Rocky Enterprise Linux 9.2 Manual Pages on command 'nanorc.5'

$ man nanorc.5

NANORC(5) File Formats Manual NANORC(5)

NAME

nanorc - GNU nano's configuration file

DESCRIPTION

The nanorc files contain the default settings for nano, a small and friendly editor. During startup, if --rcfile is not given, nano will read two files: first the system-wide settings, from /etc/nanorc (the exact path might be different on your system), and then the user-spe? cific settings, either from ~/.nanorc or from $XDG_CON? FIG_HOME/nano/nanorc or from ~/.config/nano/nanorc, whichever is en? countered first. If --rcfile is given, nano will read just the speci? fied settings file.

NOTICE

Since version 4.0, nano by default:

? does not automatically hard-wrap lines that become overlong,

? includes the line below the title bar in the editing area,

? does linewise (smooth) scrolling.

To get the old, Pico behavior back, you can use set breaklonglines, set

emptyline, and set jumpyscrolling.

OPTIONS

The configuration file accepts a series of set and unset commands, which can be used to configure nano on startup without using command-line options. Additionally, there are some commands to define syntax highlighting and to rebind keys -- see the two separate sections on those. nano reads one command per line. All commands and keywords should be written in lowercase.

Options in nanorc files take precedence over nano's defaults, and com? mand-line options override nanorc settings. Also, options that do not take an argument are unset by default. So using the unset command is only needed when wanting to override a setting of the system's nanorc file in your own nanorc. Options that take an argument cannot be un? set.

Quotes inside the characters parameters below should not be escaped. The last double quote on the line will be seen as the closing quote.

The supported commands and arguments are:

set afterends

Make Ctrl+Right and Ctrl+Delete stop at word ends instead of begin? nings.

set allow_insecure_backup

When backing up files, allow the backup to succeed even if its per? missions can't be (re)set due to special OS considerations. You should NOT enable this option unless you are sure you need it.

set atblanks

When soft line wrapping is enabled, make it wrap lines at blank characters (tabs and spaces) instead of always at the edge of the screen.

set autoindent

Automatically indent a newly created line to the same number of tabs and/or spaces as the previous line (or as the next line if the pre? vious line is the beginning of a paragraph).

set backup

When saving a file, create a backup file by adding a tilde (~) to the file's name.

set backupdir directory

Make and keep not just one backup file, but make and keep a uniquely numbered one every time a file is saved -- when backups are enabled with set backup or --backup or -B. The uniquely numbered files are stored in the specified directory.

set boldtext

Use bold instead of reverse video for the title bar, status bar, key combos, function tags, line numbers, and selected text. This can be overridden by setting the options titlecolor, statuscolor, keycolor, functioncolor, numbercolor, and selectedcolor.

set bookstyle

When justifying, treat any line that starts with whitespace as the beginning of a paragraph (unless auto-indenting is on).

set brackets "characters"

Set the characters treated as closing brackets when justifying para? graphs. This may not include blank characters. Only closing punc? tuation (see set punct), optionally followed by the specified clos? ing brackets, can end sentences. The default value is "")>]}".

set breaklonglines

Automatically hard-wrap the current line when it becomes overlong.

set casesensitive

Do case-sensitive searches by default.

set constantshow

Constantly display the cursor position in the status bar. This overrides the option quickblank.

set cutfromcursor

Use cut-from-cursor-to-end-of-line by default, instead of cutting the whole line.

set emptyline

Do not use the line below the title bar, leaving it entirely blank.

set errorcolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the status bar when an error message is displayed. The default value is brightwhite,red. See set title? color for valid color names.

set fill number

Set the target width for justifying and automatic hard-wrapping at this number of columns. If the value is 0 or less, wrapping will occur at the width of the screen minus number columns, allowing the wrap point to vary along with the width of the screen if the screen is resized. The default value is -8.

set functioncolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the concise function descriptions in the two help lines at the bottom of the screen. See set titlecolor for more details.

