Próximo:Variáveis com valores restritos, Anterior:Variáveis Locais de Conexão, Acima:Variáveis [

Conteúdo][Índice]

12.15 Pseudônimos Variáveis

Às vezes é útil fazer duas variáveis sinônimos, de modo que ambas as variáveis sempre tenham o mesmo valor, e alterar uma também altera a outra. Sempre que você altera o nome de uma variável - seja porque percebe que seu nome antigo não foi bem escolhido ou porque seu significado mudou parcialmente - pode ser útil manter o nome antigo como um *alias* do novo para compatibilidade. Você pode fazer isso com `defvaralias`.

Função: defvaralias *new-alias base-variable & docstring opcional*

Esta função define o símbolo *new-alias* como um alias de variável para o símbolo *base-variable* . Isso significa que recuperar o valor de *new-alias* retorna o valor de *base-variable* e alterar o valor de *new-alias* altera o valor de *base-variable* . Os dois nomes de variáveis com alias sempre compartilham o mesmo valor e as mesmas ligações.

Se o argumento *docstring* for non- `nil`, ele especifica a documentação para *new-alias* ; caso contrário, o alias obtém a mesma documentação que *base-variable* tem, se houver, a menos que *base-variable* seja ele próprio um alias, caso em que *new-alias* obtém a documentação da variável no final da cadeia de aliases.

Esta função retorna *a variável base* .

Os aliases de variáveis são convenientes para substituir um nome antigo de uma variável por um novo nome. `make-obsolete-variable`declara que o nome antigo é obsoleto e, portanto, pode ser removido em algum momento no futuro.

Função: variável de criação obsoleta *nome-obsoleto nome-atual quando &tipo de acesso opcional*

Esta função faz com que o compilador de bytes avise que a variável *obsolete-name* está obsoleta. Se *current-name* for um símbolo, é o novo nome da variável; em seguida, a mensagem de aviso diz para usar *nome atual em* vez de nome *obsoleto* . Se *nome-corrente* for uma string, esta é a mensagem e não há variável de substituição. *when* deve ser uma string indicando quando a variável se tornou obsoleta pela primeira vez (geralmente uma string de número de versão).

O argumento opcional *access-type* , se não for `nil`, deve especificar o tipo de acesso que acionará os avisos de obsolescência; pode ser `get`ou `set`.

Você pode tornar duas variáveis sinônimos e declarar uma obsoleta ao mesmo tempo usando a macro `define-obsolete-variable-alias`.

Macro: define-obsolete-variable-alias *nome-obsoleto nome-atual &opcional quando docstring*

Essa macro marca a variável *obsolete-name* como obsoleta e também a torna um alias para a variável *current-name* . É equivalente ao seguinte:

```
(defvaralias nome-obsoleto nome  -corrente  docstring )
(criar variável obsoleta nome-obsoleto nome  -atual  quando )
```

Função: variável de *variável* indireta

Esta função retorna a variável no final da cadeia de aliases da *variável* . Se a *variável* não for um símbolo ou se *a variável* não for definida como um alias, a função retornará a *variável* .

Esta função sinaliza um `cyclic-variable-indirection`erro se houver um loop na cadeia de símbolos.

```
(defvaralias 'foo' bar)
(variável indireta 'foo)
     ⇒ barra
(barra de variável indireta)
     ⇒ barra
(configurar barra 2)
Barra
     ⇒ 2
foo
     ⇒ 2
(setq foo 0)
Barra
     ⇒ 0
foo
     ⇒ 0
```

Próximo:Variáveis com valores restritos, Anterior:Variáveis Locais de Conexão, Acima:Variáveis [Conteúdo][Índice]