Clojure Crash Course

Generic Data Types

```
{:name "Clojure"
 :features [:functional :jvm :parens]
 :creator "Rich Hickey"
 :stable-version {:number "1.6.0"
                  :release "2014/05/23"}}
```

Functions

```clojure
(+ 1 2)
> 3
```

```clojure
(:city {:name "innoQ"
        :city "Monheim"})
> "Monheim"
```

```clojure
(map inc [1 2 3])
> (2 3 4)
```

```clojure
(map inc [1 2 3])
> (2 3 4)
```

```clojure
(filter odd? [1 2 3 4 5])
> (1 3 5)
```

```clojure
(reduce + 0 [1 2 3])
> 6
```

```clojure
(fn [x y] (+ x y))
> #<user$eval775$fn__776 user$eval775$fn__776@4f9faf3>
```

```clojure
((fn [x y] (+ x y)) 1 2)
> 3
```

```clojure
(def add
  (fn [x y] (+ x y)))
```

```clojure
(defn add [x y]
  (+ x y))
```

```clojure
(add 1 2)
> 3
```

```clojure
(defn statistics [numbers]
  {:sum (reduce + 0 numbers)
   :count (count numbers)
   :average (/ (reduce + 0 numbers)
               (count numbers))})
```

```clojure
(defn statistics [numbers]
  (let [sum (reduce + 0 numbers)
        cnt (count numbers)]
    {:sum sum
     :count cnt
     :average (/ sum cnt)}))
```

```clojure
(statistics [1 2 3 4 5])
> {:sum 15 :count 5 :average 3}
```

```clojure
(defn statistics
  "computes sum, count and average for a collection of numbers"
  [numbers]
  (let [sum (reduce + 0 numbers)
        cnt (count numbers)]
    {:sum sum
     :count cnt
     :average (/ sum cnt)}))
```

```clojure
(statistics [1 2 3 4 5])
> {:sum 15 :count 5 :average 3}
```

```clojure
(ns my.company.numerics)

(defn statistics
  "computes sum, count and average for a collection of numbers"
  [numbers]
  (let [sum (reduce + 0 numbers)
        cnt (count numbers)]
    {:sum sum
     :count cnt
     :average (/ sum cnt)}))
```

```clojure
(statistics [1 2 3 4 5])
> {:sum 15 :count 5 :average 3}
```