

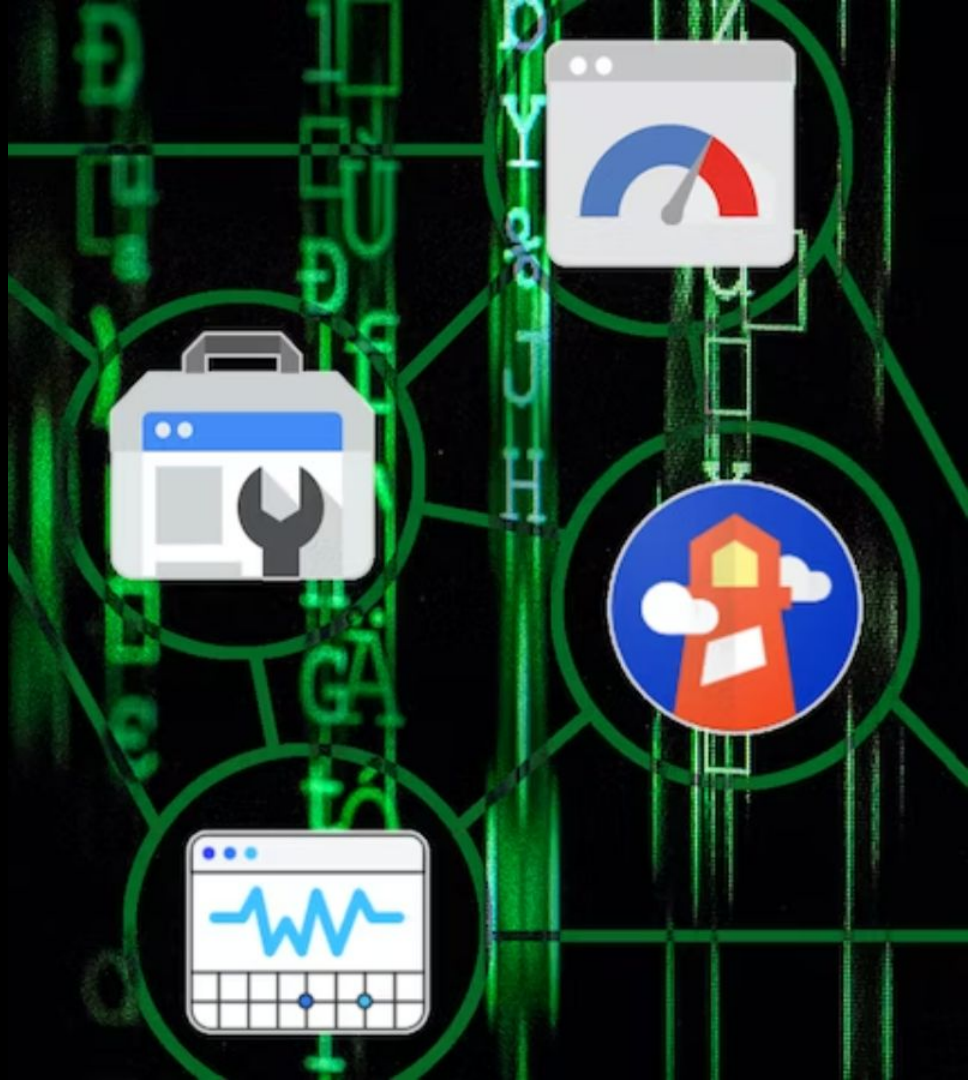
Better than loading fast... is loading instantly!

Barry Pollard

Web Performance Developer Advocate,
Google Chrome



FOSDEM'24
Brussels / 3 & 4 February 2024



**Hands up who likes the
web?**

**Hands up who likes the
slow
web?**

Core Web Vitals

(Loading)

LCP

Largest Contentful Paint



(Visual Stability)

CLS

Cumulative Layout Shift



(Interactivity)

INP

Interaction to Next Paint



Largest Contentful Paint



Largest Contentful Paint



2.5 seconds is seen as “Good”

Largest Contentful Paint



2.5 seconds is seen as “Good”,
but can we do “Better than Good”?

