

ПРАКТИЧЕСКАЯ ПРОГРАММА ОБУЧЕНИЯ

Apache Kafka для Java-разработчиков

От Producer и Consumer API до надёжной обработки событий

Java 21 • Apache Kafka 4 • KRaft • Kafka Streams • Schema Registry • REST Proxy • Docker

Общий объём	С преподавателем	Самостоятельно
72 ак. ч.	24 ак. ч.	48 ак. ч.

Проект после каждого модуля

CarEvents: система событий автомобильного каталога. Java-приложения публикуют изменения автомобилей, потребители обновляют представление данных, Kafka Streams рассчитывает агрегаты. Ошибочные события направляются в отдельный топик.

Обучение лично у основателя ТЕРАКОР

Сергей Валерьевич Герасименко

Практикующий full-stack разработчик, технический директор и основатель ТЕРАКОР и ТЕРАКОР Digital, ведущий преподаватель и практикующий эксперт.

Лицензированный учебный центр • Живые онлайн-занятия

Редакция программы: 28 сентября 2026 года

01 / КОМУ ПОДОЙДЁТ КУРС

Вход и результаты обучения

Для кого

Java-разработчикам, которым нужно интегрировать Kafka и понимать поведение системы при сбоях.

Что нужно знать до старта

Java Core, коллекции, исключения, основы многопоточности, Maven и Docker. Для основного пути Spring не обязателен.

После обучения вы сможете

- Проектировать топики, ключи и consumer groups.
- Настраивать Producer/Consumer и управлять offsets.
- Использовать Admin API и контракты событий.
- Проверять транзакционную обработку и ограничения exactly-once.
- Создавать потоковую агрегацию и диагностировать отставание потребителей.

Как организована практика

Каждый модуль завершается проектным этапом с отдельным результатом и критериями приёмки. Этапы развивают единое приложение; при отдельном входе используется стартовый репозиторий.

Работа с преподавателем включает объяснение, практику, разбор ошибок и проверку результата. Самостоятельные часы включают завершение проектов, тесты и документацию.



02 / УЧЕБНЫЙ ПЛАН

Распределение часов

Модуль	С преподавателем	Самостоятельно
1. Архитектура Kafka и локальный стенд	4	8
2. Producer API и надёжная публикация	3	6
3. Consumer API и управление смещениями	3	6
4. Admin API и сериализация	4	8
5. Транзакции и границы exactly-once	2	4
6. Kafka Streams и агрегаты	2	4
7. Схемы и эволюция контрактов	2	4
8. REST-интеграция	2	4
9. Эксплуатация и защита проекта	2	4
ИТОГО	24	48

Все значения приведены в академических часах. Один академический час - 45 минут. Проекты и защита входят в указанный объём; дополнительные часы сверху не добавляются.

Аттестация: 2 ак. ч. встроенных контрольных проверок (модуль 4 - 0,5; 5 - 0,5; 6 - 0,5; 7 - 0,25; 8 - 0,25) и 2 ак. ч. итогового разбора в модуле 9. Итого 4 ак. ч. внутри 24.

03 / МОДУЛЬ 01

Архитектура Kafka и локальный стенд

4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно

Содержание модуля

- Распределённый лог, события и команды; отличие от очереди задач.
- Kafka 4 и KRaft; broker, controller, topic и partition.
- Ключ и порядок внутри партиции; replication factor и ISR.
- Retention и compaction; delivery semantics.
- Docker Compose и CLI; учебный single-broker стенд и ограничения отказоустойчивости.

Проект после модуля 1

CarEvents Setup: первый поток событий

Развернуть Kafka, создать cars, отправить и прочитайте JSON-события. Сравнить распределение записей с ключом и без него.

Что сдаёт участник

Compose-файл, команды CLI и схема топиков.

Критерии приёмки

- Стенд запускается по README.
- Автор объясняет порядок в партиции и отсутствие глобального порядка.
- Single-broker стенд не выдаётся за отказоустойчивый кластер.

