Faysal Aziz

# Kotlin in the New Era with MVC

## From Mobile Code to Distributed Systems, Building Scalable, Intelligent, and Secure Kotlin Applications - 3 days

FA-0279 · 3 days

Software Development

## Introduction

Modern Kotlin development no longer sits neatly inside the boundaries of a mobile device. Today's applications are expected to scale across services, handle real-time interactions, integrate intelligence, and remain maintainable under constant platform change. Kotlin has grown with that reality. It is now a language used not only for Android interfaces, but also for backend systems, distributed services, and modern application architecture.

Led by an instructor with over 30 years of industry experience, this course is grounded in real industry-demanded content rather than academic theory. The emphasis is on how Kotlin systems are actually structured, optimized, integrated, and evolved in production environments.

## Learning outcomes

By the end of this course, participants will be able to:

- Design scalable Kotlin applications using modern architectural patterns
- Implement and extend MVC with production-grade practices
- Apply controller strategies that support maintainability and testability
- Optimize performance across memory, threading, and app size dimensions
- Navigate platform changes and migrate Android apps from version 12 to 16
- Integrate AI capabilities both on-device and via external APIs

- Build and consume backend microservices using Kotlin and Java
- Understand and apply event-driven architecture using streaming platforms such as Kafka

# Prerequisites

- Good proficiency in Kotlin programming
- Experience with Android development fundamentals
- Understanding of object-oriented programming concepts
- Basic knowledge of REST APIs and networking
- Exposure to Git and version control workflows
- Optional but helpful: prior experience with Java or backend systems

---

# Training outline

## 1. Kotlin Project Structure & Code Architecture

Modularization strategies for large-scale applications

1. Feature-based modules
2. Layered architecture separation

Clean architecture principles in Kotlin

1. Domain, data, and presentation layers
2. Dependency inversion and abstraction boundaries

3. Code organization for scalability and maintainability
4. Dependency injection patterns and frameworks
5. Multi-module Gradle configurations

## 2. MVC Pattern Deep Dive

1. Core principles of MVC in modern Kotlin application development
2. Separation of concerns between model, view, and controller
3. Data flow and request handling in MVC

   State management strategies

   1. UI state handling
   2. Business logic coordination

   Reactive programming integration

   1. Flow and StateFlow in MVC-based applications

6. Handling complex UI interactions

    Testing MVC architecture

        1. Unit testing controllers
        2. Mocking models, services, and data sources

## 3. Controller Best Practices

1. Lifecycle awareness and configuration changes
2. Avoiding memory leaks and improper references
3. State persistence strategies
4. Coroutine scope management within controllers
5. Error handling and retry mechanisms
6. Structuring controllers for scalability
7. Coordinating data flow across multiple controllers

## 4. Performance Tuning and Optimization

    Memory management in Android

        1. Heap usage and garbage collection
        2. Detecting and fixing memory leaks

    Profiling tools and techniques

        1. CPU profiler
        2. Memory profiler
        3. Network profiler

    Application size reduction strategies

        1. Resource shrinking
        2. Code minification

    Threading and concurrency

        1. Kotlin coroutines deep dive
        2. Structured concurrency
        3. Dispatchers and thread pools

5. Avoiding UI thread blocking
6. Performance monitoring in production

## 5. Migrating from Android 12 to Android 16

1. Overview of platform changes across versions

2. Behavioral changes impacting existing apps
3. Permissions and privacy updates
4. Background execution limitations

    Compatibility strategies

        1. Feature flags
        2. Backward compatibility handling

6. Refactoring legacy code for modern APIs
7. Testing across multiple Android versions

## 6. Integrating AI into Kotlin Applications

1. Overview of AI in modern applications

    On-device AI integration

        1. ML model deployment
        2. Performance considerations

    API-based AI integration

        1. Consuming AI services via REST
        2. Handling latency and reliability

    Use cases for AI in mobile apps

        1. Recommendations
        2. Natural language processing

5. Data handling and preprocessing
6. Ethical considerations and data privacy

## 7. Backend Services and Microservices Architecture

    Fundamentals of microservices design

        1. Service decomposition
        2. API gateway patterns

    Communication between services

        1. REST vs gRPC
        2. Event-driven communication

3. Service discovery and orchestration

    Fault tolerance and resilience

        1. Circuit breakers
        2. Retry policies

Cloud deployment models

1. Containerization concepts
2. Scaling strategies

## 8. Backend Development with Kotlin and Java

Kotlin for backend development

1. Frameworks and ecosystem overview

2. Interoperability with Java

Building RESTful APIs

1. Routing and controllers
2. Request/response handling

Database integration

1. ORM and query handling

Asynchronous processing

1. Coroutines in backend systems

6. Logging and monitoring
7. Testing backend services

## 9. Event Streaming and Real-Time Systems

1. Introduction to streaming architectures

Kafka ecosystem fundamentals

1. Topics and partitions
2. Producers and consumers

3. Designing event-driven systems
4. Handling data consistency and ordering
5. Integrating applications with streaming backends
6. Real-time analytics and processing pipelines

---

This course positions Kotlin developers to work across modern application layers, from client-side structure to backend services, AI integration, and real-time systems, using practical approaches shaped by real industry demand.

© 2026 Faysal Aziz · Kuala Lumpur