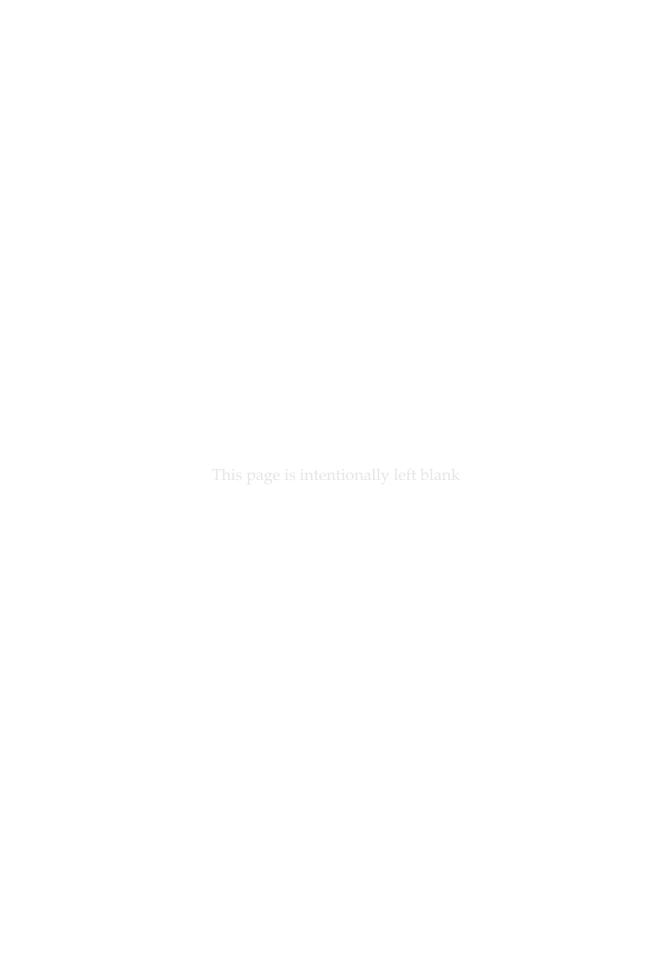
ProjLib

Simplivre Write your books in A SIMPLE AND CLEAR WAY

Corresponding to: simplivre 2021/07/27

Jinwen XU

July 2021, Beijing



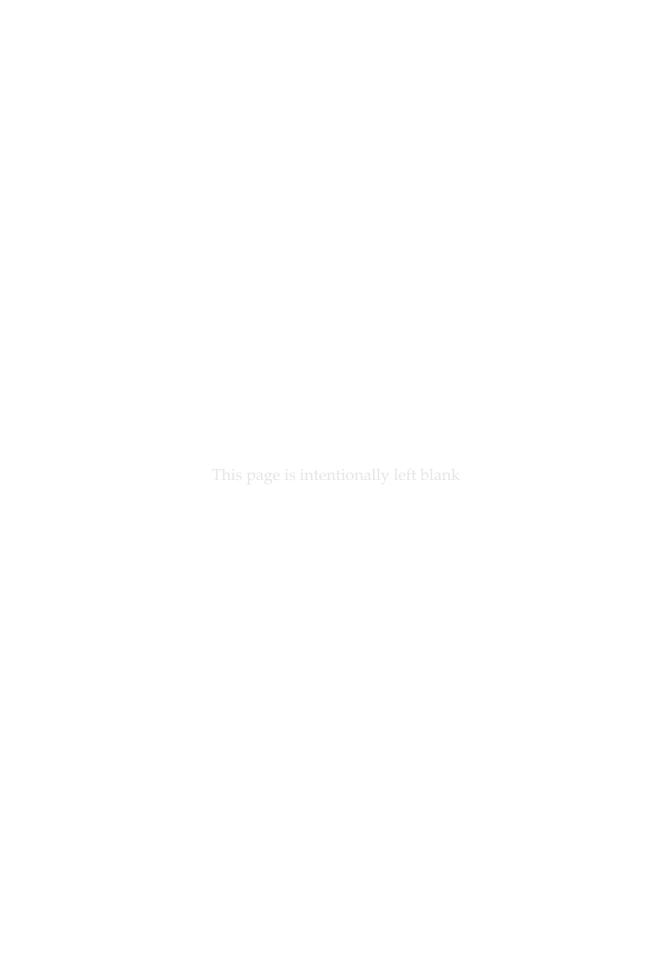


Preface

- simplivre is part of the minimalist class series. Its name is taken from French words "simple" and "livre" (for "book"). The entire collection includes minimart and einfart for typesetting articles, and minimbook and simplivre for typesetting books. My original intention in designing them was to write drafts and notes that look simple yet not shabby.
- simplivre has multi-language support, including Chinese (simplified and traditional), English, French, German, Italian, Japanese, Portuguese (European and Brazilian), Russian and Spanish. These languages can be switched seamlessly in a single document. Due to the usage of custom fonts, einfart requires XaLATeX or LuaLATeX to compile.
- This documentation is typeset using simplivre. You can think of it as a short introduction 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

