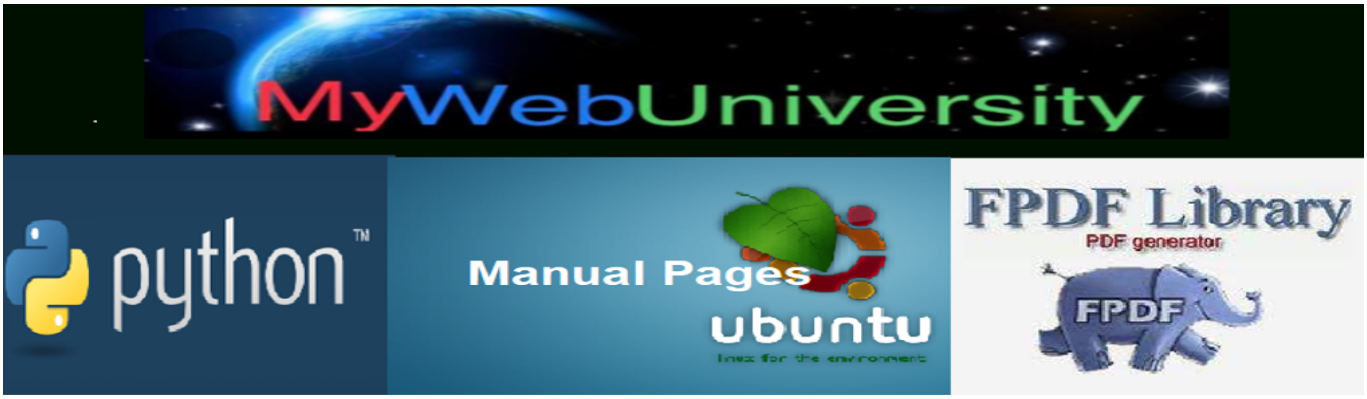

Rocky Enterprise Linux 9.2 Manual Pages on command 'infocmp.1'

$ man infocmp.1

infocmp(1) General Commands Manual infocmp(1)

NAME

infocmp - compare or print out terminfo descriptions

SYNOPSIS

infocmp [-1CDEFGIKLTUVWcdegilnpqrtux]

[-v n] [-s d| i| l| c] [-Q n] [-R subset]

[-w width] [-A directory] [-B directory]

[termname...]

DESCRIPTION

infocmp can be used to compare a binary terminfo entry with other terminfo entries, re? write a terminfo description to take advantage of the use= terminfo field, or print out a terminfo description from the binary file (term) in a variety of formats. In all cases, the boolean fields will be printed first, followed by the numeric fields, followed by the string fields.

Default Options

If no options are specified and zero or one termnames are specified, the -I option will be assumed. If more than one termname is specified, the -d option will be assumed.

Comparison Options [-d] [-c] [-n]

infocmp compares the terminfo description of the first terminal termname with each of the descriptions given by the entries for the other terminal's termnames. If a capability is defined for only one of the terminals, the value returned depends on the type of the capa? bility:

? F for missing boolean variables

? NULL for missing integer or string variables

Use the -q option to show the distinction between absent and cancelled capabilities.

These options produce a list which you can use to compare two or more terminal descrip? tions:

-d produces a list of each capability that is different between two entries. Each item in the list shows ?:? after the capability name, followed by the capability values, separated by a comma.

-c produces a list of each capability that is common between two or more entries. Miss? ing capabilities are ignored. Each item in the list shows ?=? after the capability name, followed by the capability value.

The -u option provides a related output, showing the first terminal description rewritten to use the second as a building block via the ?use=? clause.

-n produces a list of each capability that is in none of the given entries. Each item in the list shows ?!? before the capability name.

Normally only the conventional capabilities are shown. Use the -x option to add the BSD-compatibility capabilities (names prefixed with ?OT?).

If no termnames are given, infocmp uses the environment variable TERM for each of the termnames.

Source Listing Options [-I] [-L] [-C] [-r]

The -I, -L, and -C options will produce a source listing for each terminal named.

-I use the terminfo names

-L use the long C variable name listed in <term.h>

-C use the termcap names

-r when using -C, put out all capabilities in termcap form

-K modifies the -C option, improving BSD-compatibility.

If no termnames are given, the environment variable TERM will be used for the terminal name.

