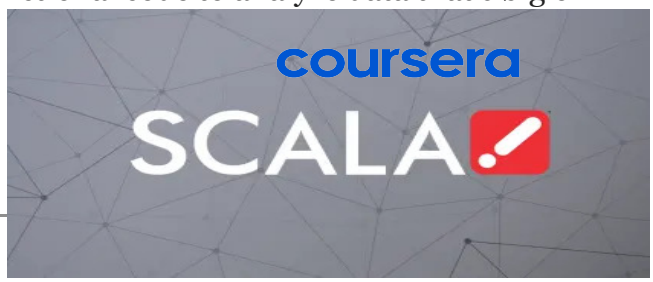
click to go to course

Functional Programming in Scala | Coursera

coursera.org/specializations/scala

Functional Programming in Scala Specialization

Program on a Higher Level. Write elegant functional code to analyze data that's big or small



WHAT YOU WILL LEARN

- Write purely functional programs using recursion, pattern matching, and higher-order functions
- Design immutable data structures
- Write programs that effectively use parallel collections to achieve performance
- Manipulate data with Spark and Scala

SKILLS YOU WILL GAIN

Scala ProgrammingParallel ComputingApache SparkFunctional ProgrammingRecursionImmutable Data TypesHigher-Order FunctionLazinessType ClassReferential TransparencyReactive ProgrammingData Structure

About this Specialization

Discover how to write elegant code that works the first time it is run.

This Specialization provides a hands-on introduction to functional programming using the widespread programming language, Scala. It begins from the basic building blocks of the functional paradigm, first showing how to use these blocks to solve small problems, before building up to combining these concepts to architect larger functional programs. You'll see how the functional paradigm facilitates parallel and distributed programming, and through a series of hands on examples and programming assignments, you'll learn how to analyze data sets small to large; from parallel programming on multicore

architectures, to distributed programming on a cluster using Apache Spark. A final capstone project will allow you to apply the skills you learned by building a large data-intensive application using real-world data.

Applied Learning Project

Learners will build small to medium size Scala applications by applying knowledge and skills including: functional programming, parallel programming, manipulation of large data sets, higher-order functions, property-based testing, functional reactive programming.

How the Specialization Works

Take Courses

A Coursera Specialization is a series of courses that helps you master a skill. To begin, enroll in the Specialization directly, or review its courses and choose the one you'd like to start with. When you subscribe to a course that is part of a Specialization, you're automatically subscribed to the full Specialization. It's okay to complete just one course — you can pause your learning or end your subscription at any time. Visit your learner dashboard to track your course enrollments and your progress.

Hands-on Project

Every Specialization includes a hands-on project. You'll need to successfully finish the project(s) to complete the Specialization and earn your certificate. If the Specialization includes a separate course for the hands-on project, you'll need to finish each of the other courses before you can start it.

Earn a Certificate

When you finish every course and complete the hands-on project, you'll earn a Certificate that you can share with prospective employers and your professional network.

There are 5 Courses in this Specialization

COURSE

1

Functional Programming Principles in Scala

Functional programming is becoming increasingly widespread in industry. This trend is driven by the adoption of Scala as the main programming language for many applications. Scala fuses functional and object-oriented programming in a practical package. It interoperates seamlessly with both Java and Javascript. Scala is the implementation

language of many important frameworks, including Apache Spark, Kafka, and Akka. It provides the core infrastructure for sites such as Twitter, Netflix, Zalando, and also Coursera.

In this course, you will discover the elements of the functional programming style and learn how to apply them usefully in your daily programming tasks, such as modeling business domains or implementing business logic. You will also develop a solid foundation for reasoning about functional programs, by touching upon proofs of invariants and the tracing of execution symbolically. The course is hands-on; most units introduce short programs that serve as illustrations of important concepts and invite you to play with them, modifying and improving them. The course is complemented by a series of programming projects as homework assignments. Recommended background: You should have at least one year of programming experience. Proficiency with Java or C# is ideal, but experience with other languages such as C/C++, Python, Javascript, or Ruby is also sufficient. You should have some familiarity using the command line.

