Optimizing the GitHub Homepage

Braden Huttash

<https://github.com/>

I chose to inspect the GitHub homepage for the Prove assignment write-up. I used Lighthouse to display page performance specs, and it already achieved a great score of 96%. There was only one audit in the Opportunities list, with a Needs Improvement rating, and then there was another in the Diagnostics list, as well as two audits with a Poor rating.

The only Metric that wasn’t in the green was a fast orange for Largest Contentful Paint, which is the time it takes for the largest text or image on the page to render. Oddly enough, the recommended page on LCP has a table that suggests GitHub’s 1.2-second LCP should have put it in the green, as it is still rather fast, and I’m not sure messing with the CSS would improve anything, so I didn’t focus on it.

The opportunity presented for the page suggested eliminating render-blocking resources. The resource in question was the CSS file for a light theme for the webpage, which was not in use, so I’m not sure it really needs fixed. However, the style could be deferred so it ceases to block rendering, as it is non-critical due to not being in use.

As for the Diagnostics list, two of the audits here had nothing to do with CSS. The first Poor audit was because the page prevents back/forward cache restoration, with 4 failure reasons. Only two were “Actionable,” meaning that they could be fixed, but the other two “Not actionable” reasons cannot be fixed for the page because something outside the page’s control prevents caching. So there wasn’t much that could be done to pass this audit, nor does it have to do with CSS. Secondly, the Needs Improvement audit in Diagnostics also was a caching issue, dealing with images and user avatars.

The second Poor audit was a result of an excessive DOM size. The GitHub homepage has nearly 1700 Total DOM Elements. This one also doesn’t have a lot to do with CSS, but one potential method of improving the performance in this area is simplifying CSS selectors for the page. Some suggestions for this include reducing the complexity of selectors and/or reducing the number of elements being styled. Another great approach is using the BEM methodology for naming and organizing the classes of the page’s elements.