

NAME

tlmgr – the TeX Live Manager

SYNOPSIS

tlmgr [*option*]... *action* [*option*]... [*operand*]...

DESCRIPTION

tlmgr manages an existing TeX Live installation, both packages and configuration options. It performs many of the same actions as **texconfig**(1), and more besides. (**texconfig** continues to be included and supported, but **tlmgr** is now preferred.)

The most up-to-date version of this documentation is available at <<http://tug.org/texlive/tlmgr.html>>, along with procedures for updating tlmgr itself, disaster recovery when it's broken, and test versions.

OPTIONS

The following options to **tlmgr** have to be given *before* you specify the main action to perform.

--location *location*

Specifies the location from which packages should be installed or updated, overriding the location found in the installation's TeX Live Package Database (TLPDB).

--gui [*action*]

You can give this option together with an action to be brought directly into the respective screen of the GUI. For example, running

```
tlmgr --gui update
```

starts you directly at the update screen.

--gui-lang *llcode*

Normally the GUI tries to deduce your language from the environment (on Windows via the registry, on Unix via LC_MESSAGES). If that fails you can select a different language by giving this option a two-letter language code.

--package-logfile *file*

tlmgr logs all package actions (install, remove, update, failed updates, failed restores) to a separate log file, by default `TEXMFSYSVAR/web2c/tlmgr.log`. This option allows you to select a different file for that. This is separate from normal logging; for that, see the option `-v` below, and `TeXLive::TLUtils`.

--pause

This option make **tlmgr** wait for user input before exiting.

The standard options for TeX Live programs are also accepted: `--help/-h/-?`, `--version`, `-q` (no informational messages), `-v` (debugging messages, can be repeated). For the details about these, see the `TeXLive::TLUtils` documentation.

The `--version` option shows version information about the TeX Live release as well as the **tlmgr** script itself.

ACTIONS

help

Gives this help information (same as `--help`).

version

Gives version information (same as `--version`).

gui Start the graphical user interface.

install [*option*]... *pkg*...

Install each *pkg* given on the command line. By default this installs all packages on which the given *pkgs* are dependent, also. Options:

--reinstall

Reinstall a package (including dependencies for collections) even if it seems to be already installed (i.e. is present in the TLPDB). This is useful to recover from accidental removal of files in the hierarchy.

When re-installing, only dependencies on normal packages are followed (not those of category Scheme or Collection).

--no-depends

Do not install dependencies. (By default, installing a package ensures that all dependencies of this package are fulfilled.)

--no-depends-at-all

When you install a package which ships binary files the respective binary package will also be installed. (So for package `bin-foobar` also the package `bin-foobar.i386-linux` will be installed on an `i386-linux` system.) This switch supresses this behaviour, and also implies `--no-depends`. Don't use it unless you are sure of what you are doing.

--dry-run

Nothing is actually installed; instead, the actions to be performed are written to the terminal.

--force If updates to the `tlmgr` itself (and the underlying infrastructure) are present `tlmgr` will bail out and not perform the installation unless this option is given.

update [*option*] [*pkg*]...

Updates the packages given as arguments to the latest version available at the installation source. Either `--all` or at least one *pkg* name must be specified. Options:

--all Update all installed packages. For packages contained in an installed collection, also install new packages and remove deleted packages in those collections. Packages not contained in an installed collection are not considered for addition or removal.

More precisely, if both the server and the local installation have a collection *C*, and *C* on the server contains a package *A* that is not present locally, *A* will be added. Conversely, if the local collection *C* contains a package *B*, but *B* is no longer in the server's *C*, then *B* will be deleted.

--list Concisely list the packages which would be updated, newly installed, or removed, without actually changing anything.

--dry-run

Nothing is actually installed; instead, the actions to be performed are written to the terminal. This is a more detailed report than `--list`.

--force If updates to the `tlmgr` itself (and the underlying infrastructure) are present `tlmgr` will bail out and not perform the update unless this option is given (or **--list** is given).

--no-remove

Under normal circumstances `tlmgr` tries to remove packages which have disappeared on the server when called with **--all**, as described above. This prevents any such removals.

