

reptheorem*

Jesse Straat

2026-08-11

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	1
2 Replicating theorems between files	3
2.1 Replicating theorems to subfiles	3
3 Source code	3

1 Repeating theorems

Let's say we define a theorem as follows:

```
\begin{theorem}[Yoneda Lemma]
  For  $(F\colon \mathcal{C}^{\mathrm{op}}\to \mathbf{Set})$  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 $F\colon \mathcal{C}^{\mathrm{op}} \rightarrow \mathbf{Set}$ 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  $(F\colon \mathcal{C}^{\mathrm{op}}\to \mathbf{Set})$  a functor,
```

*Version v1.5, last revised 2026-08-11.

```

\([\mathcal{C}^{\mathrm{op}}, \mathbf{Set}](YA, F) \cong F(A)\)
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.

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 $F: \mathcal{C}^{\mathrm{op}} \rightarrow \mathbf{Set}$ 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 `repthm`: 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, `\repthm*{thm:later}` returns

Theorem. *Alligator!*

2 Replicating theorems between files

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

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

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` `\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

```
1 (*package)
2 \ProvidesPackage{reptheorem}[2026-08-11 v1.5 Reptheorem package]

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

```

3 \def\reptheorem@theoremfile{\relax}
4 \NewDocumentCommand{\theoremfile}{0}{\jobname.thm} }{
5 % 0: the path of the file to which we should save theorems
6 %
7 \def\reptheorem@theoremfile{#1}
8 \newwrite\@thmlist
9 \immediate\openout\@thmlist=#1
10 }

```

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

```

11 \NewDocumentCommand{\loadtheorems}{m}{
12 \IfFileExists{#1}{
13 \makeatletter
14 \input{#1}
15 \makeatother
16 }{
17 \PackageWarning{reptheorem}{%
18 File #1 not found. I will not import any theorems%
19 }
20 }
21 }

```

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

`\repthm@firstoffive` This returns the first of five arguments. It is used to get the theorem number of a label.

```

22 \NewExpandableDocumentCommand{\repthm@firstoffive}{m m m m m}{%
23 #1%
24 }

```

`\repthm@ifrefexists` This command checks whether a label was defined in the auxiliary file. Using just `\ifcsname r@foo\endcsname` is not sufficient, since `\ref{foo}` defines `\r@foo` using a placeholder variable if the reference does not exist.

```

25 \NewExpandableDocumentCommand{\repthm@ifrefexists}{m +m +m}{%
26 % m: the label
27 % m: code to run if true
28 % m: code to run if false
29 \ifcsname r@#1\endcsname
30 % reference exists but might be dummy
31 \expandafter\ifx\csname r@#1\endcsname\relax
32 % reference is a dummy
33 #3%
34 \else
35 % reference exists and is not a dummy
36 #2%
37 \fi
38 \else
39 % reference doesn't exist
40 #3%
41 \fi
42 }

```

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

```

43 \ExplSyntaxOn
44 \tl_new:N \g_reptheorem_thminput_tl
45 \NewDocumentEnvironment{makethm}{ m m o +b }
46 % m: the type of theorem environment
47 % m: the name of the theorem
48 % o: optional parameter for environment
49 % b: the content of the theorem
50 %
51 {%
52   \IfValueTF{#3}{% Check if theorem has optional arguments
53     \begin{#1}[#3]\label{#2}
54   }{
55     \begin{#1}\label{#2}
56   }
57   % \begin{theorem}
58   #4

```

Saving labels inside theorems leads to all sorts of issues (multiply defining labels, the .aux file breaking), so we should filter those out using regex.

```

59   \tl_set:Nn \l_tmpa_tl {#4}
60   \regex_replace_all:nnN { \c{label}}{\{[^\}]*\} } { } \l_tmpa_tl
61   \tl_gset_eq:NN \g_reptheorem_thminput_tl \l_tmpa_tl
62   \expandafter\gdef\csname thmtype@#2\endcsname{#1}%
63   \expandafter\long\expandafter\gdef\csname thm@#2\endcsname{
64     \tl_use:N \g_reptheorem_thminput_tl
65   }%
66   \IfValueT{#3}{% Only save theorem name if it exists
67     \expandafter\gdef\csname thmdesc@#2\endcsname{#3}%
68   }
69   % Saving parameters to aux file
70   \expandafter\long\expandafter\gdef\csname thmoutput@#2\endcsname{%
71     \string\expandafter\string\gdef\noexpand%
72     \csname thmtype@#2\string\endcsname{#1}%
73     ^^J%
74     \string\expandafter\string\long\string\expandafter%
75     \string\gdef\noexpand\csname thm@#2\string\endcsname{
76       \tl_use:N \g_reptheorem_thminput_tl
77     }%
78     \IfValueT{#3}{%
79       ^^J%
80       \string\expandafter\string\gdef\noexpand%
81       \csname thmdesc@#2\string\endcsname{#3}%
82     }%
83     ^^J%
84     \string\expandafter\string\gdef\noexpand%
85     \csname thmlabel@#2\string\endcsname{%
86       \repthm@ifrefexists{#2}{%
87         \expandafter\repthm@firstoffive\expanded{\csname r@#2\endcsname}%
88       }{ }%
89     }%
90   }
91   \write\@auxout{\csname thmoutput@#2\endcsname}
92   \if\reptheorem@theoremfile\relax

```

```

93 % No file has been set
94 \else
95 % We have a theorem file
96 % Saving parameters to theorem file
97 \write\thmlist{\csname thmoutput@#2\endcsname}
98 \fi
99 \end{#1}
100 }{}
101 \ExplSyntaxOff

```

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

```

102 \newcounter{old@counter}
103 \NewDocumentCommand{\repthm}{ s m +o }{
104 % s: optional star to add to theorem environment
105 % m: the name of the theorem
106 % o: alt text
107 \begingroup
108 % Check if thmtype is given
109 \ifcsname thmtype@#2\endcsname%
110 \expandafter\let\expandafter\@@thmtype\csname thmtype@#2\endcsname%
111 \else%
112 \def\@@thmtype{theorem}%
113 \PackageWarning{reptheorem}{%
114 Theorem '2' has unknown theorem type. Assuming it is of
115 type 'theorem'. If you just changed the theorem's type,
116 rerun the file%
117 }
118 \fi%
119 \edef\@@thmcounter{\@@thmtype}
120 \IfBooleanT{#1}{\edef\@@thmtype{\@@thmtype*}}
121 %
122 % Save theorem counter so we don't increase it
123 \ifcsname c@\@@thmcounter\endcsname
124 \else
125 \PackageWarning{reptheorem}{%
126 Counter '\@@thmcounter' not defined. Reverting to counter
127 'theorem'. If you just changed the theorem's type,
128 rerun the file%
129 }
130 \def\@@thmcounter{theorem}
131 \fi
132 \setcounter{old@counter}{\value{\@@thmcounter}}
133 \setcounter{\@@thmcounter}{-900}
134 %
135 % Set label number
136 \repthm@ifrefexists{#2}{%
137 % Reference exists: set number as reference
138 \expandafter\renewcommand\csname the\@@thmcounter\endcsname{\ref{#2}}
139 }{%
140 % Force label number as saved
141 \ifcsname thmlabel@#2\endcsname
142 \expandafter\renewcommand\csname the\@@thmcounter\endcsname{%
143 \csname thmlabel@#2\endcsname%
144 }

