Dired — Directory Editor

Description	Keystroke	Function	Note
Dired	Dired is Emacs directory editor. You can open a Dired buffer by visiting a directory instead of a file. With it you can operate on the files of the directory and its sub-directories. Dired support is based on several Emacs Lisp files (called *libraries* in Emacs-speak). They are all part of Emacs • dired : built-in, main functionality • dired-aux : auxiliary features, less commonly used parts of dired • dired-x : dired extra features. PEL lazily loads it when **pel-use-dired-x** is set to **t**. • The dired-x commands are coloured in **blue** in the following table. • epa : EasyPG Assistant (GNU Privacy Guard) : built-in , but requires installed and configured OpenPGP compliant software like GNU PG.		
Open this PDF file. See also: ∑ **Help/Info**	`<f11> f <f1> 2`	(**pel-help-pdf**)	Open ∑M **Dired** local PDF file.
Configure PEL Dired Support See also:∑ **Customize**	`<f11> f <f2> 2`	(**pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL support for dired, directory editor. • If OTHER-WINDOW is non-nil (use `C-u`), display in other window.
Configure Emacs Dired Support See also:∑ **Customize**	`<f11> f <f3> 2`	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs support for dired, directory editor. Other choices are available for neotree and ztree.
Entering Dired Mode	The following commands can be used to list files in a directory a=in a buffer that operates in Dired Mode.		
Open a directory editor See also: • ∑ **File mngt** • ∑ **Completion/Input**	• `C-x d` • `⌘-D`	• (**dired** DIRNAME &optional SWITCHES) – – – – – – – – – – • (**ido-dired**)	Opens a Dired-mode buffer on the specified directory. Prompt for the directory name. With PEL, you can activate and dynamically select one of several input completion modes that impact this command. See the ∑ **Completion/Input** for how to activate and select the mode. Only two of the possible functions are listed here: the standard Emacs function and the function used when the IDO mode is used. Others are available and easily accessible with PEL.
Open (visit) a file/directory See also: • ∑ **File mngt** • ∑ **Completion/Input**	`C-x C-f`	• (**find-file** FILENAME &optional WILDCARDS) – – – – – – – – – – • (**ido-find-file**)	Prompt for the file or directory name to open. Open the selected file/directory in a buffer with the appropriate mode. For directory, the buffer opens in Dired-mode. This can be replaced by the ido-mode by the ido-find-file: it provides suggestions. When ido mode is used, you can also: • Type `C-x f` to change to original find-file • Type `C-j` to accept the file/directory name verbatim without replacement or suggestion. Note: it is also possible to change the read-only state of a buffer with `C-x C-q`. So you can open a file with `C-x C-f` and then change the buffer to read-only mode. With PEL, you can activate and dynamically select one of several input completion modes that impact this command. See the ∑ **Completion/Input** for how to activate and select the mode. Only two of the possible functions are listed here: the standard Emacs function and the function used when the IDO mode is used. Others are available and easily accessible with PEL.
Entering Dired via ffap commands See also: ∑ **File mngt**	Emacs provides the ffap (find file at point) command set with Dired commands, These are listed below. PEL activates the Emacs built-in ffap library when the **pel-use-ffap** user option is set to either **t** or to **ffap-bindings**. In both cases these activate the key bindings shown below. • When **pel-use-ffap** is set to ffap-bindings, then PEL also activates the standard ffap bindings which take over the behaviour of the main file finding and dired commands. This means that Ido, Ivy or Helm are no longer available for these commands. • If **pel-use-ffap** is only set to **t** then the standard ffap bindings is not activated.		
Start Dired with file at point	`<f11> f a d`	(**dired-at-point** &optional FILENAME)	Start Dired, defaulting to file at point. See 'ffap'.
Start Dired with file at point in other window	`<f11> f a D`	(**ffap-dired-other-window**)	Like 'dired-at-point', but put buffer in another window.
Start Dired with file at point in other frame	`<f11> f a M-d`	(**ffap-dired-other-frame**)	Like 'dired-at-point', but put buffer in another frame.
Dired Mode commands	A buffer in Dired Mode provides a large set of specialized single key commands. There's also a set of secondary/extended Dired Modes that can be activated via some of these commands. As an example on what can be done, the following describes the keys to use to show only the files that have a specific file extension: • Mark all .el files: `%f \.el$ RET` • Toggle the marking to mark all other files: `t` • Kill the lines that are now marked: `k` • To refresh the file list: `g`		
Help - summary of available commands	`?`	(**dired-summary**)	Summarize basic Dired commands on the echo line and show recent Dired errors.
