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. For information on initially downloading and installing TeX Live, see <http://tug.org/texlive/acquire.html>.

The most up-to-date version of this documentation (updated nightly from the development sources) is available at <http://tug.org/texlive/tlmgr.html>, along with procedures for updating `tlmgr` itself and information about test versions.

TeX Live is organized into a few top-level *schemes*, each of which is defined as a different set of *collections* and *packages*, where a collection is a set of packages, and a package is what contains actual files. Schemes typically contain a mix of collections and packages, but each package is included in exactly one collection, no more and no less. Installation can be customized and managed at any level.

For the full documentation of TeX Live, see <http://tug.org/texlive/doc>.

EXAMPLES

After successfully installing TeX Live, here are a few common operations with `tlmgr`:

`tlmgr update --all`
: Make your local TeX installation correspond to what is in the package repository (typically on CTAN).

`tlmgr update --list`
: Report what would be updated without actually updating anything.

`tlmgr option repository http://mirror.ctan.org/systems/texlive/tlnet`
: Tell `tlmgr` to use a nearby CTAN mirror for future updates; useful if you installed TeX Live from the DVD image.

`tlmgr show` *pkgname*
: Display detailed information about *pkgname*, such as the installation status and description.

For all the capabilities and details of `tlmgr`, please see the following voluminous information.

OPTIONS

The following options to `tlmgr` are global options, not specific to any action. All options, whether global or action-specific, can be given at any place and in arbitrary order. The first non-option argument will be the main action. In all cases, `--`*option* and `-`*option* are equivalent, and an `=` is optional between an option name and its value.

`--`**repository** *url|path*
: Specifies the package repository from which packages should be installed or updated, overriding the default package repository found in the installation's TeX Live Package Database (TLPDB, a file named `texlive.tlpdb`). The documentation for `install-tl` has more details about this (<http://tug.org/texlive/doc/install-tl.html>).

  `--repository` changes the repository location only for the current run; to make a

permanent change, use `option repository` (see the option action).

For backward compatibility and convenience, `--location` and `--repo` are accepted as aliases for this option.

--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 (ISO 639-1) language code (with the exception for selecting simplified or traditional Chinese). Currently supported (but not necessarily completely translated) are: English (en, default), Czech (cs), German (de), French (fr), Italian (it), Dutch (nl), Polish (pl), Russian (ru), Slovak (sk), Slovenian (sl), Vietnamese (vi), simplified Chinese (zh-cn), and traditional Chinese (zh-tw).

--machine-readable
In lieu of the normal output intended for human consumption, write to standard output in a fixed format more suitable for machine parsing. See the ''MACHINE-READABLE OUTPUT'' section below for details.

--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 makes **tlmgr** wait for user input before exiting. Useful on Windows to avoid command windows disappearing.

--persistent-downloads
--no-persistent-downloads
For net installs, activating this option makes tlmgr try to set up a persistent connection using the Net::LWP Perl module. This opens only one connection between your computer and the server per session and reuses it, instead of initiating a new download for each package.

This option is turned on by default, and tlmgr will fall back to using wget if this is not possible. If you want to disable usage of LWP and persistent connections, please use **--no-persistent-downloads**.

--no-execute-actions
Suppress the execution of the execute actions as defined in the tlpsrc files. Only use at your own risk.

--debug-translation
In GUI mode, this switch makes `tlmgr` report any missing, or more likely untranslated, messages to standard error. Helpful for translators to see what remains to be done.

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 and about the `tlmgr` script itself. If paired with `-v`, revision number for the used TeX Live Perl modules are shown, too.

ACTIONS

help

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

version

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

If `-v` has been given the revisions of the used modules are reported, too.

gui

Start the graphical user interface. See GUI below.

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

Install each *pkg* given on the command line. By default this installs all packages on which the given *pkg*s 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. That is, for a package `foo`, the package `foo.i386-linux` will also be installed on an `i386-linux` system. This switch suppresses this behavior, 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 `tlmgr` itself (or other parts of the basic infrastructure) are present, `tlmgr` will bail out and not perform the installation unless this option is given. Not recommended.

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 except for `tlmgr` itself. Thus, if updates to `tlmgr` itself are present, this will simply give an error, unless also the option `--force` or `--self` is given. (See below.)

In addition to updating the installed packages, during the update of a collection the local installation is (by default) synchronized to the status of the collection on the server, for both additions and removals.

