

# Linguagem Scala



---

Andrei Alves Cardoso  
Erika Mayumi Saito Tagima  
Tales Henrique Carvalho  
Leonardo Said da Costa  
Breno Salgado de Oliveira  
Caíque Cléber Dias de Rezende

# Histórico da linguagem

- Ano de criação: 2001
- Onde: École Polytechnique Fédérale de Lausanne (EPFL), na Suíça
- Publicação: 2004



Martin Odersky  
idealizador da Scala



# **Características principais**

# Multi-paradigma

Orientada a Objetos

```
class Greeter(prefix: String, suffix: String) {  
    def greet(name: String): Unit =  
        println(prefix + name + suffix)  
}
```

Funcional

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



# Linguagem de alto nível

- Baseada em Java e C#
- Tem uma sintaxe fácil e familiar
- Os tipos de variável são inferidos pelo compilador



# Linguagem Escalável

- Definição de escalabilidade: capacidade de um sistema em suportar um aumento de carga
  - Garantida pelos conceitos de orientação a objeto
  - Exemplos: abstração e herdagem
- Destina-se à programação de componentes
  - Bibliotecas, classes, frameworks, web services, etc.



# Roda na máquina virtual Java (JVM)

- Compatível com classes feitas em Java
  - E vice-versa!
- Tem o mesmo processo de compilação do Java
  - Output intermediário que é interpretado pelo JVM
- Ou seja, o processo de rodar um programa em Scala é lento



**Quem está usando  
o Scala?**



Linked 

SIEMENS

SONY

Reaktor

The Guardian

Quem está usando  
o Scala?

Thatcham  
Research



OPower



edf

FOURSQUARE

Novell®

ebay

<https://www.scala-lang.org/old/node/1658>

# Exemplo de aplicação: Finagle



- Open Source, feito pelo Twitter
  - <https://github.com/twitter/finagle>
- Sistema de chamada de procedimento remoto para construir servidores com alta concorrência
- Oferece uma API que pode ser adaptada para qualquer protocolo cliente-servidor



# Programando em Scala

```
class Folder(name: String, path: Option[String]) {
  def fullPath: String = path match {
    case Some(p) => List(p, name).mkString("/")
    case None => s"./$name"
  }
}

class File(name: String, folder: Option[Folder]) {
  def fullPath: String = folder match {
    case Some(f) => List(f.fullPath, name).mkString("/")
    case None => s"./$name"
  }
}

val resources = Vector[Resource](
  File("ex1.scala", Some(Folder("example", Some("/usr")))),
  Folder("tmp"),
  Folder("bin", Some("/usr")),
  File(".zshrc")
)

foreach {
  println(s"File")
  println(s"Folder")
}
```

# Expressões

## Valores

Valor imutável

Implícito:

```
val x = 1 + 1  
println(x)
```

Output:  
2

Explícito:

```
val x: Int = 1 + 1
```

## Variáveis

Valor mutável

Implícito:

```
var x = 1 + 1  
x = 3  
println(x * x)
```

Output:  
9

Explícito:

```
var x: Int = 1 + 1
```

# Blocos

## Combinação de diferentes códigos

```
println({  
    val x = 1 + 1  
    x + 1  
})
```

Output:  
3

# Estruturas de condição

**If-else:** idêntico ao Java

**If-else compacto:**

```
def abs(x: Double) = if (x >= 0) x else -x
```

Observação: similar ao operador ternário do C++

# Estruturas de repetição

## For:

```
for( var x <- 1 to 10 ){  
    // O loop repete 10 vezes  
}
```

## For com múltiplos ranges:

```
for(a <- 1 to 2; b <- 1 to 3){  
    print("(" + a + ", " + b + ")");  
}
```

Output:

```
(1, 1) (1, 2)  
(1, 3) (2, 1)  
(2, 2) (2, 3)
```

**While e do-while:** idênticos ao Java

# Funções

## Declaração com nome

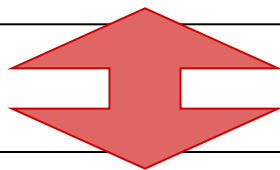
```
def incremento1(x:Int) = x + 1  
println(incremento1(1))
```

Output:  
2

## Anônima

```
var incremento2 = (x:Int) => x + 1  
println(incremento2(1))           // 2
```

Output:  
2



```
var incremento2 = new Function1[Int, Int]{  
    def apply(x: Int): Int = x + 1  
}  
println(incremento2(1))           // 2
```