INSTRUCTION

Bef	fore you start								3
1	Usage and examples								5
	/1/ How to load it								5
	/2/ Example - A complete document								5
	2.1 Initialization								6
	2.2 Set the language								6
	2.3 Draft marks								6
	2.4 Theorem-like environments								7
2	On the default fonts								9
3	The options								11
4	Instructions by topic								13
	/1/ Language configuration								
	/2/ Theorems and how to reference them								
	/3/ Define a new theorem-like environment.								
	/4/ Draft mark								
	/5/ Miscellaneous								
	5.1 On the line numbers								16
	5.2 On the footnotes in the title								16
	5.3 On the QED symbols								16
5	Known issues								17

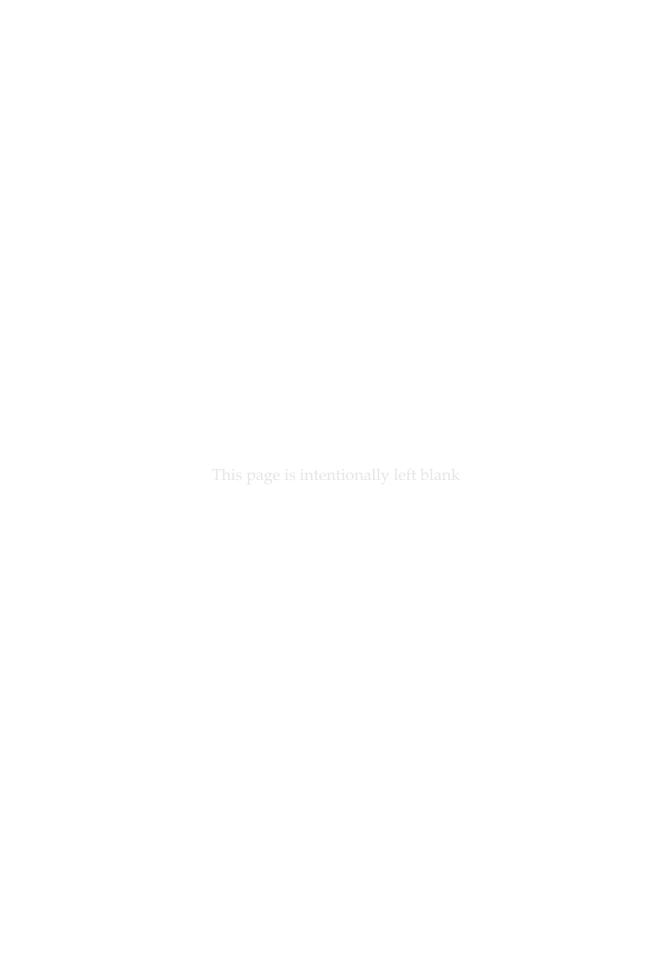
II DEMONSTRATION

6	Heading on Level 0 (chapter)	1
	/1/ Heading on Level 1 (section)	1
	1.1 Heading on Level 2 (subsection)	1
	/2/ Lists	2
	2.1 Example for list (itemize)	2
	2.2 Example for list (enumerate)	
	2.3 Example for list (description)	3

