

SCALA

Yuhan Duan, Hongye Xu, Zhibo Hu



If I had to select a language to use other than Java, it would be Scala.

-James Gosling,
Creator of Java

None other than Scala can be seen as being 'replacement of Java', and the momentum behind Scala is now unquestionable.

-Charles Nutter,
Co-creator of Ruby

If I would have seen 'Programming in Scala' book back in 2003, I'd probably have never created Groovy.

-James Strachan,
Creator of Groovy

We have found that Scala has enabled us to deliver things faster with less code. It's reinvigorated the team.

-Graham Tackley,
The Guardian

Introduction

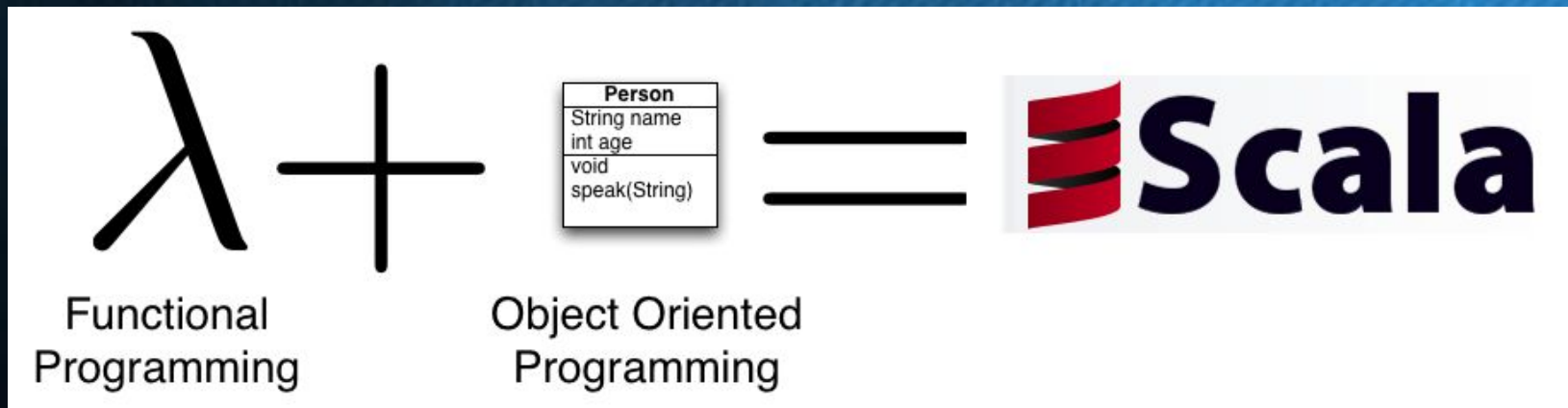
- History and Background
- Scala vs. Java
- Apache Spark : Scala vs. Python
- Applications
- OOP vs. FP
- Special Syntax of Scala
- Data types in Scala
- Special features
- Demo

Timeline



What is Scala?

- Scala is the abbreviation of the word Scalable Language.
- Scala is a modern multi-paradigm programming language that is a combination of object-oriented(OOP) and functional programming(FP).
- Scala is a strong static type of programming language and is influenced by the Java programming language.



Comparison Between Scala and Java

Java	Scala
Complex syntax	Simple syntax
Rewriting is needed	Rewriting is not required
Statically-typed	Statically-typed(but feels dynamic)
No assurance of bug-free codes	Assurance of lesser defects

Java vs Scala WordCount

```
public class WordCountJava {  
    public static void main(String[] args) {  
        StringTokenizer st  
            = new StringTokenizer(args[0]);  
        Map<String, Integer> map =  
            new HashMap<String, Integer>();  
        while (st.hasMoreTokens()) {  
            String word = st.nextToken();  
            Integer count = map.get(word);  
            if (count == null)  
                map.put(word, 1);  
            else  
                map.put(word, count + 1);  
        }  
        System.out.println(map);  
    }  
}
```

