Free Monads

Advanced Scala, London

7th April 2015

Noel Welsh, @noelwelsh

underscore

What?

Free monad = monad + interpreter

Why?

It solves difficult problems

E.g.

Facebook Haxl / Twitter Stitch

Orchestrate web service requests

Batch requests

Cache results

Compare recent Etsy blog post

Monads

Functional programming is about transforming values

A => B => C

FP patterns are just special cases of this

A => B => C

F[A] => F[B] => F[C]

F[A] => F[B] => F[C]

F[A] => F[B]

F[A] flatMap (A => F[B])

Monads are about sequencing computations

Broadly applicable (& applicative)

Aside: monads were introduced to model language semantics

Aside: all monads can be expressed in terms of the continuation monad

Interpreters

Separate structure and meaning

Structure: represent computation as data

E.g.

$1 + 2 + 3 = \text{Add}(1, \text{Add}(2, 3))$

Abstract syntax tree

Meaning: run or “interpret” the structure

E.g.

Compute with Int, Double, or arbitrary precision

E.g.

Compute with Dual Numbers
(automatic differentiation)

E.g.

Compute with SIMD or GPU

Doodle: One AST. JS and JVM interpreters

Free Monads

Free monad provides an AST for monadic operations

We can then write custom interpreters

Haxl / Stitch custom interpreter

What does the AST look like?

Monad has two operations: flatMap and point

So AST has two cases:
flatMap and point

Aside: Scalaz AST slightly more complex to support trampolining

What does an interpreter look like?

A natural transformation

Example

That’s it!

Aside: The free monad requires that it's payload is a Functor

Aside: We can construct a Functor from any value using the Coyoneda

Aside: We often want to combine different payloads in the free monad.

Aside: We can do this with Coproducts yielding composable monads and interpreters

Conclusions

Free monads are simple

It’s just an AST and an interpreter for that AST

Functional programming is just the same stuff over and over again

Meta-Conclusions

Exciting times for functional programming

New techniques being discovered *now*

Composable interpreters via the free monad was 2008

Industry adoption driving compression of transfer time from academia to practice

Programming practice is being reclaimed from software engineering

Build your toolbox

Invest in excellence

Reap the rewards

Advanced Scala

with Scalaz and Essential Interpreters

underscore

underscore.io/training/courses/advanced-scala/