Output:  
2



# Funções especiais das... Funções!

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

## Função que recebe função como parâmetro

```
def sqr(x:Int) = x * x  
println(soma(1, 2, sqr))
```

Output: 7

```
println(  
soma(1,2, (x:Int) => x*x))
```

Output: 7

## Currying (Retorna função)

```
def add(x:Int)(y:Int)=x+y  
val op1 = add(1)(5) //1+5  
val op2 = add(4)_  
//Retorna função Int=>Int  
val op3 = op2(2) //4+2
```

```
println(op1)  
println(op2)  
println(op3)
```

Output:

```
6  
<function1>  
6
```

# Scala Vs Java



Scala

```
def products = orders.flatMap(o => o.products)
```

```
public List<Product>getProducts(){  
    List<Product> products = new ArrayList<Product>();  
    for (Order order : order) {  
        products.addAll(order.getProducts());  
    }  
    return products;  
}
```



# Declaração de classes

```
public class Book {  
    private String name;  
    private String author;  
    private int numberOfPages;  
    public Book(String n, String a, int num) {  
        this.name = n;  
        this.author = a;  
        this.numberOfPages = num;  
    }  
}
```



# Declaração de classes

```
case class Book(name: String, author: String,  
                numberOfPages: Int)
```



```
public class WordCountJava {  
    public static void main(String[] args) {  
        StringTokenizer st = new  
            StringTokenizer(args[0]);  
        Map<String, Integer> map = new  
            HashMap<String, Integer>();  
        while (st.hasMoreTokens()) {  
            String word = st.nextToken();  
            Integer count = map.get(word);  
            if (count == null)  
                map.put(word, 1);  
            else  
                map.put(word, count + 1);  
        }  
        System.out.println(map);  
    }  
}
```

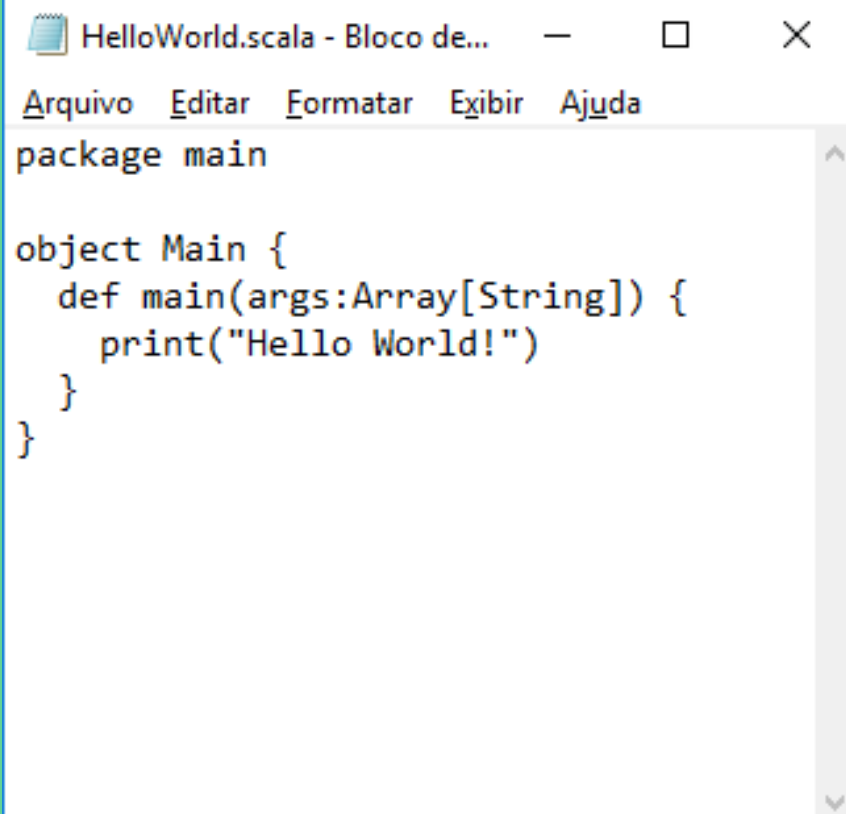


# Programa de contagem de palavras

```
object WordCountScala extends App {  
  println(  
    args(0)  
      .split(" ")  
      .groupBy(x => x)  
      .map(t => t._1 -> t._2.length))  
}
```



# Como usar o Scala no meu computador?

A screenshot of a text editor window titled "HelloWorld.scala - Bloco de...". The window has a menu bar with "Arquivo", "Editar", "Formatar", "Exibir", and "Ajuda". The code inside is a Scala program that prints "Hello World!".

```
package main

object Main {
  def main(args:Array[String]) {
    print("Hello World!")
  }
}
```

