



simplivre
WRITE YOUR BOOKS IN
A SIMPLE AND CLEAR WAY

Corresponding to: simplivre 2021/06/23

JINWEN XU

June 2021, Beijing



Preface

1 simplivre is part of the minimalist class series. Its name is taken from French words “simple”
2 and “livre” (for “book”). The entire collection includes minimart and einfart for typesetting
3 articles, and minimbook and simplivre for typesetting books. My original intention in de-
4 signing them was to write drafts and notes that look simple yet not shabby.

5 simplivre has multi-language support, including Chinese (simplified and traditional), En-
6 glish, French, German, Italian, Japanese, Portuguese (European and Brazilian), Russian and
7 Spanish. These languages can be switched seamlessly in a single document. Due to the
8 usage of custom fonts, einfart requires \LaTeX or \Lua\LaTeX to compile.

9 This documentation is typeset using simplivre. You can think of it as a short introduction
10 and demonstration.

TIP

Multi-language support, theorem-like environments, draft marks and some other features are provided by the [ProjLib](#) toolkit. Here we only briefly discuss how to use it with this document class. For more detailed information, you can refer to the documentation of [ProjLib](#).



Contents

I INSTRUCTION

1	Initialization	3
1	1 How to load it	3
2	2 Options	3
2	On the default fonts	5
3	Some instructions	7
1	1 Language configuration	7
2	2 Theorems and how to reference them	8
3	3 Define a new theorem-like environment	8
4	4 Draft mark	10
5	5 On the line numbers	10
6	6 On the footnotes in the title	10
7	7 Known issues	10

II DEMONSTRATION

4	Document templates	13
1	1 The standard way	13
2	2 The $\mathcal{A}\mathcal{M}\mathcal{S}$ way	14
5	Heading on Level 0 (chapter)	15
1	1 Heading on Level 1 (section)	15
1.1	1.1 Heading on Level 2 (subsection)	15
2	2 Lists	16
2.1	2.1 Example for list (itemize)	16
2.2	2.2 Example for list (enumerate)	16
2.3	2.3 Example for list (description)	17

PART I

INSTRUCTION

You can add some introduction text here via `\parttext<text>`.

1

Initialization

/ 1 /

How to load it

One only needs to put

```
\documentclass{simplivre}
```

as the first line to use the `simplivre` class.

ATTENTION

You need to use either $\text{X}\text{\LaTeX}$ or $\text{Lua}\text{\LaTeX}$ engine to compile.

/ 2 /

Options

`simplivre` offers the following options:

- `draft` or `fast`
 - The option `fast` enables a faster but slightly rougher style, main differences are:
 - * Use simpler math font configuration;
 - * Do not use `hyperref`;
 - * Enable the fast mode of `ProjLib` toolkit.

TIP

During the draft stage, it is recommended to use the `fast` option to speed up compilation. At the end, one should remove the “fast” option to get the final version. When in `fast` mode, there will be a watermark “DRAFT” to indicate that you are currently in the draft mode.

- `a4paper` or `b5paper`
 - Optional paper size. The default paper size is $7\text{in} \times 10\text{in}$.
- `palatino`, `times`, `garamond`, `biolinum` | `useosf`
 - Font options. As the name suggest, font with corresponding name will be loaded.
 - The `useosf` option is used to enable the old-style figures.

- `allowbf`
 - Allow boldface. When this option is enabled, the title, titles of all levels and names of theorem-like environments will be bolded.
- `classical`
 - Classic mode. When this option is enabled, the style will become more regular: paragraphs are indented, the use of underlines are reduced, heading styles are changed, and the theorem styles will be much closer to common styles.

TIP

`allowbf + classical` is probably a good choice if you prefer traditional style.

- `useindent`
 - Use paragraph indentation instead of inter-paragraph spacing.
- `runin`
 - Use the “runin” style for `\subsubsection`
- `puretext` or `nothms`
 - Pure text mode. Does not load theorem-like environments.
- `nothmnum`
 - Theorem-like environments will not be numbered.
- The language options `EN / english / English`, `FR / french / French`, etc.
 - For the option names of a specific language, please refer to *⟨language name⟩* in the next section. The first specified language will be used as the default language.

