Creating a Nuxt App

Now that you have the Nuxt Prerequisites, in this lesson we gonna start with the creation of our Nuxt app. We will use the Nuxt CLI to create our app, and we will be exploring a couple of things in the Nuxt app, like the folder structure, and the pages, layouts, components, and assets directories.

Prerequisites Recap

From the prerequisites lesson, we're going to assume that you've a nodejs version of 18, 19, 20, and 21 installed, the Yarn package manager, and the Nuxt CLI.

Creating our Initial Nuxt.js App

To get started creating our application with the Nuxt.js, create-nuxt-app, Create Nuxt App is a command-line interface (CLI) tool that helps you quickly set up a new Nuxt.js project. It simplifies the process of setting up a new project by automatically installing dependencies and generating a basic project structure. It's a convenient way to get started with Nuxt.js without manually configuring everything from scratch. To use create-nuxt-app, you typically install it globally using npm or yarn and then run the command to create a new project.

To get started, make sure you're using a version of Yarn 1.5.1 or higher. Then, create a new project by running the following command:

```
yarn create nuxt-app <project-name>
```

Then, navigate to the project directory (replace `my-project` with the actual name of your project):

```
cd my-project
```

Finally, start the development server:

```
yarn dev
```

This will start a local server, typically at `http://localhost:3000`. You can open this URL in your web browser to see your Nuxt.js application in action.

Keep in mind that `create-nuxt-app` is a convenient way to quickly start a new project, but you may need to customize the generated project structure and configuration files to suit your specific requirements.

What Yoe (not Nuxt) Would Have Given Us

```
$ yarn create nuxt-app <project-name>
create-nuxt-app v3.12.0
Generating Nuxt.js project in project-name
? Project name: project-name
? Programming language: JavaScript
? Package manager: Yarn
? UI framework: none
? Nuxt.js modules: jsdoc
? Linting tools: prettier, standard
? Testing framework: none
? Rendering mode: Single Page App
? Deployment target: Static (Static/Jamstack)
? Development tools: jsconfig.json, semantic-release, commitlint
? What would you like to do? Create project
? What would you like to do? Open interactive shell
$ cd project-name
$ yarn dev
$ nuxt
Nuxt 2.15.8 (socket-io: 2.3.0, connect: 3.7.0, serve-static: 1.14.1, vue-server-renderer: 2.6.14, vue-template-compiler: 2.6.14, chokidar: 3.5.3, hash-sum: 1.0.2, lodash.template: 4.5.0, minimatch: 3.0.4, mkdirp: 0.5.5, node-fetch: 2.6.9, open: 7.4.2, pathe: 0.2.0, vue-bundle-renderer: 0.4.1, consola: 2.15.3, bibtexparse: 2.0.0, defu: 3.0.1, esm: 3.2.25, hookable: 2.0.2, ioredis: 4.28.5, jiti: 1.21.6, mime: 3.0.0, mlly: 1.4.2, pm2-axon: 4.0.1, pm2-axon-rpc: 0.7.1, pretty-bytes: 5.3.0, radix3: 1.1.0, serve-placeholder: 2.0.2, serve-static: 1.14.1, ufo: 1.5.4, unctx: 2.3.1, unenv: 1.10.0, unstorage: 1.10.2, vue-meta: 2.4.0, vue-server-renderer: 2.6.14, vue-template-compiler: 2.6.14, webpack: 4.46.0, webpack-bundle-analyzer: 4.9.1, webpack-dev-middleware: 3.7.3, webpack-hot-middleware: 2.25.4, webpack-node-externals: 3.5.1, ws: 7.4.6, zlib-to-gzip-stream: 1.1.1)
Preparing project for development
Initial build may take a while
Discovered Components: components/README.md
Discovered Layouts: layouts/default.vue
Discovered Pages: pages/index.vue
Discovered Middleware: middleware/middleware.js
Discovered Store modules: store/index.js
Listening on: http://172.28.0.2:3000/
Waiting for file changes
Memory usage: 303 MB (RSS: 1.1 GB)
Finished building all sources
```

This would be a good point to create our first git commit, and add a repo to github or wherever you like to store your source code.

Now we're in a good position to start building the actually app. Before we get started, let's give some the folder structure that was created for us by `create-nuxt-app`.

What Yoe (not Nuxt) Would Have Given Us

```
$ yarn create nuxt-app <project-name>
create-nuxt-app v3.12.0
Generating Nuxt.js project in project-name
? Project name: project-name
? Programming language: JavaScript
? Package manager: Yarn
? UI framework: none
? Nuxt.js modules: jsdoc
? Linting tools: prettier, standard
? Testing framework: none
? Rendering mode: Single Page App
? Deployment target: Static (Static/Jamstack)
? Development tools: jsconfig.json, semantic-release, commitlint
? What would you like to do? Create project
? What would you like to do? Open interactive shell
$ cd project-name
$ yarn dev
$ nuxt
Nuxt 2.15.8 (socket-io: 2.3.0, connect: 3.7.0, serve-static: 1.14.1, vue-server-renderer: 2.6.14, vue-template-compiler: 2.6.14, chokidar: 3.5.3, hash-sum: 1.0.2, lodash.template: 4.5.0, minimatch: 3.0.4, mkdirp: 0.5.5, node-fetch: 2.6.9, open: 7.4.2, pathe: 0.2.0, vue-bundle-renderer: 0.4.1, consola: 2.15.3, bibtexparse: 2.0.0, defu: 3.0.1, esm: 3.2.25, hookable: 2.0.2, ioredis: 4.28.5, jiti: 1.21.6, mime: 3.0.0, mlly: 1.4.2, pm2-axon: 4.0.1, pm2-axon-rpc: 0.7.1, pretty-bytes: 5.3.0, radix3: 1.1.0, serve-placeholder: 2.0.2, serve-static: 1.14.1, ufo: 1.5.4, unctx: 2.3.1, unenv: 1.10.0, unstorage: 1.10.2, vue-meta: 2.4.0, vue-server-renderer: 2.6.14, vue-template-compiler: 2.6.14, webpack: 4.46.0, webpack-bundle-analyzer: 4.9.1, webpack-dev-middleware: 3.7.3, webpack-hot-middleware: 2.25.4, webpack-node-externals: 3.5.1, ws: 7.4.6, zlib-to-gzip-stream: 1.1.1)
Preparing project for development
Initial build may take a while
Discovered Components: components/README.md
Discovered Layouts: layouts/default.vue
Discovered Pages: pages/index.vue
Discovered Middleware: middleware/middleware.js
Discovered Store modules: store/index.js
Listening on: http://172.28.0.2:3000/
Waiting for file changes
Memory usage: 303 MB (RSS: 1.1 GB)
Finished building all sources
```

