

Elm

Voyage en Terre Inconnue - Episode 2



zendesk



Functional Meetup Montpellier #2
23 Octobre 2017 - Tabmo

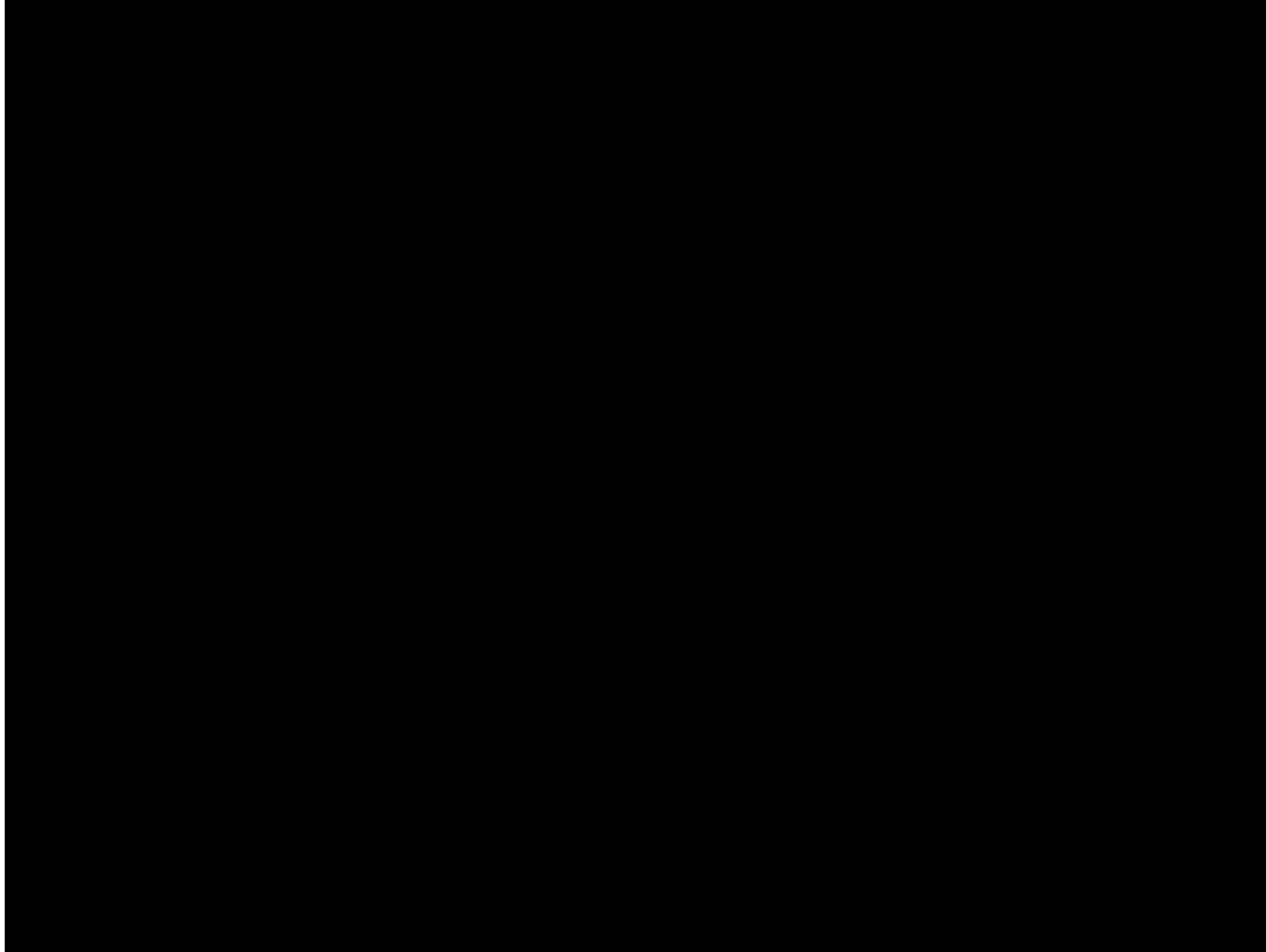
Romain Endelin

*A long time ago, in a **village** far,
far away...*



Conway's Game of life

Disclaimer: Le pire code Elm que vous verrez jamais



Mais alors, Elm c'est quoi?



- Créé en 2012
- BDFL: *Evan Czaplicki*
- Version actuelle: 0.18

- Utilisation: *Front-End Web*
 - Transpile en JavaScript

- Paradigmes:
 - *Fonctionnel*
 - *Reactive*

Et pourquoi c'est si bien?

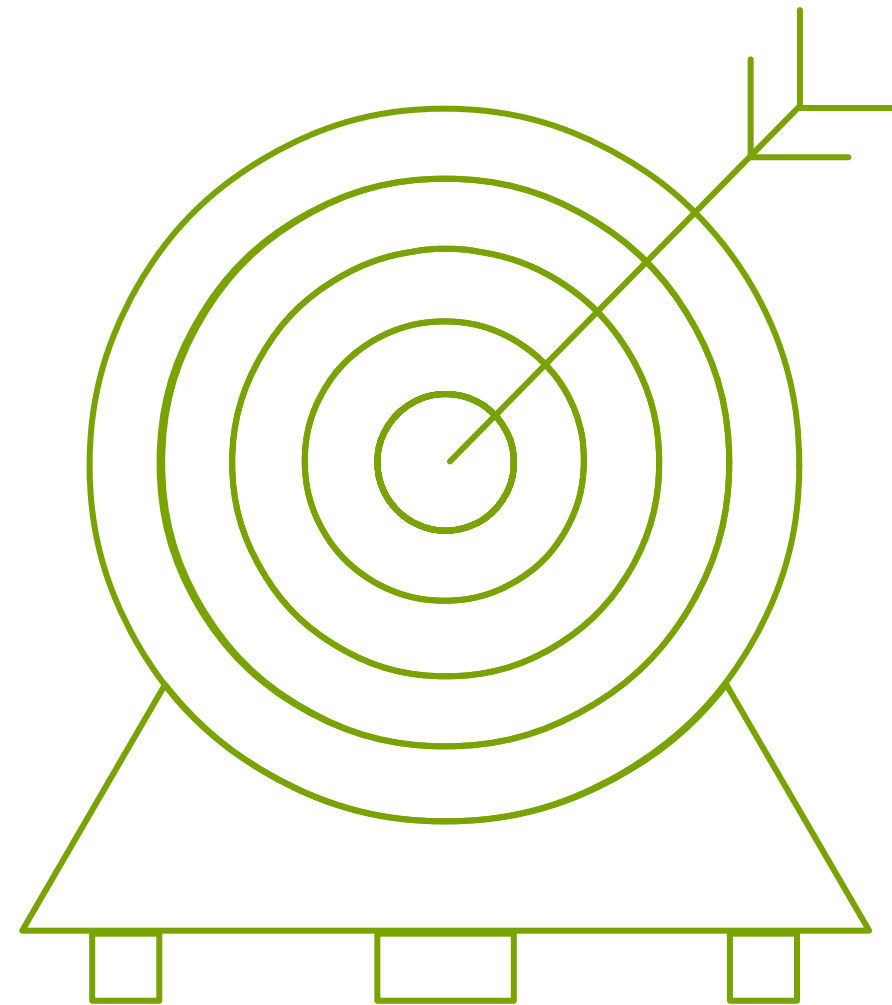
Elm, je t'aime!



Si ça compile, ça marche

Si ça compile, ça marche

- *"You can definitely mess with Elm if you want to, but the language makes it very hard for you to do so"*
- Pas de *Null Pointer Exceptions (NPE)*
- Typage fort
- On ne touche pas au DOM, compris?
- Pas de problèmes de concurrences
- Tout est immuable, TOUT!
- Pas de récursion infinie



Un exemple

```
type Operation -- Union Type
  = ToggleDone
  | SetName String -- Type with parameter

type alias Item = -- Type alias, storing records
  { id : Int
  , name : String
  , done : Bool
  }

modifyItem : Operation → Item → Item -- arg1 → arg2 → returnTypes
modifyItem operation item =
  case operation of -- Will not compile unless we handle all possible cases
    ToggleDone → -- Updating a record
      { item | done = (not item.done) }
    SetName name →
      { item | name = name }

create : Item
create =
  { id = 1, name = "my item", done = False }
  ▷ modifyItem ToggleDone
  ▷ modifyItem (SetName "my new name")
-- Returns { id =1, name = "my new name", done = True }
```



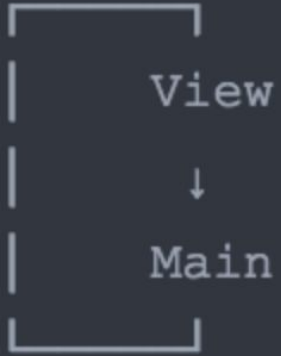
Si ça ne compile pas...

