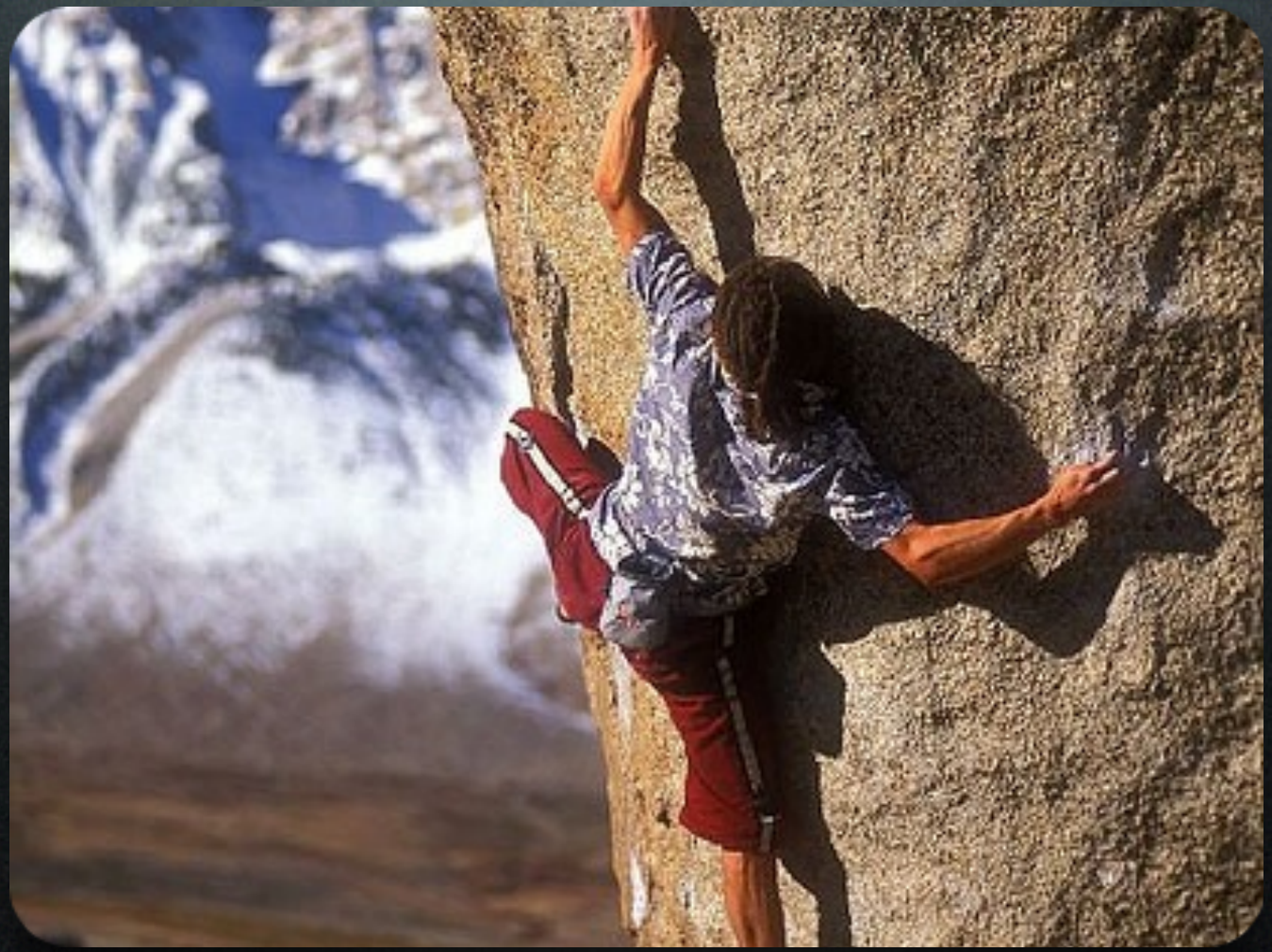
Scala e Programação Funcional

Michael Schuenck dos Santos, MSc.
@michaelss

Eu

- Bacharel em Sistemas de Informação pelo CEULP/ULBRA e ex-professor
- Mestre em Sistemas e Computação pela UFRN e ex-professor
- Analista de Sistemas no TRE-TO



Scalable **La**nguage

Por que meu interesse em Scala?

