

L^AT_EX 新手上路指南

🌀 AlphaZTX | C_TE_X-org

2022 年 5 月

L_AT_EX 与 T_EX

- L^AT_EX

- L^AT_EX
 - “a software system for document preparation”

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害

L^AT_EX 与 T_EX

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害
 - 基于 T_EX

L^AT_EX 与 T_EX

- L^AT_EX

- “a software system for document preparation”
- 写论文/提交作业要用的
- 排版公式很厉害
- 基于 T_EX

L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages

L^AT_EX 与 T_EX

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害
 - 基于 T_EX
 - L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages
- T_EX

L^AT_EX 与 T_EX

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害
 - 基于 T_EX
 - L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages
- T_EX
 - “a typesetting system which was designed and written by Donald Knuth and first released in 1978”

L^AT_EX 与 T_EX

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害
 - 基于 T_EX
 - L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages
- T_EX
 - “a typesetting system which was designed and written by Donald Knuth and first released in 1978”
 - 真正负责排版的東西

L^AT_EX 与 T_EX

- L^AT_EX
 - “a software system for document preparation”
 - 写论文/提交作业要用的
 - 排版公式很厉害
 - 基于 T_EX
 - L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages
- T_EX
 - “a typesetting system which was designed and written by Donald Knuth and first released in 1978”
 - 真正负责排版的東西——排版引擎

L^AT_EX 与 T_EX

- L^AT_EX

- “a software system for document preparation”
- 写论文/提交作业要用的
- 排版公式很厉害
- 基于 T_EX

L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages

- T_EX

- “a typesetting system which was designed and written by Donald Knuth and first released in 1978”
- 真正负责排版的東西——排版引擎
- 不只是 T_EX: pdfT_EX, ϵ -T_EX, X_YT_EX, LuaT_EX, pT_EX, ...

L^AT_EX 与 T_EX

- L^AT_EX

- “a software system for document preparation”
- 写论文/提交作业要用的
- 排版公式很厉害
- 基于 T_EX

L^AT_EX \approx T_EX + L^AT_EX kernel + document classes + packages

- T_EX

- “a typesetting system which was designed and written by Donald Knuth and first released in 1978”
- 真正负责排版的東西——排版引擎
- 不只是 T_EX: pdfT_EX, ϵ -T_EX, X_YT_EX, LuaT_EX, pT_EX, ...
- 目前最流行的中文支持: X_YT_EX

下载和安装 T_EX Live

「L^AT_EX 在哪儿下载？」

「L^AT_EX 在哪儿下载？」

解构：

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”
- 支持不同的格式，例如：plain T_EX, L^AT_EX, ConT_EXt

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”
- 支持不同的格式，例如：plain T_EX, L^AT_EX, ConT_EXt

应该安装哪一个 T_EX 发行版？

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”
- 支持不同的格式，例如：plain T_EX, L^AT_EX, ConT_EXt

应该安装哪一个 T_EX 发行版？

- 推荐 T_EX Live（跨平台，每年稳定更新）

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”
- 支持不同的格式，例如：plain T_EX, L^AT_EX, ConT_EXt

应该安装哪一个 T_EX 发行版？

- 推荐 T_EX Live（跨平台，每年稳定更新）
- macOS 用户也可以考虑 MacT_EX 

「L^AT_EX 在哪儿下载？」

解构：

- 你要下载的并不是一个叫「L^AT_EX」的软件
- 你要下载的东西其实叫做「T_EX 发行版」

什么是「T_EX 发行版」？

- T_EX distribution——套装
- 引擎、宏包、文档类的“全家桶”
- 支持不同的格式，例如：plain T_EX, L^AT_EX, ConT_EXt

应该安装哪一个 T_EX 发行版？

- 推荐 T_EX Live（跨平台，每年稳定更新）
- macOS 用户也可以考虑 MacT_EX 
- **坚决不推荐 C_T_EX 套装!!!**

安装 T_EX Live

安装 T_EX Live

- 通过网络安装: 

安装 T_EX Live

- 通过网络安装: 
- Windows: `install-tl-windows.exe`
- *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`

