TopicMessages

August 18, 2026

TopicMessages

Kind: Interface

Source: `packages/microservices/external/kafka.interface.ts`

Part of: Microservices

`TopicMessages` groups a collection of Kafka `Message` objects under a single topic name. It is used when producing or handling batches of messages that should be sent to the same Kafka topic within the microservices integration layer.

Properties

Property	Type
`topic`	`string`
`messages`	`Message[]`

Diagram

```
graph LR
  T[TopicMessages] --> Topic[topic: string]
  T --> Messages[messages: Message[]]
  Messages --> M1[Kafka Message]
  Messages --> M2[Kafka Message]
```

Usage

```
import type { Message } from 'kafkajs';
import type { TopicMessages } from './kafka.interface';

const messages: Message[] = [
  {
    key: 'user-123',
```

```
    value: JSON.stringify({
      event: 'user.created',
      userId: 'user-123',
    }),
  },
];

const topicMessages: TopicMessages = {
  topic: 'user-events',
  messages,
};

// Example: pass grouped messages to a Kafka producer operation.
await producer.send({
  topic: topicMessages.topic,
  messages: topicMessages.messages,
});
```

AI Coding Instructions

- Always provide a non-empty `topic` that matches the configured Kafka topic naming conventions.
- Populate `messages` with Kafka-compatible `Message` objects, including serialized `value` payloads where required.
- Group only messages targeting the same topic in a single `TopicMessages` object.
- Preserve message keys when ordering, partition affinity, or consumer routing depends on them.
- Validate payload serialization and topic configuration before passing this object to producer APIs.

How it works

`TopicMessages` is an exported TypeScript interface in a file intended to represent KafkaJS package types rather than NestJS logic.
packages/microservices/external/kafka.interface.ts:1-4
packages/microservices/external/kafka.interface.ts:759-762

It describes one topic's set of producer messages:

- `topic` is required and has type `string`.
  packages/microservices/external/kafka.interface.ts:759-761
- `messages` is required and has type `Message[]`.
  packages/microservices/external/kafka.interface.ts:759-762

- Each `Message` requires a `value` of `Buffer`, `string`, or `null`; it can also contain an optional key, partition, headers, and timestamp.
  packages/microservices/external/kafka.interface.ts:121-127

`ProducerBatch.topicMessages` optionally accepts an array of `TopicMessages`, and `sendBatch` accepts that `ProducerBatch` and returns `Promise<RecordMetadata[]>`.
packages/microservices/external/kafka.interface.ts:764-769
packages/microservices/external/kafka.interface.ts:785-788

This interface declares no runtime validation, thrown errors, or side effects.
packages/microservices/external/kafka.interface.ts:759-762