your critical path

to render your “above-the-fold” portion of the page as quickly as possible

Streamline critical path

1) Prioritise CSS required to render the visible part of your page

2) Make sure we're not prioritising resources that aren't required to render the visible part of the page

AKA...

1) Inline your critical CSS in `<head>`

2) Defer all JS and non-critical CSS

Critical CSS

Previously had all CSS in partials and we'd choose which ones to include in our critical.css and which to include in the main.css

critical.css would be added in `<head>`

main.css loading would be deferred via `loadCSS()` function

Tailwind

Don't really work in partials anymore

Also means much smaller final CSS files

Usually small enough to just inline main.css in <head>

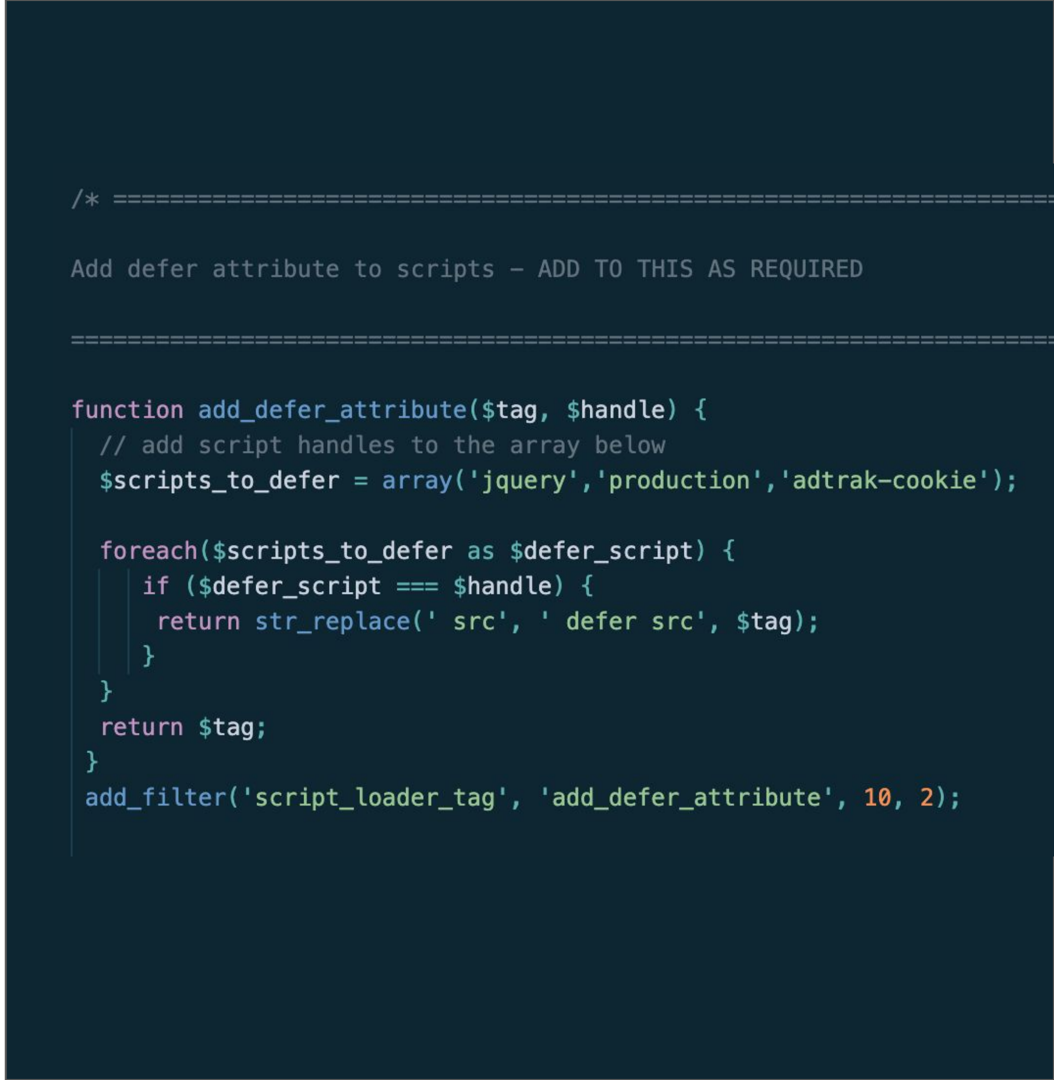
Defer JS

Load scripts in the footer

Ideally add the `defer` attribute to ensure the browser doesn't try to load scripts too early

