# Lifecycle Hooks

Nuxt provides a powerful hooking system to expand almost every aspect using hooks. This feature is powered by unjs/hookable.

## Nuxt Hooks (Build Time)

These hooks are available for Nuxt Modules and build context.

### Usage with `nuxt.config`

nuxt.config

```
export default defineNuxtConfig({
  hooks: {
    'close': () => { }
  }
})
```

## Usage with Nuxt Modules

```
import { defineNuxtModule } from '@nuxt/kit'

export default defineNuxtModule({
  setup (options, nuxt) {
    nuxt.hook('close', async () => { })
  }
```

```
})
```

# App Hooks (Runtime)

App hooks can be mainly used by Nuxt Plugins to hook into rendering lifecycle but could also be used in Vue composables.

## Usage with Plugins

plugins/test.ts

```ts
export default defineNuxtPlugin((nuxtApp) => {
  nuxtApp.hook('page:start', () => {
    /* your code goes here */
  })
})
```

👉 Learn more about available lifecycle hooks

# Nitro App Hooks (Runtime)

These hooks are available for Nitro plugins to hook into Nitro's runtime behavior.

## Usage within a Nitro Plugin

~/server/plugins/test.ts

```ts
export default defineNitroPlugin((nitroApp) => {
  nitroApp.hooks.hook('render:html', (html, { event }) => {
    console.log('render:html', html)
    html.bodyAppend.push('<hr>Appended by custom plugin')
  })

  nitroApp.hooks.hook('render:response', (response, { event }) => {
    console.log('render:response', response)
  })
```

```
})
```

👉 Learn more about available Nitro lifecycle hooks.

# Add additional hooks

You can add additional hooks by augmenting the types provided by Nuxt. This can be useful for modules.

```ts
import { HookResult } from "@nuxt/schema";

declare module '#app' {
  interface RuntimeNuxtHooks {
    'your-nuxt-runtime-hook': () => HookResult
  }
  interface NuxtHooks {
    'your-nuxt-hook': () => HookResult
  }
}

declare module 'nitropack' {
  interface NitroRuntimeHooks {
    'your-nitro-hook': () => void;
  }
}
```

Edit on Github