# If we are going to transpile JavaScript, why not use ClojureScript?

Shivek Khurana

JUXT

Krim Labs

# About me

- Apps since 2008, Clojure since 2017
- Consultant at Juxt Ltd.
- Enterprise Apps
- Currently building a system for Vodafone

JUXT

Krim Labs

# Built an LMS from 2012 to 2016

Long projects == Living with your past mistakes

JUXT

Krim Labs

# Built an LMS from 2012 to 2016

Long projects == Living with your past mistakes


JUXT
Krim Labs

# JS Community Conundrum

$$LI + EC = HVOD + C$$

JUXT

Krim Labs

# JS Community Conundrum

Enthusiatic Community

Abandoned Libs & Ideas

$$LI + EC = HVOD + C$$

Language Imperfections

High Velocity Open Source Development

JUXT

Krim Labs

# Clojure/Script

JUXT

Krim Labs

*All the best games are easy to learn and difficult to master.*

— *Nolan Bushnell (Founder, Atari)*

JUXT

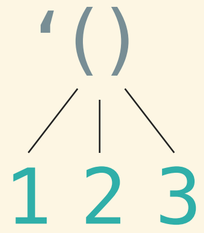
Krim Labs

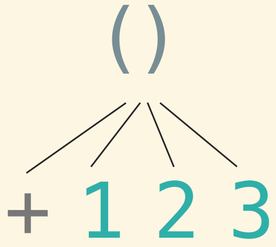
# ;; LISP – LISt Processing

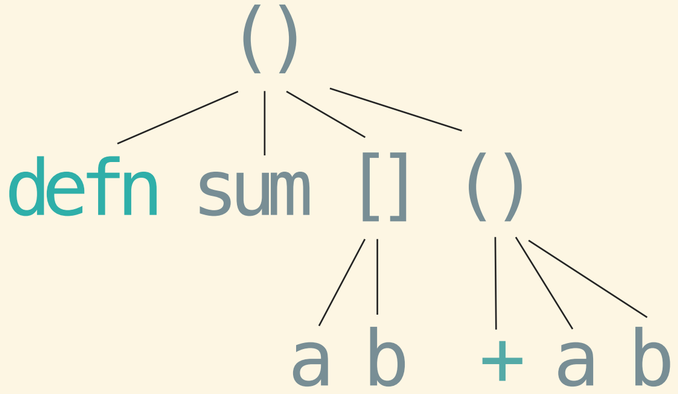
```
‘(1 2 3)

(+ 1 2 3)

(defn sum [a b]
  (+ a b))
```

# ;; LISP – LISt Processing

```
‘(1 2 3)

(+ 1 2 3)

(defn sum [a b]
  (+ a b))
```




JUXT

Krim Labs

# ;; Functional

```clojure
(def person
  {:name "Vienaa"
   :age 24})

(get person :name)
(get person :age)
```

```javascript
class Person {
  init(name, age) {…}
  getName() => {…}
  getAge() => {…}
}

const v = Person("Viena", 24)
v.getName()
v.getAge()
```

# ;; Atoms
## ;; thread safe/ mutable constructs

```
(defonce count (atom 0))
(swap! count inc) ;; => 1
count ;; => 1
(reset! count 4) ;; => 4
count ;; => 4
```

# ;; Homoiconic
# ;; Code is data

```
‘(1 2 4)

[:a :b]

(defn sum [a b]
  (+ a b))
```

```
{name: “Shivek Khurana”}

[“apples”, “mangoes”, 8]

function sum(a, b) {
  return a+b;
}
```

