

Read the Tea Leaves SOFTWARE AND OTHER DARK ARTS, BY NOLAN LAWSON

23 AUG

The asteroid currently hitting frontend web development

Posted August 23, 2026 by Nolan Lawson in Web. Tagged: AI. [17 Comments](#)

A lot of the educators I admire in the frontend web space seem to be either bowing out or dialing back their efforts: Axel Rauschmayer (<https://exploringjs.com/>), Salma Alam-Naylor (<https://whitep4nth3r.com/blog/goodbye-forever-probably/>), Josh W. Comeau (<https://bsky.app/profile/joshwcomeau.com/post/3mkxyqfwwm22t>), to name a few. Other well-known luminaries like Kent C. Dodds (<https://kentcdodds.com>), Addy Osmani (<https://addyosmani.com/>), Rachel Nabors (<https://nearestnabors.com/>), and Lydia Hallie (<https://x.com/lydiahallie>) have pivoted from talking about frontend development to talking about… well, take a wild guess.

I never made a living from talking about web dev, but I do have this blog, and I've done conference talks and podcasts about stuff I'm excited about. Recently for example, my passion was explaining all the ins and outs of CSS performance: how shadow DOM works (<https://nolanlawson.com/2022/06/22/style-scoping-versus-shadow-dom-which-is-fastest/>), how the browser's style engine works (<https://nolanlawson.com/2023/01/17/my-talk-on-css-runtime-performance/>), the traps of CSS-in-JS (<https://nolanlawson.com/2022/10/22/style-performance-and-concurrent-rendering/>), etc.

Just for kicks, I decided to ask Claude Sonnet how it would handle a Chrome trace with high “Style Calculation” costs. This is one of my favorite brain-teasers, because it's the kind of thing that I saw web devs – even experienced ones! – trip up on over and over again. Claude gave a perfectly admirable answer:

► [Click to see](#)

These days, given a slow-loading website, even as someone who's written about browser performance for years and who literally *worked on a browser performance team*, I would probably just chuck a Chrome trace at Claude Code and have it suggest improvements. In fact I've done this very thing in my day job and gotten some good results.

The future of frontend

So where does this leave frontend dev education? Not in a great place obviously; I wish I had some more uplifting answers for people who (like me) used to get a lot of fulfillment out of trying to raise the bar for frontend developers everywhere. I do have some guesses though, and I think the problem is still worth puzzling through.

The core question is where frontend development itself lands in this new era. Sadly it feels to me like there are several trends pointing against increased investment in frontend knowledge:

The frontend is less risky to just hand to an agent. If you're using an agent to write a database migration, you probably want to put it through several rounds of AI code review, scrutinize it yourself, run it on staging first, etc. If you write a React component with an agent, though, then the risk of just yolo'ing it into production is (typically) much lower.

Note I'm *not* saying there are zero risks: the agent could mess up accessibility, it could cause an infinite loop that blocks users, etc. But in general, frontend code is a lot more ephemeral and replaceable than other types of code. So I expect many AI coders will feel comfortable just letting their agent handle it unsupervised (for better or worse).

DevExp is becoming less critical overall. A lot of the pre-LLM discussion in the frontend space was about ergonomics versus outcomes: “The ‘developer experience’ bait-and-switch” (<https://infrequently.org/2018/09/the-developer-experience-bait-and-switch/>) by Alex Russell is a great example. For another example, Svelte and Solid have long argued that their ergonomics lead to better outcomes than React: less code, better performance, etc.

Meanwhile, Cursor (<https://x.com/poteto/status/2089227731305464150>) and Viget (<https://www.viget.com/articles/using-ai-to-migrate-from-lit-to-react>) have blogged about migrating their codebases from Solid and Lit, respectively, to React. Since rewrites are less expensive with agents, this may be a bit surprising: why not move to the more performant/less verbose framework? The answer (explicitly in Cursor's case, and I suspect for Viget as well) is of course: “the agents know React.” For better or worse, React is heavily overrepresented in the training weights, and “agent experience” is starting to matter more than developer experience.

Standards will catch up. I've been out of the web standards space for a couple years now, so this is pure speculation on my part. But I imagine that a lot of the efforts to improve the ergonomics of building websites – better CSS shorthands, terser JavaScript syntax, etc. – will become perceived as less important relative to things that actually move the needle on performance, capabilities, etc. At the end of the day, it's just not very different for an agent to write 3 lines of CSS instead of 1, and anyway using the newer syntax might actually be *harder* because you have to coach the agent about things that aren't in its training weights.

