

## Kotlin Programming

### Audience Course Kotlin Programming

The course [Kotlin](#) Programming is intended for developers who want to learn how Kotlin differentiates itself from Java and who want to learn how to program in Kotlin.

### Prerequisites training Kotlin Programming

To participate in this course experience with programming is required. Knowledge of programming in Java is beneficial for the understanding.

### Realization course Kotlin Programming

The theory is treated on the basis of presentations. Illustrative demos are used to clarify the concepts. There is ample opportunity to practice and theory and exercises are interchanged. The course times are from 9.30 to 16.30

### Certification course Kotlin Programming

Participants receive an official certificate Kotlin Programming after successful completion of the course.

Duration: 3 days

Price: € 1999

[Open Schedule](#)


## Content Course Kotlin Programming

In the course Kotlin Programming participants learn to use the object oriented and functional language Kotlin for software development. Kotlin is a modern statically typed language that is fully compatible with Java and runs on the Java Virtual Machine (JVM). Kotlin has modern language features such as type inferences, null safety, coroutines and unchecked exception which are lacking in Java. Moreover Kotlin code is much more compact compared to [Java](#).

### Kotlin Intro

The course Kotlin Programming starts with an overview of a number of important features of Kotlin. Attention is paid to the syntax simplifications in the Kotlin language compared to Java. It is also discussed that Kotlin code can be compiled not only to Java, but also to JavaScript or native code.

### Language Syntax

Next the language syntax is covered with type inference, mutable and immutable variable declarations, if and when expressions, ranges, loops and iterators.

### Classes and Objects

With regard to object oriented programming, class initialization, primary and secondary constructors final and open classes, abstract classes and interfaces are treated. And also attention is paid to data classes in which methods such as equals, toString and hashCode are automatically generated.

### Functions

Kotlin also supports functional programming and part of the program of the course are lambda functions, higher order function, passing functions as parameters and returning functions. Also to extension methods, destructuring declarations, nested functions and extracting parameters with the spread operator are discussed.

### Collections and Generics

Also covered is the Collection Framework in Kotlin that supports mutable and immutable collections and sequences with lazy evaluation. Parameterized types with generics are also covered. And delegation in Kotlin with lazy and observable properties is explained.

### Coroutines

Finally attention is paid to the use of coroutines in Kotlin which can be considered as lightweight threads and which are excellent for asynchronous handling.

## Modules Course Kotlin Programming

<b>Module 1 : Kotlin Intro</b>	<b>Module 2 : Language Syntax</b>	<b>Module 3 : Classes and Objects</b>
What is Kotlin? Variables Type Inference Kotlin Characteristics Null Handling Safe Call Operator Properties Custom Accessors Kotlin Exceptions Kotlin versus Java Interoperability Run as ECMAScript Potential Downsides	Packages and Imports Default Imports Basic Types Boxing Explicit Conversions Characters Arrays If and When Expressions Loops and Iterators Ranges Jumps and Labels Elvis Operator !! Operator	Kotlin Class Initialization Property Settings Inheritance Calling Base Constructors Secondary Constructors Visibility Modifiers Abstract Classes and Interfaces Nested and Inner Classes Data Classes Destructuring Declarations Sealed Classes Kotlin Objects Companion Objects
<b>Module 4 : Functions</b>	<b>Module 5 : Collections</b>	<b>Module 6 : Generics</b>
Function Scope Local Functions Extension Functions Static Resolvment Extension Properties Recursive Functions Kotlin Tail Recursion Higher Order Functions Lambda Expressions Closures Infix Functions Operator Functions Scope Functions	Collection Types Immutable Collections Mutable Collections Collection Hierarchy Iterators Ranges and Progressions Sequences Common Operations Write Operations Transformations Filtering Plus and Minus Grouping	Generic Classes Generic Functions Type Inference PECS Principle out Keyword in Keyword Type Projections Subtype to Supertype Variances Covariance Contravariance Star Projections Generic Constraints
<b>Module 7 : Delegation</b>	<b>Module 8 : Interoperability</b>	<b>Module 9 : Coroutines</b>
Delegation Design Pattern by Keyword Inheritance Alternative Delegated Properties Lazy Properties Property as Input Return Type as Lazy Observable Properties Standard Delegates Properties in Map Local Delegated Properties Delegate Requirements Translation Rules	Calling Java from Kotlin Calling Static Methods Using Java Collection Reserved Words in Kotlin Calling Kotlin in Java Calling Kotlin Functions Calling Extension Functions Using Mutable Collections Immutable Collections Files with JVM Annotation Functions with JVM Annotation Calling Kotlin Class Calling Kotlin Singleton	What are Coroutines? Concurrency Pattern Light-weight Threads Coroutine Scope launch Method Blocking versus non-Blocking Structured Concurrency Scope Builder Global Coroutines Cancelation and Timeout Coroutine Context Channels Asynchronous Flow