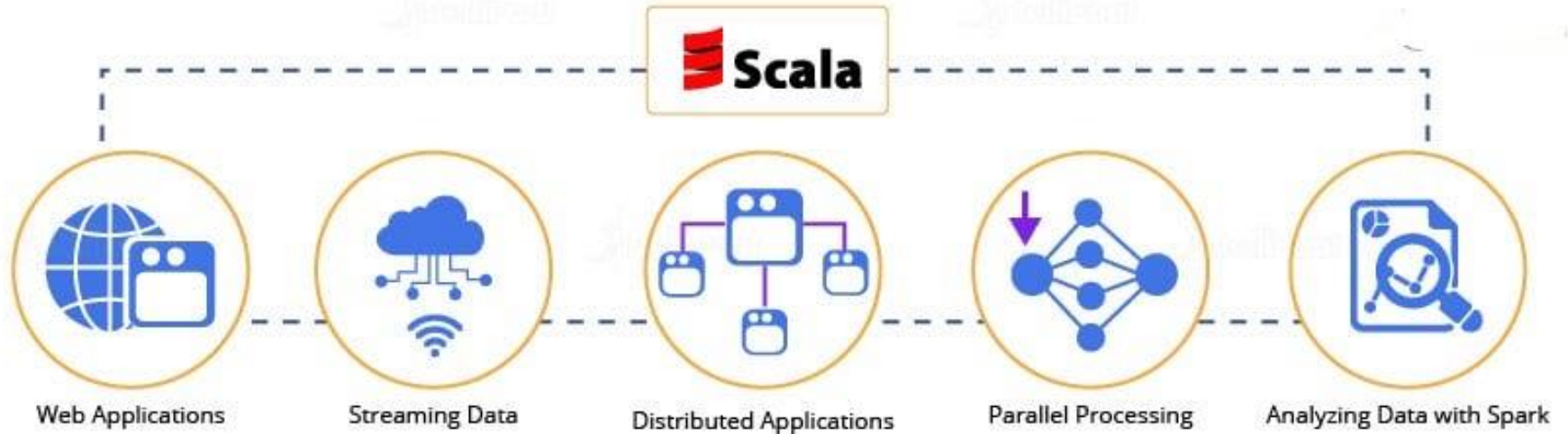
```
> runMain WordCountJava "a b a c a b"  
[info] Running WordCountJava a b a c a b  
{a=3, b=2, c=1}
```



```
object WordCountScala extends App {  
    println(  
        args(0)  
        .split(" ")  
        .groupBy(x => x)  
        .map(t => t._1 -> t._2.length))  
}
```

```
> runMain WordCountScala "a b a c a b"  
[info] Running WordCountScala a b a c a b  
Map(b -> 2, a -> 3, c -> 1)
```

Applications



Apache Spark : Scala vs. Python

- Spark is written in Scala as it can be quite fast because it's statically typed and it compiles in a known way to the JVM.
- Spark has API's for Scala, Python, Java and R but the popularly used languages are the former two. Java does not support Read-Evaluate-Print-Loop, and R is not a general purpose language.
- Scala and Python are both easy to program and help data experts get productive fast. Python is usually the second favourite language for Apache Spark, as Scala was there first.

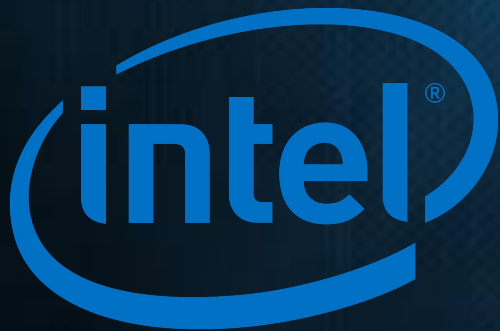
Comparison points	Scala	Python	Java
Performance	Faster (about 10x faster than Python)	Slower	Faster
Learning Curve	Steep learning curve than Java & Python	Easiest	Easier than Scala Tougher than Python
Usage	Spark Native	Data Engineering/ Machine Learning/ Data Visualization	Data Engineering/ Machine Learning/ Data Visualization
Concurrency	Support Concurrency	Does not Support Concurrency	Support Concurrency
Type Safety	Statically typed (except for Spark 2.0 Data frames)	Dynamically Typed	Statically typed
Maturated machine learning libraries and Visualization Libraries availability/ Support	Limited	Excellect	Limited
Web Notebooks Support	Apache Zeppelin Notebook Support	Jupyter Notebook Support	Ijava Kernel in Jupyter Notebook

Drawbacks / Downsides of Scala:

- Scala is complex to learn due to the functional nature of language.
- Steep learning curve.
- Lack of matured machine learning languages.