In some ways this shift might have already been underway. I remember several years ago at TPAC (<https://www.w3.org/wiki/TPAC>), well before the AI coding boom, I told someone on the Chrome team that I was working on web component standards. They responded that they weren't interested in that, because those APIs only affected the developer experience and didn't actually make the browser more

capable (e.g. Project Fugu (<https://www.chromium.org/teams/web-capabilities-fugu/>)). That stuck with me because it's a good point: APIs like shadow DOM and custom elements don't give web developers any new superpowers; they just change where and how the code gets authored. I expect such things will move out of the spotlight as AI coding takes over.

This doesn't mean that standards will disappear from the topics a frontend dev needs to keep up on, but I imagine it will become less about "use this newer syntax" (an evergreen source of material for conference talks and articles) and more "here are these emerging capabilities." And I predict the latter group will be much smaller than the former, since they tend to be more contentious for standards bodies and there's just a smaller pool of features to draw from.

Whither frontend education?

So how can the field of frontend education adapt to this hostile future? To avoid being utterly bleak, here are some positive directions I think it could go in.

First off, the agents still need to be educated about the big picture. Agents and harnesses seem to love writing React and specifically SPAs, but SPAs are not the answer to everything (<https://nolanlawson.com/2022/05/21/the-balance-has-shifted-away-from-spas/>). You can burn a lot of tokens having an agent write a big complex SPA for your marketing site, and then fix all the bugs with the back button, focus state, performance, etc., or you can just choose an MPA framework like Astro or Eleventy and call it a day. Maybe these frameworks will be a bit harder for the agents to work with (specifically Astro since it kinda-sorta looks like React but isn't), but my guess is that since you're writing ~50% less code overall it won't matter.

Second, making websites that work well for agents is probably going to be a fruitful endeavor for the near future. Vercel's is-agentic (<https://is-agentic.com/>) is a good example of this. Ironically, this points back to good fundamentals that public-facing websites should have been doing anyway: server-rendered content, proper accessibility, page speed, etc. But if slapping the word "AI" on it is what gets people to care about it then hey, I'm all for it.

Note that I'm a little bit less sanguine about this second point, because I'm not sure the web even survives in its current form as agents become more of a thing. If I want to figure out how much it costs to fly from Seattle to Paris, I'd much rather ask an agent than click through an infuriating series of buttons on a slow-loading website. The only reason I can't is because these websites explicitly block bots, or they don't offer an MCP, but I'm sure there are several startups champing at the bit to solve that problem. So I'm not sure how sustainable the current situation is.

Third, we can offer consulting services for vibe-coded monstrosities. A massive amount of AI-generated frontend code is being pumped out right now, and some of it (to use a Claude-ism) will certainly become "load-bearing." If those websites are slow, non-compliant, and riddled with security holes, then it may not be enough to ask the agent "fix my website pls." There could be an opportunity here for real expertise, especially if there's money on the line and the vibe coder's knowledge of web development doesn't extend past "websites are apps hosted on the internet."

(I acknowledge that this is the shakiest of my three points, since I can totally imagine the next generation of “self-healing” web apps to eclipse the average expert in 2027 or 2028. But for the time being: yes, expertise still matters.)

Conclusion

The point of this post wasn't to make myself feel better, or to dance on the graves of all the careers that have been upended by the recent AI boom. I'm a naturally gloomy person, and this post was me allowing myself to wallow in my own gloominess. I don't take any pleasure from noting that the huge body of knowledge I've built up over the years has been rendered nearly obsolete, nor am I happy to see the same thing happen to my much-more-qualified peers. But pretending that it's not happening isn't a valid strategy either.

There's a mood in some of the blogs I read these days along the lines of “I'm so tired of talking about AI” or “Please don't mention AI to me ever again.” I'm sure some of this is a kind of world-weary, above-it-all air that feels good to wear as a badge of distinction. But I think a lot of it also comes from real fear. It's scary to admit that you don't know what's going to happen in a year. It's destabilizing to imagine your career going along a certain trajectory, serenely landing at retirement, and then to see everything upset just a few years from your goal.

