A Simple Odissey

Riccardo Cambiassi - Clojure eXchange 2016

Riccardo Cambiassi
Co-Founder, 100Starlings

@bru on Twitter

on a typical day…

- Ruby Development
- Devops
- Team Coordination & Mentoring

What is this about?

- A way to build cross platform desktop apps
- deeply integrated with the native platform
- using Clojurescript and friends

Why?

Why hybrid?

- Web is what I do
- Convenience over Performance
- Cross platform

Why Clojurescript?

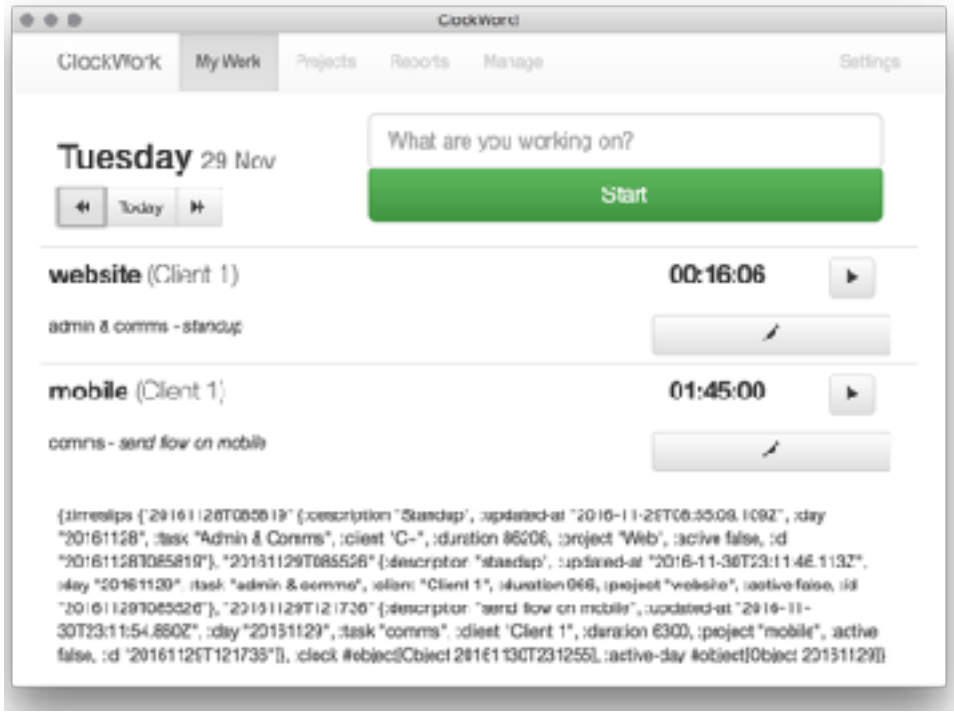
- A good excuse to practice
- elegant (core.)async programming
- you know, it's clojure…

Fan Beat / Aerials

Interactive installation, WebGL, data pipeline, auto-update, Windows environment

ClockWork

Desktop time tracker, FRP, macOS



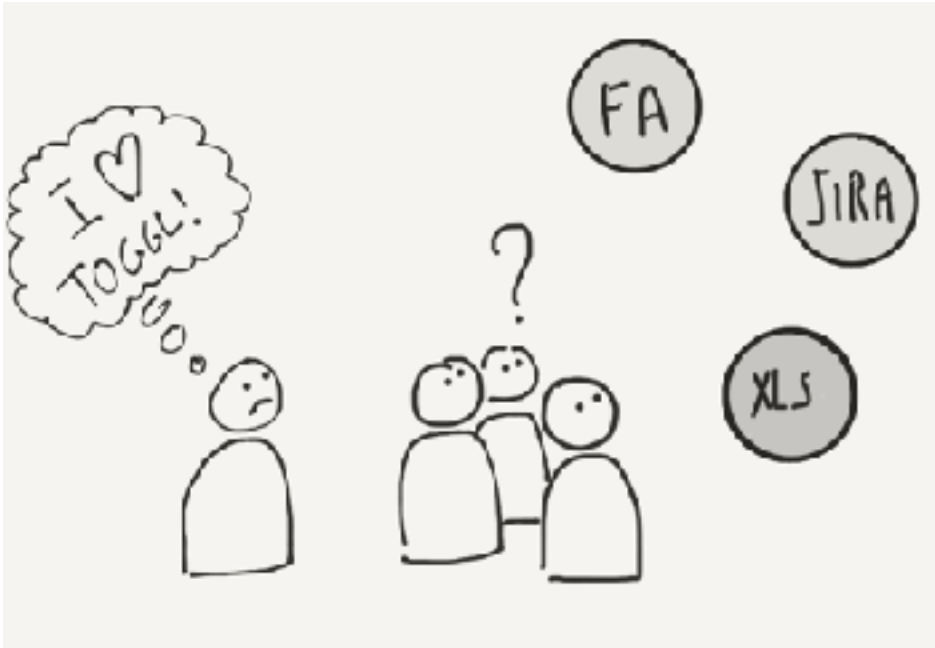
“Please read my diary, look through my things and figure me out.”