Twitter

A Infra e a JVM

Paradigma Funcional

História

Martin Odersky
Prof. na Univ. de Lausanne (Suíça)

Criador de Generics e do compilador Java

2001 — Início do projeto de Scala

2003 — Primeiro release público, para a JVM

2004 — Liberada para .NET

2006 — Redesign: versão 2.0

2011 — Martin Odersky funda a Typesafe, que logo recebe um investimento de US$ 3 milhões

Performance

O Google fez um benchmark entre C++, Java, Scala e Go

Benchmark	Virt	Real	Factor Virt	Factor Real
C++ Opt	184m	163m	1.0	1.0
C++ Dbg	474m	452m	2.6-3.0	2.8
Java	1109m	617m	6.0	3.7
Scala	1111m	293m	6.0	1.8
Go	16.2g	501m	90	3.1

Uso de memória

Performance

Benchmark	Time [sec]	Factor
C++ Opt	23	1.0x
C++ Dbg	197	8.6x
Java 64-bit	134	5.8x
Java 32-bit	290	12.6x
Java 32-bit GC*	106	4.6x
Java 32-bit SPEC GC	89	3.7x
Scala	82	3.6x
Scala low-level*	67	2.9x
Scala low-level GC*	58	2.5x
Go 6g	161	7.0x
Go Pro*	126	5.5x

Tempo

Performance

Benchmark	wc -l	Factor
C++ Dbg/Opt	850	1.3x
Java	1068	1.6x
Java Pro	1240	1.9x
Scala	658	1.0x
Scala Pro	297	0.5x
Go	902	1.4x
Go Pro	786	1.2x

Linhas de código

Quem está usando

theguardian

twitter

tumblr.

SONY PICTURES imageworks

foursquare

nature.com

LinkedIn

SIEMENS

Características

Uso de {chaves}

Muitas vezes dispensáveis

; só com mais de uma expressão por linha

Concisão

```
class MyClass {

    private int index;
    private String name;

    public MyClass(int index, String name) {
        this.index = index;
        this.name = name;
    }
}
```

```
class MyClass(index: Int, name: String)
```

Alto Nível

```
for (int i = 0; i < name.length(); ++i) {
    if (Character.isUpperCase(name.charAt(i))) {
        nameHasUpperCase = true;
        break;
    }
}
```

```
val nameHasUpperCase = name.exists(_.isUpper)
```

Tipagem Estática

- Erros em tempo de compilação
- Refatorações seguras
- Scala: inferência de tipos


```
var a = "Palmas"

if (b == 0) a = 10

println(a * 2)
```

Documentação, Clareza e Previsibilidade

Tipagem

Estática

C#

C++

Dinâmica

python

Ruby

JS

Funcional

Paradigma

OO

C#

Smalltalk

Ruby

Funcional

Haskell

Scheme

Lisp

F#

Ruby

OO(Scala) > OO(Java)

