Other features

This page lists all the other available features that you may find useful.

Global prefix

To ignore a global prefix for routes set through `setGlobalPrefix()`, use `ignoreGlobalPrefix`:

```
const document = SwaggerModule.createDocument(app, options, {
  ignoreGlobalPrefix: true,
});
```

Global parameters

You can add parameter definitions to all routes using `DocumentBuilder`:

```
const options = new DocumentBuilder().addGlobalParameters({
  name: 'tenantId',
  in: 'header',
});
```

Multiple specifications

The `SwaggerModule` provides a way to support multiple specifications. In other words, you can serve different documentation, with different UIs, on different endpoints.

To support multiple specifications, your application must be written with a modular approach. The `createDocument()` method takes a 3rd argument, `extraOptions`, which is an object with a property named `include`. The `include` property takes a value which is an array of modules.

You can setup multiple specifications support as shown below:

```
import { NestFactory } from '@nestjs/core';
import { SwaggerModule, DocumentBuilder } from '@nestjs/swagger';
import { AppModule } from './app.module';
import { CatsModule } from './cats/cats.module';
import { DogsModule } from './dogs/dogs.module';

async function bootstrap() {
  const app = await NestFactory.create(AppModule);

  /**
   * createDocument(application, configurationOptions, extraOptions);
   *
   * createDocument method takes an optional 3rd argument "extraOptions"
   * which is an object with "include" property where you can pass an
Array
   * of Modules that you want to include in that Swagger Specification
   * E.g: CatsModule and DogsModule will have two separate Swagger
Specifications which
   * will be exposed on two different SwaggerUI with two different
endpoints.
   */

  const options = new DocumentBuilder()
    .setTitle('Cats example')
    .setDescription('The cats API description')
    .setVersion('1.0')
    .addTag('cats')
    .build();

  const catDocument = SwaggerModule.createDocument(app, options, {
    include: [CatsModule],
  });
  SwaggerModule.setup('api/cats', app, catDocument);

  const secondOptions = new DocumentBuilder()
    .setTitle('Dogs example')
    .setDescription('The dogs API description')
    .setVersion('1.0')
    .addTag('dogs')
    .build();

  const dogDocument = SwaggerModule.createDocument(app, secondOptions, {
    include: [DogsModule],
  });
  SwaggerModule.setup('api/dogs', app, dogDocument);

  await app.listen(3000);
}
bootstrap();
```

Now you can start your server with the following command:

```
$ npm run start
```

Navigate to `http://localhost:3000/api/cats` to see the Swagger UI for cats:

In turn, `http://localhost:3000/api/dogs` will expose the Swagger UI for dogs: