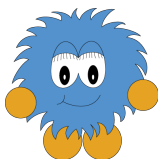


What The Hell Are Monads?

Alex Vondrak

ajvondrak@csupomona.edu

January 28, 2011

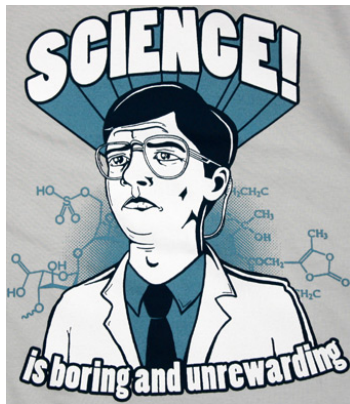


“Mo-what?”

- Tried doing this in a half-hour CS 664 talk
- Reviews were fairly unanimous:

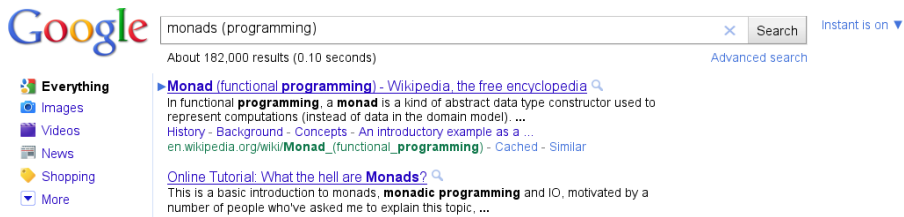
“Mo-what?”

- Tried doing this in a half-hour CS 664 talk
- Reviews were fairly unanimous:



Intended Audience

- “Monads? Those functional programming thingies for PhDs?”



A screenshot of a Google search interface. The search bar contains the text "monads (programming)". To the right of the search bar is a "Search" button and a "Instant is on" indicator. Below the search bar, it says "About 182,000 results (0.10 seconds)". On the left side, there is a vertical menu with icons and labels: "Everything", "Images", "Videos", "News", "Shopping", and "More". The main search results area shows two entries. The first entry is a blue link: "Monad (functional programming) - Wikipedia, the free encyclopedia". Below this link is a snippet of text: "In functional programming, a monad is a kind of abstract data type constructor used to represent computations (instead of data in the domain model). ... History - Background - Concepts - An introductory example as a ... en.wikipedia.org/wiki/Monad_(functional_programming) - Cached - Similar". The second entry is a blue link: "Online Tutorial: What the hell are Monads?". Below this link is a snippet of text: "This is a basic introduction to monads, monadic programming and IO, motivated by a number of people who've asked me to explain this topic, ...".

Intended Audience

- “Monads? Those functional programming thingies for PhDs?”

ALL - RANDOM | PICS - REDDIT.COM - FUNNY - POLITICS - ASKREDDIT - WTF - GAMING - SCIENCE - WORLDNEWS - PROGRAMM



reddit

SEARCH RESULTS

previous search

search reddit

advanced search: by author, community...

Submit Query

about 140 results in 1.218 seconds Powered by:

satisfied? yes no

sorted by: [relevance](#) ▼

- ↑ 0 [Yet Another Monad Article: Monads in C#, or you maybe just be using Monads and didn't know it \(LINQ\)](#) (blogs.msdn.com)
 ↓ submitted 18 hours ago by banuday to programming
 1 comment share
- ↑ 58 [The IO monad is 45 years old -- Peter Landin's 1965 paper](#) (article.gmane.org)
 ↓ submitted 28 days ago by dons to programming
 8 comments share
- ↑ 2 [The Writer Monad using Scala with example](#) (blog.tmorris.net)
 ↓ submitted 1 month ago by [deleted] to programming
 comment share
- ↑ 27 [Monad Macros in Common Lisp](#) (common-lisp.net)
 ↓ submitted 1 month ago by shenglong to programming
 12 comments share

Intended Audience

- “Monads? Those functional programming thingies for PhDs?”



Questions

Tags

Users

Badges

Unanswered

Ask Question

Cast your vote in the Stack Overflow 2011 [community moderator primary!](#)

Search Results

relevance

newest

votes

active

72

votes

15

answers

5k views

What is a monad?

... explanation as to what a **monad** essentially is? I have found most explanations I' ...