Help - complete help	• `h` • `<f1> m`	(**describe-mode** &optional BUFFER)	Display documentation of Dired major mode and available minor modes. • A brief summary of the minor modes comes first, followed by the major mode description. This is followed by detailed descriptions of the minor modes, each on a separate page.
Navigate in Dired Buffer	The following commands select a new file line in the buffer.		
Move down one line	• `SPC` • `n` • `C-n`	(**dired-next-line** ARG)	Down one line
Move up one line	• `S-SPC` • `p` • `C-p`	(**dired-previous-line** ARG)	Up one line
Move to next marked file	• `M-}` • `* C-n`	(**dired-next-marked-file** ARG &optional WRAP OPOINT)	Move to the ARGth next marked file. • ARG is the numeric prefix argument and defaults to 1. • If WRAP is non-nil, which happens interactively, wrap around to the beginning of the buffer and search from there, if no marked file is found after this line. • Optional argument OPOINT specifies the buffer position to return to if no ARGth marked file is found; it defaults to the position where this command was invoked.
Move to previous marked file	• `M-{` • `* C-p`	(**dired-prev-marked-file** ARG &optional WRAP)	Move to the ARGth previous marked file. • ARG is the numeric prefix argument and defaults to 1. • If WRAP is non-nil, which happens interactively, wrap around to the end of the buffer and search backwards from there, if no ARGth marked file is found before this line.
Move to the next directory line	`>`	(**dired-next-dirline** ARG &optional OPOINT)	Goto ARGth next directory file line.
Move to previous directory line	`<`	(**dired-prev-dirline** ARG)	Goto ARGth previous directory file line.
Move to line of specified file	`j`	(**dired-goto-file** FILE)	Go to line describing file FILE in this Dired buffer. Prompts for file name.
Searching for Files	The following commands search for file names in the dired buffer, ignoring all other text in the buffer. By setting dired-isearch-filenames user option to **t**, you can use the normal search commands (and bindings) in Dired buffer to also only search in the file names and ignoring the other text. Then the following key bindings are not needed.		
ISearch for file name	`M-s f C-s`	(**dired-isearch-filenames**)	Search for a string using Isearch only in file names in the Dired buffer.

Description	Keystroke	Function	Note
ISearch for marked file name	`M-s a C-s`	(**dired-do-isearch**)	Search for a string through all marked files using Isearch.
Regex ISearch for file name	`M-s f C-M-s`	(**dired-isearch-filenames-regexp**)	Search for a regexp using Isearch only in file names in the Dired buffer.
Regex ISearch for marked file name	`M-s a C-M-s`	(**dired-do-isearch-regexp**)	Search for a regexp through all marked files using Isearch.
Control Visibility			
Toggle sorting lines by date	`s`	(**dired-sort-toggle-or-edit** &optional ARG)	Toggle sorting by date, and refresh the Dired buffer. With a prefix argument, edit the current listing switches instead.
Toggle visibility of details info	`(`	(**dired-hide-details-mode** &optional ARG)	Toggle visibility of detailed information in current Dired buffer. • When this minor mode is enabled, details such as file ownership and permissions are hidden from view. User options: 'dired-hide-details-hide-symlink-targets', 'dired-hide-details-hide-information-lines'.
Hide/unhide current sub-directory, move to next one	`$`	(**dired-hide-subdir** ARG)	Hide or unhide the current subdirectory and move to next directory. • Optional prefix arg is a repeat factor.
Hide/Show all sub-directories	`M-$`	(**dired-hide-all** &optional IGNORED)	Hide all subdirectories, leaving only their header lines. • If there is already something hidden, make everything visible again. • Use M-x dired-hide-subdir to (un)hide a particular subdirectory.
Refresh	`g`	(**revert-buffer** &optional IGNORE-AUTO NOCONFIRM PRESERVE-MODES)	Refresh: read directory again.
Re-display from point	`l`	(**dired-do-redisplay** &optional ARG TEST-FOR-SUBDIR)	Relist the file at point or the marked files, or a subdirectory. • Redisplay all marked (or next ARG) files. • If on a subdir line, redisplay that subdirectory. In that case, a prefix arg lets you edit the 'ls' switches used for the new listing.
Quit Dired Window	`q`	(**quit-window** &optional KILL WINDOW)	Quit WINDOW and bury its buffer. • WINDOW must be a live window and defaults to the selected one. • With prefix argument KILL non-nil, kill the buffer instead of burying it.
