Server Engine

Nuxt 3 is powered by a new server engine, Nitro.

- ✓ Cross-platform support for Node.js, Browsers, service-workers and more.
- ✓ Serverless support out-of-the-box.
- ✓ API routes support.
- ✓ Automatic code-splitting and async-loaded chunks.
- ✓ Hybrid mode for static + serverless sites.
- ✓ Development server with hot module reloading.

API Layer

Server API endpoints and Middleware are added by Nitro that internally uses h3.

Key features include:

- Handlers can directly return objects/arrays for an automatically-handled JSON response
- Handlers can return promises, which will be awaited (`res.end()` and `next()` are also supported)
- Helper functions for body parsing, cookie handling, redirects, headers and more

Check out the h3 docs for more information.

ℹ Learn more about the API layer in the `server/` directory.

Direct API Calls

Nitro allows 'direct' calling of routes via the globally-available `$fetch` helper. This will make an API call to the server if run on the browser, but will directly call the relevant function if run on the server, **saving an additional API call**.

`$fetch` API is using ofetch, with key features including:

- Automatic parsing of JSON responses (with access to raw response if needed)
- Request body and params are automatically handled, with correct `Content-Type` headers

For more information on `$fetch` features, check out ofetch.

Typed API Routes

When using API routes (or middleware), Nitro will generate typings for these routes as long as you are returning a value instead of using `res.end()` to send a response.

You can access these types when using `$fetch()` or `useFetch()` .

Standalone Server

Nitro produces a standalone server dist that is independent of `node_modules` .

The server in Nuxt 2 is not standalone and requires part of Nuxt core to be involved by running `nuxt start` (with the `nuxt-start` or `nuxt` distributions) or custom programmatic usage, which is fragile and prone to breakage and not suitable for serverless and service-worker environments.

Nuxt 3 generates this dist when running `nuxt build` into a `.output` directory.

The output contains runtime code to run your Nuxt server in any environment (including experimental browser service workers!) and serve your static files, making it a true hybrid framework for the JAMstack. In addition, Nuxt implements a native storage layer, supporting multi-source drivers and local assets.

ℹ Check out the Nitro engine on GitHub: unjs/nitro.

Edit on Github

Enterprise Design Kit NuxtLabs

Nuxt Studio