[functional-programming](#) [terminology](#) [monads](#)

asked Sep 4 '08 at 23:26



kronoz

10.8k • 8 • 43 • 83

posts containing

monad

Want better search results?

[See our search tips!](#)

57

votes

14

answers

6k views

Can anyone explain Monads?

I think I understand what 'Maybe Monads' are, but I'm not sure about the other types. ...

[haskell](#) [functional-programming](#) [monads](#) [glossary](#)

asked Aug 5 '08 at 14:16



Steve Willard

1,081 • 1 • 9 • 16

16

votes

11

answers

1k views

Monad in non-programming terms

... How would you describe a **monad** in non-programming terms? Is there some concept/ ... ng, not just FP) which could be said to act or be **monad**-like in a significant way? ...

[haskell](#) [functional-programming](#) [monads](#)

asked Jul 16 '10 at 3:26



fig

2,870 • 9 • 25

Intended Audience

- “Monads? Those functional programming thingies for PhDs?”

Google search results for "site:4chan.org monad".

Search bar: site:4chan.org monad

Results: About 227 results (0.09 seconds)

Navigation: Instant is on, Advanced search

Filters: Everything, Images, Videos, News, Shopping, More

Results:

- [4chan BBS - /prog/ writes a monad tutorial](#)
 - 30 posts - 1 author - Last post: Jan 5
 - s/**Monads** is a way way away of seeples abstract nonsense wich implements nothing/I'm a huge Faggot please rape my face/ ...
 - dis.4chan.org/read/prog/1294059438
- [4chan BBS - Meet Monica **Monad**](#)
 - 43 posts - 1 author - Last post: Aug 3, 2009
 - It's an artist's depiction of a **Monad**. It's not finished because the artist shot himself in the

Intended Audience

- “Monads? Those functional programming thingies for PhDs?”

monads are like burritos	X	Search
monads are like burritos		
monads are elephants		
monads are windowless		
monads are monoids in the category of endofunctors		
monads are		

Reality

- Never heard of monads
- Heard even less about functional programming
- And what the hell is category theory, anyway?



Why Do We Care?

Functional Programming

- Haskell: the poster-child for monads
- Purely functional = No side-effects
- Why functional programming matters
 - Variables don't change unexpectedly
 - Functions always compute the same result
 - Major source of bugs is eliminated
 - Order of execution doesn't matter—easier to reason about
 - Easier for compiler to reason about, too
- Not going to sell you on functional programming with one slide, but...

Why Do We Care?

Functional Programming

- Haskell: the poster-child for monads
- Purely functional = No side-effects
- **Why functional programming matters**
 - Variables don't change unexpectedly
 - Functions always compute the same result
 - Major source of bugs is eliminated
 - Order of execution doesn't matter—easier to reason about
 - Easier for compiler to reason about, too
- Not going to sell you on functional programming with one slide, but...

Why Do We Care?

Monads

- “What are the advantages of monads?”
 - Nothing
 - Zilch
 - \emptyset
 - Absolutely 0
- Monads **are not** a language feature, they’re a **structure**
- Some languages are explicit about their monads

```
public interface Monad { ... }
```

- Certain data types *are* monads

```
class Foo implements Monad { ... }
```

- Utilities can work on monads in **general**

```
bar(Monad m1, Monad m2) { ... }
```

Why Do We Care?



“Because it’s cool.”

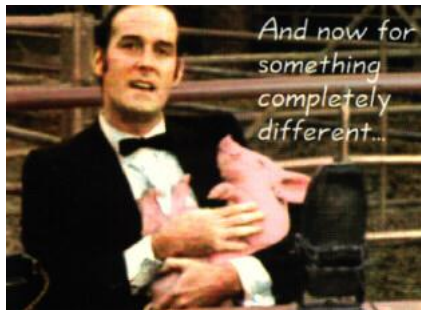
–Hologram from *Invader Zim* explaining why Martians decided to turn their planet into a giant spaceship

Outline



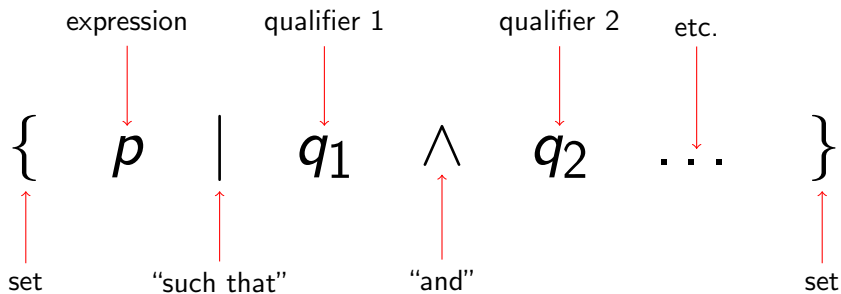
Comprehending Monads—Philip Wadler

- How are monads **defined**?
- Less time for how are monads used. . .



