Elm

http://elm-lang.org/

Mostovenko Alexander

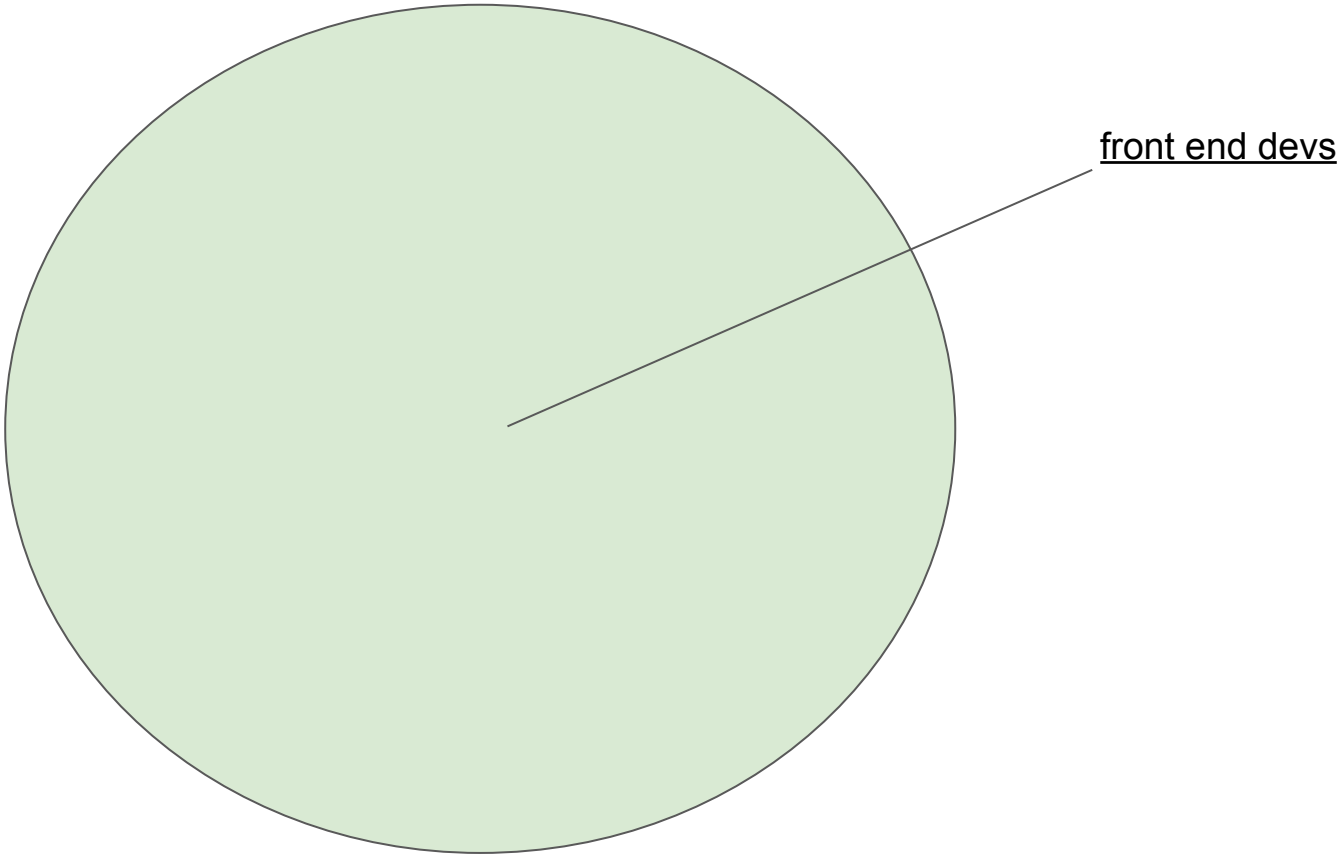
- fullstack engineer in datarobot 
- for inspiration: haskell, elm, purescript
- for job: python, javascript
- twitter - https://twitter.com/MostovenkoA
- github - https://github.com/AlexMost

