# What's Missing From the Web

# What’s Missing From the Web

The survey had one question where the response was open-ended. The question was, ”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.

We asked this open-ended question because we didn’t know what answers to suggest as possibilities, but intend to use the results gathered to inform future iterations of this question.

Of the 28,474 completed survey responses, we had 12,359 respondents answer this question, which is just shy of half. 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 9,570.

From there, we selected a random 1,000 answers to categorize into themes.

Aside from a surprising number of responses alluding to the desire to have the Web make a pot of coffee and achieve world domination, developers’ wants fell into 109 categories. However, of those 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%)

Falling just shy of the 3%, at 2.9%, were answers about needs related to CSS. Rounding out the 2% level were answers about avoiding JavaScript on the web platform (2.3%) and needs about gaming (2%).

The remaining 99 categories, each accounted for less than 2% of the 1000 responses analyzed.

Another note on the methodology for analyzing these answers is that since the question was open-ended, we received answers that had a list of things the respondent wanted to achieve on the web, but lacked the platform features to do so. To avoid giving one respondent’s answer more weight than those who only provided one item, we took the first response out of the list. In some cases, we could not make sense of the first response or it did not answer the question. In those instances, we moved on to the next item in the list, until we either deleted their response (for not answering the question) or categorized the first answer in the list that made sense considering the question.

On the following pages are short descriptions, which further explain the category, beyond just the title, as well as verbatims from survey responses. We chose the verbatims that best capture the deeper meaning of the category.

## Access to Hardware

Many of the answers categorized as access to hardware said simply that. Respondents wrote in answers such as, "access to hardware." Some responses included more context. Access to hardware APIs was a common response. Other common requests were access to contacts, calendars, and the camera. Verbatims that best capture the deeper meaning of the category:

- Access Hardware APIs on devices, deploy applications with native-like performance and functionality using only web technologies.
- Access low-level hardware on clients' browsers.
- Better access to hardware. We do a lot of scanning, it would be super helpful if there were an easier way to go from a scan to a canvas, to reading what is marked on a page.

## Browser Compatibility

Answers in this category asked for consistency in cross-browser compatibility, and across various devices and platforms. Another common desire was for better testing across browsers. CSS and JavaScript are the two most referenced languages in context to Browser Compatibility. Verbatims that best capture the deeper meaning of the category:

- The same cross-browser rendering, so clients don't complain fonts look good in one browser but not in another.
- Use new technologies as standard without worrying about browser support
- Supporting IE is a hell which we do because we have to, not because we want to.
- Create the same experience across browsers without worrying about CSS caveats. It's beyond infuriating coding in Chrome, turning to IE11 where everything is messed up.

## Access to Filesystem

Access to Filesystem could have been a subcategory of Access to Hardware, however the sheer volume of responses asking for access to the filesystem felt large enough to make this its own category. Many of the answers categorized with Access to Filesystem said simply that. Some answers acknowledge that users might need to grant permission before gaining access to the filesystem. Verbatims that best capture the deeper meaning of the category:

- Secure and usable solution for persistent local storage.
- Save files to the user's computer directly (after being granted permission)
- Granting limited access to file system directly from browser. It's highly infeasible but would allow direct interactions if done right.
- Handle the system files (to allow a user to work locally while having an online web application without Node.js)

# Performance

Respondents talked about performance in different ways. Some referenced knowing a users' bandwidth capacity. Other referred to specific programming languages, most notably JavaScript. Performance was also measured compared to the speed of native apps. Verbatims that best capture the deeper meaning of the category:

- I would love to see better performance. A lot of stuff is only possible with a strong back-end, because the browsers are just too slow. Performance is the most important factor to improve.
- See how much environmental resources I am using, as a part of performance.
- Native mobile app speed in web apps.
- JS performance is poor. It would be fun to write a Python or Java browser, but unfortunately, this has not changed.
- Accurately determine a user's current bandwidth capacity.

## PWA Support

Most responses categorized as PWA Support were about the lack of support and the desire to see full support. Not all vendors offer the same support for PWAs, which creates frustration for developers. Another common answer is having PWA as an option in app stores. Verbatims that best capture the deeper meaning of the category:

- Full PWA support from all vendors. I would like to develop web-view-based apps publicly available on the web and supported on all platforms.
- PWAs are getting better — we're still waiting to have a similar experience on iOS as on Android. Also Firefox does not offer to "install" a PWA.
- Web can do most of the things which native does but web app is not commonly used like app, PWA should be there in major app stores as a category.
- Have installable apps appear in app stores.

