Experiences with Scala Across the College-Level Curriculum

Konstantin Läufer (presenter) › Loyola U. Chicago
Mark C. Lewis › Trinity U. (San Antonio, TX)
George K. Thiruvathukal › Loyola U. Chicago
Let us know what you think

Please use the Scala Days app to rate sessions.
Motivation

• Colleges and universities (we) produce talent.
• Industry (you) “consumes” talent.
• This could be a match made in heaven!
• Where do we stand with respect to Scala talent?
• We’ll share our side of the story.
• Then you’ll get to share yours!
Context: the Higher Ed landscape

• Pre-college: CS for All
  – CS Principles AP/CS0 - Python
  – CS AP/CS1 - Java
• Community colleges (2y)
• 4y colleges and universities
  – Wide spectrum btw. teaching and research
  – What can they offer?
  – Which are the best match?
Us in the Higher Ed landscape 3

Loyola: private not-for-profit, 16,000 students
- Doctoral Universities: Higher Research Activity ("R2")
- BS and MS in CS, SE, IT, BIOI; BS in Cybersecurity
- Producing about 60 BS and 80 MS per year

Trinity: private non-profit, 2,500 students
- Master's Colleges & Universities: Small Programs
- BS in CS
- Producing 15-30 BS per year

Us: senior faculty with 16-25 years post-PhD experience
Us in the Higher Ed landscape 2

LUC CS ENROLLMENTS

- CNWS-BS
- COMP-BS
- COMPS-BS
- ITEC-BA
- ITEC-BS
- MCSC-BS
- MCSC-BS
- PCSC-BS
- SWDV-BS
- SWEN-BS

Year vs. Enrollment Count
Does this scale (out)?
Using Scala since 2010 across these courses:

- CS1, CS2
- Intermediate OO Development
- Theory (and Practice) of Programming Languages
- Advanced OO Development
- Server Side Software Development
- Web Services Programming
- Independent study/directed research

https://github.com/LoyolaChicagoCode/?q=scala
Portfolio of Scala-based courses

For each course, we will show

• functional and nonfunctional objectives
• role of Scala
• examples
• what worked and what needs improvement
• current status: how regularly offered, using Scala or not
CS1: Intro Programming for CS/SE Majors 3

Functional objectives

• Solve simple symbolic and numeric problems programmatically

Nonfunctional objectives

• Proficiency in a programming language
• Values, constants, variables, and types
• Branching, iteration, control abstraction (functions)

Role of Scala: like a statically typed scripting language
for (i <- 1 to 100) {
  if (i % 3 == 0) print("fizz")
  if (i % 5 == 0) print("buzz")
  if (i % 3 != 0 && i % 5 != 0) print(i)
  println()
}

Can also use pattern matching to set up a decision table but first-year students might find this less clear.
Reflection

+ Scala worked like a statically typed Python without Java’s warts
+ (Lightweight) functions first
+ Unlike Java, supports structural typing, not only nominal
  - IO, readInt and readLine now require an import
  - Various other complications and lost opportunities

Status

• Loyola: Scala: one-time pilot in fall 2015, Java: active/regular
• Trinity: Scala - active as a regular offering
CS2: Intro Data Structures

Functional objectives

• (Mostly linear) data structures
• Searching and sorting algorithms
• A bit of parallelism

Nonfunctional objectives

• Using an OO language to provide abstract data types
• Appreciation of performance and speedup on multicore HW

Role of Scala: largely as a better Java, works well after Scala in CS1

Status: Trinity: Scala active, regular offering; Loyola: Java or C++
Intermediate OO Development 3

Functional objectives

• Custom domain models + recursive behaviors
• Interactive/GUI applications

Nonfunctional objectives

• Design and architectural patterns, separation of concerns
• Event-based programming and background activities
• Testing, including event-based/concurrent systems
• Some experience with Android
• Initial exposure to CI/CD
override def start() = {
    // in ticking clock
    timer = new Timer
    timer.schedule(new TimerTask {
        override def run() = listener.onTick() // fire event
    }, /*initial delay*/ DELAY, /*periodic delay*/ DELAY)
}

private object RUNNING extends StopwatchState { // in state machine
    override def onStartStop() = { actionStop() ; goToState(STOPPED) }
    override def onTick() = { actionInc() ; goToState(RUNNING) }
    override def updateView() = updateUIRuntime()
}

val model: StopwatchModel = new ConcreteStopwatchModelFacade {
    lazy val listener = MainActivity.this // inject Android activity
Intermediate OO Development 1

Reflection

+ Better, more concise Java
+ Very versatile, multi-paradigm
  - Steep learning curve for some
  - Considerable friction with Android development (ProGuard)

Status - Loyola

• Scala: one-time graduate-level online pilot in fall 2014
• Java: active, five sections per year including summer
Theory (and Practice) of Programming Languages 7

