

△Stop Making TUIsvibecodingsockpuppet.org

25via🌐tusharhero3 days ago | caches | 30 comments | +29, 4 spam

You must be logged in to leave a comment.

PostPreview

🇩🇪 creesch3 days ago

But building good UI is very hard: this kind of code is tedious, repetitive, exacting, and gated by platform conceptual knowledge. It takes years to get good at this kind of work. Which is why I would never hand-write this program. Instead, I summoned it.

And here they lost me, as LLMs are also not good at making well designed coherent user interfaces. They make slick looking prototypes and demo material but actually making a good UI with good UX takes a holistic approach and a visual understanding LLMs fundamentally do not have. It is already difficult to make them adhere to the same patterns and practices in the area they are good in (code).

Anyway, that is where the author lost me. I did continue reading a bit but the article reads a bit like a feverish ramble as it jumps all over the place with things like this

I'm not packaging this application up. If you want it, just screenshot this section of the post and give it to Claude. It'll build something useful. You see where I'm going with this.

No I am not, you just went on a tangent about something and then make a sudden statement about not packaging it up. There is all sorts of assumed context and subtext the author takes for granted that is not being introduced. It honestly reads a bit like this

[−]👤 alanmeira3 days ago

△31

I'm not packaging this application up. If you want it, just screenshot this section of the post and give it to Claude. It'll build something useful. You see where I'm going with this.

This is really completely out of touch. AI psychosis. Some people think their prompted slop is so good that other people want to copy them.

[−]👤 bmo2 days ago

△8

But building good UI is very hard: this kind of code is tedious, repetitive, exacting, and gated by platform conceptual knowledge.

Building good UI *is* hard, but none of these reasons are actually why! The platform coding details are the price of entry (which arguably the AI can abstract away for you) but creating a good UX is the entirety of the hard part. It's what separates good from bad, and it's what LLMs are actually quite bad at. So if you lump together using an LLM to avoid the need to know *how* to code it, and letting the LLM

design your UI, you're going to end up with something mediocre, as we're all seeing constantly right now.

[–]  miro 3 days ago

△
6 LLMs are absolutely atrocious at making user interfaces so I second your sentiment. With TUIs being relatively simple and straightforward it cannot produce a ball of mud with a bow on its head that it usually does with websites (thanks to this I get an ick about websites that look even moderately like they were generated by an LLM).

Like yes it can probably create a nice demo website, but tell it to a navigation bar with slightly unconventional structure and a modal window and you will loose your mind.

I spent some time using it for prototyping a [rotary mixer](#) interface with Raylib and the results were hilariously bad.

[–]  simonw 3 days ago

△
11 | *LLMs are also not good at making well designed coherent user interfaces*

This really depends on what your line for a good UI is.

I will happily argue that LLMs can produce user interfaces that are *good enough* for many personal tools.

Here's [what I got](#) a few months ago for my taskbar bandwidth monitoring app. It's certainly good enough to solve the problem I wanted to solve.

[–]  creesch 3 days ago

△
28 Simon, please don't. You don't have to defend the line for LLMs at every single turn certainly not when you set an arbitrary line yourself to argue from. To be frank it is a bit tiresome and muddies the waters.

No doubt LLMs can produce something that is "good enough" for a personal project. But if you look at the bit I quoted and the way I responded to it I'd hope it is clear enough that is not the context this was being talked about.

[–]  joshka 2 days ago

△
9 But good enough UI was the author's bar - especially for their chosen field of personal apps for themself, so it feels to me like you're the one straw manning the argument here, not Simon.

[–]  simonw 2 days ago

△
10 I stand by my comment. I think it would be a mistake for people to read your comment and assume that "LLMs are also not good at making well designed coherent user interfaces" is a universal truth. I presented my screenshot as a counter-point. That's how online conversations work!

[–]  creesch 2 days ago

△
26 Again, the author made a more complex argument. One I included as quote. Again, I am not arguing about models being able to do "good enough", I know they can.

| *I presented my screenshot as a counter-point.*

But if you insist, from your screenshot alone:

- Too many different font sizes for visually similar elements.
- Alignment of elements is inconsistent
- Missing information about what various values mean:
 - Your total down and up seems to be less than what is shown in the per process bandwidth. Could be alright, but does not become clear by just looking at it.
 - Similar, there are 31 connections, 3 LAN connections yet further down the block with LAN / Local is empty. This is either an error or they have different meaning that is not made clear by the interface.