set guidestripe number

Draw a vertical stripe at the given column, to help judge the width of the text. (The color of the stripe can be changed with set stripecolor.)

set historylog

Save the last hundred search strings and replacement strings and ex? ecuted commands, so they can be easily reused in later sessions.

set indicator

Display a "scrollbar" on the righthand side of the edit window. It shows the position of the viewport in the buffer and how much of the buffer is covered by the viewport.

set jumpyscrolling

Scroll the buffer contents per half-screen instead of per line.

set keycolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the shortcut key combos in the two help lines at the bottom of the screen. See set titlecolor for more details.

set linenumbers

Display line numbers to the left of the text area. (Any line with an anchor additionally gets a mark in the margin.)

set locking

Enable vim-style lock-files for when editing files.

set magic

When neither the file's name nor its first line give a clue, try us?
ing libmagic to determine the applicable syntax. (Calling libmagic
can be relatively time consuming. It is therefore not done by de?
fault.)

set matchbrackets "characters"

Specify the opening and closing brackets that can be found by
bracket searches. This may not include blank characters. The open?
ing set must come before the closing set, and the two sets must be
in the same order. The default value is "(<[{)>]}".

set minibar

Suppress the title bar and instead show information about the cur?
rent buffer at the bottom of the screen, in the space for the status
bar. In this "minibar" the file name is shown on the left, followed
by an asterisk if the buffer has been modified. On the right are
displayed the current line and column number, the code of the char?
acter under the cursor (in Unicode format: U+xxxx), the same flags
as are shown by set stateflags, and a percentage that expresses how
far the cursor is into the file (linewise). When a file is loaded
or saved, and also when switching between buffers, the number of
lines in the buffer is displayed after the file name. This number
is cleared upon the next keystroke, or replaced with an [i/n]
counter when multiple buffers are open. The line plus column num?
bers and the character code are displayed only when set constantshow
is used, and can be toggled on and off with M-C. The state flags
are displayed only when set stateflags is used.

set morespace

Deprecated option since it has become the default setting. When
needed, use unset emptyline instead.

set mouse

Enable mouse support, if available for your system. When enabled,

mouse clicks can be used to place the cursor, set the mark (with a double click), and execute shortcuts. The mouse will work in the X Window System, and on the console when gpm is running. Text can still be selected through dragging by holding down the Shift key.

set multibuffer

When reading in a file with ^R, insert it into a new buffer by de? fault.

set noconvert

Don't convert files from DOS/Mac format.

set nohelp

Don't display the two help lines at the bottom of the screen.

set nonewlines

Don't automatically add a newline when a text does not end with one. (This can cause you to save non-POSIX text files.)

set nopauses

Obsolete option. Ignored.

set nowrap

Deprecated option since it has become the default setting. When needed, use unset breaklonglines instead.

set numbercolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for line numbers. See set titlecolor for more details.

set operatingdir directory

nano will only read and write files inside directory and its subdi? rectories. Also, the current directory is changed to here, so files are inserted from this directory. By default, the operating direc? tory feature is turned off.

set positionlog

Save the cursor position of files between editing sessions. The cursor position is remembered for the 200 most-recently edited files.

set preserve

Preserve the XON and XOFF keys (^Q and ^S).

set promptcolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the prompt bar. (When this option is not specified, the colors of the title bar are used.) See set ti? tlecolor for more details.

set punct "characters"

Set the characters treated as closing punctuation when justifying paragraphs. This may not include blank characters. Only the spec? fified closing punctuation, optionally followed by closing brackets (see brackets), can end sentences. The default value is "!.?".

set quickblank

Make status-bar messages disappear after 1 keystroke instead of af? ter 20. Note that options constantshow and minibar override this.

set quotestr "regex"

Set the regular expression for matching the quoting part of a line. The default value is "^([\t]*([!#%:;>|}]|//))+". (Note that \t stands for an actual Tab character.) This makes it possible to re? justify blocks of quoted text when composing email, and to rewrap blocks of line comments when writing source code.

