NAME

latexmk − generate LaTeX document

SYNOPSIS

latexmk [options] [file ...]

DESCRIPTION

LatexMk completely automates the process of compiling a LaTeX document. Essentially, it is like a specialized relative of the general *make* utility, but one which determines dependencies automatically and has some other very useful features. In its basic mode of operation *latexmk* is given the name of the primary source file for a document, and it issues the appropriate sequence of commands to generate a .dvi, .ps, .pdf and/or hardcopy version of the document. It can also be set to run continuously with a suitable previewer; in that case the LaTeX program, etc, are rerun whenever one of the source files is modified, and the previewer updates the on-screen view of the compiled document.

Latexmk determines which are the source files by examining the log file. When *latexmk* is run, it examines properties of the source files, and if any have been changed since the last document generation, *latexmk* will run the various LaTeX processing programs as necessary. In particular, it will repeat the run of LaTeX (or pdflatex) often enough to resolve all cross references; depending on the macro packages used. With some macro packages and document styles four, or even more, runs may be needed. If necessary, *latexmk* will also run bibtex and/or makeindex. In addition, *latexmk* can be configured to generate other necessary files. For example, from an updated figure file it can automatically generate a file in encapsulated postscript or another suitable format for reading by LaTeX.

Latexmk has two different previewing options. In the simple **-pv** option, a dvi, postscript or pdf previewer is automatically run after generating the dvi, postscript or pdf version of the document. The type of file to view is selected according to configuration settings and command line options.

The second previewing option is the powerful **-pvc** option (mnemonic: "preview continuously"). In this case, *latexmk* runs continuously, regularly monitoring all the source files to see if any have changed. Every time a change is detected, *latexmk* runs all the programs necessary to generate a new version of the document. A good previewer (like *gv*) will then automatically update its display. Thus the user can simply edit a file and, when the changes are written to disk, *latexmk* completely automates the cycle of updating the .dvi (and possibly the .ps and .pdf) file, and refreshing the previewer's display. It's not quite WYSIWYG, but usefully close.

For other previewers, the user may have to manually make the previewer update its display, which can be (some versions of xdvi and gsview) as simple as forcing a redraw of its display.

Latexmk has the ability to print a banner in gray diagonally across each page when making the postscript file. It can also, if needed, call an external program to do other postprocessing on the generated files.

Latexmk is highly configurable, both from the command line and in configuration files, so that it can accommodate a wide variety of user needs and system configurations. Default values are set according to the operating system, so *latexmk* often works without special configuration on MS-Windows, cygwin, Linux, OS-X, and other UNIX systems (notably Solaris).

A very annoying complication handled very reliably by *Latexmk*, is that LaTeX is a multiple pass system. On each run, LaTeX reads in information generated on a previous run, for things like cross referencing and indexing. In the simplest cases, a second run of LaTeX suffices, and often the log file contains a message about the need for another pass. However, there is a wide variety of add-on macro packages to LaTeX, with a variety of behaviors. The result is to break simple-minded determinations of how many runs are needed and of which programs. In its new version, *latexmk* has a highly general and efficient solution to these issues. The solution involves retaining between runs information on the source files, and a symptom is that *latexmk* generates an extra file (with extension .fdb_latexmk, by default) that contains the source file information.

LATEXMK OPTIONS AND ARGUMENTS

(All options can be introduced by single or double "-" characters, e.g., "latexmk -help" or "latexmk --help".)

file One or more files can be specified. If no files are specified, *latexmk* will, by default, run on all files in the current working directory with a ".tex" extension. This behavior can be changed: see the section concerning the *@default_files* variable.

If a file is specified without an extension, then the ".tex" extension is automatically added, just as LaTeX does. Thus, if you specify:

```
latexmk foo
```

then *latexmk* will operate on the file "foo.tex".

-bm <message>

A banner message to print diagonally across each page when converting the dvi file to postscript. The message must be a single argument on the command line so be careful with quoting spaces and such.

Note that if the **-bm** option is specified, the **-ps** option is assumed.

-bi <intensity>

How dark to print the banner message. A decimal number between 0 and 1. 0 is black and 1 is white. The default is 0.95, which is OK unless your toner cartridge is getting low.

-bs <scale>

A decimal number that specifies how large the banner message will be printed. Experimentation is necessary to get the right scale for your message, as a rule of thumb the scale should be about equal to 1100 divided by the number of characters in the message. The default is 220.0 which is just right for 5 character messages.

-commands

List the commands used by *latexmk* for processing files, and then exit.

-c Clean up (remove) all regenerateable files generated by *latex* and *bibtex* except dvi, postscript and pdf. In addition, files specified by the $clean_ext configuration variable are removed. But the file containing a database of source file information is not removed.

