## **Intro to Functional Programming**

Equational Reasonging, Compositionality, and Effects

Felix Mulder

Scala Stockholm Meetup - May 30

@felixmulder 1/11

## Who am I?

- · Scala 2.12 Docs Compiler
- · Scala 3 Compiler Engineer @ EPFL w/ Martin Odersky
- · Software Engineer @ Klarna Bank

@felixmulder 2/11

"The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise"

- Edsger W. Dijkstra

@felixmulder 3/11

def foo(a: Int): Int = ???

@felixmulder 4/11

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

@felixmulder 5/11

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

@felixmulder 6/11

@felixmulder 7/11

## Future[\_] vs Actors

@felixmulder 8/11

```
apply: A ⇒ Future[A]
map: (A ⇒ B) ⇒ Future[A] ⇒ Future[B]
flatten: Future[Future[A]] ⇒ Future[A]
```

@felixmulder 9/11

receive: Any  $\Rightarrow$  Unit

@felixmulder 10 / 11

```
\mbox{\bf def become(behavior: Any} \Rightarrow \mbox{\bf Unit, } \dots \mbox{\bf ): Unit} \mbox{\bf def unbecome(): Unit}
```

@felixmulder 11/11