Frequently Asked Questions

Is Next.js production ready?

Yes! Next.js is used by many of the top websites in the world. See the Showcase for more info.

How do I fetch data in Next.js?

Next.js provides a variety of methods depending on your use case. You can use:

Client-side rendering: Fetch data with useEffect or SWR inside your React components

Server-side rendering with getServerSideProps

Static-site generation with getStaticProps

Incremental Static Regeneration by adding the `revalidate` prop to getStaticProps

To learn more about data fetching, visit our data fetching documentation.

Why does Next.js have its own Router?

Next.js includes a built-in router for a few reasons:

It uses a file-system based router which reduces configuration

It supports shallow routing which allows you to change the URL without running data fetching methods

Routes are always lazy-loadable

If you're migrating from React Router, see the migration documentation.

Can I use Next.js with my favorite JavaScript library?

Yes! We have hundreds of examples in our examples directory.

Can I use Next.js with GraphQL?

Yes! Here's an example with Apollo and an example API Route with GraphQL.

Can I use Next.js with Redux?

Yes! Here's an example with Redux and an example with thunk.

Can I make a Next.js Progressive Web App (PWA)?

Yes! Here's our Next.js PWA Example.

Can I use a CDN for static assets?

Yes! When you deploy your Next.js application to Vercel, your static assets are automatically detected and served by the Edge Network. If you self-host Next.js, you can learn how to manually configure the asset prefix here.

How can I change the internal webpack config?

In most cases, no manual webpack configuration is necessary since Next.js automatically configures webpack. For advanced cases where more control is needed, refer to the custom webpack config documentation.

What is Next.js inspired by?

Many of the goals we set out to accomplish were the ones listed in The 7 principles of Rich Web Applications by Guillermo Rauch.

The ease-of-use of PHP is a great inspiration. We feel Next.js is a suitable replacement for many scenarios where you would otherwise use PHP to output HTML.

Unlike PHP, we benefit from the ES6 module system and every page exports a component or function that can be easily imported for lazy evaluation or testing.

As we were researching options for server-rendering React that didn't involve a large number of steps, we came across react-page (now deprecated), a similar approach to Next.js by the creator of React Jordan Walke.