Good UI/UX project defaults with types

Hello!

 @realmario

Yehuda Katz
@wycats

Following

Front end software development is:

- real-time (instant load, 60fps)
- distributed, incremental (synchronize remote data as needed)
- asynchronous
- reactive (react to user actions in realtime)

Front end is the hardest kind of dev I do. The folks who do it every day are heroes.

2:53 AM - 15 Nov 2017

1,670 Retweets **4,485** Likes

Direct Messages

Looks like you don't have any recent direct messages, why don't you send one to a teammate and say hi!


Tweet
Kris Jenkins
@krisajenkins
Son: What do frog's legs taste like?
Me: Imagine you were really hungry, and someone brought almost no chicken at all.
04/06/2016 08:37
0 RETWEETS
0 LIKES

A delightful language for reliable webapps.

Generate JavaScript with great performance and no runtime exceptions.

```
likes : Int

likes = 0

likes = -1
```

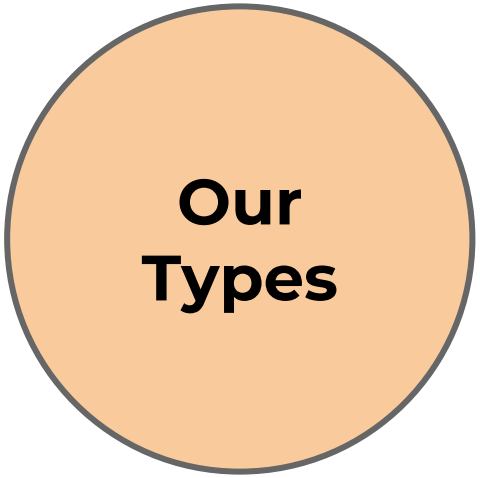
```
points : Int

points = 0

points = -1
```

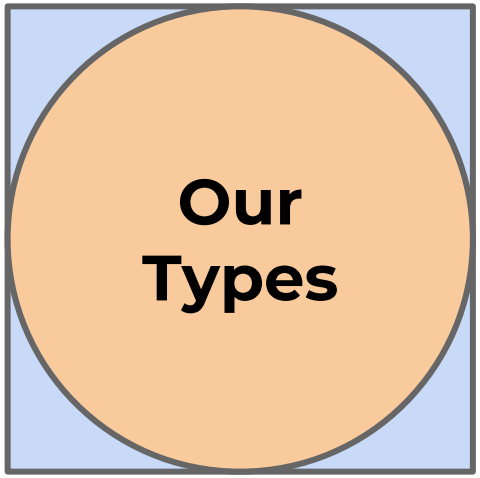
```
friends : List String

friends = ["Jane", "Bob"]

friends = []
```

```elm
friends : Maybe (List String)

friends = Nothing

friends = Just ["Jane", "Bob"]

friends = Just []
```

```elm
friends : Result Error (List String)

friends = Error "Something went wrong!"

friends = Success ["Jane", "Bob"]

friends = Success []
```


Our
Problem


Our
Types


Our
Types

remotedata

Makes it easier to represent the real state of a remote data fetch and handle it properly.

https://github.com/krisajenkins/remotedata

- Not asked
- Loading
- Failure with some error
- Success with some result

- NotAsked
- Loading
- Failure (error)
- Success (result)

```
type WebData
  = NotAsked
  | Loading
  | Failure Error
  | Success Int
```

```
type WebData a
  = NotAsked
  | Loading
  | Failure Error
  | Success a
```

```
case webData of
  NotAsked ->
    text "Initialising... "

  Loading ->
    text "Loading..."

  Failure err ->
    text <| "Error: " ++ toString err

  Success data ->
    viewFunction data
```

Better defaults?

