What's Missing From the Web

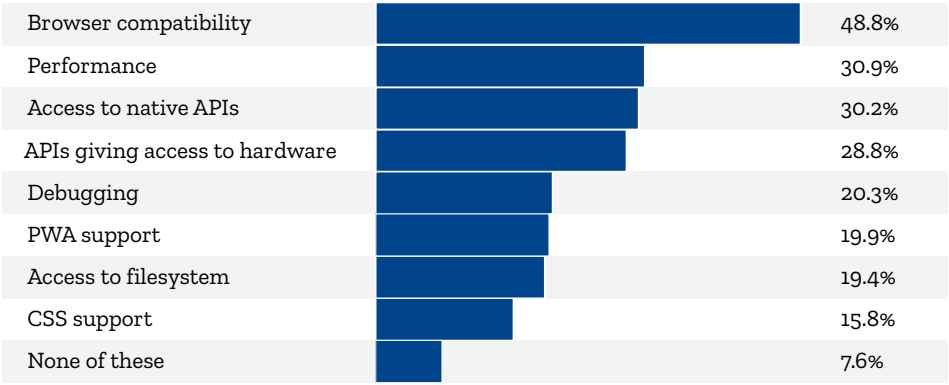
What's Missing From the Web

In 2019, we included the open-ended, "What are things that you would like to be able to do on the Web but lack web platform features to do?" This was an optional question, not requiring a response.

Last year, we hand categorized the answers. We took a random 1,000 answers and manually categorized them into 109 categories, up to three categories per answer. We used the first or most prominent issue as the first category. Of the 109 categories, only seven had 3% or more of the answers:

- Access to Hardware (12.4%)
- Browser Compatibility (8.6%)
- Access to Filesystem (4.7%)
- Performance (3.4%)
- PWA support (3.4%)
- Debugging (3.3%)
- Access to Native APIs (3%)

We added a new question this year that built upon what we learned from the open-ended answers from last year. We increased the list above by adding CSS support, which accounted for 2.9% of the categorized answers from last year. The question was, "Which do you feel are most lacking from the web platform?" Respondents were allowed to select up to three. Browser Compatibility was the most selected option at 48.8% of the responses.



n = 6645

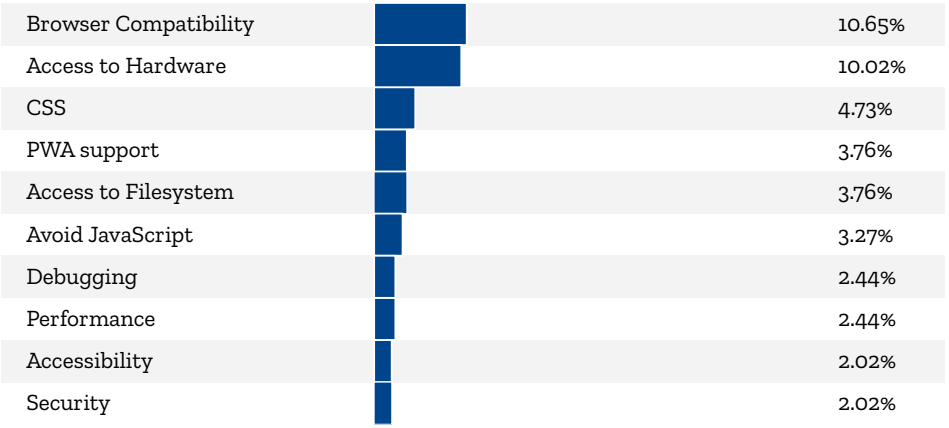
What's Missing From the Web

Following the above question, we added an open-ended question, "What other things would you like to be able to do on the Web but lack web platform features to do?" This is a slight word change between 2019 and 2020 with the phrase, 'what other things,' being key. However, that nuance may have been lost on respondents as evidenced by the analysis of their responses or they reiterated how important an option was to them.

To analyze the open-ended responses from this year, we employed natural language processing techniques. Specifically, we trained a decision-tree-based model on the 2019 answers and categories. The caveats are, it only works on a limited number of categories, and it can only assign one category per response. On the 2019 data, it predicted one of the three categories 92% of the time.

Open-ended questions are difficult to analyze because you cannot be sure how a respondent interpreted the question, and therefore what context to apply to their answer. With that in mind, we went through the responses and eliminated answers that did not answer the question at hand. For example, many responses had some form of, 'non-applicable,' or, 'nothing is missing.' After deleting non-pertinent answers, the remaining responses totaled 1,437.