Use with caution

(mainly if deferring jQuery)



```
/* ==================================================================

Add defer attribute to scripts - ADD TO THIS AS REQUIRED

====================================================================

function add_defer_attribute($tag, $handle) {
  // add script handles to the array below
  $scripts_to_defer = array('jquery','production','adtrak-cookie');

  foreach($scripts_to_defer as $defer_script) {
     if ($defer_script === $handle) {
      return str_replace(' src', ' defer src', $tag);
     }
  }
  return $tag;
 }
 add_filter('script_loader_tag', 'add_defer_attribute', 10, 2);
```

Optimise Images

Compress images as much as reasonably possible

Tiny PNG

Smush

Gulp

etc.

Serve images in NextGen Formats

<picture> for non WP images

WebP Express

Smush

ShortPixel

etc.

Properly Size Images

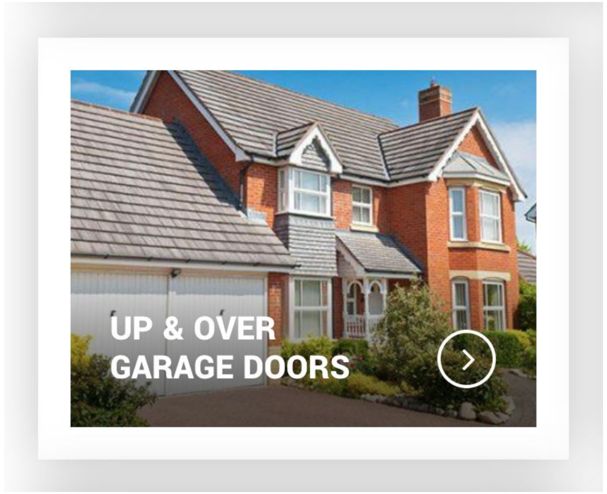


Image size on screen: 300x250

Actual image size: 350x300

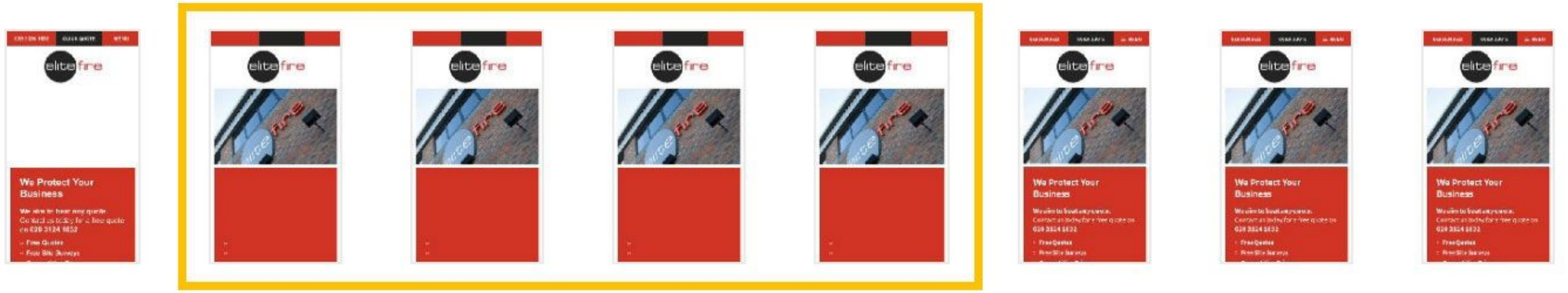
Actual image size: 2000x1500

Make use of WordPress image sizes and Twig image resize function

Web Fonts

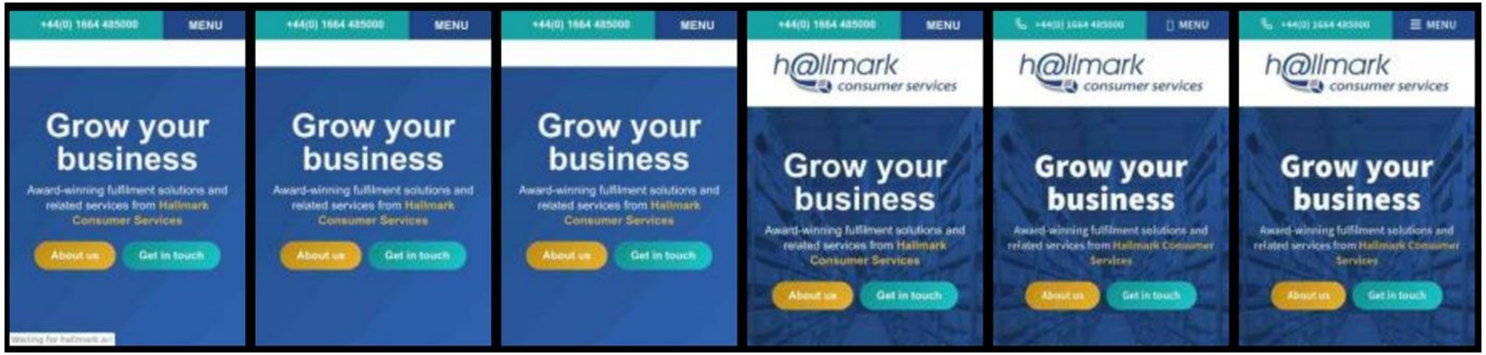
“Ensure text remains visible during webfont load”

This warning refers to the “**Flash Of Invisible Text**” (FOIT) that occurs when your web font loads



Can eliminate this by adding `font-display:swap;` to your web font CSS

This swaps your FOIT for a FOUT experience (**Flash Of Unstyled Text**) which Google prefers


+44(0) 1664 485000
MENU
h@llmark consumer services
Grow your business
Award-winning fulfilment solutions and related services from Hallmark Consumer Services
About us
Get in touch

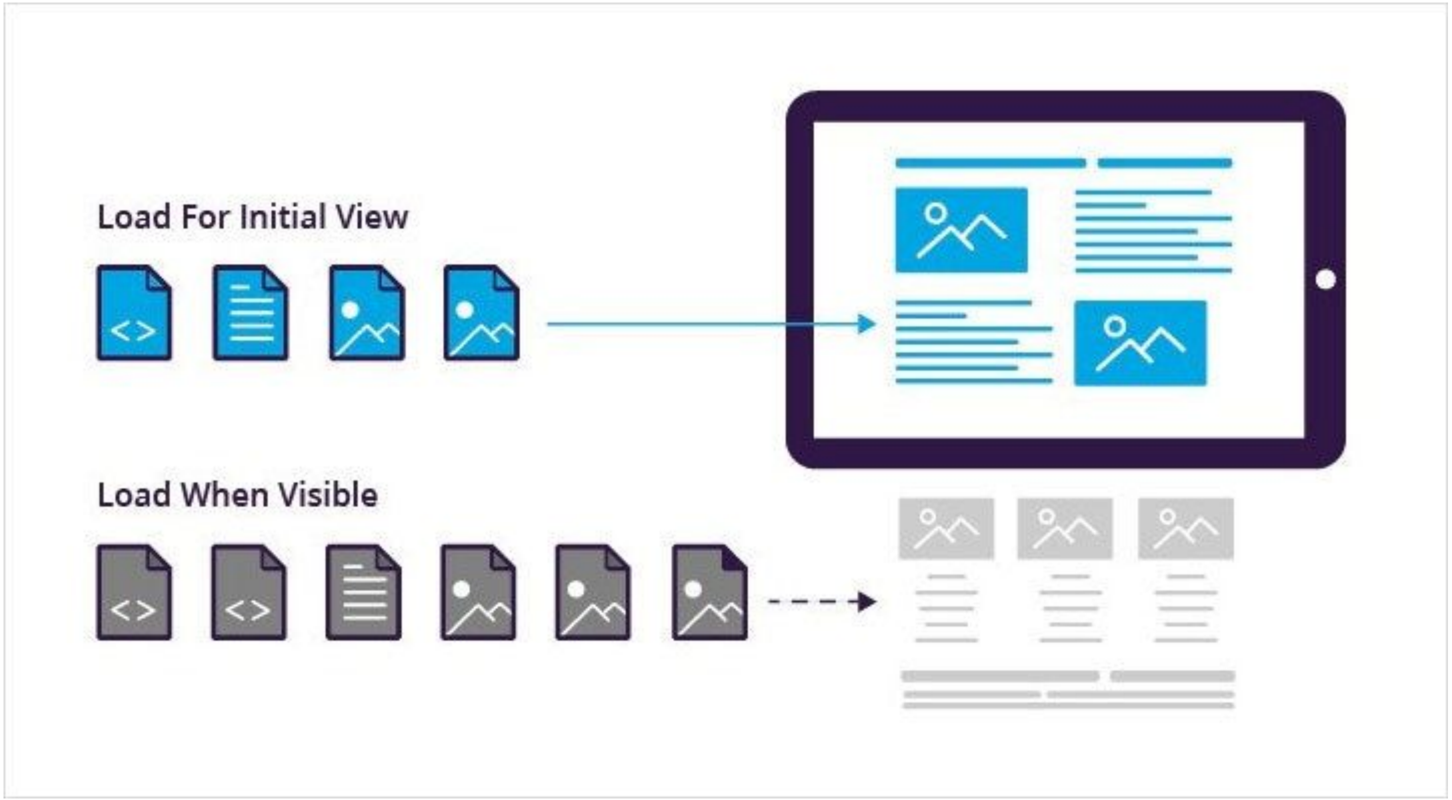
Easy to add `font-display:swap;` if your fonts are local

