GNU nano

a small and friendly text editor.
version 0.9.99pre3

Chris Allegretta

This manual documents GNU `nano`, a small and friendly text editor.

This manual is part of the GNU `nano` distribution.

You may contact the author by: e-mail: `chrisa@asty.org`

This manual documents GNU nano, a small and friendly text editor.

1 Introduction

GNU `nano` is a small and friendly text editor. Besides basic text editing, `nano` offers many extra features like an interactive search and replace, goto line number, auto-indentation, feature toggles, internationalization support, and filename tab completion.

1.1 Overview

`nano` [GNU long option] [option] +LINE [*file ...*]

The original goal for `nano` was a complete bug-for-bug compatible emulation of Pico, but consistency is now a slightly higher priority. There is a flag to implement (nearly) complete Pico emulation, (option -p or GNU long option –pico). This can also be toggled from within `nano` by typing Meta-P. See Chapter 6 [Pico Compatibility], page 9, for more info.

Email bug reports to `nano@nano-editor.org`.

1.2 Command Line Options

`nano` takes the following options from the command line:

`-T [num, --tabsize=[num]`
: Set the displayed tab length to [num] columns.

`-R, --regexp`
: Turns on regular expression search and search/replace.

`-V, --version`
: Print the version number and copyright and quit.

`-c, --const`
: Constantly display the cursor posititon and line number on the statusbar.

`-h, --help`
: Print the usage and exit.

`-i, --autoindent`
: Automatically indent new lines to the same number of spaces and tabs as the previous line.

`-k, --cut`
: Makes ^K cut from the current cursor position to the end of the current line.

`-l, --nofollow`
: When writing files, if the given file is a symbolic link it is removed and a new file is created.

`-m, --mouse`
: Enables the use of the mouse to select text (currently only useful for running under the X window system).

`-p, --pico`
: Emulate Pico as closely as possible, sacrificing consistency for correct emulation. See Chapter 6 [Pico Compatibility], page 9, for more info.

`-r [#cols], --fill=[#cols].`
: Wrap lines at column #cols. By default this is the width of the screen, less eight.

`-s [prog], --speller=[prog]`
: Invoke [prog] as the spell checker. By default, `nano` uses its own interactive spell checker that requires the `spell` program be installed on your system.

`-t, --tempfile`
: Do not ask whether or not to save the current contents of the file when exiting, assume yes. This is most useful when using `nano` as the composer of a mailer program.

`-x, --nohelp`
: In Expert Mode, the Shortcut Lists will not appear at the bottom of the screen. This affects the location of the statusbar as well, as in Expert Mode it is located at the very bottom of the editor.

  Note: When accesing the help system, Expert Mode is temporarily disabled to display the help system navigation keys.

`-v, --view`
: Do not allow the contents of the file to be altered. Note that this flag should NOT be used in place of correct file permissions to implement a read-only file.

`-w, --nowrap`
: Do not wrap long lines at any length. This option overrides any value for -r.

`-z, --suspend`
: Enable suspend ability of `nano` using the system's suspend keystroke (usually ^Z).

`+LINE`
: Start at line number LINE instead of the default of line 1.

2 Editor Basics

2.1 Entering Text

All key sequences in `nano` are entered using the keyboard. `nano` is a "modeless" editor, all keys with the exception of Control and Meta key sequences will enter text into the file being edited.

2.2 Special Functions

Special functions use the Control key (displayed in the help and shotcut lists as ^) or the Meta key (displayed as M).

- Control key sequences are entered by holding down the Control key and pressing the desired letter.
- Meta key sequences can be entered in a number of possible ways: Pressing the Escape key, then releasing it and pressing the desired key, or holding down the Alt key while pressing the desired key. This varies from keyboard to keyboard, and certain commercial operating systems "swallow" the Alt key so that it never reaches the application. If your operating system does this, you should use the Escape key to generate Meta key sequences.

2.3 The Titlebar

The titlebar is the line displayed at the top of the editor. There are three sections: left, center and right. The section on the left displays the version of `nano` being used. The center section displays the current file name, or "New Buffer" if the file has not yet been named. The section on the right will display "Modified" if the file has been modified since it ws last saved or opened.

Special modes: When nano is in "File browser" mode, the center section will display the current directory instead of the filename. See Chapter 5 [The File Browser], page 8.

2.4 The Statusbar

The statusbar is located three lines from the bottom of the screen (or the bottom line in Expert Mode. See [Expert Mode], page 3, for more info.

The Statusbar shows important and informational messages. Any error messages that occur from using the editor will appear on the statusbar. Any questions that are asked of the user will be asked on the statusbar, and any user input (serch strings, file names, etc) will be input on the statusbar.

