

# colorist, WRITE YOUR ARTICLES OR BOOKS IN A COLORFUL WAY

JINWEN XU  
May 2021, Beijing

## ABSTRACT

colorist is a series of styles and classes for you to typeset your articles or books in a colorful manner. My original intention in designing this series is to write drafts and notes that look colorful yet not dazzling.

The entire collection includes `colorist.sty`, which is the main style shared by all of the following classes; `colorart.cls` for typesetting articles and `colorbook.cls` for typesetting books. They compile with any major  $\text{\TeX}$  engine, with native support to English, French and German typesetting via `\UseLanguage` (see the instruction below for detail).

You can also found `lebhart` and `beaulivre` on CTAN. They are the enhanced version of `colorart` and `colorbook` with unicode support. With this, they can access to more beautiful fonts, and also have native support for Chinese, Japanese and Russian typesetting. On the other hand, they need to be compiled with  $\text{\XeLaTeX}$  or  $\text{\LuaLaTeX}$  (not  $\text{\pdfLaTeX}$ ).

This documentation is typeset using `colorart` (with the option `allowbf`). You can think of it as a short introduction and demonstration.

## CONTENTS

1	Initialization	2
1.1	How to load it	2
1.2	Options	2
2	Some instructions	2
2.1	Language configuration	2
2.2	Theorems and how to reference them	3
2.3	Define a new theorem-like environment	4
2.4	Draft mark	5
2.5	On the title, abstract and keywords	6
3	Known issues	6
4	Document templates	7
4.1	The standard way	7
4.2	The $\mathcal{AMS}$ way	8

### Remind

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](#).

## 1 INITIALIZATION

### 1.1 HOW TO LOAD IT

You can directly use `colorart` or `colorbook` as your document class. In this way, you can directly begin writing your document, without having to worry about the configurations.

---

```
\documentclass{colorart} or \documentclass{colorbook}
```

---

And of course, you can also use the default classes `article` or `book`, and load the `colorist` package. This way, only the basic styles are set, and you can thus use your preferred fonts and page layout. All the features mentioned in this article are provided.

---

```
\usepackage{colorist}
```

---

### 1.2 OPTIONS

`colorist` 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.
- `allowbf`
  - Allow boldface. When this option is enabled, the title, titles of all levels and names of theorem-like environments will be bolded.

#### Remind

- 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.

Additionally, `colorart` and `colorbook` offers the following options:

- `a4paper` or `b5paper`
  - Optional paper size. The default paper size is 8.5in × 11in.

## 2 SOME INSTRUCTIONS

Many of the features described next are provided by the `ProjLib` toolkit. Only the basic usage is mentioned here. For more details, please refer to its user documentation.

### 2.1 LANGUAGE CONFIGURATION

`colorist` has multi-language support, including simplified Chinese, traditional Chinese, English, French, German, Japanese, and Russian. The language can be selected by the following macros:

- `\UseLanguage{<language name>}` is used to specify the language. The corresponding setting 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.
- `\UseOtherLanguage{<language name>}{<content>}`, which uses the specified language settings to typeset `<content>`. Compared with `\UseLanguage`, it will not modify the line spacing, 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
- Japanese: JP or Japanese
- Russian: RU or Russian

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

- `\AddLanguageSetting{<settings>}`
  - 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).

## 2.2 THEOREMS AND HOW TO REFERENCE THEM

Environments such as definitions and theorems 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 display of these environments will change according to the current language.

When referencing a theorem-like environment, it is recommended to use clever reference `\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 2.1** (Strange things) This is the definition of some strange objects.

`\cref{def: strange}` will be displayed as: **DEFINITION 2.1**.

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

**THÉORÈME 2.2** (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 2.1** and **THÉORÈME 2.2**. If you want the name of the theorem to match the current context when referencing, you can add `regionalref` to the global options.

