Audit usage of navigator.userAgent, navigator.appVersion, and navigator.platform

A page or script is accessing at least one of navigator.userAgent, navigator.appVersion, and navigator.platform. Starting in Chrome 101, the amount of information available in the User Agent string will be reduced.

To fix this issue, replace the usage of navigator.userAgent, navigator.appVersion, and navigator.platform with feature detection, progressive enhancement, or migrate to navigator.userAgentData.

Note that for performance reasons, only the first access to one of the properties is shown.

[](https://blog.chromium.org/)

[Chromium Blog](https://blog.chromium.org/)

News and developments from the open source browser project

[User-Agent Reduction Origin Trial and Dates](https://blog.chromium.org/2021/09/user-agent-reduction-origin-trial-and-dates.html)

Tuesday, September 14, 2021

Back in May, we published [an update on our User-Agent string reduction plans](https://blog.chromium.org/2021/05/update-on-user-agent-string-reduction.html) with a promise to publish further details on timing. Now that we have an [origin trial ready for testing](http://developers.chrome.com/blog/user-agent-reduction-origin-trial) the Reduced User-Agent header (and associated JS interfaces) we have estimated timelines to share. What follows is repeated from the original blog post, but contains estimated Chrome versions where these Phases will begin to help you prepare.

The [Chromium schedule dashboard](https://chromiumdash.appspot.com/schedule) will be useful for understanding dates associated with each Chrome version and its progression from Canary into Beta and Stable Release.

Note: The usual disclaimers about estimating engineering deadlines apply—unforeseen circumstances may dictate delays. But in the case that we encounter delays, we do not intend to accelerate timelines between Phases.

Proposed Rollout Plan

We plan to roll out these changes slowly and incrementally in 7 Phases—pending origin trial feedback.

Reduction Preparation

Phase 1: **Since Chrome 92 (July 20, 2021)**

**Call to Action (CTA):** Audit your site usage to understand where migration may be necessary.

Warn about accessing navigator.userAgent, navigator.appVersion, and navigator.platform in DevTools, beginning in M92.

Phase 2: **Chrome 95 to Chrome 100**

**CTA:** Enroll in the origin trial for your site, and provide feedback until Chrome 101 is released.

Launch an origin trial for sites to opt into the final reduced UA string for testing and feedback, for at least 6 months.

We will evaluate feedback from origin trial partners and the community, and based on this feedback proceed to Phases 3 through 7 of our plan, giving the ecosystem adequate time to adapt in between them. Otherwise, depending on feedback we will reconsider the best course of action.

Reduction Rollout

Phase 3: **Chrome 100**

**CTA:** Enroll in the deprecation trial or Enterprise policy for your site, when needed.

Launch deprecation trial and Enterprise policy, for instances where a site may need more time for migration.

Phase 4: **Chrome 101**

**CTA:** Ensure your site is compatible with the reduced Chrome version number, and [migrate to UA Client Hints](https://web.dev/migrate-to-ua-ch/) if not.

Ship reduced Chrome MINOR.BUILD.PATCH version numbers (“0.0.0”). Once rolled-out, the reduced UA string would apply to all page loads on desktop and mobile operating systems for sites that do not opt into the deprecation trial.

Phase 5: **Chrome 107**

**CTA:** Ensure your site is compatible with the reduced Desktop UA string and related JS APIs, and [migrate to UA Client Hints](https://web.dev/migrate-to-ua-ch/) if not.

Begin roll-out of reduced Desktop UA string and related JS APIs (navigator.userAgent, navigator.appVersion, navigator.platform). Once rolled-out, the reduced UA string would apply to all page loads on desktop operating systems for sites that do not opt into the deprecation trial.

Phase 6: **Chrome 110**

**CTA:** Ensure your site is compatible with the reduced Mobile UA string and related JS APIs, and [migrate to UA Client Hints](https://web.dev/migrate-to-ua-ch/) if not.

Begin roll-out of reduced Android Mobile (and Tablet) UA string and related JS APIs. Once rolled-out, the reduced UA string would apply to all page loads on Android that do not opt into the deprecation trial.

