

# LA BOÎTE À OUTILS ProjLib

## MANUEL D'UTILISATION

JINWEN XU

ProjLib@outlook.com

juillet 2021, à Pékin

### RÉSUMÉ

La boîte à outils ProjLib est conçue pour simplifier la préparation avant d'écrire des documents  $\text{\LaTeX}$ . Avec le package ProjLib ajouté, vous n'avez plus besoin de créer des environnements de type théorème, ni de configurer les paramètres multilingues. De plus, une série de fonctionnalités auxiliaires sont introduites.

### TABLE DES MATIÈRES

Avant de commencer . . . . .	1	4 Les composants . . . . .	5
1 Introduction . . . . .	1	4.1 PJLamssim : écrire de la manière $\mathcal{AMS}$ . . . . .	5
2 Exemple d'utilisation . . . . .	2	4.2 PJLauthor : bloc auteur amélioré . . . . .	5
2.1 Comment l'ajouter . . . . .	2	4.3 PJLdate : traitement de date-heure . . . . .	5
2.2 Exemple - Un document complet . . . . .	2	4.4 PJLdraft : marques de brouillon . . . . .	5
2.2.1 Initialisation . . . . .	3	4.5 PJLlang : support multilingue . . . . .	6
2.2.2 Choisir la langue . . . . .	3	4.6 PJLlogo : le logo ProjLib . . . . .	7
2.2.3 Le titre et les informations de l'auteur . . . . .	3	4.7 PJLmath : symboles et raccourcis mathématiques . . . . .	7
2.2.4 Marques de brouillon . . . . .	4	4.8 PJLpaper : configuration papier . . . . .	8
2.2.5 Environnements de type théorème . . . . .	4	4.9 PJLthm : environnements de type théorème avec référence intelligente et support multilingue . . . . .	8
3 Options du package principal . . . . .	4	5 Problèmes connus . . . . .	10

### AVANT DE COMMENCER

Pour utiliser la boîte à outils, vous devez :

- installer TeX Live ou MikTeX de la dernière version possible, et vous assurer que projlib est correctement installé dans votre système  $\text{\TeX}$ .
- être familiarisé avec l'utilisation de base de  $\text{\LaTeX}$ , et savoir comment compiler vos documents avec pdf $\text{\LaTeX}$ , x $\text{\LaTeX}$  ou Lua $\text{\LaTeX}$ .

### 1 INTRODUCTION

Le nom ProjLib peut être considéré comme l'abréviation de *Project Library* en anglais ou de *Projet Libre* en français (l'auteur préfère l'interprétation française). Son objectif principal est de fournir un support multilingue et des environnements de type théorème avec des références intelligentes. De plus, certaines fonctionnalités supplémentaires sont fournies, telles que le bloc auteur amélioré, les marques de brouillon, les symboles mathématiques et les raccourcis, etc.

La boîte à outils ProjLib est composée du package principal ProjLib et d'une série de composants dont les noms commencent par l'abréviation « PJJ ». Vous pouvez apprendre à l'utiliser à travers les exemples d'utilisation dans la section suivante.

## 2 EXEMPLE D'UTILISATION

### 2.1 COMMENT L'AJOUTER

Ajoutez simplement la ligne suivante à votre préambule :

```
\usepackage{ProjLib}
```

#### ATTENTION

Comme `cleveref` est utilisé en interne, `ProjLib` doit être placé après `varioref` et `hyperref`.

### 2.2 EXEMPLE - UN DOCUMENT COMPLET

Regardons d'abord un document complet.

```
1 \documentclass{article}
2 \usepackage[a4paper,margin=.75in]{geometry}
3 \usepackage[hidelinks]{hyperref}
4 \usepackage[palatino]{ProjLib} % Load the toolkit and use font Palatino
5
6 \UseLanguage{French} % Use French from here
7
8 \begin{document}
9
10 \title{<title>}
11 \author{<author>}
12 \date{\PLdate{2022-04-01}}
13
14 \maketitle
15
16 \begin{abstract}
17   <abstract text> \dnf<<some hint>>
18 \end{abstract}
19
20 \section{Un théorème}
21
22 \begin{theorem}\label{thm:abc}
23   Ceci est un théorème.
24 \end{theorem}
25
26 Référence du théorème: \cref{thm:abc} % It is recommended to use clever reference
27
28 \end{document}
```

Si vous trouvez cela un peu compliqué, ne vous inquiétez pas. Examinons maintenant cet exemple pièce par pièce.

