

**NAME**

`dv2dt` – convert a binary TeX DVI file to DTL text representation

**SYNOPSIS**

**dv2dt** *input-DVI-file output-DTL-file*

If the filenames are omitted, then *stdin* and *stdout* are assumed.

**DESCRIPTION**

**dv2dt** converts a binary TeX DVI file to an editable text file in DTL (*DVI Text Language*) format. The companion **dt2dv**(1) utility can convert the DTL file back to a binary DVI file.

**DVI COMMAND DESCRIPTION**

TeX DVI files contain a compact binary description of typeset pages, as a stream of operation code bytes, each immediately followed by zero or more parameter bytes. The format of DVI files is fully described in Donald E. Knuth, *TeX: The Program*, Addison-Wesley (1986), ISBN 0-201-13437-3, as well as in the **dvitype**(1) literate program source code.

For convenience, we provide a summary of DVI commands here. In the following list, operation code bytes are given as unsigned decimal values, followed by their symbolic names (not present in the DVI file), and a short description. A designation like *b[+n]* means that the operation code byte is followed by a parameter *b* which uses *n* bytes, and is signed. Without the plus sign, the parameter is unsigned. Signed integer parameter values are always represented in two's complement arithmetic, which is the system followed by most computers manufactured today, including all personal computers and workstations.

<i>0 set_char_0</i>	Set character 0 from current font.
...	
<i>127 set_char_127</i>	Set character 127 from current font.
<i>128 set1 c[1]</i>	Set 1-byte unsigned character (uchar) number <i>c</i> .
<i>129 set2 c[2]</i>	Set 2-byte uchar number <i>c</i> .
<i>130 set3 c[3]</i>	Set 3-byte uchar number <i>c</i> .
<i>131 set4 c[+4]</i>	Set 4-byte signed character (schar) number <i>c</i> .
<i>132 set_rule a[+4] b[+4]</i>	Set rule, height <i>a</i> , width <i>b</i> .
<i>133 put1 c[1]</i>	Put 1-byte uchar <i>c</i> .
<i>134 put2 c[2]</i>	Put 2-byte uchar <i>c</i> .
<i>135 put3 c[3]</i>	Put 3-byte uchar <i>c</i> .
<i>136 put4 c[+4]</i>	Put 4-byte schar <i>c</i> .
<i>137 put_rule a[+4] b[+4]</i>	Put rule, height <i>a</i> , width <i>b</i> .
<i>138 nop</i>	Do nothing.
<i>139 bop c0[+4] ... c9[+4] p[+4]</i>	Beginning of page. The parameters <i>c0</i> ... <i>c9</i> are the TeX page counters, the contents of TeX count registers <i>\count0</i> ... <i>\count9</i> . The parameter

140 *eop*

141 *push*

142 *pop*

143 *right1*  $b[+1]$

144 *right2*  $b[+2]$

145 *right3*  $b[+3]$

146 *right4*  $b[+4]$

147 *w0*

148 *w1*  $b[+1]$

149 *w2*  $b[+2]$

150 *w3*  $b[+3]$

151 *w4*  $b[+4]$

152 *x0*

153 *x1*  $b[+1]$

154 *x2*  $b[+2]$

155 *x3*  $b[+3]$

156 *x4*  $b[+4]$

157 *down1*  $a[+1]$

158 *down2*  $a[+2]$

159 *down3*  $a[+3]$

160 *down4*  $a[+4]$

161 *y0*

162 *y1*  $a[+1]$

163 *y2*  $a[+2]$

164 *y3*  $a[+3]$

165 *y4*  $a[+4]$

166 *z0*

167 *z1*  $a[+1]$

168 *z2*  $a[+2]$

169 *z3*  $a[+3]$

170 *z4*  $a[+4]$

$p$  is the byte offset from the beginning of the DVI file of the previous *bop* operation code byte. The first such command in the file has  $p = -1$ .

End of page.

Push  $(h, v, w, x, y, z)$  onto stack.

Pop  $(h, v, w, x, y, z)$  from stack.

Move right  $b$  units.

Move right  $b$  units.

Move right  $b$  units.

Move right  $b$  units.

Move right  $w$  units.