Google Fonts - can now add `&display=swap` to the Google Font link

Typekit/Adobe - no way to add :-(

FontAwesome will also trigger the warning

Lazy Loading


Load For Initial View
Load When Visible

Lazysizes

Include lazysizes.min.js
and a few lines of CSS
and away you go

```
<script src="path-to-js/lazysizes.min.js" async=""></script>
```

```
.lazyload,
.lazyloading {
    opacity: 0;
    transition: .2s;
}

.lazyloaded {
    opacity: 1;
}
```

```
<img class="lazyload" data-src="path-to-img.jpg" alt="Img alt text">
<iframe class="lazyload" data-src="https://www.google.com/maps/embed.....
```

Lazy loading iframes - biggest page speed gains

- Google Maps
- Scribble Maps
- Social feeds
- YouTube / Vimeo embeds
- Wistia embeds
- etc.

Wistia Videos

Use their iframe embed code

Remove the EV-1.js script

If you're using their "popover" code, fairly straightforward to replicate with the iframe code and custom modal

Other Stuff

Deregister scripts and stylesheets that aren’t needed

- jQuery Migrate
- Gutenburg (block-library/style.min.css)
- NF / CF7 stylesheets

Caching plugin

- Turns your dynamic PHP pages into static HTML pages
- WP Super Cache

Testing Tips

View Source - search for.js and .css

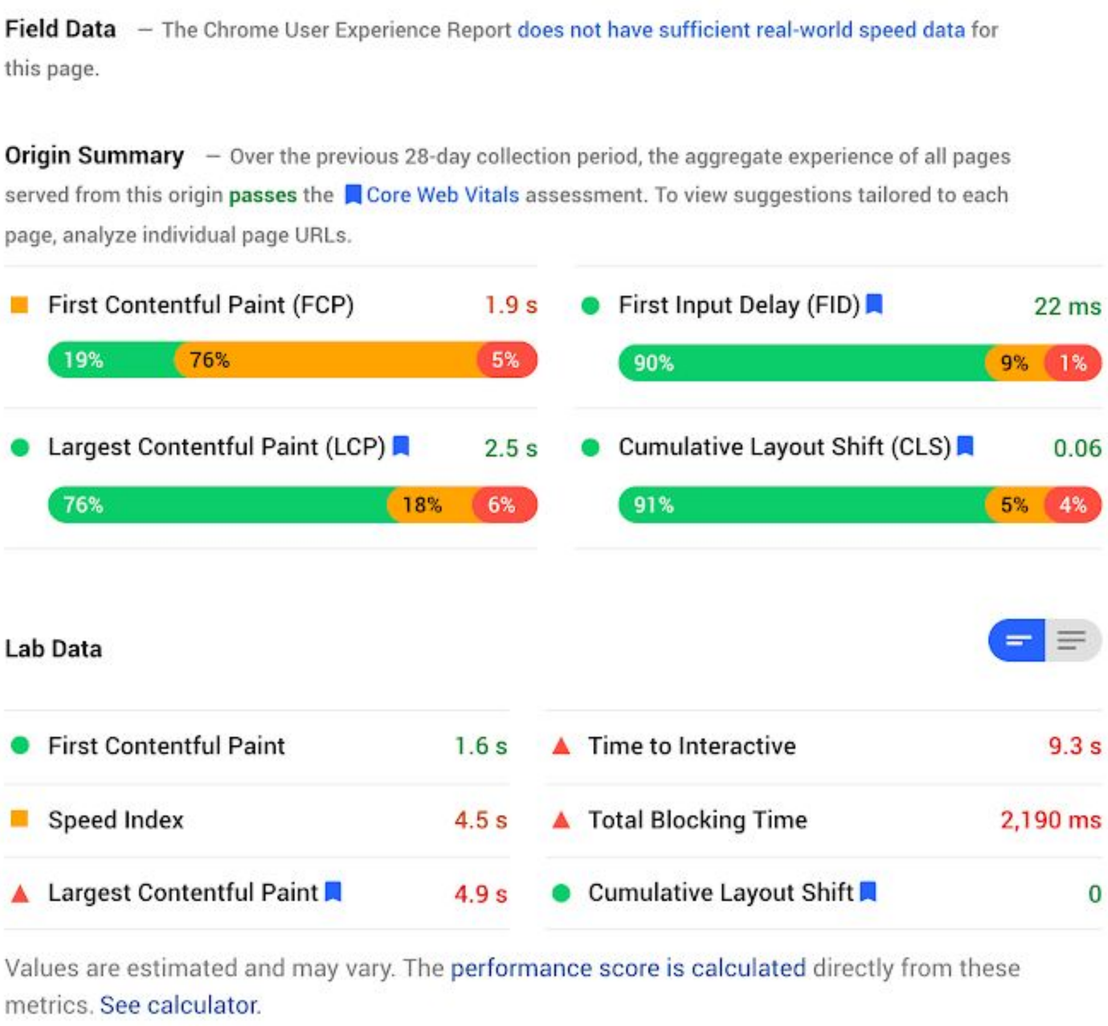
Delete stuff and re-test

Questions?

PageSpeed Insights

Real world data from previous 28 days

Test data from a simulation - Moto G4 on fast 3G / slow 4G


Field Data – The Chrome User Experience Report does not have sufficient real-world speed data for this page.

