ForbiddenException

August 17, 2026

ForbiddenException

Kind: Class

Source: `packages/common/exceptions/forbidden.exception.ts`

Part of: Common

Defines an HTTP exception for *Forbidden* type errors.

`ForbiddenException` represents an HTTP 403 error, indicating that a request was understood but the authenticated user does not have permission to perform the requested action. It extends the framework's HTTP exception handling system so controllers, guards, and services can signal authorization failures consistently. Use it when access is denied due to insufficient roles, ownership, or policy permissions.

Extends: `HttpException`

Diagram

```
graph LR
  A[Incoming request] --> B[Authorization check]
  B -->|Permission granted| C[Continue request handling]
  B -->|Permission denied| D[ForbiddenException]
  D --> E[HTTP 403 Forbidden response]
```

Usage

```
import { ForbiddenException } from '@nestjs/common';

function updateProject(userId: string, project: { ownerId: string }) {
  if (project.ownerId !== userId) {
    throw new ForbiddenException(
      'You do not have permission to update this project.',
    );
  }
```

```
  return { success: true };
}
```

AI Coding Instructions

- Throw `ForbiddenException` only for authorization failures; use `UnauthorizedException` when the user is not authenticated.
- Perform role, ownership, or policy checks before executing protected mutations or exposing restricted data.
- Provide a clear, safe error message, but do not reveal sensitive permission rules or internal resource details.
- Prefer using guards or centralized authorization policies for reusable access-control logic rather than duplicating checks across controllers.

Used by

2 references from 2 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (2)

- `ParamProperties` — `packages/core/router/router-execution-context.ts`:50
- `ParamsFactory` — `packages/core/helpers/external-context-creator.ts`:28