Próximo:Bool-Vetores, Anterior:Funções vetoriais, Acima:Vetores de matrizes de sequências [Conteúdo]

[Índice]

6.6 Tabelas de Caracteres

Uma tabela de caracteres é muito parecida com um vetor, exceto que é indexada por códigos de caracteres. Qualquer código de caractere válido, sem modificadores, pode ser usado como índice em uma tabela de caracteres. Você pode acessar os elementos de uma tabela de caracteres com `aref`e `aset`, como em qualquer array. Além disso, uma tabela de caracteres pode ter *slots extras* para armazenar dados adicionais não associados a códigos de caracteres específicos. Assim como os vetores, as tabelas de caracteres são constantes quando avaliadas e podem conter elementos de qualquer tipo.

Cada char-table tem um *subtipo* , um símbolo, que serve a dois propósitos:

- O subtipo fornece uma maneira fácil de dizer para que serve a tabela de caracteres. Por exemplo, as tabelas de exibição são tabelas de caracteres com `display-table` o subtipo e as tabelas de sintaxe são as tabelas de caracteres com `syntax-table`o subtipo. O subtipo pode ser consultado usando a função `char-table-subtype`, descrita abaixo.
- O subtipo controla o número de *slots extras* na tabela de caracteres. Este número é especificado pela `char-table-extra-slots`propriedade de símbolo do subtipo (veja Propriedades do Símbolo), cujo valor deve ser um inteiro entre 0 e 10. Se o subtipo não possui tal propriedade de símbolo, a tabela de caracteres não possui slots extras.

Uma char-table pode ter um *parent* , que é outra char-table. Se isso acontecer, sempre que a char-table especificar `nil`um caractere específico *c* , ela herdará o valor especificado no pai. Em outras palavras, retorna o valor do pai de *char-table* se o próprio *char-table* especificar . `(aref char-table c)``nil`

Uma tabela de caracteres também pode ter um *valor padrão* . Em caso afirmativo, retorna o valor padrão sempre que a char-table não especifica nenhum outro valor que não seja. `(aref char-table c)``nil`

Função: subtipo make-char-table *e inicialização opcional*

Retorna uma tabela de caracteres recém-criada, com subtipo *subtipo* (um símbolo). Cada elemento é inicializado para *init* , cujo padrão é `nil`. Você não pode alterar o subtipo de uma tabela de caracteres após a criação da tabela de caracteres.

Não há argumento para especificar o comprimento da tabela de caracteres, porque todas as tabelas de caracteres têm espaço para qualquer código de caractere válido como índice.

Se o *subtipo* tiver a `char-table-extra-slots`propriedade symbol, isso especifica o número de slots extras na tabela de caracteres. Deve ser um número inteiro entre 0 e 10; caso contrário, `make-char-table`gera um erro. Se o *subtipo* não tiver `char-table-extra-slots`propriedade de símbolo (consulte Listas de propriedades), a tabela de caracteres não terá slots extras.

Função: *objeto* char-table-p

Esta função retorna `t`se o *objeto* for uma tabela de caracteres e `nil`caso contrário.

Função: char-table-subtype *char-table*

Esta função retorna o símbolo do subtipo de *char-table* .

Não há função especial para acessar valores padrão em uma tabela de caracteres. Para fazer isso, use `char-table-range`(veja abaixo).

Função: char-table-pai *char-table*

Esta função retorna o pai de *char-table* . O pai é sempre uma `nil`ou outra tabela de caracteres.

Função: set-char-table-parent *char-table new-parent*

Esta função define o pai de *char-table* como *new-parent* .

Função: char-table-extra-slot *char-table n*

Esta função retorna o conteúdo do slot extra *n* (base zero) de *char-table* . O número de slots extras em uma tabela de caracteres é determinado por seu subtipo.

Função: set-char-table-extra-slot *char-table n valor*

Esta função armazena o *valor* no slot extra *n* (base zero) de *char-table* .

Uma tabela de caracteres pode especificar um valor de elemento para um código de caractere único; ele também pode especificar um valor para um conjunto de caracteres inteiro.

Função: char-table-range *char-table range*

Isso retorna o valor especificado em *char-table* para um intervalo de caracteres *range* . Aqui estão as possibilidades de *alcance* :

`nil`

Refere-se ao valor padrão.

Caracteres

Refere-se ao elemento para caractere *char* (supondo que *char* seja um código de caractere válido).

`(`*`from . to`*`)`

Uma célula contra refere-se a todos os caracteres no intervalo inclusivo '`[ `*`de .. a`*` ]`'.

Função: set-char-table-range *char-table range valor*

Esta função define o valor em *char-table* para um intervalo de caracteres *range* . Aqui estão as possibilidades de *alcance* :

`nil`

Refere-se ao valor padrão.

`t`

Refere-se a toda a gama de códigos de caracteres.

Caracteres

Refere-se ao elemento para caractere *char* (supondo que *char* seja um código de caractere válido).

`(`*`from . to`*`)`

Uma célula contra refere-se a todos os caracteres no intervalo inclusivo '`[ `*`de .. a`*` ]`'.

Função: map-char-table *function char-table*

Esta função chama sua *função* de argumento para cada elemento de *char-table* que tem um `nil`valor não. A chamada à *função* é com dois argumentos, uma chave e um valor. A chave é um possível argumento de *intervalo*`char-table-range` para —seja um caractere válido ou uma célula contras , especificando um intervalo de caracteres que compartilham o mesmo valor. O valor é o que retorna. (*`from . to`*)(`char-table-range` *`char-table key`*)

No geral, os pares chave-valor passados para *function* descrevem todos os valores armazenados em *char-table* .

O valor de retorno é sempre `nil`; para fazer chamadas para `map-char-table`útil, a *função* deve ter efeitos colaterais. Por exemplo, veja como examinar os elementos da tabela de sintaxe:

```
(deixe (acumulador)
   (map-char-table
    (lambda (valor chave)
      (setq acumulador
            (contras (lista
                   (se (tecla consp)
                       (lista (chave do carro) (chave cdr))
                     chave)
                   valor)
                  acumulador)))
    (tabela de sintaxe))
   acumulador)
⇒
(((2597602 4194303) (2)) ((2597523 2597601) (3))
 ... (65379 (5 . 65378)) (65378 (4 . 65379)) (65377 (1))
 ... (12 (0)) (11 (3)) (10 (12)) (9 (0)) ((0 8) (3)))
```

Próximo:Bool-Vetores, Anterior:Funções vetoriais, Acima:Vetores de matrizes de sequências [Conteúdo]

[Índice]