

# Welcome

## learn & hack Elm

**Initial Elm Zürich** - 19th May 2016

Host **XIAG AG**

## **Program: (stay tuned)**

- 1. Welcome and Introduction by Ivan & Fabio**
- 2. Future events and activities ~ 5min by Fabio**
- 3. Introduction to Elm ~ 20min by Fabio**
- 4. A look at Elm-Css ~ 30min by Jasper**
- 5. Explore, learn, share**

# Future events and activities

learn / hack Elm

# Learn Elm

Everyone is welcome! Interested in Elm? Are you a **young hacker**? What are you waiting for? ! (Do you want to give programming a try? This is your chance! )

## Topics/Activities:

We learn, hack, connect, share, discuss..

- Learn how the Web works with Elm (Web Development)
- **Mentorship** (and improve your communication ability)
- Create web sites/apps, graphics and games with Elm
- Get into functional programming with Elm
- Make cool stuff with Elm - together!
- Learn new tricks and skills
- Intro for **newcomers**
- Learn Elm!!
- ..

# Hack Elm

Everyone is welcome! **Be active and help to make Elm and the Elm-ecosystem even more awesome!**

## Topics/Activities:

We hack, learn, connect, share, discuss..

- Interoperability with HTML, CSS and JavaScript
- Functional reactive programming with Elm
- Future of web & graphical programming
- Functional programming with Elm
- **Improve projects, libraries, docs**
- **Collaborate on projects**
- Build useful stuff
- Hack Elm!!
- ..

1. passive phase -> talks
2. active phase -> learn, explore, share

<http://fabio.filli.io/elm-zurich/>

fabio.filli.io

@FabioFilli

Fysi GmbH

Fysi.World - Build Your own World on the Internet

I believe

Everyone should learn how to program - with Elm

# Elm

Initial Elm Zürich - 19th May 2016



Elm

What's  
that?

Created by **Evan Czaplicki**

## **Mission Impossible**

To make GUI programming more pleasant

➔ **Functional Programming**

To make programming more accessible



Elm is a

# Functional

## Programming Language

ML (1973)

SML (1990) OCaml (1996)

**Haskell** (1990)

F# (2005)

Elm 0.17  Erlang (1986)

..Erlang style of concurrency

ufff..

# functional

programming

difficult..      for academics..  
                         with a PHD in Math..

$$|D(T, \varepsilon, a, b)| \leq 2$$

$$\varphi(\sigma_1 t) \varphi(\sigma_2 t) = \varphi(\sqrt{\sigma_1^2 + \sigma_2^2} t)$$

$$\sum_{k=1}^r \int_{b_k v}^{x+b_k v} \left( \int_0^b \psi_k^*(\tau) d\tau \right) dt - x \int_0^{b_k v} \psi_k^*(\tau) d\tau = \frac{x^2}{2} B(v) + \int_0^x (x-u) \sum_{k=1}^r \psi_k^*(u) du \quad A(v) = \sum_{k=1}^r b_k \psi_k^*(v)$$

$$l(\alpha) = \frac{\sum_{k=1}^r p_k^* \log_2 \frac{1}{p_k}}{\sum_{k=1}^r p_k^*}$$

$$c_{ik} \sigma_k^2 = \lambda; c_{ik}$$

$$y = \phi(x) = \frac{1}{\sqrt{2\pi}}$$

Academia

$$i^2 := -1; j^2 := -1; k^2 := -1 \quad \lim_{n \rightarrow \infty} \frac{\binom{2n}{n+c}}{\binom{2n}{n}}$$

$$S_n = A_n U \pi A_n$$

$$W_k = \binom{n}{k} p^k (1-p)^{n-k}$$

$$P(\eta < y | \xi = x) = \sup_{y' < y, y' \in \mathbb{R}} P(\eta < y' | \xi = x)$$

$$dt \quad P(\eta_\infty < x) = F(x)$$

$$|A_n| = \frac{n!}{2} \left| \int_{|x| > A} f(x) \log_2 \frac{1}{f(x)} dx \right| < \varepsilon$$

$$\int_{-\alpha_k}^{+\alpha_k} dG_k(x) \geq \frac{1}{2} \sum_{k=-\infty}^{+\infty}$$

great ideas

$$g^{-1} \cdot g = e \quad \bigcup_{i=1}^{n-1} M_i; \bigcap_{n=0}^{\infty} X_n$$

$$y = \sqrt{\frac{\lambda_n}{\nu_n}} \left( \frac{\eta_{2n}}{\sqrt{\lambda_n}} + \frac{\eta_{2n} - \eta_{2n}}{\sqrt{\lambda_n}} \right)$$

$$f(t|y) = \frac{2e^{\frac{y^2}{2}}}{\sqrt{2\pi}} \int_{\frac{y}{\sqrt{t}}}^{+\infty} \frac{e^{-\frac{u^2}{2}} du}{\left(1 - \frac{y^2}{u^2}\right)^{\frac{3}{2}}}$$

