Scala: building bridges between programming domains

Germán Ferrari
Scala Meetup Montevideo

https://www.scala-lang.org 90% Search

New on the blog: Rob Norris joins Scala Center's Advisory Board as a new Community Representative

Scala

DOCUMENTATION DOWNLOAD COMMUNITY LIBRARIES CONTRIBUTE BLOG

The Scala Programming Language

Scala combines object-oriented and functional programming in one concise, high-level language. Scala's static types help avoid bugs in complex applications, and its JVM and JavaScript runtimes let you build high-performance systems with easy access to huge ecosystems of libraries.

LEARN MORE

DOWNLOAD

Getting Started

Milestones, nightlies, etc.

All Previous Releases

Scala
2.12.8

API DOCS

Current API Docs

API Docs (other versions)

Scala Documentation

Language Specification

Expressions

```
val label = if (age >= 18) "grownup" else "minor"

val result = tag match {
 case "email" =>
   try getEmail()
   catch handleIOException()
 case "postal" =>
   scanLetter()
}
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 21)

Scoping and nesting

```
def fib(n: Int): Int = {
 def loop(n: Int, a: Int, b: Int): Int = {
   if (n <= 0) a
   else loop(n - 1, b, a + b)
 }
 loop(n, 0, 1)
}
```

Case classes and pattern matching

```scala
sealed abstract class Expr
case class Number(n: Int) extends Expr
case class Plus(lhe: Expr, rhe: Expr) extends Expr

def eval(e: Expr): Int = e match {
 case Number(n)  => n
 case Plus(l, r) => eval(l) + eval(r)
}
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 22)

Function values / higher-order functions

```scala
case class Person(name: String, age: Int)
val people = List(Person("Pedro", 18), Person("María", 20),
                  Person("Juan", 2), Person("José", 17))

def isMinor(p: Person) = p.age < 18
val (minors, adults) = people.partition(isMinor)
// minors = List(Person(Juan,2), Person(José,17))
// adults = List(Person(Pedro,18), Person(María,20)
val infants = minors.filter(_.age <= 3) // List(Person(Juan,2))
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 28)

Immutable collections

```
people.map(_.name) // List(Pedro, María, Juan, José)
people.groupBy(_.age)
// Map(17 -> List(Person(José,17)), 2 -> List(Person(Juan,2)),
//     20 -> List(Person(Pedro,20), Person(María,20)))

val nums = Set(1, 4, 5, 7)
nums.map(_ / 2) // Set(0, 2, 3)

val roman = Map("I" -> 1, "V" -> 5, "X" -> 10)
roman.map { case (k, v) => (v, k) } // Map(1 -> I, 5 -> V, 10 -> X)
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 29)

