Treball Dirigit de Competències Transversals

Nil Vilas
8 Gener 2019
Llenguatges de Programació

Índex

Paradigmes de programació

Scala és un llenguatge multiparadigma, el qual implica que no es regeix per un únic paradigma de programació sinó per 2 o més. En aquest cas, Scala, el nombre de paradigmes és 2, ja que és un llenguatge amb Programació Orientada a Objectes (*Object Oriented Programming* o *OOP* en anglès) i amb certes característiques de Programació Funcional.

Programació Orientada a Objectes (OOP)

Classes

Scala és un llenguatge que pren com a base Java, i com a tal, té totes les característiques de la Programació Orientada a Objectes, com bé serien les classes. Un exemple de classe pot ser la representació d'un punt sobre el pla, amb coordenades x i y:

```
class Point(var x: Int, var y: Int) {

  def move(dx: Int, dy: Int): Unit = {
    x = x + dx
    y = y + dy
  }

  override def toString: String =
    s"($x, $y)"
}
```

Exemple de classe

Herència

I com qualsevol llenguatge amb Programació Orientada a Objectes, també presenta herència, que aprofitant l'exemple anterior, heretarem les característiques d'un punt en el pla per tal de conseguir un punt en l'espai tridimensional:

```
class Location(override val xc: Int, override val yc: Int,
   val zc :Int) extends Point(xc, yc){
   var z: Int = zc

   def move(dx: Int, dy: Int, dz: Int) {
      x = x + dx
      y = y + dy
      z = z + dz
   }
}
```

Exemple d'herència

Herència múltiple

Malgrat tenir les similituds anteriors amb Java, Scala permet la herència múltiple mitjançant una característica que anomena Traits. Les classes de Scala poden extendre aquests Traits sense límit, però cal definir el que s'anomena un Trait principal:

```
trait Openable {
  def open() { ... }
}

trait Window extends Openable {
  def open() { ... }
}

trait Door extends Openable {
  def open() { ... }
}

class WindowDoor extends Door with Window {
   ...
}
```

Exemple d'herència múltiple

Com podem veure en el cas anterior, WindowDoor implementa tant Door com Window. Scala tracta l'element posterior a 'extends' com a Trait principal i tots aquells que es trobin després del 'with' com a simples Traits. En cas de col·lisió, es triarà aquella definició que hi hagi al Trait principal.

Programació Funcional

Funcions d'ordre superior i lambda

Com a llenguatge funcional, Scala presenta una sèrie de funcions d'ordre superior. Com a exemple, podríem veure la funció map sobre una seqüència:

```
val salaries = Seq(20000, 70000, 40000)
val newSalaries = salaries.map(x ⇒ x * 2) // List(40000, 140000, 80000)

val salaries = Seq(20000, 70000, 40000)
val newSalaries = salaries.map(_ * 2)
```

Exemple de funció d'ordre superior i funció lambda

A la imatge anterior, podem veure tant la funció map com una funció en forma de lambda (x => x * 2)

Pattern matching

Scala també permet l'ús de pattern matching a les seves funcions, el qual permet fer comprovacions sobre el tipus de la variable d'entrada entre d'altres coses

```
def showNotification(notification: Notification): String = {
  notification match {
    case Email(email, title, _) ⇒
      s"You got an email from $email with title: $title"
    case SMS(number, message) ⇒
      s"You got an SMS from $number! Message: $message"
    case VoiceRecording(name, link) ⇒
      s"you received a Voice Recording from $name! Click the link to hear it: $link"
  }
}
```

Exemple de pattern matching

Compilat i interpretat

Compilació

El compilador de Scala s'anomena scalac, i genera Java Bytecode que pot ser executat si la màquina té la Java Virtual Machine (JVM) instal·lada

Use compiler options with scalac

```
scalac [ <options> ] <source files>
```

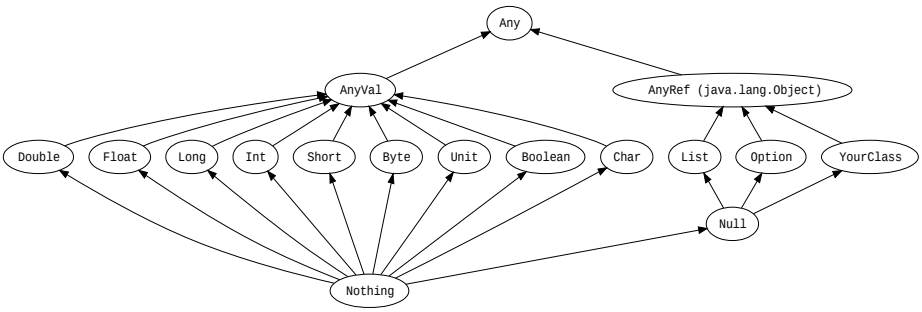
Ús del compilador de Scala

Intèrpret

L'intèrpret de Scala s'anomena REPL i permet executar codi en aquest llenguatge al moment, sense necessitat de compilar.