### 2.2.1 Initialisation

---

```
\documentclass{article}
\usepackage[a4paper,margin=.75in]{geometry}
\usepackage[hidelinks]{hyperref}
\usepackage{palatino}{ProjLib}
```

---

Dans les classes standard, il suffit généralement de configurer la taille de la page, les liens hypertexte et d'ajouter ProjLib avant de commencer à écrire le document. L'option de police palatino de ProjLib est utilisée ici. Pour toutes les options disponibles de ProjLib, veuillez vous référer à la section suivante.

Bien sûr, vous pouvez également utiliser la classe de document amsart, les configurations sont les mêmes.

### 2.2.2 Choisir la langue

---

```
\UseLanguage{French}
```

---

Cette ligne indique que le français sera utilisé dans le document (d'ailleurs, si seul l'anglais apparaît dans votre article, alors il n'est pas nécessaire de choisir la langue). Vous pouvez également changer de langue de la même manière plus tard au milieu du texte. Les langues prises en charge sont les suivantes : chinois simplifié, chinois traditionnel, japonais, anglais, français, allemand, espagnol, portugais, portugais brésilien et russe<sup>1</sup>.

Pour une description détaillée de cette commande et d'autres commandes associées, veuillez vous référer à la section sur le support multilingue.

### 2.2.3 Le titre et les informations de l'auteur

---

```
\title{\<title>}
\author{\<author>}
\date{\PLdate{2022-04-01}}
```

---

Cette partie est le titre et le bloc d'informations de l'auteur. L'exemple montre l'utilisation la plus fondamentale, mais en fait, vous pouvez également écrire comme :

```
\author{\<author 1>}
\address{\<address 1>}
\email{\<email 1>}
\author{\<author 2>}
\address{\<address 2>}
\email{\<email 2>}
...
```

---

De plus, si la simulation d' $\mathcal{A}\mathcal{M}\mathcal{S}$  est activée<sup>2</sup>, alors vous pouvez également écrire à la manière  $\mathcal{A}\mathcal{M}\mathcal{S}$  (la manière originale fonctionne encore). Dans ce cas, la ligne qui introduit ProjLib doit être écrite comme :

```
\usepackage[amsmath,palatino]{ProjLib}
```

---

Et en conséquence, vous pourrez également utiliser ces macros :

```
\dedicatory{\<dedicatory>}
\subjclass{*****}
\keywords{\<keywords>}
```

---

---

<sup>1</sup>Cependant, vous devez ajouter vous-même l'encodage et les polices de la langue correspondante. Par exemple, pour le chinois, vous devrez peut-être ajouter le package ctex et choisir les polices. Pour rappel, vous pouvez essayer les classes de documents einart ou lebart de l'auteur, dans lesquelles les paramètres correspondants ont été effectués. Pour les détails, exécutez `texdoc minimalist` ou `coloriste texdoc` en ligne de commande.

<sup>2</sup>Ceci est réalisé par le module PJLamssim. Étant donné que ce module modifie certaines macros internes de  $\text{\LaTeX}$ , il peut provoquer des conflits avec certains packages ou classes de documents, et donc il n'est pas activé par défaut.

De plus, vous pouvez également placé le résumé avant `\maketitle`, comme requis dans les classes  $\mathcal{M}\mathcal{S}$  :

---

```
\begin{abstract}
  \langle abstract text \rangle
\end{abstract}
\maketitle
```

---

#### 2.2.4 Marques de brouillon

---

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

---

Lorsque vous avez des endroits qui ne sont pas encore finis, vous pouvez les marquer avec cette commande, ce qui est particulièrement utile lors de la phase de brouillon.

#### 2.2.5 Environnements de type théorème

---

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

---

Les environnements de type théorème couramment utilisés ont été prédéfinis. De plus, lors du référencement d'un environnement de type théorème, il est recommandé d'utiliser `\cref{\langle label \rangle}` — de cette manière, il ne serait pas nécessaire d'écrire explicitement le nom de l'environnement correspondant à chaque fois.

