12 符号

1. 简介

我会在本章讲解在Lisp/Scheme程序设计语言中characteristic的数据类型——符号。

1. 有关符号的基本函数

下列都是有关符号的基本函数。

(symbol? x)

如果x是一个符号则返回#t。

(string->symbol str)

将str转换为符号。str应该都是小写的，否则地址系统可能无法正常工作。

在MIT-Scheme中，(string->symbol “Hello”)和’Hello是不同的。

(eq? (string->symbol "Hello") 'Hello)

;Value: ()

(eq? (string->symbol "Hello") (string->symbol "Hello"))

;Value: #t

(symbol->string (string->symbol "Hello"))

;Value 15: "Hello"

(symbol->string sym)

将sym转换为字符。

1. 统计文本中的单词

下面的代码是一段统计文本中单词个数的程序，这也是被经常用作演示符号的例子。这个程序使用了哈希表和关联表，这些都将在下一章中讲解。

01: ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

02: ;;; wc.scm

03: ;;; a scheme word-count program

04: ;;;

05: ;;; by T.Shido

06: ;;; on August 19, 2005

07: ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

08:

09: (define (list->symbol ls0)

10: (string->symbol (list->string (reverse! ls0))))

11:

12: (define (char-in c . ls)

13: (let loop((ls0 ls))

14: (if (null? ls0)

15: #f

16: (or (char=? c (car ls0))

17: (loop (cdr ls0))))))

18:

19: (define (read-words fname)

20: (with-input-from-file fname

21: (lambda ()

22: (let loop((w '()) (wls '()))

23: (let ((c (read-char)))

24: (cond

25: ((eof-object? c)

26: (reverse! (if (pair? w)

27: (cons (list->symbol w) wls)

28: wls)))

29: ((char-in c #\Space #\Linefeed #\Tab #\, #\. #\ #\( #\) #\= #\? #\! #\; #\:)

30: (loop '() (if (pair? w)

31: (cons (list->symbol w) wls)

32: wls)))

33: (else

34: (loop (cons (char-downcase c) w) wls))))))))

35:

36: (define (sort-by-frequency al)

37: (sort al (lambda (x y) (> (cdr x) (cdr y)))))

38:

39: (define (wc fname)

40: (let ((wh (make-eq-hash-table)))

41: (let loop((ls (read-words fname)))

42: (if (null? ls)

43: (sort-by-frequency (hash-table->alist wh))

44: (begin

45: (hash-table/put! wh (car ls) (1+ (hash-table/get wh (car ls) 0)))

46: (loop (cdr ls)))))))

(wc "opensource.txt")

⇒

((the . 208) (to . 142) (a . 104) (of . 103) (and . 83) (that . 75) (is . 73) (in . 65) (i . 64)

(you . 55) (it . 54) (they . 48) (for . 46) (what . 38) (work . 37) (but . 35) (have . 32) (on . 32)

(people . 32) (are . 30) (be . 29) (do . 29) (from . 27) (so . 26) (like . 25) (as . 25) (by . 24)

(source . 24) (not . 23) (open . 23) (can . 23) (we . 22) (was . 22) (one . 22) (it's . 22) (an . 21)

(this . 20) (about . 20) (business . 18) (working . 18) (most . 17) (there . 17) (at . 17) (with . 16)

(don't . 16) (just . 16) (their . 16) (something . 15) (than . 15) (has . 15) (if . 15) (when . 14)

(because . 14) (more . 14) (were . 13) (office . 13) (own . 13) (or . 12) (online . 12) (now . 12)

(blogging . 12) (how . 12) (employees . 11) (them . 11) (think . 11) (time . 11) (company . 11)

(lot . 11) (want . 11) (companies . 10) (could . 10) (know . 10) (get . 10) (learn . 10) (better . 10)

(some . 10) (who . 10) (even . 9) (thing . 9) (much . 9) (no . 9) (make . 9) (up . 9) (being . 9)

(money . 9) (relationship . 9) (that's . 9) (us . 9) (anyone . 8) (average . 8) (bad . 8) (same . 8)

..........)

说明：

|  |  |  |
| --- | --- | --- |
| **line** | **function** | **comment** |
| 09 | (list->symbol **ls0**) | Converting a list of characters (**ls0**) to a symbol. |
| 12 | (char-in **c**. **ls**) | Checking if a character (**c**) exists in a list (**ls**). Returning #t if it exists otherwise #f. |
| 19 | (read-words **fname**) | Reading a file named **fname** and returning a list of symbols. The function converts caps to lowers and converts a list of characters (**w**) to a symbol and adds it to a list of symbols (**wls**). |
| 36 | (sort-by-frequency **al**) | Sorting association lists (**al**) by frequency of appearance in descending order. |
| 39 | (wc **fname**) | It reads a file named **fname** and returns a sorted association list by frequency in descending order. As the function uses symbol, eq-hash-tableis applicable which uses fast eq? to compare keys (line 40). The function counts words in the list of words created by read-words and stores in a hash table (lines 44–46). It converts the hash-table to a association list and sorts it when the counting has been finished (line 43). |

1. 小节

符号是Lisp/Scheme中用于解析分析文段（例如词数统计，解析等）的一种字符式数据类型，有一些速度很快的函数适用于符号。