Move right  $b$  units, and set  $w = b$ .

Move right  $b$  units, and set  $w = b$ .

Move right  $b$  units, and set  $w = b$ .

Move right  $b$  units, and set  $w = b$ .

Move right  $x$  units.

Move right  $b$  units, and set  $x = b$ .

Move right  $b$  units, and set  $x = b$ .

Move right  $b$  units, and set  $x = b$ .

Move right  $b$  units, and set  $x = b$ .

Move down  $a$  units.

Move down  $a$  units.

Move down  $a$  units.

Move down  $a$  units.

Move right  $y$  units.

Move right  $a$  units, and set  $y = a$ .

Move right  $a$  units, and set  $y = a$ .

Move right  $a$  units, and set  $y = a$ .

Move right  $a$  units, and set  $y = a$ .

Move right  $z$  units.

Move right  $a$  units, and set  $z = a$ .

Move right  $a$  units, and set  $z = a$ .

Move right  $a$  units, and set  $z = a$ .

Move right  $a$  units, and set  $z = a$ .

171 <i>fnt_num_0</i>	Set current font number ( $f$ ) = 0.
...	
234 <i>fnt_num_63</i>	Set $f = 63$ .
235 <i>fnt1 k[1]</i>	Set $f = k$ .
236 <i>fnt2 k[2]</i>	Set $f = k$ .
237 <i>fnt3 k[3]</i>	Set $f = k$ .
238 <i>fnt4 k[+4]</i>	Set $f = k$ .
239 <i>xxx1 k[1] x[k]</i>	Special string $x$ with $k$ bytes.
240 <i>xxx2 k[2] x[k]</i>	Special string $x$ with $k$ bytes.
241 <i>xxx3 k[3] x[k]</i>	Special string $x$ with $k$ bytes.
242 <i>xxx4 k[4] x[k]</i>	Special string $x$ with (unsigned) $k$ bytes.
243 <i>fnt_def1 k[1] c[4] s[4] d[4] a[1] l[1] n[a+l]</i>	Define font $k$ . The parameters are: $c$ Checksum for TFM file. $s$ Scale factor, in DVI units. $d$ Design size, in DVI units. $a$ Length of the “area” or directory. $l$ Length of the font name. $n$ Area and font name string(s).
244 <i>fnt_def2 k[2] c[4] s[4] d[4] a[1] l[1] n[a+l]</i>	Define font $k$ .
245 <i>fnt_def3 k[3] c[4] s[4] d[4] a[1] l[1] n[a+l]</i>	Define font $k$ .
246 <i>fnt_def4 k[+4] c[4] s[4] d[4] a[1] l[1] n[a+l]</i>	Define font $k$ .
247 <i>pre i[1] num[4] den[4] mag[4] k[1] x[k]</i>	Begin preamble. The parameters are: $i$ DVI format. Standard $\text{\TeX}$ has $ID = 2$ , and $\text{\TeX-X}\text{\TeX}$ has $ID = 3$ . $num$ Numerator of 100 nm / DVI unit. $den$ Denominator of 100 nm / DVI unit. $mag$ 1000 * magnification. $k$ Comment length. $x$ Comment string.
248 <i>post p[4] num[4] den[4] mag[4] l[4] u[4] s[2] t[2]</i>	Begin postamble. The parameters are:

	<i>p</i>	Pointer to final bop.
	<i>num, den, mag</i>	Duplicates of values in preamble.
	<i>l</i>	Height-plus-depth of tallest page, in DVI units.
	<i>u</i>	Width of widest page, in DVI units.
	<i>s</i>	Maximum stack depth needed to process this DVI file.
	<i>t</i>	Total number of pages ( <i>bop</i> commands) present.
249	<i>post_post q[4] i[1]</i>	End postamble. The parameters are:
	<i>q</i>	Byte offset from the beginning of the DVI file to the <i>post</i> command that started the postamble.
	<i>i</i>	DVI format ID, as in the preamble.
223		At least four 223 bytes.
250		Undefined.
...		
255		Undefined.