The source produced by the -C option may be used directly as a termcap entry, but not all parameterized strings can be changed to the termcap format. infocmp will attempt to con? vert most of the parameterized information, and anything not converted will be plainly marked in the output and commented out. These should be edited by hand.

For best results when converting to termcap format, you should use both -C and -r. Nor? mally a termcap description is limited to 1023 bytes. infocmp trims away less essential

parts to make it fit. If you are converting to one of the (rare) termcap implementations which accept an unlimited size of termcap, you may want to add the -T option. More often however, you must help the termcap implementation, and trim excess whitespace (use the -0 option for that).

All padding information for strings will be collected together and placed at the beginning of the string where termcap expects it. Mandatory padding (padding information with a trailing ?/?) will become optional.

All termcap variables no longer supported by terminfo, but which are derivable from other terminfo variables, will be output. Not all terminfo capabilities will be translated; only those variables which were part of termcap will normally be output. Specifying the -r option will take off this restriction, allowing all capabilities to be output in term? cap form. Normally you would use both the -C and -r options. The actual format used in? corporates some improvements for escaped characters from terminfo format. For a stricter BSD-compatible translation, use the -K option rather than -C.

Note that because padding is collected to the beginning of the capability, not all capa? bilities are output. Mandatory padding is not supported. Because termcap strings are not as flexible, it is not always possible to convert a terminfo string capability into an equivalent termcap format. A subsequent conversion of the termcap file back into terminfo format will not necessarily reproduce the original terminfo source.

Some common terminfo parameter sequences, their termcap equivalents, and some terminal types which commonly have such sequences, are:

terminfo	termcap	Representative Terminals
%p1%c	%.	adm
%p1%d	%d	hp, ANSI standard, vt100
%p1%'x'%+%c	%+x	concept
%i	%iq	ANSI standard, vt100
%p1%?%'x'%>%t%p1%'y'%+%;	%>xy	concept
%p2 is printed before %p1	%r	hp

Use= Option [-u]

The -u option produces a terminfo source description of the first terminal termname which is relative to the sum of the descriptions given by the entries for the other terminals termnames. It does this by analyzing the differences between the first termname and the

other termnames and producing a description with use= fields for the other terminals. In this manner, it is possible to retrofit generic terminfo entries into a terminal's de? scription. Or, if two similar terminals exist, but were coded at different times or by different people so that each description is a full description, using infocmp will show what can be done to change one description to be relative to the other.

A capability will be printed with an at-sign (@) if it no longer exists in the first termname, but one of the other termname entries contains a value for it. A capability's value will be printed if the value in the first termname is not found in any of the other termname entries, or if the first of the other termname entries that has this capability gives a different value for the capability than that in the first termname.

The order of the other termname entries is significant. Since the terminfo compiler tic does a left-to-right scan of the capabilities, specifying two use= entries that contain differing entries for the same capabilities will produce different results depending on the order that the entries are given in. infocmp will flag any such inconsistencies be? tween the other termname entries as they are found.

Alternatively, specifying a capability after a use= entry that contains that capability will cause the second specification to be ignored. Using infocmp to recreate a descrip? tion can be a useful check to make sure that everything was specified correctly in the original source description.

Another error that does not cause incorrect compiled files, but will slow down the compi? lation time, is specifying extra use= fields that are superfluous. infocmp will flag any other termname use= fields that were not needed.

Changing Databases [-A directory] [-B directory]

Like other ncurses utilities, infocmp looks for the terminal descriptions in several places. You can use the TERMINFO and TERMINFO_DIRS environment variables to override the compiled-in default list of places to search (see curses(3X) for details).

You can also use the options -A and -B to override the list of places to search when com? paring terminal descriptions:

? The -A option sets the location for the first termname

? The -B option sets the location for the other termnames.

Using these options, it is possible to compare descriptions for a terminal with the same name located in two different databases. For instance, you can use this feature for com? paring descriptions for the same terminal created by different people.

Other Options

-0 causes the fields to be printed on one line, without wrapping.

-1 causes the fields to be printed out one to a line. Otherwise, the fields will be printed several to a line to a maximum width of 60 characters.

-a tells infocmp to retain commented-out capabilities rather than discarding them. Ca? pabilities are commented by prefixing them with a period.

-D tells infocmp to print the database locations that it knows about, and exit.