set rawsequences

Interpret escape sequences directly (instead of asking ncurses to translate them). If you need this option to get your keyboard to work properly, please report a bug. Using this option disables nano's mouse support.

set rebinddelete

Interpret the Delete and Backspace keys differently so that both Backspace and Delete work properly. You should only use this option when on your system either Backspace acts like Delete or Delete acts like Backspace.

set regexp

Do regular-expression searches by default. Regular expressions in nano are of the extended type (ERE).

set saveonexit

Save a changed buffer automatically on exit (^X); don't prompt.

(The old form of this option, set tempfile, is deprecated.)

set scrollercolor fgcolor,bgcolor

Use this color combination for the indicator alias "scrollbar". (On terminal emulators that link to a libvte older than version 0.55, using a background color here does not work correctly.) See set ti? tlecolor for more details.

set selectedcolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for selected text. See set titlecolor for more details.

set showcursor

Put the cursor on the highlighted item in the file browser, to aid braille users.

set smarthome

Make the Home key smarter. When Home is pressed anywhere but at the very beginning of non-whitespace characters on a line, the cursor will jump to that beginning (either forwards or backwards). If the cursor is already at that position, it will jump to the true begin? ning of the line.

set smooth

Deprecated option since it has become the default setting. When needed, use unset jumpyscrolling instead.

set softwrap

Display lines that exceed the screen's width over multiple screen lines. (You can make this soft-wrapping occur at whitespace instead of rudely at the screen's edge, by using also set atblanks.)

set speller "program [argument ...]"

Use the given program to do spell checking and correcting, instead of using the built-in corrector that calls hunspell(1) or spell(1).

set spotlightcolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for highlighting a search match. The de? fault value is black,lightyellow. See set titlecolor for valid color names.

set stateflags

Use the top-right corner of the screen for showing some state flags: I when auto-indenting, M when the mark is on, L when hard-wrapping (breaking long lines), R when recording a macro, and S when soft-wrapping. When the buffer is modified, a star (*) is shown after the filename in the center of the title bar.

set statuscolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the status bar. See set titlecolor for more details.

set stripecolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the vertical guiding stripe. See set titlecolor for more details.

set suspendable

Allow nano to be suspended (with ^Z by default).

set tabsize number

Use a tab size of number columns. The value of number must be greater than 0. The default value is 8.

set tabstospaces

Convert typed tabs to spaces.

set titlecolor [bold,][italic,]fgcolor,bgcolor

Use this color combination for the title bar. Valid names for the foreground and background colors are: red, green, blue, magenta, yellow, cyan, white, and black. Each of these eight names may be prefixed with the word light to get a brighter version of that color. On terminal emulators that can do at least 256 colors, other valid (but unprefixable) color names are: pink, purple, mauve, la?goon, mint, lime, peach, orange, latte, and normal -- where normal means the default foreground or background color. Either "fgcolor" or ",bgcolor" may be left out, and the pair may be preceded by bold and/or italic (separated by commas) to get a bold and/or slanting typeface, if your terminal can do those.

set trimblanks

Remove trailing whitespace from wrapped lines when automatic hard-wrapping occurs or when text is justified.

set unix

Save a file by default in Unix format. This overrides nano's de? fault behavior of saving a file in the format that it had. (This option has no effect when you also use set noconvert.)

set whitespace "characters"

Set the two characters used to indicate the presence of tabs and spaces. They must be single-column characters. The default pair for a UTF-8 locale is "??", and for other locales ">.".

set wordbounds

Detect word boundaries differently by treating punctuation charac? ters as parts of words.

set wordchars "characters"

Specify which other characters (besides the normal alphanumeric ones) should be considered as parts of words. When using this op? tion, you probably want to unset wordbounds.

set zap

Let an unmodified Backspace or Delete erase the marked region (in? stead of a single character, and without affecting the cutbuffer).

