CROSS PLATFORM DATE-TIME AWESOMENESS
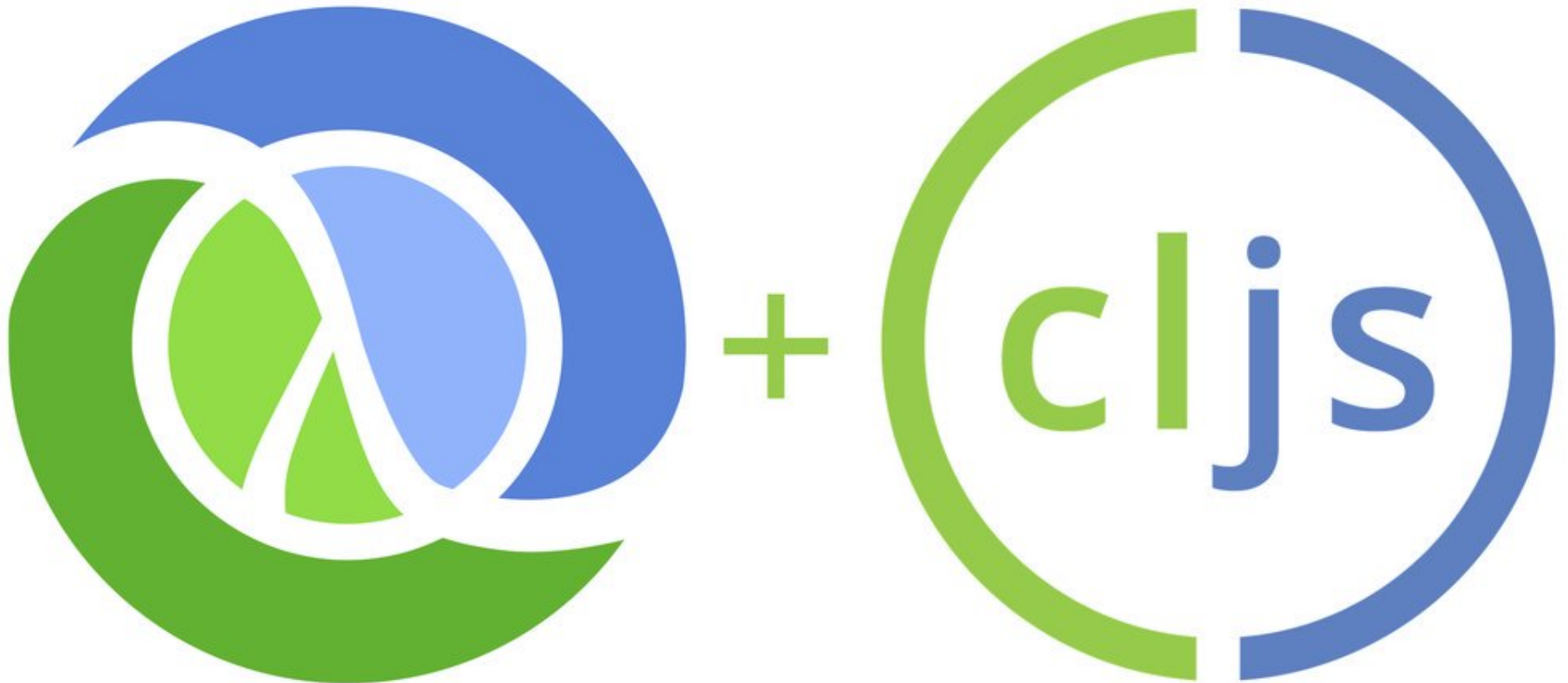