reptheorem[*]

Jesse Straat

2025-08-15

Abstract

When writing a large manuscript, it is sometimes beneficial to repeat a theorem (or lemma or . . .) at an earlier or later point for didactical purposes. However, `thmtools`'s built-in `restatable` only allows replicating theorems *after* they have been stated, and only in the same document. `reptheorem` solves the issue by making use of the `.aux` file, and also introduces its own file extension, `.thm`, to replicate theorems in other files.

Contents

1 Repeating theorems

Let's say we define a theorem as follows:

```
\begin{theorem}[Yoneda Lemma]
    For \(F\colon \mathcal{C}\to \mathbf{Set}\) a functor,
    \([\mathcal{C}^\mathrm{op},\mathbf{Set}](YA, F) \cong F(A)\)%
    for all objects \(A\) in \(\mathcal{C}\).
\end{theorem}
```

Its output is of course

Theorem 1 (Yoneda Lemma)**.** *For* $F\colon \mathcal{C}\to \mathbf{Set}$ *a functor,* $[\mathcal{C}^\mathrm{op},\mathbf{Set}](YA, F) \cong F(A)$ *for all objects* A *in* $\mathcal{C}$*.*

Now let's say we want to replicate the theorem within the same document. `makethm` (*env.*) That is what the new environment `makethm` is used for.

```
\begin{makethm}{theorem}{thm:Yoneda}[Yoneda Lemma]
    For \(F\colon \mathcal{C}\to \mathbf{Set}\) a functor,
```

[*] Version v1.4, last revised 2025-08-15.

```
    \([\mathcal{C}^\mathrm{op},\mathbf{Set}](YA, F) \cong F(A)\)
    for all objects \(A\) in \(\mathcal{C}\).
\end{makethm}
```

Its output is the same (in fact, we've secretly used `makethm` in the previous example), but the important difference is that we have saved the theorem for later use.

The `makethm` environment takes two mandatory arguments and one optional one. The first mandatory argument is the type of theorem environment as defined in amsthm, like `theorem`, `lemma`, `definition`, etc. The second is the theorem's label. The label is mandatory since, to replicate the theorem, we need to have a "name" attached to it. `makethm` automatically attaches a `\label`, as well, so `\ref{thm:Yoneda}` becomes 1. The optional argument is passed right to the optional argument of the theorem environment, giving the theorem a name.

`\repthm`

Now let's say we want to replicate the theorem later or earlier in the text. This may be done if, for example, the theorem is proven at a later point, or we want to "tease" the reader with a powerful theorem that will be proven later in the chapter. To do this, we use the `\repthm` command: `\repthm{thm:Yoneda}`. This outputs the theorem again.

Theorem 1 (Yoneda Lemma)**.** *For $F\colon\mathcal{C}\to\mathbf{Set}$ a functor, $[\mathcal{C}^{\mathrm{op}},\mathbf{Set}](YA,F)\cong F(A)$ for all objects A in $\mathcal{C}$.*

The label of this theorem is a `\ref`, and automatically links to the original theorem statement.

If the original theorem statement exists in a different file, or has not been created yet, we can add a placeholder alt text to the `\repthm` as an optional argument, which only displays if the theorem is undefined. For example, `\repthm{thm:foo}[bar]` returns

Theorem . *bar*

If we do the same without providing an alt text, we get

Theorem .

together with a warning: "Package reptheorem: Theorem `thm:foo` not defined; rebuild your project. If the issue persists, create the theorem using `\begin{makethm}` or consider adding alt text to `\repthm` using the optional parameter."

Since we're using the `.aux` file, it is possible to replicate a theorem before it is stated. For example,

```
\repthm{thm:later}
\begin{makethm}{theorem}{thm:later}
    Alligator!
\end{makethm}
```

returns

Theorem 2. *Alligator*

Theorem 2. *Alligator*

Note that it is necessary to run a `.tex` file twice to replicate theorems ahead of time, similarly to how one has to run a file twice to make sure the references are correct.

`\repthm*`

It is also possible to use a starred version, `\repthm*`. It then automatically adds a star to the end of the theorem environment. For example, `theorem` becomes `theorem*`.

