ConfigurableModuleHost

August 18, 2026

ConfigurableModuleHost

Kind: Interface

Source: `packages/common/module-utils/interfaces/configurable-module-host.interface.ts`

Part of: Common

Configurable module host. See properties for more details

`ConfigurableModuleHost` describes the object produced by NestJS's configurable module builder. It groups the generated module class, dependency-injection token, and TypeScript helper types needed to register a module synchronously or asynchronously. Use it as the contract between a module definition and the application code that imports and configures that module.

Properties

Property	Type
`ConfigurableModuleClass`	`ConfigurableModuleCls< ModuleOptions, MethodKey, FactoryClassMethodKey, ExtraModuleDefinitionOptions >`
`MODULE_OPTIONS_TOKEN`	`string
`ASYNC_OPTIONS_TYPE`	`ConfigurableModuleAsyncOptions< ModuleOptions, FactoryClassMethodKey > & Partial<ExtraModuleDefinitionOptions>`
`OPTIONS_TYPE`	`ModuleOptions & Partial<ExtraModuleDefinitionOptions>`

Diagram

```
graph LR
  Builder[ConfigurableModuleBuilder] --> Host[ConfigurableModuleHost]
  Host --> ModuleClass[ConfigurableModuleClass]
  Host --> Token[MODULE_OPTIONS_TOKEN]
  Host --> SyncType[OPTIONS_TYPE]
```

```
  Host --> AsyncType[ASYNC_OPTIONS_TYPE]

  ModuleClass --> Register[register(options)]
  ModuleClass --> RegisterAsync[registerAsync(options)]
  Register --> Token
  RegisterAsync --> Token
```

Usage

```typescript
import { Module } from '@nestjs/common';
import {
  ConfigurableModuleBuilder,
  ConfigurableModuleHost,
} from '@nestjs/common';

interface MailerModuleOptions {
  apiKey: string;
  defaultFrom: string;
}

const mailerModuleHost: ConfigurableModuleHost<MailerModuleOptions> =
  new ConfigurableModuleBuilder<MailerModuleOptions>()
    .setClassMethodName('forRoot')
    .build();

export const {
  ConfigurableModuleClass: MailerModule,
  MODULE_OPTIONS_TOKEN: MAILER_OPTIONS,
} = mailerModuleHost;

@Module({})
export class AppModule {
  static register() {
    return {
      module: AppModule,
      imports: [
        MailerModule.forRoot({
          apiKey: process.env.MAILER_API_KEY!,
          defaultFrom: 'no-reply@example.com',
        }),
      ],
    };
  }
}
```

AI Coding Instructions

- Create `ConfigurableModuleHost` instances through `ConfigurableModuleBuilder.build()` rather than constructing the interface manually.
- Use `ConfigurableModuleClass` to expose generated registration methods such as `register`, `forRoot`, or `registerAsync`.
- Inject configuration values with the generated `MODULE_OPTIONS_TOKEN`; do not replace it with a hard-coded string unless required for compatibility.
- Treat `OPTIONS_TYPE` and `ASYNC_OPTIONS_TYPE` as compile-time type helpers, not runtime values.
- Keep the module options type focused on public configuration and use extra module definition options only for module metadata customization.