03 / МОДУЛЬ 02

Producer API и надёжная публикация

3 ак. ч. с преподавателем + 6 ак. ч. самостоятельно

Содержание модуля

- ProducerRecord, ключ Long и JSON value; metadata и callback.
- acks, retries, delivery timeout и enable.idempotence.
- Batching, linger и compression: измерение компромиссов.
- Ошибки сериализации и доставки; flush и close.
- Контракт CarEvent: eventId, eventType, occurredAt и данные автомобиля.

Проект после модуля 2

CarEvents Producer: публикация изменений

Написать producer для создания и изменения автомобиля. Фиксировать результат отправки, сохранять стабильный ключ автомобиля.

Что сдаёт участник

Java producer и конфигурация надёжной отправки.

Критерии приёмки

- Ошибка отправки не теряется в callback.
- События одного автомобиля имеют одинаковый ключ.
- Различаются идемпотентность producer и дедупликация бизнес-событий.

03 / МОДУЛЬ 03

Consumer API и управление смещениями

3 ак. ч. с преподавателем + 6 ак. ч. самостоятельно

Содержание модуля

- poll loop, subscription, consumer group и rebalance.
- Ручной commit после обработки; окно повторной доставки.
- max.poll.interval и max.poll.records; обработка длинной задачи.
- Consumer не потокобезопасен; корректное завершение через wakeup.
- Невалидное событие, повтор и dead-letter topic.

Проект после модуля 3

CarEvents Consumer: устойчивое чтение

Обновлять представление автомобилей в учебном хранилище, обрабатывать дубликаты по eventId и проверять перезапуск потребителя.

Что сдаёт участник

Consumer и протокол эксперимента с остановкой.

Критерии приёмки

- Offset не подтверждается до успешной обработки.
- Повтор события не повторяет бизнес-эффект.
- Ошибочная запись не блокирует обработку бесконечно.

03 / МОДУЛЬ 04

Admin API и сериализация

4 ак. ч. с преподавателем + 8 ак. ч. самостоятельно

Содержание модуля

- AdminClient: создание, описание и конфигурация топиков.
- Partitions, consumer groups и просмотр offsets.
- Увеличение числа партиций и влияние на распределение ключей.
- Serializer/Deserializer; Long key и JSON value.
- Валидация payload и обработка неизвестного типа события.

Проект после модуля 4

CarEvents Admin: управление стендом

Написать утилиту создания cars_admin и диагностики топиков. Добавить JSON-сериализатор и отрицательные проверки.

Что сдаёт участник

Admin CLI на Java, сериализаторы и тестовые сообщения.

Критерии приёмки

- Повторный запуск создания корректно обрабатывает существующий топик.
- Битый JSON даёт диагностируемую ошибку.
- Утилита показывает конфигурацию и число партиций.

03 / МОДУЛЬ 05

Транзакции и границы exactly-once

2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно

Содержание модуля

- Idempotent producer и transactional.id.
- initTransactions, begin/commit/abortTransaction.
- Read-process-write и sendOffsetsToTransaction.
- isolation.level=read_committed; aborted records.
- Граница гарантий Kafka: внешняя БД требует отдельного решения.

Проект после модуля 5

CarEvents Transaction: атомарное преобразование

На готовом каркасе прочитать события, записать результат в другой топик и подтвердить offsets в одной Kafka-транзакции.

Что сдаёт участник

Транзакционный процессор и сценарии commit/abort.

Критерии приёмки

- read_committed не видит результат отменённой транзакции.
- При повторном запуске проверено согласованное продвижение offsets.
- Автор не распространяет exactly-once автоматически на внешнюю БД.

03 / МОДУЛЬ 06

Kafka Streams и агрегаты

2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно

Содержание модуля

- Topology, KStream и KTable.
- groupByKey, count и aggregate; Serdes.
- State store и changelog.
- Окна, event time и поздние события на простом примере.
- TopologyTestDriver; joins и Processor API - обзор.

Проект после модуля 6

CarEvents Analytics: статистика по маркам

На подготовленной topology создать агрегат числа событий по марке автомобиля и вывести результат в топик.