Visit Files	The following commands are used to '*visit*' a file (i.e. open the file for editing), to '˜view' a file (open the file in read-only mode). The last 2 commands launch an OS application to open the file: the application associated with the type of file.		
Visit file/directory on line	• `RET` • `e` • `f`	(**dired-find-file**)	Visit current line's file or directory. For directory create new Dired buffer.
Visit parent directory	`^`	(**dired-up-directory** &optional OTHER-WINDOW)	Up one directory level (creates a new buffer)
Visit file/directory on line in current buffer	`a`	(**dired-find-alternate-file**)	Access current line's file or directory. Kill old Dired buffer.
Visit file/directory in another window	`o`	(**dired-find-file-other-window**)	Visit current line's file or directory inside **other** window. • Select that other window.
Visit file/directory in another window (keep Dired window selected)	`C-o`	(**dired-display-file**)	In Dired, display this file or directory in another window. • Does not select the file window, keeps the Dired buffer window selected.
Visit all files marked	`F`	(**dired-do-find-marked-files** &optional NOSELECT)	Find all marked files displaying all of them simultaneously. • With optional NOSELECT just find files but do not select them. • The current window is split across all files marked, as evenly as possible. Remaining lines go to bottom-most window. The number of files that can be displayed this way is restricted by the height of the current window and 'window-min-height'. • To keep Dired buffer displayed, type **`C-x 2`** first. • To display just marked files, type **`C-x 1`** first.
View a file	`v`	(**dired-view-file**)	In Dired, examine a file in view mode, returning to Dired when done. • When file is a directory, show it in this buffer if it is inserted. • Otherwise, display it in another (Dired) buffer.
Open file/directory with registered OS application See also: • **∑ File mngt**	`z`	(**pel-open-in-os-app** &optional FNAME)	Open the file with the OS-registered application. Currently only works on Linux, macOS and Windows. • If several files are marked, it opens all these files, each one with its default OS application. • For example, using this on a directory name, on macOS this opens Finder on that folder.
	`W`	(**browse-url-of-dired-file**)	In Dired, ask a WWW browser to display the file named on this line. • That also opens directory browser when executed on a directory and is not restricted to macOS!
Open Info/Man files			
Open an Info file	`I`	(**dired-info**)	Run 'info' on this file, opening the info manual on it. If the file is not an info file, then the file is opened in its own mode.
Open a Man file	`N`	(**dired-man**)	Run 'man' on this file.
Subdirectories in Dired	With the first command, i, issued on a line showing a file directory, Dired adds the list of the files of this sub-directory at the end of the Dired buffer. The other 4 commands allow quick navigation to the header line of these directory lists.		
Insert content of sub-directory in current Dired buffer	`i`	(**dired-maybe-insert-subdir** DIRNAME &optional SWITCHES NO-ERROR-IF-NOT-DIR-P)	Insert a subdirectory listing in this buffer. Use on subdirectory line. • The sub-directory is listed below the lines of the current directory.
Move to header line of next sub-directory	`C-M-n`	(**dired-next-subdir** ARG &optional NO-ERROR-IF-NOT-FOUND NO-SKIP)	Go to next subdirectory, regardless of level.
Move to header line of previous sub-directory	`C-M-p`	(**dired-prev-subdir** ARG &optional NO-ERROR-IF-NOT-FOUND NO-SKIP)	Go to previous subdirectory, regardless of level. • When called interactively and not on a subdir line, go to this subdir's line.
Move to the the header line of the file-system directory up in the directory tree	`C-M-u`	(**dired-tree-up** ARG)	Go up ARG levels in the dired tree.
Move to the the header line of the file-system directory down in the directory tree	`C-M-d`	(**dired-tree-down**)	Go down in the dired tree.
Mark files			
Mark file for deletion —>D	`d`	(**dired-flag-file-deletion** ARG &optional INTERACTIVE)	Flag a file 'D' for deletion.

Description	Keystroke	Function	Note
Flag for deletion files selected by regexp —>D	`%d`	(**dired-flag-files-regexp** REGEXP)	In Dired, flag all files containing the specified REGEXP for deletion. • The match is against the non-directory part of the filename. • Use '^' and '$' to anchor matches. Exclude subdirs by hiding them. • '.' and '..' are never flagged.