PA	RT	Г	I
ľA	KI		-1

INSTRUCTION

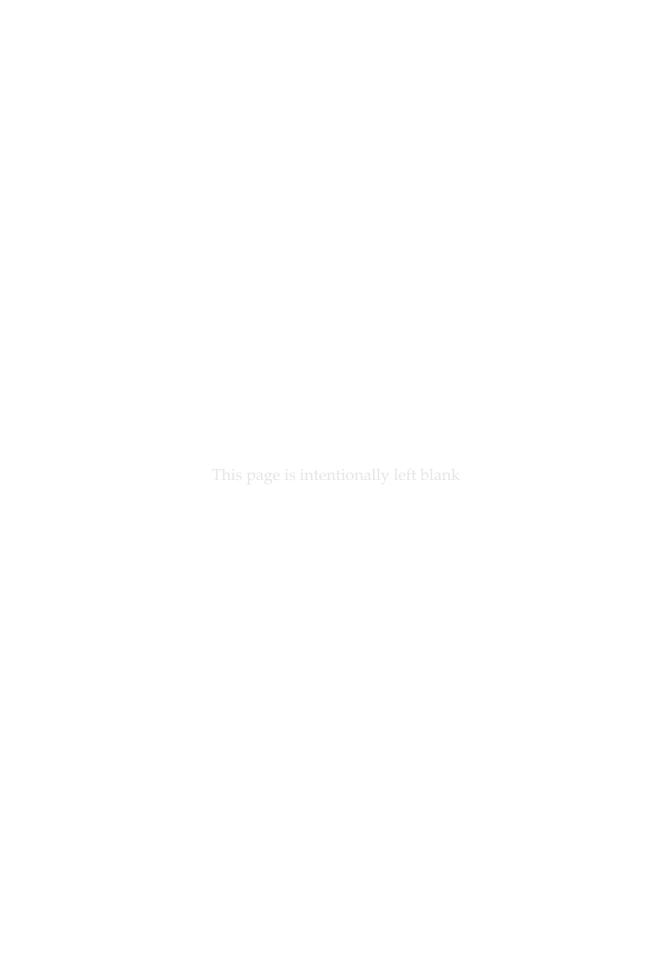
You can add some introductory text here via \parttext\(\lambda text\).





Before you start

- In order to use the package or classes described here, you need to:
- install TeX Live or MikTeX of the latest possible version, and make sure that minimalist and projlib are correctly installed in your TeX system.
- be familiar with the basic usage of LATEX, and knows how to compile your document with pdfLATEX, XATEATEX or LuaLATEX.



Usage and examples

/ 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 X¬IAT_FX or LuaIAT_FX engine to compile.

/ 2 / Example - A complete document

Let's first look at a complete document.

```
1 \documentclass{minimbook}
  \usepackage{ProjLib}
4 \UseLanguage{French}
  \begin{document}
  \frontmatter
10 \begin{titlepage}
      ⟨code for titlepage⟩
12 \end{titlepage}
13
14 \tableofcontents
15
16 \mainmatter
17
```

```
18 \part{\(\rho art \title\)}
19 \parttext{\langle text after part title \rangle}
  \chapter{\chapter title\}
21
22
  \section{\langle section title \rangle}
23
24
  \dnf<Plus de contenu est nécessaire.>
25
26
  \begin{theorem}\label{thm:abc}
27
        Ceci est un théorème.
  \end{theorem}
30 Référence du théorème: \cref{thm:abc}
31
  \backmatter
32
33
34
35
  \end{document}
```

If you find this example a little complicated, don't worry. Let's now look at this example piece by piece.

2.1 | Initialization

```
\documentclass{simplivre} \usepackage{ProjLib}
```

Initialization is straightforward. The first line loads the document class simplivre, and the second line loads the ProjLib toolkit to obtain some additional functionalities.

2.2 | Set the language

```
\UseLanguage{French}
```

- This line indicates that French will be used in the document (by the way, if only English appears in your article, then there is no need to set the language). You can also switch the language in the same way later in the middle of the text. Supported languages include Simplified Chinese, Traditional Chinese, Japanese, English, French, German, Spanish, Portuguese, Brazilian Portuguese and Russian.
- For detailed description of this command and more related commands, please refer to the section on the multi-language support.

