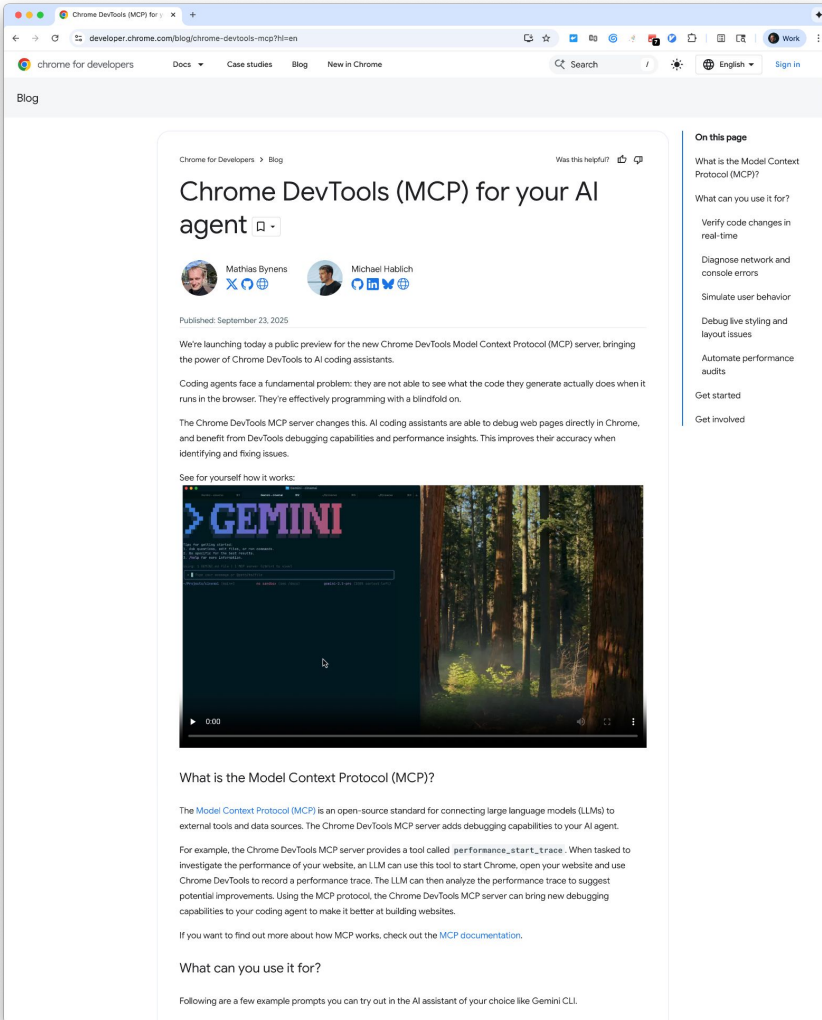




Building the Chrome DevTools MCP server

Jack Franklin, MCP Summit, October 2025



<https://developer.chrome.com/blog/chrome-devtools-mcp>



Building for AI users



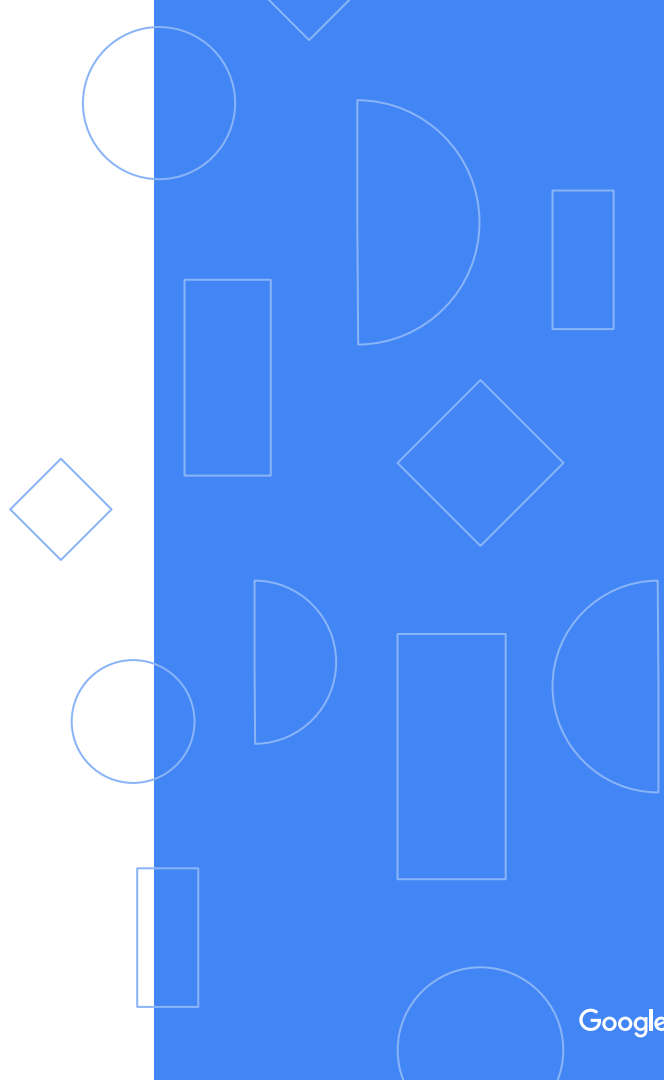
**Learning &
Abstractions**



Challenges



Building for AI users





What is our value
proposition that an
LLM can't do on its
own?



Verify in the browser that your change works as expected

Some images on localhost:8080 are not loading. What's happening?

Why does submitting the form fail after entering an email address?

localhost:8080 is loading slowly. Make it load faster.



Verify in the browser that your change works as expected

Some images on localhost:8080 are not loading. What's happening?

Why does submitting the form fail after entering an email address?

Can we enable an AI to answer these prompts well?

What can DevTools give AI agents that no one else can?



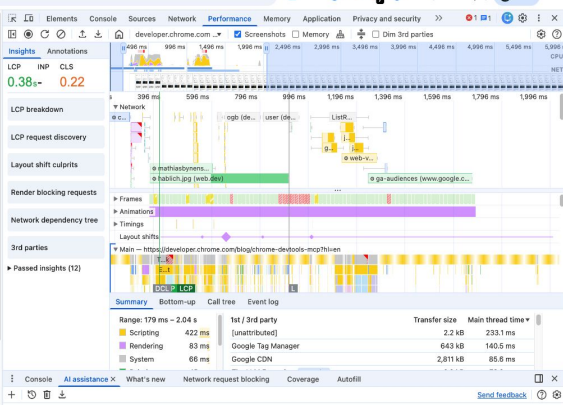
Runtime data

Relating source code to production

Enhanced debugging capabilities

Learning from AI in DevTools





Jack Franklin

What performance issues exist with my page?

AI

Analyzing trace

Based on the trace data, here are some potential performance issues with your page:

- Layout shifts:** Your page has a Cumulative Layout Shift (CLS) of 0.22. This is above the recommended threshold of 0.1 and can negatively impact user experience.
- Render-blocking resources:** There are requests that are blocking the initial render of your page, which can delay the Largest Contentful Paint (LCP).
