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.

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

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

Tip

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

It has the following options:

- **name as context**
  - When referencing, the resulted names correspond to the current context. For example, the English names will be displayed when referencing a theorem-like environment in English context, no matter which linguistic context the original environment is in.
  - Synonymous names: `name-as-context` | `nameascontext` | `regionalref`

- **name as is**
  - When referencing, the resulted names correspond to the contexts in which the corresponding environments appeared. For example, if the environment is written in an English context, then it shall always be the English names displayed when referencing it, regardless of the current linguistic context.
  - Synonymous names: `name-as-is` | `nameasis` | `originalref`

- **name in link**
  - Include the names in the hyperlinks when referencing.
  - Synonymous names: `name-in-link` | `nameinlink`

- **no preset names**
  - Disable preset names. Use this option if you want to define you own name set.
  - Synonymous names: `no-preset-names` | `nopresetnames`
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⟩⟩}

**crefname style** = ⟨style⟩
- The style of the “crefname”, you can specify the font, text style, color, etc.
- Synonymous names: crefname-style | crefnamestyle

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

**Crefname style** = ⟨style⟩
- The style of the “Crefname”, you can specify the font, text style, color, etc.
- Synonymous names: Crefname-style | Crefnamestyle

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

**use name** = ⟨list of existed environment(s) separated with semicolon “;”⟩
- Use the name(s) and style(s) of given environment(s). If there are multiple ones specified, the result would be a string combining the names, separated with “-”.
- Synonymous names: combined | use-name | usename
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{⟨list of the name of environments⟩}{⟨key-value configuration⟩}

Supported keys are:

- \texttt{name} = ⟨configuration⟩ or \texttt{name style} = ⟨configuration⟩
  - Setting the names. Same as \NameTheorem{⟨name of environment⟩}{⟨configuration⟩}.
  - Synonymous names: name-style | namestyle

- \texttt{use name} = ⟨list of existed environment(s) separated with semicolon “;”⟩
  - Using existed name(s). Same as in \NameTheorem.
  - Synonymous names: combined | use-name | usename

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

- \texttt{parent counter} = ⟨parent counter⟩
  - Specifying the ⟨parent counter⟩ for the current environment, \textit{i.e.}, numbering will restart whenever that sectional level is encountered.
  - Synonymous names: parent-counter | parentcounter | number-within | number-within | numberwithin

- \texttt{shared counter} = ⟨shared counter⟩
  - Specifying the ⟨shared counter⟩ for the current environment, \textit{i.e.}, numbering will progress sequentially for all theorem-like environments using this counter.
  - Synonymous names: shared-counter | sharedcounter | number-like | number-like | numberlike

- \texttt{numberless}
  - Defining the current environment to be unnumbered.

- \texttt{create starred version}
  - Defining a corresponding starred (unnumbered) version of the current environment.
  - Synonymous names: create-starred-version | createstarredversion | create-numberless-version | create-numberless-version | createnumberlessversion

- \texttt{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-existent | copyexisted
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, by default, an normal environment \textlangle env \textrangle and its starred relative \textlangle env \textrangle* do not share the same set of names, for the sake of generality. However, with proper usage of create starred version and copy existed, you are already able to produce all of the following combinations that shares the same set of names: 1) numbered \textlangle env \textrangle, numbered \textlangle env \textrangle*; 2) numbered \textlangle env \textrangle, unnumbered \textlangle env \textrangle*; 3) unnumbered \textlangle env \textrangle, numbered \textlangle env \textrangle*; and 4) unnumbered \textlangle env \textrangle, unnumbered \textlangle env \textrangle*. I left it as an easy exercise for you ;‑)

2.3 Configuring theorem-like environments with \textbackslash 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 \textbackslash SetTheorem, the syntax of which is as follows:

\textbackslash SetTheorem\{\langle list of the name of environments\}\{\langle key-value configuration\}\}

\textbf{TIP}

\textbackslash SetTheorem should only be used in the preamble of your document.

Supported keys are:

- \textbf{name} = \langle configuration \rangle and \textbf{name style} = \langle configuration \rangle
  - Same as \textbackslash NameTheorem\{\langle name of environment\}\{\langle configuration\}\}.
  - Note that this configuration can overwrite those already specified in \textbackslash NameTheorem.
  - Synonymous names: name-style | namestyle

- \textbf{parent counter} = \langle parent counter \rangle
  - Specifying the \langle parent counter \rangle for the current environment, \textit{i.e.}, numbering will restart whenever that sectional level is encountered.
  - Note that this configuration can overwrite those already specified in \textbackslash CreateTheorem.
  - Synonymous names: parent-counter | parentcounter | number within | number-within | numberwithin

- \textbf{shared counter} = \langle shared counter \rangle
  - Specifying the \langle shared counter \rangle for the current environment, \textit{i.e.}, numbering will progress sequentially for all theorem-like environments using this counter.
  - Note that this configuration can overwrite those already specified in \textbackslash CreateTheorem.
  - Synonymous names: shared-counter | sharedcounter | number like | number-like | numberlike
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.

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

or to do it in one turn:

\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 bilingual note in English and French. (I shall omit the \NameTheorem version and do it all at once in \CreateTheorem.)

\CreateTheorem{idea}{
  name = {
    heading = { english = Idea, 
                 french = Idée, },
    crefname = { english = {idea}{ideas}, 
                 french = [l']{idée}[les]{idées}, },
    Crefname = { english = {Idea}{Ideas}, 
                 french = [L']{idée}[Les]{idées}, },
  },
  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 showed up properly just as expected.

Next we shall deal with the numbering problem. Let’s continue to use this environment idea for demonstration – suppose that we have set the names up 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:

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

3.2.2 Numbered idea and unnumbered idea*

This is the easiest, create starred version will do.

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

Notice that you cannot use \CreateTheorem{idea*}{numberless} here, since we don’t have names defined for idea*.

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.

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

3.2.4 Unnumbered idea and unnumbered idea*

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

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

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

/ 4 /

Known issues

- The current mechanism does not work well for German, a problem originated in the package crefthe. The author plans to adopt a more refined approach in a later version so as to support the various grammatical situations in German.
- create-theorem modifies some undocumented internal macros of cleveref, so the behaviour might not be stable if cleveref gets updated.
- There may 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.