安装 T_EX Live

- 通过网络安装: 
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`
- 速度太慢, 不推荐

安装 T_EX Live

- 通过网络安装：[🔗](#)
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`速度太慢，不推荐
- 通过 ISO 镜像安装：
清华 [tuna](#) [🔗](#)，中国科大镜像站 [🔗](#)，其他 CTAN 镜像 [🔗](#)

安装 T_EX Live

- 通过网络安装：[🔗](#)
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`速度太慢，不推荐
- 通过 ISO 镜像安装：
清华 [tuna](#) [🔗](#)，中国科大镜像站 [🔗](#)，其他 CTAN 镜像 [🔗](#)
下载 `texlive<年份>.iso`，例如 `texlive2022.iso`

安装 T_EX Live

- 通过网络安装：[🔗](#)
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`速度太慢，不推荐
- 通过 ISO 镜像安装：
清华 [tuna](#) [🔗](#)，中国科大镜像站 [🔗](#)，其他 CTAN 镜像 [🔗](#)
下载 `texlive<年份>.iso`，例如 `texlive2022.iso`
 - Windows: 双击 `install-tl-windows.bat`
 - *nix: 终端输入 `./install-tl --gui`

安装 T_EX Live

- 通过网络安装: 
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`

速度太慢, 不推荐

- 通过 ISO 镜像安装:
清华 tuna , 中国科大镜像站 , 其他 CTAN 镜像 
下载 `texlive<年份>.iso`, 例如 `texlive2022.iso`
 - Windows: 双击 `install-tl-windows.bat`
 - *nix: 终端输入 `./install-tl --gui`

速度较快

安装 T_EX Live

- 通过网络安装: [🔗](#)
 - Windows: `install-tl-windows.exe`
 - *nix (Unix, Linux, macOS): `install-tl-unx.tar.gz`

速度太慢, 不推荐

- 通过 ISO 镜像安装:
清华 tuna [🔗](#), 中国科大镜像站 [🔗](#), 其他 CTAN 镜像 [🔗](#)
下载 `texlive<年份>.iso`, 例如 `texlive2022.iso`
 - Windows: 双击 `install-tl-windows.bat`
 - *nix: 终端输入 `./install-tl --gui`

速度较快

可以在安装界面配置环境变量 [🔗](#)

TeX Live 太大了，能不能不安装？

TeX Live 太大了，能不能不安装？

能

T_EX Live 太大了，能不能不安装？

能

在线 T_EX: Overleaf , T_EXPage 

T_EX Live 太大了，能不能不安装？

能

在线 T_EX: Overleaf , T_EXPage 

不用花时间下载和安装 T_EX Live, 但是也少了 `texdoc`

T_EX Live 太大了，能不能不安装？

能

在线 T_EX: Overleaf [🔗](#), T_EXPage [🔗](#)

不用花时间下载和安装 T_EX Live, 但是也少了 `texdoc`
没关系, 我们还有在线的 `texdoc`: [🔗](#)

texdoc

不允许你不知道的好东西: `texdoc`

不允许你不知道的好东西: texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

不允许你不知道的好东西: `texdoc`

`texdoc` 是 TeX Live 提供的一个命令行程序

- 需要看什么, 就直接用 `texdoc` 查什么

不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档

不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找

不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找
- texdoc lshort-zh-cn



不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找
- texdoc lshort-zh-cn
得到《一份（不太）简短的 L^AT_EX 2_ε 介绍》



不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找
- texdoc lshort-zh-cn 
得到《一份（不太）简短的 L^AT_EX 2_ε 介绍》——**看完它!!!**

不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找
- texdoc lshort-zh-cn 
得到《一份（不太）简短的 L^AT_EX 2_ε 介绍》——**看完它!!!**
- 觉得 lshort 不够详细?

