



React SSR

IVICA BATINIĆ



IVICA BATINIĆ

WHAT NEXT.JS DO FOR US?

- Automatic routing 📍
- Hot Code Reloading 🔥
- Server side rendering 🧑💻
- Code splitting 🛠️
- Prefetching 💉
- Static HTML exports 📦
- Custom server 🧑🔧
- React ecosystem compatibility ⚛️

AUTOMATIC ROUTING

- By default every `.js` file inside the `pages` folder is a route
- Every non existing route will be rendered as `404` page
- Route masking

```
1 import Link from 'next/link';
2
3 <Link href={` /post?title=${props.id}`} as={` /p/${props.id}`}>
4   <a>
5     {props.title}
6   </a>
7 </Link>
```

IMPERATIVE ROUTING

```
1 import Router from 'next/router';  
2  
3 <span  
4   onClick={() => Router.push(`/post?title=${props.id}`, `/p/${props.id}`)}  
5 >  
6   {props.title}  
7 </span>
```

HOT CODE RELOADING 🔥

- Enabled by default
- Better development experience
- Using `react-error-overlay`

SERVER SIDE RENDERING 🧑💻

- Improve SEO
- Better page speed



CODE SPLITTING

- Automatic
- Each **page** is separate bundle
- Every **import** you declare gets bundled and served with each page.
- Never load unnecessary code!
- Webpack **SplitChunksPlugin** is used to bundle common libs together

DYNAMIC IMPORT

```
1 import dynamic from 'next/dynamic';
2
3 const DynamicComponent = dynamic(
4   () => import('../components/hello'),
5   {
6     loading: () => (
7       <p>Loading</p>
8     ),
9     ssr: true,
10  }
11 );
12
13 function Home() {
14   return (
15     <div>
16       <Header />
17       <DynamicComponent />
18       <p>HOME PAGE is here!</p>
19     </div>
20   );
21 };
22
23 export default Home;
```

DYNAMIC IMPORT

```
1 import dynamic from 'next/dynamic';
2
3 const DynamicComponent = dynamic(
4   () => import('../components/hello'),
5   {
6     loading: () => (
7       <p>Loading</p>
8     ),
9     ssr: true,
10  }
11 );
12
13 function Home() {
14   return (
15     <div>
16       <Header />
17       <DynamicComponent />
18       <p>HOME PAGE is here!</p>
19     </div>
20   );
21 };
22
23 export default Home;
```

DYNAMIC IMPORT

```
1 import dynamic from 'next/dynamic';
2
3 const DynamicComponent = dynamic(
4   () => import('../components/hello'),
5   {
6     loading: () => (
7       <p>Loading</p>
8     ),
9     ssr: true,
10  }
11 );
12
13 function Home() {
14   return (
15     <div>
16       <Header />
17       <DynamicComponent />
18       <p>HOME PAGE is here!</p>
19     </div>
20   );
21 };
22
23 export default Home;
```

PREFETCHING

- **Next.js** allows us to prefetch pages to optimise the display speed
- To do this we just need to add the prefetch props to our **Link**

```
1 import Link from 'next/link';
2
3 function Header() {
4   return (
5     <Link prefetch href="/"><a>Home</a></Link>
6     <Link prefetch href="/about"><a>About</a></Link>
7   );
8 }
9
10 export default Header;
```

PREFETCHING

- **Next.js** allows us to prefetch pages to optimise the display speed
- To do this we just need to add the prefetch props to our **Link**

```
1 import Link from 'next/link';
2
3 function Header() {
4   return (
5     <Link prefetch href="/"><a>Home</a></Link>
6     <Link prefetch href="/about"><a>About</a></Link>
7   );
8 }
9
10 export default Header;
```

STATIC HTML EXPORT

- Next.js allows you to export everything to static files
- No need for a Node.js server
- Exported app supports almost every feature of Next.js, including dynamic urls, prefetching, preloading and dynamic imports.
- For dynamic routes add a dynamic exportPathMap in `next.config.js`

```
> next build  
> next export
```

```
1 // next.config.js
2 module.exports = {
3   exportPathMap: async function(defaultPathMap) {
4     return {
5       '/': {
6         page: '/',
7       },
8       '/about': {
9         page: '/about',
10      },
11      '/p/example': {
12        page: '/post',
13        query: {
14          title: 'example',
15        }
16      },
17    };
18  },
19 };
```

CUSTOM SERVER

- Next.js allows you to extend the base functionality with your prefer **node.js** framework


```
1 const express = require('express');
2 const next = require('next');
3
4 const port = parseInt(process.env.PORT, 10) || 3000;
5 const dev = process.env.NODE_ENV !== 'production';
6 const app = next({ dev });
7 const handle = app.getRequestHandler();
8
9 app.prepare().then(() => {
10   const server = express()
11
12   server.get('/posts/:id', (req, res) => {
13     return app.render(req, res, '/posts', { id: req.params.id });
14   });
15
16   server.get('*', (req, res) => {
17     return handle(req, res)
18   });
19
20   server.listen(port, err => {
21     if (err) throw err
22     console.log(`> Ready on http://localhost:${port}`)
23   });
24 });
```

```
1 const express = require('express');
2 const next = require('next');
3
4 const port = parseInt(process.env.PORT, 10) || 3000;
5 const dev = process.env.NODE_ENV !== 'production';
6 const app = next({ dev });
7 const handle = app.getRequestHandler();
8
9 app.prepare().then(() => {
10   const server = express()
11
12   server.get('/posts/:id', (req, res) => {
13     return app.render(req, res, '/posts', { id: req.params.id });
14   });
15
16   server.get('*', (req, res) => {
17     return handle(req, res)
18   });
19
20   server.listen(port, err => {
21     if (err) throw err
22     console.log(`> Ready on http://localhost:${port}`)
23   });
24 });
```

```
1 const express = require('express');
2 const next = require('next');
3
4 const port = parseInt(process.env.PORT, 10) || 3000;
5 const dev = process.env.NODE_ENV !== 'production';
6 const app = next({ dev });
7 const handle = app.getRequestHandler();
8
9 app.prepare().then(() => {
10   const server = express()
11
12   server.get('/posts/:id', (req, res) => {
13     return app.render(req, res, '/posts', { id: req.params.id });
14   });
15
16   server.get('*', (req, res) => {
17     return handle(req, res)
18   });
19
20   server.listen(port, err => {
21     if (err) throw err
22     console.log(`> Ready on http://localhost:${port}`)
23   });
24 });
```

```
1 const express = require('express');
2 const next = require('next');
3
4 const port = parseInt(process.env.PORT, 10) || 3000;
5 const dev = process.env.NODE_ENV !== 'production';
6 const app = next({ dev });
7 const handle = app.getRequestHandler();
8
9 app.prepare().then(() => {
10   const server = express()
11
12   server.get('/posts/:id', (req, res) => {
13     return app.render(req, res, '/posts', { id: req.params.id });
14   });
15
16   server.get('*', (req, res) => {
17     return handle(req, res)
18   });
19
20   server.listen(port, err => {
21     if (err) throw err
22     console.log(`> Ready on http://localhost:${port}`)
23   });
24 });
```

```
1 const express = require('express');
2 const next = require('next');
3
4 const port = parseInt(process.env.PORT, 10) || 3000;
5 const dev = process.env.NODE_ENV !== 'production';
6 const app = next({ dev });
7 const handle = app.getRequestHandler();
8
9 app.prepare().then(() => {
10   const server = express()
11
12   server.get('/posts/:id', (req, res) => {
13     return app.render(req, res, '/posts', { id: req.params.id });
14   });
15
16   server.get('*', (req, res) => {
17     return handle(req, res)
18   });
19
20   server.listen(port, err => {
21     if (err) throw err
22     console.log(`> Ready on http://localhost:${port}`)
23   });
24 });
```


