# Websockets

September 15, 2026

## Websockets

### What it is responsible for

Websockets manages WebSocket-facing package entry points and validation boundaries evidenced by `BaseWsInstance`, `AbstractWsAdapter`, and `MESSAGE_MAPPING_METADATA`. Work enters through those symbols, so the subsystem turns their named boundaries into its visible integration surface for consumers. The supplied evidence identifies no additional protocol behavior, transport mechanics, or mapping rules. `MESSAGE_MAPPING_METADATA` marks a metadata-named boundary, while `BaseWsInstance` and `AbstractWsAdapter` mark instance- and adapter-named boundaries.

### What it refuses

The subsystem refuses the `!canActivate` condition by raising `WsException`. It refuses filters when `!Array.isArray(filters)` by raising `InvalidExceptionFilterException`. It refuses a socket port when `!Number.isInteger(port)` by raising `InvalidSocketPortException`. The evidence supplies these exception names and triggering expressions, but no literal exception-message text. Therefore the unsoftened developer-facing identifiers available here are `WsException`, `InvalidExceptionFilterException`, and `InvalidSocketPortException`, rather than invented prose messages.

### What it needs, and who needs it

Websockets depends on Common and Core. The listed dependency direction says this package needs both named subsystems; no supplied evidence describes their APIs or a more specific consequence of either absence. Conversely, integration/inspector/src/chat, integration, sample/02-gateways/src/events, sample/16-gateways-ws/src/events, Core, Microservices, Platform Socket.io, and Platform Ws depend on Websockets. Without Websockets, each named dependent loses that

declared dependency; the evidence does not establish which calls, startup paths, or behaviors then fail.

## Notable members

`BaseWsInstance` is a work-entry symbol and a class-named instance boundary. `AbstractWsAdapter` is also a work-entry symbol and an abstract adapter-named boundary. `MESSAGE_MAPPING_METADATA` is the remaining work-entry symbol, a constant whose name explicitly associates it with message mapping metadata. Together they are the only supplied work-entry points, so they carry the clearest documented responsibility within Websockets. The evidence does not state their methods, values, or internal interactions.

87 entities in `packages/websockets`. **4 other subsystems depend on it**, which makes it the 3rd most depended-upon part of this codebase.

## What it is made of

Its 87 entities sit in 36 files under `packages/websockets`: 29 doc comments, 15 classes, 14 constants, 14 functions and 15 more.
`constants.ts` holds 12 of them — more than any other file here.
`WebSocketsController` declares 10 methods, the widest surface here.

## Where work enters

- `Readme.md` — `packages/websockets/Readme.md`:1
- `BaseWsInstance` — `packages/websockets/adapters/ws-adapter.ts`:8
- `AbstractWsAdapter` — `packages/websockets/adapters/ws-adapter.ts`:13
- `MESSAGE_MAPPING_METADATA` — `packages/websockets/constants.ts`:3

## How it refuses and fails

10 of its components record a refusal or a failure handler.
9 of them refuse work outright, under a condition written into the component itself.
Their `catch` blocks handle a failure that already happened in 1 place.

## Boundaries

**4 other subsystems depend on this one** — `Core`, `Microservices`, `Platform Socket.io`, `Platform Ws`. Changing what it exposes changes them.

Those 4 hold 10 edges between them, unevenly: `Platform Ws` reaches in across 5 edges, while 2 of them hold one each. 79 edges leave it against 10 arriving — it reads more of this repository than this repository reads of it. What they reach is narrower than the folder: 7 of its 87 members carry every inbound edge — `SocketModule` (2), `AbstractWsAdapter` (2) and `MessageMappingProperties` (2), plus 4 more. Of the 79 it sends out, 43 go to `Common` — more than to any other.

It depends on `Common`, `Core`, and on nothing else in this repository.

## How this code is named

These conventions cover most of the codebase. Learning them is faster than reading an index —
each one lets you find any member of its family without looking it up.

| Pattern | Where | Count | Examples |
|---|---|---|---|
| `*.interface.ts` | across the repository | 9 | `ws-response.interface.ts`, `nest-gateway.interface.ts`, `on-gateway-init.interface.ts`, `gateway-metadata.interface.ts` |
| `*.decorator.ts` | `packages/websockets/decorators/` | 6 | `ack.decorator.ts`, `message-body.decorator.ts`, `gateway-server.decorator.ts`, `socket-gateway.decorator.ts` |