Course : Kotlin Multiplatform, develop multiplatform applications

Practical course - 3d - 21h00 - Ref. LKD
Price : 1940 CHF E.T.

NEW

On completion of the course, participants will be able to significantly reduce development and maintenance costs, accelerate the time-to-market of their applications, and communicate effectively with native teams when integrating KMP modules into existing projects. This training program is aimed at employees of professional branches covered by the OPCO Atlas.

Teaching objectives

At the end of the training, the participant will be able to:

- Kotlin Multiplateform concepts
- Setting up a development environment
- Design the architecture of a multiplatform mobile application
- Building a fluid, high-performance user interface
- Perform tests to ensure application quality and reliability

Intended audience

For OPCO Atlas members: developers, architects.

Prerequisites

Knowledge of programming languages (Java, C#, C++, JavaScript, Python, etc.).

Practical details

Teaching methods

To optimize the learning experience, e-learning modules can be provided before and after the classroom session or virtual class, at the participant's request.

Course schedule

PARTICIPANTS

For OPCO Atlas members: developers, architects.

PREREQUISITES

Knowledge of programming languages (Java, C#, C++, JavaScript, Python, etc.).

TRAINER QUALIFICATIONS

The experts leading the training are specialists in the covered subjects. They have been approved by our instructional teams for both their professional knowledge and their teaching ability, for each course they teach. They have at least five to ten years of experience in their field and hold (or have held) decision-making positions in companies.

ASSESSMENT TERMS

The trainer evaluates each participant’s academic progress throughout the training using multiple choice, scenarios, hands-on work and more.
Participants also complete a placement test before and after the course to measure the skills they’ve developed.

1 Algorithms, reasoning before design - Pre-training digital learning content

- Introduction to algorithms.
- Basic instructions in pseudo-code.

Digital activities

This online course teaches you how to think before you design a program, and introduces you to the basics of algorithmics. In particular, participants will study the fundamental instructions in pseudo-code.

2 Introduction to Kotlin Multiplatform

- Traditional multiplatform development issues.
- Evolution from KMM to KMP: history and JetBrains vision.
- Comparison with other solutions (React Native, Flutter, Xamarin).
- Use cases and success stories (Netflix, VMware, Philips).

Hands-on work

Group brainstorming: What are the current challenges in mobile development? Comparative analysis of multiplatform solutions in sub-groups.

3 Architecture and fundamental concepts

- Layered architecture: Common, Android, iOS, web.
- Expect/actual mechanism: declarations and implementations.
- Code-sharing strategies: what to share versus what not to share.

Hands-on work

Creation of first simple expect/actual statements.

4 Environment configuration

- System requirements by platform (Windows, macOS, Linux).
- Installing tools: Android Studio, Xcode, KMP plugins.
- Gradle configuration and project structure.

Hands-on work

Guided installation at workstations with collective verification.

5 First KMP Project

- Creating a KMP project with the IntelliJ wizard.
- Folder structure: commonMain, androidMain, iosMain.
- First shared class and its specific implementations.
- Build and run on different platforms.

Hands-on work

Development in pairs: creation of a [Hello KMP " app with display of the current platform. Presentation of achievements and feedback.

TEACHING AIDS AND TECHNICAL RESOURCES

• The main teaching aids and instructional methods used in the training are audiovisual aids, documentation and course material, hands-on application exercises and corrected exercises for practical training courses, case studies and coverage of real cases for training seminars.
• At the end of each course or seminar, ORSYS provides participants with a course evaluation questionnaire that is analysed by our instructional teams.
• A check-in sheet for each half-day of attendance is provided at the end of the training, along with a course completion certificate if the trainee attended the entire session.

TERMS AND DEADLINES

Registration must be completed 24 hours before the start of the training.

ACCESSIBILITY FOR PEOPLE WITH DISABILITIES

Do you need special accessibility accommodations? Contact Mrs. Fosse, Disability Manager, at psh-accueil@orsys.fr to review your request and its feasibility.

6 Dependency management

- Common versus specific dependencies.
- KMP library ecosystem: kotlinx, Ktor, SQLDelight.
- Advanced Gradle configuration for sourceSets.