Parameterized types

```
class List[+A]
class Set[A]
class Function1[-A, +B]

type T1 = List[Number]
type T2 = Set[String]
type T3 = Function1[String, Int]
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 47)

Implicit parameters

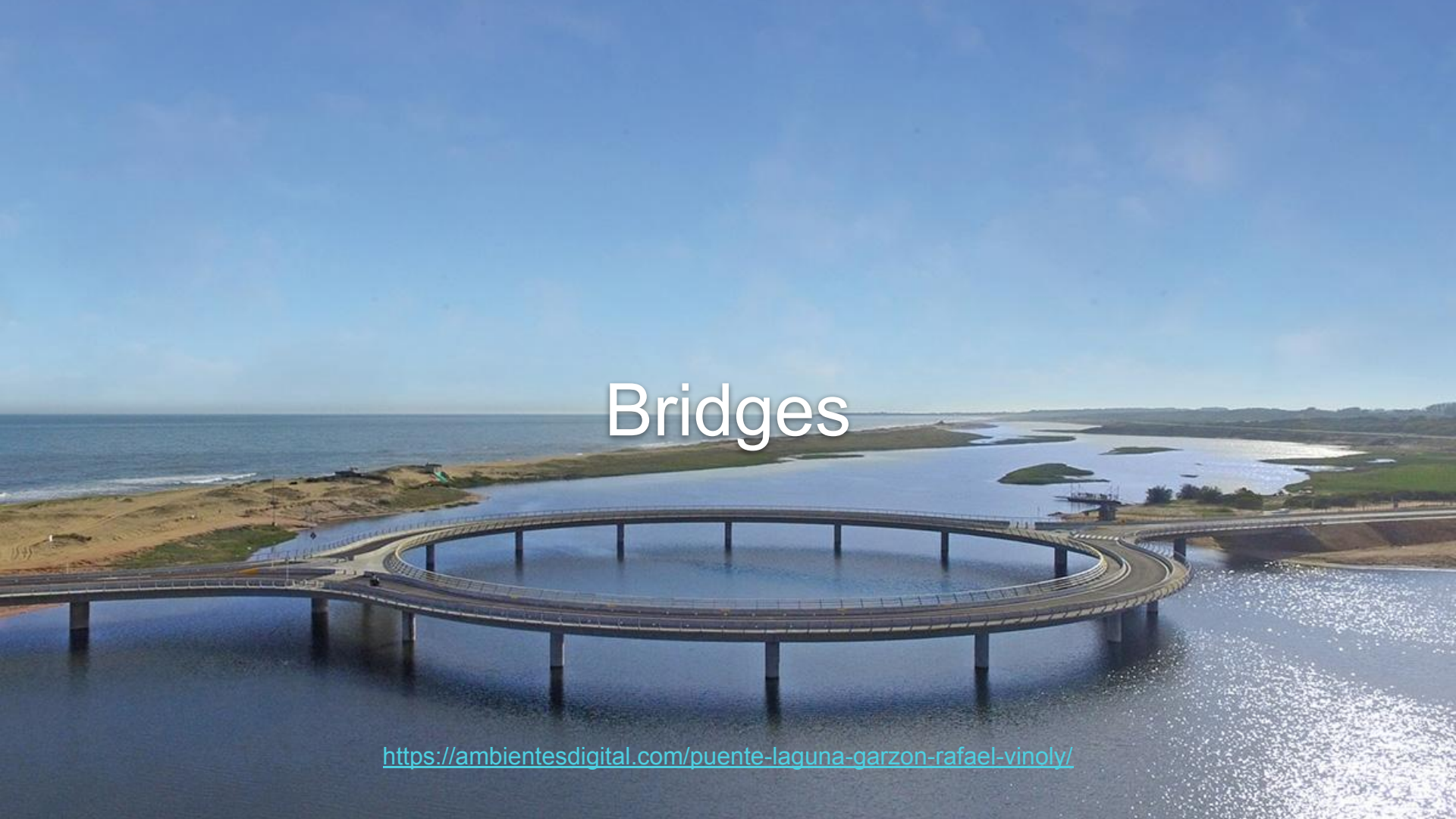
```scala
def min[A](x: A, y: A)(implicit ord: Order[A]): A =
  if (ord.lteqv(x, y)) x else y