JUXT

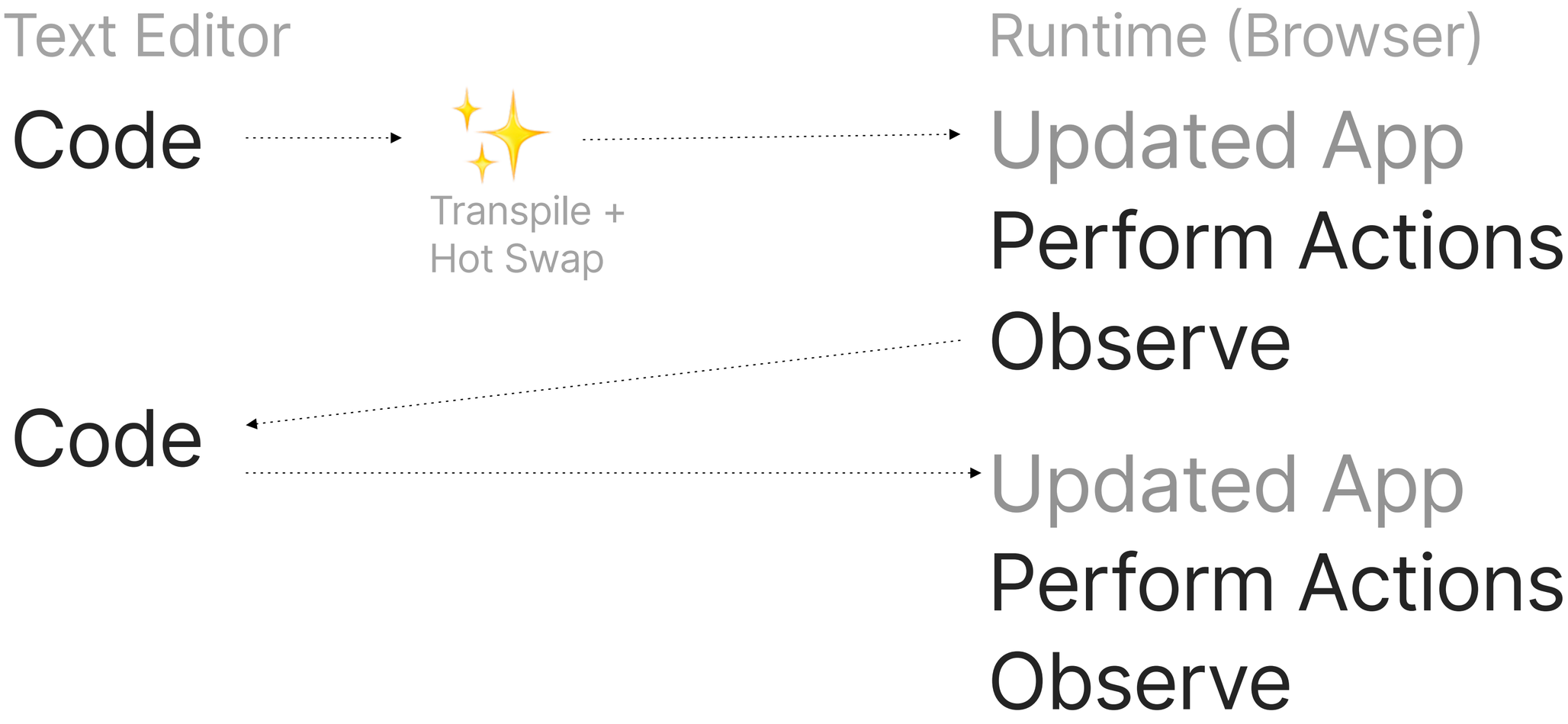
Krim Labs

# REPL - Developing inside the Runtime

JUXT

Krim Labs

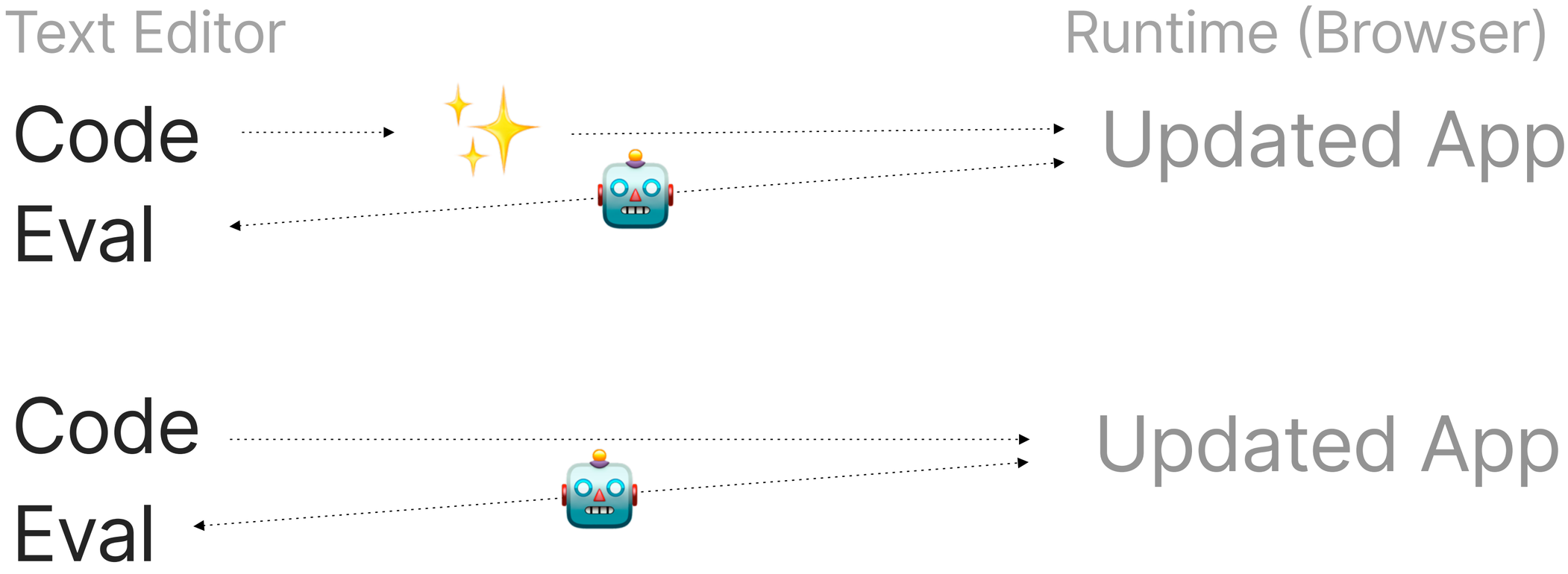
# Observation Driven Dev




JUXT

Krim Labs

# REPL Driven Dev



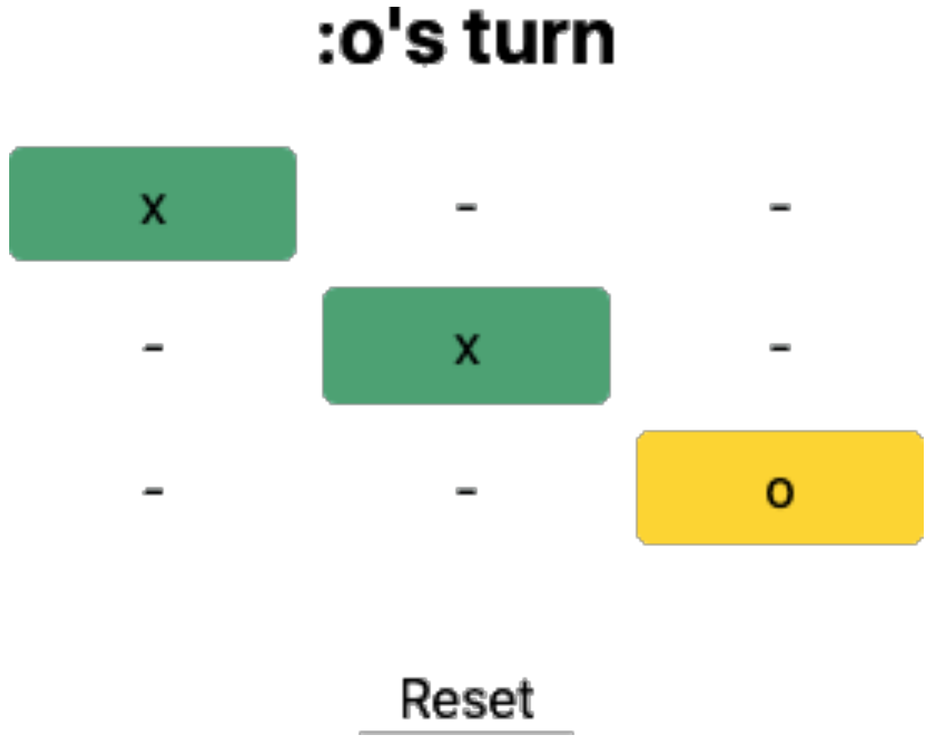

JUXT
Krim Labs

# REPL Demo - Tic Tac Toe

## State

```
Lattice
[[:x nil nil]
 [nil :x nil]
 [nil nil :o]]

Turn :o
```

## React




JUXT

Krim Labs

# With REPL - Develop only a function

- Precisely load the relavent function
- Test with inline feedback
- Rich Comments in git history

Demo source
**krimlabs/workshops**

JUXT
Krim Labs

# Editor Integration

(cider[])