- Large Largest Contentful Paint (LCP) render delay:** The render delay for your LCP element is 113 ms, which is a significant portion of the total LCP time (380 ms).
- Long tasks:** There are several long tasks on the main thread that are taking up significant time and could be impacting responsiveness.
- Third-party impact:** Some third-party code is contributing to main thread work and network traffic.

Which of these would you like to investigate further?

🔍 🔄 📄

Jack Franklin

Tell me more about render-blocking resources.

AI

Investigating insight RenderBlocking...

Several CSS resources are marked as render-blocking, which can delay the time it takes for your page to display content to the user. These include:

- <https://fonts.googleapis.com/css2?family=Material+Icons&family=Material+Symbols&outline&display=block>
- <https://fonts.googleapis.com/css2?family=Material+Icons&family=Material+Symbols&outline&display=block>
- <https://www.gstatic.com/branding/fonts/branding/branding.css>
- <https://www.gstatic.com/branding/fonts/branding/branding.css>

To reduce the impact of render-blocking resources, consider:

- Deferring or inlining critical CSS:** Identify the minimal set of CSS required for the initial render and inline it in the HTML. Defer the loading of the rest of the CSS until after the initial render.
- Using media attributes:** For CSS that only applies to specific media types or queries (e.g., print styles), use the `media` attribute on the `<Link>` tag to avoid blocking the initial render for other media.