Tipat

Scala és un llenguatge de programació de tipatge estàtic, però que presenta també inferència de tipus. Els tipus per defecte són els següents, també apareix la categoria on es trobarien les classes creades pel programador notat com a 'YourClass':



Tipus de les variables a Scala

Llenguatges Similars

Java

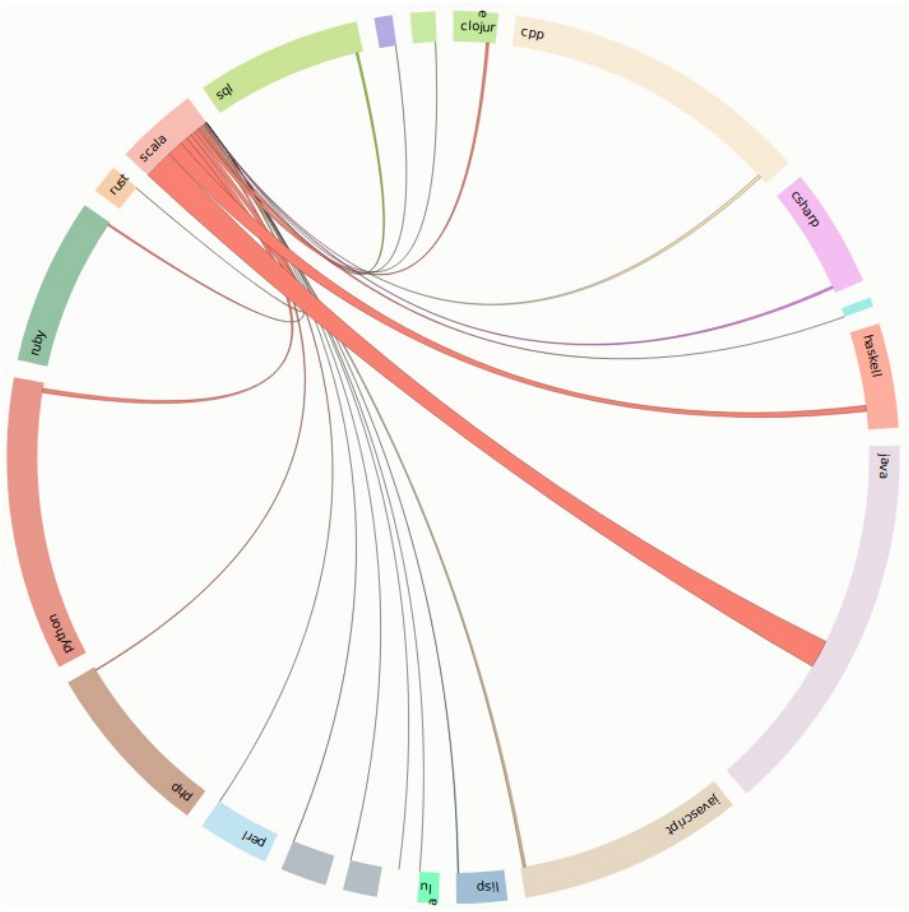
Java no pot faltar en qualsevol comparació que es vulgui fer sobre Scala. Ja que Scala permet ser executat a la JVM i pren com a base aquest llenguatge, tenen moltes similituds. Tot i així, no són exactament iguals, ja que Java és molt més verbós i Scala té moltes propietats afegides.

Haskell

Com a llenguatge funcional, Scala es compara amb altres llenguatges funcionals purs com vindria a ser Haskell. Tenen poca similitud a nivell de codi, però Scala també té inferència de tipus, té funcions d'ordre superior, lambda, etc.

Altres

A continuació s'adjunta una imatge on surten els llenguatges amb els quals s'acostuma a esmenar quan s'escriu sobre Scala a les xarxes socials, útil a l'hora de veure amb quins llenguatges es compara. Quant més gruixuda és la línia que els uneix, s'esmenen en conjunt amb més freqüència.



Llenguatges que es comparen amb Scala

Principals aplicacions

Scala és un llenguatge principalment Orientat a Objectes, i la seva connexió amb Java permet que pugui ser utilitzat a quasi qualsevol dispositiu. A més a més, com que el compilador genera Java Bytecode, totes les llibreries de Java poden ser usades en conjunt amb Scala, així que té un munt de recursos a la seva disposició. Per aquests motius, Scala és un bon llenguatge per a crear codis escalables que es beneficiin de la Programació Orientada a Objectes, així com petits projectes, el seu propòsit és general. A comparació amb Java, la seva menor verbositat permet tenir més codi en menys temps, pel qual pot ser usat en petits prototips amb OOP, ja que després pot ser utilitzat per a acabar el projecte a diferència d'un llenguatge de scripting.

Descripció de les fonts

Al tractar-se d'un llenguatge de programació relativament nou, la majoria de la documentació es troba a internet. La documentació oficial proporcionada pels desenvolupadors del llenguatge a la seva pàgina és força detallada i permet aprendre el llenguatge fàcilment. Malgrat l'anterior, també cal contrastar la informació amb la opinió de la gent que l'ha utilitzat i les dificultats que ha tingut a l'hora de desenvolupar codi en aquest llenguatge, motiu pel qual també hi ha pàgines a la bibliografia que no formen part de la documentació oficial. També és molt útil l'intèrpret on-line de Scala, que permet escriure codi i executar-lo inmediatament per a provar petites funcionalitats del llenguatge. Totes les fotografies de codi s'han tret d'aquest entorn.

Bibliografia

· Anònim. Scala on-line interpreter. Scastie. https://scastie.scala-lang.org/

· Anònim. Scala documentation. Scala-lang docs. https://docs.scala-lang.org/

· TutorialsPoint. Scala Tutorial. Scala – Classes & Objects. https://www.tutorialspoint.com/scala/scala_classes_objects.htm

· Nicolas Fränkel, 9 Juliol 2017. A Java Geek. Scala vs Kotlin: Multiple Inheritance and the Diamond problem. https://blog.frankel.ch/scala-vs-kotlin/4/

· @theotown, 13 Juliol 2015. Lightbend. How Scala compares with 20 other programming languages according to Reddit analysis. https://www.lightbend.com/blog/how-scala-compares-20-programming-languages-reddit-analysis