NAME

grn − groff preprocessor for gremlin files

SYNOPSIS

grn [**−Cv**] [**−T** *dev*] [**−M** *dir*] [**−F** *dir*] [*file* ...]

DESCRIPTION

grn is a preprocessor for including *gremlin* pictures in *groff* input. *grn* writes to standard output, processing only input lines between two that start with **.GS** and **.GE**. Those lines must contain *grn* commands (see below). These commands request a *gremlin* file, and the picture in that file is converted and placed in the *troff* input stream. The **.GS** request may be followed by a C, L, or R to center, left, or right justify the whole *gremlin* picture (default justification is center). If no *file* is mentioned, the standard input is read. At the end of the picture, the position on the page is the bottom of the *gremlin* picture. If the *grn* entry is ended with **.GF** instead of **.GE**, the position is left at the top of the picture.

Please note that currently only the −me macro package has support for **.GS**, **.GE**, and **.GF**.

OPTIONS

Whitespace is permitted between a command-line option and its argument.

−T*dev* Prepare output for printer *dev*. The default device is **ps**. See **groff**(1) for acceptable devices.

−M*dir* Prepend *dir* to the default search path for *gremlin* files. The default path is (in that order) the current directory, the home directory, */usr/lib/groff/site−tmac*, */usr/share/groff/site−tmac*, and */usr/share/groff/1.22.4/tmac*.

−F*dir* Search *dir* for subdirectories *dev*name (*name* is the name of the device) for the *DESC* file before the default font directories */usr/share/groff/site−font*, */usr/share/groff/1.22.4/font*, and */usr/lib/font*.

−C Recognize **.GS** and **.GE** (and **.GF**) even when followed by a character other than space or newline.

−v Print the version number.

GRN COMMANDS

Each input line between **.GS** and **.GE** may have one *grn* command. Commands consist of one or two strings separated by white space, the first string being the command and the second its operand. Commands may be upper or lower case and abbreviated down to one character.

Commands that affect a picture's environment (those listed before **default**, see below) are only in effect for the current picture: The environment is reinitialized to the defaults at the start of the next picture. The commands are as follows:

1 *N*
2 *N*
3 *N*
4 *N* Set *gremlin*'s text size number 1 (2, 3, or 4) to *N* points. The default is 12 (16, 24, and 36, respectively).

roman *f*
italics *f*
bold *f*
special *f*
Set the roman (italics, bold, or special) font to *troff*'s font *f* (either a name or number). The default is R (I, B, and S, respectively).

l *f*
stipple *f*
Set the stipple font to *troff*'s stipple font *f* (name or number). The command **stipple** may be abbreviated down as far as 'st' (to avoid confusion with **special**). There is *no* default for stipples (unless one is set by the default command), and it is invalid to include a *gremlin* picture with polygons without specifying a stipple font.

x *N*

scale *N* Magnify the picture (in addition to any default magnification) by *N*, a floating point number larger than zero. The command **scale** may be abbreviated down to 'sc'.

narrow *N*
medium *N*
thick *N*

Set the thickness of *gremlin*'s narrow (medium and thick, respectively) lines to *N* times 0.15pt (this value can be changed at compile time). The default is 1.0 (3.0 and 5.0, respectively), which corresponds to 0.15pt (0.45pt and 0.75pt, respectively). A thickness value of zero selects the smallest available line thickness. Negative values cause the line thickness to be proportional to the current point size.

pointscale <*off/on*>

Scale text to match the picture. Gremlin text is usually printed in the point size specified with the commands **1**, **2**, **3**, or **4**, regardless of any scaling factors in the picture. Setting **pointscale** will cause the point sizes to scale with the picture (within *troff*'s limitations, of course). An operand of anything but *off* will turn text scaling on.

default Reset the picture environment defaults to the settings in the current picture. This is meant to be used as a global parameter setting mechanism at the beginning of the *troff* input file, but can be used at any time to reset the default settings.

width *N*

Forces the picture to be *N* inches wide. This overrides any scaling factors present in the same picture. '**width** *0*' is ignored.

height *N*

Forces picture to be *N* inches high, overriding other scaling factors. If both 'width' and 'height' are specified the tighter constraint will determine the scale of the picture. **Height** and **width** commands are not saved with a **default** command. They will, however, affect point size scaling if that option is set.

file *name*

Get picture from *gremlin* file *name* located the current directory (or in the library directory; see the **−M** option above). If two **file** commands are given, the second one overrides the first. If *name* doesn't exist, an error message is reported and processing continues from the **.GE** line.

NOTES ABOUT GROFF

Since *grn* is a preprocessor, it doesn't know about current indents, point sizes, margins, number registers, etc. Consequently, no *troff* input can be placed between the **.GS** and **.GE** requests. However, *gremlin* text is now processed by *troff*, so anything valid in a single line of *troff* input is valid in a line of *gremlin* text (barring '.' directives at the beginning of a line). Thus, it is possible to have equations within a *gremlin* figure by including in the *gremlin* file *eqn* expressions enclosed by previously defined delimiters (e.g. *$$*).

When using *grn* along with other preprocessors, it is best to run *tbl* before *grn*, *pic*, and/or *ideal* to avoid overworking *tbl*. *Eqn* should always be run last.

A picture is considered an entity, but that doesn't stop *troff* from trying to break it up if it falls off the end of a page. Placing the picture between 'keeps' in −me macros will ensure proper placement.

grn uses *troff*'s number registers **g1** through **g9** and sets registers **g1** and **g2** to the width and height of the *gremlin* figure (in device units) before entering the **.GS** request (this is for those who want to rewrite these macros).

