Query

August 18, 2026

Query

Kind: Function

Source: `packages/common/decorators/http/route-params.decorator.ts`

Part of: Common

Route handler parameter decorator. Extracts the `query` property from the `req` object and populates the decorated parameter with the value of `query`. May also apply pipes to the bound query parameter.

For example:

```
async find(@Query('user') user: string)
```

`@Query()` is a route-handler parameter decorator that reads the `query` property from the incoming request object. It binds either the full query object or a named query value to the decorated parameter, and can apply pipes during parameter binding.

Signature

```
function Query(property: string | (Type<PipeTransform> | PipeTransform), pipes: (Type<PipeTrans
```

Parameters

Name	Type
`property`	`string
`pipes`	`(Type

Returns: `ParameterDecorator`

Diagram

```
graph LR
  Request[Incoming request] --> QueryObject[req.query]
  QueryObject --> Decorator["@Query()"]
  Decorator --> HandlerParam[Route handler parameter]
  Pipes[Pipes] --> Decorator
```

Usage

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

@Controller('users')
export class UsersController {
  @Get()
  find(@Query('user') user: string) {
    return { user };
  }

  @Get('search')
  search(@Query() query: Record<string, string>) {
    return query;
  }
}
```

AI Coding Instructions

- Use `@Query('key')` when a handler needs a single query parameter, and `@Query()` when it needs the full query object.
- Keep query parameter names aligned with the request contract, such as `?user=value` for `@Query('user')`.
- Apply pipes through the decorator when query values need validation or transformation.
- Treat query values as request input and account for missing, repeated, or non-string values where applicable.