2.3 | Draft marks

```
\dnf<\langle some\ hint \rangle>
```

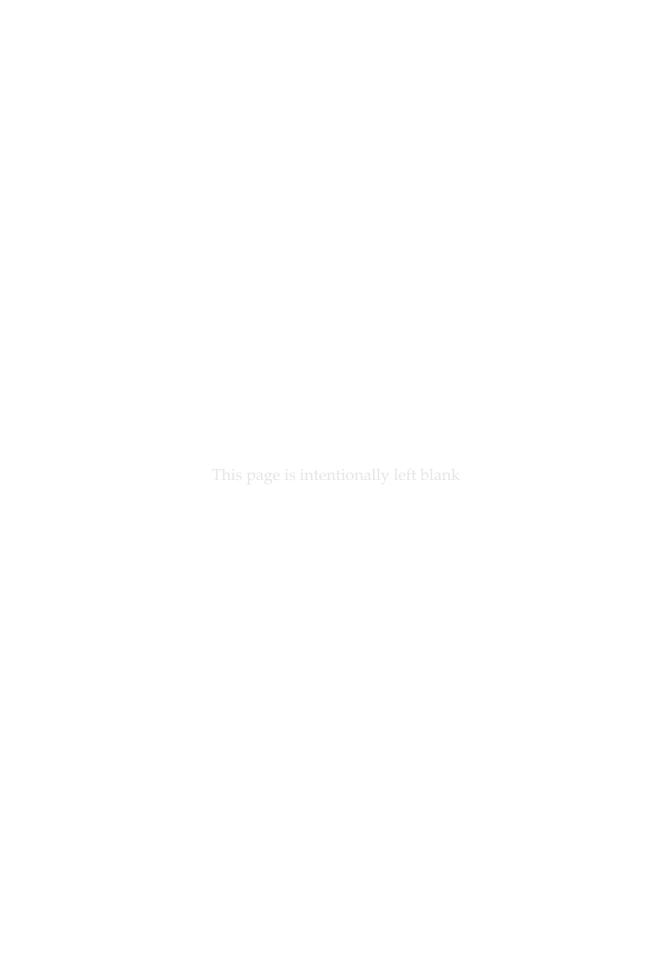
When you have some places that have not yet been finished yet, you can mark them with

this command, which is especially useful during the draft stage.

2.4 | Theorem-like environments

```
\begin{theorem}\label{thm:abc}
    Ceci est un théorème.
\end{theorem}
Référence du théorème: \cref{thm:abc}
```

- Commonly used theorem-like environments have been pre-defined. Also, when referencing
- a theorem-like environment, it is recommended to use $\cref{\langle label \rangle}$ in this way, there
- is no need to explicitly write down the name of the corresponding environment every time.



