BeClojure School
A gentle introduction to Clojure

Stijn Opheide
WebComrades

BeClojure introduction

Belgian Clojure User Group
Started in 2013
83 members

Our Mission

BeClojure tries to bring together companies, universities, students and hobbyists to share their experiences and expand their knowledge.

What is Clo*j*ure?

“Hello World”

Everything is Data

```
{:firstName "Mary"
 :lastName "Rose"
 :age 26
 :address {
   :street "610 5th Avenue"
   :city "New York"
   :state "NY"
   :postalCode "10021"}
 :phoneNumber
   [{:type "mobile" :number "378 228-6789"}
    {:type "work" :number "271 172-8271"}]}
```

Clojure Data

Integer	12345
Double	2.34
Ratio	3/4
String	“beclojure”
Character	\c

Clojure Data

Boolean	true, false
Symbol	brujug
Keyword	:stijn
Null	nil
Regex	#”[abc].*\d"

Clojure Data: Collections

Vector	[1 2 3 4 5]
List	(1 2 3 4 5)
Map	{:a 1 :b 2}
Set	#{1 2 3 “a”}

Function call

```
(+ 1 2)
```

Function call

```
(defn hello
  "Says hello to who."
  [who]
  (str "Hello, " who))
```

What is Clojure?

LISP

What is Clojure?

LISP

Functional

What is Clo*j*ure?

LISP

Functional

Dynamic

What is Clojure?

LISP
Functional
Dynamic
Hosted

LISP

LISt Processing

Code as data

REPL

LISP

REPL

Read Eval Print Loop

Functional

First-class functions

Functional

First-class functions

Immutable data

Functional

First-class functions
Immutable data

In computer science, functional programming is a programming paradigm—a style of building the structure and elements of computer programs—that treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data.

— Wikipedia

Dynamic

Flexible
Interactive
Easier to explore problem
Less verbose

Hosted

JVM

Javascript

(CLR)

Control flow

```
(if true
  "Brussels"
  "Antwerp")

;=> "Brussels"
```

Control flow

```
(if true
  "Brussels"
  "Antwerp")

;=> "Brussels"

(if false
  "Brussels"
  "Antwerp")

;=> "Antwerp"
```

Expressions

```
(defn max
  [a b]
  (if (> a b) a b))
```

Expressions

```clojure
(defn max
  [a b]
  (if (> a b) a b))

(defn create-person
  [first-name last-name]
  {:first-name (or first-name "John")
   :last-name (or last-name "Doe")})
```

Truthy, falsey

```
(if "this is true"
  "yep"
  "no")

;=> "yep"

(if nil
  "not this"
  "but this")

;=> "but this"
```

Truthy, falsey

```clojure
(or false nil :beer :wine)
; => :beer

(or (= 0 1) (= "yes" "no"))
; => false

(or nil)
; => nil
```

Truthy, falsey

```
(and :beer :wine)
; => :wine

(and true :macaroni nil false)
; => nil
```

Data structures: maps

```clojure
(get {:a 0 :b 1} :b)
; => 1

({:a 0 :b 1} :a)
; => 0

(:a {:a 0 :b 1})
; => 0

(assoc {:a 0 :b 1} :c 2)
; => {:a 0 :b 1 :c 2}

(dissoc {:a 0 :b 1} :a)
; => {:b 1}
```

Data structures: vectors & lists

```clojure
(get [3 2 1] 0)
; => 3

(conj [1 2 3] 4)
; => [1 2 3 4]

(nth '(:a :b :c) 2)
; => :c

(conj '(1 2 3) 4)
; => (4 1 2 3)
```

Data structures: sets

```clojure
(conj #{:a :b} :c)
; => #{:a :b :c}

(conj #{:a :b} :b)
; => #{:a :b}

(set [3 3 3 4 4])
; => #{3 4}

(get #{:a :b} :a)
; => :a

(:a #{:a :b})
; => :a
```

DEMO

Why Clojure?

Simple (not easy)

Functional

Data oriented

Made for concurrency

JVM

How do I start?

