Próximo:Acessando variáveis, Anterior:Definindo Variáveis, Acima:Variáveis [Conteúdo][Índice]

12.6 Dicas para definir variáveis de forma robusta

Quando você define uma variável cujo valor é uma função, ou uma lista de funções, use um nome que termine em '`-função`' ou '`-funções`', respectivamente.

Existem várias outras convenções de nomes de variáveis; aqui está uma lista completa:

'…-gancho'

A variável é um gancho normal (consulte Hooks).

'…-função'

O valor é uma função.

'…-funções'

O valor é uma lista de funções.

'…-Formato'

O valor é um formulário (uma expressão).

'…-formas'

O valor é uma lista de formulários (expressões).

'…-predicado'

O valor é um predicado — uma função de um argumento que retorna não- `nil`para sucesso e `nil`falha.

'…-bandeira'

O valor é significativo apenas se for `nil`ou não. Como essas variáveis geralmente acabam adquirindo mais valores ao longo do tempo, essa convenção não é fortemente recomendada.

'…-programa'

O valor é um nome de programa.

'…-comando'

O valor é um comando shell inteiro.

'…-comuta'

O valor especifica opções para um comando.

'*prefixo* --…'

A variável é destinada ao uso interno e é definida no arquivo *`prefixo`*`.el`. (O código Emacs contribuído antes de 2018 pode seguir outras convenções, que estão sendo eliminadas.)

'…-interno'

A variável destina-se ao uso interno e é definida em código C. (O código Emacs contribuído antes de 2018 pode seguir outras convenções, que estão sendo eliminadas.)

Ao definir uma variável, sempre considere se você deve marcá-la como segura ou arriscada; consulte Variáveis Locais do Arquivo .

Ao definir e inicializar uma variável que contém um valor complicado (como um mapa de teclas com associações), é melhor colocar todo o cálculo do valor no `defvar`, assim:

```
(defvar my-mode-map
  (deixe ((mapa (make-mapa de teclas esparsas))))
    (definir mapa-chave "\Cc\Ca" 'meu-comando)
    …
    mapa)
  docstring )
```

Este método tem vários benefícios. Primeiro, se o usuário sair enquanto carrega o arquivo, a variável ainda não foi inicializada ou inicializada corretamente, nunca no meio. Se ainda não foi inicializado, recarregar o arquivo irá inicializá-lo corretamente. Segundo, recarregar o arquivo uma vez que a variável é inicializada não irá alterá-lo; isso é importante se o usuário tiver executado ganchos para alterar parte do conteúdo (como, para religar chaves). Terceiro, avaliar o `defvar`formulário com `C-M-x`irá reinicializar o mapa completamente.

Colocar tanto código no `defvar`formulário tem uma desvantagem: coloca a string de documentação longe da linha que nomeia a variável. Aqui está uma maneira segura de evitar isso:

```
(defvar my-mode-map nil
   docstring )
(a menos que meu-modo-mapa
  (deixe ((mapa (make-mapa de teclas esparsas))))
    (definir mapa-chave "\Cc\Ca" 'meu-comando)
    …
    (setq my-mode-map map)))
```

Isso tem as mesmas vantagens de colocar a inicialização dentro do `defvar`, exceto que você deve digitar `C-M-x`duas vezes, uma vez em cada formulário, se quiser reinicializar a variável.

Próximo:Acessando variáveis, Anterior:Definindo Variáveis, Acima:Variáveis [Conteúdo][Índice]