WsExceptionsHandler

August 17, 2026

WsExceptionsHandler

Kind: Class

Source: `packages/websockets/exceptions/ws-exceptions-handler.ts`

Part of: Websockets

`WsExceptionsHandler` is the WebSocket transport exception dispatcher in NestJS. It selects and invokes matching custom exception filters first, then falls back to the base WebSocket exception filter to serialize and emit an error response to the connected client.

Extends: `BaseWsExceptionFilter`

Methods

Method	Signature	Returns
`handle`	`handle(exception: Error	WsException, host: ArgumentsHost)`
`setCustomFilters`	`setCustomFilters(filters: ExceptionFilterMetadata[])`	`void`
`invokeCustomFilters`	`invokeCustomFilters(exception: T, args: ArgumentsHost)`	`boolean`

Where it refuses work

- `WsExceptionsHandler` stops the work with `InvalidExceptionFilterException` when `!Array.isArray(filters)`.
- `WsExceptionsHandler` stops the work with an early return when `this.invokeCustomFilters(exception, host) || !client.emit`.
- `WsExceptionsHandler` stops the work with an early return when `isEmpty(this.filters)`.

Diagram

```
graph LR
  A[WebSocket handler throws exception] --> B[WsExceptionsHandler.handle]
  B --> C{Matching custom filter?}
  C -->|Yes| D[invokeCustomFilters]
  D --> E[Custom filter catch method]
  C -->|No| F[BaseWsExceptionFilter.catch]
  F --> G[Emit error to WebSocket client]
```

Usage

```
import { ArgumentsHost, Catch } from '@nestjs/common';
import { WsException } from '@nestjs/websockets';
import { WsExceptionsHandler } from '@nestjs/websockets/exceptions/ws-exceptions-handler';

@Catch(WsException)
class WsErrorFilter {
  catch(exception: WsException, host: ArgumentsHost) {
    const client = host.switchToWs().getClient();

    client.emit('error', {
      message: exception.getError(),
      source: 'custom-ws-filter',
    });
  }
}

// This is typically configured internally by NestJS.
const exceptionsHandler = new WsExceptionsHandler();
const filter = new WsErrorFilter();

exceptionsHandler.setCustomFilters([
  {
    exceptionMetatypes: [WsException],
    func: filter.catch.bind(filter),
  },
]);

// `host` is the ArgumentsHost created by NestJS for an incoming WS event.
exceptionsHandler.handle(new WsException('Invalid payload'), host);
```

AI Coding Instructions

- Register custom WebSocket filters before calling `handle()`; matching filters take precedence over the default error behavior.

- Bind filter methods with `filter.catch.bind(filter)` when creating filter metadata so class instance state remains available.
- Return control after a custom filter handles an exception; otherwise the base filter may emit a duplicate error response.
- Use `WsException` for expected client-facing WebSocket failures and reserve unexpected `Error` instances for server-side faults.
- Prefer NestJS gateway-level filter registration (such as `@UseFilters()`) over manually constructing this internal handler in application code.

How it works

`WsExceptionsHandler`

`WsExceptionsHandler` is a public WebSocket exception handler that extends `BaseWsExceptionFilter`. It keeps an initially empty list of custom exception-filter metadata and handles an exception either through a matching custom filter or through the base WebSocket error-emission path. ws-exceptions-handler.ts:12-13

Relationships

- IMPORTS → `ArgumentsHost`
- IMPORTS → `ExceptionFilterMetadata`
- IMPORTS → `selectExceptionFilterMetadata`
- IMPORTS → `isEmpty`
- IMPORTS → `InvalidExceptionFilterException`