SHOW ALL ABOUT FUNCTIONAL PROGRAMMING PRINCIPLES IN SCALASHOW ALL

COURSE
2

Functional Program Design in Scala

In this course you will learn how to apply the functional programming style in the design of larger Scala applications. You'll get to know important new functional programming concepts, from lazy evaluation to structuring your libraries using monads. We'll work on larger and more involved examples, from state space exploration to random testing to discrete circuit simulators. You'll also learn some best practices on how to write good Scala code in the real world. Finally, you will learn how to leverage the ability of the compiler to infer values from types.

Several parts of this course deal with the question how functional programming interacts with mutable state. We will explore the consequences of combining functions and state. We will also look at purely functional alternatives to mutable state, using infinite data structures or functional reactive programming. Recommended background: You should have at least one year programming experience. Proficiency with Java or C# is ideal, but experience with other languages such as C/C++, Python, Javascript or Ruby is also sufficient. You should have some familiarity with using the command line. This course is intended to be taken after Functional Programming Principles in Scala: https://www.coursera.org/learn/progfun1.

SHOW ALL ABOUT FUNCTIONAL PROGRAM DESIGN IN SCALASHOW ALL

COURSE
3

Parallel programming

With every smartphone and computer now boasting multiple processors, the use of functional ideas to facilitate parallel programming is becoming increasingly widespread. In this course, you'll learn the fundamentals of parallel programming, from task parallelism to data parallelism. In particular, you'll see how many familiar ideas from functional programming map perfectly to to the data parallel paradigm. We'll start the nuts and bolts how to effectively parallelize familiar collections operations, and we'll build up to parallel collections, a production-ready data parallel collections library available in the Scala standard library. Throughout, we'll apply these concepts through several hands-on examples that analyze real-world data, such as popular algorithms like k-means clustering.

Learning Outcomes. By the end of this course you will be able to: - reason about task and data parallel programs, - express common algorithms in a functional style and solve them in parallel, - competently microbenchmark parallel code, - write programs that effectively use parallel collections to achieve performance Recommended background: You should have at least one year programming experience. Proficiency with Java or C# is ideal, but experience with other languages such as C/C++, Python, Javascript or Ruby is also sufficient. You should have some familiarity using the command line. This course is intended to be taken after Functional Program Design in Scala: https://www.coursera.org/learn/progfun2.

SHOW ALL ABOUT PARALLEL PROGRAMMINGSHOW ALL

COURSE

4

Big Data Analysis with Scala and Spark

Manipulating big data distributed over a cluster using functional concepts is rampant in industry, and is arguably one of the first widespread industrial uses of functional ideas. This is evidenced by the popularity of MapReduce and Hadoop, and most recently Apache Spark, a fast, in-memory distributed collections framework written in Scala. In this course, we'll see how the data parallel paradigm can be extended to the distributed case, using Spark throughout. We'll cover Spark's programming model in detail, being careful to understand how and when it differs from familiar programming models, like shared-memory parallel collections or sequential Scala collections. Through hands-on examples in Spark and Scala, we'll learn when important issues related to distribution like latency and network communication should be considered and how they can be addressed effectively for improved performance.

Learning Outcomes. By the end of this course you will be able to: - read data from persistent storage and load it into Apache Spark, - manipulate data with Spark and Scala, - express algorithms for data analysis in a functional style, - recognize how to avoid shuffles and recomputation in Spark, Recommended background: You should have at least one year programming experience. Proficiency with Java or C# is ideal, but

experience with other languages such as C/C++, Python, Javascript or Ruby is also sufficient. You should have some familiarity using the command line. This course is intended to be taken after Parallel Programming: https://www.coursera.org/learn/parprog1.

Offered by

Start Learning Today

- Shareable Specialization and Course Certificates
- Self-Paced Learning Option
- Course Videos & Readings
- Practice Quizzes
- Graded Assignments with Peer Feedback
- Graded Quizzes with Feedback
- Graded Programming Assignments

Shareable on
You can share your Course Certificates in the Certifications section of your LinkedIn profile, on printed resumes, CVs, or other documents.

.

go to course