Kafka by Confluent - Confluent Developer Skills for Building Apache Kafka

Code:	AK-DEV
Length:	3 days
URL:	View Online

In this three-day, hands-on Developer training course, you will learn how to build an application that can publish data to, and subscribe to data from, an Apache Kafka® cluster. You will learn the role of Kafka in the modern data distribution pipeline, Kafka architectural concepts and components, and review Kafka developer APIs.

Skills Gained

In this hands-on course, you will:

- Write Producers and Consumers to send data to and read data from Kafka

- Integrate Kafka with external systems using Kafka Connect

- Write streaming applications with Kafka Streams & ksqlDB

- Integrate a Kafka client application with Confluent Cloud

Who Can Benefit

Application developers and architects who want to write applications that interact with Apache Kafka®. The course treats Java as a first-class citizen, but students will derive value even if Java is not their primary programming language. Python, C# and NodeJS clients will also be used.

Prerequisites

Attendees should be familiar with developing professional apps in Java (preferred), C#, or Python. Additionally, a working knowledge of the Apache Kafka architecture is required for this course, either through:

Course Details

Hands-on Training:
The hands-on lab exercises in the course follow the coherent story of building and upgrading a driver location app. This gives a throughline throughout the course where concepts are applied directly to a working application. Exercises are available in Java, C# and Python.

Exercises include:

- Working with Kafka command line tools

- Producing driver location data to Kafka and consuming that data in real-time

- Refactoring the application to use Avro and Schema Registry

- Creating a Kafka Streams application to do real-time distance aggregation

- Extracting a table from an external database into Kafka using Kafka Connect

- Creating a full event streaming application using kslqDB that enriches driver location data with driver profile data

- Experimenting with semantic partitioning

Course Outline

Fundamentals of Apache Kafka

- Explain the value of a *Distributed Event Streaming Platform*

- Explain how the "log" abstraction enables a distributed event streaming platform

- Brokers, Topics, Partitions, and Segments

- Records (a.k.a. Messages, Events)

- Retention Policies

- Producers, Consumers, and Serialization

- Replication

- Kafka Connect

Producing Messages to Kafka

- Sketch the high level architecture of a Kafka producer

- Illustrate key-based partitioning

- Explain the difference between `acks=0`, `acks=1`, and `acks=all`

- Configure `delivery.timeout.ms` to control retry behavior

- Create a custom `producer.properties` file

- Tune throughput and latency using batching

- Create a producer with Confluent REST Proxy

Consuming Messages from Kafka

- Illustrate how consumer groups and partitions provide scalability and fault tolerance

- Tune consumers to avoid excessive rebalances

- Explain the difference between "range" and "round robin" partition assignment strategies

- Create a custom `consumer.properties` file

- Use the Consumer API to manage offsets

- Tune fetch requests

- Create a consumer with Confluent REST Proxy

Schema Management in Apache Kafka

- Explain what Avro is

- Write Avro messages to Kafka

- Use the Confluent Schema Registry to guide schema evolution

Stream Processing with Kafka Streams

- Compare KStreams to KTables
- Create a Custom `streams.properties` file
- Explain what co-partitioning is and why it is important
- Write an application using the Streams DSL (Domain-Specific Language)

Data Pipelines with Kafka Connect

- Explain the motivation for Kafka Connect
- List commonly used Connectors
- Explain the differences between standalone and distributed mode
- Configure and use Kafka Connect

Stream Processing with Kafka Streams

- Compare KStreams to KTables
- Create a Custom `streams.properties` file
- Explain what co-partitioning is and why it is important
- Write an application using the Streams DSL (Domain-Specific Language)

Data Pipelines with Kafka Connect

- Explain the motivation for Kafka Connect
- List commonly used Connectors
- Explain the differences between standalone and distributed mode
- Configure and use Kafka Connect

Event Streaming Apps with ksqlDB

- Use ksqlDB to filter and transform a stream
- Write a ksqlDB query that joins two streams or a stream and a table
- Write a ksqlDB query that aggregates values per key and time window
- Write Push and Pull queries and explain the differences between them
- Create a Connector with ksqlDB

Design Decisions

- List ways to avoid large message sizes
- Decide when to use ksqlDB vs. Kafka Streams vs. Kafka Connect SMTs
- Explain differences and tradeoffs between processing guarantees
- Address decisions that arise from key-based partitioning
- Authenticate a client app with a secure Kafka cluster

Confluent Cloud

- Explain what "fully-managed" means in the context of Confluent Cloud

- Authenticate a Kafka client to Confluent Cloud

- Do basic operations with the `ccloud` CLI

Schedule (as of 2)

Date	Location

Download Whitepaper: Accelerate Your Modernization Efforts with a Cloud-Native Strategy

Get Your Free Copy Now