## Debugging

Respondents who wrote answers about debugging often wanted better tools to make it easier and more convenient to debug from capturing bugs to resolving bugs. Verbatims that best capture the deeper meaning of the category:

- Better capture and report a scenario that leads to a bug in the user's browser.
- More convenient way to debug.
- Modify the source code while debugging.
- Mobile debugging.

## Access to Native APIs

Answers categorized as Access to Native APIs were similar to Access to Hardware or Access to Filesystem in that developers wanted integration with a device OS. Also, having web apps behave more like native apps. Verbatims that best capture the deeper meaning of the category:

- Interacting with other desktop applications.
- System OS integrations.
- Use an expanded set of platform provided components.
- I would like Web Apps to be more native.

# Technologies

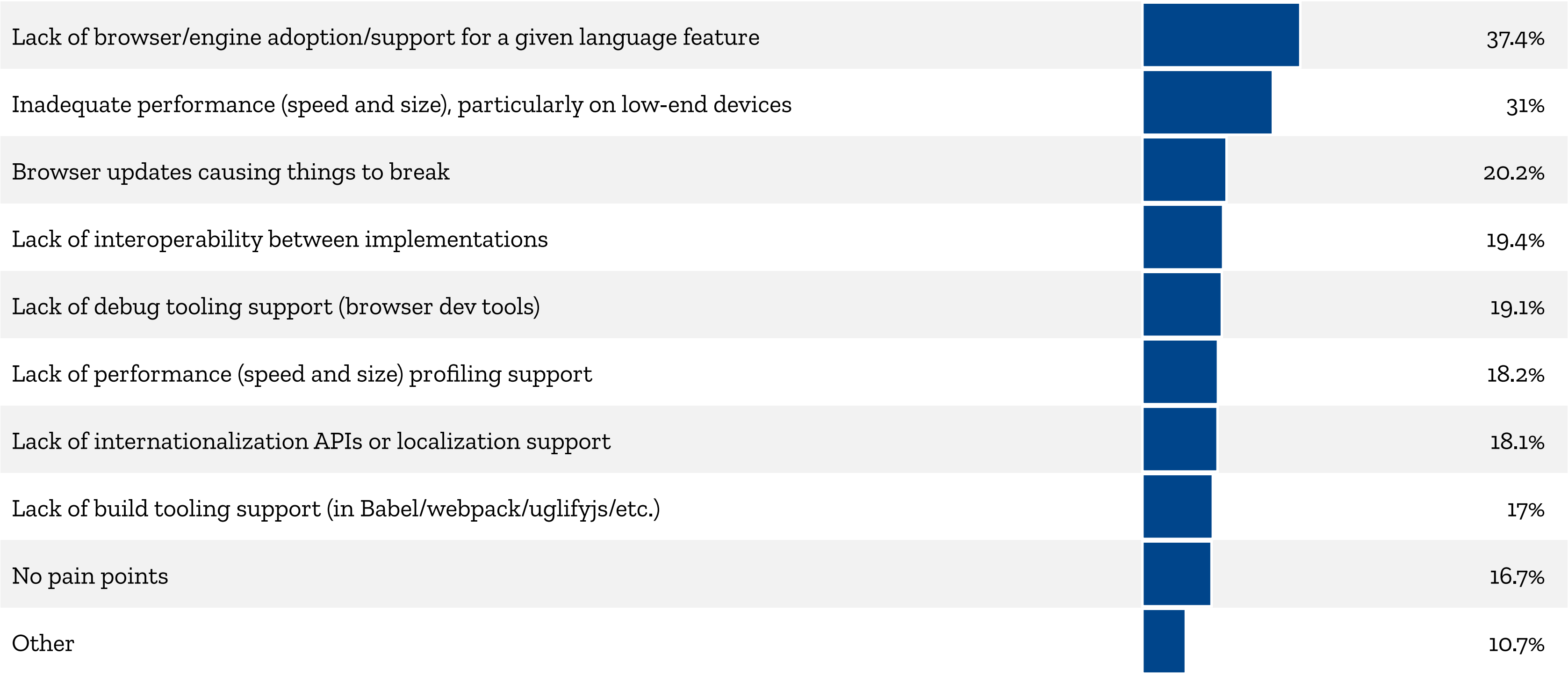
## Programming Languages

As core technologies for the web, we wanted to understand developers’ pain points using JavaScript, HTML, CSS, and WebAssembly.