不允许你不知道的好东西：texdoc

texdoc 是 T_EX Live 提供的一个命令行程序

- 需要看什么，就直接用 texdoc 查什么
- 打开命令行/终端，输入
texdoc <文件名>
即可查看对应的手册或文档，支持模糊查找
- texdoc lshort-zh-cn 
得到《一份（不太）简短的 L^AT_EX 2_ε 介绍》——**看完它!!!**
- 觉得 lshort 不够详细？——等我的书

准备文档

How to start?

How to start?

1. 在你需要的路径下新建一个 `.tex` 文件;

How to start?

1. 在你需要的路径下新建一个 `.tex` 文件;
2. 用文本编辑器打开它; `TEXworks`, `TEXstudio`, `VS Code`, ...

How to start?

1. 在你需要的路径下新建一个 `.tex` 文件;
2. 用文本编辑器打开它; T_E^X works, T_E^X studio, VS Code, ...
3. 开始编辑;

How to start?

1. 在你需要的路径下新建一个 `.tex` 文件;
2. 用文本编辑器打开它; $\text{T}_\text{E}\text{X}$ works, $\text{T}_\text{E}\text{X}$ studio, VS Code, ...
3. 开始编辑;
4. 编辑好了, 就可以编译了。

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}
```

```
\usepackage{amsmath}
```

```
\title{假装是一个标题}
```

```
\author{张三}
```

```
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}  
\usepackage{amsmath}
```

`\documentclass{ctexart}` 指定文档类为
`ctexart`;

```
\title{假装是一个标题}  
\author{张三}  
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}
```

```
\usepackage{amsmath}
```

```
\title{假装是一个标题}
```

```
\author{张三}
```

```
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

`\documentclass{ctexart}` 指定文档类为 `ctexart`;

`\usepackage{amsmath}` 使用 `amsmath` 包, 这是一个数学公式相关的使用频率很高的宏包;

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}
```

```
\usepackage{amsmath}
```

```
\title{假装是一个标题}
```

```
\author{张三}
```

```
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

`\documentclass{ctexart}` 指定文档类为 `ctexart`;

`\usepackage{amsmath}` 使用 `amsmath` 包, 这是一个数学公式相关的使用频率很高的宏包;

`\title`、`\author`、`\date` 分别指定标题、作者、日期。其中, `\today` 为文档编译时所在的日期。如果不想标注日期的话, 可以使用 `\date{}`;

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}
```

```
\usepackage{amsmath}
```

```
\title{假装是一个标题}
```

```
\author{张三}
```

```
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

`\documentclass{ctexart}` 指定文档类为 `ctexart`;

`\usepackage{amsmath}` 使用 `amsmath` 包，这是一个数学公式相关的使用频率很高的宏包；

`\title`、`\author`、`\date` 分别指定标题、作者、日期。其中，`\today` 为文档编译时所在的日期。如果不想标注日期的话，可以使用

`\date{}`;

在 `\begin{document}` 和 `\end{document}` 之间的称为正文，`\begin{document}` 之前的称为导言区。导言区禁止写入任何正文中的语句。

1 分钟速成 L^AT_EX (才怪)

```
\documentclass{ctexart}
```

```
\usepackage{amsmath}
```

```
\title{假装是一个标题}
```

```
\author{张三}
```

```
\date{\today}
```

```
\begin{document}
```

```
\maketitle
```

极其敷衍的正文。

```
\[ E = m c^2. \]
```

```
\end{document}
```

`\documentclass{ctexart}` 指定文档类为 `ctexart`;

`\usepackage{amsmath}` 使用 `amsmath` 包, 这是一个数学公式相关的使用频率很高的宏包;

`\title`、`\author`、`\date` 分别指定标题、作者、日期。其中, `\today` 为文档编译时所在的日期。如果不想标注日期的话, 可以使用

```
\date{}
```

在 `\begin{document}` 和 `\end{document}` 之间的称为正文, `\begin{document}` 之前的称为导言区。导言区禁止写入任何正文中的语句。

`\maketitle` 也就是在当前位置生成带有格式的标题。

1 分钟速成 L^AT_EX (才怪)