Le compilateur est votre meilleur ami

Si ça ne compile pas...

- *"Le compilateur de Elm, c'est comme avoir un programmeur expert derrière votre épaule, qui vous prévient gentiment chaque fois que vous perdez le cap"*
- Le compilateur est formidable 🐼

```
Your dependencies form a cycle:
```



```
graph TD; View --> Main; Main --> View;
```

The diagram illustrates a circular dependency between two modules, 'View' and 'Main'. 'View' is at the top and 'Main' is at the bottom. A vertical line on the left connects them, with a downward arrow pointing from 'View' to 'Main'. Horizontal lines extend from the left of 'View' and 'Main' to the vertical line, indicating that each module depends on the other, forming a cycle.

```
You may need to move some values to a new module to get rid of the cycle.
```

Refactorer du code Elm



Ricardo García Vega

@bigardone

Follow



933 additions & 942 deletions in 35 files, one single browser reload, and everything is working like a charm. Can't get enough of [#elmlang](#)

sections epic refactor [Browse files](#)

bigardone committed 4 minutes ago

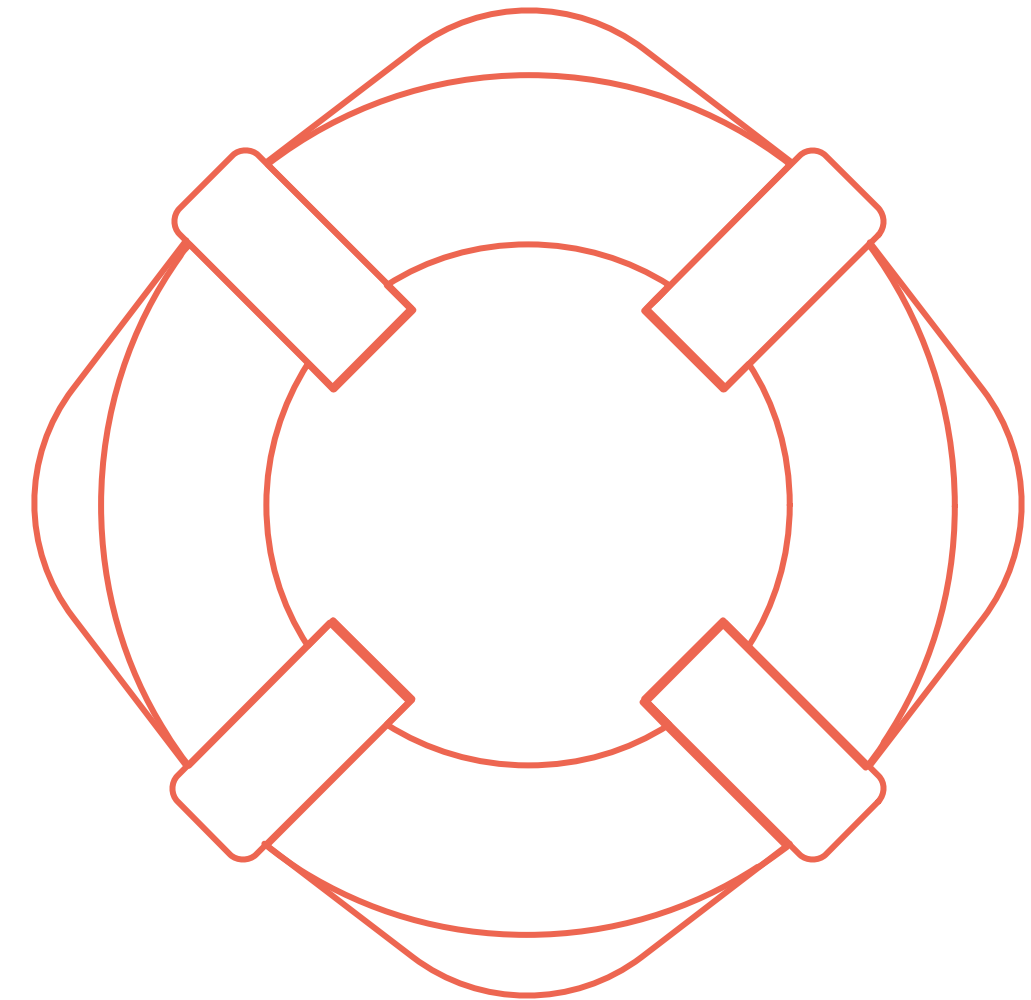
Showing **35 changed files** with 933 additions and 942 deletions. [No Whitespace](#) [Unified](#) [Split](#)

7:42 PM - 9 Sep 2017

Simple et efficace

Ouvert aux débutants

- On ne parle pas de monades!
- Beaucoup plus facile d'accès que Haskell
- Bonne documentation
- Le compilateur, *encore et toujours*
- Facile de comprendre ses erreurs, grâce aux types



Pas de tooling complexe

- elm-repl — play with Elm expressions
- elm-reactor — get a project going quickly
- elm-make — compile Elm code directly
- elm-package — download packages
- create-elm-app (<https://github.com/halfzebra/create-elm-app>) :
 - \$ *create-elm-app my_app*
 - \$ *elm-app start*
 - \$ *elm-app build*
 - \$ *elm-app eject*

Auto-formatter

```
main : Program Never Model Msg

main =
  | | | Html.program
    | | { init = init , view = view
    | |   , update = update
    | |   , subscriptions = subscriptions
    | | }

grid_width: Int
grid_width =
  | 100
grid_height : Int
grid_height = 60
frequency : Float
frequency = 100 * millisecond

cell_size : String
cell_size =
  | "5px"
```



```
main : Program Never Model Msg
main =
  | | | Html.program
    | | { init = init
    | |   , view = view
    | |   , update = update
    | |   , subscriptions = subscriptions
    | | }

grid_width : Int
grid_width =
  | 100

grid_height : Int
grid_height =
  | 60

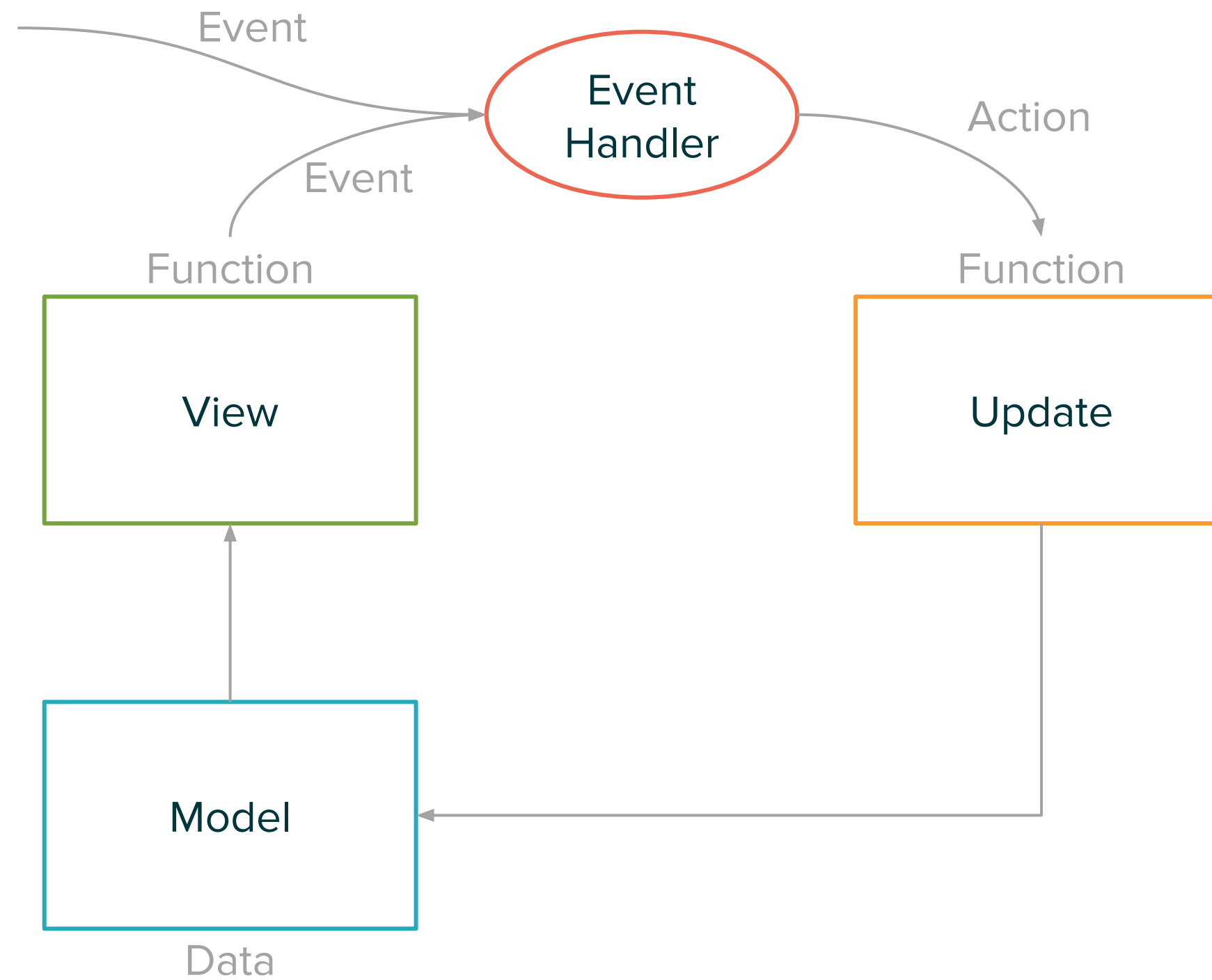
frequency : Float
frequency =
  | 100 * millisecond

cell_size : String
cell_size =
  | "5px"
```

