

Beyond JavaScript Frameworks: Writing Reliable Web Apps With

Elm

Erik Wendel
DevDays Vilnius 2018

BEKK

Who is
Jonathan Ive?

BEKK

Apple Leadership

Jonathan Ive

Chief Design Officer



Jonathan Ive is Apple's Chief Design Officer, reporting to CEO Tim Cook.

Jony is responsible for all design at Apple, including the look and feel of Apple hardware, user interface, packaging, major architectural projects such as Apple Park and Apple's retail stores, as well as new ideas and future initiatives.

Since 1996, Jony has led Apple's design team, which is widely regarded as one of the world's best.

He holds over 5,000 patents and has been recognized with numerous design awards, including the Design Museum London's first Designer of the Year in 2003, the Design and Art Direction (D&AD) President's Award in 2005 and the Cooper-Hewitt National Design Museum's Product Design Award in 2007.

In 2012, D&AD named Jony and his team the Best Design Studio of the past 50 years. Their work is featured in the permanent collections of museums around the world, including the Museum of Modern Art in New York and the Pompidou in Paris.

Jony earned a Bachelor of Arts degree at Newcastle Polytechnic. As an undergraduate, he twice won the Royal Society of Arts' prestigious Student Design Award, and years later the RSA awarded him the title of Royal Designer for Industry. He also holds honorary doctorates from the Royal College of Art, the Rhode Island School of Design and Northumbria University.

A native of London, Sir Jonathan Ive was made a Knight Commander of the British Empire in 2013 "for services to design and enterprise."

Shop and Learn

- Mac
- iPad
- iPhone
- Watch
- TV
- Music
- HomePod
- Apple Watch
- Accessories
- Gift Cards

Apple Store

- Find a Store
- Genius Bar
- Today at Apple
- Apple Game
- Field Trip
- Apple Store App
- Re-Released and Clearances
- Financing
- Personal Recycling
- Order Status
- Shipping Help

For Education

- Apple and Education
- Shop for Schools

For Business

- Apple and Business
- Shop for Business

Account

- Manage Your Apple ID
- Apple Store Account
- iCloud.com

Apple Values

- Accessibility
- Education
- Environment
- Inclusion and Diversity
- Privacy
- Supplier Responsibility

About Apple

- Newsroom
- Apple Leadership
- Investor Relations
- Events
- Contact Apple

Elm is like *Jonathan Ive* would have designed a programming language – it is **minimalistic, user-friendly** and **it just works**

– Peder Korsveien

BEKK

How many of you...

...write JavaScript at work?

Did you ever...

...ship an app with **confidence** it
wouldn't crash in production?

(without loads of QA)

Did you ever...

...feel completely safe after a
large refactor of the frontend code?

Did you ever...

...become **overwhelmed** by
the amount of frontend tech
in 2018?

Did you ever...

...feel like not all team members
are **comfortable** with frontend
tasks?

Check, check, check...



JavaScript fatigue



Worrisome refactors



Dedicated frontend devs



Nail-biting deploys

I will argue that Elm addresses these

while also providing

a dedicated pair-programmer

no runtime errors

error messages that actually help



John Carmack ✓

@ID_AA_Carmack

Oculus VR CTO

Dallas, TX

oculus.com

Joined August 2010

Tweet to

Message

57 Followers you know



122 Photos and videos



John Carmack ✓

@ID_AA_Carmack

Follow

That should be an inspiration for every error message.



Gregory Schier ✓ @GregorySchier

Thanks @elmlang for the most useful error message I've ever seen

9:55 PM · 24 May 2016

429 Retweets 507 Likes



14 429 507



Tweet your reply



Rob Pike @rob_pike · 25 May 2016

Replying to @ID_AA_Carmack

@GregorySchier @elmlang Errors: the first Unix Fortran compiler had JIM (junk in middle) and JOE (junk on end). That was it.

2 26 57



Awesomeclaw @Awesome_Claw · 25 May 2016

I feel like that's actually slightly more useful than 5k lines of C++ template errors.

1 7



Keith Kaisershot @ablitter · 24 May 2016

Replying to @ID_AA_Carmack

MPW used to generate my favorite ones. :D "This array has no size, and that's bad" "Too many errors on one line (make fewer)"

2 9 44



Stefan aRe.Ytz @satefan · 25 May 2016

"You ran out of RAM, go visit your Apple dealer" was a good one too.

