Scala boots backend development!
Berlin Linux Tag 2011 - Berlin

Romain PELISSE
Software Engineer Infrastructure at NumberFour AG

April 29, 2011
Romain PELISSE

- **Software Engineer Infrastructure and Open Source Evangelist** at NumberFour AG (2010)
  - Infrastructure (on Amazon EC2) / server-side development (Scala)
  - and also a lot of javascript, bash script, python along with Puppet, MongoDB,...
- Committer PMD and XRadar
- Translation for HgBook
- Teach build technologies and OPP at ESME Sudria
NumberFour AG - helping to run your business

Two words on the company...

• N4 is a technology start-up providing business tools and intelligence to small business (less than 20 employees) by building a revolutionary platform.

• It is the 4th venture of successful entrepreneur Marco Boerries and is staffed with a talented team from different areas (apps, cloud, mobile, erp...)

• And, of course, we’re still looking for even more passionate people to work with us in Berlin!
What is an (web based) application today?

Front end

• *get the feature!* (quick dev. cycle)
• business-oriented: logic must be easy to implement
• "dirty" code - it has to work no matter what

Back end

• stability before all, code must reliable and controller
• robustness, resilience, scalability
• clutter code is death
Java: best of both worlds?

What made the success of Java on the server side?

- is easy to learn
- many frameworks (Struts, JSF,...)
- OPP allowed "complex system design"

State today

- lack of advanced feature
- lack of integration

Once right in the middle, Java became too crude for both sides...
Focus on the backend: the limit of Java

What are the limits of Java today?

- concurrent programming still cumbersome - critical today
- no lambda function leads to code clutter or even worst duplication
- limited type checking - bugprone
- at core Java remains a "simple" language - more code to write to implement high-level behavior
What is Scala?

Scala in a few words...

- Java based language, runs over JVM
- Mixes OOP and ...
- Functionnal Programming (FP)!
- (very) rich syntax (case class, xml, ...)
- Strongly typed - inference
Expressiveness: a big misunderstanding

What expressiveness is (and is not)

• it’s not about readability
• it’s not about conciveness
• it’s about expressing high-level concept
Expressiveness

Some expressiveness comparison

(python)
range(3, -5, -2)

(java)
List<Integer> range =
    new ArrayList<Integer>(4);
for (int i = 3; i > -5; i += 2)
    range.add(3);

(python)
re.sub(myPattern, replStr, myText, myCount)

(perl)
$smyText =~ s/myPattern/replStr/
And so what?

What does expressiveness bring us?

- **high level abstraction** - less code to write to describe a complex algorithm
- **less bugs** - as less code "surface" is exposed to bug
- **increase readability** - less "boilerplate" code to read
- **increase code resusability** - with functionnal programming, less code duplication
- no more dumb debug, more code reading...
Scalability

What is it?

In telecommunications and software engineering, scalability is a desirable property of a system, a network, or a process, which indicates its ability to either handle growing amounts of work in a graceful manner or to be readily enlarged. - Wikipedia (15/02/2010)

- Scalability
  - Give more resources to the system as load or number of user increases
- Difficult to achieve
  - Increased complexity
  - Human brain
  - Sharing is hard
Java Thread Model

- Primitives
  - Threads
  - Volatile
  - Synchronized
    - variable
    - methods
- Java Memory Model
  - atomic operation
  - immutability
  - defense copies
- Threads Pools
- Executor framework
The Scala (functionnal) way: Actors

Actors

- Mailbox
- Non blocking
- Managed resources

In essence, actor are...

runnables with a blocking queue

Actors unify thread based and event based models
Scala and actors for concurrent programming?

• Benefits for sure ...
  • Safe space to think sequentially
  • Messages as sole way to communicate
  • Can fallback on old share data / lock model
  • Typing harness providing even more robustness
  • Java integration

• ... but arguably
  • Actors and messages passing are nothing new
  • Functionnal programming on top of OOP might get you sick
  • Still a new world, lack of “Effective Actors” book...
Is scala right for your backend?

- willingness to learn functionnal program or existing knowledge?
- needs for concurrent programming on the server side?
- integration with existing Java application?
- spread accross algorithm?
- needs for advanced OOP design?
- ...
Questions and answers
Pattern matching makes things safer

```java
String message = "Invalid result";
if ( list.size() == 2 ) {
    if ( list.get(0) == 1 &&
        list.get(1) == 2 )
        message = "Noop (special case)";
    else if ( list.get(0) == 2 &&
              list.get(1) == 4 )
        message = "Valid result";
    } else if ( list.size() > 2 ) {
    if ( list.get(0) == 2 &&
        list.get(1) == 4 )
        message = "Partial valid result";
    }
System.out.println(message);

val message = list match {
  case List( 1, 2 ) =>
    "Noop (special case)"
  case List( 2, 4 ) =>
    "Valid result"
  case List( 2, 4, _ ) =>
    "Partial valid result"
  case _ => "Invalid result"
}
println( message )
```

Author: Alexis Agahi
Other interesting example of pattern matching usage from Ikai Lan
Lists handling makes it efficient

```scala
1 case class Person(firstName : String, lastName : String, age : Int)
2 val persons = List(
3   Person("John", "Doe", 36),
4   Person("Patrick", "Vinz", 17))
5 val (older, younger) =
6     persons partition ( _ .age >= 18 )
```

- Just try to think about the amount of code to get this in Java, would it be clearer?
- See how easy is to understand, even if you don’t understand the syntax!
Image credits 1/2

- http://www.flickr.com/photos/jakecaptive/3205277810/
- http://www.flickr.com/photos/imuttoo/2123301945/
- ...

...
Image credits 2/2

- ...

- **Geek’n’Poke:**
  - How to make a good code review
  - Simply explained NPE
  - Behind the lines
  - That’s code