This means that if a package has been removed on the server (and thus has also been removed from the respective collection), `tlmgr` will remove the package in the local installation. This is called ‘‘auto−remove’’ and is announced as such when using the option `−−list`. This auto-removal can be suppressed using the option `−−no−auto−remove`.

Analogously, if a package has been added to a collection on the server that is also installed locally, it will be added to the local installation. This is called ‘‘auto−install’’ and is announced as such when using the option `−−list`. This auto-installation can be suppressed using the option `−−no−auto−install`.

An exception to the collection dependency checks (including the auto-installation of packages just mentioned) are those that have been ‘‘forcibly removed’’ by you, that is, you called `tlmgr remove −−force` on them. (See the `remove` action documentation.) To reinstall any such forcibly removed packages use `−−reinstall−forcibly−removed`.

If you want to exclude some packages from the current update run (e.g., due to a slow link), see the `−−exclude` option below.

−−self

Update `tlmgr` itself (that is, the infrastructure packages) if updates to it are present. On Windows this includes updates to the private Perl interpreter shipped inside TeX Live.

If this option is given together with either `−−all` or a list of packages, then `tlmgr` will be updated first and, if this update succeeds, the new version will be restarted to complete the rest of the updates.

In short:

```
tlmgr update −−self        # update infrastructure only
tlmgr update −−self −−all  # update infrastructure and all packages
tlmgr update −−force −−all # update all packages but *not* infrastructu
                           # ... this last at your own risk, not recomm
```

−−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`.

−−list [*pkg*]

Concisely list the packages which would be updated, newly installed, or removed, without actually changing anything. If no package is given, this acts like `−−list −−all`, otherwise it lists only updates to the packages given as arguments (and its dependencies).

−−exclude *pkg*

Exclude *pkg* from the update process. If this option is given more than once, its arguments accumulate.

An argument *pkg* excludes both the package *pkg* itself and all its related platform-specific packages *pkg.ARCH*. For example,

```
tlmgr update −−all −−exclude a2ping
```

will not update `a2ping`, `a2ping.i386−linux`, or any other `a2ping.`*ARCH* package.

If this option specifies a package that would otherwise be a candidate for auto-installation, auto-removal, or reinstallation of a forcibly removed package, `tlmgr` quits with an error message. Excludes are not supported in these circumstances.

`--`**no-auto-remove** [*pkg*]...
Under normal circumstances `tlmgr` tries to remove packages which have disappeared on the server, as described above under `--all`. This option prevents any such removals, either for all packages (with `--all`), or the given *pkg* names.

`--`**no-auto-install** [*pkg*]...
Under normal circumstances `tlmgr` will install packages which are new on the server, as descrbed above under `--all`. This option prevents any such automatic installation, either for all packages (with `--all`), or the given *pkg* names.

Furthermore, after the `tlmgr` run using this has finished, the packages that would have been auto-installed *will be considered as forcibly removed*. So, if `foobar` is the only new package on the server, then

```
tlmgr update --all --no-auto-install
```

is equivalent to

```
tlmgr update --all
tlmgr remove --force foobar
```

`--`**reinstall-forcibly-removed**
Under normal circumstances `tlmgr` will not install packages that have been forcibly removed by the user; that is, removed with `remove --force`, or whose installation was prohibited by `--no-auto-install` during an earlier update.

This option makes `tlmgr` ignore the forcible removals and re-install all such packages. This can be used to completely synchronize an installation with the server's idea of what is available:

```
tlmgr update --reinstall-forcibly-removed --all
```

`--`**backup** and `--`**backupdir** *directory*
These two options control the creation of backups of packages *before* updating; that is, backup of packages as currently installed. If neither of these options 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. In contrast, 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 behavior.

--no-depends-at-all
See above under **install** (and beware).

--force
Force update of normal packages, without updating `tlmgr` itself (unless the `--self` option is also given). Not recommended.

Also, `update --list` is still performed regardless of this option.

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 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, as explained above.

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

restore [--backupdir *dir***] [--all** | *pkg* **[***rev***]]**
Restore a package from a previously-made backup.

If `--all` is given, try to restore the latest revision of all package backups found in the backup directory.

Otherwise, 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. If not given it will be taken from the configuration setting in the TLPDB.

--all
Try to restore the latest revision of all package backups found in the backup directory. Additional non-option arguments (like *pkg*) are not allowed.

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

--no-depends
Do not remove dependent packages.

--no-depends-at-all
See above under **install** (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.

A package that has been removed using the `--force` option because it is still listed in an installed collection or scheme will not be updated, and will be mentioned as **forcibly removed** in the output of **tlmgr update --list**.