Bonus: Upgrade automatique de Elm (v0.17 -> v0.18)

L'architecture de Elm

L'architecture de Elm



L'architecture - Côté code

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

```
main =
```

```
  Html.program
```

```
    { view = view
```

```
    , init = init
```

```
    , update = update
```

```
    , subscriptions = always Sub.none
```

```
    }
```

```
type Msg
```

```
  | SetDone Bool Item
```

```
update : Msg → Model → ( Model, Cmd Msg )
```

```
update msg model =
```

```
  case msg of
```

```
    SetDone isDone item →
```

```
      ( { model
```

```
        | items =
```

```
          (replaceIf ((=) item) { item | done = isDone } model.items)
```

```
        }
```

```
      , Cmd.none
```

```
    )
```

```
view : Model → Html Msg
```

```
view model =
```

```
  div [] (List.map model.items)
```

```
viewItem : Item → Html Msg
```

```
viewItem item =
```

```
  div
```

```
    [ onMouseEnter (Select item)
```

```
    , onClick (SetDone (not item.done) item)
```

```
    ]
```

```
    [ text item.name ]
```

```
type alias Model =
```

```
  { items : List Item
```

```
  }
```

```
init : ( Model, Cmd Msg )
```

```
init =
```

```
  ( { items =
```

```
    [ { id = 1, name = "Item1", done = False }
```

```
    , { id = 2, name = "Item2", done = False }
```

```
    ]
```

```
  }
```

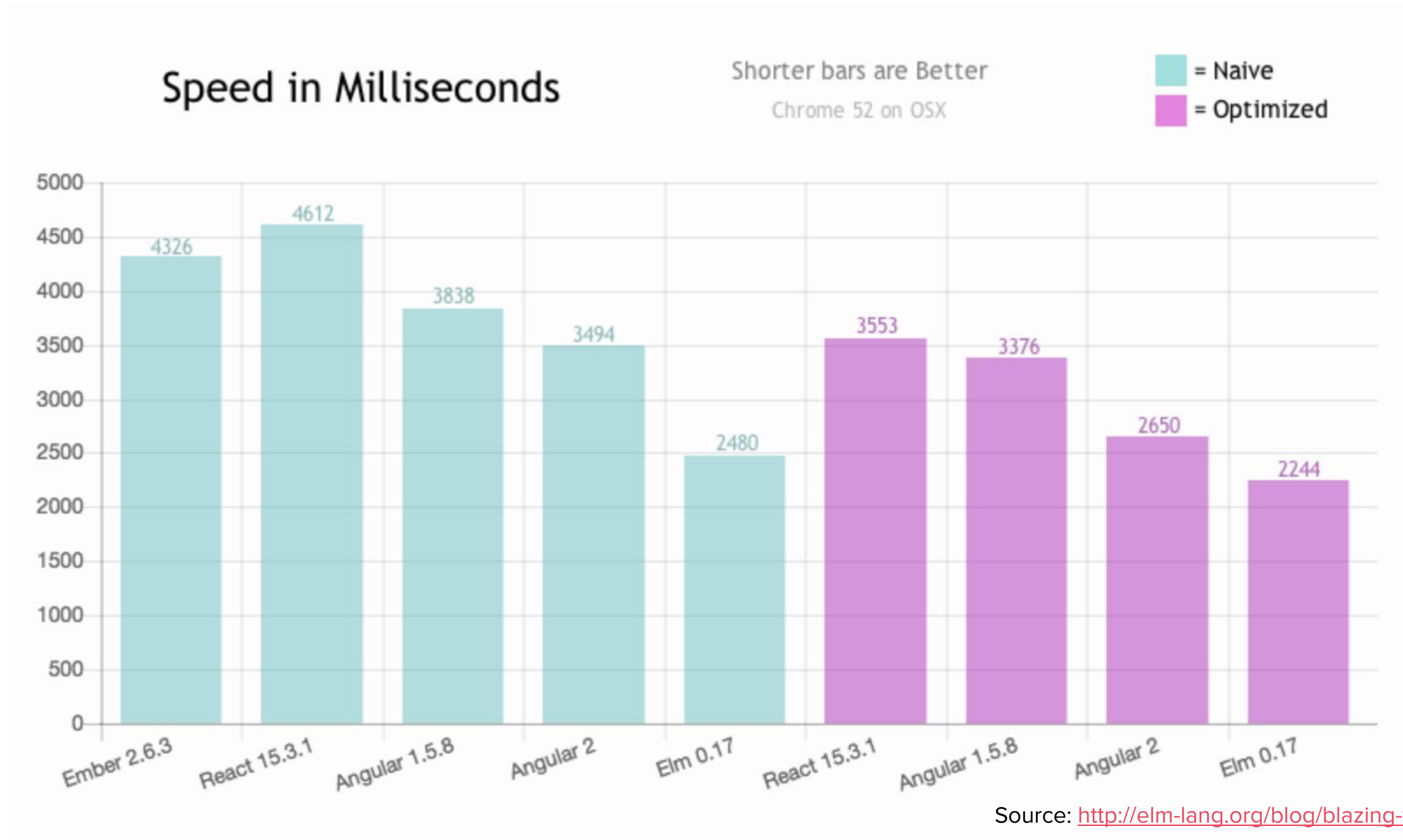
```
  , Cmd.none
```

```
  )
```


Les performances

Les performances

Ça va vite!



La communauté

La communauté

- Elm News (<http://elm-news.com/>)
- Reddit (<https://www.reddit.com/r/elm/>)
- Slack (<https://elmlang.herokuapp.com/> - <https://elmlang.slack.com>)

The image shows a presentation slide with a light green background. The title 'Let's be Mainstream!' is written in a large, dark brown, sans-serif font. Below the title, there is a blue horizontal bar with some faint, illegible text. To the right of the bar, the text 'User-focused Design in Elm' is written in a smaller, dark brown font. In the top right corner, the name 'Evan Czaplicki' is displayed above 'Prezi / @czaplic', with a small circular profile picture to the right. In the top right corner of the slide, there is a small inset video frame showing a man in a blue shirt standing on a stage and gesturing towards an audience. To the right of the slide, there is a dark blue vertical banner. At the top of the banner is a green circle containing a white double-parenthesis symbol '()'. Below this, the text 'CURRY ON' is written in white, uppercase letters inside a white rectangular box. Below the box, the word 'Prague' is written in a large, orange, cursive font. At the bottom of the banner, the text 'JULY 6-7TH, 2015' is written in a small, white, sans-serif font.

L'écosystème

Les restes du monde

Elm Package Manager

- Environ 750 packages
- Killer feature: Semantic Versioning

```
----- Changes to module Basics - MINOR -----  
  
Added:  
  type Never  
  
----- Changes to module Date - MINOR -----  
  
Added:  
  now : Task.Task x Date.Date  
  
----- Changes to module Debug - MAJOR -----  
  
Removed:  
  trace : String -> Graphics.Collage.Form -> Graphics.Collage.Form  
  watch : String -> a -> a  
  watchSummary : String -> (a -> b) -> a -> a  
  
----- Changes to module Dict - MINOR -----  
  
Added:  
  merge : (comparable -> a -> result -> result) -> (comparable -> a -> b -> result -> result) -> (comparable -> b -> result -> result) -> Dict.Dict comparable a -> Dict.Dict  
  comparable b -> result -> result  
  
----- Changes to module Random - MAJOR -----  
  
Added:  
  step : Random.Generator a -> Random.Seed -> (a, Random.Seed)  
  
Changed:  
  - generate : Random.Generator a -> Random.Seed -> (a, Random.Seed)  
  + generate : (a -> msg) -> Random.Generator a -> Platform.Cmd.Cmd msg  
  
----- Changes to module Task - MAJOR -----  
  
Added:  
  type alias Task err ok = Platform.Task err ok  
  perform : (x -> msg) -> (a -> msg) -> Task.Task x a -> Platform.Cmd.Cmd msg  
  
Removed:  
  type Task x a  
  type ThreadID  
  sleep : Task.Time -> Task.Task x ()  
  spawn : Task.Task x a -> Task.Task y Task.ThreadID
```



Interopérabilité avec Javascript

On place Javascript en quarantaine

js

elm

```
port module Spelling exposing (..)

...

-- port for sending strings out to JavaScript
port check : String -> Cmd msg

-- port for listening for suggestions from JavaScript
port suggestions : (List String -> msg) -> Sub msg

...
```

Pourquoi l'utiliser ?