Plenty of books

Rick Hickey's Greatest Hits

Youtube ClojureTV

4Clojure, Clojure Koans

Parens of the Dead

Clojure Gazette

Grimoire

Attend BeClojure meetups ;)

Editors / IDEs

Cursive (IntelliJ)
Counterclockwise (Eclipse)
Vim (Fireplace)
Emacs (CIDER, Spacemacs)
Lighttable

More?

core.logic: Logic programming
core.typed: Static typing
core.match: Pattern matching
core.async: Async through CSP

Thank you!

http://www.meetup.com/BeClojure/

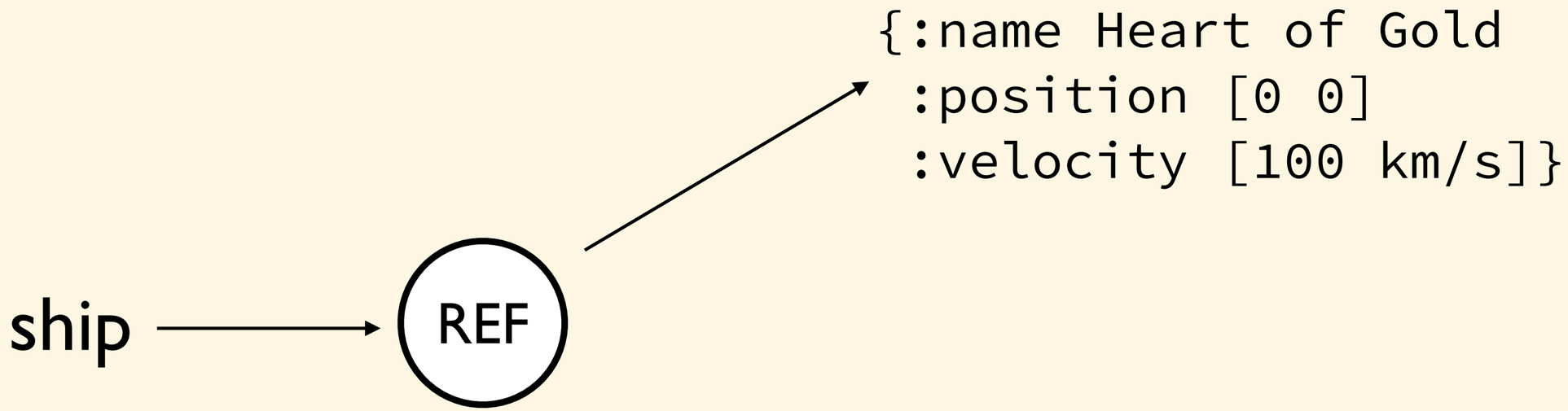
State