The following are the main styles of theorem-like environments:

**THEOREM 2.3** Theorem style: theorem, proposition, lemma, corollary, ...

*Proof* | Proof style



*Remark style*



**CONJECTURE 2.4** Conjecture style

**EXAMPLE** Example style: example, fact, ...

**PROBLEM 2.5** Problem style: problem, question, ...

For aesthetics, adjacent definitions will be connected together automatically:

**DEFINITION 2.6** First definition.

**DEFINITION 2.7** Second definition.

## 2.3 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⟩]{⟨name of environment⟩}{⟨name string⟩}`

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, environments 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*{⟨name of environment⟩}`
  - Define an unnumbered environment *⟨name of environment⟩*
- `\CreateTheorem{⟨name of environment⟩}`
  - Define a numbered environment *⟨name of environment⟩*, numbered in order 1,2,3,...
- `\CreateTheorem{⟨name of environment⟩}[⟨numbered like⟩]`
  - Define a numbered environment *⟨name of environment⟩*, which shares the counter *⟨numbered like⟩*
- `\CreateTheorem{⟨name of environment⟩}<⟨numbered within⟩>`
  - Define a numbered environment *⟨name of environment⟩*, numbered within the counter *⟨numbered within⟩*

- `\CreateTheorem{⟨name of environment⟩}(⟨existed environment⟩)`  
`\CreateTheorem*{⟨name of environment⟩}(⟨existed environment⟩)`
  - 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`.
    2. To remove the numbering. For example, one can remove the numbering of the `remark` environment with `\CreateTheorem{remark}(remark*)`.


#### Remind


It uses `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}<subsection>
```

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

**Idea** | The `proofidea*` environment. 

**Idea 2.3.1** | The `proofidea` environment. 

## 2.4 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` :

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

`A graph is needed here #1`

or

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

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`

## 2.5 ON THE TITLE, ABSTRACT AND KEYWORDS

colorart has both the features of standard classes and that of the  $\mathcal{AMS}$  classes.

Therefore, the title part can either be written in the usual way, in accordance with the standard class article:

---

```
\title{\langle title \rangle}
\author{\langle author \rangle \thanks{\langle text \rangle}}
\date{\langle date \rangle}
\maketitle
\begin{abstract}
  \langle abstract \rangle
\end{abstract}
\begin{keyword}
  \langle keywords \rangle
\end{keyword}
```

---

or written in the way of  $\mathcal{AMS}$  classes:

---

```
\title{\langle title \rangle}
\author{\langle author \rangle}
\thanks{\langle text \rangle}
\address{\langle address \rangle}
\email{\langle email \rangle}
\date{\langle date \rangle}
\keywords{\langle keywords \rangle}
\subjclass{\langle subclass \rangle}
\begin{abstract}
  \langle abstract \rangle
\end{abstract}
\maketitle
```

---

The author information can contain multiple groups, written as:

---

```
\author{\langle author 1 \rangle}
\address{\langle address 1 \rangle}
\email{\langle email 1 \rangle}
\author{\langle author 2 \rangle}
\address{\langle address 2 \rangle}
\email{\langle email 2 \rangle}
...
```

---

Among them, the mutual order of `\address`, `\curraddr`, `\email` is not important.

## 3 KNOWN ISSUES

- The font settings are still not perfect.
- Since many features are based on the ProjLib toolkit, colorist 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.

## 4 DOCUMENT TEMPLATES

### 4.1 THE STANDARD WAY

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

---

```
\documentclass{colorart}
\usepackage{PJLtoolkit} % Load ProjLib toolkit

\UseLanguage{French} % Use French from here

\begin{document}

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

\maketitle

\begin{abstract}
  Ceci est un résumé. \dnf<Plus de contenu est nécessaire.>
\end{abstract}
\begin{keyword}
  AAA, BBB, CCC, DDD, EEE
\end{keyword}

\section{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}
```

---

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}{PJLtoolkit} % Load ProjLib toolkit
```

---

## 4.2 THE $\mathcal{AMS}$ WAY

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

---

```
\documentclass{colorart}
\usepackage{PJLtoolkit} % 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{...}

\begin{abstract}
  Ceci est un résumé. \dnf<Plus de contenu est nécessaire.>
\end{abstract}

\maketitle

\section{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}
```

---

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}{PJLtoolkit} % Load ProjLib toolkit
```

---