

TRAINING COURSE

Confluent Developer Skills for Apache Kafka®

Course Objectives

During 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

Hands-on Training:

The hands-on lab exercises in the course follow the coherent story of building and upgrading a driver location app. Throughout the course 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 ksqlDB that enriches driver location data with driver profile data
- Experimenting with semantic partitioning

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:

- Prior experience, or
- By taking **Confluent Fundamentals for Apache Kafka®**, which can be accessed [here](#).

Participants are required to provide a laptop computer with unobstructed internet access to fully participate in the class.

To evaluate your Kafka knowledge for this course, please complete the self-assessment:

<https://cnfl.io/fundamentals-quiz>

To sign-up for one of our courses, visit us [here](#).

Who Should Attend?

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. C# and Python clients will also be used.

Content

MODULE	DESCRIPTION
Fundamentals of Apache Kafka®	<ul style="list-style-type: none"> • Explain the value of a *Distributed Event Streaming Platform* • Explain how the "log" abstraction enables a distributed event streaming platform • Explain the basic concepts of: <ul style="list-style-type: none"> – Brokers, Topics, Partitions, and Segments – Records (a.k.a. Messages, Events) – Retention Policies – Producers, Consumers, and Serialization – Replication – Kafka Connect
Producing Messages to Kafka	<ul style="list-style-type: none"> • Sketch the high level architecture of a Kafka producer • Illustrate key-based partitioning • Explain the difference between <code>`acks=0`</code>, <code>`acks=1`</code>, and <code>`acks=all`</code> • Configure <code>`delivery.timeout.ms`</code> to control retry behavior • Create a custom <code>`producer.properties`</code> file • Tune throughput and latency using batching • Create a producer with Confluent REST Proxy
Consuming Messages from Kafka	<ul style="list-style-type: none"> • 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 <code>`consumer.properties`</code> file • Use the Consumer API to manage offsets • Tune fetch requests • Create a consumer with Confluent REST Proxy
Schema Management in Apache Kafka®	<ul style="list-style-type: none"> • Describe Kafka schemas and how they work • Use the Confluent Schema Registry to guide schema evolution • Write and read messages using schema-enabled Kafka
Stream Processing with Kafka Streams	<ul style="list-style-type: none"> • Compare KStreams to KTables • Create a Custom <code>`streams.properties`</code> 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	<ul style="list-style-type: none"> • Explain the motivation for Kafka Connect • List commonly used Connectors • Explain the differences between standalone and distributed mode • Configure and use Kafka Connect

(continued)

Content (continued)

MODULE	DESCRIPTION
Event Streaming Apps with ksqlDB	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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

Confluent offers instructor-led courses in both traditional and virtual classroom formats, as well as in a self-paced format available through the Confluent Self-Paced Subscription. Visit confluent.io/training for more information.

Disclaimer: Subscriptions purchased with Training Credits will have an end date matching the Training Credits order.