2 Replicating theorems between files

Let’s say we have the following files for our project:

```
foo.tex
bar.tex
```

`\theoremfile`

`\loadtheorems`

Let’s say that we have defined a theorem `thm:baz` in `bar.tex`, and we want to replicate it in `foo.tex`. To achieve this, we first use the `\theoremfile` command in the preamble of `bar.tex`. This compiles all theorems defined in `bar.tex` and outputs them into a file `bar.thm`. To then import these into `foo.tex`, we use `\loadtheorems{bar.thm}` in the preamble, which loads all theorems saved in `bar.thm`. One can then use `\repthm` as usual.

Since the `.aux` file is loaded at `\begin{document}`, putting `\loadtheorems` in the preamble of a file will guarantee that the loaded theorem file will be overwritten by the theorems in the `.aux` file, i.e., theorems defined in the same document. In our example, if we also defined a `thm:baz` in `foo.tex`, loading `bar.thm` into `foo.tex` will not overwrite the local `thm:baz`.

2.1 Replicating theorems to subfiles

Replicating theorems to different files is particularly useful when working in big documents with multiple subfiles. For example, let’s say we have the files

```
main.tex
foo.tex
bar.tex
```

Here, `main.tex` is generated by including `foo.tex` and `bar.tex` as chapters, creating a single large document. It is now possible to replicate theorems within the subfiles by running `\theoremfile` in `main.tex`, and then using `\loadtheorems{main.thm}` in `foo.tex` and `bar.tex`. This will allow us to use all theorems in the final `main.tex` in each of the subfiles.

3 Source code

```
⟨*package⟩
\ProvidesPackage{reptheorem}[2025-08-15 v1.4 Reptheorem package]
```

`\theoremfile`

Using `\theoremfile` will output all saved theorems into an output file. By default, if your LaTeXfile is `foo.tex`, the output file is `foo.thm`.

```
\def\reptheorem@theoremfile{\relax}
```

```
\NewDocumentCommand{\theoremfile}{ O{\jobname.thm} }{
 % O: the path of the file to which we should save theorems
 %
   \def\reptheorem@theoremfile{#1}
 \newwrite\@thmlist
 \immediate\openout\@thmlist=#1
}
```

\loadtheorems If you have exported saved theorems to a file, you can load them into another file using the macro \loadtheorems.

```
\NewDocumentCommand{\loadtheorems}{ m }{
 \IfFileExists{#1}{
  \makeatletter
  \input{#1}
  \makeatother
 }{
  \PackageWarning{reptheorem}{%
   File #1 not found. I will not import any theorems.%
  }
 }
}
```

The \makeatletter is included here to assure that any macros that are expanded into macros that contain an @ are interpreted correctly.

makethm (*env.*) On to defining the actual theorems to be saved.

```
\NewDocumentEnvironment{makethm}{ m m o +b }
 % m: the type of theorem environment
 % m: the name of the theorem
 % o: optional parameter for environment
 % b: the content of the theorem
 %
 {%
  \IfValueTF{#3}{% Check if theorem has optional arguments
   \begin{#1}[#3]\label{#2}
  }{
   \begin{#1}\label{#2}
  }
  % \begin{theorem}
   #4
   \providecommand{\label}[1]{}
   \expandafter\gdef\csname thmtype@#2\endcsname{#1}%
   \expandafter\long\expandafter\gdef\csname thm@#2\endcsname{#4}%
   \IfValueT{#3}{% Only save theorem name if it exists
    \expandafter\gdef\csname thmdesc@#2\endcsname{#3}%
   }
   % Saving parameters to aux file
   \expandafter\long\expandafter\gdef\csname thmoutput@#2\endcsname{%
    \string\expandafter\string\gdef\noexpand%
    \csname thmtype@#2\string\endcsname{#1}%
    ^^J%
    \string\expandafter\string\long\string\expandafter%
    \string\gdef\noexpand\csname thm@#2\string\endcsname{#4}%
    \IfValueT{#3}{%
    ^^J%
    \string\expandafter\string\gdef\noexpand%
    \csname thmdesc@#2\string\endcsname{#3}%
    }%
    ^^J%
    \string\expandafter\string\gdef\noexpand%
    \csname thmlabel@#2\string\endcsname{\getrefnumber{#2}}%
   }
   \write\@auxout{\csname thmoutput@#2\endcsname}
   \if\reptheorem@theoremfile\relax
    % No file has been set
   \else
    % We have a theorem file
    % Saving parameters to theorem file
    \write\@thmlist{\csname thmoutput@#2\endcsname}
   \fi
  \end{#1}
 }{}
```

\repthm To repeat a theorem, use the \repthm command.

