Section 1: Xinu Shell Commands

This section of the manual describes commands which run under the Xinu shell. Each page explains a single command, showing the syntax along with the possible arguments.

The Xinu shell accepts input from any I/O device, including the CONSOLE, a command file, or a TCP connection. Most commands are executed as independent processes. Features such as command pipelines, redirection of I/O devices, and background execution are supported. This version of the shell most closely resembles the shell described in Chapter 21 of **Comer & Munson: Operating System Design Vol. 1 MACINTOSH Edition (Prentice-Hall 1989)**.

The Xinu shell also understands Tcl scripts. A few sample Tcl scripts have been included and are described in the TCL(1) manual page. The Tcl language itself is described in **John K. Ousterhout: Tcl and the Tk Toolkit (Addison-Wesley 1994)**.

NAME

arp - print information about the arp cache

SYNOPSIS

arp

DESCRIPTION

Arp extracts information from the arp table and formats it in a readable form. Each line of output contains information about a single arp entry and shows the arp entry state (RESOLVED or PENDING), protocol address (IP address), hardware address (Ethernet address), hardware type, protocol type, interface number, and time to live (in seconds) for that entry.

SEE ALSO

conf(1), ifstat(1), route(1), stat(1)

NAME

bpool - print information about buffer pools

SYNOPSIS

bpool

DESCRIPTION

Bpool extracts information from the buffer pool table and formats it in a readable form. Each line of output contains information about a single buffer pool and shows the pool id, buffer size, semaphore, and number of buffers currently in the pool.

SEE ALSO

getbuf(2), freebuf(2), mkpool(2)

NAME

butler - print process information and semaphore information

SYNOPSIS

butler

DESCRIPTION

Butler extracts information from the process table and formats it in a readable form. Each line of output contains information about a single process and shows pid, pstate, pprio, pname, sp, ss, top of stack, di, si, flags and bp. Information displayed about the current process is out of date.

Butler also displays the PRWAIT, PRREADY and PRSLEEP semaphore queues. The queue linkages are checked and errors noted.

BUGS

Butler does not know about the PRTRECV state and thus flags it as an invalid state.

SEE ALSO

ps(1), kill(1)

NAME

cat - concatenate files and write to standard output

SYNOPSIS

cat [file ...]

DESCRIPTION

Cat concatenates a set of files by opening each file in turn and writing its contents to standard output. Because standard output normally goes to the window in which the shell runs, invoking *cat* on a single file causes the contents of the file to be written on the screen. Shell redirection can be used to combine a set of files into a single file as in: *cat file1 file2 >file3*

SEE ALSO

cp(1), echo(1)

NAME

chdsk - change Xinu disk drives

SYNOPSIS

chdsk [drv]

DESCRIPTION

Chdsk will change the default Xinu disk drive to the specified device *drv*. If *drv* is omitted, *chdsk* will log in a new floppy disk present in the default drive.

NOTES

Chdsk is used only with Xinu disks, not with MSDOS pseudo-devices.

SEE ALSO

dir(1), dsk(4), format(1), stat(1)

NAME

chwin - change the current window of the cursor

SYNOPSIS

chwin name

DESCRIPTION

Chwin places the cursor in the window specified by *name*. Keyboard input always goes to the process(es) reading from the window in which the cursor blinks.

NOTE

The cursor can also be repositioned by means of the PC Function Keys. F1 positions the cursor in the first window, F2 in the second, and so on. Windows appear in the device switch table in the order in which they were created. F10 is special, it connects the keyboard to the underlying CONSOLE device (see CON(4)).

SEE ALSO

clear(1), close(1), color(1), devs(1), goto(1), home(1), redraw(1), shell(1), window(1)

NAME

clear - clear a window

SYNOPSIS

clear [window]

DESCRIPTION

Clear clears either the named window or STDOUT (if it is a window). A border is not affected.

SEE ALSO

chwin(1), close(1), color(1), devs(1), goto(1), home(1), redraw(1), shell(1), window(1)

NAME

close - close a device descriptor

SYNOPSIS

close device

DESCRIPTION

Close takes as an argument a device descriptor and invokes the system call *close* on that device. Each device has a reference count that is incremented by the system call *open* and decremented by the system call *close*. Thus, if the device has been opened *n* times, the user must invoke the *close* command *n* times to close it.

SEE ALSO

devs(1)

NAME

color - change color parameters for the standard output window

SYNOPSIS

color [attr]

DESCRIPTION

Without any arguments, *color* changes the color of the standard output window to white on black. If an attribute string is given, the color of the standard output window is changed. When used with a black and white monitor, all colors except white map to black.

An attribute string has the format "fff/bbb" in which 'fff' and 'bbb' are three-character color codes representing the foreground and background colors to be used with a color display. The color codes are:

blk	black
blu	blue
grn	green
cyn	cyan
red	red
mag	magenta
yel	yellow
wht	white

The 'fff' or 'bbb' fields may be replaced by a single decimal digit in the range 0..7.

The color codes may be preceded by optional blink or intensify specifiers in the attribute string. The specifiers are:

? blink on ***** blink off **+** intensity on **-** intensity off

SEE ALSO

chwin(1), clear(1), close(1), goto(1), home(1), redraw(1), window(1)

NAME

conf - format and print system configuration information

SYNOPSIS

conf

DESCRIPTION

Command *conf* displays the following system configuration information: Xinu Version, Ethernet Address, IP Address and Domain Name for all physical network interfaces in the UP state, the process table size, the semaphore table size, the total number of devices, Time Server name, Remote File Server name, and the Domain Name Server name.

SEE ALSO

arp(1), ifstat(1), route(1), stat(1)

NAME

cp - copies one or more files to a target

SYNOPSIS

cp sourcefile destinationfile

DESCRIPTION

Cp copies file sourcefile to file destinationfile. *Cp* uses the namespace when opening files, so it is possible to copy a file to any device (e.g., the console).

SEE ALSO

cat(1), mv(1), nam(4), rm(1)

NAME

date, time - print the current time and date

SYNOPSIS

date

time

DESCRIPTION

Date extracts the time from the operating system and formats it in a readable form.

Time is a synonym for the same function as *date*.

BUGS

Date does not understand daylight savings time.

NAME

devs - print information from the device switch table

SYNOPSIS

devs

DESCRIPTION

Devs extracts information from the device switch table and formats it in a readable form. Each line of output contains information about a device or pseudo-device. Columns give the major device number, device name, minor device number, reference count, CSR, input and output interrupt vector addresses, and the device control block address.

SEE ALSO

close(1), mount(1), stat(1)

NAME

dir, ls - print the disk directory of a Xinu formatted disk

SYNOPSIS

dir

ls [drv]

DESCRIPTION

Dir prints disk directory information for Xinu formatted disks. *Ls* prints UNIX-style directory information for the Xinu disk in drive *drv*, or for the default drive if *drv* is omitted.

NOTES

Xinu has no shell command for displaying directory information for MSDOS files. The Tcl *file* commands can be used for this purpose.

SEE ALSO

chdsk(1), df(4), dsk(4), format(1), stat(1), tcl(1)

NAME

dos - terminate Xinu and return to MSDOS.

SYNOPSIS

dos

DESCRIPTION

Dos causes Xinu to terminate and return to MSDOS.

SEE ALSO

exit(1)

NAME

echo - echo arguments to standard output device

SYNOPSIS

echo argument...

DESCRIPTION

Echo prints its arguments separated by blanks on standard output. It is useful for testing argument interpretation and for inserting a constant string into a file (e.g. a comment in a shell script).

SEE ALSO

cat(1)

NAME

edit - edit a file in a window

SYNOPSIS

edit [file]

DESCRIPTION

Edit is an ultra-simple text editor. It is intended for writing short TCL and Shell command scripts. *Edit* understands all the cursor positioning keys on a PC keyboard plus a few commands:

ctrl-s	show editor status on status device
ctrl-y	delete line
ESC	exit editor, save file if necessary

The diamond character indicates the EOF position on-screen.

NOTES

Edit is a screen-oriented editor and uses a function called *show*, which writes directly to VGA video RAM. It does not accept STDOUT or STDIN redirection. Also, *edit* cannot handle files which are larger than its edit buffer size (currently 12 K).

SEE ALSO

sh(1), tcl(1)

NAME

exit - cause the process executing the shell to terminate

SYNOPSIS

exit

DESCRIPTION

Exit causes the process executing the shell to terminate immediately. Termination removes the shell process permanently, and ceases the command interpretation. *Exit* is used mostly by shell command scripts to terminate prematurely. In an interactive shell, it is equivalent to logging out from the computer. When invoked from a subshell, *exit* returns to the shell from which the subshell was invoked.

SEE ALSO

dos(1), sh(1)

NAME

finger - print information on logged in users

SYNOPSIS

finger [userid]@hostname

DESCRIPTION

Finger opens a TCP connection to the finger port at host *hostname*, which returns information on user *userid* or (if *userid* is omitted) on all logged in users, and then closes the connection. The returned information is printed on STDOUT.

Xinu has a **remote finger daemon**, which responds to incoming finger requests and which logs such requests to the STATUS device.

SEE ALSO

ping(1), tcp(1), tcp(4), udp(1)

NAME

format - format a floppy disk

SYNOPSIS

format idnum

DESCRIPTION

Format takes an ID number *idnum* as an argument and formats the floppy disk in device DS0 (the DOS A: drive). If no disk was present when Xinu was started, *format* prints an error message. Xinu disks can be read or written while formatting is taking place.

SEE ALSO

chdsk(1), dir(1), df(4), dsk(4)

BUGS

Because the disk routines in ROM disable interrupts for lengthy periods, *format* causes the clock to miss ticks and all other processes to run very slowly. The network device will experience receive buffer overflow in some cases.

NAME

goto - position the cursor within a window

SYNOPSIS

goto coord

home

DESCRIPTION

Goto positions the cursor at the coordinates specified in string *coord* within the current window. The coordinates are specified as **col, row** and are decimal numbers indicating a position relative to the top-left corner of the window (irrespective of a border).

Home positions the cursor at **0,0** (the top-left corner of the window), irrespective of a border.

SEE ALSO

chwin(1), clear(1), close(1), color(1), devs(1), redraw(1), shell(1), window(1)

NAME

help, man - print a help message or manual page

SYNOPSIS

help [cmd]

man [cmd]

DESCRIPTION

Help with no argument prints a help message that shows the possible commands available under the shell. Only command names are printed; the output is displayed with four columns across the page to limit the length of the output. If a command is specified, *help* prints the manual page for the specified command. *Help* uses *more* to print the page one window at a time.

Man is a synonym for *help*.

Manual pages are text files called *cmd.doc* located in directory /src/shell.

SEE ALSO

more(1)

NAME

ifstat - print information about a network interface

SYNOPSIS

ifstat ifnum

DESCRIPTION

The *ifstat* command requires a parameter *ifnum*, which specifies which of the Xinu network interface data structures is to be examined (interface 0 is the local software interface, interface 1 the first network device and so on). The command prints the following information:

For the local interface: whether the interface is UP or DOWN

For other interfaces: Interface state, IP address, Name, Mask, IP broadcast address, Maximum Transfer Unit, Hardware Address, Hardware Broadcast Address, Input Queue number.

SEE ALSO

arp(1), conf(1), route(1), stat(1)

NAME

kill - terminate a process

SYNOPSIS

kill pid

DESCRIPTION

Kill attempts to kill the process with id given by argument *pid*. The *kill* shell command invokes the system call *kill*, which will terminate the named process (unless it is immortal). Termination is final and permanent. All record of the process is removed, so the process cannot be recovered or continue.

SEE ALSO

kill(2), ps(1)

NAME

memstat - print information about current memory use

SYNOPSIS

memstat

DESCRIPTION

Memstat extracts information from the system's memory management data structures and summarizes it in a readable form. The output shows the amount of memory currently available, the number of allocation holes, and the minimum, maximum and average hole sizes.

SEE ALSO

getmem(2), freemem(2), pmem(1), xmalloc(2), xfree(2)

NAME

more - echo a file to STDOUT, *rowsize-1* lines at a time

SYNOPSIS

more [file]

DESCRIPTION

More displays *file* on STDOUT, *rowsize-1* lines at a time, prompting for a continuation character every *rowsize-1* lines. STDOUT must be a TTY device. If *file* is not specified, *more* will act on STDIN. The continuation character can be a single CR character, in which case output continues, or the letters **q** or **Q** followed by a CR, in which case *more* is terminated. Ctrl-d also terminates output.

SEE ALSO

help(1)

NAME

mount - print or modify information in the namespace mapping table

SYNOPSIS

mount [prefix device replacement]

DESCRIPTION

With no arguments, *mount* extracts information from the namespace prefix mapping table and displays it in readable form. Given three arguments, *mount* interprets them as a new entry to be added to the table. Argument *prefix* specifies a name mapping prefix. It can be a new prefix or an exact match of an existing prefix. *Device* is the name of a device to be used for this prefix (see DEVS(1)) or the special string "SYSERR" which means no process can open a name that starts with the prefix. Finally, argument *replacement* gives the replacement string for the entry.

Mount uses the underlying system call MOUNT(2) to insert entries in the table, so new items are always inserted just before the first entry that would mask their effect.

SEE ALSO

devs(1), mount(2), nam(4), unmount(1), unmount(2)

NAME

mv - move a file from one name to another

SYNOPSIS

mv oldname newname

DESCRIPTION

Mv changes the name of a file. Argument *oldname* specifies an existing name as recognized by the namespace (NAM(4)) and argument *newname* specifies a new name for the file. *Mv* invokes the underlying system call RENAME(2) which changes names but does not copy objects across device boundaries.

SEE ALSO

cat(1), cp(1), rm(1)

NAME

ns - resolve a Domain Name

SYNOPSIS

ns [hostname]+

DESCRIPTION

The *ns* command will interact with a Domain Name Server and resolve one or more specified host names, printing the IP address of each resolved name or the string "not resolved" if the name could not be resolved.

SEE ALSO

ping(1)

NAME

ping - send ICMP Echo Requests to an Internet site

SYNOPSIS

ping name [size]

DESCRIPTION

Command *ping* sends ten ICMP Echo Requests of size to the site *name* at one second intervals. If *size* is omitted, packets of 64 bytes are sent. Each received ICMP Echo Request Response is noted and a summary of the number of responses received is printed on STDOUT.

Ping is useful for testing connectivity to a remote site.

SEE ALSO

tcp(1), udp(1)

NAME

pmem - display memory allocation for a process

SYNOPSIS

pmem pid

DESCRIPTION

Pmem displays the paragraph address and size of each Xinu memory block allocated by *xmalloc* to process *pid*. The total number of blocks and the total amount of memory allocated is also displayed. *Pmem* is useful for debugging purposes.

SEE ALSO

memstat(1), ps(1), xmalloc(2)

NAME

ps - print information from the process table at run-time

SYNOPSIS

ps

DESCRIPTION

Ps extracts information from the process table and formats it in a readable form. Each line of output contains information about a single Xinu process. Columns give the process identifier, process state, process priority, process name, amount of memory allocated, and the approximate amount of CPU time (in ticks since startup) which a process has consumed.

SEE ALSO

kill(1), butler(1)

BUGS

The amount of CPU time a process has consumed is calculated by incrementing a counter belonging to the process which happens to be running when a clock interrupt occurs. CPU time used by processes in between clock ticks is not captured.

NAME

redraw - redraw a window

SYNOPSIS

redraw [name]

DESCRIPTION

Redraw redraws either the window specified by *name*, or STDOUT (which must be a window) if *name* is omitted.

SEE ALSO

chwin(1), clear(1), close(1), color(1), devs(1), goto(1), home(1), shell(1), window(1)

BUGS

If the window to be redrawn has a border, *redraw* does not redraw the border and window name correctly.

NAME

rlogin - login to a remote Xinu system

SYNOPSIS

rlogin hostname

DESCRIPTION

Command *rlogin* is used to login to a remote Xinu system named *hostname* via a TCP connection to the Remote Login Port. An **rlogin daemon** at the remote site creates a shell with STDIO connected to the TCP device at that site. The local user can then issue commands for execution by the remote shell. *Rlogin* creates an additional process, which reads keyboard input and passes that input to the remote shell. Conversely, *Rlogin* reads input from the TCP connection and displays that input on the local device. Typing **exit** or **ctrl-d** will terminate the command, terminate the remote shell, and close the TCP connection.

BUGS

Some shell commands (eg. edit, snmp, status) do not work on a device which is not a TTY.

NAME

rm - remove a file

SYNOPSIS

rm filename

DESCRIPTION

Rm removes (destroys) the file with name *filename*. *Rm* uses the namespace when accessing files and then invokes the system call *control* to remove the file. Once a file has been removed it is no longer accessible; there is no way to recover the data.

SEE ALSO

control(2), cp(1), mv(1)

NAME

route - add or delete a routing table entry
routes - display the routing table

SYNOPSIS

route add dest mask nexthop metric ttl
route delete dest mask
routes [-n]

DESCRIPTION

The *route add* command is used to add a destination entry to the routing table. Argument *dest* specifies the destination IP address (or name) of an entry, argument *mask* is the network mask for *dest*, argument *nexthop* is the IP address (or name) of the host or gateway to which IP packets for *dest* should be sent, argument *metric* is the distance (in hops) to *nexthop*, and argument *ttl* is the time-to-live (in seconds) for this entry. Specifying a value of 16 for *metric* indicates an infinite metric, and specifying a value of 999 for *ttl* indicates an infinite time-to-live. An infinite metric indicates that the route should not be used, while an infinite time-to-live indicates that the routing entry is permanent (i.e. the entry will not be aged). The routing software automatically determines the network interface to which packets for *dest* should be sent. The *route delete* command deletes the entry specified by *dest* and *mask*.

The *routes* command formats and prints the routing table. The optional argument **-n** indicates that names rather than IP addresses should be displayed in the printout. In addition to the above information, *routes* displays the network interface to which a packet is sent (interface 0 is the interface to local protocol software), the number of processes currently accessing an entry (field *refcnt*), and the number of times an entry has been used (field *usecnt*).

SEE ALSO

arp(1), conf(1), ifstat(1), stat(1)

NAME

sh - invoke a subshell to interpret commands

DESCRIPTION

Sh creates a separate, fresh invocation of the command interpreter (shell). *Sh* is most useful for executing command scripts with the input taken from a file or another window. However, recursive invocation of interactive shells works as expected - the user interacts with the innermost (most recent) shell, returning to the next outer shell after the inner shell terminates.

SEE ALSO

exit(1), shell(1)

NAME

shell - create a new shell in a window

SYNOPSIS

shell border [color] name

DESCRIPTION

Shell creates a shell in a new window specified by *border* and the optional argument *color*. The window is displayed with the string *name* in its top border. *Shell* calls WINDOW(1) to create the new window.

SEE ALSO

chwin(1), clear(1), close(1), color(1), devs(1), goto(1), home(1), redraw(1), shell(1), window(1)

NAME

sleep - delay for a specified number of seconds

SYNOPSIS

sleep delay

DESCRIPTION

Sleep delays for *delay* seconds and then continues executing. It is often useful in debugging because it gives a way to have a process stay around with its standard input, output and error files open.

SEE ALSO

sleep(2)

NAME

snmp - invoke an SNMP shell in the current window

SYNOPSIS

snmp agent-name

DESCRIPTION

Snmp opens a simple SNMP shell, which can be used to display or modify SNMP MIB variables at host *agent-name*. The agent name can be specified as an IP address or as a Domain Name.

The SNMP shell understands the following commands:

exit	exit the SNMP shell
agent	specify a new agent name
[object-name]+	display the contents of one or more MIB variables
next [object-name]+	perform a GET NEXT on the specified MIB variables
set [object-name type value]+	set the variable *object-name* to the value specified in *value* and the type specified in *type*.

Valid types are: [**int** | **counter** | **gauge** | **timeticks** | **str** | **objid** | **ipaddr**]

Control Characters:

TAB	complete the next object name
ctrl-n	get next object
ctrl-w	kill word

Xinu has an **SNMP daemon**, which responds to incoming SNMP requests and which logs such requests to the STATUS device.

NAME

dgstat, dskst, netstat, pipstat, slstat, ttystat - device status commands

SYNOPSIS

dgstat
dskst
netstat [i]
pipstat
slstat
ttystat

DESCRIPTION

The device status commands provide debugging information about the devices: UDP, DSK, TCP, PIPE, SL0 and TTY. All commands print a header line followed by information on instances of the device.

The *netstat* command takes an optional parameter **i**, which specifies that network interface statistics are to be printed. Otherwise, *netstat* prints the status of all TCP connections (both active and passive).

SEE ALSO

udp(4), dsk(4), eth(4), tcp(4), pip(4), sl(4), tty(4), arp(1), conf(1), ifstat(1), route(1)

NAME

status - display system status in a status window

SYNOPSIS

status [time]

DESCRIPTION

Status displays the Domain Name of the primary network interface, the current number of processes, the number of packets sent and received, the amount of system memory available, the pid of the *status* process, and the current time. The display is updated at the interval specified as tenths of seconds in *time* (default .2 seconds). The status information is normally displayed in a window called *status*, which should be created before the command is issued (usually in file STARTUP). If STDOUT is redirected to another device (e.g. a file), that device can be used to log system status.

To avoid screen flicker at high output rates, information is displayed using direct video RAM access if STDOUT is a window.

Some network daemons also log activity on the *status* window.

NAME

tcl - Tool Command Language Version 6.7 Shell

SYNOPSIS

tcl [script] [args...]
tclsh

DESCRIPTION

This Xinu adaptation of John Ousterhout's Tool Command Language supports most features of TCL as described in "Tcl and the Tk Toolkit"(Addison-Wesley 1994).

