FP: Lift the Fear

From Beginner to Novice

Goal

Help programmers learn and step into the world of Functional Programming.

And not be afraid of those scary terms. (Don’t think this will ever happen)

Have a better way to tackle these terms.

WARNING

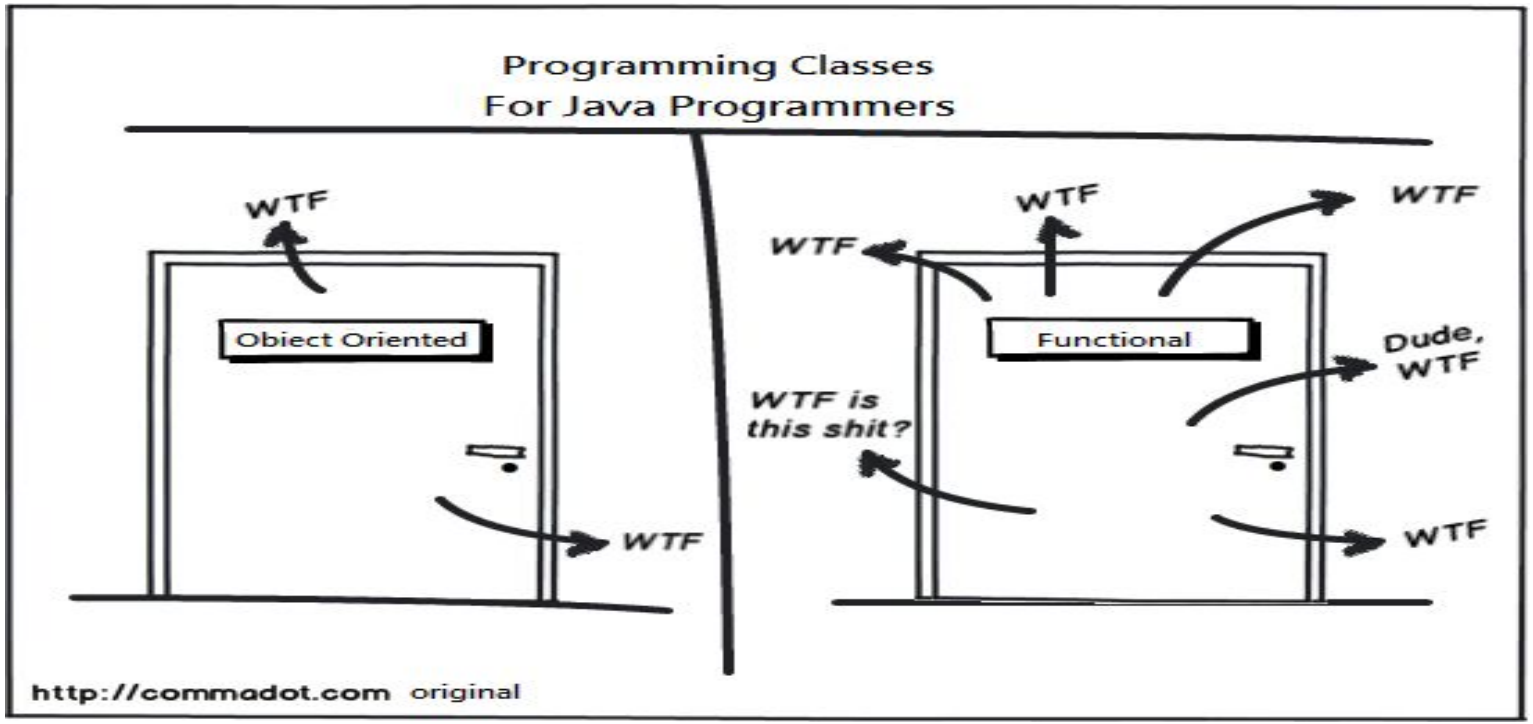
The following presentation contains explicit content, which may dramatically alter the way you view functional programming. There are cases where people who have been contaminated with this content never escapes the ideas and cannot progress any further in the functional programming world.

Viewer discretion is advise.