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

option
option [show]
option showall
option *key* [*value*]

The first form shows the global TeX Live settings currently saved in the TLPDB with a short description and the `key` used for changing it in paranthesis.

The second form acts like the first, but also shows options which can be defined but are not currently set to any value.

In the third 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 (but see **tlmgr option showall** for the definitive list):

```
repository (default package repository),
formats    (create formats at installation time),
docfiles   (install documentation files),
srcfiles   (install source files),
backupdir  (default directory for backups),
autobackup (number of backups to keep).
sys_bin    (directory to which executables are linked by the path action)
sys_man    (directory to which man pages are linked by the path action)
sys_info   (directory to which Info files are linked by the path action)
```

One common use of `option` is to permanently change the installation to get further updates from the Internet, after originally installing from DVD. To do this, you can run

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

The `install-tl` documentation has more information about the possible values for repository.

To keep backward compatibility `location` can be used as alternative name for `repository`.

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 automatic generation of backups. Its value is an integer. If the `autobackup` value is `-1`, no backups are removed. If `autobackup` is 0 or more, it specifies the number of backups to keep. Thus, backups are disabled if the value is 0. In the `--clean` mode of the `backup` action this option also specifies the number to be kept.

To setup `autobackup` to `-1` on the command line, use

```
tlmgr option autobackup infty
```

Or you can use:

```
tlmgr option -- autobackup -1
```

The `--` avoids having the `-1` treated as an option. (For most programs, `--` stops parsing for arguments at the point where it appears; this is a general feature of option parsing.)

conf [texmf|tlmgr [*key* **[***value***]]]**

With only the `conf`, show general configuration information for TeX Live, including active configuration files, path settings, and more. This is like the `texconfig conf` call, but works on all supported platforms.

With either `texmf` or `tlmgr` given in addition, shows all key/value pairs (i.e., all settings) as saved in `ROOT/texmf.cnf` or the tlmgr configuration file (see below), respectively.

If *key* is given in addition, shows the value of only that given *key* in the respective file.

If *value* is given in addition, *key* is set to *value* in the respective file. *No error checking is done!*

Practical application: if the execution of (some or all) system commands via `\write18` was left enabled during installation, you can disable it afterwards:

```
tlmgr conf texmf shell_escape 0
```

WARNING: The general facility is here, but tinkering with settings in this way is very strongly discouraged. Again, no error checking is done, so any sort of breakage is possible.

paper

paper [a4|letter]
xdvi|pdftex|dvips|dvipdfmx|dvipdfm|context paper [*papersize*|−−list]

With no arguments (`tlmgr paper`), shows the default paper size setting for all known programs.

With one argument (e.g., `tlmgr paper a4`), sets the default for all known programs to that paper size.

With a program given as the first argument and no paper size specified (e.g., `tlmgr dvips paper`), shows the default paper size for that program.

With a program given as the first argument and a paper size as the last argument (e.g., `tlmgr dvips paper a4`), set the default for that program to that paper size.

With a program given as the first argument and `−−list` given as the last argument (e.g., `tlmgr dvips paper −−list`), shows all valid paper sizes for that program. The first size shown is the default.

Incidentally, this syntax of having a specific program name before the `paper` keyword may seem strange. It is inherited from the longstanding `texconfig` script, which supports other configuration settings for some programs, notably `dvips`. `tlmgr` does not support those extra settings at present.

platform list|add|remove *platform*...

`platform list` lists the TeX Live names of all the platforms (a.k.a. architectures), (`i386−linux`, ...) available at the package repository.

`platform add` *platform*... adds the executables for each given platform *platform* to the installation from the repository.

`platform remove` *platform*... removes the executables for each given platform *platform* from the installation, but keeps the currently running platform in any case.

`arch` is a synonym for `platform`.

Options:

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

print-platform
Print the TeX Live identifier for the detected platform (hardware/operating system) combination to standard output, and exit. `−−print−arch` is a synonym.

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***...**

Display 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.

It also displays the information taken from the TeX Catalogue (license, date, version), but note that there is a high probability that this information is slightly off due to timing issues.

If the option `−−list` is given with a package, the list of contained files is also shown, including those for platform-specific dependencies. When given with schemes and collections, `−−list` outputs their dependencies in a similar way.

list [−−only−installed] [collections|schemes|*pkg***...]**

With no argument, lists all packages available at the package repository, 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.