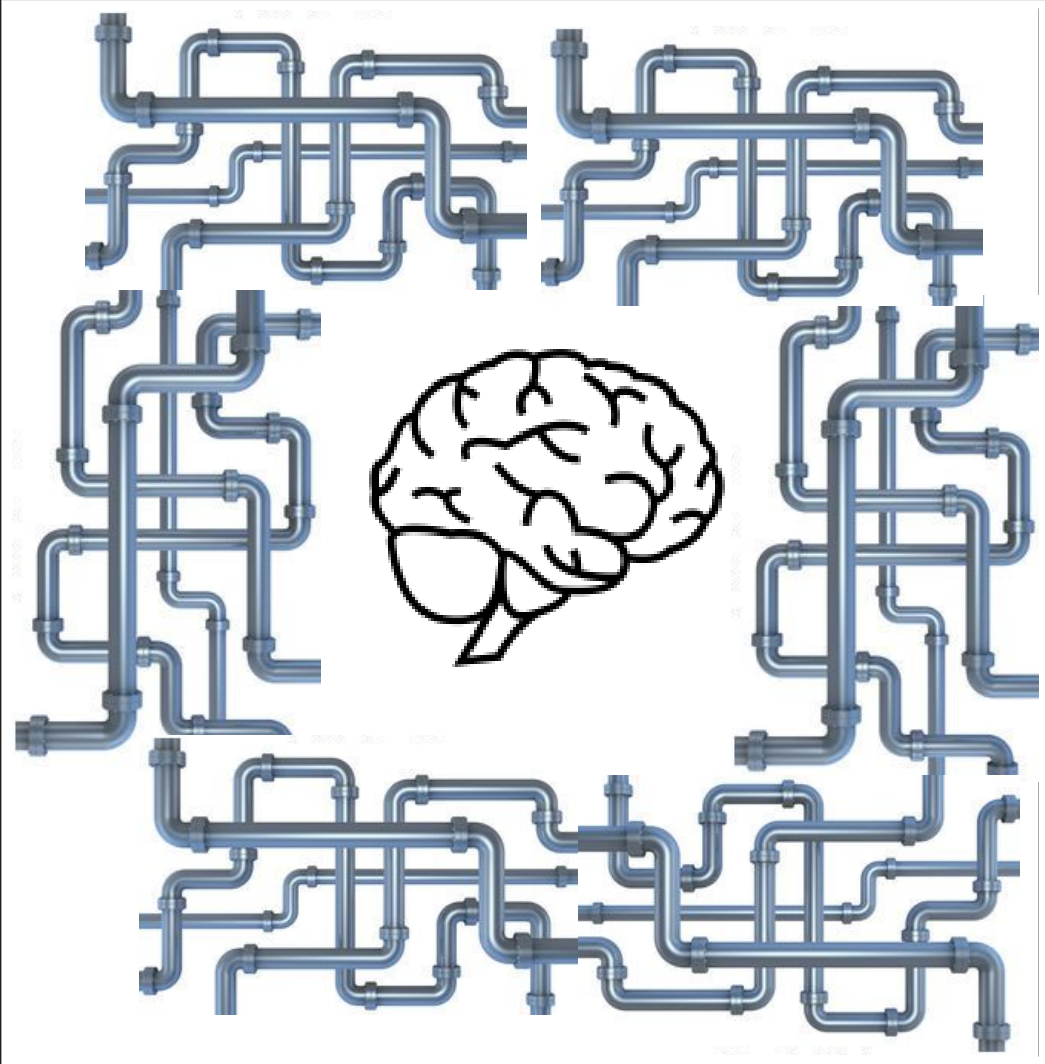