ProTech "Charting the Course ...

protechtraining.com

... to Your Success!"

Scala Fundamentals

Course Summary

Description

Scala is an object-functional programming language that runs on the Java virtual machine. Scala has full support for functional programming, while also supporting traditional object-oriented programming. Scala is also fully interoperable with Java.

Scala provides advanced and powerful capabilities, well beyond what's available in most other JVM languages. Like Java, Scala has compile-time type safety, unlike popular dynamic languages (though Scala is even more type-safe than Java). But Scala also provides type inference, pattern matching, higher-order functions, a powerful and flexible collections library, and advanced support for the concurrent programming that's so necessary to scale today's demanding applications.

There's a sudden explosion of interest in Scala, and this course will help prepare you to take advantage of this intriguing and capable language. Based on Cay Horstmann's excellent book, Scala for the Impatient, our course starts with Scala basics and gradually works its way through more complicated and powerful language capabilities.

Objectives

By the end of this course, students will be able to:

- How to create and build Scala projects using the Scala Build Tool, SBT.
- How Scala's language features compare to Java.
- Scala's rich collections library.
- How Scala language constructs, such as implicits, allow you to enrich existing libraries, even including final classes.
- How classes and object-oriented programming work in Scala.
- How to use higher-order functions (passing functions as parameters to other functions) to simplify and refactor your code.
- How Scala traits differ from Java interfaces, even the more powerful Java 8 interfaces.
- How operator overloading works in Scala.
- Scala's powerful pattern matching syntax.
- Scala's type system.

Topics

- Getting Started
- The Basics
- Control Structures and Functions
- Working with Arrays
- Maps and Tuples
- Classes
- Objects
- Inheritance
- Files and Regular Expressions
- Traits
- Operators
- Collections

Audience

Java or C++ developers interested in diving into Scala development.

Scala Fundamentals

Course Summary (cont'd)

Prerequisites

- A working knowledge of the Java programming language, since the course often compares Scala constructs to Java constructs. Knowledge of C++ is a reasonable substitute for Java knowledge.
- A solid programming background. This course is not for managers or CIOs; it's aimed squarely at developers.
- A desire to dig into a new language and get your hands dirty.
- A familiarity with the Unix or Windows command line is helpful.

Duration

Four days

ProTech "Charting the Course ...

protechtraining.com

... to Your Success!"

Scala Fundamentals

Course Outline

I. Getting Started
- A. Install a Java runtime
- B. Download and Install Scala
- C. Editors

II. The Basics
- A. The Scala Interpreter
- B. Declaring Values and Variables
- C. Lab
- D. Commonly Used Types
- E. Arithmetic and Operator Overloading
- F. Calling Functions and Methods
- G. The apply Method

III. Control Structures and Functions
- A. Conditional Expressions
- B. Statement Termination
- C. Block Expressions and Assignments
- D. Input and Output
- E. Loops
- F. Advanced for Loops and for Comprehensions
- G. Functions
- H. Default and Named Arguments
- I. Variable Arguments
- J. Procedures
- K. Lazy Values
- L. Exceptions

IV. Working with Arrays
- A. Fixedlength Arrays
- B. Variablelength Arrays: Array Buffers
- C. Traversing Arrays and Array Buffers
- D. Transforming Arrays
- E. Common Algorithms
- F. Deciphering Scaladoc
- G. Multidimensional Arrays
- H. Interoperating with Java
- I. Maps and Tuples
- J. Constructing a Map
- K. Accessing Map Values
- L. Updating Map Values
- M. Iterating over Maps
- N. Sorted Maps
- O. Interoperating with Java
- P. Tuples
- Q. Zipping

V. Classes
- A. Simple Classes
- B. Properties (with Getters and Setters)
- C. Properties with only Getters
- D. Objectprivate Fields
- E. Bean Properties
- F. Auxiliary Constructors
- G. Primary Constructor
- H. Nested Classes

VI. Objects
- A. Singletons
- B. Companion Objects
- C. Objects Extending a Class or Trait
- D. The apply Method
- E. Application Objects
- F. Enumerations
- G. Packages and Imports
- H. Packages
- I. Scope Rules
- J. Chained Package Clauses
- K. Topoffile Notation
- L. Package Objects
- M. Package Visibility
- N. Imports
- O. Imports Can Be Anywhere
- P. Renaming and Hiding Members
- Q. Implicit Imports

VII. Inheritance
- A. Extending a Class
- B. Overriding Methods
- C. Type Checks and Casts
- D. Protected Fields and Methods
- E. Superclass Construction
- F. Overriding Fields
- G. Anonymous Subclasses
- H. Abstract Classes
- I. Abstract Fields
- J. Construction Order and Early Definitions
- K. The Scala Inheritance Hierarchy
- L. Object Equality

Scala Fundamentals

Course Outline (cont'd)

VIII. Files and Regular Expressions
- A. Reading Lines Reading Characters
- B. Reading Tokens and Numbers
- C. Reading from URLs and Other Sources
- D. Reading Binary Files
- E. Writing Text Files
- F. Visiting Directories
- G. Serialization
- H. Process Control
- I. Regular Expressions
- J. Regular Expression Groups

IX. Traits
- A. Why No Multiple Inheritance?
- B. Traits as Interfaces
- C. Traits with Concrete Implementations
- D. Objects with Traits
- E. Layered Traits
- F. Overriding Abstract Methods in Traits
- G. Traits for Rich Interfaces
- H. Concrete Fields in Traits
- I. Abstract Fields in Traits
- J. Trait Construction Order
- K. Initializing Trait Fields
- L. Traits Extending Classes
- M. Self Types
- N. What Happens under the Hood

X. Operators
- A. Identifiers
- B. Infix Operators
- C. Unary Operators
- D. Assignment Operators
- E. Precedence
- F. Associativity
- G. The apply and update Methods
- H. Extractors
- I. The unapplySeq Method

XI. Collections
- A. The Main Collections Traits
- B. Mutable and Immutable Collections
- C. Sequences
- D. Lists
- E. Mutable Lists
- F. Sets
- G. Operators for Adding or Removing Elements
- H. Common Methods
- I. Mapping a Function
- J. Reducing, Folding, and Scanning
- K. Zipping
- L. Iterators
- M. Streams
- N. Lazy Views
- O. Interoperability with Java Collections
- P. Threadsafe Collections
- Q. Parallel Collections