This cleanup is instead of a regular make. See the **-gg** option if you want to do a cleanup then a make.

-C Clean up (remove) all regenerateable files generated by *latex* and *bibtex* including aux, dep, dvi, postscript and pdf. In addition, those specified by the $clean_ext and $clean_full_ext configuration variables are removed. But the file containing a database of source file information is not removed.

This cleanup is instead of a regular make. See the **-gg** option if you want to do a cleanup than a make.

-CA Clean up (remove) absolutely all regenerateable files generated by *latex* and *bibtex* including aux, dep, dvi, postscript and pdf. In addition, those specified by the $clean_ext, $clean_full_ext, and @generated_exts configuration variables are removed, and the file containing a database of source file information.

This cleanup is instead of a regular make. It is the same as **-C -CF**. See the **-gg** option if you want to do a cleanup then a make.

-CF Remove the file containing a database of source file information, before doing the other actions requested.

-d Set draft mode. This prints the banner message "DRAFT" across your page when converting the dvi file to postscript. Size and intensity can be modified with the **-bs** and **-bi** options. The **-bm** option will override this option as this is really just a short way of specifying:

```
latexmk -bm DRAFT
```

Note that if the **-d** option is specified, the **-ps** option is assumed.

-dF Dvi file filtering. The argument to this option is a filter which will generate a filtered dvi file with the extension ".dviF". All extra processing (e.g. conversion to postscript, preview, printing) will then be performed on this filtered dvi file.

Example usage: To use dviselect to select only the even pages of the dvi file:

```
latexmk -dF ’dviselect even’ foo.tex
```

-diagnostics

Print detailed diagnostics during a run. This may help for debugging problems or to understand .*latexmk*’s behavior in difficult situations.

-dvi Generate dvi version of document.

-dvi- Turn off generation of dvi version of document. (This may get overridden, if some other file is made (a .ps file) that is generated from the dvi file, or if no generated file at all is requested.)

-f Force *latexmk* to continue document processing despite errors. Normally, when *latexmk* detects that LaTeX or another program has found an error which will not be resolved by further processing, no further processing is carried out.

-f- Turn off the forced processing-past-errors such as is set by the **-f** option. This could be used to override a setting in a configuration file.

-g Force *latexmk* to process document fully, even under situations where *latexmk* would normally decide that no changes in the source files have occured since the previous run. This option is useful, for example, if you change some options and wish to reprocess the files.

-g- Turn off **-g**.

-gg "Super go mode" or "clean make": clean out generated files as if **-CA** had been given, and then do a regular make.

-h, -help

Print help information.

-l Run in landscape mode, using the landscape mode for the previewers and the dvi to postscript converters. This option is not normally needed nowadays, since current previewers normally determine this information automatically.

-l- Turn off **-l**.

-new-viewer

When in continuous-preview mode, always start a new viewer to view the generated file. By default, *latexmk* will, in continuous-preview mode, test for a previously running previewer for the same file and not start a new one if a previous previewer is running. However, its test sometimes fails (notably if there is an already-running previewer that is viewing a file of the same name as the current file, but in a different directory). This option turns off this default behavior.

-new-viewer-

The inverse of the **-new-viewer** option. It puts *latexmk* in its normal behavior that in preview-continuous mode it checks for an already-running previewer.

-p Print out the document. By default on a UNIX or Linux system, this is done using lpr after generating the postscript file. But you can use the **-print=...** option to print the dvi or pdf files instead, and you can configure this in a start up file (by setting the *$print_type* variable).

However, the correct behavior for printing very much depends on your system's software. In particular, under MS-Windows you must have suitable program(s) available, and you must have configured the print commands used by *latexmk*. This can be non-trivial.

This option is incompatible with the **-pv** and **-pvc** options, so it turns them off.

-pdf Generate pdf version of document using pdflatex.

-pdfdvi

Generate pdf version of document from the dvi file, by default using dvipdf.

-pdfps Generate pdf version of document from the ps file, by default using ps2pdf.

-pdf- Turn off generation of pdf version of document. (This can be used to override a setting in a configuration file. It may get overridden if some other option requires the generation of a pdf file.)

-print=dvi, -print=ps, -print=pdf

Define which kind of file is printed. This option also ensures that the requisite file is made, and turns on printing.

-ps Generate postscript version of document.

-ps- Turn off generation of postscript version of document. This can be used to override a setting in a configuration file. (It may get overridden by some other option that requires a postscript file, for example a request for printing.)

-pF Postscript file filtering. The argument to this option is a filter which will generate a filtered postscript file with the extension ".psF". All extra processing (e.g. preview, printing) will then be performed on this filtered postscript file.

