



Dive into Scala

# Joost den Boer

*Freelancer / Contractor*

email : [jdboer@diversit.eu](mailto:jdboer@diversit.eu)

blog : <http://www.diversit.eu>

twitter : [@diversit](https://twitter.com/diversit)

code : <https://bitbucket.org/diversit/dive-into-scala-class>





1996



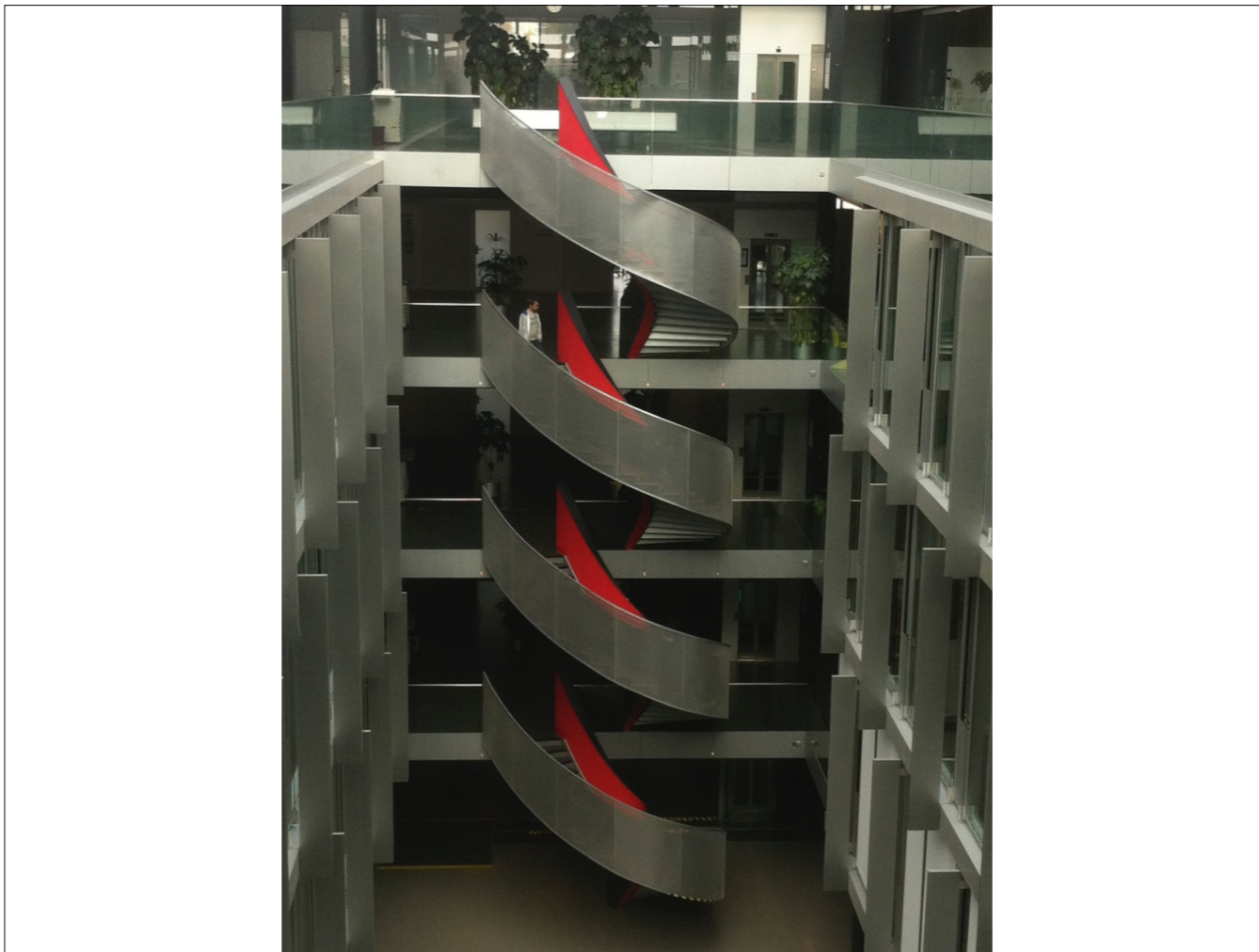
Jaar van verkiezingen Bill Clinton, film Independence Day, Ajax verliest Champions League final van Juventus, Page's (Google) first webcrawler.



# Scala



2011: Tsunami Japan, end of space shuttle program  
Does anyone know why this is the logo of Scala?



Stairs @ EPFL in Lausanne, home of Scala.

## Resources

- [Typesafe.com](http://Typesafe.com) (Scala, docs, Activator)
- [Parleys.com](http://Parleys.com) (presentations, courses)
- [Twitter Scala School](#)
- [Cake Solutions' Week-in-Scala](#)
- [Coursera courses](#)
- [Daniel Westheide's "Neophyte's Guide to Scala"](#)

Some good resources for learning Scala.

My own Scala presentation on [slides.com](http://slides.com) (via LinkedIn profile).



What do YOU want to learn today?

## Agenda

- Create project from scratch
- Import into IDE
- Hello World
- LogParser
- RestAPI



Learn Scala via code.

Plenty stuff to do.

