Let the Types Work for You

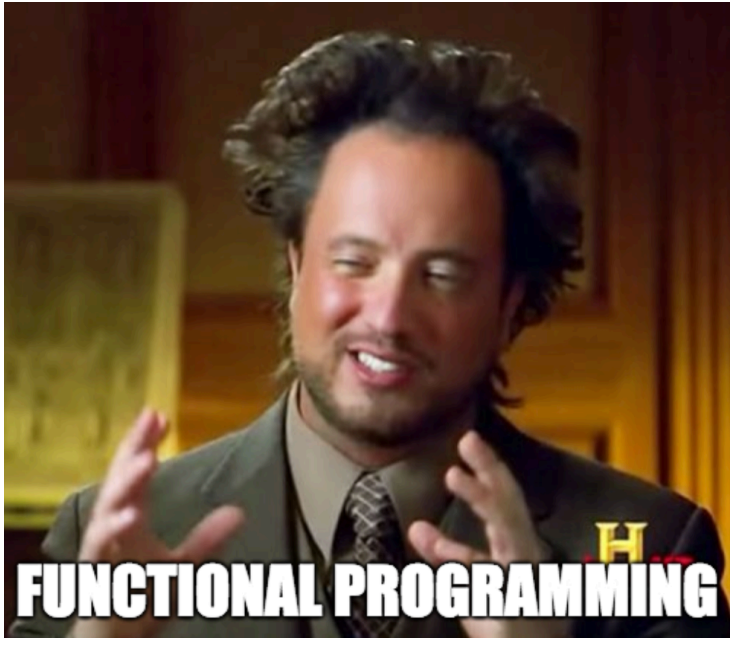
Klarna Konferense

Felix Mulder

August 2018

Agenda

- Functional Programming
- Type systems
- FP + Types == amazing!


FUNCTIONAL PROGRAMMING

“Do you know that feeling of having to hold too many things in your head at once?”

Functional Programming gets rid of that by definition.

Referential Transparency

```
x = 5
y = x + x
z = y + x

// ⟹
z = (5 + 5) + 5
```

- Equational reasoning
- Compositionality

Referential Transparency + Types

==

Refactor All The Things! (without fear)

Game over, OO. Right?

What about the downsides?

- Downsides here

What if you could negate those downsides?

- Smarter inference
- Better compiler messages
- and...

alt-center

Today we’re exploring type-level induction and recursion

What we’re actually doing

Writing a compile-time serializer for data types - with no need for scary runtime reflection.

DISCLAIMER!

Coding time!

Felix’s Conjecture

“By being able to do anything, we can assume nothing”

Constraints Liberate, and Liberties Constrain

```
def foo(i: Int): Int = ???
```

Constraints Liberate, and Liberties Constrain

```
def foo[A](a: A): A = ???
```

Constraints Liberate, and Liberties Constrain

```
def foo[A](a: A): A = a
```

Constraints Liberate, and Liberties Constrain

```
def id[A](a: A): A = a
```

In Closing

- Type level recursion for fun and profit!
- Built a type-level, compile-time JSON serializer

MEME

Thank You!

References

- Constraints Liberate, Liberties Constrain - Runar Bjarnason
- Type Astronaut’s Guide to Shapeless - Dave Gurnell