Controller

October 1, 2026

Controller

Kind: Function

Source: `packages/common/decorators/core/controller.decorator.ts`

Part of: Common

Decorator that marks a class as a Nest controller that can receive inbound requests and produce responses.

An HTTP Controller responds to inbound HTTP Requests and produces HTTP Responses.
It defines a class that provides the context for one or more related route handlers that correspond to HTTP request methods and associated routes for example `GET /api/profile`, `POST /users/resume`

A Microservice Controller responds to requests as well as events, running over a variety of transports (read more here).
It defines a class that provides a context for one or more message or event handlers.

`@Controller()` marks a class as a Nest controller. It establishes the class as the context for related HTTP route handlers or microservice message and event handlers.

Signature

```
function Controller(prefixOrOptions: string | string[] | ControllerOptions): ClassDecorator
```

Parameters

Name	Type
`prefixOrOptions`	`string

Returns: `ClassDecorator`

Diagram

```
graph LR
  Request[Inbound request or event] --> Controller["@Controller() class"]
  Controller --> Handler[Route or message handler]
  Handler --> Response[HTTP response or event result]
```

Usage

```
import { Controller, Get, Post, Body } from '@nestjs/common';

@Controller('users')
export class UsersController {
  @Get()
  findAll() {
    return ['Ada', 'Grace'];
  }

  @Post()
  create(@Body() user: { name: string }) {
    return user;
  }
}
```

AI Coding Instructions

- Apply `@Controller()` to classes that group related route, message, or event handlers.
- Pass a route prefix such as `'users'` when handlers share a common URL path.
- Add method decorators such as `@Get()` and `@Post()` for HTTP endpoints within the controller class.
- Keep request handling in controllers and delegate business logic to injected services.
- Match controller routes with the application's configured global prefix and module registration.