Register 7













Table of Contents

TypeScript

Remove Modules

- Remove @nuxt/typescript-build: Bridge enables same functionality
- Remove @nuxt/typescript-runtime and nuxt-ts: Nuxt 2 has built-in runtime support

Set bridge.typescript

```
import { defineNuxtConfig } from '@nuxt/bridge'
export default defineNuxtConfig({
 bridge: {
   typescript: true,
    nitro: false // If migration to Nitro is complete, set to true
  }
})
```

Update tsconfig.json

If you are using TypeScript, you can edit your tsconfig.json to benefit from auto-generated Nuxt types:

```
tsconfig.json
+ "extends": "./.nuxt/tsconfig.json",
  "compilerOptions": {
```

```
...
}
}
```

As .nuxt/tsconfig.json is generated and not checked into version control, you'll need to generate that file before running your tests. Add nuxi prepare as a step before your tests, otherwise you'll see TS5083: Cannot read file '~/.nuxt/tsconfig.json'

You may also need to add <code>@vue/runtime-dom</code> as a devDependency if you are struggling to get template type inference working with <code>Volar</code>.

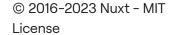
Keep in mind that all options extended from ./.nuxt/tsconfig.json will be overwritten by the options defined in your tsconfig.json . Overwriting options such as "compilerOptions.paths" with your own configuration will lead TypeScript to not factor in the module resolutions from

./.nuxt/tsconfig.json . This can lead to module resolutions such as #imports not being recognized.

In case you need to extend options provided by ./.nuxt/tsconfig.json further, you can use the alias property within your nuxt.config . nuxi will pick them up and extend ./.nuxt/tsconfig.json accordingly.

Edit on Github





Enterprise Design Kit NuxtLabs

Nuxt Studio





