Let the Types Work for You

Klarna Konferense

Felix Mulder

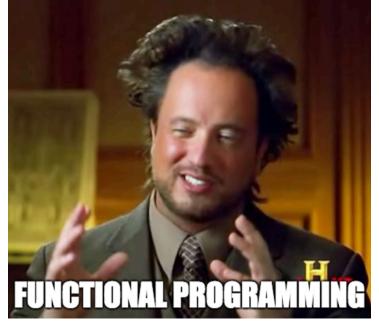
August 2018

@felixmulder 1/24

Agenda

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

@felixmulder 2 / 24



@felixmulder 3/24

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

@felixmulder 4/24

Functional Programming gets rid of that by definition.

@felixmulder 5/24

Referential Transparency

```
x = 5
y = x + x
z = y + x
// \Longrightarrow
z = (5 + 5) + 5
```

- · Equational reasoning
- Compositionality

Referential Transparency + Types

==

Refactor All The Things! (without fear)

@felixmulder 7/24

Game over, OO. Right?

@felixmulder 8 / 24

What about the downsides?

· Downsides here

@felixmulder 9/24

What if you could negate those downsides?

- Smarter inference
- Better compiler messages

· and...

@felixmulder 10 / 24

alt-center

@felixmulder 11/24

Today we're exploring type-level induction and recursion

@felixmulder 12/24

What we're actually doing

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

@felixmulder 13/24

DISCLAIMER!

@felixmulder 14/24

Coding time!

@felixmulder 15/24

Felix's Conjecture

"By being able to do anything, we can assume nothing"

@felixmulder 16 / 24

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

@felixmulder 17 / 24

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

@felixmulder 18/24

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

@felixmulder 19/24

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

@felixmulder 20/24

In Closing

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

@felixmulder 21/24

MEME

@felixmulder 22 / 24

Thank You!

@felixmulder 23 / 24

References

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

@felixmulder 24/24