Origin Summary – Over the previous 28-day collection period, the aggregate experience of all pages served from this origin passes the Core Web Vitals assessment. To view suggestions tailored to each page, analyze individual page URLs.

Metric	Value	Good	Needs improvement	Poor
First Contentful Paint (FCP)	1.9 s	19%	76%	5%
First Input Delay (FID)	22 ms	90%	9%	1%
Largest Contentful Paint (LCP)	2.5 s	76%	18%	6%
Cumulative Layout Shift (CLS)	0.06	91%	5%	4%

Lab Data

First Contentful Paint	1.6 s	Time to Interactive	9.3 s
Speed Index	4.5 s	Total Blocking Time	2,190 ms
Largest Contentful Paint	4.9 s	Cumulative Layout Shift	0

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

First Contentful Paint

When first text/image is painted

First Input Delay

Time between user's first interaction and the site responding to that interaction

Largest Contentful Paint

When largest element in viewport has loaded

Cumulative Layout Shift

The amount the layout shifts during page load

Time to Interactive

Time it takes for page to become fully interactive

Speed Index

How quickly the contents of the page are populated

Total Blocking Time

Time between First Contentful Paint and Time to Interactive

Core Web Vitals

May 2020 Update

Largest Contentful Paint

When largest element in viewport has loaded

First Input Delay

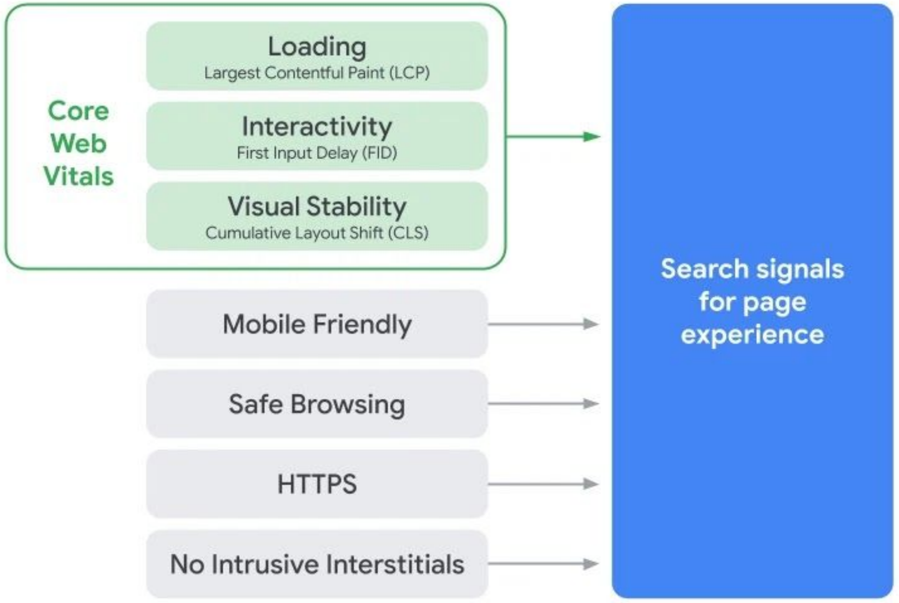
Time between user's first interaction and the site responding to that interaction