Что сдаёт участник

Streams-приложение и детерминированный тест topology.

Критерии приёмки

- Набор входных событий даёт ожидаемый агрегат.
- Ключ агрегата и Serdes заданы явно.
- Различаются число событий и число уникальных автомобилей.

03 / МОДУЛЬ 07

Схемы и эволюция контрактов

2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно

Содержание модуля

- Avro, Protobuf и JSON: задачи форматов.
- Schema Registry как отдельный компонент экосистемы.
- Subject, version и compatibility policy.
- Добавление необязательного поля и несовместимое изменение.
- Совместная работа старого consumer и нового producer.

Проект после модуля 7

CarEvents Schema: безопасное изменение события

На готовом стенде зарегистрировать Avro-схему, добавить поле с default и проверить несовместимое изменение.

Что сдаёт участник

Две версии схемы и результаты проверок совместимости.

Критерии приёмки

- Совместимое изменение принято registry.
- Несовместимое изменение отклонено.
- Consumer читает сообщения старой и новой версии.

03 / МОДУЛЬ 08

REST-интеграция

2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно

Содержание модуля

- Confluent REST Proxy как отдельная HTTP-прослойка.
- HTTP-запросы к топику; media types и формат payload.
- Consumer instance, subscription и чтение через REST.
- Статусы, тайм-ауты и закрытие consumer instance.
- Доступ и ограничения REST-пути по сравнению с Java client.

Проект после модуля 8

CarEvents HTTP: внешняя интеграция

Подготовить curl-сценарии публикации и чтения; проверить невалидное сообщение и очистку экземпляра consumer.

Что сдаёт участник

Набор curl-команд и описание последовательности вызовов.

Критерии приёмки

- Команды воспроизводятся на предоставленном стенде.
- HTTP-ошибка обработана и объяснена.
- Consumer instance удаляется после завершения работы.

03 / МОДУЛЬ 09

Эксплуатация и защита проекта

2 ак. ч. с преподавателем + 4 ак. ч. самостоятельно

Содержание модуля

- Consumer lag, throughput и ошибки клиентов.
- Retention, дисковое место и признаки перегрузки.
- TLS/SASL и ACL: назначение, обзор готовой конфигурации.
- Connect, ksqldb и Spring Kafka: границы применения, демонстрация.
- Защита сквозного сценария и разбор сбоя.

Проект после модуля 9

CarEvents Release: наблюдаемый поток

Объединить producer, consumer и Streams, показать повтор события, остановку потребителя и восстановление обработки.

Что сдаёт участник

Итоговый репозиторий и краткая инструкция диагностики.

Критерии приёмки

- Демонстрация включает штатный сценарий и сбой.
- Lag наблюдается при остановке и сокращается после возобновления.
- Автор объясняет гарантии своего решения и ограничения стенда.

04 / ИТОГОВЫЙ РЕЗУЛЬТАТ

Проект для портфолио

CarEvents: система событий автомобильного каталога. Java-приложения публикуют изменения автомобилей, потребители обновляют представление данных, Kafka Streams рассчитывает агрегаты. Ошибочные события направляются в отдельный топик.

Комплект сдачи

- Репозиторий с исходным кодом и понятной структурой.
- README: требования к окружению, команды сборки и запуска, пример использования.
- Конфигурация и данные для воспроизведения проверок без секретов.
- Результаты тестов или проверочные сценарии для каждого проектного этапа.
- Краткое описание архитектуры, ограничений и принятых решений.

Порядок защиты

Участник запускает приложение, показывает основной сценарий и обработку ошибки, объясняет ключевые решения. Проверка опирается на критерии каждого модуля. Невыполненный критерий фиксируется как конкретная доработка.

Условие завершения

Все обязательные проектные этапы приняты; итоговый сценарий воспроизводится по README. Итоговый проект объединяет ранее выполненные этапы и не является дополнительной работой сверх программы.

Результат обучения

Работающий учебный проект и подтверждённые практикой навыки. Прохождение отдельного курса не означает автоматического присвоения грейда или гарантии трудоустройства.