If the option `−−only−installed` is given the installation source will not be used and only locally installed packages, collections, or schemes are listed.

check [*option***]... [files|depends|executes|runfiles|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.

depends

Lists those packages which occur as dependencies in an installed collections, but are themselves not installed, and those packages that are not contained in any collection.

If you call `tlmgr check collections` this test will be carried out instead since former versions for `tlmgr` called it that way.

executes

Check that the files referred to by `execute` directives in the TeX Live Database are present.

runfiles

List those filenames that are occurring more than one time in the runfiles.

Options:

−−use−svn

Use the output of `svn status` instead of listing the files; for checking the TL development repository.

path [−−w32mode=user|admin] [add|remove]

On Unix adds or removes symlinks for binaries, man pages, and info pages in the directories specified by the respective options (see above).

On Windows, the registry part where the binary directory is added or removed is determined in the following way:

If the user has admin rights, and the option `−−w32mode` is not given, the setting *w32_multi_user* determines the location (i.e., if it is on then the system path, otherwise the user path is changed).

If the user has admin rights, and the option `--w32mode` is given, this option determines the path to be adjusted.

If the user does not have admin rights, and the option `--w32mode` is not given, and the setting *w32_multi_user* is off, the user path is changed, while if the setting *w32_multi_user* is on, a warning is issued that the caller does not have enough privileges.

If the user does not have admin rights, and the option `--w32mode` is given, it must be **user** and the user path will be adjusted. If a user without admin rights uses the option `--w32mode admin` a warning is issued that the caller does not have enough provileges.

postaction [--w32mode=user|admin] [--fileassocmode=1|2] [--all] [install|remove] [shortcut|fileassoc|script] [*pkg***]...**
Carry out the postaction `shortcut`, `fileassoc`, or `script` given as the second required argument in install or remove mode (which is the first required argument), for either the packages given on the command line, or for all if `--all` is given.

The option `--w32mode` is `user` all actions will only carried out in the user accessible parts of the registry/filesystem, while the `admin` mode selects the system wide parts of the registry for the file associations. Note that if you do not have enough permissions using `--w32mode=admin` will not succeed.

For the postaction `fileassoc` the mode can be set with `--fileassocmode`. If it is set to 1, only new assocations are added, if it is set to 2, all associations are set to the TeX Live programs.

uninstall
Uninstalls the entire TeX Live installation. Options:

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

generate [*option***]...** *what*
generate language
generate language.dat
generate language.def
generate language.dat.lua
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, `tlmgr` does *not* 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. The `generate` action only incorporates the changes in the `--localcfg` file mentioned below. Furthermore, `tlmgr` updates and maintains the final `updmap.cfg` in `TEXMFSYSCONFIG` (while the other files are in `TEXMFSYSVAR`), because that is the location that `updmap-sys` (via `tcfmgr`) uses.

Notwithstanding the above, if you only use the fonts and font packages within TeX Live, and maintain your local fonts (if any) using `updmap-local.cfg`, there is nothing wrong with using `generate updmap`. Indeed, we use it ourselves to generate the `updmap.cfg` file in

the live source repository.

In more detail: `generate` remakes any of the five config files `language.dat`, `language.def`, `language.dat.lua`, `fmtutil.cnf`, and `updmap.cfg` from the information present in the local TLPDB, plus locally-maintained files.

The locally-maintained files are `language-local.dat`, `language-local.def`, `language-local.dat.lua`, `fmtutil-local.cnf`, or `updmap-local.cfg`, searched for in `TEXMFLOCAL` in the respective directories. If they are present, the final file is made by starting with the main file, omitting any entries that the local file specifies to be disabled, and finally appending the local file.

Local files specify entries to be disabled with a comment line like this:

```
#!NAME
%!NAME
--!NAME
```

where `fmtutil.cnf` and `updmap.cfg` use #, `language.dat` and `language.def` use %, and `language.dat.lua` use `--`. In any case, the *name* is the respective format name, map file name (include the `.map` extension), or hyphenation pattern identifier. Examples:

```
#!pdflatex
#!lm.map
%!german
--!usenglishmax
```

(Of course, you're not likely to actually want to disable those particular items. They're just examples.)

After such a disabling line, the local file can include another entry for the same item, if a different definition is desired. In general, except for the special disabling lines, the local files follow the same syntax as the master files.

