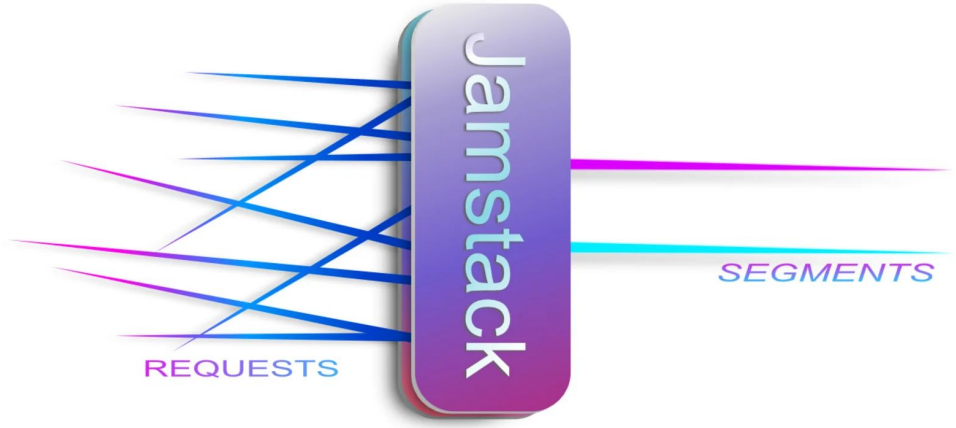


Treat your users right with Segmented Rendering

and a demo with Next.js

Conf42 JavaScript - 2022

Your speaker - Eric Burel

Founder of LBKE - Montpellier, France

Web developer
Consultant in public funding for rtob
Maintainer of Vulcan.js
Member of Devographics (State of JS/CSS/GraphQL)
Next.js teacher at Human Coders

@ericbureltech
medium.com/@eric.burel

1. What is web personalization?

It’s dangerous to go alone
Pick your theme!

It’s dangerous to go alone
Pick your theme!

fire

water

grass

Implementation via cookies

Implementation via cookies

cookies[“starter”] = “fire” || “water” || “grass”

Implementation via cookies

cookies[“starter”] = “fire” || “water” || “grass”

Various use cases

Various use cases

cookies[“starter”] = *“fire” || “water” || “grass”*

Various use cases

cookies[“starter”] = *“fire” || “water” || “grass”*

cookies[“i18n”] = *“fr” || “en” || “de”*

Various use cases

cookies[“starter”] = *“fire” || “water” || “grass”*

cookies[“i18n”] = *“fr” || “en” || “de”*

cookies[“is_paid”] = *“true” || “false”*

Various use cases

cookies[“starter”] = *“fire” || “water” || “grass”*

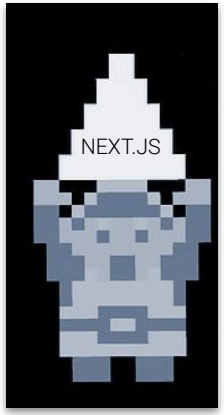
cookies[“i18n”] = *“fr” || “en” || “de”*

cookies[“is_paid”] = *“true” || “false”*

cookies[“test_bucket”] = *“A” || “B”*

2. Render the right theme

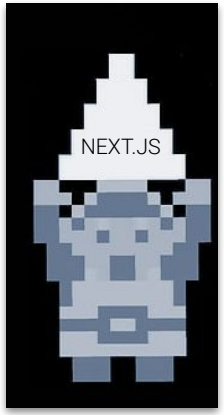
The rendering triforce (with Next.js)



The rendering triforce (with Next.js)

Client-Side Rendering (CSR)

💻 JS ~ SWR (ou fetch) + React



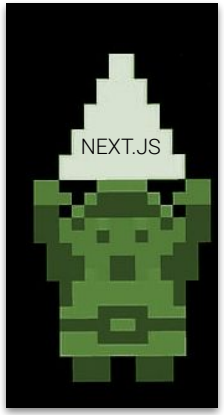
The rendering triforce (with Next.js)

