NAME

dvisvgm − converts DVI files to the XML−based SVG format

SYNOPSIS

dvisvgm [*options*] *file* [.dvi]

DESCRIPTION

The program **dvisvgm** generates an SVG file from a given DVI file. DVI is the device independent output format produced by TeX and some other typesetting systems. Output from groff can be converted to DVI using grodvi.

The recent version of **dvisvgm** provides advanced conversion functionality but currently only one page can be converted even if the DVI file is made up of multiple pages. You can select the page to be processed with option **−p**. **dvisvgm** should properly convert all pages that are made up of fonts and rules only. Additionally, some sets of specials are understood as well. For a more detailed overview, see section **support of specials** below.

Since SVG is a vector based graphics format, **dvisvgm** tries to convert the glyph outlines of all used fonts into scalable path descriptions. The fastest way to do that is to extract the path information from PFB (PostScript Type 1) files. So if **dvisvgm** is able to find a PFB file for a used font, it will read the necessary information from it.

TeXs main source for font descriptions is Metafont which produces bitmap output .Thats why not all obtainable TeX fonts are available in PFB format. In these cases **dvisvgm** tries to vectorize Metafonts output by tracing the glyph bitmaps. The results are not as perfect as most (manually optimized) PFB outlines but are nonetheless really nice in most cases.

OPTIONS

−a, −−trace−all

This option forces dvisvgm to trace not just the actually needed glyphs but all glyphs of all bitmap fonts used in the DVI file. Since the tracing results are stored in the font cache, all following DVI conversions (without option **−−trace−all**) where these fonts are involved will be much faster. By default, dvisvgm traces only the actually needed glyphs and adds them to the cache.

Note

This option only takes affect if font caching is active. Thus, **−−trace−all** cannot be combined with option **−−cache=none**.

−b, −−bbox=*fmt*

Sets the bounding box of the generated graphic to the specified format. The parameter *fmt* takes either one of the format specifiers listed below or a sequence of four comma− or whitespace−separated length values *x1*, *y1*, *x2* and *y2*. The latter define two diagonal corners of the bounding box. Each length value consists of a floating point number and an optional length unit (pt, bp, cm, mm, in, or pc). If the unit is omitted, TeX points (pt) are assumed.

Its also possible to give only one length value *l*. In this case the minimal bounding box is computed and enlarged by adding (*−l*,*−l*) to the upper left and (*l*,*l*) to the lower right corner.

Alternatively, the following format specifiers are supported:

International DIN/ISO paper sizes

A*n*, B*n*, C*n*, D*n*, where *n* is a non−negative integer, e.g. A4 or a4 for DIN/ISO A4 format (210mm × 297mm).

North American paper sizes

invoice, executive, legal, letter, ledger

Special bounding box sizes

tab(:); lt lt lt lt lt lt. T{ **dvi** T}:T{ the page size stored in the DVI file will be used T} T{ **min**

T}:T{ the minimal bounding box will be computed and assigned T} T{ **none** T}:T{ no bounding box will be assigned T}

Page orientation

The default page orientation for DIN/ISO and American paper sizes is *portrait*, i.e. *width* < *height*. Appending **−landscape** or simply **−l** to the format string switches to *landscape* mode (*width* > *height*). For symmetry reasons you can also explicitly add **−portrait** or **−p** to indicate the default portrait format. Note that these suffixes are part of the size string and not separate options. Thus they must directly follow the above mentioned size specifiers without additional blanks. Furthermore, the orientation suffixes cant be used with **dvi**, **min** and **none**.

Note

Option **−b, −−bbox** only affects the bounding box and does not transform the page content. Hence if you choose a landscape format the page wont be rotated.

−c, −−scale=*sx*[,*sy*]

Scales page content horizontally by *sx* and vertically by *sy*. This option is equivalent to *−TS**sx*,*sy*.

−C, −−cache[=*dir*]

To speed up the conversion process of bitmap fonts, **dvisvgm** saves intermediate conversion information in cache files. By default, these files are stored in $HOME/.dvisvgm. If you prefer a different location use option **−−cache** to overwrite the default. Furthermore, it is also possible to disable the font caching mechanism completely with option **−−cache=none**. If argument *dir* is omitted, dvisvgm prints the path of the default cache directory and some information about the stored fonts.

−−libgs=filename

This option is only available if the Ghostscript library is not directly linked to dvisvgm and PostScript support was not completely disabled. In this case, dvisvgm tries to load the shared GS library dynamically during runtime. By default, it expects the librarys name to be libgs.so (Linux) or gsdll32.dll (Windows). Option **−−libgs** can be used to give a different name. Alternatively, the GS library name can also be set by the environment variable **LIBGS**. The latter has a less precedence than the command−line option.

