Próximo:Carregamento repetido, Anterior:Carregando não ASCII, Acima:Carregando [Conteúdo][Índice ]

## 16.5 Carregamento Automático

O recurso de *autoload* permite registrar a existência de uma função ou macro, mas adia o carregamento do arquivo que a define. A primeira chamada para a função carrega automaticamente a biblioteca apropriada, para instalar a definição real e outro código associado, então executa a definição real como se ela tivesse sido carregada o tempo todo. O carregamento automático também pode ser acionado consultando a documentação da função ou macro (consulte Noções básicas de documentação ) e o preenchimento de nomes de variáveis e funções (consulte Carregamento automático por prefixo abaixo).



Existem duas maneiras de configurar uma função autocarregada: chamando `autoload`, e escrevendo um comentário “mágico” na fonte antes da definição real. `autoload`é a primitiva de baixo nível para carregamento automático; qualquer programa Lisp pode chamar `autoload`a qualquer momento. Comentários mágicos são a maneira mais conveniente de fazer uma função autoload, para pacotes instalados junto com o Emacs. Esses comentários não fazem nada por conta própria, mas servem como um guia para o comando `update-file-autoloads`, que constrói chamadas para `autoload` e organiza para executá-las quando o Emacs é compilado.

**Função: nome do arquivo da função de carregamento automático *e tipo interativo opcional docstring***

Esta função define a função (ou macro) nomeada *função* para carregar automaticamente de *filename* . A string *filename* especifica o arquivo a ser carregado para obter a definição real da *função* .

Se *filename* não contiver um nome de diretório, nem o sufixo `.elou` `.elc`, essa função insiste em adicionar um desses sufixos e não será carregada de um arquivo cujo nome seja apenas *filename* sem sufixo adicionado. (A variável `load-suffixes`especifica os sufixos exatos necessários.)

O argumento *docstring* é a string de documentação para a função. Especificar a string de documentação na chamada para `autoload`possibilita ver a documentação sem carregar a definição real da função. Normalmente, isso deve ser idêntico à string de documentação na própria definição da função. Se não for, a string de documentação da definição da função entrará em vigor quando for carregada.

Se *interativo* for non- `nil`, isso diz que a *função* pode ser chamada interativamente. Isso permite a conclusão no `M-x`trabalho sem carregar a definição real da *função* . A especificação interativa completa não é fornecida aqui; não é necessário a menos que o usuário realmente chame *function* , e quando isso acontece, é hora de carregar a definição real.

Você pode carregar automaticamente macros e mapas de teclas, bem como funções comuns. Especifique o *tipo* como `macro`se a *função* fosse realmente uma macro. Especifique o *tipo* como `keymap`se a *função* fosse realmente um mapa de teclas. Várias partes do Emacs precisam conhecer essas informações sem carregar a definição real.

Um mapa de teclado carregado automaticamente é carregado automaticamente durante a pesquisa de chave quando a ligação de uma chave de prefixo é a *função* de símbolo . O carregamento automático não ocorre para outros tipos de acesso ao mapa de teclas. Em particular, isso não acontece quando um programa Lisp obtém o mapa de teclas do valor de uma variável e chama `define-key`; nem mesmo se o nome da variável for a mesma *função* de símbolo .

Se *a função* já tiver uma definição de função não void que não seja um objeto de carregamento automático, essa função não fará nada e retornará `nil`. Caso contrário, ele constrói um objeto de carregamento automático (consulte Tipo de carregamento automático ) e o armazena como a definição de função para *function* . O objeto autoload tem este formato:

```
(autoload filename  docstring  tipo interativo  )
```

Por exemplo,

```
(função-símbolo 'run-prolog)
     ⇒ (autoload "prolog" 169681 t nil)
```

Neste caso, `"prolog"`é o nome do arquivo a ser carregado, 169681 refere-se à string de documentação no `emacs/etc/DOC`file (consulte Documentação Básica ), `t`significa que a função é interativa e `nil`que não é uma macro ou um mapa de teclas.

**Função: *objeto* autoloadp**

Esta função retorna objeto não -`nil` se é um objeto de carregamento automático. Por exemplo, para verificar se está definido como uma função carregada automaticamente, avalie `run-prolog`

```
(autoloadp (função de símbolo 'run-prolog))
```

O arquivo carregado automaticamente geralmente contém outras definições e pode exigir ou fornecer um ou mais recursos. Se o arquivo não estiver completamente carregado (devido a um erro na avaliação de seu conteúdo), quaisquer definições de função ou `provide`chamadas ocorridas durante o carregamento serão desfeitas. Isso é para garantir que a próxima tentativa de chamar qualquer função de carregamento automático desse arquivo tente novamente carregar o arquivo. Se não for por isso, algumas das funções no arquivo podem ser definidas pelo carregamento abortado, mas não funcionam corretamente pela falta de certas sub-rotinas não carregadas com sucesso porque vêm mais tarde no arquivo.