### 3 OPTIONS DU PACKAGE PRINCIPAL

ProjLib a les options suivantes :

- `draft` ou `fast`
  - Mode brouillon. La fonctionnalité sera réduite de manière appropriée pour obtenir une vitesse de compilation plus rapide, recommandée à utiliser pendant la phase de brouillon.
- `palatino`, `times`, `garamond`, `biolinum` | `useosf`
  - Options de police. Comme les noms l'indiquent, la police avec le nom correspondant sera utilisée.
  - L'option `useosf` est pour activer les chiffres à l'ancienne.
- `nothms`, `nothmnum`, `regionalref`
  - Options de `PJLthm`, veuillez vous référer à la section sur ce package pour plus de détails.
- `amsfashion`
  - Permet à l'utilisateur d'écrire à la manière  $\mathcal{M}\mathcal{S}$ . Identique à l'option `amssim`.
- `author`, `amssim`
  - Utilisez `PJLauthor` ou `PJLamssim`. Pour plus d'informations sur leurs fonctionnalités, consultez la section sur les packages correspondants.

De plus, il existe également certaines options des composants qui doivent être passées en tant qu'options globales de votre classe de document, telles que les options de langue de `PJLlang` comme `EN / english / English`, `FR / french / French` etc., et les options de papier de `PJLpaper` comme `paperstyle` et `preview`. Pour plus d'informations, veuillez vous référer aux sections correspondantes.

## 4 LES COMPOSANTS

### 4.1 PJLAMSSIM : ÉCRIRE DE LA MANIÈRE $\mathcal{AMS}$

PJLamssim est utilisé pour simuler certaines fonctionnalités de la classe `amsart` dans une classe standard, notamment :

- les macros `\address`, `\curraddr`, `\email` et `\dedicatory` (les trois premiers sont fournis par PJLauthor) ;
- la macro `\keywords` ;
- la macro `\subjclass` ;
- `\thanks` peut être écrit en dehors de `\author` ;
- L’environnement `abstract` peut être placé avant `\maketitle`.

Ces modifications n’auraient lieu que dans les classes standard. Dans les classes  $\mathcal{AMS}$ , PJLamssim n’a aucun effet.

### 4.2 PJLAUTHOR : BLOC AUTEUR AMÉLIORÉ

PJLauthor propose les macros `\address`, `\curraddr` et `\email`, et vous permet de saisir plusieurs groupes d’informations sur l’auteur. L’utilisation standard est comme ceci :

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

L’ordre mutuel de `\address`, `\curraddr` et `\email` n’est pas important.

### 4.3 PJLDATE : TRAITEMENT DE DATE-HEURE

PJLdate propose la macro `\PLdate{yyyy-mm-dd}` (ou `\PJLdate{yyyy-mm-dd}`) pour convertir `⟨yyyy-mm-dd⟩` dans le format de date de la langue actuellement sélectionnée. Par exemple, dans le contexte français actuel, `\PLdate{2022-04-01}` deviendrait « 1<sup>er</sup> avril 2022 », tandis que dans le contexte anglais « April 1, 2022 ».

Pour plus de détails sur la façon de sélectionner une langue, veuillez vous référer à la section sur PJLlang.

### 4.4 PJLDRAFT : MARQUES DE BROUILLON

PJLdraft propose les macros suivantes :

- `\dnf` ou `\dnf<...>`. L’effet est : `Pas encore fini #1` ou `Pas encore fini #2: ...`.  
Le texte à l’intérieur changera en fonction de la langue actuelle. Par exemple, il sera affiché sous la forme `To be finished #3` en mode anglais.
- `\needgraph` ou `\needgraph<...>`. L’effet est :

`Il manque une image ici #1`

ou

`Il manque une image ici #2: ...`

Le texte de l’invite change en fonction de la langue actuelle. Par exemple, en mode anglais, il sera affiché sous la forme

`A graph is needed here #3`

Pour plus de détails sur la façon de sélectionner une langue, veuillez vous référer à la section sur PJLlang.