# JavaScript

The biggest pain point for JavaScript was, "Lack of browser/engine adoption/support for a given language feature." Though, 16% of respondents have no pain points with JavaScript.

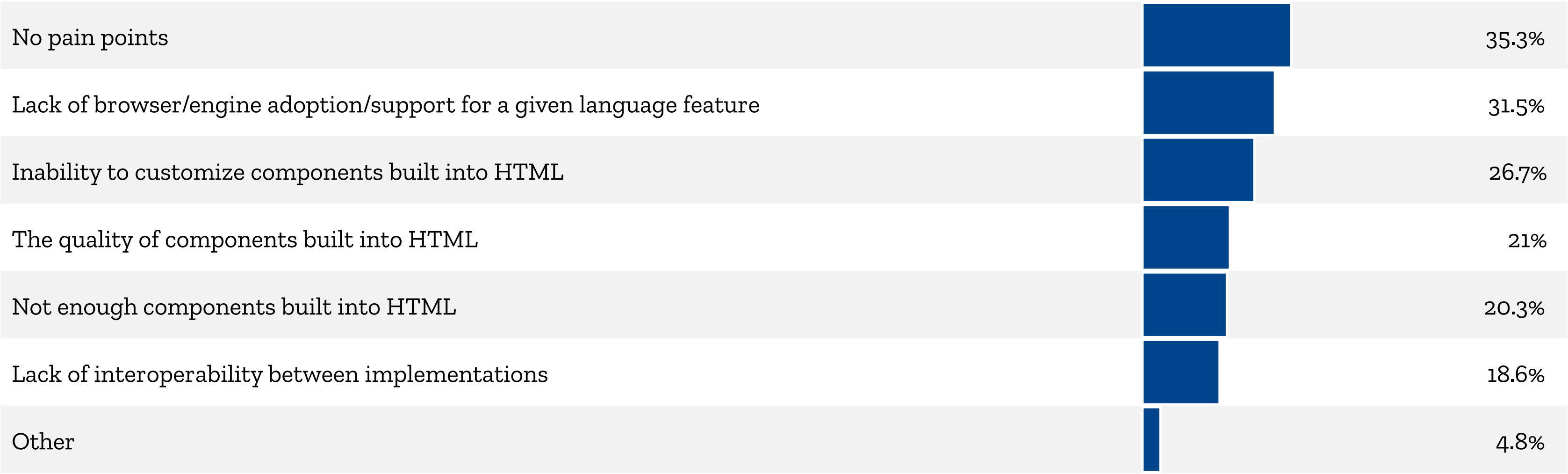
**JavaScript - Biggest Pain Points**



# HTML

HTML seems to be the shining star of these languages as roughly a third of respondents (35.3%) have no issues with the language. That said, the biggest pain point with HTML is a lack of adoption and support for a given feature at 31.5%.

