useRouter

The `useRouter` composable returns the router instance and must be called in a setup function, plugin, or route middleware.

Within the template of a Vue component, you can access the router using `$router` instead.

If you have a `pages/` folder, `useRouter` is identical in behavior to the one provided by `vue-router`. Feel free to read the router documentation for more information on what each method does.

> 👉 Read more in https://router.vuejs.org/api/interfaces/Router.html#Properties-currentRoute.

Basic Manipulation

- **addRoute:** Add a new route to the router instance. `parentName` can be provided to add new route as the child of an existing route.
- **removeRoute:** Remove an existing route by its name.
- **getRoutes:** Get a full list of all the route records.
- **hasRoute:** Checks if a route with a given name exists.

Based on History API

- **back:** Go back in history if possible, same as `router.go(-1)`.
- **forward:** Go forward in history if possible, same as `router.go(1)`.

- **go:** Move forward or backward through the history without the hierarchical restrictions enforced in `router.back()` and `router.forward()` .
- **push:** Programmatically navigate to a new URL by pushing an entry in the history stack. **It is recommended to use `navigateTo` instead.**
- **replace:** Programmatically navigate to a new URL by replacing the current entry in the routes history stack. **It is recommended to use `navigateTo` instead.**

> TIP: `router.addRoute()` adds route details into an array of routes and it is useful while building Nuxt plugins while `router.push()` on the other hand, triggers a new navigation immediately and it is useful in Nuxt Page components, Vue components and composable.

```js
const router = useRouter();
router.back();
router.forward();
router.go();
router.push({ path: "/home" });
router.replace({ hash: "#bio" });
```

👉 Read more in https://developer.mozilla.org/en-US/docs/Web/API/History.

Navigation Guards

`useRouter` composable provides `afterEach` , `beforeEach` and `beforeResolve` helper methods that acts as navigation guards.

However, Nuxt has a concept of **route middleware** that simplifies the implementation of navigation guards and provides a better developer experience.

👉 Read more in Docs > Guide > Directory Structure > Middleware.

Promise and Error Handling

- **isReady:** Returns a Promise that resolves when the router has completed the initial navigation.
- **onError:** Adds an error handler that is called every time a non caught error happens during navigation.

- **resolve:** Returns the normalized version of a route location. Also includes an `href` property that includes any existing base.

> 👉 Read more in https://router.vuejs.org/api/interfaces/Router.html#Methods.

Universal Router Instance

If you do not have a `pages/` folder, then `useRouter` will return a universal router instance with similar helper methods, but be aware that not all features may be supported or behave in exactly the same way as with `vue-router`.

Edit on Github