Functional objectives

• Efficient Unix-like stdin-stdout pipes
• Custom domain models + recursive behaviors
• Lexers, parsers, interpreters

Nonfunctional objectives

• Understand the programming language design space
• Build and use increasingly powerful abstractions
• Separation of concerns in software design, e.g.
  – structure, content, traversal, processing
Role of Scala

- Thin cake *idiom* for simple dependency injection
- Algebraic data types
- Higher-order functions
- Higher-kindled types
trait TreeBuilder { ... } // SUT-abstraction/contract
trait IO { ... } // provider
trait Main extends App with IO with TreeBuilder { ... } // hybrid
trait FoldTreeBuilder extends TreeBuilder { ... } // SUT-provider

object FMain extends Main with FoldTreeBuilder // DI - no body!

abstract class Spec
    extends WordSpec with TreeBuilder { ... } // hybrid

class FSpec extends Spec with FoldTreeBuilder // DI - no body!
sealed trait ExprF[A]
case class Constant[A](value: Int) extends ExprF[A]
case class Plus[A](left: A, right: A) extends ExprF[A]

object exprFFunctor extends Functor[ExprF] { // scalaz
    case Constant(v) => Constant[B](v)
    case Plus(l, r) => Plus(f(l), f(r))
  }
}

type Expr = Fix[ExprF] // Matryoshka
// behavior = traversal + processing

val evaluate: Algebra[ExprF, Int] = { // processing
  case Constant(c) => c
  case Plus(l, r) => l + r
  ...

assert { (fixtures.complex1 cata evaluate) == -1 }
// cata (generalized fold provided by Fix[F]): traversal

functor.laws[ExprF]
Reflection

+ Multi-paradigm lang: imperative, OO, functional, concurrent
+ Powerful value and type abstractions
  - Steep learning curve for many
  - JVM ignores SIGPIPE => can’t write composable Unix tools

Status - Loyola

• Active, taught in Scala every spring semester since 2013 to 15-25 students (70-80% undergrad)
• U: alternative to Operating Systems, G: elective
Advanced OO Development

Objectives: depends on who teaches it!
• (1) Enterprise computing focus vs.
• (2) Modeling and simulation

Nonfunctional objectives
• (1) Architecture, ORM
• (2) Architecture, concurrency/actors

Role of Scala: (2) powerful abstractions and support for actors

Status - Loyola: in Scala 2x before 2012, then back to Java
object Application extends Controller {
  def guess(value: Long) = Action { implicit request => ...
    val model = previousModel.guess(value.toInt)
    if (model.comparison == 0)
      Ok(views.html.right(guessForm, model))
    else
      ...
};

Objectives: multi-tier, human-centric app design/implementation
Role of Scala: architecture/frameworks, better Java
Loyola: Scala/Play 2x bef. 2012, then back to Java, now suspended*
Trinity: active using Scala and Play  *alt. course: full-stack JS
class ClickcounterServiceActor
    extends Actor with RedisRepositoryProvider {
    ...
    path("increment") {
        post {
            updateIt(_.value + 1, "counter at max, cannot increment")
        }
    ...

Objectives: REST API design and implementation
Role of Scala: architecture/libraries, concurrency, scalability
Status - Loyola: Scala/spray 2x before 2012, then back to Java
How can we make “it” happen? 4

“it” = scaling out the talent production

Some observations:

• We have trouble finding Scala talent ourselves…
• How many instructors fully understand functional programming?
• Need to identify “the hook”: for what courses is Scala a/the compelling choice?
• Can we add Scala support to Processing as an onramp?
Can we convince an entire (education) community that Scala is a compelling choice for each of these areas?

- Full-stack web
- Web front end, preferably isomorphic
- High-scalability server-side
- Mobile
- Systems
- Embedded
- Data analytics
How can we make “it” happen? 2

Can we convince an entire (education) community that Scala is a compelling choice for each of these areas? Lots of competition!

- Full-stack web: JavaScript, Python, Java, Scala/Scala.js
- Web front end, preferably isomorphic: JavaScript, Elm, Scala.js
- High-scalability server-side: Java, Scala
- Mobile: Java, Kotlin, Swift, JavaScript, C#/Xamarin
- Systems: Go, Rust, Scala native, nim?
- Embedded: Go, Rust, Erlang/Elixir, JavaScript, Scala native?, nim?
- Data analytics: lightweight - Python, R; high-performance - Spark
To make it scale out, we need a *multiplier effect*. Some ideas:

- Building out the community
- Working with educators across the whole spectrum
  - Scala workshops at CS edu conferences, e.g., ours @ SIGCSE
  - ambassador program?
  - internships?
- Including the batteries, curate the choices, include exemplars “C_AN” - Comprehensive Archive Network
- Stepping up support for lightweight data analytics (CSV, JSON) to compete with R, Python
Conclusion: We need your input


Scala Days 2017: Survey on Industry Recruiting Needs
Please Remember to rate this session

Thank you!