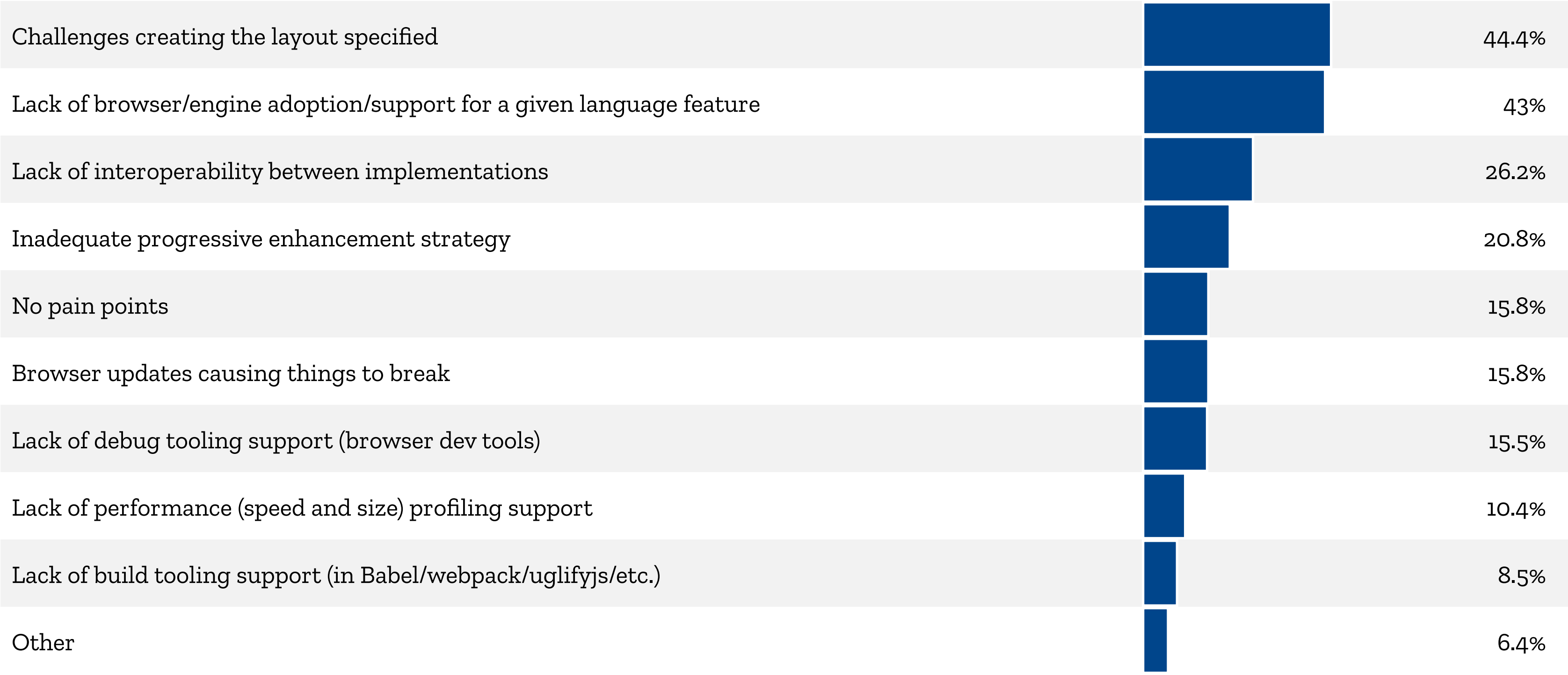
**HTML - Biggest Pain Points**



# CSS

Nearly half (44%) of respondents said their biggest pain point with CSS is challenges creating the layout specified. Since CSS is the language to style webpages, the results suggest that achieving a desired layout is difficult to do using CSS. The results can be interpreted in different ways so further investigation is needed. One way to interpret this response is that CSS is causing developers a lot of grief when using the language to create layouts. Another way to interpret this is that developers are trying to do their best using CSS to achieve a web-friendly layout from a design that wasn't intended for the web.