Set-Builder Notation

- How many of you remember CS 130?



Set-Builder Notation

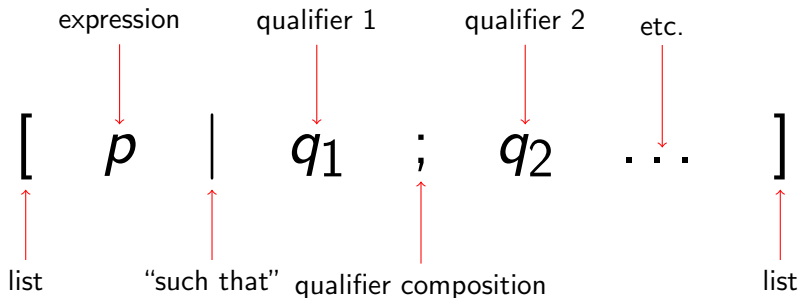
Examples

$$\{x \mid x \in \{1, 2, 3\}\}$$
$$= \{1, 2, 3\}$$

$$\{x + 1 \mid x \in \{1, 2, 3\}\}$$
$$= \{2, 3, 4\}$$

$$\{x + y \mid x \in \{1, 2, 3\} \wedge y \in \{10, 20\}\}$$
$$= \{1 + 10, 1 + 20, 2 + 10, 2 + 20, 3 + 10, 3 + 20\}$$

List Comprehensions



- **Difference**: qualifiers are all of the form $x \leftarrow L$
- In Python: `[p for x in L]`
- In C#: `from x in L select p`

Lists

- Ordered sequence of elements
- Elements all have the **same** type
- Doesn't matter what that type is

Examples (Types)

```
[1, 2, 3]: List<int>
```

```
["a", "b", "c"]: List<string>
```

```
[[1], [2, 3]]: List<List<int>>
```

```
[1, "a", [3]]: X
```

- Rest of this talk = “stuff”

unit

For some item x of type t ...

- List comprehension:

$$[x \mid \emptyset]$$

- Function:

$$\text{unit} : t \rightarrow \text{List}\langle t \rangle$$

Examples

$$\text{unit}(5) = [5]$$

$$\text{unit}(\text{"a"}) = [\text{"a"}]$$

$$\text{unit}(\text{unit}(10)) = [[10]]$$

map

For some list L of type $\text{List}\langle t \rangle$ and a function $f: t \rightarrow t' \dots$

- List comprehension:

$$[f(x) \mid x \leftarrow L]$$

- Function:

$$\text{map}: (t \rightarrow t') \times \text{List}\langle t \rangle \rightarrow \text{List}\langle t' \rangle$$

Examples

$$\text{map}(\text{add1}, [1, 2, 3]) = [2, 3, 4]$$

$$\text{map}(\text{uppercase}, ["a", "b", "c"]) = ["A", "B", "C"]$$

$$\text{map}(\text{ascii}, ['a', 'b', 'c']) = [97, 98, 99]$$

map

For some list L of type $\text{List}\langle t \rangle$ and a function $f: t \rightarrow t' \dots$

- List comprehension:

$$[f(x) \mid x \leftarrow L]$$

- Function:

$$\text{map}: (t \rightarrow t') \rightarrow (\text{List}\langle t \rangle \rightarrow \text{List}\langle t' \rangle)$$

Examples

$$\text{map}(\text{add1}, [1, 2, 3]) = [2, 3, 4]$$

$$\text{map}(\text{uppercase}, ["a", "b", "c"]) = ["A", "B", "C"]$$

$$\text{map}(\text{ascii}, ['a', 'b', 'c']) = [97, 98, 99]$$

join

For some list of lists L' of type $\text{List}\langle\text{List}\langle t \rangle\rangle \dots$

- List comprehension:

???