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

Special consideration for updmap.cfg: in addition to font map files, this file specifies the setting of five options: `dvipsPreferOutline`, `LW35`, `dvipsDownloadBase35`, `pdftexDownloadBase14`, and `dvipdfmDownloadBase14`. The defaults for these as set in `updmap-hdr.cfg` are usually fine. If you want to change them, you can include changed settings for any or all of these five options in your `updmap-local.cfg` file and they will be respected by `generate updmap`, for example:

```
dvipsDownloadBase35 true
```

Options:

`--`**dest** *output file*

specifies the output file (defaults to the respective location in `TEXMFSYSVAR` for `language*` and `fmtutil`, and `TEXMFSYSCONFIG` for `updmap`). If `--dest` is given to `generate language`, it serves as a basename onto which `.dat` will be appended for the name of the `language.dat` output file, `.def` will be appended to the value for the name of the `language.def` output file, and `.dat.lua` to the name of the `language.dat.lua` 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 `TEXMFLOCAL`).

`--`**rebuild-sys**
tells tlmgr to run necessary programs after config files have been regenerated. These are: `updmap-sys` after `generate updmap`, `fmtutil-sys --all` after `generate fmtutil`, `fmtutil-sys --byhyphen .../language.dat` after `generate language.dat`, and `fmtutil-sys --byhyphen .../language.def` after `generate language.def`.

These subsequent calls cause the newly-generated files to actually take effect. This is not done by default since those calls are lengthy processes and one might want to made several related changes in succession before invoking these programs.

The respective locations are as follows:

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

Final repetition: as explained above, `tlmgr` does *not* use the equivalent of `generate updmap` for map files. Therefore, if you want to make use of `updmap-local.cfg`, you need to run `tlmgr generate updmap` yourself. After that, network updates and your local changes should be merged.

CONFIGURATION FILE

A small subset of the command line options can be set in a config file for `tlmgr` which resides in `TEXMFVAR/tlmgr/config`. Note that it is **not** `TEXMFSYSVAR` so the file is specific to a single user. By default thus the config file is in `~/.texlive2010/tlmgr/config`.

Format of this file is: empty lines and lines starting with # are ignored. All other lines must look like

```
key = value
```

where the allowed keys are `gui_expertmode` and `persistent-downloads`. The values can only be 0 or 1 for those settings. `persistent-downloads` corresponds to the respective command line options of `tlmgr`. `gui_expertmode` switches between the full GUI and a simplified with only the important and mostly used settings.

GUI

The graphical user interface for `tlmgr` needs Perl/TK being installed. For Windows the necessary modules are shipped within TeX Live, for all other (i.e., Unix-based) systems Perl/Tk (as well as perl of course) has to be installed.

When started with `tlmgr gui` the graphical user interface will be shown. The main window contains a menu bar, the main display, and a status area where messages normally shown on the console are displayed.

Within the main display there are three main parts: The *Display configuration*, the list of packages, and the buttons for actions. In addition to these three on the top right there is some text showing the currently loaded repository (this text also acts as button and will load the default

repository).

Menu bar

The following entries can be found in the menu bar:

tlmgr

Provides access to load various repositories (the default as specified in the texlive database, the default network repository, if given the repository specified on the command line, and an arbitrary other one. Furthermore is allows to quit `tlmgr`.

Options

Provides access to three groups of options, *General* (for almost all options), *Paper* (configuration of default paper sizes), *Platforms* (only on Unix, configuration of the supported/installed platforms), as well as some toggles to turn on and off various options.

There is also a toggle for `Expert options` which defaults to on. If you turn this off the next time you start the GUI a simplified screen will be shown that exhibits only the most important and necessary functionality. This setting is saved in the configuration file of `tlmgr`, see ''CONFIGURATION FILE'' for details.

Actions

Provides access to a variety of items, such as updating the filename database (aka ls-R, mktexlsr, texhash), rebuilding of all formats (`fmtutil-sys --all`), updating the font map database (`updmap-sys`), and handling of symbolic links in system directories (only Unix), restoring backups of packages, as well as removal of the full TeX Live installation (only Unix).

Help

Provides access to the manual and other basic information on the installed version, authors, license.

Main display

The main display lists all packages, installed and, if a repository is loaded, also those that are available but not installed.

Each line of the package list contains of the following items:

a checkbox

used for selecting packages, some of the action buttons (see below) will work only on the selected packages.

the package name

that is the name of the package as given in the database.

local revision (and version)

If the package is installed the revision of the installed package will be shown. If there is a catalogue version given in the database for this package, it will be shown in parenthesis.

remote revision (and version)

If a repository has been loaded the revision of the package in the repository (if present) is shown. As with the local variant, if a catalogue version is provided it will be displayed.

short description

The short description (or title) is shown.