Cursive

Calva

Fireplace

JUXT

Krim Labs

# Interop with JS ecosystem

JUXT

Krim Labs

# Google Closure Compiler

- Open for 10+ years
- Google Closure Lib
- Cross Platform

JUXT

Krim Labs

# ;; Just add js/foo

```
(.getElementById js/document "root")
```

```
document.getElementById(“root”)
```

```
(new js/FormData)
```

```
new FormData()
```

```
(clj->js {:hello :world})
```

```
#js {:hello “world”}
```

JUXT

Krim Labs

# ;; Consuming NPM Packages

```
(:require ["react-dom/server" :refer [renderToString]])

(renderToString react-element)
```

# ;; Write Node Libs

```
(ns demo.lib)
(defn shdILrnTypScrpt []
  ;; hope that ts people are not paying attention
  (prn “really?”)
  false)

{...
  {:target :node-library
           :output-to "out/demo-library/lib.js"
           :exports {:hello demo.lib/shdILrnTypScrpt}}}
```

JUXT

Krim Labs

# ;; Write NPM Packages

```
(defn ^:export shdILrnTypScrpt [] ...)
```

# ;; Write Node Libs

```
$ cd out/demo-library
$ node
> var x = require('./lib');
undefined
> x.shdILrnTypScrpt()
really?
false
```

JUXT

Krim Labs

U:— - -  * scratch *  All  L1  [(Lisp Paredit)]

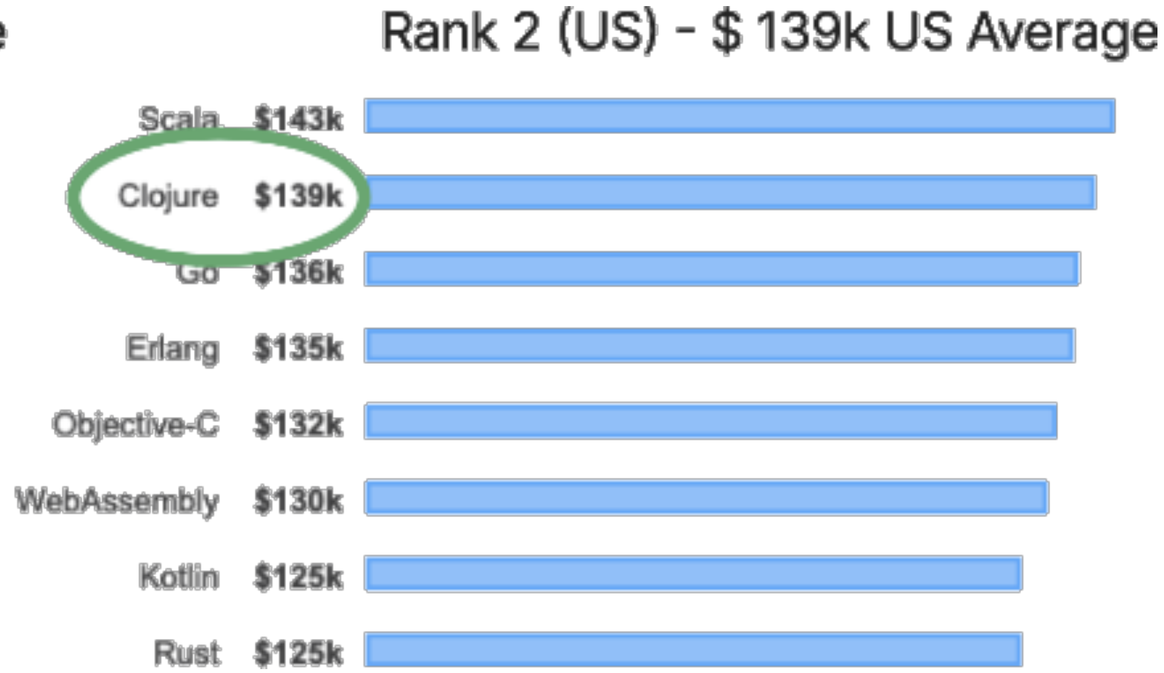
# ;; React using Hiccup

```
(defn Settings [username realname email]
  [:div#settings.margin-all-3
    [:h3 username]
    [:div realname]
    [:div email]
    [:button {:onClick #()} “Edit”]])
```

# ;; JS Equivalent

```
function Settings(username, realname, email) {
  return (<div id="settings" class="margin-all-3">
    <h3>{username}</h3>
    <div>{realname}</div>
    <div>{email}</email>
    <button onClick={() => {…}}>Edit</button>
  </div>);
}
```