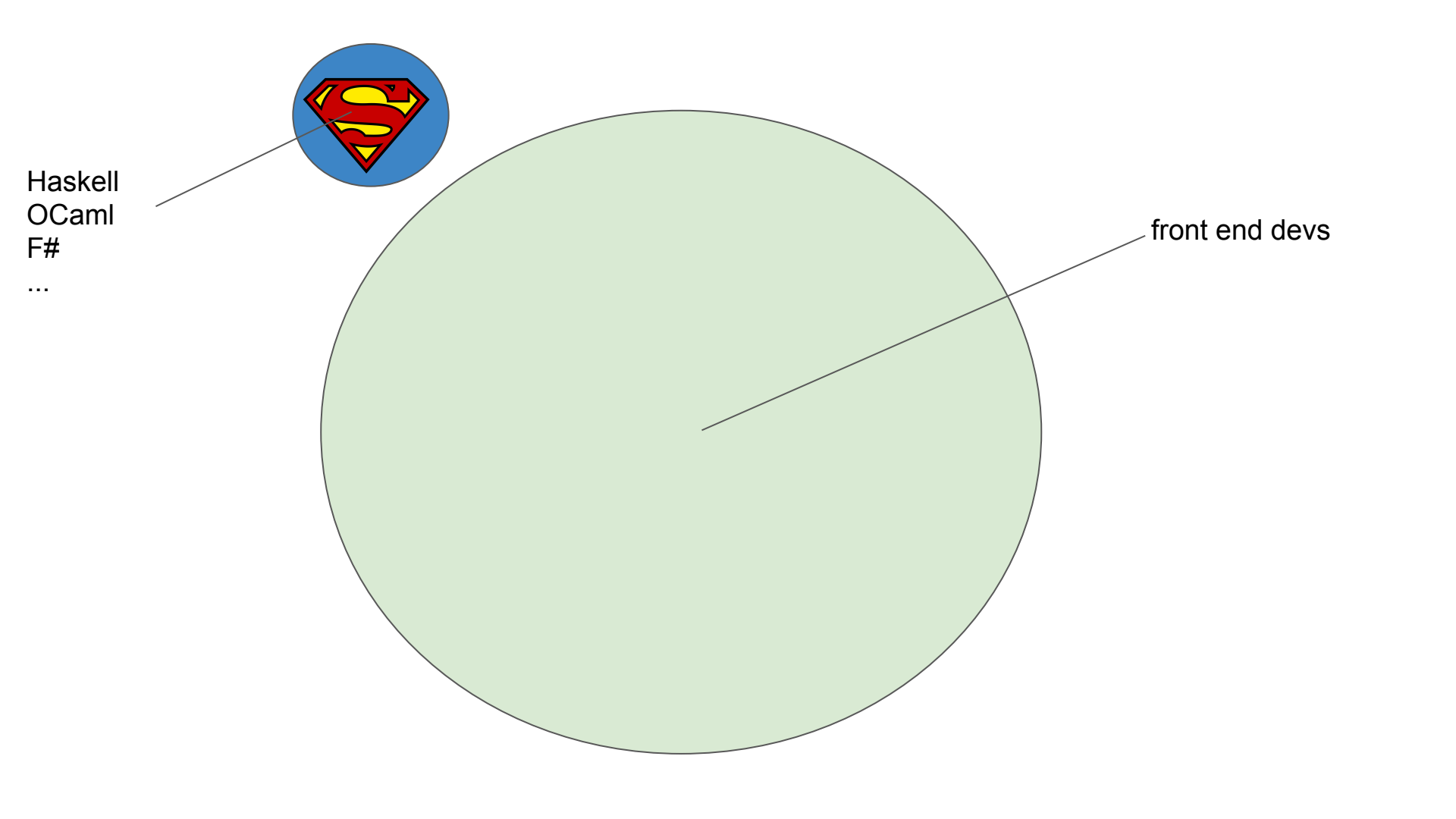
Elm in general

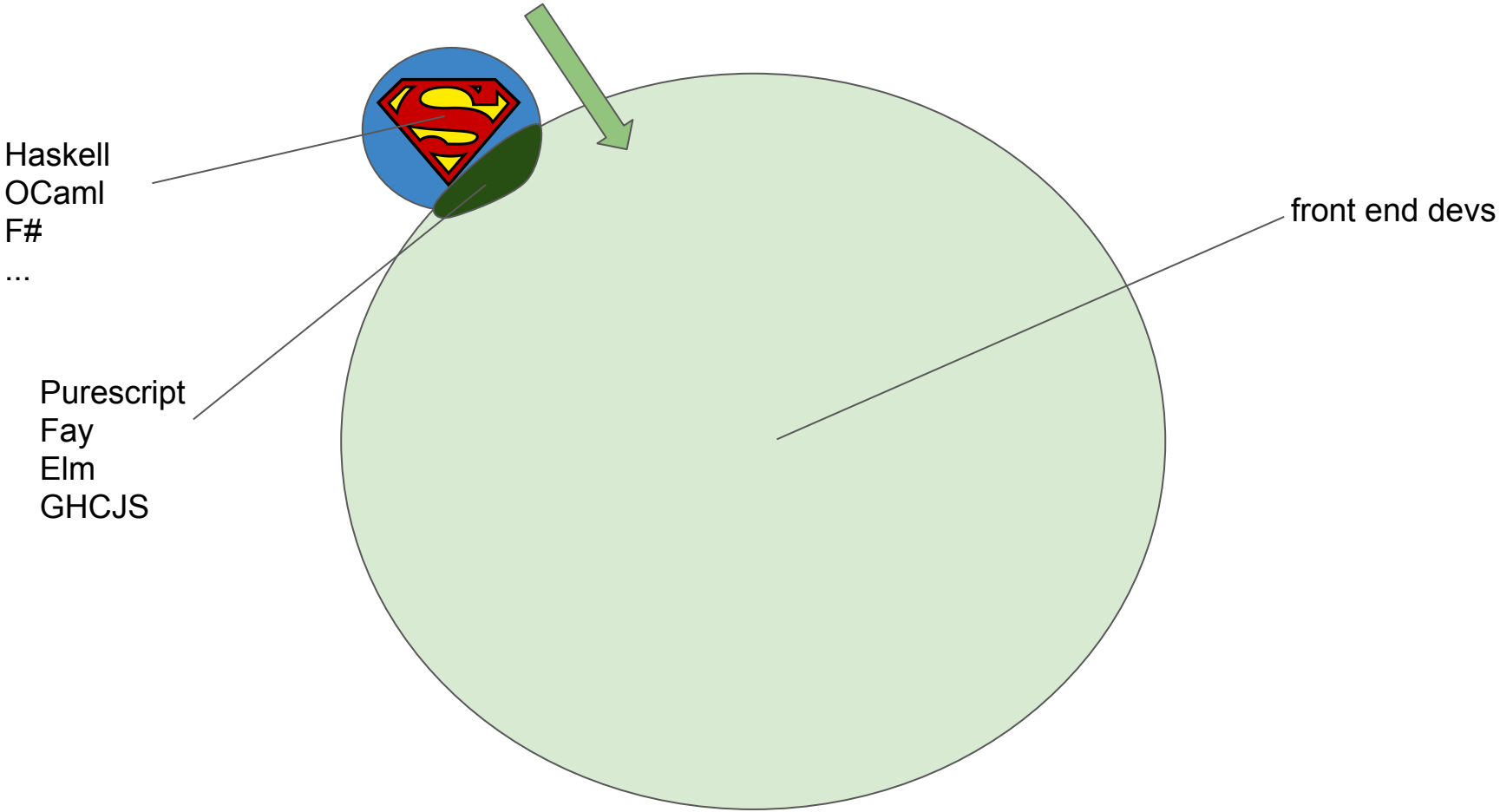
- Functional (ML-like, compiles to js).
- Immutable data structures.
- Statically typed. (ADT, type inference).
- With FRP in mind (Signals are built in language).
- Created for building rich UI apps.