Example usage: Use psnup to print two pages on the one page:

```
latexmk -ps -pF 'psnup -2' foo.tex
```

or

```
latexmk -ps -pF "psnup -2" foo.tex
```

Whether to use single or double quotes round the "psnup -2" will depend on your command interpreter, in particular on the operating system.

-pv Run file previewer. If the **-view** option is used, this will select the kind of file to be previewed (dvi, ps or pdf). Otherwise the viewer views the "highest" kind of file selected, by the **-dvi**, **-ps**, **-pdf**, **-pdfps** options, in the order dvi, ps, pdf (low to high). If no file type has been selected, the dvi previewer will be used. This option is incompatible with the **-p** and **-pvc** options, so it turns them off.

-pv- Turn off **-pv**.

-pvc Run a file previewer and continually update the .dvi, .ps, and/or .pdf files whenever changes are made to source files (see the Description above). Which of these files is generated and which is viewed is governed by the other options, and is the same as for the **-pv** option. This option also turns on the **-f** option, since it is normally desirable in preview-continuous-mode to continue working even if errors are found. The preview-continuous option **-pvc** can only work with one file. So in this case you will normally only specify one filename on the command line. It is also incompatible with the **-p** and **-pv** options, so it turns these options off

With a good previewer the display will be automatically updated. (Under *some but not all* versions of UNIX/Linux "gv -watch" does this for postscript files; this can be set by a configuration variable. This would also work for pdf files except for an apparent bug in gv that causes an error when the newly updated pdf file is read.) Many other previewers will need a manual update.

Important note: the acroread program on MS-Windows locks the pdf file, and prevents new versions being written, so it is a bad idea to use acroread to view pdf files in preview-continuous mode. It is better to use a dvi or ps viewer, as set by one of the **-view=dvi** and **-view=ps** options.

There are some other methods for arranging an update, notably useful for many versions of xdvi and xpdf. These are best set in *latexmk*'s configuration; see below.

Note that if *latexmk* dies or is stopped by the user, the "forked" previewer will continue to run. Successive invocations with the **-pvc** option will not fork new previewers, but *latexmk* will normally use the existing previewer. (At least this will happen when *latexmk* is running under an operating system where it knows how to determine whether an existing previewer is running.)

-pvc- Turn off **-pvc**.

-quiet Same as -silent

-r <rcfile>

Read the specified initialization file ("RC file") before processing.

Be careful about the ordering: (1) Standard initialization files -- see the section below on "Initialization (RC) files" -- are read first. (2) Then the options on the command line are acted on in the order they are given. Therefore if an initialization file is specified by the **-r** option, it is read during this second step. Thus an initialization file specified with the **-r** option can override both the standard initialization files and *previously* specified options. But all of these can be overridden by *later* options. See below for more details about initialization (RC) files.

-silent Run commands silently, i.e., with options that reduce the amount of diagnostics generated. For example, with the default settings for commands under UNIX, the command "latex -interaction=batchmode" is used for latex.

Also reduce the number of informational messages that *latexmk* generates.

-v, -version

Print version number of *latexmk*.

-verbose

Opposite of **-silent**. This is the default setting.

-view=default, -view=dvi, -view=ps, -view=pdf

Set the kind of file used when previewing is requested (e.g., by the **-pv** or **-pvc** switches). The default is to view the "highest" kind of requested file (in the order dvi, ps, pdf).

The preview-continuous option **-pvc** can only work with one file. So in this case you will normally only specify one filename on the command line.

Options **-p**, **-pv** and **-pvc** are mutually exclusive. So each of these options turns the others off.

EXAMPLES

% **latexmk thesis**	*# run latex enough times to resolve cross-references*
% **latexmk -pvc -ps thesis**	*# run latex enough times to resolve cross-references, make a postscript file, start a previewer. Then watch for changes in the source file thesis.tex and any files it*

uses. After any changes rerun latex the appropriate number of times and remake the postscript file. If latex encounters an error, latexmk will keep running.

% **latexmk -c**	*# remove .aux, .log, .bbl, .blg, .dep, .dvi, .pdf, .ps & .bbl files*

INITIALIZATION (RC) FILES

There are four initialization files ("RC files") that *latexmk* can read at startup:

1) The system RC file, if it exists.
On a UNIX system, *latexmk* searches for following places for its
system RC file, in the following order, and reads the first it finds:
"/opt/local/share/latexmk/LatexMk",
"/usr/local/share/latexmk/LatexMk",
"/usr/local/lib/latexmk/LatexMk".
On a MS-WINDOWS system it looks for "C:\latexmk\LatexMk".

2) The user’s RC file in "$HOME/.latexmkrc", where $HOME is the value of the environment variable HOME. On UNIX and clones (including LINUX), this variable is set by the system; on MS-Windows, the user may choose to set it.

