HUGE

Hello
React MDE

SSR - Next.JS

August 13, 2019

Agenda.

1. CSR vs. SSR
2. Motivations
3. Solution: Next.js
4. First steps
5. Rendering
6. Server endpoints

Who am I?

William Gómez

Engineer at Huge

(Full stack wannabe web engineer at Huge)

Python Medellin meetups co-organizer

Freelancer

Machine learning lover

Data science FEM - Mentor

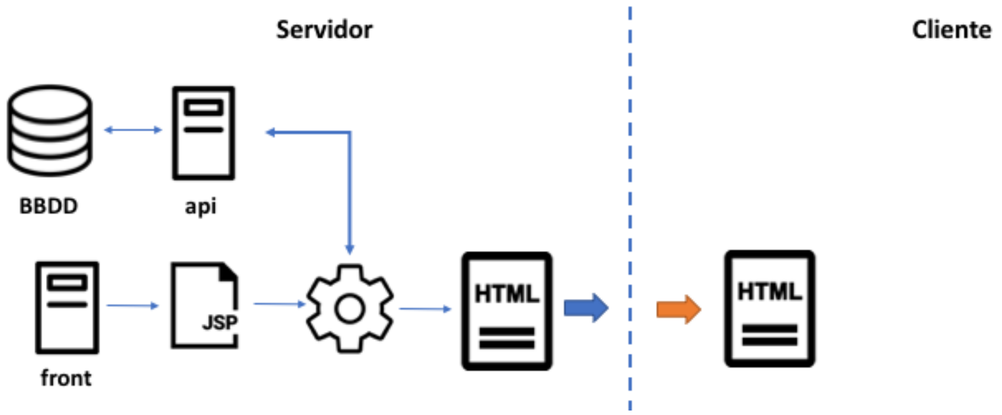
Dos razones principales por las cuales deberías considerar usar SSR:

1. Aumento en la velocidad de carga inicial del sitio.
2. SEO Friendly.

Aumento en la velocidad de carga inicial

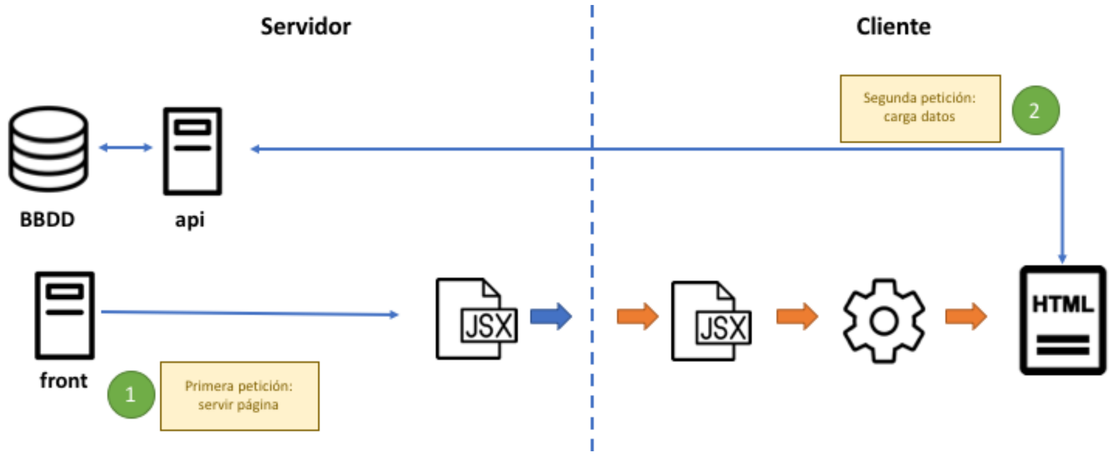
SSR

Generación desde recursos locales.



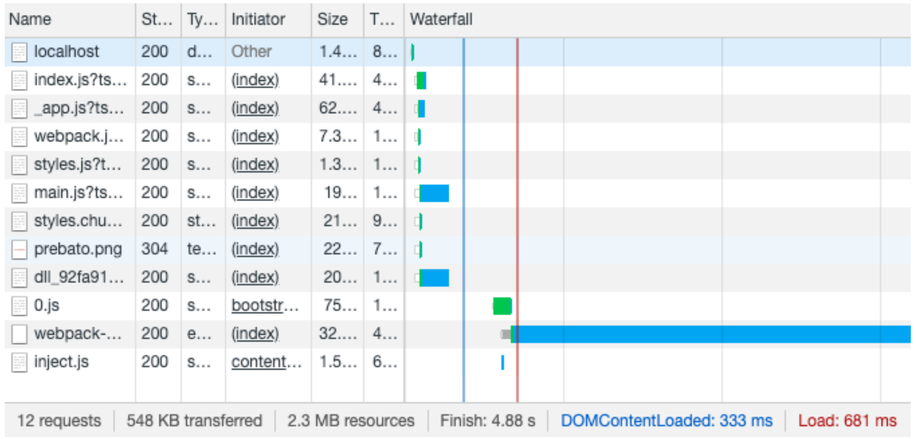
CSR:

Primero hay que cargar el HTML + JS, y luego pedir lo que falte.



Aumento en la velocidad de carga inicial

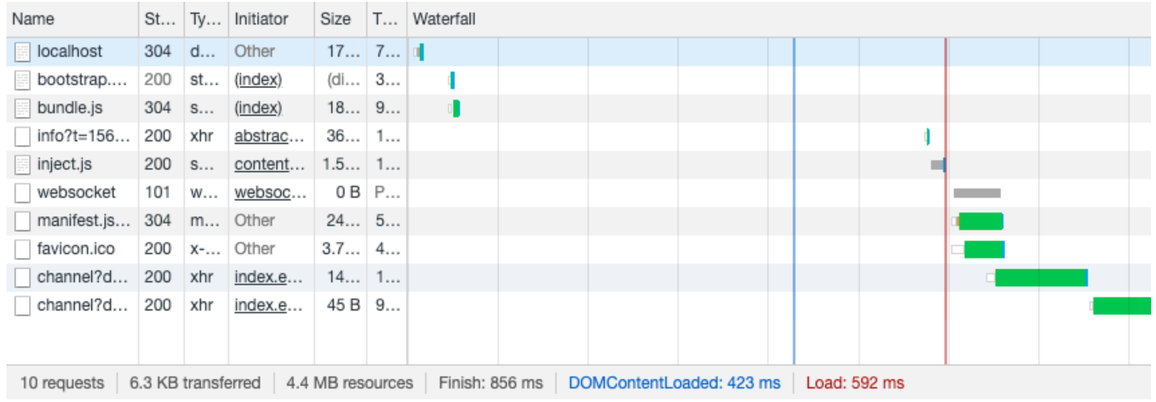
SSR



Name	St…	Ty…	Initiator	Size	T…	Waterfall
localhost	200	d…	Other	1.4…	8…	
index.js?ts…	200	s…	(index)	41.…	4…	
_app.js?ts…	200	s…	(index)	62.…	4…	
webpack.j…	200	s…	(index)	7.3…	1…	
styles.js?t…	200	s…	(index)	1.3…	1…	
main.js?ts…	200	s…	(index)	19…	1…	
styles.chu…	200	st…	(index)	21…	9…	
prebato.png	304	te…	(index)	22…	7…	
dll_92fa91…	200	s…	(index)	20…	1…	
0.js	200	s…	bootstr…	75…	1…	
webpack-…	200	e…	(index)	32.…	4…	
inject.js	200	s…	content…	1.5…	6…	

12 requests | 548 KB transferred | 2.3 MB resources | Finish: 4.88 s | DOMContentLoaded: 333 ms | Load: 681 ms