1 6

1 more reply

Who to follow Refresh View all



Ken Levine ✓ @levine

Follow



Brenda Romero ✓ @br

Follow



Robert Bowling ✓ @four...

Follow

Find people you know

Trends for you Change

Norwegian

8,591 Tweets

#SkyLibraries

Acer

3,302 Tweets

Stortinget

#kirkemotet

#ZuckerbergHearing

10.1K Tweets

Oslo

5,163 Tweets

#poikvart

@Tonki is Tweeting about this

Norges

Skei Grande

User interface expert, experienced
with javascript and single-page
apps

Worked with large
Norwegian companies
like SpareBank1 and
NSB

Erik Wendel

Serving Elm in
production to 1M+
users



Web Development
Lead at BEKK
Consulting, Oslo
(450 people)

Founder of
Oslo Elm Meetup (2016)
Oslo Elm Day (2017)

Agenda

1. How Elm works

Compared to the JS of today

2. App Example

Counter app

3. Does Anyone Use Elm?

A few stories and examples

Agenda

1. How Elm works

Compared to the JS of today

2. App Example

Counter app

3. Does Anyone Use Elm?

A few stories and examples



How does Elm compare to React?

Elm is a language compiling to JavaScript

it is *not* another library or a framework



How does Elm compare to React?

Elm and JavaScript are totally different

In terms of syntax and semantics



How does Elm compare to React?

Elm uses pure functions and virtual dom

to create a tree of *components*



How does Elm compare to React?

Elm does not allow component state

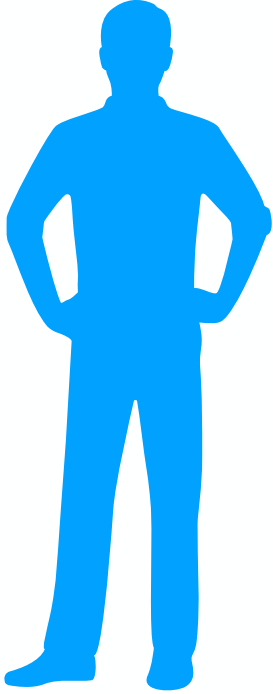
all state is stored in top-level store, like *Redux*



How does Elm compare to React?

Elm uses the Redux architecture

actually, it is the other way around - Redux was inspired by Elm



*"Elm is basically
React-Redux with
type safety"*

This is how it works

```
<div id="container"></div>  
<script src="main.js"></script>
```

```
<script>  
  var node = document.getElementById('container');  
  var app = Elm.Main.embed(node);  
</script>
```


A quick overview

1. Correctness, maintainability and developer-friendliness comes first
2. A functional language of the ML family
(F#, OCaml, Haskell)
3. No run-time errors (!)
4. Heavily opinionated

Key Language Features

1. All data is immutable, and there is no null
2. Expression-oriented, no statements
Everything evaluates to a value
3. Pure (side-effects handled by runtime)
Like redux-saga
4. Architecture as a built-in feature
Redux is a JavaScript-adaptation of The Elm Architecture
5. Small but expressive feature set
Fits together like Lego
Therefore: it's pretty easy to learn

Let's see some code!

```
increment x =  
  x + 1
```

```
five = increment 4
```

Functions

Kind of important in functional programming

```
increment : Int -> Int
increment x =
  x + 1
```

```
five : Int
five = increment 4
```

Type Inference

*Elm is smart, but you'd still
want to have explicit types*

```
-- constant
```

```
x : Int
```

```
x = 42
```

```
-- tuple
```

```
position : (Int, Int)
```

```
position = (3, 2)
```

```
-- object (called record)
```

```
person : { name : String, age : Int }
```

```
person =
```

```
  { name = "Erik"
```

```
  , age = 30
```

```
  }
```

Data

Constants, tuples og objects

```
type alias Coordinates = (Int, Int)
```

```
playerPosition : Coordinates  
playerPosition = (0,0)
```

```
type alias Discount = Int
```

```
studentDiscount: Discount  
studentDiscount = 10
```