Et quand ne pas l'utiliser ?

Pourquoi l'utiliser ?

- Pour le fun!
- Pour découvrir la programmation fonctionnelle
- Pour faire du Front-End sans se salir les mains sur Javascript
- Pour rejoindre une très bonne communauté
- Pour en finir avec les Null-Pointer Exceptions



Comment commencer ?

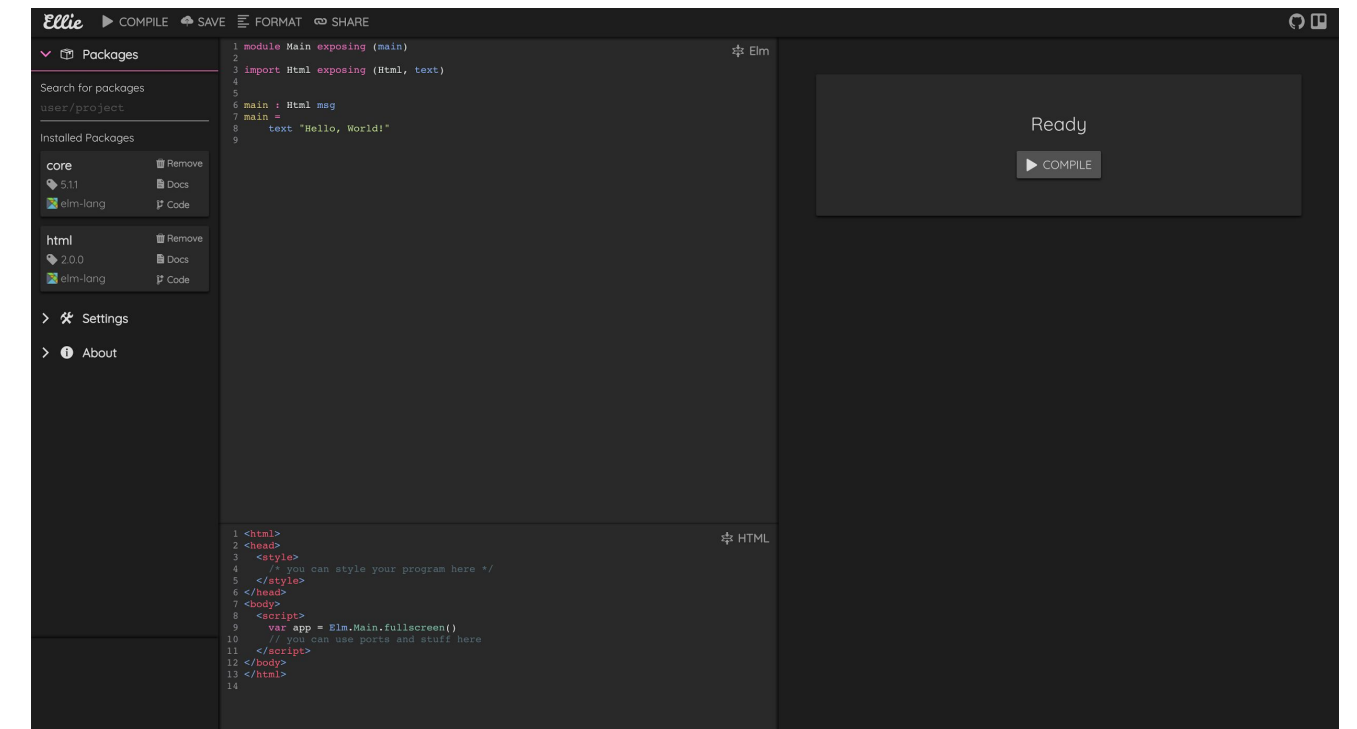
Essayez par vous-mêmes!

En entreprise:

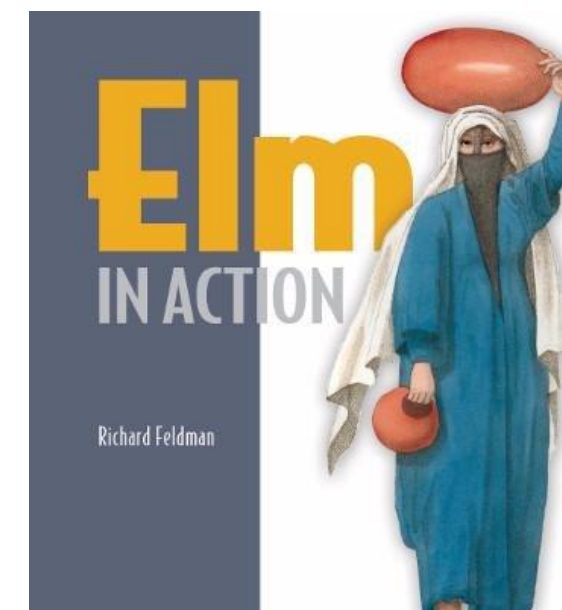
- Commencez par écrire un composant modeste en Elm.
- Pas de réécriture complète pour commencer!

Quelques ressources:

- La documentation officielle
- Elm SPA Example, par Richard Feldman (<https://github.com/rtfeldman/elm-spa-example>)
- Elm in Action, par Richard Feldman (early access)