-E Dump the capabilities of the given terminal as tables, needed in the C initializer for a TERMTYPE structure (the terminal capability structure in the <term.h>). This option is useful for preparing versions of the curses library hardwired for a given terminal type. The tables are all declared static, and are named according to the type and the name of the corresponding terminal entry.

Before ncurses 5.0, the split between the -e and -E options was not needed; but sup? port for extended names required making the arrays of terminal capabilities separate from the TERMTYPE structure.

-e Dump the capabilities of the given terminal as a C initializer for a TERMTYPE struc? ture (the terminal capability structure in the <term.h>). This option is useful for preparing versions of the curses library hardwired for a given terminal type.

-F compare terminfo files. This assumes that two following arguments are filenames. The files are searched for pairwise matches between entries, with two entries consid? ered to match if any of their names do. The report printed to standard output lists entries with no matches in the other file, and entries with more than one match. For entries with exactly one match it includes a difference report. Normally, to reduce the volume of the report, use references are not resolved before looking for differ? ences, but resolution can be forced by also specifying -r.

-f Display complex terminfo strings which contain if/then/else/endif expressions in? dented for readability.

-G Display constant literals in decimal form rather than their character equivalents.

-g Display constant character literals in quoted form rather than their decimal equiva? lents.

-i Analyze the initialization (is1, is2, is3), and reset (rs1, rs2, rs3), strings in the entry, as well as those used for starting/stopping cursor-positioning mode (smcup, rmcup) as well as starting/stopping keymap mode (smkx, rmkx).

For each string, the code tries to analyze it into actions in terms of the other capabilities in the entry, certain X3.64/ISO 6429/ECMA-48 capabilities, and certain DEC VT-series private modes (the set of recognized special sequences has been selected for completeness over the existing terminfo database). Each report line consists of the capability name, followed by a colon and space, followed by a printable expansion of the capability string with sections matching recognized actions translated into {}-bracketed descriptions.

Here is a list of the DEC/ANSI special sequences recognized:

Action	Meaning
??????????????????????????????????????	
RIS	full reset
SC	save cursor
RC	restore cursor
LL	home-down
RSR	reset scroll region
??????????????????????????????????????	
DECSTR	soft reset (VT320)
S7C1T	7-bit controls (VT220)
??????????????????????????????????????	
ISO DEC G0	enable DEC graphics for G0
ISO UK G0	enable UK chars for G0
ISO US G0	enable US chars for G0
ISO DEC G1	enable DEC graphics for G1
ISO UK G1	enable UK chars for G1
ISO US G1	enable US chars for G1
??????????????????????????????????????	
DECPAM	application keypad mode
DECPNM	normal keypad mode
DECANSI	enter ANSI mode
??????????????????????????????????????	
ECMA[+-]AM	keyboard action mode
ECMA[+-]IRM	insert replace mode
ECMA[+-]SRM	send receive mode

ECMA[+-]LNM linefeed mode

??????????????????????????????????????

DEC[+-]CKM application cursor keys

DEC[+-]ANM set VT52 mode

DEC[+-]COLM 132-column mode

DEC[+-]SCLM smooth scroll

DEC[+-]SCNM reverse video mode

DEC[+-]OM origin mode

DEC[+-]AWM wraparound mode

DEC[+-]ARM auto-repeat mode

It also recognizes a SGR action corresponding to ANSI/ISO 6429/ECMA Set Graphics Ren? dition, with the values NORMAL, BOLD, UNDERLINE, BLINK, and REVERSE. All but NORMAL may be prefixed with

? ?+? (turn on) or

? ?-? (turn off).

An SGR0 designates an empty highlight sequence (equivalent to {SGR:NORMAL}).

-I Set output format to terminfo.

-p Ignore padding specifications when comparing strings.

-Q n Rather than show source in terminfo (text) format, print the compiled (binary) format in hexadecimal or base64 form, depending on the option's value:

1 hexadecimal

2 base64

3 hexadecimal and base64

For example, this prints the compiled terminfo value as a string which could be as? signed to the TERMINFO environment variable:

infocmp -0 -q -Q2

-q This makes the output a little shorter:

? Make the comparison listing shorter by omitting subheadings, and using ?-? for absent capabilities, ?@? for canceled rather than ?NULL?.

? However, show differences between absent and cancelled capabilities.

? Omit the ?Reconstructed from? comment for source listings.