3) The RC file in the current working directory called "latexmkrc".

4) Any RC file(s) specified on the command line with the **-r** option.

Each RC file is a sequence of Perl commands. Usually it will be just a sequence of assignment statements that override the built-in settings of *Latexmk*. Comment lines are introduced by the "#" character.

Note that command line options are obeyed in the order in which they are written; thus any RC file specified on the command line with the **-r** option can override previous options but can be itself overridden by later options on the command line.

RC VARIABLES IN INITIALIZATION FILES

Many of the available variables that can be set are shown in the next section. Syntax for the statements in an initialization file is of the form:

```
$bibtex = ’bibtex’;
```

for the setting of a string variable,

```
$preview_mode = 1;
```

for the setting of a numeric variable, and

```
@default_files = (’paper’, ’paper1’);
```

for the setting of an array of strings.

Some of the variables set the names of the commands that *latexmk* uses. Here are some tricks to note for these:

"Detaching" a command: If a command is to be run detached this is indicated by preceding it with "start", as in

```
$dvi_previewer = ’start xdvi’;
```

This will be translated to whatever is appropriate for your operating system. (Note: in some circumstances, *latex* will always run a command detached. This is the case for a previewer in preview continuous mode, since otherwise previewing continuously makes no sense.)

Command names containing spaces: Under MS-Windows it is common that the name of a command includes spaces, since software is often installed in a subdirectory of "C:Program Files". Such command names should be enclosed in double quotes, as in

```
$lpr_pdf = '"c:/Program Files/Ghostgum/gsview/gsview32.exe" /p';
```

Using MS-Windows file associations: A useful trick under modern versions of MS-Windows (e.g., WinXP) is to use just the command

```
$dvi_previewer = 'start';
```

Under recent versions of MS-Windows, this will cause to be run whatever program the system has associated with dvi files. (The same applies for a postscript viewer and a pdf viewer.)

Not using a certain command: If a command is not to be run, the command name NONE is used, as in

```
$lpr  = 'NONE lpr';
```

This means that an appropriate command has not been configured. The string after the 'NONE' is effectively a comment.

Options to commands: Setting the name of a command can be used not only for changing the name of the command called, but also to add options to command. Suppose you want *latexmk* to use latex with source specials enabled. Then you might use the following line in an initialization file:

```
$latex = 'latex --src-specials';
```

Advanced tricks: Normally *latexmk* assumes certain behavior for commands and in particular it assumes certain kinds and ordering of command line arguments. Sometimes this assumption is wrong. For example you might want to use Distiller to convert postscript files to pdf files. You cannot simply change the name of the ps2pdf conversion program, as in

```
$ps2pdf = 'distiller';  ######### WRONG
```

because the command line arguments will be wrong. In such a situation, your best bet is to write a batch file (under MS-Windows) or a script (under UNIX) that will do the conversion. Then you set

```
$ps2pdf = 'special_script';
```

Your script will be invoked by *latexmk* in its usual way as "Special_script file.ps file.pdf". Your script calls Distiller with Distiller's correct arguments.

LIST OF RC VARIABLES IN INITIALIZATION FILES

Default values are indicated in brackets.

$banner [0]

If nonzero, the banner message is printed across each page when converting the dvi file to postscript. Without modifying $banner_message, this is equivalent to specifying the **-d** option.

Note that if **$banner** is nonzero, the **$postscript_mode** is assumed and the postscript file is always generated, even if it is newer than the dvi file.

$banner_intensity [0.95]

Equivalent to the **-bi** option, this is a decimal number between 0 and 1 that specifies how dark to print the banner message. 0 is black, 1 is white. The default is just right if your toner cartridge isn't running too low.

$banner_message ["DRAFT"]

The banner message to print across each page when converting the dvi file to postscript. This is equivalent to the **-bm** option.

$banner_scale [220.0]

A decimal number that specifies how large the banner message will be printed. Experimentation is necessary to get the right scale for your message, as a rule of thumb the scale should be about equal to 1100 divided by the number of characters in the message. The Default is just right for 5 character messages. This is equivalent to the **-bs** option.

@BIBINPUTS

This is an array variable that specifies directories where *latexmk* should look for .bib files. By default it is set from the BIBINPUTS environment variable of the operating system. If that environment variable is not set, a single element list consisting of the current directory is set. The format of the directory names depends on your operating system, of course. Examples are:

```
@BIBINPUTS = ( ’.’, ’C:\bibfiles’ );
@BIBINPUTS = ( ’.’, ’\\server\bibfiles’ );
@BIBINPUTS = ( ’.’, ’C:/bibfiles’ );
@BIBINPUTS = ( ’.’, ’//server/bibfiles’ );
@BIBINPUTS = ( ’.’, ’/usr/local/texmf/bibtex/bib’ );
```

Note that under MS Windows, either a forward slash ’/’ or a backward slash ’´ can be used to separate pathname components, so the first two and the second two examples are equivalent. Each backward slash should be doubled to avoid running afoul of Perl’s rules for writing strings.

