Abstract

The package `create-theorem` provides commands for naming, initializing and configuring theorem-like environments. All of these commands have key-value based interface and are especially useful in multi-language documents, allowing the easy declaration of theorem-like environments that can automatically adapt to the language settings.

How to load it

First, you need a backend to provide the command \newtheorem with the usual behaviour, for example, \texttt{amsthm} or \texttt{ntheorem}. After that, you can simply load the current package with:

\begin{verbatim}
\usepackage[(options)]{create-theorem}
\end{verbatim}

\textbf{Attention}

Since `create-theorem` uses cleveref internally, it should usually be placed near the last of your preamble — notably, it needs to be loaded after \texttt{varioref} and \texttt{hyperref}.

It has the following options:

\texttt{name as context}

- When referencing, the resulted names shall correspond to the current context of your text. For example, the names shall be displayed in English when you are referencing a theorem-like environment in an English context, no matter in which linguistic context the target environment appeared.
- This is the default behavior.
- Synonymous names: \texttt{name-as-context} \texttt{nameascontext} \texttt{regionalref}

\texttt{name as is}

- When referencing, the resulted names shall correspond to the contexts in which the target environments appeared. For example, if the target environment is written in an English context, then its name shall always be displayed in English when referencing, regardless of the current linguistic context.
- Synonymous names: \texttt{name-as-is} \texttt{nameasis} \texttt{originalref}

\texttt{name in link}

- Include the names in the hyperlinks when referencing.
- Synonymous names: \texttt{name-in-link} \texttt{nameinlink}

\texttt{no preset names}

- Disable the preset names. Use this option if you want to define you own name set.
- Synonymous names: \texttt{no-preset-names} \texttt{nopresetnames}
How to use it

2.1 Naming theorem-like environments with \NameTheorem

The syntax of \NameTheorem is as follows:

\NameTheorem{⟨name of environment⟩}{⟨key-value configuration⟩}

Supported keys are:

heading = ⟨configuration⟩
- The heading of the environment, where ⟨configuration⟩ can be:
  - a single string in monolingual documents: heading = ⟨string⟩;
  - a key-value name list in multilingual documents:

  heading = {
    ⟨language name⟩ = ⟨string⟩
  }

heading style = ⟨style⟩
- The style of the heading, you can specify the font, text style, color, etc.
- Synonymous names: heading-style | headingstyle

crefname = ⟨configuration⟩
- The name for \cref the environment, where ⟨configuration⟩ can be:
  - a single string in monolingual documents: crefname = {name}{names};
  - a key-value name list in multilingual documents:

  crefname = {
    ⟨language name⟩ = {singular name}{plural name}
  }

- Also supports the syntax of \crefthename, thus you can assign names of the form:

  [⟨singular definite article⟩]{⟨singular name⟩} [⟨plural definite article⟩]{⟨plural name⟩}

  This would be useful for languages like French, Italian, Spanish, etc.
- Also supports the syntax of \crefthevariantname, thus you can assign different set
  of names for different variants/declensions (the first line in the configuration is the
  default name set, which is used in case no variants is specified when referencing):

  crefname = {
    ⟨language name⟩ = { [...]{...}{...}{...}
      , ⟨variant 1⟩ = [...]{...}{...}{...}
      , ⟨variant 2⟩ = [...]{...}{...}{...}
      ... 
    }
  }

  This would be useful for languages like German, Russian, etc.

crefname style = ⟨style⟩
- The style of “crefname” when referencing, you may specify the font, text style, color, etc.
- Synonymous names: crefname-style | crefnamestyle
Crefname = \langle configuration \rangle

- The name for \Cref the environment, its syntax is the same as that of crefname.
- Also supports the syntax of \Crefthename and \Crefthevariantname.

Crefname style = \langle style \rangle

- The style of “Crefname” when referencing, you may specify the font, text style, color, etc.
- Synonymous names: Crefname-style | Crefnamestyle

numbering style = \langle style \rangle

- The style of numbering in the reference, you can specify the font, text style, color, etc.
- Synonymous names: numbering-style | numberingstyle

use name = \langle list of existed environment(s) separated with semicolon “;” \rangle

- Use the name(s) and style(s) of the given environment(s). If there are multiple ones specified, the result would be a string combining the names, separated with “–”.
- The definite articles (if exist) are chosen to be that of the last given environment.
- Synonymous names: combined | use-name | usename

**Tip**

You can also define the names within \CreateTheorem while initializing the theorem-like environments. \NameTheorem is especially useful for package or class authors who wish to preset suitable names (with styles) in their packages or classes.

