

# Basic Usage of `tlmgr`, the T<sub>E</sub>X Live Manager

edited by Bob Tennent\*  
rdt@cs.queensu.ca

July 23, 2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Usage</b>	<b>2</b>
<b>3</b>	<b>Examples</b>	<b>3</b>
<b>4</b>	<b>Actions</b>	<b>4</b>
4.1	The <code>info</code> Action . . . . .	4
4.2	The <code>search</code> Action . . . . .	6
4.3	The <code>install</code> Action . . . . .	7
4.4	The <code>update</code> Action . . . . .	8
4.5	The <code>remove</code> Action . . . . .	10
4.6	The <code>option</code> Action . . . . .	11
4.7	The <code>path</code> Action . . . . .	13
<b>5</b>	<b>Global Options</b>	<b>14</b>

---

\*Derived from the man page `tlmgr.1` and licensed under the GNU General Public Licence, Version 2 or later.

## 1 Introduction

`tlmgr` manages an existing TeX Live installation, both packages and configuration options. For information on initially downloading and installing TeX Live, see <https://tug.org/texlive/acquire.html>.

TeX Live is organized into a few top-level *schemes*, each of which specifies a 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. A TeX Live installation can be customized and managed at any level.

We will describe usage of `tlmgr` in the command-line interface. There are graphical front-ends for `tlmgr`: `tlshell`, `tlcockpit` and, for Macs, a TeX Live Utility; these must be started as separate programs. There is also a graphical user interface in `tlmgr` itself; consult the documentation here:

<https://tug.org/texlive/doc/tlmgr.html#GUI-FOR-TLMGR>.

## 2 Usage

`tlmgr` [*global options*] *action* [*action-specific options*] [*operands*]

The actions and relevant action-specific options and operands are described in Section 4. The global options are described in Section 5.

Options, whether global or action-specific, may be given anywhere on the command line, and in any order. The first non-option argument will be treated as the main action. In all cases, `--option` and `-option` are equivalent, and an `=` is optional between an option name and its value.

This report documents only the most commonly-used actions and options for `tlmgr`. Complete documentation may be found at

<https://tug.org/texlive/doc/tlmgr.html>.

### 3 Examples

After successfully installing T<sub>E</sub>X Live, the following are examples of common operations with `tlmgr`:

```
tlmgr -help install
```

Displays reference documentation on the `install` action.

```
tlmgr option repository ctan
```

Tells `tlmgr` to use a nearby CTAN mirror for future updates; `ctan` is just an alias for `http://mirror.ctan.org/systems/texlive/tlnet`. Caveat: `mirror.ctan.org` resolves to many different hosts, and they are not perfectly synchronized; we recommend updating only daily (at most), and not more often. You can choose a particular mirror in place of `mirror.ctan.org` if there are problems; the list of all CTAN mirrors with the status of each is at `https://ctan.org/mirrors/mirmon`. Don't forget to add `.../systems/texlive/tlnet` to the path.

```
tlmgr update -all
```

Makes your local T<sub>E</sub>X installation correspond to what is in the default package repository.

```
tlmgr update -self
```

Updates `tlmgr` itself (that is, the infrastructure packages) if updates to it are present. If the `-self` 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.

```
tlmgr info fontspec
```

Displays detailed information about package `fontspec`, such as the installation status and description.

```
tlmgr info -list fontspec
```

The list of files in package `fontspec` is also displayed, including those for platform-specific dependencies.

```
tlmgr install fontspec
```

Installs the `fontspec` package (unless it is already installed). By default all packages on which the given package is dependent are also installed. Existing packages are not touched.

```
tlmgr search -file cabin
```

Search locally installed packages and display all filenames containing `cabin`.

## 4 Actions

### 4.1 The info Action

#### Usage:

```
tlmgr info
tlmgr info collections
tlmgr info schemes
tlmgr info pkgs
```

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 all the collections and schemes, respectively, instead of all packages.

