useRuntimeConfig

The `useRuntimeConfig` composable is used to expose config variables within your app.

Usage

app.vue

```vue
<script setup lang="ts">
const config = useRuntimeConfig()
</script>
```

server/api/foo.ts

```ts
export default defineEventHandler((event) => {
  const config = useRuntimeConfig()
})
```

Define Runtime Config

The example below shows how to set a public API base URL and a secret API token that is only accessible on the server.

We should always define `runtimeConfig` variables inside `nuxt.config`.

nuxt.config.ts

```ts
export default defineNuxtConfig({
  runtimeConfig: {
    // Private keys are only available on the server
    apiSecret: '123',
```

```
    // Public keys that are exposed to the client
    public: {
      apiBase: process.env.NUXT_PUBLIC_API_BASE || '/api'
    }
  }
})
```

Variables that need to be accessible on the server are added directly inside `runtimeConfig`. Variables that need to be accessible on both the client and the server are defined in `runtimeConfig.public`.

👉 Read more in Docs > Guide > Going Further > Runtime Config.

Access Runtime Config

To access runtime config, we can use `useRuntimeConfig()` composable:

server/api/test.ts

```
export default async () => {
  const config = useRuntimeConfig()

  // Access public variables
  const result = await $fetch(`/test`, {
    baseURL: config.public.apiBase,
    headers: {
      // Access a private variable (only available on the server)
      Authorization: `Bearer ${config.apiSecret}`
    }
  })
  return result
}
```

In this example, since `apiBase` is defined within the `public` namespace, it is universally accessible on both server and client-side, while `apiSecret` **is only accessible on the server-side**.

Environment Variables

It is possible to update runtime config values using a matching environment variable name prefixed with `NUXT_`.

👉 Read more in Docs > Guide > Going Further > Runtime Config.

Using the `.env` File

We can set the environment variables inside the `.env` file to make them accessible during **development** and **build/generate**.

```
NUXT_PUBLIC_API_BASE = "https://api.localhost:5555"
NUXT_API_SECRET = "123"
```
.env

Any environment variables set within `.env` file are accessed using `process.env` in the Nuxt app during **development** and **build/generate**.

In **production runtime**, you should use platform environment variables and `.env` is not used.

When using git, make sure to add `.env` to the `.gitignore` file to avoid leaking secrets to the git history.

`app` namespace

Nuxt uses `app` namespace in runtime-config with keys including `baseURL` and `cdnURL`. You can customize their values at runtime by setting environment variables.

This is a reserved namespace. You should not introduce additional keys inside `app`.

`app.baseURL`

By default, the `baseURL` is set to `'/'`.

However, the `baseURL` can be updated at runtime by setting the `NUXT_APP_BASE_URL` as an environment variable.

Then, you can access this new base URL using `config.app.baseURL` :

/plugins/my-plugin.ts

```ts
export default defineNuxtPlugin((NuxtApp) => {
  const config = useRuntimeConfig()

  // Access baseURL universally
  const baseURL = config.app.baseURL
})
```

app.cdnURL

This example shows how to set a custom CDN url and access them using `useRuntimeConfig()` .

You can use a custom CDN for serving static assets inside `.output/public` using the `NUXT_APP_CDN_URL` environment variable.

And then access the new CDN url using `config.app.cdnURL` .

server/api/foo.ts

```ts
export default defineEventHandler((event) => {
  const config = useRuntimeConfig()

  // Access cdnURL universally
  const cdnURL = config.app.cdnURL
})
```

👉 Read more in Docs > Guide > Going Further > Runtime Config.

Edit on Github