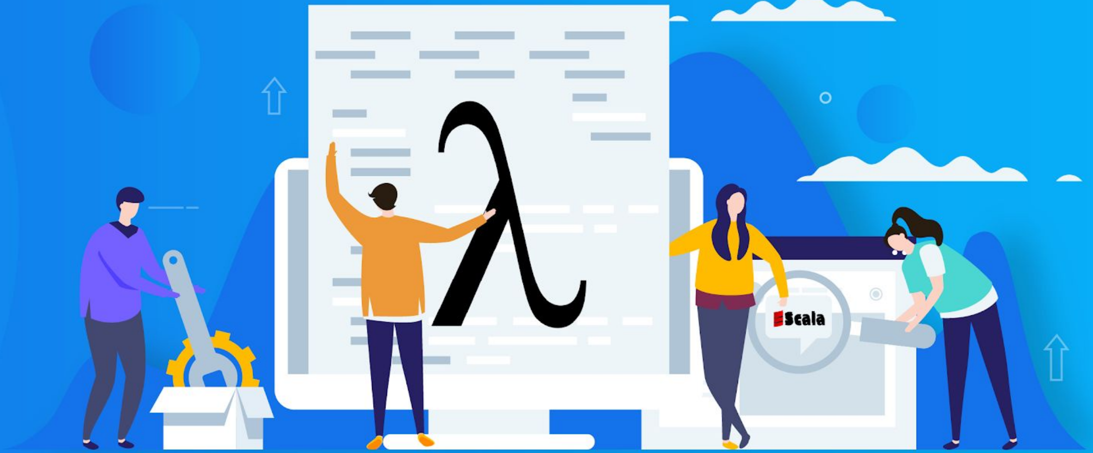
knóldus

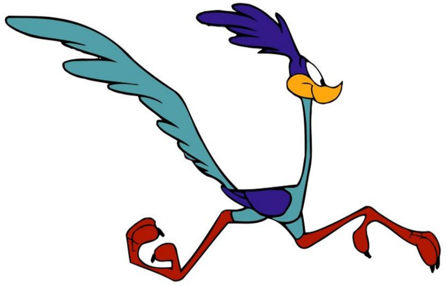
Simplified Scala Monads and Transformation



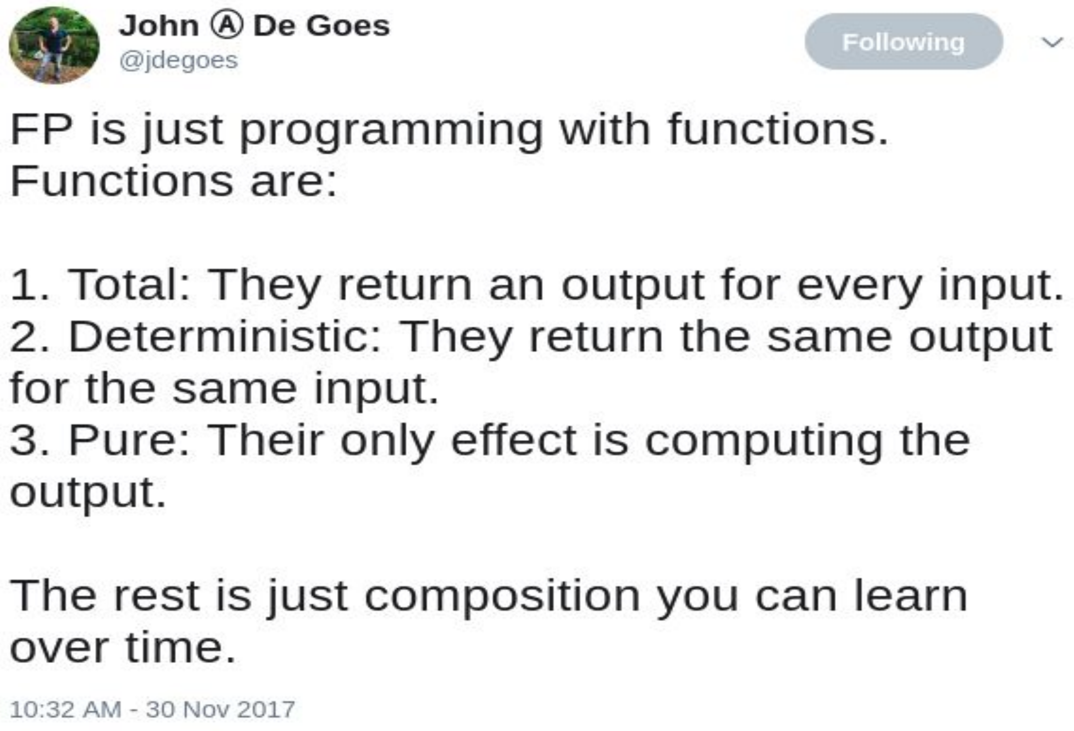
Presented by:
Harmeet Singh
@singh_harmeet13

