# 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. | | |
| **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) | • `C-x d`  • `⌘-D` | • (**dired** DIRNAME &optional SWITCHES)  — — — — — — — — — — — — —  • (**ido-dired**) | Opens a Dired-mode buffer on the specified directory.  Prompt for the directory name.  📑The PEL package use IDO when the **pel-use-ido-mode** customize variable is set to **t** and then the keys are bound to the (ido-dired) command. |
| **Open (visit) a file/directory**  (See also: ∑ File mngt) | `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.  📑The PEL package use IDO when the **pel-use-ido-mode** customize variable is set to **t** and then the keys are bound to the (ido-dired) command. |
| **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`  Then if you want to re-display all files, refresh with: `g` | | |
| **Mark file for deletion** | `d` | (**dired-flag-file-deletion** ARG &optional INTERACTIVE) | Flag a file 'D' for deletion. |
| **Mark file for later command** | `m` | (**dired-mark** ARG &optional INTERACTIVE) | Mark a file or subdirectory for later command |
| **Unmark file** | `u` | (**dired-unmark** ARG &optional INTERACTIVE) | Unmark a file or all files of inserted subdirectory |
| **Unmark file and move up** | `<DEL>` | (**dired-unmark-backward** ARG) | Back up one line and unmark or unflag. |
| **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. |
| **Delete files marked for deletion** | `x` | (**dired-do-flagged-delete** &optional NOMESSAGE) | Delete (eXpunge) the files flagged 'D'. |
| **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). |
| **Visit file/directory on line** | `<RET>` | (**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. |
| **Open file with registered OS application** | `z` | (**pel-dired-open**) | Open the file with the OS-registered application.   Currently only works on macOS.  • For example, using this on a directory name, Finder opens on that folder. |
| **Dired-subtree** | `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. |
| **Rename file/move file into other directory** | `R` | (**dired-do-rename** &optional ARG) | Rename a file or move the marked files to another directory. |
| **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. |
| **Toggle lines** | `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. |
| **Refresh** | `g` | (**revert-buffer** &optional IGNORE-AUTO NOCONFIRM PRESERVE-MODES) | Refresh: read directory again. |
| | `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. |
| **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 |
| **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. |
| **Byte compile marked Emacs Lisp files** | `B` | (**dired-do-byte-compile** &optional ARG) | Byte compile marked (or next ARG) Emacs Lisp files. |

| Description | Keystroke | Function | Note |
|---|---|---|---|
| **Execute shell command on marked files** | `!` | (**dired-do-shell-command** COMMAND &optional ARG FILE-LIST) | Execute shell command in the directory. Run a shell command COMMAND on the marked files.<br>• 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.<br>• The prompt mentions the file(s) or the marker, as appropriate.<br>• If there is a '*' in COMMAND, surrounded by whitespace, this runs COMMAND just once with the entire file list substituted there.<br>• 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 '`?`'.<br>• Otherwise, this runs COMMAND on each file individually with the file name added at the end of COMMAND (separated by a space). |
| **Mark for deletion all autosaved files** | `#` | (**dired-flag-auto-save-files** &optional UNFLAG-P) | Flag for deletion files whose names suggest they are auto save files.<br>A prefix argument says to unmark or unflag those files instead. |
| **Hide/unhide current sub-directory, move to next one** | `$` | (**dired-hide-subdir** ARG) | Hide or unhide the current subdirectory and move to next directory.<br>Optional prefix arg is a repeat factor. |
| **Mark for deletion all "*garbage*" files** | `%&` | (**dired-flag-garbage-files**) | Flag for deletion all files that match 'dired-garbage-files-regexp', which, by default, has the following value:<br>"\\(?:\\.\\(?:aux\\\|bak\\\|dvi\\\|log\\\|orig\\\|rej\\\|toc\\\)\\)\\'" |
| **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.<br>• 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.<br>• 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.<br>• With a zero prefix arg, renaming by regexp affects the absolute file name.<br>• Normally, only the non-directory part of the file name is used and  changed. |
| **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.<br>• 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.<br>• 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.<br>• With a zero prefix arg, renaming by regexp affects the absolute file name.<br>• Normally, only the non-directory part of the file name is used and  changed. |
| **Rename files selected by regexp** | `%R` | (**dired-do-rename-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME) | Rename selected files whose names match REGEXP to NEWNAME.<br>• 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.<br>• 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.<br>• 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. |
| **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.<br>See function 'dired-do-rename-regexp' for more info. |
| **Flag for deletion files selected by regexp** | `%d` | (**dired-flag-files-regexp** REGEXP) | In Dired, flag all files containing the specified REGEXP for deletion.<br>• The match is against the non-directory part of the filename. Use '^'  and '$' to anchor matches.  Exclude subdirs by hiding them.<br>• '.' and '..' are never flagged. |
| **Flag all files with content matching regexp** | `%g` | (**dired-mark-files-containing-regexp** REGEXP &optional MARKER-CHAR) | Mark all files with contents containing REGEXP for use in later commands.<br>• A prefix argument means to unmark them instead.<br>• '.' and '..' are never marked.<br>• Note that if a file is visited in an Emacs buffer, and 'dired-always-read-filesystem' is nil, this command will look in the buffer without revisiting the file, so the results might be inconsistent with the file on disk if its contents has changed since it was last visited. |
| **Rename all marked files to lowercase names** | `%l` | (**dired-downcase** &optional ARG) | Rename all marked (or next ARG) files to lower case. |
| **Mark all files with names matching regexp** | `%m` | (**dired-mark-files-regexp** REGEXP &optional MARKER-CHAR) | Mark all files matching REGEXP for use in later commands.<br>• A prefix argument means to unmark them instead.<br>• '.' and '..' are never marked.<br>• REGEXP is an Emacs regexp, not a shell wildcard.  Thus, use '\.o$' for object files--just '.o' will mark more than you might think. |
| **Rename all files matching regexp** | `%r` | (**dired-do-rename-regexp** REGEXP NEWNAME &optional ARG WHOLE-NAME) | Rename selected files whose names match REGEXP to NEWNAME.<br>• 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.<br>• 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.<br>• With a zero prefix arg, renaming by regexp affects the absolute file name.<br>• Normally, only the non-directory part of the file name is used and hanged. |
| **Rename all marked files to uppercase names** | `%u` | (**dired-upcase** &optional ARG) | Rename all marked (or next ARG) files to upper case. |
| **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.<br>📦 The following commands require the external  dired-narrow package to be installed.<br>🔧 This PEL bindings below are mode sensitive and are only available when the ***pel-use-dired-narrow*** customization variable is **t**. | |
| **Narrow the file list to specific filter** | **`<f12> s`** | (**dired-narrow**) | Narrow a dired buffer to the files matching a string. (not a regexp)<br>• 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.<br>• 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. |
| **Narrow the file list to specific fuzzy filter** | **`<f12> f`** | (**dired-narrow-fuzzy**) | Narrow a dired buffer to the files matching a fuzzy string.<br><br>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. |

**References**

| Description & link | Notes |
|---|---|
| **Dired @ wikipedia** | An overview. |
| **Dired in Emacs Manual** | Dired table of Content in the emacs 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. |