Introduction to Scala

Principles of Programming Languages

What Is Scala?

The name Scala is derived from **Sca(lable) La(nguage)** and is a multi-paradigm language, incorporating Object-Oriented approaches with functional programming.

What Is Scala?

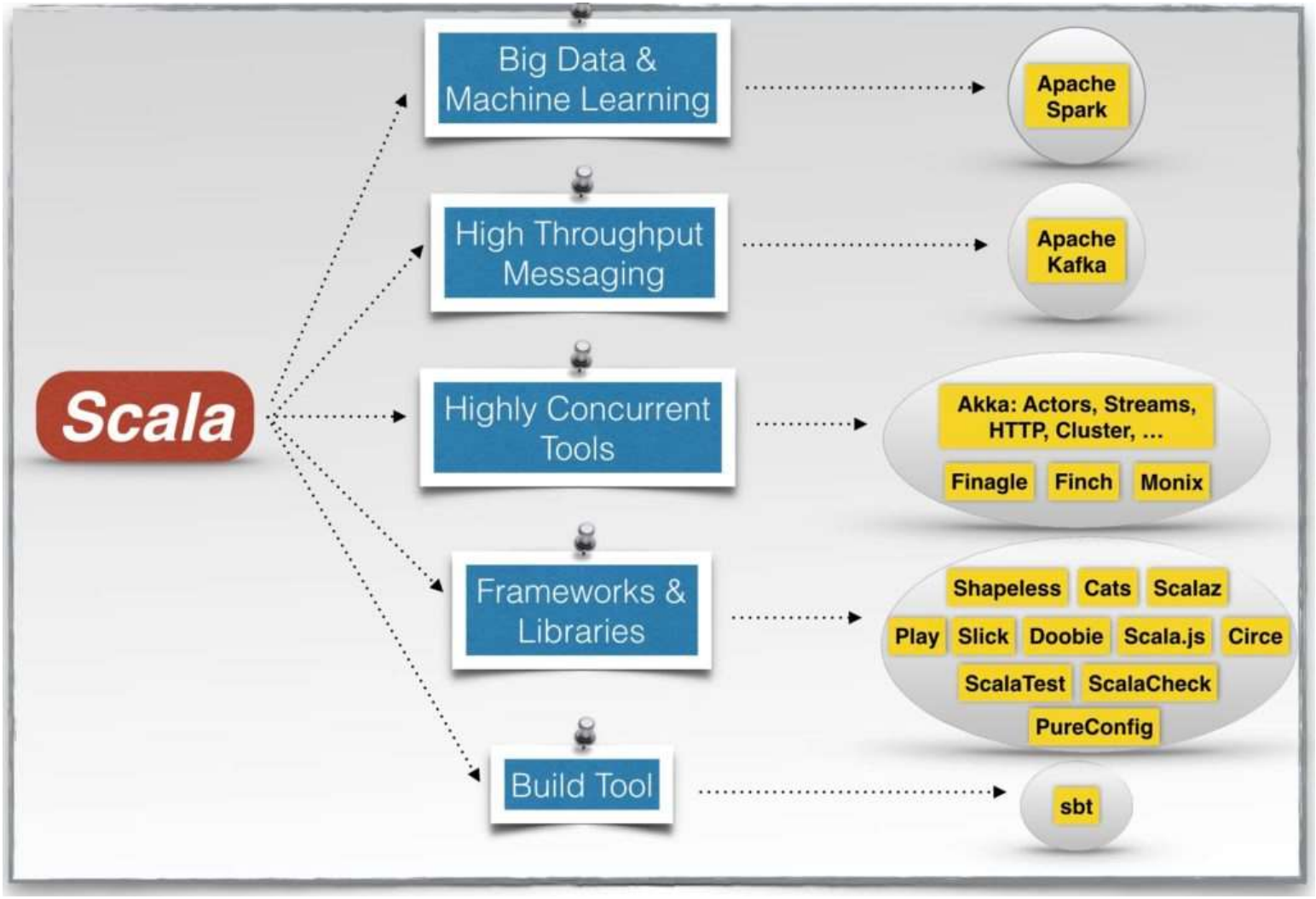
- General purpose, high-level language.
- Created by Martin Odersky at ´Ecole Polytechnique F´ed´erale de Lausanne. First version 2001.
- Uses the Java Virtual Machine (JVM).
- Multi-paradigm, focus on functional and object-oriented programming.
- Expressive, static type system.
- Huge language, built on a small core.