Client-Side Rendering (CSR)

💻 JS ~ SWR (ou fetch) + React

Server-Side Rendering (SSR)

🗄️ PHP ~ getServerSideProps



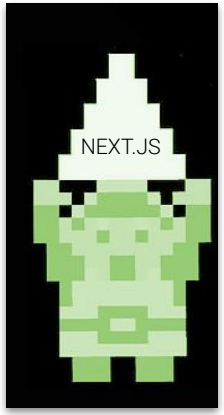
The rendering triforce (with Next.js)

Client-Side Rendering (CSR)

💻 JS ~ SWR (ou fetch) + React

Server-Side Rendering (SSR)

🗄️ PHP ~ getServerSideProps



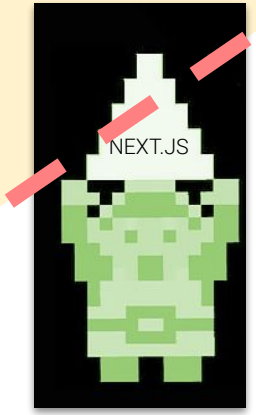
Static Site Generation (SSG)

🗄️ HTML ~ getStaticProps

Dynamic but slower

Client-Side Rendering (CSR)

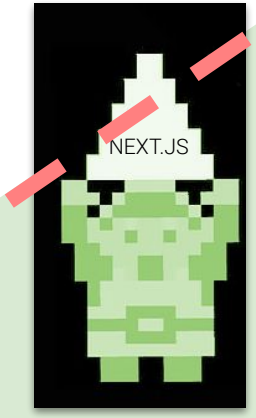
Server-Side Rendering (SSR)



Static Site Generation (SSG)

Client-Side Rendering (CSR)

Server-Side Rendering (SSR)



Static Site Generation (SSG)

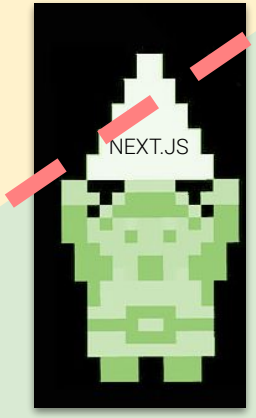
Faster but static

What to use for personalization?

Dynamic but slower

Client-Side Rendering (CSR)

Server-Side Rendering (SSR)



Static Site Generation (SSG)