If the theorem type shares its counter with another theorem type, e.g., lemma having the same counter as thoerem, make sure you have thmtools imported. Its \@counteralias macro is essential for the counters to work.

```
\newcounter{old@counter}
\NewDocumentCommand{\repthm}{ s m +o }{
 % s: optional star to add to theorem environment
 % m: the name of the theorem
 % o: alt text
 \begingroup
  % Check if thmtype is given
  \ifcsname thmtype@#2\endcsname%
   \expandafter\let\expandafter\@@thmtype\csname thmtype@#2\endcsname%
  \else%
   \def\@@thmtype{theorem}%
   \PackageWarning{reptheorem}{%
    Theorem ’#2’ has unknown theorem type. Assuming it is of
    type ’theorem’.%
   }
  \fi%
  \edef\@@thmcounter{\@@thmtype}
  \IfBooleanT{#1}{\edef\@@thmtype{\@@thmtype*}}
  %
  % Save theorem counter so we don’t increase it
  \ifcsname c@\@@thmcounter\endcsname
  \else
   \PackageWarning{reptheorem}{%
    Counter ’\@@thmcounter’ not defined; if theorem
    ’\@@thmcounter’ shares its counter with another
    theorem, make sure thmtools is imported.%
   }
  \fi
  \setcounter{old@counter}{\value{\@@thmcounter}}
  \setcounter{\@@thmcounter}{-900}
  %
  % Set label number
  \ifcsname r@#2\endcsname
   % Reference exists: set number as reference
   \expandafter\def\csname the\@@thmtype\endcsname{\ref{#2}}
  \else
   % Force label number as saved
   \expandafter\def\csname the\@@thmtype\endcsname{\csname thmlabel@#2\endcsname}
  \fi
  %
  \let\@@theoremnotdefined\relax
  %
  \ifcsname thm@#2\endcsname% Check if theorem is even defined
   % Theorem is defined
   \expandafter\let\expandafter\@@thm\csname thm@#2\endcsname
   % Output theorem
   \ifcsname thmdesc@#2\endcsname % Check if theorem has name
    \begin{\@@thmtype}[\csname thmdesc@#2\endcsname]
     \@@thm
    \end{\@@thmtype}
   \else % No optionals
    \begin{\@@thmtype}
     \@@thm
    \end{\@@thmtype}
   \fi
  \else
   % Theorem undefined
   \IfValueTF{#3}{
    \begin{\@@thmtype}
     #3
    \end{\@@thmtype}
   }{% No theorem or alt text provided: throw warning
    \begin{\@@thmtype}
    \end{\@@thmtype}
    \PackageWarning{reptheorem}{%
     Theorem '#2' not defined; rebuild your project.
     If the issue persists, create the theorem using
     \begin{makethm} or consider adding alt text to \repthm
     using the optional parameter.%
    }
   }
  \fi
  \setcounter{\@@thmcounter}{\value{old@counter}}
  % Reset theorem counter back to original
    \endgroup
}
```

⟨/package⟩

Change History

v1.0

General: First public release 1

v1.1

`makethm`: Now saves theorem

environment type, breaking backwards compatibility. 4

`\repthm`: Now saves theorem environment type, breaking backwards compatibility. 5

v1.2

`makethm`: Environment end moved to fix vertical spacing. 4

Renamed theorem output variable to be unique for each theorem. 4

Theorem name is only saved if it exists. 4

`\repthm`: Fixed bug where theorems got a name even if undefined. 5

v1.3

`\repthm`: Added hyperref named destination compatibility by setting counter to very low value. 5

Changed thetheorem to csname to fix compatibility with theorem types not called "theorem". 5

v1.4

`\loadtheorems`: Now makes @ catcode 11 to fix incompatibility. 4

`makethm`: Added theorem label to aux file. 4

`\repthm`: Added warnings for unknown counter and unknown theorem type. 5

If reference doesn't exist, saved label is now used instead of ??. Added star option. 5

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.