Who uses Scala?

- Twitter
- LinkedIn
- Foursquare
- Coursera
- …
- Used in academia as an alternative to Java

What can you do with Scala?

- Frontend web development with ScalaJS
- Mobile development, both Android Development and IOS – with Scala Native
- Server-side libraries like HTTP4S, Akka-Http, Play Framework
- Internet of things
- Game development
- NLP – Natural Language Processing using a suite of libraries ScalaNLP
- Testing advanced programming techniques such as Functional Programming and in Object-Oriented Programming
- Build highly concurrent communication application using actors a library for the JVM inspired by Erlang
- Use it for machine learning using libraries like Figaro that does probabilistic programming and Apache Spark that


Scala
Big Data & Machine Learning
Apache Spark
High Throughput Messaging
Apache Kafka
Highly Concurrent Tools
Akka: Actors, Streams, HTTP, Cluster, …
Finagle
Finch
Monix
Frameworks & Libraries
Shapeless
Cats
Scalaz
Play
Slick
Doobie
Scala.js
Circe
ScalaTest
ScalaCheck
PureConfig
Build Tool
sbt

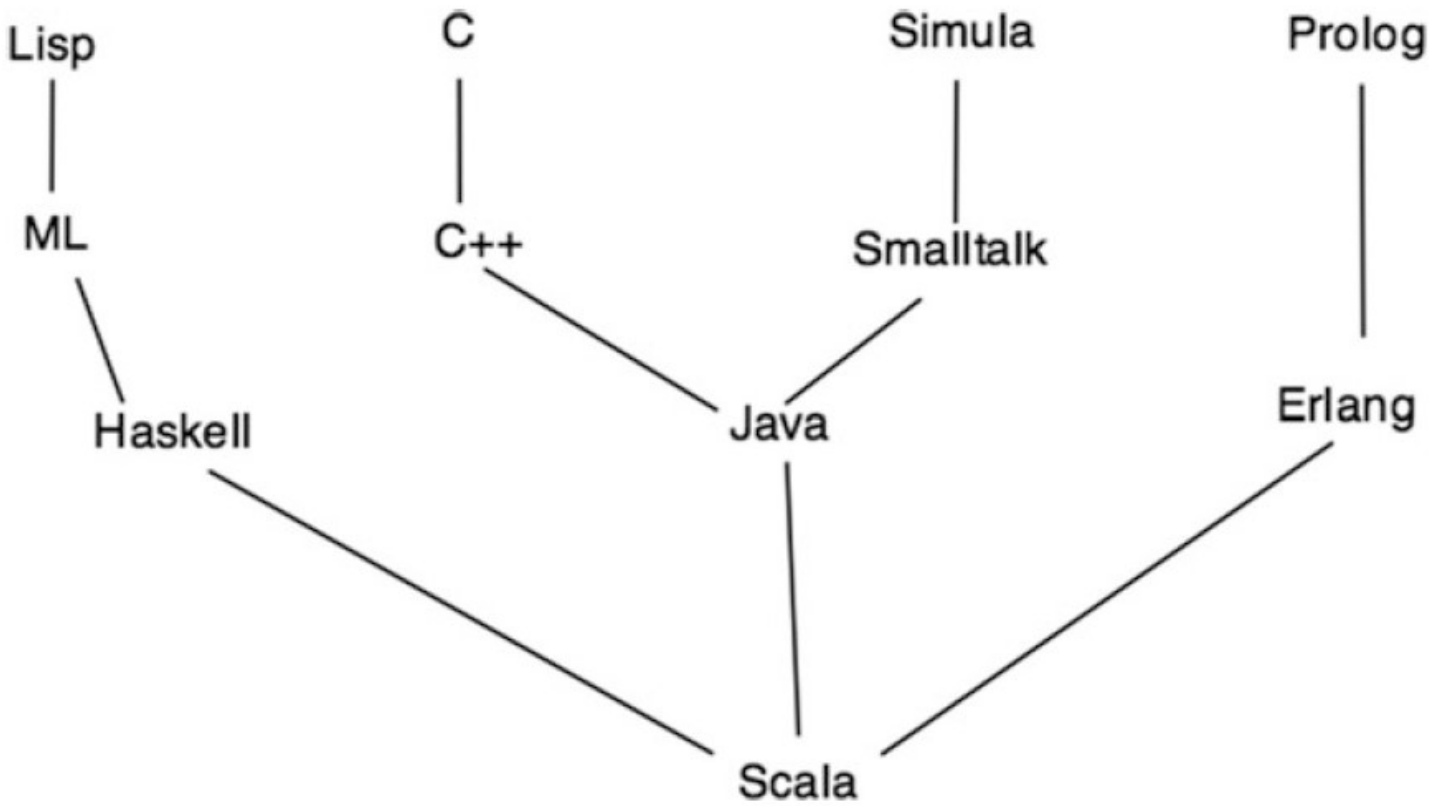
But Java 8 supports functional programming...