$$\Delta N = \frac{y}{s}$$

top research

$$H_r(x) = \frac{G_r(x)}{1 + G_r(x)} \quad \int_0^{\infty} \varphi(t) dt$$

$$U_n^{+c} = \binom{2n}{n} - \binom{2n}{n}$$

$$f_{n-1}(t) = \int_0^1 f_n(u) f_1(t-u) du = \frac{\lambda^{n+1} t^n e^{-\lambda t}}{n!}$$

$$\lim_{t \rightarrow 0} (f(t)) = 0$$

$$C_{iv} = \sum_{j=1}^n a_{ij} b_{jv}$$

$$\log \varphi(t) = i\gamma t - c|t|^\alpha \left[ 1 + i\beta \frac{t}{|t|} \omega(t,u) \right] \quad B(v) = \sum_{k=1}^r \psi^*(b_k v)$$

$$\int_{-\infty}^{\infty} e^{-\frac{u^2}{2}} du = F(x) \left( \frac{1}{\sqrt{2\pi}} \right)^{-1}$$

$$|\Psi_S(t)| = \left| \int_{-\infty}^{\infty} e^{itx} dF(x) \right| \leq \int_{-\infty}^{\infty} e^{-vx} dF(x) = \varphi_S(iv)$$

$$\lim_{n \rightarrow \infty} P \left( \frac{j_{n+1} - k_n - \log \frac{1}{q}}{\sqrt{\frac{1-q}{q}}} \right)$$

$$C_n(\alpha) \geq \frac{n!}{\prod_{k=1}^r n_k(\alpha)!}$$

$$\left| \frac{\sinh t}{t} [\varphi(t) e^{-itx} + \frac{t}{\sinh t} \varphi(t) e^{-itx}] \right|$$

$$\prod_m = \prod_r | \prod_{m-r}$$

$$g^{-1} N g =$$

VIP community

$$) = \frac{p_k^*}{\sum_{j=1}^r p_j^*}$$

encrypted information:  $\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$

$$P_n(k) = \frac{c(w)}{p^k}$$

$$P \left( \lim_{n \rightarrow \infty} \sup \frac{|h_n|}{\sqrt{2n \log \log n}} \leq 1 \right) = 1 \quad (p+1) = 1 - |$$

$$f: X \rightarrow X \cap W$$

$$Q(A) = \int_A \chi(\omega) dP \quad l'(\alpha) = -\log_2 \left( \frac{\sum_{k=1}^r p_k^* \log_2 \frac{1}{p_k}}{\sum_{k=1}^r p_k^*} - \left( \frac{\sum_{k=1}^r p_k^* \log_2 \frac{1}{p_k}}{\sum_{k=1}^r p_k^*} \right)^2 \right)$$

$$fg(u_i) = f \left( \sum_{j=1}^{\dim V_2} a_{ji} v_j \right) = \sum_{j=1}^{\dim V_2} a_{ji} \left( \sum_{k=1}^{\dim V_3} b_{kj} w_k \right) \frac{\binom{2k}{k}}{2^{2k}} \approx$$

$$q \left( c^{-x} \sqrt{\frac{1-q}{nq}} - 1 \right) = -x \sqrt{\frac{q(1-q)}{n}} + o \left( \frac{1}{n} \right)$$

$$\prod_{k=1}^r \left[ g_k \left( \frac{t}{\sqrt{N_0}} \right) \right]^{N_0 \alpha_k} = e^{-\frac{t^2}{2}}$$

$$P_{j,k}^{(m)} = \sum_{c=0}^{\infty} P_{j,c}^{(r)} P_{c,k}^{(m-r)} \quad \frac{1}{2\pi} \int_{-\infty}^{\infty} \operatorname{Re} \left\{ \varphi(t) \frac{e^{-ita} - e^{-itb}}{it} \right\} dt$$

$$P(\| \omega \| > \varepsilon) :$$

$$\liminf_{N \rightarrow \infty} \int_{-\infty}^{+\infty} f_N(x)^\alpha dx \geq \int_{-\infty}^{+\infty} f(x)^\alpha dx$$

$$M(\|\delta_j - 1\|^s) = \int |x-1|^s e^{-x} dx$$

$$\lim_{N \rightarrow \infty} \int_{-A}^{+A} f_N(x) \log_2 \frac{1}{f_N(x)} dx = \int_{-A}^{+A} f(x) \log_2 \frac{1}{f(x)} dx$$

$$N\varepsilon_n - \varepsilon_k = \binom{2n}{n+k}$$

# functional programming

```
isPositive n =  
  if n > 0 then "jep" else "nop"
```

```
sin()
```

```
isPositive 7
```

```
jep
```

```
sin( $\pi/2$ )
```

```
1
```

```
isPositive -7
```

```
nop
```

# functional

## programming

it's  
**statically typed**

Elm enforces safe programming practices at the language level.

**No "runtime errors"**

**No "null"**

**No "undefined is not a function"**

hell yeah :) more pleasant!!

don't forget immutability / pure functions

Created by **Evan Czaplicki**

**Mission Impossible**

To make GUI programming more pleasant

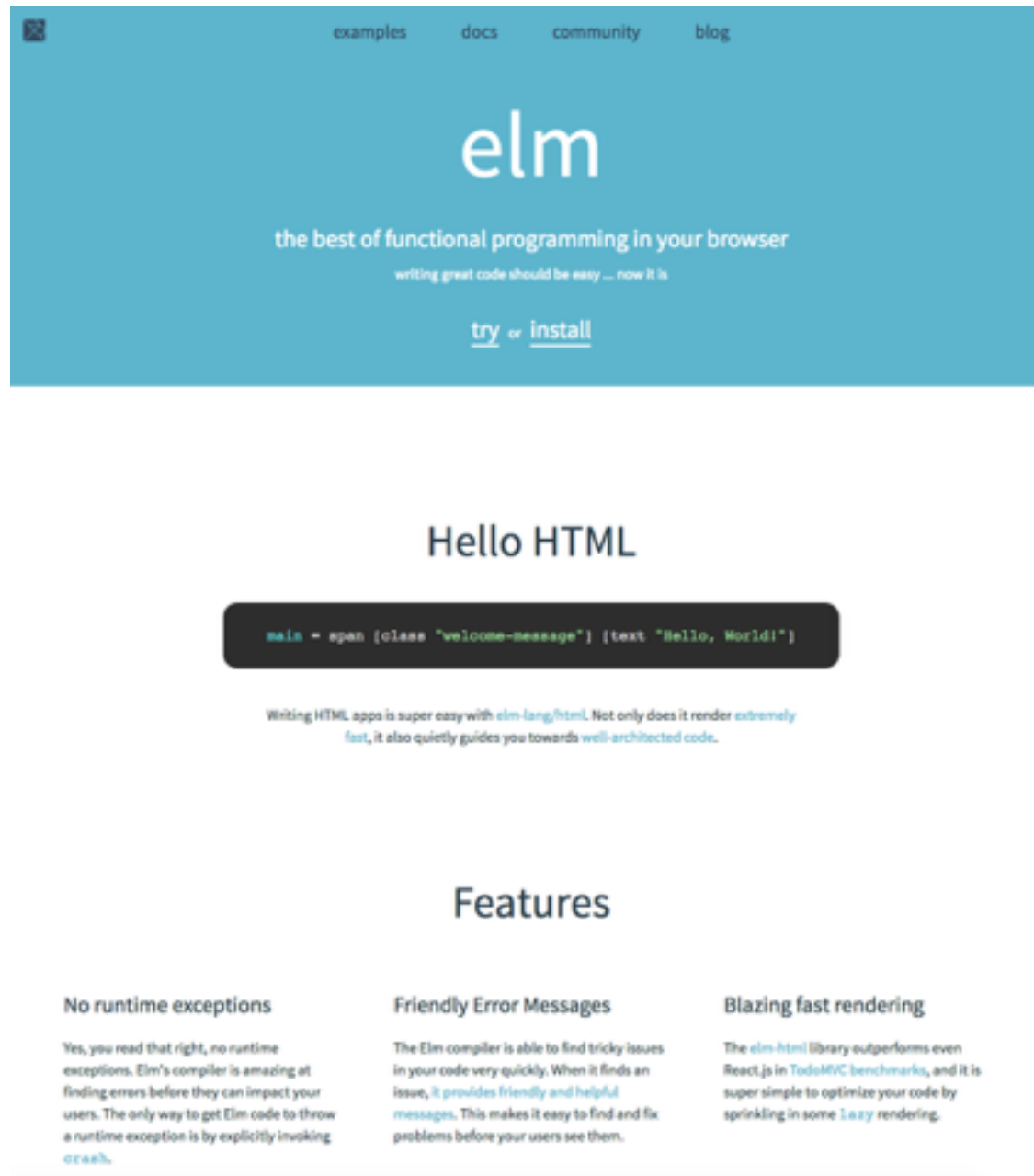
➔ **Functional Programming**

To make programming more accessible

➔ **Web**



# Websites



The screenshot shows the Elm website homepage. At the top, there is a navigation bar with links for 'examples', 'docs', 'community', and 'blog'. The main header features the 'elm' logo in a large, white, sans-serif font on a teal background. Below the logo, the text 'the best of functional programming in your browser' is displayed, followed by the tagline 'writing great code should be easy ... now it is'. Two links, 'try' and 'install', are provided. The 'Hello HTML' section includes a code snippet in a dark box: `main = span [class "welcome-message"] [text "Hello, World!"]`. Below this, a paragraph describes how easy it is to write HTML apps with `elm-lang/html`. The 'Features' section is divided into three columns: 'No runtime exceptions', 'Friendly Error Messages', and 'Blazing fast rendering', each with a brief description of the feature.

