1. Lightbend Scala Language - Expert
by Jerzy Muller

bio: Scala programmer since 2011, Scala Trainer since 2014. Dabbles in all kind of technology, mostly JVM-related.

Level: Advanced - knowledge of and practical experience with Scala is assumed as taught in Lightbend Scala Language - Professional
Length: 4 days - 4 hours of classroom time each day, plus breaks (3pm-7pm)
Approach: Hands-on mastery - attendees code through workshop case study and explore how to solve the toughest Scala challenges in their own work
Requirements: Students bring their own laptops with Java 8

next sessions May 29th-30th and June 12th-13th

About

Lightbend Scala Language - Expert
The power of Scala's type system, unleashed - advanced object functional programming, implicits, and more. Leverage rich language features to create well-designed libraries or DSL's, utilizing proven best practices.

Participants
Developers who have experience and proficiency in Scala including topics covered in "Lightbend Scala Language - Professional"
Developers who want to understand advanced features in Scala
Managers who want to gain a deep understanding of functional programming
Benefits
Developers gain knowledge and skills to confidently program in Scala at a high level
Certified Lightbend Trainers share deep insights that drive business results
Advance to the limits of Scala capability!
Outline

- Recap of important basics
- Object-Functional Programming in Depth
    - Recursion and tail-recursion
    - Partial functions and partial function literals
    - Curried methods, partially applied functions
    - Lifting methods into functions
    - Folding
- Mastering the Type System
    - Scala type hierarchy
    - Value classes
    - Type parameters
    - Variance
    - Package objects
    - Lower and upper bounds
    - (Abstract) Type members
    - Singleton Types
    - Type refinements
    - Static duck-typing
    - Path-dependent types
    - Self Types
- Explicitly Implicit

    - Implicit conversions
    - Implicit resolution, scopes and precedence
    - Library extension via implicit (value) classes
    - Implicit parameters
    - Implicit Values
    - Type classes
    - Implicit parameter chaining
    - Type class examples in the Scala core library
    - Type constructors
    - Context bounds
    - Type witnesses
    - Type tags
- Domain Specific Languages (DSLs)
    - DSL Building blocks:
        - By-name parameters
        - Currying
        - Higher-order functions
        - Dot-free operator notation
        - Implicit conversions
    - Phantom types
    - Finite State Machines (FSM)
- Scala Futures and Promises
    - Execution contexts and Thread Pools
    - Creating & working with Futures
    - Futures & Failures - callbacks & recovery
    - Futures, concurrency & parallelism
    - Future.sequence/Future.traverse
    - Creating an already completed Future
    - Dealing with Future[Option[_]] and for comprehensions
    - Futures - Do's and Don'ts
    - Promises
- Custom Scala Collections
    - Uniform return type principle
    - Collection Builders
    - Like traits
    - Type classes for the tricky cases

authorized by Lightbend