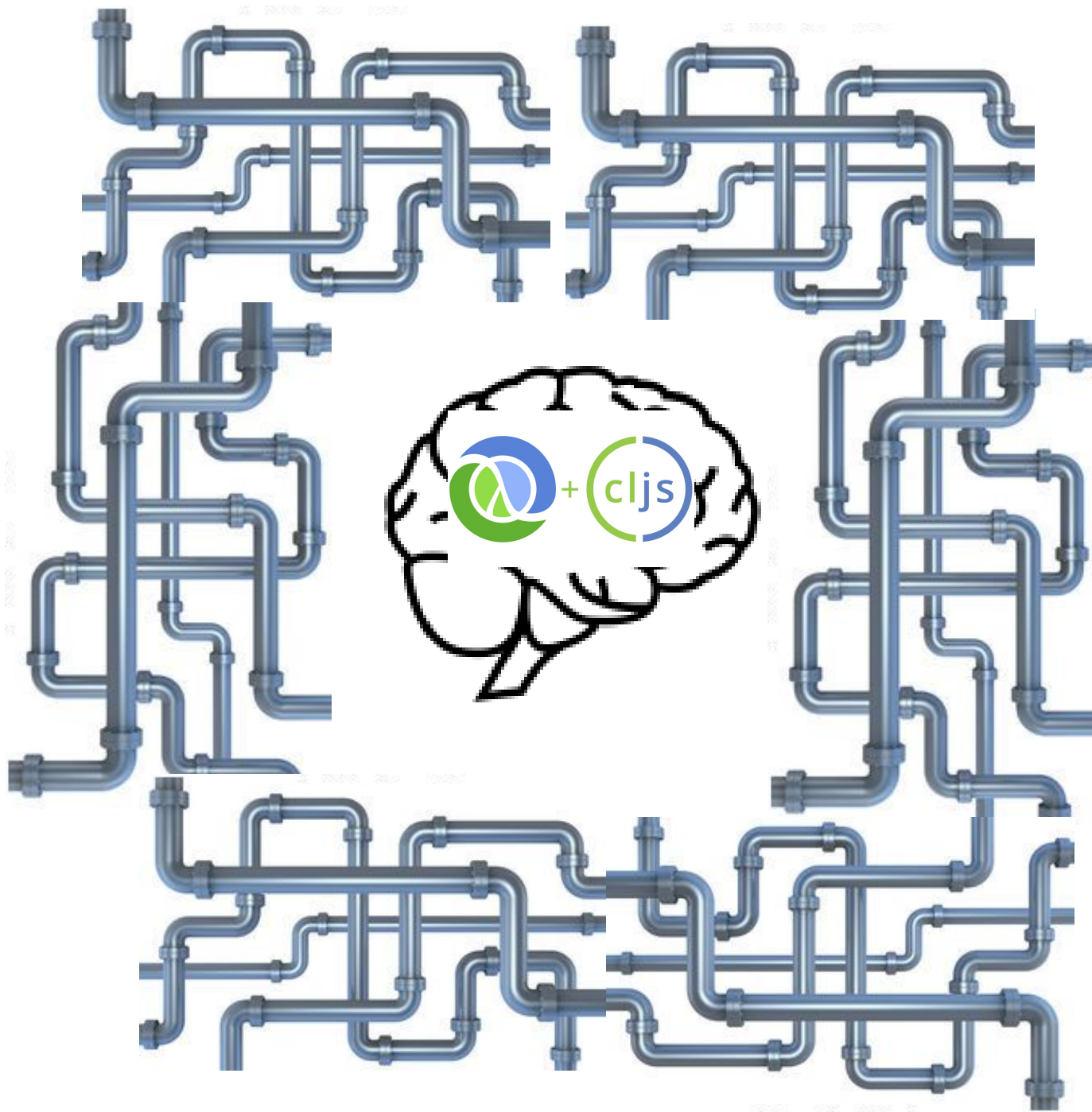
BY HENRY WIDD