Who use Scala?

Walmart 



Popularity of Scala

- Twitter have announced that it had switched large portions of its backend from Ruby to Scala and intended to convert the rest.
- Apple Inc. uses Scala in certain teams, along with Java and the Play framework.
- The New York Times revealed in 2014 that its internal content management system Blackbeard is built using Scala, Akka and Play Framework.
- There are teams within Google that use Scala, mostly due to acquisitions such as Firebase and Nest.
- The Walmart Canada Uses Scala for their back end platform.

Installment

1. Check for Spark version

<http://spark.apache.org/docs/latest/>

Downloading

Get Spark from the [downloads page](#) of the project website. This documentation is for Spark version 2.4.5. Spark uses Hadoop's client libraries for HDFS and YARN. Downloads are pre-packaged for a handful of popular Hadoop versions. Users can also download a "Hadoop free" binary and run Spark with any Hadoop version [by augmenting Spark's classpath](#). Scala and Java users can include Spark in their projects using its Maven coordinates and in the future Python users can also install Spark from PyPI.

If you'd like to build Spark from source, visit [Building Spark](#).

Spark runs on both Windows and UNIX-like systems (e.g. Linux, Mac OS). It's easy to run locally on one machine — all you need is to have java installed on your system PATH, or the JAVA_HOME environment variable pointing to a Java installation.

Spark runs on Java 8, Python 2.7+/3.4+ and R 3.1+. For the Scala API, Spark 2.4.5 uses Scala 2.12. You will need to use a compatible Scala version (2.12.x).

Note that support for Java 7, Python 2.6 and old Hadoop versions before 2.6.5 were removed as of Spark 2.2.0. Support for Scala 2.10 was removed as of 2.3.0. Support for Scala 2.11 is deprecated as of Spark 2.4.1 and will be removed in Spark 3.0.

2. Download the proper version of Scala at <http://www.scala-lang.org/download/all.html>

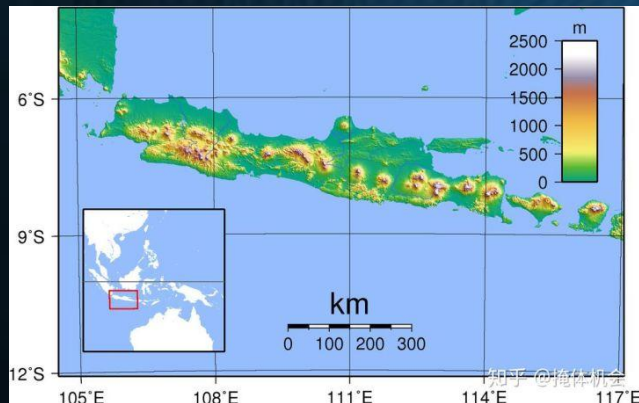
3. IntelliJ IDEA <http://www.jetbrains.com/idea/download/>

Why the name is Scala?

JAVA: it's name from an island that famous for coffee beans.

Python: in memory of a comedy troupe: monty python

Scala: simply because it is scalable!



Idea behind Scala and why it is scalable

Combination of OOP and FP

There are some languages can support both of OOP and FP, but none can fuse them so well as Scala does.

(Python and Java can not support tail call optimization for example)

Let's get some basic idea of FP and OOP based on some examples.

OOP vs FP

```
1 class a(object):  
2     def __inti__(self,number):  
3         self.number=number  
4     def increase(self):  
5         self.number++|
```

OOP:

All the programs need some form of structure.

(everything as an object)

```
1 def increase(a:int):  
2     return a+1
```

FP:

functions are 1st class values. (function can be passed or returned as parameters of others)

All functions should have no side-effect. (No value is changed and each function's return value only rely on it's input.)

