Should we decouple?

DrupalCamp NJ, March 14, 2023

Who am I?

Tom Mount

Sr. Solutions Engineer, Edgio

Software engineer for 20 years, 15 of those in web development

Lived and worked in the greater Philadelphia area for 13+ years

What are we going to cover?

Few things that this presentation will NOT do:

- I won’t tell you if you should or should not make a specific decision.
- I won’t tell you one path is better than another.
- I won’t turn this into a pros vs. cons list.

This presentation SHOULD (hopefully):

- Raise some questions.
- Help you make a more informed decision.
- Shine a light on some overlooked concerns so you can take action now.

Definitions

Definitions

Headless
hĕd'lĭs

Adjective

1. An imprecise industry buzzword that the presenter will try his hardest not to say but some might sneak in there anyway because despite efforts to the contrary that's what his coworkers in Sales talk about all day long.
2. Referring to sites that are **decoupled** in some format.

See also: ***decoupled***

Definitions

Coupled/Decoupled
(dē-)ˈkə-pəld

Adjectives

Referring to the how much the **backend** of a website is involved in routing and sending the initial HTML response to the **frontend**.

a. *Tightly Coupled*: the backend handles all the routing for the site and sends over all the HTML content necessary to display a page on the frontend.
b. *Partially Decoupled*: the backend handles the routing and sends over the initial HTML framework, but without some/all of the content to be displayed; the frontend dynamically loads content via API calls after the initial response is displayed.
c. *Fully Decoupled*: the backend does not handle request routing nor does it send over initial HTML content as a response. The frontend is considered a completely separate system to the backend. Content is retrieved from one or more **backends** via API calls.

*See also: **frontend/backend***

Definitions

Frontend/Backend

ˈfrənt-ˌend, ˈbak-ˌend

Nouns

1. A source of neverending frustration for many web developer job-seekers when asked “What kind of developer are you?”
2. A shorthand for separating the interactive presentation layer (“frontend”) from everything else in a web stack (“backend”).

Contextualizing the last few definitions

- Out of the box, Drupal is **tightly coupled**. The menus, block, views, etc. are all compiled when requested and served as a single chunk by Drupal (the **backend**).
- D8 and D9 experimented with some of the admin pages being **partially decoupled**. For those pages, the Drupal admin theme would have served the initial compiled HTML response, but the response would have included references to JavaScript libraries, and the actual admin content (eg. settings) would have been injected as reactive web components.
- It is possible to run Drupal as a **fully decoupled** architecture, where a hosted **frontend** built in React, Vue, etc. receive and respond to web requests, and Drupal is used as a CMS and an API service.

Definitions

Framework

ˈfrām-ˌwərk

Noun

A collection of JavaScript classes/methods that provide a standardized way of writing web components. Examples include: React, Svelte, Vue. These generally do not contain robust routing and middleware solutions, if at all.

Definitions

Meta Framework

ˈme-tə ˈfrām-ˌwərk

Noun

A collection of JavaScript classes/methods that further abstract a basic framework to provide a standardized way of handling routing, middleware, etc.

Examples: Next.js is to React what Sveltekit is to Svelte, and what Nuxt is to Vue.

Definitions

Monolith
ˈmä-nə-ˌlith

Noun
Adjective form: ***monolithic***

A web architecture not in use much these days where the server, backend language processor (eg. PHP, Java, etc.), database, and CMS application were all housed on a single server, either physical or cloud-based (eg. Rackspace, Digital Ocean). Because this model represents a substantial single point of failure, most modern web apps are broken apart and hosted in containers or on separate servers (eg. AWS primitives like S3, RDS, EC2, etc.).

Let’s consider decoupling.

Business Concerns

The Whos, Whats, Whys, Whens, and Hows

"Before we start, we ask important questions, like what are you trying to accomplish? What pain points are you trying to solve? How will we deliver a better customer experience? How fast does the project need to be done?"

--Molly Duggan, interview with Pantheon

Business Concerns

What pain points are we trying to solve? Will some sort of decoupled architecture solve them?

Some typical pain points for the business include:

- Speed
- Agility (ability to turn around a new product, content update, etc. in a timely fashion)
- Low/lost revenue

Recommendation: use “The Five Whys.”

Business Concerns

What business goals are we aligning to? Is a decoupled architecture the best way to align to those goals?

- Decoupled is usually a good fit for e-commerce but not always for marketing sites.
- How many sites are being transitioned?

Business Concerns

Who owns the process for creating new content/updating existing content?

- Marketing and development workflows will probably change.
- Make sure everyone affected by those changes has a voice in the decision.

Business Concerns

What is the timeline for this transition?