With any other arguments, displays information about each of the *pkgs*: the name, category, short and long description, sizes, installation status, and T<sub>E</sub>X Live revision number. If the package is not locally installed, searches in the remote installation source.

For normal packages (not collections or schemes), the sizes of the four groups of files (`run/src/doc/bin`) are shown separately. For collections, the cumulative size is shown, including all directly-dependent packages (but not dependent collections). For schemes, the cumulative size is also shown, including all directly-dependent collections and packages.

If a package is not found locally or remotely, the search action (see Section 4.2) is used and lists matching packages and files.

It also displays information taken from the T<sub>E</sub>X Catalogue, namely the package version, date, and license. Consider these, especially the package version, as approximations only, due to timing skew of the updates of the different pieces. By contrast, the revision value comes directly from TL and is reliable.

#### Options for info:

`-list`

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.

`-only-installed`

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

**-only-remote**

Only list packages from the remote repository. Useful when checking what is available in a remote repository using

```
tlmgr -repo ... -only-remote info.
```

Note that `-only-installed` and `-only-remote` cannot both be specified.

## 4.2 The search Action

### Usage:

```
tlmgr search what
```

By default, search the names, short descriptions, and long descriptions of all locally installed packages for the argument *what*, interpreted as a (Perl) regular expression.

### Options for search:

`-file`

List all filenames containing *what*.

`-global`

Search the T<sub>E</sub>X Live Database of the installation medium, instead of the local installation.

`-word`

Restrict the search of package names and descriptions (but not filenames) to match only full words. For example, searching for `table` with this option will not output a package containing the word `tables` (unless it also contains the word `table` on its own).

### 4.3 The install Action

#### Usage:

```
tlmgr install pkgs
```

Install each of the *pkgs* (unless it is already installed). By default all packages on which the given package is dependent are also installed. Existing packages are not touched; use the update action described in Section 4.4 to get the latest version of a package.

#### Options for install:

##### -dry-run

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

##### -reinstall

Reinstall a package (including dependencies for collections) even if it already seems to be installed (i.e, is present in the T<sub>E</sub>X Live Package Database). This is useful to recover from accidental removal of files in the hierarchy.

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

##### -with-doc, -with-src

The `install-tl` program provides the option of omitting installation of all documentation and/or source files. (By default, everything is installed.) After such an installation, you may find that you want the documentation or source files for a given package after all. You can get them by using these options in conjunction with `-reinstall`, as in

```
tlmgr install -reinstall -with-doc -with-src pkgs
```

##### -no-dependends

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

The `install` action does not automatically add new symlinks in system directories; you need to run

```
tlmgr path add
```

yourself if you are using this feature and want new symlinks added; see Section 4.7.

## 4.4 The update Action

### Usage:

```
tlmgr update pkgs
```

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

### Options for update:

#### `-all`

Update all installed packages (except for `tlmgr` itself). If updates to `tlmgr` itself are listed, this gives 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`.

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`.

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.



**-dry-run**

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

**-list [pkg]**

Concisely list the packages which would be updated, newly installed, or removed, without actually changing anything. If `-all` is also given, all available updates are listed. If `-self` is given, but not `-all`, only updates to the critical packages (tlmgr, T<sub>E</sub>X Live infrastructure, Perl on Windows, etc.) are listed.

If neither `-all` nor `-self` is given, and in addition no *pkg* is given, then `-all` is assumed. Thus, `tlmgr update -list` is the same as `tlmgr update -list -all`. If neither `-all` nor `-self` is given, but specific package names are given, those packages are checked for updates.

**-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 any related platform-specific packages. For example,

```
tlmgr update -all -exclude a2ping
```

will not update `a2ping`, `a2ping.i386-linux`, or any other `a2ping.xxx` 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-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.

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.

This action does not automatically add or remove new symlinks in system directories. You need to run `tlmgr path` yourself if you are using this feature and want new symlinks added; see Section 4.7.