## DTL COMMAND DESCRIPTION

A DTL file contains one line per command, with a limit of 1024 characters per line. Each command contains a symbolic operation name, followed by zero or more parameter values. The parameter value descriptions are not repeated here; they can be found in the previous section.

variety <variety-name>	This command specifies the name of the DTL file type; it has no DVI file equivalent.
( <i>text</i> )	Series of <i>set_char</i> commands, for printable ASCII text.
\(	Literal ASCII left parenthesis in ( <i>text</i> ).
\)	Literal ASCII right parenthesis in ( <i>text</i> ).
\\	Literal ASCII backslash in ( <i>text</i> ).
\"	Literal ASCII double quote in ( <i>text</i> ).
\XY	<i>Set_char</i> for character with hexadecimal code XY, not in parentheses, but by itself for readability.
<i>s1, s2, s3</i>	Set, with (1,2,3,4)-byte charcodes.
<i>sr</i>	<i>set_rule</i> .

<i>p1, p2, p2, p3</i>	Put, with (1,2,3,4)-byte charcodes.
<i>pr</i>	<i>put_rule</i> .
<i>nop</i>	<i>nop</i> (do nothing).
<i>bop</i>	<i>bop</i> (beginning of page).
<i>eop</i>	<i>eop</i> (end of page).
<i>[</i>	Push.
<i>]</i>	Pop.
<i>r1, r2, r3, r4</i>	Right, with (1,2,3,4)-byte argument.
<i>w0, w1, w2, w3, w4</i>	As in DVI.
<i>x0, x1, x2, x3, x4</i>	As in DVI.
<i>d1, d2, d3, d4</i>	Down, with (1,2,3,4)-byte argument.
<i>y0, y1, y2, y3, y4</i>	As in DVI.
<i>z0, z1, z2, z3, z4</i>	As in DVI.
<i>fn</i>	<i>fn_num</i> (set current font to font number in 0 to 63).
<i>f1, f2, f3, f4</i>	<i>fnt</i> (set current font to (1,2,3,4)-byte font number).
<i>special</i>	<i>xxx</i> (special commands with (1,2,3,4)-byte string length).
<i>fd</i>	<i>fnt_def</i> (assign a number to a named font).
<i>pre</i>	Preamble.
<i>post</i>	<i>post</i> (begin postamble).
<i>post_post</i>	<i>post_post</i> (end postamble).
<i>opcode</i>	Undefined DVI command (250 to 255).

## SAMPLE DTL FILE

The following 2-line T<sub>E</sub>X file

Hello.

\bye

when processed with the commands

tex hello.tex

dv2dt hello.dvi hello.dtl

produces this DTL file:

variety sequences-6

pre 2 25400000 473628672 1000 27 ' TeX output 1995.03.02:2334'

bop 1 0 0 0 0 0 0 0 0 -1

[

d3 -917504

]

d4 42152922

[

d4 -41497562

[

r3 1310720

```

fd1 0 11374260171 655360 655360 0 5 '' 'cmr10'
fn0
(Hello.)
]
]
d3 1572864
[
r4 15229091
(1)
]
eop
post 42 25400000 473628672 1000 43725786 30785863 2 1
fd1 0 11374260171 655360 655360 0 5 'cmr10'
post_post 152 2 223 223 223 223

```

The command

```
dt2dv hello.dtl hello.dvi
```

will reconstruct the original DVI file.

## SEE ALSO

**dt2dv(1)**, **dvitype(1)**, **tex(1)**.

## FILES

\*.*dvi* binary T<sub>E</sub>X DVI file.

\*.*dtl* text representation of a T<sub>E</sub>X DVI file in *DVI Text Language* format.

## AUTHOR

**dv2dt** and **dt2dv(1)** were written by

Geoffrey Tobin  
 Department of Electronic Engineering  
 La Trobe University  
 Bundoora, Victoria 3083  
 Australia  
 Tel: +61 3 479 3736  
 FAX: +61 3 479 3025  
 Email: <G.Tobin@ee.latrobe.edu.au>

These manual pages were primarily written by

Nelson H. F. Beebe, Ph.D.  
 Center for Scientific Computing  
 Department of Mathematics  
 University of Utah  
 Salt Lake City, UT 84112  
 Tel: +1 801 581 5254  
 FAX: +1 801 581 4148  
 Email: <beebe@math.utah.edu>