IncomingRequestDeserializer

August 17, 2026

IncomingRequestDeserializer

Kind: Class

Source: `packages/microservices/deserializers/incoming-request.deserializer.ts`

Part of: Microservices

`IncomingRequestDeserializer` converts raw inbound microservice payloads into normalized `IncomingRequest` or `IncomingEvent` domain objects. It determines whether an incoming payload is external and maps its fields to the schema consumed by downstream request and event handling code.

Implements: `ConsumerDeserializer`

Methods

Method	Signature	Returns
`deserialize`	`deserialize(value: any, options: Record<string, any>)`	`IncomingRequest
`isExternal`	`isExternal(value: any)`	`boolean`
`mapToSchema`	`mapToSchema(value: any, options: Record<string, any>)`	`IncomingRequest

Where it refuses work

- `IncomingRequestDeserializer` stops the work with an early return when `!value`.
- `IncomingRequestDeserializer` stops the work with an early return when `!isUndefined((value as IncomingRequest).pattern) || !isUndefined((value as IncomingReques⋯`.
- `IncomingRequestDeserializer` stops the work with an early return when `!options`.

Diagram

```
graph LR
  A[Raw incoming payload] --> B[IncomingRequestDeserializer]
  B --> C{isExternal()}
  C -->|External request| D[mapToSchema()]
  C -->|Internal event| D
  D --> E[IncomingRequest]
  D --> F[IncomingEvent]
  E --> G[Request processing pipeline]
  F --> H[Event processing pipeline]
```

Usage

```
import { IncomingRequestDeserializer } from "./deserializers/incoming-request.deserializer";

const rawPayload = {
  method: "POST",
  url: "/orders",
  headers: {
    "content-type": "application/json",
  },
  body: {
    orderId: "order_123",
  },
};

const deserializer = new IncomingRequestDeserializer(rawPayload);

const incomingMessage = deserializer.deserialize();

if (deserializer.isExternal()) {
  // Handle a normalized external IncomingRequest.
  console.log("Received an external request", incomingMessage);
} else {
  // Handle a normalized internal IncomingEvent.
  console.log("Received an internal event", incomingMessage);
}
```

AI Coding Instructions

- Use `deserialize()` as the public entry point; avoid calling `mapToSchema()` directly unless extending the deserialization flow.
- Keep request/event normalization logic centralized in this class so downstream handlers only consume `IncomingRequest` or `IncomingEvent`.
- Update `isExternal()` when adding new transport metadata or payload types that affect external-versus-internal classification.

- Preserve the expected schema shape in `mapToSchema()` and validate optional headers, body fields, and transport-specific metadata defensively.

Relationships

- IMPORTS → `isUndefined`