

NAME

`gxl2gv`, `gv2gxl` – GXL-GV converters

SYNOPSIS

`gxl2gv` [`-gd?`] [`-ooutfile`] [*files*]

`gv2gxl` [`-gd?`] [`-ooutfile`] [*files*]

DESCRIPTION

`gxl2gv` converts between graphs represented in GXL and in the GV language. Unless a conversion type is specified using a flag, **`gxl2gv`** will deduce the type of conversion from the suffix of the input file, a ".gv" (or a ".dot") suffix causing a conversion from GV to GXL, and a ".gxl" suffix causing a conversion from GXL to GV. If no suffix is available, e.g. when the input is from a pipe, and no flags are used then **`gxl2gv`** assumes the type of the input file from its executable name so that **`gxl2gv`** converts from GXL to GV, and **`gv2gxl`** converts from GV to GXL.

GXL supports a much richer graph model than GV. **`gxl2gv`** will attempt to map GXL constructs into the analogous GV construct when this is possible. If not, the GXL information is stored as an attribute. The intention is that applying **`gxl2gv|gv2gxl`** is semantically equivalent to the identity operator.

OPTIONS

The following options are supported:

`-g` The command name and input file extensions are ignored, the input is taken as a GV file and a GXL file is generated.

`-d` The command name and input file extensions are ignored, the input is taken as a GXL file and a GV file is generated.

`-?` Prints usage information and exits.

`-o outfile`

If specified, the output will be written into the file *outfile*. Otherwise, output is written to standard out.

OPERANDS

The following operand is supported:

files Names of files containing 1 or more graphs in GXL or GV. If no *files* operand is specified, the standard input will be used.

RETURN CODES

Both **`gxl2gv`** and **`gv2gxl`** return **0** if there were no problems during conversion; and non-zero if any error occurred.

BUGS

`gxl2gv` will only convert in one direction even if given multiple files with varying suffixes.

The conversion can only handle one graph per GXL file.

There are some GXL constructs which **`gxl2gv`** cannot handle.

AUTHORS

Krishnam Pericherla <kp@research.att.com>

Emden R. Gansner <erg@research.att.com>

SEE ALSO

`dot(1)`, `libgraph(3)`, `libagraph(3)`, `neato(1)`, `twopi(1)`