ProducerBatch

August 17, 2026

ProducerBatch

Kind: Interface

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

Part of: Microservices

`ProducerBatch` defines the configuration and payload for sending multiple Kafka topic messages in a single producer operation. It combines delivery guarantees (`acks`), request timing (`timeout`), compression settings, and one or more topic-specific message groups.

Properties

Property	Type
`acks`	`number`
`timeout`	`number`
`compression`	`CompressionTypes`
`topicMessages`	`TopicMessages[]`

Diagram

```
graph LR
  PB[ProducerBatch]
  PB --> A[acks: number]
  PB --> T[timeout: number]
  PB --> C[compression: CompressionTypes]
  PB --> TM[topicMessages: TopicMessages[]]

  TM --> Topic[Kafka topic]
  TM --> Messages[Messages to publish]
```

Usage

```
import { CompressionTypes, ProducerBatch } from '@nestjs/microservices';

const batch: ProducerBatch = {
  acks: -1,
  timeout: 30_000,
  compression: CompressionTypes.GZIP,
  topicMessages: [
    {
      topic: 'orders.created',
      messages: [
        {
          key: 'order-123',
          value: JSON.stringify({ orderId: '123', status: 'created' }),
        },
      ],
    },
    {
      topic: 'audit.events',
      messages: [
        {
          value: JSON.stringify({ action: 'order.created', orderId: '123' }),
        },
      ],
    },
  ],
};

await producer.sendBatch(batch);
```

AI Coding Instructions

- Use `topicMessages` to group messages by Kafka topic when publishing a batch.
- Set `acks` according to the required delivery guarantee; use `-1` when all in-sync replicas must acknowledge writes.
- Keep `timeout` appropriate for broker latency and retry expectations, especially for larger batches.
- Select a `CompressionTypes` value supported by the configured Kafka client and brokers.
- Ensure message keys and values are serialized consistently with consumers before adding them to `topicMessages`.

How it works

`ProducerBatch` is an exported TypeScript interface representing the argument accepted by the Kafka producer batch-send API. It is part of a file intended to represent KafkaJS package types only, rather than NestJS logic. packages/microservices/external/kafka.interface.ts:1-8

All of its properties are optional:

- `acks?: number`
- `timeout?: number`
- `compression?: CompressionTypes`
- `topicMessages?: TopicMessages[]`
  packages/microservices/external/kafka.interface.ts:764-769

`topicMessages`, when present, is an array of objects with a required `topic: string` and required `messages: Message[]`. packages/microservices/external/kafka.interface.ts:759-762 Each `Message` requires a `value` of `Buffer`, `string`, or `null`; it can also include a key, partition, headers, and timestamp. packages/microservices/external/kafka.interface.ts:121-127

The `compression` field accepts the `CompressionTypes` enum: `None` (`0`), `GZIP` (`1`), `Snappy` (`2`), `LZ4` (`3`), or `ZSTD` (`4`). packages/microservices/external/kafka.interface.ts:1129-1135

A `Producer` and a `Transaction` both inherit the `Sender` type, whose `sendBatch(batch: ProducerBatch)` method returns `Promise<RecordMetadata[]>`. packages/microservices/external/kafka.interface.ts:785-788 packages/microservices/external/kafka.interface.ts:798-829 packages/microservices/external/kafka.interface.ts:831-836 Each returned metadata entry includes the topic name, partition, and error code, with optional offset and timestamp-related fields. packages/microservices/external/kafka.interface.ts:748-757

This interface declares no runtime validation, thrown errors, or direct side effects. Its members are type declarations only. packages/microservices/external/kafka.interface.ts:764-769