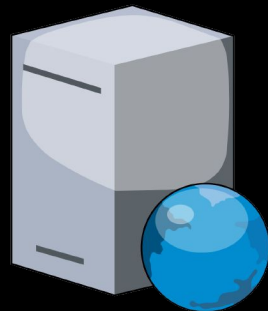
Largest Contentful Paint



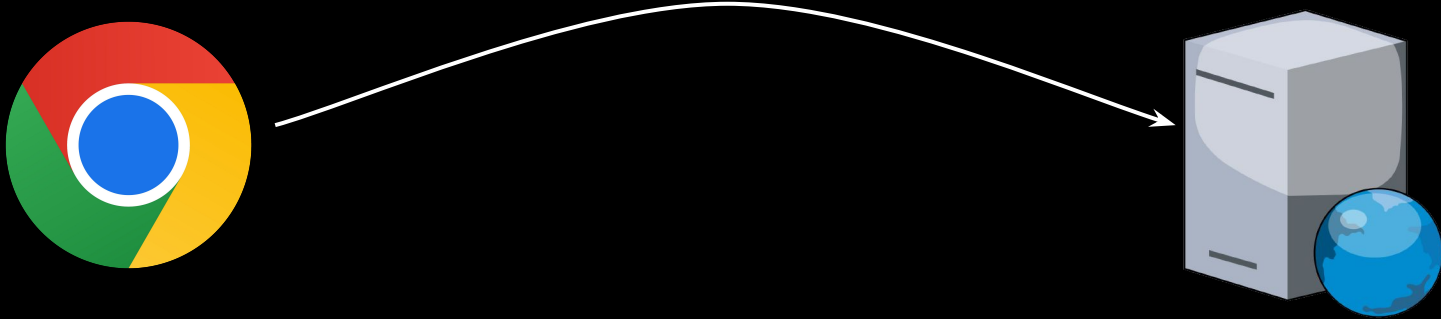
2.5 seconds is seen as “Good”,
but can we do “Better than Good”?

How can we get instant?

