The naproche package*

Marcel Schütz

September 14, 2025

This `naproche` package[1] provides markup for ℕaproche formalizations written in the LaTeX dialect of ForTheL. It is intended to be a "beginner-friendly" LaTeX package (i.e. one that can be easily adapted to experiment with the typesetting it provides) to be used for small Naproche formalizations that

- do *not* depend on libraries of ℕaproche formalizations and
- are *not* intended to be converted to interactive HTML.

For formalizations that require any of the above features, consider to use the "advanced" `naproche` package (see footnote 1) instead.

Contents

*This document corresponds to naproche v1.0.0, dated 2025-09-05.

[1]There is another `naproche` package for more advanced usecases, shipped with ℕaproche at `math/latex/lib/naproche.sty`

Part I
Documentation

1 Usage

To use the `naproche` package (without installing it), write

```
\usepackage{path/to/naproche}
```

in the preamble of the `.ftl.tex` file of your Naproche formalization, where `path/to/naproche` is the path to the `.sty` file of this package. See the `.ftl.tex` files in the `math/examples` directory for examples.

2 Package Options

`puzzle` This package provides two typesetting styles: A "standard" and a "puzzle" style, where – as the name suggests – the "standard" style is the default one. To use the "puzzle" style, provide the package option `puzzle`. The "standard" style is used by all formalizations in the `math/examples` directory; the "puzzle" style is used by, e.g., `math/examples/puzzles/agatha.ftl.tex` and `math/examples/puzzles/dwarfs.ftl.tex`. The following sections contain notes on how they differ from each other.

`numberswithinsection` `numberswithinsubsection` If the option `numberswithinsection` or `numberswithinsubsection` is given, the numbering of top-level sections (i.e. definitions, theorems, axioms etc.) is reset at every new section or subsection, resp.

3 Commands Required by ForTheL

The syntax of ForTheL specifies keywords that have the shape of TEX macros. This package loads certain LATEX packages that provide some of those macros and provides its own definitions for the remaining ones:

`\dom` The domain of a map, rendered as "dom".

`\fun` The λ-operator for locally defined maps, rendered as "λ".

4 The Naproche Logo

`\Naproche` Prints the Naproche logo, i.e. "Naproche". Note that it is *not* required to append an empty pair of curly braces to `\Naproche` to get its trailing whitespace right. E.g. `\Naproche foo` will be rendered as "Naproche foo" and not (as you might expect) as "Naprochefoo".

5 Comprehension Terms

`\class` `\class{`⟨*lhs*⟩`|`⟨*rhs*⟩`}`

Prints a comprehension term: "{⟨*lhs*⟩ | ⟨*rhs*⟩}".

`\classtext` `\classtext{`⟨*arg*⟩`}`

When using the `\class` command to print a comprehension term it sometimes happens that its right-hand side requires more space than is left in the current line. In this situation the right-hand side can be wrapped in the `\classtext` command which behaves much like the `\text` command (i.e. leaves math mode[2]) but automatically insert linebreaks at appropriate positions. Thus, when using the `\classtext` command, the containing `\class` command should only be used in *display* math mode. For example,

```
\[
\class{n \in X | \classtext{$n$ is an odd prime number that is greater than
$5$ and divides every number that is contained in $Y$}}
\]
```

is rendered as:

$$\left\{ n \in X \;\middle|\; \begin{array}{l} n \text{ is an odd prime number that is greater than 5 and divides every} \\ \text{number that is contained in } Y \end{array} \right\}$$

6 Printing Labels

`\printref` `\printref{`⟨*id*⟩`}`

Prints the label ⟨*id*⟩ of a top-level section verbatimly. This can be usefull to print labels of top-level sections that are defined in an imported file (and therefore not accessible for LaTeX).

For example, assume that we have a file `foo.ftl.tex` which contains the following theorem environment:

```
\begin{forthel}
  \begin{theorem}\label{cantor}
    There exists no surjection from $X$ to the powerset of $X$.
  \end{theorem}
\end{forthel}
```

Moreover, assume that we have another file `bar.ftl.tex` which imports `foo.ftl.tex` (e.g. via `[readtex \path{foo.ftl.tex}]`) and in which we want to reference the above theorem. Then LaTeX would complain if we would try to use, e.g.,

```
$|X|$ is strictly smaller than $|2^{X}|$ (by \ref{cantor}).
```

[2] In most cases an overlong right-hand side of a comprehension term is due to a long piece of text anyway, and not due to a long formula.

in a ForTheL proof since the label `cantor` has not been defined in the file `bar.ftl.tex`. To reference that theorem anyway we can use

```
$|X|$ is strictly smaller than $|2^{X}|$ (by \printref{cantor}).
```

instead, which is rendered as "$|X|$ is strictly smaller than $|2^X|$ (by `cantor`).".

