- -4pc - -4pc

Name

DD -- convertir y copiar un fichero

dd
dd [--help] [--version] [*if=fichero*] [*of=fichero*] [*ibs=bytes*] [*obs=bytes*] [*bs=bytes*] [*cbs=bytes*][*skip=bloques*][*seek=bloques*][*count=bloques*][*conv=* [*ascii,ebcdic,ibm,block,unbloc*

DESCRIPCIÓN

dd copia un fichero (de la entrada estándar a la salida estándar, por omisión) con un tamaño de bloque seleccionable por el usuario, a la par que, opcionalmente, realiza sobre él ciertas conversiones.

Lee la entrada un bloque cada vez, usando el tamaño de bloque de entrada especificado (el valor predeterminado es 512 bytes). Si se dio la opción *bs=bytes* y no se especificó otra conversión aparte de *sync*, *noerror*, o *notrunc*, escribe la cantidad de datos leídos (que puede ser menor que la solicitada) en un bloque de salida separado. Este bloque de salida tiene precisamente la misma longitud que fue leída a menos que se especificara la conversión *sync*, en cuyo caso, los datos se rellenan con NULos (o espacios, véase debajo).

De otro modo, la entrada, leída un bloque de cada vez, se procesa t la salida resultante es recogida y escrita en bloques del tamaño de bloque de salida especificado. El último bloque de salida puede ser más corto.

Las opciones de más abajo con valores numéricos (bytes y bloques) pueden ir seguidas por un factor multiplicador: ‘k’=1024, ‘b’=512, ‘w’=2, ‘c’=1 (‘w’ y ‘c’ son extensiones de GNU; ‘w’ nunca debería utilizarse: significa 2 en System V y 4 en 4.2BSD). Dos o más de tales expresiones numéricas pueden multiplicarse poniendo una ‘x’ (equis minúscula) entre ellas. La versión fileutils-4.0 de GNU también permite los siguientes sufijos multiplicativos al especificar tamaños de bloque (en bs=, cbs=, ibs=, obs=): M=1048576, G=1073741824, y así para T, P, E, Z, Y. Un sufijo ‘D’ los convierte en decimal: kD=1000, MD=1000000, GD=1000000000, etc. (Dese cuenta que para ls, df, du, el tamaño de M, etc., viene determinado por variables de entorno, pero para dd es fijo.)

OPCIONES

* 0.60+1em

* 0.60+1em *if= fichero*
  Lee desde *fichero* en lugar de desde la entrada estándar.

* 0.60+1em *of=fichero*
  Write to *file* instead of standard output. Unless *conv=notrunc* is given, **dd** truncates *file* to zero bytes (or the size specified with *seek*=).

* 0.60+1em *ibs=bytes*
  Lee *bytes* bytes de una vez. El valor predeterminado es 512.

* 0.60+1em *obs=bytes*
  Escribe *bytes* bytes de una vez. El valor predeterminado es 512.

* 0.60+1em *bs=bytes*
  Lee y escribe *bytes* bytes cada vez. Esto sustituye a *ibs* y *obs*. (Y poner *bs* no es equivalente a poner *ibs* y *obs* al mismo valor, al menos si no se ha especificado una conversión aparte de *sync*, *noerror* y *notrunc*, puesto que se estipula que cada bloque de entrada será copiado en la salida como un único bloque, sin agregar bloques cortos.)

* 0.60+1em *cbs=bytes*
  Especifica el tamaño de bloque de conversión para *block* y *unblock*.

- -4pc - -4pc
* 0.60+1em *skip=bloques*
Salta *bloques* bloques de tamaño en bytes determinado por *ibs* del fichero de entrada antes de la copia.

* 0.60+1em *seek=bloques*
Salta *bloques* bloques de tamaño en bytes determinado por *obs* en el fichero de salida antes de la copia.

* 0.60+1em *count=bloques*
Copia *bloques* bloques de tamaño en bytes determinado por *ibs* del fichero de entrada, en vez de todo hasta el final del fichero.

* 0.60+1em *conv=CONVERSIÓN[,CONVERSIÓN]...*
Convierte el fichero según se haya especificado en el o los argumentos *CONVERSIÓN*. (No se permite ningún espacio al lado de ninguna coma.)

Conversiones:
* 0.60+1em

* 0.60+1em *ascii*
Convierte EBCDIC a ASCII.

* 0.60+1em *ebcdic*
Convierte ASCII a EBCDIC.

* 0.60+1em *ibm*
Convierte ASCII a un EBCDIC alternativo.

* 0.60+1em *block*
Para cada línea de la entrada, saca *cbs* bytes, reemplazando el salto de línea de la entrada con un espacio y rellenando con más espacios si fuera necesario.

* 0.60+1em *unblock*
Reemplaza espacios del final en cada bloque de entrada de tamaño determinado por *cbs* por un salto de línea.

* 0.60+1em *lcase*
Cambia las letras mayúsculas por minúsculas.

* 0.60+1em *ucase*
Cambia las letras minúsculas por mayúsculas.

* 0.60+1em *swab*
Intercambia cada par de bytes de la entrada. Si se lee un número impar de bytes el último byte se copia tal cual (ya que no tiene con quién intercambiarse). [POSIX 1003.2b, PASC interpretaciones 1003.2 nº 3 y nº 4]

* 0.60+1em *noerror*
Continúa despues de producirse errores de lectura.

* 0.60+1em *notrunc*
No trunca el fichero de salida.

* 0.60+1em *sync*
Rellena cada bloque de entrada hasta el tamaño determinado por *ibs* con bytes cero al final.

- -4pc - -4pc

OPCIONES ESTÁNDAR DE GNU

* 0.60+1em

* 0.60+1em `--help`
  Muestra un mensaje en la salida estándar sobre el modo de empleo y acaba con código de éxito.

* 0.60+1em `--version`
  Muestra en la salida estándar información sobre la versión y luego acaba con código de éxito.

* 0.60+1em `--`
  Termina la lista de opciones.

ENTORNO

Las variables LANG, LC_ALL, LC_TYPE y LC_MESSAGES tienen el significado habitual.

CONFORME A

POSIX 1003.2

EJEMPLO

A menudo, una unidad de cinta no aceptará bloques de tamaño arbitrario y **dd** obtendrá un error de E/S en el último fragmento de datos que no ocupe un bloque entero. Use ‘dd if=mifichero of=/dev/miunidaddecinta conv=sync’ para asegurarse de que todo se ha grabado en la cinta. Naturalmente, leerla de nuevo producirá ahora un fichero ligeramente más grande, con caracteres nulos añadidos al final.

ERRORES

Comandos como ‘dd if=mifichero of=/dev/fd0 bs=1k seek=172’ fallan en algunos sistemas porque **dd** intenta truncar el fichero de salida, pero el truncado de un dispositivo de bloques no es posible. En dichos casos, añádase la opción ‘conv=notrunc’.

NOTAS

Esta página describe **dd** como se encuentra en el paquete fileutils-4.0; otras versiones pueden diferir ligeramente.