Hands-on work
Add and test common dependencies (kotlinx-coroutines, kotlinx-serialization).

7 Patterns and best practices

- Package and module organization.
- Recommended patterns: Repository, UseCase, ViewModel.
- Cross-platform error handling with sealed classes.

Hands-on work
Refactoring of the previously created project according to the patterns presented in sub-groups

8 MVVM architecture and Repository Design Pattern

- Clean multi-platform architecture.
- Implementation of the Repository pattern with common interfaces.
- ViewModels shared with StateFlow and coroutines.
- Dependency injection with Koin.

Hands-on work
Development of a user manager with complete Repository Design Pattern.

9 Data management and APIs

- HTTP client with Ktor: configuration and use.
- JSON serialization with kotlinx.serialization.
- Local storage with SQLDelight: setup and queries.
- Cache strategies and synchronization.

Hands-on work
Creation of a REST API client to retrieve weather data. - Implemented local caching with SQLDelight.

10 Status management and responsiveness

- StateFlow and SharedFlow: concepts and differences.
- UI status management: Loading, Success, Error.
- Combination of flow and zip.

Hands-on work
Implementation of a global state manager for the weather application.

11 User interfaces with Compose Multiplatform

- Introduction to Compose Multiplatform: basic concepts.
- UI components: Text, Button, LazyColumn, Card.
- Themes and Material Design management.
- Navigate between screens with Compose Navigation.
- Reactivity with collectAsState.

Hands-on work
Creation of the weather application interface with Compose. Implemented navigation between list and detail screens.

12 Native integration

- Android integration: using the shared module.
- iOS integration: Swift/Kotlin framework and bridge.
- Data transfer between native and shared layers.

Hands-on work
Integration of KMP ViewModels in Android activities and iOS ViewControllers.

13 Management of specific platforms

- Specific APIs: geolocation, camera, notifications.
- System API abstraction strategies.
- Performance and optimization by platform.

Hands-on work
Geolocation added to weather app with specific implementations.

14 KMP test strategy

- Shared unit tests : CommonTest and specific tests.
- Mocking with MockK in a multiplatform context.
- Testing ViewModels and Repositories.
- Integration tests with databases and APIs.

Hands-on work
Writing unit tests for the components developed in the previous days.

15 User interface testing

- Compose tests with ComposeTestRule.
- User interaction testing and status verification.
- Navigation and user flow tests.
- Test strategies for native parts.

Hands-on work
Creation of UI tests for weather application screens.

16 Deployment and CI/CD

- GitHub Actions configuration for KMP.
- Automated build for Android (APK/AAB) and iOS (IPA).
- Deployment on stores: Play Store, App Store.
- Certificate and signature management.

Hands-on work
Setting up a basic CI/CD pipeline with GitHub Actions.

17 Final Project - E-commerce application

- Product catalog with search and filters.
- Persistent shopping cart.
- User authentication.
- Responsive interface and fluid navigation.
- Unit and integration testing.

Hands-on work
Architecture et planification du projet. Développement des fonctionnalités core. Tests et finalisations.

18 Perspectives and "going further"

- KMP ecosystem: new libraries and developments.
- Community and resources: documentation, forums, conferences.
- Corporate adoption strategies: gradual migration, team training.
- Roadmap KMP: Compose Multiplatform Web, Desktop, Wasm.

Storyboarding workshops
Development of a personalized action plan for each participant.

19 Kotlin, the essential basics - Post-training digital learning content

- Introducing Kotlin.
- Kotlin fundamentals.
- Functions.
- Coroutines.
- Object-oriented programming.
- Android development.
- Native development.
- JavaScript development.
- Server-side development.

Digital activities
This online training course presents the Kotlin language, its fundamentals, its various functions, the notion of coroutine, object-oriented programming with a demonstration of developing an application in Android Studio, how to use Kotlin for native or server-side development and how Kotlin can generate JavaScript code.

Dates and locations

REMOTE CLASS
2026 : 10 Mar., 19 May, 6 Oct., 8 Dec.