−l, −−list−specials

Prints a list of registered special handlers and exits. Each handler processes a set of special statements belonging to the same category. The categories are identified by the prefix of the special statements. Its usually a leading word separated from the rest of the statement by a colon or a blank, e.g. *color* or *ps*.

−m, −−map−file=*file*

Sets the map file that is used to look up font names and encodings. **dvisvgm** does not provide its own map file but tries to use available ones coming with **dvips** or **dvipdfm**. If option **−m** is not given **dvisvgm** looks for ps2pk.map, dvipdfm.map, and psfonts.map (in this order). Without further options, the file specified by option **−m** will be used instead of the default maps. If you want **dvisgvm** to load the given file additionally to the default maps, add a leading + to the file/path, e.g. −−map−file=+myfonts.map. For further information about the file format see the manuals of **dvips** and **dvipdfm**.

−M, −−mag=*factor*

Sets the maginfication factor used for Metafont calls prior tracing the glyphs. The larger this value the better the tracing results. Nevertheless large magnification values can cause Metafont arithmetic errors due to number overflows. So use this option with care. The default setting usually produces nice results.

−n, −−no−fonts

If this option is given, **dvisvgm** doesnt create font elements but uses paths instead. The resulting SVG file is usually bigger but concurrently more compatible with most applications that dont support SVG fonts yet. This option implies **−−no−styles**.

−S, −−no−specials[=*names*]
Disable processing of special commands that are embedded in the DVI file. If no further parameter is given, all specials are ignored. To selectively disable sets of specials, an optional comma−separated list of names can be appended to this option. A *name* is a unique identifier referencing a special handler. Option **−−list−specials** lists all currently available handlers and their names. All unsupported special statements are silently ignored.

−−no−styles
By default, **dvisvgm** uses CSS styles and class attributes to reference fonts because its more compact than repeatedly set the complete font information in each text element. However, if you prefer direct font references, the default behavior can be disabled with option **−−no−styles**.

−−no−mktexmf
Suppresses the generation of missing font files. If **dvisvgm** cant find a font file through the kpathsea lookup mechanism, it calls the external tools mktextfm or mktexmf by default. This option disables these calls.

−o, −−output=*file*
Sets the name of the output file.

−p, −−page=*number*
With this option you can choose the page to be processed. Please consider that the parameter of this option dont refer to the page numbers printed on the page. Instead the physical page count is expected, where the first page always gets number 1. If this option is omitted, the first page will be selected.

−P, −−progress[=*skip*]
Enables a simple progress indicator shown when DVI specials are processed, since this can be a time−consuming operation, especially when PostScript specials are involved. The optional non−negative parameter *skip* controls the indicator length: Every time a set of *skip* specials has been processed, an indicator character is printed. Therefore, smaller values lead to a longer, more frequently updated progress indicator.

−r, −−rotate=*angle*
Rotates page content clockwise by *angle* degrees around the page center. This option is equivalent to **−TR***angle*.

−s, −−stdout
Dont write output to a file but redirect it to **stdout**.

−t, −−translate=*tx*[,*ty*]
Translates (moves) page content in direction of vector (*tx*,*ty*). This option is equivalent to **−TT***tx*,*ty*.

−T, −−transform=*commands*
Applies a sequence of transformations to the SVG content. Each transformation is described by a *command* that begins with a capital letter followed by a list of comma−separated parameters. Following transformation commands are supported:

T *tx*[,*ty*]
Translates (moves) page in direction of vector (*tx*,*ty*). If *ty* is omitted, *ty*=0 is assumed. The expected unit length of *tx* and *ty* are TeX points (1pt = 1/72.27in). However, there are several constants defined to simplify the unit conversion (see below).

S *sx*[,*sy*]
Scales page horizontally by *sx* and vertically by *sy*. If *sy* is omitted *sy*=*sx* is assumed.

R *angle*[,*x*,*y*]
Rotates page clockwise by *angle* degrees around point (*x*,*y*). If the optional arguments *x* and *y* are omitted the page will be rotated around its center depending on the chosen page format. When option **−bnone** is given, the rotation center is origin (0,0).

KX *angle*
Skews page along the *x*−axis by *angle* degrees. Argument *angle* can take any value except

90+180*k*, where *k* is an integer.

KY *angle*
Skews page along the *y*−axis by *angle* degrees. Argument *angle* can take any value except 90+180*k*, where *k* is an integer.