### 2.2 Initializing theorem-like environments with \CreateTheorem

The syntax of \CreateTheorem is as follows:

\CreateTheorem{\langle list of the name of environments \rangle}{\langle key-value configuration \rangle}

---

**Attention**

When the \langle key-value configuration \rangle is empty, don’t forget to include the second pair of curly brackets, for example, \CreateTheorem{\langle \rangle}{\langle \rangle}.

**Supported keys are:**

**name** = \langle configuration \rangle or **name style** = \langle configuration \rangle

- Setting the names. Same as \NameTheorem{\langle name of environment \rangle}{\langle configuration \rangle}.
- Synonymous names: name-style | namestyle

**use name** = \langle list of existed environment(s) separated with semicolon “;” \rangle

- Using existed name(s). Same as in \NameTheorem.
- Synonymous names: combined | use-name | usename

**style** = \langle theorem style \rangle

- Specifying the \\texttt{theoremstyle} for the current environment.
- Synonymous names: apply style | apply-style | applystyle

**qed or qed = \langle Q.E.D. symbol \rangle**

- Specifying the Q.E.D. symbol for the current environment.
- Note that the Q.E.D. symbol has already been put in math mode. If you want regular text such as “Q.E.D.”, you need to write *\texttt{qed} = \mathrm{Q.E.D.}.*
- If you are using ntheorem as the backend, then you need to load it with option *thmmarks*.
- Synonymous names: qed symbol | qed-symbol | qedsymbol

---

2.2 Initializing theorem-like environments | 3
parent counter = (parent counter)
   Specifying the (parent counter) for the current environment, i.e., numbering will restart whenever that sectional level is encountered.
   Synonymous names: parent-counter | parentcounter | number within | number-within | numberwithin

shared counter = (shared counter)
   Specifying the (shared counter) for the current environment, i.e., numbering will progress sequentially for all theorem-like environments using this counter.
   Synonymous names: shared-counter | sharedcounter | number like | number-like | numberlike

numberless
   Defining the current environment to be unnumbered.

create starred version
   Defining a corresponding starred (unnumbered) version of the current environment.
   It must be placed before qed if you want the starred version to have a Q.E.D symbol.
   Synonymous names: create-starred-version | createstarredversion | create numberless version | create-numberless-version | createnumberlessversion

copy existed = (existed environment)
   Defining the current environment to be the same as (existed environment).
   This key is usually useful in the following two situations:
   1) To use a more concise name. For example, with \CreateTheorem{thm}{copy existed = theorem}, one can then use the name thm to write theorems.
   2) To remove the numbering of some environments. For example, one can remove the numbering of the remark environment with \CreateTheorem{remark}{copy existed = remark*}.
   Synonymous names: copy-existed | copyexisted

Tip
The names for the following environments have been preset: application, assertion, assumption, axiom, claim, conclusion, conjecture, construction, convention, corollary, definition, example, exercise, fact, hypothesis, lemma, notation, observation, postulate, problem, property, proposition, question, recall, remark, and theorem. If you are fine with the preset names, then there is no need to specify the key “name” while creating them, otherwise you shall have to use the package option “no preset names” to disable the presets and then define your own ones.