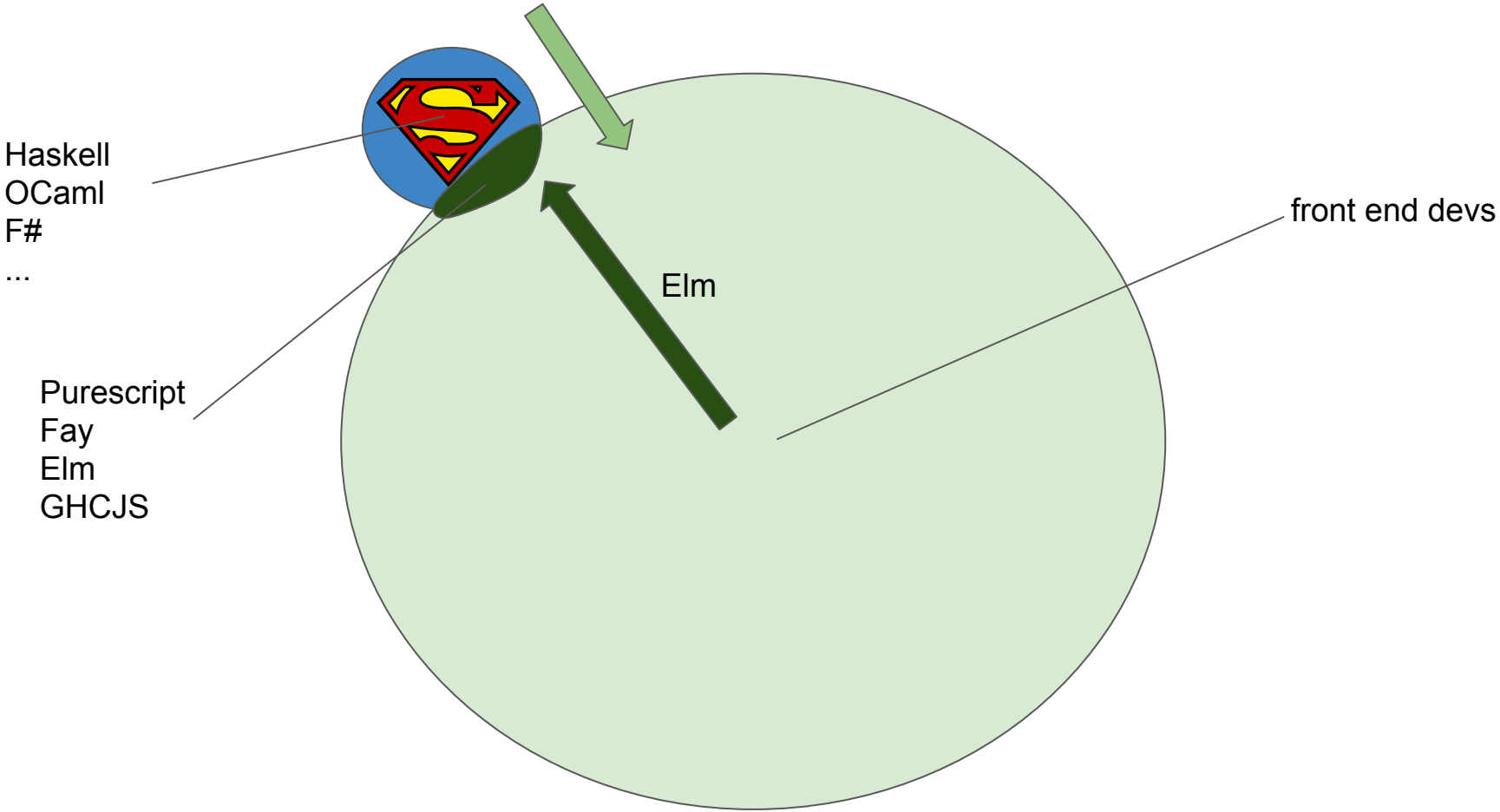
Idea

Why another to-js lang ?


front end devs


Haskell
OCaml
F#
...
front end devs


Haskell
OCaml
F#
...
Purescript
Fay
Elm
GHCJS
front end devs


Haskell
OCaml
F#
...
Purescript
Fay
Elm
GHCJS
Elm
front end devs

The main focus

- Easy to get started.
- Simple and clean syntax.
- Focus on business logic (no frameworks holy wars e.t. c.).
- Simple refactor.
- Simple add features.
- Less bugs in production.

Minimal but powerful syntax

```
type alias Model = Int
-- UPDATE


type Action = Increment | Decrement


update : Action -> Model -> Model
update action model =
 case action of
   Increment -> model + 1
   Decrement -> model - 1
```

- Comments
- Literals
- Lists
- Conditionals
- Union Types
- Records
- Functions
- Infix Operators
- Let Expressions
- Applying Functions
- Mapping with (<~) and (~)
- Modules
- Type Annotations
- Type Aliases
- JavaScript FFI