Não se utiliza os tipos primitivos

~~int~~
~~double~~
~~float~~
~~boolean~~
~~byte~~
~~short~~

Int
Double
Float
Boolean
Byte
Short

Operadores unários e binários são métodos

```
1 + 2

==

1.+(2)
```

Estrutura

Parâmetros

Retorno: obrigatório em funções recursivas

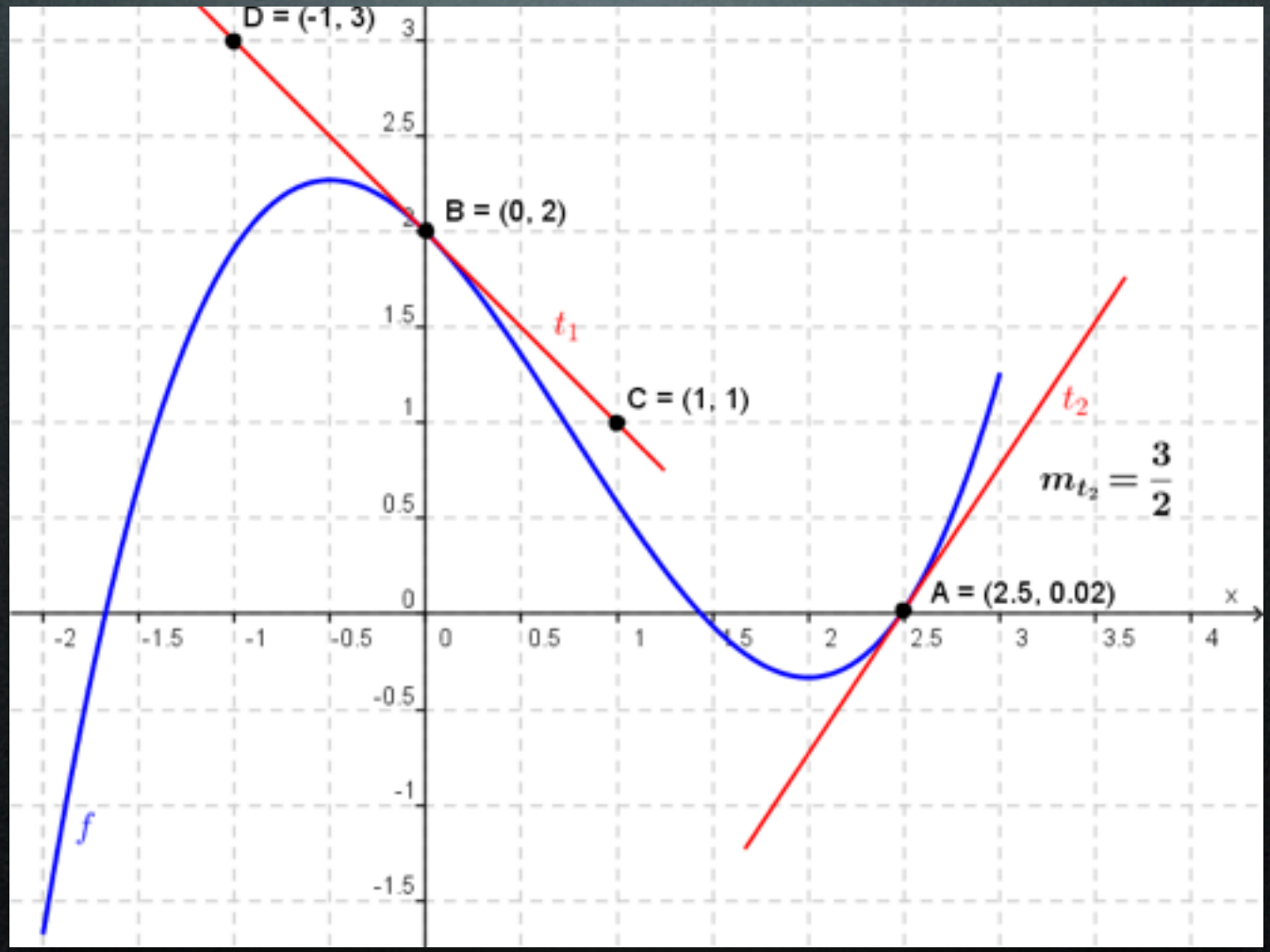
```
def max(x: Int, y: Int = 0): Int = {
  if (x > y)
    x
  else
    y
}
```

Argumento padrão

Dispensável se retorno Unit

return desnecessário

obrigatórias p/ mais de uma linha

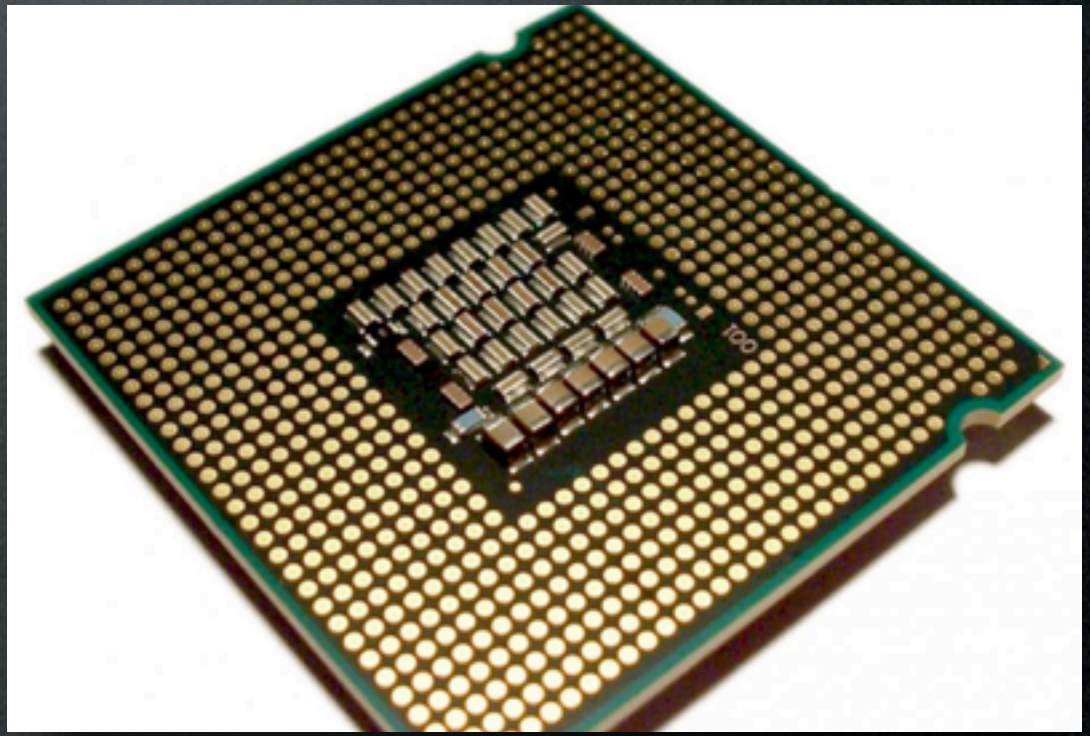


Paradigma Funcional

Interesse crescente:

Resolver o problema de multicores

Lei de Moore



OO	Funcional
Composição de objetos	Composição de funções
Alteração de estado	Ausência de efeitos colaterais
Algoritmos iterativos	Algoritmos recursivos

Imutabilidade

```
val x = "Olá"
x = "Oi" // Erro
```

```
var y = "Olá"
y = "Oi" // Ok
```

Funções de Ordem Superior

Podem ser passadas e retornadas

```
def op(x: Int, y: Int, f: (Int, Int) => Int): Int =
    f(x, y)
```

```
def soma(a: Int, b: Int) = a + b
```