min(2, 3) // 2
min("abc", "xyz") // "abc"
min(List(3, 8, 5), List(2, 7, 9)) // List(2, 7, 9)
```

Based on https://www.slideshare.net/Odersky/scala-the-simple-parts (slide 49)

Higher-kinded types

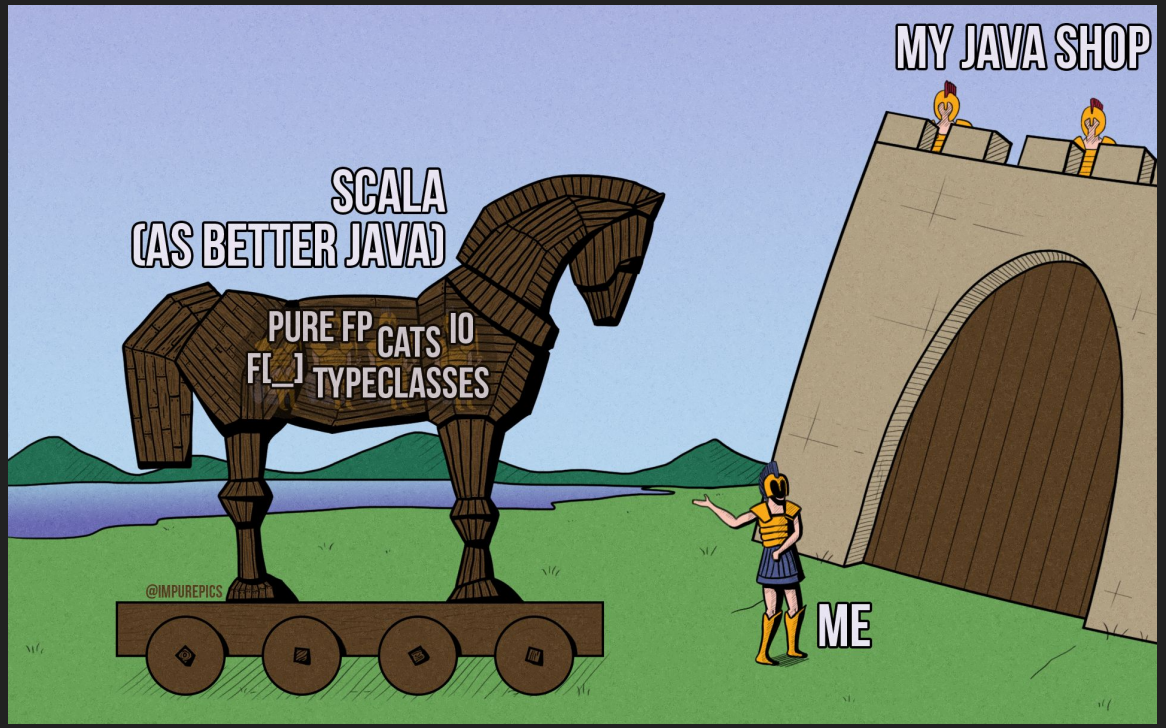
```scala
trait Functor[F[_]]
trait Applicative[F[_]]
trait Monad[F[_]]

type T1 = Functor[Option]
type T2 = Applicative[ValidatedA]
type T3 = Monad[List]
```

Bridges



https://ambientesdigital.com/puente-laguna-garzon-rafael-vinoly/

Bridge to functional programming



https://twitter.com/impurepics/status/1098663070535221253

Bridge to functional programming


```
0111001101
1000110110
1111011001
0001100101
0110001100
```

scodec

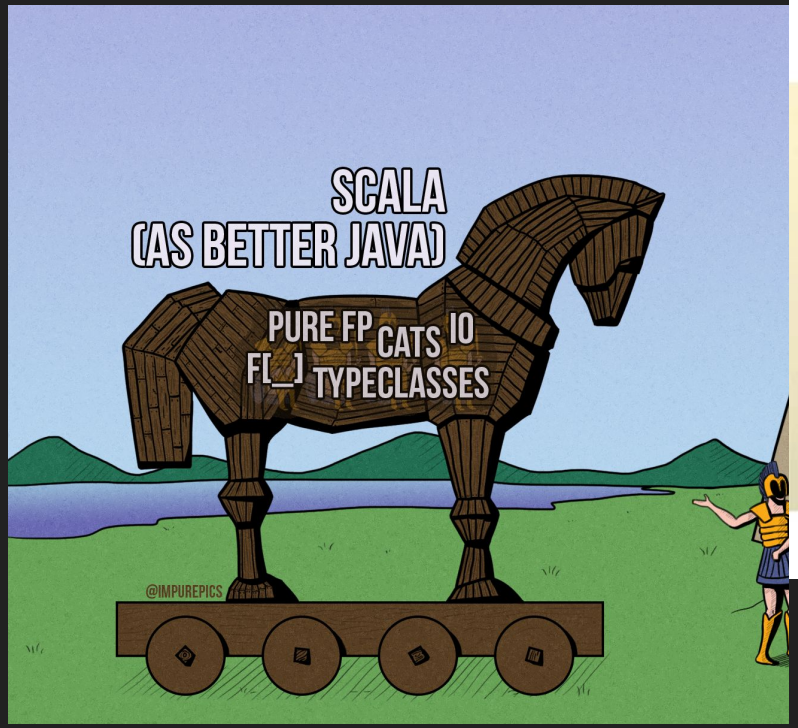
https://twitter.com/impurepics/status/1098663070535221253

Bridge to functional programming