SYNTAX HIGHLIGHTING

Coloring the different syntactic elements of a file is done via regular expressions (see the color command below). This is inherently imper? fect, because regular expressions are not powerful enough to fully parse a file. Nevertheless, regular expressions can do a lot and are easy to make, so they are a good fit for a small editor like nano.

All regular expressions in nano are POSIX extended regular expressions. This means that ., ?, *, +, ^, $, and several other characters are spe? cial. The period . matches any single character, ? means the preceding item is optional, * means the preceding item may be matched zero or more times, + means the preceding item must be matched one or more times, ^ matches the beginning of a line, and $ the end, \< matches the start of a word, and \> the end, and \s matches a blank. It also means that lookahead and lookbehind are not possible. A complete explanation can be found in the manual page of GNU grep: man grep.

For each kind of file a separate syntax can be defined via the follow?
ing commands:

syntax name ["fileregex" ...]

Start the definition of a syntax with this name. All subsequent color and other such commands will be added to this syntax, un? til a new syntax command is encountered.

When nano is run, this syntax will be automatically activated if the current filename matches the extended regular expression fileregex. Or the syntax can be explicitly activated by using the -Y or --syntax command-line option followed by the name.

The syntax default is special: it takes no fileregex, and ap? plies to files that don't match any syntax's regexes. The syn? tax none is reserved; specifying it on the command line is the same as not having a syntax at all.

header "regex" ...

If from all defined syntaxes no fileregex matched, then compare this regex (or regexes) against the first line of the current file, to determine whether this syntax should be used for it.

magic "regex" ...

If no fileregex matched and no header regex matched either, then compare this regex (or regexes) against the result of querying the magic database about the current file, to determine whether this syntax should be used for it. (This functionality only works when libmagic is installed on the system and will be silently ignored otherwise.)

formatter program [argument ...]

Run the given program on the full contents of the current buf? fer. (The current buffer is written out to a temporary file, the program is run on it, and then the temporary file is read back in, replacing the contents of the buffer.)

linter program [argument ...]

Use the given program to run a syntax check on the current buf? fer.

comment "string"

Use the given string for commenting and uncommenting lines. If the string contains a vertical bar or pipe character (|), this designates bracket-style comments; for example, "/*|*/" for CSS files. The characters before the pipe are prepended to the line and the characters after the pipe are appended at the end of the line. If no pipe character is present, the full string is prepended; for example, "#" for Python files. If empty double quotes are specified, the comment/uncomment function is dis? abled; for example, "" for JSON. The default value is "#".

tabgives "string"

Make the <Tab> key produce the given string. Useful for lan? guages like Python that want to see only spaces for indentation. This overrides the setting of the tabstospaces option.

color [bold,][italic,]fgcolor,bgcolor "regex" ...

Paint all pieces of text that match the extended regular expres? sion regex with the given foreground and background colors, at least one of which must be specified. Valid color names are: red, green, blue, magenta, yellow, cyan, white, and black. Each of these eight names may be prefixed with the word light to get a brighter version of that color. On terminal emulators that can do at least 256 colors, other valid (but unprefixable) color names are: pink, purple, mauve, lagoon, mint, lime, peach, or? ange, latte, and normal -- where normal means the default fore? ground or background color. The color pair may be preceded by bold and/or italic (separated by commas) to get a bold and/or slanting typeface, if your terminal can do those.

All coloring commands are applied in the order in which they are specified, which means that later commands can recolor stuff that was colored earlier.

icolor [bold,][italic,]fgcolor,bgcolor "regex" ...