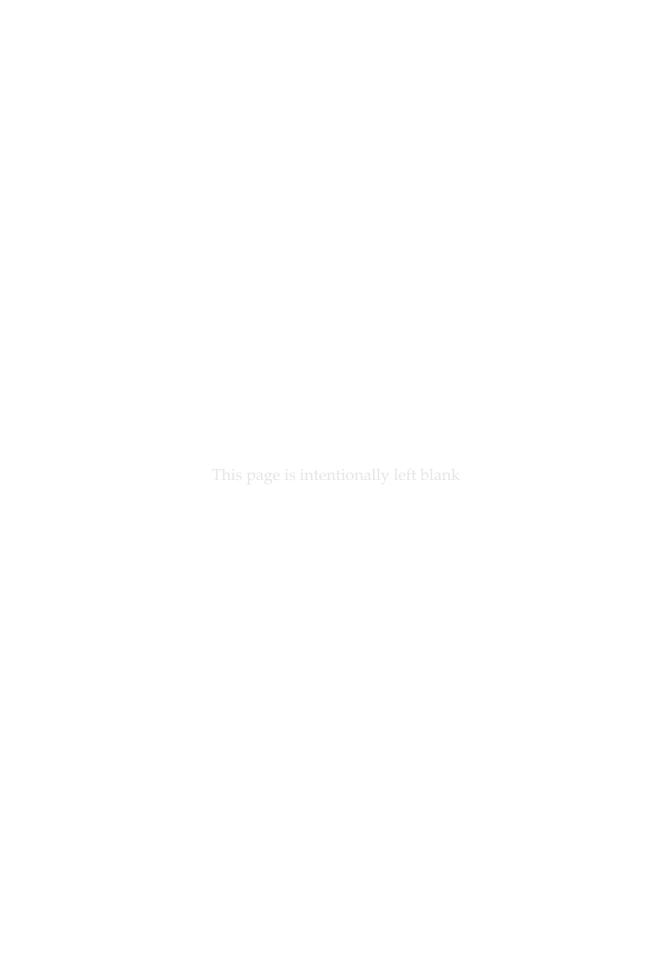
On the default fonts

- By default, einfart uses Palatino Linotype as the English font, FounderType's YouSong and YouHei GBK as the Chinese fonts¹, and partially uses Neo Euler as the math font. Among them, Neo Euler can be downloaded at https://github.com/khaledhosny/euler-otf. The other fonts are not free, you need to purchase and install them on your own.
- English main font. English sans serif font.
- 中文主要字体, 中文无衬线字体
- Math demonstration: α , β , γ , δ , 1, 2, 3, 4, a, b, c, d,

$$\mathrm{li}(x) \coloneqq \int_2^\infty \frac{1}{\log t} \, \mathrm{d}t$$

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

¹For detail, pleaser visit FounderType's website: https://www.foundertype.com.



The options

simplivre offers the following options:

- 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.
 - The language options are optional, mainly for increasing the compilation speed. Without them the result would be the same, only slower.
- 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. When in fast mode, there will be a watermark "DRAFT" to indicate that you are currently in the draft mode.

- a4paper or b5paper
 - Paper size options. The default paper size is 7in \times 10in.
- 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 main title, the titles of all levels and the 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.
- In addition, the commonly used oneside and twoside options are also available. Two-page layout is used by default.

Instructions by topic

\AddLanguageSetting{\(\settings \) \\}

Add ⟨settings⟩ to all supported languages.

 \AddLanguageSetting(\(\language\) name\(\rangle\) \(\language\) Add (settings) to the selected language (language name).

/ 1 / Language configuration

```
simplivre has multi-language support, including Chinese (simplified and traditional), En-
glish, French, German, Italian, Japanese, Portuguese (European and Brazilian), Russian and
Spanish. The language can be selected by the following macros:
• \UseLanguage {\language name\}\) is used to specify the language. The corresponding set-
  ting of the language will be applied after it. It can be used either in the preamble or in the
  main body. When no language is specified, "English" is selected by default.
tings to typeset (content). Compared with \UseLanguage, it will not modify the line spac-
  ing, so line spacing would remain stable when CJK and Western texts are mixed.
(language name) can be (it is not case sensitive, for example, French and french have the
same effect):

    Simplified Chinese: CN, Chinese, SChinese or SimplifiedChinese

• Traditional Chinese: TC, TChinese or TraditionalChinese

    English: EN or English

French: FR or French
• German: DE, German or ngerman

    Italian: IT or Italian

    Portuguese: PT or Portuguese

    Portuguese (Brazilian): BR or Brazilian

    Spanish: ES or Spanish

    Japanese: JP or Japanese

• Russian: RU or Russian
In addition, you can also add new settings to selected language:
```

- For example, \AddLanguageSetting(German){\color{orange}} can make all German
- text displayed in orange (of course, one then need to add \AddLanguageSetting{\color{
- s 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 preset 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, observation, 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 $\cref{\langle label \rangle}$. In
- this way, there is no need to explicitly write down the name of the corresponding environ-
- ment every time.

