ELM

FUNKTIONAL & REAKTIV
IM BROWSER

WER?

BASTIAN KROL

Developer at codecentric

@bastiankrol

basti1302

bastian.krol@codecentric.de

- Software Development
- Agile Coaching
- Consulting
- Continuous Delivery
- Agile Software Factory
- Software Architecture
- DevOps
- Performance Tuning
- Big Data
- Operations

WE'RE HIRING!

- Karlsruhe
- Stuttgart
- Frankfurt
- München
- Berlin
- Hamburg
- Solingen
- Düsseldorf
- Münster
- Dortmund

https://www.codecentric.de/karriere/offene-stellen/

WAS IST ELM?

- Funktionale Programmiersprache
- Statisches Typsystem
- Reactive (FRP)
- Kompiliert zu JavaScript
- Open Source
- Makes Web Development Delightful

WEN INTERESSIERT´S?

MARKETING BLURBS

- Clean syntax
- No runtime exceptions
- Blazing fast rendering
- Libraries with SemVer guarantees
- Smooth JavaScript interop
- Time-traveling debugger

FEHLERSUCHE REDUZIEREN

Bug by Nestor Ferraro / CC BY 2.0

ALLES IST ZUSTANDSLOS

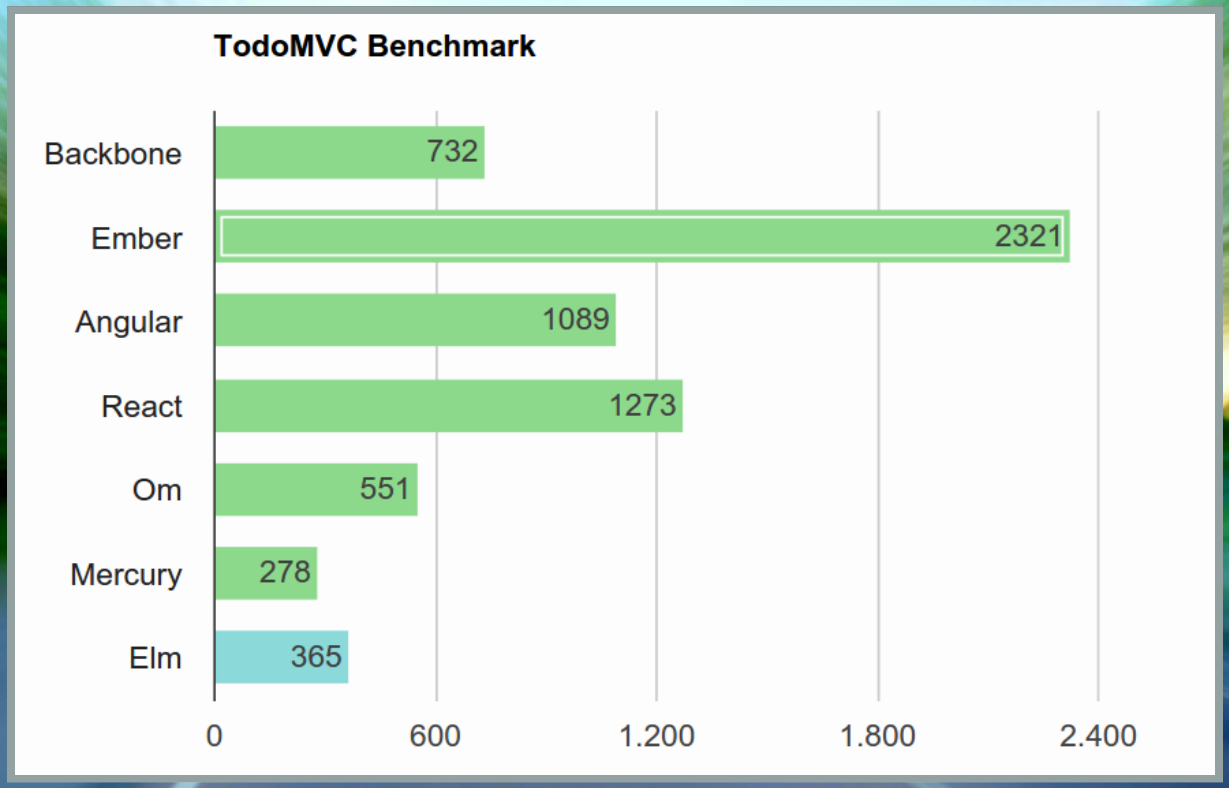
ALLES IST UNVERÄNDERLICH

Carved Stone" by Kit Carruthers / CC BY 2.0

swat by Marc Dalio / CC BY-NC-ND 2.0


ALTE ELM-WEISHEIT
Wenn es kompiliert, funktioniert es auch!

