+44 7989 401397
info@olsensoft.com

Scala Development (4 days)

Course overview

Scala is an object-oriented language that runs on the Java Virtual Machine. Scala is also a functional language, and combines the features and benefits of OO and functional programming. This course provides a fast-paced introduction to the language for developers with experience in similar languages (e.g. Java, C#, or C++), and then delves deeper into idiomatic uses of Scala in practice.

What you'll learn

- Writing and running Scala programs
- Object orientation in Scala
- Functional programming in Scala
- Using concurrency
- Implementing Domain-Specific Languages
- Best practices and advanced techniques

Prerequisites

- Experience using a contemporary object-oriented language such as Java, C#, or C++

Course details

- Introduction to Scala: Overview of Scala; Installing Scala; Writing a simple Scala program; Using the Scala interpreter
- Core Scala Syntax: Types and variables; Literals; Tuples; Organising code; Importing types; Abstract types and parameterised types
- Operators and Flow Control: Operators and operands; Decision making; Looping; Pattern matching; Enumerations
- Traits: Overview of traits; Using traits as mix-ins; Constructing traits; Traits vs. classes
- Object-Oriented Programming in Scala: Classes and objects; Visibility; Overriding members of classes and traits; Companion objects; Case classes; Object equality
- The Scala Object System: The Predef object; Statics; Sealed class hierarchies; The Scala type hierarchy
- Functional Programming in Scala: Overview of functional programming; Recursion; Function literals and closures; Functional data structures; Pattern matching; Partial functions; Implicits
- Concurrency and Actors: The problems of shared, synchronized state; Actors; Sending messages to actors; The Mailbox; Threading and events

- Domain-Specific Languages: Overview of DSLs; Internal DSLs; External DSLs; Examples
- The Scala Type System: Reflection; Parameterised types; Variance under inheritance; Type bounds; Abstract types; Path-dependent types; Value types
- Application Design: Annotations; Enumerations; Exceptions; Design trait usage; Design patterns