GrpcStreamCall

August 18, 2026

GrpcStreamCall

Kind: Function

Source: `packages/microservices/decorators/message-pattern.decorator.ts`

Part of: Microservices

`GrpcStreamCall` decorates a controller method as a gRPC streaming call handler. It registers the target service and method as a gRPC message pattern, allowing the microservice runtime to route a bidirectional stream request to the decorated handler.

Signature

```
function GrpcStreamCall(service: string | undefined, method: string): MethodDecorator
```

Parameters

Name	Type
`service`	`string
`method`	`string`

Returns: `MethodDecorator`

Diagram

```
graph LR
  Client[gRPC Client] -->|stream request| Server[gRPC Microservice Server]
  Server -->|match service + method| Metadata[GrpcStreamCall Metadata]
  Metadata --> Handler[Decorated Controller Handler]
  Handler -->|response stream| Client
```

Usage

```
import { Controller } from '@nestjs/common';
import { GrpcStreamCall } from '@nestjs/microservices';
import { Observable, map } from 'rxjs';

interface HeroById {
  id: number;
}

interface Hero {
  id: number;
  name: string;
}

@Controller()
export class HeroesController {
  @GrpcStreamCall('HeroesService', 'FindMany')
  findMany(requests$: Observable<HeroById>): Observable<Hero> {
    return requests$.pipe(
      map(({ id }) => ({
        id,
        name: `Hero ${id}`,
      })),
    );
  }
}
```

AI Coding Instructions

- Use `GrpcStreamCall` for gRPC methods that receive and return streams; use the matching service and RPC method names defined in the `.proto` file.
- Keep the decorated handler signature compatible with streaming RPCs, typically accepting and returning `Observable` values.
- Ensure the controller is registered in the microservice module and that the application is configured with the `Transport.GRPC` transport.
- Avoid changing service or method names independently from the protobuf contract, as routing depends on exact metadata matches.