OOP

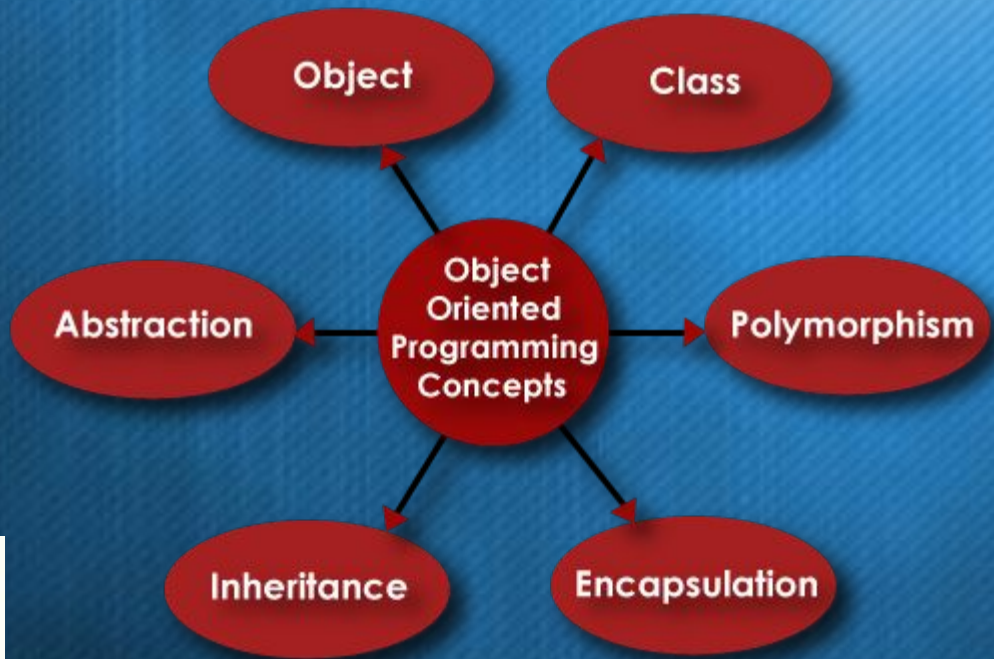
Scala is a pure object-oriented programming language (purer than Java) with every value being an object and every operation is a method call. So, + is not just an operator in Scala, it is a method which is a part of a much larger class of methods and data.

It's like in mathematics, sum can only apply to members of an Abelian group.

Also, methods in Scala can be used as if it is a operator.

```
scala> 5 times println("HI")
```

```
HI  
HI  
HI  
HI  
HI
```



Functions as 1st class values

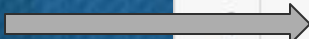
In programming languages, normal functions are passed values like strings and integers. In a functional programming language, functions hold the same status as integers and strings, hence, they can be **passed as values to other functions** and can also be **returned as results of a function**. Furthermore, like a variable, you can also **declare a function within another function**.

Thus, we can easily use higher-order functions.

examples: map-reduce, pipeline.

currying(a sequence of functions with one parameter).

It's more logical to know what to do and better for debugging.



```
1 def inc(x):  
2     def incx(y):  
3         return x+y  
4     return incx  
5 inc2=inc(2)  
6 inc5=inc(5)  
7 print(inc2(1)) #output 3  
8 print(inc5(1)) #output 6
```

3

6

All functions have no side effects

In FP, all the function just return a value and there is **no change on any variable(in place)**. This make sure functions keep **the same output** once it inputs are determined.

Outside FP, there might be some global variables that can lead the function into different ways.

In Scala, everything is immutable. variables are immutable by default; once assigned a value they cannot be modified by you or any other programmer. Default immutability is one of Scala's strongest features. It is primarily used for concurrency control which manages **concurrent (simultaneous) operations** so that they do not conflict with each other. Also it would be easier for programer to do a cell test.