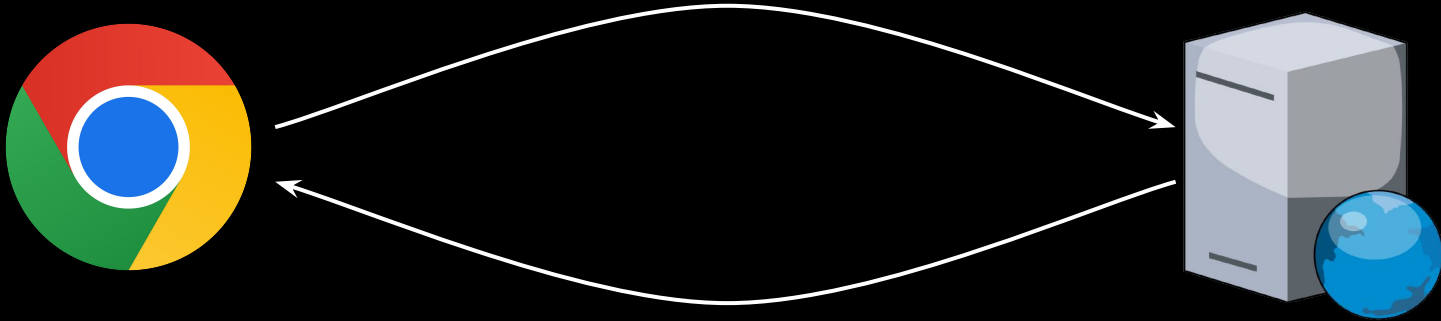
Websites have an inherent slowness



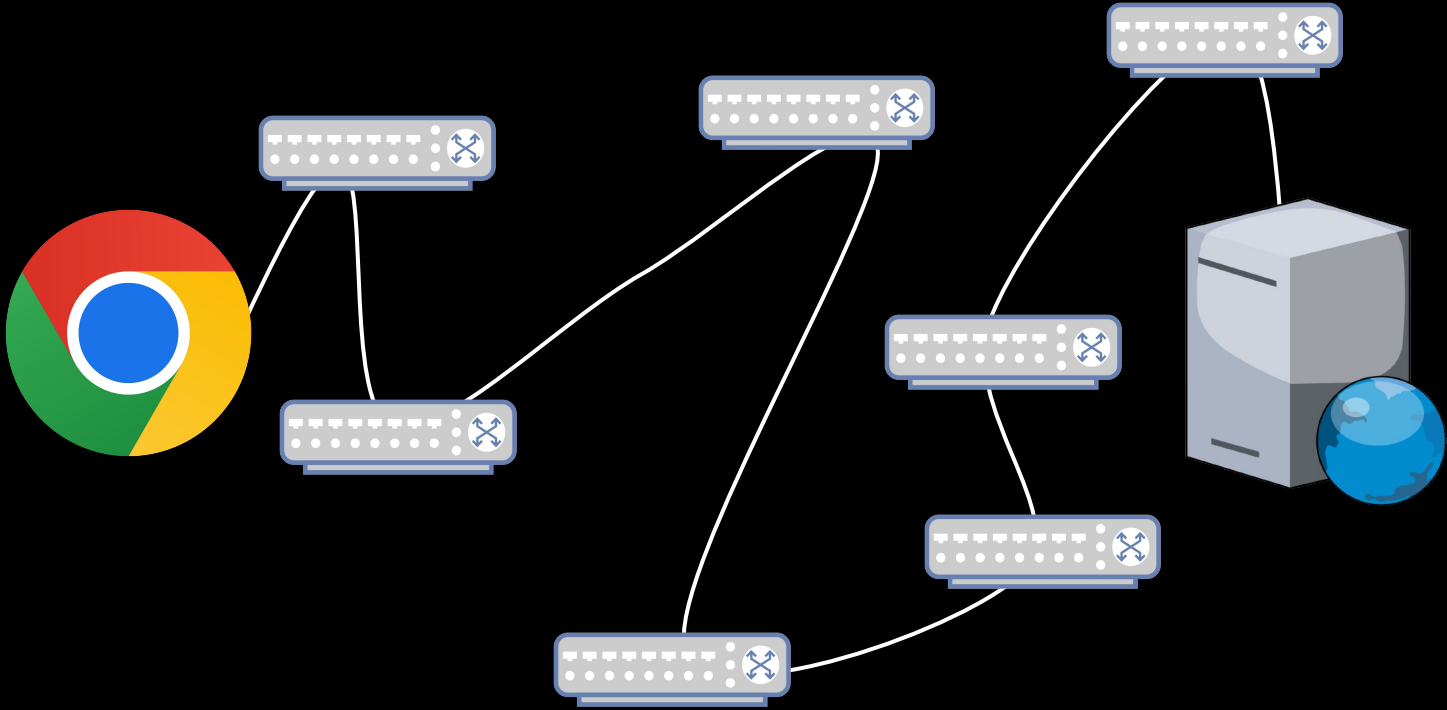
Websites have an inherent slowness



Websites have an inherent slowness



Websites have an inherent slowness



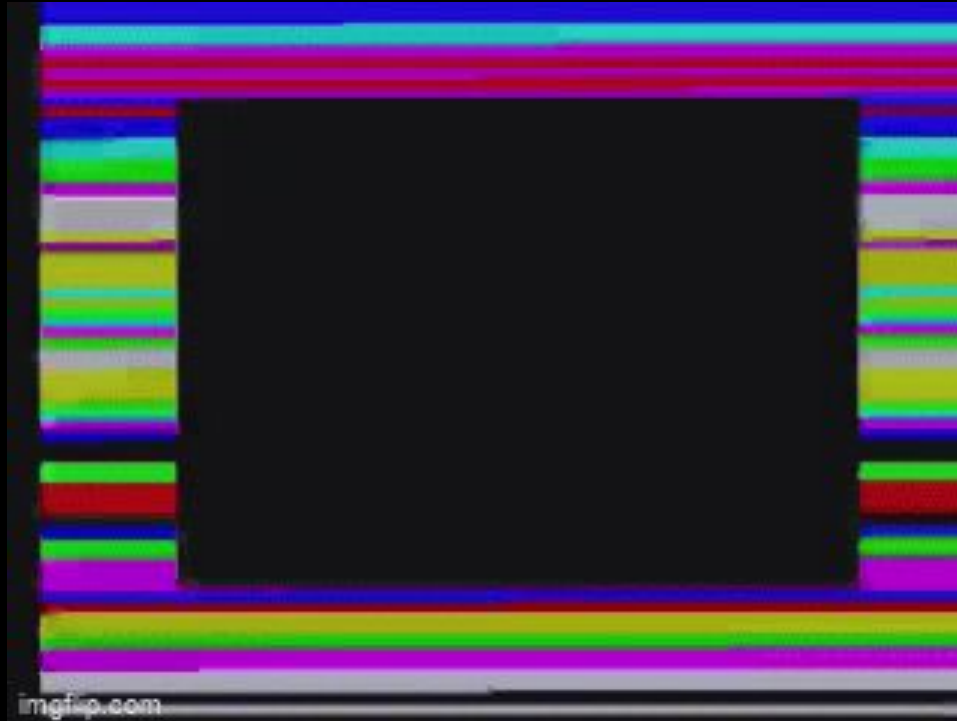
SPA is one attempt to work around this

SPA is one attempt to work around this

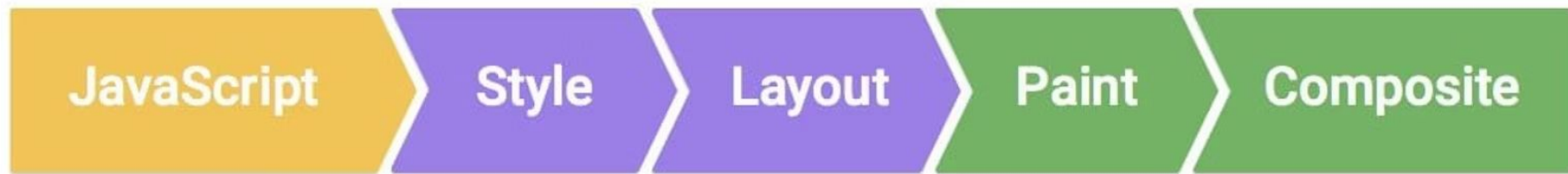


Spinner Page Application

Reminds me of 80s computer games loading



And all that's before you even get to the browser



THE TRIP FINDER

They say the world has seven wonders. We think that's only the beginning.

WHEN DO YOU WANT TO TRAVEL? ▾

WHY DO YOU WANT TO TRAVEL? ▾

TAKE ME THERE



DO NOT
TOUCH
ARTWORK

**To deliver instant loading we need to
be less reliant of both the network
and the client-side processing**

**To deliver instant loading we need to
be less reliant of both the network
and the client-side processing**

How can the browser help with this?

A number of ways you can do this, but they all basically fall into one of two categories:

- **Prefetch**
- **Prerender**

Prefetch

- Service Workers can precache
- SPA
- `<link rel="prefetch" href="...." as="...">`
- Speculation Rules API

E.g. On a login page, prefetch the static app resources

Already registered?

Email

Password

[Forgot password?](#)

We've made some important changes to our [Data Privacy Notice](#), under the new General Data Protection Regulation.

Login

```
802
803
804
805
806
807 <!-- Prefetch Member Area Content to make this fast to Load -->
