The MONAD FACT Slide Deck Series

a very simple rationale for the series

plus a list of currently available slide decks

$$\begin{tikzcd}
T^3 \arrow[r, "T\mu"] \arrow[d, "\mu T"'] \arrow[dr, phantom, "\text{FACT}"] & T^2 \arrow[d, "\mu"] \\
T^2 \arrow[r, "\mu"'] & T
\end{tikzcd}$$

slides by @philip_schwarz

slideshare https://www.slideshare.net/pjschwarz

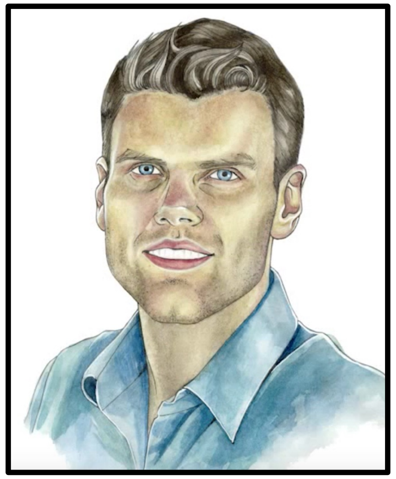
@philip_schwarz

Just a couple of slides being an entry point for the **MONAD FACT** series of short slide decks.

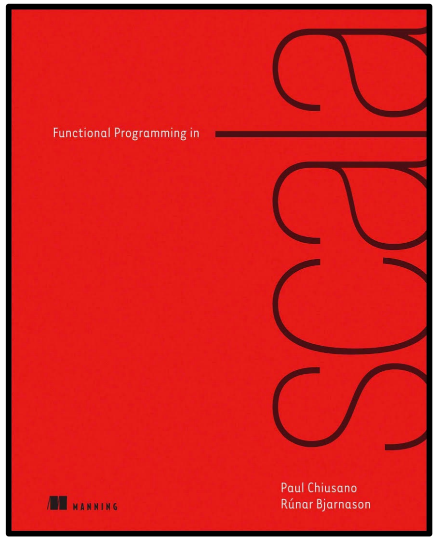
The first slide is the very simple rationale for the series, and the second slide is a list of the decks available so far.

I'll be updating the index as I add new decks to the series.

An abstract topic like this can't be fully understood all at once. It requires an iterative approach where you keep revisiting the topic from **different perspectives**. When you discover new **monads** or **new applications** of them, or see them appear in a **new context**, you'll inevitably gain **new insight**. And each time it happens, you might think to yourself, "OK, I thought I understood **monads** before, but now I really get it."



@runarorama



Functional Programming in Scala
(by Paul Chiusano and Runar Bjarnason)

@pchiusano

Each short slide deck in this series will cover one single such perspective, application, context, or insight.

MONAD FACT slide decks available so far:

#1 Scala **for comprehensions** require a **monad** to be defined in terms of **unit**, **map** and **flatMap** rather than simply in terms of **unit** and **flatMap**

#2 equivalence of **nested flatMaps** and **chained flatMaps** for **Kleisli arrow composition**

#3 how placing **kleisli composition** logic in **flatMap** permits composition of **kleisli arrows** using **for comprehensions** and what that logic looks like in six different **monads**

#4 a **monad** is an implementation of one of the **minimal sets** of **monadic combinators**, satisfying the laws of **associativity** and **identity** - see how **compositional responsibilities** are distributed in each **combinator set**

#5 a chain of **monadic flatMap** calls (or an equivalent **for-comprehension**) is like an **imperative program** with **statements** that **assign** to **variables** and the **monad** specifies what occurs at **statement boundaries**

#6 a **monad** is an **overloading** of the **semicolon**

To fully enjoy the slide decks, make sure you download them: when viewed on the slideshare site they look grainy and out of focus, whereas downloaded slides look nice and crisp, as they are meant to be seen.

The above slide decks are available at https://www.slideshare.net/pjschwarz slideshare