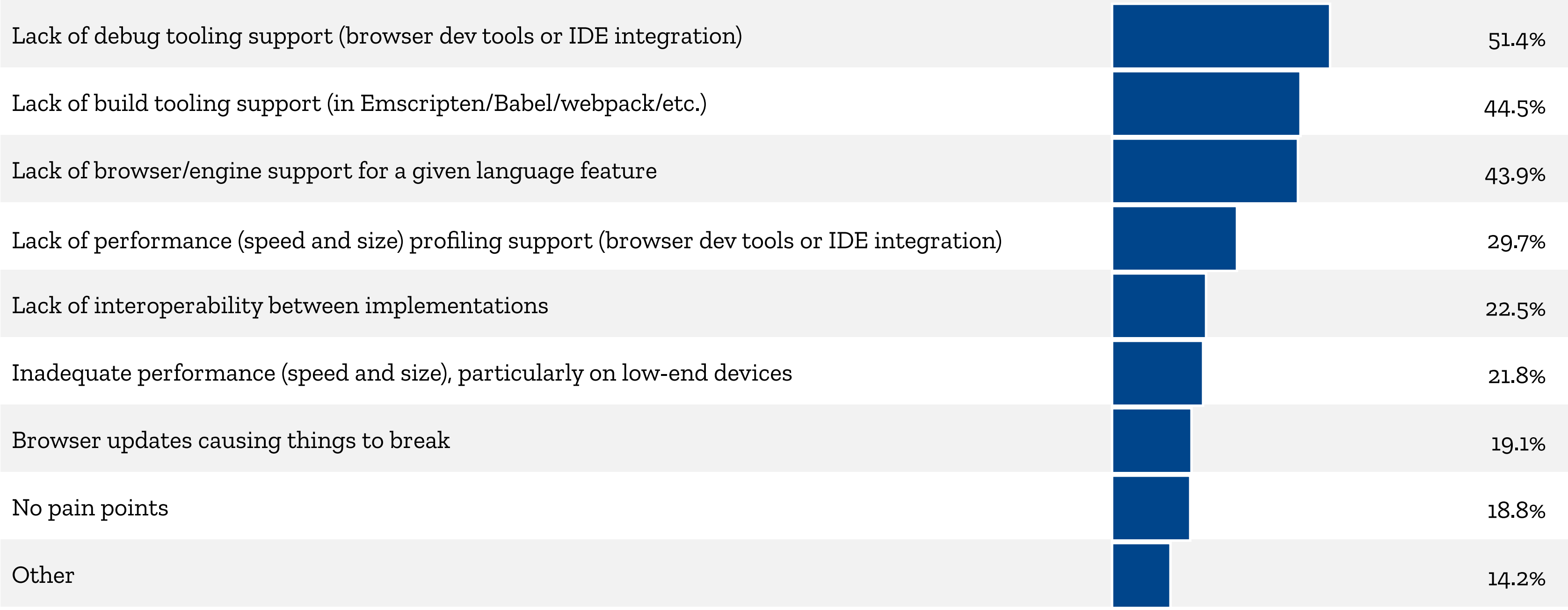
**CSS - Biggest Pain Points**



# WebAssembly

WebAssembly is a new technology, so only a handful of respondents, 851, which is just shy of 3% of the completed responses, had experience enough to respond. The results below are from a much smaller sample than those for JavaScript, HTML, and CSS. The largest pain, with 51.4% is a lack of debug tooling support.

**WebAssembly - Biggest Pain Points**

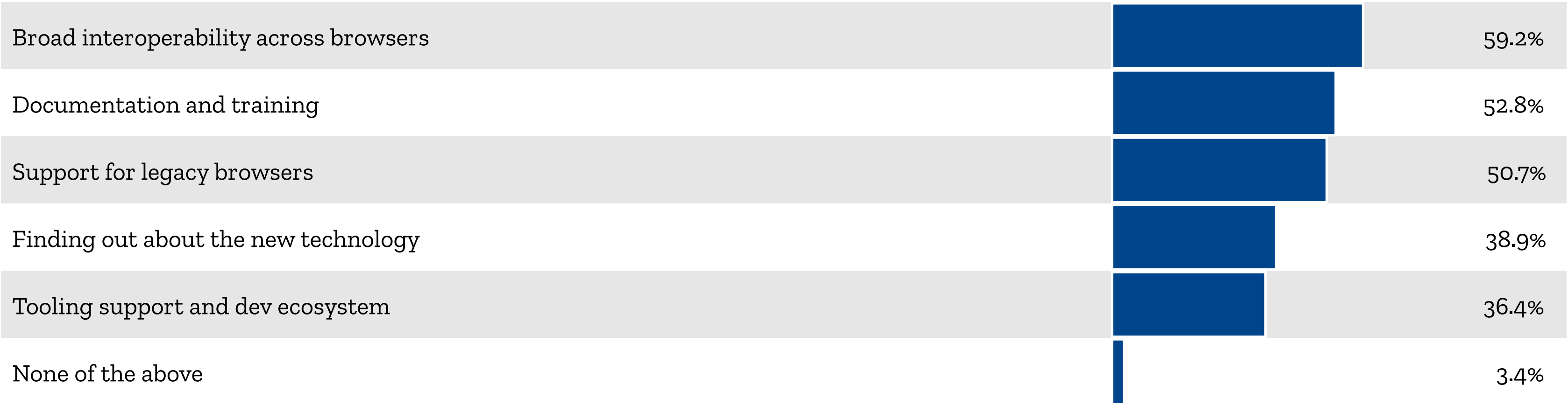


# Adoption of New Technologies

# Adoption of New Technologies

The biggest barrier developers face when adopting new technology is broad interoperability across browsers. This is in-line with the top ten needs where four had to do with browser compatibility.