7 The ForTheL Environment

`forthel` (*env.*) Content wrapped in a `forthel` environment is recognized by ℕaproche which means that ℕaproche tries to check whether it is a formally correct piece of mathematical text.

It is displayed with a gray background to distinguish its *formal* content from the other *informal* parts of the document it is contained in. Moreover, paragraphs within a `forthel` environment are not indented and are separated vertically from each other by 0.5 em.

8 Top-level Section Environments

`signature` (*env.*)
`signature*` (*env.*)
`axiom` (*env.*)
`axiom*` (*env.*)
`definition` (*env.*)
`definition*` (*env.*)
`theorem` (*env.*)
`theorem*` (*env.*)
`lemma` (*env.*)
`lemma*` (*env.*)
`proposition` (*env.*)
`proposition*` (*env.*)
`corollary` (*env.*)
`corollary*` (*env.*)
`convention` (*env.*)
`convention*` (*env.*)

Top-level sections, i.e. signature extensions, axioms, definitions, assertions or conventions, can be typset via the respective environments listed in figure 1.

Top-Level Section	Environments	Name in the "standard" typesetting
signature extension	`signature` `signature*`	Signature
axiom	`axiom` `axiom*`	Axiom
definition	`definition` `definition*`	Definition

assertion	`theorem`
	`theorem*`
	`lemma`
	`lemma*`
	`proposition`
	`proposition*`
	`corollary`
	`corollary*`
		Theorem
		
		Lemma
		
		Proposition
		
		Corollary

convention	`convention` `convention*`	Convention

Figure 1: Top-Level Section Environments

8.1 Top-Level Sections in the "Standard" Typesetting

In the "standard" typesetting, top-level section environments take an optional argument ⟨*title*⟩ which is intended to contain the title of the respective top-level section. They are rendered as "theorem-like" sections, whose heading consists of the name of the top-level section (see figure 1), followed by an automatically generated number (only in the unstarred variants), the title (if given via ⟨*title*⟩) and a punctuation mark to separate the heading from the body of the section. Unstarred top-level section environments or top-level section environments with a ⟨*title*⟩ argument may be annotated with `\label{`⟨*id*⟩`}` which allows to reference them via `\ref` or `\nameref`. For instance,

```
\begin{theorem}[Cantor's Theorem]\label{cantor}
  There is no surjection from $X$ to the powerset of $X$.
\end{theorem}
```

is rendered as:

Theorem 1 (Cantor's Theorem). There is no surjection from X to the powerset of X.

To suppress the numbering, we can replace `theorem` by `theorem*` which causes the environment to be rendered as:

Theorem (Cantor's Theorem). There is no surjection from X to the powerset of X.

Since in the first case the environment is both starred and has a label, we can reference it via `\ref`. For example,

```
$|X|$ is strictly smaller than $|2^{X}|$ (by \ref{cantor}).
```

is rendered as "$|X|$ is strictly smaller than $|2^X|$ (by 1).". Alternatively, since it also has a title, we can reference it via `\nameref`. For example,

```
$|X|$ is strictly smaller than $|2^{X}|$ (by \nameref{cantor}).
```

is rendered as "$|X|$ is strictly smaller than $|2^X|$ (by Cantor's Theorem)." instead.

8.2 Top-Level Sections in the "Puzzle" Typesetting

In the "puzzle" typesetting, top-level section environments take no argument and just print their body verbatimly. In case of an assertion, an additional "Therefore:" is prepended to the body. Note that in the "puzzle" typesetting, using the starred or the unstarred variant of a top-level section environment has the same effect.

For instance[3],

```
\begin{axiom}
```

[3] Taken from `math/examples/puzzles/dwarfs.ftl.tex`.

```
  If some dwarf $D$ names the color of the hat of $D$ then all dwarfs
  get released.
\end{axiom}
\begin{axiom}
  Sigbert names the opposite color of the color of the hat of Tormund.
\end{axiom}
\begin{axiom}
  Tormund names the color of the hat of Sigbert.
\end{axiom}
\begin{theorem}
  All dwarfs get released.
\end{theorem}
```

is rendered as:

"If some dwarf D names the color of the hat of D then all dwarfs get released. Sigbert names the opposite color of the color of the hat of Tormund. Tormund names the color of the hat of Sigbert. Therefore: All dwarfs get released."

