APACHE SCALA AND AKKA FRAMEWORK - 5 days

By Dr. Vishwanath Rao

Day 1

Introduction

- Welcome and Introductions
- Scala Background, History
- Scala Environment
- Scala IDEs / tooling
- Scala Hello World Program

Scala Classes

- Defining classes
- Constructors and Auxiliary Constructors
- Defining Methods
- Invoking methods, parameters and named parameters
- Immutable and Mutable classes

Further Scala Classes

- Class level code
- Singleton Objects
- Companion Classes
- Use of Option and None
- Pattern matching

Case Classes and Value Classes

- How to define Case classes
- Pattern matching with Case classes
- Extractors and Deep Matching
- When to use Case classes
- AnyVal class
- How to define Value classes
- Limitations of Value Classes
- When to use Value Classes

Inheritance

- Inheritance in OO
- Scala Types and Type System

- Abstract Classes
- Overriding methods and fields
- Super key word (extending versus overriding)
- Final Keyword
- Protected
- Polymorphism
- Sealed Classes

Day 2

Traits in Scala

- Introduction to Scala traits
- Traits and Classes / Objects
- Using Traits
- Trait Inheritance
- Trait Abstract Methods

Further Traits

- Marker traits
- Trait Dependencies
- Universal Traits
- Sealed traits
- Enumeration trait
- To Trait or not to Trait

Packages

- What is a Package?
- Nested Packages
- Multiple Packages per File
- Root package references
- Chained Packages
- Importing
- Access Modifiers
- Qualified Access Modifiers
- Package Objects

Collection Classes in Scala

- Arrays
- Tuples
- Lists, Sets
- Maps
- Type Parameterization

Functions In Scala

- Why Functional Programming

- Defining Functions
- Anonymous Functions
- Function Composition (compose and andThen)
- Closures
- Functions and Methods
- Recursion in Scala

Day 3

Higher Order Functions

- Higher Order Function introduction
- Functions as Parameters
- Functions as return results
- Using Types to simplify function definitions
- Partially Applied Functions
- Currying Functions
- Pas by Value v Pass By Name
- Collections and Functional Programming
- The map and filter functions, the flatMap function and foldLeft / foldRight

Other Language Features

- Lazy Vals
- Implicit Conversions
- Implicit Parameters
- Implicit Objects
- Implicit Classes
- Scala Annotations

Exception Handling

- Traditional Exception Handling
- Try-Catch blocks
- Limitations with this approach
- Functional Exception Handling
- Recovering from Exceptions functionally
- Use of Option and Either
- Defining custom exception types

Testing in Scala

- Testing Frameworks in Scala
- ScalaTest
- Functional Test Suites
- Spec Based testing

Day 4

Akka Actors

- What is the need for Actors?
- The Actor Model
- Actor Concepts
- Actor Communications
- Akka Actors
- Akka Actor Application Structure and Naming
- Actor DSL (Domain Specific Language)
- Actor traits
- Actor Incarnation
- Dispatching Actor
- Akka Does Concurrency
- Actors
- Akka Testing
- Systems, Contexts, Paths, and Locations
- Supervision and Death Watch
- Being Stateful
- Routing Messages

Day 5

- Dispatchers and Mailboxes
- Coding in the Future
- Networking with IO
- Going Multi-Node with Remote Actors
- Sharing Data with Agents
- Granular Concurrency with Dataflow
- Patterns for Akka Programming
- Antipatterns for Akka Programming
- Growing Your App with Add-On Modules
- Using Akka from Java
- Now that You're an Akka Coder