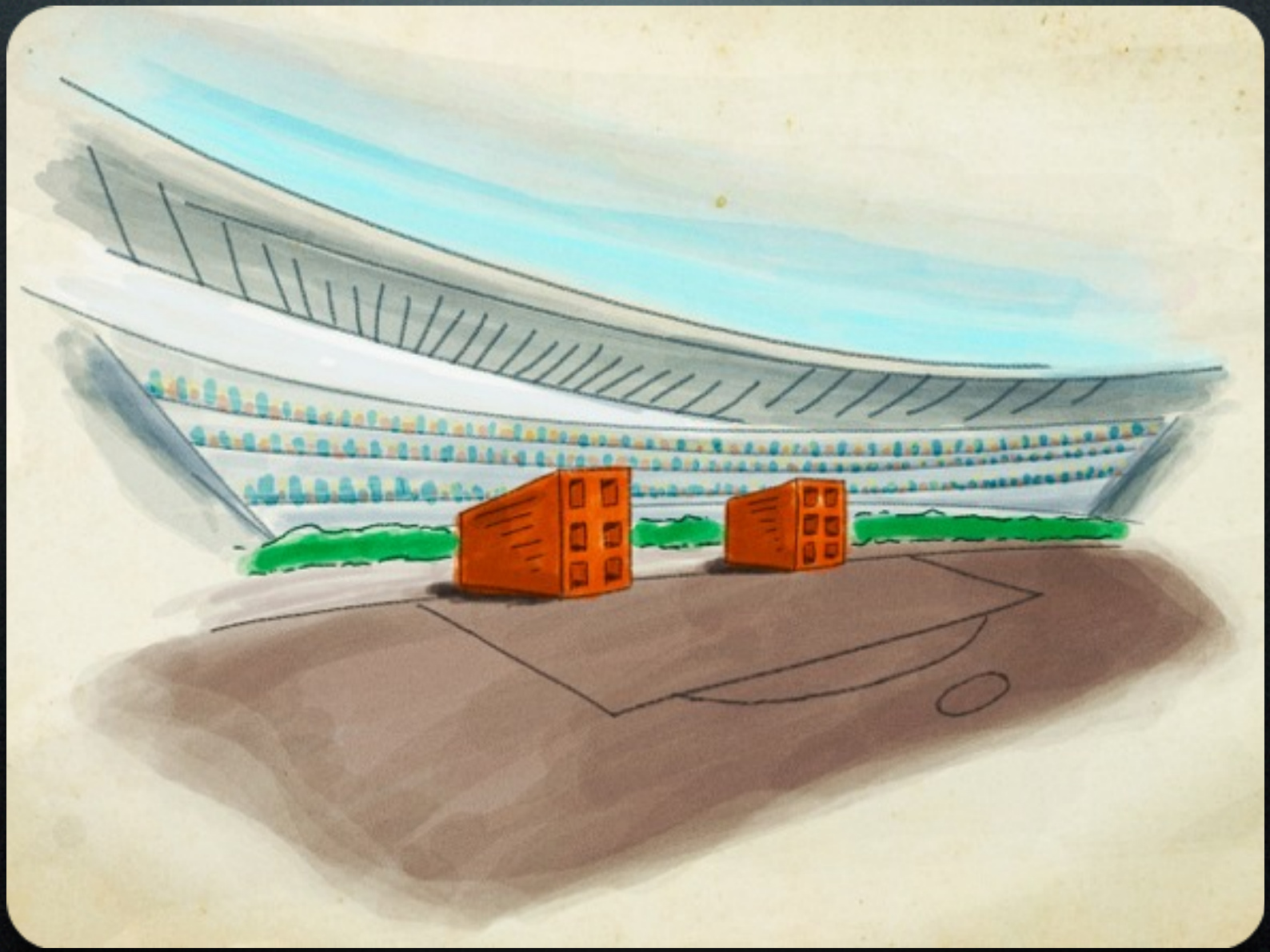
Funções Anônimas

```
def op(x: Int, y: Int, f: (Int, Int) => Int): Int =
    f(x, y)

op(1, 2, (a: Int, b: Int) => a + b)
```

Currying

```
def mul(x: Int) = (y: Int) => x * y

mul(6)(7)
```

```
(y: Int) => 6 * y
```



O que mais além da linguagem?


