The Patterns You Can't {See, Refactor}

Lisp Macros: Dizzying Heights

Kyle Burton

Relay Network

My Own Personal Blub* Paradox

*Language features that you don't yet understand just look _weird_, not ***powerful***.

Code That Writes Code

Compiler that targets Lisp

In Lisp

Without ‘eval’

!Refactorable

If, for, while

class, function, var, package

def, try, catch, throw

Special Powers

Reduce Syntax

Defer Execution

Introduce Bindings

Re-order computation

Reduce Syntax

Clojure: ..

```
java.util.Calendar.getInstance()
   .getTimeZone()
   .getDisplayName();
```

```
(.. (java.util.Calendar/getInstance)
    getTimeZone
    getDisplayName)
```

Defer Execution

```
(defmacro when [test & body]
  `(if ~test
     (do
       ~@body)))

(when (enemy-in-sight)
  (launch-missiles)
  (disallow-mineshaft-gap))
```

Introducing Bindings

Clojure: aprog1

```
(aprog1
  (construct-a-thing)
  ...manipulate ‘it’...))
=> it
```

Clojure: aprog1

```
(aprog1
  (expensive-lookup term)
  (update-cache it)
  (update-metric {:search it}))
=> lookup result
```

Re-order computation

Clojure: ->

```
x(y(z(5)));
```

```
(-> 5 z y x)
```

Clojure: ->>

```
(->>
   ["sort" "this" "by" "length"]
   (map (fn [s] [(count s) s]))
   (sort (fn [[l1 _] [l2 _]] (- l1 l2)))
   (map second))
```

Lets up the abstraction...

Warning: may induce vertigo

Scheme: FSM

```
(define m
  (automation init
    (init : (c -> more))
    (more : (a -> more)
            (d -> more)
            (r -> end))
    (end  :)))
```

```
(define-syntax automation
  (syntax-rules (: ->)
    ((_ init-state
     (state : (label -> target) ...)
     ...)
     (letrec
       ((state
         (lambda (stream)
          (cond
            ((empty? stream) #t)
            (else
              (case (first stream)
                ((label) (target (rest stream)))
                ...
                (else #f))))))
      ...)
       init-state))))
```

Power! Power! Power!

Both Ends Are Sharp

Macros Change Semantics

(Do Not Feed After Midnight)

Thank You!

(the pieces of the puzzle are waiting)

kyle.burton@gmail.com

kburton@relaynetwork.com

@kyleburton

http://relaynetwork.com

http://github.com/kyleburton/redsnake2013

More Info

http://en.wikipedia.org/wiki/Schwartzian_transform

http://c2.com/cgi/wiki?BlubParadox

http://paulgraham.com/avg.html

http://stackoverflow.com/questions/267862/what-makes-lisp-macros-so-special

http://www.gigamonkeys.com/book/

http://hipster.home.xs4all.nl/lib/scheme/gauche/define-syntax-primer.txt

Other Beautiful Macros

```
aif
if-let, when-let
with-gensyms
once-only (!)
doto
with-open
```