## 4.5 The remove Action

### Usage:

```
tlmgr remove pkgs
```

Remove each of the *pkgs*. Removing a collection removes all package dependencies (unless `-nodepends` is specified), but not any collection dependencies of that collection. However, when removing a package, dependencies are never removed.

### Options for `remove`:

`-dry-run`

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

`-no-depends`

Do not remove dependent packages.

`-all`

Uninstalls all of T<sub>E</sub>X Live, asking for confirmation unless `-force` is also specified.

The `remove` action does not automatically remove symlinks in system directories; you need to run

```
tlmgr path remove
```

yourself if you are using this feature and want stale symlinks removed; see Section 4.7.

```
tlmgr path [-w32mode=user|admin] add
```

```
tlmgr path [--w32mode=user|admin] remove
```

## 4.6 The option Action

### Usage:

```

tlmgr option
tlmgr option help
tlmgr option key
tlmgr option key value

```

The first form shows the global T<sub>E</sub>X Live settings currently saved with a short description and the key used for changing it in parentheses. The second form is similar, but also shows options which can be defined but are not currently set to any value. In the third form, the setting for *key* is displayed. In the fourth form, *key* is set to *value*.

Some of the possible values for *key* are (run `tlmgr option help` for the definitive list):

repository	default package repository,
docfiles	install documentation files,
srcfiles	install source files,
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
desktop_integration	Windows-only: create Start menu shortcuts
fileassocs	Windows-only: change file associations
multiuser	Windows-only: install for all users

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

```
tlmgr option repository ctan
```

to use a nearby CTAN mirror for future updates.

The `docfiles` and `srcfiles` keys control the installation of their respective file groups (documentation, sources; grouping is approximate) per package. By default both are enabled (1). Either or both can be disabled (set to 0) if disk space is limited or for minimal testing installations, etc. When disabled, the respective files are not downloaded at all.

The `sys_bin`, `sys_man`, and `sys_info` options are used on Unix systems to control the generation of links for executables, info files and man pages. See the `path` action in Section 4.7 for details.

The remaining options affect behavior on Windows installations.

- If `desktop_integration` is set, then some packages will install items in a sub-folder of the Start menu for `tlmgr gui`, documentation, etc.

- If `fileassocs` is set, Windows file associations are made (see also the `postaction` action).
- If `multiuser` is set, then adaptations to the registry and the menus are done for all users on the system instead of only the current user.

All three of these options are on by default.

## 4.7 The path Action

### Usage:

```
tlmgr path [-w32mode=user|admin] add
```

```
tlmgr path [--w32mode=user|admin] remove
```

On Unix, adds or removes symlinks for executables, man pages, and info pages in the system directories specified by the respective options (see the option description above). Does not change any initialization files, either system or personal. Furthermore, any executables added or removed by future updates are not taken care of automatically; this command must be rerun as needed.

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 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 privileges.

## 5 Global Options

`--help, -help, -h, -?`

These options display reference documentation for any action.

`--version, -version`

Displays version information about the T<sub>E</sub>X Live release and the `tlmgr` script itself. If `-v` is also given, revision numbers for the loaded T<sub>E</sub>X Live Perl modules are shown as well.

`-q`

Suppresses informational messages.

`-v`

Displays debugging messages; can be repeated for more output.

`-command-logfile file`

`tlmgr` logs the output of all programs invoked (`mktexlr`, `mtxrun`, `fmtutil`, `updmap`) to a log file, by default `TEXMFSYSVAR/web2c/tlmgr-commands.log`. This option allows you to specify a different *file* for the log.

`-package-logfile file`

`tlmgr` logs all package actions (`install`, `remove`, `update`, `failed updates`, `failed restores`) to a log file, by default `TEXMFSYSVAR/web2c/tlmgr.log`. This option allows you to specify a different *file* for the log.

`-pause`

This option makes `tlmgr` wait for user input before exiting. Useful on Windows to avoid disappearing command windows.