

Off the Haskell Mark

Incremental Lessons from First Programs

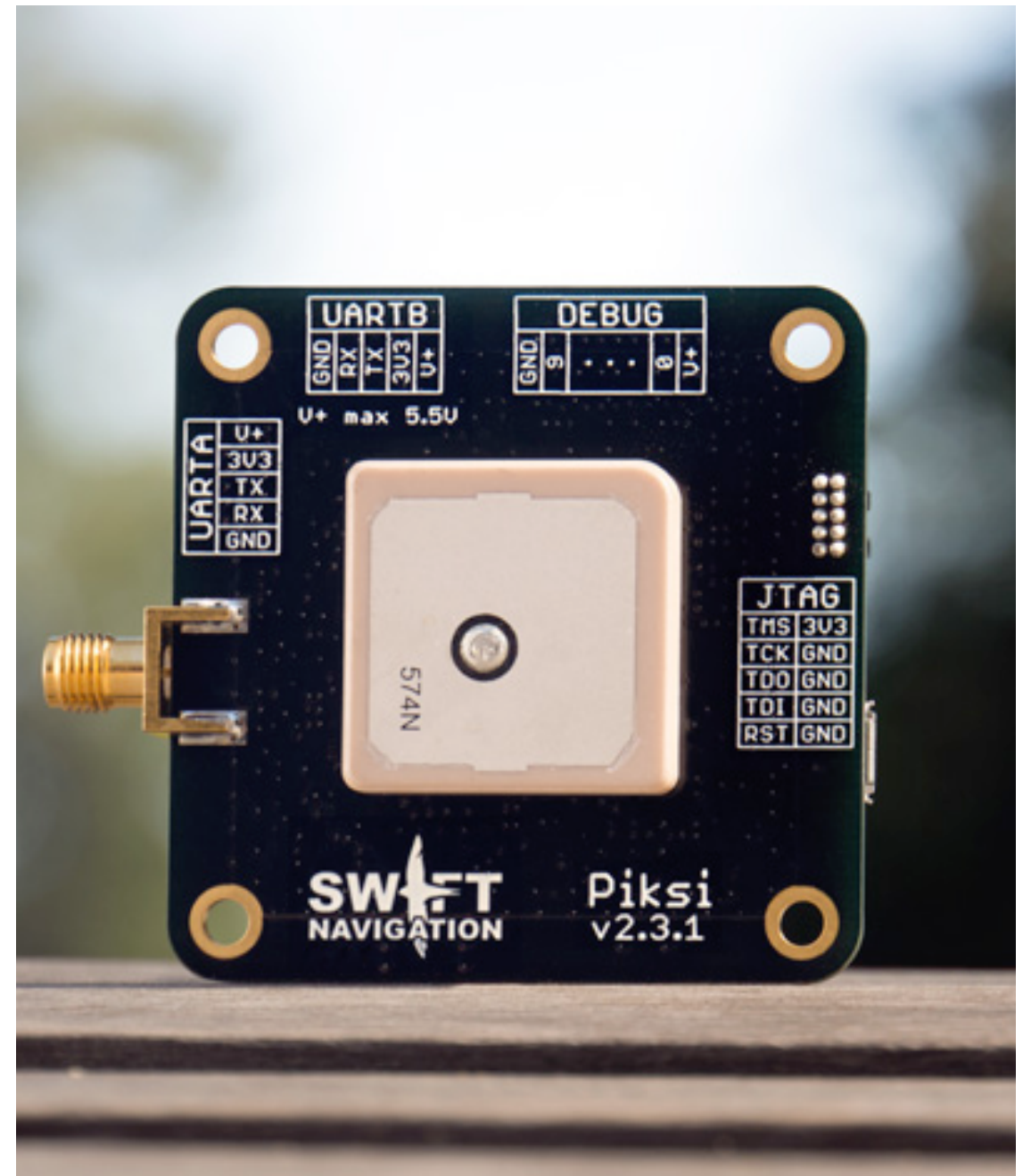
Mark Fine

mark@swift-nav.com

github.com/mfine/hs-talks

Haskell @ Swift Navigation

- Low-cost high accuracy GPS receiver with Real Time Kinematics (RTK)
- Haskell matrix libraries for embedded systems
- Haskell IoT backends for device connectivity
- Haskell infrastructure libraries for testing and analytics
- jobs.lever.co/swift-nav