In addition, the commonly used `oneside` and `twoside` options are also available. Two-page layout is used by default.

2 On the default fonts

- 1 By default, simplivre uses Palatino Linotype as the English font, FounderType's YouSong
2 and YouHei GBK as the Chinese fonts, and partially uses Neo Euler as the math font. Among
3 them, Neo Euler can be downloaded at <https://github.com/khaledhosny/euler-otf>.
4 The other fonts are not free, you need to purchase and install them on your own. (For the
5 Chinese fonts, visit FounderType's website for detail: <https://www.foundertype.com>).

FONT DEMONSTRATION

- English main font. English sans serif font. English typewriter font.
- 中文主要字体，中文无衬线字体
- 数学示例: $\alpha, \beta, \gamma, \delta, 1, 2, 3, 4, a, b, c, d,$

$$\text{li}(x) := \int_2^{\infty} \frac{1}{\log t} dt$$

- 6 When the corresponding font is not installed, fonts that comes with TeX Live will be used
7 instead. In this case, the experience might be reduced.

3

Some instructions

1 Many of the features described next are provided by the [ProjLib](#) toolkit. Only the basic usage
2 is mentioned here. For more details, please refer to its user documentation.

/ 1 /

Language configuration

3 simplivre has multi-language support, including Chinese (simplified and traditional), En-
4 glish, French, German, Italian, Japanese, Portuguese (European and Brazilian), Russian and
5 Spanish. The language can be selected by the following macros:

- 6 • `\UseLanguage{<language name>}` is used to specify the language. The corresponding
7 setting of the language will be applied after it. It can be used either in the preamble or
8 in the main body. When no language is specified, “English” is selected by default.
- 9 • `\UseOtherLanguage{<language name>}{<content>}`, which uses the specified language
10 settings to typeset *<content>*. Compared with `\UseLanguage`, it will not modify the line
11 spacing, so line spacing would remain stable when CJK and Western texts are mixed.

12 *<language name>* can be (it is not case sensitive, for example, French and french have the
13 same effect):

- 14 • Simplified Chinese: CN, Chinese, SChinese or SimplifiedChinese
- 15 • Traditional Chinese: TC, TChinese or TraditionalChinese
- 16 • English: EN or English
- 17 • French: FR or French
- 18 • German: DE, German or ngerman
- 19 • Italian: IT or Italian
- 20 • Portuguese: PT or Portuguese
- 21 • Portuguese (Brazilian): BR or Brazilian
- 22 • Spanish: ES or Spanish
- 23 • Japanese: JP or Japanese
- 24 • Russian: RU or Russian

25 In addition, you can also add new settings to selected language:

- 26 • `\AddLanguageSetting{<settings>}`
27 – Add *<settings>* to all supported languages.

- `\AddLanguageSetting(<language name>){<settings>}`
– Add `<settings>` to the selected language `<language name>`.

For example, `\AddLanguageSetting(German){\color{orange}}` can make all German text displayed in orange (of course, one then need to add `\AddLanguageSetting{\color{black}}` in order to correct the color of the text in other languages).

/ 2 /

Theorems and how to reference them

Environments such as `definition` and `theorem` have been pre-defined and can be used directly.

More specifically, preset environments include: `assumption`, `axiom`, `conjecture`, `convention`, `corollary`, `definition`, `definition-proposition`, `definition-theorem`, `example`, `exercise`, `fact`, `hypothesis`, `lemma`, `notation`, `problem`, `property`, `proposition`, `question`, `remark`, `theorem`, and the corresponding unnumbered version with an asterisk `*` in the name. The titles will change with the current language. For example, `theorem` will be displayed as “Theorem” in English mode and “Théorème” in French mode.