Same as above, except that the matching is case insensitive.

color [bold,][italic,]fgcolor,bgcolor start="fromrx" end="torx"

Paint all pieces of text whose start matches extended regular expression fromrx and whose end matches extended regular expres? sion torx with the given foreground and background colors, at least one of which must be specified. This means that, after an initial instance of fromrx, all text until the first instance of torx will be colored. This allows syntax highlighting to span multiple lines.

icolor [bold,][italic,]fgcolor,bgcolor start="fromrx" end="torx"

Same as above, except that the matching is case insensitive.

include "syntaxfile"

Read in self-contained color syntaxes from syntaxfile. Note that syntaxfile may contain only the above commands, from syntax to icolor.

extendsyntax name command argument ...

Extend the syntax previously defined as name with another com? mand. This allows adding a new color, icolor, header, magic, formatter, linter, comment, or tabgives command to an already defined syntax -- useful when you want to slightly improve a syntax defined in one of the system-installed files (which nor? mally are not writable).

REBINDING KEYS

Key bindings can be changed via the following three commands:

bind key function menu

Rebinds the given key to the given function in the given menu (or in all menus where the function exists when all is used).

bind key "string" menu

Makes the given key produce the given string in the given menu (or in all menus where the key exists when all is used). The string can consist of text or commands or a mix of them. (To enter a command into the string, precede its keystroke with M-V.)

unbind key menu

Unbinds the given key from the given menu (or from all menus

where the key exists when all is used).

The format of key should be one of:

^X where X is a Latin letter, or one of several ASCII characters (@,], \, ^, _), or the word "Space". Example: ^C.

M-X where X is any ASCII character except [, or the word "Space". Example: M-8.

Sh-M-X where X is a Latin letter. Example: Sh-M-U. By default, each Meta+letter keystroke does the same as the corresponding Shift+Meta+letter. But when any Shift+Meta bind is made, that will no longer be the case, for all letters.

FN where N is a numeric value from 1 to 24. Example: F10. (Of? ten, F13 to F24 can be typed as F1 to F12 with Shift.)

Ins or Del.

Rebinding ^M (Enter) or ^I (Tab) is probably not a good idea. Rebind? ing ^[(Esc) is not possible, because its keycode is the starter byte of Meta keystrokes and escape sequences. Rebinding any of the dedi? cated cursor-moving keys (the arrows, Home, End, PageUp and PageDown) is not possible. On some terminals it's not possible to rebind ^H (un? less --raw is used) because its keycode is identical to that of the Backspace key.

Valid function names to be bound are:

help

Invokes the help viewer.

cancel

Cancels the current command.

exit

Exits from the program (or from the help viewer or file browser).

writeout

Writes the current buffer to disk, asking for a name.

savefile

Writes the current file to disk without prompting.

insert

Inserts a file into the current buffer (at the current cursor po?

sition), or into a new buffer when option multibuffer is set.

whereis

Starts a forward search for text in the current buffer -- or for filenames matching a string in the current list in the file browser.

wherewas

Starts a backward search for text in the current buffer -- or for filenames matching a string in the current list in the file browser.

findprevious

Searches the next occurrence in the backward direction.

findnext

Searches the next occurrence in the forward direction.

replace

Interactively replaces text within the current buffer.

cut

Cuts and stores the current line (or the marked region).

copy

Copies the current line (or the marked region) without deleting it.

paste

Pastes the currently stored text into the current buffer at the current cursor position.

zap

Throws away the current line (or the marked region). (This func? tion is bound by default to <Meta+Delete>.)

chopwordleft

Deletes from the cursor position to the beginning of the preceding word. (This function is bound by default to <Shift+Ctrl+Delete>. If your terminal produces ^H for <Ctrl+Backspace>, you can make <Ctrl+Backspace> delete the word to the left of the cursor by re? binding ^H to this function.)

chopwordright

Deletes from the cursor position to the beginning of the next word. (This function is bound by default to <Ctrl+Delete>.)

cutrestoffile

Cuts all text from the cursor position till the end of the buffer.

mark

Sets the mark at the current position, to start selecting text. Or, when it is set, unsets the mark.

location

Reports the current position of the cursor in the buffer: the line, column, and character positions. (The old name of this function, 'curpos', is deprecated.)