### Adoption of New Technologies

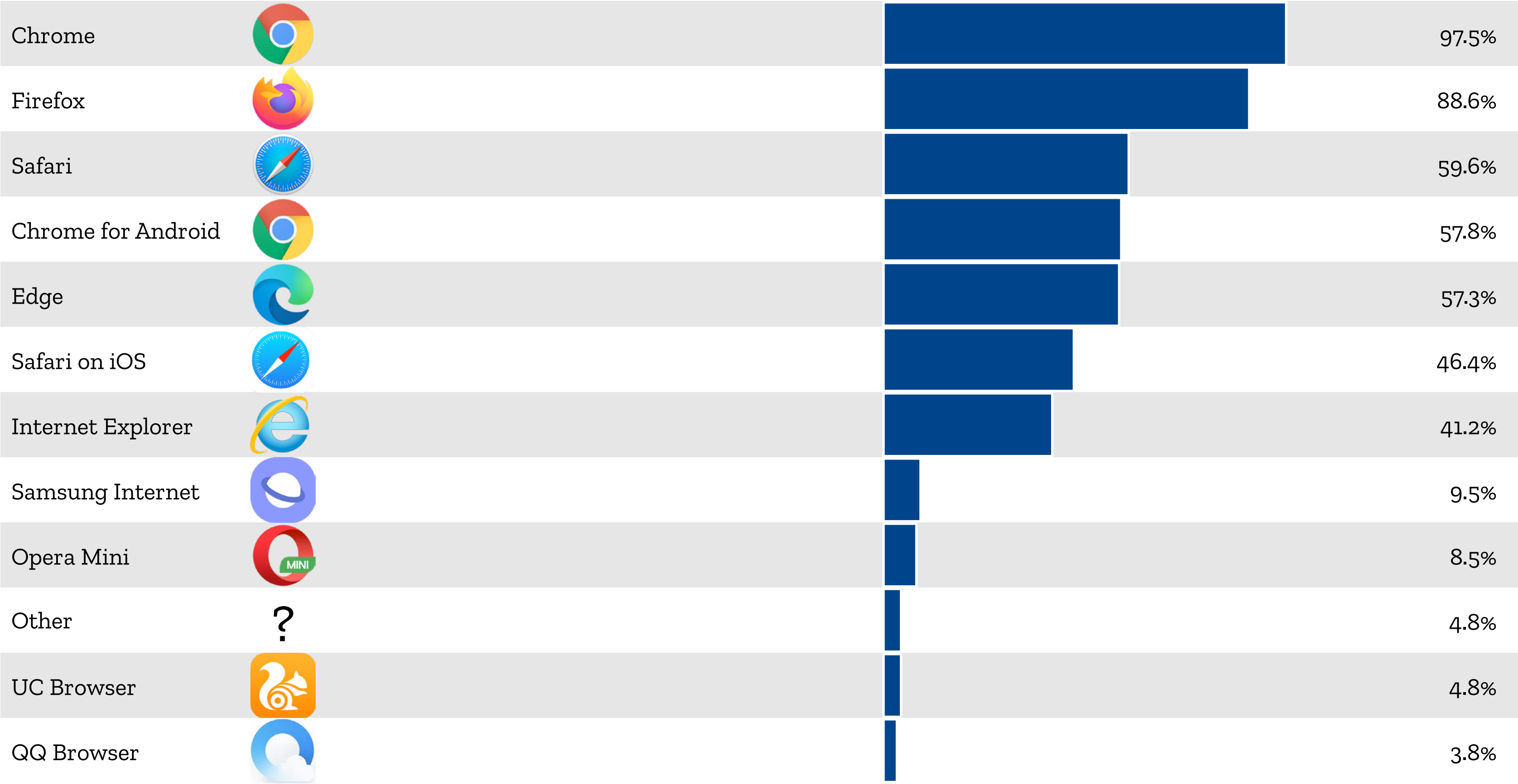


# Browsers

# Browsers

Chrome and Firefox lead the pack in terms of browsers developers support, 97.5% and 88.6% respectively. Third is Safari at 59.6%.

**Browsers Developers Support**


Chrome
Firefox
Safari
Chrome for Android
Edge
Safari on iOS
Internet Explorer
Samsung Internet
Opera Mini
MINI
Other
?
UC Browser
QQ Browser