- Expose all APIs WebData
- Provide view helpers

```elm
default : (a -> Html) -> WebData a -> Html
default viewFunction webData =
  case webData of
    NotAsked ->
      text "Initialising... "

    Loading ->
      text "Loading..."

    Failure err ->
      text <| "Error: " ++ toString err

    Success data ->
      viewFunction data
```

```
-- MISSING PATTERNS ---------------------------------- ./src/View/WebData.elm

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

49|>    case webData of
50|>        Loading ->
51|>            text "Loading..."
52|>
53|>        Failure err ->
54|>            text <| "Error: " ++ toString err
55|>
56|>        Success data ->
57|>            viewFunction data

You need to account for the following values:

    RemoteData.NotAsked

Add a branch to cover this pattern!
```

Other UI issues?

```
navigation =
  { elements = [ "Home", "About", "Contact" ]
  , selected = "Home"
  }
```

```
navigation =
  { elements = [ "Home", "About", "Contact" ]
  , selected = "News"
  }
```

```
navigation =
   { elements = []
   , selected = "Home"
   }
```

Zipper

Select one of a list

```elm
nav =
  Zipper.fromList [ "Home", "About", "Contact" ]

current = Zipper.current nav

nav = Zipper.next nav
```

Smart constructors

Hiding impossibilities

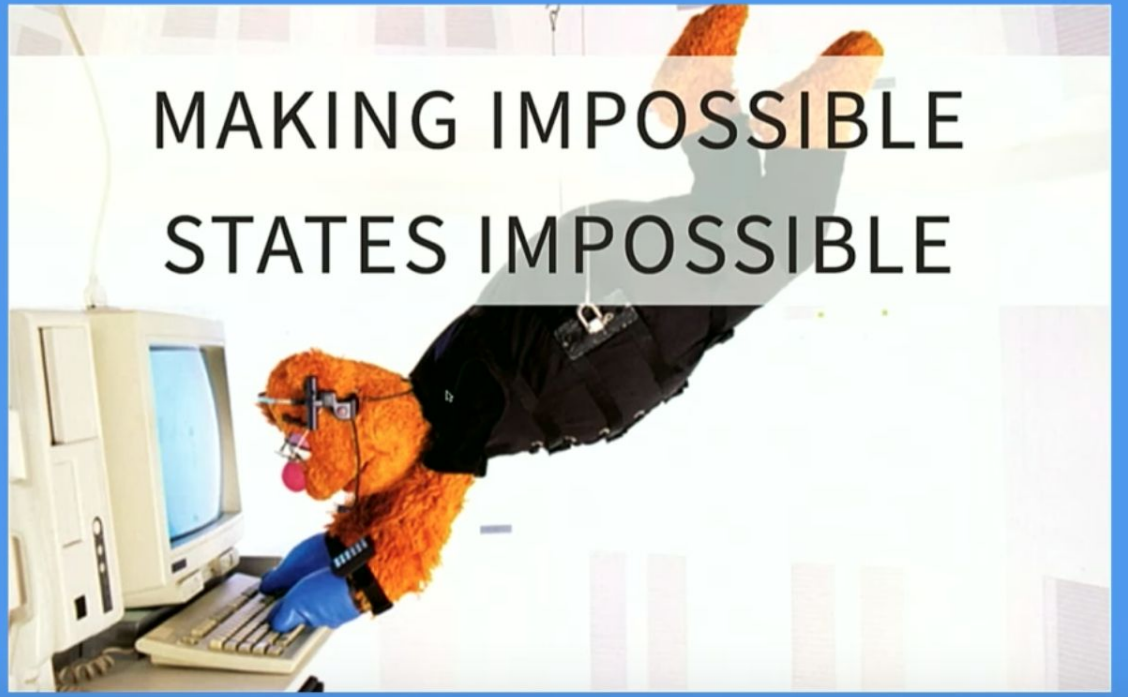
```
users = Success ["Haha!"]
```

Tests are good...

Impossible is better.

elm-conf

September 15, 2016
St. Louis, MO



0:12 / 25:05

Good UI/UX as a project default

- Expose all APIs as WebData
- Types to remove the impossible
- Smart cons to hide the impossible
- Provide easy UI/UX defaults

Thanks!

 @realmario

Come learn Elm!

Elm-London-Meetup

- Learn with other beginners
- Get help with your projects
- Pair with advanced folk