# Early Hints DevTools support

**Public document**
Last update: May 18, 2023
Author: bashi@chromium.org
Input from: yoavweiss@, kenjibaheux@, barrypollard@

Lack of DevTools support for Early Hints preloads is the top complaint from Web developers and CDN providers. Debuggability is a key aspect for wider adoption of Early Hints. This document discusses how to support Early Hints preload in DevTools.

## Debugging use cases

Web developers need to perform various types of analysis when developing their site with Early Hints. This section explores those analysis types:

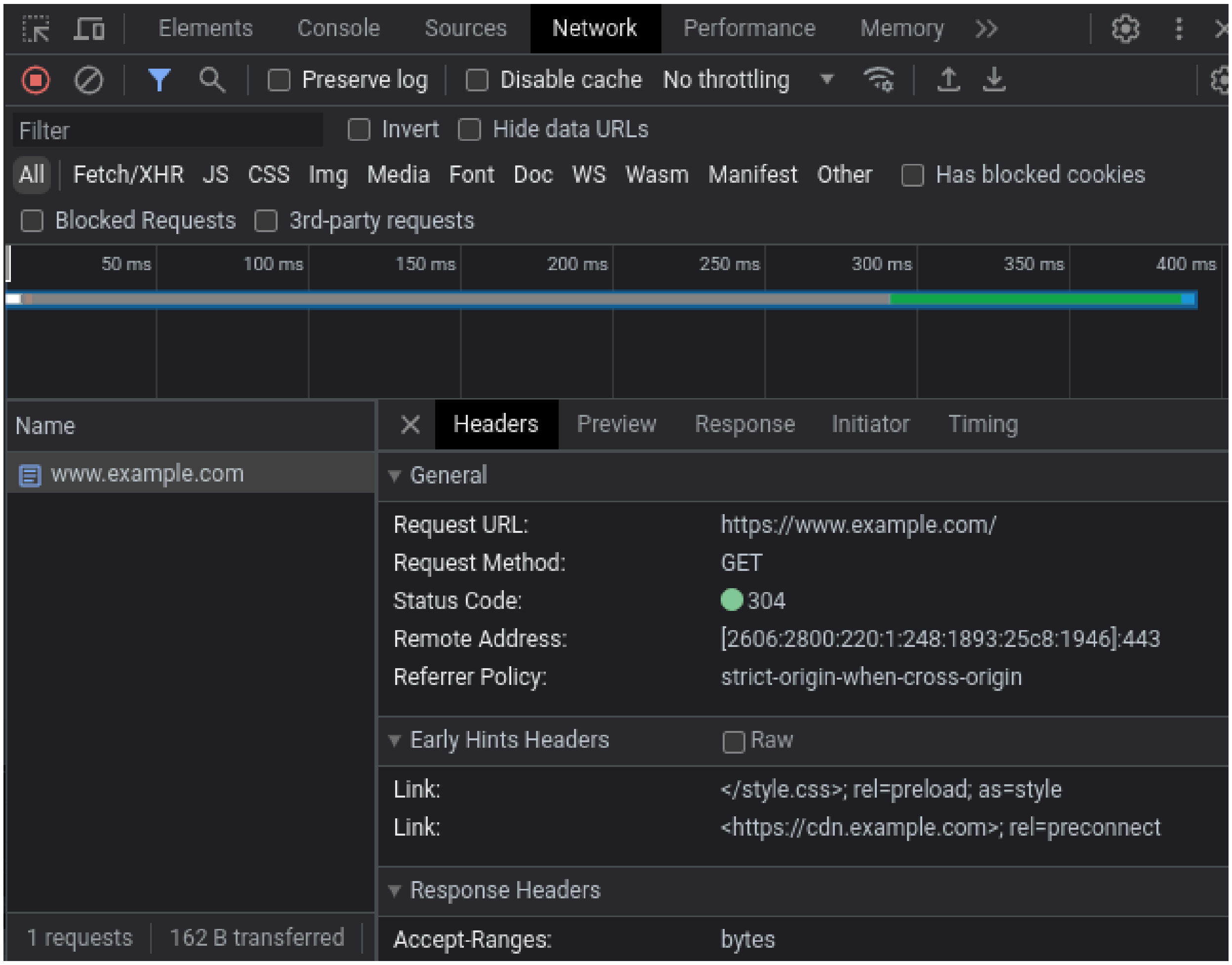
- Developers need to know that Early Hints headers were received, when they were received, and what was their type (`preconnect`, `preload`, unsupported types (e.g. `prefetch`))
- They need to know when a resource fetch was already cached when needed due to an Early Hints
- The need to know when a resource connection already existed and hence reused due to an Early Hint

Beyond that, they need Early Hints to work well even with DevTools' "disable cache" mode, which is often used in development.