Would you like me to explore any of these resources further?

🔍 🔄 📄

1

Trace: developer.chrome.com

Relevant data is sent to Google

<https://developer.chrome.com/docs/devtools/ai-assistance>

#1

Prompt engineering

#2

How LLMs use function calls

#3

Context window engineering

Representing call trees

```
# All URL #s:
* 0: https://www.example.com/spa/123.hash-7363f4688350186a92b4.js
* 1: https://www.example.com/spa/hash-8d85eb1943d61b172196.js
* 2: https://www.example.com/spa/124.hash-285098f1d43ad35851b6.js
* 3: https://www.example.com/ruxitagentjs_A7NVfghqrtux_10291240606133530.js
* 4: https://www.example.com/kfrcdn/vice_loade
* 5: https://www.example.com/spa/11.hash-262d4
* 6: https://www.example.com/spa/2.hash-a8a5bf
* 7: https://www.example.com/spa/1.hash-d8f430
* 8: https://www.example.com/spa/30.hash-30736
```

Call tree:

Node: 1 – Task

dur: 541.4

self: 0.3

Children:

* 2 – Evaluate script

Node: 2 – Evaluate script

dur: 541.1

self: 1.7

URL #: 0

Children:

* 3 – Compile script

* 4 – (anonymous)

* 5 – Run microtasks

Node: 3 – Compile script

dur: 29.3

self: 29.3

URL #: 0

Node: 4 – (anonymous)

All URL #s:

* 0: https://www.gstatic.com/firebasejs/6.6.1/firebase-performance.js

Call tree:

1;Task;0.9;0;;2;S

2;Timer fired;0.9;0;;3

3;Function call;0.9;0.1;0;4

4;(anonymous);0.8;;0;5

5;(anonymous);0.8;;0;6-8

6;Ot.getEntriesByType;0.1;;0;8

7;le.createOobTrace;0.6;0.2;0;9-11

8;getEntriesByType;0.1;0.1;;

9;le;0.1;0.1;0;

10;ie;0.2;;0;11-13

11;Ot.requiredApisAvailable;0.2;0.2;0;

12;oe;0;;0;13

13;setTimeout;0;0;;`;

Finding the right abstraction level

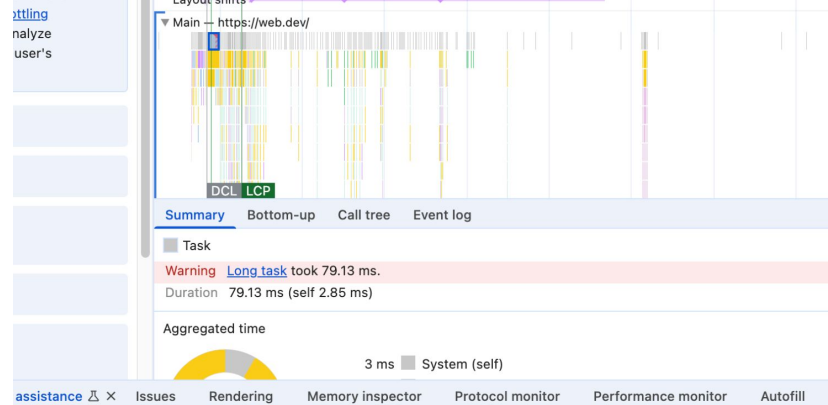
High level, powerful tools, or
low level building blocks?



Our first MCP server exposed high level tools

```
debugPerformanceProblem({  
  url: 'www.google.com',  
  prompt: 'why does the page load slowly'  
});  
  
debugStylingProblem({  
  url: 'www.google.com',  
  prompt: 'why does the page load slowly',  
  selector: 'h1.main-heading'  
});
```

```
debugPerformanceProblem({  
  url: 'www.google.com',  
  prompt: 'why does the page load slowly'  
});
```



Jack Franklin

What's the purpose of this work?

AI

✓ Analyzing call tree

The selected call frame is 'Task', serving as the root of the traced activity. It encompasses the entire workload within this performance snapshot.

The main activity within this Task is the 'Evaluate script' call, which represents the parsing and execution of scripts. This particular 'Evaluate script' call is primarily involved in processing a script from `polyfills/webcomponents-ce.js` (URL 0).