Type Alias

*Allows us to define
new types*

```
type alias Customer =  
  { name: String  
    , age: Int  
  }
```

```
erik : Customer
```

```
erik =  
  { name = "Erik"  
    , age = 24  
  }
```

Type Alias

Works best with objects


```
type alias Customer =  
  { name: String  
    , age: Int,  
    , customerType: String  
    , studentDiscount: Int  
  }  
  
erik : Customer  
erik =  
  { name = "Erik"  
    , age = 25  
    , customerType = "Student",  
    , studentDiscount = 50  
  }
```

Example

*Three types of customers:
ordinary, students and
companies*

```
type alias Customer =  
  { name: String  
    , age: Int,  
    , customerType: String  
    , studentDiscount: Int  
    , companyName: String  
  }
```

```
erik : Customer
```

```
erik =
```

```
{ name = "Erik"  
  , age = 30  
  , customerType = "Corporate",  
  , studentDiscount = 0  
  , companyName = "BEKK Consulting"  
}
```

1. Easy to mistype

2. Hard to find the possible values of `CustomerType`

3. You don't get any help from the compiler

4. You end up with lots of fields with dummy values

```
type CustomerType  
  = Student  
  | Corporate  
  | Private
```

Union Types

Surprisingly useful!

```
type CustomerType
  = Student Int
  | Corporate String
  | Private
```

Union Types

Every branch can contain different values

```
type CustomerType = Student Discount | Corporate CompanyName | Private
```

```
getDiscount : CustomerType -> Discount
```

```
getDiscount class =
```

```
  case class of
```

```
    Student discount ->  
      discount
```

```
    Corporate name ->  
      0
```

```
    Private ->  
      0
```

Union Types

*Values are unwrapped
with pattern matching*

```
type Maybe a = Just a | Nothing
```

Maybe

*Eliminating the need for null and
undefined*

```
type Maybe a = Just a | Nothing
```

```
type alias Game =  
  { highscore: Maybe Int  
  }
```

Maybe

*Eliminating the need for null and
undefined*

```
type Maybe a = Just a | Nothing
```

```
type alias Game =  
  { highscore: Maybe Int  
  }
```

```
getHighscore : Game -> String
```

```
getHighscore game =  
  case game.highscore of  
    Just score ->  
      toString score  
    Nothing ->  
      "No highscore"
```

Maybe

The compiler will force us to handle all cases (similarly with ajax and other unsafe operations)


```
<div class="ninja">  
  <span>Banzai!</span>  
</div>
```

HTML

Creating it in Elm

```
main =  
  div [ class "ninja" ]  
    [ span []  
      [ text "Banzai!" ]  
    ]
```

HTML

Like React without JSX (hyperscript)

```
main : Html a
```

```
main =
```

```
  div [ class "ninja" ]  
      [ span []  
        [ text "Banzai!" ]  
      ]
```

HTML

What does a mean?

```
main : Html Msg
main =
  div [ class "ninja" ]
    [ span [ onClick DoSomethingCool ]
      [ text "Banzai!" ]
    ]
```

HTML

The Html-type includes the type that will be created by user interactions (like Redux actions)

```
-- our entire app state (store)
model: Model

-- represent data with html (react)
view: Model -> Html Msg

-- changes to app state (reducers)
update: Msg -> Model -> Model
```

The Elm Architecture

Which JavaScript-libraries would you need to get this out of the box?