## 4.5 PJLLANG : SUPPORT MULTILINGUE

PJLlang offre le support multilingue, notamment : chinois simplifié, chinois traditionnel, anglais, français, allemand, japonais et russe (parmi eux, le chinois, le japonais et le russe requièrent des moteurs  $\text{T}_{\text{E}}\text{X}$  et des polices appropriés).

PJLlang fournit des options de langue. Les noms de ces options ont trois types, qui sont des abréviations (comme EN), des minuscules (comme english) et des majuscules (comme English). Pour les noms d'options d'une langue spécifique, veuillez vous référer à *<language name>* ci-dessous. Parmi eux, la première langue spécifiée *<first language>* sera considérée comme langue par défaut, ce qui équivaut à spécifier `\UseLanguage{<first language>}` au début de votre document.

### ASTUCE

Il est recommandé d'utiliser ces options de langue et de les passer en tant qu'options globales. De cette façon, seules les langues spécifiées sont configurées, économisant ainsi la mémoire  $\text{T}_{\text{E}}\text{X}$  et améliorant considérablement la vitesse de compilation.

La langue peut être sélectionnée par les macros suivantes :

- `\UseLanguage{<language name>}` est utilisé pour spécifier la langue. Le réglage correspondant de la langue sera appliqué après celui-ci. Il peut être utilisé soit dans le préambule ou dans le texte. Lorsqu'aucune langue n'est spécifiée, « English » est sélectionné par défaut.
- `\UseOtherLanguage{<language name>}{<content>}`, qui utilise les paramètres de langue spécifiés pour composer *<content>*. Par rapport à `\UseLanguage`, il ne modifiera pas l'interligne, donc l'interligne restera stable lorsque les textes CJK et occidentaux sont mélangés.

*<language name>* peut être (il n'est pas sensible à la casse, par exemple, French et french ont le même effet) :

- chinois simplifié : CN, Chinese, SChinese ou SimplifiedChinese
- chinois traditionnel : TC, TChinese ou TraditionalChinese
- anglais : EN ou English
- français : FR ou French
- allemand : DE, German ou ngerman
- italien : IT ou Italian
- portugais : PT ou Portuguese
- portugais (brésilien) : BR ou Brazilian
- espagnol : ES ou Spanish
- japonais : JP ou Japanese
- russe : RU ou Russian

De plus, vous pouvez également ajouter de nouveaux paramètres à la langue sélectionnée :

- `\AddLanguageSetting{<settings>}`
  - Ajoutez *<settings>* à toutes les langues prises en charge.
- `\AddLanguageSetting(<language name>){<settings>}`
  - Ajoutez *<settings>* à la langue *<language name>* sélectionnée.

Par exemple, `\AddLanguageSetting(German){\color{orange}}` peut rendre tout le texte allemand affiché en orange (bien sûr, il faut alors ajouter `\AddLanguageSetting{<language name>}{\color{black}}` afin de corriger la couleur du texte dans d'autres langues).

## 4.6 PJLLOGO : LE LOGO ProjLib

PJLlogo propose la macro `\ProjLib` pour dessiner le logo, qui ressemble à ProjLib. Elle est similaire aux macros de texte ordinaires et peut être utilisée avec différentes macros de taille de texte :

<code>\tiny :</code>	ProjLib
<code>\scriptsize :</code>	ProjLib
<code>\footnotesize :</code>	ProjLib
<code>\normalsize :</code>	ProjLib
<code>\large :</code>	ProjLib
<code>\Large :</code>	ProjLib
<code>\LARGE :</code>	ProjLib
<code>\huge :</code>	ProjLib
<code>\Huge :</code>	ProjLib

## 4.7 PJLMATH : SYMBOLES ET RACCOURCIS MATHÉMATIQUES

PJLmath propose les raccourcis suivants :

- i) `\mathfrak{.}`  $\longrightarrow$  `\mf.` ou `\frac.` . Par exemple, `\mfA` (ou `\mf{A}`) a le même effet que `\mathfrak{A}`. Cela fonctionne à la fois pour l'alphabet majuscule et minuscule, produisant :

$\text{abcdefghijklmnopqrstuvwxyz}$   
 $\text{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

- ii) `\mathbb{.}`  $\longrightarrow$  `\bb.` . Cela ne fonctionne que pour l'alphabet majuscule et le nombre 1.

