Próximo:Manipulando Erros, Anterior:Erros de sinalização, Acima:Erros [Conteúdo][Índice]

11.7.3.2 Como o Emacs processa erros

Quando um erro é sinalizado, `signal`procura um *manipulador* ativo para o erro. Um manipulador é uma sequência de expressões Lisp designadas para serem executadas se ocorrer um erro em parte do programa Lisp. Se o erro tiver um manipulador aplicável, o manipulador será executado e o controle será retomado após o manipulador. O handler executa no ambiente do `condition-case`que o estabeleceu; todas as funções chamadas dentro que `condition-case` já foram encerradas e o manipulador não pode retornar a elas.

Se não houver um manipulador aplicável para o erro, ele encerra o comando atual e retorna o controle ao loop de comando do editor. (O loop de comando tem um manipulador implícito para todos os tipos de erros.) O manipulador do loop de comando usa o símbolo de erro e os dados associados para imprimir uma mensagem de erro. Você pode usar a variável `command-error-function`para controlar como isso é feito:

Variável: comando-erro-função

> Essa variável, se não- `nil`, especifica uma função a ser usada para tratar erros que retornam o controle ao loop de comando do Emacs. A função deve receber três argumentos: *data* , uma lista da mesma forma que `condition-case`vincularia sua variável; *context* , uma string descrevendo a situação em que o erro ocorreu, ou (mais frequentemente) `nil`; e *caller* , a função Lisp que chamou a primitiva que sinalizou o erro.

Um erro que não tenha nenhum manipulador explícito pode chamar o depurador Lisp. O depurador é habilitado se a variável `debug-on-error`(consulte Error Debugging) não for `nil`. Ao contrário dos manipuladores de erro, o depurador é executado no ambiente do erro, para que você possa examinar os valores das variáveis exatamente como estavam no momento do erro.

Próximo:Manipulando Erros, Anterior:Erros de sinalização, Acima:Erros [Conteúdo][Índice]