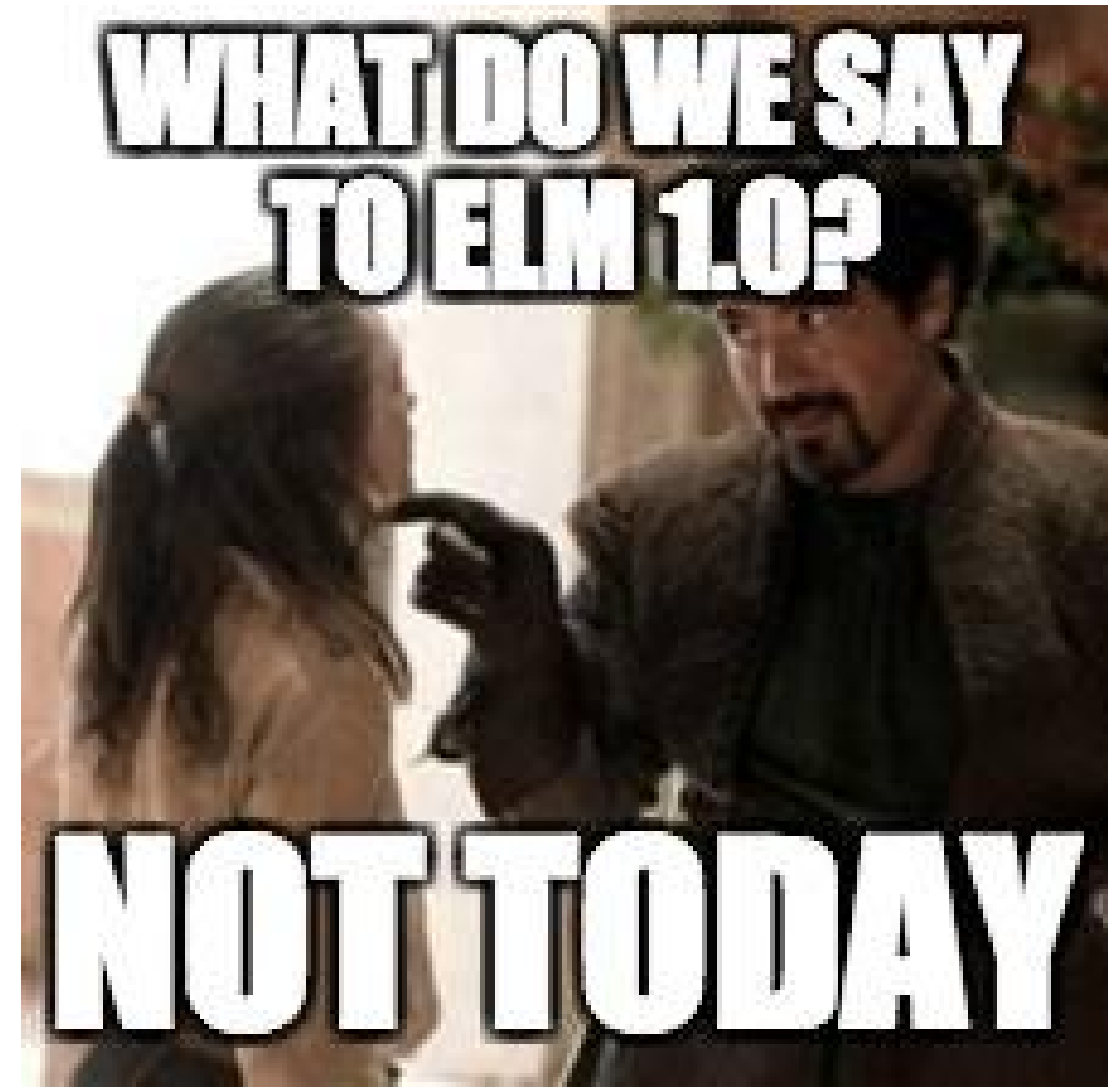


Ellie App, pour essayer sans rien installer



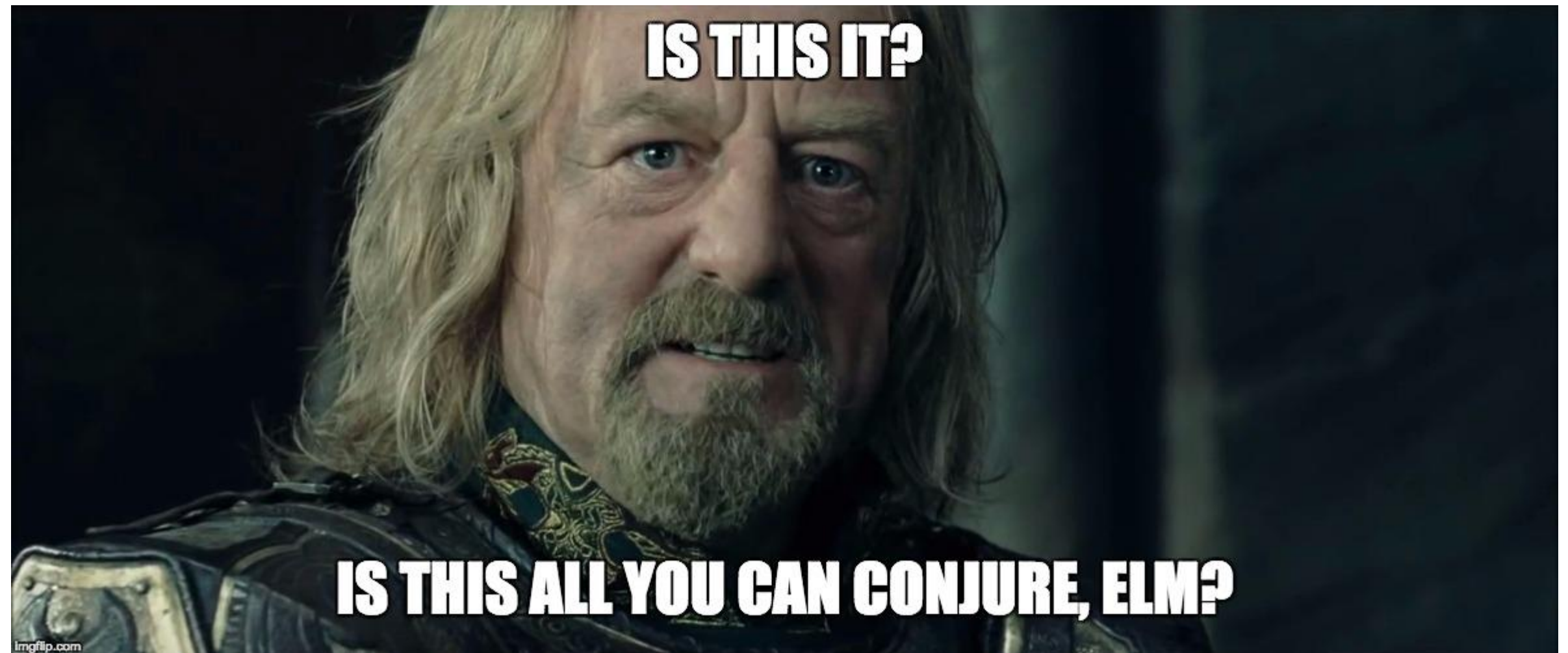
Quand ne pas l'utiliser?

- Si vous utilisez beaucoup de librairies Javascript
- Si vous souhaitez un langage stable
- Si vous êtes déjà un expert en Haskell
- Si vous souhaitez faire du développement mobile
- Si vous souhaitez utiliser Elm en Back-End



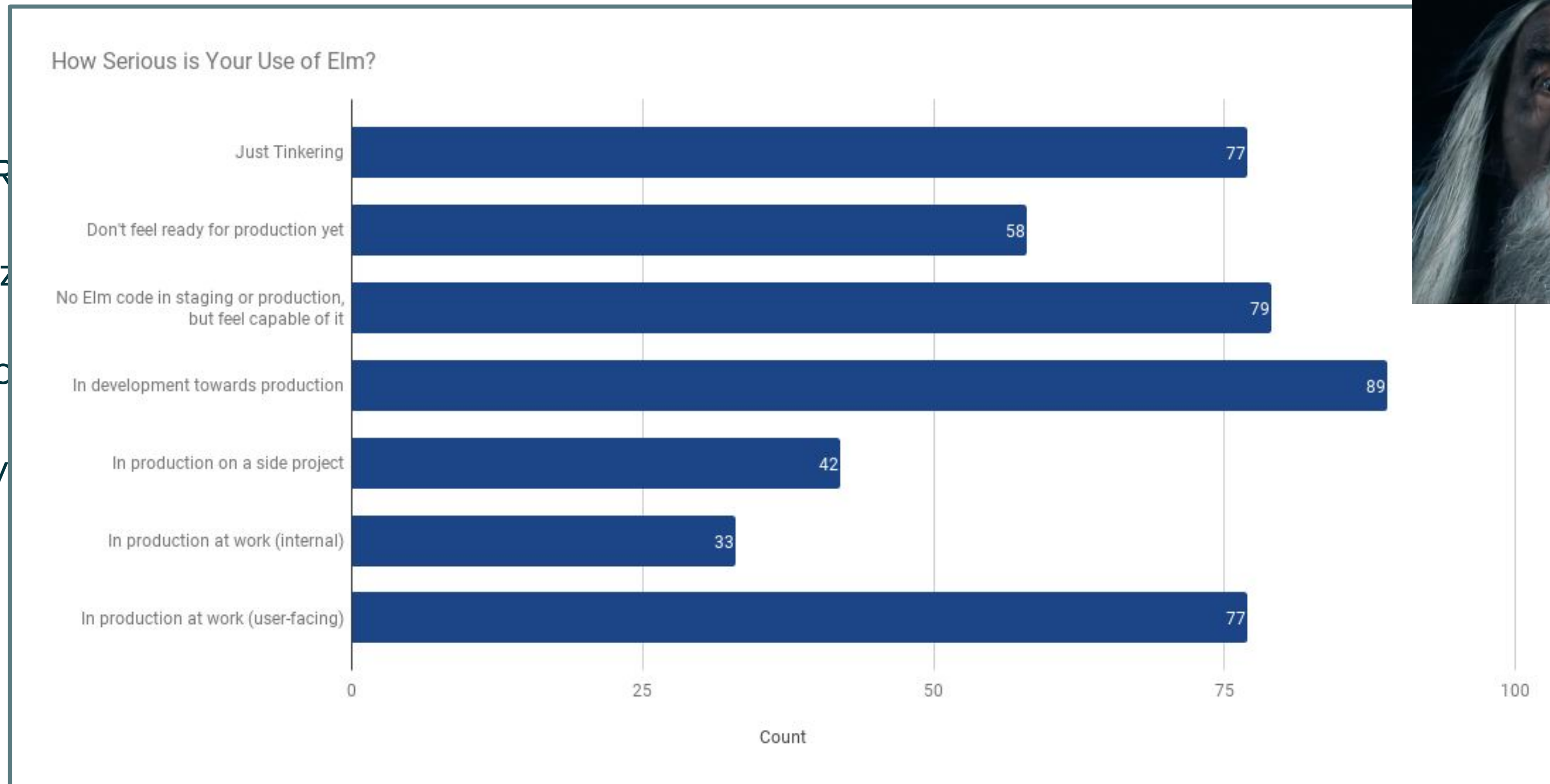
Quelques entreprises qui développent en Elm

- NoRedInk
- Prezi
- Pivotal Tracker
- Day One



Quelques entreprises qui développent en Elm

- NoR
- Prez
- Pivo
- Day



Source: <https://www.brianthicks.com/post/2017/07/27/state-of-elm-2017-results/>

Elm 0.19: Asset Management

- Séparation en plusieurs modules
- Navigation par URL / Routing
- Dead-Code Elimination
- Lazy Loading
- Server-Side Rendering
- Routing