examples docs community blog

# elm

the best of functional programming in your browser

writing great code should be easy ... now it is

[try](#) or [install](#)

## Hello HTML

```
main = span [class "welcome-message"] [text "Hello, World!"]
```

Writing HTML apps is super easy with `elm-lang/html`. Not only does it render *extremely fast*, it also quietly guides you towards *well-architected code*.

## Features

### No runtime exceptions

Yes, you read that right, no runtime exceptions. Elm's compiler is amazing at finding errors before they can impact your users. The only way to get Elm code to throw a runtime exception is by explicitly invoking `crash`.

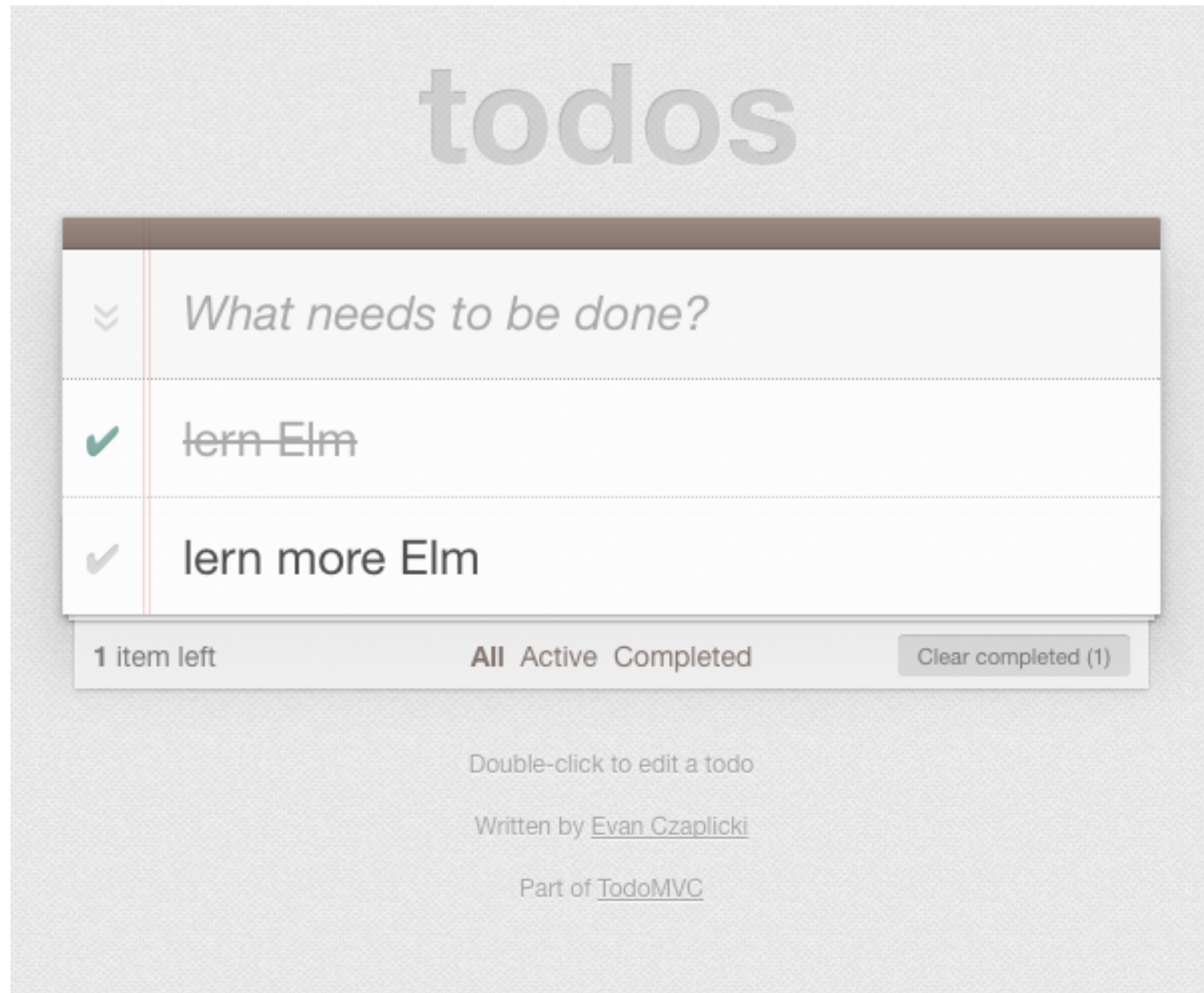
### Friendly Error Messages

The Elm compiler is able to find tricky issues in your code very quickly. When it finds an issue, it *provides friendly and helpful messages*. This makes it easy to find and fix problems before your users see them.

