↖ Topic Index

Package Management

Description	Keystroke	Function	Note
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`.		
Find Elisp Package See also: ⅀ **Help/Info**	• `C-h p` • `<f1> p`	(**finder-by-keyword**)	Find packages matching a given keyword. Useful to search for packages supporting a specific concept.
Describe a package See also: ⅀ **Help/Info**	• `C-h P` • `<f1> P`	(**describe-package** PACKAGE)	Display the full documentation of PACKAGE (a symbol). • Prompts for the package name. • Shows whether it is installed or not, its version, the features it implements and some extra notes.
Install Package	M-x package-install	(**package-install** PKG &optional DONT-SELECT)	Install the package PKG. • Prompts for the package name.
Download fresh copy of package archive contents	M-x package-refresh-contents	(**package-refresh-contents** &optional ASYNC)	Download descriptions of all configured ELPA packages. • For each archive configured in the variable ‘package-archives’, inform Emacs about the latest versions of all packages it offers, and make them available for download. • Optional argument ASYNC specifies whether to perform the downloads in the background.
Show PEL user option and package info See also: ⅀ **Help/Info** ⅀ **Customize**	`<f11> ? e ?`	(**pel-package-info** &optional FULL-REPORT ON-STDOUT	Display the following information inside a *pel-user-options* buffer: • name of custom file, package-user-dir, the number of PEL user-options, and the number of them that are active, number of loaded files, and features. • The number of Elpa packages active: the count of the ones directly installed because of active PEL user-options and the count of them installed as dependencies of the first group. • The number of Emacs Lisp files stored in the ~/.emacs.d/utils (or equivalent directory) as a result of PEL user options. • The number of elpa-compliant packages that have a newer version and could be updated. • With optional argument, like `C-u`, generates a full report with more details.
List Packages	• `<f11> ? e P` • `<f11> SPC l 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. See minor mode commands below.
	• `<f12> l p`		☛ The `<f12> l p` binding is available for buffers in Emacs-Lisp mode.
List Packages Mode Operations	The following rows describe the commands that can be issued from the buffer that lists the packages, in the *Package* buffer.		
Quick Help	`h`	(**package-menu-quick-help**)	Show short key binding help for ‘package-menu-mode’. • The full list of keys can be viewed with `C-h m` (or its equivalent `<f1> m`)
Move point to next line	`n`	(**next-line** &optional ARG TRY-VSCROLL)	Move to next line
Move point to previous line	`p`	(**previous-line** &optional ARG TRY-VSCROLL)	Move to previous line
Filter list of packages	`f`	(**package-menu-filter** KEYWORD)	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’.
Toggle visibility of obsolete packages	`(`	(**package-menu-toggle-hiding**)	Toggle visibility of obsolete available packages.
Hide package/describe associate package	`H`	(**package-menu-hide-package**)	hide-package : Hide a package under point. • If optional arg BUTTON is non-nil, describe its associated package.
Sort columns	`S`	(**tabulated-list-sort** &optional N)	Sort Tabulated List entries by the column at point. • With a numeric prefix argument N, sort the Nth column.
Describe package at point	• `?` • `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.
Refresh buffer	`g`	(**revert-buffer** &optional IGNORE-AUTO NOCONFIRM PRESERVE-MODES)	Redisplay/refresh information.
Refresh content (download package list again)	`r`	(**package-menu-refresh**)	Download the Emacs Lisp package archive. • This fetches the contents of each archive specified in ‘package-archives’, and then refreshes the package menu.
Mark package for deletion	`d`	(**package-menu-mark-delete** &optional NUM)	Mark an (installed) package for deletion and move to the next line. • NUM argument is unused.
Mark package for installation	`i`	(**package-menu-mark-install** &optional NUM)	Mark a package for installation and move to the next line. • NUM argument is unused.
Unmark package	• `u` • `<DEL>`	(**package-menu-mark-unmark** &optional NUM)	Clear any marks on a package and move to the next line. • NUM argument is unused.
Mark all upgradable packages: • **‘D’ for old versions to delete** • **‘I’ for new versions to install**	`U`	(**package-menu-mark-upgrades**)	Mark all upgradable packages in the Package Menu. • For each installed package with a newer version available, place an (**I**)nstall flag on the available version and a (**D**)elete flag on the installed version. • A subsequent M-x package-menu-execute call will upgrade the package. It can also be done with the `x` key in the *Packages* buffer. • If there’s an async refresh operation in progress, the flags will be placed as part of ‘package-menu--post-refresh’ instead of immediately.
Execute action specified by marks	`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.
Quit Window	`q`	(**quit-window** &optional KILL WINDOW)	Quit WINDOW and bury its buffer.

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.
Emacs Package Managers	
• package.el	Emacs provides the built-in package.el package manager. This is also what PEL currently uses. package.el provides an easy-to-use package management facility where each package is maintained inside its own directory and where the directory identifies the package name and version. Each package directory is placed in Emacs load-path variable. Several other package managers (if not all) use the same mechanism. Unfortunately as the number of packages increase over time as you use more packages, the longer load-path causes Emacs startup time to increase. I have been investigating the possibility of putting all Emacs Lisp code from all installed packages inside a single directory to speed up Emacs startup. This investigation is not yet complete but could lead to significant startup time reduction if all problems remaining from this re-organization are resolved.
Emacs Lisp Packages	
Preparing Lisp code for distribution	
• Quelpa	
• 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.
• el-get	
• Borg	
• Straight	Another package manager. External package meant to replace package.el
Articles on Straight	• Borg vs Straight.el?
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	