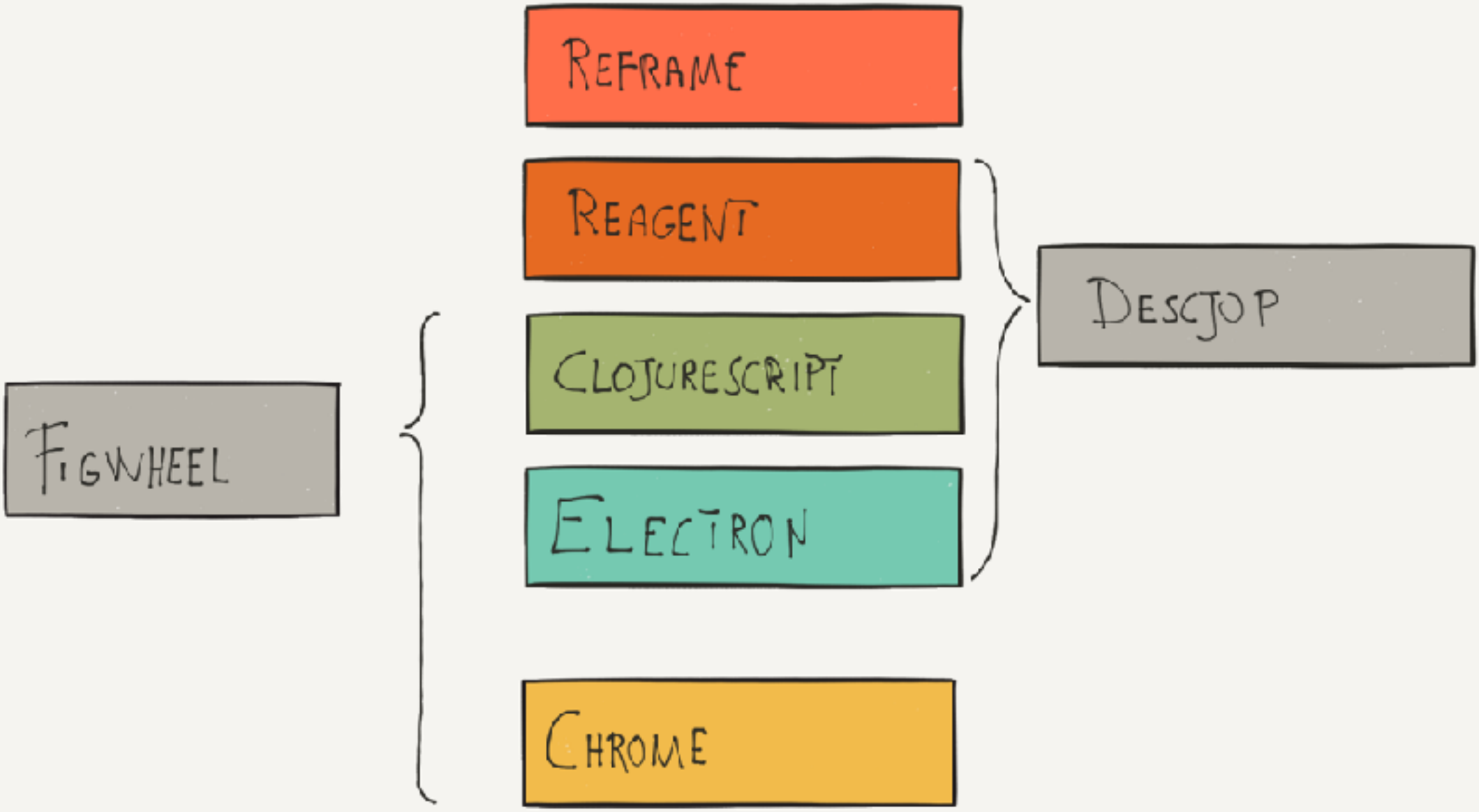
– Kurt Cobain

Project Spotlight: ClockWork

A boring Time Tracker…

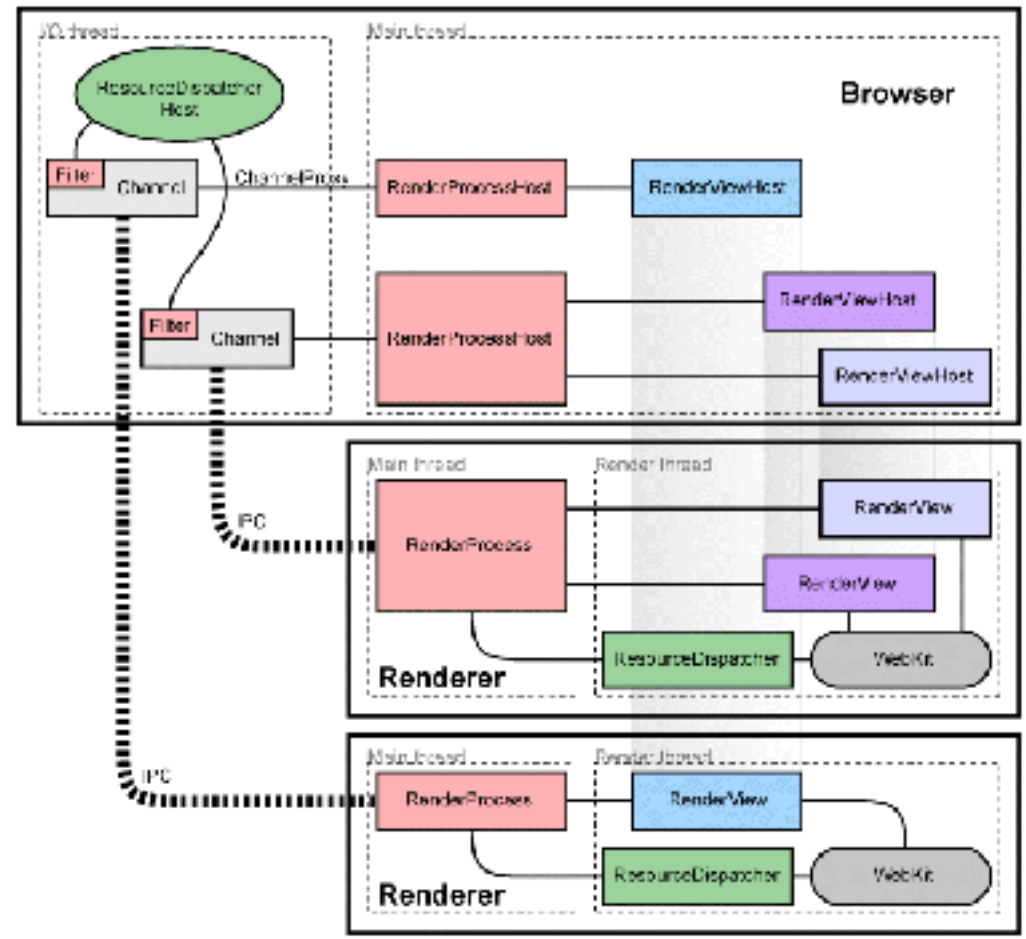
- offline first
- easy access to data
- integration with existing apps




REFRAME
REAGENT
CLOJURESCRIPT
ELECTRON
CHROME
DESCJOP
FIGWHEEL

Electron

- Chromium views (Renderers) orchestrated by a Node process (Main)
- Full access to the system
- WebView (Chrome) support
- built in IPC
- Active community with big players (Slack, Atom)
- Bonus: tray, status bar, notifications, auto-update