 Data

 Page

 Request

 Views

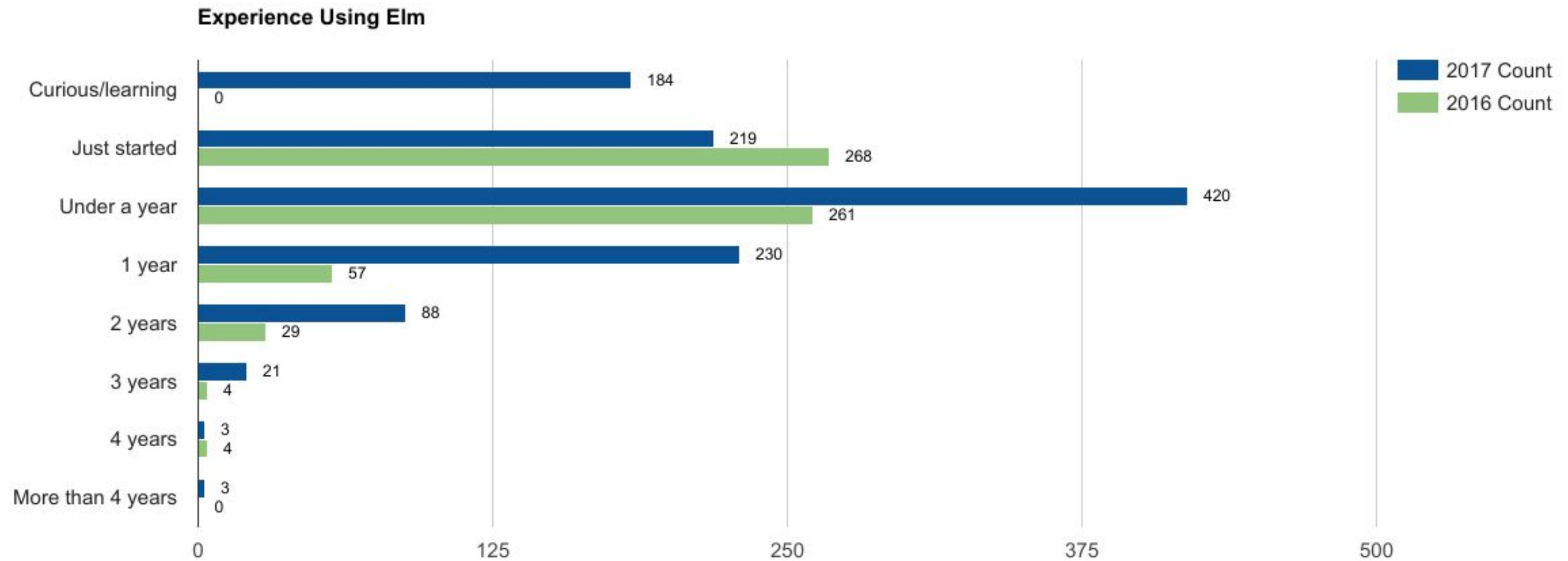
 Main.elm

 Ports.elm

 Route.elm

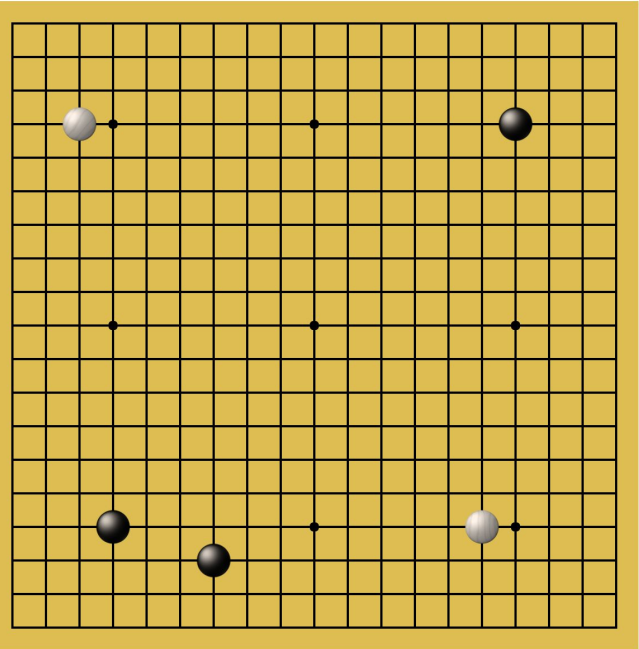
 Util.elm

La communauté



Source: <https://www.brianthicks.com/post/2017/07/27/state-of-elm-2017-results/>

Pour finir... Quelques exemples!



Elm Goban

black time: 2:52 white time: 2:47

black captures: 0 white captures: 0

Pause game

New 19x19 game

New 13x13 game

New 9x9 game

[Project source](#)

 **2048**

Join numbers to get the **2048** tile!
Written in Elm just for fun.

SCORE
1136

BEST
3144

New Game

128	8	4	
64	4		
16		2	

This is a clone of the **2048 Game** rewritten in Elm programming language by **Oleksiy Golovko**. Have fun playing!

