

# Package ‘tinytex’

July 24, 2020

**Type** Package

**Title** Helper Functions to Install and Maintain TeX Live, and Compile LaTeX Documents

**Version** 0.25

**Description** Helper functions to install and maintain the 'LaTeX' distribution named 'TinyTeX' (<<https://yihui.org/tinytex/>>), a lightweight, cross-platform, portable, and easy-to-maintain version of 'TeX Live'. This package also contains helper functions to compile 'LaTeX' documents, and install missing 'LaTeX' packages automatically.

**Imports** xfun (>= 0.5)

**Suggests** testit, rstudioapi

**License** MIT + file LICENSE

**URL** <https://github.com/yihui/tinytex>

**BugReports** <https://github.com/yihui/tinytex/issues>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Author** Yihui Xie [aut, cre, cph] (<<https://orcid.org/0000-0003-0645-5666>>),  
RStudio, PBC [cph],  
Fernando Cagua [ctb],  
Ethan Heinzen [ctb]

**Maintainer** Yihui Xie <[xie@yihui.name](mailto:xie@yihui.name)>

**Repository** CRAN

**Date/Publication** 2020-07-24 04:30:02 UTC

## R topics documented:

<code>copy_tinytex</code> . . . . .	2
<code>install_tinytex</code> . . . . .	3

latexmk . . . . .	4
parse_install . . . . .	6
parse_packages . . . . .	6
r_texmf . . . . .	7
tlmgr . . . . .	8
tl_pkgs . . . . .	10
tl_sizes . . . . .	10

<b>Index</b>	<b>11</b>
--------------	-----------

---

copy_tinytex	<i>Copy TinyTeX to another location and use it in another system</i>
--------------	--

---

## Description

The function `copy_tinytex()` copies the existing TinyTeX installation to another directory (e.g., a portable device like a USB stick). The function `use_tinytex()` runs `tlmgr path add` to add the copy of TinyTeX in an existing folder to the `PATH` variable of the current system, so that you can use utilities such as `tlmgr` and `pdflatex`, etc.

## Usage

```
copy_tinytex(
  from = tinytex_root(),
  to = select_dir("Select Destination Directory")
)

use_tinytex(from = select_dir("Select TinyTeX Directory"))
```

## Arguments

<code>from</code>	The root directory of the TinyTeX installation. For <code>copy_tinytex()</code> , the default value <code>tinytex_root()</code> should be a reasonable guess if you installed TinyTeX via <code>install_tinytex()</code> . For <code>use_tinytex()</code> , if <code>from</code> is not provided, a dialog for choosing the directory interactively will pop up.
<code>to</code>	The destination directory where you want to make a copy of TinyTeX. Like <code>from</code> in <code>use_tinytex()</code> , a dialog will pop up if <code>to</code> is not provided in <code>copy_tinytex()</code> .

## Note

You can only copy TinyTeX and use it in the same system, e.g., the Windows version of TinyTeX only works on Windows.

---

install_tinytex	<i>Install/Uninstall TinyTeX</i>
-----------------	----------------------------------

---

## Description

The function `install_tinytex()` downloads the installation script from <https://github.com/yihui/tinytex> according to the platform (Unix or Windows), and executes it to install TinyTeX (a custom LaTeX distribution based on TeX Live). The function `uninstall_tinytex()` removes TinyTeX; `reinstall_tinytex()` reinstalls TinyTeX as well as previously installed LaTeX packages by default; `tinytex_root()` returns the root directory of TinyTeX if found.

## Usage

```
install_tinytex(
  force = FALSE,
  dir = "auto",
  repository = "ctan",
  extra_packages = NULL,
  add_path = TRUE
)

uninstall_tinytex(force = FALSE, dir = tinytex_root())

reinstall_tinytex(packages = TRUE, dir = tinytex_root(), ...)

tinytex_root(error = TRUE)
```

## Arguments

<code>force</code>	Whether to force to install (override) or uninstall TinyTeX.
<code>dir</code>	The directory to install or uninstall TinyTeX (should not exist unless <code>force = TRUE</code> ).
<code>repository</code>	The CTAN repository to be used. By default, a random fast mirror is automatically chosen (via <a href="http://mirror.ctan.org">http://mirror.ctan.org</a> ). You can manually set one if the automatic mirror is not really fast enough, e.g., if you are in China, you may consider ' <a href="http://mirrors.tuna.tsinghua.edu.cn/CTAN/">http://mirrors.tuna.tsinghua.edu.cn/CTAN/</a> ', or if you are in the midwest in the US, you may use ' <a href="https://mirror.las.iastate.edu/tex-archive/">https://mirror.las.iastate.edu/tex-archive/</a> '. You can find the full list of mirrors at <a href="https://ctan.org/mirrors">https://ctan.org/mirrors</a> . This argument should end with the path '/systems/texlive/tlnet', and if it is not, the path will be automatically appended.
<code>extra_packages</code>	A character vector of extra LaTeX packages to be installed.
<code>add_path</code>	Whether to run the command <code>tlmgr path add</code> to add the bin path of TeX Live to the system environment variable <code>PATH</code> .
<code>packages</code>	Whether to reinstall all currently installed packages.
<code>...</code>	Other arguments to be passed to <code>install_tinytex()</code> (note that the <code>extra_packages</code> argument will be set to <code>tl_pkgs()</code> if <code>packages = TRUE</code> ).

error Whether to signal an error if TinyTeX is not found.

## References

See the TinyTeX documentation (<https://yihui.org/tinytex/>) for the default installation directories on different platforms.

---

latexmk	<i>Compile a LaTeX document</i>
---------	---------------------------------

---

## Description

The function `latexmk()` emulates the system command `latexmk` (<https://ctan.org/pkg/latexmk>) to compile a LaTeX document. The functions `pdflatex()`, `xelatex()`, and `lualatex()` are wrappers of `latexmk(engine = , emulation = TRUE)`.

## Usage

```
latexmk(
  file,
  engine = c("pdflatex", "xelatex", "lualatex", "latex"),
  bib_engine = c("bibtex", "biber"),
  engine_args = NULL,
  emulation = TRUE,
  min_times = 1,
  max_times = 10,
  install_packages = emulation && tlmgr_writable(),
  pdf_file = gsub("tex$", "pdf", file),
  clean = TRUE
)

pdflatex(...)

xelatex(...)

lualatex(...)
```

## Arguments

file	A LaTeX file path.
engine	A LaTeX engine (can be set in the global option <code>tinytex.engine</code> , e.g., <code>options(tinytex.engine = 'xelatex')</code> ).
bib_engine	A bibliography engine (can be set in the global option <code>tinytex.bib_engine</code> ).
engine_args	Command-line arguments to be passed to engine (can be set in the global option <code>tinytex.engine_args</code> , e.g., <code>options(tinytex.engine_args = '-shell-escape')</code> ).
emulation	Whether to emulate the executable <code>latexmk</code> using R.

<code>min_times, max_times</code>	The minimum and maximum number of times to rerun the LaTeX engine when using emulation. You can set the global options <code>tinytex.compile.min_times</code> or <code>tinytex.compile.max_times</code> , e.g., <code>options(tinytex.compile.max_times = 3)</code> .
<code>install_packages</code>	Whether to automatically install missing LaTeX packages found by <code>parse_packages()</code> from the LaTeX log. This argument is only for the emulation mode and TeX Live. Its value can also be set via the global option <code>tinytex.install_packages</code> , e.g., <code>options(tinytex.install_packages = FALSE)</code> .
<code>pdf_file</code>	Path to the PDF output file. By default, it is under the same directory as the input file and also has the same base name. When <code>engine == 'latex'</code> , this will be a DVI file.
<code>clean</code>	Whether to clean up auxiliary files after compilation (can be set in the global option <code>tinytex.clean</code> , which defaults to TRUE).
<code>...</code>	Arguments to be passed to <code>latexmk()</code> (other than <code>engine</code> and <code>emulation</code> ).

## Details

The `latexmk` emulation works like this: run the LaTeX engine once (e.g., `pdflatex`), run `makeindex` to make the index if necessary (the `*.idx` file exists), run the bibliography engine `bibtex` or `biber` to make the bibliography if necessary (the `*.aux` or `*.bcf` file exists), and finally run the LaTeX engine a number of times (the maximum is 10 by default) to resolve all cross-references.

If `emulation = FALSE`, you need to make sure the executable `latexmk` is available in your system, otherwise `latexmk()` will fall back to `emulation = TRUE`. You can set the global option `options(tinytex.latexmk.emulation = FALSE)` to always avoid emulation (i.e., always use the executable `latexmk`).

The default command to generate the index (if necessary) is `makeindex`. To change it to a different command (e.g., `zhmakeindex`), you may set the global option `tinytex.makeindex`. To pass additional command-line arguments to the command, you may set the global option `tinytex.makeindex.args` (e.g., `options(tinytex.makeindex = 'zhmakeindex', tinytex.makeindex.args = c('-z', 'pinyin'))`).