Please note that, for the sake of generality, the environment ⟨env⟩ and its starred relative ⟨env⟩* do not share the same set of names when they are separately defined. However, with proper usage of create starred version and copy existed, you should already be able to produce all of the following combinations that shares the same set of names: 1) numbered ⟨env⟩, numbered ⟨env⟩*; 2) numbered ⟨env⟩, unnumbered ⟨env⟩*; 3) unnumbered ⟨env⟩, numbered ⟨env⟩*; and 4) unnumbered ⟨env⟩, unnumbered ⟨env⟩*. I left it as an easy exercise for you ;-) The answer can be found in section 3.2.
2.3 Configuring theorem-like environments with \SetTheorem

The previous two commands are especially useful for package or class writers, while this one is more for the users. If you are not satisfied with preset name styles or numbering settings, then even after initializing the environments, you can still further configure them by means of \SetTheorem, the syntax of which is as follows:

\SetTheorem{⟨list of the name of environments⟩}{⟨key-value configuration⟩}

Supported keys are:

- \texttt{name} = ⟨configuration⟩ and \texttt{name style} = ⟨configuration⟩
  - Same as \NameTheorem{⟨name of environment⟩}{⟨configuration⟩}.
  - Note that this configuration can overwrite those already specified in \NameTheorem.
  - Synonymous names: name-style | namestyle
- \texttt{qed} = ⟨Q.E.D. symbol⟩
  - Specifying the Q.E.D. symbol for the current environment.
  - Note that this configuration only works if you have already enabled the Q.E.D. symbol during the creating phase of the corresponding environment.
  - Synonymous names: qed symbol | qed-symbol | qedsymbol
- \texttt{parent counter} = ⟨parent counter⟩
  - Specifying the ⟨parent counter⟩ for the current environment, i.e., numbering will restart whenever that sectional level is encountered.
  - Note that this configuration can overwrite those already specified in \CreateTheorem.
  - Synonymous names: parent-counter | parentcounter | parent-counter | parentcounter | parent-counter | parentcounter
- \texttt{shared counter} = ⟨shared counter⟩
  - Specifying the ⟨shared counter⟩ for the current environment, i.e., numbering will progress sequentially for all theorem-like environments using this counter.
  - Note that this configuration can overwrite those already specified in \CreateTheorem.
  - Synonymous names: shared-counter | sharedcounter | shared-counter | sharedcounter | shared-counter | sharedcounter

In some cases, you may define an internal environment (for example, a generic version) first and then use it to define the final environment. You may wish to hide the internal names from the users so that they can use \SetTheorem with the name of the final environments. This can be done with the following command:

\SetTheoremBinding{⟨list of the name of environments⟩}{⟨the environment to bind with⟩}

2.4 Setting the names in external language configuration files with \NameTheorems

The command \NameTheorem introduced earlier is for defining the names of a given environment for each language, which is more natural to use within a real-life document. However, for package/class authors wishing to maintain their language configuration files, it would be more convenient to use the following \NameTheorems, which assigns the names for a given language all at once, made it possible to preset the names inside external files.

The syntax of \NameTheorems is as follows (please note that the ⟨language name⟩ here should be consistent with \languagename):

\NameTheorems{⟨language name⟩}{⟨key-value configuration⟩}
Supported keys are (notice that you cannot set the styles via \NameTheorems):

**heading** = ⟨configuration⟩
- The headings of the environments, where ⟨configuration⟩ is a key-value name list:

```latex
heading = {
    ⟨name of environment⟩ = ⟨string⟩
}
```

**crefname** = ⟨configuration⟩
- The names for \cref the environments, where ⟨configuration⟩ is a key-value name list:

```latex
crefname = {
    ⟨name of environment⟩ = {⟨singular name⟩}{⟨plural name⟩}
}
```
- Also supports the syntax of \crefthename and \crefthevariantname. Please refer to the description of \NameTheorem for more details.

**Crefname** = ⟨configuration⟩
- The names for \Cref the environments, its syntax is the same as that of crefname.
- Also supports the syntax of \Crefthename and \Crefthevariantname. Please refer to the description of \NameTheorem for more details.