Elm's game of life

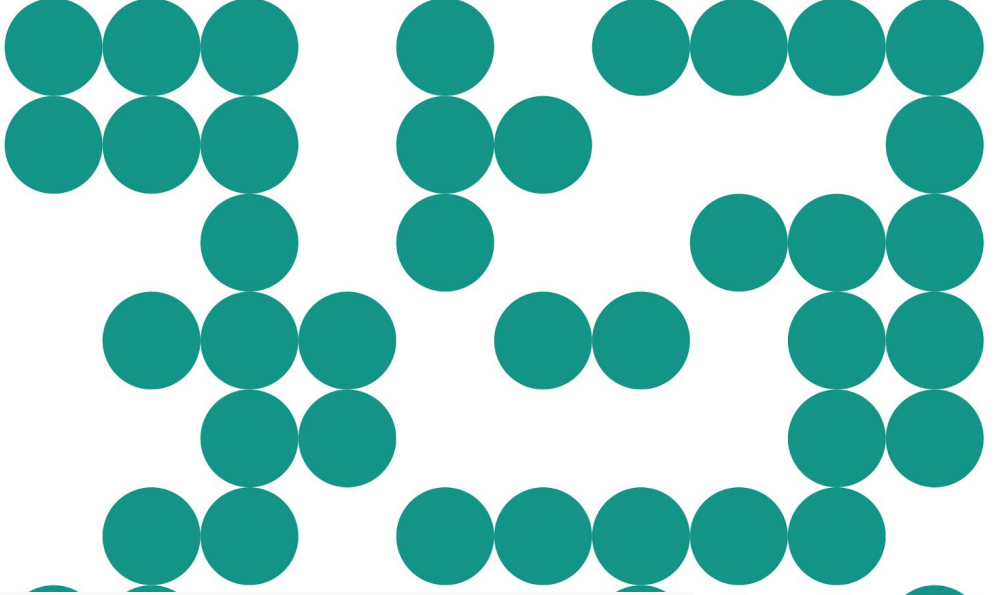
See on github Follow me on Twitter

Setup

setup your game and it will reset accordingly

select the speed: Normal

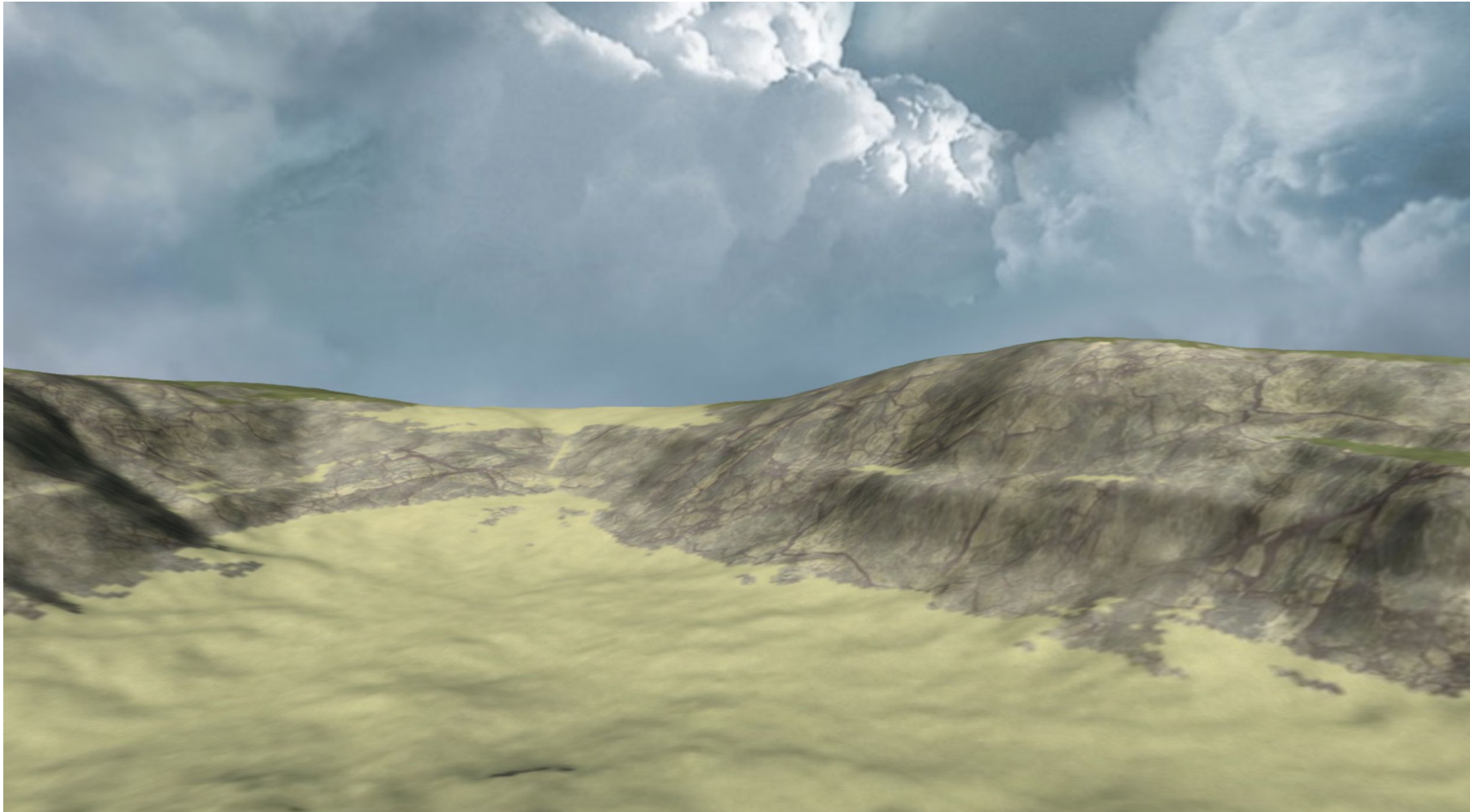
▶



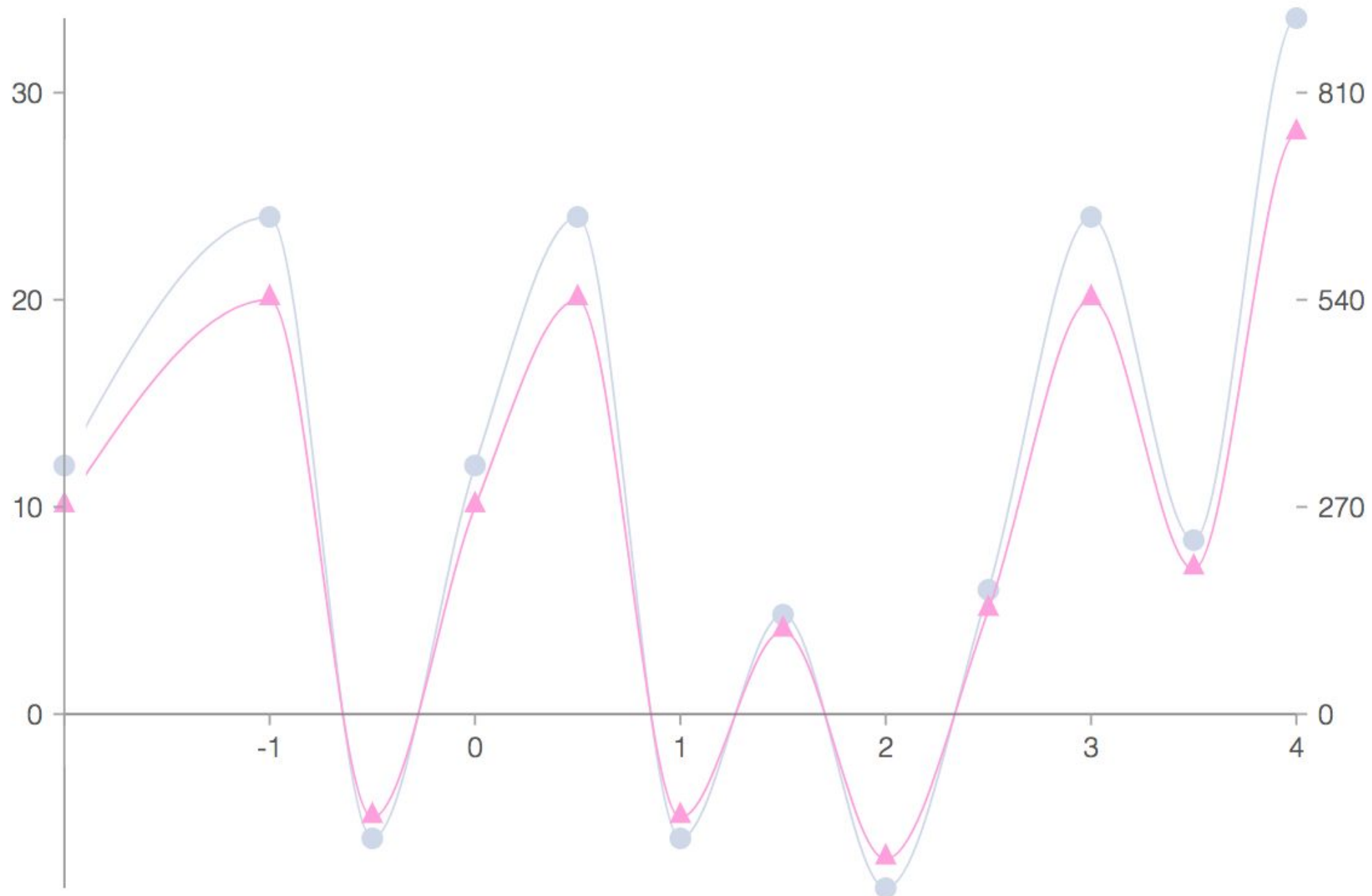
New Clear Fix Reset Undo Redo Edit Candidates Solution

							3	
5		2 3 4		6	7	9		
1	8			4	9			
					1			
6			8		3	2	9	
8	2							
7					8			3
2	6					7	8	
					4	5		

WebGL - Génération de Terrain



Dataviz - Elm Plot



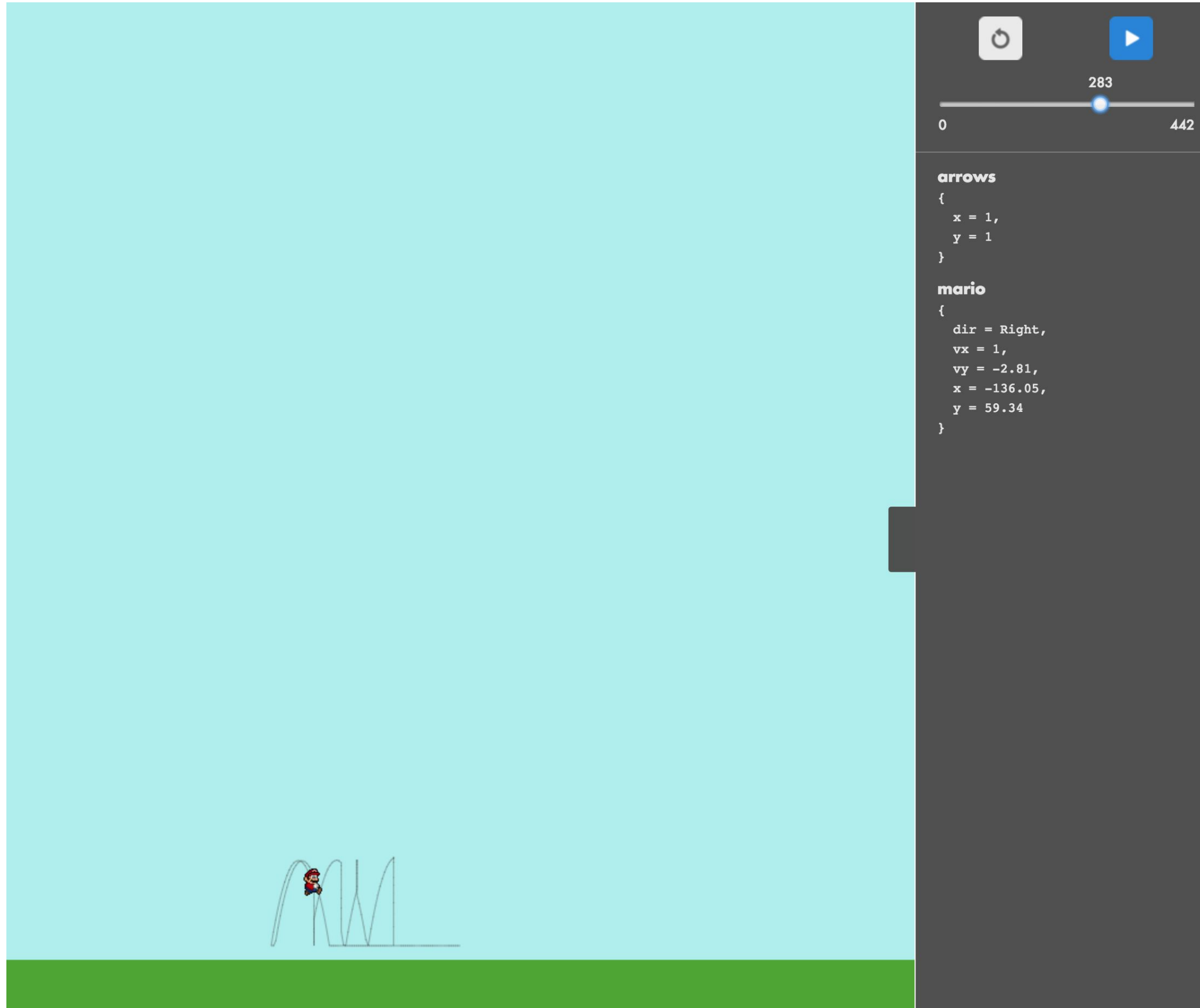
```
customArea : Series (List ( Float, Float )) msg
customArea =
  { axis = rightAxis
  , interpolation = Monotone Nothing [ stroke pinkStroke ]
  , toDataPoints = List.map \( x, y ) -> triangle x y
  }
```

```
customLine : Series (List ( Float, Float )) msg
customLine =
  { axis = axisAtMin
  , interpolation = Monotone Nothing [ stroke blueStroke ]
  , toDataPoints = List.map blueCircle
  }
```

```
blueCircle : ( Float, Float ) -> DataPoint msg
blueCircle ( x, y ) =
  dot (viewCircle 5 blueStroke) x (y * 1.2)
```

```
rightAxis : Axis
rightAxis =
  customAxis <| \summary ->
    { position = Basics.max
    , axisLine = Nothing
    }
```

