

Próximo:[Criando Buffer-Local](#), Acima:[Variáveis locais de buffer](#) [Conteúdo][Índice]

### 12.11.1 Introdução às variáveis locais de buffer

Uma variável local de buffer tem uma ligação local de buffer associada a um buffer específico. A ligação está em vigor quando esse buffer é atual; caso contrário, não está em vigor. Se você definir a variável enquanto uma associação de buffer-local estiver em vigor, o novo valor entrará nessa associação, de modo que suas outras associações não sejam alteradas. Isso significa que a alteração é visível apenas no buffer em que você a fez.

A ligação comum da variável, que não está associada a nenhum buffer específico, é chamada de *ligação padrão*. Na maioria dos casos, esta é a ligação global.

Uma variável pode ter ligações de buffer-local em alguns buffers, mas não em outros buffers. A associação padrão é compartilhada por todos os buffers que não possuem suas próprias associações para a variável. (Isso inclui todos os buffers recém-criados.) Se você definir a variável em um buffer que não tenha uma associação de buffer-local para ela, isso definirá a associação padrão, de modo que o novo valor seja visível em todos os buffers que veem o padrão vinculativo.

O uso mais comum de ligações de buffer-local é para modos principais para alterar variáveis que controlam o comportamento dos comandos. Por exemplo, o modo C e o modo Lisp definem a variável `paragraph-start` para especificar que apenas linhas em branco separam os parágrafos. Eles fazem isso tornando a variável buffer-local no buffer que está sendo colocado no modo C ou no modo Lisp e, em seguida, configurando-o para o novo valor para esse modo. Consulte [Modos principais](#).

A maneira usual de fazer uma ligação de buffer-local é com `make-local-variable`, que é o que os comandos de modo principal normalmente usam. Isso afeta apenas o buffer atual; todos os outros buffers (incluindo aqueles ainda a serem criados) continuarão compartilhando o valor padrão, a menos que recebam explicitamente suas próprias ligações de buffer-local.

Uma operação mais poderosa é marcar a variável *automaticamente como buffer-local* chamando `make-variable-buffer-local`. Você pode pensar nisso como tornar a variável local em todos os buffers, mesmo aqueles que ainda não foram criados. Mais precisamente, o efeito é que definir a variável automaticamente torna a variável local para o buffer atual, se ainda não o for. Todos os buffers começam compartilhando o valor padrão da variável como de costume, mas definir a variável cria uma ligação local de buffer para o buffer atual. O novo valor é armazenado na ligação local do buffer, deixando a ligação padrão intocada. Isso significa que o valor padrão não pode ser alterado `setq` em nenhum buffer; a única maneira de alterá-lo é com `setq-default`.

**Aviso:** quando uma variável tem associações de buffer-local em um ou mais buffers, `let` reassocia a associação que está em vigor no momento. Por exemplo, se o buffer atual tiver um valor local de buffer, `let` revincule-o temporariamente. Se nenhuma ligação local de buffer estiver em vigor, `let` revinculará o valor padrão. Se dentro do `let` você mudar para um buffer atual diferente no qual uma ligação diferente está em vigor, você não verá `let` mais a ligação. E se você sair `let` enquanto ainda estiver no outro buffer, não verá a desvinculação ocorrer (embora ocorra corretamente). Segue um exemplo para ilustrar:

```
(setq foo 'g)
(set-buffer "a")
(fazer variável local 'foo)
```

```
(setq foo 'a)
(deixe ((foo 'temp))
  ;; foo ⇒ 'temp; deixe a ligação no buffer 'uma'
  (definir buffer "b")
  ;; foo ⇒ 'g; o valor global já que foo não é local em 'b'
  corpo ...)
foo ⇒ 'g; sair restaurou o valor local no buffer 'uma',
      ; mas não vemos isso no buffer 'b'
(set-buffer "a"); verifique se o valor local foi restaurado
fo ⇒ 'a'
```

Observe que as referências foono *corpo* acessam a ligação local do buffer de buffer 'b'.

Quando um arquivo especifica valores de variáveis locais, eles se tornam valores de buffer local quando você visita o arquivo. Veja [Variáveis de Arquivo](#) no Manual do GNU Emacs .

Uma variável de buffer-local não pode se tornar terminal-local (veja [Múltiplos Terminais](#) ).

Próximo:[Criando Buffer-Local](#), Acima:[Variáveis locais de buffer](#) [Conteúdo][Índice]