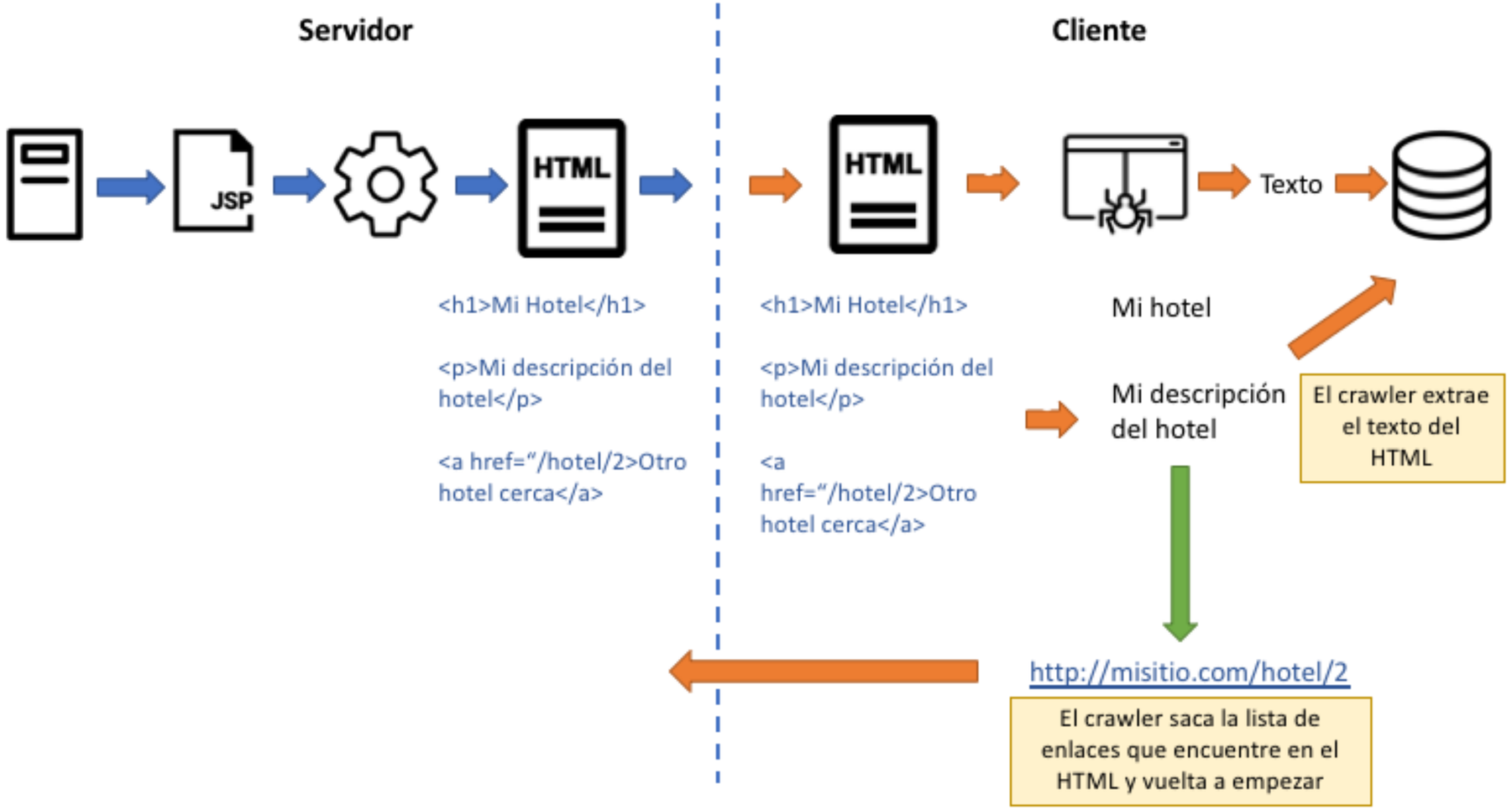
CSR:



Name	St…	Ty…	Initiator	Size	T…	Waterfall
localhost	304	d…	Other	17…	7…	
bootstrap.…	200	st…	(index)	(di…	3…	
bundle.js	304	s…	(index)	18…	9…	
info?t=156…	200	xhr	abstrac…	36…	1…	
inject.js	200	s…	content…	1.5…	1…	
websocket	101	w…	websoc…	0 B	P…	
manifest.js…	304	m…	Other	24…	5…	
favicon.ico	200	x-…	Other	3.7…	4…	
channel?d…	200	xhr	index.e…	14…	1…	
channel?d…	200	xhr	index.e…	45 B	9…	

10 requests | 6.3 KB transferred | 4.4 MB resources | Finish: 856 ms | DOMContentLoaded: 423 ms | Load: 592 ms

SEO Friendly



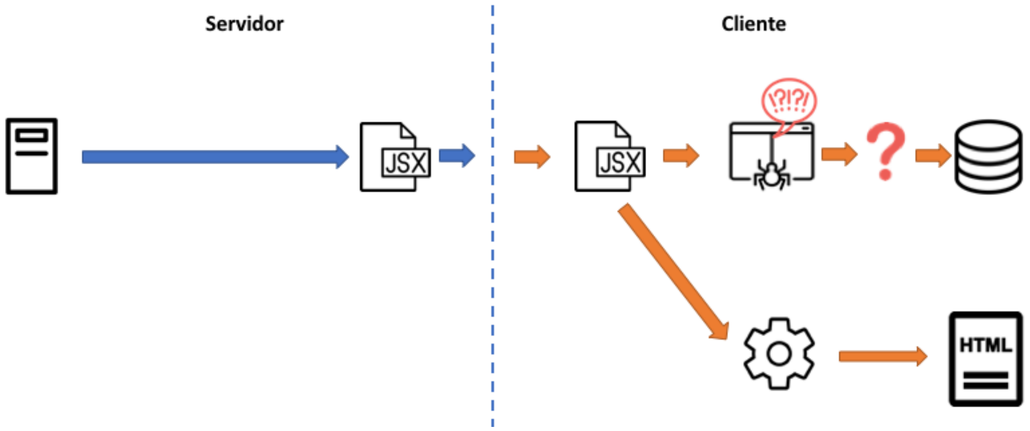
SEO Friendly.