The model used 60 categories. However, of those categories and removing 'Other,' only ten had 2% or more of the answers.



n = 648

The results of the open-ended responses match up pretty well with the options provided in the question before the open-ended question, with accessibility and security bubbling up as strong contenders for what's missing from the web. Select verbatims that help convey the deeper meaning of the category are on the following pages.

What's Missing From the Web

Browser Compatibility

"Browser cross-compatibility and extensibility in secure configurations under which plugins are created for platforms."

"Fully cross-browser compatibility. No 'implementation-specific' points in standards or drafts."

"I'd love to write some HTML/CSS and not have it behave differently in six months when the specs/browsers change yet again."

Access to Hardware

"Hardware/Native API. We do a lot of automation to support doctors, think automatically position windows across multiple screens among other things. We currently have to install a desktop app that the website can talk to make this work well. That combined with dictation software creates a barrier between us and the doctors."

It's worth noting that the following quote is a bit of an outlier in that they are asking for something many web developers are asking for, however, there's a lack of trust that browser vendors will implement it fairly.

"As a developer, I'd like native and hardware API access in order to build a wider range of software on the web platform. But the certainty that such APIs would instantly be abused, and that browser vendors (looking at you, Google) would fail to provide sufficiently strong and user-comprehensible security and privacy controls, leads me to hope those APIs never ship."

"User-allowed access to hardware."

CSS

"Tons of CSS things.. better attr(), better calc()..."

"Support more features only available via CSS preprocessors (e.g. SCSS), e.g. loops, conditions, macros, more functions."

"Support for CSS across browsers is super important to me -- I am one of the only developers in my office who works on CSS but it has the biggest impact on what our clients care about (branding, etc)."

Access to Filesystem

"I clicked Access to filesystem, but to emphasize it, I put it here, too."

"Full access to a folder for media management. Persistent state storage on the filesystem, not subject to the whims of browser storage cleanup."

"Having access to a virtual filesystem without all of the extra steps and external libraries normally associated with it."

PWA Support

"I would love to make a webapp, that Chrome/Chromium doesn't break with each third release, because they hate users so hard. Other than that: PWA on desktop, not mobile."

"The problem is that PWA doesn't work in Safari, and there's no way to run chrome on iOS, so we are blocked on iOS."

"Bundle a PWA in a single file to distribution, such as a zip file that can be run like a native app from the OS."

What's Missing From the Web

Avoid JavaScript

"Completely Delete JavaScript from the whole Universe."

"The lack of precise decimal numbers in JavaScript has been a pain point for a long time particularly considering that the apps I work on have to deal primarily with money. If there is one thing I would love to have in JavaScript is a decimal type like the one in .Net."

"Basically I want all browsers to be rewritten to use WebAssembly on top of which DOM/DOM manipulation is implemented on top of which, next level up, JavaScript, etc. are implemented in the VM/IL virtual assembly op code language of the WebAssembly VM implementation."

Performance

"Better bind back and frontend. Node/TS helps a lot, but the solutions are immature and pretty much non-performant, and that severely limits the amount of stuff that can be done on the web. Nowadays browsers have incredible capabilities, and are as complex as an OS, so I would love to see support for more complex, performant, and united apps."

"Build high-performance solutions (C++ like performant) with strong graphical features. An app that would resemble an AAA video game but in the browser. I'm looking forward to WASM going into mainstream."

"Rendering performance and options are disappointing compared to native. The primary issue is the 1-2 frames of additional delay introduced by the web browser when using canvas/WebGL to display mouse/keyboard inputs. Configuring vsync/gsync/high refresh rate can also be frustrating."

Debugging

"Easier debugging of browser apps from command line for automated testing. Yes, there are hacky ways to do it, but first class ways of debugging (tests passed/tests failed) from the command line would be such a time saver."

"Debugging should be more traceable. For example using Vue or other web frameworks sometimes an error is marked as if the root cause were the framework (does not pinpoint the correct js file)."

"Containers have made the use of a lot of debugging/dev tools harder or not able to be used."

Security

"While I still may be new to the coding journey, security...on the internet is still the number one concern. Personally, I feel the complex jargon used to describe how web security functions in the modern era makes learning and implementing best security practices one of the most difficult things as a new programmer, even if you go with the third-party options."

"Proper Firefox extensions, I would love to go back to the state before crippling them with webextensions and questionable "security" decisions (companies should allow users to install extensions from anyone, not just subjects approved by browser companies. You cannot in Firefox nor Chrome fully install an extension which was not signed by Mozilla/Google)."