REACT ECOSYSTEM COMPATIBILITY

- Next.js is compatible with the whole **react** and **npm** ecosystem
- Problems with libs that depends on **window** unconditionally

01 GETTING STARTED

PROJECT SETUP

- Install dependencies

```
npm install --save next react react-dom
```

- Populate `./pages/index.js` inside your project:

```
1 function Home( ) {  
2   return <div>Welcome to Next.js!</div>;  
3 }  
4  
5 export default Home;
```

carbon.now.sh

- Start dev server with `next` command

CREATE NEXT APP

```
$ npx create-next-app my-app  
$ cd my-app/  
$ npm run dev
```

- From Github [examples](#)

```
$ npx create-next-app --example basic-css example-app
```

GET INITIAL PROPS

- To load data when the page loads
- **async** static method
- Data returned from `getInitialProps` is serialized when server rendering
- For the initial page load, `getInitialProps` will execute on the server only
- Will only be executed on the client when navigating to a different route via the `Link` component or using the routing APIs
- Can't be used in children components. Only in **pages**

CLASS COMPONENT EXAMPLE

```
1 import React from 'react';
2
3 class Page extends React.Component {
4   static async getInitialProps({ req }) {
5     const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;
6     return { userAgent };
7   }
8
9   render() {
10    return <div>Hello World {this.props.userAgent}</div>;
11  }
12 }
13
14 export default Page;
```

CLASS COMPONENT EXAMPLE

```
1 import React from 'react';
2
3 class Page extends React.Component {
4   static async getInitialProps({ req }) {
5     const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;
6     return { userAgent };
7   }
8
9   render() {
10    return <div>Hello World {this.props.userAgent}</div>;
11  }
12 }
13
14 export default Page;
```

CLASS COMPONENT EXAMPLE

```
1 import React from 'react';
2
3 class Page extends React.Component {
4   static async getInitialProps({ req }) {
5     const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;
6     return { userAgent };
7   }
8
9   render() {
10    return <div>Hello World {this.props.userAgent}</div>;
11  }
12 }
13
14 export default Page;
```

CLASS COMPONENT EXAMPLE

```
1 import React from 'react';
2
3 class Page extends React.Component {
4   static async getInitialProps({ req }) {
5     const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;
6     return { userAgent };
7   }
8
9   render() {
10    return <div>Hello World {this.props.userAgent}</div>;
11  }
12 }
13
14 export default Page;
```

FUNCTION COMPONENT EXAMPLE

```
1 function Page({ stars }) {  
2   return <div>Hello World {this.props.userAgent}</div>;  
3 }  
4  
5 Page.getInitialProps = async ({ req }) => {  
6   const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;  
7   return { userAgent };  
8 };  
9  
10 export default Page;
```

FUNCTION COMPONENT EXAMPLE

```
1 function Page({ stars }) {  
2   return <div>Hello World {this.props.userAgent}</div>;  
3 }  
4  
5 Page.getInitialProps = async ({ req }) => {  
6   const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;  
7   return { userAgent };  
8 };  
9  
10 export default Page;
```


FUNCTION COMPONENT EXAMPLE

```
1 function Page({ stars }) {  
2   return <div>Hello World {this.props.userAgent}</div>;  
3 }  
4  
5 Page.getInitialProps = async ({ req }) => {  
6   const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;  
7   return { userAgent };  
8 };  
9  
10 export default Page;
```

FUNCTION COMPONENT EXAMPLE

```
1 function Page({ stars }) {  
2   return <div>Hello World {this.props.userAgent}</div>;  
3 }  
4  
5 Page.getInitialProps = async ({ req }) => {  
6   const userAgent = req ? req.headers['user-agent'] : navigator.userAgent;  
7   return { userAgent };  
8 };  
9  
10 export default Page;
```

SSR CHALLENGES

DETECT WHERE THE CODE IS RUNNING

- server or client?
- `window !== undefined`
- `process.browser`
- The `req` and `res` fields of the `context` object passed to `getInitialProps` are only available on the server