Web
play!
Web
Lift
ScalaIDE
FOR ECLIPSE
akka
Organização de projetos e Build
sbt
Apps concorrentes e

REPL: Read-Evaluate-Print Loop


```
MacMichael:projetos michael$ scala
Welcome to Scala version 2.10.1 (Java HotSpot
1).
Type in expressions to have them evaluated.
Type :help for more information.

scala> def sum(x: Int, y: Int = 0) = x + y
sum: (x: Int, y: Int)Int

scala> sum(1)
res0: Int = 1

scala>
```

ScalaDoc

http://www.scala-lang.org/api

Nivelamento da Equipe

- Curva de aprendizado
  - Código Java-like no início
  - Uso das demais funcionalidades de Scala depois

- Comunidade pequena, mas crescente
  - Acessível
  - Scala Days scaladays.org

Prática
(com Scala IDE)

Material e instruções iniciais disponíveis em

https://github.com/michaelss/minicurso-scala

Atribuições

introducao/Atribuicoes.scala

Loops

introducao/Loops.scala
introducao/Exercicio1.scala

Recursividade

introducao/RecursividadeJ.java
introducao/Recursividade.scala

Funções!!!

introducao/funcoes/Funcoes1.scala
introducao/funcoes/Funcoes2ruim.scala
introducao/funcoes/Funcoes2legal.scala
introducao/funcoes/Funcoes3.scala
introduca/Exercicio1.scala (de novo!)

Arquivos

introducao/arquivos/Arquivo.scala

Closures e Currying

introducao/funcoes/Closures.scala

Classes

introducao/classes/Pessoa1.scala
introducao/classes/Principal.scala
introducao/classes/Pessoa2.scala

Traits

introducao/traits/Funcionario.scala
introducao/traits/Gerente.scala
introducao/traits/Pessoa.scala
introducao/traits/Principal.scala
introducao/traits/Motorista.scala

Traits

```
trait Plural {
  def pluralizar(x: String) = x + "s"
}

trait Dupla {
  def duplicar(x: String) = x + x
}

class MyString(x: String) extends Plural
    with Dupla { }
```

Pattern Matching

introducao/patterns/Matching.scala
introducao/patterns/Exercicio2.scala

Pattern Matching

```
var sinal = ...
val ch: Char = ...

ch match {
   case '+' => sinal = 1
   case '-' => sinal = -1
   case _ => sinal = 0
}
```

Pattern Matching (2)

```
var sinal = ...
val ch: Char = ...

sinal = ch match {
    case '+' => 1
    case '-' => -1
    case _ => 0
}
```

Pattern Matching (3)


```
def getInt(obj: Any) = obj match {
    case x: Int => x
    case s: String => Integer.parseInt(s)
    case _: BigInt => Int.MaxValue
    case _=> 0
}
```

Case Classes

introducao/patterns/CaseClasses1.scala
introducao/patterns/CaseClasses2.scala

Atores

≅

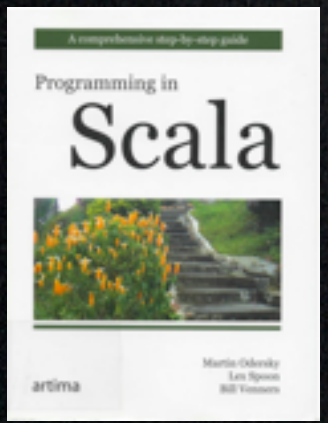
Atores

introducao/atores/Ator1.scala

Livros (alguns)

Cay S. Horstman. Scala for the Impatient. Addison-Wesley: 2012

Nilanjan Raychaudhuri. Scala in Action. Manning: 2013



ODERSKY, Martin; SPOON, Lex; VENNERS, Bill. Programming in Scala. 2. Ed. Artima Inc, 2011. 1ª Ed: http://www.artima.com/pins1ed/

Recursos

- http://twitter.github.com/scala_school/
- http://twitter.github.com/effectivescala/
- https://www.coursera.org/course/progfun

Scala e Programação Funcional

Michael Schuenck dos Santos, MSc.
@michaelss