SSR

El html ya llega con el contenido.

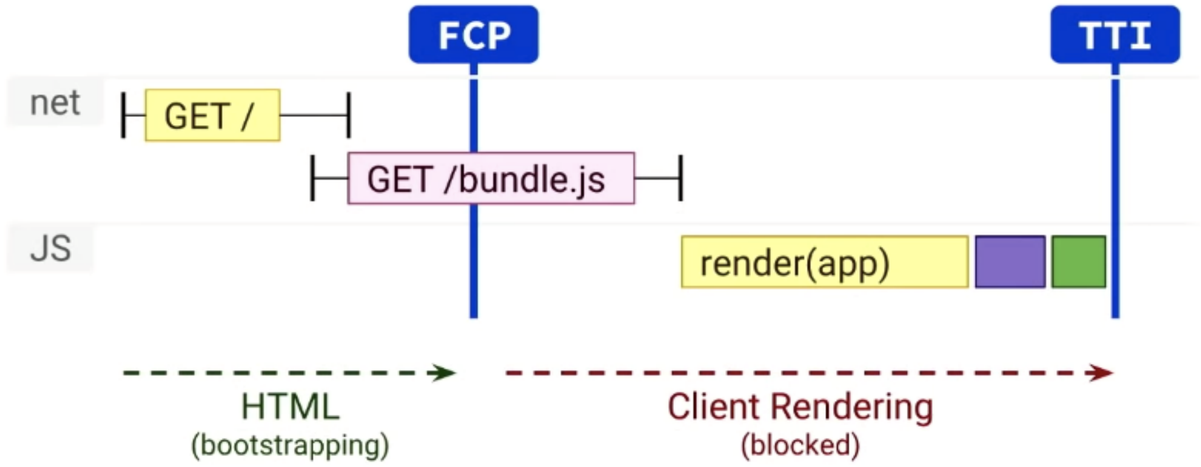
CSR

Es difícil para el crawler indexar porque debe esperar a que se cargue todo el contenido.



Crea una app SSR, y una CSR, dale en View Page Source, ¿qué ves?

How do you make your app scale with size?



Taken from: https://www.youtube.com/watch?v=WKJVXN2kMS8

Create React App simplifies a lot the things...

But there is still a lot of things to solve:

Client-side routing, page layout, api's... and so on.

Solution: Next.js

Features:

- Server-rendered by default
- Automatic code splitting for faster page loads
- Simple client-side routing (page based)
- Webpack-based dev environment which supports Hot Module Replacement(HMR)
- Able to implement with Express or any other Node.js HTTP server
- Customizable with your own Babel and Webpack configurations

First steps

1. Create a folder
2. Run inside the folder: `npm init`
3. Install packages: `npm install --save next react react-dom`
4. Add the following scripts to package.json:

```
{
  "scripts": {
    "dev": "next",
    "build": "next build",
    "start": "next start"
  }
}
```

4. Create a folder called pages

Every .js file inside pages becomes a route that gets automatically processed and rendered.

With these simples steps we have:

- Server rendering and indexing of ./pages/
- Every React component inside this folder is a page.
- Every import you declare gets bundled and served with each page.

```
function Home() {
  return <div>Welcome to Next.js!</div>;
}

export default Home;
```

First steps: Other basics

5. Create a folder called static

- Static file serving. ./static/ is mapped to /static/

```
function MyImage() {
  return <img src="/static/my-image.png" alt="my image" />;
}

export default MyImage;
```

6. Routing:

Client-side transitions between routes can be enabled via a <Link> component.

```
import Link from 'next/link';

function Home() {
  return (
    <>
      <ul>
        <li>Home</li>
        <li>
          <Link href="/about">
            <a>About Us</a>
          </Link>
        </li>
      </ul>

      <h1>This is our homepage.</h1>
    </>
  );
}

export default Home;
```

6. Dynamic routing:

In Next.js you can add brackets to a page ([param]) to create a dynamic route

pages/post/[pid].js

Any route like /post/1, /post/abc, etc will be matched by pages /post/[pid].js.

```
import { useRouter } from 'next/router';

const Post = () => {
  const router = useRouter();
  const { pid } = router.query;

  return <p>Post: {pid}</p>;
};

export default Post;
```

6. Dynamic routing:

- href: the path
  inside pages directory.
- as: the path that will be rendered
  in the browser URL bar.

```
<Link href="/post/[pid]" as="/post/abc">
  <a>First Post</a>
</Link>
```

```
const pids = ['id1', 'id2', 'id3'];
{
  pids.map(pid => (
    <Link href="/post/[pid]" as={`/post/${pid}`}>
      <a>Post {pid}</a>
    </Link>
  ));
}
```

7. Prerendering:

Next.js automatically determines that a page is static (can be prerendered) if it has no blocking data requirements. This determination is made by the absence of `getInitialProps` in the page.

