Sistemas tecnológicos na gestão do Sistema Único de Saúde (SUS)

Teoria Geral da Administração

Daniel Terra Gomes

Estefânio Silva Ribeiro

19 de setembro de 2021

UENF - UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO

CCT - CENTRO DE CIÊNCIA E TECNOLOGIA
LCMAT - LABORATÓRIO DE MATEMÁTICAS
CC - CURSO DE CIÊNCIA DA COMPUTAÇÃO

Primeira edição, Setembro 2021

Sumário

1. Resumo

Scala é a abreviação das palavras SCalable LAnguage. Scala vem com o objetivo de suprir as necessidades dos desenvolvedores de Software modernos. Scala foi iniciada por Martin Odersky em 2001. A sua primeira publicação foi em 20 de janeiro de 2004. Martin, por sua vez, é um professor na School of Computer and Communication Sciences em Ecole Polytechnique Fédérale de Lausanne (EPFL). É uma linguagem com tipagem estática e dinâmica, orientada a objeto, funcional e de alta performance. Esses fundamentos possibilitam trabalhar em cima de problemas de maneira eficiente e com boa abstração. Scala deriva da linguagem de programação Java. Assim sendo, Scala é compilado em "Java-specific formar" chamado de Java Bytecode. Java Bytecode por sua vez é uma linguagem abstraída de máquina ou um conjunto de instruções executadas pelo Java Virtual Machine (JVM). Se pode pensar no JVM como um programa que executa outros programas como uma ambiente de máquina virtual e roda em uma Java Virtual Machine (JVM). Apesar das similaridades com Java há muitas diferenças entre as duas linguagens. A linguagem Scala vem com a intenção de superar o Java em suas limitações.

No geral, linguagens de programações são categorizadas nos segundos tipos:

- High level or Low Level: Sendo relacionado ao seu nível abstração do mundo real. Essa abstração diz sobre o quão próximo à linguagem de programação trabalha próximo ao que os computadores interpretam.
- Orientação a objeto: Se baseia no desenvolvimento de programas pensando em seus blocos; objetos. Esses objetos são programados pensando em suas intenções uns com os outros.
- Estático ou Dinâmico: Estético consiste em que os tipos de dados são checados antes do programa ser executado. Dinâmico, por outro lado, os tipos só são conhecidos após o programa estar em execução.

Ademais, podemos ver que Scala permite Tipagem estática dando a possibilidade de criar aplicações robustas, consertando muitas das falhas presente na linguagem Java.

A Orientação a Objeto (POO) é suportada em Scala disponibilizando uma maneira completa de trabalhar de diversas formas com os Objetos criados. Tudo em Scala é um Objeto. Adicionalmente a isso, a programação funcional permite que a linguagem trabalhe com Big Data. Pois há a possibilidade de valores imutáveis, funções sem efeitos colaterais. Dessa forma, o com o crescimento da linguagem Scala desde seu lançamento em 2004 com a sua presença crescente nas áreas mais atuais, como Big Data, mostra que é uma linguagem para a atualidade. [Ela19], [Ray13], [Wam21], [Xia20], [Hav14], [Dro12], [Wha20].

1.1 Aspectos históricos da linguagem Scala

O Design inicial da Linguagem de programação Scala foi iniciado no ano de 2001, e teve a sua primeira publicação em Janeiro de 2004. Desde seu projeto foi pensada para ser uma linguagem Funcional e Orientada a Objeto. Um dos princípios levados em conta nos primeiros passos foi de introduzir uma linguagem derivada do Java, sendo assim algo melhor que o Java. Mas conectando com as suas infraestruturas, logo o JVM e suas bibliotecas. Inicialmente, os desenvolvedores da linguagem Scala trabalhavam em uma linguagem chamada de Funnel, que buscava ser uma linguagem de classes puras, e de forma padronizada. Apesar de ser uma boa abordagem vista do âmbito acadêmico, mas não foi para os demais. Dessa forma, o time de desenvolvedores decidiram recomeçar no desenvolvimento de uma linguagem derivada do Java, que se encontrava no meio de uma linguagem para a comunidade acadêmica como a Funnel, mas também sendo uma linguagem prática. Sem demora, surgiu a linguagem Scala para cumprir esses requisitos.

1.2 Áreas de Aplicação da Linguagem

As aplicações da linguagem Scala varia entre o espaço acadêmico, e prático. Sendo assim, se aplica na área de Inteligência Artificial, primordialmente na parte de Processamento de Dados, Big Data, Machine Learning. Indo até ao Desenvolvimento Web trabalhando com Scala Js, e Scala Native. Fazendo uso extensivo do JVM e de bibliotecas Java.

1.2.1 Big Data

A linguagem Scala é usada no desenvolvimento de software na indústria para Big Data algumas das ferramentas usadas a partir da linguagem Scala e o Apache Spark e Apache Kafka que oferecem diversas APIs. Scala vem com o seu teor estatístico para a análise de dados nesses âmbitos. Entretanto, especialistas da área de Dados no geral por exemplo: Deep Learning(DL), Reinforcement learning (RL), and artificial intelligence (AI) que sao as areas maior destaque dentro do escopo de Machine Learning (ML). É possível ver que os desenvolvedores desses campos fazem uso majoritariamente da linguagem de programação Python, C + +. Devido a isso o uso dessas linguagens cresceram e o uso da Scala vem caindo comparado com as mesmas. A quantidade de dados produzidos vem crescendo a cada momento, e o processamento de dados para pequenos e grandes volumes de dados estão cada vez mais comuns. Assim para extrair informações valiosas no meio de milhares ou milhões de dados, ferramentas e Frameworks de processamento de dados. Scala se torna um dessas ferramentas para trabalhar com Big Data. Scala, diferentemente de umas das linguagens mais usadas para Big Data; Python, se diferencia na necessidade de especificar o seu tipo de

dado, sendo que Python não requisita isso. Ademais, o objetivo central da linguagem ao trabalhar com BIg Data é alcançar uma solução fácil de processamento, sendo assim com opções de processamento paralelo com pouca mudança no código inicial. [Wam21], [Xia20], [Hav14].

1.2.2 Programação Web

Scala também é usado para programação Web através do uso conjunto com JavaScript por meio do Scala.js. O Scala ainda está disponível no Scala Native. Atrelado com a sua possibilidade de programação funcional e Programação Orientada a Objeto (POO) proporciona uma maneira elegante e concisa de desenvolvimento. Como Scala é integrável com Java rodando através da JVM, os programadores usando Scala podem usar bibliotecas de Java de maneira direta, podendo até mesmo chamar códigos Scala a partir do Java. Por Scala ser uma linguagem de programação Open Source podendo ser usada por qualquer desenvolvedor, diminuindo os custos de desenvolvimento de páginas Web.

As vantagens do desenvolvimento Web com a linguagem Scala:

- Alta versatilidade;
- Alta performance e código mais enxuto;
- Comunidade Open Source;
- Vasto conjunto de ferramentas, bibliotecas.

Dessa forma, Scala vem para trazer essas vantagens com um ecossistema bem equipado de Ferramentas para a várias situações permitindo suporte modular para as necessidades de cada implementação.

Ferramentas para o Scala:

- Scalatra: Scalatra é um Framework que é uma forma de se conectar com Ruby Sinatra. Dando aos desenvolvedores uma forma poderosa de conectar o JVM com o Scala ajudando a criar páginas Web e APIs.
- BlueEyes: BlueEyes Framework focado em performance e composição.
- Akka HTTP: O Akka HTTP é uma implementação para o Scala pensando para o seu desenvolvimento como se fosse uma idealização de um formato. Traz em seus princípios de não ser uma Framework. Mas mais uma opção de ferramental universal para serviços HTTP. Desse modo, apesar de não ter sido a sua intenção inicial. É visto por muitos como um excelente Framework para o Scala.

A lista de opções de Framework para a linguagem Scala é imensa, caberá a cada desenvolvedor entender as suas necessidades e buscar entender mais das opções de ferramentas para trabalhar junto ao Scala.

[Wha20], [Dro12].

