Critical CSS and Webpack: Automatically Minimize Render-Blocking CSS

BY ANTHONY GERA ON JULY 21, 2017 | 0 COMMENT

"Eliminate render-blocking JavaScript and CSS"—Yo Yo one Google Page Speed Insights suggestion that always get stuck with.

What a web page is accessed, Google wants it to only load what's useful for the initial view, and use little time to load anything else. That way, the user can see the page as early as possible.

Mobile | Desktop

Speed

Should Fix:

Eliminate render-blocking JavaScript and CSS in above-the-fold content

There are many things we can do to minimize render-blocking JavaScript, e.g. code splitting, tree-shaking, caching and so on.

But what about CSS? For this, we can minimize render-blocking by inlining the CSS needed for above-the-fold content (a.k.a. the critical CSS) and loading that first. We can then load the non-critical CSS afterwards.

Inlining critical CSS is something that can be done programmatically, and in this article I'll show you how to accomplish it in your Webpack projects.

What does render-blocking mean?

If a resource is "render-blocking", it means the browser can't display the page until the resource is downloaded or otherwise dealt with.

Typically, we will load our CSS in a render-blocking way by linking to our stylesheet in the head of the document, like this:

```
<head>
  <link rel="stylesheet" href="style.min.c
  ...
</head>
<body>
  <p>Hi, I'm a lame static original site from Internet 1.0</p>
</body>
```

When this page is loaded to a web browser, it will wait if from top to bottom. When the browser gets to the Line tag, it will start downloading the stylesheet straight away, and will not render the page until it's finished.

For a large site, particularly one with a generously sized framework like Bootstrap, this stylesheet might be several hundred kilobytes, and the user will have to patiently wait until the fully downloads.

So, should we just link to the stylesheet in the header, where rendering is not blocked? No, could, but the thing is render-blocking is not entirely bad, and we actually want to exploit it. We want the prioritized retrieval of our CSS loaded, we'd just like it to load *quicker* instead.

Page Speed
Jumbotron heading
Sign up free today
Sign up today
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer ac ante ac ante ac ante