Se o arquivo carregado automaticamente não definir a função ou macro Lisp desejada, um erro será sinalizado com data . `"Autoloading failed to define function` *`function-name`*"

Um comentário mágico de carregamento automático (geralmente chamado de *cookie de carregamento automático* ) consiste em '`;;;###carregamento automático`', em uma linha por si só, logo antes da definição real da função em seu arquivo de origem autocarregável. O comando `M-x update-file-autoloads` grava uma `autoload`chamada correspondente em`loaddefs.el`. (A string que serve como cookie de carregamento automático e o nome do arquivo gerado por `update-file-autoloads`podem ser alterados dos padrões acima, veja abaixo.) Construindo cargas do Emacs`loaddefs.el`e assim chama `autoload`. `M-x update-directory-autoloads`é ainda mais poderoso; ele atualiza os carregamentos automáticos de todos os arquivos no diretório atual.

O mesmo comentário mágico pode copiar qualquer tipo de formulário em `loaddefs.el`. O formulário após o comentário mágico é copiado literalmente, *exceto* se for um dos formulários que o recurso de carregamento automático manipula especialmente (por exemplo, por conversão em uma `autoload`chamada). Os formulários que não são copiados literalmente são os seguintes:

**Definições para funções ou objetos semelhantes a funções:**

`defun` `defmacro`; também `cl-defun` `cl-defmacro`(veja Argument Lists in Common Lisp Extensions ) e `define-overloadable-function`(veja o comentário em `mode-local.el`).

**Definições para modos maiores ou menores:**

`define-minor-mode`, `define-globalized-minor-mode`, `define-generic-mode`, `define-derived-mode`, `easy-mmode-define-minor-mode`, `easy-mmode-define-global-mode`, `define-compilation-modee` `define-global-minor-mode`.

**Outros tipos de definição:**

`defcustom`, `defgroup`, `defclass` ( consulte EIEIO em EIEIO ) e `define-skeleton` (consulte Autotipagem em Autotipagem ).

Você também pode usar um comentário mágico para executar um formulário em tempo de compilação *sem* executá-lo quando o próprio arquivo for carregado. Para fazer isso, escreva o formulário *na mesma linha* do comentário mágico. Como está em um comentário, não faz nada quando você carrega o arquivo de origem; mas `M-x update-file-autoloads`copia para`loaddefs.el`, onde é executado durante a construção do Emacs.

O exemplo a seguir mostra como `doctor`está preparado para o carregamento automático com um comentário mágico:

```
;;;###carregamento automático
(defun doutor ()
  "Mude para *médico* buffer e comece a dar psicoterapia."
  (interativo)
  (mudar para buffer "*doutor*")
  (modo médico))
```

Aqui está o que isso produz em`loaddefs.el`:

```
(autoload 'doutor "doutor" "\
Mude para o tampão *médico* e comece a dar psicoterapia.

\(fn)" t nulo)
```

A barra invertida e a nova linha imediatamente após as aspas duplas são uma convenção usada apenas nos arquivos Lisp não compilados pré-carregados, como `loaddefs.el`; eles dizem `make-docfile`para colocar a string de documentação no`etc/DOC`Arquivo. Consulte Construindo o Emacs . Veja também o comentário em`lib-src/make-docfile.c`. '`(fn)`' na parte de uso da string de documentação é substituído pelo nome da função quando as várias funções de ajuda (consulte Funções de ajuda ) o exibem.

Se você escrever uma definição de função com uma macro incomum que não seja um dos métodos de definição de função conhecidos e reconhecidos, o uso de um comentário de carregamento automático

mágico comum copiaria toda a definição em `loaddefs.el`. Isso não é desejável. `autoload`Você pode colocar a chamada desejada `loaddefs.el`escrevendo isto:

```
;;;###autoload (autoload 'foo "meuarquivo")
(mydefunmacro foo
  ...)
```

Você pode usar uma string não padrão como o cookie de carregamento automático e ter as chamadas de carregamento automático correspondentes gravadas em um arquivo cujo nome é diferente do padrão`loaddefs.el`. O Emacs fornece duas variáveis para controlar isso:

**Variável: gerar-autoload-cookie**

O valor desta variável deve ser uma string cuja sintaxe seja um comentário Lisp. `M-x update-file-autoloads`copia o formulário Lisp que segue o cookie no arquivo de carregamento automático que ele gera. O valor padrão dessa variável é `";;;###autoload"`.

**Variável: arquivo de carregamento automático gerado**

O valor desta variável nomeia um arquivo Emacs Lisp onde as chamadas de carregamento automático devem ir. o valor padrão é`loaddefs.el`, mas você pode substituir isso, por exemplo, na seção de variáveis locais de um `.el`arquivo (consulte Variáveis Locais de Arquivo ). Presume-se que o arquivo de carregamento automático contenha um trailer começando com um caractere de alimentação de formulário.

A seguinte função pode ser usada para carregar explicitamente a biblioteca especificada por um objeto de carregamento automático:

**Função: autoload-do-load *autoload &opcional name macro-only***

Esta função realiza o carregamento especificado por *autoload* , que deve ser um objeto autoload. O argumento opcional *name* , se não `nil-` , deve ser um símbolo cujo valor de função seja *autoload* ; nesse caso, o valor de retorno dessa função é o novo valor da função do símbolo. Se o valor do argumento opcional *macro-only* for `macro`, esta função evita carregar uma função, apenas uma macro.

Próximo:Carregamento repetido, Anterior:Carregando não ASCII, Acima:Carregando [Conteúdo][Índice

]