### Blazing fast rendering

The `elm-html` library outperforms even React.js in *TodoMVC benchmarks*, and it is super simple to optimize your code by sprinkling in some *lazy* rendering.

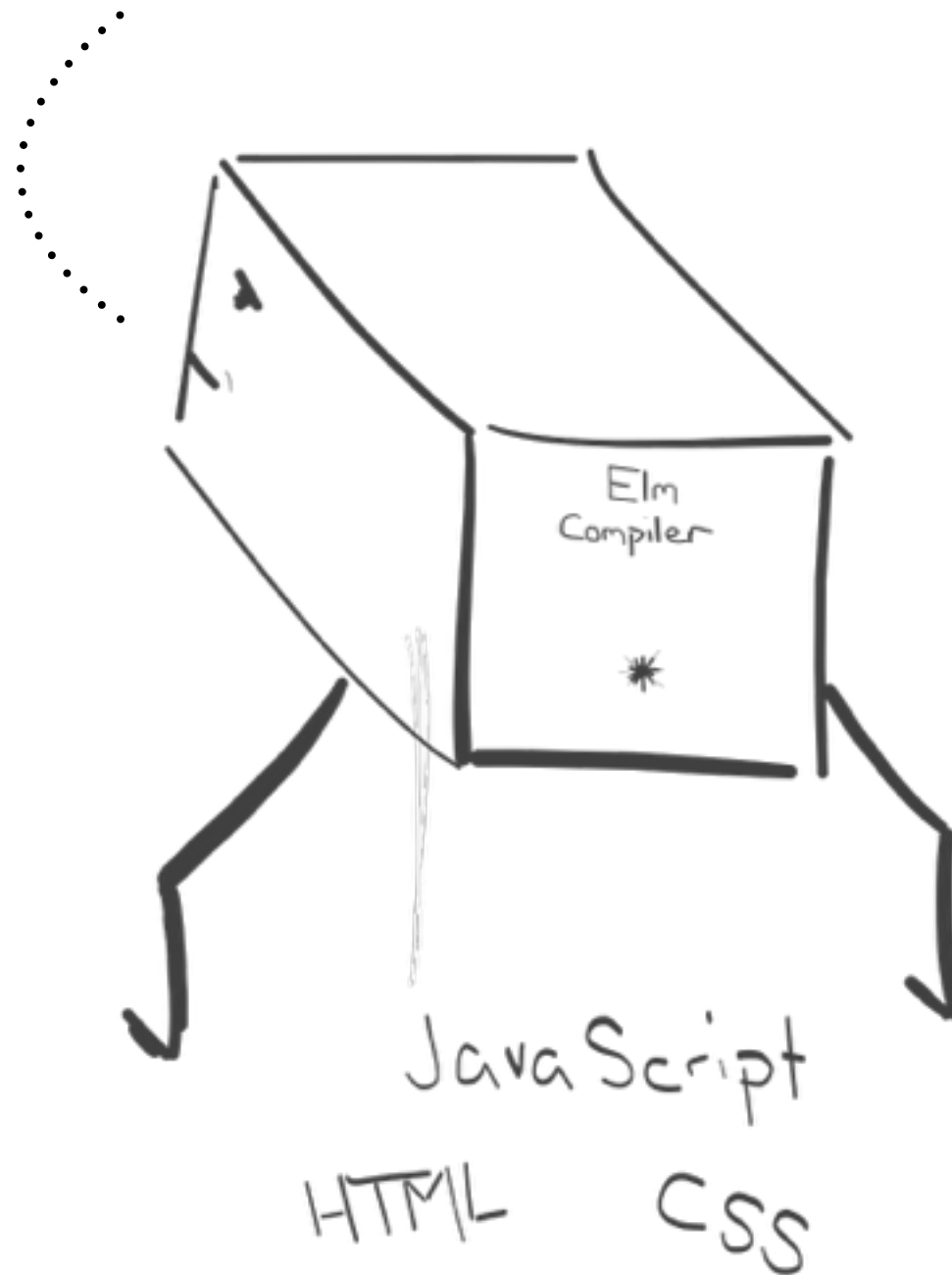
# interactive Websites



but there is already

# JavaScript

# No "runtime errors"





**Elm's awesomeness**

**No "runtime errors"**

**No "null"**

**No "undefined is not a function"**

**Well-Organised Code**

**Awesome Error Messages**

**Fast HTML rendering**

**Libraries with semantic versioning**

**JavaScript interoperable**

**Live Debugger**

**...**

**No “runtime errors”**

**No “undefined is not a function”**

**No “null”**

# **Well-Organised Code**

**Awesome Error Messages**

**Fast HTML rendering**

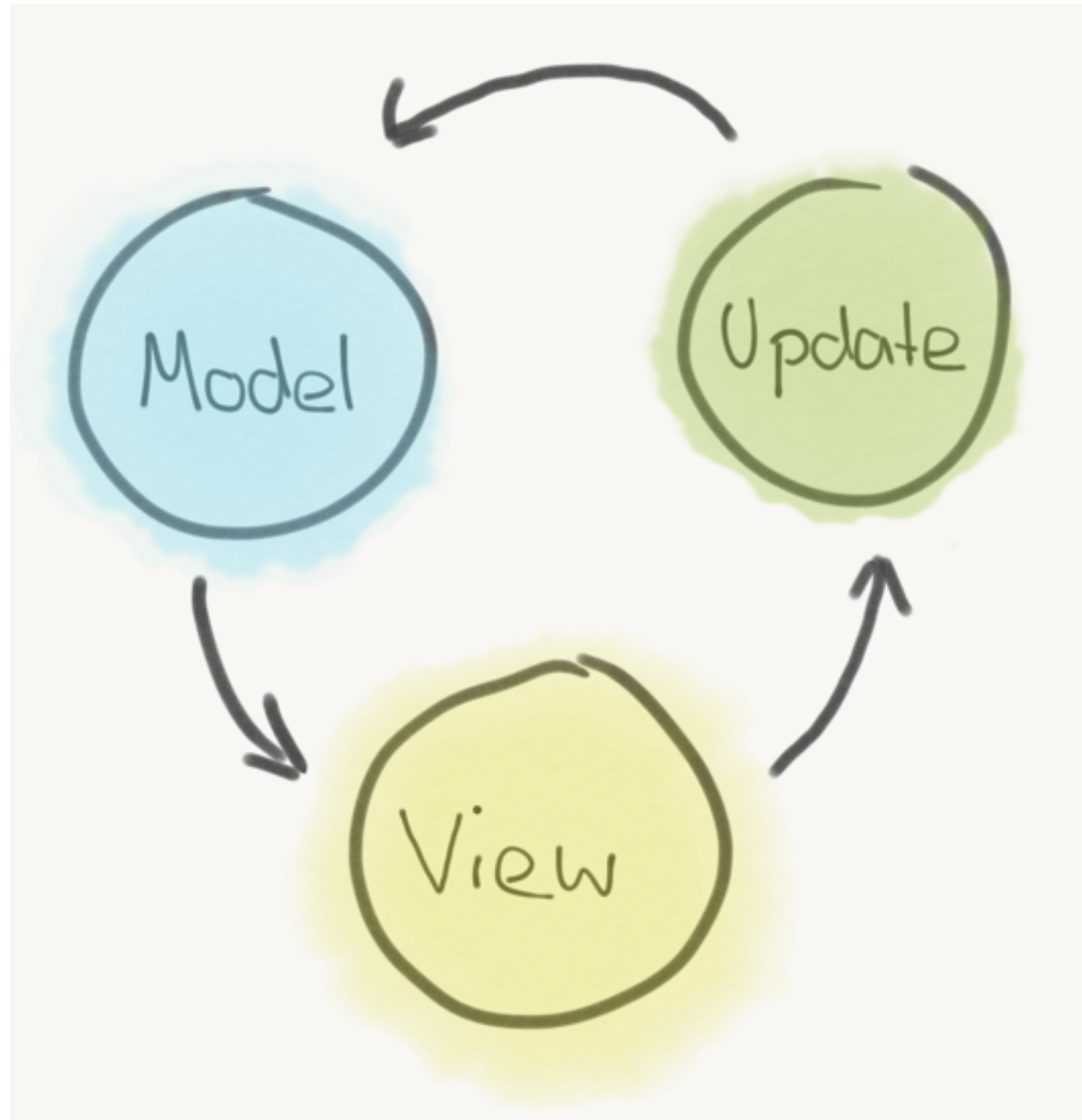
**Libraries with semantic versioning**

**JavaScript interoperable**

**Live Debugger**

**...**

# Elm stylee





# Well-Organised Code

**Model** - the state of your app

**Update** - a way to update your state

**View** - a way to view your state as HTML

```
1  module HtmlAppBeginner exposing (..)
2
3  import Html.App as Html
4  import Html exposing (..)
5
6
7  main =
8    Html.beginnerProgram
9      { model = model
10        , view = view
11        , update = update
12        }
13
14
15  -- MODEL
16
17
18  type alias Model = {...}
19
20
21  -- UPDATE
22
23
24  type Msg = Sth | ...
25
26
27  update : Msg -> Model -> Model
28  update msg model =
29    case msg of
30      Sth -> ...
31      ...
32
33
34  -- VIEW
35
36
37  view : Model -> Html Msg
38  view model =
39    ...
40
```

