10/31/14

# **17:35:08** symbols.scm **1/2**

1:

2: ;; $Id: symbols.scm,v 1.2 2014-10-31 17:35:08-07 - - $ 3:

4: ;;

5: ;; NAME

6: ;; symbols - illustrate use of hash table for a symbol table

7: ;;

8: ;; DESCRIPTION

9: ;; Put some entries into the symbol table and then use them.

10: ;; 11:

12: ;;

13: ;; Create the symbol table and initialize it.

14: ;; 15:

16: (define \*symbol-table\* (make-hash))

17: (define (symbol-get key)

18: (hash-ref \*symbol-table\* key))

19: (define (symbol-put! key value)

20: (hash-set! \*symbol-table\* key value)) 21:

22: (for-each

23: (lambda (pair)

24: (symbol-put! (car pair) (cadr pair))) 25: ‘( 26:

27: (log10\_2 0.301029995663981195213738894724493026768189881)

28: (sqrt\_2 1.414213562373095048801688724209698078569671875)

29: (e 2.718281828459045235360287471352662497757247093)

30: (pi 3.141592653589793238462643383279502884197169399)

31: (div ,(lambda (x y) (floor (/ x y))))

32: (log10 ,(lambda (x) (/ (log x) (log 10.0))))

33: (mod ,(lambda (x y) (- x (\* (div x y) y))))

34: (quot ,(lambda (x y) (truncate (/ x y))))

35: (rem ,(lambda (x y) (- x (\* (quot x y) y))))

36: (+ ,+)

37: (^ ,expt)

38: (ceil ,ceiling)

39: (exp ,exp)

40: (floor ,floor)

41: (log ,log) 42: (sqrt ,sqrt) 43: 44: )) 45:

46: ;;

47: ;; What category of object is this?

48: ;; 49:

50: (define (what-kind value)

51: (cond ((real? value) ’real)

52: ((vector? value) ’vector)

53: ((procedure? value) ’procedure) 54: (else ’other))) 55:

56: ;;

57: ;; Main function. 58: ;;

|  |  |  |
| --- | --- | --- |
| 10/31/14  17:35:08 | symbols.scm | 2/2 |

59:

60: (define (main argvlist)

61: (symbol-put! ’n (expt 2.0 32.0))

62: (symbol-put! ’a (make-vector 10 0.0))

63: (vector-set! (symbol-get ’a) 3 (symbol-get ’pi))

64: (printf "2 ^ 16 = ˜s˜n" ((symbol-get ’^) 2.0 16.0)) 65: (printf "log 2 = ˜s˜n" ((symbol-get ’log) 2.0)) 66: (printf "log10 2 = ˜s˜n" ((symbol-get ’log10) 2.0)) 67:

68: (newline)

69: (printf "\*symbol-table\*:˜n")

70: (hash-for-each \*symbol-table\*

71: (lambda (key value)

72: (printf "˜s : ˜s = ˜s˜n" key (what-kind value) value)) 73: )) 74:

75: (main ’()) 76:

|  |  |  |
| --- | --- | --- |
| 11/04/14  16:27:01 | symbols.scm.out | 1/1 |

1: 2 ^ 16 = 65536.0

2: log 2 = 0.6931471805599453 3: log10 2 = 0.30102999566398114 4:

5: \*symbol-table\*:

6: exp : procedure = #<procedure:exp>

7: ceil : procedure = #<procedure:ceiling>

8: ^ : procedure = #<procedure:expt>

9: rem : procedure = #<procedure:...es/./symbols.scm:35:18>

10: quot : procedure = #<procedure:...es/./symbols.scm:34:18>

11: log10 : procedure = #<procedure:...es/./symbols.scm:32:18>

12: div : procedure = #<procedure:...es/./symbols.scm:31:18>

13: e : real = 2.718281828459045

14: sqrt\_2 : real = 1.4142135623730951

15: log10\_2 : real = 0.3010299956639812

16: + : procedure = #<procedure:+>

17: floor : procedure = #<procedure:floor>

18: n : real = 4294967296.0

19: a : vector = #(0.0 0.0 0.0 3.141592653589793 0.0 0.0 0.0 0.0 0.0 0.0)

20: pi : real = 3.141592653589793

21: mod : procedure = #<procedure:...es/./symbols.scm:33:18>

22: sqrt : procedure = #<procedure:sqrt>

23: log : procedure = #<procedure:log>