The metaphor I've been using is that an asteroid just hit the earth, and we're still surveying the wreckage. It's hard to predict what will happen after the dust settles (let alone which tiny rodents will usher in the Age of Mammals!), but ignoring the crater altogether seems like the worst kind of denial. Another metaphor is covid: when covid hit, I don't recall thinking, “Ugh, I'm so tired of talking about covid” – instead, I wanted to learn everything I could about viruses, epidemiology, masking, etc. This turned out to be a good idea, since covid was going to dominate my life for the next few years (at which point yes, I did finally get tired of talking about it!).

I hardly have a crystal ball, but this post was my attempt to think through where my most cherished field might be going in the future. I admit that I have a lot less skin in the game these days: I'm out of web standards, I don't even work on the frontend at my current gig, and my blog has mostly been a lot of wailing and gnashing of teeth about AI rather than my usual menu of browsers, performance, accessibility, etc. That said, I still have a lot of love and respect for the frontend field, and I care about what happens to it in the future. It may be unrecognizable in just a few years, but if nothing else, I hope my peers find a way to navigate all these changes and to thrive in this weird new world.

17 responses to this post.

Posted by Rob Mathews on August 23, 2026 at 4:43 PM

Front end development is dead.

[Reply](#)

Posted by Sander van Dragt on August 23, 2026 at 11:02 PM

thanks for an interesting article. As a web developer who did not have the understanding that you received from the long agent I was a bit surprised with the conclusions you've drawn.

I feel you've put what is a reasonably deep technical question (for the average web developer) to the ai agent, I learned a whole lot from the answer, and got a lot of context to follow up on, but instead of seeing the opportunity for learning the conclusion seems to be that it's too easy to just ask the agent? (In a simplified way)

This points to one of the things that keeps surprising me. Why not ask the ai to help you learn, follow up on pros and cons of the answer, inspect it from different angles? Ask for examples, what the pros and cons are of each proposed next action? Instead the argument seems to be that if you use it like a better stack overflow, you get ready to use answers that are not much better than it.

It has the opportunity to bring a lot of people into a field that they are casually familiar with, and be a mentor for those who are starting out. Ask it for 5 lessons on Shadow Dom with a breakdown followed by 3 exercises each (I've created a skill at svandragt/lessons to help with this) and if you work through this you can have personalised follow-ups adapted to your learning path.

I think you get out what you put in to a certain extend.

Reply

Posted by Nolan Lawson on August 24, 2026 at 8:16 AM

I did cover this a bit in a previous post ("Using AI to write better code more slowly"), but I don't totally disagree with you. Agents can be used as more than shortcuts: they can be used as teaching aids, as you describe. And I agree that the agent's answer was surprisingly thorough and well-researched (including some things that aren't well known, like the cost of duplicating styles across shadow trees!).

I think the challenge here for educators is 1) when presented with an "easy" button, many people will click it rather than do the work (we can see this with how AI is disrupting homework), and 2) as you've stated so well, agents are already excellent teachers, infinitely patient and tuned to their students' exact questions. In this world, it seems hard to compete in the traditional way with courses, articles, tutorials, etc.

I think the other point I elided is that I wouldn't stop with just asking the agent a question: I would give it the Chrome trace, the source code, and a browser, and let it write benchmarks and iterate on the solution. In a world where you can just type "make it fast" and be presented with a benchmark a few minutes later proving that code A is faster than code B, it seems hard to justify the investment of learning how to do such analyses yourself. Although of course there are still cases where the agent hallucinates, misses something, etc., so judgment is still valuable for now.

Reply

Posted by Quebo on August 24, 2026 at 12:04 PM

What a silly article and I didn't even read it. Backend is more deterministic and easier for AI to take over, expected some data-in and some data-out, the strength about backend is that nobody knows that exists and C-level peeps don't understand it neither. nobody can see or understand errors in the backend.

The frontend is harder, dealing with way too many complex things, screen sizes, colors, resolutions, and the hardest one the human eyes, the weak side of frontend is that everyone can see it and have opinions even CEOs and C-level peeps, everyone can see a miss-aligned card or error.

Even harder is design, designing things is really hard, if AI can design and do frontend work then backend is a piece of cake, it can invalidate cache, services or call what ever things

Reply

Posted by Nolan Lawson on August 24, 2026 at 1:25 PM

At the micro-level I might agree with you, but once you start talking about Kubernetes, backoff, retries, auto-scaling, load, database transactions, queues, microservices, race conditions, etc., I think the backend is full of a lot more things that can go catastrophically wrong and cause an incident. In a typical software company, I'd imagine you're much more likely to get paged at 2am by a backend problem than a frontend problem. (Frankly this is something I loved about frontend for so many years; it felt much less like a high-wire act!)