If `getInitialProps` is absent, Next.js will statically optimize your page automatically by prerendering it to static HTML.

Prerendered:

If getInitialProps is absent.

```
function Home() {
  return <div>Welcome to Next.js!</div>;
}

export default Home;
```

Statically generated pages are still reactive: Next.js will hydrate your application client-side to give it full interactivity.

Render the page on-demand:

If getInitialProps is present.

```
import fetch from 'isomorphic-unfetch';

function Page({ stars }) {
  return <div>Next stars: {stars}</div>;
}

Page.getInitialProps = async ({ req }) => {
  const res = await fetch('https://api.github.com/repos/zeit/next.js
  const json = await res.json();
  return { stars: json.stargazers_count };
};

export default Page;
```


williamegomezo next-talk feat/initial-props

Fetching data and component lifecycle

7. Using React.Component:

It can asynchronously fetch anything that resolves to a JavaScript plain Object, which populates props.

For the initial page load, getInitialProps will execute on the server only.

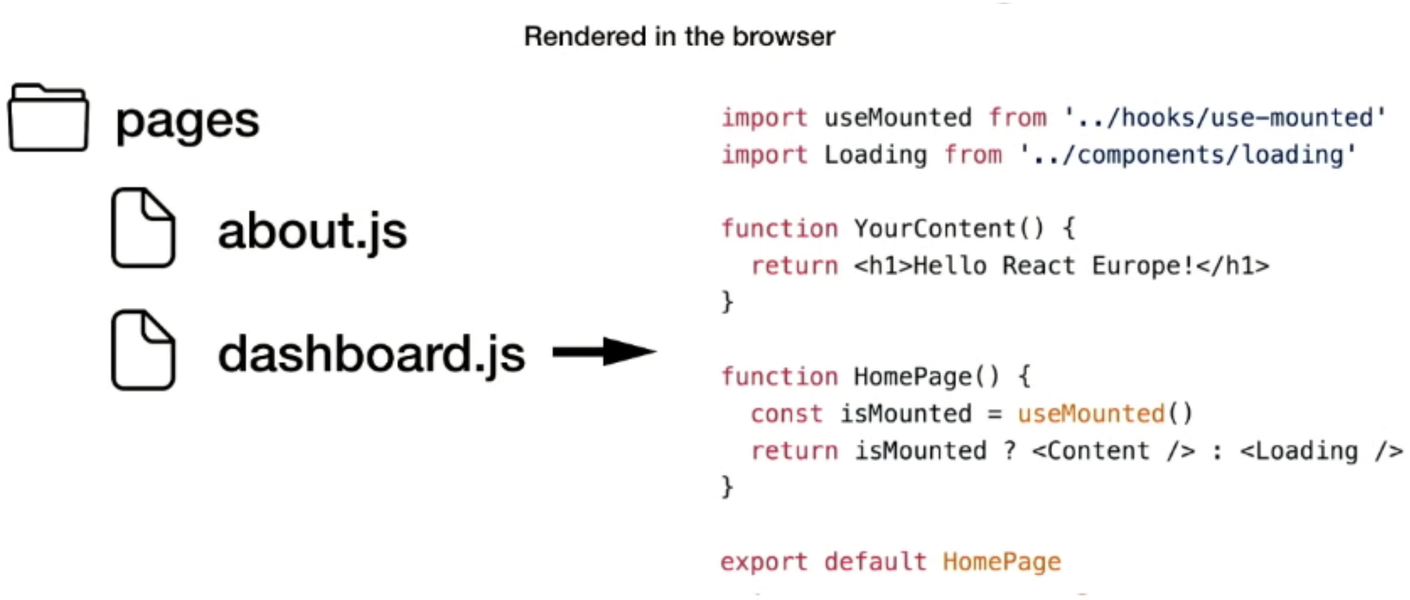
```
import React from 'react';

class HelloUA extends React.Component {
  static async getInitialProps({ req }) {
    const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;
    return { userAgent };
  }

  render() {
    return <div>Hello World {this.props.userAgent}</div>;
  }
}

export default HelloUA;
```

8. Context object with the following properties:

- pathname - path section of URL
- query - query string section of URL parsed as an object
- asPath - String of the actual path (including the query) shows in the browser
- req - HTTP request object (server only)
- res - HTTP response object (server only)
- err - Error object if any error is encountered during the rendering

Client-side rendering: Coming!