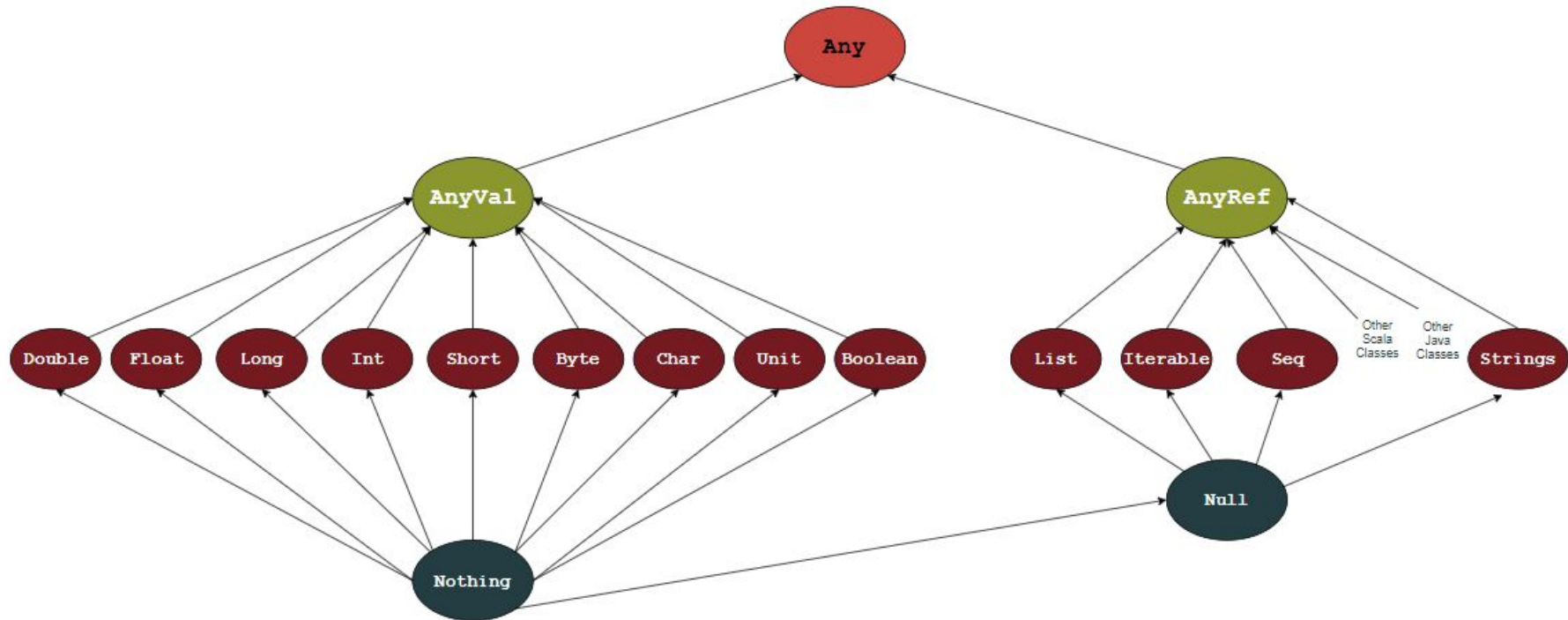
That's the main reason why spark and scala is so related, not only because spark is written in scala(maybe that's why it it written in scala).

Some special syntax of Scala(compared with Java)

Scala is a java-like language, most of its syntax is similar.

- Variables of type “val” are immutable variables; once they are initialized, they can never be reassigned.(all parameters inside function must be val)
- Variables of type “var” are mutable variables; they can be reassigned throughout their lifetime as long as they are valid.
- void is unit in scala(but it still return something)
- not required declaration of type, Scala can “guess” it for you.
- return the last line of function, return value is always required
- Implicit: conversion, class and parameter. for example long to int. default parameter
- semicolon (;) is optional in scala

Data types in Scala



Scala's special features

Features	explanation	Benifits
Immuntability	values can not be changed after it is created	support FP and concurrent operations
Lazy computation	only evaluates an expression when required	Reducing complie time
Type inference	“guess” type of variable (like python)	makes programming tasks easier
Higher-Order Functions	functions as 1st class values	support FP
Traits	A collection of abstract and non-abstract methods. Similar to Java’s interface.	Classes/objects can only inherit from a single class, but multiple traits.
Case Classes	Classes with the added feature of pattern matching	for
String Interpolation	embed variables directly inside a string literal	allowing the creation of strings through data.
Extensive Collection	both mutable and immutable collections	huge library always help(JVM)

```
val name = "James"
println(s"Hello, $name") // Hello, James
```

Sample cases

Thank you !