- Haskell DSL for compiling linear algebra into robust, efficient C code suitable for running on embedded systems
- Generates code free of dynamic memory allocation and provides compile-time checking of matrix and vector
- swift-nav.github.io/plover/



Real World Haskell

O'Sullivan,
Goerzen &
Stewart

O'REILLY

Texts in
Computing


The Haskell Road to Logic, Maths and Programming

Kees Doets
Jan van Eijck

KING'S
College
LONDON
Publications

BIRD THINKING FUNCTIONALLY with HASKELL

CAMBRIDGE




Beginning Haskell

Serrano
Mena

Apress®

SECOND EDITION



Haskell

The Craft of Functional
Programming

Thompson





ADDISON
WESLEY

The Haskell School of Expression

CAMBRIDGE

consume putObject language:Haskell

Search

-  [Repositories](#)
-  [Code](#) 9
-  [Issues](#)
-  [Users](#)

Languages

Text	618
JSON	272
Java	220
HTML	29
XML	17
C#	17
JavaScript	16
PHP	14
Markdown	11
Haskell	9

[Advanced search](#) [Cheat sheet](#)

 **tathougies/Eddy – DownloadXBRL.hs** Haskell

Showing the top two matches. Last indexed on May 31.

```
37     sourceHandle stdin $= B.lines $= CL.map (decode . fromChunks . (\x -> [x])) $=
    CL.catMaybes $= CL.map (\x -> (rXBRLRef x, rCIKNumber x)) $$ CL.consume :: IO [(Text,
    Text)])
38
39     let serviceConf = s3 HTTPS s3EndpointUsClassic True
    ...
48         response <- http request manager
49         fromChunks <$> (responseBody response $$+ (Z.compress 9 (WindowBits 31) =$
    CL.consume))
50         let s3objectBody = RequestBodyLBS dat
```

 **jwiegley/gitlib – Main.hs** Haskell

Showing the top two matches. Last indexed on May 29.

```
48     lbs <- BL.fromChunks <$> (src $$ consume)
49     let req = putObject bucket filepath (RequestBodyLBS lbs)
50     res <- aws config svcConfig manager req
```

 **yihuang/cloud-ftp – Aliyun.hs** Haskell

Showing the top two matches. Last indexed on May 29.

```
47         , putBucket      = _putBucket
48         , getObject      = _getObject
49         , putObject      = _putObject
50         , renameObject   = _renameObject
    ...
60         C.$= C.map printBucketContent
61         C.$$ C.consume
62
63     _putBucket bucket =
64         void $ run conf $ Ali.putBucket bucket Nothing
```

What is a monad?



asked 7 years ago

viewed 107025 times

active 1 month ago



724



447

Having briefly looked at Haskell recently, what would be a *brief, succinct, practical* explanation as to what a monad essentially is?

I have found most explanations I've come across to be fairly inaccessible and lacking in practical detail.

[haskell](#)

[functional-programming](#)

[monads](#)

[terminology](#)

[share](#) [edit](#)

edited Aug 28 at 17:05


community wiki

7 revs, 7 users 67%

[Peter Mortensen](#)

5 Eric Lippert wrote an answer to this questions (stackoverflow.com/questions/2704652/...), which is due to some issues lives in a separate page. – [Pavel Shved](#) Apr 25 '10 at 5:24

4 possible duplicate of [Can anyone explain Monads?](#) – [Roger Pate](#) May 27 '10 at 1:10

4 This article got me closer than any others to understanding monads: ertes.de/articles/monads.html – [sarnold](#) Jan 31 '11 at 2:17 

36 [Here's](#) a new introduction using javascript - I found it very readable. – [Benjol](#) Mar 31 '11 at 20:57

3 See also [Different ways to see a monad.](#) – [Petr Pudlák](#) Sep 27 '12 at 8:56

[show 9 more comments](#)

41 Answers

[active](#)

[oldest](#)

[votes](#)

Featured on Meta



[The Power of Teams: A Proposed Expansion of Stack Overflow](#)

Hot Meta Posts

20 [Should the \[auto\] tag be renamed?](#)

14 [Adding details and hyperlinks in edits](#)

30 [Needless bulk tag edits](#)

Looking for a job?