My Experience

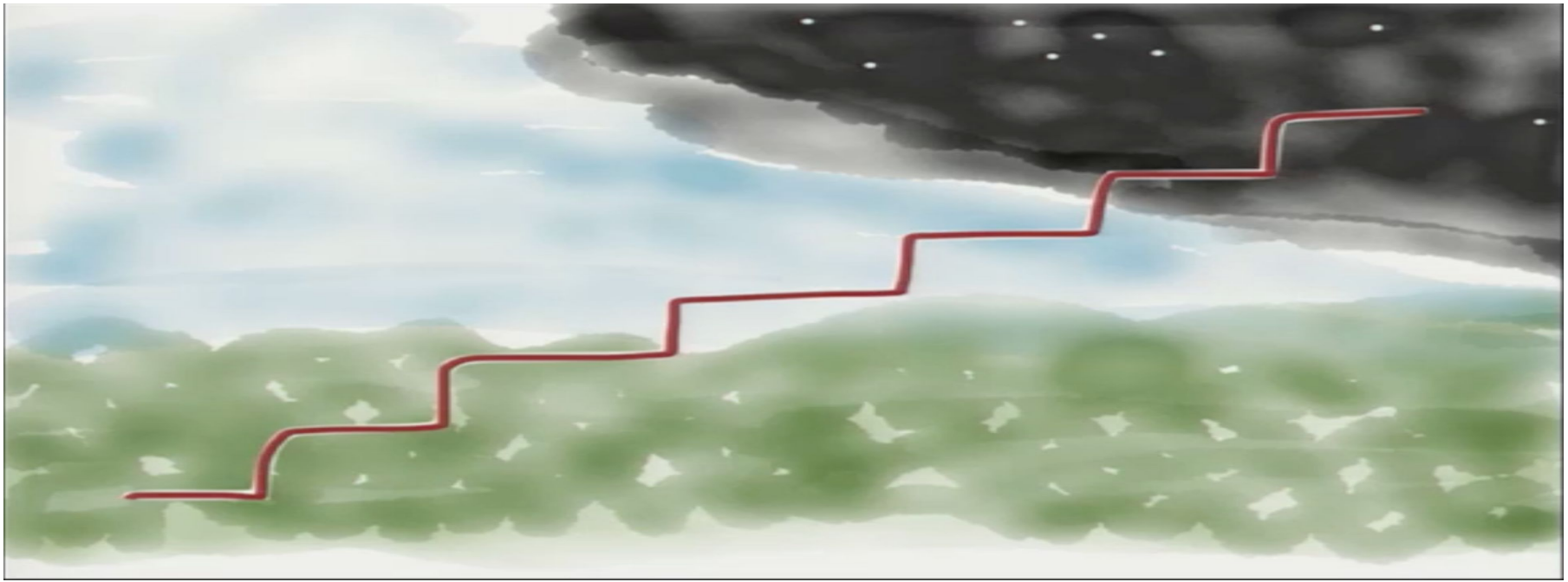
1. Recursion/Tail-Recursion (in first-year university using Scheme)
2. Higher-Order Function/Function as First-class citizen (Javascript call-back)
3. The word “Monad” (Scala)
4. Learning “List”(Option/Future) is a “Monad”
5. Learning “Monad” is just a “Context”
6. ...trapped in the idea for 3-some years and can’t explain why a semi-colon is also a Monad
7. Re-learned that “Monad” is about “Sequential Computation”
8. Finally comfortable with 1 Category Theory term

Audience OS


Programming Classes
For Java Programmers
WTF
Obiect Oriented
WTF
WTF
WTF
WTF
Functional
Dude,
WTF
WTF is
this shit?
WTF
http://commadot.com original

Why is learning FP hard for most people?

The usual learning process for anything.



From: Jessica Kerr https://skillsmatter.com/skillscasts/6483-keynote-scaling-intelligence-moving-ideas-forward

Why is learning FP hard for most people?

