## NAME

wm − window manager

## SYNOPSIS

**wm** [ **−n** ] [ **−f** rfile ]

## DESCRIPTION

*Wm* manages a collection of windows on a display terminal.

It determines what type of terminal you are using from the environment parameter $TERM (see *environ*(5)) and then uses *curses*(3) to adapt to it.

Each window has its own UNIX shell (as specified by the environment parameter $SHELL; the default is *sh*(1)), running in parallel with those in the other windows. The idea behind this scheme is that you can use each window for a series of commands dealing with a single topic. Then, you can change back and forth between the windows without losing your place in any of them.

At any time, you can give commands to change the window you are typing in, to move a window or change its size, or to create or kill a window. Windows can overlap or completely obscure one another. Obscured windows can subsequently be "lifted" up and placed on top of the other windows. When a window is partially or completely obscured, subsequent output to that window will be held until the window becomes completely visible again, effectively suspending the job running in the obscured window.

When the program is started, it will attempt to restore the sizes and positions of the windows in use when you last ran *wm* (unless you use **−n** option). If you give the **−f** *rfile* option, *wm* tries to restore windows from the file *rfile.* Otherwise, it first tries the file *.wmrc* in the current directory, then, if necessary, *.wmrc* in your home directory. Failing all this, *wm* will give you a window the size of your screen. Now, you can type UNIX commands in the new window.

*Wm* commands can be issued at any time. Every *wm* command consists of a prefix character <*wm-esc*> followed by one more character, possibly followed by some arguments. Any other characters typed on the terminal are treated as input to the program in the current window. To enter <*wm-esc*> itself as input to a program, type it twice. The <*wm-esc*> is initially set to ASCII ESC, but there is a command for changing it.

The available *wm* commands are <*wm-esc*> followed by:

**n** Create a new window. To do this, move the cursor to the position you want the lower left corner of your window to occupy, using the keys **k** (up), **j** (down), **h** (left), and **l** (right) and also **K**, **J**, **H**, and **L** (for large increments in the respective directions). Then type an **x**. Then move the cursor (using the same commands) to the upper right corner, and type another **x**.

**i** Identify each window by outlining it with its window number (each window has a one digit name, e.g. #**1**, #**2**, #**3**, used in referring to that window).

*digit* Change the current window. The named window is placed on "top" of the screen (that is, in front of any other windows that had been obscuring it). Input from the keyboard will now be sent to this window. (Delayed responses to commands entered in other windows will, of course, continue to be routed to their correct windows.)

**l** Change to the last-used window. Change back to the window that had most recently been the current window.

**m** Move and/or change the size of the current window. *Wm* asks for the lower left and upper right corners of desired new position of the window (same convention as for new windows). It is not advisable to move a window except when the program running in it is at the shell prompt level. In particular, screen-oriented programs such as *vi* can get very confused if their window size is changed while they are running.

**t** Reset the environment variables $TERM and $TERMCAP for the current window. This may be necessary if these variables get unset or scrambled, for instance during a remote login. This command should only be invoked when the program running in it is at the shell prompt level because it operates by sending commands to the shell as if they were typed in from the keyboard.