SCHNELLES RENDERING



SEMVER GARANTIEN

1. Bibliothek veröffentlichen
2. Öffentliche API verändern
3. Als Minor/Bugfix Update veröffentlichen
4. =>Nope
5. Ermäglicht durch statisches Typsystem

SYNTAX GRUNDLAGEN

HELLO WORLD

Elm

```
import Html

main = Html.text "Hallo Dortmund!"
```

Ergebnis

Hallo Dortmund!

FUNKTIONEN

Elm

```
import Html

greet str = Html.text str

main =
  greet "Hallo Dortmund!"
```

Ergebnis

Hallo Dortmund!

TYP ANNOTATIONEN

Elm

```
import Html exposing (Html)

greet : String -> Html
greet str =
  Html.text str

main : Html
main =
  greet "Hallo Dortmund!"
```

Ergebnis

Hallo Dortmund!

PARAMETER-TYPEN & RÜCKGABE-TYPEN

Elm

```
import Html exposing (Html)
import String

greet : Int -> String -> Html
greet number string =
  Html.text
    (String.repeat number string)

main : Html
main =
  greet 3 "Hallo Dortmund! "
```

Ergebnis

Hallo Dortmund!
Hallo Dortmund!
Hallo Dortmund!

FUNKTIONS-APPLIKATIONS-OPERATOR

AKA DATEN-PIPELINE

Elm

```
greet : Int -> String -> Html
greet number string =
  string
  |> String.repeat number
  |> Html.text

main : Html
main =
  greet 3 "Hallo Dortmund! "
```

Ergebnis

Hallo Dortmund!
Hallo Dortmund!
Hallo Dortmund!

