Why TanStack Start is Ditching Adapters

by Tanner Linsley on Nov 22, 2024.

To “adapter” or not?

Building a new front-end Javascript framework is a daunting task, as I’ve been learning with building TanStack Start, my new TanStack-powered full stack framework. There’s so many moving pieces:

- Routing
- Server Side Rendering
- RPCs & APIs
- Development Tools
- **Deployment & Hosting**

That last one, **deployment & hosting** can be especially tricky, since these days it seems that every single cloud environment has it’s own quirky incantations to get things to work “just right”. When faced with this decision for TanStack Start, I obviously knew which hosts I wanted to start supporting out of the gate and Vercel was at the top of that list.

My first gut-reaction was to start building a system that could have “adapters” for each host, then just focus on writing the Vercel adapter first.

This plan, however, was flawed from the start. It didn't take long to realize that I personally was going to be responsible (at least in the beginning) for writing most if not all of the code to make TanStack Start come to life on not only Vercel, but many other targets and platforms. After some quick research, this task alone was daunting enough to make me question my motivations in building a server-bound JS framework at all.

Technically, the work required to deploy to Vercel alone is already very simple by just adhering to some output file/directory naming conventions. However the paralysis came from just the sheer number of environments/hosts there are to support. There's a lot! Just look at Remix's growing list of server adapters! Remix isn't the only framework with this list either. Most server-bound JS frameworks have something similar. I was essentially committing to writing at least 10 adapters in the first few months of the framework and I had barely gotten into the exciting features of the framework itself (not to mention the work in maintaining and updating these adapters).

The harsh reality here is that there **isn't a way around this.** If your framework is providing **any server-targeted code in your framework, you need to ensure it will run perfectly anywhere you can deploy it.**

So, as I was about to succumb to the infinite sadness of writing a hundred adapters and dealing with upstream breaking changes for the rest of my life, I spoke with my friend Ryan Carniato about how Solid JS is approaching this problem with our cousin framework Solid Start and he confidently said "Just use Vinxi".

Vinxi = Nitro + H3 + Vite

Vinxi is a "JavaScript toolkit to build full stack apps and frameworks with your own opinions", powered by Nitro and Vite. So what makes it so special?

There are a ton of awesome features in Vinxi that make it extremely useful for building a framework, but one of the coolest pieces is that it's powered by Nitro, H3 and Vite. Nitro's tagline is literally "create web servers with everything you need and **deploy them wherever you prefer**" (emphasis is mine).

In simple terms, Nitro effectively makes TanStack Start "*adapter-less".* It uses H3, an HTTP framework that maintains its own lower-level adapters on your behalf so you can write your server code once and use it anywhere (sounds a lot like React!).

By using Vinxi (which uses Nitro, H3 and Vite under the hood), all of TanStack Start's adapter problems were gone. I never even had to think about them!

In fact, to deploy to Vercel, it was even easier than I had initially planned: just pass a `vercel` target to our `defineConfig`'s `server.preset` option (which uses Vinxi):

jsx

```
import { defineConfig } from '@tanstack/start/config'

export default defineConfig({
  server: {
    preset: 'vercel',
  },
})
```

What does it support?

Vinxi, Nitro, H3 and Vite are **impressive to say the least.** We were pleased to see that on our first try, a slew of Vercel features worked perfectly out of the box including the GitHub integration, server functions, edge functions/middleware, immutable deploys, environment variables, server-side rendering, and even streaming.

That’s a massive list that we essentially got for free by using Vinxi and friends.

TanStack Start is coming!

With builds and deployments solved and built-in support for integrating my GitHub repos right into to my personal favorite hosting providers, I could finally focus on what I think makes TanStack Start special:

- A best-in-class fully type-safe Router
- Flexible primitives for building server-bound RPCs
- Opt-in Server Functionality (SSR, APIs, RSCs, etc)
- And deep integration with other TanStack primitives like TanStack Query
- And even more to come!

Going the Extra Mile

It’s awesome that we were able to offload so much to Vinxi and gain so many awesome features, but it’s definitely not a 100% complete solution to using *every* feature of a hosting platform, especially Vercel where we have access to more than just deployments. We’ve also been thinking more about features like edge network caching and my personal favorite, *skew protection.*

For instance, skew protection, which ensures that client and server stay in sync for their respective deployments requires more than just a build step. It involves the ability to deeply integrate platform primitives into the framework at runtime as well, or in the specific case, being able to inject specific cookies or headers into outgoing API/server requests directed at Vercel.

I’m happy to report that TanStack Start is going to ship with amazingly powerful middleware primitives (for both API routes and Server Function RPCs) that will make this a one-liner, or possibly even automatic (hopefully).

This level of DX and integration is what makes me excited for the future and I believe is what open source is truly about: composing powerful tools from the ecosystem together to deliver amazing experiences for both developer and user.

I couldn’t think of a better mashup than TanStack Start, Vinxi (and friends) and Vercel to give you a best-in-class web app experience.

Try it in 60 seconds

TanStack Start is currently in Beta! Click the "Deploy" button below to both create and deploy a fresh copy of the TanStack Start “Basic” template to Vercel in ~1 minute.

▲ Deploy

We hope you enjoy what we’re building and can’t wait to get your feedback!

Edit on GitHub