$\text{ABCDEFGHIJKLMNOPQRSTUVWXYZ1}$

Il y a aussi des commandes spéciales pour les structures algébriques bien connues : `\N`, `\Z`, `\Q`, `\R`, `\C`, `\F`, `\A`.

$\text{NZQRCFA}$

- iii) `\mathcal{.}`  $\longrightarrow$  `\mc.` or `\cal.` . Cela ne fonctionne que pour l'alphabet majuscule.

$\text{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

- iv) `\mathscr{.}`  $\longrightarrow$  `\ms.` or `\scr.` . Cela ne fonctionne que pour l'alphabet majuscule.

$\text{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

De plus, PJLmath fournit également des symboles mathématiques qui ne sont pas inclus par défaut avec L<sup>A</sup>T<sub>E</sub>X.

<code>\abs</code>	<code>\abs{a}</code> $\rightarrow  a $	symbole de valeur absolue
<code>\norm</code>	<code>\norm{a}</code> $\rightarrow \ a\ $	symbole de norme
<code>\injection</code>	<code>\injection</code> $\rightarrow \hookrightarrow$	symbole de flèche pour l'injection
<code>\surjection</code>	<code>\surjection</code> $\rightarrow \twoheadrightarrow$	symbole de flèche pour la surjection
<code>\bijection</code>	<code>\bijection</code> $\rightarrow \xrightarrow{\sim}$	symbole de flèche pour la bijection
<code>\legendre</code>	<code>\legendre{a}{p}</code> $\rightarrow \left(\frac{a}{p}\right)$	Symbole Legendre
	<code>\legendre[z]{a}{p}</code> $\rightarrow \left(\frac{a}{p}\right)_z$	

## 4.8 PJLPAPER : CONFIGURATION PAPIER

PJLPaper est principalement utilisé pour ajuster la couleur du papier. Il a les options suivantes :

- `paperstyle = <paper style name>`
  - Définit le style de couleur du papier. Les options disponibles pour *<paper style name>* sont : `yellow`, `dark` et `nord`.
- `yellowpaper, darkpaper, nordpaper`
  - Identique à `paperstyle` avec le *<paper style name>* correspondant spécifié.
- `preview`
  - Mode aperçu. Recadrez les bords blancs du fichier pdf pour faciliter la lecture.

Il est recommandé de les passer comme options globales de la classe de document. De cette façon, les paramètres du papier seraient clairs en un coup d’œil.

## 4.9 PJLTHM : ENVIRONNEMENTS DE TYPE THÉORÈME AVEC RÉFÉRENCE INTELLIGENTE ET SUPPORT MULTILINGUE

PJLthm offre la configuration d’environnements de type théorème. Il a l’option suivante :

- `nothms`
  - Les environnements de type théorème ne seront pas définis. Vous pouvez utiliser cette option si vous souhaitez appliquer vos propres styles de théorème.
- `nothmnum`
  - Les environnements de type théorème ne seront pas numérotés.
- `regionalref`
  - Lors du référencement, le nom de l’environnement de type théorème changera avec la langue actuelle (par défaut, le nom restera toujours le même ; par exemple, lors du référencement d’un théorème écrit dans le contexte anglais, même si l’on est actuellement dans le contexte français, il sera toujours affiché comme « Theorem »). En mode `fast`, cette option est automatiquement activée.

Les environnements prédéfinis incluent : `assumption`, `axiom`, `conjecture`, `convention`, `corollary`, `definition`, `definition-proposition`, `definition-theorem`, `example`, `exercise`, `fact`, `hypothesis`, `lemma`, `notation`, `observation`, `problem`, `property`, `proposition`, `question`, `remark`, `theorem`, et la version non numérotée correspondante avec un astérisque `*` dans le nom. Les titres changeront avec la langue actuelle. Par exemple, `theorem` sera affiché comme « Theorem » en mode anglais et « Théorème » en mode français. Pour plus de détails sur la façon de sélectionner une langue, veuillez vous référer à la section sur PJLlang.