wordcount

Counts the number of words, lines and characters in the current buffer.

execute

Prompts for a program to execute. The program's output will be inserted into the current buffer (or into a new buffer when M-F is toggled).

speller

Invokes a spell-checking program, either the default hunspell(1) or GNU spell(1), or the one defined by --speller or set speller.

formatter

Invokes a full-buffer-processing program (if the active syntax de? fines one).

linter

Invokes a syntax-checking program (if the active syntax defines one).

justify

Justifies the current paragraph. A paragraph is a group of con? tiguous lines that, apart from possibly the first line, all have the same indentation. The beginning of a paragraph is detected by either this lone line with a differing indentation or by a preced? ing blank line.

fulljustify

Justifies the entire current buffer.

indent

Indents (shifts to the right) the currently marked text.

unindent

Unindents (shifts to the left) the currently marked text.

comment

Comments or uncomments the current line or marked lines, using the comment style specified in the active syntax.

complete

Completes the fragment before the cursor to a full word found elsewhere in the current buffer.

left

Goes left one position (in the editor or browser).

right

Goes right one position (in the editor or browser).

up

Goes one line up (in the editor or browser).

down

Goes one line down (in the editor or browser).

scrollup

Scrolls the viewport up one row (meaning that the text slides down) while keeping the cursor in the same text position, if pos? sible.

scrolldown

Scrolls the viewport down one row (meaning that the text slides up) while keeping the cursor in the same text position, if possi? ble.

center

Scrolls the line with the cursor to the middle of the screen.

prevword

Moves the cursor to the beginning of the previous word.

nextword

Moves the cursor to the beginning of the next word.

home

Moves the cursor to the beginning of the current line.

end

Moves the cursor to the end of the current line.

beginpara

Moves the cursor to the beginning of the current paragraph.

endpara

Moves the cursor to the end of the current paragraph.

prevblock

Moves the cursor to the beginning of the current or preceding block of text. (Blocks are separated by one or more blank lines.)

nextblock

Moves the cursor to the beginning of the next block of text.

pageup

Goes up one screenful.

pagedown

Goes down one screenful.

firstline

Goes to the first line of the file.

lastline

Goes to the last line of the file.

gotoline

Goes to a specific line (and column if specified). Negative numbers count from the end of the file (and end of the line).

findbracket

Moves the cursor to the bracket (or brace or parenthesis, etc.) that matches (pairs) with the one under the cursor. See set matchbrackets.

anchor

Places an anchor at the current line, or removes it when already present. (An anchor is visible when line numbers are activated.)

prevanchor

Goes to the first anchor before the current line.

nextanchor

Goes to the first anchor after the current line.

prevbuf

Switches to editing/viewing the previous buffer when multiple buf?

fers are open.

nextbuf

Switches to editing/viewing the next buffer when multiple buffers

are open.

verbatim

Inserts the next keystroke verbatim into the file.

tab

Inserts a tab at the current cursor location.

enter

Inserts a new line below the current one.

delete

Deletes the character under the cursor.

backspace

Deletes the character before the cursor.

recordmacro

Starts the recording of keystrokes -- the keystrokes are stored as

a macro. When already recording, the recording is stopped.

runmacro

Replays the keystrokes of the last recorded macro.

undo

Undoes the last performed text action (add text, delete text,

etc).

redo

Redoes the last undone action (i.e., it undoes an undo).

refresh

Refreshes the screen.

suspend

Suspends the editor (if the suspending function is enabled, see

the suspendable toggle item below).

casesens

Toggles whether searching/replacing ignores or respects the case of the given characters.

regexp

Toggles whether searching/replacing uses literal strings or regular expressions.

backwards

Toggles whether searching/replacing goes forward or backward.

older

Retrieves the previous (earlier) entry at a prompt.

newer

Retrieves the next (later) entry at a prompt.

flipreplace

Toggles between searching for something and replacing something.

flipgoto

Toggles between searching for text and targeting a line number.

flipexecute

Toggles between inserting a file and executing a command.

flippipe

When executing a command, toggles whether the current buffer (or marked region) is piped to the command.

flipnewbuffer

Toggles between inserting into the current buffer and into a new empty buffer.

flipconvert

When reading in a file, toggles between converting and not converting it from DOS/Mac format. Converting is the default.

dosformat

When writing a file, switches to writing a DOS format (CR/LF).

macformat

When writing a file, switches to writing a Mac format.

append

When writing a file, appends to the end instead of overwriting.

prepend

When writing a file, 'prepends' (writes at the beginning) instead of overwriting.

backup

When writing a file, creates a backup of the current file.

discardbuffer

When about to write a file, discard the current buffer without saving. (This function is bound by default only when option --saveonexit is in effect.)

browser

Starts the file browser (in the Read File and Write Out menus), allowing to select a file from a list.

gotodir

Goes to a directory to be specified, allowing to browse anywhere in the filesystem.

firstfile

Goes to the first file in the list when using the file browser.

lastfile

Goes to the last file in the list when using the file browser.

nohelp

Toggles the presence of the two-line list of key bindings at the bottom of the screen. (This toggle is special: it is available in all menus except the help viewer and the linter. All further tog? gles are available in the main menu only.)

constantshow

Toggles the constant display of the current line, column, and character positions.

softwrap

Toggles the displaying of overlong lines on multiple screen lines.

linenumbers

Toggles the display of line numbers in front of the text.

whitespacedisplay

Toggles the showing of whitespace.

nosyntax

Toggles syntax highlighting.

smarthome

Toggles the smartness of the Home key.

autoindent

Toggles whether a newly created line will contain the same amount of leading whitespace as the preceding line -- or as the next line if the preceding line is the beginning of a paragraph.

cutfromcursor

Toggles whether cutting text will cut the whole line or just from the current cursor position to the end of the line.

breaklonglines

Toggles whether long lines will be hard-wrapped to the next line. (The old name of this function, 'nowrap', is deprecated.)

tabstospaces

Toggles whether typed tabs will be converted to spaces.

mouse

Toggles mouse support.

suspendable

Toggles whether the suspend keystroke (^Z by default) will actu? ally suspend the editor. (The old name of this function, 'sus? pendenable', is deprecated.)

Valid menu sections are:

main

The main editor window where text is entered and edited.

help

The help-viewer menu.

search

The search menu (AKA whereis).

replace

The 'search to replace' menu.

replacewith

The 'replace with' menu, which comes up after 'search to replace'.

yesno

The 'yesno' menu, where the Yes/No/All/Cancel question is asked.

gotoline

The 'goto line (and column)' menu.

writeout

The 'write file' menu.

insert

The 'insert file' menu.

browser

The 'file browser' menu, for selecting a file to be opened or in? serted or written to.

whereisfile

The 'search for a file' menu in the file browser.

gotodir

The 'go to directory' menu in the file browser.

execute

The menu for inserting the output from an external command, or for filtering the buffer (or the marked region) through an external command, or for executing one of several tools. (The old form of this menu name, 'extcmd', is deprecated.)

spell

The menu of the integrated spell checker where the user can edit a misspelled word.

linter

The linter menu, which allows jumping through the linting mes? sages.

all

A special name that encompasses all menus. For bind it means all menus where the specified function exists; for unbind it means all menus where the specified key exists.

FILES

/etc/nanorc

System-wide configuration file.

~/.nanorc or $XDG_CONFIG_HOME/nano/nanorc or ~/.config/nano/nanorc

Per-user configuration file.

/usr/share/nano/*

Syntax definitions for the syntax coloring of common file types

(and for less common file types in the extra/ subdirectory).

SEE ALSO

nano(1)

https://nano-editor.org/cheatsheet.html

An overview of the default key bindings.

March 2021 version 5.6.1 NANORC(5)