GREMLIN FILE FORMAT

There exist two distinct *gremlin* file formats, the original format from the *AED* graphic terminal version, and the *SUN* or *X11* version. An extension to the *SUN*/*X11* version allowing reference points with negative coordinates is **not** compatible with the *AED* version. As long as a *gremlin* file does not contain negative coordinates, either format will be read correctly by either version of *gremlin* or *grn*. The other difference from *SUN*/*X11* format is the use of names for picture objects (e.g., POLYGON, CURVE) instead of

numbers. Files representing the same picture are shown in Table 1 in each format.

sungremlinfile	gremlinfile
0 240.00 128.00	0 240.00 128.00
CENTCENT	2
240.00 128.00	240.00 128.00
185.00 120.00	185.00 120.00
240.00 120.00	240.00 120.00
296.00 120.00	296.00 120.00
*	−1.00 −1.00
2 3	2 3
10 A Triangle	10 A Triangle
POLYGON	6
224.00 416.00	224.00 416.00
96.00 160.00	96.00 160.00
384.00 160.00	384.00 160.00
*	−1.00 −1.00
5 1	5 1
0	0
−1	−1

Table 1. File examples

- The first line of each *gremlin* file contains either the string **gremlinfile** (*AED* version) or **sungremlinfile** (*SUN*/*X11*)
- The second line of the file contains an orientation, and **x** and **y** values for a positioning point, separated by spaces. The orientation, either **0** or **1**, is ignored by the *SUN*/*X11* version. **0** means that *gremlin* will display things in horizontal format (drawing area wider than it is tall, with menu across top). **1** means that *gremlin* will display things in vertical format (drawing area taller than it is wide, with menu on left side). **x** and **y** are floating point values giving a positioning point to be used when this file is read into another file. The stuff on this line really isn't all that important; a value of "1 0.00 0.00" is suggested.
- The rest of the file consists of zero or more element specifications. After the last element specification is a line containing the string "−1".
- Lines longer than 127 characters are chopped to this limit.

ELEMENT SPECIFICATIONS

- The first line of each element contains a single decimal number giving the type of the element (*AED* version) or its ASCII name (*SUN*/*X11* version). See Table 2.

gremlin File Format − Object Type Specification

AED Number	*SUN*/*X11* Name	Description
0	BOTLEFT	bottom-left-justified text
1	BOTRIGHT	bottom-right-justified text
2	CENTCENT	center-justified text
3	VECTOR	vector
4	ARC	arc
5	CURVE	curve
6	POLYGON	polygon
7	BSPLINE	b-spline
8	BEZIER	Bézier

10	TOPLEFT	top-left-justified text
11	TOPCENT	top-center-justified text
12	TOPRIGHT	top-right-justified text
13	CENTLEFT	left-center-justified text
14	CENTRIGHT	right-center-justified text
15	BOTCENT	bottom-center-justified text

Table 2.
Type Specifications in *gremlin* Files

- After the object type comes a variable number of lines, each specifying a point used to display the element. Each line contains an x-coordinate and a y-coordinate in floating point format, separated by spaces. The list of points is terminated by a line containing the string "−1.0 −1.0" (*AED* version) or a single asterisk, "*" (*SUN*/*X11* version).
- After the points comes a line containing two decimal values, giving the brush and size for the element. The brush determines the style in which things are drawn. For vectors, arcs, and curves there are six valid brush values:

1 –	thin dotted lines
2 –	thin dot-dashed lines
3 –	thick solid lines
4 –	thin dashed lines
5 –	thin solid lines
6 –	medium solid lines

  For polygons, one more value, 0, is valid. It specifies a polygon with an invisible border. For text, the brush selects a font as follows:

1 –	roman (R font in groff)
2 –	italics (I font in groff)
3 –	bold (B font in groff)
4 –	special (S font in groff)

  If you're using *grn* to run your pictures through *groff*, the font is really just a starting font: The text string can contain formatting sequences like "\fI" or "\d" which may change the font (as well as do many other things). For text, the size field is a decimal value between 1 and 4. It selects the size of the font in which the text will be drawn. For polygons, this size field is interpreted as a stipple number to fill the polygon with. The number is used to index into a stipple font at print time.
- The last line of each element contains a decimal number and a string of characters, separated by a single space. The number is a count of the number of characters in the string. This information is only used for text elements, and contains the text string. There can be spaces inside the text. For arcs, curves, and vectors, this line of the element contains the string "0".

NOTES ON COORDINATES

gremlin was designed for *AED*s, and its coordinates reflect the *AED* coordinate space. For vertical pictures, x-values range 116 to 511, and y-values from 0 to 483. For horizontal pictures, x-values range from 0 to 511 and y-values range from 0 to 367. Although you needn't absolutely stick to this range, you'll get best results if you at least stay in this vicinity. Also, point lists are terminated by a point of (−1, −1), so you shouldn't ever use negative coordinates. *gremlin* writes out coordinates using format "%f1.2"; it's probably a good idea to use the same format if you want to modify the *grn* code.

NOTES ON SUN/X11 COORDINATES

There is no longer a restriction on the range of coordinates used to create objects in the *SUN/X11* version of *gremlin*. However, files with negative coordinates **will** cause problems if displayed on the *AED*.

FILES

*/usr/share/groff/1.22.4/font/dev*name*/DESC*
Device description file for device *name*.

AUTHORS

David Slattengren and Barry Roitblat wrote the original Berkeley *grn*. Daniel Senderowicz and Werner Lemberg modified it for *groff*.

SEE ALSO

gremlin(1), **groff**(1), **pic**(1), **ideal**(1)