Cumulative Layout Shift

The amount the layout shifts during page load

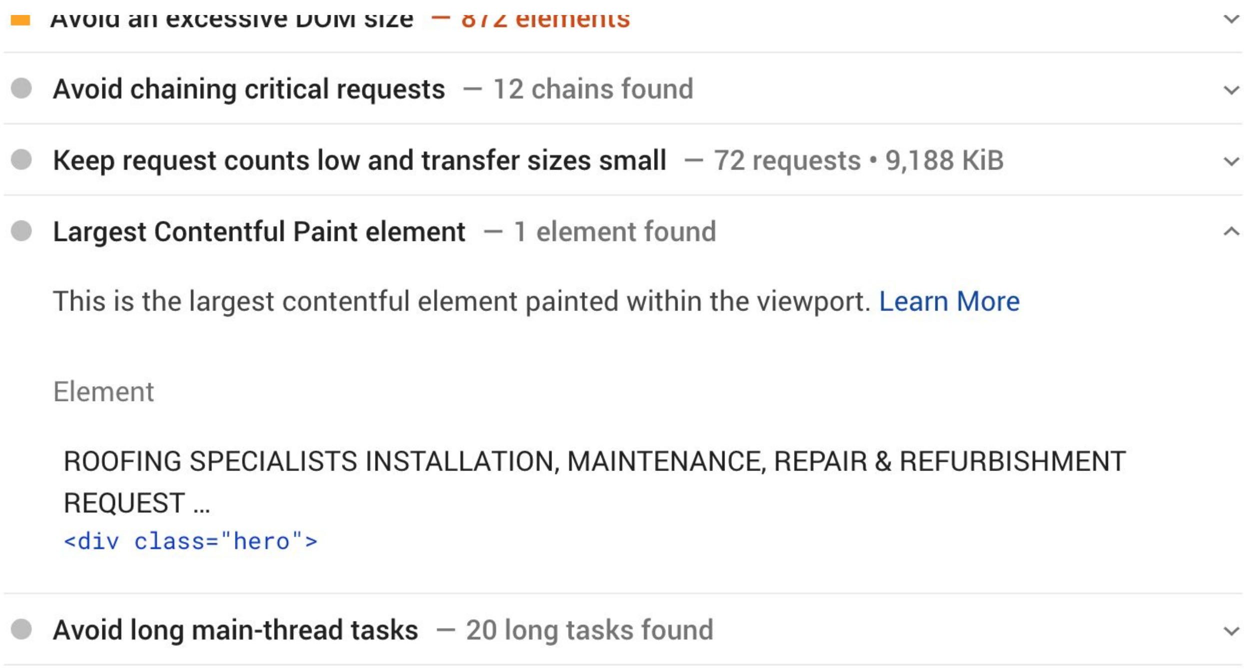
Defined by Google as "Core Web Vitals" meaning they will be factorec into Google's new "Page Experience' ranking signal

Not in effect yet - to be launched next year according to Google


Core Web Vitals
Loading
Largest Contentful Paint (LCP)
Interactivity
First Input Delay (FID)
Visual Stability
Cumulative Layout Shift (CLS)
Mobile Friendly
Safe Browsing
HTTPS
No Intrusive Interstitials
Search signals for page experience

Largest Contentful Paint

When largest element in viewport
has loaded



Avoid an excessive DOM size — 872 elements

● Avoid chaining critical requests — 12 chains found

● Keep request counts low and transfer sizes small — 72 requests • 9,188 KiB

● Largest Contentful Paint element — 1 element found

This is the largest contentful element painted within the viewport. Learn More

Element

ROOFING SPECIALISTS INSTALLATION, MAINTENANCE, REPAIR & REFURBISHMENT REQUEST ...
`<div class="hero">`

● Avoid long main-thread tasks — 20 long tasks found

Largest Contentful Paint

When largest element in viewport has loaded

How to Optimise

- Same principles as optimising for First Meaningful Paint
- Prioritise critical resources
- Don’t lazy load anything above the fold
- Don’t append hero <video> elements via JS