This variable is likely to become obsolete in a future version of *latexmk* which uses a better method of searching for files.

$bibtex ["bibtex"]

The BibTeX processing program.

$bibtex_silent_switch ["-terse"]

Switch(es) for the BibTeX processing program when silent mode is on.

$cleanup_mode [0]

If nonzero, specifies cleanup mode: 1 for full cleanup, 2 for cleanup except for dvi, ps and pdf files, 3 for cleanup except for dep and aux files. (There is also extra cleaning as specified by the $clean_ext, $clean_full_ext and @generated_exts variables.)

This variable is equivalent to specifying one of the **-c**, **-c1**, or **-C** options. But there should be no need to set this variable from an RC file.

$clean_ext [""]

Extra extensions of files for *latexmk* to remove when any of the clean-up options (**-c**, **-c1**, or **-C**) is selected.

$clean_full_ext [""]

Extra extensions of files for *latexmk* to remove when the **-C** option is selected, i.e., extensions of files to remove when the .dvi, etc files are to be cleaned-up.

@cus_dep_list [()]

Custom dependency list -- see section on "Custom Dependencies".

@default_files [(’*.tex’)]

Default list of files to be processed.

Normally, if no filenames are specified on the command line, *latexmk* processes all tex files specified in the @default_files variable, which by default is set to all tex files (’*.tex’) in the current directory. This is a convenience: just run *latexmk* and it will process an appropriate set of files. But sometimes you want only some of these files to be processed. In this case you set the (PERL array variable) *@default_files* in an initialization file (e.g., the file "latexmkrc" in the current directory). Then if no files are specified on the command line then the files you specify by setting

@default_files are processed.

Three examples:

```
@default_files = ('paper_current');

@default_files = ('paper1', 'paper2.tex');

@default_files = ('*.tex', '*.dtx');
```

Note that more than file may be given, and that the default extension is '.tex'. Wild cards are allowed. The parentheses are because *@default_files* is an array variable, i.e., a sequence of filename specifications is possible.

$dvi_filter [empty]

The dvi file filter to be run on the newly produced dvi file before other processing. Equivalent to specifying the **-dF** option.

$dvi_previewer ["start xdvi" under UNIX]

The command to invoke a dvi-previewer. [Default is "start" under MS-WINDOWS; under more recent versions of Windows, this will cause to be run whatever command the system has associated with .dvi files.]

$dvi_previewer_landscape ["start xdvi"]

The command to invoke a dvi-previewer in landscape mode. [Default is "start" under MS-WINDOWS; under more recent versions of Windows, this will cause to be run whatever command the system has associated with .dvi files.]

$dvipdf ["dvipdf"]

Command to convert dvi to pdf file.

WARNING1: The default dvipdf script generates pdf files with bitmapped fonts, which don't look good when viewed by acroread. The script should be modified to give dvips the options "-P pdf" to ensure that type 1 fonts are used in the pdf file.

WARNING 2: If you want to use one of the programs dvipdfm or dvipdfmx instead dvipdf, you will find their use of command line arguments is incompatible with that of dvipdf. You should use these programs indirectly, invoking them through a script or batch file to convert the command line. See the extra_scripts directory of the latexmk distribution, where the necessary scripts dvipdfm_call, etc are to be found, together with instructions for their use in the file README1.

$dvips ["dvips"]

The program to used as a filter to convert a .dvi file to a .ps file. If pdf is going to be generated from pdf, then the value of the $dvips_pdf_switch -- see below -- will be appended.

$dvips_landscape ["dvips -tlandscape"]

The program to used as a filter to convert a .dvi file to a .ps file in landscape mode.

$dvips_pdf_switch ["-P pdf"]

Switch(es) for dvips program when pdf file is to be generated from ps file.

$dvips_silent_switch ["-q"]

Switch(es) for dvips program when silent mode is on.

$dvi_update_command [""]

When the dvi previewer is set to be updated by running a command, this is the command that is run. See the information for the variable *$dvi_update_method*.

$dvi_update_method [2 under UNIX, 1 under MS-Windows]

How the dvi viewer updates its display when the dvi file has changed.
0 => update is automatic,
1=> manual update by user, which may only mean a mouse click on the viewer's window or may mean a more serious action.
2 => Send the signal, whose number is in the variable $dvi_update_signal. The default value under UNIX is suitable for xdvi.
3 => Viewer cannot do an update, because it locks the file. (As with acroread under MS-Windows.)
4 => run a command to do the update. The command is specified by the variable $dvi_update_command.