808 <link rel="prefetch" href="/memberarea/" as="document">
809 <link rel="prefetch" href="/tools/vue-memberarea/css/app.css" as="style">
810 <link rel="prefetch" href="/tools/vue-memberarea/js/app.js" as="script">
811 <link rel="prefetch" href="/memberarea/cms/config.json" as="fetch">
812 <link rel="prefetch" href="/api/address/towncounties.json" as="fetch">
813 <link rel="prefetch" href="/memberarea/cms/banner/" as="fetch">
814 <link rel="prefetch" href="/memberarea/cms/notice/claims-history/" as="fetch">
815 <link rel="prefetch" href="/memberarea/cms/subbanners/" as="fetch">
816 <link rel="prefetch" href="/memberarea/cms/subbanneryourwellbeing/" as="fetch">
817
818
819
820
821
822
823
824
825
826
827
```

<link rel="prefetch href="..." as="...">

Resource Hints: prefetch - WD

Informs the browsers that a given resource should be prefetched so it can be loaded more quickly. This is indicated using `<link rel="prefetch" href="(url)">`

Current aligned

Usage relative

Date relative

Filtered

All



Chrome	Edge [*]	Safari	Firefox	Opera	IE
4-7		3.1-13		10-12.1	
8-120	12-120	13.1-17.2	2-121	15-105	6-10
121	121	17.3	122	106	11
122-124		17.4-TP	123-125		

**Prefetch can help improve
future web page performance
but it it doesn't get that
INSTANT feel we want**

Prerender

Options for this:

- `<link rel="prerender" href="...">`
- Speculation Rules

<link rel="prerender href="...">

Resource Hints: prerender - WD

Gives a hint to the browser to render the specified page in the background, speeding up page load if the user navigates to it.