Within the 'Evaluate script' call, the following sub-tasks are performed:

- 'Compile script': This compiles the script from `polyfills/webcomponents-ce.js`. This is the most time-consuming part of the evaluation, taking 42.4ms.
- Several anonymous functions are executed from both `polyfills/webcomponents-ce.js` and `content_compiled.js` (URL 1), likely related to the initialization or setup of the scripts.
- Specific functions like `$jscomp.polyfill`, `$jscomp.polyfillUnisolated`, `registerLanguage`, `Ia`, `m.define`, `Y.m`, `m.H`, `U`, `Ø`, and `M` are called, indicating initialization and definition processes, potentially related to polyfills or module definitions.
- 'Minor GC' calls suggest memory management operations are occurring during script execution.

In summary, the purpose of this work, as captured by the selected 'Task', is to load, compile, and execute scripts, likely related to web components polyfills and extension content, including initialization, module definition, and garbage collection.

👍 🗨 🔄

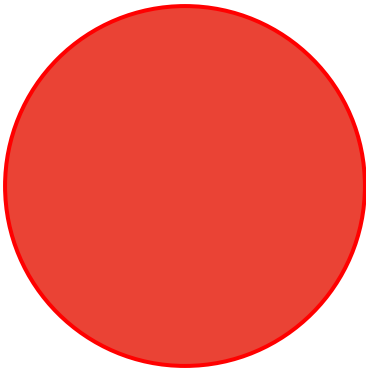
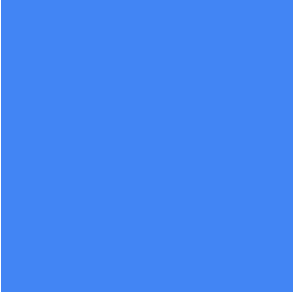
**But...we didn't find
it provided the
value we wanted**



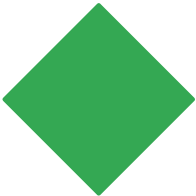
Too niche and not composable

Complex setup process

Hard when the AI client can't get at the browser



V2: expose lower
level building blocks



We focused on
tools to replicate
common manual
user actions when
debugging

`click, fill, fill_form, hover, upload_file`

`navigate_page, new_page, wait_for`

`emulate_cpu, emulate_network, resize_page`

`performance_start_trace,
performance_stop_trace`

`get_network_request, list_network_requests`

`evaluate_script, take_screenshot,`



*via code reuse of
DevTools in MCP
server*



Now DevTools can talk to a regular Chrome, with no extra setup required.

The challenges of building MCP servers and working with AI



The amount of different environments



The matrix of clients & models is vast.

VSCode

Cursor

Cline

WebStorm

Zed

*(and model specific options
like Claude Code, Gemini CLI)*



Auto	Variable
✓ GPT-4.1	0x
GPT-4o	0x
GPT-5 mini	0x
Grok Code Fast 1 (Preview)	0x
Claude Sonnet 3.5	1x
Claude Sonnet 3.7	1x
Claude Sonnet 4	1x
Gemini 2.5 Pro	1x
GPT-5	1x
GPT-5-Codex (Preview)	1x
o3-mini	0.33x
o4-mini (Preview)	0.33x
Gemini 2.5 Flash Preview	Gemini
Gemini 2.5 Pro	Gemini
Manage Models...	



Variance in
user
experience

*“Client X doesn’t call the
performance tracing
functions in the right
order”*

*“Client Y doesn’t call the
performance tracing
functions in the right
order”*

Great, I’ve landed PR#24
that updates the function
description to remove the
confusion.

