Próximo:Carregar sufixos, Acima:Carregando [Conteúdo][Índice]

16.1 Como os programas são carregados

O Emacs Lisp possui várias interfaces para carregamento. Por exemplo, `autoload`cria um objeto de espaço reservado para uma função definida em um arquivo; tentar chamar a função de carregamento automático carrega o arquivo para obter a definição real da função (consulte Autoload). `require`carrega um arquivo se ainda não estiver carregado (consulte Recursos nomeados). Em última análise, todas essas facilidades chamam a `load`função para fazer o trabalho.

Função: carregar o *nome do arquivo &opcional ausente-ok nomessage nosuffix must-suffix*

Esta função encontra e abre um arquivo de código Lisp, avalia todos os formulários nele e fecha o arquivo.

Para encontrar o arquivo, `load`primeiro procure um arquivo chamado *`nome do arquivo`* `.elc`, ou seja, para um arquivo cujo nome é *filename* com a extensão '`.elc`' anexado. Se esse arquivo existir, ele será carregado. Se não houver nenhum arquivo com esse nome, `load`procura um arquivo chamado*`nome do arquivo`* `.el`. Se esse arquivo existir, ele será carregado. Se o Emacs foi compilado com suporte para módulos dinâmicos (veja Módulos Dinâmicos), `load`em seguida procura por um arquivo chamado*`nome do arquivo . ramal`*, em que *ext* é uma extensão de nome de arquivo dependente do sistema de bibliotecas compartilhadas. Finalmente, se nenhum desses nomes for encontrado, `load`procura um arquivo chamado *filename* sem nada anexado e carrega-o se existir. (A `load`função não é inteligente em olhar para *filename* . No caso perverso de um arquivo chamado`foo.el.el`, avaliação de `(load "foo.el")`vai de fato encontrá-lo.)

Se o modo de compactação automática estiver ativado, como é por padrão, se `load`não encontrar um arquivo, ele procurará uma versão compactada do arquivo antes de tentar outros nomes de arquivo. Ele descompacta e carrega se existir. Ele procura por versões compactadas anexando cada um dos sufixos `jka-compr-load-suffixes`ao nome do arquivo. O valor desta variável deve ser uma lista de strings. Seu valor padrão é `(".gz")`.

Se o argumento opcional *nosuffix* for non- `nil`, `load`não tentará os sufixos '`.elc`' e '`.el`'. Nesse caso, você deve especificar o nome de arquivo preciso que deseja, exceto que, se o modo de compactação automática estiver ativado, `load`ainda será usado `jka-compr-load-suffixes`para localizar versões compactadas. Ao especificar o nome do arquivo preciso e usar `t`para *nosuffix* , você pode evitar nomes de arquivo como`foo.el.el`de ser julgado.

Se o argumento opcional *must-suffix* for non- `nil`, então `load`insiste que o nome do arquivo usado deve terminar em '`.el`' ou '`.elc`' (possivelmente estendido com um sufixo de compactação) ou a extensão de biblioteca compartilhada, a menos que contenha um nome de diretório explícito.

Se a opção `load-prefer-newer`for non- `nil`, ao pesquisar sufixos, `load`selecione qualquer versão de um arquivo ('`.elc`', '`.el`', etc.) foi modificado mais recentemente.

Se *filename* for um nome de arquivo relativo, como`foo`ou `baz/foo.bar`, `load`procura o arquivo usando a variável `load-path`. Ele anexa o nome do *arquivo* a cada um dos diretórios listados em `load-path`e carrega o primeiro arquivo que encontrar cujo nome corresponda. O diretório padrão atual é tentado apenas se for especificado em `load-path`, onde `nil`representa o diretório padrão. `load`tenta todos os três sufixos possíveis no primeiro diretório em `load-path`, depois todos os três sufixos no segundo diretório e assim por diante. Consulte Pesquisa de biblioteca .

