
A Beginner's Guide To Scala
Adam Pridmore
@adampridmore

Who am I

Adam Pridmore

A developer

C++, C#, Java and Scala

A little F# as my introduction to functional programing

My FP Story

Concurrency is hard

Scalability is therefore hard

Creating traditional OO services in C#
Threading is hard (in OO)
- Writing
- Testing
- Debugging
...

Functional programming does not allow the assignment of values to variables

This prevents race conditions

In OO / imperitive concurrency and scale need to be decided upfront.
And it costs more to build

In FP tend not to worry, and it can be added later.

Put off by

You can’t assign values to variables

Lack of program flow control

Like Linq (which is a lot like Streaming API in Java)
But appeared in ~2008

What is Imperative?

Uses flow control statements to alter a programs state

Functions can manipulate state
Functions don't have to return values
Function called twice can have a different effect.

AKA A Turing Machine.
Also know as a modern computer.

Evolution from assembler to machine code to higher level modern languages like Java
Sprinkle Garbage Collection and OO

Loops, if branches, variables changing state
Traditional Java / C#

What is FP

Treats computation as the evaluation of mathematical functions

Functions return values
Output of function only dependent upon inputs to function
Maps a set of values into another possibly different set of values.
No difference between calling twice and calling once and storing the result (aka referential transparency)
Makes inferring what a function does very easy. And easy to refactor with the compile tell you if you are wrong.
Less debugging.
Things tend to work!

Java vs Scala WordCount

```java
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}
```

```scala
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)
```

Scala a lot Java streaming (or C# Linq) very familiar
You can write FP in Java. But it's a lot easier in a language that supports FP (like Scala)

What is  Scala

A language that runs on the JVM

Interoperable with Java Libraries

A multi-paradigm programing language (OO and FP)

Statically typed

Runs on Oracle JVM &' Open JDK, and others?

So you can explore the world of Functional Programming without abandoning OOP as it has both.

Because it runs on the JVM you have a ton of libraries that you can use

Free

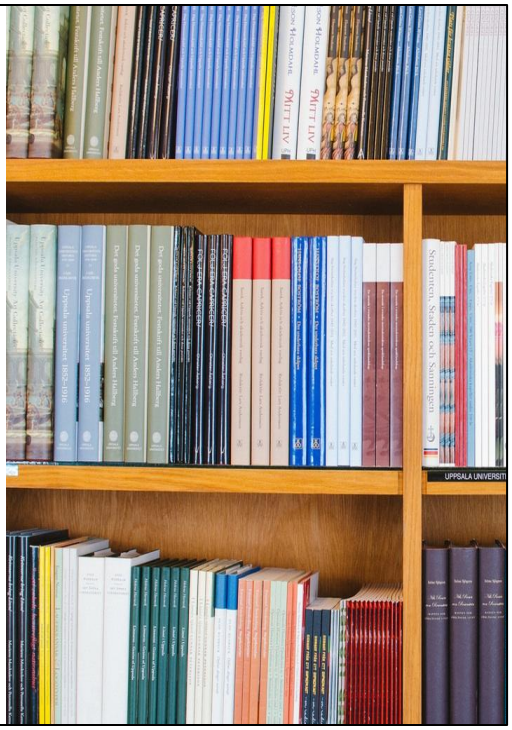
People tend to write FP. Examples and libraries.

History

Version 1.0 Mar-2004

Version 2.0 Mar-2006

Latest 2.13 Jun-2019



Been around a while.

Almost all scala is V2.

Version 3 is arriving later this year. I don’t know what in it!

Why?

- FP is popular with data analytics / big data (Spark, Hadoop)
- Machine Learning / AI
- Mathematics
- Boring old business apps
- Can also be used anywhere you would use Java*

*I think

Shut up and show me some code

```
sbt new scala/scalatest-example.g8
```

https://github.com/adampridmore/scala-talk

Install scala
Create hello world with this command
SBT -> Simple Build Tool (aka Maven / Gradle)

IntelliJ + Scala Plugin (from JetBrains)
Other IDE Available. VSCode + Plug-in / Metals