-Rsubset

Restrict output to a given subset. This option is for use with archaic versions of

terminfo like those on SVr1, Ultrix, or HP-UX that do not support the full set of SVR4/XSI Curses terminfo; and variants such as AIX that have their own extensions in? compatible with SVr4/XSI.

? Available terminfo subsets are ?SVr1?, ?Ultrix?, ?HP?, and ?AIX?; see terminfo(5) for details.

? You can also choose the subset ?BSD? which selects only capabilities with termcap equivalents recognized by 4.4BSD. The -C option sets the ?BSD? subset as a side-effect.

? If you select any other value for -R, it is the same as no subset, i.e., all ca? pabilities are used. The -I option likewise selects no subset as a side-effect.

-s [d|i|l|c]

The -s option sorts the fields within each type according to the argument below:

d leave fields in the order that they are stored in the terminfo database.

i sort by terminfo name.

l sort by the long C variable name.

c sort by the termcap name.

If the -s option is not given, the fields printed out will be sorted alphabetically by the terminfo name within each type, except in the case of the -C or the -L op? tions, which cause the sorting to be done by the termcap name or the long C variable name, respectively.

-T eliminates size-restrictions on the generated text. This is mainly useful for test? ing and analysis, since the compiled descriptions are limited (e.g., 1023 for term? cap, 4096 for terminfo).

-t tells tic to discard commented-out capabilities. Normally when translating from ter? minfo to termcap, untranslatable capabilities are commented-out.

-U tells infocmp to not post-process the data after parsing the source file. This fea? ture helps when comparing the actual contents of two source files, since it excludes the inferences that infocmp makes to fill in missing data.

-V reports the version of ncurses which was used in this program, and exits.

-v n prints out tracing information on standard error as the program runs.

The optional parameter n is a number from 1 to 10, inclusive, indicating the desired level of detail of information. If ncurses is built without tracing support, the op? tional parameter is ignored.

-W By itself, the -w option will not force long strings to be wrapped. Use the -W op?
tion to do this.

-w width
changes the output to width characters.

-x print information for user-defined capabilities (see user_caps(5). These are exten?
sions to the terminfo repertoire which can be loaded using the -x option of tic.

FILES

/etc/terminfo Compiled terminal description database.

HISTORY

Although System V Release 2 provided a terminfo library, it had no documented tool for de?
compiling the terminal descriptions. Tony Hansen (AT&T) wrote the first infocmp in early
1984, for System V Release 3.

Eric Raymond used the AT&T documentation in 1995 to provide an equivalent infocmp for
ncurses. In addition, he added a few new features such as:

? the -e option, to support fallback (compiled-in) terminal descriptions

? the -i option, to help with analysis

Later, Thomas Dickey added the -x (user-defined capabilities) option, and the -E option to
support fallback entries with user-defined capabilities.

For a complete list, see the EXTENSIONS section.

In 2010, Roy Marples provided an infocmp program for NetBSD. It is less capable than the
SVr4 or ncurses versions (e.g., it lacks the sorting options documented in X/Open), but
does include the -x option adapted from ncurses.

PORTABILITY

X/Open Curses, Issue 7 (2009) provides a description of infocmp. It does not mention the
options used for converting to termcap format.

EXTENSIONS

The -0, -1, -E, -F, -G, -Q, -R, -T, -V, -a, -e, -f, -g, -i, -l, -p, -q and -t options are
not supported in SVr4 curses.

SVr4 infocmp does not distinguish between absent and cancelled capabilities. Also, it
shows missing integer capabilities as -1 (the internal value used to represent missing in?
tegers). This implementation shows those as ?NULL?, for consistency with missing strings.

The -r option's notion of ?termcap? capabilities is System V Release 4's. Actual BSD
curses versions will have a more restricted set. To see only the 4.4BSD set, use -r

-RBSD.

BUGS

The -F option of infocmp(1) should be a toe(1) mode.

SEE ALSO

captoinfo(1), infotocap(1), tic(1), toe(1), ncurses(3NCURSES), terminfo(5). user_caps(5).

https://invisible-island.net/ncurses/tctest.html

This describes ncurses version 6.3 (patch 20211021).

AUTHOR

Eric S. Raymond <esr@snark.thyrsus.com> and

Thomas E. Dickey <dickey@invisible-island.net>