clojure & cfml sitting in a tree

sean corfield
world singles

how to go faster (with (parentheses))

sean corfield
world singles

world singles

world singles

- founded in 2001
- internet dating platform
  - niche ethnic markets (primarily)
- multi-lingual: half dozen languages
- multi-tenant: 50 sites, one code base

DESIKISS.COM™
Connecting Indians Worldwide

Already a member? Sign In

Find Your Indian Partner For Life, Love and Marriage.

I am -Select-

From -Select-

Birthday -Month- -Day- -Year-

Username

Password

Email Address

By clicking Join Now, you agree to the Terms and Conditions

Join Now >

Arjun contacted me on DesiKiss and our lives haven't been the same since. We are now happily married and are awaiting our first child. After trying several other online dating sites, DesiKiss is where our fates collided. We are both eternally grateful for the opportunity you have afforded us.

- Sreeja (3jaLove)

LovingBBW.com™
Meet BBW Singles

Already a member? Sign In

Find Your BBW Partner For Life, Love and Marriage.

I am -Select-

From -Select-

Birthday -Month- -Day- -Year-

Username

Password

Email Address

By clicking Join Now, you agree to the Terms and Conditions

Join Now >

I met someone very special on this site. This is the best site I've ever been on and I've tried EVERY SINGLE ONE OF THEM. Thank you LovingBBW. We are talking about getting married and buying a home next year. Thanks again!

- Rick (HonestOne8)

HaitiDating.com™
Connecting Haitians Worldwide

English français Déjà membre? Se connecter

Trouvez votre partenaire Haitien(ne) pour la vie, l'amour et le mariage.

Je suis – Sélectionnez –

De – Sélectionnez –

Anniversaire – Jour – – Mois – – Année –

Pseudo

Mot de passe

Adresse email

En cliquant sur S'inscrire maintenant, vous acceptez les Conditions Générales

S'inscrire maintenant >

Au début, je n'étais pas sûr que les rencontres en ligne étaient faites pour moi. J'ai hésité à créer mon profil. Mais le lendemain, j'ai reçu un message de ma future âme sœur! Nous avons bavardé pendant quelques jours puis nous nous sommes rencontrés. C'était le jour le plus fou de ma vie. Il était doux et beau. Nous avions tous les deux fréquenté la même université, mais à des moments différents. Nous étions faits pour être ensemble. Milles mercis d'avoir rendu cette rencontre possible !!

- Martine (LaBelle75)

world singles

- cfml from day one
- monolithic procedural code base
  - no frameworks, no cfcs, coldfusion 8

world singles

- started green field rewrite in spring 2009
  - oop, frameworks, coldfusion 9 / railo
  - unit tests, continuous integration
  - first site live on new platform fall 2010
  - all sites live by may 1st 2012

not shown to scale

- 3,000,000 members
- 1,000,000 emails / day
- 80,000 logins / day
- 2,000 concurrent users (average)

not shown to scale

- cfml
  - 210 cfcs, 38.0kloc - model, controller
  - 426 cfms, 33.4kloc - views (& layouts)
  - 52 test cfcs, 5.3kloc
- coldbox, coldspring, reactor, cfmljure
- mxunit, selenium, ant (and a little fw/1)

not shown to scale

- clojure
  - 46 source files, 7.1kloc - model
  - 28 test files, 1.5kloc

not shown to scale

- clojure
  - 46 source files, 7.1kloc - model
  - 28 test files, 1.5kloc
- equivalent to 4x - 10x as much cfml!

not shown to scale

- clojure
  - 46 source files, 7.1kloc - model
  - 28 test files, 1.5kloc
- equivalent to 4x - 10x as much cfml!
- effectively over half of our model is clojure

View.cfm

Controller.cfc

Model.cfc

ColdBox

ColdSpring

cfmljure

Reactor

Model.clj

MySQL

MongoDB

Discovery

clojure

what is clojure?

- "a dynamic programming language that targets the Java Virtual Machine"
- a modern lisp

```
(func arg1  arg2 arg3)
;; func( arg1, arg2, arg );
```

what is clojure?

- everything is a function

```
(* (+ 1 2 3) (- 4 5) (/ 6 7))
;; (1 + 2 + 3) * (4 - 5) * (6 / 7)
```

what is clojure?

- java libraries are easily accessible

```
(defn ** [x y] (Math/pow x y))
(** 4 2) ;; 16

;; function starstar( x, y ) {
;; var m=createObject("java","java.lang.Math");
;; return m.pow( x, y ); }
```

what is clojure?

- immutable data - no "variables"
  - no assignment - only initialization
- operate on collections - no "loops"
  - map, filter, reduce
- higher-order functions
  - take fns as arguments and/or return fns

what is clojure?

```
(map send-match-email
    (query db
        (select * :user
            (where {:status "approved"}))))
```

```
;; select from user where status = 'approved'
;; call send-match-email on each row
```

what is clojure?

```
(pmap send-match-email
    (query db
        (select * :user
            (where {:status "approved"}))))

;; select from user where status = 'approved'
;; call send-match-email on each row
;; in parallel!
```

what is clojure?

```
(defn is-approved? [user]
    (= "approved" (:status user)))

;; function is_approved(user) {
;;   return "approved" = user.status; }

(filter is-approved?
    (query db (select * :user)))

;; select all users, call is-approved? on each
;; row, return rows for which it is true
```

what is clojure?

```
(filter (fn [user] (= "approved" (:status user)))
    (query db (select * :user)))

;; like closures in coldfusion 10 / railo 4

;; function(user) {
;;   return "approved" = user.status; }
```

what is clojure?

```
(let [users (query db (select :age :user))
      sum (reduce (fn [s u] (+ s (:age u))) 0 users)
;; var sum = 0; for ( var u in users ) sum += u.age;
      n (count users)]
    (/ sum count))

;; return average age of users - makes two passes
;; also holds on to entire list of users
```

what is clojure?

```
(let [[count sum]
      (query db :result-set (partial reduce
         (fn [[n s] u] [(inc n) (+ s (:age u))])
         [0 0])
         (select :age :user))]
  (/ sum count)

;; return average age of users
;; uses just one pass and fixed space
```

what is clojure?

```
(let [[count sum]
      (query db :result-set (partial reduce
         (fn [[n s] u] [(inc n) (+ s (:age u))])
         [0 0])
         (select :age :user))]
  (/ sum count)

;; return average age of users
;; uses just one pass and fixed space
```

what is clojure?

```
(let [[count sum]
      (query db :result-set (partial reduce
         (fn [[n s] u] [(inc n) (+ s (:age u))])
         [0 0])
         (select :age :user))]
  (/ sum count)

;; return average age of users
;; uses just one pass and fixed space
```

what is clojure?

```
(let [[count sum]
        (query db :result-set (partial reduce
            (fn [[n s] u] [(inc n) (+ s (:age u))])
            [0 0])
            (select :age :user))]
    (/ sum count)

;; return average age of users
;; uses just one pass and fixed space
```

what is clojure?

```
(let [[count sum]
      (query db :result-set (partial reduce
        (fn [[n s] u] [(inc n) (+ s (:age u))])
        [0 0])
        (select :age :user))]
  (/ sum count)

;; return average age of users
;; uses just one pass and fixed space
```

clojure is good for?

- big data & analytics
- logic programming & pattern matching
- heavy concurrency & massive scale
- hard problems!

clojure is also...

- "It endeavors to be a general-purpose language suitable in those areas where Java is suitable."
- -- http://clojure.org/rationale

world singles and cfml
...and clojure

clojure & us

- needed to process large amounts of data
- 2009 introduced scala for "heavy lifting"
  - (irony: one of clojure's touted strengths)
  - scala not a good cultural fit for our team
- 2010 started evaluating clojure
- 2011 first production usage

why add clojure?