Mark file for later command —> *	• `m` • `*m`	(**dired-mark** ARG &optional INTERACTIVE)	Mark the file at point in the Dired buffer. Used to identify files for later commands. • If the region is active, mark all files in the region. • Otherwise, with a prefix arg, mark files on the next ARG lines. • If on a subdir headerline, mark all its files except '.' and '..'.
Mark all files with names matching regexp —> *	• `%m` • `*%`	(**dired-mark-files-regexp** REGEXP &optional MARKER-CHAR)	Mark all files matching REGEXP for use in later commands. • A prefix argument means to unmark them instead. • '.' and '..' are never marked. ☝REGEXP is an Emacs regexp, not a shell wildcard. Thus, use '`\.o$`' for object files--just '`.o`' will mark more than you might think.
Mark for deletion all autosaved files —> D	`#`	(**dired-flag-auto-save-files** &optional UNFLAG-P)	Flag for deletion files whose names suggest they are auto save files. • A prefix argument says to unmark or unflag those files instead.
Flag/unflag backup files for deletion —> D	`~`	(**dired-flag-backup-files** &optional UNFLAG-P)	Flag all backup files (names ending with '~') for deletion. • With prefix argument, unmark or unflag these files.
Flag numerical backup files for deletion —> D	`.`	(**dired-clean-directory** KEEP)	Flag numerical backups for deletion. • Spares 'dired-kept-versions' latest versions, and 'kept-old-versions' oldest. • Positive prefix arg KEEP overrides 'dired-kept-versions'; • Negative prefix arg KEEP overrides 'kept-old-versions' with KEEP made positive. To clear the flags on these files, you can use M-x dired-flag-backup-files with a prefix argument.
(Un/)Mark executables —> *	`**`	(**dired-mark-executables** UNFLAG-P)	Mark all executable files. • With prefix argument, unmark or unflag all those files.
(Un/)Mark all files with specific extension	`*.`	(**dired-mark-extension** EXTENSION &optional MARKER-CHAR)	Mark all files with a certain EXTENSION for use in later commands. • A '.' is automatically prepended to EXTENSION when not present. • EXTENSION may also be a list of extensions instead of a single one. • Optional MARKER-CHAR is marker to use. • Interactively, ask for EXTENSION. • Prefixed with one **`C-u`**, unmark files instead. • Prefixed with two **`C-u`**'s, prompt for MARKER-CHAR and mark files with it.
(Un/)Mark directories	`*/`	(**dired-mark-directories** UNFLAG-P)	Mark all directory file lines except '.' and '..'. • With prefix argument, unmark or unflag all those files.
(Un/)Mark all symlinks	`*@`	(**dired-mark-symlinks** UNFLAG-P)	Mark all symbolic links. • With prefix argument, unmark or unflag all those files.
Mark files that Dired normally omits	`*O`	(**dired-mark-omitted**)	Mark files that Dired consider files that should be ignored in most operations: like output files, hidden files (files that start with a period including (D)VCS depots, etc. 👀 These are the file matching 'dired-omit-files' and 'dired-omit-extensions'.
Mark for deletion all "garbage" files	`%&`	(**dired-flag-garbage-files**)	Flag for deletion all files identified as "garbage". 👀 These are the files that match 'dired-garbage-files-regexp'. The default is set to: "`\\(?:\\.\\(?:aux\\|bak\\|dvi\\|log\\|orig\\|rej\\|toc\\)\\)\\'`"
Flag all files with content matching regexp	`%g`	(**dired-mark-files-containing-regexp** REGEXP &optional MARKER-CHAR)	Mark all files with contents containing REGEXP for use in later commands. • A prefix argument means to unmark them instead. • '.' and '..' are never marked. • Note that if a file is visited in an Emacs buffer, and 'dired-always-read-filesystem' is nil, this command will look in the buffer without revisiting the file, so the results might be inconsistent with the file on disk if its contents has changed since it was last visited.
Mark files using lisp predicate	• `M-(` • `*(`	(**dired-mark-sexp** PREDICATE &optional UNFLAG-P)	Mark files for which PREDICATE returns non-nil. Prompts for a lisp expression. • With a prefix arg, unmark or unflag those files instead. *See the command help for more information.*
Unmark files			
Unmark file	• `u` • `*u`	(**dired-unmark** ARG &optional INTERACTIVE)	Unmark a file or all files of inserted subdirectory
Unmark file and move up	• `DEL` • `*DEL`	(**dired-unmark-backward** ARG)	Back up one line and unmark or unflag.
Remove all marks from all files in Dired buffer	• `U` • `*!`	(**dired-unmark-all-marks**)	Remove all marks from all files in the Dired buffer.