1.2.3 outras

- Engenharia de Dados;
- Infraestrutura de Sistemas;
- Computação distribuída.

2. Introdução

Os livros básicos para o estudo da Linguagem Scala são: [Ela19], [Wam21], [Ray13] e []
Neste capítulo é apresentado .... Segundo [], a linguagem Scala, . . .
De acordo com [] e [], a linguagem Scala . . . [] afirma que a linguagem Python . . .
Considerando que a linguagem Scala ([], []) é considerada como ....

2.1 Variáveis e constantes

2.2 Tipos de Dados Básicos

2.2.1 String

Código fonte para a linguagem Scala:

```
    object HelloWorld {
  def main(args: Array[String]): Unit = {
    println("Hello, world!")
  }
}
```

```
  class Rational(n: Int, d: Int) {

    require(d != 0)

    private val g = gcd(n.abs, d.abs)
    val numer = n / g
    val denom = d / g

    def this(n: Int) = this(n, 1)

    def + (that: Rational): Rational =
      new Rational(
```

```
      numer * that.denom + that.numer * denom,
      denom * that.denom
    )

  def + (i: Int): Rational =
    new Rational(numer + i * denom, denom)

  def - (that: Rational): Rational =
    new Rational(
      numer * that.denom - that.numer * denom,
      denom * that.denom
    )
```

2.3 Operadores e Expressões em Scala

3. Problemática

3.1 Entradas e saídas

3.1.1 Entrada e Saída formatada

3.2 Seleção

Tipos de IF

Select

3.3 Repetição

3.4 Funções

3.5 Módulos e Subprogramas

```
class Rational(n: Int, d: Int) {

    require(d != 0)

    val numer: Int = n
    val denom: Int = d

    def this(n: Int) = this(n, 1) // auxiliary constructor

    override def toString = numer +"/"+ denom

    def add(that: Rational): Rational =
      new Rational(
        numer * that.denom + that.numer * denom,
        denom * that.denom
      )
  }
```

4. Hipótese

Devem ser mostradas pelo menos CINCO aplicações completas da linguagem, e em cada caso deve ser apresentado:

- Uma breve descrição da aplicação
- O código completo da aplicação,
- Imagens do código fonte no compilador,
- Imagens dos resultados após a compilação-interpretação do código fonte
- Links e referencias bibliográficas de onde foi obtido a aplicação

4.1 Operações básicas

Pode ser a implementação de um menu das operações aritméticas básicas

4.2 O algoritmo Quicksort en Scala

Este algoritmo esta disponível na internet: só copiar, adaptar, comentar o código e compilar

4.3 Programa de Cálculo Numérico

Pode ser a solução de um sistema de equações, o cálculo das raízes de uma função, interpolação, etc.

4.4 Aplicação usando Matrizes

4.5 Aplicações Profissionais

Aqui pode ser qualquer aplicação de outra área de conhecimento, por exemplo: Física, Mecãnica de Fluídos, Biologia, Astronomia, Jogos, Química, etc. Pesquisar na Internet, para aplicações prontas e pequenas.

5. Ferramentas existentes e utilizadas

Neste capítulo devem ser apresentadas pelo menos DUAS (e no máximo 5) ferramentas consultadas e utilizadas para realizar o trabalho, e usar nas aplicações. Considere em cada caso:

- Nome da ferramenta (compilador-interpretador)
- Endereço na Internet
- Versão atual e utilizada
- Descrição simples (máx 2 parágrafos)
- Telas capturadas da ferramenta
- Outras informações

5.1 Editores para Scala

5.2 Compiladores

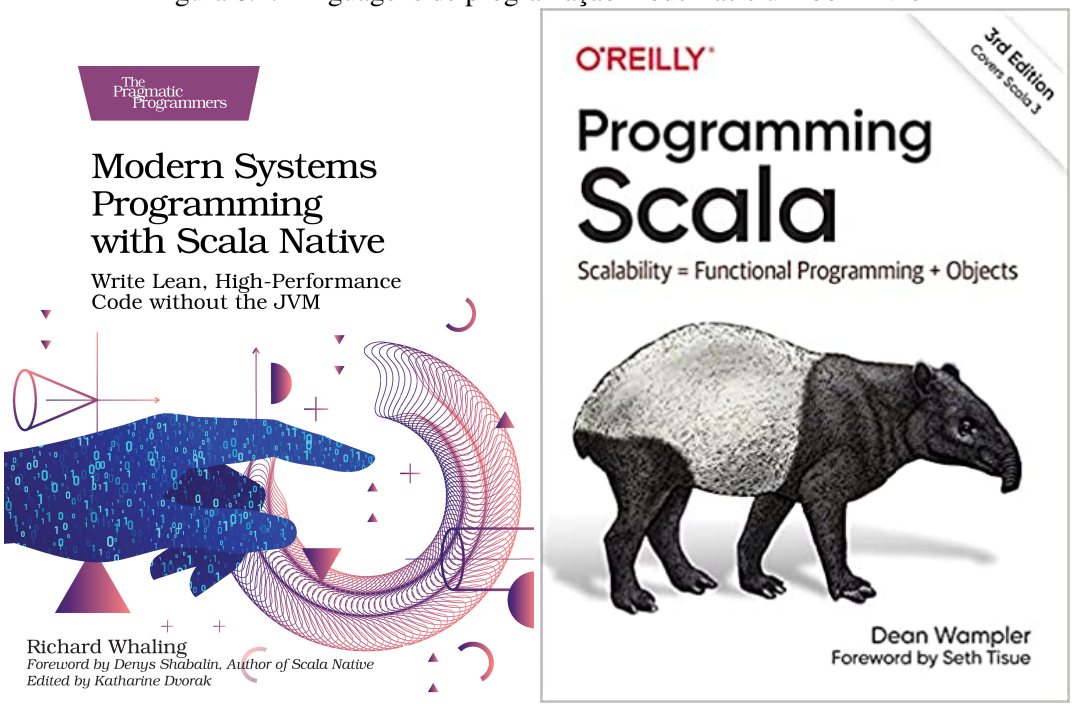
- Site principal : https://www.scala-lang.org/
- Scala 3 : https://www.scala-lang.org/download/
-
-
-

5.3 Ambientes de Programação IDE para Scala

6. Conclusões

Os problemas enfrentados neste trabalho ...
O trabalho que foi desenvolvido em forma resumida ...
Aspectos não considerados que poderiam ser estudados ou úteis para ...

Figura 6.1: Linguagens de programação modernas e um bom livro



Fonte: O autor

Referências Bibliográficas

[Dro12] Sadek Drobi. Play2: A new era of web application development. *IEEE Internet Computing*, 16(4), 2012. Citado 2 vezes nas páginas 6 e 7.

[Ela19] Irfan Elahi. *Scala Programming for Big Data Analytics*. APRESS L.P., Notting Hill, VIC, Australia, 1 edition, July 2019. Citado 2 vezes nas páginas 6 e 9.

[Hav14] Klaus Havelund. Data automata in scala. In *2014 Theoretical Aspects of Software Engineering Conference*, 2014. Citado 2 vezes nas páginas 6 e 7.

[Ray13] Nilanjan Raychaudhuri. *Scala in Action: Covers Scala 2.10*. MANNING PUBN, Shelter Island, NY, 2 edition, April 2013. Citado 2 vezes nas páginas 6 e 9.

[Wam21] Dean Wampler. *Programming Scala*. O'Reilly Media, Inc., Sebastopol, CA, 3 edition, July 2021. Citado 3 vezes nas páginas 6, 7 e 9.

[Wha20] Richard Whaling. *Modern Systems Programming with Scala Native: Write Lean, High-Performance Code Without the Jvm*. PRAGMATIC BOOKSHELF, 1 edition, February 2020. Citado 2 vezes nas páginas 6 e 7.

[Xia20] Guangming Xian. Parallel machine learning algorithm using fine-grained-mode spark on a mesos big data cloud computing software framework for mobile robotic intelligent fault recognition. *IEEE Access*, 8, 2020. Citado 2 vezes nas páginas 6 e 7.