If you are using the LaTeX distribution TinyTeX, but its path is not in the `PATH` variable of your operating system, you may set the global option `tinytex.tlgr.path` to the full path of the executable `tlmgr`, so that `latexmk()` knows where to find executables like `pdflatex`. For example, if you are using Windows and your TinyTeX is on an external drive `Z:/` under the folder `TinyTeX`, you may set `options(tinytex.tlgr.path = "Z:/TinyTeX/bin/win32/tlgr.bat")`. Usually you should not need to set this option because TinyTeX can add itself to the `PATH` variable during installation or via `use_tinytex()`. In case both methods fail, you can use this manual approach.

## Value

A character string of the path of the output file (i.e., the value of the `pdf_file` argument).

---

parse_install	<i>Parse the LaTeX log and install missing LaTeX packages if possible</i>
---------------	---

---

**Description**

This is a helper function that combines `parse_packages()` and `tlmgr_install()`.

**Usage**

```
parse_install(...)
```

**Arguments**

... Arguments passed to `parse_packages()`.

---

parse_packages	<i>Find missing LaTeX packages from a LaTeX log file</i>
----------------	--

---

**Description**

Analyze the error messages in a LaTeX log file to figure out the names of missing LaTeX packages that caused the errors. These packages can be installed via `tlmgr_install()`. Searching for missing packages is based on `tlmgr_search()`.

**Usage**

```
parse_packages(
  log,
  text = readLines(log),
  files = detect_files(text),
  quiet = rep(FALSE, 3)
)
```

**Arguments**

log	Path to the LaTeX log file (typically named ‘*.log’).
text	A character vector of the error log (read from the file provided by the log argument by default).
files	A character vector of names of the missing files (automatically detected from the log by default).
quiet	Whether to suppress messages when finding packages. It should be a logical vector of length 3: the first element indicates whether to suppress the message when no missing LaTeX packages could be detected from the log, the second element indicate whether to suppress the message when searching for packages via <code>tlmgr_search()</code> , and the third element indicates whether to warn if no packages could be found via <code>tlmgr_search()</code> .

**Value**

A character vector of LaTeX package names.

---

r_texmf	<i>Add/remove R's texmf tree to/from TeX Live</i>
---------	---

---

**Description**

R ships a custom texmf tree containing a few LaTeX style and class files, which are required when compiling R packages manuals ('Rd.sty') or Sweave documents ('Sweave.sty'). This tree can be found under the directory `file.path(R.home('share'), 'texmf')`. This function can be used to add/remove R's texmf tree to/from TeX Live via `tlmgr_conf('auxtrees')`.

**Usage**

```
r_texmf(action = c("add", "remove"))
```

**Arguments**

action            Add/remove R's texmf tree to/from TeX Live.

**References**

See the **tlmgr** manual for detailed information about `tlmgr conf auxtrees`. Check out <https://tex.stackexchange.com/q/77720/9128> if you don't know what texmf means.

**Examples**

```
# running the code below will modify your texmf tree; please do not run
# unless you know what it means

# r_texmf('remove')
# r_texmf('add')

# all files under R's texmf tree
list.files(file.path(R.home('share'), 'texmf'), recursive = TRUE, full.names = TRUE)
```

---

 tlmgr

*Run the TeX Live Manager*


---

## Description

Execute the `tlmgr` command to search for LaTeX packages, install packages, update packages, and so on.

## Usage

```
tlmgr(args = character(), usermode = FALSE, ..., .quiet = FALSE)

tlmgr_search(what, file = TRUE, all = FALSE, global = TRUE, word = FALSE, ...)

tlmgr_install(
  pkgs = character(),
  usermode = FALSE,
  path = !usermode && os != "windows",
  ...
)

tlmgr_remove(pkgs = character(), usermode = FALSE)

tlmgr_update(
  all = TRUE,
  self = TRUE,
  more_args = character(),
  usermode = FALSE,
  run_fmtutil = TRUE,
  delete_tlpdb = getOption("tinytex.delete_tlpdb", FALSE),
  ...
)

tlmgr_path(action = c("add", "remove"))

tlmgr_conf(more_args = character())
```

## Arguments

<code>args</code>	A character vector of arguments to be passed to the command <code>tlmgr</code> .
<code>usermode</code>	(For expert users only) Whether to use TeX Live's <b>user mode</b> . If <code>TRUE</code> , you must have run <code>tlmgr('init-usertree')</code> once before. This option allows you to manage a user-level texmf tree, e.g., install a LaTeX package to your home directory instead of the system directory, to which you do not have write permission. This option should not be needed on personal computers, and has some limitations, so please read the <b>tlmgr</b> manual very carefully before using it.