# SBT (Scala Build Tools)

- Usa comandos do JDK (Java Development Kit)
- É executado no terminal
- Consegue compilar e interpretar o programa





# SBT (Scala Build Tools)

```
C:\Users\Cliente\Documents\Faculdade\S08\ECOM04\TrabalhoScala...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins
Window ?
Main.scala
33
34 //Classe que encapsula uma cobrinha.
35 //Uma "cobrinha" é composta por vários segmentos, cada
36 //um em uma posição x e y definidas.
37 class Cobrinha(val bordasTabuleiro:Point2D, val
38 //O comprimento máximo da cobrinha corresponde à
39 //área do tabuleiro.
40 private var _comprimentoMaximo = (bordasTabuleiro.x
41 *bordasTabuleiro.y).toInt;
42 //A posição de cada segmento da cobra é colocada
43 //uma array.
44 private var segmentos:Array[Point2D] = new Array[
45 Point2D](_comprimentoMaximo);
46 //Inicializa aleatoriamente a direção da cobrinha.
47 //0 = direita, 1 = cima, 2 = esquerda, 3 = baixo.
48 private var _direcao:Int = (new scala.util.Random
49 ().nextInt(4));
50 //Estado da cobrinha: 0 = viva, -1 = morta, 1 =
51 venceu
52 private var _estado:Int = 0;
53 //Comprimento da cobrinha, em número de segmentos.
54 private var _comprimento:Int = 1;
55 //Inicializa a posição da cabeça.
56 segmentos(0) = posicaoInicial;
57
58 //Reconhecedores de estado.
59 def comprimento = _comprimento;
60 def estado = _estado;
61 def estaViva = if(estado==0) true else false;
```

```
Prompt de Comando - sbt
C:\Users\Cliente\Documents\Faculdade\S08\ECOM04\TrabalhoScala\sna
keScala>sbt
Java HotSpot(TM) 64-Bit Server VM warning: Ignoring option MaxPer
mSize; support was removed in 8.0
Java HotSpot(TM) 64-Bit Server VM warning: Ignoring option MaxPer
mSize; support was removed in 8.0
[info] Loading settings for project snakescala-build from assembl
y.sbt ...
[info] Loading project definition from C:\Users\Cliente\Documents
\Faculdade\S08\ECOM04\TrabalhoScala\snakeScala\project
[info] Loading settings for project snakescala from build.sbt ...
[info] Set current project to snakeScala (in build file:/C:/Users
/Cliente/Documents/Faculdade/S08/ECOM04/TrabalhoScala/snakeScala/
)
[info] sbt server started at local:sbt-server-e5a69c77b437ded421c
5
sbt:snakeScala> run_
```