$dvi_update_signal [Under UNIX: SIGUSR1, which is a system-dependent value]

The number of the signal that is sent to the dvi viewer when it is updated by sending a signal -- see $dvi_update_method. The default value is the one appropriate for xdvi on a UNIX system.

$fdb_ext ["fdb_latex"]

The extension of the file which *latexmk* generates to contain a database of information on source files. You will not normally need to change this.

$force_mode [0]

If nonzero, continue processing past minor *latex* errors including unrecognized cross references. Equivalent to specifying the **-f** option. Note that specifying the **-pvc** sets $force_mode to 1.

@generated_exts [('ind', 'lof', 'lot', 'out', 'toc', $fdb_ext)]

This contains a list of extensions for files that are generated (directly or indirectly) during a LaTeX run and that are read in by LaTeX in later runs. In some previous versions of *latexmk*, this list of extensions was used to avoid making mistakes on understanding the true source files for LaTeX. New versions (3.20 onwards) no longer make this use of this list of extensions.

Currently the primary use of the list is in determining which files are to be deleted when doing a cleanup; the extensions are added to those specified by $clean_ext.

The extensions "aux" and "bbl" are always excluded from the dependents, because they get special treatment, so they do not need to be in this list.

A convenient way to add an extra extension to the list, without losing the already defined ones is to use a push command in the line in an RC file. E.g.,

```
push @generated_exts, 'end';
```

adds the extension 'end' to the list of predefined generated extensions.

$go_mode [0]

If nonzero, process files regardless of timestamps. Equivalent to the **-g** option.

$index_mode [0 and then as determined from the results of a run]

If nonzero, run *makeindex* to produce index of document. Normally you should not need to set this variable in an RC file, since latexmk determines automatically if *makeindex* needs to be run.

$landscape_mode [0]

If nonzero, run in landscape mode, using the landscape mode previewers and dvi to postscript converters. Equivalent to the **-l** option. Normally not needed with current previewers.

$latex ["latex"]

The LaTeX processing program. Note that as with other programs, you can use this variable not just to change the name of the program used, but also specify options to the program. E.g.,

```
$latex = 'latex --src-specials';
```

$latex_silent_switch ["-interaction=batchmode"]

Switch(es) for the LaTeX processing program when silent mode is on. Under MS-Windows, the default value is changed to "-interaction=batchmode -c-style-errors", as used by MikTeX and fpTeX.

$lpr ["lpr"]

[Default is "NONE lpr" under MS-WINDOWS.] The command to print postscript files.

Under MS-Windows (unlike UNIX/LINUX), there is no standard program for printing files. But there are ways you can do it. For example, if you have gsview installed, you could use it with the option '/p':

```
$lpr = '"c:/Program Files/Ghostgum/gsview/gsview32.exe" /p';
```

If gsview is installed in a different directory, you will need to make the appropriate change. Note the combination of single and double quotes around the name. The single quotes specify that this is a string to be assigned to the configuration variable $lpr. The double quotes are part of the string passed to the operating system to get the command obeyed; this is necessary because one part of the command name ('Program Files') contains a space which would otherwise be misinterpreted.

$lpr_dvi ["NONE lpr_dvi"]

The printing program to print dvi files.

$lpr_pdf ["NONE lpr_pdf"]

The printing program to print pdf files.

Under MS-Windows you could set this to use gsview, if it is installed, e.g.,

```
$lpr = '"c:/Program Files/Ghostgum/gsview/gsview32.exe" /p';
```

If gsview is installed in a different directory, you will need to make the appropriate change. Note the double quotes around the name: this is necessary because one part of the command name ('Program Files') contains a space which would otherwise be misinterpreted.

$makeindex ["makeindex"]

The index processing program.

$new_viewer_always [0]

This variable applies to *latexmk* **only** in continuous-preview mode. If $new_viewer_always is 0, *latexmk* will check for a previously running previewer on the same file, and if one is running will not start a new one. If $new_viewer_always is non-zero, this check will be skipped, and *latexmk* will behave as if no viewer is running.

$pdf_mode [0]

If zero, do NOT generate a pdf version of the document. If equal to 1, generate a pdf version of the document using pdflatex. If equal to 2, generate a pdf version of the document from the ps file, by using the command specified by the $ps2pdf variable. If equal to 3, generate a pdf version of the document from the dvi file, by using the command specified by the $dvipdf variable. Equivalent to the **-pdf-**, **-pdf**, **-pdfdvi**, **-pdfps** options.

$pdflatex ["pdflatex"]

The LaTeX processing program in the version that makes a pdf file instead of a dvi file.

