

**Functional Programming in Kotlin with Arrow**

**Course Number:** MBL-230
**Duration:** 2 days

**Overview**

This Functional Programming in Kotlin with Arrow training course teaches developers how to enhance their functional Kotlin programming skills and get the most out of additional capabilities provided by the [Arrow library](https://arrow-kt.io/).

**Prerequisites**

All attendees must have several years of Kotin development experience. They must have experience applying the standard FP operators (filter, map, flatMap, reduce, etc.) to solve real-world problems.

**Materials**

All attendees receive comprehensive courseware.

Course outline and materials are copyrighted and owned by [Instil Software](https://instil.co/).

**Software Needed on Each Student PC**

* Windows or Mac minimum 8 GB RAM
* Android Studio installed
* Provided lab files from Accelebrate

**Objectives**

* Use advanced operations, such as Traversal and Applicatives
* Use functional types (such as Either, Validated, and State) to improve safety and maintainability.
* Use Optics to work with deeply nested immutable data
* Compose operations efficiently via suspending functions
* Design and implement sample applications, which follow the declarative and immutable FP style
* Implement algorithms using the more advanced FP operators, such as traverse and bimap
* Use effects to produce polymorphic programs that can be executed across multiple environments

**Outline**

* Introduction
* Kotlin FP Fundamentals (Optional)
	+ Working with function references and lambdas
	+ Lambdas with receiver and patterns for DSL’s
	+ Declaring functions as parameters and return types
	+ Understanding Partial Invocation and Currying
	+ Choosing between code blocks and local functions
	+ Common misunderstandings regarding enclosure
* Advanced FP Concepts (Examples from Arrow 0.12)
	+ Programming using Algebraic Data Types in Kotlin
	+ Adding operators to data structures via Typeclasses
	+ Abstracting generic types via Higher Kinded Types
	+ Functional composition and rules for monadic types
	+ Combining different monadic types via Transformers
* Functional Types Supported in Arrow
	+ The Identity type and situations where it is useful
	+ Why the Option and Try types are not needed in Kotlin
	+ Modeling exceptions and cached/default values via Either
	+ Collecting errors via the Validated type and Semigroups
	+ Using the Reader type to build a record across invocations
	+ Using the State / Writer type to pass data between calls
	+ Arrow wrappers to Kotlin collections and NonEmptyList
* Additional Operators Supported by Arrow
	+ Inverting collections of monadic types via traverse
	+ Using Applicatives to handle multiple type instances
	+ Composition in Arrow using suspending functions
	+ Kleisli as an alternative means of composition
	+ Applying fold, bimap, and swap to Monadic Types
* Manipulating Immutable Data with Optics
	+ Problems posed by deep nesting in immutable collections
	+ Advantages and limits of data classes and the copy method
	+ Using Optics to focus on specific fields in nested data
	+ Different forms of Lens in Arrow, and how to create them
	+ Maintaining codebases that use the Optics library heavily
* Building Purely Functional Designs with Effects
	+ Why pure functions are desirable but side effects inevitable
	+ How the IO type can be used to separate pure from impure code
	+ Effects libraries, delimited continuations, and effectful coding
	+ Understanding Polymorphic Programs within functional designs
	+ How Arrow 1.x uses suspending functions instead of an IO type
	+ Emerging patterns for building applications using Effects
* Conclusion