When referencing a theorem-like environment, it is recommended to use `\cref{<label>}`. In this way, there is no need to explicitly write down the name of the corresponding environment every time.

EXAMPLE

```
\begin{definition}[Strange things] \label{def: strange} ...
```

will produce

DEFINITION 3.1 | (Strange things) This is the definition of some strange objects. There is approximately an one-line space before and after the theorem environment, and there will be a symbol to mark the end of the environment. □

`\cref{def: strange}` will be displayed as: DEFINITION 3.1.

After using `\UseLanguage{French}`, a theorem will be displayed as:

THÉOREME 3.1 | (Inutile) Un théorème en français. □

By default, when referenced, the name of the theorem always matches the language of the context in which the theorem is located. For example, the definition above is still displayed in English in the current French mode : DEFINITION 3.1 and THÉOREME 3.1. If you want the name of the theorem to match the current context when referencing, you can add `regionalref` to the global options.

/ 3 /

Define a new theorem-like environment

If you need to define a new theorem-like environment, you must first define the name of the environment in the language to use:

1 • `\NameTheorem`[$\langle\textit{language name}\rangle$]{ $\langle\textit{name of environment}\rangle$ }{ $\langle\textit{name string}\rangle$ }

2 For $\langle\textit{language name}\rangle$, please refer to the section on language configuration. When $\langle\textit{language}$
3 $\textit{name}\rangle$ is not specified, the name will be set for all supported languages. In addition, envi-
4 ronments with or without asterisk share the same name, therefore, `\NameTheorem{envname}`
5 `*`{...} has the same effect as `\NameTheorem{envname}{...}`.

6 And then define this environment in one of following five ways:

- 7 • `\CreateTheorem*`{ $\langle\textit{name of environment}\rangle$ }
- 8 – Define an unnumbered environment $\langle\textit{name of environment}\rangle$
- 9 • `\CreateTheorem`{ $\langle\textit{name of environment}\rangle$ }
- 10 – Define a numbered environment $\langle\textit{name of environment}\rangle$, numbered in order 1,2,3,...
- 11 • `\CreateTheorem`{ $\langle\textit{name of environment}\rangle$ }[$\langle\textit{numbered like}\rangle$]
- 12 – Define a numbered environment $\langle\textit{name of environment}\rangle$, which shares the counter
- 13 $\langle\textit{numbered like}\rangle$
- 14 • `\CreateTheorem`{ $\langle\textit{name of environment}\rangle$ } $\langle\textit{numbered within}\rangle$
- 15 – Define a numbered environment $\langle\textit{name of environment}\rangle$, numbered within the
- 16 counter $\langle\textit{numbered within}\rangle$
- 17 • `\CreateTheorem`{ $\langle\textit{name of environment}\rangle$ }($\langle\textit{existed environment}\rangle$)
- 18 `\CreateTheorem*`{ $\langle\textit{name of environment}\rangle$ }($\langle\textit{existed environment}\rangle$)
- 19 – Identify $\langle\textit{name of environment}\rangle$ with $\langle\textit{existed environment}\rangle$ or $\langle\textit{existed environment}\rangle*$.
- 20 – This method is usually useful in the following two situations:
- 21 1. To use a more concise name. For example, with `\CreateTheorem{thm}`
- 22 (theorem), one can then use the name `thm` to write theorem.
- 23 2. To remove the numbering. For example, one can remove the numbering of
- 24 the remark environment with `\CreateTheorem{remark}(remark*)`.

TIP

This macro utilizes the feature of `amsthm` internally, so the traditional `theoremstyle` is also applicable to it. One only needs declare the style before the relevant definitions.