$pdflatex_silent_switch ["-interaction=batchmode"]

Switch(es) for the LaTeX processing program when silent mode is on. Under MS-Windows, the default value is changed to "-interaction=batchmode -c-style-errors", as used by MikTeX and fpTeX.

$pdf_previewer ["start acroread"]

The command to invoke a pdf-previewer. [Default is changed to "start" on MS-WINDOWS; under more recent versions of Windows, this will cause to be run whatever command the system has associated with .pdf files.]

Potential problem under MS-Windows: if acroread is used as the pdf previewer, and it is actually viewing a pdf file, the pdf file cannot be updated. Thus makes acroread a bad choice of previewer if you use *latexmk*'s previous-continuous mode (option **-pvc**) under MS-windows. This problem does not occur if ghostview, gv or gsview is used to view pdf files.

$pdf_update_command [""]

When the pdf previewer is set to be updated by running a command, this is the command that is run. See the information for the variable *$pdf_update_method*.

$pdf_update_method [1 under UNIX, 3 under MS-Windows]

How the pdf viewer updates its display when the pdf file has changed. See $dvi_update_method for the codes, with the change that for the value 4, to run a command to do the update, the command is specified by the variable $pdf_update_command.

Note that acroread under MS-Windows (but not UNIX) locks the pdf file, so the default value is then 3.

$pdf_update_signal [Under UNIX: SIGHUP, which is a system-dependent value]

The number of the signal that is sent to the pdf viewer when it is updated by sending a signal -- see $pdf_update_method. The default value is the one appropriate for gv on a UNIX system.

$pid_position = [1 under UNIX, -1 under MS-Windows]

Command used to get all the processes currently run by the user. The -pvc option uses the command specified by the variable $pscmd to determine if there is an already running previewer, and to find the process ID (needed if *latexmk* needs to signal the previewer about file changes). The variable $pid_position is used to specify which word in lines of the output from $pscmd corresponds to the process ID. The first word in the line is numbered 0. The default value of 1 (2nd word in line) is correct for Solaris 2.6 and Linux. Setting the variable to -1 is used to indicate that $pscmd is not to be used.

$postscript_mode [0]

If nonzero, generate a postscript version of the document. Equivalent to the **-ps** option.

$preview_continuous_mode [0]

If nonzero, run a previewer to view the document, and continue running *latexmk* to keep .dvi up-to-date. Equivalent to the **-pvc** option. Which previewer is run depends on the other settings, see the command line options **-view=**, and the variable *$view*.

$preview_mode [0]

If nonzero, run a previewer to preview the document. Equivalent to the **-pv** option. Which previewer is run depends on the other settings, see the command line options **-view=**, and the variable *$view*.

$printout_mode [0]

If nonzero, print the document using *lpr*. Equivalent to the **-p** option. This is recommended **not** to be set from an RC file, otherwise you could waste lots of paper.

$print_type = ["ps"]

Type of file to printout: possibilities are "dvi", "none", "pdf", or "ps".

$pscmd

[On UNIX, the default is "ps -f -u $ENV{USER}", with changes for Linux and OS-X. On MS-WINDOWS the default in "NONE pscmd".] Command used to get all the processes currently run by the user. This is used by the -pvc option to determine if there is an already running previewer. The command line options for this command under the different flavors of UNIX are quite variable. The command given above is suitable for Solaris 2.6 and above, and *latexmk* corrects it for Linux and OSX.

NOTE: The variable *$pid_position* must also be set; see its description.

$ps2pdf ["ps2pdf"]
Command to convert ps to pdf file.

$ps_filter [empty]
The postscript file filter to be run on the newly produced postscript file before other processing. Equivalent to specifying the **-pF** option.

$ps_previewer ["start gv -watch"]
The command to invoke a ps-previewer. [Default is "start" on MS-WINDOWS; under more recent versions of Windows, this will cause to be run whatever command the system has associated with .ps files.]

Note that gv with the -watch option updates its display whenever the postscript file changes, whereas ghostview does not.

$ps_previewer_landscape ["start gv -swap -watch"]
The command to invoke a ps-previewer in landscape mode. [Default is "start" on MS-WINDOWS; under more recent versions of Windows, this will cause to be run whatever command the system has associated with .ps files.]

$ps_update_command [""]
When the postscript previewer is set to be updated by running a command, this is the command that is run. See the information for the variable *$ps_update_method*.

$ps_update_method [0 under UNIX, 1 under MS-Windows]
How the postscript viewer updates its display when the ps file has changed. See $dvi_update_method for the codes, with the change that for the value 4, to run a command to do the update, the command is specified by the variable $ps_update_command.