Electron

```
(defn -main []
  (.start crash-reporter
    (clj->js {:companyName "100Starlings"
              :productName "ClockWork"
              :submitURL   "http://apps.100starlings.com/crash"}))

  (.on app "ready"
       (fn []
         (reset! *win* (BrowserWindow.
                         (clj->js {:width 800 :height 600
                                   :minWidth 400 :minHeight 400 })))

         (.loadURL @*win* index_html_file_path)
         (.on @*win* "closed" (fn [] (reset! *win* nil))))))
```

Electron

```
(def Electron (nodejs/require "electron"))
(def remote (.-remote Electron))
(def app (.-app remote))
(def filepath (nodejs/require "path"))

(defn timeslips-file [day]
  "Build the filename for this day"
  (let [data-path (.getPath app "userData")
        year (time/year day)
        week (time/week-number-of-year day)
        filename (str year "-" week "-timeslips.edn")]
    (.resolve filepath data-path filename)))
```

Clojurescript

A few personal notes:

- I get more attached to cljs code
- No “autopilot” (yet)
- Less code poetry (or the Perl kind)

Reagent

```
(defn debug-db []
  (fn []
    [:div.debug-db
      (str @db)]))

(defn today []
  [:div.container
   [head-row]
   [timeslips-table]
   [debug-db]])
```

Reagent

Pros:

- Good level of abstraction over React
- There's a lot more underneath though (explore!)

Cons (*none really*, but notable mentions to):

- Multiple ways of defining a component
- "Too much freedom"

Discovery

or, the walk of shame, part I

TL;DR Watch out!

The idea was, get the basics working, or fail fast, but…

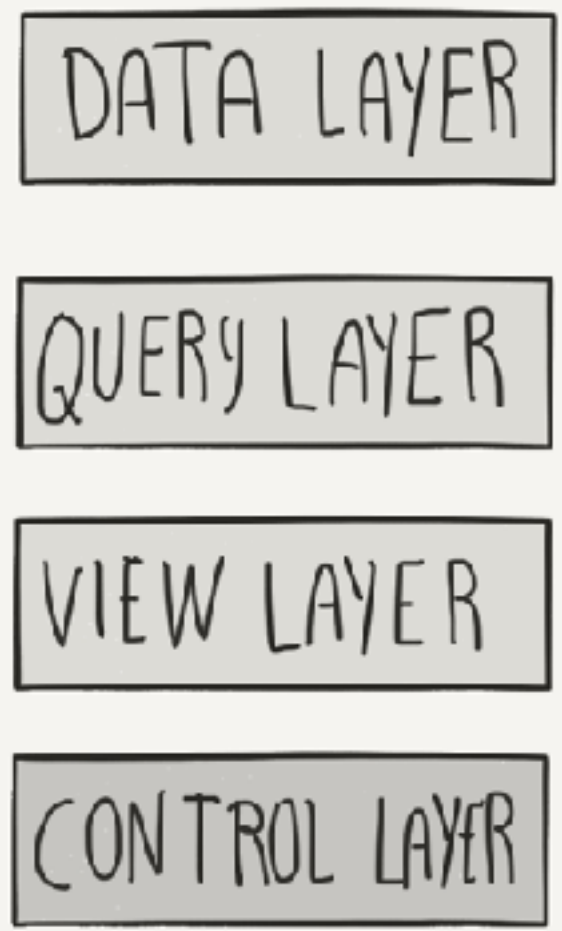
The Good Parts:

- Dev Tools
- App State debugging dump

Re-Frame

A pattern for writing SPAs in Clojurescript, using Reagent

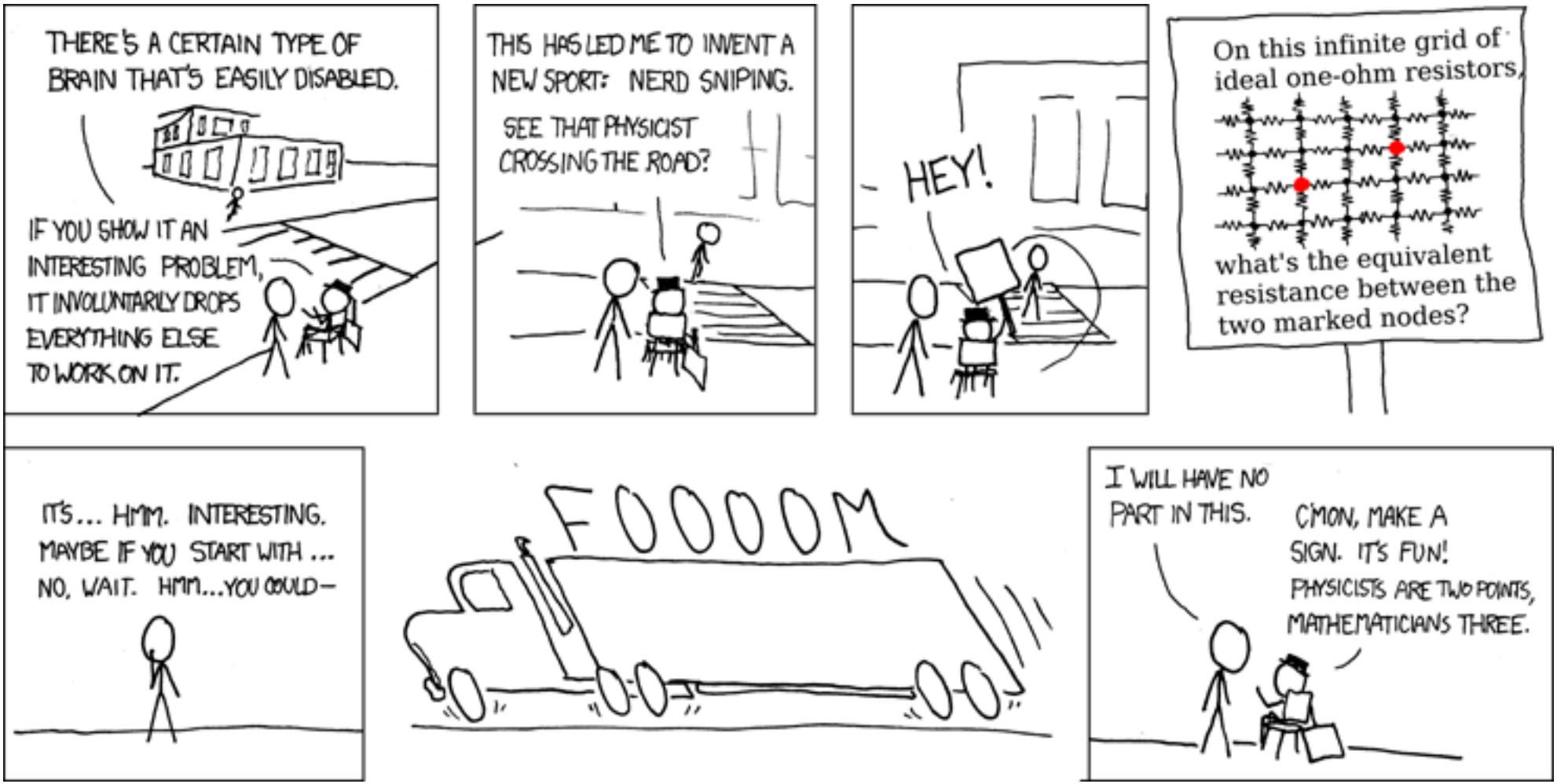
- You write Pure Functions
- The “ugly bits” remain hidden
- Unidirectional Signal Graph
- Small & Simple



Modelling the data

or, of living with the consequences or, the walk of shame part II

- Mutable Timers
- Focus the view on 1 day at a time
- Can run multiple timers at once
- Can start timer in the past


THERE'S A CERTAIN TYPE OF BRAIN THAT'S EASILY DISABLED.
IF YOU SHOW IT AN INTERESTING PROBLEM, IT INVOLUNTARILY DROPS EVERYTHING ELSE TO WORK ON IT.
THIS HAS LED ME TO INVENT A NEW SPORT: NERD SNIPING.
SEE THAT PHYSICIST CROSSING THE ROAD?
HEY!
On this infinite grid of ideal one-ohm resistors,
what's the equivalent resistance between the two marked nodes?
IT'S... HMM. INTERESTING. MAYBE IF YOU START WITH ... NO, WAIT. HMM...YOU COULD—
FOOOOM
I WILL HAVE NO PART IN THIS.
C'MON, MAKE A SIGN. IT'S FUN! PHYSICISTS ARE TWO POINTS, MATHEMATICIANS THREE.