Remove a specific mark from all files	• `M-DEL` • `*?`	(**dired-unmark-all-files** MARK &optional ARG)	Remove a specific mark (or any mark) from every file. • After this command, type the mark character to remove, or type RET to remove all marks. • With prefix arg, query for each marked file. Type C-h at that time for help.
Toggle al marks	`t`	(**dired-toggle-marks**)	Toggle marks: marked files become unmarked, and vice versa. • Files marked with other flags (such as 'D') are not affected. • '.' and '..' are never toggled. • As always, hidden subdirs are not affected.
Undo Dired Buffer Changes	Changes to the Dired buffer like marking, un-marking file lines can be un-done with the dired-undo command described below. ⚠ Performing an undo after some commands may get the Dired buffer out of sync with the corresponding file system. For example if you execute dired-undo after renaming a file the old file name will be displayed back in the Dired buffer and that will not reflect the real file name. Refresh the Dired buffer in such cases.		
	• `M-u` • `C-x u` • `C-/`	(**dired-undo**)	Undo in a Dired buffer. • This doesn't recover lost files, it just undoes changes in the buffer itself. • You can use it to recover marks, killed lines or subdirs.
Change mark type			
Change mark type	`*c`	(**dired-change-marks** &optional OLD NEW)	Change all OLD marks to NEW marks. • OLD and NEW are both characters used to mark files.
Kill/Delete			
Kill all marked lines (not files)	`k`	(**dired-do-kill-lines** &optional ARG FMT)	Kill all marked lines (not the files). • With a prefix argument, kill that many lines starting with the current line. • (A negative argument kills backward.) • If you use this command with a prefix argument to kill the line for a file that is a directory, which you have inserted in the Dired buffer as a subdirectory, then it deletes that subdirectory from the buffer as well. • To kill an entire subdirectory (without killing its line in the parent directory), go to its directory header line and use this command with a prefix argument (the value does not matter).
Delete files marked for deletion	`x`	(**dired-do-flagged-delete** &optional NOMESSAGE)	Delete (eXpunge) the files flagged 'D'.

Description	Keystroke	Function	Note
Delete marked files/ directories	`D`	(**dired-do-delete** &optional ARG)	Delete all marked (or next ARG) files. 'dired-recursive-deletes' controls whether deletion of non-empty directories is allowed.
Create			
Create directory	`+`	(**dired-create-directory** DIRECTORY)	Create a new directory. • Parent directories of DIRECTORY are created as needed. • If DIRECTORY already exists, signal an error.
Copy files			
Copy file/directory	`C`	(**dired-do-copy** &optional ARG)	Copy all marked (or next ARG) files, or copy the current file. When operating on just the current file, prompt for the new name.
Copy files which match regexp	`%C`	(**dired-do-copy-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME)	Copy selected files whose names match REGEXP to NEWNAME. • With non-zero prefix argument ARG, the command operates on the next ARG files. Otherwise, it operates on all the marked files, or the current file if none are marked. • As each match is found, the user must type a character saying what to do with it. For directions, type **`C-h`** at that time. NEWNAME may contain \<n> or \& as in 'query-replace-regexp'. REGEXP defaults to the last regexp used. • With a zero prefix arg, renaming by regexp affects the absolute file name. • Normally, only the non-directory part of the file name is used and changed.
Rename files			
Rename file/move file into other directory	`R`	(**dired-do-rename** &optional ARG)	Rename a file or move the marked files to another directory.
Rename files selected by regexp	• `%R` • `%r`	(**dired-do-rename-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME)	Rename selected files whose names match REGEXP to NEWNAME. • With non-zero prefix argument ARG, the command operates on the next ARG files. Otherwise, it operates on all the marked files, or the current file if none are marked. • As each match is found, the user must type a character saying what to do with it. For directions, type **`C-h`** at that time. NEWNAME may contain \<n> or \& as in 'query-replace-regexp'. REGEXP defaults to the last regexp used. • With a zero prefix arg, renaming by regexp affects the absolute file name. Normally, only the non-directory part of the file name is used and hanged.
Rename all marked files to lowercase names	`%l`	(**dired-downcase** &optional ARG)	Rename all marked (or next ARG) files to lower case.
Rename all marked files to uppercase names	`%u`	(**dired-upcase** &optional ARG)	Rename all marked (or next ARG) files to upper case.
Compare Files			

Diff file	`=`	(**dired-diff** FILE &optional SWITCHES)	Compare file at point with FILE using 'diff'.