EXAMPLE

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

will produce

Definition $4.1 \mid (Strange things)$ 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 4.1.
After using \UseLanguage{French}, a theorem will be displayed as:

Тне́окѐме 4.2 | (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 4.1 and Théorème 4.2. 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:
- \NameTheorem [\language name\rangle] {\language name of environment\rangle} {\language name string\rangle}
- For *(language name)*, please refer to the section on language configuration. When *(language name)* is not specified, the name will be set for all supported languages. In addition, envi-

ronments with or without asterisk share the same name, therefore, \NameTheorem{envname *}{} has the same effect as \NameTheorem{envname}{}.
And then define this environment in one of following five ways:
• \CreateTheorem*{\(\text{name of environment} \)}
 Define an unnumbered environment (name of environment)
• \CreateTheorem{\(\lame\) of environment\)}
 Define a numbered environment (name of environment), numbered in order 1,2,3,
• \CreateTheorem{\(\lame\) of environment\)} [\(\lambda\) like\)]
 Define a numbered environment (name of environment), which shares the counter (numbered like)
• \CreateTheorem{\(\lame\) of environment\)}<\(\numbered\) within\>
 Define a numbered environment (name of environment), numbered within the counter (numbered within)
• \CreateTheorem{\(\lame\) of environment\)} (\(\lame\) (\(\lame\) environment\(\rangle\))
\CreateTheorem*{\(\lambda\) ame of environment\)} (\(\lambda\) is ted environment\(\rangle\))
 Identify ⟨name of environment⟩ with ⟨existed environment⟩ or ⟨existed environment⟩*.
 This method is usually useful in the following two situations:
1. To use a more concise name. For example, with \CreateTheorem{thm}(theorem),
one can then use the name thm to write theorem.
To remove the numbering of some environments. For example, one can remove the numbering of 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.
Here is an example. The following code:
\NameTheorem[EN]{proofidea}{Idea}
\CreateTheorem*{proofidea*}
\CreateTheorem{proofidea} <section></section>
defines an unnumbered environment proofidea* and a numbered environment proofidea (numbered within section) respectively. They can be used in English context. The effect is as follows:
Idea The proofidea* environment.
Idea 3.1 The proofidea environment.

/ 4 / Draft mark

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

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

Similarly, there is \needgraph:

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

Il manque une image ici #3

/ 5 / Miscellaneous

5.1 | On the line numbers

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

5.2 On the footnotes in the title

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

```
first write \mbox{\protect\footnotemark},
then add \footnotetext{...} afterwards.
```

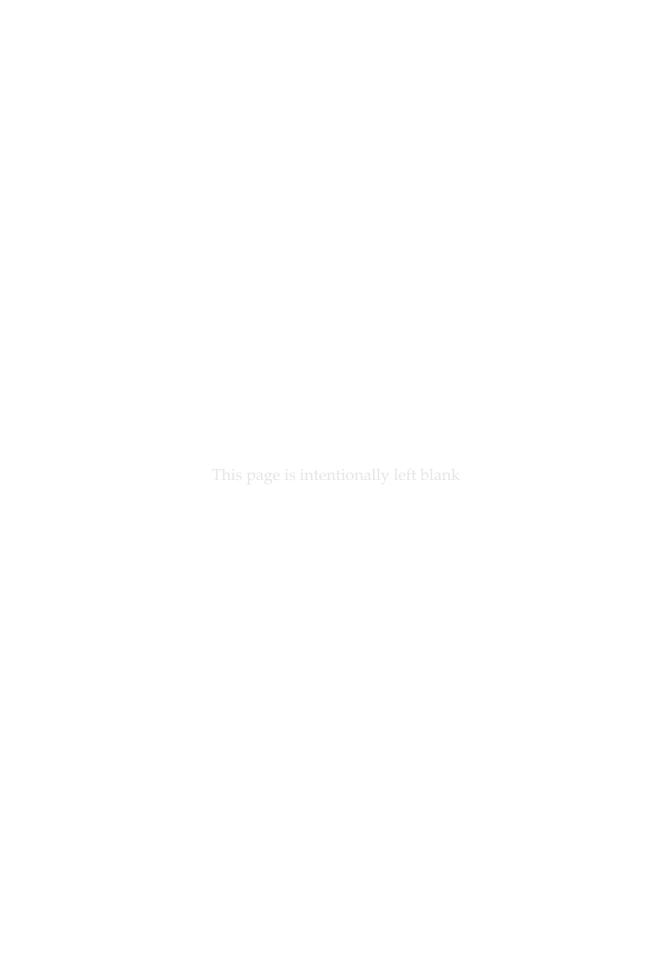
This is a disadvantage brought about by the underline decoration of the title.