- It will always take longer than you think.
- **Very few people want to completely relaunch a site all at once anymore.**

Business Concerns

How will we know if this transition was a success?

- Establish KPIs.
- Remember: be SMART (Specific, Measurable, Achievable, Relevant, Time-Boxed).
- Document data collection methods and take baseline readings.

Team Concerns

All about the Whos

“Now we have to go out and hire a new set of frontend devs because we decided to decouple; and it’s easier to hire React devs than it is to find Vue devs, so we’re probably going to have to use React for the new site.”

-- *Edgio prospect, paraphrased*

Team Concerns

Do we have developers with experience writing decoupled applications?

- Decoupled JavaScript applications are generally more complex than managing a tightly-coupled Drupal application.
- JS applications may require more understanding of the request lifecycle.

Recommendation: start out small with a partially-decoupled architecture on some less-trafficked pages in a pre-production environment. Keep a short feedback loop with the dev team to raise any issues quickly.

Team Concerns

Do your developers have experience writing in the framework we're evaluating?

- Being able to write JS doesn't guarantee the ability to write an Angular app from scratch.
- Consider staff augmentation.

Team Concerns

How big is our Frontend development team?

- FE dev in a tightly coupled stack: focus on presentation, not sitebuilding.
- FE dev in a partially decoupled stack: all of the above, plus build pipeline.
- FE dev in a fully decoupled stack: all of the above, plus routing.

The more you decouple, the more responsibility falls on the shoulders of the FE dev team.

Recommendation: keep your dev team, especially FE devs, in the decision-making process as much as possible.

Technology Ecosystem

The nebulous “other”

> “Our modernisation isn’t just a refactoring of the tech stack. If we were to continue to do things in the same way as before there would be a reasonable chance we’d end up back where we are today.”
>
> *-- Dave Charles, Kingfisher*

Technology Ecosystem

What about our culture may need to change as a result of choosing a new decoupling strategy?

- Dev and deploy processes may need to change.
- Not changing dev and/or deploy processes will limit the effectiveness of the new architecture.

Recommendation: map your existing processes and look for process improvements as well as architecture improvements.

Technology Ecosystem

What parts of our data model may (or must) change as a result of choosing a new decoupling strategy?

- Will the JSON API serve everything you need?
- What are the interdependencies between content types?

Technology Ecosystem

Do we need to re-evaluate hosting partners and costs?

- Not all hosting solutions for your Drupal site support all modes of decoupled architecture.
- You may need to change or supplement hosting partners.
    - Different or increased cost
    - Different support structures
    - Potential DNS changes

Recommendation: any hosting provider that runs Drupal today will also completely support partially-decoupled architectures, even if they don’t support fully-decoupled architectures.

Technology Ecosystem

Will our security footprint change if we choose a new decoupling strategy?

- New API routes = new potential security holes.
  - Scraping attacks
  - Phishing/proxy attacks
  - Brand damage
- WAF
- Bot Management

Technology Ecosystem

Do we have the testing and debugging bandwidth to support this change?

- “Adding layers to your e-commerce environment can increase the time and skillsets required to identify the root cause of an issue and troubleshoot it.” -- Adobe whitepaper
- Complexity **will** increase over time (see also: 2nd Law of Thermodynamics).

Recommendation: begin working tests into your development process early, and choose a standard library and methodology for those tests.

Wrapping It Up

Wrapping It Up

So…should we decouple?

…you should definitely consider it!

Wrapping It Up

Set yourself up for success if you choose to decouple.

- Know where you are now, identify where you want to be at the conclusion of your project, and document how you will know you’re on track as you’re working.
- Give your contributors a seat at the decision-making table and get their feedback early and often.
- Identify any potential hidden costs around hosting, support, and staffing before you start.
- Choose a technology partner that has experience building decoupled applications, that can help guide you and your team while you work.

Wrapping It Up

Sources:

- Delivering Purpose-Driven Decoupled Experiences with Molly Duggan. Michaela Morgan. https://pantheon.io/blog/delivering-purpose-driven-decoupled-experiences-molly-duggan
- Headless commerce primer for growing merchants. Whitepaper from adobe.com.
- Decomposition Journey at CarGurus - monolith to microservices. Frank Fodera. https://www.cargurus.dev/Decomposition-Journey-at-CarGurus-monolith-to-microservices/
- Introducing Application Modernisation at Kingfisher. Dave Charles. https://medium.com/kingisher-technology/introducing-application-modernisation-at-kingfisher-5994a627d4d3

Wrapping It Up

Special Thanks:

- Juan Pineda
- Chris Ferrell
- Will Rowe

Questions

Thank you!

Keep in touch!

Tom Mount
tmountjr@gmail.com
linkedin.com/in/tom-mount