Next.js vs React
The Difference and Which Framework to Choose

Definition

What Is React

React is a UI library created by Meta to build reactive apps based on event triggers. In a traditional approach, a website reloads when data needs to be changed, which means when something is clicked, the whole page reloads to show another state which in most cases can be slow to reload.

Furthermore, the concept of using React components stops the reprocessing of every code or logic for that particular page. React components can be stateless or stateful and only re-render within the scope of the applied state.

React is built to be declarative. That means you get to decide the workflow. You get to control how your app works, making React a powerful tool.

What Is Next.js

Next.js is a light framework built on top of React that makes it easy to create fast, server-rendered websites. It was created by the team at Vercel and has been open-source from the beginning.

Next.js is used by some of the biggest names in the tech industry, including Airbnb, Twitter, and Uber. One of Next.js's key features is its ability to automatically code-split your application, meaning that each page only loads the necessary JavaScript for that page view. This results in faster page loads and an improved user experience.

Another key feature is its seamless integration with React Hooks, allowing you to use stateful components without extra configuration.

Features

Features of React

JavaScript Syntax Extension

JSX is a combination of JavaScript and HTML. This syntax extension is basically used to create React elements.

const myjsxelement = < h1 > I LOVE Ninetailed! < /h1>;

Writing HTML directly in React is quite a pain as you always have to call **React.createElement(component, props, ...children)** every time you want to write HTML.

render() { **return** React.createElement('div', **null**, `Hello ${**this**.props.toWhat}`); }

But with JSX, all you need do is write the HTML you are most familiar with while Babel transpiles the code to JavaScript during runtime.

Component

Everything in React is a component. Multiple React components are coupled together to create simple user interfaces for very large and complex UI. Each of these components can have its logic and behaviors. Components are reusable in any part of the web page by just calling it.

Virtual DOM

The virtual DOM is an instance of the original or actual DOM. The browser's DOM is in its first slow and still re-renders everything to show a small state change which makes the case even worse, but the React team came up with a concept of creating another instance of the actual DOM with JavaScript, which is actually fast and when a state change occurs, the virtual DOM makes a comparison between it and another new virtual DOM.

If the two DOMs are equal, the actual DOM never gets touched; the actual DOM gets updated.

One-Way Data Binding

A components logic contains the data to be displayed on the user interface. The connection flow between the data displayed and the logic component is called data binding in React.js.

One-way binding means data flows from the parent components to the children only. This is a simpler way of data flow than two-way binding, which can become complex.

These data are passed using a read-only prop, which cannot pass back data to the parent component but has a way of communicating to the parent for state changes based on inputs.

Features of Next.js

The following features make Nextjs a cutting-edge tool for developers:

File System Routing

Next.js is a JavaScript framework that makes creating routes for your web app easy.

When you add a file to the page's directory, it is automatically available as a route. This makes it very easy to create complex web apps with many different pages.

In addition, Next.js provides a number of features that make it easy to manage your routes. For example, you can easily specify the order in which your routes are displayed, and you can also specify how your routes are displayed on mobile devices.

As a result, Next.js is an ideal choice for creating web apps with many different pages.

Server-Side Rendering

Next.js supports the rendering of pages on user requests on the server-side by generating a non-interactive HTML, while React uses JSON data and JavaScript instruction to make the page interactive on the client-side.

Static Site Generator

Statically generated web pages are said to be SEO optimized because of their speed, which can make Google rank that page higher. With Next.js supporting static page generation makes it stands out against React.

Image Optimization

The HTML **<img>** tag has been evolved by the Next.js team with built-in performance to help with picture optimization. To use this feature, the **next/image** component is imported. With this feature, images automatically resize to the screen size seamlessly, even images from a remote location. This new feature provides developers with an easy way to optimize their images for performance without having to manually resize or compress them.

In addition, the Next.js team has also added a built-in loading spinner that will be displayed while an image is loading, further improving the user experience. With these new features, the Next.js framework continues to be an excellent choice for building high-performance web applications.

Automatic Code Splitting

As your Next.js applications grow bigger, the size of a third-party library, CSS, and JavaScript files or bundles increases. Instead of downloading a large file on page load, these code/scripts can be split into smaller units, and for every feature required, these scripts are downloaded immediately, thereby increasing performance.

What is so intriguing about this is that Next.js does this automatically.

Typescript Support

Typescript is a superset of JavaScript that adds type checking and other features that can help to improve the quality of code. While Typescript is not required for Next.js, it can be a valuable tool for developers looking to improve the quality of their code.

API Route

Next.js provides a built-in way to create your own APIs, called API routes. With API routes, you can create your own endpoints and handle incoming requests however you want. You can use API routes to create a custom backend for your Next.js application or to expose data from your database to the front end. Either way, API routes give you a lot of flexibility in how you build your Next.js application.

Next.js vs. React Comparison

In this section, we will be making a comparison between two of the most used tool for front-end development in the JavaScript ecosystem to help you decide which of the tool to use based on your requirements.

Let’s delve right in:

State of Education

If you're already familiar with React, you'll find Next.js easy to learn. That's because Next.js is built on top of React, so it inherited all the benefits of React (such as being component-based and declarative) while also adding its own features and functionality.

React, however, has a low difficult learning level. Over time, resources have been made in a very substantial amount; with all these, the learning curve is not too steep.

Performance

Next.js is a static site generator, and static web pages tend to load the fastest because these pages are pre-rendered on build, cached, and severed over a CDN. It also supports server-side rendering, which is another fast rendering method.

React uses client-side rendering, which is considered to be rather slow but React can be configured to also use other rendering methods. The only difference is the work you have to put in to get it working.

Documentation

The learn-by-doing documentation on the Next.js website is a great resource. The documentation provides concise, step-by-step instructions that will help you get up and running in no time.

In addition, the learn-by-doing approach ensures that you don't just learn the theory but also get practical experience working with the tool. As a result, you'll be able to hit the ground running and start building Next.js applications as soon as you finish the documentation.

The React documentation is comprehensive and easy to follow. It includes tutorials, articles, and videos covering all React development's basics. In addition, the React community is very active, and there are many resources available to help developers solve problems they encounter. As a result, React is an excellent choice for developers who want to create modern user interfaces.

Configuration

React is also opinionated about how code should be organized, making it hard to work with other libraries with different conventions.

In addition, React doesn’t provide a lot of built-in features, so you have to build everything from scratch or use third-party libraries. This can be time-consuming and adds complexity to your project. However, the tradeoff is that React is easy to learn and use and makes development more efficient.

So while React may not be very configurable, it can still be a good choice for your project depending on your needs.

One of the benefits of using Next.js is that almost everything is configurable through the use of templates. For example, you can create **babelrc**, **jest.config**, and **eslintrc** files to customize the build process and linting rules for your project. This flexibility makes Next.js an attractive option for developers who want complete control over their project's configuration.

Experienced Developers

We can say that React is the winner here because of its popularity. Finding React developers for your web development or software project might not be so much of a headache.

While Next.js is a popular framework for building web applications, finding developers with experience working with the platform can be difficult. This is because Next.js is relatively new and has only recently gained widespread adoption.

Server-Side Rendering

Next.js supports SSR; you don’t need any configuration to make this happen. On the other hand, React doesn't support server-side rendering by default but can be done with some extra effort.

Overview of Next.js vs. React
Below is a table for a brief overview:

Yardstick	Next.js	React
Performance	Web apps built with Next.js are very performant thanks to SSR and SSG	No code splitting really causes poor performance in React apps
State of Education	Can be hard to learn if there is no prior knowledge of React	Easy
Configuration	Almost everything is configurable	Strict
Talent Pool	Narrow	Broad
Community	Small with a good amount of resources	Broad with lots of resources
Documentation	Well written	Well written
Development Cost	Low	Low
Feature	Server-side rendering, static site generation, automatic routing, build size optimization, fast refresh/reload	React is pretty much extensible and some of these features can be enabled
SEO	More SEO friendly	Slightly SEO friendly
Third-Party API	It is possible to have third-party API using API routes	React is mainly focused on building UI
Crossplatform Applications	Next.js is mainly meant for the web	React native is based on the React.js library
Typescript	Supports	Supports
Image Optimization	Makes pictures adaptable in small viewports	It isn't built but can be achieved using third-party libraries
Offline Support	Supports	Supports
Dynamic Route	Most SSR frameworks support dynamic routes	Routes

Is Next.js Better than React?

When deciding whether to use React or Next.js for a project, it is important to consider the specific needs of the project or problem you are trying to solve.

While React is a great general-purpose solution, Next.js offers some advantages for specific types of projects. For example, if you are building a static site or an application that doesn't require complex routing, Next.js can be a good choice because it automates many of the build processes and offers built-in support for server-side rendering.

On the other hand, if you are working on a large-scale project that requires dynamic routing and heavily data-driven components, React may be a better option.

Ultimately, there is no clear "winner" between Next.js vs React - it all depends on the particular requirements of your project.

When to Use React Over Next.js?

React is a great option when you're working on a large-scale web application that requires complex routing and heavily data-driven components. React is a front-end JavaScript library that helps developers create user interfaces and reusable UI components. When used in combination with other technologies such as Redux, React Router, and Webpack, React can be used to build large-scale web applications. One of the benefits of using React is that it makes code maintenance easy by allowing developers to create isolated components that can be reused throughout the application. React also has a broad talent pool, which makes it easier to find experienced developers to work on your project.

Furthermore, React is a great choice for building client-side rendered applications. It uses a virtual DOM to efficiently update and render components and has a number of features that make it easy to work with data. React supports both one-way and two-way data binding, which means that you can easily bind data from your application state to your UI. In addition, React comes with a number of additional libraries that can be used to add extra functionality to your application.

So if you're working on a project that requires complex routing and heavily data-driven components, or if you're building a client-side rendered application, React may be the right choice for you.