### ASTUCE

Lors du référencement d’un environnement de type théorème, il est recommandé d’utiliser `\cref{<label>}`. De cette façon, il n’est pas nécessaire d’écrire explicitement le nom de l’environnement correspondant à chaque fois.

Si vous avez besoin de définir un nouvel environnement de type théorème, vous devez d’abord définir le nom de l’environnement dans le langage à utiliser :

- `\NameTheorem[<language name>]{<name of environment>}{<name string>}`

Pour *<language name>*, veuillez vous référer à la section sur PJLlang. Lorsque *<language name>* n’est pas spécifié, le nom sera défini pour toutes les langues prises en charge. De plus, les environnements avec ou sans astérisque partageant le même nom, donc, `\NameTheorem{envname*}{...}` a le même effet que `\NameTheorem{envname}{...}`.



Ensuite, créez cet environnement de l'une des cinq manières suivantes :

- `\CreateTheorem*{<name of environment>}`
  - Définir un environnement non numéroté *<name of environment>*
- `\CreateTheorem{<name of environment>}`
  - Définir un environnement numéroté *<name of environment>*, numéroté dans l'ordre 1, 2, 3, ...
- `\CreateTheorem{<name of environment>}[<numbered like>]`
  - Définir un environnement numéroté *<name of environment>*, qui partage le compteur *<numbered like>*
- `\CreateTheorem{<name of environment><numbered within>}`
  - Définir un environnement numéroté *<name of environment>*, numéroté dans le compteur *<numbered within>*
- `\CreateTheorem{<name of environment>}<existed environment>`  
`\CreateTheorem*{<name of environment>}<existed environment>`
  - Identifiez *<name of environment>* avec *<existed environment>* ou *<existed environment>\**.
  - Cette méthode est généralement utile dans les deux situations suivantes :
    1. Pour utiliser un nom plus concis. Par exemple, avec `\CreateTheorem{thm}(theorem)`, on peut alors utiliser le nom `thm` pour écrire le théorème.
    2. Pour supprimer la numérotation de certains environnements. Par exemple, on peut supprimer la numérotation de l'environnement `remark` avec `\CreateTheorem{remark}(remark*)`.

#### ASTUCE

Cette macro utilise la fonctionnalité de `amsthm` en interne, donc le traditionnel `theoremstyle` lui est également applicable. Il suffit de déclarer le style avant les définitions pertinentes.

Voici un exemple. Le code suivant :

```
\NameTheorem[FR]{proofidea}{Idée}
\CreateTheorem*{proofidea*}
\CreateTheorem{proofidea}<subsection>
```

définit un environnement non numéroté `proofidea*` et un environnement numéroté `proofidea` (numérotés dans la sous-section) respectivement. Ils peuvent être utilisés dans le contexte français. L'effet est le suivant (le style réel est lié à votre classe de document) :

**Idée** | La environnement `proofidea*`. ☐

**Idée 4.9.1** | La environnement `proofidea`. ☐

## 5 PROBLÈMES CONNUS

- PJLauthor est encore à son stade préliminaire, son effet n'est pas aussi bon que le authblk qui est relativement mature.
- PJLlang : C'est encore assez problématique avec la configuration de polyglossia, donc les fonctionnalités principales sont implémentées via babel pour le moment.
- PJLlang : Il y a quelques problèmes avec les options de langue. Par exemple, chinese provoquera des erreurs avec babel. D'autre part, des conflits entre plusieurs options peuvent survenir.
- PJLpaper : l'option preview est principalement implémentée à l'aide du package geometry, elle ne fonctionne donc pas aussi bien dans les classes de documents KOMA.
- PJLthm : les paramètres de numérotation et de style théorème des environnements de type théorème ne sont actuellement pas accessibles à l'utilisateur.
- PJLthm : la localisation de cleveref n'est pas encore finie pour toutes les langues prises en charge par PJLlang, en particulier pour le chinois, le japonais et le russe.
- Le mécanisme de gestion des erreurs est incomplet : pas de messages correspondants lorsque certains problèmes surviennent.
- Il y a encore beaucoup de choses qui peuvent être optimisées dans le code. Certains codes prennent trop de temps à s'exécuter, en particulier la configuration d'environnements de type théorème dans PJLthm.