1. React

Virtual DOM

2. Redux

Our built-in architecture

3. ImmutableJS

For full immutability

4. TypeScript eller Flow

For (a considerably weaker) type safety

5. ESLint

Enforcing code style and code-level sanity

Agenda

1. How Elm works

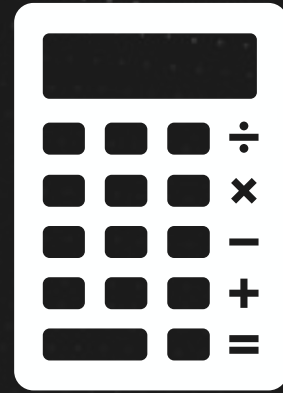
Compared to the JS of today

2. App Example

Counter app

3. Does Anyone Use Elm?

A few stories and examples



+ 0 -


```
module Main exposing (..)
```

```
import Html exposing (Html, button, div, text)
```

```
import Html.Events exposing (onClick)
```

```
main : Program Never Model Msg
```

```
main =
```

```
    Html.beginnerProgram
      { model = model
      , view = view
      , update = update
      }
```

```
-- MODEL
```

```
type alias Model =  
    Int
```

```
model : Model  
model =  
    0
```

```
-- VIEW
```

```
view : Model -> Html Msg
```

```
view model =
```

```
  div []
```

```
    [ button [ onClick Decrement ] [ text "-" ]
```

```
    , div [] [ text (toString model) ]
```

```
    , button [ onClick Increment ] [ text "+" ]
```

```
  ]
```

```
-- UPDATE
```

```
type Msg
  = Increment
  | Decrement
```

```
update : Msg -> Model -> Model
update msg model =
  case msg of
    Increment ->
      model + 1

    Decrement ->
      model - 1
```

Agenda

1. How Elm works

Compared to the JS of today

2. App Example

Counter app

3. Does Anyone Use Elm?

A few stories and examples

So...

Elm has been around for quite a few years, attracting attention and generating conference talks, but is it really **ready for production?**

Is anyone using it in their business-critical, user-facing applications? If so, **what's their stories?**