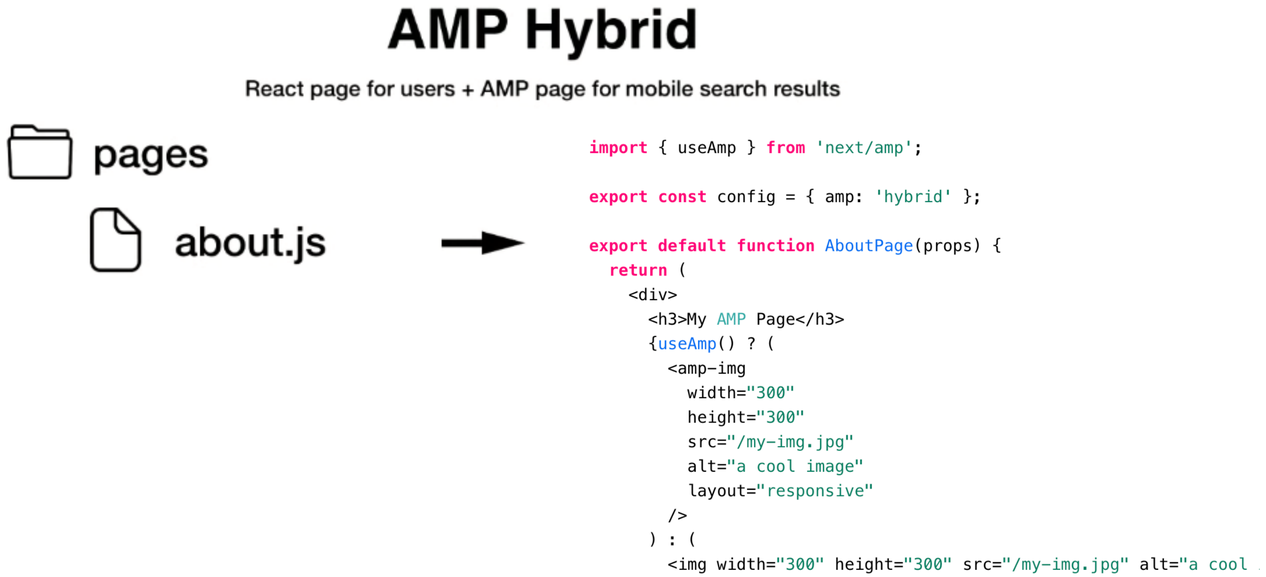
9. Client side rendering applications:

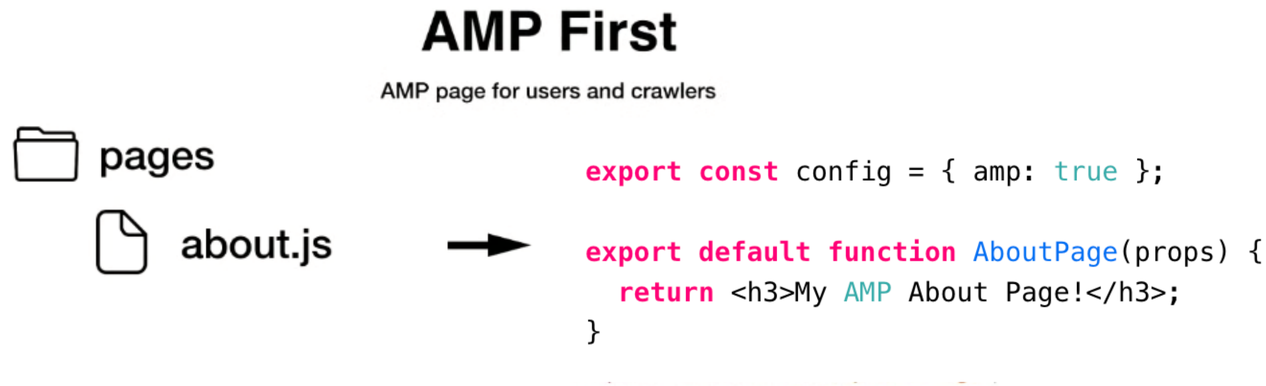


Taken from: https://www.youtube.com/watch?v=WKJVXN2kMS8

AMP Pages

10. Accelerated Mobile Pages, a imple mente project by Google with the goal of improving website load speed in mobiles





Taken from: https://www.youtube.com/watch?v=WKJVXN2kMS8

AMP-only:

Pages have no Next.js or React client-side runtime.