*If you’re feeling confused, don’t worry. Let’s now take a look at some examples.*

### 3.1 The environment idea

First, let’s getting familiar with these two commands by creating the environment idea.

```latex
\NameTheorem{idea}
{
    heading = Idea,
    crefname = {idea}{ideas},
    Crefname = {Idea}{Ideas},
}
\CreateTheorem{idea}{ parent counter = section }
```

or to do it in one turn:

```latex
\CreateTheorem{idea}
{
    name = {
        heading = Idea,
        crefname = {idea}{ideas},
        Crefname = {Idea}{Ideas},
    },
    parent counter = section,
}
```
This is not exciting at all. Now, let’s say we are writing a trilingual note in English, French and German. (I shall omit the `\NameTheorem` version and do it all at once in `\CreateTheorem`.)

```latex
\CreateTheorem{idea}
{
  name = {
    heading = {
      english = Idea,
      french = Idée,
      ngerman = Idee, },
    crefname = {
      english = {idea}{ideas},
      french = [l']{idée}[les]{idées},
      ngerman = { {Idee}{Idee}
                     , Nominativ = [die]{Ideee}[die]{Ideen}
                     , Genitiv = [der]{Ideee}[der]{Ideen}
                     , Dativ = [der]{Ideee}[den]{Ideen}
                     , Akkusativ = [die]{Ideee}[die]{Ideen}
                     },
    }},
  parent counter = section,
}
```

With this, if you use `\selectlanguage{french}`, the `idea` environment shall be automatically displayed as “Idée”. And if you `\crefthe` it, the definite article and the name would show up properly just as expected.

The same happens for German with `\selectlanguage{ngerman}`, and when referencing an `idea` environment, you may specify the declension as with `\crefthe[{⟨prep⟩},variant=Nominativ]{⟨label⟩}`, or simply with shortcut such as `\crefthe[{⟨prep⟩},nom.]{⟨label⟩}`.

---

**Tip**

For more detailed usage of the referencing command `\crefthe`, please refer to the documentation of the package `crefthe`.

Next we shall deal with the problem of numbering. Let’s continue to use this environment `idea` for demonstration — suppose that we have already set the names with `\NameTheorem`.

3.2 | Let’s play with numbering

Remember the exercise I left you in the previous section? Let’s do it together now.
3.2.1 *Numbered* idea and numbered idea*

This is easy, *copy existed* suffices:

\begin{verbatim}
\CreateTheorem{idea}{parent counter = section}
\CreateTheorem{idea*}{copy existed = idea}
\end{verbatim}

3.2.2 *Numbered* idea and unnumbered idea*

This is the most common situation, *create starred version* will do.

\begin{verbatim}
\CreateTheorem{idea}
{
    parent counter = section,
    create starred version,
}
\end{verbatim}

\begin{attention}
Please note that you cannot use \texttt{\CreateTheorem{idea*}{numberless}} here, since we don't have the names defined for idea*.
\end{attention}

3.2.3 Unnumbered idea and numbered idea*

This is a bit tricky: by default we can only create numbered idea or unnumbered idea*, and the question is how to switch them. We shall need an intermediary for this purpose.

\begin{verbatim}
\CreateTheorem{idea}{create starred version}
\CreateTheorem{idea-temp}{copy existed = idea*}
\CreateTheorem{idea*}{copy existed = idea}
\CreateTheorem{idea}{copy existed = idea-temp}
\end{verbatim}

3.2.4 Unnumbered idea and unnumbered idea*

This is essentially the combination of the first two cases — we need to create idea* first and then copy it to idea:

\begin{verbatim}
\CreateTheorem{idea}{create starred version}
\CreateTheorem{idea}{copy existed = idea*}
\end{verbatim}

In each case, the two environments idea and idea* share the same set of names.

\begin{attention}
The sole purpose of this section is to demonstrate the feature of this package — some combinations are not recommended to use in the actual documents.
\end{attention}

3.3 The *proofless* version — theorems with a Q.E.D. symbol

Sometimes you may encounter a theorem without a proof, in which case you might want a Q.E.D. symbol when the theorem is finished. This can be easily achieved via:

\begin{verbatim}
\CreateTheorem { theorem } { create starred version }
\CreateTheorem { theorem+ } { copy existed = theorem, qed }
\CreateTheorem { theorem++ } { copy existed = theorem*, qed }
\end{verbatim}
The code above defines two new environments `theorem+` and `theorem+*` in addition to `theorem` and `theorem*`. The + version behaves exactly the same as the usual version, except that it has a Q.E.D. symbol.

