Catch

August 17, 2026

Catch

Kind: Function

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

Part of: Common

Decorator that marks a class as a Nest exception filter. An exception filter handles exceptions thrown by or not handled by your application code.

The decorated class must implement the `ExceptionFilter` interface.

`Catch()` marks a class as a Nest exception filter, allowing it to intercept exceptions thrown during request processing. The decorated class must implement Nest's `ExceptionFilter` interface and can optionally target one or more specific exception types.

Signature

```
function Catch(exceptions: Array<Type<any> | Abstract<any>>): ClassDecorator
```

Parameters

Name	Type
`exceptions`	`Array<Type

Returns: `ClassDecorator`

Diagram

```
graph LR
  A[Request Handler] -->|throws exception| B[Nest Exception Layer]
  B --> C{Matching @Catch Filter?}
```

```
  C -->|Yes| D[Custom ExceptionFilter.catch]
  D --> E[Create HTTP Response]
  C -->|No| F[Default Nest Exception Handler]
```

Usage

```
import {
  ArgumentsHost,
  Catch,
  ExceptionFilter,
  HttpException,
} from '@nestjs/common';

@Catch(HttpException)
export class HttpExceptionFilter implements ExceptionFilter {
  catch(exception: HttpException, host: ArgumentsHost): void {
    const response = host.switchToHttp().getResponse();
    const request = host.switchToHttp().getRequest();

    response.status(exception.getStatus()).json({
      statusCode: exception.getStatus(),
      path: request.url,
      message: exception.message,
    });
  }
}

// Register globally:
// app.useGlobalFilters(new HttpExceptionFilter());
```

AI Coding Instructions

- Decorate exception filter classes with `@Catch()`; pass exception constructors such as `HttpException` to handle only matching errors.
- Always implement the `ExceptionFilter` interface and provide a `catch(exception, host)` method.
- Use `ArgumentsHost` to select the active transport context, such as `host.switchToHttp()` for HTTP requests.
- Register filters globally with `app.useGlobalFilters()`, at controller scope with `@UseFilters()`, or provide them through Nest dependency injection.
- Avoid swallowing exceptions without sending or delegating an appropriate response for the active transport.

Used by

6 references from 6 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (6)

- `UnauthorizedFilter` — `integration/graphql-code-first/src/common/filters/unauthorized.filter.ts`:4
- `HttpExceptionFilter` — `integration/inspector/src/common/filters/http-exception.filter.ts`:8
- `RequestFilter` — `integration/websockets/src/request.filter.ts`:4
- `HttpExceptionFilter` — `sample/01-cats-app/src/common/filters/http-exception.filter.ts`:8
- `ExceptionFilter` — `sample/03-microservices/src/common/filters/rpc-exception.filter.ts`:5
- `HttpExceptionFilter` — `sample/36-hmr-esm/src/common/filters/http-exception.filter.ts`:8