Faster but static

Rich guest/poor customer

Rich guest/poor customer

=

Faster static renders for public content

Slower dynamic renders for personalized content

3. Personalized staticness via “Segmented Rendering”

First try: using the URL directly

foobar.com/fire/home

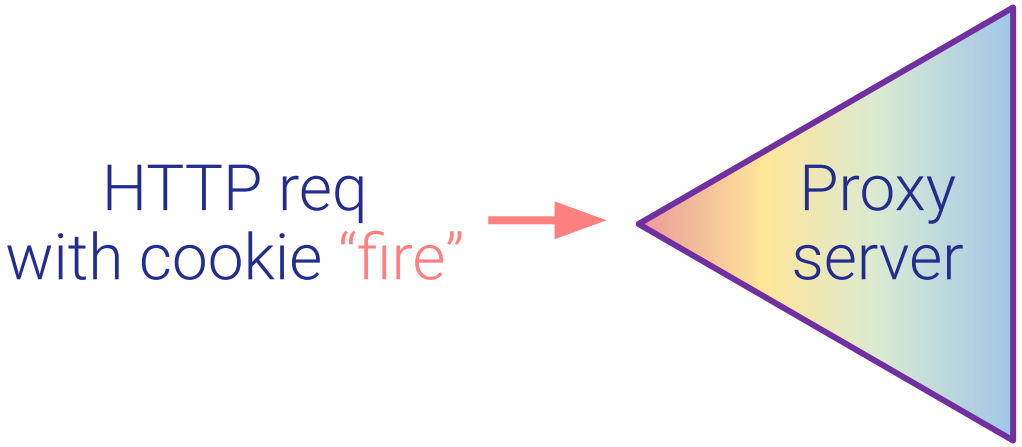
water

grass

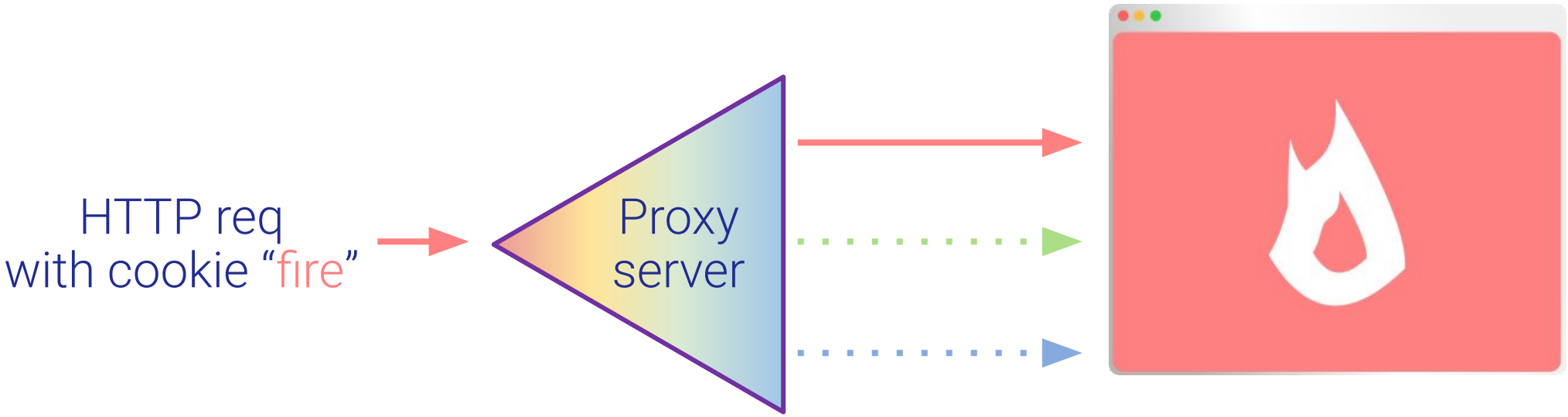
Many issues

- user can change URL manually: /paid/my-article
- they see the param: /bucket-a/my-article
- URL becomes long: /fr/dark/my-company/a/my-article

Second try: adding a rewrite server



Second try: adding a rewrite server



foobar.com/fire/home

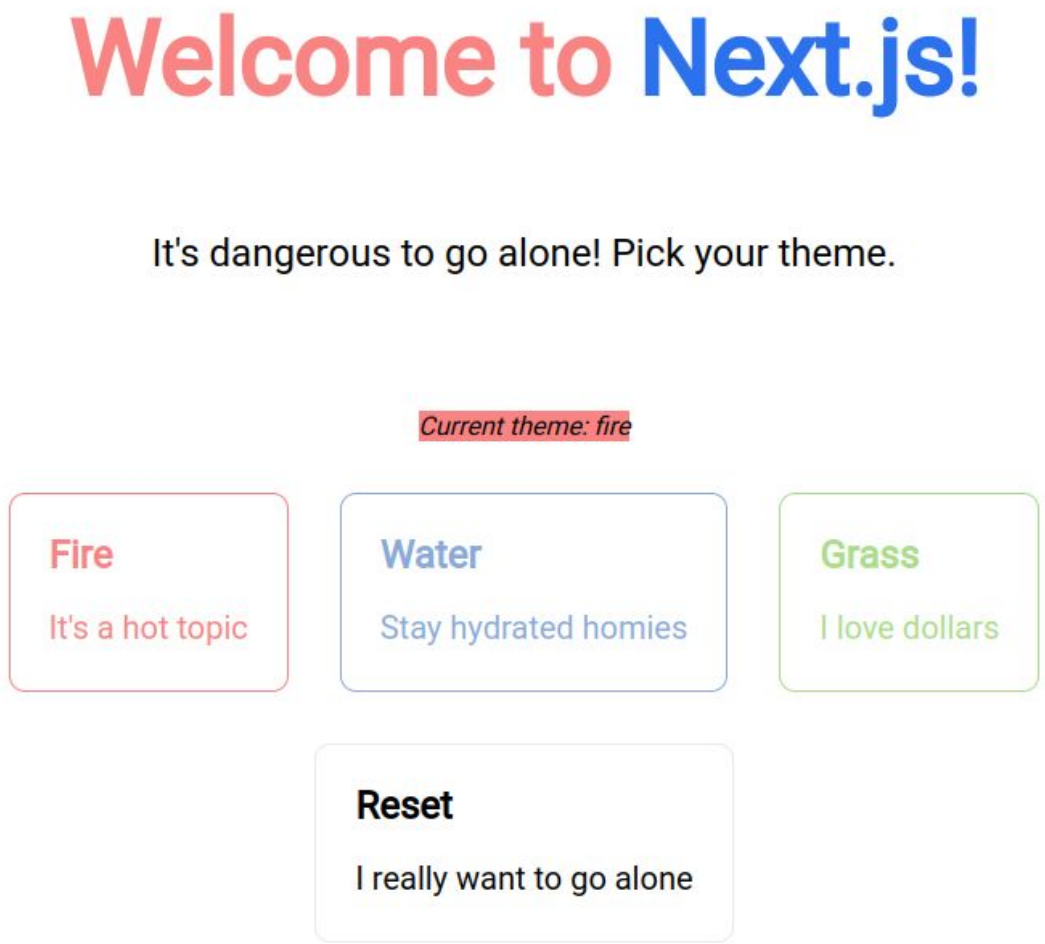
This is Segmented Rendering!

This is Segmented Rendering!

=

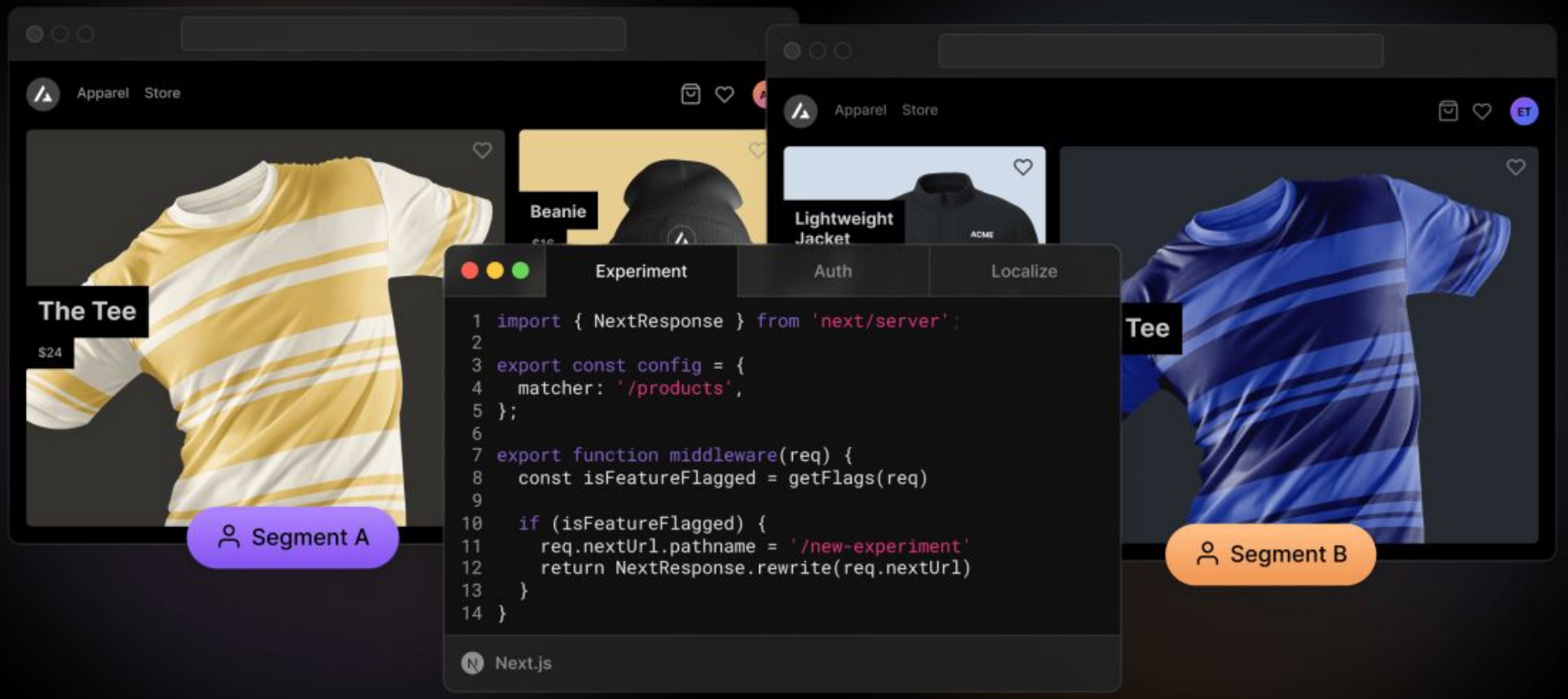
1 URL per segment + a rewrite server

4. Implementation with Next



Live demo: https://segmented-rendering-demo.vercel.app/
Code: https://github.com/lbke/segmented-rendering-demo

Deliver at the edge



5. Conclusion: the future of SSR?

It’s all about caching

Static = SSR cached at build-time

Per-request SSR = cache miss

It’s all about caching

Static = SSR cached at build-time

Per-request SSR = cache miss

Cache key = full request, not just the URL

Cache value = rendered HTML and/or data

“In the future, we will have an unified API for server-rendering that encompasses static, per-request and everything in between”

Eric Burel taking a relatively safe bet at Conf 42, given that Next.js 13 brings exactly this, circa 2022

https://tinyurl.com/ssr-theory

Thank you!

A new pattern for the Jamstack: Segmented Rendering published at Smashing Magazine:
https://www.smashingmagazine.com/2022/07/new-pattern-jamstack-segmented-rendering/
Megaparam pattern with Next.js (implement Segmented Rendering for many segments):
https://blog.vulcanjs.org/render-anything-statically-with-next-js-and-the-megaparam-4039e66ffde
Next.js personalization by Plasmic:
https://www.plasmic.app/blog/nextjs-personalization
Static A/B tests by Plasmic:
https://www.plasmic.app/blog/nextjs-ab-testing
At the edge rendering by Eleventy
https://www.11ty.dev/blog/eleventy-edge/
The unicorn architecture (early version of Segmented Rendering)
https://blog.vulcanjs.org/lets-bring-the-jamstack-to-saas-introducing-rainbow-rendering-ad1834fe62ff
Segmented Rendering with HTTP Cache
https://blog.vulcanjs.org/treat-your-users-right-with-http-cache-and-segmented-rendering-7a4f4761b549
An implementation of the Megaparam pattern :
https://github1s.com/VulcanJS/vulcan-next/blob/devel/src/pages/vn/examples/[M]/megaparam-demo.tsx
SSR theory draft research paper (looking for contributions!)
https://tinyurl.com/ssr-theory

Images https://vercel.com/

contact@lbke.fr

(+33) 06 89 97 95 70

www.lbke.fr

@LBKE_FR

Lebrun Burel SARL au capital de 3000 euros
SIRET : 83487043800027 - N°TVA : FR 31 834870438
Siège Social : 2 avenue d'Unterschleissheim 34920 Le Crès

LB
KE