This is indicated using `<link rel="prerender" href="(url)">`

Current aligned

Usage relative

Date relative

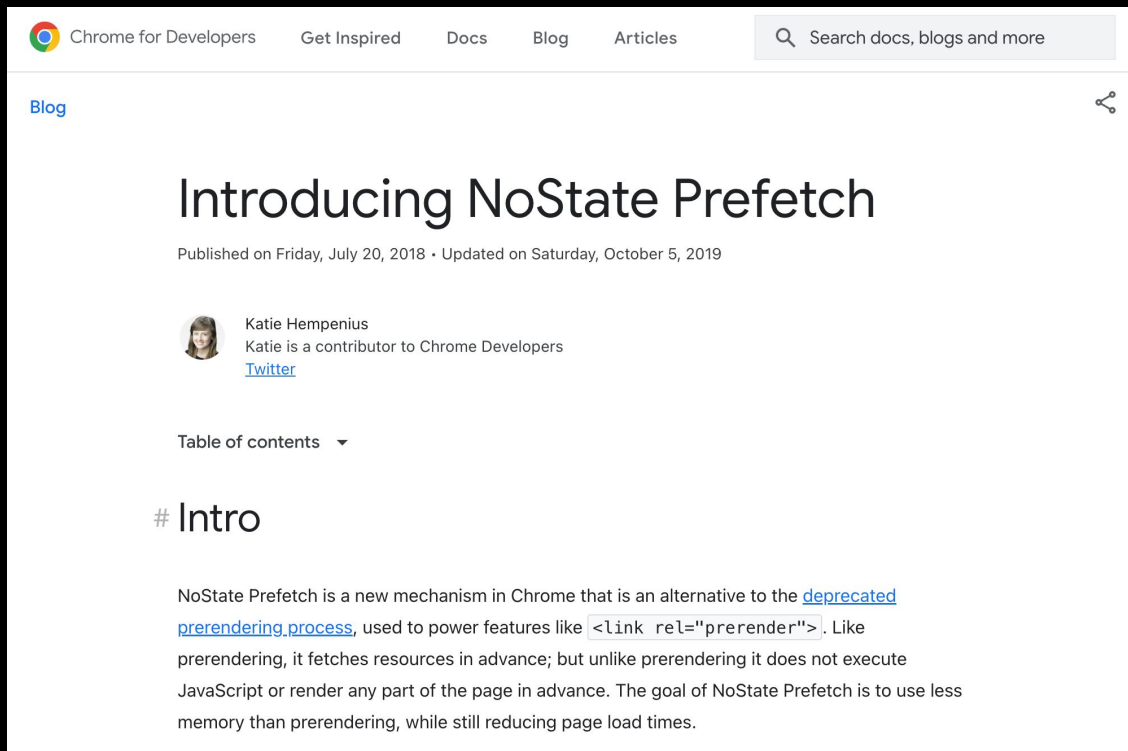
Filtered

All



Chrome	Edge *	Safari	Firefox	Opera	IE
4-12	12-18			10-12.1	
13-120	79-120	3.1-17.2	2-121	15-105	6-10
121	121	17.3	122	106	11
122-124		17.4-TP	123-125		

But it doesn't actually prerender anymore 🥲



The screenshot shows the Chrome for Developers website. At the top is a navigation bar with links for 'Chrome for Developers', 'Get Inspired', 'Docs', 'Blog', and 'Articles'. A search bar on the right contains the text 'Search docs, blogs and more'. Below the navigation bar, the word 'Blog' is displayed on the left, and a share icon is on the right. The main heading of the page is 'Introducing NoState Prefetch'. Below the heading, it says 'Published on Friday, July 20, 2018 • Updated on Saturday, October 5, 2019'. The author's name 'Katie Hempenius' is listed, along with a small profile picture and a bio stating 'Katie is a contributor to Chrome Developers' and a link to her 'Twitter' profile. A 'Table of contents' dropdown menu is visible. The section '# Intro' begins with a paragraph explaining that NoState Prefetch is a new mechanism in Chrome, an alternative to the 'deprecated prerendering.process' used for features like `<link rel="prerender">`. It notes that unlike prerendering, it does not execute JavaScript or render any part of the page in advance, aiming to use less memory while still reducing page load times.

<https://developer.chrome.com/blog/nostate-prefetch>

<link rel="prerender href="...">

Resource Hints: prerender - WD

Gives a hint to the browser to render the specified page in the background, speeding up page load if the user returns to the page.