The Nuxt.js Component Folder Structure

Open the project generated by `create-nuxt-app`, in your code editor of choice, and you should see a folder structure similar to the following application in the VS Code editor.

components
layouts
pages
middleware
store
static
assets
.gitignore
.prettierignore
.prettierrc
.editorconfig
.eslintignore
.eslintrc.js
.gitattributes
.gitignore
.jsdoc.conf.js
.nuxt
.nuxtignore
.nuxtrc
.npmrc
.yarnrc
.yarn
.yarnrc.yml
.yarn.lock
.yarn.lock.bak
.yarn.lock.bak.bak
.yarn.lock.bak.bak.bak
.yarn.lock.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak
.yarn.lock.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak.bak

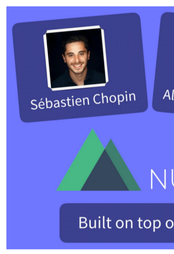
Why Use Nuxt.js?

7 Problems you can avoid by using Nuxt.js for your next Vue app

Vue.js is a great choice as a framework for your application. But, there are a couple concerns you'll likely run into:

- How do I combine and configure all my JavaScript & Vue libraries to work together effectively?
- How do I structure my application code using best practices?
- How do I make sure my application is properly indexed by search engines?
- How do I optimize my Vue application for speed?

These are some of the concerns that Sebastien and Alexandre Chopin set out to solve when they created Nuxt.js, a higher-level framework that's built *on top of* Vue to help you build production-ready Vue applications.

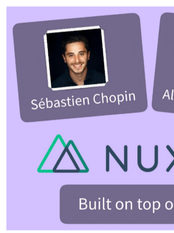


I want to let you in 7 problems you're going to run into building a Vue app, and tell you how Nuxt solves them.

Problem #1: Building a production-ready Vue app from scratch is difficult

Not only does Nuxt come preconfigured with Vuex, Vue Router, and Vue-Meta, but it sets your project up with intelligent defaults based on well researched best practices that Vue won't give you out of the box.

Creating a Nuxt app is as easy as typing:



This Nuxt Starter Kit will ask you for the libraries you want to start your project with, such as CSS Lint, and which CSS framework to use.

Problem #2: No standard folder structure

As your application grows your code organization becomes more important. Vue's default build gives you an **assets** and **components** directory, which is a good start. Nuxt.js sets you up with additional folders based on best practices, such as:

A `pages` directory for your Application views and routes.

A `layouts` directory for your layout templates.

A `store` directory for your Vuex store files.

Nuxt ships with more folders and more default conventions which are all configurable, but you get the idea.

Since everything has its place, moving from one Nuxt application to another as a developer and getting up to speed is super simple.

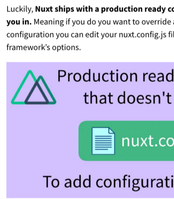
Problem #3: Routing configuration can get long in a big Vue app

With Nuxt, you simply place your single file Vue components into the pages folder and Nuxt automatically generates your routes with zero configuration.

This is just one more example of how Nuxt gives you the pre-configuration you need to create production-ready Vue applications. This leads us to the next problem with big applications…

Problem #4: No standard way to configure things all together

Luckily, **Nuxt ships with a production ready configuration that doesn't lock you in.** Meaning if you do you want to override any of the smart defaults or pre-configuration you can edit your nuxt.config.js file and modify any of the framework's options.



Problem #5: Vue apps aren't not SEO-friendly

You're going to want certain pages of your app properly indexed by search engines so they're easily discoverable.

One of the best solutions is to pre-render your Vue pages on the server, but this can be tricky to setup on your own. Nuxt is pre-configured to generate your application on the server, along with powering up your routes to make it easy to add SEO-related tags.

Problem #6: Vue applications on initial load can be slow

With Nuxt.js if you choose to render your application universally or statically, the pre-rendered HTML causes your page to load faster in the browser. Once it's loaded, it'll start running as a normal SPA. Plus with automatic code-splitting it's only going to load the JavaScript needed to make the route function.

This makes for the optimal user experience.

Problem #7: With a Vue app you may find it difficult to change the behavior of the framework

When you are developing production-grade applications, at some point you'll need to change the behavior of the framework. Nuxt provides a **higher-order module system** that makes it easy to customize every aspect of Nuxt.

🏁 Conclusion

So there you have it, 7 problems that Nuxt solves for you. As you can see, **Nuxt.js allows you to spend less time on configuration, and more time solving problems and building awesome Vue apps.**

To get started with Nuxt we recommend jumping into the great documentation, and consider downloading the Vue Mastery Nuxt cheat sheet. By doing this you'll also be notified when we start releasing additional lessons of our Nuxt course.