## Showing Early Hints headers

Prototype CL

We add "Early Hints Headers" section in RequestHeadersView. The following screenshot shows what it looks like with a mocked Early Hints response.



▼ General

Request URL: https://www.example.com/
Request Method: GET
Status Code: ● 304
Remote Address: [2606:2800:220:1:248:1893:25c8:1946]:443
Referrer Policy: strict-origin-when-cross-origin

▼ Early Hints Headers □ Raw

Link: </style.css>; rel=preload; as=style
Link: <https://cdn.example.com>; rel=preconnect

▼ Response Headers

Accept-Ranges: bytes

To plumb Early Hints headers, we add an optional Headers value to the Response object in browser_protocol.pdl. The browser process passes Early Hints headers via devtools_instrumentation::OnNavigationResponseReceived().

We consider styling headers based on how Chrome handled them:

- Gray-out ignored headers
- Show warning icon for unused preloads and unhandled headers due to misconfiguration (e.g. duplicates and security check failures)

The Early Hints Headers section doesn't support overrides. The purpose of these headers is to trigger preload/preconnect before receiving the main document. Modifying these headers locally after receiving the main document wouldn't make sense.

## Showing resource fetches are triggered by Early Hints

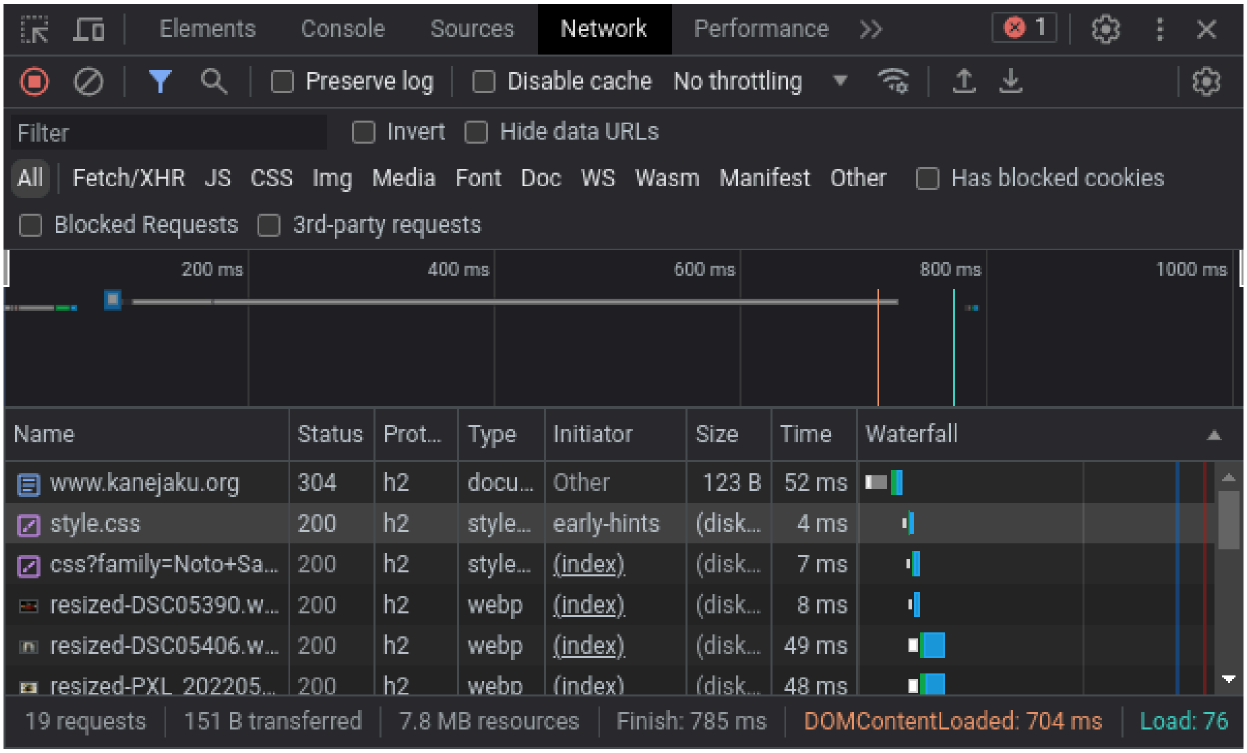
### Approach 1: Use “Size” column of Network panel

Historically “Size” column has been a place to indicate that the resource is served from disk cache, service workers, and HTTP/2 push etc. The column can be used for Early Hints preloads too.

