UC Berkeley Scheme Environment Diagram with EnvDraw

Bill Hung

August 10, 2006

Introduction

You can login to a Berkeley Unix machine through SSH, and at the command prompt, type: envdraw. A special stk (Berkeley Scheme) will start and at the prompt, type: (envdraw). A graphical interface will appear if your X11 settings are set properly. Type the scheme definition at the command prompt and the corresponding environment diagrams will appear.

Step-by-Step to draw Environment Diagram

1. Evaluate Procedures
2. Evaluate Arguments
3. Make Frame, and extend to where the second bubble of the procedure points to.
4. Evaluate Body in that Frame

EnvDraw Examples

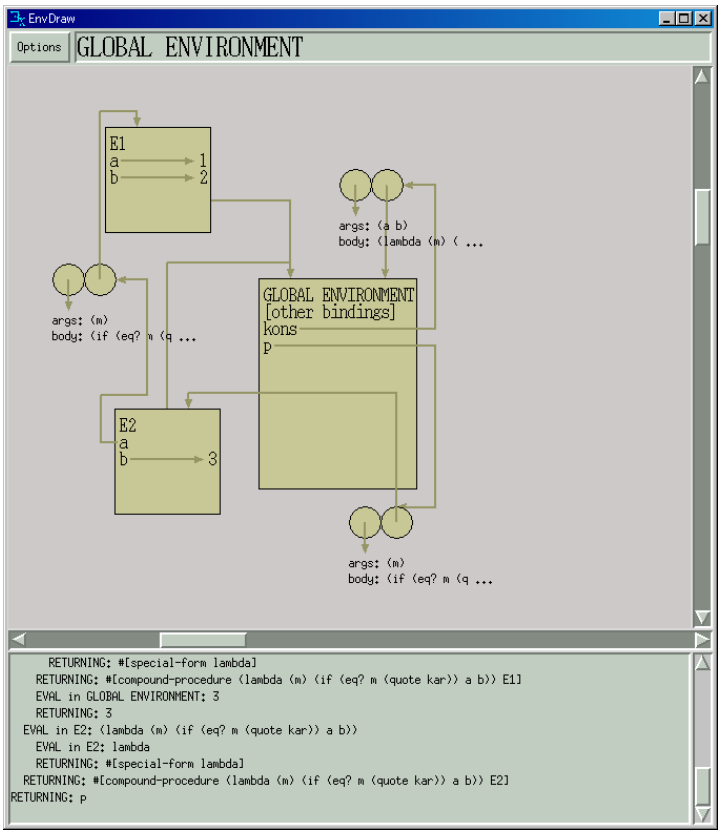
- Example 1

INPUT

```
( define ( kons a b)
   ( lambda (m)
      (if (eq? m 'kar ) a b)))
( define p ( kons ( kons 1 2) 3))
```

OUTPUT

```
STk> p
```



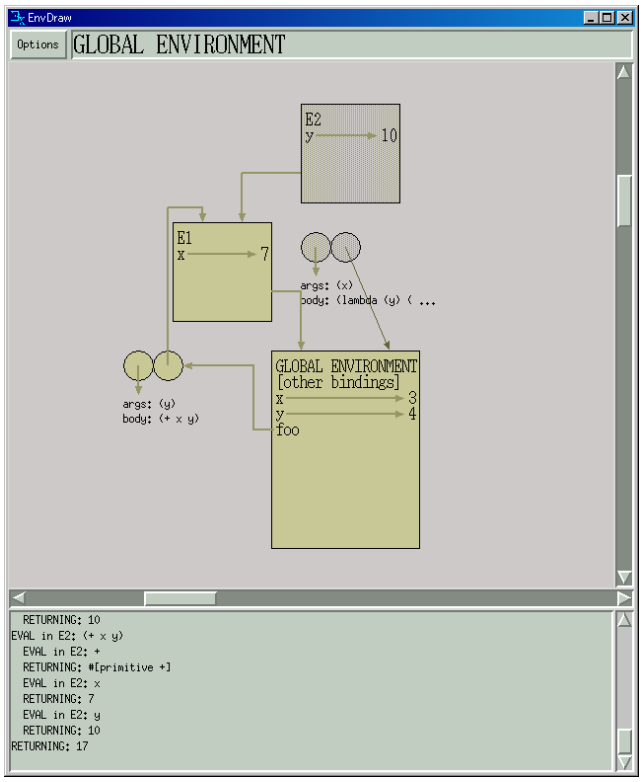
- Example 2

INPUT

```
( define x 3)
( define y 4)
( define foo
   (( lambda (x) ( lambda (y) (+ x y)))
    (+ x y )))
(foo 10)
```