FH [*y*]
Mirrors (flips) page at the horizontal line through point (0,*y*). Omitting the optional argument leads to *y*=*h*/2, where *h* denotes the page height (see *pre−defined constants* below).

FV [*x*]
Mirrors (flips) page at the vertical line through point (*x*,0). Omitting the optional argument leads to *x*=*w*/2, where *w* denotes the page width (see *pre−defined constants* below).

M *m1*,...,*m6*
Applies a transformation described by the 3×3 matrix ((*m1*,*m2*,*m3*),(*m4*,*m5*,*m6*),(0,0,1)), where the inner triples denote the rows.

Note

All transformation commands of option **−T, −−transform** are applied in the order of their appearance. Multiple commands can optionally be separated by spaces. In this case the whole transformation string has to be enclosed in double quotes. All parameters are expressions of floating point type. You can either give plain numbers or arithmetic terms combined by the operators **+** (addition), **−** (substraction), ***** (multiplication), **/** (division) or **%** (modulo) with common associativity and precedence rules. Parentheses may be used as well.

Additionally, some pre−defined constants are provided: tab(:); lt lt lt lt lt lt lt lt. T{ **ux** T}:T{ horizontal position of upper left page corner in TeX point units T} T{ **uy** T}:T{ vertical position of upper left page corner in TeX point units T} T{ **h** T}:T{ page height in TeX point units (0 in case of **−bnone**) T} T{ **w** T}:T{ page width in TeX point units (0 in case of **−bnone**) T}

Furthermore, you can use the length constants **pt**, **mm**, **cm** and **in**, e.g. 2cm or 1.6in. Thus, option −TT1in,0R45 moves the page content 1 inch to the right and rotates it by 45 degrees around the page center afterwards.

For single transformations you can also use options **−c**, **−t** and **−r**. Note that the order in which these options are given is not significant, i.e. you cant use them to describe transformation sequences. They are simply independent shorthand options for common transformations.

SUPPORT OF SPECIALS

dvisvgm supports several sets of special commands that can be used to enrich DVI files with additional features, like color, graphics or hyperlinks. The evaluation of special commands is done by various handlers, where each handler is responsible for all special statements of the same command set, i.e. commands beginning with the same prefix. To get a list of actually provided special handlers, use option **−−list−specials** (see above).

bgcolor
Special statement for changing the background/page color. Since SVG 1.1 doesnt support background colors, **dvisvgm** inserts a rectangle of the chosen color. In the current version, this rectangle always gets the size of the minimal bounding box. This command is part of the color special set but is handled separately in order to let the user turn it off. For an overview of the command syntax, see the documentation of **dvips**, for instance.

color
Statements of this command set provide instructions to change the text/paint color. For an overview of the exact syntax, see the documentation of **dvips**, for instance.

dvisvgm

dvisvgm offers its own small set of specials. The following list gives a brief overview.

dvisvgm:raw *text*
: Adds an arbitrary sequence of characters to the SVG output. **dvisvgm** does not perform any validation here, thus the user of this special has to ensure that the resulting SVG is still valid. The parameter *text* may contain the macros **{?x}**, **{?y}**, and **{?color}** which are expanded to the current *x* or *y* coordinate and the current color, respectively. Also, the macro **{?nl}** expands to a newline character.

dvisvgm:img *width height file*
: Creates an image element at the current graphic position referencing the given file. JPEG, PNG, and SVG images can be used here. However, **dvisvgm** does not check the file format or the file name suffix. The lengths *width* and *height* must be given as plain floating point numbers in TeX point units (1in = 72.27pt).

dvisvgm:bbox *width height* [*depth*]
: Updates the bounding box of the current page by embedding a virtual rectangle (*x*, *y*, *width*, *height*) where the lower left corner is located at the current DVI drawing position (*x*,*y*). If the optional parameter *depth* is specified, dvisvgm embeds a second rectangle (*x*, *y*, *width*, −*depth*). The lengths *width*, *height* and *depth* must be given as plain floating point numbers in TeX point units (1in = 72.27pt). Depending on size and position of the virtual rectangle, this command either enlarges the overall bounding box or leaves it as is. Its not possible to reduce its extent. This special should be used in conjunction with **dvisvgm:raw** in order to properly update the viewport of the page.

dvisvgm:bbox a[bs] *x1 y1 x2 y2*
: This variant of the bbox special updates the bounding box by embedding a virtual rectangle (*x1*,*y1*,*x2*,*y2*). The points (*x1*,*y1*) and (*x2*,*y2*) denote two diagonal corners of the rectangle given in TeX point units.

dvisvgm:bbox f[ix] *x1 y1 x2 y2*
: This variant of the bbox special assigns an absolute (final) bounding box to the resulting SVG. After executing this command, dvisvgm doesnt further alter the bounding box coordinates, except this special is called again later. The points (*x1*,*y1*) and (*x2*,*y2*) denote two diagonal corners of the rectangle given in TeX point units.

