Headless Session 2

Date

05 Dec 2018

Attendees

- Christopher Kaiser

Headless Setups / Approaches

Some are using headless. Some are curious about it.

Three Different approaches to Headless

- React with NodeJS and Magnolia Custom REST endpoints
- NGinx with prefilles Content / static pages enriched from Magnlia
- Vanilla Magnolia with lightmodule and angular

Interest in Paperboy

Some are doing everything headless - they have a very strong React team.

Chris from nexum

nginex. node.

Editors are using the pages app.

FTL only has a structural view - allows them to find the components.

This then gets transformed to json. Gets used by frontend app.

Using atomic design approach.

Editors think in "organisms"

Biggest problem: How to make the dialogs more approachable. WIthout drilling down into nested components. Painful in the page editor.

Second problem: Duplicated content.

Third problem: Scalability on the endpoint. Operations is only used to running standard magnolia (not with node and everything.)

What is the advanatge / Selling point for headless:

They are either independent of CMS, independent of the frontend.

Swap out with new frontend in 2 years when it arrives.

Benefit for the project owner / not author.

Easier to do mobile first - use default frontend developers.

DECOUPLED - flexibilty.

Erne Diekman.

Another advantage - forcing you to think about your content from the start from the actual content structure - rather than the design.

But also a down side. Might be a bad presentation.

Workaround - in preview mode - it opens an iframe and includes the built frontend.

Another one

In page editor with mega dialogs.

Angular just imports rendered HTML - and then in Ajax injects images and so forth.

Frontend team. Use an external team to do Java developerment when they need it.

Another advantage

Hard to find specialized Magnolia developers.

Marvin

What is the advantage for the author?

Chris

Professional authors: do not care a bout WYSIWIG - they just want to create content as fast as possible. Visual page at the end.

For SEO reasons - doing SSR.

Performance reasons. Build out a static site - a rendered app.

10000 requests/second. How. Magnolia? Maybe works if its just the delivery endpoint and caching.

What about personalization and REST endpoints?

We just do it in the frontend, not in the Admincentral.

Performance is the main reason for Headless.

Challenge - how to let authors SEE what they are going to create - how to create an editorial experience.

Pages App vs Stories App

Nesting of page - difference

Missing in the competition - hierarchies! Page hierarchies as well. Not just in page - but out of page.

Faster to develop in the pages app then in the pages app. Building custom components. Hard in Stories app.

A lot of customers are really into WYSIWIG - want to see it. See it in the final layout.

DIFF tool is also really big benefit in the Pages app.

Do we focus on happy developers or happy editors. Cost for developer time. Fast development time.

Conclusion

It depends on the usage-scenario. Are you going for development time or delivery speed or author / editor happiness 😁 = usability for editor. Find the trade of of different approaches.

And of course: It depends on money!

<summary>