FUNCTIONALE REAKTIVE PROGRAMMIERUNG

IN ELM

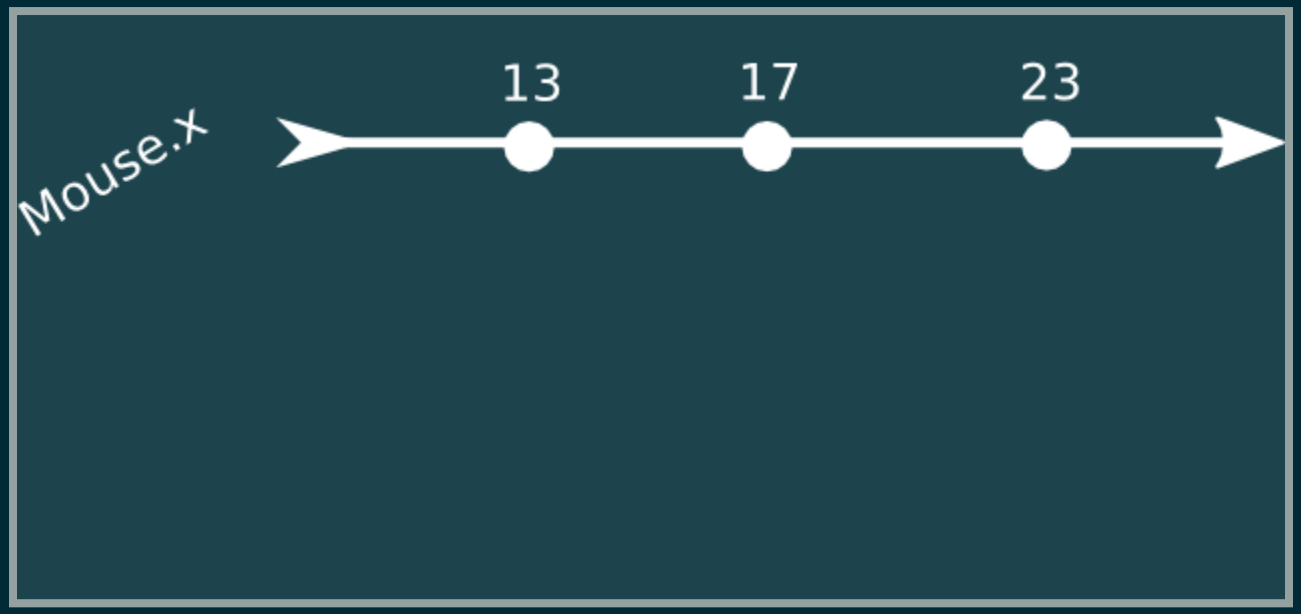
SIGNALS

Elm

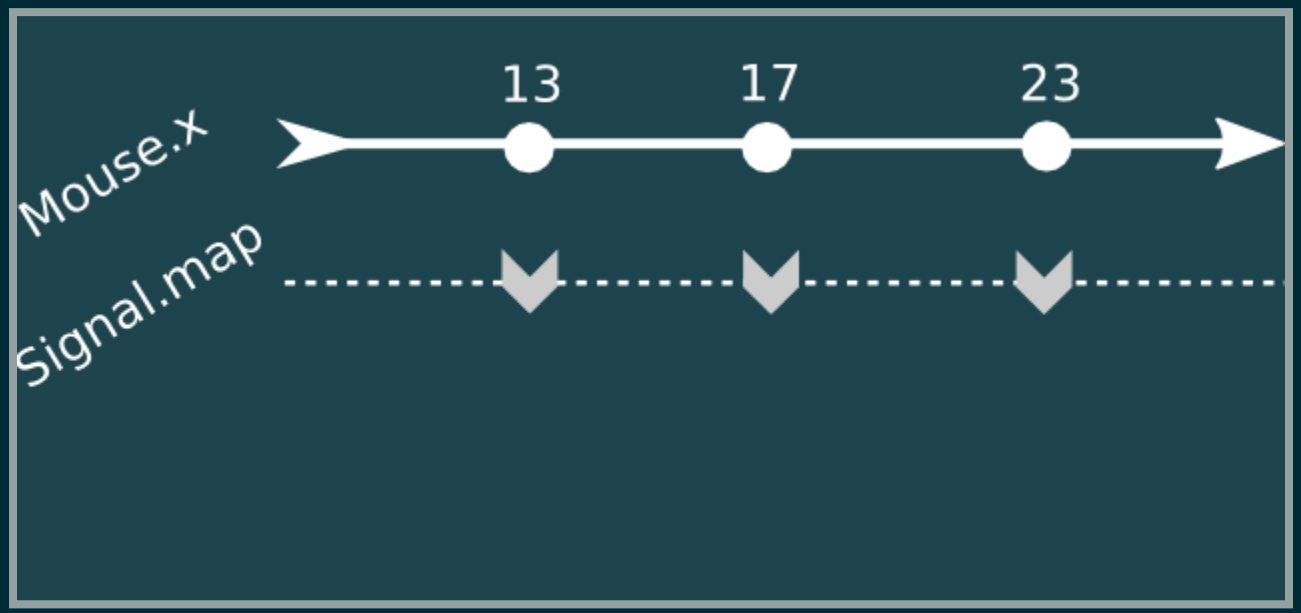
```
import Html exposing (Html)
import Mouse

toHtml : Int -> Html
toHtml x =
  Html.text (toString x)

main : Signal Html
main =
  Signal.map toHtml Mouse.x
```

Ergebnis

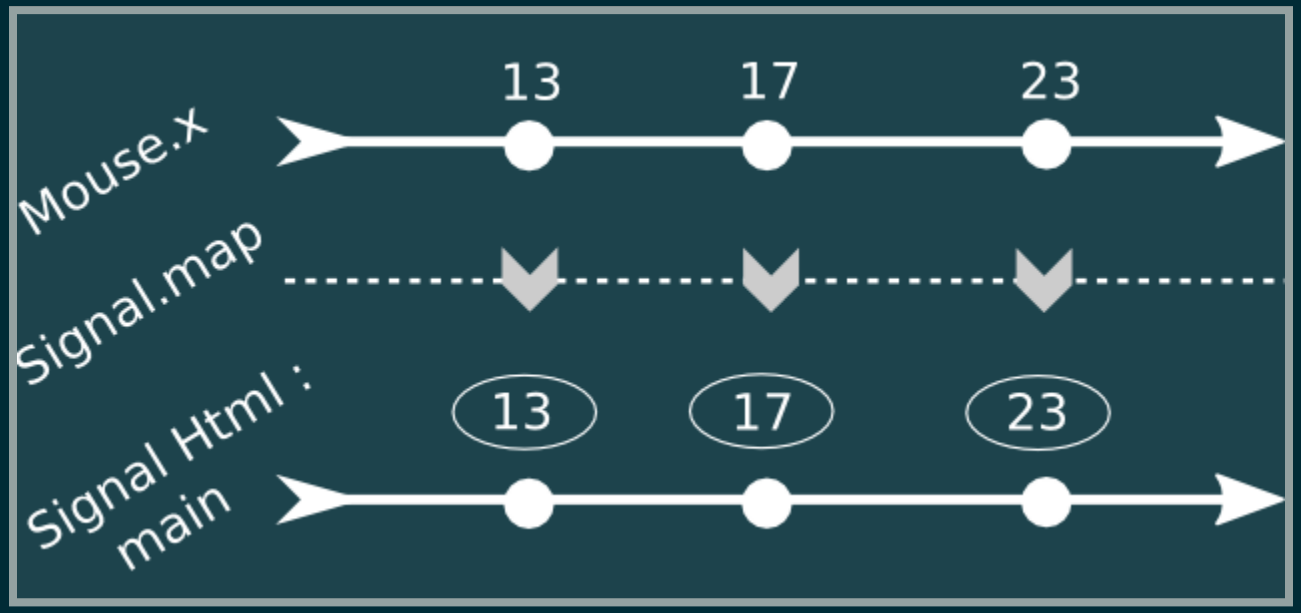
SIGNALS (CONT.)