			• Prompt for FILE, but :
			  • If the mark is active in Transient Mark mode, use the file at the mark as the default for FILE. (That's the mark set by C-SPC, not by Dired's M-x dired-mark command.)
			  • If the file at point has a backup file, use that as the default FILE.
			  • If the file at point is a backup file, use its original, if that exists and can be found.
			• FILE is the first argument given to the 'diff' function (the "old" version). The file at point is the second argument given to 'diff'.
			• With prefix arg, prompt for second argument SWITCHES, which is the string of command switches used as the third argument of 'diff'.
			Customizations of 'backup-directory-alist' and 'make-backup-file-name-function' change where this function searches for the backup file, and affect its ability to find the original of a backup file.

Grep & Replace			
Grep for text in marked file(s)	`A`	(**dired-do-find-regexp** REGEXP)	Find all matches for REGEXP in all marked files. • For any marked directory, all of its files are searched recursively. • However, files matching 'grep-find-ignored-files' and subdirectories matching 'grep-find-ignored-directories' are skipped in the marked directories. ☝ REGEXP should use constructs supported by your local 'grep' command.
Replace text found in marked file(s) See also: **Search/Replace**	`Q`	(**dired-do-find-regexp-and-replace** FROM TO)	Replace matches of FROM with TO, in all marked files. • For any marked directory, matches in all of its files are replaced, recursively. However, files matching 'grep-find-ignored-files' and subdirectories matching 'grep-find-ignored-directories' are skipped in the marked directories. ☝ REGEXP should use constructs supported by your local 'grep' command.
Byte Compile & Load	When writing Emacs Lisp source code, use the following commands to byte-compile the files or load them into Emacs.		
Byte compile marked Emacs Lisp files	`B`	(**dired-do-byte-compile** &optional ARG)	Byte compile marked (or next ARG) Emacs Lisp files.
Load Emacs Lisp files	`L`	(**dired-do-load** &optional ARG)	Load the marked (or next ARG) Emacs Lisp files.
Compress/Uncompress	Compress and uncompress marked files with the following command.		
Compress/Uncompress file(s)	`Z`	(**dired-do-compress** &optional ARG)	Compress or uncompress marked (or next ARG) files. • If invoked on a directory, compress all of the files in the directory and all of its subdirectories, recursively, into a .tar.gz archive. • If invoked on a .tar.gz or a .tgz or a .zip or a .7z archive, uncompress and unpack all the files in the archive.
Shell Commands	Use the following commands to execute shell commands on each marked file and also run a shell command with the current directory set to the directory visited by the Dired buffer.		
Execute shell command on marked files	• `!` • `X`	(**dired-do-shell-command** COMMAND &optional ARG FILE-LIST)	Execute shell command in the directory. Run a shell command COMMAND on the marked files. • If no files are marked or a numeric prefix arg is given, the next ARG files are used. Just **`C-u`** means the current file. • The prompt mentions the file(s) or the marker, as appropriate. • If there is a '*' in COMMAND, surrounded by whitespace, this runs COMMAND just once with the entire file list substituted there. • If there is no '*', but there is a '?' in COMMAND, surrounded by whitespace, or a "?" this runs COMMAND on each file individually with the file name substituted for '?' or "?". • Otherwise, this runs COMMAND on each file individually with the file name added at the end of COMMAND (separated by a space). • When COMMAND runs, its working directory is the top-level directory of the Dired buffer, so output files usually are created there instead of in a subdir.

Description	Keystroke	Function	Note
Asynchronously execute shell command on marked files	**`&`**	(**dired-do-async-shell-command** COMMAND &optional ARG FILE-LIST)	Like 'dired-do-shell-command', but adds '&' at the end of COMMAND to execute it asynchronously. • When operating on multiple files, asynchronous commands are executed in the background on each file in parallel. • In shell syntax this means separating the individual commands with '&'. However, when COMMAND ends in ';' or ';&' then commands are executed in the background on each file sequentially waiting for each command to terminate before running the next command. • In shell syntax this means separating the individual commands with ';'. The output appears in the buffer '*Async Shell Command*'.
Execute a shell command inside the current Dired directory	**`M-!`**	(**dired-smart-shell-command** COMMAND &optional OUTPUT-BUFFER ERROR-BUFFER)	Like function 'shell-command', but in the current Virtual Dired directory. • Does not use the marked files; just uses the directory identified by Dired as the current directory for the command. • For example, to execute `make` in the directory identified by the Dired buffer. That way you don't need to open a shell. The disadvantage is that you'll have to wait for the command to finish to see the output inside the *Shell Command Output* buffer.