Pages are automatically optimized with AMP Optimizer, an optimizer that applies the same transformations as AMP caches (improves performance by up to 42%)

Pages have a user-accessible (optimized) version of the page and a search-engine indexable (unoptimized) version of the page

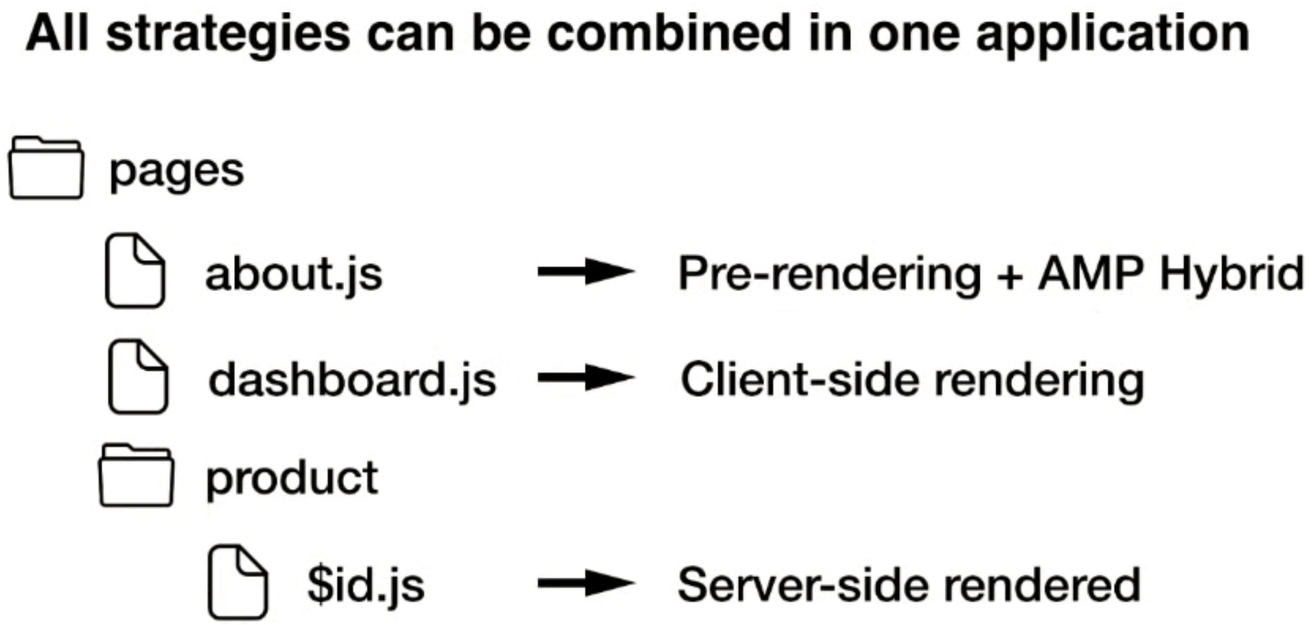
Hybrid:

Pages are able to be rendered as traditional HTML (default) and AMP HTML (by adding ?amp=1 to the URL)

The AMP version of the page only has valid optimizations applied with AMP Optimizer so that it is indexable by search-engines

Able to differentiate between modes using useAmp from next/amp

All strategies in the same project



Taken from: https://www.youtube.com/watch?v=WKJVXN2kMS8

API: Endpoints

API routes provides a straightforward solution to build your API with Next.js. Start by creating the api/ folder inside the ./pages/ folder.

./pages/api/posts.js

```
export default (req, res) => {
  res.setHeader('Content-Type', 'application/json');
  res.statusCode = 200;
  res.end(JSON.stringify({ name: 'Nextjs' }));
};
```

```
export default (req, res) => {
  if (req.method === 'POST') {
    // Process your POST request
  } else {
    // Handle the rest of your HTTP methods
  }
};
```

Custom Configuration

Create a next.config.js in the root of your project directory

```
// next.config.js
module.exports = {
  /* config options here */
};
```

```
const { PHASE_DEVELOPMENT_SERVER } = require('next/constants');
module.exports = (phase, { defaultConfig }) => {
  if (phase === PHASE_DEVELOPMENT_SERVER) {
    return {
      /* development only config options here */
    };
  }

  return {
    /* config options for all phases except development here */
  };
};
```

Custom Configuration

```
module.exports = withSass({
  webpack: (config) => {
    config.module.rules.push({
      enforce: 'pre',
      test: /.scss$/,
      loader: 'sass-resources-loader',
      options: {
        resources: ['./styles/globals.scss'],
      },
    });
    config.module.rules.push(
      {
        test: /\.svg$/,
        exclude: /node_modules/,
        loader: 'svg-react-loader',
      },
    );
    return config;
  },
  cssModules: true,
  cssLoaderOptions: {
    importLoaders: 1,
    localIdentName: '[local]',
  },
});
```

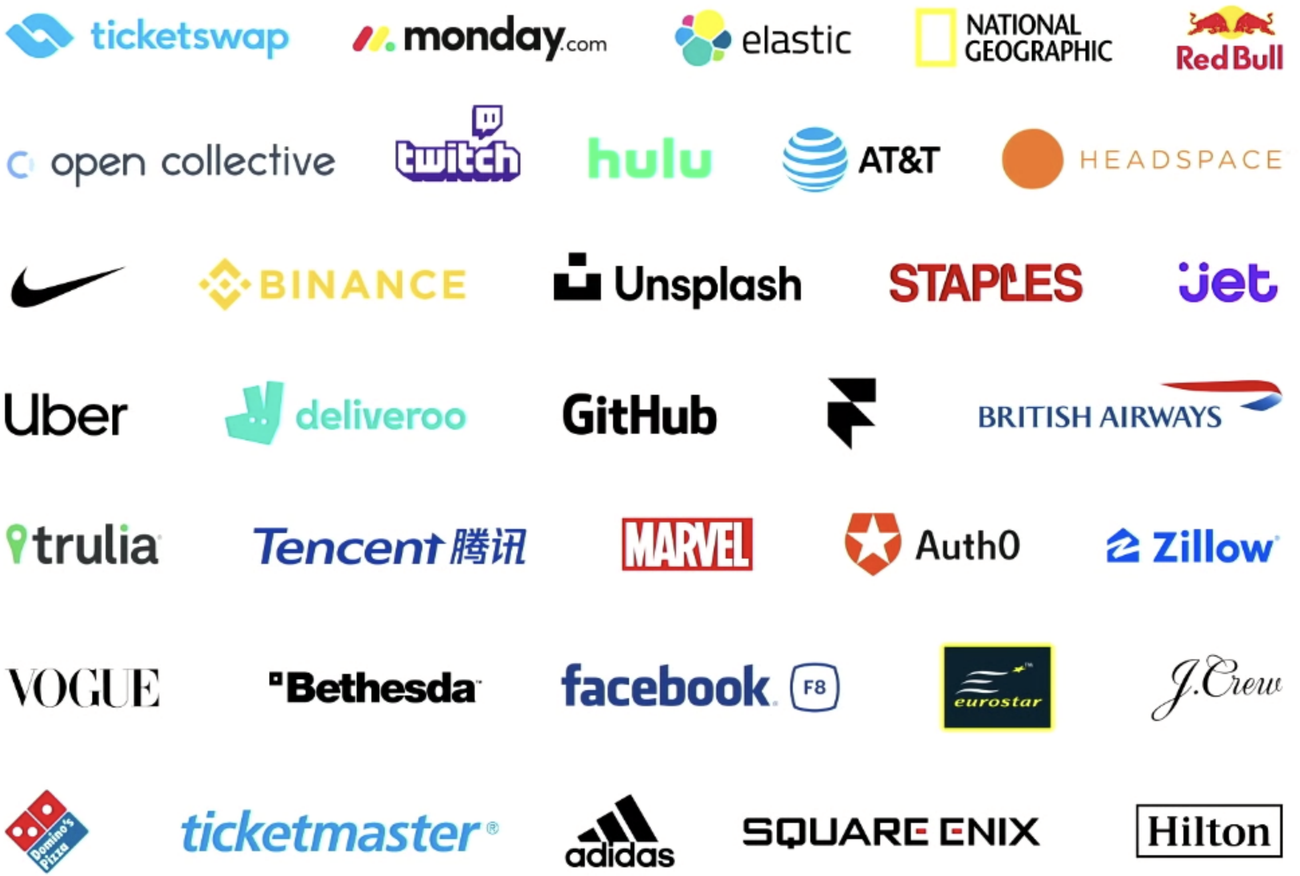

williamegomezo next-talk feat/config

One more thing

Next will include a feature that emulates how yarn works. It won't recompile again all the components of a page, but the ones that have changed.

Reduces build times from hours to milliseconds

Pages using next



Hulu study case

nextjs.org/case-studies/hulu

HUGE

Done.

Questions?