- fast (compiles to "static java" style code)
- immutable (automatic thread safety)
- built-in concurrency / parallelism
- lazy functions (big data scale in fixed space)

clojure & us

- environment control
  - dev / ci / qa / production
  - per-environment settings
  - weaned us off coldbox environment stuff

clojure & us

- environment control
- logging
  - wrapped around log4j
  - custom dbappender writes to mysql
  - weaned us off coldbox logging stuff

clojure & us

- environment control
- logging
- persistence
  - evolved out of logging to db
  - full-fledged data mapping library
  - start weaning us off reactor!

clojure & us

- email
  - html generation & sending
  - tracking & log file analysis
- geo location (rest/json)
- i18n (lookup & formatting)
- reporting (parallel queries)
- search engine interaction (json/xml)

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...evaluate...)
```

putting it all together

```
// CFML
var searchData = variables.clj.worldsingles.search
    .populate_my_match_criteria(
        rc.siteUser.asClojure(),
        rc.siteObject.asClojure() );

;; Clojure
(ns worldsingles.search)
(defn populate-my-match-criteria [user site]
  ...)
```

ClojureService.cfc

ClojureService.cfc

- code example shows how to use cfmljure to integrate clojure into cfml easily
- also shows how many namespaces we import into our cfml code and call directly
- plus some examples of cfml code and equivalent clojure code side-by-side

persistence
abstraction

persistence

- crud wrapper around clojure.java.jdbc
  - jdbc talks to mysql (and others)
- (get-by-id :table id)
- (find-by-keys f :table {:key val})
- (save-row :table {:id pk ...})
- (execute f "sql statement" [params])

persistence

- congomongo is a mongodb library
- same crud wrapper around congomongo!
  - (except for "execute")
- abstraction allows us to mix'n'match mysql data and mongodb documents

persistence

- generic application code uses mysql or mongodb transparently based on :table
- (get-by-id :table id)
- (find-by-keys f :table {:key val})
- (save-row :table {:id pk ...})

simple concurrency

reporting.clj

reporting.clj

- code example shows use of `future` to easily run multiple queries in separate threads and to 'join' threads via `@` / `deref`
- also showed other code that used `delay` to create cached data / singletons and some uses of `pmap` for parallel computations

clojure ecosystem & summary

community

- over 6,700 developers on mailing list
- about 300 developers online on irc 24x7
- very active library development
  - #23 language on github (cfml is #40)
  - over 7,000 projects on github
  - nearly 2,000 libraries on clojars.org

books

- clojure programming - o'reilly
- the joy of clojure - manning
- programming clojure (2ed) - pragmatic
- clojure in action - manning (outdated)
- practical clojure - apress (very outdated)

learn clojure online

- try clojure: http://tryclj.com
- 4clojure (puzzles): http://4clojure.com
- clojure koans - download, run & code: https://github.com/functional-koans/clojure-koans/

clojure benefits

- performs well (better than cfml)
- scales well for data complexity
- immutable data provides thread safety
- maintenance / flexibility is good
- easily callable from cfml (on railo)
- all of java easily accessible (incl. libraries)

clojure & the future

- training more of our team
- clojure code base is growing
- most new backend code will be clojure
- looking at cascalog - big data processing

cfml & the future

- view-controller will stay in cfml
  - cfml is still the best web (html) language
- model will shift to clojure over time
  - as needed for performance / convenience
  - prototype new model code in cfml?

cfml & the future

- reactor will go away
  - clojure persistence layer is much faster!
- coldbox will go away (replaced by fw/1)
  - eventually... weaning off will be slow :(
- coldspring may go away (replaced by di/1)
  - with the model in clojure, we don't need much dependency injection in cfml

questions? / contact

- @seancorfield
- sean@worldsingles.com
- http://worldsingles.com
- http://corfield.org