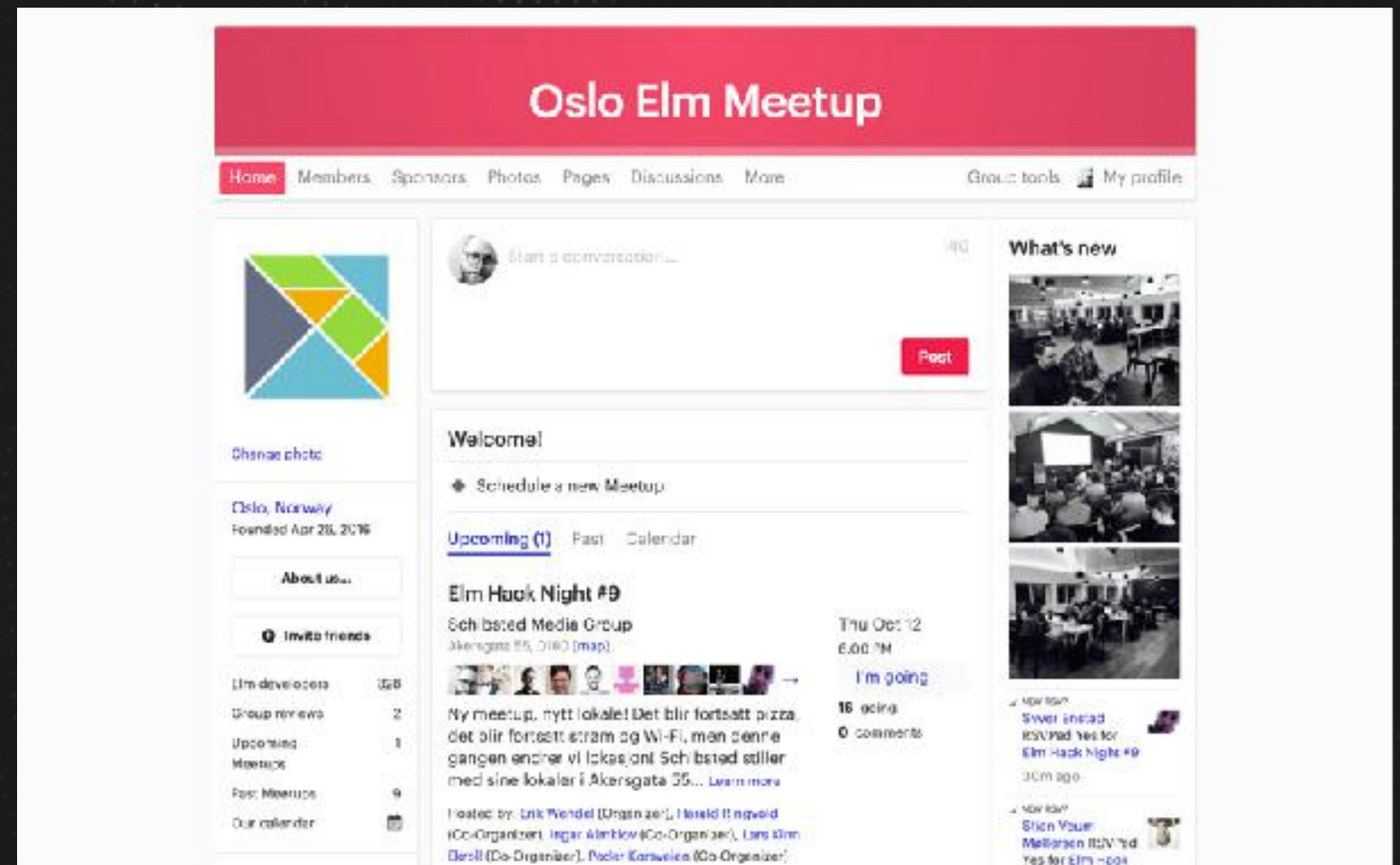
Oslo Elm Meetup

First meetup in May 2016

*Huge interest with no marketing
150 members and 42 attendees*

*January 2018:
398 members*

Monthly meetups



Oslo Elm Day

*One-day, single-track conference in
Oslo, June 2017*

105 attendees

10 speakers from 5 countries

All talks are on YouTube

Next edition: most likely fall 2018



Does Anyone Use Elm?

... yes!

Here's two example apps
written in Elm for different reasons



SWITCHAROO
CONFERENCE INFORMATION SYSTEM

KODERAKER
KODERAKER

SYS

Sys Biol

bouvet

sopro steria

JSON IN,
JSON OUT

JOBBI
SUND

Switcharoo

Conference information system

NSB
 (Norwegian Railways, like
 SJ)
 Summer interns doing
 seat reservation app

Velg plass

Tirsdag, 10/10/2017

Oslø S - Nelaug  Region mot Strøwanger

09:25 - T285 (31-30min)

- Standard plass
- NSB Familje
- NSB Komfort (+ kr. 90,- per setel)

Setel ikke valgt 

NSB Denne er togets kjøretning fra togets anreisestasjon. Toget endrer kjøretning i Kristiansand.



Velg setel i karer
 Setel ikke valgt

Bekreft setevalg

Velg setel() og trykk Bekreft setevalg for å lukke og gå videre.

Du kan endre setevalg ved å trykke på angre/tilbake. Setevalg kan ikke gjøres.

 Valgt setel

 Tilgjengelig setel

 Ikke reserverbart


Setelene er illustrert som ovenfra med hvit bakgrunn i lyggene.

WC: Toilett

 EXIT_LEFT

 EXIT_RIGHT

Din togbillett: kr 339,-

Deltoget 

[Gå videre](#)



Good morning, where do you want to go?

Travellers

1 adult

I will be travelling from

↓ Station

To

✎ Station

Time

✎ Now



Search

Get inspiration for your holiday in Norway here!



Summary Of Experience Reports

1. Many are experimenting with or using Elm
2. People are happy! (*100% "would use again"*)
3. Easy onboarding, especially for FPers and React-developers
4. World's biggest adoptor has around 250k LOC

Bad-use cases for Elm:

you're making a prototype

...or something else that's small

when dev speed trumps code quality

you use a lot of third party code (like map libs)

the team doesn't know func. prog. and don't want to learn

Great use-cases for Elm:

you know the refactors are coming

complex domain logic

you're re-creating Excel et. al in the browser

code correctness is especially important

team has little or no knowledge of javascript

Challenges, adressed!

JavaScript fatigue

Worrysome refactors

Dedicated frontend devs

Nail-biting deploys

Elm gives you..

a lang without runtime errors
superb refactoring support
frontend for the entire team

en delightful dev experience
with Redux built-in
and React's virtual DOM

a currently small ecosystem
a lang without a huge backer
some boilerplate

Is it a good deal?

I think so!

What do you think?

Did this make you curious?

1. Check out my free workshop material

github.com/ewendel/elm-workshop

(shameless plug)

2. Join the Elm slack team

The elm community is truly amazing

elmlang.herokuapp.com

3. Check out your local Elm meetup

meetup.com/topics/elm-programming/



Thanks for listening!

@ewnd1



*slides and workshop available at
is.gd/forward_elm*

twitter.com/osloelmday

BEKK