```
object StringSpecification extends Properties("String") {
 property("startsWith") = forAll { (a: String, b: String) =>
   (a + b).startsWith(a)
 }
 // fails when `a` or `b` are the empty `String`
 property("concatenate") = forAll { (a: String, b: String) =>
   (a + b).length > a.length && (a + b).length > b.length
 }
 property("substring") = forAll { (a: String, b: String, c: String) =>
   (a + b + c).substring(a.length, a.length + b.length) == b
 }}
```

https://www.scalacheck.org

Bridge to functional programming



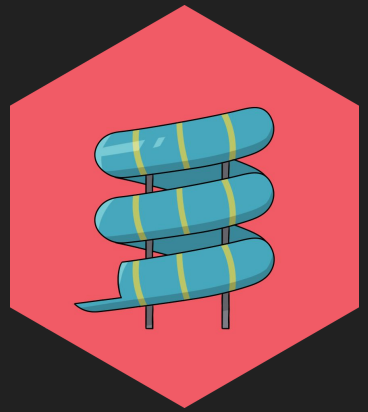
The future of functional programming languages. David MacQueen, Xavier Leroy, Simon Peyton-Jones, Martin Odersky, Don Syme and Phil Wadler - Milner Symposium 2012

https://web.archive.org/web/20130113210808if_/http://podcast.is.ed.ac.uk:8080/Podcasts/informatics/Milner2012/2012-04-16/Milner2012_Panel-FunctProgLang-video.mp4

Bridge to concurrent and distributed programming

Reactive Streams

Monix



FS2

Bridge to concurrent and distributed programming

```
Stream.resource(blockingExecutionContext).flatMap { blockingEC =>
 io.file
   .readAll[IO](Paths.get("fahrenheit.txt"), blockingEC, 4096)
   .through(text.utf8Decode).through(text.lines)
   .filter(s => !s.trim.isEmpty && !s.startsWith("//"))
   .map(line => fahrenheitToCelsius(line.toDouble).toString)
   .intersperse("\n")
   .through(text.utf8Encode)
   .through(io.file.writeAll(Paths.get("celsius.txt"), blockingEC))
}.compile.drain.unsafeRunSync()
```

https://github.com/functional-streams-for-scala/fs2/blob/series/1.0/docs/ReadmeExample.md

Bridge to web and front-end programming

scalajs-react

scalajs-angular

slinky

Bridge

scal

Haoyi's Programming Blog

About Resume Github

Twitter Subscribe Talks

From first principles: Why I bet on Scala.js

Posted 2016-08-10

← Scala Scripting and the 15 Minute Blog Engine

Easy Parsing with Parser Combinators →

Three years ago, I downloaded the nascent Scala.js compiler and tried to use it on a toy project.

Since then, it has matured greatly: the compiler itself is rock-solid. It has a huge ecosystem of libraries. It has a vibrant community, been adopted by some of the largest commercial users of the Scala language, and is playing a key role in shaping evolution of the language. By any measure, it is a success, and I was one of the key people who evangelized it and built foundations for the open-source community and ecosystem that now exists.

However, three years ago in late 2013, when I first got involved in the project, things were different. The compiler was unstable, buggy, slow, and generated incredibly bloated, inefficient Javascript. No ecosystem, no libraries,

http://www.lihaoyi.com/post/FromfirstprinciplesWhyIbetonScalajs.html

Bridge to web and front-end programming

```
// file: conf/routes
GET   /clients/:id          controllers.Clients.show(id: Long)