```

```

145 \else
146 % No label number saved: revert to ??
147 \expandafter\renewcommand\csname the\@thmcounter\endcsname{\ref{#2}}
148 \fi
149 }
150 %
151 \ifcsname thm@#2\endcsname% Check if theorem is even defined
152 % Theorem is defined
153 \expandafter\let\expandafter\@thm\csname thm@#2\endcsname
154 % Output theorem
155 \ifcsname thmdesc@#2\endcsname % Check if theorem has name
156 \begin{\@thmtype}[\csname thmdesc@#2\endcsname]
157 \@thm
158 \end{\@thmtype}
159 \else % No optionals
160 \begin{\@thmtype}
161 \@thm
162 \end{\@thmtype}
163 \fi
164 \else
165 % Theorem undefined
166 \IfValueTF{#3}{
167 \begin{\@thmtype}
168 #3
169 \end{\@thmtype}
170 }{% No theorem or alt text provided: throw warning
171 \begin{\@thmtype}
172 \end{\@thmtype}
173 \PackageWarning{repthm}{%
174 Theorem '#2' not defined; rebuild your project.
175 If the issue persists, create the theorem using
176 \begin{makethm} or consider adding alt text to \repthm
177 using the optional parameter%
178 }
179 }
180 \fi
181 \setcounter{\@thmcounter}{\value{old@counter}}
182 % Reset theorem counter back to original
183 \endgroup
184 }
185 \end{package}

```

Change History

v1.0		environment type, breaking
General: First public release	1	backwards compatibility. 5
v1.1		v1.2
makethm: Now saves theorem		makethm: Environment end moved
environment type, breaking		to fix vertical spacing. 4
backwards compatibility.	4	Renamed theorem output
\repthm: Now saves theorem		variable to be unique for each

theorem.	4	If reference doesn't exist, saved label is now used instead of ??.	
Theorem name is only saved if it exists.	4	Added star option.	5
<code>\repthm</code> : Fixed bug where theorems got a name even if undefined.	5		
v1.3		v1.4.1	
<code>\repthm</code> : Added hyperref named destination compatibility by setting counter to very low value.	5	<code>\makethm</code> : Replaced theorem label saving macro.	4
Changed thetheorem to csname to fix compatibility with theorem types not called "theorem".	5	<code>\repthm</code> : Added fallback for when no label is saved.	5
v1.4		<code>\repthm@firstoffive</code> : Added firstoffive command	4
<code>\loadtheorems</code> : Now makes @catcode 11 to fix incompatibility.	4	<code>\repthm@ifrefexists</code> : Added ifrefexists command	4
<code>\makethm</code> : Added theorem label to aux file.	4	v1.5	
<code>\repthm</code> : Added warnings for unknown counter and unknown theorem type.	5	<code>\makethm</code> : Labels are now disabled in saved theorem.	4
		<code>\repthm</code> : Changed defs to renewcommands in reference for beamer compatibility.	5
		Improved warnings (instructing a rerun), added counter backup, and replaced some thmtypes with thmcounters. . .	5
		Removed reference to thmtools since counter aliases are now part of L ^A T _E X.	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.

Symbols	<code>\ExplSyntaxOn</code>	43	R
<code>\@@thm</code>	L		<code>\reptheorem@theoremfile</code>
<code>\@thmcounter</code>	<code>\loadtheorems</code>	3, <u>11</u>	
.	M		<code>\repthm</code>
119, 123, 126, 130, 132, 133, 138, 142, 147, 181	<code>\makethm (env.)</code>	1, <u>43</u>	<code>\repthm*</code>
<code>\@thmtype</code> 110, 112, 119, 120, 156, 158, 160, 162, 167, 169, 171, 172	N		<code>\repthm@firstoffive</code>
<code>\@thmlist</code>	<code>\newwrite</code>	8	
	O		<code>\repthm@ifrefexists</code>
E	<code>\openout</code>	9	
environments:	P		T
<code>\makethm</code>	<code>\PackageWarning</code> . . .		<code>\theoremfile</code>
<code>\ExplSyntaxOff</code>		17, 113, 125, 173	<u>3</u> , 3