Reduction Completion

Phase 7: **Chrome 113**

Deprecation trial ends and all page loads receive the reduced UA string and related JS APIs.

See the companion [Reduced User Agent string updates page](https://www.chromium.org/updates/ua-reduction) for more details and example User Agent strings at each of these phases. We will note any significant delays or changes on this page as well.

Posted by Mike Taylor and Jade Kessler, Chrome Team

Share on Twitter Share on Facebook

Page layout may be unexpected due to Quirks Mode

One or more documents in this page is in Quirks Mode, which will render the affected document(s) with quirks incompatible with the current HTML and CSS specifications.

Quirks Mode exists mostly due to historical reasons. If this is not intentional, you can add or modify the DOCTYPE to be `<!DOCTYPE html>` to render the page in No Quirks Mode.

# **Page lacks the HTML doctype, thus triggering quirks mode**

May 2, 2019 — Updated Aug 28, 2019

Available in: [Español](https://web.dev/i18n/es/doctype/), [日本語](https://web.dev/i18n/ja/doctype/), [한국어](https://web.dev/i18n/ko/doctype/), [Português](https://web.dev/i18n/pt/doctype/), [Русский](https://web.dev/i18n/ru/doctype/), [中文](https://web.dev/i18n/zh/doctype/), [English](https://web.dev/i18n/en/doctype/)

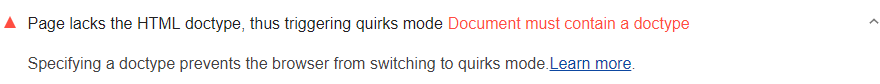
Appears in: [Best Practices audits](https://web.dev/lighthouse-best-practices)

On this page

Specifying a doctype prevents the browser from switching to [quirks mode](https://developer.mozilla.org/docs/Web/HTML/Quirks_Mode_and_Standards_Mode), which can cause your page to [render in unexpected ways](https://quirks.spec.whatwg.org/#css).

## How the Lighthouse doctype audit fails [#](https://web.dev/doctype/?utm_source=devtools#how-the-lighthouse-doctype-audit-fails)

[Lighthouse](https://developers.google.com/web/tools/lighthouse/) flags pages without the <!DOCTYPE html> declaration:



Each Best Practices audit is weighted equally in the Lighthouse Best Practices Score. Learn more in [The Best Practices score](https://developers.google.com/web/tools/lighthouse/v3/scoring#best-practices).

## How to add a doctype declaration [#](https://web.dev/doctype/?utm_source=devtools#how-to-add-a-doctype-declaration)

Add the <!DOCTYPE html> declaration to the top of your HTML document:

<!DOCTYPE html>  
<html lang="en">  
…

See MDN's [Doctype](https://developer.mozilla.org/docs/Glossary/Doctype) page for more information.

## Resources [#](https://web.dev/doctype/?utm_source=devtools#resources)

* [Source code for **Page lacks the HTML doctype, thus triggering quirks mode** audit](https://github.com/GoogleChrome/lighthouse/blob/ecd10efc8230f6f772e672cd4b05e8fbc8a3112d/lighthouse-core/audits/dobetterweb/doctype.js)
* [Doctype](https://developer.mozilla.org/docs/Glossary/Doctype)
* [Quirks Mode and Standards Mode](https://developer.mozilla.org/docs/Web/HTML/Quirks_Mode_and_Standards_Mode)

Indicate whether to send a cookie in a cross-site request by specifying its SameSite attribute

Because a cookie’s SameSite attribute was not set or is invalid, it defaults to SameSite=Lax, which prevents the cookie from being sent in a cross-site request. This behavior protects user data from accidentally leaking to third parties and cross-site request forgery.

Resolve this issue by updating the attributes of the cookie:

* Specify SameSite=None and Secure if the cookie should be sent in cross-site requests. This enables third-party use.
* Specify SameSite=Strict or SameSite=Lax if the cookie should not be sent in cross-site requests