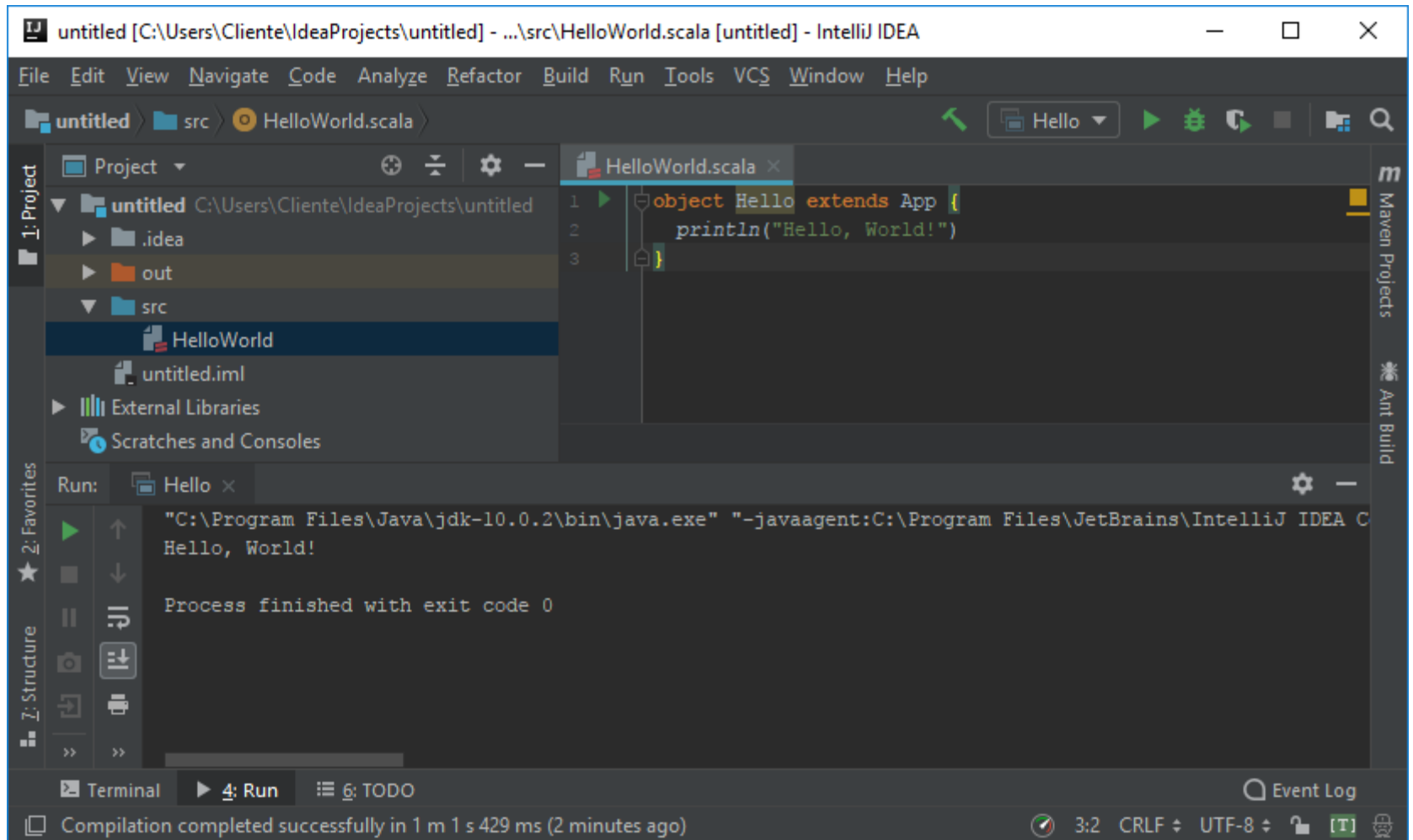


# IntelliJ IDEA

- IDE mais utilizada na comunidade Scala
- Necessita da instalação do plugin da Scala
  - A instalação é feita na própria IDE
- Curiosidade: IDE que originou o Android Studio



# IntelliJ IDEA



# IntelliJ IDEA

The screenshot displays the IntelliJ IDEA interface with a Scala project named 'untitled'. The main editor shows the following code in `HelloWorld.scala`:

```
1 object Hello extends App {  
2   println("Hello, World!")  
3 }
```

A performance monitoring window is overlaid on the code editor, showing resource usage for the current process and other applications:

Nome	85% CPU	84% Memória	21% Disco	0% Rede
> Google Chrome (36)	3,1%	1.471,9 MB	0,1 MB/s	0 Mbps
> IntelliJ IDEA (9)	75,9%	1.335,5 MB	0,3 MB/s	0 Mbps

The Run tool window at the bottom shows the execution output:

```
Run: Hello x  
"C:\Program Files\Java\jdk-10.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA C  
Hello, World!  
  
Process finished with exit code 0
```

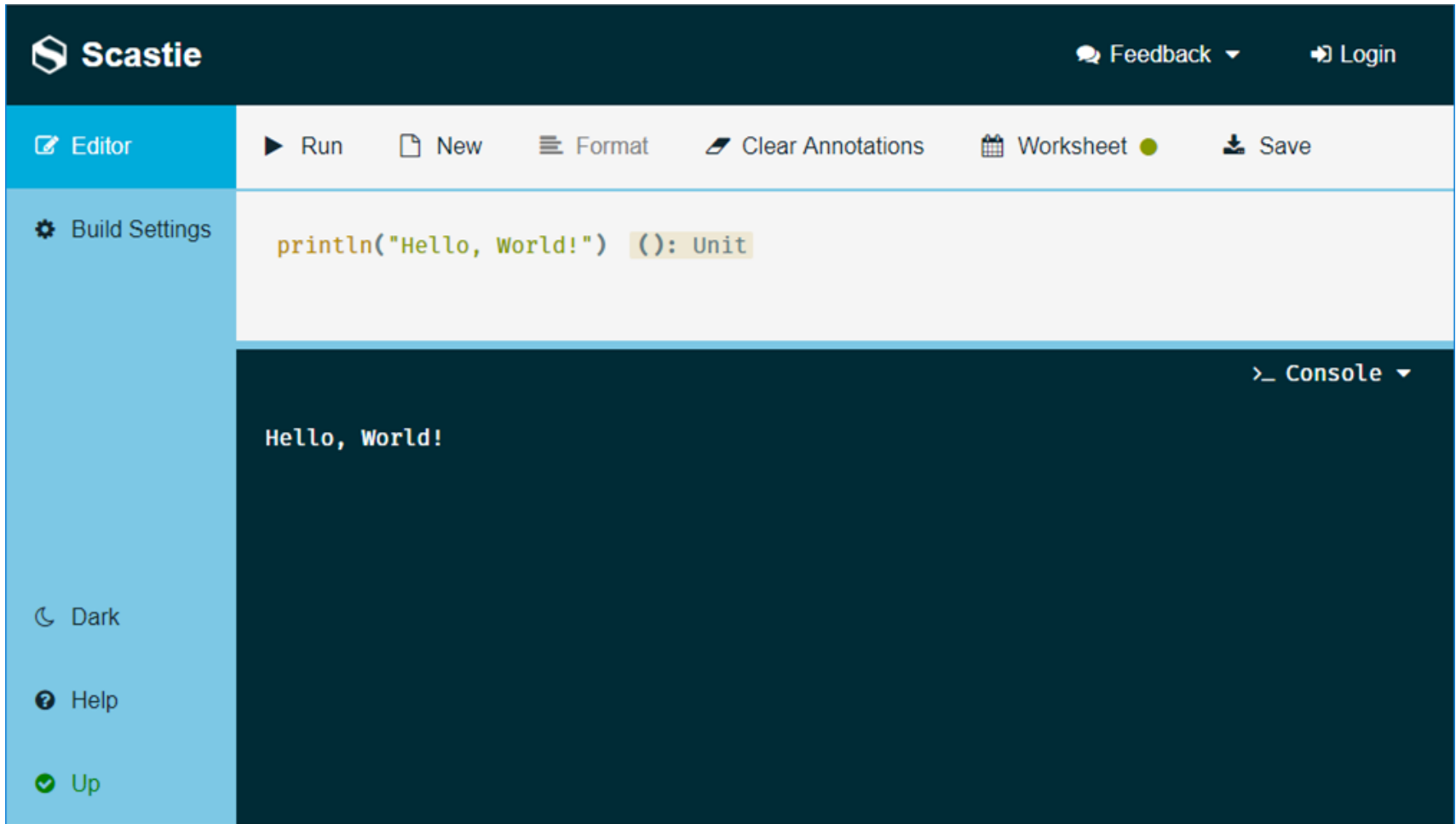
The status bar at the bottom indicates: "Compilation completed successfully in 1 m 1 s 429 ms (2 minutes ago)".

# Scastie - <https://scastie.scala-lang.org/>

- Executada no navegador
  - Não precisa de nenhuma instalação (nem do SBT)
- Usada para realizar testes simples



# Scastie - <https://scastie.scala-lang.org/>



The screenshot displays the Scastie web application interface. At the top, the Scastie logo is on the left, and 'Feedback' and 'Login' links are on the right. Below the header is a toolbar with icons for 'Run', 'New', 'Format', 'Clear Annotations', 'Worksheet', and 'Save'. A left sidebar contains links for 'Editor', 'Build Settings', 'Dark' theme, 'Help', and 'Up' status. The main area shows a Scala code editor with the text `println("Hello, World!") (): Unit`. Below the editor is a console output area labeled '>\_ Console' which displays the text 'Hello, World!'.

Scastie

Feedback Login

Editor Run New Format Clear Annotations Worksheet Save

Build Settings

```
println("Hello, World!") (): Unit
```

>\_ Console

Hello, World!

Dark

Help

Up

# Exemplo

- Banco de dados de **peessoas, trabalhadores e estudantes**.
  - Pessoas têm nome, idade, altura, peso.
  - Trabalhadores estão em uma empresa, tem cargo, salário e data de admissão.
  - Estudantes estão matriculados em matérias, tem notas e índices.
- **Trabalhadores são pessoas.**
- **Estudantes são pessoas.**

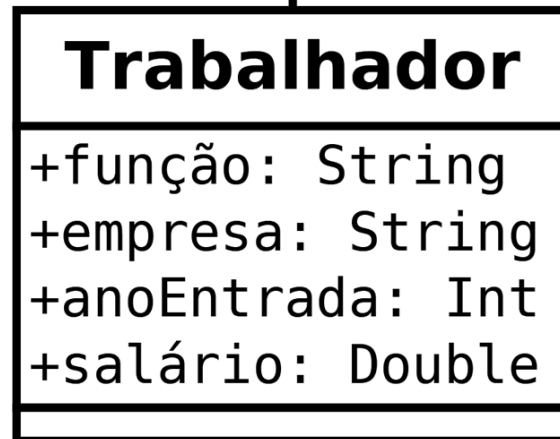
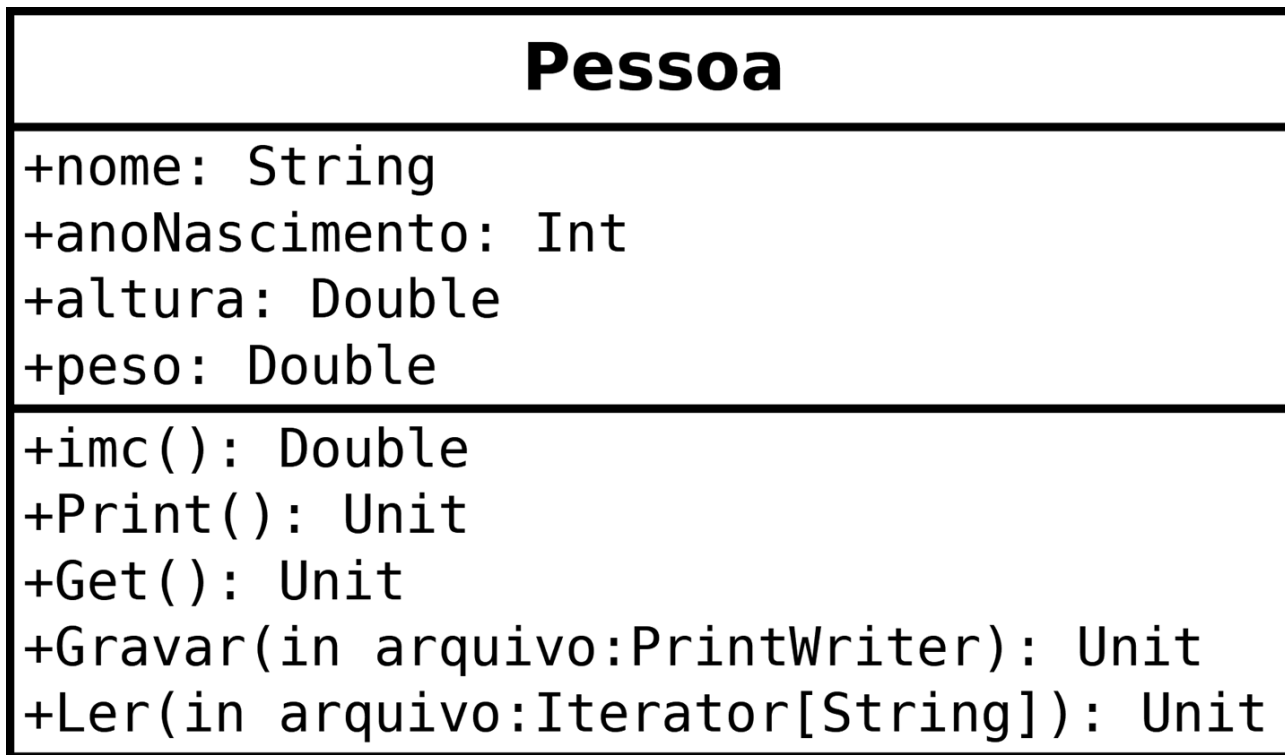


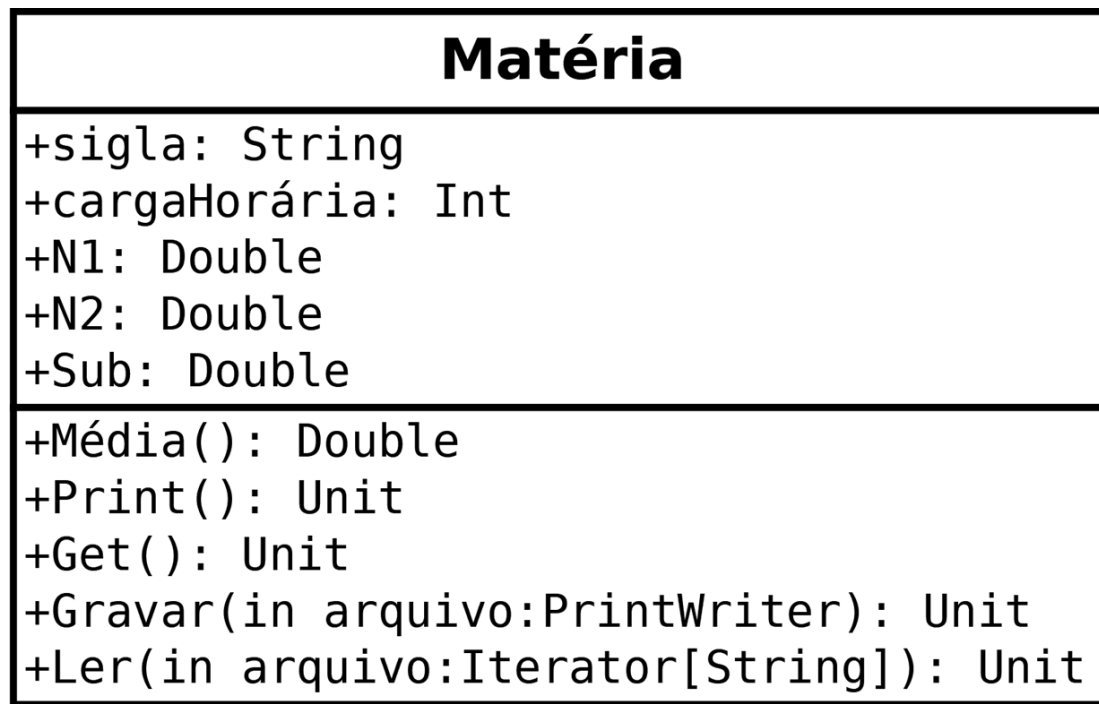
## **Pessoa**

+nome: String  
+anoNascimento: Int  
+altura: Double  
+peso: Double

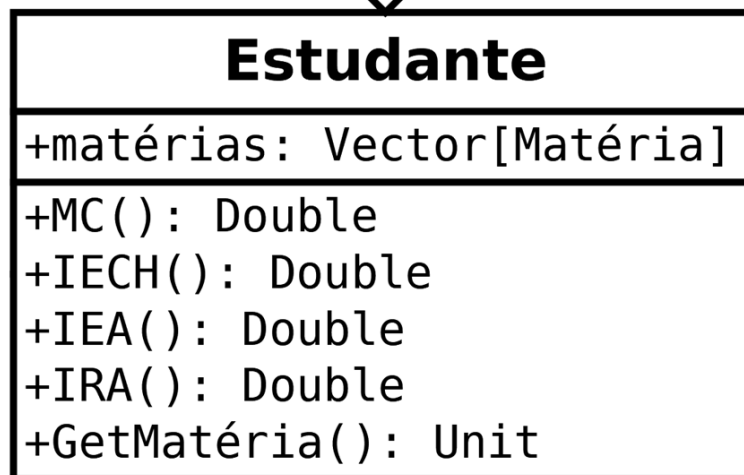
+imc(): Double  
+Print(): Unit  
+Get(): Unit  
+Gravar(in arquivo:PrintWriter): Unit  
+Ler(in arquivo:Iterator[String]): Unit

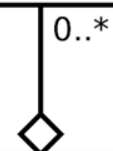
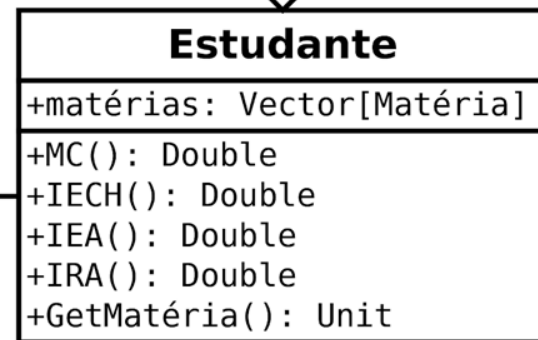
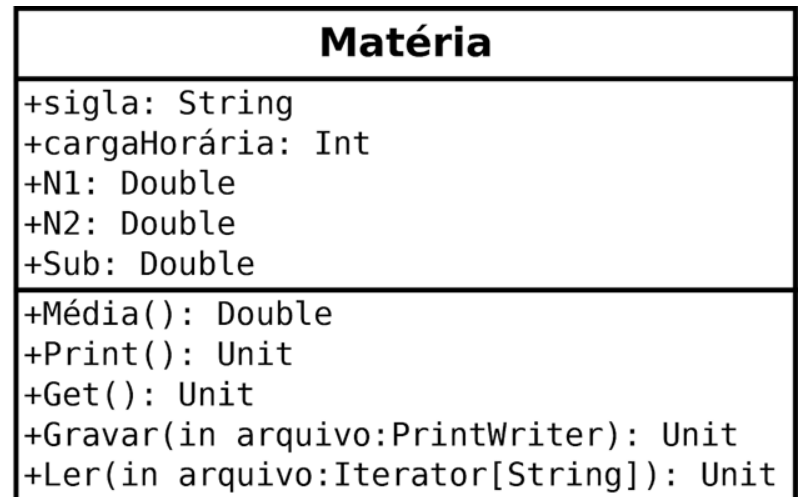
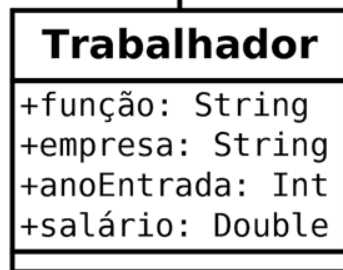
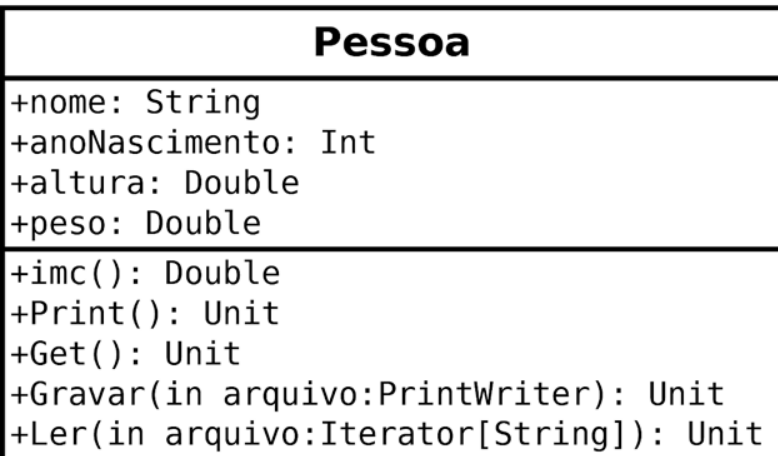






0..\*









```
eMenu.MenuH();
//Imprime um cabeçalho sobre as materias
io.File;
writer; def MenuH():Unit={
def LerMateria(arq:Iterator[Str
sigla = arq.next();
```

Nome Maria  
Nascimento 1990  
Altura 1.6 m  
Peso 55.0 kg  
IMC 21.484374999999996 kg/m2

Matriculado em 4 materia(s):

Sigla	Carga	N1	N2	Sub	Media	Resultado
ECAC02	64 [h]	6,0	9,0	0,00	7,50	APR
ECOM04	32 [h]	10,0	10,0	0,00	10,00	APR
ELTA03	48 [h]	7,0	3,0	5,50	6,25	APR
MAT001	96 [h]	6,0	4,0	5,30	5,65	REP

IRA: 6,84  
MC: 7,64  
IECH: 0,60  
IEA: 4,58

O que quer fazer?

- 1-Ver detalhes
- 2-Reentrar dados
- 3-Cadastrar materia
- 4-Excluir

```
Console.flush();
return valor;
ci += mat.cargaHoraria;
class Matnext();
var ciTotal = 0.0;
```

# Exemplo

- Resolver um problema de **motivação** e **produtividade** no campo de trabalho.
- **Um jogo simples!**





# Exemplo snakeScala

- Desafios:
  - Gráficos
  - Controles
  - Temporização
- Biblioteca:
  - **scalafx**: janela, IO, gráficos, temporização





## Cobrinha

+segmentos: Array[Point2D]  
-comprimentoMaximo: Int  
+bordasTabuleiro: Point2D  
+direção: Int  
+comprimento: Int  
+estado: Int

+Cobrinha(in bordasTabuleiro:Point2D,in posicaoInicial:Point2D)  
+Pertence(in ponto:Point2D): Boolean  
+ObterProximaPosicao(): Point2D  
+ObterPosicao(in indice:Int): Point2D  
+Alimenta(): Unit  
+Movimenta(): Unit

# Janela

```
-cena: Scene
-cv: Canvas
-gc: GraphicsContext
+largura: Int
+altura: Int
+teclaPausa: Boolean
+teclaCorre: Boolean
+direcao: Int

+Janela(in nome:String,in largura:Int,in altura:Int)
+LimpaTela(out cor:Color): Unit
+DesenhaRetangulo(in x:Double,in y:Double,
                  in larg:Double,in alt:Double,
                  in cor:Color): Unit
+DesenhaGrade(in larg:Double,in alt:Double): Unit
+DesenhaTexto(in texto:String,in x:Double,
              in y:Double,in cor:Color,in fonte:String,
              in tamanho:Int): Unit
+LimpaEntradas(): Unit
```

# Exemplo snakeScala - Classe Principal

- Cria uma Janela e uma Cobrinha
- Repassa dados de IO da Janela para a Cobrinha
- Redesenha a tela em intervalos regulares
- Pausa e despausa (inibe IO e movimento na Cobrinha)
- Anuncia vitória e derrota

