

Intro to Functional Programming

Equational Reasoning, Compositionality, and Effects

Felix Mulder

Scala Stockholm Meetup - May 30

Who am I?

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

“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

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

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

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


Future[_] vs Actors

`apply: A \Rightarrow Future[A]`

`map: (A \Rightarrow B) \Rightarrow Future[A] \Rightarrow Future[B]`

`flatten: Future[Future[A]] \Rightarrow Future[A]`

receive: Any \Rightarrow Unit

```
def become(behavior: Any  $\Rightarrow$  Unit, ... ): Unit
```

```
def unbecome(): Unit
```