Spare: UDP sender / receiver



## Topics

Traits, regular expressions, pattern matching,  
case classes, currying, singletons, collection api,  
call-by-name, box types, string interpolation, tuples



While coding, these topics will pass by

## Some basics

- Everything is an object (extends Any)
- Every operation is a method
- Every statement is an expression
- ‘;’ ‘.’ and ‘return’ are optional
- Unchecked (Runtime) exception



Binary compatible with Java

Functions are objects and can be passed around.

True singleton objects

Methods in methods

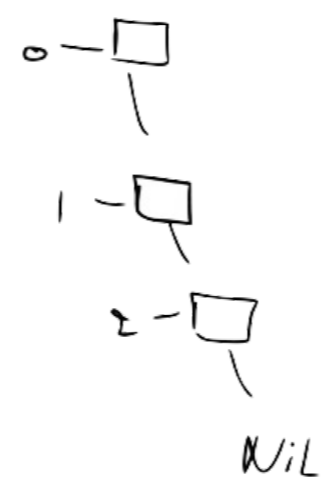
Support for Java's checked exceptions.

# Let's code!

```
case class Branch(left: Tree, right: Tree) extends Tree
case class Leaf(x: Int) extends Tree
val tree1 = Branch(Branch(Leaf(1), Leaf(2)), Leaf(3))
def sumLeaves(t: Tree): Int = t match {
  case Branch(l, r) => sumLeaves(l) + sumLeaves(r)
  case Leaf(x) => x
}
```



Scala List



List(0,1,2)

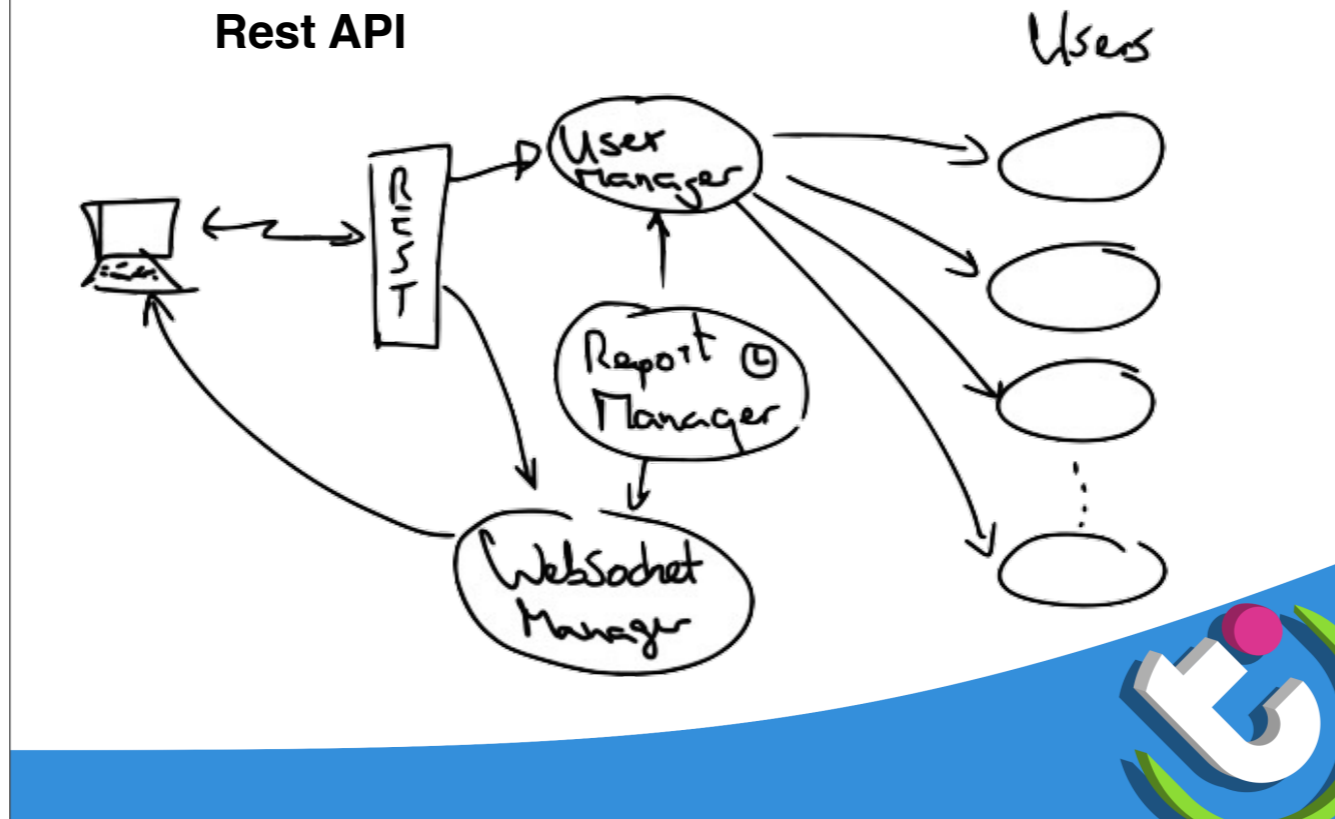
List(1,2)

List(2)

List()

$l = 0 :: 1 :: 2 :: Nil$





Overview Rest app implementation using Akka actors.

Goal: count tags per user and present summaries for all and per user.

User : counts tags for single user

UserManager: manages users and forwards messages to correct user.

WebSocketManager : maintains list of sockets. Can forwards messages to sockets.

ReportManager : time-based, asks UserManager for summary, transforms it into ChartData and sends it to websockets.