5.3 | On the QED symbols

Since the font in the theorem-like environments is the same as that of the main text, in order to indicate where the environments end, a hollow QED symbol is placed at the end of the theorem-like environments. However, if your theorem ends with an equation or list (itemize, enumerate, description, etc.), this symbol cannot be automatically placed in the correct position. In this case, you need to manually add a \quad \quad \quad \quad \text{qedhere} at the end of your equation or the last entry of your list to make the QED symbol appear at the end of the line.

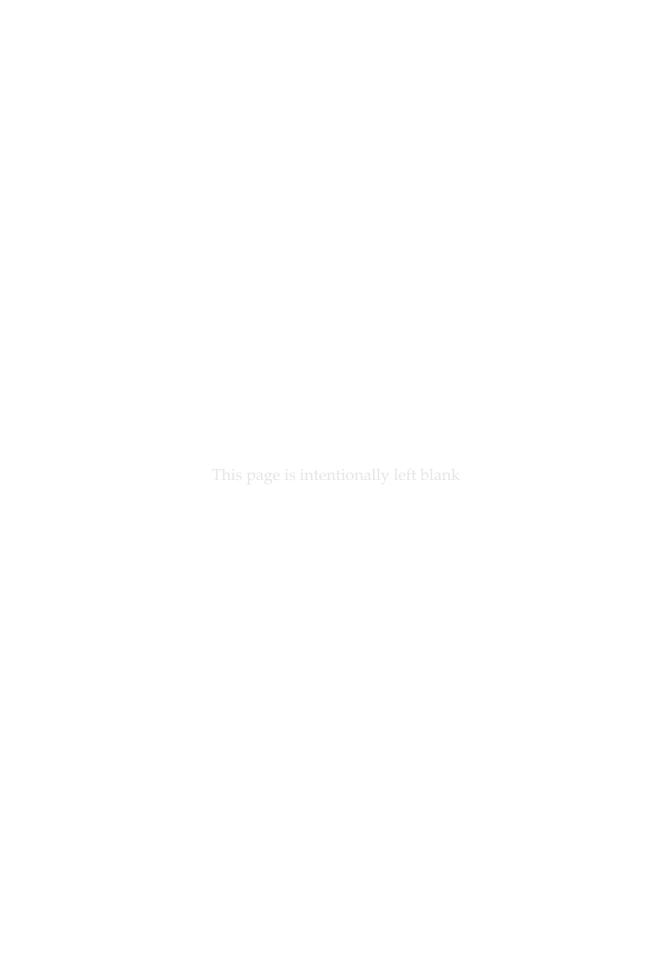
Known issues

- The font settings are still not perfect.
- Since many features are based on the ProjLib toolkit, minimalist (and hence minimart, einfart and minimbook, simplivre) inherits all its problems. For details, please refer to the "Known Issues" section of the ProjLib documentation.
- The error handling mechanism is incomplete: there is no corresponding error prompt when some problems occur.
- There are still many things that can be optimized in the code.



PART II

DEMONSTRATION



Heading on Level 0 (chapter)

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

Heading on Level 1 (section)

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

1.1 | Heading on Level 2 (subsection)

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

Heading on Level 3 (subsubsection)

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

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

/ 2 / Lists

2.1 | Example for list (itemize)

- First item in a list
- Second item in a list
- Third item in a list
- Fourth item in a list
- Fifth item in a list

Example for list (4*itemize)

- First item in a list
 - First item in a list
 - * First item in a list
 - · First item in a list
 - · Second item in a list
 - * Second item in a list
 - Second item in a list
- Second item in a list

2.2 | Example for list (enumerate)

- 4 1. First item in a list
 - 2. Second item in a list
 - 3. Third item in a list
 - 4. Fourth item in a list
 - 5. Fifth item in a list

*Example for list (4*enumerate)*

- 1. First item in a list
 - a) First item in a list
 - (i) First item in a list
 - A. First item in a list
 - B. Second item in a list
 - (ii) Second item in a list

b) Second item in a list 2. Second item in a list 2.3 | Example for list (description) First item in a list Second item in a list Third item in a list Fourth item in a list Fifth item in a list Example for list (4*description) First item in a list Second item in a list