Today’s Roadmap

- ✓ Brief Of Functional Programming
- ✓ Functions
- ✓ Functions Composition
- ✓ Monads
- ✓ Handle Side Effects Using Monads
- ✓ Monads Transformation



Brief Of Functional Programming



John Ⓐ De Goes
@jdegoes

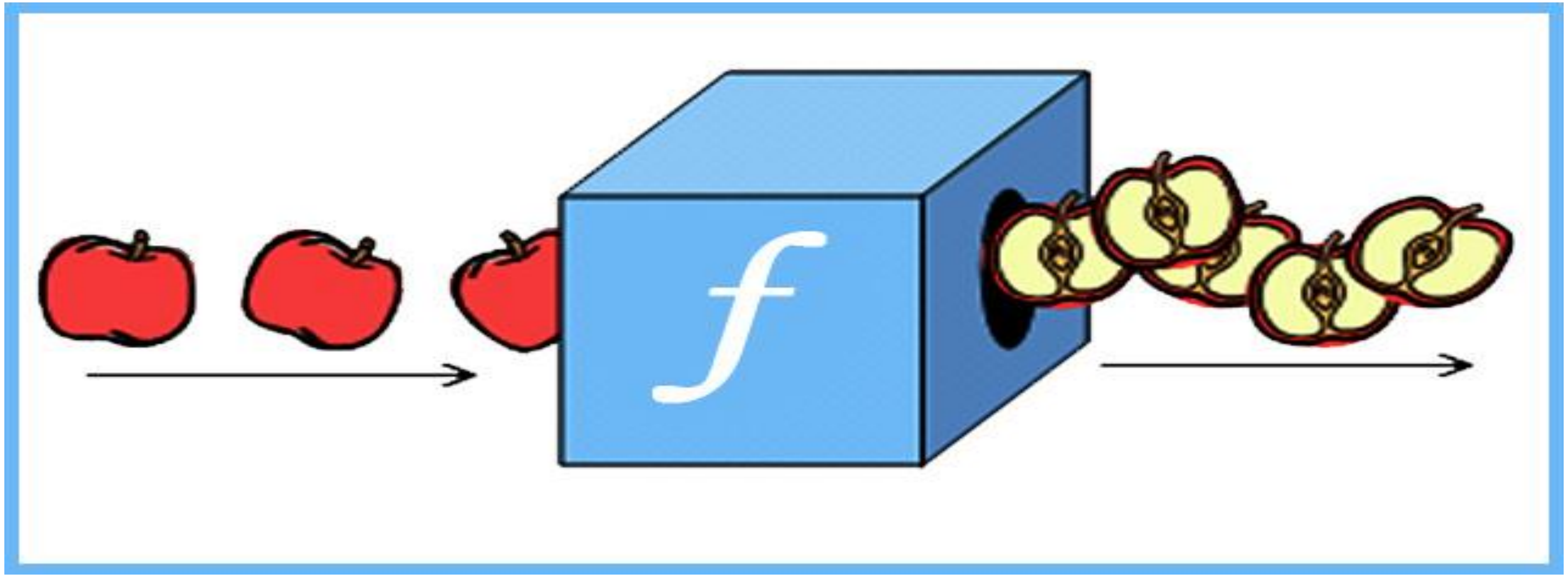
FP is just programming with functions.
Functions are:

1. Total: They return an output for every input.
2. Deterministic: They return the same output for the same input.
3. Pure: Their only effect is computing the output.

The rest is just composition you can learn over time.

10:32 AM - 30 Nov 2017

Functions



Functions: Mathematics

```
f(x) = x + 1

f(x, y) = x + y

f(a, b, c, x) = a * x^2 + b*x + c
```

Functions: Mathematics

Properties:

- Functions are pure.
- Output of the functions depends only on its input.
- Functions have no side effects.
- All values are immutable.
- and more...

Functions: Scala

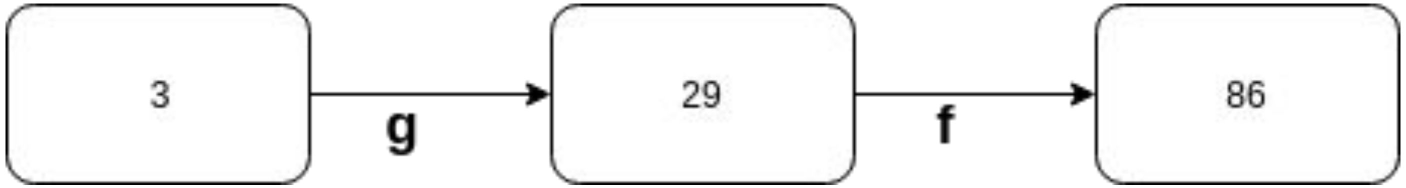
```scala
def f(x: Int) = x + 1

def f(x: Int, y: Int) = x + y

def f(a: Int, b: Int, c: Int, x: Int)
   = a * x*x + b*x + c
```

Function Composition

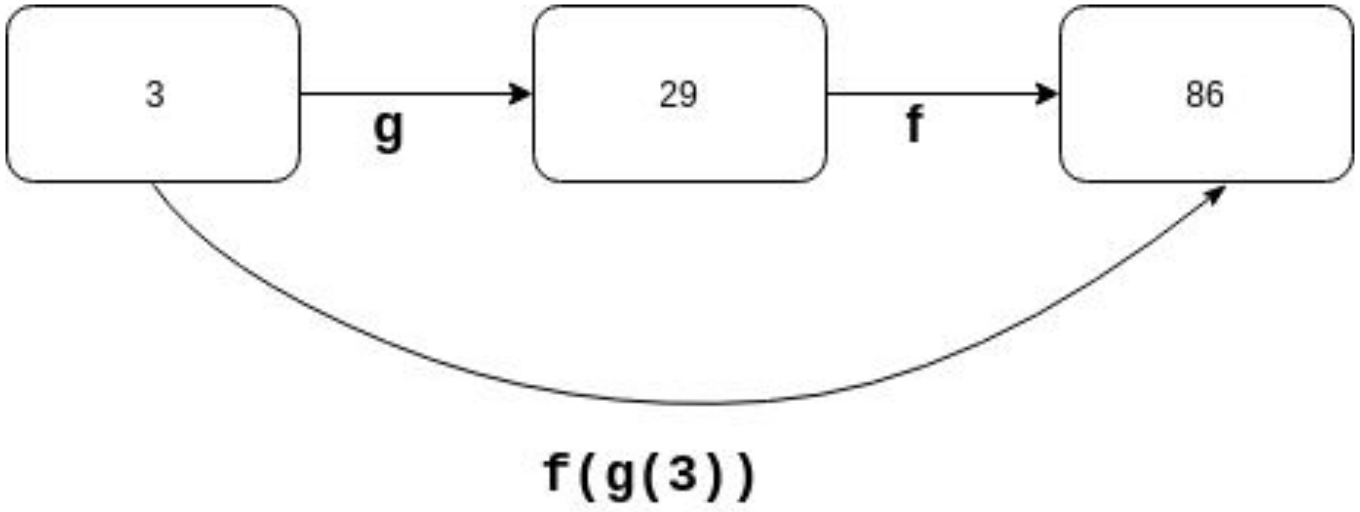
Given two functions, we can combine them in such a way so that the outputs of one function become the inputs of the other.

Source: Khanacademy

Function Composition


$$\begin{tikzcd}\boxed{3} \arrow[r, "g"'] & \boxed{29} \arrow[r, "f"'] & \boxed{86}\end{tikzcd}$$

Function Composition



Function Composition

Monads

- ➔ If a class contains methods signature “map”, “flatMap” and “pure”.
- ➔ If we process some value and requires to pass intermediate results to the next method.
- ➔ If we need to perform some side effects within our applications.
- ➔ and more...

Sample

```
class Something[A] {

   def map[B](f: A => B): Something[B]

   def flatMap[B](f: A => Something[B]):
Something[B]
}
```

Side Effects

- ➔ If methods returns Unit.
- ➔ If methods change the application state.
- ➔ If methods talk with any third party resources like (database, network.. etc)
- ➔ and more....

Monads Transformation

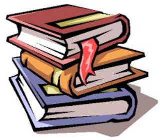
Monad Transformer is a type constructor which takes a monad as an argument and returns a monad as a result. It can be used to compose features encapsulated by monads – such as state, exception handling, and I/O – in a modular way.

Source: Wikipedia

Examples / Slides

https://github.com/knoldus/simplified-scala-monads

References



- ✓ https://en.wikibooks.org/wiki/Algebra/Functions
- ✓ https://blog.buildo.io/monad-transformers-for-the-working-programmer-aa7e981190e7
- ✓ https://www.amazon.com/Functional-Programming-Simplified-Alvin-Alexander/dp/1979788782/ref=sr_1_1?ie=UTF8&qid=1536166154&sr=8-1&keywords=simplified+functional+programming

THANK YOU!