View Layer

```
(defn timeslips-table []
  (let [timeslips (subscribe [:active-day-timeslips])]
    (fn []
      (let [label-task ""
            label-time ""]
        [:div.row
         [:table.table.table-hover
          [:thead
           [:tr
            [:th.col-sm-8 label-task]
            [:th.col-sm-4 label-time]]]
          [:tbody
           (for [timeslip @timeslips]
             ^{:key (str "timeslip-row-" (:id timeslip))}
             [timeslip-row timeslip]
             )]]])))))
```

UI: The Other Side

Not exactly your average website:

- multiple tabs
- tray icon badge
- status bar

Event handling to the rescue!

Query Layer

```
(reg-sub
 :active-day
 (fn [db _]
   (:active-day db)))

(reg-sub
 :timeslips
 (fn [db _]
   (:timeslips db)))

(reg-sub
 :active-day-timeslips
 (fn [query-v _]
   [(subscribe [:timeslips])
    (subscribe [:active-day])])

 (fn [[timeslips active-day] _]
   (let [timeslips (if (nil? timeslips) [] (vals timeslips))
         day (str active-day)]
     (filter #(= day (:day %))
             timeslips))))
```

Event Handling

```
(reg-cofx
 :current-day
 (fn [cofx _]
   "Get active date"
   (assoc cofx :current-day (time/today))))

(reg-event-fx
 :goto-today
 [(inject-cofx :current-day)]
 (fn [{:keys [db current-day]} _]
   {:db (assoc db :active-day current-day)
    :dispatch [:load-timeslips]}))
```

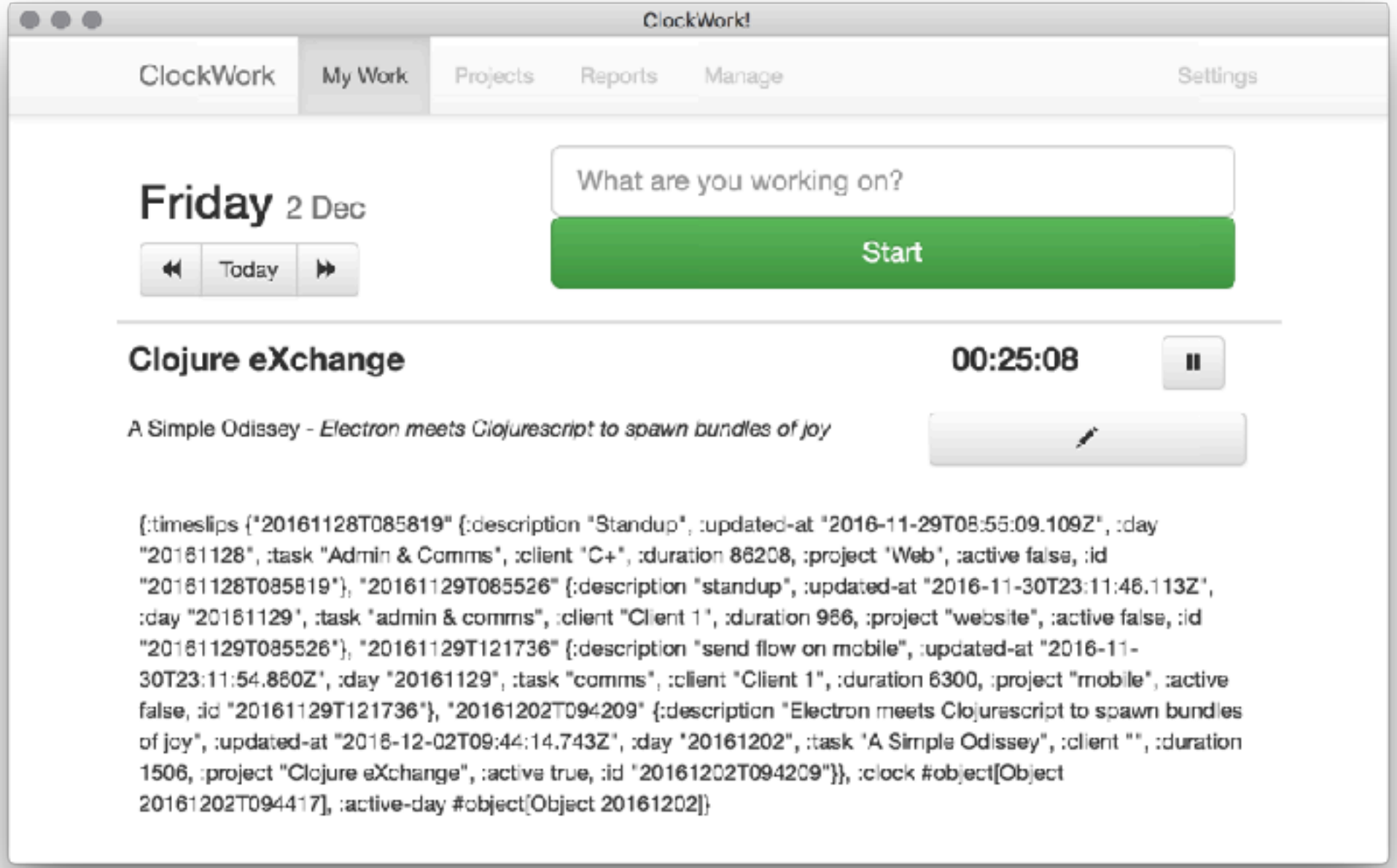
Event Handling (WAT?!)

```
(defonce timeslips-saver
  (js/setInterval #(dispatch [:save-timeslips]) 10000))

(reg-event-fx
 :save-timeslips
 (fn [{:keys [db]} _]
   (let [timeslips (:timeslips db)
         active-day (:active-day db)]
     {:timeslips->file [timeslips active-day]})))

(reg-fx
 :timeslips->file
 (fn [[ timeslips active-day ]]
   (save-timeslips timeslips active-day)))

(defn save-timeslips [timeslips day]
  (let [file (timeslips-file day)]
    (try
      (.writeFileSync fs file timeslips)
      (catch js/Error e nil))))
```