...	For <code>tlmgr()</code> , additional arguments to be passed to <code>system2()</code> (e.g., <code>stdout = TRUE</code> to capture stdout). For other functions, arguments to be passed to <code>tlmgr()</code> .
<code>.quiet</code>	Whether to hide the actual command before executing it.
<code>what</code>	A search keyword as a (Perl) regular expression.
<code>file</code>	Whether to treat <code>what</code> as a filename (pattern).
<code>all</code>	For <code>tlmgr_search()</code> , whether to search in everything, including package names, descriptions, and filenames. For <code>tlmgr_update()</code> , whether to update all installed packages.
<code>global</code>	Whether to search the online TeX Live Database or locally.
<code>word</code>	Whether to restrict the search of package names and descriptions to match only full words.
<code>pkgs</code>	A character vector of LaTeX package names.
<code>path</code>	Whether to run <code>tlmgr_path('add')</code> after installing packages ( <code>path = TRUE</code> is a conservative default: it is only necessary to do this after a binary package is installed, such as the <b>metafont</b> package, which contains the executable <code>mf</code> , but it does not hurt even if no binary packages were installed).
<code>self</code>	Whether to update the TeX Live Manager itself.
<code>more_args</code>	A character vector of more arguments to be passed to the command <code>tlmgr update</code> or <code>tlmgr conf</code> .
<code>run_fmtutil</code>	Whether to run <code>fmtutil-sys --all</code> to (re)create format and hyphenation files after updating <b>tlmgr</b> .
<code>delete_tlpdb</code>	Whether to delete the <code>'texlive.tlpdb.HASH'</code> files (where <code>HASH</code> is an MD5 hash) under the <code>'tlpkg'</code> directory of the root directory of TeX Live after updating.
<code>action</code>	On Unix, add/remove symlinks of binaries to/from the system's <code>PATH</code> . On Windows, add/remove the path to the TeXLive binary directory to/from the system environment variable <code>PATH</code> .

## Details

The `tlmgr()` function is a wrapper of `system2('tlmgr')`. All other `tlmgr_*` functions are based on `tlmgr` for specific tasks. Please consult the **tlmgr** manual for full details.

## References

The **tlmgr** manual: <https://www.tug.org/texlive/doc/tlmgr.html>

## Examples

```
# search for a package that contains titling.sty
tlmgr_search('titling.sty')

#' to match titling.sty exactly, add a slash before the keyword, e.g.
#' tlmgr_search('/titling.sty')

#' use a regular expression if you want to be more precise, e.g.
```

```
#' tlmgr_search('/titling\sty$')

# list all installed LaTeX packages
tlmgr(c('info', '--list', '--only-installed', '--data', 'name'))
```

---

tl_pkgs	<i>List the names of installed TeX Live packages</i>
---------	--

---

### Description

Calls `tlmgr info --list --only-installed --data name` to obtain the names of all installed TeX Live packages. Platform-specific strings in package names are removed, e.g., "tex" is returned for the package **tex.x86\_64-darwin**.

### Usage

```
tl_pkgs()
```

### Value

A character vector of package names.

---

tl_sizes	<i>Sizes of LaTeX packages in TeX Live</i>
----------	--

---

### Description

Use the command `tlmgr info --list --only-installed` to obtain the sizes of installed LaTeX packages.

### Usage

```
tl_sizes(show_total = TRUE)
```

### Arguments

`show_total` Whether to show the total size.

### Value

A data frame of three columns: `package` is the package names, `size` is the sizes in bytes, and `size_h` is the human-readable version of sizes.

# Index

copy\_tinytex, 2

install\_tinytex, 3

latexmk, 4

lualatex (latexmk), 4

parse\_install, 6

parse\_packages, 5, 6, 6

pdflatex (latexmk), 4

r\_texmf, 7

reinstall\_tinytex (install\_tinytex), 3

system2, 9

tinytex\_root (install\_tinytex), 3

tl\_pkgs, 10

tl\_sizes, 10

tlmgr, 8

tlmgr\_conf, 7

tlmgr\_conf (tlmgr), 8

tlmgr\_install, 6

tlmgr\_install (tlmgr), 8

tlmgr\_path (tlmgr), 8

tlmgr\_remove (tlmgr), 8

tlmgr\_search, 6

tlmgr\_search (tlmgr), 8

tlmgr\_update (tlmgr), 8

uninstall\_tinytex (install\_tinytex), 3

use\_tinytex, 5

use\_tinytex (copy\_tinytex), 2

xelatex (latexmk), 4