9 The Proof Environment

`proof` (*env.*) Proofs can be typeset via the `proof` environment which takes an optional argument ⟨*method*⟩. It is rendered as a "proof-like" section whose heading consists of the string "Proof", followed by ⟨*method*⟩ (if given) and a punctuation mark to separate the heading from the body of the section. Moreover, a QED symbol is added to the end of the body.

For example,

```
\begin{proof}[by induction]
  Let $n$ be a natural number.
  Then $n + 1 = 1 + n$.
  Hence ...
\end{proof}
```

is rendered as:

Proof by induction. Let n be a natural number. Then $n+1 = 1+n$. Hence ... □

Part II
Implementation

This section is only relevant for *developers* of the `naproche` package. If you are an *end-user* of this package, you can ignore this section.

10 Preliminary Setup

`naproche@forthel` A Boolean variable that tracks iff we are currently in a `forthel` environment.

```
\newbool{naproche@forthel}
```

`naproche@puzzle` A Boolean variable whose value determines if the "puzzle" typesetting is to be used instead of the "standard" typesetting.

```
\newbool{naproche@puzzle}
```

`naproche@tlscounter` A counter for numbered top-level section environments.

```
\newcounter{naproche@tlscounter}
```

11 Options

`puzzle` See section 2

```
\DeclareOption{puzzle}{
  \setbool{naproche@puzzle}{true}
}
```

`numberswithinsection` See section 2

```
\DeclareOption{numberswithinsection}{
  \counterwithin{naproche@tlscounter}{section}
}
```

`numberswithinsubsection` See section 2

```
\DeclareOption{numberswithinsubsection}{
  \counterwithin{naproche@tlscounter}{subsection}
}
```

Throw a warning for any unknown option.

```
\DeclareOption*{
  \PackageWarning{naproche}{Unknown option `\CurrentOption'}
}
```

Process the options.

```
\ProcessOptions\relax
```

12 Commands Required by ForTheL

`\dom` See section 3

```
\NewDocumentCommand{\dom}{}{\textrm{dom}}
```

`\fun` See section 3

```
\NewDocumentCommand{\fun}{}{\mathrm{\lambda}}
```

13 The ℕaproche Logo

`\Naproche` See section 4

```
\NewDocumentCommand{\Naproche}{}{\mbox{\ensuremath{\mathbb{N}}aproche}\xspace}
```

14 Comprehension Terms

`\naproche@lproj` Left and right projection.
`\naproche@rproj`

```
\NewDocumentCommand{\naproche@lproj}{m m}{#1}
\NewDocumentCommand{\naproche@rproj}{m m}{#2}
```

`\class` See section 5

```
\NewDocumentCommand{\class}{>{\SplitArgument{1}{|}} m}{
  \left\{\naproche@lproj#1~\middle\vert~\naproche@rproj#1\right\}
}
```

`\classtext` See section 5

```
\NewDocumentCommand{\classtext}{m}{\parbox{\linegoal}{#1}}
```

15 Printing Labels

`\printref` See section 6

```
\NewDocumentCommand{\printref}{m}{\path{#1}}
```

16 The ForTheL Environment

`naproche@forthelgray` Background color of `forthel` environments.

```
\colorlet{naproche@forthelgray}{lightgray!30}
```

`forthel` (*env.*) See section 7.

```
\NewDocumentEnvironment{forthel}{}{
  \begin{mdframed}[backgroundcolor=naproche@forthelgray,linecolor=naproche@forthelgray]
  \setbool{naproche@forthel}{true}
  \setlength{\parindent}{0pt}
  \setlength{\parskip}{0.5em}
}{
  \end{mdframed}
  \setbool{naproche@forthel}{false}
}
```

17 Top-level Section Environments

`\naproche@SetCurrentLabelName` Redefine the `\@currentlabelname` macro. (See https://ctan.net/macros/latex/contrib/hyperref/doc/nameref.pdf) for details.

```
\NewDocumentCommand{\naproche@SetCurrentLabelName}{m}{
  \protected@edef\@currentlabelname{#1}
}
```

`\naproche@NewTLS@standard` `\naproche@NewTLS@standard{`⟨*env*⟩`}{`⟨*name*⟩`}` defines two environments ⟨*env*⟩ and ⟨*env*⟩`*` (which both take an optional argument ⟨*title*⟩) which do the following:

1. Only applies to the environment ⟨*env*⟩: The counter `naproche@tlscounter` (whose value is referred to as ⟨*tls*⟩ in the following) is increased.
2. Enter a new paragraph and if `naproche@forthel` is `false` (i.e. if we are currently not in a `forthel` environment), insert a horizontal space of 0.5 em. (Inside a `forthel` environment this is not necessary since there all paragraphs are separated by 0.5 em anyway.)
3. In case of ⟨*env*⟩ print "⟨***name***⟩ ⟨***tls***⟩**.**" or, if ⟨*title*⟩ is given, "⟨***name***⟩ ⟨***tls***⟩ **(**⟨***title***⟩**).**" without indentation. In case of ⟨*env*⟩`*` print "⟨***name***⟩**.**" or, if ⟨*title*⟩ is given, "⟨***name***⟩ **(**⟨***title***⟩**).**" without indentation.
4. Print the body of the environment.
5. Enter a new paragraph and if `naproche@forthel` is `false` (i.e. if we are currently not in a `forthel` environment), a horizontal space of 0.5 em is inserted (cf. step 2).

```
\NewDocumentCommand{\naproche@NewTLS@standard}{m m}{
  \NewDocumentEnvironment{#1}{o}{
    \refstepcounter{naproche@tlscounter}
    \par
    \ifbool{naproche@forthel}{}{\vspace{0.5em}}
    \noindent\textbf{#2~\thenaproche@tlscounter\IfValueT{##1}{~(##1)}.}%
    \IfValueT{##1}{\naproche@SetCurrentLabelName{##1}}%
  }{
    \par
    \ifbool{naproche@forthel}{}{\vspace{0.5em}}
  }
  \NewDocumentEnvironment{#1*}{o}{
```

```
    \par
    \ifbool{naproche@forthel}{}{\vspace{0.5em}}
    \noindent\textbf{#2\IfValueT{##1}{~(##1)}.}%
    \IfValueT{##1}{\naproche@SetCurrentLabelName{##1}}%
  }{
    \par
    \ifbool{naproche@forthel}{}{\vspace{0.5em}}
  }
}
```

`\naproche@NewTLS@puzzle` `\naproche@NewTLS@puzzle{`⟨*env*⟩`}[`⟨*prefix*⟩`]` defines two environments ⟨*env*⟩ and ⟨*env*⟩`*` (which both take an optional argument) which just print their bodies. In case the optional argument ⟨*prefix*⟩ is given, it is prepended to the body.

```
\NewDocumentCommand{\naproche@NewTLS@puzzle}{m o}{
  \NewDocumentEnvironment{#1}{}{\unskip\IfValueT{#2}{#2}}{\unskip}
  \NewDocumentEnvironment{#1*}{}{\unskip\IfValueT{#2}{#2}}{\unskip}
}
```

`signature` (*env.*)
`axiom` (*env.*)
`definition` (*env.*)
`theorem` (*env.*)
`lemma` (*env.*)
`proposition` (*env.*)
`corollary` (*env.*)
`convention` (*env.*)

Top-level section environments (see section 8). If `naproche@puzzle` is `true`, define them via `\naproche@NewTLS@puzzle`, otherwise via `\naproche@NewTLS@standard`.

```
\ifbool{naproche@puzzle}{
  \naproche@NewTLS@puzzle{signature}
  \naproche@NewTLS@puzzle{axiom}
  \naproche@NewTLS@puzzle{definition}
  \naproche@NewTLS@puzzle{theorem}[Therefore:]
  \naproche@NewTLS@puzzle{lemma}[Therefore:]
  \naproche@NewTLS@puzzle{proposition}[Therefore:]
  \naproche@NewTLS@puzzle{corollary}[Therefore:]
  \naproche@NewTLS@puzzle{convention}
}{
  \naproche@NewTLS@standard{signature}{Signature}
  \naproche@NewTLS@standard{definition}{Definition}
  \naproche@NewTLS@standard{axiom}{Axiom}
  \naproche@NewTLS@standard{theorem}{Theorem}
  \naproche@NewTLS@standard{lemma}{Lemma}
  \naproche@NewTLS@standard{proposition}{Proposition}
  \naproche@NewTLS@standard{corollary}{Corollary}
  \naproche@NewTLS@standard{convention}{Convention}
}
```

18 The Proof Environment

`proof` (*env.*) See section 9.

```
\NewDocumentEnvironment{proof}{o}{
  \par
  \ifbool{naproche@forthel}{}{\vspace{0.5em}}
  \noindent\textit{Proof\IfValueT{#1}{~#1}.}%
}
{
```

```
90   \hfill\ensuremath{\square}
91   \par
92 }
```

Change History

Index

R

S

T

U

V

X