First Input Delay

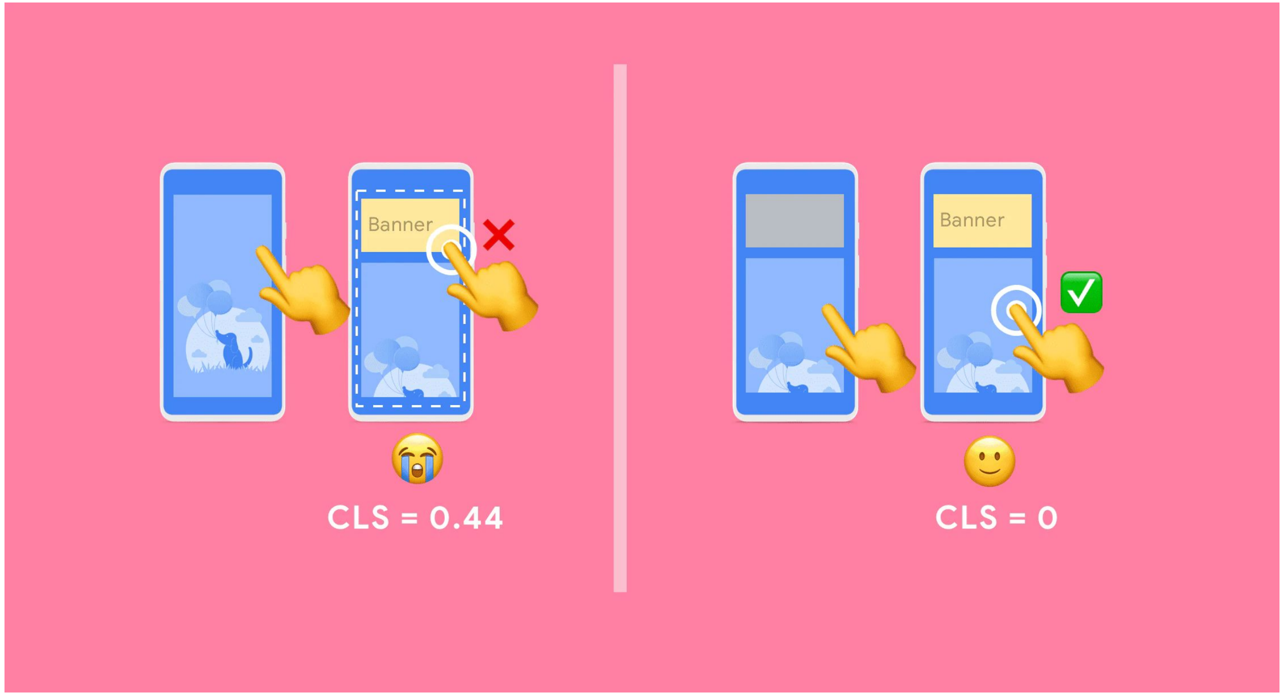
Time between user's first interaction and the site responding to that interaction

How to Optimise

- Not available in the Lab Data (only Field) because there is no first user interaction in the lab test - **Total Blocking Time** is a good indicator
- All about JavaScript - will directly correlate to amount of JS the browser has to parse
- Highlights importance of lazy loading 3rd party embeds that include their own scripts

Cumulative Layout Shift

The amount the layout shifts during page load



Cumulative Layout Shift

The amount the layout shifts during page load

How to Optimise

- Elements that load in and cause the page to jump around
- Web fonts that are very different to the fallback
- In dev tools, change your connection to Slow 3G and watch
- If images cause layout shift, try to reserve their space

TL;DR

- Lazy load all non critical images and iframes
- Do whatever you can to minimise impact of 3rd party stuff
- Inline critical CSS in `<head>`
- Include scripts in footer with `defer` attribute
- Clean up your JS - limit usage of heavy jQuery plugins
- Compress images and videos as much as is reasonable
- Ensure hero video isn’t being added via JS
- Serve WebP images via plugin
- Use `font-display: swap;` to prevent FOIT
- Remove unnecessary fonts / weights

Questions?