Qualquer que seja o nome sob o qual o arquivo seja encontrado e o diretório onde o Emacs o encontrou, o Emacs define o valor da variável `load-file-name`para o nome desse arquivo.

Se você receber um aviso de que`foo.elc`É mais velho que`foo.el`, significa que você deve considerar recompilar`foo.el`. Consulte Compilação de bytes .

Ao carregar um arquivo fonte (não compilado), `load`executa a tradução do conjunto de caracteres exatamente como o Emacs faria ao visitar o arquivo. Consulte Sistemas de Codificação .

Ao carregar um arquivo não compilado, o Emacs tenta expandir todas as macros que o arquivo contém (consulte Macros). Referimo-nos a isso como *expansão macro ansiosa* . Fazer isso (em vez de adiar a expansão até que o código relevante seja executado) pode acelerar significativamente a execução do código não compilado. Às vezes, essa macro expansão não pode ser feita, devido a uma dependência cíclica. No exemplo mais simples disso, o arquivo que você está carregando refere-se a uma macro definida em outro arquivo e esse arquivo, por sua vez, requer o arquivo que você está carregando. Isso geralmente é inofensivo. O Emacs imprime um aviso ('`A macro-expansão ansiosa foi ignorada devido ao ciclo…`') dando detalhes do problema, mas ainda carrega o arquivo, apenas deixando a macro sem expansão por enquanto. Você pode querer reestruturar seu código para que isso não aconteça. Carregar um arquivo compilado não causa macroexpansão, pois isso já deveria ter acontecido durante a compilação. Consulte Compilando Macros .

Mensagens como '`Carregando fo...`' e '`Carregando foo...concluído`' aparecem na área de eco durante o carregamento, a menos que *nomessage* não seja `nil`.

Quaisquer erros não tratados durante o carregamento de um arquivo encerram o carregamento. Se o carregamento foi feito por causa de `autoload`, quaisquer definições de função feitas durante o carregamento são desfeitas.

Se `load`não encontrar o arquivo para carregar, normalmente ele sinaliza um `file-error`(com '`Não é possível abrir o nome do arquivo de` *`carregamento`*'). Mas se *missing-ok* for non- `nil`, `load`apenas retornará `nil`.

Você pode usar a variável `load-read-function`para especificar uma função para `load`usar em vez de `read`ler expressões. Veja abaixo.

`load`retorna `t`se o arquivo for carregado com sucesso.

Comando: load-file nome do *arquivo*

Este comando carrega o arquivo *filename* . Se *filename* for um nome de arquivo relativo, o diretório padrão atual será assumido. Este comando não usa `load-path`e não acrescenta sufixos. No entanto, ele procura versões compactadas (se o modo de compactação automática estiver ativado). Use este comando se desejar especificar precisamente o nome do arquivo a ser carregado.

Comando: load-library *library*

Este comando carrega a biblioteca chamada *library* . É equivalente a `load`, exceto pela maneira como lê seu argumento interativamente. Veja Bibliotecas Lisp no Manual do GNU Emacs .

Variável: carregamento em andamento

Esta variável é non- `nil`se o Emacs está no processo de carregamento de um arquivo, e `nil`caso contrário.

Variável: load-file-name

Quando o Emacs está carregando um arquivo, o valor desta variável é o nome desse arquivo, como o Emacs o encontrou durante a pesquisa descrita anteriormente nesta seção.

Variável: função de leitura de carga

Essa variável especifica uma função de leitura de expressão alternativa para `load`e `eval-region`a ser usada em vez de `read`. A função deve aceitar um argumento, assim como `read`faz.

Por padrão, o valor desta variável é `read`. Consulte Funções de entrada .

Em vez de usar essa variável, é mais limpo usar outro recurso mais novo: passar a função como o argumento da *função de leitura*`eval-region` para . Veja Eval .

Para obter informações sobre como `load`é usado na construção do Emacs, consulte Construindo o Emacs .

Próximo:Carregar sufixos, Acima:Carregando [Conteúdo][Índice]