

# The `graphviz` package\*

Derek Rayside `<drayside@uwaterloo.ca>`  
with contributions from Ralf Hemmecke `<ralf@hemmecke.de>`

September 25, 2014

## 1 Introduction

`graphviz.sty` is a  $\text{\LaTeX}$  package for writing `graphviz/dot/neato` graphs inside of  $\text{\LaTeX}$  documents. `graphviz.sty` was inspired by a feature that Daniel Jackson added to his `tagger` text markup tool.

`graphviz` is a freely available package for doing automated graph layout from AT&T Research, distributed under the Common Public License (CPL). `graphviz` includes the `dot` and `neato` programs, which read a textual description of a graph and produces a graphical rendering of it. Many different graphics formats, include PostScript, are supported.

There are two main web pages for the `graphviz` project:

- <http://www.graphviz.org>
- <http://www.research.att.com/sw/tools/graphviz/>

`graphviz.sty` is provided as-is, with no warranty or claim to fitness for any purpose, use at your own risk, etc. `graphviz.sty` is distributed under the  $\text{\LaTeX}$  Project Public License.

## 2 Example

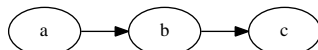
Put this in your document:

```
\digraph[scale=0.5]{abc}{rankdir=LR; a->b->c;}
```

Run these commands (only the first run needs `-shell-escape`):

```
latex -shell-escape main.tex  
latex main.tex
```

And here's what you get:



---

\*This document corresponds to `graphviz` v0.94, dated 2013/08/15.

### 3 Usage

`\digraph[i]{n}{g}` The `\digraph` (`dot`) and `\neatograph` (`neato`) commands take three arguments:

- [*i*] parameters to the `\includegraphics` command that will include the PostScript file of the graph [this is optional]: eg, `'scale=0.5'`
- {*n*} the name of the graph; a file `name.dot` is created, and a file `name.ps` is expected to be produced from `dot`: eg, `'MyGraph'`  
*{n}* has to be a valid file name and a valid identifier name.
- {*g*} the graph, specified in the `dot/graphviz` language:  
eg, `'rankdir=LR; a->b->c;'`

### 4 Options

**singlefile** L<sup>A</sup>T<sub>E</sub>X has a small number of file handles (about 16 or so). So if you can't have too many digraphs in your tex file before you run out of file handles. The `singlefile` option is a work-around: it writes all of your digraphs to a single file (`tmpmaster.graphviz`), and then uses `gvpr` to split that file into individual dot files for processing by `dot`.

The `GVPR` commands are all written to a second file (`tmpmaster.gvpr`), which is executed once the `tmpmaster.graphviz` file has been closed.

`gvpr` does not seem to be packaged with the Windows version of `dot`.

```

1 \newif\ifsinglefile
2 \DeclareOption{singlefile}{
3   \singlefiletrue
4   \AtBeginDocument{% open a new file handle
5     \newwrite\masterdotfile%
6     \immediate\openout\masterdotfile=\@tmpdir tmpmaster.graphviz%
7     \newwrite\mastergvprfile%
8     \immediate\openout\mastergvprfile=\@tmpdir tmpmaster.gvpr}
9   \AtEndDocument{% close the file
10    % close the dot file and the gvpr file
11    \immediate\closeout\masterdotfile%
12    \immediate\closeout\mastergvprfile%
13    % execute the gvpr file
14    \immediate\write18{gvpr -f \@tmpdir tmpmaster.gvpr \@tmpdir tmpmaster.graphviz}%
15  }}
```

**psfrag** The `psfrag` option uses the `psfrag` package to enable you to overlay T<sub>E</sub>X fragments over included postscript files, such as those generated via the `\digraph` command.

The `ladot` script from Brighten Godfrey uses Perl to extend the syntax of the `graphviz` language with T<sub>E</sub>X fragments, and `psfrag` to super-impose those fragments.

The `psfrag` option requires `sed`. `psfrag` seems to only work with `dvips`: ie, it is not compatible with `pdflatex` or `dvipdfm`. The PDF files produced by `LATEX/psfrag/ps2pdf` seem to view ok with Acrobat, but not with `gv`. Oddly, the PS files produced this way work in `gv`.

Put this in your document:

```
\psfrag{x2}[cc][cc]{$x^2$}
\digraph{xy}{rankdir=LR; x2->y;}
```

And here's what you get:



```
16 \newif\ifpsfrag
17 \DeclareOption{psfrag}{\psfragtrue }
```

`ps` Tell Graphviz to generate Postscript files as output.

```
18 \newcommand{\@outext}{ps}
19 \newcommand{\@outextspace}{ps }
20 \DeclareOption{ps}{
21     \renewcommand{\@outext}{ps}
22     \renewcommand{\@outextspace}{ps }}
```

`pdf` Tell Graphviz to generate PDF files as output.

```
23 \DeclareOption{pdf}{%
24     \renewcommand{\@outext}{pdf}%
25     \renewcommand{\@outextspace}{pdf }}
```

`tmpdir` Write all generated files in `./tmp/`

```
26 \newcommand{\@tmpdir}{}
27 \DeclareOption{tmpdir}{%
28     \immediate\write18{mkdir ./tmp}%
29     \renewcommand{\@tmpdir}{./tmp/}}
```

**Set the default options**

```
30 \ExecuteOptions{ps}
31 \ProcessOptions\relax % LaTeX class guide says it is wise to relax
```

## 5 Implementation

### 5.1 Required Packages

This package requires `graphicx` to include PostScript renderings of graphs.

```

32 \RequirePackage{graphicx}
33 \ifpsfrag \RequirePackage{psfrag} \fi

```

## 5.2 Command Implementation

`\digraph` This is the command the user uses for `dot`.

It is very important that this command is not defined with 3 parameters although it will be used with 3 parameters in the form `\digraph[OPTIONS]{FILENAME}{GRAPH}`. The reason is that the catcode for `^M` must be changed *before* `TEX` reads the `GRAPH` argument.

The order of the command (first `\inputdigraph` then `\@digraph`) may look a bit odd, but it simplifies the code. In order to include the digraph, `LATEX` has to be run at least two times anyway. In the first run the file `dot` will be generated and only the second run the digraph will be included.

```

34 \newcommand{\digraph}[2][scale=1]{
35   \inputdigraph[#1]{#2}{dot}%      % Include the generated ps/pdf.
36   \@digraph{digraph}{#2}%          % Generate the .dot file.
37 }

```

`\neatograph` This is the command the user uses for `neato`. The syntax is the same as for `\digraph`.

```

38 \newcommand{\neatograph}[2][scale=1]{
39   \inputdigraph[#1]{#2}{neato}%    % Include the generated ps/pdf.
40   \@digraph{graph}{#2}%             % Generate the .dot file.
41 }

```

`\@digraph` Internal implementation.

The macro `\@digraph` prepares the actual output of the digraph to a file (which is done by `\@@digraph`) by a special treatment of the newline character. Before entering `\@@digraph`, the input newline character (`^M`) is made active, and redefined to expand to `^^J`. Note that `\@digraph` has a `\begingroup` that is closed in `\@@digraph`.

The purpose of this is to preserve line breaks in the digraph.

```

42 \begingroup
43   \catcode'\^M=\active%
44   \gdef\@digraph{\begingroup\catcode'\^M=\active\def^^M{^^J}\@@digraph}%
45 \endgroup

```

`\@@digraph` Internal implementation.

The parameters of the macro `\@@digraph` are the `TYPE`, `FILENAME` and `GRAPH` of the initial `\digraph[OPTIONS]{FILENAME}{GRAPH}`. Note that if

`\@@digraph` is entered the `^^M` character is active. Thus every newline character (`^^M`) in the following macro is hidden through a `%` sign at the end of line.

```

46 \def\@@digraph#1#2#3{%
47   \ifsinglefile% write the graph to the master file
48     \expandafter\def\csname -\endcsname{\string\n}%
49     \immediate\write\masterdotfile{#1 #2 {#3}}%
50     \immediate\write\mastergvprfile{BEG_G { if ($.name == "#2") {writeG($G,"\@tmpdir#2.d
51   \else% open a new file handle
52     \newwrite\dotfile%
53     \immediate\openout\dotfile=\@tmpdir#2.dot%
54     \expandafter\def\csname -\endcsname{\string\n}%
55     \immediate\write\dotfile{#1 #2 {#3}}%
56     \immediate\closeout\dotfile%
57   \fi%
58 % Here comes the closing \endgroup that closes the group opened in \@digraph.
59   \endgroup}%
60 % Now ^^M is no longer active.

```

`\inputdigraph` This is usually only called by `\digraph`, but may be called by the user.

