NAME

tabs – set tabs on a terminal

SYNOPSIS

tabs [*options*]] *[tabstop-list]*

DESCRIPTION

The **tabs** program clears and sets tab-stops on the terminal. This uses the terminfo *clear_all_tabs* and *set_tab* capabilities. If either is absent, **tabs** is unable to clear/set tab-stops. The terminal should be configured to use hard tabs, e.g.,

```
stty tab0
```

Like **clear**(1), **tabs** writes to the standard output. You can redirect the standard output to a file (which prevents **tabs** from actually changing the tabstops), and later **cat** the file to the screen, setting tabstops at that point.

These are hardware tabs, which cannot be queried rapidly by applications running in the terminal, if at all. Curses and other full-screen applications may use hardware tabs in optimizing their output to the terminal. If the hardware tabstops differ from the information in the terminal database, the result is unpredictable. Before running curses programs, you should either reset tab-stops to the standard interval

```
tabs -8
```

or use the **reset** program, since the normal initialization sequences do not ensure that tab-stops are reset.

OPTIONS

General Options

−T*name*

Tell **tabs** which terminal type to use. If this option is not given, **tabs** will use the **$TERM** environment variable. If that is not set, it will use the *ansi+tabs* entry.

−d The debugging option shows a ruler line, followed by two data lines. The first data line shows the expected tab-stops marked with asterisks. The second data line shows the actual tab-stops, marked with asterisks.

−n This option tells **tabs** to check the options and run any debugging option, but not to modify the terminal settings.

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

The **tabs** program processes a single list of tab stops. The last option to be processed which defines a list is the one that determines the list to be processed.

Implicit Lists

Use a single number as an option, e.g., "**−5**" to set tabs at the given interval (in this case 1, 6, 11, 16, 21, etc.). Tabs are repeated up to the right margin of the screen.

Use "**−0**" to clear all tabs.

Use "**−8**" to set tabs to the standard interval.

Explicit Lists

An explicit list can be defined after the options (this does not use a "−"). The values in the list must be in increasing numeric order, and greater than zero. They are separated by a comma or a blank, for example,

```
tabs 1,6,11,16,21
tabs 1 6 11 16 21
```

Use a "+" to treat a number as an increment relative to the previous value, e.g.,

```
tabs 1,+5,+5,+5,+5
```

which is equivalent to the 1,6,11,16,21 example.

Predefined Tab-Stops

X/Open defines several predefined lists of tab stops.

−a Assembler, IBM S/370, first format

−a2 Assembler, IBM S/370, second format

−c COBOL, normal format

−c2 COBOL compact format

−c3 COBOL compact format extended

−f FORTRAN

−p PL/I

−s SNOBOL

−u UNIVAC 1100 Assembler

PORTABILITY

IEEE Std 1003.1/The Open Group Base Specifications Issue 7 (POSIX.1-2008) describes a **tabs** utility. However

- This standard describes a **+m** option, to set a terminal's left-margin. Very few of the entries in the terminal database provide the **smgl** (**set_left_margin**) or **smglp** (**set_left_margin_parm**) capability needed to support the feature.
- There is no counterpart in X/Open Curses Issue 7 for this utility, unlike **tput(1)**.

The **−d** (debug) and **−n** (no-op) options are extensions not provided by other implementations.

A **tabs** utility appeared in PWB/Unix 1.0 (1977). There was a reduced version of the **tabs** utility in Unix 7th edition and in 3BSD (1979). The latter supported a single "−n" option (to cause the first tab stop to be set on the left margin). That option is not documented by POSIX.

The PWB/Unix **tabs** utility, which was included in System III (1980), used built-in tables rather than the terminal database, to support a half-dozen terminal types. It also had built-in logic to support the left-margin, as well as a feature for copying the tab settings from a file.

Later versions of Unix, e.g., SVr4, added support for the terminal database, but kept the tables, as a fallback. In an earlier development effort, the tab-stop initialization provided by **tset** (1982) and incorporated into **tput** uses the terminal database,

POSIX documents no limits on the number of tab stops. Documentation for other implementations states that there is a limit on the number of tab stops (e.g., 20 in PWB/Unix's **tabs** utility). While some terminals may not accept an arbitrary number of tab stops, this implementation will attempt to set tab stops up to the right margin of the screen, if the given list happens to be that long.

The *Rationale* section of the POSIX documentation goes into some detail about the ways the committee considered redesigning the **tabs** and **tput** utilities, without proposing an improved solution. It comments that

> no known historical version of tabs supports the capability of setting arbitrary tab stops.

However, the *Explicit Lists* described in this manual page were implemented in PWB/Unix. Those provide the capability of setting abitrary tab stops.

SEE ALSO

tset(1), **infocmp**(1), **ncurses**(3NCURSES), **terminfo**(5).

This describes **ncurses** version 6.1 (patch 20190803).