Extensible records

```
point = { x = 3, y = 4 }          -- create a record


point.x                           -- access field
map .x [point,{x=0,y=0}]          -- field access function
{ point - x }                     -- remove field
{ point | z = 12 }                -- add field
{ point - x | z = point.x }       -- rename field
{ point - x | x = 6 }             -- update field
{ point | x <- 6 }                -- nicer way to update a field
{ point | x <- point.x + 1
        , y <- point.y + 1 }      -- batch update fields
```

Extensible records

```
sumCoordinates: {a | x: Int, y: Int} -> Int

sumCoordinates {x, y} = x + y
```

http://research.microsoft.com/pubs/65409/scopedlabels.pdf

No runtime exceptions

(almost)

js

```
let list = [1, 2, 3];

let first = (lst) => lst[0];

let doubleFirst = (lst) => first(list) * 2
```

Elm

```
list = [1, 2, 3]

first lst = List.get 0 list

doubleFirst lst = (first lst) * 2
```

js

```
let list = [1, 2, 3];

let first = (lst) => lst[0];

let d
```

Elm

```
list = [1, 2, 3]

first lst = List.get 0 list
```

2

TYPE MISMATCH

jump to error

The left argument of (*) is causing a type mismatch.

```
9| doubleFirst list = (first list) * 2
```

As I infer the type of values flowing through your program, I see a conflict between these two types:

    number

    Maybe a

js

```
let list = [1, 2, 3];

let first = (lst) => lst[0];

let doubleFirst = (lst) => first(list) * 2
```

Elm

```
list = [1, 2, 3]

first list = get 0 list

doubleFirst list = case first list of
  Just n -> Just (n * 2)
  Nothing -> Nothing
```

But it doesn’t fail if []?

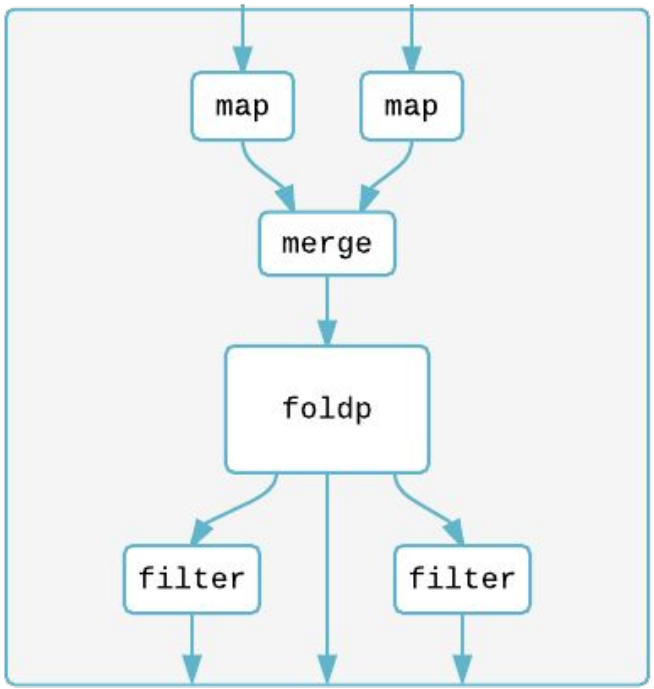
```
let list = [1, 2, 3];

let first = (lst) => lst[0];

let doubleFirst = (lst) => first(list) * 2
```

NaN

Signals



transform inputs into the right shape

merge the inputs into a single signal

update the state of your application
(The Elm Architecture)