3.4 | Redefine the proof environment

If you wish to have a proof environment with a custom theorem style, or to have a numbered version `proof*` of it, the following code could be helpful:

```latex
\ExplSyntaxOn
\newcounter { proof }
\tl_new:N \l_mymodule_name_of_proof_tl
\CreateTheorem { proof_inner }
  { name = { heading = { \l_mymodule_name_of_proof_tl } },
    create-starred-version,
    style = remark,
    qed,
    shared-counter = proof,
  }
\cs_undefine:c { proof }
\cs_undefine:c { endproof }
\NewDocumentEnvironment { proof } { O{\proofname} } { \tl_set:Nn \l_mymodule_name_of_proof_tl { #1 } \begin { proof_inner* } }
{ \end { proof_inner* } }
\NewDocumentEnvironment { proof* } { O{\proofname} } { \tl_set:Nn \l_mymodule_name_of_proof_tl { #1 } \begin { proof_inner } }
{ \end { proof_inner } }
\SetTheoremBinding { proof } { proof_inner* }
\SetTheoremBinding { proof* } { proof_inner }
\ExplSyntaxOff
```

It defines an environment `proof_inner` (with its starred variant) with theorem style `remark` to mimic the default style (you are welcome to use your own style here), and with the name to be a variable which is latter used to define the actual environments `proof` and `proof*`. These
two environments are defined in such a way that proof is the usual unnumbered version and proof* is the numbered version. The \SetTheoremBinding lines are to ensure that user can directly write \SetTheorem{proof} instead of \SetTheorem{proof_inner*}.

\begin{center}
\textbf{\textit{Attention}}
\end{center}

The code above requires amsthm. If you are using ntheorem as the backend, then you need to load it with option amsthm, and remove the \newcounter line.

3.5 | Advanced topic: setting the names in an external file

A typical configuration looks like this:

\begin{verbatim}
\NameTheorems { english }
{
    , heading = {
        , theorem       = Theorem
        , proposition   = Proposition
        ...
    }
    , crefname = {
        , theorem       = {theorem}{theorems}
        , proposition   = {proposition}{propositions}
        ...
    }
    , Crefname = {
        , theorem       = {Theorem}{Theorems}
        , proposition   = {Proposition}{Propositions}
        ...
    }
}
\end{verbatim}

Here is an example for French:

\begin{verbatim}
\NameTheorems { french }
{
    , heading = {
        , theorem       = Théorème
        , proposition   = Proposition
        , example       = Exemple
        ...
    }
    , crefname = {
        , theorem       = [le]{théorème}[les]{théorèmes}
        , proposition   = [la]{proposition}[les]{ propositions}
        , example       = [l']{exemple}[les]{exemples}
        ...
    }
    , Crefname = {

\end{verbatim}
And an example for German:

\NameTheorems { ngerman }
{
    , heading = {
        , theorem = Satz
        ...
    }
    , crefname = {
        , theorem = { {Satz}\{Sätze}\}
            , Nominativ = [der]\{Satz\}[die]\{Sätze\}
            , Genitiv = [des]\{Satzes\}[der]\{Sätze\}
            , Dativ = [dem]\{Satz\}[den]\{Sätze\}
            , Akkusativ = [den]\{Satz\}[die]\{Sätze\}
        }
        ...
    }
    , Crefname = {
        , theorem = { {Satz}\{Sätze\}
            , Nominativ = [Der]\{Satz\}[Die]\{Sätze\}
            , Genitiv = [Des]\{Satzes\}[Der]\{Sätze\}
            , Dativ = [Dem]\{Satz\}[Den]\{Sätze\}
            , Akkusativ = [Den]\{Satz\}[Die]\{Sätze\}
        }
        ...
    }
}

The configuration using \NameTheorems is compatible with that using \NameTheorem and there is no need to worry about duplicated definitions — new settings will automatically overwrite the old ones.
Known issues

- `create-theorem` modifies some undocumented internal macros of `cleveref`, so the behaviour might not be stable if `cleveref` gets updated.
- The counter aliasing function is still not perfect, (sometimes) causing incorrect ordering in the result of `\cref`.
- There might be inaccuracies in the translation of those preset names.

If you run into any issues or have ideas for improvement, feel free to discuss on:

    https://github.com/Jinwen-XU/create-theorem/issues

or email me via ProjLib@outlook.com.