Tcl can be invoked from the Xinu shell with an optional argument *script* specifying the name of a Tcl script file. If *script* is omitted, a Tcl shell will be invoked in the current window. If a Tcl script contains **#!tcl** in its first line, its filename can be typed directly to the Xinu shell, and the Tcl interpreter will be invoked to execute that script. In both cases, arguments are passed as expected.

Command *tclsh* is a Xinu shell command file which invokes a Tcl interpreter in a previously created window called **tcl** (see WINDOW(1)).

The following Tcl scripts are included in directory pcxnet\main and can be executed from either the Xinu shell or the Tcl shell:

filter regexp copy lines containing *regexp* from STDIN to STDOUT

show shcmd regexp execute *shcmd* and write output containing *regexp* to STDOUT

tgrep regexp fname write all lines in *fname* containing *regexp* to STDOUT

NOTES

Not all of the Tcl *file*, *exec* and *open* options have been ported. Floating-point operations are **not** supported.

SEE ALSO

sh(1), window(1)

NAME

tcp - send and receive data over a TCP connection

SYNOPSIS

tcp hostname [port]

DESCRIPTION

Tcp opens a connection to *port* at host *hostname* and then transfers data read from STDIN to the port. Data received from the port is written to STDOUT. If *port* is omitted, the ECHO PORT (7) is used. The *tcp* command is used for testing TCP software.

Xinu has an **echo daemon**, which responds to incoming echo requests and which logs such requests to the STATUS device.

SEE ALSO

udp(1)

NAME

test - shell command stub

SYNOPSIS

test [args...]

DESCRIPTION

Command *test* is provided as a stub for extension of the command set.

NAME

tftp - conduct one TFTP send/receive transaction

SYNOPSIS

tftp {put | get} host srcname [destname]

DESCRIPTION

The trivial file transfer protocol (TFTP) is used for transferring files between hosts. The *put* operation copies file *srcname* to file *destname* on *host*. If *destname* is omitted, the copy of the file is also called *srcname*.

The *get* operation behaves in a similar way in the reverse direction.

BUGS

Tftp does not transfer binary files correctly.

NAME

timerq - format and print the TCP timer queue entries

SYNOPSIS

timerq

DESCRIPTION

Command *timerq* displays the following information on the TCP timer queue: the time left (in 100 msec units) for this entry (relative to either the time the entry was queued or the previous entry), the time (in 100 msec units) when this entry was inserted, the port to which a message is sent on expiry, the length of the port, and the message to be sent when the time expires.

SEE ALSO

netstat(1)

NAME

udp - send and receive UDP datagrams

SYNOPSIS

udp hostname [port]

DESCRIPTION

Udp sends a string containing an e-mail address to the UDP port *port* at host *hostname*. Data received from *port* is copied to STDOUT. If *port* is omitted, the UDP ECHO port (7) is used. *Udp* is used for testing communications with UDP ports.

Xinu has a **UDP echo daemon**, which responds to incoming packets at UDP port 7, and which logs activity to the STATUS device.

SEE ALSO

tcp(1)

NAME

unmount - remove an entry from the namespace prefix table

SYNOPSIS

unmount prefix

DESCRIPTION

Unmount removes the entry from the namespace prefix mapping that has prefix exactly equal to argument *prefix*.

SEE ALSO

mount(1), mount(2), nam(4), unmount(2)

NAME

wc - count words in a file

SYNOPSIS

wc

DESCRIPTION

Wc is a simple word count program which counts words read from STDIN.

BUGS

Wc does not count characters and lines as does its UNIX counterpart.

NAME

window - create or close a window

SYNOPSIS

window border [attr] name

window close name

DESCRIPTION

In its first form, *window* creates a window of the size and at the location specified by the *border* string. The *border* string has the format: **#c1,r1:c2,r2** where **c1,r1** is a string representing the decimal coordinates (in column, row order) of the upper-left corner of the window, and **c2,r2** is a string representing the coordinates of the bottom-right corner. If the **#** character is omitted, the window will be created without a border.

Argument *name* specifies the pseudo-device name of the created window. If the window has a border, the string *name* is displayed centered within the top border. The optional argument *attr* describes the color and other attributes of the displayed window (see COLOR(1)).

In its second form, *window* closes the named window.

SEE ALSO

chwin(1), clear(1), close(1), color(1), devs(1), goto(1), home(1), redraw(1), shell(1)