Modify file attributes	Use the following commands to modify the files attributes.		
Change the file(s) group (chgrp)	**`G`**	(**dired-do-chgrp** &optional ARG)	Change the group of the marked (or next ARG) files. • Type **`M-n`** to pull the file attributes of the file at point into the minibuffer.
Change the file(s) mode (chmod)	**`M`**	(**dired-do-chmod** &optional ARG)	Change the mode of the marked (or next ARG) files. • Symbolic modes like 'g+w' are allowed. • Type **`M-n`** to pull the file attributes of the file at point into the minibuffer.
Change the file(s) ownership (chown)	**`O`**	(**dired-do-chown** &optional ARG)	Change the owner of the marked (or next ARG) files. • Type **`M-n`** to pull the file attributes of the file at point into the minibuffer.
Touch the marked file(s)	**`T`**	(**dired-do-touch** &optional ARG)	Change the timestamp of the marked (or next ARG) files using 'touch'. • Type M-n to pull the file attributes of the file at point into the minibuffer. ☝ Use this to set the timestamp of all marked files to timestamp of the current file!
File Hard/Soft Links	The following commands can be used to manage hard and soft links.		
Create hard link for current or all marked files.	**`H`**	(**dired-do-hardlink** &optional ARG)	Add names (hard links) for current file or all marked (or next ARG) files. • When operating on just the current file, prompts and you specify the new name. • When operating on multiple or marked files, you specify a directory and new hard links are made in that directory with the same names that the files currently have. The default suggested for the target directory depends on the value of 'dired-dwim-target'.
Hardlink files selected by regexp	**`%H`**	(**dired-do-hardlink-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME)	Hardlink selected files whose names match REGEXP to NEWNAME. • With non-zero prefix argument ARG, the command operates on the next ARG files. Otherwise, it operates on all the marked files, or the current file if none are marked. • As each match is found, the user must type a character saying what to do with it. For directions, type **`C-h`** at that time. NEWNAME may contain \<n> or \& as in 'query-replace-regexp'. REGEXP defaults to the last regexp used. • With a zero prefix arg, renaming by regexp affects the absolute file name. • Normally, only the non-directory part of the file name is used and changed.
Create symbolic links for current or all marked files	**`S`**	(**dired-do-symlink** &optional ARG)	Make symbolic links to current file or all marked (or next ARG) files. • When operating on just the current file, you specify the new name. • When operating on multiple or marked files, you specify a directory and new symbolic links are made in that directory with the same names that the files currently have. The default suggested for the target directory depends on the value of 'dired-dwim-target', which see. • For relative symlinks, use M-x dired-do-relsymlink.
Symlink files selected by regexp	**`%S`**	(**dired-do-symlink-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME)	Symlink selected files whose names match REGEXP to NEWNAME. See function 'dired-do-rename-regexp' for more info.

Create relative symbolic links for current or all marked files	**`Y`**	(**dired-do-relsymlink** &optional ARG)	Relative symlink all marked (or next ARG) files into a directory.
			• Otherwise make a relative symbolic link to the current file.
			• This creates relative symbolic links like:
			`foo -> ../bar/foo`
			not absolute ones like:
			`foo -> /ugly/file/name/that/may/change/any/day/bar/foo`

Create relative symbolic links for files selected by regexp	**`%Y`**	(**dired-do-relsymlink-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME)	RelSymlink all marked files containing REGEXP to NEWNAME. • See functions 'dired-do-rename-regexp' and 'dired-do-relsymlink' or more info.
OpenPGP Support	The following commands require OpenPGP-compliant encryption software tools (such as GNU Privacy Guard) to be available.		
Decrypt marked file(s)	**`:d`**	(**epa-dired-do-decrypt**)	Decrypt marked files.
Encrypt marked file(s)	**`:e`**	(**epa-dired-do-encrypt**)	Encrypt marked files.
Sign marked file(s)	**`:s`**	(**epa-dired-do-sign**)	Sign marked files.
Verify	**`:v`**	(**epa-dired-do-verify**)	Verify marked files.
Miscellaneous			
Print the file(s)	**`P`**	(**dired-do-print** &optional ARG)	Print the marked (or next ARG) files. 🎛 Uses the shell command coming from variables 'lpr-command' and 'lpr-switches' as default.