# Elm


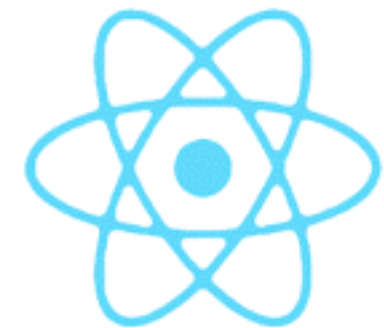

## Architecture

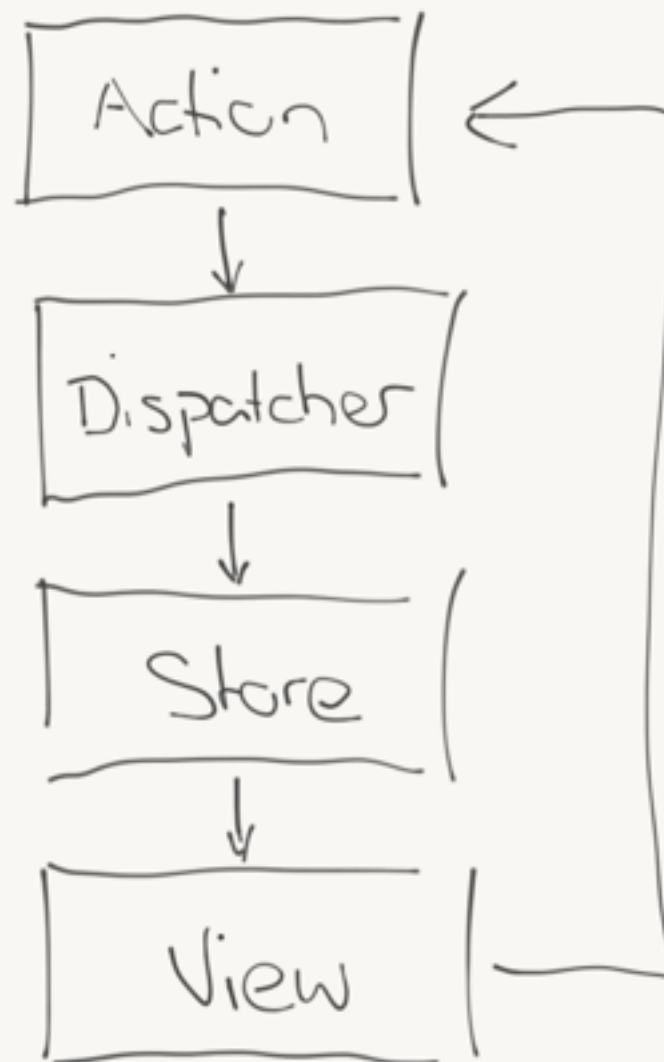
Model - the state of your app

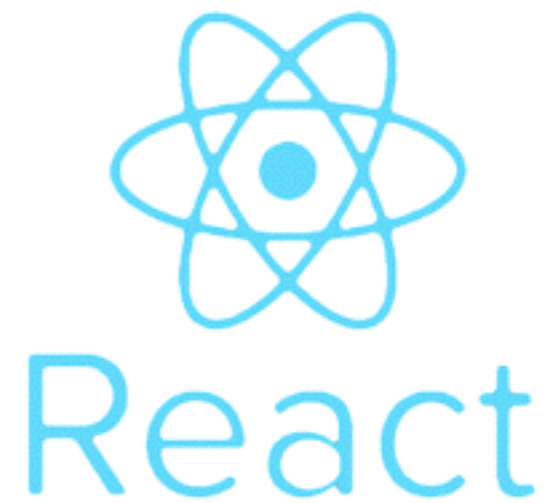
Update - a way to update your state

View - a way to view your state as HTML

```
1  module HtmlAppBeginner exposing (..)
2
3  import Html.App as Html
4  import Html exposing (..)
5
6
7  main =
8      Html.beginnerProgram
9          { model = model
10            , view = view
11            , update = update
12            }
13
14
15  -- MODEL
16
17
18  type alias Model = {...}
19
20
21  -- UPDATE
22
23
24  type Msg = Sth | ...
25
26
27  update : Msg -> Model -> Model
28  update msg model =
29      case msg of
30          Sth -> ...
31          ...
32
33
34  -- VIEW
35
36
37  view : Model -> Html Msg
38  view model =
39      ...
40
```