--backup and **--backupdir** *directory*

These two options control the creation of backups of packages before an update is started; that is, the backup is of the package as it's installed. If neither of the two are given, no backup package will be saved. If **--backupdir** is given and specifies a writable directory then a backup will be made in that location. If only **--backup** is given, then a backup will be made to the directory previously set via the `option` action (see below). If both are given then a backup will be made to the specified *directory*.

You can set options via the `option` action to automatically create backups for all packages, and/or keep only a certain number of backups. Please see the `option` action for details.

`tlmgr` always makes a temporary backup when updating packages, in case of download or other failure during an update. The purpose of this **--backup** option is to allow you to save a persistent backup in case the actual *content* of the update causes problems, e.g., introduces an incompatibility.

The `restore` action explains how to restore from a backup.

--no-depends

If you call for updating a package normally all depending packages will also be checked for updates and updated if necessary. This switch suppresses this behaviour.

--no-depends-at-all

By default, when you update a package which ships binary files the respective binary package will also be updated. That is, for an update of a package `foo`, the package `foobar.i386-linux` will also be updated if the `i386-linux` platform is installed. This option suppresses this behaviour, and implies **--no-depends**. Don't use it unless you are sure of what you are doing.

If the package on the server is older than the package already installed (e.g., if the selected mirror is out of date), `tlmgr` does not downgrade. Also, packages for uninstalled platforms are not installed.

backup [**--clean**[=*N*]] [**--backupdir** *dir*] [**--all**] [*pkg*]...

If the **--clean** option is not specified, this action makes a backup of the given packages, or all packages given **--all**. These backups are saved to the value of **--backupdir** option if that is an existing and writable directory; if **--backupdir** is not given, the `backupdir` option setting in the TLPDB is used, if present. If both are missing, no backups are made.

If the **--clean** option is specified, backups are cleaned (pruned) instead of made. If the value for the **--clean** switch is not given, the value of the **autobackup** option from is used. If both are missing, an error is issued. For the details of backup cleaning, see the `option`

action below.

Options:

--backupdir *directory*

The *directory* argument is required and must specify an existing directory where backups are to be placed.

--all Make a backup of all packages in the TeX Live installation. This will take quite a lot of space and time.

--clean[=*N*]

Instead of making backups, prune the backup directory of old backups.

--dry-run

Nothing is actually backed up or removed; instead, the actions to be performed are written to the terminal.

restore **--backupdir** *dir* [*pkg* [*rev*]]

Restore a package from a previously-made backup.

If neither *pkg* nor *rev* are given, list the available backup revisions for all packages.

With *pkg* given but no *rev*, list all available backup revisions of *pkg*.

With both *pkg* and *rev*, tries to restore the package from the specified backup.

Options:

--dry-run

Nothing is actually restored; instead, the actions to be performed are written to the terminal.

--backupdir *directory*

Specify the directory where the backups are to be found.

remove [*option*]... *pkg*...

Remove each *pkg* specified. Removing a collection removes all package dependencies, but not collection dependencies, in that collection (unless **--no-depends** is specified). However, when removing a package, dependencies are never removed. Options:

--no-depends

Do not remove dependent packages.

--no-depends-at-all

See above under **action** (and beware).

--force By default, removal of a package or collection that is a dependency of another collection or scheme is not allowed. With this option, the package will be removed unconditionally. Use with care.

--dry-run

Nothing is actually removed; instead, the actions to be performed are written to the terminal.

option [*show*]

option *key* [*value*]

The first form shows the global TeX Live settings currently saved in the TLPDB.

In the second form, if *value* is not given, the setting for *key* is displayed. If *value* is present, *key* is set to *value*.

Possible values for *key* are:

`location` (default installation location),
`formats` (create formats at installation time),
`docfiles` (install documentation files),
`srcfiles` (install source files),
`autobackup` (number of backups to be kept),
`backupdir` (default directory for backups).

Perhaps the most common use of `option` is if you originally installed from DVD, and want to permanently change the installation to get further updates from the Internet. To do this, you can run

```
tlmgr option location http://mirror.ctan.org/systems/texlive/tlnet/YYYY
```

where YYYY is the TeX Live release year.

The two options `autobackup` and `backupdir` determine defaults for the `update`, `backup` and `restore` actions. These three actions need a directory to write to or read from the backups. If `--backupdir` is not specified on the command line, the `backupdir` option value is used (if set).