Mario - Time-Travel Debugger

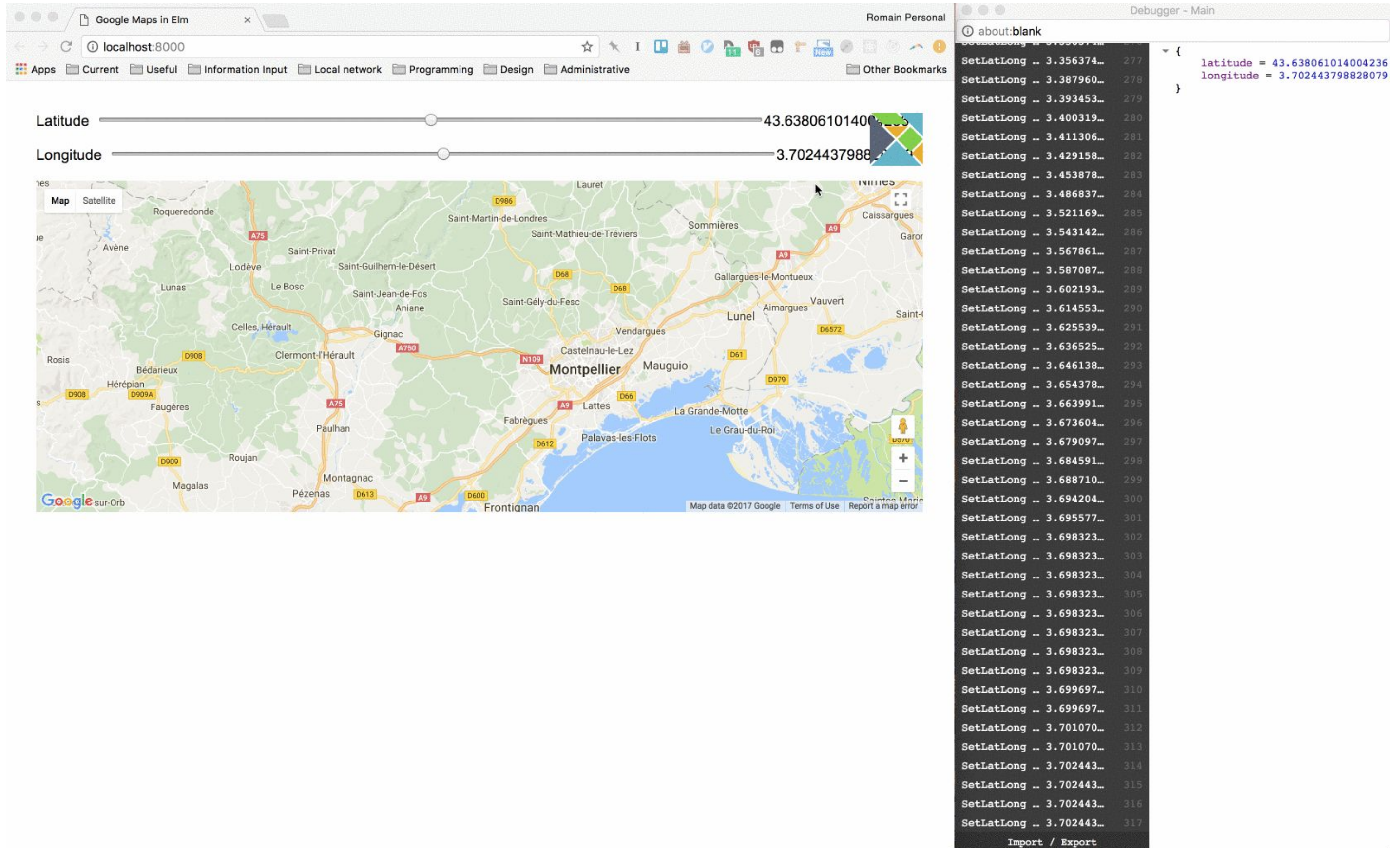


The screenshot displays the Mario Time-Travel Debugger interface. The main window shows a 2D game scene with a light blue sky and a green ground. A small Mario character is positioned on a series of grey, wave-like platforms. To the right of the game window is a dark grey sidebar containing a timeline and a code editor.

The sidebar features a timeline at the top with a play button icon, a slider bar, and numerical values 0, 283, and 442. Below the timeline, the sidebar is divided into two sections: **arrows** and **mario**.

```
arrows  
{  
  x = 1,  
  y = 1  
}  
  
mario  
{  
  dir = Right,  
  vx = 1,  
  vy = -2.81,  
  x = -136.05,  
  y = 59.34  
}
```

Google Map - Time-Travel debugger



The screenshot displays a web browser window with the address bar showing `localhost:8000`. The browser's bookmark bar includes categories like 'Apps', 'Current', 'Useful', 'Information Input', 'Local network', 'Programming', 'Design', and 'Administrative'. Below the browser window, a Google Map of Montpellier is visible. Overlaid on the map is a time-travel debugger interface. It features two horizontal sliders: 'Latitude' with a value of `43.638061014004236` and 'Longitude' with a value of `3.702443798828079`. To the right of the map, a list of `SetLatLng` calls is shown, each with a line number and a corresponding object containing `latitude` and `longitude` values. The list starts at line 277 and ends at line 317. The final values in the list match the values shown in the sliders.

```
SetLatLng ... 3.356374... 277 { latitude = 43.638061014004236, longitude = 3.702443798828079 }
SetLatLng ... 3.387960... 278
SetLatLng ... 3.393453... 279
SetLatLng ... 3.400319... 280
SetLatLng ... 3.411306... 281
SetLatLng ... 3.429158... 282
SetLatLng ... 3.453878... 283
SetLatLng ... 3.486837... 284
SetLatLng ... 3.521169... 285
SetLatLng ... 3.543142... 286
SetLatLng ... 3.567861... 287
SetLatLng ... 3.587087... 288
SetLatLng ... 3.602193... 289
SetLatLng ... 3.614553... 290
SetLatLng ... 3.625539... 291
SetLatLng ... 3.636525... 292
SetLatLng ... 3.646138... 293
SetLatLng ... 3.654378... 294
SetLatLng ... 3.663991... 295
SetLatLng ... 3.673604... 296
SetLatLng ... 3.679097... 297
SetLatLng ... 3.684591... 298
SetLatLng ... 3.688710... 299
SetLatLng ... 3.694204... 300
SetLatLng ... 3.695577... 301
SetLatLng ... 3.698323... 302
SetLatLng ... 3.698323... 303
SetLatLng ... 3.698323... 304
SetLatLng ... 3.698323... 305
SetLatLng ... 3.698323... 306
SetLatLng ... 3.698323... 307
SetLatLng ... 3.698323... 308
SetLatLng ... 3.698323... 309
SetLatLng ... 3.699697... 310
SetLatLng ... 3.699697... 311
SetLatLng ... 3.701070... 312
SetLatLng ... 3.701070... 313
SetLatLng ... 3.702443... 314
SetLatLng ... 3.702443... 315
SetLatLng ... 3.702443... 316
SetLatLng ... 3.702443... 317
```

Import / Export

Conclusion

Elm, c'est bien!

Questions?