OUTPUT

```
STk> x
STk> y
STk> foo
STk> 17
```



- Example 3

INPUT

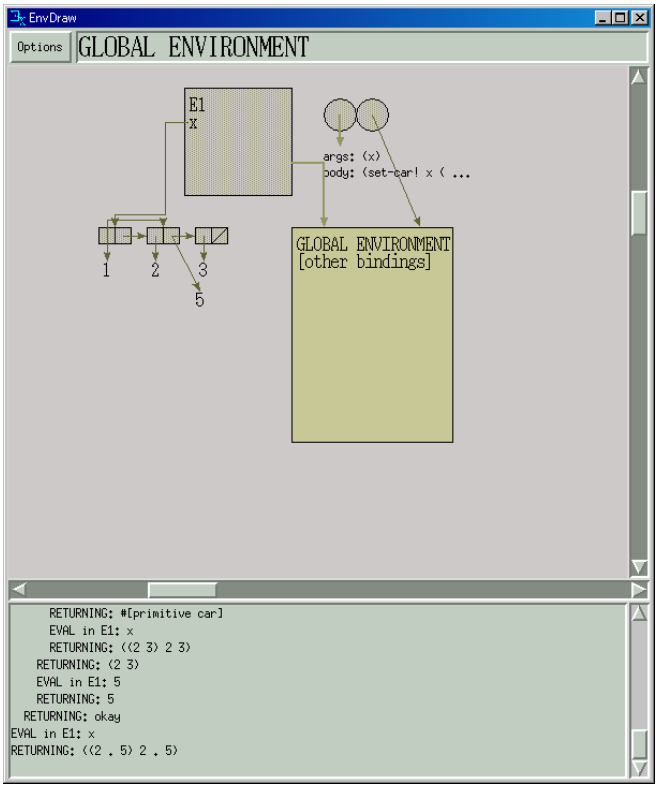
```
 (let ((x ( list 1 2 3)))
   (set-car! x (cdr x))
   (set-cdr! (car x) 5)
   x)
```

OUTPUT

```
((2 . 5) 2 . 5)
```

Note

```
(let ((<var> <exp>)) <body>)
 is equivalent to
((lambda (<var>) <body>) <exp>)
```



- Example 3

INPUT

```
(let ((x ( list 1 2 3)))
   (set-car! x ( list 'a 'b 'c))
   (set-car! ( cdar x) 'd)
   x)
```

OUTPUT

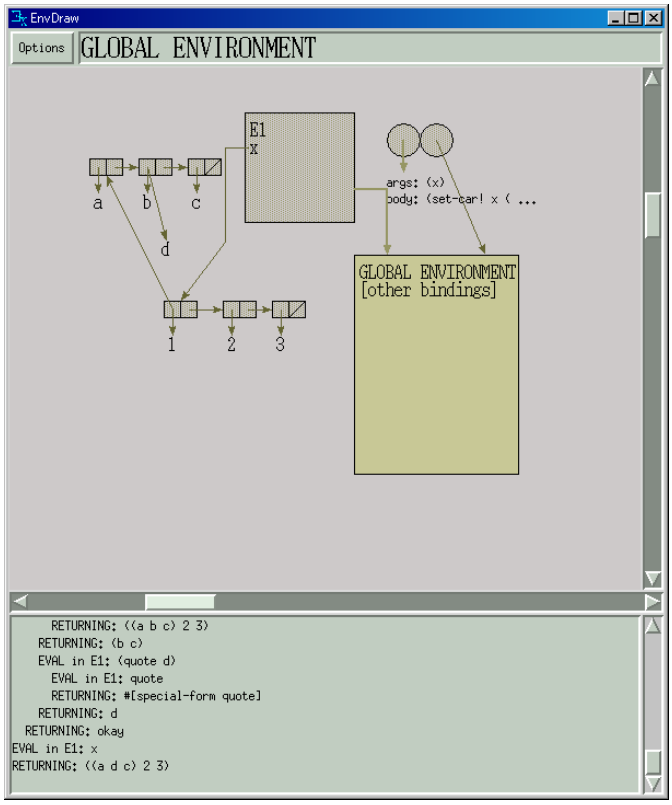
```
((a d c) 2 3)
```

Note

(let ((<var> <exp>)) <body>)

is equivalent to

((lambda (<var>) <body>) <exp>)