25 Here is an example. The following code:

```
\NameTheorem[EN]{proofidea}{Idea}
\CreateTheorem*{proofidea*}
\CreateTheorem{proofidea}<section>
```

26 defines an unnumbered environment `proofidea*` and a numbered environment `proofidea`
27 (numbered within section) respectively. They can be used in English context. The effect is
28 as follows:

29 Idea | The `proofidea*` environment. ☐

30 Idea 3.1 | The `proofidea` environment. ☐

Draft mark

1 You can use `\dnf` to mark the unfinished part. For example:

- 2 • `\dnf` or `\dnf<...>`. The effect is: `To be finished #1` or `To be finished #2: ...`.
 3 The prompt text changes according to the current language. For example, it will be
 4 displayed as `Pas encore fini #3` in French mode.

5 Similarly, there is `\needgraph` :

- 6 • `\needgraph` or `\needgraph<...>`. The effect is:

7 `A graph is needed here #1`

8 or

9 `A graph is needed here #2: ...`

10 The prompt text changes according to the current language. For example, in French
 11 mode, it will be displayed as

12 `Il manque une image ici #3`

On the line numbers

13 Line numbers can be turned on and off at any time. `\linenumbers` is used to enable the
 14 line numbers, and `\nolinenumbers` is used to disable them. For the sake of beauty, the
 15 title, table of contents, index and some other elements are not numbered.

On the footnotes in the title

16 In `\section` or `\subsection` , if you wish to add footnotes, you can only:

- 17 • first write `\mbox{\protect\footnotemark}`,
 18 • then add `\footnotetext{...}` afterwards.

19 This is a disadvantage brought about by the underline decoration of the title.

Known issues

- 20 • The font settings are still not perfect.
- 21 • Since many features are based on the ProjLib toolkit, simplivre inherits all its problems.
 22 For details, please refer to the "Known Issues" section of the ProjLib documentation.
- 23 • The error handling mechanism is incomplete: there is no corresponding error prompt
 24 when some problems occur.
- 25 • There are still many things that can be optimized in the code.

PART II

DEMONSTRATION

4

Document templates

/ 1 /

The standard way

- 1 If you want to write in the standard way, you can refer to the following example:

```
\documentclass{simplivre}
\usepackage{ProjLib} % Load ProjLib toolkit

\UseLanguage{French} % Use French from here

\begin{document}

\title{Le Titre}
\author{Auteur}
\date{\PJLdate{2022-04-01}}

\maketitle

\chapter{Un théorème}

%% Theorem-like environments can be used directly
\begin{theorem}\label{thm:abc}
    Ceci est un théorème.
\end{theorem}

Référence du théorème: \cref{thm:abc}
    % It is recommended to use clever reference

\end{document}
```

- 2 If you wish to switch to the standard class later, just replace the first two lines with:

```
\documentclass{article}
\usepackage[a4paper,margin=1in]{geometry}
\usepackage[hidelinks]{hyperref}
\usepackage{palatino}{ProjLib} % Load ProjLib toolkit
```

- 1 If you intend to switch to the journal template in the future and thus want to use the writing
2 style as in the \mathcal{AMS} classes, you can refer to the following example:

```
\documentclass{simplivre}
\usepackage{ProjLib} % Load ProjLib toolkit

\UseLanguage{French} % Use French from here

\begin{document}

\title{Le Titre}
\author{Auteur 1}
\address{Adresse 1}
\email{\href{Courriel 1}{Courriel 1}}
\author{Auteur 1}
\address{Adresse 1}
\email{\href{Courriel 2}{Courriel 2}}
\date{\PJLdate{2022-04-01}}
\subjclass{*****}
\keywords{...}

\maketitle

\chapter{Première section}

%% Theorem-like environments can be used directly
\begin{theorem}\label{thm:abc}
    Ceci est un théorème.
\end{theorem}

Référence du théorème: \cref{thm:abc}
    % It is recommended to use clever reference

\end{document}
```

- 3 In this way, if you wish to switch to \mathcal{AMS} class later, just replace the first two lines with:

```
\documentclass{amsart}
\usepackage[a4paper,margin=1in]{geometry}
\usepackage[hidelinks]{hyperref}
\usepackage{palatino}{ProjLib} % Load ProjLib toolkit
```

5

Heading on Level 0 (chapter)

1 Hello, here is some text without a meaning. This text should show what a printed text will
2 look like at this place. If you read this text, you will get no information. Really? Is there
3 no information? Is there a difference between this text and some nonsense like “Huardest
4 gefburn”? Kjift – not at all! A blind text like this gives you information about the selected
5 font, how the letters are written and an impression of the look. This text should contain all
6 letters of the alphabet and it should be written in of the original language. There is no need
7 for special content, but the length of words should match the language.