@HENRYW374

```
(ns foo.cljc
  (:require
    #?(:clj  [clj-time.core  :as ct]
       :cljs [cljs-time.core :as ct])))

(ct/before? (ct/today) (ct/today))
; => true (in cljs)
```

WHY CROSS-PLATFORM CODE?

#?@(...)

#?@(...)

Maths: kixi.stats

#?@(. . .)

Maths: kixi.stats

Strings: cuerdas

#?@ (. . .)

Maths: kixi.stats

Strings: cuerdas

Data: frankiesardo/linked

#?@ (. . .)

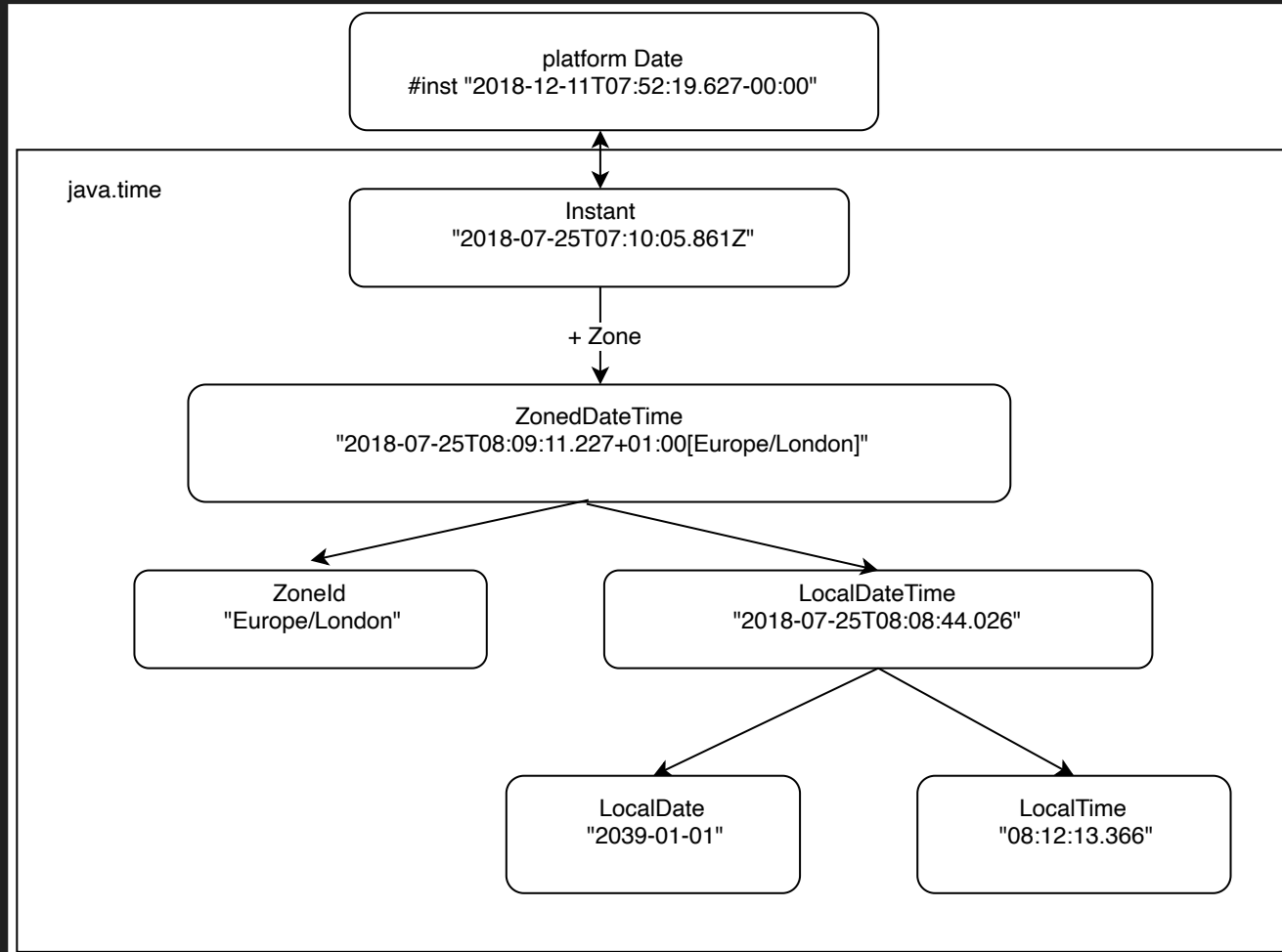Maths: kixi.stats

Strings: cuerdas

Data: frankiesardo/linked

Dates: _____

JAVA.TIME & JS-JODA DATE OBJECTS

platform Date
#inst "2018-12-11T07:52:19.627-00:00"

java.time

Instant
"2018-07-25T07:10:05.861Z"

