# **14:43:21** evalexpr.scm **1/2**

1:

2: ;; $Id: evalexpr.scm,v 1.7 2020-01-13 14:43:13-08 - - $ 3:

4: ;;

5: ;; NAME

6: ;; eval-apply examples of evaluating expressions.

7: ;;

8:

9: (define \*functions\* (make-hash))

10: (for-each

11: (lambda (symfun) (hash-set! \*functions\* (car symfun) (cadr symfun)))

12: ‘(

13: (+ ,+)

14: (- ,-)

15: (\* ,\*)

16: (/ ,/)

17: (^ ,expt)

18: (sqrt ,sqrt) 19: 20: )) 21:

22: (define \*variables\* (make-hash))

23: (for-each

24: (lambda (varval)

25: (hash-set! \*variables\* (car varval) (cadr varval)))

26: ‘(

27: (pi ,(acos -1))

28: (e ,(exp 1))

29: (i ,(sqrt -1))

30: (one 1)

31: (zero 0) 32: )) 33:

34: (define NAN (/ 0.0 0.0)) 35:

36: (define (eval-expr expr)

37: (cond ((number? expr) (+ expr 0.0))

38: ((symbol? expr) (hash-ref \*variables\* expr NAN))

39: ((pair? expr)

40: (let ((func (hash-ref \*functions\* (car expr) NAN))

41: (opnds (map eval-expr (cdr expr))))

42: (if (null? func) NAN

43: (apply func (map eval-expr opnds))))) 44: (else NAN))) 45:

46: (define (test expr)

47: (printf "expr: ˜s˜n" expr)

48: (printf "value: ˜s˜n" (eval-expr expr)) 49: (newline)) 50:

# **14:43:21** evalexpr.scm **2/2**

51:

52: (test 3)

53: (test 1/2)

54: (test ’(+ (\* 3 4) (\* 5 6)))

55: (test ’(+ (^ e (\* i pi)) 1))

56: (test ’(/ zero zero)) 57: (test ’(/ one zero)) 58:

59: (define (dump-hash hash\_name)

60: (printf "˜s:˜n" hash\_name)

61: (hash-for-each (eval hash\_name)

62: (lambda (key value) (printf " ˜s: ˜s˜n" key value) #t)) 63: (newline)) 64:

65: (dump-hash ’\*functions\*)

66: (dump-hash ’\*variables\*) 67:

68: ;;TEST: evalexpr.scm 69:

# **14:43:22** evalexpr.scm.lis **1/1**

1: \*\*\*\*mks\*\*\*\* evalexpr.scm 2>&1

2: expr: 3 3: value: 3.0 4:

5: expr: 1/2 6: value: 0.5 7:

8: expr: (+ (\* 3 4) (\* 5 6))

9: value: 42.0 10:

11: expr: (+ (^ e (\* i pi)) 1) 12: value: 0.0+1.2246467991473532e-16i 13:

14: expr: (/ zero zero) 15: value: +nan.0 16:

17: expr: (/ one zero) 18: value: +inf.0 19:

20: \*functions\*:

21: ^: #<procedure:expt>

22: /: #<procedure:/>

23: sqrt: #<procedure:sqrt>

24: +: #<procedure:+>

25: \*: #<procedure:\*>

26: -: #<procedure:-> 27:

28: \*variables\*:

29: e: 2.718281828459045

30: i: 0+1i

31: pi: 3.141592653589793

32: zero: 0 33: one: 1 34:

35: \*\*\*\*status: 0x0000 36: