The Scala Programming Language Training

Duration: **5 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

Overview

The Scala Programming Language was invented in 2004 in an attempt to bring functional programming to the jvm, and the java development community. It has risen to prominence in recent years as the de-facto language to compliment and replace java in existing teams, and as an advanced and capable language on greenfield projects.

Scala combines advanced techniques long established in the functional programming community with the best practices of object-orientation, allowing developers to pick and chose a style and an approach that suits their team and their problems.

By the end of this course delegates will have a clear understanding of object orientation and functional programming, and in particular, all the major features of scala which enable these styles of programming to be used and combined.

The course is divided in to 15 modules (listed in the overview) delivered over five days with an attention to an interactive Q&A-style approach to teaching and exercise practice.

Prerequisites

Delegates must have prior programming experience in java or a related object-oriented language.

Outline

Day 1

1 Introduction

- Why scala?
- Libraries
- Static typing
- Compositional syntax
- OO
- Functional
- Powerful
- Paradigms
- OO
- Functional
- Running scala
- Anatomy of scala programs
- Try the repl
- Scalac
- Scala interpreter
- Sbt

The Scala Programming Language Training

Duration: **5 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

1 Fundamentals

- Language
- Objects
- Calling methods
- Operators as methods
- Values and variables
- Types
- Basic types
- Boolean
- Numeric types
- Unit
- Strings
- String methods
- Introduction to collections
- Type arguments
- Tuples, Lists, Maps

1 Flow

- Branching, matching & selecting
- Conditionals
- Intro to pattern matching
- Destructuring
- For comprehensions
- Yield vs Unit
- Comprehensions over lists
- Comprehensions over maps
- Comprehensions over ranges
- Option
- Multiple extraction
- Guards
- Ranges
- Let expressions
- While loops

Day 2

2 Methods

- Code blocks
- Methods
- Method bodies
- Returning unit
- Passing arguments
- Variadics
- Def vs val
- Lazy vals
- Lazy arguments
- Recursion

2 Functions

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- What Is Functional Programming?
- Pure Functions
- Functions
- The Function Type
- ‘Function’ vs Method
- Higher Order Functions
- Functions as Data
- Currying
- Aside: Type Aliases

2 Collections

- Review: types
- Collections heirachy
- Creating collections
- Range
- Array & array buffer
- List & list buffer
- Vector
- Maps
- Sets
- Empty
- Idioms
- Pattern matching
- For comprehnsions review
- Zipping
- Traits

Day 3

3 Transformation

- Streams
- Combinators
- .map & .flatMap
- Folding & Reducing
- .exists and .forall
- Option
- For comprehensions
- Foreach & Flatmap
- Pattern matching
- Review
- Casting
- Sequences
- For-yield-match
- Regex
- Guards
- Option

3 OO

- Classes
- Reading class definitions

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Defining classes
- Constructors
- Properties
- Method overloading
- Java inspection of scala classes
- Getters and setters
- Operator methods
- Objects
- The object keyword
- Companions
- The mechanism of pattern matching
- Case classes
- Pattern matching with case classes
- Aside: case objects

3 Inheritance

- Inheritance: a type relation
- Inheritance: class relationship
- Overriding methods
- Overriding tostring
- Calling the parent method
- Abstracts
- Traits
- Trait inheritance
- Composing objects with traits
- The meaning of super
- Traits as mixins
- Self types
- Cake pattern
- Aside: type members

Day 4

4 Design

- Imports
- Package objects
- Algebra
- Algebraic data types: defining
- Algebraic data types
- Example: document store
- Tuples types
- Generics
- Type arguments
- Variance
- Special types
- Option
- Example: user data
- Bad: exceptions
- Try
- Either

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

4 SBTest

- Simple build tool
- Build files
- Creating a project
- Dependencies
- Project structure
- Running tests
- Sbt task runner
- Testing
- Type systems as tests
- Using type systems
- Assertions
- Scala test
- Flatspec
- Featurespec
- Matchers

4 Implicits

- Implicit conversions
- Implicit conversions: def
- Implicit conversions: implicit classes
- Implicit parameters
- Contexts
- Scoping
- Type classes
- Typeclasses with implicit classes
- Without typeclasses
- Typeclasses: generalizing
- Typeclasses: implicit objects
- Typeclasses: multiple instances

Day 5

5 HigherKinds

- Review: Polymorphism
- Review: Typeclasseses: Monoid
- Scalaz
- Sbt for Scalaz
- Simple Typeclass: Equal
- Simple Typeclass: Order
- Simple Typeclass: Implementing Equal
- Kinds
- Higher-kinded Types: Functor
- Generalizing Functor
- Higher-kinded Types: Applicative
- Monads

5 Appendix

- Concurrency

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Libraries