This is indicated using `<link rel="prerender href="...">`

Current aligned

Usage relative

Chrome

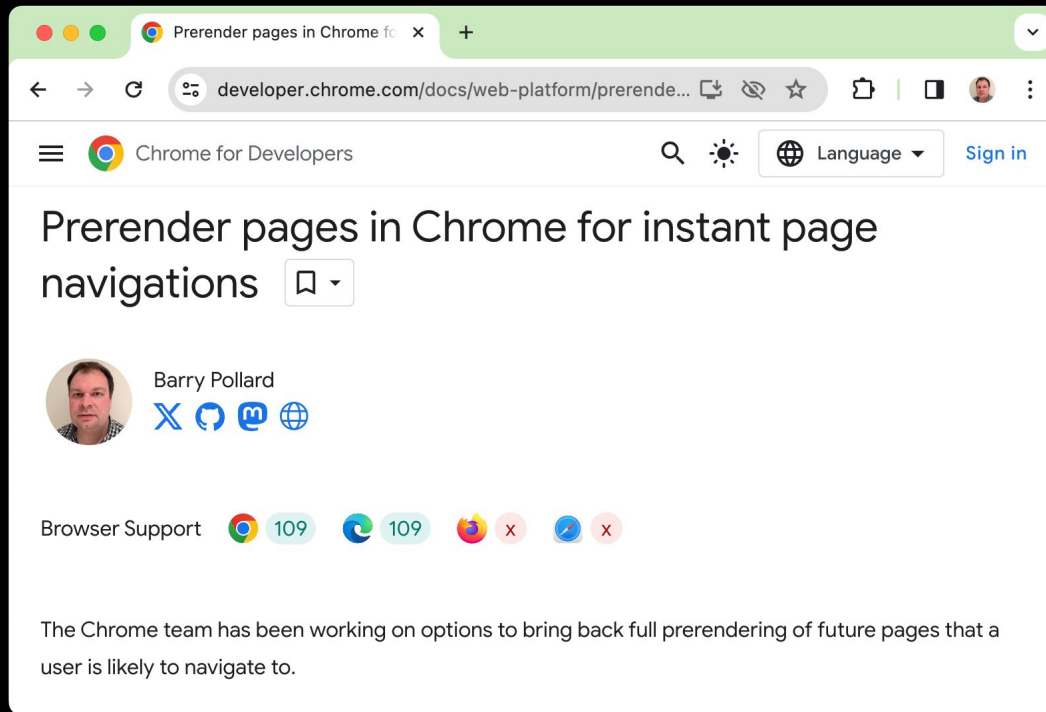
Opera

IE

			10-12.1	
1-17.2	2-121	15-105	6-10	
17.3	122	106	11	
17.4-TP	123-125			

Deprecated

For that we need the new Prerender 🎉



<https://developer.chrome.com/docs/web-platform/prerender-pages>


Speculation Rules



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "list",
      "urls": ["next.html", "next2.html"]
    }
  ]
}
</script>
```

Speculation Rules

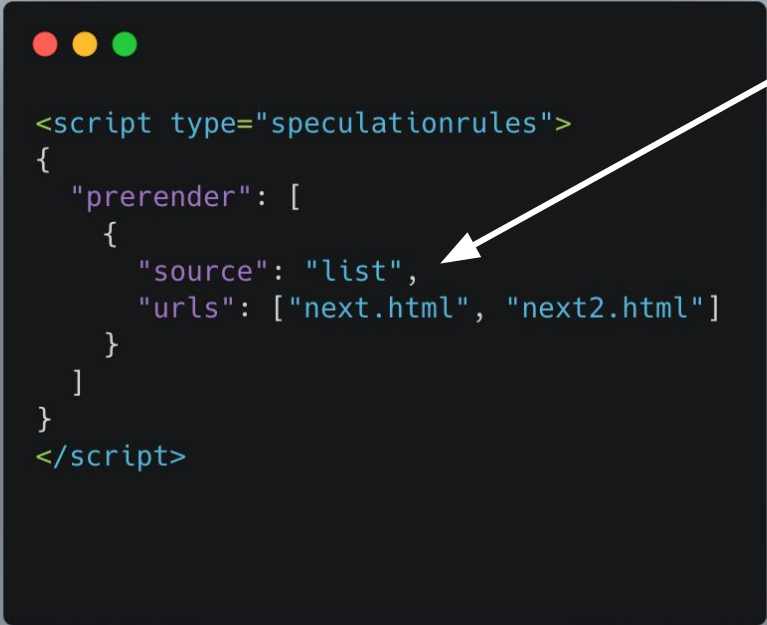
Can also
change to
“prefetch”
for
prefetching
future pages



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "list",
      "urls": ["next.html", "next2.html"]
    }
  ]
}
</script>
```

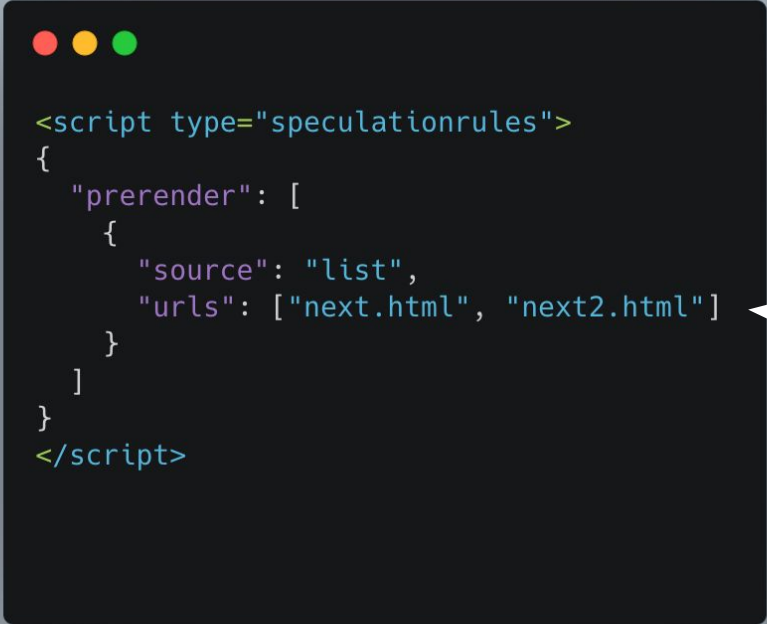
Speculation Rules

The source is
(optional from
Chrome 122)