That said you might argue that the above is self-inflicted, and that simple monoliths are less prone to these kinds of problems than microservice architectures (and I'd agree with you), but the same affliction also affects the frontend, where SPAs can make things needlessly complicated compared to MPAs (as I mention in the article).

I do think you're absolutely right though that design is one of those areas that's very resistant to disruption by GenAI. Agents today have very poor taste and don't understand the customer requirements or know how to properly think through user flows. And they don't have eyes. So yeah I think pivoting to more of a design focus could be a great career move for a lot of frontenders.

Reply

Posted by keira on September 1, 2026 at 8:45 AM

How, precisely, are you able to comment on something you haven't bothered to read?

The article is about educating people on front-end becoming moot, so.... good analysis?

Reply

Posted by mbavio on August 26, 2026 at 12:43 PM

On the matter of education, I feel something similar to what I feel about AI writing: even if AI could definitely surpass the abilities of most humans, I still find joy on reading (and learning) from real people with real life stories.

Reply

Posted by Søren Frederiksen on August 26, 2026 at 10:34 PM

Good read. Interesting times 🧐

Reply

Posted by DHPersonal on August 27, 2026 at 7:37 AM

It's destabilizing to imagine your career going along a certain trajectory, serenely landing at retirement, and then to see everything upset just a few years from your goal.

If it can be disrupted once then it can be disrupted again. Everyone is so quick to say “AI is here to stay” when it wasn’t here just a few years ago. Public of AI is not improving — if anything, it’s getting worse — and quality of work via AI is not meeting expectations across the industry in ways that the AI providers declared. The cost of AI is so incredibly unrepresented in the current token costs — prices that are already going up and *have to* because it’s insanely expensive to run these systems — that we are fools to assume that the present is now the future, that this can’t change once again when the bubble bursts. We’ve been through this before: Covid-19 had the big tech companies over-hiring because they were certain the current way of things was the forever way of things. Let’s not get so hasty into believing front-end web development is over because some sales people brought out a shiny new toy.

Reply

Posted by DHPersonal on August 27, 2026 at 7:38 AM

Sorry, that should be “public *opinion* of AI” in my previous post.

Reply

Posted by Nolan Lawson on August 27, 2026 at 11:41 AM

Yeah, I definitely think things will change dramatically when the bubble pops. What I don’t believe is that LLMs or AI agents will dissolve into the ether when that happens. I’ve already gotten a lot of value out of open-weight models: Kimi K3 for example is pretty adequate for routine coding tasks in my experience. And models seem to be coming out that can run on consumer hardware. If anything, the current frenzy seems to be driving a race to the bottom on token costs, although I’m not an expert on the economics of the situation so I don’t really want to speculate. It feels like for better or worse, the technology is here to stay.

Posted by Ario Setiawan on August 27, 2026 at 8:16 AM

About “AI Tiredness” , i facing too, even i stop open youtube because of too many youtubers make content about AI

Reply

Posted by Short notes on tech 35/2026 - Rule of Tech on August 28, 2026 at 12:49 AM

[...] The asteroid currently hitting frontend web developmentGreat writeup of how AI affect frontend web development. “The core question is where frontend development itself lands in this new [AI] era. Sadly it feels to me like there are several trends pointing against increased investment in frontend knowledge” tl;dr; The frontend is less risky to just hand to an agent.; DevExp is becoming less critical overall.; DevExp is becoming less critical overall. [...]

[Reply](#)

Posted by The asteroid currently hitting front end web development — Getlan on September 3, 2026 at 12:37 PM

[...] Источник: Hacker News [...]

[Reply](#)

Posted by 정동우 on September 3, 2026 at 1:47 PM

I agree with you. I'm an independent developer based in Korea, and in the past, there were plenty of tasks that developers like me could take on. I mainly worked on creating ad landing pages. But these days, most of those small jobs are being replaced by AI, and my income has dropped by nearly half.

[Reply](#)

Posted by The asteroid currently hitting front end web development - on September 3, 2026 at 2:33 PM

[...] September 3, 2026  Sourced FromHacker NewsRead Full Article → [...]

[Reply](#)

Posted by GPT-6 Astra (8/10) - Arakava News on September 3, 2026 at 10:36 PM

[...] Original Post [...]

[Reply](#)

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)

Blog at WordPress.com.