The `autobackup` option (de)activates the automatic generation of backups. Its value is an integer. If `autobackup` is `-1`, backups will not be removed by default. If `autobackup` is 0 or positive, it specifies the number of backups to be kept. Thus, backups are disabled with 0 as the values.

In the `--clean` mode of the `backup` action this option also specifies the number to be kept.

paper a4**paper letter**

[`xdvi`|`dvips`|`pdftex`|`dvipdfm`|`dvipdfmx`|`context`] paper [`help`|*papersize*|`--list`]

Configures the system wide paper settings, either for all programs at once (first two forms), or just for the specified program (third form). The third form with `--list` outputs all known papersizes for the specified program.

arch list|add *arch*...

`arch list` lists the TeX Live names of all the architectures (`i386-linux`, ...) available at the default install location.

`arch add arch` adds the executables for each given architecture *arch* to the installation.

Options:

--dry-run

Nothing is actually installed; instead, the actions to be performed are written to the terminal.

search [*option*]... *what*

By default searches the names, short and long descriptions of all locally installed packages for the given argument (interpreted as regexp). Options:

--file List all filenames containing *what*.

--global

Search the TeX Live Database of the installation medium, instead of the local installation.

show [--list] *pkg*...

Shows information about *pkg*: the name, category, installation status, short and long description. Searches in the remote installation source for the package if it is not locally installed.

If the option **--list** is given the list of contained files is also shown, including those for architecture specific dependencies.

list [collections|schemes|*pkg*...]

With no argument, lists all packages available at the default install location, prefixing those already installed with *i*.

With the single word `collections` or `schemes` as the argument, lists the request type.

With anything else as arguments, operates as the `show` action.

check [*option*]... [files|collections|all]

Executes one (or all) check(s) on the consistency of the installation.

files Checks that all files listed in the TeX Live Database (`texlive.tlpdb`) are actually present, and lists those missing.

collections

Lists those packages which occur as dependencies in an installed collections, but are themselves not installed.

Available options are:

--use-svn

Use `svn` status instead of listing the files, for checking the development repository.

uninstall

Uninstalls the entire TeX Live installation. Options:

--force Do not ask for confirmation, remove immediately.

generate language**generate language.dat****generate language.def****generate fmtutil****generate updmap**

The *generate* action overwrites any manual changes made in the respective files: it recreates them from scratch.

For `fmtutil` and the language files, this is normal, and both the TeX Live installer and `tlmgr` routinely call *generate* for them.

For `updmap`, however, neither the installer nor `tlmgr` use *generate*, because the result would be to disable all maps which have been manually installed via `updmap-sys`

--enable, e.g., for proprietary or local fonts. Only the changes in the --localcfg file mentioned below are incorporated by *generate*.

On the other hand, if you only use the fonts and font packages within TeX Live, there is nothing wrong with using *generate updmap*. Indeed, we use it to generate the updmap.cfg file that is maintained in the live source repository.

In more detail: *generate* remakes any of the four config files language.dat, language.def, fmtutil.cnf, and updmap.cfg from the information present in the local TLPDB. If the files language-local.dat, language-local.def, fmtutil-local.cnf, or updmap-local.cfg are present under TEXMFLOCAL in the respective directories, their contents will be simply merged into the final files, with no error checking of any kind.

The form `generate language` recreates both the language.dat and the language.def files, while the forms with extension only recreates the given file.

Options:

--**dest** *output file*

specifies the output file (defaults to the respective location in TEXMFSYSVAR). If --dest is given to `generate language`, its value will be used for the language.dat output, and .def will be appended to the value for the name of the language.def output file. (This is just to avoid overwriting; if you want a specific name for each output file, we recommend invoking tlmgr twice.)

--**localcfg** *local conf file*

specifies the (optional) local additions (defaults to the respective location in TEXMFSYSVAR).

The respective locations are as follows:

tex/generic/config/language.dat (and language-local.dat)
tex/generic/config/language.def (and language-local.def)
web2c/fmtutil.cnf (and fmtutil-local.cnf)
web2c/updmap.cfg (and updmap-local.cfg)

AUTHORS AND COPYRIGHT

This script and its documentation were written for the TeX Live distribution (<<http://tug.org/texlive>>) and both are licensed under the GNU General Public License Version 2 or later.