SIGNALS (CONT.)



SIGNALS (CONT.)



SIGNALEN VERARBEITEN

- `merge` – Mehrere Signale zusammen führen
- `map` – Wert im Signal transformieren
- `filter` – Bestimmte Werte filtern
- ... & noch viele andere interessante Sachen ... :)

MEHR ÜBER SIGNALE

http://elm-lang.org/guide/reactivity
https://github.com/evancz/elm-architecture-tutorial

DER ELM-COMPILER IST DEIN FREUND

The dog and his man by C. I. Ştefănescu / CC BY 2.0

TYPOS

```elm
myFunction str =
  String.repeat 3 str

main : Html
main =
  Html.text (myFunctino "Elm")
```

```
-- NAMING ERROR ---------------------- errors/Spelling.elm

Cannot find variable `myFunctino`

9|    Html.text (myFunctino "Elm")
                 ^^^^^^^^^^
Maybe you want one of the following?

    myFunction
```

TYPE MISMATCH

```
subMismatch =
{ name = "Alice", age = 24 } == { name = "Bob", age = "30" }
```

```
-- TYPE MISMATCH ---------------------- errors/SubMismatch.elm

The right argument of (==) is causing a type mismatch.

2| { name = "Alice", age = 24 } == { name = "Bob", age = "30"
                                    ^^^^^^^^^^^^^^^^^^^^^^^^^^
(==) is expecting the right argument to be a:

    { ..., age : number }

But the right argument is:

    { ..., age : String }
```

DER VERGESSENE FALL

```
type Colour = Red | Green | Blue

colourToHex : Colour -> String
colourToHex colour =
  case colour of
    Red   -> "#f00"
    Green -> "#0f0"
```

DER VERGESSENE FALL

```
-- MISSING PATTERNS --------------- errors/ForgottenCase.elm

This `case` does not have branches for all possibilities.

5|>  case colour of
6|>    Red   -> "#f00"
7|>    Green -> "#0f0"

You need to account for the following values:

    Main.Blue

Add a branch to cover this pattern!
```

LAUFZEIT-FEHLER?

JavaScript

```
function repeatFirst(list) {
  var repeated = list[0].repeat(3);
  alert(repeated);
}
```

```
var list1 =
  [ 'hip', 'hop', 'hooray!' ];
...
repeatFirst(list1);
```

Run

```
var list2 = [];
```

Run

```
...
repeatFirst(list2);
```

COMPILE-FEHLER > LAUFZEIT-FEHLER

```
repeatFirst list =
  let elem = List.head list
  in String.repeat 3 elem
```

```
> elm-make RepeatFirst.elm --output repeat-first.html
-- TYPE MISMATCH ---------------------------- RepeatFirst.elm

The 2nd argument to function repeat is causing a mismatch.

8|       String.repeat 3 elem
                        ^^^^
Function `repeat` is expecting the 2nd argument to be:
    String
But it is:
    Maybe a

Hint: I always figure out the type of arguments from left
to right. If an argument is acceptable when I check it, I
assume it is "correct" in subsequent checks. So the problem
```

KORRIGIERTE FASSUNG

```
repeatFirst list =
  let elem = List.head list |> Maybe.withDefault ""
  in String.repeat 3 elem
```

```
> elm-make RepeatFirst.elm --output repeat-first.html
Success! Compiled 1 modules.
Successfully generated repeat-first.html
```

SCHUTZ VOR LAUFZEIT-FEHLERN

ZUSAMMENFASSUNG

- *Keine Laufzeitfehler!*
- Refactoring ist *einfach* und *sicher*
- Kohärenter Ansatz für Benutzer-Interaktion mit *Signalen*
- Elm ist *schnell*
- Kann in *bestehende* JS-Projekte eingeführt werden
- Gleichförmige Struktur für Apps (Elm Architecture)


Q. E. D.
ELM MAKES
WEB DEVELOPMENT
DELIGHTFUL

LINKS:

- http://elm-lang.org/
- Online Editor/Sandbox
- Elm Friday (Tutorial/Blog Series)

FRAGEN?

DANKE!

 @bastiankrol

 basti1302

 bastian.krol@codecentric.de