introduction to SCALA

FEDERICO RAMPAZZO

HI@FRAMP.ME

SCALA

is OBJECT ORIENTED

is FUNCTIONAL

is STATICALLY TYPED

is STRONGLY TYPED

has TYPE INFERENCE

can be LAZY

can be PURE

can use JVM CLASSES

SETUP

Download: http://scala-lang.org/
Or use your package manager

Compile a program

Run it as a script

Interactive Shell

```
% wget http://scala-lang.org/files/archive/scala-2.10.3.tgz
# pacman -S scala sbt

object Wat {
  def main(args: Array[String]) {
    println("WAT")
  }
}
% scalac wat.scala
% scala Wat

#!/usr/bin/env scala
println("SCRIPT")

% scala
Welcome to Scala
Type in expressions to have them evaluated.
Type :help for more information.

scala> println("OHAI")
OHAI
scala>
```

TYPES

Everything is an Object

Immutability: val vs var

Any
Int, Double, Char...
String, List, Array...

Unit and Nothing

Type Inference

```
1.to(2)      #... Range(1, 2)
1 to 2       #... Range(1, 10)

val i = 3    #i: Int = 3
i = 5        #error: reassignment to val i= 5

List(1, 2).head          #Int = 1
"lolcat".length          #Int = 6
s"${2*4}"                #String = 8

()           #
List()       #List[Nothing] = List()

var i = 3    #i: Int = 3
i = 3.14     #error: type mismatch;

val b : Boolean = true    #b: Boolean = true
```

```
"".+(5)      #String = 5
"" + 5       #String = 5

var i = 3    #i: Int = 3
i = 5        #i: Int = 5
```

Import

Importing from Java

```
import util.Random
Random.nextInt(10)                          #5

import java.util.{Date, Locale}
import java.text.DateFormat
import java.text.DateFormat._

val now = new Date
val df = getDateInstance(LONG, Locale.ITALY)
println(df format now)                      #29 novembre 2013
```

EXPRESSIONS

Conditionals

```
if (1==1) true                          #true
if (1!=1) false                         #()
if (1==1) true else false               #true
```

List comprehension

```
for (i <- (1 to 10)) yield i*2          #Vector(2, 4, 6, 8...18, 20)
for (i <- (1 to 10) if i%2==0) yield i*2 #Vector(4, 8, 12, 16, 20)
```

Loops

```
var x = 1; while (x < 1000) x *=2       #x: Int = 1024
```

Case

```
val x : Any = "one"
x match {                               #Boolean true
  case 1 => true
  case "one" => true
  case "1" => true
  case _ => false
}
```

FUNCTIONS

Defining functions

```
(x: Int) => x*2                    #Int => Int = <function1>
(() =>Random.nextInt(100))()       #Int = 42

def f(x: Int): Int = {
 val factor = 2
 return factor*x
}
```

Using Higher Order Functions

```
"UPPERCASE".map(x => x toLower)    #String = uppercase
"UPPERCASE" map(_ toLower)         #String = uppercase
List(1,2) foreach (println _)      #1 2
List(1,2) map f                    #List[Int] = List(2, 4)

def debug (x: String, y: String) = {
 println(x);
 println(y);
 x+y
}
List("A", "B", "C").reduceLeft(debug)   #A B AB C
                                        #String = ABC
```

Closure

```
def hello() : ()=>String = {
  val name = "Jessica"
  () => "Hi, " + name
}
val JessicaGreeting = hello()                    #() => String
JessicaGreeting()                                #Hi, Jessica
```

Laziness

```
val a = { println("Right now"); "a"}
Right now
String = a
```

```
lazy val b = { println("Maybe later"); "b"}
b: String = <lazy>
b
Maybe later
String = b
```

Currying

```
def hello(name: String) : () =>
  String = {
    () => "Hi, " + name
  }
hello("Jessica")                                 #() => String
hello("Jessica")()                               #Hi, Jessica
```

Function composition

```
val x2 = (x: Int) => x*2
val x3 = (x: Int) => x*3

List(1,2,3,4,5) map x2                  #List(2, 4, 6, 8, 10)
List(1,2,3,4,5) map x3                  #List(3, 6, 9, 12, 15)
List(1,2,3,4,5) map (x2 andThen x3)     #List(6, 12, 18, 24, 30)
```

Futures and Promises

```
import scala.concurrent._
import ExecutionContext.Implicits.global

val p = promise[Unit]
val f = p.future map { _ =>  println("Working"); Thread.sleep
(1000) }
f onComplete { _ => println("DONE") }
p.success()

#Working
#... [after 1s]
#DONE
```

stay tuned for new slides about

OOP IN SCALA
ACTOR PATTERN
AKKA
SPRAY.IO

FEDERICO RAMPAZZO

HI@FRAMP.ME