Quick Intro to Clojure

Evan Porter

Clojure

- Functional language
- A dialect of LISP
- Can target JVM / fully interoperable with Java
- Can target JavaScript (ClojureScript)
- Can target .Net’s CLR (ClojureCLR)

Why Clojure?

- Encourages immutability
- REPL (read-eval-print loop)
- It’s fun
- (Your definition of fun may differ)

Types

- nil
- Numbers: 42 or 3.14
- Ratios (note lack of whitespace): 3/4
- Characters: \A
- Strings: “Stuff”
- Keyword: :anything

Types (Cont.)

- List: (1 2 3 4 5)
- Vector: [1 2 3 4 5]
- Sets (note duplicates not allowed): #{1 2 3 4 5}
- Maps: {“key1” “value1” “key2” “value2”}

Some Code

- (println “Hello World”)
- (+ 2 3)
- (def some-name “Bill”)
- (def my-map { :first-name “Al”, :last-name “Pratt”})

Notes on the Code

- All the code examples listed are Lists
- Commas are whitespace (1 2 3) is the same as (1,,,,2,3,,,,)
- Keywords are ideally suited to be the keys on a map.

Code

- (def my-map { :name “Evan Porter” :occupation “Not Grand Master Flash”})
- (my-map :name) is equivalent to (:name my-map) and both return “Evan Porter”

How to start?

- Leiningen - http://leiningen.org
- Mascot has fantastic moustache
- lein new app my-app-name
- lein repl
- There’s also something called “boot”

Live Examples?