Double clicking on one of the lines pops up an informational window with further details, the

long description, included files, etc.

Above the list of package there are options to configure the list of packages to be shown. Changing any of them will automatically update the list of packages. The different display configurations are:

Status
: allows selecting to show all, only the installed, only those packages that are not installed, or only those with packages available.

Category
: allows to select which categories are shown.

Match
: allows searching for a specific pattern. This uses the same algorithm as `tlmgr search`, i.e., searches in the title, short and long descriptions.

Selection
: this allows to restrict the list of packages to those selected, not selected, or all (selected means that the checkbox in the beginning of the line of a package is selected).

buttons
: to the right there are three buttons, one to select all packages, one to select none (deselect all), and one to reset all filters to the defaults, i.e., show all available.

Below the list of packages there are five buttons:

Update all installed
: this calls `tlmgr update --all` internally, i.e., tries to update all available packages.

  Below this button there is a toggle that allows reinstallation of previously removed packages.

  The following four buttons only work on the *selected* packages, i.e., those where the checkbutton at the beginning of the line is ticked.

Update
: only update the selcted packages

Install
: install the selected packages, acts like `tlmgr install`, i.e., also installs dependencies. This installing a collection will install all depending packages, too.

Remove
: removes the selected packages, acts like `tlmgr remove`, i.e., it will also remove dependencies of collections (but not dependencies of normal packages).

Backup
: makes a backup of the selected packages, acts like `tlmgr backup`. This action needs the option `backupdir` set (see Options -> General).

Finally, the status area at the bottom of the window gives additional information what is going on.

the old (v1) GUI

For the time being the first GUI written for `tlmgr` is still included and available via <tlmgr gui-old>. Note that there is no development of this GUI anymore, so at some point it might stop to actually work, or even get removed from the packages.

MACHINE-READABLE OUTPUT

Given the `--machine-readable` option, `tlmgr` writes to stdout in the fixed line-oriented format described here, and the usual informational messages for human consumption are written to stderr (normally they are written to stdout). The idea is that a program can get all the information it needs by reading stdout.

Currently this option only applies to the update, the install, and the option actions.

update and install actions

The output format is as follows:

```
fieldname "\t" value
...
"end-of-header"
pkgname status localrev serverrev size runtim esttot
...
"end-of-updates"
other output from post actions, not in machine readable form
```

The header section currently has two fields: `location-url` (the repository source from which updates are being drawn), and `total-bytes` (the total number of bytes to be downloaded).

The *localrev* and *serverrev* fields for each package are the revision numbers in the local installation and server repository, respectively. The *size* field is the number of bytes to be downloaded, i.e., the size of the compressed tar file for a network installation, not the unpacked size. The runtim and esttot fields are only present for updated and auto-install packages, and contain the currently passed time since start of installation/updates and the estimated total time.

Line endings may be either LF or CRLF depending on the current platform.

`location-url` *location*

The *location* may be a url (including `file:///foo/bar/...`), or a directory name (`/foo/bar`). It is the package repository from which the new package information was drawn.

`total-bytes` *count*

The *count* is simply a decimal number, the sum of the sizes of all the packages that need updating or installing (which are listed subseqently).

Then comes a line with only the literal string `end-of-header`.

Each following line until a line with literal string `end-of-updates` reports on one package. The fields on each line are separated by a tab. Here are the fields.

pkgname

The TeX Live package identifier, with a possible platform suffix for executables. For instance, `pdftex` and `pdftex.i386-linux` are given as two separate packages, one on each line.

status

The status of the package update. One character, as follows:

`d` The package was removed on the server.

`f` The package was removed in the local installation, even though a collection depended on it. (E.g., the user ran `tlmgr remove --force`.)

`u` Normal update is needed.

`r` Reversed non-update: the locally-installed version is newer than the version on the server.

`a` Automatically-determined need for installation, the package is new on the server and is (most probably) part of an installed collection.

`i` Package will be installed and isn't present in the local installation (action install).

`I` Package is already present but will be reinstalled (action install).

localrev
The revision number of the installed package, or − if it is not present locally.

serverrev
The revision number of the package on the server, or − if it is not present on the server.

size
The size in bytes of the package on the server. The sum of all the package sizes is given in the `total-bytes` header field mentioned above.

runtime
The run time since start of installations or updates.

esttot
The estimated total time.

option action
The output format is as follows:

```
key "\t" value
```

If a value is not saved in the database the string `(not set)` is shown.

If you are developing a program that uses this output, and find that changes would be helpful, do not hesitate to write the mailing list.

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.