/ 1 /

Heading on Level 1 (section)

8 Hello, here is some text without a meaning. This text should show what a printed text will
9 look like at this place. If you read this text, you will get no information. Really? Is there
10 no information? Is there a difference between this text and some nonsense like “Huardest
11 gefburn”? Kjift – not at all! A blind text like this gives you information about the selected
12 font, how the letters are written and an impression of the look. This text should contain all
13 letters of the alphabet and it should be written in of the original language. There is no need
14 for special content, but the length of words should match the language.

1.1 | Heading on Level 2 (subsection)

15 Hello, here is some text without a meaning. This text should show what a printed text will
16 look like at this place. If you read this text, you will get no information. Really? Is there
17 no information? Is there a difference between this text and some nonsense like “Huardest
18 gefburn”? Kjift – not at all! A blind text like this gives you information about the selected
19 font, how the letters are written and an impression of the look. This text should contain all
20 letters of the alphabet and it should be written in of the original language. There is no need
21 for special content, but the length of words should match the language.

Heading on Level 3 (subsubsection)

22 Hello, here is some text without a meaning. This text should show what a printed text will
23 look like at this place. If you read this text, you will get no information. Really? Is there
24 no information? Is there a difference between this text and some nonsense like “Huardest
25 gefburn”? Kjift – not at all! A blind text like this gives you information about the selected
26 font, how the letters are written and an impression of the look. This text should contain all

1 letters of the alphabet and it should be written in of the original language. There is no need
2 for special content, but the length of words should match the language.

3 **HEADING ON LEVEL 4 (PARAGRAPH)** Hello, here is some text without a meaning. This text
4 should show what a printed text will look like at this place. If you read this text, you will
5 get no information. Really? Is there no information? Is there a difference between this text
6 and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives
7 you information about the selected font, how the letters are written and an impression of
8 the look. This text should contain all letters of the alphabet and it should be written in of
9 the original language. There is no need for special content, but the length of words should
10 match the language.

/ 2 /
Lists

2.1 | Example for list (itemize)

- 11 ● First item in a list
- 12 ● Second item in a list
- 13 ● Third item in a list
- 14 ● Fourth item in a list
- 15 ● Fifth item in a list

*Example for list (4*itemize)*

- 16 ● First item in a list
 - 17 – First item in a list
 - 18 * First item in a list
 - 19 · First item in a list
 - 20 · Second item in a list
 - 21 * Second item in a list
 - 22 – Second item in a list
- 23 ● Second item in a list

2.2 | Example for list (enumerate)

- 24 1. First item in a list
- 25 2. Second item in a list
- 26 3. Third item in a list
- 27 4. Fourth item in a list
- 28 5. Fifth item in a list

*Example for list (4*enumerate)*

- 29 1. First item in a list
 - 30 (a) First item in a list
 - 31 i. First item in a list
 - 32 A. First item in a list
 - 33 B. Second item in a list
 - 34 ii. Second item in a list

- 1 (b) Second item in a list
- 2 2. Second item in a list

2.3 | Example for list (description)

- 3 **First** item in a list
- 4 **Second** item in a list
- 5 **Third** item in a list
- 6 **Fourth** item in a list
- 7 **Fifth** item in a list

*Example for list (4*description)*

- 8 **First** item in a list
 - 9 **First** item in a list
 - 10 **First** item in a list
 - 11 **First** item in a list
 - 12 **Second** item in a list
 - 13 **Second** item in a list
 - 14 **Second** item in a list
 - 15 **Second** item in a list