`\documentclass{ctexart}`

`\usepackage{amsmath}`

`\title{假装是一个标题}`

`\author{张三}`

`\date{\today}`

`\begin{document}`

`\maketitle`

极其敷衍的正文。

`\[E = m c^2. \]`

`\end{document}`

`\documentclass{ctexart}` 指定文档类为 ctexart;

`\usepackage{amsmath}` 使用 amsmath 包, 这是一个数学公式相关的使用频率很高的宏包;

`\title`、`\author`、`\date` 分别指定标题、作者、日期。其中, `\today` 为文档编译时所在的日期。如果不想标注日期的话, 可以使用

`\date{}`;

在 `\begin{document}` 和 `\end{document}` 之间的称为正文, `\begin{document}` 之前的称为导言区。导言区禁止写入任何正文中的语句。

`\maketitle` 也就是在当前位置生成带有格式的标题。

看完 `\short-zh-cn!!!`

把 T_EX 变成 PDF——编译

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

命令行编译

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

命令行编译

1. 在文件所在的路径下打开命令行；

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

命令行编译

1. 在文件所在的路径下打开命令行；
2. 输入：〈编译命令〉 〈文件名〉

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

命令行编译

1. 在文件所在的路径下打开命令行；
2. 输入：〈编译命令〉 〈文件名〉

编译命令：tex, latex, pdftex, pdf_latex, xetex, xelatex, luatex, lua_latex, ...

把 T_EX 变成 PDF——编译

- 编辑器提供的编译按钮
 - 傻瓜式，简单易上手
 - 通常需要自己预先配置
 - 功能可能会比较单一
- 命令行编译
 - 编辑器的按钮本质上是使用命令行编译
 - 支持更多编译模式
 - More debug-friendly

命令行编译

1. 在文件所在的路径下打开命令行；
2. 输入：〈编译命令〉 〈文件名〉

编译命令：tex, latex, pdftex, pdflatex, xetex, xelatex, luatex, lualatex, ...

举个例子：xelatex document.tex

搞定中文文档

搞定中文文档

- 字符集超大

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_LA_TE_X
- 需要对于汉字输出、标点压缩等方面的支持

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe₂LaTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_{La}TeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_YLaTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_YLaTeX 编译

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_YLaTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_YLaTeX 编译 (不是 CTeX 套装)

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好： $\text{X}_{\text{L}}\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + $\text{X}_{\text{L}}\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 编译 (不是 $\text{C}_{\text{T}}\text{E}_{\text{X}}$ 套装)

不推荐的解决方案：

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_ΛTeX 编译 (不是 CT_EX 套装)

不推荐的解决方案：

- ctex 宏集 + 其他引擎编译

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_ΛTeX 编译 (不是 CT_EX 套装)

不推荐的解决方案：

- ctex 宏集 + 其他引擎编译
- 裸用 xeCJK 宏包

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_ΛTeX 编译 (不是 CT_EX 套装)

不推荐的解决方案：

- ctex 宏集 + 其他引擎编译
- 裸用 xeCJK 宏包 (高手除外)

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_ΛTeX 编译 (不是 CT_EX 套装)

不推荐的解决方案：

- ctex 宏集 + 其他引擎编译
- 裸用 xeCJK 宏包 (高手除外)

过时的解决方案：

搞定中文文档

- 字符集超大——原生支持 Unicode 的引擎更好：Xe_ΛTeX
- 需要对于汉字输出、标点压缩等方面的支持：ctex 宏集

推荐的解决方案：

- 使用 ctex 宏集 + Xe_ΛTeX 编译 (不是 CT_EX 套装)

不推荐的解决方案：

- ctex 宏集 + 其他引擎编译
- 裸用 xeCJK 宏包 (高手除外)

过时的解决方案：

- CJK 宏包 (别用!)