- Function:

$\text{join} : \text{List}\langle\text{List}\langle t \rangle\rangle \rightarrow \text{List}\langle t \rangle$

Examples

$\text{join} ([[1, 2, 3]]) = [1, 2, 3]$

$\text{join} ([[1], [2, 3]]) = [1, 2, 3]$

$\text{join} ([[[1]], [[2]]]) = [[1], [2]]$

join

For some list of lists L' of type $\text{List}\langle\text{List}\langle t \rangle\rangle \dots$

- Set notation:

$$\bigcup_{L \in L'} L$$

- Function:

$$\text{join} : \text{List}\langle\text{List}\langle t \rangle\rangle \rightarrow \text{List}\langle t \rangle$$

Examples

$$\text{join} ([[1, 2, 3]]) = [1, 2, 3]$$

$$\text{join} ([[1], [2, 3]]) = [1, 2, 3]$$

$$\text{join} ([[[1]], [[2]]]) = [[1], [2]]$$

Random Facts — 1/3

For some list L ...

- List comprehensions:

$$\text{join}([L \mid \emptyset]) = L$$

- Functions:

$$\text{join}(\text{unit}(L)) = L$$

Definition

$$\text{join} \circ \text{unit} = \text{id}$$

Example

$$\text{join}(\text{unit}([1, 2, 3])) = \text{join}([[1, 2, 3]]) = [1, 2, 3]$$

Random Facts — 2/3

For some list L ...

- List comprehensions:

$$\text{join} \left(\left[[x \mid \emptyset] \mid x \leftarrow L \right] \right) = L$$

- Functions:

$$\text{join}(\text{map}(\text{unit}, L)) = L$$

Definition

$$\text{join} \circ \text{map}(\text{unit}) = \text{id}$$

Example

$$\text{join}(\text{map}(\text{unit}, [1, 2, 3])) = \text{join} \left(\left[[1], [2], [3] \right] \right) = [1, 2, 3]$$

Random Facts — 3/3

For some list-of-lists-of-lists $L'' \dots$

- List comprehensions:

$$\text{join}([\text{join}(L') \mid L' \leftarrow L'']) = \text{join}(\text{join}(L''))$$

- Functions:

$$\text{join}(\text{map}(\text{join}, L'')) = \text{join}(\text{join}(L''))$$

Definition

$$\text{join} \circ \text{map}(\text{join}) = \text{join} \circ \text{join}$$

Example

$$\text{join}(\text{join}([\![1]\!], [\![2]\!]))) = \text{join}([\![1], [2]\!]]) = [1, 2]$$

Random Facts — 3/3

For some list-of-lists-of-lists $L'' \dots$

- List comprehensions:

$$\text{join}([\text{join}(L') \mid L' \leftarrow L'']) = \text{join}(\text{join}(L''))$$

- Functions:

$$\text{join}(\text{map}(\text{join}, L'')) = \text{join}(\text{join}(L''))$$

Definition

$$\text{join} \circ \text{map}(\text{join}) = \text{join} \circ \text{join}$$

Example

$$\begin{aligned} \text{join}(\text{map}(\text{join}, [[1], [2]])) &= \text{join}([\text{join}([1]), \text{join}([2])]) \\ &= \text{join}([1, 2]) = [1, 2] \end{aligned}$$



*Our biggest mistake: using the scary term “monad” rather than “warm fuzzy thing”.
(Simon Peyton Jones)*

What The Hell Are Monads?

Definition

A **monad** is an operator on types, $M\langle t \rangle$ (like $\text{List}\langle t \rangle$), together with three functions:

- $\text{unit}: t \rightarrow M\langle t \rangle$
- $\text{map}: (t \rightarrow t') \rightarrow (M\langle t \rangle \rightarrow M\langle t' \rangle)$
- $\text{join}: M\langle M\langle t \rangle \rangle \rightarrow M\langle t \rangle$

such that these functions obey the three **monadic laws**—the composition properties we just saw.

- A few other restrictions on unit, map, and join. . .
- But they're essentially “how it works for lists”

In Conclusion

- Monads are warm fuzzy things
- They compose in certain ways that turn out to be **convenient**
 - Which is really the whole point. . .
- Many things are monads:
 - Lists
 - Arrays
 - Exceptions
 - Parsers
 - Continuations
 - . . .
- Math is hard. Let's go shopping!