Sr. Software Engineer, Big Data
TubeMogul
Emeryville, CA

[linux](#) [unix](#)

Service Engineer (Work from anywhere in the World!)

GitLab

San Francisco, CA / remote

Hoogle Search

Within [LTS Haskell 3.9 - GHC 7.10.2](#)

☐ Exact lookup

[caseMaybe :: Maybe a -> b -> \(a -> b\) -> b](#)

[Agda](#) [Agda.Utils.Maybe](#)

Version of maybe with different argument ordering. Often, we want to case on a Maybe, do something interesting in the Just case, but only a default value in the Nothing case. Then, the argument ordering of caseMaybe is preferable.

```
> caseMaybe m d f = flip (maybe d) m f
```

[caseMaybe :: Maybe a -> b -> \(a -> b\) -> b](#)

[Agda](#) [Agda.Utils.Maybe.Strict](#)

Version of maybe with different argument ordering. Often, we want to case on a Maybe, do something interesting in the Just case, but only a default value in the Nothing case. Then, the argument ordering of caseMaybe is preferable.

```
> caseMaybe m err f = flip (maybe err) m f
```

[maybe :: b -> \(a -> b\) -> Maybe a -> b](#)

[base](#) [Prelude Data.Maybe](#)

[basic-prelude](#) [CorePrelude](#)

The maybe function takes a default value, a function, and a Maybe value. If the Maybe value is Nothing, the function returns the default value. Otherwise, it applies the function to the value inside the Just and returns the result.

```
Tag "h4" [Tag "b" [Char 'E',Char 'x',Char 'a',Char 'm',Char 'p',Char 'l',Char 'e',Char 's']]
```

Basic usage:

```
> >>> maybe False odd (Just 3)
```

```
> True
```

```
> >>> maybe False odd Nothing
```

```
> False
```

Haskell in real life can be scary

```
./Network/AWS/Flow.hs:70:45: Couldn't match type 'm0' with 'DecideT m' ...
  because type variable 'm' would escape its scope
  This (rigid, skolem) type variable is bound by
    the type signature for decide :: MonadFlow m => Plan -> m ()
    at /Users/mark/swift/wolf/src/Network/AWS/Flow.hs:65:11-37
  Expected type: DecideT m [Decision]
  Actual type: m0 [Decision]
  Relevant bindings include
    decide :: Plan -> m ()
      (bound at /Users/mark/swift/wolf/src/Network/AWS/Flow.hs:66:1)
  In the fourth argument of 'runDecide', namely 'select'
  In a stmt of a 'do' block:
    decisions <- runDecide logger plan events select
./Network/AWS/Flow.hs:105:3: No instance for (Monad m0) arising from a do statement ...
  The type variable 'm0' is ambiguous
  Relevant bindings include
    select :: m0 [Decision]
      (bound at /Users/mark/swift/wolf/src/Network/AWS/Flow.hs:104:1)
  Note: there are several potential instances:
    instance adjunctions-4.2.2:Data.Functor.Rep.Representable f =>
      Monad (adjunctions-4.2.2:Data.Functor.Rep.Co f)
      -- Defined in 'adjunctions-4.2.2:Data.Functor.Rep'
    instance Monad aeson-0.8.0.2:Data.Aeson.Types.Internal.Parser
      -- Defined in 'aeson-0.8.0.2:Data.Aeson.Types.Internal'
    instance Monad aeson-0.8.0.2:Data.Aeson.Types.Internal.Result
      -- Defined in 'aeson-0.8.0.2:Data.Aeson.Types.Internal'
    ...plus 62 others
  In a stmt of a 'do' block:
    event <- nextEvent
      [WorkflowExecutionStarted, ActivityTaskCompleted, TimerFired,
       StartChildWorkflowExecutionInitiated]
  In the expression:
    do { event <- nextEvent
      [WorkflowExecutionStarted, ActivityTaskCompleted, ....];
      case event ^. heEventType of {
        WorkflowExecutionStarted -> start event
        ActivityTaskCompleted -> completed event
        TimerFired -> timer event
        StartChildWorkflowExecutionInitiated -> child
        _ -> throwM (userError "Unknown Select Event") } }
  In an equation for 'select':
    select
      = do { event <- nextEvent [WorkflowExecutionStarted, ....];
        case event ^. heEventType of {
          WorkflowExecutionStarted -> start event
          ActivityTaskCompleted -> completed event
          TimerFired -> timer event
          StartChildWorkflowExecutionInitiated -> child
          _ -> throwM (userError "Unknown Select Event") } }
./Network/AWS/Flow.hs:105:12: No instance for (MonadCatch m0) arising from a use of 'nextEvent'
  The type variable 'm0' is ambiguous
  Relevant bindings include
    select :: m0 [Decision]
      (bound at /Users/mark/swift/wolf/src/Network/AWS/Flow.hs:104:1)
  Note: there are several potential instances:
    instance MonadCatch m =>
      MonadCatch (Control.Monad.Trans.AWS.AWST' r m)
      -- Defined in 'Control.Monad.Trans.AWS'
    instance MonadCatch GHC.Conc.Sync.STM
      -- Defined in 'Control.Monad.Catch'
    instance MonadCatch m =>
      MonadCatch
        (conduit-1.2.5:Data.Conduit.Internal.Conduit.ConduitM i o m)
      -- Defined in 'conduit-1.2.5:Data.Conduit.Internal.Conduit'
    ...plus 19 others
  In a stmt of a 'do' block:
    event <- nextEvent
      [WorkflowExecutionStarted, ActivityTaskCompleted, TimerFired,
       StartChildWorkflowExecutionInitiated]
  In the expression:
    do { event <- nextEvent
  *- *wolf* 17% of 5.3k (18,0) (Interactive-Haskell WS)
```


“We are using here a powerful strategy of synthesis: wishful thinking.”

Gerald Jay Sussman and Hal Abelson in SICP

“As you're writing your program down toward the language, you build the language up toward your program. You work bottom-up, as well as top-down.”

Paul Graham in ANSI Lisp

Program Wishfully, Type-fully

- Grow your program forwards from your inputs
- Grow your program backwards from your outputs
- Use the type system to help find your way
- Use **:type**, **:info**, **hoogle** liberally, constantly

Program Incrementally

- Make ***Minimum Viable Changes***
- Use **undefined** to defer choices
- Keep the compiler happy
- Compile early, often, always

ALWAYS BE

COMPILING

- github.com/mfine/hs-talks/pulls
- 12 step program to a Haskell program
- (also 12 incremental versions of Lib.hs in src/)



Which is the best NFL division?

East vs. West? North vs. South?

RANK

1



4-0 PATRIOTS

1 ▲

Let's be real: Patriots at Cowboys was a mismatch of "Harry Hamlin vs. The Kraken" proportions. And the fact that the football player who has referred to himself by the latter name managed to lay a few licks on Tom Brady didn't translate to diddly-poo in terms of the outcome. Often, sacks are overrated, like when they come in the first quarter of a contest that ends up 30-6 in favor of the other guys. New England is making everyone and everything (that is, any word with a "gate" on the back end) fade into bolivian right now, and it's becoming more and more difficult to keep the Patriots from the top perch in these here rankings. Never mind -- I just moved them.

RANK

2



5-0 PACKERS

1 ▼

"You see?! He's *not* a machine! He's a man!!" Somewhere, Duke from Rocky IV was watching Aaron Rodgers toss not one but *two* picks at Lambeau -- his first such transgressions there since 2012. One pick came on a tipped ball, while the other was the result of some real professional thievery by Trumaine Johnson. Meanwhile, the Packers drop a spot -- but not because of any shortcomings at home against the Rams. (Well, besides the run defense against Todd Gurley. But that guy's a *beast*. Whatcha gonna do?) Rather, the Patriots have been walking all over everyone on the schedule -- and they deserve a bump for that.

RANK

3



5-0 BENGALS

Wow. What a response in the second half from the Bengals, down 24-7, to climb back into that football game. Andy Dalton is deserving of the rose petals suddenly being thrown at his feet, much like King Joffe, but don't forget the defense, which got after Russell Wilson in the fourth quarter and overtime. Stopping the run -- Seattle finished with 200 yards on the ground -- is another matter, but let's not spoil a 5-0 start.

NFL Power Rankings by Divisions (Average)

