create-theorem

Initializing theorem-like environments with multilingual support

Jinwen XU

February 2022, in Paris

Abstract

The package create-theorem provides the commands \NameTheorem and \CreateTheorem for naming and initializing theorem-like environments. They both have key-value based interface and are especially useful in multi-language documents, allowing you to easily declare theorem-like environments that automatically adapt to the language settings.

/ 1 / How to load it

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{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:

- `draft` or `fast`
  - Fast mode. The functionality will be appropriately reduced to get faster compilation speed, recommended to use during the writing stage.
- `name as context`
  - When referencing, the names correspond to the current language context. For example, the English names will be displayed when referencing a theorem-like environment in English context, no matter which language context the original environment is in.
  - Synonymous names: `name-as-context` | `nameascontext` | `regionalref`
- `name as is`
  - When referencing, the names correspond to the language contexts in which the environments are defined. For example, if the environment is defined in English context, then the English names will be displayed when referencing it, regardless of the current language 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`

Corresponding to: create-theorem 2022/02/27

/ 2 /

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⟩
}
```

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

If you're feeling confused, don't worry, there is an Examples section later illustrating its usage.

> **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{⟨name of environment⟩}{⟨key-value configuration⟩}
```

Supported keys are:

- `name = ⟨configuration⟩`
  - Same as \NameTheorem{⟨*name of environment*⟩}{⟨*configuration*⟩}.

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

If you're feeling confused, don't worry, check out the Examples section below.

Tip

The names for the following environments have been preset: `assertion`, `assumption`, `axiom`, `conclusion`, `conjecture`, `convention`, `corollary`, `definition`, `example`, `exercise`, `fact`, `hypothesis`, `lemma`, `notation`, `observation`, `postulate`, `problem`, `property`, `proposition`, `question`, `remark` and `theorem`. If you are fine with the preset names, then you don't 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 define your own ones.

Please note that, by default, an normal environment ⟨*env*⟩ and its starred relative ⟨*env*⟩`*` 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 ⟨*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 ;-)

/ 3 /

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 to you at the end of the previous section? Let's do this together.

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 cref the. The author plans to adopt a more refined approach in later versions so as to support the various situations in German.
- If the option "`name in link`" is enabled, the "`name as is`" mode shall take longer time to produce the final result, for the current implementation uses `\regexpatchcmd` to patch some of the referencing formats, which is not very efficient.
- 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`.