The leanify package

A LaTeX article checked by Lean

Marcus Rockel
https://github.com/Corrram/leanify

Version 0.3, 2026-11-06

Abstract

Leanify compiles an ordinary mathematical article, written in LaTeX, to the Lean proof assistant. Every definition and claim of the article becomes a Lean statement checked by the kernel, and the proofs are checked sentence by sentence against the results they cite. There is no second, hand-written formal version that could drift away from the paper: one source is typeset as the article you submit and compiled by Lean. This document describes the LaTeX package leanify, the typesetting side of Leanify. The Lean side, the grammar of the articles it reads and a tutorial are documented at https://corrram.github.io/leanify/.

1 What the package does

Leanify reads the article source itself: its sentences, formulas and the macros of its preamble. The package does not change what the article says, and Leanify reads no markup from it. The package

- loads the packages for the notation the Leanify grammar reads: `\coloneqq` from mathtools, `\mathbb` and `\le` from amssymb;
- checks the article while it compiles, when it can (Section 3);
- with the option `status`, marks every labelled result in the margin with its verification status (Section 4);
- records the Lean module that certifies the article (`\leanifymodule`).

Without options and without shell escape the typeset article is unchanged, so an article that loads leanify compiles anywhere, for instance on arXiv or a journal's server.

For example, Leanify compiles the sentence

```
Define $f:(0,\infty)\to\mathbb{R}$ by $f(x)\coloneqq x+\frac1x$.
```

to the Lean definition `f := fun x => x + 1 / x`, with the obligation that it is well defined, and the theorem

```
\begin{theorem}\label{thm:bound}
For every $x\in(0,\infty)$, $f(x)\ge2$.
\end{theorem}
```

to the Lean statement $\forall\,(x : \mathbb{R}),\ 0 < x \to 2 \le f\,x$, derived in Lean from exactly the sentences of its proof and the results they cite.

2 Usage

```
\usepackage{leanify}             % check as you compile
\usepackage[status]{leanify}     % and mark results in the margin
\usepackage[certify]{leanify}    % and build the certificate first
\leanifymodule{Note.Certified}   % optional: the certifying module
\leanifyreport                   % optional: a one-line summary
```

The options are:

Option	Effect
`status`	Marks every labelled result in the margin with its verification status, read from `\jobname.leanify.tex`.
`check=false`	No quick check during the compile.
`certify`	Before the marks are read, updates the pin of the article and builds its certificate with `lake build`. The first time, when the article has no Lean files yet, writes them, and a warning gives the `[[lean_lib]]` entry to add to `lakefile.toml`. Implies `status`.
`vocab=NAME`	The vocabulary of the article: `core` (the default) or `unitsquare` (probability on the unit square).
`lean=FILE`	The compiled Lean file of the article, by default `Compiled.lean` next to it.

The commands are:

`\leanifymodule{`*module*`}` The Lean module whose `#leanify_certify` certifies the article. By default it is `NS.Certified`, for the namespace `NS` that the compiled Lean file compiles.

`\leanifyreport` Prints a one-line summary of the last build where it appears, for instance: *12 of 12 claims verified (3 derived in the text from their citations); 85 of 90 sentences compiled; 7 definitions well defined.*

3 Checking as you compile

When the compiler may run programs (shell escape: `pdflatex -shell-escape`, `latexmk -shell-escape`, or the setting of the editor) and the article lies in a Lean project (a `lakefile.toml` or `lakefile.lean` in its folder or up to three folders above), every compile runs `lake exe leanify check` on the article. Its findings become LaTeX warnings at their lines of the article,

```
Package leanify Warning: sentence not compiled (undefined symbol: g); it
(leanify)                asserts nothing checked on input line 70.
```

so editors that read the LaTeX log list them with the other warnings. With `certify`, the errors of the Lean build and the refuted claims are warnings too. The full output of Leanify is in `\jobname.leanify.log`. `lake` must be on the `PATH` of the compiler; the first check also builds the checker, which takes a few minutes once.

Without shell escape nothing runs, and in a Lean project a warning says so. Outside a Lean project the package runs nothing and says nothing.

4 Verification status in the margin

With `status`, every labelled result whose status is known gets one of these marks in the margin:

Mark	Meaning
✓ checked	Every sentence compiled, and the claim is verified in Lean.
✓ derived	Every sentence compiled, and the proof in the text derives the claim from its citations.
○ partly checked	The result is verified in part.
○ compiled	The statement compiled, but is not verified yet.
× false	The claim is refuted in Lean.

The marks are read from `\jobname.leanify.tex`, which the Lean command `#leanify_certify` writes next to the article. If the article is newer than that file, a warning says that the marks may be out of date. Leave `status` out for the version you submit.

5 Requirements

The package needs LaTeX 2022-06-01 or newer, mathtools and amssymb. Checking and the status marks need the Lean toolchain and the Leanify Lean package in the article's Lake project; typesetting does not. Installation and a step-by-step tutorial are at https://corrram.github.io/leanify/.

6 Licence

Copyright © 2026 Marcus Rockel. The package is released under the Apache License, Version 2.0.