Clojure Web Development

Dynabyte
  -> Stardoll

www.cleancode.se
@mikaelsundberg
micke@cleancode.se

Clojure

Dynamic
JVM
Functionall
Lisp
Macros
Ho

Conjure

“The rails like framework”

- Scaffolding
- Form
- Ajax
- Migrations
- MVC

Webprojects

```
(defproject MyCoolProject "1.0.0-SNAPSHOT"
  :description "FIXME: write"
  :dependences [[org.clojure/clojure "1.2.0"]
                [org.clojure/clojure-contrib "1.2.0"]
                [enlive  "1.0.0-SNAPSHOT"]
                [net.cgrand/moustache "1.0.0-SNAPSHOT"]
                [ring/rig-core "0.2.5"]
                [ring/ring-devel "0.2.5"]
                [ring/ring-jetty-adapter "0.2.5"]])
```

```
cat project.clj | wc -l  => 13
cat pom.xml | wc -l => 104
```

Basics

```
(defroutes my-routes
 (ANY "/hello" [] "<h1>hello</h1>"
  (GET "/cars/:model/:color"  model color]
       (get-cars model color))
 (POST "/cars/"  {params :params}]
       (save-car params)))


(run-jetty (app #'my-routes) {:port 8080
                               :join? false})
```

Hiccup

```
(html [:h1 "Hello"]
    [:div {:id "main"}
     [:ul
      (for [item (range 1 5)]
        [:li item])]])
```

```
<h1>Hello</h1>
<div id=\"main\">
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
</ul>
</div>
```

Hiccup

```
(include-js "javascript.js")

(html (include-js "javascript.js"))

(html (select-options [["ddd" 1] ["bdd" 2] ["tdd" 3]]))
```

Templating

```
<table id="items">
   <tr class="bookmark">
      <td class="title"></td>
      <td class="url"></td>
    </tr>
 </table>
```

Enlive

```html
<table id="items">
   <tr class="bookmark">
       <td class="title"></td>
       <td class="url"></td>
    </tr>
 </table>
```

```clojure
(deftemplate index "index.html" [items]
 [:table#items :tr.bookmark]
        (clone-for [item items]
             [:td  title]   (content (:title item))
             [:td url]    (content (:url item))))
```

Enlive

```
[{:title "Community driven docs" :url "www.clojuredocs.org"}
{:title "Clojure MainPage" :url "www.clojure.org"}]

<table id="items">
   <tr class="bookmark">
       <td class="title">Clojure MainPage</td>
       <td class="url">www.clojure.org</td>
    </tr>
   <tr class="bookmark">
       <td class="title">Community driven docs</td>
       <td class="url">www.clojuredocs.org</td>
    </tr>
 </table>
```

Json

```
(defroutes my-routes
  (GET "/json/cars/:model/:color"  model color
    (json/json-str (get-cars model color))
```

Google App Engine

Appengine magic https://github.com/gcv/appengine-magic

lein appengine-new

```
(ds/defentity Book [^:key isbn, title, author])

(def mybook (Book. "9789197901406" "Virka Söta Djur och andra
figurer" "Petra Sundberg"))

(ds/save! [mybook anotherbook athirdbook])

(ds/query :kind Book
          :filter (= :author "Petra Sundberg")
          :sort [[title :dsc] :isbn])
```

SQL

ClojureQL

```
(select posts (where (= :submitter “micke”)))

(conj! posts {:title      “Using moustache and enlive”
              :url        “www.cleancode.se”
              :submitter “micke”})
```

Java Interop

Vaadin

```
(defn main [args]
  (proxy [com.vaadin.Application] []
    (init []
      (let [app this]
        (.setMainWindow this
          (doto (new com.vaadin.ui.Window "Test application")
            (.addComponent
              (new com.vaadin.ui.Label "Hello Vaadin/LISP user!"))
            (.addComponent              (doto (new com.vaadin.ui.Button "button")
                (.addListener (proxy [com.vaadin.ui.Button$ClickListener] []
                                (buttonClick [event] (. (. app (getMainWindow))
(showNotification "test"))))))))))))
```

Testing

```
(defn show-bookmarks []
  (let [bookmarks (get-bookmarks)] // call to the database
    (if (empty? bookmarks)
      {:status 401}
      (response (index bookmarks)))))
```

Testing(Midje)

```
(fact
 "should return 401 when no bookmarks are present"
 (:status (show-bookmarks)) => 401
 (provided (get-bookmarks) => []))

(fact
 "should return 200 when there are bookmarks"
 (:status (show-bookmarks)) => 200
 (:body (show-bookmarks)) => (index [{:title "test" :url "www.cleancode.se"}])
 (provided (get-bookmarks) => [{:title "test" :url "www.cleancode.se"}]))
```

Testing (Midje)

```
fact
 "should return 401 when no bookmarks are present"
 :status show-bookmarks => 401
 provided get-bookmarks =>

fact
 "should return 200 when there are bookmarks"
  :status  show-bookmarks => 200
   :body show-bookmarks => index :title "test" :url "www.cleancode.se"
  provided get-bookmarks => :title "test" :url "www.cleancode.se"
```

Not in Clojure Web

"The Framework"
DB migrations.
Widgets. go learn javascript!

Why Clojure?

How Clojure?

http://groups.google.com/group/stockholm-clojure-user-group

http://clojure.org/
http://clojure.blip.tv/
http://clojuredocs.org/s
http://cleancode.se/
http://www.infoq.com/author/Rich-Hickey

freenode
#clojure
#clojure.se
#clojure-web