// file: Clients.scala
package controllers
class Clients() {
 def show(id: Long) = Action {
   Client.findById(id)
         .map { client => Ok(views.html.Clients.display(client)) }
         .getOrElse(NotFound)
 }
}
```

https://www.playframework.com/documentation/2.7.x/ScalaActions

Bridge to big data and data science

Breeze

almond

Spire

Bridge

jupyter spark (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Scala

Run Code

In [5]:
```
val n = rdd.map(_ + 1).sum()
```

sum at cmd4.sc:1

100 / 100

Out[5]:
```
n: Double = 5.00000015E15
```

In [6]:
```
val m = rdd.map(n => (n % 10, n)).reduceByKey(_ + _).collect()
```

map at cmd5.sc:1

100 / 100

collect at cmd5.sc:1

100 / 100

Out[6]:
```
m: Array[(Int, Int)] = Array(
  (0, 1432236160),
  (1, 1342236160),
  (2, 1352236160),
  (3, 1362236160),
  (4, 1372236160),
  (5, 1382236160),
  (6, 1392236160),
  (7, 1402236160),
  (8, 1412236160),
  (9, 1422236160)
)
```

eze

alr

Bridge to big data and data science

mandubianhotep
@mandubian

En respuesta a @ChiefScientist y 2 más

Scala has arguments but JVM is not where DL will happen IMHO... Rust is cool but google chose Swift for its C/C++ replacement for multi-stage compiling... Rust would have been cool but it might be too sharp for most experimentation cases

2 8:03 - 27 feb. 2019

https://twitter.com/mandubian/status/1100713055426699264

Bridge to big data and data science

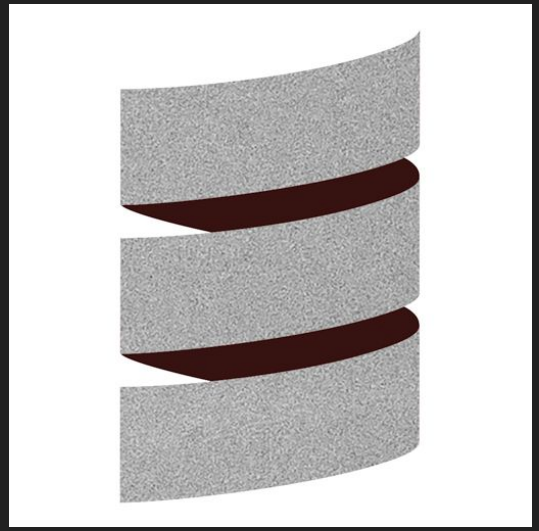
TensorFlow

Tensorflow MLIR

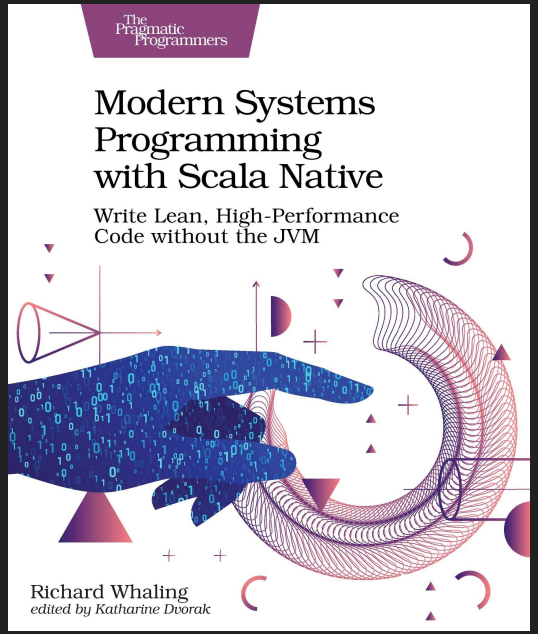
"Multi-Level Intermediate Representation" Compiler Infrastructure

https://github.com/tensorflow/mlir

Bridge to systems programming



Scala Native



http://www.scala-native.org
https://twitter.com/RichardWhaling/status/1090681460955271168

Bridge to systems programming

```
type Vec = CStruct3[Double, Double, Double]

val vec = stackalloc[Vec] // allocate c struct on stack
!vec._1 = 10.0            // initialize fields
!vec._2 = 20.0
!vec._3 = 30.0
length(vec)               // pass by reference
```

http://www.scala-native.org

What is "Scalable"?

- 1st meaning: "Growable"
  - can be molded into new languages by adding libraries (domain specific or general)

  See: "Growing a language" (Guy Steele, 1998)

- 2nd meaning: "Enabling Growth"
  - can be used for small as well as large systems
  - allows for smooth growth from small to large.

Typesafe

"flexible syntax, flexible types, user defined operators, higher order functions, implicits"

Scala is a Unifier

Agile, with lightweight syntax

Object-Oriented → Scala ← Functional

Safe and performant, with strong static tpying

Typesafe

6

https://www.slideshare.net/Odersky/scala-the-simple-parts
https://slideshare.net/Odersky/oscon-keynote-working-hard-to-keep-it-simple


FIN