编辑公式

编辑公式

- 公式都是在数学环境中完成的

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来： $\$ \cdots \$$

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来： $\$ \cdots \$$
- 行间公式

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来： $\$ \cdots \$$
- 行间公式
 - 不编号： $\backslash[\cdots\backslash]$

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来： $\$ \cdots \$$
- 行间公式
 - 不编号： $\backslash[\cdots\backslash]$
 - 编号：equation 环境，可以加标签

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来: $\$ \cdots \$$
- 行间公式
 - 不编号: $\backslash[\cdots\backslash]$
 - 编号: `equation` 环境, 可以加标签
- `amsmath` 宏包几乎每次都要使用

编辑公式

- 公式都是在数学环境中完成的
- 数学环境不受外界字体命令控制
- 行内公式
 - 用两个美元符号括起来: $\$ \cdots \$$
- 行间公式
 - 不编号: $\backslash[\cdots\backslash]$
 - 编号: `equation` 环境, 可以加标签
- `amsmath` 宏包几乎每次都要使用
- **看 `lshort-zh-cn` 第四章!**

遭遇问题

我怎么知道我该用什么包、该查什么文档？

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索
 - 彻底搜索过了，还是没有答案

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索
 - 彻底搜索过了，还是没有答案——在社区提问 

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索
 - 彻底搜索过了，还是没有答案——在社区提问 
不要匿名提问！截图而不是拍照！
 - 遇到问题给出可以复现问题的最小工作样例（MWE） 

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索
 - 彻底搜索过了，还是没有答案——在社区提问 
不要匿名提问！截图而不是拍照！
遇到问题给出可以复现问题的最小工作样例（MWE） 
5. 有些问题是无解的，或者问题本身就是错误的。

我怎么知道我该用什么包、该查什么文档？

1. 看完 `lshort-zh-cn`;
2. 如果你做到了第 1 点，且对常用的宏包有了基本的认识，则往下看；否则回到第 1 点；
3. 如果 `lshort-zh-cn` 里面提供的宏包可以解决你的问题，就直接 `texdoc` 它；否则继续往下看；
4. 向互联网寻求答案：
 - 你的问题很有可能是别人以前就遇到过的——善用搜索
 - 彻底搜索过了，还是没有答案——在社区提问 
不要匿名提问！截图而不是拍照！
遇到问题给出可以复现问题的最小工作样例（MWE） 
5. 有些问题是无解的，或者问题本身就是错误的。
例如：如何在数学环境中使用 Times New Roman？如何让浮动体在指定的位置固定不动？

×× 符号怎么输入？

×× 符号怎么输入?

“常规”的符号, 比如: $\mathcal{L}, \mathbb{R}, \overline{\lim}, \ll, \dot{=}, \dots$

×× 符号怎么输入?

“常规”的符号, 比如: \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

×× 符号怎么输入?

“常规”的符号, 比如: $\mathcal{L}, \mathbb{R}, \overline{\lim}, \ll, \dot{=}, \dots$

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有, 比如: $\mathfrak{f}, \mathfrak{f}, \dots$

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \oint , \oint , ...

- 先不要去互联网提问

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \oint , \oint , ...

- 先不要去互联网提问
- `texdoc comprehensive`

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , \doteq , ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \mathfrak{f} , \mathfrak{f} , ...

- 先不要去互联网提问
- `texdoc comprehensive`

`texdoc comprehensive` 要是还没有……

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \oint , \oint , ...

- 先不要去互联网提问
- `texdoc comprehensive`

`texdoc comprehensive` 要是还没有……

- 恭喜你，可以去互联网提问了！

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \mathbb{f} , \mathbb{f} , ...

- 先不要去互联网提问
- `texdoc comprehensive`

`texdoc comprehensive` 要是还没有……

- 恭喜你，可以去互联网提问了！
- 使用一些「dirty tricks」拼接符号 

×× 符号怎么输入？

“常规”的符号，比如： \mathcal{L} , \mathbb{R} , $\overline{\lim}$, \ll , $\dot{=}$, ...

- 看 `lshort-zh-cn!!!`

`lshort` 里面也没有，比如： \mathfrak{f} , \mathfrak{f} , ...

- 先不要去互联网提问
- `texdoc comprehensive`

`texdoc comprehensive` 要是还没有……

- 恭喜你，可以去互联网提问了！
- 使用一些「dirty tricks」拼接符号 
- 甚至可以自己做一个字体 

Thank you!

Special thanks to  stone-zeng