Just some things that stand out at first glance. Yes, for a personal app it is likely good enough. However, a human with two functional eyeballs and a little bit eye for detail could already have made improvements on this if we wanted to take it further. The argument that you get a highly more superior UI by just summoning it, as the author quite literally says is simply false and ridiculous.

Which is also why I stand by what I am saying. The author wrote what they wrote and that simply does not hold up. There is really no need to defend this fever dream of a blog post simply because it is about AI usage. Sometimes a bad article is just a bad article.

[–]  simonw 2 days ago

△
1 I'm glad that my shared screenshot helped produce a more useful conversation than words alone.

[–]  alanmeira edited 2 days ago

△
5 | *Simon, please don't. You don't have to defend the line for LLMs at every single turn certainly not when you set an arbitrary line yourself to argue from. To be frank it is a bit tiresome and muddies the waters.*

I'm getting tired of using the "Me-too" and spam flags without any effect. I can understand HackerNews not doing anything (and even promoting his stuff), but lobsters should be different.

[–]  simonw 2 days ago

[Comment removed by author]

 kryptiskt edited 3 days ago

If I make a TUI it's not because it's easier to make than a GUI because it's not, there are easy ways to make GUIs, even before AI. It's because I want it right there in the terminal, without the distraction of a separate window, and I want something more expansive than a CLI.

[–]  animatronic 2 days ago

△
6 I want to use my programs over mosh, inside of tmux.

 panekj 3 days ago

| *People will tell you that TUIs are accessible.*

Who are those "People"? I'm only mildly following various terminals development spaces and I've never seen that claim.

🌿 marginalia edited 3 days ago

TUIs are a consequence of the unmitigated dumpster fire that is GUI programming in the 2020s.

In the '90s making a GUI was as easy as drag and drop in Delphi or Visual Basic. Children could design a native-looking GUI. I made GUIs in these tools for fun when I was 11 years old.

30 years later GUI programming sucks so much it's not a bad option for cleaning the floor mats in your car. The dozen or so frameworks that exist all suck and every single one looks slightly different.

Designing for an unknown DPI sucks. Designing UIs not knowing whether the user will be using a mouse cursor or a finger, a real keyboard or a virtual, on a horizontal or a vertical screen sucks. The resulting GUIs suck too.

TUIs win because you can make strong assumptions about how the user interacts with the software, and are thus you can make UIs that don't suck.

[–] 🍄 k749gtnc9l3w 3 days ago

△
6 I am afraid that even stronger claim might be true: TUIs are good because they are unacceptable to people who push the GUIs down the slippery slope. TUIs require interaction design but strongly limit visual design. TUIs do not look corporate enough nowadays. That alone is already enough of an advantage...

🌑 fiatjaf 3 days ago

I've been trying to like TUIs for years, but I've concluded I really don't. They have all the disadvantages of GUIs and all the disadvantages of CLIs.

[–] 🍄 dsr 3 days ago

△
6 You might change your mind by learning Lotus 1-2-3.
(<https://github.com/taviso/123elf> if you want to spend some time.)

The limitations forced advantages for the skilled user that were lost in the transition to GUIs. I don't say that they could *not* come back, but I do say that very few, if any, GUI programs learned those lessons.

🍄 technojamin 3 days ago

While I agree that I've seen a lot of people make TUIs where a proper GUI would be more appropriate (especially with the rise of TUI frameworks), I can think of many advantages of TUIs that this post seems to ignore:

- Cross-platform compatibility
- Packaging a CLI and a TUI in a single command (see [dua](#))
- Providing interactive pieces of a workflow within a shell (see [fzf](#))
- Composability with CLIs (see [lf](#))

[–] varesa edited 2 days ago

△

2 They also work elegantly over SSH connections on remote development environments and such
EDIT: that was mentioned in the post so doesn't belong into the list of ignored advantages. My bad

[–]  k749gtnc9l3w 3 days ago

△ Indeed, about integrating workflows; I have a workflow which is Vim based with frequent calls to fuzzy
2 matcher, I started with rofi and switched to fzf.

faassen 3 days ago

Okay but then I have to evaluate which Rust UI toolkit I shall use and I will never start.