Visit current file as mailbox	**`V`**	(**dired-do-run-mail**)	Visit the current file as a mailbox, using VM or RMAIL. • Prompt for confirmation first; if the user says yes, call 'dired-vm' if 'dired-bind-vm' is non-nil, 'dired-rmail' otherwise.
WDired — Writable Dired : Editable Dired	When the Wdired mode is on, the file names, symlinks and file permissions (if enabled via customization) can be modified by simply editing their names as they show in the buffer. 🎛 Wdired group user options. With PEL to customize them use: **`<f11> <f2> G wdired`** to see, change them or dead more info about them. • wdired-allow-to-change-permissions: t: allow editing the permission bits directly in the buffer to change them. Useful with multiple-cursors to change permissions of several files at the same time! • wdired-allow-to-redirect-links : t: allow editing targets of symbolic links. • wdired-confirm-overwrite : t: request confirmation when renames overwrite files. This is the default. Otherwise it does not prompt. • wdired-create-parent-directories : t: creates parent directories automatically when renaming a file with a destination path within non-existent directory. Otherwise the rename fails. • wdired-keep-marker-rename : controls whether Dired file marks are kept when a file is renamed & committed by **`C-c C-c`** or **`C-x C-s`** • • wdired-use-interactive-rename : controls whether confirmation is required before renaming file. Off by default. • wdired-use-dired-vertical-movement : controls the way the cursor moves. Like in normal editing (default), like in dired-mode or specialized.		
Toggle read-only status of buffer	• **`C-x C-q`** • **`<f11> b r`**	(**dired-toggle-read-only**)	Toggle the Editable Dired (WDired mode) on/off. When exiting, checks if there are un-committed changes and prompts to commit them.

Description	Keystroke	Function	Note
Commit the changes	• **C-c C-c** • **C-x C-s**	(**wdired-finish-edit**)	Commit the changes: actually rename files based on your editing in the Dired buffer.
Abort, does not commit changes.	• **C-c Esc** • **C-c C-C-k**	(**wdired-abort-changes**)	Abort changes and return to dired mode.
Using dired-narrow	The external dired-narrow package provides a way to limit the number of files shown in the dired buffer by using expressions typed following one of the following commands. 📦 The following commands require the external dired-narrow package to be installed. 🏄 This PEL bindings below are mode sensitive and are only available when the ***pel-use-dired-narrow*** user option is **t**. ☝ To widen the list back, refresh the Dired buffer: press **g**.		
Narrow the file list to specific filter	**<f12> s**	(**dired-narrow**)	Narrow a dired buffer to the files matching a string. (not a regexp) • If the string contains spaces, then each word is matched against the file name separately. To succeed, all of them have to match but the order does not matter. • For example "foo bar" matches filename “bar-and-foo.el".
Narrow the file list to specific regexp filter	**<f12> r**	(**dired-narrow-regexp**)	Narrow a dired buffer to the files matching a regular expression. • For example, to list all .el files, use: **.el$**
Narrow the file list to specific fuzzy filter	**<f12> f**	(**dired-narrow-fuzzy**)	Narrow a dired buffer to the files matching a fuzzy string. A fuzzy string is constructed from the filter string by inserting ".*" between each letter. This is then matched as regular expression against the file name.

Dired — References

Description & link	Notes
Dired @ wikipedia	An overview.
Dired in Emacs Manual	Dired table of Content in the emacs manual.
GNU Dired Extra Manual	
GNU Easy PG User Assistant User's Manual	
Mike Sperber's dired page	Page of Dired current maintainer
Emacs: techniques to narrow Dired - Youtube video	A quick video on how to use straight dired-mode to list a sub-set of all files. It also describes the dired-narrow package though.
Integrating OS X and Emacs Dired	Jason Blevins describes how to launch macOS registered application for a specific file in Dired mode.
WDired: Editable Dired Buffers	Mickey Petersen quickly describes the power of WDired mode to manage file names.
Dired: Rename Multiple Files @ Pragmatic Emacs	A quick animated Gif demo showing how to rename files using wdired-mode (Editable Dired Mode)
Batch-Edit File Permissions In Dired @ Pragmatic Emacs	A quick animated Gif demo using wdired-mode, dired-narrow and multiple-cursors to set permissions of several files. Note: instead of setting wdired-allow-to-change-permissions in your init.el file, PEL proposes to change it in the customization system. It's easy to change and you don't have to write any Emacs Lisp code. You can also read the full description and see alternatives.