Grep

Description	Keystroke	Function	Note
Grep under Emacs	Emacs support several find and grep commands that can be executed from within Emacs. Doing so has several advantages: • The command output is collected in the *grep* Emacs buffer while the command runs asynchronously. The buffer is read-only, you can search within it right away. If you type <RET> on a found line, Emacs visit the file of the match at the appropriate line. • Without even going inside the search result buffer you can type one of the 'goto match' commands to move to the next or previous match: • (next-error) : `C-x `` ` ``, `M-g n` or `M-g M-n` • (previous_error): `M-g p` or `M-g M-p` • Stop the grep asynchronous operation with `C-c C-k` or `<f11> g k` • The search commands keep a history for the searches. You can browse the history at the prompt.		
Run grep via find See also: **∑ File mngt**	• `<f11> f g` • `<f11> g f`	(**find-grep** COMMAND-ARGS)	Run grep via find, with user-specified args COMMAND-ARGS. • Collect output in a buffer. • While find runs asynchronously, you can use the `C-x `` ` `` command to find the text that grep hits refer to. • This command uses a special history list for its arguments, so you can easily repeat a find command.
Run grep	`<f11> g g`	(**grep** COMMAND-ARGS)	Run Grep with user-specified COMMAND-ARGS, collect output in a buffer. • While Grep runs asynchronously, you can use `C-x `` ` `` (M-x next-error), or **`<RET>`** in the *grep* buffer, to go to the lines where Grep found matches. To kill the Grep job before it finishes, type `C-c C-k`. • For doing a recursive 'grep', see the 'rgrep' command. For running Grep in a specific directory, see 'lgrep'. • This command uses a special history list for its COMMAND-ARGS, so you can easily repeat a grep command. • A prefix argument says to default the COMMAND-ARGS based on the current tag the cursor is over, substituting it into the last Grep command in the Grep command history (or into 'grep-command' if that history list is empty).
Local grep	`<f11> g l`	(**lgrep** REGEXP &optional FILES DIR CONFIRM)	Run grep, searching for REGEXP in FILES in directory DIR. • The search is limited to file names matching shell pattern FILES. • FILES may use abbreviations defined in 'grep-files-aliases', e.g. entering 'ch' is equivalent to '*.[ch]'. As whitespace triggers completion when entering a pattern, including it requires quoting, e.g. '**`C-q<space>`**'. • With `C-u` prefix, you can edit the constructed shell command line before it is executed. • With `C-u C-u` prefix, directly edit and run 'grep-command'. • Collect output in a buffer. While grep runs asynchronously, you can use `C-x `` ` `` (M-x next-error), or RET in the grep output buffer, to go to the lines where grep found matches. • This command shares argument histories with `<f11> g r` and `<f11> g g`.
Open RipGrep transient menu	• `<f11> g m` • `C-c s`	(**rg-menu**)	Shows the transient menu for rg. • While the menu is active all keys and mouse actions are controlled by the rg menu, even though point remains where it was previously. The menu list options and commands. Type the options first and include the dash in what you type. The select the command. ☛📦 Requires the rg.el package and ripgrep command line utility. ☛☑ PEL activates this when the **pel-use-ripgrep** user option is set to **t**.
Recursive regexp grep over files in directory tree using RipGrep	• `<f11> g i` • *`C-c s r`*	(**rg** QUERY FILES DIR)	Run ripgrep, searching for REGEXP in FILES in directory DIR. • The search is limited to file names matching shell pattern FILES. • FILES may use abbreviations defined in 'rg-custom-type-aliases' or ripgrep builtin type aliases, e.g. entering 'elisp' is equivalent to '*.el'. REGEXP is a regexp as defined by the ripgrep executable. • With `C-u` prefix (CONFIRM), you can edit the constructed shell command line before it is executed. ☛ Useful to add the -z option to search into compressed files (.zip, .el.gz, etc…). • Collect output in a buffer. While ripgrep runs asynchronously, you can use `C-x `` ` `` (M-x 'next-error'), or RET in the rg output buffer, to go to the lines where rg found matches. ☛📦 Requires the rg.el package and ripgrep command line utility. ☛☑ PEL activates this when the **pel-use-ripgrep** user option is set to **t**. ☝⚠️ With PEL, only `<f11> g i` forces loading of the rg package. At first the `C-s s r` key binding is not set. It will be as soon as rg is loaded (forced by the other bindings).
Recursive literal grep over files in directory tree using RipGrep	• `<f11> g I` • *`C-c s t`*	(**rg-literal** QUERY FILES DIR)	Run ripgrep, searching for literal PATTERN in FILES in directory DIR. • With C-u prefix (CONFIRM), you can edit the constructed shell command line before it is executed. ☛ Useful to add the -z option to search into compressed files (.zip, .el.gz, etc…). ☛📦 Requires the rg.el package and ripgrep command line utility. ☛☑ PEL activates this when the **pel-use-ripgrep** user option is set to **t**. ☝⚠️ With PEL, only `<f11> g I` forces loading of the rg package. At first the `C-s s t` key binding is not set. It will be as soon as rg is loaded (forced by the other bindings).
Recursive grep over files in directory tree using grep	`<f11> g r`	(**rgrep** REGEXP &optional FILES DIR CONFIRM)	Recursively grep for REGEXP in FILES in directory tree rooted at DIR. • The search is limited to file names matching shell pattern FILES. • FILES may use abbreviations defined in 'grep-files-aliases', e.g. entering 'ch' is equivalent to '*.[ch]'. As whitespace triggers completion when entering a pattern, including it requires quoting, e.g. '**`C-q<space>`**'. • With `C-u` prefix, you can edit the constructed shell command line before it is executed. With `C-u C-u` prefix, directly edit and run 'grep-find-command'. • Collect output in a buffer. While the recursive grep is running, you can use `C-x `` ` `` or `M-g n` (M-x next-error), or **`<RET>`** in the grep output buffer, to visit the lines where matches were found. • To kill the job before it finishes, type `C-c C-k`. • This command shares argument histories with M-x lgrep and M-x grep-find. • When called programmatically and FILES is nil, REGEXP is expected • to specify a command to run. As seen above, typing the **C-u** prefix before the command keystroke, we can modify the command line found, e.g. for adding a **| sort -V** so that output list the files in sorted order with lines number also sorted.
Recursive Gzip grep	`<f11> g z`	(**zrgrep** REGEXP &optional FILES DIR CONFIRM TEMPLATE)	Recursively grep for REGEXP in gzipped FILES in tree rooted at DIR. • Like 'rgrep' but uses 'zgrep' for 'grep-program', sets the default file name to '*.gz', and sets 'grep-highlight-matches' to 'always'.
Kill grep process	• `C-c C-k` • `<f11> g k`	(**kill-grep**)	Kill the grep process that runs asynchronously.