

Próximo: [Adicionando Variáveis Generalizadas](#), Acima: [Variáveis Generalizadas](#) [[Conteúdo](#)][[Índice](#)]

### 12.17.1 A `setf`Macro

A `setf`macro é a maneira mais básica de operar em variáveis generalizadas. O `setf`formulário é como `setq`, exceto que aceita formas de lugar arbitrárias no lado esquerdo em vez de apenas símbolos. Por exemplo, `(setf (car a) b)` configura o carro de `a` para `b`, fazendo a mesma operação que `(setcar a b)`, mas sem que você precise usar duas funções separadas para configurar e acessar esse tipo de lugar.

#### Macro: `setf` [*formulário de lugar*]...

Essa macro avalia o *formulário* e o armazena no *local*, que deve ser um formulário válido de variável generalizada. Se houver vários pares de *lugares* e *formas*, as atribuições são feitas sequencialmente, assim como com `setq`. `setf` retorna o valor da última *forma*.

Os seguintes formulários Lisp são os formulários no Emacs que funcionarão como variáveis generalizadas e, portanto, podem aparecer no argumento *placesetf* de :

- Um símbolo. Em outras palavras, `(setf x y)` é exatamente equivalente a `(setq x y)`, e `setf` em si é estritamente falando redundante, dado que `setf` existe. A maioria dos programadores continuará preferindo `setq` definir variáveis simples, porém, por razões estilísticas e históricas. A macro `(setf x y)` realmente se expande para `(setq x y)`, portanto, não há penalidade de desempenho por usá-la em código compilado.
- Uma chamada para qualquer uma das seguintes funções Lisp padrão:

```
função-símbolo aref caddr
car elt symbol-plist
caar obter valor-símbolo
cadr gethash
cdr enésimo
cdar nthcdr
```

- Uma chamada para qualquer uma das seguintes funções específicas do Emacs:

```
alist-get process-get
sentinela de processo de parâmetro de quadro
buffer de janela de parâmetro de terminal
keymap-parent window-display-table
match-data window-dedicated-p
overlay-get window-hscroll
parâmetro de janela de início de sobreposição
ponto de janela de sobreposição
início da janela do buffer de processo
valor padrão do filtro de processo
```

`setf` sinaliza um erro se você passar um formulário de *lugar* que ele não sabe lidar.

Observe que para `nthcdr`, o argumento de lista da função deve ser uma forma de *lugar* válida. Por exemplo, `(setf (nthcdr 0 foo) 7)` será definido `foo` como 7.

As macros `push` (veja [List Variables](#) ) e `pop` (veja [List Elements](#) ) podem manipular variáveis generalizadas, não apenas listas. `remove` e `return` o primeiro elemento da lista armazenado no *local* . É análogo a `car` , exceto pelo fato de ter o cuidado de avaliar todos os subformulários apenas uma vez. `insert` *x* na frente da lista armazenada no *local* . É análogo a `cons` , exceto para avaliação dos subformulários. Observe que `local` pode ser usado para inserir ou excluir em qualquer posição de uma lista. (`pop place`) (`prog1 (car place) (setf place (cdr place))`) (`push x place`) (`setf place (cons x place)`) `pushpopnthcdr`

`oc1-library` define várias extensões para variáveis generalizadas, incluindo `setf` lugares adicionais. Consulte [Variáveis Generalizadas](#) em Extensões Common Lisp .

Próximo: [Adicionando Variáveis Generalizadas](#), Acima: [Variáveis Generalizadas](#) [[Conteúdo](#)][[Índice](#)]