React.js vs Next.js

Routing

React.js:

- **Manual Routing:** React itself does not include a built-in routing solution. You typically use libraries like React Router for routing.

Next.js:

- **File-Based Routing:** Next.js has a built-in file-based routing system, which automatically creates routes based on the file structure in the`pages`directory.

Rendering Methods

React.js:

- **Client-Side Rendering (CSR):** React renders components on the client side after the initial HTML is loaded.

Next.js:

- **Server-Side Rendering (SSR):** Next.js can render pages on the server before sending the HTML to the client.

- **Static Site Generation (SSG):** Generates static HTML at build time for better performance.

- **Incremental Static Regeneration (ISR):** Allows you to update static content after the site is built.

Purpose and Scope

React.js:

- **Library:** React.js is a JavaScript library for building user interfaces, primarily for single-page applications (SPAs).
- **Focus:** It focuses on the view layer of the application (i.e., the component-based UI).

Next.js:

- **Framework:** Next.js is a React framework that provides a more complete solution for building web applications.
- **Focus:** Includes additional features such as server-side rendering (SSR), static site generation (SSG), and API routes.

Performance

React.js:

- **Depends on Implementation:** Performance depends on how the developer implements optimizations.

Next.js:

- **Optimized by Default:** Offers optimized performance through SSR, SSG, and code splitting by default.

Configuration and Setup

React.js:

- **Flexible:** Requires more setup and configuration, giving you the flexibility to choose your own tools and structure.
- **Tooling:** Often set up using Create React App (CRA) which provides a standard structure but is still less opinionated compared to Next.js.

Next.js:

- **Convention over Configuration:** Comes with built-in conventions and requires less setup, making it easier to get started with common patterns.
- **Pre-Configured:** Provides a lot of configuration out of the box (e.g., webpack, Babel).

Development Experience

React.js:

- **Component-Based:** Focuses purely on building UI components, which can be integrated into various parts of an application.
- **Hot Module Replacement:** Supported through tools like Webpack for faster development.

Next.js:

- **Full Stack Development:** Supports both frontend and backend development within the same project.
- **Hot Reloading:** Comes with hot reloading out of the box for a smoother development experience.

API Routes

React.js:

- **External APIs:** React does not handle backend logic or APIs directly. You need to set up a separate backend server.

Next.js:

- **Built-In API Routes:** Allows you to create API routes within the same project, providing a simple way to handle server-side logic and endpoints.

Deployment

React.js:

- **External APIs:** Static Files: Typically results in a bundle of static files that can be served by any static file server.

Next.js:

- **Versatile Deployment:** Can be deployed as a static site, on serverless platforms, or traditional server environments.
- **Vercel:** Officially supported by Vercel for deployment, providing seamless integration and additional features.

Did you find it Useful?

Leave a comment!

Alamin CodePapa

@CodePapa360

FOLLOW FOR MORE

Like Comment Repost