  The following TeX snippet adds two raw SVG elements to the output and updates the bounding box accordingly:

```
\special{dvisvgm:raw <circle cx=´{?x}´ cy=´{?y}´ r=´10´ stroke=´black´ fill=´red´/>}
\special{dvisvgm:bbox 20 10 10}

\special{dvisvgm:raw <path d=´M50 200 L10 250 H100 Z´ stroke=´black´ fill=´blue´/>}
\special{dvisvgm:bbox abs 10 200 100 250}
```

em

These specials were introduced with the emTeX distribution by Eberhard Mattes. They provide line drawing statements, instructions for embedding MSP, PCX, and BMP image files, as well as two PCL commands. **dvisvgm** supports only the line drawing statements, all other em specials are silently ignored. A description of the command syntax can be found in the DVI driver documentation coming with emTeX (see CTAN).

ps

The famous DVI driver **dvips** introduced its own set of specials in order to embed PostScript code into DVI files, which greatly improves the capabilities of DVI documents. One aim of **dvisvgm** is to completely evaluate the PostScript code and to convert a large amount of it to SVG. Since PostScript is a rather complex language, **dvisvgm** does not try to implement its own PostScript interpreter but uses Ghostscript instead. If the Ghostscript library was not linked while building dvisvgm, it is looked up and dynamically loaded during runtime. In this case, dvisvgm looks for libgs.so on Linux systems, and

gsdll32.dll on Windows. You can override these default file names with the environment variable LIBGS. The library must be installed and reachable through the PATH environment variable. If it cannot be found, the evaluation of PostScript specials is disabled. Use option **−−list−specials** to check whether PS support is available, i.e. the entry *ps* is present.

tpic
The TPIC special set defines instructions for drawing simple geometric objects. Some LaTeX packages, like eepic and tplot, use these specials to describe graphics.

EXAMPLES

dvisvgm file

Converts first page of *file.dvi* to *file.svg*.

dvisvgm −z file

Converts first page of *file.dvi* to *file.svgz* with default compression level 9.

dvisvgm −p5 −z3 −ba4−l −onewfile file

Converts fifth page of *file.dvi* to *newfile.svgz* with compression level 3. The bounding box is set to DIN/ISO A4 in landscape format.

dvisvgm −−transform="R20,w/3,2h/5 T1cm,1cm S2,3" file

Converts first page of *file.dvi* to *file.svg* where three transformations are applied.

ENVIRONMENT

dvisvgm uses the **kpathsea** library for locating the files that it opens. Hence, the environment variables described in the librarys documentation influence the converter.

If dvisvgm was linked without the Ghostscript library and PostScript support has not been disabled, the shared Ghostscript library is looked up during runtime. The environment variable LIBGS can be used to specify the file name of library.

The pre−compiled Windows version of **dvisvgm** requires a working installation of MiKTeX 2.7 or above. To enable evaluation of PostScript specials, the original Ghostscript DLL must be present.

FILES

The location of the following files is determined by the kpathsea library. To check the actual kpathsea configuration you can use the **kpsewhich** utility. tab(:); lt lt lt lt lt lt lt lt lt lt lt lt lt lt lt lt lt. T{

***.enc** T}:T{

Font encoding files T} T{

***.fgd** T}:T{

Font glyph data files (cache files created by dvisvgm) T} T{

***.map** T}:T{

Font map files T} T{

***.mf** T}:T{

Metafont input files T} T{

***.pfb** T}:T{

PostScript Type 1 font files T} T{

***.pro** T}:T{

PostScript header/prologue files T} T{

***.tfm** T}:T{

TeX font metric files T} T{

***.ttf** T}:T{

TrueType font files T} T{

***.vf** T}:T{

Virtual font files T}

SEE ALSO

tex(1), mf(1), mktexmf(1), grodvi(1), potrace(1), and the **kpathsea library** info documentation.

RESOURCES

Project home page

http://dvisvgm.sourceforge.net

SourceForge project site

http://sourceforge.net/projects/dvisvgm

AUTHOR

Written by Martin Gieseking <martin.gieseking@uos.de>

COPYING