Maybe maybe I should just use Electron (cue a million blog posts about resource usage).

The argument about using AI tooling making it easy to build a GUI also makes starting a project with a TUI a low risk, low decision good first step. If you later transition to a GUI it will have forced you to have worked out your keyboard interactions real well, along with compact information design.

sunshowers 2 days ago

FWIW the biggest benefit of TUIs is that constraints spur creativity. The primarily text-mode interface means that things like text boxes have to support text manipulation for power users. For example, Claude Code and Codex both support history search with ctrl-r and dropping into vim/other text editor with ctrl-g. I don't use the GUI versions of these tools but I bet this isn't supported nearly as well there.

bjeanes 2 days ago

I consider myself an AI sceptic and would wish them away if I could. But I have been trying to use them in for projects-that-are-only-for-me too, mostly so that my scepticism is informed by some practice and not just armchair musing. It hasn't changed my opinion about them very much at all, mind you, and probably even deepened my objections.

All that being said...

The truth is, I barely think about these things as “apps” (I have no intention to distribute them). They’re artifacts of me making my computer do stuff for me, the way I want it to.

If I table all my moral, sociopolitical, *and* utilitarian objections to AI for a moment, this quote does bring up some interesting thoughts. To the extent that I understand the vision behind Smalltalk, it seems like it would be an interesting context to have access to these kinds of AI tools. The idea of having a fungible, personalised system moulded around ones specific workflows is attractive, but broadly impenetrable by non-technical users. With LLMs (though ideally in an entirely different sociopolitical context than we currently have), it's interesting to me to think about how this would make that kind of "moulded" computing experience more accessible to broader range of computer users.

MarkMLI 2 days ago

A user interface should be as fancy as needed, and no fancier. The examples in the cited article are distinguished either by being "viewers" of various kinds, or by running on a platform that has effectively deprecated anything except GUIs.

Something like ps (on unix) isn't interactive so doesn't need a UI. Something like top (again on unix) is interactive, but there's no need to muddy the water by introducing a substantial amount of IPC (as does X11) or library references (Wayland etc.).

[–]  simonw 2 days ago

△ I still don't know what all of the columns in top and ps mean, and I've been using them for decades.
5 A GUI version with expandable inline explanations (or text that isn't just obscure acronyms) would be an improvement for both.

malxau 46 hours ago

This article seems a lot more about AI than TUIs. It's a better article when ignoring the core point it attempts to make.

| *[W]e can all agree there remains an important distinction between vibe-coding and vibe-shipping...
It's an AI-assisted music player that includes 90% of the interface of Music.app. Music.app. My ever-present personal computing nemesis.*

The real issue here is we have many GUI frameworks and it seems like a new one is born every minute. Learning how to be productive with each one is infeasible. But if AI can spit out a prototype that gets all the boilerplate somewhat right, a developer has a huge head start in learning where it went wrong and researching how to do better - the barrier to writing a real program is lowered. The developer still needs to obsess over details and polish to make something good, it just means not having to start with zero code, zero context, and no idea which framework to choose this week.

I hope this is an accurate observation, and it means we enter an era where ISVs can't just bundle bad music players full of advertising and upsells, because the barrier to replacing them with a cleaner version got lower. That may be naive and may not happen, but it's an interesting thought.

pushcx 2 days ago

| *it would be more useful if I cleaned up all my genre labels*

OK, this nudged me to kick a post out of my drafts folder on [fixing up the tags on my 17k mp3 collection](#), which included leaning Claude in a different way. I could spend ~100 hours making tedious decisions, have Claude do that and spend 1 hour reviewing, again spend a few hours finding a tagging tool and hoping it didn't also dump me back into a similar situation, or write my own tool.

johnklos 2 days ago

Not mentioned is how we get scripts and other programs to call and interact with non-TUI programs. It's a bit strange to suggest not doing a thing, but not suggesting how that'd actually work.

[–]  k749gtnc9l3w 2 days ago

△ The post explicitly separates TUI (with layout) and CLI, and recommends CLI/REPL + GUI wrappers.
2 Not that it is right about TUIs, but for external driving CLIs and REPLs are indeed simpler than TUIs.

