

Próximo:[Limpezas](#), Anterior:[Exemplos de captura](#), Acima:[Saídas não locais](#) [[Conteúdo](#)][[Índice](#)]

### 11.7.3 Erros

Quando o Emacs Lisp tenta avaliar um formulário que, por algum motivo, não pode ser avaliado, ele *sinaliza* um *erro* .

Quando um erro é sinalizado, a reação padrão do Emacs é imprimir uma mensagem de erro e encerrar a execução do comando atual. Esta é a coisa certa a fazer na maioria dos casos, como se você digitar C - fno final do buffer.

Em programas complicados, o término simples pode não ser o que você deseja. Por exemplo, o programa pode ter feito alterações temporárias nas estruturas de dados ou criado buffers temporários que devem ser excluídos antes que o programa seja concluído. Nesses casos, você usaria `unwind-protect` para estabelecer *expressões de limpeza* a serem avaliadas em caso de erro. (Consulte [Limpezas](#) .)

Ocasionalmente, você pode desejar que o programa continue a execução apesar de um erro em uma sub-rotina. Nesses casos, você usaria `condition-case` para estabelecer *manipuladores de erros* para recuperar o controle em caso de erro.

Resista à tentação de usar o tratamento de erros para transferir o controle de uma parte do programa para outra; usar `catch` `throw` em vez disso. Veja [Pegar e jogar](#) .

- |  |  |
|--|--|
| • <a href="#">Erros de sinalização</a>   | Como relatar um erro.                                    |
| • <a href="#">Processamento de Erros</a> | O que o Emacs faz quando você relata um erro.            |
| • <a href="#">Manipulação de Erros</a>   | Como você pode interceptar erros e continuar a execução. |
| • <a href="#">Símbolos de erro</a>       | Como os erros são classificados para capturá-los.        |