AUTHENTICATION

- Save user info in global store
- Validate user on server side
- Never store user sensitive data in LocalStorage
- Always use cookies 🍪

CLIENT SIDE AUTH

```
1 // login form onSubmit handler
2 const handleSubmit = async (event) => {
3   event.preventDefault();
4
5   if (error) {
6     setError(null);
7   }
8
9   try {
10    const response = await login(null, formData);
11    store['me'] = response;
12    redirect(null, '/');
13  } catch (error) {
14    setError('Wrong username or password!')
15  }
16 };
```

```
1 export async function login(ctx, payload) {
2   const rawResponse = await fetch(`http://localhost:5000/api/login`, {
3     method: 'POST',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6     body: JSON.stringify(payload),
7   });
8
9   if (rawResponse.ok) {
10    const response = await rawResponse.json();
11    return response;
12  } else {
13    throw null;
14  }
15 };
```

CLIENT SIDE AUTH

```
1 // login form onSubmit handler
2 const handleSubmit = async (event) => {
3   event.preventDefault();
4
5   if (error) {
6     setError(null);
7   }
8
9   try {
10    const response = await login(null, formData);
11    store['me'] = response;
12    redirect(null, '/');
13  } catch (error) {
14    setError('Wrong username or password!')
15  }
16 };
```

```
1 export async function login(ctx, payload) {
2   const rawResponse = await fetch(`http://localhost:5000/api/login`, {
3     method: 'POST',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6     body: JSON.stringify(payload),
7   });
8
9   if (rawResponse.ok) {
10    const response = await rawResponse.json();
11    return response;
12  } else {
13    throw null;
14  }
15 };
```

CLIENT SIDE AUTH

```
1 // login form onSubmit handler
2 const handleSubmit = async (event) => {
3   event.preventDefault();
4
5   if (error) {
6     setError(null);
7   }
8
9   try {
10    const response = await login(null, formData);
11    store['me'] = response;
12    redirect(null, '/');
13  } catch (error) {
14    setError('Wrong username or password!')
15  }
16 };
```

```
1 export async function login(ctx, payload) {
2   const rawResponse = await fetch(`http://localhost:5000/api/login`, {
3     method: 'POST',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6     body: JSON.stringify(payload),
7   });
8
9   if (rawResponse.ok) {
10    const response = await rawResponse.json();
11    return response;
12  } else {
13    throw null;
14  }
15 };
```


CLIENT SIDE AUTH

```
1 // login form onSubmit handler
2 const handleSubmit = async (event) => {
3   event.preventDefault();
4
5   if (error) {
6     setError(null);
7   }
8
9   try {
10    const response = await login(null, formData);
11    store['me'] = response;
12    redirect(null, '/');
13  } catch (error) {
14    setError('Wrong username or password!')
15  }
16 };
```

```
1 export async function login(ctx, payload) {
2   const rawResponse = await fetch(`http://localhost:5000/api/login`, {
3     method: 'POST',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6     body: JSON.stringify(payload),
7   });
8
9   if (rawResponse.ok) {
10    const response = await rawResponse.json();
11    return response;
12  } else {
13    throw null;
14  }
15 };
```

CLIENT SIDE AUTH

```
1 // login form onSubmit handler
2 const handleSubmit = async (event) => {
3   event.preventDefault();
4
5   if (error) {
6     setError(null);
7   }
8
9   try {
10    const response = await login(null, formData);
11    store['me'] = response;
12    redirect(null, '/');
13  } catch (error) {
14    setError('Wrong username or password!')
15  }
16 };
```

```
1 export async function login(ctx, payload) {
2   const rawResponse = await fetch(`http://localhost:5000/api/login`, {
3     method: 'POST',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6     body: JSON.stringify(payload),
7   });
8
9   if (rawResponse.ok) {
10    const response = await rawResponse.json();
11    return response;
12  } else {
13    throw null;
14  }
15 };
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```


SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  }));
13 }
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```


SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```

SERVER SIDE AUTH

```
1 // pages/_app.js
2 class MyApp extends App {
3   static async getInitialProps({ Component, ctx }) {
4     // ...
5     const isServer = ctx.req;
6     const store = getOrCreateStore(isServer);
7
8     if (isServer) {
9       const me = await getMe(ctx);
10      store['me'] = me;
11    }
12    // ...
13  }
14
15  constructor(props) {
16    super(props);
17    this.store = getOrCreateStore(false, props.store)
18  }
19
20  render() {
21    const { Component, pageProps } = this.props;
22
23    return (
24      <StoreProvider value={this.store}>
25        <Component {...pageProps} />
26      </StoreProvider>
27    );
28  }
29 }
30
31 export default MyApp;
```

```
1 export function getMe(ctx) {
2   return fetch(`http://localhost:5000/api/me`, {
3     method: 'GET',
4     credentials: 'include',
5     headers: getApiHeaders(ctx),
6   }).then((response) => {
7     if (response.ok) {
8       return response.json();
9     } else {
10      return null;
11    }
12  });
13 }
```

RENDERING SECURE ROUTES

- Use HOC to share logic between pages (withAuth)
- Check if data transferred from the server side includes logged in user info

```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```

```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```

```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```



```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```

```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```



```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```

```

1 import Error from '../pages/_error';
2
3 export function withAuth(Page) {
4   function WithAuth(props) {
5     // ...
6   }
7
8   WithAuth.getInitialProps = async function(ctx) {
9     // ...
10  }
11
12  return WithAuth;
13 }

```

```

1 function WithAuth(props) {
2   return props.isAuthenticated ? (
3     <Page {...props} />
4   ) : (
5     <Error statusCode={401} />
6   );
7 }

```

```

1 WithAuth.getInitialProps = async function(ctx) {
2   const { me } = ctx.store;
3
4   if (!me) {
5     if (ctx.res) {
6       ctx.res.statusCode = 401;
7     }
8
9     return {};
10  }
11
12  let pageProps;
13
14  if (Page.getInitialProps) {
15    pageProps = await Page.getInitialProps(ctx);
16  }
17
18  return {
19    ...pageProps,
20    isAuthenticated: Boolean(me),
21    me,
22  };
23 }

```

USAGE

```
1 import { withAuth } from '../libs/auth/withAuth';
2
3 function Profile(props) {
4   return (
5     <div>Profile token {props.me.email}</div>
6   );
7 }
8
9 Profile.getInitialProps = async function(ctx) {
10   console.log(ctx.me);
11   return {};
12 }
13
14 export default withAuth(Profile);
```

USAGE

```
1 import { withAuth } from '../libs/auth/withAuth';
2
3 function Profile(props) {
4   return (
5     <div>Profile token {props.me.email}</div>
6   );
7 }
8
9 Profile.getInitialProps = async function(ctx) {
10   console.log(ctx.me);
11   return {};
12 }
13
14 export default withAuth(Profile);
```

USAGE

```
1 import { withAuth } from '../libs/auth/withAuth';
2
3 function Profile(props) {
4   return (
5     <div>Profile token {props.me.email}</div>
6   );
7 }
8
9 Profile.getInitialProps = async function(ctx) {
10   console.log(ctx.me);
11   return {};
12 }
13
14 export default withAuth(Profile);
```

USAGE

```
1 import { withAuth } from '../libs/auth/withAuth';
2
3 function Profile(props) {
4   return (
5     <div>Profile token {props.me.email}</div>
6   );
7 }
8
9 Profile.getInitialProps = async function(ctx) {
10   console.log(ctx.me);
11   return {};
12 }
13
14 export default withAuth(Profile);
```

SETTING CORRECT RESPONSE STATUS CODE

- Only possible on first load
- Should be set on SSR
- Only inside `getInitialProps`

```
1 Page.getInitialProps = async function(ctx) {  
2   if (ctx.res) { // SSR check  
3     ctx.res.statusCode = 403;  
4   }  
5 };
```

REDIRECTS