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "list",
      "urls": ["next.html", "next2.html"]
    }
  ]
}
</script>
```


Speculation Rules



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "list",
      "urls": ["next.html", "next2.html"]
    }
  ]
}
</script>
```

 **List of URLs**

**But how can you “know”
which link a user will click on?**

**But how can you “know”
which link a user will click on?**

Isn't this wasteful?

Document rules



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "document",
      "where": {
        "and": [
          {"href_matches": "/*"},
          {"not": { "href_matches": "/logout/*"}}
        ]
      },
      "eagerness": "moderate"
    }
  ]
}
</script>
```

Document rules

The “source”
is the
document
(again this
attribute is
optional from
122)



```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "document",
      "where": {
        "and": [
          {"href_matches": "/*"},
          {"not": { "href_matches": "/logout/*"}}
        ]
      },
      "eagerness": "moderate"
    }
  ]
}
</script>
```

Document rules

```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "document",
      "where": { ←
        "and": [
          {"href_matches": "/*"},
          {"not": { "href_matches": "/logout/*"}}
        ]
      },
      "eagerness": "moderate"
    }
  ]
}
</script>
```

A “where”
object tells
you what
URLs are in
play

Document rules

```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "document",
      "where": {
        "and": [
          {"href_matches": "/*"},
          {"not": { "href_matches": "/logout/*"}}
        ]
      },
      "eagerness": "moderate"
    }
  ]
}
</script>
```

An “eagerness” setting tells you *when* to prerender:

- Eager: ASAP
- Moderate: On hover for 200ms or mouse/touchdown
- Conservative: On mouse/touchdown

Document rules