The purpose is to include the ps/pdf rendering of the graph if it exists, or to give instructions on how to generate it.

```

61 \newcommand{\inputdigraph}[3][scale=1]{
62   % execute dot or neato (nb: requires latex -shell-escape)
63   \immediate\write18{#3 -T\@outextspace -o \@tmpdir#2.\@outextspace \@tmpdir#2.dot}
64   \IfFileExists{\@tmpdir#2.\@outext}{ % the postscript/pdf exists: include it
65     \ifpsfrag
66       % per the ladot 2.2 source code, psfrag has a problem with
67       % graphviz 2.2, and some sed hackery is necessary to work around
68       \write18{sed -ibackup -e "s/xshow/pop show/g" \@tmpdir#2.ps}
69     \fi
70     \includegraphics[#1]{\@tmpdir#2.\@outext}
71   }
72   % else: the postscript/pdf doesn't exist: tell the user how to create it
73   {
74     \fbox{ \begin{tabular}{l}
75       The file \texttt{\@tmpdir#2.\@outext} hasn't been created from
76       \texttt{\@tmpdir#2.dot} yet. \\
77       Run '\texttt{dot -T\@outextspace -o \@tmpdir#2.\@outextspace \@tmpdir#2.dot}
78       to create it. \\
79       Or invoke \LaTeX\ with the \texttt{-shell-escape} option
80       to have this done automatically. \\
81       \end{tabular}}
82   }
83 }

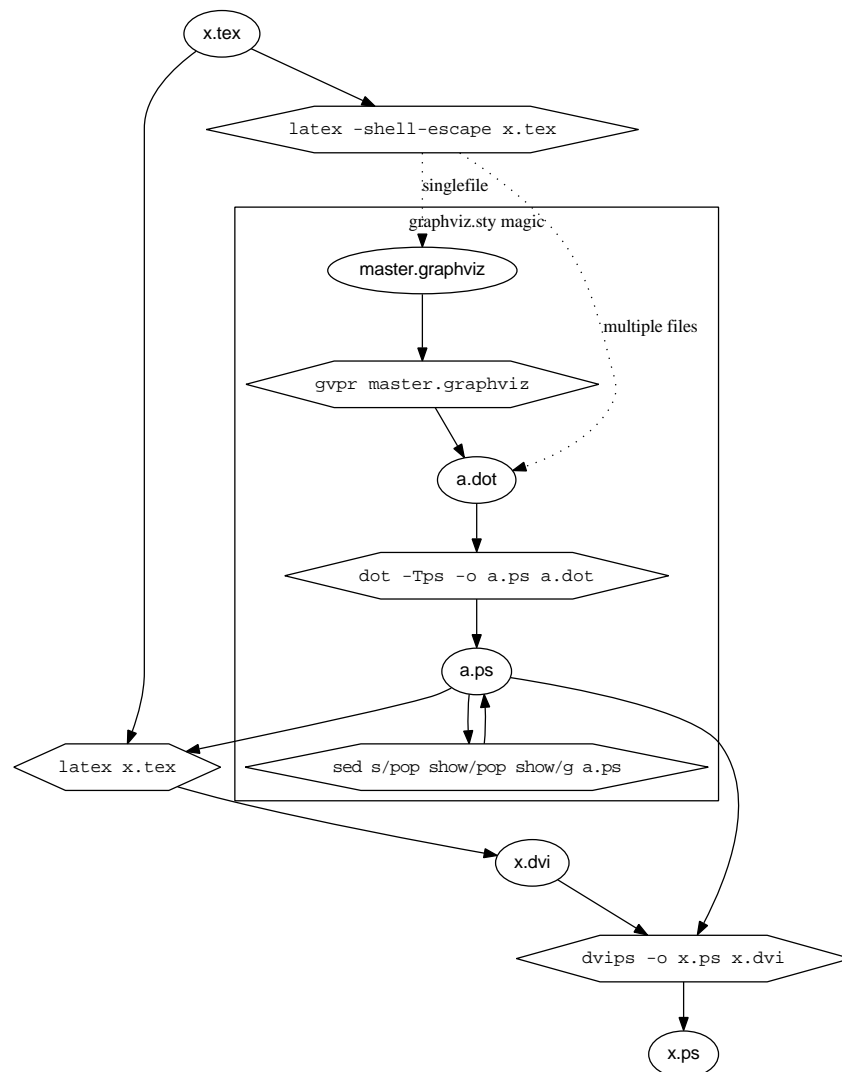
```

### 5.3 Process

`\digraph` writes out a dot file, and then invokes dot on it.

Note: `\digraph` can only invoke `dot` if the `LATEX` was invoked with the `-shell-escape` option, to enable execution of external programs. If you do not want to allow `LATEX` to execute external programs, then you will have to invoke `dot` yourself. `graphviz` will also need to execute `gvpr` if the `singlefile` option has been selected, and `sed` if the `psfrag` option has been selected.

Here's a picture of the process (drawn with `dot`, naturally). The picture shows the process using `dvips`, but `pdflatex` is now also supported with the `pdf` option.



## Change History

v0.1		<b>singlefile</b> : now using gvpr instead of gawk to break out individual digraphs from master.graphviz .	3
General: Initial version	1		
v0.2			
\digraph: minor adjustments	5	v0.8	
\inputdigraph: minor adjustments	6	\inputdigraph: added psfrag support	6
v0.4		psfrag: added psfrag option	3
General: converted to dtx format	1	v0.9	
\digraph: new comments	5	\digraph: refactored for control-M by Ralf Hemmecke	5
v0.5		\neatograph: added support for neato	5
General: renamed package to dotla	1	v0.91	
\digraph: added automatic invocation of dot	5	\digraph: a bit of cleanup and modernization	5
v0.6		v0.92	
\digraph: added singlefile option	5	pdf: added pdf option	4
singlefile: added singlefile option	3	ps: added ps option (previously default behaviour)	4
v0.7		v0.93	
General: renamed package back to graphviz	1	tmpdir: added tmpdir option	4
\digraph: added backslash-hyphen line breaks by Ralf Hemmecke	5	v0.94	
now using gvpr instead of gawk to break out individual digraphs from master.graphviz	5	singlefile: writing gvpr commands to separate script to be executed when master.graphviz is closed	3
removed redundant invocation of dot from digraph; only inputdigraph needs to invoke dot	5		