- Java 8 has very basic support for lambda expressions.
- There are many more features characterizing a functional language
  - easy-to-use container types (lists etc).
  - lots of syntactic sugar to make programs more concise.
- Scala has many other nice features:
  - A type system that makes sense.
  - Traits.
  - Implicit conversions.
  - Pattern Matching.
  - XML literals, Parser combinators, ...

Why Scala

- It coherently brings together two very powerful programming paradigms that combined can allow very elegant, concise and maintainable systems to be created.
- Some other interesting facts
  - The first is that Scala can be compiled to Java Byte Codes.
  - Includes functional concepts, such as functions as first-class entities in the language, as well as concepts such as Partially Applied functions and Currying which allow new functions to be constructed from existing functions.
  - Uses statically typed variables and constants with type inference used wherever possible to avoid unnecessary repetition.
  - Has interoperability (mostly) with Java.

Scala Genealogy



Java Vs Scala
(A quick comparison)

```
// this is Java
class MyClass {

    private int index;
    private String name;

    public MyClass(int index, String name) {
        this.index = index;
        this.name = name;
    }
}
```

In Scala, you would likely write this instead:

```
class MyClass(index: Int, name: String)
```

Properties of Scala

Scala is compatible

- Scala programs compile to JVM bytecodes.
- Scala code can call Java methods, access Java fields, inherit from Java classes, and implement Java interfaces.
- Similarly, Scala code can also be invoked from Java code.

Scala is concise

- Scala programs tend to be short.
- Scala programmers have reported reductions in number of lines of up to a factor of ten compared to Java.

Properties of Scala

Scala is high-level

- Scala helps you manage complexity by letting you raise the level of abstraction in the interfaces you design and use.
- As an example, imagine you have a String variable name, and you want to find out whether or not that String contains an upper case character.

Java Code

```
boolean nameHasUpperCase = false;
for (int i = 0; i < name.length(); ++i) {
    if (Character.isUpperCase(name.charAt(i))) {
        nameHasUpperCase = true;
        break;
    }
}
```

Whereas in Scala, you could write this:

```
val nameHasUpperCase = name.exists(_.isUpperCase)
```

Properties of Scala

Scala is statically typed

- Scala stands out as a language with a very advanced static type system.
- Static type systems can prove the absence of certain run-time errors*.

Static Type - Documentation

- Static types are program documentation that is checked by the compiler for correctness.

**For instance, they can prove properties like: booleans are never added to integers; private variables are not accessed from outside their class; functions are applied to the right number of arguments; only strings are ever added to a set of strings.*

Summary

- You will need to apply Scala artfully, and that will require some learning and practice.
- If you're coming to Scala from Java, the most challenging aspects of learning Scala may involve Scala's type system (which is richer than Java's) and its support for functional programming.

Next – Use Scala Interpreter