This approach doesn’t make sense because EH preloads are orthogonal to where the resource is coming from. Web developers would also want to see the actual size of the response when the resource was fetched from the network.

### Approach 2: Use “Initiator” column of Network panel

The Fetch standard defines “early-hints” initiator type. It’s a natural extension for DevTools to show that the initiator is “early-hints” when the resource fetch is triggered by EH preload.



| Name | Status | Prot... | Type | Initiator | Size | Time | Waterfall |
|---|---|---|---|---|---|---|---|
| www.kanejaku.org | 304 | h2 | docu... | Other | 123 B | 52 ms | |
| style.css | 200 | h2 | style... | early-hints | (disk... | 4 ms | |
| css?family=Noto+Sa... | 200 | h2 | style... | (index) | (disk... | 7 ms | |
| resized-DSC05390.w... | 200 | h2 | webp | (index) | (disk... | 8 ms | |
| resized-DSC05406.w... | 200 | h2 | webp | (index) | (disk... | 49 ms | |
| resized-PXL_202205... | 200 | h2 | webp | (index) | (disk... | 48 ms | |

19 requests | 151 B transferred | 7.8 MB resources | Finish: 785 ms | DOMContentLoaded: 704 ms | Load: 76

This approach would be easy to implement. To plumb the information from blink to DevTools, we add a new initiator type in browser_protocol.pdl. We add a similar initiator type to NetworkRequet . NetworkDataGridNode uses the initiator type to show the Early Hints preload initiator.

### Approach 3: Introduce a dedicated panel

Prerendering has its own panel to show prerendering events. EH preloads is somewhat similar to prerendering in a sense that EH preloads happen before the main document fetch. It may make sense to add a separate panel for EH preloads.

This approach is complex compared to other approaches, requiring a bunch of UI work. Also, EH preloads are much simpler than prerendering. There wouldn’t be much to display in the panel. We will only show a list of URLs that are fetched by EH preloads for each page.

### Proposal

Given the pros/cons of these approaches, this document proposes to take Approach 2: Use the “Initiator” column.

## Estimated amount of work

The estimated amount of work is 1Q or 1.5Q for a SWE.

## Appendix

### Feedback from Google Docs

- Want the ability to track (e.g. via the resource timing API) which resources were preloaded via early hints and when the early hints preload started/finished;
- "early-hints" initiator type is useful, but it’s currently broken. TODO: Docs found some documentation suggesting that we fixed it but that’s not (yet) true; Kenji to find out which documentation this was.
- It seems that the early hints timings are only reported if the actual response is used in the page. It would be nice to have that information regardless so that we can tell when we are preloading resources that aren't actually used. Being able to monitor this in production would be useful.
- The specification says that early hints should work over HTTP/1.1. But, as we discussed earlier, Chrome only intends for this feature to work over HTTP/2 for compatibility reasons.
    - There seems to be quite a few online examples of HTTP/1.1 early hints which is confusing.

    - Supporting simple HTTP/1.1 would help devs prototype their implementation given it's a simpler protocol. Maybe we could log a warning to the console when sending early hints over HTTP/1.1? Or maybe, in addition to logging, also not preload the resources over HTTP/1.1 unless a specific flag is passed to Chrome during startup?
- Ability to inspect (and maybe override) interim headers in DevTools. Currently, to inspect the interim response, we need to start a chrome://net-export trace and dig through the logs to find the specific response.
- Uncachable Resources: not limited to Early Hints, but it seems that Chrome discards preloaded resources when they aren't cacheable, including resources preloaded over early hints. This can lead to potentially "double sending" requests for resources.
    - It might be worth issuing a warning in the console (or via other means) when the server responds with an uncachable resource in the preload response.