Übungen Programmieren in Clojure Serie 10

1. Infix in Clojure

Definieren Sie ein Makro `infix`, das es erlaubt eine binäre Operation in Infix-Notation auszuwerten, z.B. `(infix 1 + 2) => 3`

2. Trace

Schreiben Sie ein Makro `trace` mit dem man einen Funktionsaufruf protokollieren kann, z.B. `(trace + 1 2 3)` schreibt `(+ 1 2 3) => 6` auf die Standardausgabe und gibt 6 als Wert zurück.

3. Zufallsauswertung

Schreiben Sie ein Makro `(rand-expr x y)`, das zufällig `x` oder `y` auswertet, nicht aber den anderen Ausdruck.

4. Testüberprüfung

Schreiben Sie ein Makro `(assert-eq actual expected)`, das die beiden Ausdrücke auf Gleichheit überprüft und eine aussagekräftige Exception wirft, wenn sie nicht gleich sind, z.B. `(assert-eq (+ 1 2) 4)` führt zu einer Exception mit der Meldung `Expected '(+ 1 2)' to be '4', but got '3'!`

5. Addition vieler Werte

Schreiben Sie ein Makro `(++ & exprs)`, das eine Folge von Zahlen in einen geschachtelten Aufruf von + transformiert und dann auswertet, z.B. (++ 1 2 3 4) wird zu (+ 1 (+ 2 (+ 3 4)))

Rev 4.0 – 11. August 2017