Evaluating our changes



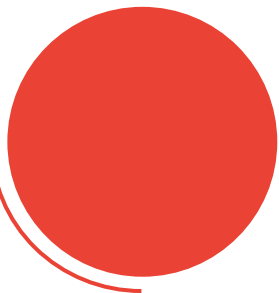
Google

PR: “Improve function
tool descriptions for
Performance tools”

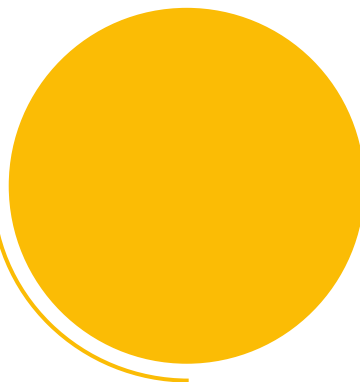
**Code
quality**



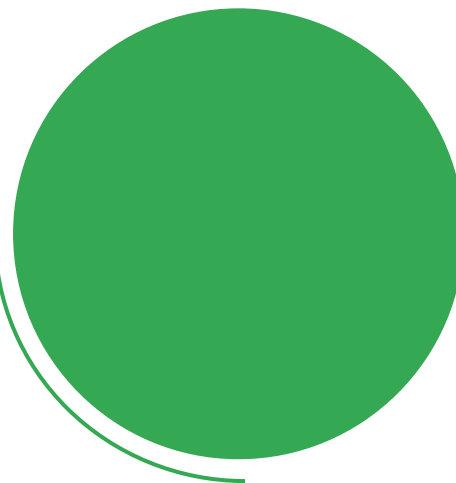
**Unit
tests**



**Well
documented**



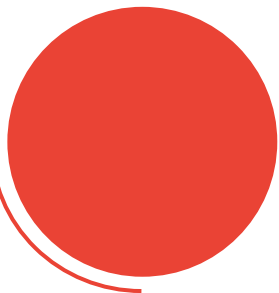
**Does it
work?**



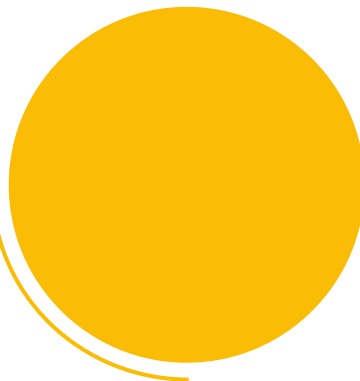
**Code
quality**



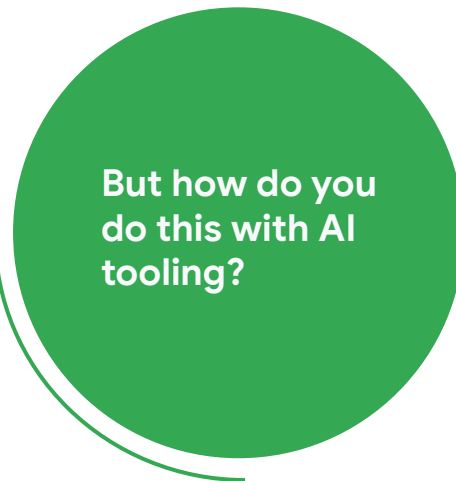
**Unit
tests**



**Well
documented**



**Does it
work?**



**But how do you
do this with AI
tooling?**

Good dataset

Representative set of websites with a variety of problems that DevTools can fix.

Effortless collection

Make it really easy to make a change and gather output from an AI interaction with DevTools.

Evaluation framework

Using “LLM as judge” and another techniques to score quality of output.

(index)	id	average	median	allScores
0	'lcp_render_delay/v1'	'2.10'	'2.00'	'-6, -1, -1, -1, 2, 2, 5, 5, 8, 8'
1	'lcp_render_delay/v2'	'1.22'	'-1.00'	'-3, -2, -1, -1, -1, 1, 2, 8, 8'
2	'lcp_render_delay/v3'	'3.33'	'4.00'	'-1, 4, 7'
3	'lcp_render_delay/v4'	'6.63'	'8.00'	'2, 2, 2, 5, 5, 7, 7, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8'
4	'lcp_render_delay/v5'	'7.00'	'8.00'	'2, 5, 8, 8, 8, 8, 8, 8, 8, 8'
5	'lcp_render_delay/v6'	'6.00'	'8.00'	'-1, 2, 5, 5, 5, 8, 8, 8, 8, 8, 8, 8, 8, 8'