Problem: swap values of 2 variables

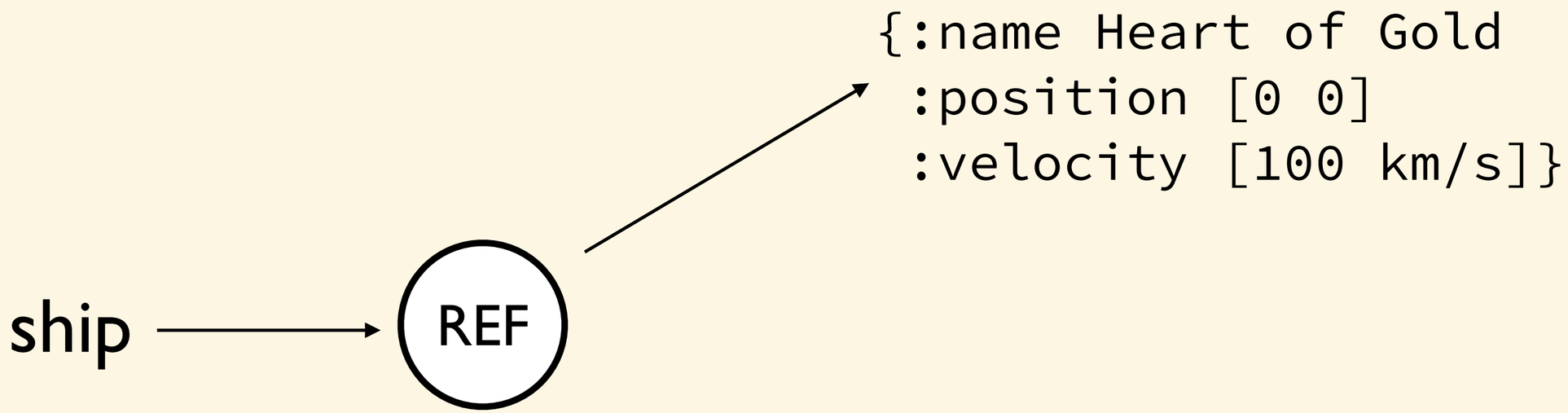
```
a=42; b=3;
c=a;
a=b;
```


STATE
You're Doing It Wrong

State

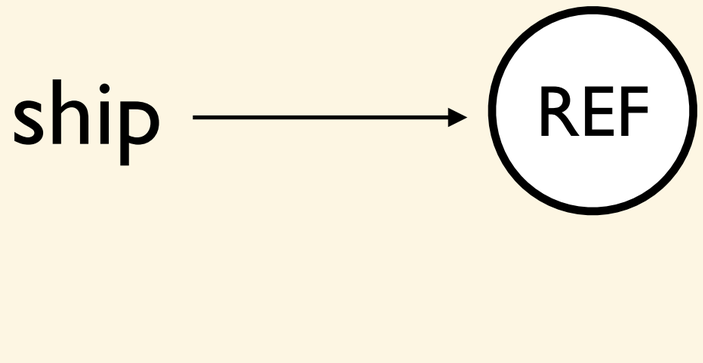


State


ship
REF
{:name Heart of Gold
:position [0 0]
:velocity [100 km/s]}
{:name Heart of Gold
:position [100 50]
:velocity [200 km/s]}

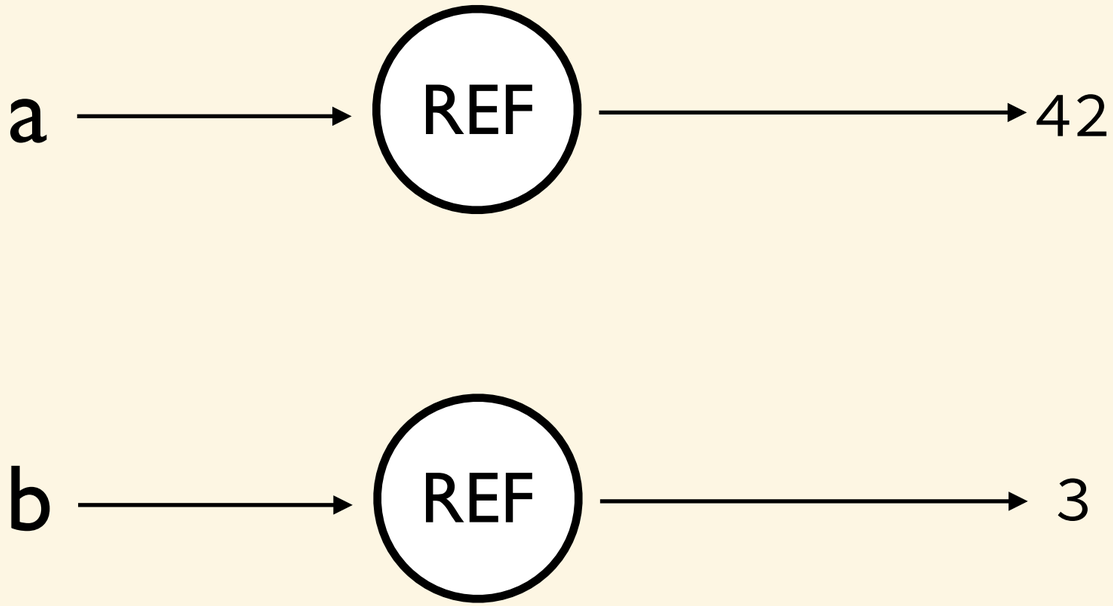
State

```
{:name Heart of Gold
 :position [0 0]
 :velocity [100 km/s]}
```




```
{:name Heart of Gold
 :position [100 50]
 :velocity [200 km/s]}
```

State



State