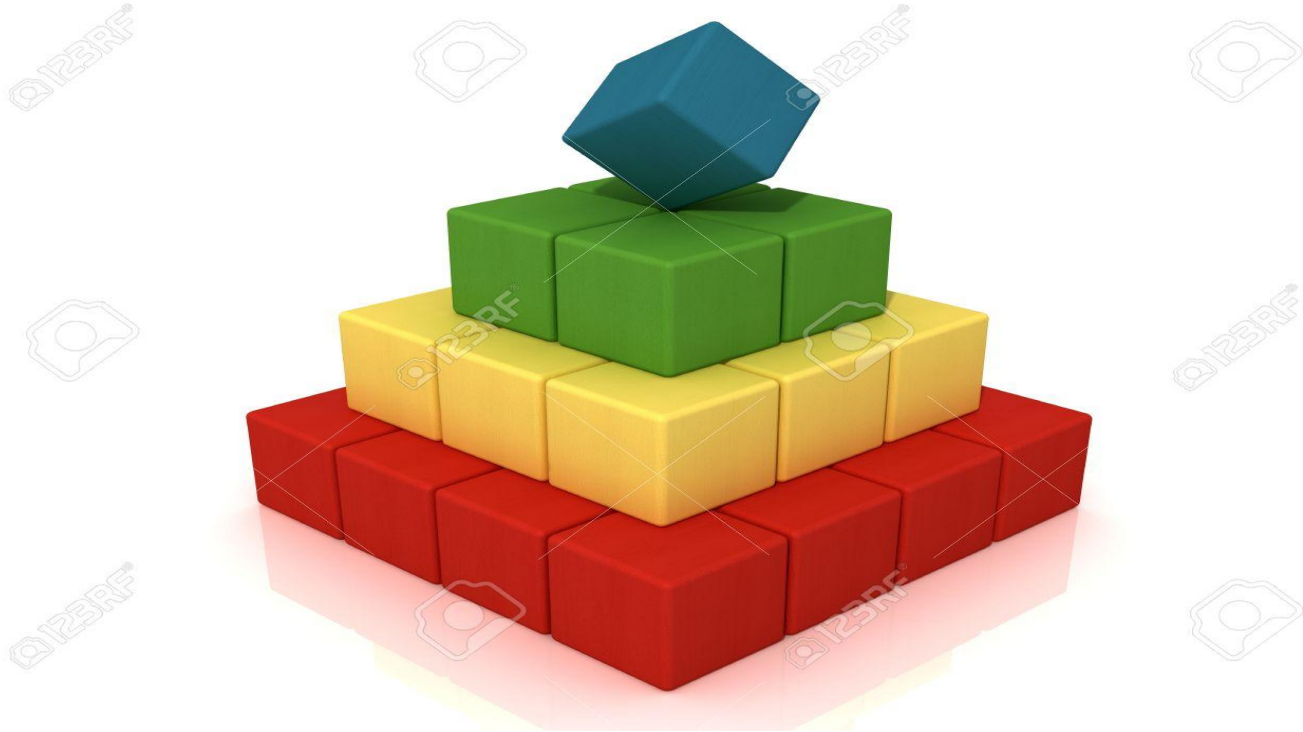
The usual learning process for Functional Programming.

Credit: Jessica Kerr

Why is learning FP hard for most people?

1. Laws/Mathematical Theorems aka Category Theory
2. Abstractions
3. Big Words and weird symbols
    - |+| and |@| and ø
4. Discontinued Learning process
5. Learning the “Basics” is hard, getting the “Basics” into your mind is harder
6. We are not elementary students, but experienced social elites with no time. We learn by apply.
7. We are not smart enough

Abstraction



Abstraction

One Ring to rule them all, One Ring to find them,

One Ring to bring them all and in the darkness bind them

J. R. R. Tolkien's *The Lord of the Rings*

Replace “One Ring” with anything should work.

Abstraction

Beer → Fermented drink → Alcoholic drink → Any

Abstraction (Monad)

- unit
- return
- point
- pure
- lift
- wrap?

Abstraction (Monad)

- flatMap
- feedThrough
- processBy
- shove
- bind
- >>=

Beginner: Functional Programming

Don’t need to go all the way.

Know your problem first.

Start small and work your way up.

Beginner: Functional Programming Concepts

What Functional Programming can do for us?

(for us Average Joes at least)

1. Ease the pain from what's between the tool you use to program and the business logic you're trying to achieve
2. Better at abstraction of any non-business logic related problems
3. Additional language syntax
    a. Just can't live without the good old `for( int i = 0; i < x; i++)` idiom?

Beginner: Functional Programming Concepts

- **First-class and higher-order functions**
  - Higher-order functions are functions that can either take other functions as arguments or return them as results.
- **Pure functions**
  - Pure functions (or expressions) have no side effects (memory or I/O)**(hardest)**
  - http://blog.higher-order.com/blog/2012/09/13/what-purity-is-and-isnt/
  - Immutable data structure
- **Referential transparency (Pure is a dependency)**
  - Functional programs do not have assignment statements, that is, the value of a variable in a functional program never changes once defined.
- **Types**
  - Not Class

Beginner: FP => Category Theory

In Joe’s eyes:

1. Words and symbols that are impossible to comprehend

Actually:

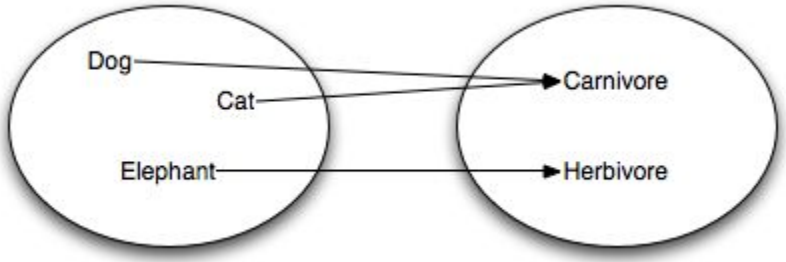
1. Mapping
2. Composition

https://hseeberger.wordpress.com/2010/11/25/introduction-to-category-theory-in-scala/

Beginner: Category Theory - Map

From a “type” map/transform to a “type”

m: A => B (or m: A => A)

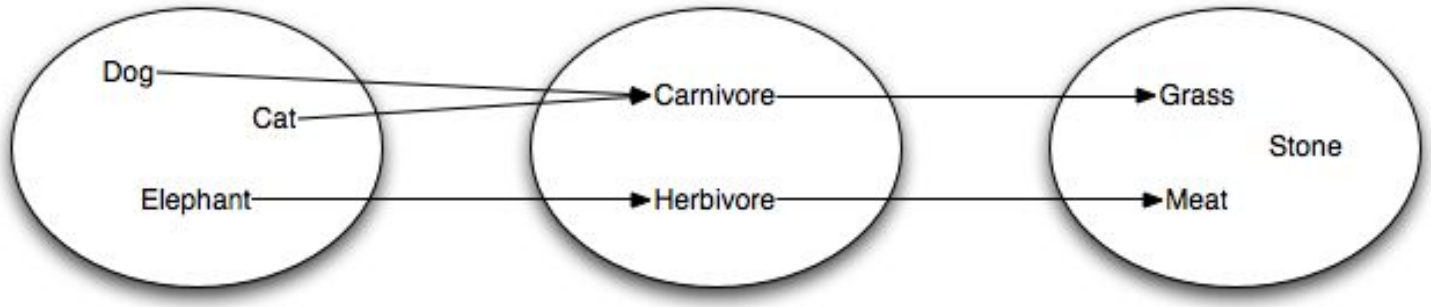


Note: you can see this as a type-constructor for values which takes an A and create an B

Beginner: Category Theory - Composition

Compose/Combine/Append/Concat two “Map” together

m2: B => C *compose* m1: A => B *becomes* m3: A => C



Beginner: Category Theory - Composition

Compose to create complexity (start small and build big)

- f: A => B
- g: B => C
- h: B => A

Creates

- i: A => C in g(f) or g(f(h(f)))
- j: B => B in h(f)

Note: Referential transparency is important

FP for Noobies

Standardized Ladder of Functional Programming

1. Immutability
2. Higher-ordered function
3. map/flatMap
4. Generalize from real life problems you actually encounter with the help of higher-ordered function
5. There’s no need to go all the way

Most General case

Java → Scala

Generalisation Example

Live code

Making it async?

Conclusion

1. as a very generalisation of your everyday problem
2. Start everything with a problem. If it's not a problem for you, then it's probably not worth your time (for now).
    a. Don't go OptionT if you are not even used to Option and other similar types
    b. Unit Test?
3. Start from very simple case and work your way up
    a. Practise generalisation
    b. Don't go GuessTheNumber3.scala if you haven't seen GuessTheNumber1 and GuessTheNumber2. Actually don't even go there if you can't find it better than what you are currently doing.
4. Simply making your program more readable is a reason to go FP

Reference

https://skillsmatter.com/skillscasts/6483-keynote-scaling-intelligence-moving-ideas-forward#video

http://blog.jessitron.com/2013/01/from-imperative-to-data-flow-to.html

http://mandubian.com/2012/08/27/understanding-play2-iteratees-for-normal-humans/

https://stackoverflow.com/questions/7076128/best-way-to-merge-two-maps-and-sum-the-values-of-same-key

https://en.wikipedia.org/wiki/Iteratee

https://www.infoq.com/presentations/functional-pros-cons

http://blog.higher-order.com/blog/2010/10/14/scalaz-tutorial-enumeration-based-io-with-iteratees/

http://loicdescotte.github.io/posts/play-akka-streams-twitter/

https://stackoverflow.com/questions/7746894/are-there-pronounceable-names-for-common-haskell-operators