**Use passive listeners to improve scrolling performance**

May 2, 2019 • Updated Aug 28, 2019

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

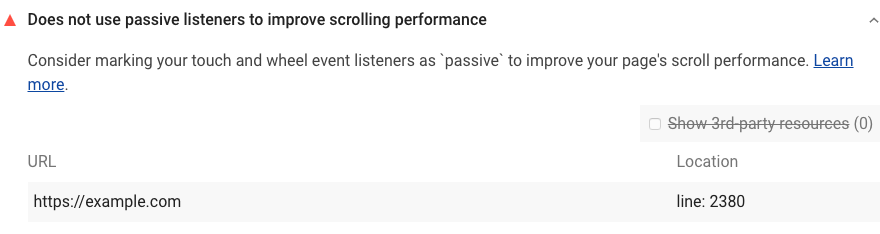
Touch and wheel event listeners are useful for tracking user interactions and creating custom scrolling experiences, but they can also delay page scrolling. Currently, browsers can't know if an event listener will prevent scrolling, so they always wait for the listener to finish executing before scrolling the page. Passive event listeners solve this problem by letting you indicate that an event listener will never prevent scrolling.

**Browser compatibility** [**#**](https://web.dev/uses-passive-event-listeners/?utm_source=lighthouse&utm_medium=devtools#browser-compatibility)

Most browsers support passive event listeners. See [Browser compatibility](https://developer.mozilla.org/docs/Web/API/EventTarget/addEventListener#Browser_compatibility)

**How the Lighthouse passive event listener audit fails** [**#**](https://web.dev/uses-passive-event-listeners/?utm_source=lighthouse&utm_medium=devtools#how-the-lighthouse-passive-event-listener-audit-fails)

[Lighthouse](https://developers.google.com/web/tools/lighthouse/) flags event listeners that may delay page scrolling:



Lighthouse uses the following process to identify event listeners that may affect scrolling performance:

1. Collect all event listeners on the page.
2. Filter out non-touch and non-wheel listeners.
3. Filter out listeners that call preventDefault().
4. Filter out listeners that are from a different host than the page.

Lighthouse filters out listeners from different hosts because you probably don't have control over these scripts. There may be third-party scripts that are harming your page's scrolling performance, but these aren't listed in your Lighthouse report.

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 make event listeners passive to improve scrolling performance** [**#**](https://web.dev/uses-passive-event-listeners/?utm_source=lighthouse&utm_medium=devtools#how-to-make-event-listeners-passive-to-improve-scrolling-performance)

Add a passive flag to every event listener that Lighthouse identified.

If you're only supporting browsers that have passive event listener support, just add the flag. For example:

document.addEventListener('touchstart', onTouchStart, {passive: true});

If you're supporting [older browsers that don't support passive event listeners](https://developer.mozilla.org/docs/Web/API/EventTarget/addEventListener#Browser_compatibility), you'll need to use feature detection or a polyfill. See the [Feature Detection](https://github.com/WICG/EventListenerOptions/blob/gh-pages/explainer.md#feature-detection) section of the WICG [Passive event listeners](https://github.com/WICG/EventListenerOptions/blob/gh-pages/explainer.md) explainer document for more information.