ArgumentMetadata

August 18, 2026

ArgumentMetadata

Kind: Interface

Source: `packages/common/interfaces/features/pipe-transform.interface.ts`

Part of: Common

Interface describing a pipe implementation's `transform()` method metadata argument.

`ArgumentMetadata` describes the contextual information passed to a pipe's `transform()` method for a controller argument. It identifies where the value came from, optionally provides the runtime type associated with the argument, and includes parameter-specific data such as a route parameter name.

Properties

Property	Type
`type`	`Paramtype`
`metatype`	`Type
`data`	`string

Diagram

```
graph LR
  A[Incoming request value] --> B[Pipe transform(value, metadata)]
  B --> C[ArgumentMetadata]
  C --> D[type: Paramtype]
  C --> E[metatype: Type or undefined]
  C --> F[data: string or undefined]
  B --> G[Transformed controller argument]
```

Usage

```
import { ArgumentMetadata, Injectable, PipeTransform } from '@nestjs/common';

@Injectable()
export class ParseIdPipe implements PipeTransform {
  transform(value: string, metadata: ArgumentMetadata): number | string {
    if (metadata.type === 'param' && metadata.data === 'id') {
      const id = Number(value);

      if (Number.isNaN(id)) {
        throw new Error('The id parameter must be a number.');
      }

      return id;
    }

    return value;
  }
}
```

AI Coding Instructions

- Implement pipes with the `PipeTransform` interface and accept `ArgumentMetadata` as the second `transform()` argument.
- Use `metadata.type` to distinguish values from request bodies, query parameters, route parameters, or custom arguments.
- Treat `metadata.metatype` as optional; it may be `undefined` when no runtime type information is available.
- Treat `metadata.data` as optional and use it for parameter-specific identifiers, such as a route parameter name.
- Avoid relying solely on `metatype` for validation when handling primitive values or interfaces, which may not retain useful runtime metadata.

Used by

15 references from 15 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (15)

- `UserByIdPipe` — `integration/hello-world/src/hello/users/user-by-id.pipe.ts`:4
- `UserByIdPipe` — `integration/hello-world/src/host/users/user-by-id.pipe.ts`:4
- `UserByIdPipe` — `integration/hello-world/src/host-array/users/user-by-id.pipe.ts`:4

- `UserByIdPipe` — `integration/inspector/src/circular-hello/users/user-by-id.pipe.ts`:4
- `ParseIntPipe` — `integration/inspector/src/common/pipes/parse-int.pipe.ts`:8
- `UserByIdPipe` — `integration/scopes/src/circular-hello/users/user-by-id.pipe.ts`:4
- `UserByIdPipe` — `integration/scopes/src/circular-transient/users/user-by-id.pipe.ts`:8
- `UserByIdPipe` — `integration/scopes/src/hello/users/user-by-id.pipe.ts`:9

…and 7 more.