Scala Introduction

Mikhail Mutcianko, Alexey Shcherbakov

СПБгУ, СП

11 февраля 2021 г.

Intro

Disclaimer

Prior knowledge

You are expected to alrady posess certain knowledge from previous programming courses. We will not explain basic programming concepts like classes, functions, variables etc. . .

Disclaimer

Prior knowledge

You are expected to alrady posess certain knowledge from previous programming courses. We will not explain basic programming concepts like classes, functions, variables etc. . .

- knowledge of OOP and FP

Disclaimer

Prior knowledge

You are expected to alrady posess certain knowledge from previous programming courses. We will not explain basic programming concepts like classes, functions, variables etc. . .

- knowledge of OOP and FP
- knowledge of JVM platform

Disclaimer

Prior knowledge

You are expected to alrady posess certain knowledge from previous programming courses. We will not explain basic programming concepts like classes, functions, variables etc. . .

- knowledge of OOP and FP
- knowledge of JVM platform
- knowledge of algorithms and data structures from CS

What is Scala?

Scala is a general-purpose programming language providing support for functional programming and a strong static type system. Designed to be concise, many of Scala's design decisions aimed to address criticisms of Java.

History

Java Generics

- Java had no generics pre 1.4
- May 1999: Sun proposes to Add Generics to Java, based on GJ
- May 2001: Sun releases prototype for Adding Generics to Java
- January 2003: Generics headed for inclusion in Java 1.5

WTF GJ?

GJ is an extension of the Java programming language that supports generic types

- support for generics
- superset of the Java programming language
- compatible with existing libraries
- efficient translation by erasure

WTF GJ?

GJ is an extension of the Java programming language that supports generic types

Pizza

- move some ideas from FP into the Java space
- take three features from functional programming:
  - generics
  - higher-order functions
  - pattern matching
- work on Pizza has more or less stopped since 2002 †

Pizza

```
1  public final class Main {
2    public int main(String args[]) {
3      System.out.println(
4        new Lines(new DataInputStream(System.in))
5          .takeWhile(nonEmpty)
6          .map(fun(String s) -> int { return Integer.parseInt(s); })
7          .reduceLeft(0, fun(int x, int y) -> int { return x + y; }));
8          while(x == 0) { map.create.newInstance() }
9    }
10 }
```

> "I wanted to design a language that was different from Java, it would always connect to the Java infrastructure — to the JVM and its libraries."
>
> — Martin Odersky

- `Funnel` — first attempt, beautiful but too abstract †
- midway between academic Funnel, and pragmatic but restrictive GJ
- first public release of `Scala` in 2003
- large redesign in early 2006

Academic vs Industrial

Academic

- Haskell
- Agda
- Idris
- ...

Industrial

- Java
- Python
- Scala ?
- ...

Scala benefits

Multiparadigm

Multiparadigm

OOP & Imperative

- traits & objects
- trait mixins
- abstract overrides
- self types
- ...

Multiparadigm

OOP & Imperative

- traits & objects
- trait mixins
- abstract overrides
- self types
- ...

Functional

- type inference
- lambdas
- higher-order functions
- lazy evaluation
- immutability
- pattern matching
- currying
- ADTs

Type Safety

What is type safety?

Type-safety is making use of what we know of our values at compile-time to minimize the consequences of most mistakes.

Type Safety

What is type safety?

Type-safety is making use of what we know of our values at compile-time to minimize the consequences of most mistakes.

- monads
- ADTs
- immutability
- value types
- ...

Libraries and Frameworks

- Akka — actor model implementation

Libraries and Frameworks

- Akka — actor model implementation
- Spark — cluster computing system

Libraries and Frameworks

- Akka — actor model implementation
- Spark — cluster computing system
- Shapeless — type class & dependent types

Libraries and Frameworks

- Akka — actor model implementation
- Spark — cluster computing system
- Shapeless — type class & dependent types
- Cats, Scalaz — principled FP

Libraries and Frameworks

- Akka — actor model implementation
- Spark — cluster computing system
- Shapeless — type class & dependent types
- Cats, Scalaz — principled FP
- Play! — full-scale web

Libraries and Frameworks

- Akka — actor model implementation
- Spark — cluster computing system
- Shapeless — type class & dependent types
- Cats, Scalaz — principled FP
- Play! — full-scale web
- ...

Why Scala?

Banking and Financial Institute majorly concerns for security, stability and sustainability support in the long terms. Scala will satisfy this needs.

- Tinkoff
- JP Morgan
- Credit Suisse
- Morgan Stanley
- ...

Scala responsibility

Compilation times

- using macros
- overuse of implicits
- overuse of libraries with macros and implicits

Codestyle

Infix notation

```
1  def main(args:Array[String]) = {
2    10 PRINT "Enter a number"
3    20 INPUT n
4    30 PRINT "Square root of " % "n is " % SQRT(n)
5    40 END
6    RUN
7  }
```

Codestyle

Types

```
1  trait WRGraph[N <: GNode[N, L], L <: GLink[N]] extends GNode[N, L] { this: N =>
2
3  def combo[AA <: A >: B <% String <% Int : M : Ordering] = ???
4  ...
5  }
```

Codestyle

- implicit hell
- mixing codestyles
- ...

Hands-on

IDEA

Install Scala plugin

SBT

```
$ sbt
sbt:project> ;compile;test:compile
...
sbt:project> run
sbt:project> test
sbt:project> console
```

Practice