route values to the appropriate service

UI app architecture

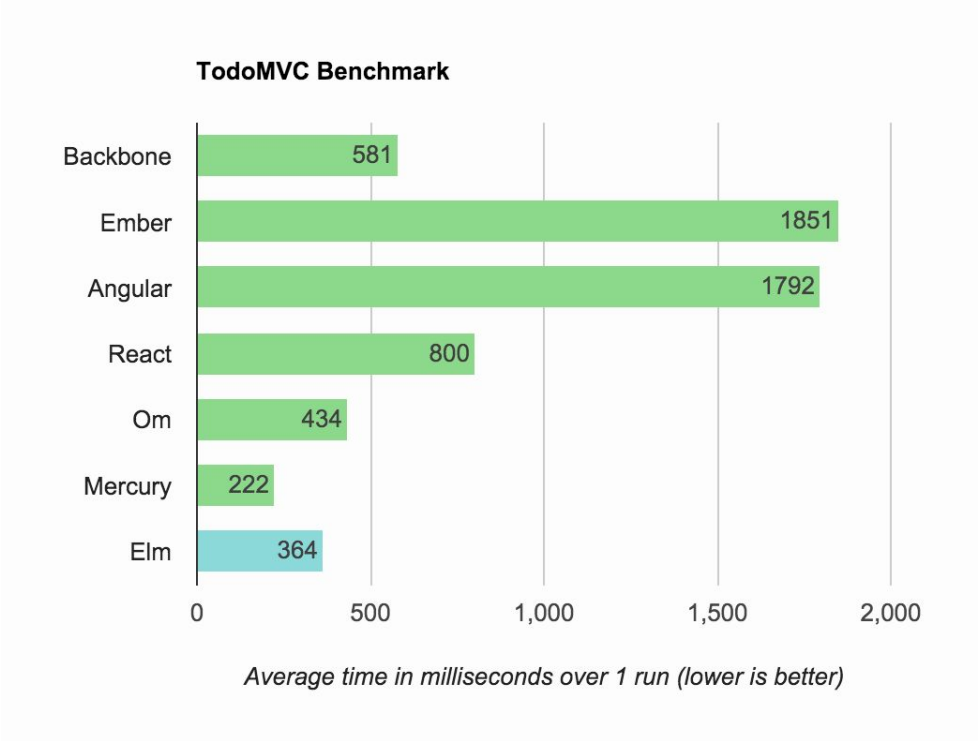
- Unidirectional data flow
- Virtual dom
- FRP based approach

The Elm architecture

https://github.com/evancz/elm-architecture-tutorial/

- Model (State)
- Update
- View

Blazing fast rendering



http://evancz.github.io/todomvc-perf-comparison/

Ecosystem

- Package manager
- Packege catalog - http://package.elm-lang.org/
- build tool (elm-make, elm-reactor)
- time travel debugger
- hot swap

Elm package manager

Semantic versioning (1.0.0 - major.minor.patch)

elm-package bump - new version

can see diff - **elm-package diff 1.0.1**

Time travel debugger

mario demo - http://debug.elm-lang.org/edit/Mario.elm

Unit Testing

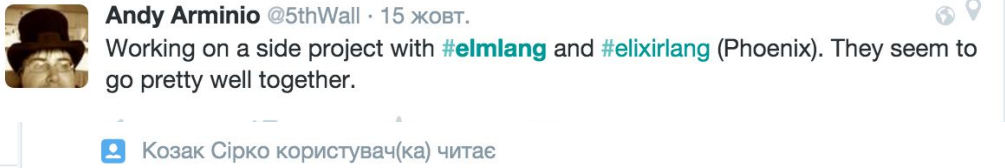
https://github.com/deadfoxygrandpa/Elm-Test

---

1 suites run, containing 5 tests
0 suites and 4 tests passed
1 suites and 1 tests failed

Test Suite: Utils test suite: FAILED
trans: passed.
zerosMap: FAILED. Expected: Dict.fromList [(0,0),(1,1),(2,6),(3,5)]; got: Dict.fromList [(0,0),(1,1),(2,4),(3,5)]
reaplace zero: passed.
bubble zeros: passed.
test move: passed.

Elm exists in production



Andy Arminio @5thWall · 15 жовт.
Working on a side project with **#elmlang** and #elixirlang (Phoenix). They seem to go pretty well together.

Козак Сірко користувач(ка) читає

Zbigniew Siciarz @zsiciarz · 17 жовт.
And as a followup there's slownews - a web app written in #elixirlang and **#elmlang**. github.com/srid/slownews

2

Переглянути підсумки

Mostovenko Alexander ретвітнув(ла)

Ossi Hanhinen @ohanhi · 9 жовт.
Some stats of our @elmlang project so far:
- 2-3 devs, 3mo
- 148 #Elm modules
- 1 JS file
- 3 major refactors
- 0 runtime exceptions

69 96

Mostovenko Alexander ретвітнув(ла)

Richard Feldman @rtfeldman · 19 жовт.
Had our first hire start today who applied because we use @elmlang in production. He rocked it!

PS: still hiring :)
noredink.com/jobs/senior-fr…

15 37

Demo

2048 - https://github.com/AlexMost/2048

more demos - http://elm-lang.org/examples

:)