```
1 import Router from 'next/router';
2
3 export function redirect(ctx, url) {
4   if(ctx.res) {
5     ctx.res.writeHead(302, {
6       Location: url
7     });
8     ctx.res.end();
9   } else {
10    Router.push(url);
11  }
12 }
```

```
1 import { redirect } from '../utils/redirect';
2
3 function Login() {
4   return (
5     <div>Login</div>
6   );
7 }
8
9 Login.getInitialProps = async function(ctx) {
10   if (loggedIn) {
11     redirect(ctx, '/')
12   }
13   return {};
14 }
15
16 export default Login;
```


REDIRECTS

```
1 import Router from 'next/router';
2
3 export function redirect(ctx, url) {
4   if(ctx.res) {
5     ctx.res.writeHead(302, {
6       Location: url
7     });
8     ctx.res.end();
9   } else {
10     Router.push(url);
11   }
12 }
```

```
1 import { redirect } from '../utils/redirect';
2
3 function Login() {
4   return (
5     <div>Login</div>
6   );
7 }
8
9 Login.getInitialProps = async function(ctx) {
10   if (loggedIn) {
11     redirect(ctx, '/')
12   }
13   return {};
14 }
15
16 export default Login;
```

REDIRECTS

```
1 import Router from 'next/router';
2
3 export function redirect(ctx, url) {
4   if(ctx.res) {
5     ctx.res.writeHead(302, {
6       Location: url
7     });
8     ctx.res.end();
9   } else {
10    Router.push(url);
11  }
12 }
```

```
1 import { redirect } from '../utils/redirect';
2
3 function Login() {
4   return (
5     <div>Login</div>
6   );
7 }
8
9 Login.getInitialProps = async function(ctx) {
10   if (loggedIn) {
11     redirect(ctx, '/')
12   }
13   return {};
14 }
15
16 export default Login;
```

REDIRECTS

```
1 import Router from 'next/router';
2
3 export function redirect(ctx, url) {
4   if(ctx.res) {
5     ctx.res.writeHead(302, {
6       Location: url
7     });
8     ctx.res.end();
9   } else {
10     Router.push(url);
11   }
12 }
```

```
1 import { redirect } from '../utils/redirect';
2
3 function Login() {
4   return (
5     <div>Login</div>
6   );
7 }
8
9 Login.getInitialProps = async function(ctx) {
10   if (loggedIn) {
11     redirect(ctx, '/')
12   }
13   return {};
14 }
15
16 export default Login;
```

STATE TRANSFER

- All data returned from `getInitialProps` is serialized to `__NEXT_DATA__`
- On first client load `getInitialProps` is skipped
- Avoiding unnecessary duplication of requests

USING ENVIRONMENT VARIABLES

- Defined in `next.config.js`
- Build time configuration:
 - `env`
- Runtime configuration:
 - `serverRuntimeConfig`
 - `clientRuntimeConfig`

```

1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;

```

```

1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client

```

```

1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;

```

```

1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client

```

```
1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;
```

```
1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client
```



```
1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;
```

```
1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client
```

```
1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;
```

```
1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client
```

```
1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;
```

```
1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client
```

```
1 // next.config.js
2 module.exports = {
3   env: {
4     customKey: 'value',
5   }
6 };
7
8 // pages/index.js
9 function Index() {
10   return <h1>The value of customKey is: {process.env.customKey}</h1>
11 }
12
13 export default Index;
```

```
1 // next.config.js
2 module.exports = {
3   serverRuntimeConfig: {
4     mySecret: 'secret',
5     secondSecret: process.env.SECOND_SECRET,
6   },
7   publicRuntimeConfig: {
8     staticFolder: '/static',
9   }
10 };
11
12 // pages/index.js
13 import getConfig from 'next/config';
14 const { serverRuntimeConfig, publicRuntimeConfig } = getConfig();
15
16 console.log(serverRuntimeConfig.mySecret); // Will only be available on the server side
17 console.log(publicRuntimeConfig.staticFolder); // Will be available on both server and client
```

03
DEMO



Questions?

Visit infinum.co or find us on social networks:

