Trabalho 01 de EDL

Alunos: Ricardo Rolo e Renan Carvalho
Linguagem: Scala

Introdução

- Mistura métodos de Orientação a Objetos e programação funcional
- Estilo de programação mais conciso
- Redução da quantidade de código

Origens e Influências

- Criada por Martin Odersky e seu grupo
- Possui as JVMs como mecanismo de execução
- Começou a ser desenvolvida por Odersky em 2001
- Utiliza os mesmos operadores que Java e C
- Em 2011 recebeu incentivos, o que alavancou o crescimento da linguagem

Algumas Características Sintáticas

- Não é preciso declarar explicitamente o tipo das variáveis
- Definições começam com uma palavra reservada
- Funções podem ser aninhadas dentro de outras funções (Funções de Alta Ordem)
- Arrays são declarados por “<nome>: Array[<tipo>]” ao invés de “<tipo> <nome>[ ]”

Classificação

- Orientada a objetos
- Funcional
- Tipagem estática
- Extensível

Lazy Evaluation

Lazy Evaluation é uma estratégia de avaliação de expressões utilizada em linguagens funcionais como Haskell, por exemplo. Com essa estratégia, os valores das expressões são avaliados somente quando forem requisitados.

Scala

```scala
object exemploparam {

    def main(args: Array[String]) : Unit = {
        exemploStatic(add(2,3))
        exemploLazy(add(2,3))
    }

    def add(a: Int, b: Int) : Int = {
        println("adicionando...");
        a+b
    }

    def exemploLazy(x: => Int) : Unit = {
       println("exemplificando Lazy...")
       println(x)
    }

    def exemploStatic(x: Int) : Unit = {
        println("exemplificando Static...")
        println(x)
    }
}
```

```
$ scala -cp . exemploparam
adicionando...
exemplificando Static...
5
exemplificando Lazy...
adicionando...
5
```

C++

```cpp
#include <iostream>

using namespace std;

int add(int a, int b) {
    cout << "adicionando..." << endl;
    return (a+b);
}

void exemploStatic(int x) {
    cout << "exemplificando Static..." << endl << x << endl;
}
int main() {
    exemploStatic(add(2,3));
    return 0;
}
```

```
$ ./a.exe
adicionando...
exemplificando Static...
5
```

Implicits

- Funções Implícitas
  - **Classes Implícitas**
  - **Métodos Implícitos**
  - **Parâmetros Implícitos**
- Mais expressividade
- Ganho em prototipação
- Possibilidade de parametrizar métodos, com Tipos que não atendam o método, uma vez que declare um método implícito de conversão. (Exemplo no próximo slide)

Parâmetros Implícitos

```
object exemplo{

  def myName(implicit n: String) = println("my name is " + n + " !!!")

  def main(args: Array[String]): Unit = {

    implicit val name = "Nome implicito"

    myName("renan") // Printa o método aplicado ao parâmetro que foi passado.
    myName // Printa o método aplicado à variável implícita que foi inicializada no escopo e,
           // implicitamente, parametrizada no método.
  }
}
```

Classes Implícitas

```
object NewString {
  implicit class MinhaString(s: String) {
    def meu_metodo = s + " !!!!"
  }
}
```

```
import com.projeto.utils.NewString._

val s = "Olá"
s.meu_metodo
```

Scala

```scala
object exemplo{

    implicit def intToString (x:Int): String = { // Aqui definimos um método implícito que vai transformar o inteiro que passamos ao método imprime().
        return x.toString;
    }

    def imprime (s:String) = println(s); // Definindo um método que recebe uma string e imprime a mesma.

    def main(args: Array[String]): Unit ={

        imprime(1234); // Chamando o método imprime, que naturalmente receberia um parametro <String>, e passamos um parâmetro <Inteiro>.
                       // O compilador vê que o método chamado recebeu um <Inteiro>, então ele busca no escopo da chamada, algum método marcado com *implicit* que
                       // receba um <Inteiro> e retorne uma <String> (nesse caso, o método intToString atende esse requisito).
                       // Assim, ele transforma "imprime(1234)" em "imprime(intToString(1234))".

    }
}
```

```
PS C:\Users\zuand\Documents\Programação\Scala> scalac exemplo.scala
PS C:\Users\zuand\Documents\Programação\Scala> scala exemplo
1234
```

Java

```java
public class exemplo{

  public static void imprime(String s){
    System.out.println("Nossa frase e: "+s);
  }

  public static String intToString(Integer x){
    return Integer.toString(x);
  }

  Run | Debug
  public static void main(String[] args) {
    Integer var = 5;
    String frase = "exemplo";

    imprime(var);

  }
}
```

Erro na compilação pois o método em Java não aceita, como parâmetro, um tipo diferente do que foi definido anteriormente.

imprime(var) // Erro !
// Inteiro passado como parâmetro, o método só aceita String.

Aqui, verificamos se x é uma referência de Inteiro, caso true, vamos imprimir o mesmo.

Convertemos através do método intToString(), que nos retorna uma String. Assim, poderemos usar o método imprime().

```java
public class exemplo{

  public static void imprime(String s){
    System.out.println("Nossa frase e: "+s);
  }

  public static String intToString(Integer x){
    return Integer.toString(x);
  }

  Run | Debug
  public static void main(String[] args) {
    Integer var = 5;
    String frase = "exemplo";

    if (var instanceof Integer){
      imprime(intToString(var));
    }

  }
}
```

Bibliografia

- Site da linguagem: https://www.scala-lang.org/
- "What is Scala?": https://searchbusinessanalytics.techtarget.com/definition/Scala-Scalable-Language
- Conheça a linguagem Scala: https://www.devmedia.com.br/conheca-a-linguagem-scala/32850
- Wikipedia: https://pt.wikipedia.org/wiki/Scala_(linguagem_de_programa%C3%A7%C3%A3o)
- Diferença entre tipagens: http://felipequadros.com/programa%C3%A7%C3%A3o/b%C3%A1sico/iniciante/geral/teoria/2016/06/05/tipagem-fraca-forte-est%C3%A1tica-din%C3%A2mica-e-inferencia-de-tipo/
- Factorial in scala: https://alvinalexander.com/scala/scala-factorial-recursion-example-recursive-programming
- Call By Name parameters:https://stackoverflow.com/questions/13337338/call-by-name-vs-call-by-value-in-scala-clarification-needed