ClockWork!
ClockWork
My Work
Projects
Reports
Manage
Settings
Friday 2 Dec
Today
What are you working on?
Start
Clojure eXchange
00:25:08
A Simple Odissey - Electron meets Clojurescript to spawn bundles of joy
{:timeslips {"20161128T085819" {:description "Standup", :updated-at "2016-11-29T08:55:09.109Z", :day "20161128", :task "Admin & Comms", :client "C+", :duration 86208, :project "Web", :active false, :id "20161128T085819"}, "20161129T085526" {:description "standup", :updated-at "2016-11-30T23:11:46.113Z", :day "20161129", :task "admin & comms", :client "Client 1", :duration 966, :project "website", :active false, :id "20161129T085526"}, "20161129T121736" {:description "send flow on mobile", :updated-at "2016-11-30T23:11:54.860Z", :day "20161129", :task "comms", :client "Client 1", :duration 6300, :project "mobile", :active false, :id "20161129T121736"}, "20161202T094209" {:description "Electron meets Clojurescript to spawn bundles of joy", :updated-at "2016-12-02T09:44:14.743Z", :day "20161202", :task "A Simple Odissey", :client "", :duration 1506, :project "Clojure eXchange", :active true, :id "20161202T094209"}}, :clock #object[Object 20161202T094417], :active-day #object[Object 20161202]}

What Next?

- package (e.g. application icon)
- navigation (e.g. menu, keyboard shortcuts)
- content (e.g. help, about)
- security (e.g. binary signature)
- maintenance (e.g. auto update)

References

- http://descjop.org
- https://github.com/bhauman/lein-figwheel
- http://electron.atom.io
- http://reagent-project.github.io
- https://github.com/binaryage/cljs-devtools
- https://github.com/Day8/re-frame
- https://xkcd.com/356/

Thank You!

Riccardo Cambiassi

riccardo.cambiassi@100starlings.com

https://twitter.com/bru

https://github.com/bru/clockwork-electron