Supporting
richer output



Google

5.2.1 Text Content

```
{
  "type": "text",
  "text": "Tool result text"
}
```

5.2.2 Image Content

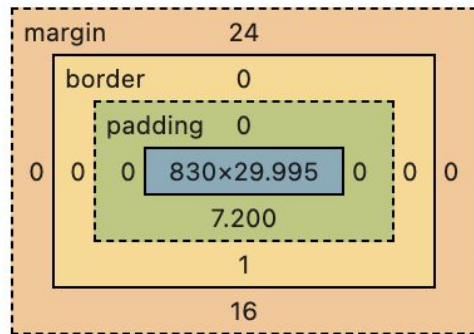
```
{
  "type": "image",
  "data": "base64-encoded-data",
  "mimeType": "v
  "annotations": {
    "audience": "(a...s) s (anonymous) s (anonymous) a.O (anonymous) (anonymous) _webpac...q
    "priority": 26..3 36553 :
  }
}
```

This example demo

5.2.3 Audio Conf

```
{
  "type": "audi
  "data": "base
  "mimeType": "
}
```

- Seems like a good fit for web debugging
- ...but what if the user is in a terminal client and cannot view images?



The pace of change



The movement in this space is not slowing down



MCP specification changes and new capabilities

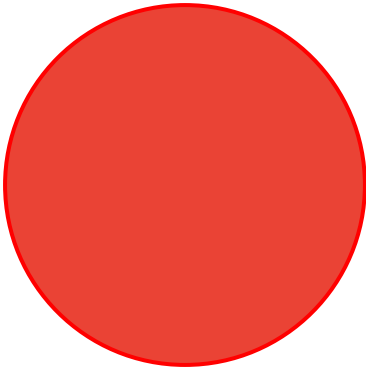
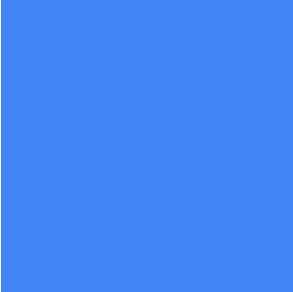
Elicitation, authentication, and much more.

New model capabilities

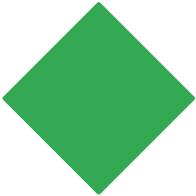
Models continue to improve and enlarge their supported context window sizes.

User expectations and patterns

The entire industry is figuring out how best to work with AI tooling and MCP servers.



What's next for
DevTools & MCP?



Please give us your feedback!

We want to continue evolving our MCP server to provide the best debugging experience for you and **your AI agents**.

Your feedback is extremely valuable and helps us prioritise features and fixes.

Please feel free to raise issues and tell us what's missing.

Chrome DevTools MCP

`chrome-devtools-mcp` lets your coding agent (such as Gemini, Claude, Cursor or Copilot) control and inspect a live Chrome browser. It acts as a Model-Context-Protocol (MCP) server, giving your AI coding assistant access to the full power of Chrome DevTools for reliable automation, in-depth debugging, and performance analysis.

[Tool reference](#) | [Changelog](#) | [Contributing](#) | [Troubleshooting](#)

Key features

- **Get performance insights:** Uses [Chrome DevTools](#) to record traces and extract actionable performance insights.
- **Advanced browser debugging:** Analyze network requests, take screenshots and check the browser console.
- **Reliable automation:** Uses [puppeteer](#) to automate actions in Chrome and automatically wait for action results.

Disclaimers

`chrome-devtools-mcp` exposes content of the browser instance to the MCP clients allowing them to inspect, debug, and modify any data in the browser or DevTools. Avoid sharing sensitive or personal information that you don't want to share with MCP clients.

Requirements

- [Node.js 20](#) or a newer [latest maintenance LTS](#) version.
- [Chrome](#) current stable version or newer.
- [npm](#).

Getting started

Add the following config to your MCP client:

```
{
  "mcpServers": {
    "chrome-devtools": {
      "command": "npm",
      "args": ["-y", "chrome-devtools-mcp@latest"]
    }
  }
}
```

Note

Using `chrome-devtools-mcp@latest` ensures that your MCP client will always use the latest version of the Chrome DevTools MCP server.

MCP Client configuration

- Claude Code
- Cline
- Codex
- Copilot CLI
- Copilot / VS Code
- Cursor
- Gemini CLI
- Gemini Code Assist
- JetBrains AI Assistant & Junie
- Visual Studio

<https://github.com/ChromeDevTools/chrome-devtools-mcp>



Thank you



<https://github.com/ChromeDevTools/chrome-devtools-mcp>