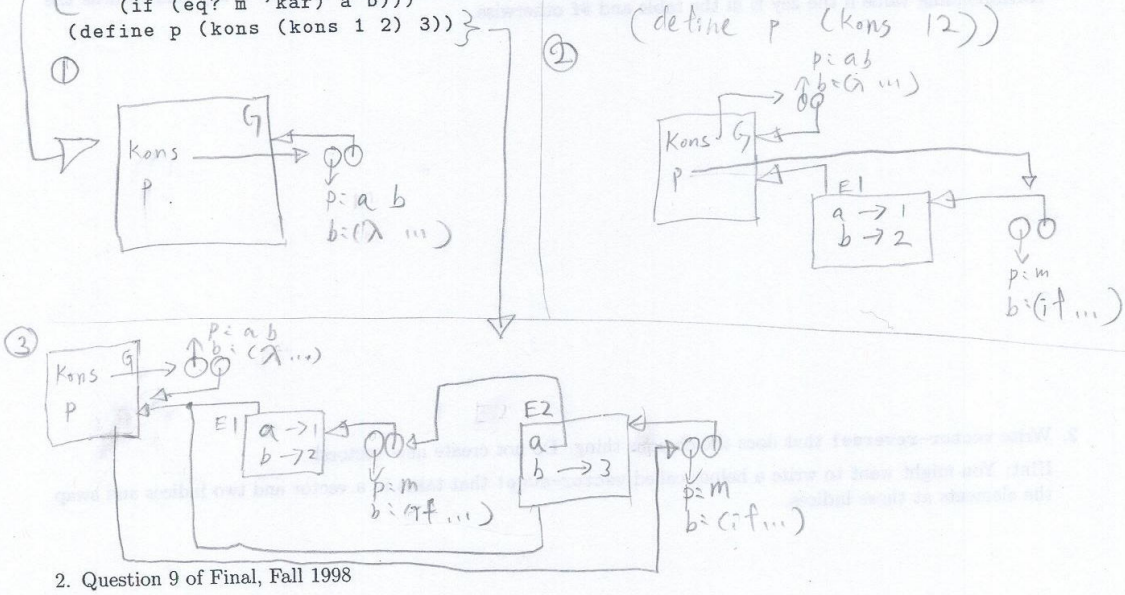
Hand Drawings

G: Global environment
p: parameter
b: body

Environment Diagram

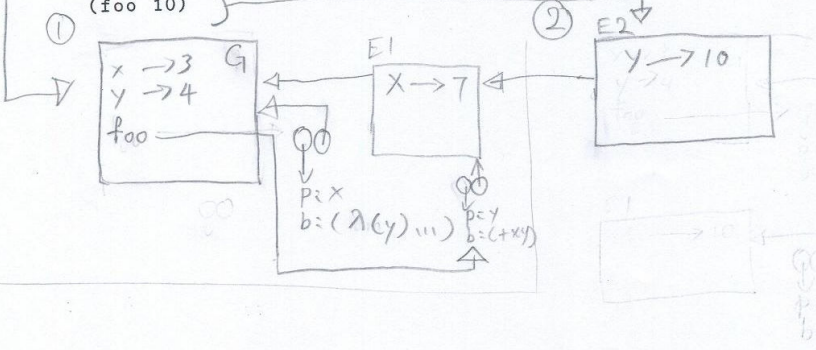
1. Question 6 of Final, Fall 1997

```
(define (kons a b)
  (lambda (m)
    (if (eq? m 'kar) a b)))
(define p (kons (kons 1 2) 3))
```

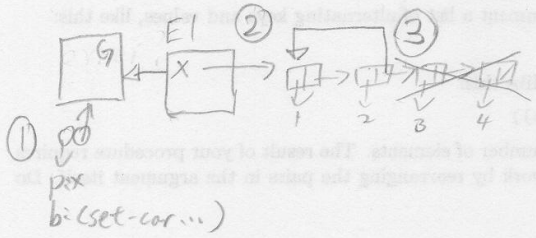


2. Question 9 of Final, Fall 1998

```
(define x 3)
(define y 4)
(define foo
    ((lambda (x) (lambda (y) (+ x y)))
     (+ x y)))
(foo 10)
```

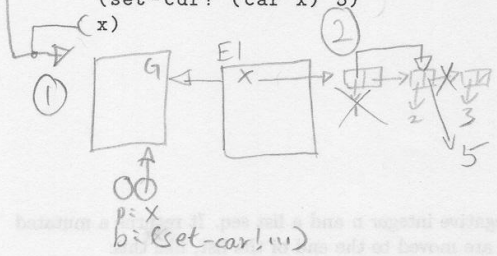


3.
```
(let ((x (list 1 2 3 4)))
     (set-car! (cddr x) x)
     x)
```



infinite loop

4.
```
(let ((x (list 1 2 3)))
     (set-car! x (cdr x))
     (set-cdr! (car x) 5)
     x)
```



5.
```
(let ((x (list 1 2 3)))
     (set-car! x (list 'a 'b 'c))
     (set-car! (cdar x) 'd)
     x)
```