$ps_update_signal [Under UNIX: SIGHUP, which is a system-dependent value]
The number of the signal that is sent to the pdf viewer when it is updated by sending a signal -- see $ps_update_method. The default value is the one appropriate for gv on a UNIX system.

$sleep_time [2]
The time to sleep (in seconds) between checking for source file changes when running the **-pvc** option.

$texfile_search [""]
This is an obsolete variable, replaced by the *@default_files* variable.

For backward compatibility, if you choose to set *$texfile_search*, it is a string of space-separated filenames, and then *latexmk* replaces *@default_files* with the filenames in *$texfile_search* to which is added '*.tex'.

$tmpdir [See below for default]
Directory to store temporary files that *latexmk* may generate while running.

The default under MSWindows (including cygwin), is to set *$tmpdir* to the value of the first of whichever of the system environment variables TMPDIR or TEMP exists, otherwise to the current directory. Under other operating systems (expected to be UNIX/Linux, including OS-X), the default is the value of the system environment variable TMPDIR if it exists, otherwise to "/tmp".

$view ["default"]
Which kind of file is to be previewed if a previewer is used. The possible values are 'default', 'dvi', 'ps', 'pdf'. The value of "default" means that the "highest" of the kinds of file generated is to be used (among dvi, ps and pdf).

CUSTOM DEPENDENCIES

In any RC file a set of custom dependencies can be set up to convert a file with one extension to a file with another. An example use of this would be to allow *latexmk* to convert a *.fig* file to *.eps* to be included in the *.tex* file. A table of custom dependencies are set up by using the **@cus_dep_list** array. Each string in the array has four arguments, separated by a space:

from extension:

The extension of the file we are converting from (e.g. "fig").

to extension:

The extension of the file we are converting to (e.g. "eps").

must: If non-zero, the file we are converting from **must** exist, if it doesn't exist *latexmk* will give an error message and exit unless the **-f** option is specified. If *must* is zero and the file we are converting from doesn't exist, then no action is taken.

function:

The name of the subroutine that *latexmk* should call to perform the file conversion. The first argument to the subroutine is the base name of the file to be converted without any extension. The subroutines are declared in the syntax of *perl*. The function should return 0 if it was successful and a nonzero number if it failed.

Example in an RC file to convert a *.fig* file to a *.eps* file:

```
@cus_dep_list = (@cus_dep_list, "fig eps 0 fig2eps");

sub fig2eps {
  system("fig2dev -Lps $_[0].fig $_[0].eps"); }
```

The subroutine *fig2eps* will only be called if the *.fig* file was modified more recently then the *.eps* file, or the *.eps* file does not exist.

If the return value of the subroutine is non-zero, then *latexmk* will assume an error occurred during the execution of the subroutine.

SEE ALSO

latex(1), bibtex(1).

BUGS

Search for .bib files is not correct if they are not in the current directory; the problem is that the log file generated by bibtex does not give the full path to the .bib files. The easiest fix at the moment is to set the BIBINPUTS environment variable to include explicitly the path containing your .bib files. Or you can set the @BIBINPUTS variable explicitly in one of latexmk's startup files.

If .bbl file exists and is used, but the .bib file does not exist, then latexmk should not try to run bibtex, but it does.

Sometimes a viewer (gv) tries to read an updated .ps or .pdf file after its creation is started but before the file is complete. Work around: manually refresh (or reopen) display.

(The following isn't really a bug, but concerns features of previewers.) Preview continuous mode only works perfectly with certain previewers: Xdvi on UNIX/LINUX works for dvi files. Gv on UNIX/LINUX works for both postscript and pdf. Ghostview on UNIX/LINUX needs a manual update (reopen); it views postscript and pdf. Gsview under MS-Windows works for both postscript and pdf, but only reads the updated file when its screen is refreshed. Acroread under UNIX/LINUX views pdf, but the file needs to be closed and reopened to view an updated version. Under MS-Windows, acroread locks its input file and so the pdf file cannot be updated. (Remedy: configure *latexmk* use gsview instead.)

THANKS TO

Authors of previous versions. Many users with their feedback, and especially David Coppit (username david at node coppit.org) who made many useful suggestions that contributed to version 3. (Please note that the e-mail addresses are not written in their standard form to avoid being harvested by worms and viruses.)

AUTHOR

Current version, with substantial modifications, enhancements and bug fixes by John Collins (username collins at node phys.psu.edu). (Version 3.09).

It can be obtained from CTAN: <http://www.tug.org/tex-archive/support/latexmk/>, and from the author's website <http://www.phys.psu.edu/~collins/software/latexmk/>.
Modifications and enhancements by Evan McLean (Version 2.0)
Original script called "go" by David J. Musliner (RCS Version 3.2)