# Clojure Devs are best paid


StackOverflow Developer Survey-2019

JUXT

Krim Labs

# What about Types ?

JUXT

Krim Labs

# clojure.spec

- Optional type library
- Define types as predicates

JUXT

Krim Labs

# ;; Spec domain entities

```
(s/def ::first-name string?)
(s/def ::last-name string?)
(s/def ::phone-number (s/and int?
                             #(<= % 9999999999)
                             #(>= % 100000000)))

(s/def ::person
  (s/keys :req-un [::first-name ::last-name]
          :opt-un [::phone-number]))
```

JUXT

Krim Labs

# Spec Demo

## State

```
:first-name
:last-name
:phone-number
```

## Validation

```
(s/valid?  ::person %)
```

## API (Stub)

```
POST /person
```

JUXT

Krim Labs

# Spec Benefits

- Optional opt in
- Generative testing inside source
- Just predicates

- Spec code (not just data)*

*https://clojure.org/guides/spec

JUXT
Krim Labs

# ;; Spec – going crazy

 danielcompton/defn-spec

```
(s/def ::int int?)

(ds/defn my-inc :- ::int
  [a :- ::int]
  (+ a 1))
```

 jeaye/orchestra

```
(defn-spec my-inc int?
  [a int?]
  (+ a 1))
```

JUXT

Krim Labs

# Going crazy demo

JUXT

Krim Labs

# Building on strong foundations

- Simple & robust semantics
- Language level functionality as lib
- Even bigger on the server side

JUXT

Krim Labs

# Will stay in prod after 10 years ?

JUXT

Krim Labs

# Getting started

JUXT

Krim Labs

# Build Systems

## Lein + Figwheel

https://leiningen.org/

https://figwheel.org/

## Shadow CLJS

https://shadow-cljs.org

JUXT

Krim Labs

# React Wrappers

**Reagent**

https://reagent-project.github.io/

*Reagent is what React hopes to be one day*

*— Luke Whorton (https://medium.com/@lwhorton/)*

JUXT

Krim Labs

# React Wrappers

**Reagent**

https://reagent-project.github.io/

**hx**

https://github.com/Lokeh/hx

JUXT

Krim Labs

# People

**Rich Hickey**
Creator of Clojure

**Stuart Halloway**
Core Team Member

**Alex Miller**
Core Team Member

**Stuart Sierra**
Component; Reloaded

**Malcolm Sparks**
Yada/Bidi/Tick

**James Reeves**
Integrant/ Hiccup/ Medley

**Bruce Hauman**
Figwheel

**Bozhidar Batsov**
nrepl

**David Nolen**
ClojureScript

**David Miller**
ClojureCLR

JUXT

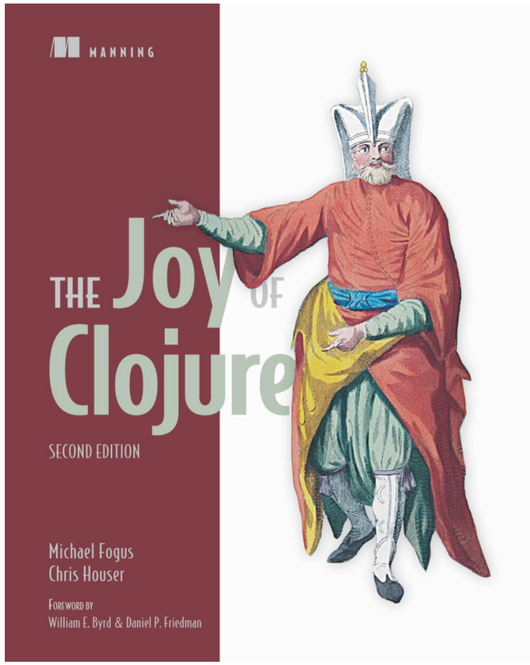
Krim Labs

# Blogs & Docs

braveclojure.com

PurelyFunctional.tv

purelyfunctional.tv




The Joy of Clojure;
By Fogus & Houser

JUXT

juxt.pro/blog

CLOJURE KOANS

clojurekoans.com
clojurescriptkoans.com

JUXT

Krim Labs

# Thank You


## Shivek Khurana

Clojure Consultant at **JUXT Inc**
Independant Consultant at **Krim Labs**


## Around the web

medium.com/@shivekkhurana
github.com/shivekkhurana

**twitter.com/shivek_khurana**

JUXT

Krim Labs