WsMessageHandler

August 18, 2026

WsMessageHandler

Kind: Interface

Source: `packages/common/interfaces/websockets/web-socket-adapter.interface.ts`

Part of: Common

`WsMessageHandler<T>` describes a WebSocket message handler registered by an adapter or gateway. It pairs an incoming message payload with a callback that produces either an `Observable` or `Promise`, and records whether acknowledgement handling is performed manually.

Properties

Property	Type
`message`	`T`
`callback`	`(...args: any[]) => Observable`
`isAckHandledManually`	`boolean`

Diagram

```
graph LR
  Client[WebSocket Client] --> Message[Incoming Message]
  Message --> Handler[WsMessageHandler<T>]
  Handler --> Payload[message: T]
  Handler --> Callback[callback(...args)]
  Callback --> Result[Observable<any> or Promise<any>]
  Handler --> AckFlag[isAckHandledManually]
  AckFlag --> Ack[WebSocket Acknowledgement]
```

Usage

```ts
import { Observable, of } from 'rxjs';
import { WsMessageHandler } from './web-socket-adapter.interface';

interface ChatMessage {
  roomId: string;
  text: string;
}

const handler: WsMessageHandler<ChatMessage> = {
  message: {
    roomId: 'general',
    text: 'Hello, everyone!',
  },

  callback: (client, payload: ChatMessage): Observable<any> => {
    console.log(`Received message for ${payload.roomId}: ${payload.text}`);

    return of({
      event: 'chat.message.received',
      data: { accepted: true },
    });
  },

  isAckHandledManually: false,
};
```

AI Coding Instructions

- Keep `message` strongly typed with the generic `T` so handler callbacks receive predictable payload data.
- Ensure `callback` returns either an `Observable` or a `Promise`; do not return plain synchronous values.
- Set `isAckHandledManually` to `true` only when the callback explicitly sends or manages the WebSocket acknowledgement.
- Preserve callback argument ordering expected by the surrounding WebSocket adapter or gateway integration.

How it works

`WsMessageHandler<T = string>` is an exported, public WebSocket message-handler shape with a generic message identifier type that defaults to `string`. [web-socket-adapter.interface.ts:3-10]

- `message` is the message identifier and has type `T`. [web-socket-adapter.interface.ts:6-8]
- `callback` accepts any number of arguments and must return either `Observable<any>` or `Promise<any>`. [web-socket-adapter.interface.ts:8]
- `isAckHandledManually` is a required boolean. [web-socket-adapter.interface.ts:9]

`WebSocketAdapter.bindMessageHandlers` accepts an array of these handlers together with a function that transforms callback data into an `Observable`. [web-socket-adapter.interface.ts:23-27] The WebSocket controller binds each callback to the gateway instance and connected client, retains `message` and `isAckHandledManually`, then passes those handler objects to the configured adapter. [web-sockets-controller.ts:172-187]

Handler metadata is discovered only for gateway methods marked with message-mapping metadata; the `message` field comes from that method's message metadata. [gateway-metadata-explorer.ts:38-57] `isAckHandledManually` is set when the method's parameter metadata contains an ACK parameter type. [gateway-metadata-explorer.ts:60-78]

In the Socket.IO adapter, each handler listens on its `message` event, invokes `callback(data, ack)`, and sends non-null transformed results either as a named socket event or through the acknowledgement function. [io-adapter.ts:50-68] When `isAckHandledManually` is `true`, that adapter skips its automatic call to the acknowledgement function. [io-adapter.ts:61-67]

In the `ws` adapter, handlers are indexed by `message`; for a parsed inbound event, the matching callback is invoked with the parsed data and event name, and non-null transformed responses are serialized and sent while the client is open. [ws-adapter.ts:128-151][ws-adapter.ts:154-169] This adapter does not read `isAckHandledManually` in its message-handling code. [ws-adapter.ts:133-166]

The interface declaration itself contains no validation, error handling, or side effects. [web-socket-adapter.interface.ts:6-10]

Used by

1 reference from 1 file. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (1)

- `BaseWsInstance` — `packages/websockets/adapters/ws-adapter.ts`:8