+ Zone

ZonedDateTime
"2018-07-25T08:09:11.227+01:00[Europe/London]"

ZoneId
"Europe/London"

LocalDateTime
"2018-07-25T08:08:44.026"

LocalDate
"2039-01-01"

LocalTime
"08:12:13.366"

CROSS-PLATFORM JAVA.TIME

```clojure
(ns foo
  (:require
    #?(:cljs [java.time :refer [LocalDate]])    ;(1)
    [time-literals.read-write])
   #?(:clj (:import [java.time LocalDate])))    ;(2)

  (def d (. LocalDate parse "2015-10-21"))      ;(3)
  ;=> #time/date"2015-10-21"                     ;(4)

  (= d (. LocalDate now))
  ; => false

  (-> d
      (.atStartOfDay (. ZoneId systemDefault))) ;(5)
; #time/zoned-date-time"2015-10-21T00:00-04:00[America/New Yor
```

A CLOJURE WRAPPER FOR X

- Provides?

  - Avoid Interop Code
  - Clojurey Version of the API (e.g. data in/out)
  - Extra features (API, Immutability etc)

- Need to know:

  - Do I need to know x? Is the API tied to x?
  - Do x's docs, community & Stack overflow translate

CLJC.JAVA-TIME

```clojure
(ns my.cljc
  (:require [cljc.java-time.local-date :as local-date]) ;(1)

(def d (local-date/parse "2019-01-01"))                 ;(2)

(local-date/at-start-of-day d)
; => #time/date-time"2020-12-01T00:00"
```

CLOJURE.JAVA-TIME

- Clojurey-ness

  - Data API - fields and units
  - Single namespace

- Features

  - `(with-clock ...)`
  - Conversions to/from Joda, java.sql.* & other date types

TICK

TICK

Authored by Malcolm Sparks @ JUXT

TICK

Authored by Malcolm Sparks @ JUXT

Power Tools for working with Time

How much annual leave has an employee used up?

How much annual leave has an employee used up?

Inputs:

How much annual leave has an employee used up?

Inputs:

what periods have they been out of office?

How much annual leave has an employee used up?

Inputs:

what periods have they been out of office?

Relevant public holiday calendar

How much annual leave has an employee used up?

Inputs:

what periods have they been out of office?

Relevant public holiday calendar

annual leave allowance start/end

INTERVAL OPERATIONS

```
(require '[tick.alpha.api :as t])

(def working-days
  (t/difference                                   ;(1)
    #:tick{:beginning #time/date "2012-01-01",
           :end       #time/date "2020-01-01"}
   weekends
   public-holidays))

; => (#:tick{:beginning #time/date "2012-01-03" ;(2)
;           :end       #time/date "2012-01-07"}
;     .... )

(def holidays-by-year
  (->
```

TICK

```
(-> (t/time "10am")
    (t/on "2019-04-20")
    (t/in "America/Toronto"))
```

RECAP

```
[tick] ;(alpha)
  [cljc.java-time]
    [cljs.java-time]
      [cljsjs/js-joda]

[time-literals]
  [cljs.java-time]
```

FOR THE FUTURE ...

- tc39/proposal-temporal & Intl api
- specs (and generators, similar to `s/inst-in`)
- tick - '1.0' - more docs etc
- automated tracking and publishing of discrepancies between java.time and js-joda
  - incl. methods added to java.time in Java 9 and higher
  - decide what cljc.java-time should do

TO WRAP UP...

- I got the cross platform date-time APIs I wanted
  - big shoutout to the JS-Joda authors & others
- Consider defaulting to cljc

LINKS

@henryw374

- My Blog
- tick
- time-literals
- cljc.java-time