 Flux +  React +  **Redux**



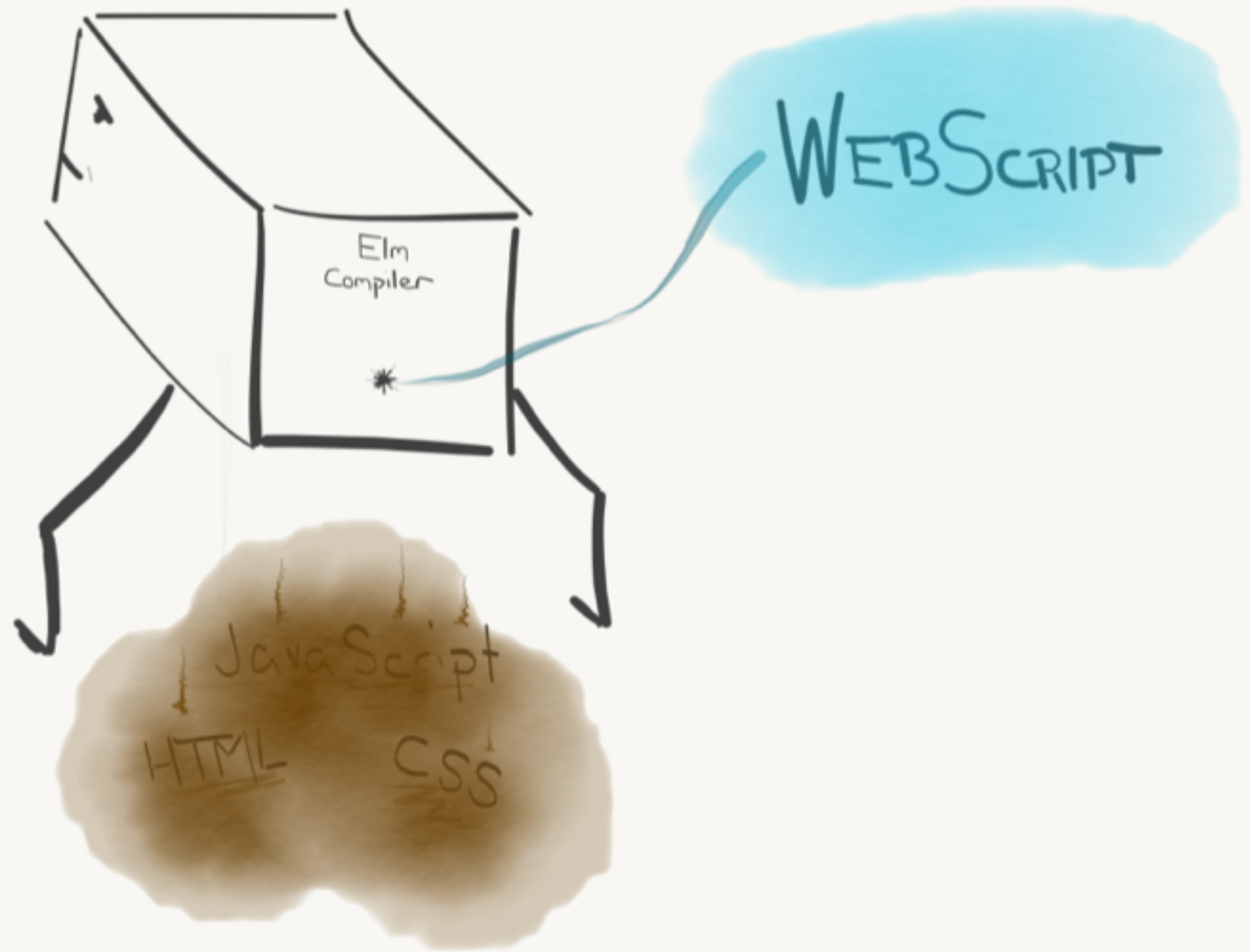


Angular



Cycle.js

# JavaScript



Elm

What's  
that?

# Elm rocks!!

It's made for the **Web**

It's **functional**

It's **easy** to learn

# Elm rocks!!

## The Tools

### Elm Platform

elm-compiler   elm-reactor   elm-repl  
elm-make   elm-package

### Time-Traveling Debugger

Online editor [elm-lang.org/try](http://elm-lang.org/try)



Elm

For me?

Pro  
developer

&

Young Hacker  
people new to programming

# Where to start..

<http://elm-lang.org/>

The Guide - An introduction to Elm

<http://guide.elm-lang.org/>

<http://package.elm-lang.org/>

core, html..

check out the **examples**

and **start** right away

with the online editor

~~0.16~~

0.17

# Elm is now easier than ever to learn

Every Elm project will define main to be some sort of Program.

```
7  main =  
8    Html.beginnerProgram  
9      { model = model  
10      , view = view  
11      , update = update  
12      }
```

Html.App  
Cmd Sub

```
7  main =  
8    Html.program  
9      { init = init  
10      , view = view  
11      , update = update  
12      , subscriptions = subscriptions  
13      }
```

WebSocket support  
geolocation page-visibility

..Erlang style of concurrency