Package Management

Move Operation	Keystroke	Function	Notes
Lisp Package Management	The list-package command opens a buffer listing all packages available from the sites identified by the URLs listed in the ‘*package-archives*’. Once the information has been retrieved from the sites, the packages are listed one per line with name, version, state and description. Then you can use one of the single keys listed below to operate on the list. For example, you can select a set of packages to install by marking them with `i` and then installed them with `x`.		
List Packages	• `<f11> SPC l l p` • `<f12> l p`	(**list-packages** &optional NO-FETCH)	List packages available in from used package managers in a *Packages* buffer, showing the package name, its package manager site name, status and description. Clicking (or pressing enter) on the name opens a buffer with the description of the package. ☛ The `<f12> l p` binding is available for buffers in Emacs-Lisp mode.
	`(`	(**package-menu-toggle-hiding**)	(-toggle-hidden : Toggle visibility of obsolete available packages.
	• `?` • `<RET>`	(**package-menu-describe-package** &optional BUTTON)	Describe the current package. • If optional arg BUTTON is non-nil, describe its associated package. • The description buffer also contains buttons to install the package.
	`d`	(**package-menu-mark-delete** &optional NUM)	delete : Mark an (installed) package for deletion and move to the next line.
	`f`	(**package-menu-filter** KEYWORD)	filter : Filter the *Packages* buffer. • Show only those items that relate to the specified KEYWORD. • KEYWORD can be a string or a list of strings. If it is a list, a package will be displayed if it matches any of the keywords. Interactively, it is a list of strings separated by commas. • To restore the full package list, type ‘q’.
	`g`	(**revert-buffer** &optional IGNORE-AUTO NOCONFIRM PRESERVE-MODES)	redisplay : Redisplay/refresh information.
	`h`	(**package-menu-quick-help**)	help : Show short key binding help for ‘package-menu-mode’. • The full list of keys can be viewed with C-h m.
	`H`	(**package-menu-hide-package**)	hide-package : Hide a package under point. • If optional arg BUTTON is non-nil, describe its associated package.
	`i`	(**package-menu-mark-install** &optional NUM)	install : Mark a package for installation and move to the next line.
	`n`	(**next-line** &optional ARG TRY-VSCROLL)	next : Move to next line
	`p`	(**previous-line** &optional ARG TRY-VSCROLL)	previous : Move to previous line
	`r`	(**package-menu-refresh**)	refresh-content : Download the Emacs Lisp package archive. • This fetches the contents of each archive specified in ‘package-archives’, and then refreshes the package menu.
	• `u` • `<DEL>`	(**package-menu-mark-unmark** &optional NUM)	unmark : Clear any marks on a package and move to the next line.
	`x`	(**package-menu-execute** &optional NOQUERY)	execute : Perform marked Package Menu actions. • Packages marked for installation are downloaded and installed; • packages marked for deletion are removed. • Optional argument NOQUERY non-nil means do not ask the user to confirm.
Download fresh copy of package archive contents	**M-x package-refresh-contents**		
Install Package	**M-x package-install**	(**package-install** PKG &optional DONT-SELECT)	

Package Managers — References

Topic & Link	Note
Emacs Package Repositories	
Elpa	The original GNU package distribution site.
Emacs Wiki - ELPA	
MELPA	This seems to include more packages than Elpa; for example SLIME and rust-mode are available at MELPA but not at Elpa. In some other cases, Elpa has packages that are also in MELPA, like crisp. • This can also be installed, see: http://melpa.org/#/getting-started
Marmalade repo	An older site. Deprecated. Seems no longer active. There seems to be a security issue with this site.
Cask	
Modern Emacs package management with Cask and Pallet @ LambdaCat	General Intro and experience about Cask & Pallet. **Read First!**
Cask Documentation	Read the complete info there as the second step.
Cask @ Github	
EPL @ Github	Same author than Cask’s.
Binding keys and loading code	
Auto loading	
Keymaps	
Rebinding keys in your init file	
Emacs's Key Syntax Explained	
How can I simulate an arbitary key event from Elisp?	
writing lisp emacs key binding and cannot specify the <delete> character	
If <DEL> Fails to Delete	
Modifier Keys	
Emacs: print key binding for a command or list all key bindings	
Mastering Key Bindings in Emacs	