"Interact with OS specific APIs to store and receive secrets like e.g. PGP keys in order to allow E2E and "crowd" encryption. Currently it is

only possible to store such secrets in the local storage which is a security risk and can end in data loss when the user clears the data."

Accessibility

"Accessibility tools and features are grossly lacking."

"Accessibility support beyond most trivial of basics requires heavy research, and knowing if you're doing it well is [shrug].

"Accessibility, screen reader supports are chaos right now. As every application is trying to be WCAG compliant it make sense [that] ARIA comes by default or have better APIs to handle."

Technologies

Programming Languages

Programming Languages

Being core programming languages for the web, we wanted to know what pain points developers have when using JavaScript, HTML, CSS, and WebAssembly.

In 2019, for each of the languages we asked, "What are the biggest pain points for you when it comes to [programming language] development? Select all that apply."

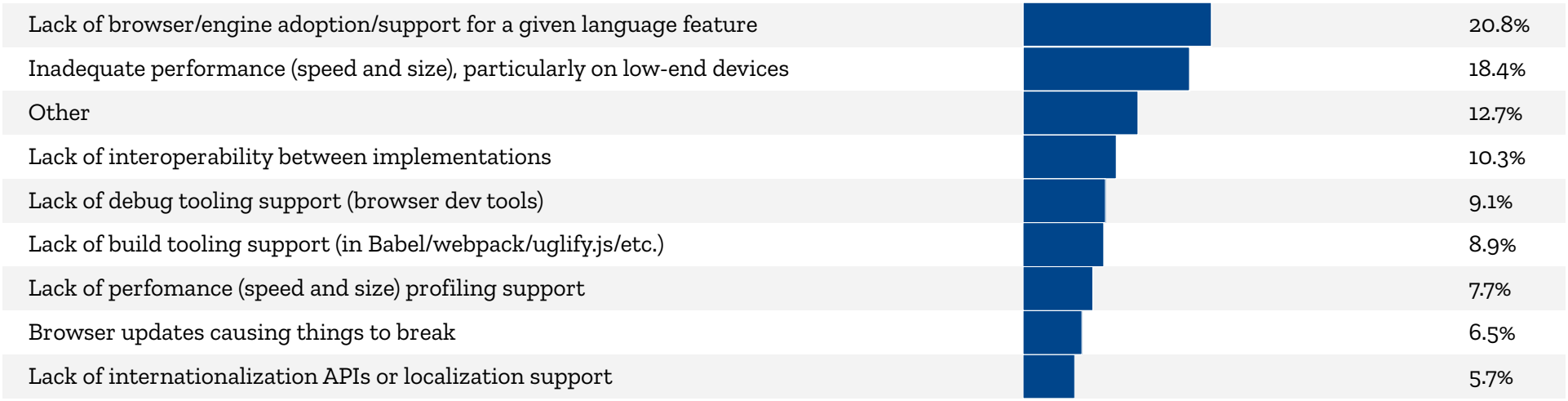
We asked the same question, but new for this year, we then followed up with a second question, taking all the answers selected from the previous question, we asked them to pick the biggest. This allowed us to report more accurate numbers in terms of which is truly the biggest pain point.

JavaScript

2.5% of our overall respondents said they do not use JavaScript. Of those who do use JavaScript, 14.9% said they have no pain points. Of those who do have issues with JavaScript, the biggest pain point is the same as last year, "Lack of browser/engine adoption/support for a given language feature." Other ranks third, which suggests there are pain points not captured in our list.

We had respondents who selected JavaScript as one of the languages they use define where they are using JavaScript:

- 45.1% are using JavaScript on a browser
- .8% are using JavaScript on a server
- 51.6% are using JavaScript on both a browser and a server



n = 5,472

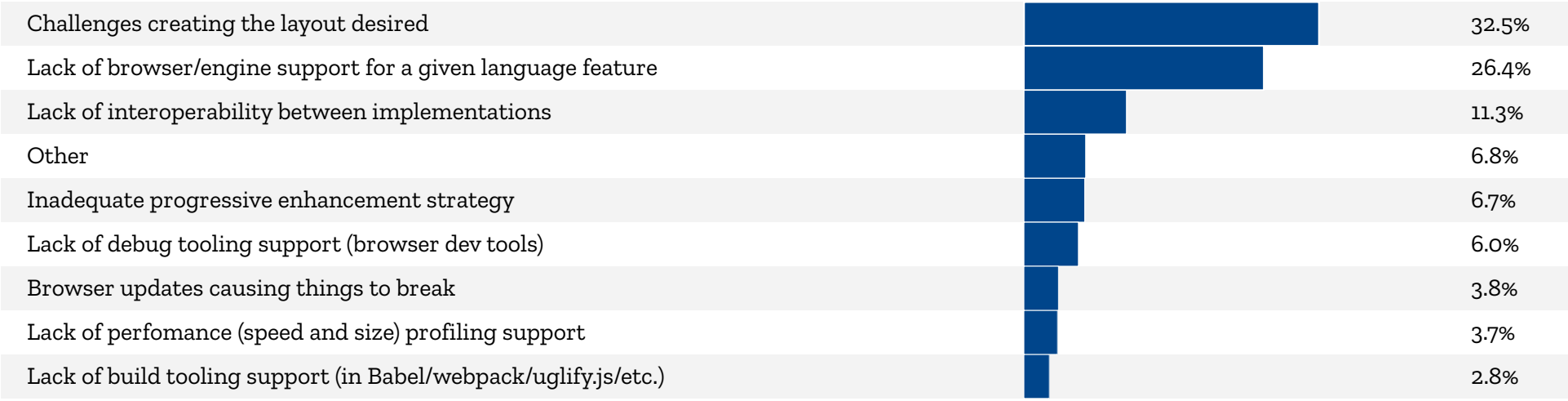
HTML

For those who use HTML, 32.6% said they have no pain points. Of those who do have issues with HTML, the biggest pain point is the same as JavaScript, "Lack of browser/engine adoption/support for a given language feature." A close second is, "Inability to customize components built into HTML."

n = 4,063

CSS

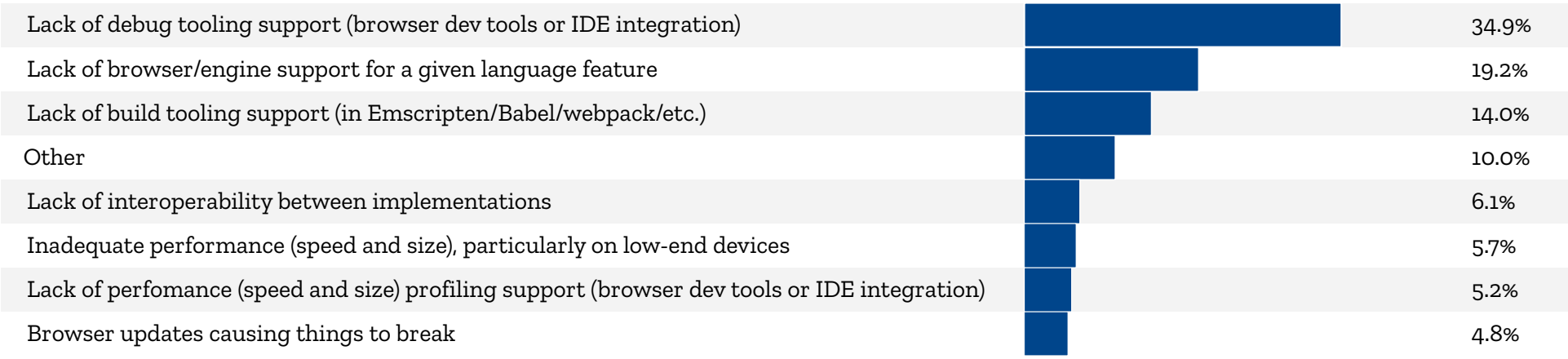
For those who use CSS, 12.5% said they have no pain points. Of those who do use CSS, 32.5% said their biggest pain point is challenges creating the layout specified. This was the same pain point as last year. Similar to JavaScript, Other ranks high on the lists, which suggests there are pain points not captured in our list.



n = 5,017

WebAssembly

For those who use WebAssembly, 20.8% said they have no pain points. Since Web Assembly is still considered a relatively new language, the respondents who use the language were able to provide information about their pain points. The largest pain, with 34.9%, is a lack of debug tooling support. This was the same pain point as last year. Like CSS, Other ranks fourth, which suggests there are pain points not captured in our list.



Lack of debug tooling support (browser dev tools or IDE integration)	34.9%
Lack of browser/engine support for a given language feature	19.2%
Lack of build tooling support (in Emscripten/Babel/webpack/etc.)	14.0%
Other	10.0%
Lack of interoperability between implementations	6.1%
Inadequate performance (speed and size), particularly on low-end devices	5.7%
Lack of perfomance (speed and size) profiling support (browser dev tools or IDE integration)	5.2%
Browser updates causing things to break	4.8%

n = 229