```
<script type="speculationrules">
{
  "prerender": [
    {
      "source": "document",
      "where": {
        "and": [
          {"href_matches": "/*"},
          {"not": { "href_matches": "/logout/*"}}
        ]
      },
      "eagerness": "moderate"
    }
  ]
}
</script>
```

Chrome 121!

Live demo time!!!

<https://speculative-rules.glitch.me/>

OK, but that doesn't fix
the first page load?

**Chrome uses this to
improve first page loads!**

Predictors

Chrome | chrome://predictors

Autocomplete Action Predictor

Resource Prefetch Predictor

☒ Filter zero confidences

Entries: 41

User Text	URL	Hit Count	Miss Count	Confidence
d	https://developer.chrome.com/	16	19	0.45714285714285713
de	https://developer.chrome.com/	16	6	0.7272727272727273
dev	https://developer.chrome.com/	14	4	0.7777777777777778
deve	https://developer.chrome.com/	6	0	1
https://web.dev/	https://web.dev/	5	7	0.4166666666666667
p	https://pagespeed.web.dev/	5	11	0.3125
pa	https://pagespeed.web.dev/	5	0	1
pag	https://pagespeed.web.dev/	4	0	1
page	https://pagespeed.web.dev/	4	0	1
t	https://twitter.com/	12	119	0.0916030534351145
tw	https://twitter.com/	12	30	0.2857142857142857
twi	https://twitter.com/	8	7	0.5333333333333333
twit	https://twitter.com/	3	9	0.25
twitt	https://twitter.com/	3	7	0.3
w	https://web.dev/	100	200	0.3333333333333333

**Why am I so obsessed
with “instant”?**

**It introduces new options
for web developers**

...like View Transitions

View Transitions

<https://developer.chrome.com/docs/web-platform/view-transitions/>

Live demo time #2!!!

Without Prerender

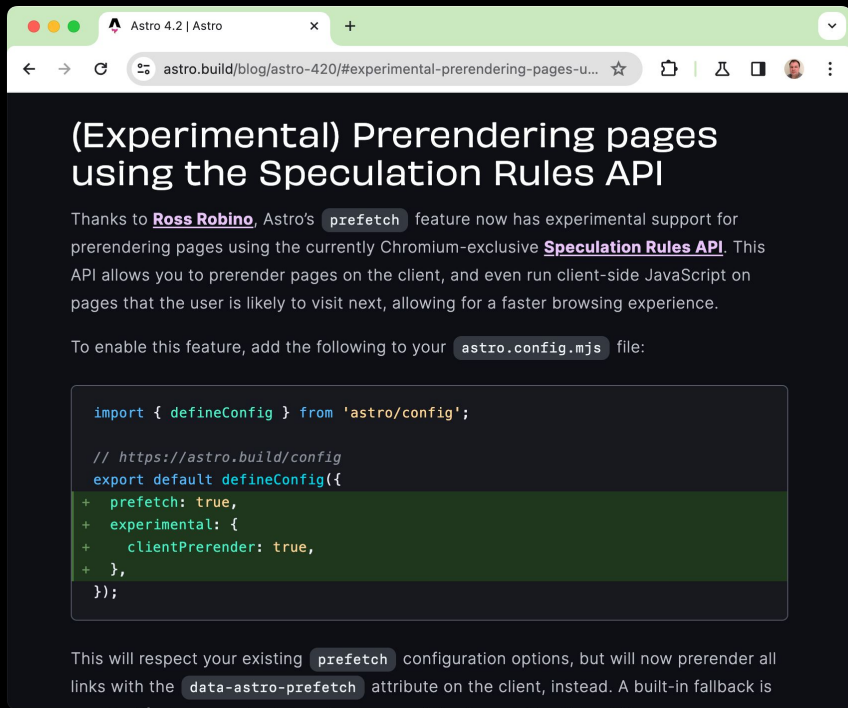
With Prerender

(needs `chrome://flags/#view-transition-on-navigation`)

**What's this got to
do with open source?**

**If you've an open source project
consider adding support for the
Speculation Rules API**

Astro has added support!



<https://astro.build/blog/astro-420/>

There's a WordPress Plugin

Feedback

Appearance

Plugins 8

Users

Tools

Settings

General

Writing

Reading

Discussion

Media

Permalinks

Privacy

Performance

Speculation Rules

This section allows you to control how URLs that your users navigate to are speculatively loaded to improve performance.

Speculation Mode

☐ Prefetch

☒ Prerender

Prerendering will lead to faster load times than prefetching. However, in case of interactive content, prefetching may be a safer choice.

Eagerness

☐ Conservative (typically on click)

☒ Moderate (typically on hover)

☐ Eager (on slightest suggestion)

The eagerness setting defines the heuristics based on which the loading is triggered. "Eager" will have the minimum delay to start speculative loads, "Conservative" increases the chance that only URLs the user actually navigates to are loaded.

Save Changes

<https://wordpress.org/plugins/speculation-rules/>

**Let's make 2024
the year of
Instant Navigations!**

Thank you

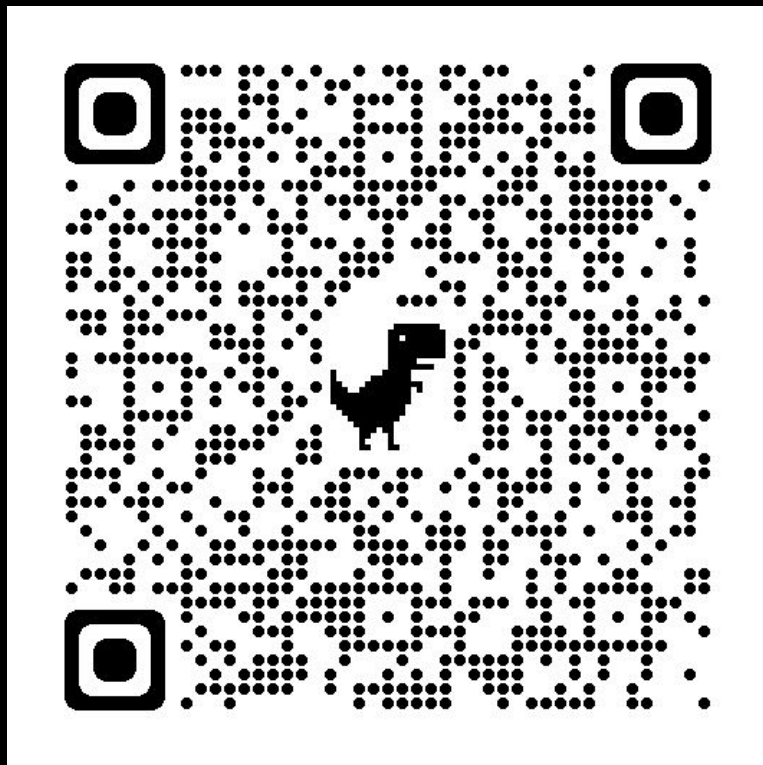
Barry Pollard



@tunetheweb



@tunetheweb@webperf.social



FOSDEM'24

Brussels / 3 & 4 February 2024