2.5 Shortcut Lists

The Shorcut Lists are the two lines at the bottom of the screen which show some of the more commonly used functions in the editor. The exact functions which are displayed depend on whether Pico Compatibility Mode mode is enabled. See Chapter 6 [Pico Compatibility], page 9, for more info.

3 Online Help

The online help system in `nano` is available by pressing ^G. It is fairly self explanatory, documenting the various parts of the editor and available keystrokes. Navigation is via the ^Y (Page Up) and ^V (Page Down) keys. ^X exits the help system.

4 Feature Toggles

Toggles allow you to change certain aspects of the editor that would normally be done via command line flags. They are invoked via certain Meta key sequenced. See Section 2.2 [Special Functions], page 4, for more info. The following toggles are available:

`Constant Update Toggle (Meta-C)`
: toggles the -c (–const) command line flag.

`Regular Expressions Toggle (Meta-E)`
: toggles the -R (–regexp) command line flag.

`AutoIndent Toggle (Meta-I)`
: toggles the -i (–autoindent) command line flag.

`Cut To End Toggle (Meta-K)`
: toggles the -k (–cut) command line flag.

`Cut To End Toggle (Meta-M)`
: toggles the -m (–mouse) command line flag.

`Pico Mode Toggle (Meta-P)`
: toggles the -p (–pico) command line flag. See Chapter 6 [Pico Compatibility], page 9, for more info.

`AutoWrap Toggle (Meta-W)`
: toggles the -w (–nowrap) command line flag.

`Expert/Nohelp Toggle (Meta-X)`
: toggles the -x (–nohelp) command line flag.

`Suspend Toggle (Meta-Z)`
: toggles the -z (–suspend) command line flag.

5 The File Browser

When reading or writilg files, pressing ^T will invoke the file browser. Here, one can navigate directories in a graphical manner in order to find the desired file.

Basic movement in the file browser is accomplished with he arrow keys and page up/down. The behavior of the enter (or 's') key varies by what is currently selected. If the currently selected object is a directory, the file browser will enter and display the contects of the directory. If the object is a file, this filename and path are copied to the statusbar and the file browser is exited.

6 Pico Compatibility

Nano does not completely emulate Pico by default. The following differences apply to the default mode and Pico Compatibility mode:

`Displayed Shortcuts`

By default, the following shortcuts are displayed in the Shortcut Lists:

^G ^O ^\ ^Y ^K ^C
^X ^R ^W ^V ^U ^T

Related functions are listed above or below each other by default. The Justify function is not listed, instead the "Replace" function is displayed. Also, he "Read File" and "WriteOut" functions are aligned for consistency.

In Pico Compatibility mode, the default Pico shortcuts are displayed:

^G ^O ^R ^Y ^K ^C
^X ^J ^W ^V ^U ^T

`Previous String Text`

By default, previously entered string for a function (search string, file name) will be placed on the statusbar, and is editable. This is done so there is consistency across all functions. For example: even if there is a previous replace string, it can always be deleted if one wishes to perform an empty string replace.

In Pico Compatibility Mode, the previously entered text in a search or replace will appear in brackets, and is not editable. It is not a simple matter to do an empty string replace when a previous replace string exists, for example. When writing a file, the previous filename will be displayed in the editable text portion of the editor.

`Interactive Replace and Spell Checker`

It is worth noting that the `nano` replace function is interactive, i.e. it does not stop after one search string is found and automatically replace it. The `nano` implementation will stop at each search string found and query whether to replace this instance or not. The internal spell checker operates similarly. Note that these is no way to force these functions to behave in the Pico fashion.

7 Building and Configure Options

Building nano from source is fairly straightforward if you are familiar with compiling programs with autoconf support:

tar xvfz nano-x.y.z.tar.gz (where x.y.z is the version of nano) cd nano-x.y.z/ ./configure make make install

if you are looking to optimize nano for size, you may want to consider the following command line options:

`--disable-tabcomp`
: Disable the tab completion code when reading or writing files.

`--disable-justify`
: Disable the justify (^J)/unjustify (^U) functions in the editor.

`--disable-speller`
: Disables spell checker abillity.

`--disable-help`
: Disables the help function (^G). Disabling this option maks the binary much smaller, but makes it difficult for new users to learn more than very basic things about using the editor.

`--disable-browser`
: Disables the mini file browser (^O) when reading or writing files.

`--enable-tiny`
: This options disables all the above. It also disables some of the larger internals of the editor, like the marker code (^^) and the cut to line (-k) option which it depends on to work properly. It also disables the function toggles and mouse support.

Table of Contents